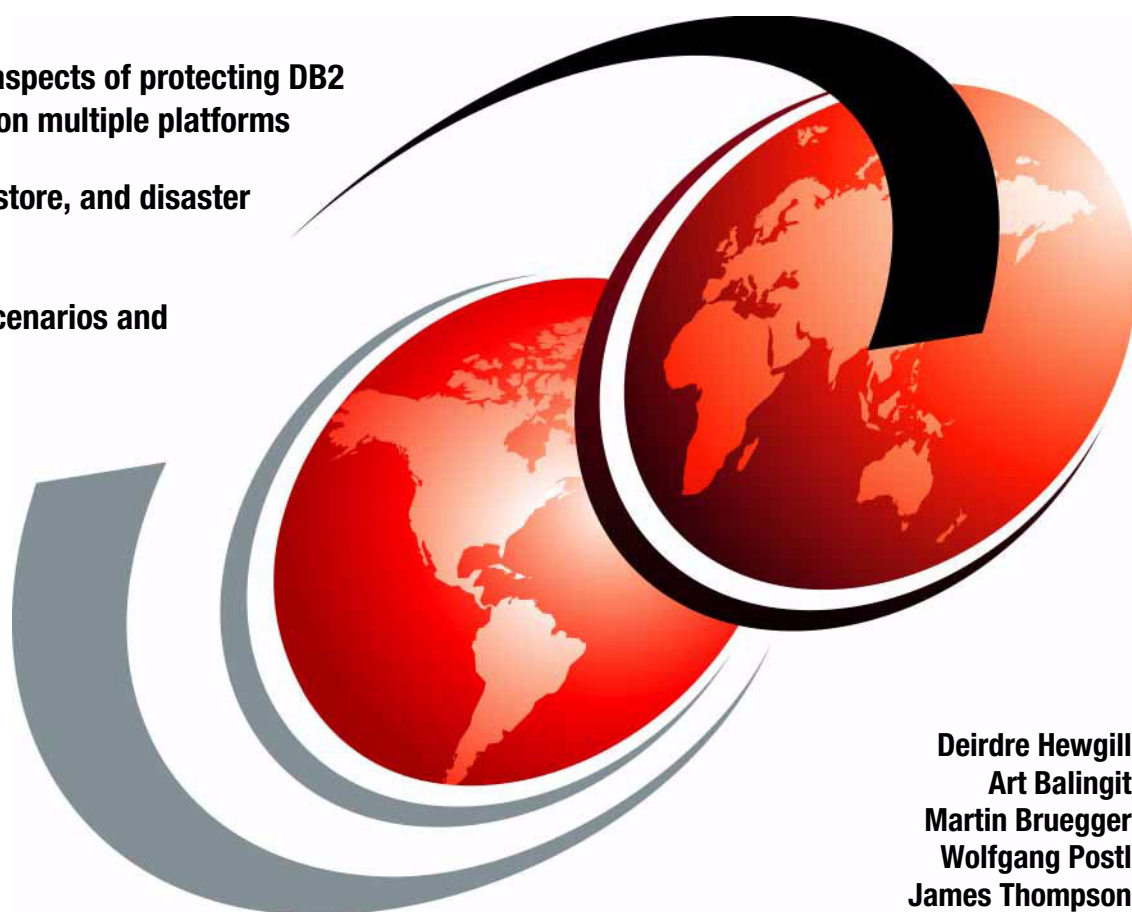


Backing up DB2 with IBM Tivoli Storage Management

Covers all aspects of protecting DB2
databases on multiple platforms

Backup, restore, and disaster
recovery

Practical scenarios and
how-tos



Deirdre Hewgill
Art Balingit
Martin Bruegger
Wolfgang Postl
James Thompson

ibm.com/redbooks

Redbooks



International Technical Support Organization

Backing up DB2 with IBM Tivoli Storage Management

Take Note!

Before using this information and the product it supports, be sure to read the general information in "Notices" on page 321.

First Edition (May 2001)

This edition applies to Version 4, Release 1 of Tivoli Storage Manager, Program Number 5698-TSM and Version 7, Release1 of IBM DB2 UDB, Program Number 5648-D48, for use with the IBM AIX, Sun Solaris,HP-UX and Microsoft Windows NT and Windows 2000 operating systems.

Comments may be addressed to:

IBM Corporation, International Technical Support Organization
Dept. QXXE Building 80-E2
650 Harry Road
San Jose, California 95120-6099

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 2001. All rights reserved.

Note to U.S Government Users – Documentation related to restricted rights – Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

Contents

Figures	11
Tables	17
Preface	19
The team that wrote this redbook	19
Comments welcome	21
<hr/>	
Part 1. Tivoli Storage Manager and DB2 primer	1
Chapter 1. Tivoli Storage Manager for database administrators	3
1.1 Introducing Tivoli Storage Management	3
1.1.1 Tivoli Storage Manager Backup-Archive client	5
1.1.2 Tivoli Storage Manager API	5
Chapter 2. Relational databases and DB2 UDB product overview	7
2.1 Fundamentals of Relational Database Management Systems	7
2.1.1 Database	7
2.1.2 Tables	7
2.1.3 Indexes	8
2.1.4 Tablespaces	9
2.1.5 Partitioning options	11
2.1.6 Log files	11
2.1.7 Control files	13
2.1.8 Configuration parameters	13
2.2 DB2 UDB concepts	13
2.2.1 DB2 UDB products	14
2.2.2 Instance	16
2.2.3 Database and database partitions	16
2.2.4 Nodegroups	17
2.2.5 Tablespaces	17
2.2.6 Recovery logs	18
2.2.7 Recovery history file	19
Chapter 3. Planning considerations	21
3.1 Backup requirements	21
3.1.1 Types of events	22
3.1.2 Speed of recovery	24
3.1.3 Backup windows	24
3.1.4 Recovery points	25
3.1.5 Units of recovery	25

3.1.6 Backup of RDBMS supporting files	25
3.2 Backup techniques	26
3.2.1 Disk mirroring	26
3.2.2 Offline backup	27
3.2.3 Online backup	27
3.2.4 Database export	28
3.2.5 Full database backup	28
3.2.6 Partial database backup	29
3.2.7 Incremental backup	29
3.2.8 Log file backup (simulated incremental)	30
3.2.9 LAN-free backup	30
3.2.10 Backup using split mirror features.	31
Chapter 4. Tivoli Storage Manager server considerations.	33
4.1 Initial requirements	33
4.2 Tivoli Storage Manager server base functionality explained	33
4.2.1 Registering a node with the Tivoli Storage Manager server	34
4.2.2 Installing the Tivoli Storage Manager client.	34
4.2.3 Configure the Tivoli Storage Manager client	35
4.3 Tivoli Storage Manager data objects	35
4.3.1 Archive or backup object	36
4.3.2 Managing Tivoli Storage Manager data objects	36
4.3.3 Management class to which to bind object	39
4.3.4 Life cycle of Tivoli Storage Manager data objects	43
4.4 Tivoli Storage Manager server considerations for UDB DB2 backups.	46
4.4.1 How UDB DB2 stores data objects	47
4.5 Policy management considerations	47
4.5.1 Domain considerations	47
4.5.2 Tivoli Storage Manager management class considerations.	48
4.5.3 Tivoli Storage Manager client include-exclude option	50
4.6 Node considerations.	53
4.6.1 Choosing a nodename	53
4.6.2 Choosing a password	54
4.6.3 Setting the BACKDELETE option	54
4.6.4 Setting the ARCHDELETE option	54
4.6.5 Specifying the domain	54
4.6.6 Setting the MAXNUMMP	54
4.7 Storage pool considerations	55
4.8 Our Tivoli Storage Manager server setup	56

Part 2. Backing up DB2 on UNIX platforms	59
Chapter 5. Backing up DB2 on AIX using Tivoli Storage Manager . . .	61
5.1 System requirements	61
5.2 Installation of Tivoli Storage Manager related software.	61
5.2.1 Verification of Tivoli Storage Manager client API installation.	62
5.3 Configuration and setup of Tivoli Storage Manager client API	63
5.3.1 API client setup	64
5.4 Setting up the DB2 user exit for Tivoli Storage Manager.	67
5.4.1 Compile the user exit for Tivoli Storage Manager	68
5.4.2 Enable the database for roll-forward recovery.	69
5.5 Optional DB2 configurations.	69
5.6 Using the Tivoli Storage Manager baclient in conjunction with DB2 . . .	71
5.6.1 Using the baclient include/exclude file	72
5.6.2 Using the archive utility.	72
5.7 Backing up DB2 using Tivoli Storage Manager	73
5.7.1 Full offline backup	73
5.7.2 Online database backup	78
5.7.3 Tablespace backup	83
5.7.4 Load utility	88
Chapter 6. Backing up DB2 UDB on the Sun Solaris platform.	91
6.1 Installation of Tivoli Storage Manager related software.	91
6.1.1 System requirements	92
6.1.2 Installing the Tivoli Storage Manager client.	92
6.2 Configuration and setup of Tivoli Storage Manager client API	94
6.2.1 Setup prerequisites	94
6.2.2 Define the environment variables	94
6.2.3 Create the Tivoli Storage Manager configuration files.	95
6.3 Setting up the DB2 user exit for Tivoli Storage Manager.	97
6.3.1 Compile the user exit for Tivoli Storage Manager	97
6.3.2 Enable the database for roll-forward recovery.	98
6.4 Using Tivoli Storage Manager baclient in conjunction with DB2	99
6.5 Backing up DB2 using Tivoli Storage Manager	100
6.5.1 Full offline backup	101
6.5.2 Online database backup	107
6.5.3 Tablespace backup	111
6.5.4 Load utility	116
Chapter 7. Day to day management: DB2 backups on UNIX	119
7.1 The db2adutl utility	119
7.2 Information about backups using Tivoli Storage Manager commands	121
7.3 Verification of DB2 backups	124

7.3.1	Verify backup using db2ckbcp	124
7.3.2	Verification using DB2 list history	125
7.3.3	Verification using db2adutl	126
7.3.4	Sample script for backup and verification	128
7.3.5	Automatic notification for user exit failure	129
7.4	Deletion of backups and logfiles	131
7.4.1	Deletion of backup objects	131
7.4.2	Deletion of logfiles	131
7.4.3	Automated deletion of backups and logfiles	133
7.5	Automation of DB2 backup	134
7.5.1	Automate DB2 backup using cron	134
7.5.2	Automate DB2 backup using DB2	135
7.5.3	Automate DB2 backup using Tivoli Storage Manager	141
7.6	Maintaining the history file	145

Part 3. Backing up DB2 on the Windows 2000 platform 147

Chapter 8. Backing up DB2 on Windows 2000	149
8.1 Registering a node on the TSM server for DB2 backups	149
8.2 Downloading latest Tivoli Storage Manager baclient and API	150
8.3 Installation	151
8.4 Configuring the API	156
8.5 Configuring the client options file	161
8.6 Generating the encrypted password	163
8.7 Setting up the DB2 user exit for Tivoli Storage Manager	166
8.7.1 Modifying the user exit	166
8.8 DB2 Tivoli Storage Manager configuration values	169
8.9 Backing up DB2 using Tivoli Storage Manager	170
8.9.1 Full offline backup	171
8.9.2 Full online backup	175
8.9.3 Tablespace backup	178
Chapter 9. Day to day management: DB2 backups on Windows 2000	183
9.1 The db2adutl utility	183
9.2 Information about backups using TSM commands	185
9.3 Verification of DB2 backups	188
9.3.1 Verify backup using db2ckbcp	188
9.3.2 Verification using DB2 list history	189
9.3.3 Verification using db2adutl	190
9.4 Deletion of backups and logfiles	193
9.4.1 Deletion of backups	193
9.4.2 Deletion of logfiles	193
9.4.3 Auto deletion of backups and logfiles	194

9.5 Automation of DB2 backup	195
9.5.1 Automate DB2 backup using Windows	195
9.5.2 Automate DB2 backup using DB2	196
9.5.3 Automate DB2 backup using Tivoli Storage Manager	196
9.5.4 Automate DB2 backup using a Windows service starting a script	199
9.6 Maintaining the history file	208

Part 4. Recovering DB2 UDB databases using TSM 211

Chapter 10. Recovering DB2 UDB databases	213
10.1 Version recovery	213
10.1.1 Version recovery example	214
10.1.2 Version recovery using the command line	214
10.1.3 Version recovery using the Control Center	215
10.2 Database roll-forward recovery	218
10.2.1 Database roll-forward recovery example	218
10.2.2 Database roll-forward recovery using the command line	218
10.2.3 Database roll-forward recovery using the Control Center	220
10.3 Tablespace roll-forward recovery	224
10.3.1 Tablespace roll-forward recovery example	224
10.3.2 Tablespace roll-forward recovery using the command line	225
10.3.3 Tablespace roll-forward recovery using the Control Center	227
10.4 Point in time recovery	235
10.4.1 Point in time recovery concepts and lab experience	235
10.4.2 Point in time recovery considerations	236
10.4.3 Point in time recovery example	237
10.4.4 Point in time recovery using the command line	238
10.4.5 Point in time recovery using the Control Center	240
10.5 Restoring a DB2 database to a new DB2 instance	246
10.5.1 Restore using db2adutl	246
10.5.2 Restore using same Tivoli Storage Manager client setup	246
10.5.3 Restore using DB2 parameter	247
10.6 Tablespace redirected restore	254
10.6.1 Tablespace redirected restore example	254
10.6.2 Tablespace redirected restore using the command line	255
10.6.3 Tablespace redirected restore using the Control Center	257
10.7 Recovering the history file	265
10.8 Additional notes on recovery	266
10.8.1 Roll-forward status	266
10.8.2 Restarting restore and roll-forward	266
10.8.3 Using more than one backup image for a recovery	267
10.8.4 Restore without rolling forward	268

Appendix A. Quick start/checklist for configuration	271
A.1 Windows quick start	271
A.2 AIX quick start	272
A.3 Sun Solaris quick start	273
Appendix B. Troubleshooting	277
B.1 Gotchas	277
B.1.1 RC 406 options file not found	277
B.1.2 Running a Tivoli Storage Manager CONFIG trace	278
B.1.3 RC 137 authentication failure, incorrect password	279
B.1.4 DB2 User exit on Windows NT cannot find .h (header) files	280
B.2 Checklist for the Tivoli Storage Manager server	280
B.3 Isolating the problem	280
B.4 List of logfiles	281
B.4.1 User exit return codes	283
Appendix C. Performance	285
C.1 General performance considerations	285
C.2 DB2 backup command performance options	286
C.3 DB2 restore command performance options	288
Appendix D. Split mirror and split copy functions with DB2	289
D.1 Using split mirror features in conjunction with DB2	291
D.1.1 DB2 clone database	292
D.1.2 DB2 standby database	293
D.1.3 DB2 mirror database	294
D.2 Using AIX splitcopy feature for DB2 backup	296
Appendix E. “New” backup features (DB2 V7.1 Fixpak3 Beta)	301
E.1 Incremental backup	301
E.2 On demand log archive	305
Appendix F. DB2 backup using TSM LAN-free setup	307
F.1 Configuration for LAN-free	307
F.1.1 Define new managementclass for LAN-free backups	308
F.1.2 Download and install Tivoli Storage Manager Storage Agent	309
F.1.3 Modify dsmsta.opt	310
F.1.4 Check adsmcsi	310
F.1.5 Define drive mappings	312
F.1.6 Define server for storage agent on Tivoli Storage Manager server	313
F.1.7 Run dsmsta setstorageserver on storage agent	314
F.1.8 Install storage agent as a service	314
F.1.9 Specify enablelanfree in client options file (dsm.opt)	315
F.2 Running a backup and verifying that LAN-free is working	315

F.3 Problems	316
F.3.1 Problems with the userexit	316
F.3.2 Problems with the db2adutl.exe	319
Appendix G. Related publications	323
G.1 IBM Redbooks	323
G.2 IBM Redbooks collections	323
G.3 Other resources	323
G.4 Referenced Web sites	324
How to get IBM Redbooks	325
IBM Redbooks fax order form	326
Index	327
IBM Redbooks review	331

Figures

1. User interaction with tables	8
2. Using index in a table.	9
3. Tables, tablespaces, and data storage	10
4. Log files	12
5. Database partitioning	15
6. DB2 object hierarchy in an instance	16
7. Nodegroups	17
8. Native support or Tivoli Data Protection interfacing with the TSM API . . .	34
9. Archive object life cycle	44
10. Backup object life cycle	45
11. Sample include-exclude list file	52
12. Define domain, policy and management class	57
13. Define tape pools.	57
14. Define disk pools and assign some volumes.	57
15. Define copygroups.	58
16. Validate and activate policy	58
17. Define Tivoli Storage Manager node	58
18. Example of dsm.sys file	66
19. Use DB2 Tivoli Storage Manager database configuration parameter	71
20. Tivoli Storage Manager baclient include-exclude file example	72
21. Control Center	75
22. Attach to the administration server	76
23. Drop down menu for database.	77
24. Backup database with Tivoli Storage Manager	77
25. Backup successful dialog box	78
26. Control Center	79
27. Attach to the administration server	80
28. Drop down menu for database.	81
29. Backup database with Tivoli Storage Manager	82
30. Backup using the online option	82
31. Backup successful dialog box	83
32. Control Center	84
33. Attach to the administration server	85
34. Drop down menu for a tablespace.	86
35. Backup tablespace using Tivoli Storage Manager.	87
36. Backup tablespace using the online option	87
37. Backup successful dialog box	88
38. Tivoli Storage Manager Backup-Archive include-exclude file example . .	100
39. Control Center	103
40. Attach to the administration server	104

41. Drop down menu for database	105
42. Backup database with Tivoli Storage Manager	106
43. Backup successful dialog box	106
44. Control Center	108
45. Attach to the administration server	109
46. Drop down menu for database	110
47. Backup database with Tivoli Storage Manager	110
48. Backup using the online option	111
49. Backup successful dialog box	111
50. Control Center	113
51. Attach to the administration server	114
52. Drop down menu for a tablespace	115
53. Backup tablespace using Tivoli Storage Manager	116
54. Backup tablespace using the online option	116
55. Backup successful dialog box	117
56. Syntax of db2adutl	119
57. Output of Tivoli Storage Manager Select command	122
58. Example db2ckbkp	125
59. DB2 list history	125
60. Example db2adutl query	126
61. Example db2adutl extract	127
62. Example db2adutl verify	128
63. Sample backup and verification script	129
64. Example db2adutl delete logfiles	132
65. Example script of deletion of obsolete backups and logfiles	133
66. Sample backup shell script	135
67. Select backup database in DB2 Control Center	136
68. DB2 backup window	137
69. DB2 schedule window	137
70. DB2 journal	138
71. Script center	139
72. New command script	140
73. Scheduling a script	141
74. Example dsm.sys file for Tivoli Storage Manager client scheduler	142
75. Tivoli Storage Manager client version and release	150
76. Levels of Tivoli Storage Manager client code	151
77. Files available for latest level	151
78. Location to Save Files window	152
79. Language selection	152
80. InstallShield Wizard	153
81. Installation directory selection	153
82. Complete or custom	154
83. Selecting the API SDK and administrative client files	154

84. Installation confirmation window	155
85. Successful installation confirmation	155
86. Right-click My Computer icon	156
87. System Properties: General tab	156
88. System Properties: Advanced tab	157
89. Environment variables	158
90. Setting DSMI_CONFIG system variable	159
91. Setting the DSMI_DIR system variable	160
92. Setting the DSMI_LOG system variable	161
93. Verifying the system variables	161
94. Creating the client options file	162
95. Editing the client options file	163
96. Location of DB2 UDB executable dsmapipw.exe	163
97. Issuing set to confirm system variables	164
98. dsmapipw.exe	165
99. Warning messages that confirm configuration is correct	165
100. Control Center offline backup	173
101. Select Tivoli Storage Manager option in Control Center GUI	174
102. DB2 start backup window	174
103. End of DB2 offline backup message using the DB2 GUI	175
104. DB2 select database for online backup	176
105. Select Tivoli Storage Manager option in backup GUI	177
106. Select Online option in Options menu	177
107. DB2 start backup window	178
108. End of DB2 online backup message using the DB2 GUI	178
109. Select Tablespace Backup Menu	180
110. Select Tivoli Storage Manager option in backup GUI	180
111. Select Offline or Online option in options menu	181
112. DB2 start backup window	181
113. End of tablespace backup message using the DB2 GUI	182
114. Syntax of db2adutl	184
115. Output of Tivoli Storage Manager Select command	187
116. Example db2ckbkp with two backup images in the set	189
117. DB2 list history	190
118. Example db2adutl query	191
119. Example db2adutl extract	191
120. Example db2adutl verify	192
121. Example db2adutl delete logfiles	194
122. Entering the backup script in the registry	202
123. Set startup type to manual	203
124. Set Log On properties	204
125. Automated backup process flow	207
126. Container error in db2diag.log	214

127.	List history command result.	215
128.	Version recovery using the Control Center	216
129.	Backup image selection page	217
130.	Version recovery successful dialog box	217
131.	List history command result.	219
132.	Database roll-forward using the Control Center	221
133.	Backup image selection page	222
134.	Roll-forward page	223
135.	Roll-forward recovery successful dialog box	224
136.	List tablespaces.	225
137.	List history command result.	226
138.	Restore tablespace using the Control Center	228
139.	Warning dialog box	229
140.	Manually entering the backup image.	229
141.	Database drop down menu	230
142.	Backup image selection page	231
143.	Table Spaces page	232
144.	Roll forward page	233
145.	Options page	234
146.	Tablespace roll-forward recovery successful	234
147.	Point in time recovery	235
148.	Log files being reused after a point in time recovery.	236
149.	List tablespaces show detail result	238
150.	List history command result.	239
151.	Tablespace in backup-pending state.	240
152.	Tablespace point in time recovery.	241
153.	Backup image selection page	242
154.	Table Spaces page	243
155.	Roll-forward page	244
156.	Options page	245
157.	Point in time recovery successful	245
158.	List tablespace command	255
159.	List history command result.	256
160.	Tablespace redirected restore.	258
161.	Backup image selection page	259
162.	Table Spaces page	260
163.	Container page	261
164.	Change container dialog	262
165.	Roll-forward page	263
166.	Options page	264
167.	Tablespace redirected restore successful.	264
168.	Split mirror concept	289
169.	Implementation of split mirror	290

170.AIX LVM mirroring with three physical copies	297
171.Splitcopy read-only filesystem.	298
172.Incremental backup	302
173.Restore incremental backups	304
174.Showing hidden devices in Device Manager.	311
175.Opening Non-Plug and Play Drivers entry	311
176.AdsmScsi properties	312

Tables

1. Rollforward status	266
-----------------------------	-----

Preface

This IBM® Redbook discusses techniques and gives guidelines for backing up DB2® UDB using Tivoli® Storage Management products. It is intended for database administrators (DBAs) and system/storage administrators, and anyone who needs to protect their critical DB2 databases. We focus on the use of the Tivoli Storage Manager API client and provide installation, setup, customization and day-to-day management examples.

Tivoli Storage Manager is a full-function storage software product that addresses the challenges of complex storage management across distributed environments. It protects and manages a broad range of data, from the workstation to the corporate server environment. Tivoli Storage Manager provides:

- Centralized administration for data and storage management
- Efficient management of information growth
- Customized backup solutions for major groupware and database products

All testing was carried out on DB2 V7.1 and with Tivoli Storage Manager V4.1.2. We used these platforms: AIX®, Sun™ Solaris™ and Windows® 2000. The design of the project includes recovery scenarios, as well as, different backup methods that provide practical assistance for DBAs.

The team that wrote this redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization San Jose Center.

Deirdre Hewgill is a Tivoli Storage Manager Project Leader at the International Technical Support Organization, San Jose Center. Before joining the ITSO this year, she provided EMEA Level 2 Technical Support for Tivoli Storage Manager products. She has worked for IBM for the last 13 years primarily in support, but also in services roles with Enterprise customers.

Art Balingit is a DBA Technical Specialist with IBM in New Zealand, where he installs and supports Oracle and DB2 databases. He has 17 years of IT experience using a wide range of mainframe and client/server platforms. His experience includes four years of database design, implementation, administration, and support of Oracle and DB2 databases on UNIX® systems and Windows NT®. He holds a degree in Psychology from the Ateneo de

Manila University (Philippines). His areas of expertise include transaction processing systems, analysis and design, and programming.

Martin Bruegger is an Advisory IT Specialist in Switzerland, where he is working in the outsourcing department. He has more than 10 years of experience as a database administrator in a production environment. He has worked with IBM for 11 years. His areas of expertise include installation and support of Oracle databases on UNIX and S/390® servers.

Wolfgang Postl is an IT Specialist with IBM in Austria, where he installs and supports RS/6000® and SP systems. He has six years of experience with DB2 and RS/6000 products. Wolfgang holds a degree in Mathematics from the University of Graz (Austria). His areas of expertise include system management, storage servers and relational databases.

James Thompson is a Tivoli Storage Manager Level 2 Technical Support Engineer with IBM in Tucson, Arizona. He has seven years of IT experience across multiple platforms and applications. He graduated Cum Laude with a degree in Computer Science from Utah State University. He is currently a Technical Support Team Lead for Tivoli Storage Manager, Tivoli Data Protection and API products. His areas of expertise include Tivoli Storage Manager, Windows, Novell, AIX, DB2, Oracle, and Storage Area Networks.

Thanks to the following people for their invaluable contributions to this project:

Pat Randall
International Technical Support Organization, San Jose Center

Charlotte Brooks
International Technical Support Organization, San Jose Center

Corinne Baragoin
International Technical Support Organization, San Jose Center

Kathy Pang
Tivoli Systems, San Jose

David Godwin, Dale Mcinnis,
DB2 IBM, Toronto

Stjepan Cvitkovic
IBM Global Services, Germany

Comments welcome

Your comments are important to us!

We want our Redbooks™ to be as helpful as possible. Please send us your comments about this or other Redbooks in one of the following ways:

- Fax the evaluation form found in “IBM Redbooks review” on page 331 to the fax number shown on the form.
- Use the online evaluation form found at ibm.com/redbooks
- Send your comments in an Internet note to redbook@us.ibm.com

Part 1. Tivoli Storage Manager and DB2 primer

2 Backing Up DB2 Using IBM Tivoli Storage Manager

Chapter 1. Tivoli Storage Manager for database administrators

In this chapter, we give an overview of Tivoli Storage Manager, including the details of the various components. The details should assist database administrators who want to gain a basic understanding of Tivoli Storage Manager before configuring their system for database backup and recovery.

1.1 Introducing Tivoli Storage Management

Tivoli Storage Manager is the core application of the Tivoli Storage Management solution set. Tivoli Storage Manager is an enterprise-wide storage management application for the network. It provides automated storage management services (including backup and restore, archive and retrieve, hierarchical space management and disaster recovery) to multi-vendor workstations, personal computers, mobile laptops and servers of all sizes and operating systems, which are connected via WAN, LAN, and SAN. Tivoli Storage Manager includes these components:

- Server

The server provides backup, archive, and space management services to its defined clients. The server maintains its own database and recovery log for information about Tivoli Storage Manager resources, users, and user data including all backed-up, archived and migrated files. The client data itself is stored in server-controlled entities called storage pools. These are groups of random and sequential access media that store backed-up, archived, and space-managed files.

The Tivoli Storage Manager server is responsible for maintaining the integrity of client sessions, reliably receiving client data, storing client data in storage pools, and efficiently managing that data internally so that it can be restored or retrieved when required. You can set up multiple servers in your enterprise network to balance storage, processor, and network resources. Tivoli Storage Manager allows you to manage and control multiple servers from a single interface that runs in a Web browser (the enterprise console).

- Administrative interface

The administrative interface allows administrators to control and monitor server activities, define management policies for client files, and set up schedules to provide services at regular intervals. Administrative functions are available from an administrative client command line and from a Web browser interface. A server console is also available.

- Backup-Archive client

The Backup-Archive client allows users to maintain backup versions of their files, which they can restore if the original files are lost or damaged. Users can also archive files for long-term storage and retrieve the archived files when necessary. A command line interface, native GUI interface, and Web browser interface are available for the Backup-Archive clients.

- Application program interface (API)

The API allows users to enhance existing application programs with backup, archive, restore, and retrieve services. When users install the Tivoli Storage Manager API client on their clients, they can register as client nodes with a Tivoli Storage Manager server.

The Tivoli Storage Management solution set also includes the following client programs:

- Tivoli Data Protection (TDP) for applications (application clients)

This program allows users to perform online backups of data that is used by particular applications such as database programs. After the database initiates a backup or restore, the application client uses the API to interface to Tivoli Storage Manager. The Tivoli Storage Manager server then applies its storage management functions to the data. The application client can perform its functions while users are working, with minimal disruption. Tivoli Data Protection clients are available for Oracle, Informix®, SAP R/3, Lotus® Notes® R4, Lotus Domino® R5, MS Exchange and MS SQL Server.

- Tivoli Space Manager (hierarchical storage management client)

This program provides space management services for clients on some platforms. Tivoli Space Manager users can free client storage by migrating less frequently used files to server storage. These migrated files are also called space-managed files. Users can recall space-managed files automatically simply by accessing them as they would normally. You can learn more about Tivoli Storage Manager in *Tivoli Storage Management Concepts*, SG24-4877.

- Tivoli Disaster Recovery Manager

Tivoli Disaster Recovery Manager automatically generates a disaster recovery plan containing the information, scripts, and procedures needed to automate restoration to help ensure quick recovery of your data after a disaster. It automatically manages and tracks the media on which your data is stored, whether on-site, in-transit, or off-site in a vault, so your data can be easily located if disaster strikes.

1.1.1 Tivoli Storage Manager Backup-Archive client

The Tivoli Storage Manager Backup-Archive client is designed to back up and restore, archive and retrieve client file system data. The client therefore can back up any non-database and database files. Tivoli Storage Manager clients use standard operating system functions to access files within file systems, but they do not require to understand any logical structure that might exist within a file.

This affects how DB2 and other database systems are backed up. Each database appears as an individual file on the server or client file systems. A Tivoli Storage Manager Backup-Archive client running on an DB2 server or client can back up and restore, archive and retrieve entire DB2 databases. It cannot back up smaller increments.

Other than the issues of size and replication, using a Tivoli Storage Manager Backup-Archive client to back up DB2 databases is straightforward. Each database is a self-contained data file that is backed up and restored. Tivoli Storage Manager restores a database in its entirety, because it is just a file for Tivoli Storage Manager. If a database is deleted or corrupted, it is a simple task for Tivoli Storage Manager to restore the most recent or any previous backup version of this database from the Tivoli Storage Manager server to the DB2 server or client.

The Tivoli Storage Manager Backup-Archive client, however, does not meet all requirements for an ideal storage management solution in a DB2 environment.

Some drawbacks when using the Tivoli Storage Manager Backup-Archive client are:

- Consider a 5 GB database that changes everyday. The Tivoli Storage Manager Backup-Archive client will back up the full 5 GB even if only a 2 MB document has changed. You waste a lot of time and storage space using this strategy.
- Many databases need to operate twenty four hours a day, seven days a week so they are in use all the time and a consistent backup cannot be taken. The alternative is to quiesce the DB2 database and take backups, but this would result in server unavailability, which is not good for business.

1.1.2 Tivoli Storage Manager API

To overcome the above restrictions of the Backup-Archive client, DB2 uses the Tivoli Storage Manager API code to facilitate database backup. DB2

provides its own backup utility which allows backup at the tablespace level as well as a full database. The backup utility can be setup to use Tivoli Storage Manager as the backup media, as you will see later. Therefore, the two client types work together to provide full data protection for your DB2 environment.

The API client and the Tivoli Storage Manager Backup-Archive client can run simultaneously on the same DB2 server, however, they are totally separate clients as far as the Tivoli Storage Manager server is concerned.

Chapter 2. Relational databases and DB2 UDB product overview

In this chapter, we will discuss Relational Database Management System (RDBMS) concepts. It is aimed at the system administrators or Tivoli Storage Manager administrators who may want to familiarize themselves with some of the terminology of RDBMS in order to implement an RDBMS backup strategy with database administrators.

The fundamentals of relational databases common to all RDBMS are discussed first. This is followed by concepts specific to the DB2 UDB products.

2.1 Fundamentals of Relational Database Management Systems

Relational Database Management Systems (RDBMS) share a common set of principles. The purpose of this section is to explain a subset of these principles that a systems administrator or Tivoli Storage Manager administrator needs to understand in order to design a backup and recovery system for data held on a relational database. Please note that although all RDBMS products are based on the same set of principles, not all use the same terminology or structures. For example, the concept of tablespace does not exist on some RDBMS.

2.1.1 Database

A database presents data as a collection of tables. A table consists of a defined number of columns and any number of rows. A database can include a data dictionary or a set of system tables that describe the logical and physical structure of the data, a configuration file containing the parameter values allocated for the database, and a recovery log with ongoing transactions and archivable transactions. Some RDBMSs use control files as an extension of the data dictionary.

2.1.2 Tables

A table consists of data logically arranged in columns and rows. Figure 1 on page 8 shows that tables are assigned to tablespaces and that users interact with tables. Table data is accessed through Structured Query Language, a standardized language for defining and manipulating data in a relational database. The data in the table is logically related, and relationships can be defined between tables.

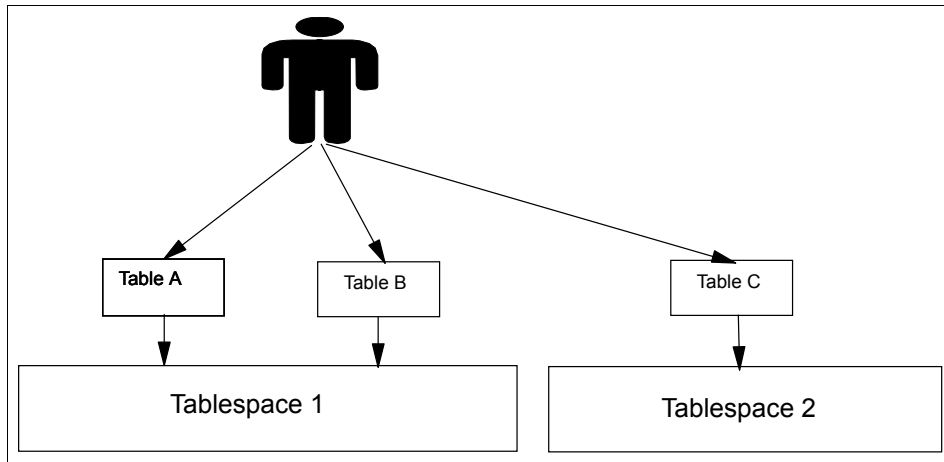


Figure 1. User interaction with tables

2.1.2.1 Data dictionary

It is common for RDBMS to maintain a data dictionary in a set of system tables. They describe the logical and physical structure of the data. They are like any other tables but are owned by the database administrator or by the database. They are created either when the database is created or when the database administrator runs a set of scripts supplied by the RDBMS. These tables contain information about the definitions of database objects such as user tables, and indexes, as well as security information and details relating to recovery.

2.1.3 Indexes

An index is a set of keys, each pointing to rows in a table. For example, Table A in Figure 2 has an index based on the employee numbers in the table. This key value provides a pointer to the rows in the table: employee number 19 points to employee KMP. An index allows more efficient access to rows in a table by creating a direct path to the data through pointers. It is possible for the data storage of an index to grow larger than the table to which it refers.

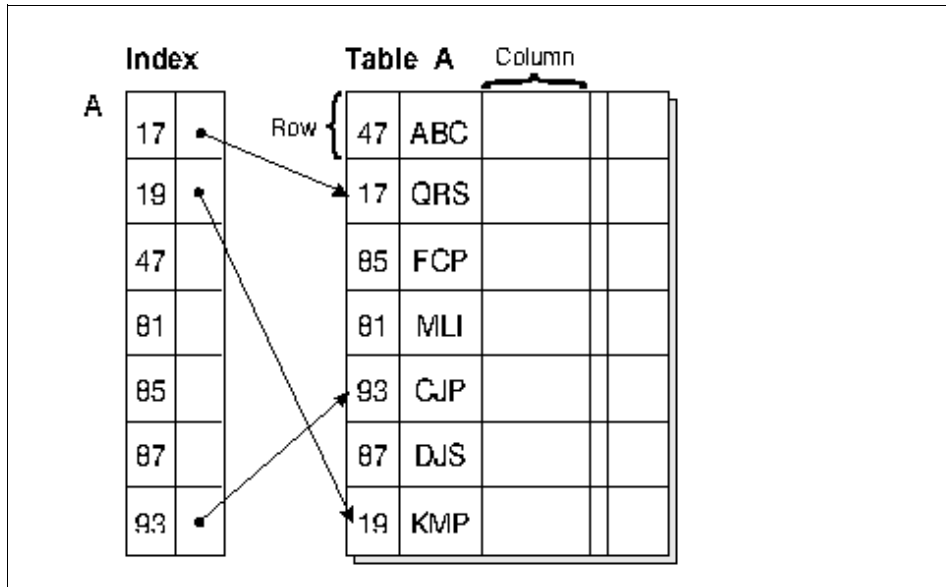


Figure 2. Using index in a table

2.1.4 Tablespaces

Tables and indexes are assigned to tablespaces as shown in Figure 3. This figure also shows that one or more data file can be allocated to a tablespace, whereas, different tablespaces cannot share the same data files. Some RDBMSs allow the several tablespaces to be defined on the same logical volume.

Indexes can be assigned to a different tablespace to the one where their tables reside to improve access speed. Normally, the data dictionary tables reside in their own tablespace.

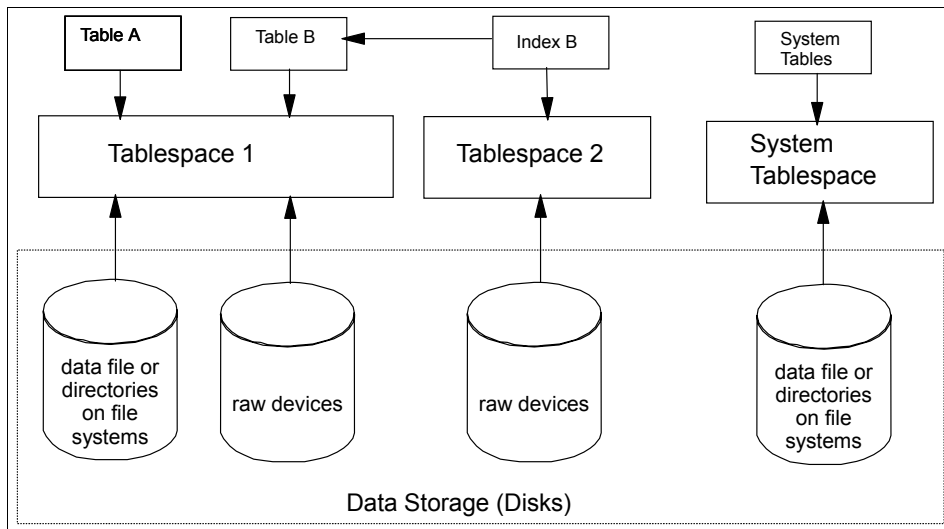


Figure 3. Tables, tablespaces, and data storage

Tablespaces are logical concepts used with RDBMSs. They provide a convenient way of separating the user's view of data from some of the practical considerations associated with storing that data on disk. For example, a database administrator can make more disk space available to several tables by adding disk space to the appropriate table space, therefore ensuring that tables do not run out of space and that disk space is used efficiently. Furthermore, the tablespace concept means that neither users nor application programs need to be aware of the fact that the database administrator has made more disk space available.

Data storage in tablespaces can be implemented using either data files or directories on file systems or raw devices. (For information on file systems and raw devices, please see your operating system documentation.)

Tablespaces provide the link between logical views and data storage. Points to note are:

- The data for a table or index may be contained in only one data file.
- Alternatively, the data for a table or index may be spread over several data files.
- Each of the data files may contain data for one or more tables in the tablespace.
- Each data file or directory may reside in a separate file system.

The significance of these alternatives is that the only way to back up or recover individual tables is by using the facilities that the RDBMS provides.

Normally, you would back up or restore tablespaces instead of the individual data files of the tablespaces. This ensures that all data storage for a tablespace is backed up consistently with the same timestamp.

You would use tablespace backup instead of full database backup depending on the volatility or importance of data. You have the option of backing up tablespaces which have more update activity more often than tablespaces which have less activity.

The tablespace where the data dictionary tables reside is the most important tablespace. You must ensure that this tablespace is backed up successfully and consistently with the other databases. Corruptions in the data dictionary can cause the database to be unusable.

2.1.5 Partitioning options

RDBMSs may provide partitioning options to handle very large amounts of data. This will allow workload parallelization of very large objects, and will allow for the manipulating of subsets of these large objects. You should investigate whether partitioning is used in your database and know how partitioning options are implemented by your RDBMS. This can affect your backup strategy.

2.1.6 Log files

As shown in Figure 4, most RDBMSs maintain details of updates to databases in log files. If, for some reason, a transaction that makes a change to the database fails to complete successfully, the RDBMS' recovery procedure will use the log file to detect that an update may be only partially complete and undo any changes that the transaction had made to the database.

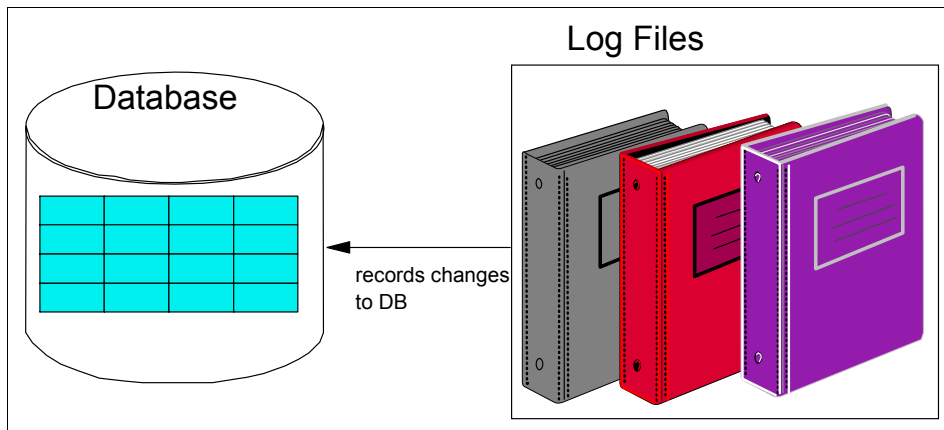


Figure 4. Log files

Some RDBMSs support the use of log files to perform forward recovery. Forward recovery takes advantage of the fact that log files hold details of all changes that have been made to the database, and therefore you do not necessarily need to undo changes, but instead can reapply changes. With forward recovery, the recovery process can:

- Restore a database to the state it was in at the time the last backup was taken
- Use the log files to reapply the changes that had been made since the last backup was taken
- Back out (undo) any partially completed changes

A standard RDBMS concept related to log files is the checkpoint process. All RDBMSs use buffers in memory to hold changes to the database and log files. The purpose of buffers is to improve the operational performance of the RDBMS. However, the use of buffers means that most changes to databases and log files do not get written to disk until some time after the RDBMS has indicated to the user application that the update has been made successfully.

Checkpoints ensure that all database and log file changes held in the RDBMS' buffers are flushed out to disk. This shortens the time it takes to recover a database after a system crash, because the number of redundant log records processed during the recovery is reduced. All RDBMSs support checkpoints and issue them automatically at intervals. We recommend that these files are mirrored or duplexed.

2.1.7 Control files

Some RDBMS maintain control files to hold additional information about the physical structure of the database, such as which physical files are used by each table space and which is the current log file.

For those RDBMSs which use control files, you need to define policies for backing up those files. We recommend that these files are mirrored or duplexed.

2.1.8 Configuration parameters

All RDBMSs provide a range of options. Some are set permanently, and others can be modified even when a database is in use (running). Some options allow you to tune the performance of the database; others allow you to specify how you want logging, for example, to be implemented. Depending on the RDBMS, you can either change the configuration parameters using database commands or modify them in an initialization file. The configuration parameters may be stored in a file as in the case of an initialization file.

For RDBMSs which use initialization files, a database may have multiple initialization. One reason for having multiple initialization files for a single database might be to optimize performance for different circumstances. For example, you may decide to allocate one set of values when the database is used for batch processing and another set when it is used for online transactions. Although some of the options are set differently for each situation, many will be the same. Some RDBMSs allow you to specify options that are common to multiple initialization files in configuration files. Instead of repeating all options and their values in each of the initialization files you can select the configuration file that contains the options that you want to use.

You need to define policies for backing up both initialization files and configuration files.

2.2 DB2 UDB concepts

The previous section provides a generic overview of databases for system administrators responsible for designing a backup and recovery system for data held on a relational database. The purpose of this section is to introduce DB2 UDB concepts that will help system administrators or Tivoli Storage Manager administrators implement a backup strategy that is specific to DB2.

Common concepts like tables and indexes have already been covered in 2.1, “Fundamentals of Relational Database Management Systems” on page 7. Now, we introduce concepts that are new or specific to DB2 UDB.

2.2.1 DB2 UDB products

DB2 UDB comes in many flavors. There is a Personal Edition (PE), a Workgroup Edition (WE), an Enterprise Edition (EE), and an Extended Enterprise Edition (EEE). For system administrators, we only need to understand the DB2 UDB EEE product databases that have the partitioning option. That means that one database can be created across one or more nodes or even machines. For the other products, each database created is typically only on one machine.

2.2.1.1 EEE partitioning option

Databases created in DB2 UDB EEE can use database partitioning. Each node in a partitioned database supports a subset of the overall database. They are called database partitions. As shown in Figure 5, a node will contain a database partition of the database which has its own data, configuration files, and transaction logs.

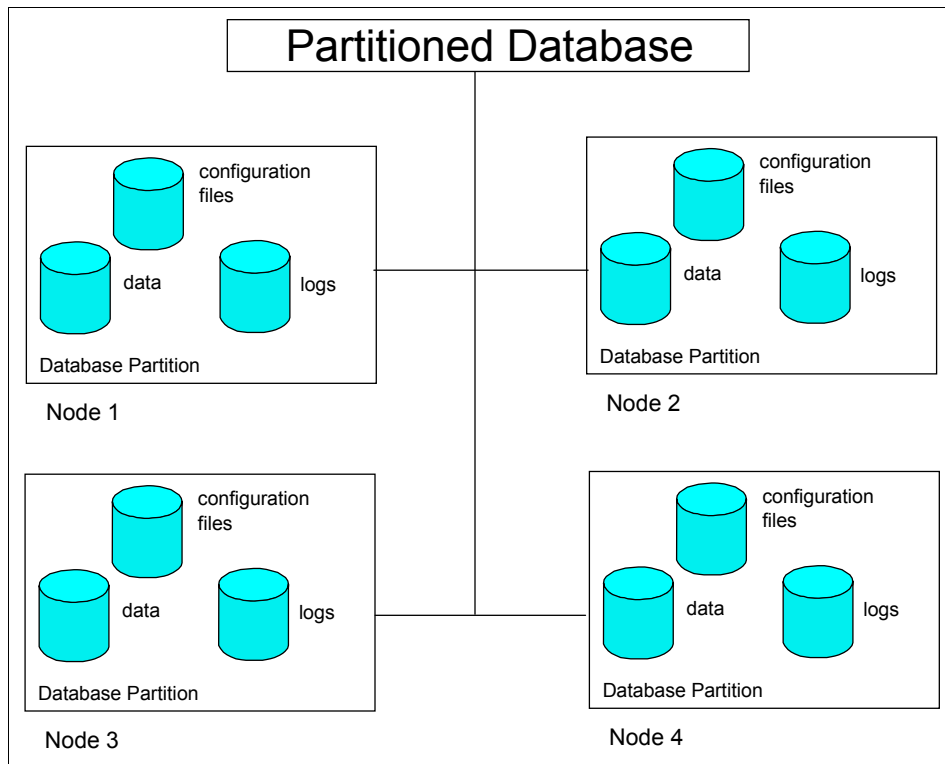


Figure 5. Database partitioning

The partitioned database can be configured to be flexible so that objects can be created on one or on many nodes depending on their use and on their size. So a small table can be stored on a tablespace that only exists in one node, while a large table can be stored on a tablespace which spans several nodes. Data in a table can, therefore, be distributed on several nodes. Data distribution is done using a hashing algorithm.

When there is a data retrieval or update request on tables which span several nodes, the request is decomposed automatically into subrequests, and executed in parallel among the applicable database partitions.

A database partition fits together with the MPP hardware architecture that is called a shared-nothing architecture, because each has its own data, configuration files, and transaction logs. When using media storage managers like Tivoli Storage Manager, you should install and configure the media manager on all machines with database partitions. Backup must also be done on each database partition. Recovering to a point in time must be carefully planned with database partitioning.

2.2.2 Instance

An instance (database manager) is an environment for managing data and system resources assigned to it. A machine or system can have more than one instance. Each instance will have its own database manager configuration parameters and security. An instance can have one or more databases. Figure 6 shows the hierarchy of DB2 objects in an instance.

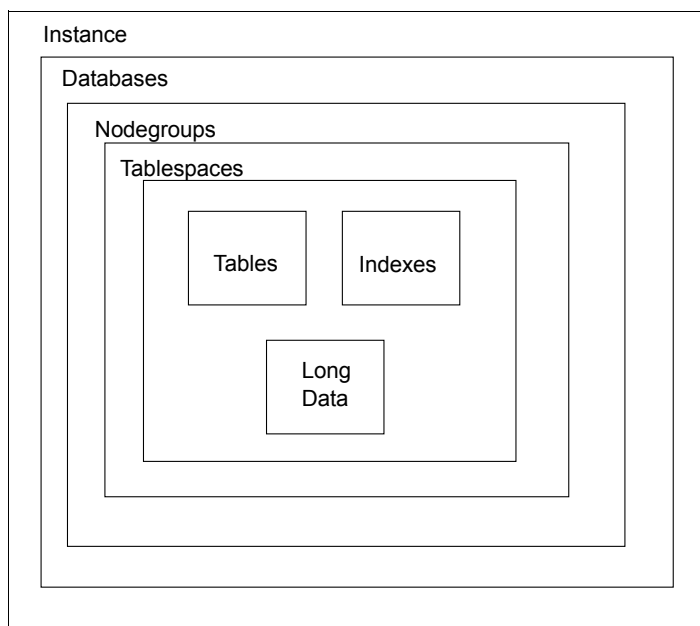


Figure 6. DB2 object hierarchy in an instance

2.2.3 Database and database partitions

An instance can have one or more databases. Each database has its own set of system tables (data dictionary). In DB2 UDB EEE, an instance can also have database partitions. A database or database partition has its own data, configuration files, and transaction logs.

Database partitions are initially defined in a file called `db2nodes.cfg`. Each database partition is assigned a node number which represents a node. A node is assigned to a hostname (machine). A node can be physical or logical. Physical nodes are nodes assigned on separate machines. Nodes participating on the same machine are called logical nodes. Logical nodes can be useful when exploiting the symmetrical multiprocessor (SMP) architecture. It is therefore possible to have more than one database partition for a partitioned database residing on the same machine.

You should perform individual backup for each database partition even if the database partitions reside on the same machine, because each database partition is a shared-nothing architecture.

You must also have a policy to backup the db2nodes.cfg file.

2.2.4 Nodegroups

A nodegroup is a set of one or more database partitions. It is only relevant for DB2 UDB EEE. Tablespaces are created in nodegroups, and tables and indexes are created in tablespaces. The data of a table or index can be distributed across several database partitions if it is defined in a tablespace, and the tablespace is defined in a nodegroup containing more than one database partition. A database partition can be a member of more than one nodegroup as shown in Figure 7.

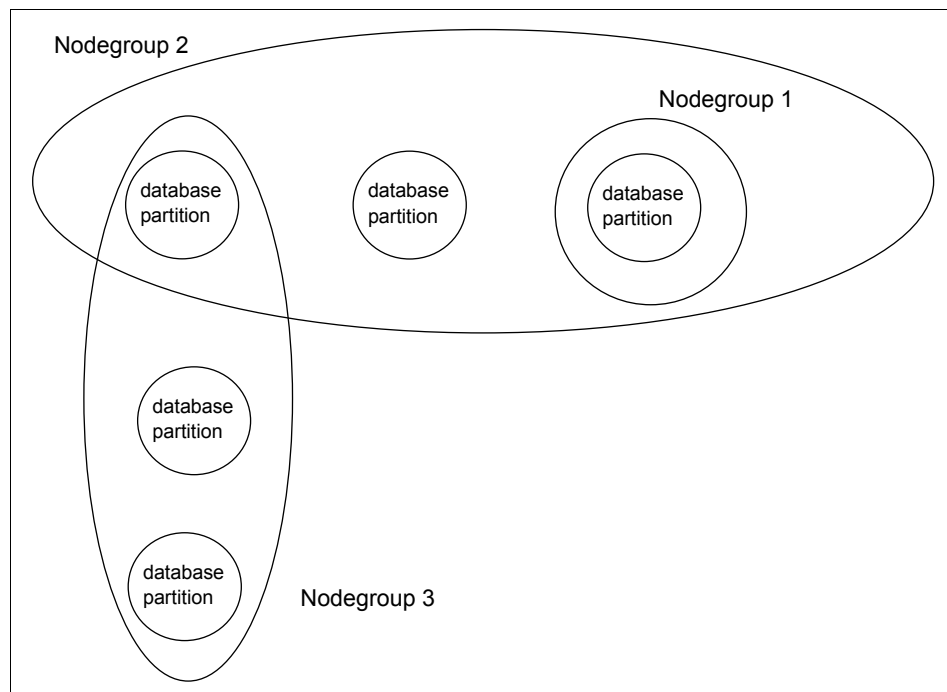


Figure 7. Nodegroups

2.2.5 Tablespaces

Tablespaces are created in nodegroups. A tablespace can span one or more physical storage device called containers. A container can be a directory

name, a file name or a device name (raw device). Backup can be done on a tablespace level if roll-forward recovery is enabled. (See 2.2.6, “Recovery logs” on page 18 for a discussion on roll-forward recovery).

There are two types of tablespaces: System Managed Space and Database Managed Space.

2.2.5.1 System Managed Space (SMS)

A SMS tablespace will have directories as containers. The operating system manages the space for the data. Files are created in the directories when the tablespace is created. The size of these files are increased when required. You should not make direct changes to these files, move, or remove them.

When tables are created in an SMS tablespace, their indexes and long fields or large objects (LOBs) will be created in the same tablespace.

2.2.5.2 Database Managed Space (DMS)

A DMS tablespace provides better performance in certain situations. A DMS tablespace will have files and raw devices as containers. The database manager controls the storage space. The space for the files or raw device are pre-allocated, and they cannot increase in size. To increase space for a tablespace, containers can be added.

Tables created in DMS tablespaces can have their indexes and long fields or large objects (LOBs) in separate tablespaces. Better performance can be achieved, for example, when indexes are created on tablespaces with faster devices. A good design will have a tablespace to contain tables only, indexes only or LOBs only data.

A DMS tablespace can be defined as:

- A regular tablespace to store tables and indexes
- A long tablespace to store long fields or LOBs

When using tablespace backup for DMS tablespaces containing tables, consider backing up their corresponding index and LOB tablespaces at the same time.

2.2.6 Recovery logs

Recovery logs or log files are used to recover from applications or system errors. By default, the log files are used in a circular fashion. When the last log file is full, the first log file is reused. With circular logging, you can only do offline database backup. Users cannot use the database while an offline backup is ongoing. Also, you can only use version recovery, that is,

recovering from the last available backup. Log files are not applied for version recovery.

Roll-forward recovery can be enabled by setting `logretain` or `userexit` to `on`. When roll-forward recovery is enabled, log files are kept and not reused, so they can be applied when performing roll-forward recovery. You can do online database backup where users can remain connected to the database while the backup is ongoing. You can also do tablespace backup either offline or online.

There are two type of log files:

- Active logs which contain current transaction data needed to do rollback or crash recovery
- Archive logs which contain committed data

Since DB2 does not provide multiplexing of log files, you must mirror the directory or the file system where the active logs are located.

2.2.7 Recovery history file

The recovery history file contains backup summary information that you can use to recover all or part of the database to a point in time. The recovery history file is always backed up during a database or tablespace backup. You have the option to restore only the history file when it is deleted or corrupted.

Chapter 3. Planning considerations

Planning is one of the most important areas for consideration before beginning to use Tivoli Storage Manager for database backups. It is important that the database administrator and the Tivoli Storage Manager administrator work together to anticipate the circumstances in which recovery will be required, as well as the resource and configuration requirements. These ideas apply to all types of databases.

In order to assist with this, this chapter will include details of some possible data recovery situations. We also cover the factors which should be weighed against one another in planning for recovery, for example, type of database, backup windows and relative speed of backup and recovery methods. We introduce various backup methods, which are covered in more detail in later chapters.

3.1 Backup requirements

A backup strategy is only one part of your overall data management plan. You must consider how important your data is to the function (or even existence) of your organization. The less time that your organization can function without out its data the more important that data is to you. Your system must be designed in such a way as to keep important data available when a failure occurs. However, reliance on backups is not necessarily sufficient. You must also consider the following:

- Redundant Array of Inexpensive Disk (RAID) devices
- Dual access paths
- Dual I/O controllers
- Dual power supplies
- Backup or standby processors
- Uninterruptable power supplies

None of these on their own can guarantee the availability of your data but in combination they can reduce the impact of a failure.

Before you can design a backup strategy you need to define the requirements that the strategy must satisfy. Factors that you will need to consider when defining the requirements for your backup strategy include:

- Types of events (The categories of incidents that may occur)
- Speed of recovery (How quickly you need to be able to recover)
- Backup windows (The periods of time at which backups can be performed)

- Recovery points (To which points in time you need to be able to recover)
- Units of recovery (Which other tables and files need to be recovered to the same point in time)

Let us look at each of these factors in more detail.

3.1.1 Types of events

We identify five categories of events that may require data recovery:

- User error
- Statement failure
- Transaction failure
- Media failure
- Disaster

Let us look at each category in more detail.

3.1.1.1 User error

There is considerable opportunity for a user to make an error that causes data to be lost. For example, a user may inadvertently delete or update rows in a table or accidentally drop an entire table, or a programmer could make a logic error that results in data loss or corruption.

RDBMSs provide facilities that reduce the risk or impact of user errors. For example, you can use RDBMS security to restrict the data that individual users can access or update. However, it is not possible to eliminate the risk entirely, and you need to consider how to handle such situations.

One approach is to say that it is the user's responsibility to recover from such errors. This approach may not be acceptable to users or their management, however. Another approach is to restore the entire database to the point in time at which the last backup was taken. This may not be satisfactory for other users who will lose the updates that they have made to the database since the last backup.

A third approach is to restore the table space that contains the damaged table. This approach is likely to be more acceptable than the other two, because:

- It removes the responsibility for data recovery from the users.
- It may impact fewer users. The number of users impacted will depend partly on the number of tables included in the affected table space.

You may, however, need to be able to restore individual tables, in which case you need to have backed up the tables individually.

3.1.1.2 Statement failure

SQL statements that are syntactically correct may fail, because, for example, the database is full. RDBMSs will usually detect such problems, roll back the effects of the failing statement, and report the problem to the user. Once the fundamental cause of the problem has been resolved, the user can retry the statement and continue to work. There is normally no need to take any special action to recover from SQL statement failures.

3.1.1.3 Transaction failure

Transactions may fail for a variety of reasons:

- Programming errors
- Network failures
- Failures of the operating system or RDBMS
- Power failures

The actions required to recover from these situations vary according to the particular circumstances. However, the RDBMS will ensure that the integrity of the data it manages is preserved. You do not need to restore data to recover from transaction failures.

3.1.1.4 Media failure

RDBMSs normally use magnetic disk as the medium on which they store the data that they manage. If a disk volume is physically damaged or destroyed, at a minimum, you need to restore the data files that have been lost to the state they were in when they were last backed up.

3.1.1.5 Disaster

Many organizations have developed plans for recovery from disasters such as floods, fires, accidents, earthquakes, and terrorist attacks. You need to ensure that your strategy for backing up and recovering data fits in with any such plans. For example, you may need to arrange for backups to be made to a removable medium and stored off-site. The subject of disaster recovery is too broad for us to address in this book and is not discussed in any more detail.

Almost all RDBMSs provide the facilities necessary to bring databases up to date by applying log files. They also provide the facilities necessary to undo changes made by partially completed transactions. This means that designers of database backup and recovery solutions do not need to concern themselves with recovering database data after statement failures or

transaction failures. For each type of event that may occur, designers of a database backup and recovery solution must:

- Ensure that operational procedures specify who needs to do what to recover from loss or corruption of data used by the RDBMS.
- Ensure that the data files that the RDBMS recovery routines use are available when needed.
- Ensure that data that the RDBMS does not manage can be recovered to a state that is consistent with the database.

3.1.2 Speed of recovery

If you ask users how quickly they would like you to be able to recover lost data, they usually answer "immediately". In practice, however, recovery takes time. The actual time taken depends on a number of factors, some of which are outside your control (for example, hardware may need to be repaired or replaced). Nevertheless, there are certain things that you can control and that will help to ensure that recovery time is acceptable:

- Develop a strategy that strikes the right balance between the cost of backup and the speed of recovery.
- Document the procedures necessary to recover from the loss of different groups or types of data files.
- Estimate the time required to execute these procedures (and do not forget the time involved in identifying the problem and the solution).
- Set user expectations realistically, for example, by publishing service levels that you are confident you can achieve.

3.1.3 Backup windows

Some RDBMSs do not allow databases to be backed up while they are in use. In such cases, you need to shut down the database before the backup starts, and you cannot restart the database until after the backup has completed.

Shutting down a database often means that users cannot use applications. You need to ensure that the times at which databases are shut down and unavailable are acceptable to your users.

Even if you can perform backups while the database is operational, you need to ensure that any load on processors or networks caused by the backup process does not result in performance or response degradation that is unacceptable to your users.

3.1.4 Recovery points

You need to define the points in time to which you will restore data. For example, you may need to recover the data to the state it was in when the last transaction was completed. Alternatively, it may be acceptable to restore the data to a consistent state that is no more than 24 hours old. In addition to either of these, you may be required to restore individual tables to the state they were in at any particular date within the last 30 days.

Whatever your situation, you need to consider recovery points and define a policy that is both achievable and acceptable to your user community.

3.1.5 Units of recovery

In some circumstances, it may not be sufficient to restore individual tables (or even entire databases) to the state they were in at some point in the past. Sometimes, in order to maintain data consistency, you may need to restore data held in tables or files that have not been lost or damaged. This undamaged data needs to be restored to the same point in time as the damaged data.

In developing your backup strategy, you need to understand the relationships between the data objects on which user applications rely. Many applications rely on relationships that extend beyond the data held in a single database. For example, an engineering database application holds references to documents that exist as independent file system files. If the engineering database is lost and restored to the point in time at which the last backup was taken, references to documents may be lost. Alternatively, the medium on which some of the documents are stored may be damaged. If the data files used to hold the documents are restored to the point in time at which the last backup was taken, the engineering database may contain references to documents that do not exist.

There are many other situations where you need to ensure that data consistency is preserved. The key point is that your backup and recovery strategy must take into account the needs of the applications that use the data.

3.1.6 Backup of RDBMS supporting files

Most RDBMS have certain non-database files which are required for their operation but which may not be part of its normal backup utility. These files can be initialization parameter files, password file, files that define the environment or network configuration files. They are external files and are not part of the database, because they must be accessible for reading, or editing

even when the database is down. For example, the password file provides authentication for database administration especially when starting up a database from a remote site.

You must ensure that these files are also backed up using operating system or third party tools like Tivoli Storage Manager.

3.2 Backup techniques

You can use any number of techniques to back up data managed by RDBMSs. These techniques are, at least at a conceptual level, common to most RDBMSs. The purpose of this section is help you understand the techniques well enough to design an appropriate backup strategy.

Often, a combination of techniques is used.

The techniques we consider are:

- Disk mirroring
- Offline backup
- Online backup
- Database export
- Full database backup
- Partial database backup
- Incremental backup
- Log file backup
- LAN-free backup
- Backup using splitcopy features

3.2.1 Disk mirroring

Disk mirroring is a useful technique for maximizing the availability of your database. Mirroring is the process of writing the same data to multiple storage devices at the same time. This is done either sequentially, when data is only written to the mirror once the master write is successful or, in parallel, when both master and mirror writes occur at the same time. The first method is slower but you are more likely to have at least one good copy of the data if a failure occurs.

When reading from a mirrored logical volume, AIX will read from either the master or the mirror, whichever is quicker at the time.

If a media failure occurs, operations are automatically switched to the good copy and AIX marks the faulty copy as stale. Mirroring allows your users to

continue working even though a media failure has occurred. Mirroring can be implemented in either software or hardware.

However, mirroring does not remove the need to back up databases. For example, disk mirroring will not allow you to restore a table that has been lost or damaged as a result of user error. Also, although disk mirroring dramatically reduces the impact of media failures, there is still a risk of damage to both sides of the mirror. If a database is held on one set of physical volumes, and a mirror image of the same database is maintained on a separate set of physical volumes, it is possible for both sets of physical volumes to be damaged or destroyed. This could happen as a result of a disaster or it could just be bad luck. In such instances, it will be necessary to recover the database from backup copies.

In DB2 UDB, you should at least consider mirroring the log file directory where the active logs reside. This will help ensure full recoverability to the point of failure.

3.2.2 Offline backup

Offline backup involves shutting the database down before you start the backup and restarting the database after backup is complete.

Offline backups are relatively simple to administer. However, they suffer from the obvious but significant disadvantage that neither users nor batch processes can access the database while the backup is taking place. You need to schedule sufficient time to perform the backup to ensure that the periods when the database will be unavailable are acceptable to your users.

Some RDBMSs provide a *single-user mode* or *quiesced mode*. You can think of this as an *almost offline* mode. A database administrator can still use the database, but general users cannot.

3.2.3 Online backup

Some (but not all) RDBMSs allow backups to be performed while the database is started and in use.

Clearly, if a database is being backed up while users are updating it, it is likely that the data backed up will be inconsistent ("fuzzy"). The RDBMSs that support online backup use log files during the recovery process to recover the database to a fully consistent state. This approach requires that you retain the RDBMS log files and indicate to the RDBMS when you are about to start the backup and when you have completed the backup.

Some RDBMSs allow you to quiesce activity on portions of the database (for example, a particular table space) so that a set of complete tables is temporarily "frozen" in a consistent state. You then can back up the set of tables that has been frozen. Once the backup has completed, you can reactivate the table space.

3.2.4 Database export

All RDBMSs provide export and import utilities. These utilities operate on logical objects as opposed to physical objects. For example, you can use an export command to copy an individual table to a file system file. At some later time, you might want to restore the table, in which case you would use the import command. Although export and import can be used for backup and restore operations, they are really designed for moving data, for example, for workload balancing or migration.

Note: You should assume that import will work only with files that have been created by the same RDBMS's export utility.

Most other utilities operate on the physical data files that RDBMSs use to store their databases. Therefore, other utilities cannot normally be used to back up and restore a single table, because:

- A single physical data file may contain data belonging to several tables.
- The data contained in a single table may be spread across multiple data files.

Therefore, the only way to gain access to the set of data contained in a single table is through the RDBMS itself.

Export utilities are usually slower than most other utilities and should be used only when you need access to database objects or raw devices.

3.2.5 Full database backup

Full database backups involve making copies of:

- Data files used to hold user data
- Data files that hold tables used by the RDBMS itself
- RDBMS log files
- Any control files and parameter files that the RDBMS uses

Many RDBMSs allow you to perform full database backup when the database is either online or offline. However, the technique for full database backup when the database is online can be quite different from offline.

To perform offline backup you can use the operating system utilities, RDBMS utilities, or Tivoli Storage Manager to back up the data files that constitute the database. To perform online backup you need to use an RDBMS utility to create data files containing a copy of the database. You can then use Tivoli Storage Manager to back up these data files together with the parameter files that you use to start up the RDBMS.

The simplest approach to database backup is to perform only full, offline backups at regular intervals. This approach is relatively easy to administer, and recovery is relatively straightforward. However, it may not be practical to take databases offline for the period of time that is necessary to perform full backups at the frequency you need. You may have to adopt a more flexible approach.

3.2.6 Partial database backup

Many RDBMSs allow partial database backups when the database is online or offline.

Partial database backups involve backing up a subset of the full database (such as the data files that make up a table space). Make sure that the subset you back up (as part of a partial backup) represents a complete logical unit of recovery from the point of view of the application. You may also need to back up data files that the RDBMS does not manage.

You must also ensure that the unit of recovery is consistent from the point of view of the RDBMS. If you have added a new data file to a table space, you must ensure that any control file that the RDBMS uses to define the relationship between data files and table spaces is also backed up.

3.2.7 Incremental backup

Some RDBMSs provide backup facilities for data that has changed since the last offline or online database backup. This will save tape or disk space but may or may not reduce the time to do a backup, because the RDBMS still needs to read the data blocks to determine if they have changed since the last backup. When recovery is needed, the database backup and incremental backups are both required to fully recover the database. This can take more time to recover a database. Incremental backups are useful when saving space or when saving bandwidth when backing up over the network.

Note

DB2 introduces the incremental backup option in V7.1 fixpak 3.

3.2.8 Log file backup (simulated incremental)

For some applications, the units of recovery are too large to be backed up on a daily basis. Sometimes the constraining factor is the elapsed time that is available (the backup window). Sometimes the load that the backup would place on the network would have an unacceptably high impact on other processes and users.

In such situations it may be possible to capture only the changes to the database by backing up the RDBMS' log files. Some database vendors refer to log file backup as incremental backup, but it is really a "simulated" incremental backup as opposed to a "true" incremental backup. A true incremental backup, backs up changed database blocks or pages, whereas a simulated incremental backup backs up the database transactions. Recovery from a simulated incremental can be much longer than from a true incremental, because you must reapply all of the transactions in the logs.

To recover from a log file or simulated incremental backup:

1. Restore the database from a full database backup (in some circumstances, restoring from a partial backup may be sufficient).
2. Restore the log files.
3. Apply the log files to the restored database.

3.2.9 LAN-free backup

Normally, a database backup has to go over the local area network (LAN) to the storage destination and may impact the users or applications using the same LAN. One way to overcome this problem is to use a dedicated LAN for backup so that the data transfer will no longer interfere with the work of other users and applications.

However, even when using a dedicated LAN all the data and metadata (file permissions, owner, and so on) must still be handled by the application that will receive the backup data. This application will need resources such as CPU, memory or disk space to buffer and manage data and metadata. To free the application from the work needed for handling the backup data itself a LAN-free solution can be used.

LAN-free means that a group of machines are able to share the same storage devices over a high performance connection. LAN-free also provides an easy way of defining storage devices to machines without much cabling effort, as the devices all communicate over the same high performance connection.

In our context LAN-free can be used in the following way. The clients can send their backup data files directly to the tape library (LAN-free) and only send the metadata information relating to the files to the storage application.

3.2.10 Backup using split mirror features

A backup always degrades the performance of the production system. Especially if the database is big or in a 7x24 hour environment, it is hard to find a timeframe when to plan the backup to not interfere with the normal operation. To free the production system from the overhead of backup, a copy or mirror of the database would be nice to be available for backup, report or other purposes.

Some intelligent storage servers such as IBM Enterprise Storage Server® ESS, support the split mirror feature. Split mirror means that identical and independent copies of disk volumes can be established within those storage servers. These copies can normally be established in a very short time. (For example, 5 to 20 seconds depending on device).

If the database resides on a storage server that supports the split mirror feature, a copy of the disk volumes can be established and assigned to another (backup) machine. On the backup machine, the (backup) database can then be accessed exclusively for backup or other purposes without reference to users.

During the creation of the copy volumes, an important requirement is that the data on the disk volumes is consistent. One way to establish this is to shutdown the database and to synchronize all the data that may reside in the memory of the operating system to disk. After the split mirror is established the database can be started again.

If the database cannot be stopped then the database itself must provide features to ensure that the data on the disk will be in a consistent state at the time of establishing the split mirror volumes.

Chapter 4. Tivoli Storage Manager server considerations

Tivoli Storage Manager server is an advanced storage management tool and provides data storage and retrieval services to Tivoli Storage Manager client programs. The client programs can be a native Tivoli Storage Manager Backup-Archive client, Tivoli Data Protection clients, or other third party products that interface to the Tivoli Storage Manager server using the Tivoli Storage Manager Application Program Interface (API). In this chapter, we discuss the steps, concepts, and considerations necessary to configure your Tivoli Storage Manager server and API client to manage UDB DB2 backups. See also Chapter 1, "Tivoli Storage Manager for database administrators" on page 3.

4.1 Initial requirements

The chapter assumes that you already have a Tivoli Storage Manager server installed and that it either has sufficient storage available in existing storage pools for your DB2 UDB database backups or that you know how to configure new ones. However, if this is a new Tivoli Storage Manager installation, you will first need to perform the initial configuration of the Tivoli Storage Manager server. Assistance with this can be found in the Tivoli Storage Manager server product documentation and other ITSO Tivoli Storage Manager Redbooks. Once this is complete you can begin to configure your Tivoli Storage Manager server to enable backup of the DB2 databases by defining your storage policies and registering nodes.

4.2 Tivoli Storage Manager server base functionality explained

The Tivoli Storage Manager server base functionality is to provide storage and retrieval of data objects (that is, files, directories, databases tables, and so on).

To store and retrieve objects at an application level, the application needs to provide native support for Tivoli Storage Manager. Or, a TDP product needs to interface between the application and the Tivoli Storage Manager API.

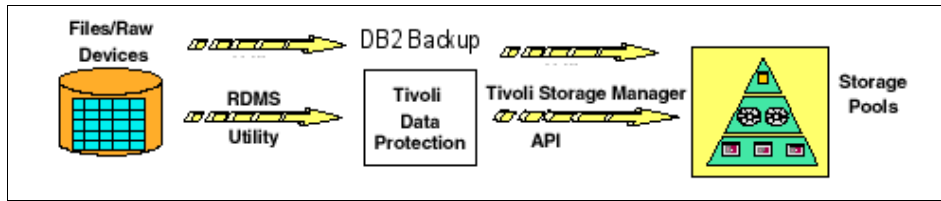


Figure 8. Native support or Tivoli Data Protection interfacing with the TSM API

There are three basic steps that must take place before you can use the Tivoli Storage Manager server. These steps will be covered in greater detail in subsequent chapters. A brief look into why these steps are necessary is included here.

1. Register a node with the Tivoli Storage Manager server.
2. Install the Tivoli Storage Manager client (Backup-Archive client, TDP and/or API client).
3. Configure the Tivoli Storage Manager client.

4.2.1 Registering a node with the Tivoli Storage Manager server

Before a Tivoli Storage Manager client can use the storage services of a Tivoli Storage Manager server, it must first authenticate to the Tivoli Storage Manager server using a nodename and password. A node is created by the Tivoli Storage Manager administrator using the REGISTER NODE command.

When the node is registered, it is given six key values that play an important role for performing backups and restores: NODENAME, PASSWORD, DOMAIN, ARCHDELETE, BACKDELETE, and MAXNUMMP.

These values will be explained in 4.6, “Node considerations” on page 53.

4.2.2 Installing the Tivoli Storage Manager client

This step includes installing the Tivoli Storage Manager Backup-Archive client (baclient), Tivoli Storage Manager API, and the Tivoli Data Protection product. Each of these pieces can be updated independent of the others. However, the Tivoli Storage Manager Backup-Archive client and the TDP product each are shipped with their own version of the Tivoli Storage Manager API. You may encounter situations where upgrading the Tivoli Storage Manager Backup-Archive or upgrading the Tivoli Data Protection product may change the Tivoli Storage Manager API level.

The baclient is not required to do Tivoli Data Protection or DB2 backups. It is needed to setup Tivoli Storage Manager scheduling and to backup files and directories that the DB2 backup utilities do not.

Since DB2 provides native support for Tivoli Storage Manager, a TDP product is not necessary. The only element required to do DB2 backups using Tivoli Storage Manager is to install and configure the Tivoli Storage Manager API.

4.2.3 Configure the Tivoli Storage Manager client

To configure the Tivoli Storage Manager client, you must perform the following tasks: set environment variables, create and specify options in the client options file, set up the scheduler and perform configuration tasks specific to the TDP product being used.

Tivoli Storage Manager client programs cannot “discover” Tivoli Storage Manager servers on a network. The communication method and the address of the Tivoli Storage Manager server must be specified in a client options file. The client options file is a plain text file, the default name of this file is `dsm.opt`. On UNIX systems, the client options file is a combination of two plain text files, `dsm.opt` and `dsm.sys`.

The Tivoli Storage Manager Backup-Archive client looks for the client options file in a default location or in a location specified by environment variables. The environment variables that are used are `DSM_CONFIG`, `DSM_DIR`, and `DSM_LOG`.

Each product that interfaces with the Tivoli Storage Manager server through the Tivoli Storage Manager API uses its own set of environment variables. The environment variables that are used are `DSMI_CONFIG`, `DSMI_DIR`, and `DSMI_LOG`. If these environment values are not set, default values are used that are different to the default values that the Tivoli Storage Manager Backup-Archive client uses.

4.3 Tivoli Storage Manager data objects

A data object is a copy or backup of information sent to the Tivoli Storage Manager server by a Tivoli Storage Manager client. This can be a file, directory, database, database tablespace, database log, or so forth. When the Tivoli Storage Manager server stores a data object, it requires the Tivoli Storage Manager client to provide three types of information:

- Whether to manage the data object as a backup or an archive object
- Description of the data object

- Management class to bind the object to

4.3.1 Archive or backup object

The Tivoli Storage Manager server stores and manages data objects as either backup objects or archive objects. The Tivoli Storage Manager client determines whether the data object is sent to the Tivoli Storage Manager server as an archive object or a backup object. With the baclient, you send a file or directory as a backup object by invoking the incremental or selective backup command. To send a data object as an archive object, you invoke the archive command. TDP and API products store data objects as backup or archive objects depending on the specific program.

UDB DB2 stores database and tablespace backups as backup objects on Tivoli Storage Manager storage. It stores logs as archive objects.

4.3.2 Managing Tivoli Storage Manager data objects

When a data object is sent, the Tivoli Storage Manager server places the copy of the client data out on Tivoli Storage Manager pre-allocated storage (in a storage pool) and records information in its own internal database describing the object. This description includes such things as the node name, filesystem name, high level qualifier, low level qualifier, and management class. This information is made available in a Tivoli Storage Manager internal table that can be queried using an SQL select statement. For backup objects you can run the command `select * from backups`. For archive objects you use the command `select * from archives`. Both commands are run using the Tivoli Storage Manager administrative command line program (`dsmadm`) or as a script.

4.3.2.1 Viewing the description of a backup object

Information about each backup object is externalized in a Tivoli Storage Manager database table named **backups**. You can use a SQL select statement to view the entries in this table. We will restrict the output of the select command by nodename. Before running this command we made a DB2 backup of the sample database using Tivoli Storage Manager.

```
db2 => backup db sample user db2admin using itsosj use tsm
Backup successful. The timestamp for this backup image is : 20010322134658
db2 =>
```

After taking the backup we ran the SQL query `select * from backups where node_name='JAMAICA_DB2'`. The entry in the quotes is case sensitive and must be upper case. The output of this command tells us that the backup object

was sent by a client program (DB2) that authenticated with the Tivoli Storage Manager server as node JAMAICA_DB2. The client program did not specify which management class to use with this object, so the Tivoli Storage Manager server bound the default management class for the node to this backup object. DB2 uses the database name as the FILESPACE_NAME when sending the object to the Tivoli Storage Manager server. The HL_NAME or high level qualifier corresponds to the DB2 structure nodegroup. Information on nodegroups and how DB2 is structured can be found in 2.2, “DB2 UDB concepts” on page 13. Our environment consists of single-partition DB2 configurations, so only a single nodegroup exists. For the LL_NAME or low level qualifier, DB2 uses the type of backup concatenated with the timestamp for the backup.

```
tsm: BRAZIL>select * from backups where node_name='JAMAICA_DB2'
ANR2963W This SQL query may produce a very large result table, or may require a
significant amount of time to compute.
Do you wish to proceed? (Yes (Y)/No (N)) y

NODE_NAME: JAMAICA_DB2
FILESPACE_NAME: /SAMPLE
STATE: ACTIVE_VERSION
TYPE: FILE
HL_NAME: \NODE0000\
LL_NAME: FULL_BACKUP.20010322134658.1
OBJECT_ID: 46908
BACKUP_DATE: 2001-03-22 13:48:42.000000
DEACTIVATE_DATE:
OWNER:
CLASS_NAME: DEFAULT
```

4.3.2.2 Viewing the description of an archive object

Information about each archive object is externalized in a Tivoli Storage Manager internal database table named **archives**. You can use an SQL select statement to view the entries in this table. We will restrict the output of the select command by nodename. Before running this command, we configured the DB2 sample database to invoke a user exit that archives logs to Tivoli Storage Manager. After performing this configuration we ran the SQL query `select * from archives where node_name='JAMAICA_DB2'`. The entry in the quotes is case sensitive and must be upper case. The output of this command tells us that the archive object was sent by a client program (DB2) that authenticated with the Tivoli Storage Manager server as node JAMAICA_DB2. The client program did not specify which management class to use with this object, so the Tivoli Storage Manager server bound the default management class for the node to this archive object. DB2 uses the database name as the FILESPACE_NAME when sending the object to the Tivoli Storage Manager server. The HL_NAME or high level qualifier

corresponds to the DB2 structure nodegroup. Information on nodegroups and how DB2 is structured can be found in 2.2, “DB2 UDB concepts” on page 13. Our environment consists of single-partition DB2 configurations, so only a single nodegroup exists. For the LL_NAME or low level qualifier, DB2 uses the filename for the log file.

```
tsm: BRAZIL>select * from archives where node_name='JAMAICA_DB2'  
ANR2963W This SQL query may produce a very large result table, or may require a  
significant amount of time to compute.  
Do you wish to proceed? (Yes (Y)/No (N)) y  
  
      NODE_NAME: JAMAICA_DB2  
      FILESPACE_NAME: \SAMPLE  
      TYPE: FILE  
      HL_NAME: \NODE0000\  
      LL_NAME: S0000164.LOG  
      OBJECT_ID: 48188  
      ARCHIVE_DATE: 2001-03-23 07:10:44.000000  
      OWNER:  
      DESCRIPTION: Log file for DB2 database SAMPLE  
      CLASS_NAME: DEFAULT
```

4.3.2.3 Differences between backup and archive objects

Based on the output of the SQL commands for backup and archive objects, you can see that there are some similarities and some differences.

The backups table contains two fields that the archives table does not: STATE and DEACTIVATE_DATE. These two fields are used in implementing versioning which is specific to backup objects. The STATE field can be either ACTIVE_VERSION or INACTIVE_VERSION. When the backup object is inactivated, the STATE changes to INACTIVE_VERSION and the timestamp when this occurred is placed in the DEACTIVATE_DATE field.

The archives table contains one field that the backups table does not: DESCRIPTION. The DESCRIPTION field is used to logically group archive objects together. For example, the Tivoli Storage Manager Backup-Archive client GUI uses this field during the retrieve operation to group archive objects that were backed up with similar descriptions.

4.3.2.4 Owner field in backups and archives tables

Both the backups and archives tables have an OWNER field for each data object. This is used for client data sent from a UNIX operating system. For files and directories, the OWNER corresponds to the UNIX security owner for the file or directory. For API data this field normally comes from the UNIX user that is used to perform the backup. Only the owner of the data object or a root user can access data objects on a Tivoli Storage Manager server or

authorize that access to other nodes. The Windows operating system that does not use this field, so in our examples in the prior sections, the OWNER field is blank.

DB2 backups and restores source the OWNER value from the user performing the backup or restore when PASSWORDACCESS is set to GENERATE. When PASSWORDACCESS is set to PROMPT, the OWNER is sourced from the DB2 database configuration parameter, TSM_OWNER. This plays an important role when doing redirected restores.

4.3.3 Management class to which to bind object

Every data object is bound to a management class when it is first sent to the Tivoli Storage Manager server. The management class determines how the object will be managed on the Tivoli Storage Manager server. A management class is one of the structures that is used in Tivoli Storage Manager policy management. We will not cover all details of Tivoli Storage Manager policy management in this book — only what is needed for backing up DB2 databases to Tivoli Storage Manager.

For a given node, the following can be said: every node belongs to one and only one Tivoli Storage Manager policy domain. The domain a node belongs to can be viewed with the Tivoli Storage Manager administrative command `query node <nodename>`. Replace `<nodename>` with the appropriate nodename. The output of that command for our node JAMAICA_DB2 shows that it belongs to the API_DOMAIN.

```
tsm: BRAZIL>q node jamaica_db2
Node Name           Platform Policy Domain  Days Since Days Since Locked?
                   Name          Name          Last Acce- Password
                   Name          Name          ss         Set
-----
JAMAICA_DB2        DB2          API_DOMAIN    <1         1         No
```

Each domain contains one and only one ACTIVE policysset. The active policysset has one *default* management class. The active policysset represents what policysset is currently in use. The default management class represents which management class will be used if a Tivoli Storage Manager client does not use a specific one, or if the Tivoli Storage Manager client specifies a management class that does not exist in the active policysset for the domain that the node belongs to. The active policysset for a domain and the default management class can be viewed with the command `query policy <domain> active`. Replace `<domain>` with the appropriate domain. The output of that

command for our API_DOMAIN shows that the default management class for nodes in the domain API_DOMAIN is API_MGMTCLASS.

```
tsm: BRAZIL>q policyset api_domain active
```

Policy Domain Name	Policy Set Name	Default Mgmt Class Name	Description
API_DOMA-IN	ACTIVE	API_MGMT-CLASS	PO for all DB-Backups

The active policyset for a domain can contain more than one management class. Only one can be the default management class. All management classes for the active policyset can be viewed with the command `query mgmtclass <domain> active`. Replace domain with the appropriate domain name. The output of that command from our API_DOMAIN shows that there are two management classes in the active policyset for the domain API_DOMAIN. These classes are API_LONGER and API_MGMTCLASS with API_MGMTCLASS being the default management class.

Policy Domain Name	Policy Set Name	Mgmt Class Name	Default Mgmt Class ?	Description
API_DOMA-IN	ACTIVE	API_LONGER	No	MG for all DB-Backups
API_DOMA-IN	ACTIVE	API_MGMT-CLASS	Yes	MG for all DB-Backups

Each management class can contain one archive copygroup and one backup copygroup. The copygroup is the final structure in the Tivoli Storage Manager policy management scheme and contains the most important information. The copygroup is where you specify where the initial destination storage pool of the data object should be. You also specify how the life cycle of the data object is to be managed. The life cycle concept will be covered in the next section. The life cycle is the term used in this book to describe how long a data object (backup or archive) resides on Tivoli Storage Manager storage before being purged.

The retention settings for a backup copygroup can be viewed with the command `query copygroup <domain> active <mgmtclass>`. Replace <domain> and <mgmtclass> with the appropriate values for your configuration. The output of that command for our default management class API_MGMTCLASS

in the domain API_DOMAIN shows the retention settings for the backup copygroup.

```
tsm: BRAZIL>q copygroup api_domain active api_mgmtclass
```

Policy Domain Name	Policy Set Name	Mgmt Class Name	Copy Group Name	Versions Data Exists	Versions Data Deleted	Retain Extra Versions	Retain Only Version
API_DOMA- IN	ACTIVE	API_MGMT- CLASS	STANDARD	1	0	0	0

To see the initial destination storage pool for this copygroup, you have to use the `format=detailed` option. Each of the query commands can be used with the additional parameter `format=detailed` to obtain additional information. The same command used previously to view the backup copygroup plus the `format=detailed` option shows that the initial destination (COPY DESTINATION) is the API_DISK_B storage pool.

```
tsm: BRAZIL>q copygroup api_domain active api_mgmtclass format=detailed
```

```

Policy Domain Name: API_DOMAIN
Policy Set Name: ACTIVE
Mgmt Class Name: API_MGMTCLASS
Copy Group Name: STANDARD
Copy Group Type: Backup
Versions Data Exists: 1
Versions Data Deleted: 0
Retain Extra Versions: 0
Retain Only Version: 0
Copy Mode: Modified
Copy Serialization: Shared Dynamic
Copy Frequency: 0
Copy Destination: API_DISK_B
Last Update by (administrator): ADMIN
Last Update Date/Time: 03/21/2001 14:23:48
Managing profile:

```

To see the retention settings for an archive copygroup, you add the option `type=archive` to the query copygroup command `query copygroup <domain> active <mgmtclass> type=archive`. Replace `<domain>` and `<mgmtclass>` with the appropriate values for your configuration. The output of that command for our default management class API_MGMTCLASS in the domain API_DOMAIN shows the retention settings for the archive copygroup.

```
tsm: BRAZIL>q copygroup api_domain active api_mgmtclass type=archive
```

Policy Domain Name	Policy Set Name	Mgmt Class Name	Copy Group Name	Retain Version
API_DOMA- IN	ACTIVE	API_MGMT- CLASS	STANDARD	365

To see the initial destination storage pool for this copygroup, you have to use the `format=detailed` option again. The same command used previously to view the archive copygroup plus the `format=detailed` option shows that the initial destination (`COPY DESTINATION`) is the `API_DISK_A` storage pool.

```
tsm: BRAZIL>q copygroup api_domain active api_mgmtclass type=archive format=detailed
```

```
Policy Domain Name: API_DOMAIN
Policy Set Name: ACTIVE
Mgmt Class Name: API_MGMTCLASS
Copy Group Name: STANDARD
Copy Group Type: Archive
Retain Version: 365
Copy Serialization: Shared Static
Copy Frequency: CMD
Copy Mode: Absolute
Copy Destination: API_DISK_A
Last Update by (administrator): ADMIN
Last Update Date/Time: 03/08/2001 17:30:39
Managing profile:
```

To summarize the information presented in this section with regards to data objects:

- Every archive data object sent to the Tivoli Storage Manager server is associated with an archive copygroup that determines the initial destination storage pool and how long the archive data object should reside on Tivoli Storage Manager storage.
- Every backup data object sent by the Tivoli Storage Manager server is associated with a backup copygroup that determines the initial destination storage pool and how long the backup data object should reside on Tivoli Storage Manager storage.
- For the archive or backup data object, the copygroup comes from the management class to which the object is bound. The management class that is used to bind comes from the active policysset for the domain of the node that sent the data object. The management class used will either be

the default management class for the active policyset or one specified by the Tivoli Storage Manager client.

- Once the object is sent to the Tivoli Storage Manager server you can verify which management class was used by performing a SQL query of the archives or backups Tivoli Storage Manager internal database table.

4.3.4 Life cycle of Tivoli Storage Manager data objects

Backup data objects and archive data objects are managed differently. We will use the term *life cycle* to describe how these data objects exist on Tivoli Storage Manager storage from initial creation to when they are purged. An understanding of these concepts and processes is important for the database administrator when working with the Tivoli Storage Manager administrator to configure Tivoli Storage Manager for database backups. These settings will determine how many backup copies are retained and for what length of time.

4.3.4.1 Life cycle of archive data objects

An archive object exists in two states *current* and *expired* before being purged from the Tivoli Storage Manager server. Figure 9 shows the three steps involved in the life cycle of an archive data object.

1. A copy of the client data is sent to the Tivoli Storage Manager server as an archive object. Upon initial creation the archive object is in a current state.
2. It remains in a current state until the Tivoli Storage Manager client program deletes the archive object manually, or the archive object exceeds its retention setting. At this point the archive object changes state from current to expired.
3. The archive object remains in the expired state until expiration processing runs on the Tivoli Storage Manager server. This process is invoked by a Tivoli Storage Manager administrator with the `expire inventory` command. When expiration processing encounters an archive object in the expired state, it purges that object from the Tivoli Storage Manager database and frees up the storage space where the archive object resided.

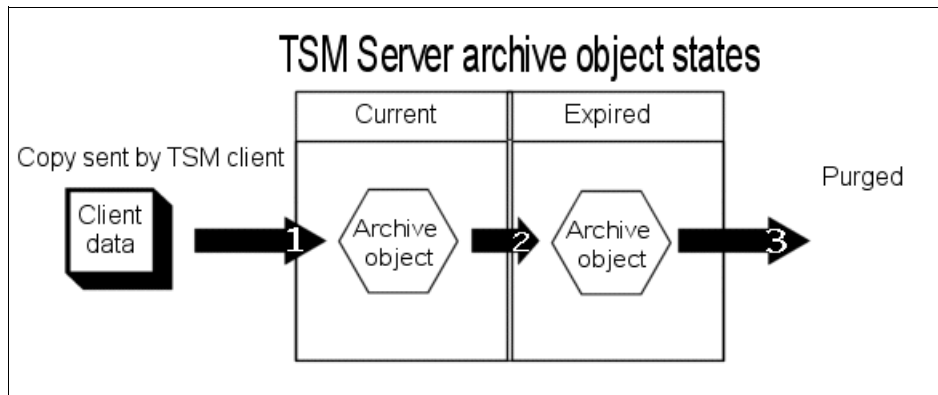


Figure 9. Archive object life cycle

When an archive object moves into the expired state, it is no longer accessible by the Tivoli Storage Manager client. Additionally, there is no way for the archive object to change back to a current state once it has become expired.

4.3.4.2 Life cycle of backup data objects

A backup object exists in three states, *active*, *inactive*, and *expired* before being purged from the Tivoli Storage Manager server. Figure 10 shows the four steps involved in the life cycle of a backup data object.

1. A copy of the client data is sent to the Tivoli Storage Manager server as a backup object. When a backup object is sent to the Tivoli Storage Manager server, it becomes the active version.
2. It remains in an active state until the Tivoli Storage Manager client program deletes the backup object manually, or a newer version of the backup object is sent. At this point the backup object changes state from active to inactive.
3. The backup object remains inactive until it exceeds its retention settings. A backup object can exceed retention settings by either time or number of versions. At this point the backup object changes state from inactive to expired.
4. The backup object remains in the expired state until expiration processing runs on the Tivoli Storage Manager server. This process is invoked by a Tivoli Storage Manager administrator with the `expire inventory` command. When expiration processing encounters a backup object in the expired state, it purges that object from the Tivoli Storage Manager database and

releases the storage space where the backup object resided to be reclaimed later.

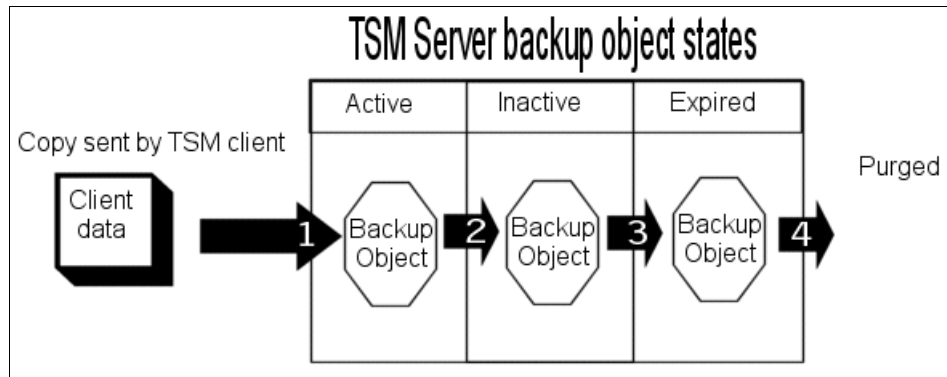


Figure 10. Backup object life cycle

A backup object that is the active version or in the active state will never be purged from Tivoli Storage Manager storage (never expires). It must first be inactivated by the Tivoli Storage Manager client program. The Tivoli Storage Manager client program can do this by manually deleting the backup object or sending a new version of the backup object.

When a backup object becomes inactive or moves into the inactive state, it is still accessible by the Tivoli Storage Manager client. A main difference between active and inactive is that an active object becomes inactive due to a client operation. An inactive object becomes expired automatically by the Tivoli Storage Manager server as soon as it exceeds its retention criteria. Changing from inactive to expired does not require a client operation. A backup object cannot change back to the active state once it has become inactive, nor to the inactive state once it has been expired. When a backup object moves into the expired state, it is no longer accessible to the Tivoli Storage Manager client.

If the retention for the backup object is set to retain zero inactive objects (`verexist=1, verdel=0`) or to retain inactive copies for zero days (`retextra=0, reonly=0`), the active backup object will change to the expired state as soon as the active backup is inactivated.

What a unique backup object is to Tivoli Storage Manager

Both backup and archive objects can be manually deactivated by the client program that initially backed them up. A key difference between backup and archive objects is that a backup object changes states when a newer version of the backup object is sent to the Tivoli Storage Manager server. This brings

up the question: how does the Tivoli Storage Manager determine what a unique version is? A backup object is considered *unique* based on NODE_NAME, FILESPACE_NAME, HL_NAME, LL_NAME. These fields and how to view them were discussed in 4.3.2.1, “Viewing the description of a backup object” on page 36. When a backup data object is sent to the Tivoli Storage Manager server, it has the same NODE_NAME, FILESPACE_NAME, HL_NAME, LL_NAME as an existing backup data object. The new data object becomes the ACTIVE_VERSION, and the older version changes state and becomes an INACTIVE_VERSION.

Many API products use a unique value for the LL_NAME based on a timestamp or a random non-recurring value. Because of this unique value for the LL_NAME the backup object will only change states from active to inactive when the API product manually inactivates (deletes) the backup object. An active object is not subject to retention settings until it is inactivated. You must run the appropriate command from the API product to inactivate these backup objects or else they will remain forever on the Tivoli Storage Manager server.

4.4 Tivoli Storage Manager server considerations for UDB DB2 backups

Whenever you use the Tivoli Storage Manager server to backup and restore data objects, it is of the utmost importance to consider which management class the data objects will be bound to. This is true of both API and Backup-Archive clients. Failure to do so will result in storing the data objects in one of three situations: too long, too short, and just right. It is highly unlikely that you will manage the objects *just right* if you do not take the time to define your storage requirements, configure the Tivoli Storage Manager server appropriately, and configure the Tivoli Storage Manager client to use the correct management classes. If you store the data objects too long, then you waste space and storage resources on the Tivoli Storage Manager server. If you store the data objects for too short a time, then you don't have the required files when you need them.

Each of the TDP products and any product that uses the Tivoli Storage Manager API should have a section in the documentation that describes exactly what retention settings the product uses and how to bind the data objects to the appropriate management class. If the documentation does not contain this information, you should call support for the product and request this information. The Tivoli Storage Manager server cannot and does not know how long a client program needs to keep the data objects. This must be done by the client.

4.4.1 How UDB DB2 stores data objects

Database objects stored on the Tivoli Storage Manager server by the DB2 BACKUP command with the USE Tivoli Storage Manager option are stored as backup objects. Each DB2 backup is stored as a unique object by specifying a time stamp as part of the low level qualifier (LL_NAME). This means that the DB2 backups must be manually inactivated. This is done by using the `db2adut1 delete` command. This also means that the management class that the backup objects are bound to should have retention settings that change the inactivated backup objects to be expired immediately. The retention settings for a backup copy group that would facilitate this is `RETONLY=0` and `VERDELETED=0`.

The userexit that can be compiled to automate storage of the DB2 log files, stores the logs as archive objects on the Tivoli Storage Manager server. The log archives should be manually deleted. This is done by using the `db2adut1 delete` command. The management class that the archive objects are bound to should have retention settings that keep the archive objects forever until they are deleted manually. The retention settings for an archive copygroup that would facilitate this is `RETVER=NOLIMIT`.

4.5 Policy management considerations

When deciding how to setup your Tivoli Storage Manager server to store DB2 backup and archive objects, you need to decide on how you are going to organize your nodes into domains. You must also correctly define and configure your management classes.

4.5.1 Domain considerations

You can use one domain for all your client data or you can specify multiple Tivoli Storage Manager domains to group nodes with the same backup characteristics together. Using multiple domains helps ensure that the data objects get bound to the appropriate management class. The following are some items to consider if you want to use multiple domains:

- BA-client, API-client
- Platforms separated: AIX, SUN, WIN and so on
- Critical data, non-critical data
- Group nodes together with same data characteristics

Because the policy requirements for DB2 backups are different from the desired settings for regular Tivoli Storage Manager backup clients, a different management class must be defined within Tivoli Storage Manager for

managing these DB2 backups. There are two ways to implement this different management class setup:

- Define a new management class within an existing policy domain.
- Define a separate policy domain where the default management class contains the required settings.

4.5.1.1 Define a new management class in an existing domain

If you choose to define a new management class within an existing policy domain (which is not the default management class for that domain), then you must add an include statement in the client options file (`dsm.opt` for Windows; `dsm.sys` for UNIX) that is used by the DB2 node. This include statement binds the DB2 backup objects to that management class that you have defined for managing these backups.

The include statement would be:

```
include * ManagementClassName
```

With DB2 you can also specify the management class by setting the `TSM_MGMTCLASS` option within the DB2 database. The DB2 command would be:

```
db2 update cfg for sample using TSM_MGMTCLASS ManagementClassName
```

4.5.1.2 Define a separate policy domain

The recommended way is to define a separate policy domain where the default management class has the required settings. Then just register the node that will be used for DB2 backups to this new domain. If the node is already defined, the Update Node command can be used to move the node to this new domain. This method allows the default management class to be utilized for the DB2 backups and there is no concern over an include statement not being recognized during the backup process.

4.5.2 Tivoli Storage Manager management class considerations

This section describes how the management class and its associated copygroups should be configured.

4.5.2.1 Backup copy group considerations

Normally, Tivoli Storage Manager backup copy groups are designed to hold multiple versions of files and directories to restore not only the latest (active) but also older versions that had been changed or deleted (inactive).

DB2 database will send its full and tablespace backups through the Tivoli Storage Manager client API directly to the backup copy group of the default management class to which the node is assigned to. DB2 assigns unique names to every database backups. The settings that pertain to multiple versions do not apply.

The following retention settings should be used for the management class that will be bound to the DB2 backups:

- VEREXISTS=1
Keeps only one version of the backup file as the name of the backup is unique. (There will not be a newer version of the backup image with the same name).
- VERDELETED=0
If the backup file has been deleted (via `db2adutl`), then Tivoli Storage Manager should not keep an inactive version of this file.
- RETEXTRA=0 (the same value as RETONLY)
RETEXTRA parameter will never be used as you will never have more than one version of the backup file. To prevent confusion set this parameter to the same value as RETONLY.
- RETONLY=0
When a backup image file becomes inactive it will be purged from the Tivoli Storage Manager server at the next expiration.

4.5.2.2 Special consideration

The values of VERDELETED and RETONLY can be changed so that DB2 database and tablespace backups become inactive for a period of time before becoming expired. To do this, change VERDELETED=1 and RETONLY=number of days to keep inactive. You can then use the `db2adutl` command with the `show inactive` option to access the inactivated backup objects.

Doing this is not recommended for a number of reasons. Archive logs do not have the same ability to be kept, once deleted. Online backups without the associated logs are worthless. Space is wasted on the Tivoli Storage Manager server during normal operation.

4.5.2.3 Archive copy group considerations

Archive copy groups are designed to hold archive objects for a dedicated time and then to automatically delete the object.

DB2 database will send its logfiles through the Tivoli Storage Manager client API directly to the archive copy group of the default management class to which the node is assigned. The logfiles are numbered from S0000000.LOG to S9999999.LOG. In normal operation it is unlikely to need to access such a vast number of logfiles. But, be aware when doing a point in time restore of a DB2 database, some logfiles may be reused and therefore stored twice.

You should set the retention for the archive copy group to nolimit.

- RETVER: NOLIMIT

This will hold logfiles forever unless they are deleted by db2adutl. Like the deletion of the backup images, the deletion of the logfiles should be the responsibility of the DB2 administrator.

4.5.3 Tivoli Storage Manager client include-exclude option

The include-exclude list is normally configured for the Tivoli Storage Manager Backup-Archive client. It provides the Tivoli Storage Manager Backup-Archive client with the default information which files to include in a backup and which to exclude. Another issue of this option is to define to which management class each file should be bound.

Each DB2 backup file will be bound, if not other specified, to the default management class to which the DB2 node is assigned to at the Tivoli Storage Manager server. To send a backup image or logfile to another management class the include-exclude option can be used. A similar thing can also be achieved using DB2 configuration (See 5.5, “Optional DB2 configurations” on page 69).

Before we start we need to know how the Tivoli Storage Manager server stores the DB2 backups and logfiles. Following are the commands to get the required information.

Login via the administrative command line interface to the Tivoli Storage Manager server. Use `dsmadm` with your userid and password. Then enter following command on the tsm prompt. Be aware to write the Tivoli Storage Manager node name in uppercase.

```
tsm: BRAZIL>select * from backups where node_name='BRAZIL_DB2'  
ANR2963W This SQL query may produce a very large result table, or may require a  
significant amount of time to compute.
```

```
Do you wish to proceed? (Yes/No) y
```

```
      NODE_NAME: BRAZIL_DB2  
      FILESPACE_NAME: /SAMPLE  
      STATE: ACTIVE_VERSION  
      TYPE: FILE  
      HL_NAME: /NODE0000/  
      LL_NAME: FULL_BACKUP.20010308190920.1  
      OBJECT_ID: 25958  
      BACKUP_DATE: 2001-03-08 19:09:20.000000  
      DEACTIVATE_DATE:  
      OWNER: db2inst1  
      CLASS_NAME: DEFAULT  
  
      NODE_NAME: BRAZIL_DB2  
      FILESPACE_NAME: /SAMPLE  
      STATE: ACTIVE_VERSION  
      TYPE: FILE  
      HL_NAME: /NODE0000/  
      LL_NAME: FULL_BACKUP.20010308200920.1
```

To get information about the stored logfiles, enter following command at the Tivoli Storage Manager prompt.

```
tsm: BRAZIL>select * from archives where node_name='BRAZIL_DB2'  
ANR2963W This SQL query may produce a very large result table, or may require a  
significant amount of time to compute.
```

```
Do you wish to proceed? (Yes/No) y
```

```
      NODE_NAME: BRAZIL_DB2  
      FILESPACE_NAME: /SAMPLE  
      TYPE: FILE  
      HL_NAME: /NODE0000/  
      LL_NAME: S0000009.LOG  
      OBJECT_ID: 29812  
      ARCHIVE_DATE: 2001-03-12 09:13:56.000000  
      OWNER: db2inst1  
      DESCRIPTION: Log file for DB2 database SAMPLE  
      CLASS_NAME: DEFAULT  
  
      NODE_NAME: BRAZIL_DB2  
      FILESPACE_NAME: /SAMPLE  
      TYPE: FILE  
      HL_NAME: /NODE0000/  
      LL_NAME: S0000010.LOG
```

The name of the file consists of three parts: the filespace name, the high level name and the low level name. To decide which file should go into which

management class, any combination can be specified in the include-exclude option.

For Windows based systems, the `INCLUDE` statement must be specified in the `dsm.opt` file. For UNIX based systems, the `INCLUDE` statement must be specified in the `dsm.sys` file. Additionally, there is an `INCLEXCL` option for UNIX based systems to define a file that contains all include-exclude statements for a specific server.

```
Servername          Brazil_db2 * Name of this stanza
COMMethod          TCPip
TCPPort            1500 * Port where Server is listening
TCPServeraddress   9.1.150.57 * IP-Adress of TSM Server
NODename           Brazil_db2 * Must match node name on TSM Server
TCPCLIENTAddress  9.1.150.57 * IP-Adress of machine running
                  * the TSM Client API
TCPCLIENTPort     1501 * Each locally client must have a
                  * different TCP Port if running at
                  * the same time
PASSWORDAccess     Generate * password will be crypted and saved
INCLUDE            /DB/.../* MGMTCLASS
INCLEXCL           /usr/tivoli/tsm/client/api/bin/inclexcl.list
```

Figure 11 shows a sample include-exclude list file. The include-exclude file will be compared from the bottom to the top every time a file is being backed up to the Tivoli Storage Manager server. The management class of the line, where the filename first fit, will be used to send the file to.

```
INCLUDE /.../*
INCLUDE /SAMPLE/.../* SAMPLE_BACKUP
INCLUDE /SAMPLE/NODE0000/*.LOG SAMPLE_LOGMGMT
INCLUDE /SPLITDB/.../* SPLIT_MGMTCLASS
```

Figure 11. Sample include-exclude list file

In our example, all database backups and logfiles of database `SPLITDB` will go to the management class `SPLIT_MGMTCLASS` (the backups in the backup copy group and the logfiles into the archive copy group).

All files from the `SAMPLE` database ending with `LOG` are going to the `SAMPLE_LOGMGMT` management class. (That means the logfiles will go to the archive copy group of this management class). All other `SAMPLE` database backups are going to the backup copygroup of the `SAMPLE_BACKUP` management class.

Keep in mind that each management class can have a separate copy destination for the archive and backup copygroup. If there is a need to send database files and logfiles into different storage pools, the above example of the SPLIT_MGMTCLASS will be sufficient if there are separate copy destinations for the archive and backup copygroup. This is only an example to show what is possible.

All the other database backups will go to the DEFAULT management class of this Tivoli Storage Manager node.

The include-exclude list itself can have any name and it can reside anywhere on the system. Because there can be more clients on the system using include-exclude list files, it is good to use the default directory `/usr/tivoli/tsm/client/api/bin` to put all the `inclexcl.list` files in. This file should be owned and controlled by the Tivoli Storage Manager server administrator.

4.6 Node considerations

A Tivoli Storage Manager client program must authenticate to the Tivoli Storage Manager server using a previously defined node. A Tivoli Storage Manager client can be a Backup-Archive-client or an API-client. There may be more than one Tivoli Storage Manager client on a single machine.

We strongly recommend registering a dedicated node for the DB2 backups that is separate from the node that is used with the Backup-Archive client. If you have a separate policy domain for DB2 backups this is required, because a node can only belong to one domain.

When registering a node, be sure to specify the following parameters:

- NODENAME
- PASSWORD
- BACKDELETE
- ARCHDELETE
- MAXNUMMP
- DOMAIN

4.6.1 Choosing a nodename

The Tivoli Storage Manager Backup-Archive client defaults the nodename to be the hostname of the machine. We chose to register separate nodes for the DB2 backups and used the naming convention of `hostname_db2`.

4.6.2 Choosing a password

This password will be used with the `dsmapipw` command to encrypt the password for future use when `PASSWORDACCESS` is set to `GENERATE`. If `PASSWORDACCESS` is set to `PROMPT`, you will be asked to input this password every time that it is required. When doing redirected restores you will set this password in the database into which you are restoring.

4.6.3 Setting the `BACKDELETE` option

The node that is used to backup the database backups to Tivoli Storage Manager must be able to manually delete the backup objects. To do this, specify `BACKDELETE=YES` when registering the node. You can update a node that has this value set to `NO` and set it to `YES` with the `update node` command.

If this value is set to `NO`, the `db2adut1 delete` command will not be able to manually inactivate the DB2 backup objects and they will reside on Tivoli Storage Manager storage forever.

4.6.4 Setting the `ARCHDELETE` option

The node that is used to archive the DB2 logs to Tivoli Storage Manager must be able to manually delete the archive objects. To do this, specify `ARCHDELETE=YES` when registering the node. You can update a node that has this value set to `NO` and set it to `YES` with the `update node` command.

If this value is set to `NO`, the `db2adut1 delete` command will not be able to manually inactivate the DB2 archive logs and they will reside on Tivoli Storage Manager storage forever.

4.6.5 Specifying the domain

The node that is used for backing up DB2 to Tivoli Storage Manager must belong to a domain that contains the required management class. The domain that a node belongs to can be specified when the node is initially registered, or the node can be updated to belong to a new domain. To do this, specify `DOMAIN=domain_name` when registering or updating the node.

4.6.6 Setting the `MAXNUMMP`

A node is restricted to the number of tape mounts specified with this option. In order for the Tivoli Storage Manager client to use multiple sessions when sending backups to tape, the maximum number of mount points parameter `MAXNUMMP` must be equal to or greater than the number of sessions involved in the backup.

The MAXNUMMP must not be set to a value greater than the actual number of physical drives defined to the Tivoli Storage Manager server. If DB2 is unable to acquire enough tape mounts, either because this value is set lower than the number of sessions, or because the tape mounts are not available, the backups may fail, or have to wait for one session to finish. For example, MAXNUMMP=2 (to allow the client use a maximum of 2 drives).

4.7 Storage pool considerations

Storage pools (stgpool) represent the actual physical devices that will hold your data. Each copy group (backup or archive) will send its data to the associated storage pool (configuration parameter: *destination*). It is possible to have only one storage pool (for example, one tape storage pool) to hold all the data from all the Tivoli Storage Manager clients; however, we do not recommend that you do this.

Here are some considerations for designing your storage pools:

- MAXSIZE (Disk stgpool Parameter)

For a storage pool, you can specify the maximum size object that it will accept. If a data object is larger than the maxsize, it will try to go to the next storage pool (typically tape). If the data object is larger than the maxsize of all storage pools in the hierarchy, then the Tivoli Storage Manager server will reject that object.

You can utilize this option to send really large database objects to tape instead of to your disk pool. If you do not have enough disk space to hold an entire days worth of backups, and you use migration to free up space, you can save time by sending the biggest objects straight to tape. This way all of your client sessions can be to send data to either disk or tape, and you don't have client sessions waiting on migration to finish. The other reason you save time is because the end result of migration is that the objects end up on tape. If you send it straight to tape, you don't spend the time sending it to disk, and then migrating it to tape.

We updated our storage pool so that the maxfile size was 50% of the total size of the disk storage pool.

```
update stgpool api_disk maxsize=2g
```

- USE DEDICATED STORAGE POOLS

Dedicated in this context means that the storage pool is reserved for one specific client. No other client sends files to this storage pool. There is no configuration parameter to implement this. It is a matter of how the policy setup had been done. There are different reasons why you may want to

use dedicated storage pools. Certain Tivoli Storage Manager server processes operate at the storage pool level, but you may want to have the server processes affect only certain clients. The work around for this is to send these clients to dedicated storage pools, you can then run the Tivoli Storage Manager server processes on these dedicated storage pools independently of your standard storage pools. Among these processes, the use of copy storage pools, collocation, and reclamation may warrant the use of dedicated storage pools.

- **COPY STORAGE POOL**

To safeguard against defective tapes and for off-site copies, copy storage pools can be used. If you have dedicated storage pools you can be more flexible as to which data should be copied to a copy storage pool and which not. As backups and logfiles are often very critical to a company this data is a good candidate to be copied. It may be useful to define a dedicated storage pool for the backup copy group and the archive copy group and only setup the archive copy group (containing the logfiles) to be copied to a copy storage pool.

- **RECLAMATION**

Reclamation relates to the management of sequential media by Tivoli Storage Manager, in particular, the release of empty tapes after data objects have expired and the storage space has been freed. The reclamation values are managed by the Tivoli Storage Manager administrator based on the tape usage agreed with database administrators during the planning stages.

4.8 Our Tivoli Storage Manager server setup

In this section we show the specific setup that we made to our existing Tivoli Storage Manager server to be ready to receive the DB2 backup files.

Our existing environment already contained the following parts:

- Tivoli Storage Manager Server 4.1.2
- Tivoli Storage Manager Client 4.1.1
- The atape driver 5.4.4.0
- Ultrium Tape Library 3583 with two 3580 tape drives.
- Ten formatted disk volumes named vol1 to vol10, each 2GB in size.

In the first step, we create the domain the policy and the management class (Figure 12).


```

tsm: BRAZIL>def domain api_domain DESC='DO for all DB-Backups'
ANR1500I Policy domain API_DOMAIN defined.

tsm: BRAZIL>

tsm: BRAZIL>def policysset api_domain api_policy DESC='PO for all DB Backups'
ANR1510I Policy set API_POLICY defined in policy domain API_DOMAIN.

tsm: BRAZIL>def mgmtclass api_domain api_policy api_mgmtclass DESC='MC for all DB-Ba
ckups'
ANR1520I Management class API_MGMTCLASS defined in policy domain API_DOMAIN,
set API_POLICY.

tsm: BRAZIL>

```

Figure 12. Define domain, policy and management class

Next, the tape and disk storage pools are being created and some pre-formatted disk volumes are assigned to the disk pool (Figure 13).

```

tsm: BRAZIL>def stgpool api_3580_a 3580class DESC='Archive tape pool' rec=95
maxscr=10 maxsize=10g
ANR2200I Storage pool API_3580_A defined (device class 3580CLASS).

tsm: BRAZIL>def stgpool api_3580_b 3580class DESC='Backup tape pool' rec=95
maxscr=10 maxsize=10g
ANR2200I Storage pool API_3580_B defined (device class 3580CLASS).

tsm: BRAZIL>

```

Figure 13. Define tape pools

```

tsm: BRAZIL>def stg api_disk_a DISK DESC='Archive disk pool' NEXT=api_3580_a
hl=70 lo=20
ANR2200I Storage pool API_DISK_A defined (device class DISK).

tsm: BRAZIL>def stg api_disk_b DISK DESC='Backup disk pool' NEXT=api_3580_a
hl=70 lo=20
ANR2200I Storage pool API_DISK_B defined (device class DISK).

tsm: BRAZIL>def vol api_disk_a /tsm/dskpool/vol1
ANR2206I Volume /tsm/dskpool/vol1 defined in storage pool API_DISK_A (device
class DISK).
tsm: BRAZIL>def vol api_disk_b /tsm/dskpool/vol2
ANR2206I Volume /tsm/dskpool/vol2 defined in storage pool API_DISK_B (device
class DISK).

```

Figure 14. Define disk pools and assign some volumes

After the storage pools are created, the copygroups can be configured and the above storage pools can be assigned as their destination storage pools (Figure 14 and Figure 15).

```
tsm: BRAZIL>def copygroup api_domain api_policy api_mgmtclass type=archive DEST=api_disk_a
retver=nolimit
ANR1535I Archive copy group STANDARD defined in policy domain API_DOMAIN, set
API_POLICY, management class API_MGMTCLASS.

tsm: BRAZIL>def copygroup api_domain api_policy api_mgmtclass type=backup DEST=api_disk_b
vere=1 verd=0 rete=0 reto=0
ANR1530I Backup copy group STANDARD defined in policy domain API_DOMAIN, set
API_POLICY, management class API_MGMTCLASS.

tsm: BRAZIL>
```

Figure 15. Define copygroups

To actually use the above configuration, the policy set needs to be validated and activated (Figure 16).

```
tsm: BRAZIL>validate policyset api_domain api_policy
ANR1515I Policy set API_POLICY validated in domain API_DOMAIN (ready for
activation).

tsm: BRAZIL>activate policyset api_domain api_policy

Do you wish to proceed? (Yes/No) y
ANR1514I Policy set API_POLICY activated in policy domain API_DOMAIN.

tsm: BRAZIL>
```

Figure 16. Validate and activate policy

Now the configuration is usable and we can define the client nodes to work with this setup (Figure 17).

```
tsm: BRAZIL>reg n brazil_db2 brazil_db2 do=api_domain archdel=yes backdel=yes ma
xnummp=2
ANR2060I Node BRAZIL_DB2 registered in policy domain API_DOMAIN.
ANR2099I Administrative userid BRAZIL_DB1 defined for OWNER access to node
BRAZIL_DB1.

tsm: BRAZIL>
```

Figure 17. Define Tivoli Storage Manager node

This completes the server setup.

Part 2. Backing up DB2 on UNIX platforms

Chapter 5. Backing up DB2 on AIX using Tivoli Storage Manager

In this chapter, we will describe in detail how to setup the Tivoli Storage Manager and DB2 environments to send DB2 backups and logfiles to a Tivoli Storage Manager server.

5.1 System requirements

The following products are needed to back up DB2 V7.1 using Tivoli Storage Manager.

- Tivoli Storage Manager client API V3 or higher.
- C compiler (only necessary when the user exit is needed). See 5.4, “Setting up the DB2 user exit for Tivoli Storage Manager” on page 67.

In general, all Tivoli Storage Manager client versions work together with every DB2 version. In our setup we used DB2 UDB V7.1 with Fixpak 2 and Tivoli Storage Manager client API V4.1.1.

Tivoli Storage Manager client API V4.1.1 will work with Tivoli Storage Manager server Version 3.7 or higher. We used Tivoli Storage Manager server 4.1.2.12.

For the Tivoli Storage Manager client API, check this Web site for hardware and software requirements:

http://www.tivoli.com/support/storage_mgr/requirements.html

Also check the IBM application availability guide for information about the end of service date for the products you use:

<http://www-1.ibm.com/servers/aix/products/ibmsw/list>

5.2 Installation of Tivoli Storage Manager related software

The Tivoli Storage Manager client API can be found on a Tivoli Storage Manager server CD or on the Internet. We recommend that you download the Tivoli Storage Manager client from the Internet as this will be the latest version or release. The Internet site for the clients is:

<ftp://ftp.software.ibm.com/storage/tivoli-storage-management/maintenance/client>

The details for how to download the client code from this site are found in 8.2, “Downloading latest Tivoli Storage Manager baclient and API” on page 150.

After successfully downloading the software use `smitty install_latest` or `installp` to install the Tivoli Storage Manager client API. A fileset `tivoli.tsm.client.api` must be installed. The extension refers to the supported AIX version and whether to run a 32 bit or 64 bit version.

In our lab environment, we used the 32 bit version of DB2, but if you want to use the 64-bit version of DB2, the following requirements must be fulfilled:

- The IBM @server pSeries® machine must support 64-bit.
- AIX 4.3.3.0 or higher is required.
- The Tivoli Storage Manager 64-bit client must be used.
- DB2 Version must be V7.1 Fixpak1 or higher.
- The DB2 instance must be created in 64-bit mode using `db2icrt -w 64`. A DB2 instance can be upgraded to 64-bit later using the `db2iupdt -w 64` command. To check if the DB2 instance is already created in 64-bit mode use the following procedure. (The following is an example for an instance which is not created in 64-bit mode.)

```
$ cd sqllib/adm
$ dump -HX64 db2sysc

db2sysc:
dump: db2sysc: 0654-108 file is not valid in the current object file mode.
      Use the -X option to specify the desired object mode.
```

Do not mix 32-bit and 64-bit Tivoli Storage Manager client and DB2 instances. The backup utilities will fail. For further information on 64-bit platforms, see the DB2 Technical Library and look for this publication *Using DB2 Universal Database™ on 64-bit Platforms*, which can be found at:

http://www-4.ibm.com/cgi-bin/db2www/data/db2/udb/winos2unix/support/v7pubs.d2w/en_main

Scroll down this Web page to find the publication within the Administration section.

5.2.1 Verification of Tivoli Storage Manager client API installation

After installation, check with `lsllpp -L "tivoli.tsm.client.api*"` that the fileset is correctly installed.

```
# lspp -L "tivoli.tsm.client.api*"
Fileset              Level  State  Description
-----
tivoli.tsm.client.api.aix43.32bit
                    4.1.1.0  C      TSM Client - Application
                    Programming Interface

tivoli.tsm.client.api.aix43.64bit
                    4.1.1.0  C      TSM Client - 64 Bit Application
                    Programming Interface

State Codes:
A -- Applied.
B -- Broken.
C -- Committed.
O -- Obsolete. (partially migrated to newer version)
? -- Inconsistent State...Run lppchk -v.
```

Now you should review the README.API in the /usr/tivoli/tsm/client/api/bin directory. Important files in this directory are the dsmtca (trusted client agent file); and libApiDS.a (API library file). Make sure that there is a link from /usr/lib/libApiDS.a to /usr/tivoli/tsm/client/api/bin/libApiDS.a.

5.3 Configuration and setup of Tivoli Storage Manager client API

After the installation of the Tivoli Storage Manager client software, the Tivoli Storage Manager client must be configured to communicate with the Tivoli Storage Manager server. In this section, we will only cover how to setup the communication. We assume that the configuration on the Tivoli Storage Manager server is done and validated prior to configuring the client. We assume that the Tivoli Storage Manager server configuration has already been completed and verified prior to configuring the client.

It is possible to configure different management classes for DB2 databases under the same DB2 instance. You can do this by setting the TSM_MGMTCLASS option. This option is set from the DB2 command line using the syntax.

```
'db2 update db cfg for <dbname> using TSM_MGMTCLASS <mgmtclass>'
```

(Replace <dbname> and <mgmtclass> with the desired database name and management class.)

The DB2 options that can be set for each database that relate to Tivoli Storage Manager are described in greater detail in 5.5, "Optional DB2 configurations" on page 69.

You can also configure different management classes using the client include/exclude list. This is done by specifying an INCLUDE statement for the database object that is being backed up. You can then add the desired management class to the INCLUDE statement. This works the same way as for the Tivoli Storage Manager Backup-Archive client. The only difference is that the include statement object matches the way DB2 sends the backup object rather than specifying a path and filename. This is described in greater detail in 4.5.3, “Tivoli Storage Manager client include-exclude option” on page 50.

The following information is needed before you begin to configure the client; ask your Tivoli Storage Manager administrator to get the relevant details.

- Tivoli Storage Manager server name and IP-address
- The nodename by which the API client will be known to the Tivoli Storage Manager server
- The password of the API node

5.3.1 API client setup

Root and instance owner permissions are needed to perform the setup. It is best to create a newly dedicated directory for all Tivoli Storage Manager related configuration and logfiles. Do not use the \$HOME/sql/lib/adsm directory of the instance owner, because this directory is only a link to the /usr/lpp/db2_07_01/adsm directory and may be overwritten at the next DB2 upgrade. In our lab environment we created and used the directory \$HOME/sql/lib/tsm for our working directory. (Also, the \$HOME/sql/lib/db2dump directory could be used so that DB2 logfiles and Tivoli Storage Manager logfiles reside together under one directory.)

We performed the following steps:

1. Set the environment variables for the API client. The *instance owner* of your database must set the following environment variables in either the *\$HOME/.profile* or in the *\$HOME/sql/lib/userprofile* file. **Note:** It is not recommended to define the environment variables in the *\$HOME/sql/lib/db2profile* file, because this file can be overridden when installing fixpaks. These DSMI variables will be read by the Tivoli Storage Manager API client (they all start with DSMI rather than with DSM which identifies those used by the Tivoli Storage Manager Backup-Archive client). The default location for the API is the /usr/tivoli/tsm/client/api/bin directory. (The default directory for the Backup-Archive client is the /usr/tivoli/tsm/client/ba/bin directory.)

The DB2 `backup` command reads and uses the Tivoli Storage Manager client API configuration.

- DSMI_DIR

Identifies the directory path where the agent file, *dsmtca*, is located. A korn shell example is:

```
export DSMI_DIR=/usr/tivoli/tsm/client/api/bin
```

- DSMI_CONFIG

Identifies the full directory path and file name of the Tivoli Storage Manager user options file, *dsm.opt*. This file contains the name of the server to be used. A korn shell example is:

```
export DSM_CONFIG=/home/db2inst1/tsm/dsm.opt
```

- DSMI_LOG

Identifies the directory path where the error log file, *dsierror.log*, is to be created. A korn shell example is:

```
export DSMI_LOG=/home/db2inst1/tsm
```

2. Log out and log in again as instance owner so that the environment variables take effect.
3. Create the *dsm.sys* file.

The `root` user of your system must create or modify the Tivoli Storage Manager system options file, *dsm.sys*, which must be located in the `/usr/tivoli/tsm/client/api/bin` directory.

The sample *dsm.sys.smp* in the `/usr/tivoli/tsm/client/api/bin` directory provides the basic information needed for DB2 backup to work.

Note

The `passwordaccess` parameter must be set to `generate`. This parameter specifies that Tivoli Storage Manager encrypts and stores the user password locally and generates a new password when the old one expires. Then DB2 is not required to supply a password each time it initializes a session with the Tivoli Storage Manager server.

The following is a sample *dsm.sys* file. We only need to look at the first stanza. The other stanzas belong to other Tivoli Storage Manager API clients so only part of the *dsm.sys* file is shown.

```

SErvername          Brazil_db2 * Name of this stanza
COMMethod           TCPip
TCPPort             1500      * Port where Server is listening
TCPServeraddress    9.1.150.57 * IP-Address of TSM Server
NODename            Brazil_db2 * Must match node name on TSM Server
TCPCLIENTAddress    9.1.150.57 * IP-Address of machine running
PASSWORDAccess      Generate * password will be crypted and saved

SErvername          Brazil_API2 * Next stanza in dsm.sys file
COMMethod           TCPip
TCPPort             1500
TCPServeraddress    9.1.150.57
NODename            Brazil_TDP
TCPCLIENTAddress    193.1.1.12

```

Figure 18. Example of dsm.sys file

4. Create the dsm.opt file.

The instance owner user can create or modify dsm.opt. This file must be located in the directory specified by the DSMI_CONFIG environment variable.

The dsm.opt file only needs to have one line in it which is a reference to the server stanza in the dsm.sys file. The Servername given here is only a symbolic one and does not need to match the name of the Tivoli Storage Manager server. It is possible to have two or more API clients on the same system (for example, two instances) with different server characteristics. Each client has its own dsm.opt file pointing to a different stanza in the dsm.sys file.

```

# cat dsm.opt
SErvername      Brazil_db2

```

5. As instance owner, we stop and start the DB2 instance. This allows DB2 to read the Tivoli Storage Manager configuration performed so far. (DB2 only reads the DSMI environment variables at the time when db2start is issued.)

```

$ db2stop
SQL1064N  DB2STOP processing was successful.
$ db2start
SQL1063N  DB2START processing was successful.
$

```

6. Set the Tivoli Storage Manager password.

Each Tivoli Storage Manager client must have a password to access a server. For that reason, the *root user* of your system must run the executable file, *dsmapipw*, installed in the `$HOME/sql/lib/adsm` directory of the instance owner, to establish and reset the Tivoli Storage Manager password. Make sure that the correct DSML environments (that of the instance owner) are set when root user executes the *dsmapipw* program. When executed, the *dsmapipw* program prompts you for the:

- *Old password*, which is the current password for the Tivoli Storage Manager node stored in the server.
- *New password*, which is the new password for the node.

After that you should check that a new file was created in the `/etc/security/adsm` directory that contains the encrypted password. The name of this file is the same as the value specified to the `servername` option in the `dsm.opt` file.

If you receive any errors, you should look at the `dserror.log` file which will show any API error messages. For a listing of API error messages, see “Appendix D. API Return Codes with Explanation” in *Tivoli Storage Manager Using the Application Program Interface V4R1*, SH26-4123, at:

http://www.tivoli.com/support/public/Prodman/public_manuals/storage_mgr/admanual.html

For further help, see Appendix B, “Troubleshooting” on page 277.

If there are multiple DB2 databases under one DB2 instance all databases make use of the same Tivoli Storage Manager client API setup. That means all the database backups will go to the same (the default) management class to which this Tivoli Storage Manager node belongs. How to configure different management classes for DB2 databases under the same DB2 instance will be covered in 4.5.3, “Tivoli Storage Manager client include-exclude option” on page 50, and 5.5, “Optional DB2 configurations” on page 69.

5.4 Setting up the DB2 user exit for Tivoli Storage Manager

By default, DB2 reuses logs in a circular fashion and are not archived. This will only allow you to do off-line backup and version recovery. You can enable archive logging for the database by setting `logretain` or `userexit` to `on`. You can set them both if you want. When one of these is set, the database is enabled for roll-forward recovery, and you can perform an online backup of the database and tablespaces.

If your database has to be up 24 x 7, you should enable roll-forward recovery by setting `logretain` or `userexit` to `on`, so that you can do online backups.

The following sections will describe how to enable roll-forward recovery by using the supplied user exit for Tivoli Storage Manager.

Note

The implementation of the user exit requires a full offline database backup in order to become effective. Ensure that you allocate sufficient time for doing this backup.

5.4.1 Compile the user exit for Tivoli Storage Manager

A user exit is an executable file that the DBMS calls for archive and retrieval of log files. Sample user exits are supplied in `$HOME/sql/lib/samples/c` of the instance owner. At the beginning of each sample file is a good description on how DB2 uses the specific user exit and which parameters you should set. For Tivoli Storage Manager, the user exit sample is `db2uext2.cadsm`. Perform the following steps to create the user exit executable.

1. Become root, then copy the `db2uext2.cadsm` to your working directory and rename it to `db2uext2.c`. Change file ownership and permissions so the instance owner can modify the code.
2. Read the comments at the beginning of the source code to understand the defined variables. Change the defined variables to suit your environment. In our lab, we took the default setting and only change the path of the user exit logfiles (`AUDIT_ERROR_PATH`). Our settings are as follows:

```
#define BUFFER_SIZE          4096      /* transmit or receive the log    */
                               /* file in 4k portions           */
#define AUDIT_ACTIVE         1        /* enable audit trail logging     */
#define ERROR_ACTIVE         1        /* enable error trail logging     */
#define AUDIT_ERROR_PATH    "/home/db2inst1/tsm/" /* path must end with a slash */
#define AUDIT_ERROR_ATTR    "a"      /* append to text file           */
```

3. As the instance owner, compile the C program. In our lab, we used the Visual Age C Compiler V4.4. Check your compiler for the correct syntax.

```
$ cc -I/opt/tivoli/tsm/client/api/bin/samples -L/usr/lib -lApiDS -o db2uext2 db2uext2.c
```

4. Copy the db2uext2 file into the \$HOME/sqllib/adm directory of the instance owner. We do not recommend that you copy the file to the \$HOME/sqllib/bin directory, because this directory is only a link to a directory in the /opt/IBMDB2/V7.1/bin directory. Make sure the instance owner has execute permission for this file.

Note

If there are multiple databases under one DB2 instance only one user exit program can be used for all the databases. As a result all the user exit logs of these databases merge to the same logfile.

5.4.2 Enable the database for roll-forward recovery

Finally, the DB2 environment must be customized to make use of the user exit program. For this purpose the user exit parameter must be set to ON in the database configuration. This will bring the database into rollforward mode so that full logfiles, which no longer contain active transactions, will automatically be moved off the system to Tivoli Storage Manager.

```
$ db2 update db cfg for sample using userexit on
```

All applications need to disconnect from the database before the change becomes effective. After all applications are disconnected the database is in backup pending state and no new connections are allowed to the database until a full database backup was made.

Please see 5.7.1, “Full offline backup” on page 73 for how to make a full offline backup.

5.5 Optional DB2 configurations

After the Tivoli Storage Manager client API is setup, additional configuration may be done at the DB2 database configuration level.

There are four database configuration parameters that can, but should not be, configured for backup purposes. If set, they will override the default Tivoli Storage Manager configuration done at the Tivoli Storage Manager server or client. Their main purpose is to provide an easy way to restore a DB2 database that was backed up to another Tivoli Storage Manager client node. For more information about this see 10.5.3, “Restore using DB2 parameter” on page 247.

Note

These database configuration parameters are designed to temporarily overwrite the Tivoli Storage Manager client API setup for the purpose of restore. They should not be used for normal backup operations.

- TSM_MGMTCLASS

This parameter specifies the management class the DB2 backups should go to. The management class must be in the same Tivoli Storage Manager policy domain where the Tivoli Storage Manager API node is defined.

Note: The logfiles will always go to the default management class of the Tivoli Storage Manager client API node, even if this parameter is specified. There are two ways to prevent this:

- One way is to use the `objectAttr.mcNameP` variable in the user exit program to set the management class, before the `dsmSendObj` function was called. (Be aware that if you have more DB2 databases under one DB2 instance, they will use the same `userexit` command!)

```
#define MGMT_CLASS "MY_DB2_CLASS"  
objectAttr.mcNameP=MGMT_CLASS;
```

- The other way is to use the Tivoli Storage Manager include-exclude list file. This is one out of three places where the destination management class can be set. The list of these places in top-down order of override priority is:

1. TSM_MGMTCLASS (DB2 configuration)
2. Entry in include-exclude list
3. The default management class of the Tivoli Storage Manager clients)

- TSM_NODENAME

- TSM_PASSWORD

- TSM_OWNER

Note

Each of these parameters, if set, will overwrite the related Tivoli Storage Manager configuration parameter. The password access needs to be set to `prompt` in the `dsm.sys` file; otherwise, the DB2 backup command will fail.

We recommend not to use these parameters. The decision as to which management class and to which node the backups and logfiles will go should be in the responsibility of the Tivoli Storage Manager administrator and it is best to handle by Tivoli Storage Manager.

Here is an example how to set these parameters and also how to unset them again.

```
$ db2 update db cfg for sample using TSM_MGMTCLASS api_mgmttest
DB20000I The UPDATE DATABASE CONFIGURATION command completed successfully.
DB21026I For most configuration parameters, all applications must disconnect
from this database before the changes become effective.

$ db2 update db cfg for sample using TSM_MGMTCLASS NULL
DB20000I The UPDATE DATABASE CONFIGURATION command completed successfully.
DB21026I For most configuration parameters, all applications must disconnect
from this database before the changes become effective.
```

Figure 19. Use DB2 Tivoli Storage Manager database configuration parameter

5.6 Using the Tivoli Storage Manager baclient in conjunction with DB2

In this section we show how Tivoli Storage Manager Backup-Archive (baclient) can be used to back up a DB2 database at a filesystem level and also how to back up other DB2 relevant files. We assume that a Tivoli Storage Manager baclient is already installed and configured.

Besides the normal DB2 database backup cycle, some operating system files need to be backed up to be able to restore the entire DB2 environment for the DB2 database. These files normally reside in the home directory of the instance owner. Examples of such files are:

- *.profile, db2profile, userprofile* of the instance owner
- Additional shell scripts or DB2 SQL scripts
- Tivoli Storage Manager config files (dsm.opt, dsm.sys, include-exclude list, and so on)
- The user exit program db2uext2 and the modified source code

The DB2 datafiles which compose the DB2 database itself are already being backed up by the DB2 `backup` command using Tivoli Storage Manager. Therefore, these files should be excluded from the list of files that will be backed up by the Tivoli Storage Manager baclient.

5.6.1 Using the baclient include/exclude file

The best place to define which files should be backed up by the Tivoli Storage Manager baclient is the include-exclude file of the Tivoli Storage Manager baclient. This file will be checked from bottom to top for every file that is being backed up by the Tivoli Storage Manager baclient. The first match of an include or exclude line decides if the file will be backed up or not.

In our setup we have databases that reside in the /home/db2inst1/db2inst1 and the /db2db directory. In the next example we only show the additional entries in the existing include-exclude file in order that the files in the /home/db2inst1 directory will be backed up; and files that reside in the /home/db2inst1/db2inst1 and /db2db directory will not be backed up.

```
INCLUDE /home/db2inst1/.../*
EXCLUDE /home/db2inst1/db2inst1/.../*
EXCLUDE.FS /db2db/.../*
```

Figure 20. Tivoli Storage Manager baclient include-exclude file example

Note

The EXCLUDE.FS statement will take effect no matter where it is placed in the include-exclude list file. This means that it will overrule every INCLUDE statement that is related to the same filesystem. For example, an EXCLUDE.FS /home will overrule an INCLUDE /home/db2inst1/.../* no matter where it is placed within the include-exclude list file.

5.6.2 Using the archive utility

If there is a special requirement to back up a full copy of the DB2 database datafiles, the archive option of the Tivoli Storage Manager baclient could be used. Archive operates independently of the database backup feature.

The *archive* option will create an archived set of files in the archive copy destination of the related management class at the Tivoli Storage Manager server and it will NOT read the include-exclude list file. It will archive all files specified no matter whether they have changed or not since the last backup or archive.

A description should be assigned to each archive set to facilitate identification at retrieval time. The database needs to be stopped or suspended in order to have a consistent state during archive. The command shown below will make an archive backup of all files that are under the /db2db/ directory.


```
# dsmc archive -subdir=yes -desc='DB2 file archive' /db2db/
```

To restore the archive again the *retrieve* option must be specified. See following example.

```
dsmc retrieve /db2db/ -desc='DB2 file archive'
```

5.7 Backing up DB2 using Tivoli Storage Manager

In this section, we examine how you can use Tivoli Storage Manager to back up DB2 databases. We cover:

- Offline backup
- Online backup
- Tablespace backup
- Load utility

5.7.1 Full offline backup

Please note that while the offline backup is ongoing, users cannot connect to the database. If you require users to have 24 x 7 availability, see 5.7.2, “Online database backup” on page 78.

The DB2 backup utility can be used from either:

- Command line interface (CLI)
- Graphical user interface (GUI)
- Your own C, COBOL, or Fortran language program

Our examples use the command line and graphical user interfaces.

5.7.1.1 Preparatory steps

Before you can start a backup or recovery operation, make sure that the database manager is up and running. You can issue the `db2start` command to start the database manager. If it already running, it will tell you that it is already active.

```
$ db2start
SQL1026N The database manager is already active.
$
```

Also make sure that the Tivoli Storage Manager server is online and policies are in place.

5.7.1.2 Offline backup using the command line

This example shows an offline backup using the DB2 command line interface.

1. Log in as the database administrator or higher authority and make sure that all applications are logged off from the database you want to back up. Use the following DB2 command to verify that all applications are logged off. Assuming you are backing up the SAMPLE database, the command and output can look like this:

```
$ db2 list applications for db sample
```

Auth Id	Application Name	Appl. Handle	Application Id	DB Name	# of Agents
DB2INST1	db2bp	6	*LOCAL.db2inst1.010312180936	SAMPLE	1

```
$
```

2. Logoff all applications connected to the database. In the above results, there is one application using the database. Use the force command enumerating the application handles (separated with commas) inside the parenthesis:

```
$ db2 "force application ( 6 )"
```

```
DB20000I The FORCE APPLICATION command completed successfully.
```

```
DB21024I This command is asynchronous and may not be effective immediately.
```

```
$
```

Note that you can use the command `force application all` to log off all applications if there are too many. However, be careful when using this command, because it will also logoff applications connected to other databases within the instance.

3. Verify that there are no more users connected to the database by again issuing the `list application` command.
4. Backup the database with the `tsm` option. The use `tsm` option tells the DBMS to use the Tivoli Storage Manager API to write the output backup file instead of using common devices. A confirmation will appear to indicate that the backup is successful.

```
$ db2 backup db sample use tsm

Backup successful. The timestamp for this backup image is : 20010312104622

$
```

5.7.1.3 Offline backup using the DB2 Control Center

For this example we will be using a Windows client to backup a remote database in the AIX platform.

1. Start the DB2 Control Center. For a Windows system, you will find the Control Center under **Start->Programs->IBM DB2->Control Center**. For command line or UNIX, type `db2cc`. Figure 21 shows the Control Center with systems that it can manage.

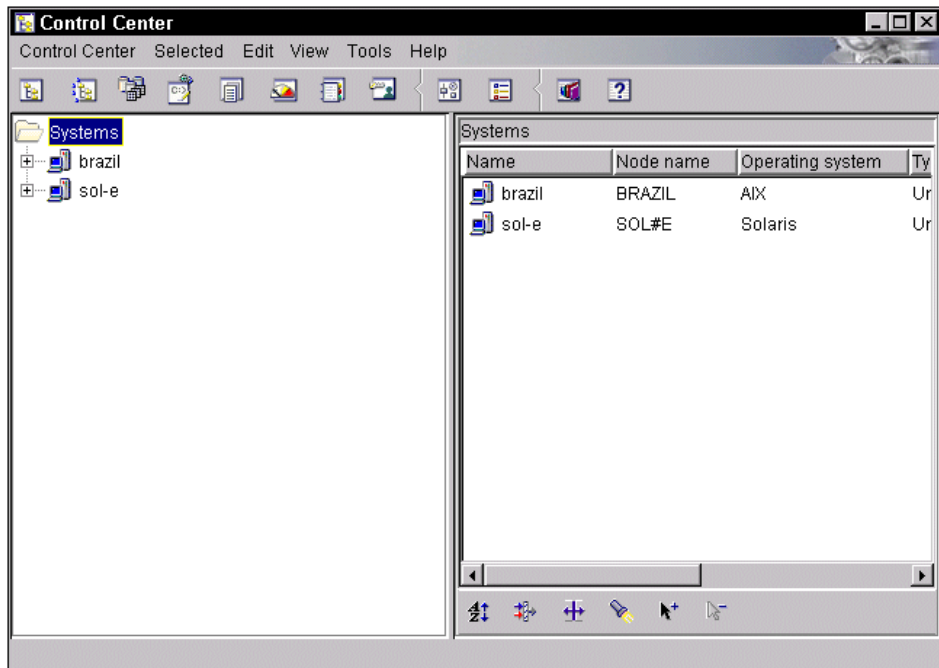


Figure 21. Control Center

2. Select the system where the database you want to backup resides. Since we are managing from a remote system, we are prompted as shown in Figure 22 with a dialog to enter the userid and password of the database administration server.

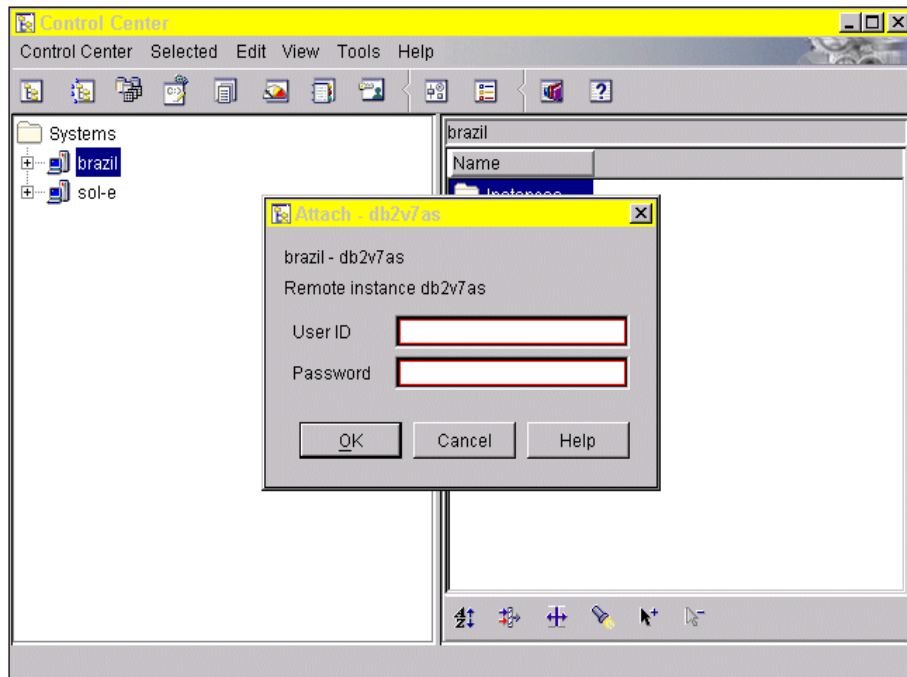


Figure 22. Attach to the administration server

Note that for most administration tasks, we use the database administration server instead of the database manager.

3. Review the systems until you see the database you want to backup. Right-click the database and select **Backup->Database** from the drop down menus as shown in Figure 23.

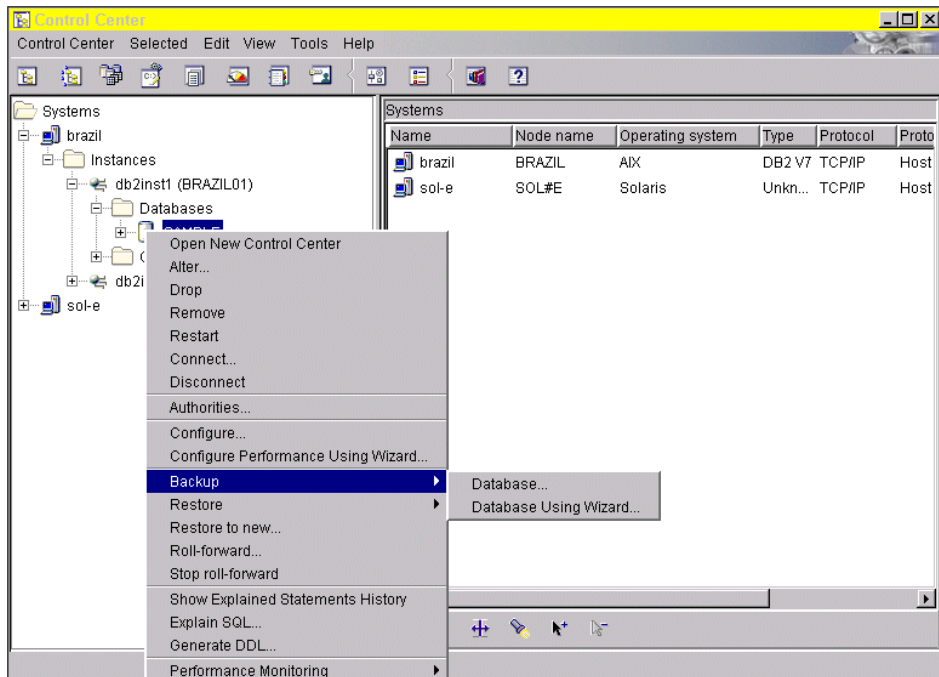


Figure 23. Drop down menu for database

4. The backup database dialog will now appear. From the Media Manager drop down list, select **Tivoli Storage Manager**. Figure 24 shows the dialog box.

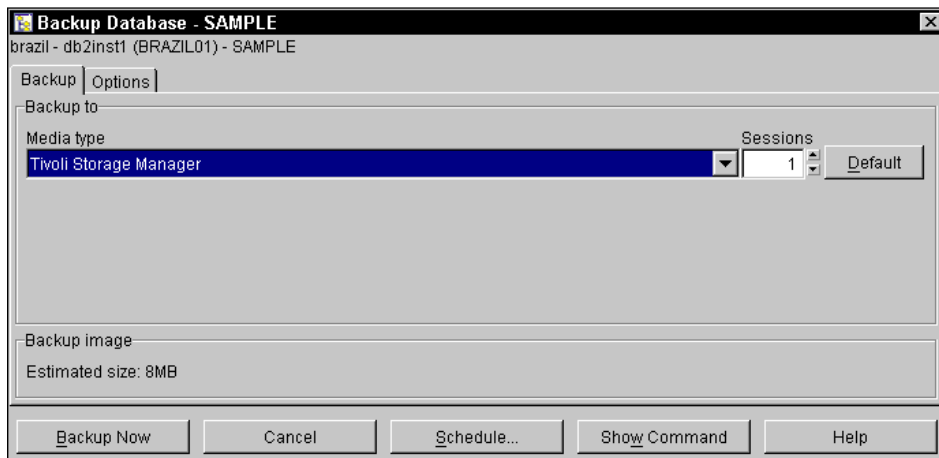


Figure 24. Backup database with Tivoli Storage Manager

5. Select **Backup Now** to backup the database. When the backup completes, a confirmation dialog appears as shown in Figure 25. Close the dialog box.

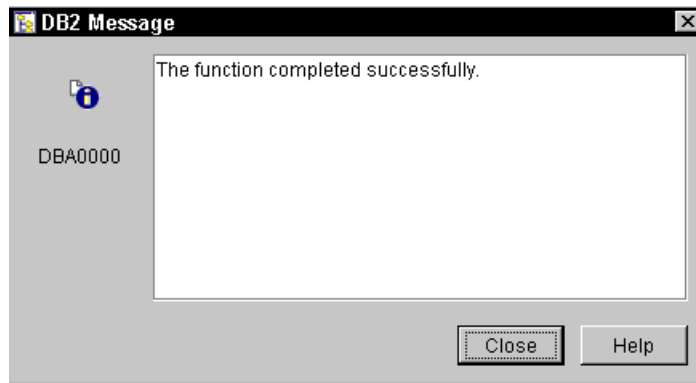


Figure 25. Backup successful dialog box

Check the DB2 Journal for additional information about the success of the backup. The DB2 journal is under the Tools Menu of the DB2 Control Center. We will discuss the Journal in more detail in the Chapter 7, “Day to day management: DB2 backups on UNIX” on page 119.

5.7.2 Online database backup

With online backup, users can continue to use the database while the backup is taking place.

5.7.2.1 Preparatory steps

You must enable roll-forward recovery in order to use online backup. See 5.4.2, “Enable the database for roll-forward recovery” on page 69.

5.7.2.2 Online backup using the command line

This example shows an online backup using the DB2 command line interface.

1. Log in as the database administrator or higher authority.
2. Backup the database with the `use tsm` option which tells the DBMS to use the Tivoli Storage Manager API to write the output backup file instead of using common devices. Assuming that you are backing up database SAMPLE, the command and results should look like this:

```
$ db2 backup db sample online use tsm
```

```
Backup successful. The timestamp for this backup image is : 20010312132637
```

```
$
```

5.7.2.3 Online backup using the DB2 Control Center

For this example we will be using a Windows client to backup a remote database in the AIX platform.

1. Start the DB2 Control Center. For a Windows system, you will find the Control Center under **Start->Programs->IBM DB2->Control Center**. For command line or UNIX, type `db2cc`. Figure 26 shows the Control Center with system that it can manage.

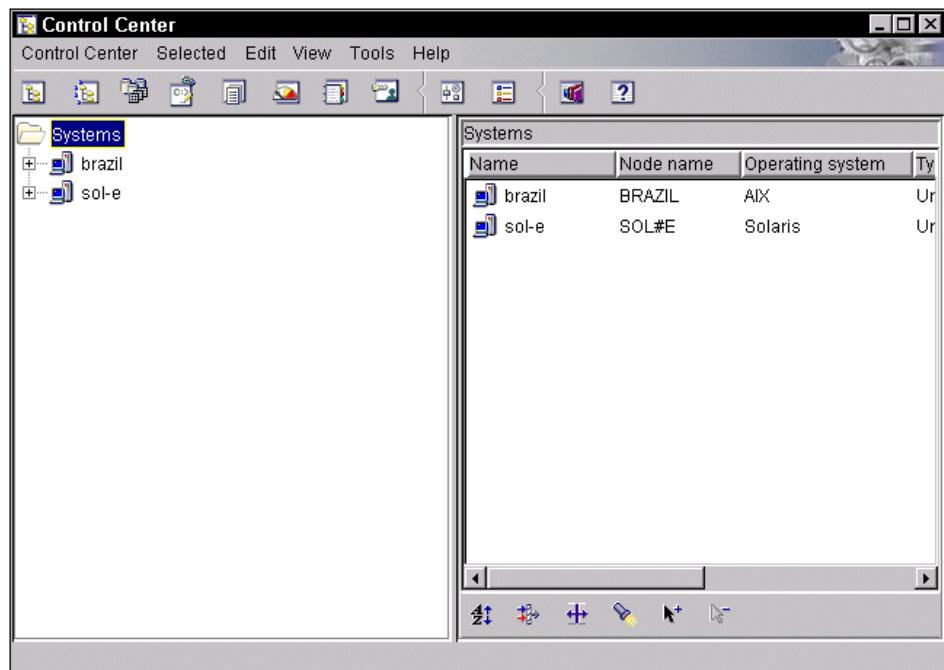


Figure 26. Control Center

2. Click the system where the database you want to back up resides. Since we are managing from a remote system, we are prompted as shown in Figure 27 with a dialog to enter the userid and password of the database administration server.

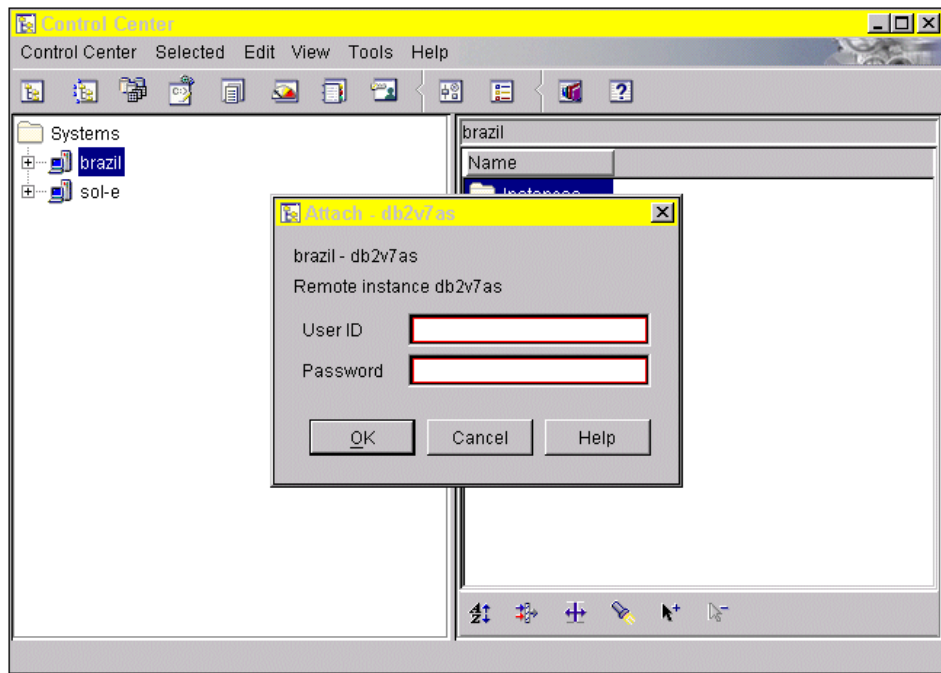


Figure 27. Attach to the administration server

Note that for most administration tasks, we attach to the database administration server instead of the database manager.

3. Click down the system until you see the database you want to back up. Right-click the database and select **Backup->Database** from the drop down menus as shown in Figure 28.

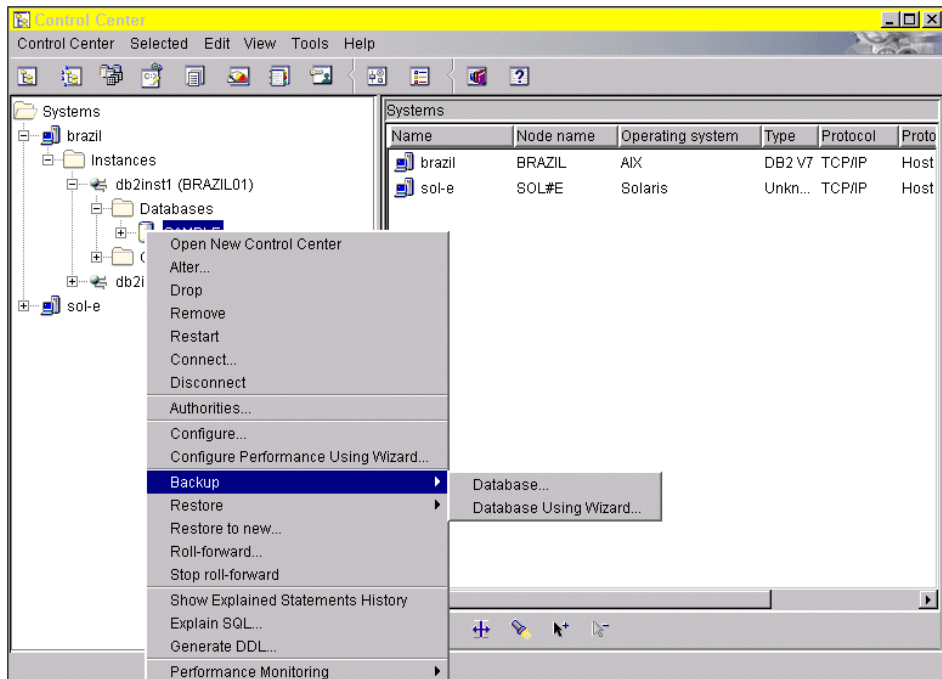


Figure 28. Drop down menu for database

- The backup database dialog appears. From the Media Manager drop down list, select **Tivoli Storage Manager**. Figure 29 shows the resulting dialog box.

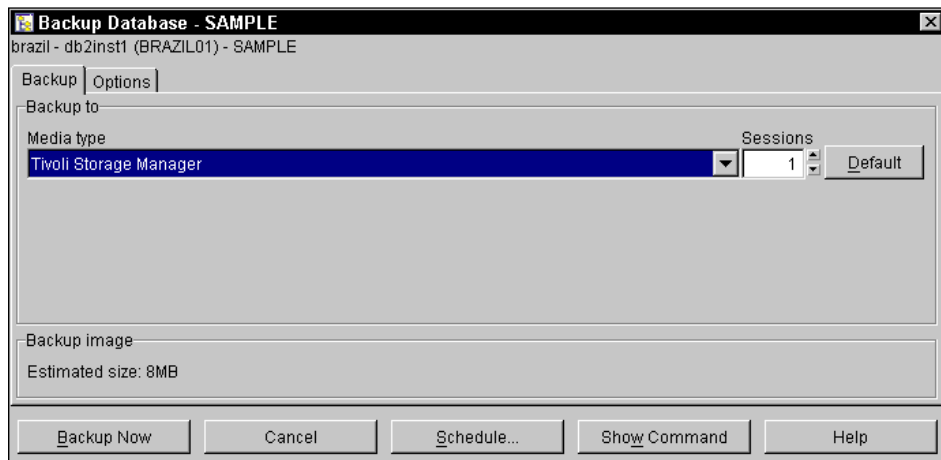


Figure 29. Backup database with Tivoli Storage Manager

5. Click on the Options page and select the Online radio button as shown in Figure 30.

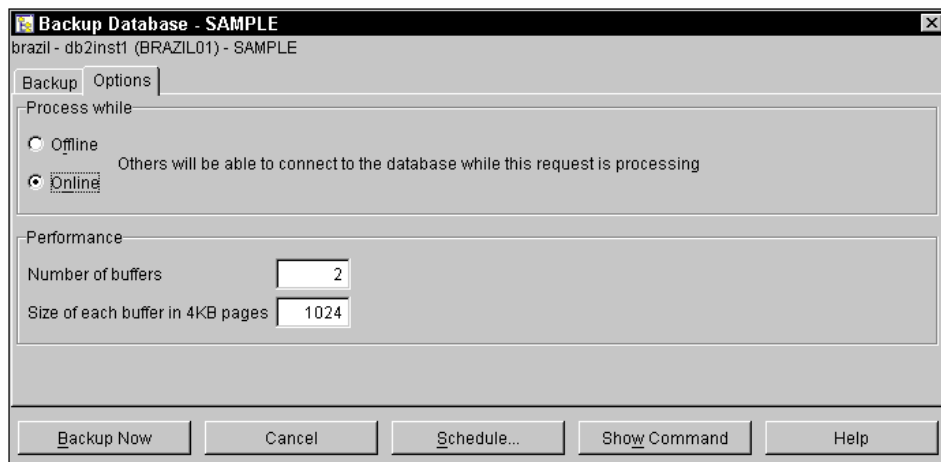


Figure 30. Backup using the online option

6. Select **Backup Now** to back up the database. When the backup completes, a confirmation dialog appears as shown in Figure 31. Close the dialog box.

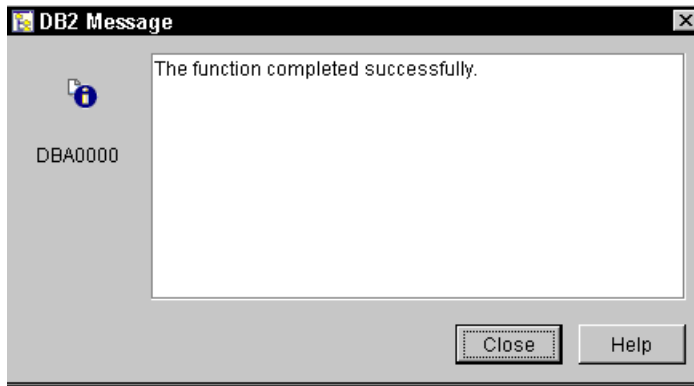


Figure 31. Backup successful dialog box

5.7.3 Tablespace backup

You have the option of doing a tablespace backup in certain situations where you do not want to do a full database backup. For example, you have just completed your nightly database backup when a user performs a large insert or load on important tables in a tablespace. The user wants the new data backed up. Since you do not want to repeat the database backup, you can do a tablespace backup. The next database backup will be done on your next nightly backup schedule.

You can perform tablespace backup either offline or online.

5.7.3.1 Preparatory step

You must enable roll-forward recovery in order to use online backup. See 5.4.2, “Enable the database for roll-forward recovery” on page 69.

5.7.3.2 Tablespace backup using the command line

This example shows an tablespace backup using the DB2 command line interface.

1. Log in as the database administrator or higher authority.
2. Back up the tablespace with the tsm option. The use tsm option tells the DBMS to use the Tivoli Storage Manager API to write the output backup file instead of using common devices. If you want to an online backup, use the online option. Assuming that you are backing up database SAMPLE tablespace USERSPACE1, the command and results should look like this:

```
$ db2 backup db sample tablespace userspace1 online use tsm

Backup successful. The timestamp for this backup image is : 20010312143618

$
```

5.7.3.3 Tablespace backup using the DB2 Control Center

For this example we will be using a Windows client to backup a tablespace from a remote database in the AIX platform.

1. Start the DB2 Control Center. For a Windows system, you will find the Control Center under **Start->Programs->IBM DB2->Control Center**. For command line or UNIX, type `db2cc`. Figure 32 shows the Control Center with systems that it can manage.

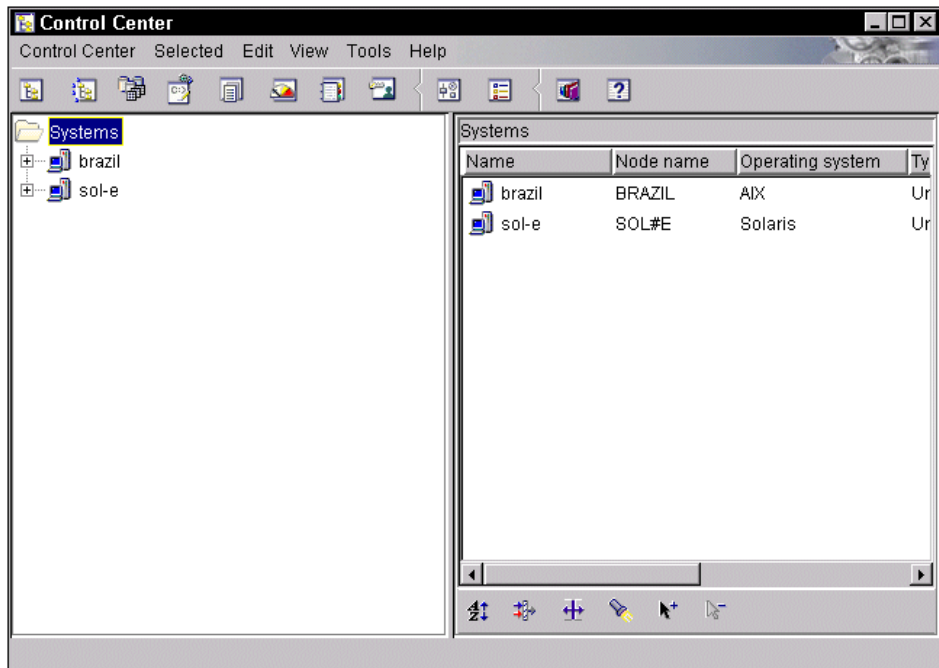


Figure 32. Control Center

2. Select the system where the database for which you want to do a tablespace backup resides. Since we are managing from a remote system, we are prompted as shown in Figure 33 with a dialog to enter the userid and password of the database administration server.

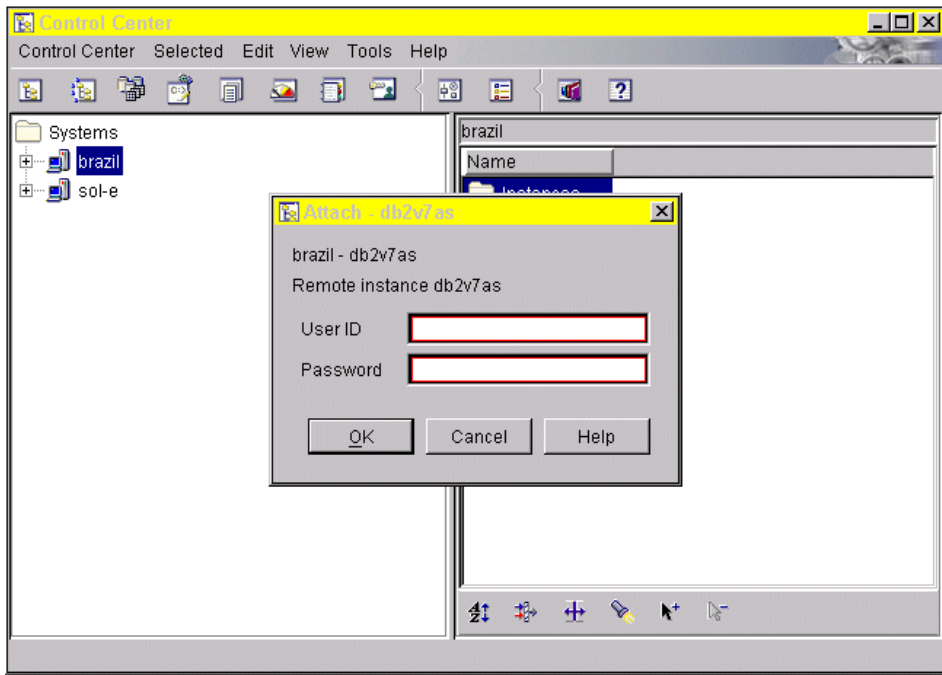


Figure 33. Attach to the administration server

3. Click down the system until you find the tablespace under the database tree which you want to back up. On the right panel select the tablespace you want to back up, right-click the tablespace and select **Backup** from the drop down menu as shown in Figure 34.

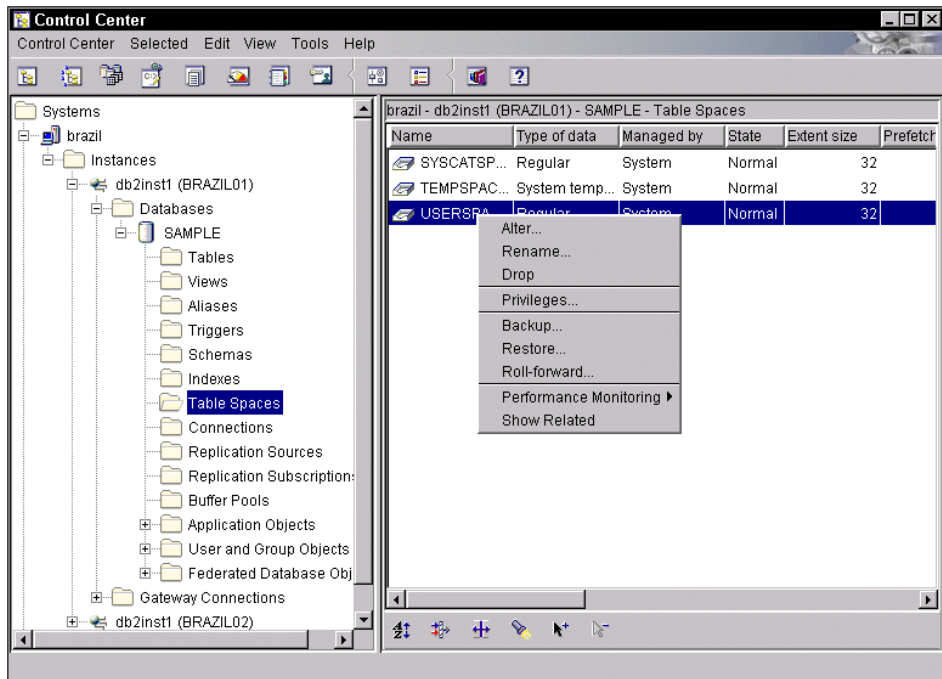


Figure 34. Drop down menu for a tablespace

Note that for most administration tasks, we login as the database administration server instead of the database manager.

4. The backup tablespace dialog appears. For the Media Manager drop down list, select **Tivoli Storage Manager**. Figure 35 shows the dialog box.

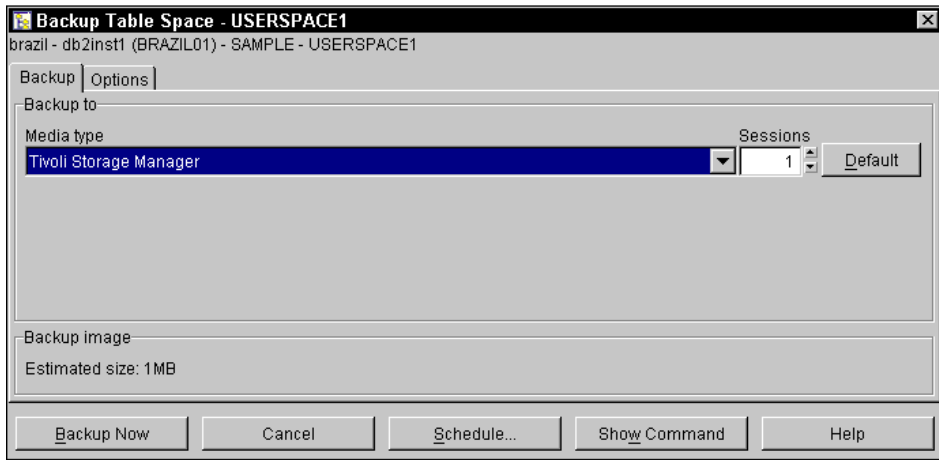


Figure 35. Backup tablespace using Tivoli Storage Manager

- Click on the Options page and select the Online radio button as shown in Figure 36, if you want an online backup.

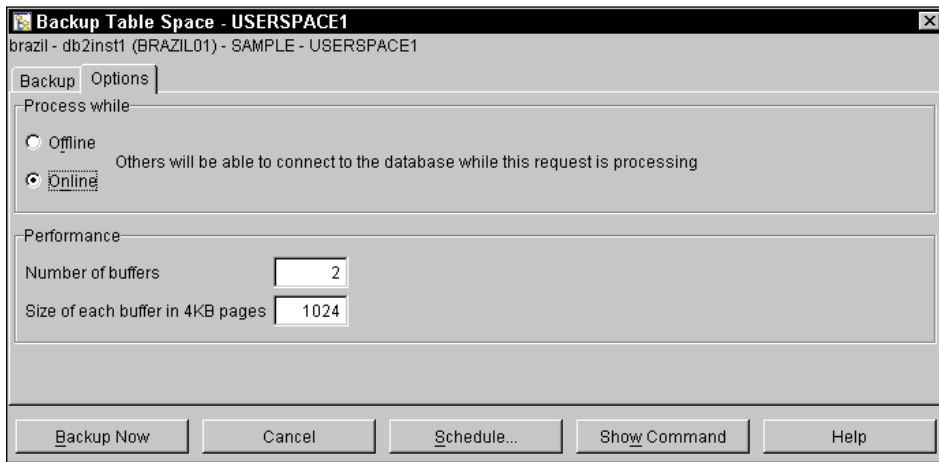


Figure 36. Backup tablespace using the online option

- Select **Backup Now** to back up the database. When the backup completes a confirmation dialog appears as shown in Figure 37. Close the dialog box.

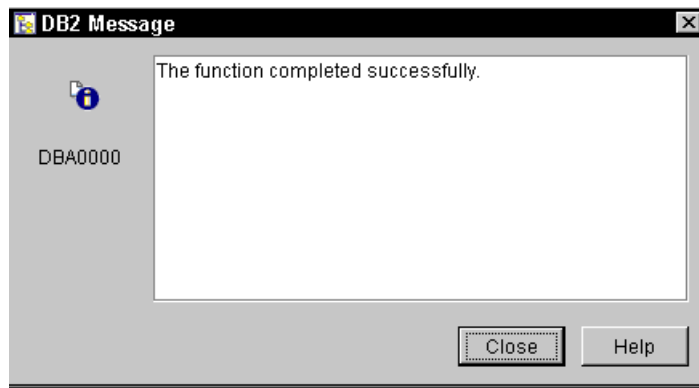


Figure 37. Backup successful dialog box

5.7.4 Load utility

In this section we will look at the `use tsm` option of the DB2 load utility.

Normally, every insert or update into a DB2 database will be logged in the logfiles. This will allow the database to be used in rollforward recovery mode. After restoring a full database backup, every transaction after that backup can be rolled forward.

The DB2 load utility doesn't log its activity in the DB2 logfiles. The advantage is that the DB2 load utility is able to load the data very fast into the DB2 database. The disadvantage is that, unless the `copy` option is specified, the database will not be available for rollforward recovery.

To allow rollforward recovery when data is loaded via the DB2 load utility the `copy yes` option of the load utility *must* be specified. This will generate a copy of the loaded data that will be saved to a specified location. In our example, we will save this copy to Tivoli Storage Manager server using the `use tsm` option.


```

$ db2 load from data.load of del insert into org copy yes use tsm

SQL3109N The utility is beginning to load data from file
"/home/db2inst1/tsm/load/data.load".

SQL3500W The utility is beginning the "LOAD" phase at time "03-27-2001
14:15:19.565784".

SQL3519W Begin Load Consistency Point. Input record count = "0".

SQL3520W Load Consistency Point was successful.

SQL3110N The utility has completed processing. "900" rows were read from the
input file.

SQL3519W Begin Load Consistency Point. Input record count = "900".

SQL3520W Load Consistency Point was successful.

SQL3515W The utility has finished the "LOAD" phase at time "03-27-2001
14:15:20.397451".

Number of rows read          = 900
Number of rows skipped       = 0
Number of rows loaded        = 900
Number of rows rejected      = 0
Number of rows deleted       = 0
Number of rows committed    = 900

```

This will send a copy of the data loaded to the DB2 database also to the Tivoli Storage Manager server. (The data will go into the backup copy group of the associated management class).

This data will be downloaded again when the DB2 rollforward command calls for it.

Chapter 6. Backing up DB2 UDB on the Sun Solaris platform

This chapter will discuss how to prepare the database to use Tivoli Storage Manager on the Sun Solaris platform, and how to perform backup using Tivoli Storage Manager as a media type.

Preparing the database to use Tivoli Storage Manager involves:

- Installation of Tivoli Storage Manager related software
- Configuration and setup of Tivoli Storage Manager client API
- Setting up the DB2 user exit for Tivoli Storage Manager

The types of backups discussed are:

- Offline database backup
- Online database backup
- Tablespace backup

Some examples of backup using command line and using the Control Center are given. This chapter only covers the backups using Tivoli Storage Manager. For other options, please see the DB2 documentation. Backup options for performance will be discussed in Appendix C, "Performance" on page 285.

6.1 Installation of Tivoli Storage Manager related software

For DB2 UDB Version 7.1, you must use Tivoli Storage Manager Client Version 3 or later. You can download the latest Tivoli Storage Manager client software from the IBM software ftp site:

```
ftp://ftp.software.ibm.com/storage/tivoli-storage-management/maintenance/client
```

At the time of writing, the latest version and patch level of the Tivoli Storage Manager client is 4.1.2.12. This client version should work with Tivoli Storage Manager server Version 3.7 and higher.

Note

You should not use Tivoli Storage Manager client software Version 4.1.2.00, because there is a problem when using the db2adutl utility. You should use either 4.1.1 or 4.1.2.12. In our lab, we used 4.1.2.12.

DB2 only requires the Tivoli Storage Manager client API. If you want to install the client Backup-Archive (baclient) and the documentation, please see the README for prerequisites and installation instructions. The Backup-Archive

client can be useful when backing up non-database files, and for using the `dsmc` utility.

The following sections will discuss the requirements and procedures for installing and configuring the Tivoli Storage Manager client API Version 4.1.2.

6.1.1 System requirements

The following are the current system requirements for the Tivoli Storage Manager clients on the Sun Solaris platform.

6.1.1.1 Software requirements

For the Application Programming Interface (API), you must have:

- Sun Solaris 2.6 (32 bit) or Sun Solaris 7 (32 or 64 bit) or Sun Solaris 8 (32 or 64 bit)
- For Sun Solaris 7, 64 bit, you must have applied the following patches which can be downloaded from [sunsolve.Sun.COM](http://sunsolve.sun.com):
 - 106327-05
 - 106300-06
- A C Compiler, if you want to compile a user-exit. See 6.3.1, “Compile the user exit for Tivoli Storage Manager” on page 97.

6.1.1.2 Hardware requirements

The Tivoli Storage Manager client API requires:

- 32MB of RAM
- 15MB of disk space

6.1.2 Installing the Tivoli Storage Manager client

We installed the Tivoli Storage Manager client using the following steps:

1. Delete the previous version of Tivoli Storage Manager or ADSM API as root.
 - a. To determine the Tivoli Storage Manager or ADSM package installed on the machine, do the following command:

```
# pkginfo | grep Tivoli
```

```
# pkginfo | grep ADSM
```

The result could be something like this:

```
# pkginfo | grep Tivoli
application TDPoracle      Tivoli Data Protection for Oracle 32-bit
application TDPorcllic    Tivoli Data Protection for Oracle License
application TIVsmCapi     Tivoli Storage Manager Solaris 2.6/7/8 API
application TIVsmCba     Tivoli Storage Manager Solaris 2.6/7/8 Client
application TIVsmCdoc     Tivoli Storage Manager Solaris 2.6/7/8 Documentation
#
```

- b. We are only concerned with the API package. Query the version of the package using the `pkginfo` command, for example:

```
# pkginfo -l TIVsmCapi
```

- c. If the API package is from a previous version, uninstall the package using the `pkgrm` command, for example:

```
# pkgrm TIVsmCapi
```

2. The `/etc/system` must have, as a minimum, the following values. Modify them as necessary, then reboot.

```
set lwp_default_stksize = 0x4000
set rpcmod:svc_run_stksize = 0x4000
set semsys:seminfo_semmap = 50
set semsys:seminfo_semmni = 50
set semsys:seminfo_semmns = 300
set semsys:seminfo_semmnu = 150
set semsys:seminfo_semopm = 50
set semsys:seminfo_semume = 50
set semsys:seminfo_semmnl = 125
```

3. Install the Tivoli Storage Manager package. If you download the latest client replace the pathname to the path where you downloaded the software.

```
# pkgadd -d /cdrom/tsmcli/Solaris TIVsmCapi
```

6.2 Configuration and setup of Tivoli Storage Manager client API

After you have successfully installed the Tivoli Storage Manager client software, you need to configure the client to communicate to the Tivoli Storage Manager server. In this section, we will only cover how to setup the communication. We assume that the configuration on the server is done and valid, prior to configuring the client. (For example, the node has a backup and archive destination, and enough space to hold the backups).

6.2.1 Setup prerequisites

Here are the setup prerequisites you need to follow:

- You need to have the following information prior to begin to configure the client. Talk to your Tivoli Storage Manager administrator to get the required information:
 - Tivoli Storage Manager server name and IP-address
 - The node name under which the API client is known to the Tivoli Storage Manager server
 - The password of the node
- Verify that there is a symbolic link in directory /usr/lib for libApiDS.so, by issuing the following command:

```
# cd /usr/lib
# ls -l libApiDS.so
lrwxrwxrwx 1 root  other      42 Mar  5 14:29 libApiDS.so -> /opt/tivoli
/tsm/client/api/bin/libApiDS.so
#
```

- If the symbolic link does not exist, create one:

```
# ln -s libApiDS.so /opt/tivoli/tsm/client/api/bin/libApiDS.so
```

6.2.2 Define the environment variables

Log in as the instance owner, define the following environment variables in your \$HOME/.profile. In DB2 UDB 7.1, you can also define them in \$HOME/sql/lib/userprofile. We do not recommend for you to define the environment variables in sql/lib/db2profile, because they can be overridden by fixpaks.

- **DSMI_DIR**
Identifies the directory path where `dsm.sys`, `dsmtca`, and the subdirectory `en_US` is located. The `en_US` subdirectory must contain the `dsmclientV3.cat` file.
- **DSMI_CONFIG**
Identifies the full directory path and file name of the Tivoli Storage Manager user options file `dsm.opt`. This file contains the name of the server to be used.
- **DSMI_LOG**
Identifies the directory path where the error log file, `dserror.log`, is to be created.

A sample `.profile` file may look like this:

```
#      This is the default standard profile provided to a user.
#      They are expected to edit it to meet their own needs.

MAIL=/usr/mail/${LOGNAME:?}

# The following three lines have been added by UDB DB2.
if [ -f sqllib/db2profile ]; then
    . sqllib/db2profile
fi

DSMI_DIR=/opt/tivoli/tsm/client/api/bin
DSMI_CONFIG=$HOME/tsm/dsm.opt
DSMI_LOG=$HOME/tsm
export DSMI_DIR DSMI_CONFIG DSMI_LOG
```

6.2.3 Create the Tivoli Storage Manager configuration files

To configure the Tivoli Storage Manager client, do the following steps:

1. Create the `dsm.sys` file

As root, create or modify the Tivoli Storage Manager system options file, `dsm.sys`, which must be located in the `/opt/tivoli/tsm/client/api/bin` directory.

The sample `dsm.sys.smp` in the `/opt/tivoli/tsm/client/api/bin` directory provides the basic information needed for DB2 to work. The `passwordaccess` parameter must be set to `generate`. This parameter specifies that Tivoli Storage Manager encrypts and stores the user password locally and generates a new password when the old one expires. Then DB2 is not required to supply a password each time it

initializes a session with the Tivoli Storage Manager server. A sample `dsm.sys` file may look like this:

```
SERVERNAME          brazil-db2
COMMETHOD         TCPip
TCPSPORT           1500
TCPSEVERADDRESS    193.1.1.11
NODENAME           sol-db2
PASSWORDACCESS     generate
```

2. Create the `dsm.opt` file

The instance owner can create or modify `dsm.opt`. This file must be located in the directory specified by your `DSMI_CONFIG` environment variable.

The `dsm.opt` file only need to have one line in it which is a reference to the Server Stanza in the `dsm.sys` file. The Name given here is only a symbolic one and does not need to match the name of the Tivoli Storage Manager server. It is possible to have two or more API clients on the same system (for example, two instances) with different server characteristics. Each client has its own `dsm.opt` file pointing to a different stanza in the `dsm.sys` file.

Here is a sample `dsm.opt` file:

```
SErvername          brazil-db2
```

3. Set the Tivoli Storage Manager password

Each Tivoli Storage Manager client must have a password to access a server. For that reason, the root user of your system must run the executable file, `dsmapipw`, installed in the `$HOME/instance/sqllib/adsm` directory, to establish and reset the Tivoli Storage Manager password. If you are using multiple Tivoli Storage Manager servers from this machine, make sure that root has the above environment variables set correctly. When executed, the `dsmapipw` program prompts you for the:

- *Old password*, which is the current password for the Tivoli Storage Manager node stored in the server.
- *New password*, which is the new password for the node.

You may check that there will be a new file created in the `/etc/adsm` directory that contains the encrypted password. The name of this file is the same as the `servername` entry in the `dsm.opt` file.

4. Login as the instance owner for the new environment variables Tivoli Storage Manager environment variables in `$HOME/profile` or `$HOME/sqllib/userprofile` to take effect.
5. Start and stop DB2.

```
$ db2stop  
$ db2start
```

6.3 Setting up the DB2 user exit for Tivoli Storage Manager

By default, DB2 reuses logs in a circular fashion and are not archived. This will only allow you to do offline backup and version recovery. You can enable archive logging for the database by setting *logretain* or *userexit* to On. You can set them both if you want. When one of these is set, the database is enabled for roll-forward recovery, and you can perform an online backup of the database and tablespaces.

If your database is required to be up 24 x 7, you should enable roll-forward recovery by setting *logretain* or *userexit* to On, so you can do online backups.

The following sections will describe enabling roll-forward recovery by using the supplied user exit for Tivoli Storage Manager.

6.3.1 Compile the user exit for Tivoli Storage Manager

A user exit is an executable file that the DBMS calls for archive and retrieval of log files. Sample user exits are supplied in `$HOME/sqllib/samples/c` of the instance owner. At the beginning of each sample file is a good description on how DB2 uses the specific user exit and which parameter you may set. For Tivoli Storage Manager, the user exit sample is `db2uext2.cadsm`. Use the following steps to create the user exit executable.

1. Become root, and copy the `db2uext2.cadsm` to your working directory and rename it to `db2uext2.c`. Change file ownership and permissions so the instance owner can modify the code.
2. Read the comments at the beginning of the source code to understand the defined variables. Change the defined variables to suit your environment. In our lab, we took the default setting and only change the path of the user exit logfiles (`AUDIT_ERROR_PATH`). Our settings are as follows:

```

#define BUFFER_SIZE          4096      /* transmit or receive the log */
                                /* file in 4k portions */
#define AUDIT_ACTIVE         1        /* enable audit trail logging */
#define ERROR_ACTIVE        1        /* enable error trail logging */
#define AUDIT_ERROR_PATH    "/export/home/db2inst1/tsm/" /* path must en
d with a slash */
#define AUDIT_ERROR_ATTR    "a"      /* append to text file */

```

3. As the instance owner, compile the C program. In our lab, we used the GNU C Compiler gcc. Check your compiler for the correct syntax.

```
$ gcc -I/opt/tivoli/tsm/client/api/bin/samples -L/usr/lib -lApiDS -o db2uext2 db2uext2.c
```

4. Copy the db2uext2 file into the \$HOME/sql/lib/adm directory of the instance owner. We do not recommend for you to copy the file to the \$HOME/sql/lib/bin directory, because this directory is only a link to a directory in the /opt/IBMDB2/V7.1/bin directory. Make sure the instance owner has execute permission for this file.

6.3.2 Enable the database for roll-forward recovery

Follow the steps to enable roll-forward recovery:

1. Update the userexit parameter of the database configuration file. If, for example, the database name is SAMPLE, the command would be:

```
$ db2 update db cfg for sample using userexit on
```

The changes will not take effect until all applications disconnect from the database. If the `activate database` command was issued, the `deactivate database` command must be issued. When all applications disconnect from the database, the database will be in backup pending state.

2. When all applications disconnect from the database and the database is in backup pending state, no connections are allowed until a backup is completed. Perform a backup of the database. For more information, see 6.5.1, “Full offline backup” on page 101 for a detailed discussion on offline backups.

```
$ db2 backup db sample use tsm
```

After the backup, the database is ready for use and will be able to archive logs to Tivoli Storage Manager.

6.4 Using Tivoli Storage Manager baclient in conjunction with DB2

In this section we show how Tivoli Storage Manager Backup-Archive client can be used to back up a DB2 database at a filesystem level and also how to back up other DB2 relevant files. We assume that a Tivoli Storage Manager Backup-Archive client is already installed and configured.

Beside the normal DB2 database backup cycle, some operating system files need to be backed up to be able to restore the surrounding environment for the DB2 database. Those files normally reside in the home directory of the instance owner. Some examples of these files are:

- The *.profile*, *db2profile*, *userprofile* of the instance owner
- Additional shell scripts or DB2 SQL scripts
- Tivoli Storage Manager config files (*dsm.opt*, *dsm.sys*, include-exclude list, and so on)
- The userexit program *db2uext2* and the modified source code

On the other hand there is no need to backup the DB2 datafiles where the DB2 database is built on. These datafiles will be backed up by the DB2 `backup` command. These files should be excluded from the list of files that will be backed up by the Tivoli Storage Manager Backup-Archive client.

The best place to define which files should be backed up by the Tivoli Storage Manager Backup-Archive client is the include-exclude file of the Tivoli Storage Manager Backup-Archive client. This file will be checked from bottom to top for every file that is being backed up by the Tivoli Storage Manager Backup-Archive client. The first match of an include or exclude line decides if the file will be backed up or not.

In our example, we have databases that reside in the `/home/db2inst1/db2inst1` and the `/db2db` directory. In the next example, we only show the additional entries in the existing include-exclude file in order that file in the `/home/db2inst1` directory will be backed up and files that reside in the `/home/db2inst1/db2inst1` and `/db2db` directory are not.

```
INCLUDE /home/db2inst1/.../*
EXCLUDE /home/db2inst1/db2inst1/.../*
EXCLUDE.FS /db2db/.../*
```

Figure 38. Tivoli Storage Manager Backup-Archive include-exclude file example

Note

The EXCLUDE.FS statement will take effect no matter where it is placed in the include-exclude list file. This means that it will overrule every INCLUDE statement that is related to the same filesystem. For example, an EXCLUDE.FS /home will overrule an INCLUDE /home/db2inst1/.../*, no matter where it is placed within the include-exclude list file.

If there is a need to backup the DB2 datafiles where the DB2 database is built on, the archive option of the Tivoli Storage Manager Backup-Archive client should be used.

The archive option will create an archived set of files in the archive copy destination of the related management class at the Tivoli Storage Manager server and it will NOT read the include-exclude list file. It will back up all files specified, no matter if they have changed or not since the last archive.

A description should be assigned to each archive set in order to more easily retrieve it. The database needs to be stopped or suspended to have a consistent state during backup. The command shown below will make an archive backup of all files that are under the /db2db/ directory.

```
# dsmc archive -subdir=yes -desc='DB2 file archive' /db2db/
```

To restore the archive again the retrieve option must be specified. See following example.

```
dsmc retrieve /db2db/ -desc='DB2 file archive'
```

6.5 Backing up DB2 using Tivoli Storage Manager

In this section, we examine how you can use Tivoli Storage Manager to back up DB2 databases. We cover:

- Offline backup
- Online backup
- Tablespace backup
- Load utility

6.5.1 Full offline backup

Please note that while the offline backup is ongoing, users cannot connect to the database. If you require users to have 24 x 7 availability, see 6.5.2, “Online database backup” on page 107.

The DB2 backup utility can be used from either:

- Command line interface
- Graphical user interface
- Your own C, COBOL, or Fortran language program

Our examples use the command line and graphical user interfaces.

6.5.1.1 Preparatory steps

Before you can start a backup or recovery operation, make sure that the database manager is up and running. You can issue the `db2start` command to start the database manager. If it already running, it will tell you that it is already active.

```
$ db2start
SQL1026N The database manager is already active.
$
```

Also make sure that the Tivoli Storage Manager server is online and policies are in place.

6.5.1.2 Offline backup using the command line

This example shows an offline backup using the DB2 command line interface.

1. Log in as the database administrator or higher authority and make sure that all applications are logged off from the database you want to back up. Use the following DB2 command to verify that all applications are logged off. Assuming you are backing up the SAMPLE database, the command and output may look like this:

```

$ db2 list applications for db sample

Auth Id  Application      Appl.      Application Id          DB      # of
   Name                                     Handle                                     Name    Agents
-----
DB2INST1 db2bp           6          *LOCAL.db2inst1.010312180936  SAMPLE  1

$

```

2. Logoff all applications connected to the database. In the above results, there is one application using the database. Use the force command enumerating the application handles (separated with commas) inside the parenthesis:

```

$ db2 "force application ( 6 )"
DB20000I The FORCE APPLICATION command completed successfully.
DB21024I This command is asynchronous and may not be effective immediately.

$

```

Note that you can use the command `force application all` to log off all applications if there are too many. However, be careful when using this command, because it will also logoff applications connected to other databases within the instance.

3. Verify that there are no more user connected to the database by issuing again the `list application` command.
4. Backup the database with the `tsm` option. The use `tsm` option tells the DBMS to use the Tivoli Storage Manager API to write the output backup file instead of using common devices. A confirmation will appear to indicate that the backup is successful.

```

$ db2 backup db sample use tsm

Backup successful. The timestamp for this backup image is : 20010312104622

$

```

6.5.1.3 Offline backup using the DB2 Control Center

For this example, we will be using a Windows client to backup a remote database in the Solaris platform.

1. Start the DB2 Control Center. For a Windows system, you will find the Control Center under **Start->Programs->IBM DB2->Control Center**. For

command line or UNIX, type `db2cc`. Figure 39 shows the Control Center with system that it can manage.

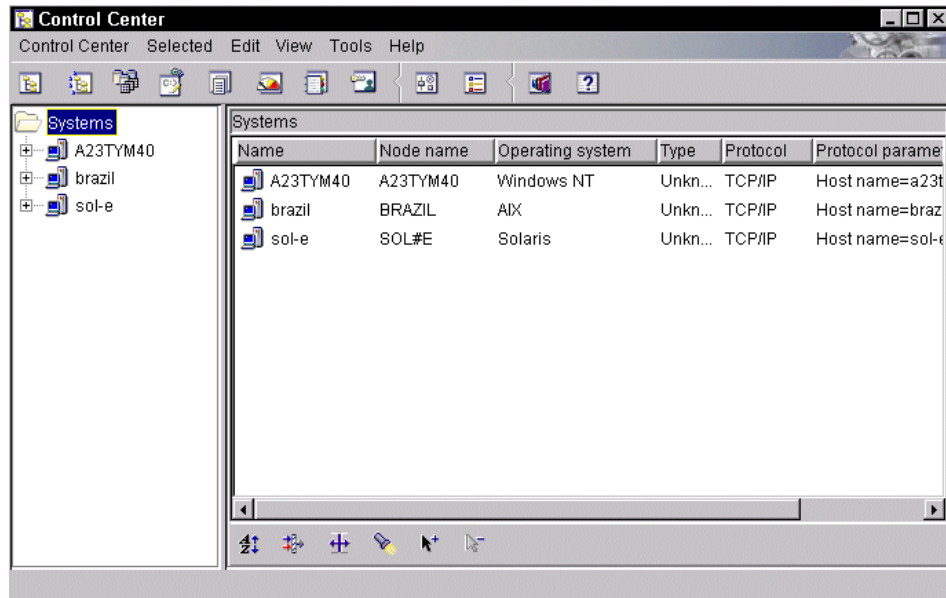


Figure 39. Control Center

2. Click the system where the database you want to backup resides. Since we are managing from a remote system, we are prompted as shown in Figure 40 with a dialog to enter the userid and password of the database administration server.

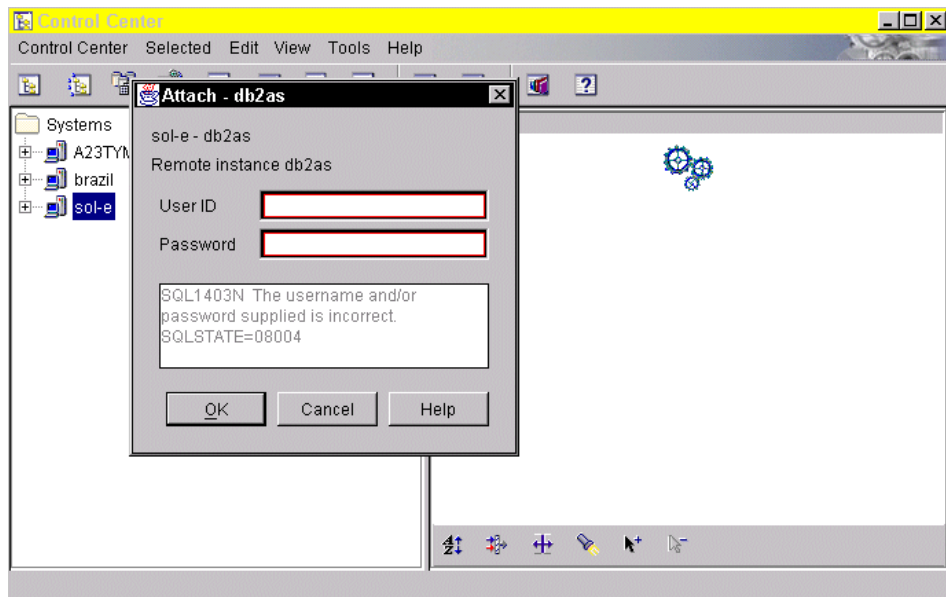


Figure 40. Attach to the administration server

Note that for most administration tasks, we attach to the database administration server instead of the database manager.

3. Click down the system until you see the database you want to backup. Right-click the database and select **Backup->Database** from the drop down menus as shown in Figure 41.

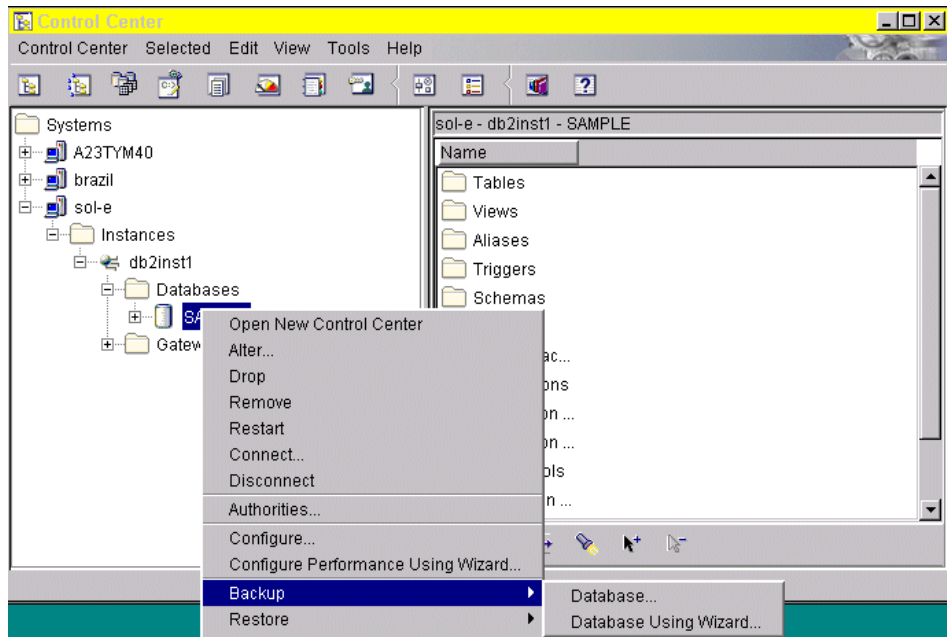


Figure 41. Drop down menu for database

Note that for most administration tasks, we attach to the database administration server instead of the database manager.

4. The backup database dialog appears. For the Media Manager drop down list, select **Tivoli Storage Manager**. Figure 42 shows the dialog box.

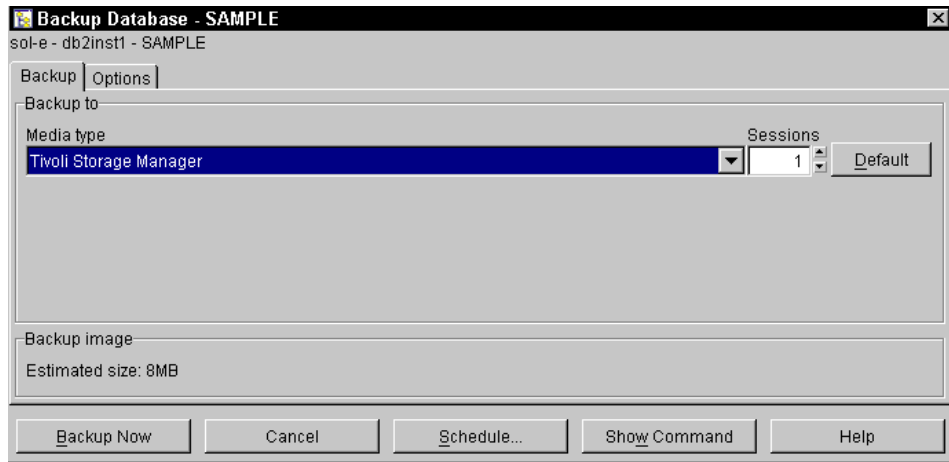


Figure 42. Backup database with Tivoli Storage Manager

5. Select **Backup Now** to backup the database. When the backup completes, a confirmation dialog appears as shown in Figure 43. Close the dialog box.

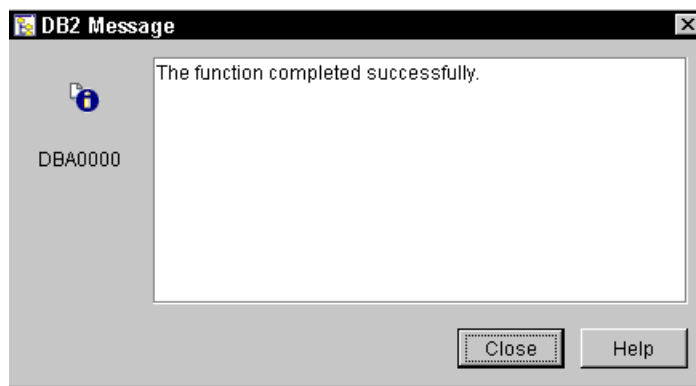


Figure 43. Backup successful dialog box

The DB2 Journal will now provide additional information about the success of the backup. The DB2 Journal is in the Tools menu of the DB2 Control Center. We will discuss the Journal in more detail in 7.5.2, “Automate DB2 backup using DB2” on page 135.

6.5.2 Online database backup

With an online backup, users can use the database while the backup is ongoing.

6.5.2.1 Preparatory steps

You must enable roll-forward recovery to use online backup. See 6.3, “Setting up the DB2 user exit for Tivoli Storage Manager” on page 97.

6.5.2.2 Online backup using the command line

This example shows an online backup using the DB2 command line interface.

1. Log in as the database administrator or higher authority.
2. Backup the database with the tsm option. The use tsm option tells the DBMS to use the Tivoli Storage Manager API to write the output backup file instead of using common devices. Assuming that you are backing up the database SAMPLE, the command and results may look like this:

```
$ db2 backup db sample online use tsm  
  
Backup successful. The timestamp for this backup image is : 20010312132637  
  
$
```

6.5.2.3 Online backup using the DB2 Control Center

For this example, we will be using a Windows client to backup a remote database in the Solaris platform.

1. Start the DB2 Control Center. For a Windows system, you will find the Control Center under **Start->Programs->IBM DB2->Control Center**. For command line or UNIX, type `db2cc`. Figure 44 shows the Control Center with system that it can manage.

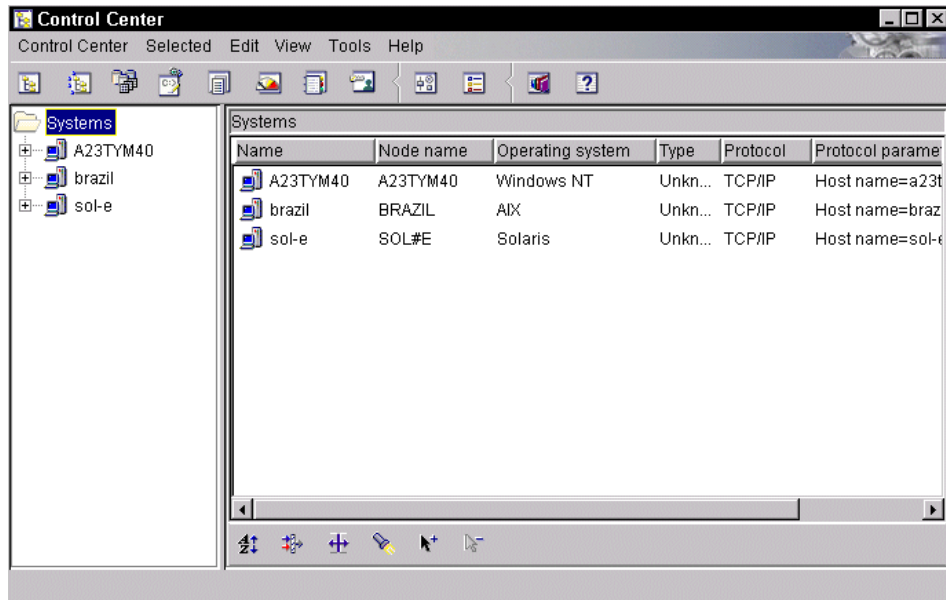


Figure 44. Control Center

2. Click the system where the database you want to backup resides. Since we are managing from a remote system, we are prompted as shown in Figure 45 with a dialog to enter the userid and password of the database administration server.

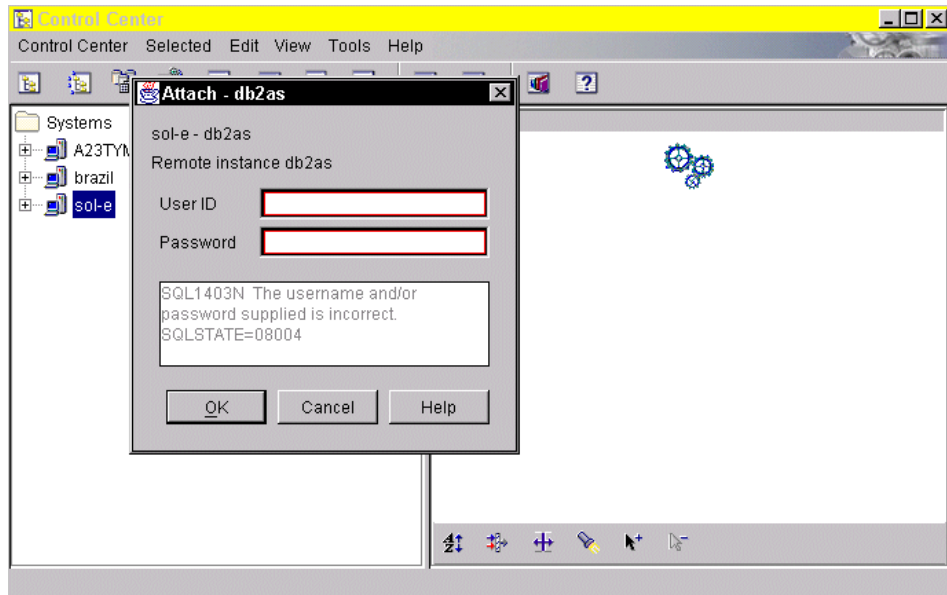


Figure 45. Attach to the administration server

Note that for most administration tasks, we attach to the database administration server instead of the database manager.

3. Click down the system until you see the database you want to backup. Right-click the database and select **Backup->Database** from the drop down menus as shown in Figure 46.

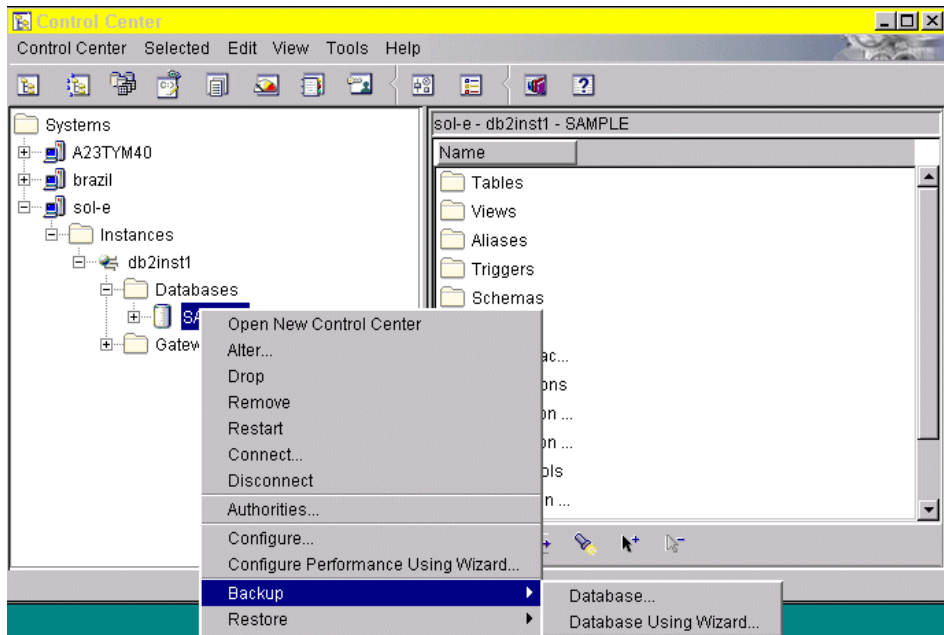


Figure 46. Drop down menu for database

4. The backup database dialog appears. For the Media Manager drop down list, select **Tivoli Storage Manager**. Figure 47 shows the dialog box.

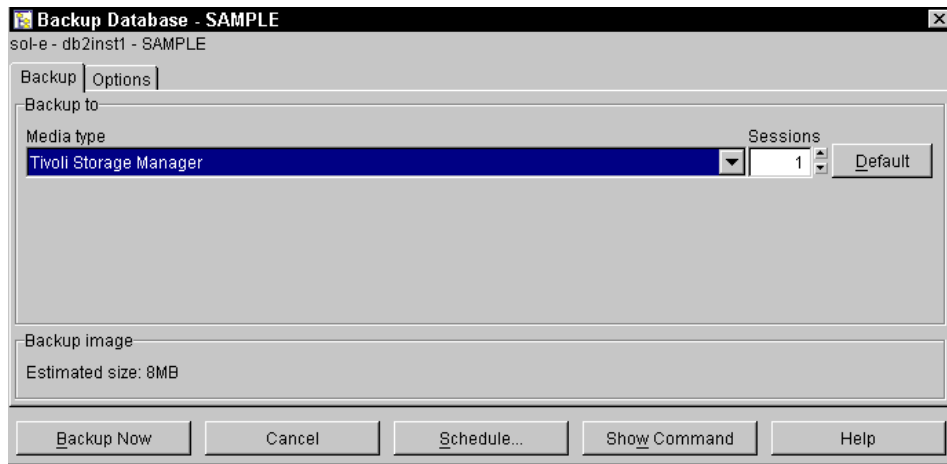


Figure 47. Backup database with Tivoli Storage Manager

- Click on the Options page and select the Online radio button as shown in Figure 48.

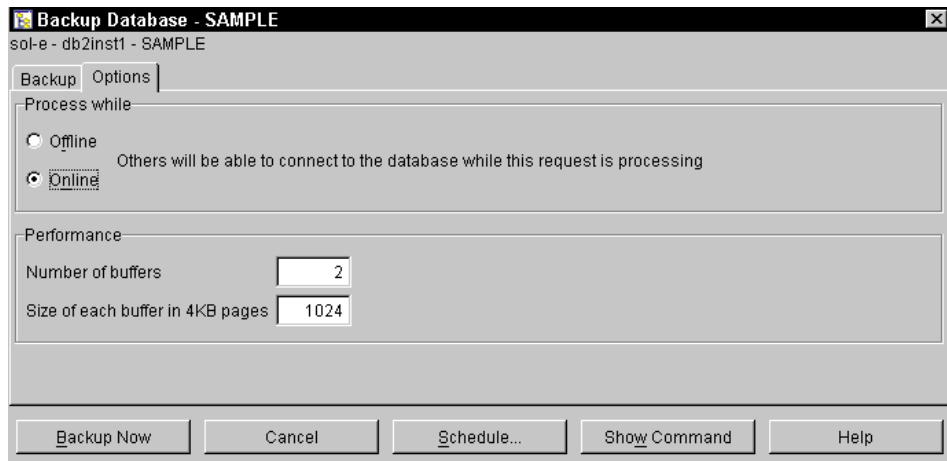


Figure 48. Backup using the online option

- Select **Backup Now** to backup the database. When the backup completes, a confirmation dialog appears as shown in Figure 49. Close the dialog box.

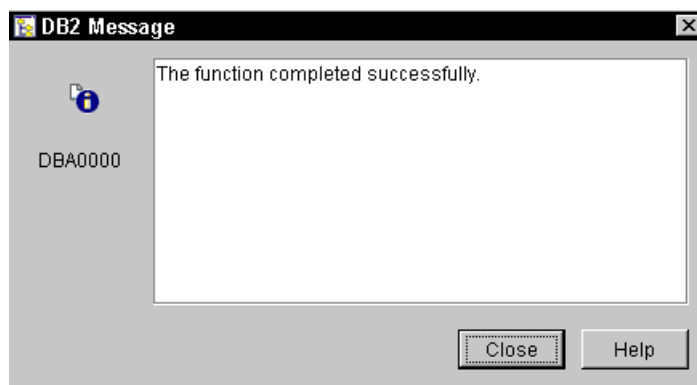


Figure 49. Backup successful dialog box

6.5.3 Tablespace backup

You have the option of doing a tablespace backup in certain situations where you do not want to do a full database backup. For example, you just completed your nightly database backup, then a user performs a large insert

or load on important tables on a tablespace. The user wants the new data backed up. Since you do not want to repeat the database backup, you can do a tablespace backup. The next database backup will be done on your next nightly backup schedule.

You can perform a tablespace backup either offline or online.

6.5.3.1 Preparatory step

You must enable roll-forward recovery to use online backup. See 6.3, “Setting up the DB2 user exit for Tivoli Storage Manager” on page 97.

6.5.3.2 Tablespace backup using the command line

This example shows an tablespace backup using the DB2 command line interface.

1. Log in as the database administrator or higher authority.
2. Backup the tablespace with the tsm option. The use tsm option tells the DBMS to use the Tivoli Storage Manager API to write the output backup file instead of using common devices. If you want to an online backup, use the online option. Assuming that you are backing up a database SAMPLE tablespace USERSPACE1, the command and results may look like this:

```
$ db2 backup db sample tablespace userspace1 online use tsm
Backup successful. The timestamp for this backup image is : 20010312143618
$
```

6.5.3.3 Tablespace backup using the DB2 Control Center

For this example, we will be using a Windows client to backup a tablespace from a remote database in the Solaris platform.

1. Start the DB2 Control Center. For a Windows system, you will find the Control Center under **Start->Programs->IBM DB2->Control Center**. For command line or UNIX, type `db2cc`. Figure 50 shows the Control Center with systems that it can manage.

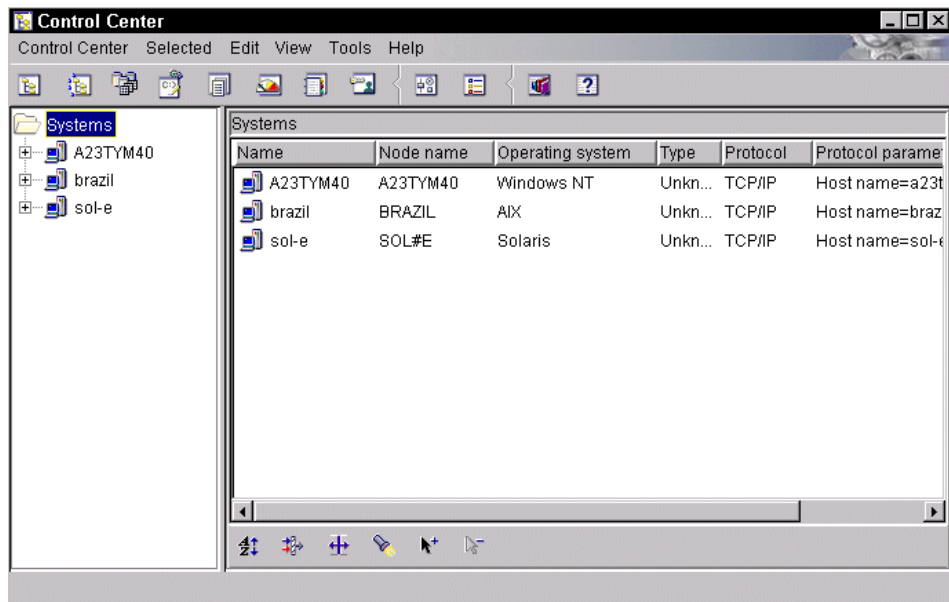


Figure 50. Control Center

2. Click the system where the database you want to do a tablespace backup resides. Since we are managing from a remote system, we are prompted as shown in Figure 51 with a dialog to enter the userid and password of the database administration server.

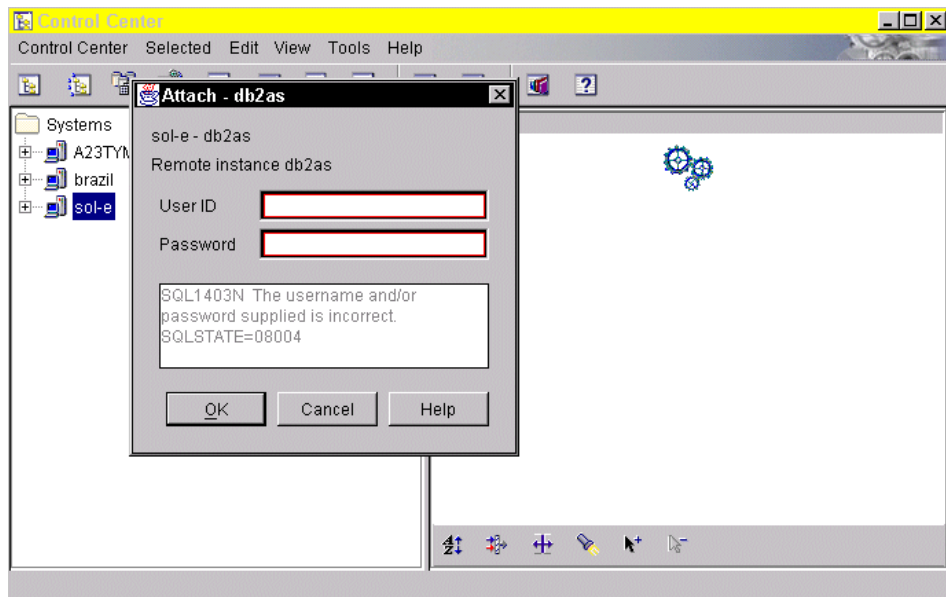


Figure 51. Attach to the administration server

3. Click down the system until you click the tablespace under the database tree you want to backup. On the right pane select the tablespace you want to backup, right-click the tablespace and select **Backup** from the drop down menu as shown in Figure 52.

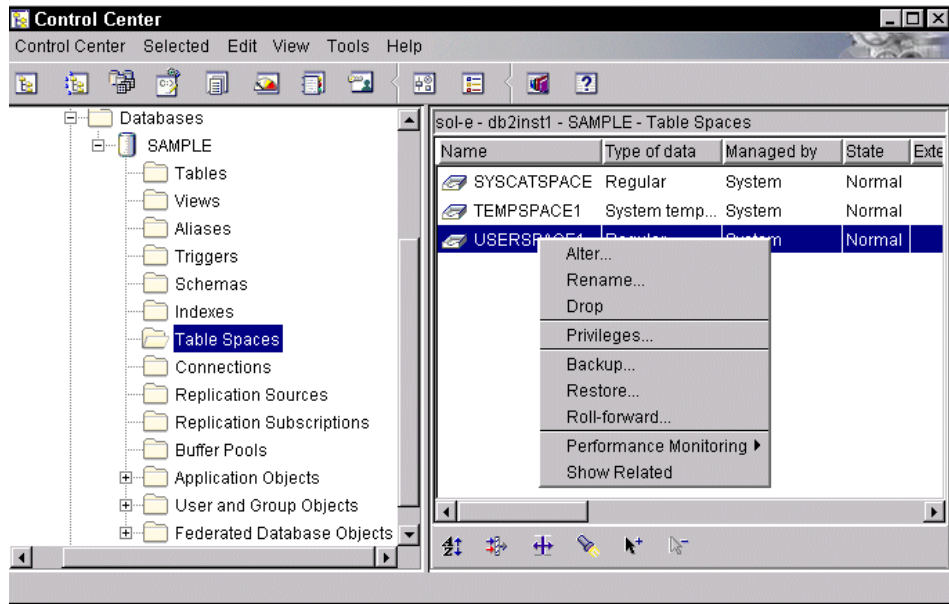


Figure 52. Drop down menu for a tablespace

Note that for most administration tasks, we attach to the database administration server instead of the database manager.

4. The backup tablespace dialog appears. For the Media Manager drop down list, select **Tivoli Storage Manager**. Figure 53 shows the dialog box.

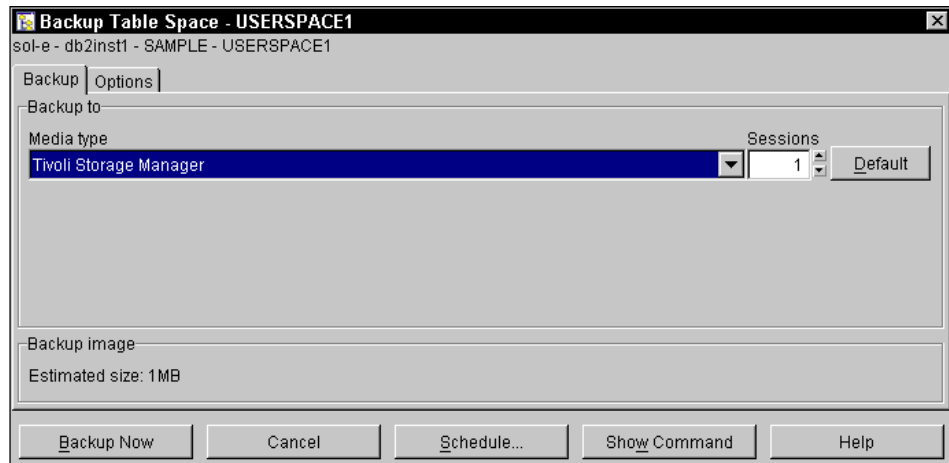


Figure 53. Backup tablespace using Tivoli Storage Manager

- Click on the Options page and select the Online radio button as shown in Figure 54, if you want an online backup.

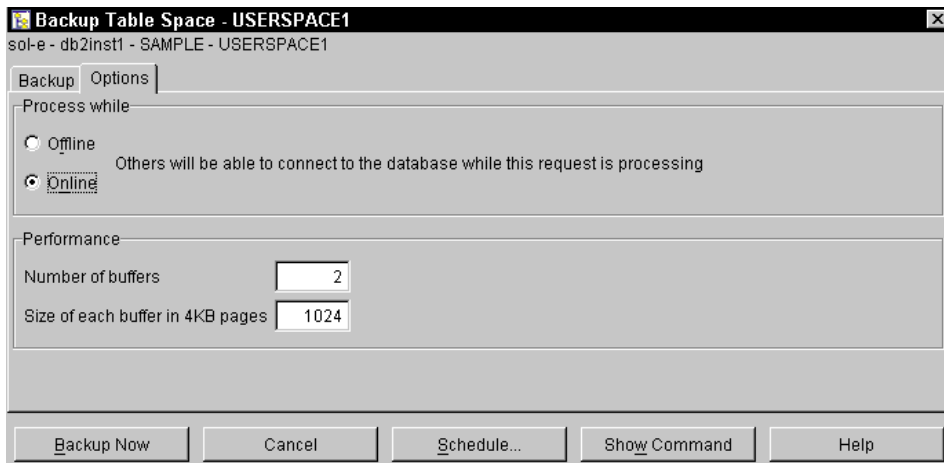


Figure 54. Backup tablespace using the online option

- Select **Backup Now** to backup the database. When the backup completes, a confirmation dialog appears as shown in Figure 55. Close the dialog box.

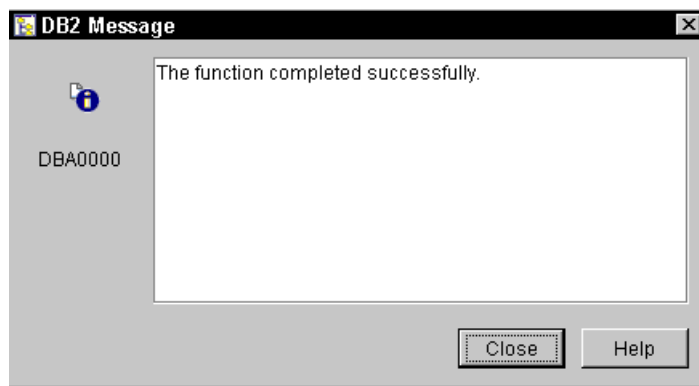


Figure 55. Backup successful dialog box

6.5.4 Load utility

In this section, we will look at the `use tsm` option of the DB2 load utility.

Normally, every insert or update into a DB2 database will be logged in the logfiles. This allows the database to be used in rollforward recovery mode.

After restoring a full database backup, every transaction after that backup can be rolled forward.

The DB2 load utility doesn't log its activity into the DB2 logfiles. The advantage is that the DB2 load utility is able to load the data very quickly into the DB2 database. The disadvantage is that, if not otherwise specified, the database will not be able for rollforward recovery.

To allow rollforward recovery when data will be loaded via the DB2 load utility the *copy yes* option of the load utility must be specified. This will generate a copy of the loaded data that will be saved to a specified location. In our example, we will save this copy to Tivoli Storage Manager server using the *use tsm* option.

```
$ db2 load from data.load of del insert into org copy yes use tsm

SQL3109N The utility is beginning to load data from file
"/home/db2inst1/tsm/load/data.load".

SQL3500W The utility is beginning the "LOAD" phase at time "03-27-2001
14:15:19.565784".

SQL3519W Begin Load Consistency Point. Input record count = "0".

SQL3520W Load Consistency Point was successful.

SQL3110N The utility has completed processing. "900" rows were read from the
input file.

SQL3519W Begin Load Consistency Point. Input record count = "900".

SQL3520W Load Consistency Point was successful.

SQL3515W The utility has finished the "LOAD" phase at time "03-27-2001
14:15:20.397451".

Number of rows read           = 900
Number of rows skipped        = 0
Number of rows loaded         = 900
Number of rows rejected       = 0
Number of rows deleted        = 0
Number of rows committed     = 900
```

This will send a copy of the data loaded to the DB2 database and also to the Tivoli Storage Manager server. (The data will go into the backup copy group of the associated management class.)

This data will be downloaded again when the DB2 rollforward command will need it.

Chapter 7. Day to day management: DB2 backups on UNIX

In this section we will discuss how to automate the DB2 backup and how to verify the success of these backups.

7.1 The db2adutl utility

The db2adutl is a very useful command that comes with DB2 to manage old backups and logfiles that reside on the Tivoli Storage Manager server. The tool is located in the \$HOME/sql/lib/adsm directory of the instance owner.

The db2adutl utility provides the ability to query, extract, delete, and verify backups, loadcopies, and log archives from Tivoli Storage Manager. All backup file names include a time stamp and therefore are unique for every backup to Tivoli Storage Manager. The backup copies in Tivoli Storage Manager are therefore always active and never deleted by normal Tivoli Storage Manager file expiration processing. The db2adutl utility provides a way of deleting backups and log files that are no longer required.

Figure 19 shows the syntax to invoke the db2adutl utility.

```
Usage: db2adutl
{ QUERY [ [ [ TABLESPACE | FULL | LOADCOPY ]
          [ SHOW INACTIVE ] ] |
          [ LOGS [ BETWEEN <Sn1> AND <Sn2> ] ] ] |

  EXTRACT [ [ [ TABLESPACE | FULL | LOADCOPY ]
            [ SHOW INACTIVE ]
            [ TAKEN AT <timestamp> ] ] |
           [ LOGS [ BETWEEN <Sn1> AND <Sn2> ] ] ] |

  DELETE [ [ [ TABLESPACE | FULL | LOADCOPY ]
           [ KEEP <n> |
             OLDER [THAN] [ <timestamp> |
                          <n> DAYS ] ] |
           [ TAKEN AT <timestamp> ] ] |
          [ LOGS [ BETWEEN <Sn1> AND <Sn2> ] ] ] }

  VERIFY [ [ TABLESPACE | FULL ]
         [ SHOW INACTIVE ]
         [ TAKEN AT <timestamp> ] ] |
         [ {DATABASE | DB} <database name> ]
         [ NODE <n> ]
         [ PASSWORD <password> ]
         [ NODENAME <nodename> ]
         [ OWNER <owner> ]
         [ WITHOUT PROMPTING ]
```

Figure 56. Syntax of db2adutl

QUERY queries either tablespace, full backup, loadcopy images, or log archives. If you type query only, it returns all four.

EXTRACT, if you do not qualify it, displays the name of each backup image and prompts you to extract the image. If you qualify the command you reduce the scope of the search. For this option, time stamps do not apply to log archives.

DELETE, if you do not qualify it, works similarly to extract, except that backup images are marked inactive and archive objects (log files) are deleted.

Note: The Tivoli Storage Manager server can be configured to retain deleted backup versions. In this case, backup objects remain on the server even if you delete them by using db2adutl. If you do not want to keep these inactive objects, you have to configure your Tivoli Storage Manager server to either of the following:

- Set the number of backup versions, if client data is deleted (VERDELETED), to 0.
- Or
- Set the length of time to retain the last remaining copy of a deleted file (REONLY) to 0.

For further Tivoli Storage Manager server considerations, see 4.5.2, “Tivoli Storage Manager management class considerations” on page 48.

The VERIFY qualifier, performs consistency checking on the tablespace or backup copy that is on the Tivoli Storage Manager server. This parameter causes the entire backup image to be transferred over the network. See 7.3.3, “Verification using db2adutl” on page 126 for further information.

The KEEP qualifier *n* keeps only the newest *n* images.

The OLDER qualifier (THAN is optional) deletes any images older than the specified time stamp (this must be the complete time stamp string) or the specified number of days. **Warning:** If you automated the deletion of backups using this qualifier, be aware backups may fail and your automation may delete your last valid backups.

The TAKEN AT <timestamp> qualifier is the time stamp with which you want to perform the operation. This qualifier is not allowed with LOGS, because LOGS have no time or date association.

The SHOW INACTIVE qualifier shows also the inactive versions of full database backup, tablespace or loadcopy images that have been deleted

with db2adutl and therefore marked deleted on the Tivoli Storage Manager server. This qualifier will only work if the Tivoli Storage Manager server is configured to allow inactive versions.

The DATABASE qualifier restricts the search to a particular database.

The WITHOUT PROMPTING qualifier disables the prompt. Be very careful when using this option because it may delete more than you have expected to delete.

7.2 Information about backups using Tivoli Storage Manager commands

With your Tivoli Storage Manager node name and password you have access to the Tivoli Storage Manager server. The same information that is available with db2adutl can also be directly acquired using Tivoli Storage Manager utilities. db2adutl presents the Tivoli Storage Manager information in a more readable view, but for problem determination it can be useful to view the data directly.

We will only give a few examples of what can be done using the SQL select commands. These queries can be made as complex as needed, see *Tivoli Storage Manager for Windows Administrator's Reference*, GC35-0411.

To login to the server open an administrative client session. On the command line enter the following command:

```
dsmadm
```

You will be prompted for the node name and the password. Use any administrator nodename as it should have sufficient privileges for the SELECT command. We used the node name associated with the Tivoli Storage Manager client API node name, as used in the dsm.sys file in 5.3, "Configuration and setup of Tivoli Storage Manager client API" on page 63.

```

$ id
uid=203(db2inst1) gid=102(db2iadml) groups=1(staff),101(db2asgrp)
$ dsmdmnc
Tivoli Storage Manager
Command Line Administrative Interface - Version 4, Release 1, Level 1.0
(C) Copyright IBM Corporation, 1990, 1999, All Rights Reserved.

Enter your user id: brazil_db2

Enter your password:

Session established with server BRAZIL: AIX-RS/6000
Server Version 4, Release 1, Level 2.0
Server date/time: 03/07/01 16:38:07 Last access: 03/07/01 16:25:13

tsm: BRAZIL> .

```

Type `help` to get a list of the possible commands you can use. Every backup or logfile is indexed in the Tivoli Storage Manager server database. To get a list of the backups you have already made, enter the following command:

```
select * from backups where node_name='<client API node name>'
```

Note: The `<client API node name>` must be in uppercase.

See Figure 57 for the results of the command.

```

tsm: BRAZIL>select * from backups where node_name='BRAZIL_DB2'
ANR2963W This SQL query may produce a very large result table, or may require a
significant amount of time to compute.

Do you wish to proceed? (Yes/No)y

      NODE_NAME: BRAZIL_DB2
      FILESPACE_NAME: /SAMPLE
      STATE: ACTIVE_VERSION
      TYPE: FILE
      HL_NAME: /NODE0000/
      LL_NAME: FULL_BACKUP.20010301184338.1
      OBJECT_ID: 19470
      BACKUP_DATE: 2001-03-01 18:43:39.000000
      DEACTIVATE_DATE:
      OWNER: db2inst1
      CLASS_NAME: DEFAULT

      NODE_NAME: BRAZIL_DB2
      FILESPACE_NAME: /SAMPLE
      STATE: ACTIVE_VERSION

```

Figure 57. Output of Tivoli Storage Manager Select command

To get more specific output, the query can be extended by combining more search queries together, as in the next example.

```
tsm: BRAZIL>select * from backups where node_name='BRAZIL_DB2' and FILESPACE_NAME='/SPLITDB'
Session established with server BRAZIL: AIX-RS/6000
Server Version 4, Release 1, Level 2.0
Server date/time: 03/07/01 17:50:18 Last access: 03/07/01 17:29:59

ANR2963W This SQL query may produce a very large result table, or may require a
significant amount of time to compute.

Do you wish to proceed? (Yes/No) y

      NODE_NAME: BRAZIL_DB2
    FILESPACE_NAME: /SPLITDB
          STATE: ACTIVE_VERSION
           TYPE: FILE
        HL_NAME: /NODE0000/
        LL_NAME: FULL_BACKUP.20010302101250.1
      OBJECT_ID: 19498
    BACKUP_DATE: 2001-03-02 10:12:50.000000
DEACTIVATE_DATE:
           OWNER: db2inst1
      CLASS_NAME: DEFAULT

tsm: BRAZIL>
```

There is also a command line interface to the Tivoli Storage Manager server combining the logon process and the Tivoli Storage Manager command in a single step. This can be useful if you are creating shell scripts that need information from the Tivoli Storage Manager server directly, such as:

```
dsmsadm -id=brazil-db2 -password=brazil_db2 "select * from backups"
```

When using the user exit, the logfiles will be backed up into the archive copy group. To get information about archived items, the query must be changed to select items from the *archives* table. See the following example.

```
tsm: BRAZIL>select * from archives where node_name='BRAZIL_DB2'  
ANR2963W This SQL query may produce a very large result table, or may require a  
significant amount of time to compute.
```

```
Do you wish to proceed? (Yes/No) y
```

```
      NODE_NAME: BRAZIL_DB2  
      FILESPACE_NAME: /SAMPLE  
      TYPE: FILE  
      HL_NAME: /NODE0000/  
      LL_NAME: S0000000.LOG  
      OBJECT_ID: 19487  
      ARCHIVE_DATE: 2001-03-01 19:52:47.000000  
      OWNER: db2inst1  
      DESCRIPTION: Log file for DB2 database SAMPLE  
      CLASS_NAME: DEFAULT
```

```
      NODE_NAME: BRAZIL_DB2  
      FILESPACE_NAME: /SAMPLE  
      TYPE: FILE  
      HL_NAME: /NODE0000/  
      LL_NAME: S0000000.LOG
```

7.3 Verification of DB2 backups

In this section we will discuss the different possible ways to validate the DB2 backups.

The most important and, of course, the most reliable way is to see whether a backup is recoverable — to actually restore it. At least once, a typical backup should be restored to test if the backup and the restore process itself is valid. There is no way to restore an unsuccessful, incomplete or even missing backup. See Chapter 10, “Recovering DB2 UDB databases” on page 213 for further information.

Once the backup process itself has been validated by doing a restore, the main purpose of this chapter is to outline which other possibilities there are to verify the success of the backup operation and to verify the backup object itself.

7.3.1 Verify backup using db2ckbkp

The db2ckbkp checks the integrity of the backup image and determines whether or not it can be restored. Some or all parts of the backup can be checked. The backup image must reside physically on the disk. See Figure 58 for an example.

```

$ db2ckbkp SAMPLE.0.db2inst1.NODE0000.CATN0000.20010308132920.001

[1] Buffers processed: ####

Image Verification Complete - successful.
$

```

Figure 58. Example db2ckbkp

7.3.2 Verification using DB2 list history

The DB2 history file (db2rhist.asc) contains logged events from different administrative events like backup, load, drop table, reorganize and so on.

The success of backups can be verified by listing the backup information using the DB2 `list history` command. See Figure 59 for an example (only one entry shown).

```

$ db2 list history backup all for sample

Op Obj Timestamp+Sequence Type Dev Earliest Log Current Log Backup ID
-----
B D 20010315070920001 F A S0000032.LOG S0000032.LOG
-----
Contains 2 tablespace(s):

00001 SYSCATSPACE
00002 USERSPACE1
-----
Comment: DB2 BACKUP SAMPLE OFFLINE
Start Time: 20010315070920
End Time: 20010315070953
-----
00005 Location: adsm/libadsm.a

```

Figure 59. DB2 list history

The relevant information is contained within the first line of each stanza. These lines show the:

- Operation (Op:B=backup)
- Object (Obj:D=Database)
- Type (Type:F=Offline,N=Online)
- Device (Dev:A=TSM)

For a more detailed description of the `list history` command see the *DB2 UDB Command Reference V7*, SC09-2951.

7.3.3 Verification using db2adutl

In this section, we will focus on some options of the db2adutl command and how they may be used to verify the DB2 backups.

The *query* option of the db2adutl utility can be used to get general information about which database backups, logfiles and load copies reside on the Tivoli Storage Manager server.

```
$ db2adutl query db sample

Query for database SAMPLE

Retrieving full database backup information.
  full database backup image: 1, Time: 20010314084325
  Oldest log: S0000019.LOG, Node: 0, Sessions used: 1
  full database backup image: 2, Time: 20010314084246
  Oldest log: S0000019.LOG, Node: 0, Sessions used: 2
Retrieving tablespace backup information.

  No tablespace backup images found for SAMPLE

Retrieving load copy information.

  No load copy images found for SAMPLE

Retrieving log archive information.
  Log file: S0000019.LOG, Node: 0, Taken at: 2001-03-14-08.43.13
  Log file: S0000020.LOG, Node: 0, Taken at: 2001-03-14-08.43.48
```

Figure 60. Example db2adutl query

An alternative test would be to only download one file from the Tivoli Storage Manager server and see if the whole file is readable. The db2adutl *extract* option must be used.

```

$ db2adutl extract full taken at 20010314084325 db sample

Query for database SAMPLE

Retrieving full database backup information. Please wait.

full database backup image:
./SAMPLE.0.db2inst1.NODE0000.CATN0000.20010314084325.001
, Node: 0

Do you wish to extract ALL of these images (Y/N)? y

Writing to file:
./SAMPLE.0.db2inst1.NODE0000.CATN0000.20010314084325.001

$

```

Figure 61. Example db2adutl extract

The best way to verify the backup is the *verify* option of the db2adutl utility. It is basically a way to run the db2ckbkp functionality through Tivoli Storage Manager to verify if the image is going to be restorable.

The whole image will be read from the Tivoli Storage Manager server into a local memory buffer (not the whole image at once but piece by piece). There is nothing written to local disk, but the actual data transfer over the local area network (LAN) should be considered.

Issuing db2adutl *verify* will cause the utility to read through the image in the same way that restore does. It will verify that all of the required objects (media header, DB configuration, DB history, list of tablespace, tablespace definitions, and so on) exist in the object. The tool performs numerous checks but it will not, and cannot, actually check the integrity of any of the userdata that exists in the backup image.

```

$ db2adutl verify full taken at 20010314084325 db sample

Query for database SAMPLE

Retrieving full database backup information. Please wait.

full database backup image:
./SAMPLE.0.db2inst1.NODE0000.CATN0000.20010314084325.001
, Node: 0

Do you wish to verify this image (Y/N)? y

Verifying file: ./SAMPLE.0.db2inst1.NODE0000.CATN0000.20010314084325.001

=====
BUFFER 0
=====
=====
BUFFER 1
=====
=====
BUFFER 2
=====
=====
BUFFER 3
=====

WARNING only partial image read, bytes read: 28672 of 4194304

Read 0 bytes, assuming we are at the end of the image

Image Verification Complete - successful.

```

Figure 62. Example db2adutl verify

Note: The Warning message shown in Figure 62 is misleading and can be ignored. The last line is the important one and it indicates that the verification was successful.

7.3.4 Sample script for backup and verification

The following is a sample script that can be used to backup and afterwards verify the backup image. This script must be adapted to fit the specific environment and is only an example of one way to do this. This script may also be used for an automated backup as described in 7.5.2, “Automate DB2 backup using DB2” on page 135.


```

#!/bin/ksh

LOG=/home/db2inst1/tsm/db2backup.log
DB=SAMPLE

#
# First backup the database
#

db2 backup db $DB use tsm >> $LOG 2>&1

#
# Verify if return code from backup is 0
#
RC=$?
if [ $RC != 0 ]
then
    echo "Backup was not successful" | mail db2inst1@brazil
    exit 1
fi

#
# Get the timestamp from backup out of logfile
#

TIMESTAMP=$(tail -3 $LOG | awk '{print $11}')

#
# Verify backup
#

db2adutl verify full taken at $TIMESTAMP db $DB without prompting >> $LOG 2>&1

#
# Check if verification is ok
#

RC=$?
if [ $RC != 0 ]
then
    echo "Verify was not successful" | mail db2inst1@brazil
    exit 1
fi

exit 0

```

Figure 63. Sample backup and verification script

7.3.5 Automatic notification for user exit failure

In this section, we give a short example about how to setup an automatic error notification, so that you will be informed whenever user exit encountered a problem.

The `db2uext2` command is responsible for sending or downloading the DB2 logfiles to Tivoli Storage Manager. If this command encounters an error a new entry is created in the `USEREXIT.ERR` command.

One approach is to check the `USEREXIT.ERR` file periodically to see whether it contains a new entry. However, a more direct way is to modify the `userexit` command itself to send the information not only to the `USEREXIT.ERR` file, but also to an automated operation product.

The `ErrorLog` function within the `db2uext2` C source-code is responsible for writing the required error information to the `USEREXIT.ERR` file. We can add some additional program code to this routine to send the data to another location or to run an operating system program. Try to add only simple additional code in order to keep the source code maintainable.

In the following example, we use the C `system` routine. The C `system` routine is designed to execute an operating system command on a command line. In our example, we use this routine to send a mail to the root user. (Only a short part of the `db2uext2.c` file is shown.) We do not cover any error handling in this example.

```
/* ----- */
/* Close the error log file */
/* ----- */
fclose( errorLogFp ) ;

/* Newly added line for automatic error notification */

system ( "echo 'userexit error' | mail -s userexit root" );
```

If the user exit writes an error message to the `USEREXIT.ERR` file, the root user will also get notification like the following:

```
Message 1:
From db2inst1 Fri Mar 30 16:48:29 2001
Date: Fri, 30 Mar 2001 16:48:29 -0600
From: db2inst1
To: root
Subject: userexit

userexit error
```

7.4 Deletion of backups and logfiles

In this section, we will focus on how to use the `db2adutl` command to delete backups and logfiles. We will give some example on how `db2adutl` may be used within shell scripts.

7.4.1 Deletion of backup objects

After some time, the database backups may not be needed anymore. The time when they will expire depends on the requirements users have on their data and the service level agreements drawn up between users and database administrators. Sometimes, special laws (for example, financial ones) require that backups must be accessible for many years. In other, very dynamic environments, backups older than one week may be of no value anymore.

Warning: Every customer installation has different data retention requirements but it is vital that the database and Tivoli Storage Manager administrators understand fully the implications of backup deletion and the policy management rules that are to be implemented. We cannot stress strongly enough that automated deletion should be planned and tested thoroughly.

Whatever the criteria, after some period, a backup will eventually become eligible for deletion. The backup and any corresponding logfiles will need to be deleted, usually to free storage if the databases are large.

To delete old backups that reside on the Tivoli Storage Manager server the `db2adutl` utility with the *delete* qualifier can be used. There is no other easy way to delete old backups. There are three options for the delete qualifier.

- TAKEN AT <timestamp>
- OLDER [THAN] [<timestamp> | n DAYS]
- KEEP n

We strongly recommend you use the KEEP **n** option. This will ensure that the last **n** successful backups will be kept. With the other options the last remaining backup could be accidentally removed.

7.4.2 Deletion of logfiles

Logfiles are used to rollforward a database to a certain point beginning from a database backup. Each database has an oldest logfile associated with it. All logfiles from the oldest forwards must be available to ensure that the rollforward procedure will work.

Only those logfiles should be deleted that will not be needed by any of the existing database backups for rollforward.

To delete old backed up logfiles the *delete* qualifier of the db2adutl utility must be used. The deletion of logfiles is itself a little bit complicated, because there is no timestamp associated with backed up logfiles. The range of the logfiles must be specified. See Figure 64 for an example (no tablespace or loadcopy backups are shown!).

```
Retrieving full database backup information.
full database backup image: 1, Time: 20010315135917
  Oldest log: S0000041.LOG, Node: 0, Sessions used: 1
full database backup image: 2, Time: 20010315134808
  Oldest log: S0000038.LOG, Node: 0, Sessions used: 1

Retrieving log archive information.
Log file: S0000035.LOG, Node: 0, Taken at: 2001-03-15-13.37.54
Log file: S0000036.LOG, Node: 0, Taken at: 2001-03-15-13.44.27
Log file: S0000037.LOG, Node: 0, Taken at: 2001-03-15-13.47.10
Log file: S0000038.LOG, Node: 0, Taken at: 2001-03-15-13.47.53
Log file: S0000039.LOG, Node: 0, Taken at: 2001-03-15-13.48.38
Log file: S0000040.LOG, Node: 0, Taken at: 2001-03-15-13.55.29
Log file: S0000041.LOG, Node: 0, Taken at: 2001-03-15-14.00.03
Log file: S0000042.LOG, Node: 0, Taken at: 2001-03-15-14.00.48

$ db2adutl delete logs between S0000035.LOG and S0000037.LOG db sample

Query for database SAMPLE

Retrieving log archive information.
Log file: S0000035.LOG, Node: 0, Taken at: 2001-03-15-13.37.54

Do you want to delete this log image (Y/N)? y

Are you sure (Y/N)? y
Log file: S0000036.LOG, Node: 0, Taken at: 2001-03-15-13.44.27

Do you want to delete this log image (Y/N)? y

Are you sure (Y/N)? y
Log file: S0000037.LOG, Node: 0, Taken at: 2001-03-15-13.47.10

Do you want to delete this log image (Y/N)? y

Are you sure (Y/N)? y
```

Figure 64. Example db2adutl delete logfiles

The between range is a little bit misleading. As shown in the example, the logfiles 35 and 37 are also being deleted.

7.4.3 Automated deletion of backups and logfiles

It will be useful to automate not only the backup operation but also the deletion of backups and logfiles in some way. There is no way other than to use the `db2adutl` command to delete selective backups or logfiles that reside on the Tivoli Storage Manager server. So the automation must be a script that will be executed by some automated scheduling mechanism.

Within an automated running script it is almost impossible to use interactive commands. To overcome this problem the `without prompting` option of the `db2adutl` utility can be used, but be aware of the possible impact and test your script very carefully so that it will not delete more than it is expected to. See Figure 65 for an example script.

```
#!/bin/ksh
set -x

DB=SAMPLE
KEEP_FULL=5

#
# Delete full DB backups and keep $KEEP_FULL
#

db2adutl delete full keep $KEEP_FULL db $DB without prompting

#
# This is the tricky part to determine the oldest logfilename
# We assume that the last image in the query output has the oldest log number
# We had to subtract one from the oldest logfile in order to keep the
# last backup valid.
#

let OLDEST_LOG=$(db2adutl query full db $DB | tail -2 | awk '{print $3}' | cut
-c2-8 )
let OLDEST_LOG=$OLDEST_LOG-1
OLD_LOG=$(printf "%-7.7d" $OLDEST_LOG)
OLD_LOG="$S$OLD_LOG.LOG"

# Now delete obsolete logs
#

db2adutl delete logs between S0000000.LOG and $OLD_LOG db $DB without prompting
```

Figure 65. Example script of deletion of obsolete backups and logfiles

This script does not contain any error handling, notification or recovery in case of problems and must be adapted and well tested in every specific environment.

Generally such a script needs to be reviewed whenever the output of the `db2adutl` utility changes, for example, after applying a fixpak. Also keep in mind that after the restore of a backup, the logfile numbering may change.

7.5 Automation of DB2 backup

This section will show which possibilities there are to automate the backup. We will describe ways to automate the backup using `cron`, DB2 and Tivoli Storage Manager utilities.

The examples shown here include some simple shell scripts written to automate parts of the backup process. The scripts do not include any error checking or automation recovery as those processes depend heavily on the configuration of your databases and how you want to structure your backups.

7.5.1 Automate DB2 backup using cron

This section will cover the use of the utility `cron` to automate the backup of a DB2 database.

The `cron` utility runs shell commands at specified dates and times. It consists of two parts: the `cron` daemon and the `crontab` file.

The `cron` daemon checks every minute for an entry in the user-related `crontab` file (in the `/var/spool/cron/crontabs` directory) that can be executed. Every user can have its own `crontab` file, but the `root` user has to create this file first. If the time specified in the entry matches the current time, the entry will be executed. The example below shows an entry in a `root` user `crontab` file where a shell script called `/etc/stop_db` is performed every day at 01:30 (a.m.). Use the command, `crontab -l`, to display the contents of the `crontab` file.

```
# crontab -l
30 1 * * * /etc/stop_db
#
```

Each `crontab` file entry consists of a line with six fields, separated by spaces and tabs. The fields contain, respectively:

1. The minute (0 through 59)
2. The hour (0 through 23)
3. The day of the month (1 through 31)
4. The month of the year (1 through 12)

5. The day of the week (0 through 6 for Sunday through Saturday)
6. The shell command

Cron starts a limited bsh (Bourne shell) for each command user. This is important when starting a single command from crontab, make sure that the command is able to run in bsh. If the command started from the crontab is a shell script every shell can be used inside the shell script, like the following example.

```
#!/bin/ksh
# Backup the database
db2 backup db sample use tsm
```

Figure 66. Sample backup shell script

To imbed the shell script into cron use the `crontab -e` command. This will open the default editor or the editor specified by the `EDITOR` environment variable.

```
# export EDITOR=/usr/bin/vi
# crontab -e
```

Now edit a line similar to the following. Remember to use full pathnames for the executables and logfiles. If stdout and stderr is not redirected the owner, named the same as the crontab file, will receive mail every time the command produces output to stdout or stderr.

```
30 1 * * * /home/db2inst1/tsm/db2backup.ksh >/tmp/stdout.out 2>/tmp/err.out
```

7.5.2 Automate DB2 backup using DB2

This section covers the DB2 Journal, the DB2 Scheduler, and the DB2 Script Center and how it can be used to automate DB2 backups.

To access the DB2 Journal, the DB2 Scheduler, or the DB2 Script Center, the DB2 Control Center can be used. The DB2 Control Center is a graphical interface to perform administrative tasks, such as configuring the system, managing directories, backing up and recovering the system. The DB2 Control Center does not need to be installed on the same machine where the database resides.

To launch the DB2 Control Center enter `db2cc` on UNIX systems. On Windows systems, the DB2 Control Center can be found under **Start->Programs->IBM DB2**.

Open the DB2 Control Center and click down the tree until reaching the database folder.

Select the database to be backed up and right-click on the database name and select **Backup and Database** as shown in Figure 67.

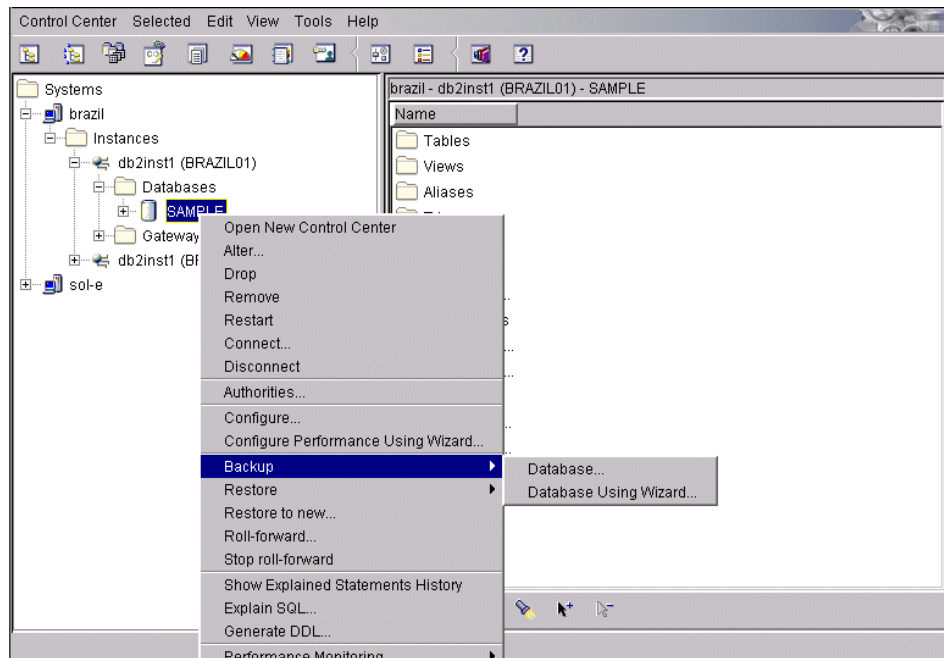


Figure 67. Select backup database in DB2 Control Center

Choose whether you want to do an online or offline, full or tablespace backup. Do not forget to set the media type to Tivoli Storage Manager.

After you make your selection press the **Schedule** button as shown in Figure 68.

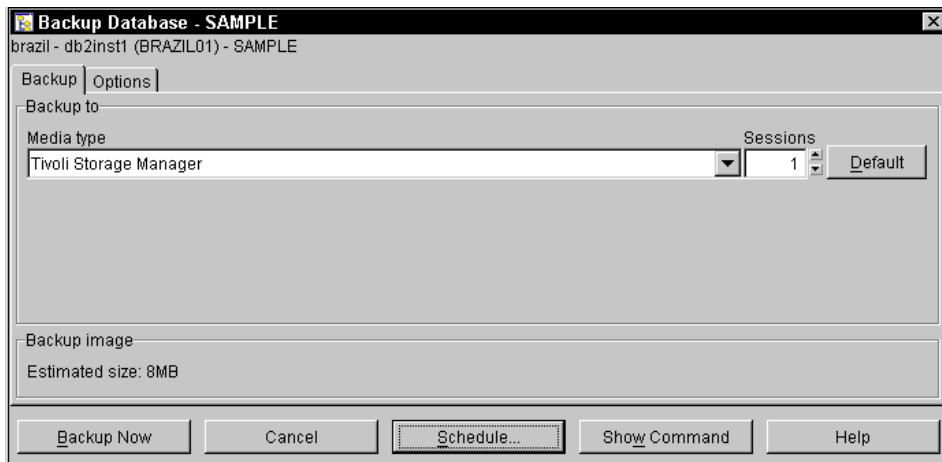


Figure 68. DB2 backup window

This will open the DB2 schedule window as shown in Figure 69.

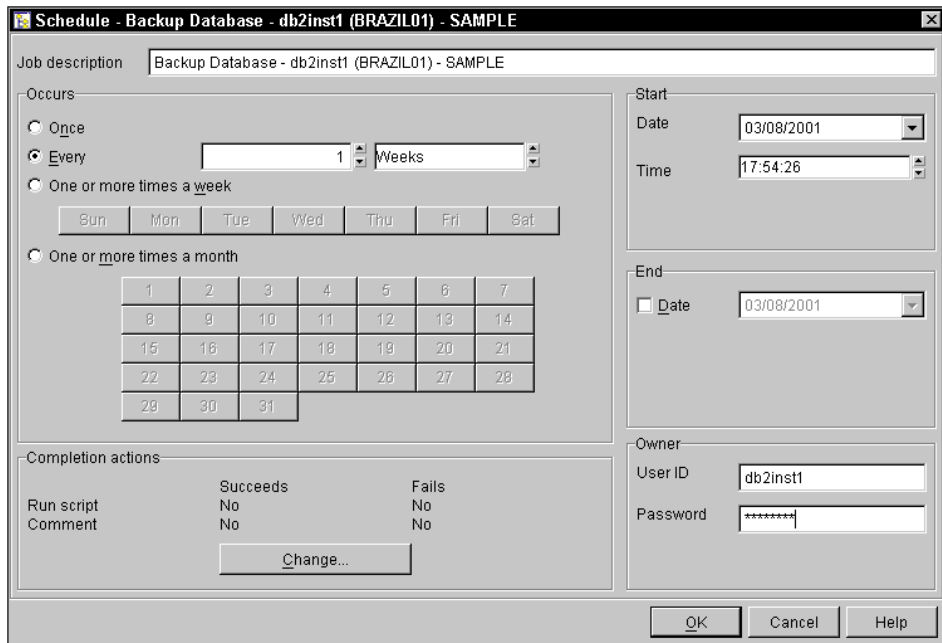


Figure 69. DB2 schedule window

This window is very useful. Click on the desired actions when the backup should take place and for how long, and the backup should be started automatically. The userid and password are required and the userid must have enough privileges to do a backup.

It is not possible to select the type of backup (offline or online, full or tablespace) from this window. This can only be done by starting again from the backup window as shown in Figure 68.

To monitor the success of scheduled backups the DB2 Journal can be used. The DB2 Journal is located under the Tools Menu on the DB2 Control Center. Once started a window will open like the one in Figure 70.

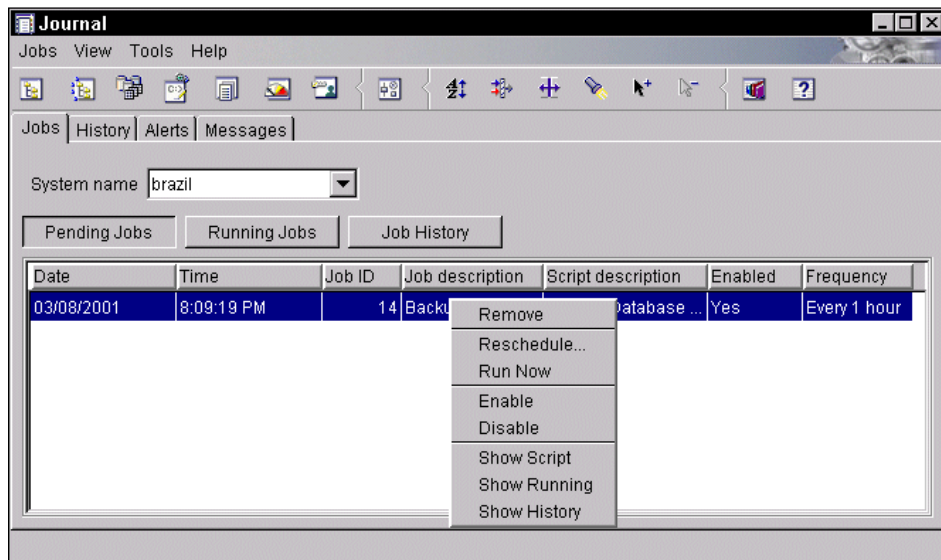


Figure 70. DB2 journal

The system name must be selected first. By pressing the corresponding button, information about the jobs that are already finished (Job History), running (Running Jobs) or are expected to run (Pending Jobs) will appear. In order to read all the information, resize the column by moving the border of the column description.

To manage a job, the job must be highlighted. The possible options will be selectable either from the Jobs menu or by right-clicking on the highlighted job.

Sometimes, you need to customize the DB2 commands or run a series of commands as a script or batch files. You can use the Script Center to create a script. This is located under the Tools menu of the Control Center. Figure 71 shows the Script Center, and the scripts available under a system named BRAZIL.

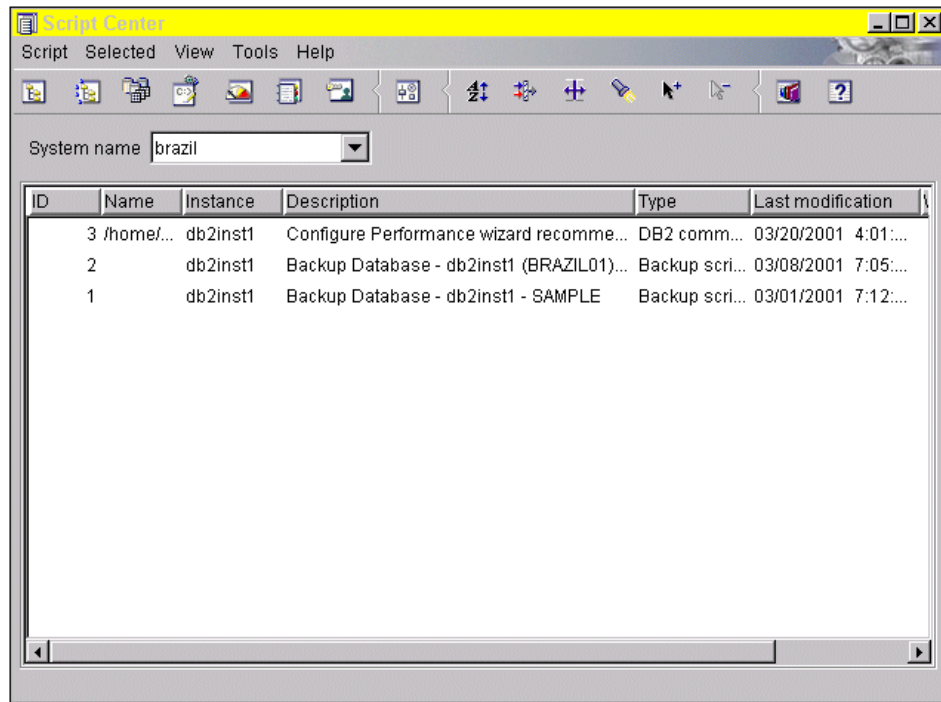


Figure 71. Script center

You can create a new script from the **Scripts->New** menu. Figure 72 shows a backup script using the parallelism option.

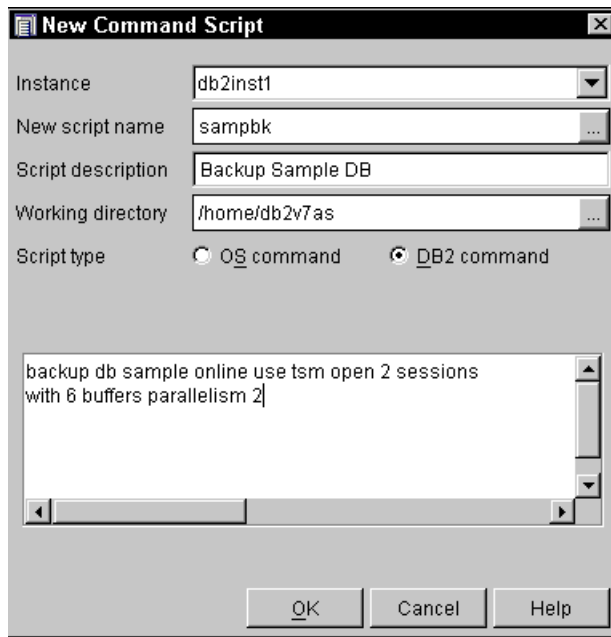


Figure 72. New command script

To schedule the script, highlight the entry in the Script Center, and right-click to display the drop down menu as shown in Figure 73. You can then select **Schedule** to schedule the script.

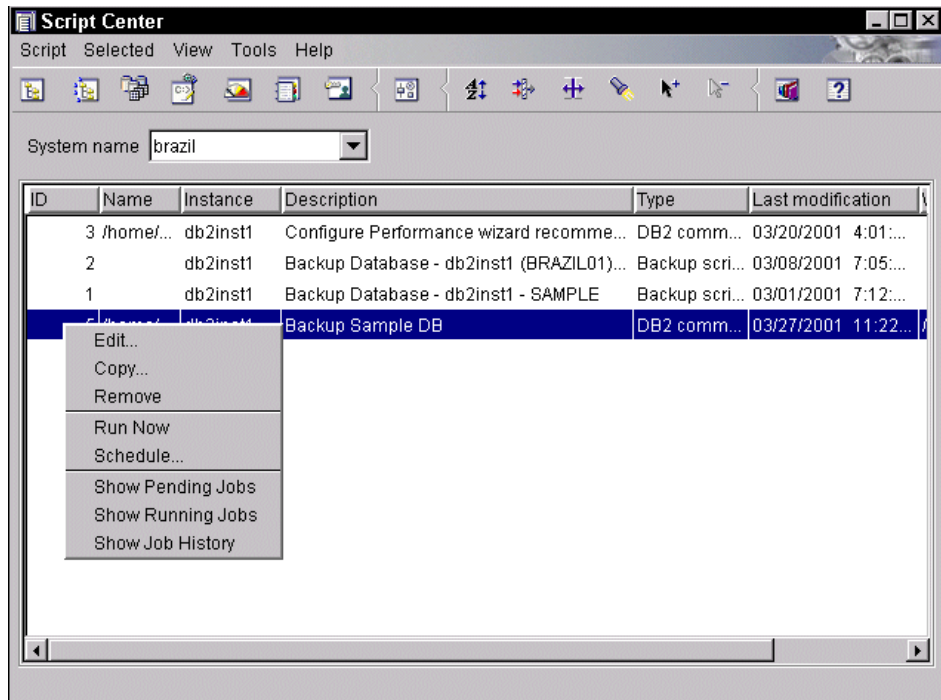


Figure 73. Scheduling a script

7.5.3 Automate DB2 backup using Tivoli Storage Manager

We will look at how the Tivoli Storage Manager central scheduling facility can be used to automate the backup of a database. Some database backup products, such as SQL-BackTrack, provide their own automation facilities. Details about how to use their automation facilities are in the respective product documentation.

In this section, we give an example of a Tivoli Storage Manager server prompted schedule. We will not go into every detail, as this will exceed the scope of this book. For further information on the Tivoli Storage Manager scheduler, see “Chapter 13, Scheduling operations for client nodes” in *Tivoli Storage Manager for AIX Administrator’s Guide*, GC35-0403.

The simplest way of scheduling an automated backup of a database is to write a short script to perform the backup and then use the Tivoli Storage Manager central scheduler to issue a switch user (su) command to execute the script.

To make use of the Tivoli Storage Manager Scheduler, root access and a Tivoli Storage Manager user with administrative privileges is needed. The necessary steps are:

- Register and setup a Tivoli Storage Manager baclient for this node.
- Define a shell script to be executed.
- Define the Tivoli Storage Manager schedule to execute shell script.
- Start the Tivoli Storage Manager schedule daemon on the client.

7.5.3.1 Setup Tivoli Storage Manager baclient

The Tivoli Storage Manager baclient must be setup in order to use the Tivoli Storage Manager client scheduler. If there is already a Tivoli Storage Manager baclient defined, this one can be used. We assume that the Tivoli Storage Manager baclient software is already installed.

As root user, create a stanza in the `/usr/tivoli/tsm/client/ba/bin/dsm.sys` file for the baclient. This stanza is similar to that one already defined in the `/usr/tivoli/tsm/client/api/bin/dsm.sys` file for the Tivoli Storage Manager API client.

Note

Every Tivoli Storage Manager client that runs a scheduler in prompted mode must connect over a dedicated port to the server. If there is more than one Tivoli Storage Manager client scheduler running on the same machine, the TCP port must be different.

```
SErvername  Brazil_db2
COMMmethod  TCPip
TCPport     1500
TCPserveraddress  9.1.150.57
TCPCLIENTport  1501      * Each locally client must have a
              * different TCP Port if running a
              * prompted scheduler at the same time

NODename    Brazil_db2
PasswordAccess  Generate
SCHEDMode   Prompted
SCHEDLOGName  /home/db2inst1/tsm/dsmsched.log
ERRORLOGName /home/db2inst1/tsm/dsmerror.log
```

Figure 74. Example `dsm.sys` file for Tivoli Storage Manager client scheduler

7.5.3.2 Define a script to be executed

The instance owner can create a script to automate the backup. We do not cover any error handling or recovery in this script. The sample script below needs to be adapted to fit into your environment.

```
#!/bin/ksh
db2 backup db sample use tsm
```

7.5.3.3 Configure the TSM schedule to perform the backup script

This step requires Tivoli Storage Manager administrator privileges.

Logon to the sever via an *administrative client session* with a user that has Tivoli Storage Manager server administrator privileges:

```
dsmamdc
```

The following information is necessary to define the schedule:

- Name of the shell script
- Name of the domain to which the node belongs
- Time when schedule should run

```
tsm: BRAZIL>def sched api_domain db2sched act=command objects='su - db2inst1 -c
/home/db2inst1/db2backup.ksh' starttime=09:35
ANR2500I Schedule DB2SCHED defined in policy domain API_DOMAIN.
tsm: BRAZIL>q sched
```

Domain	* Schedule Name	Action	Start Date/Time	Duration	Period	Day
API_DOMAIN	DB2SCHED	CMD	03/08/01 09:35:00	5 M	1 D	Any

```
tsm: BRAZIL>
```

Now the Tivoli Storage Manager client node needs to be associated to the previously defined schedule. Use the define association command as shown here:

```
tsm: BRAZIL>def assoc api_domain db2sched brazil_db2
ANR2510I Node BRAZIL_DB2 associated with schedule DB2SCHED in policy domain
API_DOMAIN.
tsm: BRAZIL>
```

7.5.3.4 Start the client scheduler

The client scheduler is part of the baclient and only the root user is allowed to start it. It is started with the `dsmc sched` command. Because we may use a specific stanza from our `dsm.sys` file, we need to specify which one to use with the `-se` option:

```
dsmc sched -se=brazil_db2
```

This command runs in the foreground. The output will look like the following.

```
# dsmc sched -se=brazil_db2
Tivoli Storage Manager
Command Line Backup Client Interface - Version 4, Release 1, Level 1.0
(C) Copyright IBM Corporation, 1990, 2000, All Rights Reserved.

Querying server for next scheduled event.
Node Name: BRAZIL_DB2
Session established with server BRAZIL: AIX-RS/6000
  Server Version 4, Release 1, Level 2.0
  Server date/time: 03/08/01 09:12:13 Last access: 03/08/01 09:11:38

Next operation scheduled:
-----
Schedule Name:      DB2SCHED
Action:             Command
Objects:            su - db2inst1 -c /home/db2inst1/tsm/db2backup.ksh
Options:
Server Window Start: 09:25:00 on 03/09/01
-----
Waiting to be contacted by the server.
```

The same information will also be logged into the `dsmsched.error` logfile that we specified in the `dsm.sys` file in Figure 74 on page 142.

The Tivoli Storage Manager client scheduler now waits until a schedule to which this Tivoli Storage Manager client is associated reaches its start time. At this specified time, the Tivoli Storage Manager server will prompt the client to execute the specified action. In this case, it will be the command as described by the `tsm` option *objects*.

To start the Tivoli Storage Manager client scheduler automatically every time the system is rebooted, a line must be inserted in `/etc/inittab`. Use the `inittab` action *once* as `dsmc sched` runs in foreground, otherwise, this line will block the start of the next lines in `/etc/inittab`.

```
imgss:2:once:/usr/IMNSearch/bin/img_start >/dev/console 2>&1
db:2:once:/etc/rc.db2 > /dev/console 2>&1 # Autostart DB2 Services
i4ls:2:wait:/etc/i4ls.rc > /dev/null 2>&1 # Start i4ls
orapw:2:wait:/etc/loadext -l /etc/pw-syscall
dsmc:once:/usr/bin/dsmc sched -se=brazil_db2
```

For AIX, the Tivoli Storage Manager client scheduler can be defined as an owned resource for the system resource controller (SRC). This can be done using the `mkssys` command. After the resource has first been defined, it can be started, controlled and stopped with the `startsrc`, `lssrc` and `stopsrc`

commands. Other UNIX systems may have their own resource management where the Tivoli Storage Manager client scheduler can be included.

```
# mkssys -s dsmc.brazil_db2 -p /usr/tivoli/tsm/client/ba/bin/dsmc -u 0 -a"sched \  
-se=brazil_db2 -i/dev/null -o/dev/null -e/dev/null -O -S -f 9 -n 15 -G tsm  
0513-071 The dsmc.brazil_db2 Subsystem has been added.  
# startsrc -s dsmc.brazil_db2  
0513-059 The dsmc.brazil_db2 Subsystem has been started. Subsystem PID is 24698  
# lssrc -g tsm  
Subsystem      Group      PID      Status  
dsmc.brazil_db2 tsm        24698    active  
# stopsrc -s dsmc.brazil_db2  
0513-044 The dsmc.brazil_db2 Subsystem was requested to stop  
# lssrc -s dsmc.brazil_db2  
Subsystem      Group      PID      Status  
dsmc.brazil_db2 tsm        24698    inoperative  
#
```

Information about scheduled events will show up in the dsmsched.log as defined in dsm.sys file in Figure 74.

7.6 Maintaining the history file

After a while, the history file (db2rhist.asc) would get large, and it would contain more entries than are really required. You need to delete entries to keep the size of the file small. Note that deleting the entries only deletes the information, and **not** the backup. You still need to use db2adutl to delete backup and logs from Tivoli Storage Manager.

For example, to delete entries with timestamps on or before March 24, 2001 00:00 (20010324000000) on database BART, you can issue the following:

```
$ db2 prune history 20010324  
DB20000I The PRUNE command completed successfully.
```

Note that you can specify the initial prefix for the timestamp. The minimum specification is `yyyy`.

Part 3. Backing up DB2 on the Windows 2000 platform

Chapter 8. Backing up DB2 on Windows 2000

To backup DB2 databases using Tivoli Storage Manager, you must register a node on the Tivoli Storage Manager server, install the Tivoli Storage Manager API, define environment variables, configure the client options file (dsm.opt), and generate the encrypted password using `dsmapipw.exe`. This chapter discusses these required steps.

8.1 Registering a node on the TSM server for DB2 backups

The first step in configuring DB2 to backup to a Tivoli Storage Manager server is to register a node with the Tivoli Storage Manager server. Before registering the node, you should create a new storage policy for the nodes that will be performing DB2 backups. The storage policy consists of a policy domain, policy set, management classes, backup copy groups, and archive copy groups. This is discussed in Chapter 4, "Tivoli Storage Manager server considerations" on page 33.

On our Tivoli Storage Manager server we had already defined a new policy domain called 'API_DOMAIN', a policy set 'API_POLICY', a management class 'API_MGMTCLASS'. The management class contains both a backup copy group and an archive copy group.

The backup copygroup has retention settings of VERExists=1, VERDelete=0, RETExtra=0, RETOnly=0, and a destination to a disk storage pool. The rest of the values were left as defaults when defining the backup copy group. The backup copy group will be used for all the data objects that are sent by DB2 to the Tivoli Storage Manager server with the exception of the log files.

The archive copy group has retention settings of RETVersion=NOLIMIT and a destination to a disk storage pool. The archive copy group is only used for the log files that are sent by the user exit.

With this storage policy defined, the nodename of JAMAICA_DB2 was registered with the Tivoli Storage Manager server. We made sure to specify that the node could delete backup objects with the BACKDELETE=YES option. If this option was *not* set, the `db2adutl delete` command would fail. Also we set the maximum number of mount points to 2, corresponding to the number of drives.

```
tsm: BRAZIL>register node jamaica_db2 jamaica_db2 domain=api_domain
backdelete=yes maxnummp=2

ANR2060I Node JAMAICA_DB2 registered in policy domain API_DOMAIN.
ANR2099I Administrative userid JAMAICA_DB2 defined for OWNER access to node
JAMAICA_DB2.
```

8.2 Downloading latest Tivoli Storage Manager baclient and API

The current version of the Tivoli Storage Manager client code which contains the Tivoli Storage Manager API, can be downloaded from the following ftp site.

```
ftp://ftp.software.ibm.com/storage/tivoli-storage-management/maintenance
/client
```

From this site you can select which Tivoli Storage Manager client version and release. The latest is Version 4 Release 1. In Figure 75 this is represented by the folder v4r1.

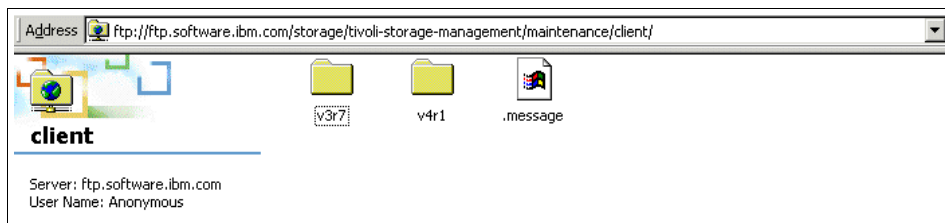


Figure 75. Tivoli Storage Manager client version and release

After selecting the version and release, you are presented with a list of operating systems to choose from. Select the **Windows** directory, then select the **i386™** directory, and now select which level of code to download. Figure 76 shows the levels of the Version 4 Release 1 Tivoli Storage Manager clients that were available in March of 2001. The folder **LATEST** always links to the latest level of client code available. Currently the latest level is v412, so the **v412** folder or the **LATEST** folder will take you to the same place.

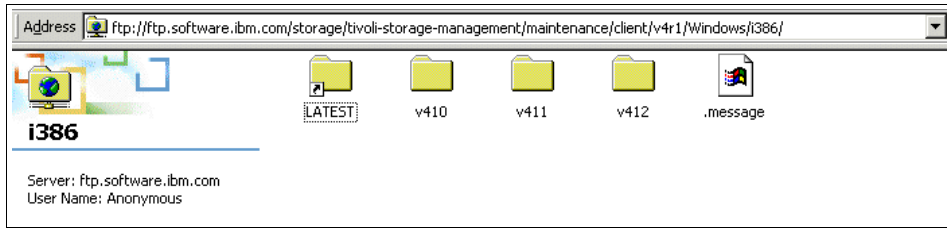


Figure 76. Levels of Tivoli Storage Manager client code

Figure 77 shows the files available for the latest level of the Tivoli Storage Manager client. At the time of writing, the file ending readme.ftp (IP22151_12_readme.ftp) contains instructions on how to download and install the client code. The file ending read1stc.txt (IP22151_12_read1stc.txt) is the readme for this level of code. The readme contains useful information that should be read before download and installation. Among other things the readme contains: system requirements, warnings, APARS fixed in the PTF, limitations, and so on.

We downloaded the file IP22151_12.exe and saved it to the local hard drive of the machine that the DB2 server is running on. The full level information of this package is Version 4 Release 1 Level 2.12.

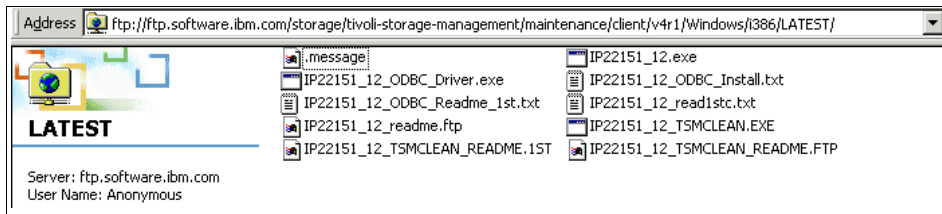


Figure 77. Files available for latest level

8.3 Installation

After downloading the Tivoli Storage Manager client file, double-click the package to start the installation process. You will be presented with a window requesting a temporary location to extract the contents of the package (Figure 78).

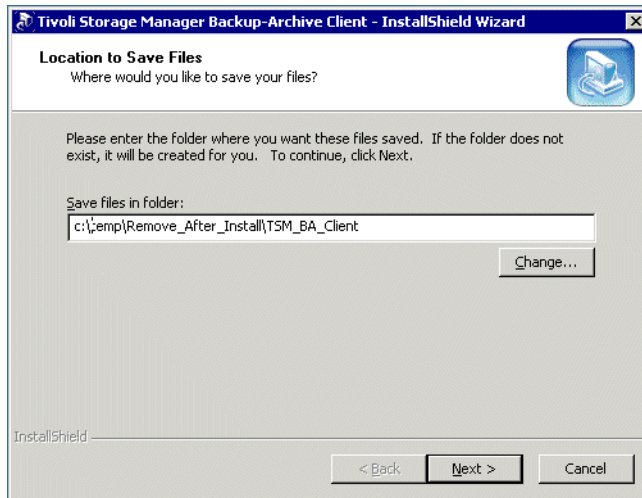


Figure 78. Location to Save Files window

The next window will prompt you for which language you want to use (Figure 79).

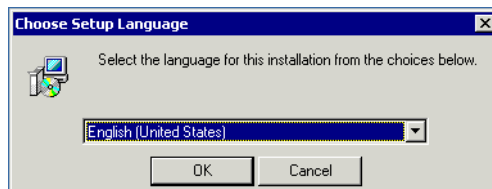


Figure 79. Language selection

After selecting the desired language, you will be presented with a welcome window to InstallShield, informing you that this will install the Tivoli Storage Manager Client (Figure 80).

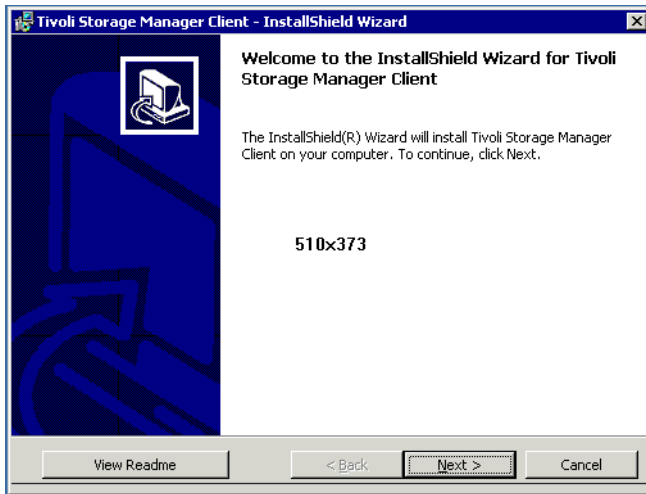


Figure 80. InstallShield Wizard

Select the desired installation directory. We selected the default location of c:\Program Files\Tivoli\TSM\ (Figure 81).

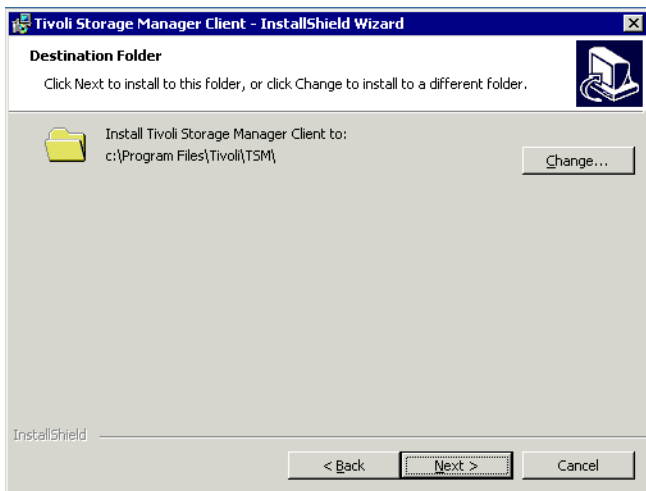


Figure 81. Installation directory selection

Select either Complete or Custom (Figure 82). Complete includes the Tivoli Storage Manager Backup-Archive client, Tivoli Storage Manager API Runtime files, and documentation. Complete does not include the Tivoli Storage Manager API Software Developer Kit nor the Tivoli Storage Manager Server

Administrative Client. If you select Custom, you can select only those elements you require. We needed the Tivoli Storage Manager API Run-time files, so we chose the Custom option.

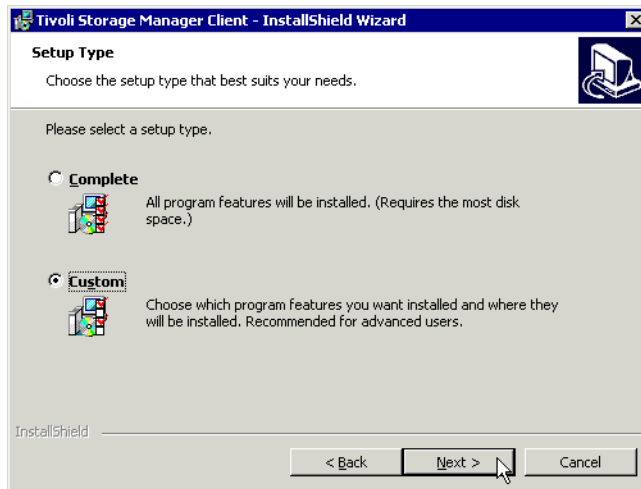


Figure 82. Complete or custom

We also selected the administrative client for troubleshooting purposes and the API SDK files which are required when using the DB2 user exit program (Figure 83).

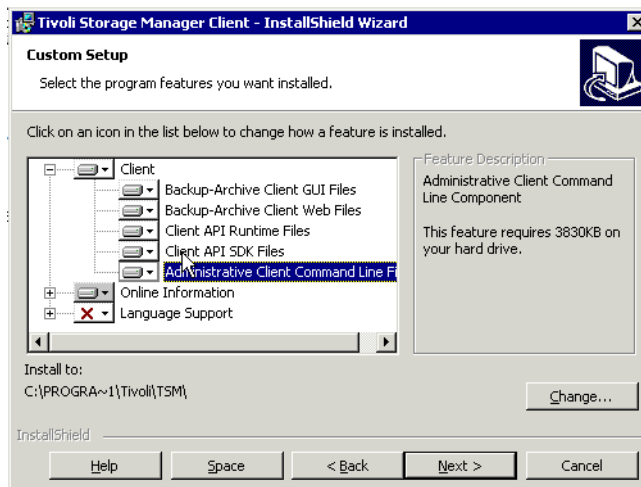


Figure 83. Selecting the API SDK and administrative client files

After selecting a complete or custom installation you are presented with a window like Figure 84, where you can confirm your choice.

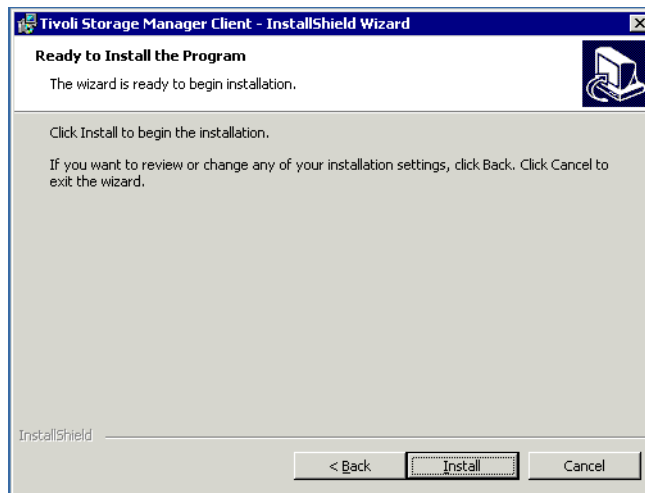


Figure 84. Installation confirmation window

After the files have been installed you are presented with a window informing you of successful completion of the installation (Figure 85).

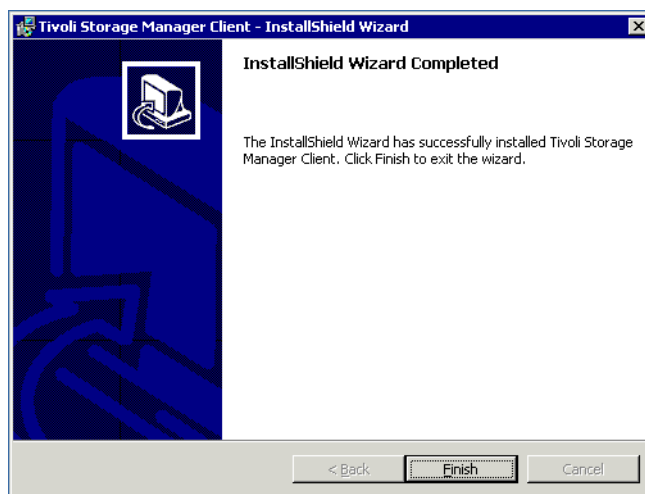


Figure 85. Successful installation confirmation

8.4 Configuring the API

After installing the Tivoli Storage Manager API you must then define the DSMI_CONFIG, DSMI_DIR, and DSMI_LOG environment variables which will be used later by the api client during backup. On Windows 2000, right-click the **My Computer** icon and select **Properties** (Figure 86).

You can also open up **Control Panel** and select **System**.

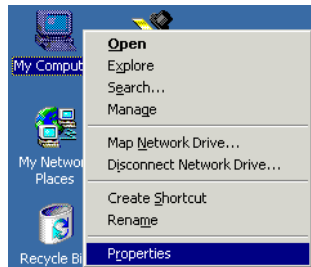


Figure 86. Right-click My Computer icon

From the **System Properties** window, select the **Advanced** tab (Figure 87).

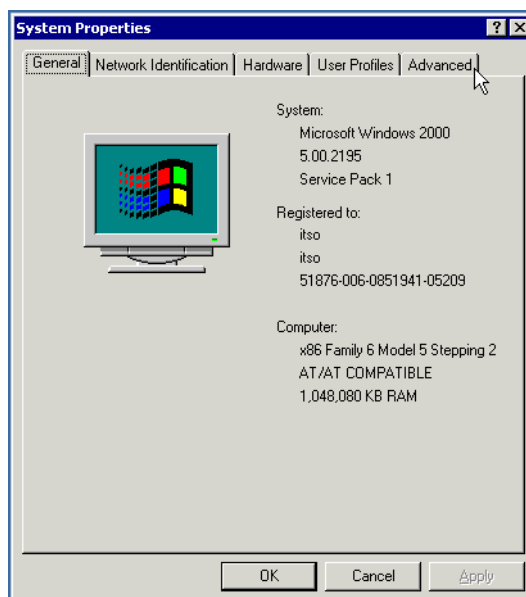


Figure 87. System Properties: General tab

From the **System Properties->Advanced** tab, select **Environment Variables** (Figure 88).

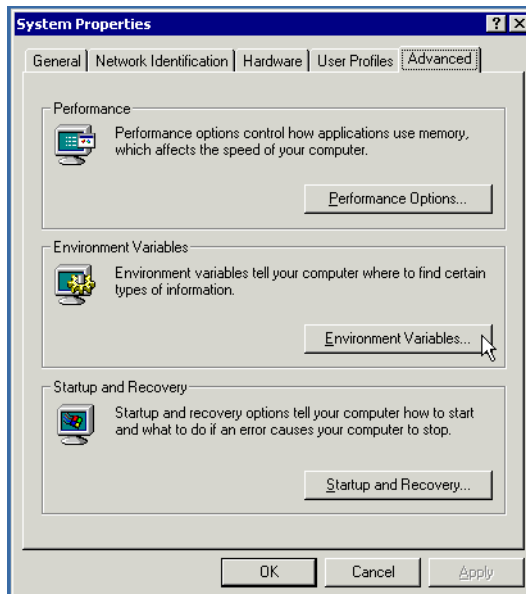


Figure 88. System Properties: Advanced tab

Select **New** from the System variables window to enter the first environment variable (Figure 89). Do **NOT** enter these environment variables as User variables. DB2 UDB runs as a Windows service, so it reads environment variables from the system and not from a user. If you set the environment variables as User variables they will *not* have any effect on the DB2 runtime engine.

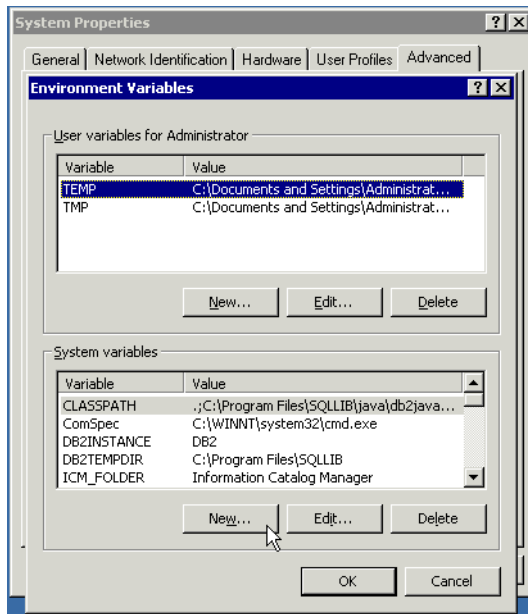


Figure 89. Environment variables

The order in which you define these variables is not important. We chose to define DSMI_CONFIG first (Figure 90). This variable points to the location of the api client options file. We gave it the default value of 'c:\program files\tivoli\tsm\api\dsm.opt'. If you have installed into a different directory you will need to set the path accordingly.

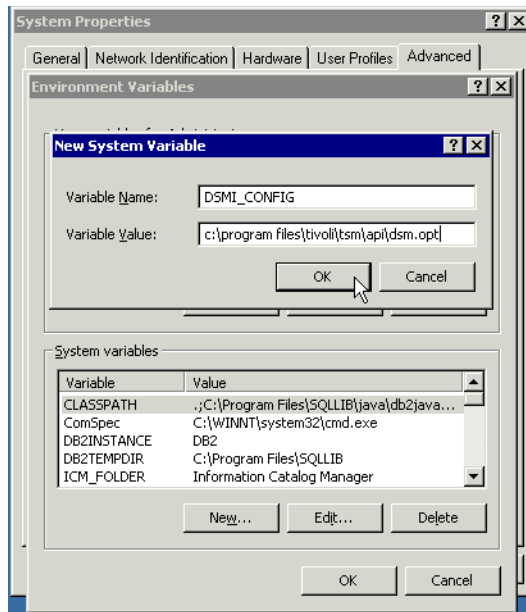


Figure 90. Setting DSMI_CONFIG system variable

The second system environment variable that we defined was DSMI_DIR (Figure 91). This value needs to be set to the installation directory of the Tivoli Storage Manager API. Based on our installation, we set this value to 'c:\program files\tivoli\tsm\api'.

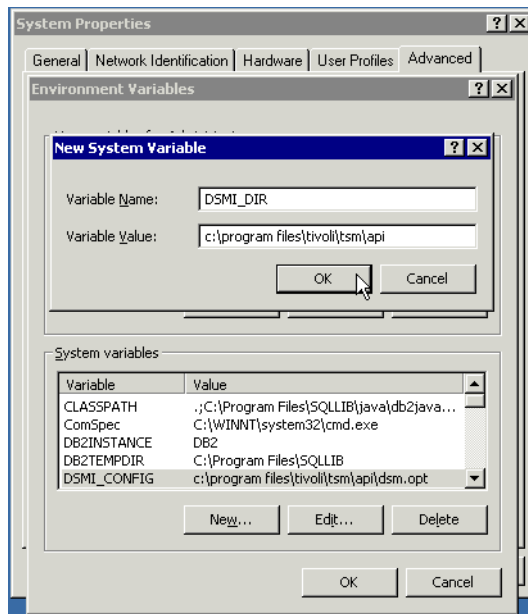


Figure 91. Setting the DSMI_DIR system variable

The third and last variable that needs to be set is DSMI_LOG (Figure 92). This value specifies the directory where the dserror.log is to be located. As the name suggests this log will be the repository for any error messages generated by the API client. You cannot change the name of the log file that is created. If you do specify a logname at the end of the path, Tivoli Storage Manager will consider your logname as part of the directory path, and if the directory path does not exist, Tivoli Storage Manager will not generate the dserror.log.

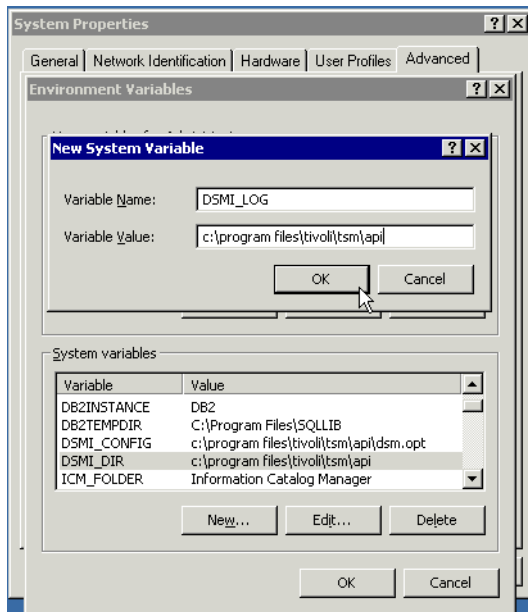


Figure 92. Setting the DSMI_LOG system variable

Once the environment variables have been set, we verified that they were set correctly by viewing them in the System variables window (Figure 93).

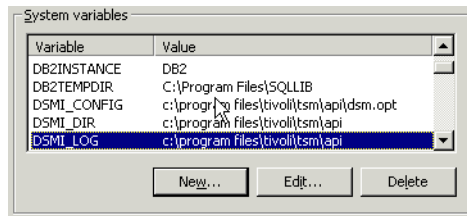


Figure 93. Verifying the system variables

8.5 Configuring the client options file

We created the client options file by right-clicking the folder `c:\program files\tivoli\tsm\client\api` and selecting **New->Text Document** (Figure 94). We named this file `dsm.opt`. This corresponds to the value in environment variable `DSMI_CONFIG`.

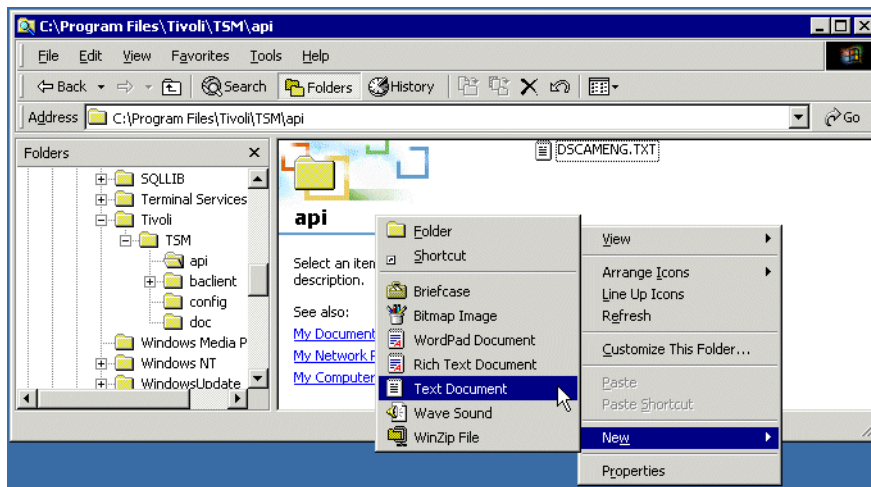


Figure 94. Creating the client options file

We then edited the `dsm.opt` client options file using Notepad (Figure 95). We specified values for `COMMETHOD`, `TCPSERVERADDRESS`, `TCPPORT`, `NODENAME`, and `PASSWORDACCESS`. Our Tivoli Storage Manager server is on an RS/6000 AIX machine with a TCP/IP address of 193.1.1.11 and is listening for client connections on port 1500. Therefore, we chose `TCPIP` as our communication method, 193.1.1.11 as our tcp server address, and 1500 as our tcp port. We used the node that we registered at the beginning in 8.1, “Registering a node on the TSM server for DB2 backups” on page 149.

For the `nodename`, we had a choice to set `PASSWORDACCESS` to `prompt` or to `generate`. For ease of management, we chose `GENERATE` as this allows Tivoli Storage Manager to generate a password automatically.

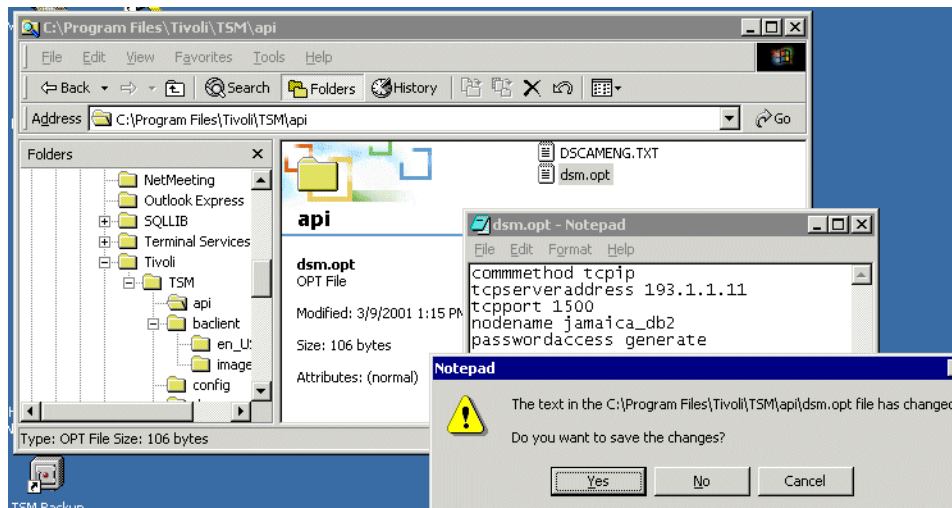


Figure 95. Editing the client options file

At this point we rebooted the system so that all the changes would take effect.

8.6 Generating the encrypted password

After rebooting the system, we opened up a command prompt and changed to the directory where the `dsmapiw.exe` executable resides (Figure 96). On our system this was 'c:\program files\sqllib\adsm'.

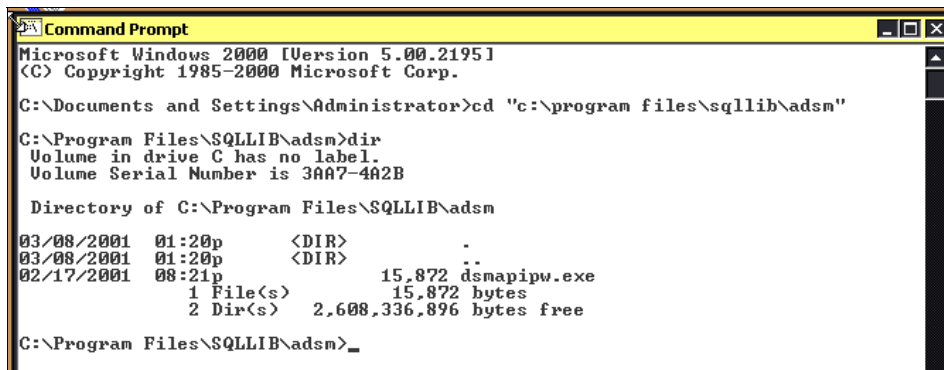
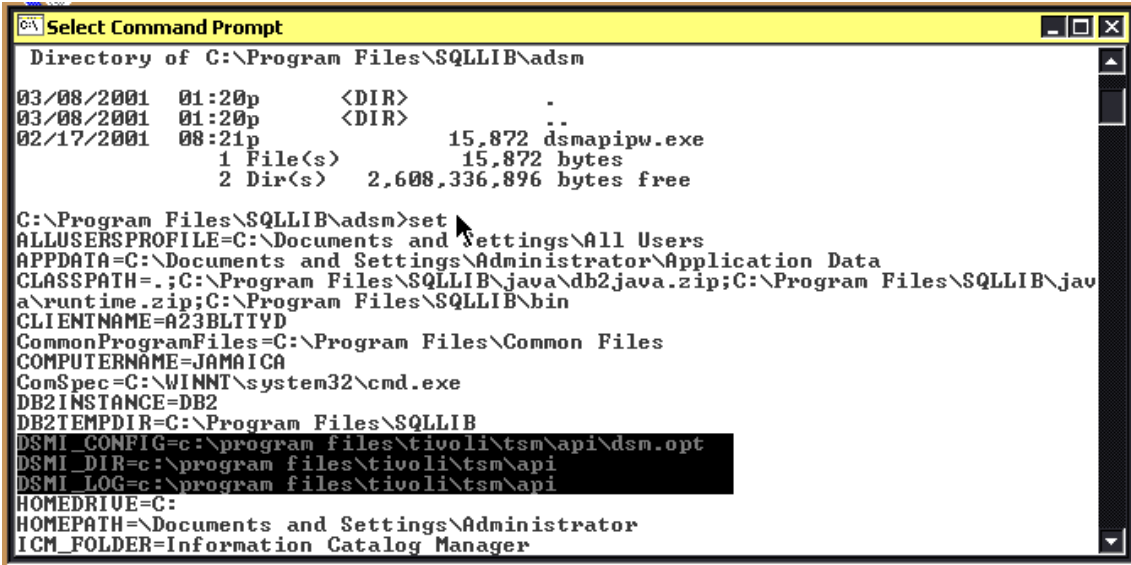


Figure 96. Location of DB2 UDB executable `dsmapiw.exe`

We confirmed that the environment variables were set by issuing the `set` command and then looking for the DSMI variables (Figure 97).

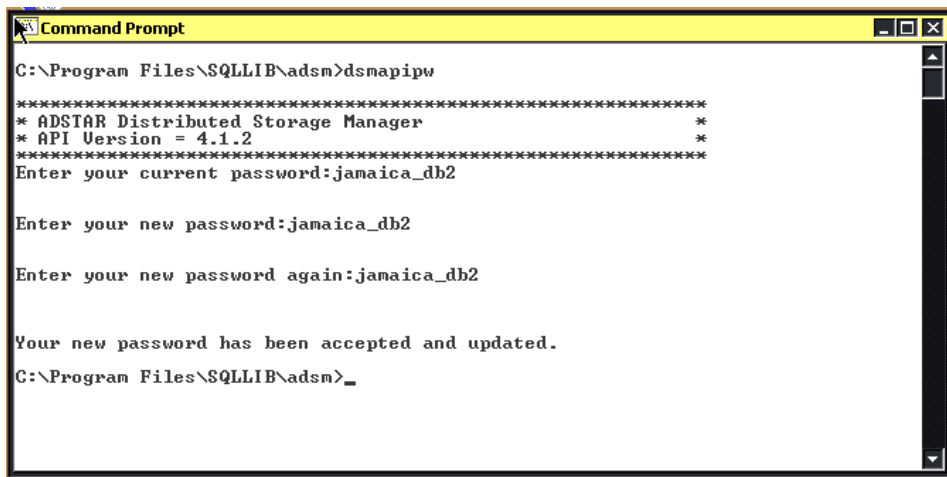


```
Select Command Prompt
Directory of C:\Program Files\SQLLIB\adsm
03/08/2001  01:20p    <DIR>      .
03/08/2001  01:20p    <DIR>      ..
02/17/2001  08:21p                15,872 dsmapiw.exe
           1 File(s)                15,872 bytes
           2 Dir(s)      2,608,336,896 bytes free

C:\Program Files\SQLLIB\adsm>set
ALLUSERSPROFILE=C:\Documents and Settings\All Users
APPDATA=C:\Documents and Settings\Administrator\Application Data
CLASSPATH=.;C:\Program Files\SQLLIB\java\db2java.zip;C:\Program Files\SQLLIB\java\runtime.zip;C:\Program Files\SQLLIB\bin
CLIENTNAME=A23BLITYD
CommonProgramFiles=C:\Program Files\Common Files
COMPUTERNAME=JAMAICA
ComSpec=C:\WINNT\system32\cmd.exe
DB2INSTANCE=DB2
DB2TEMPDIR=C:\Program Files\SQLLIB
DSMI_CONFIG=c:\program files\tivoli\tsm\api\dsm.opt
DSMI_DIR=c:\program files\tivoli\tsm\api
DSMI_LOG=c:\program files\tivoli\tsm\api
HOMEDRIVE=C:
HOMEPATH=\Documents and Settings\Administrator
ICM_FOLDER=Information Catalog Manager
```

Figure 97. Issuing `set` to confirm system variables

We then ran the `dsmapiw.exe` executable to generate the encrypted password (Figure 98). The password is encrypted in the Windows registry. **Note:** If `PASSWORDACCESS` is set to `PROMPT` and *NOT* generate, this command will succeed, but the password will *NOT* be generated. The `dsmapiw.exe` will ask for the current password, a new password, and then confirmation of the new password.



```
C:\Program Files\SQLLIB\adsm>dsmapiw
*****
* ADSTAR Distributed Storage Manager *
* API Version = 4.1.2 *
*****
Enter your current password:jamaica_db2


Enter your new password:jamaica_db2

Enter your new password again:jamaica_db2

Your new password has been accepted and updated.
C:\Program Files\SQLLIB\adsm>_
```

Figure 98. *dsmapiw.exe*

We can now run `db2adutl.exe query` to confirm that the configuration of system environment variables, client options file, and encrypted password is correct (Figure 99). The executable `db2adutl.exe` uses the `DSMI_CONFIG` to locate the correct options file to use to communicate with the Tivoli Storage Manager server. It uses the `DSMI_DIR` to locate the directory containing the API files. It uses the `DSMI_LOG` as the directory to write error messages in the `dsierror.log`. And it uses the encrypted registry password to authenticate with the Tivoli Storage Manager server. If successful, the `db2adutl.exe` command will output warning messages that there are no file space and no DB2 backup images on the Tivoli Storage Manager server. This is accurate since we have not yet performed any backups.



```
DB2 CLP
C:\Program Files\SQLLIB\adsm>cd "c:\program files\sqllib\bin"
C:\Program Files\SQLLIB\bin>db2adutl query
Warning: There are no file spaces created by DB2 on the ADSM server
Warning: No DB2 backup images found in ADSM for any alias.

C:\Program Files\SQLLIB\bin>_
```

Figure 99. *Warning messages that confirm configuration is correct*

If the setup is not correct, the `db2adutl.exe` will fail and provide an error message. It will also log information in the `dsierror.log`. Check Appendix B, “Troubleshooting” on page 277 for help in determining what is wrong.

8.7 Setting up the DB2 user exit for Tivoli Storage Manager

The user exit is an executable file that DB2 calls for archive and retrieval of log files. It enables DB2 to interact with storage devices that are not directly connected by the operating system. The user exit automates the removal of archive logs from local disk to remote storage (Tivoli Storage Manager, for example).

Note: The implementation of the user exit requires a full offline database backup to become effective. Ensure that you allow enough time to do this backup.

The user exit requires that the API SDK files is installed. This is required for both compiling the user exit and running it. To install the API SDK files, you must choose Custom installation when installing the client. See 8.3, “Installation” on page 151.

To use the user exit, you must perform the following steps:

- Modify the DB2 provided user exit for your environment
- Compile the user exit c program
- Place the resulting executable file in the ...\\SQLLIB\\bin directory
- Change the database parameter to utilize the user exit

8.7.1 Modifying the user exit

The sample user exit programs are located in the ...\\SQLLIB\\samples\\c directory. The one that can be used with Tivoli Storage Manager is named db2uext2.cadsm.

```
C:\Program Files\SQLLIB\samples\c>dir
Volume in drive C has no label.
Volume Serial Number is 3AA7-4A2B

Directory of C:\Program Files\SQLLIB\samples\c

03/21/2001  08:35a    <DIR>          .
03/21/2001  08:35a    <DIR>          ..
05/11/2000  12:35a             94,955 db2uext2.cadsm
02/17/2001  04:14a             69,751 db2uext2.cdisk
02/17/2001  04:14a             90,524 db2uext2.cxbsa
              3 File(s)      255,230 bytes
              2 Dir(s)  2,393,616,384 bytes free

C:\Program Files\SQLLIB\samples\c>
```

Instructions for compiling the user exit are contained within the db2uext2.cadsm file. You can open this file with a text editor.

To compile the user exit, the Tivoli Storage Manager API is required. You should use the same version of the Tivoli Storage Manager API for compiling the user exit that is used for running the user exit. In other words, since the 4.1.2.12 API is installed on the DB2 server, we installed the 4.1.2.12 API on the machine where the compiler is installed.

We copied the user exit file into the directory C:\Program Files\Microsoft® Visual Studio\VC98\Bin, and renamed it to db2uext2.c.

```
C:\Program Files\Microsoft Visual Studio\VC98\Bin>dir db2uext2.cadsm
Volume in drive C is WinNT
Volume Serial Number is E466-7802

Directory of C:\Program Files\Microsoft Visual Studio\VC98\Bin

05/11/00  12:35a                94,955 db2uext2.cadsm
           1 File(s)                94,955 bytes
                               4,058,756,096 bytes free

C:\Program Files\Microsoft Visual Studio\VC98\Bin>rename db2uext2.cadsm db2uext.c
C:\Program Files\Microsoft Visual Studio\VC98\Bin>
```

There are some user configurable variables that can be changed in the user exit using a text editor. Here is a sample extract of this:

```
/*== AUDIT_ERROR_PATH: Path where Audit and Error logs will reside ==*/
/*==                               Notes: 1. the path must exist ( the user exit will ==*/
/*==                               not create the path ) ==*/
/*==                               2. the path must end with a back slash ==*/
/*==                               3. the default is "c:\mylogs\" ==*/
/*==                               ==*/
/*== AUDIT_ERROR_ATTR: Standard C file open attributes for the Audit and ==*/
/*==                               Error logs ==*/
/*==                               Notes: 1. the default is "a" (text append) ==*/
/*==                               ==*/

#define BUFFER_SIZE          4096          /* transmit or receive the log */
                                   /* file in 4k portions */
#define AUDIT_ACTIVE         1             /* enable audit trail logging */
#define ERROR_ACTIVE         1             /* enable error trail logging */
#define AUDIT_ERROR_PATH    "c:\mylogs\\" /* path must end with a slash */
#define AUDIT_ERROR_ATTR    "a"          /* append to text file */
```

We changed the location of the audit and error logs to be the same directory that the dserror.log is written to, for ease of problem determination. This was

done by changing the default value of the AUDIT_ERROR_PATH from "c:\\mylogs\\" to "c:\\progra~1\\tivoli\\tsm\\api\\". After this change the value looked like this:

```
#define AUDIT_ERROR_PATH "c:\\progra~1\\tivoli\\tsm\\api\\" /* path must end with a slash */
```

8.7.1.1 Compiling the DB2 user exit

Because we were using Microsoft Visual C++®, we set the environment variables needed to run the command line compiler by executing the VCVARS32.BAT batch program located in the ...\\VC98\\bin directory.

```
C:\Program Files\Microsoft Visual Studio\VC98\Bin>dir vc*.bat
Volume in drive C is WinNT
Volume Serial Number is E466-7802

Directory of C:\Program Files\Microsoft Visual Studio\VC98\Bin

03/22/01  11:48a                989 VCVARS32.BAT
           1 File(s)                989 bytes
           4,058,756,096 bytes free

C:\Program Files\Microsoft Visual Studio\VC98\Bin>vcvars32
Setting environment for using Microsoft Visual C++ tools.
C:\Program Files\Microsoft Visual Studio\VC98\Bin>
```

Note: The instructions that come with the user exit are for the older V3.1 ADSM API. They show a different path and also say to use adsmv3.lib instead of tsmapi.lib. If you install the Tivoli Storage Manager API using the default path, the correct syntax to compile the user exit is as follows:

```
cl db2uext2.c -Ic:\progra~1\tivoli\tsm\api\include
-linkc:\progra~1\tivoli\tsm\api\lib\tsmapi.lib
```

This was our output from the command prompt when we compiled and linked the user exit program:


```

C:\Program Files\Microsoft Visual Studio\VC98\Bin>cl db2uext2.c -Ic:\progra-1\ti
voli\tsm\api\include -linkc:\progra-1\tivoli\tsm\api\lib\tsmapi.lib

Microsoft (R) 32-bit C/C++ Optimizing Compiler Version 12.00.8168 for 80x86
Copyright (C) Microsoft Corp 1984-1998. All rights reserved.

db2uext2.c
Microsoft (R) Incremental Linker Version 6.00.8168
Copyright (C) Microsoft Corp 1992-1998. All rights reserved.

/out:db2uext2.exe
c:\progra-1\tivoli\tsm\api\lib\tsmapi.lib
db2uext2.obj

C:\Program Files\Microsoft Visual Studio\VC98\Bin>

```

Compiling and linking the user exit generated the db2uext2.exe executable. We copied it into the ...\\sqllib\bin directory.

If there are multiple databases under one DB2 instance only one user exit program can be used for all the databases. As a result, all the user exit logs of these databases are written to the same logfile.

8.7.1.2 Enable the database for rollforward recovery

Finally, the DB2 environment must be customized to make use of the user exit program by setting the user exit parameter to ON in the database configuration. This will bring the database into rollforward mode so that full logfiles, which no longer contain active transactions, will automatically be moved off the system to Tivoli Storage Manager.

```
db2 update db cfg for sample using userexit on
```

```

db2 => update db cfg for sample using USEREXIT ON
DB20000I The UPDATE DATABASE CONFIGURATION command completed successfully.
DB21026I For most configuration parameters, all applications must disconnect
from this database before the changes become effective.

```

Attention: All applications need to disconnect from the database before the change becomes effective. After all applications are disconnected the database is in backup pending state and no new connections are allowed to the database until a full database backup is made.

8.8 DB2 Tivoli Storage Manager configuration values

Every DB2 database has four Tivoli Storage Manager configuration parameters that can be specified:

- TSM_MGMTCLASS

The TSM_MGMTCLASS parameter specifies to which management class database backups are bound. This value, if specified, takes precedence over both the default management class and an include statement. This parameter is different to the other three parameters in two important ways. This parameter only applies during backup operations (it is not used during restores) and it applies whether or not password access is set to prompt or generate. **Note:** This value has no affect on the user exit program.

The other three values are only used when PASSWORDACCESS PROMPT is used. When PASSWORDACCESS is set to PROMPT, these values are required and override the values from the client options file. If these values are set when PASSWORDACCESS is set to GENERATE, the DB2 backup or restore will fail:

- TSM_NODENAME

Specifies the nodename used to authenticate with the Tivoli Storage Manager server. See 4.2.1, “Registering a node with the Tivoli Storage Manager server” on page 34.

- TSM_PASSWORD

Specifies the password used to authenticate with the Tivoli Storage Manager server.

- TSM_OWNER

Specifies the owner of the backup object. This value is used to provide additional security for UNIX systems. This value does not apply to DB2 running on a Windows operating system.

For most purposes you should use passwordaccess generate for ease of management. The only time passwordaccess prompt is required is when performing a redirected restore on a UNIX system.

These values are set either from the DB2 command line processor using the command `update db cfg for database using TSM_<parameter> <value>` or from the Control Center. See 5.5, “Optional DB2 configurations” on page 69 for details and warnings regarding the use of these parameters.

8.9 Backing up DB2 using Tivoli Storage Manager

In this section, we examine how you can use Tivoli Storage Manager to back up DB2 databases. We cover:

- Offline backup
- Online backup
- Tablespace backup

8.9.1 Full offline backup

The DB2 backup utility can be used from the:

- Command line interface (CLI)
- Graphical user interface (GUI)
- Your own C, COBOL, or Fortran language program

Our examples use the command line and graphical user interfaces.

8.9.1.1 Preparatory steps

Before you can start a backup or recovery operation, make sure that the database manager is online. In the ...\`sqllib\bin` directory is an executable `db2start.exe`. This executable will start the database manager, if it is not running; or if it is running, it will indicate this.

```
C:\PROGRA~1\SQLLIB\BIN>db2start
SQL1026N The database manager is already active.
```

Another way to check is to see if the corresponding Windows service is running. With this method the Windows OS command `net start` will list the services running. Our instance is named DB2, so the corresponding Windows service is DB2 - DB2.

```
C:\PROGRA~1\SQLLIB\BIN>net start
These Windows 2000 services are started:

Alerter
Application Management
COM+ Event System
Computer Browser
DB2 - DB2
DB2 - DB2CTLSV
DB2 - DB2DAS00
DB2 JDBC™ Applet Server
```

The Tivoli Storage Manager server must be running, and the Tivoli Storage Manager API installed and configured.

8.9.1.2 Offline backup using the command line

This example shows an offline backup using the DB2 command line interface.

Log in to Windows with a user that has rights to run the DB2 Command Line Processor and make sure that all applications are logged off from the database you want to backup. Use the following DB2 command to verify that all applications are logged off:

```
db2 list applications for database sample
```

The result should be as follows:

```
db2 => list applications for database sample
```

Auth Id	Application Name	Appl. Handle	Application Id	DB Name	# of Agents
DB2ADMIN	FLTABLES.EXE	61	0901975E.480D.010326210930	SAMPLE	1

In our example, a user is connected to our SAMPLE database. Use the following DB2 force command to force the user off:

```
db2 "force application ( 61 )"
```

If there are too many connections to the database, use the following command to force all applications off the instance. **Warning:** if you have multiple databases, this command will force off all applications from all databases within the same instance:

```
db2 force application all
```

Start the backup operation from the command line:

```
db2 backup database sample use tsm
```

The use tsm option tells DB2 to use the Tivoli Storage Manager API to write the output backup file instead of using common devices.

Wait until a message like the following appears.

```
$ db2 backup db sample use tsm  
  
Backup successful. The timestamp for this backup image is : 20010307102944  
  
$
```

The backup operation has created an image file of your database, SAMPLE, and put it in Tivoli Storage Manager server storage.

8.9.1.3 Offline backup using the DB2 Control Center

This example shows an offline backup using the DB2 Control Center:

1. Start the DB2 Control Center. On a Windows system, you will find the Control Center under **Start->Programs->IBM DB2**.
2. Click down the tree until you reach the database folder.
3. Select the database you want to back up and right-click on the database name and select **Backup->Database** as shown in Figure 100.

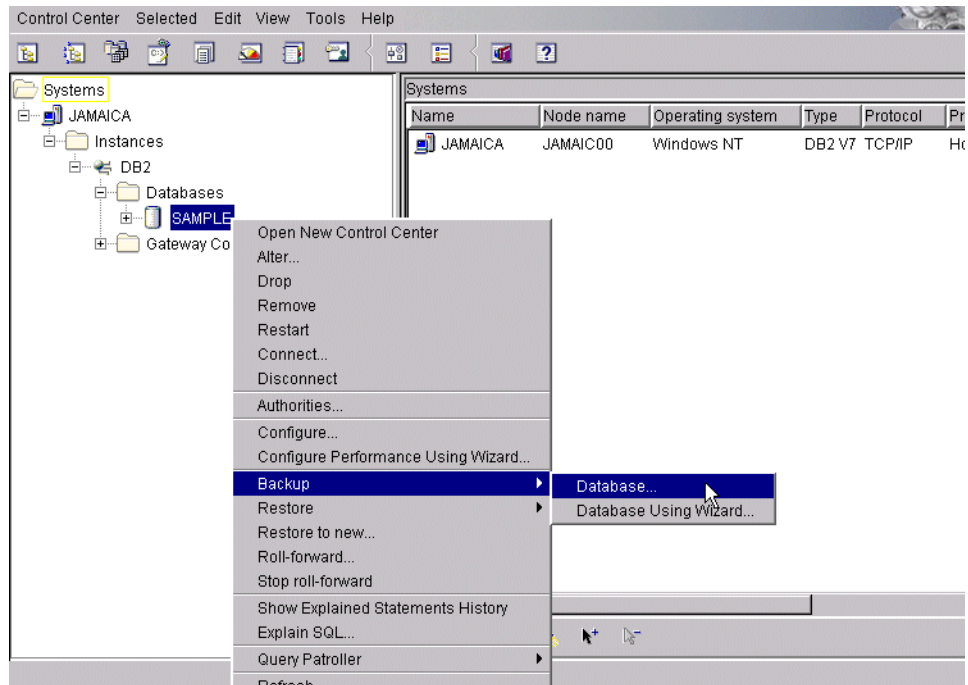


Figure 100. Control Center offline backup

The Backup Database window appears (Figure 101). Select Media Type: **Tivoli Storage Manager**.

Don't worry about the sessions and the options menu now. It will be described when you are doing online backups; see 8.9.2.3, "Online backup using the DB2 Control Center" on page 176.

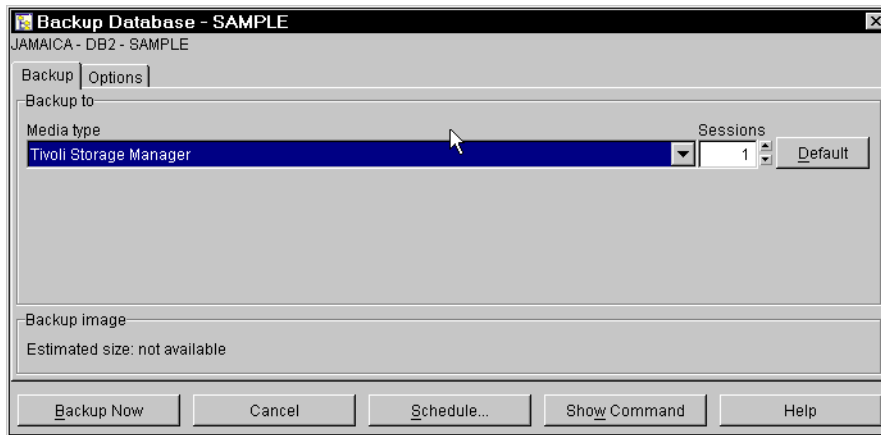


Figure 101. Select Tivoli Storage Manager option in Control Center GUI

Now click **Backup Now** to execute the backup. If there is a problem starting the backup an error message will appear. Depending on the system where the DB2 Control Center was started, the following windows may differ slightly.

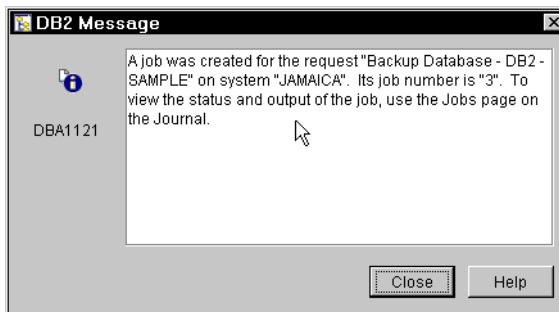


Figure 102. DB2 start backup window

The Close button must be selected to actually start the backup. If the window stays open the backup will not start.

After the backup is finished another window appears (Figure 103).

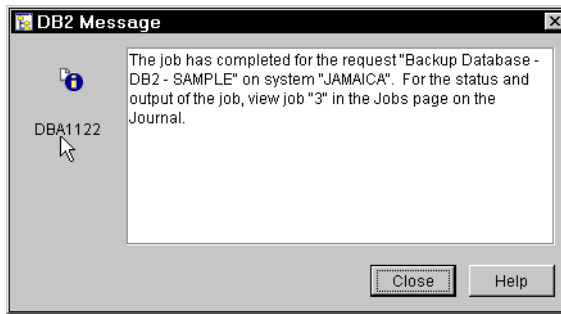


Figure 103. End of DB2 offline backup message using the DB2 GUI

The DB2 Journal will provide additional information about the success of the backup. The DB2 Journal is in the Tools menu of the DB2 Control Center. For more detailed information about the DB2 Journal, see 7.5.2, “Automate DB2 backup using DB2” on page 135.

8.9.2 Full online backup

DB2 online backup and recovery require the use of logfiles to enable roll-forward recovery. Our example shows both command line and GUI interfaces.

8.9.2.1 Preparatory steps

The user exit must be setup as described in 8.7, “Setting up the DB2 user exit for Tivoli Storage Manager” on page 166.

8.9.2.2 Online backup using the command line

Start the backup operation from the command line.

```
db2 backup database sample online use tsm
```

The *use tsm* option tells the DBMS to use the Tivoli Storage Manager API to write the output backup file to Tivoli Storage Manager instead of to local storage (disk or tape).

Wait until a message like the following appears:

```
db2 => backup db sample online use tsm

Backup successful. The timestamp for this backup image is : 20010328075649

db2 =>
```

The backup operation created an image file of your database, SAMPLE, and put it in Tivoli Storage Manager server storage.

8.9.2.3 Online backup using the DB2 Control Center

This example shows an online backup using the DB2 Control Center;

Start the backup operation from the GUI:

1. Start the DB2 Control Center. On a Windows system, you will find the Control Center under **Start->Programs->IBM DB2**. For UNIX systems, type `db2cc`.
2. Click down the tree until you reach the database folder.
3. Select the database you want to back up and right-click on the database name and select **Backup->Database** as shown in Figure 103.

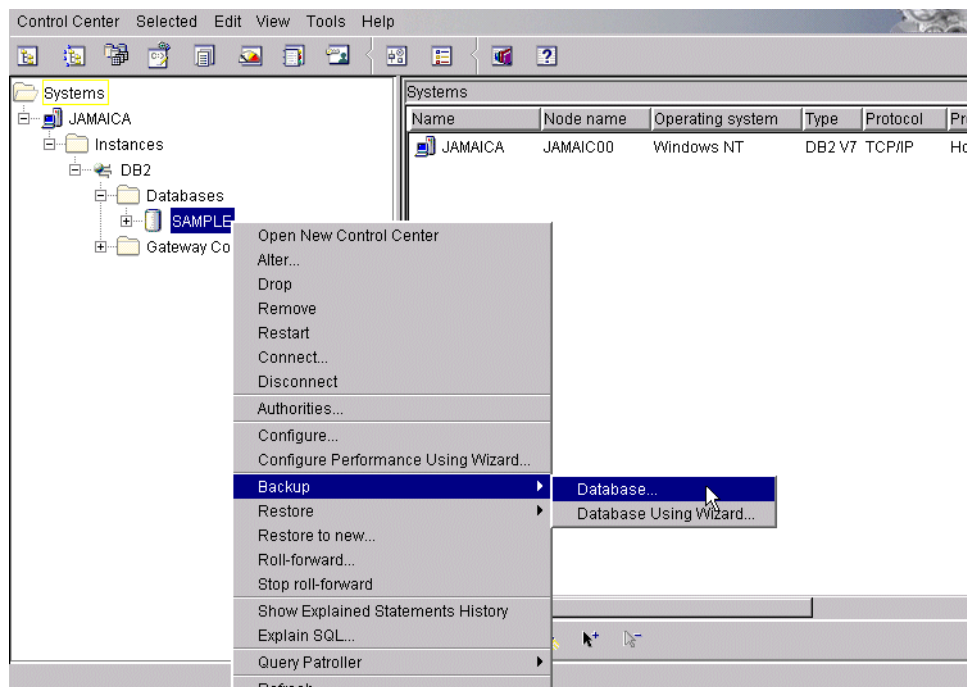


Figure 104. DB2 select database for online backup

The Backup Database window appears (Figure 105). Select Media Type: **Tivoli Storage Manager**.

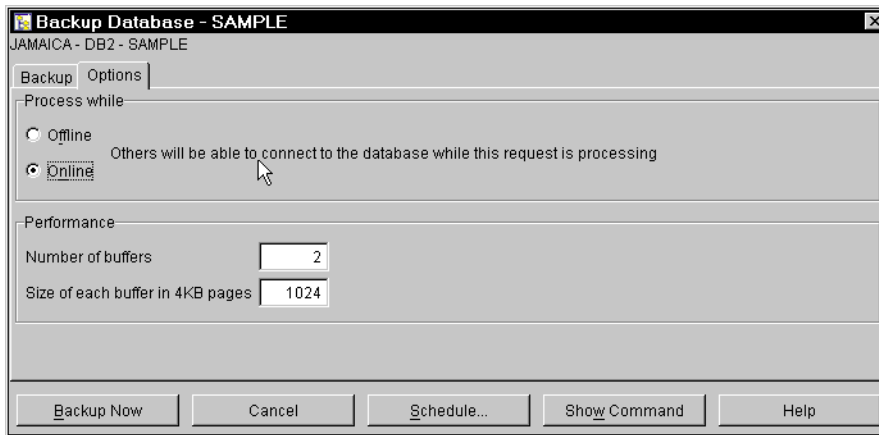


Figure 105. Select Tivoli Storage Manager option in backup GUI

Click the Options tab and select **Online** as shown in Figure 106.

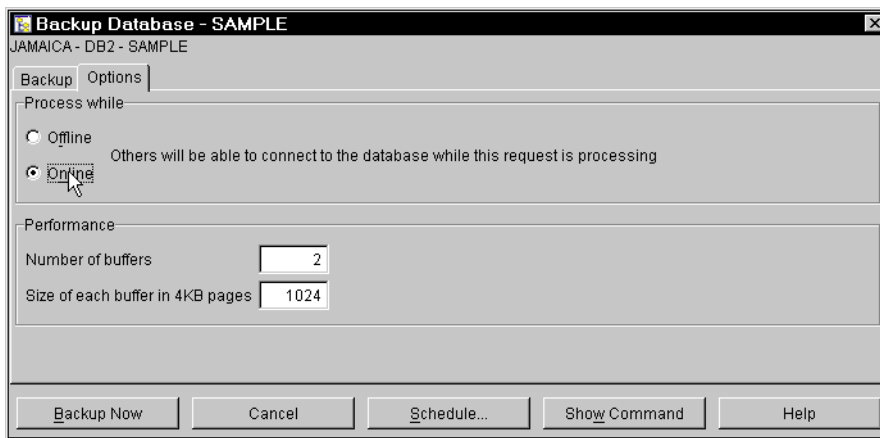


Figure 106. Select Online option in Options menu

Then click **Backup Now** to execute the backup. If there is a problem starting the backup, an error message will be generated. Depending on the system where the DB2 Control Center was started, the following windows may differ slightly.

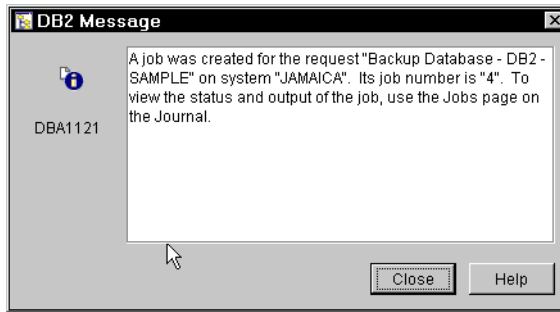


Figure 107. DB2 start backup window

The Close button must be pressed to actually start the backup. If the window stays open the backup will not start.

After the backup is finished another window appears (Figure 108).

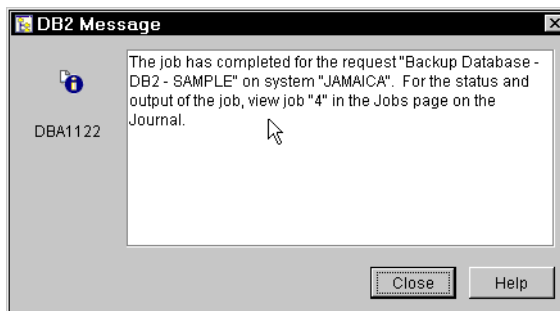


Figure 108. End of DB2 online backup message using the DB2 GUI

The DB2 Journal will provide additional information about the success of the backup. The DB2 Journal is in the Tools menu of the DB2 Control Center. The DB2 Journal is discussed in more detail in 7.5.2, "Automate DB2 backup using DB2" on page 135.

8.9.3 Tablespace backup

Tablespace backups can be done either offline or online. The only difference is the option on the backup command. In this chapter, we will not differ between the offline or online backup mode. We only discuss the online mode. For the offline mode, remove the online option from the command or select the Offline mode in the GUI. Remember when using the online option that the database must be enabled for roll-forward recovery. See 8.7.1.2, "Enable the database for rollforward recovery" on page 169.

8.9.3.1 Tablespace backup using the command line

Start the backup operation from the DB2 command line:

```
backup db sample tablespace ( userspacel ) online use tsm
```

The `use tsm` option tells DB2 to use the Tivoli Storage Manager API to write the output backup file instead of using common devices.

Wait until a message like the following appears:

```
db2 => backup db sample tablespace ( userspacel ) online use tsm
Backup successful. The timestamp for this backup image is : 20010328084316
db2 =>
```

The backup operation created an image file of your tablespace and put it in Tivoli Storage Manager server storage.

8.9.3.2 Tablespace backup using the DB2 Control Center

This example shows an online backup using the DB2 Control Center:

1. Start the DB2 Control Center. On a Windows system, you will find the Control Center under **Start->Programs->IBM DB2**. For UNIX systems, type `db2cc`.
2. Click down the tree until you reach the database folder.
3. Select the folder of your database you want to back up and right-click on the tablespace in the left frame as shown in Figure 109.

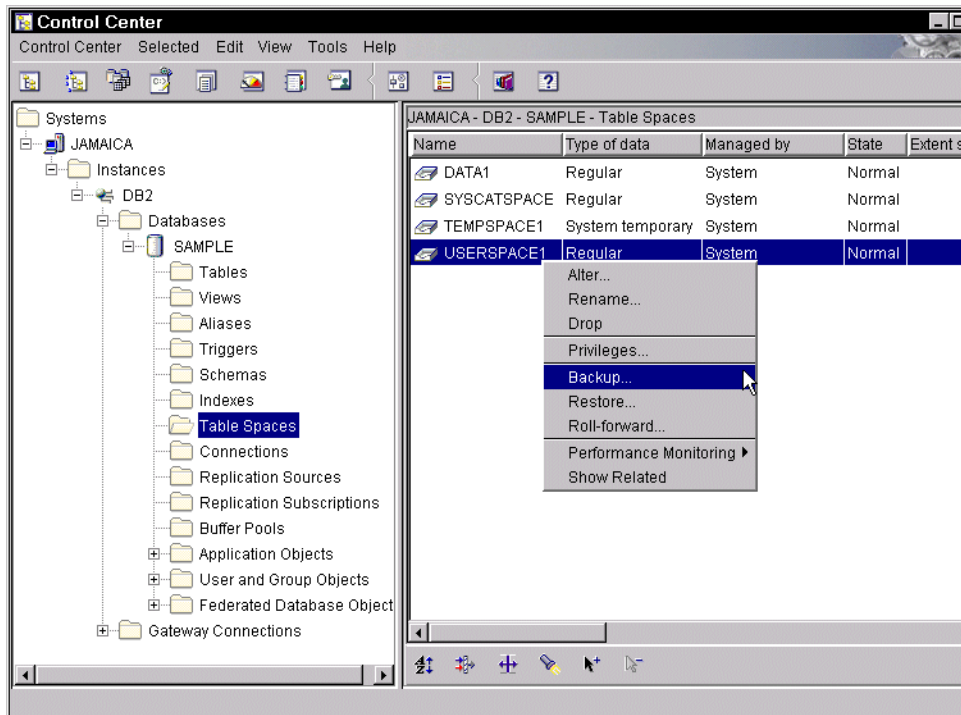


Figure 109. Select Tablespace Backup Menu

The Backup Database window appears (Figure 110). Select Media Type: **Tivoli Storage Manager**.

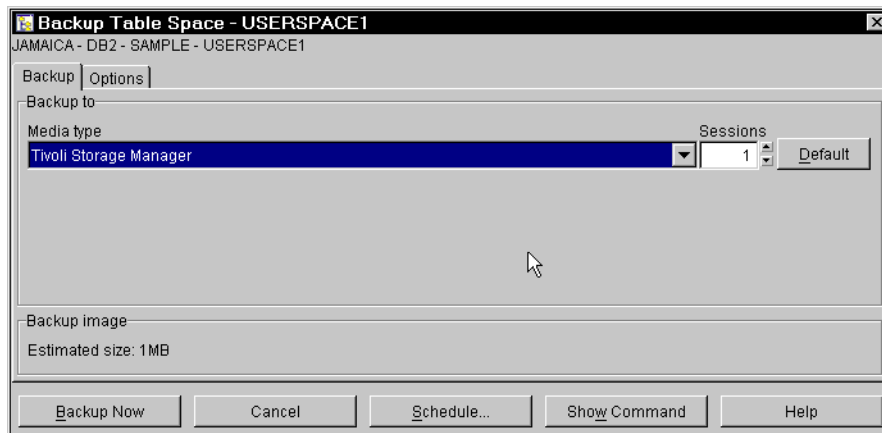


Figure 110. Select Tivoli Storage Manager option in backup GUI

Click the Options tab and select the Online radio button as shown in Figure 111.

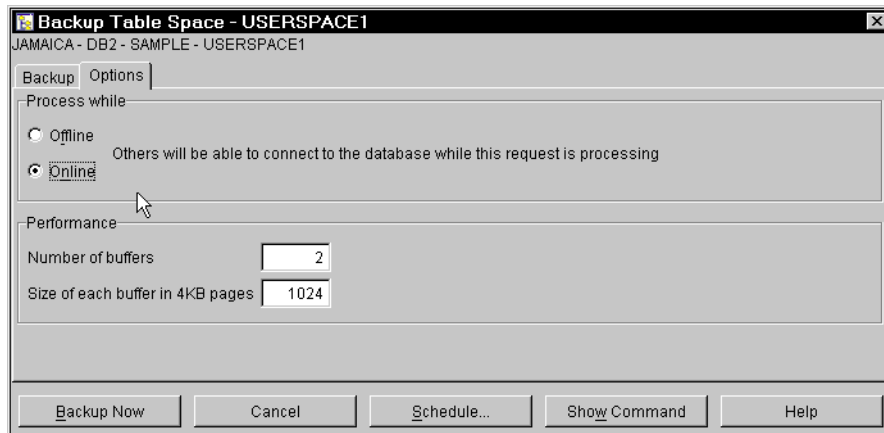


Figure 111. Select Offline or Online option in options menu

Then click **Backup Now** to execute the backup. If there is a problem starting, the backup an error message will be generated. Depending on the system where the DB2 Control Center was started, the following windows may differ slightly.

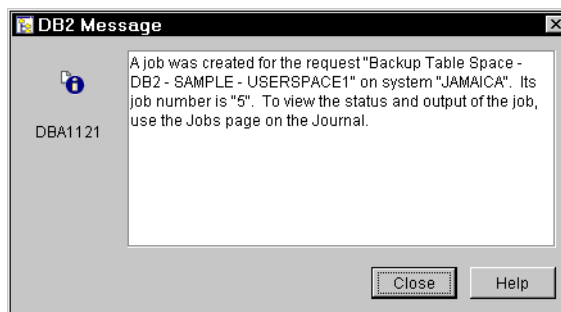


Figure 112. DB2 start backup window

You need to press the Close button to actually start the backup. If you leave the window open, the backup will not start.

After the backup is finished another window appears (Figure 113).

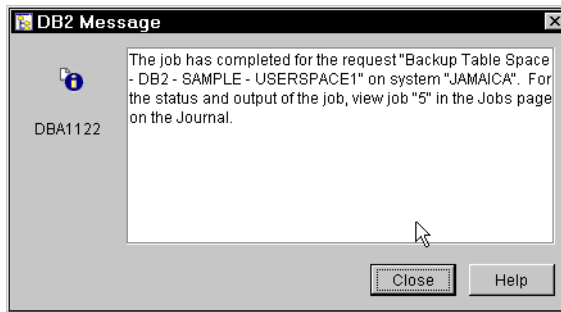


Figure 113. End of tablespace backup message using the DB2 GUI

The DB2 Journal will provide additional information about the success of the backup. The DB2 Journal is in the Tools menu of the DB2 Control Center. We discuss the DB2 Journal in more detail in 7.5.2, “Automate DB2 backup using DB2” on page 135.

Chapter 9. Day to day management: DB2 backups on Windows 2000

In this chapter we will discuss the daily work to be done to verify or monitor the success of the backups. We will not discuss DB2 management in general.

9.1 The db2adutl utility

The db2adutl.exe is a very useful command that comes with DB2 to manage old backups and logfiles that reside on the Tivoli Storage Manager server. The tool is located in the ...\\sqlib\\bin directory.

The executable db2adutl.exe provides the ability to query, extract, and delete backups, loadcopies and log archives from Tivoli Storage Manager. All backup file names include a time stamp and therefore are unique for every backup to Tivoli Storage Manager. The backup copies in Tivoli Storage Manager are therefore always active and never deleted by normal Tivoli Storage Manager file expiration processing. db2adutl.exe provides a way of deleting backups and log files that are no longer required. To automate the management of DB2 objects in Tivoli Storage Manager, you can run db2adutl.exe in a batch file using Windows or Tivoli Storage Manager scheduling. For example, you can maintain a certain number of generations of backups by using the KEEP n parameter on the DELETE option. This feature is only available for backups. Logfiles can only be deleted by specifying a range of logfile numbers.

Loadcopy are copies of images that are loaded into the database via the DB2 load utility. To also get the loaded data copied to Tivoli Storage Manager the `copy yes use tsm` option of the DB2 load utility must be specified. In case of a recovery this loadcopy will be needed for forward recovery. **Note:** The `copy yes` function is only valid if forward recovery is enabled.

Figure 114 shows the syntax to invoke the db2adutl.exe utility. This information was obtained by running the executable without any parameters.

Usage: db2adutl.exe

```
{ QUERY [ [ [ TABLESPACE | FULL | LOADCOPY ]
          [ SHOW INACTIVE ] ] |
          [ LOGS [ BETWEEN <Sn1> AND <Sn2> ] ] ] |

EXTRACT [ [ [ TABLESPACE | FULL | LOADCOPY ]
           [ SHOW INACTIVE ]
           [ TAKEN AT <timestamp> ] ] |
          [ LOGS [ BETWEEN <Sn1> AND <Sn2> ] ] ] |

DELETE [ [ [ TABLESPACE | FULL | LOADCOPY ]
          [ KEEP <n> |
            OLDER [THAN] [ <timestamp> |
                          <n> DAYS ] ] |
          [ TAKEN AT <timestamp> ] ] |
        [ LOGS [ BETWEEN <Sn1> AND <Sn2> ] ] ] }

VERIFY [ [ TABLESPACE | FULL ]
        [ SHOW INACTIVE ]
        [ TAKEN AT <timestamp> ] ] |

[ {DATABASE | DB} <database name> ]

[ NODE <n> ]

[ PASSWORD <password> ]

[ NODENAME <nodename> ]

[ OWNER <owner> ]

[ WITHOUT PROMPTING ]
```

Figure 114. Syntax of db2adutl

QUERY queries either tablespace, full backup, loadcopy images, or log archives. If you type query only, it returns all four.

EXTRACT, if you do not qualify it, displays the name of each backup image and prompts you to extract the image. If you qualify the command you reduce the scope of the search. For this option, time stamps do not apply to log archives.

DELETE, if you do not qualify it, works similarly to extract, except that backup images are marked inactive and archive objects (log files) are deleted.

Note: The Tivoli Storage Manager server can be configured to retain deleted backup versions. In this case, backup objects remain on the server even if you delete them by using db2adutl.exe. If you do not want to keep these inactive objects, you have to configure your Tivoli Storage Manager server to either of the following:

- Set the number of backup versions, if client data is deleted (VERDELETED), to 0.
- Or
- Set the length of time to retain the last remaining copy of a deleted file (REONLY) to 0.

For further Tivoli Storage Manager server considerations, see 4.5.2.1, “Backup copy group considerations” on page 48.

The `VERIFY` qualifier, performs consistency checking on the tablespace or backup copy that is on the Tivoli Storage Manager server. This parameter causes the entire backup image to be transferred over the network.

The `KEEP` qualifier `n` keeps only the newest `n` images.

The `OLDER THAN` qualifier (`THAN` is optional) deletes any images older than the specified time stamp (this must be the complete time stamp string) or the specified number of days. If you automated the deletion of backups using this qualifier be aware that backups may fail and your automation may delete your last valid backups.

The `TAKEN AT <timestamp>` qualifier is the time stamp with which you want to perform the operation. This qualifier is not allowed with `LOGS`, because `LOGS` have no time or date association.

The `SHOW INACTIVE` qualifier also shows the inactive versions of full database backup, tablespace or loadcopy images that have been deleted with `db2adutl` and therefore marked deleted on the Tivoli Storage Manager server. This qualifier will only work if the Tivoli Storage Manager server is configured to allow inactive versions.

The `DATABASE` qualifier reduces the search to a particular database.

The `WITHOUT PROMPTING` qualifier disables the prompt. Be very careful when using this option, because if you issue the wrong commands it may delete more than you expected to delete.

9.2 Information about backups using TSM commands

With a Tivoli Storage Manager administrator userid and password you have access to the Tivoli Storage Manager server database. The same information that is available with `db2adutl.exe` can also be directly acquired using Tivoli Storage Manager SQL commands. The `db2adutl.exe` presents the Tivoli

Storage Manager information in a more readable view, but for problem determination it can be useful to view the data directly.

We will only give a few examples of what can be done using the SQL queries. These queries can become as complex as needed.

To login to the server open an *administrative client session*. From a machine with the administrative client installed enter the following command:

```
dsmadm
```

You will be prompted for the node name and the password. Any administrator node has privileges to issue the SELECT command.

```
C:\Program Files\Tivoli\TSM\baclient>dsmadm
Tivoli Storage Manager
Command Line Administrative Interface - Version 4, Release 1, Level 2.12
(C) Copyright IBM Corporation, 1990, 1999, All Rights Reserved.

Enter your user id:  jamaica_db2
Enter your password:  *****

Session established with server BRAZIL: AIX-RS/6000
Server Version 4, Release 1, Level 2.0
Server date/time: 03/28/2001 11:09:58 Last access: 03/28/2001 11:09:24

tsm: BRAZIL>
```

Type help to get a first help on what possible commands you can use. Every backup or logfile is indexed in the Tivoli Storage Manager server database. To get a list of the backups which already exist for the API client, enter the following command:

```
select * from backups where node_name='<client API node name>'
```

Note: The <client API node name> must be in uppercase.

See Figure 115 for the results of the command.

```

tsm: BRAZIL>select * from backups where node_name='JAMAICA_DB2'
ANR2963W This SQL query may produce a very large result table, or may require a
significant amount of time to compute.

Do you wish to proceed? (Yes (Y)/No (N)) y

      NODE_NAME: JAMAICA_DB2
      FILESPACE_NAME: /SAMPLE
      STATE: ACTIVE_VERSION
      TYPE: FILE
      HL_NAME: \NODE0000\
      LL_NAME: FULL_BACKUP.20010322170159.1
      OBJECT_ID: 47185
      BACKUP_DATE: 2001-03-22 17:03:41.000000
      DEACTIVATE_DATE:
      OWNER:
      CLASS_NAME: DEFAULT

```

Figure 115. Output of Tivoli Storage Manager Select command

To get more detailed output the query can be extended by combining more search queries together as in the next example:

```

tsm: BRAZIL>select * from backups where node_name='JAMAICA_DB2' and filespace_name='/SAMPLE'

```

There is also a command line interface to the Tivoli Storage Manager server combining the logon process and the Tivoli Storage Manager command in a single step. This can be useful if you are creating shell scripts that need information from the Tivoli Storage Manager server directly:

```

dsmadm -id=jamaica_db2 -password=jamaica_db2 select * from backups

```

When using the DB2 user exit, the logfiles will be backed up into the archive copy group (see 8.7, “Setting up the DB2 user exit for Tivoli Storage Manager” on page 166). To get information about the archived items, the query must be changed to select items from the archives table. See the following example.

```
tsm: BRAZIL>select * from archives where node_name='BRAZIL_DB2'  
ANR2963W This SQL query may produce a very large result table, or may require a  
significant amount of time to compute.
```

```
Do you wish to proceed? (Yes/No) y
```

```
      NODE_NAME: BRAZIL_DB2  
FILESPACE_NAME: /SAMPLE  
      TYPE: FILE  
      HL_NAME: /NODE0000/  
      LL_NAME: S0000000.LOG  
      OBJECT_ID: 19487  
ARCHIVE_DATE: 2001-03-01 19:52:47.000000  
      OWNER: db2inst1  
DESCRIPTION: Log file for DB2 database SAMPLE  
      CLASS_NAME: DEFAULT
```

```
      NODE_NAME: BRAZIL_DB2  
FILESPACE_NAME: /SAMPLE  
      TYPE: FILE  
      HL_NAME: /NODE0000/  
      LL_NAME: S0000000.LOG
```

9.3 Verification of DB2 backups

In this section we discuss the different methods of verifying the DB2 backups.

The most important and only way to see if a backup can be used for restore is to actually restore the backup. At least once a backup should be restored to test whether the backup and the restore processes are valid. There is no way to restore an unsuccessful, incomplete or missing backup. See Chapter 10, “Recovering DB2 UDB databases” on page 213 for further information.

Once the backup process itself is validated by doing a restore, the main purpose of this chapter is to detail the other possibilities for verifying the success of the backup and verifying the backup itself.

9.3.1 Verify backup using db2ckbkp

The db2ckbkp.exe checks the integrity of the backup image and determines whether or not it can be restored. Some or all parts of the backups can be checked. The backup image must reside physically on the disk. This can be a backup that was initially taken to disk, or one that was extracted from Tivoli Storage Manager storage using db2adutl extract. For this example, we first extracted a backup image from Tivoli Storage Manager storage using the db2adutl extract command.

```

Retrieving full database backup information. Please wait.

full database backup image:
.\SAMPLE.0\DB2\NODE0000\CATN0000\20010328\113323.001
, Node: 0
.\SAMPLE.0\DB2\NODE0000\CATN0000\20010328\113323.002, Node: 0

Do you wish to extract ALL of these images (Y/N)? y

Writing to file:
.\SAMPLE.0\DB2\NODE0000\CATN0000\20010328\113323.001
Writing to file:
.\SAMPLE.0\DB2\NODE0000\CATN0000\20010328\113323.002

```

In Figure 116, we ran the db2ckbkp.exe against the two backup images that were part of the backup set. The reason there are two images is because this particular backup was taken using two sessions.

```

C:\Program Files\SQLLIB\bin>db2ckbkp .\SAMPLE.0\DB2\NODE0000\CATN0000\20010328\1
13323.001 .\SAMPLE.0\DB2\NODE0000\CATN0000\20010328\113323.002

[1] Buffers processed: #####
[2] Buffers processed: #####

Image Verification Complete - successful.

C:\Program Files\SQLLIB\bin>

```

Figure 116. Example db2ckbkp with two backup images in the set

9.3.2 Verification using DB2 list history

The DB2 history file (db2rhist.asc) contains logged events from different administrative events like backup, load, drop table, reorganize, and so on.

The success of backups can be verified by listing the backup information using the DB2 list history command. See Figure 117 for an example (only one entry shown).

```
db2 => list history backup all for sample

                List History File for sample

Number of matching file entries = 1

Op Obj Timestamp+Sequence Type Dev Earliest Log Current Log Backup ID
-----
B D 20010328113323002 F A S0000000.LOG S0000000.LOG
-----

-----
Comment: DB2 BACKUP SAMPLE OFFLINE
Start Time: 20010328113323
End Time: 20010328113913
-----

00001 Location: bin\db2adsm.dll

db2 =>
```

Figure 117. DB2 list history

The interesting information is contained within the first line of each stanza. These lines show the Operation (Op:B=backup); the Object (Obj:D=Database); the Type (Type:F=Offline,N=Online); and the Device (Dev:A=TSM). For an in-depth description of the `list history` command, see the *DB2 UDB Command Reference V7, SC09-2951*.

9.3.3 Verification using db2adutl

In this section, we will focus on some more options of the `db2adutl` command and how they may be used to verify the DB2 backups.

The query option of the `db2adutl` utility can be used to get general information about which database backups, logfiles, and loadcopies reside on the Tivoli Storage Manager server.

```

C:\Program Files\SQLLIB\bin>db2adutl query

Query for database SAMPLE

Retrieving full database backup information.
full database backup image: 1, Time: 20010328113323
  Oldest log: S0000000.LOG, Node: 0, Sessions used: 2

Retrieving tablespace backup information.
No tablespace backup images found for SAMPLE

Retrieving load copy information.
No load copy images found for SAMPLE

Retrieving log archive information.
Log file: S0000000.LOG, Node: 0, Taken at: 2001-03-28-13.29.31

```

Figure 118. Example db2adutl query

A possible test would be to download a backup from the Tivoli Storage Manager server and see if the whole file is readable, for which the db2adutl extract option must be used. However, this is not a practical method for most databases!

```

C:\Program Files\SQLLIB\bin>db2adutl extract

Query for database SAMPLE

Retrieving full database backup information. Please wait.

full database backup image:
  .\SAMPLE.0\DB2\NODE0000\CATN0000\20010328\113323.001
, Node: 0
  .\SAMPLE.0\DB2\NODE0000\CATN0000\20010328\113323.002, Node: 0

Do you wish to extract ALL of these images (Y/N)? y

Writing to file:
  .\SAMPLE.0\DB2\NODE0000\CATN0000\20010328\113323.001
Writing to file:
  .\SAMPLE.0\DB2\NODE0000\CATN0000\20010328\113323.002

```

Figure 119. Example db2adutl extract

The best way to verify the backup is the verify option of the db2adutl utility. It is basically a way to run the db2ckbkp functionality through Tivoli Storage Manager to verify if the image is going to be restorable.

The whole image will be read from the Tivoli Storage Manager server into local memory buffer (not the whole image at once, but piece by piece). There

is nothing written to local disk, but the implications of the data transfer over the LAN should be calculated.

db2adutl verify attempts to read through the image in the same way that a true restore does. It will verify that all of the required objects (such as, media header, DB configuration, DB history, list of tablespace, tablespace definitions, and so on) exist in the object. The tool performs numerous checks, but it will not and cannot actually check the integrity of the userdata that exists in the backup image.

```
C:\Program Files\SQLLIB\bin>db2adutl verify full taken at 20010328113323 db sample

Query for database SAMPLE
Retrieving full database backup information. Please wait.

full database backup image:
.\SAMPLE.0\DB2\NODE0000\CATN0000\20010328\113323.001
, Node: 0
.\SAMPLE.0\DB2\NODE0000\CATN0000\20010328\113323.002, Node: 0

Do you wish to verify this image (Y/N)? y
Verifying file: .\SAMPLE.0\DB2\NODE0000\CATN0000\20010328\113323.001
=====
BUFFER 0
=====
***BUFFERS 1 - 41 deleted to save space in this redbook.***
=====
BUFFER 42
=====

Read 0 bytes, assuming we are at the end of the image
Image Verification Complete - successful.
.\SAMPLE.0\DB2\NODE0000\CATN0000\20010328\113323.002
=====
BUFFER 0
=====
***BUFFERS 1-42 deleted to save space in this redbook.***
=====
BUFFER 43
=====

WARNING only partial image read, bytes read: 28672 of 4194304
Read 0 bytes, assuming we are at the end of the image

Image Verification Complete - successful.
```

Figure 120. Example *db2adutl verify*

The warning message shown in Figure 120 is misleading and can be ignored. The last line is the important one and it indicates that the verification was successful.

9.4 Deletion of backups and logfiles

After some time, the database backups may not be needed anymore. The time when they will expire depends on the requirements users have on their data and the service level agreements drawn up between users and database administrators. Sometimes, special laws (for example, financial ones) require that backups must be accessible for many years. In other, very dynamic environments, backups older than one week may be of no value anymore.

Warning: Every customer installation has different data retention requirements, but it is vital that the database and Tivoli Storage Manager administrators understand fully the implications of backup deletion and the policy management rules that are to be implemented. We cannot stress strongly enough that automated deletion should be planned and tested thoroughly.

Whatever the criteria, after some period, a backup will eventually become eligible for deletion. The backup and any corresponding logfiles will need to be deleted, usually to free storage if the databases are large.

9.4.1 Deletion of backups

If you need to delete old backups that reside on the Tivoli Storage Manager server, the `db2adutl` utility with the `delete` qualifier can be used. There is no other easy way to delete old backups. There are three options for the delete qualifier.

- TAKEN AT <timestamp>
- OLDER [THAN] [<timestamp> | n DAYS]
- KEEP n

We strongly recommend you use the KEEP n option. This will ensure that the last n successful backups will be kept. With the other options the last remaining backup could be accidentally removed.

DB2 backup images will NEVER expire from the Tivoli Storage Manager server, regardless of the policy retention settings, if the DB2 backup images are not deleted with `db2adutl delete`.

9.4.2 Deletion of logfiles

Logfiles are used to rollforward a database to a certain point beginning from a database backup. Each database has an oldest logfile associated with it. All logfiles from the oldest forwards must be available to ensure that the rollforward procedure will work.

The only logfiles that should be deleted are those that will not be needed by any of the existing database backups for rollforward.

To delete old backed up logfiles the *delete* qualifier of the db2adutl utility must be used. The deletion of logfiles is, itself, a little bit complicated, because there is no timestamp associated with backed up logfiles. The range of the logfiles must be specified (Figure 121).

```
C:\Program Files\SQLLIB\bin>db2adutl query
Query for database SAMPLE
Retrieving full database backup information.
  full database backup image: 1, Time: 20010328143808
  Oldest log: S0000012.LOG, Node: 0, Sessions used: 2
Retrieving tablespace backup information.
No tablespace backup images found for SAMPLE
Retrieving load copy information.
No load copy images found for SAMPLE
Retrieving log archive information.
  Log file: S0000010.LOG, Node: 0, Taken at: 2001-03-28-14.37.18
  Log file: S0000011.LOG, Node: 0, Taken at: 2001-03-28-14.37.19
  Log file: S0000011.LOG, Node: 0, Taken at: 2001-03-28-14.39.23
  Log file: S0000012.LOG, Node: 0, Taken at: 2001-03-28-14.39.39
  Log file: S0000013.LOG, Node: 0, Taken at: 2001-03-28-14.39.48
  Log file: S0000014.LOG, Node: 0, Taken at: 2001-03-28-14.40.43
C:\Program Files\SQLLIB\bin>db2adutl delete logs between S0000010.LOG and S0000011.LOG
without prompting
Query for database SAMPLE
Retrieving log archive information.
  Log file: S0000010.LOG, Node: 0, Taken at: 2001-03-28-14.37.18
  Log file: S0000011.LOG, Node: 0, Taken at: 2001-03-28-14.37.19
  Log file: S0000011.LOG, Node: 0, Taken at: 2001-03-28-14.39.23
C:\Program Files\SQLLIB\bin>
```

Figure 121. Example db2adutl delete logfiles

You may see duplication of log names, this is normal behavior and can happen when database backups are taken.

9.4.3 Auto deletion of backups and logfiles

There is no easy way to automate the deletion of the log files on the Windows NT or 2000 platforms. A resourceful Tivoli Storage Manager or database administrator could utilize the output of a db2adutl query to find the oldest database backup and then subtract one from the oldest log that the oldest backup requires. Then use that value to delete all the logs preceding this number. However, this would need careful handling if it were to be automated.

Otherwise, this is a manual process that must be done on a regular basis to avoid storing of large database copies that are no longer required.

9.5 Automation of DB2 backup

This chapter will show which possibilities there are for automation of the backup process. We will describe ways to automate the backup using Windows 2000, DB2 and Tivoli Storage Manager utilities.

The examples shown here include some simple command files written to automate parts of the backup process. The scripts do not include any error checking or automation recovery as those processes depend heavily on the configuration of your databases and how you want to structure your backup strategy.

9.5.1 Automate DB2 backup using Windows

This section will cover the use of the Windows utility *AT* to automate the backup of a DB2 database.

The Windows *AT* utility runs shell commands at specified dates and times. You can type `at /?` to view Help on the command. Additional Help with examples for this command can be found by searching **Start -> Help** from Windows.

On our machine, we created a folder `c:\scripts` to contain the files to be used for automation of the DB2 backups. In this folder, we have a script `db2online.cmd` that executes a db2 online backup. You must specify the `/c` option when using these scripts. Otherwise, the `db2cmd.exe` will never close, and you will have processes running in the background that you cannot stop.

```
set DB2INSTANCE=DB2

"C:\Program Files\SQLLIB\bin\db2cmd.exe" /c DB2.EXE backup db sample user db2admin
using itsosj online use tsm
```

We wanted to schedule this command to run every morning at 8:00am. To do this, we specified the following *AT* command.

```
C:\scripts>at 08:00 /every:M,T,W,Th,F,S,Su c:\scripts\db2online.cmd
Added a new job with job ID = 1
```

If you type *AT* without specifying any parameters, you see the current scheduled jobs.

```

C:\scripts>at
Status ID    Day                Time                Command Line
-----
1           Each M T W Th F S Su  8:00 AM            c:\scripts\db2online.cmd
C:\scripts>

```

9.5.2 Automate DB2 backup using DB2

Automating backups using DB2 Control Center is identical whether DB2 is running on Windows or UNIX. For more about UNIX, see 7.5.2, “Automate DB2 backup using DB2” on page 135.

9.5.3 Automate DB2 backup using Tivoli Storage Manager

In order to use the Tivoli Storage Manager scheduling services, you must perform configuration on both the Tivoli Storage Manager server and the Tivoli Storage Manager client.

9.5.3.1 Tivoli Storage Manager server steps for scheduling

On the Tivoli Storage Manager server you need to perform two commands.

1. DEFINE SCHEDULE

With this command you must specify the domain where the DB2 backup node belongs, ACTION=COMMAND and OBJECT=<path to backup command>. The other options regarding time and frequency can be tailored to your environment. See the Tivoli Storage Manager server product documentation for details, or type `HELP DEFINE SCHEDULE` at the Tivoli Storage Manager administrative prompt.

This will show you how to schedule the `c:\scripts\db2online.cmd` command that we previously scheduled using Windows AT scheduling in 9.5.1, “Automate DB2 backup using Windows” on page 195.

```

tsm: BRAZIL>define schedule api_domain db2_online_backup action=command
object="c:\scripts\db2online.cmd"
ANR2500I Schedule DB2_ONLINE_BACKUP defined in policy domain API_DOMAIN.

```

2. DEFINE ASSOCIATION

Now we associate this schedule with the node that will execute the command. Our Windows DB2 backups use JAMAICA_DB2 as the nodename, so that is what we used for this command.

```
tsm: BRAZIL>define association api_domain db2_online_backup jamaica_db2
ANR2510I Node JAMAICA_DB2 associated with schedule DB2_ONLINE_BACKUP in policy
domain API_DOMAIN.
```

9.5.3.2 Tivoli Storage Manager client steps for scheduling

Once the schedule has been defined on the Tivoli Storage Manager server, and the schedule has been associated with the nodename, the only thing left to do is setup a scheduler service.

We want to use a different scheduler service for the DB2 backups to make it easier to check whether the DB2 commands have been executed. In our Windows environment we chose scheduling mode client polling rather than server prompted, because if you have more than one scheduler service on a machine and you are using prompted scheduling mode, you *must* specify the TCPCLIENTPORT option in one of the scheduler service's to be different from the default. Otherwise, both services will try to listen on the same port. See *Tivoli Storage Manager for Windows Administrator's Guide*, GC35-0410.

The Tivoli Storage Manager Backup-Archive client provides a wizard to assist with installing a scheduler service. We chose to use the command line utility `dsmcutil.exe` located in the `...\baclient` directory. To keep this service separate from the backup client scheduler service, you must specify options for NAME, NODE, PASSWORD, OPTFILE, SCHEDLOG, and ERROR.LOG.

If you do not use different values for SCHEDLOG and ERROR.LOG, both scheduler services will write to the same log files.

```
C:\Program Files\Tivoli\TSM\baclient>dsmcutil install /name:"TSM Scheduler DB2" /node:"jamaica_db2"
/password:"jamaica_db2" /optfile:"c:\progra~1\tivoli\tsm\api\dsm.opt"
/schedlog:"c:\progra~1\tivoli\tsm\api\dsm Sched.log"
/errorlog:"c:\progra~1\tivoli\tsm\api\dsmerror.log"
```

After running that command a number of messages are sent to the console. This will let you know if there are any errors. The most important ones to see are the following:

```
The service was successfully installed.
Starting the 'TSM Scheduler DB2' service .....
The service was successfully started.
```

If you have problems you can easily remove the scheduler service by specifying:

```
dsmcutil remove /name:"Name of Scheduler Service".
```

To check which scheduler services are installed, use the `dsmcutil list` command.

```
C:\Program Files\Tivoli\TSM\baclient>dsmcutil list
```

```
Installed TSM Client Services:
```

1. TSM BAClient Scheduler
2. TSM Scheduler DB2

To verify the settings and configuration of our scheduler we used the `dsmcutil query /name:"Name of Scheduler Service"` command.

```
C:\Program Files\Tivoli\TSM\baclient>dsmcutil query /name:"TSM Scheduler DB2"
```

```
Service Name      : TSM Scheduler DB2
Logon Account     : LocalSystem
Start Type        : Demand
Current Status    : Started
TSM Client Service Registry Settings:

Client Service Type = Client Scheduler Service
Client Directory   = "C:\Program Files\Tivoli\TSM\baclient\dsmsvc.exe"
Options file       = c:\progra~1\tivoli\tsm\api\dsm.opt
Event Logging      = YES
TSM Client Node    = JAMAICA_DB2
Comm Protocol      = (value not currently set)
Server             = (value not currently set)
Server Port        = (value not currently set)
Schedule Log       = c:\progra~1\tivoli\tsm\api\dsmsched.log
Error Log          = c:\progra~1\tivoli\tsm\api\dsmerror.log
MSCS Enabled Node = (value not currently set)
Cluster name       = (value not currently set)
```

Upon successful installation of the scheduler service, the scheduler service will automatically start and connect to the Tivoli Storage Manager server to retrieve any associated schedules. The `dsmsched.log` for the scheduler service will show whether a schedule was retrieved from the Tivoli Storage Manager server. Our `dsmsched.log` shows that the `c:\scripts\db2online.cmd` will be executed in seven minutes.

```

C:\Program Files\Tivoli\TSM\api>type dsmsched.log
03/30/2001 11:33:28 Querying server for next scheduled event.
03/30/2001 11:33:28 Node Name: JAMAICA_DB2
03/30/2001 11:33:28 Session established with server BRAZIL: AIX-RS/6000
03/30/2001 11:33:28 Server Version 4, Release 1, Level 2.0
03/30/2001 11:33:28 Server date/time: 03/30/2001 11:35:02 Last access: 03/30/
2001 11:00:04

03/30/2001 11:33:28 --- SCHEDULEREC QUERY BEGIN
03/30/2001 11:33:28 --- SCHEDULEREC QUERY END
03/30/2001 11:33:28 Next operation scheduled:
03/30/2001 11:33:28 -----

03/30/2001 11:33:28 Schedule Name:      DB2ONLINE
03/30/2001 11:33:28 Action:          Command
03/30/2001 11:33:28 Objects:         c:\scripts\db2online.cmd
03/30/2001 11:33:28 Options:
03/30/2001 11:33:28 Server Window Start: 11:39:19 on 03/30/2001
03/30/2001 11:33:28 -----

03/30/2001 11:33:28 Command will be executed in 7 minutes.

```

Seven minutes later we checked the dsmsched.log again and saw that it had executed the command.

```

03/30/2001 11:40:28
Executing scheduled command now.
03/30/2001 11:40:28 Node Name: JAMAICA_DB2
03/30/2001 11:40:28 Session established with server BRAZIL: AIX-RS/6000
03/30/2001 11:40:28 Server Version 4, Release 1, Level 2.0
03/30/2001 11:40:28 Server date/time: 03/30/2001 11:42:02 Last access: 03/30/
2001 11:35:02

03/30/2001 11:40:28
Executing Operating System command or script:
c:\scripts\db2online.cmd
03/30/2001 11:40:28 Finished command. Return code is:
0
03/30/2001 11:40:28 Scheduled event 'DB2ONLINE' completed successfully.
03/30/2001 11:40:28 Sending results for scheduled event 'DB2ONLINE'.
03/30/2001 11:40:28 Results sent to server for scheduled event 'DB2ONLINE'.

```

9.5.4 Automate DB2 backup using a Windows service starting a script

This section shows how to use a Windows service to automatically run a Tivoli Storage Manage DB2 backup script. This method does not require specifying the password in the script. The service itself is started with a user who has DB2 administrator rights.

Following is a summary of the steps you should follow:

1. Create a script that the Tivoli Storage Manager server will start through the Tivoli Storage Manager Scheduler.
2. Configure the Tivoli Storage Manager Scheduler.
3. Create a backup script.
4. Install a service that will start the backup script with a defined user.
5. Test the configuration.

9.5.4.1 Creating a script to be started through the Tivoli Storage Manager Scheduler

Inform the Tivoli Storage Manager server administrators that you will be using a script for starting the backup. Create a script as shown called `db2_backup.cmd` and store it in the following directory `C:\Program Files\Tivoli\TSM\baclient\DB2\`. This script simply stops and starts the `DB2_Backup Service`. See 9.5.4.4, “Installing a service to start the backup script with a defined user” on page 201.

```
net stop "DB2_Backup"  
net start "DB2_Backup"  
sleep 60
```

9.5.4.2 Configuring the Tivoli Storage Manager Backup-Archive Scheduler

We will use the Tivoli Storage Manager client service configuration utility, `dsmcutil`. To install the Tivoli Storage Manager Scheduler, issue the following command:

```
dsmcutil install scheduler /name:"TSM Scheduler Service SERVERNAME_DB2"  
/clientdir:"C:\Program Files\Tivoli\TSM\baclient" /optfile:"C:\Program  
Files\Tivoli\TSM\baclient\DB2\dsm.opt" /node:SERVERNAME_DB2  
/password:xxxxxx /validate:yes /autostart:yes /startnow:no
```

This installs the service on the client `SERVERNAME`. If the client is connected to the Tivoli Storage Manager server, you will see the entry in the scheduler log, `dsmsched.log` as follows, indicating it is ready to start the script. You will also see that the Tivoli Storage Manager server is configured to start the script.


```

03/23/2006 14:08:45 Next operation scheduled:
03/23/2006 14:08:45 -----
03/23/2006 14:08:45 Schedule Name:      CS_CM_DAS_0200_DA
03/23/2006 14:08:45 Action:          Command
03/23/2006 14:08:45 Objects:         "C:\Program Files\Tivoli\TSM\baclient\DB2\db
03/23/2006 14:08:45 Options:
03/23/2006 14:08:45 Server Window Start:  02:00:00 on 03/24/2006
03/23/2006 14:08:45 -----
03/23/2006 14:08:45 Command will be executed in 11 hours and 57 minutes.

```

9.5.4.3 Creating the backup script

We created a simple backup script called `db2_backup_script.cmd`. Modify this script to specify the database you want to backup.

```

set DB2INSTANCE=DB2
C:
cd \Program Files\IBM\SQLLIB\bin\
db2cmd.exe /c DB2.EXE -v -zC:\DB2_backup_log\dsmdb2dts.log backup db
entw_dts online use tsm
db2cmd.exe /c DB2.EXE -v -zC:\DB2_backup_log\dsmdb2sfc.log backup db
entw_sfc online use tsm

```

Specify the `-c` option when using this script. Otherwise, the `db2cmd.exe` never closes, and you will have processes running in the background that you cannot stop. The `-v` option means "Echo current command" so that you can see in the output which command was started. The `-z` option saves all output to an output file. If the DB2 installation is located on another drive, then modify the script.

9.5.4.4 Installing a service to start the backup script with a defined user

For security reasons we recommend that you do not include user IDs and passwords in scripts. However, you have to specify the authentication to start the backup or use DB2 commands. To get around this, install a service called `DB2_Backup`, using the Windows resource kit tools `instsrv.exe` and `srvany.exe`, as follows:

1. Install the service using the Windows resource kit tools as follows.

```
instsrv DB2_Backup "C:\Program Files\Windows Resource Kits\Tools\srvany.exe"
```

This creates a simple service without any parameters.

- To add the parameters, edit the registry using `regedit`. In the entry `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\DB2_Backup\`, create a new key called `Parameters` and in this key, create a new string value called `Application`. The value you should add is `C:\Program Files\Tivoli\TSM\baclient\DB2\db2_backup_script.cmd`. This is shown in Figure 122.

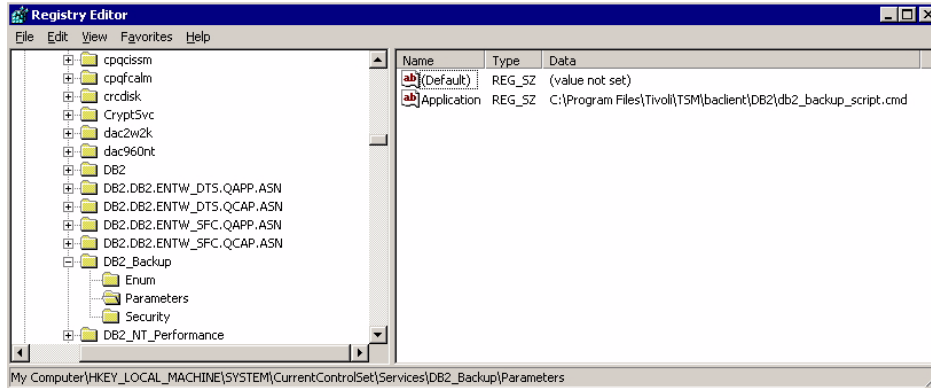


Figure 122. Entering the backup script in the registry

3. Modify the service itself in the Windows services applet. Right-click the newly installed service, **DB2_Backup**, and select **Properties**. Set the Startup type to `Manual` as Figure 123 shows.

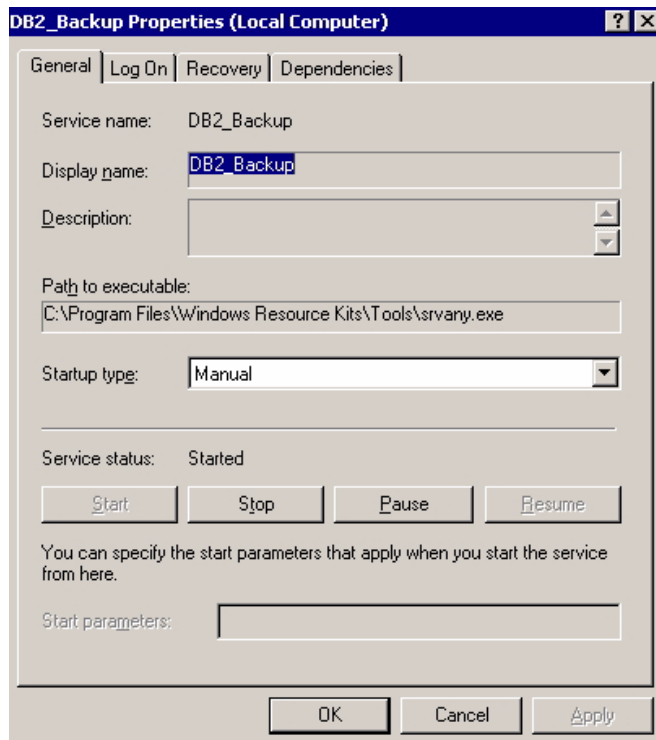


Figure 123. Set startup type to manual

4. Set the **Log On** properties. This is shown in Figure 124 where we specified the user name DASSCO with a password.

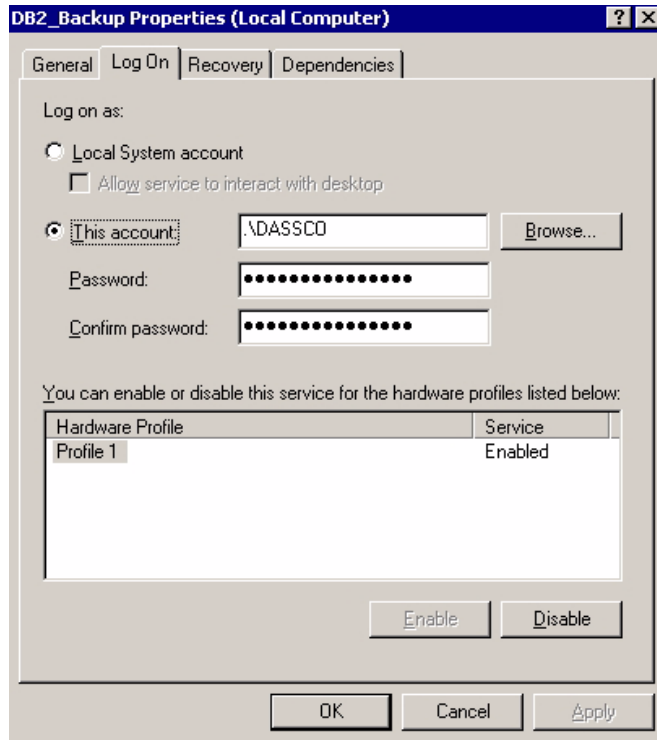


Figure 124. Set Log On properties

9.5.4.5 Testing the configuration

Next, perform a test backup to check if everything worked. Start the service DB2_Backup, and check if the Tivoli Storage Manager backup was successful. Use the `db2 list history backup` or `db2adutl query` command. For example, using the command `db2 list history backup all for entw_sfc` should display output similar to the following:

```
Op Obj Timestamp+Sequence Type Dev Earliest Log Current Log Backup ID
-----
B D 20060317021830001 N A S0038481.LOG S0038481.LOG
-----
Contains 4 tablespace(s):
00001 SYSCATSPACE
00002 USERSPACE1
00003 SYSTOOLSPACE
00004 TSASNQC
-----
Comment: DB2 BACKUP ENIW_SFC ONLINE
Start Time: 20060317021830
End Time: 20060317030947
Status: E
-----
EID: 19097 Location: bin\db2tsm.dll.
```

The interesting information is contained within the first line of each stanza. These lines show the following: Operation (Op:B=backup), the Object (Obj:D=Database), the Type (Type:F=Offline,N=Online), and the Device (Dev:A=TSM). For more details about the `list history` command, see *DB2 UDB Command Reference V7, SC09-2951*.

Using the `db2adutl query` shows output similar to the following:

```
Query for database ENIW_SFC

Retrieving FULL DATABASE BACKUP information.
1 Time: 20060324020547 Oldest log: S0038492.LOG DB Partition Number: 0 Sessions: 1
2 Time: 20060323020745 Oldest log: S0038490.LOG DB Partition Number: 0 Sessions: 1
3 Time: 20060322020043 Oldest log: S0038489.LOG DB Partition Number: 0 Sessions: 1
4 Time: 20060321020841 Oldest log: S0038487.LOG DB Partition Number: 0 Sessions: 1
5 Time: 20060320020238 Oldest log: S0038485.LOG DB Partition Number: 0 Sessions: 1
6 Time: 20060319020336 Oldest log: S0038483.LOG DB Partition Number: 0 Sessions: 1
7 Time: 20060318020634 Oldest log: S0038482.LOG DB Partition Number: 0 Sessions: 1
8 Time: 20060317021830 Oldest log: S0038481.LOG DB Partition Number: 0 Sessions: 1
9 Time: 20060316164250 Oldest log: S0038475.LOG DB Partition Number: 0 Sessions: 1
10 Time: 20060316020830 Oldest log: S0038474.LOG DB Partition Number: 0 Sessions: 1
```

The query option of the `db2adut1` utility can be used to show general information about which database backups, log files, and load copies reside on the Tivoli Storage Manager server. It also shows which log file was used for the backup. This is very important for a restore.

9.5.4.6 How does it work?

The Tivoli Storage Manager server can communicate with the started Tivoli Storage Manager Scheduler on the server you want to backup. Following is the process that occurs.

- When the Tivoli Storage Manager server initiates a backup, it starts the `db2_backup.cmd` that is located in the following directory:
C:\Program Files\Tivoli\TSM\baclient\DB2 on every server where we configured the Tivoli Storage Manager DB2 backup.
- The `db2_backup.cmd` stops and starts the DB2_Backup Service.
- The DB2_Backup Service starts the `db2_backup_script.cmd`, which itself runs the `DB2 backup` command.
- The DB2 backup creates logs in C:\DB2_backup_log\ called `dsmdb2<database>.log`. This process is shown in Figure 125.

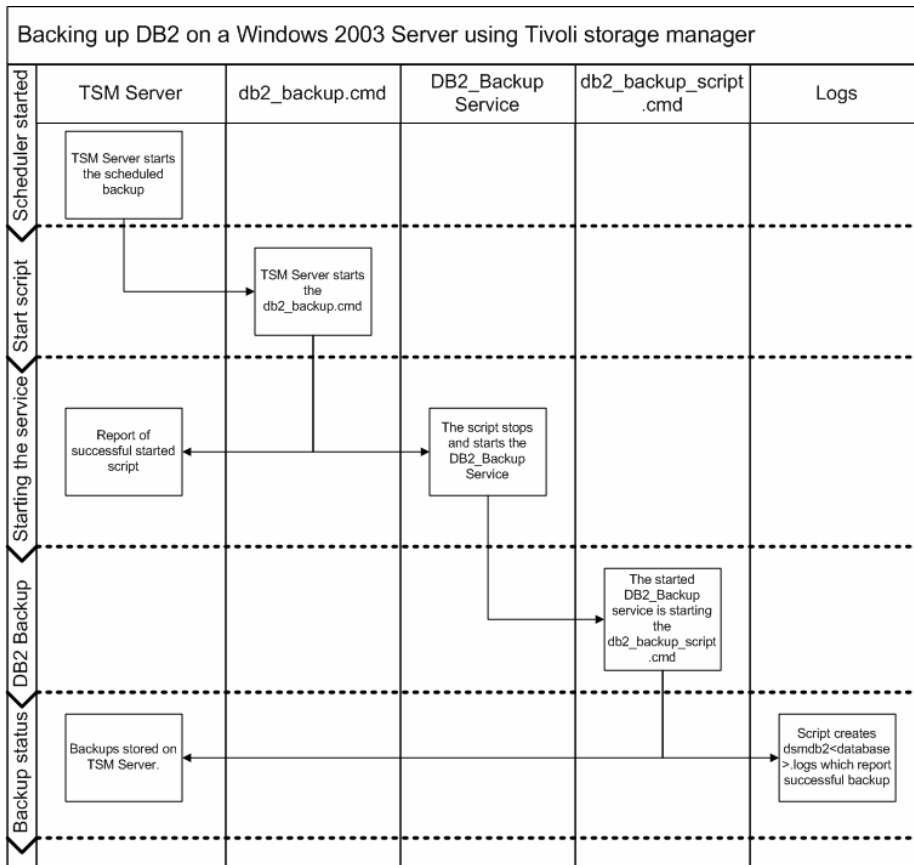


Figure 125. Automated backup process flow

9.6 Maintaining the history file

The history file (db2rhist.asc) contains a record of recovery and administrative events. Recovery events include full database and table space level backup, restore, and roll forward. Additional logged events include alter table space, run statistics, reorganize table, drop table, and load.

During normal operation the history file (db2rhist.asc) grows until pruned. As time goes on, entries in the history file are no longer required. You need to delete entries to keep the size of the file small. Note that deleting the entries only deletes the information, and **not** the backup. You still need to use db2adutl to delete backup and logs from Tivoli Storage Manager as seen in 9.4.1, “Deletion of backups” on page 193.

The DB2 command `list history backup all for dbname` can be used to view only the backup entries in the history file. However, you can only restrict which entries you delete with the `prune history` command by specifying a timestamp. You cannot simply remove the backup entries.

Neither does the history file get updated when you delete backups on the Tivoli Storage Manager server using the `db2adutl delete` command. The Control Center uses the history file to display backups for restore. **Note:** If you delete backups using `db2adutl` but do not prune the history file, the backups will still appear during a restore using the Control Center. However, upon executing the restore, it will fail with a message stating that the backup images are not available.

If you prune the history file and delete entries in the history file for backups that still reside on Tivoli Storage Manager storage, those backups can still be restored. The only issue is that the Control Center will not display them in a list. You must manually enter the backup information in the Control Center restore GUI by using the information from the `db2adutl query` command.

The DB2 product documentation has more detail on the `prune history` command. For example, to delete entries with timestamps on or before March 24, 2001 00:00 (20010324000000) on database BART, you can issue the following:

```
$ db2 prune history 20010324
DB20000I The PRUNE command completed successfully.
```


Note that you can specify the initial prefix for the timestamp. The minimum specification is `yyyy` where `yyyy` is the year.

Chapter 10. Recovering DB2 UDB databases

In this chapter, we will look at different methods of database recovery. Restoring a database would require us to select an image taken from a backup, using a timestamp to identify an image.

There are two ways of selecting an image:

- The `db2adutl`, as discussed in Chapter 7, “Day to day management: DB2 backups on UNIX” on page 119
- The `list history` command

For the examples in this chapter, we will introduce you to the `list history` command to identify which backup image to use. The command lists entries in the history file, which contain recovery events, such as backups, restore and roll-forward. It is also contains entries of additional events, such as load, run statistics, drop table, reorganize table, and alter tablespace.

The command line examples are done in the UNIX environment, but should be applicable to all environments. All commands in this example are done on the server. If you are managing the database remotely from a client machine, the `backup`, `restore`, and `rollforward` commands may require the `userid` and `password` for the database. See the *DB2 UDB Command Reference V7*, SC09-2951, for the syntax of the commands. For instance commands like `list history`, you must first issue the `attach` statement to connect to the node. Remote administration is not covered in this book. Please see the DB2 UDB documentation listed in Appendix G, “Related publications” on page 323.

For recovering the databases or tablespaces using the GUI Control Center, we will use a Windows client to recover a database from a remote server. For the examples, we will not be using wizards, so that we could show the options that are available in the restore database option.

10.1 Version recovery

Databases that are not enabled for roll-forward recovery can only do version recovery. They are said to be non-recoverable databases. Version recovery restores a previous version of the database using an image created by a backup. No log files are applied, and any transactions committed after the backup are lost. All users must disconnect from the database for version recovery.

10.1.1 Version recovery example

A container /db2dat1/bart/data01.dat for a tablespace data01 was accidentally deleted. DB2 reports the error on its diagnostic file db2diag.log as shown in Figure 126.

```
2001-03-14-13.53.26.749998 Instance:db2inst1 Node:000
PID:143556(db2agent (BART)) Appid:*LOCAL.db2inst1.010314195028
buffer_pool_services sqlbMSDoContainerOp Probe:810 Database:BART
DIA9999E An internal error occurred. Report the following error code :
"FFFFC11E".

2001-03-14-13.53.26.801830 Instance:db2inst1 Node:000
PID:143556(db2agent (BART)) Appid:*LOCAL.db2inst1.010314195028
buffer_pool_services sqlbMSDoContainerOp Probe:810 Database:BART
Error checking container 0 (/db2dat2/bart/data01.dat) for tbsp 3. Rc = FFFFE60A

2001-03-14-13.53.26.878856 Instance:db2inst1 Node:000
PID:143556(db2agent (BART)) Appid:*LOCAL.db2inst1.010314195028
buffer_pool_services sqlbMSStartPool Probe:800 Database:BART
DIA9999E An internal error occurred. Report the following error code :
"FFFFC11E".

2001-03-14-13.53.26.895437 Instance:db2inst1 Node:000
PID:143556(db2agent (BART)) Appid:*LOCAL.db2inst1.010314195028
buffer_pool_services sqlbMSStartPool Probe:800 Database:BART

Tablespace 3 (DATA01)

2001-03-14-13.53.26.961386 Instance:db2inst1 Node:000
PID:143556(db2agent (BART)) Appid:*LOCAL.db2inst1.010314195028
buffer_pool_services sqlbStartPools Probe:30 Database:BART

Tablespace DATA01 (3) is in state x0.
Starting it failed. rc=ffff c11e yyA.

2001-03-14-13.53.27.112612 Instance:db2inst1 Node:000
PID:143556(db2agent (BART)) Appid:*LOCAL.db2inst1.010314195028
base_sys_utilities sqlledint Probe:95 Database:BART

Database is started, but some pools remain offline.
```

Figure 126. Container error in db2diag.log

10.1.2 Version recovery using the command line

Follow these steps to recover the database:

1. Use the `list history` command to select a backup image you want to use for the restore. For database BART, our command will be:

```
$ db2 list history backup all for bart
```

An excerpt from the results of the command is shown in Figure 127. There can be several entries, so you need to scroll to select the backup image you need. You need the timestamp portion of the Timestamp+Sequence field or you can use the value of the Start Time field.

```

Op  Obj  Timestamp+Sequence  Type  Dev  Earliest Log  Current Log  Backup ID
-----
B  D    20010316165106001  N     A    S0000016.LOG S0000017.LOG
-----
Contains 5 tablespace(s):

00001 SYSCATSPACE
00002 USERSPACE1
00003 DATA01
00004 INDEX01
00005 LONG01
-----
Comment: DB2 BACKUP BART ONLINE
Start Time: 20010316165106
End Time: 20010316165140
-----
00030 Location: adsm/libadsm.a

Op  Obj  Timestamp+Sequence  Type  Dev  Earliest Log  Current Log  Backup ID
-----
B  D    20010316165106002  N     A    S0000016.LOG S0000017.LOG
-----

Comment: DB2 BACKUP BART ONLINE
Start Time: 20010316165106
End Time: 20010316165140
-----
00031 Location: adsm/libadsm.a

```

Figure 127. List history command result

2. Use the `restore` command, specifying the timestamp of the image you want to restore, and then reply `y` to continue:

```

$ db2 restore db bart use tsm taken at 20010316165106
SQL2539W Warning! Restoring to an existing database that is the same as the
backup image database. The database files will be deleted.
Do you want to continue ? (y/n) y
DB20000I The RESTORE DATABASE command completed successfully.

```

10.1.3 Version recovery using the Control Center

Follow these steps to recover the database using the Control Center.

1. Right-click the database, and then from the drop down menu select **Restore->Database** as shown in Figure 128.

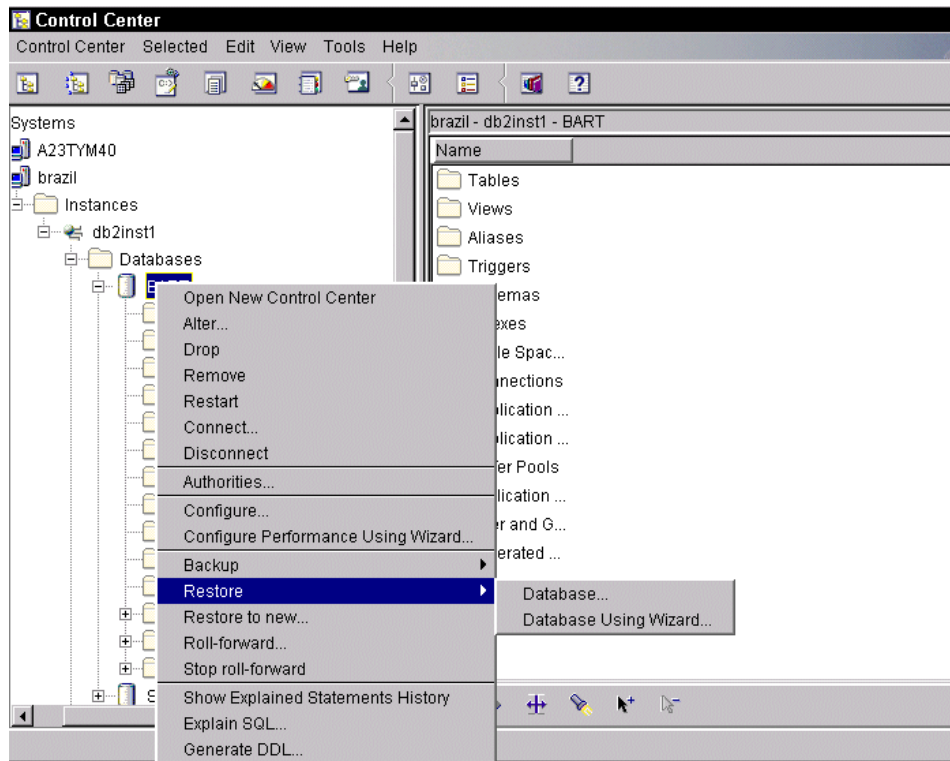


Figure 128. Version recovery using the Control Center

2. When the database image selection page appears as shown in Figure 129, select the database image you want to use for the restore, then select **OK**.

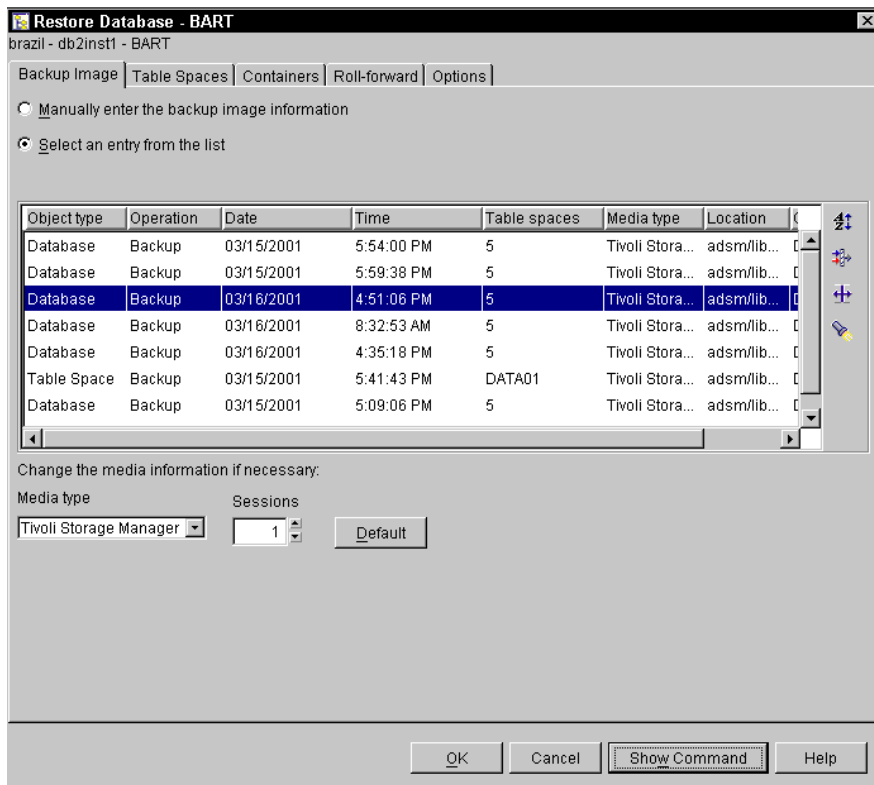


Figure 129. Backup image selection page

3. A dialog box will appear as shown in Figure 130, to confirm if the restore is successful. Select **Close**.

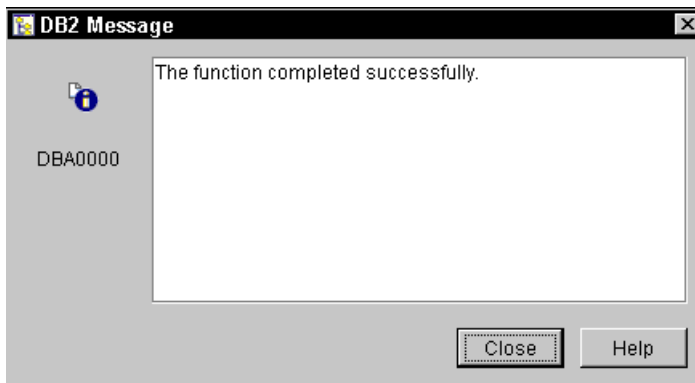


Figure 130. Version recovery successful dialog box

10.2 Database roll-forward recovery

Recoverable databases are databases that are enabled to do roll-forward recovery. With recoverable databases, you can restore the database and apply the logs to the point of failure. No transactions are lost.

10.2.1 Database roll-forward recovery example

A file system containing the containers became corrupted. You would see a similar error in db2diag.log to that shown in Figure 126 on page 214 when a user connects to the database. In addition, you can display the database configuration file to check the database status, for example:

```
$ db2 get db cfg for bart | more

      Database Configuration for Database bart

Database configuration release level      = 0x0900
Database release level                   = 0x0900

Database territory                        = US
Database code page                        = 819
Database code set                         = ISO8859-1
Database country code                     = 1

Dynamic SQL Query management             (DYN_QUERY_MGMT) = DISABLE

Directory object name                    (DIR_OBJ_NAME) =
Discovery support for this database       (DISCOVER_DB) = ENABLE

Default query optimization class          (DFT_QUERYOPT) = 5
Degree of parallelism                     (DFT_DEGREE) = 1
Continue upon arithmetic exceptions        (DFT_SQLMATHWARN) = NO
Default refresh age                       (DFT_REFRESH_AGE) = 0
Number of frequent values retained         (NUM_FREQVALUES) = 10
Number of quantiles retained              (NUM_QUANTILES) = 20

Backup pending                            = NO

Database is consistent                  = NO
Rollforward pending                   = TABLESPACE
Restore pending                       = NO
```

10.2.2 Database roll-forward recovery using the command line

Follow these steps to recover your database:

1. Use the `list history` command to select a backup image that you want to use for the restore. For database BART, our command will be:

```
$ db2 list history backup all for bart
```

An excerpt from the results of the command is shown in Figure 131. There can be several entries, so you need to scroll to select the backup image you need. You need the timestamp portion of the Timestamp+Sequence field or you can use the value in the Start Time field.

```
Op  Obj  Timestamp+Sequence  Type  Dev  Earliest Log  Current Log  Backup ID
-----
B   D   20010316165106001  N     A   S0000016.LOG S0000017.LOG
-----
Contains 5 tablespace(s):

00001 SYSCATSPACE
00002 USERSPACE1
00003 DATA01
00004 INDX01
00005 LONG01
-----
Comment: DB2 BACKUP BART ONLINE
Start Time: 20010316165106
End Time: 20010316165140
-----
00030 Location: adsm/libadsm.a

Op  Obj  Timestamp+Sequence  Type  Dev  Earliest Log  Current Log  Backup ID
-----
B   D   20010316165106002  N     A   S0000016.LOG S0000017.LOG
-----

Comment: DB2 BACKUP BART ONLINE
Start Time: 20010316165106
End Time: 20010316165140
-----
00031 Location: adsm/libadsm.a
```

Figure 131. List history command result

2. Use the `restore` command, specifying the timestamp of the image you want to restore, then reply `y` to continue:

```
$ db2 restore db bart use tsm taken at 20010316165106
SQL2539W Warning! Restoring to an existing database that is the same as the
backup image database. The database files will be deleted.
Do you want to continue ? (y/n) y
DB20000I The RESTORE DATABASE command completed successfully.
```

3. Use the `rollforward` command to apply the logs to completion:

```
$ db2 rollforward db bart to end of logs and stop

                                Rollforward Status

Input database alias              = bart
Number of nodes have returned status = 1

Node number                       = 0
Rollforward status                = not pending
Next log file to be read          =
Log files processed                = S0000000.LOG - S0000003.LOG
Last committed transaction        = 2001-03-14-22.09.08.000000

DB20000I  The ROLLFORWARD command completed successfully.
```

10.2.3 Database roll-forward recovery using the Control Center

There are a couple of ways to do a database recovery using the Control Center. One is to use the steps as described in 10.1.3, “Version recovery using the Control Center” on page 215, and then use the `Roll-forward` option in the drop down menu for the database. The second way is to do it all at the same time with the **Restore->Database**, which is discussed in the steps below:

1. Right-click the database, and then from the drop down menu select **Restore->Database** as shown Figure 132.

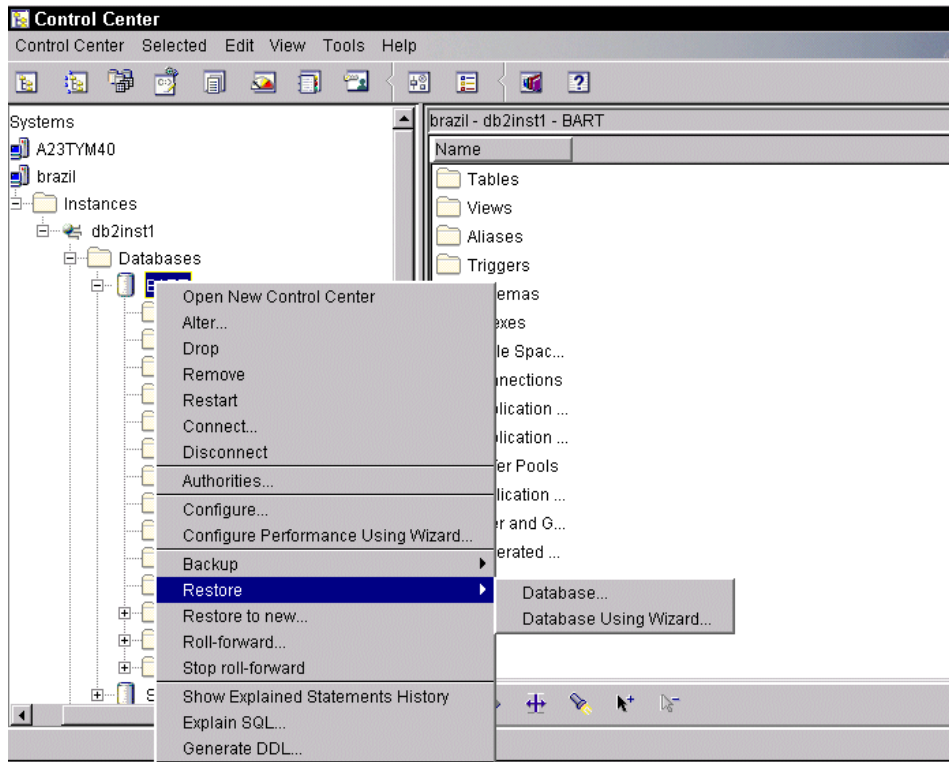


Figure 132. Database roll-forward using the Control Center

2. When the backup image selection page appears as shown in Figure 133, select the database image you want to use for the restore.

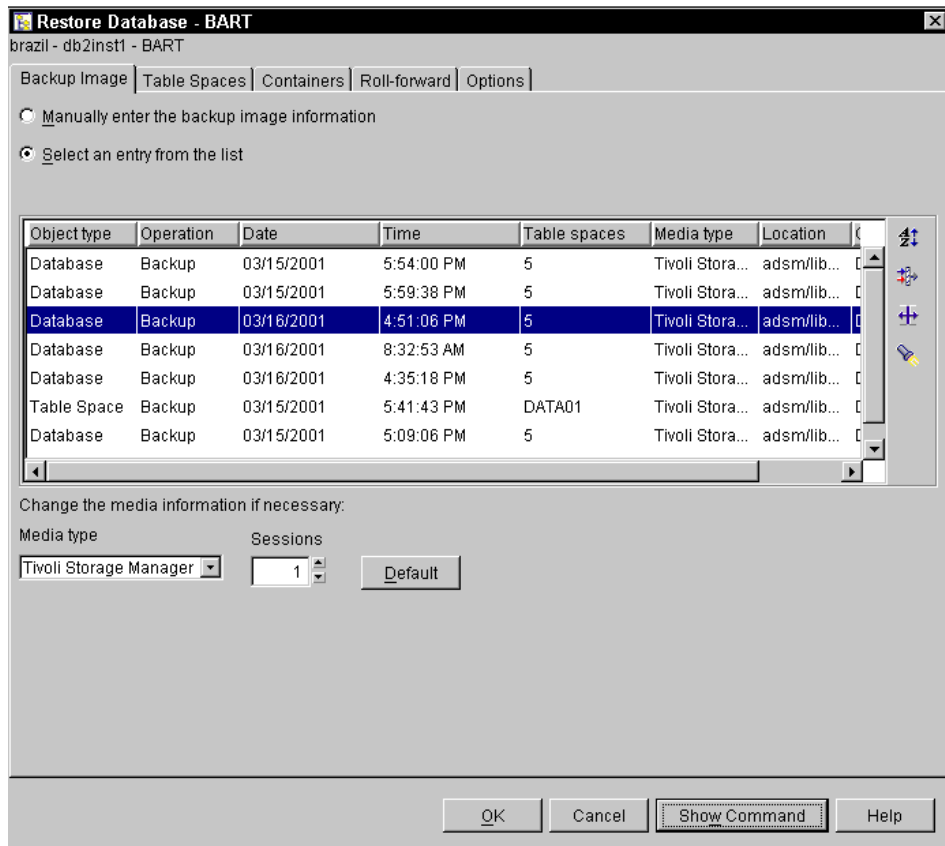


Figure 133. Backup image selection page

3. Click the **Roll-forward** tab as shown in Figure 134, check the *Roll-forward (re-apply logs)* option, uncheck the *Leave in roll-forward pending state* option, then select **OK**.

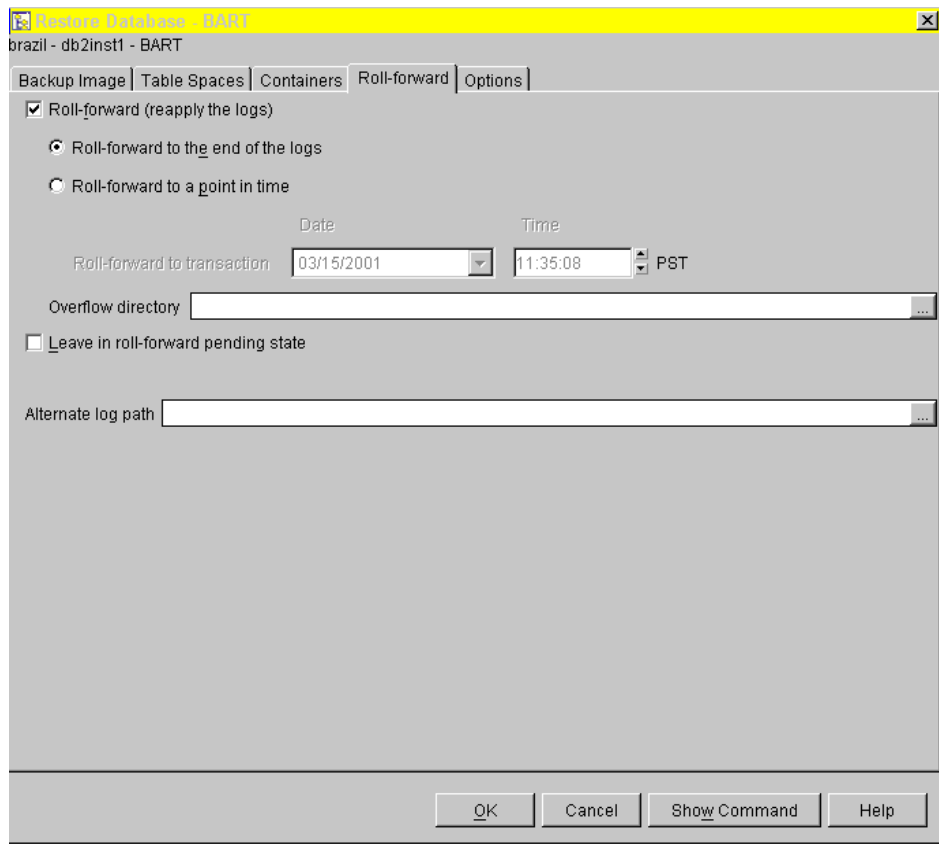


Figure 134. Roll-forward page

4. A dialog box appears as shown in Figure 135 to confirm if the roll-forward is successful. Select **Close**.

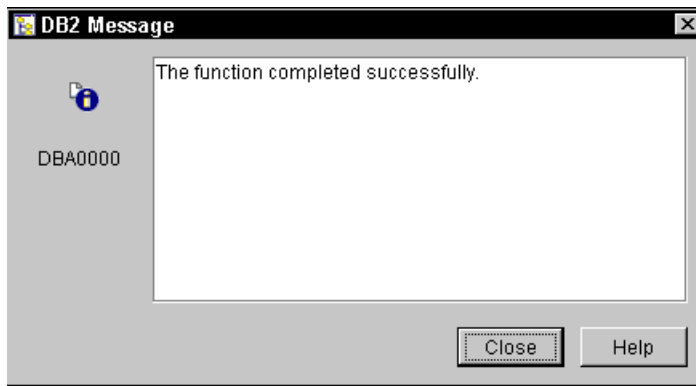


Figure 135. Roll-forward recovery successful dialog box

10.3 Tablespace roll-forward recovery

With recoverable databases, you have the option of recovering the whole database or only the tablespaces that need recovery. You can recover tablespaces either offline or online. You can use an image taken from either a previous tablespace backup or a database backup.

10.3.1 Tablespace roll-forward recovery example

Users complain that they cannot access a tablespace when querying a table. You did a `list tablespaces` as shown in Figure 136 and discovered that one of the tablespaces is offline. Then looking at the `db2diag.log`, you see an error when checking a container, and the tablespace 3 (DATA01) could not be started. The `db2diag.log` would be similar to the one shown in Figure 126 on page 214. You only need to recover the tablespace DATA01.


```

$ db2 list tablespaces

          Tablespaces for Current Database

Tablespace ID          = 0
Name                   = SYSCATSPACE
Type                   = System managed space
Contents               = Any data
State                  = 0x0000
  Detailed explanation:
    Normal

Tablespace ID          = 1
Name                   = TEMPSPACE1
Type                   = System managed space
Contents               = System Temporary data
State                  = 0x0000
  Detailed explanation:
    Normal

Tablespace ID          = 2
Name                   = USERSPACE1
Type                   = System managed space
Contents               = Any data
State                  = 0x0000
  Detailed explanation:
    Normal

Tablespace ID          = 3
Name                   = DATA01
Type                   = Database managed space
Contents               = Any data
State                  = 0x4000
  Detailed explanation:
    Offline

```

Figure 136. List tablespaces

10.3.2 Tablespace roll-forward recovery using the command line

In this example, the following steps recover the tablespace online:

1. Use the `list history` command to select a backup image that you want to use for the restore. For database BART, our command will be:

```
$ db2 list history backup all for bart
```

An excerpt from the results of the command is shown in Figure 137. There can be several entries, so you need to scroll to select the backup image you need. You need the timestamp portion of the Timestamp+Sequence field or you can use the value in the Start Time field.

```

Op Obj Timestamp+Sequence Type Dev Earliest Log Current Log Backup ID
--- ---
B D 20010316165106001 N A S0000016.LOG S0000017.LOG
-----
Contains 5 tablespace(s):

00001 SYSCATSPACE
00002 USERSPACE1
00003 DATA01
00004 INDEX01
00005 LONG01
-----
Comment: DB2 BACKUP BART ONLINE
Start Time: 20010316165106
End Time: 20010316165140
-----
00030 Location: adsm/libadsm.a

Op Obj Timestamp+Sequence Type Dev Earliest Log Current Log Backup ID
--- ---
B D 20010316165106002 N A S0000016.LOG S0000017.LOG
-----

Comment: DB2 BACKUP BART ONLINE
Start Time: 20010316165106
End Time: 20010316165140
-----
00031 Location: adsm/libadsm.a

```

Figure 137. List history command result

2. Use the `restore` command, specifying the timestamp of the image you want to restore, and then reply `y` to continue:

```

$ db2 "restore db bart tablespace (DATA01) online use tsm taken at 20010316165106
DB20000I The RESTORE DATABASE command completed successfully.
$

```

3. Use the `rollforward` command to apply the logs to completion:

```
db2 "rollforward db bart to end of logs and stop tablespace (data01) online"
```

Rollforward Status

```
Input database alias           = bart
Number of nodes have returned status = 1

Node number                   = 0
Rollforward status           = not pending
Next log file to be read     = S0000004.LOG
Log files processed           = -
Last committed transaction    = 2001-03-14-22.09.08.000000
```

```
DB20000I  The ROLLFORWARD command completed successfully.
```

10.3.3 Tablespace roll-forward recovery using the Control Center

You can do tablespace roll-forward recovery using the database drop down menu or the tablespace drop down menu.

When using the tablespace drop down menu, select **Tablespaces** under the database to display a list of tablespaces on the right pane, then select the tablespace to be recovered and right-click to drop down a menu as shown Figure 138.

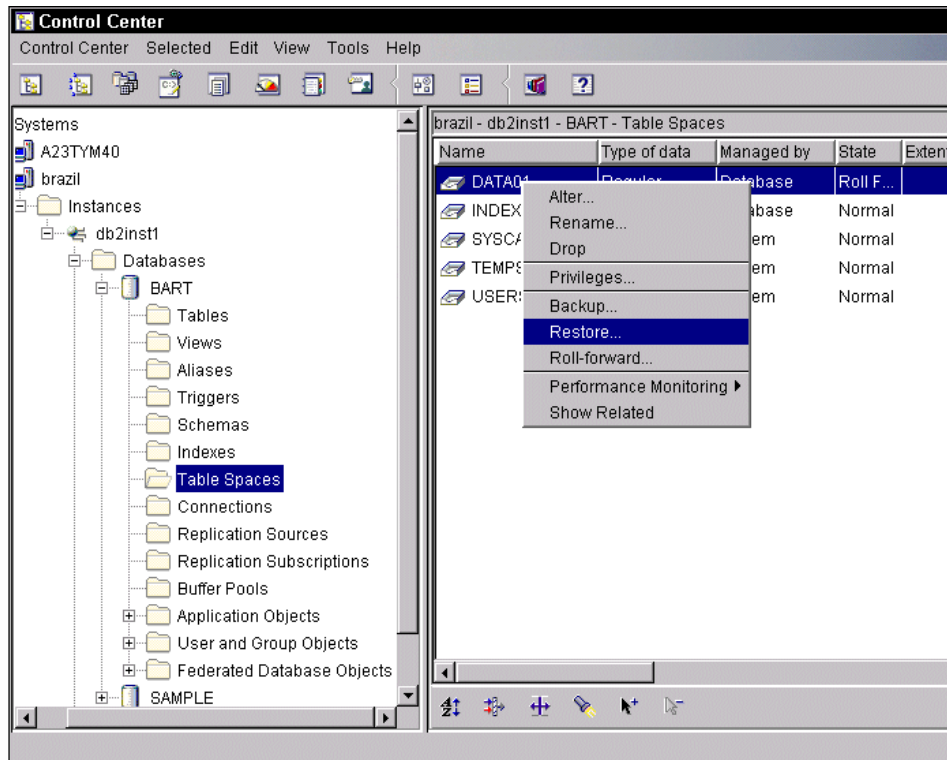


Figure 138. Restore tablespace using the Control Center

However, if you have not taken a previous tablespace backup, a warning dialog box appears informing you that there is no backup image for the database as shown in Figure 139. You can ignore the message, and continue.

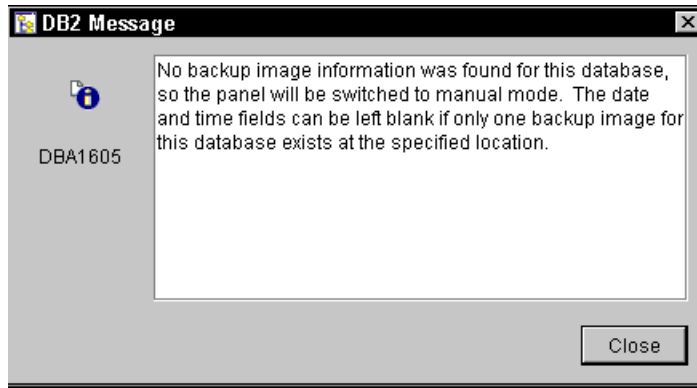


Figure 139. Warning dialog box

You will be presented with the next dialog as shown in Figure 140. The difficulty here is that you must manually specify the backup image timestamp.

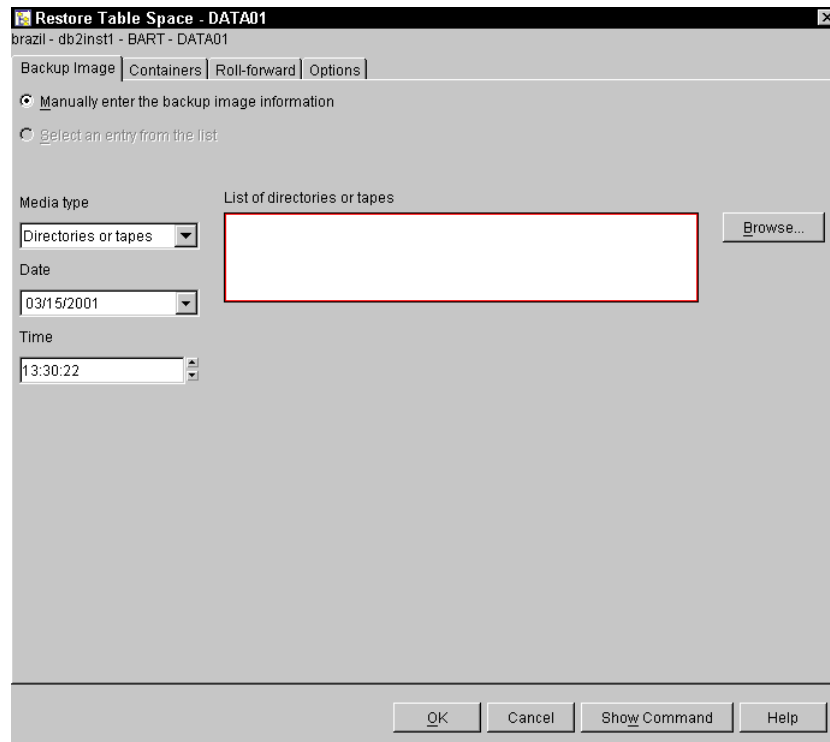


Figure 140. Manually entering the backup image

Using the database drop down menu is easier, and it presents you with options to use either a database backup image or a tablespace image. It also lets you restore more than one tablespace at a time.

For this example, we will be using the database drop down menu, as discussed in the following steps:

1. Right-click the database, and then from the drop down menu select **Restore->Database** as shown in Figure 141.

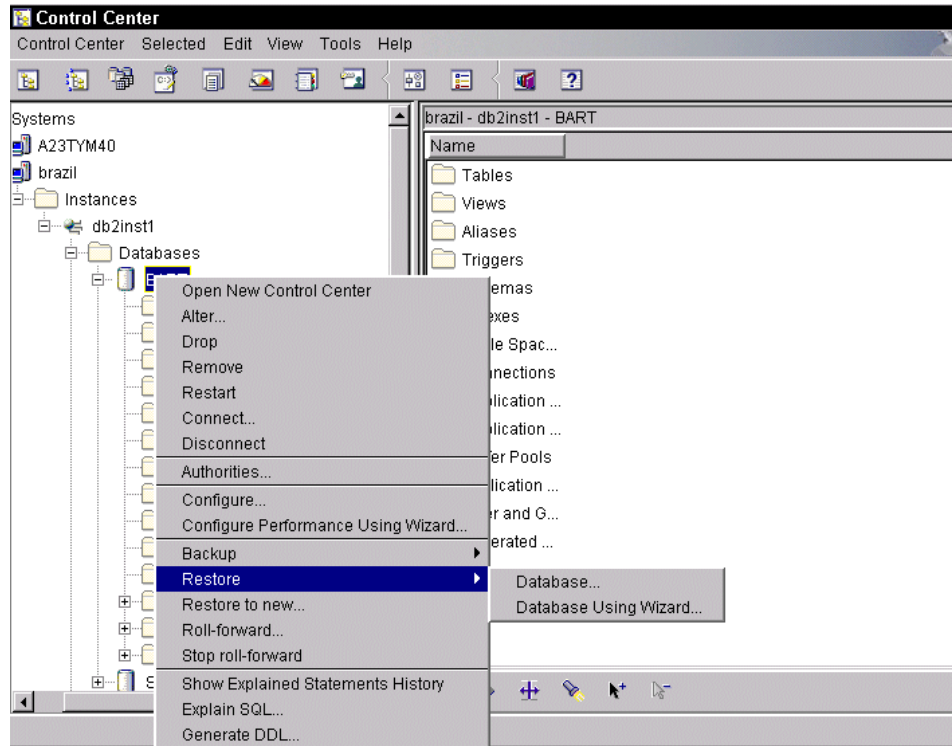


Figure 141. Database drop down menu

2. When the backup image selection page appears as shown in Figure 142, select a backup image in the list. This can be a backup image or a tablespace image.

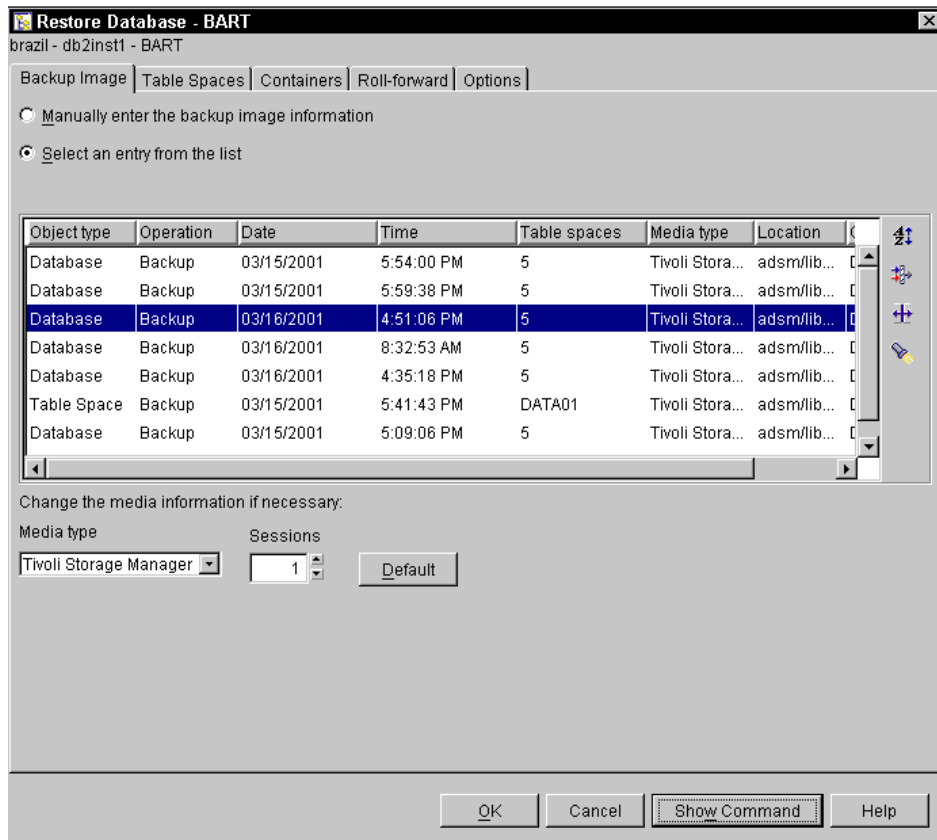


Figure 142. Backup image selection page

3. If you select a database backup image, on the Table Spaces tab, you can select which tablespaces you can restore. For this example, we use a backup image, and select tablespace DATA01, as shown in Figure 143.

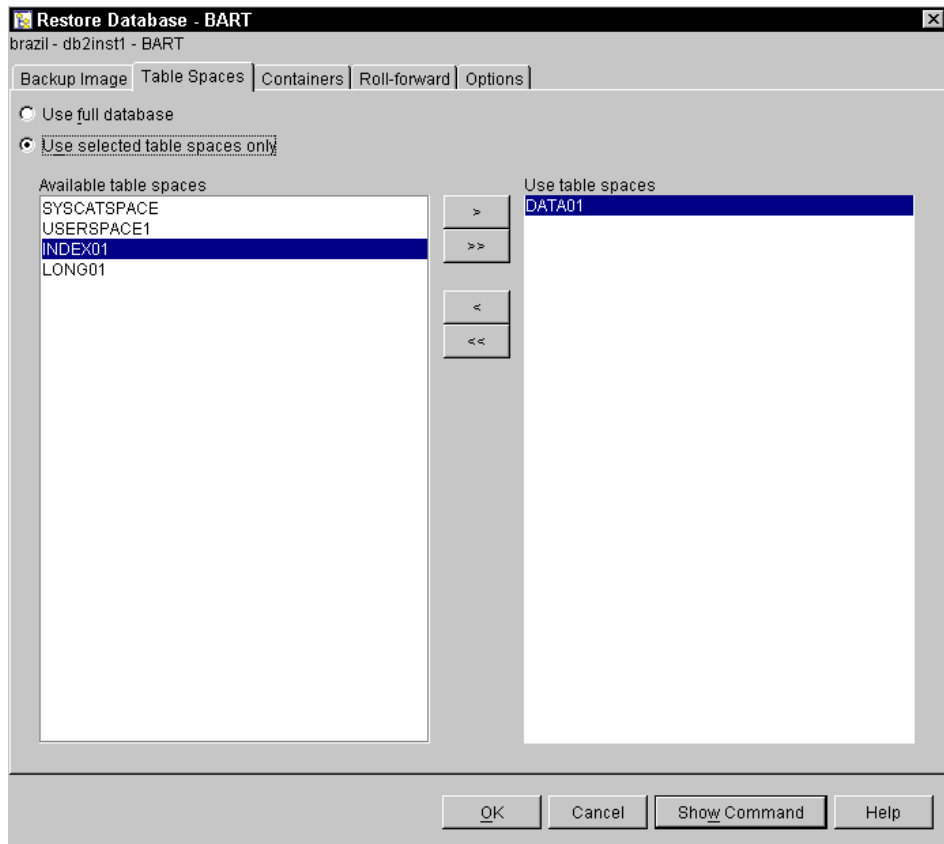


Figure 143. Table Spaces page

4. On the Roll-forward tab as shown in Figure 144, check the *Roll-forward (re-apply logs)* option, and uncheck the *Leave in roll-forward pending state* option.

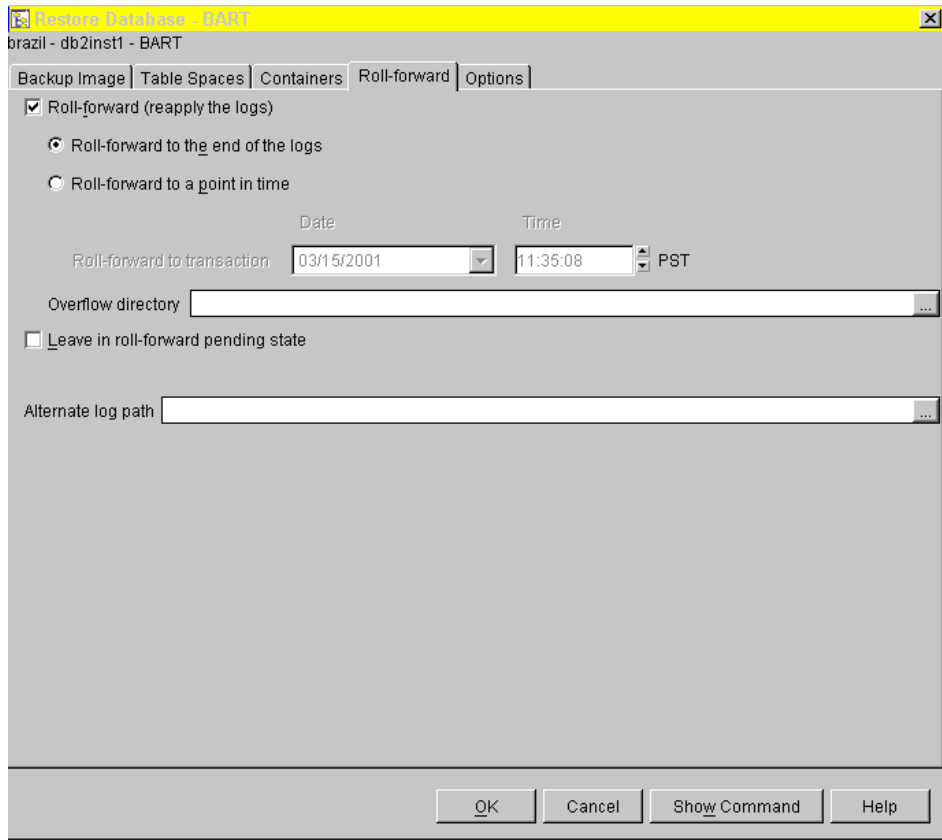


Figure 144. Roll forward page

5. Go to the Options tab as shown in Figure 145, and select Online for an online restore. Select **OK**.

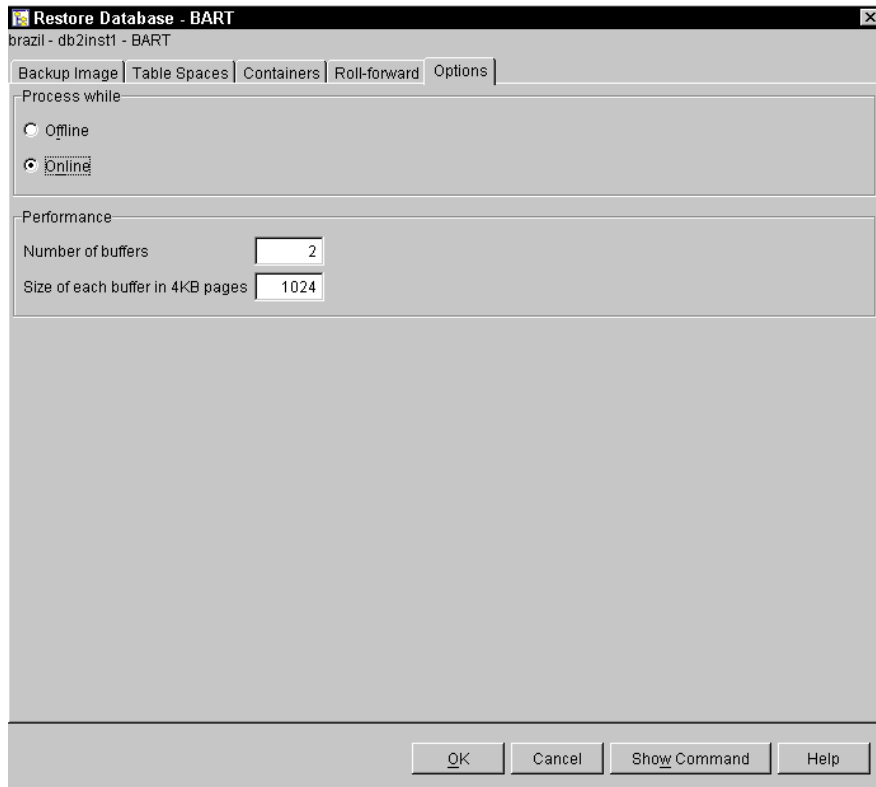


Figure 145. Options page

6. A dialog box appears as shown in Figure 146 to confirm if the roll-forward is successful. Select **Close**.

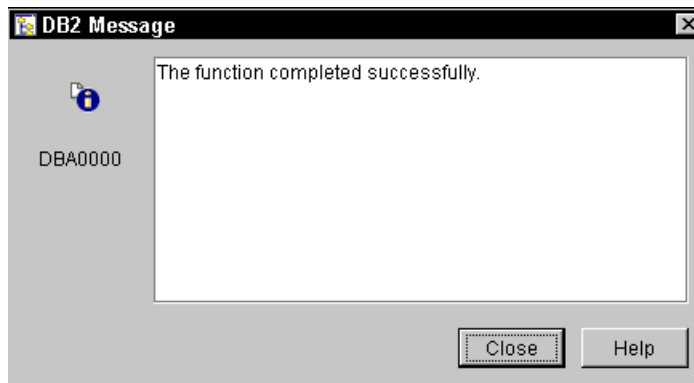


Figure 146. Tablespace roll-forward recovery successful

10.4 Point in time recovery

Before going through the procedures on doing point in time recovery, we review some of concepts and considerations that are needed to understand point in time recovery.

We also discuss our experiences and the problems we encountered when performing point in time recovery.

10.4.1 Point in time recovery concepts and lab experience

Rollback removes uncommitted transactions in the database. However, when the transaction is committed, rollback is no longer possible. To remove unwanted data that is mostly due to user errors, you have to do point in time recovery.

Point in time recovery can also be used if the archive logs are unusable, and roll-forwarding produces errors when reading the archive logs.

When using a user exit and Tivoli Storage Manager, all the archive logs required are automatically retrieved by Tivoli Storage Manager. So, the `to isotime` option must be used in the `rollforward` command for the user exit to retrieve only the logs that are necessary for point in time recovery.

Figure 147 shows an illustration of point in time recovery where the last log used was S0000013.

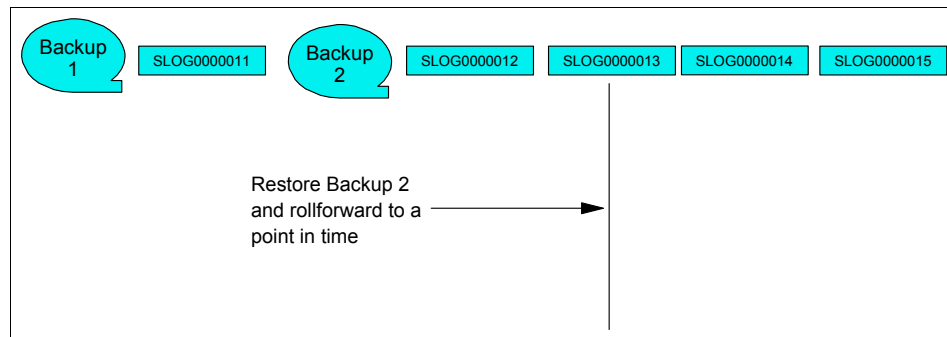


Figure 147. Point in time recovery

After the point in time recovery, new transactions will reuse the logs that were not applied as shown in Figure 148.

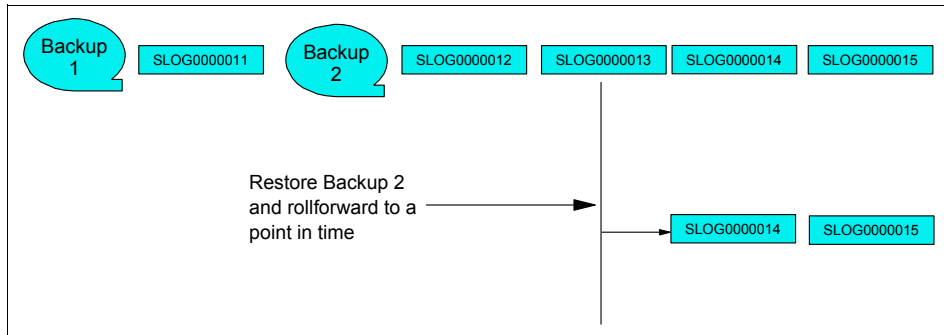


Figure 148. Log files being reused after a point in time recovery

When Tivoli Storage Manager archive the reused logs, there will be at least two versions of the same log file. During roll-forward, Tivoli Storage Manager will always retrieve the latest version of the archive logs. From our experience in the lab, when we try to recover BACKUP 1 or BACKUP 2 again and rollforward to end of logs, the roll-forward produces an error somewhere between SLOG0000013 and the reused SLOG0000014. So we have to roll-forward to the same or prior point in time as we have specified in the previous point in time recovery. We then took an online backup after each point in time recovery. However, when performing recovery using the new backups taken after the roll-forward, there was one occasion where the roll-forward failed. We therefore recommend a full offline backup after each database point in time recovery.

10.4.2 Point in time recovery considerations

Here are the requirements when doing point in time recovery.

10.4.2.1 Database point in time recovery considerations

During an online backup, logs are still being updated from new transaction. When recovering using online backups, there is a minimum recovery time that must be specified. This is usually the time when the online backup completed.

Tip

As pointed out in our experience in 10.4.1, "Point in time recovery concepts and lab experience" on page 235, we recommend that you do a full offline backup after a database point in time recovery.

10.4.2.2 Tablespace point in time recovery considerations

For tablespace point in time recovery, here are some considerations:

- You cannot do point in time recovery on the tablespace containing the system catalogs.
- If a table spans more than one tablespace, for example, data and indexes are in separate tablespaces, you must roll-forward those tablespaces to the same point in time.
- If tables in a tablespace being rolled forward to a point in time have referential integrity constraints in tables in other tablespaces. The tables having relationships will be in check-pending state. You must resolve the inconsistencies caused by the point in time recovery by using the `SET CONSTRAINTS` command.
- If tables in tablespaces being rolled forward to a point in time execute triggers that insert, update, or delete rows from tables in other tablespaces, rows from tables from tablespaces not being rolled forward may become inconsistent with the tables in tablespaces being rolled forward. You must correct the inconsistencies after the point in time recovery, or you can include the related tablespaces in the point in time recovery.
- Because all data must correlate with data in the system catalogs, there may be a minimum point in time to recover for a tablespace. For example, when a table is created, the system catalogs are updated to reflect this. You cannot do point in time recovery prior to the table creation, because the system catalogs have information for this table and is not removed by the tablespace point in time recovery process. This also applies to dropping tables. You cannot recover a dropped table unless you specify the `DROP TABLE RECOVERY` option turned on for the tablespace where the table resides.

To determine the minimum point in time for a tablespace to be recovered, use the `LIST TABLESPACES SHOW DETAIL` command.

- After the tablespace is rolled-forward to a point in time, it will be in backup-pending state. You must backup the tablespace.

10.4.3 Point in time recovery example

In this is example, a user committed wrong data to the database and now wants to remove unwanted data. The user would have to recover to a point in time prior to the user inserting wrong data. The tables are in tablespaces DATA01, INDEX01, and LONG01.

10.4.4 Point in time recovery using the command line

The following steps show how to recover DATA01, INDEX01, and LONG01 tablespaces:

1. Determine the minimum point in time for the tablespace by using the `LIST TABLESPACES SHOW DETAIL` command.

```
$ db2 list tablespaces show detail
```

Figure 149 shows an excerpt of the results. Note that the timestamp is in Coordinated Universal Time (CUT) format. In this example, CUT is six hours ahead of the local time.

```
Tablespace ID          = 3
Name                   = DATA01
Type                   = Database managed space
Contents               = Any data
State                  = 0x0000
  Detailed explanation:
    Normal
Total pages            = 5120
Useable pages         = 5088
Used pages             = 2912
Free pages             = 2176
High water mark (pages) = 2976
Page size (bytes)     = 4096
Extent size (pages)   = 32
Prefetch size (pages) = 32
Number of containers   = 1
Minimum recovery time = 2001-03-23-14.55.03.000000
```

Figure 149. List tablespaces show detail result

2. Use the `list history` command to select a backup image that you want to use for the restore. For database BART, our command will be:

```
$ db2 list history backup all for bart
```

An excerpt from the results of the command is shown in Figure 150. There can be several entries, so you need to scroll to select the backup image you need. You need the timestamp portion of the Timestamp+Sequence field or you can use the value in the Start Time field.

Op	Obj	Timestamp+Sequence	Type	Dev	Earliest Log	Current Log	Backup ID
B	P	20010323084231001	N	A	S0000027.LOG	S0000028.LOG	
Contains 3 tablespace(s):							
00001 DATA01							
00002 INDEX01							
00003 LONG01							
Comment: DB2 BACKUP BART ONLINE							
Start Time: 20010323084231							
End Time: 20010323084257							
00039 Location: adsm/libadsm.a							

Figure 150. List history command result

- Use the `restore` command, specifying the timestamp of the image you want to restore:

```
$ db2 "restore db bart tablespace (data01, index01, long01) online use tsm \
> taken at 20010323084231"
DB20000I The RESTORE DATABASE command completed successfully.
```

- Use the `rollforward` command to apply the logs. Specify the point in time in CUT format, and it must be past the minimum recovery time as shown in Figure 149 on page 238.

```
$ db2 "rollforward db bart to 2001-03-23-15.00.00.000000 \
> tablespace (data01, index01, long01) online"

Rollforward Status

Input database alias           = bart
Number of nodes have returned status = 1

Node number                    = 0
Rollforward status            = TBS working
Next log file to be read      = S0000029.LOG
Log files processed           = -
Last committed transaction    = 2001-03-23-15.11.15.000000

DB20000I The ROLLFORWARD command completed successfully.
$
```

- Notice we did not do a rollforward stop on the previous command. We can do them separately as:

```

$ db2 "rollforward db bart stop tablespace (data01, index01, long01) online"
SQL1278W Roll-forward operation has completed successfully. Active or
indoubt transactions required rollback on node(s) "0".
$

```

The tablespaces will now be in backup state pending as shown in Figure 151, using the `LIST TABLESPACES SHOW DETAIL` command.

```

Tablespace ID           = 3
Name                    = DATA01
Type                    = Database managed space
Contents                 = Any data
State                   = 0x0020
  Detailed explanation:
    Backup pending
Total pages              = 5120
Useable pages           = 5088
Used pages              = 2912
Free pages              = 2176
High water mark (pages) = 2912
Page size (bytes)       = 4096
Extent size (pages)     = 32
Prefetch size (pages)  = 32
Number of containers    = 1
Minimum recovery time   = 2001-03-23-14.55.03.000000

```

Figure 151. Tablespace in backup-pending state

6. Finally, we do a backup of the tablespaces.

```

$ db2 "backup db bart tablespace (data01, index01, long01) online use tsm"
Backup successful. The timestamp for this backup image is : 20010323100030

```

10.4.5 Point in time recovery using the Control Center

The following steps shows how to recover DATA01, INDEX01, and LONG01 tablespaces using the Control Center:

1. Right-click the database, and then from the drop down menu select **Restore->Database** as shown in Figure 152.

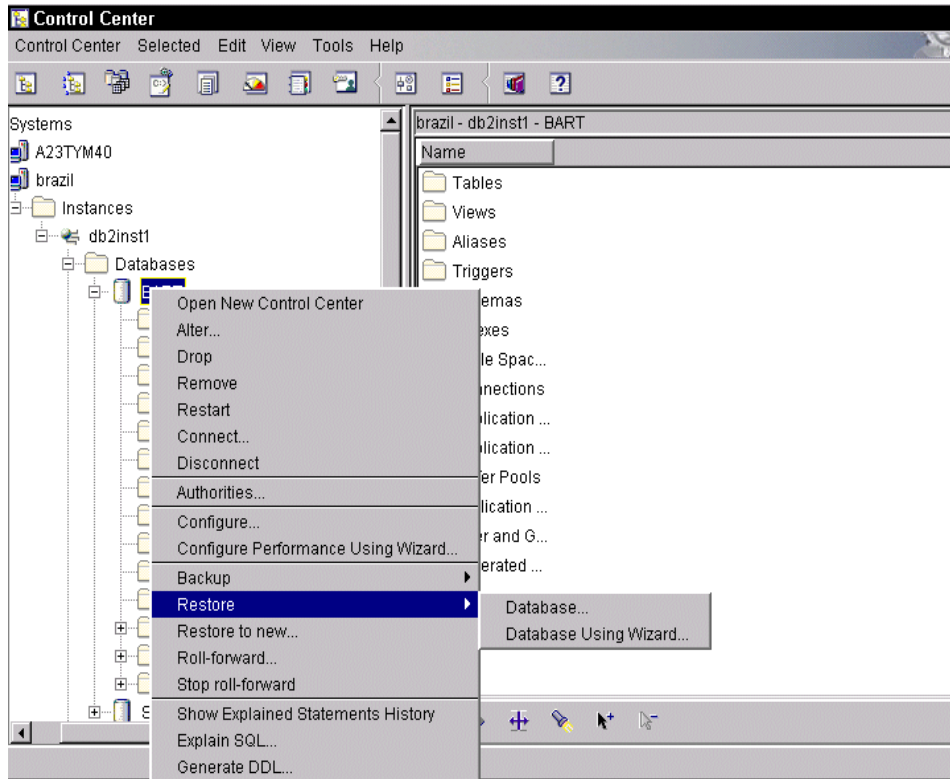


Figure 152. Tablespace point in time recovery

2. When the backup image selection page appears as shown in Figure 153, select a backup image in the list. This can be a backup image or a tablespace image.

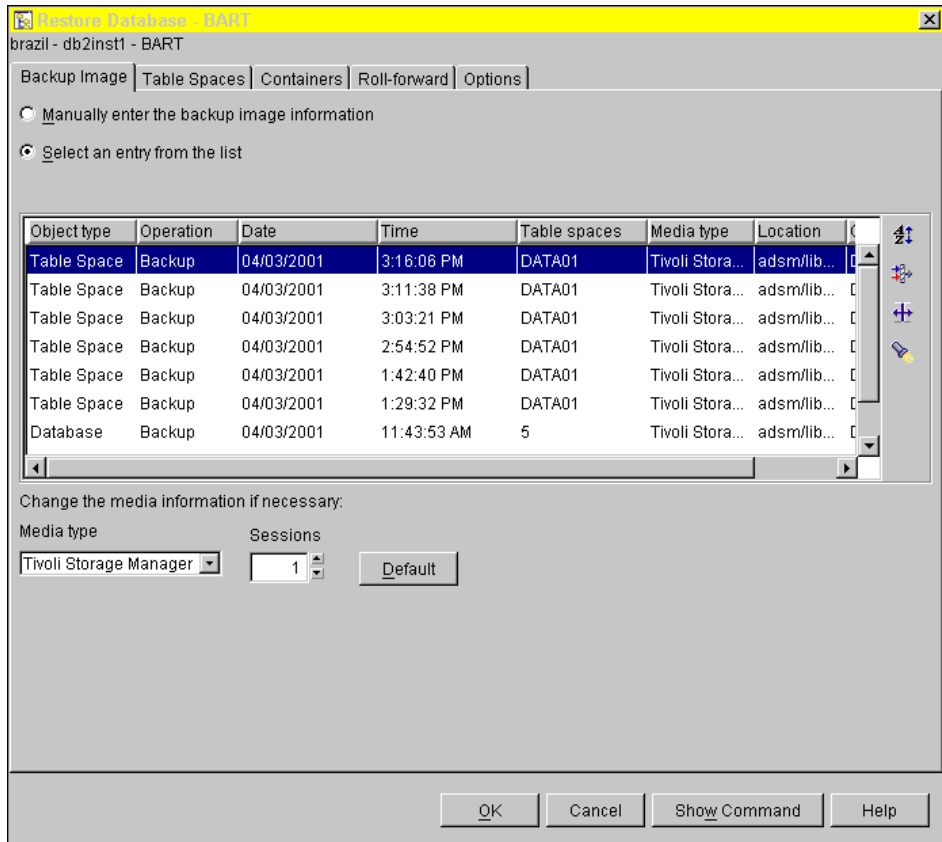


Figure 153. Backup image selection page

3. If you select a database backup image, on the Table Spaces tab, you can select which tablespaces you can restore. For this example, we used a backup image taken for tablespace DATA01, INDEX01, and LONG01 as shown in Figure 154.

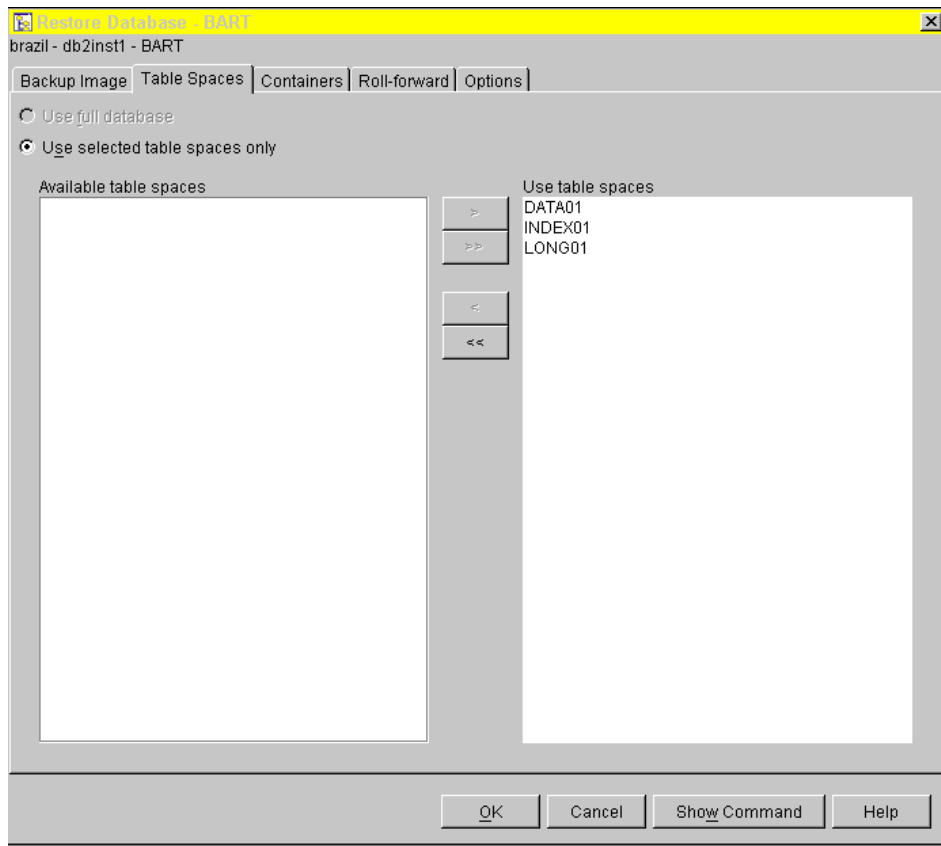


Figure 154. Table Spaces page

4. On the Roll-forward tab as shown in Figure 155, check the *Roll-forward (re-apply logs)* option, select *Roll-forward to a point in time*, enter the date and time in the Roll-forward to transaction fields, and uncheck the *Leave in roll-forward pending state* option.

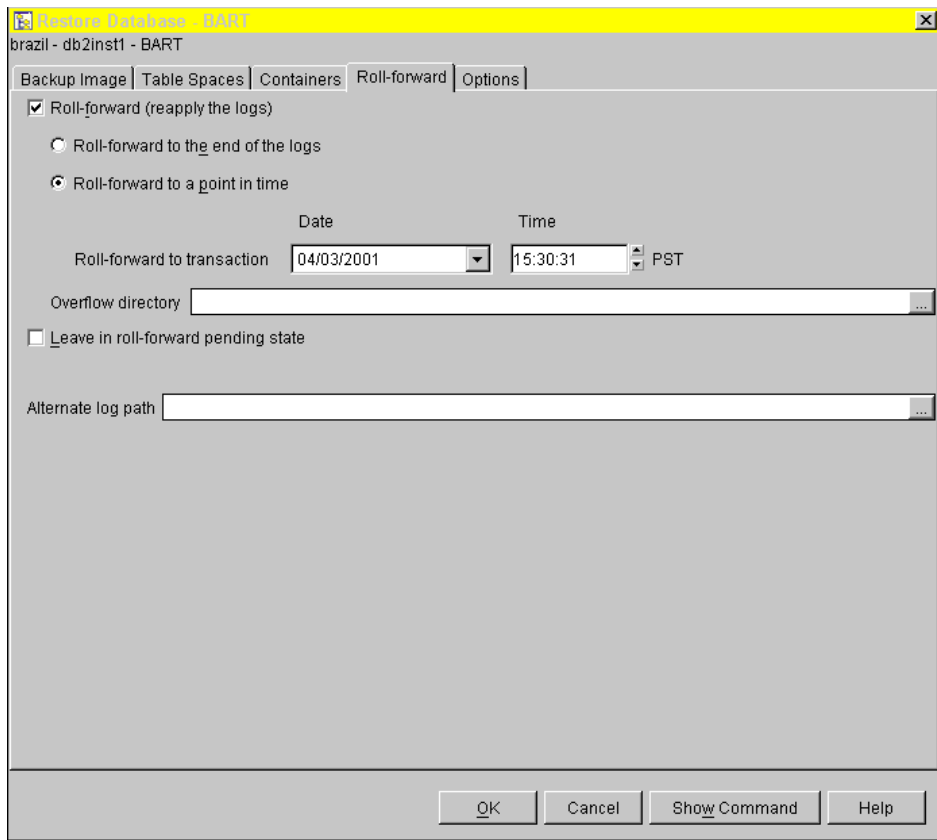


Figure 155. Roll-forward page

5. Go to the Options tab as shown in Figure 156, and select **Online** for an online restore. Select **OK**.

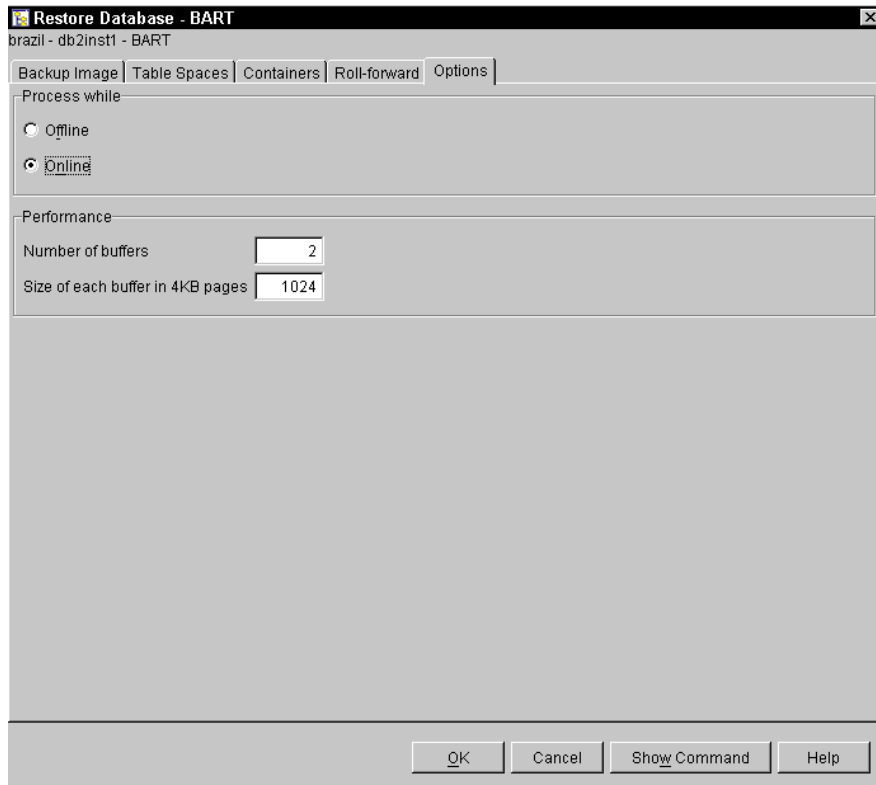


Figure 156. Options page

6. A dialog box appears as shown in Figure 157 to confirm if the roll-forward is successful. Select **Close**.

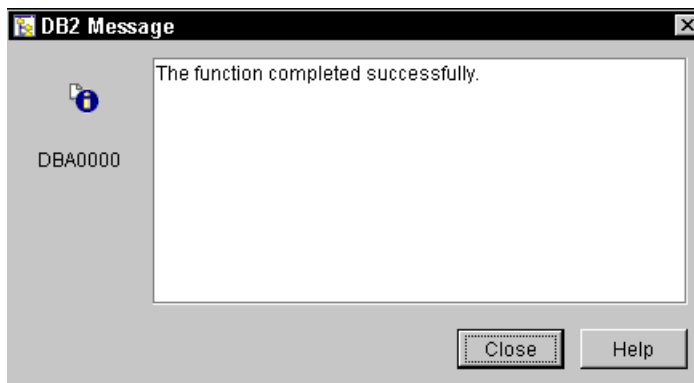


Figure 157. Point in time recovery successful

7. Finally, do a backup of the database or the tablespaces in backup pending mode. See 10.4.4, “Point in time recovery using the command line” on page 238 for backing up tablespaces with one command.

10.5 Restoring a DB2 database to a new DB2 instance

In this section, we will discuss how to restore a DB2 database using Tivoli Storage Manager that was backed up on another instance by Tivoli Storage Manager. The target instance can be on the same system or on another system.

10.5.1 Restore using db2adutl

Our first approach is to download the file on the source machine and to restore this image on the target machine. The advantage is that no Tivoli Storage Manager client configuration must be done at the target instance.

With the `db2adutl extract` command, the image must be downloaded on the source machine. Afterwards, the file needs to be transferred to the target machine and then restored from this image.

This approach can be established very easily but it has some disadvantages. It requires disk resources for the database image and it will also stress the communication network.

10.5.2 Restore using same Tivoli Storage Manager client setup

This approach can be used when restoring a DB2 backup to another instance that resides on a different physical machine (with the same operating system).

The target instance must have following requirements so that the backup image can be restored on that target machine.

- Same Tivoli Storage Manager client setup

The Tivoli Storage Manager client setup must be done in the same way as on the original machine. This includes the use of the same Tivoli Storage Manager client node and password. Also the generation of the password and if used also the user exit must be done the same way as on the source machine.

- Same instance name

These two requirements mean that the same setup which was done at the source machine must be repeated on the target machine. See the appropriate

sections for how to setup the Tivoli Storage Manager client API on AIX, SUN or Windows:

- 5.3.1, “API client setup” on page 64
- 6.2, “Configuration and setup of Tivoli Storage Manager client API” on page 94
- 8.4, “Configuring the API” on page 156

After the restore is completed, we recommend that you do *not* continue to use the temporary Tivoli Storage Manager client setup on the target machine. Otherwise, database backup objects from the original machine and the target would be mixed in the Tivoli Storage Manager server filespace.

10.5.3 Restore using DB2 parameter

This approach can be used if the instance name or the Tivoli Storage Manager node name or both on the target machine differ from that on the source machine. For example, this approach is necessary when restoring to a database in an existing instance that already has a Tivoli Storage Manager client API setup.

The Tivoli Storage Manager server stores not only the backup image but also some attributes that belong to this image. Two of those attributes are the owner of the backup and the Tivoli Storage Manager client node name that sent the backup. Another operating system user or another Tivoli Storage Manager client is not able to read or download this database image.

DB2 provides database configuration parameters to overcome this problem.

Note

These database configuration parameters are designed to temporarily overwrite the Tivoli Storage Manager client API setup for the purpose of restore. They can but should not be used for normal backup operations.

The following steps are required. We assume that the Tivoli Storage Manager API client is already setup.

1. Set password access to prompt
2. Create new target DB2 database
3. Get Tivoli Storage Manager information about old backup
4. Update the DB2 database configuration
5. Restore the database
6. Enable Tivoli Storage Manager client for normal backup operation

10.5.3.1 Set password access to prompt

To use the DB2 database configuration and not the generated password, the `dsm.sys PASSWORDACCESS` parameter must be set to `prompt`.

10.5.3.2 Create new target DB2 database

To be able to set the database parameter described later, the target database must be created first. (There is no way to set database parameters if there is no database!) Any operating system containers and required disk space for the database must also be available before the restore.

If the logfiles are also to be restored, then the new target database name must be the same as that of the original database. Otherwise, the newly created database name can be different.

```
$ db2 create db sample1
DB20000I The CREATE DATABASE command completed successfully.
$
```

10.5.3.3 Get Tivoli Storage Manager information about old backup

Following information about the database backup file that reside on the Tivoli Storage Manager server will be needed.

- Management class (optional)
- Nodename (required)
- Password of nodename (required)
- Owner of the file (optional)

One way to get this information is to ask the Tivoli Storage Manager administrator. Another way is to get the information directly out of the Tivoli Storage Manager server.

To login to the server open an administrative client session. On the command line enter following command:

```
dsmadm
```

You will be prompted for the node name and the password. The node name and the password are the ones used in 10.5.3.1, “Set password access to prompt” on page 248.


```

$ id
uid=203(db2inst1) gid=102(db2iadml) groups=1(staff),101(db2asgrp)
$ dsmdmnc
Tivoli Storage Manager
Command Line Administrative Interface - Version 4, Release 1, Level 1.0
(C) Copyright IBM Corporation, 1990, 1999, All Rights Reserved.

Enter your user id: brazil_db2

Enter your password:

Session established with server BRAZIL: AIX-RS/6000
Server Version 4, Release 1, Level 2.0
Server date/time: 03/07/01 16:38:07 Last access: 03/07/01 16:25:13

tsm: BRAZIL> .

```

Type `help`, to get help on what possible commands you can use. Every backup or logfile is indexed in the Tivoli Storage Manager server database. To get a list of the backups you already made, enter following command.

```
select * from backups where node_name='<source client API node name>'
```

Note: The `<source client API node name>` must be in uppercase.

The following screen shows the results of this command:

```

tsm: BRAZIL>select * from backups where node_name='BRAZIL_DB2'
ANR2963W This SQL query may produce a very large result table, or may require a
significant amount of time to compute.

Do you wish to proceed? (Yes/No)y

      NODE_NAME: BRAZIL_DB2
    FILESPACE_NAME: /SAMPLE
          STATE: ACTIVE_VERSION
           TYPE: FILE
        HL_NAME: /NODE0000/
        LL_NAME: FULL_BACKUP.20010301184338.1
      OBJECT_ID: 19470
    BACKUP_DATE: 2001-03-01 18:43:39.000000
DEACTIVATE_DATE:
          OWNER: db2inst1
    CLASS_NAME: DEFAULT

      NODE_NAME: BRAZIL_DB2
    FILESPACE_NAME: /SAMPLE
          STATE: ACTIVE_VERSION

```

The required information for our purposes is the values related to the following attributes: CLASS_NAME, NODE_NAME, OWNER. The password of the Tivoli Storage Manager client node is not readable as stored in the Tivoli Storage Manager server. If the password had been forgotten the Tivoli Storage Manager administrator can assign a new one.

10.5.3.4 Update the DB2 database configuration

The following DB2 database parameters will, if set, overwrite the Tivoli Storage Manager client setup on the target machine.

- TSM_NODENAME

This is the Tivoli Storage Manager client node name to which the DB2 backup originally was sent. This parameter is required. Note, the `PASSWORDACCESS` parameter in the `dsm.sys` file must be set to `prompt`; otherwise, the Tivoli Storage Manager API client will not be able to connect to the Tivoli Storage Manager server.

- TSM_PASSWORD

This is the password of the Tivoli Storage Manager client where the DB2 backup originally was sent. This parameter is required. Note, the `PASSWORDACCESS` parameter in the `dsm.sys` file must be set to `prompt`; otherwise, the Tivoli Storage Manager API client will not be able to connect to the Tivoli Storage Manager server.

- TSM_MGMTCLASS

This defines the management class where the backup image should be sent. This value can be left blank for the restore of a database. The restore will find the file (that belongs to a specific Tivoli Storage Manager node) no matter in which management class the file actually resides.

- TSM_OWNER

Each database file was backed up by a specific user. This value can be left blank for the restore. (It seems that DB2 reads the database backup file as root user from the Tivoli Storage Manager server and is therefore able to restore the database file that was originally backed up by another user.)

In our example, we created the database `sample1` and set only the required database configuration parameters (only Tivoli Storage Manager parameters are shown).

```

$ db2 update db cfg for sample using tsm_nodename brazil_db2
DB20000I The UPDATE DATABASE CONFIGURATION command completed successfully.
DB21026I For most configuration parameters, all applications must disconnect
from this database before the changes become effective.

$ db2 update db cfg for sample using tsm_password brazil_db2
DB20000I The UPDATE DATABASE CONFIGURATION command completed successfully.
DB21026I For most configuration parameters, all applications must disconnect
from this database before the changes become effective.

$ db2 get db cfg for sample1
TSM management class          (TSM_MGMTCLASS) =
TSM node name                 (TSM_NODENAME) = brazil_db2
TSM owner                     (TSM_OWNER) =
TSM password                  (TSM_PASSWORD) = *****

```

10.5.3.5 Restore the database

To restore a DB database backup that was originally backed up by another TSM client, you will need to get a listing of the DB2 backups from the other client using `db2adutl` with the nodename and password details from the original backup client. Again, the `passwordaccess` in the `dsm.sys` file must be set to `prompt`; otherwise, this command will fail with an authentication error.

```

$ db2adutl query full db sample nodename brazil_db2 password brazil_db2

Query for database SAMPLE

Retrieving full database backup information.
  full database backup image: 1, Time: 20010320112706
    Oldest log: S0000116.LOG, Node: 0, Sessions used: 1
  full database backup image: 2, Time: 20010320101516
    Oldest log: S0000091.LOG, Node: 0, Sessions used: 1
  full database backup image: 3, Time: 20010320093553
    Oldest log: S0000062.LOG, Node: 0, Sessions used: 1

```

Select the database to restore, and use a `db2 restore` command to restore the database. In our example, we restore a database image from database `sample` to database `sample1`. Afterwards we rollforward to the end of logs.

```

$ db2 restore db sample use tsm taken at 20010320112706 into sample1
SQL2528W Warning! Restoring to an existing database that is the same as the ba
ckup image database, but the alias name "SAMPLE1" of the existing database does
not match the alias "SAMPLE" of backup image, and the database name "SAMPLE1" of
the existing database does not match the database name "SAMPLE" of the backup i
mage. The target database will be overwritten by the backup version.
Do you want to continue ? (y/n) y
DB20000I The RESTORE DATABASE command completed successfully.
$ db2 rollforward db sample1 stop

                                Rollforward Status

Input database alias              = sample1
Number of nodes have returned status = 1

Node number                       = 0
Rollforward status                = not pending
Next log file to be read          =
Log files processed               = -
Last committed transaction        = 2001-03-20-16.43.40.000000

DB20000I The ROLLFORWARD command completed successfully.

```

To also roll-forward the logfiles, backed up by the user exit of the original Tivoli Storage Manager node, the user exit on the target machine needs to be configured. In addition, the `db2uext2.c` source code needs to be modified to connect to the original Tivoli Storage Manager node; otherwise, it will use the target Tivoli Storage Manager API client setup.

Also the destination database name should be named the same as the original database. DB2 will create the name of the logfiles to restore using the name of the target database. If this name differs from the name of the original database, the roll-forward process will not find the logfiles on the Tivoli Storage Manager server.

To make the necessary changes in the userexit look for the `dsmInit` routine in the `db2uext2.c` script. The third parameter specifies the Tivoli Storage Manager client nodename to connect to and the fifth parameter specifies the Tivoli Storage Manager client password. For further information about the `dsmInit` Routine, see *Tivoli Storage Manager Using the Application Program Interface V4R1*, SH26-4123.

In the example, both the original and the modified versions of the `dsmInit` call are shown. The variables `ORIG_NODENAME` and `ORIG_PASSWORD` are normal defined string variables that contain the original Tivoli Storage Manager nodename and password, for example, `#define ORIG_NODENAME "BRAZIL_DB2"`.

```

adsmRc = dsmInit( &adsmHandle, &adsmApiVer, NULL, NULL,
                 inputParms->adsmPasswd, FILE_SPACE_TYPE,
                 NULL, NULL ) ;

```

This line must be changed to something like the follow:

```

adsmRc = dsmInit( &adsmHandle, &adsmApiVer, ORIG_NODENAME , NULL,
                 ORIG_PASSWORD, FILE_SPACE_TYPE,
                 NULL, NULL ) ;

```

After the user exit is setup with the above modification, the `db2 rollforward` command is able to roll-forward the logfiles from the original database. If the `stop` option on the rollforward is never specified, the roll-forward can be restarted after new logs are available on the Tivoli Storage Manager server from the old database. (With this setup we implemented a kind of shadow database!)

```

$ db2 rollforward db sample to end of logs

                                Rollforward Status

Input database alias              = sample
Number of nodes have returned status = 1

Node number                       = 0
Rollforward status                = DB working
Next log file to be read          = S0000203.LOG
Log files processed                = S0000161.LOG - S0000202.LOG
Last committed transaction        = 2001-03-20-21.31.29.000000

DB20000I The ROLLFORWARD command completed successfully.
$
$ db2 rollforward db sample to end of logs

                                Rollforward Status

Input database alias              = sample
Number of nodes have returned status = 1

Node number                       = 0
Rollforward status                = DB working
Next log file to be read          = S0000206.LOG
Log files processed                = S0000161.LOG - S0000204.LOG
Last committed transaction        = 2001-03-20-21.38.45.000000

DB20000I The ROLLFORWARD command completed successfully.

```

This roll-forward can be repeated until the `stop` option of the `db2 rollforward` command is specified. This will enable the clients to access the database on the destination node.

10.5.3.6 Enable TSM client for normal backup operation

This step is required if both the original and the destination database want to back up their database and logfiles using Tivoli Storage Manager, after the restore to the destination machine is finished. If the DB2 database parameters stay the same the backups and the logfiles will probably mix up on the Tivoli Storage Manager server.

To enable the Tivoli Storage Manager client for normal backup operation, the DB2 configuration parameters and, if modified, the `db2uext2` program needs to be changed back as they were before the restore. For example, to set the `TSM_NODENAME` parameter back to `NULL` the following `db2` command needs to be executed:

```
$ db2 update db cfg for sample using tsm nodename NULL
DB20000I The UPDATE DATABASE CONFIGURATION command completed successfully.
DB21026I For most configuration parameters, all applications must disconnect
from this database before the changes become effective.
```

The changes made to `db2uext2.c` must be redone. The `db2uext2` compiled again and then copied to the `adm` directory.

After that the destination database should be ready for normal backup operations.

10.6 Tablespace redirected restore

Tablespace redirected restore is used when tablespace containers cannot be restored to their original location. This may be due to disk or hardware errors. Redirected restore can also be used if you want to reorganize the layout of the containers in the tablespace where you can add, change, remove containers. You can move containers to another device, for example, to distribute I/O. Since `ALTER TABLESPACE` does not allow you to add containers once an SMS tablespace is created, you can add containers by using redirected restore.

10.6.1 Tablespace redirected restore example

You have over-allocated the space for the `INDEX01` tablespace, and you want to resize the container to free up space, so it can be used by other objects in the database.

10.6.2 Tablespace redirected restore using the command line

In this example, the following steps performs a redirected restore for tablespace INDEX01:

1. Determine the Tablespace ID of INDEX01 by using the `list tablespaces` command, which shows that the Tablespace ID is 4. See Figure 158.

```
$ db2 list tablespaces

      Tablespaces for Current Database

Tablespace ID          = 0
Name                   = SYSCATSPACE
Type                   = System managed space
Contents               = Any data
State                  = 0x0000
  Detailed explanation:
    Normal

Tablespace ID          = 1
Name                   = TEMPSPACE1
Type                   = System managed space
Contents               = System Temporary data
State                  = 0x0000
  Detailed explanation:
    Normal

Tablespace ID          = 2
Name                   = USERSPACE1
Type                   = System managed space
Contents               = Any data
State                  = 0x0000
  Detailed explanation:
    Normal

Tablespace ID          = 3
Name                   = DATA01
Type                   = Database managed space
Contents               = Any data
State                  = 0x0000
  Detailed explanation:
    Normal

Tablespace ID          = 4
Name                   = INDEX01
Type                   = Database managed space
Contents               = Any data
State                  = 0x2002100
  Detailed explanation:
    Restore pending
    Restore in progress
    Storage may be defined

Tablespace ID          = 5
Name                   = LONG01
Type                   = Database managed space
```

Figure 158. List tablespace command

2. Using the Tablespace ID, list the containers for the tablespace using the `list tablespace containers` command:

```
$ db2 list tablespace containers for 4 show detail
```

Tablespace Containers for Tablespace 4

```
Container ID           = 0
Name                   = /db2dat1/bart/index01.dat
Type                   = File
Total pages           = 5120
Useable pages         = 5104
Accessible             = Yes
```

3. Use the `list history` command to select a backup image you want to use for the restore. For database BART, our command will be:

```
$ db2 list history backup all for bart
```

An excerpt from the results of the command is shown in Figure 159. There can be several entries, so you need to scroll to select the backup image you need. You need the timestamp portion of the Timestamp+Sequence field or you can use the value in the Start Time field.

```
Op  Obj  Timestamp+Sequence  Type  Dev  Earliest Log  Current Log  Backup ID
---  ---  ---                ---  ---  ---          ---          ---
B   D   20010323150025001  F     A   S0000038.LOG S0000038.LOG
-----
Contains 5 tablespace(s):

00001 SYSCATSPACE
00002 USERSPACE1
00003 DATA01
00004 INDEX01
00005 LONG01
-----
Comment: DB2 BACKUP BART OFFLINE
Start Time: 20010323150025
End Time: 20010323150057
-----
00067 Location: adsm/libadsm.a
```

Figure 159. List history command result

4. Use the `restore` command with the `redirect` option, specifying the tablespace to be restored and the timestamp of the image you want to use for the restore:


```

$ db2 "restore db bart tablespace (INDEX01) online use tsm \
> taken at 20010323150025 redirect"
SQL1277N Restore has detected that one or more table space containers are
inaccessible, or has set their state to 'storage must be defined'.
DB20000I The RESTORE DATABASE command completed successfully.

```

5. Use the `set tablespace containers` to resize the container for tablespace ID 4. The ignore rollforward container operations means that the *ALTER TABLESPACE* operations in the logs will be ignored during the roll-forward.

```

$ db2 "set tablespace containers for 4 \
> ignore rollforward container operations \
> using (file '/db2dat1/bart/index01.dat' 2560)"
DB20000I The SET TABLESPACE CONTAINERS command completed successfully.

```

6. Continue the restore with the `continue` option:

```

$ db2 restore db bart continue
DB20000I The RESTORE DATABASE command completed successfully.

```

7. Do a roll-forward for the tablespace:

```

$ db2 "rollforward db bart to end of logs and stop \
> tablespace (index01) online"

                                Rollforward Status

Input database alias              = bart
Number of nodes have returned status = 1

Node number                       = 0
Rollforward status                 = not pending
Next log file to be read           = S0000039.LOG
Log files processed                 = -
Last committed transaction         = 2001-03-23-21.05.31.000000

DB20000I The ROLLFORWARD command completed successfully.

```

10.6.3 Tablespace redirected restore using the Control Center

Follow these steps to do a tablespace redirected restore using the Control Center.

1. Right-click the database, and then from the drop down menu select **Restore->Database** as shown in Figure 160.

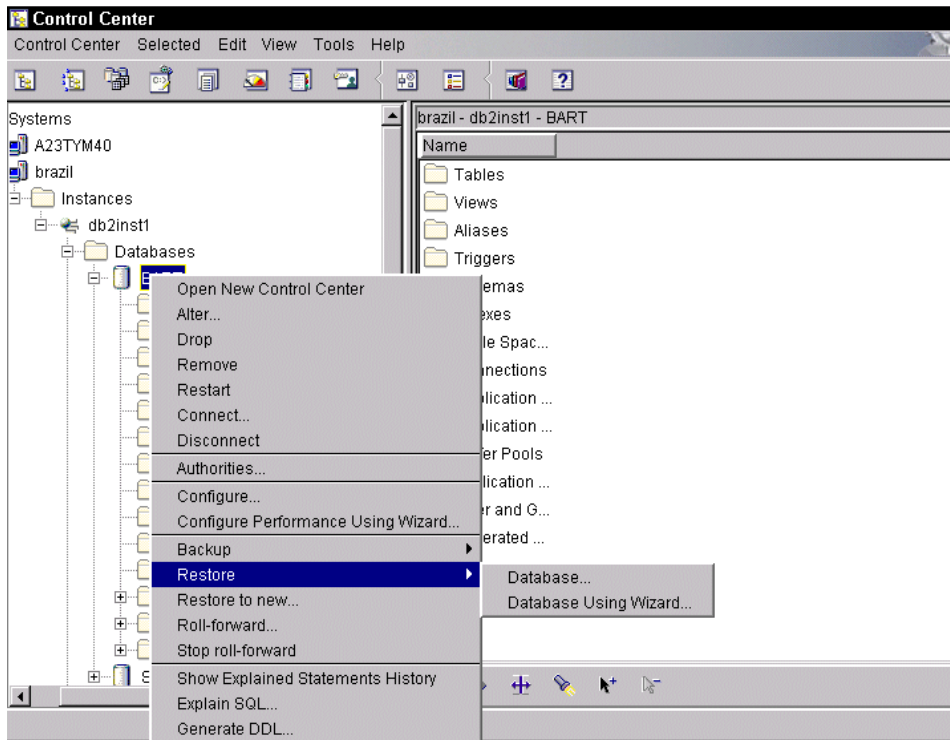


Figure 160. Tablespace redirected restore

2. When the backup image page appears as shown in Figure 161, select the database image you want to use for the restore, and then select the Table Spaces tab.

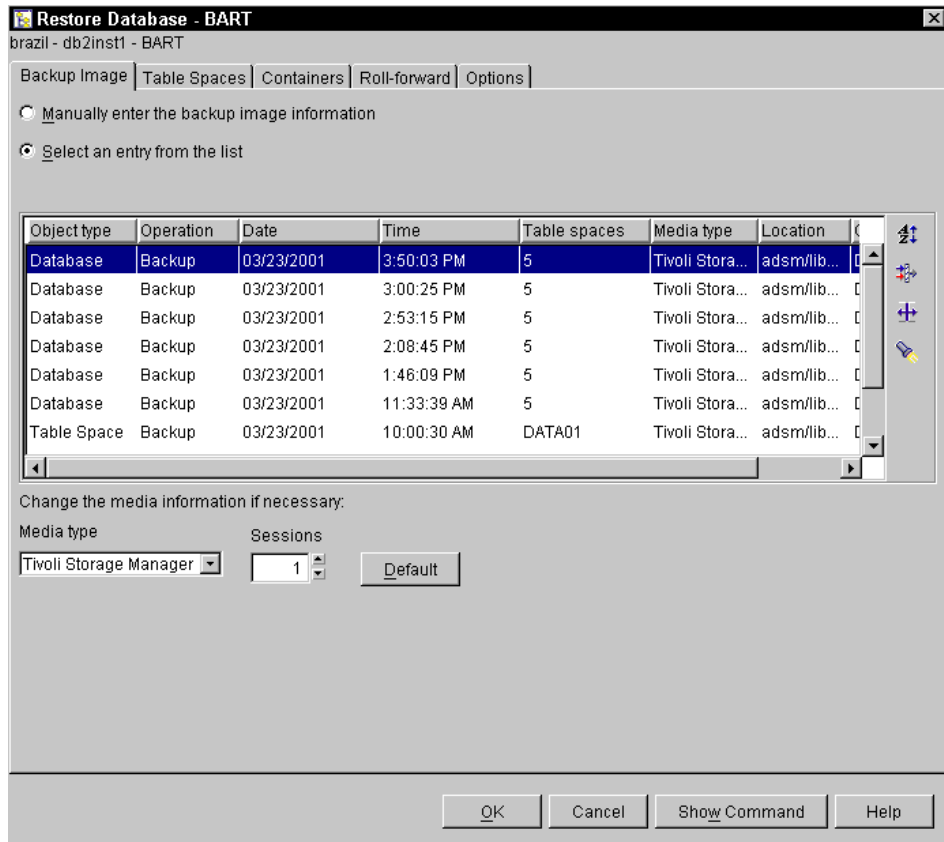


Figure 161. Backup image selection page

- On the Table Spaces page as shown in Figure 162, click the *Use selected tablespaces only* radio button, select INDEX01, and then select the Containers tab.

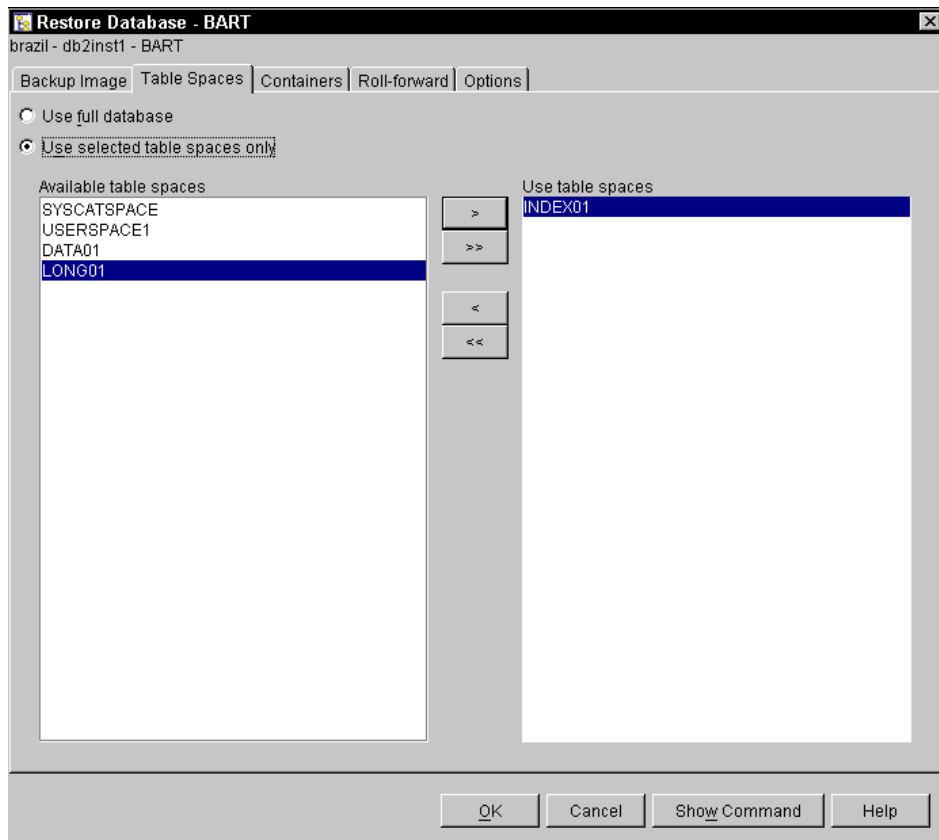


Figure 162. Table Spaces page

4. On the Containers page, when you check the *Redirect table space containers*, and it display the list of containers for the tablespace as shown in Figure 163, highlight the container on the lower frame, and then select **Change**.

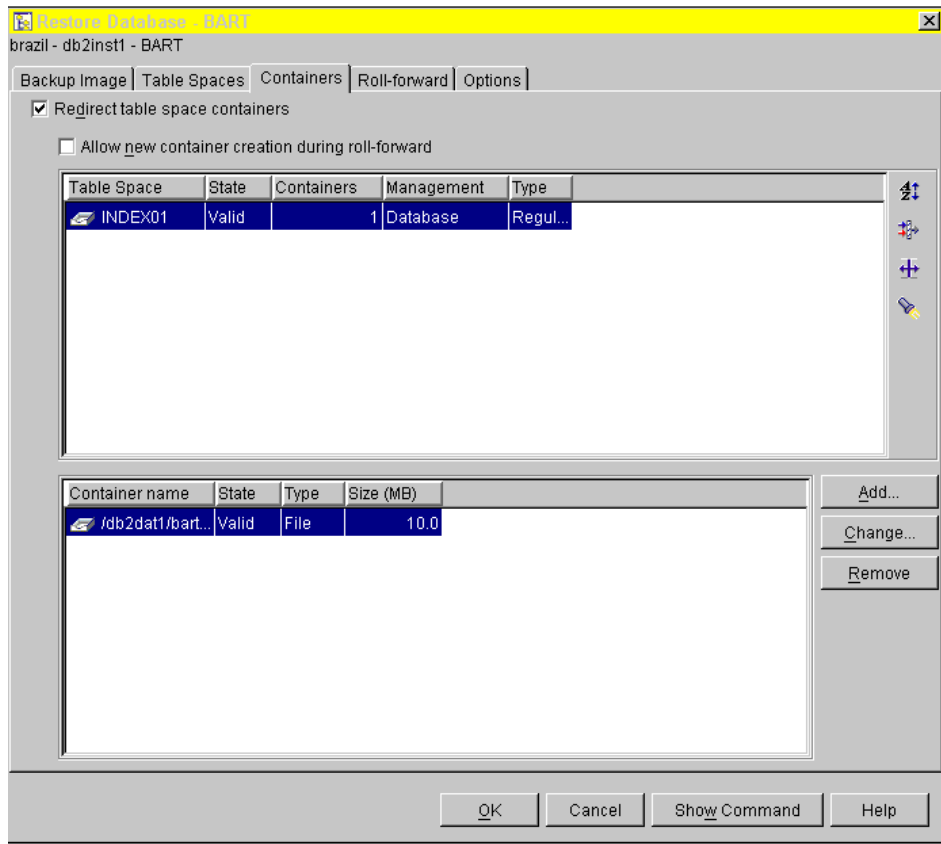


Figure 163. Container page

5. A Change Container dialog appears as shown in Figure 164. Change the file size, and then select **OK**.

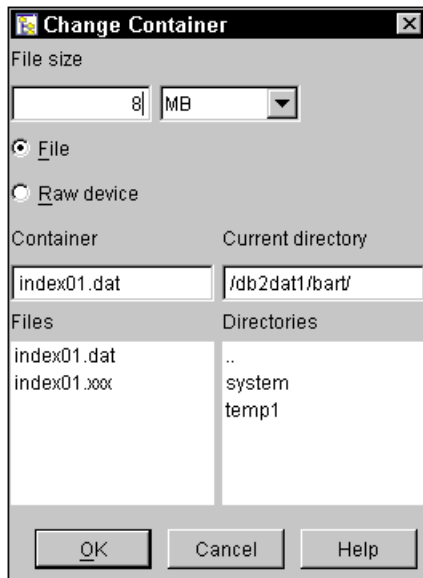


Figure 164. Change container dialog

6. You are now back to the Container page as shown previously in Figure 163. Select the Roll-forward tab. Check the *Roll-forward (re-apply logs)* option, uncheck the *Leave in roll-forward pending state* option as shown in Figure 165. Select the Options tab.

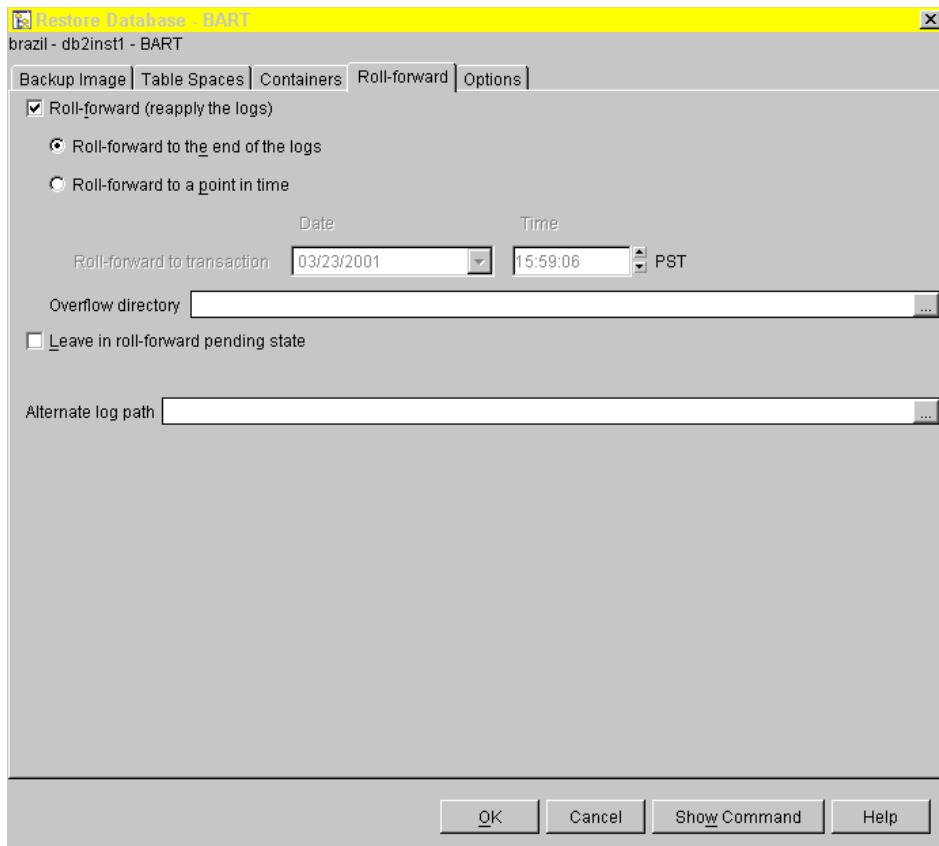


Figure 165. Roll-forward page

7. On the Options page as shown in Figure 166, select Online if you want to do recovery online. Select **OK** when done.

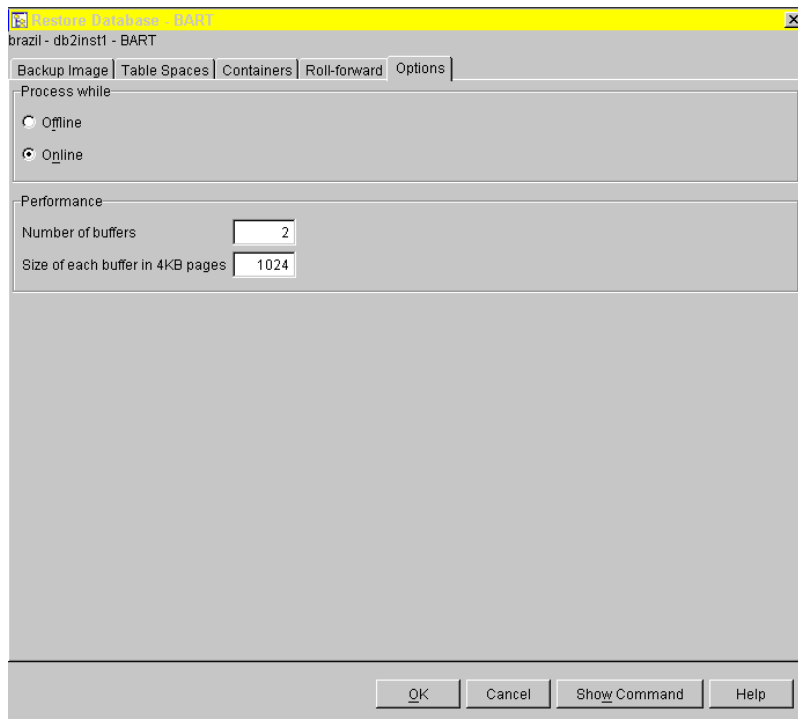


Figure 166. Options page

8. A dialog box appears as shown in Figure 167 to confirm if the tablespace redirected restore is successful. Select **Close**.

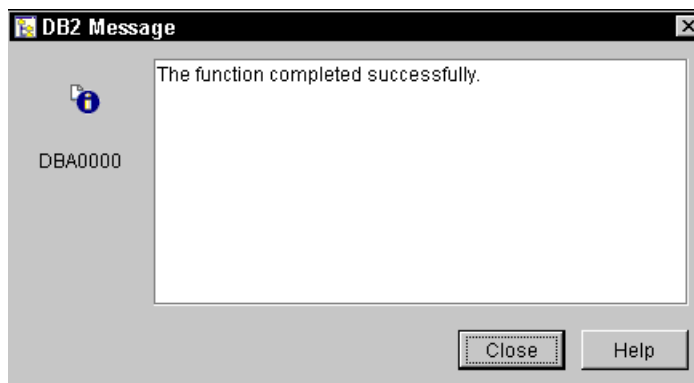


Figure 167. Tablespace redirected restore successful

10.7 Recovering the history file

The history file is used to get information about database backup image you can use to do recovery. The Control Center also uses this file on the Backup Image page as shown in Figure 161 when giving the user the option to select a backup image. The history file is always backed up during a database or tablespace backup.

You can use the `db2adutl` utility as an alternative to get the same information. The history file, however, also contains information about load operations and backups and restore operations that were done outside Tivoli Storage Manager. For this reason, you might want to recover the history file if the file `db2rhist.asc` has been corrupted or the entries were accidentally deleted using the `DB2 PRUNE` command.

To recover the history file, perform the following steps:

1. Use `db2adutl` to get the latest backup image:

```
$ db2adutl query db bart | more

Query for database BART

Retrieving full database backup information.
full database backup image: 1, Time: 20010323170923
  Oldest log: S0000042.LOG, Node: 0, Sessions used: 1
full database backup image: 2, Time: 20010323165055
  Oldest log: S0000042.LOG, Node: 0, Sessions used: 1
full database backup image: 3, Time: 20010323155003
  Oldest log: S0000040.LOG, Node: 0, Sessions used: 1
full database backup image: 4, Time: 20010323150025
  Oldest log: S0000038.LOG, Node: 0, Sessions used: 1
full database backup image: 5, Time: 20010323145315
  Oldest log: S0000037.LOG, Node: 0, Sessions used: 1
full database backup image: 6, Time: 20010323140845
  Oldest log: S0000036.LOG, Node: 0, Sessions used: 1
```

2. Recover the history file:

```
$ db2 restore db bart history file online use tsm taken at 20010323170923
DB20000I The RESTORE DATABASE command completed successfully.
```

10.8 Additional notes on recovery

Here are some additional notes that can be helpful when you are doing recovery.

10.8.1 Roll-forward status

When a roll-forward operation successfully completes, it gives you the status of the roll-forward. You can also issue the `rollforward query status` command at any time to give you the roll-forward status:

```
$ db2 rollforward db bart query status

                                Rollforward Status

Input database alias              = bart
Number of nodes have returned status = 1

Node number                       = 0
Rollforward status                 = not pending
Next log file to be read           =
Log files processed                 = -
Last committed transaction         = 2001-04-03-20.30.50.000000
```

Table 1 shows the possible values for the Rollforward status.

Table 1. Rollforward status

Rollforward status	Description
Not pending	The database or tablespace does not require the rollforward operation
DB pending	The database is in rollforward-pending state
DB working	The database is in rollforward-in-progress state
TBS pending	One or more tablespaces are in a rollforward-pending state
TBS working	One or more tablespaces are in a rollforward-in-progress state

10.8.2 Restarting restore and roll-forward

During a restore or roll-forward operation, you may decide to cancel the current restore or roll-forward operation and start again. You may want to use

a different backup image. You can use any backup image that you want, and these can be images with timestamps that were taken before or after the backup image that you used on the previous restore operation. For example, you may want to use an earlier backup image for a point in time recovery.

However, you must be careful in the roll-forward operation that you do not use archive logs that have been reused, as mentioned in 10.4.1, “Point in time recovery concepts and lab experience” on page 235; otherwise, with Tivoli Storage Manager your roll-forward will fail.

Before repeating the restore operation, cancel the roll-forward. For example:

```
db2 rollforward db bart cancel tablespace(data01,index01,long01)
```

The database or tablespaces that were previously restored will be placed in a restore-pending state. You can now use a backup image to restore a database, or you can use one or more backup images to restore tablespaces, so that all tablespaces that were previously in a restore-pending state will be in a rollforward-pending state.

10.8.3 Using more than one backup image for a recovery

You can use more than one backup image when restoring a database or restoring tablespaces. For example, your backup routine may be:

- Backup the database daily
- Backup the most volatile tablespaces (DATA01 and INDEX01) twice a day

You can do the following, for example, when the SYSCATSPACE and DATA01 containers gets damaged:

1. Restore the SYSCATSPACE tablespace from the latest database backup image:

```
$ db2 "restore db bart tablespace (syscatspace) use tsm \  
> taken at 20010404070515"  
DB20000I The RESTORE DATABASE command completed successfully.
```

2. Because you have a more recent tablespace backup for DATA01 than the last database backup image, you can restore DATA01 from the latest tablespace backup image:

```
$ db2 "restore db bart tablespace (data01) use tsm \  
> taken at 20010404171748"  
DB20000I The RESTORE DATABASE command completed successfully.
```

3. Roll-forward the tablespaces:

```
$ db2 "rollforward db bart to end of logs and stop  
> tablespace(syscatspace,data01)"  
  
Rollforward Status  
  
Input database alias           = bart  
Number of nodes have returned status = 1  
  
Node number                    = 0  
Rollforward status             = not pending  
Next log file to be read       = S0000045.LOG  
Log files processed            = -  
Last committed transaction     = 2001-04-05-18.45.49.000000
```

10.8.4 Restore without rolling forward

If the database is roll-forward enabled, a restore will always put the database or tablespaces in a rollforward-pending state. Depending on your situation in a point in time recovery, you may only want to do a restore and stop there without applying the logs. However, you must consider the minimum recovery time, as discussed in 10.4.2, “Point in time recovery considerations” on page 236.

You do this with the `without rolling forward` option of the `restore` command:

```
$ db2 restore db bart use tsm taken at 20010404170515 without rolling forward  
SQL2539W Warning! Restoring to an existing database that is the same as the backup image database.  
The database files will be deleted.  
Do you want to continue ? (y/n) y  
DB20000I The RESTORE DATABASE command completed successfully.
```

The database will not be in a rollforward-pending state most of the time if you use this option. You can query the roll-forward state, as discussed in 10.8.1, “Roll-forward status” on page 266, with the following:

```
$ db2 rollforward db bart query status
```

Rollforward Status

```
Input database alias           = bart
Number of nodes have returned status = 1

Node number                    = 0
Rollforward status             = not pending
Next log file to be read      =
Log files processed            = -
Last committed transaction     = 2001-04-03-16.41.19.000000
```

This does not work all the time. Sometimes, DB2 still puts it in rollforward-pending state. To remove the rollforward-pending state, stop the rollforward:

```
$ db2 rollforward db bart stop
```

Appendix A. Quick start/checklist for configuration

Here is a quick guide to backing up UDB DB2 databases using Tivoli Storage Manager.

A.1 Windows quick start

1. Install Tivoli Storage Manager Backup-Archive client (including the API).
2. Create a plain text file dsm.opt in x:\progra~1\Tivoli\TSM\api.

It must contain at least the following.

```
COMMMETHOD TCPIP
TCPSEVERADDRESS xxx.xxx.xxx.xxx
NODENAME DB2TSM
PASSWORDACCESS GENERATE
```

3. Register the node DB2TSM on the Tivoli Storage Manager server.

It is recommended to create a new POLICY DOMAIN on the Tivoli Storage Manager server for this node. If you do create a new domain, make sure the node is updated with the new domain.

The domain needs only one management class with a backup copygroup that has the following retention settings:

```
VEREXIST=1
VERDELETED=0
RETEXTRA=0
RETONLY=0
```

4. In Control Panel/System for the environment **SYSTEM** variables, specify the variable DSMI_CONFIG with the value:
x:\progra~1\Tivoli\TSM\api\dsm.opt
Note: DSMI_CONFIG is case sensitive. The SYSTEM not the USER variable needs to be set.
5. Reboot the system or restart the DB2 services, so that the DSMI_CONFIG variable will be loaded into the DB2 runtime engine. Otherwise, a backup to Tivoli Storage Manager will fail with error '406 DSM.OPT file not found.'
6. Sign-on as db2admin and change to the ...\sqllib\adsm directory.
7. Run the executable 'dsmapipw' and follow the prompts. It will ask for the current password, then for a new password, and confirmation of the new password. It should report if it is successful.

8. Run the db2 command 'db2adutl query' to confirm that the password file was correctly set. If the command comes back and says no db2 objects found, then it is working (you haven't done any backups yet). You should be able to check the activity log on the Tivoli Storage Manager server to confirm that the node DB2TSM authenticated. Db2adutl.exe is located in the x:\program files\sqllib\bin directory.
9. You can now perform a backup using the db2 command line or the db2 connect GUI. (For example, 'backup db DBNAME use tsm'.)

A.2 AIX quick start

Here is a quick start guide for backing up DB2 to Tivoli Storage Manager. This is based on the AIX OS and the 3.7 or 4.1 Tivoli Storage Manager API level.

DB2 provides native support for backing up to Tivoli Storage Manager. DB2 utilizes the Tivoli Storage Manager API to perform these backups. The following is the minimum steps necessary to get DB2 to backup to Tivoli Storage Manager:

1. Install DB2.
2. Install Tivoli Storage Manager API.
3. Create a plain text file dsm.opt in /usr/tivoli/tsm/client/api/bin.

It only needs one line, the value for SERVERNAME is arbitrary, but must match the dsm.sys file:

```
SERVERNAME DB2
```

4. Create a plain text file dsm.sys in /usr/tivoli/tsm/client/api/bin.

It must contain at least the following:

```
SERVERNAME DB2  
COMMETHOD TCPIP  
TCPSEVERADDRESS xxx.xxx.xxx.xxx  
NODENAME DB2TSM  
PASSWORDACCESS GENERATE
```

5. Register the node DB2TSM on the Tivoli Storage Manager server.

We recommend for you to create a new DOMAIN on the Tivoli Storage Manager server for this node. It needs only one management class with a copygroup that has the following retention settings:

```
VEREXIST=1  
VERDELETED=0  
RETEXTRA=0
```


REONLY=0

6. Sign-on as root and cd to the db2 directory that contains the files db2adutl and dsmapipw.
7. Run the executable 'dsmapipw' and follow the prompts.

It will ask for the current Tivoli Storage Manager node password, then for a new password, and confirmation of the new password. It should report if it is successful.

8. Run the command 'db2adutl query' to confirm that the password file was correctly set.

If the command comes back and says no db2 objects found, then it is working (you haven't done any backups yet). You should be able to check the activity log on the Tivoli Storage Manager server to confirm that the node DB2TSM authenticated.

9. You can now perform a backup using the command line or the db2 connect GUI. For example, db2 backup db DBNAME use tsm.

Additional DB2 information:

- In the home/db2instance/sqllib directory run '. db2profile' to update the signed on user with the needed environment variables (DB2INSTANCE=db2inst1, for example).
- Start db2 command line processor via /home/db2instance/sqllib/bin/db2.
- Get db cfg for DBNAME.
- Update db cfg for DBNAME using TSM_PASSWORD NULL. (Similar syntax for another Tivoli Storage Manager parameters. NULL causes the parameter to be reset to nothing.)
- Backup db DBNAME to /test (backup to a directory).

A.3 Sun Solaris quick start

Here is a quick start guide for backing up DB2 to Tivoli Storage Manager. This will be based on the Sun Solaris and the 3.7 or 4.1 Tivoli Storage Manager API level.

DB2 provides native support for backing up to Tivoli Storage Manager. DB2 utilizes the Tivoli Storage Manager API to perform these backups. The following is the minimum steps necessary to get DB2 to backup to Tivoli Storage Manager:

1. Install DB2.

2. Install Tivoli Storage Manager API.
3. Create a plain text file `dsm.opt` in `/opt/tivoli/tsm/client/api/bin`.
It only needs one line, the value for `SERVERNAME` is arbitrary, but must match the `dsm.sys` file:

```
SERVERNAME DB2
```

4. Create a plain text file `dsm.sys` in `/opt/tivoli/tsm/client/api/bin`.

It must contain at least the following:

```
SERVERNAME DB2  
COMMETHOD TCPIP  
TCPSEVERADDRESS xxx.xxx.xxx.xxx  
NODENAME DB2TSM  
PASSWORDACCESS GENERATE
```

5. Register the node `DB2TSM` on the Tivoli Storage Manager server.
We recommend for you to create a new `DOMAIN` on the Tivoli Storage Manager server for this node. It needs only one management class with a copygroup that has the following retention settings:

```
VEREXIST=1  
VERDELETED=0  
RETEXTRA=0  
REONLY=0
```

6. Sign-on as root and `cd` to the `db2` directory that contains the files `db2adutl` and `dsmapiw`.
7. Run the executable '`dsmapiw`' and follow the prompts.

It will ask for the current Tivoli Storage Manager node password, then for a new password, and confirmation of the new password. It should report if it is successful.

8. Run the command '`db2adutl query`' to confirm that the password file was correctly set.

If the command comes back and says no `db2` objects found, then it is working (you haven't done any backups yet). You should be able to check the activity log on the Tivoli Storage Manager server to confirm that the node `DB2TSM` authenticated.

9. You can now perform a backup using the command line or the `db2 connect` GUI. For example, `db2 backup db DBNAME use tsm`.

Additional `DB2` information:

- In the /export/home/db2instance/sqllib directory run '. db2profile' to update the signed on user with the needed environment variables (DB2INSTANCE=db2inst1, for example).
- Start db2 command line processor via /export/home/db2instance/sqllib/bin/db2.
- Get db cfg for DBNAME.
- Update db cfg for DBNAME using TSM_PASSWORD NULL. (Similar syntax for other Tivoli Storage Manager parameters. NULL causes the parameter to be reset to nothing.)
- Backup db DBNAME to /test (backup to a directory).

Appendix B. Troubleshooting

In this appendix, we will have a look at common errors and how they can be fixed. We give a list of the most useful logfiles and techniques to determine the source of the problem.

A useful thing to do when troubleshooting a problem is to start a Tivoli Storage Manager administrative client in console mode. This will allow you to see the activity that is occurring on the Tivoli Storage Manager server. Among other things you will see node sessions start and stop and any Tivoli Storage Manager server error messages.

The administrative client console can be started with the `dsmadm -cons` command. See the following example where we monitor how the Tivoli Storage Manager client BRAZIL_DB2 connects and disconnects from the Tivoli Storage Manager server:

```
$ dsmadm -cons
Tivoli Storage Manager
Command Line Administrative Interface - Version 4, Release 1, Level 2.0
(C) Copyright IBM Corporation, 1990, 1999, All Rights Reserved.

Enter your user id: admin

Enter your password:

Session established with server BRAZIL: AIX-RS/6000
  Server Version 4, Release 1, Level 2.0
  Server date/time: 04/02/01 17:02:02 Last access: 04/02/01 17:01:01

ANR0406I Session 9114 started for node BRAZIL_DB2 (DB2/6000) (Tcp/Ip
9.1.150.57(39918)).
ANR0403I Session 9114 ended for node BRAZIL_DB2 (DB2/6000).
```

B.1 Gotchas

One of the most common problems is that the Tivoli Storage Manager API client is not able to connect to the Tivoli Storage Manager server.

B.1.1 RC 406 options file not found

If you receive a return code of 406, it means that the client options file `dsm.opt` cannot be found. Here is an example of the error:

```
db2 => backup db sample user db2admin using itsosj use tsm
SQL2062N An error occurred while accessing media
"C:\PROGRA~1\SQLLIB\bin\db2tadsm.dll". Reason code: "406"
```

B.1.1.1 Windows

Make sure that the environment variable `DSMI_CONFIG` is specified and that it is specified as a `SYSTEM` variable and not a `USER` variable. Verify that the `DSMI_CONFIG` contains the complete path and filename to the options file, and that the options file has been created at that path with that filename. Perform a `db2stop` and then a `db2start` to ensure that the variable is loaded into the DB2 run-time engine. The variable is case sensitive so be sure that it is in upper case.

B.1.1.2 UNIX

Make sure that the environment variable `DSMI_CONFIG` is specified in the `.profile` or `db2` profile of the instance owner. Verify that the `DSMI_CONFIG` contains the complete path and filename to the options file, and that the options file has been created at that path with that filename. Perform a `db2stop` and then a `db2start` to ensure that the variable is loaded into the DB2 run-time engine. A simple way to check this is to login as the instance owner and execute the command `env | grep DSMI`. Then view the contents of the file with the command `more $DSMI_CONFIG`. The variable is case sensitive so be sure that it is in upper case.

B.1.2 Running a Tivoli Storage Manager CONFIG trace

You can waste a lot of time troubleshooting, because you were using the wrong Tivoli Storage Manager client options file (see 4.6, “Node considerations” on page 53). There is a simple trace that can be run that provides a lot of useful information. To run this trace add two trace lines to your `dsm.opt` file (for both UNIX and Windows). The trace lines are `traceflag config` and `tracefile c:\temp\apitrace.out`. Modify the path and filename of the `TRACEFILE` line to your operating system. The path must exist.

On Windows our `dsm.opt` file looks like this after adding these trace lines.

```
commethod tcpip
tcpport 1500
tcpserveraddress 193.1.1.11

nodename jamaica_db2
passwordaccess generate

traceflag config
tracefile c:\temp\apitrace.out
```

With the trace lines in the client options file, the next time db2adutl or a DB2 backup using Tivoli Storage Manager is run, a file will be created as specified by the tracefile command.

If this file is created, you have already verified something important, that is, the client options file that you modified is the one that the db2adutl or DB2 is using.

Open this file with a text editor. This trace will show you information regarding the level of the Tivoli Storage Manager API, tcpserveraddress, commethod, errorlogname, ds_dir, passwordaccess, and so on. Pay special attention to the errorlogname and the ds_dir.

The value for errorlogname will either be dserror.log or a path and dserror.log. For UNIX, the DB2 instance owner must have write access to the directory and path of the errorlogname. If this value is just dserror.log with no path, DB2 will try and create the dserror.log in the root directory /dserror.log. Since the DB2 instance owner does not have write access to this directory the dserror.log will not be created. This is set with the DSMI_LOG environment variable.

The ds_dir value corresponds to the DSM_DIR environment variable. For UNIX, this value also determines from which directory the dsm.sys file will be read. Most Tivoli Storage Manager installations have a dsm.sys in the ../tivoli/tsm/ba/bin and the ../tivoli/tsm/api/bin directories. When matching the servername value in the dsm.opt file with the corresponding servername in the dsm.sys, use the ds_dir value to determine which dsm.sys is being used.

B.1.3 RC 137 authentication failure, incorrect password

The most common cause is that passwordaccess generate is not specified. Another possible cause is that you are not specifying the correct password, or your nodename is wrong. To update the password on the Tivoli Storage Manager server, run the command update node nodename password. Replace

nodename and password with the appropriate values. Use the Tivoli Storage Manager administrative client in console mode to verify which node is trying to access the Tivoli Storage Manager server.

To regenerate the password file, run `dsmapipw.exe` again.

B.1.4 DB2 User exit on Windows NT cannot find .h (header) files

Reinstall the Backup-Archive client. Select a custom install and make sure to select the API SDK files.

B.2 Checklist for the Tivoli Storage Manager server

If you are encountering errors, there are some things that you should check on the Tivoli Storage Manager server using the administrative client.

Run the command `query node nodename f=d`, replacing `nodename` with the nodename that is being used to backup DB2. The values 'Archive delete allowed' and 'Backup deleted allowed' should both be set to YES. The value 'Maximum Mount Points Allowed' should be equal to or less than the number of drives on the Tivoli Storage Manager server. The value 'Locked?' should be set to NO. The value for 'Policy Domain Name' should be checked to make sure it belongs to the correct domain. This value determines which management classes are available to the node for managing the DB2 backups.

Using the value of 'Policy Domain Name' you can run some queries to see if the correct management classes for the DB2 backups are available. The command `query mgmtclass <domain> active` will show which is the DEFAULT management class for the node. The command `query copygroup <domain> active` will tell you what the retention settings are for the managementclasses.

B.3 Isolating the problem

When troubleshooting, it is necessary to try and isolate the cause of the problem. The three main areas to try to isolate the problem: DB2, Tivoli Storage Manager API, and Tivoli Storage Manager server. A key thing to remember is that Tivoli Storage Manager is primarily a storage repository for DB2 backups. DB2 is responsible for creating all the backup objects and the archivelogs. Tivoli Storage Manager is responsible for storing the backup objects and the archivelogs, and then retrieving them when asked. If DB2

creates a corrupt backup object or corrupt archive log, Tivoli Storage Manager will store and retrieve them in the same corrupt state.

Tivoli Storage Manager is not required to do a DB2 backup so you can use DB2 alone to isolate a problem. For example, you can create a DB2 backup on a local filesystem. In fact you can use the `db2adutl.exe` to extract a backup from Tivoli Storage Manager storage to disk. Then you can restore the backup without using Tivoli Storage Manager. If Tivoli Storage Manager fails when extracting a backup to disk, then it is a Tivoli Storage Manager server or Tivoli Storage Manager API problem. If DB2 fails in restoring the backup from disk, then it is probably a DB2 problem.

Another useful way to isolate a problem is to compare a failing machine with a working one. Then you can see what is the same and what is different. Are they using the same API level, DB2 level and fixpack, Tivoli Storage Manager server? Do the nodes belong to the same domain and have the same settings? What do the client options file look like between a failing and a working machine?

B.4 List of logfiles

The following is a collection of logfiles to get valuable information for problem determination. They are relevant for our purpose (backing up DB2 using Tivoli Storage Manager).

- `dsierror.log`

The file resides in the directory specified by the `DSMI_LOG` environment variable. It contains the most valuable information for problem determination for our purpose. Each entry may refer to a Tivoli Storage Manager API error number. A description of those errors can be found in “Appendix D, API Return Codes” in *Tivoli Storage Manager Using the Application Program Interface V4R1*, SH26-4123.

- `ARCHIVE.LOG`

The file resides in the directory specified by the `AUDIT_ERROR_PATH` in the user exit program. All successful and unsuccessful user exit archive operations are logged to this file.

- `RETRIEVE.LOG`

This is found in the directory specified by the `AUDIT_ERROR_PATH` in the user exit program. All successful and unsuccessful user exit retrieve operations are logged to this file.

- `USEREXIT.ERR`

The file resides in the directory specified by the AUDIT_ERROR_PATH in the user exit program. Only the unsuccessful user exit archive or retrieve operations are logged to this file.

The most common user exit return code is return code 16 which means a software error. The additional (mostly Tivoli Storage Manager API) return codes supplied within the USEREXIT.ERR file must be analyzed by looking at the related software product error messages reference book. For a listing of user exit return codes, see B.4.1, "User exit return codes" on page 283.

The example below shows an example USEREXIT.ERR entry. Tivoli Storage Manager API return code 186 means that no archive copy group has been defined for the destination management class.

```
*****
Time of Error:      Thu Mar  1 19:10:15 2001

Parameter Count:   8
Parameters Passed:
Database name:     SAMPLE
Logfile name:      S0000000.LOG
Logfile path:      /home/db2inst1/db2inst1/NODE0000/SQL00001/SQLLOGDIR/
Node number:       NODE0000
Operating system:  AIX
Release:           SQL07010
Request:           ARCHIVE
Audit Log File:    /home/db2inst1/tsm/ARCHIVE.LOG
System Call Params:
Media Type:        ADSM
User Exit RC:      16

> Error isolation: dsmBindMC() returned 186
```

- `$HOME/sqllib/db2dump/db2diag.log`

This file contains all logged information from DB2. To get the most information the default diaglevel (which is 3) can be changed to 4, with following command:

```
db2 update dbm cfg using diaglevel 4
```

In general, a DB2 log entry may contain a hexadecimal number representing an DB2 internal return code. For a listing of these numbers, see "Interpreting the db2diag.log" in *DB2 UDB Troubleshooting Guide V7*, GC09- 2850.

If DB2 receives a return code (>0) from the `db2uext2` command a line like the following will appear in the `db2diag.log` file:

```
User Exit returned error on ARCHIVE log file S0000114.LOG from /home/db2inst1/db2inst1/NODE0000/SQL00001/SQLLOGDIR/ for database SAMPLE, error code 21
```

The error code numbers are defined in the db2uext2 source code. See B.4.1, “User exit return codes” on page 283, for a complete listing of user exit return codes.

- \$HOME/sqllib/db2dump/db2alert.log

If an error is determined to be an alert, then an entry is made in the db2alert.log file.

- dsmsched.log

By default, Tivoli Storage Manager stores the schedule log information in a file named dsmsched.log. The default name can be overwritten by specifying the SCHEDLOGNAME option in the client option files.

B.4.1 User exit return codes

```
#define RC_OK 0 /* ok */
#define RC_RES 4 /* resource allocation error */
#define RC_OPATTN 8 /* operator/user attention required*/
#define RC_HARDWARE 12 /* hardware error */
#define RC_DEFECT 16 /* software error */
#define RC_PARM 20 /* invalid parameters */
#define RC_AUDIT 21 /* error open audit file */
#define RC_NOTFOUND 24 /* db2uext2() / file not found */
#define RC_UNKNOWN 28 /* unknown error */
#define RC_OPCAN 32 /* operator/user terminated */
```

Appendix C. Performance

In this appendix, we give some suggestions about how to accelerate the DB2 backup process when using Tivoli Storage Manager. Most of the information was gathered from *DB2 UDB V7.1 Performance Tuning Guide*, SG24-6012.

First, we show the possible bottlenecks that the backup data has to go through until arriving at the Tivoli Storage Manager server. All these parts may require tuning. Some of these will bring a big improvement to the backup performance, some less so. Then we discuss the DB2 backup and restore commands in more detail.

C.1 General performance considerations

General performance considerations are those which may include factors outside the boundaries of the DB2 and Tivoli Storage Manager products themselves. Some of these factors could affect any backup strategy or product.

- Physical disk

A DB2 database should be placed over multiple disks so the disks can be read in parallel. Also, the faster the disk drives, the faster the data for backup can be read. Read or write cache will bring small or no improvement, because the whole database needs to be read and it is unlikely that the same data is read again by the backup process.

- DB2 database

DB2 can be tuned to gain faster access to its data that relies on the physical disk. It can also be tuned to better utilize the resources of the operating system like memory, multiple CPUs or I/O subsystem. This theme is worth its own redbook and luckily there is a good one. See *DB2 UDB V7.1 Performance Tuning Guide*, SG24-6012.

- Backup command

The DB2 backup command provides special options to speed up the backup process. These options try to use multiple processes to read the data and multiple sessions to send the data to Tivoli Storage Manager. We will discuss these parameters in more detail later in this appendix, because they will bring the best improvements for a better backup performance.

- Operating system
Tuning the operating system normally will not bring that much of an increase in performance to the backup process. Although, one tuning measure is not to restrict the DB2 process' access to memory, CPU or I/O subsystem.
- Tivoli Storage Manager client
Some tuning configurations can be done in the dsm.sys file. The purpose is to tune TCPIP (for example, TCPWindowSize) or to group smaller files together (TXN-parameters).
- Communication
In some cases, the performance of the backup is reduced by the LAN that is used. The best practice is to use a dedicated LAN for backup needs. This will prevent the backup from interference from other users or processes, and vice versa.
- Tivoli Storage Manager server
Similar to the Tivoli Storage Manager client, the Tivoli Storage Manager server can be tuned. However, this tuning will often take effect for *all* the Tivoli Storage Manager clients and must be planned very carefully. One important tuning issue on the server is to schedule the time when client backups take place, to have enough resources (for example, tape mounts) available for the backup or restore.

C.2 DB2 backup command performance options

There are four options on the DB2 backup command that can improve backup performance. We will give some suggestions for the values to use for these parameters, but review of DB2 documentation is recommended.

- OPEN n SESSIONS
This parameter is only valid if the `use tsm` option is used at the DB2 backup. If specified the backup process will open n I/O sessions to the Tivoli Storage Manager server to send the data to. The number of sessions is limited by the Tivoli Storage Manager node parameter MAXNUMMP which should not be greater than the maximum number of drives at the Tivoli Storage Manager server. If the backup goes directly to a tape storage pool and there are not enough tape drives available at the Tivoli Storage Manager server at the time of backup, some of the sessions have to wait until there are free tape mounts available again. In some cases, the backup may fail if there are not enough mount points available

within a defined amount of time (IDLETIMEOUT parameter at Tivoli Storage Manager server).

- WITH num-buff BUFFERS

The numbers of buffers should be: $\#sessions + \#parallelism + 2$. Also, the following calculation must fit: $(num-buffers * buffer-size) < UTIL_HEAP_SZ$ (UTIL_HEAP_SZ is the database utility heap size).

- BUFFER buff-size

This value is used as the buffer allocation size in pages (4 KB) when building the backup image. For a buffer size of zero, the value of the database manager configuration parameter BACKBUFSZ will be used as the buffer allocation size. When backing up a database, the data is first copied to an internal buffer. Data is then written from this buffer to the backup media when the buffer is full. Tuning BACKBUFSZ can help improve the performance of the backup utility as well as minimize the impact on the performance of other concurrent database operations.

We recommend setting the buffer size to a multiple of the extent size. If multiple table spaces have different extent sizes, the buffer size value should be a multiple of the largest extent size.

- PARALLELISM n

Using this parameter can dramatically reduce the time required to complete the backup (especially if the backup is going to a disk). This parameter defines the number (n) of processes that are started to read data from the database. Each process is assigned to backup a specific table space. When it completes backing up the tablespace, it requests another. Each process will be assigned a tablespace to complete, therefore, to get better performance, let this value be less than the number of tablespaces, since setting up the value higher than the number of tablespaces does not show any significant difference in performance. If backing up to different targets (for example, using multiple sessions to send the data to Tivoli Storage Manager) parallelism should not be greater than the number of targets (sessions)).

In the next example, we assume that we have two tape mount points available at the Tivoli Storage Manager server. We can use two sessions and also a parallelism of 2 (no matter how many tablespaces we have):

```
db2 backup db sample use tsm open 2 sessions with 6 buffers parallelism 2
```

C.3 DB2 restore command performance options

There are also factors that affect the restore process. We will discuss only the restore command options that will bring a better restore performance. They are almost the same as for the backup command.

- OPEN n SESSIONS:

This parameter is only valid if the `use tsm` option is used on the DB2 restore command. If specified, the restore process will open n I/O sessions to the Tivoli Storage Manager server to receive the data. The number of sessions is limited by the Tivoli Storage Manager node parameter `MAXNUMMP`, which cannot be greater than the maximum number of drives at the Tivoli Storage Manager server. This number should also not be higher than the number of sessions used for backup. If there are not enough tape drives available at the Tivoli Storage Manager server at the time of restore, some of the sessions have to wait until there are free tape mounts available again.

- WITH num-buff BUFFERS

The numbers of buffers should be: `#sessions + #parallelism + 2`. Also, the following calculation must fit: `(num-buffers * buffer-size) < UTIL_HEAP_SZ` (`UTIL_HEAP_SZ` is the database utility heap size).

- BUFFER buff-size

We recommend setting the restore buffer size to a multiple of the extent size. The value given must be equal to, or a multiple of, the backup buffer size that was specified when the backup image was taken. Otherwise, the smallest acceptable size of the buffer (4KB) is used for the restore buffer. If you do not specify the number of pages, each buffer will be allocated based on the database manager configuration parameter `RESTBUFSZ`.

- PARALLELISM

This specifies the number of buffer manipulators to be spawned during the restore process. The default value is 1. For restore, the number can be increased, but not higher than the number of Tivoli Storage Manager client sessions that have been started for the restore. The number of buffers should be at least the value selected for parallelism.

In the next example, we assume that the backup was made using two Tivoli Storage Manager sessions. For restore, it is also reasonable to use two Tivoli Storage Manager sessions and a parallelism of 2 (no matter how many tablespaces we have):

```
db2 restore db sample use tsm open 2 sessions with 6 buffers parallelism 2
```


Appendix D. Split mirror and split copy functions with DB2

In this section, we describe the split mirror feature of intelligent disk storage servers, such as the IBM Enterprise Storage Server (ESS), and some possible ways that DB2 can use this feature.

Split mirror is a feature that allows you to establish an identical and independent copy of a disk volume. The source and target volumes must reside in the storage server and, depending on the vendor, need to have special requirements to allow such a copy. For example, the target disk must be at least the same size as the original disk, or the target disk must be in a special relation to the source disk. Please see the vendor documentation for the requirements for split mirrors and how to set up the split mirror.

The target volumes should be assigned to another physical machine. If the target volumes are assigned to the same machine where the source volumes are, the operating system may have problems identifying the new disk volume, as it is an exact copy of the original disk. Unique disk identifiers, size of the disk, the copied data itself, and so on will appear twice on the same machine which may cause problems.

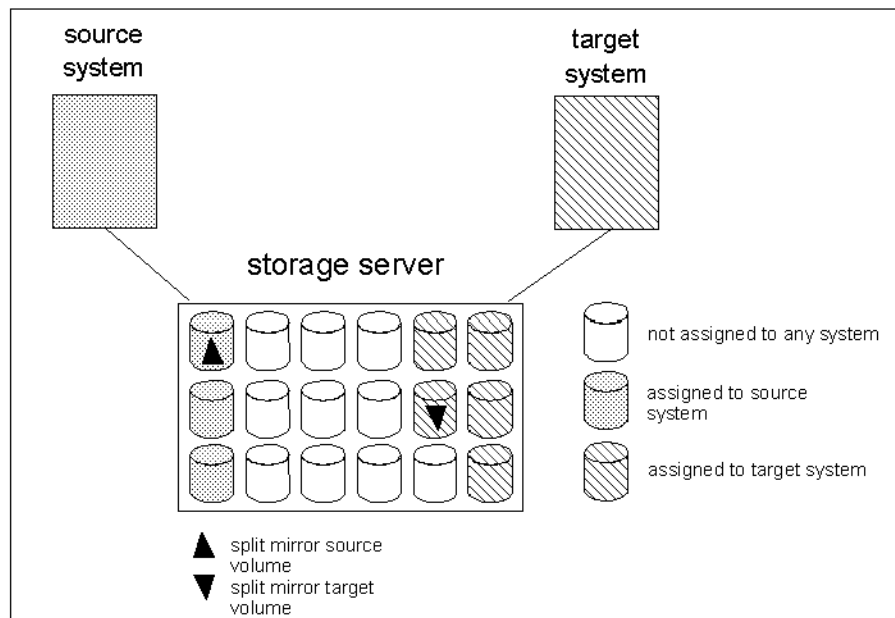


Figure 168. Split mirror concept

Now, we look at how the split mirror feature is implemented on the IBM ESS. On the IBM ESS, the split mirror feature is named FlashCopy®.

Both the source and target volumes reside in the ESS. Normally, the source volume is only visible to the source machine and the target volume only to the target machine.

At a specific time the FlashCopy relationship between source and target volumes will be established. This is known as the T0 (time zero) copy. The process of establishing this T0 copy takes only a few seconds. A bitmap will be created to identify the T0 data on the source disk.

At the time when FlashCopy is started, the target volume is basically empty. The background copy task copies data from the source to the target. The FlashCopy bitmap keeps track of which data has been copied from source to target. If an application wants to read some data from the target that has not yet been copied to the target, the data is read from the source; otherwise, the read is satisfied from the target volume.

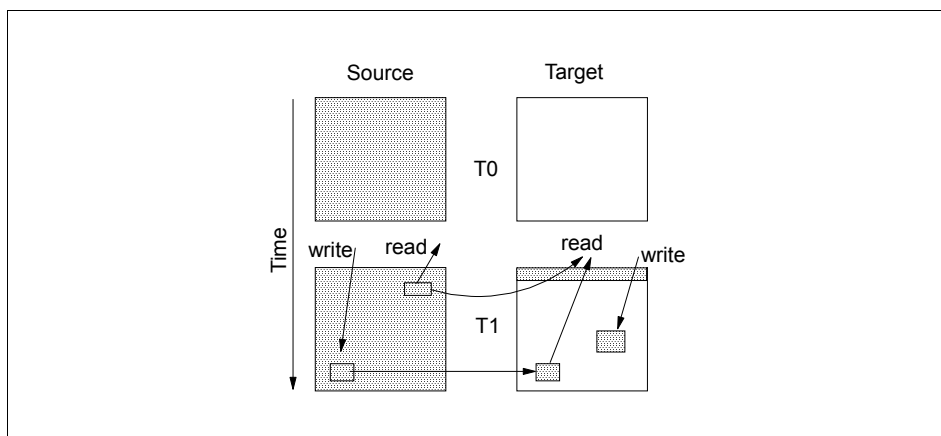


Figure 169. Implementation of split mirror

Before an application can update a track on the source that has not yet been copied, the track is copied to the target volume. After some time all the data is copied either by the background copy process or by write operation on either the source or the target volume.

Note

when establishing the split mirror the data on the source volume must be in a consistent state. This can either be achieved by stopping the application and flushing the data from memory to disk, or if the application cannot be stopped, then by application itself providing a feature to ensure that the data is consistent while the FlashCopy is being established and can recover from that state afterwards on the target side.

DB2 provides the suspended I/O feature and `db2inidb` tools to make use of the split mirror feature.

D.1 Using split mirror features in conjunction with DB2

The DB2 suspend I/O feature and the `db2inidb` tool were introduced with DB2 V7.1 Fixpak2. We show how, and in which order, the commands must be used to make use of the split mirror features. (Also, see the Release Notes of Fixpak 2 for further documentation.) We will describe three scenarios, which are the three parameters *snapshot*, *standby*, *mirror* of the `db2inidb` command.

The suspended I/O support is necessary to bring the database into a consistent state before establishing the split mirror. With the `write suspend` command, the database enters a well defined state where it can recover later using the `db2inidb` tool. During the suspend of the database, the DB2 clients will see that their transactions will hang. As soon as the database is resumed again for writing (`write resume`), the transactions will proceed normally.

The `db2inidb` tool is necessary in combination with the split mirror feature, to bring the target database into a running state again after the split mirror was established. The `db2inidb` tool can perform crash recovery or put a database in rollforward pending. There are three possible options:

- Snapshot

This initiates crash recovery, making the database consistent. A new log chain starts, and the database will not be able to roll forward through any of the logs from the original database. The database is available for any operation, including backup.

- Standby

This places the database in rollforward pending state. Crash recovery is not performed, and the database remains inconsistent.

- Mirror

This causes a mirrored copy of the database to replace the original database. The database is placed in rollforward pending state, and the `write suspend` state (discussed later in D.1.3, “DB2 mirror database” on page 294) is turned off. Crash recovery is not performed, and the database remains inconsistent.

We will now show three scenarios in which DB2 can use the split mirror features of storage servers. It is assumed that the source and target DB2 instance reside on different machines. Furthermore, the setup of the DB2 instance must be the same on the source and on the target machine, for example, `userid` and `groupid` of the instance owner. Also, the operating system and DB2 versions should be the same.

D.1.1 DB2 clone database

The first scenario is to setup a clone database on the target system. A clone database can be used for taking an offline backup, generating a test database, or generating reports.

- Suspend I/O on source database

The following DB2 command will suspend all write activities from DB2 clients, but they will not be disconnected from the database:

```
db2 set write suspend for database
```

- Establish split mirror

This step is dependent on the vendor. See the appropriate vendor documentation on how to establish the split mirror. The whole database must be copied to use it on the target machine.

- Resume I/O on source database

After a few seconds the split mirror is established and the database can be made available again for writing. The transactions will proceed as normal.

```
db2 set write resume for database
```

- Make data visible on target machine

The target volumes will now contain all the data from the time the split mirror was established, but they may not yet be visible to the target operating system. The operating system on the target machine has to provide access to the data (`import`, `mount`, and so on). (For initial setup, the database needs to be cataloged at the target machine.)

- Start target (clone) database

Issue the `db2start` command at the instance at the target machine.

- Bring the clone database into a consistent state. (Perform crash recovery.)

The following command will rollback every uncommitted transaction:

```
db2inidb <target-database> as snapshot
```

The clone database can now be used for an offline backup, or as a test database.

Note

Any DB2 backup image taken on the cloned database can *not* be used for restore on the original database for the purpose of performing rollforward recovery using the log files produced on the original database after the split mirror.

For the purpose of a full offline database backup the same Tivoli Storage Manager client configuration can be used on the target machine to send the backup to Tivoli Storage Manager. This will allow you to restore the database image on the source database using Tivoli Storage Manager. It is not possible to do rollforward recovery after the full offline database image is restored.

Warning: If the database is to be used as a test database, then the full database backups and the user exit, if configured, must use another Tivoli Storage Manager node. Otherwise, the logfiles from the test and the source database will get mixed up at the Tivoli Storage Manager server, and there is no way to separate them afterwards.

D.1.2 DB2 standby database

The second scenario is to setup a standby database. A standby database in this context is a database where the logfiles of the source database will be applied on the target (standby) database. The standby database will be in a permanent rollforward process as long as the source-database produces logfiles (and as long as the source database is available).

Note

Do *not* copy the logfiles using the split mirror feature. This will break your rollforward chain. The necessary logs for rollforward must all be acquired from the source database as soon as they are available (no longer used by the source database). If logfiles have been copied to the target machine using the split mirror feature, then they must be deleted.

The steps required to setup a standby database differ only at the last step of scenario one (setup a clone database). We only cover the last step here.

- Set the database into rollforward mode:

```
db2inidb <target-database> as standby
```

Now the logfiles from the source-database can be used to rollforward the target-database. As long as there are new logs available at the source-database, the rollforward step can be repeated. (To make the transfer of the logfiles easier, a user exit may be configured.)

```
db2 rollforward db <target-database> to end of logs
```

As long as the database stays in rollforward pending mode, no database backups are allowed.

If the source database crashes the standby database can be activated for access. (Make sure that all data has already been copied to the target volumes.) The rollforward must be stopped with the `stop` option of the DB2 rollforward command.

```
db2 rollforward db <target-database> stop
```

Then the users can switch over to the standby database to continue their work.

D.1.3 DB2 mirror database

The third scenario shows how to use the `mirror` option of the `db2inidb` tool. The purpose of this option is to provide the possibility of using a split mirror database for restore on the source database and then to rollforward the logs of the source database.

Note

Do *not* copy the logfiles using the split mirror feature. All necessary logs for recovery must stay at the source database or on another location, but not on the target database. If logfiles have been copied to the target machine using the split mirror feature, then they must be deleted.

The steps required are:

- Suspend I/O on source database

The following DB2 command will suspend all write activities from DB2 clients, but they will not be disconnected them from the database.

```
db2 set write suspend for database
```

- Establish split mirror

This step is dependent on the vendor. See the appropriate documentation for how to establish the split mirror. The logfiles should not be copied to the target database. Make sure that all data will be copied to the target machine sooner or later.

- Resume I/O on source database

After a few seconds the split mirror is established and the database can be made available again for writing. The transactions will proceed as normal.

```
db2 set write resume for database
```

- Make data visible on target machine

The target volumes will now contain all the data from the time the split mirror was established, but they may not yet be visible to the target operating system. The operating system on the target machine has to provide access to the data (import, mount, and so on). (For first time setup, the database needs to be cataloged at the target machine.)

Note

No DB2 operations are allowed on the target (split mirror) database after the split mirror was established in order to use the target database for restore on the source database. The database needs to stay in this “frozen” state.

The target database cannot be backed up using DB2, but can be backed up using operating system tools.

If the source database happens to crash it can be restored with the split mirror image at the target database. See the following steps:

- Stop the source database (we want to restore this database)

```
db2stop
```

- Restore the database

With operating system tools (for example, copy, xcopy, cp, tar and so on) copy the datafiles of the split mirror database over the original database. But do *not* copy the logfiles from the target database to the source database. (Make sure that all data has already been copied to the target database.)

Another restore possibility would be to “reverse” the split mirror and copy the data on the target disk volumes back to the source disk volumes using the split mirror feature of the storage server again.

- Start the source database again.

```
db2start
```

- Reestablish the mirrored copy on the source database

```
db2inidb <database> as mirror
```

Now a rollforward to end of logs of the database can be started. The logs from the original source database will be used.

D.2 Using AIX splitcopy feature for DB2 backup

In this section, we will look at the AIX splitcopy feature and how it can be used for DB2 backup purposes. The AIX splitcopy feature was introduced with AIX 4.3.3.0. Its purpose is to provide an online backup of an AIX journaled filesystem (JFS).

DB2 can use this splitcopy feature to create a DB2 datafile backup. This backup is simply a copy of all the datafiles of a DB2 database. This backup can be restored and then a rollforward can be done. This is almost the same as the third scenario of the D.1, “Using split mirror features in conjunction with DB2” on page 291.

First, we describe the AIX splitcopy feature.

Each AIX JFS must reside on a logical volume. All logical volumes are managed by the AIX logical volume manager (LVM). With the AIX LVM, it is possible to mirror a JFS by mirroring the logical volumes the JFS is built on. A logical volume can have one (no mirror), two, or three physical copies of the same data.

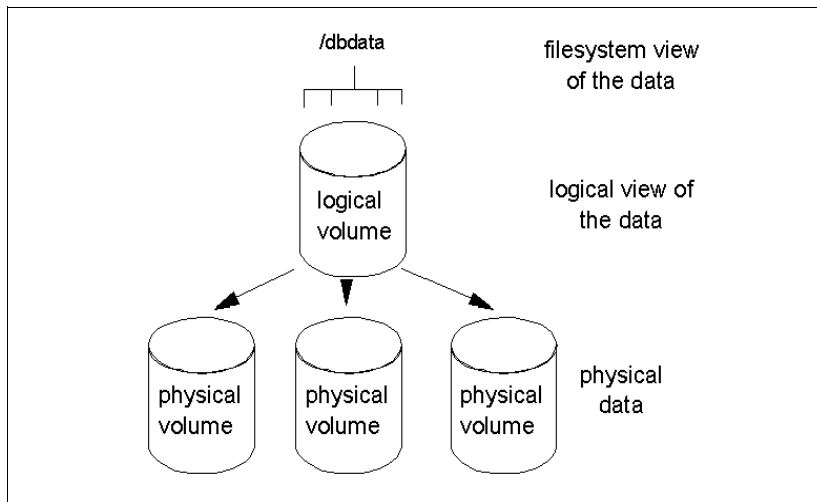


Figure 170. AIX LVM mirroring with three physical copies

The following requirements are necessary to use the AIX splitcopy feature:

- Logical volume must be mirrored with AIX LVM
- Journaled files system log (JFS log) must also be mirrored
- The number of copies for the JFS log must be equal the number of copies for the filesystem's logical volume
- If multiple filesystems needs to use the splitcopy feature at the same time, each filesystem must have its own mirrored journal file system log

Splitcopy separates one physical copy of the mirrored logical volume and assigns it to a new filesystem. This new splitcopy filesystem will be read only. The splitcopy can be established with the following command.

```

# chfs -a splitcopy=/dbcopy /dbdata
splitlvcopy00
backup requested(0x100000)...
log redo processing for /dev/splitlvcopy00
syncpt record at de9c08
syncpt record at d69b74
end of log e67488
syncpt record at de9c08
syncpt address d49074
number of log records = 10552
number of do blocks = 59
number of nodo blocks = 4
# lsvg -l rootvg | grep db
splitlv          jfs          6      12      2      open/stale  /dbdata
splitlvcopy00    jfs          0      0      0      open??????? /dbcopy

```

The original JFS will now be in a stale mirrored state, because the one physical disk that was split off the mirror will not be synchronized any more. But the relation between the split off physical volume and the LVM mirrored logical volume still exists.

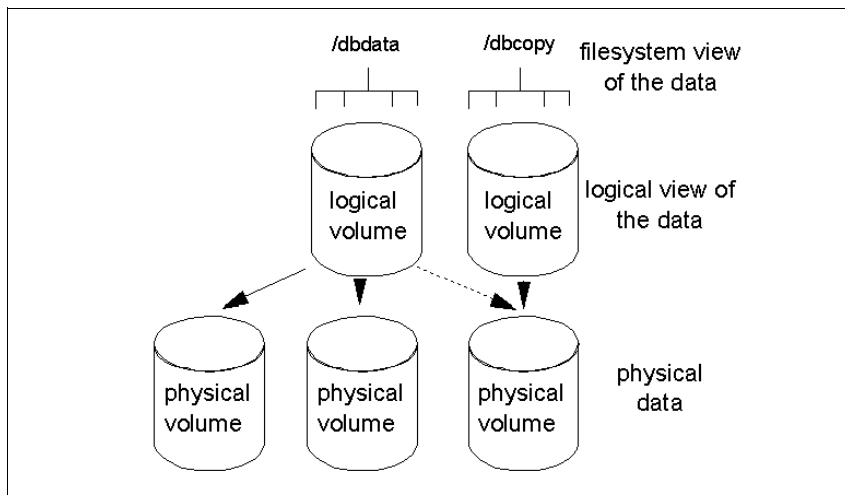


Figure 171. Splitcopy read-only filesystem

For more information about splitcopy, see “5.8 Online JFS Backup” in *AIX Version 4.3 Differences Guide*, SG24-2014.

DB2 can use this splitcopy feature to create a database backup image on the splitcopy filesystem that can be used to restore the DB2 database and also rollforward the logs.

The following steps are required for backup using AIX splitcopy:

- Create the mirrored filesystems and, if needed, the mirrored logfiles.

The best practice is to use three copies, so that the production database will have at least two copies to provide availability after the splitcopy was established.

- Create database on one or more (LVM) mirrored filesystems.

We do not recommend for you to put the logfiles on this mirrored filesystem. The backup and the restore of the database will be easier.

- Suspend I/O to database.

Suspend access from clients to the database. The clients will not lose their connection, but the write and update transactions will hang.

```
db2 set write suspend for database
```

- Establish splitcopy.

For each filesystem, issue the following command as root user. It is possible to specify which mirror copy to split off by using the `-a copy` option. The default is to split off the second copy. Valid options are 1,2, or 3.

```
chfs -a splitcopy=/dbcop -a copy=3 /dbdata
```

- Resume I/O on database.

```
db2 set write resume for database
```

- Backup database.

As instance owner (or root user) use an AIX backup command to backup the datafile of the database. Ensure that all files are being backed up. For example:

```
cd /dbcop; tar -cvf /tmp/db2inst1.tar ./db2inst1
```

- Re-establish original mirror.

This can be done by removing the generated splitcopy filesystem. As root user issue a command like the following:

```
rmfs -r /dbcop
```

This will synchronize the splitcopy physical volume again into the original mirror.

The following steps are required for restoring a backup that was made using AIX splitcopy:

- Stop the database.

```
db2stop
```

- Restore the backup image on the original destination.

As instance owner (or root user) restore the backup image. Be sure that no original logs will be overwritten. For example:

```
cd /dbdata; tar -xvf /tmp/db2inst1.tar
```

- Start the database instance.

```
db2start
```

- Reestablish the mirrored copy.

As instance owner issue following command:

```
db2inidb <database> as mirror
```

- Rollforward the original logs.

```
db2 rollforward db <database> to end of logs and stop
```

Appendix E. “New” backup features (DB2 V7.1 Fixpak3 Beta)

In this appendix we will look at the DB2 incremental and the on demand log archive feature of DB2. **Note:** All the features described in this appendix require DB2 V7.1 fixpak 3 or higher.

E.1 Incremental backup

A big database results in a big database backup image. Every time a full database backup is started, resources like network or disk I/O will be required during the time of backup. Also if only a few percent of data in the database has changed the whole database will be backed up again when doing a full database backup. To overcome this problem incremental backups can be used.

There are three types of backups:

- Full

This is a classical full database backup. Also known as level 0 backup.

- Incremental

This is a database backup that contains only the changes since the last full (level 0) database backup. Incremental backups are also known as level 1 backups.

- Delta

This is a database backup that contains only the changes since the last backup of every type (full, incremental or delta).

See figure Figure 172 for an example of a backup strategy using incremental or delta backups.

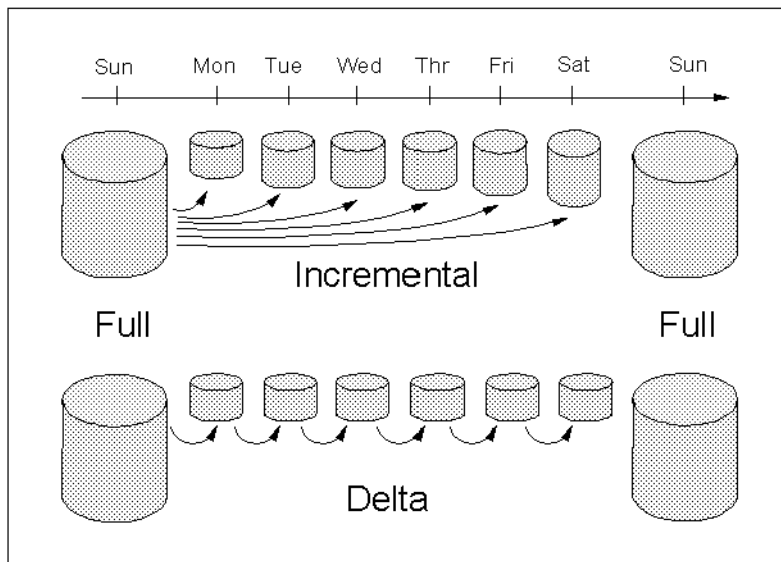


Figure 172. Incremental backup

To enable the use of incremental backups, the DB2 database configuration parameter `TRACKMOD` must be set to on. To activate this parameter a `db2stop` and `db2start` may be necessary.

```
$ db2 update db cfg for sample using TRACKMOD ON
DB20000I The UPDATE DATABASE CONFIGURATION command completed successfully.
DB21026I For most configuration parameters, all applications must disconnect
from this database before the changes become effective.
$
```

After that a full database backup is required to initialize the chain of backups. This can be an offline or online backup.

```
$ db2 backup db sample online use tsm

Backup successful. The timestamp for this backup image is : 20010327153632
$
```

Now we are able to generate incremental or delta backups.

Incremental and delta backups can also be sent to a Tivoli Storage Manager server. From a Tivoli Storage Manager point of view they are treated the same as database backups. They will also be sent to the backup copy group

of the associated Tivoli Storage Manager management class. There is no further Tivoli Storage Manager setup required.

To start an incremental backup the `incremental` option of the backup command must be specified.

```
$ db2 backup db sample online incremental use tsm
Backup successful. The timestamp for this backup image is : 20010327161558
```

To start a delta backup the `incremental delta` option of the backup command must be specified.

```
$ db2 backup db sample online incremental delta use tsm
Backup successful. The timestamp for this backup image is : 20010327162148
```

If the incremental and delta backups are being sent to Tivoli Storage Manager they can be viewed with the `db2adutl` command (only relevant section is shown).

```
Query for database SAMPLE

Retrieving FULL DATABASE BACKUP information.
  1 Time: 20010327170534 Oldest log: S0000099.LOG Node: 0 Sessions: 1

Retrieving INCREMENTAL DATABASE BACKUP information.
  1 Time: 20010327170812 Oldest log: S0000101.LOG Node: 0 Sessions: 1

Retrieving DELTA DATABASE BACKUP information.
  1 Time: 20010327170859 Oldest log: S0000102.LOG Node: 0 Sessions: 1
  2 Time: 20010327170836 Oldest log: S0000102.LOG Node: 0 Sessions: 1
  3 Time: 20010327170736 Oldest log: S0000100.LOG Node: 0 Sessions: 1
  4 Time: 20010327170711 Oldest log: S0000100.LOG Node: 0 Sessions: 1
```

To restore a database the following database backups are required:

- Last full database backup
- Last incremental database backup
- All delta backups since the last incremental backup
- Any logfile before failure

See Figure 173 for an example restore.

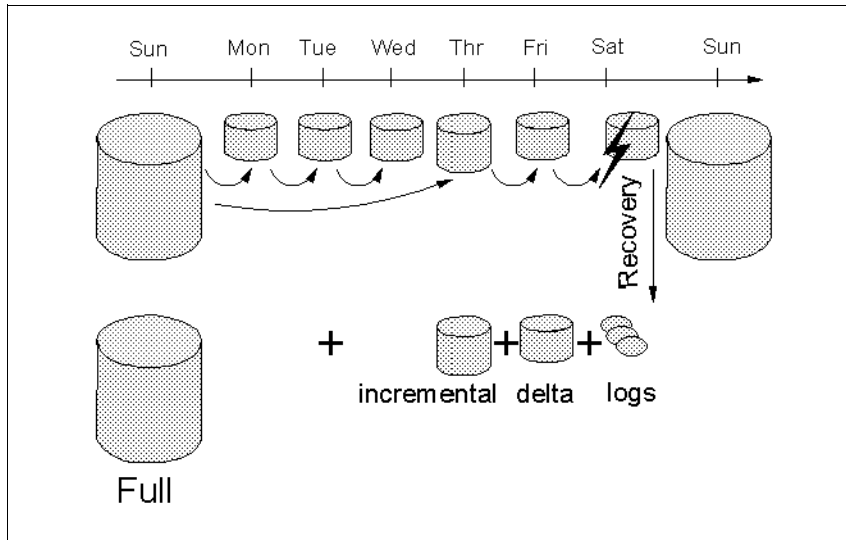


Figure 173. Restore incremental backups

To restore an incremental backup, the `incremental` option on the DB2 backup command must be used. This option is valid for restoring incremental or incremental delta backups.

```

$ db2 restore db sample use tsm taken at 20010327170534
DB20000I The RESTORE DATABASE command completed successfully.
$ db2 restore db sample incremental use tsm taken at 20010327170812
DB20000I The RESTORE DATABASE command completed successfully.
$ db2 restore db sample incremental use tsm taken at 20010327170836
DB20000I The RESTORE DATABASE command completed successfully.
$ db2 restore db sample incremental use tsm taken at 20010327170859
DB20000I The RESTORE DATABASE command completed successfully.
$ db2 rollforward db sample to end of logs and stop

                                Rollforward Status

Input database alias              = sample
Number of nodes have returned status = 1

Node number                       = 0
Rollforward status                 = not pending
Next log file to be read           =
Log files processed                 = S0000102.LOG - S0000106.LOG
Last committed transaction         = 2001-03-27-23.10.15.000000

DB20000I The ROLLFORWARD command completed successfully.

```


Finding all the backups that are necessary to restore to the end of logs can be time consuming. Fortunately, the database stores the information about which database backups has been made in the past in its history file. So the database knows which database backups are necessary to recover to the end of logs.

To make use of the history file during the restore the `automatic` option on the restore command must be used. We recommend you use the `automatic` option of the restore command. The necessary backups are acquired automatically by the restore process.

```
$ db2 restore db sample incremental automatic use tsm
SQL2539W Warning! Restoring to an existing database that is the same as the ba
ckup image database. The database files will be deleted.
Do you want to continue ? (y/n) y
DB20000I The RESTORE DATABASE command completed successfully.
$ db2 rollforward db sample to end of logs and stop

                                Rollforward Status

Input database alias              = sample
Number of nodes have returned status = 1

Node number                       = 0
Rollforward status                 = not pending
Next log file to be read           =
Log files processed                 = S0000102.LOG - S0000106.LOG
Last committed transaction         = 2001-03-27-23.10.15.000000

DB20000I The ROLLFORWARD command completed successfully.
$
```

E.2 On demand log archive

The on demand log archive feature is designed to force a specific database to close its oldest active (used) logfile. This logfile is then available for backup purposes (for example, when the user exit is enabled for archiving).

This feature can also be used in conjunction with a DB2 standby database. All logfiles up to a known time point can be selected on the source database and then be applied at the standby database.

The following is an example of how to use the on demand log archive feature.

```
$ db2 archive log for database sample
DB20000I The ARCHIVE LOG command completed successfully.
$
```

Note

On demand log archiving does not guarantee the log files will be archived immediately; it truncates the log file and issues an archive request, but it is still subject to delays associated with the user exit program.

Appendix F. DB2 backup using TSM LAN-free setup

This appendix shows how we configured DB2 to send backups via the Storage Area Network (SAN).

These are our assumptions:

- The Tivoli Storage Manager server is already configured, for example, libraries, drives, devclasses, stgpools and server-to-server have all been defined.
- The Tivoli Storage Manager client has Fibre Channel card installed, SAN zoning set (if required), client can 'see' the tape drives via the operating system and can perform regular Local Area Network (LAN) backups.

For information on how to get to the point where the tape drives are visible and the Tivoli Storage Manager server is configured, see *Using Tivoli Storage Manager in a SAN Environment*, SG24-6132. This book also discusses the reasons for and benefits of using a SAN to perform backups.

These are the configuration steps to enable DB2 backup using the TSM LAN-free setup:

1. Define new managementclass for LAN-free backups.
2. Download and install storage agent.
3. Modify dsmsta.opt.
4. Check adsm SCSI.
5. Define drive mappings.
6. Define server for storageagent on Tivoli Storage Manager server.
7. Run dsmsta setstorageserver on storage agent.
8. Install storage agent as service.
9. Specify enablelanfree in client options file (dsm.opt).

F.1 Configuration for LAN-free

Support for LAN-free is wholly contained within the Tivoli Storage Manager API. No additional code is needed in any of the products (DB2, TDP for Oracle) that use the Tivoli Storage Manager API. DB2 backups using LAN-free has not been fully tested; therefore, it is not currently supported. If a LAN backup succeeds and a LAN-free fails, you will need to use the LAN backup.

F.1.1 Define new managementclass for LAN-free backups

On the Tivoli Storage Manager server we already have a domain with a managementclass that has the correct policy retention for DB2 backups. So instead of going through the steps to define a new management class, we just copied an existing one; and since it already has the correct retention settings, the only thing we needed to do was update the destination to be the SAN attached storage pool. At the time of writing, LAN-free is only supported when backing up to SAN attached tape devices.

Here is the command we used to copy the management class:

```
tsm: BRAZIL>copy mgmtclass api_domain api_policy api_mgmtclass api_lanfree
ANR1523I Management class API_MGMTCLASS copied to class API_LANFREE in policy
domain API_DOMAIN, set API_POLICY.
```

We then need to update the backup copy group for this new `api_lanfree` management class, so the destination is the SAN attached tape storage pool.

We decided it was not worth a tape mount to send our archive logs using LAN-free, so we left the destination for the archive copygroup to be disk.

```
tsm: BRAZIL>update copygroup api_domain api_policy api_lanfree dest=3570san
ANR1532I Backup copy group STANDARD updated in policy domain API_DOMAIN, set
API_POLICY, management class API_LANFREE.
```

The only thing left to do is activate the policy set so that this new management class can be used.

```
tsm: BRAZIL>activate policy api_domain api_policy

Do you wish to proceed? (Yes (Y)/No (N)) y
ANR1514I Policy set API_POLICY activated in policy domain API_DOMAIN.
```

Now we can check the backup copygroup for this new managementclass in the active policy. We can see that the retention is correct. *Versions exists* is set to one, and the rest of the retention settings are zero. One of the most important things is that the copy destination is to the SAN attached tape storage.

```

tsm: BRAZIL>q copygroup api_domain active api_lanfree f=d

Policy Domain Name: API_DOMAIN
Policy Set Name: ACTIVE
Mgmt Class Name: API_LANFREE
Copy Group Name: STANDARD
Copy Group Type: Backup
Versions Data Exists: 1
Versions Data Deleted: 0
Retain Extra Versions: 0
Retain Only Version: 0
Copy Mode: Modified
Copy Serialization: Shared Dynamic
Copy Frequency: 0
Copy Destination: 3570SAN
Last Update by (administrator): ADMIN
Last Update Date/Time: 03/29/2001 07:05:33
Managing profile:

```

To make sure that our DB2 backups go to this new management class, in the client options file (dsm.opt), we added this include statement.

```
include * api_lanfree
```

F.1.2 Download and install Tivoli Storage Manager Storage Agent

The Tivoli Storage Manager Storage Agent is a key element in doing LAN-free backups. The Tivoli Storage Manager Storage Agent is a scaled down version of the Tivoli Storage Manager server. Instead of the Tivoli Storage Manager API client sending its backups straight to the Tivoli Storage Manager server, as it does in a regular LAN backup, the Tivoli Storage Manager API client sends the backup via NamedPipes to the Tivoli Storage Manager Storage Agent. The Tivoli Storage Manager Storage Agent then sends the backup data directly to the tape storage across the SAN. It also communicates with the Tivoli Storage Manager server using TCP/IP so that the data objects that the Storage Agent sends across the SAN are updated in the Tivoli Storage Manager server's database.

The Storage Agent software for Windows can be downloaded from:

```
ftp://ftp.software.ibm.com/storage/tivoli-storage-management/maintenance/s
torage-agent/
```

The package name that was current at the time of writing was IP22268_StorageAgent.exe. Once downloaded to the client machine, we just double-clicked the executable to install it. We accepted all the defaults and it installed into the directory c:\program files\tivoli\tsm\storageagent.

Ensure that you are correctly licensed for use of the client software.

After installing the software, we rebooted the machine.

F.1.3 Modify dsmsta.opt

In the ...\`storageagent` directory, we opened the `DSMSTA.OPT` file using a text editor and inserted these values. This is the minimum that is needed. These values are for a Windows NT or 2000 environment.

The storage agent is a stripped down version of the Tivoli Storage Manager server, so this `DSMSTA.OPT` file is equivalent to the `DSMSERV.OPT` file. Therefore, the `COMMmethod` value is not selecting the communication method (like a client options file), but rather enabling those two communications method, just like you could do in the `DSMSERV.OPT` on a Tivoli Storage Manager server installation.

The `DEVCONFIG` value specifies the filename to use when running the `DSMSTA SETSTORAGESERVER` command.

The `ENABLEALLCLIENTS` option is an undocumented option which enables clients not yet supported to utilize LAN-free. At this time DB2 backups using LAN-free is not supported.

```
COMMmethod TCPIP  
COMMmethod NAMEDPIPE  
DEVCONFIG devconfig.txt  
ENABLEALLCLIENTS yes
```

F.1.4 Check adsm SCSI

After rebooting, we checked that the Tivoli Storage Manager device driver `AdsmScsi` was running by running the Windows command `net start`.

```
C:\Program Files\Tivoli\TSM\storageagent>net start adsm SCSI  
The requested service has already been started.  
  
More help is available by typing NET HELPMSG 2182.
```

In Windows 2000, the check for `Adsm SCSI` is different than in Windows NT. From Device Manager in Windows 2000, you must change the view to display hidden devices.



Figure 174. Showing hidden devices in Device Manager

After selecting the view to show hidden devices, an entry Non-Plug and Play Drivers will show up, which you can open. When you open this you will see AdsmScsi device.

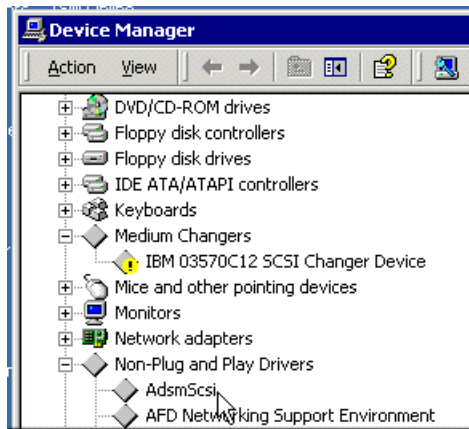


Figure 175. Opening Non-Plug and Play Drivers entry

Right-clicking AdsmScsi and selecting properties presents the next window.

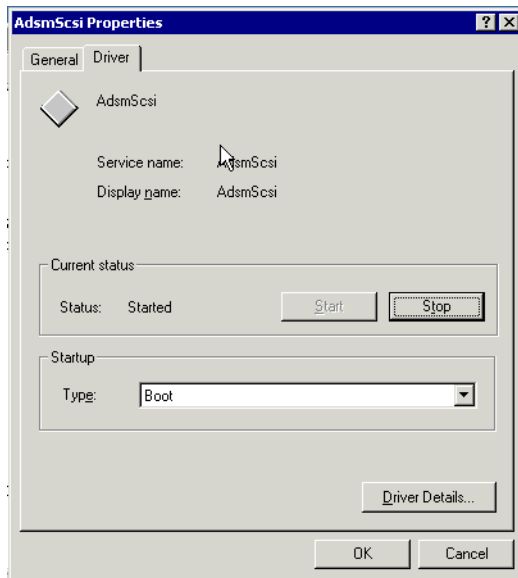


Figure 176. AdsmScsi properties

One confusing thing is the `tmscsi.exe` file located in the `storageagent` directory. This is *not* an updated device driver for use with Tivoli Storage Manager. This is a utility that enables and disables Windows 2000 support with the AdsmScsi device driver. When you use this utility to disable Windows 2000 support, the utility changes the startup mode for AdsmScsi to Demand. When you enable support, it changes the startup mode for AdsmScsi to Boot. When you install the `storageagent`, AdsmScsi defaults to a startup mode of Boot.

F.1.5 Define drive mappings

We then ran the Tivoli Storage Manager executable `TSMDLST.EXE` located in the `...\StorageAgent` directory to determine how the storage agent “sees” the SAN attached tape library. From this output, we see that there are two tape drives.


```
C:\Program Files\Tivoli\TSM\storageagent>tssmdlst
```

```
Computer Name: JAMAICA  
TSM Device Driver: Running
```

TSM Device Name	ID	LUN	Bus	Port	TSM Device Type	Device Identifier
mt0.2.0.3	0	2	0	3	3570	IBM 03570C12 5424
lb0.3.0.3	0	3	0	3	LIBRARY	IBM 03570C12 5424
mt0.4.0.3	0	4	0	3	3570	IBM 03570C12 5424

From the Tivoli Storage Manager server administrative client, we run the command QUERY DRIVE to determine how the Tivoli Storage Manager server “sees” the SAN attached tape library.

```
tssm: BRAZIL>q drive
```

Library Name	Drive Name	Device Type	Device	ON LINE
3570LIB	DRIVE1	3570	/dev/rmt2	Yes
3570LIB	DRIVE2	3570	/dev/rmt4	Yes

In a SAN, hosts access the same storage, but they may show up differently. The drive mapping makes the association between how the storage agent “sees” the tape drives, with how the Tivoli Storage Manager server “sees” the tape drives. Using the output from the TSMDLST.EXE and QUERY DRIVE, we will now define a drive mapping for these two drives.

```
tssm: BRAZIL>define drivemapping jamaica_sta 3570lib drive1 device=mt0.2.0.3  
ANR8916I Drivemapping for drive DRIVE1 in library 3570LIB on storage agent  
JAMAICA_STA defined.
```

```
tssm: BRAZIL>define drivemapping jamaica_sta 3570lib drive2 device=mt0.4.0.3  
ANR8916I Drivemapping for drive DRIVE2 in library 3570LIB on storage agent  
JAMAICA_STA defined.
```

F.1.6 Define server for storage agent on Tivoli Storage Manager server

Using the storage agent name that was specified during the definition of the drive mappings, we now use the command DEFINE SERVER from the Tivoli Storage Manager administrative command line, to configure server to server communications on the Tivoli Storage Manager side. Setting the servername, serverpassword, serverhladdress, and serverlladdress has already been done. The command QUERY STATUS can be used to check if this is the case.

```
tsm: BRAZIL>define server jamaica_sta serverpassword=stapassword hla=193.1.1.85  
lla=1500 comm=tcpip  
ANR1660I Server JAMAICA_STA defined successfully.
```

It is very important that you use the same server name in this command as that used to define the drive mappings.

F.1.7 Run dsmsta setstorageserver on storage agent

From the ...\`storageagent` directory, we need to run the DSMSTA SETSTORAGESERVER command to complete the server to server communications on the storage agent.

The values for MYNAME and MYPASSWORD come from the define server command in the previous section. The other values are for the Tivoli Storage Manager server that we are communicating with. These values can be viewed with a QUERY STATUS.

```
C:\Program Files\Tivoli\TSM\storageagent>dsmsta setstorageserver myname=jamaica  
sta mypassword=stapassword servername=brazil serverpassword=brazil haddress=193  
.1.1.11 lladdress=1500  
  
ANR0900I Processing options file C:\PROGRA  
1\Tivoli\TSM\STORAG  
1\dsmsta.opt.  
ANR7800I DSMSErv generated at 15:55:08 on Feb 28 2001.  
  
Tivoli Storage Manager for Windows NT  
Version 4, Release 1, Level 3.0  
  
Licensed Materials - Property of IBM  
  
5698-TSM (C) Copyright IBM Corporation 1999,2000. All rights reserved.  
U.S. Government Users Restricted Rights - Use, duplication or disclosure  
restricted by GSA ADP Schedule Contract with IBM Corporation.  
  
ANR1432I Updating device configuration information to defined files.  
ANR1433I Device configuration information successfully written to devconfig.txt.  
ANR2119I The SERVERNAME option has been changed in the options file.  
ANR0467I The setstorageserver command completed successfully.
```

F.1.8 Install storage agent as a service

To install the storage agent as a service, we use the INSTALL.EXE executable located in the ...\`storageagent` directory. Do *not* modify the value for the service name. You may modify the location of the `dstsvc.exe` file if you installed the storageagent software to a different directory.

After installing the service, start it with net start.

```
C:\Program Files\Tivoli\TSM\storageagent>install "TSM Storage Agent" "c:\program
files\tivoli\tsm\storageagent\dstasvc.exe"
Service installed

C:\Program Files\Tivoli\TSM\storageagent>net start "TSM Storage Agent"
The TSM Storage Agent service is starting.
The TSM Storage Agent service was started successfully.
```

If you receive an error message that states that the service could not start, but did not return an error, it is most likely caused by specifying a service name other than the one listed above. In this case, use the REMOVE.EXE located in the ...storageagent directory to remove the service, reboot your machine, and repeat the instructions above.

Once the service is started, you can run a QUERY SESSION from the Tivoli Storage Manager server and see the server to server sessions that are *always* running.

F.1.9 Specify enablelanfree in client options file (dsm.opt)

Now that all the previous steps have been completed, the only thing left to do is specify ENABLELANFREE YES in the dsm.opt file. You should have these backups going straight to the SAN attached library pool, by specifying the management class with an include statement. Our dsm.opt file looks like this:

```
commethod tcpip
tcpport 1500
tcpserveraddress 193.1.1.11

nodename jamaica_db2
passwordaccess generate

enablelanfree yes
include * api_lanfree
```

F.2 Running a backup and verifying that LAN-free is working

There is no difference in the backup commands for LAN-free. We chose a simple online backup to verify that LAN-free is working.

```
db2 => backup db sample user db2admin using itsosj online use tsm
```

After executing the command, we connected to the storageagent using a Tivoli Storage Manager administrative client. Change the

TCPSERVERADDRESS for the administrative client to be the IP address of the machine running the storage agent. We had the Tivoli Storage Manager administrative client installed on the same machine as the storage agent, so we just used `localhost` as our IP address. The admin name and password is the same as your Tivoli Storage Manager server name and password.

The query session shows that the LAN-free backup is working. This is indicated by the named pipe session that is sending data to the storage agent. If LAN-free was not working, either the backup would fail or you would see a session on the Tivoli Storage Manager server that was sending the data.

```
C:\Program Files\Tivoli\TSM\baclient>dsmadm -tcps=localhost
Tivoli Storage Manager
Command Line Administrative Interface - Version 4, Release 1, Level 2.12
(C) Copyright IBM Corporation, 1990, 1999, All Rights Reserved.
Enter your user id: admin
Enter your password: *****

Session established with server JAMAICA STA: Windows NT
Server Version 4, Release 1, Level 3.0
Server date/time: 03/30/2001 15:48:45 Last access: 03/30/2001 15:37:43
tsm: JAMAICA_STA>q session
```

Sess Number	Comm. Method	Sess State	Wait Time	Bytes Sent	Bytes Recvd	Sess Type	Platform	Client Name
1	Tcp/Ip	Start	0 S	16.5 K	23.2 K	Server-		JAMAICA_STA
3	Tcp/Ip	Start	0 S	68.0 K	92.0 K	Server-		JAMAICA_STA
7	Tcp/Ip	Start	0 S	128.9 K	150.6 K	Server-		JAMAICA_STA
12	Tcp/Ip	Start	0 S	190.6 K	206.5 K	Server-		JAMAICA_STA
141	Tcp/Ip	Start	0 S	31.1 K	31.9 K	Server-		JAMAICA_STA
167	Named Pipe	RecvW	7 S	349	20.0 M	Node	DB2/NT	JAMAICA_DB2
169	Tcp/Ip	Run	0 S	18.1 K	472	Admin	WinNT	ADMIN

F.3 Problems

After specifying the `enablelanfree` option in the client options file that the db2 backups used, we encountered problems with the `userexit` and `db2adutl`. Each of the programs complained that the option `enablelanfree yes` was not valid for single threaded applications.

F.3.1 Problems with the userexit

After specifying `enablelanfree yes` in the `dsm.opt` file, the `userexit` stopped working. The `dsierror.log` contained the following error:

03/30/2001 15:57:10 ApiInitEx: Error: ENABLELANFREE option not valid for single threaded applications.

The USEREXIT.ERR file contained this error:

```
*****
Time of Error:      Fri Mar 30 15:57:10 2001

Parameter Count:   8
Parameters Passed:
Database name:     SAMPLE
Logfile name:      S0000008.LOG
Logfile path:      C:\DB2\NODE0000\SQL00002\SQLLOGDIR\
Node number:       NODE0000
Operating system:  NT
Release:           SQL07010
Request:           ARCHIVE
Audit Log File:    c:\progra~1\tivoli\tsm\api\ARCHIVE.LOG
System Call Params:
Media Type:        ADMSM
User Exit RC:      16

> Error isolation: dsmInit() returned 109
```

We discovered two methods to work around this problem. Both required modifying the C source code for the user exit and recompiling the user exit to produce a new db2uext2.exe. The end result of both was that the option ENABLELANFREE for the user exit program must be set to no.

In 8.7, “Setting up the DB2 user exit for Tivoli Storage Manager” on page 166, we discussed how to modify the user exit to use a different path for some log files. We can do a similar type of thing to either hardcode the ENABLELANFREE option or to specify that the user exit use a different client options file than what the regular DB2 backups to Tivoli Storage Manager uses. Once the user exit uses a different options file, then it is a simple matter of specifying ENABLELANFREE NO in the user exit’s options file. Otherwise, it will pick up the ENABLELANFREE YES in the client options file.

To facilitate either method, we modified the db2uext2.c to contain two extra string literals: CLIENT_OPTIONS_PATH and ENABLELANFREE_NO.

```

#define BUFFER_SIZE          4096      /* transmit or receive the log    */
                                  /* file in 4k portions           */
#define AUDIT_ACTIVE        1         /* enable audit trail logging     */
#define ERROR_ACTIVE        1         /* enable error trail logging     */
#define AUDIT_ERROR_PATH "c:\\progra~1\\tivoli\\tsm\\api\\" /* path must end with a slash */
#define AUDIT_ERROR_ATTR   "a"       /* append to text file           */
#define CLIENT_OPTIONS_PATH "c:\\progra~1\\tivoli\\tsm\\api\\userexit.opt" /*For lanfree or
redirected restores*/
#define ENABLELANFREE_NO "-enablelanfree=no" /* for lanfree */

```

With these entries in place, we can now choose whether to hardcode ENABLELANFREE NO or use the additional options file method.

F.3.1.1 Hardcoding ENABLELANFREE NO

To hardcode ENABLELANFREE NO in the user exit program, change the call to dsmInit from the initial settings of:

```

adsmRc = dsmInit( &adsmHandle, &adsmApiVer, NULL, NULL,
                 inputParms->adsmPasswd, FILE_SPACE_TYPE,
                 NULL, NULL );

```

And, change it to use the ENABLELANFREE_NO string literal that we added earlier.

```

adsmRc = dsmInit( &adsmHandle, &adsmApiVer, NULL, NULL,
                 inputParms->adsmPasswd, FILE_SPACE_TYPE,
                 NULL, ENABLELANFREE_NO );

```

Then recompile the user exit and place the resulting db2uext2.exe in the ...\\sqlib\\bin path, as described in 8.7, "Setting up the DB2 user exit for Tivoli Storage Manager" on page 166.

F.3.1.2 Using an additional options file

To have the user exit read an additional options file, you need to change the call to dsmInit from the initial settings of:

```

adsmRc = dsmInit( &adsmHandle, &adsmApiVer, NULL, NULL,
                 inputParms->adsmPasswd, FILE_SPACE_TYPE,
                 NULL, NULL );

```

And, change it to use the CLIENT_OPTIONS_PATH string literal that we defined earlier.

```

adsmRc = dsmInit( &adsmHandle, &adsmApiVer, NULL, NULL,
                 inputParms->adsmPasswd, FILE_SPACE_TYPE,
                 CLIENT_OPTIONS_PATH, NULL ) ;

```

Then recompile the user exit and place the resulting db2uext2.exe in the ...\\sqllib\\bin path, as described in 8.7, “Setting up the DB2 user exit for Tivoli Storage Manager” on page 166. With this method you must also create an additional options file that contains, at a minimum, the option ENABLELANFREE NO in the path and with the file name specified in the CLIENT_OPTIONS_PATH string literal. Otherwise, the user exit will fail with RC 406, options file not found.

F.3.2 Problems with the db2adutl.exe

To resolve this problem, you need to have DSMI_CONFIG point to a different options file that does *not* have ENABLELANFREE YES specified. Since DB2 reads the DSMI_CONFIG during instance startup, this will not affect the DB2 backups, *unless* you stop and start the instance (db2stop db2start) with the DSMI_CONFIG set to the other value.

In our environment, because we chose the user exit to use an additional options file, we just set DSMI_CONFIG to use that options file.

We also put the set statement into a batch file (setdb2adutl.bat) and placed it in the ...\\sqllib\\bin directory for easy use. This example shows the error, setting DSMI_CONFIG, and a successful db2adutl query:

```

C:\Program Files\Tivoli\TSM\api>db2adutl query
Error: Initialize session failed with ADSM return code 109
Error: Initialize session failed with ADSM return code 109

C:\Program Files\Tivoli\TSM\api>type dserror.log
04/04/2001 12:27:25 dsminit.cpp      ( 941): ApiInitEx: Error: ENABLELANFREE
option not valid for single threaded applications.

C:\Program Files\Tivoli\TSM\api>setdb2adutl.bat
C:\Program Files\Tivoli\TSM\api>set DSMI_CONFIG=c:\progra~1\tivoli\tsm\api\userex
it.opt

C:\Program Files\Tivoli\TSM\api>db2adutl query

Query for database SAMPLE
Retrieving full database backup information.
full database backup image: 1, Time: 20010331151746
Oldest log: S0000012.LOG, Node: 0, Sessions used: 1

```


Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law. INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.



This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

AIX®	FlashCopy®	Redbooks™
Domino®	Informix®	Redbooks (logo)  ™
DB2 Universal Database™	IBM®	RS/6000®
DB2®	Lotus Notes®	S/390®
Enterprise Storage Server®	Lotus®	Tivoli®
@server®	Notes®	
 server®	pSeries®	

The following terms are trademarks of other companies:

JDBC, Solaris, Sun, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Visual C++, Windows NT, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

i386, Intel logo, Intel Inside logo, and Intel Centrino logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

Appendix G. Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

G.1 IBM Redbooks

For information on ordering these publications see “How to get IBM Redbooks” on page 325.

- *DB2 UDB V7.1 Performance Tuning Guide*, SG24-6012
- *Using ADSM to Backup Databases*, SG24-4335
- *AIX Version 4.3 Differences Guide*, SG24-2014
- *Implementing ESS Copy Services on UNIX and Windows NT/2000*, SG24-5757
- *Tivoli Storage Management Concepts*, SG24-4877
- *Using Tivoli Storage Manager in a SAN Environment*, SG24-6132

G.2 IBM Redbooks collections

Redbooks are also available on the following CD-ROMs. Click the CD-ROMs button at ibm.com/redbooks for information about all the CD-ROMs offered, updates and formats.

CD-ROM Title	Collection Kit Number
IBM System/390 Redbooks Collection	SK2T-2177
IBM Networking Redbooks Collection	SK2T-6022
IBM Transaction Processing and Data Management Redbooks Collection	SK2T-8038
IBM Lotus Redbooks Collection	SK2T-8039
Tivoli Redbooks Collection	SK2T-8044
IBM AS/400 Redbooks Collection	SK2T-2849
IBM Netfinity Hardware and Software Redbooks Collection	SK2T-8046
IBM RS/6000 Redbooks Collection	SK2T-8043
IBM Application Development Redbooks Collection	SK2T-8037
IBM Enterprise Storage and Systems Management Solutions	SK3T-3694

G.3 Other resources

These publications are also relevant as further information sources:

- *Tivoli Storage Manager Using the Application Program Interface V4R1*, SH26-4123

- *Tivoli Storage Manager for Windows Administrator's Guide*, GC35-0410
- *Tivoli Storage Manager for Windows Administrator's Reference*, GC35-0411
- *Tivoli Storage Manager for AIX Administrator's Guide*, GC35-0403
- *Tivoli Storage Manager Installing the Clients V4R1*, SH26-4119
- *DB2 UDB Administration Guide: Implementation*, SC09-2944
- *DB2 UDB Administration Guide: Planning*, SC09-2946
- *DB2 UDB Command Reference V7*, SC09-2951
- *DB2 UDB Troubleshooting Guide V7*, GC09- 2850
- *Using DB2 Universal Database on 64-bit Platforms*, which can be found at:

http://www-4.ibm.com/cgi-bin/db2www/data/db2/udb/winos2unix/support/v7pubs.d2w/en_main

Scroll down the previously mentioned Web page to find the publication within the Administration section.

G.4 Referenced Web sites

These Web sites are also relevant as further information sources:

- http://www.tivoli.com/support/storage_mgr/requirements.html
Tivoli Storage Manager Product Requirements and Supported Devices
- http://www.tivoli.com/support/public/Prodman/public_manuals/storage_mgr/admanual.html
Tivoli Storage Manager Product Publications and Documentation
- <http://www.storage.ibm.com/hardsoft/products/ess/supserver>
IBM ESS Open Systems Support
- <http://www.storage.ibm.com/snetwork/index.html>
Storage Networking (SAN/NAS/iSCSI)
- <http://www-1.ibm.com/servers/aix/products/ibmsw/list/>
IBM application availability guide

How to get IBM Redbooks

This section explains how both customers and IBM employees can find out about IBM Redbooks, redpieces, and CD-ROMs. A form for ordering books and CD-ROMs by fax or e-mail is also provided.

- **Redbooks Web Site** ibm.com/redbooks

Search for, view, download, or order hardcopy/CD-ROM Redbooks from the Redbooks Web site. Also read redpieces and download additional materials (code samples or diskette/CD-ROM images) from this Redbooks site.

Redpieces are Redbooks in progress; not all Redbooks become redpieces and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

- **E-mail Orders**

Send orders by e-mail including information from the IBM Redbooks fax order form to:

	e-mail address
In United States or Canada	pubscan@us.ibm.com
Outside North America	Contact information is in the "How to Order" section at this site: http://www.elink.ibm.com/pbl/pbl

- **Telephone Orders**

United States (toll free)	1-800-879-2755
Canada (toll free)	1-800-IBM-4YOU
Outside North America	Country coordinator phone number is in the "How to Order" section at this site: http://www.elink.ibm.com/pbl/pbl

- **Fax Orders**

United States (toll free)	1-800-445-9269
Canada	1-403-267-4455
Outside North America	Fax phone number is in the "How to Order" section at this site: http://www.elink.ibm.com/pbl/pbl

This information was current at the time of publication, but is continually subject to change. The latest information may be found at the Redbooks Web site.

IBM Intranet for Employees

IBM employees may register for information on workshops, residencies, and Redbooks by accessing the IBM Intranet Web site at <http://w3.itso.ibm.com/> and clicking the ITSO Mailing List button. Look in the Materials repository for workshops, presentations, papers, and Web pages developed and written by the ITSO technical professionals; click the Additional Materials button. Employees may access MyNews at <http://w3.ibm.com/> for redbook, residency, and workshop announcements.

Index

A

- active logs 19
- AIX splitcopy 296
- API 156
 - configuring 156
 - Tivoli Storage Manager 4
- API client
 - installation 34
 - setup 63
- Application Program Interface (API) 33
- archive 71
- archive logs 19
- archive object 35, 37
- ARCHIVE.LOG 281

B

- backup
 - automation 134, 195
 - deletion 127, 131, 193
 - offline backup 101
 - online backup 107
 - verification 124
- backup features
 - DB2 V7.1 Fixpak3 Beta 301
- backup object 36
 - automated deletion 133
 - deletion of 131
- backup strategy 25
- backup techniques 26
- backup windows 24
- Backup-Archive (baclient) 91

C

- client options file 161
- configuration
 - DSMI_CONFIG 65
 - DSMI_DIR 65
 - DSMI_LOG 65
- configuration parameters
 - TSM_MGMTCLASS 169
 - TSM_NODENAME 169
 - TSM_OWNER 169
 - TSM_PASSWORD 169
- containers 17, 260
- Control Center 75, 79, 102, 112

- control files 13
- copygroup 40
- cron 134

D

- data dictionary 8
- data objects 35
 - life cycle 43
- database 7
 - recovery 213
- database administrator (DBA) 19
- database export 28
- database managed space (DMS) 18
- database partitions 16
- DB2
 - user exit 67, 97, 166
- DB2 instance 246
 - restoring to new 246
- DB2 list history 125
- DB2 UDB 7
 - concepts 13
- db2adutl 183
 - db2adutl.exe query 165
 - EXTRACT 120
 - QUERY 120
 - restore 246
 - utility 119
 - verify 127
- db2adutl utility 119
- db2adutl.exe
 - DATABASE 185
 - DELETE 184
 - EXTRACT 184
 - KEEP 185
 - OLDER THAN 185
 - QUERY 184
 - REONLY 184
 - SHOW INACTIVE 185
 - TAKEN AT 185
 - VERDELETED 184
 - VERIFY 185
 - WITHOUT PROMPTING 185
- db2ckbkp 124
- db2inidb 291
- db2nodes.cfg 16
- db2rhist.asc 125
- db2uext2 130

delete qualifier
 KEEP n 131
 OLDER THAN 131
 TAKEN AT 131
deletion 131, 193
delta 301
disk mirroring 26
DRM 4
dsierror.log 160, 281
dsm.opt 35
dsmapiw.exe 164
dsmsched.log 283

E

Enterprise Storage Server (ESS) 31, 289
environment variable 64, 94, 156, 157
 DSMI_CONFIG 158, 278
 DSMI_DIR 159
 DSMI_LOG 160

F

FlashCopy 290
full database backup 28
full offline backup 171
full online backup 175

H

history file 125, 189
 recovery 265

I

include-exclude 50, 72
incremental backup 29
 delta 301
 full 301
 incremental 301
 TRACKMOD 302
incremental delta 303
indexes 8
instance 16

J

journaled filesystem (JFS) 296

L

LAN-free backup 30, 307

list history 125, 189
load utility 88, 117
log file backup 30
log files 11, 18, 131, 193, 281
 deletion of 131
logical volume manager (LVM) 296

M

management class 39, 170
 default 39
 LAN-free 308
MAXNUMMP 286

N

node name
 choosing 53
 registering 149
nodegroups 17, 38
non-database files 25

O

offline backup 27, 73, 74, 101, 171
 command line 101
 DB2 Control Center 102
on demand log archive 305
online backup 27
options file 35, 161

P

partial database backup 29
PASSWORDACCESS 164
performance 285
performance options
 BUFFER buff-size 286, 288
 OPEN n SESSIONS 286, 288
 PARALLELISM n 286, 288
 WITH num-buff BUFFERS 286, 288
planning considerations 21
point in time recovery 235
policy domain 39
policysset 39, 55, 58
 activation 39
 active policysset 39

Q

quick start 271

R

- RDBMS 22
- recovering 265
- recovery 218
 - point in time 235
 - roll-forward 218
 - tablespace 227
- recovery history file 19
- recovery points 25
- redirected restore 254
- Redundant Array of Inexpensive Disk (RAID) 21
- Relational Database Management System (RDBMS) 7
- restore 251
 - redirected 254
 - to new instance 246
- roll-forward recovery 97, 98, 218
 - AIX 69
 - SUN 98
 - tablespace 224

S

- SELECT command 121
 - list of backups 122
- simulated incremental 30
- snapshot 291
- split copy 289, 296
- split mirror 31, 289
 - clone database 292
 - standby database 293
 - target volumes 289
- split mirror feature 31
- stale mirrored state 298
- standby database 293, 305
- storage agent 307, 309, 310, 312, 313
- Storage Area Network (SAN) 3
- storage pool 41, 55
- Sun Solaris 91
- system managed space (SMS) 18

T

- tables
 - definition 7
- tablespace backup 178
- tablespaces 9, 17
 - database managed space (DMS) 17
 - point in time recovery 237
 - roll-forward recovery 224

- system managed space (SMS) 17
- target volumes 289
- TDP application clients 4
- Tivoli Data Protection (TDP) 4
- Tivoli Data Protection for applications 4
- Tivoli Disaster Recovery Manager 4
- Tivoli Space Manager 4
- Tivoli Storage Management 3
- Tivoli Storage Manager
 - administrator 3
 - API 4
 - backup/archive client 5
 - copygroup 40
 - data objects 35
 - database backup and restore 5
 - DRM 4
 - expiration 44
 - HSM 4
 - include/exclude file 72
 - server 3
- Tivoli Storage Manager (TSM) 19
- Tivoli Storage Manager Backup-Archive client 4
- Tivoli Storage Manager server
 - DEFINE ASSOCIATION 196
 - DEFINE SCHEDULE 196
- trace 278
 - TSM CONFIG 278
- transaction logs 14

U

- units of recovery 25
- user exit 67
 - automatic error notification 129
 - return codes 282

V

- verification of DB2 backups 124
 - db2adutl 126
 - db2ckbkp 124
- version recovery 214

IBM Redbooks review

Your feedback is valued by the Redbook authors. In particular we are interested in situations where a Redbook "made the difference" in a task or problem you encountered. Using one of the following methods, **please review the Redbook, addressing value, subject matter, structure, depth and quality as appropriate.**

- Use the online **Contact us** review redbook form found at ibm.com/redbooks
- Fax this form to: USA International Access Code + 1 845 432 8264
- Send your comments in an Internet note to redbook@us.ibm.com

Document Number	SG24-6247-00
Redbook Title	Backing up DB2 with IBM Tivoli Storage Management
Review	
What other subjects would you like to see IBM Redbooks address?	
Please rate your overall satisfaction:	<input type="radio"/> Very Good <input type="radio"/> Good <input type="radio"/> Average <input type="radio"/> Poor
Please identify yourself as belonging to one of the following groups:	<input type="radio"/> Customer <input type="radio"/> Business Partner <input type="radio"/> Solution Developer <input type="radio"/> IBM, Lotus or Tivoli Employee <input type="radio"/> None of the above
Your email address: The data you provide here may be used to provide you with information from IBM or our business partners about our products, services or activities.	
Questions about IBM's privacy policy?	The following link explains how we protect your personal information. ibm.com/privacy/yourprivacy/



Redbooks

Backing up DB2 with IBM Tivoli Storage Management

(0.5" spine)

0.475" <-> 0.875"

250 <-> 459 pages



Backing up DB2 with IBM Tivoli Storage Management



Redbooks

Covers all aspects of protecting DB2 databases on multiple platforms

Backup, restore and disaster recovery

Practical scenarios and how-tos'

This IBM Redbook covers techniques and practical guidelines for backing up DB2 UDB using Tivoli Storage Manager. It is intended for database administrators (DBAs) and system/storage administrators and anyone who needs to protect their critical DB2 databases. We focus on the use of the Tivoli Storage Manager API client and provide installation, setup, customization and day-to-day management examples.

All testing was carried out on DB2 V7.1 and with Tivoli Storage Manager V4.1.2. The platforms used were AIX, Sun Solaris and Windows 2000. The design of the project includes recovery scenarios, as well as, different backup methodologies in order to provide practical assistance for DBAs.

There are hints and tips for performance and troubleshooting. Coverage has also been given to enhanced strategies for backup including hardware mirroring.

**INTERNATIONAL
TECHNICAL
SUPPORT
ORGANIZATION**

**BUILDING TECHNICAL
INFORMATION BASED ON
PRACTICAL EXPERIENCE**

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks

SG24-6247-00

ISBN