

Tips, Tricks & Recipes for Sybase ASE

2nd edition

Rob Verschoor

Sypron Publications

ISBN 90-806117-2-7

More information about this book is available at
www.sypron.nl/ttr

© 2003-2006 Sypron B.V.

TABLE OF CONTENTS

INTRODUCTION	xiii
TERMINOLOGY AND ABBREVIATIONS	xvii
UNDOCUMENTED/UNSUPPORTED FEATURES: WARNING & DISCLAIMER.....	xviii
INDEX OF DOs AND DON'Ts	xix
1 VERSIONS AND FLAVOURS OF ASE	1
1.1 Historical overview of ASE versions	1
1.2 The ASE version string	5
1.2.1 ASE version terminology: EBF, ESD, SWR, GA, IR?.....	6
1.2.2 Major and minor ASE versions.....	8
1.2.3 Hierarchy of ASE release types.....	8
1.2.4 Exactly what is an EBF?.....	9
1.3 Checking the ASE version from T-SQL.....	11
1.3.1 Check for major ASE versions: system table existence	12
1.3.2 Checking for minor ASE versions	15
1.3.3 The ASE version in @@version	16
1.3.4 The ASE version in @@version_number.....	17
1.3.5 The ASE version in @@version_as_integer.....	18
1.3.6 Configuration parameter 'upgrade version'	18
1.3.7 Checking for presence of built-in function/variables	19
1.3.8 Checking for 32-bit or 64-bit ASE.....	22
1.4 ASE version check without starting ASE	22
1.4.1 Beware of the wrong PATH... ..	23
1.5 Flavours of ASE	24
1.5.1 Obtaining a free version of ASE	26
2 ASE-RELATED FILES	27
2.1 Introduction.....	27
2.2 Locating the RUN_server file	28
2.3 The ASE errorlog file	29
2.3.1 Determining the errorlog pathname	30
2.3.2 Truncating the errorlog while ASE is running	31
2.3.3 Writing custom messages to the errorlog	32
2.3.4 Reading your own errorlog from ASE.....	36
2.3.5 Using CIS file access (ASE_XFS) to access the errorlog	37
2.3.6 Determining the OS version of the ASE host.....	38
2.4 Forcing ASE error messages into the errorlog.....	38
2.4.1 Setting the 'with_log' option for an ASE error	39
2.5 Who's that spid?.....	42
2.6 The ASE 'console'	44
2.7 The NT event log	47
2.8 The interfaces (SQL.INI) file.....	48
2.8.1 Determining the 'interfaces' file pathname	48
2.8.2 Other connectivity-related information.....	49
2.9 Determining the master device pathname	49

3	MANY WAYS TOWARDS...THE END	51
3.1	Terminating others with T-SQL 'kill'	52
3.1.1	<i>Please commit suicide now! - a polite kind of 'kill'</i>	52
3.1.2	<i>Internals: how a killed process terminates itself</i>	54
3.1.3	<i>Don't kill the wrong process</i>	56
3.1.4	<i>Limitations of 'kill' and workarounds</i>	57
3.1.5	<i>sp_kill_all: killing many processes in one command</i>	58
3.2	When a process cannot be killed	60
3.2.1	<i>The killed process is rolling back a long transaction</i>	61
3.2.2	<i>The killed process' spid number has been re-used</i>	62
3.2.3	<i>The killed process 'hangs'</i>	63
3.2.4	<i>syb_terminate: the secret super-kill command?</i>	64
3.3	Terminating yourself	65
3.3.1	<i>Terminating your own connection with syb_quit()</i>	65
3.3.2	<i>set background: syb_quit() for ASE pre-12.0</i>	68
3.3.3	<i>Killing yourself with the T-SQL kill statement</i>	70
3.3.4	<i>Why 'quit' & 'exit' in isql aren't useful</i>	70
3.4	Aborting other users' transactions.....	71
3.4.1	<i>LOG SUSPEND: abort, kill, or enlarge?</i>	72
3.5	Shutting down ASE	76
3.5.1	<i>T-SQL shutdown commands</i>	77
3.5.2	<i>OS-level kill commands</i>	81
3.5.3	<i>Shutdown by 'other causes'</i>	85
3.6	Reconstructing the startup/shutdown history	85
3.7	Disappearing clients.....	89
3.7.1	<i>How KEEPALIVE helps to kill orphan queries</i>	91
4	HANDLING DUPLICATE DATA	93
4.1	Identifying duplicate keys in a single table	94
4.1.1	<i>Avoiding (instead of detecting) duplicate keys in a table</i> ..	95
4.1.2	<i>Case 1: No unique row identifier</i>	95
4.1.3	<i>Case 2: Unique row identifier, index on key column(s)</i>	97
4.1.4	<i>The lazy solution: just create a unique index</i>	99
4.1.5	<i>Performance comparison</i>	99
4.2	Removing duplicate keys from an individual table.....	101
4.2.1	<i>Method 1: deleting duplicates directly with 'delete'</i>	102
4.2.2	<i>Method 2: removing duplicates with 'ignore_dup_key'</i>	104
4.2.3	<i>Performance comparison</i>	105
4.3	Identifying duplicate keys between multiple tables	105
4.4	Merging two tables	106
4.5	Handling duplicate rows.....	107
4.5.1	<i>Exactly when are rows (not) considered duplicates?</i>	108
4.5.2	<i>Methods for handling duplicate rows</i>	109
4.5.3	<i>Removing duplicate rows</i>	109
4.5.4	<i>Avoiding duplicate rows in DOL tables</i>	110
4.6	What if a table is too big?	110
5	TRICKS WITH 'ignore_dup_key'.....	113
5.1	How ignore_dup_key works	113
5.2	Dynamically changing ignore_dup_key	118
5.2.1	<i>Toggling ignore_dup_key</i>	119
5.2.2	<i>Setting ignore_dup_key for constraints</i>	122

5.2.3	<i>Toggleing other index properties? (No!)</i>	123
5.3	Intercepting duplicate keys in an insert trigger.....	124
5.3.1	<i>Intercepting duplicates in single-row inserts</i>	124
5.3.2	<i>Intercepting duplicates in multiple-row inserts</i>	127
5.3.3	<i>Internals: why rejected rows occur in an insert trigger</i>	129
5.4	How index order can affect insert performance.....	131
6	FINDING MISSING DATA	135
6.1	Data is missing - but how do we know?.....	135
6.2	Queries for the 'set difference' of two tables	136
6.2.1	<i>Simplest method: 'not in'/'not exists' subqueries</i>	137
6.2.2	<i>Classic T-SQL outer join (*=, =*)</i>	138
6.2.3	<i>Fastest: ANSI outer join (12.0+)</i>	140
6.2.4	<i>Differences between ANSI / T-SQL outer joins</i>	141
6.2.5	<i>Combining two comparisons ('full outer join')</i>	142
6.2.6	<i>Performance comparison</i>	143
6.2.7	<i>Avoiding missing data: use constraints</i>	144
6.3	Queries for missing sequential data (single table).....	145
6.3.1	<i>Slow: the classic self-join</i>	145
6.3.2	<i>Fastest: ANSI outer join (12.0+)</i>	146
6.3.3	<i>Update-with-variables (also finds duplicates)</i>	148
6.3.4	<i>Performance comparison</i>	150
6.3.5	<i>What about an old-fashioned loop?</i>	151
6.4	Find missing data with checksums (hash totals)	151
6.4.1	<i>Understanding checksum comparisons</i>	152
6.4.2	<i>Example of using a checksum in T-SQL</i>	153
6.4.3	<i>Using checksums to verify data conversion correctness</i> ...	155
6.4.4	<i>Using checksums to verify replicated data</i>	155
6.4.5	<i>Using checksums to verify schema completeness</i>	156
7	RECIPES FOR SEQUENTIAL PRIMARY KEYS.....	159
7.1	Introduction: sequential keys	159
7.1.1	<i>Overview of key generation algorithms</i>	161
7.2	Simple but unreliable: the 'max+1' algorithm	162
7.2.1	<i>Why 'max+1' is not suitable for primary keys</i>	163
7.2.2	<i>Failed attempts to improve 'max+1'</i>	164
7.2.3	<i>Conclusion: avoid 'max+1'</i>	166
7.3	Commonly used: the 'key table' algorithm.....	166
7.3.1	<i>Improvements to the 'key table' algorithm</i>	169
7.3.2	<i>Practical example: multiple key counters in one table</i>	172
7.4	The 'key trigger' algorithm: nice, but slow.....	176
7.5	The fastest algorithm: the identity column	178
7.5.1	<i>An identity column as a stand-alone key counter</i>	181
7.6	Including a check digit in a primary key	182
7.7	Deferred generation of sequential keys	186
7.8	Generating unique keys in replication systems	187
7.9	Tracking gaps in the key sequence	188
7.10	Primary keys: clustered or nonclustered indexes?.....	190
7.11	Choosing a lock scheme.....	191
8	RECIPES FOR NON-SEQUENTIAL PRIMARY KEYS	195
8.1	Introduction: non-sequential keys.....	195

8.1.1	Overview of key generation algorithms.....	195
8.2	(Pseudo-)random keys	196
8.3	Keys derived from the system clock time	201
8.4	Worldwide uniqueness: GUIDs/UUIDs & newid()	205
8.4.1	Different behaviour of newid() in 12.5.0.3.....	208
8.5	Keys based on @@dbts	210
8.5.1	Introducing the ASE 'database timestamp'	210
8.5.2	Generating unique keys from @@dbts	213
8.5.3	Internals: details about the database timestamp.....	216
8.6	Key generation by the client application	220
8.6.1	Is an IP address always unique?	221
8.7	When duplicate keys occur... ..	221
8.7.1	Duplicate key: error 2601.....	222
9	TRICKS WITH THE 'UPDATE' STATEMENT	225
9.1	Selecting with 'update': 'update-with-variables'.....	225
9.2	How it works: the details	228
9.3	Examples of update-with-variables.....	230
9.3.1	Combining statements: select+update=update.....	230
1.1.2	Avoiding loops	231
1.1.3	Concatenating columns	231
1.1.4	Parsing a string into different rows, columns or variables	232
1.1.5	Assigning sequential numbers to subsets of rows	233
1.4	Case study: monthly customer retention report.....	234
10	EXECUTE-IMMEDIATE (DYNAMIC SQL).....	239
10.1	A new dimension to T-SQL.....	239
10.1.1	How it works	239
10.1.2	Separate execution context for execute-immediate	240
10.1.3	Passing data to/from execute-immediate	241
10.1.4	Limitations of execute-immediate	243
10.1.5	Detecting execute-immediate.....	244
10.2	Applications of execute-immediate	246
10.2.1	Putting 'union views' to work.....	246
10.2.2	Optionally selecting to the client or into a table.....	248
10.2.3	Drop & recreate an object in the same batch/procedure..	248
10.2.4	Query optimization and local variables.....	251
10.2.5	Fixing classic optimizer problems	252
10.2.6	Data archiving/restoring application	254
10.2.7	Miscellaneous applications of execute-immediate	258
10.3	Simulating dynamic SQL in pre-12.0.....	259
11	EVERYTHING YOU NEVER WANTED TO KNOW ABOUT IDENTITY COLUMNS .	261
11.1	Identity columns are great.....	261
11.1.1	Usage guidelines for identity columns.....	263
11.1.2	What to do if you have an identity gap	263
11.1.3	Overview of identity-related features.....	264
11.1.4	Similar features by other RDBMS vendors	265
11.2	How to use identity columns	266
11.2.1	Applications of identity columns.....	270
11.3	What is an identity gap?	271
11.4	Internals: why identity gaps occur	272

11.4.1	<i>A day in the life of an identity counter</i>	272
11.4.2	<i>Why do we need a 'max-burned value' at all?</i>	274
11.4.3	<i>Identity gaps due to shutdown with nowait</i>	274
11.4.4	<i>Identity gaps due to dump & load</i>	276
11.4.5	<i>Missing individual identity values</i>	278
11.4.6	<i>Missing identity values due to 'identity grab size'</i>	280
11.4.7	<i>Identity counter in sysobjects.identburnmax (15.0)</i>	281
11.5	Preventing/protecting against identity gaps	282
11.5.1	<i>Always define 'identity_gap'! (12.0+)</i>	282
11.5.2	<i>The 'identity burning set factor'</i>	284
11.5.3	<i>Internals: identity counter re-synch, DES & disk</i>	285
11.5.4	<i>Avoid identity gaps due to 'shutdown with nowait'</i>	286
11.5.5	<i>Avoid identity gaps due to dump & load</i>	287
11.6	Repairing identity gaps	289
11.6.1	<i>The classic identity gap repair procedure(pre-12.5.1)</i>	289
11.6.2	<i>Automatically repairing identity gaps in 12.5.1+</i>	290
11.6.3	<i>Resetting the identity counter upwards</i>	291
11.6.4	<i>Resetting the identity counter downwards</i>	292
11.7	Updating existing identity column values	294
11.7.1	<i>Updating identity columns in pre-12.5.0.3</i>	294
11.7.2	<i>More tricks with identity columns</i>	299
11.7.3	<i>Updating identity columns in 12.5.0.3+</i>	301
11.8	Useful identity-related queries & commands	302
11.8.1	<i>Identity-related T-SQL queries</i>	302
11.8.2	<i>Identity-related 'dbcc' commands</i>	305
12	TRANSACTIONS, PROCESSES, AND YOU	309
12.1	Transaction programming tips	309
12.2	Handling a full transaction log	313
12.2.1	<i>Internals: why a full transaction log isn't really full</i>	314
12.2.2	<i>Expanding a full transaction log?</i>	315
12.2.3	<i>Automatically aborting suspended transactions</i>	316
12.2.4	<i>Automatically expanding a full database?</i>	317
12.3	When tempdb is full	318
12.3.1	<i>What's fake about 'fake' tables?</i>	318
12.3.2	<i>Full tempdb: fake tables cannot be read</i>	319
12.3.3	<i>Tempdb full? Use another one! (12.5.0.3+)</i>	320
12.3.4	<i>Limiting tempdb space usage</i>	321
12.4	What's going on inside that process?	321
12.4.1	<i>Determining currently executing T-SQL statements</i>	322
12.4.2	<i>Currently executing procedure: sysprocesses.id</i>	324
12.4.3	<i>Information in transaction names</i>	325
12.4.4	<i>Disk I/O activity: sysprocesses.physical_io</i>	326
12.4.5	<i>Processing activity: sysprocesses.cpu</i>	327
12.4.6	<i>Identifying the client corresponding to a spid</i>	327
12.4.7	<i>Original and actual login name</i>	329
12.4.8	<i>Identifying CIS connections</i>	330
12.4.9	<i>Information from the MDA tables (12.5.0.3+)</i>	331
12.5	Is a process currently performing a rollback?	332
12.5.1	<i>Internals: clues in the stacktrace</i>	334
12.5.2	<i>Measuring performance from the past</i>	336
12.5.3	<i>Internals: more about dbcc pss</i>	337

12.6	Can the duration of a rollback be predicted?	338
	12.6.1 <i>Showing rollback progress: kill..with statusonly(12.5.2+)</i>	339
	12.6.2 <i>Guesstimating rollback progress (pre-12.5.2)</i>	341
12.7	Speeding up rollbacks	342
	12.7.1 <i>Cache configuration and run-time rollbacks</i>	342
	12.7.2 <i>Cache configuration and recovery-time rollbacks</i>	342
	12.7.3 <i>Speeding up rollbacks with a shutdown/restart?</i>	344
	12.7.4 <i>Recovery vs. normal processing</i>	344
12.8	Non-transactional DML	345
	12.8.1 <i>Classic site handler RPCs</i>	345
	12.8.2 <i>Transactional CIS RPCs (11.5+)</i>	347
	12.8.3 <i>External files (ASE_XFS) in 12.5+</i>	348
	12.8.4 <i>Simulating non-transactional DML</i>	349
13	ASSORTED ASE INTERNALS	351
13.1	The database timestamp and database recovery	351
13.2	Why 'dbcc rebuild_log' is dangerous	354
13.3	Timestamp columns	356
13.4	Determining the current transaction ID for a spid	360
13.5	About index IDs	362
13.6	Why spid numbers are re-used	364
13.7	The 'kpid' (kernel process ID)	365
	13.7.1 <i>Applications of the kpid</i>	367
13.8	The 'set background' command	368
	13.8.1 <i>Effect of 'set background' on T-SQL output</i>	369
	13.8.2 <i>Detecting background mode</i>	371
	13.8.3 <i>Required permissions for 'set background'</i>	371
	13.8.4 <i>Effect when 'set background off' is missing</i>	371
	13.8.5 <i>Peculiarities of 'set background'</i>	372
	13.8.6 <i>Recommendations for using 'set background'</i>	374
13.9	Finding the data/index page for a row	375
	13.9.1 <i>Locating data pages and index pages for APL tables</i>	375
	13.9.2 <i>Locating data pages for DOL tables</i>	377
	13.9.3 <i>Locating index pages for DOL tables</i>	378
	13.9.4 <i>Traceflag 1212/1217; troubleshooting corrupted tables</i>	380
13.10	Uniquely identifying a server: @@nodeid	382
13.11	Session-specific settings in stored procedures	384
14	MISCELLANEOUS TOOLS & TRICKS	389
14.1	sp_repeat: boring work is for computers!	389
14.2	Collectively locking/unlocking all logins	392
14.3	Executing sp_* procedures across databases	392
14.4	Storing binary files in ASE	393
14.5	Sending messages with syb_sendmsg()	396
14.6	Implementing application locks	399
14.7	Quickly counting I/O usage by SQL batches	402
14.8	Selected tricks with strings in T-SQL	403
	14.8.1 <i>The empty (and the not-so-empty) string</i>	403
	14.8.2 <i>Left-padding a number with zeroes</i>	404
	14.8.3 <i>Replacing strings with str_replace() (12.5.0.3/12.5.1+)</i>	404
	14.8.4 <i>Finding the number of substring occurrences</i>	405
	14.8.5 <i>Concatenating strings across multiple rows</i>	406

15 GETTING STARTED WITH QUERY PROCESSING IN ASE 15	409
15.1 What's the big deal with ASE 15.0?	409
15.1.1 <i>D-I-Y ASE 15.0 performance test: convince yourself</i>	411
15.1.2 <i>Which queries gain most from ASE 15.0?</i>	413
15.1.3 <i>Internals: what is the query execution engine, anyway?</i> ..	414
15.1.4 <i>Why redesign query processing in ASE 15.0?</i>	415
15.2 Guidelines for using ASE 15.0 query processing	415
15.2.1 <i>Procedure cache usage in ASE 15.0</i>	416
15.2.2 <i>Partition descriptors in ASE 15.0</i>	419
15.2.3 <i>Generating better statistics</i>	420
15.2.4 <i>How can you tell if better statistics are needed?</i>	422
15.2.5 <i>Identifying missing statistics with 'show_missing_stats'</i> ..	423
15.2.6 <i>Leave parallel processing initially disabled in ASE 15.0</i>	424
15.3 Optimization goals	425
15.3.1 <i>Overview of existing optimization goals</i>	426
15.3.2 <i>Query optimization: a delicate balancing act</i>	427
15.3.3 <i>Setting the optimization goal - the standard way</i>	428
15.3.4 <i>Setting the optimization goal in a login trigger</i>	429
15.3.5 <i>Fine-tuning optimization goals</i>	431
15.3.6 <i>Scope of 'set plan optgoal'</i>	433
15.3.7 <i>Scope of 'set merge_join', 'set hash_join', (etc.)</i>	437
15.3.8 <i>Choosing the right optimization goal in practice</i>	437
15.3.9 <i>Measuring query optimization time</i>	438
15.4 Advanced query processing options	439
15.4.1 <i>Optimization timeout</i>	440
15.4.2 <i>Repartitioning</i>	440
15.4.3 <i>Resource granularity</i>	440
15.5 What if a query runs slower in ASE 15.0?	441
15.6 Internals: overview of join types.....	442
15.6.1 <i>What is a nested-loop join?</i>	442
15.6.2 <i>What is an N-ary nested-loop join?</i>	443
15.6.3 <i>What is a merge join?</i>	445
15.6.4 <i>What is a hash join?</i>	447
15.7 Some philosophical thoughts for ASE 15 DBAs	448
15.8 Brief overview of new ASE 15.0 features	450
APPENDIX A RECENT CHANGES THAT MAY AFFECT APPLICATIONS	453
A.1 Global variables allowed in defaults, constraints?.....	453
A.2 Concatenating columns with 'select'	455
APPENDIX B DIAGNOSTICS FOR SYBASE TECHSUPPORT	457
APPENDIX C AVOID IDENTITY GAPS IN PRE-12.0.....	459
C.1 Introducing the 2-table design technique	459
APPENDIX D THE ELECTRONIC SUPPLEMENT TO THIS BOOK	464
INDEX	465

Changes in the second edition

For the second edition of this book, text and examples have been updated, improved, rewritten or added in many places. Also, a series of new topics was added. As a result, the second edition has expanded by about 80 additional pages.

Many changes have been made to all chapters. The most important differences with the first edition are:

- In chapters 4 and 6, the performance comparisons for the different types of queries have been extended with their respective results in ASE 15.0.
- In 8.2, a new trick is described to force `rand()` to be evaluated for every row in a multiple-row result set.
- In 8.4, the section about `newid()` was rewritten following functionality improvements in ASE 12.5.1, after the first edition of this book was published.
- Parts of chapter 10 were rewritten to cover new functionality in ASE 12.5.4 and 15.0, which allows `execute-immediate` to access externally declared variables.
- Virtually every page in chapter 11 was modified to cover enhancements to identity column functionality and internals in 12.5.1 and 15.0.
- Section 12.6.1 discusses a useful new command `kill with statusonly`.
- Section 13.11 is new, and discusses internals of setting session-specific options in stored procedures.
- Sections 14.7 and 14.8 are new, and discuss a handy tool for doing quick performance comparisons between queries, and a selected set of tricks with strings in T-SQL, respectively.
- The most significant change in the second edition is the new chapter 15. This chapter is dedicated to the new query processing infrastructure in ASE 15.0, and is a must-read for anyone upgrading to ASE 15.0.

Finally, I'd like to thank everyone who bought the first edition of this book: you've made it worthwhile writing it. Also thanks to all those who sent me additional tips & tricks and notified me of typos and errors.

I hope you'll find the second edition even more useful than the first.

Rob Verschoor
Rotterdam, July 2006