

ISUG

techcast series

ASE 15: Inside the New Query Processing Engine


April 18, 2007

SYBASE

ISUG

techcast series

ASE 15: Inside the New Query Processing Engine




Mike Harrold,
President, International
Sybase User Group

SYBASE

ISUG

techcast series

ASE 15: Inside the New Query Processing Engine



Sudipto R. Chowdhuri
Sr. Staff Engineer

SYBASE

ISUG

Agenda

- **The New Query Processing Engine In ASE 15**
 - Tackles Complex Queries
 - Handles Large Data Sets
 - Offers Advanced Monitoring & Diagnostics
- **Diagnosing Query Performance in ASE 15**
 - Query issues
 - Configuration parameter tips
 - Collecting information for bug reports
- **Q & A**

SYBASE 4

ISUG

Tackles Complex Queries

- **Large number of tables**

```
select * from T1, T2, ..... T50
where T1.a = T2.a AND T2.a = T3.a AND ..... T49.a = T50.a
```

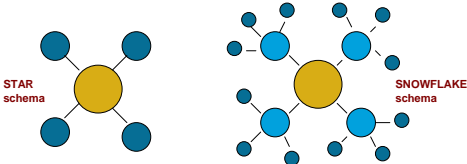
 - ASE 15.0 completes much faster
 - In built time-out stops optimization at the best plan chosen so far
 - Extra optimization may not yield a significantly better plan
 - Optimization time may surpass execution time
 - Accurate cost-based plan pruning for large join
 - No more "set table count <x>" adjustments for large joins

SYBASE 5

ISUG

Tackles Complex Queries

- **Superior plans for STAR/SNOWFLAKE schema joins**
 - STAR, SNOWFLAKE schemas have
 - Large FACT table, smaller (relative to FACT) DIMENSION tables
 - Joins only between FACT and DIMENSION tables
 - Index available on joining columns



SYBASE 6

ISUG
techcast series

Handles Large Data Sets

- Improved algorithms for most SQL operation
 - Uses Hash based operations
 - Selects merge based algorithm when proper ordering available
- Intelligent Partition elimination

~~c_custkey <= 100~~ c_custkey <= 200 c_custkey <= 300 customers
 select * from customers, orders
 where c_custkey = o_custkey and c_custkey > 110
~~o_custkey <= 100~~ o_custkey <= 200 o_custkey <= 300 orders

SYBASE 7

ISUG
techcast series

Handles Large Data Sets

- Parallelism
 - SQL operations can be done in parallel and pipelined to work in tandem


```
select sum(o_price), c_region from orders, customers
where o_orderdate > '01/31/2003' and
c_custkey = o_custkey group by c_region
```
 - Query has 2 parts join of customers and orders followed by Grouping, all of which will be done in tandem

SYBASE 8

ISUG
techcast series

Offers Advanced Monitoring and Diagnosis

- QP Metrics
 - Captures per query execution metrics
 - Min/max/avg PIO/LIO/CPU and number of times query is executed
 - SQL query text is captured as well
 - Stores in a system catalog - sysquerymetrics
 - Easy to identify the "problem queries"
 - Simple ORDER BY selects from the catalog
 - Integrated in front end tools
 - Easy to use
 - Metrics gathering can be enabled thru a configurable option
 - Stored metrics can be queried from sysquerymetrics

SYBASE 9

ISUG
techcast series

Graphical Plan Viewer

- Graphical plan generated by "planviewer" Sample

SYBASE 10

ISUG
techcast series

Diagnosing Query Performance Issues in ASE 15.0

- Query issues
 - Out of procedure cache (701 error)
 - Query running slow
- Configuration parameter tips
- Collecting information for bug reports

SYBASE 11

ISUG
techcast series

Query Issues – Out of Procedure Cache (701 errors)

- Default 'procedure cache size' has been increased to 14 Mb
 - Application specific, most likely 2 times the 12.5 procedure cache usage
- Best known for holding query plans / query trees
 - Query plans for stored procedures, triggers and dynamic sql are stored in the procedure cache
 - Defaults, rules and views are shared within procedure cache
- Sort operations
 - create index, update index statistics on non-leading index columns, group by, order by, etc.
 - Sorts due to query plans for merge joins
- Query Optimizer
 - Uses memory for sub-plans

SYBASE 12

Procedure Cache Mitigation

- Increase procedure cache 2X or more
- Optimizer procedure cache usage
 - search engine
 - more aggressive timeout values
 - use abstract plan or force options
 - histograms – use fewer steps or tuning factor
 - use stored procedures instead of ad hoc batch queries if possible

Query Issues – Change Query Plan

- Use abstract plan or force options
- Turn off new join algorithms
 - set merge_join 0/1
 - set hash_join 0/1
 - set nl_join 0/1
- Other operators to turn off/on
 - append_union_all, merge_union_all, merge_union_distinct, hash_union_distinct, store_index, index_intersection, multi_table_store_index, opportunistic_distinct_view, opportunistic_grouping, bushy_space_search, distinct_sorted, distinct_sorting, distinct_hashing, group_sorted, group_hashing, parallel_query

Query Running Slow

- Quick fix
 - Missing or poor Statistics
 - Early timeout
 - Runtime adjustment
 - Force options
- More analysis for Poor plan
 - Set plancost on – actuals vs estimates
 - Enable/disable algorithms
 - Index not chosen
 - Poor join order
 - Local index – no runtime partition elimination

Query Running Slow – Statistics

- Missing Statistics
 - set option show_missing_stats on
 - update index statistics <tablename>
 - update statistics <tablename> (col)
 - update statistics <tablename> (col1, col2 ...)
- Out of Date Statistics
 - datachange() functionality
- Poor Quality Statistics
 - using 200 steps (for tables in problem query)
 - sp_configure "histogram tuning factor", 10
 - uses 20 * 10 steps only when skew is found
 - sp_configure "number of histogram steps", 200
 - caveat – uses more procedure cache at compilation

Actuals vs. Estimates

- Set statistics plancost on
 - New in ASE 15.0
 - Straightforward association of estimates to actuals
 - Presents query plan as semi-graphical bushy tree shape to vastly improve readability
 - Used to view chosen plan
 - Includes estimated and actual cost per node
 - Need wide terminal or no word-wrap for large plans
 - Use isql ... -w140 ... to view output
- Plan viewer
 - gui interface to set plancost information

Sample Query

```

1> -- *** Query #7 ***
2> -- Find prospects for voice mail based on the
   criteria
3> -- that customers with call waiting and caller id
   are
4> -- good prospects
5> SELECT state, count(*)
6> FROM telco_facts T, service S,
   residential_customer C, month M
7> WHERE
8>     T.service_key = S.service_key
9>     AND T.customer_key = C.customer_key
10>     AND T.month_key = M.month_key
11>     AND S.call_waiting_flag = 'Y'
12>     AND S.caller_id_flag = 'Y'
13>     AND S.voice_mail_flag = 'N'
14>     AND C.state in ('NY', 'NJ', 'PA')
15>     AND M.fiscal_period = 'Q1'
16> GROUP BY state
    
```

ISUG
techcast series

```

Emit
(VA = 11)
3 rows est: 3
cpu: 600

HashVectAgg
Count
(VA = 10)
3 rows est: 3
lio: 5 est: 5
pio: 0 est: 0
bufct: 16

MergeJoin
Inner Join
(VA = 9)
9477 rows est: 1551

Sort
(VA = 6)
56088 rows est: 10001
lio: 407 est: 17
pio: 0 est: 23
cpu: 5400 bufct: 16

MergeJoin
Inner Join
(VA = 9)
50050 rows est: 10001
lio: 1945 est: 1947
pio: 0 est: 1945

TableScan
residential_customer (C)
(VA = 7)
100000 rows est: 15509
lio: 1945 est: 1947
pio: 0 est: 1945

Sort
(VA = 3)
208173 rows est: 120001
lio: 2231 est: 239
pio: 0 est: 468
cpu: 3700 bufct: 19

TableScan
service (S)
(VA = 4)
2 rows est: 1
lio: 1 est: 2
pio: 0 est: 2

NestLoopJoin
Inner Join
(VA = 2)
300000 rows est: 120001

TableScan
month (M)
(VA = 0)
6 rows est: 3
lio: 1 est: 2
pio: 0 est: 2

IndexScan
telco_fact_70400 (T)
(VA = 1)
300000 rows est: 120001
lio: 5616 est: 2416
pio: 0 est: 2413

Execution Time 60.
SQL Server cpu time: 6000 ms. SQL Server elapsed time: 6130 ms.
CONTINUED IN UPPER RIGHT CORNER
  
```

SYBASE 19

ISUG
techcast series

SHOW_MISSING_STATS

```

- Previous example used "update index statistics" on all
tables in the query
- dbcc traceon( 3604 )
go
SET OPTION SHOW_MISSING_STATS ON
go
<same query as before>
- NO STATS on column service.voice_mail_flag
NO STATS on column service.caller_id_flag
NO STATS on column service.call_waiting_flag
NO STATS on column month.fiscal_period
state
-----
NY          4215
NJ          2306
PA          2956

( 3 rows affected)
- NOTE THE MESSAGES ON THE NON-INDEXED COLUMNS ABOVE
  
```

SYBASE 20

ISUG
techcast series

Update Statistics on Columns

- Based on show_missing_stats!
 - update statistics month(fiscal_period)
 - update statistics service(voice_mail_flag)
 - update statistics service(caller_id_flag)
 - update statistics service(call_waiting_flag)
- Note the improved join and restrict estimates on next slide, with improved timing on the modified plan

SYBASE 21

ISUG
techcast series

```

Emit
(VA = 11)
3 rows est: 3
cpu: 300

HashVectAgg
Count
(VA = 10)
3 rows est: 3
lio: 5 est: 5
pio: 0 est: 0
bufct: 16

MergeJoin
Inner Join
(VA = 9)
9477 rows est: 10189

Sort
(VA = 6)
34773 rows est: 50302
lio: 200 est: 133
pio: 0 est: 255
cpu: 4300 bufct: 16

MergeJoin
Inner Join
(VA = 9)
50301 rows est: 50302
lio: 1945 est: 1947
pio: 0 est: 1945

TableScan
residential_customer (C)
(VA = 7)
100000 rows est: 15509
lio: 1945 est: 1947
pio: 0 est: 1945

Sort
(VA = 3)
209395 rows est: 300001
lio: 1350 est: 591
pio: 0 est: 1171
cpu: 1550 bufct: 30

TableScan
service (S)
(VA = 4)
2 rows est: 1
lio: 1 est: 2
pio: 0 est: 2

NestLoopJoin
Inner Join
(VA = 2)
300000 rows est: 120001

TableScan
month (M)
(VA = 0)
6 rows est: 6
lio: 1 est: 2
pio: 0 est: 2

IndexScan
telco_fact_70400 (T)
(VA = 1)
300000 rows est: 300001
lio: 5616 est: 2413
pio: 0 est: 2413

Execution Time 47.
SQL Server cpu time: 4700 ms. SQL Server elapsed time: 4716 ms.
CONTINUED IN UPPER RIGHT CORNER
  
```

SYBASE 22

ISUG
techcast series

Statistics Quality

- Datachange()
 - Helps determine if stats are out of date
 - Remember! datachange() only flushed on server shutdown unless "sysstatistics flush interval" set
- set option show on
 - not needed for missing statistics, as was the case for 302 in which one searched for magic numbers.
 - "set option show_missing_stats on" is more precise
 - Useful for statistics quality if datachange indicates that statistics are up to date with little DML
 - Use more steps if selectivity is inaccurate
 - Preferable to use histogram tuning factor

SYBASE 23

ISUG
techcast series

Slow Plans – Optimizer Timeout Troubleshooting

- Timeout possible with queries referencing many tables
- Set option show on
 - run query and search for possible timeout message
 - !! Optimizer has timed out in this Opt block !!
 - Raise optimization timeout limit to avoid timeout
- Increasing timeout will increase compilation time
 - increase may help some queries and hurt others
 - good stats needed for accurate timeout estimates!
 - place ad hoc queries into stored procedures / statement cache if possible
- Increasing timeout may increase procedure cache usage
- sp_configure "optimizer timeout limit", 100
- Select plan "(use opttimeoutlimit 100)"

SYBASE 24

Abstract Plan Tips

- **Easier to edit an existing plan**
- **To start writing AP's**
 - Let ASE generate a plan and edit it
 - Use SET OPTION SHOW_ABSTRACT_PLAN ON
 - Edit plan to easier read and change
 - % in vi is very helpful to match parenthesis
 - Attach to query using PLAN clause
 - Change to test hypothesis
- **Plan returned by**

- SET OPTION SHOW_ABSTRACT_PLAN ON

The Abstract Plan (AP) of the final query execution plan:

```
( group_hashing ( h_join ( h_join ( nl_join ( i_scan pk_month (
  table ( M month ) ) ) ( i_scan pk_telco_facts ( table ( T
  telco_facts ) ) ) ( i_scan pk_service ( table ( S service ) )
  ) ) ( i_scan pk_customer ( table ( C residential_customer ) ) )
  ) ) ( prop ( table ( M month ) ) ( parallel 1 ) ( prefetch 4 ) (
  lru ) ) ( prop ( table ( T telco_facts ) ) ( parallel 1 ) (
  prefetch 4 ) ( lru ) ) ( prop ( table ( S service ) ) ( parallel
  1 ) ( prefetch 4 ) ( lru ) ) ( prop ( table ( C
  residential_customer ) ) ( parallel 1 ) ( prefetch 4 ) ( lru ) )
  ) ) ) ) )
```

To experiment with the optimizer behavior, this AP can be modified and then passed to the optimizer using the PLAN clause:
SELECT/INSERT/DELETE/UPDATE ... PLAN '(...)'

Abstract Plan Tips

- **Indent using join, union, etc. nodes**

```
( group_hashing
  ( h_join
    ( h_join
      ( nl_join
        ( i_scan pk_month ( table ( M month ) ) )
        ( i_scan pk_telco_facts ( table ( T telco_facts ) ) )
      )
      ( i_scan pk_service ( table ( S service ) ) )
    )
    ( i_scan pk_customer ( table ( C residential_customer ) ) )
  )
)
```

Abstract Plan Hints

- **Attach to query**

```
SELECT ... FROM ... WHERE ...
PLAN
\
( group_hashing
  ( h_join
    ( h_join
      ( nl_join
        ( i_scan pk_month ( table ( M month ) ) )
        ( i_scan pk_telco_facts ( table ( T telco_facts ) ) )
      )
    )
  )
\
```

Abstract Plan Hints

- **Modify to test hypothesis**

```
SELECT ... FROM ... WHERE ...
PLAN
\
( group_hashing
  ( h_join
    ( h_join
      ( nl_join
        ( i_scan pk_month ( table ( M month ) ) )
        ( i_scan pk_telco_facts ( table ( T telco_facts ) ) )
      )
    )
  )
\
```

New Configuration

- **[Query Tuning]**

```
optimization goal = DEFAULT
allow backward scans = DEFAULT
abstract plan load = DEFAULT
.....
max scan parallel degree = DEFAULT
max repartition degree = DEFAULT
max resource granularity = DEFAULT
enable metrics capture = DEFAULT
optimization timeout limit = DEFAULT
sproc optimize timeout limit = DEFAULT
min pages for parallel scan = DEFAULT
prod-consumer overlap factor = DEFAULT
```

Configuration Details

- **Number of histogram steps**
 - Same as pre-15.0
 - More important in 15.0 than in pre-15.0
- **Enable sort-merge join and JTC**
 - Configuration parameter deprecated
 - JTC enabled by default in 15.0, cannot turn off
 - Merge join is enabled by default
 - can be controlled through optimization goal, SET options and AP's

Configuration Details

- **Max repartition degree**
 - New in 15.0
 - Transient repartitioning parallelism when underlying tables are not partitioned appropriately
 - Very CPU intensive
- **Max resource granularity**
 - New in 15.0
 - Percentage of system resources to be used
 - 100 / (max number of competing parallel threads)
 - Minimum value is 1% for 1/100 of system
 - Worker threads
 - Buffer cache

Configuration Details

- **Optimization timeout limit**
 - New in 15.0
 - Controls time spent in search engine in approx per cent of estimated execution time
 - 1 – 1,000 are valid values
 - `SET PLAN OPTTIMEOUTLIMIT <n>`
 - `SELECT ... FROM ... WHERE ...`
`PLAN '(use opttimeoutlimit <n>).'`
 - Not applicable to object

Collecting Information for Bug Reports

- **Repro always help**
- **Otherwise,**
 - `optdiag simulate`
 - **force good plan, as well as info on bad plan**
 - set statistics plancost on
 - set statistics time on
 - set option show on (caveat – could be huge)
 - set showplan on (with dbcc traceon(526))
 - **Wrong answer problems**
 - set option show_code_gen on
 - Traceflags 302, 310 subsumed by "set option show on"

Query Processing in Galaxy

Q & A