Practices and Features for Performance on BI Query Workloads – Informix 11.70

Chat with the Lab
Presented by: Veronica Gomes - vgomes@us.ibm.com
IT Specialist, IBM Informix Development
Agenda

• Understanding the query workload to improve
• Database design considerations for BI query performance
• Informix features for BI query performance
• Informix Warehouse Accelerator for breakthrough BI speed
• Q&A
Understanding the query workload to tune

- Business Intelligence (BI) query workloads
- Also referred to as those queries typical of:
  - On-Line Analytical Processing (OLAP) systems
  - Decision Support Systems (DSS)
  - Business Intelligence (BI) or Analytics applications
## Queries in Analytical vs. in Transaction processing

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>OLTP (Transactional)</th>
<th>OLAP (Analytical)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main purpose they support</td>
<td>Day-to-day operations, control and run fundamental business tasks</td>
<td>Managerial, strategic planning, problem solving, decision support</td>
</tr>
<tr>
<td>Queries</td>
<td>Short and relatively simple SQL, accessing one or a few records (look-up)</td>
<td>Long complex SQL with calculations/aggregations across many records</td>
</tr>
<tr>
<td>Access to data</td>
<td>Random, through indexes</td>
<td>Sequential access</td>
</tr>
<tr>
<td>Nature of most queries</td>
<td>Row-oriented</td>
<td>Column-oriented</td>
</tr>
<tr>
<td>Best database design</td>
<td>Normalized (3NF+)</td>
<td>Multi-Dimensional –STAR</td>
</tr>
<tr>
<td>Response times</td>
<td>Immediate, sub-second</td>
<td>Several secs, mins to hours</td>
</tr>
<tr>
<td>Workload predictability and tuning</td>
<td>Pre-compiled queries; repeated execution of same queries; come from known applications</td>
<td>Ad-hoc queries coming from different apps/tools; unpredictable workload; not optimized in advance</td>
</tr>
</tbody>
</table>
Queries in Analytical vs. in Transaction processing

• 4 examples of **OLTP** queries (Transaction processing):

  - SELECT * FROM customer WHERE phone='555-555-5555';

  - SELECT stock_num, manu_code FROM stock
    WHERE unit_price BETWEEN 125.00 AND 200.00 ;

  - SELECT DISTINCT customer_num, stock_num, manu_code
    FROM orders, items WHERE order_date BETWEEN '6/1/11' AND '9/1/11' ;

  - SELECT cc.* FROM cust_calls cc, customer c
    WHERE c.company='A_GIVEN_COMPANY'
    AND cc.customer_num = c.customer_num AND
    cc.call_dtime BETWEEN (CURRENT - INTERVAL(7) DAY TO DAY) AND CURRENT ;
Queries in Analytical vs. in Transaction processing

- Example of **OLAP** query (Analytical processing):
  - Profits generated by a given category of products, per region, during first half of 2010 period:

  ```sql
  SELECT FIRST 100 SUBCATEGORY_DESC,
          SUM(CASE REGION WHEN 'North' THEN EXTENDED_PRICE-EXTENDED_COST ELSE 0 END) AS NORTHERN_REGION,
          SUM(CASE REGION WHEN 'South' THEN EXTENDED_PRICE-EXTENDED_COST ELSE 0 END) AS SOUTHERN_REGION,
          SUM(CASE REGION WHEN 'East' THEN EXTENDED_PRICE-EXTENDED_COST ELSE 0 END) AS EASTERN_REGION,
          SUM(CASE REGION WHEN 'West' THEN EXTENDED_PRICE-EXTENDED_COST ELSE 0 END) AS WESTERN_REGION,
          SUM(CASE WHEN REGION IN ('North', 'South', 'East', 'West') THEN EXTENDED_PRICE-EXTENDED_COST ELSE 0 END) as ALL_REGIONS
  FROM PERIOD per, PRODUCT prd, STORE st, DAILY_SALES s
  WHERE per.PERKEY=s.PERKEY AND prd.PRODKEY=s.PRODKEY AND t.STOREKEY=s.STOREKEY
  AND per.CALENDAR_DATE BETWEEN '01/01/2010' AND '06/30/2010' AND CATEGORY=88
  GROUP BY SUBCATEGORY_DESC
  ORDER BY SUBCATEGORY_DESC;
  ```
Some examples of questions BI queries help solve

- What were the top 10 most returned products last month?
- What are our most/least profitable products, customers, seasons, months, regions, channels...?
- Our top 5 performer sales’ employees last month, and the ones consistently in the top 20 over the last 3 months?
- How often does a customer buy from us again, from our brick & mortar stores, and from our website?
- In how much do our channels contribute to our revenue?
- How often do we gain / lose a customer?
- Difference in sales during the past season during a new campaign, compared to the same season last year?
SQL query processing of a BI query

- Runs against data warehouses or mixed workload databases
  - Current and historic large databases, constantly growing
  - Mixed OLTP/OLAP databases used to be very challenging
- The BI query execution in RDBMS typically involves:
  - Multiple-table joins, with often one or couple of very large tables
  - Complex calculations and aggregate functions on the largest tables
  - Need to access few columns but through large amount of rows
  - Often produce sequential scans on large tables
  - Predicates and filters before and after joins
  - Intermediate or temporary result sets
  - Sort and/group of result sets, which may be large
Database design considerations for best BI query performance

• Best database model for performance in BI/OLAP queries
  – Star / Snowflake schemas
  – Which are the typical schemas in data marts

• Informix 11.70 key capabilities for OLAP queries on Star/Snowflake database subset:
  – Informix Warehouse Accelerator (breakthrough speed plug-in)
  – STAR joins
Logical and Physical Database Design

- IBM Informix 11.70 – specially Informix Ultimate Warehouse Edition which includes the Informix Warehouse Accelerator –
  - is an advanced, easy to use and support database
  - that combines the best row and columnar database technologies
  - to achieve extreme performance in mixed OLTP/OLAP workloads

- However, everything starts with a good design… 😊

- Whether we use Informix for a data warehouse or for a mixed transactional/analytical system…

- We should start by taking a look at the logical and physical design of the database subset of interest for the BI queries

- http://www.slideshare.net/journalofinformix/informix-physical-database-design-for-data-warehousing
Dimensional Model is ideal for data marts and OLAP

Typical Data Model for OLTP (Highly-Normalized)

Typical Data Model for OLAP/Warehouse
(Dimensional design: STAR/Snowflake schema)
A Dimensional Model…

- Is a flat representation of a **multi-dimensional** business model
- Contains a set of detailed business **facts** surrounded by **multiple dimensions** that describe those facts and provide hierarchical summarization paths
- May produce a **star schema** or a **snowflake schema**
- Is easy to understand and relate to business users and business needs
- Provides **superior query performance** to OLAP and business queries by minimizing table joins
- Star schema approach to dimensional data modeling was pioneered by Ralph Kimball (founder of Red Brick Systems)
STAR vs Snowflake schema

**STAR**
More space, Less # of joins

**SNOWFLAKE**
Less space, More # of joins
Questions…

- Are the tables used in our complex BI/DSS queries, related in a way that looks like a **star/snowflake** schema?
- Could the data of interest for the complex queries and reports be designed this way, in a historic copy –datamart?

```sql
SELECT t.quarter, sum(o.quantity)
FROM orders o, location l, customer c, time t
WHERE o.location_id = l.location_id
    AND o.cust_num = c.cust_num
    AND o.date_time = t.date_time
    AND c.type <= 3
    AND l.region_id = 4
    AND t.quarter between 1 and 20
GROUP BY t.quarter;
```
If the data follows a star/snowflake schema (1)

- Breakthrough performance in complex/analytic queries on that data subset can be achieved through the Informix Warehouse Accelerator
  - Ex: 50x, 100x, 500x, 1000 times+ faster responses!
  - Thanks to the advanced combination of technologies in the Accelerator for unprecedented speed in BI/OLAP queries
  - Informix database server uses the Accelerator behind the scenes, transparently to the users submitting the BI queries

**Informix Warehouse Accelerator:**

Is an advanced in-memory deep compressed *columnar* database technology plug-in for Informix *(relational)* databases to provide transparent breakthrough speed in OLAP queries
If the data follows a star/snowflake schema (2)

When Informix Warehouse Accelerator cannot be used…

• The Informix database server 11.70 alone can use this new Join method with its underlying technologies to improve performance of BI queries on that data subset:

  – **STAR Join** uses:
    - Uses “push-down” hash joins
    - Leverage new multi-index scans, skip scan and bitmap vector technologies to significantly reduce the data on each pair-join via:
      → Index push-down
      → Bit-vector push down
Tuning OLAP/warehousing queries

• **Star Join Optimization**
  – Requires minimum Update Statistics Low collected, for the Optimizer to identify Fact Tables in the query
  – PDQPRIORITY needs to be set for the Optimizer to consider a star join and to choose the right fact table
  – Directives: STAR_JOIN and FACT (<fact table(s),>)

• **Multi-Index Scan**
  – Indexes must be B-tree indexes, attached or detached
  – Table cannot be remote, external table or hierarchical
  – A multi-index scan uses only leading column in composite indexes
  – Directives: +MULTI_INDEX, AVOID_MULTI_INDEX
Tuning OLAP/warehouse queries

- **Light Scans**
  - The fastest way to do a sequential scans
  - We have no direct control of it to happen
  - Avoid using Cursor Stability isolation (this level prevents light scans)
  - Set big enough Virtual Memory and Memory for Decision support: `SHM VIRT SIZE, DS_TOTAL_MEMORY` for light scans to use it and big buffers allocation
  - `onstat -g scn`
TuningOLAP/warehousing queries

- If using Informix Warehouse Accelerator
  - SETENVIRONMENT use_dwa '1'; to enable acceleration
  - SETENVIRONMENT use_dwa '3'; to enable acceleration + related debug info in online.log
  - Use sysdbopen() to set use_dwa to 1, for transparency to BI tools
OPTCOMPIND

• At ONCONFIG or overwritten by a session
  – Consider `sysdbopen()` to enable/force STAR Join Optimization

• Affects join strategy and index use in a query plan

• 2 is default and typical value in data warehouse and mixed environments
  – Unless you want to force index precedence and nested joins

• 0: Nested-loop join is preferred over a sort-merge or hash join

• 1: If isolation level is not Repeatable Read, the optimizer behaves as in setting 2; otherwise, it behaves as in setting 0

• 2: Nested-loop joins are not necessarily preferred. The optimizer bases its decision purely on costs
Data Storage and Placing

- Dbspaces and disks layout / separation / IO throughput
  - DIRECT_IO, KAIO, AIO VPs (auto)
- Detached indexes (fragmented versus not fragmented)
- Use larger page size (dbspaces & bufferpools) as needed
- Use non-logging databases for DSS/DW as possible
  - Or Dirty Read as possible, if logged database
- Right Partitioning (Fragmentation) for largest tables
  - Round-Robin, Expression, Range Interval, List
  - For scalability, performance, time-cyclic data management
- Deep Compression, Repack and Shrink, defragmenter
PDQPRIORITY and its ONCONFIG Parameters

- **SET PDQPRIORITY** at DSS/BI session level
  - Transparently using `sysdbopen()`
  - Use low value (ex: 1) if only parallel scans
  - Use high value (ex: 80+) for high parallelism in all operations
  - Required for STAR Join Optimization directives to be taken

- **MAX_PDQPRIORITY**: Set to 90 or 100

- **DS_TOTAL_MEMORY**: 90% of `SHMVIRTSIZE` or 50-90% of `SHMTOTAL`

- **DS_MAX_QUERIES** – The maximum number of PDQ queries that can be executed concurrently. Start with default.

- **DS_MAX_SCANS** – The maximum number of scan threads available for PDQ queries. Start with default.
Temporary Tables and Temp DBspaces

• Non-Table Temporary Tables: TEMPTAB_NOLOG to 1
• Temporary dbspaces do not use direct I/O, so is important to check for enough AIO VPs or AUTO_AIOVPS
• Keep several (3+) big enough temporary dbspaces (DBSPACETEMP), with similar sizes
  – operations like MERGE, temp tables, hash overflows, autoindex, sorts for group by and order by, and index builds will make use of them
Partitioning (Fragmentation)

- Different strategies for different /right needs (table, index)
- Performance in parallel scans and partition elimination
- Update statistics per fragment
- Useful for time cyclic data management: online attach / detach partition (roll-on/roll-off)
- Do not over-partition
- Consider I/O throughput of where dbpsaces are located
- Take advantage of partitioning along with PDQ
Considerations on Indexes and Constraints (1)

• Use Surrogate keys for Primary Keys:
  – Generally more efficient and take less space
    ▪ They use best data types (ex: integers) instead of less-efficient ones
    ▪ Although it may cause additional joins for retrieval of the business key
  – Good in data integration scenarios
  – Can be generated automatically within the DBMS (SQL) or ETL
  – They allow to keep things “simple” and flexible
  – Avoid using “smart” keys or the same keys used in Production systems
    ▪ Since a key could be reused after a long while or business rules may change
Considerations on Indexes and Constraints (2)

- Start with the unique indexes and right PK constraints
  - Composite first (highest selective column leads), then single column
- Then, the Referential Constraints and Non-Unique indexes (Joins)
- Consider Forest of Trees indexes on deep B+ tree with many concurrent users (mixed OLTP/OLAP systems)
- Decide: attached/detached indexes on your fragmented tables?
- In STAR schemas and queries, it is important that:
  - Each table has at least one Unique Index (or else, a Primary Key)
  - There is referential integrity between the fact and its dimension tables
  - In order to use the Informix Warehouse Accelerator (columnar plug-in)
  - To get more benefits in star queries even though STAR Join Optimization can work without the indexes in Informix (relational DB)
Considerations on Indexes and Constraints (3)

• Avoid creating more indexes than the ones really needed

• Other indexes that might want to add (depending on several factors) are those on columns way too often mentioned in:
  – Filters/predicates (with high selectivity)
  – Order by, Group by
  – Select’s projection list
  – In order to:
    ▪ avoid autoindex or temporary tables for sorts
    ▪ allow index-only scans over table scans in large tables

• Finally, run Update Statistics using the recommended practices:
  – (Script) UPDATE STATISTICS commands to allow the optimizer to work its best
Summary/aggregate tables

• If you need to maintain pre-computing summarized data (based on the detailed data on the fact table and some hierarchy level on one or more dimension tables)
  – Help save time and resources to process summarized queries
    ▪ Ex: montly_sales_per_customer
• Keep refreshed via ETL at the same time as the fact table
• Informix Warehouse Accelerator prevents the need for summary / MQT tables in most cases
  – However, you can use them with the Accelerator to get extremely fast results on summary tables too
Join Types (1)

Nested Loop Join

In a nested-loop join, the database server scans the first, or outer table, and then joins each of the rows that pass table filters to the rows found in the second, or inner table.

To be able to use it…

- OPTCOMPIND in 0, 1 or 2
- Statistics Up to Date
- Find efficient index on largest table (join columns)

Forces / promotes use:
- OPTCOMPIND in 0
- SET OPTIMIZATION FIRST_ROWS
- USE_NL directive

Prevents:
- SET OPTIMIZATION ALL_ROWS
- AVOID_NL directives
Join Types (2)

Hash Join

The optimizer usually uses a hash join when at least one of the two join tables does not have an index on the join column or when the database server must read a large number of rows from both tables. No index and no sorting is required when the database server performs a hash join.

To be able to use it…

- OPTCOMPIND in 2
- Update Statistics Low
- Enough Memory and temporary space for hash table

To force / promote it:

- SET OPTIMIZATION ALL_ROWS
- Directive USE_HASH
- Other: <table>/PROBE, <table>/BUILD

To avoid it:

- AVOID_HASH
Join Types (3)

STAR Join

How it works to reduce probe tables in Hash Join

An insider's guide to star join optimization in IBM Informix Server

Join Types (4)

STAR Join

To be able to use it…

- PDQPRIORITY needed
- Equi join, unique Indexes desired
- Update Statistics Low
- 1 Fact table; star or snowflake

To enable or force it: (sysdbopen())

- set PDQPRIORITY 90;
- SET OPTIMIZATION ENVIRONMENT STAR_JOIN 'enabled'|'forced'
  Other: FACT '<table_list>' | AVOID_FACT <table_list>' | NON_DIM '<table_list>'
  Or directive(s): --+ STAR_JOIN, FACT(table_name),…
Other topics to consider

- Sub-queries (semi-joins) and correlated sub-queries
- Derived tables (select from (select...))
- Views (folding a view to parent table)
- Temporary tables for intermediate results to other queries
- Distributed queries
- BI users and the Accelerator on MACH clusters secondaries

**Support and Troubleshooting:**
- SET EXPLAIN, dbschema, SQLTRACING and onstat –g his
- Optimizer directives to change Optimizer behavior
- **Informix Warehouse Accelerator:** onstat -g aqt, AQT=1 env variable before dbschema; CONSOLE and MSGPATH files; use_dwa ‘3’ or ‘debug on’ modes; O/S messages log
Informix Warehouse Accelerator – Breakthrough Technology for OLAP Performance

**Extreme Compression**
3 to 1 compression ratio

**Row & Columnar Database**
Row format within IDS for transactional workloads and columnar data access via accelerator for OLAP queries.

**In Memory Database**
3rd generation database technology avoids I/O. Compression allows huge databases to be completely memory resident

**Multi-core and Vector Optimized Algorithms**
Avoiding locking or synchronization

**Predicate evaluation on compressed data**
Often scans w/o decompression during evaluation

**Frequency Partitioning**
Enabler for the effective parallel access of the compressed data for scanning. Horizontal and Vertical Partition Elimination.

**Massive Parallelism**
All cores are used for each query

Comes with Smart Analytics Studio, a GUI tool, for configuring data mart and monitoring IWA
How the accelerator works

BI App

Configure, offload data mart
(Using ISAO)

Bulk Loader

Linux-64, Intel

Query Processor

Compressed DB partition

Informix Warehouse Accelerator

Informix env

HPUX-64, AIX-64, SOL-64, Linux-64

Redirect queries

Warehouse

DataMart

Query Results
IBM Smart Analytics Studio (ISAO)

Easy to use graphical tool for data mart configuration
# Customer Warehouse Scenarios and Performance Results

<table>
<thead>
<tr>
<th>Federal Agency - Germany</th>
<th>Federal Agency - USA</th>
</tr>
</thead>
<tbody>
<tr>
<td>- 537 Queries executed over a 30 GB data mart</td>
<td>- Sample data set from 2 TB warehouse</td>
</tr>
<tr>
<td>- 432 Queries accelerated via IWA</td>
<td>- 8 representative queries ran 127 times faster</td>
</tr>
<tr>
<td>- Performance with IDS+IWA was 90 times faster than XPS</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Large Shoe Retailer</th>
<th>Large Global Retailer</th>
</tr>
</thead>
<tbody>
<tr>
<td>- BI Queries on 150 GB data set</td>
<td>- 10 GB data mart sample</td>
</tr>
<tr>
<td>- Fact table – 1 billion rows</td>
<td>- Fact table – 25 million rows</td>
</tr>
<tr>
<td>- Queries against Informix – 22 minutes</td>
<td>- Queries against Informix – over 20 mins</td>
</tr>
<tr>
<td>- Queries against Informix + IWA: 4 secs, 330 times faster</td>
<td>- Queries against Informix + IWA: 4 secs, 300 times faster</td>
</tr>
</tbody>
</table>
Thank you!!!!

• Questions & Answers