Data Warehouse Performance

DB2 Version 8: More Opportunities!

David Beulke
Principal Consultant,
Pragmatic Solutions, Inc.

DBeulke@compserve.com
703 798-3283

DB2 V8 Performance Opportunities

- Leverage your environment & designs
  - DB2 V8 for z/OS – G/A 1st Half of 2004
  - DB2 V8 for LUW – Fixpak 3
  - IDUG V8 articles
    - January 2002 special briefing

- **Z/OS http://www.idug.org/idug/member/journal/may03/article02.cfm**
- **LUW http://www.idug.org/idug/member/journal/aug02/unveiling8.0.cfm**

David Beulke is an internationally recognized DB2 consultant, author and lecturer. He is known for his extensive expertise in database performance, data warehouses and internet applications. He is currently President of the International DB2 Users Group (IDUG), a member of IBM DB2 Gold Consultant program, a columnist for DB2 Magazine, co-author of the IBM V7 and V8 z/OS DB2 Administration Certification exam, co-author of the Business Intelligence Certification exam, former instructor for The Data Warehouse Institute (TDWI), and former editor of the IDUG Solutions Journal. He has helped clients be successful on the mainframe, Linux, UNIX and Windows platforms and has vast performance tuning experience in the systems, applications and programming areas.

Dave can be reached at 703 798-3283 or through email at DBeulke@compuserve.com

There are many other new Version 8 enhancements that make the upcoming IDUG conference in May an educational priority for preparing for Version 8. Getting information from real users in IBM Quality Partnership Program (QPP) and IBM developers will help your company leverage this state-of-the-art technology. IDUG has published Version 8 white papers on the z/OS and Linux, UNIX and Windows platforms that exhaustively describe many other Version 8 enhancements. These white papers can be found at the links below.

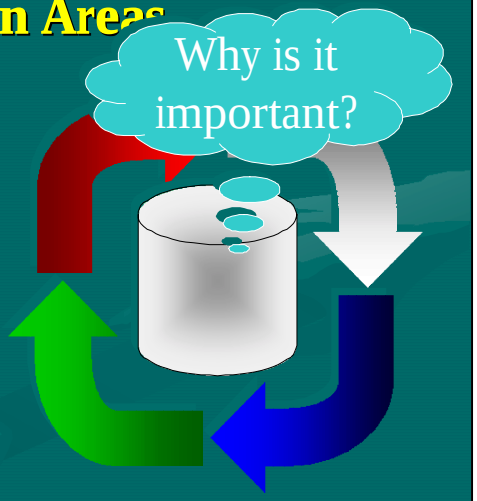
Z/OS http://www.idug.org/idug/member/journal/may03/article02.cfm

LUW
http://www.idug.org/idug/member/journal/aug02/unveiling8.0.cfm

DB2 UDB Server Version 8 for z/OS is a tremendous release and IBM continues to extend its reputation as the state-of-the-art leader in DBMS technology.

Main Areas

- New SQL
  - OLAP & ETL
- Data Structures
  - Databases
  - Indexes
- Partitioning
- Parallelism



LUW SQL Features

- OLAP functions
- RANK
- DENSE_RANK
- ROW_NUMBER
  - ORDER BY clause
- OVER clause
  - PARTITION BY clause
  - RANGE clause
  - ROW clause
- ROLLUP
- CUBE
  - Group By or Grouping Sets
- DB2 Cube Views
  - Virtual cube backed by real structures
- XML and Metadata extensions

RANK Example:

```
SELECT WORKDEPT,
AVG(SALARY+BONUS)
      AS AVG_TOTAL_SALARY,
RANK() OVER (ORDER BY
      AVG(SALARY+BONUS)
      DESC)
      AS RANK_AVG_SAL
FROM EMPLOYEE
GROUP BY WORKDEPT
ORDER BY RANK_AVG_SAL
```

DENSE RANK Example:

```
SELECT WORKDEPT, EMPNO,
      LASTNAME, FIRSTNME,
      EDLEVEL
DENSE_RANK()
      OVER (PARTITION BY
      WORKDEPT
      ORDER BY EDLEVEL DESC)
            AS RANK_EDLEVEL
FROM EMPLOYEE
ORDER BY WORKDEPT,
      LASTNAME
```

ROW NUMBER Example:

```
SELECT ROW_NUMBER()
OVER (ORDER BY WORKDEPT, LASTNAME)
      AS NUMBER,
LASTNAME,
SALARY
FROM EMPLOYEE
ORDER BY WORKDEPT, LASTNAME
```


There are many data warehouse performance aspects, and knowing all the different performance aspects, issues and considerations for building your data warehouse can be a tremendous task. DB2 offers many performance aspects missing from other RDBMS that can make a huge performance difference. This presentation will highlight the performance advantages of DB2 and how to handle all the design issues, alternatives and considerations experienced while building a large high performance data warehouse.

Learn how to build your data warehouse for maximum SQL access performance while maintaining the key values for partitioning and parallelism considerations. Also evaluating OLAP tool performance aspects, the connectivity issues to the Internet and discussing the considerations for large numbers of concurrent users are some of the other design points that will be discussed. Through this presentation you will learn many design options and alternatives to maximum performance for your data warehouse design.

SQL - OLAP Features

The SQL OLAP functions performed inside DB2 provide the answers much more efficiently then manipulating the data in a program. Like join activities and other data manipulation that DB2 can do directly the SQL OLAP functions can greatly reduce overall I/O and CPU utilization.

The functions are particularly good for getting the top number of data rows that match a criteria.

The OLAP RANK Function can be used to order and prioritize your data according to your specific criteria. RANK orders your data assigning successive sequential numbers to the rows returned from the SQL query. The ranked rows can be individual data rows or groups of data rows.

The OLAP DENSE_RANK Function can also be used to order and prioritize your data according to your specific criteria.

DENSE_RANK orders your data and assigns successive sequential numbers based on the OVER PARTITION data values found. DENSE_RANK differs from RANK because common values or ties are assigned the same number.

The OLAP ROW NUMBER function can be used to assign row numbers to the retrieved data. In addition ROW NUMBER can be used to order your data assigning successive sequential numbers for evaluation for application decisions or en-user screen processing.

Success, ROI, and performance are all determined by the end-user getting their questions answered. The common preliminary questions for a sales based DW that must be answered are:

- **How much product did we sell last (year, month or time period)?**
- **How do these sales figures compare to last (year, month or AP)?**
- **How much profit did we make on sales during this period?**
- **Who did we sell our products to?**
- **What categories or classifications separate our customers ?**
- **What products were sold to which customers classifications?**
- **What was the response rate on the marketing promotion?**
- **What percentage of our customer are new?**

How many I/Os does it take with your DW design to get these common questions answered?

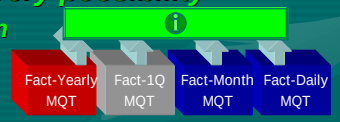
Z/OS SQL Features

- CTEs – Common Table Expressions
  - Recursive SQL
  - Build of Material
    - Parts of parts
- Multi-Row Actions
  - INSERT
  - FETCH
    - NEXT/PRIOR Cursors
  - UPDATE/DELETE with cursor
- Scrollable Cursors
  - Sensitive Dynamic
- INSERT with SELECT
- Identity column improvements
- New Sequence column
- Group by A+B
- Multiple DISTINCT

Next the new Version 8 features of Common Table Expressions (CTEs) and recursive SQL can be combined to provide a powerful data warehousing design solution for extracting and working on a distinct set of information. CTEs provide a new way to extract a result set from the database based on desired criteria. Next this unique result set can be referenced in additional SQL, further refining the answer for the end-user. CTEs avoid the catalog overhead of views, provide the ability to use host variables and avoid data changes from other INSERT, UPDATE or DELETE SQL operations. With recursive SQL, SQL that references itself, distinct result sets can have the power of SQL repeatedly applied to quickly and efficiently derive the answers. Combining these techniques provides a great way to give the data warehouse end-users answers to their unique criteria while avoiding conflicts with other users, maintaining data security and easily repeating the power of SQL criteria.

MQT – 10 to 1000 times improvement!

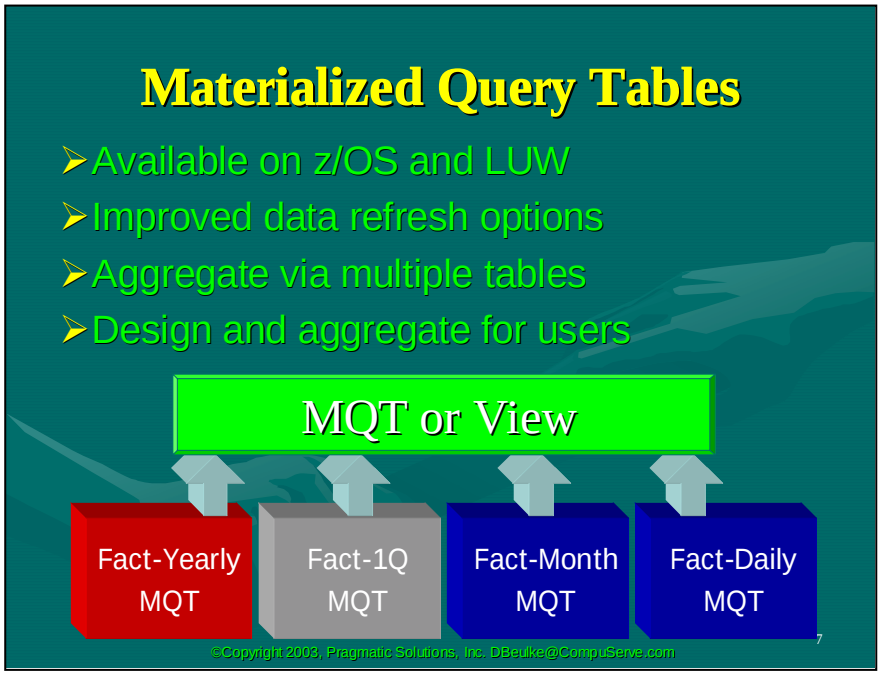
- *5B rows per year–10 per 4k page= ½B pages*
- *MQT aggregates save large amounts of everything*
  - *Create aggregates for every possibility*
    - *"On Demand" information*
    - *Sales by department*
    - *Sales by zip code*
    - *Sales by time period – day/week/month/quarter/AP*

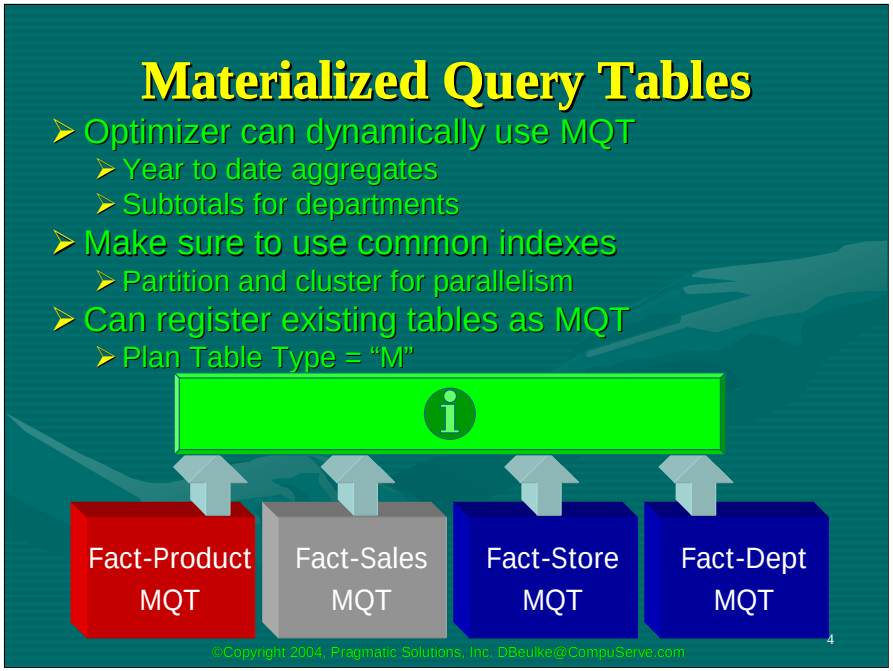


- *All reporting and analysis areas*
- *Trace usage to create/eliminate aggregates*
- *Total by month ½B I/Os versus 12 I/Os*

MQT as aggregates

Data aggregation and summaries can save a tremendous amount of I/Os and CPU. Make sure the aggregates and summaries are monitored to demonstrate and justify their creation.





Materialized Views

Another method of speeding analysis is through the use of Materialized Query Tables (MQTs) as aggregate data stores that specialize in a limited dimensional function. These MQTs are good for taking complicated join predicates and stabilizing the access path for end-users. Warehouse history can also be summarized into MQTs to provide standard comparisons for standard accounting periods or management reports.

Comparison Points

MQTs function best when defined to existing end-user comparison points. These aggregates can be used extensively for functions and formulas because of their totaled data

Intelligent Queries

Meta-data about aggregates must be well documented and published to all end-users and their tools. Tools should be aggregate aware and be able to include the different appropriate aggregate if needed.

MQTs and Horizontal Aggregation

Tracking query activity can sometimes point to special data requirements or queries that happen on a frequent basis. These queries may be totaling particular products or regions that could be optimized through a Materialized Query Table (MQT or horizontal aggregate).

Eliminate I/Os for answers

Analysis must be done to justify the definition of an MQT to make sure it is used enough. Like all aggregates, MQTs and DGTTs if used enough can eliminate I/Os and conserve CPU resources. MQTs aggregates work from a particular dimension key that is can be easily separated from the rest of the data.

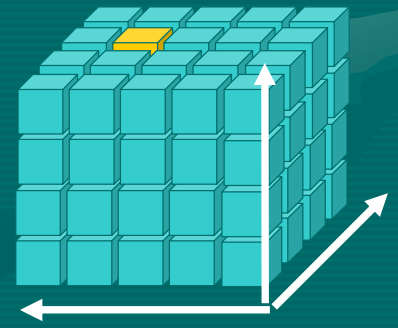
Declared Global Temporary Tables

- DGTT- Can be defined as needed
  - Limiting technique
  - Department or subject scope
  - More complex than a MQT
- Transition to permanent tables/MQT
  - Indexed
  - Isolated
  - Performance driven
- Useful situations

Product-ID	Time-ID	Region-ID	Cost	Items	Sales
1	1	1	25	4958	835
1	2	1	27	1234	876
2	3	1	19	3321	765
2	4	1	21	1134	730
2	1	1	23	5103	715
3	2	2	31	4908	934
3	3	2	28	3456	954
3	3	2	33	1443	987
3	4	3	32	2012	967
3	4	3	29	2849	931
4	1	3	24	1986	623

MDC- Multidimensional Clustering

- New type of table
  - Special MDC syntax
- Clustering along multiple dimension keys
  - **Possible to cluster on many different dimensions of a DW**
- Clustered equally across all dimensions
  - **Block Space management**
- New smaller Dimension Block indexes
- Faster loads, queries and updates



DGTT with tool processing and security aspects

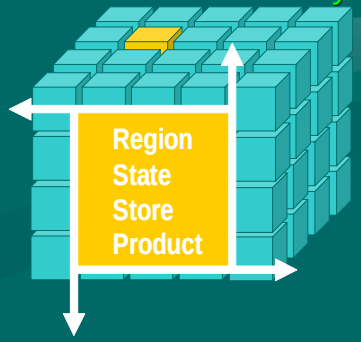
Care needs to be taken to include the GTTs information in all end-user tool information so it can be evaluated for query result sets. Sometimes global temporary tables can also be used to provide security against departments looking at other department's data. The GTTs or horizontal aggregate security technique is very effective and also maximizes query performance.

Another new feature with DB2 is the new patented clustering technique MDC - Multi-Dimensional Clustering. This feature does exactly as the name implies, it clusters the data against multiple dimensional keys. The MDC clustering is achieved by managing data row placement into page extent blocks based on their dimensional key values. The management, placement and access are facilitated through a new Version 8 Block Index object type. This new Block Index object type is created for each of the dimensions, is similar in structure to a normal index but cross references rows to a dimensional data block instead of an individual row.

The extent data page block sizes are chosen at Multi-Dimensional Clustering definition time and if additional space is needed consecutive block extents are defined. Since the rows are managed to a data block, the cross-referencing Block index information needed is smaller, resulting in a smaller index structure. With consecutive pages and the Block index only referencing data blocks, Block index reorganization will not be needed as often as a regular indexes referencing individual rows.

MDC- Multi Keys within new blocks

- Cluster keys create blocks of clustered pages
  - DB2 manages space via blocks of data
- Block index references groups of data
  - Block index is much smaller than regular Type 2 index
- Block indexed can be combined with regular Type 2 indexes
- Design planning for MDC Block density



MDC Considerations

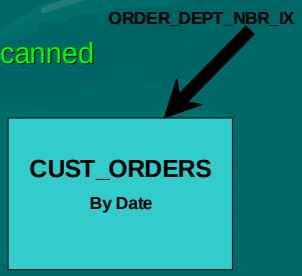
- Best used with < 7 dimensions
  - Must calculate the possible block population
    - 4 dimension: Region:5 State:50 Store: 250 Product: 10000
    - 5x50x250x10,000 = 625,000,000
  - How many rows go to each MDC Block?
- How many partitioning dimensions?
  - Can become a space problem
- Design Advisor: suggests - Generated columns?
- Utilities still needed?
  - REORG needed to reclaim space

Taking data placement management one-step further than partitioning, Multi-Dimensional Clustering groups the rows to the various dimensional key blocks ideally organizing the rows for data warehousing and OLAP application access. The Multi-Dimensional Clustering Block indexes can be used individually, combined with regular indexes and utilized in all the intra and inter parallelism optimizer access methods to quickly retrieve large amounts of data. Since the Multi-Dimensional Clustering blocks are defined and extended in consecutive page blocks, similar data is contained in consecutive pages making caching, pre-fetching, RID lists and accessing data that much quicker.

Clustering along multi dimensional keys also has tremendous value for regular insert, update activities also. With a regular table, the data is placed via a single clustering value and becomes un-clustered with more insert and update activity. Multi-Dimensional Clustering tables maintain their clustering continuously over time because the clustering is based on multiple clustering keys that point to data blocks instead of individual rows.

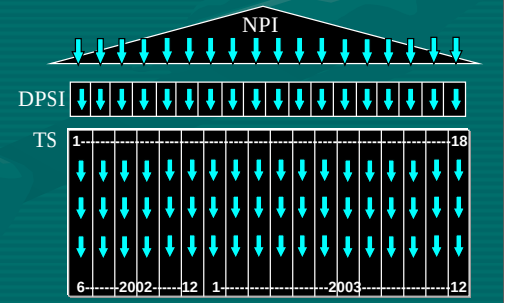
New Indexes Opportunities

- Separate clustering and partitioning indexes
  - Clustering does not need to the through partitioning index
  - Partitioning can be done in table definition DDL
    - PARTITION ENDING AT clause
- Cluster for biggest workload
  - Data load/inserts/maintenance
  - SQL activity usually ?10-25%? scanned
  - Compliment MQT aggregates



Data-Partitioned Secondary Index (DPSI)

- Must be NON-UNIQUE
- Eliminates the Build2 phase of utility processing
- Aligned with partitioning scheme
- DPSIs can be a performance considerations



Clustering for Performance

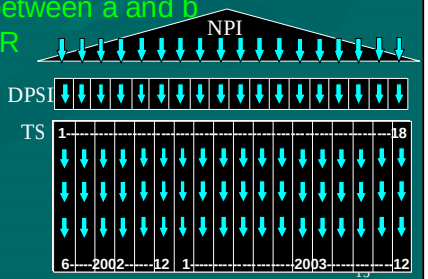
Eliminating and minimizing I/Os can make or break performance objectives. A prototype design should be put together and typical queries estimated. SQL Traces on the system during the user testing can help point out system issues. Clustering should be modeled against the most popular usage of the table and to eliminate as many sorts as possible.

DPSI indexes were implemented to eliminate the huge overhead and downtime associated with the utility BUILD2 phase or building a NPI index. The DPSI index structure is partition dependent potentially allowing the parts of the index to be built while its partition utility processing is being done.

DPSI index must be defined as non-unique. This causes the DPSI index to be potentially a poor performing index choice for DB2 because multiple DPSI partitions may need to be searched.

DPSI Considerations

- SELECT could scan all parts of a DPSI
  - SELECT FROM table WHERE PRODNBR=x
- ORDER BY
  - SELECT FROM table
    WHERE PRODNBR between a and b
    ORDER BY PRODNBR



Determine whether your DPSI design causes your SQL queries to search multiple partitions of the DPSI index to produce the result set.

Examples of potential DPSI performance opportunities.

Index Enhancements

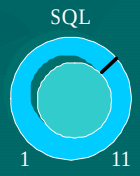
- Better comparison capabilities
  - Unlike Column and Host variable definitions
  - Character comparisons of unequal length
  - Decimal and integer comparisons
    - Especially important for Java, C#, .NET apps
- Better local and remote join operations
  - WHERE host= x + z
- EBCDIC and UNICODE Joins index-able
- Varchar padding or not
  - More index entries per page fewer pages/levels
  - Index only access available

DB2 Version 8 can now use an index when it is comparing variables of different data types and lengths. This ability allows more indexes to be used during join operations and when comparing host variables to db2 columns.

VARCHAR columns that are used in indexes now have the option of being not padded to their largest length. By reducing the index entry size, more index entries can be fit per DB2 page. This will reduce the number of index pages and potentially the number of levels within an index structure.

Index Enhancements

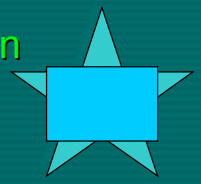
- More knobs - zParms
  - Size of SQL now 2M
  - Tables_join_threshold
  - MXTBJOIN – max tables joined default 225
  - MXQBCE - max combinations considered
  - Optimization resource thresholds
    - Max_Opt_Stor - RDS storage 20m
    - Max_Opt_CPU – CPU 100 seconds
    - Max_Opt_Elap – Elapse Time 100 seconds
  - SJMXPOOL – VPool to 1,024 – 20m



The sparse index capability reduces I/O associated with unqualified rows that might have been considered and then eliminated through a sort of a large result set. The new sparse index capability builds an index of only qualified rows. This eliminates the unqualified data early in the access path greatly improving query response time and CPU performance. Improvements in DB2 optimizer algorithms also determine the cost effectiveness of when the snowflake or raw data warehouse dimensions should be materialized. The improved algorithms help DB2 analyze the SQL and determine whether and when the access path would benefit from materializing the snowflake dimensions. All of these improvements can have a dramatic performance impact on your ever-increasing complex data warehousing front-end OLAP tool queries.

STAR Join Enhancements

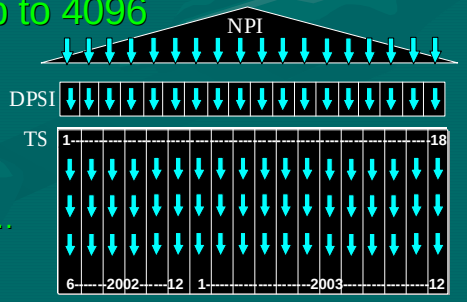
- Dynamic creation and use of sparse index
  - Binary search of index
  - Up to 240kb index
    - V7 PQ61458
- Dedicated Vpool for STAR Join work-file
  - Better utilization of memory
- Improve parallel sort optimization



Also new in Version 8 is the ability to create a dedicated virtual pool for star join workfiles usually created for materialized dimensions or composite data. By dedicating a data warehousing workfile virtual pool, repeated scanning or access to this work data is done in memory. Since this star join workfile pool is in addition to any other buffer pools, sort operations and query performance is improved.

Altering DBMS reality

- Change table/column attributes
  - Expand char columns
  - Numeric
  - Table Versioning
- More partitions up to 4096
  - Add or rotate partitions
  - More flexible designs
    - A partition a day…



Version 8 also provides a powerful data warehouse design alternatives with the expansion and rotation capabilities of DB2 z/OS table partitioning. In Version 8 the maximum number of partitions has been expanded from 254 to 4096. Having the capability to define a separate table partition for every day for over ten years is capability unique of all DBMSs to DB2 z/OS and remarkable design flexibility for data warehouses. Also having the ability to rotate partitions for implemented partitioned table designs provides the ability to enhance existing partitioned databases.

In Version 8 star join related features have been improved in several new ways. The most important features are the new sparse indexing capability, enhancements to the DB2 optimizer materialization and a dedicated virtual memory pool. These features along with new additional ZPARMs and the RUNSTATS statistical distribution and sampling capabilities directly help the DB2 optimizer choose the most efficient star join data warehouse table access path.

Partitioning parallelism reduces time

- Load Set Size
  - Determine I/O requirements per year/month/week/day
  - Formula = CPU ms + 2-20 ms per call
    - **5B per year = rows per month = 400,000,000 rows = 400,000 CPU seconds + 800,000- 8,000,000 I/O seconds**
      - ***Elapsed time 222 to 2,222 hours processing each month***
- **10 parallel weekly schedules**
  - **(2,222 / 4) / 10 = 55.55 hours**
  - **Same CPU requirements**
- **SQL Queries**
  - **Partitioning encourages query parallelism**

How many partitions?

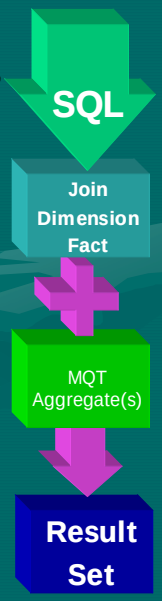
Determine how many partitions and parallel processing streams your CPU, I/O and network can support.

Do your calculations

Make sure you include all the various SQL validation and dimensional key translation work.

Parallelism – All Aspects

- All aspects
  - ETL, validation and transformation
  - Loading query and correction
- What keys provide parallelism?
  - Partitioning key only?
- How are MQT involved?
  - Can they be joined within a query



Summary

- DB2 SQL continues to lead the industry
  - Performance advantages of new SQL OLAP
  - Functions in z/OS "leapfrog" the competition
- MDC offer great opportunities
- MQTs offer HUGE opportunities
- Indexing continues to get better
- DPSI show great potential
- Partitioning & parallelism for performance

Extract, Transformation, Load and Maintenance

Parallelism designs for reducing the time windows for ETL, maintenance and end-user access processing should be weighed against each other to come up with the best overall design.

Partition Keys

When analyzing different partitioning key candidates, study the keys that are available to the various processes that maintain the warehouse. Try to find a key that is common among all the processes for the partitioning and data distribution scheme. The various processes can then be split via the partitioning key and run in parallel drastically cutting the overall processing time.

MQT - Aggregates Keys

Also verify that the same partitioning keys and scheme are available to the aggregate tables. This is critical to joining the tables effectively and efficiently.

Evenly distribute the data

The separation rules are very important because of the potential to evenly distribute the data. Distributing the data allows multiple entry points into the data based on the rules or keys that separate the data.

References

IDUG Papers DB2 UDB Version 8 (for z/OS & LUW)

DB2 UDB for z/OS Programming Functional Specification

DB2 UDB for z/OS V8.1 Administration Guide

DB2 UDB for AIX V8.1 Administration Guide

DB2 UDB for z/OS V8.1 SQL Reference

Approaches and Methodologies for Capacity Planning for Business Intelligence Applications SG24-5689

DB2 UDB for z/OS V7 Administration Guide

DB2 UDB for Unix, Linux and Windows V8 Administration Guide

Meet the Experts –DB2 OLAP Functions by Bob Lyle IDUG EU 2001