Virtual Partitioning in Oracle Very Large Data Warehouses

Brian Dougherty
Chief Architect
Business Intelligence
CMA Consulting

Agenda

- VLDWs Defined
- Large Fact Table “query problem spaces”
- Oracle 10gR2 Partitioning Solutions
- An Incomplete Arsenal for the VLDW Architect
- Virtual Partitioning – The Architectural Solution
- Virtual Partitioning Using Oracle 10gR2 Materialized Views
- Oracle 11g – Better But Still Short

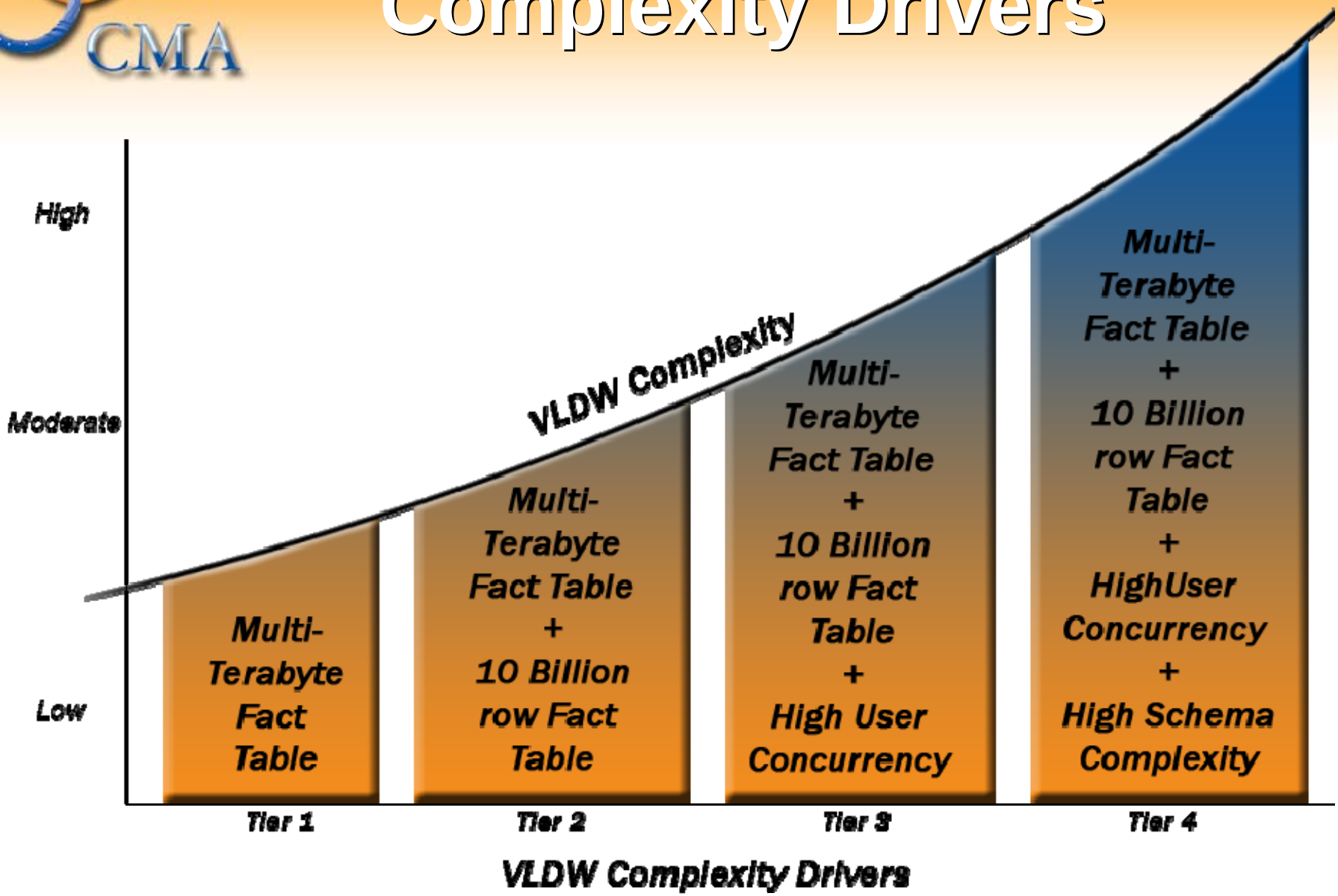
VLDWs Defined

- Everyone has a different definition
- Growing rapidly each year (5TB/10TB/50TB)
- Measured in terms of size ***and*** row count
- For VLDW architects, concurrent usage and dimensional model complexity add difficulty
- Hardware mitigates size ***to a point***

VLDWs Defined
Our Definition

- At Least One Multi-Terabyte Fact Table (1-5 TB uncompressed)
- At Least One Very Large Fact Table Containing between 1 and up to as high as 50 Billion rows
- Dimensional Model exhibits moderate to high schema complexity
- Querys are diverse and unbounded – many Fact Table “entry points/dimensional probes”

VLDWs Defined Complexity Drivers



Large Fact Table

Partition Pruning Query Problem Spaces

Query Profile
Query predicate includes column matching Single column partitioning key
Query predicate includes columns which match both or the higher order columns of a range/range partitioning key
Query predicates includes equality or in-list clause
Query predicate includes combination of range and equality or in-list

An Incomplete Arsenal for the VLDW Architect

- Lack efficient method for table to be ***simultaneously*** partitioned by ***two*** *or more* statistically independent columns

- In other words, 2 or “n” distinct or unrelated columns (unrelated with respect to how data values within the column domains occur) cannot be ***simultaneously*** defined

- Partition Pruning point of vulnerability

An Incomplete Arsenal for the VLDW Architect

- Composite and Multi-column range/range imply ***correlation*** between high and low order columns

- Optimizer partition pruning on low order must be qualified with high order column

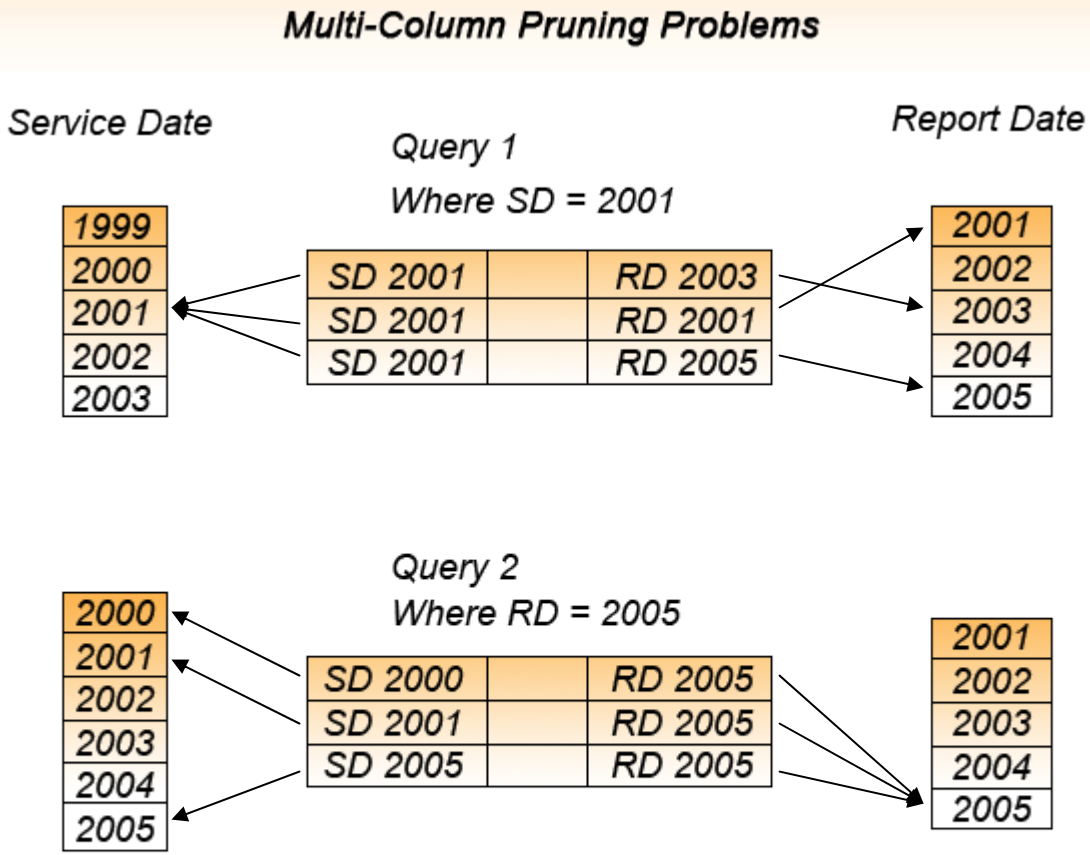
An Incomplete Arsenal for the VLDW Architect

- **What is Required for pruning to take place ?**

- Low order or “n”th order column pruning must be ***independent*** of the high order columns

- n independent inversion entries into the multi-terabyte fact table query space must be supported
- Inversion entry “code path” must be supported by direct path multi-block I/O

An Incomplete Arsenal for the VLDW Architect



Partitioning Scheme Multi – Column Range : (Service Date / Report Date)
Partitioning Scheme Composite Range / List : (Service Date / Provider)

An Incomplete Arsenal for the VLDW Architect

Hey Optimizer – Why Not Indexes ?

Two problems with indexes:

1. Ratio of column cardinality to row count
   - 5 billion row fact table renders all but extremely high cardinality to weak filter status
   - (10k providers / 5 billion rows) = 500,000 rows still qualify – 500k single block I/Os
   - Less than 10k distinct service dates make bitmap index on service date even worse
2. Oracle I/O semantics
   - Single block I/O vs. multi-block direct path I/O
   - 16k vs. 1MB per I/O request
   - ASM is kernelized asynch I/O
   - Parallel I/O is direct path multiplier

Virtual Partitioning
The Architectural Solution

- Virtual Partitioning is defined as the ability of the Oracle Optimizer to recognize and execute several or many independent segment pruning paths into the same table while leveraging direct path I/O code path

- Ultimate solution would store the 3 Terabyte Fact table *once* but provide several independent partitioning schemes

Virtual Partitioning

The Architectural Solution.. Continued

- A Less elegant solution provides for building and storing the same (column set) Fact table “n” times and associating ***a different partitioning*** scheme with each representation

- Intelligent query “router” required to associate the table collection (base table + MVs)

- Grain of derivative fact tables same as primary fact table (MV aggregates inflexible)

Virtual Partitioning

The Architectural Solution.. Continued

- Reduce disk space overhead by:
  - Oracle 10gR2 compression
  - Temporal Fragments
  - Column Fragments
  - Cascading Fragments
- **The final solution consists of several *virtual*, but not *actual,* physical partitioning schemes against the original underlying 3TB fact table**

VLDW Real World Example

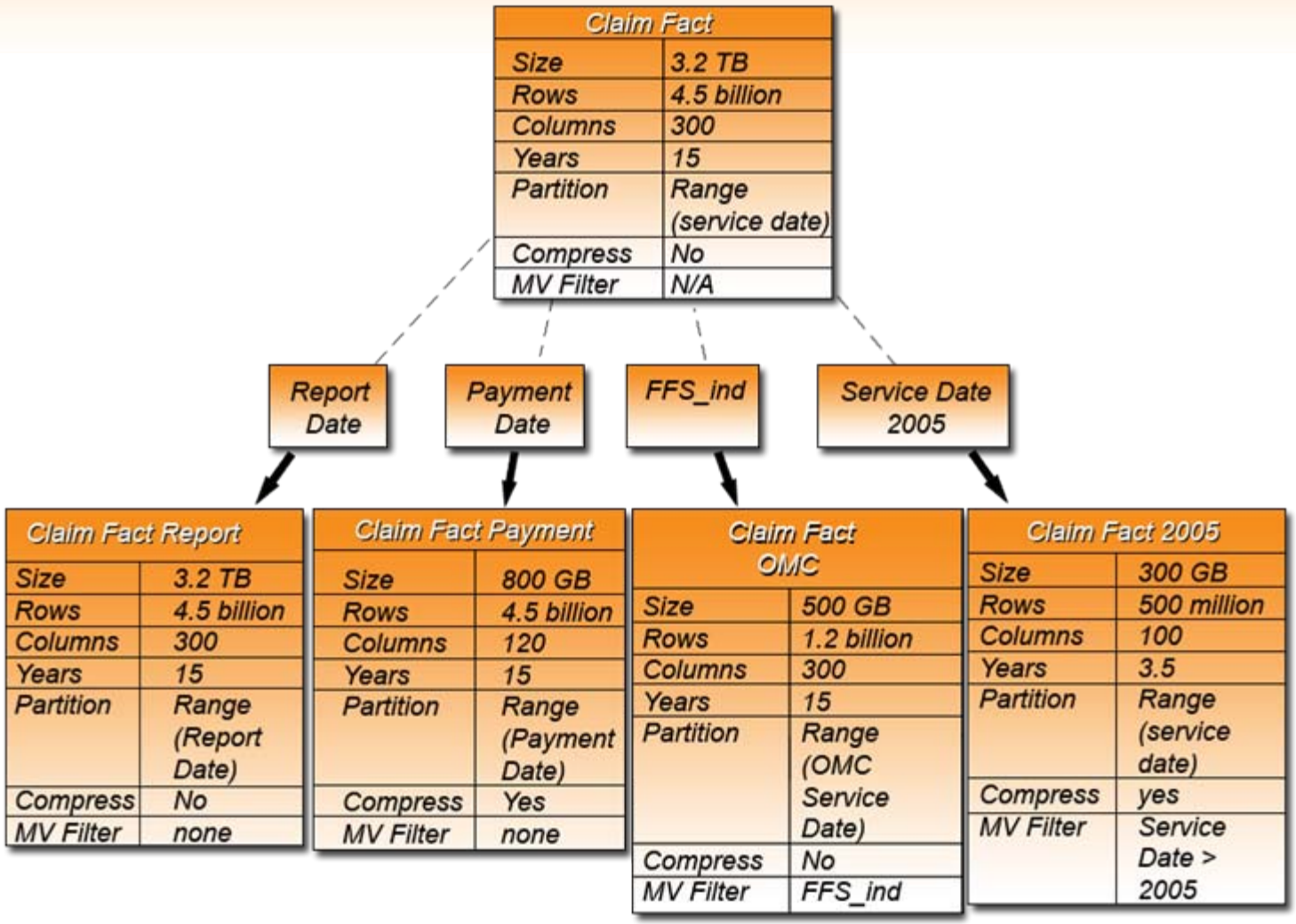
- New York State Medicaid Claims
- 3.2 terabyte and 4.5 billion row fact table
- 15 years of NYS Medicaid Claims
- At least 4 major dates to filter on within the fact tables
  - Service Date
  - Report Date
  - Payment Date
  - Managed Care Service Date

VLDW Real World Example ..continued

- All 4 dates have a loose statistical correlation at best. For example, ***payment*** for services may occur many months after the service date

- The four major dates support diverse user populations (e.g., managed care, prescriptions, fiscal planning, budgeting, etc.)

Virtual Partitioning
Conceptual Representation



Virtual Partitioning Using Oracle 10gR2

- Oracle 10gR2 Materialized Views and Oracle optimizer query rewrite functionality
- Provides order of magnitude query speedup for range based queries on multi-terabyte Fact Tables (versus full scan of base fact or indexes)
- Tradeoffs:

  An increase in disk storage

  Less than 100% query re-write reliability

  Additional fact table maintenance

Virtual Partitioning Using Oracle 10gR2
Implementation Steps

1. Create Primary Fact Table using an actual partitioning scheme against one of the range based filters (e.g., service date). Map to Tablespace set one to distribute I/O load.

2. Create 2nd, 3rd, ..nth, instantiation (replicate) of the Primary Fact Table. Define column set, filter and actual partitioning scheme of replicate table (This represents the nth virtual partitioning scheme with respect to the base table). Map to Tablespace set two thru n

Virtual Partitioning Using Oracle 10gR2 Steps.. continued

3. Register the 2nd, 3rd, nth instantiation of the primary fact table as an Oracle 10gR2 Materialized View with **query rewrite enabled**.

   Allows queries submitted against primary fact table to be candidates for transparent re-write

4. Determine optimal data population scheme for replicate tables (ETL Load and MV registration vs. MV create within Data warehouse)

Virtual Partitioning Using Oracle 10gR2 Example 1

NYS Department of Health – 20Terabyte DW

IBM P690 and EMC DMX3 – Very Large SMP

Large Fact Table : CLAIM_FACT

Virtual Partitioned Table: CLAIM_FACT_PAYMENT

CLAIM_FACT: 4+ billion rows and 2.1 terabytes**

** At time of whitepaper, currently is 3.2TB

Query Type	Query	Start Partition	Stop Partition	Elapsed Seconds	Segment Actually Scanned
Full *non-partition pruned* **service date** based range scan	Select /*+ full(c) parallel (c, 64) */ count(*) from CLAIM_FACT c	1	180	1820	CLAIM_FACT
Full *partition pruned* **service date** based range scan	Select /*+ full(c) parallel (c, 64) */ count(*) from CLAIM_FACT c where **service_date** between '01-JAN-2005' and '31-DEC-2006'	160	172	265	CLAIM_FACT
Full *non-partition pruned* **payment date** based range scan *Forcing no rewite with the No rewrite hint*	Select /*+ full(c) parallel (c, 64) **no_rewrite** */ count(*) from CLAIM_FACT c where **payment_date** between '01-JAN-2005' and '31-DEC-2006'	1	180	1815	CLAIM_FACT
Full *partition pruned* **payment date** based range scan using Virtual Partitioning *Same query as above but allow query re-write*	Select /*+ full(c) parallel (c, 64) */ count(*) from CLAIM_FACT c where **payment_date** between '01-JAN-2005' and '31-DEC-2006'	164	176	280	**CLAIM_FACT_PAYMENT Oracle 10gR2 Materialized View**

Virtual Partitioning Using Oracle 10gR2
Example 2

CMA MicroTerabyte BI Appliance

Dell/EMC CX3 2/4/8 Node RAC Cluster

Qlogic Unified Infiniband Fabric (Interconnect + I/O)

Large Fact Table : CHARGE_FACT

Virtual Partitioned Table: CHARGE_FACT_CDATE

CHARGE_FACT: 44 billion rows and 2.1 terabytes

Query Type	Query	Start Partition	Stop Partition	Elapsed Seconds	Segment Actually Scanned
Full *non-partition pruned* **ppyr** based range scan	Select count(*) from CHARGE_FACT	1	250	1080	CHARGE_FACT
Full *partition pruned* **ppyr** based range scan	Select count(*) from CHARGE_FACT where **ppyr** between 200501 and 200601	125	150	108	CHARGE_FACT
Full *non-partition pruned* **charge date** based range scan *Forcing no rewite with the No rewrite hint*	Select /*+ **no_rewrite** */ count(*) from CHARGE_FACT where **charge_date** between 20050101 and 20060101	1	250	1090	CHARGE_FACT
Full *partition pruned* **charge date** based range scan using Virtual Partitioning *Same query as above but allow query re-write*	Select count(*) from CHARGE_FACT where **charge_date** between 20050101 and 20060101	60	71	115	**CHARGE_FACT_CDATE** **Oracle 10gR2 Materialized View** **Range Partitioned on CHARGE_DATE**

Finally, What About 11g

- Optimizer enhanced to support two-dimensional column independent partition pruning (range/range composite)
- Low order column, pruned independently of high order in Composite Scheme (In-list hash)
- 3rd, 4th ..nth statistically independent column still not supported
- Look to 11gR2 ….

Wrap - Up

- Email : bdougherty@cma.com
- CMA MicroTerabyte BI Platform
- www.cma.com
- Questions