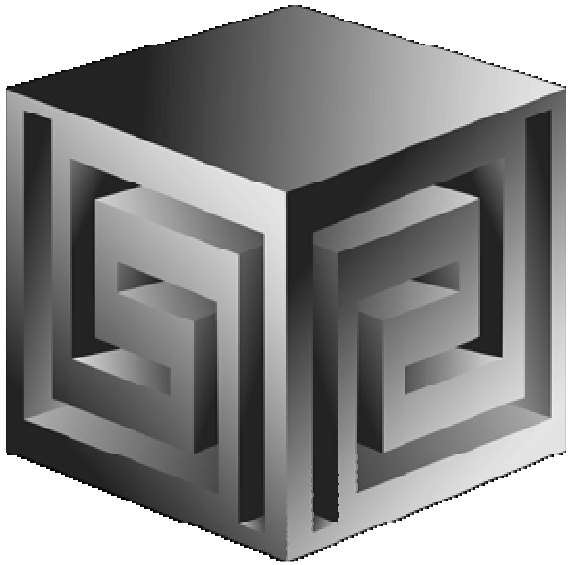
Oracle Business Intelligence in the Real World: Case Studies from the Trenches

ODTUG 2006



Dan Vlamis

dvlamis@vlamis.com

Vlamis Software Solutions, Inc.

816-781-2880

http://www.vlamis.com

Vlamis Software Solutions, Inc.

- Founded in 1992 in Kansas City, Missouri
- Oracle Partner and reseller since 1995
- Specializes in ORACLE-based:
  - Data Warehousing
  - Business Intelligence
  - Data Transformation (ETL)
  - Web development and portals
  - Express-based applications
- Delivers
  - Design and integrate BI and DW solutions
  - Training and mentoring
- Expert presenter at major Oracle conferences

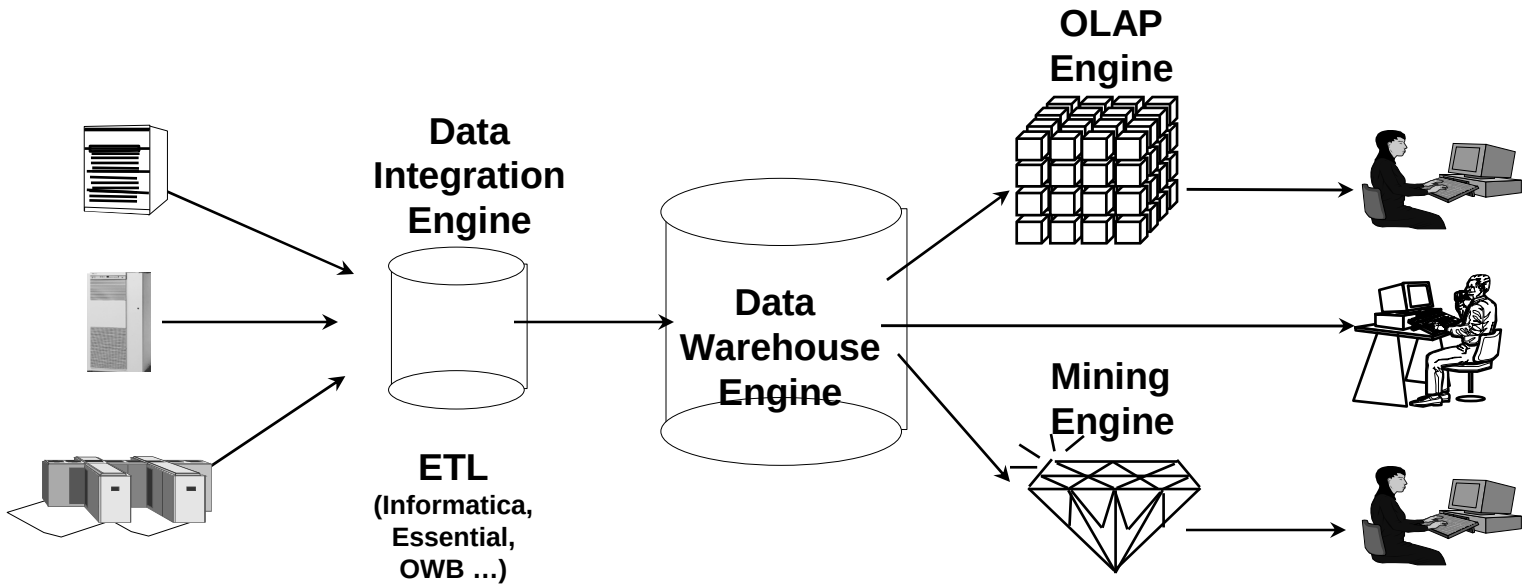
Who Am I?

- **Dan Vlamis, President of Vlamis Software**
  - **Developer for IRI (former owners of Express)**
  - **Founded Vlamis Software in 1992**
  - **Beta tester and early adopter of Oracle OLAP**
  - **Expert speaker and author**
  - **“Techie” on OLAP DML**
  - **Recognized expert in Express and OLAP industry**

Agenda

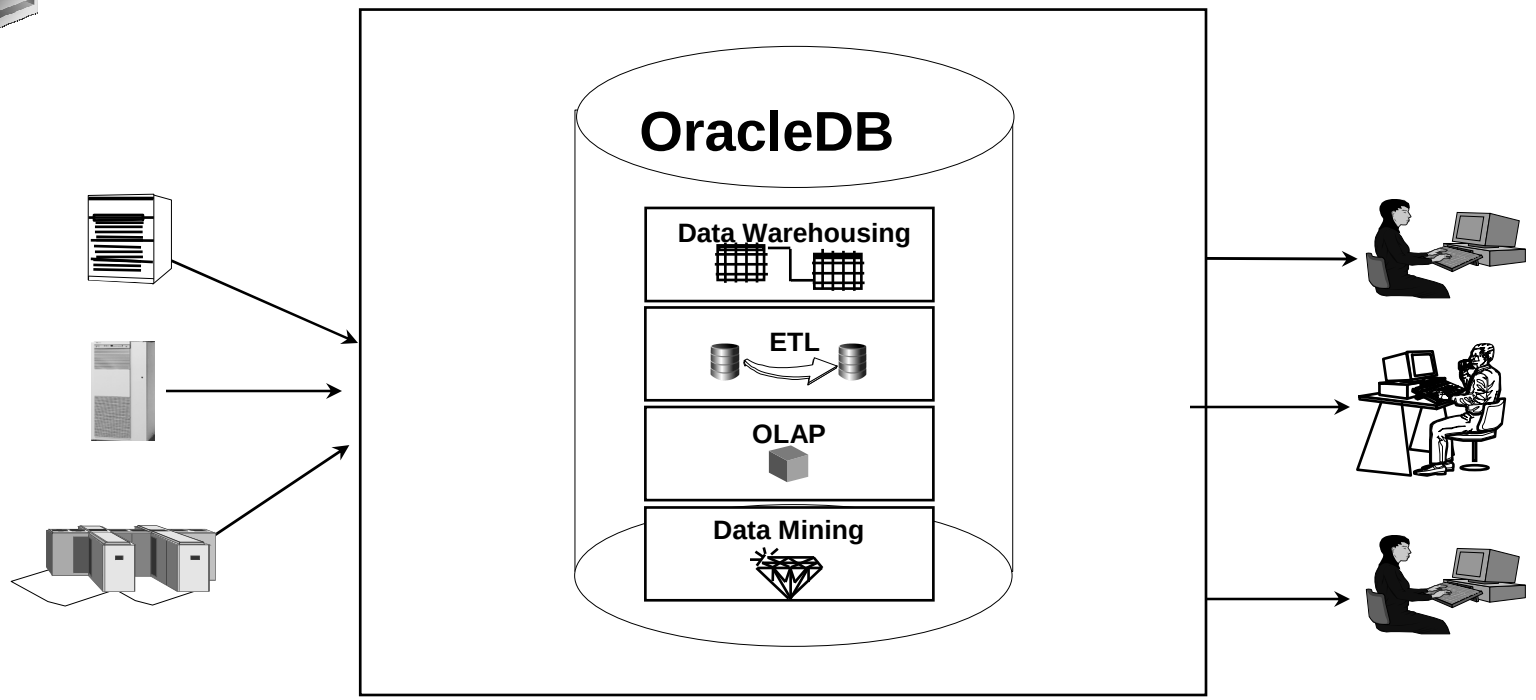
- **Create an OLAP Cube in 5 Minutes**
- **Why Oracle BI?**
- **How Oracle BI Been Used - Case Studies**
- **Specific Tips on Using Oracle BI derived from case studies**

Business Intelligence the Old Way



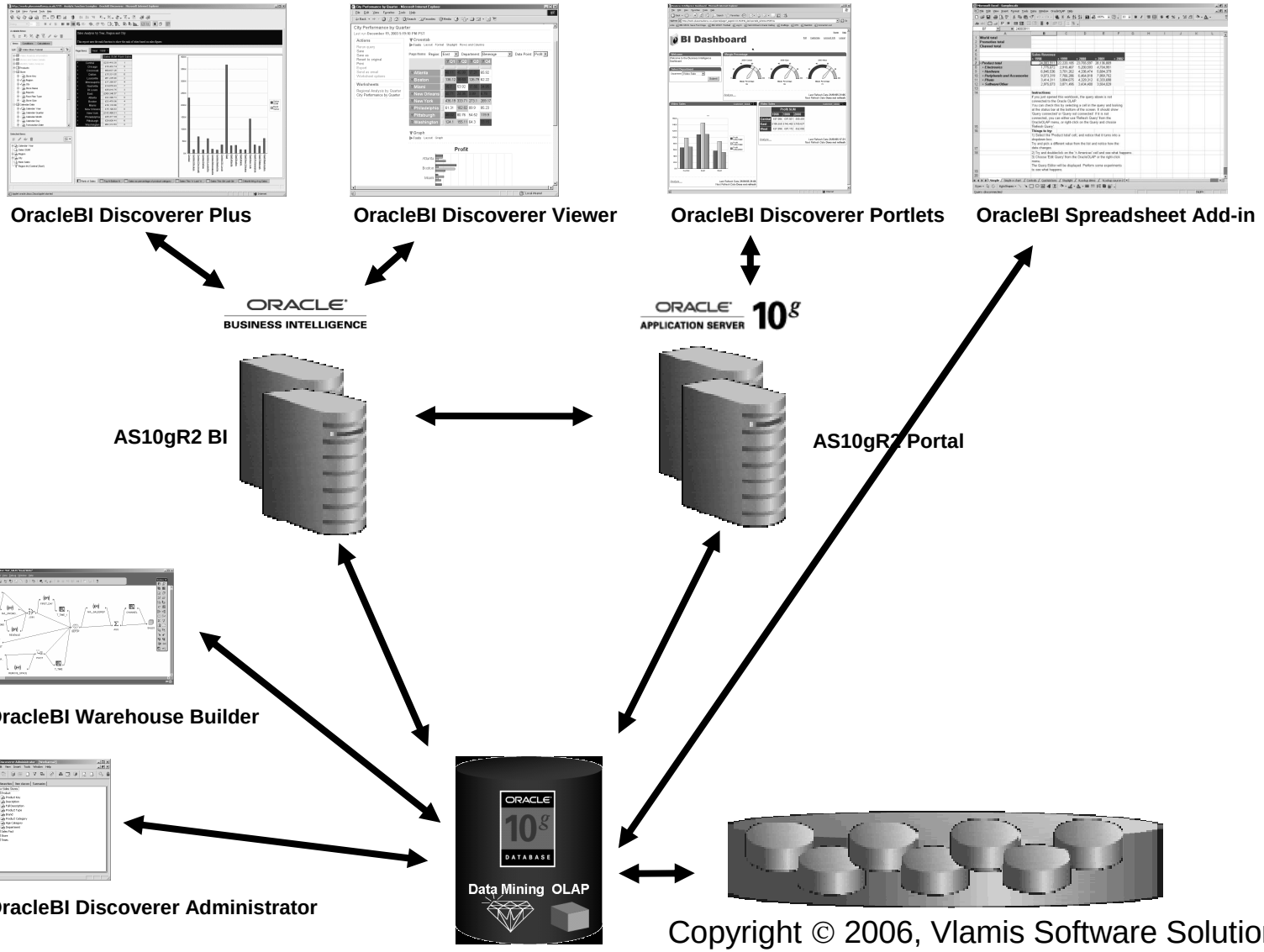
- Special purpose engines for differing tasks
- Metadata migration tools ease replication
- User interfaces generally different for different tools

BI the New Way: Oracle DB



- Single business intelligence platform
  - –Reduce administration, implementation costs
  - –Faster deployment & Improved scalability and reliability

Oracle BI Product Architecture



Definition of OLAP

OLAP stands for On Line Analytical Processing**.**
That has two immediate consequences: the *on line* part requires the answers of queries to be fast, the *analytical* part is a hint that the queries itself are complex.

i.e. Complex Questions with FAST ANSWERS!

Why a Separate OLAP Tool?

- Empowers end-users to do own analysis
- Frees up IS backlog of report requests
- Ease of use
- Drill-down
- No knowledge of SQL or tables required
- Exception Analysis
- Variance Analysis

What Does Oracle OLAP Add to a DW?

- Multidimensional user view of data
- Users create own reports
- Users create own measures
- Easy drill-down, rotate
- Iterative discovery process (not just reports)
- Ad-hoc analysis
- Easy selection of data with business terms
- OLAP DML with what-if, forecasting
- Platform for extensions

OLAP Option – High-level View

- Advanced analytics
- Integrated in RDBMS
- Easy to develop
- Easy to use
- Facilitate collaboration
- Flexible deployment
- Scaleable and performant
- True Relational – Multidimensional database

OLAP Option – Technical View

The OLAP Option consists of five key elements:

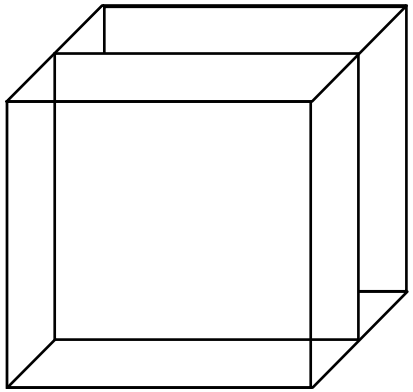
1. **Multidimensional data types, used for holding cubes and dimensions, temporary or stored permanently in LOBs within schemas**
2. **A multidimensional calculation engine**
3. **A Java development framework with reusable OLAP components**
4. **Extensions to SQL to allow SQL access to these multidimensional datatypes**
5. **An additional layer of OLAP-specific metadata known as the OLAP Catalog**

AWs Allow for What-if

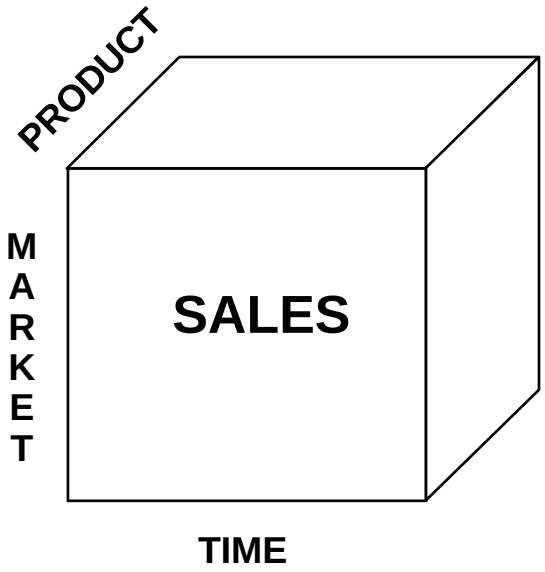
- **Modeling organizational changes**
  - **territory realignments**
  - **product hierarchy changes**
- **Product new launches**
  - **model new products after established product**
- **Forecasting**
  - **multiple scenarios**
  - **personal overrides of forecast**
  - **spread down of higher-level overrides**
  - **locks**

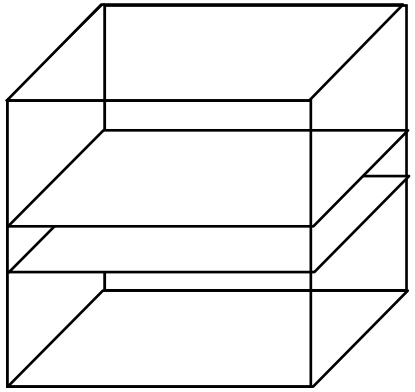
OLAP AW Stores Data in Cubes

Fast Flexible Access to Summarized Data

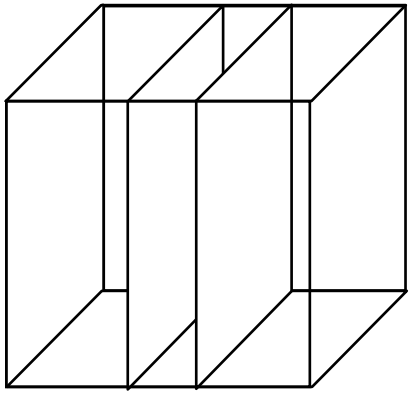


Product Mgr. View

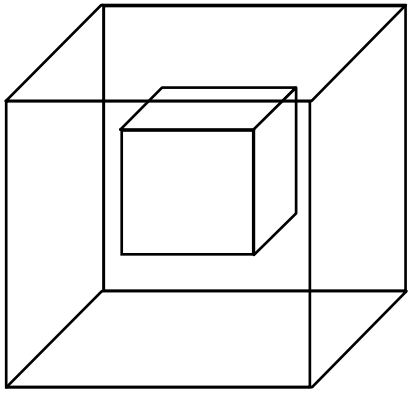




Regional Mgr. View



Financial Mgr. View



Ad Hoc View

Cubes Defined

- **Definition:**
- **Cubes are collections of measures. They are a logical way to organize data. All measures in a cube share the same dimensionality**
- **Examples:**
  - **Sales_Cube (with Units, Dollars, Profit)**
  - **Finance_Cube (with Actual, Budget, Variance)**

What Are AW Cubes?

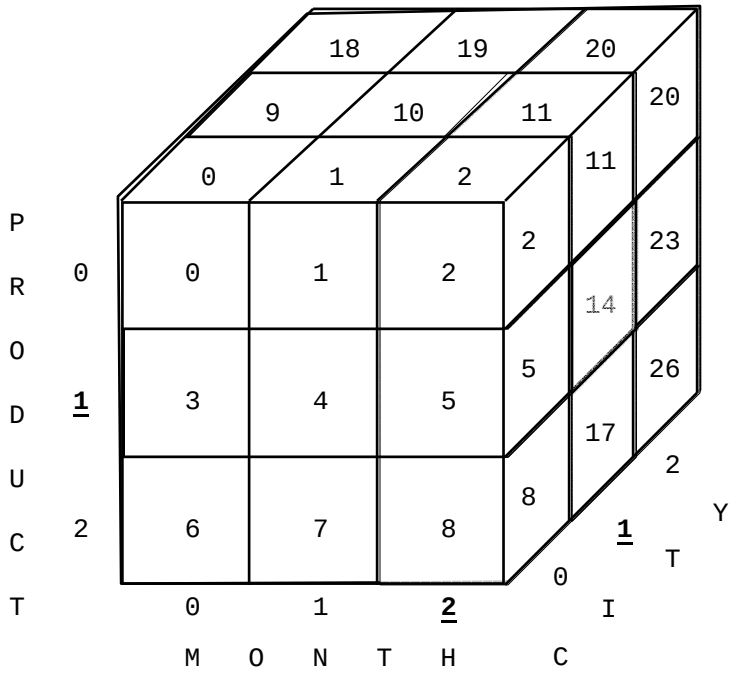
- Data stored as arrays
- Dimension values are internally integers
- Offset calculated using simple multiplication
- Offset tells exactly where to look for data
- Pages and segmentation complicate design
- Conjoints and composites handle sparsity

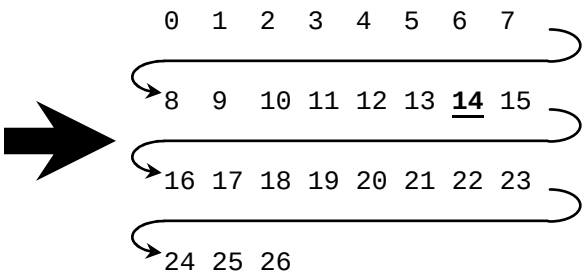
Finding data is simple multiplication and addition in an AW

Formula for calculating cell offset:

month + product * (# of months) + city*(# of months * # of products)

2 + **1** * (3) + **1** *(3 * 3) = 14

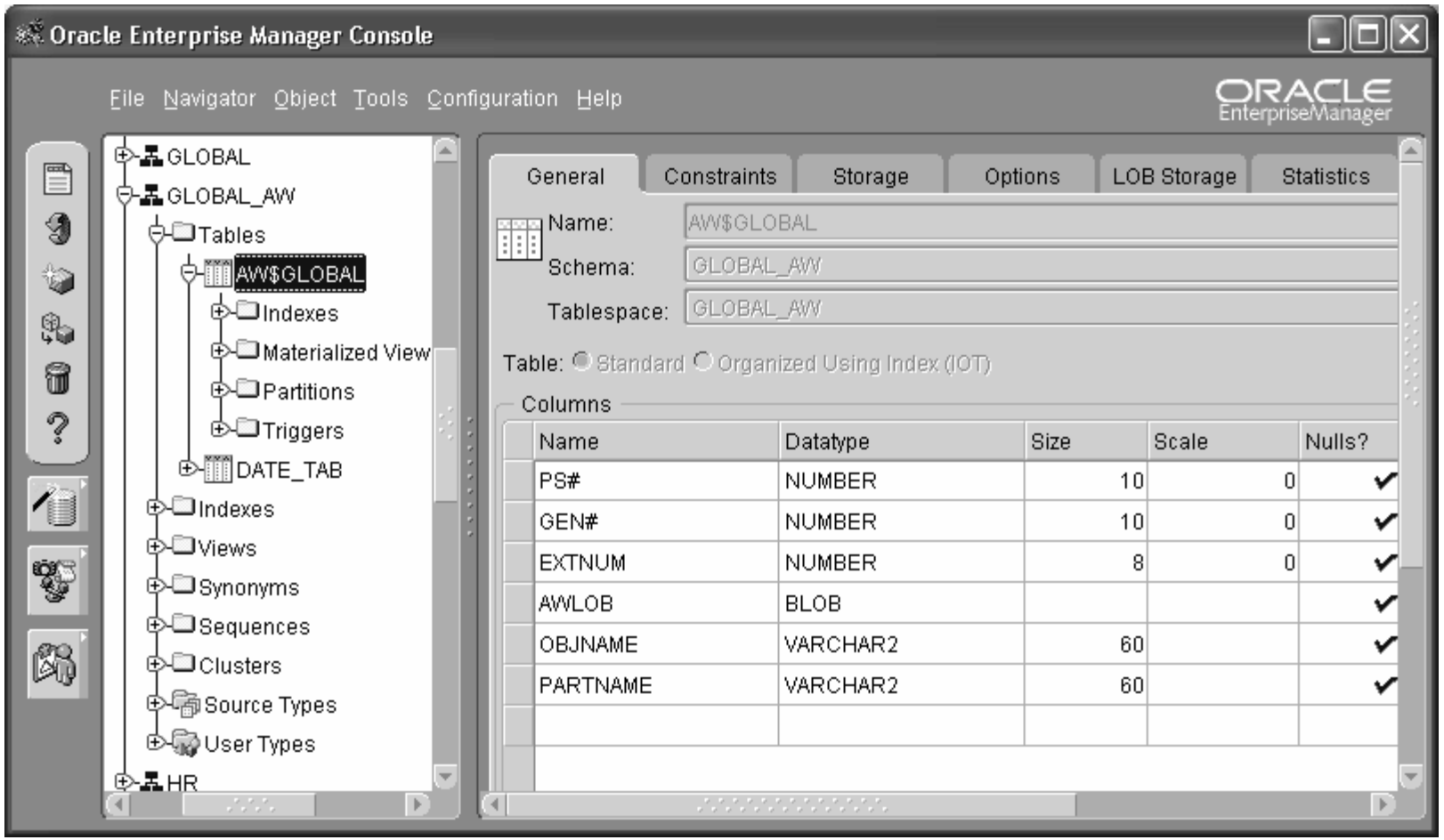




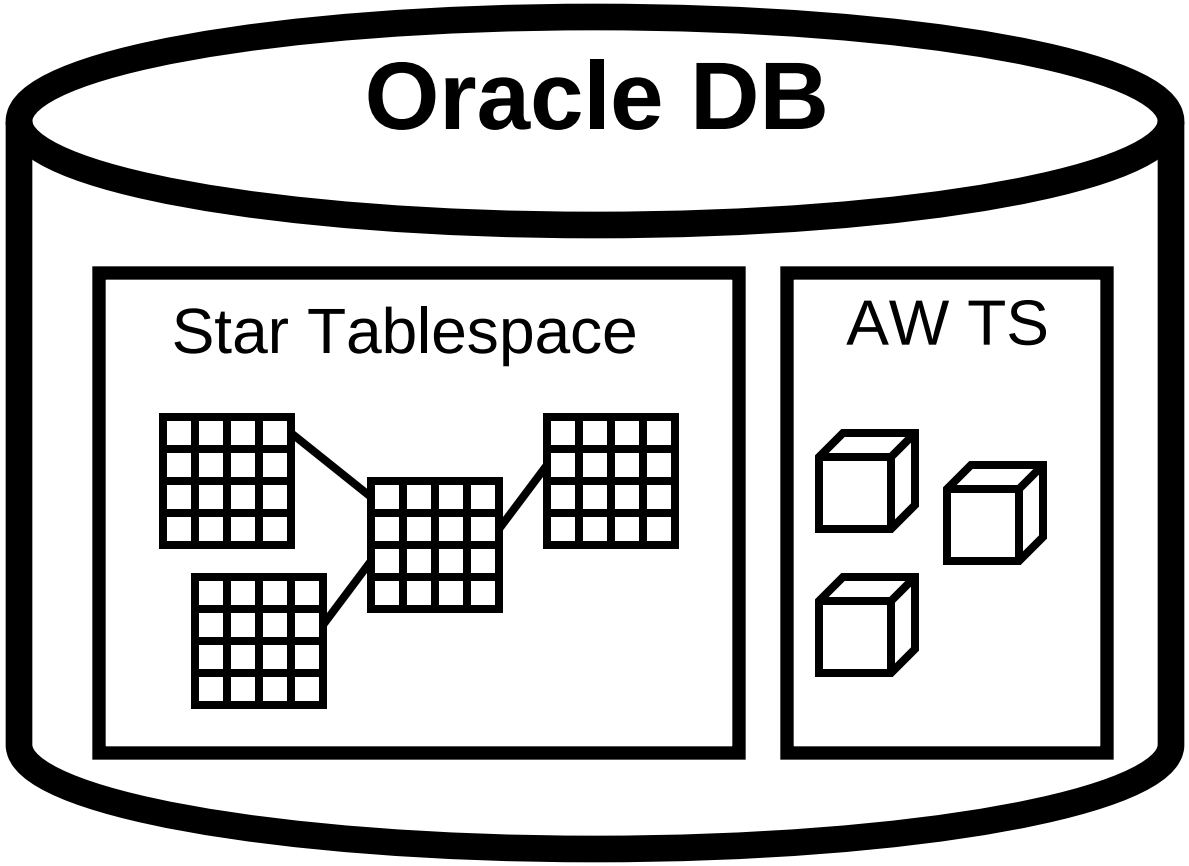
Offset 14 * 8 bytes each = 112.
Fseek to byte 112 to find data.

See http://www.vlamis.com for
"How Does Express Really Work Anyway" for details.

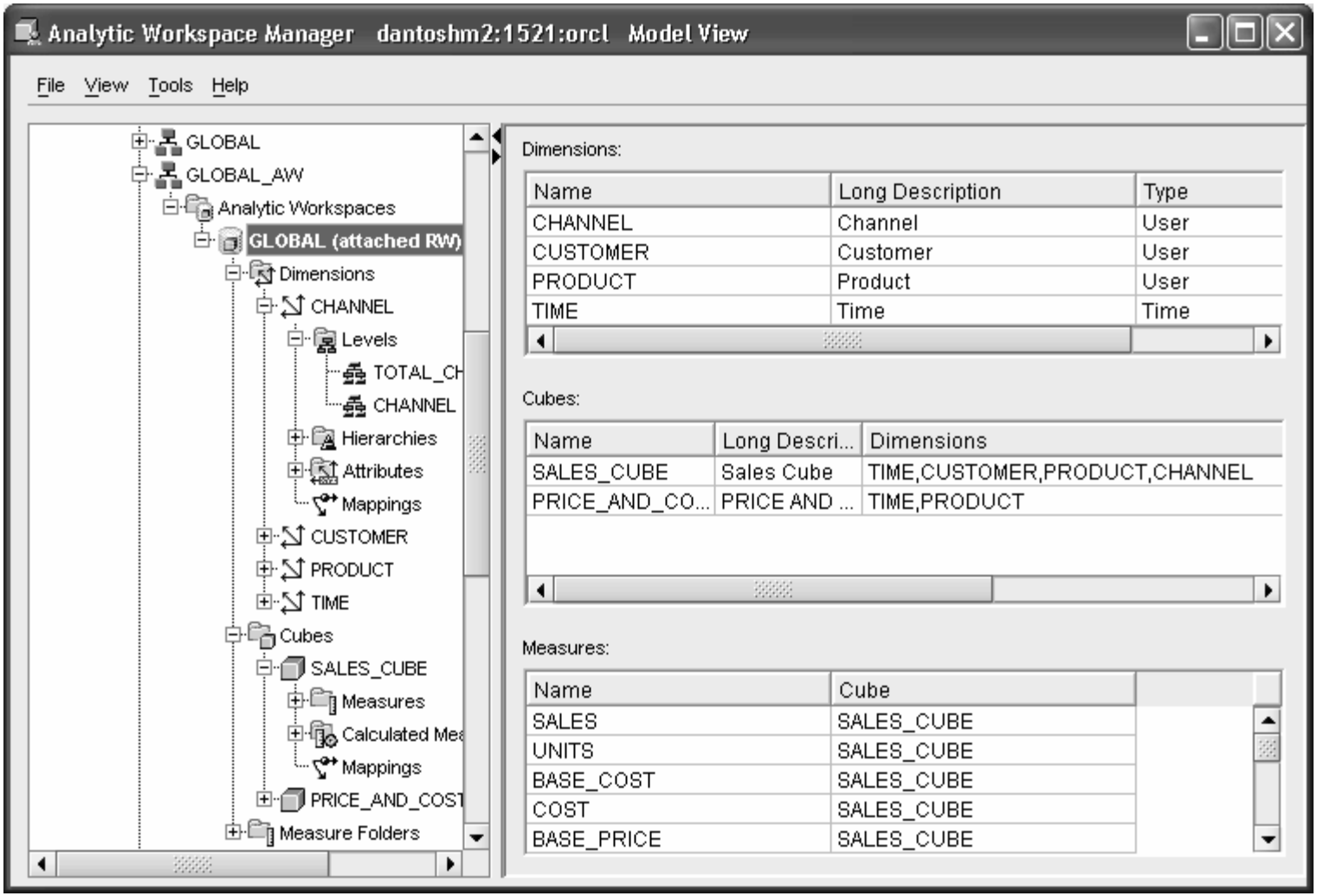
What is an Analytic Workspace?



Analytic Workspaces Are Stored in Tablespaces in OLAP



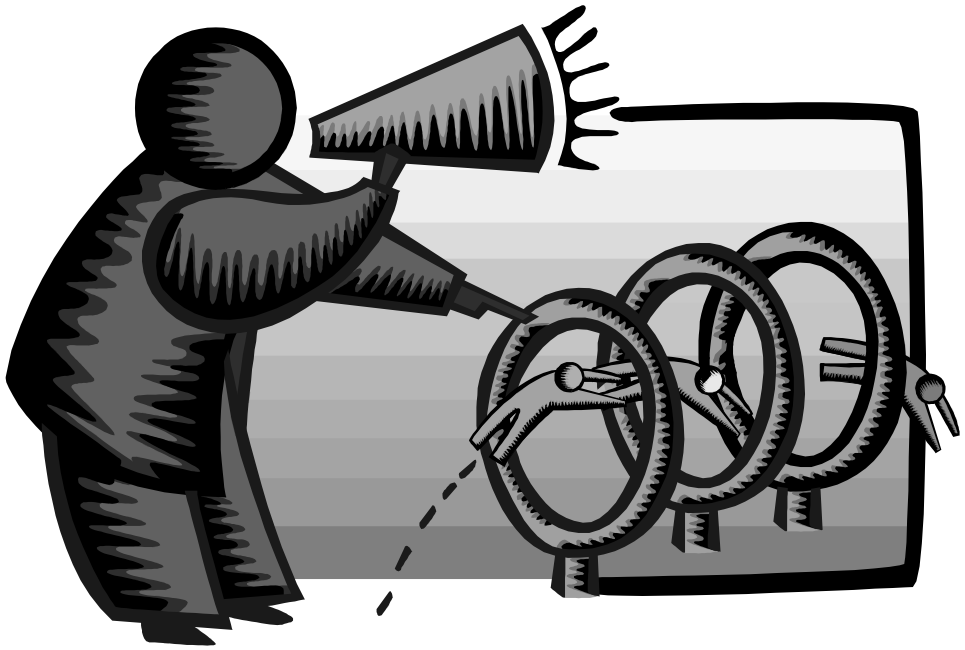
Managing Analytic Workspaces



AWs Allow for Rules Based Apps

- OLAP DML for manipulating data in DB
- Aggregation
- Allocation
- Decision Trees
- Transform data via complex OLAP DML logic
- High-power statistical functions built-in
- Rules or logic that differs by organization
- Expert systems

Building Cubes in AWM



See November / December 2005 Oracle Magazine for 4-page article "Use Oracle AWM 10g to build analytic workspaces" with details

AWM Cube Builder Tips

- **Remember to save Everything to XML files**
- **Remember this is Realtime…. So changes are nearly immediate (may need to reload data)**
- **Use “View” to see results in tool – No Need for BI Beans to validate success!**
- **Move Measures to Folders**
- **Can save Calculated Measures to XML – Then you can Edit!**

What Access Tool?

- Java OLAP API designed for products
- Discoverer for ad hoc analysis
- BI Beans for custom applications (using JDev)
- Spreadsheet Add-in for access from Excel
- Oracle Reports for highly formatted reports
- Oracle Apps for analysis of Apps data
- 3rd Party tools fill in gaps

Oracle OLAP Case 1

- Manufacturing company needs to reduce inventory levels
- Uses OLAP DML Forecast command based on orders
- Users can override forecasts and add their own promotional campaigns
- Computes more accurate forecasts of production needs, reducing inventory levels
- Can compare accuracy of monthly forecasts by comparing various “scenarios” each month with actual shipments
- Application presented as JSP for business forecasters / managers

Oracle OLAP Case 2

- Service organization with call center wants to minimize hold time but not increase headcount
- Solution is to analyze hold time and customer resolution time for each support analyst
- Can rank support engineers / departments by customer satisfaction / resolution / callback rates
- Can pay bonus based on quantifiable results

Oracle OLAP Case 3

- Oil company has complex GL and existing Express-based “business rules engine” for allocating costs and income
- Uses Oracle OLAP engine to develop models to allocate data based on rules analysts develop
- Users can develop their own way of analyzing the data rather than relying on IT
- IT sets up infrastructure, users develop actual analyses

Oracle OLAP Case 3 (continued)

- Company has existing Express application that meets user needs, but wants to modernize U/I and run with web interface
- Export/import existing Express databases to Oracle OLAP AWs
- Back-end code works as-is
- Front-end code rewritten in Oracle OLAP Web Agent (OLAP DML)
- "Application Generator" allows business users to create entirely new applications with their own multi-dimensional objects

Oracle OLAP Case 4

- Manufacturer wants an ad-hoc analysis and reporting against sales data warehouse
- Users need easy-to-use interface and limited custom analysis capabilities
- Front-end is BI Beans custom JSP with cross-tabs customized for user needs
- "Custom selector" allows users to select data
- Highlights importance of "returns"
- Daily data allows managers to impact EOM numbers
- Company changing business practices now

Oracle OLAP Case 5

- CPG company has existing Oracle Sales Analyzer implementation
- Company wants to explore using OracleBI to update technology
- Created Proof-of-concept dimensional model in less than 40 hours
- Demonstrated two techniques:
  - Export out data and import into Oracle OLAP
  - Use AWM to map to star schema data warehouse
- Company evaluating Discoverer OLAP

Oracle OLAP Case 6

- Shipping company wants to flexibly report data with many custom calculations
- Company used to multidimensional tools, but wants solution integrated with Oracle
- Many users accustomed to Excel
- Company wants training, but ends up needing consulting to get going
- Company now creating cubes on their own, using Excel add-in as their front-end of choice

Oracle OLAP Case 7

- Financial analysis company wants to analyze stocks against benchmarks using proprietary models
- Presentation of data is by various attributes of Equities such as Market Capitalization, Industry, etc.
- Users want to drill from groups of stocks to individual equities, changing dimensionality
- Custom OLAP DML code transforms data with models when copying from one cube to another

Oracle OLAP Case 8

- ASP Company using Oracle OLAP to deliver analysis of web traffic to clients
- Building separate AW for each client
- Uses templates to share common "dimensions" across multiple implementations
- Each client gets separate AW so each can customize dimensional model to their needs
- Building ASP offering around Oracle BI/OLAP

OLAP Design Tips

- Eliminate duplicate keys across levels yourself (e.g. Terr 5 vs. Division 5) by concatenating level-based text in ETL
- Can use remote DB link to grab data from other versions of Oracle
- Use true "keys" for data so users can save presentations across DB loads
- Avoid creating "too many" (>7?) dimensions – problems in presenting data from technical and user perspective

Schema Best Practices

- **Separate Schemas for ROLAP and AW cubes**
  - **i.e. DEV_DW = ROLAP & DEV_AW = MOLAP**
  - **Allows for better security (restrict access to "_AW" objects)**
  - **Better backup and restores (exp exports AWs also!)**
- **Separate Tablespaces for AW(s)**
- **Backup AWs separate from DB Backup (extra measure of safety)**
- **Optional separate Schema for Code (procs, packages, Code AWs)**

Natural vs. Surrogate Keys

- **Remember: dimension values in an AW must be unique.**

- **Natural keys:**
  - **Created in the AW as is from the source table (except numerics become text).**
  - **Examples:**
    - **1, 2, 3**
    - **Jan.2004, Feb.2004, Mar.2004, Q1.2004**

- **Surrogate keys:**
  - **Level is prepended to the source table id value**
  - **Examples:**
    - **ITEM_1, ITEM_2, ITEM_3**
    - **MONTH_Jan.2004, MONTH_Feb.2004, MONTH_Mar.2004, QUARTER_Q1.2004**

The Term "Surrogate" Has Other Meanings

- **In Data Warehousing / relational schemas:**
  - **Use of dummy, usually numeric keys in place of longer, usually alphanumeric keys to speed up joins, searching.**
- **In an AW:**
  - **An AW object which contains alternate key values of a dimension.**
  - **Analogous to a relational surrogate key.**
- **In AWM 10g:**
  - **Dimension values derived from a source table key column by prepending the level.**
- **Your users will want to refer to dimension values by their keys! Something must remain consistent!**

Handling a Snowflake-based Dimension

ITEM_ID	ITEM_DSC	ITEM_PACKAG...	FAMILY_ID
13	Envoy Stan...	Laptop Val...	4
14	Envoy Exec...	Executive	4
15	Envoy Amba...		4
16	Sentinel S...		5
17	Sentinel F...		5

FAMILY_ID	FAMILY_DSC	CLASS_ID
4	Portable PCs	2
5	Desktop PCs	2
6	Operating ...	3

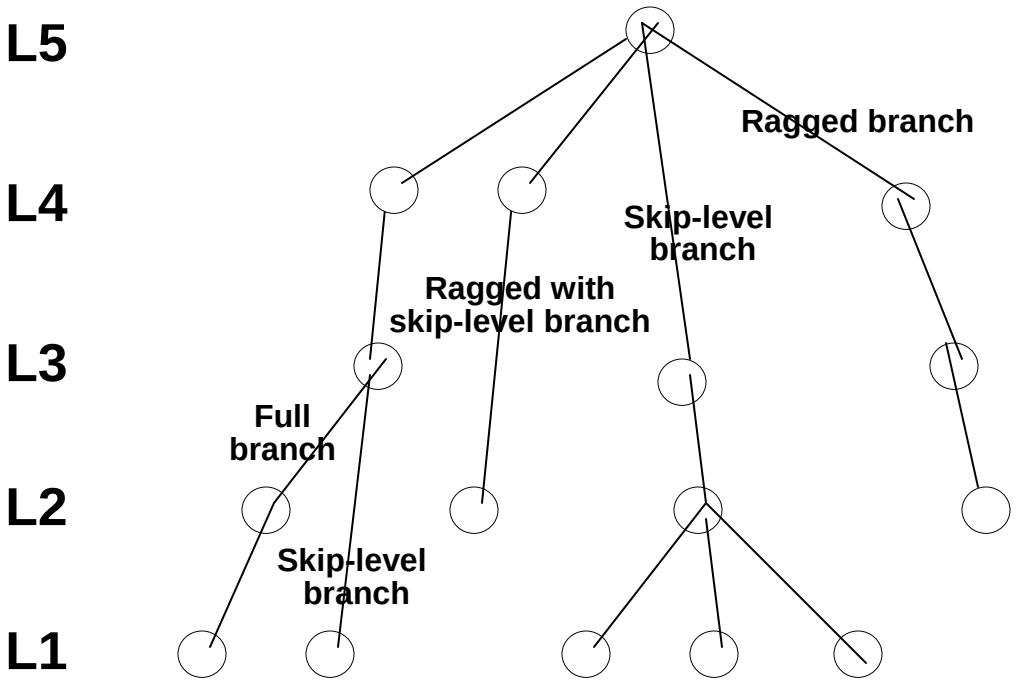
CLASS_ID	CLASS_DSC	TOTAL_PROD...
2	Hardware	1
3	Software/O...	1

TOTAL_PROD...	TOTAL_PROD...
1	Total Product

- **Natural or surrogate keys allowed**
  - **Must use surrogate keys if dim values are not unique across levels.**
- **Level-based hierarchy**
- **Snowflake mapping**

Skip-level vs. Ragged

- **Ragged is a special case of skip-level**
  - **Skipped level or levels is (are) the lowest level(s).**



AWM Cube Builder Tips

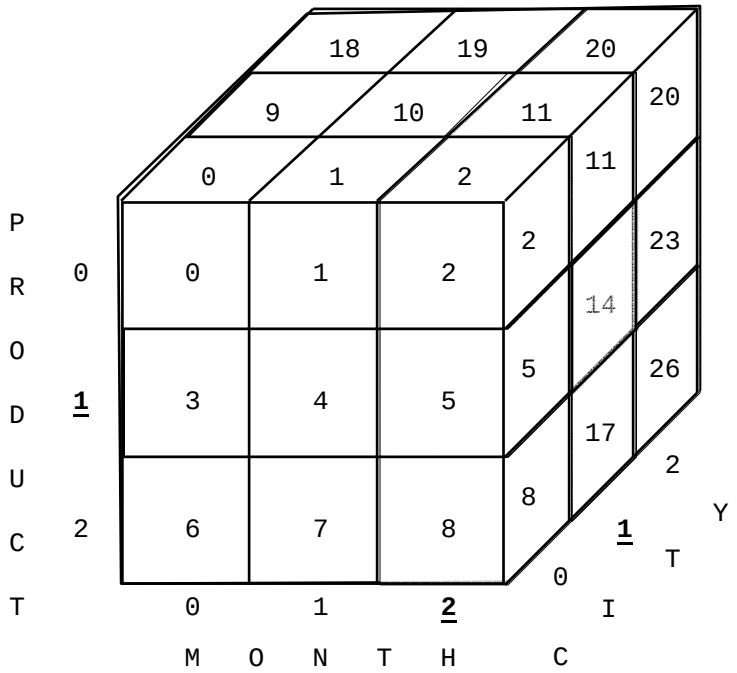
- **Remember to save Everything to XML files**
- **Remember this is Realtime…. So changes are nearly immediate (may need to reload data)**
- **Use “View” to see results in tool – No Need for BI Beans to validate success!**
- **Move Measures to Folders**
- **Can save Calculated Measures to XML – Then you can Edit!**

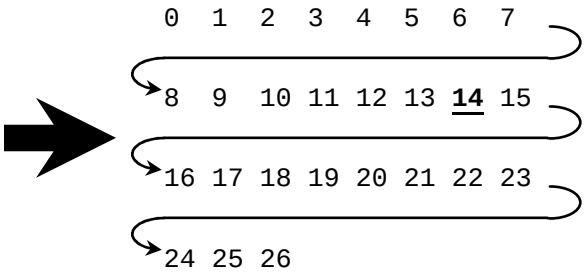
Dense Data in Cubes

Formula for calculating cell offset:

month + product * (# of months) + city*(# of months * # of products)

2 + **1** * (3) + **1** *(3 * 3) = 14





Offset 14 * 8 bytes each = 112.
Fseek to byte 112 to find data.

See http://www.vlamis.com for
"How Does Express Really Work Anyway" for details.

Sparse Multidimensional Data

Dim2

Dim1	T	U	V	W
D	na	na	na	65
E	na	na	35	na
F	na	20	na	na
G	10	na	50	na

Multi-dimensional measure

(16 cells)

Composite

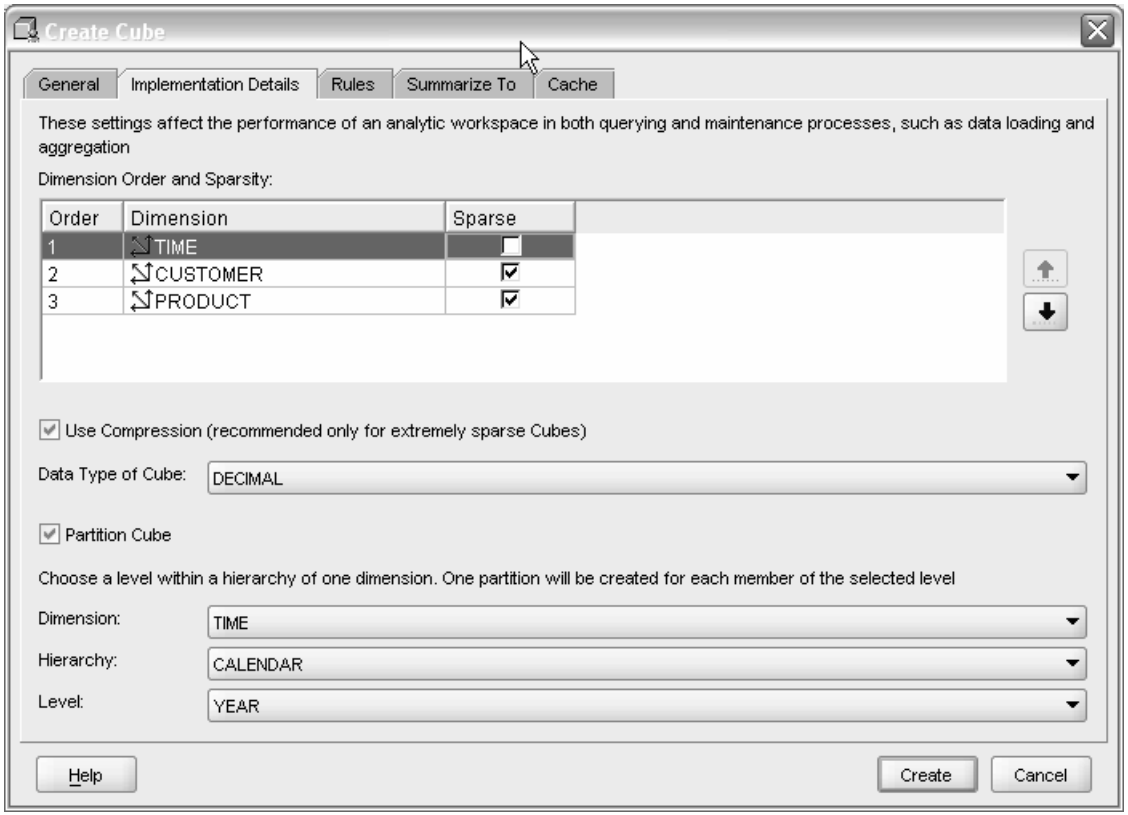
(5 cells)

	Dim1	Dim2	Value
1	D	W	65
2	E	V	35
3	F	U	20
4	G	T	10
5	G	V	50

- na's take as much space as values
- Composites save space with sparse data
- Composites use significant amount of overhead

Implementation Details Tab: Dimension Order

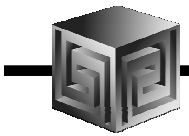
Think about sparsity and use of compression first.
(Compression means the use of compressed composites)



What Is a Compressed Composite

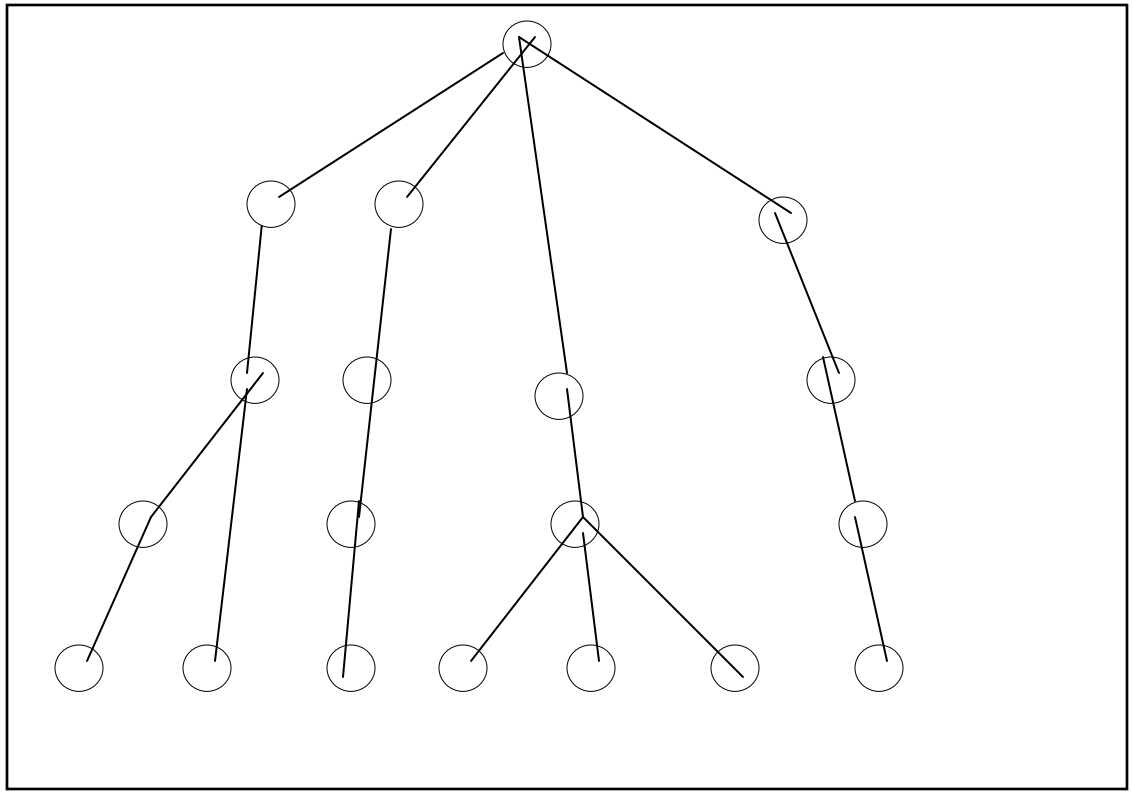
- Normal composite has tuples for
  - all the leaf values, and
  - all the precomputed aggregate values (aggindex no), or
  - all the aggregate values (aggindex yes)

- With sparse data many aggregate tuples may have only a single child and hence have the same data value as their child.

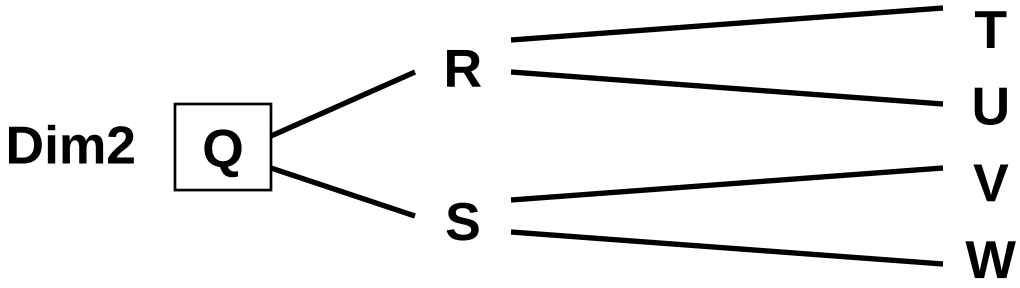


In the Real World… Single Child Situation Is Common

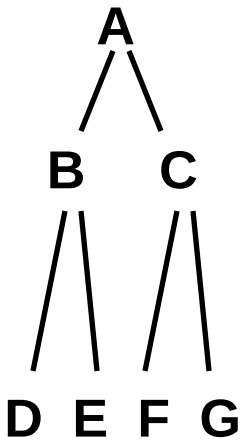
- **Especially in a multidimensional situation.**
- **The red nodes can be compressed out.**



Compression Increases with Multiple Dimensions



Dim1



	Q	R	S	T	U	V	W
A	180	30	150	10	20	85	65
B	100		100			35	65
C	80	30	50	10	20	50	
D	65		65				65
E	35		35			35	
F	20	20			20		
G	60	10	50	10		50	

Base data

Compressible cell

Compressed Composite (CC)

- CC knows where these runs of single child parent tuples are
- Stores the common value for these runs only once
- Doesn't materialize the tuples in these runs
- Smaller footprint on disk and in memory
- Faster aggregation
- Can be orders of magnitude smaller/faster!

CC Limitations in 10gR1

- **The only thing you really need worry about is:**
  - **SUM method or NOAGG method of aggregation only.**

- **Less importantly but in the spirit of full disclosure:**
  - **No partial aggregation – CC's are so good this doesn't matter (usually).**
  - **A CC can dimension only a single variable – not a concern to you.**
  - **A CC's aggregate tuples cannot be updated once built**
    - **To make changes, the aggregates are thrown away.**
    - **CC's are so good this doesn't matter (usually).**

When Can Compression Be Used?

- SUM method of aggregation
- Data are sparse

- How sparse is sparse?
- Not as sparse as you might think

Sparsity Use Case #1

- Existing OSA application
- 14 measures
- Time at week, month, year (260 values)
- Product (4,220), customer (7,804) and channel (22)
- Deepish hierarchies on product and customer
- 2.9M input rows
- 9i OSA build on 6GB Machine
  - 616 minutes
  - 100GB on disk

Data not incredibly sparse, so OK to use regular composites

Sparsity Use Case #2

- **In 9i:**
  - **Year level data only with skiplevel aggregation.**
  - **Took >1 day to load and aggregate.**
- **In 10g with AWM10g:**
  - **1 cpu, 2 Gb. RAM machine**
  - **Time dense, other dims in CC.**
  - **Partition on time at year level.**
  - **No parallelization**
  - **89 min. load & upd. + 115 min. agg = 204 minutes**
- **Note: daily load of data would take about 12 or 13 minutes.**
- **With SEG dimension out of the CC aggregation was significantly slower.**

Sparsity Use Case #3 – Regular Composites in 9i

SH schema.
TIME: 2,261 values (5 years)
- Calendar hierarchy: day, month, quarter, year
- Calendar_week hierarchy: day, week, quarter, year
- Fiscal hierarchy: day, month, fiscal quarter, fiscal year

CHANNEL: 9 values
CUSTOMER: 56,303 values
PRODUCT: 100 values
PROMOTION: 535 values
Fact table: 918,000 rows, 2 measures

In 9i, build took hours plus OLAP DML skills.

Sparsity Use Case #3 – Compressed Composites in 10G

- All dimensions in a CC (because of daily data)
- Two cubes
  - Calendar cube (the two calendar hierarchies)
    - Partitioned on calendar hierarchy at quarter level.
  - Fiscal cube (fiscal hierarchy)
    - Partitioned on fiscal hierarchy at quarter level. This cube was used so that the Fiscal hierarchy would be solved (up to the quarter level).
    - The partitioning allows fast processing of a daily update (only one quarter need be recalculated) and parallelization of a full build.
- Build time: 9 minutes on a laptop (no parallelization)

Sparsity Use Case #4 – 9i

- DATE: 14 leaf values, 5 levels
- LOB: 162 leaf values, 4 levels
- COMPL_RATING: 23 leaf values, 3 levels
- INSTRUMENTS: 171 leaf values, 3 levels
- OWNERSHIP: 69,771 leaf values, 6 levels, 2 hierarchies
- DATE dense; other dims in composite
- 190,676 leaf tuples
- In 9i:
  - Full agg: 166 min., 3.65 Gb., 4.65M tuples
  - Partial agg: 37 min., 1.3 Gb., 1.52M tuples

Sparsity Use Case #4 – 10g

- DATE dense; other dims in CC
- Agg time: 1 minute 12 seconds!!!
- AW size: 148 Mb
- 138x faster
- 25x smaller

Roles and Privileges

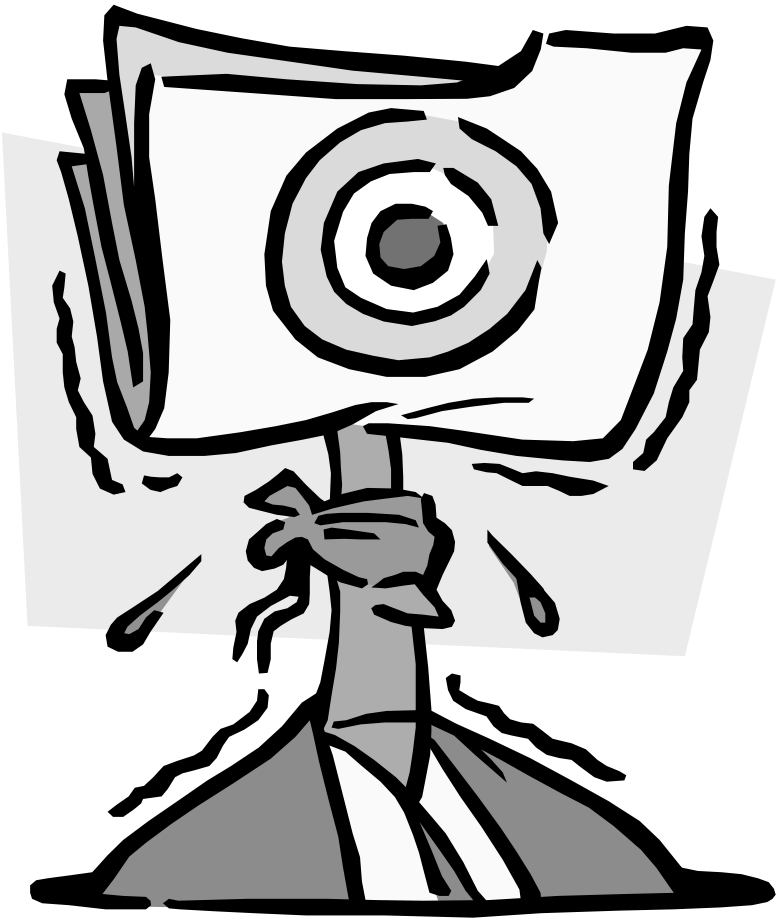
- Roles (OLAP_USER, DBA, OLAP_DBA)
- Privileges (System and Object)
- Minimum for OLAPI (connect/resource)
- Too much can be bad
- To hide ROLAP Cubes from AW users revoke select on only one table. NOTE: Requires refresh to be run by ROLAP user, which means ROLAP user must have insert/update priv on AW.

Diagnostics / Monitoring: XML_LOAD_LOG

- **Select the messages:**

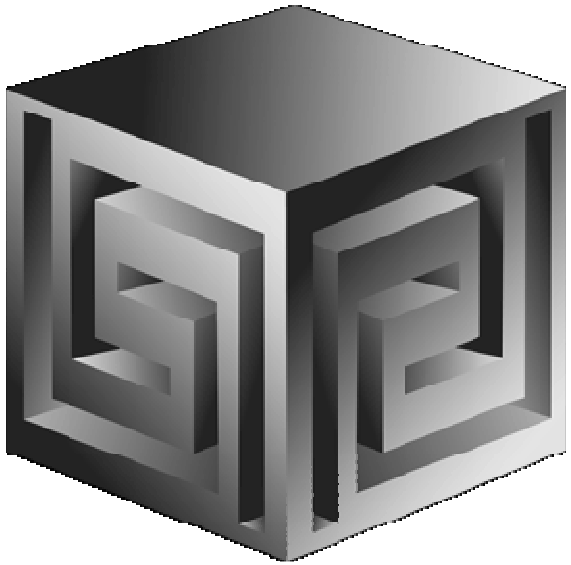
```
SQL> set linesize 132 pagesize 100
SQL> select xml_message from olapsys.xml_load_log
   2  where xml_loadid=710 order by xml_loadid;

XML_MESSAGE
--------------------------------------------------------------------------------
16:00:05 Started Build(Refresh) of PRICING.PRICE1 Analytic Workspace.
16:00:05 Attached AW PRICING.PRICE1 in RW Mode.
16:00:05  Started Loading Measures.
16:00:05   Started Load of Measures: UPG_PRICE, UPG_COST from Cube UPG.CUBE.
16:00:08   Finished Load of Measures: UPG_PRICE, UPG_COST from Cube UPG.CUBE.
            Processed 9 Records. Rejected 6 Records.

16:00:08   Started Auto Solve for Measures: UPG_COST, UPG_PRICE from Cube UPG.CUBE.
16:00:10   Finished Auto Solve for Measures: UPG_COST, UPG_PRICE from Cube UPG.CUBE.
16:00:10  Finished Loading Measures.
16:00:10 Completed Build(Refresh) of PRICING.PRICE1 Analytic Workspace.

9 rows selected.
```

QUESTIONS?



Oracle Business Intelligence in the Real World: Case Studies from the Trenches

ODTUG 2006



Dan Vlamis

dvlamis@vlamis.com

Vlamis Software Solutions, Inc.

816-781-2880

http://www.vlamis.com