Planning for High Performance OBI

September 19, 2010

Jeff McQuigg

Sr. Architect, KPI Partners

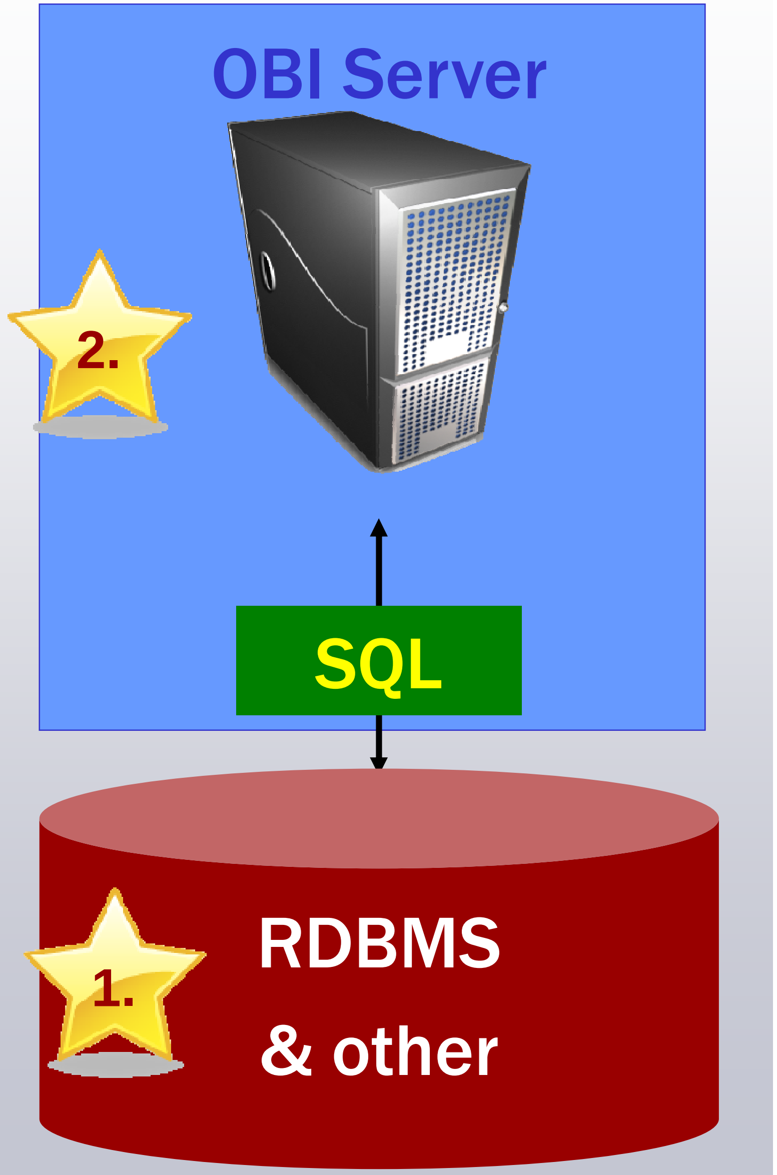
www.kpipartners.com

Agenda

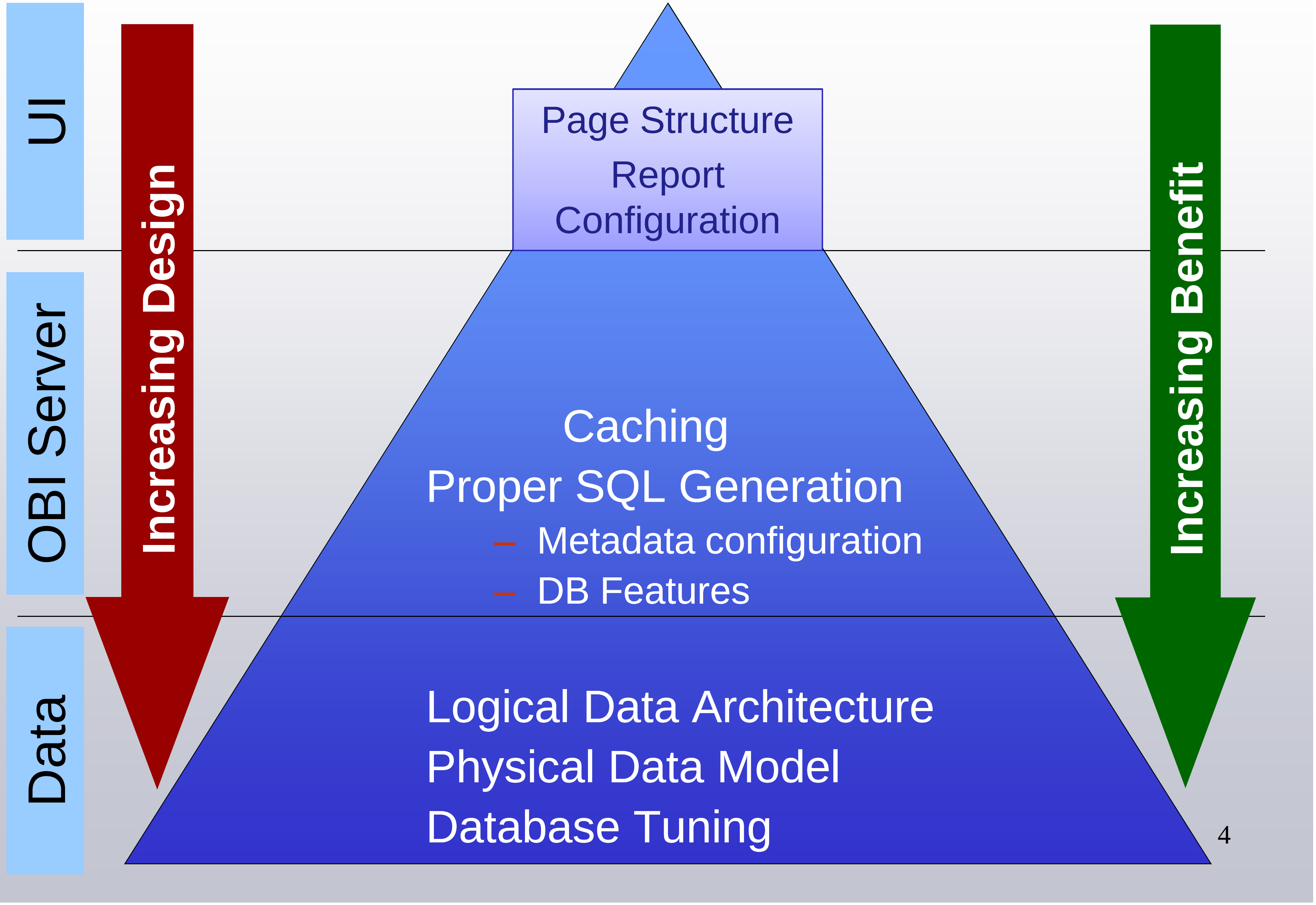
- Introduction
- Logical Data Architecture
- Physical Data Model
- Physical Database – Oracle Focused
- OBI Server
- User Interface
- Q&A

The Big Picture

- OBI/OBIEE is a ROLAP engine
  - Relational OLAP
- ROLAP BI Tools = SQL Generation engines
- Great Performance in a nutshell:
  1. Proper Data Design
  2. Proper SQL



Tuning Focus



UI

OBI Server

Data

Increasing Design

Increasing Benefit

Page Structure

Report Configuration

Caching

Proper SQL Generation

- Metadata configuration
- DB Features

Logical Data Architecture

Physical Data Model

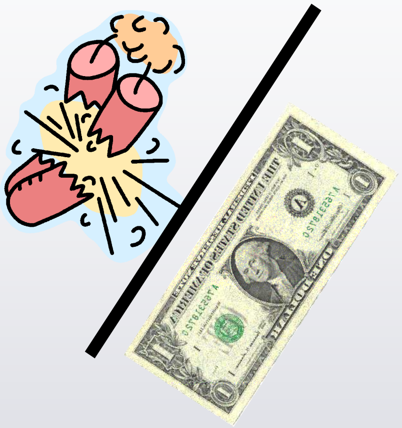
Database Tuning

Great Performance Preview

- Most of Great Performance is achieved before the 1st report is ever run

- 95% of great performance comes down to
  - Data Design
  - Database Tuning
  - Generating the corresponding *Perfect SQL*

- This will be the focus for today's seminar



- Dashboards and Answers impact performance to a lesser degree
  - Highlight a few basic UI design issues

- Caching will be briefly discussed

- Not discussed today:
  - OBI Server settings
  - Clustering

KPI Partners - Leaders in Oracle BI and EPM

Largest and Most Experienced ORACLE BI Services Company

- Over 80 consultants with 55+ in USA and 30+ in India
- Strong presence in San Francisco, Southern California, Atlanta, Boston, New York City, Washington DC, Chicago and Toronto

Extensive customer references and accolades

- Pillar Partner for Oracle Business Intelligence in Southern California, Northern California, North East and South East Regions
- PNC bank won Oracle's 2008 customer excellence award
- Netshops and PNC Bank presented at Oracle open world 2008
- Novartis Pharmaceuticals awarded "Above and Beyond" award to KPI Partners
- 5 customers presented at Oracle Open World 2009

Proven Implementation Methodology

- Templates of Process Maps that ensures a successful deployment

Leader in innovation

- First partner to integrate EBS, BI applications, OBIEE and Hyperion

PNC Bank wins 2008 Oracle Customer Excellence Award

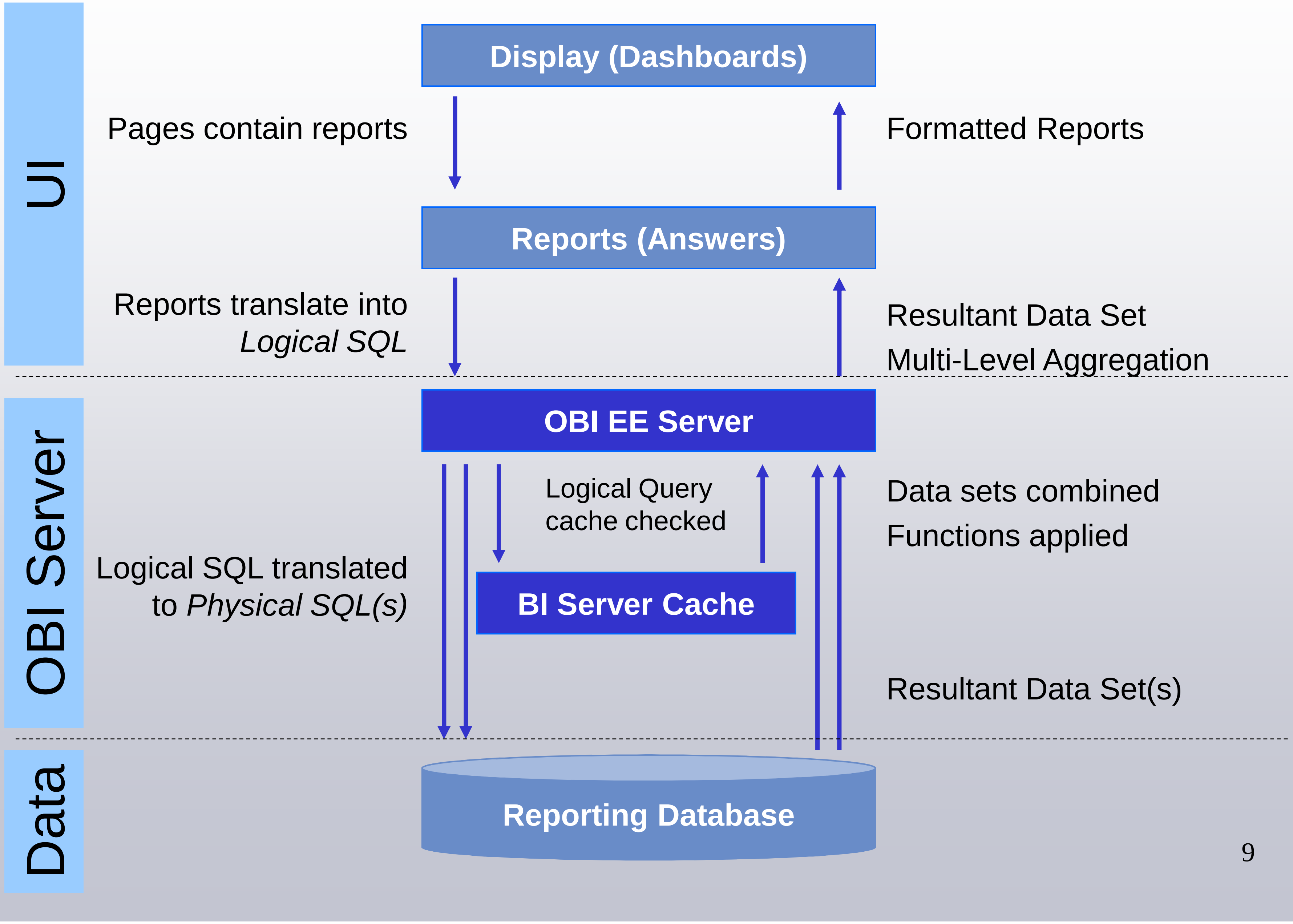
KPI PARTNERS
NAMED ORACLE'S BI ROOKIE
PARTNER OF THE YEAR '07

About Jeff McQuigg

- Senior Architect at KPI Partners
- 9 years Siebel Analytics & OBI consulting experience, 18 years overall
- Developed many of Siebel's early Analytics v7.0 best practices & wrote part of the certification exam
- Personally been involved with over 30+ OBI projects in every capacity (BI Architect, Data Modeling, RPD Metadata, Business Analyst, Report Developer, ETL Architect/Developer, Project Manager)
- BI & OBI Thought Leader:
  - Longtime IT Toolbox Blog & now greatobi.wordpress.com
  - Co-Moderator of new architect-level OBIEE Enterprise Methodology discussion board
  - Frequent Oracle Open World Speaker
  - Developed OBI Deployment Methodology, deliverable templates and all Training programs with Metricsphere
- Currently day-to-day managing & architecting a $2.1 million custom DW & OBI project

Logical Data Architecture

Anatomy of an OBI Query



UI

OBI Server

Data

Display (Dashboards)

Pages contain reports

Formatted Reports

Reports (Answers)

Reports translate into *Logical SQL*

Resultant Data Set

Multi-Level Aggregation

OBI EE Server

Logical Query cache checked

Data sets combined

Functions applied

Logical SQL translated to *Physical SQL(s)*

BI Server Cache

Resultant Data Set(s)

Reporting Database

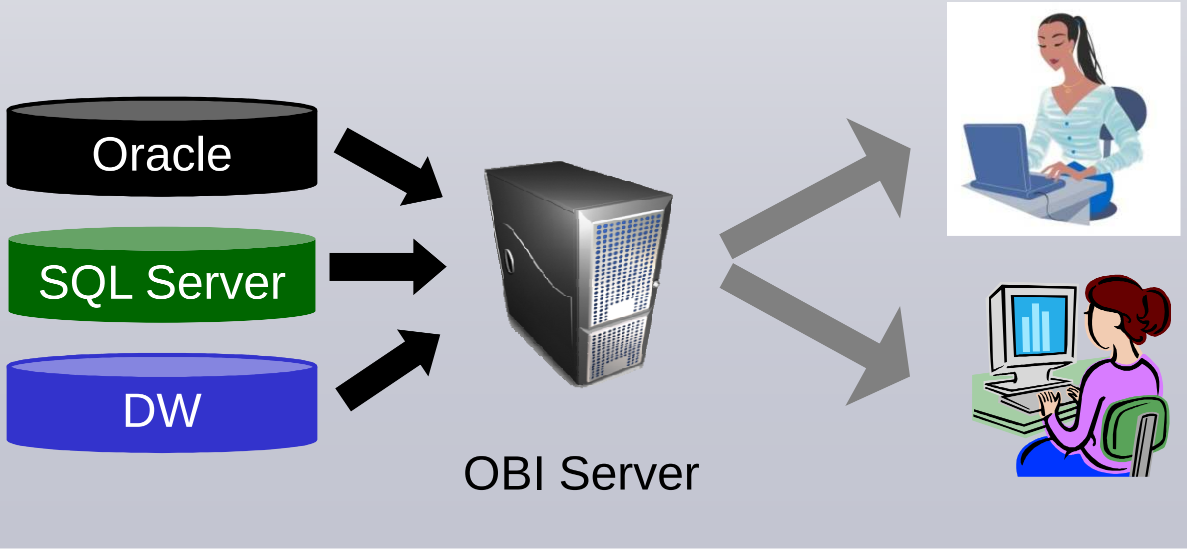
Logical Data Architecture - Centralize

Step #1: Consolidate

- Data can reside in multiple physical places
  - OBI can map to each database

- Federation – “Play the data where it lies”
  - nQuire/Siebel Analytics/OBI strength from Day 1
  - Enterprise Information Integration (EII)

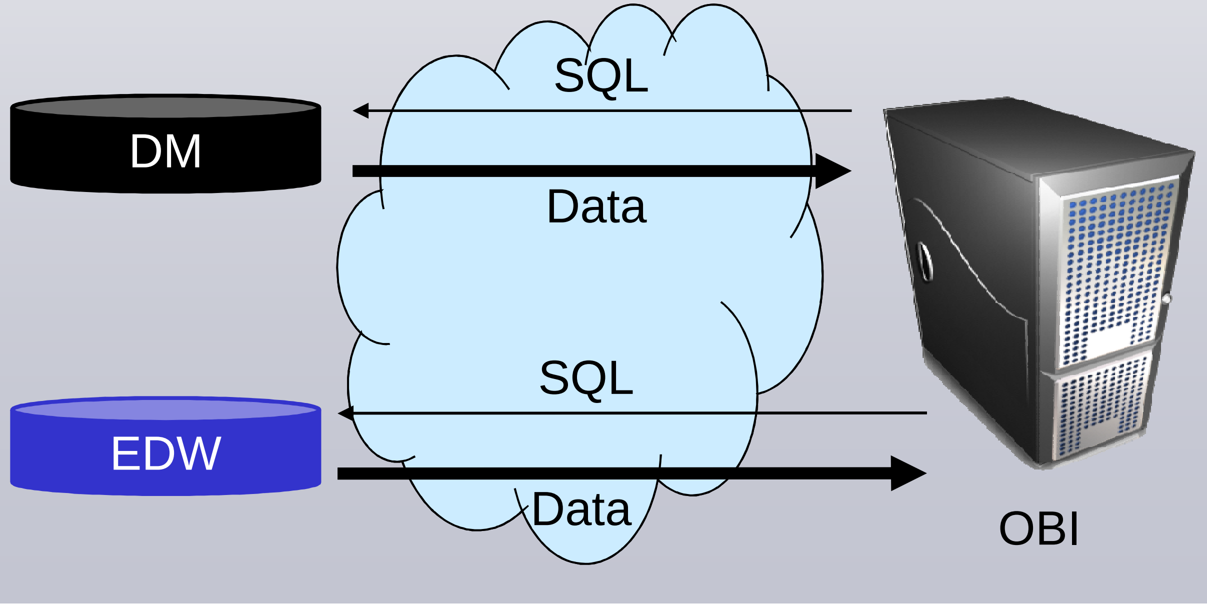
- Virtual Database with distributed tables



Logical Data Architecture - Centralize

Federated Model – Good for the Oracle Sales Pitch

- OBI generates queries for each data source
- Data returns from each data source
  - Sometimes at a low level grain (detail records!)
- Data must then be integrated on the OBI Server
  - Essentially ‘Network Joins’
- Large data over the wire = poor performance
  - Good for prototypes and small sources, such as forecast XLSs



Join,
Merge,
Aggregate
and Sort

Logical Data Architecture - Centralize

- Centralize data sources into a single database schema

- Let the ETL do the hard work of data movement
  - Do it ***once*** at night vs. do it ***every time*** for a query

- When data resides in one place:
  - A single query can be generated
  - Small result sets returned
  - Network joins eliminated

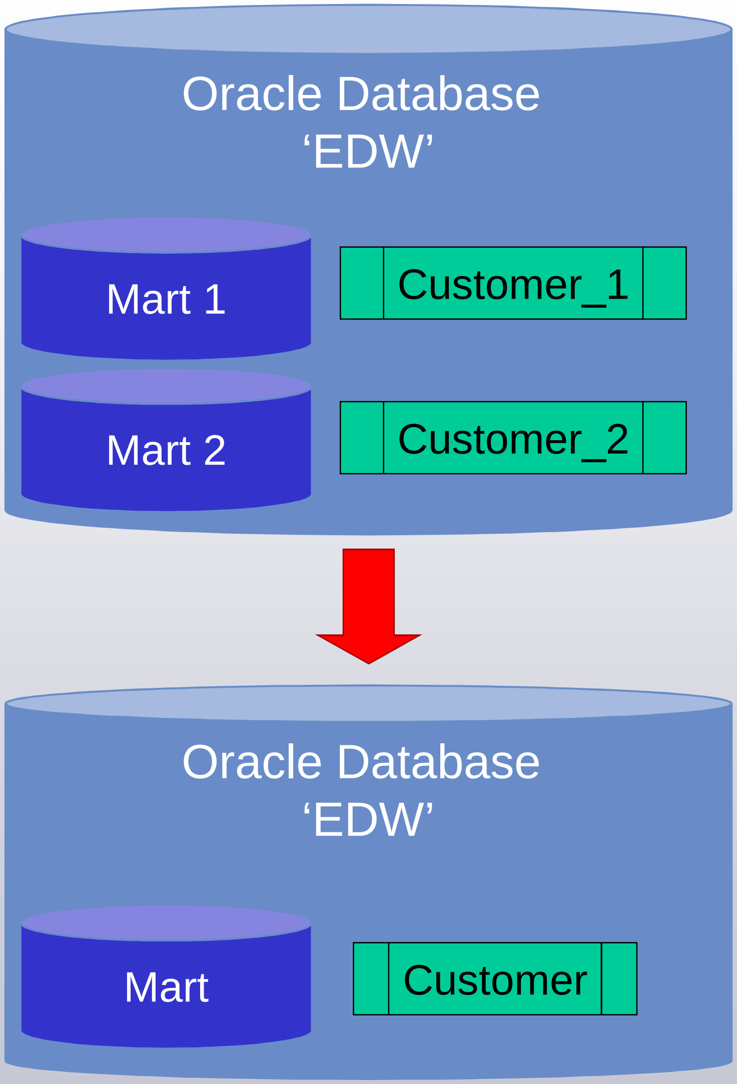
- This goes double when using transactional sources
  - Data Warehouses were invented to help large queries

Logical Data Architecture - Integrate

Step #2: Integrate

- Data may be physically in one schema but still in separate tables
  - Multiple OBI mappings ➔ multiple queries
  - More tables to link ➔ slower performance
- Integration merges and links data together
  - Complex integration rules are done by ETL not user queries
- Fewer database objects ➔ better object reuse

- Data Integration is the hard part of any DW or BI System
  - Complex rules are needed
  - Poor data quality
  - Differing definitions



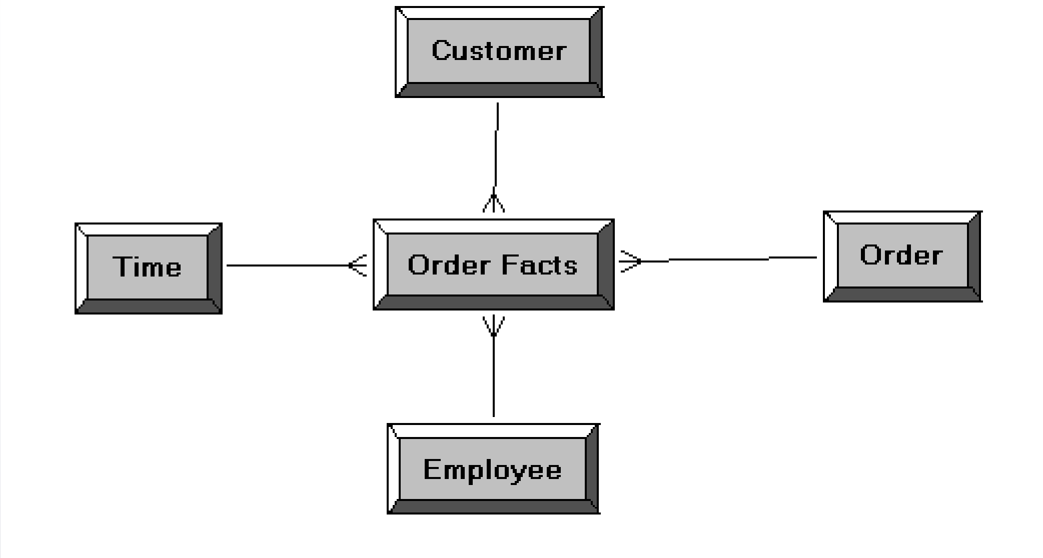
Physical Data Model

Physical Data Model Overview

- Your main performance weapon is a good Dimensional Model
  - Like a computing Algorithm – not all models are created equal
- Dimensional Modeling is:
  - A data modeling approach
  - Conceptually different to relational
  - Designed for large query performance
  - A skill set unto its own

- A proper ‘Star Schema’ Dimensional Model:
  - Makes mapping OBI Metadata very easy
  - Optimal database query performance
  - Uses database optimizations
- Poorly designed models are common
- Think Top-Down design

Dimensional Modeling Overview



- "Fact tables" hold measurements/metrics/facts
  - Counts, Cycle Times, $ Amts, Qtys, Prices, Estimates, Forecasts

- Link to several "Dimension tables", which contain descriptions
  - Geography, Customers, Products, Time, Employees, etc

- *The* Book on Dimensional Modeling
  - *The Data Warehouse Toolkit* by Ralph Kimball

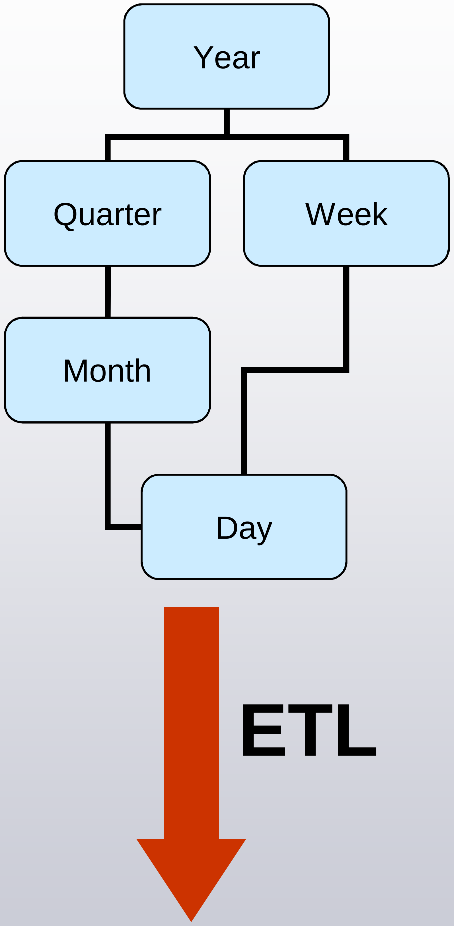
Dimensional Modeling Overview

- Dimensional Modeling is a topic beyond the scope of this session
  - Discuss 5 general purpose techniques/tips
- Tips will work great for mainstream databases
  - Oracle
  - DB2
  - Sybase/SQL Server
- Different recommendations for MPP databases
  - Teradata
  - DW Appliances (e.g., Netezza)

Dimensional Modeling Toolkit

Technique #1: De-normalize

- Refers to dimensional objects, not facts
- Merge several tables into one flat table
  - Opposite of relational modeling
  - Do in an OBI Logical table anyway
- Nightly ETL does the de-normalization
- **Goal: Reduce Joins for a query**
  - Joins slow execution
- Additional Benefit: Simple Model
  - ➔ Less OBI Mapping effort
- Caution: Facts must remain normalized
  - Inaccuracies will arise



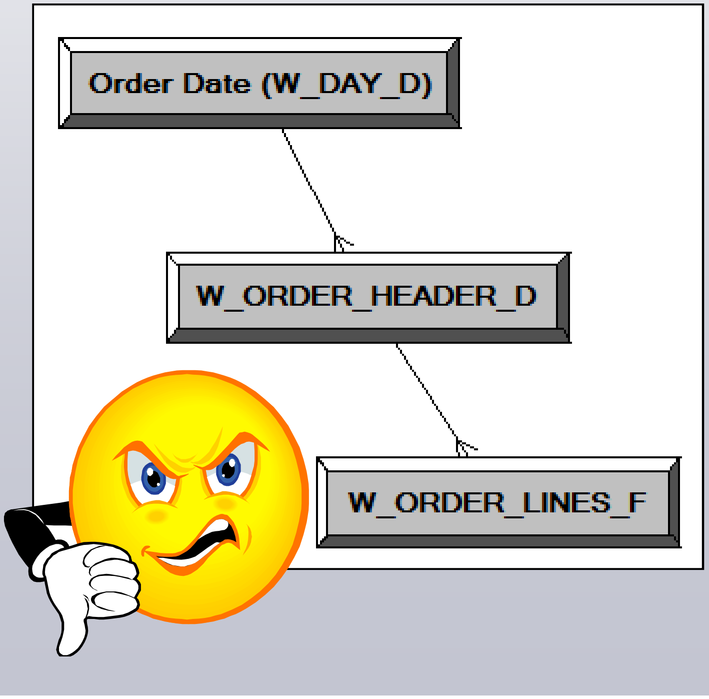
Day | Week | Month | Quarter | Year

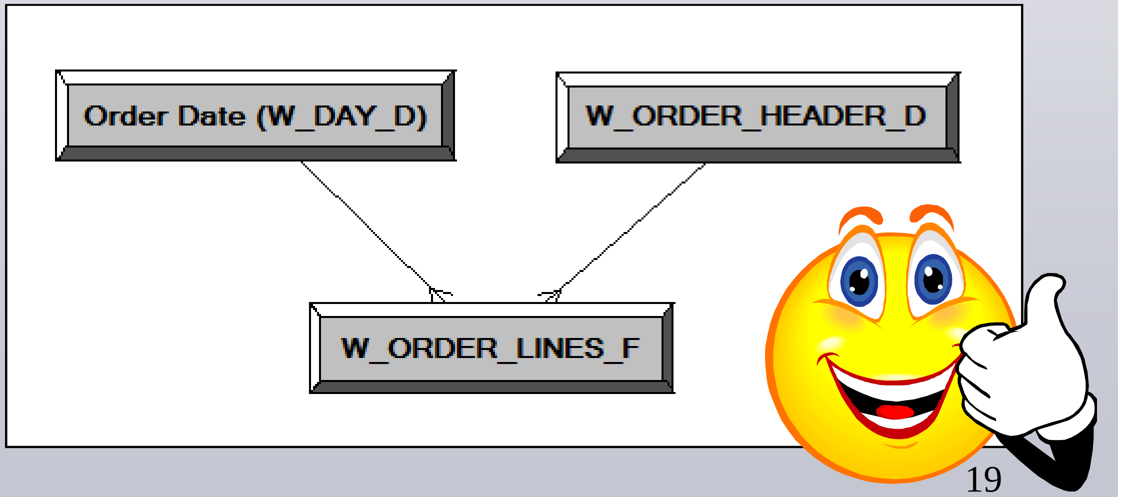
Dimensional Modeling Toolkit

- De-normalize dimensions to link to a fact

- A fact table links ***directly*** to its dimension tables
  - There are no intermediate tables (there are exceptions)

- Use the ETL to directly link each load



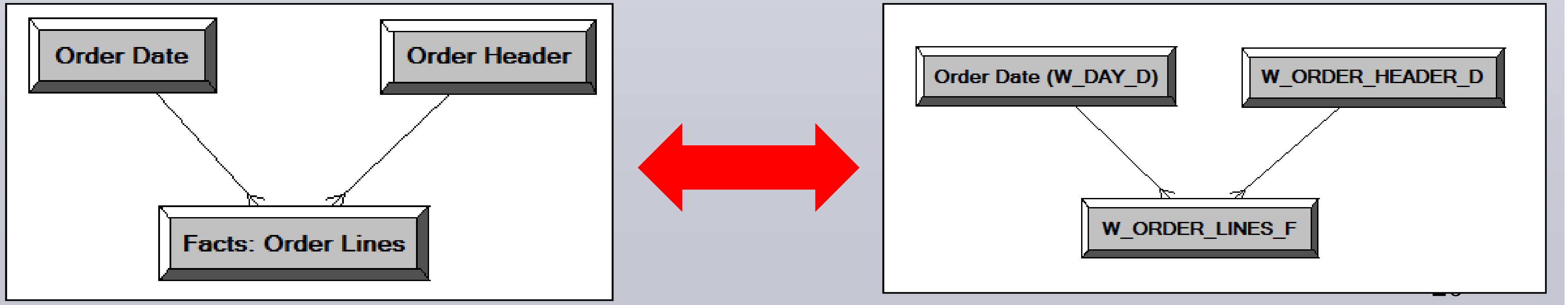


Dimensional Modeling Toolkit

- The OBI Business Model (BM) defines your business view of your information
  - Defines ‘what you want’ out of your system

- Design rule of thumb: Make the Physical Data Model resemble the Business Model
  - The BM can map to a variety of physical models
  - Clean & simple mapping is best

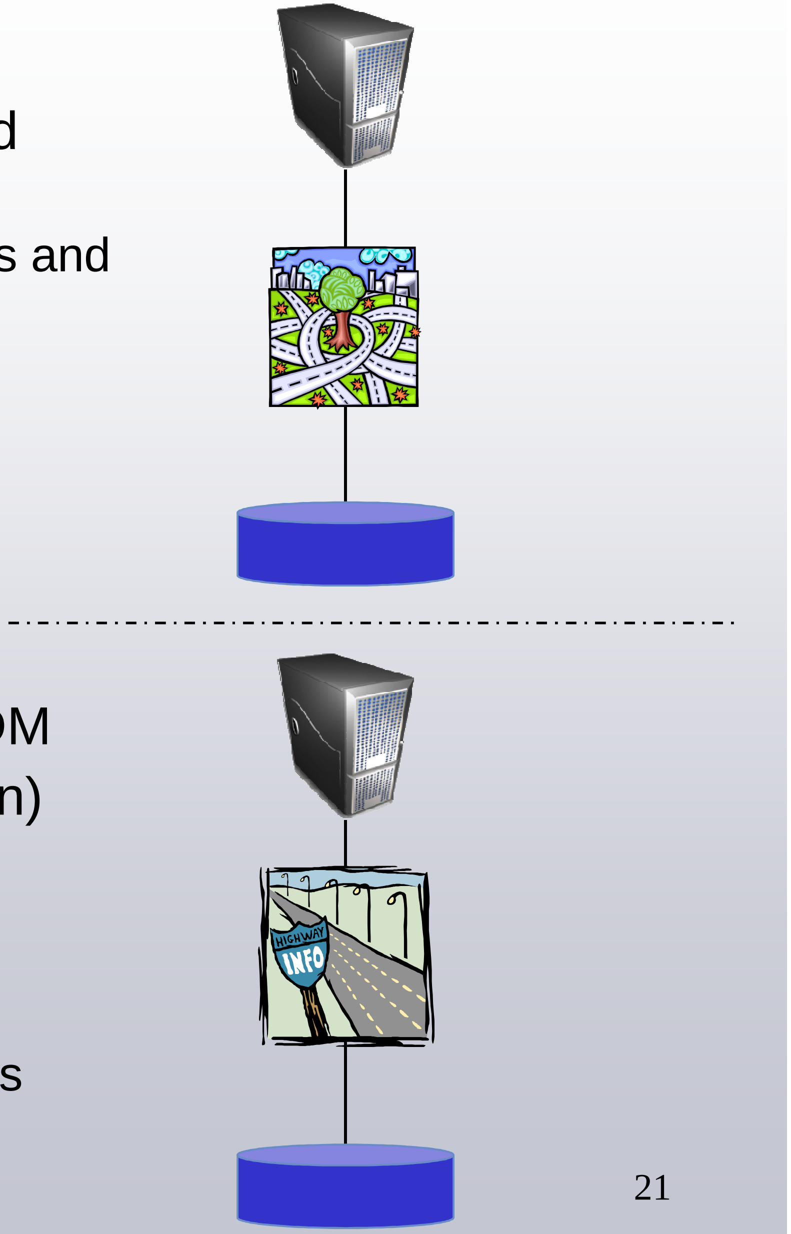
Business Model **Physical Model**



Dimensional Modeling Toolkit

Technique #2: Put Logic in Data Model

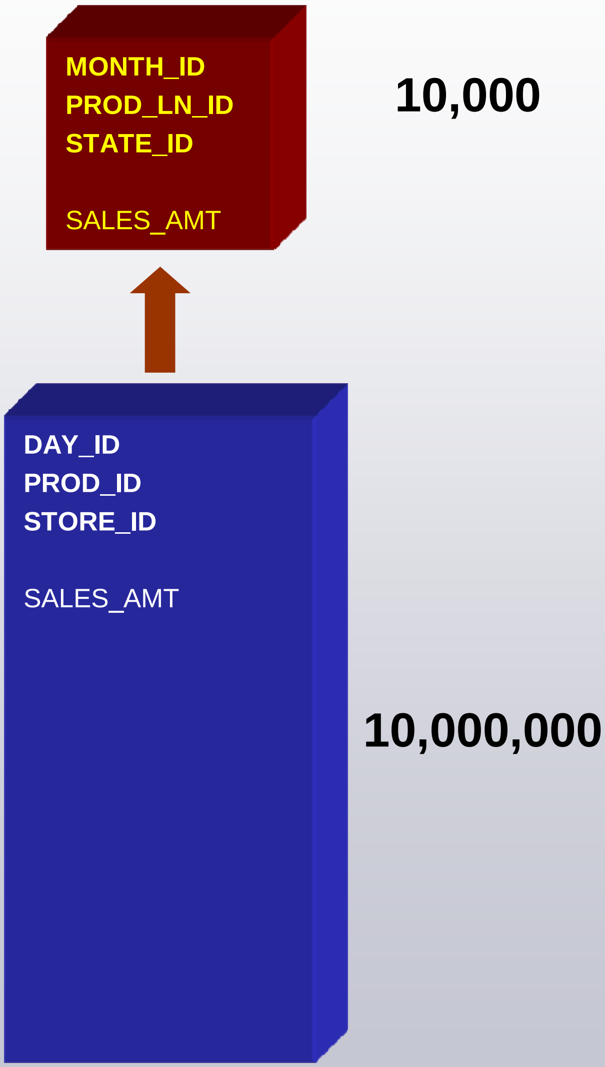
- Move complex calculations to the ETL and store the results as a new field/table
  - Some calculations may require many steps and differing input data sets
- *Do once* & use often vs. *do often*
- **Goal: Simple Query SQL**
- One of main differences between DW & DM
  - DM tailors to a specific use (Top-Down)
  - DW is generic in nature (Bottom-Up)
- Benefits:
  - Can use other database performance tricks
    - Can index the column
  - Simple SQL performs better



Dimensional Modeling Toolkit

Technique #3: Aggregate Tables

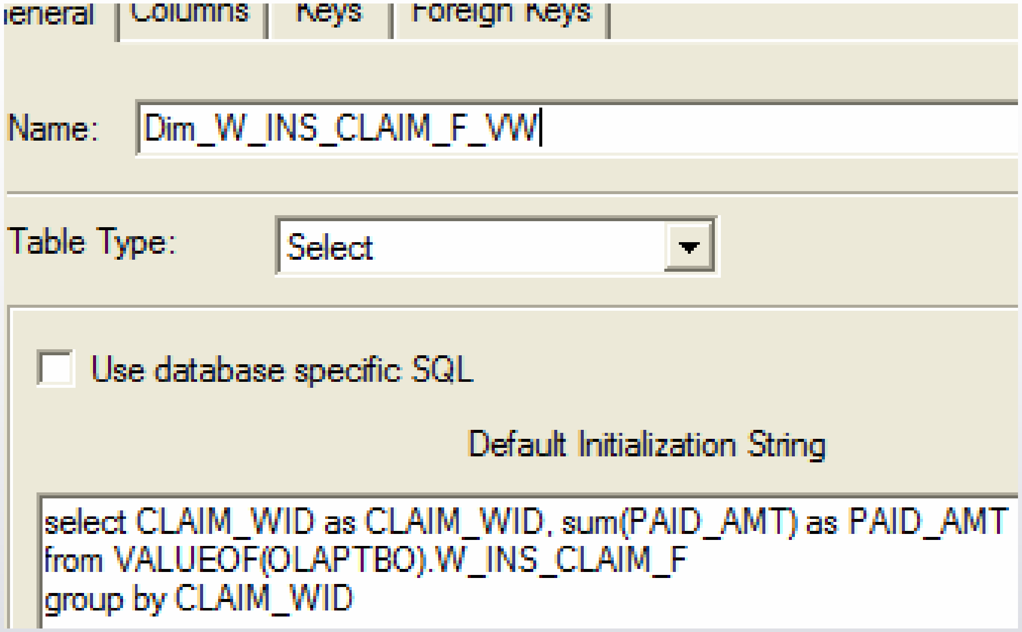
- Summarize detail records to useful levels
- Use ETL or Materialized Views to create ‘higher’ tables
  - Higher grain tables have fewer combinations & are smaller
- Consider when at least 10:1 compression
  - Be careful with Day → Week aggregates
- Map into OBI to have it pick the right table
  - Remember to set the Content Tab
- **Goal: Database processes fewer rows per query**
- Caution: Consider the costs of each new aggregate table:
  - ETL Batch window, storage, complex mapping, more code, more QA
  - Will other, easier techniques suffice?



Dimensional Modeling Toolkit

Technique #4: Views

- Views ***do not help or hurt*** performance
- *Opaque Views* and Database Views are ***identical*** to the database
  - A view is a logic encapsulation device
- Exception: Materialized views
  - Physical tables computed in the ETL process
- Try to eliminate views in your system



Name: Dim_W_INS_CLAIM_F_VW

Table Type: Select

☐ Use database specific SQL

```
select CLAIM_WID as CLAIM_WID, sum(PAID_AMT) as PAID_AMT
from VALUEOF(OLAPTBO).W_INS_CLAIM_F
group by CLAIM_WID
```

Identical to the database

```
CREATE VIEW Dim_W_INS_CLAIM_F_VW AS
select CLAIM_WID as CLAIM_WID,
sum(PAID_AMT) as PAID_AMT
from VALUEOF(OLAPTBO).W_INS_CLAIM_F
group by CLAIM_WID;
```

Dimensional Modeling Toolkit

Technique #5: Creativity

- Dimensional Modeling is 25% Science, 75% Art
- Create solutions to problems as needed, keeping in mind the basic rules of Dimensional Modeling
  - Example: Mini Dimensions & Junk Dimensions
- Keep in mind the following goals:
  - Want to use simple SQL queries
  - Reduce the amount of data the database has to read
  - Use the ETL engine to offload logic from user queries
- Be sure to weigh costs $ of every design decision

Physical Database (for Oracle)

Physical Database Features

- Sometimes there is simply a lot of data
- Highlight 2 key Oracle database features
  - Partitioning
  - Star Transformations
- Can be designed into the physical data model by the application team
- Commonly used features for data warehousing
  - Have been in Oracle database for years
- Are very efficient tools to improve many queries

Physical Database Features

Partitioning

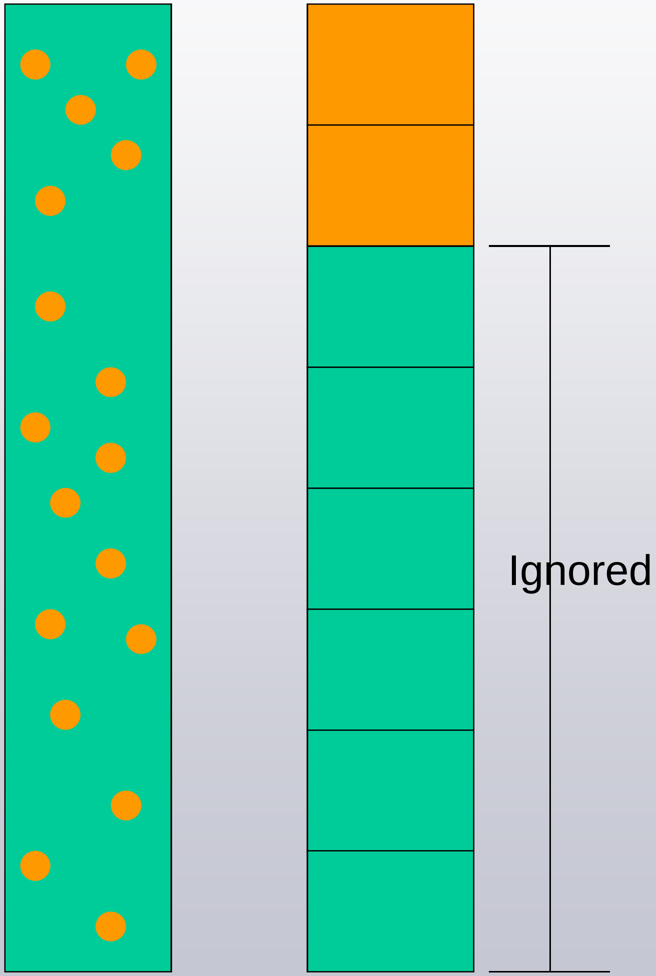
- Breaks up large tables into several smaller pieces (partitions)
- Database reads only a subset of the whole table
- The database has metadata about each piece
  - Defines what range of data each partition contains
- **Goal**: eliminate non-needed data reads
  - Ex: Monthly partitions on a fact table with 4 years of daily data
    - Many queries are interested only in recent data
    - Ex: 2 months of data read instead of all 48 months
- Dramatic performance benefits are possible

UnPartitioned

Data spread across whole table

Partitioned

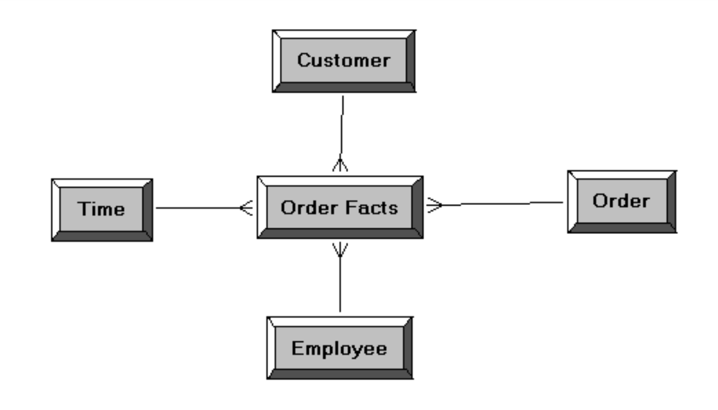
Data contained in specific portions



Physical Database Features

Star Transformations

- Alternative execution plan for a Star Schema
- Uses additional information to recognize the Star pattern
- Requires bitmap indexes and FK constraints
- Optimizer rewrites the query
- **Goal:** Use *all* of the dimensions to go after a smaller set of fact rows
- Useful for highly selective queries



Normal:

Use ***only 1*** dimension to filter the Fact table

e.g., Filter on Customer only

Star Transformation:

Use ***all*** dimensions to filter the Fact table

e.g., Filter on Customer & Date & Employee

Other Tuning Items

- Ensure Indexes are heavily used
  - Review the Explain Plan and eliminate Full table scans
- Oracle Parallel Query may help
  - Best for large table or index scans
  - Consider other techniques first
- Star Transformations don’t always help
- Narrow down large tables to reduce bytes
  - Remove unused fields & metadata columns

OBI Server

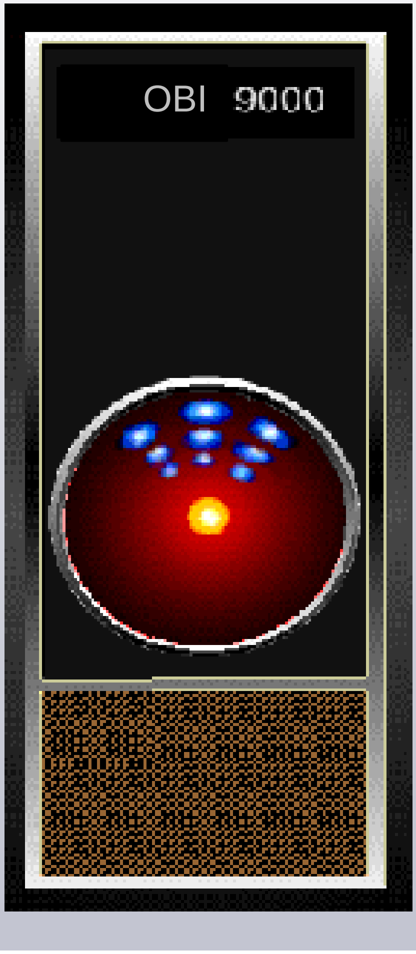
OBI Server Tuning

- A good dimensional model is worthless if questions to it are poorly constructed
- **OBI Repository Goal**: Configure it to generate ***good SQL***

If it's not perfect, it's wrong

- Get to know the NQQuery.log file!
- ***Design Key***: Know what the proper SQL should be beforehand
  - correct # of queries being generated?
  - correct tables being used?
  - correct joins being used?
  - How does the design of a report change the SQL?

OBI 9000 says: "It can only be attributable to human error."



Know Your SQL

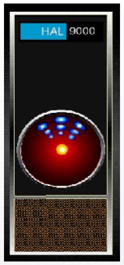
- It all starts by knowing the correct SQL for each query & report

- Brush up on your SQL!
  - Become knowledgeable of:
    - The WITH clause
    - What the ROW_NUMBER() does to SQL
    - How Time Series SQL works
    - Aggregate window functions (e.g., MAVG())
    - Subtotals (PARTITION BY)
    - Set Math (Union)
  - Newer OBI versions generate more complex SQL

- When tuning:
  - Start simple (not totals/subtotals & single fact tables)
  - Then add report logic to see when a problem arises
  - Always start with the Table view
    - Confirm your base record set and underlying query

Common Tuning: Wrong # of Queries

- Know the # of queries expected
  - Some requests should generate 2 queries
  - If 3 are generated, then OBI has been set up incorrectly



- OBI in some cases will generate one large query with 2 queries merged together

```
(Select A,Sum(B) from Fact B, Dim A)
FULL OUTER JOIN
(Select A, Sum(X) from Fact X, Dim A)
```

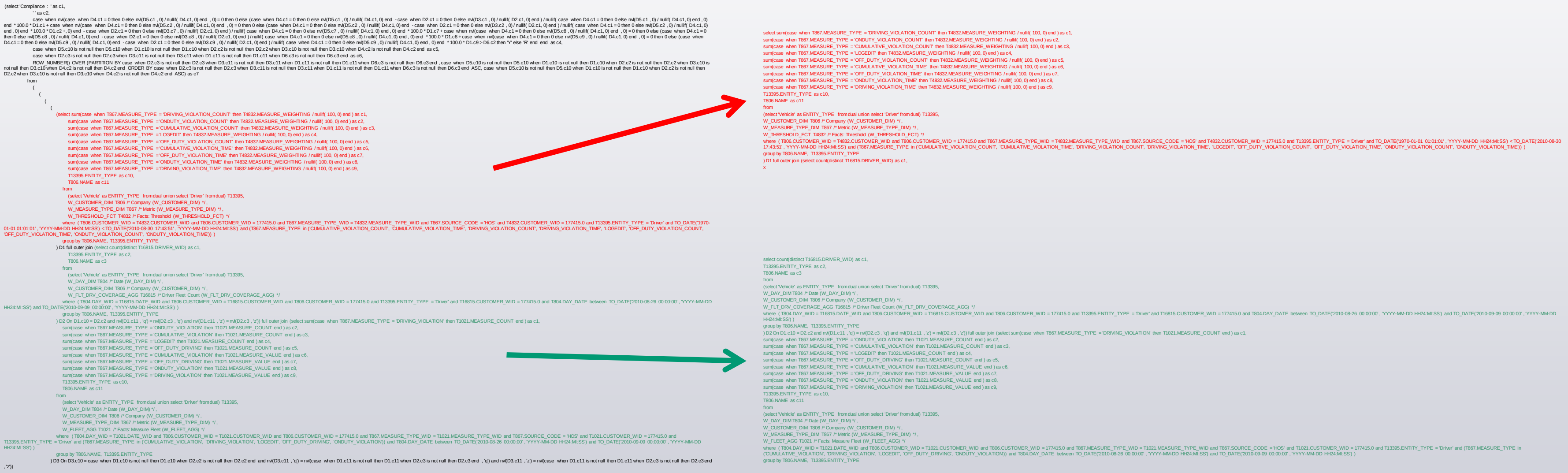
  - Check the Perf_Prefer_Internal_Stitch_Join and ROWNUM_SUPPORTED parameters to control

- Subtotals and report totals may generate sub-queries

Database Features Tab

- The default settings are not always optimal!
- Plan on running performance tests with different options

- Deselecting ROWNUM_SUPPORTED
- 1 massive query → many smaller queries



```
(select 'Compliance :  ' as c1,
        ' ' as c2,
        case  when nvl(case  when D4.c1 = 0 then 0 else nvl(D5.c1 , 0) / nullif( D4.c1, 0) end  , 0) = 0 then 0 else (case  when D4.c1 = 0 then 0 else nvl(D5.c1 , 0) / nullif( D4.c1, 0) end  - case  when D2.c1 = 0 then 0 else nvl(D3.c1 , 0) / nullif( D2.c1, 0) end ) / nullif( case  when D4.c1 = 0 then 0 else nvl(D5.c1 , 0) / nullif( D4.c1, 0) end , 0) end  * 100.0 * D1.c1 + case  when nvl(case  when D4.c1 = 0 then 0 else nvl(D5.c2 , 0) / nullif( D4.c1, 0) end  , 0) = 0 then 0 else (case  when D4.c1 = 0 then 0 else nvl(D5.c2 , 0) / nullif( D4.c1, 0) end  - case  when D2.c1 = 0 then 0 else nvl(D3.c2 , 0) / nullif( D2.c1, 0) end ) / nullif( case  when D4.c1 = 0 then 0 else nvl(D5.c2 , 0) / nullif( D4.c1, 0) end , 0) end  * 100.0 * D1.c2 +, 0) end  - case  when D2.c1 = 0 then 0 else nvl(D3.c7 , 0) / nullif( D2.c1, 0) end ) / nullif( case  when D4.c1 = 0 then 0 else nvl(D5.c7 , 0) / nullif( D4.c1, 0) end , 0) end  * 100.0 * D1.c7 + case  when nvl(case  when D4.c1 = 0 then 0 else nvl(D5.c8 , 0) / nullif( D4.c1, 0) end  , 0) = 0 then 0 else (case  when D4.c1 = 0 then 0 else nvl(D5.c8 , 0) / nullif( D4.c1, 0) end  - case  when D2.c1 = 0 then 0 else nvl(D3.c8 , 0) / nullif( D2.c1, 0) end ) / nullif( case  when D4.c1 = 0 then 0 else nvl(D5.c8 , 0) / nullif( D4.c1, 0) end , 0) end  * 100.0 * D1.c8 + case  when nvl(case  when D4.c1 = 0 then 0 else nvl(D5.c9 , 0) / nullif( D4.c1, 0) end  , 0) = 0 then 0 else (case  when D4.c1 = 0 then 0 else nvl(D5.c9 , 0) / nullif( D4.c1, 0) end  - case  when D2.c1 = 0 then 0 else nvl(D3.c9 , 0) / nullif( D2.c1, 0) end ) / nullif( case  when D4.c1 = 0 then 0 else nvl(D5.c9 , 0) / nullif( D4.c1, 0) end , 0) end  * 100.0 * D1.c9 > D6.c2 then 'Y' else 'R' end  end  as c4,
        case  when D5.c10 is not null then D5.c10 when D1.c10 is not null then D1.c10 when D2.c2 is not null then D2.c2 when D3.c10 is not null then D3.c10 when D4.c2 is not null then D4.c2 end  as c5,
        case  when D2.c3 is not null then D2.c3 when D3.c11 is not null then D3.c11 when D1.c11 is not null then D1.c11 when D6.c3 is not null then D6.c3 end  as c6,
        ROW_NUMBER() OVER (PARTITION BY case  when D2.c3 is not null then D2.c3 when D3.c11 is not null then D3.c11 when D1.c11 is not null then D1.c11 when D6.c3 is not null then D6.c3 end , case  when D5.c10 is not null then D5.c10 when D1.c10 is not null then D1.c10 when D2.c2 is not null then D2.c2 when D3.c10 is not null then D3.c10 when D4.c2 is not null then D4.c2 end  ORDER BY case  when D2.c3 is not null then D2.c3 when D3.c11 is not null then D3.c11 when D1.c11 is not null then D1.c11 when D6.c3 is not null then D6.c3 end  ASC, case  when D5.c10 is not null then D5.c10 when D1.c10 is not null then D1.c10 when D2.c2 is not null then D2.c2 when D3.c10 is not null then D3.c10 when D4.c2 is not null then D4.c2 end  ASC) as c7
      from
        (
          (
            (
              (
                (select sum(case  when T867.MEASURE_TYPE  = 'DRIVING_VIOLATION_COUNT'  then T4832.MEASURE_WEIGHTING  / nullif( 100, 0) end ) as c1,
                    sum(case  when T867.MEASURE_TYPE  = 'ONDUTY_VIOLATION_COUNT'  then T4832.MEASURE_WEIGHTING  / nullif( 100, 0) end ) as c2,
                    sum(case  when T867.MEASURE_TYPE  = 'CUMULATIVE_VIOLATION_COUNT'  then T4832.MEASURE_WEIGHTING  / nullif( 100, 0) end ) as c3,
                    sum(case  when T867.MEASURE_TYPE  = 'LOGEDIT'  then T4832.MEASURE_WEIGHTING  / nullif( 100, 0) end ) as c4,
                    sum(case  when T867.MEASURE_TYPE  = 'OFF_DUTY_VIOLATION_COUNT'  then T4832.MEASURE_WEIGHTING  / nullif( 100, 0) end ) as c5,
                    sum(case  when T867.MEASURE_TYPE  = 'CUMULATIVE_VIOLATION_TIME'  then T4832.MEASURE_WEIGHTING  / nullif( 100, 0) end ) as c6,
                    sum(case  when T867.MEASURE_TYPE  = 'OFF_DUTY_VIOLATION_TIME'  then T4832.MEASURE_WEIGHTING  / nullif( 100, 0) end ) as c7,
                    sum(case  when T867.MEASURE_TYPE  = 'ONDUTY_VIOLATION_TIME'  then T4832.MEASURE_WEIGHTING  / nullif( 100, 0) end ) as c8,
                    sum(case  when T867.MEASURE_TYPE  = 'DRIVING_VIOLATION_TIME'  then T4832.MEASURE_WEIGHTING  / nullif( 100, 0) end ) as c9,
                    T13395.ENTITY_TYPE  as c10,
                    T806.NAME  as c11
                  from
                    (select 'Vehicle'  as ENTITY_TYPE   from dual  union select 'Driver'  from dual)  T13395,
                    W_CUSTOMER_DIM  T806 /* Company (W_CUSTOMER_DIM) */ ,
                    W_MEASURE_TYPE_DIM  T867  /* Metric (W_MEASURE_TYPE_DIM)  */ ,
                    W_THRESHOLD_FCT  T4832  /* Facts: Threshold  (W_THRESHOLD_FCT)  */
                  where  ( T806.CUSTOMER_WID = T4832.CUSTOMER_WID and T806.CUSTOMER_WID = 177415.0 and T867.MEASURE_TYPE_WID = T4832.MEASURE_TYPE_WID and T867.SOURCE_CODE  = 'HOS'  and T4832.CUSTOMER_WID = 177415.0 and T13395.ENTITY_TYPE  = 'Driver' and TO_DATE('1970-01-01 01:01:01' , 'YYYY-MM-DD  HH24:MI:SS') < TO_DATE('2010-08-30  17:43:51' , 'YYYY-MM-DD  HH24:MI:SS')  and (T867.MEASURE_TYPE  in ('CUMULATIVE_VIOLATION_COUNT',  'CUMULATIVE_VIOLATION_TIME',  'DRIVING_VIOLATION_COUNT',  'DRIVING_VIOLATION_TIME',  'LOGEDIT',  'OFF_DUTY_VIOLATION_COUNT',  'OFF_DUTY_VIOLATION_TIME',  'ONDUTY_VIOLATION_COUNT',  'ONDUTY_VIOLATION_TIME')) )
                  group by T806.NAME,  T13395.ENTITY_TYPE
                ) D1 full outer join (select count(distinct T16815.DRIVER_WID)  as c1,
                    T13395.ENTITY_TYPE  as c2,
                    T806.NAME  as c3
                  from
                    (select 'Vehicle'  as ENTITY_TYPE   from dual  union select 'Driver'  from dual)  T13395,
                    W_DAY_DIM T804  /* Date (W_DAY_DIM) */ ,
                    W_CUSTOMER_DIM  T806 /* Company (W_CUSTOMER_DIM)  */ ,
                    W_FLT_DRV_COVERAGE_AGG  T16815  /* Driver Fleet  Count (W_FLT_DRV_COVERAGE_AGG)  */
                  where  ( T804.DAY_WID = T16815.DATE_WID and T806.CUSTOMER_WID = T16815.CUSTOMER_WID and T806.CUSTOMER_WID  = 177415.0 and T13395.ENTITY_TYPE  = 'Driver'  and T16815.CUSTOMER_WID = 177415.0 and T804.DAY_DATE  between  TO_DATE('2010-08-26  00:00:00' , 'YYYY-MM-DD HH24:MI:SS') and TO_DATE('2010-09-09  00:00:00' , 'YYYY-MM-DD  HH24:MI:SS')  )
                  group by T806.NAME,  T13395.ENTITY_TYPE
                ) D2 On D1.c10 = D2.c2 and nvl(D1.c11 , 'q') = nvl(D2.c3 , 'q') and nvl(D1.c11 , 'z') = nvl(D2.c3 , 'z')) full outer join (select sum(case  when  T867.MEASURE_TYPE  = 'DRIVING_VIOLATION'  then T1021.MEASURE_COUNT  end ) as c1,
                  sum(case  when  T867.MEASURE_TYPE  = 'ONDUTY_VIOLATION'  then T1021.MEASURE_COUNT  end ) as c2,
                  sum(case  when  T867.MEASURE_TYPE  = 'CUMULATIVE_VIOLATION'  then T1021.MEASURE_COUNT  end ) as c3,
                  sum(case  when  T867.MEASURE_TYPE  = 'LOGEDIT'  then T1021.MEASURE_COUNT  end ) as c4,
                  sum(case  when  T867.MEASURE_TYPE  = 'OFF_DUTY_DRIVING'  then T1021.MEASURE_COUNT  end ) as c5,
                  sum(case  when  T867.MEASURE_TYPE  = 'CUMULATIVE_VIOLATION'  then T1021.MEASURE_VALUE  end ) as c6,
                  sum(case  when  T867.MEASURE_TYPE  = 'OFF_DUTY_DRIVING'  then T1021.MEASURE_VALUE  end ) as c7,
                  sum(case  when  T867.MEASURE_TYPE  = 'ONDUTY_VIOLATION'  then T1021.MEASURE_VALUE  end ) as c8,
                  sum(case  when  T867.MEASURE_TYPE  = 'DRIVING_VIOLATION'  then T1021.MEASURE_VALUE  end ) as c9,
                  T13395.ENTITY_TYPE  as c10,
                  T806.NAME  as c11
                from
                  (select 'Vehicle'  as ENTITY_TYPE   from dual  union select 'Driver'  from dual)  T13395,
                  W_DAY_DIM T804  /* Date (W_DAY_DIM) */ ,
                  W_CUSTOMER_DIM  T806  /* Company (W_CUSTOMER_DIM)  */ ,
                  W_MEASURE_TYPE_DIM  T867  /* Metric (W_MEASURE_TYPE_DIM)  */ ,
                  W_FLEET_AGG  T1021  /* Facts: Measure Fleet  (W_FLEET_AGG)  */
                where  ( T804.DAY_WID = T1021.DATE_WID and T806.CUSTOMER_WID = T1021.CUSTOMER_WID and T806.CUSTOMER_WID = 177415.0 and T867.MEASURE_TYPE_WID = T1021.MEASURE_TYPE_WID and T867.SOURCE_CODE  = 'HOS' and T1021.CUSTOMER_WID = 177415.0 and T13395.ENTITY_TYPE  = 'Driver' and (T867.MEASURE_TYPE  in ('CUMULATIVE_VIOLATION',  'DRIVING_VIOLATION',  'LOGEDIT',  'OFF_DUTY_DRIVING',  'ONDUTY_VIOLATION')) and T804.DAY_DATE  between  TO_DATE('2010-08-26  00:00:00' , 'YYYY-MM-DD  HH24:MI:SS') and TO_DATE('2010-09-09  00:00:00' , 'YYYY-MM-DD  HH24:MI:SS') )
                group by T806.NAME,  T13395.ENTITY_TYPE
              ) D3 On D3.c10 = case  when D1.c10 is not null  then D1.c10 when D2.c2 is not null  then D2.c2 end  and nvl(D3.c11  , 'q') = nvl(case  when D1.c11 is not null  then D1.c11 when  D2.c3 is not null  then D2.c3 end  , 'q') and nvl(D3.c11  , 'z') = nvl(case  when D1.c11 is not null  then D1.c11 when D2.c3 is not null  then D2.c3 end , 'z'))
```

```
select sum(case  when  T867.MEASURE_TYPE  = 'DRIVING_VIOLATION_COUNT'  then T4832.MEASURE_WEIGHTING  / nullif( 100, 0) end ) as c1,
sum(case  when  T867.MEASURE_TYPE  = 'ONDUTY_VIOLATION_COUNT'  then T4832.MEASURE_WEIGHTING  / nullif( 100, 0) end ) as c2,
sum(case  when  T867.MEASURE_TYPE  = 'CUMULATIVE_VIOLATION_COUNT'  then T4832.MEASURE_WEIGHTING  / nullif( 100, 0) end ) as c3,
sum(case  when  T867.MEASURE_TYPE  = 'LOGEDIT'  then T4832.MEASURE_WEIGHTING  / nullif( 100, 0) end ) as c4,
sum(case  when  T867.MEASURE_TYPE  = 'OFF_DUTY_VIOLATION_COUNT'  then T4832.MEASURE_WEIGHTING  / nullif( 100, 0) end ) as c5,
sum(case  when  T867.MEASURE_TYPE  = 'CUMULATIVE_VIOLATION_TIME'  then T4832.MEASURE_WEIGHTING  / nullif( 100, 0) end ) as c6,
sum(case  when  T867.MEASURE_TYPE  = 'OFF_DUTY_VIOLATION_TIME'  then T4832.MEASURE_WEIGHTING  / nullif( 100, 0) end ) as c7,
sum(case  when  T867.MEASURE_TYPE  = 'ONDUTY_VIOLATION_TIME'  then T4832.MEASURE_WEIGHTING  / nullif( 100, 0) end ) as c8,
sum(case  when  T867.MEASURE_TYPE  = 'DRIVING_VIOLATION_TIME'  then T4832.MEASURE_WEIGHTING  / nullif( 100, 0) end ) as c9,
T13395.ENTITY_TYPE  as c10,
T806.NAME  as c11
from
(select 'Vehicle'  as ENTITY_TYPE   from dual  union select 'Driver'  from dual)  T13395,
W_CUSTOMER_DIM  T806 /* Company (W_CUSTOMER_DIM)  */ ,
W_MEASURE_TYPE_DIM  T867  /* Metric (W_MEASURE_TYPE_DIM)  */ ,
W_THRESHOLD_FCT  T4832  /* Facts: Threshold  (W_THRESHOLD_FCT)  */
where  ( T806.CUSTOMER_WID = T4832.CUSTOMER_WID and T806.CUSTOMER_WID = 177415.0 and T867.MEASURE_TYPE_WID = T4832.MEASURE_TYPE_WID and T867.SOURCE_CODE  = 'HOS'  and T4832.CUSTOMER_WID = 177415.0 and T13395.ENTITY_TYPE  = 'Driver' and TO_DATE('1970-01-01  01:01:01' , 'YYYY-MM-DD  HH24:MI:SS') < TO_DATE('2010-08-30  17:43:51' , 'YYYY-MM-DD  HH24:MI:SS')  and (T867.MEASURE_TYPE  in ('CUMULATIVE_VIOLATION_COUNT',  'CUMULATIVE_VIOLATION_TIME',  'DRIVING_VIOLATION_COUNT',  'DRIVING_VIOLATION_TIME',  'LOGEDIT',  'OFF_DUTY_VIOLATION_COUNT',  'OFF_DUTY_VIOLATION_TIME',  'ONDUTY_VIOLATION_COUNT',  'ONDUTY_VIOLATION_TIME')) )
group by T806.NAME,  T13395.ENTITY_TYPE
) D1 full outer join (select count(distinct T16815.DRIVER_WID)  as c1,
x
```

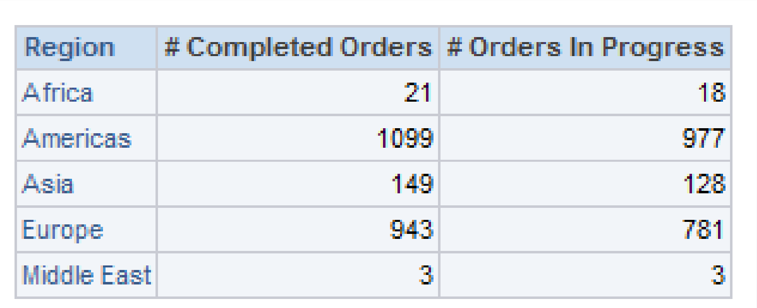
```
select count(distinct T16815.DRIVER_WID)  as c1,
T13395.ENTITY_TYPE  as c2,
T806.NAME  as c3
from
(select 'Vehicle'  as ENTITY_TYPE   from dual  union select 'Driver'  from dual)  T13395,
W_DAY_DIM T804  /* Date (W_DAY_DIM) */ ,
W_CUSTOMER_DIM  T806 /* Company (W_CUSTOMER_DIM)  */ ,
W_FLT_DRV_COVERAGE_AGG  T16815  /* Driver Fleet  Count (W_FLT_DRV_COVERAGE_AGG)  */
where  ( T804.DAY_WID = T16815.DATE_WID and T806.CUSTOMER_WID = T16815.CUSTOMER_WID and T806.CUSTOMER_WID  = 177415.0 and T13395.ENTITY_TYPE  = 'Driver'  and T16815.CUSTOMER_WID = 177415.0 and T804.DAY_DATE  between  TO_DATE('2010-08-26  00:00:00' , 'YYYY-MM-DD HH24:MI:SS') and TO_DATE('2010-09-09  00:00:00' , 'YYYY-MM-DD HH24:MI:SS') )
group by T806.NAME,  T13395.ENTITY_TYPE
) D2 On D1.c10 = D2.c2 and nvl(D1.c11 , 'q') = nvl(D2.c3 , 'q') and nvl(D1.c11 , 'z') = nvl(D2.c3 , 'z')) full outer join (select sum(case  when  T867.MEASURE_TYPE  = 'DRIVING_VIOLATION'  then T1021.MEASURE_COUNT  end ) as c1,
sum(case  when  T867.MEASURE_TYPE  = 'ONDUTY_VIOLATION'  then T1021.MEASURE_COUNT  end ) as c2,
sum(case  when  T867.MEASURE_TYPE  = 'CUMULATIVE_VIOLATION'  then T1021.MEASURE_COUNT  end ) as c3,
sum(case  when  T867.MEASURE_TYPE  = 'LOGEDIT'  then T1021.MEASURE_COUNT  end ) as c4,
sum(case  when  T867.MEASURE_TYPE  = 'OFF_DUTY_DRIVING'  then T1021.MEASURE_COUNT  end ) as c5,
sum(case  when  T867.MEASURE_TYPE  = 'CUMULATIVE_VIOLATION'  then T1021.MEASURE_VALUE  end ) as c6,
sum(case  when  T867.MEASURE_TYPE  = 'OFF_DUTY_DRIVING'  then T1021.MEASURE_VALUE  end ) as c7,
sum(case  when  T867.MEASURE_TYPE  = 'ONDUTY_VIOLATION'  then T1021.MEASURE_VALUE  end ) as c8,
sum(case  when  T867.MEASURE_TYPE  = 'DRIVING_VIOLATION'  then T1021.MEASURE_VALUE  end ) as c9,
T13395.ENTITY_TYPE  as c10,
T806.NAME  as c11
from
(select 'Vehicle'  as ENTITY_TYPE   from dual  union select 'Driver'  from dual)  T13395,
W_DAY_DIM T804  /* Date (W_DAY_DIM) */ ,
W_CUSTOMER_DIM  T806 /* Company (W_CUSTOMER_DIM)  */ ,
W_MEASURE_TYPE_DIM  T867  /* Metric (W_MEASURE_TYPE_DIM)  */ ,
W_FLEET_AGG  T1021  /* Facts: Measure Fleet  (W_FLEET_AGG)  */
where  ( T804.DAY_WID = T1021.DATE_WID and T806.CUSTOMER_WID = T1021.CUSTOMER_WID and T806.CUSTOMER_WID = 177415.0 and T867.MEASURE_TYPE_WID = T1021.MEASURE_TYPE_WID and T867.SOURCE_CODE  = 'HOS' and T1021.CUSTOMER_WID = 177415.0 and T13395.ENTITY_TYPE  = 'Driver' and (T867.MEASURE_TYPE  in ('CUMULATIVE_VIOLATION',  'DRIVING_VIOLATION',  'LOGEDIT',  'OFF_DUTY_DRIVING',  'ONDUTY_VIOLATION')) and T804.DAY_DATE  between  TO_DATE('2010-08-26  00:00:00' , 'YYYY-MM-DD  HH24:MI:SS') and TO_DATE('2010-09-09  00:00:00' , 'YYYY-MM-DD  HH24:MI:SS') )
group by T806.NAME,  T13395.ENTITY_TYPE
```

- Database CPU had to parse and merge recordsets
- Take advantage of OBI CPU power if database is overloaded
- Recent test yielded 9x improvement *at load!*

Common Tuning: Incorrect Nullability

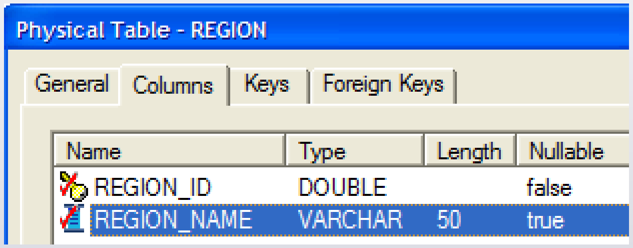
- A simple thing like the Nullable flag in the physical layer can have negative effects

- Sample query requires 2 Fact tables



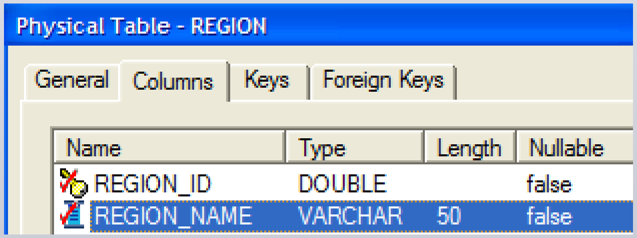
Region	# Completed Orders	# Orders In Progress
Africa	21	18
Americas	1099	977
Asia	149	128
Europe	943	781
Middle East	3	3

- If REGION_NAME is set to allow NULLS:
  - Join between the two is inefficient
  - **On nvl(D1.c2 , 'q') = nvl(D2.c2 , 'q') and nvl(D1.c2 , 'z') = nvl(D2.c2 , 'z')**



Name	Type	Length	Nullable
REGION_ID	DOUBLE		false
REGION_NAME	VARCHAR	50	true

- If REGION_NAME is set to NOT NULL
  - **On D1.c2 = D2.c2**

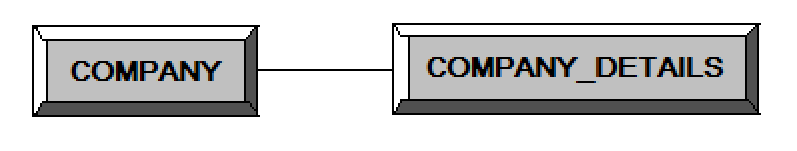


Name	Type	Length	Nullable
REGION_ID	DOUBLE		false
REGION_NAME	VARCHAR	50	false

- Make sure the physical database is set correctly ***before*** import of tables

Common Tuning: Extra Tables

- If an additional table is added to a query that is not needed, performance will suffer
  - Worse: Result could be wrong!



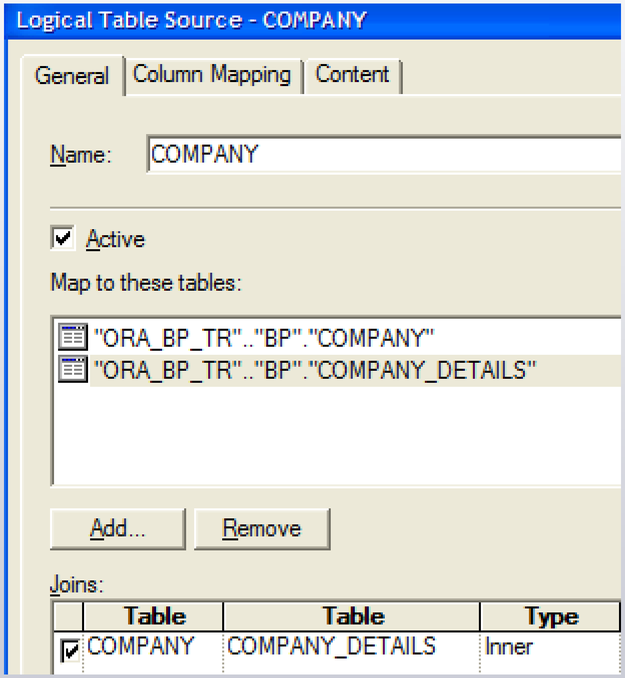
COMPANY

COMPANY_DETAILS

- This example has a 1:1 extension table with Company using a complex join
  - There are no 1:1 joins in the Physical Layer

- A query of COMPANY and ORDERS table will also include COMPANY_DETAILS

- This happens when there is a **complex** join between 2 tables in a Logical Table Source



Name: COMPANY

☑ Active

Map to these tables:

"ORA_BP_TR".."BP"."COMPANY"
"ORA_BP_TR".."BP"."COMPANY_DETAILS"

Joins:

	Table	Table	Type
☑	COMPANY	COMPANY_DETAILS	Inner

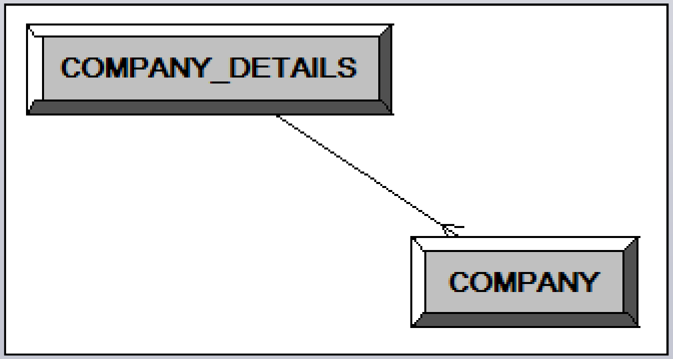
```
select T188.COMPANY_NAME as c1,
          count(T174.ORDER_ID) as c2,
          T188.COMPANY_ID as c3
       from
            COMPANY T188,
            COMPANY_DETAILS T1960,
            ORDERS T174
       where  ( ….
```

Common Tuning: Extra Tables

- To resolve, change the join to be a FK join
  - Keep the main table on the *many* side of the 1:M

- OBI will not include a ‘higher’ grain table unless needed

- Trick OBI to use the optional table only when needed and not all of the time



```
select T188.COMPANY_NAME as c1,
        count(T174.ORDER_ID) as c2,
        T188.COMPANY_ID as c3
     from
          COMPANY T188,
          ORDERS T174
     where  ( ….
```

Common Tuning: No Group Bys

- One of the biggest performance problems is missing Group Bys

- Detailed data is returned to the OBI Server which does the Group By
  - Usually returns a much larger dataset than ideal
  - Can be as bad as Gigabytes vs. 1 Kbyte

- Variety of reasons this can occur
  - Using a function not supported by the database
    - E.g., Count(Distinct) does not exist in MS Access
    - All detailed records are returned to OBI Server for aggregation
  - Using a function not enabled in the Database Features tab
  - Configuration error, usually with the Content tab

- Start with the Features tab to see if support is enabled

Database Features Tab

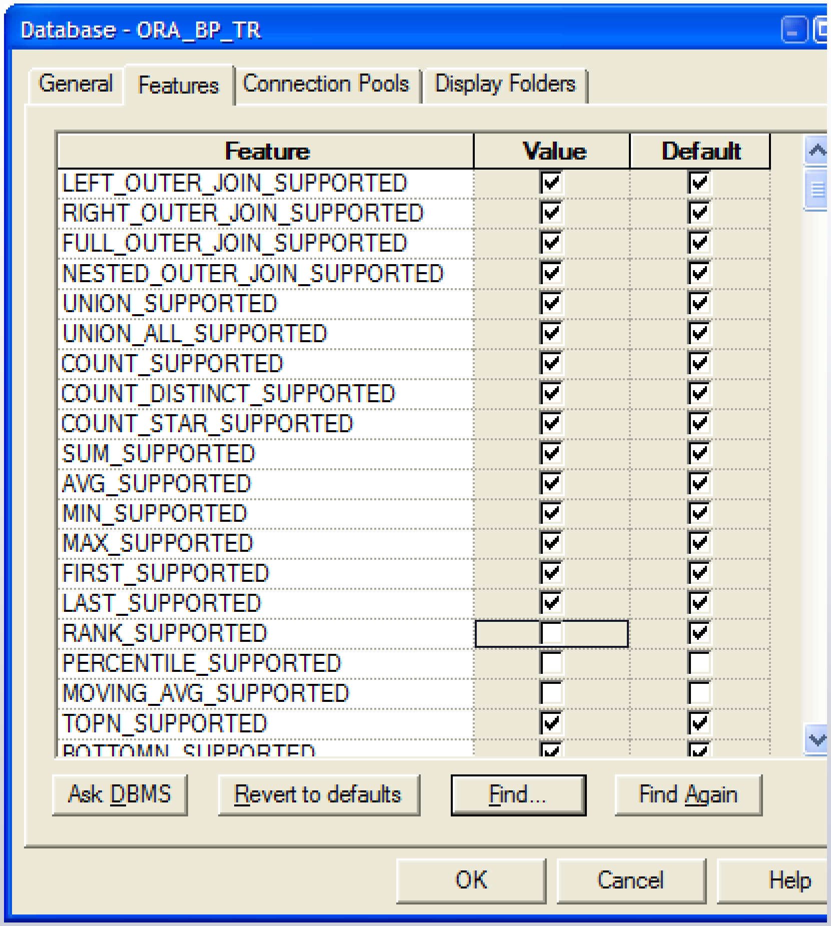
- The Features tab controls what SQL is generated

- It is uncommon to alter the defaults

- Here, the RANK() function is not supported
  - Any Rank Metrics will require all detail records to be sent to the OBI Server
  - OBI server then does the Rank()

- A simple check box may have benefits of orders of magnitude

- These are global settings – Regression test!

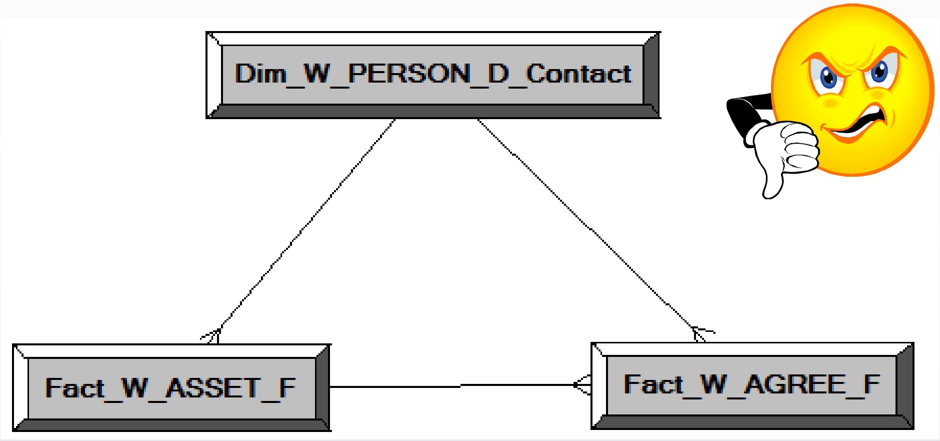


Feature	Value	Default
LEFT_OUTER_JOIN_SUPPORTED	☑	☑
RIGHT_OUTER_JOIN_SUPPORTED	☑	☑
FULL_OUTER_JOIN_SUPPORTED	☑	☑
NESTED_OUTER_JOIN_SUPPORTED	☑	☑
UNION_SUPPORTED	☑	☑
UNION_ALL_SUPPORTED	☑	☑
COUNT_SUPPORTED	☑	☑
COUNT_DISTINCT_SUPPORTED	☑	☑
COUNT_STAR_SUPPORTED	☑	☑
SUM_SUPPORTED	☑	☑
AVG_SUPPORTED	☑	☑
MIN_SUPPORTED	☑	☑
MAX_SUPPORTED	☑	☑
FIRST_SUPPORTED	☑	☑
LAST_SUPPORTED	☑	☑
RANK_SUPPORTED	□	☑
PERCENTILE_SUPPORTED	□	□
MOVING_AVG_SUPPORTED	□	□
TOPN_SUPPORTED	☑	☑
BOTTOMN_SUPPORTED	☑	☑

Know the proper SQL beforehand!

Common Tuning: Fact-to-Fact Joins

- Fact to Fact joins are a no-no
  - Joining 2 very large tables together

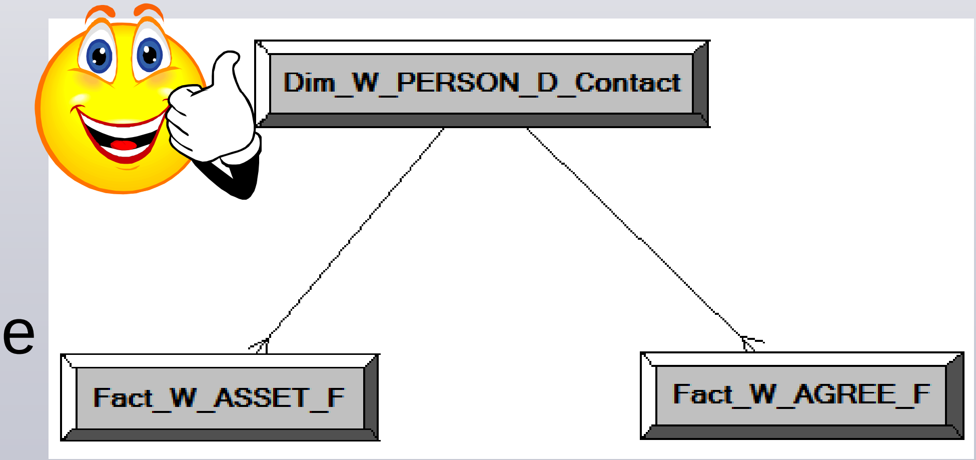


Dim_W_PERSON_D_Contact

Fact_W_ASSET_F

Fact_W_AGREE_F

- Instead, use conformed dimensions
  - Two queries will be issued instead
  - Each query will have a small result set
  - The small result sets are then joined



Dim_W_PERSON_D_Contact

Fact_W_ASSET_F

Fact_W_AGREE_F

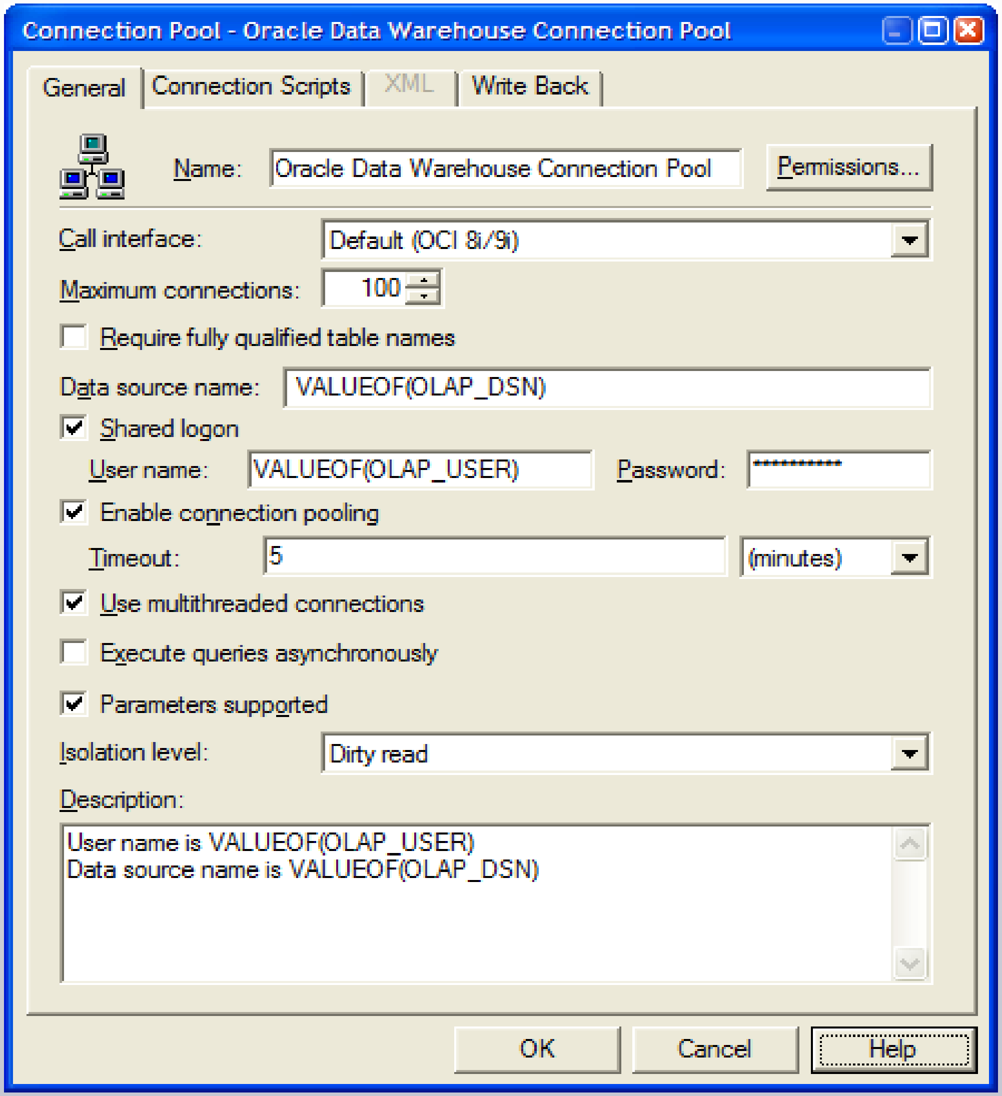
Connection Pools

- Most of the default settings are rarely changed

✓ Make sure to use ***native*** Call Interface and not ODBC!

✓ Set to support 10-20% of concurrent users X # logical queries per page

✓ Connection Pooling should be enabled



Connection Pool - Oracle Data Warehouse Connection Pool

Name: Oracle Data Warehouse Connection Pool

Call interface: Default (OCI 8i/9i)

Maximum connections: 100

☐ Require fully qualified table names

Data source name: VALUEOF(OLAP_DSN)

☑ Shared logon

User name: VALUEOF(OLAP_USER) Password: ***********

☑ Enable connection pooling

Timeout: 5 (minutes)

☑ Use multithreaded connections

☐ Execute queries asynchronously

☑ Parameters supported

Isolation level: Dirty read

Description:

User name is VALUEOF(OLAP_USER)
Data source name is VALUEOF(OLAP_DSN)

Caching

- OBI has a sophisticated caching engine

- It caches the results of SQL queries
  - Siebel Analytics caches the whole Logical query or not at all
  - OBI 10.x caches portions of a multi-part logical query

- Caches can be reused across queries
  - Across users depending on security
  - Currently cannot share cache across clustered servers (except iBots)

- Cache repopulation
  - Manual table by table cache durations
  - Using an Event Polling Table is a flexible approach
    - Integration with ETL via a table that is periodically polled

- Caching can be pre-loaded for users via iBots to pre-execute dashboards and reports
  - First users in the morning will cache hit and experience good performance

Caching – Poor Man’s Performance

- Caching can help provide excellent performance in many scenarios

- It is not a replacement for any of the topics covered
  - Caching Ad-hoc is impossible
  - Caching real-time systems is impossible
  - Caching when using tight security becomes very difficult
  - Caching highly interactive dashboards becomes difficult to impossible

- Cherry on top of a great sundae

Follow OBI RPD Best Practices

If you :

- Follow good Dimensional Modeling design
- Follow OBI configuration best practices
- Set up database performance features properly

You will generally have a sound, good performing model

- Keep in mind these best practices
  - Always fill out the content tab
    - The Content tab controls which table OBI uses and how it writes much of its SQL
  - Verify your Physical Layer Null Flags
  - Use conformed dimensions
  - Replace large or complex Views with tables
  - Make your Physical Model look like your Business Model
  - Design your Dimensional Models based on your reports
    - Keep ad-hoc flexibility in mind

User Interface

General UI Recommendations

From a UI design and build perspective, a dashboard page's performance depends on:

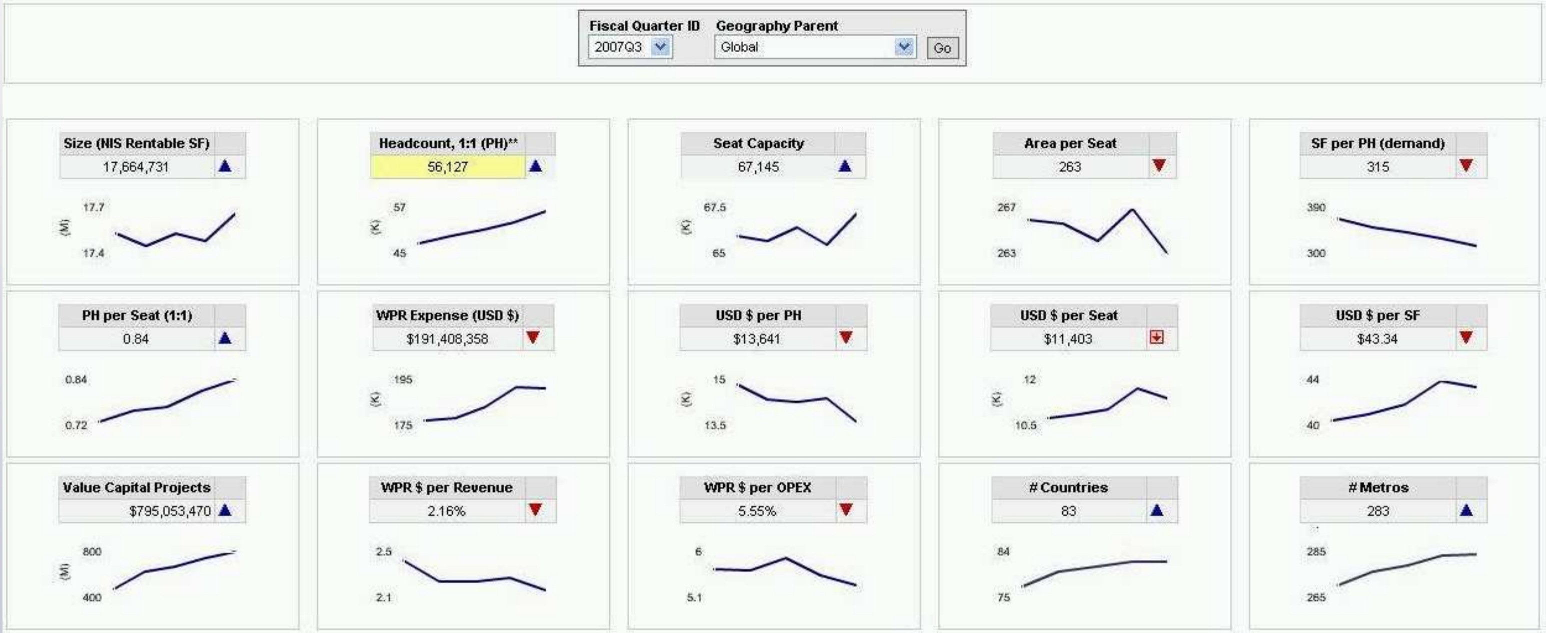
1. The quantity and amount of data loaded into the prompts
   - Try to limit the number of rows returned
     - Use Multi-selects & constraining
   - Be mindful of users with slow network connections
   - Constraining across dimensions will be relatively slow

2. Use fewer reports but more views
   - Each report dragged to a page makes a new request

     ➔ New Logical SQL & new Physical SQL
   - Use Compound Layouts as much as possible instead of multiple report objects
     - Combines multiple views into one request

General UI Recommendations

3. The number of reports & report views on a page
   - More reports are more UI objects to manage and draw
     - Not to mention more queries!
   - This sample page will generate 32 Logical queries!



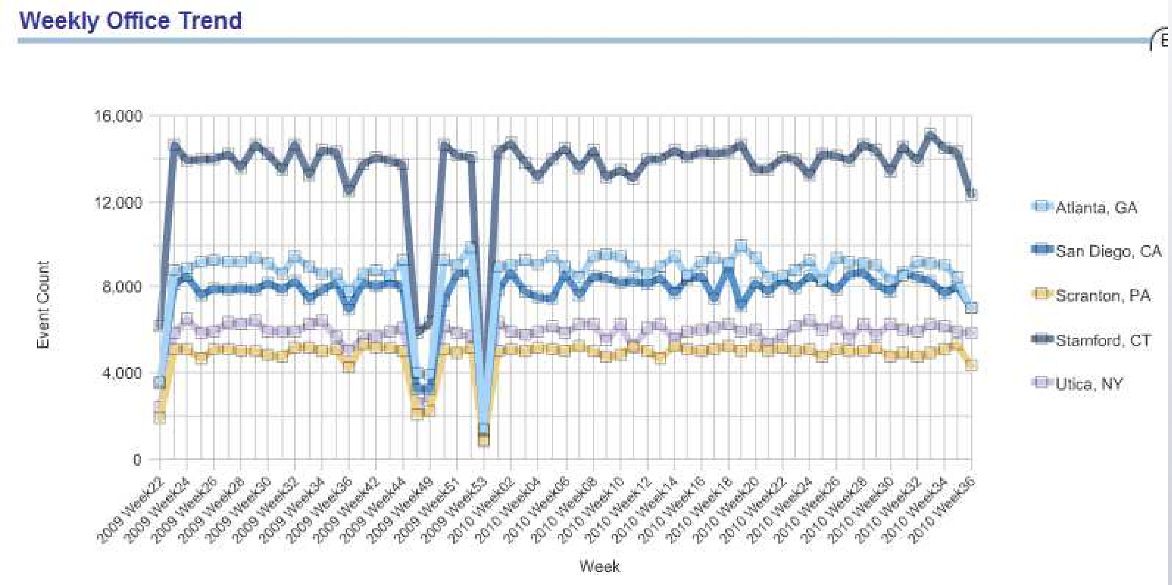
Fiscal Quarter ID: 2007Q3
Geography Parent: Global
Go

Size (NIS Rentable SF): 17,664,731 ▲
Headcount, 1:1 (PH)**: 56,127 ▲
Seat Capacity: 67,145 ▲
Area per Seat: 263 ▼
SF per PH (demand): 315 ▼
PH per Seat (1:1): 0.84 ▲
WPR Expense (USD $): $191,408,358 ▼
USD $ per PH: $13,641 ▼
USD $ per Seat: $11,403 ▼
USD $ per SF: $43.34 ▼
Value Capital Projects: $795,053,470 ▲
WPR $ per Revenue: 2.16% ▼
WPR $ per OPEX: 5.55% ▼
Countries: 83 ▲
Metros: 283 ▲

Secret: Hidden Sections / Conditional Sections are always executed!

General UI Recommendations

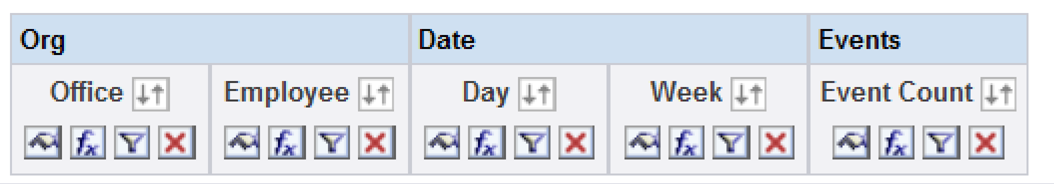
4. Sync up report criteria columns with view grain
   - Remove extra columns from the criteria tab
   - Lower dimensionality increases the record set



Report Grain:

Week X Office

e.g, 26 X 5 = 130 rows



Org		Date		Events
Office	Employee	Day	Week	Event Count

Query Grain:

Day X Employee

e.g., 182 X 400 = 72,800 rows!

General UI Recommendations

5. Reduce the quantity of data each report shows
   - Use paging controls or place large data dumps on separate pages
   - Pivot Tables do not have the ability to limit the # of rows displayed at once
   - Pre-filter before user prompt selection
     - Use default values in the report

6. Seek out and destroy UNION queries
   - Use the other techniques to eliminate them
   - UNIONs have other report & UI limitations anyway
   - Worst case, use UNION ALL instead – databases prefer

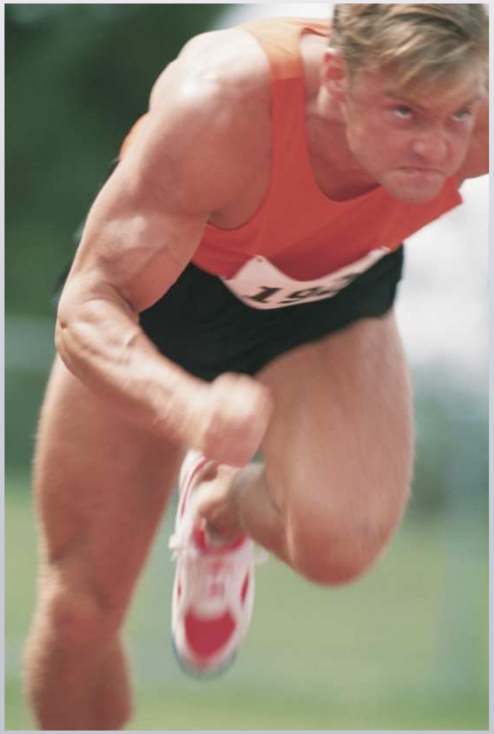
7. Avoid fully flexible date ranges
   - Fix time selections (e.g., ‘last 2 weeks’)
   - Allows aggregates to be built

Summary

- OBI is a SQL generation engine

- The database does the heavy lifting

- Make it easy on the poor database!
  - Put everything it needs together in one place
  - Organize data in an easy to access way
  - Tune the database to be fast
  - Ask good questions to it via good SQL



CONTACT INFORMATION

kpi partners

KPI Partners Sales Team

Visit Booth # 2235

Kusal Swarnakar
Managing Partner
email: kusal.swarnakar@kpipartners.com
phone: (925) 984-1371

Norman Dy
Director – US West
email:
norman.dy@kpipartners.com
phone: (619) 245-5090

Jimmy Dahlan
Director – US West
email:
jimmy.dahlan@kpipartners.com
phone: (408) 981-4284

Jaime Seagraves
Director – US North East
email:
jaime.seagraves@kpipartners.com
phone: (630) 854-0450

Mark Joslin
Director – South East
email: mark.joslin@kpipartners.com
phone: (336) 882-8185

Keith Weisz
Director – US Central
email: keith.weisz@kpipartners.com
phone: (816) 304-1005

Think ORACLE BI
Think KPI...

WWW.KPIPARTNERS.COM

ORACLE CERTIFIED PARTNER