OBIEE 11g Data Modeling Best Practices

Mark Rittman, Director, Rittman Mead
UKOUG Technical & EBS Conference, Birmingham November 2010

T : +44 (0) 8446 697 995 or (888) 631 1410 (USA)  E : enquiries@rittmanmead.com  W: www.rittmanmead.com
Introductions

- Mark Rittman, Co-Founder of Rittman Mead
- Oracle ACE Director, specialising in Oracle BI&DW
- 14 Years Experience with Oracle Technology
- Regular columnist for Oracle Magazine
- Author of forthcoming Oracle Press book on OBIEE 11g
- Email: mark.rittman@rittmanmead.com
- Twitter: @markrittman
Rittman Mead

- Oracle BI&DW Project Specialists providing consulting, training and support
- Clients in the UK, USA, Europe, Middle-East
- Voted UKOUG BI Partner of the Year 2008
- Group companies in USA and India
- Come and see us to discuss your requirements in more detail
The Rittman Mead Blog

- The longest-running Oracle blog, established in June 2003
- http://www.rittmanmead.com/blog
- Over 1300 postings
- Covers Oracle tech, including
  - OBIEE
  - OWB
  - Oracle Essbase
  - Oracle Database
  - BI Publisher
  - Fusion Middleware
- Early news on events
- 10 authors
Rittman Mead Training Days and BI Forum Events

Friday, 10 December 2010
Rittman Mead Oracle BI Training Days 2010/2011

- Running in London, San Francisco, Johannesburg and Bangalore, Q1 2011
- Three days of intense OBIEE 11gR1 hands-on training
- Led by the Rittman Mead 11g Beta Testing team
Oracle Business Intelligence 11g

- The latest release of OBIEE (Oracle BI Enterprise Edition)
- Major changes to front-end, integration, architecture
- Objective is to make it more “enterprise-ready”
- Closely integrated with Oracle Fusion Middleware 11g
New User Interface, Ad-Hoc Querying, Scorecards, Dashboards

- Updated, task-based interface based on Oracle ADF
- Integrated reporting, ad-hoc analysis, dashboards, scorecard, alerting
- Reporting technology that will be embedded in Fusion Applications
- Web 2.0, fully thin-client
Action Framework, and Integration with Business Processes

- Web Service, HTTP and Java integration with applications, business processes
- “closed-loop” business intelligence
- Leverages Oracle/standard WS technologies
- “Actionable Intelligence”
Based on Fusion Middleware 11g Platform

- Fully-integrated with Oracle Fusion Middleware 11g
- Common security (Oracle Identity Management, Oracle Access Manager)
- Built on WebLogic Server
- Managed through Fusion Middleware Control (Enterprise Manager)
- Built using JDeveloper (ADF)
Questions This Presentation Will Answer

- What is the purpose of the OBIEE data model (repository, RPD)?
- What are your objectives as a repository developer?
- What are some good practices for developing a repository?
- What are logical table sources, and federation, used for?
- What are the new features in OBIEE 11gR1 for repository modeling?
OBIEE Metadata and Data Layers

• The **Web Catalog** contains definitions of reports, alerts dashboards, KPIs, based on...

• A **Semantic Model** made up of Metrics, dimensions, hierarchies and calculations, derived from...

• **Physical Data Sources**, such as relational databases, OLAP Cubes and application source adapters
Flow of Data Through the Three-Layer Semantic Model

- Simplification of the Data Model
- Integration of Disparate Data Sources
- Addition of Business Logic and Calculations
Semantic Model Design Objectives

- Create a simple, logical, dimensional model for users to query
- Integrate where possible disparate data sources using conformed dimensions
- Add dimensions and hierarchies to facilitate drilling to detail
- Add common calculations and aggregations
- Present data to users through subject areas (a.k.a. data marts)
- Add role-based data and subject-area security
OBIEE Data Modeling : Basic Good Practices

- Ideally start with a data warehouse data source
- Think in terms of dimensional modeling, most importantly for the business model
- Define PK and FK keys in the physical layer, against aliased copies of tables
- Create outline business model, then map across physical columns, renaming to business terminology and adding business logic (calculations, drill paths)
- Publish one subject area per logical star (fact + dimensions)
- Develop offline initially, reduces contention and check-in/check-out
  - 11gR1 now supports >5 online developers, but still requires checkin/checkout
- Ensure dimension member counts are (reasonably) accurate
- Use the semantic model to integrate, and simplify, what can be complex and disparate data sources
Step 1: Importing Data Source Metadata

- Assuming source data is a DW, import metadata into enterprise semantic model
- Create aliases that describe the role of each imported table, add additional aliases for multiple roles played by tables (i.e. time, person etc)
- Define PK and FK constraints against alias tables
- Resolve any issues around recursive joins etc
Step 2 : Define Outline Business Model

- Define, at a high level, the business model that will represent your data
- Aim for single model; in short-term, multiple models may be required
  ‣ Migrate to single model as dimensions become conformed
- Define high level logical tables, starting with logical facts
Step 3: Map Physical Columns into Business Model

- Starting with the logical fact columns, drag and drop physical columns into the outline logical model, and rename columns to reflect business terms.
- Do not drag and drop fact table FK columns across:
  - These are referenced through physical model.
- Only drag those columns needed for reporting.
- Define default aggregation for measures.
Step 4: Define Logical Dimensions, Levels and Hierarchies

- Define logical dimensions, hierarchies and levels
- Ensure logical level keys are unique
- Add chronological key(s) for time dimensions
  ‣ Try to define chron. key at each time dim level
- Add level-based measures if required
- Important: Enter values for “number of elements at this level” (used for agg. navigation)
Step 5 : Define Calculations

- Define calculations within the business model
- Can include regular calculations, analytic function, aggregations, time-series functions
- Uses OBIEE syntax; will either be function-shipped to database functions, or performed by BI Server

![Expression Builder - Derived logical column](image-url)

T : +44 (0) 8446 697 995 or (888) 631 1410 (USA)  E : enquiries@rittmanmead.com  W: www.rittmanmead.com
Step 6 : Create Subject Areas for each Logical Fact

- Make business area available for reporting through publishing as subject area(s)
- Either one subject area per business model, or (better) one per logical fact table
  - In 11gR1, analyses can be written across subject areas
- Also in 11gR1, logical dimensions now appear in subject areas (more on this later)
Step 7: Define Security Policy, Providers and Filters

- Use Security Manager to define (10g) or view (11g) users, application groups/roles
- Apply row-level filters to business model tables and columns
- Apply permissions to subject areas, tables and columns
- From 11g onwards, best practice is to define LDAP sources through WLS (or supporting JEE server)
Variation: Modeling Against OLTP Sources

- For OLTP data sources, it may be better to completely create the business model first, then map in physical column sources.
- Allows you to manipulate logical table sources to denormalize data.
- OLTP schema may require modeling, manipulation to map into business dimensional model.
What is a Logical Table Source?

- Facts and dimensions are **mapped** to physical sources **Logical Table Sources**
- LTS are chosen at run time based on RPD definitions (level mapping, fragmentation, federation etc)
When Do We Create Multiple Logical Table Sources?

- Each LTS represents a mapping of logical columns to physical sources.
- Additional LTS mappings are required when one logical SQL statement couldn’t span all required data sources:
  - Example: aggregate and detail-level data sources (known as **Horizontal Federation**).
  - Example: two sources on different physical databases (known as **Vertical Federation**).
- 1-n LTS mapping may be used for an analyses, depending on context of query:
  - Does analysis require data spanning multiple physical databases?
  - Does analysis require data of differing levels of granularity?
When Do We Extend an Existing Logical Table Source?

- Typically used when denormalizing fact or dimension sources
- Allows a single LTS mapping to extend to additional joined physical tables
- Preferable to adding new LTS mappings, when a single logical SQL could cover both tables
How Is SQL Generated Affected By Multiple LTSs?

- Strategy for the BI Server is to try and push-down joins where possible.
- In most cases, multiple LTS will resolve to >1 physical SQL query (or MDX, etc).
- However, many queries (cross-fact etc) involve >1 fact LTS, and these can in fact be resolved as a single physical SQL.
  - **WITH_CLAUSE_SUPPORTED** = **YES** in Database Features setting.
- Guideline: prepare for >1 physical SQL, but BI Server will try and push joins down to database if possible.

![Database - orcl](image)

```sql
WITH
SANITHO AS (select sub(T2830.UNIT_COST) as c1, T2874.PROD_SUBCATEGORY_DESC as c2
from PRODUCTS T2874,
COSTS T2830
where ( T2830.PROD_ID = T2874.PROD_ID )
group by T2874.PROD_SUBCATEGORY_DESC ),
SANITH1 AS (select sub(T2911.AMOUNT_SOLD) as c1, T2874.PROD_SUBCATEGORY_DESC as c2
from PRODUCTS T2874,
SALES T2911
where ( T2830.PROD_ID = T2911.PROD_ID )
group by T2874.PROD_SUBCATEGORY_DESC ),
SANITH2 AS (sub(T2874.UNIT_COST) as c1, T2874.PROD_SUBCATEGORY_DESC as c2
from T2874
where T2874.PROD_ID = '123')

select distinct SANITH0.c1 as c1, SANITH1.c1 as c2,
   when SANITH0.c2 is not null then SANITH0.c2
   when SANITH1.c2 is not null then SANITH1.c2
   end as c3
   ...
order by c3
```
New Features in OBIEE 11gR1 For Data Modeling

• Support for Ragged, Skip-Level and Parent-Child Hierarchies
• Hierarchical Subject Area Columns
• Lookup Logical Tables
• ID Columns for Descriptive Columns
• LTS Priority Ordering
• Repository Passwords
• Deployment of Repository Files using Fusion Middleware Control
Ragged and Skip-Level Logical Dimensions

- Level-based hierarchies may have skip-levels, or ragged hierarchies
- Skip-levels are when members may not have an immediate parent
- Ragged hierarchies are when leaf members are not all at the same level
Ragged and Skip-Level Logical Dimensions

- Level-based hierarchies may have skip-levels, or ragged hierarchies
- Skip-levels are when members may not have an immediate parent
- Ragged hierarchies are when leaf members are not all at the same level
Ragged and Skip-Level Logical Dimensions in the Dataset

- Skip-levels and Ragged hierarchies are represented in the underlying table through NULLs.
Creating Ragged and Skip-Level Logical Dimensions

- Select Ragged and/or Skip-Level in the logical dimension properties
  - Do not select for non-ragged, non-skip, as detecting NULLs adds overhead
- For ragged hierarchies, add surrogate key to ensure consistent logical PK
  - Delete this additional level when you create the presentation hierarchy
Parent-Child (Value-Based) Dimensions

- Some data sets contain recursive, parent-child (value-based) hierarchies
  - Hierarchy is defined in the data, rather than the column relationships
- Typically used for organizations and other implicitly ragged hierarchies
- Difficult to model relationally as recursive SQL queries would be required to traverse
Creating Parent-Child Hierarchies

- Command to create logical dimensions now includes Parent-Child option
- Initial logical dimension dialog then prompts you to select **Parent Column**
- Once selected, you then need to define the **closure table** to resolve recursion
Parent-Child Hierarchy Closure Tables

- To avoid the need for recursive SQL, a closure table is defined for the hierarchy
- Iterates through the hierarchy once, speeds up all subsequently queries
- Creates two scripts; one to create closure table, one to populate it
- Population script will need to be re-run each time underlying data changes
Parent-Child vs. Ragged Hierarchies: Which to Use?

- Parent-child hierarchies are naturally ragged, so which should you choose?
- Depends on the data structures you are reporting on
  ‣ If each hierarchy level is distinct and named, go for level-based with ragged hierarchy option
  ‣ If hierarchy levels are not named, or data is stored in RDBMS recursively, go for parent-child hierarchy
- Parent-child takes more maintenance, so only use if necessary
Hierarchical Columns and Essbase Sources

- Essbase data sources can be amended to use parent-child ("value") hierarchies
- Avoids issue where changing outlines requires re-import
- Business Model columns no longer based on outline generations
- New generations can be added dynamically as Essbase outline evolves
Essbase Accounts Dimensions, Aliases and UDAs

- Accounts dimensions now brought in as a dimension, with a single measure
- UDAs can now be imported, and act as filters
- Aliases can be imported, and are represented as logical columns
Hierarchical Columns

- Logical dimensions in the business model can now be dragged across to create Hierarchical Columns in subject areas
- One hierarchical column per logical dimension hierarchy
- Allows in-column drilling in analyses
Lookup Tables and Function

- LOOKUP function introduced with OBIEE 11g allows lookups on logical and physical tables
- Useful for several scenarios
  - Pushing currency conversions to separate calculations
  - Accessing columns that cannot use GROUP BY (e.g. CLOB)
  - Avoiding unnecessary outer joins in LTS
Lookup Logical Tables

- Allows you to define a logical table as being a lookup table
  - Removes check as to whether it is a fact, or joined to a fact table
- Can be accessed via a logical calculation using SPARSE or DENSE lookup
- BI Server generates separate lookup SQL query, joined to main dataset by BI Server in-memory
Accessing Lookup Tables

- Lookup tables are accessed using the new LOOKUP function
- Can be SPARSE (outer join) or DENSE (inner join)
- Physical table lookups push join to SQL query; logical creates two SQL queries

```
Lookup(DENSE <<lookupColumn>>, <<sourceKeyorExpression>>)  
Lookup(SPARSE <<lookupColumn>>, <<alternateColumn>>, <<sourceKeyorExpression>>)  
```
ID Columns for Descriptive Columns (Double-Columns)

- In OBIEE 10g, it was difficult to define dashboard prompts that display descriptions, but pass back IDs
- OBIEE 11gR1 allows you to define an ID column for descriptive column(s)
  - Multiple columns can reference same ID
- Dashboard prompts are aware of IDs
Double-Columns in Use

- User can select using just descriptions, or can elect to show IDs as well
- Set analyses prompt to “Is Prompted”
- SQL will automatically filter on IDs, rather than descriptions

WITH
SAWITH0 AS (select sum(T117.REV_AMT) as c1, T107.PROD_CAT_DESC as c2, T107.PROD_ID as c3
from GCBC_SALES.PRODUCTS T107, GCBC_SALES.SALES T117
where (T107.PROD_ID = T117.PROD_ID
and (T107.PROD_ID in (12, 17, 31)))

group by T107.PROD_CAT_DESC, T107.PROD_ID),

SAWITH1 AS (select 0 as c1,
D1.c2 as c2,
D1.c3 as c3,
D1.c1 as c4,
sum(D1.c1) as c5
from SAWITH0 D1

group by D1.c1, D1.c2, D1.c3)

select D1.c1 as c1,
D1.c2 as c2,
D1.c3 as c3,
D1.c4 as c4,
sum(D1.c5) over (partition by D1.c2)
as c5

from SAWITH1 D1
**LTS Priority Group Order**

- A single logical table can have multiple logical table sources defined.
- Usually, the selection of LTS by the BI Server is straightforward and defined by the context of the query.
- When multiple aggregate LTS are defined, however, which one is chosen can be difficult to determine (based on *Number of Members At This Level*).
- OBIEE 11gR1 LTS Priority Group Order makes this more explicit:
  - Lower value = higher priority
  - LTS Priority Group becomes main decider in which LTS to use.
LTS Priority Group Example

- A logical fact table has two LTS mapped; one detail-level and one aggregate
- By default, the aggregate source is used for aggregated queries (due to dimension logical level mapping)
- This behaviour will be over-ridden though when LTS Priority Group is amended
  ‣ Setting it to 1 makes it lower priority than the detail-level LTS

```sql
select sum(T879.FCAST_SAL_AMT) as c1,
       T873.PROD_CAT_DESC as c2
from
    GCBC_AGGS.A_PROD_CATEGORIES T873
    GCBC_AGGS.A_SALES_AGG T879
where  ( T873.PROD_CAT_DESC = T879.PROD_CAT_DESC )
group by T873.PROD_CAT_DESC

select sum(T117.FCAST_SAL_AMT) as c1,
       T107.PROD_CAT_DESC as c2
from
    GCBC_SALES.PRODUCTS T107  ,
    GCBC_SALES.SALES T117
where  ( T107.PROD_ID = T117.PROD_ID )
group by T107.PROD_CAT_DESC
```
Repository Passwords, and Repository Identity Management

- Repository files now have a password
  - Secures RPD file when there is no access to WebLogic Server
- RPD files are now encrypted, and compressed
- RPD Password is all that is required to edit RPD offline; **BIAdministrator** application role required in addition to edit online
Deployment of RPD Files Through Enterprise Manager

- Repository files are now deployed using Fusion Middleware Control
- Select Deployment > Repository > Lock and Edit Configuration
- Select RPD file and enter repository password
- Press Apply > Activate Changes > Restart to Apply Changes
Summary

• The goal of the semantic model is to simplify reporting data using a conformed logical dimensional model
• Federation capabilities allow us to model across multiple data sources
• Think in terms of dimensional modeling, particularly for the business model
• New features in OBIEE 11gR1 allow us to extend our modeling capability
• More complex hierarchies can be defined
• Lookup tables, and ID/Descriptive double columns can be defined
• Logical Table Source ordering is now more explicit
• RPD management is now handled through Fusion Middleware Control
• RPD administration security is externalized, and RPDs are now encrypted and secured
OBIEE 11g Data Modeling Best Practices

Mark Rittman, Director, Rittman Mead
UKOUG Technical & EBS Conference, Birmingham November 2010