ORACLE BI, ORACLE OLAP, ESSBASE
THE BENEFITS AND COST OF OPENNESS

Dan Vlamis, Vlamis Software Solutions, Inc.
dvlamis@vlamis.com

PREFACE

What a difference a year makes! For the Collaborate 2007 conference I wrote about how Oracle's direction with multi-dimensional tools was unclear, especially with the recent announcement of the acquisition of Hyperion, including Essbase. Now Oracle has made it quite clear that Oracle OLAP, Hyperion Essbase, and Oracle Business Intelligence Enterprise Edition (OBIEE) will be continued indefinitely.

In the past year, Oracle has clearly chosen OBIEE as the platform for BI applications. It is especially important now to show how OBIEE can be used as an interface for Oracle OLAP cubes.

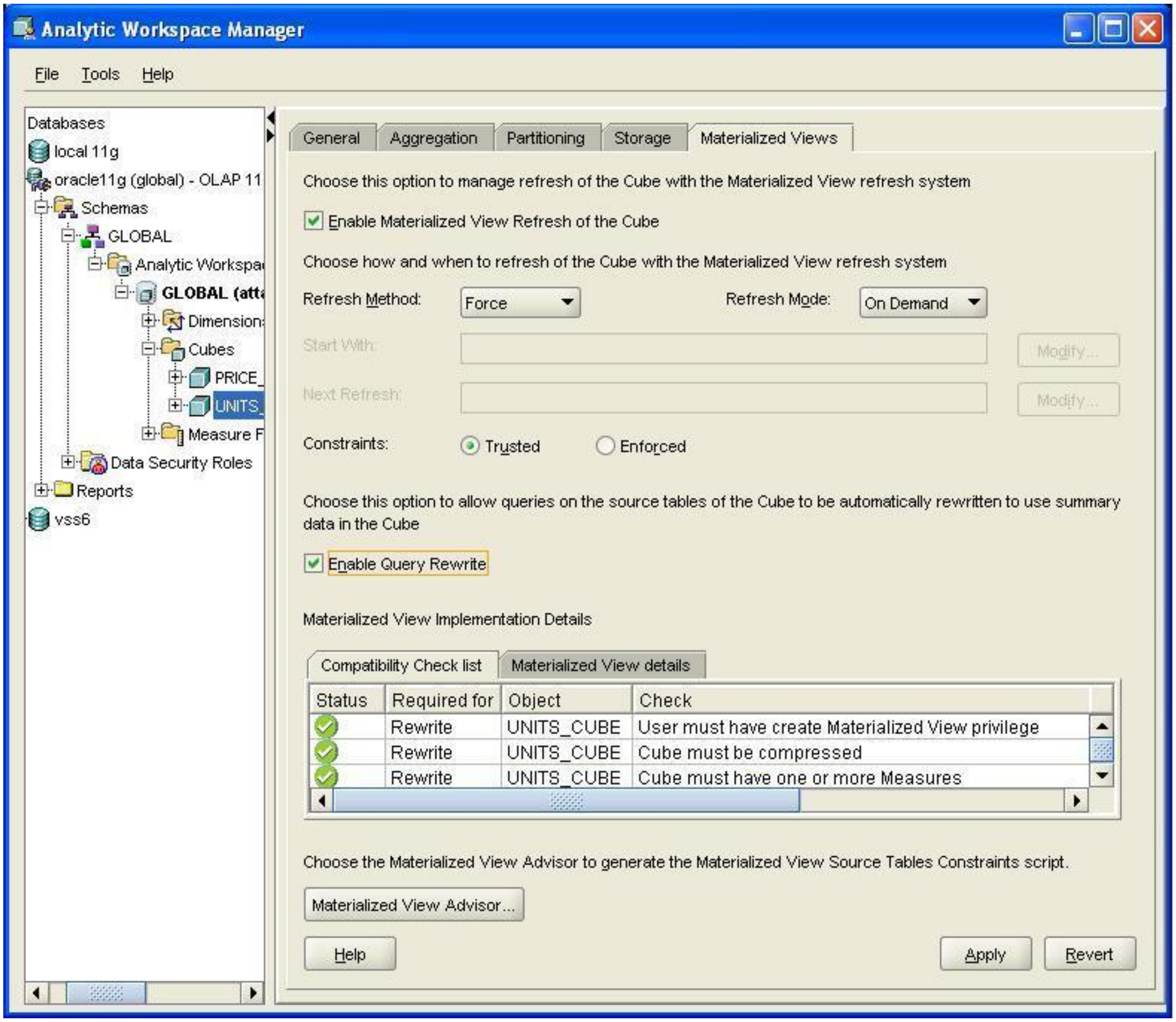
INTRODUCTION

A prior technical paper, Integration of Oracle BI EE and Oracle Analytic Workspaces detailed the process of using OBIEE against Oracle OLAP 10g cubes. This paper will rather deal with the process of using OBIEE against Oracle OLAP 11g cubes.

There are two basic ways to use OBIEE against Oracle OLAP 11g – one is to take advantage of the materialized view query redirection; the other is to use the Oracle OLAP views.

THE EASY WAY

Oracle OLAP 11g can be used as a summary management tool, relying on materialized views to re-direct queries that would normally run against a standard relational fact table to a cube. If this is the mode you want to deploy, your task is extremely simple. Simply use Oracle BI Administrator to create an RPD file that maps to a star schema with a central fact table and dimension tables. Then create a cube from your star schema using Analytic Workspace Manager and check the Enable Query Rewrite checkbox for the cube as shown in Figure 1. You do not even need to tell OBIEE that Oracle OLAP exists! OBIEE will create its queries against the fact table and the Oracle optimizer will re-direct the queries to take advantage of the cube.



Choose this option to manage refresh of the Cube with the Materialized View refresh system

☑ Enable Materialized View Refresh of the Cube

Choose how and when to refresh of the Cube with the Materialized View refresh system

Refresh Method: Force

Refresh Mode: On Demand

Start With:

Next Refresh:

Constraints: ● Trusted ○ Enforced

Choose this option to allow queries on the source tables of the Cube to be automatically rewritten to use summary data in the Cube

☑ Enable Query Rewrite

Materialized View Implementation Details

Status	Required for	Object	Check
✓	Rewrite	UNITS_CUBE	User must have create Materialized View privilege
✓	Rewrite	UNITS_CUBE	Cube must be compressed
✓	Rewrite	UNITS_CUBE	Cube must have one or more Measures

Choose the Materialized View Advisor to generate the Materialized View Source Tables Constraints script.

Figure 1

The (Harder) More Efficient Way

OBIEE relies on its data being accessed relationally. In order for it to access data, it must issue SQL to get at the data. Furthermore, the data must be presented to it as a series of tables or views. With Oracle OLAP 10g, we used the OLAP_TABLE function to translate a multi-dimensional view of the data into a table object. With Oracle OLAP 11g, this is accomplished via views that use the CUBE_TABLE function. Furthermore, with OLAP 11g, these views are automatically created whenever an analytic workspace is created. This greatly simplifies the process of accessing Oracle OLAP cubes via OBIEE, since the views are automatically created.

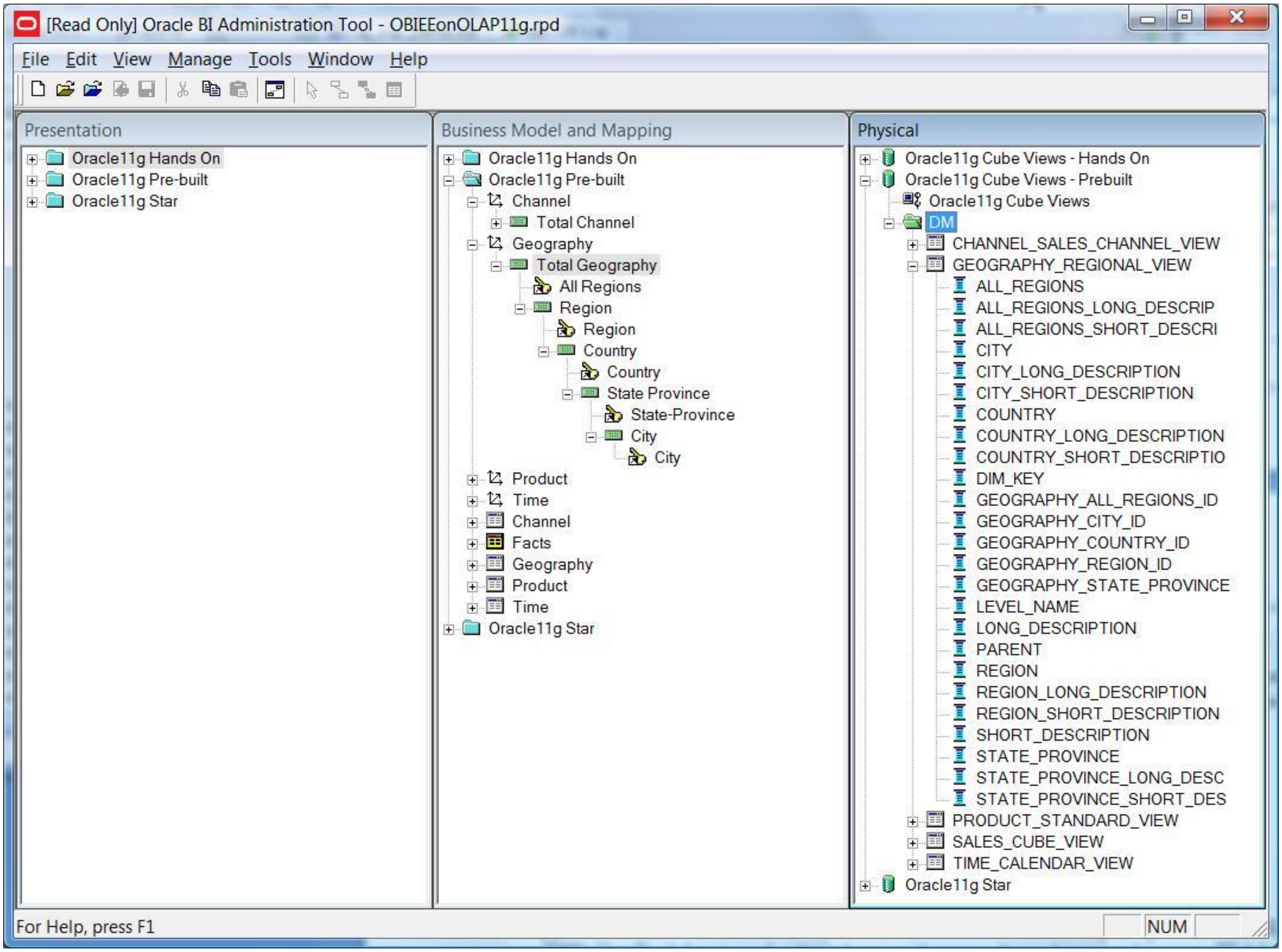
Overview of Process

OBIEE stores the information needed to access data in an RPD file. You use Oracle BI Administrator to create and edit the RPD file. Since the necessary views are already created by Oracle OLAP 11g, all you really need to do is use Oracle BI Administrator to point to the Oracle OLAP views and perform a bit of "trickery" to ensure that only one level of the cube is returned in a query.

OBIEE needs a dimension table or view for each of your dimensions. These are automatically created by Oracle OLAP 11g. When you create a dimension in Analytic Workspace Manager 11g, the appropriate view is automatically created. Here are the views created:

- One view for each dimension (e.g. GEOGRAPHY_VIEW)
- One view for each hierarchy (e.g. GEOGRAPHY_REGIONAL_VIEW)
- One view for each cube (e.g. SALES_CUBE_VIEW)

The easiest approach is to use the hierarchy views as your dimension views. Open up Oracle the BI Administration Tool and create a new RPD file. Find the views for the schema with your cube and import them into BI Administration Tool. Just as you would with a relational star schema, create the joins in the physical pane and the logical (Business Model and Mapping) pane. Now, make a dimension in the logical layer for each hierarchy view. You will end up creating the dimensional and level structure in OBIEE, just as you did in AWM. This is the step that OBIEE 11g promises to automate. The resulting screen should look something like the Business Model and Mapping shown in Figure 2.


[Read Only] Oracle BI Administration Tool - OBIEEonOLAP11g.rpd
File
Edit
View
Manage
Tools
Window
Help
For Help, press F1
NUM

Presentation

- Oracle11g Hands On
- Oracle11g Pre-built
- Oracle11g Star

Business Model and Mapping

- Oracle11g Hands On
- Oracle11g Pre-built
  - Channel
    - Total Channel
  - Geography
    - Total Geography
      - All Regions
      - Region
        - Region
        - Country
          - Country
          - State Province
            - State-Province
            - City
              - City
  - Product
  - Time
  - Channel
  - Facts
  - Geography
  - Product
  - Time
- Oracle11g Star

Physical

- Oracle11g Cube Views - Hands On
- Oracle11g Cube Views - Prebuilt
  - Oracle11g Cube Views
  - DM
    - CHANNEL_SALES_CHANNEL_VIEW
    - GEOGRAPHY_REGIONAL_VIEW
      - ALL_REGIONS
      - ALL_REGIONS_LONG_DESCRIP
      - ALL_REGIONS_SHORT_DESCRI
      - CITY
      - CITY_LONG_DESCRIPTION
      - CITY_SHORT_DESCRIPTION
      - COUNTRY
      - COUNTRY_LONG_DESCRIPTION
      - COUNTRY_SHORT_DESCRIPTIO
      - DIM_KEY
      - GEOGRAPHY_ALL_REGIONS_ID
      - GEOGRAPHY_CITY_ID
      - GEOGRAPHY_COUNTRY_ID
      - GEOGRAPHY_REGION_ID
      - GEOGRAPHY_STATE_PROVINCE
      - LEVEL_NAME
      - LONG_DESCRIPTION
      - PARENT
      - REGION
      - REGION_LONG_DESCRIPTION
      - REGION_SHORT_DESCRIPTION
      - SHORT_DESCRIPTION
      - STATE_PROVINCE
      - STATE_PROVINCE_LONG_DESC
      - STATE_PROVINCE_SHORT_DES
    - PRODUCT_STANDARD_VIEW
    - SALES_CUBE_VIEW
    - TIME_CALENDAR_VIEW
- Oracle11g Star

Figure 2

After you specify the appropriate keys for your CUBE_VIEW, you'll end up with something like Figure 3.

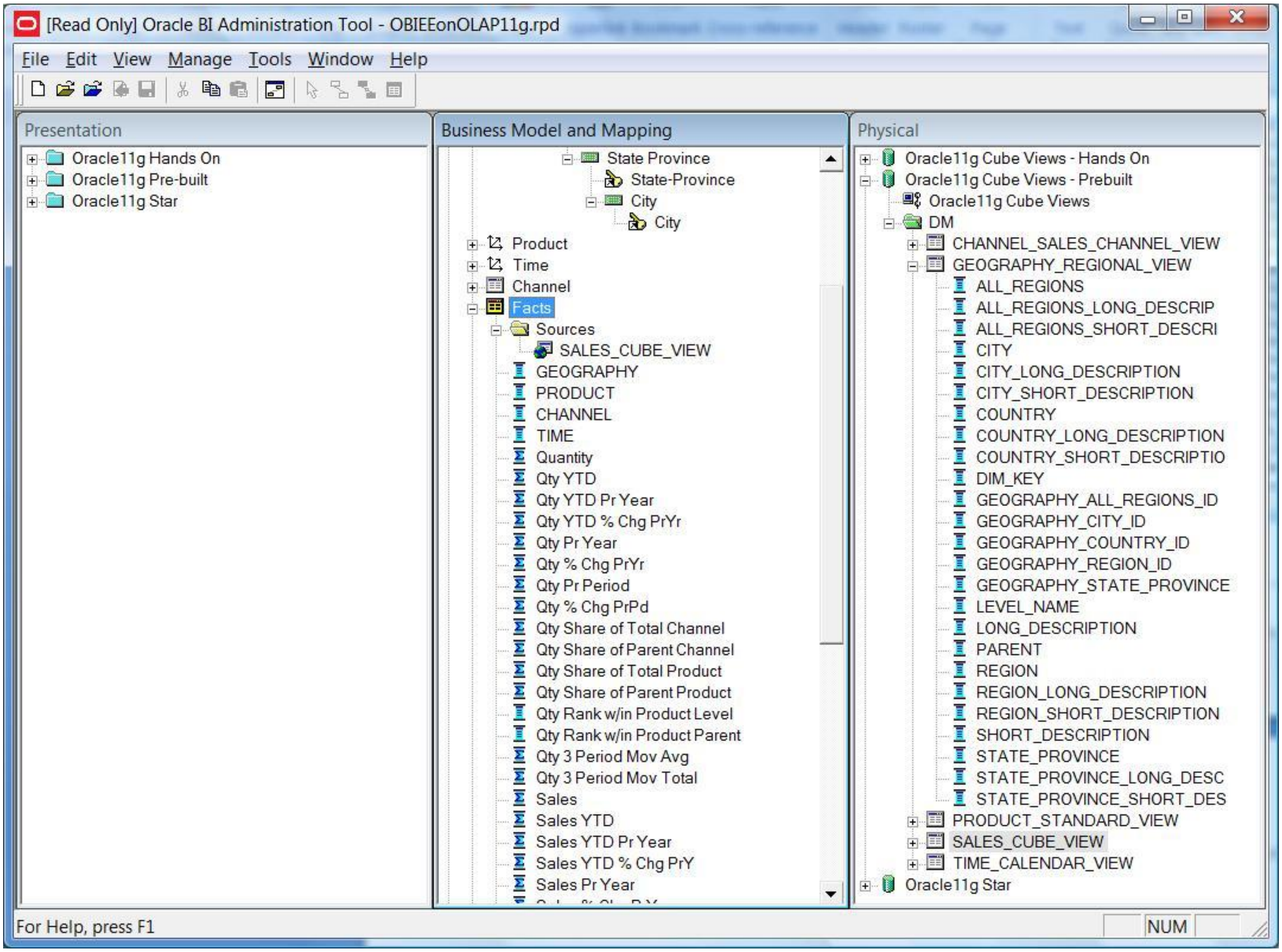
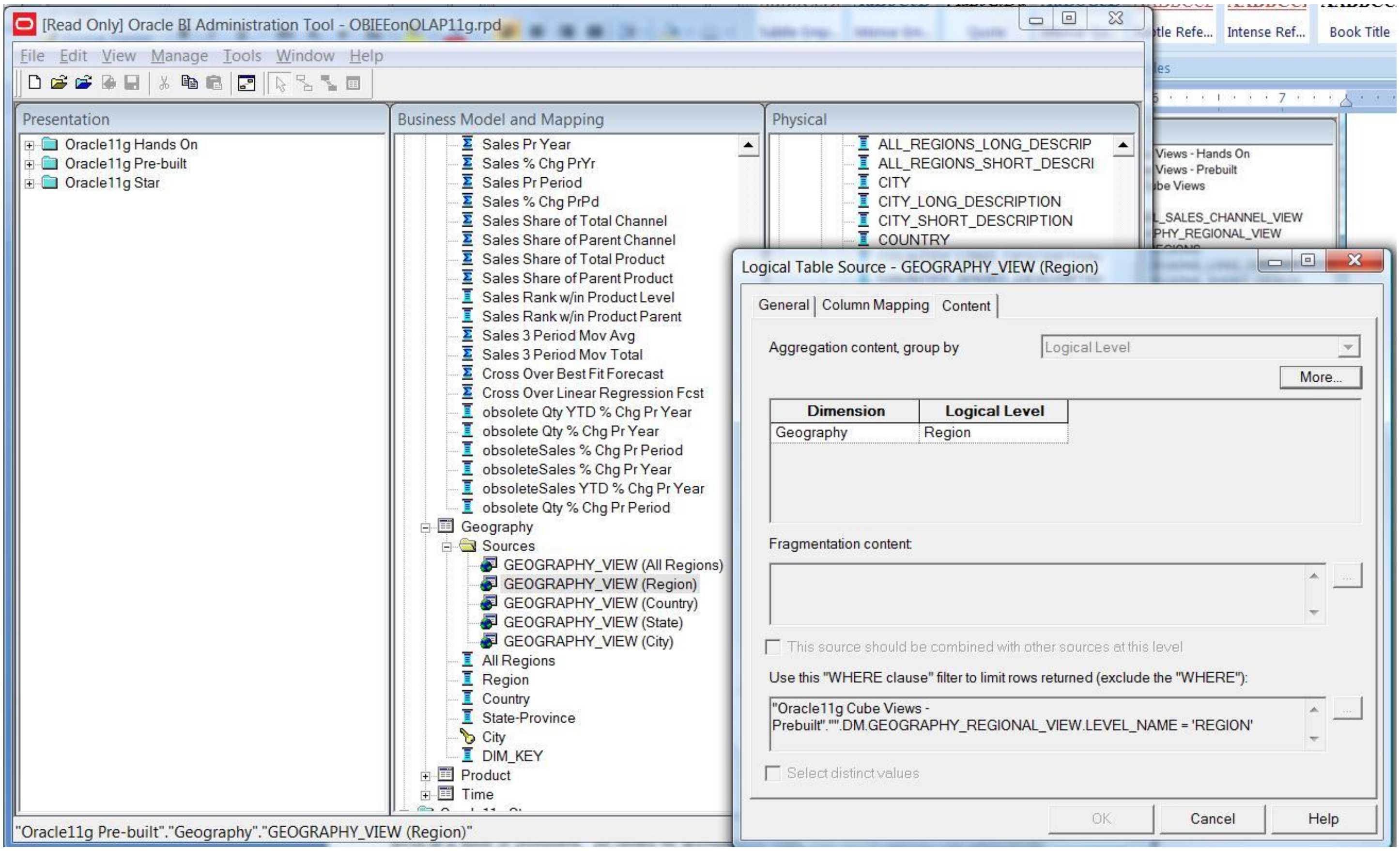


Figure 3

The trickiest part of this is that you must ensure that each level of a dimension only returns that level. This is the magic that makes it so that only one level at a time is returned. In order to accomplish this, you must specify a single logical level for each level of a dimension and specify a WHERE clause for each level in a dimension as shown in Figure 4. This where condition will ensure that when selecting REGIONs from the GEOGRAPHY_REGIONAL_VIEW, that we only return those rows where the LEVEL_NAME column contains 'REGION'.


[Read Only] Oracle BI Administration Tool - OBIEEonOLAP11g.rpd
File Edit View Manage Tools Window Help
Presentation
Oracle11g Hands On
Oracle11g Pre-built
Oracle11g Star
Business Model and Mapping
Sales Pr Year
Sales % Chg PrYr
Sales Pr Period
Sales % Chg PrPd
Sales Share of Total Channel
Sales Share of Parent Channel
Sales Share of Total Product
Sales Share of Parent Product
Sales Rank w/in Product Level
Sales Rank w/in Product Parent
Sales 3 Period Mov Avg
Sales 3 Period Mov Total
Cross Over Best Fit Forecast
Cross Over Linear Regression Fcst
obsolete Qty YTD % Chg Pr Year
obsolete Qty % Chg Pr Year
obsoleteSales % Chg Pr Period
obsoleteSales % Chg Pr Year
obsoleteSales YTD % Chg Pr Year
obsolete Qty % Chg Pr Period
Geography
Sources
GEOGRAPHY_VIEW (All Regions)
GEOGRAPHY_VIEW (Region)
GEOGRAPHY_VIEW (Country)
GEOGRAPHY_VIEW (State)
GEOGRAPHY_VIEW (City)
All Regions
Region
Country
State-Province
City
DIM_KEY
Product
Time
Physical
ALL_REGIONS_LONG_DESCRIP
ALL_REGIONS_SHORT_DESCRI
CITY
CITY_LONG_DESCRIPTION
CITY_SHORT_DESCRIPTION
COUNTRY
Logical Table Source - GEOGRAPHY_VIEW (Region)
General
Column Mapping
Content
More...
OK
Cancel
Help

Aggregation content, group by: Logical Level

Dimension	Logical Level
Geography	Region

Fragmentation content:

□ This source should be combined with other sources at this level

Use this "WHERE clause" filter to limit rows returned (exclude the "WHERE"):

```
"Oracle11g Cube Views - Prebuilt".""DM.GEOGRAPHY_REGIONAL_VIEW.LEVEL_NAME = 'REGION'
```

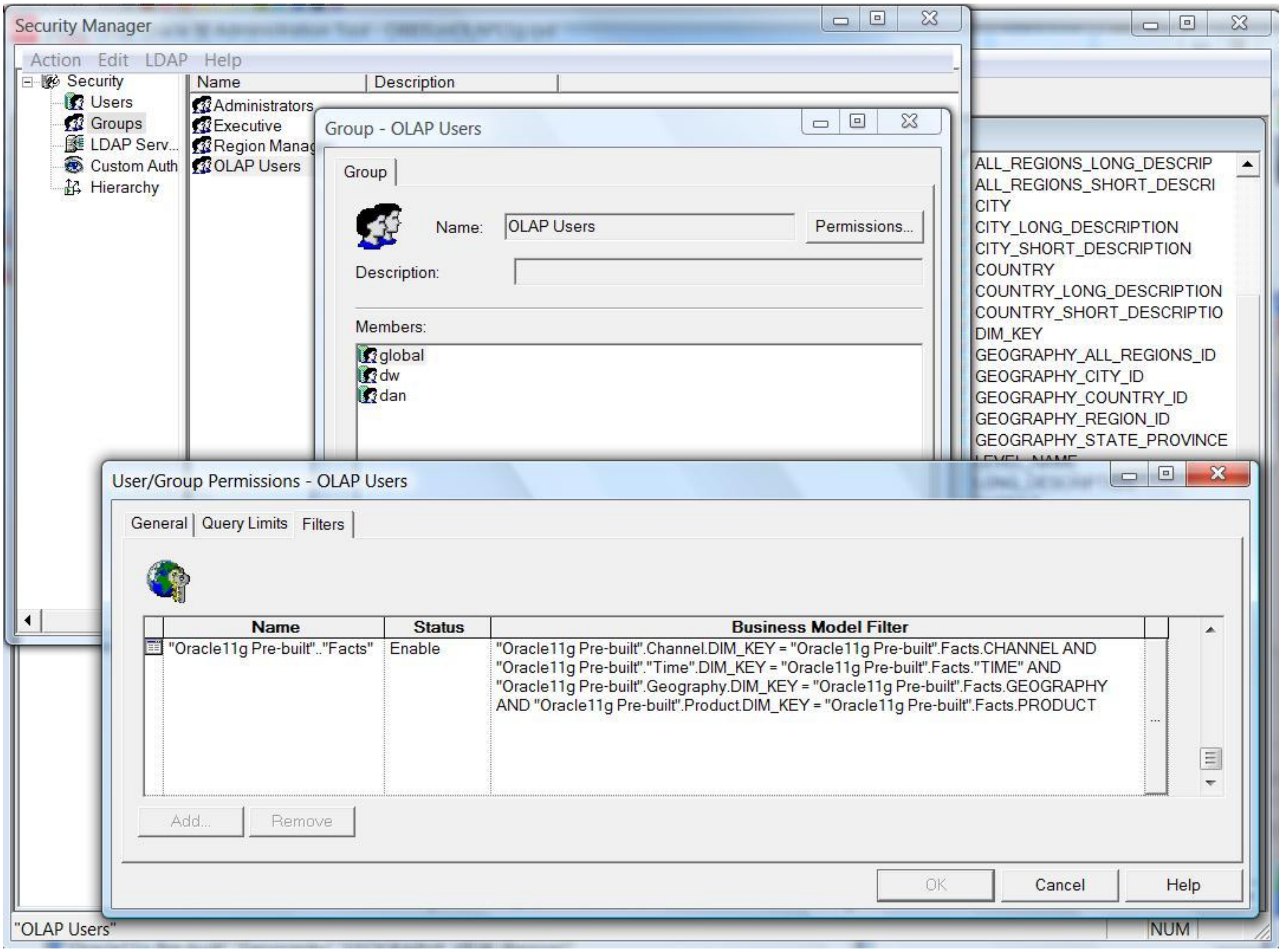
□ Select distinct values

"Oracle11g Pre-built"."Geography"."GEOGRAPHY_VIEW (Region)"

Figure 4

Finally, you must join the views together. This can be accomplished via the security manager in the Oracle BI Administration tool as shown in Figure 5.

My goal here is to outline the general steps needed to have OBIEE work with Oracle OLAP views to demonstrate what needs to be accomplished, not to provide a step-by-step "recipe book" for making this work. There are additional steps here that are beyond the scope of this paper.



Name: OLAP Users

Description:

Members:

global
dw
dan

Name	Status	Business Model Filter
"Oracle11g Pre-built".."Facts"	Enable	"Oracle11g Pre-built".Channel.DIM_KEY = "Oracle11g Pre-built".Facts.CHANNEL AND "Oracle11g Pre-built"."Time".DIM_KEY = "Oracle11g Pre-built".Facts."TIME" AND "Oracle11g Pre-built".Geography.DIM_KEY = "Oracle11g Pre-built".Facts.GEOGRAPHY AND "Oracle11g Pre-built".Product.DIM_KEY = "Oracle11g Pre-built".Facts.PRODUCT

Figure 5

Of course, once OBI EE 11g ships this becomes a moot point, as OBI EE will know how to map to an OLAP cube automatically.

CREATING CALCULATED MEASURES

You can create calculated measures that are presented to the end application as fully-solved data. Once the rules for calculating these measures are defined, a user does not have to worry about how the data is calculated; the data is presented as fully-solved. Examples of calculated measures that you can create include moving averages, change from year ago, moving totals, share calculations and much more.

GETTING MORE INFORMATION

There is much more information on Oracle OLAP on Oracle's web site. You will find most of the information linked to from the page http://www.oracle.com/technology/products/bi/olap/olap.html. For information on various client frontends, see the Oracle's Business Intelligence home page at http://www.oracle.com/technology/products/bi/index.html.