Business Intelligence Solution Providers
Specialized in creating talent resource pool

"OBIEE 11g Working with PARTITIONS"

This is another document from OBIEE11g Beginner's Guide Series. This document briefs you the step by step approach to understand Partitioning for OBIEE to speed up query performance. Learn from expert

History:

Version	Description Change	Author	Publish Date
0.1	Initial Draft	Hitesh Mankar	15th Sep 2011
0.1	1st Review	Amit Sharma	15th Sep 2011

Partitions

A partition is a division of a logical database or its constituting elements into distinct independent parts. Database partitioning is mostly used to manage, increase performance and also speeds up performance of OBI server. Using partitioning we splits big tables into small tables and more manageable form to get high performance from database and also from OBIEE.

Partitioning enables user to store one logical object a table transparently in several independent physical segments. Partitioning can provide great performance improvements because of partition elimination (pruning) capabilities, but also because parallel execution plans can take advantage of partitioning. The partitioning feature Database enables to partition stored data segments such as tables and indexes for easier management and improved performance.

Partitioning for OBIEE:-

For a single logical table source in a Business Model data is often partitioned into multiple physical sources. And when a logical table source does not contain the entire set of data at a given level, need to specify the partition of the set that it contains.

If the metadata is built using multiple sources OBI Server handles all the navigation to appropriate source. So OBI decides on its own to access which source for faster and correct result to meet the rusers request.

Different types of partitioning used to increase performance of OBI server.

1. Fact Based Partitioning: - if partitioning is done like data related to fact is stored in different tables. For example Yearly Amount sales are stored in different table other then Fact_sales table.

Below is the look of detail and partitioned fact.

Detail FACT table

DAY_KEY	PRODUCT_KEY	STORE_KEY	AMOUNT_SALES	UNIT_SALES
366	11	10	2709	246
366	12	10	10198	246
366	16	10	3847	246
366	23	10	6800	246
366	50	10	11333	308
366	75	10	4165	185
366	89	10	4073	246

Partitioned table based on FACT

YEAR	AMOUNT_SALES
2001	114266330
2002	18615322

2. Level Based Partitioning: - If the same facts (measures) are stored in different or separate table's at different levels of aggregation. The data inside level based partitioned table is always calculated to a specific level of aggregation.

This technique allows to mix two facts table with two different grains (the level based partitioning).

Below is the look of Level Based Partitioned table.

Level based Partitioned Table

YEAR	REGION	AMOUNT_SALES
2001	Asia Pacific	24785446
2002	Asia Pacific	4770584

3. Value Based Partitioning: - According to the values of the data partitioning is possible, can partition data into separate tables. Depending of a value of column, we can split the query against a table or another.

Value based partitioning can create complexity within query processing. Creating partitions value based partitions number of tables will increase, but help to get faster result.

Below is the look of value based partitioned tables.

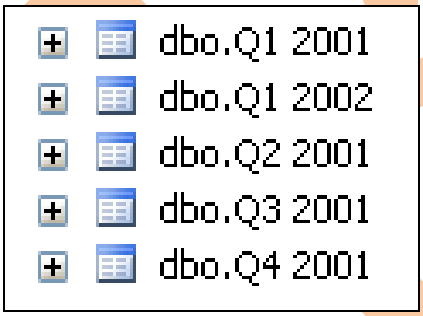
Value based Partitioned Table

YEAR	QUARTER	AMOUNT_SALES
2001	Q1	20729336

YEAR	QUARTER	AMOUNT_SALES
2001	Q2	20816579

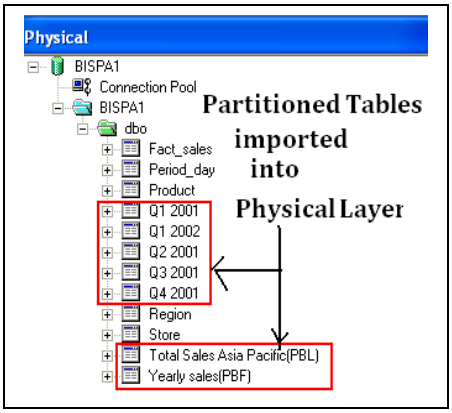
YEAR	Quarter	AMOUNT_SALES
2001	Q4	47173497

Multiple value based partitioned tables.

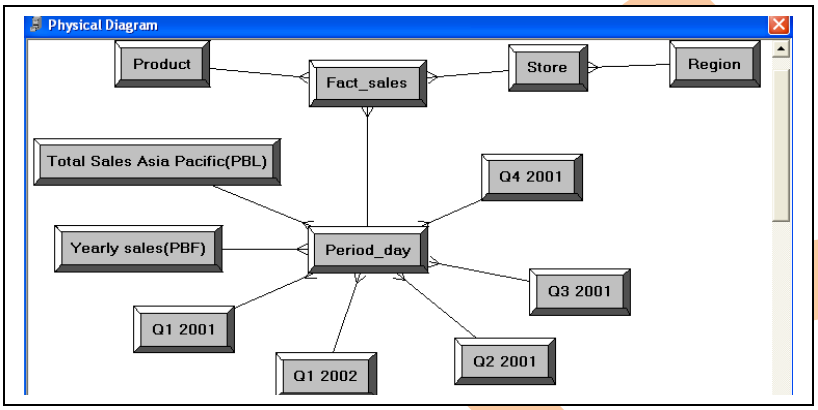


Follow below steps to create request for partitioned tables.

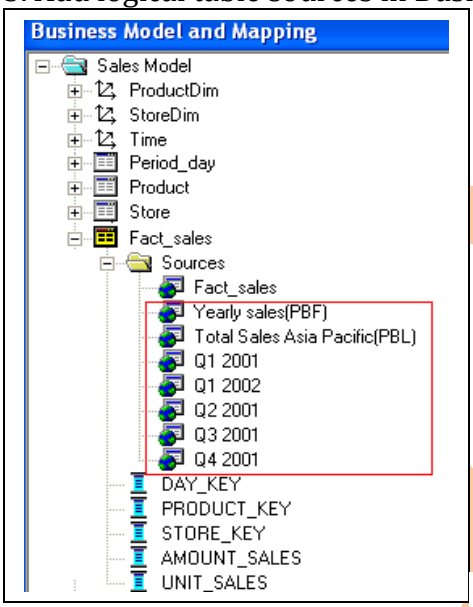
1. Import metadata into physical layer.

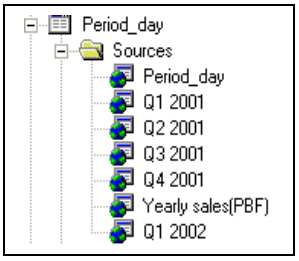


2. Create Physical Joins.

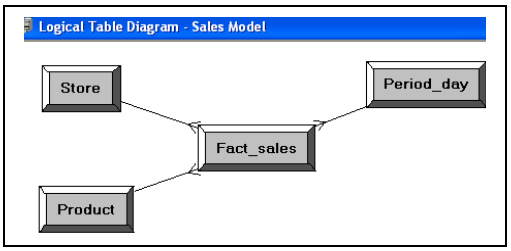


3. Add logical table sources in Business Model and Mapping Layer, as per requirement.





4. Check Business Model is proper or not.



5. Drag and drop business model into presentation layer→ Check global consistency and save the rpd.

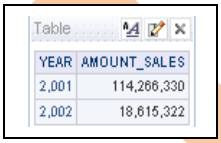
6. Configure rpd and start services to deploy into answers → Login into answers.

7. Create a request

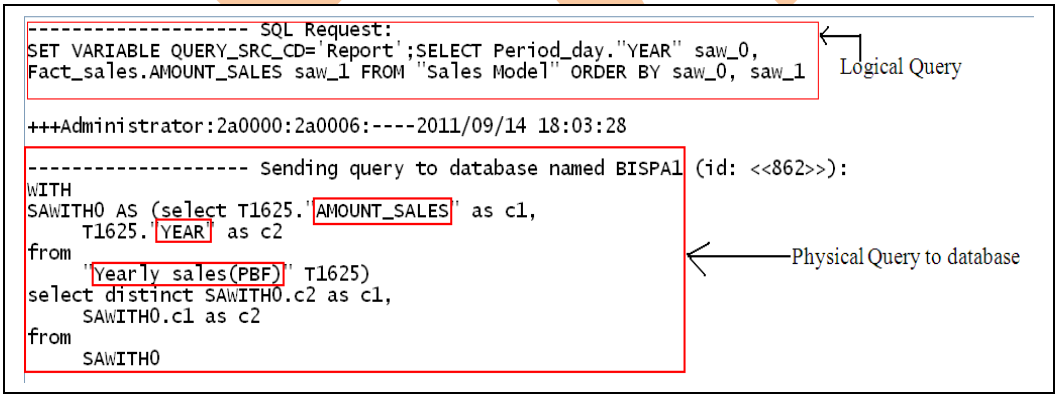
Instance 1).Select columns to make a request from criteria tab into Answers to get result from FACT based

partitioned tables .

8. Click on Result tab to check result, check is it desired result or not.



(Result is correct now, check the query log for query)



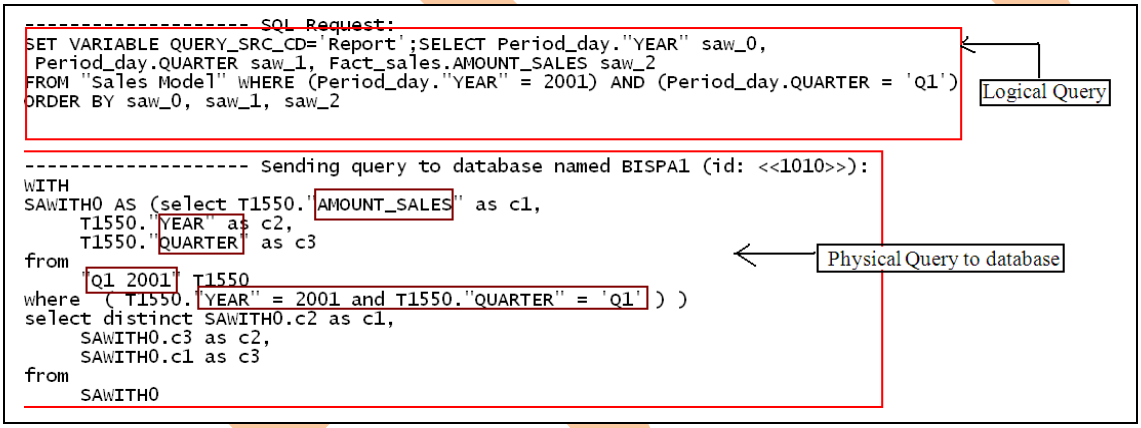
Instance 2) A filtered request for Value Based Partition.

Check Result is desired or not.

Table

YEAR	QUARTER	AMOUNT_SALES
2,001	Q1	20,729,336

Check Query Log.



```
-------------------- SQL Request:
SET VARIABLE QUERY_SRC_CD='Report';SELECT Period_day."YEAR" saw_0,
 Period_day.QUARTER saw_1, Fact_sales.AMOUNT_SALES saw_2
FROM "Sales Model" WHERE (Period_day."YEAR" = 2001) AND (Period_day.QUARTER = 'Q1')
ORDER BY saw_0, saw_1, saw_2

-------------------- Sending query to database named BISPA1 (id: <<1010>>):
WITH
SAWITH0 AS (select T1550."AMOUNT_SALES" as c1,
     T1550."YEAR" as c2,
     T1550."QUARTER" as c3
from
     "Q1 2001" T1550
where  ( T1550."YEAR" = 2001 and T1550."QUARTER" = 'Q1' ) )
select distinct SAWITH0.c2 as c1,
     SAWITH0.c3 as c2,
     SAWITH0.c1 as c3
from
     SAWITH0
```

Logical Query

Physical Query to database

Instance 3) A filtered request from get result from Level Based Partitioned tables.

Check Result is desired or not.

Table

YEAR	REGION	AMOUNT_SALES
2,001	Asia Pacific	24,785,446
2,002	Asia Pacific	4,770,584

Check Query Log.

```
-------------------- SQL Request:
SET VARIABLE QUERY_SRC_CD='Report';SELECT Period_day."YEAR" saw_0,
Store.REGION saw_1, Fact_sales.AMOUNT_SALES saw_2 FROM "Sales Model"
 WHERE Store.REGION = 'Asia Pacific' ORDER BY saw_0, saw_1, saw_2

+++Administrator:2a0000:2a0007:----2011/09/15 17:18:22

-------------------- Sending query to database named BISPA1 (id: <<777>>):
WITH
SAWITH0 AS (select T1621."AMOUNT_SALES" as c1,
     T1621."YEAR" as c2,
     T1621."REGION" as c3
from
     "Total Sales Asia Pacific(PBL)" T1621
where  ( T1621."REGION" = 'Asia Pacific' ) )
select distinct SAWITH0.c2 as c1,
     SAWITH0.c3 as c2,
     SAWITH0.c1 as c3
from
     SAWITH0
```