|  |
| --- |
| Practical 2 POSTLABOLAP operations |

**1)In data warehouse technology, a multiple dimensional view can be implemented by a relational database technique (*ROLAP*), by a multidimensional database technique (*MOLAP*), or by a hybrid database technique (*HOLAP*).**

**(a) Briefly describe each implementation technique.**

**(b) For each explain how each of the following functions implemented:**

**i. The generation of a data warehouse (including aggregation)**

**ii. Roll-up**

**iii. Drill-down**

**iv. Incremental updating**

**(c) Which implementation techniques do you prefer, and why?**

**Ans:**

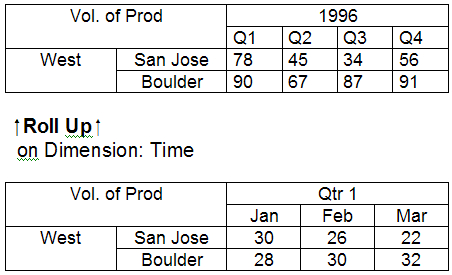
**a)There are 3 main types of OLAP servers are as following:**

1. **Relational OLAP (ROLAP) – Star Schema based –  
   The ROLAP is based on the premise that data need not to be stored multidimensionally in order to viewed multidimensionally, and that it is possible to exploit the well-proven relational database technology to handle multidimensionality of data.In ROLAP data is stored in a relational database.In essence, each action of slicing and dicing is equivalent to adding a “WHERE” clause in SQL statement. ROLAP can handle large amounts of data. ROLAP can leverage functionalities inherent in the relational database.**

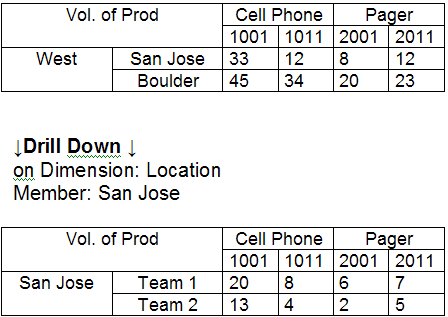
**Explain how each of the following functions may be implemented:**

* 1. **The generation of a data warehouse (including aggregation).**

**Initial aggregation may be accomplished in SQL via group-bys. The compute cube operator computes aggregates over all subsets of the dimensions in the specified operation; this leads to the generation of a single cube. ROLAP relies on tuples and relational tables as its basic data structures. The base fact table (a relational table) stores data at the abstraction level indicated by join keys in the schema for the given data cube. Aggregated data can also be stored in fact tables (summary fact tables). ROLAP uses value-based addressing, where dimension values are accessed via key-based addressing search strategies. To optimize ROLAP cube computation we may use the following techniques:**

* **sorting, hashing, grouping operations**
* **grouping is performed on sub-aggregates**
* **aggregates derived from previously computed aggregates**
  1. **[](http://www.12345w.xyz/info-source-us/data_warehousing_mining/images/roll-up.png)Roll-up:**

**Aggregation on a data cube (aka dimension reduction). In ROLAP, this means that the relational tables are aggregated from more to less specific.**

**[](http://www.12345w.xyz/info-source-us/data_warehousing_mining/images/drill-down.png)**

* 1. **Drill-down:**

**The opposite of Roll-up. We introduce additional dimensions into the relation tables and, hence, cubes.**

* 1. **Incremental updating:**

**Data warehouse implementation can be broken down into segments or increments. An increment is a defined data warehouse implementation project that has a specified beginning and end. An increment may also be referred to as a departmental data warehouse within the context of an enterprise.**

1. **Multidimensional OLAP (MOLAP) – Cube based –  
   MOLAP stores data on disks in specialised multidimensional array structure. OLAP is performed on it relying on the random access capability of the arrays. Arrays element are determined by dimension instances, and the fact data or measured value associated with each cell is usually stored in the corresponding array element. In MOLAP, the multidimensional array is usually stored in a linear allocation according to nested traversal of the axes in some predetermine order. But unlike ROLAP, where only records with non-zero facts are stored, all array elements are defined in MOLAP and as a result, the arrays generally tend to sparse, with empty elements occupying a greater part of it.Since both storage and retrieval costs are important while assessing online performance efficiency, MOLAP systems typically include provision such as advanced indexing and hashing to locate data while performing queries for handling sparse arrays. MOLAP cubes are fast data retrieval, optimal for slicing and dicing and it can perform complex calculation. All calculation are pre-generated when the cube is created.**

***Explain how each of the following functions may be implemented:***

* 1. **The generation of a data warehouse (including aggregation).**

**MOLAP uses array structures to store data for OLAP. Initial aggregation may be accomplished in SQL via group-bys. The compute cube operator computes aggregates over all subsets of the dimensions in the specified operation; this leads to the generation of a single cube. MOLAP follows very different cube computation scheme than ROLAP. It uses direct array addressing, where dimension values are accessed via the position or index of the corresponding array. The approach to generate an array-based cube is as follows:**

* **Partition array into chunks. A chunk is a subcube that is small enough to fit into the memory for cube computation.**
* **Compute aggregates by visiting (access the values at) cube cells.**
  1. **Roll-up:**

**The Roll-up method for MOLAPs would be somewhat similar to the process described above for ROLAP. Except, now, we are rolling up chunks that make up the subcubes…and rolling up the subcubes that make up the array.**

* 1. **Drill-down: The opposite of roll-up.**

**We introduce additional dimensions into the subcubes or array.**

* 1. **Incremental updating.**

**Please see “Incremental Updating” for ROLAPs. The technique for MOLAP updates would be more sophisticated due to the additional complexity of arrays and subcubes. Selecting the right tools would seem to be the key. An example would be Microsoft Data Warehousing Framework and OLAP Manager which is included with SQL Server OLAP Services**

1. **Hybrid OLAP (HOLAP) –  
   HOLAP is a combination of ROLAP and MOLAP. HOLAP servers allows storing the large data volumes of detail data.On the one hand, HOLAP leverages the greater scalability of ROLAP. On the other hand, HOLAP leverages the cube technology for faster performance and for summary-type inforamtion. Cubes are smaller than MOLAP since detail data is kept in the relational database. The database are used to stores data in the most functional way possible.**

***Explain how each of the following functions may be implemented:***

* 1. **The generation of a data warehouse (including aggregation).**

**The generation would consist of a combined approach of both MOLAP and ROLAP. A HOLAP would be generated to store large volumes of detail in a relational database (see i. for ROLAP generation methods) while a MOLAP would be used to store aggregations separately (see i. for MOLAP generation methods)**

* 1. **Roll-up.**

**A combination of ROLAP and MOLAP roll up methods.**

* 1. **Drill-down.**

**A combination of ROLAP and MOLAP roll up methods**

* 1. **Incremental updating.**

**A combination of ROLAP and MOLAP roll up methods.**

***c) Which implementation do you prefer?***

**I would have to say that the Hybrid approach seems to be the best solution for most applications. It would appear to be backward compatible with older(?) ROLAPs and retains the scalability of this implementation. At the same time, it incorporates the more sophisticated features and faster computation of MOLAP.**

**2]** **Write syntax of rollup and cube of postgressql and write version of postgresql for these commands**

**Ans.** **Version Used PostgreSQL 9.5**

**ROLLUP**

**The PostgreSQL ROLLUP is a subclause of the**[**GROUP BY**](https://www.postgresqltutorial.com/postgresql-group-by/)**clause that offers a shorthand for defining multiple**[**grouping sets**](https://www.postgresqltutorial.com/postgresql-grouping-sets/)**. A grouping set is a set of columns to which you want to group.**

**The ROLLUP assumes a hierarchy among the input columns and generates all grouping sets that make sense considering the hierarchy. This is the reason why ROLLUP is often used to generate the subtotals and the grand total for reports.**

**For example, the CUBE (c1,c2,c3) makes all eight possible grouping sets:**

|  |  |
| --- | --- |
| **1**  **2**  **3**  **4**  **5**  **6**  **7**  **8** | **(c1, c2, c3)**  **(c1, c2)**  **(c2, c3)**  **(c1,c3)**  **(c1)**  **(c2)**  **(c3)**  **()** |

**However, the ROLLUP(c1,c2,c3) generates only four grouping sets, assuming the hierarchy c1 > c2 > c3 as follows:**

|  |  |
| --- | --- |
| **1**  **2**  **3**  **4** | **(c1, c2, c3)**  **(c1, c2)**  **(c1)**  **()** |

**The following illustrates the syntax of the PostgreSQL ROLLUP:**

|  |  |
| --- | --- |
| **1**  **2**  **3**  **4**  **5**  **6**  **7**  **8**  **9** | **SELECT**  **c1,**  **c2,**  **c3,**  **aggregate(c4)**  **FROM**  **table\_name**  **GROUP BY**  **ROLLUP (c1, c2, c3);** |
|  |  |

**CUBE**

**CUBE is a subclause of the**[**GROUP BY**](https://www.postgresqltutorial.com/postgresql-group-by/)**clause. The CUBE allows you to generate multiple grouping sets. A grouping set is a set of columns to which you want to group. For more information on the grouping sets, check it out the**[**GROUPING SETS**](https://www.postgresqltutorial.com/postgresql-grouping-sets/)**tutorial.**

**The following illustrates the syntax of the CUBE subclause:**

|  |  |
| --- | --- |
| **1**  **2**  **3**  **4**  **5**  **6**  **7**  **8**  **9** | **SELECT**  **c1,**  **c2,**  **c3,**  **aggregate (c4)**  **FROM**  **table\_name**  **GROUP BY**  **CUBE (c1, c2, c3);** |

**In this syntax:**

* **First, specify the CUBE subclause in the the**[**GROUP BY**](https://www.postgresqltutorial.com/postgresql-group-by/)**clause of the**[**SELECT**](https://www.postgresqltutorial.com/postgresql-select/)**statement.**
* **Second, in the select list, specify the columns (dimensions or dimension columns) which you want to analyze and**[**aggregation function**](https://www.postgresqltutorial.com/postgresql-aggregate-functions/)**expressions.**
* **In the GROUP BY clause, specify dimension columns within parentheses of the CUBE subclause.**

**The Equivalent CUBE Subclause**

|  |  |
| --- | --- |
| **1**  **3**  **4**  **5**  **6**  **7**  **8**  **9**  **10**  **11**  **12** | **CUBE(c1,c2,c3)**  **GROUPING SETS (**  **(c1,c2,c3),**  **(c1,c2),**  **(c1,c3),**  **(c2,c3),**  **(c1),**  **(c2),**  **(c3),**  **()**  **)** |

**The following shows the syntax:**

|  |  |
| --- | --- |
| **1**  **2**  **3**  **4**  **5**  **6**  **7**  **8**  **9**  **10** | **SELECT**  **c1,**  **c2,**  **c3,**  **aggregate (c4)**  **FROM**  **table\_name**  **GROUP BY**  **c1,**  **CUBE (c1, c2);** |