

Experiment No.8

Title: Execution of OLAP Operations

Batch:B1 Roll No.: 16010420133**Experiment No.:8****Aim:** Execution of OLAP operations

Resources needed: MySQL, Postgres

Theory

OLAP:

In computing, online analytical processing, or OLAP is an approach to answering multi-dimensional analytical (MDA) queries. OLAP is part of the broader category of business intelligence, which also encompasses relational database report writing and data mining. Typical applications of OLAP include business reporting for sales, marketing, management reporting, business process management (BPM), budgeting and forecasting, financial reporting and similar areas, with new applications coming up, such as agriculture. The term OLAP was created as a slight modification of the traditional database term OLTP (Online Transaction Processing).

OLAP tools enable users to analyze multidimensional data interactively from multiple perspectives. OLAP consists of three basic analytical operations: consolidation (roll-up), drill-down, and slicing and dicing. Consolidation involves the aggregation of data that can be accumulated and computed in one or more dimensions. For example, all sales offices are rolled up to the sales department or sales division to anticipate sales trends. By contrast, the drill-down is a technique that allows users to navigate through the details. For instance, users can view the sales by individual products that make up a region's sales. Slicing and dicing is a feature whereby users can take out (slicing) a specific set of data of the OLAP cube and view (dicing) the slices from different viewpoints.

OLAP queries can be implemented by using analytical SQL functions. Oracle has extensions to ANSI SQL to allow to quickly computing aggregations and rollups. These new statements include:

- rollup
- cube
- grouping

These simple SQL operators allow creating easy aggregations directly inside the SQL.

Creating tabular aggregates with ROLLUP:

ROLLUP enables an SQL statement to calculate multiple levels of subtotals across a specified group of dimensions. It also calculates a grand total. ROLLUP is a simple extension to the GROUP BY clause, so its syntax is extremely easy to use. Create cross-tabular reports with CUBE:

In multidimensional jargon, a “cube” is a cross-tabulated summary of detail rows. CUBE enables a SELECT statement to calculate subtotals for all possible combinations of a group of dimensions. It also calculates a grand total.

This is the set of information typically needed for all cross-tabular reports, so CUBE can calculate a cross-tabular report with a single select statement

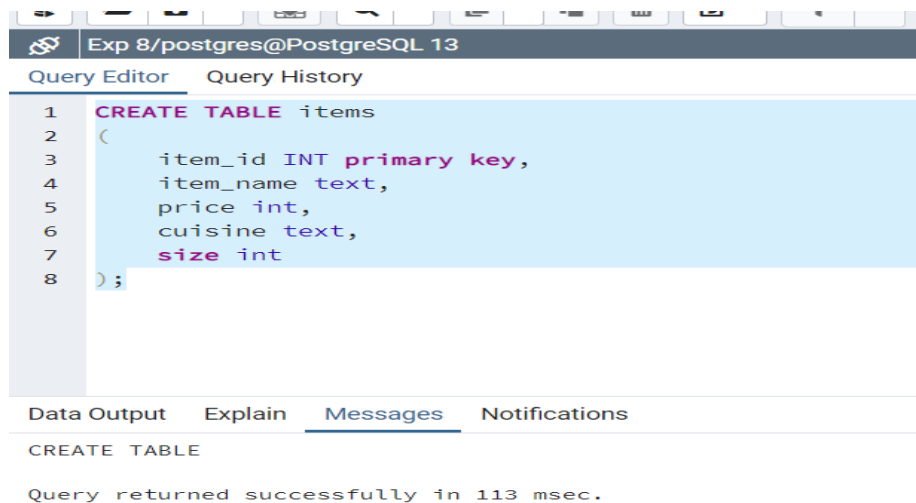
Activities:

1. Create a dataset in PostgreSQL and MySQL
 2. Apply rollup and cube operations to the same
-

Create database in Postgres:

- *First, we create our 'items' table in the database:*

```
CREATE TABLE items
(
    item_id INT primary key,
    item_name text,
    price int,
    cuisine text,
    dish_size text
);
```



- *Now, we insert values into our 'items' table in the database:*

```
insert into items values(1,'Hakka Noodles', 250, 'Chinese', 'Medium');
insert into items values(2,'Fried Rice', 300, 'Chinese', 'Large');
insert into items values(3,'Shezwan Noodles', 270, 'Chinese', 'Medium');
insert into items values(4,'Pizza', 350, 'Italian', 'Small');
insert into items values(5,'Pasta', 250, 'Italian', 'Medium');
insert into items values(6,'Idli', 200, 'South Indian', 'Large');
insert into items values(7,'Sada Dosa', 100, 'South Indian', 'Medium');
insert into items values(8,'Cheese Dosa', 150, 'South Indian', 'Large');
```

```
SELECT * FROM items
```

Exp 8/postgres@PostgreSQL 13

Query Editor Query History

```

10 drop table items;
11
12 insert into items values(1,'Hakka Noodles', 250, 'Chinese', 'Medium');
13 insert into items values(2,'Fried Rice', 300, 'Chinese', 'Large');
14 insert into items values(3,'Shezwan Noodles', 270, 'Chinese', 'Medium');
15 insert into items values(4,'Pizza', 350, 'Italian', 'Small');
16 insert into items values(5,'Pasta', 250, 'Italian', 'Medium');
17 insert into items values(6,'Idli', 200, 'South Indian', 'Large');
18 insert into items values(7,'Sada Dosa', 100, 'South Indian', 'Medium');
19 insert into items values(8,'Cheese Dosa', 150, 'South Indian', 'Large');
20
21 SELECT * FROM items

```

Data Output Explain Messages Notifications

	item_id [PK] integer	item_name text	price integer	cuisine text	dish_size text
1	1	Hakka Noodles	250	Chinese	Medium
2	2	Fried Rice	300	Chinese	Large
3	3	Shezwan Noodl...	270	Chinese	Medium
4	4	Pizza	350	Italian	Small
5	5	Pasta	250	Italian	Medium
6	6	Idli	200	South Indian	Large
7	7	Sada Dosa	100	South Indian	Medium
8	8	Cheese Dosa	150	South Indian	Large

Execute OLAP queries on database table using rollup, cube and grouping sets and observe the results:

- Now, we use 'GROUP BY' on our 'Cuisine' column and display each value in the cuisine column along with the sum of the prices of each item of that particular value(i.e. cuisine):

```

SELECT cuisine, SUM(price) AS TotalPrice
FROM items
GROUP BY cuisine;

```

Exp 8/postgres@PostgreSQL 13

Query Editor Query History

```

13 insert into items values(4,'Pizza', 350, 'Italian', 'Small');
14 insert into items values(5,'Pasta', 250, 'Italian', 'Medium');
15 insert into items values(6,'Idli', 200, 'South Indian', 'Large');
16 insert into items values(7,'Sada Dosa', 100, 'South Indian', 'Medium');
17 insert into items values(8,'Cheese Dosa', 150, 'South Indian', 'Large');
18
19 SELECT * FROM items
20
21 SELECT cuisine, SUM(price) AS TotalPrice
22 FROM items
23 GROUP BY cuisine;

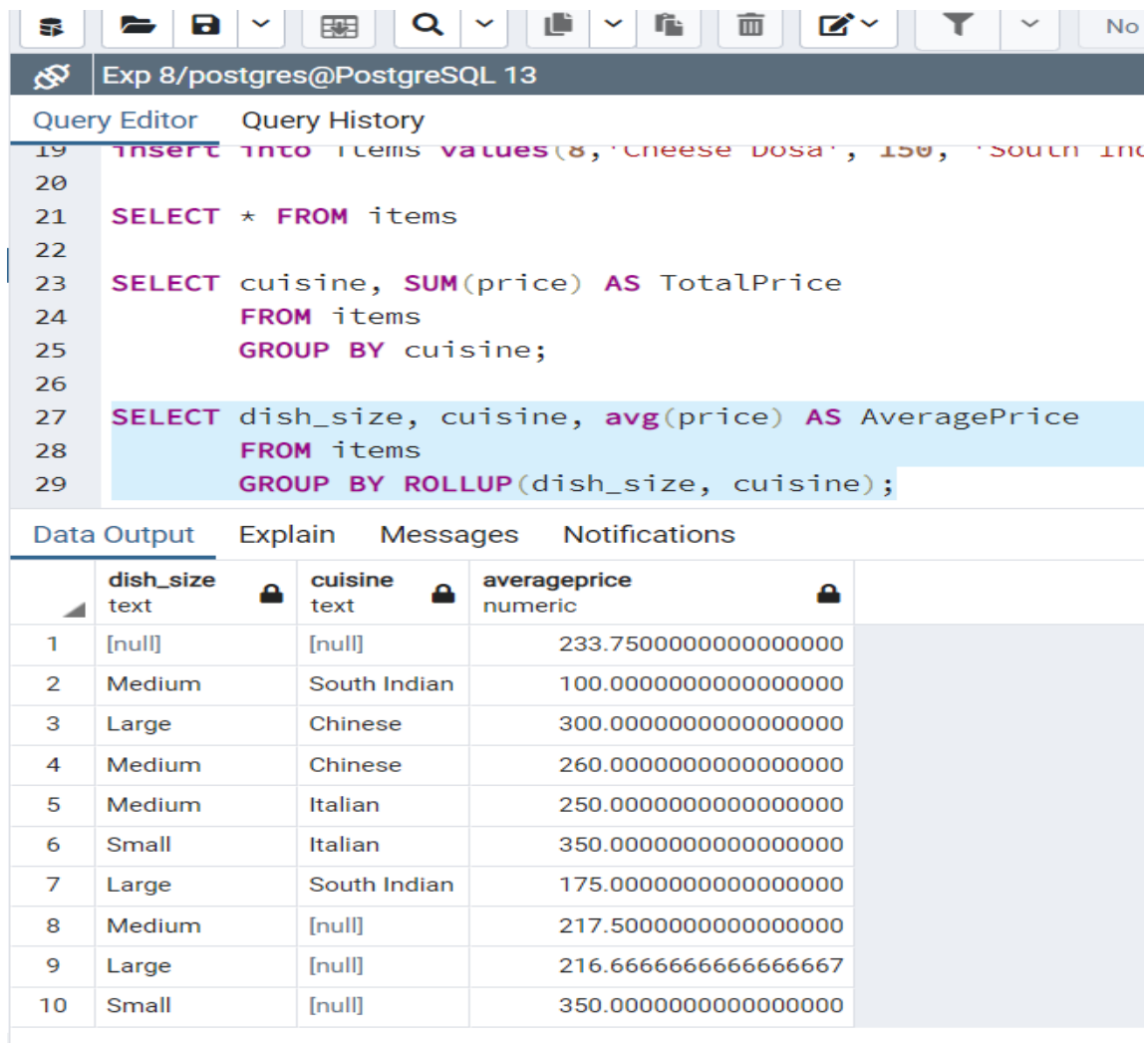
```

Data Output Explain Messages Notifications

	cuisine text	totalprice bigint
1	Chinese	820
2	South Indian	450
3	Italian	600

- Now, we use 'GROUP BY' along with 'ROLLUP'.

```
SELECT dish_size, cuisine, avg(price) AS AveragePrice
FROM items
GROUP BY ROLLUP(dish_size, cuisine);
```



Exp 8/postgres@PostgreSQL 13

Query Editor Query History

```
19 Insert Into Items values(8, 'Cheese Dosa', 150, 'South Ind
20
21 SELECT * FROM items
22
23 SELECT cuisine, SUM(price) AS TotalPrice
24 FROM items
25 GROUP BY cuisine;
26
27 SELECT dish_size, cuisine, avg(price) AS AveragePrice
28 FROM items
29 GROUP BY ROLLUP(dish_size, cuisine);
```

Data Output Explain Messages Notifications

	dish_size text	cuisine text	averageprice numeric
1	[null]	[null]	233.750000000000000000000000
2	Medium	South Indian	100.000000000000000000000000
3	Large	Chinese	300.000000000000000000000000
4	Medium	Chinese	260.000000000000000000000000
5	Medium	Italian	250.000000000000000000000000
6	Small	Italian	350.000000000000000000000000
7	Large	South Indian	175.000000000000000000000000
8	Medium	[null]	217.500000000000000000000000
9	Large	[null]	216.666666666666666666666667
10	Small	[null]	350.000000000000000000000000

Observation:

The logic of rollup is that it takes multiple arguments and applies group by while taking both into consideration. As shown above it group by of all the possible groups between dish_size and cuisine and then it appends a group by clause applied to the first argument given, in this case dish_size.

- Now, we use 'GROUP BY' along with 'ROLLUP' on our 'Cuisine' column and display each value in the cuisine column along with the sum of the prices of each item of that particular value(i.e. cuisine):

```
SELECT dish_size, cuisine, sum(price) as TotalPrice
FROM items
GROUP BY CUBE(dish_size, cuisine);
```

</

Observation:

The logic of cube is that it takes multiple arguments and applies group by while taking both into consideration. As shown above it group by of all the possible groups between dish_size and cuisine and then it appends a group by clause applied to each argument given like, in this case dish_size and cuisine individually.

- Now, we use 'GROUP BY' along with 'ROLLUP' on our 'Cuisine' column and display each value in the cuisine column along with the sum of the prices of each item of that particular value(i.e. cuisine):

```
SELECT dish_size, cuisine, sum(price) as TotalPrice
FROM items
GROUP BY GROUPING SETS(ROLLUP(dish_size, cuisine), dish_size);
```

Questions:

1. Elaborate on the operations applied and results generated to your dataset

Roll Up:

The roll-up operation (also known as drill-up or aggregation operation) performs aggregation on a data cube, by climbing down concept hierarchies, i.e., dimension reduction. Roll-up is like zooming-out on the data cubes.

```
SELECT city, customer_id, state, avg(sales)
FROM sales
GROUP BY rollup(customer_id, state, city)
ORDER BY city ASC;
```

In the above operation, it will group and then rollup, which means it will merge all the customers having the same customer id, state, and city. and after grouping up the average of the sales and the city, customer id, state will display on the output.

Group By:

```
SELECT city, customer_id, state, sum(sales)
FROM sales
GROUP BY customer_id, state, city
ORDER BY city ASC;
```

In the above operation it will group the data on the basis of customer_id, state and city then the sum of sales is displayed on the basis of that.

Cube:

```
SELECT city, state, segment, category, AVG(sales)
FROM sales
GROUP BY CUBE(city, state, segment, category)
ORDER BY city ASC;
```

in the above cube operation it will extract the data having same city,state, segment and category and then those data will form a new cube and the average operation is performed on that particular cube and then the output is displayed which contains city,state,segment,category and average

2. Explain if Drill-down, Drill-across can be applied in relational database, Justify with a query implementation.

Yes, drill down and drill across can be applied on the relational database.

Drill Down:

Drilling down in a relational database means “adding a row header” to an existing SELECT statement. For instance, if you’re analyzing the sales of products at a manufacturer level, the select list of the query reads SELECT MANUFACTURER, SUM(SALES). Of course the rest of the query contains joint specifications and constraints on other parts of the database, such as time and geography. If you wish to drill down on the list of manufacturers to show the brands sold, you add the BRAND row header: SELECT MANUFACTURER, BRAND, SUM(SALES). Now each manufacturer row expands into multiple rows listing all the brands sold. This is the essence of drilling down.

Drill Across:

Drilling across simply means making separate queries against two or more fact tables where the row headers of each query consist of identical conformed attributes. The answer sets from the two queries are aligned by performing a sort-merge operation on the common dimension attribute row headers. BI tool vendors refer to this functionality by various names, including stitch and multipass query.

Outcomes:

CO4:Apply ETL processing and Online Analytical Processing on the warehouse data.

Conclusion: (Conclusion to be based on the outcomes achieved)

Through this experiment we executed OLAP operations and learnt how to use simple SQL operators like rollup, cube and grouping in creating easy aggregations directly inside the SQL. We executed these operators on the table created in our database and made various observations about the different results of these operators.

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of faculty in-charge with date

References:

- Paulraj Ponniah, “Data Warehousing: Fundamentals for IT Professionals”, Wiley India