# Hands-on Lab: Querying the Data Warehouse (Cubes, Rollups, Grouping Sets and Materialized Views)

Estimated time needed: 30 minutes

## Objectives

In this lab you will learn how to create:

- Grouping sets
- Rollup
- Cube
- Materialized Query Tables (MQT)

## Exercise 1 - Launch a PostgreSQL server instance on Cloud IDE and open up the pgAdmin Graphical User Interface.

This lab requires that you complete the previous lab Populate a Data Warehouse.

If you have not finished the Populate a Data Warehouse Lab yet, please finish it before you continue.

**GROUPING SETS, CUBE, and ROLLUP** allow us to easily create subtotals and grand totals in a variety of ways. All these operators are used along with the GROUP BY operator.

**GROUPING SETS** operator allows us to group data in a number of different ways in a single SELECT statement.

The **ROLLUP** operator is used to create subtotals and grand totals for a set of columns. The summarized totals are created based on the columns passed to the ROLLUP operator.

The **CUBE** operator produces subtotals and grand totals. In addition, it produces subtotals and grand totals for every permutation of the columns provided to the CUBE operator.

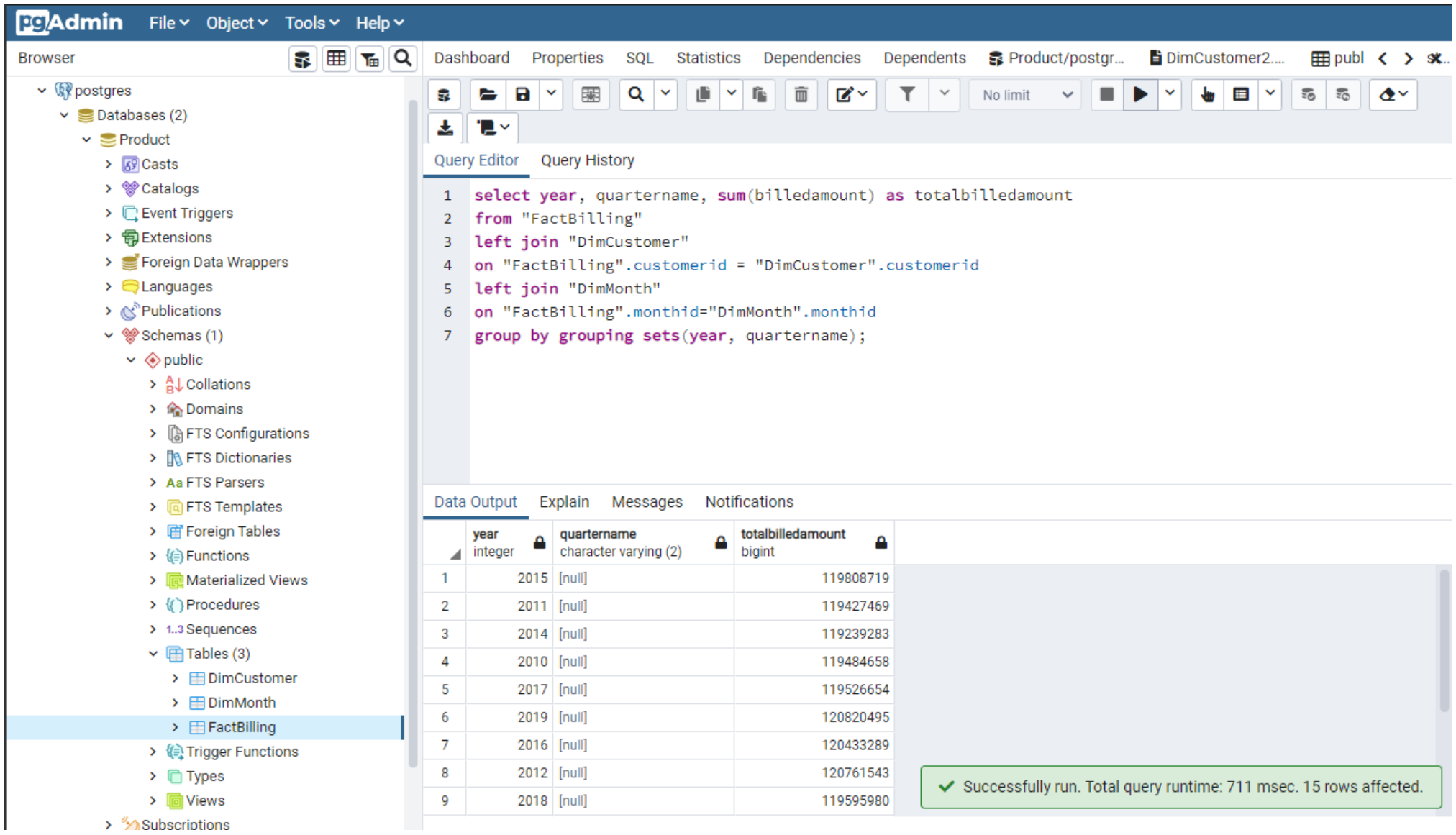
## Exercise 2 - Write a query using grouping sets

After you launch a PostgreSQL server instance on Cloud IDE and open up the pgAdmin Graphical User Interface run the below query.

To create a grouping set for three columns labeled year, category, and sum of billedamount, run the sql statement below.

```
select year,quartername, sum(billedamount) as totalbilledamount
from "FactBilling"
left join "DimCustomer"
on "FactBilling".customerid = "DimCustomer".customerid
left join "DimMonth"
on "FactBilling".monthid="DimMonth".monthid
group by grouping sets(year,quartername);
```

The partial output can be seen in the image below.

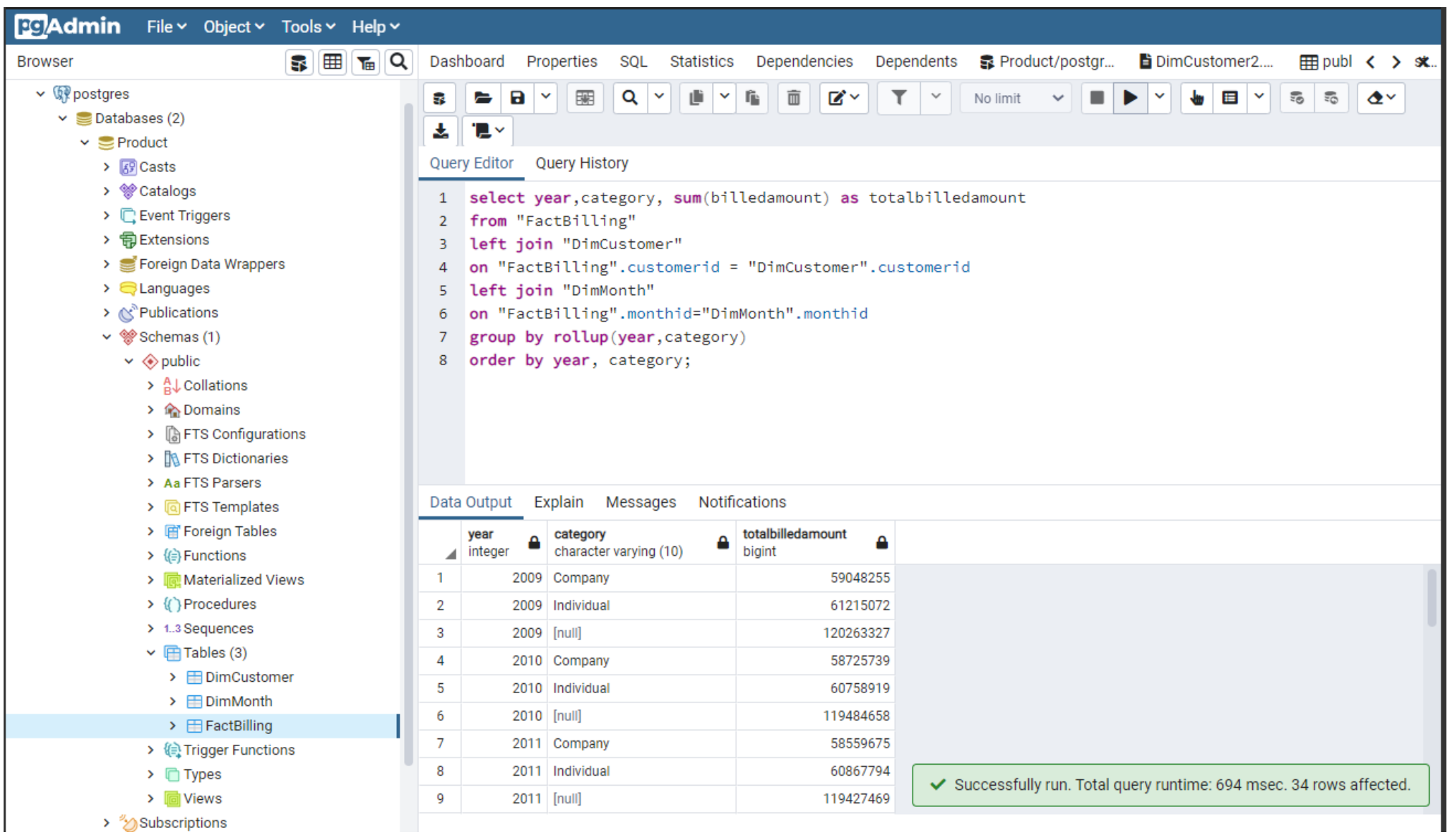



# Exercise 3 - Write a query using rollup

To create a rollup using the three columns year, category and sum of billedamount, run the sql statement below.

```
select year,category, sum(billedamount) as totalbilledamount
from "FactBilling"
left join "DimCustomer"
on "FactBilling".customerid = "DimCustomer".customerid
left join "DimMonth"
on "FactBilling".monthid="DimMonth".monthid
group by rollup(year,category)
order by year, category;
```

The partial output can be seen in the image below.

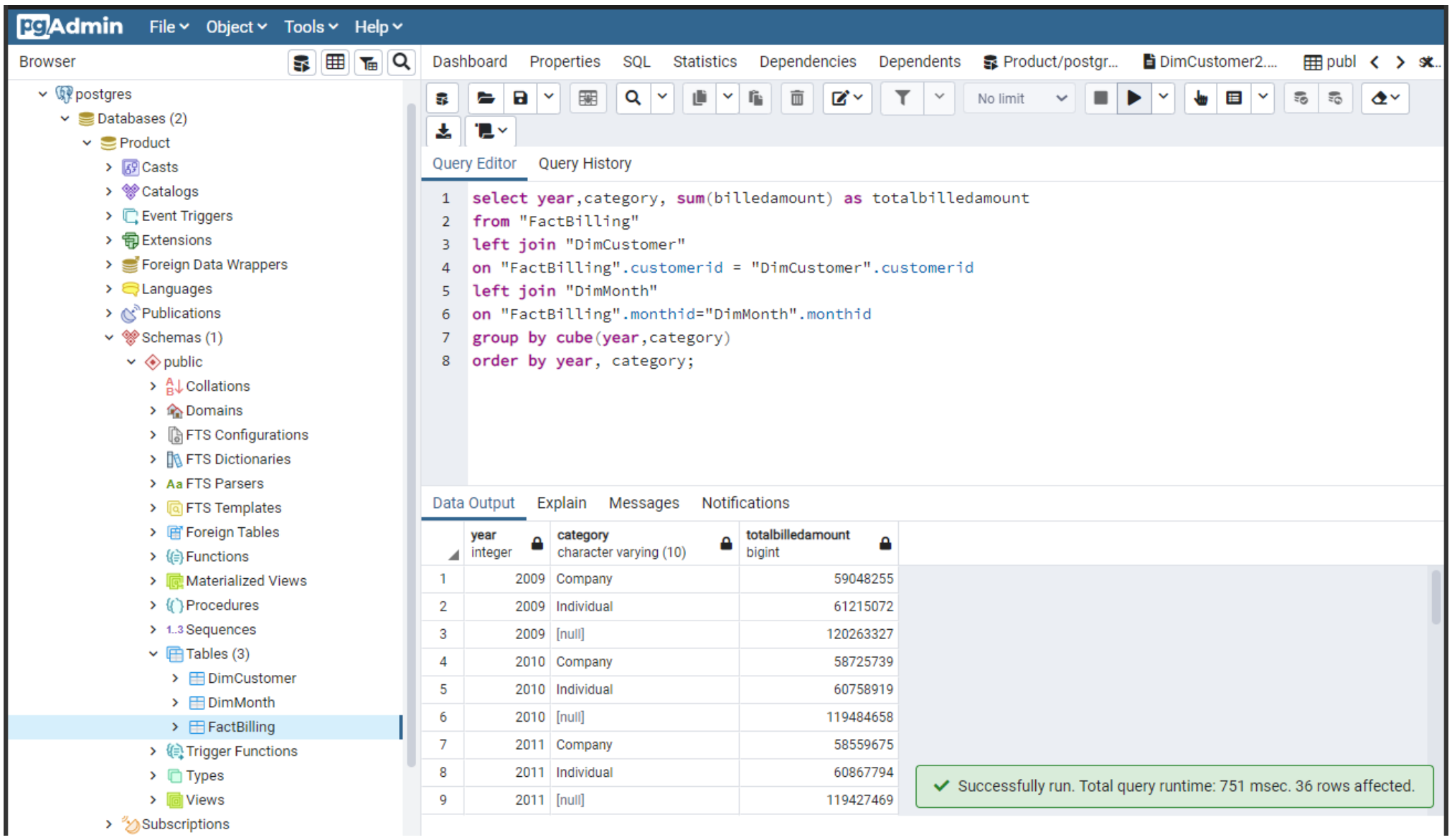


# Exercise 4 - Write a query using cube

To create a cube using the three columns labeled year, category, and sum of billedamount, run the sql statement below.

```
select year,category, sum(billedamount) as totalbilledamount
from "FactBilling"
left join "DimCustomer"
on "FactBilling".customerid = "DimCustomer".customerid
left join "DimMonth"
on "FactBilling".monthid="DimMonth".monthid
group by cube(year,category)
order by year, category;
```

The partial output can be seen in the image below.




# Exercise 5 - Create a Materialized Query Table(MQT)

In pgAdmin we can implement materialized views using Materialized Query Tables.

## Step 1: Create the MQT.

Execute the sql statement below to create an MQT named countrystats.

```
CREATE MATERIALIZED VIEW countrystats (country, year, totalbilledamount) AS
(select country, year, sum(billedamount)
from "FactBilling"
left join "DimCustomer"
on "FactBilling".customerid = "DimCustomer".customerid
left join "DimMonth"
on "FactBilling".monthid="DimMonth".monthid
group by country,year);
```

The above command creates an MQT named countrystats that has 3 columns.

- Country
- Year
- totalbilledamount

The MQT is essentially the result of the below query, which gives you the year, quartername and the sum of billed amount grouped by year and quartername.

```
select year, quartername, sum(billedamount) as totalbilledamount
from "FactBilling"
left join "DimCustomer"
on "FactBilling".customerid = "DimCustomer".customerid
left join "DimMonth"
on "FactBilling".monthid="DimMonth".monthid
group by grouping sets(year, quartername);
```

## Step 2: Populate/refresh data into the MQT.

Execute the sql statement below to populate the MQT countrystats.

```
REFRESH MATERIALIZED VIEW countrystats;
```

The command above populates the MQT with relevant data.

Step 3: Query the MQT.

Once an MQT is refreshed, you can query it.

Execute the sql statement below to query the MQT countrystats.

```
select * from countrystats;
```

# Practice exercises

## Problem 1: Create a grouping set for the columns year, quartername, sum(billedamount).

▼ Click here for Hint

Make sure that this table contains the year and quartername.

▼ Click here for Solution

```
select year, quartername, sum(billedamount) as totalbilledamount
from "FactBilling"
left join "DimCustomer"
on "FactBilling".customerid = "DimCustomer".customerid
left join "DimMonth"
on "FactBilling".monthid="DimMonth".monthid
group by grouping sets(year, quartername);
```

## Problem 2: Create a rollup for the columns country, category, sum(billedamount).

▼ Click here for Hint

Select columns year, quartername, sum(billedamount), and use a group by query and join the dimcustomer and dimmonth tables to factbilling table_.

▼ Click here for Solution

```
select year, quartername,  sum(billedamount) as totalbilledamount
from "FactBilling"
left join "DimCustomer"
on "FactBilling".customerid = "DimCustomer".customerid
left join "DimMonth"
on "FactBilling".monthid="DimMonth".monthid
group by rollup(year, quartername)
order by year, quartername;
```

## Problem 3: Create a cube for the columns year,country, category, sum(billedamount).

▼ Click here for Hint

Select columns year,quartername , sum of billedamount, and use a group by query and join the dimcustomer and dimmonth tables to factbilling table_.

▼ Click here for Solution

```
select year, quartername, sum(billedamount) as totalbilledamount
from "FactBilling"
left join "DimCustomer"
on "FactBilling".customerid = "DimCustomer".customerid
left join "DimMonth"
on "FactBilling".monthid="DimMonth".monthid
group by cube(year,quartername);
```

## Problem 4: Create an MQT named average_billamount with columns year, quarter, category, country, average_bill_amount.

▼ Click here for Hint

Select columns year, quarter, category, country, avg(billedamount), and use a group by query and join the dimcustomer and dimmonth tables to factbilling table_.

▼ Click here for Solution

```
CREATE MATERIALIZED VIEW average_billamount (year,quarter,category,country, average_bill_amount) AS
    (select   year,quarter,category,country, avg(billedamount) as average_bill_amount
    from "FactBilling"
    left join  "DimCustomer"
    on "FactBilling".customerid =  "DimCustomer".customerid
    left join "DimMonth"
    on "FactBilling".monthid="DimMonth".monthid
    group by year,quarter,category,country
    );
```

```
refresh MATERIALIZED VIEW average_billamount;
```

## Congratulations! You have successfully finished the Populating a Data Warehouse lab.

# Author

Amrutha Rao

## Change Log

| Date (YYYY-MM-DD) | Version | Changed By | Change Description |
|---|---|---|---|
| 2022-04-14 | 0.2 | Amrutha Rao | converted initial version to pgAdmin workaround. |
| 2021-09-29 | 0.1 | Ramesh Sannareddy | Created initial version of the lab |