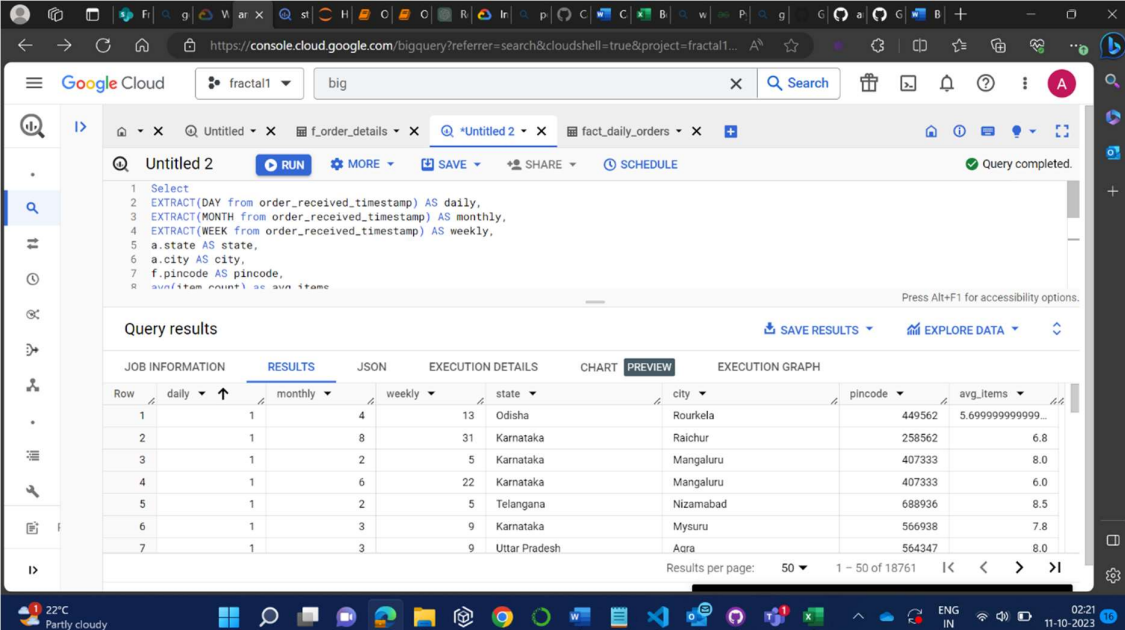


# Capstone Project Big Query Analysis

Q.1. Average number of items per order - daily, monthly, weekly, state, city, pincode.

```
Select
EXTRACT(DAY from order_received_timestamp) AS daily,
EXTRACT(MONTH from order_received_timestamp) AS monthly,
EXTRACT(WEEK from order_received_timestamp) AS weekly,
a.state AS state,
a.city AS city,
f.pincode AS pincode,
avg(item_count) as avg_items

From `fractal1a.star_schema.fact_daily_orders` f
left join `fractal1a.star_schema.dim_customer` c
  on f.customerid = c.customerid
left join `fractal1a.star_schema.dim_address` a
  on c.address_id = a.address_id
group by
daily,
monthly,
weekly,
state,
city,
pincode
```



The screenshot shows the Google Cloud BigQuery console interface. The query editor at the top contains the SQL code for Q.1. Below the editor, the 'Query results' section is displayed, showing a table with 8 columns: Row, daily, monthly, weekly, state, city, pincode, and avg\_items. The table contains 7 rows of data. The interface also shows tabs for 'JOB INFORMATION', 'RESULTS', 'JSON', 'EXECUTION DETAILS', 'CHART', 'PREVIEW', and 'EXECUTION GRAPH'. The 'RESULTS' tab is currently selected.

Row	daily	monthly	weekly	state	city	pincode	avg_items
1	1	4	13	Odisha	Rourkela	449562	5.699999999999999...
2	1	8	31	Karnataka	Raichur	258562	6.8
3	1	2	5	Karnataka	Mangaluru	407333	8.0
4	1	6	22	Karnataka	Mangaluru	407333	6.0
5	1	2	5	Telangana	Nizamabad	688936	8.5
6	1	3	9	Karnataka	Mysuru	566938	7.8
7	1	3	9	Uttar Pradesh	Agra	564347	8.0

Q.2. Average amount of sales per order - daily, monthly, weekly, state, city, pincode.

```
Select
EXTRACT(DAY from f.order_received_timestamp) AS daily,
```

```

EXTRACT(MONTH from f.order_received_timestamp) AS monthly,
EXTRACT(WEEK from f.order_received_timestamp) AS weekly,
a.state AS state,
a.city AS city,
f.pincode AS pincode,
avg(order_amount) as avg_sales

```

```

From `fractal1a.star_schema.fact_daily_orders` f
left join `fractal1a.star_schema.dim_customer` c
    on f.customerid = c.customerid
left join `fractal1a.star_schema.dim_address` a
    on c.address_id = a.address_id
group by
daily,
monthly,
weekly,
sTATE,
city,
pincode

```

The screenshot shows the Google Cloud BigQuery console. The query editor displays a query that calculates the average amount of sales per order, grouped by daily, monthly, weekly, state, city, and pincode. The query has been executed successfully, and the results are shown in a table format.

**Query results**

Row	daily	monthly	weekly	state	city	pincode	avg_sales
1	1	4	13	Odisha	Rourkela	449562	14321.8
2	1	8	31	Karnataka	Raichur	258562	14792.3
3	1	2	5	Karnataka	Mangaluru	407333	13305.33333333...
4	1	6	22	Karnataka	Mangaluru	407333	7962.428571428...
5	1	2	5	Telangana	Nizamabad	688936	11125.5
6	1	3	9	Karnataka	Mysuru	566938	8387.2
7	1	3	9	Uttar Pradesh	Agra	564347	11051.0

Results per page: 50 | 1 - 50 of 18761

Q.3. Total number of units sold per day of a product SKU and its monthly trend.

```

select
p.productname,p.sku,
EXTRACT(DAY from f.order_delivery_timestamp) as day,
EXTRACT(MONTH from f.order_delivery_timestamp) as month,
sum(quantity) as units_sold
From `fractalb.star_schema.dim_product` p

```

```

join `fractalb.star_schema.f_order_details` f
on p.productid = f.productid
group by productname,
sku,
day,
month

```

The screenshot shows the Google Cloud BigQuery console interface. The query editor displays a SQL query titled 'Untitled 2' with the following text:

```

1 -- Q3 total number of units sold per day of a product SKU and its monthly trend
2
3 select
4   p.productname, p.sku,
5   EXTRACT(DAY from f.order_delivery_timestamp) as day,
6   EXTRACT(MONTH from f.order_delivery_timestamp) as month,
7   sum(quantity) as units_sold
8 from `fractalb.star_schema.f_order_details` f

```

The query results are displayed in a table with the following columns: productname, sku, day, month, and units\_sold. The results show data for 'HUL gram flour' across different days and months.

Row	productname	sku	day	month	units_sold
1	HUL gram flour	1kg	22	1	69
2	HUL gram flour	1kg	27	1	46
3	HUL gram flour	1kg	8	1	46
4	HUL gram flour	1kg	21	1	13
5	HUL gram flour	1kg	19	1	74
6	HUL gram flour	1kg	15	1	11
7	HUL gram flour	1kg	16	1	73

The console also shows the 'Query results' section with tabs for 'RESULTS', 'JSON', 'EXECUTION DETAILS', 'CHART', 'PREVIEW', and 'EXECUTION GRAPH'. The 'RESULTS' tab is currently selected, showing the table of results. The bottom status bar indicates 'Results per page: 50' and '1 - 50 of 14051'.

Q.4. Total Order Amount on daily basis, also to be able to split by product and geography.

```

select
EXTRACT(DATE from f.order_received_timestamp) AS daily,
o.productid,
a.city,
sum(order_amount) total_sales
from `fractal1a.star_schema.fact_daily_orders` f
join `fractal1a.star_schema.f_order_details` o on f.orderid = o.orderid
join `fractal1a.star_schema.dim_customer` c on f.customerid = c.customerid
join `fractal1a.star_schema.dim_address` a on c.address_id = a.address_id
group by
daily, productid, city
order by daily

```

The screenshot shows the Google Cloud BigQuery console. The query editor contains the following SQL code:

```

1 -- 4 Total Order Amount on daily basis, also to be able to split by product and geography
2 select
3   EXTRACT(DATE from f.order_received_timestamp) AS daily,
4   o.productid,
5   a.city,
6   sum(order_amount) total_sales
7 from `fractal1a.star_schema.fact_daily_orders` f
8 join `fractal1a.star_schema.dim_order_details` o on f.orderid = o.orderid
9 join `fractal1a.star_schema.dim_address` a on o.address_id = a.address_id

```

The query results are displayed in a table with the following columns: daily, productid, city, and total\_sales. The results show data for 7 rows, all dated 2023-01-01, with cities Agra and Bengaluru.

Row	daily	productid	city	total_sales
1	2023-01-01	32	Agra	195429
2	2023-01-01	8	Agra	64334
3	2023-01-01	25	Agra	190896
4	2023-01-01	19	Agra	104846
5	2023-01-01	30	Agra	64334
6	2023-01-01	39	Agra	221323
7	2023-01-01	4	Bengaluru	93773

Q.5. Distribution of orders according to area ( state, city, pincode etc)

```

select
a.state,
a.city,
count(distinct f.orderid) NumberOfOrders
from `fractal1a.star_schema.fact_daily_orders` f
join `fractal1a.star_schema.dim_customer` c on f.customerid = c.customerid
join `fractal1a.star_schema.dim_address` a on c.address_id = a.address_id
group by
state,city

```

The screenshot shows the Google Cloud BigQuery console. The query editor contains the following SQL code:

```

1 -- Q5 Distribution of orders according to area ( state, city, pincode etc)
2 select
3   a.state,
4   a.city,
5   count(distinct f.orderid) NumberOfOrders
6 from `fractal1a.star_schema.fact_daily_orders` f
7 join `fractal1a.star_schema.dim_customer` c on f.customerid = c.customerid
8 join `fractal1a.star_schema.dim_address` a on c.address_id = a.address_id

```

The query results are displayed in a table with the following columns: state, city, and NumberOfOrders. The results show data for 7 rows, with states Odisha, Uttar Pradesh, Karnataka, Telangana, and Karnataka, and cities Rourkela, Kanpur, Raichur, Nizamabad, Warangal, Bhubaneswar, and Mangaluru.

Row	state	city	NumberOfOrders
1	Odisha	Rourkela	1067
2	Uttar Pradesh	Kanpur	1195
3	Karnataka	Raichur	742
4	Telangana	Nizamabad	1171
5	Telangana	Warangal	1007
6	Odisha	Bhubaneswar	1145
7	Karnataka	Mangaluru	985

Q.6. Average order amount per customer on daily basis.

```
select
c.customerid AS id,
c.name AS name,
EXTRACT(DATE from order_received_timestamp) date,
round(avg(order_amount),2) AS OrderAmount
from `fractal1a.star_schema.fact_daily_orders` f
left join `fractal1a.star_schema.dim_customer` c
on f.customerid = c.customerid
group by
id, date, name
```

The screenshot shows the Google Cloud BigQuery console interface. The query editor displays a SQL query for Q6, which calculates the average order amount per customer on a daily basis. The query is as follows:

```
-- Q6 Average order amount per customer on daily basis
select
c.customerid AS id,
c.name AS name,
EXTRACT(DATE from order_received_timestamp) date,
round(avg(order_amount),2) AS OrderAmount
from `fractal1a.star_schema.fact_daily_orders` f
left join `fractal1a.star_schema.dim_customer` c
on f.customerid = c.customerid
group by
id, date, name
```

The query results are displayed in a table with the following columns: Row, id, name, date, and OrderAmount. The results show 7 rows of data for customer Jivika Goswami.

Row	id	name	date	OrderAmount
1	256	Jivika Goswami	2023-02-16	14566.56
2	256	Jivika Goswami	2023-02-14	11449.0
3	256	Jivika Goswami	2023-03-06	9973.73
4	256	Jivika Goswami	2023-02-27	9055.3
5	256	Jivika Goswami	2023-01-23	14051.0
6	256	Jivika Goswami	2023-04-01	14321.8
7	256	Jivika Goswami	2023-02-08	10993.0

The console also shows the query execution details, including the job ID and the execution graph. The results are displayed in a table with 5 columns: Row, id, name, date, and OrderAmount. The results show 7 rows of data for customer Jivika Goswami.

Q.7. New Customers on daily basis.

```
select
START_DATE,
count(customerid) NewCustomers
from `fractal1a.star_schema.dim_customer`
where customerid in (select customerid from `fractal1a.star_schema.dim_customer`
group by customerid having count(*)=1)
group by START_DATE
```

The screenshot shows the Google Cloud BigQuery console interface. The query editor displays a SQL query titled "Untitled 2" that counts new customers daily. The query is as follows:

```

1 -- Q7 New Customers on daily basis
2 select
3   START_DATE,
4   count(customerid) NewCustomers
5 from `fractal1a.star_schema.dim_customer`
6 where customerid in (select customerid from `fractal1a.star_schema.dim_customer`
7   group by customerid having count(*)=1)
8 group by START_DATE

```

The query has been executed successfully, and the results are displayed in a table with two columns: `START_DATE` and `NewCustomers`. The results show data for dates from 2023-04-09 to 2023-04-15.

Row	START_DATE	NewCustomers
1	2023-04-09	2
2	2023-04-10	1
3	2023-04-11	4
4	2023-04-12	1
5	2023-04-13	5
6	2023-04-14	1
7	2023-04-15	3

The console also shows options to save results, explore data, and view execution details. The bottom status bar indicates the system temperature is 22°C and the time is 02:50 on 11-10-2023.

Q.8. Total count of customers everyday.

```

select distinct
EXTRACT(DATE from order_received_timestamp) Dates,
count(*) over (partition by EXTRACT(DATE from order_received_timestamp))
CustomerCounts
from `fractal1a.star_schema.fact_daily_orders`
group by
order_received_timestamp
order by Dates

```

The screenshot shows the Google Cloud BigQuery console interface. The query editor displays a SQL query titled "Untitled 2" that calculates the total count of customers every day. The query is as follows:

```

1 -- Q8 Total count of customers everyday
2 select distinct
3   EXTRACT(DATE from order_received_timestamp) Dates,
4   count(*) over (partition by EXTRACT(DATE from order_received_timestamp)) CustomerCounts
5 from `fractal1a.star_schema.fact_daily_orders`
6 group by
7   order_received_timestamp
8 order by Dates

```

The query has been executed successfully, and the results are displayed in a table with two columns: `START_DATE` and `NewCustomers`. The results show data for dates from 2023-04-09 to 2023-04-15.

Row	START_DATE	NewCustomers
1	2023-04-09	2
2	2023-04-10	1
3	2023-04-11	4
4	2023-04-12	1
5	2023-04-13	5
6	2023-04-14	1
7	2023-04-15	3

The console also shows options to save results, explore data, and view execution details. The bottom status bar indicates the system temperature is 22°C and the time is 02:52 on 11-10-2023.

Q.9 Average time to delivery order. Min and Max time. To be able to slice and dice on hour, weekday, weekend, daily, monthly, geography.

```
select distinct
EXTRACT(DATE from f.order_delivery_timestamp) DATES,
EXTRACT(WEEK from f.order_delivery_timestamp) WEEKS,
EXTRACT(DAYOFWEEK from f.order_delivery_timestamp) WEEKDAYS,
EXTRACT(MONTH from f.order_delivery_timestamp) MONTHS,
a.City,
Min(f.order_delivery_time_seconds) MinDeliveryTime,
max(f.order_delivery_time_seconds) MaxDeliveryTime,
avg(f.order_delivery_time_seconds) AvgDeliveryTime,
from `fractal1a.star_schema.fact_daily_orders` f
join `fractal1a.star_schema.dim_customer` c on f.customerid = c.customerid
join `fractal1a.star_schema.dim_address` a on c.address_id = a.address_id
group by
DATES,
WEEKS,
WEEKDAYS,
MONTHS,
City
```

The screenshot shows the Google Cloud BigQuery console. The query editor displays the SQL query for Q.9. The query results are shown in a table with columns: DATES, WEEKS, WEEKDAYS, MONTHS, City, MinDeliveryTime, MaxDeliveryTime, and AvgDeliveryTime. The results are filtered to show only rows where the City is 'Rourkela'.

Row	DATES	WEEKS	WEEKDAYS	MONTHS	City	MinDeliveryTime	MaxDeliveryTime	AvgDeliveryT
1	2023-09-07	36	5	9	Rourkela	1872831	20571559	14127389.03
2	2023-07-21	29	6	7	Rourkela	12980324	14733596	13660822.84
3	2023-06-27	26	3	6	Rourkela	7480934	15078098	12295931.20
4	2023-09-09	36	7	9	Rourkela	2335952	16141405	11176462.74
5	2023-05-06	18	7	5	Rourkela	5222510	8138910	5858672.869
6	2023-06-21	25	4	6	Rourkela	4958729	14658289	7443490.392

Q.10. Total orders : to be able to slice and dice on hour, weekday, weekend, daily, monthly, geography.

```
select distinct
EXTRACT(DATE from f.order_delivery_timestamp) DATES,
EXTRACT(WEEK from f.order_delivery_timestamp) WEEKS,
EXTRACT(DAYOFWEEK from f.order_delivery_timestamp) WEEKDAYS,
EXTRACT(MONTH from f.order_delivery_timestamp) MONTHS,
```



```

a.City,
count(orderid) NumberOfOrders
from `fractalb.star_schema.fact_daily_orders` f
join `fractalb.star_schema.dim_customer` c on f.customerid = c.customerid
join `fractalb.star_schema.dim_address` a on c.address_id = a.address_id
group by
DATES, WEEKS, WEEKDAYS, MONTHS, City

```

The screenshot shows the Google Cloud BigQuery console interface. The query editor displays a SQL query that extracts date components (DATES, WEEKS, WEEKDAYS, MONTHS) and counts the number of orders grouped by these components and city. The query has been executed successfully, as indicated by the 'Query completed' status.

**Query results**

Row	DATES	WEEKS	WEEKDAYS	MONTHS	City	NumberOfOrders
1	2023-09-07	36	5	9	Rourkela	52
2	2023-07-21	29	6	7	Rourkela	19
3	2023-06-27	26	3	6	Rourkela	35
4	2023-09-09	36	7	9	Rourkela	24
5	2023-05-06	18	7	5	Rourkela	23
6	2023-06-21	25	4	6	Rourkela	28
7	2023-08-10	32	5	8	Rourkela	55

Results per page: 50 | 1 - 50 of 4096