E-commerce SQL Analysis

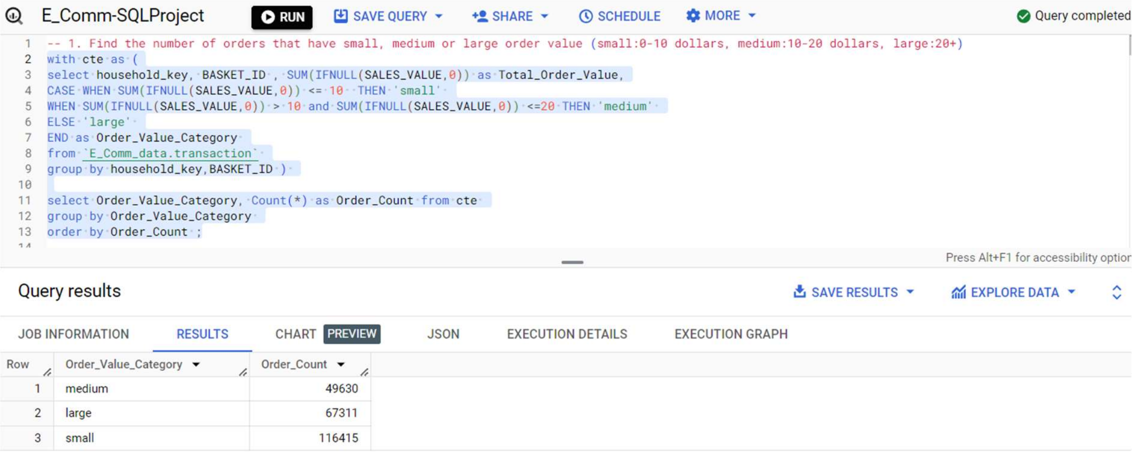
Problem Statement

Analyzing the sales, product, and customer data for an e-commerce company. getting various insights and calculating various KPI and data with SQL in Big Query.

SQL Queries :-

```
-- 1. Find the number of orders that have small, medium or large order value
(small:0-10 dollars, medium:10-20 dollars, large:20+)

with cte as (
select household_key, BASKET_ID , SUM(IFNULL(SALES_VALUE,0)) as Total_Order_Value,
CASE WHEN SUM(IFNULL(SALES_VALUE,0)) <= 10  THEN 'small'
WHEN SUM(IFNULL(SALES_VALUE,0)) > 10 and SUM(IFNULL(SALES_VALUE,0)) <=20 THEN
'medium'
ELSE 'large'
END as Order_Value_Category
from `E_Comm_data.transaction`
group by household_key,BASKET_ID )

select Order_Value_Category, Count(*) as Order_Count from cte
group by Order_Value_Category
order by Order_Count ;


-- Note : Even if we group by only BASKET_ID the count of O/P rows remains same
which means that Each BASKET_ID group has same household_key.
```

Output: -



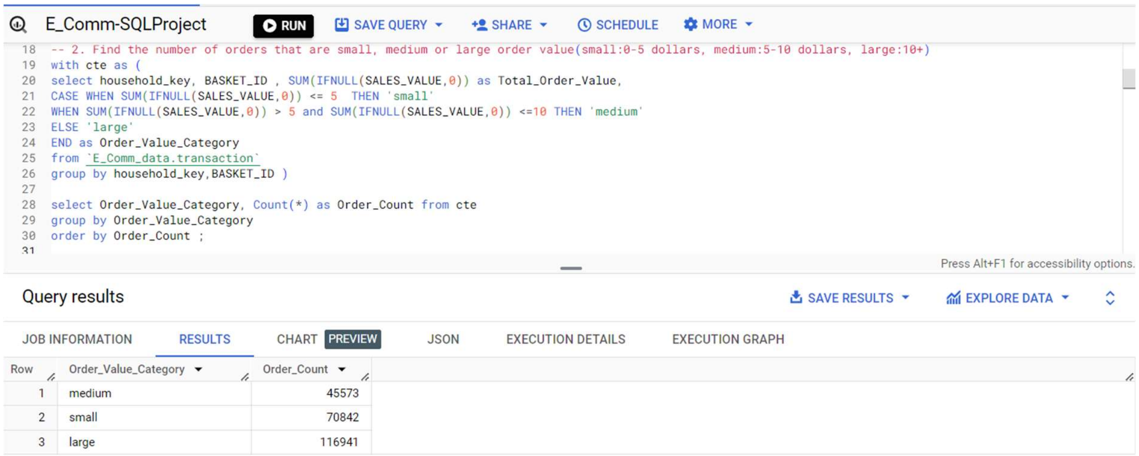
```
-- 1. Find the number of orders that have small, medium or large order value (small:0-10 dollars, medium:10-20 dollars, large:20+)
with cte as (
select household_key, BASKET_ID , SUM(IFNULL(SALES_VALUE,0)) as Total_Order_Value,
CASE WHEN SUM(IFNULL(SALES_VALUE,0)) <= 10  THEN 'small'
WHEN SUM(IFNULL(SALES_VALUE,0)) > 10 and SUM(IFNULL(SALES_VALUE,0)) <=20 THEN 'medium'
ELSE 'large'
END as Order_Value_Category
from `E_Comm_data.transaction`
group by household_key,BASKET_ID )

select Order_Value_Category, Count(*) as Order_Count from cte
group by Order_Value_Category
order by Order_Count ;
```

Row	Order_Value_Category	Order_Count
1	medium	49630
2	large	67311
3	small	116415

-----------------------------------X--X---------------------------------------

```
-- 2. Find the number of orders that are small, medium or large order
value(small:0-5 dollars, medium:5-10 dollars, large:10+)

with cte as (
select household_key, BASKET_ID , SUM(IFNULL(SALES_VALUE,0)) as Total_Order_Value,
CASE WHEN SUM(IFNULL(SALES_VALUE,0)) <= 5  THEN 'small'
WHEN SUM(IFNULL(SALES_VALUE,0)) > 5 and SUM(IFNULL(SALES_VALUE,0)) <=10 THEN
'medium'
ELSE 'large'
END as Order_Value_Category
from `E_Comm_data.transaction`
group by household_key,BASKET_ID )

select Order_Value_Category, Count(*) as Order_Count from cte
group by Order_Value_Category
order by Order_Count ;
```

Output: -



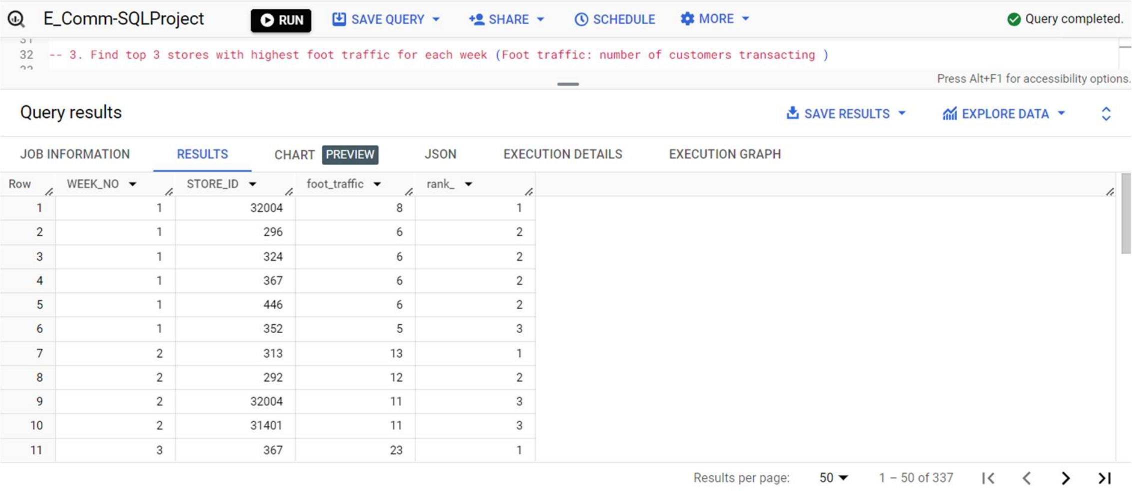
```
-- 2. Find the number of orders that are small, medium or large order value(small:0-5 dollars, medium:5-10 dollars, large:10+)
with cte as (
select household_key, BASKET_ID , SUM(IFNULL(SALES_VALUE,0)) as Total_Order_Value,
CASE WHEN SUM(IFNULL(SALES_VALUE,0)) <= 5  THEN 'small'
WHEN SUM(IFNULL(SALES_VALUE,0)) > 5 and SUM(IFNULL(SALES_VALUE,0)) <=10 THEN 'medium'
ELSE 'large'
END as Order_Value_Category
from `E_Comm_data.transaction`
group by household_key,BASKET_ID )

select Order_Value_Category, Count(*) as Order_Count from cte
group by Order_Value_Category
order by Order_Count ;
```

Row	Order_Value_Category	Order_Count
1	medium	45573
2	small	70842
3	large	116941

-----------------------------------X---X---------------------------------------

```
-- 3. Find top 3 stores with highest foot traffic for each week (Foot traffic:
number of customers transacting )

with cte as (
select WEEK_NO ,STORE_ID, Count(DISTINCT BASKET_ID) as foot_traffic
from `E_Comm_data.transaction`
group by WEEK_NO ,STORE_ID
order by WEEK_NO, STORE_ID
) , cte2 as (
select WEEK_NO,STORE_ID,foot_traffic, DENSE_RANK() OVER(partition by WEEK_NO ORDER
BY foot_traffic DESC) as rank_
from cte )

select * from cte2 where rank_ <= 3
order by WEEK_NO ,rank_;
```

Output: -



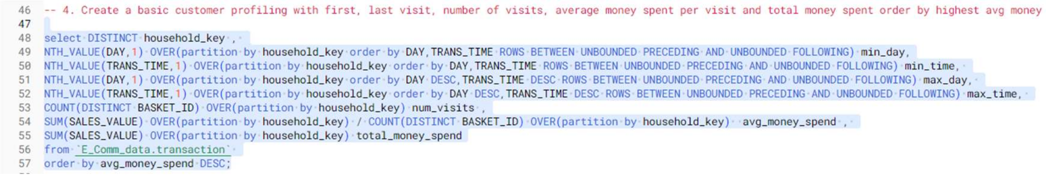
```
-- 3. Find top 3 stores with highest foot traffic for each week (Foot traffic: number of customers transacting )
```

Row	WEEK_NO	STORE_ID	foot_traffic	rank_
1	1	32004	8	1
2	1	296	6	2
3	1	324	6	2
4	1	367	6	2
5	1	446	6	2
6	1	352	5	3
7	2	313	13	1
8	2	292	12	2
9	2	32004	11	3
10	2	31401	11	3
11	3	367	23	1

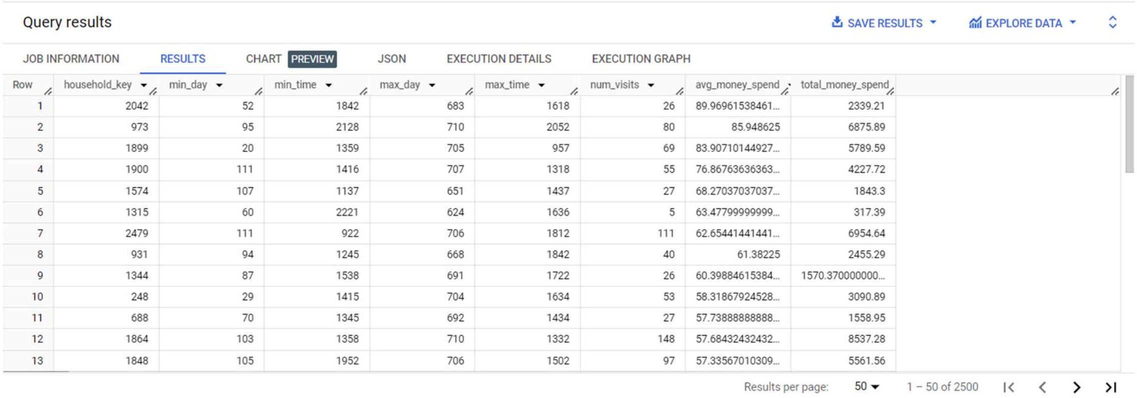
-----------------------------------X---X---------------------------------------

```
-- 4. Create a basic customer profiling with first, last visit, number of visits,
average money spent per visit and total money spent order by highest avg money

select DISTINCT household_key ,
NTH_VALUE(DAY,1) OVER(partition by household_key order by DAY,TRANS_TIME ROWS
BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) min_day,
NTH_VALUE(TRANS_TIME,1) OVER(partition by household_key order by DAY,TRANS_TIME
ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) min_time,
NTH_VALUE(DAY,1) OVER(partition by household_key order by DAY DESC,TRANS_TIME DESC
ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) max_day,
NTH_VALUE(TRANS_TIME,1) OVER(partition by household_key order by DAY
DESC,TRANS_TIME DESC ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING)
max_time,
COUNT(DISTINCT BASKET_ID) OVER(partition by household_key) num_visits ,
SUM(SALES_VALUE) OVER(partition by household_key) / COUNT(DISTINCT BASKET_ID)
OVER(partition by household_key)  avg_money_spend ,
SUM(SALES_VALUE) OVER(partition by household_key) total_money_spend
from `E_Comm_data.transaction`
order by avg_money_spend DESC;
```

Output: -



```
-- 4. Create a basic customer profiling with first, last visit, number of visits, average money spent per visit and total money spent order by highest avg money

select DISTINCT household_key ,
NTH_VALUE(DAY,1) OVER(partition by household_key order by DAY,TRANS_TIME ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) min_day,
NTH_VALUE(TRANS_TIME,1) OVER(partition by household_key order by DAY,TRANS_TIME ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) min_time,
NTH_VALUE(DAY,1) OVER(partition by household_key order by DAY DESC,TRANS_TIME DESC ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) max_day,
NTH_VALUE(TRANS_TIME,1) OVER(partition by household_key order by DAY DESC,TRANS_TIME DESC ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) max_time,
COUNT(DISTINCT BASKET_ID) OVER(partition by household_key) num_visits ,
SUM(SALES_VALUE) OVER(partition by household_key) / COUNT(DISTINCT BASKET_ID) OVER(partition by household_key)  avg_money_spend ,
SUM(SALES_VALUE) OVER(partition by household_key) total_money_spend
from `E_Comm_data.transaction`
order by avg_money_spend DESC;
```



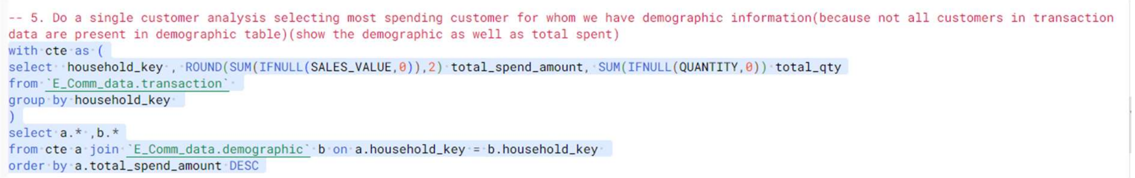
Row	household_key	min_day	min_time	max_day	max_time	num_visits	avg_money_spend	total_money_spend
1	2042	52	1842	683	1618	26	89.96961538461...	2339.21
2	973	95	2128	710	2052	80	85.948625	6875.89
3	1899	20	1359	705	957	69	83.90710144927...	5789.59
4	1900	111	1416	707	1318	55	76.86763636363...	4227.72
5	1574	107	1137	651	1437	27	68.27037037037...	1843.3
6	1315	60	2221	624	1636	5	63.47799999999...	317.39
7	2479	111	922	706	1812	111	62.65441441441...	6954.64
8	931	94	1245	668	1842	40	61.38225	2455.29
9	1344	87	1538	691	1722	26	60.39884615384...	1570.370000000...
10	248	29	1415	704	1634	53	58.31867924528...	3090.89
11	688	70	1345	692	1434	27	57.73888888888...	1558.95
12	1864	103	1358	710	1332	148	57.68432432432...	8537.28
13	1848	105	1952	706	1502	97	57.33567010309...	5561.56

Results per page: 50 1 – 50 of 2500

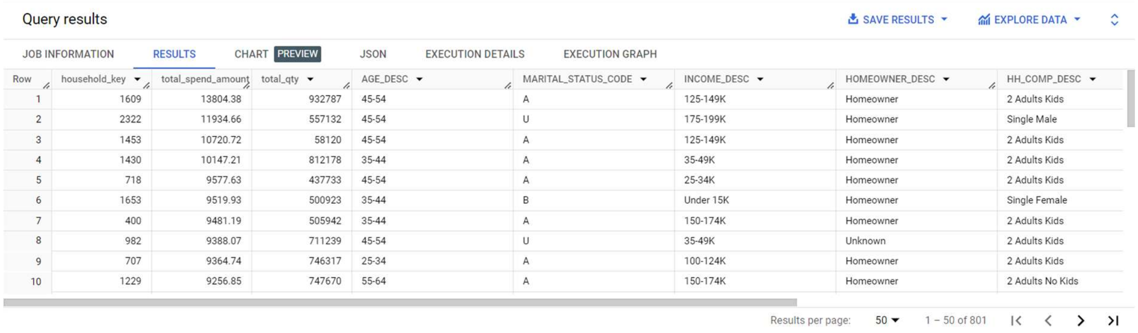
------------------------------X---------------------------------------X----------------------------------

```
-- 5. Do a single customer analysis selecting most spending customer for whom we
have demographic information(because not all customers in transaction data are
present in demographic table)(show the demographic as well as total spent)

with cte as (
select  household_key , ROUND(SUM(IFNULL(SALES_VALUE,0)),2) total_spend_amount,
SUM(IFNULL(QUANTITY,0)) total_qty
from `E_Comm_data.transaction`
group by household_key
)
select a.* ,b.*
from cte a join `E_Comm_data.demographic` b on a.household_key = b.household_key
order by a.total_spend_amount DESC
```

Output: -



```
-- 5. Do a single customer analysis selecting most spending customer for whom we have demographic information(because not all customers in transaction
data are present in demographic table)(show the demographic as well as total spent)
with cte as (
select  household_key , ROUND(SUM(IFNULL(SALES_VALUE,0)),2) total_spend_amount, SUM(IFNULL(QUANTITY,0)) total_qty
from `E_Comm_data.transaction`
group by household_key
)
select a.* ,b.*
from cte a join `E_Comm_data.demographic` b on a.household_key = b.household_key
order by a.total_spend_amount DESC
```



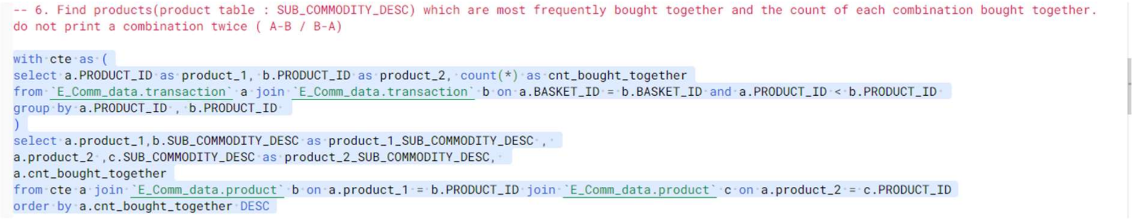
Row	household_key	total_spend_amount	total_qty	AGE_DESC	MARITAL_STATUS_CODE	INCOME_DESC	HOMEOWNER_DESC	HH_COMP_DESC
1	1609	13804.38	932787	45-54	A	125-149K	Homeowner	2 Adults Kids
2	2322	11934.66	557132	45-54	U	175-199K	Homeowner	Single Male
3	1453	10720.72	58120	45-54	A	125-149K	Homeowner	2 Adults Kids
4	1430	10147.21	812178	35-44	A	35-49K	Homeowner	2 Adults Kids
5	718	9577.63	437733	45-54	A	25-34K	Homeowner	2 Adults Kids
6	1653	9519.93	500923	35-44	B	Under 15K	Homeowner	Single Female
7	400	9481.19	505942	35-44	A	150-174K	Homeowner	2 Adults Kids
8	982	9388.07	711239	45-54	U	35-49K	Unknown	2 Adults Kids
9	707	9364.74	746317	25-34	A	100-124K	Homeowner	2 Adults Kids
10	1229	9256.85	747670	55-64	A	150-174K	Homeowner	2 Adults No Kids

Results per page: 50 1 – 50 of 801

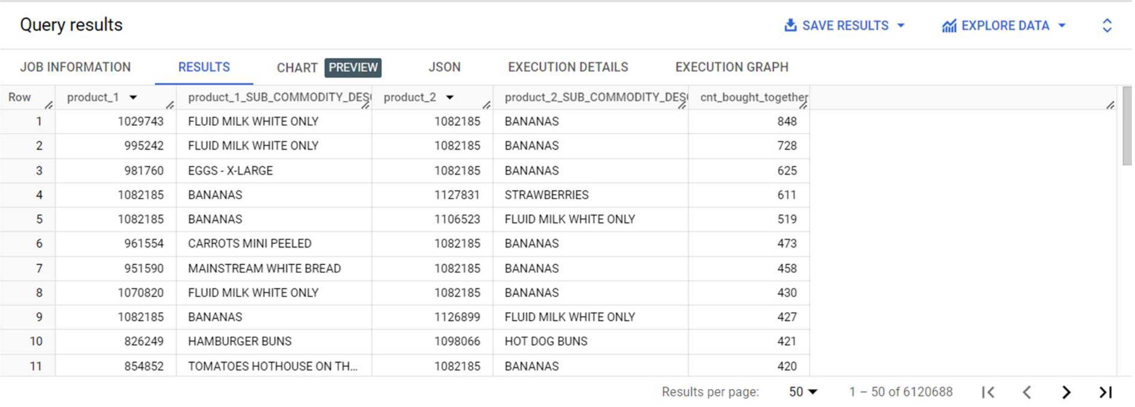
------------------------------X---------------------------------------X----------------------------------

```
-- 6. Find products(product table : SUB_COMMODITY_DESC) which are most frequently
bought together and the count of each combination bought together. do not print a
combination twice ( A-B / B-A)

with cte as (
select a.PRODUCT_ID as product_1, b.PRODUCT_ID as product_2, count(*) as
cnt_bought_together
from `E_Comm_data.transaction` a join `E_Comm_data.transaction` b on a.BASKET_ID =
b.BASKET_ID and a.PRODUCT_ID < b.PRODUCT_ID
group by a.PRODUCT_ID , b.PRODUCT_ID
)
select a.product_1,b.SUB_COMMODITY_DESC as product_1_SUB_COMMODITY_DESC ,
a.product_2 ,c.SUB_COMMODITY_DESC as product_2_SUB_COMMODITY_DESC,
a.cnt_bought_together
from cte a join `E_Comm_data.product` b on a.product_1 = b.PRODUCT_ID join
`E_Comm_data.product` c on a.product_2 = c.PRODUCT_ID
order by a.cnt_bought_together DESC
```

Output: -



```
-- 6. Find products(product table : SUB_COMMODITY_DESC) which are most frequently bought together and the count of each combination bought together.
do not print a combination twice ( A-B / B-A)

with cte as (
select a.PRODUCT_ID as product_1, b.PRODUCT_ID as product_2, count(*) as cnt_bought_together
from `E_Comm_data.transaction` a join `E_Comm_data.transaction` b on a.BASKET_ID = b.BASKET_ID and a.PRODUCT_ID < b.PRODUCT_ID
group by a.PRODUCT_ID , b.PRODUCT_ID
)
select a.product_1,b.SUB_COMMODITY_DESC as product_1_SUB_COMMODITY_DESC ,
a.product_2 ,c.SUB_COMMODITY_DESC as product_2_SUB_COMMODITY_DESC,
a.cnt_bought_together
from cte a join `E_Comm_data.product` b on a.product_1 = b.PRODUCT_ID join `E_Comm_data.product` c on a.product_2 = c.PRODUCT_ID
order by a.cnt_bought_together DESC
```

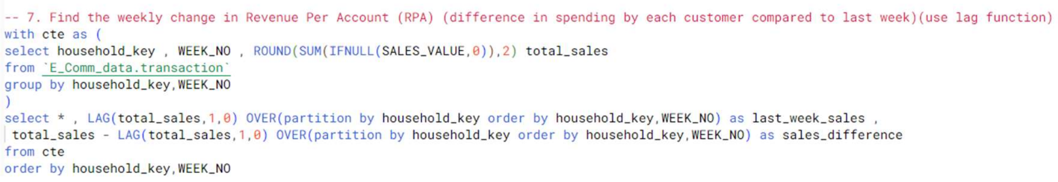


Row	product_1	product_1_SUB_COMMODITY_DES	product_2	product_2_SUB_COMMODITY_DES	cnt_bought_together
1	1029743	FLUID MILK WHITE ONLY	1082185	BANANAS	848
2	995242	FLUID MILK WHITE ONLY	1082185	BANANAS	728
3	981760	EGGS - X-LARGE	1082185	BANANAS	625
4	1082185	BANANAS	1127831	STRAWBERRIES	611
5	1082185	BANANAS	1106523	FLUID MILK WHITE ONLY	519
6	961554	CARROTS MINI PEELED	1082185	BANANAS	473
7	951590	MAINSTREAM WHITE BREAD	1082185	BANANAS	458
8	1070820	FLUID MILK WHITE ONLY	1082185	BANANAS	430
9	1082185	BANANAS	1126899	FLUID MILK WHITE ONLY	427
10	826249	HAMBURGER BUNS	1098066	HOT DOG BUNS	421
11	854852	TOMATOES HOTHOUSE ON TH...	1082185	BANANAS	420

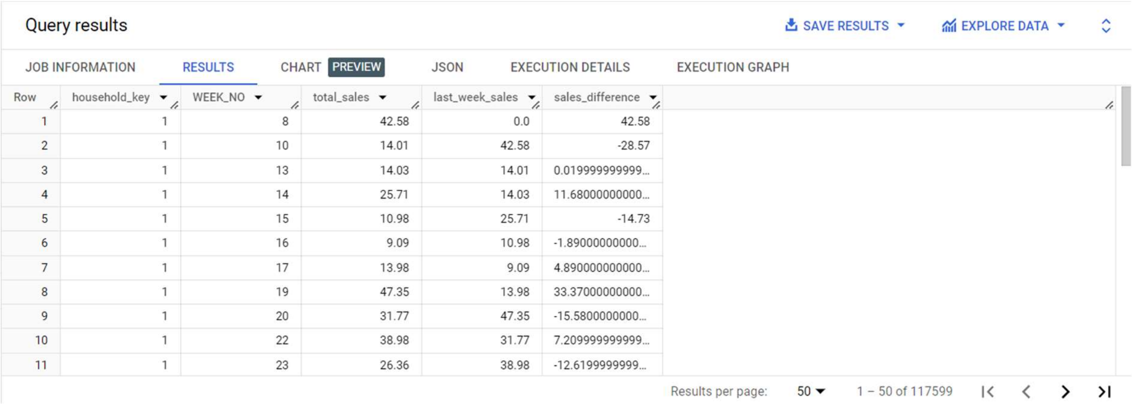
-----------------------------------X---X---------------------------------------

```
-- 7. Find the weekly change in Revenue Per Account (RPA) (difference in spending
by each customer compared to last week)(use lag function)

with cte as (
select household_key , WEEK_NO , ROUND(SUM(IFNULL(SALES_VALUE,0)),2) total_sales
from `E_Comm_data.transaction`
group by household_key,WEEK_NO
)
select * , LAG(total_sales,1,0) OVER(partition by household_key order by
household_key,WEEK_NO) as last_week_sales ,
 total_sales - LAG(total_sales,1,0) OVER(partition by household_key order by
household_key,WEEK_NO) as sales_difference
from cte
order by household_key,WEEK_NO
```

Output: -



```
-- 7. Find the weekly change in Revenue Per Account (RPA) (difference in spending by each customer compared to last week)(use lag function)
with cte as (
select household_key , WEEK_NO , ROUND(SUM(IFNULL(SALES_VALUE,0)),2) total_sales
from `E_Comm_data.transaction`
group by household_key,WEEK_NO
)
select * , LAG(total_sales,1,0) OVER(partition by household_key order by household_key,WEEK_NO) as last_week_sales ,
 total_sales - LAG(total_sales,1,0) OVER(partition by household_key order by household_key,WEEK_NO) as sales_difference
from cte
order by household_key,WEEK_NO
```

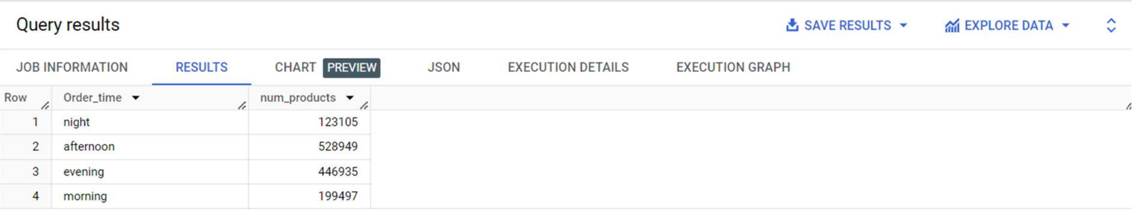


Row	household_key	WEEK_NO	total_sales	last_week_sales	sales_difference
1	1	8	42.58	0.0	42.58
2	1	10	14.01	42.58	-28.57
3	1	13	14.03	14.01	0.019999999999...
4	1	14	25.71	14.03	11.68000000000...
5	1	15	10.98	25.71	-14.73
6	1	16	9.09	10.98	-1.89000000000...
7	1	17	13.98	9.09	4.890000000000...
8	1	19	47.35	13.98	33.37000000000...
9	1	20	31.77	47.35	-15.5800000000...
10	1	22	38.98	31.77	7.209999999999...
11	1	23	26.36	38.98	-12.6199999999...

-----------------------------------X--X---------------------------------------

```
--8.  Find the number of orders that have been placed on morning ,afternoon ,
evening and night transaction time (morning 5am to 12pm ,afternoon 12 to 5pm ,
evening 5pm to 9pm and night 9pm to 4am )

with cte as (
select BASKET_ID ,
CASE WHEN TRANS_TIME >= 0500 and TRANS_TIME < 1200 THEN 'morning'
WHEN TRANS_TIME >= 1200 and TRANS_TIME <= 1700 THEN 'afternoon'
WHEN TRANS_TIME > 1700 and TRANS_TIME <= 2100 THEN 'evening'
ELSE 'night'
END as Order_Time
from `E_Comm_data.transaction` )

select Order_time , Count(*) num_products
from cte
group by Order_time
```

Output: -

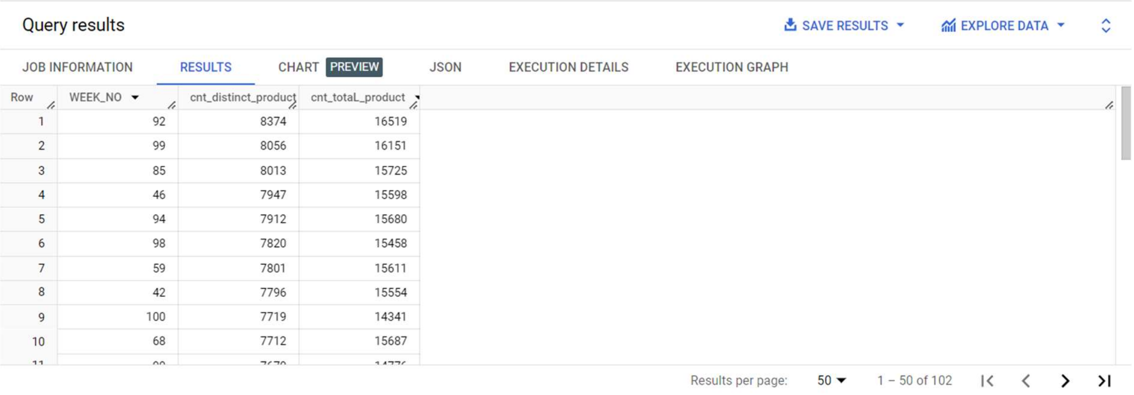


Row	Order_time	num_products
1	night	123105
2	afternoon	528949
3	evening	446935
4	morning	199497

-----------------------------------X--X---------------------------------------

```
-- 9. Week number where most distinct products sold

select WEEK_NO , COUNT(DISTINCT PRODUCT_ID)  cnt_distinct_product ,
COUNT(PRODUCT_ID)  cnt_totaL_product
from `E_Comm_data.transaction`
group by WEEK_NO
order by cnt_distinct_product DESC
```

Output: -



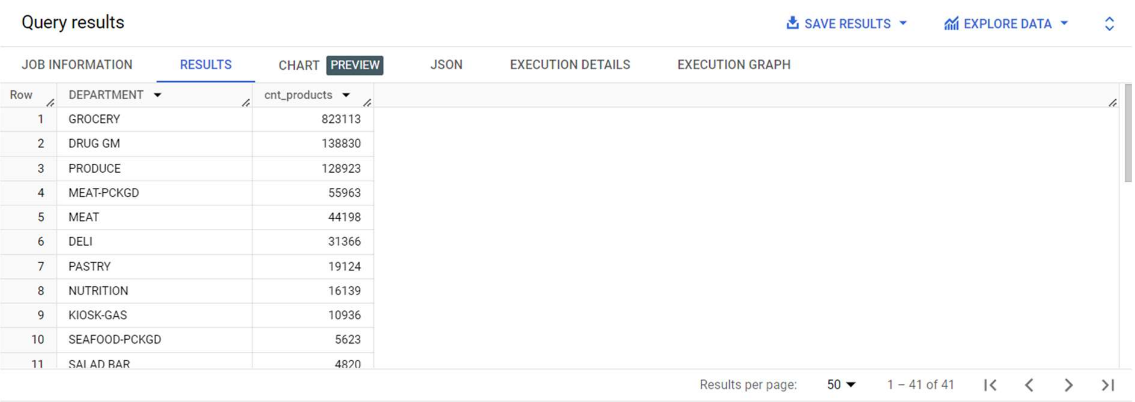
Row	WEEK_NO	cnt_distinct_product	cnt_totaL_product
1	92	8374	16519
2	99	8056	16151
3	85	8013	15725
4	46	7947	15598
5	94	7912	15680
6	98	7820	15458
7	59	7801	15611
8	42	7796	15554
9	100	7719	14341
10	68	7712	15687
11	[illegible]	[illegible]	[illegible]

Results per page: 50 1 – 50 of 102

-----------------------------------X--X---------------------------------------

```
-- 10. Count of Product sold in each department

select DEPARTMENT , Count(a.PRODUCT_ID) cnt_products
from `E_Comm_data.transaction` a join `E_Comm_data.product`  b on a.PRODUCT_ID =
b.PRODUCT_ID
group by DEPARTMENT
order by cnt_products DESC
```

Output: -



Row	DEPARTMENT	cnt_products
1	GROCERY	823113
2	DRUG GM	138830
3	PRODUCE	128923
4	MEAT-PCKGD	55963
5	MEAT	44198
6	DELI	31366
7	PASTRY	19124
8	NUTRITION	16139
9	KIOSK-GAS	10936
10	SEAFOOD-PCKGD	5623
11	SALAD BAR	4820

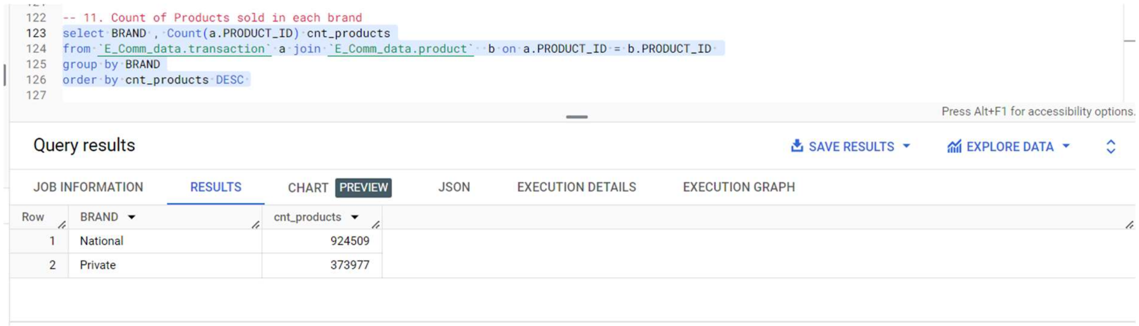
Results per page: 50 1 – 41 of 41

-----------------------------------X--X---------------------------------------

```
-- 11. Count of Products sold in each brand

select BRAND , Count(a.PRODUCT_ID) cnt_products
```

```
from `E_Comm_data.transaction` a join `E_Comm_data.product`  b on a.PRODUCT_ID =
b.PRODUCT_ID
group by BRAND
order by cnt_products DESC
```

Output: -



```
-- 11. Count of Products sold in each brand
select BRAND , Count(a.PRODUCT_ID) cnt_products
from `E_Comm_data.transaction` a join `E_Comm_data.product`  b on a.PRODUCT_ID = b.PRODUCT_ID
group by BRAND
order by cnt_products DESC
```

Press Alt+F1 for accessibility options.

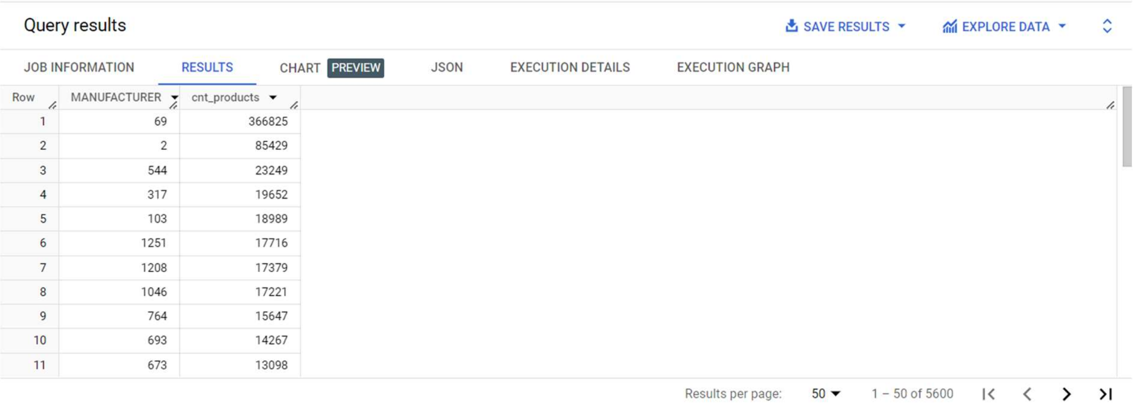
Query results

Row	BRAND	cnt_products
1	National	924509
2	Private	373977

-----------------------------------X---X---------------------------------------

```
-- 12. Count of Products sold by each manufacturer

select MANUFACTURER , Count(a.PRODUCT_ID) cnt_products
from `E_Comm_data.transaction` a join `E_Comm_data.product`  b on a.PRODUCT_ID =
b.PRODUCT_ID
group by MANUFACTURER
order by cnt_products DESC
```

Output: -



Query results

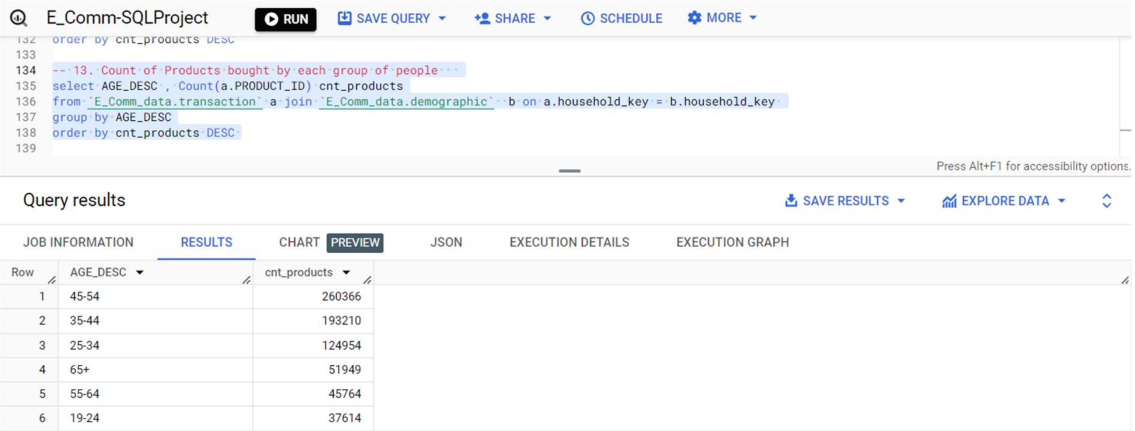
Row	MANUFACTURER	cnt_products
1	69	366825
2	2	85429
3	544	23249
4	317	19652
5	103	18989
6	1251	17716
7	1208	17379
8	1046	17221
9	764	15647
10	693	14267
11	673	13098

Results per page: 50 1 – 50 of 5600

-----------------------------------X---X---------------------------------------

```
-- 13. Count of Products bought by each age group of people

select AGE_DESC , Count(a.PRODUCT_ID) cnt_products
from `E_Comm_data.transaction` a join `E_Comm_data.demographic`  b on
a.household_key = b.household_key
group by AGE_DESC
order by cnt_products DESC
```

Output: -



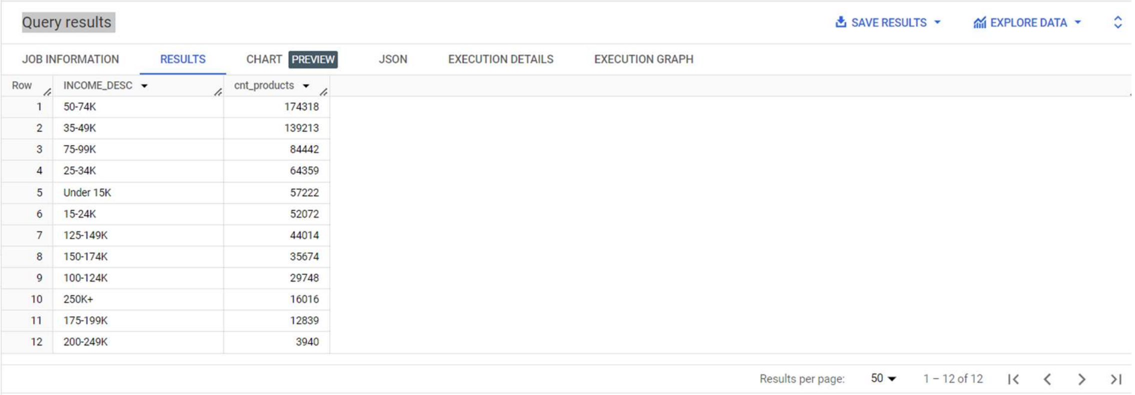
```
order by cnt_products DESC

-- 13. Count of Products bought by each group of people
select AGE_DESC , Count(a.PRODUCT_ID) cnt_products
from `E_Comm_data.transaction` a join `E_Comm_data.demographic`  b on a.household_key = b.household_key
group by AGE_DESC
order by cnt_products DESC
```

Row	AGE_DESC	cnt_products
1	45-54	260366
2	35-44	193210
3	25-34	124954
4	65+	51949
5	55-64	45764
6	19-24	37614

-----------------------------------X---X---------------------------------------

```
-- 14. Count of Products bought by customer belonging to each income bracket

select INCOME_DESC , Count(a.PRODUCT_ID) cnt_products
from `E_Comm_data.transaction` a join `E_Comm_data.demographic`  b on
a.household_key = b.household_key
group by INCOME_DESC
order by cnt_products DESC
```

Output: -

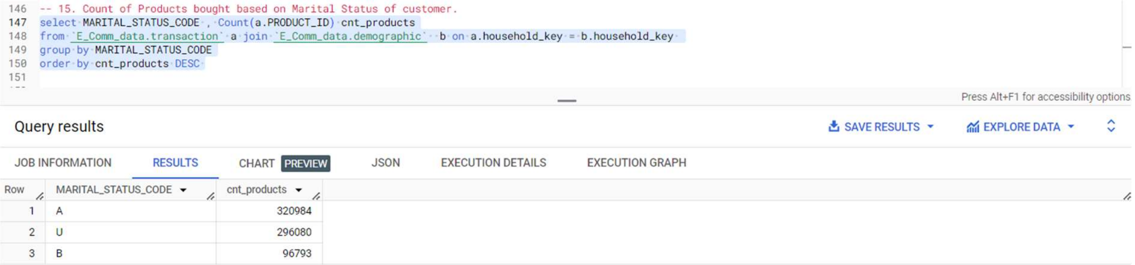


Row	INCOME_DESC	cnt_products
1	50-74K	174318
2	35-49K	139213
3	75-99K	84442
4	25-34K	64359
5	Under 15K	57222
6	15-24K	52072
7	125-149K	44014
8	150-174K	35674
9	100-124K	29748
10	250K+	16016
11	175-199K	12839
12	200-249K	3940

-----------------------------------X---X---------------------------------------

```
-- 15. Count of Products bought based on Marital Status of customer.

select MARITAL_STATUS_CODE , Count(a.PRODUCT_ID) cnt_products
from `E_Comm_data.transaction` a join `E_Comm_data.demographic`  b on
a.household_key = b.household_key
group by MARITAL_STATUS_CODE
order by cnt_products DESC
```

Output: -



```
-- 15. Count of Products bought based on Marital Status of customer.
select MARITAL_STATUS_CODE , Count(a.PRODUCT_ID) cnt_products
from `E_Comm_data.transaction` a join `E_Comm_data.demographic`  b on a.household_key = b.household_key
group by MARITAL_STATUS_CODE
order by cnt_products DESC
```

Row	MARITAL_STATUS_CODE	cnt_products
1	A	320984
2	U	296080
3	B	96793

----------------------------------X--X--------------------------------------

INSIGHTS AND RECOMMENDATIONS

- From Query 1 and 2 we can say that we have least number of orders with order value Medium as compared to Order with value Small and Large .
- Query 2 shows top 3 stores with highest foot traffic for each week
- From Query 4 we can conclude that household_key 2042 has the highest average amount spend per visit.
- From Query 5 we can conclude that household_key 1609 has done most total spending (>10K) on products followed by household_key 2322 , 1453 and 1430.
- From Query 6 - Top 5 Products which are most frequently bought together are :-
  1. Product ID 1029743 (FLUID MILK WHITE ONLY) and Product ID 1082185 (BANANAS)
  2. Product ID 995242 (FLUID MILK WHITE ONLY) and Product ID 1082185 (BANANAS)
  3. Product ID 981760 (EGGS - X-LARGE) and Product ID 1082185 (BANANAS)
  4. Product ID 1082185 (BANANAS) and Product ID 1127831 (STRAWBERRIES)
  5. Product ID 1082185 (BANANAS) and Product ID 1106523 (FLUID MILK WHITE ONLY)
- From Query 8 we can conclude that most of the products are bought during afternoon and evening time. Least orders are bought during night and morning.
- From Query 9 result we can conclude that for initial weeks (1 to 10) count of products sold is less as compared to other weeks. Count of products sold is highest in week 92 followed by week 99 and 85.
- From Query 10 we can conclude that top 3 Departments with highest count of products sold are :-
  1. Grocery
  2. Drug GM
  3. Produce
- Total count of products sold of National Brand is 3 times of that of Private Brand.
- From Query 12 we can conclude that majority of products are from top 2 manufacturers :69 and 2.
- Customer belonging to Age Groups 45-54 , 35-44 ,25-34 are the ones who have highest count of products bought from E-commerce platform.
- From Query 14 we can conclude that Customer belonging to these Income Bracket 50-74K having bought largest number of products followed by customer belonging to Income Bracket – 35-49K , 75-99K ,25-34K.

- From above Observations we can say that recommend that more range of products in other departments (other than top 3) should be introduced in order to increase diversity of products and majority products sold should not be dependent of a few major departments.
- Early Sales Event can be done to increase the product sold in initial few weeks (Week Number 1 to 10)
- Some more Products should be introduced which can attract High Salary Bracket Customer (>100K). As most of the product bought are by people belonging to Low or Medium Salary Bracket.
- Age group 19-24 customers have bought least products this can be due to :-
  1. Not having latest products which can attract younger crowd.
  2. Most of the products being not affordable for 19-24 age group customers.
  3. E-Commerce platform not offering best prices as compared to competitors who has capture most of the younger crowd.

  This age group crowd can be attracted by giving student discount or cashbacks on products.

- Order count during Morning and Night time can be increase by offering limited time discount during those time. Organizing big sale events with start time as Morning or Night can be another way to attract crowd to buy limited products (first come first serve basis).