Walmart Training session

SQL Lab Documentation

Name : Swethaa R

Reg No : 71772118148

Branch : B.Tech IT


```
show databases;
use retail_db;
select * from products;
select * from orders;
select * from users;
select * from reviews;
show tables;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: | Fetch rows:

#	id	created_at	category	ean	price	quantity	rating	title	vendor
1	1	2017-07-19 14:14:56	Gizmo	1018947080336	29.4633	5000	4.6	Rustic Paper Wallet	Swaniawski, Casper and Hilll
2	2	2019-04-11 03:19:35	Doohickey	7663515285824	70.0799	5000	0	Small Marble Shoes	Balistreri-Ankunding
3	3	2018-09-08 16:33:20	Doohickey	4966277046676	35.3887	5000	4	Synergistic Granite Chair	Murray, Watsica and Wunsch
4	4	2018-03-05 21:23:09	Doohickey	4134502155718	73.9918	5000	3	Enormous Aluminum Shirt	Regan Bradtke and Sons
5	5	2016-10-02 20:17:39	Gadget	5499736705597	82.7451	5000	4	Enormous Marble Wallet	Price, Schultz and Daniel
6	6	2017-03-29 00:13:40	Doohickey	2293343551454	64.9575	5000	3.8	Small Marble Hat	Nolan-Wolff
7	7	2017-06-02 21:37:28	Doohickey	157967025871	98.8193	5000	4.3	Aerodynamic Linen Coat	Little-Pagac
8	8	2018-04-30 09:33:53	Doohickey	1078766578568	65.8922	5000	4.1	Enormous Steel Watch	Senger-Stamm
9	9	2019-02-07 02:56:25	Widget	7217466997444	58.3131	5000	4.2	Practical Bronze Computer	Keely Stehr Group
10	10	2017-01-09 04:21:20	Gizmo	1807963902339	31.7862	5000	4.3	Mediocre Wooden able	Larson, Pfeffer and Klocko
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Problem Statement 1:

Question 1: Identify the top 5 products with the highest sales revenue in the year 2015.

Solution:

1. Using Subquery:

```
SELECT category, title, id
 FROM( SELECT category, id, title, dense_rank() OVER
    (PARTITION BY category ORDER BY price DESC) AS count FROM
    products ) ranked
 WHERE count <= 5;
```

```
Select category,title,id from (Select category,id,title,
dense_rank() over (partition by category order by price desc ) as count
from products) ranked where count<= 5;
Select name from users where id IN (
Select user_id from (Select user_id,count(id) from orders where month(created_at)=11 and year(created_at)=2018 group by user_id
order by count(id) desc limit 10) as top_users );
```

esult Grid | Filter Rows: | Export: | Wrap Cell Content:

#	category	title	id
1	Doohickey	Aerodynamic Linen Coat	7
2	Doohickey	Enormous Cotton Pants	141
3	Doohickey	Sleek Wool Wallet	45
4	Doohickey	Fantastic Leather Watch	175
5	Doohickey	Incredible Plastic Watch	138
6	Gadget	Incredible Aluminum Knife	64
7	Gadget	Durable Cotton Bench	101
8	Gadget	Sleek Plastic Shoes	148
9	Gadget	Ergonomic Silk Coat	11
10	Gadget	Incredible Silk Shoes	49
11	Gizmo	Small Marble Knife	129
12	Gizmo	Aerodynamic Leather o...	158
13	Gizmo	Enormous Granite Wallet	63

Result 4

Explanation:

The first solution ranks products based on price within each category and selects the top 5.

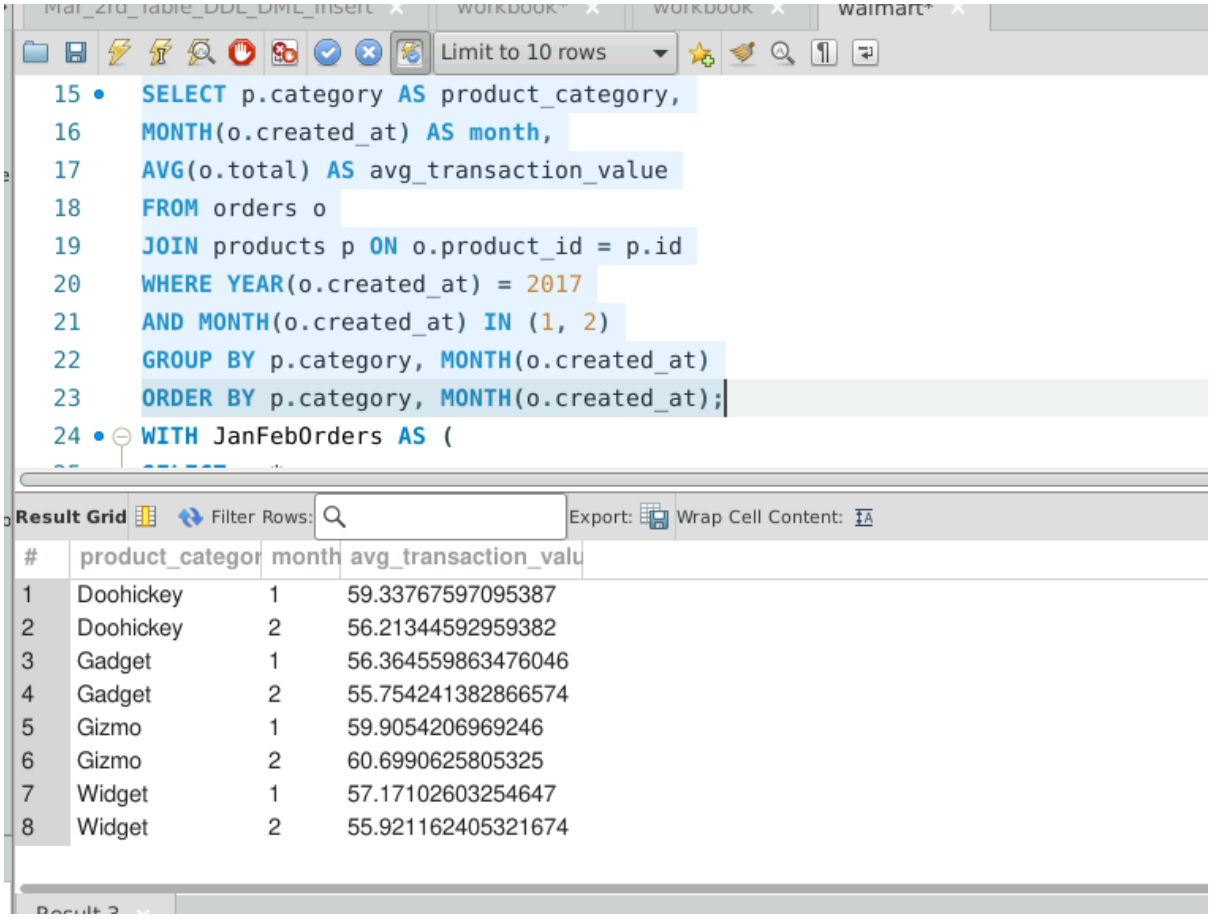
Question 2:

Determine and compare the average transaction value (ATV) for each product category MoM (Jan/Feb) in the year 2017.

Solution:

1. Using simple select and join functions:

```
SELECT p.category AS product_category,
       MONTH(o.created_at) AS month,
       AVG(o.total) AS avg_transaction_value
FROM orders o
JOIN products p ON o.product_id = p.id
WHERE YEAR(o.created_at) = 2017
    AND MONTH(o.created_at) IN (1, 2)
GROUP BY p.category, MONTH(o.created_at)
ORDER BY p.category, MONTH(o.created_at);
```



2. Using Common Table Expression (CTE):

```
WITH JanFebOrders AS (
    SELECT o.*
    FROM orders o
    WHERE YEAR(o.created_at) = 2017
        AND MONTH(o.created_at) IN (1, 2)
)
SELECT p.category AS product_category,
       MONTH(o.created_at) AS month,
       AVG(o.total) AS avg_transaction_value
FROM JanFebOrders o
JOIN products p ON o.product_id = p.id
GROUP BY p.category, MONTH(o.created_at)
```

ORDER BY p.category, MONTH(o.created_at);

```
WITH JanFebOrders AS (
SELECT o.*
FROM orders o
WHERE YEAR(o.created_at) = 2017
AND MONTH(o.created_at) IN (1, 2)
)
SELECT p.category AS product_category,
MONTH(o.created_at) AS month,
AVG(o.total) AS avg_transaction_value
```

Result Grid Filter Rows: Export: Wrap Cell Content:

#	product_categor	month	avg_transaction_valu
1	Doohickey	1	59.33767597095387
2	Doohickey	2	56.21344592959382
3	Gadget	1	56.364559863476046
4	Gadget	2	55.754241382866574
5	Gizmo	1	59.9054206969246
6	Gizmo	2	60.6990625805325
7	Widget	1	57.17102603254647
8	Widget	2	55.921162405321674

Result 3

Explanation:

The first solution directly calculates the average transaction value for each product category in January and February 2017. The second solution creates a temporary table to filter orders from January and February 2017 and then calculates the average transaction value for each product category.

Question 3:

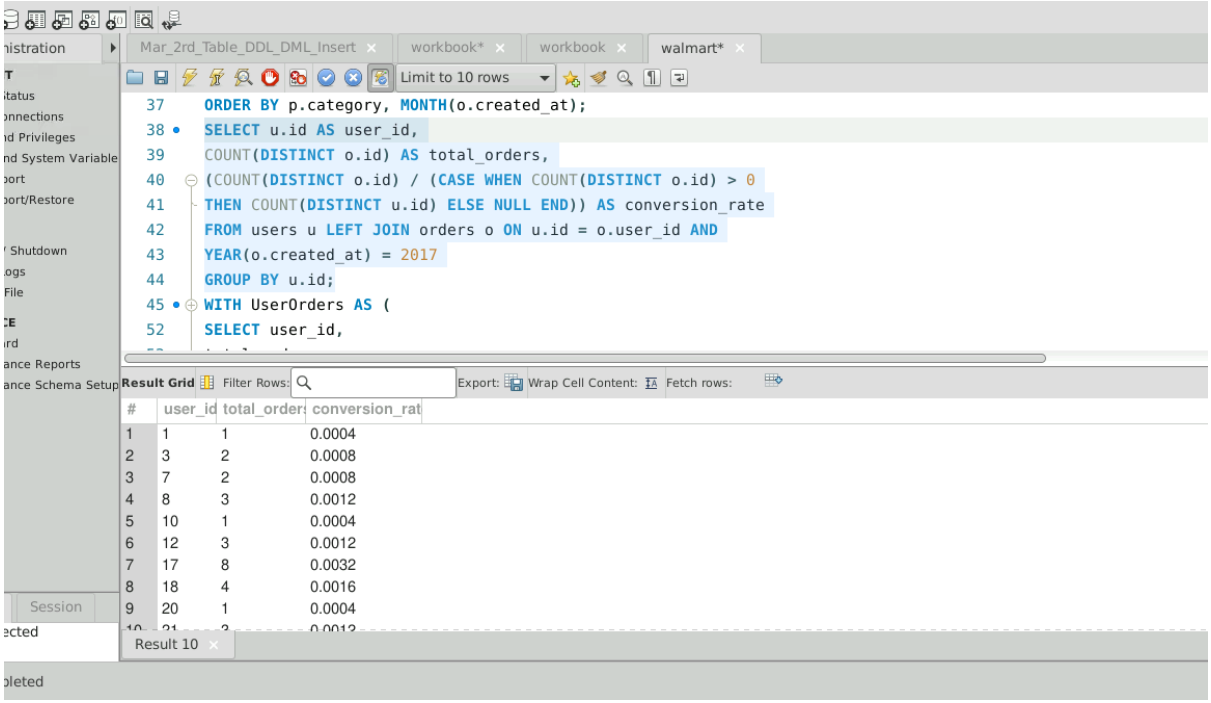
Calculate the conversion rate for each user, considering only users who have placed at least one order (use year = 2017).

Solution:

1. Using Subquery:

```
SELECT u.id AS user_id,
    COUNT(DISTINCT o.id) AS total_orders,
    (COUNT(DISTINCT o.id) / (CASE WHEN COUNT(DISTINCT o.id) > 0
THEN COUNT(DISTINCT u.id) ELSE NULL END)) AS conversion_rate
FROM users u  RIGHT JOIN orders o ON u.id = o.user_id AND
 YEAR(o.created_at) = 2017
```

GROUP BY u.id;



2. Using CTE:

WITH UserOrders AS (
SELECT u.id AS user_id, COUNT(DISTINCT o.id) AS total_orders

FROM users u LEFT JOIN orders o ON u.id = o.user_id AND YEAR(o.created_at) =2017 GROUP BY u.id)

SELECT user_id, total_orders, (total_orders / (CASE WHEN total_orders > 0 THEN (SELECT COUNT(DISTINCT id) FROM users) ELSE NULL END)) AS conversion_rate

FROM UserOrders WHERE total_orders>0;

```
YEAR(o.created_at) = 2017
GROUP BY u.id;
WITH UserOrders AS (
SELECT user_id,
total_orders,
(total_orders / (CASE WHEN total_orders > 0 THEN (SELECT
COUNT(DISTINCT id) FROM users) ELSE NULL END)) AS conversion_rate
FROM
UserOrders WHERE total_orders>0;
SELECT u.id AS user_id,
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

#	user_id	total_orders	conversion_rat
11	22	2	0.0008
12	32	1	0.0004
13	35	1	0.0004
14	36	1	0.0004
15	37	8	0.0032
16	38	2	0.0008
17	39	1	0.0004
18	40	8	0.0032
19	42	3	0.0012

Result 7

Explanation:

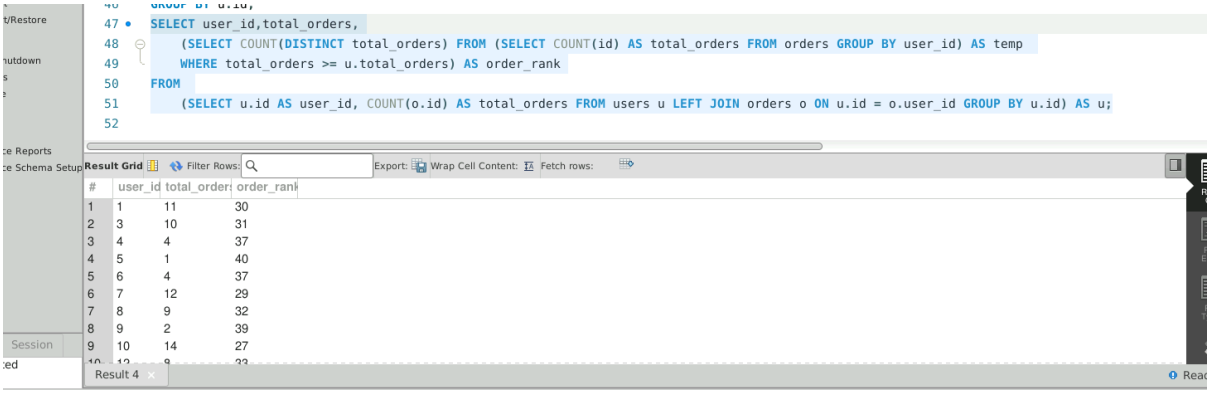
The first solution calculates the total orders and conversion rate for each user by joining the users and orders tables. It ensures the conversion rate is only calculated for users who have placed at least one order in 2017. The second solution creates a CTE to calculate the total orders for each user in 2017 and then calculates the conversion rate for each user.

Question 4:

Rank users based on their total order count, with RANK SHARING allowed.

Solution:

```
SELECT user_id, total_orders,

   (SELECT COUNT (DISTINCT total_orders)

   FROM (SELECT COUNT(id) AS total_orders

   FROM orders GROUP BY user_id) AS temp

   WHERE total_orders >= u.total_orders) AS order_rank

FROM  (SELECT u.id AS user_id, COUNT(o.id) AS total_orders

    FROM users u LEFT JOIN orders o ON u.id = o.user_id GROUP BY u.id) AS u;
```



```
SELECT user_id,total_orders,
    (SELECT COUNT(DISTINCT total_orders) FROM (SELECT COUNT(id) AS total_orders FROM orders GROUP BY user_id) AS temp
    WHERE total_orders >= u.total_orders) AS order_rank
FROM
    (SELECT u.id AS user_id, COUNT(o.id) AS total_orders FROM users u LEFT JOIN orders o ON u.id = o.user_id GROUP BY u.id) AS u;
```

Result Grid

#	user_id	total_orders	order_rank
1	1	11	30
2	3	10	31
3	4	4	37
4	5	1	40
5	6	4	37
6	7	12	29
7	8	9	32
8	9	2	39
9	10	14	27

It calculates the total order count for each user using a subquery and then ranks them based on their order count using another subquery. It ensures rank sharing by considering users with the same total order count.

Problem Statement 2:

Question 1:

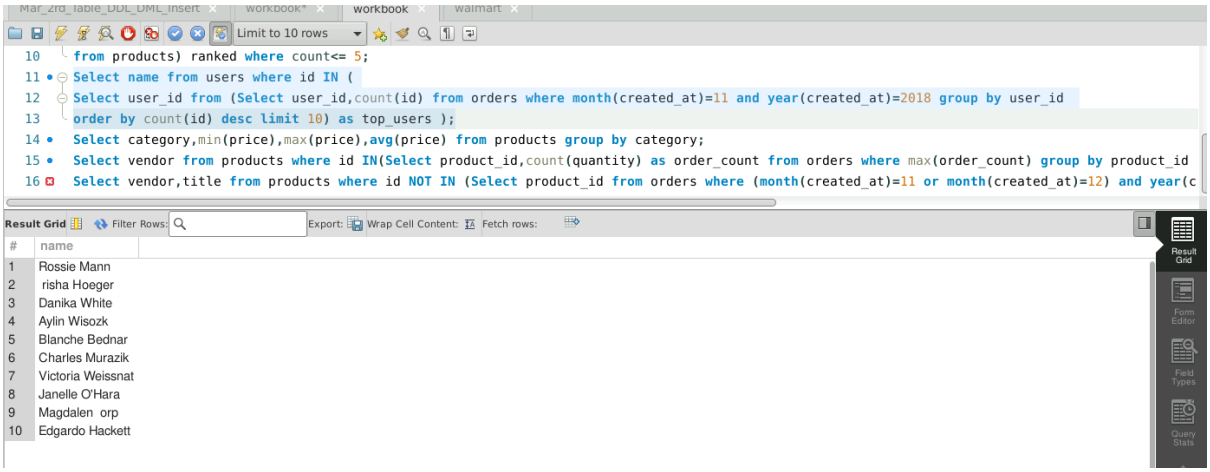
Find the top 10 active customers (customers who have made the most number of orders) in the month of November 2018

Solution:

SELECT name FROM users WHERE id IN (

SELECT user_id FROM (SELECT user_id,count(id) FROM orders WHERE month(created_at)=11 AND year(created_at)=2018 GROUP BY user_id

ORDER BY count(id) DESC limit 10) AS top_users);



```
from products) ranked where count<= 5;
Select name from users where id IN (
Select user_id from (Select user_id,count(id) from orders where month(created_at)=11 and year(created_at)=2018 group by user_id
order by count(id) desc limit 10) as top_users );
Select category,min(price),max(price),avg(price) from products group by category;
Select vendor from products where id IN(Select product_id,count(quantity) as order_count from orders where max(order_count) group by product_id
Select vendor,title from products where id NOT IN (Select product_id from orders where (month(created_at)=11 or month(created_at)=12) and year(c
```

Result Grid

#	name
1	Rossie Mann
2	risha Hoeger
3	Danika White
4	Aylin Wisozk
5	Blanche Bednar
6	Charles Murazik
7	Victoria Weissnat
8	Janelle O'Hara
9	Magdalen orp
10	Edgardo Hackett

Question 2: Calculate the percentage of growth in the Organic sales made by OrderKart in the year '2018' and '2019' in each product category (Marketing channel is available in source column from users table)

WITH OrganicSales AS (SELECT p.category,u.source,YEAR(o.created_at) AS year,SUM(o.subtotal) AS total_sales

FROM orders o JOIN products p ON o.product_id = p.id

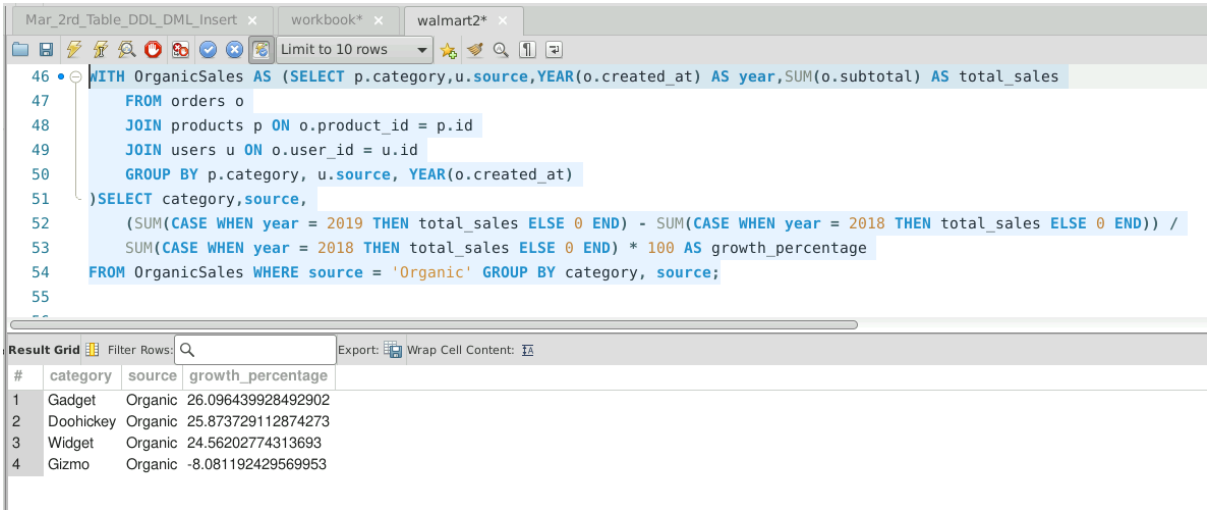
JOIN users u ON o.user_id = u.id

GROUP BY p.category, u.source, YEAR(o.created_at))SELECT category,source,

(SUM(CASE WHEN year = 2019 THEN total_sales ELSE 0 END) - SUM(CASE WHEN year = 2018 THEN total_sales ELSE 0 END)) /

SUM(CASE WHEN year = 2018 THEN total_sales ELSE 0 END) * 100 AS growth_percentage

FROM OrganicSales WHERE source = 'Organic' GROUP BY category, source;



Question 3: Identify the top 3 Marketing Vehicles contributing to the highest number of orders since '2017'

SELECT source,

COUNT(*) AS total_orders

FROM orders o

INNER JOIN users u ON o.user_id = u.id

WHERE YEAR(o.created_at) >= 2017

GROUP BY source

ORDER BY total_orders DESC

LIMIT 3;

Question 4: Find out the customer RPR, repeat purchase rate. (percent of customers who made more than 1 order).

WITH customer_orders AS (SELECT user_id,

COUNT(*) AS order_count FROM orders GROUP BY user_id)

SELECT 100.0 * SUM(CASE WHEN order_count > 1 THEN 1 ELSE 0 END) / COUNT(*) AS repeat_purchase_rate FROM customer_orders;

Problem Statement 3:

Question 1: Find the top 5 product names in descending order of prices in each category (Sort by title for items with same price)

SELECT category, title, price

FROM products p1

WHERE (SELECT COUNT(*)

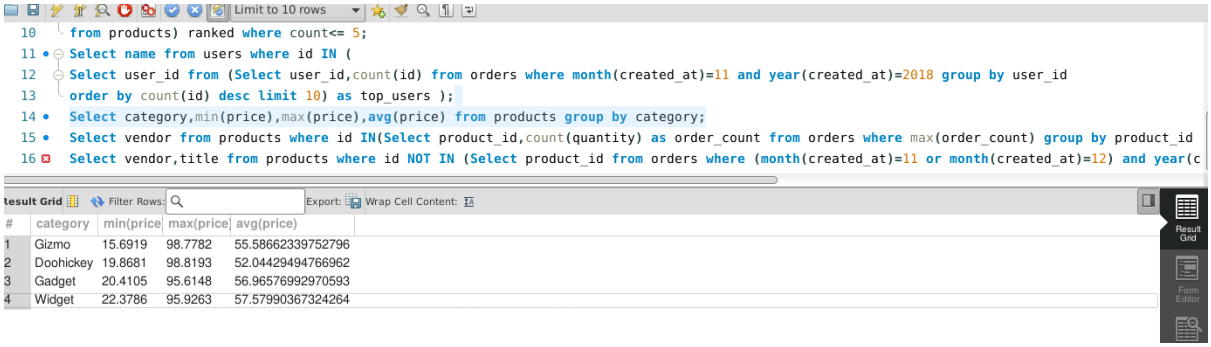
    FROM products p2

    WHERE p2.category = p1.category AND (p2.price > p1.price OR (p2.price = p1.price AND p2.title <= p1.title))) <= 5

ORDER BY category, price DESC, title ASC;

Question 2: Find the minimum, maximum and average price of products by category

SELECT category, min(price), max(price), avg(price) from products GROUP BY category;



Question 3: Find the top seller names for all the ordered products (in terms of order count)

SELECT p.vendor AS seller_name, COUNT(*) AS order_count

FROM products p JOIN orders o ON p.id = o.product_id

GROUP BY p.vendor ORDER BY order_count DESC;

Question 4: Find minimum, maximum and average rating for all ordered products .

SELECT MIN(p.rating) AS min_rating, MAX(p.rating) AS max_rating, AVG(p.rating) AS avg_rating

FROM products p JOIN orders o ON p.id = o.product_id;

Question 5: Identify sellers and their products which were never sold in the last two months of 2019

```
SELECT p.vendor AS seller_name, p.title AS product_name

FROM products p

WHERE p.id NOT IN (

    SELECT DISTINCT o.product_id

    FROM orders o WHERE YEAR(o.created_at) = 2019 AND MONTH(o.created_at) IN (11,
12));
```

Problem Statement 4:

Question 1: Xmart is not having the expected growth this year. The Ad-tech team is planning to analyse the marketing channel on the year which has the highest number of customer signup.

```
SELECT YEAR(created_at) AS signup_year,

  COUNT(*) AS customer_signups

FROM users

GROUP BY YEAR(created_at)

ORDER BY customer_signups DESC

LIMIT 1;
```

Question 2: Xmart is planning to work on a marketing advertisement to increase the company's sales. Help Xmart to find the Age group that has largest customer base to attract. (<18/18-24/25-44/45-55/55+)

```
SELECT CASE
      WHEN age < 18 THEN '<18'
      WHEN age BETWEEN 18 AND 24 THEN '18-24'
      WHEN age BETWEEN 25 AND 44 THEN '25-44'
      WHEN age BETWEEN 45 AND 55 THEN '45-55'
      ELSE '55+'
   END AS age_group,
   COUNT(*) AS customer_count
FROM users
GROUP BY age_group
ORDER BY customer_count DESC
LIMIT 1;
```

Question 3: Xmart wants to send discount coupons to customer having the highest order total amount. Help Xmart to find the customer details.

```
SELECT user_id, SUM(amount) AS total_order_amount
FROM orders GROUP BY user_id
ORDER BY total_order_amount DESC
```

```
LIMIT 1;
```

Question 4: Vendor `Fisher-Kemmer` wants to send promotion mail to the customer who made the highest purchase on his items. Help Xmart to find the customer Email Id details.

```
SELECT email FROM users WHERE id = (
      SELECT  user_id FROM orders GROUP BY user_id
      ORDER BY SUM(amount) DESC
      LIMIT 1
   );
```