Creating database

```
DROP DATABASE if exists Goldusers;
CREATE DATABASE Goldusers;
```

Utilizing the new database to create tables and insert values

```
USE Goldusers;

DROP TABLE if exists goldusers_signup;
CREATE TABLE goldusers_signup(user_id integer, gold_signup_date date);

INSERT INTO goldusers_signup(user_id, gold_signup_date)
VALUES(1,'2017-09-22'),
(3,'2017-04-21');

DROP TABLE if exists users;
CREATE TABLE users(user_id integer, signup_date date);

INSERT INTO users(user_id,signup_date)
VALUES(1,'2014-09-02'),
(2,'2015-01-15'),
(3,'2014-04-11');

DROP TABLE if exists sales;
CREATE TABLE sales(user_id integer,created_date date,product_id integer);

INSERT INTO sales(user_id,created_date,product_id)
VALUES(1,'2017-04-19',2),
(3,'2019-12-18',1),
(2,'2020-07-20',3),
(1,'2019-10-23',2),
```

(1,'2018-03-19',3),

(3,'2016-12-20',2),

(1,'2016-11-09',1),

(1,'2016-05-20',3),

(2,'2017-09-24',1),

(1,'2017-03-11',2),

(1,'2016-03-11',1),

(3,'2016-11-10',1),

(3,'2017-12-07',2),

(3,'2016-12-15',2),

(2,'2017-11-08',2),

(2,'2018-09-10',3);

```
DROP TABLE if exists product;
CREATE TABLE product(product_id integer,product_name text,price integer);

INSERT INTO product(product_id,product_name,price)
VALUES(1,'P1',980),
(2,'P2',870),
(3,'P3',330);
```

Viewing all the newly created tables

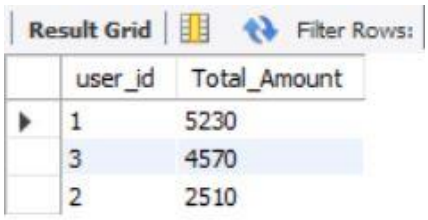
```
SELECT*FROM goldusers_signup;
SELECT*FROM users;
SELECT*FROM sales;
SELECT*FROM product;
```

1) What is the total amount each customer has spent on zomato?

Query:

```
SELECT s.user_id, SUM(p.price) as Total_Amount
FROM sales as s
INNER JOIN product as p
ON s.product_id=p.product_id
GROUP BY s.user_id;
```

Output:



Result Grid | Filter Rows:

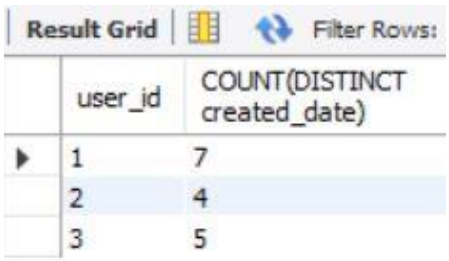
user_id	Total_Amount
1	5230
3	4570
2	2510

2) How many days have each customer visited zomato?

Query:

```
SELECT user_id, COUNT(DISTINCT created_date)
FROM sales
GROUP BY user_id;
```

Output:



Result Grid | Filter Rows:

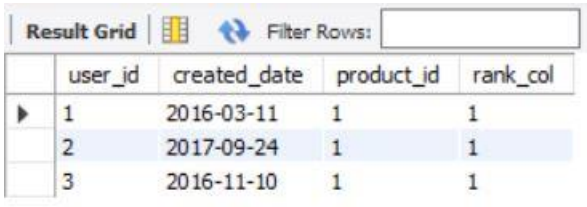
user_id	COUNT(DISTINCT created_date)
1	7
2	4
3	5

3) What was the first product purchased by each customer?

Query:

```
SELECT *
FROM (SELECT *,
RANK() OVER(PARTITION BY user_id ORDER BY created_date) as rank_col
FROM sales) as r
WHERE r.rank_col=1;
```

Output:



Result Grid | Filter Rows:

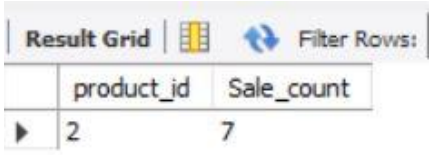
user_id	created_date	product_id	rank_col
1	2016-03-11	1	1
2	2017-09-24	1	1
3	2016-11-10	1	1

4) What is the most purchased item on menu & how many times was it purchased by all customers?

Query:

```
SELECT product_id,COUNT(product_id) as Sale_count
FROM sales
GROUP BY product_id
ORDER BY Sale_count DESC
LIMIT 1;
```

Output:



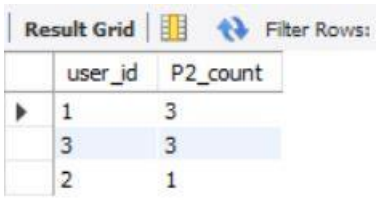
Result Grid | Filter Rows:

product_id	Sale_count
2	7

Query:

```
SELECT user_id, COUNT(product_id) as P2_count
FROM sales
WHERE product_id=2
GROUP BY user_id;
```

Output:



Result Grid | Filter Rows:

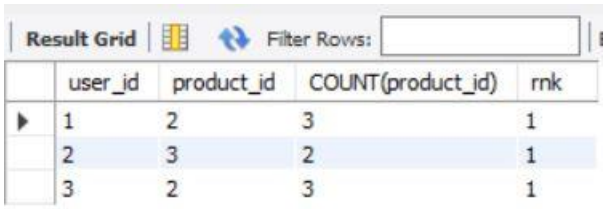
user_id	P2_count
1	3
3	3
2	1

5) Which item was most popular for each customer?

Query:

```
SELECT *
FROM (SELECT user_id, product_id, COUNT(product_id),
RANK() OVER(PARTITION BY user_id ORDER BY COUNT(product_id) DESC) as rnk
FROM sales
GROUP BY user_id,product_id) as a
WHERE rnk=1;
```

Output:



Result Grid | Filter Rows:

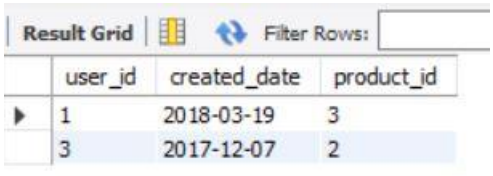
user_id	product_id	COUNT(product_id)	rnk
1	2	3	1
2	3	2	1
3	2	3	1

6) Which item was purchased first by customer after they become a member?

Query:

```
SELECT user_id,created_date,product_id
FROM(SELECT a.user_id,created_date,gold_signup_date, product_id,
RANK() OVER(PARTITION BY user_id ORDER BY created_date) as rnk
FROM goldusers_signup as a
INNER JOIN sales as b
ON a.user_id=b.user_id
WHERE a.gold_signup_date<=b.created_date
ORDER BY user_id, b.created_date) as t
WHERE rnk=1;
```

Output:



user_id	created_date	product_id
1	2018-03-19	3
3	2017-12-07	2

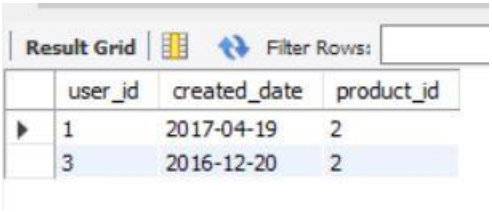
7) Which item was purchased just before the customer became a member?

Query:

```
SELECT user_id, created_date, product_id
FROM (SELECT a.user_id,b.created_date,b.product_id,
RANK() OVER(PARTITION BY user_id ORDER BY created_date DESC) as rnk
FROM goldusers_signup as a
INNER JOIN sales as b
ON a.user_id=b.user_id
WHERE a.gold_signup_date>=b.created_date
```

```
ORDER BY a.user_id,b.created_date DESC) as d
WHERE rnk=1;
```

Output:



Result Grid | Filter Rows:

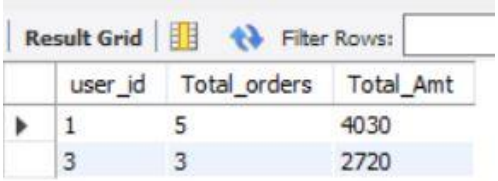
user_id	created_date	product_id
1	2017-04-19	2
3	2016-12-20	2

8) What are the total orders and amount spent for each customer before they become a member?

Query:

```
SELECT a.user_id,COUNT(a.user_id) as Total_orders,SUM(a.price) as Total_Amt
FROM (SELECT s.user_id,s.created_date,s.product_id,p.product_name,p.price
FROM sales as s
INNER JOIN product as p
ON s.product_id=p.product_id) as a
INNER JOIN goldusers_signup as b
ON a.user_id=b.user_id
WHERE a.created_date<=b.gold_signup_date
GROUP BY a.user_id
ORDER BY a.user_id;
```

Output:



Result Grid | Filter Rows:

user_id	Total_orders	Total_Amt
1	5	4030
3	3	2720

9) If buying each product generates points, for example, 5rs=2 zomato points and each product has different purchasing points, say for p1 5rs=1 zomato point, for p2 10rs=5 zomato point and p3 5rs=1 zomato point, calculate points collected by each customer and for which product most points have been given till now.

Query:

```
SELECT e.user_id, e.total_points,e.total_points*2.5 as cashback_earned
FROM(SELECT d.user_id,SUM(d.points) as Total_points
FROM (SELECT c.*,Amt/pir as points
FROM(SELECT b.*,
CASE WHEN b.product_id=1 THEN 5 WHEN b.product_id=2 THEN 2 WHEN
b.product_id=3 THEN 5 ELSE 0 END as pir
FROM (SELECT a.user_id, a.product_id, SUM(a.price) as Amt
FROM (SELECT s.user_id,p.product_id,p.price
FROM sales as s
INNER JOIN product as p
ON s.product_id=p.product_id) as a
GROUP BY user_id,product_id
ORDER BY user_id) as b) as c) as d
GROUP BY user_id) as e
GROUP BY user_id;
```

Output:

Result Grid | Filter Rows:

user_id	total_points	cashback_earned
1	1829.0000	4572.50000
2	763.0000	1907.50000
3	1697.0000	4242.50000

Query:

```
SELECT e.product_id, SUM(e.points) as Total_points
FROM(SELECT c.*,Amt/pir as points
FROM(SELECT b.*,
CASE WHEN b.product_id=1 THEN 5 WHEN b.product_id=2 THEN 2 WHEN
b.product_id=3 THEN 5 ELSE 0 END as pir
FROM (SELECT a.user_id, a.product_id, SUM(a.price) as Amt
FROM (SELECT s.user_id,p.product_id,p.price
FROM sales as s
INNER JOIN product as p
ON s.product_id=p.product_id) as a
GROUP BY user_id,product_id
ORDER BY user_id) as b) as c) as e
GROUP BY product_id
ORDER BY Total_points DESC
LIMIT 1;
```

Output:

Result Grid | Filter Rows:

	product_id	Total_points
▶	2	3045.0000

10) In the first year after a customer joins the gold program (including the join date), irrespective of what the customer has purchased, he/she earns 5 zomato points for every 10rs spent. Who earned more points 1 or 3 and what was their points earned in first year?

Query:

```
SELECT e.user_id,e.price/2 as points_earned
FROM(SELECT c.user_id,c.gold_signup_date,c.created_date,d.product_id,d.price
```

```
FROM(SELECT a.user_id,a.gold_signup_date,b.created_date,b.product_id
FROM goldusers_signup as a
INNER JOIN sales as b
ON a.user_id=b.user_id) as c
INNER JOIN product as d
ON c.product_id=d.product_id
WHERE created_date>=gold_signup_date AND
created_date<DATE_ADD(gold_signup_date, INTERVAL 1 YEAR)) as e;
```

Output:

Result Grid | Filter Rows:

user_id	points_earned
3	435.0000
1	165.0000

11) Rank all transactions of the customers.

Query:

```
SELECT *,
RANK() OVER (PARTITION BY user_id ORDER BY created_date) as rnk
FROM sales;
```

Output:

Result Grid | Filter Rows:

user_id	created_date	product_id	rnk
1	2016-03-11	1	1
1	2016-05-20	3	2
1	2016-11-09	1	3
1	2017-03-11	2	4
1	2017-04-19	2	5
1	2018-03-19	3	6
1	2019-10-23	2	7
2	2017-09-24	1	1
2	2017-11-08	2	2
2	2018-09-10	3	3
2	2020-07-20	3	4
3	2016-11-10	1	1
3	2016-12-15	2	2
3	2016-12-20	2	3
3	2017-12-07	2	4
3	2019-12-18	1	5

12) Rank all transactions for each customer whenever they are zomato gold member and for every non gold member transactions mark as NA.

Query:

```
SELECT b.user_id,a.gold_signup_date,b.created_date,

CASE WHEN created_date>=gold_signup_date THEN ROW_NUMBER() OVER
(PARTITION BY b.user_id ORDER BY created_date DESC) ELSE 'NA' END as rnk

FROM goldusers_signup as a

RIGHT JOIN sales as b

ON a.user_id=b.user_id;
```

Output:

Result Grid | Filter Rows: | Export:

user_id	gold_signup_date	created_date	rnk
1	2017-09-22	2019-10-23	1
1	2017-09-22	2018-03-19	2
1	2017-09-22	2017-04-19	NA
1	2017-09-22	2017-03-11	NA
1	2017-09-22	2016-11-09	NA
1	2017-09-22	2016-05-20	NA
1	2017-09-22	2016-03-11	NA
2	NULL	2020-07-20	NA
2	NULL	2018-09-10	NA
2	NULL	2017-11-08	NA
2	NULL	2017-09-24	NA
3	2017-04-21	2019-12-18	1
3	2017-04-21	2017-12-07	2
3	2017-04-21	2016-12-20	NA
3	2017-04-21	2016-12-15	NA
3	2017-04-21	2016-11-10	NA