DANNY'S DINER SQL PROJECT:

Q-1: Total amount spent by each customer at the restaurant?

```
SELECT customer_id, SUM(price) FROM dannys_diner.sales

INNER JOIN dannys_diner.menu ON
dannys_diner.sales.product_id=dannys_diner.menu.product_id

GROUP BY customer_id;
```

Q-2: How many days each customer visited restaurant?

```
SELECT customer_id,COUNT(DISTINCT order_date)

FROM  dannys_diner.sales

GROUP BY customer_id;
```

Q-3: Most purchased item on menu and its count**?**

```
SELECT product_name,COUNT(dannys_diner.sales.product_id) FROM
dannys_diner.sales INNER JOIN

dannys_diner.menu ON
dannys_diner.sales.product_id=dannys_diner.menu.product_id

where dannys_diner.sales.product_id=(SELECT
MAX(dannys_diner.sales.product_id) FROM dannys_diner.sales) GROUP BY
product_name ;
```

Q-4: Most popular item for each customer**?**

```
WITH r as  (SELECT
customer_id,dannys_diner.sales.product_id,product_name,COUNT(dannys_din
er.sales.product_id) ,RANK()OVER(PARTITION BY customer_id ORDER BY
COUNT(dannys_diner.sales.product_id) DESC) FROM dannys_diner.sales
```

```
 INNER JOIN  dannys_diner.menu ON
dannys_diner.sales.product_id=dannys_diner.menu.product_id

 GROUP BY customer_id,dannys_diner.sales.product_id,product_name

 ORDER BY customer_id )

 SELECT customer_id,product_name FROM r

 WHERE rank=1
```

Q-5: First item purchased by customer just after they became a member?

```
WITH temp as (SELECT
dannys_diner.sales.customer_id,product_name,ROW_NUMBER()

OVER (PARTITION BY dannys_diner.sales.customer_id ORDER BY
order_date)as r  FROM dannys_diner.sales

INNER JOIN dannys_diner.menu ON  dannys_diner.sales.product_id

=dannys_diner.menu.product_id

INNER JOIN dannys_diner.members ON dannys_diner.sales.customer_id

=dannys_diner.members.customer_id

WHERE order_date>=join_date )

SELECT customer_id,product_name FROM temp

WHERE r=1;
```

Q-6: Item purchased just before they became a member?

```
WITH t as (SELECT dannys_diner.sales.customer_id,product_name,RANK()
OVER (PARTITION BY dannys_diner.sales.customer_id ORDER BY order_date
DESC)as r  FROM dannys_diner.sales

INNER JOIN dannys_diner.menu ON  dannys_diner.sales.product_id

=dannys_diner.menu.product_id

INNER JOIN dannys_diner.members ON dannys_diner.sales.customer_id

=dannys_diner.members.customer_id WHERE order_date<join_date )
```

```
SELECT customer_id,product_name FROM t

where r=1;
```

Q-7: Total items and amount spent for each member before they became a member?

```
WITH t as (SELECT
dannys_diner.sales.customer_id,COUNT(dannys_diner.sales.product_id) as
total_items,SUM (price) as amount_spent FROM dannys_diner.sales

INNER JOIN dannys_diner.menu ON  dannys_diner.sales.product_id

=dannys_diner.menu.product_id

INNER JOIN dannys_diner.members ON dannys_diner.sales.customer_id

=dannys_diner.members.customer_id

WHERE order_date<join_date  GROUP BY dannys_diner.sales.customer_id
)

SELECT customer_id,total_items,amount_spent from t

order by customer_id;
```

Q-8: Each $1 spent equates to 10 points and sushi has a 2x points multiplier - how many points would each customer have?

```
WITH points as (SELECT dannys_diner.sales.customer_id,CASE

WHEN dannys_diner.sales.product_id=1 THEN SUM(price)*20

ELSE SUM(price)*10

END as c

FROM dannys_diner.sales

INNER JOIN dannys_diner.menu ON  dannys_diner.sales.product_id

=dannys_diner.menu.product_id

GROUP BY dannys_diner.sales.customer_id,dannys_diner.sales.product_id
```

```
)

SELECT customer_id,SUM(c) as pnts FROM points

Group by customer_id
```

CODE FOR creating a single table from the three tables where one is normal table and the other contains a ranking table according to membership:

1.
```
SELECT
dannys_diner.sales.customer_id,order_date,product_name,price,
CASE WHEN order_date<join_date OR
dannys_diner.members.customer_id
IS  NULL THEN 'N'
ELSE 'Y'
END as member
FROM dannys_diner.sales INNER JOIN dannys_diner.menu ON
dannys_diner.sales.product_id=dannys_diner.menu.product_id
FULL OUTER JOIN dannys_diner.members ON
dannys_diner.sales.customer_id=
dannys_diner.members.customer_id
ORDER BY customer_id,order_date
```

2.
```
WITH ran as(SELECT
dannys_diner.sales.customer_id,order_date,product_name,price,
CASE WHEN order_date<join_date OR
dannys_diner.members.customer_id
IS  NULL THEN 'N'
ELSE 'Y'
END as member
FROM dannys_diner.sales INNER JOIN dannys_diner.menu ON
dannys_diner.sales.product_id=dannys_diner.menu.product_id
FULL OUTER JOIN dannys_diner.members ON
dannys_diner.sales.customer_id=
dannys_diner.members.customer_id
ORDER BY customer_id,order_date
)
```

```
SELECT customer_id,order_date,product_name,price,member ,
CASE when member='N' THEN null
 ELSE  RANK() OVER (PARTITION BY customer_id,member ORDER BY
order_date)
END
FROM ran;
```

There are three tables used in this project: sales, menu, members

Conclusion: In this project, we have summarized the visiting patterns of the customer, how much they spent on each dish, their favorite dish and their ordering pattern before they became a member and after they became a member so that it helps him in expanding his customer outreach. At last we have used two queries to generate some basic dataset or table by combining all three tables.

Skills used: POSTGRE Sql, join, group by, case, rank, common table expression, etc...