



SEED

خوش حال خیر پختونخوا

atomcamp

SQL

Portfolio Project

Course Instructor

Miss Mahnoor Fatima

Teaching Assistant

Miss Nimra

Batch-02 DS

Submitted By: Muhammad Hamza

Data Science & AI Boot Camp

Table of Contents

Introduction	1
SQL Analysis	1
1. Total Customers	1
2. Monthly Distribution of Trial Plan Start Dates	1
3. Plan Start Dates After 2020	2
4. Customer Churn	2
5. Churn After Free Trial	2
6. Customer Plans After Free Trial.....	3
7. Plan Breakdown at End of 2020	3
8. Upgrades to Annual Plan in 2020.....	4
9. Average Time to Upgrade to Annual Plan.....	4
10. Breakdown of Upgrade Time into 30-Day Periods	5
11. Downgrades from Pro Monthly to Basic Monthly Plan in 2020	6
Conclusion.....	6

Project Report: Analysis of Foodie-Fi Subscription Data

Introduction

This report presents an analysis of the subscription data for Foodie-Fi, a food delivery service. The analysis is based on SQL queries that extract insights from the data. The SQL code used for the analysis is included in the report.

SQL Analysis

1. Total Customers

The total number of customers that Foodie-Fi has ever had is determined using the following SQL query:

```
SELECT
    COUNT(DISTINCT customer_id) AS num_customers
FROM subscriptions;
```



The screenshot shows a SQL query result grid. The top bar includes a 'Result Grid' tab, a 'Filter Rows' input field, and buttons for 'Export' and 'Wrap Cell Content'. The table has two columns: 'num_customers' and a value of 1000.

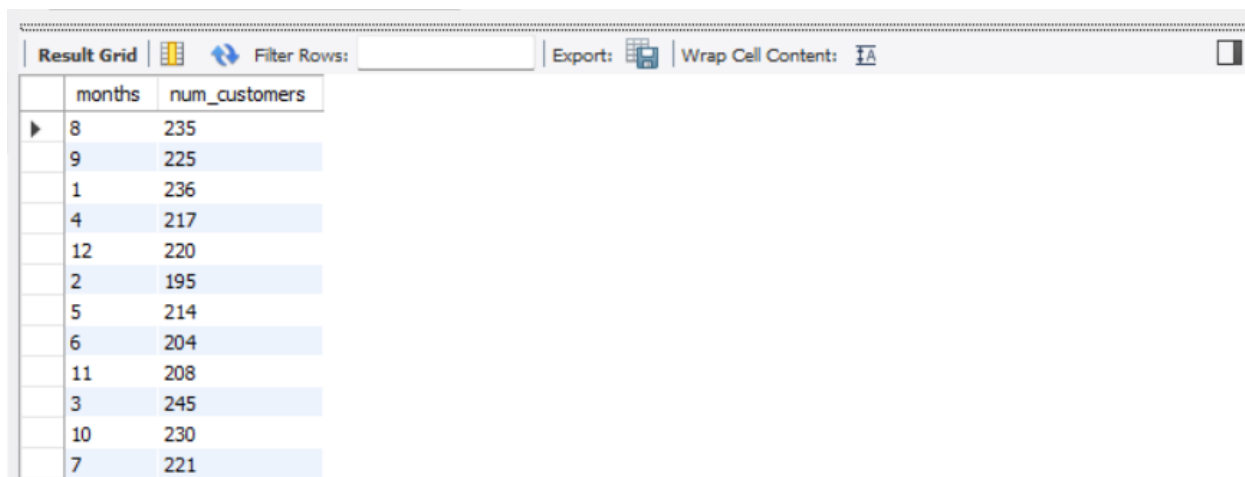
num_customers
1000

The result is 1000 customers.

2. Monthly Distribution of Trial Plan Start Dates

The monthly distribution of trial plan start dates is determined using the following SQL query:

```
SELECT
    MONTH(start_date) AS months,
    COUNT(customer_id) AS num_customers
FROM subscriptions
GROUP BY months;
```



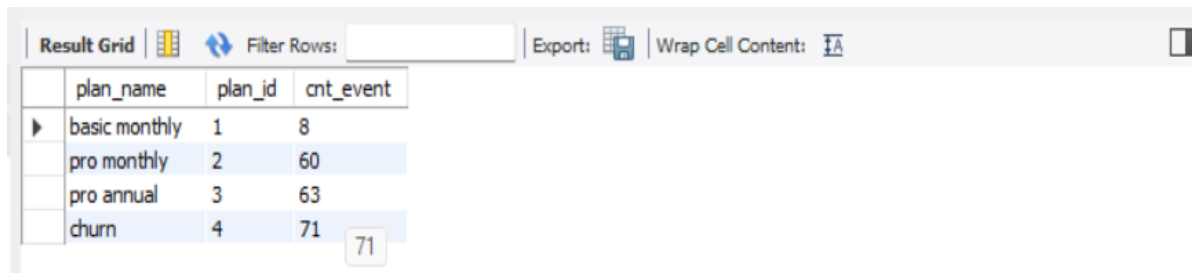
The screenshot shows a SQL query result grid. The top bar includes a 'Result Grid' tab, a 'Filter Rows' input field, and buttons for 'Export' and 'Wrap Cell Content'. The table has two columns: 'months' and 'num_customers'. The data is as follows:

months	num_customers
8	235
9	225
1	236
4	217
12	220
2	195
5	214
6	204
11	208
3	245
10	230
7	221

3. Plan Start Dates After 2020

The plan start dates that occur after the year 2020 are determined using the following SQL query:

```
SELECT
  p.plan_name,
  p.plan_id,
  COUNT(*) AS cnt_event
FROM subscriptions s
INNER JOIN plans p ON p.plan_id = s.plan_id
WHERE s.start_date >= '2021-01-01'
GROUP BY p.plan_id, p.plan_name
ORDER BY p.plan_id;
```

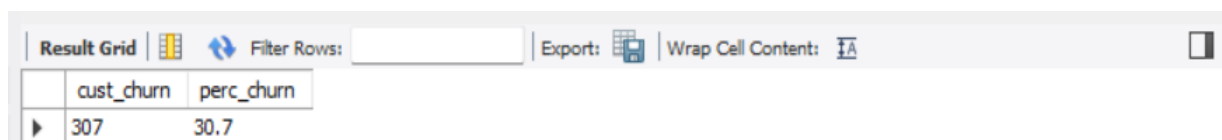


	plan_name	plan_id	cnt_event
▶	basic monthly	1	8
	pro monthly	2	60
	pro annual	3	63
	churn	4	71

4. Customer Churn

The customer count and percentage of customers who have churned is determined using the following SQL query:

```
SELECT
  COUNT(*) AS cust_churn,
  ROUND(COUNT(*) * 100 / (SELECT COUNT(DISTINCT customer_id) FROM subscriptions), 1) AS perc_churn
FROM subscriptions
WHERE plan_id = 4;
```



	cust_churn	perc_churn
▶	307	30.7

5. Churn After Free Trial

The number of customers who churned straight after their initial free trial is determined using the following SQL query:

```
WITH cte_churn AS (
  SELECT *, LAG(plan_id, 1) OVER(PARTITION BY customer_id ORDER BY plan_id) AS prev_plan
  FROM subscriptions)
SELECT
  COUNT(prev_plan) AS cnt_churn,
  ROUND(COUNT(*) * 100 / (SELECT COUNT(DISTINCT customer_id) FROM subscriptions), 0) AS
  perc_churn
```

```
FROM cte_churn
WHERE plan_id = 4 and prev_plan = 0;
```

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
cnt_churn	perc_churn		
92	9		

The result is a churn count of 92, which is 9% of the total customers.

6. Customer Plans After Free Trial

The number and percentage of customer plans after their initial free trial is determined using the following SQL query:

```
WITH cte_next_plan AS (
    SELECT
        *,
        LEAD(plan_id, 1) OVER(PARTITION BY customer_id ORDER BY plan_id) AS next_plan
    FROM subscriptions)
SELECT
    next_plan,
    COUNT(*) AS num_cust,
    ROUND(COUNT(*) * 100/(SELECT COUNT(DISTINCT customer_id) FROM subscriptions),1) AS
perc_next_plan
FROM cte_next_plan
WHERE next_plan is not null and plan_id = 0
GROUP BY next_plan
ORDER BY next_plan;
```

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
next_plan	num_cust	perc_next_plan	
1	546	54.6	
2	325	32.5	
3	37	3.7	
4	92	9.2	

7. Plan Breakdown at End of 2020

The customer count and percentage breakdown of all plan names at the end of 2020 is determined using the following SQL query:

```
WITH cte_next_date AS (
    SELECT
        *,
        LEAD(start_date, 1) OVER(PARTITION BY customer_id ORDER BY start_date) AS next_date
    FROM subscriptions
    WHERE start_date <= '2020-12-31'),
plans_breakdown AS(
    SELECT
        plan_id,
```

```

COUNT(DISTINCT customer_id) AS num_customer
FROM cte_next_date
WHERE (next_date IS NOT NULL AND (start_date < '2020-12-31' AND next_date > '2020-12-31'))
OR (next_date IS NULL AND start_date < '2020-12-31')
GROUP BY plan_id)
SELECT
    plan_id,
    num_customer,
    ROUND(num_customer * 100/(SELECT COUNT(DISTINCT customer_id) FROM subscriptions),1) AS
perc_customer
FROM plans_breakdown
GROUP BY plan_id, num_customer
ORDER BY plan_id;

```

plan_id	num_customer	perc_customer
0	19	1.9
1	224	22.4
2	326	32.6
3	195	19.5
4	235	23.5

8. Upgrades to Annual Plan in 2020

The number of customers who upgraded to an annual plan in 2020 is determined using the following SQL query:

```

SELECT
    COUNT(customer_id) AS num_customer
FROM subscriptions
WHERE plan_id = 3 AND start_date <= '2020-12-31';

```

num_customer
195

The result is 195 customers.

9. Average Time to Upgrade to Annual Plan

The average number of days it takes for a customer to upgrade to an annual plan from the day they join Foodie-Fi is determined using the following SQL query:

```

WITH annual_plan AS (
    SELECT
        customer_id,
        start_date AS annual_date
    FROM subscriptions
    WHERE plan_id = 3),
trial_plan AS (

```

```

SELECT
    customer_id,
    start_date AS trial_date
FROM subscriptions
WHERE plan_id = 0
)
SELECT
    ROUND(AVG(DATEDIFF(annual_date, trial_date)),0) AS avg_upgrade
FROM annual_plan ap
JOIN trial_plan tp ON ap.customer_id = tp.customer_id;

```

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
avg_upgrade			
105			

The result is an average of 105 days.

10. Breakdown of Upgrade Time into 30-Day Periods

The breakdown of the average upgrade time into 30-day periods is determined using the following SQL query:

```

WITH annual_plan AS (
    SELECT
        customer_id,
        start_date AS annual_date
    FROM subscriptions
    WHERE plan_id = 3),
trial_plan AS (
    SELECT
        customer_id,
        start_date AS trial_date
    FROM subscriptions
    WHERE plan_id = 0
),
day_period AS (
    SELECT
        DATEDIFF(annual_date, trial_date) AS diff
    FROM trial_plan tp
    LEFT JOIN annual_plan ap ON tp.customer_id = ap.customer_id
    WHERE annual_date is not null
),
bins AS (
    SELECT
        *, FLOOR(diff/30) AS bins
    FROM day_period)
SELECT
    CONCAT((bins * 30) + 1, ' - ', (bins + 1) * 30, ' days ') AS days,
    COUNT(diff) AS total
FROM bins
GROUP BY bins;

```

Result Grid			Filter Rows:	Export:	Wrap Cell Content:
	days	total			
	1 - 30 days	48			
▶	121 - 150 days	43			
	61 - 90 days	33			
	31 - 60 days	25			
	151 - 180 days	35			
	91 - 120 days	35			
	181 - 210 days	27			
	331 - 360 days	1			
	241 - 270 days	5			
	211 - 240 days	4			
	271 - 300 days	1			
	301 - 330 days	1			

11. Downgrades from Pro Monthly to Basic Monthly Plan in 2020

The number of customers who downgraded from a pro monthly to a basic monthly plan in 2020 is determined using the following SQL query:

```
WITH next_plan AS (
    SELECT
        *,
        LEAD(plan_id, 1) OVER(PARTITION BY customer_id ORDER BY start_date, plan_id) AS plan
    FROM subscriptions)
SELECT
    COUNT(DISTINCT customer_id) AS num_downgrade
FROM next_plan np
LEFT JOIN plans p ON p.plan_id = np.plan_id
WHERE p.plan_name = 'pro monthly' AND np.plan = 1 AND start_date <= '2020-12-31';
```

Result Grid			Filter Rows:	Export:	Wrap Cell Content:
	num_downgrade				
▶	0				

The result is 0 customers.

Conclusion

The SQL analysis provides valuable insights into the customer behavior and subscription trends of Foodie-Fi. These insights can be used to inform business decisions and improve customer retention strategies. The SQL code used for the analysis is robust and can be adapted for future analyses as the subscription data grows and evolves.