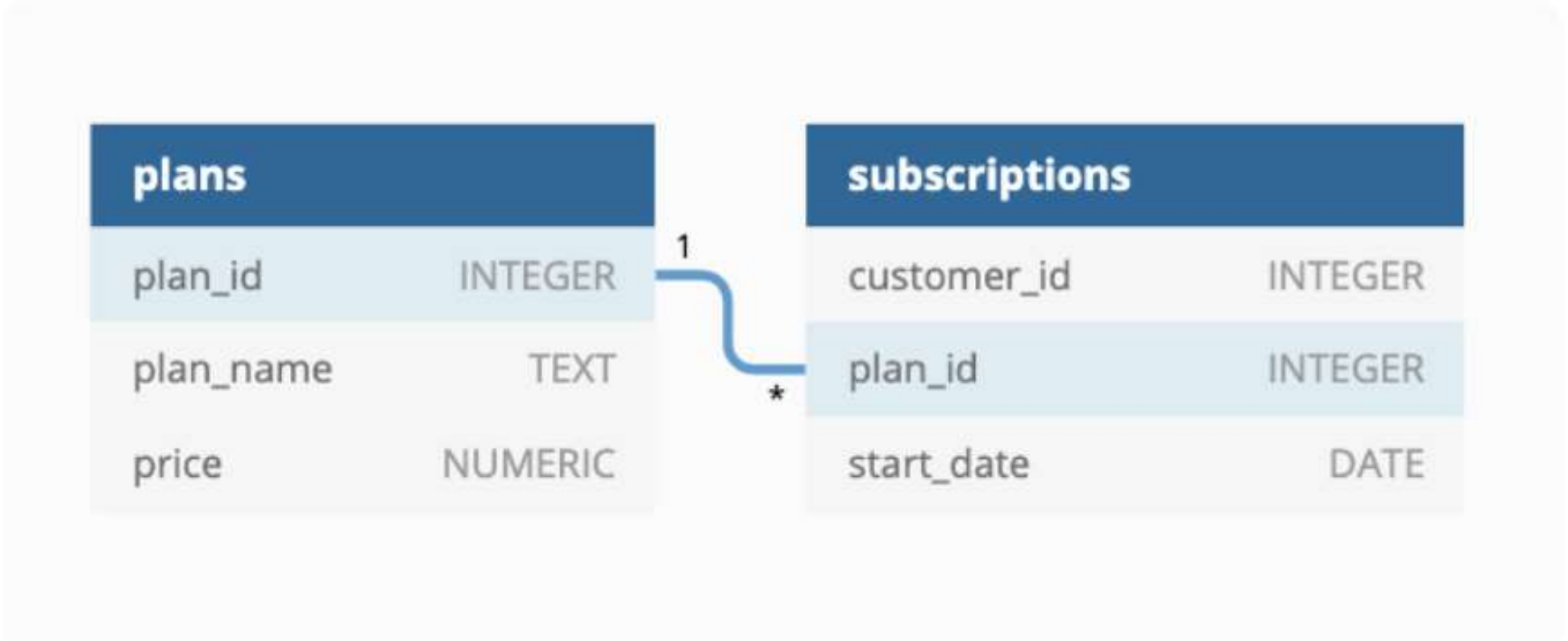
8 Week SQL Challenge

Case Study #3 - Foodie-Fi

Introduction

- Subscription based businesses are super popular and Danny realised that there was a large gap in the market - he wanted to create a new streaming service that only had food related content - something like Netflix but with only cooking shows!
- Danny finds a few smart friends to launch his new startup Foodie-Fi in 2020 and started selling monthly and annual subscriptions, giving their customers unlimited on-demand access to exclusive food videos from around the world!
- Danny created Foodie-Fi with a data driven mindset and wanted to ensure all future investment decisions and new features were decided using data. This case study focuses on using subscription style digital data to answer important business questions.

Entity Relationship Diagram



CASE STUDY

A. Customer Journey

1. Based off the 8 sample customers provided in the sample from the subscriptions table, write a brief description about each customer's onboarding journey.

- Customer id 1 Starts a free trial on 2020-08-01 and post his free trial, which is for 1 week, he upgrades his plan to "basic monthly" starting 2020-08-08 for $9.90.
- Customer id 2 Starts a free trial on 2020-09-20 and later upgrades to "pro annual" plan which costs $199.
- Customer id 4 Starts a free trial on 2020-01-17 and upgrades to "basic monthly" plan on 2020-01-24 and then churns his plan on 2020-04-21.

```
SELECT  p.plan_id, s.customer_id , p.plan_name,
    p.price, s.start_date
FROM plans AS p
INNER JOIN subscriptions AS s ON p.plan_id = s.plan_id
WHERE s.customer_id <=8
```

Output | Messages | Notifications

plan_id integer	customer_id integer	plan_name character varying (13)	price numeric (5,2)	start_date date
0	1	trial	0.00	2020-08-01
1	1	basic monthly	9.90	2020-08-08
0	2	trial	0.00	2020-09-20
3	2	pro annual	199.00	2020-09-27
0	3	trial	0.00	2020-01-13
1	3	basic monthly	9.90	2020-01-20
0	4	trial	0.00	2020-01-17
1	4	basic monthly	9.90	2020-01-24
4	4	churn	[null]	2020-04-21
0	5	trial	0.00	2020-08-03
1	5	basic monthly	9.90	2020-08-10
0	6	trial	0.00	2020-12-23
1	6	basic monthly	9.90	2020-12-30
4	6	churn	[null]	2021-02-26

B. Data Analysis Questions

1. How many customers has Foodie-Fi ever had?

```
SELECT COUNT(DISTINCT(customer_id)) as total_customer
FROM subscriptions
```

Output Messages Notifications

total_customer bigint
1000

2. What is the monthly distribution of trial plan start_date values for our dataset use the start of the month as the group by value

```sql
SELECT DATE_TRUNC('month', start_date)::date as monthly,
COUNT(customer_id) as monthly_subscribers
FROM subscriptions
WHERE plan_id = 0
GROUP BY DATE_TRUNC('month', start_date)
ORDER BY DATE_TRUNC('month', start_date)
```

monthly date	monthly_subscribers bigint
2020-01-01	88
2020-02-01	68
2020-03-01	94
2020-04-01	81
2020-05-01	88
2020-06-01	79
2020-07-01	89
2020-08-01	88
2020-09-01	87
2020-10-01	79
2020-11-01	75
2020-12-01	84

3. What plan start_date values occur after the year 2020 for our dataset? -- Show the breakdown by count of events for each plan_name

```sql
SELECT p.plan_id, plan_name,
    COUNT(*) as count_of_events
FROM plans as p
INNER JOIN subscriptions AS s ON p.plan_id = s.plan_id
WHERE DATE_PART('year', start_date) > 2020
GROUP BY plan_name, p.plan_id
ORDER BY p.plan_id
```

Output Messages Notifications

plan_id integer	plan_name character varying (13)	count_of_events bigint
1	basic monthly	8
2	pro monthly	60
3	pro annual	63
4	churn	71

4. What is the customer count and percentage of customers who have churned rounded to 1 decimal place?

```
SELECT COUNT(DISTINCT(customer_id)) as customer_count,
    ROUND(100 * COUNT(DISTINCT customer_id)
    /
    (SELECT COUNT(DISTINCT customer_id)
    FROM subscriptions),1) as churned_percentage
FROM subscriptions
WHERE plan_id = 4
```

Output Messages Notifications

customer_count bigint	churned_percentage numeric
307	30.0

5. How many customers have churned straight after their initial free trial what percentage is this rounded to the nearest whole number?

```
WITH CTE AS(
    SELECT customer_id, plan_id,COUNT(DISTINCT customer_id) as distinct_customer,
    ROW_NUMBER() OVER(PARTITION BY customer_id ORDER BY start_date ASC) as rn
    FROM subscriptions
    GROUP BY start_date, customer_id, plan_id
)
SELECT COUNT(*) as churned_after_trial,
    ROUND(100 * COUNT(DISTINCT customer_id)
    /
    (SELECT COUNT(DISTINCT customer_id)
    FROM subscriptions),0) as churned_percentage
FROM CTE
WHERE rn = 2 AND plan_id = 4
GROUP BY distinct_customer;
```

Output | Messages | Notifications

churned_after_trial bigint	churned_percentage numeric
92	9

6. What is the number and percentage of customer plans after their initial free trial?

```sql
WITH CTE AS(
    SELECT customer_id, plan_name,
    ROW_NUMBER() OVER(PARTITION BY customer_id ORDER BY start_date ASC) as rn
    FROM subscriptions AS s
    INNER JOIN plans AS p ON s.plan_id = p.plan_id
)
SELECT plan_name,
    COUNT(customer_id) as count_of_customer,
    CONCAT(ROUND(100 * COUNT(customer_id) /
    (SELECT COUNT(DISTINCT customer_id) FROM subscriptions), 1),'%') as customer_percentage
FROM CTE
WHERE rn = 2
GROUP BY plan_name
ORDER BY COUNT(customer_id)
```

plan_name character varying (13)	count_of_customer bigint	customer_percentage text
pro annual	37	3.0%
churn	92	9.0%
pro monthly	325	32.0%
basic monthly	546	54.0%

7. What is the customer count and percentage breakdown of all 5 plan_name values at 2020-12-31?

```sql
WITH CTE AS(
    SELECT *,
    ROW_NUMBER() OVER(PARTITION BY customer_id ORDER BY start_date DESC) as rn
    FROM subscriptions
    WHERE start_date <= '2020-12-31'
)
SELECT plan_name, COUNT(customer_id) as customer_count,
    CONCAT(ROUND(100 * COUNT(customer_id) /
    (SELECT COUNT(DISTINCT customer_id) FROM subscriptions), 1),'%') as customer_percentage
FROM CTE
INNER JOIN plans as p ON CTE.plan_id = p.plan_id
WHERE rn = 1
GROUP BY plan_name
ORDER BY COUNT(customer_id);
```

plan_name character varying	customer_count bigint	customer_percentage text
trial	19	1.0%
pro annual	195	19.0%
basic monthly	224	22.0%
churn	236	23.0%
pro monthly	326	32.0%

8. How many customers have upgraded to an annual plan in 2020?

```sql
SELECT plan_name, COUNT(customer_id) as total_upgraded_customers
FROM subscriptions AS s
INNER JOIN plans AS p ON s.plan_id = p.plan_id
WHERE DATE_PART('year', start_date) = 2020 AND plan_name = 'pro annual'
GROUP BY plan_name
```

plan_name character varying	total_upgraded_customers bigint
pro annual	195

9. How many days on average does it take for a customer to an annual plan from the day they join Foodie-Fi?

```sql
WITH TRIAL AS(
    SELECT customer_id, start_date AS trial_start
    FROM subscriptions
    WHERE plan_id = 0
), PRO AS(
    SELECT customer_id, start_date AS pro_start
    FROM subscriptions
    WHERE plan_id = 3
)
SELECT
    ROUND(AVG(EXTRACT(day FROM
    pro_start::timestamp - trial_start::timestamp)),0) as Avg_date_diff
FROM TRIAL t
INNER JOIN PRO as p ON t.customer_id = p.customer_id
```

10. Can you further breakdown this average value into 30 day periods (i.e. 0-30 days, 31-60 days etc)

```sql
WITH TRIAL AS(
    SELECT customer_id, start_date AS trial_start
    FROM subscriptions
    WHERE plan_id = 0
), PRO AS(
    SELECT customer_id, start_date AS pro_start
    FROM subscriptions
    WHERE plan_id = 3
)
SELECT
CASE
    WHEN EXTRACT(day FROM pro_start::timestamp - trial_start::timestamp) <= 30 THEN '0-30'
    WHEN EXTRACT(day FROM pro_start::timestamp - trial_start::timestamp) <= 60 THEN '31-60'
    WHEN EXTRACT(day FROM pro_start::timestamp - trial_start::timestamp) <= 90 THEN '61-90'
    WHEN EXTRACT(day FROM pro_start::timestamp - trial_start::timestamp) <= 120 THEN '91-120'
    WHEN EXTRACT(day FROM pro_start::timestamp - trial_start::timestamp) <= 150 THEN '121-150'
    WHEN EXTRACT(day FROM pro_start::timestamp - trial_start::timestamp) <= 180 THEN '151-180'
    WHEN EXTRACT(day FROM pro_start::timestamp - trial_start::timestamp) <= 210 THEN '181-210'
    WHEN EXTRACT(day FROM pro_start::timestamp - trial_start::timestamp) <= 240 THEN '211-240'
    WHEN EXTRACT(day FROM pro_start::timestamp - trial_start::timestamp) <= 270 THEN '241-270'
    WHEN EXTRACT(day FROM pro_start::timestamp - trial_start::timestamp) <= 300 THEN '271-300'
    WHEN EXTRACT(day FROM pro_start::timestamp - trial_start::timestamp) <= 330 THEN '301-330'
    WHEN EXTRACT(day FROM pro_start::timestamp - trial_start::timestamp) <= 360 THEN '331-360'
    END as day_periods,
    COUNT(t.customer_id) as customer_count
FROM TRIAL t
INNER JOIN PRO as p ON t.customer_id = p.customer_id
GROUP BY 1
ORDER BY 1
```

day_periods text	customer_count bigint
0-30	49
121-150	42
151-180	36
181-210	26
211-240	4
241-270	5
271-300	1
301-330	1
31-60	24
331-360	1
61-90	34
91-120	35

11. How many customers downgraded from a pro monthly to a basic monthly plan in 2020?

No one Downgraded from Pro Monthly to Basic Monthly plan in 2020.

```sql
WITH PRO_MON AS(
SELECT customer_id,
    start_date as promon_date
FROM subscriptions
WHERE plan_id = 2
), BASIC AS(
SELECT customer_id,
    start_date as basic_date
FROM subscriptions
WHERE plan_id = 1
)
SELECT pm.customer_id,
    promon_date,
    basic_date
FROM PRO_MON as pm
INNER JOIN BASIC as b ON pm.customer_id = b.customer_id
WHERE promon_date < basic_date AND DATE_PART('year', basic_date) = 2020
```