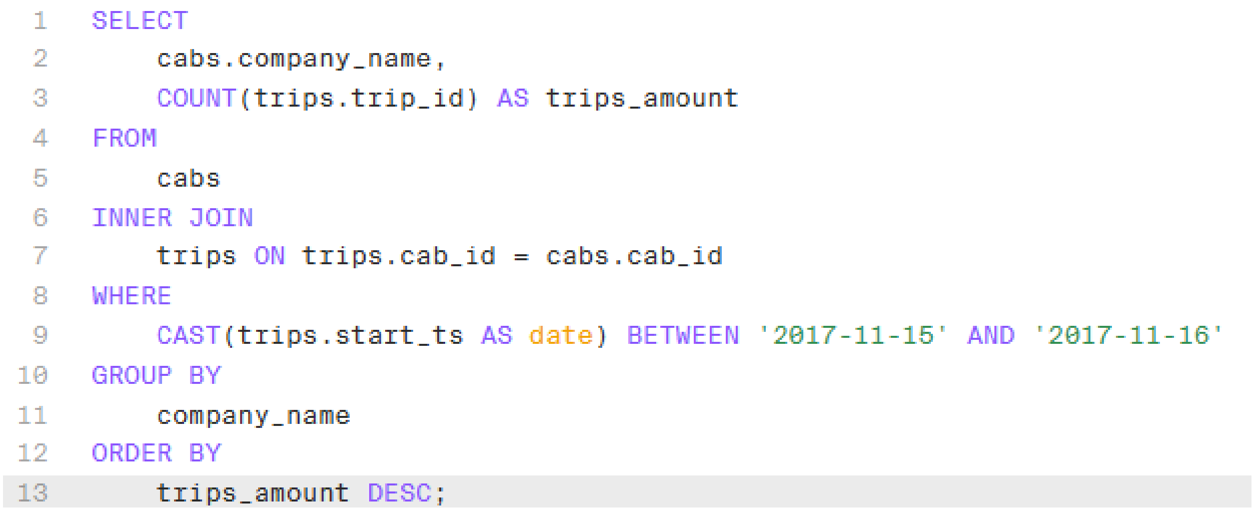
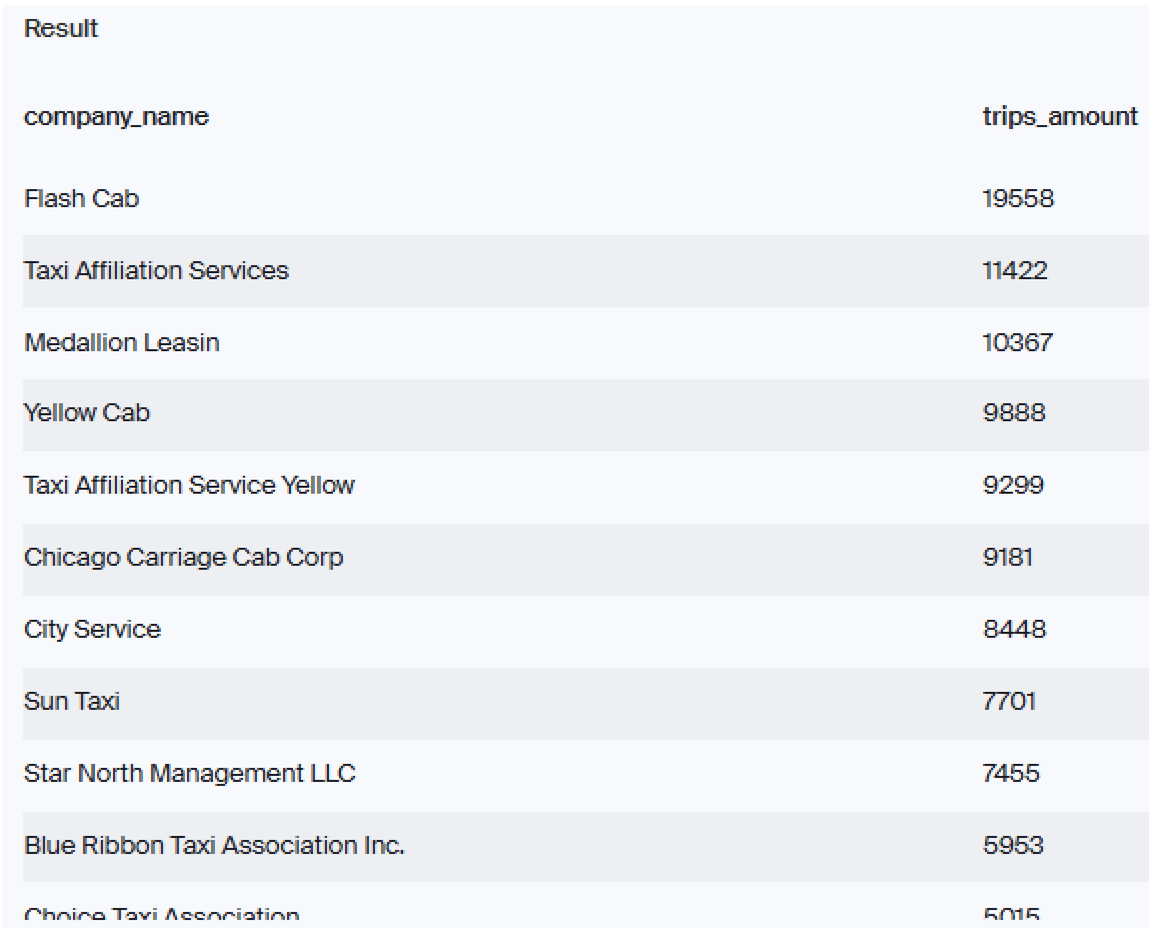
Sprint SQL Zuber project 6

1. Print the *company_name* field. Find the number of taxi rides for each taxi company for November 15-16, 2017, name the resulting field *trips_amount* and print it, too. Sort the results by the *trips_amount* field in descending order.



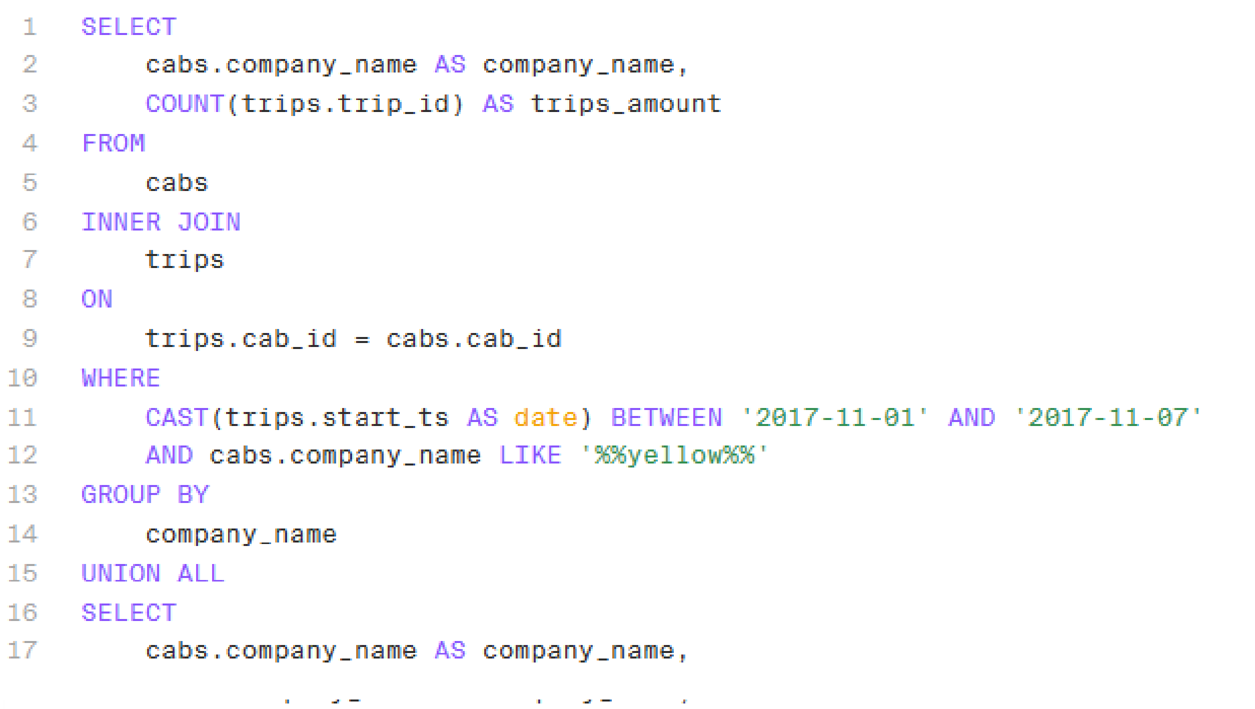
```sql
SELECT
    cabs.company_name,
    COUNT(trips.trip_id) AS trips_amount
FROM
    cabs
INNER JOIN
    trips ON trips.cab_id = cabs.cab_id
WHERE
    CAST(trips.start_ts AS date) BETWEEN '2017-11-15' AND '2017-11-16'
GROUP BY
    company_name
ORDER BY
    trips_amount DESC;
```



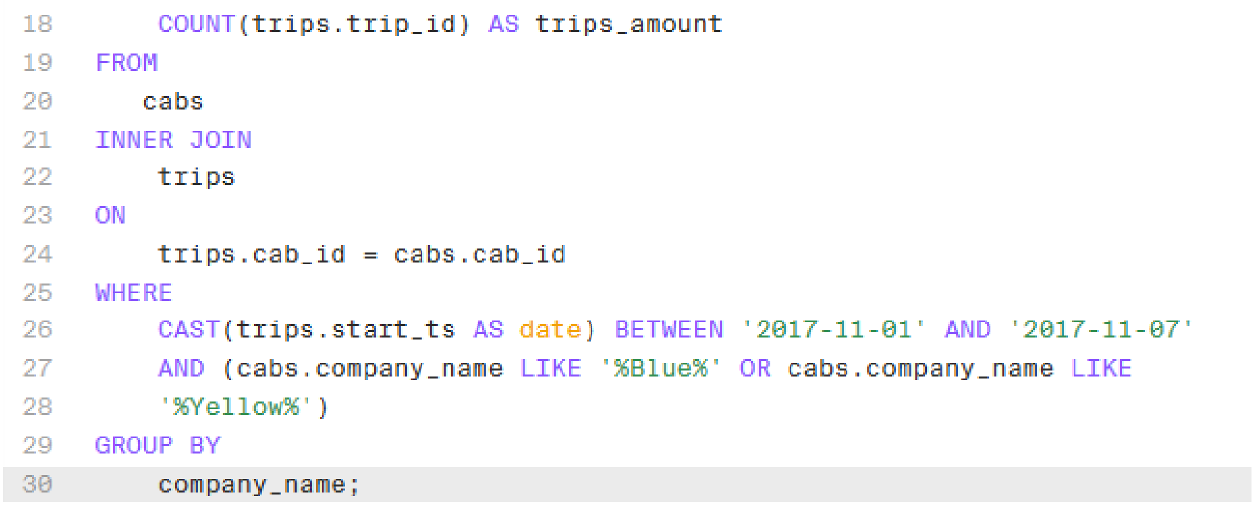
Result

company_name	trips_amount
Flash Cab	19558
Taxi Affiliation Services	11422
Medallion Leasin	10367
Yellow Cab	9888
Taxi Affiliation Service Yellow	9299
Chicago Carriage Cab Corp	9181
City Service	8448
Sun Taxi	7701
Star North Management LLC	7455
Blue Ribbon Taxi Association Inc.	5953
Choice Taxi Association	5015

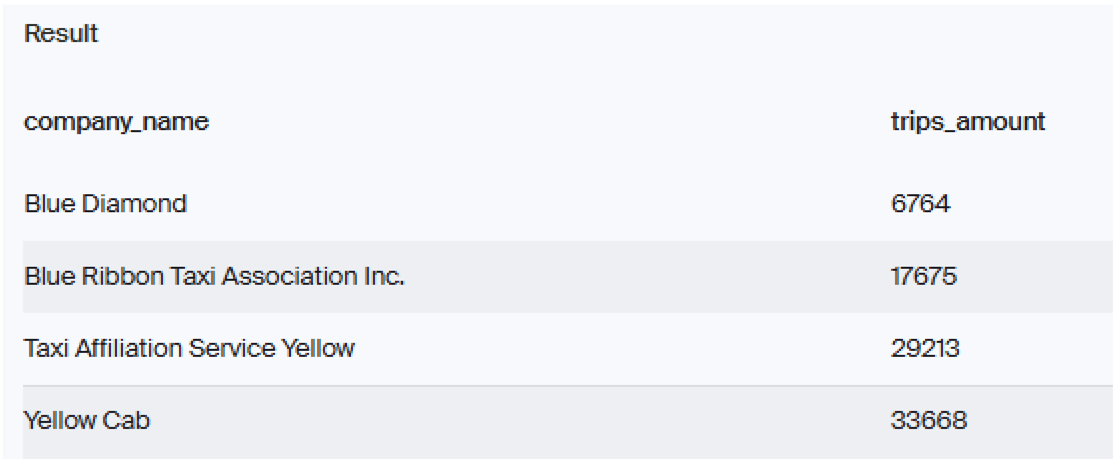
2. Find the number of rides for every taxi companies whose name contains the words "Yellow" or "Blue" for November 1-7, 2017. Name the resulting variable *trips_amount.* Group the results by the *company_name* field.



```sql
SELECT
    cabs.company_name AS company_name,
    COUNT(trips.trip_id) AS trips_amount
FROM
    cabs
INNER JOIN
    trips
ON
    trips.cab_id = cabs.cab_id
WHERE
    CAST(trips.start_ts AS date) BETWEEN '2017-11-01' AND '2017-11-07'
    AND cabs.company_name LIKE '%%yellow%%'
GROUP BY
    company_name
UNION ALL
SELECT
    cabs.company_name AS company_name,
```



```sql
    COUNT(trips.trip_id) AS trips_amount
FROM
    cabs
INNER JOIN
    trips
ON
    trips.cab_id = cabs.cab_id
WHERE
    CAST(trips.start_ts AS date) BETWEEN '2017-11-01' AND '2017-11-07'
    AND (cabs.company_name LIKE '%Blue%' OR cabs.company_name LIKE
    '%Yellow%')
GROUP BY
    company_name;
```



Result

company_name	trips_amount
Blue Diamond	6764
Blue Ribbon Taxi Association Inc.	17675
Taxi Affiliation Service Yellow	29213
Yellow Cab	33668

3. For November 1-7, 2017, the most popular taxi companies were Flash Cab and Taxi Affiliation Services. Find the number of rides for these two companies and name the resulting variable *trips_amount.* Join the rides for all other companies in the group

"Other." Group the data by taxi company names. Name the field with taxi company names *company*. Sort the result in descending order by *trips_amount*.

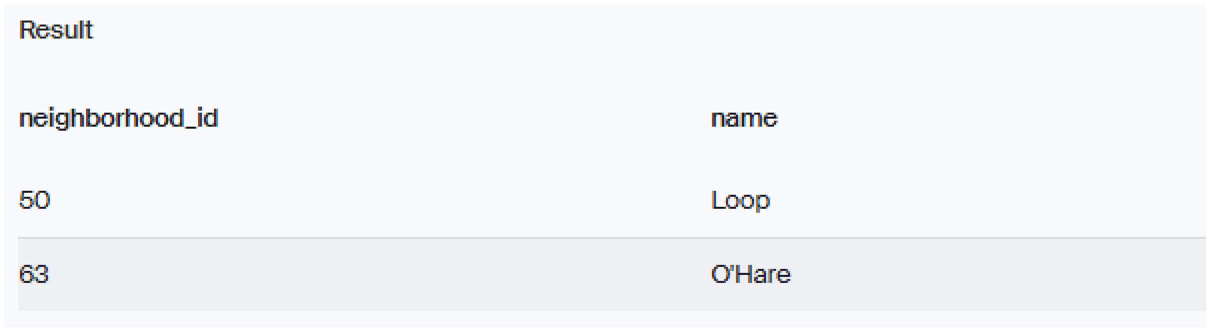
```
SELECT
    CASE
        WHEN company_name = 'Flash Cab' THEN 'Flash Cab'
        WHEN company_name = 'Taxi Affiliation Services' THEN
        'Taxi Affiliation Services'
        ELSE 'Other'
    END AS company,
    COUNT(trips.trip_id) AS trips_amount
FROM
    cabs
JOIN
    trips ON cabs.cab_id = trips.cab_id
WHERE
    CAST(trips.start_ts AS DATE) BETWEEN '2017-11-01' AND '2017-11-07'
GROUP BY
    company
ORDER BY
    trips_amount DESC;
```

Result

company	trips_amount
Other	335771
Flash Cab	64084
Taxi Affiliation Services	37583

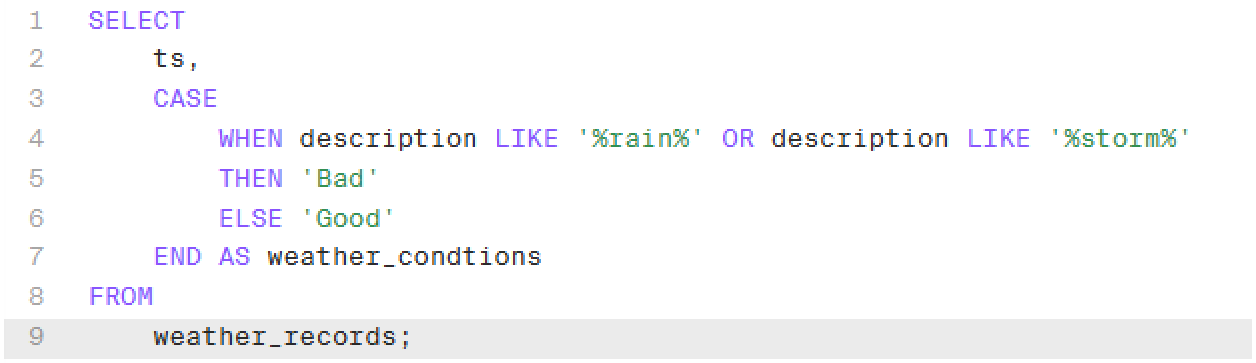
4. Retrieve the identifiers of the O'Hare and Loop neighborhoods from the *neighborhoods* table.

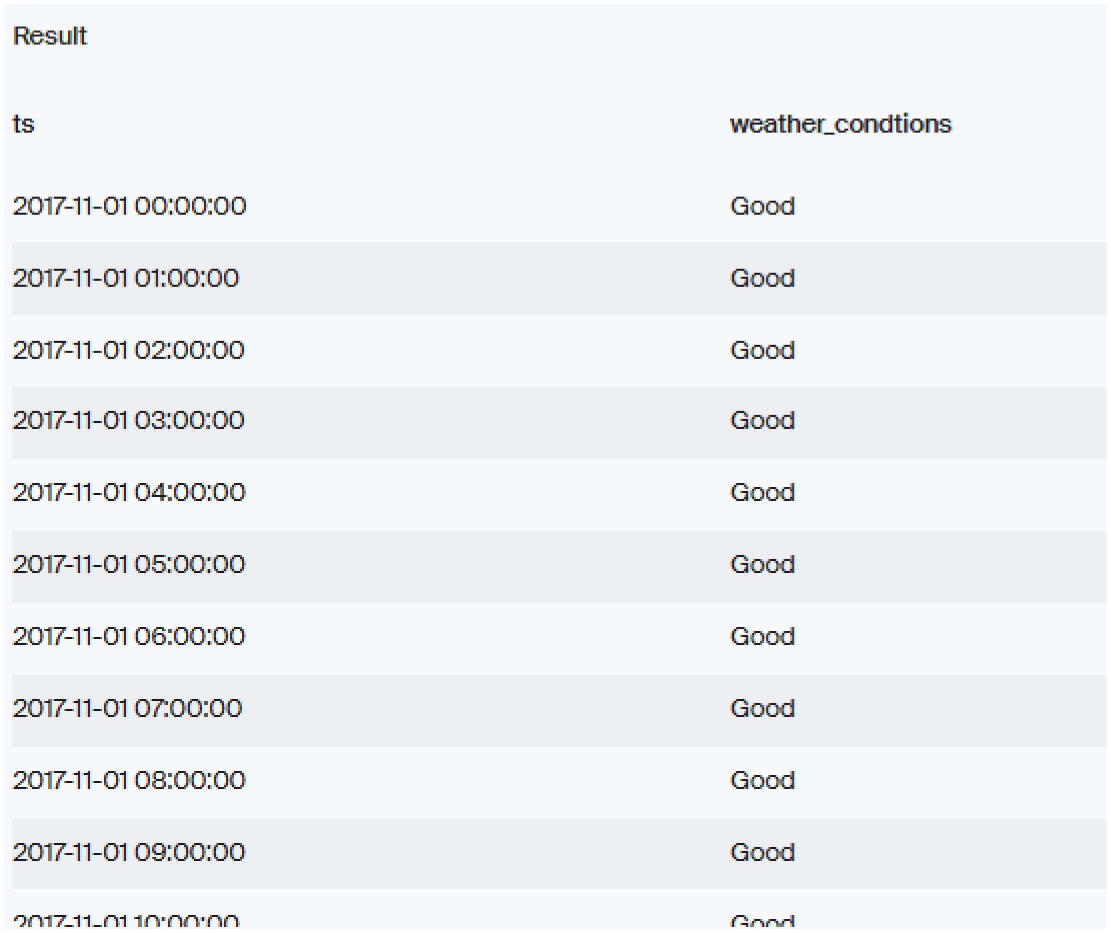
```
SELECT
    neighborhood_id,
    name
FROM
    neighborhoods
WHERE
    name LIKE '%Hare%' OR name LIKE 'Loop';
```


Result

neighborhood_id	name
50	Loop
63	O'Hare

5. For each hour, retrieve the weather condition records from the *weather_records* table. Using the CASE operator, break all hours into two groups: `Bad` if the *description* field contains the words `rain` or `storm`, and `Good` for others. Name the resulting field *weather_conditions*. The final table must include two fields: date and hour (*ts*) and *weather_conditions*.


```sql
SELECT
    ts,
    CASE
        WHEN description LIKE '%rain%' OR description LIKE '%storm%'
        THEN 'Bad'
        ELSE 'Good'
    END AS weather_condtions
FROM
    weather_records;
```



Result

ts	weather_condtions
2017-11-01 00:00:00	Good
2017-11-01 01:00:00	Good
2017-11-01 02:00:00	Good
2017-11-01 03:00:00	Good
2017-11-01 04:00:00	Good
2017-11-01 05:00:00	Good
2017-11-01 06:00:00	Good
2017-11-01 07:00:00	Good
2017-11-01 08:00:00	Good
2017-11-01 09:00:00	Good
2017-11-01 10:00:00	Good

6. Retrieve from the *trips* table all the rides that started in the Loop (*pickup_location_id:* 50) on a Saturday and ended at O'Hare (*dropoff_location_id*: 63). Get the weather conditions for each ride. Use the method you applied in the previous task. Also, retrieve the duration of each ride. Ignore rides for which data on weather conditions is not available.

The table columns should be in the following order:

- *start_ts*
- *weather_conditions*
- *duration_seconds*

Sort by *trip_id.*

```sql
SELECT
    start_ts,
    T.weather_conditions,
    duration_seconds
FROM
    trips
INNER JOIN (
    SELECT
        ts,
        CASE
            WHEN description LIKE '%rain' OR description
LIKE '%storm%' THEN 'Bad'
            ELSE 'Good'
        END AS weather_conditions
    FROM
        weather_records
) T ON T.ts = trips.start_ts
WHERE
    pickup_location_id = 50 AND trips.dropoff_location_id = 63 AND
EXTRACT(DOW FROM trips.start_ts) = 6
ORDER BY trips.trip_id;
```

Result

start_ts	weather_conditions	duration_seconds
2017-11-25 12:00:00	Good	1380
2017-11-25 16:00:00	Good	2410
2017-11-25 14:00:00	Good	1920
2017-11-25 12:00:00	Good	1543
2017-11-04 10:00:00	Good	2512
2017-11-11 07:00:00	Good	1440
2017-11-11 04:00:00	Good	1320
2017-11-04 16:00:00	Bad	2969
2017-11-18 11:00:00	Good	2280
2017-11-04 16:00:00	Bad	3120
2017-11-11 15:00:00	Good	4800