The Zuber Database

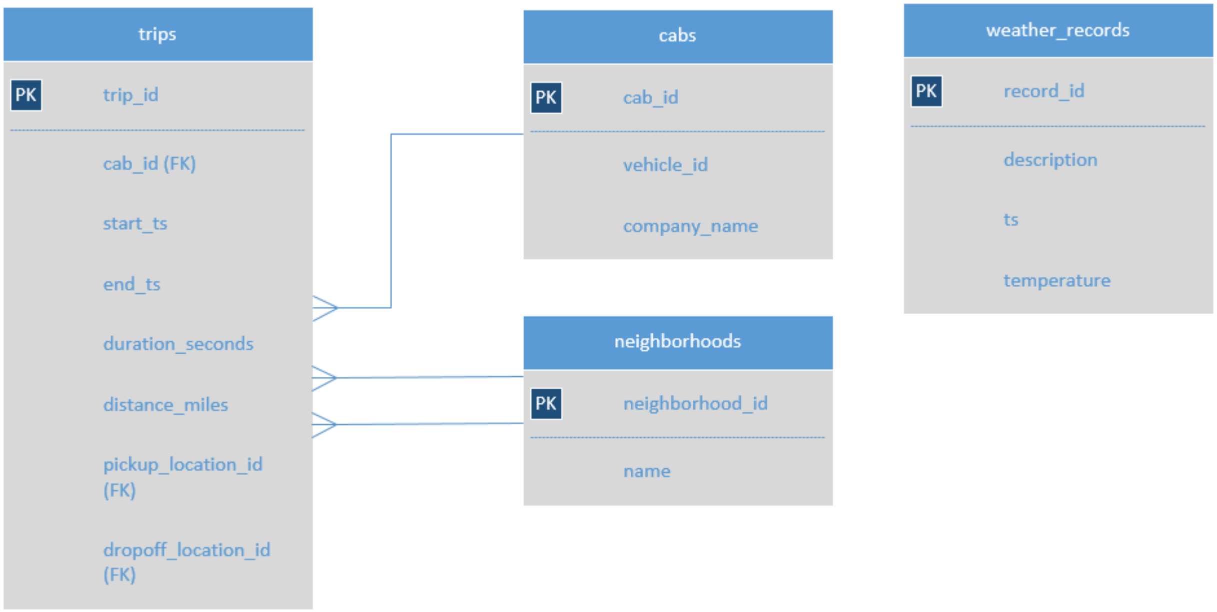
You're working as an analyst for Zuber, a new ride-sharing company that's launching in Chicago. Your task is to find patterns in the available information. You want to understand passenger preferences and the impact of external factors on rides.

You'll study a database, analyze data from competitors, and investigate the impact of weather on ride frequency.

Description of the data

A database with info on taxi rides in Chicago:

Table scheme



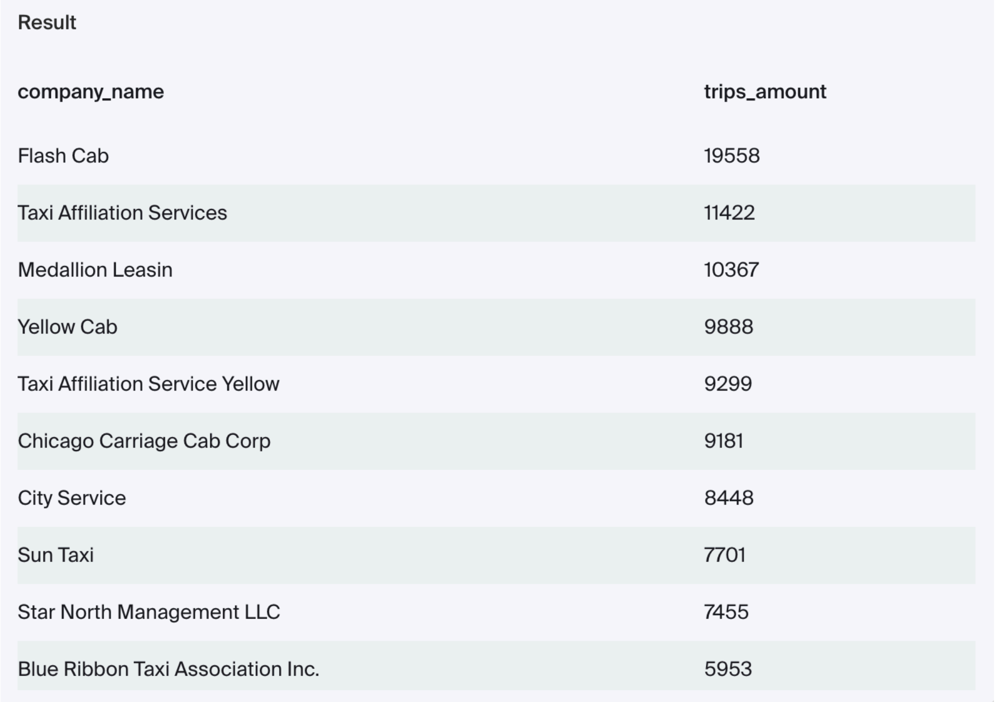
Note: there isn't a direct connection between the tables *trips* and *weather_records* in the database. But you can still use JOIN and link them using the time the ride started (*trips.start_ts*) and the time the weather record was taken (*weather_records.ts*).

Tasks 1-4: Exploratory data analysis

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.
   a. SQL:

```
SELECT
   cabs.company_name AS company_name,
   COUNT(trips.start_ts) AS trips_amount
FROM
   trips
   INNER JOIN cabs ON cabs.cab_id = trips.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
```

   b. Results



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

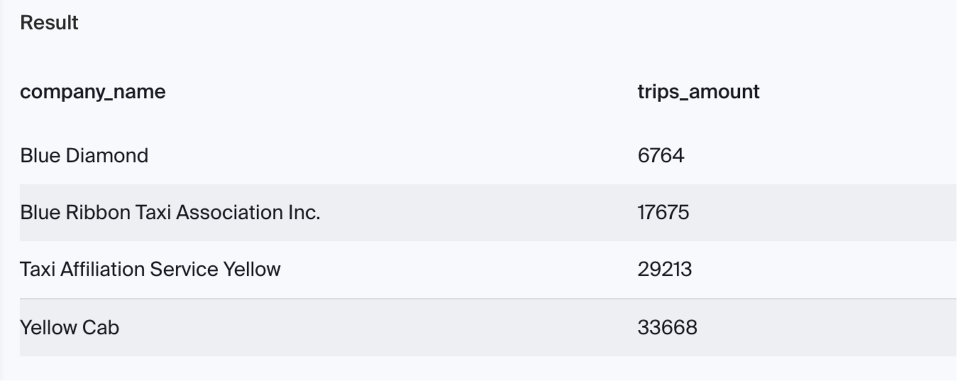
2. Find the number of rides for every taxi companies whose name contains the words "Yellow" or "Blue" for November 1-7, 2017. Name the

resulting variable *trips_amount.* Group the results by the *company_name* field.

a. SQL

```
SELECT
   cabs.company_name AS company_name,
   COUNT(trips.start_ts) AS trips_amount
FROM
   trips
   INNER JOIN cabs 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_name
HAVING
   company_name LIKE '%Yellow%' OR company_name LIKE '%Blue%';
```

b. Results



Result

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

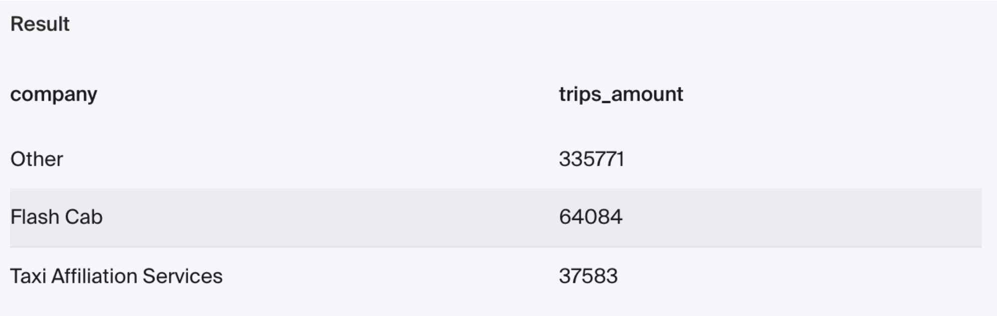
3. For November 1-7, 2017, the most popular taxi companies were Flash Cab and Taxi Affiliation Services. Find the number of rides for these two companies and name the resulting variable *trips_amount.* Join the rides for all other companies in the group "Other." Group the data by taxi company names. Name the field with taxi company names *company*. Sort the result in descending order by *trips_amount*.

a. SQL

```
SELECT
   CASE
      WHEN cabs.company_name LIKE '%Flash Cab%' THEN 'Flash Cab'
      WHEN cabs.company_name LIKE '%Taxi Affiliation Services%'
THEN 'Taxi Affiliation Services'
      ELSE 'Other'
```

```
    END AS company,
    COUNT(trips.trip_id) AS trips_amount
FROM
    trips
INNER JOIN
    cabs 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;
```

b. Resutls



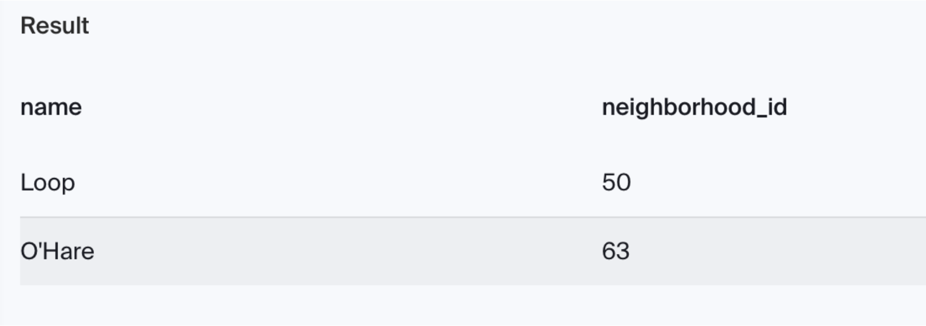
Result

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.
    a. SQL

```
SELECT
    name,
    neighborhood_id
FROM
    neighborhoods
WHERE
    name = 'O''Hare'
    OR name = 'Loop';
```

    b. Results

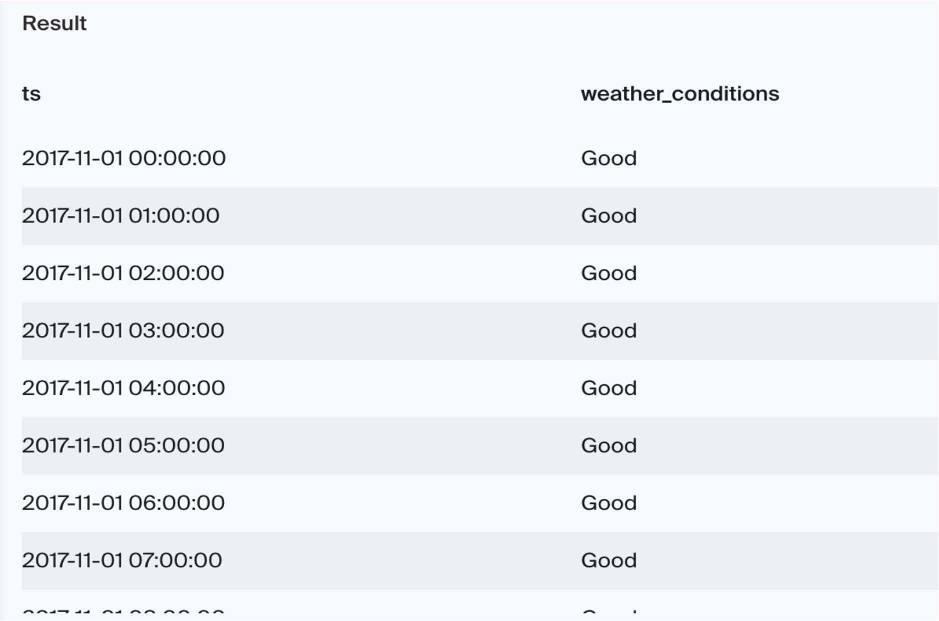


Result

name	neighborhood_id
Loop	50
O'Hare	63

Tasks 5-7: Investigate whether the duration of rides from the the Loop to O'Hare International Airport changes on rainy Saturdays

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*.
    a. SQL
    ```
    SELECT
       ts,
       CASE
          WHEN description LIKE '%rain%' THEN 'Bad'
          WHEN description LIKE '%storm%' THEN 'Bad'
          ELSE 'Good'
       END AS weather_conditions
    FROM
       weather_records;
    ```
    b. Results



Result

ts	weather_conditions
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

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:

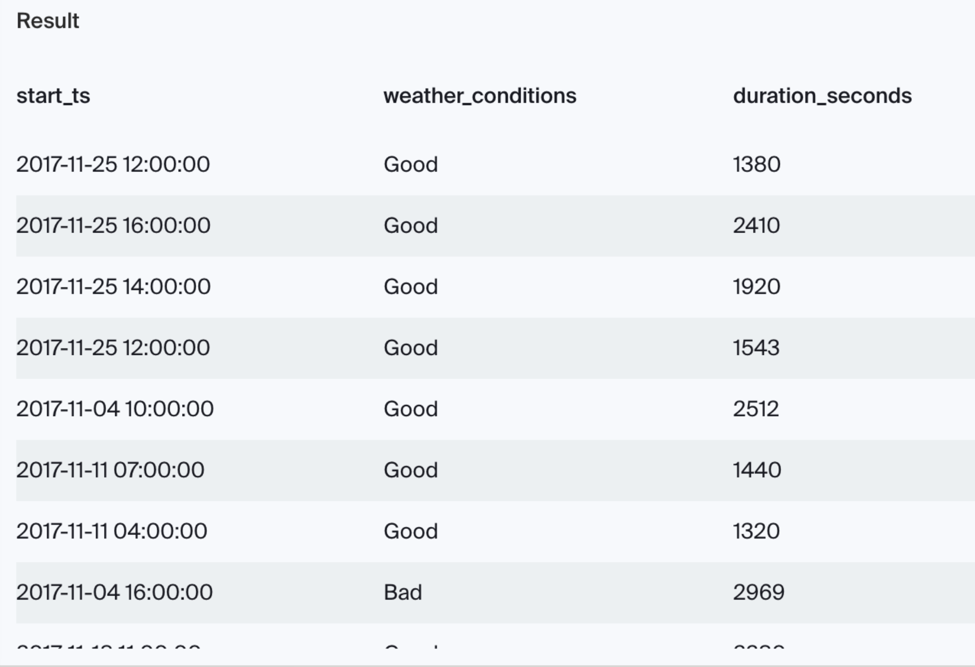
   - *start_ts*
   - *weather_conditions*
   - *duration_seconds*

   Sort by *trip_id.*

   a. SQL

```
SELECT
    trips.start_ts,
    CASE
        WHEN weather_records.description LIKE '%rain%' THEN 'Bad'
        WHEN weather_records.description LIKE '%storm%' THEN 'Bad'
        ELSE 'Good'
    END AS weather_conditions,
    trips.duration_seconds
FROM
    trips
INNER JOIN
    weather_records ON weather_records.ts = trips.start_ts
WHERE
    trips.pickup_location_id = 50
    AND trips.dropoff_location_id = 63
    AND EXTRACT(DOW FROM trips.start_ts) = 6
ORDER BY
    trips.trip_id;
```

b. Results



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	[illegible]