The Zuber database:

Project description:

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. Working with a database, you'll analyze data from competitors and test a hypothesis about the impact of weather on ride frequency.

Description of the data

A database with info on taxi rides in Chicago:

neighborhoods table: data on city neighborhoods

    name: name of the neighborhood
    neighborhood_id: neighborhood code

cabs table: data on taxis

    cab_id: vehicle code
    vehicle_id: the vehicle's technical ID
    company_name: the company that owns the vehicle

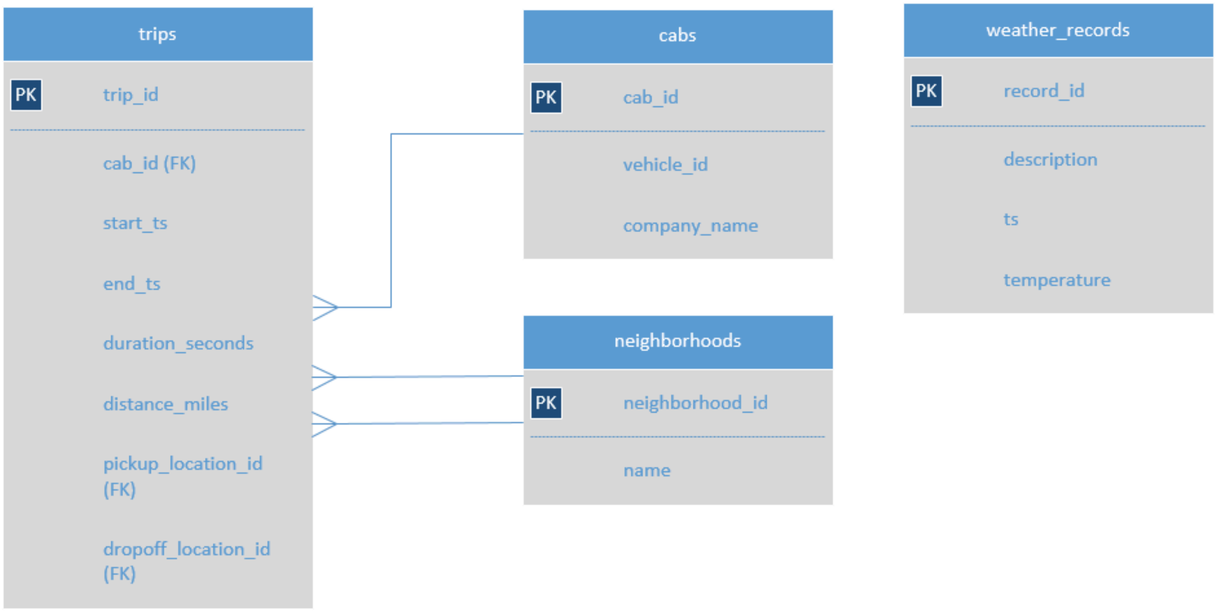
trips table: data on rides

    trip_id: ride code
    cab_id: code of the vehicle operating the ride
    start_ts: date and time of the beginning of the ride (time rounded to the hour)
    end_ts: date and time of the end of the ride (time rounded to the hour)
    duration_seconds: ride duration in seconds
    distance_miles: ride distance in miles
    pickup_location_id: pickup neighborhood code
    dropoff_location_id: dropoff neighborhood code

weather_records table: data on weather

    record_id: weather record code
    ts: record date and time (time rounded to the hour)
    temperature: temperature when the record was taken
    description: brief description of weather conditions, e.g. "light rain" or "scattered clouds"

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).

Instructions on completing the project

Step 1. Exploratory data analysis

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

Find the number of rides for every taxi company 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.

In November 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.

Step 2. Determine if and how the duration of rides from the Loop to O'Hare International Airport changes on rainy Saturdays compared to other days of the week and other weather conditions.

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

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.

Retrieve from the trips table all the rides that started in the Loop (neighborhood_id: 50) and ended at O'Hare (neighborhood_id: 63) on a Saturday. 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 takeaway sheets and summaries from previous lessons have everything you need to complete the project.

Project requirements:

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.

Code:

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

2.

Find the number of rides for every taxi company 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.

Code:

```
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,
   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%%'
GROUP BY company_name;
```

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*.

Code:

```
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
   INNER 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;
```

4.

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

Code:

```
SELECT
```

```
    neighborhood_id,
    name
FROM
    neighborhoods
WHERE
    name LIKE 'Loop'
    OR name LIKE '%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*.

Code:

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

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.*

Code:

```
SELECT
   trips.start_ts AS 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 AS 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
   trip_id;
```

Completed by Jesus Acevedo Delatorre on April 2024.