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.

```
SELECT
    c.company_name,                         -- Select the company name
    COUNT(t.trip_id) AS trips_amount        -- Count the number of rides (trips) and name it
trips_amount
FROM
    cabs c                                  -- Assuming 'cabs' is the table for taxi companies
INNER JOIN
    trips t ON c.cab_id = t.cab_id          -- Join with the trips table
WHERE
    CAST(t.start_ts AS date) BETWEEN '2017-11-15' AND '2017-11-16'  -- Filter for the specified
dates
GROUP BY
    c.company_name                          -- Group by company name
ORDER BY
    trips_amount DESC;                      -- Sort by trips_amount in descending order
```

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.

```
SELECT
    c.company_name,                         -- Select the company name
    COUNT(t.trip_id) AS trips_amount        -- Count the number of rides (trips) and name it
trips_amount
FROM
    cabs c                                  -- Assuming 'cabs' is the table for taxi companies
INNER JOIN
    trips t ON c.cab_id = t.cab_id          -- Join with the trips table
WHERE
    (c.company_name LIKE '%Yellow%' OR c.company_name LIKE '%Blue%')  -- Filter for
company names containing "Yellow" or "Blue"
    AND CAST(t.start_ts AS date) BETWEEN '2017-11-01' AND '2017-11-07'  -- Filter for the
specified dates
GROUP BY
```

```
    c.company_name                      -- Group by company name
ORDER BY
    trips_amount DESC;                  -- Sort by trips_amount in descending order (optional)
```

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 c.company_name IN ('Flash Cab', 'Taxi Affiliation Services') THEN
c.company_name
        ELSE 'Other'  -- Group all other companies as "Other"
    END AS company,  -- Name the field with taxi company names
    COUNT(t.trip_id) AS trips_amount  -- Count the number of rides and name it trips_amount
FROM
    cabs c
INNER JOIN
    trips t ON c.cab_id = t.cab_id  -- Join with the trips table
WHERE
    CAST(t.start_ts AS date) BETWEEN '2017-11-01' AND '2017-11-07'  -- Filter for the specified
dates
GROUP BY
    company  -- Group by the company field
ORDER BY
    trips_amount DESC;  -- Sort the result in descending order by trips_amount
```

4.

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

```
SELECT
    neighborhood_id,  -- Select the neighborhood identifiers
    name              -- Include the neighborhood names for clarity
FROM
    neighborhoods
```

```
WHERE
   name IN ('O''Hare', 'Loop');  -- Filter for O'Hare and Loop neighborhoods
```

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*.

```
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.*

```
SELECT
   t.start_ts,                              -- Ride start timestamp
   CASE
      WHEN LOWER(w.description) LIKE '%rain%' OR LOWER(w.description) LIKE '%storm%'
THEN 'Bad'  -- Group as Bad
      ELSE 'Good'                           -- Group as Good for other conditions
   END AS weather_conditions,                     -- Weather conditions
   t.duration_seconds                        -- Ride duration in seconds
FROM
   trips t
JOIN
   weather_records w ON DATE(t.start_ts) = DATE(w.ts) AND EXTRACT(HOUR FROM
t.start_ts) = EXTRACT(HOUR FROM w.ts)  -- Join on date and hour
WHERE
   t.pickup_location_id = 50                    -- Filter for Loop pickup
   AND t.dropoff_location_id = 63                -- Filter for O'Hare dropoff
   AND EXTRACT(DOW FROM t.start_ts) = 6          -- Only Saturdays (0 = Sunday, 1 =
Monday, ..., 6 = Saturday)
ORDER BY
   t.trip_id;                               -- Sort by trip_id
```