

```

CREATE TABLE `yellowcabsNYC.merged_2020` AS
SELECT * FROM `yellowcabsNYC.jan2020`
UNION ALL
SELECT * FROM `yellowcabsNYC.feb2020`
UNION ALL
SELECT * FROM `yellowcabsNYC.march2020`
UNION ALL
SELECT * FROM `yellowcabsNYC.april2020`
UNION ALL
SELECT * FROM `yellowcabsNYC.may2020`
UNION ALL
SELECT * FROM `yellowcabsNYC.june2020`
UNION ALL
SELECT * FROM `yellowcabsNYC.july2020`
UNION ALL
SELECT * FROM `yellowcabsNYC.august2020`
UNION ALL
SELECT * FROM `yellowcabsNYC.sep2020`
UNION ALL
SELECT * FROM `yellowcabsNYC.oct2020`
UNION ALL
SELECT * FROM `yellowcabsNYC.nov2020`
UNION ALL
SELECT * FROM `yellowcabsNYC.dec2020`;

```

1) For the first four months of 2020, find the total fare amount (one record) for yellow taxi trips taken, charged at the standard rate, with 2 or more passengers who paid with a credit card or cash.

```

SELECT
    SUM(fare_amount) AS total_fare_amount
FROM

```

```

`yellowcabsNYC.merged_2020`
WHERE
    EXTRACT(YEAR FROM tpep_pickup_datetime) = 2020
    AND EXTRACT(MONTH FROM tpep_pickup_datetime) BETWEEN 1 AND 4
    AND RateCodeID = 1
    AND passenger_count >= 2
    AND payment_type IN (1, 2);

```

The screenshot shows a SQL query editor with a toolbar at the top containing icons for home, close, search, and tabs for '\*Untitled', '\*Untitled 2', and 'merged\_2020'. Below the toolbar, the query is displayed in a monospace font with line numbers 1 through 10 on the left. The query is identical to the one in the previous block.

Query results

JOB INFORMATION		RESULTS
Row	total_fare_amount	
1	43634829.14000...	

2) What was the average fare amount per mile of a yellow taxi standard rate ride in 2020 travelled more than 8 miles but less than 20 miles (excluding trips made to airports i.e. RateCodeID 2, 3 and 4)? (Result should be 1 record).

```

SELECT
    AVG(fare_amount / trip_distance) AS average_fare_per_mile
FROM

```

```

`yellowcabsNYC.merged_2020`
WHERE
  EXTRACT(YEAR FROM tpep_pickup_datetime) = 2020
  AND trip_distance > 8
  AND trip_distance < 20
  AND RateCodeID = 1;

```

2
 

Untitled
 

RUN
 SAVE
 SHARE
 SCHE

```

1 SELECT
2   AVG(fare_amount / trip_distance) AS average_fare_per_mile
3 FROM
4   `yellowcabsNYC.merged_2020`
5 WHERE
6   EXTRACT(YEAR FROM tpep_pickup_datetime) = 2020
7   AND trip_distance > 8
8   AND trip_distance < 20
9   AND RateCodeID = 1;

```

## Query results

JOB INFORMATION		RESULTS	CHART	PREVIEW
Row	average_fare_per_mj			
1	3.091674939942...			

3) Which day of the week (from Monday-Sunday) in 2020 has the lowest number of single rider trips on average? (Result should be one record)

```

WITH SingleRiderTrips AS (
  SELECT
    DATE(tpep_pickup_datetime) AS pickup_date,

```

```

        CASE WHEN passenger_count = 1 THEN 1 ELSE 0 END AS
is_single_rider
    FROM
        `demo1-401820.yellowcabsNYC.merged_2020`
    WHERE
        EXTRACT(YEAR FROM tpep_pickup_datetime) = 2020
)
SELECT
    EXTRACT(DAYOFWEEK FROM pickup_date) AS day_of_week,
    AVG(is_single_rider) AS average_single_rider_trips
FROM
    SingleRiderTrips
GROUP BY
    day_of_week
ORDER BY
    average_single_rider_trips ASC
LIMIT 1;

```

## Query results

JOB INFORMATION		RESULTS	CHART	PREVIEW	JSON
Row	day_of_week	average_single_rider			
1	7	0.673144278937...			

**Untitled**
▶ RUN
📄 SAVE ▼
👤 SHARE ▼
🕒 SCHEDULE

```

1 WITH trip_counts AS (
2     SELECT
3         EXTRACT(DAYOFWEEK FROM tpep_pickup_datetime) AS pickup_day,
4         COUNT(*) AS trip_count
5     FROM
6         `yellowcabsNYC.merged_2020`
7     WHERE
8         EXTRACT(YEAR FROM tpep_pickup_datetime) = 2020
9         AND passenger_count = 1
10    GROUP BY
11        pickup_day
12 )
13 SELECT
14     CASE
15         WHEN pickup_day = 1 THEN 'Sunday'
16         WHEN pickup_day = 2 THEN 'Monday'
17         WHEN pickup_day = 3 THEN 'Tuesday'
18         WHEN pickup_day = 4 THEN 'Wednesday'
19         WHEN pickup_day = 5 THEN 'Thursday'
20         WHEN pickup_day = 6 THEN 'Friday'
21         WHEN pickup_day = 7 THEN 'Saturday'
22     END AS day_with_lowest_single_rider_trips
23 FROM
24     trip_counts
25 ORDER BY
26     trip_count ASC
27 LIMIT 1;

```

## Query results

JOB INFORMATION		RESULTS
Row	day_with_lowest_single_rider_trips	
1	Sunday	

4) ETL: Create new dataset from Jan 2020 data which includes the following columns

```
SELECT
```

```

DATE(tpep_pickup_datetime) AS pickup_date,
TIME(tpep_pickup_datetime) AS pickup_time,
TIME(tpep_dropoff_datetime) AS dropoff_time,
trip_distance,
RatecodeID,
fare_amount,
IF(trip_distance > 0, total_amount / trip_distance, NULL) AS
fare_per_mile
FROM
    `yellowcabsNYC.merged_2020`
WHERE
    tolls_amount = 0
    AND trip_distance < 30
    AND tpep_pickup_datetime IS NOT NULL
    AND tpep_dropoff_datetime IS NOT NULL
    AND trip_distance IS NOT NULL
    AND RatecodeID IS NOT NULL
    AND fare_amount IS NOT NULL
    AND fare_amount > 0
    AND trip_distance > 0;

```

```

(Another way):
WITH January2020Trips AS (
    SELECT
        DATE(tpep_pickup_datetime) AS pickup_date,
        TIME(tpep_pickup_datetime) AS pickup_time,
        TIME(tpep_dropoff_datetime) AS dropoff_time,
        trip_distance,
        RatecodeID,
        fare_amount,
        CASE
            WHEN trip_distance > 0 THEN total_amount /
trip_distance
            ELSE NULL
        END AS fare_per_mile
    FROM
        `demo1-401820.yellowcabsNYC.merged_2020`
    WHERE

```

```
        EXTRACT(YEAR FROM tpep_pickup_datetime) = 2020
        AND EXTRACT(MONTH FROM tpep_pickup_datetime) = 1
        AND tolls_amount = 0
        AND trip_distance < 30
    )
SELECT *
FROM January2020Trips;
```

Truncated version:

```
[{
  "pickup_date": "2020-10-24",
  "pickup_time": "00:17:07",
  "dropoff_time": "00:38:01",
  "trip_distance": "6.7",
  "RatecodeID": "1.0",
  "fare_amount": "21.0",
  "fare_per_mile": "3.7014925373134329"
}, {
  "pickup_date": "2020-08-09",
  "pickup_time": "00:47:26",
  "dropoff_time": "01:10:50",
  "trip_distance": "6.08",
  "RatecodeID": "1.0",
  "fare_amount": "21.0",
  "fare_per_mile": "4.5723684210526319"
}, {
  "pickup_date": "2020-02-27",
  "pickup_time": "22:16:19",
  "dropoff_time": "22:41:03",
  "trip_distance": "7.0",
  "RatecodeID": "1.0",
  "fare_amount": "24.0",
  "fare_per_mile": "4.6857142857142851"
}, {
  "pickup_date": "2020-11-20",
  "pickup_time": "21:57:18",
  "dropoff_time": "22:28:06",
  "trip_distance": "6.77",
```

```
"RatecodeID": "1.0",
"fare_amount": "25.0",
"fare_per_mile": "5.104874446085673"
}, {
  "pickup_date": "2020-11-11",
  "pickup_time": "10:35:06",
  "dropoff_time": "10:56:02",
  "trip_distance": "8.03",
  "RatecodeID": "1.0",
  "fare_amount": "25.0",
  "fare_per_mile": "4.5815691158156913"
}, {
  "pickup_date": "2020-01-07",
  "pickup_time": "13:53:14",
  "dropoff_time": "14:13:08",
  "trip_distance": "8.57",
  "RatecodeID": "1.0",
  "fare_amount": "25.0",
  "fare_per_mile": "3.6126021003500584"
},
```

5) In this assignment, I learned how to use SQL queries to work with a dataset. I understood how to filter and transform data, which is crucial in dealing with real-world datasets. Debugging SQL queries was a challenge, requiring careful attention to detail to get the right results. The assignment took about three hours, including understanding the task, writing queries, and making them accurate. It would be helpful to practice complex queries in class before doing them as homework.



