Lab 3 Anna Ślęzak

1. Wykonaj zapytania (3p):
    1. Sprawdź dla każdego dnia tygodnia w jakiej godzinie jest najwięcej branych taksówek.

```sql
with ranked_hours as (
    select day_of_week(tpep_pickup_datetime) as day_of_week,
        split_part(
            split_part(cast(tpep_pickup_datetime as varchar), ' ', 2),
            ':',
            1
        ) as hour,
        count(*) as no_of_rides,
        row_number() over (
            partition by day_of_week(tpep_pickup_datetime)
            order by count(*) desc
        ) as rank
    from TaxiDataYellow
    group by day_of_week(tpep_pickup_datetime),
        split_part(
            split_part(cast(tpep_pickup_datetime as varchar), ' ', 2),
            ':',
            1
        )
)
select day_of_week,
    hour,
    no_of_rides
from ranked_hours
where rank = 1;
```

    2. Znajdź przejazdy które znacząco odbiegają od ceny standardowej (outliery)

```sql
with fare_stats as (
    select avg(fare_amount) as avg_fare,
        stddev(fare_amount) as stddev_fare
    from taxidatayellow
)
select *
from taxidatayellow
    cross join fare_stats
where fare_amount < avg_fare - 1.5 * stddev_fare
    or fare_amount > avg_fare + 1.5 * stddev_fare;
```

\

    3. Znajdź średnią cenę przejazdu per osoba dla każdej firmy w zależności od liczby osób (z przedziału 1-7) w samochodzie, zlicz też liczbę przejazdów dla każdej ceny razem z procentowym udziałem we wszystkich przejazdach danej firmy.

```sql
SELECT vendorid,
    passenger_count,
    AVG(fare_amount / passenger_count) AS avg_price_per_person,
    COUNT(*) AS ride_count,
    (COUNT(*) * 100.0) / SUM(COUNT(*)) OVER (PARTITION BY vendorid) AS percentage_of_total_rides
FROM TaxiDataYellow
WHERE passenger_count BETWEEN 1 AND 7
GROUP BY vendorid,
    passenger_count;
```

2. Znajdź własny dataset w formacie innym niż Parquet, skonwertuj go do formatu Parquet, a następnie wykonaj własne nietrywialne zapytanie na obu formatach z wykorzystaniem AWS Athena i sprawdź które zapytanie działa szybciej, które procesuje więcej danych. Sprawdź równoważność zwróconych wyników. Do konwersji wystarczą 2 zapytania.
   (2p) https://docs.aws.amazon.com/athena/latest/ug/ctas-examples.html#ctas-example-format

Tabela w formacie csv:

```sql
CREATE EXTERNAL TABLE IF NOT EXISTS dataset_flight (
  airline STRING,
  date_of_journey STRING,
  source STRING,
  destination STRING,
  route STRING,
  dep_time STRING,
  arrival_time STRING,
  duration STRING,
  total_stops STRING,
  additional_info STRING,
  price bigint
)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
STORED AS TEXTFILE
LOCATION 's3://844124137610-us-east-1-athena-results-bucket-qhv015up15/dataset_flights/'
```

Tabela w formacie parquet:

```sql
CREATE TABLE flights_parquet
WITH (
      format = 'Parquet',
      write_compression = 'SNAPPY')
AS SELECT *
FROM dataset_flight;

select * from flights_parquet
```

Wybór najpopularniejszej trasy w danym miesiącu dla określonej linii lotniczej oraz średnia cena biletu dla tej trasy.

CSV:

```sql
WITH MonthlyAverages AS (
    SELECT
        airline,
        split_part(date_of_journey, '/', 2) AS travel_month,
        destination,
        AVG(price) AS avg_price,
        RANK() OVER (PARTITION BY airline, split_part(date_of_journey, '/', 2) ORDER BY COUNT(*) DESC) AS rank
    FROM
        dataset_flight
    GROUP BY
        1,2,3
)

SELECT
    airline,
    travel_month,
    destination,
    avg_price
FROM
    MonthlyAverages
WHERE
    rank = 1
ORDER BY
    airline,
    travel_month;

```

Completed | Time in queue: 127 ms | Run time: 760 ms | Data scanned: 1.26 MB

Results (38)

Copy | Download results

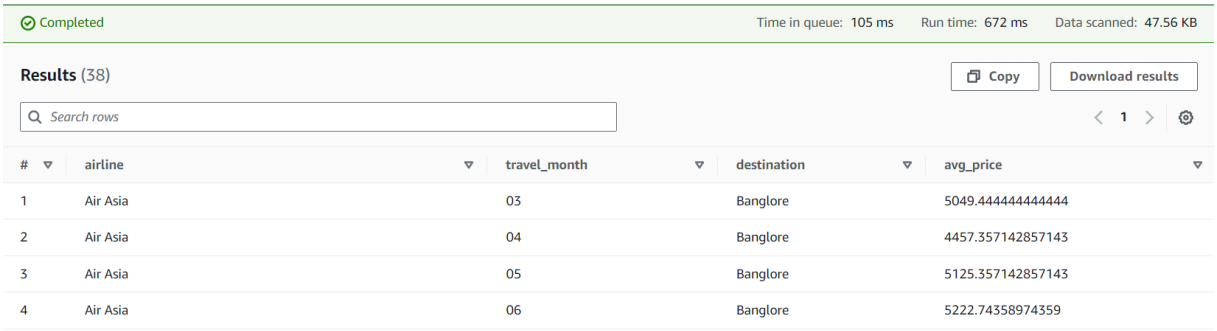
Search rows

1

#	airline	travel_month	destination	avg_price
1	Air Asia	03	Banglore	5049.444444444444
2	Air Asia	04	Banglore	4457.357142857143
3	Air Asia	05	Banglore	5125.357142857143
4	Air Asia	06	Banglore	5222.74358974359

PARQUET:

```sql
WITH MonthlyAverages AS (
    SELECT airline,
        split_part(date_of_journey, '/', 2) AS travel_month,
        destination,
        AVG(price) AS avg_price,
        RANK() OVER (
            PARTITION BY airline,
            split_part(date_of_journey, '/', 2)
            ORDER BY COUNT(*) DESC
        ) AS rank
    FROM flights_parquet
    GROUP BY 1,
        2,
        3
)
SELECT airline,
    travel_month,
    destination,
    avg_price
FROM MonthlyAverages
WHERE rank = 1
ORDER BY airline,
    travel_month;
```



#	airline	travel_month	destination	avg_price
1	Air Asia	03	Banglore	5049.444444444444
2	Air Asia	04	Banglore	4457.357142857143
3	Air Asia	05	Banglore	5125.357142857143
4	Air Asia	06	Banglore	5222.74358974359