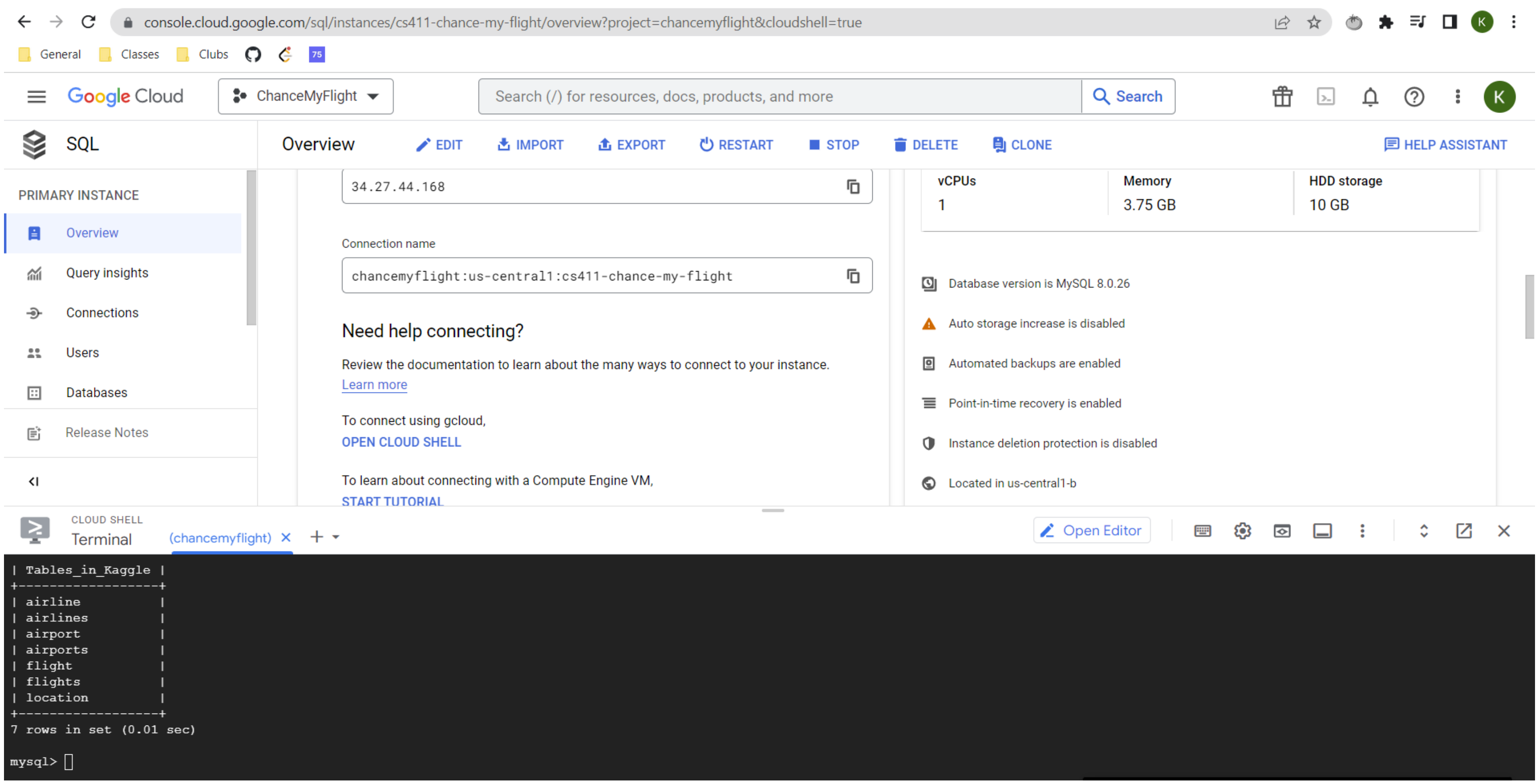
Database Connection



34.27.44.168

Connection name

chancemyflight:us-central1:cs411-chance-my-flight

Need help connecting?

Review the documentation to learn about the many ways to connect to your instance.
Learn more

To connect using gcloud,
OPEN CLOUD SHELL

To learn about connecting with a Compute Engine VM,
START TUTORIAL

vCPUs	Memory	HDD storage
1	3.75 GB	10 GB

Database version is MySQL 8.0.26

Auto storage increase is disabled

Automated backups are enabled

Point-in-time recovery is enabled

Instance deletion protection is disabled

Located in us-central1-b

```
| Tables_in_Kaggle |
+------------------+
| airline          |
| airlines         |
| airport          |
| airports         |
| flight           |
| flights          |
| location         |
+------------------+
7 rows in set (0.01 sec)

mysql> 
```

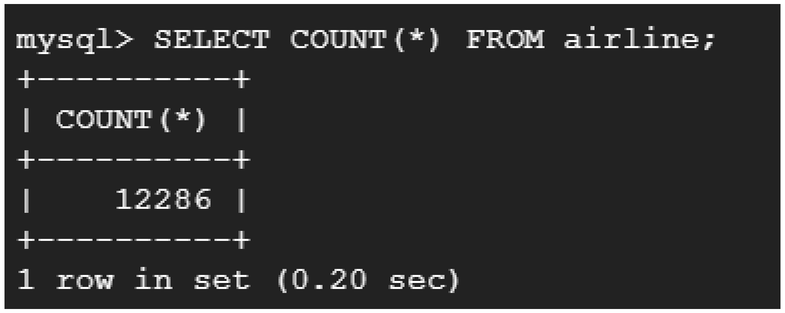
DDL Commands

USE Kaggle;
DROP TABLE IF EXISTS location;
CREATE TABLE location (
ID INT NOT NULL AUTO_INCREMENT,

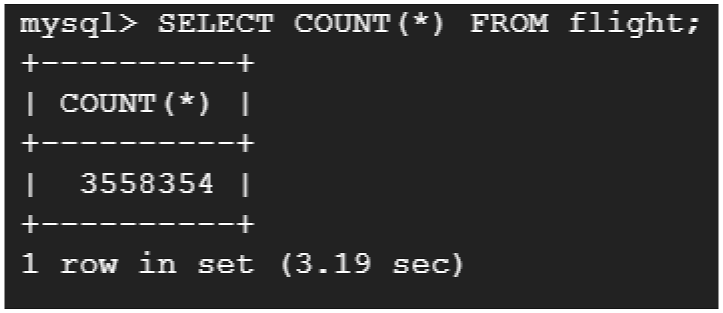
```
PRIMARY KEY(ID),
City TEXT,
State VARCHAR(2),
AirportName TEXT);
INSERT INTO location (City, State, AirportName)
SELECT CITY, STATE, AIRPORT
FROM airports;

DROP TABLE IF EXISTS airport;
CREATE TABLE airport (
PRIMARY KEY(ID),
ID INT NOT NULL AUTO_INCREMENT,
IATA_code VARCHAR(3),
LocationID INT,
FOREIGN KEY (LocationID) REFERENCES location(ID),
ON DELETE CASCADE,
Name TEXT,
Latitude INT,
Longitude INT);
INSERT INTO airport (IATA_code, Name, Latitude, Longitude)
SELECT IATA_CODE, AIRPORT, LATITUDE, LONGITUDE
FROM airports;
UPDATE airport
INNER JOIN location ON (location.AirportName = airport.Name)
SET airport.LocationID = location.ID;

DROP TABLE IF EXISTS airline;
CREATE TABLE airline (
PRIMARY KEY(ID),
ID INT NOT NULL AUTO_INCREMENT,
name TEXT,
Origin Text,
Destination Text);
INSERT INTO airline (name, Origin, Destination)
SELECT DISTINCT AIRLINE, ORIGIN_AIRPORT, DESTINATION_AIRPORT
FROM flights;


DROP TABLE IF EXISTS flight;
CREATE TABLE flight (
PRIMARY KEY(ID),
ID INT NOT NULL AUTO_INCREMENT,
FlightID TEXT, Airline TEXT,
DestinationAirport TEXT,
OriginAirport TEXT,
ArrivalDelay INT,
DepartureDelay INT);

INSERT INTO flight (FlightID, Airline, DestinationAirport, OriginAirport, ArrivalDelay, DepartureDelay)
SELECT CONCAT(FLIGHT_NUMBER, TAIL_NUMBER), AIRLINE, DESTINATION_AIRPORT, ORIGIN_AIRPORT,
ARRIVAL_DELAY, DEPARTURE_DELAY FROM flights;

ALTER TABLE airline MODIFY COLUMN name VARCHAR(100);
ALTER TABLE airline ADD INDEX name_idx(name);

DROP TABLE IF EXISTS review;
CREATE TABLE review (
  Review_ID INT NOT NULL AUTO_INCREMENT,
  PRIMARY KEY(Review_ID),
  Text TEXT NOT NULL,
  airline_name VARCHAR(100) NOT NULL,
  CONSTRAINT fk_airline_name FOREIGN KEY (airline_name) REFERENCES airline(name)
);
```

```
INSERT INTO review (text, airline_name)
SELECT
  CONCAT('This is review #', t.n) AS review_text,
  a.name AS airline_name
FROM
  (SELECT @row := @row + 1 AS n FROM (SELECT 1 UNION SELECT 2 UNION SELECT 3 UNION SELECT 4) t1, (SELECT 1
UNION SELECT 2 UNION SELECT 3 UNION SELECT 4) t2, (SELECT 1 UNION SELECT 2 UNION SELECT 3 UNION SELECT 4)
t3, (SELECT @row:=0) r) t
  CROSS JOIN airline a
ORDER BY RAND()
LIMIT 1000;
```

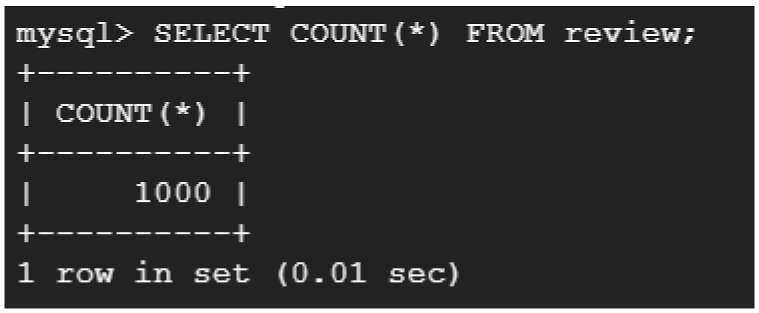
<u>1000 Rows in 3 Tables</u>



```
mysql> SELECT COUNT(*) FROM airline;
+----------+
| COUNT(*) |
+----------+
|    12286 |
+----------+
1 row in set (0.20 sec)
```



```
mysql> SELECT COUNT(*) FROM flight;
+----------+
| COUNT(*) |
+----------+
|  3558354 |
+----------+
1 row in set (3.19 sec)
```


```
mysql> SELECT COUNT(*) FROM review;
+----------+
| COUNT(*) |
+----------+
|     1000 |
+----------+
1 row in set (0.01 sec)
```

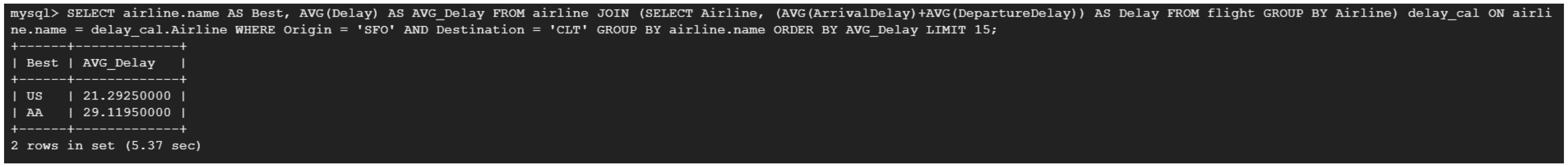
<u>Advanced Queries</u>

For now we have chosen airports for our queries, but in the future they will depend on user input

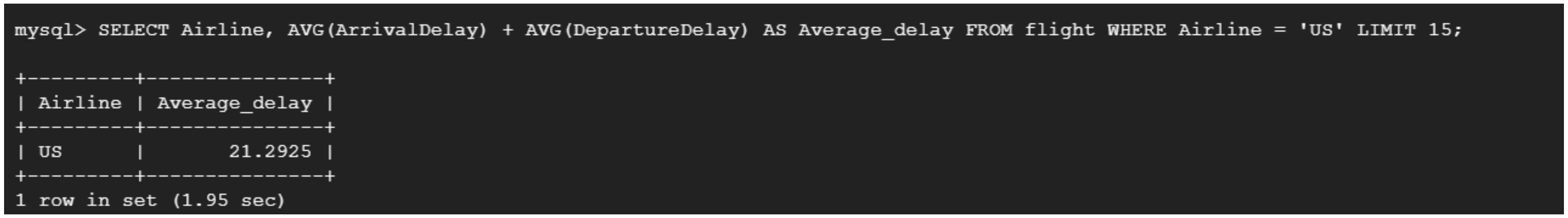
```
SELECT airline.name AS Best, AVG(Delay) AS AVG_Delay
FROM airline JOIN (SELECT Airline, (AVG(ArrivalDelay)+AVG(DepartureDelay)) AS Delay
                   FROM flight
                   GROUP BY Airline) delay_cal ON airline.name = delay_cal.Airline
WHERE Origin = 'SFO' AND Destination = 'CLT'
GROUP BY airline.name
ORDER BY AVG_Delay
LIMIT 1;

SELECT Airline, AVG(ArrivalDelay) + AVG(DepartureDelay) AS Average_delay
FROM flight
WHERE Airline = 'US';

-- Given two cities and return the average delay time between these two cities.
SELECT flight.Airline, AVG(ArrivalDelay) + AVG(DepartureDelay) AS Average_delay
FROM flight JOIN airline ON (flight.Airline = airline.name)
WHERE airline.Origin = 'SFO' AND airline.Destination = 'CLT'
GROUP BY flight.Airline;
```

```
-- Given two airports and return the average delay time between these two airports.
SELECT AVG(ArrivalDelay) + AVG(DepartureDelay) AS Average_delay
FROM flight
WHERE OriginAirport = 'Lehigh Valley International Airport' AND DestinationAirport = 'Waco Regional Airport';
```

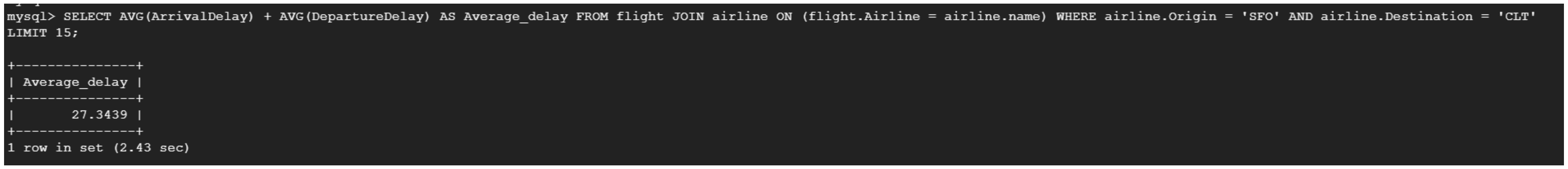
Top 15 Rows of Advanced Queries



```
mysql> SELECT airline.name AS Best, AVG(Delay) AS AVG_Delay FROM airline JOIN (SELECT Airline, (AVG(ArrivalDelay)+AVG(DepartureDelay)) AS Delay FROM flight GROUP BY Airline) delay_cal ON airli
ne.name = delay_cal.Airline WHERE Origin = 'SFO' AND Destination = 'CLT' GROUP BY airline.name ORDER BY AVG_Delay LIMIT 15;
+------+-------------+
| Best | AVG_Delay   |
+------+-------------+
| US   | 21.29250000 |
| AA   | 29.11950000 |
+------+-------------+
2 rows in set (5.37 sec)
```



```
mysql> SELECT Airline, AVG(ArrivalDelay) + AVG(DepartureDelay) AS Average_delay FROM flight WHERE Airline = 'US' LIMIT 15;

+---------+---------------+
| Airline | Average_delay |
+---------+---------------+
| US      |       21.2925 |
+---------+---------------+
1 row in set (1.95 sec)
```



```
mysql> SELECT AVG(ArrivalDelay) + AVG(DepartureDelay) AS Average_delay FROM flight JOIN airline ON (flight.Airline = airline.name) WHERE airline.Origin = 'SFO' AND airline.Destination = 'CLT'
LIMIT 15;

+---------------+
| Average_delay |
+---------------+
|       27.3439 |
+---------------+
1 row in set (2.43 sec)
```

Indexing

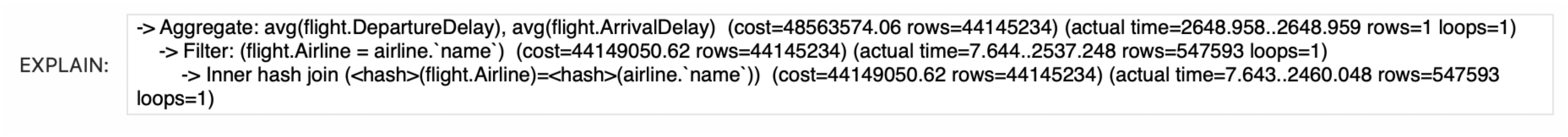
QUERY 1:

SELECT AVG(ArrivalDelay) + AVG(DepartureDelay) AS Average_delay
FROM flight JOIN airline ON (flight.Airline = airline.name)
WHERE airline.Origin = 'SFO' AND airline.Destination = 'CLT';

Best Index Design and Reason: CREATE INDEX airline_name_idx ON airline(Origin, Destination, name(50))

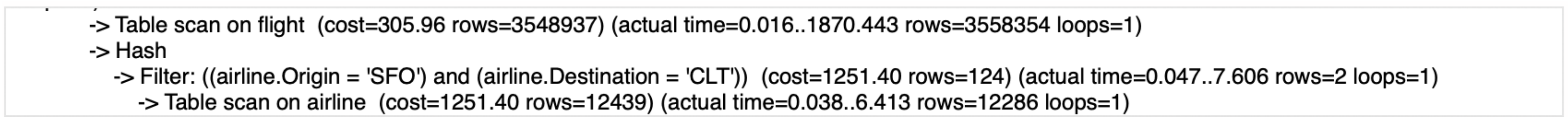
- I noticed that with all the index designs, the cost was significantly lowered from the total cost before indexing. While flight_airline_idx produced the fastest total time after indexing, airline_name_idx had the lowest cost. Since the time can also be affected by other various factors and the increase was negligible from the total time before indexing, I think the cost reduction is the most important factor when deciding between the two index designs. In addition airline_name_idx processes fewer rows, so I believe airline_name_idx is the most efficient design overall.

Total Time Before Indexing: 2648.959
Total Cost Before Indexing: 48563574.06



EXPLAIN:

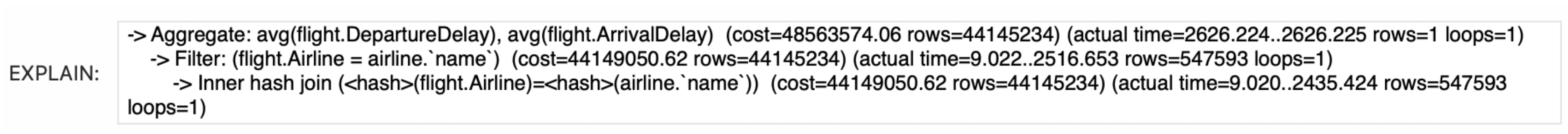
```
-> Aggregate: avg(flight.DepartureDelay), avg(flight.ArrivalDelay)  (cost=48563574.06 rows=44145234) (actual time=2648.958..2648.959 rows=1 loops=1)
    -> Filter: (flight.Airline = airline.`name`)  (cost=44149050.62 rows=44145234) (actual time=7.644..2537.248 rows=547593 loops=1)
        -> Inner hash join (<hash>(flight.Airline)=<hash>(airline.`name`))  (cost=44149050.62 rows=44145234) (actual time=7.643..2460.048 rows=547593
loops=1)
```



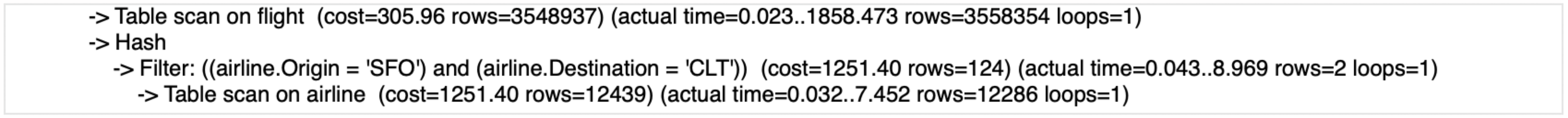
```
            -> Table scan on flight  (cost=305.96 rows=3548937) (actual time=0.016..1870.443 rows=3558354 loops=1)
            -> Hash
                -> Filter: ((airline.Origin = 'SFO') and (airline.Destination = 'CLT'))  (cost=1251.40 rows=124) (actual time=0.047..7.606 rows=2 loops=1)
                    -> Table scan on airline  (cost=1251.40 rows=12439) (actual time=0.038..6.413 rows=12286 loops=1)
```

Indexes Added:

- CREATE INDEX airline_name_idx ON airline(name(50));
    - **Total Time:** 2626.225
    - **Total Cost:** 48563574.06

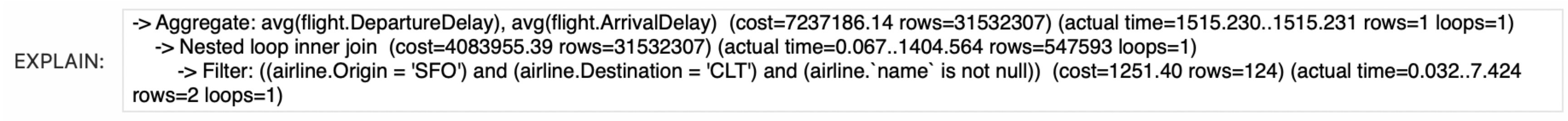

EXPLAIN:

```
-> Aggregate: avg(flight.DepartureDelay), avg(flight.ArrivalDelay)  (cost=48563574.06 rows=44145234) (actual time=2626.224..2626.225 rows=1 loops=1)
    -> Filter: (flight.Airline = airline.`name`)  (cost=44149050.62 rows=44145234) (actual time=9.022..2516.653 rows=547593 loops=1)
        -> Inner hash join (<hash>(flight.Airline)=<hash>(airline.`name`))  (cost=44149050.62 rows=44145234) (actual time=9.020..2435.424 rows=547593
loops=1)
```

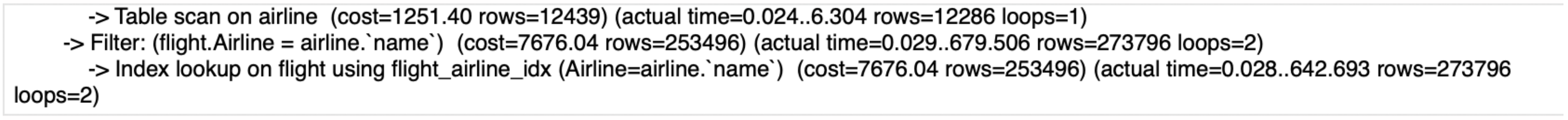


```
            -> Table scan on flight  (cost=305.96 rows=3548937) (actual time=0.023..1858.473 rows=3558354 loops=1)
            -> Hash
                -> Filter: ((airline.Origin = 'SFO') and (airline.Destination = 'CLT'))  (cost=1251.40 rows=124) (actual time=0.043..8.969 rows=2 loops=1)
                    -> Table scan on airline  (cost=1251.40 rows=12439) (actual time=0.032..7.452 rows=12286 loops=1)
```

- CREATE INDEX flight_airline_idx ON flight(Airline(50));
    - **Total Time:** 1515.231
    - **Total Cost:** 4083955.39

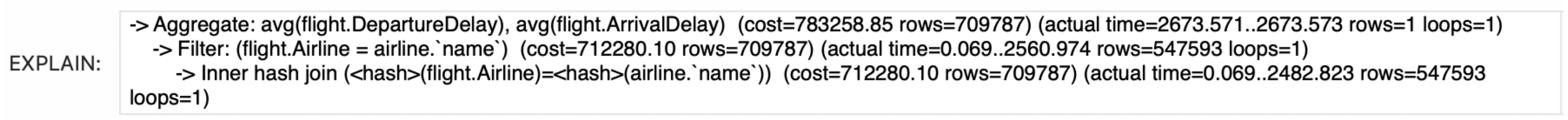

EXPLAIN:

```
-> Aggregate: avg(flight.DepartureDelay), avg(flight.ArrivalDelay)  (cost=7237186.14 rows=31532307) (actual time=1515.230..1515.231 rows=1 loops=1)
    -> Nested loop inner join  (cost=4083955.39 rows=31532307) (actual time=0.067..1404.564 rows=547593 loops=1)
        -> Filter: ((airline.Origin = 'SFO') and (airline.Destination = 'CLT') and (airline.`name` is not null))  (cost=1251.40 rows=124) (actual time=0.032..7.424
rows=2 loops=1)
```

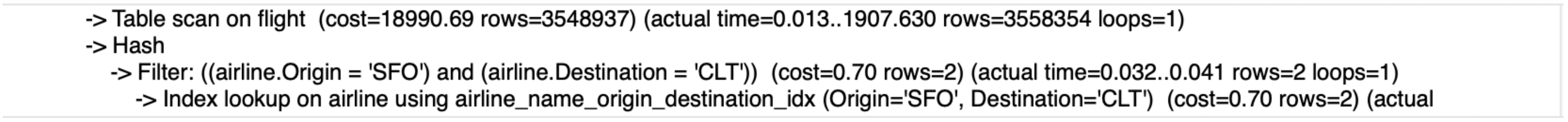


```
            -> Table scan on airline  (cost=1251.40 rows=12439) (actual time=0.024..6.304 rows=12286 loops=1)
        -> Filter: (flight.Airline = airline.`name`)  (cost=7676.04 rows=253496) (actual time=0.029..679.506 rows=273796 loops=2)
            -> Index lookup on flight using flight_airline_idx (Airline=airline.`name`)  (cost=7676.04 rows=253496) (actual time=0.028..642.693 rows=273796
loops=2)
```

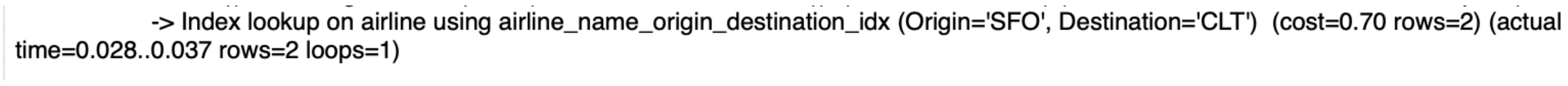
- CREATE INDEX airline_name_idx ON airline(Origin, Destination, name(50));
    - **Total Time:** 2673.573
    - **Total Cost:** 783258.85


EXPLAIN:

```
-> Aggregate: avg(flight.DepartureDelay), avg(flight.ArrivalDelay)  (cost=783258.85 rows=709787) (actual time=2673.571..2673.573 rows=1 loops=1)
    -> Filter: (flight.Airline = airline.`name`)  (cost=712280.10 rows=709787) (actual time=0.069..2560.974 rows=547593 loops=1)
        -> Inner hash join (<hash>(flight.Airline)=<hash>(airline.`name`))  (cost=712280.10 rows=709787) (actual time=0.069..2482.823 rows=547593
loops=1)
```



```
            -> Table scan on flight  (cost=18990.69 rows=3548937) (actual time=0.013..1907.630 rows=3558354 loops=1)
            -> Hash
                -> Filter: ((airline.Origin = 'SFO') and (airline.Destination = 'CLT'))  (cost=0.70 rows=2) (actual time=0.032..0.041 rows=2 loops=1)
                    -> Index lookup on airline using airline_name_origin_destination_idx (Origin='SFO', Destination='CLT')  (cost=0.70 rows=2) (actual
```



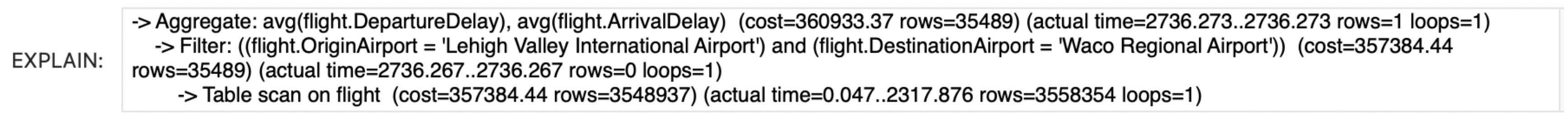
```
                 -> Index lookup on airline using airline_name_origin_destination_idx (Origin='SFO', Destination='CLT')  (cost=0.70 rows=2) (actual
time=0.028..0.037 rows=2 loops=1)
```

<u>Query 2:</u>
SELECT AVG(ArrivalDelay) + AVG(DepartureDelay) AS Average_delay
FROM flight
WHERE OriginAirport = 'Lehigh Valley International Airport' AND DestinationAirport = 'Waco Regional Airport';

Best Index Design and Reason: CREATE INDEX flight_DAD_idx ON flight(DestinationAirport(100), ArrivalDelay, DepartureDelay);

- While all of the indexing deigns I tried significantly reduced the total time and total cost in comparison to the time and cost before indexing, I noticed that flight_DAD_idx lowered the cost more while the increase in total time in comparison to the other two designs was negligible. This might be because flight_DAD_idx only needs to use the index lookup based on the DestinationAirport column while flight_ODAD_idx needs to use the index lookup on both the DestinationAirport column and OriginAirport column.

Total Time Before Indexing: 2736.226
Total Cost Before Indexing: 360933.37

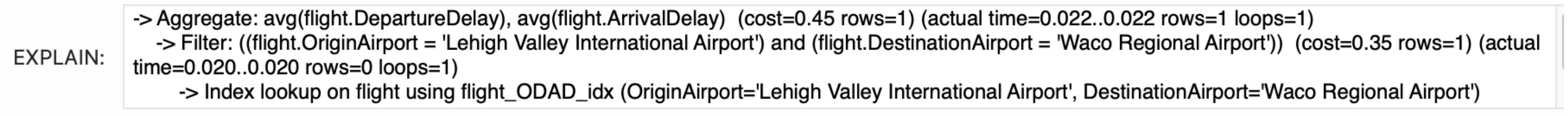


EXPLAIN:

```
-> Aggregate: avg(flight.DepartureDelay), avg(flight.ArrivalDelay)  (cost=360933.37 rows=35489) (actual time=2736.273..2736.273 rows=1 loops=1)
    -> Filter: ((flight.OriginAirport = 'Lehigh Valley International Airport') and (flight.DestinationAirport = 'Waco Regional Airport'))  (cost=357384.44
rows=35489) (actual time=2736.267..2736.267 rows=0 loops=1)
        -> Table scan on flight  (cost=357384.44 rows=3548937) (actual time=0.047..2317.876 rows=3558354 loops=1)
```

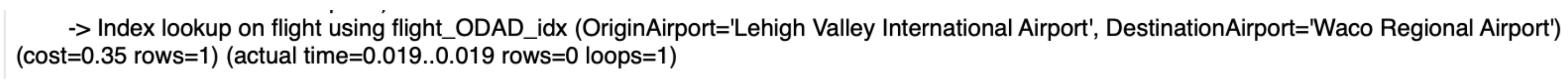
Indexes Added:

- CREATE INDEX flight_ODAD_idx ON flight(OriginAirport(100), DestinationAirport(100), ArrivalDelay, DepartureDelay);
    - **Total Time:** 0.022
    - **Total Cost:** 0.45



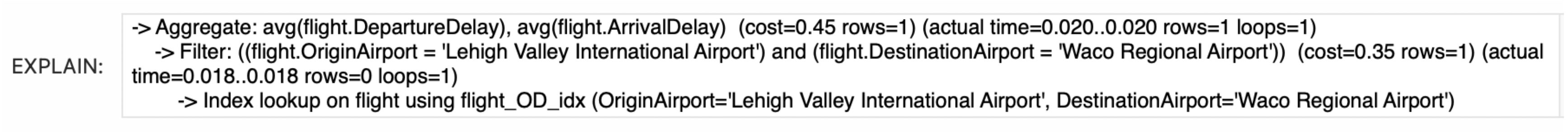
EXPLAIN:

```
-> Aggregate: avg(flight.DepartureDelay), avg(flight.ArrivalDelay)  (cost=0.45 rows=1) (actual time=0.022..0.022 rows=1 loops=1)
    -> Filter: ((flight.OriginAirport = 'Lehigh Valley International Airport') and (flight.DestinationAirport = 'Waco Regional Airport'))  (cost=0.35 rows=1) (actual
time=0.020..0.020 rows=0 loops=1)
        -> Index lookup on flight using flight_ODAD_idx (OriginAirport='Lehigh Valley International Airport', DestinationAirport='Waco Regional Airport')
```

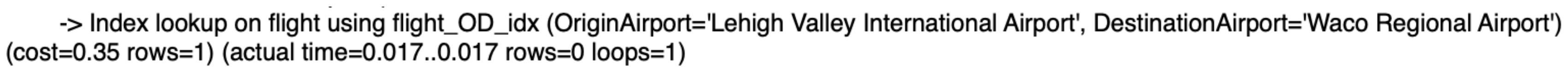


```
    -> Index lookup on flight using flight_ODAD_idx (OriginAirport='Lehigh Valley International Airport', DestinationAirport='Waco Regional Airport')
(cost=0.35 rows=1) (actual time=0.019..0.019 rows=0 loops=1)
```

- CREATE INDEX flight_OD_idx ON flight(OriginAirport(100), DestinationAirport(100));
    - **Total Time:** 0.020
    - **Total Cost:** 0.45

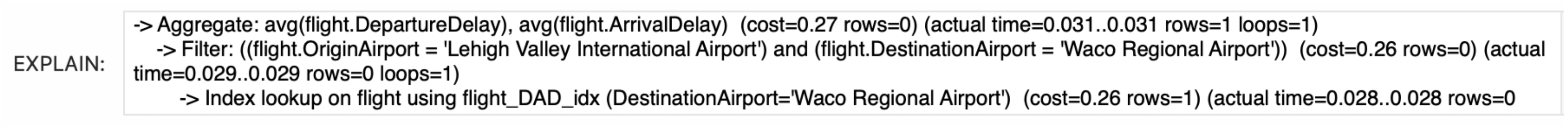


```
EXPLAIN: -> Aggregate: avg(flight.DepartureDelay), avg(flight.ArrivalDelay)  (cost=0.45 rows=1) (actual time=0.020..0.020 rows=1 loops=1)
             -> Filter: ((flight.OriginAirport = 'Lehigh Valley International Airport') and (flight.DestinationAirport = 'Waco Regional Airport'))  (cost=0.35 rows=1) (actual
         time=0.018..0.018 rows=0 loops=1)
                 -> Index lookup on flight using flight_OD_idx (OriginAirport='Lehigh Valley International Airport', DestinationAirport='Waco Regional Airport')
```

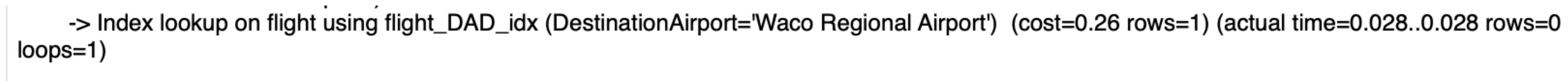


```
    -> Index lookup on flight using flight_OD_idx (OriginAirport='Lehigh Valley International Airport', DestinationAirport='Waco Regional Airport')
(cost=0.35 rows=1) (actual time=0.017..0.017 rows=0 loops=1)
```

- CREATE INDEX flight_DAD_idx ON flight(DestinationAirport(100), ArrivalDelay, DepartureDelay);
    - **Total Time:** 0.031
    - **Total Cost:** 0.27



```
EXPLAIN: -> Aggregate: avg(flight.DepartureDelay), avg(flight.ArrivalDelay)  (cost=0.27 rows=0) (actual time=0.031..0.031 rows=1 loops=1)
             -> Filter: ((flight.OriginAirport = 'Lehigh Valley International Airport') and (flight.DestinationAirport = 'Waco Regional Airport'))  (cost=0.26 rows=0) (actual
         time=0.029..0.029 rows=0 loops=1)
                 -> Index lookup on flight using flight_DAD_idx (DestinationAirport='Waco Regional Airport')  (cost=0.26 rows=1) (actual time=0.028..0.028 rows=0
```



```
    -> Index lookup on flight using flight_DAD_idx (DestinationAirport='Waco Regional Airport')  (cost=0.26 rows=1) (actual time=0.028..0.028 rows=0
loops=1)
```

Query 3:

```
SELECT airline.name AS Best, AVG(Delay) AS AVG_Delay
FROM airline JOIN (SELECT Airline, (AVG(ArrivalDelay)+AVG(DepartureDelay)) AS Delay
                                    FROM flight
                                    GROUP BY Airline) delay_cal ON airline.name = delay_cal.Airline
WHERE Origin = 'SFO' AND Destination = 'CLT'
GROUP BY airline.name
ORDER BY AVG_Delay
LIMIT 1;
```

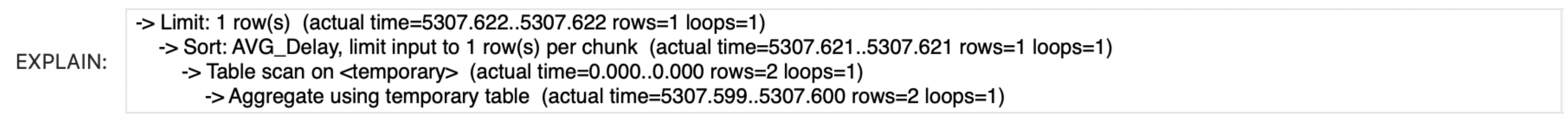
Best Index Design and Reason: CREATE INDEX flight_Airline_Arrival_Departure_idx ON flight(Airline(50), ArrivalDelay, DepartureDelay);

CREATE INDEX flight_Origin_idx ON flight(OriginAirport(50));
CREATE INDEX flight_Destination_idx ON flight(DestinationAirport(50));
CREATE INDEX airline_name_idx ON airline(name(50));

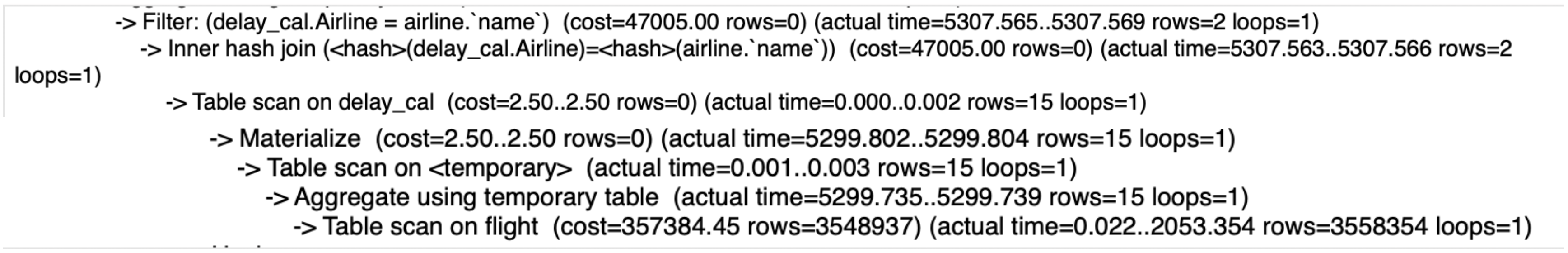
- All three of the index designs were successful in reducing the total time in comparison to the total time before indexing, but only flight_Airline_Departure_idx and flight_ODAAD_idx were successful in lowering the cost. Although flight_Airline_Departure_idx had the same cost as flight_ODAAD_idx, flight_Airline_Departure_idx was slightly faster. This could be because it allows for the database to search for a specific record based on a search condition, which improves the speed of the index finding the most relevant rows.

Total Time Before Indexing: 7360.749

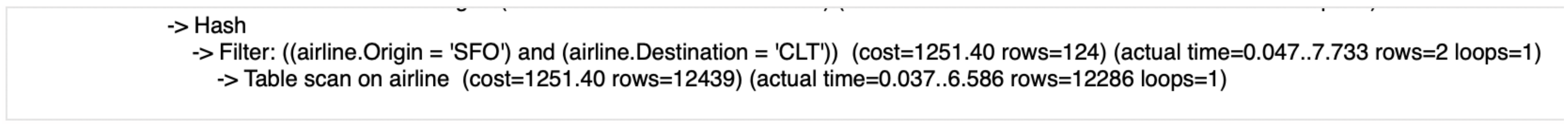
Total Cost Before Indexing: 357384.45



```
EXPLAIN:  -> Limit: 1 row(s)  (actual time=5307.622..5307.622 rows=1 loops=1)
              -> Sort: AVG_Delay, limit input to 1 row(s) per chunk  (actual time=5307.621..5307.621 rows=1 loops=1)
                  -> Table scan on <temporary>  (actual time=0.000..0.000 rows=2 loops=1)
                      -> Aggregate using temporary table  (actual time=5307.599..5307.600 rows=2 loops=1)
```



```
            -> Filter: (delay_cal.Airline = airline.`name`)  (cost=47005.00 rows=0) (actual time=5307.565..5307.569 rows=2 loops=1)
                -> Inner hash join (<hash>(delay_cal.Airline)=<hash>(airline.`name`))  (cost=47005.00 rows=0) (actual time=5307.563..5307.566 rows=2
loops=1)
                    -> Table scan on delay_cal  (cost=2.50..2.50 rows=0) (actual time=0.000..0.002 rows=15 loops=1)
                        -> Materialize  (cost=2.50..2.50 rows=0) (actual time=5299.802..5299.804 rows=15 loops=1)
                            -> Table scan on <temporary>  (actual time=0.001..0.003 rows=15 loops=1)
                                -> Aggregate using temporary table  (actual time=5299.735..5299.739 rows=15 loops=1)
                                    -> Table scan on flight  (cost=357384.45 rows=3548937) (actual time=0.022..2053.354 rows=3558354 loops=1)
```

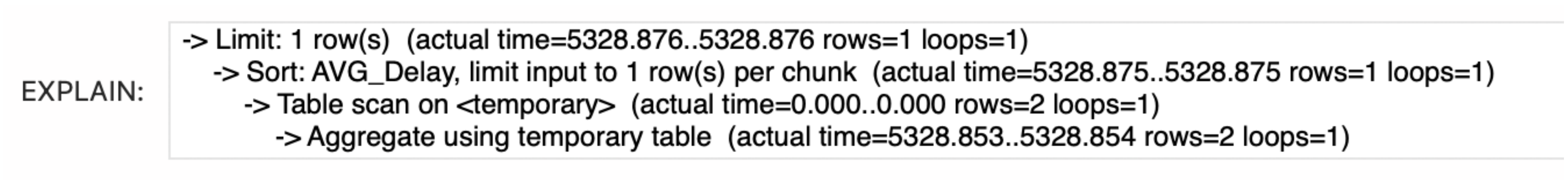


```
                    -> Hash
                        -> Filter: ((airline.Origin = 'SFO') and (airline.Destination = 'CLT'))  (cost=1251.40 rows=124) (actual time=0.047..7.733 rows=2 loops=1)
                            -> Table scan on airline  (cost=1251.40 rows=12439) (actual time=0.037..6.586 rows=12286 loops=1)
```

Indexes Added:

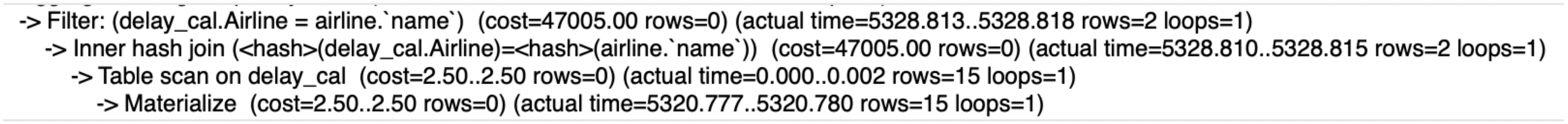
- CREATE INDEX flight_Airline_Arrival_Departure_idx ON flight(Airline(50), ArrivalDelay, DepartureDelay);
  CREATE INDEX flight_Origin_idx ON flight(OriginAirport(50));
  CREATE INDEX flight_Destination_idx ON flight(DestinationAirport(50));
  CREATE INDEX airline_name_idx ON airline(name(50));

- **Total Time:** 5238.876
- **Total Cost:** 47005

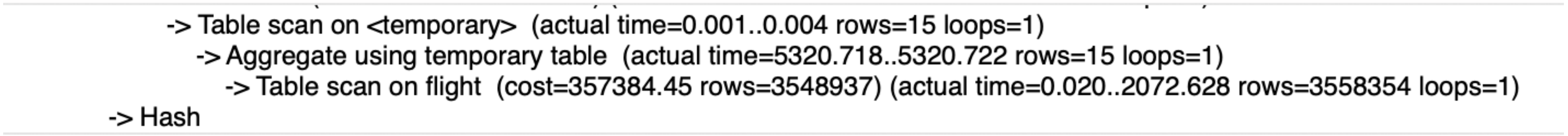


EXPLAIN:

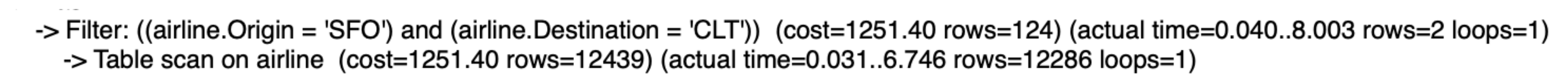
```
-> Limit: 1 row(s)  (actual time=5328.876..5328.876 rows=1 loops=1)
   -> Sort: AVG_Delay, limit input to 1 row(s) per chunk  (actual time=5328.875..5328.875 rows=1 loops=1)
      -> Table scan on <temporary>  (actual time=0.000..0.000 rows=2 loops=1)
         -> Aggregate using temporary table  (actual time=5328.853..5328.854 rows=2 loops=1)
```



```
-> Filter: (delay_cal.Airline = airline.`name`)  (cost=47005.00 rows=0) (actual time=5328.813..5328.818 rows=2 loops=1)
   -> Inner hash join (<hash>(delay_cal.Airline)=<hash>(airline.`name`))  (cost=47005.00 rows=0) (actual time=5328.810..5328.815 rows=2 loops=1)
      -> Table scan on delay_cal  (cost=2.50..2.50 rows=0) (actual time=0.000..0.002 rows=15 loops=1)
         -> Materialize  (cost=2.50..2.50 rows=0) (actual time=5320.777..5320.780 rows=15 loops=1)
```

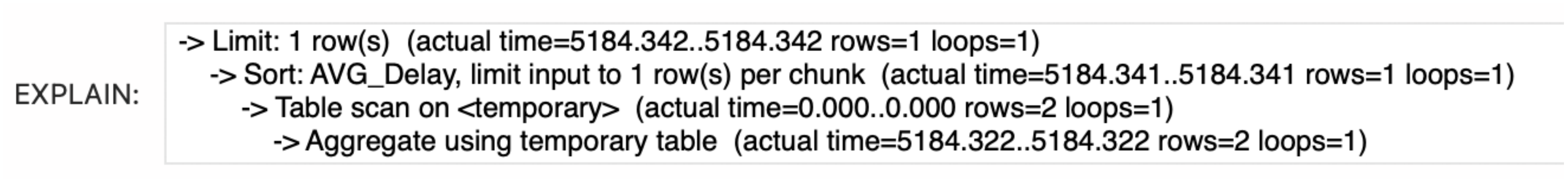


```
            -> Table scan on <temporary>  (actual time=0.001..0.004 rows=15 loops=1)
               -> Aggregate using temporary table  (actual time=5320.718..5320.722 rows=15 loops=1)
                  -> Table scan on flight  (cost=357384.45 rows=3548937) (actual time=0.020..2072.628 rows=3558354 loops=1)
      -> Hash
```



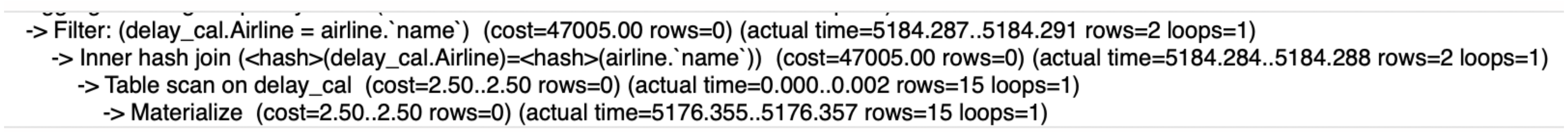
```
-> Filter: ((airline.Origin = 'SFO') and (airline.Destination = 'CLT'))  (cost=1251.40 rows=124) (actual time=0.040..8.003 rows=2 loops=1)
   -> Table scan on airline  (cost=1251.40 rows=12439) (actual time=0.031..6.746 rows=12286 loops=1)
```

- CREATE INDEX flight_ODAAD_idx ON flight(OriginAirport(50), DestinationAirport(50), Airline(50), ArrivalDelay, DepartureDelay);
  CREATE INDEX airline_name_idx ON airline(name(50));
  - **Total Time:** 5184.342
  - **Total Cost:** 357384.45

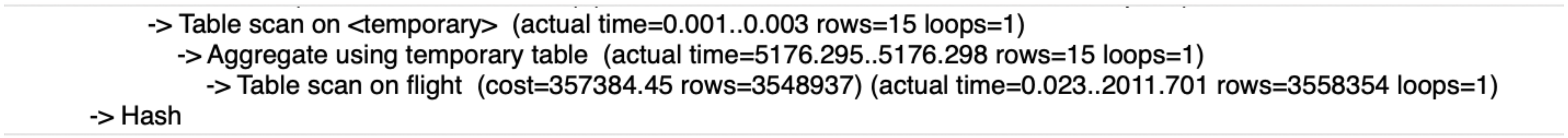


EXPLAIN:

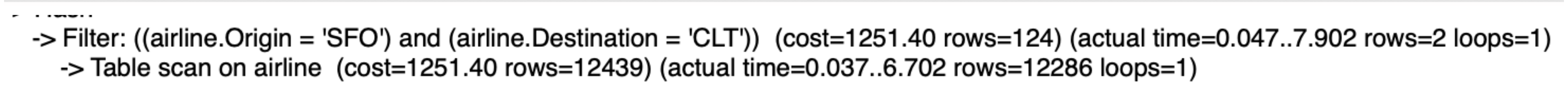
```
-> Limit: 1 row(s)  (actual time=5184.342..5184.342 rows=1 loops=1)
   -> Sort: AVG_Delay, limit input to 1 row(s) per chunk  (actual time=5184.341..5184.341 rows=1 loops=1)
      -> Table scan on <temporary>  (actual time=0.000..0.000 rows=2 loops=1)
         -> Aggregate using temporary table  (actual time=5184.322..5184.322 rows=2 loops=1)
```



```
-> Filter: (delay_cal.Airline = airline.`name`)  (cost=47005.00 rows=0) (actual time=5184.287..5184.291 rows=2 loops=1)
    -> Inner hash join (<hash>(delay_cal.Airline)=<hash>(airline.`name`))  (cost=47005.00 rows=0) (actual time=5184.284..5184.288 rows=2 loops=1)
        -> Table scan on delay_cal  (cost=2.50..2.50 rows=0) (actual time=0.000..0.002 rows=15 loops=1)
            -> Materialize  (cost=2.50..2.50 rows=0) (actual time=5176.355..5176.357 rows=15 loops=1)
```

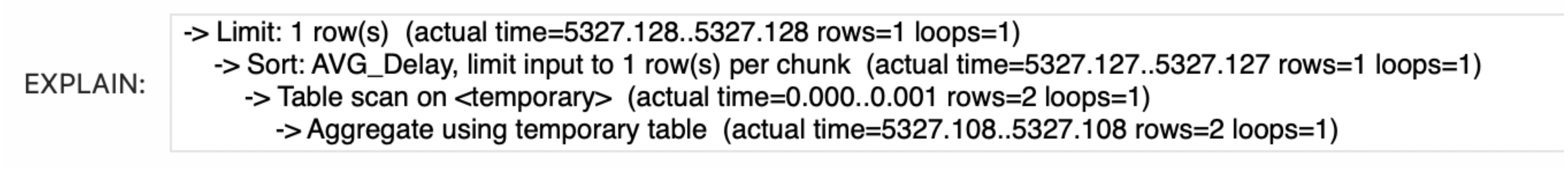


```
                -> Table scan on <temporary>  (actual time=0.001..0.003 rows=15 loops=1)
                    -> Aggregate using temporary table  (actual time=5176.295..5176.298 rows=15 loops=1)
                        -> Table scan on flight  (cost=357384.45 rows=3548937) (actual time=0.023..2011.701 rows=3558354 loops=1)
        -> Hash
```

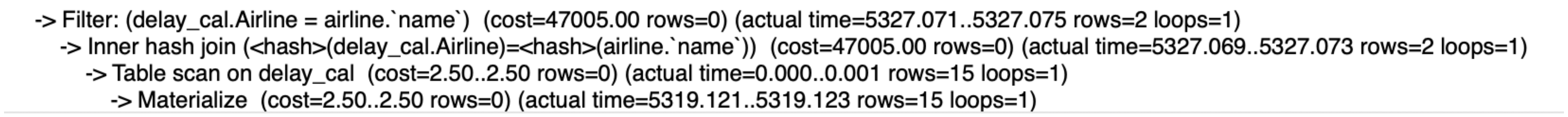


```
 -> Filter: ((airline.Origin = 'SFO') and (airline.Destination = 'CLT'))  (cost=1251.40 rows=124) (actual time=0.047..7.902 rows=2 loops=1)
     -> Table scan on airline  (cost=1251.40 rows=12439) (actual time=0.037..6.702 rows=12286 loops=1)
```

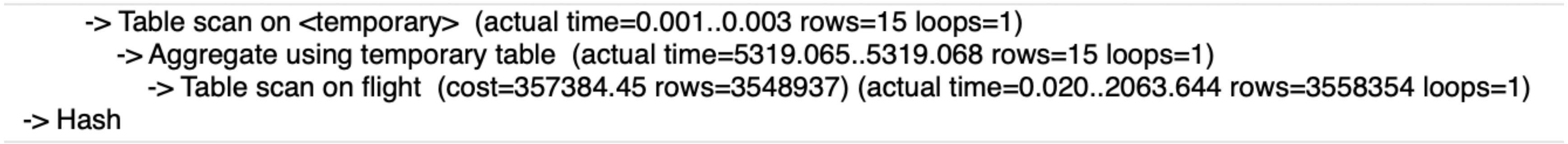
- CREATE INDEX flight_Airline_Arrival_Departure_idx ON flight(Airline(50), ArrivalDelay, DepartureDelay);
  CREATE INDEX airline_name_idx ON airline(name(50));
    - **Total Time:** 5327.128
    - **Total Cost:** 47005.00



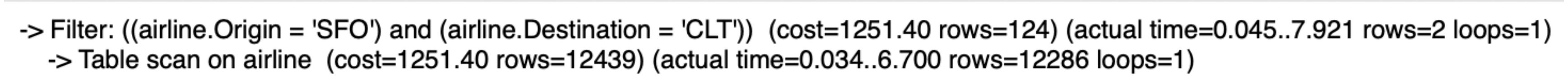
```
EXPLAIN:  -> Limit: 1 row(s)  (actual time=5327.128..5327.128 rows=1 loops=1)
              -> Sort: AVG_Delay, limit input to 1 row(s) per chunk  (actual time=5327.127..5327.127 rows=1 loops=1)
                  -> Table scan on <temporary>  (actual time=0.000..0.001 rows=2 loops=1)
                      -> Aggregate using temporary table  (actual time=5327.108..5327.108 rows=2 loops=1)
```



```
 -> Filter: (delay_cal.Airline = airline.`name`)  (cost=47005.00 rows=0) (actual time=5327.071..5327.075 rows=2 loops=1)
     -> Inner hash join (<hash>(delay_cal.Airline)=<hash>(airline.`name`))  (cost=47005.00 rows=0) (actual time=5327.069..5327.073 rows=2 loops=1)
         -> Table scan on delay_cal  (cost=2.50..2.50 rows=0) (actual time=0.000..0.001 rows=15 loops=1)
             -> Materialize  (cost=2.50..2.50 rows=0) (actual time=5319.121..5319.123 rows=15 loops=1)
```



```
         -> Table scan on <temporary>  (actual time=0.001..0.003 rows=15 loops=1)
             -> Aggregate using temporary table  (actual time=5319.065..5319.068 rows=15 loops=1)
                 -> Table scan on flight  (cost=357384.45 rows=3548937) (actual time=0.020..2063.644 rows=3558354 loops=1)
 -> Hash
```



```
 -> Filter: ((airline.Origin = 'SFO') and (airline.Destination = 'CLT'))  (cost=1251.40 rows=124) (actual time=0.045..7.921 rows=2 loops=1)
     -> Table scan on airline  (cost=1251.40 rows=12439) (actual time=0.034..6.700 rows=12286 loops=1)
```

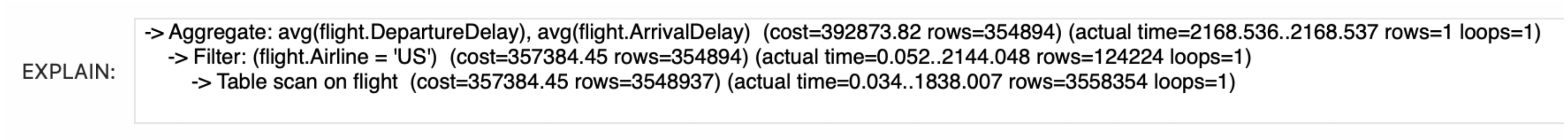
Query 4:
SELECT Airline, AVG(ArrivalDelay) + AVG(DepartureDelay) AS Average_delay

```
FROM flight
WHERE Airline = 'US';
```

Best Index Design and Reason: CREATE INDEX flight_Airline_Arrival_Departure_idx ON flight(Airline(50), ArrivalDelay, DepartureDelay);

- flight_Airline_Arrival_Departure_idx and flight_Airline_idx, flight_Arrival_Departure_idx are both successfulin reducing the time in comparison to the total time before indexing. In addition, they both significantly reduce the total cost in comparison to the total cost before indexing. On the hand, flight_Arrival_Departure_idx was barely able to reduce the total time and total cost from the original. Considering the time can be affected by various factors and the difference between the time of flight_Airline_Arrival_Departure_idx and flight_Airline_idx, flight_Arrival_Departure_idx is less significant than the fact that the total cost of flight_Airline_Arrival_Departure_idx is much lower than the total cost of flight_Airline_idx, flight_Arrival_Departure_idx, this means that flight_Airline_Arrival_Departure_idx is the most optimal indexing design for this query. It's the most optimal since it covers all the necessary columns in one index, so the query doesn't need to access the actual rows of data.

Total Time Before Indexing: 2168.537
Total Cost Before Indexing: 393122.27

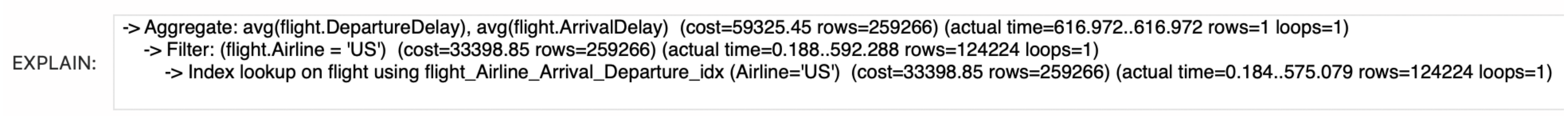


EXPLAIN:

```
-> Aggregate: avg(flight.DepartureDelay), avg(flight.ArrivalDelay)  (cost=392873.82 rows=354894) (actual time=2168.536..2168.537 rows=1 loops=1)
    -> Filter: (flight.Airline = 'US')  (cost=357384.45 rows=354894) (actual time=0.052..2144.048 rows=124224 loops=1)
        -> Table scan on flight  (cost=357384.45 rows=3548937) (actual time=0.034..1838.007 rows=3558354 loops=1)
```

Indexes Added:

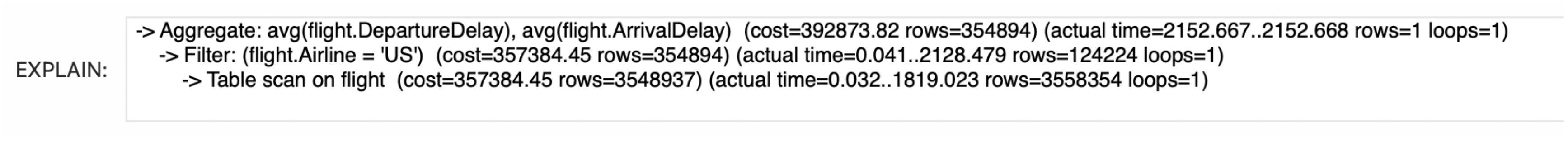
- CREATE INDEX flight_Airline_Arrival_Departure_idx ON flight(Airline(50), ArrivalDelay, DepartureDelay);
    - **Total Time:** 616.972
    - **Total Cost:** 33398.85



EXPLAIN:

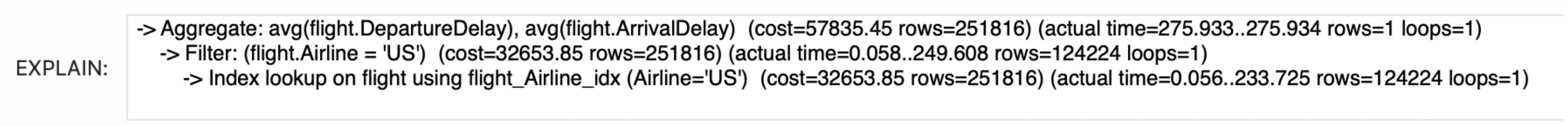
```
-> Aggregate: avg(flight.DepartureDelay), avg(flight.ArrivalDelay)  (cost=59325.45 rows=259266) (actual time=616.972..616.972 rows=1 loops=1)
    -> Filter: (flight.Airline = 'US')  (cost=33398.85 rows=259266) (actual time=0.188..592.288 rows=124224 loops=1)
        -> Index lookup on flight using flight_Airline_Arrival_Departure_idx (Airline='US')  (cost=33398.85 rows=259266) (actual time=0.184..575.079 rows=124224 loops=1)
```

- CREATE INDEX flight_Arrival_Departure_idx ON flight(ArrivalDelay, DepartureDelay);
    - **Total Time:** 2152.668
    - **Total Cost:** 392873.82


EXPLAIN:

```
-> Aggregate: avg(flight.DepartureDelay), avg(flight.ArrivalDelay)  (cost=392873.82 rows=354894) (actual time=2152.667..2152.668 rows=1 loops=1)
    -> Filter: (flight.Airline = 'US')  (cost=357384.45 rows=354894) (actual time=0.041..2128.479 rows=124224 loops=1)
        -> Table scan on flight  (cost=357384.45 rows=3548937) (actual time=0.032..1819.023 rows=3558354 loops=1)
```

- CREATE INDEX flight_Airline_idx ON flight(Airline(50));
- CREATE INDEX flight_Arrival_Departure_idx ON flight(ArrivalDelay, DepartureDelay);
    - **Total Time:** 275.934
    - **Total Cost:** 57835.45


EXPLAIN:

```
-> Aggregate: avg(flight.DepartureDelay), avg(flight.ArrivalDelay)  (cost=57835.45 rows=251816) (actual time=275.933..275.934 rows=1 loops=1)
    -> Filter: (flight.Airline = 'US')  (cost=32653.85 rows=251816) (actual time=0.058..249.608 rows=124224 loops=1)
        -> Index lookup on flight using flight_Airline_idx (Airline='US')  (cost=32653.85 rows=251816) (actual time=0.056..233.725 rows=124224 loops=1)
```