Back **Week 5 Core Quiz**
Graded Quiz • 30 min

Due Jun 11, 11:59 PM +07

Congratulations! You passed!

Grade received 100% **Latest Submission Grade** 100% **To pass** 80% or higher

Go to next item

1. The fly.flights table has the following schema:

1 / 1 point

column	type
year	smallint
month	tinyint
day	tinyint
dep_time	smallint
sched_dep_time	smallint
dep_delay	smallint
arr_time	smallint
sched_arr_time	smallint
arr_delay	smallint
carrier	string
flight	smallint
tailnum	string
origin	string
dest	string
air_time	smallint
distance	smallint

Choose the valid **SELECT** statements. Check all that apply.

☐ SELECT carrier, COUNT(*) FROM fly.flights ORDER BY carrier GROUP BY carrier;

[✓ SELECT carrier, COUNT(*) FROM fly.flights GROUP BY carrier ORDER BY carrier;]

Correct
Correct. The results set is the number of rows in the **fly.flights** table with each possible value of **carrier**, with the rows in alphabetical order.

[✓ SELECT * FROM fly.flights ORDER BY distance;]

Correct
Correct. The result set will be all rows and columns of the **fly.flights** table, with the rows in order so that the

Correct. The result set will be all rows and columns of the **fly.flights** table, with the rows in order so that the lowest **distance** value is first and the highest **distance** value is last.

- ☐ SELECT * FROM fly.flights ORDER BY distance, air_time, delay;

2. Select all the statements that return the same result as **SELECT * FROM flights ORDER BY carrier;**

1 / 1 point

- [✓ SELECT * FROM flights ORDER BY carrier ASC;]
- SELECT * FROM flights ORDER BY carrier ASCENDING;
- SELECT * FROM flights ORDER BY carrier DESC;
- SELECT * FROM flights ORDER BY -carrier ASC;

Correct
Correct. The **ASC** keyword specifies ascending sort order, which is the default sort order.

3. Suppose you want to find the longest-distance flights in the **fly.flights** table for a particular carrier, and then find the flights with the shortest air time.

1 / 1 point

Write a query to return the data in **fly.flights** for American Airlines (**carrier** is **AA**) so that they are sorted by **distance** with the longest distance first, and for those that tie distances, by **air_time** with the shortest air time first. Execute the query in Hue using Impala. What's the shortest air time for the longest distance?

411

Correct
Correct. Your query should have been similar to **SELECT air_time, distance from fly.flights WHERE carrier='AA' order by distance DESC, air_time ASC;**

4. Write and run a SQL query to determine which airport in the **fly.airports** table is closest to the geographical (not magnetic) North Pole, using the following calculation for the distance in kilometers, using the latitude (**lat**) column:
distance = 6371 * 2 * asin(least(1, sin(radians(90 - lat) / 2)))

1 / 1 point

(Note: The **least** function chooses the minimum value among two or more scalar values—similar to the **MIN** function, but **MIN** works on values in a column.)

Which airport is closest to the geographical North Pole?

- Aberdeen Regional Airport
- Cowra Airport
- Wainwright Airport
- [✓ Wiley Post Will Rogers Memorial Airport]
- Zephyrhills Municipal Airport

Correct
[illegible]

Correct. This query will provide the names in order by distance from the North Pole:

SELECT name, 6371 * 2 * asin(least(1,sin(radians(90-lat)/2))) AS distance

FROM fly.airports ORDER BY distance;

5. Select the queries that will return exactly the same result as the query:

1 / 1 point

SELECT * FROM fly.planes ORDER BY year DESC;

when executed by Impala. Check all that apply.

- [ ] SELECT * FROM fly.planes ORDER BY year ASC NULLS FIRST;
- [ ] SELECT * FROM fly.planes ORDER BY year NULLS LAST;
- [x] SELECT * FROM fly.planes ORDER BY year DESC NULLS FIRST;

Correct
Correct. When the order is descending, Impala will list **NULL** values first, so specifying **NULLS FIRST** does not change anything.

- [ ] SELECT * FROM fly.planes ORDER BY year;
- [ ] SELECT * FROM fly.planes ORDER BY year NULLS FIRST;
- [ ] SELECT * FROM fly.planes ORDER BY year ASC;
- [ ] SELECT * FROM fly.planes ORDER BY year DESC NULLS LAST;
- [ ] SELECT * FROM fly.planes ORDER BY year ASC NULLS LAST;

6. Select the queries that will run without error in Hive. Check all that apply.

1 / 1 point

- [x] SELECT * FROM fly.planes ORDER BY type;

Correct
Correct. The query uses **SELECT ***, so any column (including **type**) can be included in the **ORDER BY** clause.

- [ ] SELECT model, 2019 - year AS age_in_2019 FROM fly.planes ORDER BY year;
- [ ] SELECT model FROM fly.planes ORDER BY type;
- [x] SELECT model, 2019 - year AS age_in_2019 FROM fly.planes ORDER BY age_in_2019;

Correct
Correct. The expression **2019 - year** is used in the **SELECT** list and is given the alias **age_in_2019**, then this alias is used in the **ORDER BY** clause. This is valid in Hive.

- [x] SELECT model, type FROM fly.planes ORDER BY type;

Correct
Correct. The **type** column is included in the **SELECT** list, so it can be used in the **ORDER BY** clause.

- [x] SELECT model, year FROM fly.planes ORDER BY 2019 - year;

[✓ SELECT model, year FROM fly.planes ORDER BY 2019 - year;]

Correct
Correct. The ordering expression uses the **year** column, which is included in the **SELECT** list, so this query is valid in Hive.

7. Select the valid SQL queries. Check all that apply.

1 / 1 point

```
SELECT arr_time, AVG(arr_delay) AS avg_arr_delay
  FROM flights WHERE origin = 'LAX'
  LIMIT 100
  GROUP BY arr_time
  HAVING avg_arr_delay > 45;
```

```
SELECT arr_time, AVG(arr_delay) AS avg_arr_delay, 100 AS row_limit
  FROM flights WHERE origin = 'LAX'
  GROUP BY arr_time
  HAVING avg_arr_delay > 45
  LIMIT row_limit;
```

```
SELECT arr_time, AVG(arr_delay) AS avg_arr_delay
  FROM flights WHERE origin = 'LAX'
  GROUP BY arr_time
  HAVING avg_arr_delay > 45
  LIMIT -100;
```

[✓
```
SELECT arr_time, AVG(arr_delay) AS avg_arr_delay
  FROM flights WHERE origin = 'LAX'
  GROUP BY arr_time
  HAVING avg_arr_delay > 45
  LIMIT 1000;
```
]

Correct
Correct. This has a positive literal number in the **LIMIT** clause, which comes last.

```
SELECT arr_time, AVG(arr_delay) AS avg_arr_delay
  FROM flights LIMIT 100
  WHERE origin = 'LAX'
  GROUP BY arr_time
```

```
HAVING avg_arr_delay > 45;
```

8. Which clause should you use with Impala to return rows 1001 through 1050 of a result set?

1 / 1 point

- ○ LIMIT 50 OFFSET 1001
- ● LIMIT 50 OFFSET 1000
- ○ OFFSET 1001,1050
- ○ LIMIT 1050 OFFSET 1000
- ○ LIMIT 1001,1050
- ○ OFFSET 1001 LIMIT 1050

✓ **Correct**
Correct. **LIMIT 50** returns 50 rows, and **OFFSET 1000** says to skip the first 1000 rows and start with row 1001.

9. Select the appropriate uses for the **LIMIT** clause. Check all that apply.

1 / 1 point

- ☐ Randomly sample from a large table
- ☑ Return a few rows from a table to inspect some of the values

✓ **Correct**
Correct. Using **LIMIT** is a good way to inspect a few rows' worth of values from a table.

- ☑ Protect against returning an unexpectedly large number of rows

✓ **Correct**
Correct. When you are unsure how many rows you'll be returning, and there's a possibility of getting more than you want (such as when you're outputting to your terminal screen), using **LIMIT** provides some safety.

- ☐ Filter individual rows based on conditions
- ☑ Reduce the compute resources used by the SQL engine

✓ **Correct**
Correct. Limiting the output can also reduce the amount of resources needed.

10. In what order does a SQL engine execute the clauses of a **SELECT** statement?

1 / 1 point

- ○ FROM, WHERE, GROUP BY, SELECT, HAVING, ORDER BY, LIMIT
- ○ FROM, WHERE, SELECT, GROUP BY, HAVING, ORDER BY, LIMIT
- ○ SELECT, FROM, WHERE, GROUP BY, HAVING, ORDER BY, LIMIT
- ● FROM, WHERE, GROUP BY, HAVING, SELECT, ORDER BY, LIMIT

✓ **Correct**

Correct. The processing order is similar to the specification order, except **SELECT** comes after **HAVING** and before **ORDER BY**.