Week 4 Core Quiz
Graded Quiz • 30 min

**Due** Apr 26, 2:59 AM EDT

**! Try again once you are ready**

**TO PASS** 80% or higher

Try again

GRADE
70%

# Week 4 Core Quiz

**LATEST SUBMISSION GRADE**

70%

1. Below is a table with three rows. What is the value of **AVG(items)** for this table?

**1 / 1 point**

| order_id | items | total |
|---|---|---|
| 829 | 3 | 38.92 |
| 220 | 7 | 107.06 |
| 1043 | 2 | 19.98 |

4

✓ **Correct**

Correct. The sum of the **items** column is 3 + 7 + 2, or 12; dividing that by the number of values in the column (3) gives an average of 4.

2. Which of the following statements are valid? (The column **color** is a string column, and both **red** and **blue** are integer columns.) Check all that apply.

**0 / 1 point**

- [ ] SELECT blue + MIN(red) FROM wax.crayons;
- [x] SELECT MIN(blue + red) FROM wax.crayons;

✓ **Correct**

Correct. In this case, the scalar columns are added and the minimum is found from the resulting values, providing a single aggregated value.

- [x] SELECT -20 + MIN(red) FROM wax.crayons;

✓ **Correct**

Correct. In this case, the single aggregated value will be added to the scalar, still returning a single value.

- [x] SELECT color, MIN(red) FROM wax.crayons;

**! This should not be selected**

Incorrect. The scalar column reference **color** cannot be combined with the aggregate **MIN(red)** in the **SELECT** list.

- [x] SELECT MIN(-20 + red) FROM wax.crayons;

✓ **Correct**

Correct. In this case, a scalar value is added to the scalar column reference and the minimum is found from the resulting values, providing a single aggregated value.

3. The **flights** dataset includes the departure delay (in minutes) and the scheduled time of departure (as an integer, for example 3:14 in the afternoon is 1514). Write and run a query to find the average delay of only those flights that were scheduled to depart after 1:00 in the afternoon. Do not include those scheduled for exactly 1:00. Report to the nearest minute. Note: There are two columns related to departure time—be sure you're using the *scheduled* departure time.

**1 / 1 point**

13

✓ **Correct**

Correct. The query should look like **SELECT round(AVG(dep_delay)) FROM flights WHERE sched_dep_time > 1300;**

4. Here is the **default.orders** table:

**1 / 1 point**

| order_id | cust_id | empl_id | total |
|---|---|---|---|

| | | | |
|---|---|---|---|
| 1 | c | 1 | 24.78 |
| 2 | a | 4 | 28.54 |
| 3 | b | 3 | 48.69 |
| 4 | b | 3 | -16.39 |
| 5 | z | 2 | 29.92 |

How many columns and rows does the result of this query have?

```
SELECT cust_id, COUNT(*), SUM(total)
  FROM default.orders
  GROUP BY cust_id;
```

- ○ 2 columns, 1 row
- ○ 2 columns, 4 rows
- ○ 2 columns, 5 rows
- ○ 3 columns, 1 row
- [✓ 3 columns, 4 rows]
- ○ 3 columns, 5 rows
- ○ 4 columns, 1 row
- ○ 4 columns, 4 rows
- ○ 4 columns, 5 rows
- ○ 6 columns, 1 row
- ○ 6 columns, 4 rows
- ○ 6 columns, 5 rows

✓ **Correct**

Correct. There are four distinct values for **cust_id**, so there will be 4 rows—one for each customer ID group. The three columns will be **cust_id**, **COUNT(*)** (the count of rows in the group), and **SUM(total)** (the sum of the **total** column, for the group).

5. In the **fly.flights** table, the air time of each flight is given in minutes by the **air_time** column. Write and run a query to find the average **air_time** of the flights, in hours, to the nearest tenth of an hour. **1 / 1 point**

1.8

✓ **Correct**

Correct. Your query should look like **SELECT round(AVG(air_time/60),1) FROM fly.flights;** You could also have found the average (in minutes) and then divided by 60 before rounding.

6. Write and run a query on the **fly.planes** table that would answer the question, "How many *different* manufacturer values are there for each type of aircraft?" Then use the results to enter the number of different values for balloon manufacturers are included in the table. **0 / 1 point**

(Note: For this problem, you do not need to control for variations in how the same manufacturer is entered. For example, "ACME Balloons, Inc." and "ACME Balloons" are two different values, even though they probably are for the same manufacturer. )

11255

! **Incorrect**

Incorrect. You might want to review the "Choosing an Aggregate Function and Grouping Column" and "The COUNT Function" videos.

7. For a table of students enrolled at a college, the query **SELECT MIN(age) FROM students;** gave one row in the results, with only one column. The value was **16**. The query **SELECT COUNT(*) FROM students WHERE age IS NULL** returned the value **2827**. Choose which of the following statements is most accurate and informative: **1 / 1 point**

- ○ The lowest age of a student in the **students** table is 16.
- [✓ The lowest known age of a student in the **students** table is 16.]
- ○ The lowest age of a student in the **students** table is unknown.

✓ **Correct**

Correct. This acknowledges that because the table has at least one **NULL** value for **age**, there might be a student younger than 16, but that value is not actually known.

8. Which **SELECT** statements will return the same result as **SELECT COUNT(type) AS num_types FROM fly.planes;** Check all that apply.

0 / 1 point

[✓] SELECT COUNT(*) AS num_types FROM fly.planes WHERE type IS NOT NULL;

✓ **Correct**

Correct. Using the column reference ignores non- **NULL** values, so **COUNT(type)** and **COUNT(*) ... WHERE type IS NOT NULL** will count the same rows.

[ ] SELECT COUNT(*) AS num_types FROM fly.planes WHERE tz IS NULL;

[✓] SELECT COUNT(ALL type) AS num_types FROM fly.planes;

✓ **Correct**

Correct. The **ALL** keyword is the default, so **COUNT(type)** is the same as **COUNT(ALL type)**.

[ ] SELECT COUNT(DISTINCT type) AS num_types FROM fly.planes;

[✓] SELECT COUNT(*) AS num_types FROM fly.planes;

! **This should not be selected**

Incorrect. **COUNT(*)** will include rows in which **type** is **NULL**, but **COUNT(type)** will not. You may want to review the "The COUNT Function" video.

9. Write and run a query in the VM to find all the airports with average departure delays of more than 30 minutes. (Note that you want the origin airports, not the destinations. Also, the **dep_delay** column is given in minutes.) How many airports have more than 30 minutes for their average departure delay?

1 / 1 point

5

✓ **Correct**

Correct. Your query probably looked something like **SELECT origin, AVG(dep_delay) FROM fly.flights GROUP BY origin HAVING avg(dep_delay) > 30**.

10. Choose the **SELECT** statement that returns a result set describing, for each carrier, the average air time for the flights that have a departure delay longer than the flight's air time, and only for carriers with more than 70,000 of those flights.

1 / 1 point

( ) SELECT carrier, AVG(air_time) FROM flights

GROUP BY carrier

WHERE dep_delay > air_time

HAVING COUNT(*) > 70,000;

( ) SELECT carrier, AVG(air_time) FROM flights

WHERE dep_delay > air_time AND COUNT(*) > 70,000

GROUP BY carrier;

( ) SELECT carrier, AVG(air_time) FROM flights

GROUP BY carrier

HAVING dep_delay > air_time AND COUNT(*) > 70,000;

[✓ SELECT carrier, AVG(air_time) FROM flights

WHERE dep_delay > air_time

GROUP BY carrier HAVING count(*) > 70000;]

✓ **Correct**

Correct. For each carrier, the **WHERE** clause filters so only flights with a longer delay than air time are included. Then the **HAVING** clause ensures that they have at least 70,000 rows still being included.