PassQueen
a.k.a dumpsqueen.com

SnowPro Advanced: Data Analyst Certification Exam

Snowflake DAA-C01

Version Demo

Total Demo Questions: 8

Total Premium Questions: 83

Buy Premium PDF

https://passqueen.com

support@passqueen.com

Topic Break Down

Topic	No. of Questions
Topic 1, Data Ingestion and Transformation	19
Topic 2, Data Analysis	50
Topic 3, Data Modeling	3
Topic 4, Data Sharing and Collaboration	5
Topic 5, Data Governance and Security	4
Topic 6, Mix Questions	2
Total	83

QUESTION NO: 1

Which aggregate functions can be used to provide values for a given bucket in Snowsight charts? (Select THREE).

A. Any Value

B. Average

C. Standard Deviation

D. Mode

E. HLL

F. Max

ANSWER: A B F

Explanation:

In Snowsight chart configuration, a bucket groups the underlying result-set rows by a selected dimension, such as a date, category, or region. The value shown for each resulting group is then calculated with an aggregation. **Average** is suitable for presenting the mean numeric value of the records within each bucket, while **Max** returns the greatest value found in each bucket. **Any Value** can provide a representative value from a bucket when the value is effectively constant for that grouping or when any member value is appropriate for the visualization. These functions align with Snowflake's aggregation capabilities: AVG calculates the average of numeric input values, MAX returns the maximum expression value, and ANY_VALUE returns an arbitrary value from a group. Aggregations are applied after grouping, allowing a chart to reduce many rows into one plotted value per bucket. In practice, the selected aggregation should match the intended analytical meaning of the chart, especially when a bucket contains multiple source records. See Snowflake's documentation for ANY_VALUE and the general aggregate functions reference.

QUESTION NO: 2

A Data Analyst has a table of sales territories represented by polygons and a table of store and headquarters locations represented by points. The Analyst must identify the territory containing each store and calculate the distance from each store to its territory headquarters in meters. Which actions should be used? (Select THREE).

A. Store the territory boundaries, stores, and headquarters locations as GEOGRAPHY values.

B. Use ST_CONTAINS(territory_boundary, store_location) to identify the containing territory.

C. Use ST_DISTANCE(store_location, headquarters_location) to calculate the distance in meters.

D. Use ST_LENGTH(store_location, headquarters_location) to calculate the distance in meters.

E. Convert the GEOGRAPHY values to VARCHAR before performing the spatial join.

ANSWER: A B C

Explanation:

`GEOGRAPHY` is appropriate for longitude-and-latitude features represented on the Earth's surface. To determine whether a store point belongs to a territory polygon, the territory should be the first argument to `ST_CONTAINS` and the store should be the second argument. This tests whether the polygon contains the point.

For `GEOGRAPHY` inputs, `ST_DISTANCE` returns the minimum geodesic distance in meters, making it suitable for the headquarters-distance calculation. `ST_LENGTH` measures the length of a spatial object and does not calculate the distance between two locations. Converting spatial values to strings removes their spatial semantics.

QUESTION NO: 3

Why would a Data Analyst use a dimensional model rather than a single flat table to meet BI requirements for a virtual warehouse? (Select TWO).

A. Dimensional modelling will improve query performance over a single table.

B. Dimensional modelling will save on storage space since it is denormalized.

C. Combining facts and dimensions in a single flat table limits the scalability and flexibility.

D. Dimensions and facts allow power users to run ad-hoc analyses.

E. Snowflake generally performs better with dimensional modelling.

ANSWER: C D

Explanation:

Combining facts and dimensions in a single flat table limits the scalability and flexibility because a flat structure tightly couples business measures with every descriptive attribute used to analyze them. As reporting requirements evolve, the same dimension values and attributes are repeatedly embedded alongside transactional facts. Separating facts from conformed dimensions gives the model stable, reusable business entities such as date, customer, product, and location, while allowing descriptive attributes and analytical processes to evolve more independently. This organization also creates a consistent semantic foundation across dashboards and subject areas.

Dimensions and facts allow power users to run ad-hoc analyses because the structure maps naturally to the questions business users ask: aggregate measurable events from a fact table, then filter, group, and compare them using descriptive dimension attributes. A sales measure can therefore be explored by time period, geography, product category, or customer segment without requiring a purpose-built wide table for each reporting combination. Dimensional modeling emphasizes understandable facts, descriptive dimensions, and consistent grain, which makes self-service exploration more predictable for BI users. Snowflake's data-modeling guidance describes choosing models that fit analytical access patterns, while Kimball's dimensional-modeling techniques explain the role of fact and dimension tables in supporting analysis. See Snowflake database modeling guidance and Kimball dimensional modeling techniques.

QUESTION NO: 4

A scorecard tile on a Snowsight dashboard shows a comparison between the industry average employee age (which is 34) and a company ' s average employee age (which is 38). The scorecard tile looks like this:

Comparison with industry average

How should this tile be interpreted?

A. 38 is the comparison and 12% is the average age change in the last year.

B. 38 is the value and 12% is the standard deviation.

C. 38 is the value and 12% is the percent difference from the comparison.

D. 38 is the comparison and 12% is the projected industry growth.

ANSWER: C

Explanation:

38 is the value and 12% is the percent difference from the comparison. A Snowsight scorecard is intended to display one prominent, query-derived metric as its primary value. In this case, the primary metric is the company's calculated average employee age, displayed as 38. The scorecard also has a comparison configured using the industry-average value of 34, allowing the tile to communicate the relationship between the current metric and that benchmark at a glance.

The displayed percentage is the relative difference between the value and the comparison, calculated as `(38 - 34) / 34 × 100`. That result is approximately 11.76%, which the visualization rounds and presents as 12%. Because the company value is greater than the comparison value, the scorecard's positive directional indicator accompanies the percentage. The percentage is therefore contextual information about how far the company average differs from the industry average; it is not a separately calculated distribution statistic or a time-based change measure.

This use of a primary value plus comparison is consistent with Snowflake's documented Snowsight visualization capabilities, including scorecard charts for displaying a single KPI and an optional comparison. See Snowflake's Visualizing data in Snowsight documentation and its Snowsight dashboards documentation.

QUESTION NO: 5

What scheme is used by Snowflake to estimate the approximate similarity between two or more data sets?

A. MINHASH

B. APPROX_PERCENTILE

C. HyperLogLog

D. APPROX_TOP_K

ANSWER: A

Explanation:

MINHASH is the correct scheme because Snowflake uses MinHash signatures to estimate set similarity efficiently, without requiring an exact comparison of every distinct value in each data set. The similarity being estimated is Jaccard similarity: the size of the intersection of sets divided by the size of their union. This makes the approach particularly useful when analyzing large groups of values where an exact set comparison would be computationally expensive.

In Snowflake, the `MINHASH` aggregate function produces a compact MinHash state for a set of input values. Those states can be combined with `MINHASH_COMBINE` when data is partitioned or aggregated at multiple levels, and `APPROXIMATE_SIMILARITY` estimates the similarity from two or more MinHash states. The estimate becomes more accurate as the number of MinHash buckets increases, with an associated trade-off in computation and state size. This workflow is designed specifically for approximate data-set similarity analysis. Snowflake documents the MinHash function family and its use with approximate similarity at MINHASH and APPROXIMATE_SIMILARITY.

QUESTION NO: 6

What will the following query return?

SELECT * FROM testtable SAMPLE BLOCK (0.012) REPEATABLE (99992);

A. A sample of a table in which each block of rows has a 1.2% probability of being included in the sample where repeated elements are allowed.

B. A sample of a table in which each block of rows has a 0.012% probability of being included in the sample, with the seed set to 99992.

C. A sample of a table in which each block of rows has a 1.2% probability of being included in the sample, with the seed set to 99992.

D. A sample containing 99992 records of a table in which each block of rows has a 0.012% probability of being included in the sample.

ANSWER: B

Explanation:

`A sample of a table in which each block of rows has a 0.012% probability of being included in the sample, with the seed set to 99992.` is correct. In Snowflake, the `SAMPLE` (or `TABLESAMPLE`) clause accepts a percentage probability from 0 through 100 when sampling is specified with a numeric value. Therefore, `0.012` means 0.012 percent of the table's data, not 1.2 percent. The `BLOCK` sampling method operates at the micro-partition level: Snowflake probabilistically selects eligible data blocks and returns the rows contained in those selected blocks. Consequently, the result is an approximate sample and its row count can vary substantially, particularly when the table has relatively few or unevenly sized micro-partitions. The `REPEATABLE (99992)` clause supplies the sampling seed. Using the same seed makes block selection reproducible when the underlying table data and its physical layout remain unchanged, which is useful for repeatable analysis and testing. Snowflake documents that block sampling is intended to be more efficient than row-based sampling because it can avoid scanning all rows. See the Snowflake SAMPLE / TABLESAMPLE reference and the micro-partitions documentation.

QUESTION NO: 7

A Data Analyst needs to temporarily hide a tile in a dashboard. The data will need to be available in the future, and additional data may be added. Which tile should be used?

A. Show/Hide

B. Duplicate

C. Delete

D. Unplace

ANSWER: D

Explanation:

Unplace is the appropriate action because it removes a tile from the visible dashboard layout without removing the tile itself from the dashboard. The tile remains available in the dashboard's list of unplaced tiles, so an analyst can place it back on the dashboard later without rebuilding its visualization or query configuration. This supports temporary reporting changes, such as hiding a seasonal metric or a tile that is not currently relevant to the dashboard audience.

Unplacing preserves the tile definition rather than preserving a static result set. When the tile is placed back and its query runs again, it can use the data then available in the underlying Snowflake objects. Therefore, data added while the tile is unplaced can be reflected when the visualization is restored and refreshed, subject to the query logic and any configured refresh behavior. This makes Unplace suitable when the analyst expects both the existing tile setup and future underlying data to remain usable. Snowflake documents dashboard tile layout management, including placing and unplacing tiles, in its Dashboards in Snowsight documentation. Additional dashboard creation and management guidance is available in the Snowsight documentation.

QUESTION NO: 8

A dashboard query against a large event table is restricted to one month and displays only EVENT_DATE, REGION, and total revenue. Which query changes are most likely to reduce the amount of data scanned without changing the required result? (Select TWO).

A. Filter with EVENT_DATE >= '2025-04-01' AND EVENT_DATE < '2025-05-01'.

B. Select only EVENT_DATE, REGION, and the revenue expression required by the dashboard.

C. Filter with TO_VARCHAR(EVENT_DATE, 'YYYY-MM') = '2025-04'.

D. Select all columns from the event table before aggregating the required fields.

E. Add ORDER BY EVENT_DATE to the aggregation query before returning the dashboard result.

ANSWER: A B

Explanation:

Using a direct range predicate on the date column gives Snowflake an opportunity to prune micro-partitions whose metadata falls outside the requested month. Selecting only the required columns also avoids scanning unnecessary column data in Snowflake's columnar storage.

Applying a formatting function to the date column in the predicate can reduce pruning opportunities. Selecting every column reads unnecessary data, and adding an `ORDER BY` changes the execution work without reducing the source data required for the monthly aggregate.