Quiz -- Day 2

Modules: M04 (Delta Optimization), M05 (Streaming & Incremental), M06 (Advanced Transforms)
Format: 20 questions, single correct answer (A-D)
Time: ~15-20 min

> Write your answers in the table at the end, then check against the Answer Key at the bottom.

Q1

What does the `OPTIMIZE` command do on a Delta table?

- A. Deletes old versions from the transaction log
- B. Compacts small files into larger, optimally-sized files
- C. Adds indexes to all columns
- D. Converts the table from Parquet to Delta format

Q2

What is the purpose of `ZORDER BY` when used with `OPTIMIZE`?

- A. It sorts the table alphabetically by column name
- B. It co-locates related data in the same files for faster filter queries
- C. It compresses data using Z-algorithm
- D. It creates a secondary index on the specified columns

Q3

What does `VACUUM` do on a Delta table?

- A. Removes duplicate rows
- B. Removes data files no longer referenced by the transaction log (older than retention threshold)
- C. Compresses the transaction log
- D. Recalculates table statistics

Q4

Which Delta Lake feature replaces traditional partitioning and Z-Ordering with automatic data layout optimization?

- A. Auto Optimize
- B. Predictive Optimization
- C. Liquid Clustering
- D. Adaptive Query Execution

Q5

What is the default retention period for `VACUUM` on a Delta table?

- A. 24 hours
- B. 7 days (168 hours)
- C. 30 days
- D. 90 days

Q6

A data engineer runs the following commands. What is the impact on time travel?

```
VACUUM my_table RETAIN 0 HOURS
```

- A. No impact -- time travel still works for all versions
- B. All historical data files are removed; time travel to old versions will fail
- C. Only the latest version is vacuumed
- D. An error -- VACUUM requires at least 168 hours retention

Q7

What is the role of a checkpoint directory in Structured Streaming?

- A. It stores the output data
- B. It tracks which data has already been processed to enable exactly-once semantics
- C. It caches the stream schema
- D. It logs user queries

Q8

Which Auto Loader option specifies the file format to ingest?

- A. `cloudFiles.path`
- B. `cloudFiles.format`

- C. `cloudFiles.schemaLocation`
- D. `cloudFiles.inferColumnTypes`

Q9

What does `trigger(availableNow=True)` do in a streaming query?

- A. Runs the stream continuously with micro-batches
- B. Processes all available data incrementally and then stops
- C. Waits for new data indefinitely
- D. Runs exactly one micro-batch then pauses

Q10

What is the difference between `COPY INTO` and Auto Loader for incremental file ingestion?

- A. `COPY INTO` uses file notification; Auto Loader uses directory listing
- B. `COPY INTO` stores state in a checkpoint; Auto Loader stores state in the Delta log
- C. `COPY INTO` tracks processed files in the Delta log (idempotent SQL); Auto Loader uses checkpoints and scales to millions of files
- D. There is no difference; they are aliases

Q11

Which of the following correctly shows an Auto Loader read stream?

A.

```
spark.readStream.format("cloudFiles") \
    .option("cloudFiles.format", "json") \
    .option("cloudFiles.schemaLocation", checkpoint) \
    .load("/source/path")
```

B.

```
spark.read.format("autoLoader") \
    .option("format", "json") \
    .load("/source/path")
```

C.

```
spark.readStream.format("autoLoader") \
    .option("fileFormat", "json") \
    .load("/source/path")
```

D.

```
spark.readStream.format("delta") \
    .option("autoLoader", True) \
    .load("/source/path")
```

Q12

What is the purpose of Predictive Optimization in Databricks?

- A. Predicts query execution time
- B. Automatically runs OPTIMIZE, VACUUM, and ANALYZE TABLE based on table usage patterns
- C. Optimizes cluster autoscaling
- D. Predicts storage costs for the next month

Q13

What do Deletion Vectors improve in Delta Lake?

- A. INSERT performance
- B. DELETE, UPDATE, and MERGE performance by marking rows as deleted without rewriting files
- C. SELECT performance for aggregations
- D. VACUUM speed

Q14

Which SQL construct creates a running total (cumulative sum) of `amount` ordered by `date` within each `category`?

A.

```
SUM(amount) GROUP BY category
```

B.

```
SUM(amount) OVER (
    PARTITION BY category
    ORDER BY date
    ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW
)
```

C.

```
RUNNING_SUM(amount, category, date)
```

D.

```
CUMSUM(amount) WITHIN GROUP (ORDER BY date)
```

Q15

What does the `explode()` function do?

- A. Splits a string by delimiter into multiple rows
- B. Converts each element of an array or map into a separate row
- C. Flattens nested JSON into a flat schema
- D. Decompresses compressed columns

Q16

What is the correct syntax for a multi-step CTE (Common Table Expression) in SQL?

A.

```
WITH step1 AS (SELECT ...),
     step2 AS (SELECT ... FROM step1),
     step3 AS (SELECT ... FROM step2)
SELECT * FROM step3
```

B.

```
CTE step1 = (SELECT ...)
CTE step2 = (SELECT ... FROM step1)
SELECT * FROM step2
```

C.

```
WITH step1 AS (SELECT ...)
WITH step2 AS (SELECT ... FROM step1)
SELECT * FROM step2
```

D.

```
DECLARE step1 = SELECT ...
SELECT * FROM step1
```

Q17

Which window function assigns a unique sequential number to each row within a partition, with no gaps?

- A. `RANK()`
- B. `DENSE_RANK()`
- C. `ROW_NUMBER()`
- D. `NTILE()`

Q18

What is the difference between `RANK()` and `DENSE_RANK()` ?

- A. `RANK()` skips numbers after ties; `DENSE_RANK()` does not skip

- B. `RANK()` is for ascending only; `DENSE_RANK()` is for descending
- C. They are identical
- D. `DENSE_RANK()` works only with numeric columns

Q19

Which higher-order function applies a transformation to each element of an array?

A.

```
TRANSFORM(array_col, x -> x * 2)
```

B.

```
MAP(array_col, x -> x * 2)
```

C.

```
APPLY(array_col, x -> x * 2)
```

D.

```
FOREACH(array_col, x -> x * 2)
```

Q20

What does `DESCRIBE DETAIL my_table` return that `DESCRIBE TABLE my_table` does not?

- A. Column names and data types
- B. Table location, file count, size in bytes, partitioning info, and table properties
- C. The SQL definition of the table
- D. Access control permissions

Your Answers

Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10

Q11	Q12	Q13	Q14	Q15	Q16	Q17	Q18	Q19	Q20

Score: ___ / 20

Scroll down for Answer Key

Answer Key -- Day 2

#	Ans	Explanation
Q1	**B**	`OPTIMIZE` compacts small files (small file problem) into larger files, improving read performance.
Q2	**B**	Z-Ordering co-locates related values in the same set of files, enabling data skipping and faster filtered reads.
Q3	**B**	`VACUUM` removes stale data files no longer part of the current table version. Default retention is 7 days.
Q4	**C**	Liquid Clustering (`CLUSTER BY`) automatically optimizes data layout, replacing manual partitioning and ZORDER.
Q5	**B**	The default VACUUM retention is 7 days (168 hours). Files older than this and no longer referenced are removed.
Q6	**B**	`VACUUM RETAIN 0 HOURS` (requires disabling safety check) removes all unreferenced files, breaking time travel for those versions.
Q7	**B**	The checkpoint directory stores offset info and state, ensuring exactly-once fault-tolerant processing.
Q8	**B**	`cloudFiles.format` specifies the source file format (e.g., `json` , `csv` , `parquet`) for Auto Loader.
Q9	**B**	`availableNow=True` processes all currently available data in incremental batches and then stops. Ideal for scheduled jobs.
Q10	**C**	`COPY INTO` is SQL-based idempotent (Delta log). Auto Loader uses checkpoints + file notification, scaling to millions of files.
Q11	**A**	Auto Loader uses `format("cloudFiles")` with `readStream` , and requires `cloudFiles.format` + `cloudFiles.schemaLocation` options.

#	Ans	Explanation
Q12	**B**	Predictive Optimization automatically schedules and runs OPTIMIZE, VACUUM, and ANALYZE based on usage patterns.
Q13	**B**	Deletion Vectors mark rows as deleted in a side file without rewriting the entire Parquet file, speeding up DELETE/UPDATE/MERGE.
Q14	**B**	A window function with `ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW` creates a running total partitioned by category.
Q15	**B**	`explode()` takes an array or map column and generates one row per element (or key-value pair).
Q16	**A**	Multi-step CTEs use a single `WITH` keyword, with each step separated by commas. The final `SELECT` references the last CTE.
Q17	**C**	`ROW_NUMBER()` gives each row a unique sequential number (1, 2, 3...) with no gaps, regardless of ties.
Q18	**A**	`RANK()` leaves gaps after ties (1, 1, 3). `DENSE_RANK()` does not skip (1, 1, 2).
Q19	**A**	`TRANSFORM(array, x -> expr)` applies a lambda to each element. `FILTER` filters elements, `EXISTS` checks a condition.
Q20	**B**	`DESCRIBE DETAIL` returns physical metadata: location, size, numFiles, partitioning, properties, createdAt, etc.