Teradata Certification

Advanced Developer Exam

Exam Objectives

The Advanced Developer Exam covers the features and functionality of the Advanced SQL Engine through release 16.20. The exam objectives describe the content and focus covered on the exam.

Solution Development Process and Considerations -19%

- Given a scenario, identify methods for experimenting with and evaluating the integration of new data with existing production data.
- Given a scenario, identify the type of secondary or join indexes that should be created to improve performance of a class of queries.
- Identify the characteristics of Queue tables.
- Given a scenario, identify the type(s) of compression that should be used on columns of a table.
- Identify the considerations of having Multi-Value Compression (MVC) on a table that also has Block-Level Compression (BLC).
- Identify the characteristics of multi-level partitioning and its impact on performance.
- Given a scenario with a logical model, data profiling and access patterns, identify how to conduct effective physical design. (For example: indexes decisions, secondary structure, partitioning, etc.)
- Identify the characteristics and impacts of system defaults for a geographic region (for example: currency, current date, timestamp, etc.)
- Given a scenario, identify when various options for collecting statistics should be used (for example: expressions, max value, interval, threshold, summary stats, etc.)

Transaction Processing and Lock Management – 10%

- Given a scenario, identify advanced locking types and implications (for example: READ UNCOMMITTED (access lock), COMMITTED (load isolation), and SERIALIZABLE (read lock).)
- Identify situations when load isolation locks should be used.
- Identify the error recovery implications of using different transaction modes including deadlock situations.
- Given a scenario, identify which temporal qualifiers should be used and implications for the chosen qualifier.

Advanced SQL Concepts – 33%

- Identify the types and correct syntax of User Defined Functions (UDF) and User Defined Methods (UDM).
- Identify the usage, characteristics, and operation of external stored procedures.
- Identify the correct syntax, usage, and characteristics of specialized tables including error and QUEUE tables.
- Identify the usage, characteristics and purpose of table characteristics (for example: free space, data block size, merge block ratio, block compression, block compression algorithm, and isolated loading.)
- Identify the usage and characteristics of advanced data types (for example: geospatial, ST_geometry, UDT (arrays), DATASET format (AVRO), JSON, BSON, XML, PERIOD.)
- Identify the usage and characteristics of ALTER TABLE and RENAME.
- Identify the issues with standard referential integrity (for example: revalidate references, resolved / unresolved, consistent / inconsistent, and error table.)
- Identify methods for avoiding skew when populating tables including NoPI tables.
- Identify methods to improve access for a column partitioned table.
- Identify the performance benefits of incorporating advanced indexing strategies (for example: covered indexes, overlay indexes (different types of indexes on the same column(s)), join indexes, etc.)
- Identify the correct syntax, usage, and characteristics using ALTER TABLE to maintain a row partitioned table.
- Identify the use cases of, performance, and implications including maintenance issues of hybrid containers and row partitioning.
- Given a scenario, identify the type of join index that should be used and the implications of using it.
- Identify the correct syntax, usage, and limitations of advanced views (for example: recursive and temporal.)
- Identify the correct syntax, usage and characteristics of triggers in combination with stored procedures (SP), user defined functions (UDF), Referential Integrity (RI), and cascading triggers.
- Given a scenario, identify when identity columns could be used and the implications of using them. Identify the situations when global temporary tables or volatile temporary tables should be used and the limitations of their use.
- Identify the correct syntax, usage, and characteristics of the advanced regular expression functions (for example: REGEXP_SUBSTR, REGEXP_INSTR, REGEXP_SIMILAR, REGEXP_REPLACE, REGEXP_SPLIT_TO_TABLE.)
- Identify the correct ANSI SQL:2011 Window syntax, usage, and characteristics of the OLAP functions.
- Identify the usage and performance characteristics of table operators (for example: SCRIPT, IMPORT, EXPORT, LOAD_FROM_ASTER.)

- Identify the usage and characteristics of the advanced ANSI SQL:2011 Window aggregate functions which include Group window, cumulative window, and moving window.

- Identify the usage and characteristics of user defined data types (UDT) (for example: Array) and JSON functions (for example: compression.)
- Identify the usage and characteristics of advanced Period data type functions (for example: LDIFF, RDIFF, UNTIL_CHANGED/IS NOT UNTIL_CHANGED, IMMEDIATELY PRECEDES, and IMMEDIATELY SUCCEEDS, TD_NORMALIZE, etc.)
- Identify the characteristics and implications of using correlated subqueries.
- Identify the implications of nulls with joins including outer joins.
- Identify the characteristics and implications of a scalar or a nonscalar subquery.
- Identify the characteristics, implications, and correct syntax of UPSERT operations.
- Identify the characteristics, implications, and correct syntax of recursive SQL including within views.

Data Integration Strategies – 12%

- Identify the use cases where the MLOADX (Extended Multiload) protocol would be invoked by TPT.
- Given a scenario, identify the optimal load strategy (for example: Push Down Optimization (PDO), ETL, ELT.)
- Given a TPT job, identify the additional considerations for error handling and job handling that should be used.
- Identify the benefits and performance effects of data integration into Temporal tables.
- Identify the benefits and performance effects of data integration into Columnar tables (for example: NoPI tables, columnar indexing and maintenance, etc.)

Access Layer and Data Delivery Strategies – 12%

- Given a scenario, identify the strategy that should be used to implement Temporal views and qualifiers.
- Identify the benefits, usage, and limitations of using Query Grid as part of a data access strategy.
- Given a data aggregation scenario, identify the access layer development strategies that should be used to achieve performance and timeliness requirements.

Solution Optimization – 14%

- Given a scenario, identify the advanced tuning processes and options that improve performance (for example: Dynamic Explains analyses (IPE and PRPD), Query Rewrite, etc.)
- Identify the benefits and performance implications of using MAPS feature within application design (for example: contiguous vs. sparse maps, when to colocate tables, etc.)
- Identify the design options and characteristics of an optimized application (for example: elements of master data, flexibility, extensibility, resiliency, portability, simplicity, etc.)