**Know when and why to use partitioning and clustering in BigQuery.** Partitioning is the process of dividing tables into segments called *partitions*. BigQuery has three partition types: ingestion time partitioned tables, timestamp partitioned tables, and integer range partitioned tables. In BigQuery, clustering is the ordering of data in its stored format. Clustering is supported only on partitioned tables and is used when filters or aggregations are frequently used.

**Understand the different kinds of queries in BigQuery.** BigQuery supports two types of queries: interactive and batch queries. Interactive queries are executed immediately, whereas batch queries are queued and run when resources are available. The advantage of using these batch queries is that resources are drawn from a shared resource pool and batch queries do not count toward the concurrent rate limit, which is 100 concurrent queries. Queries are run as jobs, similar to jobs run to load and export data.

**Know that BigQuery can access external data without you having to import it into BigQuery first.** BigQuery can access data in external sources, known as federated sources. Instead of first loading data into BigQuery, you can create a reference to an external source. External sources can be Cloud Bigtable, Cloud Storage, and Google Drive. When accessing external data, you can create either permanent or temporary external tables. Permanent tables are those created in a dataset and linked to an external source. Temporary tables are useful for one-time operations, such as loading data into a data warehouse.

**Know that BigQuery ML supports machine learning in BigQuery using SQL.** BigQuery extends standard SQL with the addition of machine learning functionality. This allows BigQuery users to build machine learning models in BigQuery rather than programming models in Python, R, Java, or other programming languages outside of BigQuery.

# Review Questions

You can find the answers in the appendix.

**Q1.1** You are investigating long latencies in Cloud Bigtable query response times. Most queries finish in less than 20 ms, but the 99th percentile queries can take up to 400 ms. You examine a Key Visualizer heatmap and see two areas with bright colors indicating hotspots. What could be causing those hotspots?

**A.** Improperly used secondary index

**B.** Less than optimal partition key

**C.** Improperly designed row-key

**D.** Failure to use a read replica

**Q2. 3** An IoT startup has hired you to review their Cloud Bigtable design. The database stores data generated by over 100,000 sensors that send data every 60 seconds. Each row contains all the data for one sensor sent during an hour. Hours always start at the top of the hour. The row-key is the sensor ID concatenated to the hour of the day followed by the date. What change, if any, would you recommend to this design?

**A.** Use one row per sensor and 60-second datasets instead of storing multiple datasets in a single row.

**B.** Start the row keyrow-key with the day and hour instead of the sensor ID.

**C.** Allow hours to start an any arbitrary time to accommodate differences in sensor clocks.

**D.** No change is recommended.

**P 3. 1 2** Your company has a Cloud Bigtable database that requires strong consistency, but it also requires high availability. You have implemented Cloud Bigtable replication and specified single-cluster routing in the app profile for the database. Some users have noted that they occasionally receive query results inconsistent with what they should have received. The problem seems to correct itself within a minute. What could be the cause of this problem?

**A.** Secondary indexes are being updated during the query and return incorrect results when a secondary index is not fully updated.

**B.** You have not specified an app configuration file that includes single-cluster routing and use of replicas only for failover.

**C.** Tablets are being moved between nodes, which can cause inconsistent query results.

**D.** The row-key is not properly designed.

**R 4. 17** You have been tasked with migrating a MongoDB database to Cloud Spanner. MongoDB is a document database, similar to Cloud Firestore. You would like to maintain some of

the document organization of the MongoDB design. What data type, available in Cloud
Spanner, would you use to define a column that can hold a document-like structure?
**A.** Array
**B.** String
**C.** STRUCT
**D.** JSON
Review Questions **189**
**Q5.1** An application using a Cloud Spanner database has several queries that are taking longer
to execute than the users would like. You review the queries and notice that they all involve
joining three or more tables that are all related hierarchically. What feature of Cloud
Spanner would you try in order to improve the query performance?
**A.** Replicated clusters
**B.** Interleaved tables
**C.** STORING clause
**D.** Execution plans
**Q6.2.3.1** A Cloud Spanner database is using a natural key as the primary key for a large table. The natural
key is the preferred key by users because the values are easy to relate to other data. Database
administrators notice that these keys are causing hotspots on Cloud Spanner nodes and are
adversely affecting performance. What would you recommend in order to improve performance?
**A.** Keep the data of the natural key in the table but use a hash of the natural key as the
primary key
**B.** Keep the natural key and let Cloud Spanner create more splits to improve performance
**C.** Use interleaved tables
**D.** Use more secondary indexes
**Q7.2.1.2** You are using a UUID as the primary key in a Cloud Spanner database. You have noticed
hotspotting that you did not anticipate. What could be the cause?
**A.** You have too many secondary indexes.
**B.** You have too few secondary indexes.
**C.** You are using a type of UUID that has sequentially ordered strings at the beginning of
the UUID.
**D.** You need to make the maximum length of the primary key longer.
**Q8.1** You are working for a financial services firm on a Cloud Bigtable database. The database
stores equity and bond trading information from approximately 950 customers. Over
10,000 equities and bonds are tracked in the database. New data is received at a rate of
5,000 data points per minute. What general design pattern would you recommend?
**A.** Tall and narrow table
**B.** One table for each customer
**C.** One table for equities and one for bonds
**D.** Option A and Option B
**E.** Option A and Option C
**Q9.6.9** You have been brought into a large enterprise to help with a data warehousing initiative. The
first project of the initiative is to build a repository for all customer-related data, including sales,
finance, inventory, and logistics. It has not yet been determined how the data will be used. What
Google Cloud storage system would you recommend that the enterprise use to store that data?
**A.** Cloud Bigtable
**B.** BigQuery
**C.** Cloud Spanner
**D.** Cloud Storage
**190** Chapter 7 ■ Designing Databases for Reliability, Scalability, and Availability
**Q10.1.14** Data is streaming into a BigQuery table. As the data arrives, it is added to a partition
that was automatically created that day. Data that arrives the next day will be written to
a different partition. The data modeler did not specify a column to use as a partition key.
What kind of partition is being used?
**A.** Ingestion time partitioned tables
**B.** Timestamp partitioned tables
**C.** Integer range partitioned tables
**D.** Clustered tables
**Q11.1** You are designing a BigQuery database with multiple tables in a single dataset. The data
stored in the dataset is measurement data from sensors on vehicles in the company's fleet.
Data is collected on each vehicle and downloaded at the end of each shift. After that, it is
loaded into a partitioned table. You want to have efficient access to the most interesting
data, which you define as a particular measurement having a value greater than 100.00.
You want to cluster on that measurement column, which is a FLOAT64. When you define
the table with a timestamped partitioned table and clustering on the measurement column,

you receive an error. What could that error be?
**A.** You cannot use clustering on an external table.
**B.** You cannot use clustering with a FLOAT64 column as the clustering key.
**C.** The table is not the FLOAT64 partition type.
**D.** The clustering key must be an integer or timestamp.
**Q12.3.2** What data formats are supported for external tables in Cloud Storage and Google Drive?
**A.** Comma-separated values only
**B.** Comma-separated values and Avro
**C.** Comma-separated values, Avro, and newline-delimited JSON
**D.** Comma-separated values, Avro, newline-delimited JSON, and Parquet

## Section 4 – Attachments and Declarations

By
⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠
Dan
*Official Google Cloud Certified Professional Data Engineer Study Guide*
Sullivan

# Chapter 8 Understanding Data Operations for Flexibility and Portability

Google Cloud Prof essional Data Engineer Exam objectives co vered in this chapter includ e the fo llowing:

✓ ✓ **4.4 Ensuring flexibility and portability. Considerations include:**

■■ Mapping to current and future business requirements
■■ Designing for data and application portability
(e.g., multi-cloud, data residency requirements)
■■ Data staging, cataloging, and discovery

Data engineers are responsible for many aspects of the data lifecycle in addition to determining and designing storage systems. Data may be operated on in several ways:

- Cataloged
- Preprocessed
- Visualized
- Explored

- Processed with workflows

In this chapter, we will discuss how to use the Data Catalog, a metadata management service supporting the discovery and management of datasets in Google Cloud. Then we will turn our attention to Cloud Dataprep, a preprocessing tool for transforming and enriching data. Next, we will look at Data Studio for visualizing data and Cloud Datalab for interactive exploration and scripting. In each case, we will also discuss business requirements of typical use cases.

## Cataloging and Discovery with Data Catalog

Enterprises accumulate vast amounts of data, and one of the challenges that comes with that is keeping track of information about datasets. For example, there may be hundreds of Cloud Storage buckets and folders that contain thousands of fi les. The people responsible for managing data need to keep track of information such as the contents of the data fi les, the version of the schema if the data is structured, how the data in one fi le relates to data in other fi les, who has access to the data, and so on. This kind of metadata about the datasets is crucial for understanding what data is available, what it means, and how it can be used.

*Data Catalog* is a GCP metadata service for data management. It is fully managed, so there are no servers to provision or confi gure. Its primary function is to provide a single, consolidated view of enterprise data. Metadata is collected automatically during ingest operations to BigQuery and Cloud Pub/Sub as well through APIs and third-party tools. BigQuery metadata is collected on datasets, tables, and views. Cloud Pub/Sub topic metadata is also automatically collected.

Data Catalog is currently in beta. Until it is available for general release, it is unlikely that there will be questions about it on the Professional Data Engineer exam. Nonetheless, data engineers should understand Data Catalog in general because metadata management is essential for compliance, lifecycle data management, and other data engineering tasks.

Before you can use Data Catalog to capture metadata, you need to enable the Data Catalog API in a project that contains the resources created or accessed via the API.

### Searching in Data Catalog

The search capabilities in Data Catalog are based on the same search technology that Google uses with Gmail and Google Drive, so it should be familiar to most users of Google services. With the Data Catalog search capabilities, users can filter and find native metadata, which is captured from the underlying storage system that houses the subject data and usergenerated metadata that is collected from tags. Tagging is discussed in the next section.

To be able to search metadata with Data Catalog, a user will need permissions to read metadata for the subject assets, such as a BigQuery dataset of a Pub/Sub topic. It is important to remember that Data Catalog is collecting and searching metadata, not the data in the dataset, table, topic, and so forth. Figure 8.1 shows an example overview page of Data Catalog.