# Introduction to BigQuery

Optimizing SQL for Analytics in the Cloud

# Objectives

1. Explain what BigQuery is, its core architecture, and where it fits in the modern data ecosystem
2. Identify the key pricing components (storage, analysis) and explain how the serverless model simplifies operation
3. Demonstrate how to load data into a BigQuery table from various sources
4. Execute basic SQL queries to analyze data

# What is BigQuery?

- Google Cloud's fully managed, serverless data warehouse
- Highly scalable
- Optimized for analytical workloads (OLAP)
- Uses SQL (ANSI:2011 compliant)
- Pay-as-you-go pricing (storage + queries)

# BigQuery Architecture Overview

- Columnar storage (fast aggregations and scans)
- Distributed compute engine (parallel processing)
- Separation of storage and compute
- Close integration with other GCP products (Sheets, Looker, BigQuery ML)

# BigQuery vs PostgreSQL/RDBMS

| Feature | BigQuery | PostgreSQL |
| --- | --- | --- |
| Deployment | Cloud-native, serverless | On-prem, cloud VM |
| Scalability | Auto-scales to PBs | Limited by server |
| Workload type | OLAP analytics | OLTP + OLAP |
| Pricing | Pay-per-query | Fixed server costs |
| Maintenance | Handled by GCP | DBA required |
| Indexing | Partitioning and clustering | Strong indexing |

# BigQuery Hierarchy

- BigQuery structures data in a simple three-tier hierarchy (similar to Postgres server/database/schema/table)

Terms:

- Project: somewhat like Postgres server
  - Top-level container for all your resources; all your work happens within a project.
- Dataset: think Postgres database/schema
  - A container for organizing related tables and views. Sets a default location and access control.
- Table: like a Postgres table
  - The core data structure, storing data in a columnar format.

# BigQuery Query Format

- Very similar to Postgres in many ways
- Key difference: instead of simply referencing the table in the FROM clause, you write the full table path enclosed with backticks

```
SELECT DISTINCT year_of_birth
FROM `bigquery-public-data.cms_synthetic_patient_data_omop.person`
```

# Cost Control

- In cloud, you (or your company) pays for consumption

Keys differences from Postgres

- Do not use ‘SELECT * FROM table’ to explore data; use the preview feature instead
- Use partitioning column to limit data scanned instead of LIMIT
- Check the query size before running - costs are calculated based on bytes processed

# Table Organization

- Partitioning: divides a table into smaller, manageable chunks based on a date/timestamp or integer column
  - Always use a WHERE clause on the partition column; this ensures BigQuery only scans the necessary date range, saving time and money
- Clustering: organizes data within partitions based on one or more columns
  - Improves query performance when filtering or aggregating on those clustered columns
- Wildcard tables: allows querying multiple tables with similar names (eg daily log files)
  - Efficiently query a range of historic tables (like a date range of daily tables) using the _TABLE_SUFFIX pseudo-column

# BigQuery Console

- Project Selector: located in the top left; shows what project you're using to query (linked to billing)
- Explorer panel: lists datasets and tables
  - Click a table to view schema, details (including table storage size and partitioning), and preview data
- Query editor: where you write your SQL
  - Query validator: checks syntax as you type
  - Bytes scanned estimator: Shows approximate data to be scanned before you run the query
- Query results/job information: shows query results and important job metadata
  - Metadata includes total bytes scanned, time elapsed, slot time consumed (equivalent to compute resource usage)

# Additional Resources

- BigQuery Documentation:
  - https://cloud.google.com/bigquery/docs
- Public Datasets:
  - https://cloud.google.com/bigquery/public-data
- BigQuery SQL Reference:
  - https://cloud.google.com/bigquery/docs/reference/standard-sql/query-syntax