

Week 3

DB Scaling and BigQuery

IS457 - Advanced Database Management



Outline

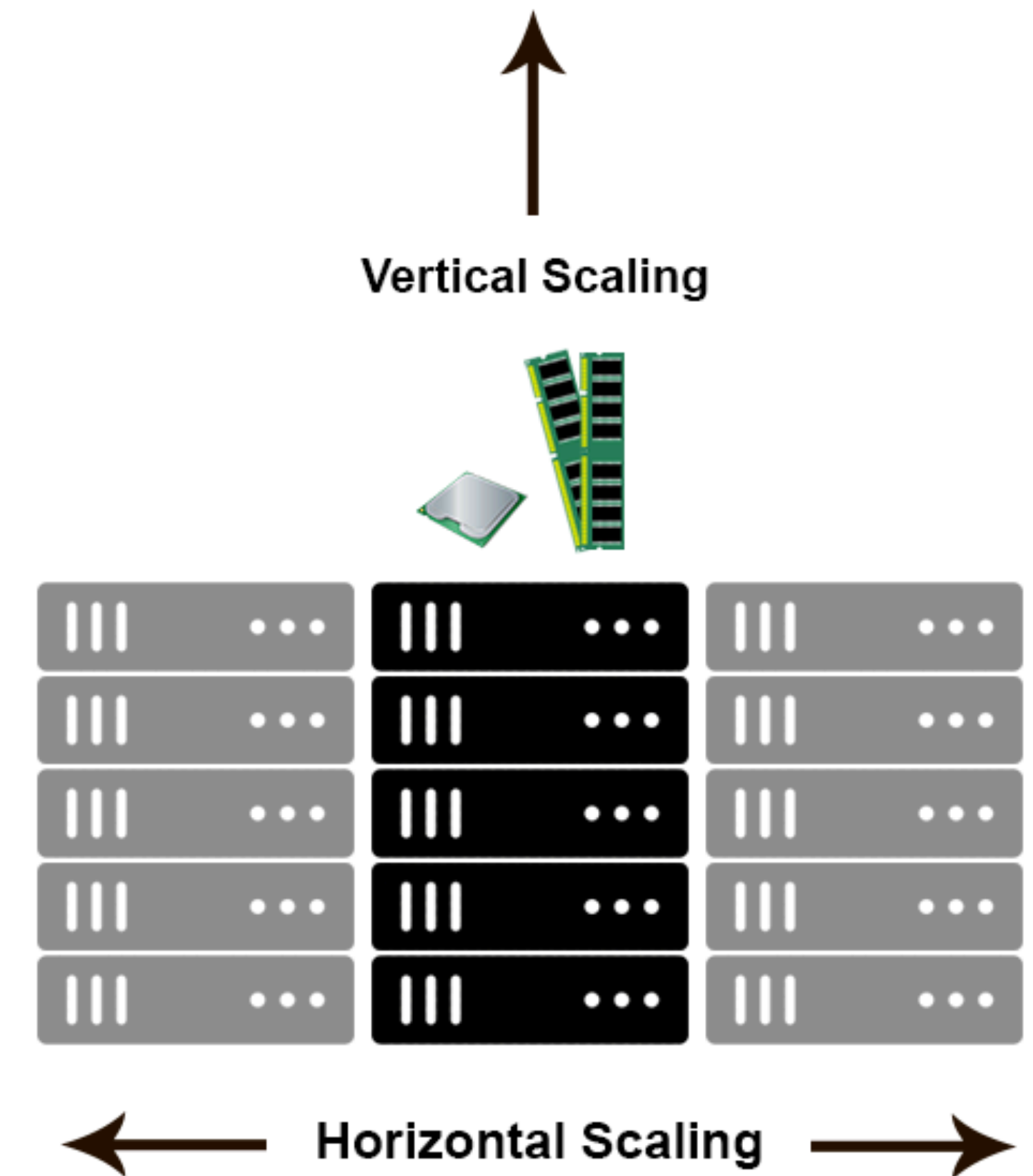
- ▶ Online Materials: <https://mentor2code.com/courses/is457/>
- ▶ Catch up Exercise - SQL vs NoSQL
 - ▶ <https://mentor2code.com/courses/is457/nosql-exercise.html>
- ▶ DB Scaling
- ▶ BigQuery

Scalability

- ▶ **Scalability** is the property of a system to **handle a growing amount of work** by **adding resources to the system**.
- ▶ In an economic context, a **scalable business model** implies that a company can increase sales given **increased resources**.
- ▶ For example, a package delivery system is scalable because more packages can be delivered by **adding more delivery vehicles**.
- ▶ However, if all packages had to first pass through a single warehouse for sorting, the system would not be scalable, because one warehouse can handle only a limited number of packages

Horizontal and Vertical Scaling

- ▶ **Scaling vertically (up/down)** means adding resources to (or removing resources from) a single node, typically involving the addition of CPUs, memory or storage to a single computer.
- ▶ **Scaling horizontally (out/in)** means adding more nodes to (or removing nodes from) a system, such as adding a new computer to a distributed software application.



Explore more at <https://stackoverflow.com/questions/11707879/difference-between-scaling-horizontally-and-vertically-for-databases>

Common Approach to Scaling MySQL

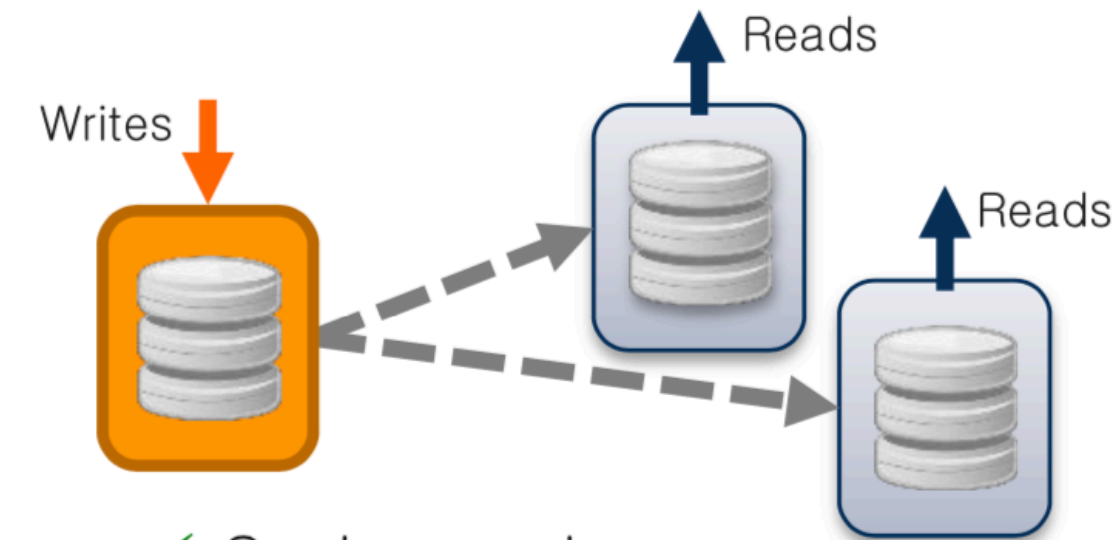
Bigger Server



- ✓ Easy to do
- ✓ No app changes

- ✗ Very limited gains
- ✗ No HA

Reader Slaves

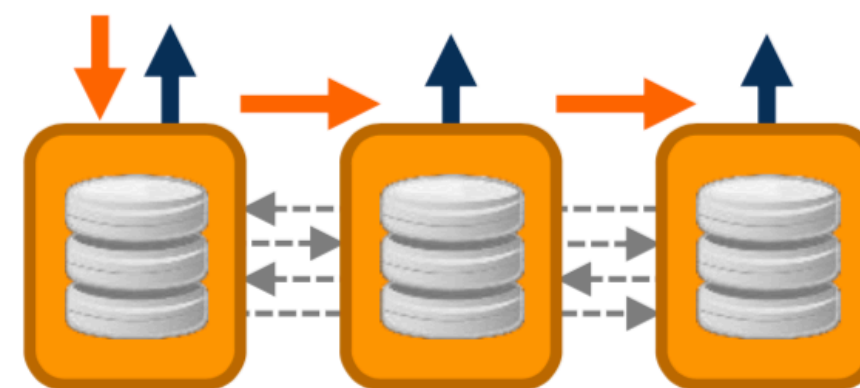


- ✓ Scales reads
- ✓ Fairly easy to setup

- ✗ No write scaling
- ✗ Adds slave lag
app needs to be aware
- ✗ App changes
needs to split reads & writes
- ✗ No HA
adds DR but needs failover
- ✗ Fragile
High burden to maintain

Synchronous Replication Clusters

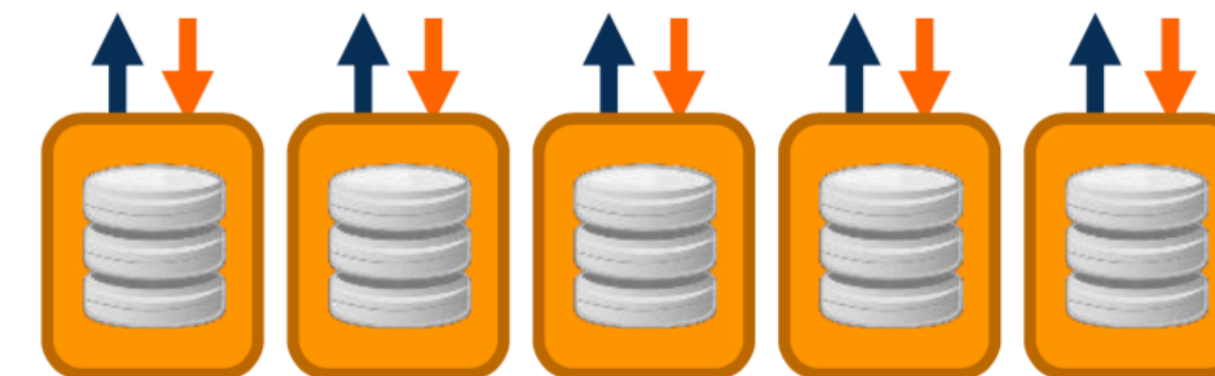
Galera | Percona XtraDB Cluster | MariaDB Galera Cluster



- ✓ Adds HA
- ✓ Scales reads

- ✗ No write scaling
Writes are multiplied
- ✗ Slows down writes
sync rep adds write latency
- ✗ No storage scaling
data fully replicated on each node

Database Sharding



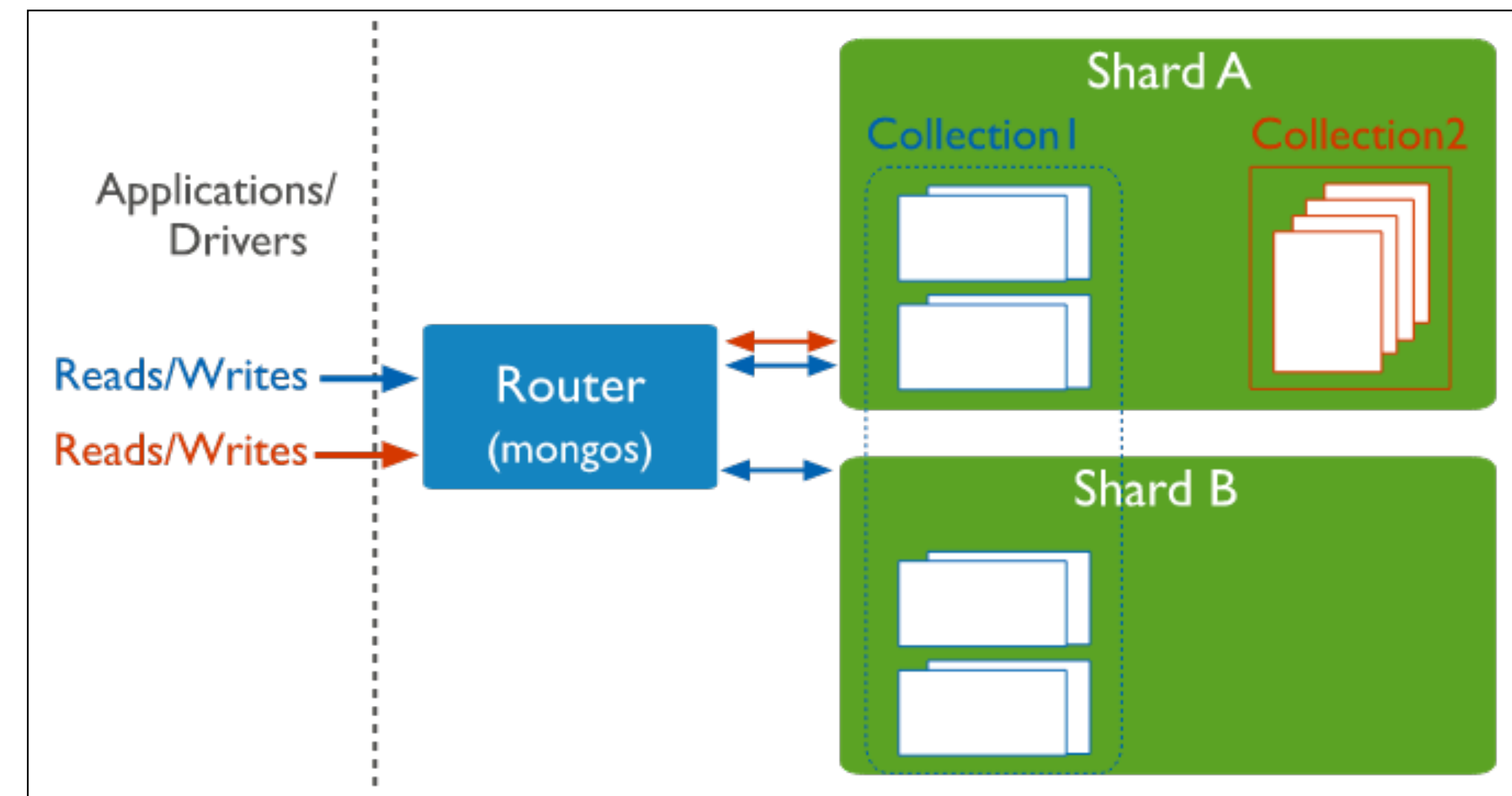
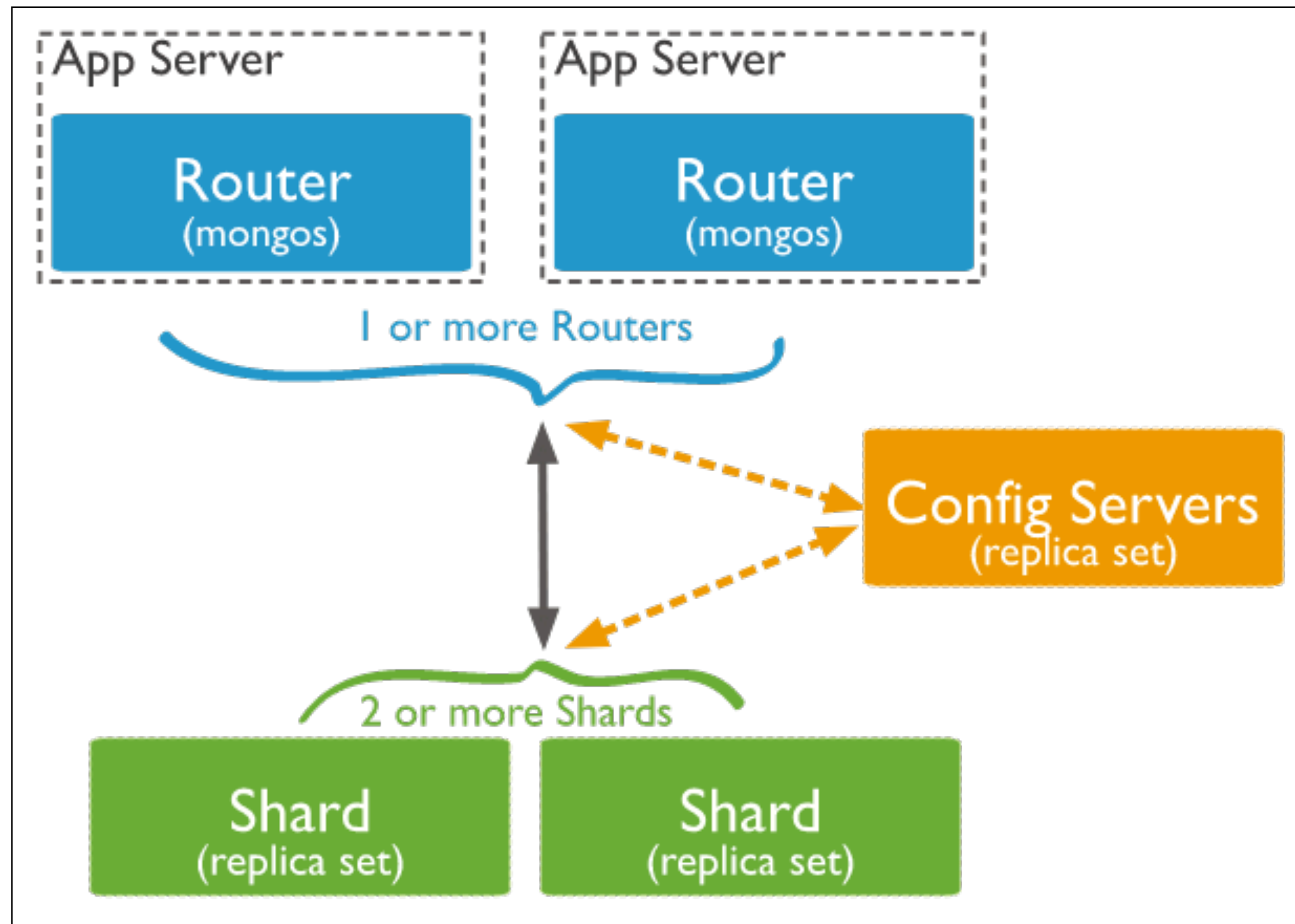
- ✓ Scales reads
- ✓ Scales writes
- ✓ Scales storage

- ✗ Significant app changes
or entire redesign
- ✗ Does not add HA
multiplies the failure points
- ✗ DB management overhead
i.e. consistent backup of data set

Scaling MongoDB

- ▶ MongoDB supports horizontal scaling through sharding.
- ▶ A MongoDB sharded cluster consists of the following components:
 - ▶ **shard:** Each shard contains a subset of the sharded data. Each shard can be deployed as a replica set.
 - ▶ **mongos:** The mongos acts as a query router, providing an interface between client applications and the sharded cluster.
 - ▶ **config servers:** Config servers store metadata and configuration settings for the cluster.

Scaling MongoDB

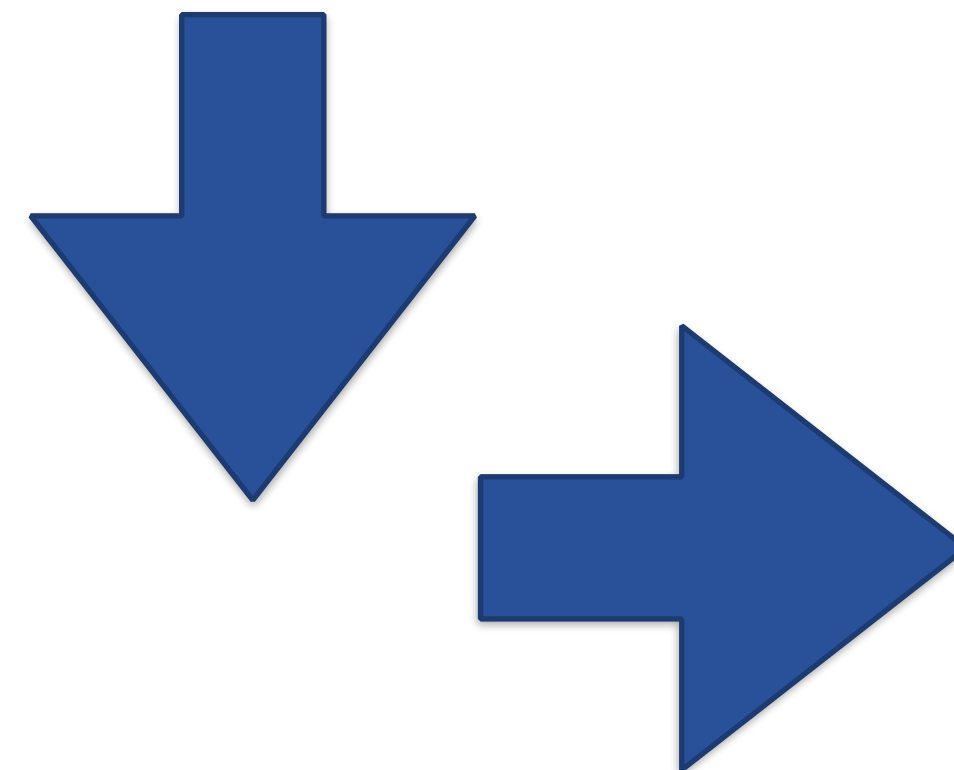
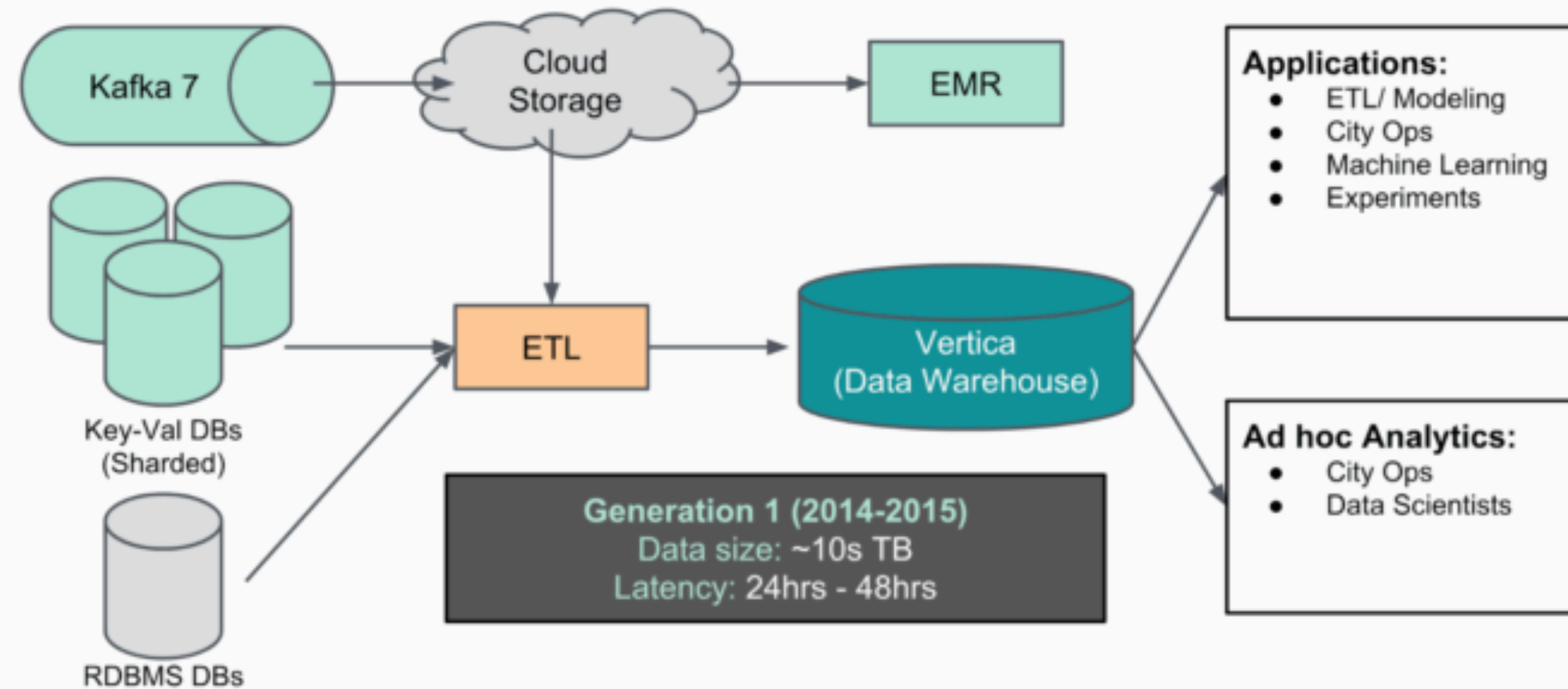


Scaling Lesson Learned

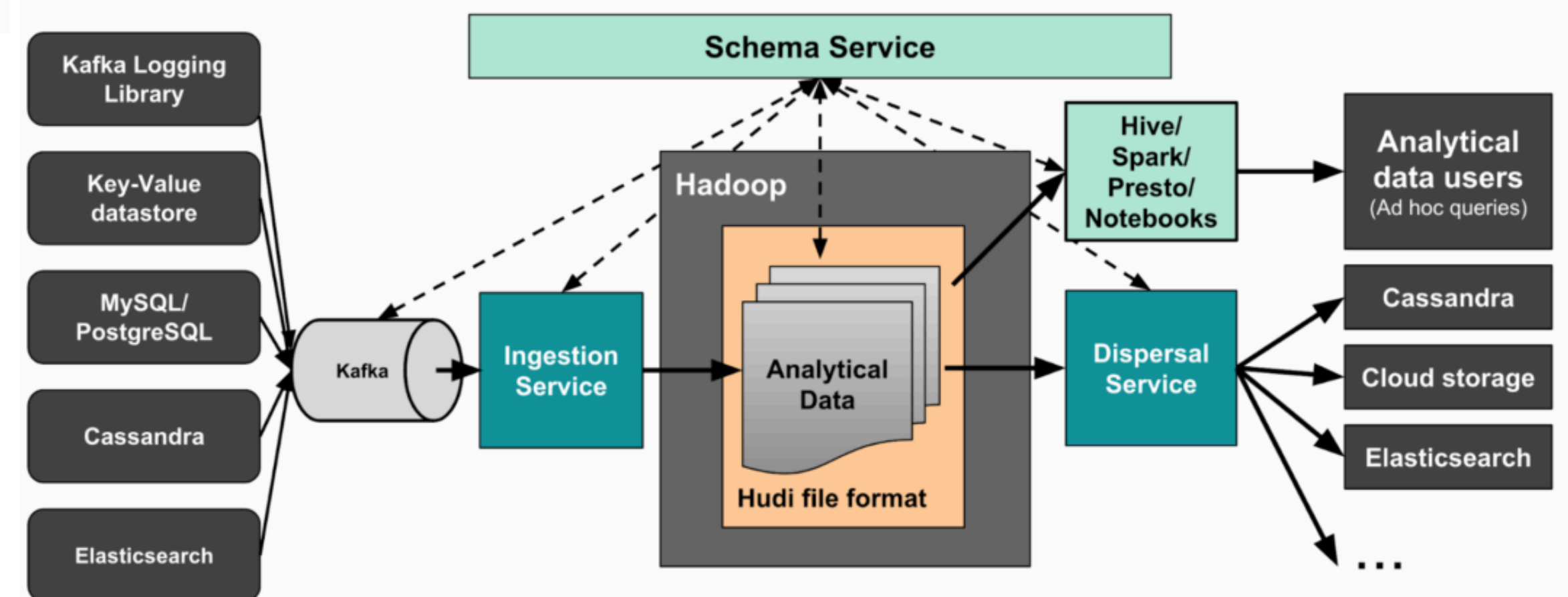
- ▶ ทำไม Pantip ปิดปรับปรุงบ่อย บทเรียน ความเจ็บปวด และคราบน้ำตาจาก MongoDB
 - ▶ <http://macroart.net/2013/10/mongodb-lessons-learned-on-pantip/>

Uber's Big Data Platform

Generation 1 (2014-2015) - The beginning of Big Data at Uber



Generic Any-to-Any Data platform



Introduction to BigQuery

- ▶ Storing and querying massive datasets **can be time consuming and expensive** without the right hardware and infrastructure.
- ▶ **BigQuery** is an enterprise data warehouse that solves this problem by enabling super-fast SQL queries **using the processing power of Google's infrastructure.**
- ▶ Simply move your data into **BigQuery** and let us handle the hard work.

Introduction to BigQuery

- ▶ You can control access to both the project and your data based on your business needs, such as **giving others the ability to view or query your data.**
- ▶ BigQuery is **fully-managed.** To get started, you don't need to deploy any resources, such as disks and virtual machines. Get started now by running a web query or using the command-line tool.

Introduction to BigQuery

- ▶ You can access BigQuery by:
 - ▶ Using the GCP Console or the classic web UI.
 - ▶ Using a command-line tool.
 - ▶ Making calls to the BigQuery REST API using a variety of client libraries such as Java, .NET, or Python.
- ▶ There are also a variety of **third-party tools** that you can use to interact with BigQuery, such as visualising the data or loading the data.

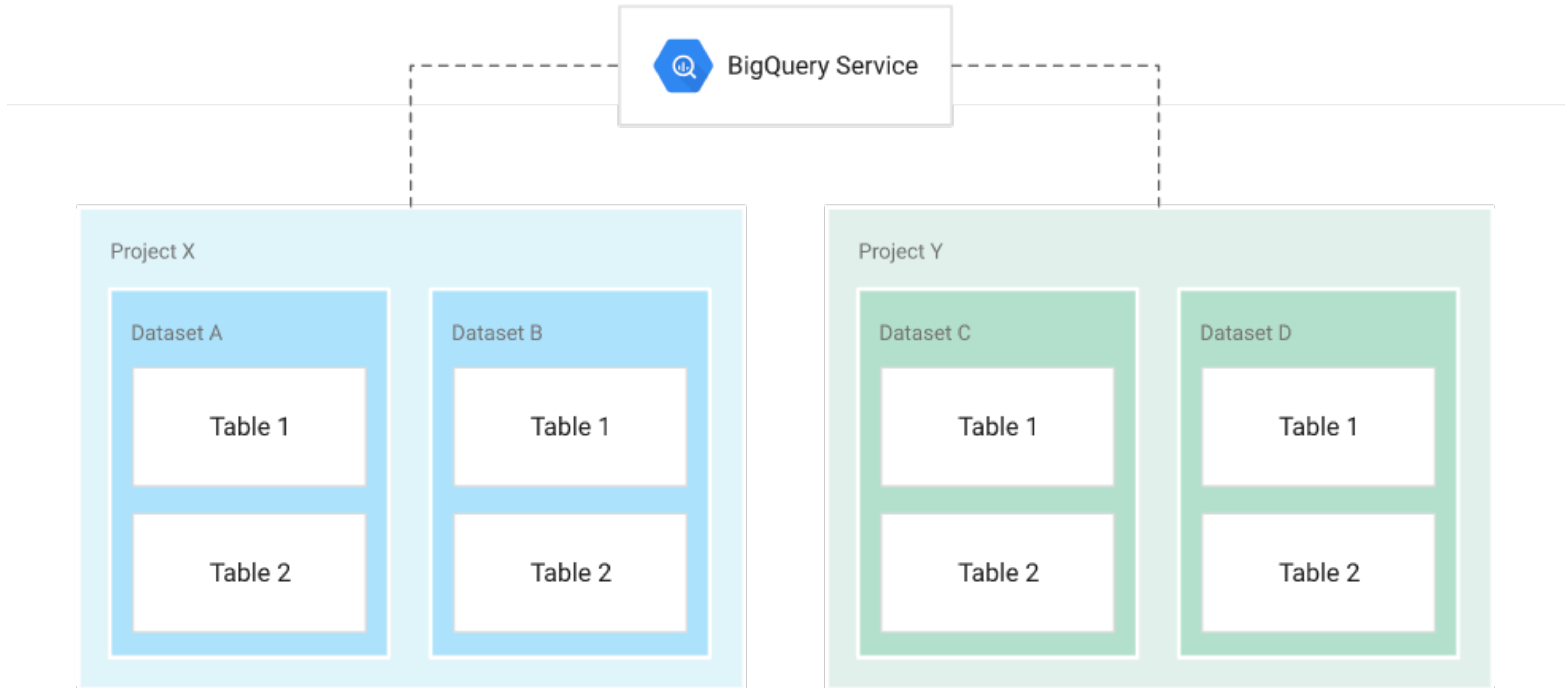
Service Model Comparison

<i>Data Warehouse</i>	<i>BigQuery</i>
Data warehouse	<i>The BigQuery service replaces the typical hardware setup for a traditional data warehouse. That is, it serves as a collective home for all analytical data in an organisation.</i>
Data mart	<i>Datasets are collections of tables that can be divided along business lines or a given analytical domain. Each dataset is tied to a Google Cloud Platform (GCP) project.</i>
Data lake	<i>Your data lake might contain files in Cloud Storage or Google Drive or transactional data in Cloud Bigtable. BigQuery can define a schema and issue queries directly on external data as federated data sources.</i>
Tables and views	<i>Tables and views function the same way in BigQuery as they do in a traditional data warehouse.</i>
Grants	<i>Cloud Identity and Access Management (Cloud IAM) is used to grant permission to perform specific actions in BigQuery.</i>

Datasets

- ▶ BigQuery organises data tables into units called datasets.
- ▶ These datasets are scoped to your GCP project. When you reference a table from the command line, in SQL queries, or in code, you refer to it by using the following construct:
 - ▶ ***project.dataset.table***
- ▶ These multiple scopes—project, dataset, and table—can help you structure your information logically.
- ▶ You can use multiple datasets to separate tables pertaining to different analytical domains.
- ▶ You can use project-level scoping to isolate datasets from each other according to your business needs.

Structural Overview of BigQuery



Provisioning and System Sizing

- ▶ You don't need to provision resources before using BigQuery, unlike many RDBMS systems. BigQuery allocates storage and query resources dynamically based on your usage patterns.
 - ▶ **Storage** resources are allocated as you consume them and deallocated as you remove data or drop tables.
 - ▶ **Query** resources are allocated according to query type and complexity. Each query uses some number of slots, which are units of computation that comprise a certain amount of CPU and RAM.

Storage Management

- ▶ Internally, BigQuery stores data in a proprietary columnar format called **Capacitor**, which has a number of benefits for data warehouse workloads.
 - ▶ <https://cloud.google.com/blog/products/gcp/inside-capacitor-bigquerys-next-generation-columnar-storage-format>
- ▶ BigQuery uses a proprietary format because it can evolve in tandem with the query engine, which takes advantage of deep knowledge of the data layout to optimise query execution.
- ▶ BigQuery uses query access patterns to determine the optimal number of physical shards and how they are encoded.

Storage Management

- ▶ The data is physically stored on Google's distributed file system, called **Colossus**, which ensures durability by using **erasure encoding** to store redundant chunks of the data on multiple physical disks.
 - ▶ https://cloud.google.com/files/storage_architecture_and_challenges.pdf
 - ▶ https://wikipedia.org/wiki/Erasure_code
- ▶ Moreover, the data is replicated to multiple data centres.
- ▶ You can also run BigQuery queries on data outside of BigQuery storage, such as data stored in Cloud Storage, Google Drive, or Cloud Bigtable, by using **federated data sources**.
 - ▶ <https://cloud.google.com/bigquery/federated-data-sources>
- ▶ However, these sources are not optimised for BigQuery operations, so they might not perform as well as data stored in BigQuery storage.

Maintenance

- ▶ BigQuery is a **fully-managed service**, which means that the BigQuery **engineering team** takes care of updates and maintenance for you.
- ▶ Upgrades shouldn't require **downtime** or **hinder system performance**.
- ▶ Many traditional systems require resource-intensive vacuum processes to run at various intervals to reshuffle and sort data blocks and recover space.
- ▶ BigQuery has **no equivalent of the vacuum process**, because the storage engine continuously manages and optimises how data is stored and replicated.
- ▶ Also, because BigQuery **doesn't use indexes on tables**, you don't need to rebuild indexes.

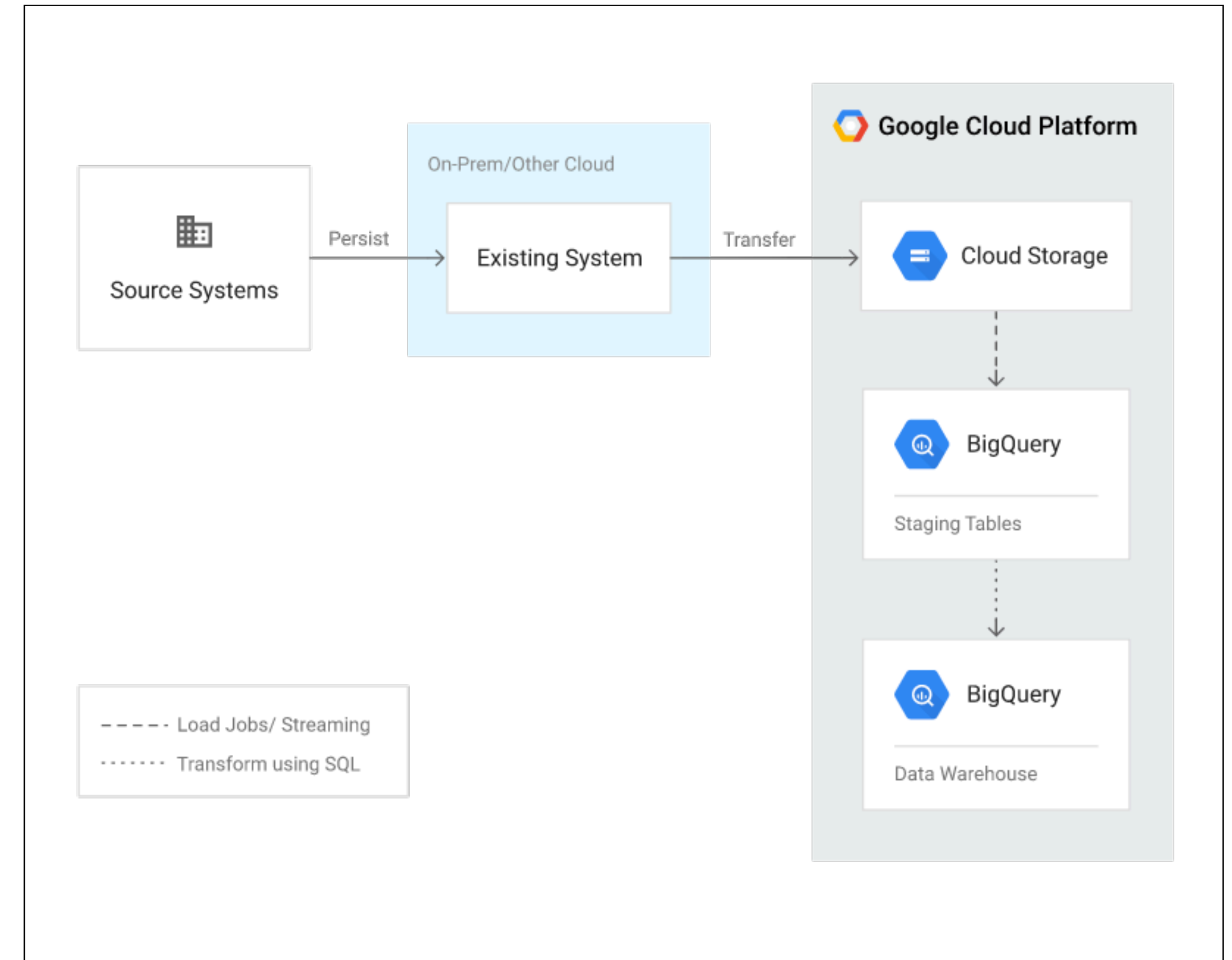
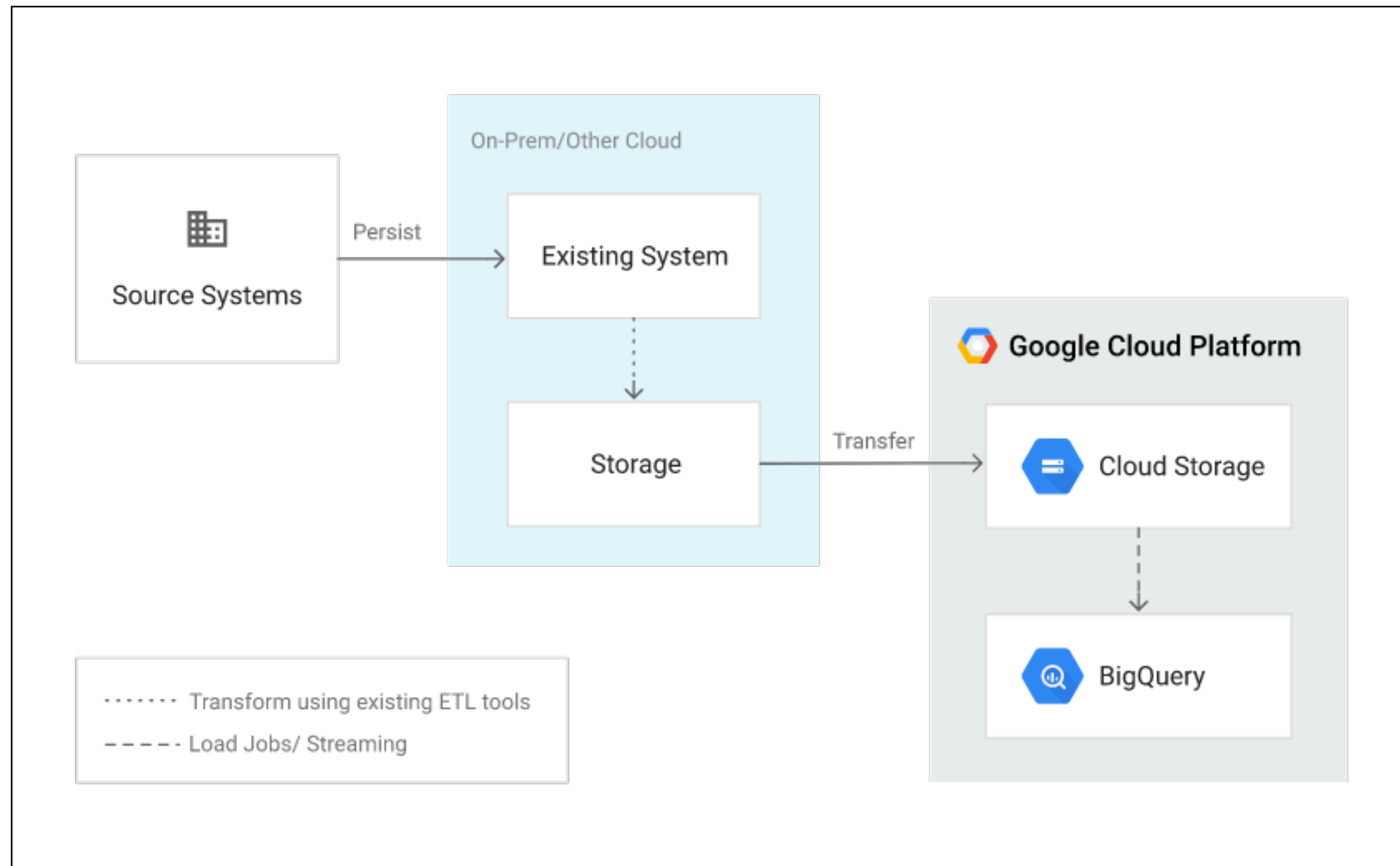
Backup and Recovery

- ▶ BigQuery addresses backup and disaster recovery **at the service level**.
- ▶ Also, by maintaining **a complete 7-day history** of changes against your tables, BigQuery allows you to query **a point-in-time snapshot** of your data.
- ▶ You can easily revert changes **without** having to request a recovery from backups. (When a table is explicitly deleted, its history is flushed after 2 days.)

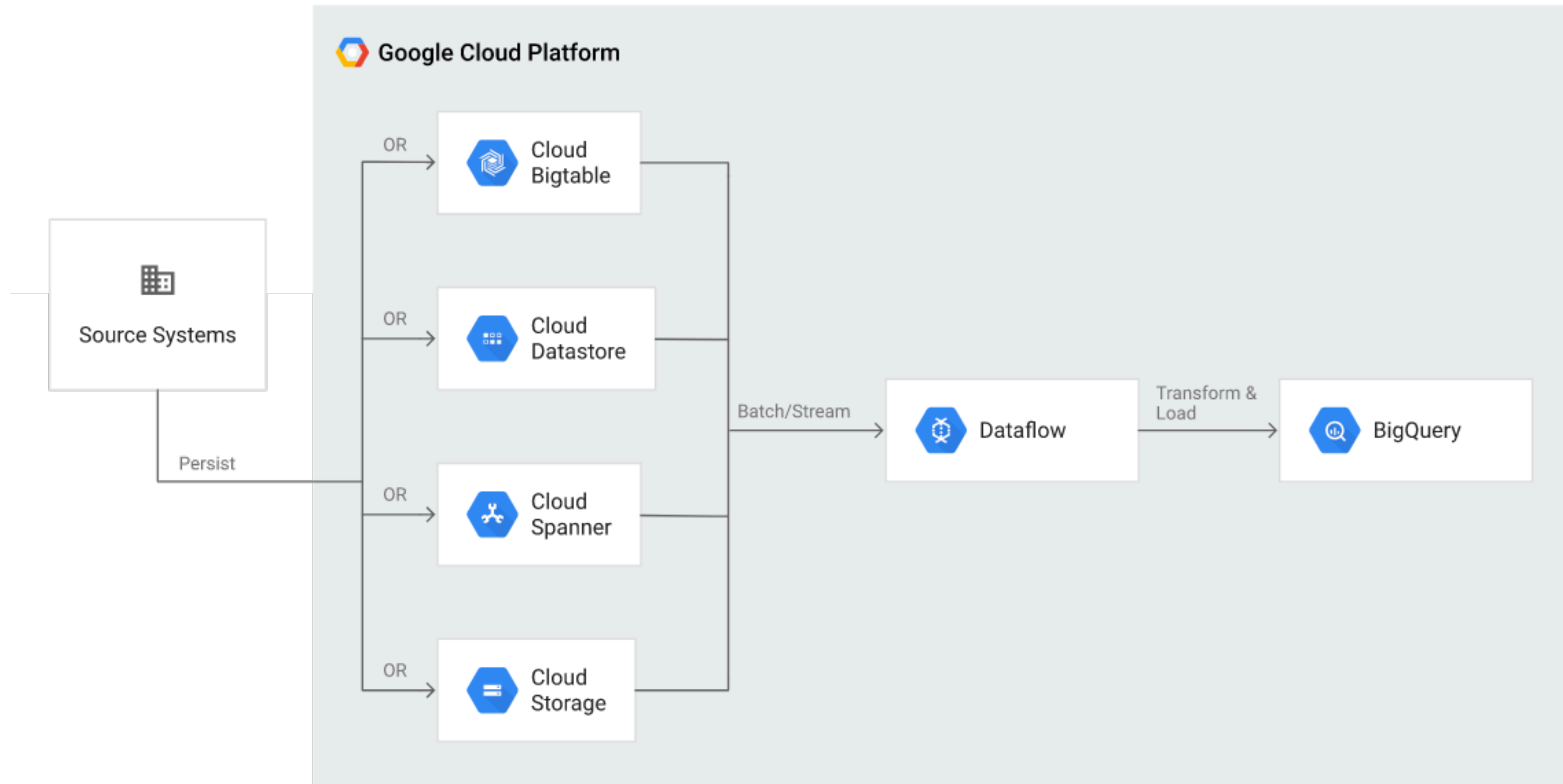
Loading Data

- ▶ Before data can be loaded into BigQuery for analytical workloads, it is typically stored in a Cloud Storage product and in **a format that is native to its origin.**
- ▶ During early stages of migration to GCP, the common pattern is to use existing **extract, transform, and load (ETL) tools** to transform data into the ideal schema for BigQuery.
- ▶ After data is transformed, it is transferred to Cloud Storage **as CSV, JSON, or Avro files,** and loaded into BigQuery by using load jobs or streaming.
- ▶ Alternatively, you can transfer files to Cloud Storage in the schema that is native to the existing **on-premises data storage,** loaded into a set of staging tables in BigQuery and then transformed into the ideal schema for BigQuery.

Loading Data



Loading Data

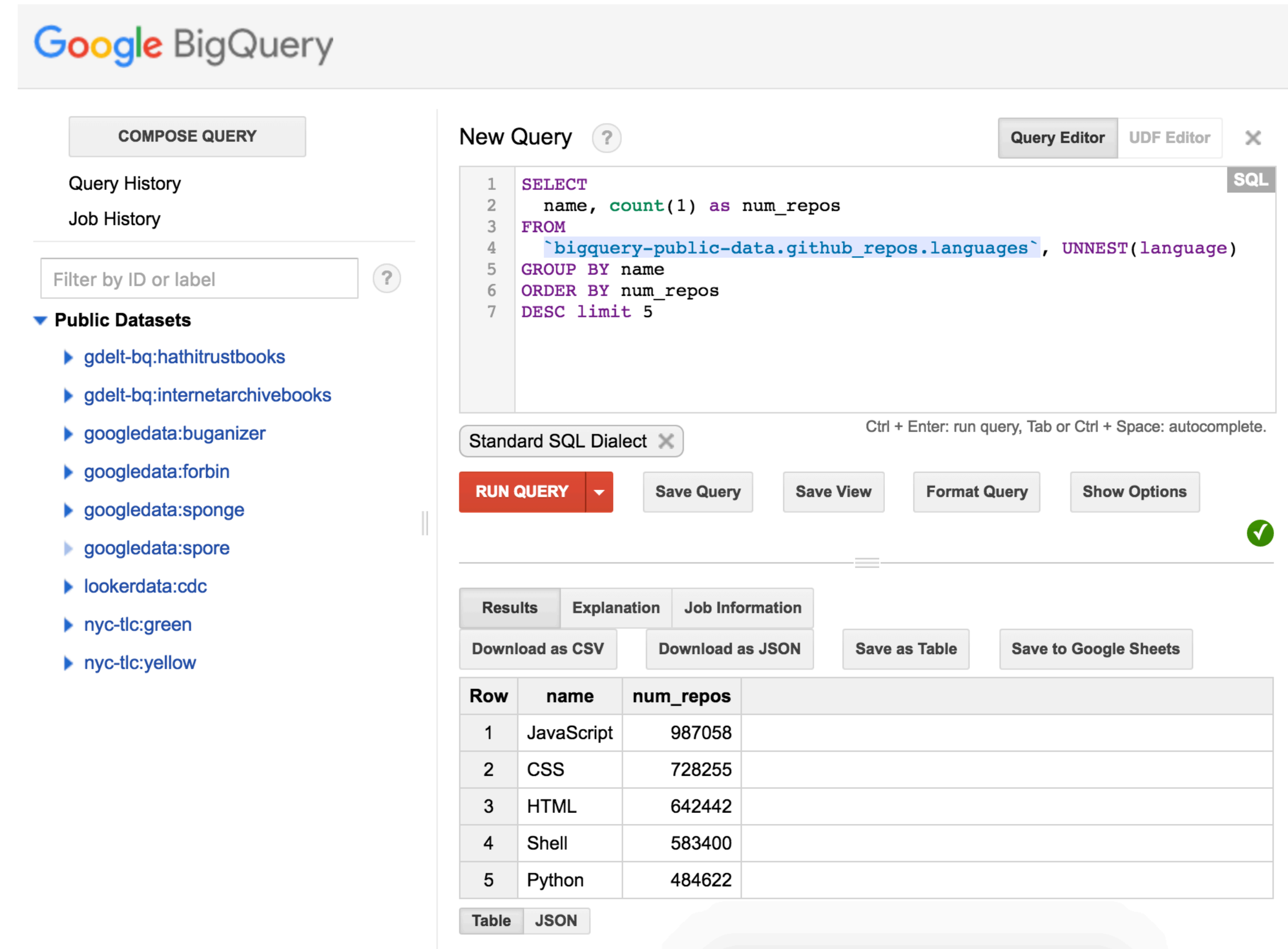


Querying Data

- ▶ BigQuery supports standard SQL queries and is compatible with ANSI SQL 2011.
- ▶ BigQuery's SQL reference provides a comprehensive description of all functions, operators, and regex capabilities that are supported.
- ▶ Because BigQuery supports nested and repeated fields as part of the data model, its SQL support has been extended to specifically support these field types.
- ▶ For example, using the **GitHub public dataset**, you could issue the **UNNEST** command, which lets you iterate over a repeated field:
 - ▶ <https://cloud.google.com/bigquery/public-data/github>
 - ▶ <https://cloud.google.com/bigquery/sql-reference/query-syntax#unnest>

Interactive Query

- ▶ The BigQuery web UI allows **interactive querying of datasets** and provides a consolidated view of datasets across projects that you have access to.
- ▶ The console also provides several useful features such as saving and **sharing ad-hoc queries, tuning and editing historical queries**, exploring tables and schemas, and gathering table metadata.



The screenshot displays the Google BigQuery web interface. On the left, the 'COMPOSE QUERY' sidebar shows 'Query History' and 'Job History' sections, with a 'Filter by ID or label' input. Below these are 'Public Datasets' including 'gdelt-bq:hathitrustbooks', 'gdelt-bq:internetarchivebooks', 'googledata:buganizer', 'googledata:forbin', 'googledata:sponge', 'googledata:spore', 'lookerdata:cdc', 'nyc-tlc:green', and 'nyc-tlc:yellow'.

The main area is titled 'New Query' and contains a 'Query Editor' tab. The editor shows a SQL query:


```
1 SELECT
2   name, count(1) as num_repos
3 FROM
4   `bigquery-public-data.github_repos.languages`, UNNEST(language)
5 GROUP BY name
6 ORDER BY num_repos
7 DESC limit 5
```

 Below the editor, the 'Standard SQL Dialect' is selected, and a 'RUN QUERY' button is visible. To the right of the run button are 'Save Query', 'Save View', 'Format Query', and 'Show Options' buttons. A green checkmark icon indicates the query is ready to run.

Below the query editor, the 'Results' tab is active, showing a table with the following data:

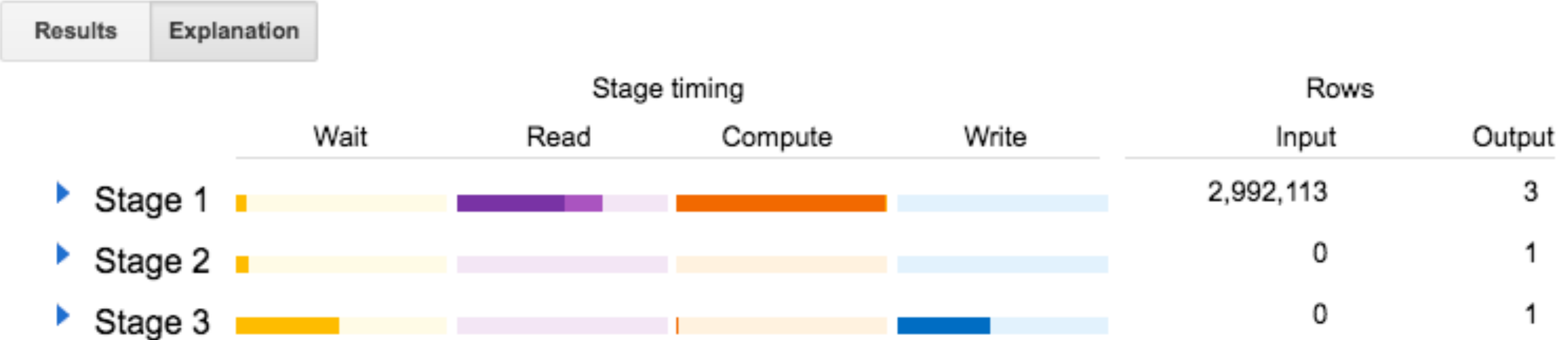
Row	name	num_repos
1	JavaScript	987058
2	CSS	728255
3	HTML	642442
4	Shell	583400
5	Python	484622

At the bottom of the results section, there are buttons for 'Download as CSV', 'Download as JSON', 'Save as Table', and 'Save to Google Sheets'. The 'Table' and 'JSON' tabs are visible at the very bottom.

Query Optimisation

- ▶ Each time BigQuery executes a query, it executes a **full-column scan**.
- ▶ BigQuery doesn't use or support indexes. Because BigQuery performance and query **costs are based on the amount of data** scanned during a query, design your queries so that they reference only the **columns that are relevant to the query**.
- ▶ When using date-partitioned tables, ensure only the relevant partitions are scanned. You can achieve this by using partition filters based on **PARTITIONTIME** or **PARTITIONDATE**.
- ▶ To understand the performance characteristics after a query executes, take a look at the detailed **query plan explanation**.
- ▶ The explanation breaks down the stages that the query went through, the number of input/output rows handled at each stage, and the timing profile within each stage.
- ▶ Using the results from the explanation can help you understand and optimise your queries.

Query Optimisation



BigQuery Pricing

- ▶ BigQuery offers scalable, flexible pricing options to meet your technical needs and your budget.
- ▶ Storage costs are based on the amount of data stored in BigQuery.
 - ▶ <https://cloud.google.com/bigquery/pricing>

BigQuery Public Datasets

- ▶ A public dataset is any dataset that is stored in BigQuery and made available to the general public through the **Google Cloud Public Dataset Program**.
- ▶ The public datasets are datasets that BigQuery hosts for you to access and integrate into your applications.
- ▶ **Google pays for the storage** of these datasets and provides public access to the data via a project.
- ▶ **You pay only for the queries** that you perform on the data. The first 1 TB per month is free, subject to **query pricing details**.

BigQuery Public Datasets

- ▶ The Cloud Public Datasets Program catalog is in [GCP Marketplace](#).
- ▶ You can find more details about each individual dataset by viewing the Marketplace pages in the Datasets section.

BigQuery web UI

- ▶ There are two user interfaces that can be used to access the public datasets:
 - ▶ The BigQuery web UI - <https://console.cloud.google.com/bigquery>
 - ▶ The classic web UI
- ▶ The bigquery-public-data project is automatically pinned to every project in both UIs.
You can find the project in the navigation pane.
- ▶ Quickstart using the web UI in the GCP Console
 - ▶ <https://cloud.google.com/bigquery/docs/quickstarts/quickstart-web-ui>
- ▶ BigQuery Documentation
 - ▶ <https://cloud.google.com/bigquery/docs/>