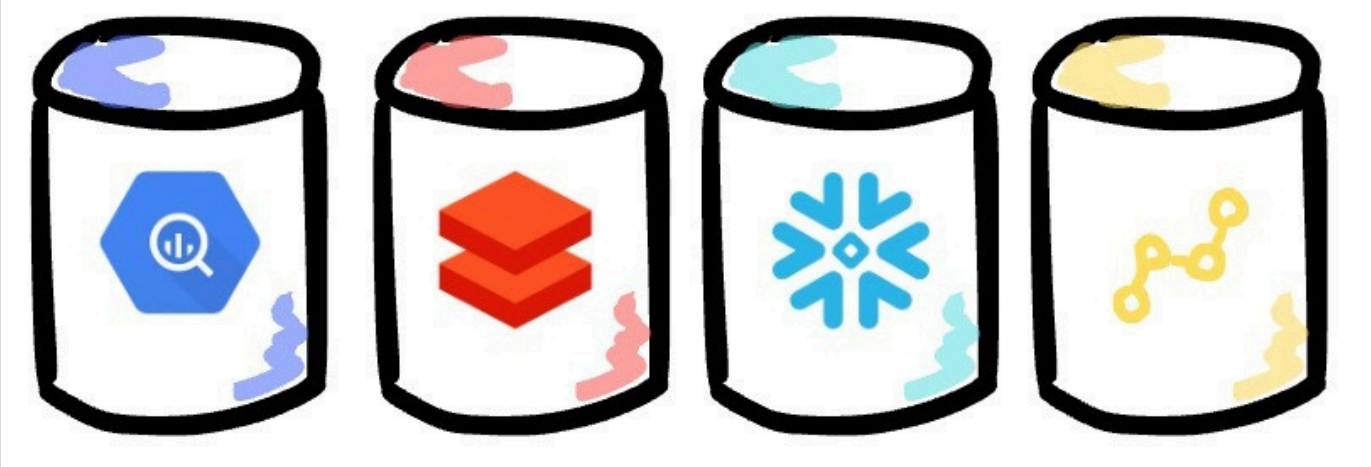


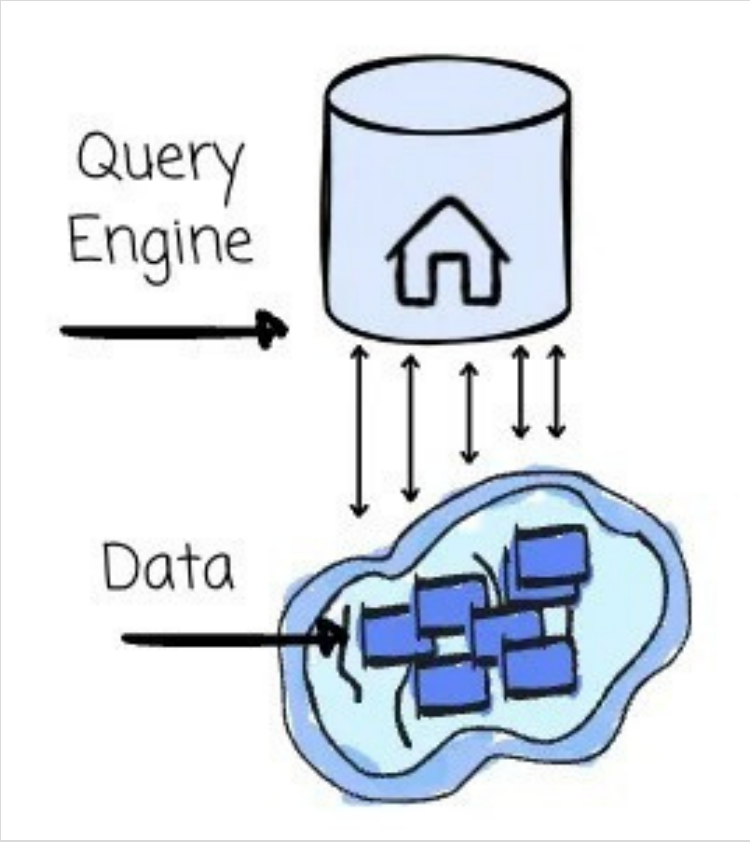
The internal of BigQuery, Snowflake, Databricks and Redshift

I spent a lot of time researching and learning about OLAP systems, especially cloud data warehouse solutions like BigQuery, Snowflake, Databricks, and Redshift.

I want to summary of what I researched. I hope my work could give you a good starting point when you begin to learn a completely new cloud data warehouse.

Cloud data warehouse

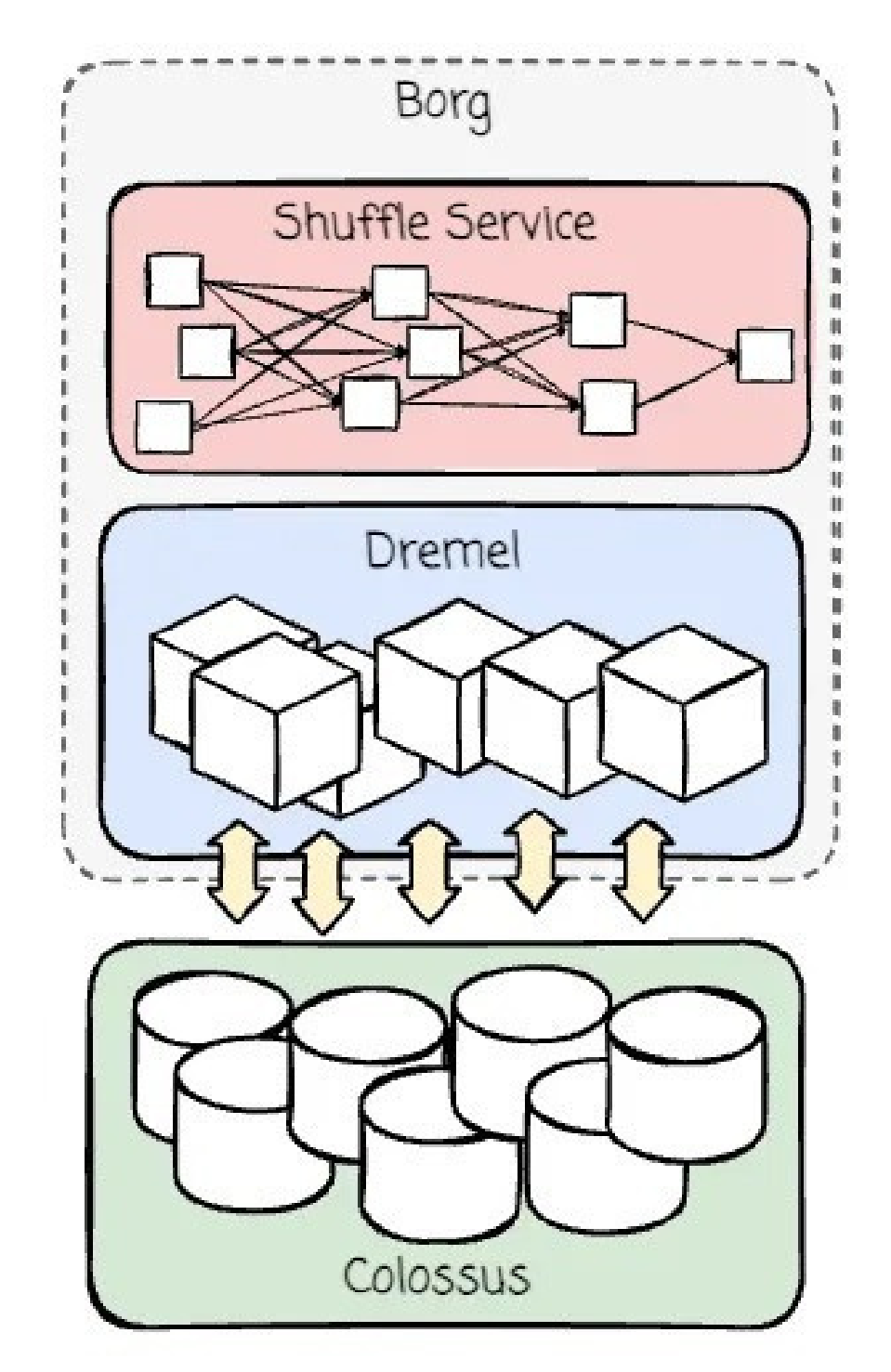
The 2010s witnessed the emergence of the cloud-native shared-disk architecture OLAP system with pioneers like Google BigQuery (2010) and Snowflake (2012).



Google BigQuery

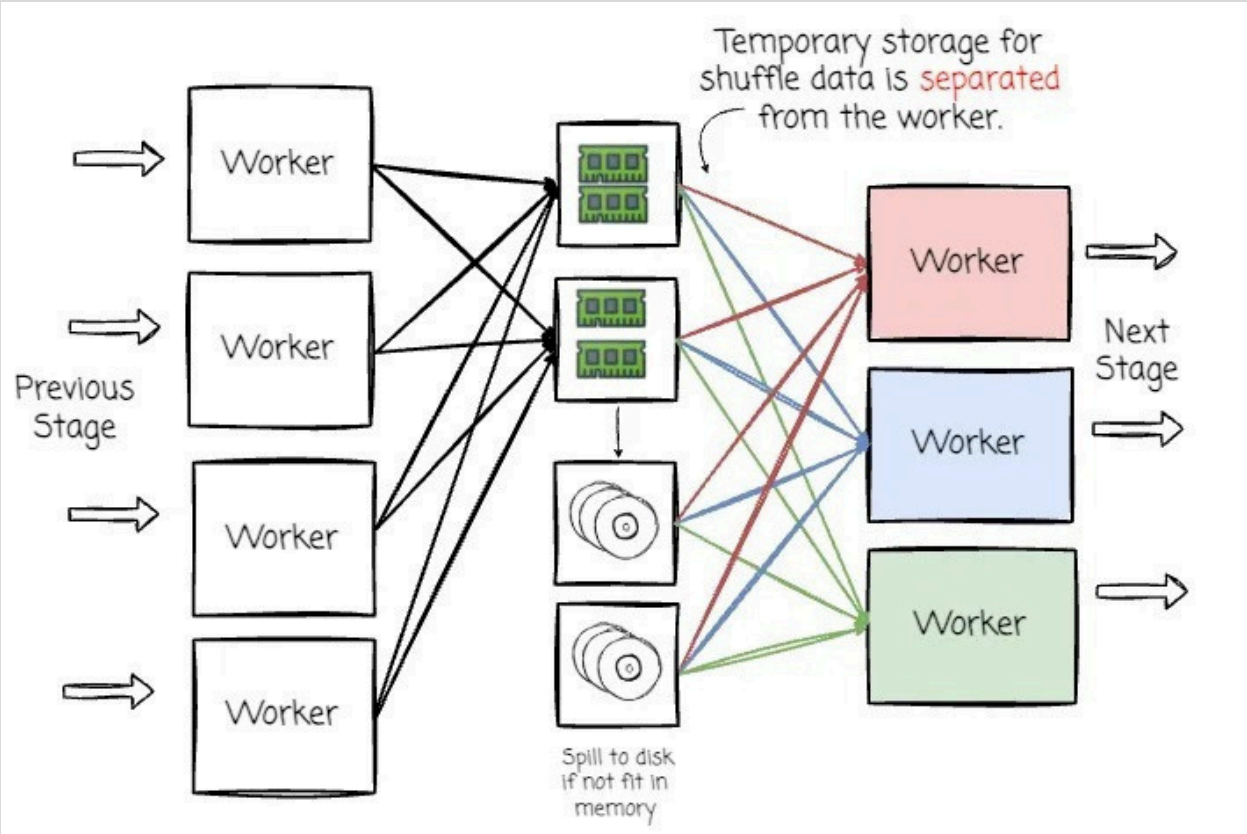
- Combination of many technologies Colossus for storage, Borg for computing management (think Kubernetes), and Dremel for query processing engines.
- In the beginning, Dremel operated on a few hundred shared-nothing servers.
- They gradually shifted to a shared-disk architecture, which leverages the Google File System (GFS), and later migrated to Colossus, the successor to the GFS.

Google BigQuery



Google BigQuery

- Dremel is inspired by MapReduce
  It had issues with data shuffling when
- dealing with large amount of data.
  To solve this, Google store shuffle data separately
-

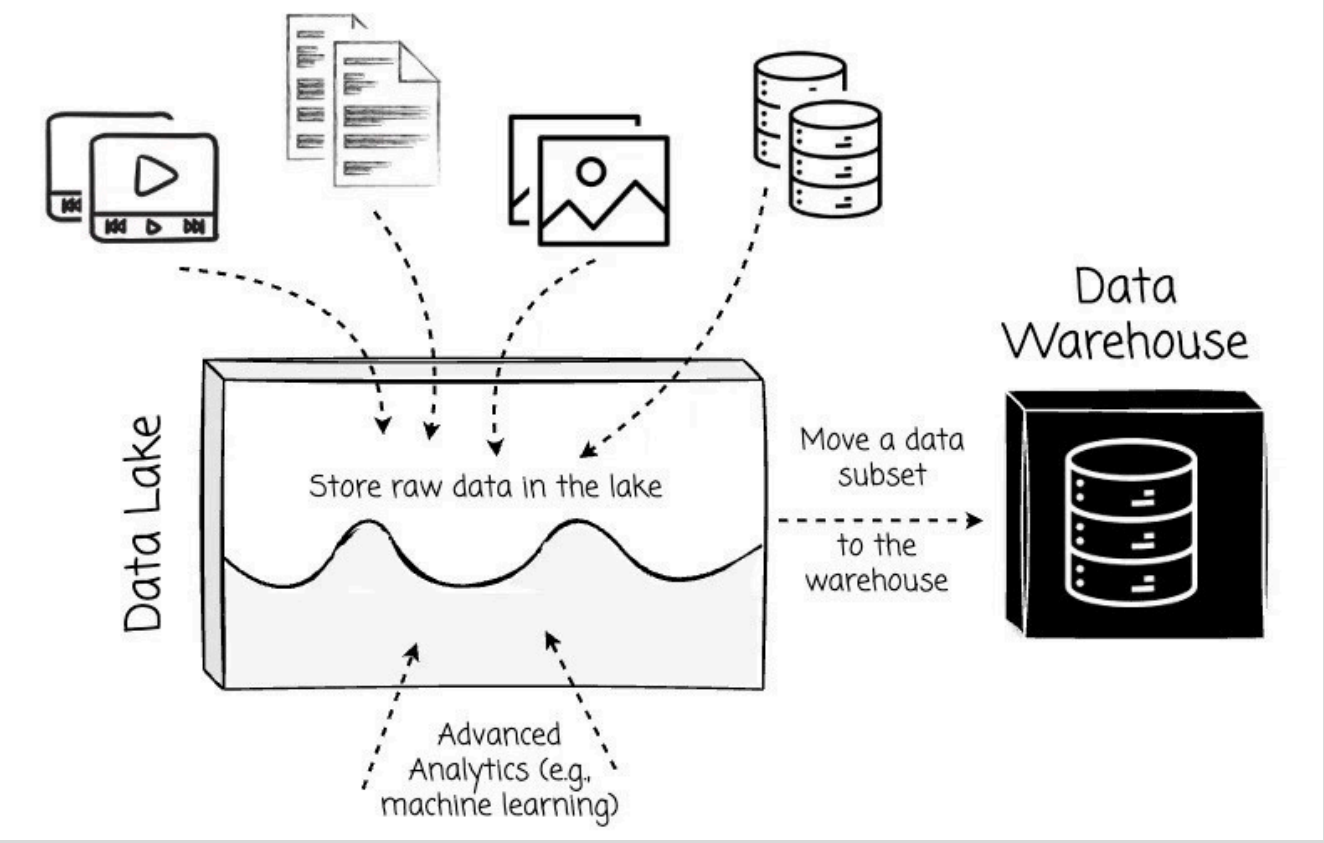


Google BigQuery

- They developed an internal data format called Capacitor. From a high level, it organizes data in a hybrid format, like Parquet.
  Capacitor has metadata to help query
- engines prune unnecessary data (e.g., min-max values of a column)
  It applies techniques like Run Length Encoding (RLE) or Dictionary encoding to
- optimize storage space.

Databricks

They have been offering the managed Lakehouse solution, which uses Delta Lake for the storage layer and Spark for the query engine.



Databricks

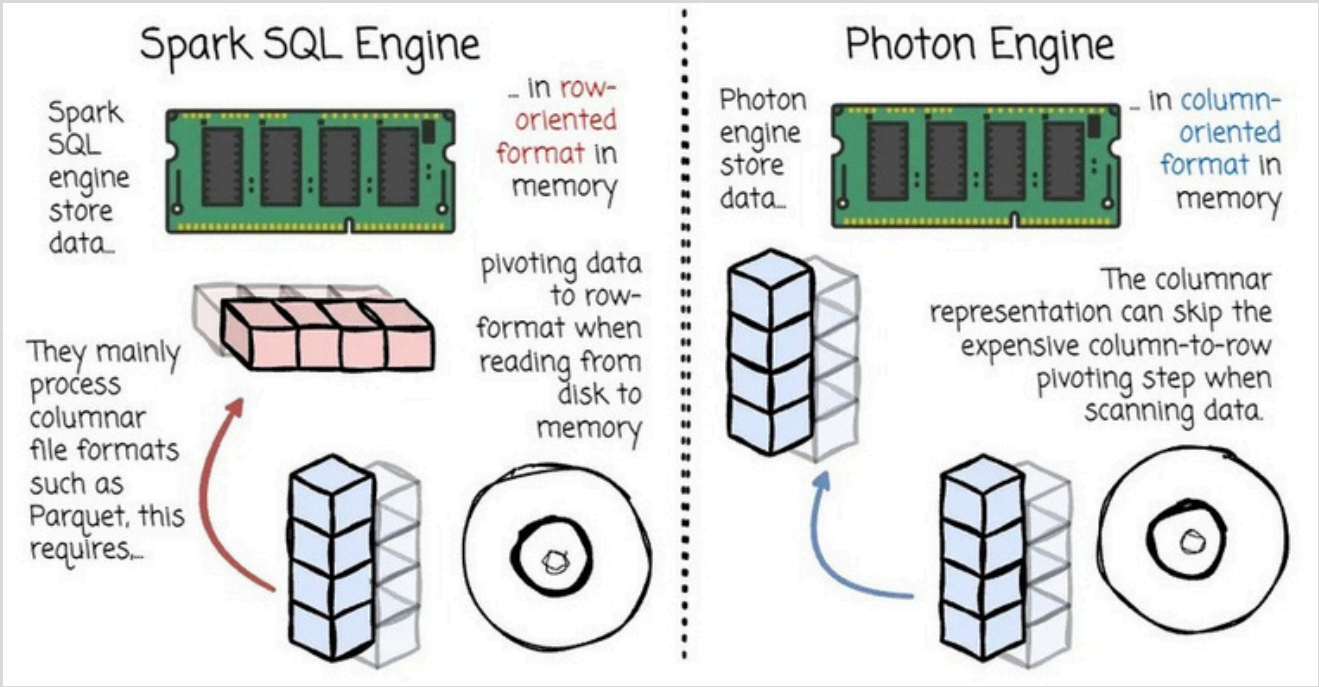
- Databricks built the Databricks Runtime (DBR), a fork of Apache Spark with enhancements for reliability and performance. But they need a little more than that.
  But they need a little more than that.
- They built the Photon engine, a library that integrates closely with the DBR.
- The engine acts as a new set of physical operators inside the DBR.
-

Databricks

- The system can run the queries partially in Photon; if it needs unsupported operations, they are switched back to SparkSQL.
  Photon is writing it in C++ instead of following the Spark JVM approach
- It use vectorized model instead of the Spark's code generation implementation
-

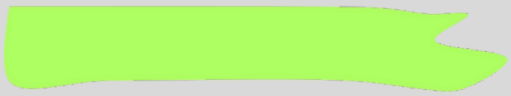
Databricks

Photon adopts columnar in-memory data representation; the system stores values of a particular column contiguously in memory.



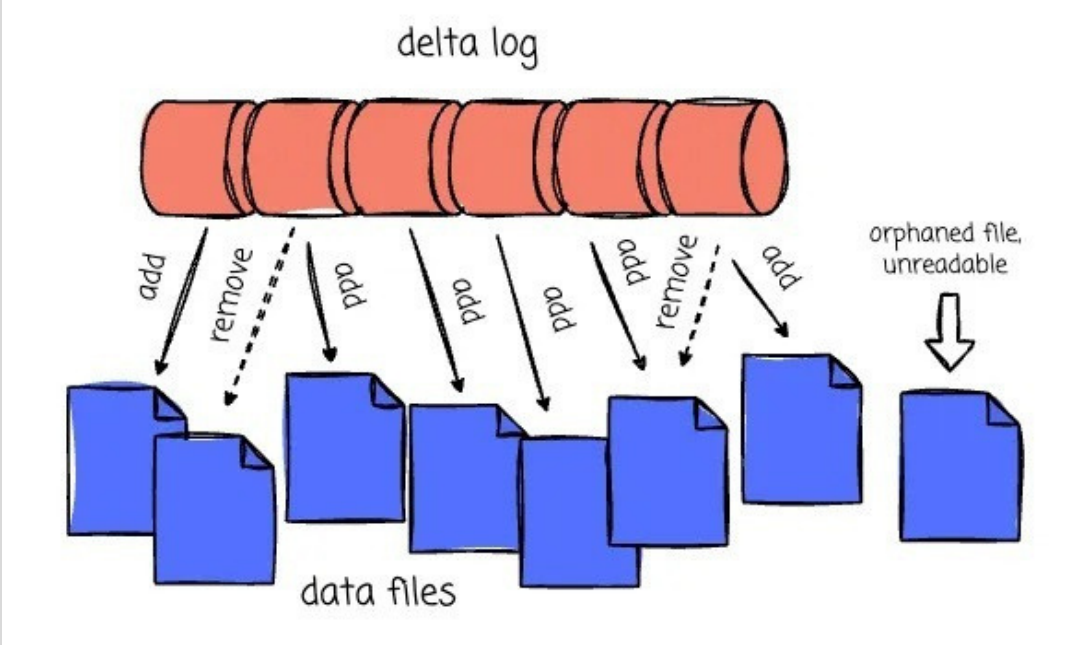
Databricks

- Databricks suggests users store data in Delta Lake, an ACID table storage layer on cloud object storage. A Delta Lake table is the cloud object storage directory or file
- system that consists of data objects and a log of transaction operations. Delta Lake identifies which object belongs to which table's version using the transaction log.

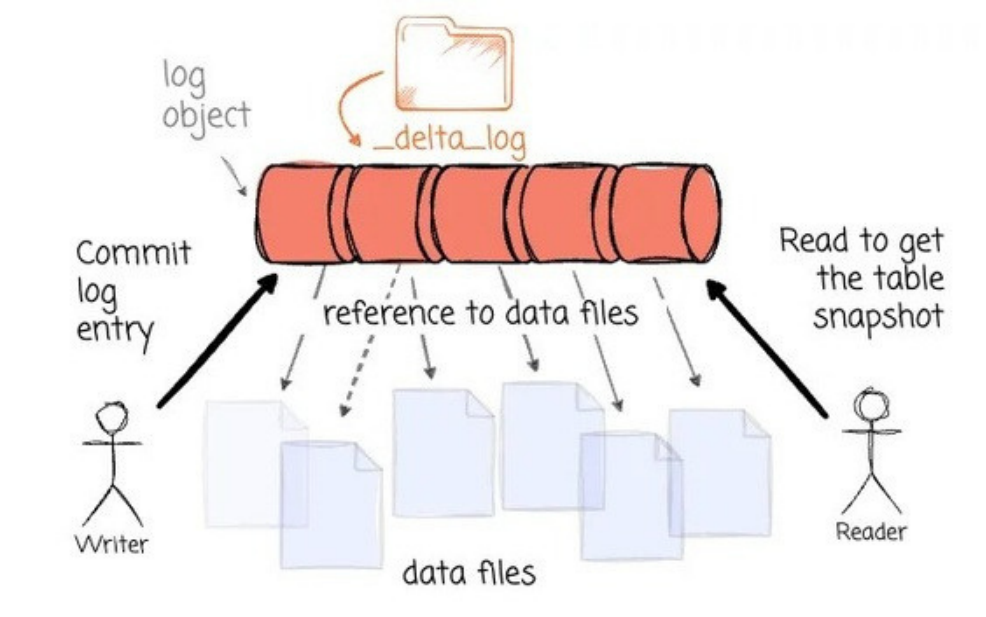


Databricks

Delta Lake table

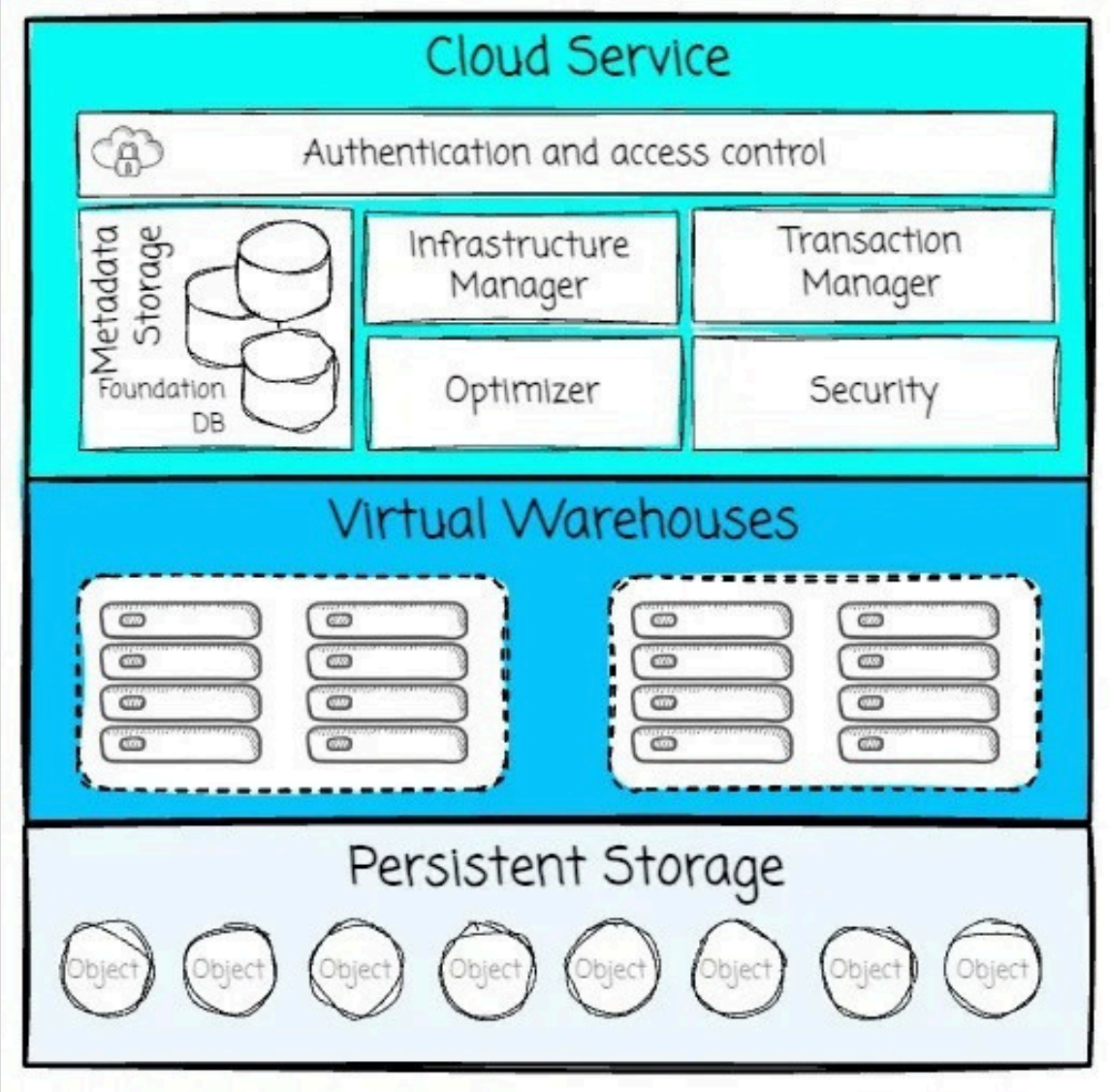


Delta Lake transaction log



Snowflake

Snowflake was founded in July 2012 by Benoit Dageville and Thierry Cruanes, two ex-Oracle engineers, and Vectorwise co-founder Marcin Zukowski.



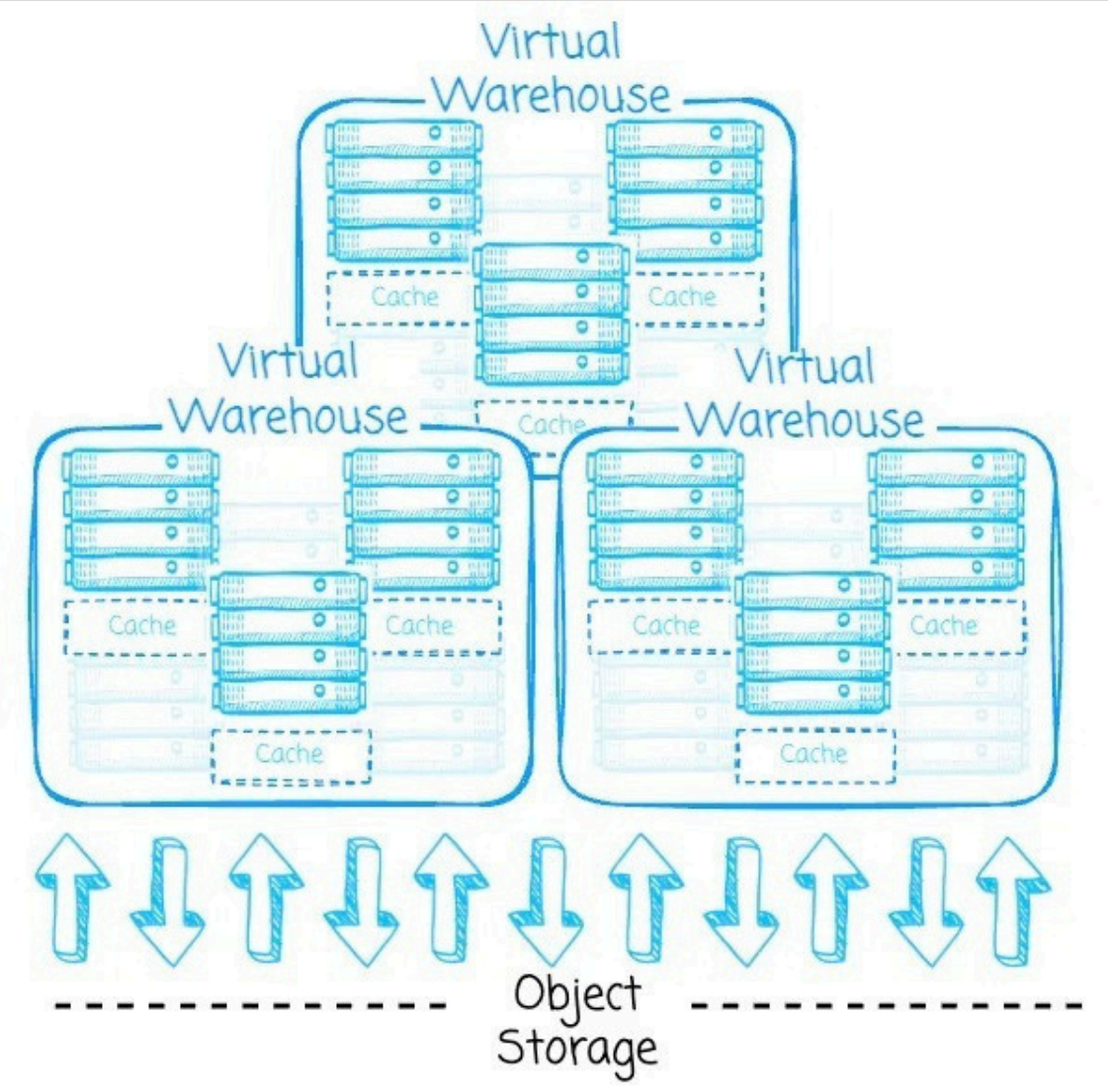
Snowflake

- They built a new OLAP database in C++. Their solution separates computing and storage, such as BigQuery or Databricks. Compute power comes from Snowflake's
- proprietary shared-nothing engine, which uses cloud virtual machines. For storage, Snowflake relies on object storage Snowflake uses local disk for data caching
- to enhance query performance by reducing API calls to object storage.

-

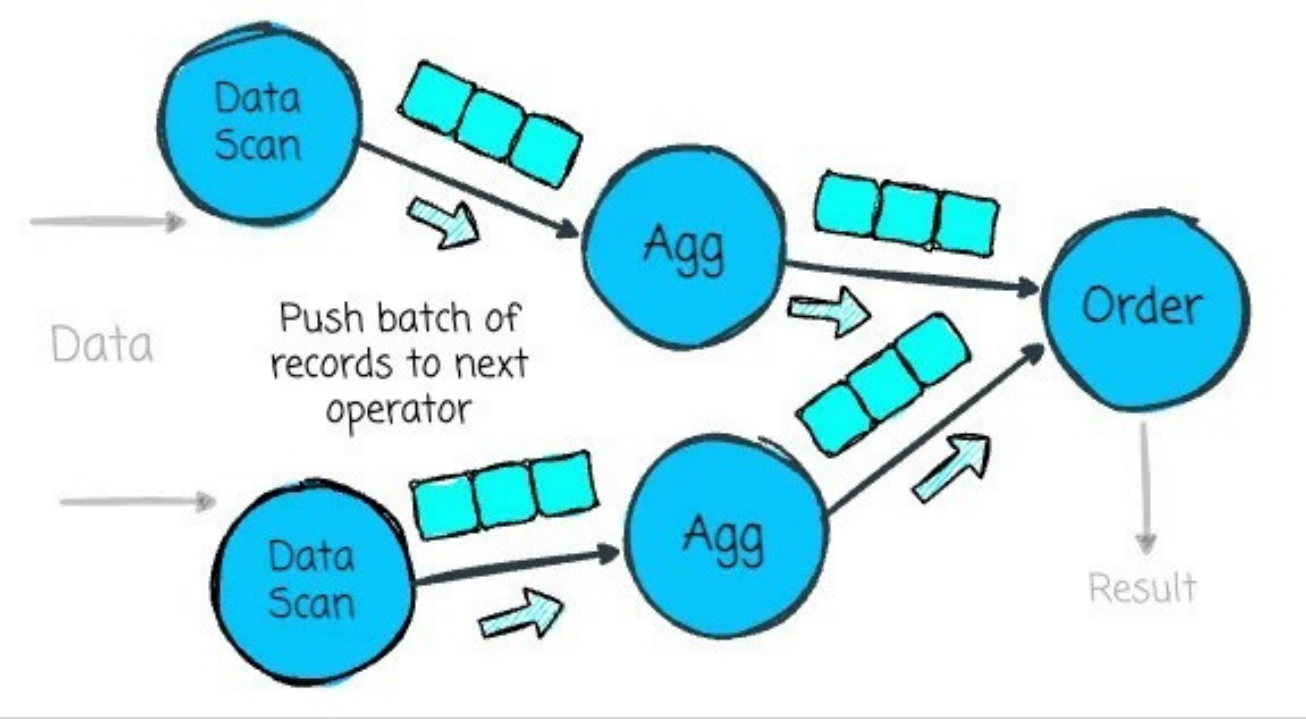
Snowflake

Snowflake introduced the concept of Virtual Warehouses (VW), essentially clusters of cloud virtual machine instances. Each instance in a cluster is referred to as a worker node.



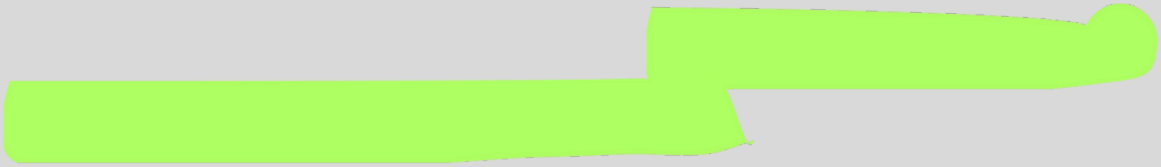
Snowflake

Snowflake employs vectorized execution, processing data in batches of thousands of rows in column format.



Snowflake

- It uses consistent hashing to improve cache hit rates and minimize redundant caching across multiple worker nodes within a VW. This process assigns table files to worker nodes based on file names,
- ensuring that queries accessing the same data will likely hit the same node.



Snowflake

- The team behind Snowflake had to choose between using object storage like S3 or building their solution on HDFS (or similar systems).
  After some experiments, they concluded
- that S3 excelled in availability and durability despite its unpredictable performance;
  They opted for object storage and focused on improving the performance of local
- caching and optimizing it with their proprietary storage format.

Snowflake

- Snowflake partitions table data into large, immutable files, similar to blocks or pages in a traditional database. Column values are grouped and heavily compressed in each file, equivalent to the hybrid file format. (hybrid format)

- It's important to note that when Snowflake was built in 2012, formats like Parquet and ORC, which were introduced in 2013, did not yet exist.

Redshift

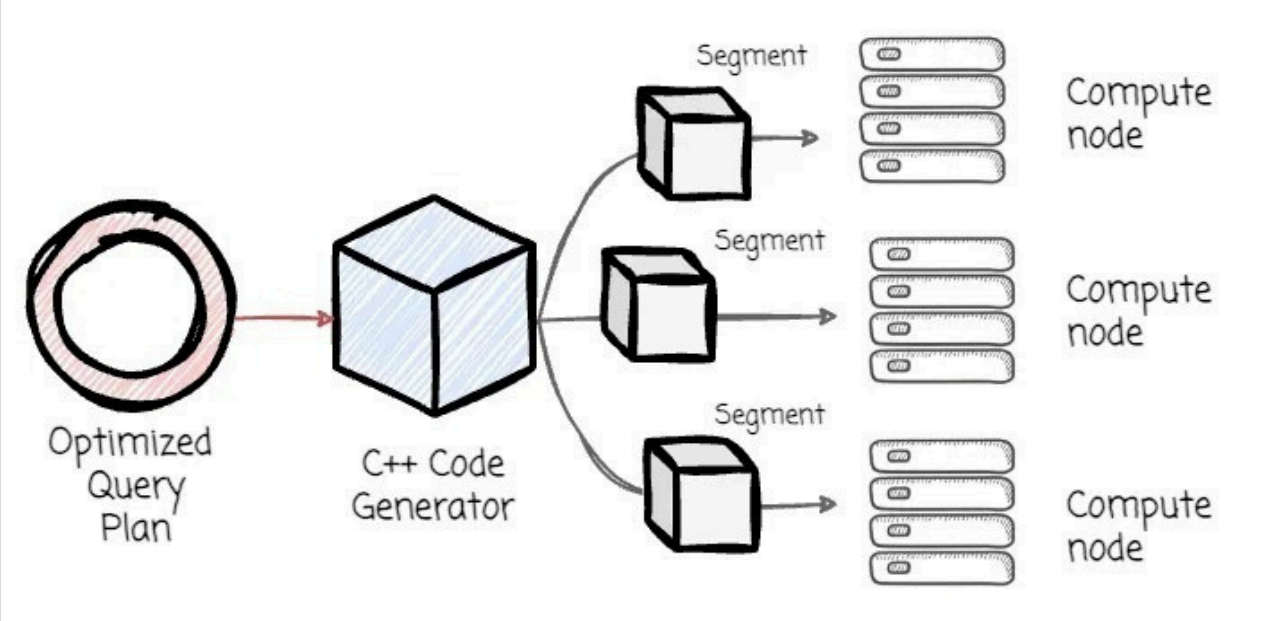
- Amazon Redshift is a column-oriented massively parallel processing data warehouse designed for the cloud. The system is built on top of technology from
- ParAccel (later acquired by Actian). It is based on an older version of PostgreSQL 8.0.2, and Redshift has made changes to that version. Redshift is a special case because it was initially designed with a share-nothing architecture. Later, they
- introduced Redshift Managed Storage (RMS), which leverages Amazon S3 behind the scenes.

Redshift

- A Redshift cluster consists of multiple compute instances that handle query execution.
  Each cluster has a single coordinator
- node (a.k.a. leader) and multiple worker nodes.

Redshift

- Redshift has applied the code generation approach. The system generates C++ code specific to the query plan and the executed schema. The generated code is then compiled, and the binary is delivered to the compute nodes for execution.



Redshift

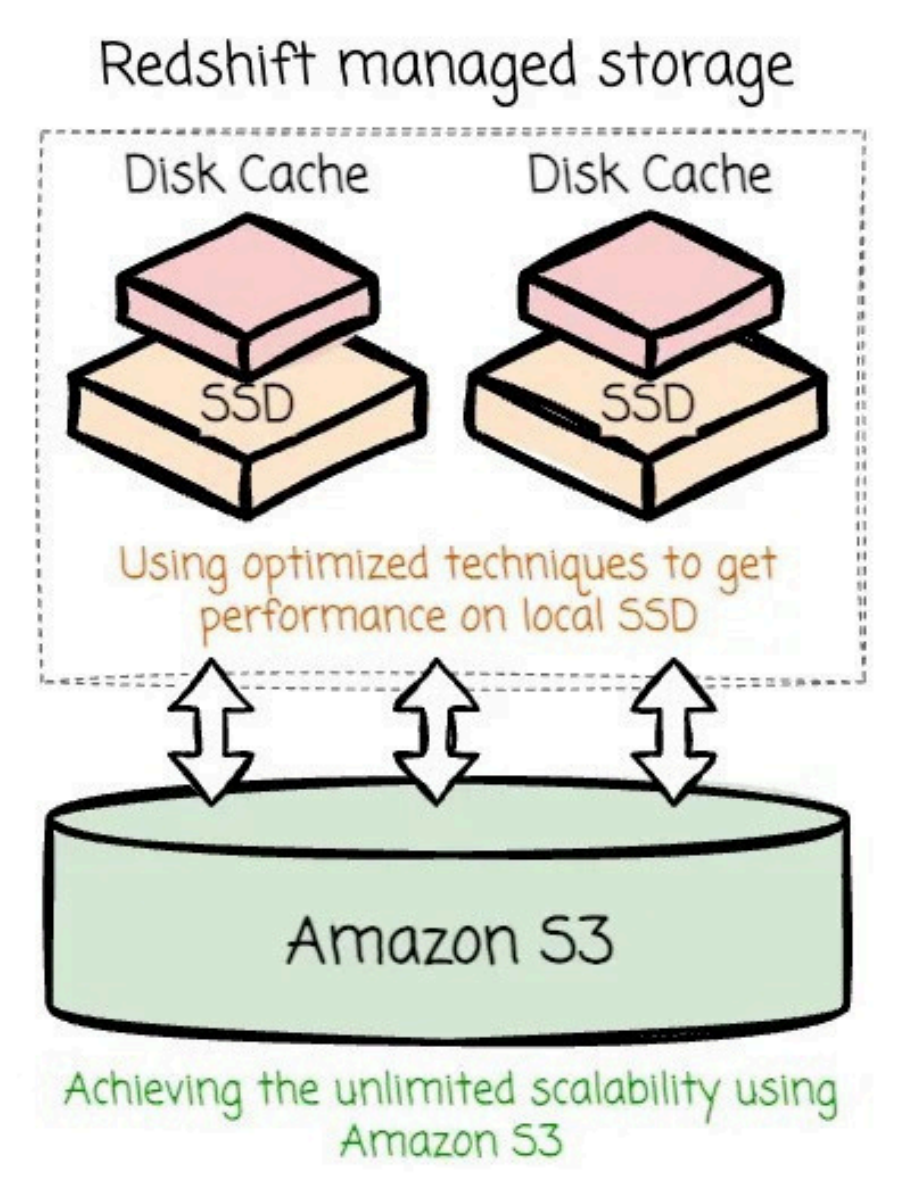
- Redshift will use the compiled optimized objects for the query execution.
  These objects will be cached in the local
- cluster cache, so whenever the same or similar queries are executed, the compiled objects are reused.
  Later, Redshift release the compilation service, which uses separate resources
- instead of cluster resources to cache compiles objects

Redshift

- As mentioned, data is offloaded to RMS, which is based on Amazon S3.
- To identify which worker node is in charge of which subset of data in RMS, Redshift partitions the table's data into multiple buckets distributed to all worker nodes.

Redshift

- Like Snowflake, Redshift caches data on worker nodes' local SSD to improve query performance.



Redshift

- Regarding storage format, instead of storing data in a hybrid format like the three data warehouses above, Redshift stores the values of each table column together. This allows Redshift to pack data together and apply compression to
- minimize disk I/O during query execution. A row can be stitched together by utilizing the offset of a specific value.