**Snowflake architecture — the three layers (what each does)**

1. **Database storage (persistent layer)**
   * Stores customer data in cloud object storage (AWS S3, Azure Blob Storage, or GCP).
   * Data is organized into **micro-partitions** (small immutable, columnar blocks), compressed and tracked by metadata so Snowflake can prune irrelevant partitions for queries. Micro-partitions enable fast scans, zero-copy cloning, time-travel and efficient storage. [Snowflake Documentation](https://docs.snowflake.com/en/user-guide/tables-clustering-micropartitions?utm_source=chatgpt.com)[Select](https://select.dev/posts/snowflake-micro-partitions?utm_source=chatgpt.com)
2. **Query processing / Compute layer (virtual warehouses)**
   * Compute is provided by *virtual warehouses* — MPP compute clusters that perform queries, DML, and data loads/unloads. Warehouses are stateless and can be started/stopped/resized on demand; each warehouse is isolated (workloads run without interfering with each other). [Snowflake Documentation](https://docs.snowflake.com/en/user-guide/warehouses?utm_source=chatgpt.com)
3. **Cloud services (control/coordination layer)**
   * Centralized services for authentication, metadata/catalog, query parsing & optimization, security, transaction management, and infrastructure orchestration. This layer doesn’t do heavy data processing but coordinates everything and stores metadata separately from compute.

The architecture of Snowflake is a combination of standard **shared-disk and shared-nothing database technologies**. Snowflake, like shared-nothing architectures, processes queries utilizing **MPP (massively parallel processing)** compute clusters, in which each node maintains a fraction of the full data set locally. This technique combines the ease of data management of a shared-disk design with the performance and scale-out advantages of a shared-nothing architecture.

Snowflake's unique architecture design enables users to automatically request for storage, pay for exclusive resources, and enjoy the benefits of a well-managed cloud data warehouse This also has **automatic scaling**, **data sharing**, **multi-cluster**, **multi-cloud** architecture For increased flexibility and efficiency it offers.

**What is the Snowflake Data Warehouse**

Snowflake Data Warehouse is a **cloud-based data warehousing** platform that is designed for scalable and efficient storage and analysis of datasets. It contains a unique architecture with separate storage and computing resources. Snowflake supports multi-cluster, **multi-cloud deployment** which enables us to choose a preferred cloud provider. It offers robust security measures, including encryption and role-based access control. With features like zero-copy cloning and data sharing, Snowflake facilitates agile data management and collaboration.

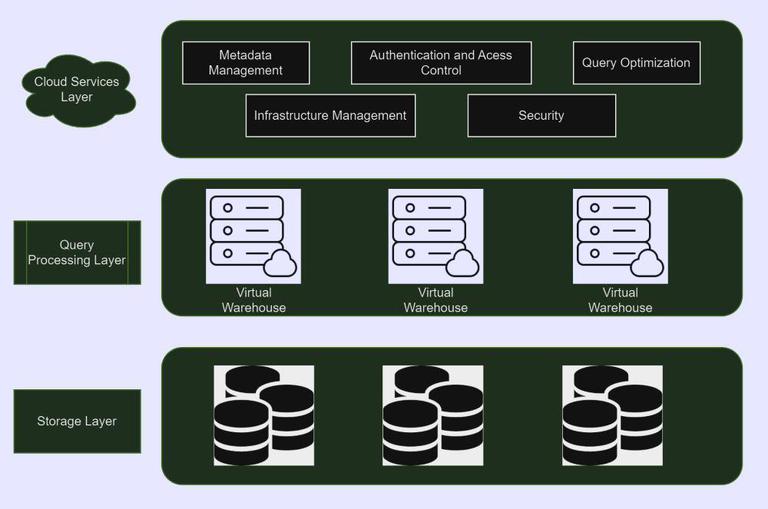
**Components of Data Warehouse Architecture**

These are the general components of that any data warehouse architecture contains but this may vary based on the technology and vendor. They are

1. **Operational Data Sources:** These systems are used to store operational data such as customer transactions, sales, etc. These are also responsible to data that are accumulated from external data sources.
2. **Database:** These are going to work as temporary storage units under data warehouse which are used as staging areas to store raw data and the central repository based databases are use as storage units of cleansed data.
3. **Data Warehouse Server:** In general this server is responsible to store and retrieve the data from Data Warehouse. It contains Query Processor which is responsible for processing [SQL](https://www.geeksforgeeks.org/sql/sql-tutorial/)queries and a metadata repository to store metadata.
4. **Security and Authentication:** This components is going to ensure data security as well as security to the cloud data warehouse by checking whether a particular user has access or not
5. **Backup and Recovery:** This component is used to backup the data warehouse to prevent data loss. It helps in case of system failures or data corruption.
6. **Monitoring:** Every data Warehouse is going to have monitor component which is used to monitor the performance and costs that are accumulated. This ensures one to perform performance tuning to optimize queries.

**Snowflake's Architecture**

Snowflake’s architecture mainly consists of three layers.

Snowflake Architecture

**Storage Layer**

The Storage layer in snowflake architecture is responsible for managing and storing data in an effective manner. The functionalities that were supported by the storage layer are:

* **Elasticity:** Snowflake's storage layer is elastic, allowing organizations to scale their storage needs independent of compute resources. It ensures to handle various data volumes without affecting performance.
* **Cloud Based Object Storage:** Snowflake uses cloud based object storage to store data. This separation of storage and compute allows for cost-effective and scalable data storage.
* **Data Clustering:** Snowflake organizes data into micro partitions within the object storage, and these micro partitions are clustered based on metadata. This clustering enhances query performance by minimizing the amount of data that needs to be scanned.
* **Zero Copy Cloning:** Snowflake enables efficient data cloning through zero-copy cloning technology. This feature allows users to create a copy of a dataset instantly without duplicating the actual data, saving both time and storage costs.

**Query Processing Layer**

The SQL query execution is handled by Snowflake's Query Processing Layer, which **dynamically optimizes** and parallelizes queries over several compute clusters. It ensures great performance and scalability by decoupling computation and storage, allowing for on-demand resource allocation based on query complexity and workload. Functionalities of Query Processing Layer are:

* **Automatic Query Processing:** Snowflake's Query Processing Layer optimizes **SQL** queries automatically, modifying execution plans based on underlying data distribution and query complexity to ensure efficient processing.
* **Parallel Execution across Clusters:** Query execution is performed in parallel across many compute clusters, leveraging Snowflake's **multi-cluster architecture** to achieve high concurrency and faster results for complex analytical workloads.
* **On Demand Resource Allocation:** Depending on the complexity and number of queries, the Query Processing Layer dynamically distributes computational resources as needed. This on-demand resource distribution provides peak performance and cost efficiency.
* **Compute and Storage Separation:** Snowflake's architecture separates computing and storage, allowing the Query Processing Layer to expand compute resources independently. This separation improves flexibility by allowing enterprises to change computer power without affecting stored data, so optimizing both performance and prices.

**Cloud Services Layer**

In Snowflake's architecture, the Cloud Services Layer serves as the control plane, managing information, security, and user access. It serves as a **centralized platform** for administration, authentication, and activity coordination across the data warehouse. This layer ensures that users and the underlying computation and storage resources in a cloud environment interact seamlessly. The functionalities of Cloud Services Layer are:

* **Metadata Management:** Snowflake's metadata management involves storing comprehensive information about data objects, structures, and statistics, facilitating efficient query optimization. This metadata layer is crucial for dynamically organizing and processing data within the cloud-based data warehousing platform.
* **Authentication and Access Control:** Snowflake employs robust authentication methods, including multi-factor authentication, to secure user access. Access control is granular, with role-based permissions and policies ensuring fine-tuned control over data and system resources.
* **Query Optimization:** Snowflake's query optimization dynamically adjusts execution plans based on data distribution and complexity, ensuring efficient processing of SQL queries. It leverages a multi-cluster, parallel processing architecture for faster and scalable query performance.
* **Infrastructure Management:** Snowflake automates infrastructure management by dynamically allocating and deallocating computing resources based on workload demands, ensuring optimal performance and cost efficiency. This approach simplifies operations for users by abstracting the complexities of underlying cloud infrastructure.
* **Security:** Snowflake prioritizes security with end-to-end encryption, role-based access controls, and features like data masking, ensuring comprehensive protection of sensitive data within the cloud-based data warehousing platform. Security measures are integrated at every level, safeguarding against unauthorized access and data breaches.

**Conclusion**

Snowflake's three-layer architecture includes a compute layer, storage layer, and cloud services layer. The compute layer handles query processing, the Storage layer manages statistics storage, and the cloud services layer guarantees metadata control and coordination. This separation permits scalability, flexibility, and green facts processing in a cloud-primarily based statistics warehouse.

Copy command:

Use employee.csv file for loading

**1. Create Table**

CREATE OR REPLACE TABLE employees (

id INT,

name STRING,

department STRING,

salary NUMBER(10,2)

);

**2. File Format**

CREATE OR REPLACE FILE FORMAT my\_csv\_format

TYPE = 'CSV'

FIELD\_OPTIONALLY\_ENCLOSED\_BY = '"'

SKIP\_HEADER = 1;

**3. Stage**

CREATE OR REPLACE STAGE my\_stage

FILE\_FORMAT = my\_csv\_format;

**4. Load Data**

COPY INTO employees

FROM @my\_stage/employees.csv

FILE\_FORMAT = (FORMAT\_NAME = my\_csv\_format)

ON\_ERROR = 'CONTINUE';

**5. Verify**

SELECT \* FROM employees;

**Continuous Loading**

**1. Create a Table for Your CSV**

First, check the CSV structure (columns). Let’s assume it has:

id, name, city, age, department

You can create a table like this:

CREATE OR REPLACE TABLE employees (

id INT,

name STRING,

city STRING,

age INT,

department STRING

);

**🔹 2. Create a File Format**

Define how Snowflake should read your CSV:

CREATE OR REPLACE FILE FORMAT my\_csv\_format

TYPE = 'CSV'

FIELD\_OPTIONALLY\_ENCLOSED\_BY='"'

SKIP\_HEADER = 1;

**🔹 3. Stage the File**

Put the CSV into a **stage**. For example, using an **internal stage**:

CREATE OR REPLACE STAGE mystage;

**🔹 4. Create a Snowpipe (Continuous Loader)**

This tells Snowflake to **auto-load** whenever new files are added to the stage:

CREATE OR REPLACE PIPE employees\_pipe

AS

COPY INTO employees

FROM @mystage

FILE\_FORMAT = (FORMAT\_NAME = my\_csv\_format);

**🔹 5. Enable Auto-Ingest (Optional for Cloud Storage)**

* If you are using **AWS S3, Azure Blob, or GCP storage**, you can set **event notifications** → Snowpipe will load files automatically.
* If using **internal stage**, you can trigger manually:

ALTER PIPE employees\_pipe REFRESH;

This checks the stage and loads any new/unloaded files.

**🔹 6. Monitor Snowpipe**

See the status of files loaded:

SELECT system$pipe\_status('employees\_pipe');

✅ After this, whenever you add new employees.csv files (like employees\_2025\_08\_30.csv, etc.) into the stage, Snowpipe will automatically load them into your **employees** table.