**Diff type of file formats:**

**CSV:**

CSV stands for Comma Separated values. A CSV file is a plain text file that stores data in comma-separated values. each line of text represents a row.

**Advantages:**

**Human-Readable:** CSV files are plain text files, making them easy to read and edit.

**Widely Supported**: Almost all data processing tools can support CSV files.

**Simple Structure**: Each row represents a record, and columns are separated by commas.

**Lightweight:** CSV files have a small storage.

**Disadvantages:**

**Inefficient for Large Data**: CSV files can be slow to read, especially when dealing with large datasets.

**Limited Data Types:** CSV supports only basic data types (strings, numbers)

**JOSN:**

JSON stands for JavaScript Object Notation. A JSON file is a text-based format used to store structured data. It consists of key-value pairs enclosed in curly braces.

**Advantages:**

**Human-Readable:** JSON files are easy to read and understand.

**Semi-Structured Data**: JSON files support semi-structured data.

**Flexibility:** JSON files support for nested objects, arrays, and key-value pairs, providing more flexibility.

**Widely Supported**: JSON is a popular format, almost all tools can support JSON files.

**Disadvantages:**

**Slower Processing:** JSON can be slower than other formats such as parquet.

**Complexity**: Nested structures can make processing more complex.

**File Size:** JSON files can have larger file sizes compared to Parquet.

**YAML:**

YAML (YAML Ain't Markup Language) is a human-readable data serialization format commonly used for configuration file.

 It is designed to be easy to read and write, making it popular for configuration management

**PARQUET:**

A Parquet file is a column-based file format used for storing structured data. Parquet files are stored in a binary format. Parquet files are commonly used in big data processing frameworks such as Apache Spark, Apache Hive.

**Advantages:**

**Columnar Storage**: Parquet files organize data by column -based format. It’s ensured the processing speed.

**Lightweight:** Parquet files are consuming less storage.

**Efficient Processing:** Parquet files are very fast processing compared to other files such as CSV, JSON etc.

**Disadvantages:**

**Not Human-Readable:** Parquet files are binary format not human readable file.

**Limited Tool Support:** Compared to other files, limited tools only supported parquet file.

**Delta lake:**

Delta Lake is an open-source storage layer designed to run on top of an existing data lake (ADLS, S3, HDFS) and improve its reliability, security, and performance.

* ACID Transactions
* Schema Enforcement and Evolution
* Time Travel (Data Versioning)
* Unified Batch & Streaming
* Optimized Performance
* Open Format & Compatibility

**ACID Properties**

* The **Atomicity** property ensures that a transaction is either executed completely or not at all.
* The **Consistency** property ensures that the database remains in a consistent state before and after a transaction.
* The **Isolation** property ensures that multiple transactions can run concurrently without interfering with each other.
* The **Durability** property ensures that the results of a committed transaction are permanent and cannot be lost due to system failure.

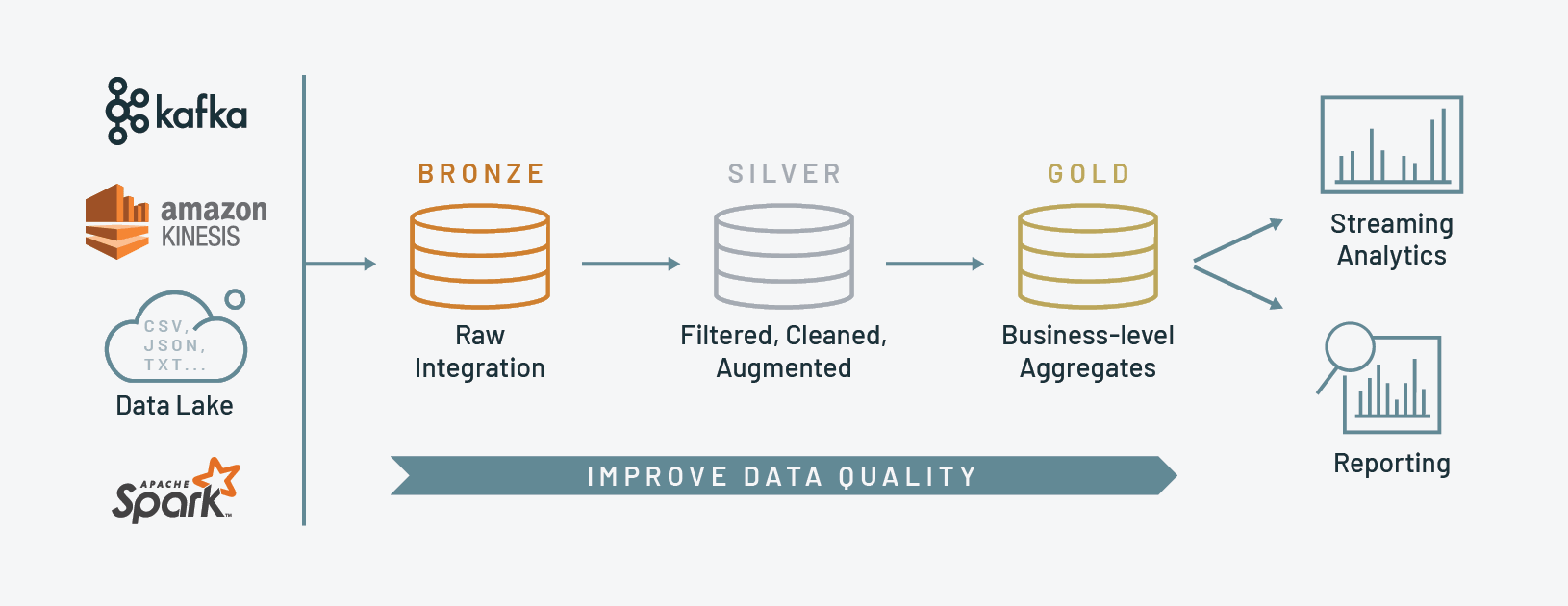
A diagram of a diagram

Description automatically generated

**Medallion Architecture:**

Medallion architecture is a data design pattern used to logically organize data in Lakehouse. Improving the structure and quality of data as it each layer of the architecture (Bronze 🡪 Silver 🡪 Gold)

Medallion architecture is also called ‘mutli-hop’ architecture.



**Bronze Layer:**

Bronze layer also known as the Staging layer; data is extracted from its original sources (raw data).

**Silver layer:**

Data from the Bronze layer undergoes a transformation in the silver layer. like filtered, cleaned, and validated the data.

**Gold layer:**

In gold layer, aggregates and prepares data for business use. It applies data quality standards and business logic to present data.

**Diff types of storage:**

**Data Warehouse:**

Data warehouse is used to storing the structure of data in a tabular format. Using SQL we can accessing the data.

Data warehouse is ensuring the ACID properties. It does not support the semi-structured data and unstructured data.

**Data Lake:**

Data Lake is used to storing the any kind of data, it can store structured, semi-structured, and unstructured data, but it does not support fully ACID properties.

**Delta Lake:**

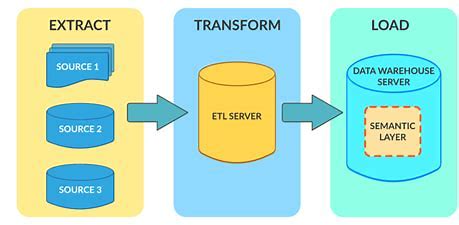
Delta Lake is stirring the structured data, semi-structured data and unstructured data and it is ensuring the ACID properties.

Delta Lake is an open-source storage layer designed to run on top of an existing data lake (ADLS, S3, HDFS) and improve its reliability, security, and performance.

* ACID Transactions
* Schema Enforcement and Evolution
* Time Travel (Data Versioning)
* Unified Batch & Streaming
* Optimized Performance
* Open Format & Compatibility

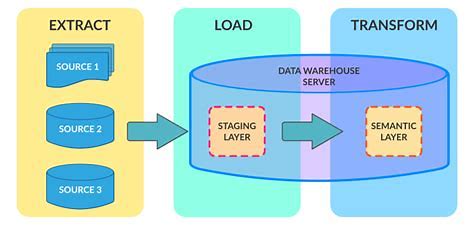
**ETL (Extract, Transform, Load):**

Data is first extracted from sources, then cleaned and transformed before loading into a data warehouse. This ensures high data quality but can be slower for large datasets.



**ELT (Extract, Load, Transform):**

Data is first extracted and loaded into storage, then transformed as needed. This is faster and more flexible, making it ideal for big data and cloud systems.



**Parquet:**

A Parquet file is a column-based file format used for storing structured data. Parquet files are stored in a binary format. Parquet files are commonly used in big data processing frameworks such as Apache Spark, Apache Hive.

**Delta lake:**

Delta Lake is an enhanced version of Parquet that adds ACID transactions, schema enforcement, and time travel, making it more reliable for big data and real-time analytics.

**Data Auditing:**

Data auditing is the process of monitoring data creation, collection, usage, storage, and destruction.

Also, it referred to as data risk management, it helps organizations improve data quality, identify and fix errors, assess data quality, and make informed decisions based on analytics.

**Data Governance**

Managing data within an organization to ensure accuracy, security, and proper usage. It focuses on data policies, access control, and quality management.

**Data Compliance**

Ensuring data follows legal, industry, and company regulations. It helps avoid penalties and ensures data privacy and security.

**Most functional tests follow the Arrange-Act-Assert model:**

1. **Arrange**, or set up, the conditions for the test
2. **Act** by calling some function or method
3. **Assert** that some end condition is true

Diff types of storage

Medallion Architecture

ACID Properties

Diff type of file formats

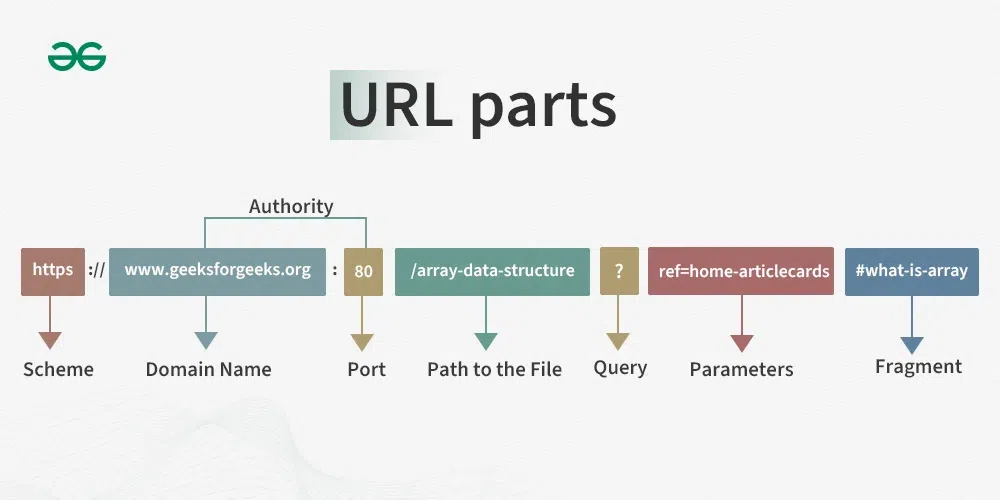
Data Auditing

ETL, ELT

Data governance and compliance

Delta and parquet difference

**Breaking Down the URL**

****

[**https://www.instagram.com/reel/DEm0\_\_ZsWX\_/?igsh=MXFhMzJsd3Q1c3R5bA==**](https://www.instagram.com/reel/DEm0__ZsWX_/?igsh=MXFhMzJsd3Q1c3R5bA==)

**Protocol: https**

This HTTPS stands for HyperText Transfer Protocol Secure. It is the protocol used for secure communication over a computer network. HTTPS is the secure version of HTTP

**Domain:**[**www.instagram.com**](http://www.instagram.com)

This is the domain name of the website. It is the address where the resource is located.

**Path: /reel/DEm0\_\_ZsWX\_/**

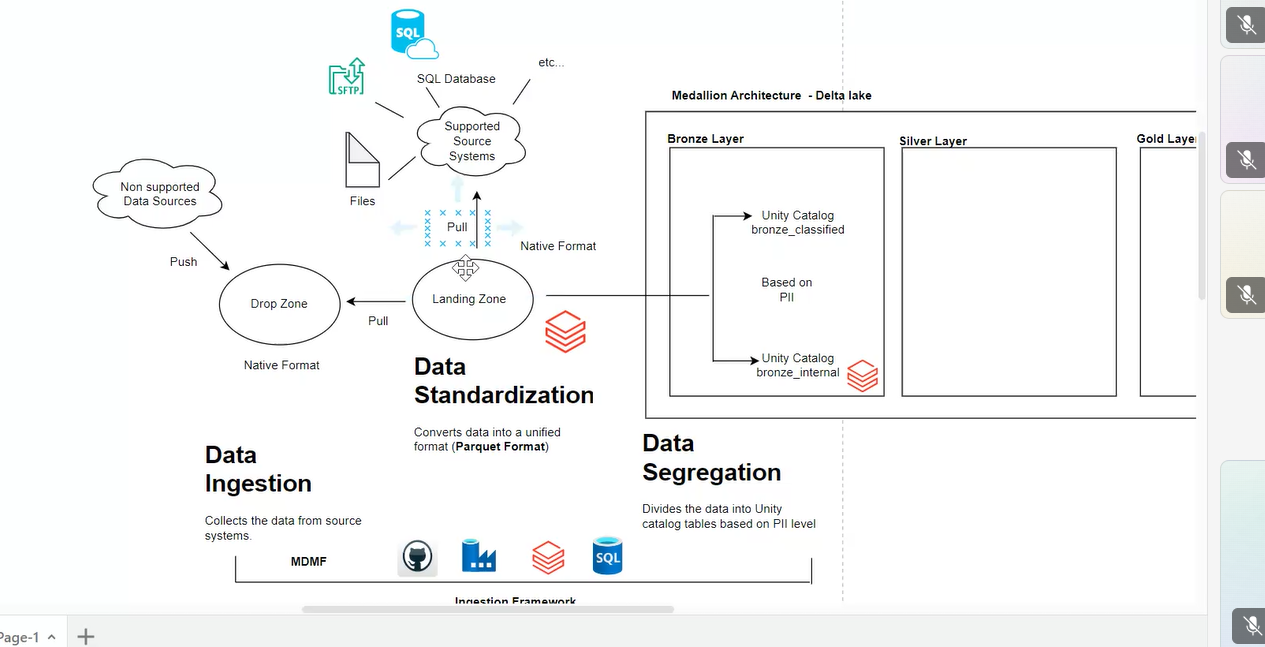
This part of the URL specifies the specific resource or endpoint on the server.

* /reel/ indicates that the resource is a reel (a type of video post on Instagram)
* /DEm0\_\_ZsWX\_ is a unique identifier for that specific reel.

**Query Parameters: ?igsh=MXFhMzJsd3Q1c3R5bA==**

Query parameters are used to pass additional information to the server. They are separated from the path by a ?

They are usually in the form of key-value pairs. In this case, igsh is the key, and MXFhMzJsd3Q1c3R5bA== is the value.



unity catalog

csv files (load into python)

Custom schema- pyspark

Data profiling

github actions

adf

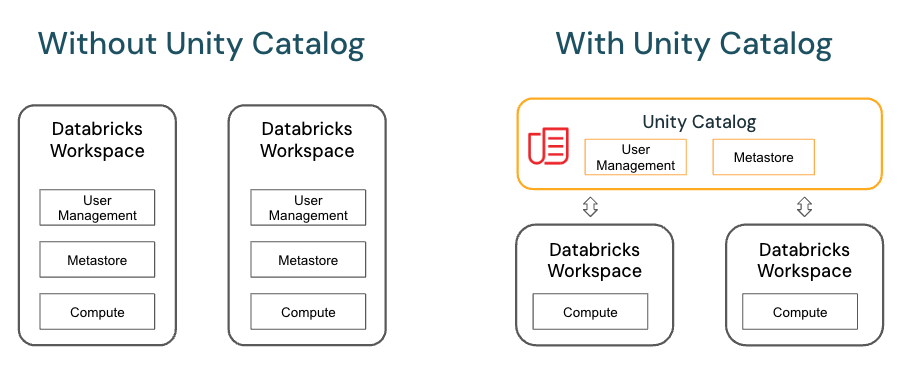
databricks - how to deal data

databricks - asset bundles

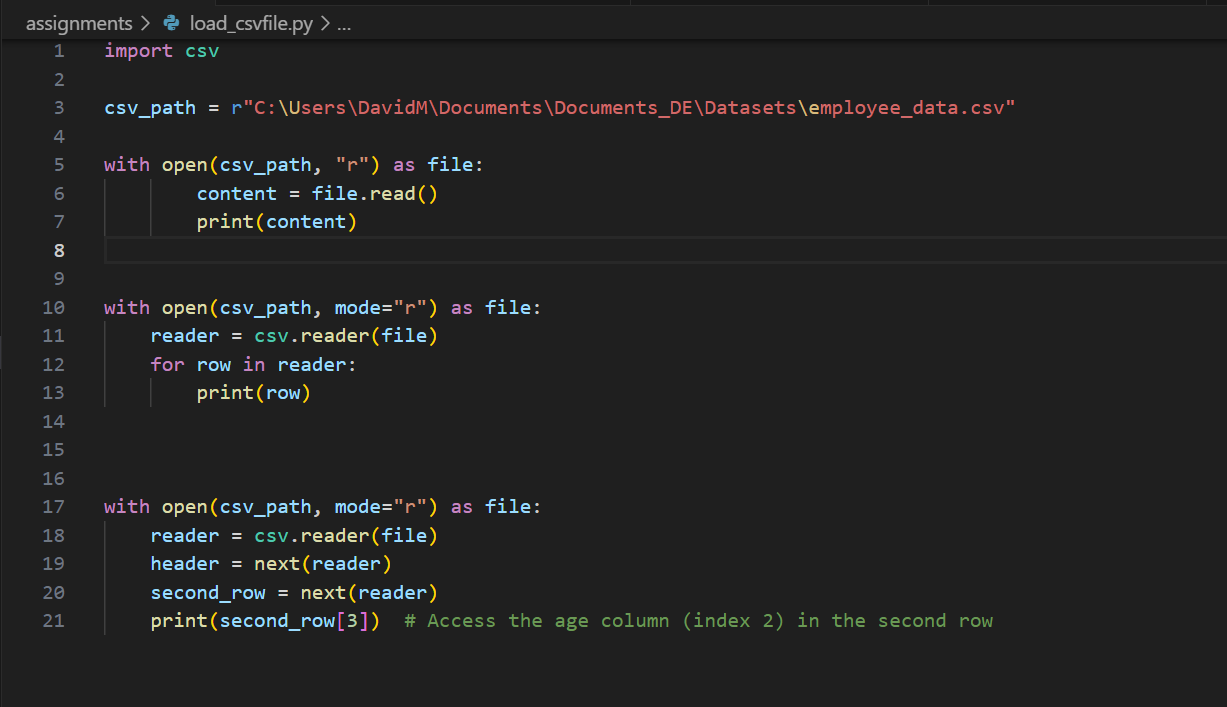
**Unity catalog:**

Unity Catalog provides centralized access control, auditing, lineage, and data discovery capabilities across Databricks workspaces.

Unity Catalog offers a single place to administer data access policies that apply across all workspaces.



csv files (load into python):



**GitHub Actions:**

GitHub Actions is a continuous integration and continuous delivery (CI/CD) platform that allows you to automate your build, test, and deployment pipeline.

You can create workflows that build and test every pull request to your repository or deploy merged pull requests to production.

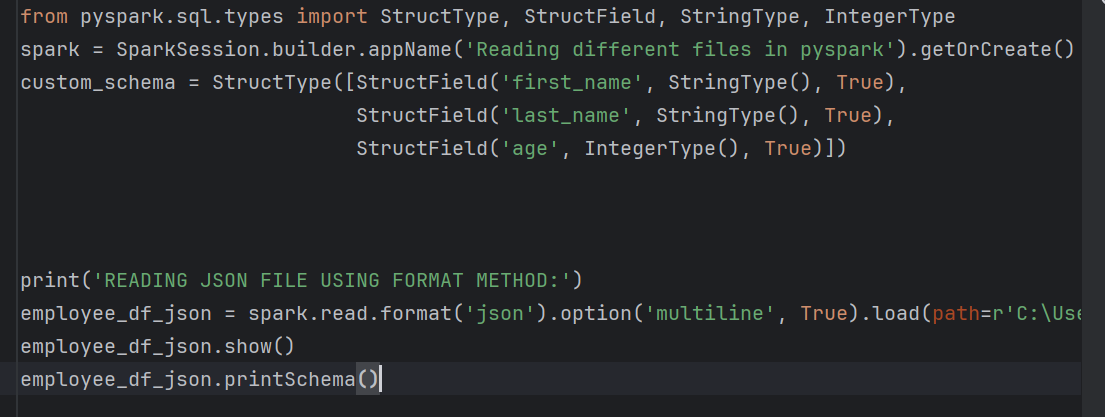
A screenshot of a computer program

Description automatically generated

A screenshot of a chat

Description automatically generated

**Custom schema- pyspark:**



**Azure Data Factory:**

Azure Data Factory is a cloud based ETL and Data Integration Services. Thet Providing Creation, Scheduling and Monitoring the Pipelines.

Azure Data Factory Components:

* IR
* Linked Services
* Datasets
* Pipeline
* Activity
* Triggers

**Most functional tests follow the Arrange-Act-Assert model:**

**Life cycle of test cases**

1. **Arrange**, or set up, the conditions for the test
2. **Act** by calling some function or method
3. **Assert** that some end condition is true

**What is the transactional system and analytical system**

**Transactional System (OLTP - Online Transaction Processing):**

Designed to manage and process day-to-day operational transactions in real-time.

* Data Structure: Typically uses normalized databases to reduce redundancy and ensure data integrity.
* Operations: Focuses on CRUD operations (Create, Read, Update, Delete).

**Analytical System (OLAP - Online Analytical Processing):**

Designed for complex querying and analysis of historical data to support decision-making.

* Data Structure: Typically uses denormalized databases (e.g., star or snowflake schema) to optimize for read-heavy operations.
* Operations: Focuses on complex queries, aggregations, and data analysis.

**surrogate key:**

* A surrogate key also called as a synthetic primary key.
* it is generated when a new record is inserted into a table automatically by a database that can be declared as the primary key of that table.
* In case we do not have a natural primary key in a table, then we need to artificially create one in order to uniquely identify a row in the table, this key is called the surrogate key or synthetic primary key of the table.

The usage of a surrogate key in a database is primarily to provide a unique, stable, and efficient way to identify records in a table.

**Storage Account:**

Azure Storage Account is a general-purpose storage solution that supports various types of data, including blobs, files, queues, and tables.

1. **Blob Storage**: Optimized for storing massive amounts of unstructured data, such as text or binary data.
2. **File Storage**: Provides managed file shares for cloud or on-premises deployments.
3. **Queue Storage**: Offers a messaging store for reliable messaging between application components.
4. **Table Storage**: A NoSQL store for schemaless storage of structured data.

**Azure Data Lake Storage Gen2 (ADLS Gen2):**  
 ADLS Gen2 is a highly scalable data storage and analytics service designed for big data analytics. It combines the features of Azure Blob Storage with a hierarchical file system.

**Referential integrity**

Referential integrity uses foreign keys to link tables and keep data consistent. A foreign key in one table points to a primary key in another, ensuring valid relationships. Without foreign keys, data can become inconsistent, with broken or orphaned records.

**Star schema:**

* A star schema is a relational database schema that organizes data in a star shape to make it easier to understand and analyse.
* It’s a multi-dimensional data model that consists of a central fact table surrounded by dimension tables.
* That fact table contains information about metrics or measures and the dimension tables contain information about descriptive attributes.

A diagram of a diagram

Description automatically generated

**Fact Table:**

* A fact table contains the quantitative data(facts) of a business process or event. These are typically numerical and measurable values, such as revenue, quantity or profit.
* Fact tables include foreign keys that reference primary keys in dimension tables, connecting them to the relevant dimensions.
* Fact tables have a final level of granularity, capturing detailed data at a transactional level

**Dimensional Tables:**

* Dimensional tables store descriptive information’s in the fact table. They Provide context to the quantitative data in the fact table.
* Dimension tables include attributes such as names, dates, locations, and other descriptive information. They have a primary key that is referenced by foreign keys in the fact table.

1.transactional system and analytical system

2.Surrogate Key?

3.difference between storage and adlsgen2

4.difference between parquet and delta

5.Referential Integrity Important

6.fact and dimensions table

7.What is a Decorators in Python

8.life cycle of test cases

**Databricks Asset Bundles:**

Databricks Asset Bundles are an infrastructure-as-code (IaC) approach to managing your Databricks projects.

Project including source control, code review, testing, and continuous integration and delivery (CI/CD).

**TDD (Test-Driven Development)** :

TDD (Test-Driven Development) is a software development approach where we write tests before writing the actual code.

The process follows a cycle known as **Red-Green-Refactor**:

* **Red:** Write a failing test for the functionality you want to implement.
* **Green**: Write the minimum amount of code to make the test pass.
* **Refactor:** Improve the code while ensuring the tests still pass

Repeat the above-mentioned steps again and again

**EDD (Example-Driven Development) :**

Before writing a code first define a clear real-world example is called EDD

* **Define Examples**: Write examples of input and expected output.
* **Implement Code**: Write code to make the examples work.
* **Refine**: Improve the code and add more examples as needed.

**merge schema, overwrite schema:**

**Merge Schema**

* Adds new columns without deleting existing ones.
* Old data remains, new columns get NULL for previous rows.
* Use .option("mergeSchema", "true") in append mode.

**Overwrite Schema**

* Completely replaces the existing schema.
* Old data and schema aredeleted.
* Use .option("overwriteSchema", "true") in overwrite mode.

**Distributed Systems**

* A distributed system is a network of multiple computers working together as a single unit.
* They improve scalability, fault tolerance, and performance by distributing tasks across nodes.
* Examples include Apache Spark and cloud computing platforms like AWS and Azure.

**EDD (Event Driven Development):**

Event-Driven Development (EDD) is a programming approach where actions (events) trigger responses in the system.

Instead of executing tasks sequentially, the system listens for events and reacts accordingly.

Event-driven systems are commonly used in real-time applications, microservices, IoT, and messaging systems

**Key Features of EDD:**

* **Asynchronous Execution** – Tasks don’t wait for each other; they happen independently.
* **Decoupled Components** – Each service works separately and only responds to relevant events.
* **Scalable & Resilient** – Easy to handle more events and recover from failures.

**merge schema, overwrite schema:**

**Merge Schema**

* Adds new columns without deleting existing ones.
* Old data remains, new columns get NULL for previous rows.
* Use .option("mergeSchema", "true") in append mode.

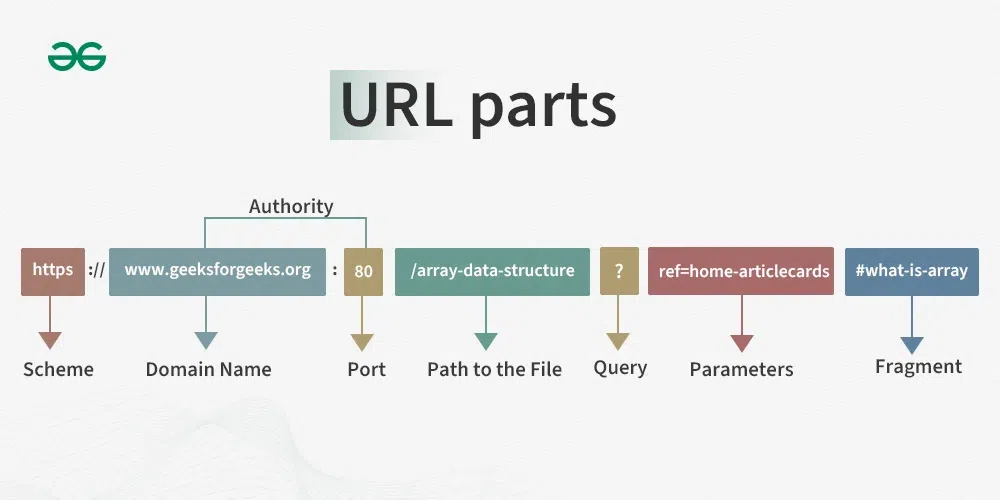
**Overwrite Schema**

* Completely replaces the existing schema.
* Old data and schema aredeleted.
* Use .option("overwriteSchema", "true") in overwrite mode.

**Distributed Systems**

* A distributed system is a network of multiple computers working together as a single unit.
* They improve scalability, fault tolerance, and performance by distributing tasks across nodes.
* Examples include Apache Spark and cloud computing platforms like AWS and Azure.

**Breaking Down the URL:**

****

[**https://www.instagram.com/reel/DEm0\_\_ZsWX\_/?igsh=MXFhMzJsd3Q1c3R5bA==**](https://www.instagram.com/reel/DEm0__ZsWX_/?igsh=MXFhMzJsd3Q1c3R5bA==)

**Protocol: https**

This HTTPS stands for HyperText Transfer Protocol Secure. It is the protocol used for secure communication over a computer network. HTTPS is the secure version of HTTP

**Domain:**[**www.instagram.com**](http://www.instagram.com)

This is the domain name of the website. It is the address where the resource is located.

**Path: /reel/DEm0\_\_ZsWX\_/**

This part of the URL specifies the specific resource or endpoint on the server.

* /reel/ indicates that the resource is a reel (a type of video post on Instagram)
* /DEm0\_\_ZsWX\_ is a unique identifier for that specific reel.

**Query Parameters: ?igsh=MXFhMzJsd3Q1c3R5bA==**

Query parameters are used to pass additional information to the server. They are separated from the path by a ?

They are usually in the form of key-value pairs. In this case, igsh is the key, and MXFhMzJsd3Q1c3R5bA== is the value.

**Different types of write modes in Databricks:**

**Append Mode**

* Adds new data to the existing dataset without modifying or deleting any existing records.
* **Use Case**: When we need to insert new records continuously, such as log data or incremental loads.

df.write.mode("append").parquet("path")

**2. Overwrite Mode**

* Replaces the existing data at the destination.
* **Use Case:** When you need to refresh the entire dataset.
* **Options:**

overwriteSchema=True: Updates schema as well when overwriting.

df.write.mode("overwrite").parquet("path")

**3. ErrorIfExists Mode (Default)**

* Fails if the target file or table already exists.
* **Use Case:** Prevents accidental overwrites.

df.write.mode("errorIfExists").parquet("path”)

**4. Ignore Mode**

* If the target exists, it does nothing and does not throw an error.
* **Use Case:** Ensures no accidental overwrites or errors in scheduled jobs.

df.write.mode("ignore").parquet("path/")

**DAG (Directed Acyclic Graph)**: A collection of tasks organized in a workflow with dependencies. It defines the sequence and execution logic of tasks.

**Task**: A single unit of work within a DAG. Each task represents an operator, such as running a Python script.

**Start Date**: The date from which the DAG starts executing. It determines when Airflow begins scheduling the DAG runs.

**End Date**: The date until which the DAG runs. If specified, the DAG stops running after this date.

**Catchup (True/False)**:

* **True**: Airflow will run all missing DAG runs from the start date until the current date.
* **False**: Airflow will only run the latest scheduled instance and ignore past dates.

A screen shot of a computer code

Description automatically generated

A screenshot of a computer program

Description automatically generated