Week VII

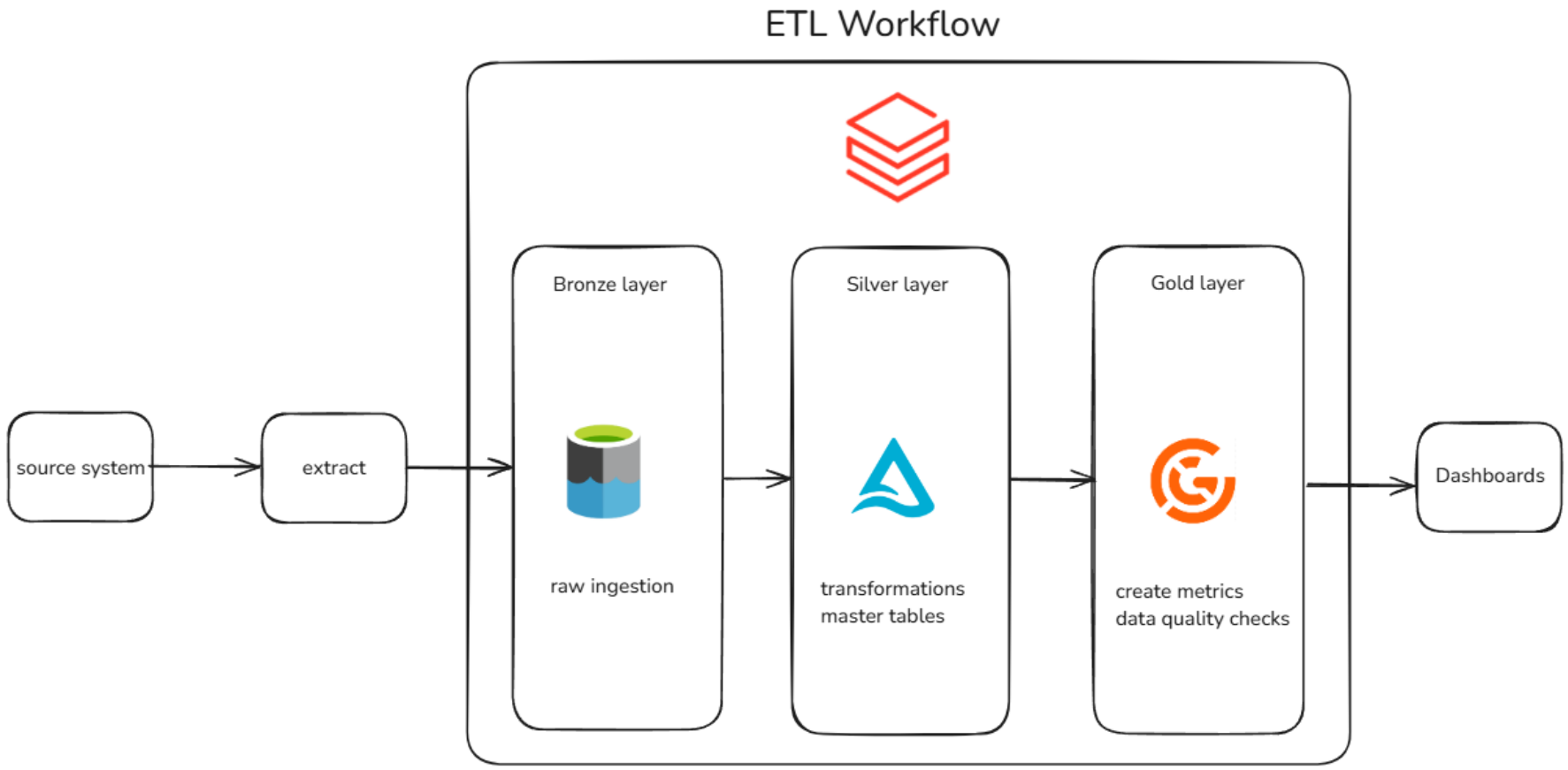
Capstone Project

Intermediate Data Engineer in Azure

Trainer: Balazs Balogh

CUBIX
INSTITUTE OF TECHNOLOGY

Project Overview



CUBIX
INSTITUTE OF TECHNOLOGY

First steps in Local

1. Create project folder

   poetry new cubix_data_engineer_capstone

   Don’t rename the “cubix_data_engineer_capstone” to “src”. You’ll have to name it back.
2. Adding dependencies (main, and group dev) to pyproject.toml
3. Initialise git with a test commit and push (in VSCode terminology: Sync)

First steps in Azure

You don’t need to create a separate Storage Account and Databricks, you can use the existing ones.
For the Storage Account, if you created a new container, don’t forget to add the Role Assigment of the App Registration (databricks_adls_connect in the video).

Connect Databricks to the Storage Account manually with **spark.conf.set()** configurations, to see if connection can be established.

Don’t forget to set the Termination time of your Cluster to a lower number than the default, like 20 minutes.

Create the **“source_system”** folder in your new container in the Storage Account. This will represent the client’s source system.

Our first function – Read file from Data Lake

```python
def read_file_from_datalake(container_name: str, file_path: str, format: str) -> DataFrame:
    """Reads a file from Azure Data Lake and returns it as a Spark DataFrame.

    :param container_name:  The name of the file system (container) in Azure Data Lake.
    :param file_path:       The path to the file in the data lake.
    :param format:          The format of the file ("csv", "json", "delta", "parquet").
    :return:                DataFrame with a loaded data.
    """

    if format not in ["csv", "parquet", "delta", "json"]:
        raise ValueError(f"Invalid format: {format}. Supported formats are: csv, json, parquet, delta.")

    full_path = f"abfss://{container_name}@{STORAGE_ACCOUNT_NAME}.dfs.core.windows.net/{file_path}"

    spark = SparkSession.getActiveSession()
    if not spark:
        raise RuntimeError("No active SparkSession found.")

    if format == "json":
        df = spark.read.json(file_path)
        return df
    else:
        df = (
            spark
            .read
            .format(format)
            .option("header", "true")
            .load(full_path, format=format)
        )

    return df
```

CUBIX
INSTITUTE OF TECHNOLOGY

Our first function – Read file from Data Lake

Test it by copying it to Databricks – not generally recommended as it can lead to errors if you are not aligning the notebook's with the local code.

Test it by creating a package with **"poetry build –f wheel"**.
Import the whl file into your Workspace, then install it directly in the notebook with **"!pip install <path_to_whl_file>"**.

Try to import it after the cluster is started, and the package is installed.

Write file from Data Lake

```python
def write_file_to_datalake(
        df: DataFrame,
        container_name: str,
        file_path: str,
        format: str,
        mode: str = "overwrite",
        partition_by: list[str] = None
) -> None:
    """Writes a DataFrame to Azure Data Lake as a parquet / csv / delta format.

    :param df:              DataFrame to be written.
    :param container_name:  The name of the file system (container) in Azure Data Lake.
    :param file_path:       The path to the file in the data lake.
    :param format:          The format of the file ("csv", "json", "delta", "parquet").
    :param mode:            Default "overwrite", write mode.
    :param partition_by:    List of column to partition by, default is None.
    """

    if format not in ["csv", "delta", "parquet"]:
        raise ValueError(f"Invalid format: {format}. Supported formats are: csv, parquet, delta.")

    full_path = f"abfss://{container_name}@{STORAGE_ACCOUNT_NAME}.dfs.core.windows.net/{file_path}"

    writer = df.write.mode(mode).format(format)

    if format == "csv":
        writer = writer.option("header", True)

    if partition_by:
        writer = writer.partitionBy(*partition_by)

    writer.save(full_path)
```

CUBIX
INSTITUTE OF TECHNOLOGY