**ETL PIPELINE AND DATA WAREHOUSE**

**CONFIGURATION**

1. We need to create a **Process Control Table** and a **Job Control Table** to log and monitor the ETL process.
   1. **Process Control Table**: This table will track the overall execution time and status of the ETL process. In the event of a failure due to technical issues, the process control table will allow the process to be restarted from the next scheduled load.
   2. **Job Control Table**: This table will monitor individual job execution times and their statuses. In case of a process failure and restart, the job control table will skip jobs that have already succeeded and restart from the job where the failure occurred.
2. Create Process Control table which contain the list of columns below

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Column Description** |
| UID | NVARCHAR | This will be a combination in Process Name & Process start time |
| ProcessName | VARCHAR | Process Name |
| ProcessRunDate | DATE | This will be the system date |
| ProcessStartTime | DATETIME | This field will update the system datetime when process will started |
| ProcessEndTime | DATETIME | This field will update the system datetime when process will ended |
| ProcessStatus | VARCHAR | This field contains the process status which Success or Failure |
| LoadType | CHAR(1) | This field contains the load type whther it is Historical or Incremental |
| ErrorDescription | NVARCHAR(4000) | If the process failed it will automatically load the error description into this field |

1. Create Job Control table which contain the list of columns below

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Column Description** |
| UID | NVARCHAR | This will be a combination in Process Id & Process start time |
| JobName | VARCHAR | Job Name |
| ProcessUID | DATE | This will the current process UID |
| JobRunDate | DATETIME | This will be the system date |
| JobStartTime | DATETIME | This field will update the system datetime when process will started |
| JobEndTime | VARCHAR | This field will update the system datetime when process will ended |
| JobStatus | CHAR(1) | This field contains the process status which Success or Failure |
| JobLoadType | VARCHAR | This field contains the load type whether it is Historical or Incremental |
| ErrorDescription | NVARCHAR(4000) | If the process failed it will automatically load the error description into this field |

1. Table Description

A white rectangular object with a black border

Description automatically generated

Table: dim\_Device

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Column Description** |
| DeviceKey | INT | Auto generated Key |
| DeviceId | UNIQUEIDENTIFIER | Device Id is a Unique for every device. This field data should not be NULL and It should be Unique |
| DeviceName | NVARCHAR(250) | Every Device will have a unique name. |
| FileGeneratedDatetime | DATETIME | This field contains the file generated date & time |
| IsActiveYN | TINYINT | This field contains the device status (Active or Inactive) |
| CreatedInstant | DATETIME | This field contains the record created date & time in the table |
| LastUpdateInstant | DATETIME | This field contains the record updated date & time in the table |



|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Column Description** |
| DeviceKey | UNIQUEIDENTIFIER | This field contains surrogate key of dim\_Divice table |
| CurrentValue | NUMERIC(18,4) | This field contains the device reading value |
| UNIT | VARCHAR(50) | This field contains the UNIT measure |
| ReadingGeneratedDatetime | DATETIME | This field contains the Reading Generated date & time |
| Version | NUMERIC(18,0) | This field contains the Version information |
| CreatedInstant | DATETIME | This field contains the record created date & time in the table |
| LastUpdateInstant | DATETIME | This field contains the record updated date & time in the table |

1. Crate a control flow table which contain the list of contents
   1. This table will maintain File path information.

**PROJECT SETUP**

1. **Create a Master Pipeline with a Unique Name in Azure Data Factory**
   1. **Naming Convention**: Use the syntax for naming pipelines:  
      pl\_<Master Pipeline ID>\_<Child Pipeline ID>\_<Pipeline Name>
   2. **Example**: A master pipeline could be named pl\_10001\_000\_<Pipeline Name>
2. **Ensure that all related activities, such as child pipelines, datasets, and data flows, reference the Master Pipeline ID**
   1. **Child Pipeline Example**: Use the format pl\_10001\_001\_<Pipeline Name> for child pipelines.
   2. **Dataset Example**: Use the format ds\_10001\_001\_<Dataset Name> for datasets.
   3. **Data Flow Example**: Use the format df\_10001\_001\_<Data Flow Name> for data flows.
3. **Pipeline Behaviour**  
   Once the Master Pipeline starts, it will determine whether the load is **Historical** or **Incremental**. The default setting will be for **Incremental load**.
4. **Data Ingestion**  
   Retrieve raw data files (in JSON format) from an FTP or shared location and use the Copy Activity in Azure Data Factory to place them in Azure Blob Storage.
5. **Backup of Raw Files**  
   After copying the raw data files to Azure Blob Storage, take a backup of the files and store them in a designated container in Azure Blob Storage using the Copy Activity.
6. **Convert and Optimize**  
   Convert the raw JSON files to **Parquet** format for faster data processing, using the Copy Activity.
7. **Load Data into Azure Synapse**  
   Finally, load the Parquet files into staging tables within Azure Synapse. Below is the script to create the required stage schema and tables:

A white rectangular object with a black border

Description automatically generated

Creating stage schema:

CREATE SCHEMA stg;

Creating Device table in stage environment:

CREATE TABLE stg.Device

(

DeviceId UNIQUEIDENTIFIER NOT NULL,

Name NVARCHAR(50) NULL,

CreatedAt DATETIME NOT NULL

)



Creating Device Reading table in stage environment:

CREATE TABLE stg.DeviceReading

(

DeviceId UNIQUEIDENTIFIER NOT NULL,

CurrentValue NUMERIC(18,4) NULL,

UNIT NVARCHAR(50) NULL,

ReadingTimestamp DATETIME NULL,

Version NUMERIC(18,0) NULL

)

1. Using the data from the staging tables, create dimension and fact tables by utilizing the **Data Flow** activity in Azure Data Factory.

CREATE SCHEMA dw;

Creating Device dimension table in DWH environment:

CREATE TABLE dw.dim\_Device

(

DeviceKey INT IDENTITY(1,1),

DeviceId UNIQUEIDENTIFIER NOT NULL,

DeviceName NVARCHAR(250) NULL,

DeviceCreatedDatetime DATETIME NULL,

IsActiveYN TINYINT NOT NULL,

CreatedInstant DATETIME NOT NULL,

LastUpdateInstant DATETIME NOT NULL

)

Creating Device Reading fact table in DWH environment:

CREATE TABLE dw.fact\_DeviceReading

(

DeviceKey INT NOT NULL,

CurrentValue NUMERIC(18,4) NULL,

UNIT NVARCHAR(50) NULL,

ReadingGeneratedDatetime DATETIME NULL,

Version NUMERIC(18,0) NULL,

CreatedInstant DATETIME NOT NULL,

LastUpdateInstant DATETIME NOT NULL

)

Creating Indexes for DWH.dim\_Device and DWH.dim\_DeviceReading:

CREATE INDEX IX\_DeviceKey ON dw.fact\_DeviceReading (DeviceKey);

CREATE INDEX IX\_d\_Device\_Key ON dw.dim\_Device(DeviceKey);

1. After all data has been processed into the dimension and fact tables, the process will conclude by updating the process status in the **Process Control Table**

**SCALABILITY**

1. Indexes have been introduced to enhance query performance.
2. **Future enhancements:**

* Partitioning the *dw.fact\_DeviceReading* table by timestamp can be implemented to improve performance when handling large datasets.