**Use Case of Watermark Table in Azure ETL Pipelines: A Detailed Technical Perspective**

**1. Context Overview**

A watermark table is a critical component in ETL (Extract, Transform, Load) pipelines designed for incremental data loads, especially when moving data from systems like SQL Server to Azure services such as Azure SQL Database, Azure Data Lake, or Azure Synapse Analytics. In large-scale data environments, full data extraction is often impractical due to the volume of records and the need for real-time or near-real-time data processing. A watermark table efficiently tracks the progress of data extraction, ensuring that only new or updated records are fetched during each ETL run.

**Sample data of Watermark Table:**



**Hints: Solution Technical Flow:**

In an Azure Data Factory (ADF) pipeline, you typically create a **watermark table** to track the "last processed value," which could be a timestamp or an ID. This value indicates the point at which the previous ETL process stopped, so only records inserted or updated after this point are processed in subsequent ETL executions.

**3. Watermark Table Structure Example**

The watermark table in Azure could be structured like this:

sql

Copy code

CREATE TABLE WatermarkTable (

Watermark\_ID INT PRIMARY KEY IDENTITY(1,1),

SourceSystem NVARCHAR(128), -- The source system (e.g., SQL Server)

Table\_Name NVARCHAR(128), -- The specific table being tracked

LastProcessedValue NVARCHAR(128), -- Holds the last processed value (timestamp or ID)

ProcessedTime DATETIME -- When the last processing occurred

);

Each record in the **WatermarkTable** represents the last point of data processed for a particular source system and table, ensuring incremental data extraction.

**4. Technical Use Case: SQL Server to Azure SQL Database**

In this scenario, you are extracting data from a SQL Server source and loading it into an Azure SQL Database using Azure Data Factory (ADF) or Synapse Pipelines.

**4.1. Initial Full Load**

* **Step 1**: Extract all records from the SQL Server source for the first time, with no filters on date or ID.
* **Step 2**: Load these records into the Azure SQL Database.
* **Step 3**: After a successful load, store the highest timestamp or ID value from the source table into the LastProcessedValue field in the watermark table.

**SQL Example to Insert Initial Watermark Entry:**

INSERT INTO WatermarkTable (SourceSystem, Table\_Name, LastProcessedValue, ProcessedTime)

VALUES ('SQLServer', 'Sales\_Data', '2024-09-15 05:00:00', GETDATE());

**4.2. Incremental Data Load**

In subsequent ETL runs, the pipeline will use the **watermark value** to filter records from the source.

* **Step 1**: Retrieve the LastProcessedValue for the table being processed.

SELECT LastProcessedValue

FROM WatermarkTable

WHERE Table\_Name = 'Sales\_Data';

* **Step 2**: Query the source system (SQL Server) for records inserted or modified after this LastProcessedValue.

SELECT \*

FROM Sales\_Data

WHERE SaleDate > '2024-09-15 05:00:00';

* **Step 3**: Load the newly fetched data into the Azure SQL Database.
* **Step 4**: After successful loading, update the watermark table with the new LastProcessedValue.

UPDATE WatermarkTable

SET LastProcessedValue = '2024-09-15 06:00:00',

ProcessedTime = GETDATE()

WHERE Table\_Name = 'Sales\_Data';

**5. Advantages of Using a Watermark Table**

1. **Incremental Loading**:
   * Avoids reprocessing the entire dataset. Only new or updated records since the last processed point are extracted.
2. **Performance Optimization**:
   * By processing only deltas (new data), the pipeline reduces the load on the source system, lowers network usage, and speeds up ETL execution.
3. **Fault Tolerance**:
   * If the ETL pipeline fails, you can restart the process from the last successfully processed record, reducing data redundancy or data loss risks.
4. **Scalability**:
   * Handles large datasets efficiently by tracking each table's progress, allowing you to design multi-table ETL pipelines.

**6. Integration with Azure Data Factory (ADF)**

* **Activity Integration**: Azure Data Factory can integrate the watermark table within a control flow using Lookup and Update activities. For instance:
  + **Lookup Activity**: Use it to fetch the LastProcessedValue from the watermark table before querying the source system.
  + **Stored Procedure/Script Activity**: Use SQL statements or stored procedures to update the watermark table after each successful load.

**7. Advanced Use Case: Handling Multiple Sources and Tables**

In a more complex scenario, your watermark table might track multiple tables and even different source systems, such as SQL Server, Oracle, or Salesforce. Each table and source would have a distinct LastProcessedValue, and the ETL pipeline would process data accordingly.

**8. Monitoring and Automation**

The watermark table makes it easier to monitor the progress of incremental loads and automate the scheduling of data extraction, allowing fully automated and hands-off data integration workflows.

This approach is particularly useful for scenarios where you have real-time or near-real-time data processing requirements, and where you need to ensure minimal latency between the source system and the destination (Azure SQL, Data Lake, Synapse, etc.).