CDC with Databricks

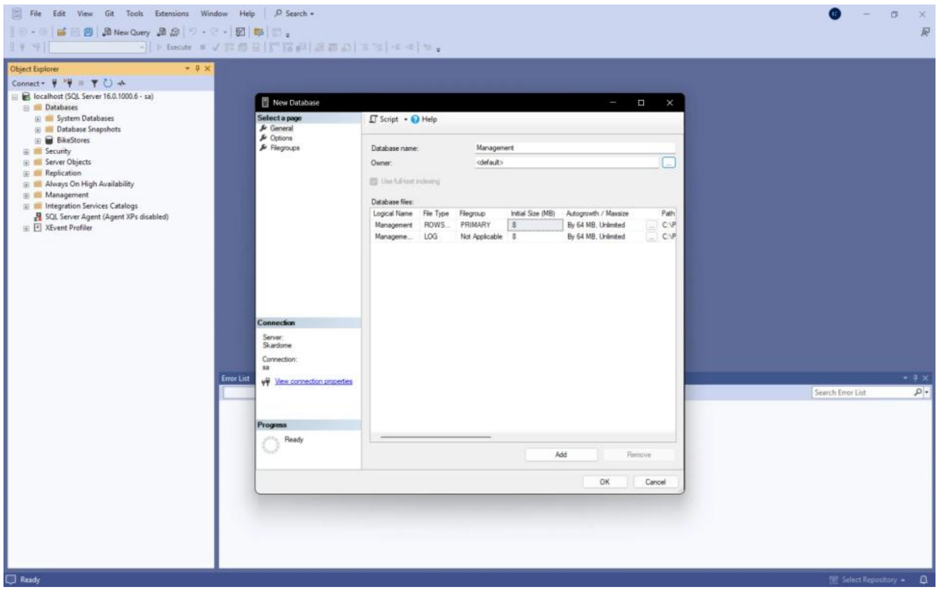
Project Documentation

Concepts to cover: Focusing on implementing Change Data Capture (CDC) using Databricks, a unified data analytics platform. The primary objective is to efficiently identify and process data changes (inserts, updates, deletes) in source systems and reflect them in the target data lake or data warehouse in near real-time. Leveraging Apache Spark, Delta Lake, and streaming capabilities within Databricks, the project aims to build a scalable and reliable CDC pipeline that supports incremental data ingestion, ensures data consistency, and minimizes processing overhead.

Step 1

Initially, the project involves setting up a reliable database environment by installing and configuring MS SQL Server in a local environment, including necessary network protocols and user permissions. Automated SQL scripts are developed to create databases and tables, specifically designed to optimize CDC operations. Additionally, scripts are prepared to populate tables with initial datasets, with scheduled jobs ensuring continuous data insertion and updates to simulate a dynamic source environment.

Setting up MS SQL Server



Database name: Management
Owner: <default>

Server: Skardome
Connection: sa

Database files:

Logical Name	File Type	Filegroup	Initial Size (MB)	Autogrowth / Maxsize	Path
Management	ROWS...	PRIMARY	8	By 64 MB, Unlimited	C:\P
Manageme...	LOG	Not Applicable	8	By 64 MB, Unlimited	C:\P

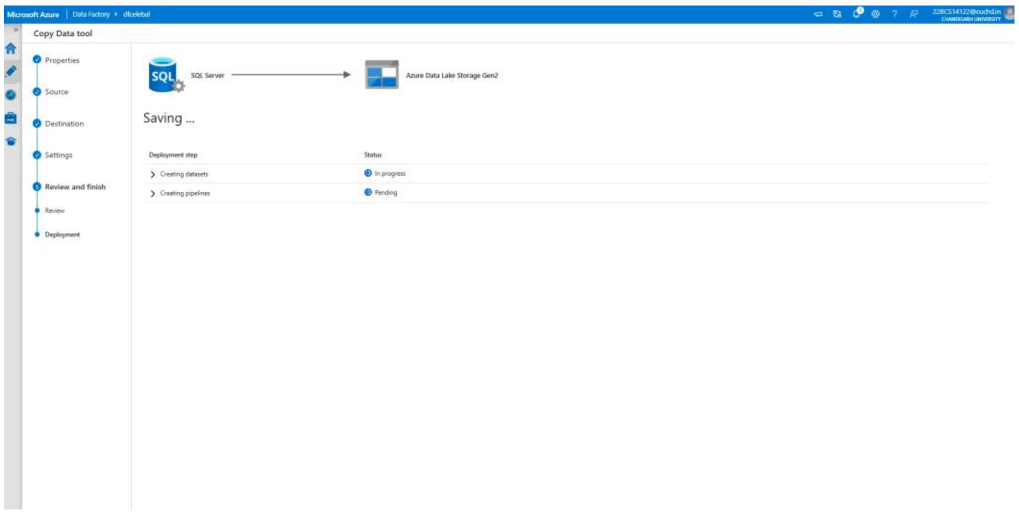
Adding Queries for Table Creation:
https://github.com/Skaelum/CDC_Databricks_Celebal/blob/main/SQL%20Queries/Table%20Creation%20.sql

Automating Table Population:
https://github.com/Skaelum/CDC_Databricks_Celebal/blob/main/SQL%20Queries/Automate%20Table%20Population.sql

Step 2

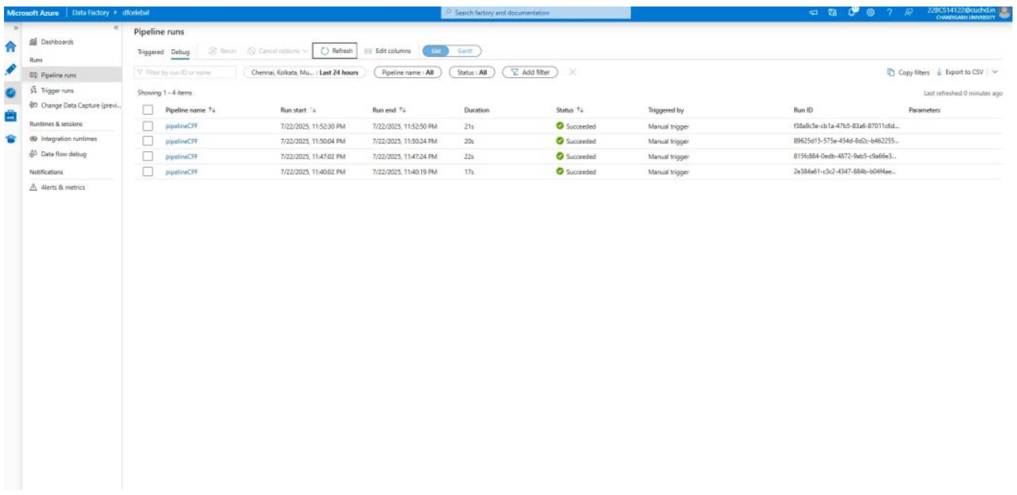
Azure Data Factory (ADF) plays a central role in data migration and ingestion, starting with its provisioning and configuration. A secure Self-Hosted Integration Runtime (SHIR) is set up to facilitate secure data transfers from the local SQL server environment to the Azure cloud. Rigorous connectivity tests are conducted to validate data movement and integration between the local infrastructure and Azure Data Factory.

Setting up a self-hosted runtime integrated (SHIR) connection for migrating the database from local to cloud server.



Saving ...

Deployment step	Status
Creating datasets	In progress
Creating pipelines	Pending

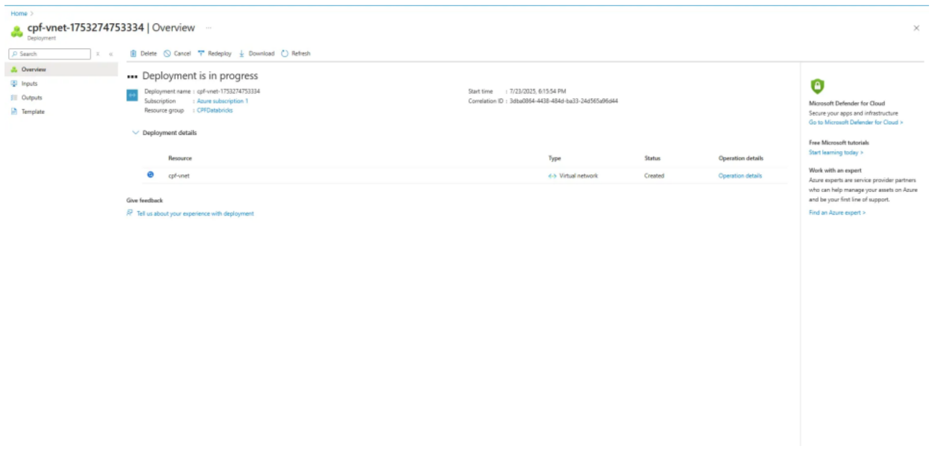


Showing 1 - 4 items

Last refreshed 0 minutes ago

Pipeline name	Run start	Run end	Duration	Status	Triggered by	Run ID	Parameters
pipelineCPF	7/22/2025, 11:52:30 PM	7/22/2025, 11:52:50 PM	21s	Succeeded	Manual trigger	f38a9c5e-cb1a-47b5-83a6-87011c6d...	
pipelineCPF	7/22/2025, 11:50:04 PM	7/22/2025, 11:50:24 PM	20s	Succeeded	Manual trigger	89625d15-575a-454d-8d2c-b462255...	
pipelineCPF	7/22/2025, 11:47:02 PM	7/22/2025, 11:47:24 PM	22s	Succeeded	Manual trigger	815fc884-0edb-4872-9ab5-c9a66e3...	
pipelineCPF	7/22/2025, 11:40:02 PM	7/22/2025, 11:40:19 PM	17s	Succeeded	Manual trigger	2e384a61-c3c2-4347-884b-b04f4ae...	

Configured a Subnet for Databricks



cpf-vnet-1753274753334 | Overview
Deployment

Deployment is in progress

Deployment name : cpf-vnet-1753274753334
Subscription : Azure subscription 1
Resource group : CPFDatabricks
Start time : 7/23/2025, 6:15:54 PM
Correlation ID : 3dba0864-4438-484d-ba33-24d565a96d44

Deployment details

Resource	Type	Status	Operation details
cpf-vnet	Virtual network	Created	Operation details

Secure your apps and infrastructure

Azure experts are service provider partners who can help manage your assets on Azure and be your first line of support.

Step 3

The ingestion process involves creating pipelines that transfer data from the local SQL server to Azure Storage Account (ADLS Gen2). Databricks is then utilized to configure Change Data Capture, leveraging Delta Lake tables to effectively track and identify row-level changes such as INSERT, UPDATE, and DELETE operations. Using Spark Structured Streaming, the incremental ingestion of data changes is streamlined, ensuring efficiency and near real-time processing. Additionally, Azure Logic Apps are implemented to create notification pipelines, which send operational alerts via email after each successful execution.

1. Creating a pipeline to ingest data into the storage account from the localhost.
2. Setting up change data capture in databricks (CDC) to detect row-level changes that reflect INSERT, UPDATE, DELETE queries in the source table.
3. Creating an email notification pipeline to notify operations after each run.

Step 4

An incremental ETL pipeline is established to run at regular hourly intervals, capturing incremental data changes automatically. Robust error handling, logging, and retry mechanisms are implemented to enhance pipeline resilience. Furthermore, Databricks workspace is seamlessly integrated within Azure Resource Groups, ensuring smooth authentication and authorization processes. Apache Spark's powerful computing capabilities are harnessed through Azure Logic Apps to manage complex CDC processes, while optimization efforts are focused on resource allocation for performance efficiency and cost management.

1. Create an ETL pipeline to run every hour to capture the changes
2. Integrated the databricks workspace with azure resource group
3. Using Azure Logic App to implement apache spark architecture

Step 5

The final phase involves configuring automated email alerts using Azure Logic Apps, triggered after successful pipeline executions. These alerts contain essential information including records inserted, updated, deleted, and various runtime metrics to facilitate operational monitoring and oversight.

1. Trigger email sending function to trigger after running every successful pipeline.