3 - Ingest from On-Premises SQL DB

Background story

Congratulations, Caladan has now centralized their cloud data! Remember that the on-premise data also needs to be incorporated. Leadership has become increasingly concerned that they are unable to effectively determine the best policies if all data is not incorporated. With the establishment of an enterprise data lake, the time to leverage the additional data is close at hand! First, this additional data needs to be extracted.

Caladan wants a repeatable data flow to pull data from both their cloud source and on-premise. They would like the team to leverage the services they provisioned for the previous import effort, which brought both the Azure SQL Databases and Azure Cosmos DB data to the enterprise data lake. Additionally, they would like the team to ensure that any solutions created to extract data can be further leveraged when any any other health related data is either made available on-prem or in the cloud.

Once there's a baseline for the services used to import data, Caladan requires that the team establish version control to ensure that the work performed by the team can be persisted, tracked, and potentially audited.

Technical details

The team will now begin working with the on-premise data.

Although the data will land in the same data lake, it should be grouped by the source system of record:

- On-Prem SQL
- Azure SQL DataBase
- Azure Cosmos DB

The team's coach can provide VM login information, or the team can reset the VM's administrator login information. By default, the lab setup process uses `vmadmin` as the VM administrator username.

The team's coach can provide VM login information, or the team can reset the VM's administrator login information. By default, the lab setup process uses `vmadmin` as the VM administrator username. By default, the lab setup process uses `sqladmin` as the SQL administrator login username.

Success criteria

- All data from the `dbo` schema of the COVID19 VM Covid19 database has been extracted and stored in the enterprise data lake.
- All extracted data is grouped by the source system of record and a flag added.
- The same DL has been used throughout.
- The team has persisted all extraction solution(s) in version control.

Tips

- You can use the SQL Server Management Studio on the VM to connect to and manage their databases.

- Regarding common Azure infrastructure:
  - It **is** acceptable to author new pipelines, create new directories, or install new runtimes on existing resources.
  - It **is not** acceptable to create new storage accounts or data factories.
- Teams choosing to use the self-hosted integration runtime should take note of the supported file formats and any additional prerequisites or conditions that may apply to the selected format. (e.g., installation of a JRE for Parquet or ORC)
  - Teams may avoid some of these additional prerequisites by copying the data "as-is" and deferring additional conversions until they perform cloud processing.
  - For security and performance purposes in production scenarios, the self-hosted integration runtime is ideally installed on a separate machine from the one that hosts the data itself (e.g., on a jumpbox). For the scope of this OpenHack, it is acceptable to install the runtime on the machine which hosts the data.
- It is only required that the team persists **their authored solutions** to version control.

Resources

Ramp Up

- Self-Hosted Integration Runtime
- About Version Control

Choose Your Tools

- GitHub Guides - Hello World

Dive In

- Copy data from an on-premises SQL Server database to Azure Blob Storage
- Copy data to or from a file system by using Azure Data Factory
- Visual authoring in Azure Data Factory