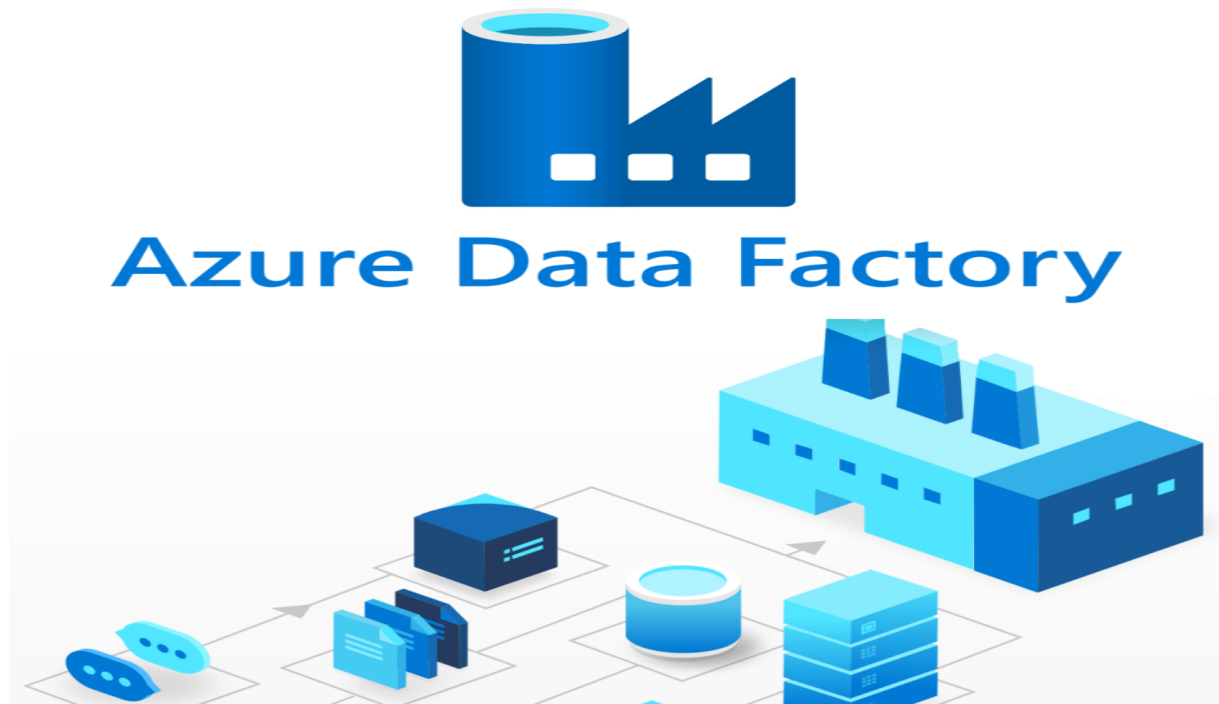


Azure data factory project.



In this document you will find a project utilizing Azure Data Factory to extract, transform, and load data from a source and file within blob storage into a SQL database. This project involved setting up data pipelines to schedule data movement and transformation activities in data flow and databricks, as well as integrating with other Azure services such as Azure Data Lake Storage and Azure Data Lake. Through the use of Azure Data Factory, I was able to efficiently process and move data.

Contents

Azure data factory project.	1
1. Introduction	4
1.1 Project overview:.....	4
2. Solution Architecture	5
3. Data ingestion.....	5
3.1 Data ingestion with Azure blob storage	6
Step 1 - Copy activity overview.....	6
Step 2 - Linked service.....	7
Step 3 - Blob data set (source).....	8
Step 4 - Data Lake data set (sink).....	9
Step 5 - Creating an ingestion pipeline	10
Step 6 - Trigger (ingestion) pipeline	11
3.2 Data ingestion from HTTP	12
Step 0 - Overview copy activity from HTTP	12
Step 1 - Created linked services	12
Step 1 - Create data set (http - source)	13
Step 2 - Create data set (DL sink)	14
Step 3 - Create pipeline for data ingestion.....	15
Step 4 - Create trigger for pipeline.....	16
Step 6 - Ecdc file list data set	16
4. Data transformation.....	17
4.1 Data flow.....	17
Step 0 – Overview transformation in data flow.....	17
(1) Step 1 - Create transformations with data flow	18
(1) Step 2 - Create data set for processed data	18
(1) Step 3 – Create pipeline	19
(2) Step 4 - Create transformations with data flow	20
Step 5 - Create data set for processed data (daily).....	20
Step 6 - Create data set for processed data (weekly)	21
Step 7 - Create pipeline	21
4.2 Databricks	23
Step 0 - Databricks overview	23
Step 1 – Mount cluster 1 / 2	23
Step 1 – Mount cluster 2 / 2	24
Step 2 – transform data 1 / 2.....	25

Step 2 – transform data 2 / 2	26
Step 4 – Create pipeline	27
5. Load into database	27
Step 1 – Create SQL script	28
Step 2 – Create linked service	29
Step 3 - Create sink dataset	30
Step 4 – Create pipeline	31
Step 5 – Create pipeline hospital admission data	32
Step 6 - Create pipeline testing data	33
6. Data orchestration – Making pipelines production ready	34
Step 1 – Build pipeline	34
Step 2 – Create trigger	35
7. Data reporting	36
Step 0 – Reporting overview in Power BI	36
Step 1 – Create report 1 / 2	36
Step 1 – Create report 2 / 2	37
8. CI / CD	38
Step 0 – CI / CD overview	38
Step 0 – Git configuration within DevOps option 1	39
Step 0 – Git configuration within DevOps option 2	39
Step 1 – Create git repo	40
Step 2 – Create Tools for azure (DEV)	41
Step 2 – Create Tools for azure (TEST)	41
Step 2 – Create Tools for azure (PROD)	42
Step 3 - Create release pipeline option 1	42
Step 4 – Create build pipeline option 2 – 1 / 2	43
Step 5 – Create release pipeline option 2	44

1. Introduction

In this project we will be building a data platform for reporting and prediction of the covid-19 outbreak.

- Data sources: European Center for Disease prevention and Control (Eurostat)
- Data flows: azure data factory
- Data transformation: HDInsight and databricks
- Ingestion of transformed data: Data Lake
- Necessary data for reporting the data trends: SQL Data warehouse
- We will orchestrate all of these pipelines using Azure data factory.

1.1 Project overview:

Our build data lake:

Data Lake to be built with the following data, to aid Data Scientists to predict the spread of the virus/mortality

- Confirmed cases
- Mortality
- Hospitalization/ ICU Cases
- Testing Numbers
- Country's population by age group

Our build data warehouse:

Data Warehouse to be built with the following data

- Confirmed cases
- Mortality
- Hospitalization/ ICU Cases
- Testing Numbers

The data sources:

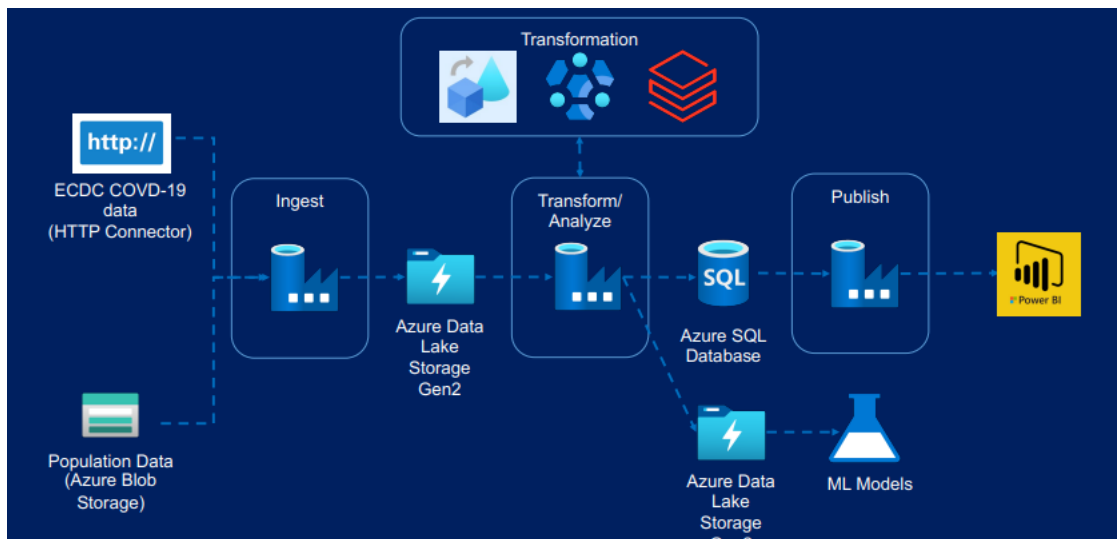
ECDC website:

- Confirmed cases
- Mortality
- Hospitalization / ICU Cases
- Testing Numbers

Eurostat Website:

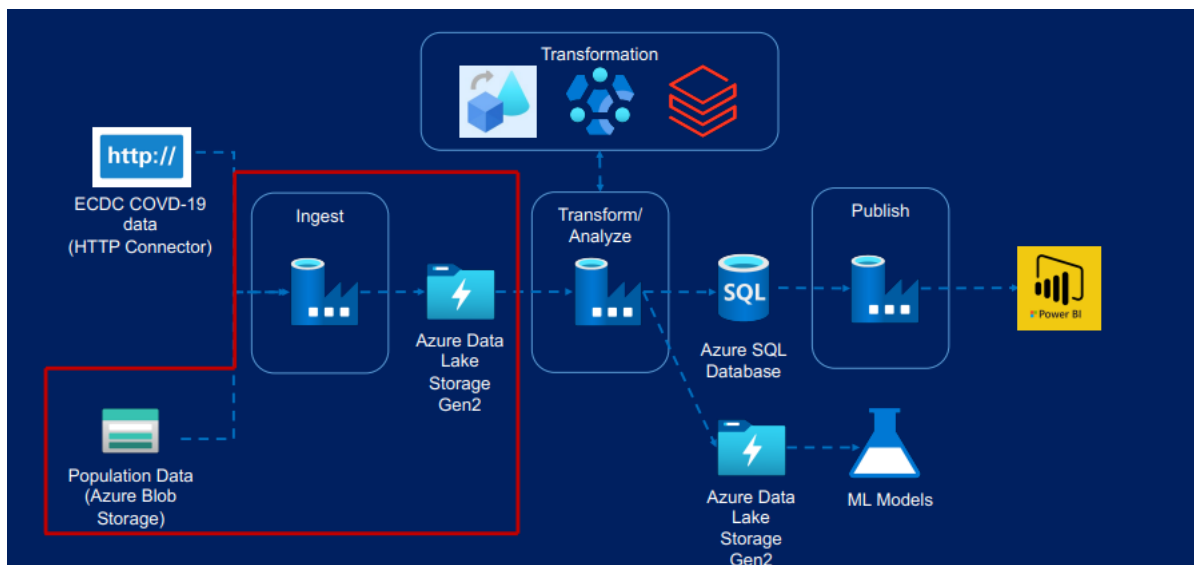
- Population by age

2. Solution Architecture



3. Data ingestion

Overview

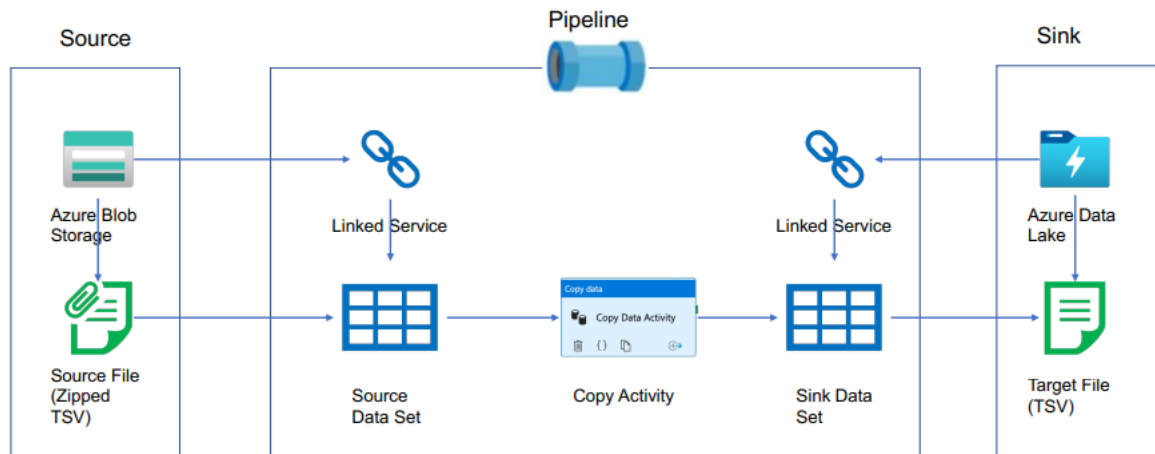


Worked with:



- Copy Activity
- Linked Services
- Datasets
- Validation Activity
- If Condition Activity
- Get Metadata Activity
- Web Activity
- Delete Activity
- Trigger

3.1 Data ingestion with Azure blob storage

Step 1 - Copy activity overview



Step 2 - Linked service

Edit linked service
 Azure Blob Storage [Learn more](#) 

Name *
ls_ablob_covidreportingsa

Description

Connect via integration runtime * ⓘ
AutoResolveIntegrationRuntime

Authentication type
Account key

Connection string **Azure Key Vault**

Account selection method ⓘ
☐ From Azure subscription ☒ Enter manually

Storage account name *
covidreportingsa0202

Storage account key **Azure Key Vault**

Storage account key *
.....

Partitioned DNS enabled ⓘ ☐

Endpoint suffix
core.windows.net


Additional connection properties
[+ New](#)

Test connection ⓘ
☒ To linked service ☐ To file path

Annotations
[+ New](#)
[> Parameters](#)
[> Advanced](#) ⓘ

Apply

Cancel

 Test connection

Step 3 - Blob data set (source)

Factory Resources

Filter resources by name

- Pipelines (9)
 - Execute (1)
 - Ingest (2)
 - Process (3)
 - sqlize (3)
- Datasets (16)
 - lookup (3)
 - processed (4)
 - raw (6)
 - ds_ecdc_raw_csv_dl
 - ds_ecdc_raw_csv_http
 - ds_population_raw_gz**
 - ds_population_raw_tsv
 - ds_raw_cases_and_deaths
 - ds_raw_hospital_admissions
 - sql (3)
- Data flows (2)
- Power Query (0)

ds_population_raw_gz

DelimitedText
ds_population_raw_gz

Connection Schema Parameters

Linked service * Is_azureblob_covidreportingsa [Test connection](#) [Edit](#) [New](#) [Learn more](#)

File path * population / Directory / population_by_age.tsv [Browse](#) [Preview data](#)

Compression type gzip (-gz)

Compression level Optimal

Column delimiter Tab (\t) [Edit](#)

Row delimiter Default (\r\n, or \n\r) [Edit](#)

Encoding Default(UTF-8)

Escape character Backslash (\) [Edit](#)

Quote character Double quote (") [Edit](#)

First row as header ☒

Null value

Step 4 - Data Lake data set (sink)

The screenshot displays the Microsoft Fabric Data Factory interface. On the left, the 'Factory Resources' pane shows a tree view with 'Pipelines' (9 items) and 'Datasets' (16 items). Under 'Datasets', the 'raw' folder is expanded, showing several datasets, with 'ds_population_raw_tsv' selected. The main workspace shows a 'DelimitedText' sink icon and the name 'ds_population_raw_tsv'. Below this, the 'Connection' tab is active, showing the configuration for the sink. The 'Linked service' is 'ls_adls_covidreportingdl'. The 'File path' is 'raw / population / population_by_age.tsv'. The 'Compression type' is 'None'. The 'Column delimiter' is 'Tab (\t)'. The 'Row delimiter' is 'Default (\r\n, or \n)'. The 'Encoding' is 'Default(UTF-8)'. The 'Escape character' is 'Backslash (\)'. The 'Quote character' is 'Double quote (")'. The 'First row as header' checkbox is checked. The 'Null value' field is empty.

Factory Resources

- Pipelines (9)
 - Execute (1)
 - Ingest (2)
 - Process (3)
 - sqlize (3)
- Datasets (16)
 - lookup (3)
 - processed (4)
 - raw (6)
 - ds_ecdc_raw_csv_dl
 - ds_ecdc_raw_csv_http
 - ds_population_raw_gz
 - ds_population_raw_tsv**
 - ds_raw_cases_and_deaths
 - ds_raw_hospital_admissions
 - sql (3)
- Data flows (2)
- Power Query (0)

ds_population_raw_tsv

DelimitedText
ds_population_raw_tsv

Connection Schema Parameters

Linked service * ls_adls_covidreportingdl [Test connection](#) [Edit](#) [+ New](#) [Learn more](#)

File path * raw / population / population_by_age.tsv [Browse](#) [Preview data](#)

Compression type None

Column delimiter Tab (\t) [Edit](#)

Row delimiter Default (\r\n, or \n) [Edit](#)

Encoding Default(UTF-8)

Escape character Backslash (\) [Edit](#)

Quote character Double quote (") [Edit](#)

First row as header ☒

Null value

Step 5 - Creating an ingestion pipeline

The screenshot displays the Azure Data Factory pipeline editor for a pipeline named `pl_ingest_populati...`. The interface includes a left-hand sidebar with the **Activities** pane, which lists various activity categories such as **Move & transform**, **Azure Data Explorer**, **Azure Function**, **Batch Service**, **Databricks**, **Data Lake Analytics**, **General**, **HDInsight**, **Iteration & conditionals**, **Machine Learning**, and **Power Query**. The main workspace shows a pipeline flow starting with a **Validation** activity (containing **Check if file exists**), followed by a **Get Metadata** activity (containing **Get File Metadata**). Both activities show a green checkmark, indicating successful execution. The flow then enters an **If Condition** activity titled **If Column Count Matches**. This activity has two paths: a **True** path containing **Copy populatio...** and **Delete source file**, and a **False** path containing **Send Email**. The bottom pane shows the **General** tab for the selected **If Condition** activity, with the **Name** field set to **If Column Count Matches** and a **Description** field below it.

Step 6 - Trigger (ingestion) pipeline

Edit trigger

Name *

Description

Type *

BlobEventsTrigger


Account selection method * ⓘ

☒ From Azure subscription ☐ Enter manually

Azure subscription ⓘ

Eigen abonnement (9b9a9691-6d3d-43d5-84a2-bddc02aa3789) ▾

Storage account name * ⓘ

covidreportingsa0202 ▾ 

Container name * ⓘ

/population/ ▾

Blob path begins with ⓘ

population_by_age.tsv.gz

Blob path ends with ⓘ

Event * ⓘ

☒ Blob created ☐ Blob deleted

Ignore empty blobs * ⓘ

☒ Yes ☐ No

Annotations

+ New

Status ⓘ

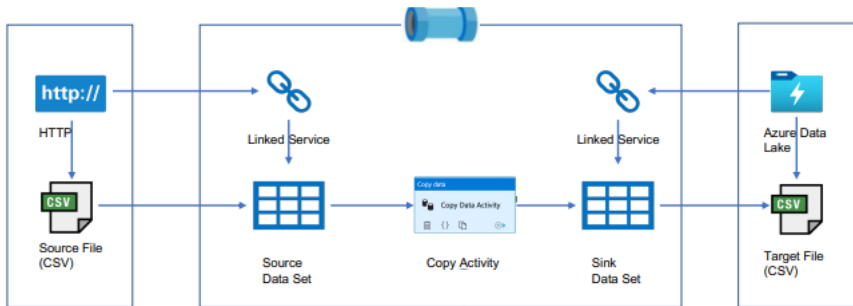
☐ Started ☒ Stopped

Continue

Cancel

3.2 Data ingestion from HTTP

Step 0 - Overview copy activity from HTTP



Step 1 - Created linked services

Edit linked service

HTTP [Learn more](#)

Name *

Is_http_opendata_ecdc_europa_eu

Description

Connect via integration runtime *

AutoResolveIntegrationRuntime

Base URL *

@(linkedService().sourceBaseUrl)

⚠ Information will be sent to the URL specified. Please ensure you trust the URL entered.

Server Certificate Validation

☒ Enable ☐ Disable

Authentication type *

Anonymous

Auth headers

+ New

Annotations

+ New

Parameters

+ New | Delete

<input type="checkbox"/>	Name	Type	Default value	
<input type="checkbox"/>	sourceBaseUrl	String	Value	

> Advanced

Step 1 - Create data set (http - source)

The screenshot displays the Microsoft Fabric Data Factory console. On the left, the 'Factory Resources' pane shows a tree view with 'Pipelines' (9 items) and 'Datasets' (16 items). The 'raw' folder under 'Datasets' is expanded, showing several datasets, with 'ds_ecdc_raw_csv_http' selected. The main pane shows the configuration for this dataset, which is a 'DelimitedText' type. The 'Connection' tab is active, showing the following settings:

- Linked service ***: ls_http_opendata_ecdc_europa_eu
- Linked service properties**:

Name	Value	Type
sourceBaseUrl	@dataset().baseUrl	string
- Relative URL**: @dataset().relativeURL
- Compression type**: None
- Column delimiter**: Comma (,)
- Row delimiter**: Default (\r,\n, or \r\n)
- Encoding**: Default(UTF-8)
- Escape character**: Backslash (\)
- Quote character**: Double quote (")
- First row as header**: ☒
- Null value**:

Step 2 - Create data set (DL sink)

The screenshot displays the Azure Data Factory (ADF) console. On the left, the 'Factory Resources' pane shows a tree view with 'Pipelines' (9 items) and 'Datasets' (16 items). Under 'Datasets', the 'raw' folder is expanded, showing several datasets, with 'ds_ecdc_raw_csv_dl' selected. The main workspace shows a preview of the selected dataset, 'ds_ecdc_raw_csv_dl', which is a 'DelimitedText' file. Below the preview, the 'Connection' tab is active, showing the configuration for the dataset. The configuration includes a linked service 'ls_adls_covidreportingdl', a file path 'raw/ecdc/@dataset().fileName', and various formatting options like compression type, column and row delimiters, encoding, escape and quote characters, and a checked 'First row as header' option.

Factory Resources

- Pipelines (9)
 - Execute (1)
 - Ingest (2)
 - pl_ingest_ecdc_data
 - pl_ingest_population_data
 - Process (3)
 - sqlize (3)
- Datasets (16)
 - lookup (3)
 - processed (4)
 - raw (6)
 - ds_ecdc_raw_csv_dl
 - ds_ecdc_raw_csv_http
 - ds_population_raw_gz
 - ds_population_raw_tsv
 - ds_raw_cases_and_deaths
 - ds_raw_hospital_admissions
 - sql (3)
 - Data flows (2)
 - Power Query (0)

Dataset Configuration: ds_ecdc_raw_csv_dl

Connection

- Linked service *: ls_adls_covidreportingdl
- File path *: raw / ecdc / @dataset().fileName
- Compression type: None
- Column delimiter: Comma (,)
- Row delimiter: Default (\r\n, or \r\n)
- Encoding: Default(UTF-8)
- Escape character: Backslash (\)
- Quote character: Double quote (")
- First row as header: ☒
- Null value:

Step 3 - Create pipeline for data ingestion

The screenshot displays the Azure Data Factory (ADF) interface. On the left, the 'Factory Resources' pane shows a hierarchy of resources: Pipelines (9), Datasets (16), and Data flows (2). The 'pl_ingest_ecdc_data' pipeline is selected. The main canvas shows the pipeline design with a 'Lookup' activity named 'Lookup ECDC File List' connected to a 'ForEach' loop. The 'ForEach' loop is configured to 'Execute copy for every record' and contains a 'Copy ECDC Data' activity. The bottom pane shows the 'General' tab for the selected activity, with the name 'Execute copy for every record' and a description field.

Factory Resources

- Pipelines (9)
 - Execute (1)
 - Ingest (2)
 - pl_ingest_ecdc_data
 - pl_ingest_population_data
 - Process (3)
 - sqlize (3)
- Datasets (16)
 - lookup (3)
 - processed (4)
 - raw (6)
 - ds_ecdc_raw_csv_dl
 - ds_ecdc_raw_csv_http
 - ds_population_raw_gz
 - ds_population_raw_tsv
 - ds_raw_cases_and_deaths
 - ds_raw_hospital_admissions
 - sql (3)
- Data flows (2)
- Power Query (0)

Activities

- Move & transform
- Azure Data Explorer
- Azure Function
- Batch Service
- Databricks
- Data Lake Analytics
- General
- HDInsight
- Iteration & conditionals
- Machine Learning
- Power Query

Lookup

- Lookup ECDC File List

ForEach

- Execute copy for every record
- Copy ECDC Data

General

Name * Execute copy for every record [Learn more](#)

Description

Step 4 - Create trigger for pipeline

Edit trigger

Name *

tr_ingest_ecdc_data

Description

Type *

TumblingWindowTrigger

Start Date (UTC) *

12/19/22 00:00:00

Recurrence * ⓘ

Every 24

Hour(s)

☐ Specify an end date

> Advanced

Annotations

+ New

Status ⓘ

☒ Started ☐ Stopped

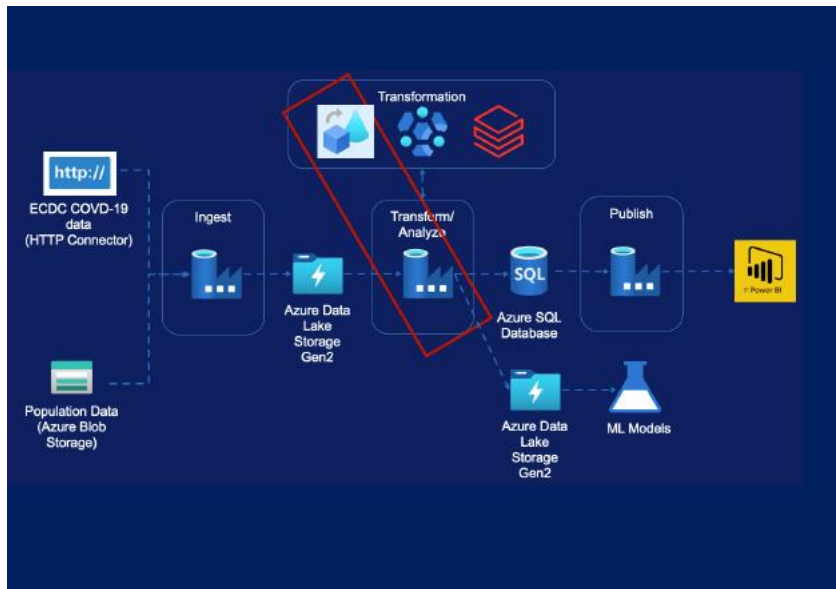
Step 6 - Ecdc file list data set

The screenshot displays the Azure Data Factory (ADF) interface. On the left, the 'Factory Resources' pane shows a tree view with categories: Pipelines (9), Datasets (16), lookup (3), processed (4), raw (6), sql (3), Data flows (2), and Power Query (0). The 'ds_ecdc_file_list' dataset is selected under the 'lookup' category. The main pane shows the configuration for this dataset, which is a JSON file. The 'Connection' tab is active, showing the 'Linked service' as 'ls_ablob_covidreportingsa'. The 'File path' is set to 'configs / Directory / ecdc_file_list.json'. The 'Compression type' is 'None' and the 'Encoding' is 'Default(UTF-8)'. The 'Preview experience' toggle is set to 'Off'.

4. Data transformation

4.1 Data flow

Step 0 – Overview transformation in data flow



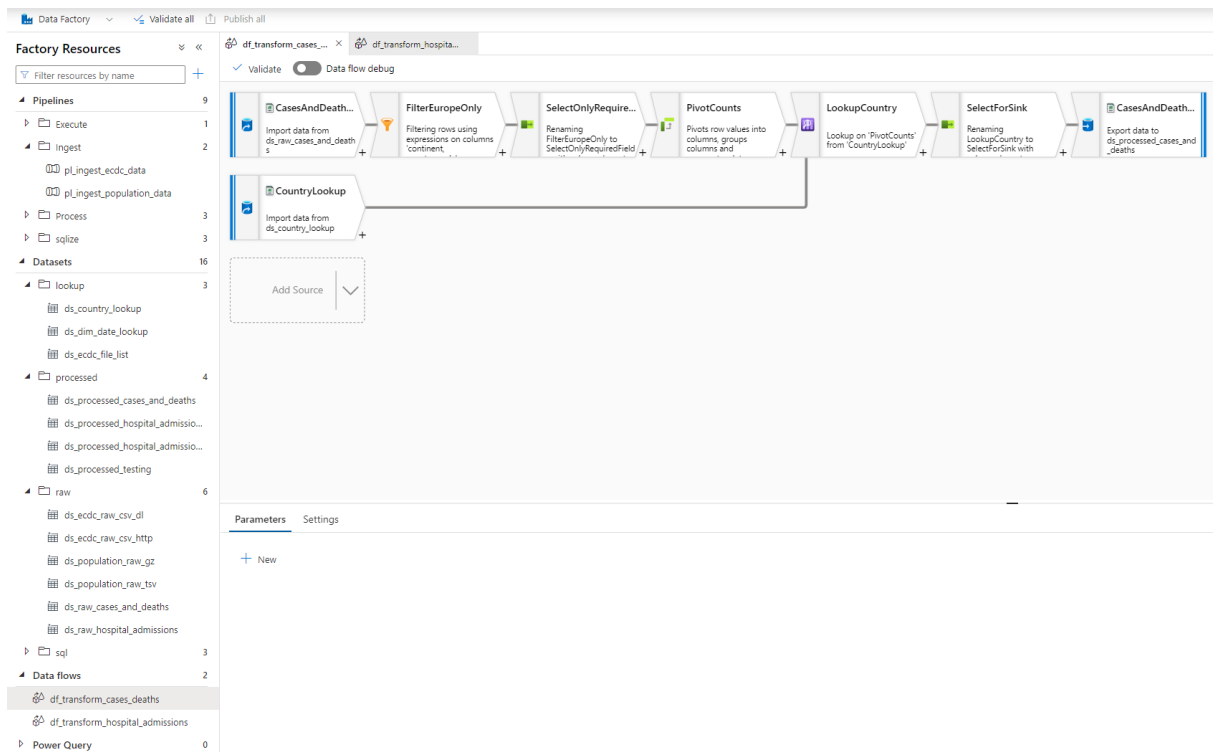
(1) Steps taken:

- Source Transformation
- Filter Transformation
- Select Transformation
- Pivot Transformation
- Lookup Transformation
- Sink Transformation
- Create Pipeline

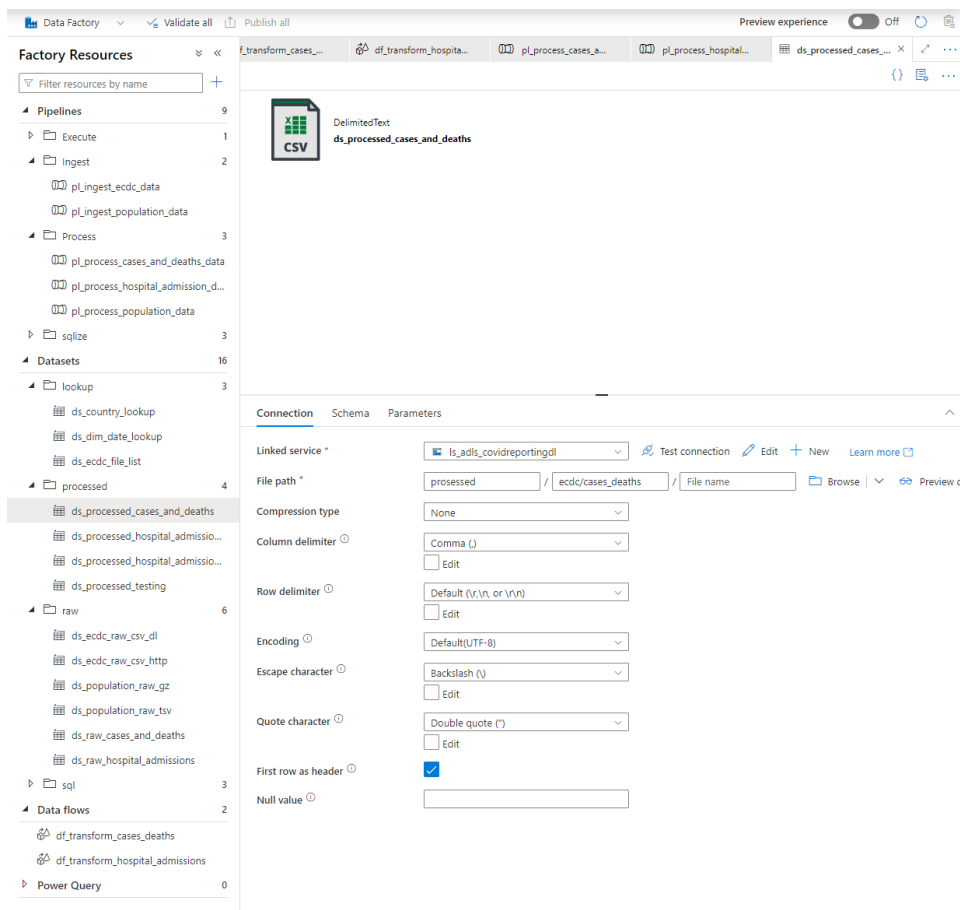
(2) Steps taken:

- Source Transformation
- Select Transformation
- Lookup Transformation
- Pivot Transformation
- Sink Transformation
- Conditional Split Transformation
- Derived Column Transformation
- Aggregate Transformation
- Sort Transformation
- Join Transformation
- Select Transformation
- Create Pipeline

(1) Step 1 - Create transformations with data flow



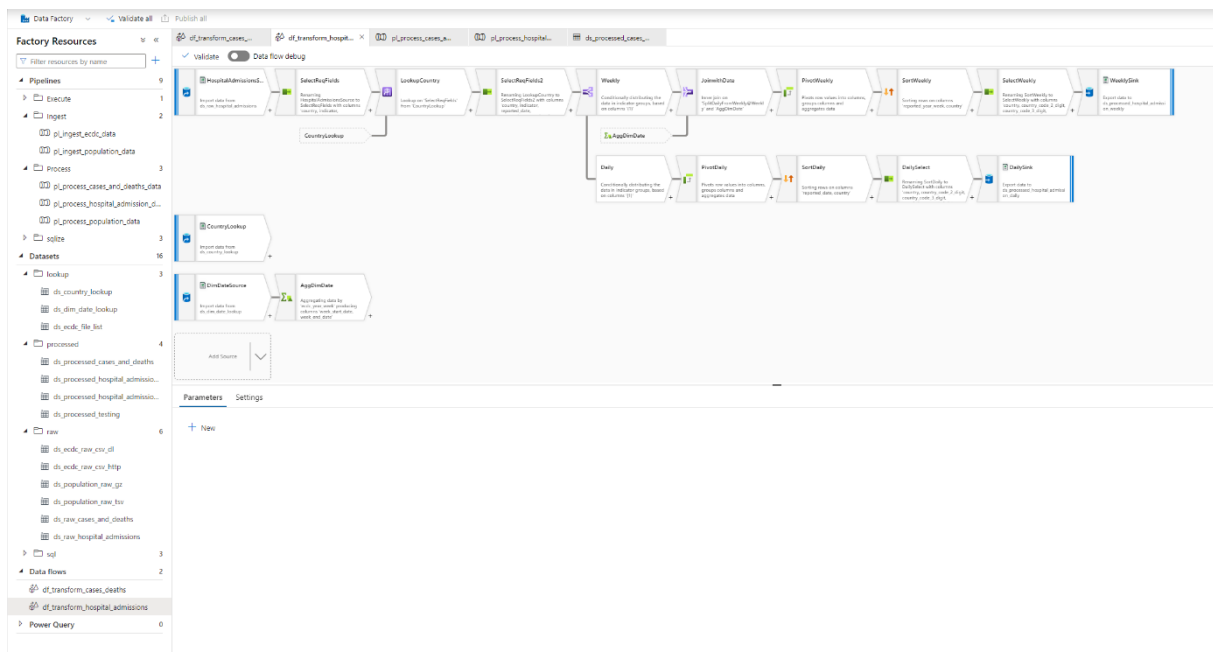
(1) Step 2 - Create data set for processed data



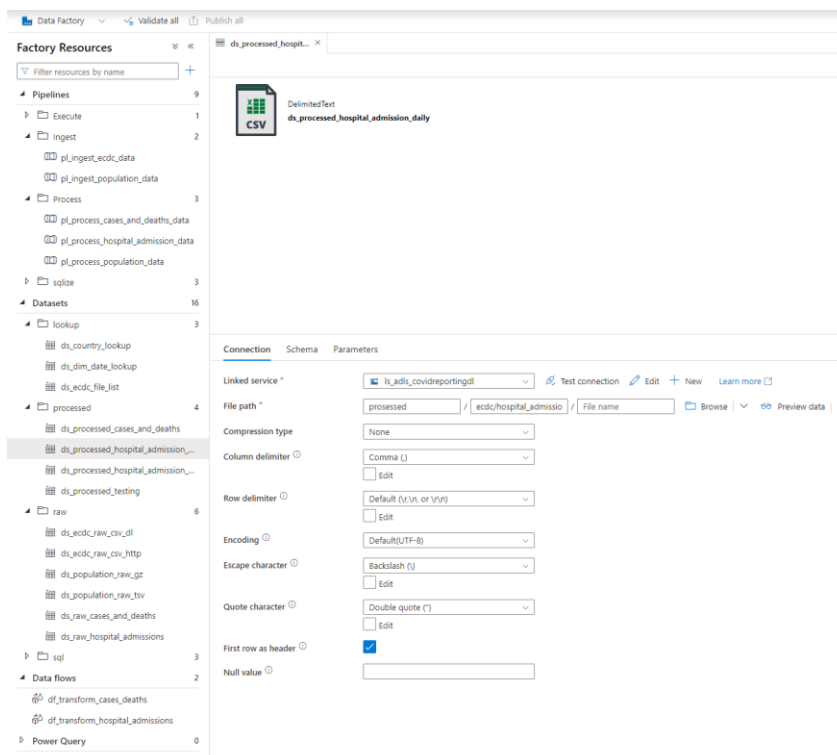
(1) Step 3 – Create pipeline

The screenshot displays the Azure Data Factory (ADF) interface. On the left, the 'Factory Resources' pane shows a hierarchical view of resources. The 'Pipelines' folder is expanded, showing a list of pipelines including 'pl_ingest_ecdc_data', 'pl_ingest_population_data', 'pl_process_cases_and_deaths_data', 'pl_process_hospital_admission_d...', and 'pl_process_population_data'. The 'Data flows' folder is also visible, containing 'df_transform_cases_deaths' and 'df_transform_hospital_admissions'. The 'Activities' pane in the center lists various activities such as 'Move & transform', 'Azure Data Explorer', 'Azure Function', 'Batch Service', 'Databricks', 'Data Lake Analytics', 'General', 'HDInsight', 'Iteration & conditionals', 'Machine Learning', and 'Power Query'. The right pane shows the configuration for a 'Data flow' named 'Data flow1'. The 'Settings' tab is active, displaying options for 'Data flow *', 'Run on (Azure IR) *', 'Compute size *', 'Logging level *', 'Sink properties', and 'Staging'. The 'Data flow *' dropdown is set to 'df_transform_cases_deaths', 'Run on (Azure IR) *' is set to 'AutoResolveIntegrationRuntime', and 'Compute size *' is set to 'Small'. The 'Logging level *' is set to 'Verbose'.

(2) Step 4 - Create transformations with data flow



Step 5 - Create data set for processed data (daily)



Step 6 - Create data set for processed data (weekly)

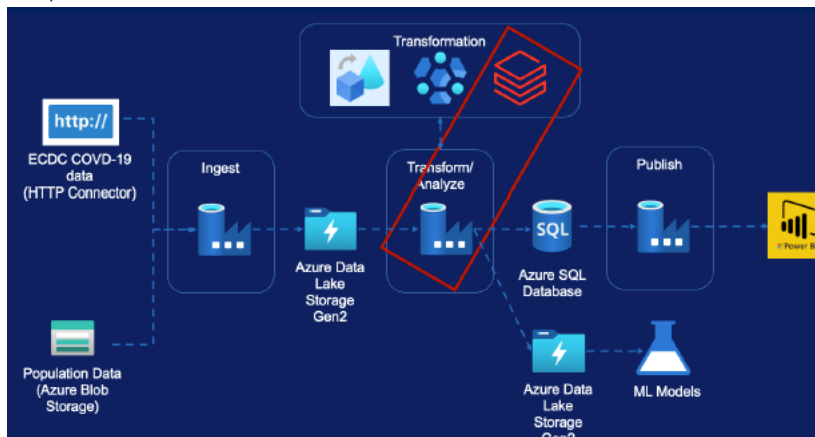
The screenshot displays the Microsoft Fabric Data Factory console. On the left, the 'Factory Resources' pane shows a tree view of the workspace. Under the 'Datasets' section, a new dataset named 'ds_processed_hospital_admission_weekly' is being created. The main pane shows the configuration for this dataset, which is a 'DelimitedText' type. The configuration is as follows:

- Linked service:** ls_adls_covidreportingdl
- File path:** processed / ecdc/hospital_admissio / File name
- Compression type:** None
- Column delimiter:** Comma (,)
- Row delimiter:** Default (\r\n, or \n)
- Encoding:** Default(UTF-8)
- Escape character:** Backslash (\)
- Quote character:** Double quote (")
- First row as header:** ☒
- Null value:** (empty field)

Step 7 - Create pipeline

4.2 Databricks

Step 0 - Databricks overview



Steps taken:

- Create Databricks Service
- Create Databricks Cluster
- Mount Storage Accounts
- Transformation requirements
- Creating Pipeline

Step 1 – Mount cluster 1 / 2

mount_storage Python

File Edit View Run Help Last edit was 22 days ago Give feedback

Run all Unknown Schedule Share

Cmd 1

Mount the following data lake storage gen2 containers

1. raw
2. processed
3. lookup

Cmd 2

Set-up the configs

Please update the following

- application-id
- service-credential
- directory-id

Cmd 3

```
1 configs = {"fs.azure.account.auth.type": "OAuth",
2           "fs.azure.account.oauth.provider.type": "org.apache.hadoop.fs.azurebfs.oauth2.ClientCredsTokenProvider",
3           "fs.azure.account.oauth2.client.id": "47846408-5bca-44f4-9de3-71fd3b08a9d6",
4           "fs.azure.account.oauth2.client.secret": "rty8Q-FbCdPpT37mATANTCisANbkYI.XyOYMasI",
5           "fs.azure.account.oauth2.client.endpoint": "https://login.microsoftonline.com/8603b7a6-42b5-4a50-a399-b1332d779cb6/oauth2/token"}
```

Command took 0.04 seconds -- by ossy-80@live.nl at 12/14/2022, 3:47:26 PM on Covid-reporting-cluster

Cmd 4

Mount the raw container

Update the storage account name before executing

Cmd 5

```
1 dbutils.fs.mount(
2   source = "abfss://raw@covidreportingdl020.dfs.core.windows.net/",
3   mount_point = "/mnt/covidreportingdl020/raw",
4   extra_configs = configs)
```

▶ (1) Spark Jobs

Out[4]: True

Command took 24.09 seconds -- by ossy-80@live.nl at 12/14/2022, 3:48:47 PM on Covid-reporting-cluster

Cmd 6

Mount the processed container

Update the storage account name before executing

Cmd 7

```
1 dbutils.fs.mount(
2   source = "abfss://processed@covidreportingdl020.dfs.core.windows.net/",
3   mount_point = "/mnt/covidreportingdl020/processed",
```

Step 1 – Mount cluster 2 / 2

mount_storage Python

File Edit View Run Help [Last edit was 22 days ago](#) [Give feedback](#)

▶ Run all ● Unknown ▼ Schedule Share

4

```
extra_configs = configs)
```

▶ (1) Spark Jobs

Out[8]: True

Command took 20.83 seconds -- by ossy-80@live.nl at 12/14/2022, 3:53:14 PM on Covid-reporting-cluster

Cmd 8

Mount the lookup container

Update the storage account name before executing

Cmd 9

1

```
dbutils.fs.mount(
2   source = "abfss://lookup@covidreportingdl020.dfs.core.windows.net/",
3   mount_point = "/mnt/covidreportingdl020/lookup",
4   extra_configs = configs)
```

▶ (1) Spark Jobs

Out[6]: True

Command took 21.14 seconds -- by ossy-80@live.nl at 12/14/2022, 3:50:45 PM on Covid-reporting-cluster

Cmd 10

1

```
dbutils.fs.ls("/mnt/covidreportingdl020/lookup")
```

Out[9]: [FileInfo(path='dbfs:/mnt/covidreportingdl020/lookup/dim_country/', name='dim_country/', size=0),
FileInfo(path='dbfs:/mnt/covidreportingdl020/lookup/dim_date/', name='dim_date/', size=0)]

Command took 0.26 seconds -- by ossy-80@live.nl at 12/14/2022, 3:54:21 PM on Covid-reporting-cluster

Cmd 11

1

Shift+Enter to run

24

Step 2 – transform data 1 / 2

transform_population_data Python
File Edit View Run Help Last edit was 22 days ago Give feedback

Run all Connect Schedule Share

Cmd 3

1 from pyspark.sql.functions import *

Cmd 4

Read the population data & create a temp view

Cmd 5

1 df_raw_population = spark.read.csv("/mnt/covidreportingdl020/raw/population", sep=r'\t', header=True)
2 df_raw_population = df_raw_population.withColumn('age_group', regexp_replace(split(df_raw_population['indic_de,geo\time'], ',')[0], 'PC_', ''))
3 df_raw_population = df_raw_population.withColumn('country_code', split(df_raw_population['indic_de,geo\time'], ',')[1])
4 df_raw_population = df_raw_population.select(col("country_code").alias("country_code"),
5 col("age_group").alias("age_group"),
6 col("2019 ").alias("percentage_2019"))
df_raw_population.createOrReplaceTempView("raw_population")

Cmd 6

Pivot the data by age group

Cmd 7

1 # Create a data frame with pivoted percentages
2 df_raw_population_pivot = spark.sql("SELECT country_code, age_group, cast(regexp_replace(percentage_2019, '[a-z]', '') AS decimal(4,2)) AS
percentage_2019 FROM raw_population WHERE length(country_code) =
2").groupBy("country_code").pivot("age_group").sum("percentage_2019").orderBy("country_code")
3 df_raw_population_pivot.createOrReplaceTempView("raw_population_pivot")

Cmd 8

Read the country lookup

Cmd 9

1 # Create a data frame for the country lookup
2 df_dim_country = spark.read.csv("/mnt/covidreportingdl020/lookup/dim_country", sep=r',', header=True)
3 df_dim_country.createOrReplaceTempView("dim_country")

Cmd 10

Join population data with country lookup

Cmd 11

1 df_processed_population = spark.sql("""SELECT c.country,
2 c.country_code_2_digit,
3 c.country_code_3_digit,
4 c.population,
5 p.Y0_14 AS age_group_0_14,
6 p.Y15_24 AS age_group_15_24,
7 p.Y25_49 AS age_group_25_49,
8 p.Y50_64 AS age_group_50_64,
9 p.Y65_79 AS age_group_65_79,
10 p.Y80_MAX AS age_group_80_max
11 FROM raw_population_pivot p
12 JOIN dim_country c ON c.country_code_2_digit = p.country_code_2_digit
13 AND c.country_code_3_digit = p.country_code_3_digit""")

Step 2 – transform data 2 / 2

transform_population_data Python

File Edit View Run Help Last edit was 22 days ago Give feedback

```
5      p.Y0_14 AS age_group_0_14,  
6      p.Y15_24 AS age_group_15_24,  
7      p.Y25_49 AS age_group_25_49,  
8      p.Y50_64 AS age_group_50_64,  
9      p.Y65_79 AS age_group_65_79,  
10     p.Y80_MAX AS age_group_80_max  
11     FROM raw_population_pivot p  
12     JOIN dim_country c ON p.country_code = country_code_2_digit  
13     ORDER BY country""")
```

Cmd 12

Write output to the processed mount point

Cmd 13

```
1 df_processed_population.write.format("com.databricks.spark.csv").option("header","true").option("delimiter",  
",").mode("overwrite").save("/mnt/covidreportingdl020/prosessed/population")
```

Cmd 14

```
1
```

Shift+Enter to run

Step 3 – Create linked service

Edit linked service
Azure Databricks Learn more

Name *
ls_db_covid_cluster

Description

Connect via integration runtime *
AutoResolveIntegrationRuntime

Account selection method *
☐ From Azure subscription ☒ Enter manually

Databricks Workspace URL *
https://adb-3875266610093583.3.azuredatabricks.net

Authentication type *
Access Token

Access token

Azure Key Vault

Access token *

Select cluster
☒ New job cluster ☐ Existing interactive cluster ☐ Existing instance pool

Cluster version *
7.3.x-cpu-ml-scala2.12

Cluster node type *
Standard_DS3_v2

Python Version *
3

Worker options
☒ Fixed ☐ Autoscaling

Workers *
1

> Additional cluster settings

Annotations

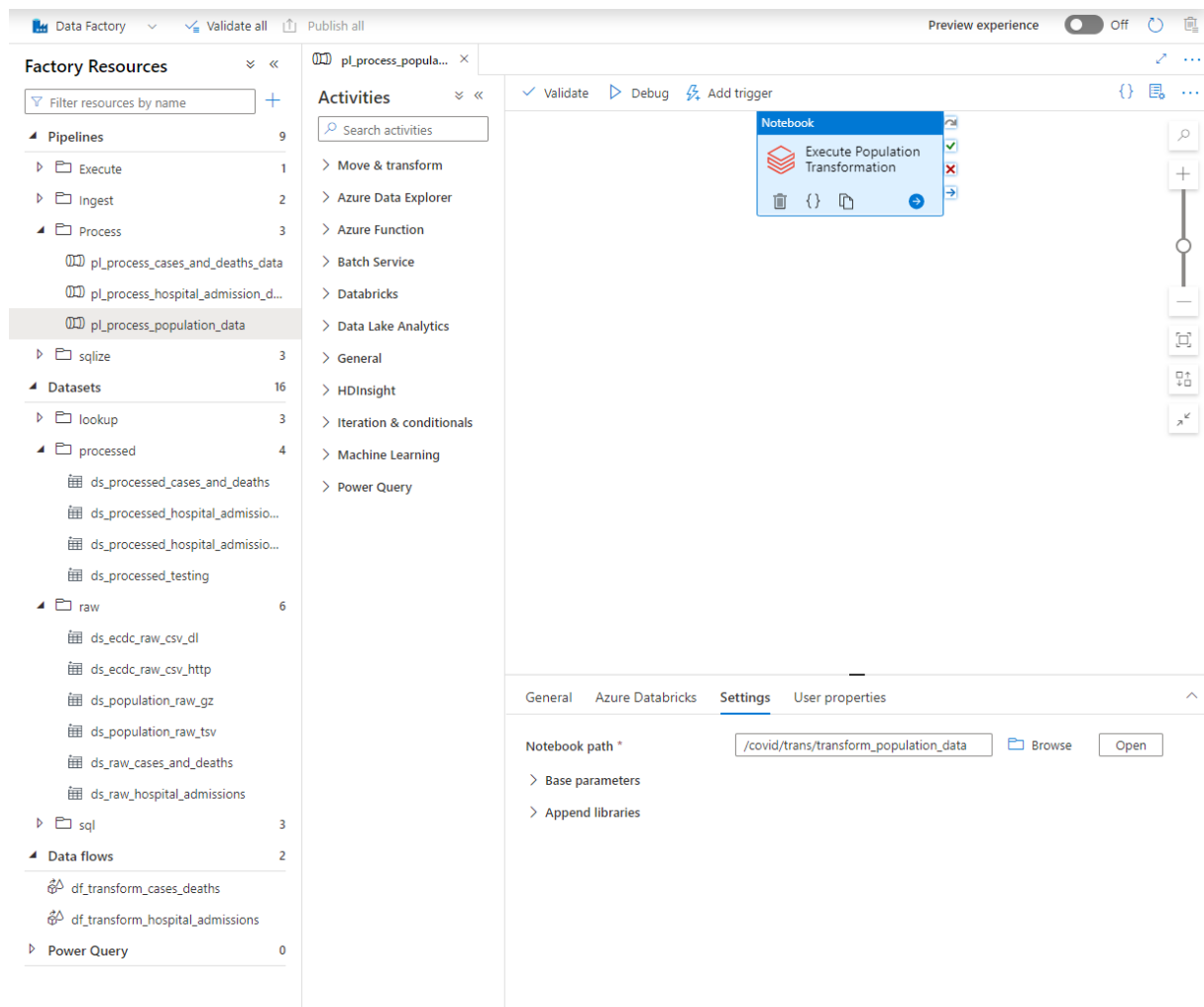
+ New

> Parameters

> Advanced

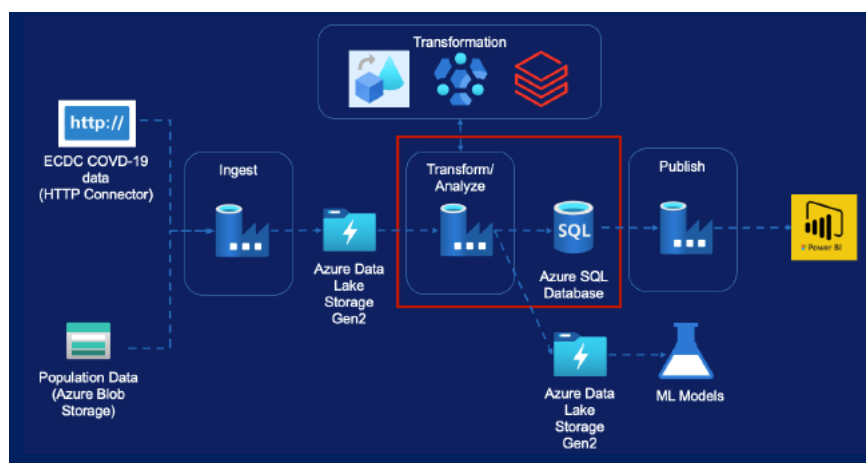
Save Cancel Test connection

Step 4 – Create pipeline



5. Load into database

Step 0 – Overview SQL database



Steps taken:


- Copy Cases & Deaths data
- Copy Hospital Admissions data
- Copy testing data

Step 1 – Create SQL script

```
1 CREATE SCHEMA covid_reporting
2 GO
3
4 CREATE TABLE covid_reporting.cases_and_deaths
5 (
6     country          VARCHAR(100),
7     country_code_2_digit VARCHAR(2),
8     country_code_3_digit VARCHAR(3),
9     population        BIGINT,
10    cases_count        BIGINT,
11    deaths_count       BIGINT,
12    reported_date      DATE,
13    source             VARCHAR(500)
14 )
15 GO
16
17 CREATE TABLE covid_reporting.hospital_admissions_daily
18 (
19     country          VARCHAR(100),
20     country_code_2_digit VARCHAR(2),
21     country_code_3_digit VARCHAR(3),
22     population        BIGINT,
23     reported_date     DATE,
24     hospital_occupancy_count BIGINT,
25     icu_occupancy_count  BIGINT,
26     source            VARCHAR(500)
27 )
28 GO
29
30 CREATE TABLE covid_reporting.testing
31 (
32     country          VARCHAR(100),
33     country_code_2_digit VARCHAR(2),
34     country_code_3_digit VARCHAR(3),
35     year_week        VARCHAR(8),
36     week_start_date   DATE,
37     week_end_date     DATE,
38     new_cases         BIGINT,
39     tests_done        BIGINT,
40     population        BIGINT,
41     testing_data_source VARCHAR(500)
42 )
43 GO
44
```

Step 2 – Create linked service

Edit linked service

 Azure SQL Database [Learn more](#)

Name *

ls_sql_covid_db

Description

Connect via integration runtime * ⓘ

AutoResolveIntegrationRuntime

Connection string

[Azure Key Vault](#)

Account selection method ⓘ

☐ From Azure subscription ☒ Enter manually

Fully qualified domain name *

covid-db020.database.windows.net

Database name *

covid-db020

Authentication type *

SQL authentication

User name *

admin020

Password

[Azure Key Vault](#)

Password *

Always encrypted ⓘ

☐

Additional connection properties

+ New

Annotations

+ New

> Parameters

> Advanced ⓘ

Step 3 - Create sink dataset

The screenshot displays the Azure Data Factory (ADF) console. On the left, the 'Factory Resources' pane shows a tree view of the workspace. Under 'Databases', the 'ds_sql_cases_and_deaths' database is selected. The main pane shows the configuration for this database. The 'Connection' tab is active, showing the 'Linked service' as 'ls_sql_covid_db' and the 'Table' as 'covid_reporting.cases_and_deaths'. The 'Schema' and 'Parameters' tabs are also visible. The top of the console shows the 'Data Factory' name, 'Validate all' button, 'Publish all' button, and 'Preview experience' toggle.

Factory Resources

- Pipelines: 9
 - Execute: 1
 - Ingest: 2
 - Process: 3
 - sqlize: 3
 - pl_sqlize_cases_and_deaths_data
 - pl_sqlize_hospital_admissions_da...
 - pl_sqlize_testing
- Datasets: 16
 - lookup: 3
 - processed: 4
 - ds_processed_cases_and_deaths
 - ds_processed_hospital_admissio...
 - ds_processed_hospital_admissio...
 - ds_processed_testing
 - raw: 6
 - ds_ecdc_raw_csv_dl
 - ds_ecdc_raw_csv_http
 - ds_population_raw_gz
 - ds_population_raw_tsv
 - ds_raw_cases_and_deaths
 - ds_raw_hospital_admissions
 - sql: 3
 - ds_sql_cases_and_deaths**
 - ds_sql_hospital_admissions_daily
 - ds_sql_testing
- Data flows: 2
 - df_transform_cases_deaths
 - df_transform_hospital_admissions
- Power Query: 0

ds_sql_cases_and_deaths

Connection Schema Parameters

Linked service * **ls_sql_covid_db** Test connection Edit + New Learn more

Table **covid_reporting.cases_and_deaths** Refresh Preview data Edit

Step 4 – Create pipeline

The screenshot displays the Azure Data Factory (ADF) interface. The top navigation bar includes 'Data Factory', 'Validate all', 'Publish all', and a 'Preview experience' toggle set to 'Off'. The left sidebar, titled 'Factory Resources', shows a tree view of the workspace. Under 'Pipelines', the 'sqlize' folder is expanded, showing 'pl_sqlize_cases_and_deaths_data' selected. The 'Activities' pane on the right lists various activity types, with 'Copy data' selected. The main canvas shows a pipeline named 'pl_sqlize_cases_and_deaths_data' with a single activity, 'Copy Cases and Deaths', which is highlighted with a green checkmark. The bottom pane shows the 'Parameters' tab with a '+ New' button.

Factory Resources

- Pipelines (9)
 - Execute (1)
 - Ingest (2)
 - Process (3)
 - sqlize (3)
 - pl_sqlize_cases_and_deaths_data
 - pl_sqlize_hospital_admissions_da...
 - pl_sqlize_testing
- Datasets (16)
 - lookup (3)
 - processed (4)
 - ds_processed_cases_and_deaths
 - ds_processed_hospital_admissio...
 - ds_processed_hospital_admissio...
 - ds_processed_testing
 - raw (6)
 - ds_ecdc_raw_csv_dl
 - ds_ecdc_raw_csv_http
 - ds_population_raw_gz
 - ds_population_raw_tsv
 - ds_raw_cases_and_deaths
 - ds_raw_hospital_admissions
 - sql (3)
 - ds_sql_cases_and_deaths
 - ds_sql_hospital_admissions_daily
 - ds_sql_testing
- Data flows (2)
 - df_transform_cases_deaths
 - df_transform_hospital_admissions
- Power Query (0)

Activities

- Move & transform
- Azure Data Explorer
- Azure Function
- Batch Service
- Databricks
- Data Lake Analytics
- General
- HDInsight
- Iteration & conditionals
- Machine Learning
- Power Query

Copy data

- Copy Cases and Deaths

Parameters

- + New

Step 5 – Create pipeline hospital admission data

The screenshot displays the Azure Data Factory (ADF) console interface. The top navigation bar includes 'Data Factory', 'Validate all', 'Publish all', and a 'Preview experience' toggle set to 'Off'. The main workspace is divided into three panels:

- Factory Resources:** A tree view on the left showing the hierarchy of resources. Under 'Pipelines', 'pl_sqlize_hospital_admissions_data...' is selected. Under 'Datasets', the 'raw' folder contains several datasets, including 'ds_raw_hospital_admissions'.
- Activities:** A central panel with a search bar and a list of activity categories. The 'Copy data' activity is highlighted, and a specific activity named 'Copy Hospital Admissions Data' is shown with a green checkmark.
- Parameters:** A bottom panel with tabs for 'Parameters', 'Variables', 'Settings', and 'Output'. The 'Parameters' tab is active, showing a '+ New' button.

The right side of the interface features a vertical toolbar with icons for zooming, panning, and other navigation functions.

Step 6 - Create pipeline testing data

The screenshot displays the Azure Data Factory (ADF) interface. The top navigation bar includes 'Data Factory', 'Validate all', 'Publish all', and a 'Preview experience' toggle set to 'Off'. The left sidebar, titled 'Factory Resources', shows a tree view of resources. Under 'Pipelines', 'pl_sqlize_testing' is selected. The 'Activities' pane on the right lists various activity types, with 'Copy data' highlighted. The main canvas shows a single 'Copy data' activity named 'Copy testing'. The bottom pane is currently empty, showing a 'Parameters' tab with a '+ New' button.

Factory Resources

- Pipelines (9)
 - Execute (1)
 - Ingest (2)
 - Process (3)
 - sqlize (3)
 - pl_sqlize_cases_and_deaths_data
 - pl_sqlize_hospital_admissions_da...
 - pl_sqlize_testing
- Datasets (16)
 - lookup (3)
 - processed (4)
 - ds_processed_cases_and_deaths
 - ds_processed_hospital_admissio...
 - ds_processed_hospital_admissio...
 - ds_processed_testing
 - raw (6)
 - ds_ecdc_raw_csv_dl
 - ds_ecdc_raw_csv_http
 - ds_population_raw_gz
 - ds_population_raw_tsv
 - ds_raw_cases_and_deaths
 - ds_raw_hospital_admissions
 - sql (3)
 - ds_sql_cases_and_deaths
 - ds_sql_hospital_admissions_daily
 - ds_sql_testing
- Data flows (2)
 - df_transform_cases_deaths
 - df_transform_hospital_admissions
- Power Query (0)

Activities

- Move & transform
- Azure Data Explorer
- Azure Function
- Batch Service
- Databricks
- Data Lake Analytics
- General
- HDInsight
- Iteration & conditionals
- Machine Learning
- Power Query

Copy data

Copy testing

Parameters Variables Settings Output

+ New

6. Data orchestration – Making pipelines production ready

Step 1 – Build pipeline

The screenshot displays the Azure Data Factory (ADF) interface. On the left, the 'Factory Resources' pane shows a tree view with 'Pipelines' expanded, containing 'pl_execute_population_pipelines'. The main canvas shows a pipeline diagram with two 'Execute Pipeline' activities. The first activity is 'Execute Ingest Population Data' and the second is 'Execute Process Population Data', connected by a sequence arrow. The 'Activities' pane on the left lists various activity types like 'Move & transform', 'Azure Data Explorer', etc. The bottom pane shows the 'Parameters' tab with a '+ New' button.

Step 2 – Create trigger

Edit trigger

Name *

tr_population_data_arrived

Description

Type *

BlobEventsTrigger

Account selection method * ⓘ

☒ From Azure subscription ☐ Enter manually

Azure subscription ⓘ

Eigen abonnement (9b9a9691-6d3d-43d5-84a2-bddc02aa3789) ▾

Storage account name * ⓘ

covidreportingsa0202 ▾

Container name * ⓘ

population ▾

Blob path begins with ⓘ

population_by_age.tsv.gz

Blob path ends with ⓘ

Event * ⓘ

☒ Blob created ☐ Blob deleted

Ignore empty blobs * ⓘ

☒ Yes ☐ No

Annotations

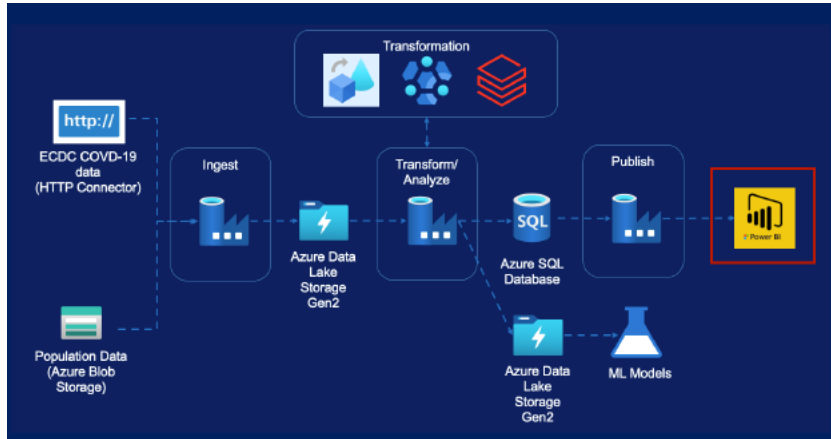
+ New

Status ⓘ

☐ Started ☒ Stopped

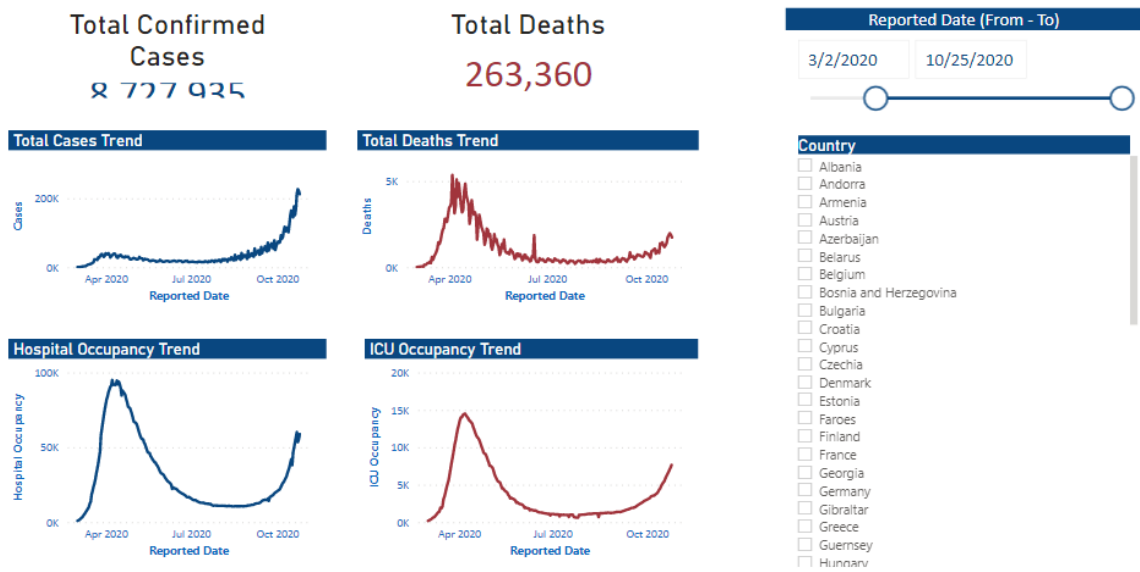
7. Data reporting

Step 0 – Reporting overview in Power BI



Step 1 – Create report 1 / 2

Covid-19 Cases EU/EEA & UK



Step 1 – Create report 2 / 2

Covid-19 Cases UK, France & Germany

Total Cases Trend - United Kingdom



Total Deaths Trend - United Kingdom



Reported Date (From - To)

8/24/2020

10/25/2020

Total Cases Trend - France



Total Deaths Trend - France



Total Cases Trend - Germany

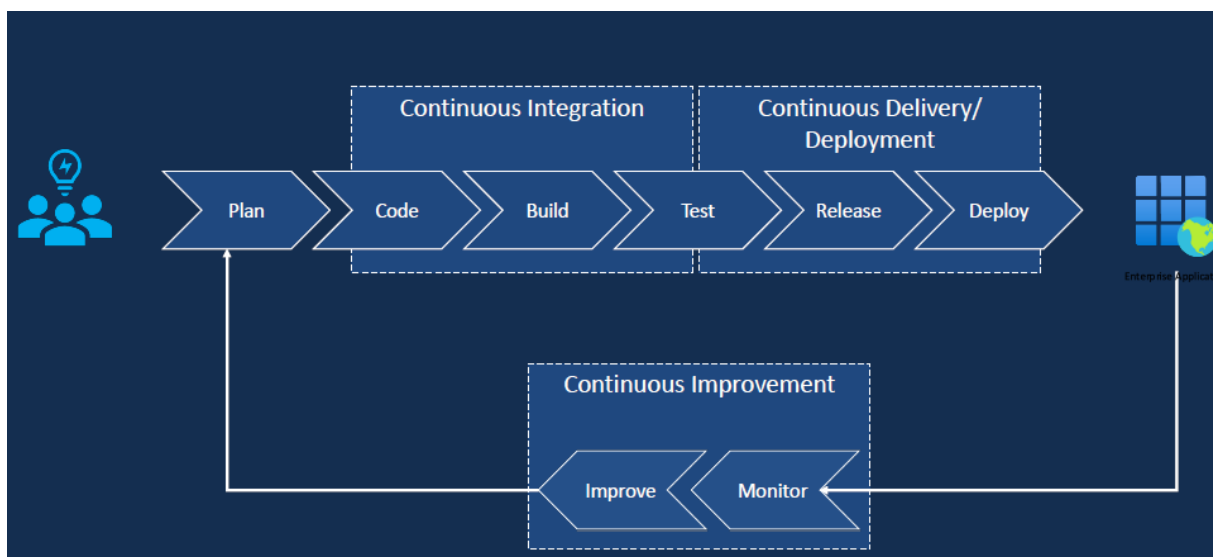
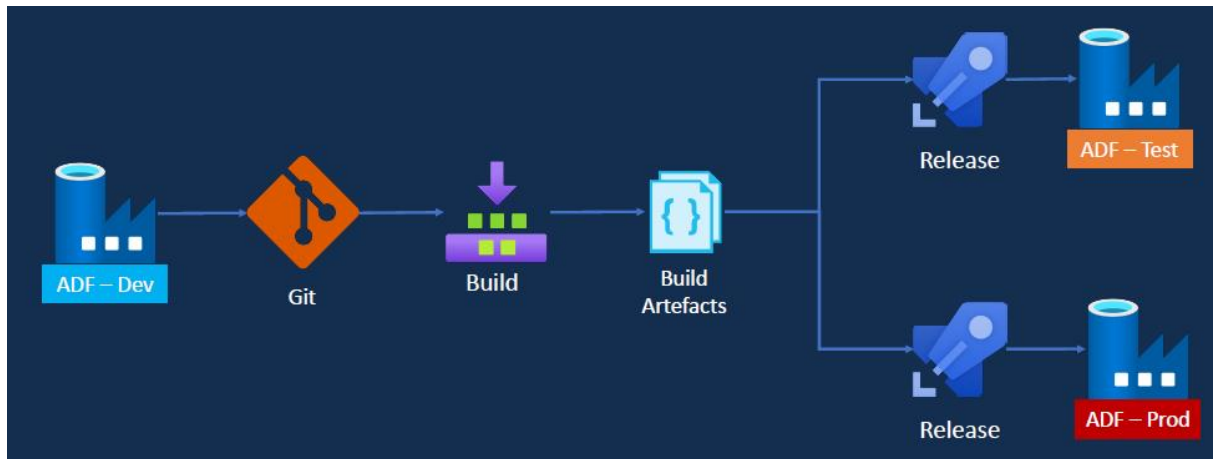


Total Deaths Trend - Germany

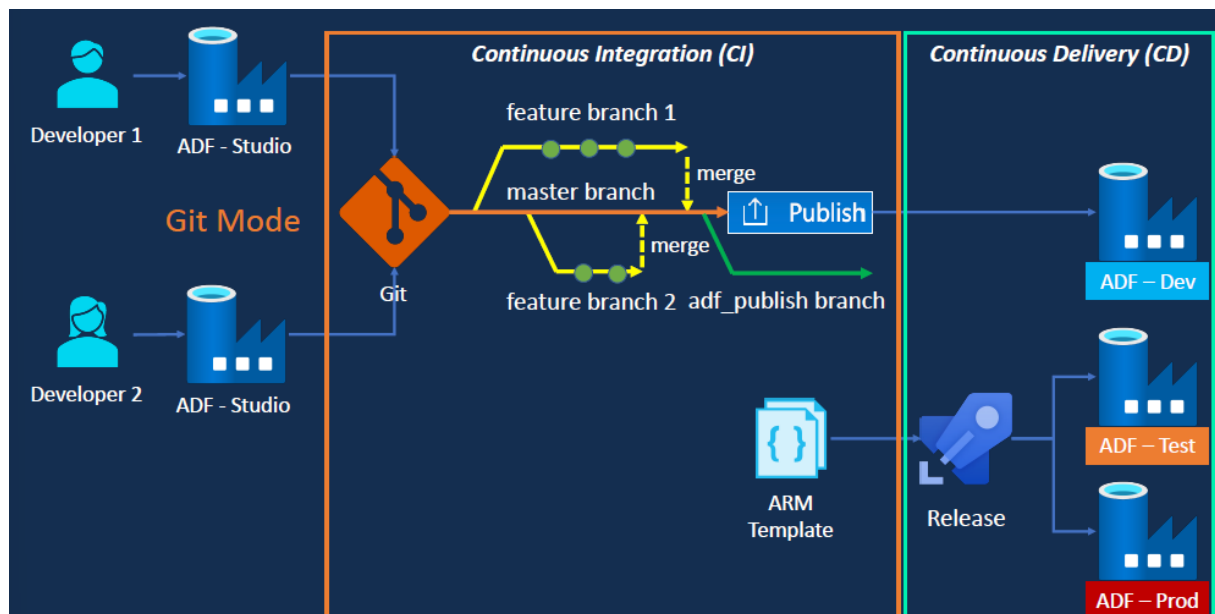


8. CI / CD

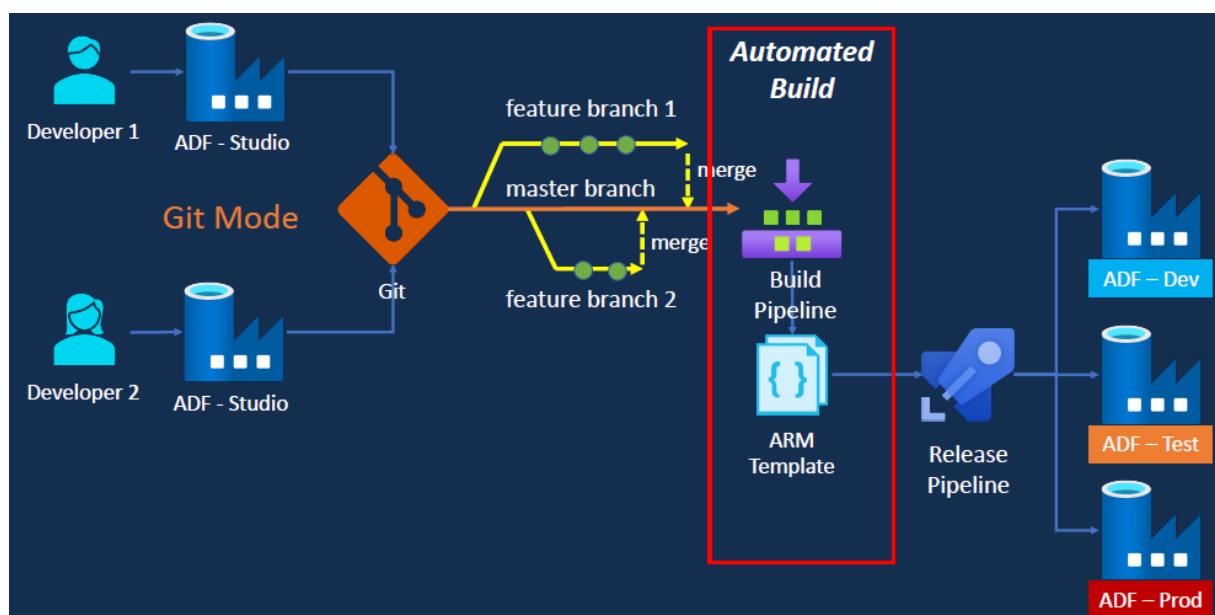
Step 0 – CI / CD overview



Step 0 – Git configuration within DevOps option 1



Step 0 – Git configuration within DevOps option 2



Step 1 – Create git repo

Azure Data Factory CI CD / Repos / Files / dev-ci-cd-adf-020

Search

BS

dev-ci-cd-adf-020

> build

> dataset

> linkedService

> pipeline

> release

> trigger

MI README.md

main

Type to find a file or folder...

Files

succeeded

Clone

Contents

History

Name ↑	Last change	Commits
build	Dec 27, 2022	4e4a2822 Added 2 files t...
dataset	Yesterday	593bfe03 Adding pipelin...
linkedService	Yesterday	93854955 Adding linkedS...
pipeline	Yesterday	593bfe03 Adding pipelin...
release	Dec 27, 2022	5a9e69e2 Added PrePost...
trigger	Dec 27, 2022	d8afe45f Updating trigg...
MI README.md	Dec 21, 2022	56ab35af Added READM...

Introduction

TODO: Give a short introduction of your project. Let this section explain the objectives or the motivation behind this project.

Getting Started

TODO: Guide users through getting your code up and running on their own system. In this section you can talk about:

1. Installation process
2. Software dependencies
3. Latest releases
4. API references

Build and Test

TODO: Describe and show how to build your code and run the tests.

Contribute

TODO: Explain how other users and developers can contribute to make your code better.

If you want to learn more about creating good readme files then refer the following [guidelines](#). You can also seek inspiration from the below readme files:

- [ASP.NET Core](#)
- [Visual Studio Code](#)
- [Chakra Core](#)

Step 2 – Create Tools for azure (DEV)

dev-ci-cd-rg Resource group

Search

Overview

- Activity log
- Access control (IAM)
- Tags
- Resource visualizer
- Events

Settings

- Deployments
- Security
- Policies
- Properties
- Locks

Cost Management

- Cost analysis
- Cost alerts (preview)

Essentials

Subscription (move) [Eigen abbonement](#)

Subscription ID: 9b9a9691-6d3d-43d5-84a2-bddc02aa3789

Tags (edit) [Click here to add tags](#)

Deployments: [3 Succeeded](#)

Location: East US

Resources Recommendations

Filter for any field... Type equals all Location equals all Add filter

Showing 1 to 3 of 3 records. ☐ Show hidden types

Name	Type	Location
<input type="checkbox"/> dev-ci-cd-adf-020	Data factory (V2)	East US
<input type="checkbox"/> dev-ci-cd-kv-020-1	Key vault	East US
<input type="checkbox"/> devcicddl	Storage account	East US

Step 2 – Create Tools for azure (TEST)

test-ci-cd-rg Resource group

Search

Overview

- Activity log
- Access control (IAM)
- Tags
- Resource visualizer
- Events

Settings

- Deployments
- Security
- Policies
- Properties
- Locks

Cost Management

- Cost analysis

Essentials

Subscription (move) [Eigen abbonement](#)

Subscription ID: 9b9a9691-6d3d-43d5-84a2-bddc02aa3789

Tags (edit) [Click here to add tags](#)

Deployments: [5 Failed](#) [6 Succeeded](#)

Location: East US

Resources Recommendations

Filter for any field... Type equals all Location equals all Add filter

Showing 1 to 3 of 3 records. ☐ Show hidden types

Name	Type	Location
<input type="checkbox"/> test-ci-cd-adf-020	Data factory (V2)	East US
<input type="checkbox"/> test-ci-cd-kv-020	Key vault	East US
<input type="checkbox"/> testcicddl	Storage account	East US

Step 2 – Create Tools for azure (PROD)

prod-ci-cd-rg Resource group

Search

Essentials

Subscription ([move](#))
[Eigen abonnement](#)
Subscription ID
9b9a9691-6d3d-43d5-84a2-bddc02aa3789
Tags ([edit](#))
[Click here to add tags](#)

Deployments
[3 Succeeded](#)
Location
East US

Resources Recommendations

Filter for any field... Type equals **all** X Location equals **all** X Add filter

Showing 1 to 3 of 3 records. ☐ Show hidden types ⓘ No grouping List view

Name ↑↓	Type ↑↓	Location ↑↓	
prod-ci-cd-adf-020	Data factory (V2)	East US	...
prod-ci-cd-kv-020	Key vault	East US	...
prodciiddl	Storage account	East US	...

Step 3 - Create release pipeline option 1

All pipelines > ADF CD Option 1 Release Pi...

Save Create release ...

Pipeline Tasks Variables Retention Options History

Artifacts | + Add

- _dev-ci-cd-adf-020
- _dev-ci-cd-adf-020_main
- Schedule not set

Stages | + Add

- Test 1 job, 3 tasks
- Prod 1 job, 3 tasks

Step 4 – Create build pipeline option 2 – 1 / 2

← ADF CI Option 2 Build Pipeline

[Variables](#)[Run](#)

main dev-ci-cd-adf-020 / build/adf-ci-option-2-build-pipeline.yml

```
1 # File Name: adf-ci-option-2-build-pipeline.yml
2 # Purpose: Build pipeline for the development Azure Data Factory
3 # ..... Automatically triggered on completion of a pull request and
4 # ..... Validates the Data Factory resources
5 # ..... Generates the ARM template and publishes for consumption via
6 # Build Pipeline Name: ADF CI Option 2 Build Pipeline
7
8 trigger:
9 - main
10
11 pool:
12 - vmImage: ubuntu-latest
13
14 variables:
15 - subscriptionId: '9b9a9691-6d3d-43d5-84a2-bddc02aa3789' # Use your sub
16 - resourceGroup: 'dev-ci-cd-rg' # Use the reso
17 - dataFactory: 'dev-ci-cd-adf-020' # Use your dev
18 - PackageFolder: 'build' # Use the GIT
19 - adfRootFolder: '' # Use the GIT
20
21 steps:
22
23 # Installs Node
24 Settings
25 - task: NodeTool@0
26   inputs:
27     - versionSpec: '14.x'
28     - displayName: 'Install Node.js'
29
30 # Installs the npm packages saved in your package.json file in the build
31 Settings
32 - task: Npm@1
33   inputs:
34     - command: 'install'
35     - workingDir: '$(Build.Repository.LocalPath)/$(packageFolder)' #replac
36     - verbose: true
37     - displayName: 'Install npm packages'
38
39 # Validates all of the Data Factory resources in the repository. You'll
40 Settings
41 - task: Npm@1
42   inputs:
43     - command: 'custom'
44     - workingDir: '$(Build.Repository.LocalPath)/build' #replace with the
45     - customCommand: 'run build validate $(Build.Repository.LocalPath)/$(a
46     - displayName: 'Validate Data Factory Resources'
47
48 # Generate the ARM template into the destination folder, which is the sa
49 Settings
50 - task: Npm@1
51   inputs:
52     - command: 'custom'
53     - workingDir: '$(Build.Repository.LocalPath)/build' #replace with the
54     - customCommand: 'run build export $(Build.Repository.LocalPath)/$(adf
55     - displayName: 'Generate ARM template'
56
57 # Publish the artifact to be used as a source for a release pipeline.
58 Settings
```

Tasks



- .NET Core**
Build, test, package, or publish a dotnet applicatio...
- Android signing**
Sign and align Android APK files
- Ant**
Build with Apache Ant
- App Center distribute**
Distribute app builds to testers and users via Visu...
- App Center test**
Test app packages with Visual Studio App Center
- Archive files**
Compress files into .7z, .tar.gz, or .zip
- ARM template deployment**
Deploy an Azure Resource Manager (ARM) templ...
- Azure App Service deploy**
Deploy to Azure App Service a web, mobile, or AP...
- Azure App Service manage**
Start, stop, restart, slot swap, slot delete, install sit...
- Azure App Service Settings**
Update/Add App settings an Azure Web App for ...
- Azure CLI**
Run Azure CLI commands against an Azure subscri...
- Azure Cloud Service deployment**
Deploy an Azure Cloud Service
- Azure Database for MySQL deployment**
Run your scripts and make changes to your Azure...
- Azure file copy**
Copy files to Azure Blob Storage or virtual machin...
- Azure Function on Kubernetes**
Deploy Azure function to Kubernetes cluster.
- Azure Functions**
Update a function app with .NET, Python, JavaScri...
- Azure Functions for container**
Update a function app with a Docker container

Step 5 – Create release pipeline option 2

