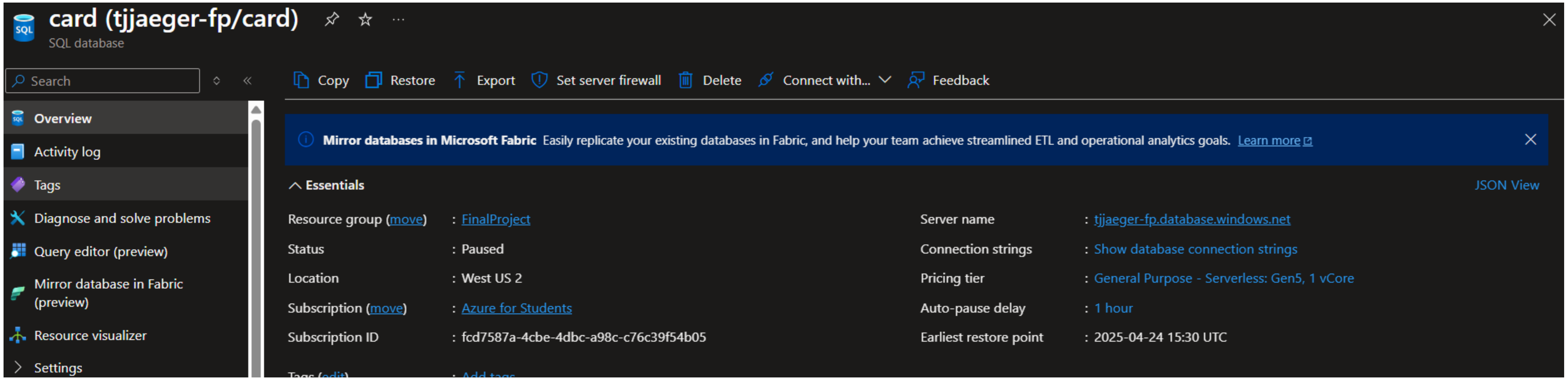
By: Tommy Jaeger

# Parking ETL Pipeline

## Creating SQLDatabase, Blob Storage Account w/ Container

# SQL Database:



**Mirror databases in Microsoft Fabric** Easily replicate your existing databases in Fabric, and help your team achieve streamlined ETL and operational analytics goals. Learn more

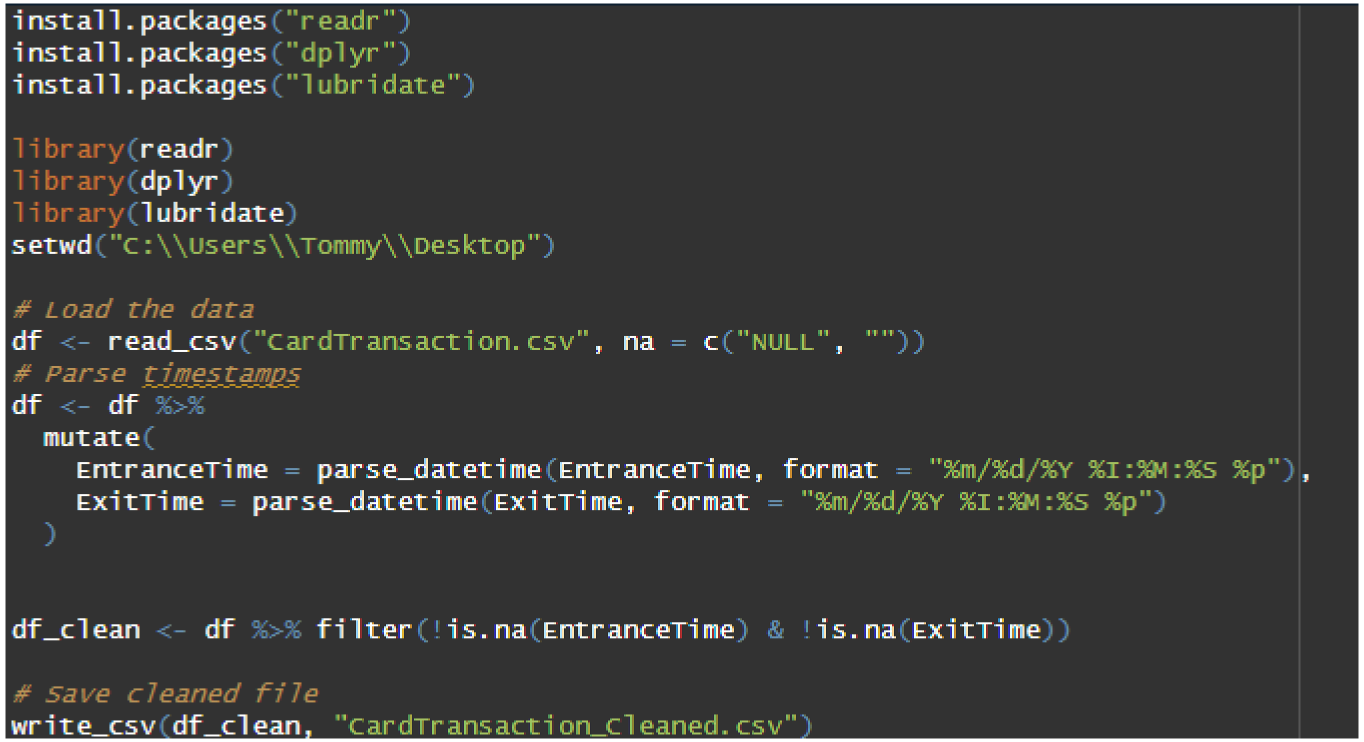
Essentials — JSON View

| | | | |
|---|---|---|---|
| Resource group (move) | : FinalProject | Server name | : tjjaeger-fp.database.windows.net |
| Status | : Paused | Connection strings | : Show database connection strings |
| Location | : West US 2 | Pricing tier | : General Purpose - Serverless: Gen5, 1 vCore |
| Subscription (move) | : Azure for Students | Auto-pause delay | : 1 hour |
| Subscription ID | : fcd7587a-4cbe-4dbc-a98c-c76c39f54b05 | Earliest restore point | : 2025-04-24 15:30 UTC |
| Tags (edit) | : Add tags | | |

## R-Studio- Cleaning (Bad Values, NA/”NULL’, Times in same format -> Upload to container

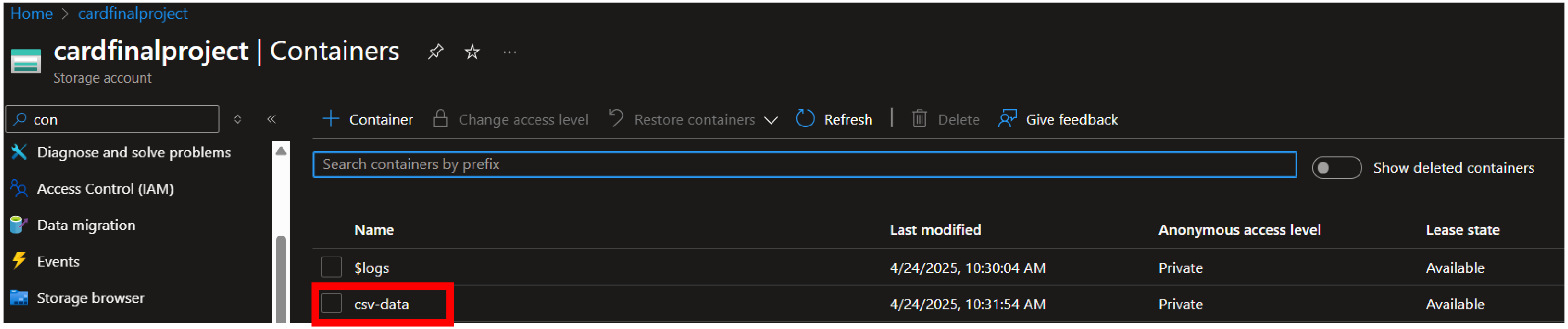
# Cleaning Using R-Studio:

- Load in the Unclean Data in R
- **Parse** through the time stamps:
    - Uses **Mutate** to transform *EntranceTime* and ExitTime
    - **Converts** to proper datetime: *month/day/year hour:minute:second AM/PM*
    - **Removes** rows with missing timestamps
    - **Save** as CardTransaction_Cleaned.csv



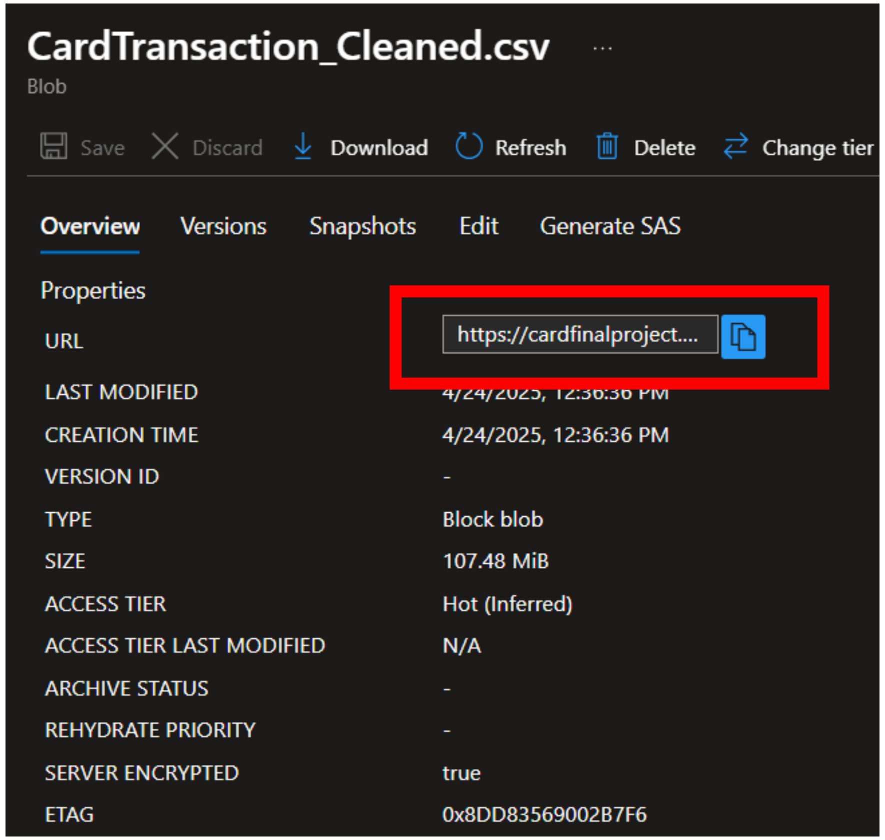
```r
install.packages("readr")
install.packages("dplyr")
install.packages("lubridate")

library(readr)
library(dplyr)
library(lubridate)
setwd("C:\\Users\\Tommy\\Desktop")

# Load the data
df <- read_csv("CardTransaction.csv", na = c("NULL", ""))
# Parse timestamps
df <- df %>%
  mutate(
    EntranceTime = parse_datetime(EntranceTime, format = "%m/%d/%Y %I:%M:%S %p"),
    ExitTime = parse_datetime(ExitTime, format = "%m/%d/%Y %I:%M:%S %p")
  )


df_clean <- df %>% filter(!is.na(EntranceTime) & !is.na(ExitTime))

# Save cleaned file
write_csv(df_clean, "CardTransaction_Cleaned.csv")
```

# Creating Storage Account with CSV Data:



| Name | Last modified | Anonymous access level | Lease state |
|---|---|---|---|
| $logs | 4/24/2025, 10:30:04 AM | Private | Available |
| csv-data | 4/24/2025, 10:31:54 AM | Private | Available |

## Csv-data > CardTransaction_Cleaned > copy URL for import to ADF



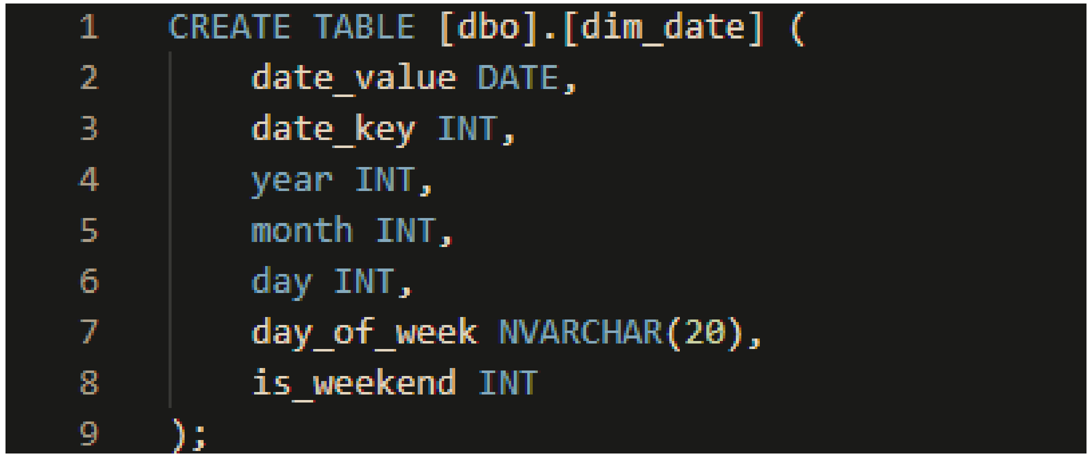
CardTransaction_Cleaned.csv

Blob

Properties

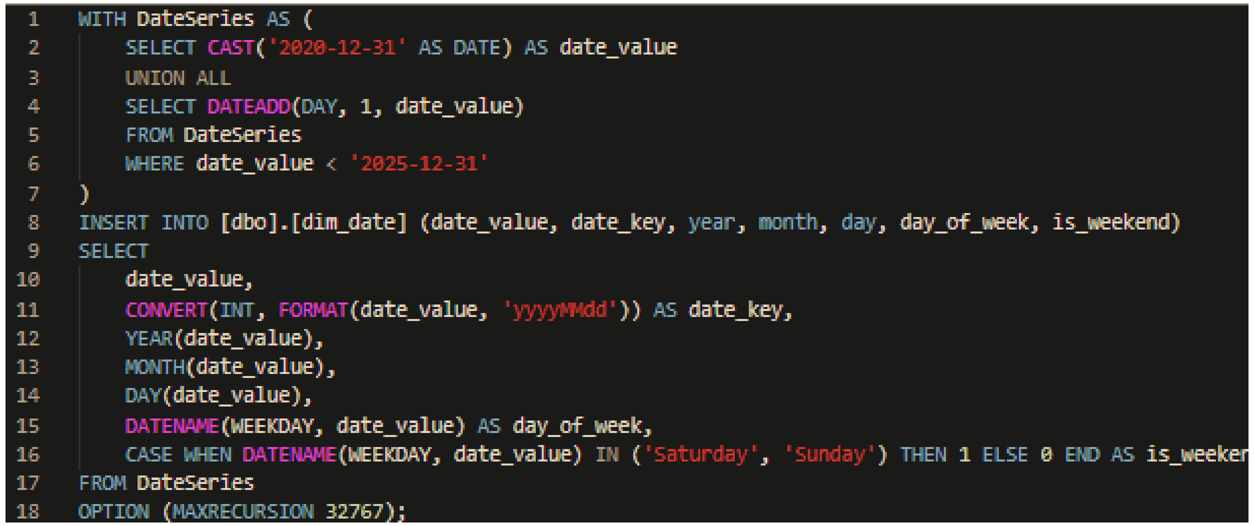
| | |
|---|---|
| URL | https://cardfinalproject.... |
| LAST MODIFIED | 4/24/2025, 12:36:36 PM |
| CREATION TIME | 4/24/2025, 12:36:36 PM |
| VERSION ID | - |
| TYPE | Block blob |
| SIZE | 107.48 MiB |
| ACCESS TIER | Hot (Inferred) |
| ACCESS TIER LAST MODIFIED | N/A |
| ARCHIVE STATUS | - |
| REHYDRATE PRIORITY | - |
| SERVER ENCRYPTED | true |
| ETAG | 0x8DD83569002B7F6 |

- Created the [dbo].[dim_date] table



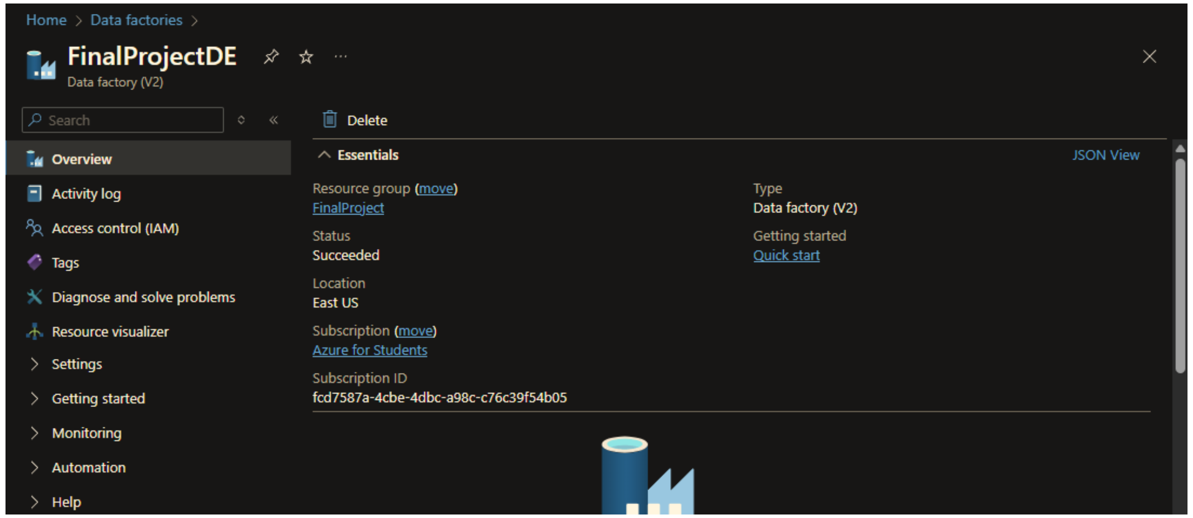
```sql
CREATE TABLE [dbo].[dim_date] (
    date_value DATE,
    date_key INT,
    year INT,
    month INT,
    day INT,
    day_of_week NVARCHAR(20),
    is_weekend INT
);
```

- Populated it with data that matched the years of the parking data



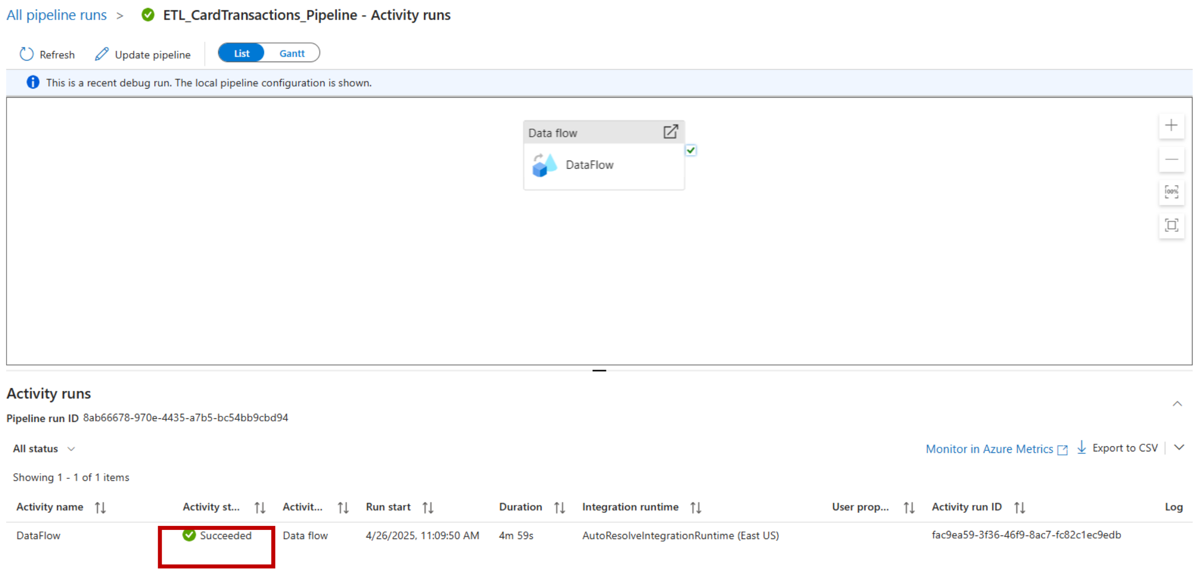
```sql
WITH DateSeries AS (
    SELECT CAST('2020-12-31' AS DATE) AS date_value
    UNION ALL
    SELECT DATEADD(DAY, 1, date_value)
    FROM DateSeries
    WHERE date_value < '2025-12-31'
)
INSERT INTO [dbo].[dim_date] (date_value, date_key, year, month, day, day_of_week, is_weekend)
SELECT
    date_value,
    CONVERT(INT, FORMAT(date_value, 'yyyyMMdd')) AS date_key,
    YEAR(date_value),
    MONTH(date_value),
    DAY(date_value),
    DATENAME(WEEKDAY, date_value) AS day_of_week,
    CASE WHEN DATENAME(WEEKDAY, date_value) IN ('Saturday', 'Sunday') THEN 1 ELSE 0 END AS is_weeker
FROM DateSeries
OPTION (MAXRECURSION 32767);
```

# Loading and Organizing ADF:



FinalProjectDE
Data factory (V2)

| | |
|---|---|
| Resource group (move) | FinalProject |
| Status | Succeeded |
| Location | East US |
| Subscription (move) | Azure for Students |
| Subscription ID | fcd7587a-4cbe-4dbc-a98c-c76c39f54b05 |
| Type | Data factory (V2) |
| Getting started | Quick start |



ETL_CardTransactions_Pipeline - Activity runs

This is a recent debug run. The local pipeline configuration is shown.

Activity runs

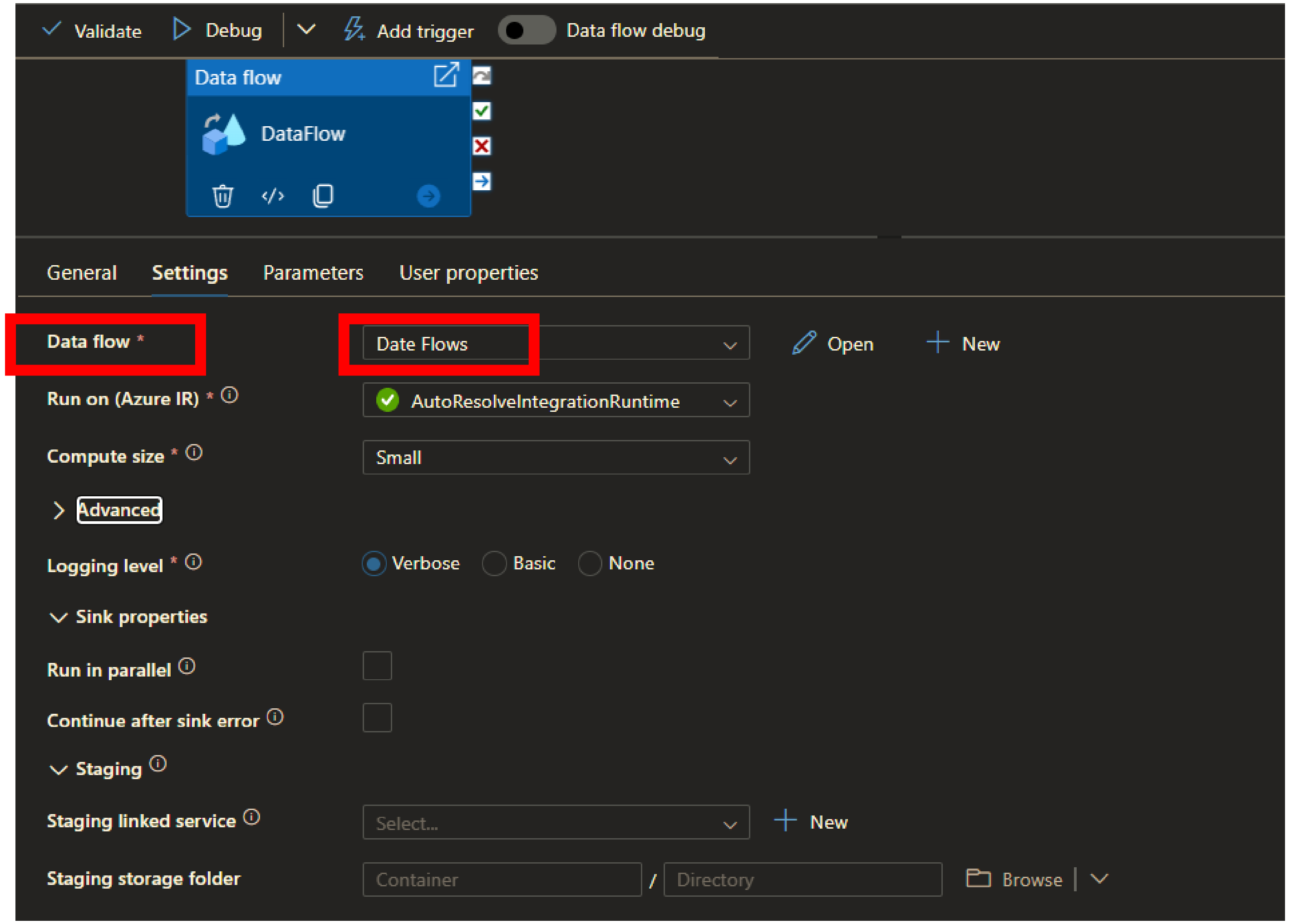
Pipeline run ID 8ab66678-970e-4435-a7b5-bc54bb9cbd94

Showing 1 - 1 of 1 items

| Activity name | Activity st... | Activit... | Run start | Duration | Integration runtime | User prop... | Activity run ID | Log |
|---|---|---|---|---|---|---|---|---|
| DataFlow | ✓ Succeeded | Data flow | 4/26/2025, 11:09:50 AM | 4m 59s | AutoResolveIntegrationRuntime (East US) | | fac9ea59-3f36-46f9-8ac7-fc82c1ec9edb | |

## Pipeline:

- Takes the data from DataFlow called Date Flows which I created the data flow steps
- Run on Azure IR
- Verbose Logging
- Next step is to set up the data flow



Data flow * : Date Flows

Run on (Azure IR) * : AutoResolveIntegrationRuntime

Compute size * : Small

Logging level * : [✓ Verbose] Basic None

Sink properties

Run in parallel : ☐

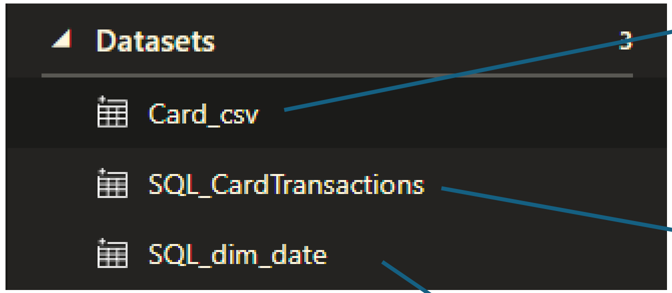
Continue after sink error : ☐

Staging

Staging linked service : Select...

Staging storage folder : Container / Directory

# Setting up DataSets: Transaction, Card, Dim_Date

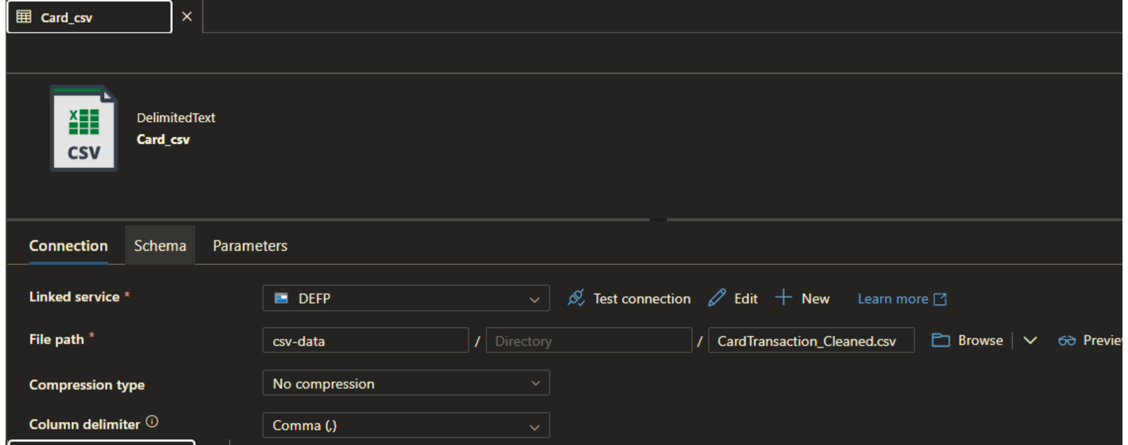


Datasets 3

Card_csv

SQL_CardTransactions

SQL_dim_date

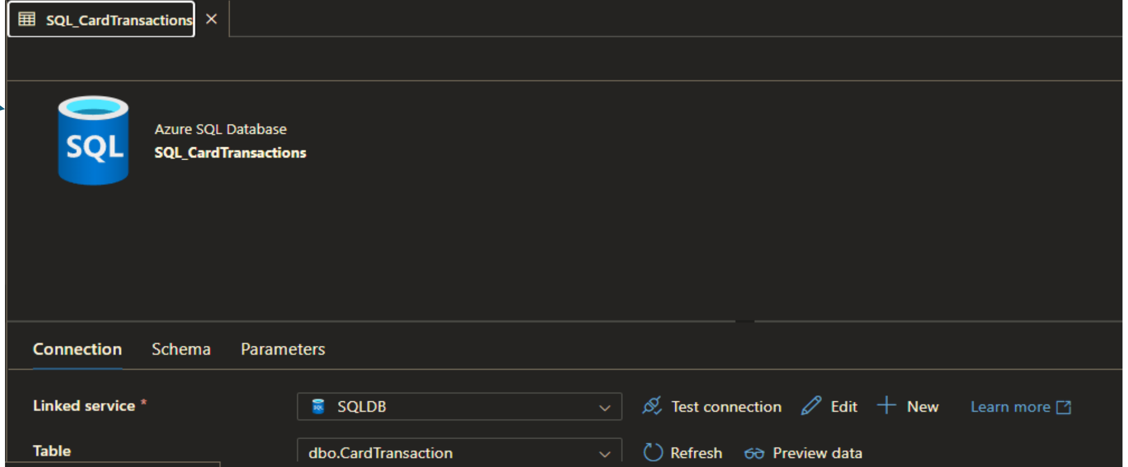


Card_csv

DelimitedText
Card_csv

Connection | Schema | Parameters

| | |
|---|---|
| Linked service * | DEFP |
| File path * | csv-data / Directory / CardTransaction_Cleaned.csv |
| Compression type | No compression |
| Column delimiter | Comma (,) |

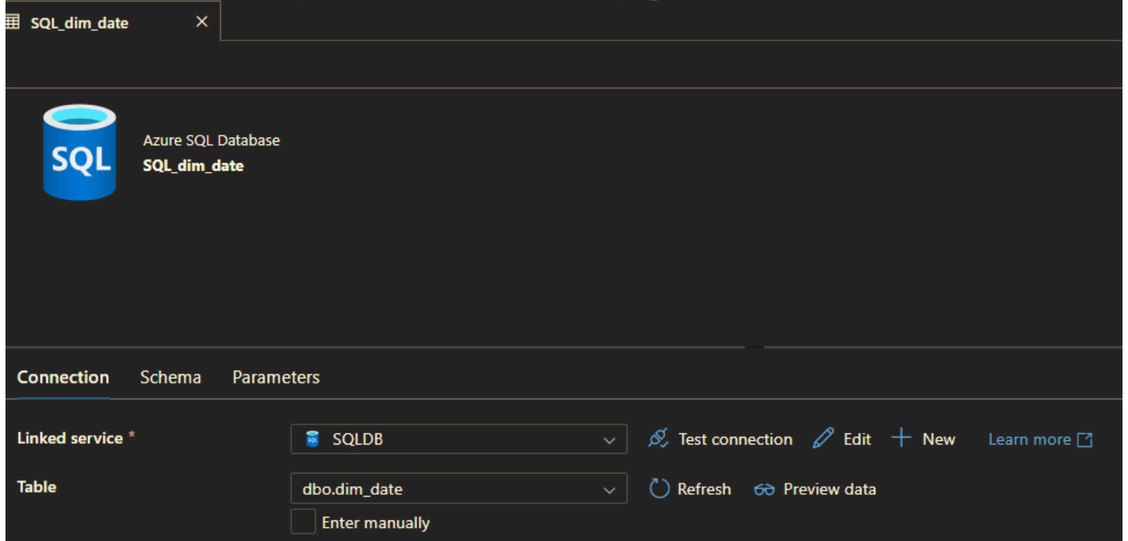


SQL_CardTransactions

Azure SQL Database
SQL_CardTransactions

Connection | Schema | Parameters

| | |
|---|---|
| Linked service * | SQLDB |
| Table | dbo.CardTransaction |



SQL_dim_date

Azure SQL Database
SQL_dim_date

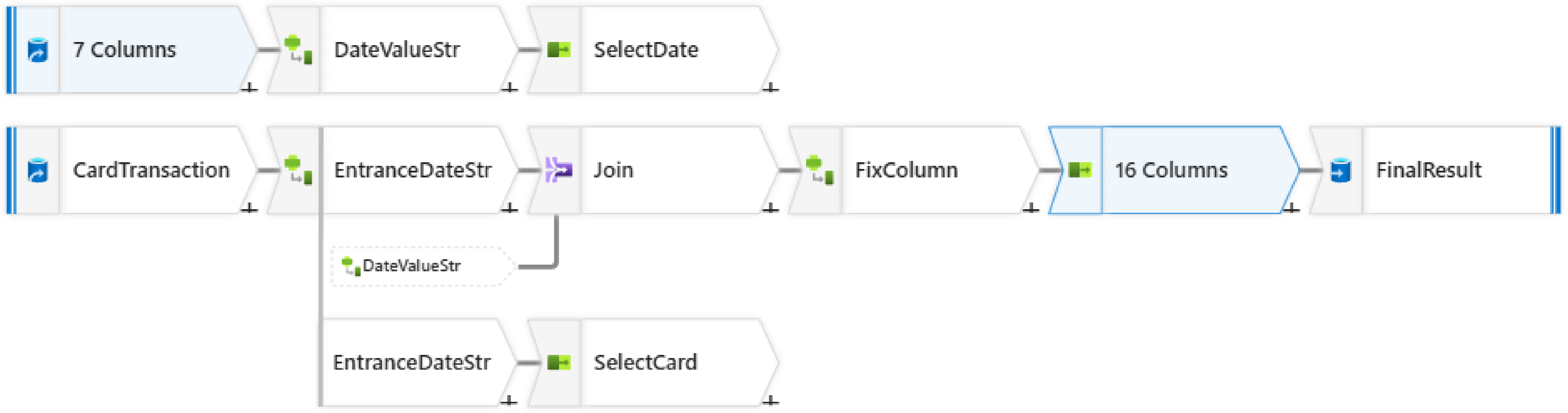
Connection | Schema | Parameters

| | |
|---|---|
| Linked service * | SQLDB |
| Table | dbo.dim_date |

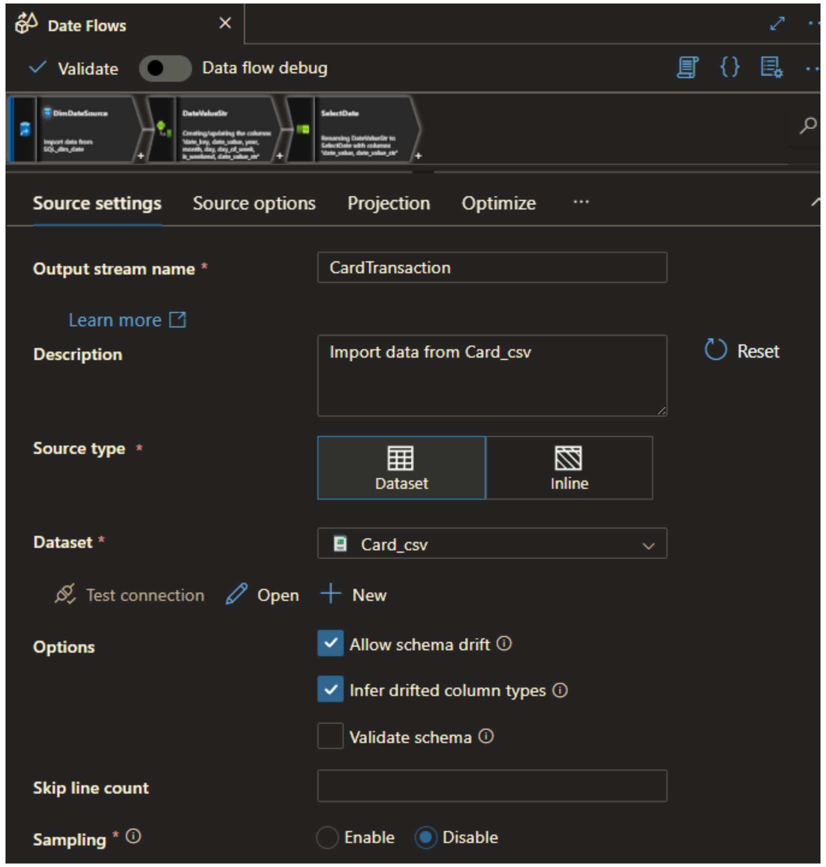
Enter manually

- These will be the DataSets that will be loaded into the data flow

## Data Flow Overview:



# Importing The Two DataSets:


Date Flows
Validate
Data flow debug
DimDateSource
Import data from SQL_dim_date
DateValueStr
Creating/updating the columns 'date_key, date_value, year, month, day, day_of_week, is_weekend, date_value_str'
SelectDate
Renaming DateValueStr to SelectDate with columns 'date_value, date_value_str'
Source settings
Source options
Projection
Optimize
Learn more
Reset
Test connection
Open
New

Output stream name *: CardTransaction

Description: Import data from Card_csv

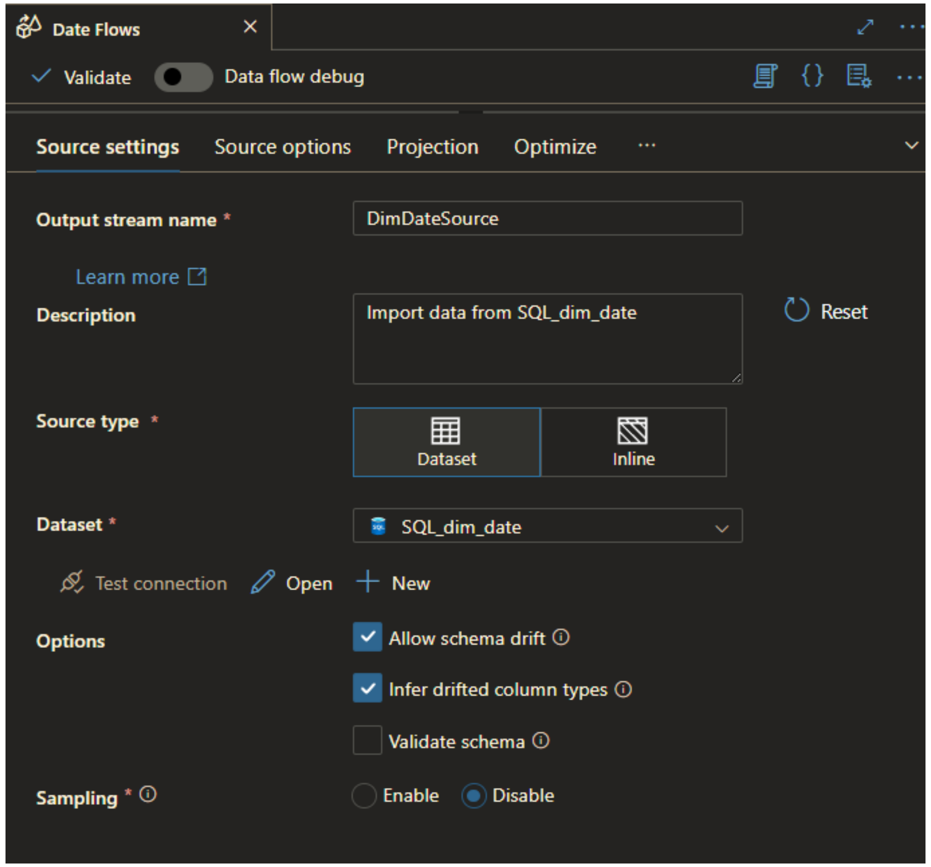
Source type *: [✓ Dataset] Inline

Dataset *: Card_csv

Options: [✓] Allow schema drift; [✓] Infer drifted column types; [ ] Validate schema

Skip line count:

Sampling *: Enable [✓ Disable]



Output stream name *: DimDateSource

Description: Import data from SQL_dim_date

Source type *: [✓ Dataset] Inline

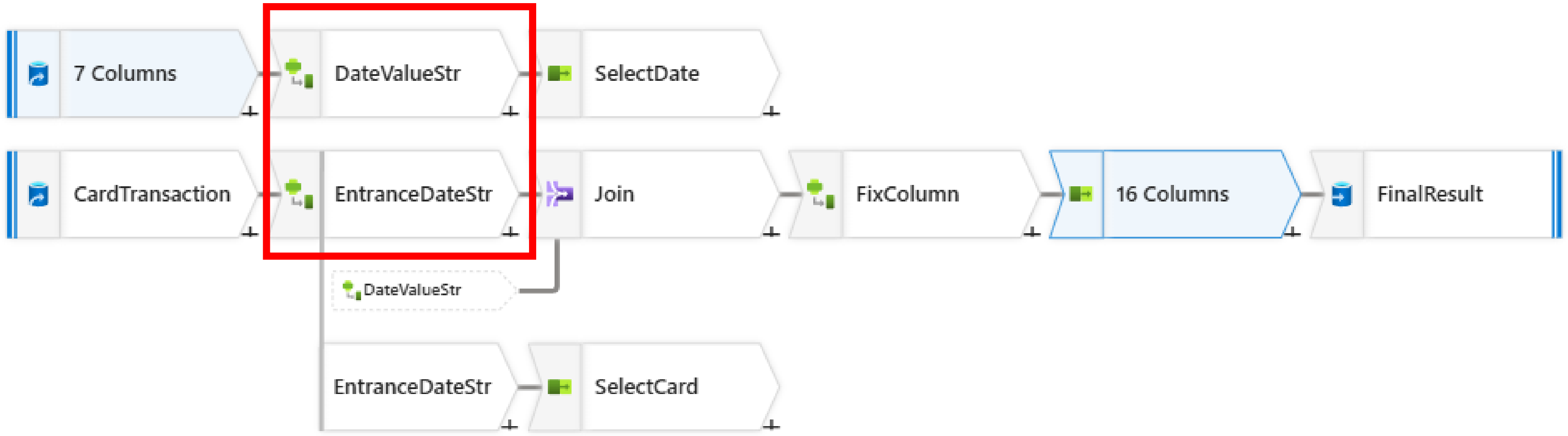
Dataset *: SQL_dim_date

Options: [✓] Allow schema drift; [✓] Infer drifted column types; [ ] Validate schema

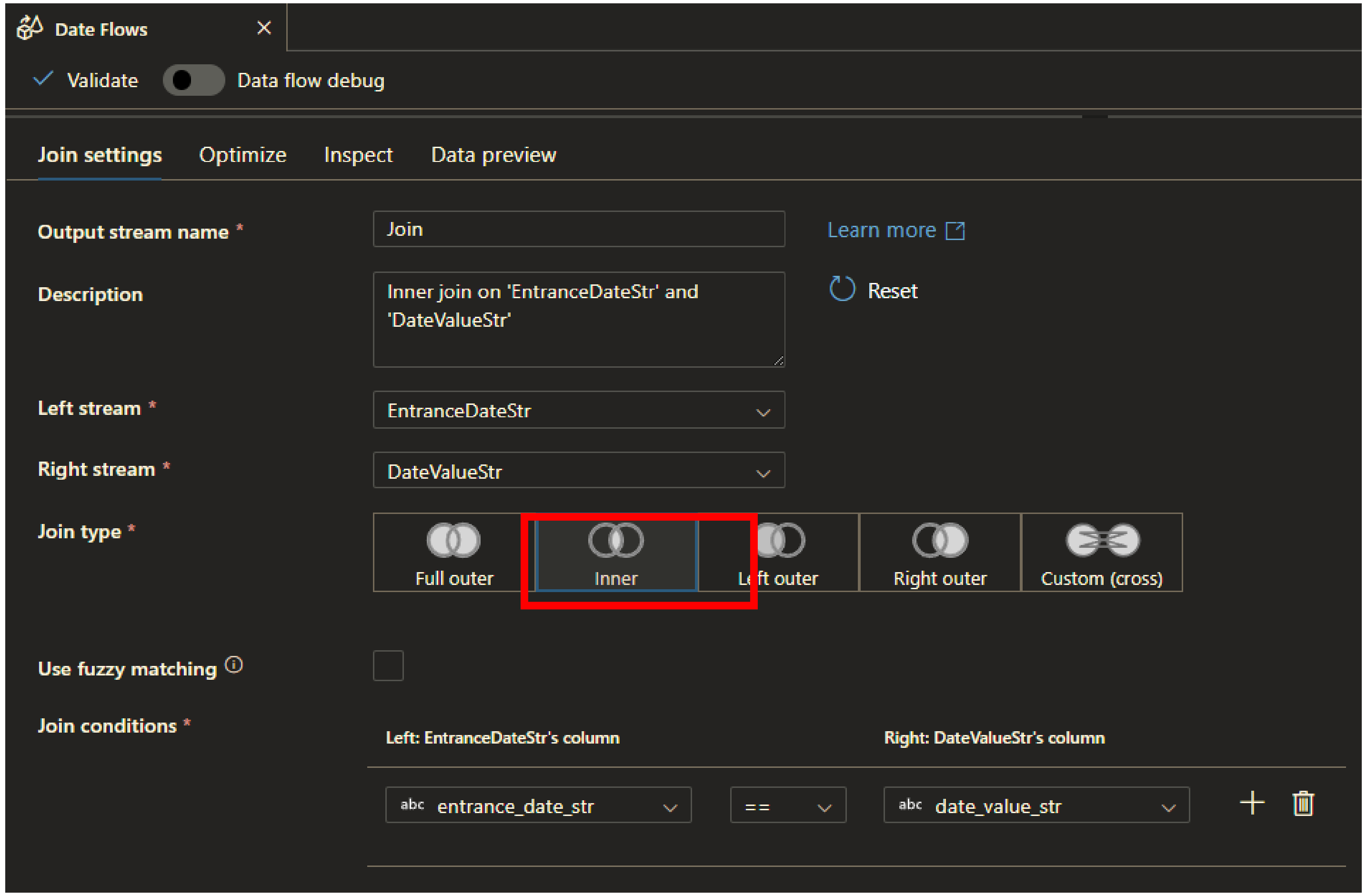
Sampling *: Enable [✓ Disable]

# Joining Dim_Date and CardTransactions:

Coverts Dates to Strings for the Join, could not join them normally.



## Join Table:



Output stream name *: Join

Description: Inner join on 'EntranceDateStr' and 'DateValueStr'

Left stream *: EntranceDateStr

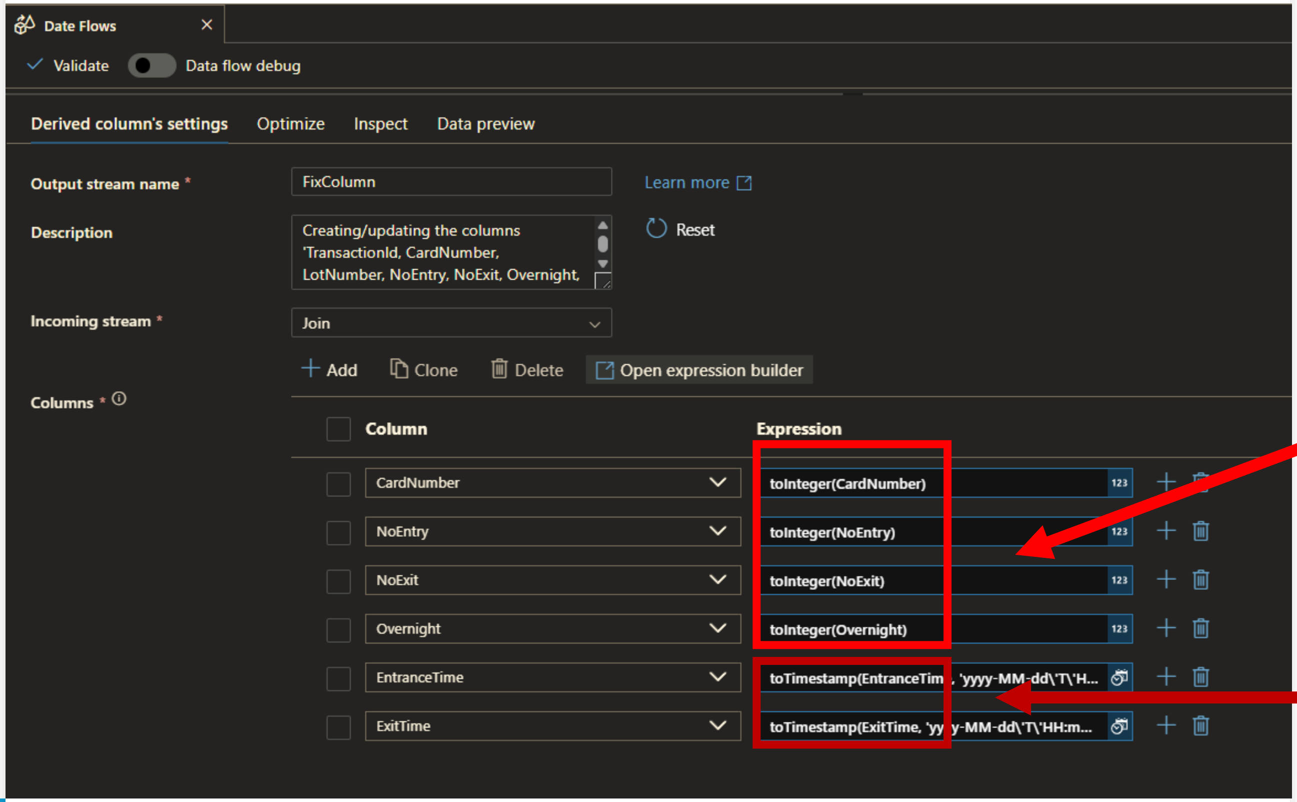
Right stream *: DateValueStr

Join type *: Full outer [✓ Inner] Left outer Right outer Custom (cross)

Use fuzzy matching: ☐

Join conditions *

| Left: EntranceDateStr's column | | Right: DateValueStr's column |
|---|---|---|
| abc entrance_date_str | == | abc date_value_str |



Date Flows

Validate | Data flow debug

Derived column's settings | Optimize | Inspect | Data preview

Output stream name *: FixColumn

Description: Creating/updating the columns 'TransactionId, CardNumber, LotNumber, NoEntry, NoExit, Overnight,

Incoming stream *: Join

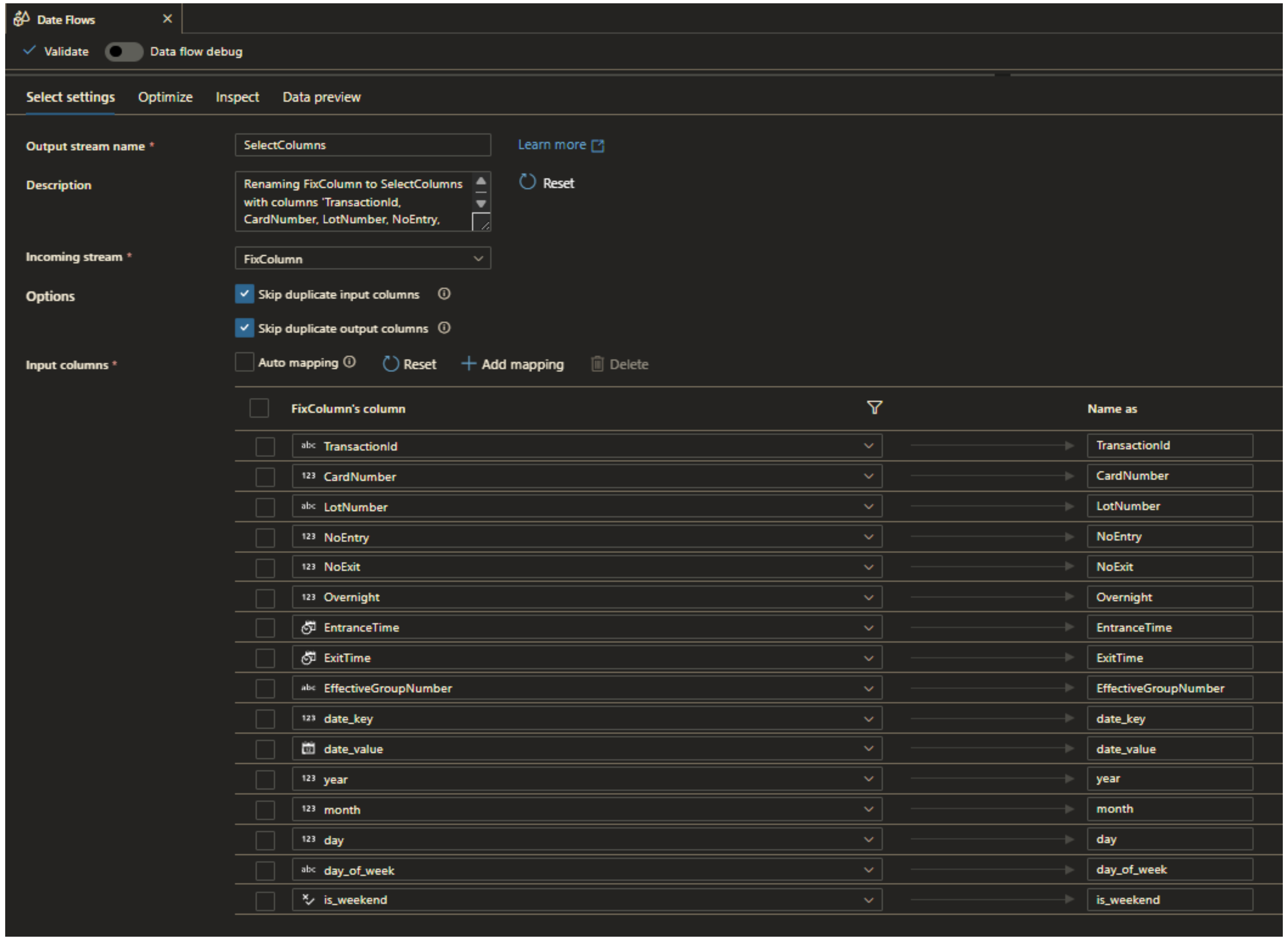
Columns *

| Column | Expression |
|---|---|
| CardNumber | toInteger(CardNumber) |
| NoEntry | toInteger(NoEntry) |
| NoExit | toInteger(NoExit) |
| Overnight | toInteger(Overnight) |
| EntranceTime | toTimestamp(EntranceTim , 'yyyy-MM-dd\'T\'H... |
| ExitTime | toTimestamp(ExitTime, 'yy y-MM-dd\'T\'HH:m... |

# Fixing the Columns back to INTs, and Times back to TimeStamps

# Select Columns for Final Output:

- Number of Columns: 16
- Do not duplicate columns



Output stream name *: SelectColumns

Description: Renaming FixColumn to SelectColumns with columns 'TransactionId, CardNumber, LotNumber, NoEntry,

Incoming stream *: FixColumn

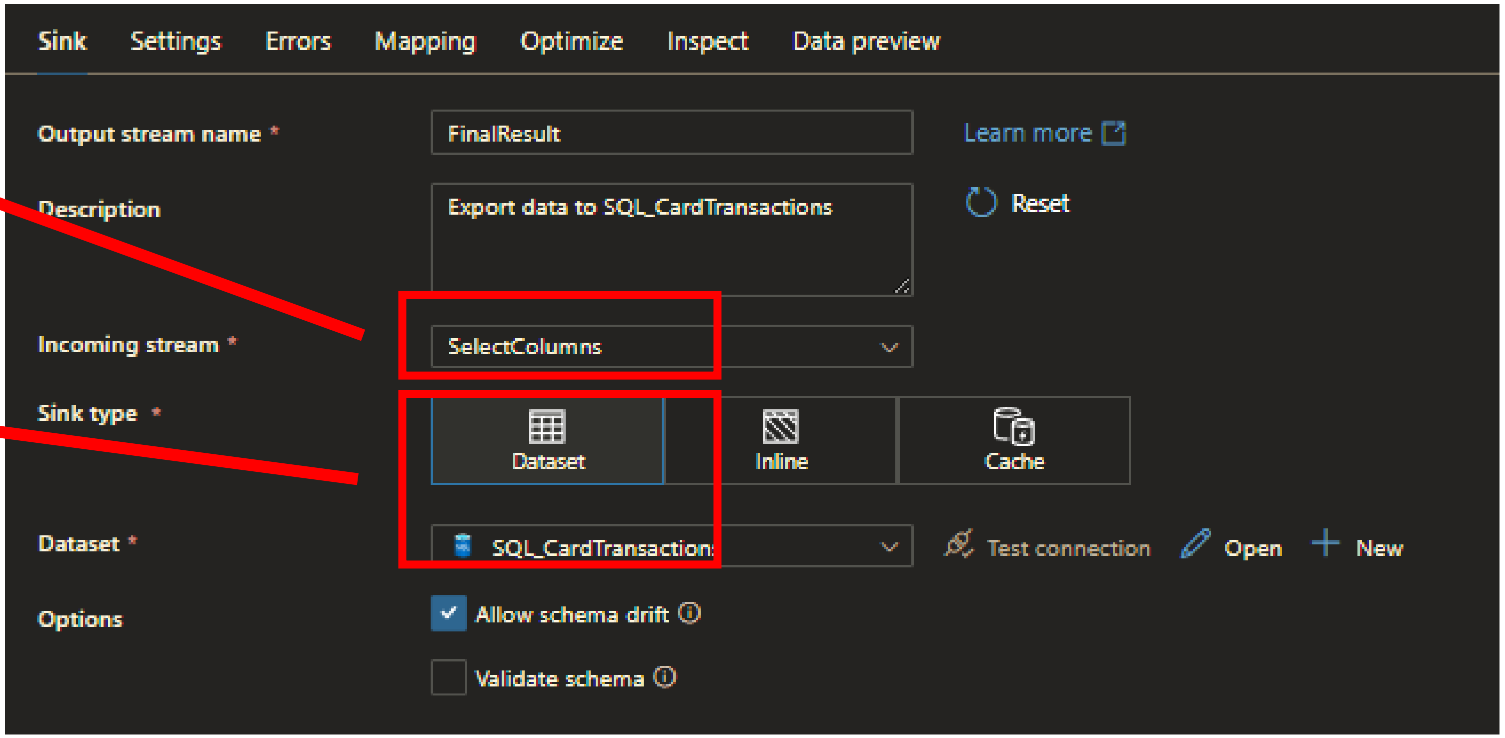
Options: [✓] Skip duplicate input columns; [✓] Skip duplicate output columns

Input columns *: [ ] Auto mapping

| FixColumn's column | Name as |
|---|---|
| abc TransactionId | TransactionId |
| 123 CardNumber | CardNumber |
| abc LotNumber | LotNumber |
| 123 NoEntry | NoEntry |
| 123 NoExit | NoExit |
| 123 Overnight | Overnight |
| EntranceTime | EntranceTime |
| ExitTime | ExitTime |
| abc EffectiveGroupNumber | EffectiveGroupNumber |
| 123 date_key | date_key |
| date_value | date_value |
| 123 year | year |
| 123 month | month |
| 123 day | day |
| abc day_of_week | day_of_week |
| is_weekend | is_weekend |

# Link The Data Flow to the Sink for Export:

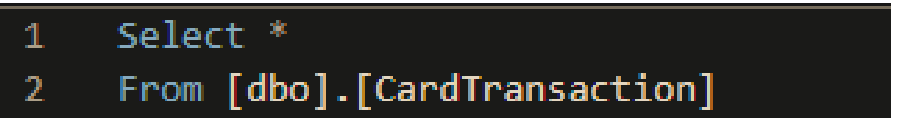
- Incoming Stream is from the SelectColumns step
- SinkType: Link this to the DataSet: SQL_CardTransactions DB to link the MS Azure
- Went back to Pipeline and triggered it



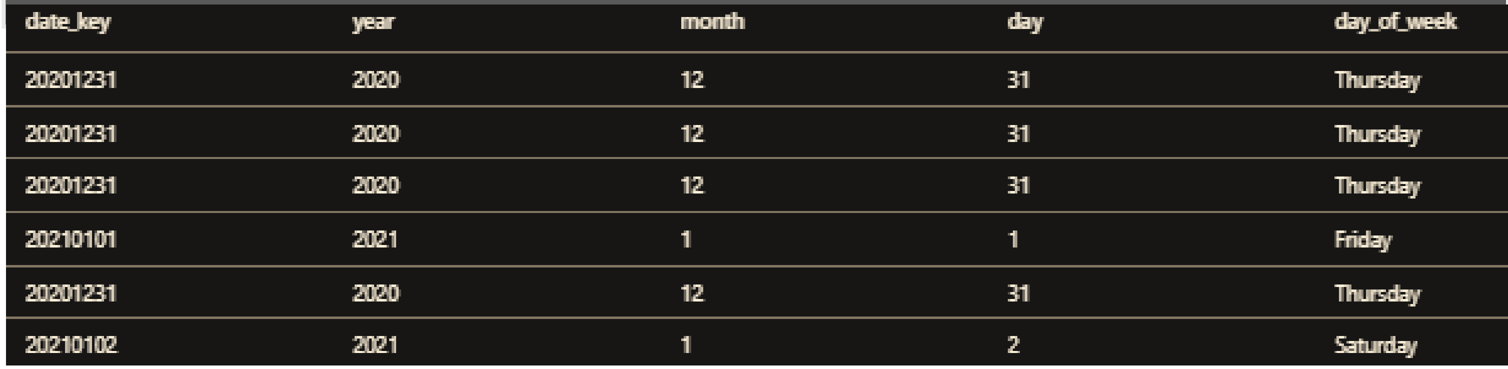
| | |
|---|---|
| Output stream name * | FinalResult |
| Description | Export data to SQL_CardTransactions |
| Incoming stream * | SelectColumns |
| Sink type * | Dataset |
| Dataset * | SQL_CardTransaction: |
| Options | ✓ Allow schema drift |
| | Validate schema |

## Check if Pipeline Succeeded in Azure SQL Datbase:

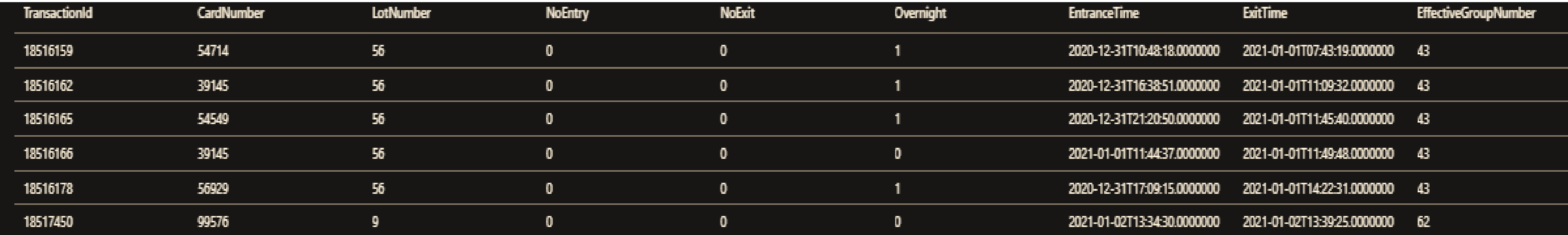
- These two screenshots are one table
- In the Azure Power Query Editor we ran the SQL query below and it resulted in one table joining these two tables together
- The ADF Pipeline was a success



```sql
Select *
From [dbo].[CardTransaction]
```



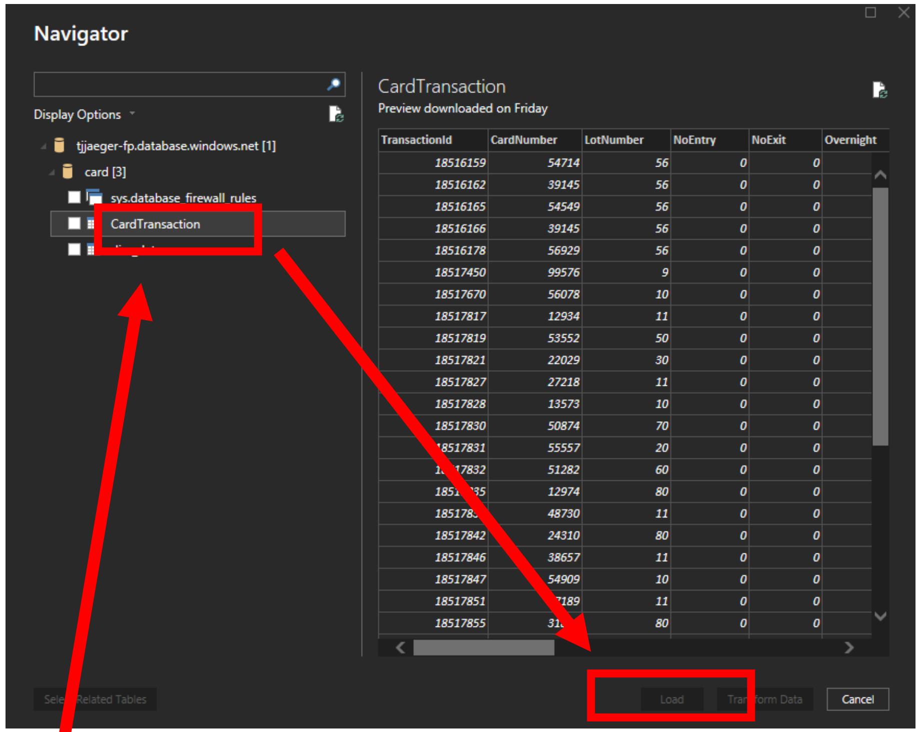
| date_key | year | month | day | day_of_week |
|---|---|---|---|---|
| 20201231 | 2020 | 12 | 31 | Thursday |
| 20201231 | 2020 | 12 | 31 | Thursday |
| 20201231 | 2020 | 12 | 31 | Thursday |
| 20210101 | 2021 | 1 | 1 | Friday |
| 20201231 | 2020 | 12 | 31 | Thursday |
| 20210102 | 2021 | 1 | 2 | Saturday |



| TransactionId | CardNumber | LotNumber | NoEntry | NoExit | Overnight | EntranceTime | ExitTime | EffectiveGroupNumber |
|---|---|---|---|---|---|---|---|---|
| 18516159 | 54714 | 56 | 0 | 0 | 1 | 2020-12-31T10:48:18.0000000 | 2021-01-01T07:43:19.0000000 | 43 |
| 18516162 | 39145 | 56 | 0 | 0 | 1 | 2020-12-31T16:38:51.0000000 | 2021-01-01T11:09:32.0000000 | 43 |
| 18516165 | 54549 | 56 | 0 | 0 | 1 | 2020-12-31T21:20:50.0000000 | 2021-01-01T11:45:40.0000000 | 43 |
| 18516166 | 39145 | 56 | 0 | 0 | 0 | 2021-01-01T11:44:37.0000000 | 2021-01-01T11:49:48.0000000 | 43 |
| 18516178 | 56929 | 56 | 0 | 0 | 1 | 2020-12-31T17:09:15.0000000 | 2021-01-01T14:22:31.0000000 | 43 |
| 18517450 | 99576 | 9 | 0 | 0 | 0 | 2021-01-02T13:34:30.0000000 | 2021-01-02T13:39:25.0000000 | 62 |

# Connect SQL Database to PowerBI

- Connect to Azure SQL Database
- Azure > SQLDB > Copy Server Name
- Paste in PowerBI Server and Connect
- Select CardTransaction > Load
- Create the Report



Navigator

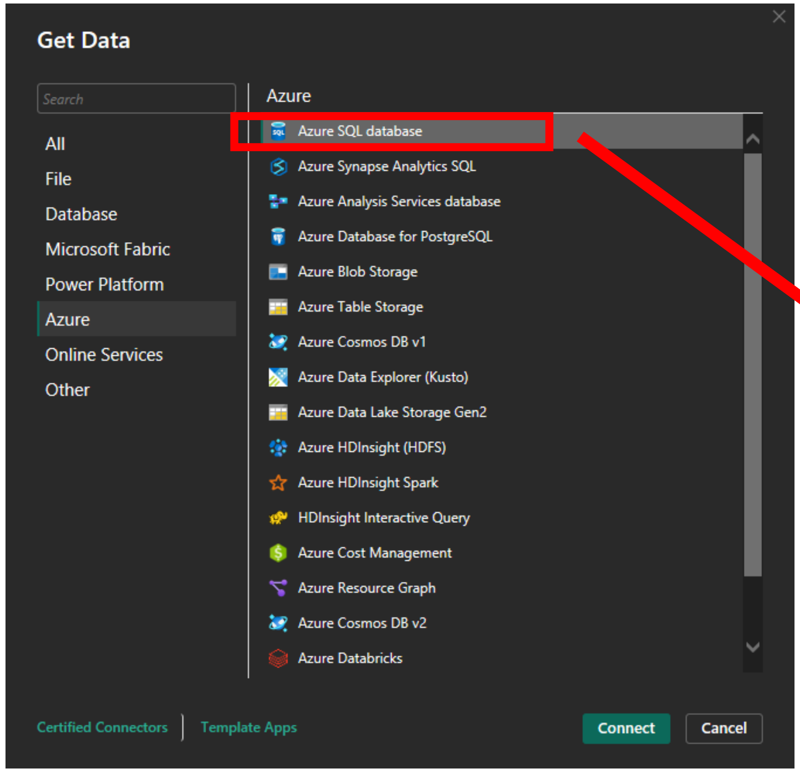
Display Options

- tjjaeger-fp.database.windows.net [1]
  - card [3]
    - sys.database_firewall_rules
    - CardTransaction

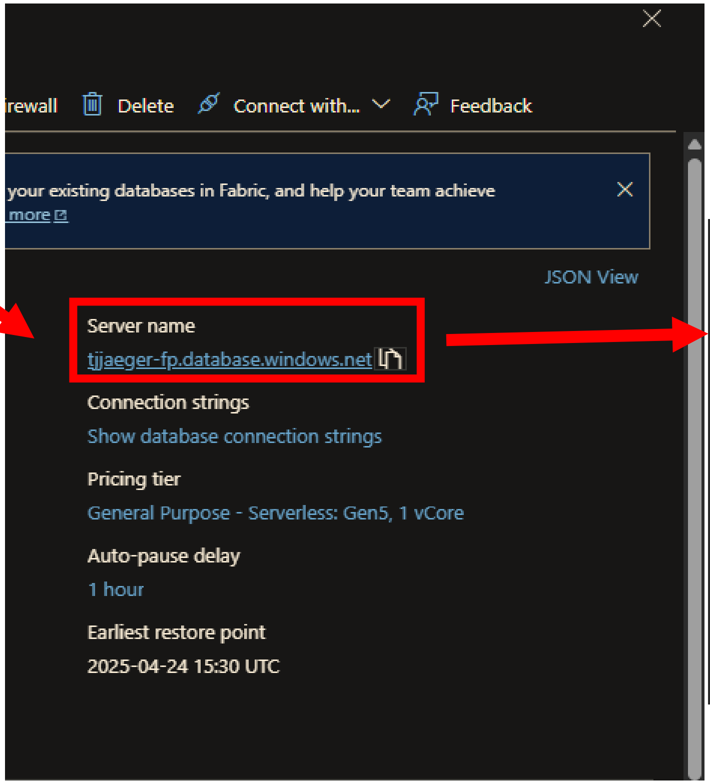
CardTransaction

Preview downloaded on Friday

| TransactionId | CardNumber | LotNumber | NoEntry | NoExit | Overnight |
|---|---|---|---|---|---|
| 18516159 | 54714 | 56 | 0 | 0 | 0 |
| 18516162 | 39145 | 56 | 0 | 0 | 0 |
| 18516165 | 54549 | 56 | 0 | 0 | 0 |
| 18516166 | 39145 | 56 | 0 | 0 | 0 |
| 18516178 | 56929 | 56 | 0 | 0 | 0 |
| 18517450 | 99576 | 9 | 0 | 0 | 0 |
| 18517670 | 56078 | 10 | 0 | 0 | 0 |
| 18517817 | 12934 | 11 | 0 | 0 | 0 |
| 18517819 | 53552 | 50 | 0 | 0 | 0 |
| 18517821 | 22029 | 30 | 0 | 0 | 0 |
| 18517827 | 27218 | 11 | 0 | 0 | 0 |
| 18517828 | 13573 | 10 | 0 | 0 | 0 |
| 18517830 | 50874 | 70 | 0 | 0 | 0 |
| 18517831 | 55557 | 20 | 0 | 0 | 0 |
| [illegible] | 51282 | 60 | 0 | 0 | 0 |
| [illegible] | 12974 | 80 | 0 | 0 | 0 |
| [illegible] | 48730 | 11 | 0 | 0 | 0 |
| 18517842 | 24310 | 80 | 0 | 0 | 0 |
| 18517846 | 38657 | 11 | 0 | 0 | 0 |
| 18517847 | 54909 | 10 | 0 | 0 | 0 |
| 18517851 | [illegible] | 11 | 0 | 0 | 0 |
| 18517855 | [illegible] | 80 | 0 | 0 | 0 |



Get Data



your existing databases in Fabric, and help your team achieve more

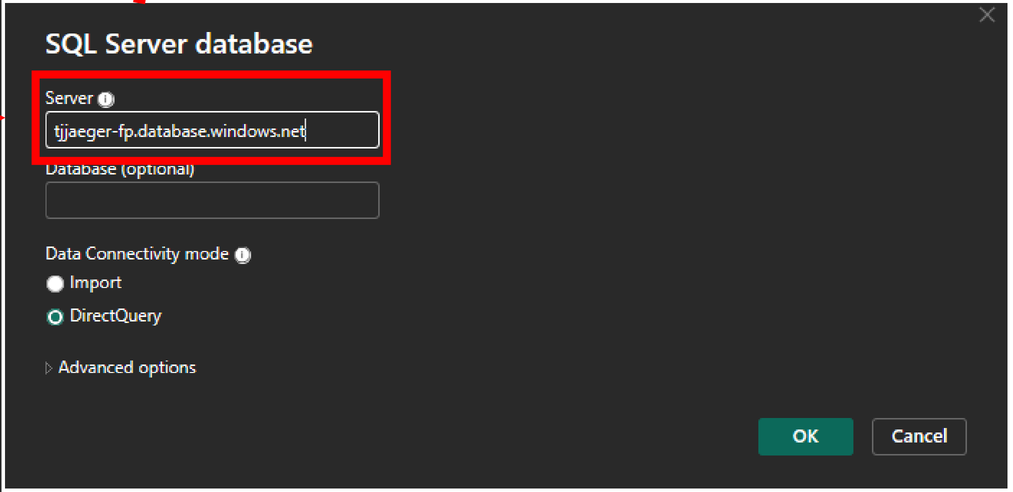
Server name: tjjaeger-fp.database.windows.net

Connection strings: Show database connection strings

Pricing tier: General Purpose - Serverless: Gen5, 1 vCore

Auto-pause delay: 1 hour

Earliest restore point: 2025-04-24 15:30 UTC



SQL Server database

Server: tjjaeger-fp.database.windows.net

Database (optional):

Data Connectivity mode: Import [✓ DirectQuery]

**Lot Number**

All

2020 | 2021 | 2022 | 2023 | 2024

## AM (Morning)

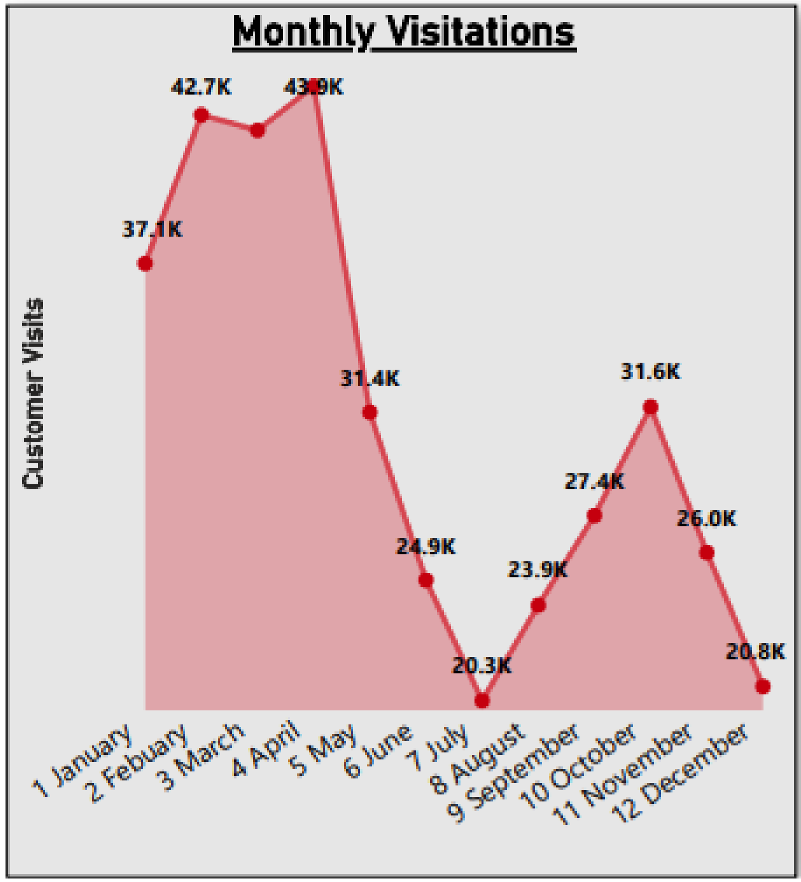
**187.5K**
Total Customers

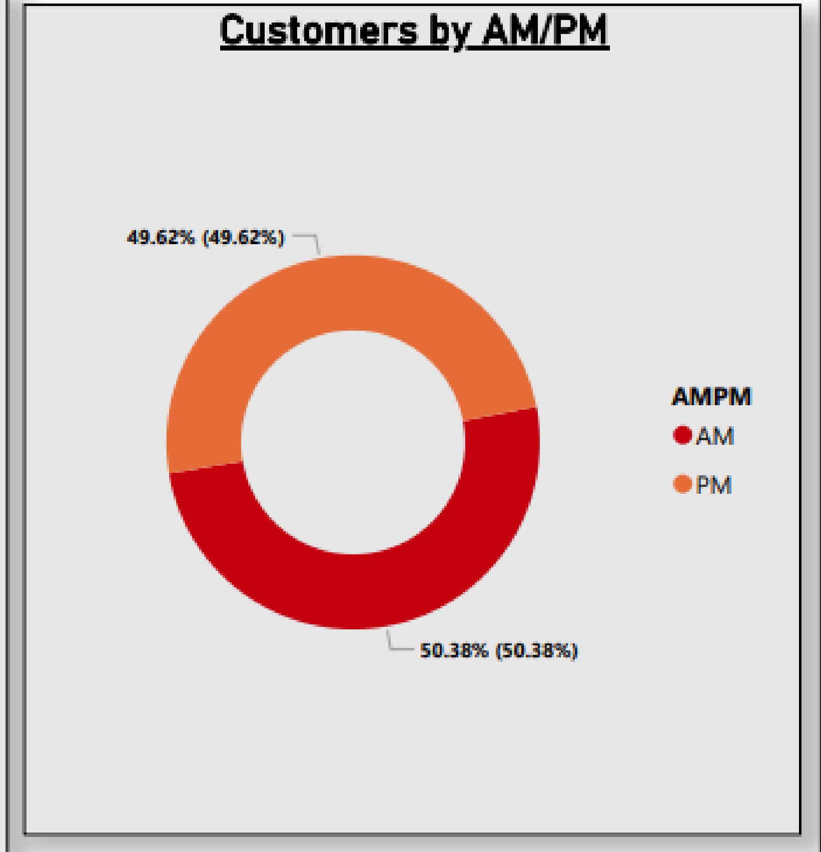
**429.46**
Avg Parking Time (Min)

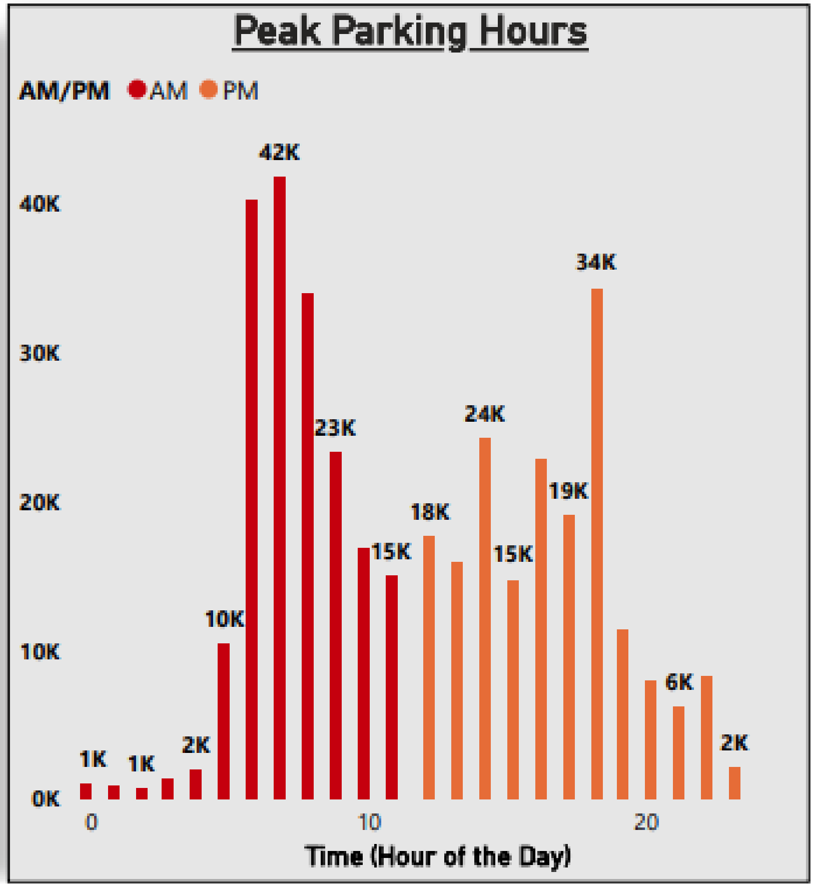
## PM (Evening)

**184.6K**
Total Customers

**445.96**
AvgParking Time (Min)







## Conclusions:

- Dip in the summer months from June to August
- 2021-2024 trends show more AM parking until 2024 where there were more PM parking
- Recently declining number of customers since 2021