A close-up of a person smiling

Description automatically generated

# Lab 05: Transferring large data volumes with Fabric Data Factory

### 

### Introduction

In this lab, you will ingest data from a Azure SQL Database to the Managed Lakehouse Sales table. The objective is to understand how to connect to this external database and how the Fabric Data Factory Pipeline can scale when having to transfer 10M, 50M and 100M of rows.

### Objectives

After completing this lab, you will be better able to:

1. Connect to an Azure SQL Database and read the tables
2. Set up and run the Fabric Data Factory pipeline to transfer data from Azure SQL tables to a table in the managed Lakehouse.

**Estimated time to complete this lab**

60 minutes

Contents

[Lab 05: Transferring large data volumes with Fabric Data Factory 1](#_Toc159609682)

[Introduction 1](#_Toc159609683)

[Objectives 1](#_Toc159609684)

[Task 1: Setup the connection to the Azure SQL Database 3](#_Toc159609685)

[Task 2: Use Data Pipeline Parameters to change the source table 10](#_Toc159609686)

[Task 3: Optimize Data Pipeline performance with parallelism and partitions 13](#_Toc159609687)

**Lab Prerequisites**

* Workspace: Fabric, Power Premium or Fabric trial
* Individual license: Power Pro or Premium Per User account

**Information provided by your training provider**

* Trial tenant (if applicable): login & password, workspace to use for the lab.
* Azure SQL Server Database: server name, login and password.

### Task 1: Setup the connection to the Azure SQL Database

In this task, you will establish the Linked Service connection to an Azure SQL Database.

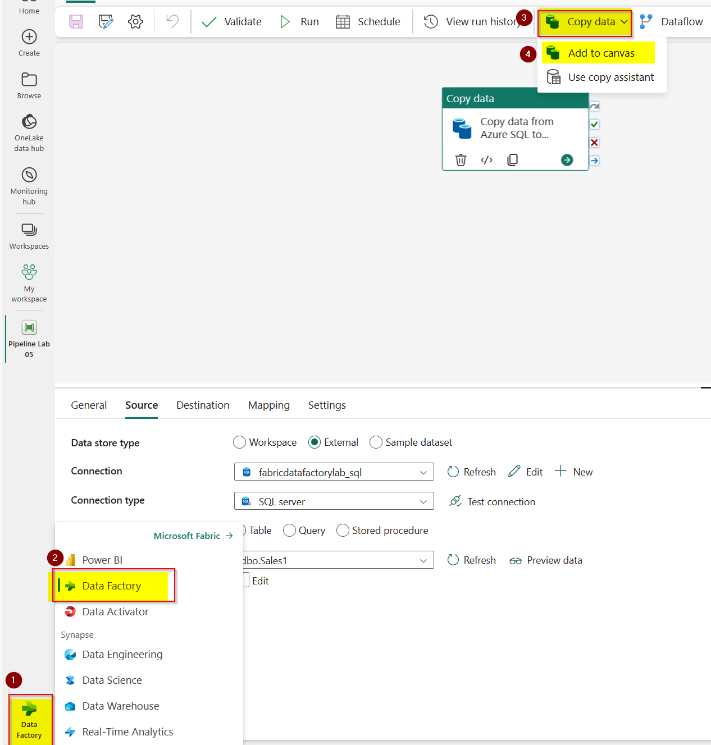
This database named FabricLabDB contains 3 tables with the same structure than the managed Sales table in the Lakehouse :

* Sales1 contains 10M rows
* Sales2 contains 50M rows
* Sales3 contains 100M rows

A screenshot of a computer

Description automatically generated

* On the MS Fabric menu, navigate to Data Factory and click on the Data pipeline. Provide the name Pipeline Lab 05 for this new Lab.
* On the newly created data pipeline, click on “Copy Data” then “Add to canvas”



* Configure the Data Copy Activity Task
  + Under the **General Tab**, provide the name for the Pipeline:***Copy data from Azure SQL to Lakehouse****.* The other settings remain unchanged.
* On the **Source Tab** choose the following options:
  + Data store type: External
  + Connection: Click on new and select Azure SQL Server, click on Continue.

**Server**: *<database server name>*.database.windows.net (information given by your trainer.

**Database**: fabriclabdb

**Connection**: Create new connection

**Connection name:** azuresqldb

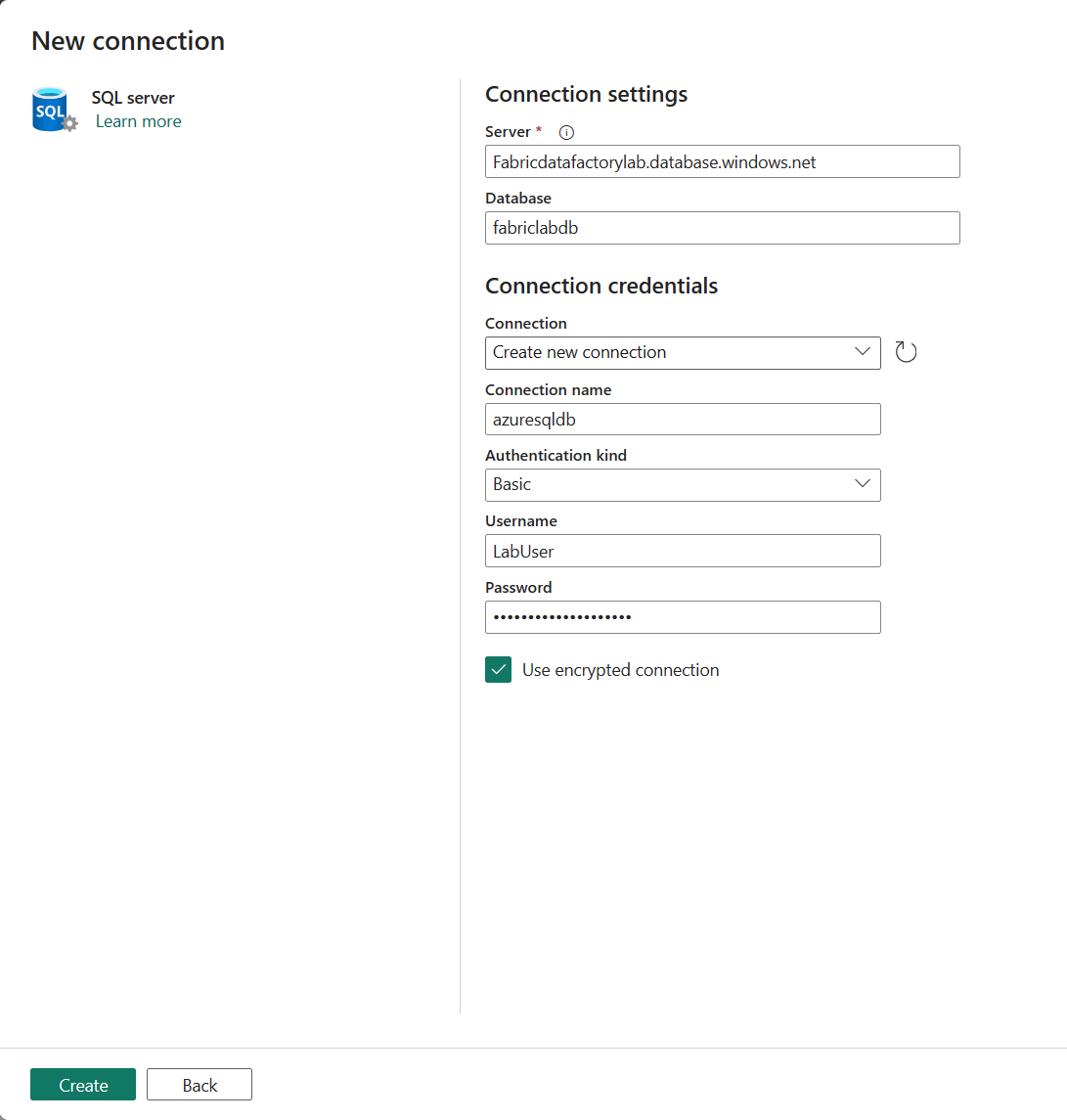
**Authentication** kind: Basic

**Username**: LabUser (to be confirmed by your trainer)

**Password**: [MicrosoftFabric@2023](mailto:MicrosoftFabric@2023) (to be confirmed by your trainer)

Thick Use **Encrypted Connection**

Make sure that the settings will look like this:



* Click on Test Connection to make sure that the connection is successful.

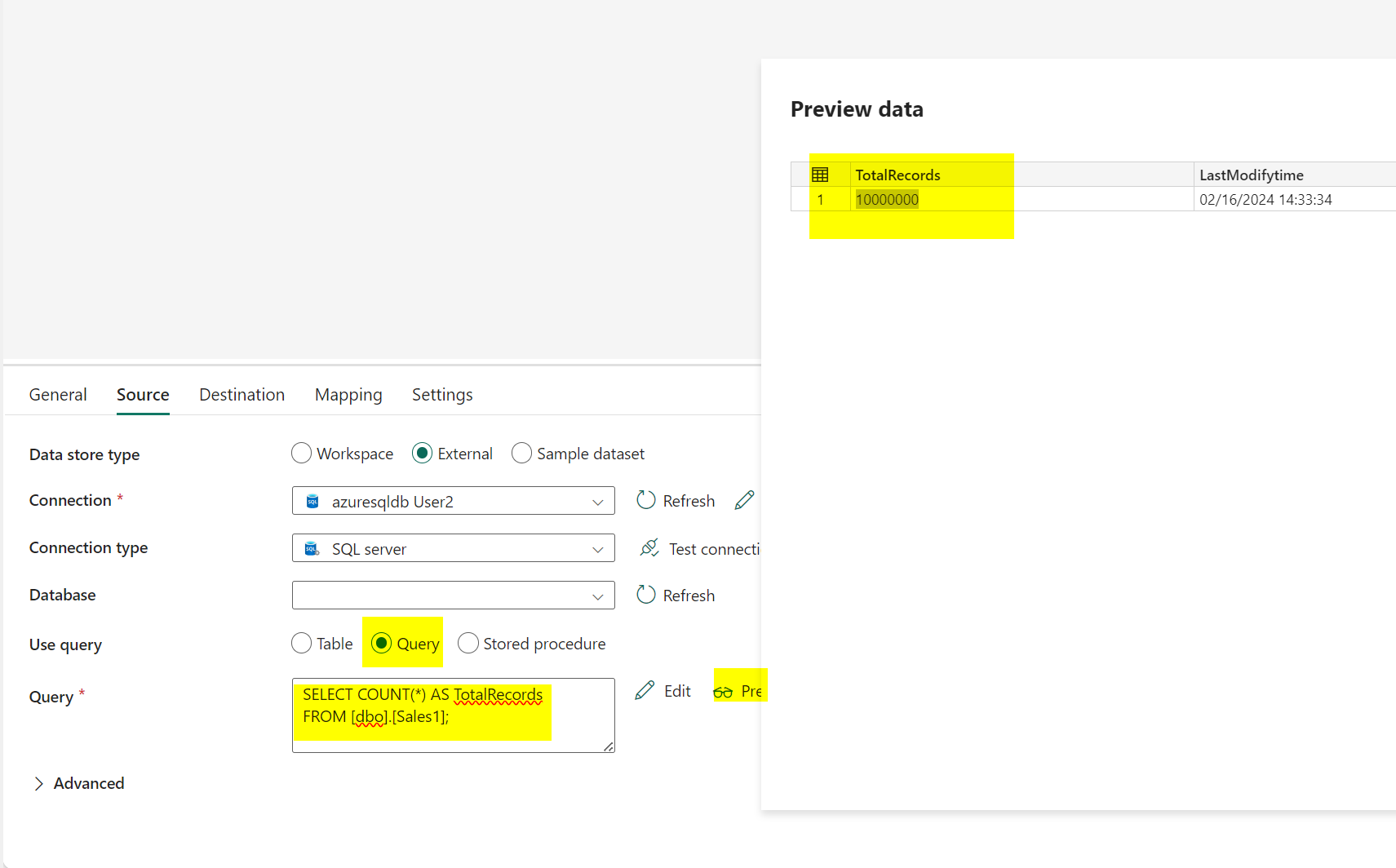
A screenshot of a computer

Description automatically generated

* Use query: Select Table
* **Table** **optional**: To find out how many records are on the dbo.Sales1, as a test you can select Query and paste this query:

SELECT COUNT(\*) AS TotalRecords

FROM [dbo].[Sales1];



* As you can see, the table dbo.Sales1 contains 10M of rows.
* **Table**: select **dbo.Sales1**. Click on Preview Data to make sure that you can read the data from the Database

A screenshot of a computer

Description automatically generated

A screenshot of a data table

Description automatically generated

* For the **Destination** tab, select the Lakehouse previously created

A screenshot of a computer

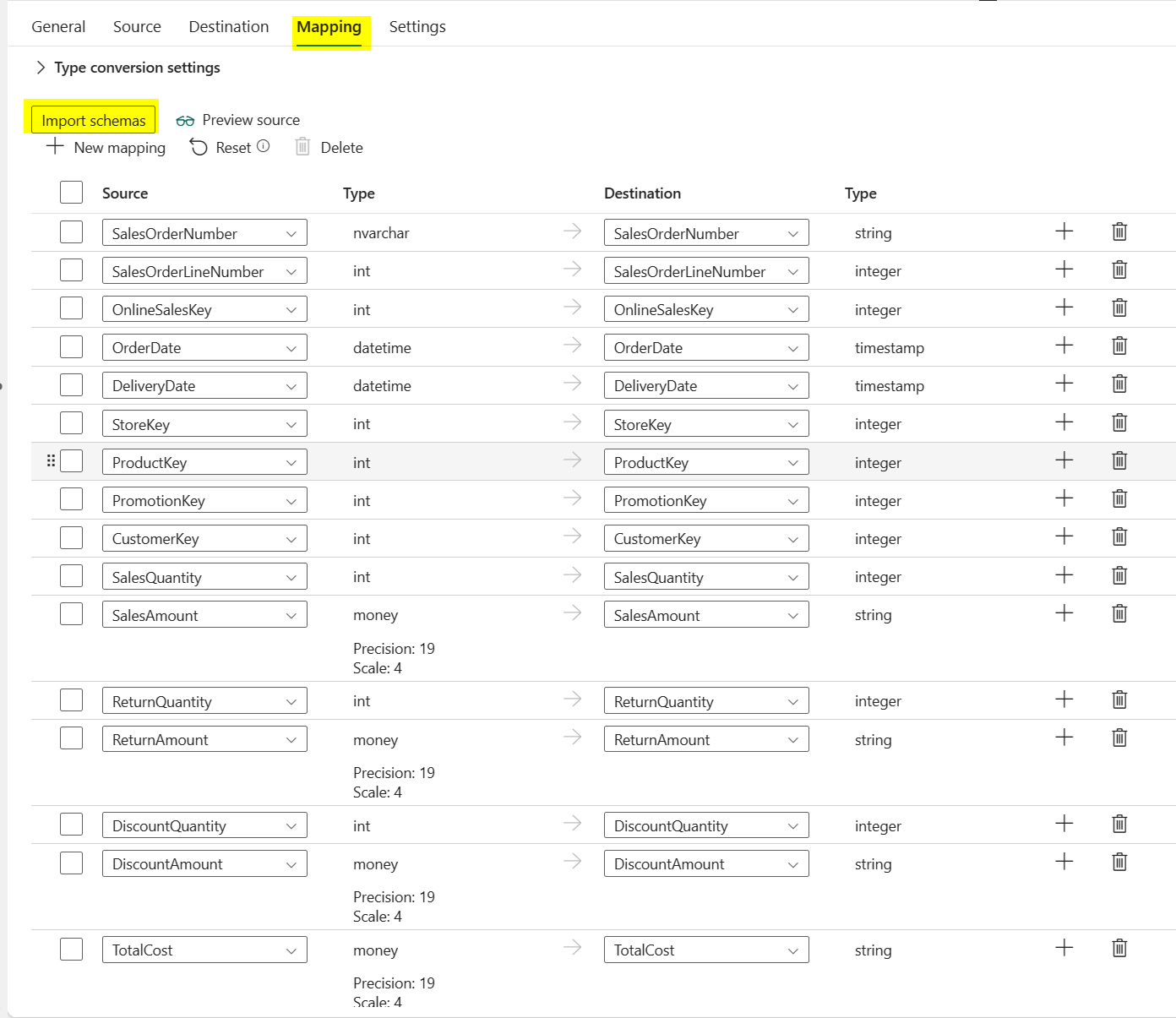
Description automatically generated

* Then select the table **Sales** on the Lakehouse

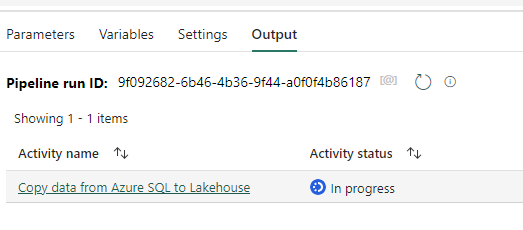
A screenshot of a computer

Description automatically generated

* On the **Mapping** tab select Import schemas:



* On the **Settings** tab, no changes are required
* Validate and Save the Pipeline
* You can now **Run** your pipeline
* During the pipeline execution, you can monitor the progression by clicking on the Activity in the Output section



* Details should appear, and 10M rows should we be written.

A screenshot of a computer

Description automatically generated

* Note the time spent to execute the pipeline to transfer 10M of rows

A screenshot of a computer

Description automatically generated

* As demonstrated in the lab 3, you can use the Duration Breakdown to understand where the execution time is mostly consumed – depending on the status of the Azure SQL DB used during the lab, the Reading from source (Light Blue) might be bigger.

A screenshot of a computer screen

Description automatically generated

* Using the Lakehouse explorer, you can also see the last Parquet file written during the last pipeline execution. This parquet file weights 50MB and contains 10M of records.

A white background with black lines

Description automatically generated

* It is also recommended to use the available analytics tools in Fabric to see the new ingested in the Sales Tables
  + **The SQL Analytics Endpoint:**

**A screenshot of a computer

Description automatically generated**

* + **The Power BI report**
    - You can add a simple visual card to count the number of rows in the Sales tables, and you don’t have to refresh the report or the semantic model, thanks to the **Direct Lakehouse Mode**

**A screenshot of a computer

Description automatically generated**

### Task 2: Use Data Pipeline Parameters to change the source table

In this task, you will declare a Parameter to be used in the Pipeline. Then you will load data from the second table (50M of records)

* Open the Fabric Data Factory Pipeline, and go to the **Parameters** section

A white background with a black line

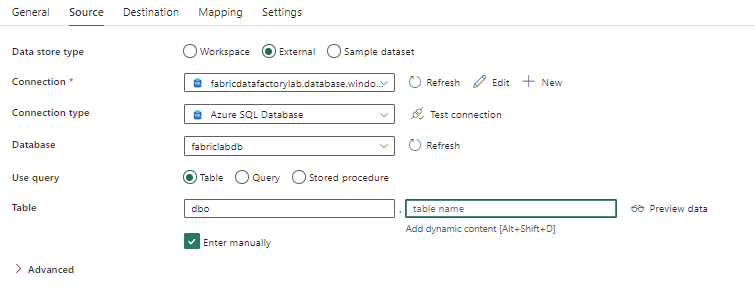
Description automatically generated with medium confidence

* Define the following parameter:
  + Name : **TableName**
  + Type : **String**
  + DefaultValue : **Sales2**

A screenshot of a computer

Description automatically generated

* Select the **Copy Data Activity** in the Pipeline, go to the **Source**, and enable the **Enter manually** check box
* Click on the **Add dynamic content** link to access to the Pipeline expression builder



* From the Pipeline Expression Builder, drag and drop the TableName parameter, and the following expression should appear. Click on OK.

A screenshot of a computer

Description automatically generated

* The expression should be now visible in the table name section.

A screenshot of a computer

Description automatically generated

* The expression should be now visible in the table name section.
* Save the pipeline and click on Run.
* As the Pipeline now contains a parameter, you will have the choice to keep the default parameter value, or change it before the execution. Keep the default parameter value for this execution to get data from the table Sales2.

A screenshot of a computer

Description automatically generated

* At the end of the pipeline execution, you can observe that :
  + 50M of records were loaded
  + 4 files were written
  + The parallelism was 1 (in the next lab you will configure the parallelism using a parameter).

A screenshot of a computer

Description automatically generated

### Task 3: Optimize Data Pipeline performance with parallelism and partitions

In this task, you will declare a second Parameter to be used as the level of parallelism in order to accelerate the data transfer and optimize performance by reducing the execution duration.

* Create a second parameter with the following settings:
  + Name: Parallelism
  + Type: Int
  + Default Value: 4

A screenshot of a computer

Description automatically generated

* Use this parameter to dynamically configure the Degree of copy parallelism (in the settings section of the Copy activity)

A screenshot of a computer

Description automatically generated

* On the Source section, enable the partitioning by selecting **Physical partitions** of table in the **Partition option** item.

A screenshot of a computer

Description automatically generated

* Save the pipeline and run it with the following configuration:
  + TableName : Sales3 (100M of rows)
  + Parralelism : 4

A screenshot of a computer

Description automatically generated

* At the end of the pipeline execution you can observe :
  + A better throughput when transferring data
  + Good performance (only 2mn more when moving from 50M to 100M of rows)

A screenshot of a computer

Description automatically generated