Quick Demo Notes

These quick-start notes are primarily intended to support those who attended the ‘Exploring data technologies using free tools’ session and would like to get up and running with a local SQL Server and Python setup for learning/experimenting on a Windows computer.

Before following these instructions, you need to have the tools/apps outlined in the ‘Install Tools and Apps’ document. This and other related files can be found in the repository at:

https://github.com/phil-a10/Talks/tree/main/Data%20Engineering%20Using%20Free%20Tools

SQL Server via SSMS: Get, Select and Export Data

This simple example shows how to download and attach one of the Microsoft example database ‘Adventure Works’ variants so you have a sample dataset to work with;

Download Database

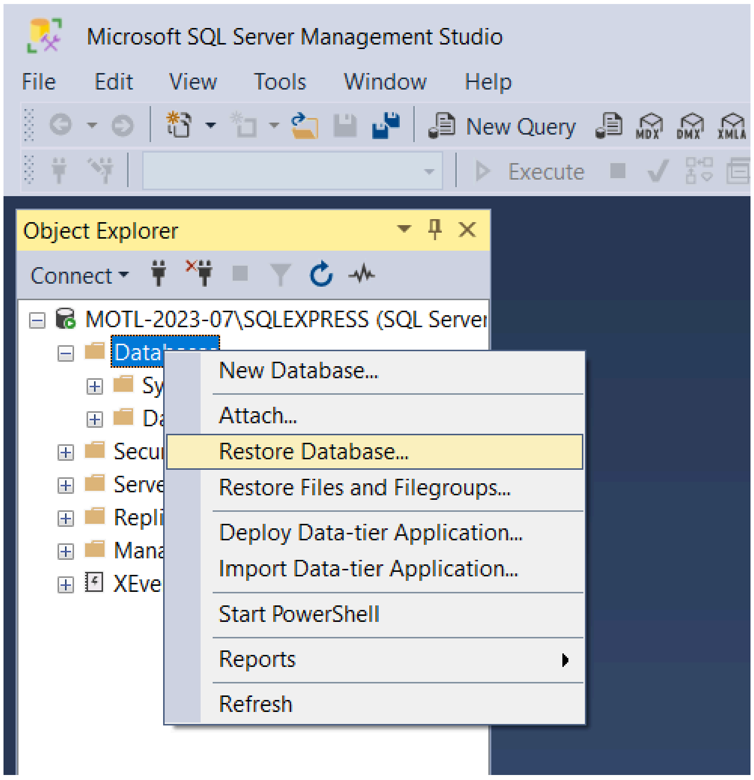
There are a number of variations of the sample database that Microsoft makes available. For this example the current ‘OLTP’ sample has been chosen.

- https://learn.microsoft.com/en-us/sql/samples/adventureworks-install-configure?view=sql-server-ver16&tabs=ssms
- Save **AdventureWorks2022.bak (OLTP sample)** to **C:\Program Files\Microsoft SQL Server\MSSQL16.SQLEXPRESS\MSSQL\Backup**
    - It is important to save it to this location, so that SQL Server Management Studio can locate it!

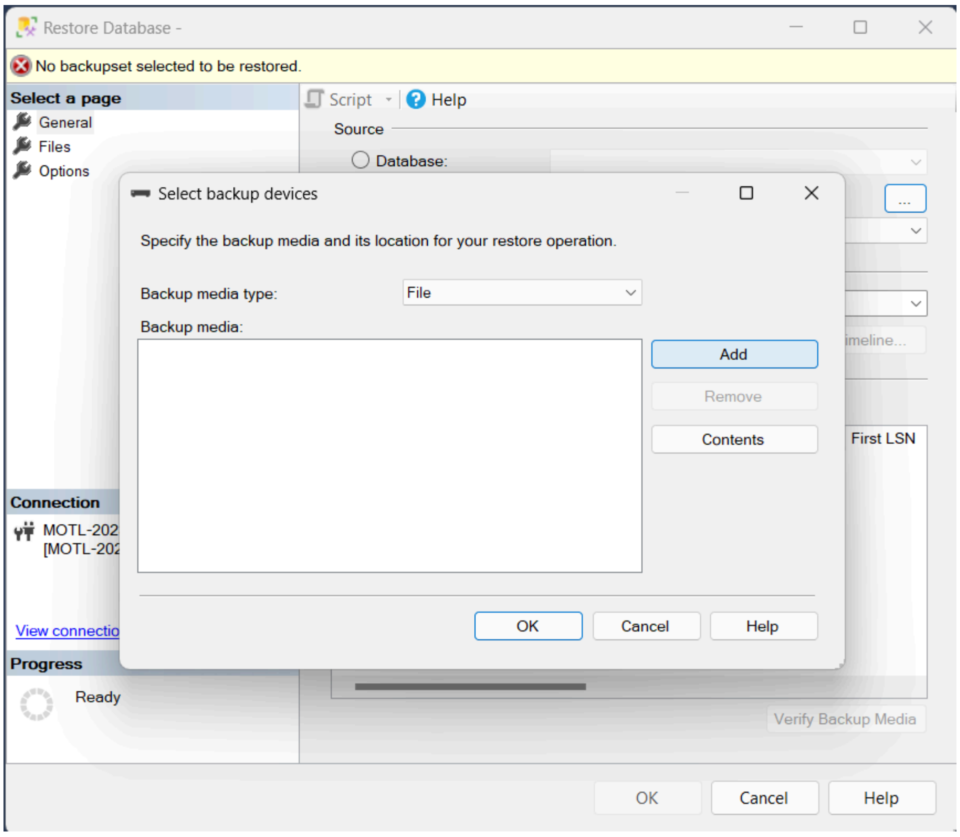
Restore Database

- Open SSMS

Right-click on **Databases** in **Object Explorer** and select **Restore Database…**


Microsoft SQL Server Management Studio
File
Edit
View
Tools
Window
Help
New Query
Execute
Object Explorer
Connect
MOTL-2023-07\SQLEXPRESS (SQL Server
Databases
New Database...
Attach...
Restore Database...
Restore Files and Filegroups...
Deploy Data-tier Application...
Import Data-tier Application...
Start PowerShell
Reports
Refresh

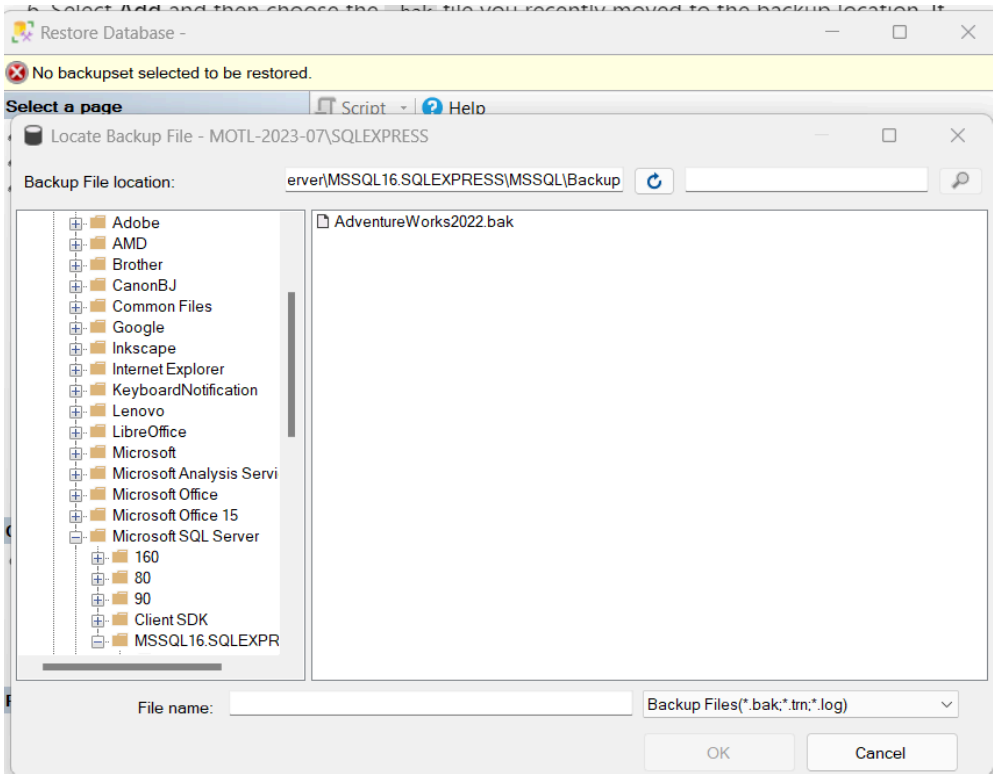
- Select **Add**



No backupset selected to be restored.

Specify the backup media and its location for your restore operation.

- Choose **AdventureWorks2022.bak**


Restore Database -
No backupset selected to be restored.
Select a page
Locate Backup File - MOTL-2023-07\SQLEXPRESS
Backup File location: erver\MSSQL16.SQLEXPRESS\MSSQL\Backup
AdventureWorks2022.bak

(Select and OK various messages… Successfully restored!)

Create views

In the talk, we showed the use of views to create the data sources that we'd export. The code to create thes in an existing database schema is below:

Create view 'vScratch1'

Code starts below (Copy and Paste into SSMS and Execute)

```
USE [AdventureWorks2022]
GO

/*

Sample user-created view to test/demonstrate use of SSMS to generate a view
which will be exported as a CSV, for import to Python.

This has been created in the [Production] schema for quick demo purposes, but
in a real-world scenario, it is likely that creating a new schema would be more
appropriate.


The un-selected columns from the joined tables have been commented out rather
than removed, in case needed but in a production scenario I would aim to clean
things up so that only relevant comments remained/redundant code was removed.
```

```
*/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE VIEW [Production].[vScratch1]
AS


SELECT a.[PurchaseOrderID]
      ,a.[PurchaseOrderDetailID]
      --,a.[DueDate]
      ,a.[OrderQty]
      ,a.[ProductID]
      ,a.[UnitPrice]
        ,b.Name
      --,a.[LineTotal]
      --,a.[ReceivedQty]
      --,a.[RejectedQty]
      --,a.[StockedQty]
      --,a.[ModifiedDate]
  FROM [AdventureWorks2022].[Purchasing].[PurchaseOrderDetail] AS a
  INNER JOIN [Production].[Product] AS b ON b.ProductID = a.ProductID

GO
```

Code ends above

(To execute it, click ‘Execute’, or press F5)

Create view ‘vScratch2’

Code starts below (Copy and Paste into SSMS and Execute)

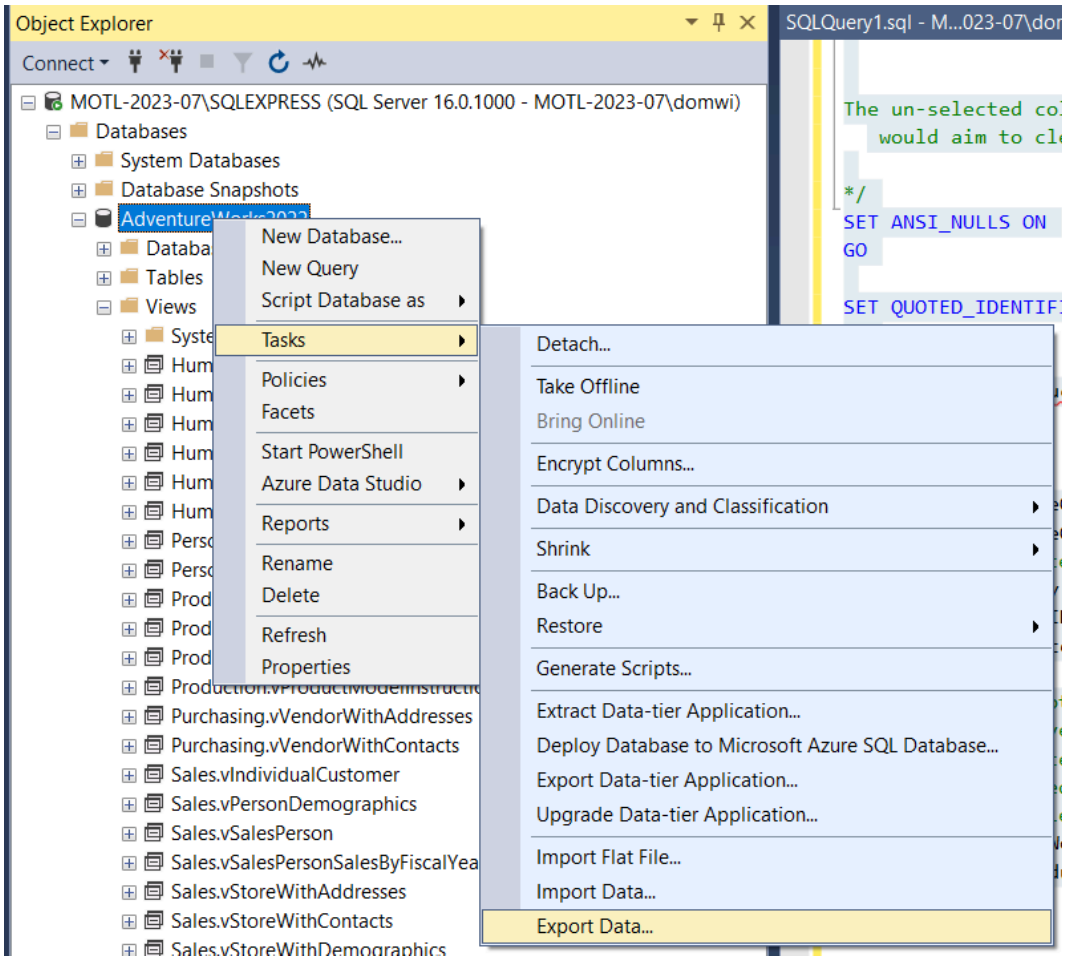
```
USE [AdventureWorks2022]
GO

/*

Sample user-created view to test/demonstrate use of SSMS to generate a view
which will be exported as a CSV, for import to Python.

This has been created in the [Production] schema for quick demo purposes, but
in a real-world scenario, it is likely that creating a new schema would be more
appropriate.


The un-selected columns from the joined tables have been commented out rather
than removed, in case needed but in a production scenario I would aim to clean
things up so that only relevant comments remained/redundant code was removed.
```

```
*/

SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO


CREATE VIEW [Production].[vScratch2]
AS

SELECT MAX([Name]) AS 'Item Name'
      ,SUM([OrderQty]) AS 'Order Quantity'
  FROM [AdventureWorks2022].[Production].[vScratch1]
  GROUP BY [ProductID]
GO
```

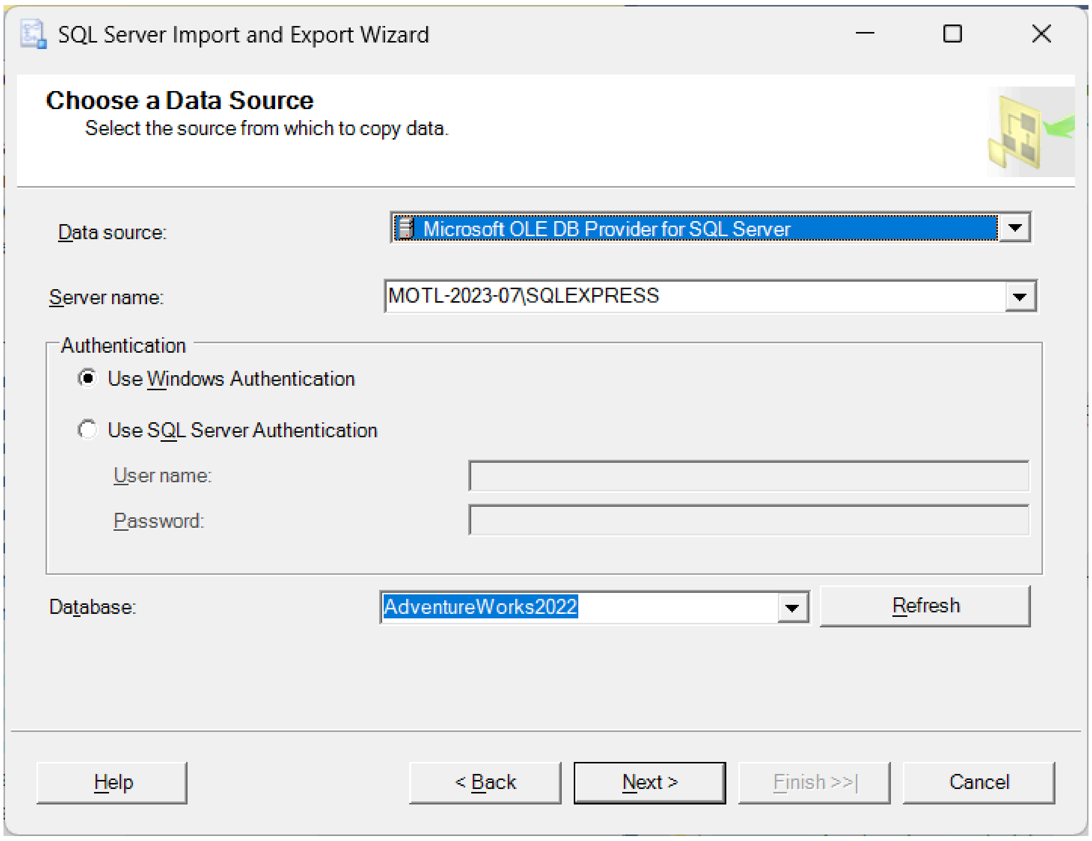
Code ends above

Export data

- Right-click on the **AdventureWords2022** database
- **Tasks > Export Data…**


Object Explorer
Connect
MOTL-2023-07\SQLEXPRESS (SQL Server 16.0.1000 - MOTL-2023-07\domwi)
Databases
System Databases
Database Snapshots
AdventureWorks2022
Tables
Views
Purchasing.vVendorWithAddresses
Purchasing.vVendorWithContacts
Sales.vIndividualCustomer
Sales.vPersonDemographics
Sales.vSalesPerson
Sales.vSalesPersonSalesByFiscalYea
Sales.vStoreWithAddresses
Sales.vStoreWithContacts
Sales.vStoreWithDemographics
New Database...
New Query
Script Database as
Tasks
Policies
Facets
Start PowerShell
Azure Data Studio
Reports
Rename
Delete
Refresh
Properties
Detach...
Take Offline
Bring Online
Encrypt Columns...
Data Discovery and Classification
Shrink
Back Up...
Restore
Generate Scripts...
Extract Data-tier Application...
Deploy Database to Microsoft Azure SQL Database...
Export Data-tier Application...
Upgrade Data-tier Application...
Import Flat File...
Import Data...
Export Data...
SQLQuery1.sql - M...023-07\dor

```
The un-selected co
 would aim to cl
*/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIF
```


SQL Server Import and Export Wizard

Choose a Data Source

Select the source from which to copy data.

Data source: Microsoft OLE DB Provider for SQL Server

Server name: MOTL-2023-07\SQLEXPRESS

Authentication

● Use Windows Authentication

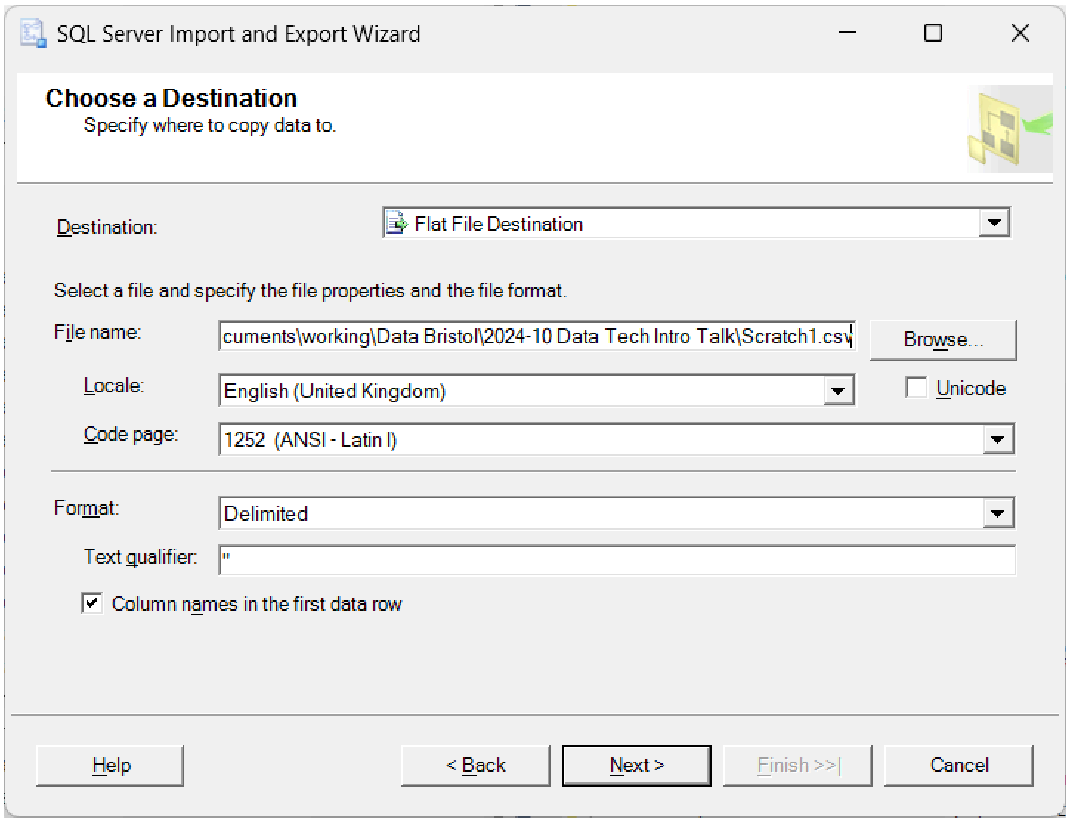
○ Use SQL Server Authentication

User name:

Password:

Database: AdventureWorks2022

- Choose **Flat File Destination**
- Browse to the directory where you'll be saving files, enter **Scratch1** as the name, and a file type of **CSV**
- **Set Text qualifier to quotes "**
  - (**Important -** So that the CSV will be suitably formatted for Python 'Panda's' import)


SQL Server Import and Export Wizard

Choose a Destination

Specify where to copy data to.

Destination: Flat File Destination

Select a file and specify the file properties and the file format.

File name: cuments\working\Data Bristol\2024-10 Data Tech Intro Talk\Scratch1.csv

Locale: English (United Kingdom)

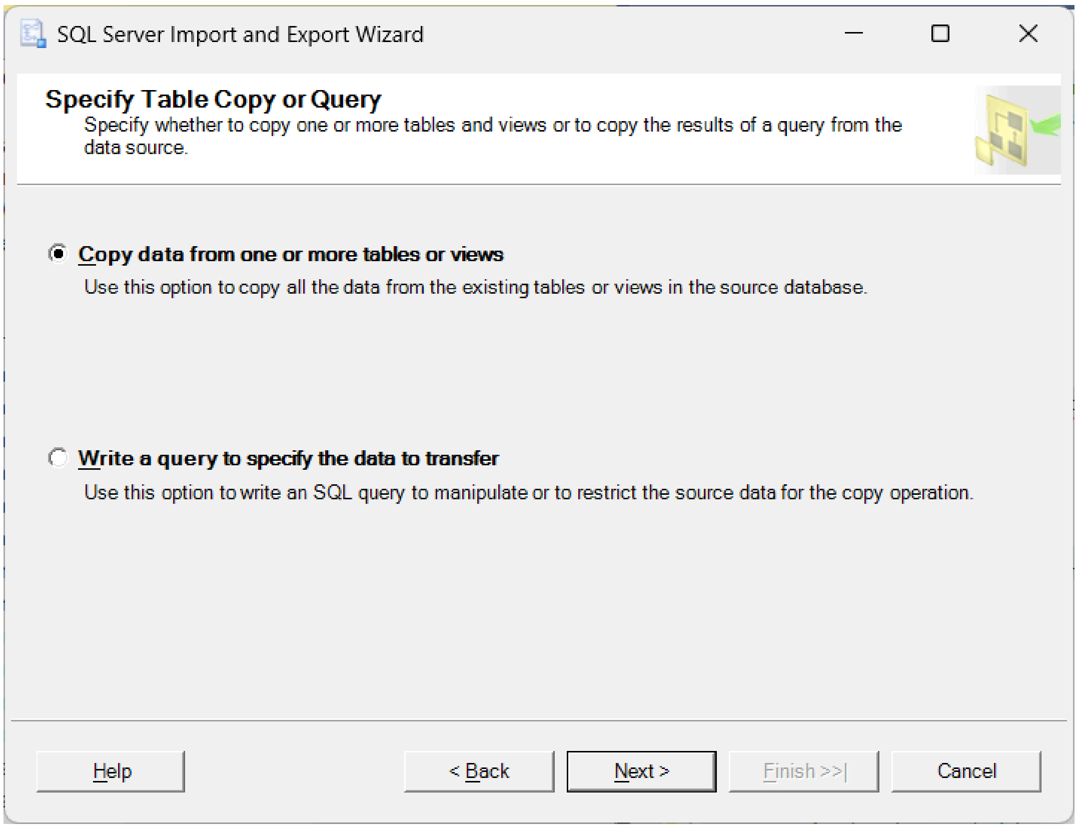
☐ Unicode

Code page: 1252 (ANSI - Latin I)

Format: Delimited

Text qualifier: "

☑ Column names in the first data row


SQL Server Import and Export Wizard

Specify Table Copy or Query

Specify whether to copy one or more tables and views or to copy the results of a query from the data source.

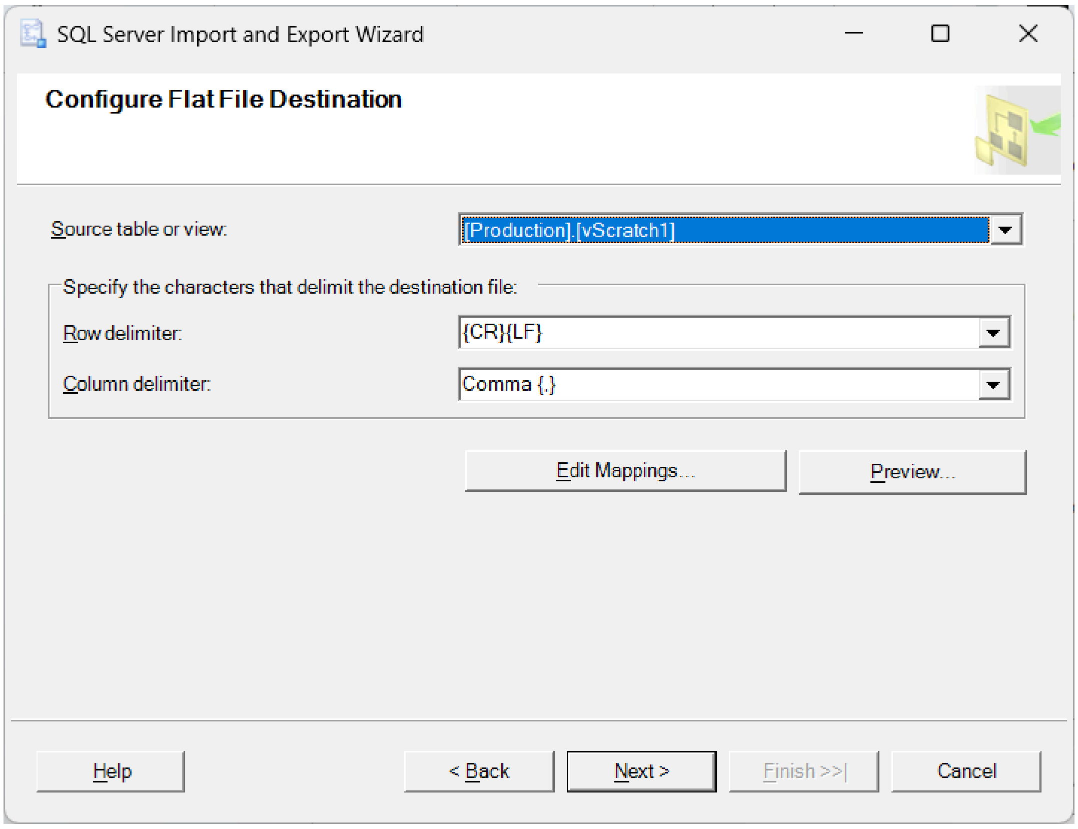
[✓ Copy data from one or more tables or views]

Use this option to copy all the data from the existing tables or views in the source database.

Write a query to specify the data to transfer

Use this option to write an SQL query to manipulate or to restrict the source data for the copy operation.

- Set the source to **[Production].[vScratch1]**
    - *Because this is a view, you'll have to scroll down the sources to the second set of 'Production' entries, as it does an A to Z list of tables first, and then and A to Z list of views*


SQL Server Import and Export Wizard

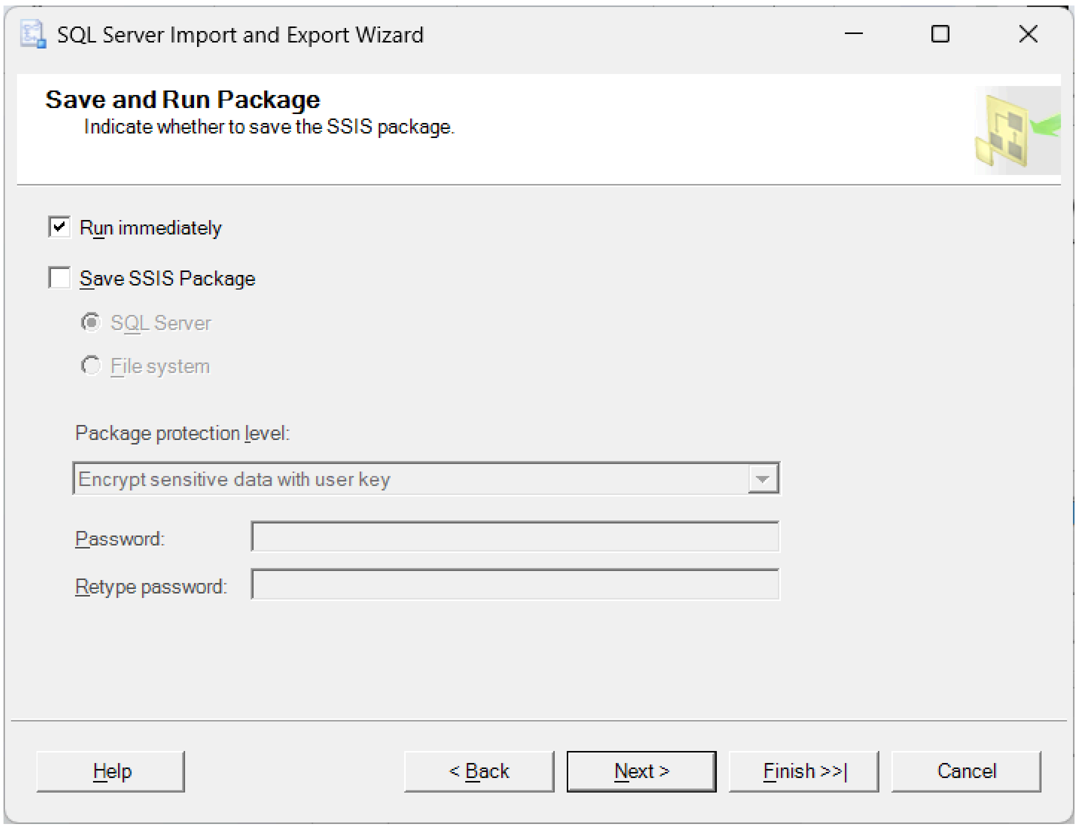
Configure Flat File Destination

Source table or view: [Production].[vScratch1]

Specify the characters that delimit the destination file:

Row delimiter: {CR}{LF}

Column delimiter: Comma {,}


SQL Server Import and Export Wizard

Save and Run Package

Indicate whether to save the SSIS package.

[✓] Run immediately

[ ] Save SSIS Package

(●) SQL Server

( ) File system

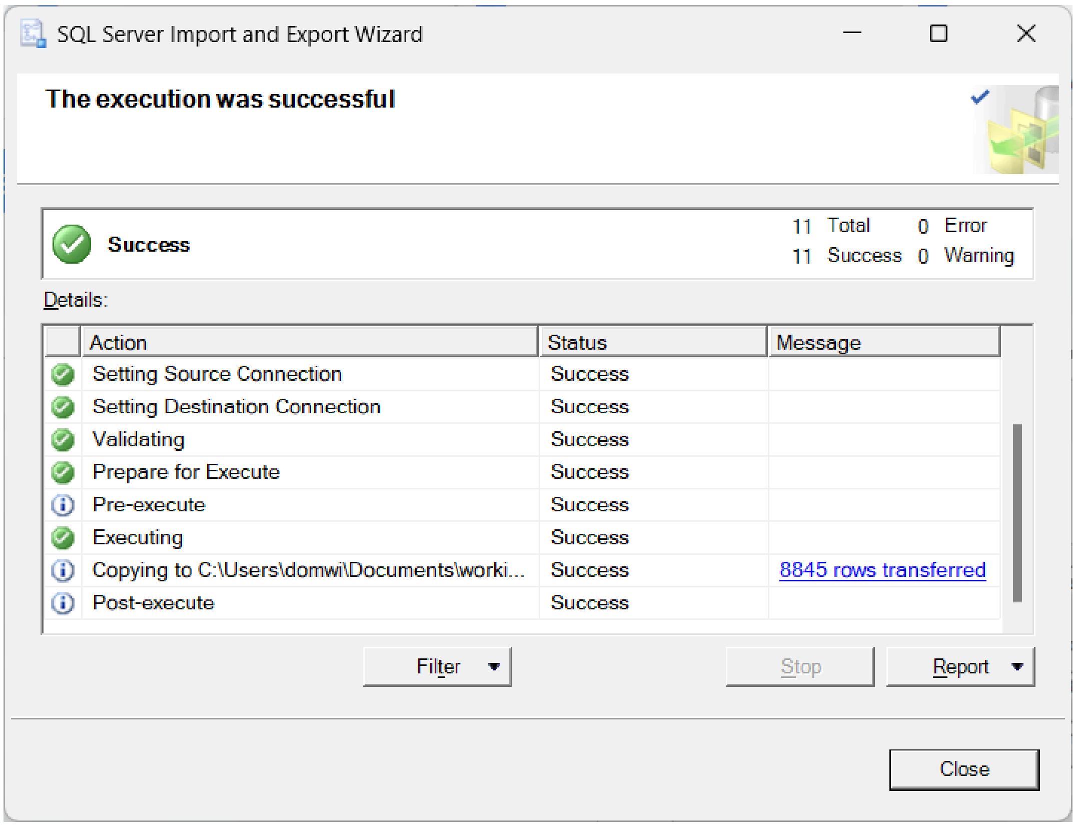
Package protection level:

Encrypt sensitive data with user key

Password:

Retype password:

(Next/Finish until done)


SQL Server Import and Export Wizard

The execution was successful

Success

11	Total	0	Error
11	Success	0	Warning

Details:

Action	Status	Message
Setting Source Connection	Success	
Setting Destination Connection	Success	
Validating	Success	
Prepare for Execute	Success	
Pre-execute	Success	
Executing	Success	
Copying to C:\Users\domwi\Documents\worki...	Success	8845 rows transferred
Post-execute	Success	

- **Repeat the above for the Scratch 2 view**

Import and use in Python

In the quick-start demo, one Python file was created/used (included further down this section), and the individual blocks were run in sequence (select code block, then SHIFT + RETURN)

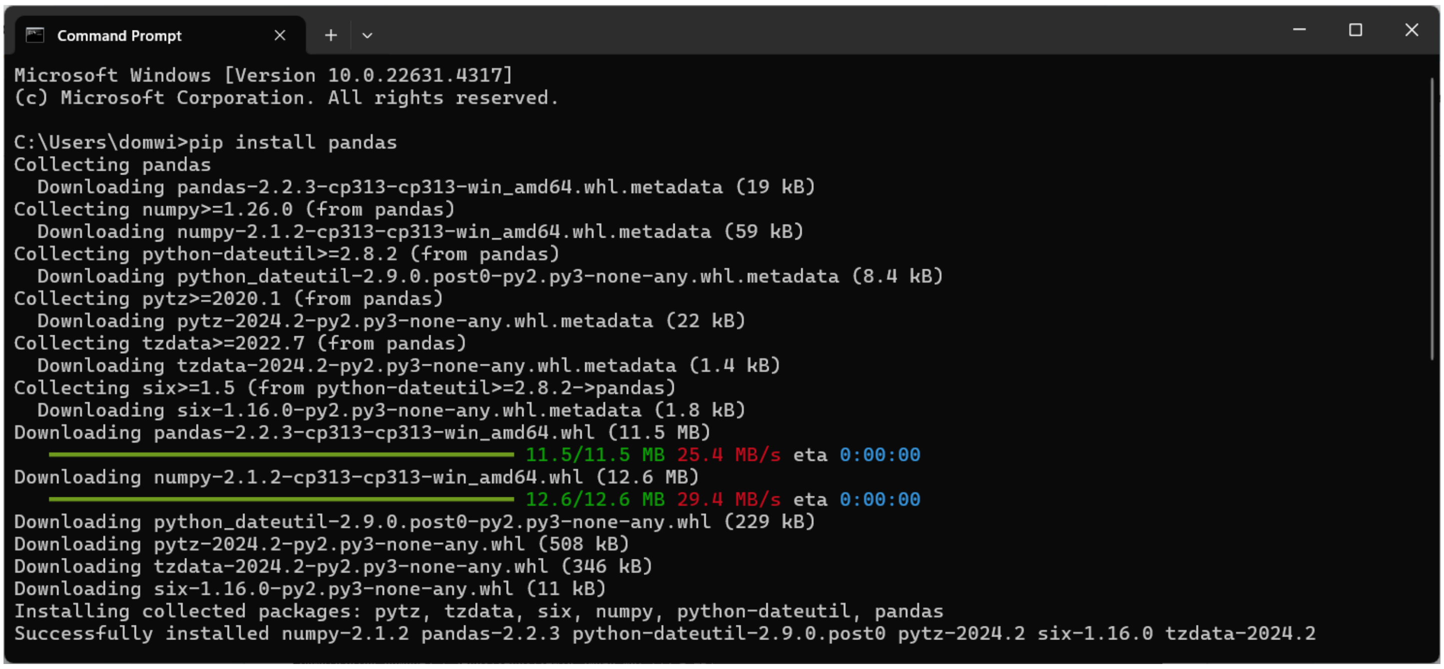
Setup Python

Install required PIP packages

Pandas and Plotly will be needed to run the example code populating a chart from one of the saved CSV files.

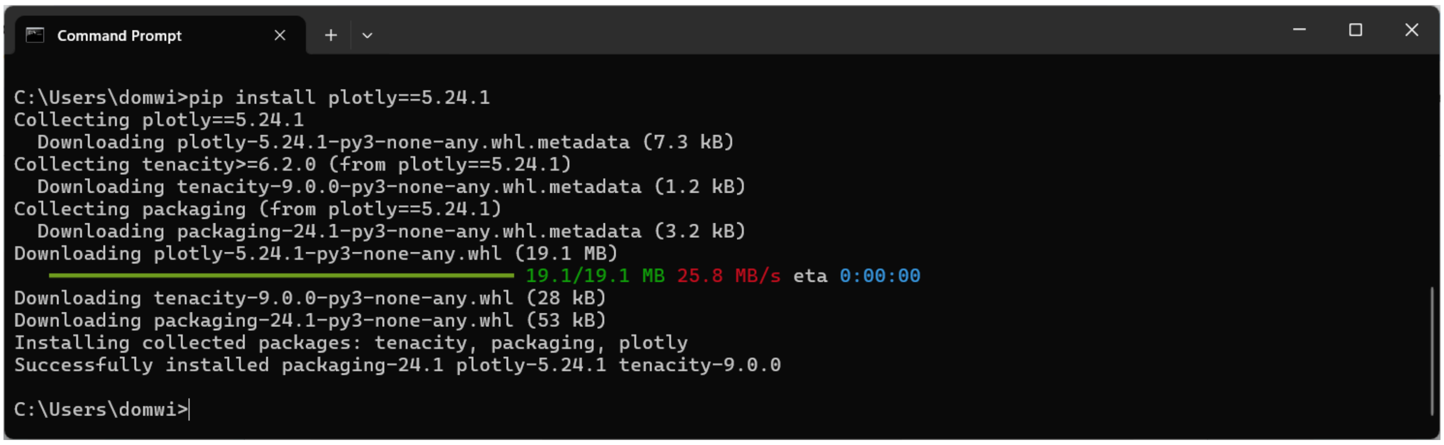
Pandas enables manipulation of data using data frames, and will be used to populate Plotly with chart data

- From the Command Prompt (*not Python!)*, type **pip install pandas**
    - This may take a few minutes. For a new installation, you should see something similar to the screenshot below
- Reference: https://pandas.pydata.org/docs/getting_started/install.html



```
Microsoft Windows [Version 10.0.22631.4317]
(c) Microsoft Corporation. All rights reserved.

C:\Users\domwi>pip install pandas
Collecting pandas
  Downloading pandas-2.2.3-cp313-cp313-win_amd64.whl.metadata (19 kB)
Collecting numpy>=1.26.0 (from pandas)
  Downloading numpy-2.1.2-cp313-cp313-win_amd64.whl.metadata (59 kB)
Collecting python-dateutil>=2.8.2 (from pandas)
  Downloading python_dateutil-2.9.0.post0-py2.py3-none-any.whl.metadata (8.4 kB)
Collecting pytz>=2020.1 (from pandas)
  Downloading pytz-2024.2-py2.py3-none-any.whl.metadata (22 kB)
Collecting tzdata>=2022.7 (from pandas)
  Downloading tzdata-2024.2-py2.py3-none-any.whl.metadata (1.4 kB)
Collecting six>=1.5 (from python-dateutil>=2.8.2->pandas)
  Downloading six-1.16.0-py2.py3-none-any.whl.metadata (1.8 kB)
Downloading pandas-2.2.3-cp313-cp313-win_amd64.whl (11.5 MB)
   ---------------------------------------- 11.5/11.5 MB 25.4 MB/s eta 0:00:00
Downloading numpy-2.1.2-cp313-cp313-win_amd64.whl (12.6 MB)
   ---------------------------------------- 12.6/12.6 MB 29.4 MB/s eta 0:00:00
Downloading python_dateutil-2.9.0.post0-py2.py3-none-any.whl (229 kB)
Downloading pytz-2024.2-py2.py3-none-any.whl (508 kB)
Downloading tzdata-2024.2-py2.py3-none-any.whl (346 kB)
Downloading six-1.16.0-py2.py3-none-any.whl (11 kB)
Installing collected packages: pytz, tzdata, six, numpy, python-dateutil, pandas
Successfully installed numpy-2.1.2 pandas-2.2.3 python-dateutil-2.9.0.post0 pytz-2024.2 six-1.16.0 tzdata-2024.2
```

- Also from the command prompt, install Plotly by typing **pip install plotly==5.24.1**
    - This may take a few minutes. For a new installation, you should see something similar to the screenshot below
- Reference: https://plotly.com/python/getting-started/



```
C:\Users\domwi>pip install plotly==5.24.1
Collecting plotly==5.24.1
  Downloading plotly-5.24.1-py3-none-any.whl.metadata (7.3 kB)
Collecting tenacity>=6.2.0 (from plotly==5.24.1)
  Downloading tenacity-9.0.0-py3-none-any.whl.metadata (1.2 kB)
Collecting packaging (from plotly==5.24.1)
  Downloading packaging-24.1-py3-none-any.whl.metadata (3.2 kB)
Downloading plotly-5.24.1-py3-none-any.whl (19.1 MB)
   ---------------------------------------- 19.1/19.1 MB 25.8 MB/s eta 0:00:00
Downloading tenacity-9.0.0-py3-none-any.whl (28 kB)
Downloading packaging-24.1-py3-none-any.whl (53 kB)
Installing collected packages: tenacity, packaging, plotly
Successfully installed packaging-24.1 plotly-5.24.1 tenacity-9.0.0

C:\Users\domwi>
```

Python code

```python
# I always like to to a check that the environment is up and running

print ('Hello, world!')

# Check that the CSV files can be read okay using the built-in CSV functionality
first

# Example 1 uses a straightforward path to the file (e.g. copy and paste it from the
address bar in Windows Explorer)
# BUT NOTE:
#
# - This will need to be replaced with the file path from the system you're using (as
I assume you haven't cloned my laptop!)
# - When copied, the path contained back-strokes (\), I have had to change these to
forward strokes (/)
#
# (The purpose of this demo is not to go into how to construct local file paths in
detail, but this is a potentially tricky area - some reading up is recommended!)

# First, make the csv functionality available by importing it

import csv

# Then let's try and open, then print the contents of, the first Scratch file

with open('C:/Users/domwi/Documents/working/Data Bristol/2024-10 Data Tech intro
talk/Scratch1.csv', newline='') as csvfile:
    DemoFile1 = csv.reader(csvfile, delimiter=',', quotechar='|')
    for row in DemoFile1:
        print(', '.join(row))

# Now the second Scratch file

with open('C:/Users/domwi/Documents/working/Data Bristol/2024-10 Data Tech intro
talk/Scratch2.csv', newline='') as csvfile2:
    DemoFile2 = csv.reader(csvfile2, delimiter=',', quotechar='|')
    for row in DemoFile2:
        print(', '.join(row))

# Import Plotly and check that it's up and running okay (simple example from Plotly
docs) - should open a web browser and show a chart

import plotly.graph_objects as go

fig = go.Figure(go.Bar(
            x=[20, 14, 23],
            y=['giraffes', 'orangutans', 'monkeys'],
            orientation='h'))

fig.show()

# Now to try loading a CSV via Pandas and displaying a chart with Plotly (example
from Plotly docs)

import pandas as pd
import plotly.express as px

df =
pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/2014_apple_stoc
k.csv')

fig = px.line(df, x = 'AAPL_x', y = 'AAPL_y', title='Apple Share Prices over time
(2014)')
fig.show()


# And then an example loading the 'Scratch2' CSV from the location I saved it in:

df = pd.read_csv('C:/Users/domwi/Documents/working/Data Bristol/2024-10 Data Tech
intro talk/Scratch2.csv', sep = ',')

fig = px.bar(df, x = 'Item Name', y = 'Order Quantity', title='Demo chart 1')
fig.show()


# But ideally, I'd like this horizontally. In Plotly this is easily achieved by
switching the x/y axis, and setting the orientation to h
# I'd also like to order the values, which is easily achieved using 'sort_values' in
the dataframe...

df = pd.read_csv('C:/Users/domwi/Documents/working/Data Bristol/2024-10 Data Tech
intro talk/Scratch2.csv', sep = ',')

fig = px.bar(df.sort_values(by=['Order Quantity']), x = 'Order Quantity', y = 'Item
Name', title='Demo chart 1', orientation='h')
fig.show()

# End of demo code
```