Accessing your data from the database

- Please follow the steps in this notebook to have access to the dataset.
- If you encounter any challenges please leave an issue on this repo here on GitHub

Steps to take to use environment variables as opposed to credentials literals

1. Install pyodbc - a package for creating connection strings to your remote database
2. Install python-dotenv - a package for creating environment variables that will help you hide sensitve configuration informantion such as database credentials and API keys
3. Import all the necessary libraies
   A. pyodbc (for creating a connection)
   B. python-dotenv (loading environment variables)
   C. os (for accessing the environement variables using the load_env function. This is not needed if you use the dotenv_values function instead)
4. Now create a file called .env in the root of your project folder (Note, the file name begins with a dot)
5. In the .env file, put all your sensitive information like server name, database name, username, and password

Example

- SERVER='server_name_here'
- DATABASE='database_name_here'
- USERNAME='username_here'
- PASSWORD='password_here'

1. Next create a .gitignore file (a new file with the name '.gitignore'. Note that gitignore file names begin with a dot)
2. Open the .gitignore file and type in the name of the .env file we just create like this "/.env". This will prevent git from tracking that file. Essesntially any file name in the gitignore file will be ignored by git and won't be checked into the repository
3. Create a connection by accessing your connection string with your defined environment variables

Step 1 and 2 - Install pyodbc and python-dotenv

In [ ]:
```
%pip install pyodbc
%pip install python-dotenv
```

Step 3 - Import all the necessary packages

In [97]:
```
import pyodbc #just installed with pip
from dotenv import dotenv_values #import the dotenv_values function from the dotenv package
import pandas as pd
import warnings

warnings.filterwarnings('ignore')
```

Out[97]:
```
True
```

Step 4 - Create your .env file in the root of your project

Step 5 - In the .env file, put all your sensitive information like server name, password etc

Step 6 & 7 - Next create a .gitignore file and type '/.env' file we just created. This will prevent git from tracking that file.

Step 8 - Create a connection by accessing your connection string with your defined environment variables

In [92]:
```
# Load environment variables from .env file into a dictionary
environment_variables = dotenv_values('.env')


# Get the values for the credentials you set in the '.env' file
database = environment_variables.get("DATABASE")
server = environment_variables.get("SERVER")
username = environment_variables.get("USERNAME")
password = environment_variables.get("PASSWORD")


connection_string = f"DRIVER={{SQL Server}};SERVER={server};DATABASE={database};UID={username};PWD={password}"
```

In [93]:
```
# Use the connect method of the pyodbc library and pass in the connection string.
# This will connect to the server and might take a few seconds to be complete.
# Check your internet connection if it takes more time than necessary

connection = pyodbc.connect(connection_string)
```

In [94]:
```
# Now the sql query to get the data is what what you see below.
# Note that you will not have permissions to insert delete or update this database table.

query = "Select * from dbo.LP2_Telco_churn_first_3000"
data = pd.read_sql(query, connection)
```

In [80]:
```
data.head()
```

Out[80]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection	TechSupport	Stream
0	7590-VHVEG	Female	False	True	False	1	False	None	DSL	False	...	False	False	
1	5575-GNVDE	Male	False	False	False	34	True	False	DSL	True	...	True	False	
2	3668-QPYBK	Male	False	False	False	2	True	False	DSL	True	...	False	False	
3	7795-CFOCW	Male	False	False	False	45	False	None	DSL	True	...	True	True	
4	9237-HQITU	Female	False	False	False	2	True	False	Fiber optic	False	...	False	False	

5 rows × 21 columns

In [81]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3000 entries, 0 to 2999
Data columns (total 21 columns):
 #   Column            Non-Null Count  Dtype 
---  ------            --------------  ----- 
 0   customerID        3000 non-null   object
 1   gender            3000 non-null   object
 2   SeniorCitizen     3000 non-null   bool  
 3   Partner           3000 non-null   bool  
 4   Dependents        3000 non-null   bool  
 5   tenure            3000 non-null   int64 
 6   PhoneService      3000 non-null   bool  
 7   MultipleLines     2731 non-null   object
 8   InternetService   3000 non-null   object
 9   OnlineSecurity    2349 non-null   object
 10  OnlineBackup      2349 non-null   object
 11  DeviceProtection  2349 non-null   object
 12  TechSupport       2349 non-null   object
 13  StreamingTV       2349 non-null   object
 14  StreamingMovies   2349 non-null   object
 15  Contract          3000 non-null   object
 16  PaperlessBilling  3000 non-null   bool  
 17  PaymentMethod     3000 non-null   object
 18  MonthlyCharges    3000 non-null   float64
 19  TotalCharges      2995 non-null   float64
 20  Churn             2999 non-null   object
dtypes: bool(5), float64(2), int64(1), object(13)
memory usage: 389.8+ KB
```

In [90]:
```
data2 = pd.read_csv('LP2_Telco-churn-last-2000.csv')
data2.head()
```

Out[90]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection	TechSupport	Stream
0	5600-PDUJF	Male	0	No	No	6	Yes	No	DSL	No	...	No	Yes	
1	8292-TYSPY	Male	0	No	No	19	Yes	No	DSL	No	...	Yes	Yes	
2	0567-XRHCU	Female	0	Yes	Yes	69	No	No phone service	DSL	Yes	...	Yes	No	
3	1867-BDVFH	Male	0	Yes	Yes	11	Yes	Yes	Fiber optic	No	...	No	No	
4	2067-QYTCF	Female	0	Yes	No	64	Yes	Yes	Fiber optic	No	...	Yes	Yes	

5 rows × 21 columns

In [83]:
```
# You can concatenate this with other DataFrames to get one data set for your work

df = pd.concat([data, data2])
df.to_csv('aba.csv')
```

Check the shapes of the dataframes

In [84]:
```
data.shape
```

Out[84]:
```
(3000, 21)
```

In [85]:
```
data2.shape
```

Out[85]:
```
(2043, 21)
```

In [86]:
```
df.shape
```

Out[86]:
```
(5043, 21)
```