1. Create a Dockerfile that creates a SQL Server Docker container ([Deploy and connect to SQL Server Linux containers - SQL Server | Microsoft Learn](https://learn.microsoft.com/en-us/sql/linux/sql-server-linux-docker-container-deployment?view=sql-server-ver16&pivots=cs1-bash))
2. Write a Python / PowerShell script that creates the following using T-SQL statements:
   1. Create a database.
   2. Create tables with the same schema (the table schema is not relevant); the tables should be created based off the input that is given to the PowerShell/Python script (e.g., input a list of table names and have all the tables created).
   3. The script should insert some data into each table that is created from the earlier step.
   4. Create a SQL login and a database user and grant select permissions on the database created above; both the login and the password should be passed as input at runtime.
   5. Generate a JSON formatted output based off the output of a select statement made on all the created tables (you should export a list of dictionaries as depicted in the sample output below:

[

{

"name": "Alice",

"age": 30,

"city": "New York"

},

{

"name": "Bob",

"age": 25,

"city": "Los Angeles"

},

{

"name": "Charlie",

"age": 22,

"city": "Chicago"

}

]

These are some recommendations for the task at hand:

* Create reusable functions as you think fit.
* The code should be easily readable and commented on.
* Make sure that no open transactions remain active on the database server.

1. Create a multi-stage Azure DevOps YAML Pipeline that checks out the code from your GitHub repository. The pipeline should:
   1. build the Docker image (you do not need to store the image inside an ACR)
   2. runs the previously created script inside a python/PowerShell task using input derived from Pipeline runtime parameters. The password for the newly created login should be stored inside a variable group referenced at runtime.

We do not expect the pipeline to be functional, as it is just a test to verify ADO “coding” knowledge at this level. The PowerShell/Python script should be able to be tested using command line parameters on your local machine.