case study explanation

https://docs.google.com/document/d/1tK-y-dMpHeZ8IYXlwt1xV2WQZ1Fj0vo0NPa9NB0BbSE/edit?usp=sharing

Report Created by: Olaniyan Farouq Olayinka

Date: 23rd November 2024

**TASK 1**

**Task 1** requires me to write a **python script** to process the files provided to me, and also to create a single dataset to perform three (3) analysis and also to generate three (3) outputs.

* The Python script file

Link: <https://drive.google.com/file/d/1ChunWyluezCvnTNFLUq3lV_cU9rtazhe/view?usp=sharing>

* The final dataset

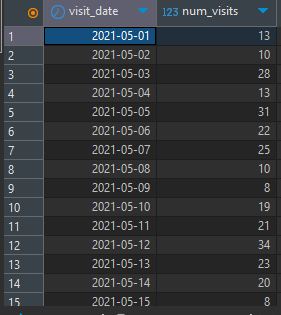
Link: https://drive.google.com/file/d/1je1tlyW8i5xSW2p-5lK0LJxTtlmS7oOr/view?usp=sharing

* The text file containing the queries on the final dataset used to obtain the outlined analyses in the case study pdf file

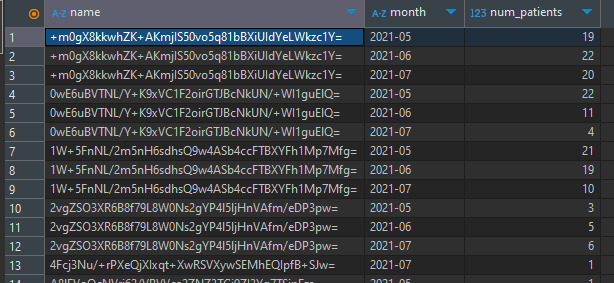
Link: https://drive.google.com/file/d/1ZglaR6J0vIIav8S7wwfkojtXuoE19-2B/view?usp=sharing

Additionally I am attaching the three (3) analyses from my Postgres Database(attaching three(3) screenshots)

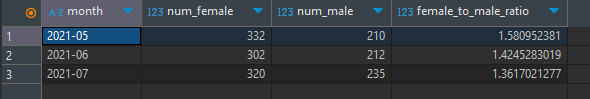
* Number of hospital visits per day over the period



* Number of patients attended to per doctor per month showing their names



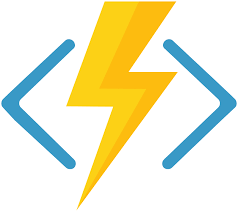
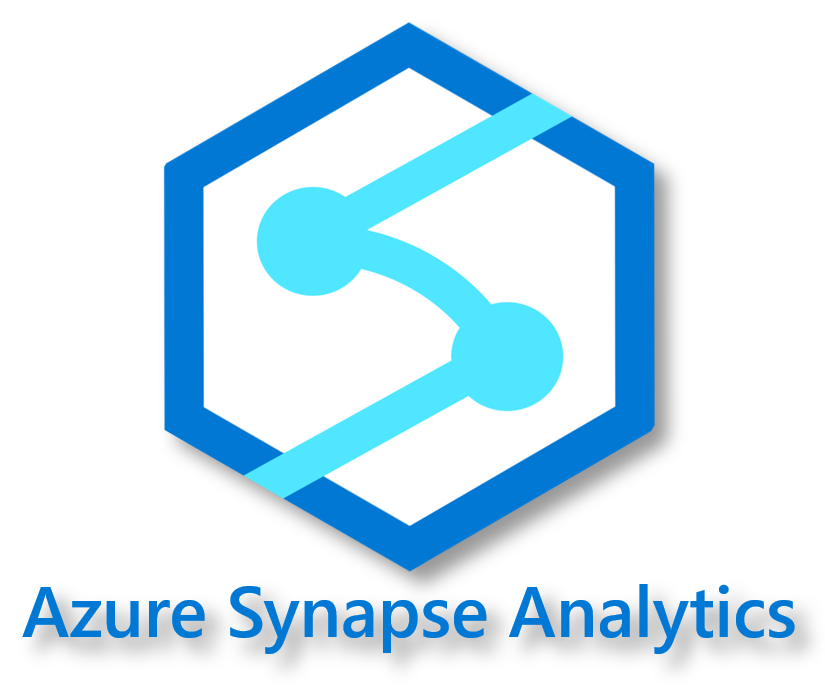
* The ratio of female to male patient visits per month

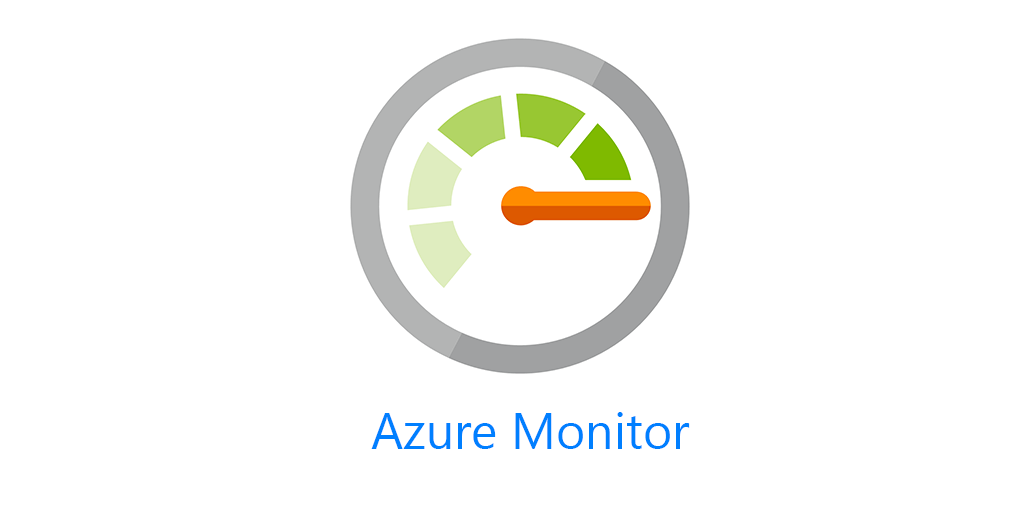


**TASK 2**

**Task 2** requires me to outline a high-level **architecture** to deploy and schedule the python script to run once a day with my **design** reading the data from an **RDBMS** and writing the resulting dataset to a **Data Warehouse**. The **architecture** would be based on **Azure cloud platform**.

Diagram:



Text:

On-Prem RDBMS

|

[Secure Connection: VPN Gateway / ExpressRoute]

|

Azure Data Factory (Orchestrator)

|

Azure Function (Python Script)

|

Azure Storage (Blob) -> Azure Synapse Analytics (Data Warehouse)

|

Azure Monitor (Error Handling & Monitoring)

**Architecture Overview**

1. **On-Prem RDBMS**: I would connect the on-premises database to Azure using **Azure VPN Gateway** or **ExpressRoute** for secure data transfer.
2. **Azure Services**:

* **Azure Data Factory (ADF)**: I would use **ADF** for **orchestrating** the **ETL** workflow, including triggering the Python script and scheduling its execution.
* **Azure Storage (Blob)**: Then store intermediate data files (e.g., CSVs) for backup or staging data.
* **Azure Functions**: Deploy the **Python script** as an **Azure Function** for scalable and serverless execution.
* **Azure SQL Database** (Optional for staging): Although this is optional, I would temporarily stage data before writing to the final Data Warehouse.
* **Azure Synapse Analytics** (Data Warehouse): Finally, I store the final processed dataset for analysis and reporting.

1. **Security and Monitoring**:

* **Azure Key Vault**: This here securely stores database credentials and other sensitive information.
* **Azure Monitor**: I would use this to track the pipeline execution and logs for debugging and monitoring.

**Workflow**

For clarity, I would break down the process into stages for better understanding

1. **Data Extraction**: As I stated earlier, I would use **Azure Data Factory** to connect to the on-prem RDBMS via Integration **Runtime** for secure and seamless access. Afterwards I can extract raw data into **Blob Storage** or pass it directly to the **Azure Function**.
2. **Data Processing**: **Azure Function** runs the **Python script**, processes the data, and writes the output to Blob Storage or Azure SQL Database.
3. **Data Loading**: **Azure Data Factory** would also be used to move the processed dataset from Blob Storage or Azure SQL Database to Azure Synapse Analytics using the “COPY INTO” command.
4. **Scheduling**: Azure Data Factory has a built-in scheduler that would be used to trigger the entire workflow once daily.
5. **Error Handling and Monitoring**: A crucial part, this helps see the progress and to know what stops the process flow. To achieve this, I would configure **Azure Monitor** for logging and alerting in case of failures during the ETL process.

WAREHOUSE DESIGN(Addition)

