PROJECT 4

*TEAM NAME: PACERS*

TEAM MEMBERS:

Megha Varshini M

Aluru Keerthana

Chekka V B Krishna Harshith Babu

Yogitha Prakash

**Problem Statement**

ABC company wants to analyze the data stored into there employee database. They want to calculate yearly additional budget for there employees . Company decides to give the hikes in the salary based on there previous experience. For calculating the total additional budget they want to know the employee count based on there experience which need input file as text file.

We need to develop the solution in Azure using Datafactory to convert the SQL Database employee Table to Blob Storage (employee.txt).

Use employee.txt using spark-shell and find out the count of employees based on experience.

Use spark-shell Create dataframe from employee.txt using filter and map method calculate the total amount.

|  |  |
| --- | --- |
| Employee Experience | Salary Hike % |
| <4 Years Experience | 10 |
| >=4 Years and < 8 Years Experience | 12 |
| >=8 Years Experience | 15 |

Solution Steps:

1. **Go to portal.azure.com using cloud lab credentials**
2. **Create an Azure SQL Database** 
   * In the Microsoft Azure portal, in the menu, click New. Then in the Databases menu, click SQL Database.
   * In the **SQL Database** blade, enter the following settings, and then click **Create**: • **Database name**: employeedb
   * • **Subscription**: *Select your Azure subscription*
   * • **Resource Group**: *Select the resource group you created previously*
   * • **Select source**: Blank database
   * • **Server:** *Create a new server with the following settings*: • **Server name**: *Enter a unique name (and make a note of it!)*
   * • **Server admin login:** *Enter a user name of your choice (and make a note of it!)*
   * • **Password:** *Enter and confirm a strong password (and make a note of it!)*
   * • **Region:** *Select the same location as your storage account*
   * • **Allow azure services to access server**: Selected
   * • **Elastic pool**: *Not enabled*
   * • **Pricing tier:** *View all and select* Basic
   * • **Collation**: SQL\_Latin1\_General\_CP1\_CI\_AS
   * • **Pin to dashboard:** Unselected
   * In the Azure portal, view **Notifications** to verify that deployment has started. Then wait for the SQL database to be deployed (this can take a few minutes.)
   * After the database has been created, browse to your Azure SQL server (not the database) and under **Settings**, click **Properties**.
   * Note the fully qualified name of your server (which should take the form *server*.database.windows.net, where *server* is the server name you specified earlier) and the server admin user name (which should be the login you specified earlier).
   * Click All Resources, and then click your Azure SQL Database.
   * On the database blade, view the Data Explorer page. This opens the web-based query interface for your Azure SQL Database.
   * In the toolbar for the query editor, click Login, and then log into your database using SQL Server authentication and entering the login name and password you specified when provisioning the Azure SQL Database server.
   * In the query editor, enter the following Transact-SQL query to create a table named transactions in your database:
   * CREATE TABLE Employee (

EmpId INTEGER NULL,

EmpName VARCHAR(255) NULL,

EmpBOD VARCHAR(255),

EmpJoiningDate VARCHAR(255),

PrevExperience INTEGER NULL,

Salary INTEGER NULL,

Address VARCHAR(255) NULL

);

* + Click Run to run the Transact-SQL statement.
  + Copy all the insert statements from the file employee.txt in the query editor and click run to run the transact-sql statement



* + Verify the data by running the query select count(\*) from employee in query editor, It should display count as 100

1. **Create a Storage Account and a Blob Container**

The source data for your data pipeline will be stored in an Azure storage account:

* In the Microsoft Azure portal, in the menu, click **New**. Then in the **Storage** menu, click **Storage account**.
* In the **Create storage account** blade, enter the following settings and click **Create**: • **Name**: *Enter a unique name (and make a note of it!)*

**Deployment model**: Resource manager

**Account kind**: General purpose

**Performance**: Standard

* 1. **Replication**: Locally-redundant storage (LRS)
  2. **Storage service encryption**: Disabled
  3. **Subscription**: *Select your Azure subscription*
  4. • **Resource group**: *Create a new resource group with a unique name*
  5. • **Location**: *Select any available region*
  6. • **Name**: adf-data
  7. • **Access type**: Blob (Read Only Access option)
* In the Azure portal, view **Notifications** to verify that deployment has started. Then wait for the storage account to be deployed (this can take a few minutes.)
* After the storage account has been created, browse to its blade in the Azure portal.
* On the blade for your storage account, click **Blobs**, and add a container with the following properties:
* In the Azure portal, view **Notifications** to verify that deployment has started. Then wait for the container to be created (this should take a few seconds.)
* After the container has been created, return to the blade for your storage account, and click **Access keys**. Note that this blade lists the storage account name and two keys that client applications can use for authentication when connecting.

1. **Create an Azure Data Factory**

* In the Microsoft Azure portal, in the menu, click **New**. Then in the **Data + Analytics** menu, click **Data Factory**.
* In the **New data factory** blade, enter the following settings, and then click **Create**: • **Name**: *Enter a unique name (and make a note of it!)*
* **Subscription**: *Select your Azure subscription*
* Version: 1
* **Resource Group**: *Select the resource group you created previously*
* **Location**: *Select the location you specified for your storage account (if it is not available, select any other location)*
* **Pin to dashboard:** Unselected
* In the Azure portal, view **Notifications** to verify that deployment has started. Then wait for the data factory to be deployed (this can take a few minutes.)

1. **Use the Azure Data Factory Copy Wizard to Copy the Data**

* In the Microsoft Azure portal, browse to the blade for your data factory, and click the **Copy data** tile. This opens a new tab in your browser.
* On the **Properties** page of the Copy Data wizard, enter the following details and then click **Next**:
  + **Task name**: Wizard Copy
  + **Task description**: Copy transactions
  + **Task cadence (or) Task schedule**: Run once now
  + **Expiration time**: 3:00:00:00
* On the **Source data store** page, on the **Connect to a Data Store** tab, select **Azure SQL Database**. Then click **Next**.[
* On the **Specify the Azure SQL database** page, enter the following details and then click **Next**: • **Connection name**: sql-database
  + **Server / database selection method**: From Azure subscriptions
  + **Azure subscription**: *Select your subscription*
  + **Server name**: *Select your Azure SQL server*
  + **Database name:** employeedb
  + **User name**: *The server admin login name you specified when creating the database*
  + **Password**: The password for your Azure SQL server admin login
* On the **Table mapping** page, in the **Destination** list, select **[dbo].[transactions]** and click **Next**.
* On the **destination data store** page, on the **Connect to a Data Store** tab, select **Azure Blob Storage**. Then click **Next**.
* On the **Specify the Azure Blob storage account** page, enter the following details and then click **Next**: • **Connection name**: blob-store
* **Account selection method**: From Azure subscriptions
* **Azure subscription**: *Select your subscription*
* **Storage account name**: *Select your storage account*
* On the **Choose the output file or folder** page, double-click the **adf-data** blob container you created previously, and then give the file name as **employee.json**. then click **Next**.
* On the **File format settings** page, • **File format**: json format then click next
* On the **Table mapping** page, ensure that all columns are selected and mapped correctly, and click **Next**:
* On the **Performance settings** page, expand **Advanced settings** to review the default values. Then click **Next**.
* On the **Summary** page, click **Finish**.
* On the **Deploying** page, wait for the deployment to complete.
* Wait a few minutes to allow the pipeline created by the wizard to run.

1. **Verify on Azure portal in Storage Account employee.json file is created**
2. **Download employee.json file and keep in your local drive**
3. **Open the spark-shell and load the employee.json file**
4. **Use the filter function and groupBy method to get the employees as suggested in the requirement.**
5. **Use the map method on various group and calculate the total additional budget**