**Building an end-to-end data pipeline using azure data bricks.**

In this we can be able to know how to mount azure blob storage to azure data bricks.

* The very first step is to create service principal (service account).
* It is an identity created to use applications, hosted services, and tools to access azure resources. This access is restricted by the roles assigned to the service principal, giving you control over which resources can be accessed and at which level.
* For security reasons, its always recommended to use service principals with automated tools rather than allowing them to log in with a user identity.
* We have azure active directory, click on app registrations.
* Give proper name for that service app and register a new one.

**Now we must create secrets.**

* After creation of service app save application id, object id and directory id.
* In the service app we will be having certificates and secrets, client secrets, give proper names for those ids.

**Now we must give access to storage account for the service principal.**

* We will be having access control (IAM) and then add role assignment.
* Now select blob data contributor role, then select members in members tab and there we must select our data bricks service app and click on review & assign.

**Now we must create secrets scope.**

* It helps us to store the credentials securely.
* Now we are going to use azure key vaults and then add our secrets to it, and then we need to create data bricks secret scope and lint it to the azure key vault.
* Then we will be able to use the secrets by using dbutils.secrets.get command.
* Now select our resource group, give a name valid to that key vault and click on create.
* Now we must create secrets of client secrets, client id.

**Now we need to link our data bricks secret scope.**

* Go to data bricks workspace and there we must add secrets/create scope after the # symbol in the URL and click enter.
* Now we have set a name for that scope and create scope.

**Now we must mount azure containers into data bricks.**

* In our workspace we need create our python file mount adls storage.
* Now open our notebook and start our cluster.
* We are using data bricks utilities and python code to mount our containers in data bricks.
* Databricks utilities(dbutils) will connect and parameterize notebooks, and work with secrets.

**Now copy the code below to our python file and execute it.**

* Create code blocks in our data bricks notebook.

# We set the values of our data bricks scope name and azure storage account.

databricks\_scope\_name = "databricks-demo-scope1"

storage\_account\_name = "smgblobstorage"

# We use dbutils to list our secrets name because we need to send them as parameter to the

# dbutils.secrets.get()

dbutils.secrets.list(databricks\_scope\_name)

display(dbutils.secrets.list(databricks\_scope\_name))

# We assign the values of our secrets to variables.

#client\_id = dbutils.secrets.get(databricks\_scope\_name, "")

client\_id = dbutils.secrets.get(scope="databricks-demo-scope1",key="ClientId")

# Directory (Tenant) ID

tanent\_id = dbutils.secrets.get(scope="databricks-demo-scope1",key="TanentId")

# Application (Client) Secret Key

client\_secret = dbutils.secrets.get(scope="databricks-demo-scope1",key="ClientSecret")

# To access to our data in Azure Storage securely we are going to use OAuth 2.0 with Azure Active Directory

# We need to set the following configuration

configs = {

"fs.azure.account.auth.type": "OAuth",

"fs.azure.account.oauth.provider.type": "org.apache.hadoop.fs.azurebfs.oauth2.ClientCredsTokenProvider",

"fs.azure.account.oauth2.client.id": client\_id,

"fs.azure.account.oauth2.client.secret": client\_secret,

"fs.azure.account.oauth2.client.endpoint": "https://login.microsoftonline.com/tanent\_id/oauth2/token"

}

display(configs)

dbutils.fs.mounts()

# dbutisl.fs.mount: Mounts the specified source directory into DBFS at the specified mount point.

def mount\_adls(container\_name):

dbutils.fs.mount(

source = f"wasbs://{container\_name}@{storage\_account\_name}.blob.core.windows.net",

mount\_point = f"/mnt/{storage\_account\_name}/{container\_name}",

extra\_configs = {f"fs.azure.account.key.{storage\_account\_name}.blob.core.windows.net":dbutils.secrets.get(scope = "secretKey", key = "storageKey")}

)

## mount\_adls("smg-source-file")

mount\_adls("smg-blob-parquet")

dbutils.fs.mounts()

## %fs

ls "/mnt/smgblobstorage/smg-source-file"

## %fs

ls "/mnt/smgblobstorage/smg-blob-parquet"

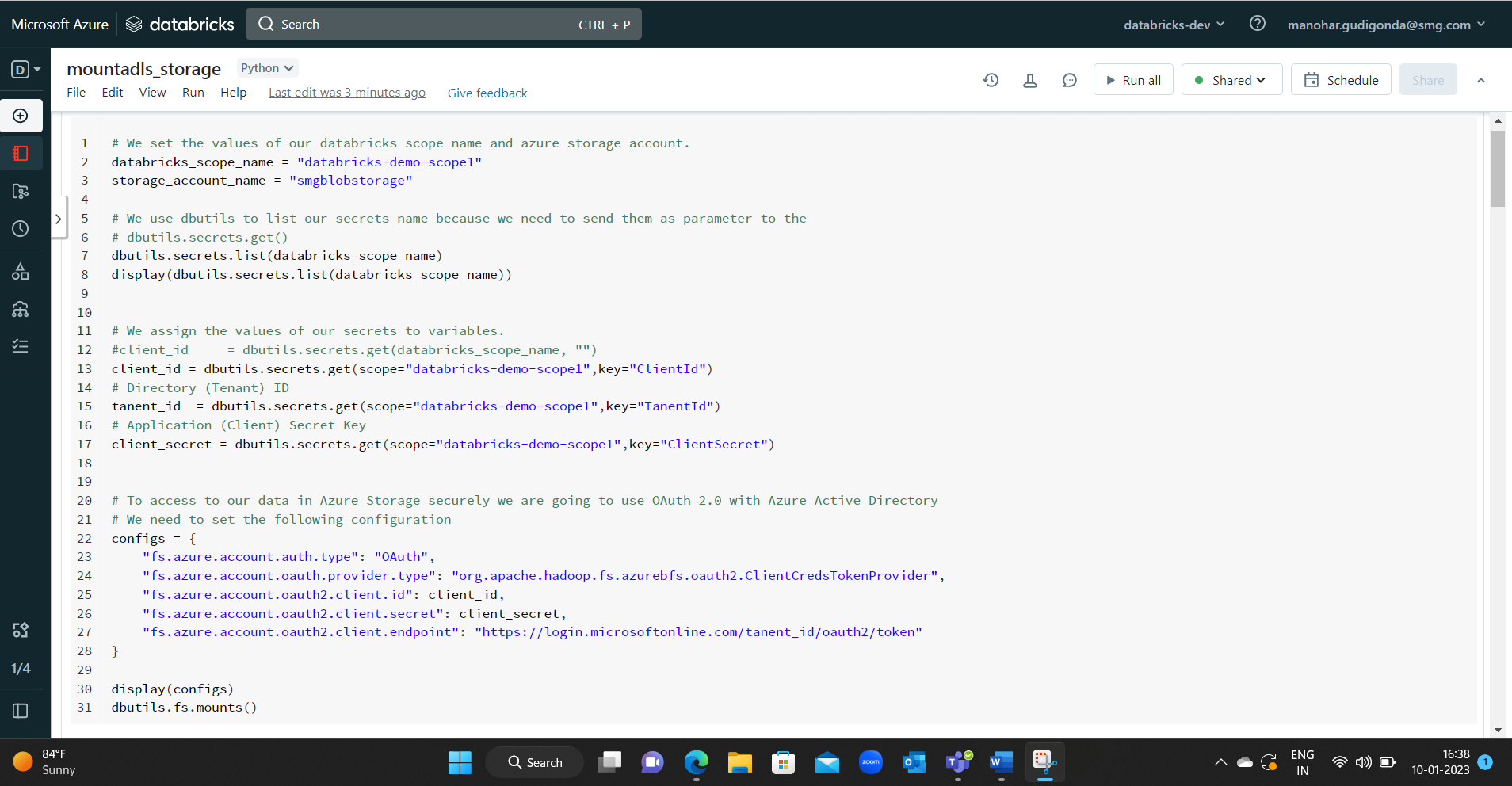
## dbutils.fs.mount(

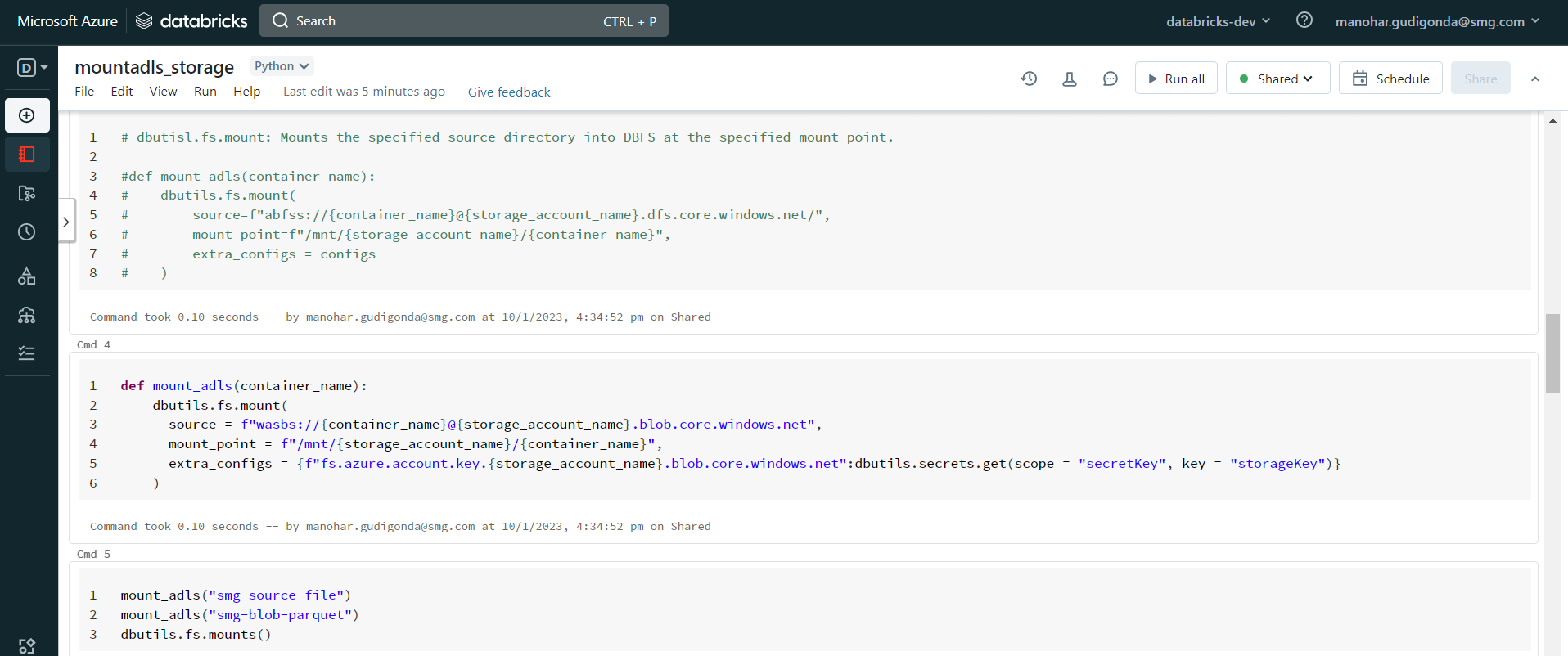
source = f"wasbs://@.blob.core.windows.net",

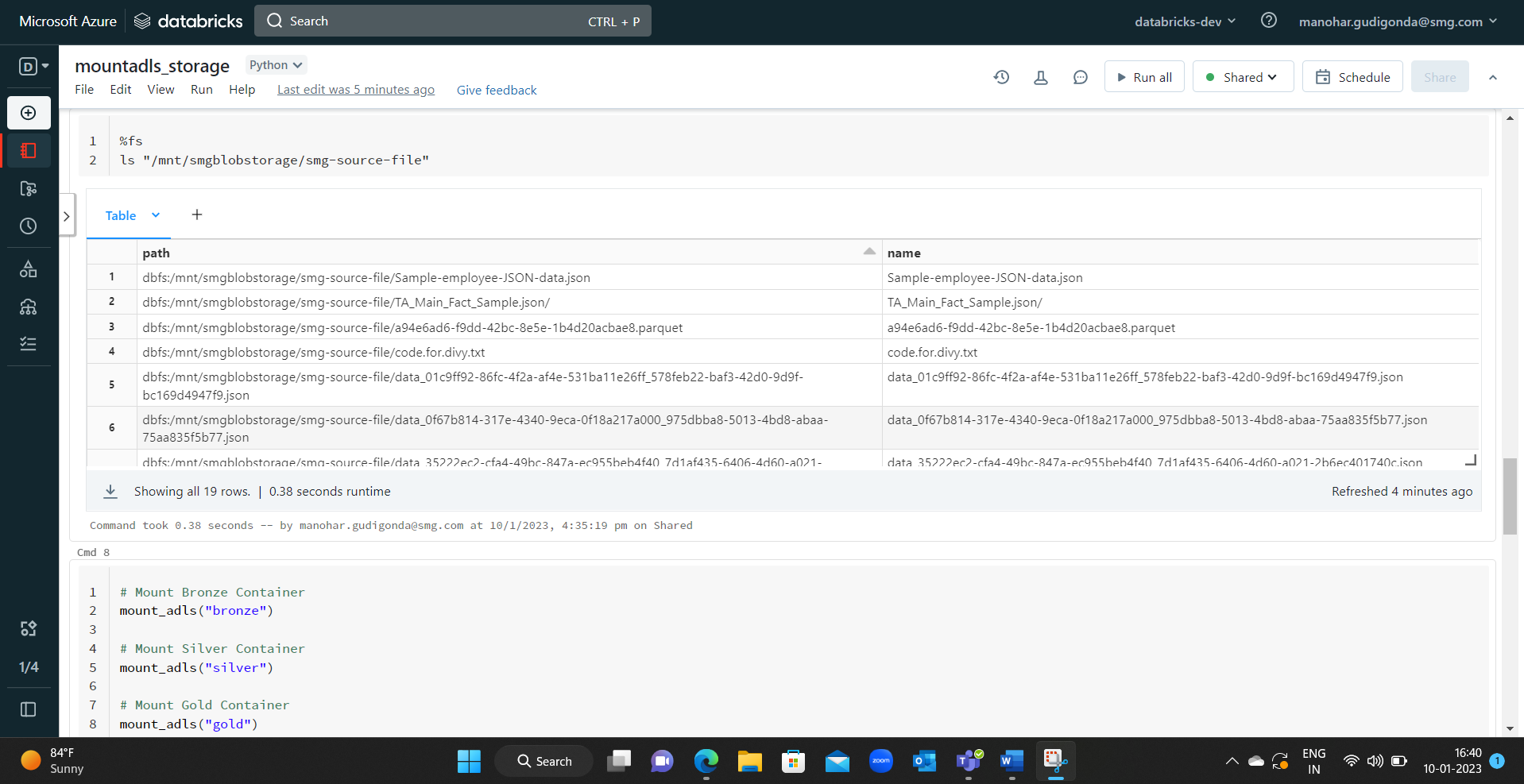
mount\_point = f"/mnt/<mount-name>",

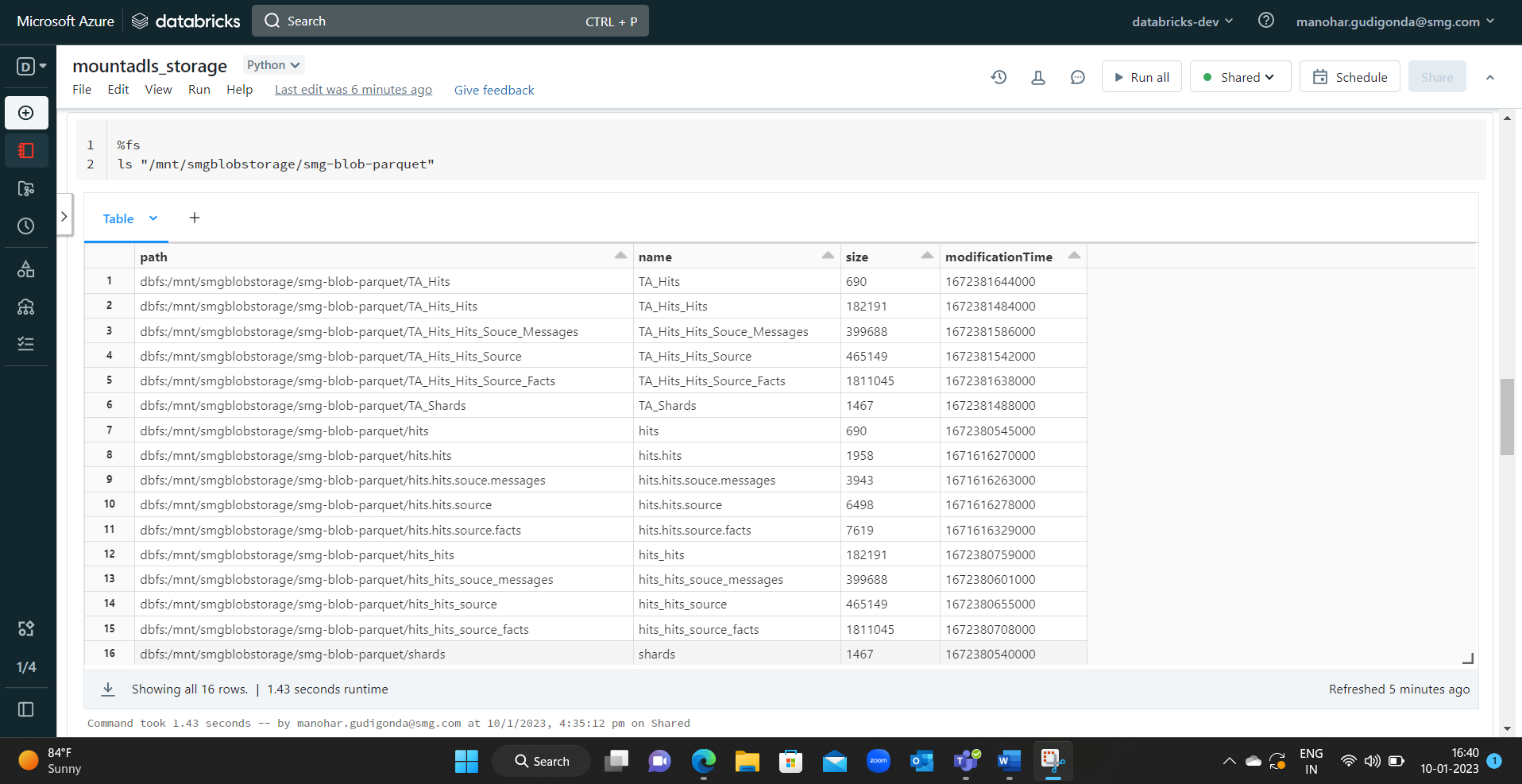
extra\_configs = {"<conf-key>":dbutils.secrets.get(scope = "secretKey", key = "storageKey")})

## dbutils.fs.mkdirs("/mnt/smgblobstorage/bronze/silver")

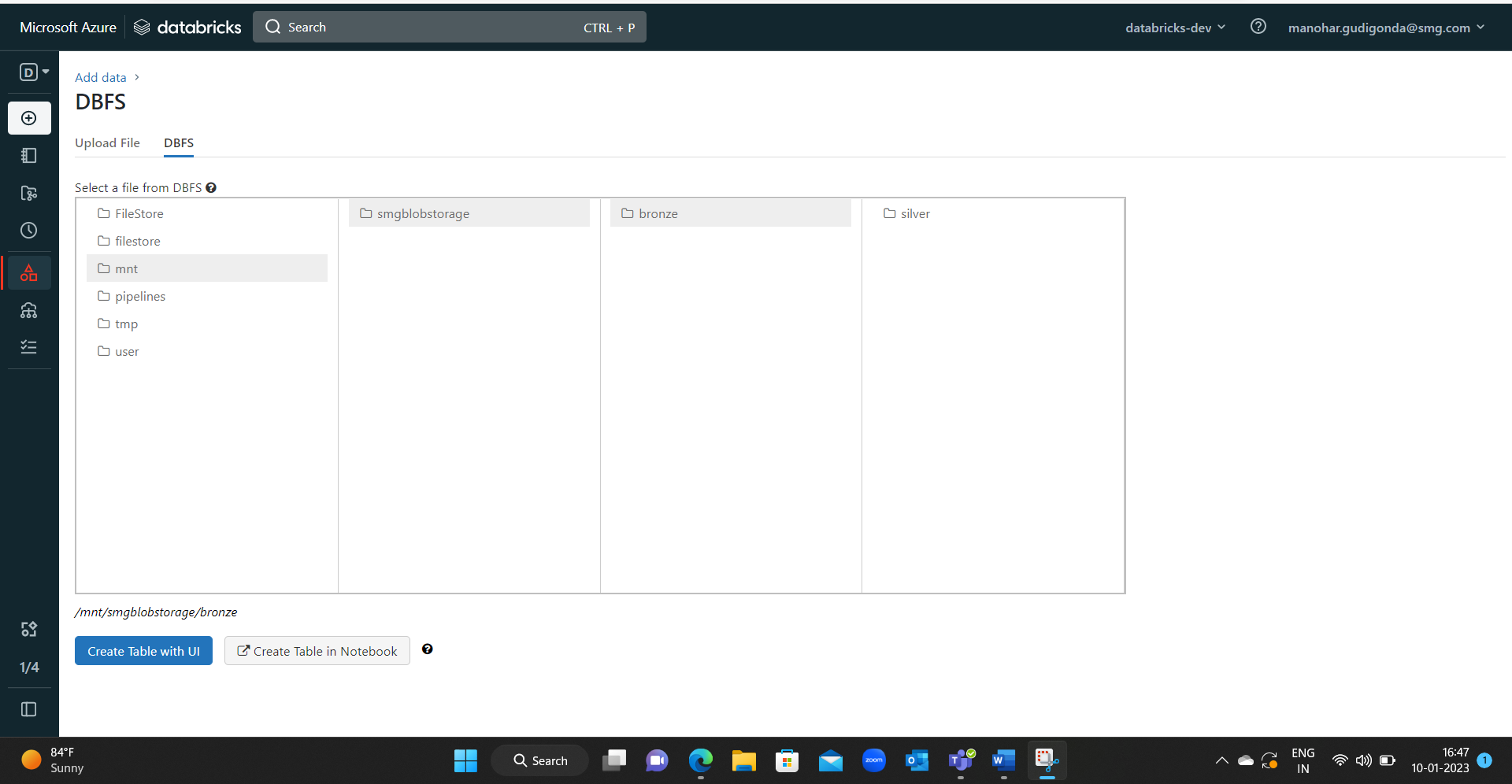












------------------------------------------------------THE END--------------------------------------------------------------------THANK YOU

G MANOHAR