AZURE RESOURSE GROUP
Create
Name= ecomm-live
Region=(asia pacific) central india

FILES.IO FOR DATABASE
Create database
Mysql
Database identifier = olistproject

MongoDB
Database identifier = olistproject

COLAB WEBSITE
We upload olist_order_payment_database.csv
By the using code we upload this file to online database files.io

CREATE DATA FACTORY
Basics-

- Resource group = ecomm-live
- Name=olist-ecomm-data-factory
- Region=central india
- Version =v2

Go to data factory

- Create pipeline-
- Name=data ingestion pipeline
- Copy data activity
- Name=datafromgithub

Sourse-
New dataset = http
Format = csv
Name= GithubDataCSV
Linked service=
New
Name= github link
Base URL =

- go to github
- Select desired csv
- Click row
- Copy the url
- Pase here

Authentication type= anonymous

Test connection
Create
First row is header =tick
Import schema = none

Sink-
New
New dataset = azure data lake storage zen2
Format = csv
Linked service =

- add new
- name= data lake
- subscription = default
- storage = default
- create

file path =
brouse
olistdata-bronze-olist-customer
first row hadder=tick
schema=none
ok

we have to create a storage account (adls data lake)
search storage account
create
resource group= ecomm-live
storage account name= olist data storage account
region= central india
redundancy= locally-redundecy-storage(lrs)
next
hierarchical name space = tick
next
create

go to storage account
container
create a container
name= olistdata
create

inside container
add directory
name= bronze
add directory
name=silver

add directory
name=gold

GO TO ADF- monitor-linked service
Delete both linked service
Create a new linked service
Data storage=http
Name=httpgithublinkedservice
Base url=https://github.com/7798akash/
Authentication type= anonymous
Create

CREATE A NEW LINKED SERVICE
Data store= mysql
Name=filessqldb
Server name=go to files.io-mysql-hostname-copy-paste here
Port=3307
Database name= files.io-go to mysql-database-copy name-paste here
User name= files.io-mysql-user-copy-paste here
Password=from files.io

NOW GO TO PIPELINE-
Delete old copy data activity
Drag and drop new copy data activity
Name=copy data test
Source=
http
csv
name=datafromgithubvialinkedservice
linked service=select our service
relative url= blank
first row hadder= tick
import source =none
ok
source = open
relative url= add dynamic content
parameter = new
name=csv_relative_url
ok

sink=
adls zen 2
csv
name=csvfromlinkedservicetosink
new=

name=adlsforcsv
create
file path=bronze
ok
import scheme=none

click open
file name
add dynamic content
name=file_name
ok

drag and drop for each activity
setting-sequential=tick
items=add dynamic content
for each activity
ok

CLICK ANYWHERE IN THE PIPELINE BORD
-PARAMETER
Name-
Foreachinput

Type-
Array

```
default value-          [
        {
        "csv_relative_url": "BigDataProjects/refs/heads/main/Project-
Brazillian%20Ecommerce/Data/olist_customers_dataset.csv",
        "file_name":"olist_customers_dataset.csv"
        },
        {
        "csv_relative_url": "BigDataProjects/refs/heads/main/Project-
Brazillian%20Ecommerce/Data/olist_geolocation_dataset.csv",
        "file_name":"olist_geolocation_dataset.csv"
        },
        {
        "csv_relative_url": "BigDataProjects/refs/heads/main/Project-
Brazillian%20Ecommerce/Data/olist_order_items_dataset.csv",
        "file_name":"olist_order_items_dataset.csv"
        },
        {
```

```
	"csv_relative_url": "BigDataProjects/refs/heads/main/Project-
Brazillian%20Ecommerce/Data/olist_order_reviews_dataset.csv",
	"file_name":"olist_order_reviews_dataset.csv"
	},
	{
	"csv_relative_url": "BigDataProjects/refs/heads/main/Project-
Brazillian%20Ecommerce/Data/olist_orders_dataset.csv",
	"file_name":"olist_orders_dataset.csv"
	},
	{
	"csv_relative_url": "BigDataProjects/refs/heads/main/Project-
Brazillian%20Ecommerce/Data/olist_products_dataset.csv",
	"file_name":"olist_products_dataset.csv"
	},
	{
	"csv_relative_url": "BigDataProjects/refs/heads/main/Project-
Brazillian%20Ecommerce/Data/olist_sellers_dataset.csv",
	"file_name":"olist_sellers_dataset.csv"
	}
]
```

Inside for each
Drag and frop copy data activity
Source
Source dataset = data from github via linked service
Csv_related_url=add dynamic content-forEach1-@item().csv_relative_url

Sink
Csv from linked service to sink
File_name=add dynamic content
@item().file_name

FOREACH- COPY DATA
Source-
new dataset
Mysql
Linked service=filesssqldb
Table move=olist_order_payment
Ok

Sink-
New dataset
Adls zen2
Csv
Name= sql to adls

Linked service –
New
Name-sql to adls linked service
Create
File path
Olistdata-bronze
Name-olist-order-payment-dataset.csv
First row hadder= tick
Import schema = none
Ok

LOOKUP ACTIVITY
Name= lookup for each input
Setting =
source dataset=
new
http
json
linked service =
new
name=json from github for loop
base url= copy and past from github
ok
first row only= untick

IN FOREACH ACTIVITY
Setting =
Items= delete previous data
Add dynamic content
Lookup for each input
@activity('lookup for each input').output.value

CLICK ANYWHERE IN THE PIPELINE BORD
Delete your parameter
Delete all data from bronze
Now debug

AZURE DATABRICKS
Create
Resource group= ecomm-live
Workspace name= olist-spark-warkspace
Region= central india
Pricing tier= premium

Managed resource group name= ecomm-databricks-resource-group
Create

IN DATABRICKS
Compute=
Create compute
Name=first compute
Policy=unrestricted
Single node
Standard_d4ds_v5   16gb, 4core
Termination after = 20 mint
Create

GO TO files.io
Go to your mongodb
Code=copy python code

Go to colab = paste that code
Run
If fail
=pip install pymongo
Now run again it will work

Upload csv file to colab
Product_category_name_translation.csv
Generate your code using –
Read the product_category csv and create a collectiona dn upload it to above mangodb

AZURE
Search= app registrations
New registration
Name= olist-app-registration-db-adls
Register

OPEN APP REGISTRATION TO THAT REGISTRATION AND GO TO
Certificate and secrate
Description= db-client-secret
Add

OPEN A TUTORIAL PAGE TO CONNECT
 Databricks to azure
Go to
https://learn.microsoft.com/en-us/azure/databricks/connect/storage/tutorial-azure-storage

copy the code

go to azure databricks
new notebook
paste the code

SEARCH APP REGISTRATIONS ON MICROSOFT AZURE PORTAL
New registration
Name=olist-app-registration-db-adls

Now go to certificate & secrets in app registrations
New client secrates
Description= db-client-secret
Add

Now go to databricks and generate modify code by using AI this code is which we are previously code from learn.microsoft.com
The prompt is=
Change the below code to have storage account and other key id as a variable outside
And give values and key to this code
Storage account =”olistdatastorageak”
Application id= app registration vali
Directory id= app registration vali
Service credential = secrate and app registration vala

Now go to storage account
Access control (iam) = add
Add role assignment
Job function role= storage blob Data contribution
Click next
Assign access to = user, group or service principle
Member = add
select mumbers = olist-app-registration-db-adls
review+assign

now go to databrics

this is a boilerplate code to connect data storage to databrics-

```
storage_account = "olistdatastorageak"
application_id = "382f442b-90f5-4d04-b80e-dfc89b893dc7"
directory_id = "08aac124-05be-4957-a467-de198eab2803"

spark.conf.set(f"fs.azure.account.auth.type.{storage_account}.dfs.core.windows.net", "OAuth")
spark.conf.set(f"fs.azure.account.oauth.provider.type.{storage_account}.dfs.core.windows.net",
"org.apache.hadoop.fs.azurebfs.oauth2.ClientCredsTokenProvider")
```

```
spark.conf.set(f"fs.azure.account.oauth2.client.id.{storage_account}.dfs.core.windows.net",
application_id)
spark.conf.set(f"fs.azure.account.oauth2.client.secret.{storage_account}.dfs.core.windows.net",
"OtL8Q~DdJxJPSlvqdqdcFOj4-A_YBLV1WWbIFbLT")
spark.conf.set(f"fs.azure.account.oauth2.client.endpoint.{storage_account}.dfs.core.windows.net",
f"https://login.microsoftonline.com/{directory_id}/oauth2/token")
```

generate the code by using this prompt
-give me a code to read a ADLS gen2 blob as a csv spark dataframe
Code=

```
file_path =
"abfss://olistdata@olistdatastorageak.dfs.core.windows.net/bronze/olist_customers_dataset.csv"

customers_df = spark.read.format("csv").option("header", "true").option("inferSchema",
"true").load(file_path)
display(df)
```

MODIFY YOUR CODE TO EXTRACT DATA FROM BRONZE TABLE

```
file_path =
"abfss://olistdata@olistdatastorageak.dfs.core.windows.net/bronze/olist_orders_dataset.csv"
orders_df = spark.read.format("csv").option("header", "true").option("inferSchema",
"true").load(file_path)

file_path =
"abfss://olistdata@olistdatastorageak.dfs.core.windows.net/bronze/olist_order_payments_dataset.cs
v"
payments_df = spark.read.format("csv").option("header", "true").option("inferSchema",
"true").load(file_path)

file_path =
"abfss://olistdata@olistdatastorageak.dfs.core.windows.net/bronze/olist_order_reviews_dataset.csv"
reviews_df = spark.read.format("csv").option("header", "true").option("inferSchema",
"true").load(file_path)

file_path =
"abfss://olistdata@olistdatastorageak.dfs.core.windows.net/bronze/olist_order_items_dataset.csv"
items_df = spark.read.format("csv").option("header", "true").option("inferSchema",
"true").load(file_path)

file_path =
"abfss://olistdata@olistdatastorageak.dfs.core.windows.net/bronze/olist_sellers_dataset.csv"
sellers_df = spark.read.format("csv").option("header", "true").option("inferSchema",
"true").load(file_path)
```

file_path =
"abfss://olistdata@olistdatastorageak.dfs.core.windows.net/bronze/olist_products_dataset.csv"
products_df = spark.read.format("csv").option("header", "true").option("inferSchema",
"true").load(file_path)

AFTER THAT GO TO COMPUTE –
Libraries –
install new
pypi package=pymongo
import pymongo

in databricks
-from pymongo import MongoClient

go files.io – mongodb
code- copy code
in databricks
paste the code
# importing module
from pymongo import MongoClient

hostname = "h24a8.h.filess.io"
database = "olistprojectNoSql_scalesungo"
port = "27018"
username = "olistprojectNoSql_scalesungo"
password = "b25572b0a82752b45f58cc29c243bff052203170"

uri = "mongodb://" + username + ":" + password + "@" + hostname + ":" + port + "/" + database

# Connect with the portnumber and host
client = MongoClient(uri)

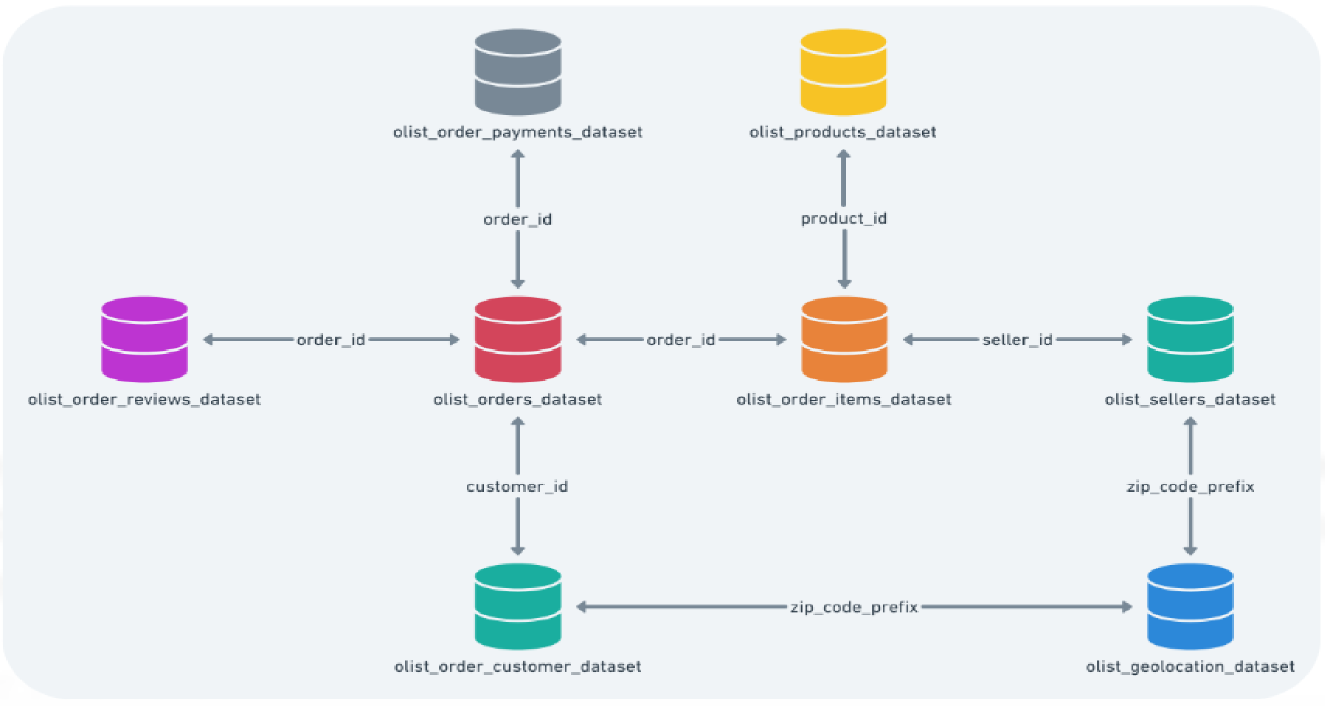
# Access database
mydatabase = client[database]

in databricks
import pandas as pd
collection = mydatabase['product_categories']
mongo_data=pd.DataFrame(list(collection.find()))

display(products_df)

mongo_data

```
from pyspark.sql.functions import col,to_date,datediff,current_date

def clean_datafram(df, name):
   print("cleaning"+name)
   return df.dropDuplicates().na.drop('all')
orders_df=clean_datafram(orders_df,"orders")
display(orders_df)



#convert date column
orders_df = orders_df.withColumn("order_purchase_timestamp",
to_date(col("order_purchase_timestamp")))\
                  .withColumn("order_delivered_customner_date",
to_date(col("order_delivered_customer_date")))\
                     .withColumn("order_estimated_delivery_date",
to_date(col("order_estimated_delivery_date")))



# calculate delivery and time delays

from pyspark.sql.functions import datediff, when

orders_df=orders_df.withColumn("actual_delivery_time",
datediff("order_delivered_customner_date","order_purchase_timestamp"))
orders_df=orders_df.withColumn("estimated_delivery_time",
datediff("order_estimated_delivery_date","order_purchase_timestamp"))
orders_df=orders_df.withColumn("delay time",col("actual_delivery_time")-
col("estimated_delivery_time"))

display(orders_df)
```



JOINING

```
orders_customer_df=orders_df.join(customers_df,orders_df.customer_id==customers_df.customer_id,"left")
orders_payments_df=orders_customer_df.join(payments_df,orders_customer_df.order_id==payments_df.order_id,"left")
order_items_df=orders_payments_df.join(items_df,"order_id","left")
order_items_products_df=order_items_df.join(products_df,order_items_df.product_id==products_df.product_id,"left")
final_df=order_items_products_df.join(sellers_df,order_items_products_df.seller_id==sellers_df.seller_id,"left")
```

```
display(final_df)
```

```
mongo_data.drop('_id',axis=1,inplace=True)
mongo_spark_df=spark.createDataFrame(mongo_data)
display(mongo_spark_df)
```

```
final_df=final_df.join(mongo_spark_df,"product_category_name","left")
```

```
def remove_duplicate_columns(df):
    columns = df.columns
    seen_columns = set()
    columns_to_drop = []

    for column in columns:
        if column in seen_columns:
            columns_to_drop.append(column)
        else:
            seen_columns.add(column)
    df_cleaned=df.drop(*columns_to_drop)
    return df_cleaned
final_df=remove_duplicate_columns(final_df)
display(final_df)
```

```
final_df.write.mode("overwrite").parquet("abfss://olistdata@olistdatastorageak.dfs.core.windows.net/silver")
```

SEARCH AZURE SYNAPSE:
 Create synapse workspace
Resourse group = ecomm-live
Manage resource group =synapse-workspace-olist-rg
Workspace name= olist-synapse-workspace
Region=central india
Account name=synapse storage default olist
File system name= synapse olist fs
Next
Sql server admin login= olist sql admin
Sql passwaord=_______
Next
Create

Go to storage account = olistdatastorageaccount
Access control(iam)
Add new
Role assignment
Search = storage blob data contributer
Next
Assign access to
Managed identity
new
Select member
Managed identity

Synapse workspace
Select
Olist-synapse
Assign access to user,group,our service principle
Select (your name)
Review and assign

Open synapse analytics

Go to database(2nd option)
Add new sql database
Select sql pool type- serverless
Database- olist

Go to develop(3rd option)
Add sql script
Name= sql on olist data

```
select * from
OPENROWSET(
   BULK'https://olistdatastorageak.blob.core.windows.net/olistdata/silver/',
   FORMAT='parquet'
)
```

Add sqlscript
Name= create view

```
Create schema gold
Create view gold.final
As
select * from
OPENROWSET(
   BULK'https://olistdatastorageak.blob.core.windows.net/olistdata/silver/',
   FORMAT='parquet'
) as result
Select * from gold.final
```

Add sqlscript
Name=view final2

```
Create view gold.final2
As
select * from
OPENROWSET(
   BULK'https://olistdatastorageak.blob.core.windows.net/olistdata/silver/',
   FORMAT='parquet'
) as result2
Where order_status = ‘delivered’

Select * from gold.final2
```

Add sqlscript
Name=sql to gold layer

```
-- create master key ENCRYPTION by PASSWORD = 'Akash@1234';
-- CREATE DATABASE SCOPED CREDENTIAL akashadmin with IDENTITY = 'Managed Identity';
-- SELECT * from sys.database_credentials

CREATE EXTERNAL FILE FORMAT extfileformat WITH (
   FORMAT_TYPE = PARQUET,
   DATA_COMPRESSION = 'org.apache.hadoop.io.compress.SnappyCodec'
);

CREATE EXTERNAL DATA SOURCE goldlayer WITH (
   LOCATION = 'https://olistdatastorageak.dfs.core.windows.net/olistdata/gold/',
   CREDENTIAL = akashadmin
);

CREATE EXTERNAL TABLE gold.finaltable WITH (
      LOCATION = 'Serving',
      DATA_SOURCE = goldlayer,
      FILE_FORMAT = extfileformat
) AS
SELECT * FROM gold.final2;


select * from gold.finaltable

select * from gold.final2
```