------ 1. creating the warehouse --------

Create Warehouse CAPSTONEPRO\_WH with

Warehouse\_size = 'xsmall'

Auto\_suspend = 300

Auto\_resume = true

Initially\_suspended = true;

USE WAREHOUSE CAPSTONEPRO\_WH;

show databases;

------ 2. creating the databases --------------

create or replace database RAWDATA\_DB;

create or replace database CLEANSEDATA\_DB;

use database RAWDATA\_DB;

------ 3. creating the schema -----------------

create schema RAWDATA\_DB.RAWDATA\_SCHEMA;

------ 4. create external storage integration --------

CREATE OR REPLACE STORAGE INTEGRATION mycap\_csv

TYPE = EXTERNAL\_STAGE

STORAGE\_PROVIDER = S3

ENABLED = TRUE

STORAGE\_AWS\_ROLE\_ARN ='arn:aws:iam::339713157319:role/mycap\_role'

STORAGE\_ALLOWED\_LOCATIONS =('s3://captrail3/capfolder3/');

desc integration mycap\_csv;

------ 5. Create external stage ------------------

CREATE OR REPLACE STAGE my\_ext\_cap

STORAGE\_INTEGRATION = mycap\_csv

URL = 's3://captrail3/capfolder3/';

------ 6. creating the tables ---------------------

CREATE TABLE raw\_transactions (

transaction\_id STRING,

customer\_id STRING,

transaction\_date TIMESTAMP\_NTZ,

amount FLOAT,

currency STRING,

transaction\_type STRING,

channel STRING,

merchant\_name STRING,

merchant\_category STRING,

location\_country STRING,

location\_city STRING,

is\_flagged BOOLEAN

);

CREATE TABLE raw\_customers (

customer\_id STRING,

first\_name STRING,

last\_name STRING,

date\_of\_birth DATE,

gender STRING,

email STRING,

phone\_number STRING,

address STRING,

city STRING,

country STRING,

occupation STRING,

income\_bracket STRING,

customer\_since DATE

);

CREATE TABLE raw\_accounts (

account\_id STRING,

customer\_id STRING,

account\_type STRING,

account\_status STRING,

open\_date DATE,

current\_balance FLOAT,

currency STRING,

credit\_limit FLOAT

);

CREATE TABLE raw\_credit\_data (

customer\_id STRING,

credit\_score INT,

number\_of\_credit\_accounts INT,

total\_credit\_limit FLOAT,

total\_credit\_used FLOAT,

number\_of\_late\_payments INT,

bankruptcies INT

);

CREATE TABLE raw\_watchlist (

entity\_id STRING,

entity\_name STRING,

entity\_type STRING,

risk\_category STRING,

listed\_date DATE,

source STRING

);

----- 7. Create file formats ----------

CREATE OR REPLACE FILE FORMAT csv\_raw\_data

TYPE = 'CSV'

FIELD\_OPTIONALLY\_ENCLOSED\_BY = '"'

FIELD\_DELIMITER = ','

SKIP\_HEADER = 1;

----- 8. creating the pipes ------------

CREATE OR REPLACE PIPE transactions\_pipe

auto\_ingest = true AS

COPY INTO raw\_transactions

FROM @my\_ext\_cap

FILE\_FORMAT = (FORMAT\_NAME = 'RAWDATA\_DB.RAWDATA\_SCHEMA.csv\_raw\_data')

ON\_ERROR = CONTINUE;

show pipes;

desc pipe transactions\_pipe;

CREATE OR REPLACE PIPE customers\_pipe

auto\_ingest = true AS

COPY INTO raw\_customers

FROM @my\_ext\_cap

FILE\_FORMAT = (FORMAT\_NAME = 'RAWDATA\_DB.RAWDATA\_SCHEMA.csv\_raw\_data')

ON\_ERROR = CONTINUE;

CREATE OR REPLACE PIPE accounts\_pipe

auto\_ingest = true AS

COPY INTO raw\_accounts

FROM @my\_ext\_cap

FILE\_FORMAT = (FORMAT\_NAME = 'RAWDATA\_DB.RAWDATA\_SCHEMA.csv\_raw\_data')

ON\_ERROR = CONTINUE;

CREATE OR REPLACE PIPE credit\_data\_pipe

auto\_ingest = true AS

COPY INTO raw\_credit\_data

FROM @my\_ext\_cap

FILE\_FORMAT = (FORMAT\_NAME = 'RAWDATA\_DB.RAWDATA\_SCHEMA.csv\_raw\_data')

ON\_ERROR = CONTINUE;

CREATE OR REPLACE PIPE watchlist\_pipe

auto\_ingest = true AS

COPY INTO raw\_watchlist

FROM @my\_ext\_cap

FILE\_FORMAT = (FORMAT\_NAME = 'RAWDATA\_DB.RAWDATA\_SCHEMA.csv\_raw\_data')

ON\_ERROR = CONTINUE;

-------9. to refresh pipes ---------

alter pipe transactions\_pipe refresh;

alter pipe customers\_pipe refresh;

alter pipe accounts\_pipe refresh;

alter pipe credit\_data\_pipe refresh;

alter pipe watchlist\_pipe refresh;

select \* from raw\_transactions;

select \* from raw\_customers;

select \* from raw\_accounts;

select \* from raw\_credit\_data;

select \* from raw\_watchlist;

e4

---create udf

create or replace function trans\_func(amount number)

returns string

language sql

as

$$

select case when amount > 500 then 'High'

when amount between 100 and 200 then 'Medium'

else 'Low' end

$$

;

grant USAGE on FUNCTION trans\_func(NUMBER) to role PC\_DBT\_ROLE;

--select \* from transaction ;

--select \*, trans\_func(amount) as risk\_level from transaction;

REVOKE APPLYBUDGET ON DATABASE RAWDATA\_db FROM ROLE PC\_DBT\_ROLE;

grant all privileges on DATABASE RAWDATA\_db to role PC\_DBT\_ROLE;

grant all privileges on schema RAWDATA\_schema to role PC\_DBT\_ROLE;

grant select on all tables in schema RAWDATA\_schema to role PC\_DBT\_ROLE;

GRANT SELECT ON FUTURE TABLES IN DATABASE RAWDATA\_db TO ROLE PC\_DBT\_ROLE;

select \* from TRANS\_FUNC;