Capstone 1 Project

Presentation

Presented by **Ahmed Alkesayer** ,**Rahaf mohammed**, and **Haneen Alghamdi**

Agenda

Capstone 1 Project

01 Project Objects

02 Data Overview

03 Project Steps

04 Project Achievement

05 Future

WeCloudData

Project Objectives

- Project Requirements

Tasks include data gathering, storage, transformation, and preparation for BI.

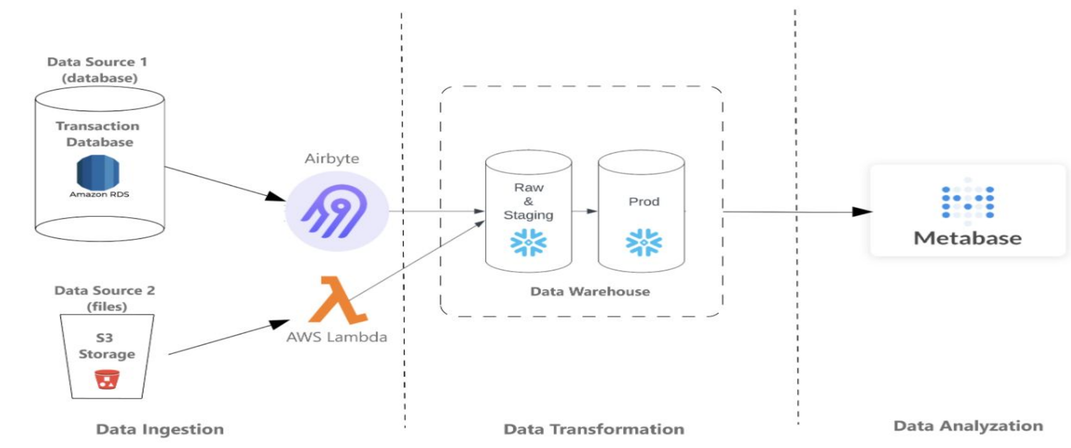
- Tools Used

AWS S3, AWS RDS, Lambda, Airbyte, EC2, Docker, Snowflake, Metabase.

Project Objectives

- Project Process Design

Structured steps: requirements, data model, infrastructure, data processing, dashboards.

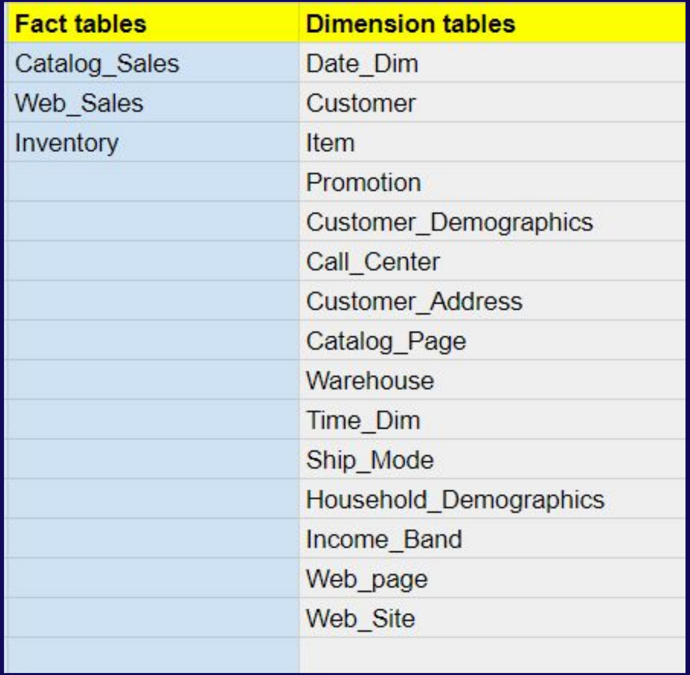


- Project Work Split

Independent work: everyone involved in all phases, with open communication.

Data Overview

- The Tables Included
- Data Cleansing and Processing
- Continuous Data Loading vs. One-time Loading



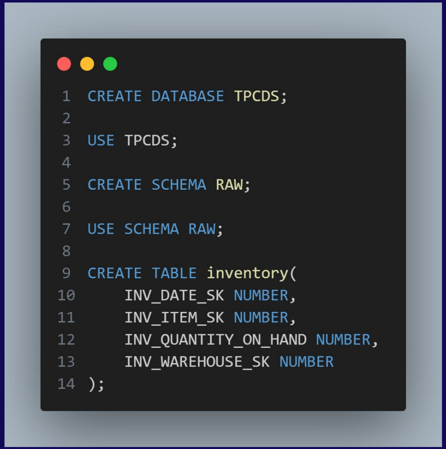
Fact tables	Dimension tables
Catalog_Sales	Date_Dim
Web_Sales	Customer
Inventory	Item
	Promotion
	Customer_Demographics
	Call_Center
	Customer_Address
	Catalog_Page
	Warehouse
	Time_Dim
	Ship_Mode
	Household_Demographics
	Income_Band
	Web_page
	Web_Site

Project Steps

- The processes of making our project is entirely composed of the whole ETL process for us as a data engineers and it will be explained in the coming slides

Project Steps: Data Infrastructure

[Week 1, Day 1]
Creating the database TPCDS, and within it is the RAW schema which will store the ingested data from extraction, and within the schema will be the creation of the Inventory Table that would later be used to get the data from the RDS source.



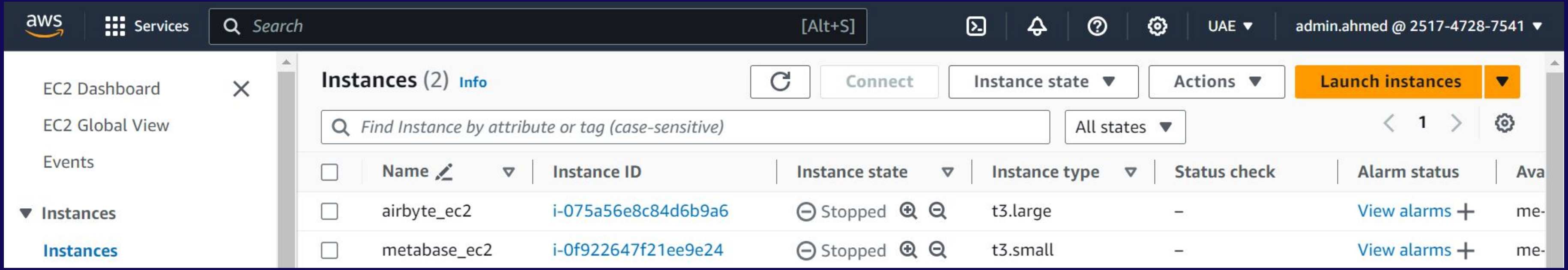
```sql
CREATE DATABASE TPCDS;

USE TPCDS;

CREATE SCHEMA RAW;

USE SCHEMA RAW;

CREATE TABLE inventory(
    INV_DATE_SK NUMBER,
    INV_ITEM_SK NUMBER,
    INV_QUANTITY_ON_HAND NUMBER,
    INV_WAREHOUSE_SK NUMBER
);
```

Project Steps: Data Infrastructure

[Week 1, Day 1]

Now, Into Configuring the EC2 instances for both **Airbyte** *(t3.large)* and **Metabase** *(t3.small)* as both are tools that are needed to be deployed on cloud for data processing (Airbyte) and production (Metabase).

Also installing + configuring **Docker** to have the ability to use these tools.



□	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Ava
□	airbyte_ec2	i-075a56e8c84d6b9a6	⊖ Stopped	t3.large	–	View alarms +	me-
□	metabase_ec2	i-0f922647f21ee9e24	⊖ Stopped	t3.small	–	View alarms +	me-

WeCloudData

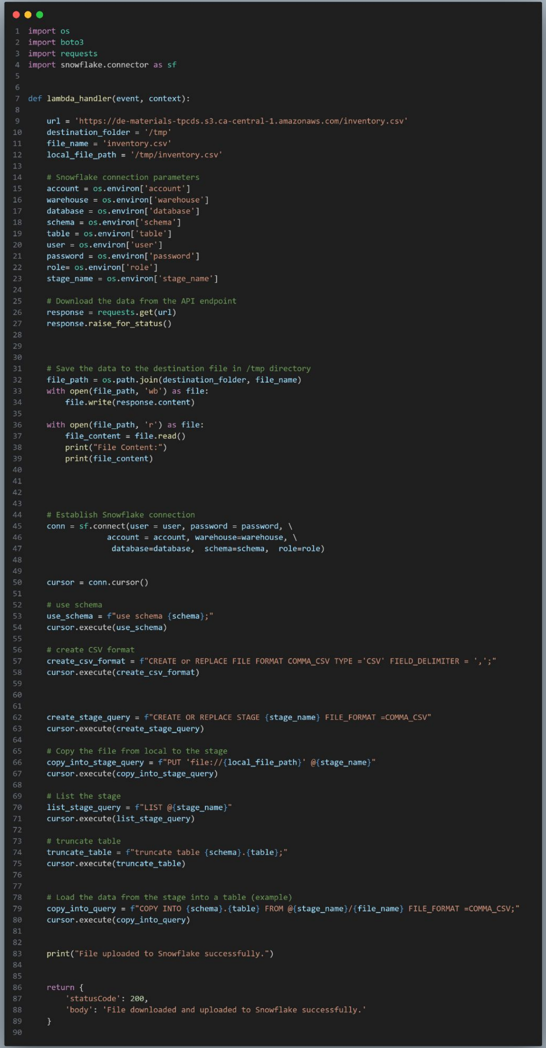
Project Steps: Data Infrastructure

[Week 1, Day 2]
For this task, the main goal was to create ***AWS Lambda*** to extract the inventory table data from the **AWS S3** bucket that has already been provided by WCD and the AWS Lambda is triggered by the ***AWS EventBridge***, the loading destination will be the Snowflake Warehouse where the TCPDS database is, and inside the RAW schema.

Cont’d

[Week 1, Day 2]

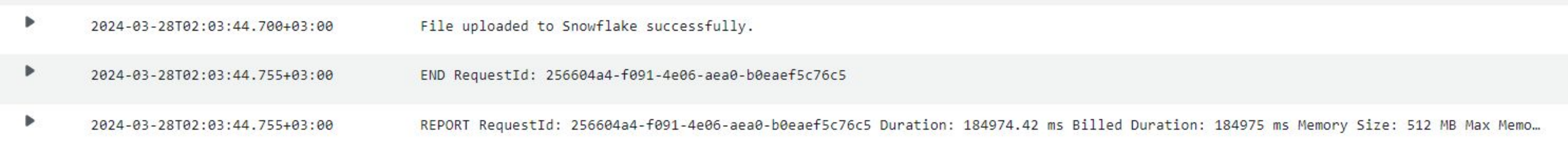
Code Snippet of the ***AWS Lambda*** handler.



```python
import os
import boto3
import requests
import snowflake.connector as sf


def lambda_handler(event, context):

    url = 'https://de-materials-tpcds.s3.ca-central-1.amazonaws.com/inventory.csv'
    destination_folder = '/tmp'
    file_name = 'inventory.csv'
    local_file_path = '/tmp/inventory.csv'

    # Snowflake connection parameters
    account = os.environ['account']
    warehouse = os.environ['warehouse']
    database = os.environ['database']
    schema = os.environ['schema']
    table = os.environ['table']
    user = os.environ['user']
    password = os.environ['password']
    role= os.environ['role']
    stage_name = os.environ['stage_name']

    # Download the data from the API endpoint
    response = requests.get(url)
    response.raise_for_status()



    # Save the data to the destination file in /tmp directory
    file_path = os.path.join(destination_folder, file_name)
    with open(file_path, 'wb') as file:
        file.write(response.content)

    with open(file_path, 'r') as file:
        file_content = file.read()
        print("File Content:")
        print(file_content)




    # Establish Snowflake connection
    conn = sf.connect(user = user, password = password, \
                 account = account, warehouse=warehouse, \
                  database=database,  schema=schema,  role=role)


    cursor = conn.cursor()

    # use schema
    use_schema = f"use schema {schema};"
    cursor.execute(use_schema)

    # create CSV format
    create_csv_format = f"CREATE or REPLACE FILE FORMAT COMMA_CSV TYPE ='CSV' FIELD_DELIMITER = ',';"
    cursor.execute(create_csv_format)



    create_stage_query = f"CREATE OR REPLACE STAGE {stage_name} FILE_FORMAT =COMMA_CSV"
    cursor.execute(create_stage_query)

    # Copy the file from local to the stage
    copy_into_stage_query = f"PUT 'file://{local_file_path}' @{stage_name}"
    cursor.execute(copy_into_stage_query)

    # List the stage
    list_stage_query = f"LIST @{stage_name}"
    cursor.execute(list_stage_query)

    # truncate table
    truncate_table = f"truncate table {schema}.{table};"
    cursor.execute(truncate_table)


    # Load the data from the stage into a table (example)
    copy_into_query = f"COPY INTO {schema}.{table} FROM @{stage_name}/{file_name} FILE_FORMAT =COMMA_CSV;"
    cursor.execute(copy_into_query)


    print("File uploaded to Snowflake successfully.")


    return {
        'statusCode': 200,
        'body': 'File downloaded and uploaded to Snowflake successfully.'
    }

```

Cont’d

[Week 1, Day 2]
To prove the process of successful table loading to ***Snowflake*** after the ***AWS EventBridge*** trigger at every 3 AM in the morning, here’s a snapshot of the ***CloudWatch***.



```
▶    2024-03-28T02:03:44.700+03:00    File uploaded to Snowflake successfully.
▶    2024-03-28T02:03:44.755+03:00    END RequestId: 256604a4-f091-4e06-aea0-b0eaef5c76c5
▶    2024-03-28T02:03:44.755+03:00    REPORT RequestId: 256604a4-f091-4e06-aea0-b0eaef5c76c5 Duration: 184974.42 ms Billed Duration: 184975 ms Memory Size: 512 MB Max Memo…
```

WeCloudData

Project Steps: Data Infrastructure

[Week 1, Day 3]

As Day 3 begun, we've came to the final task in setting up the infrastructure, which is using ***Airbyte*** to take the data from the ***AWS RDS (PostgreSQL)*** which is already been provided by WCD to extract the other tables along with its skeleton which means we don't have to define those tables like we did with inventory and ***Snowflake*** gets synced with the RDS to take the tables there.

Project Steps: Data Modeling

[Week 2]

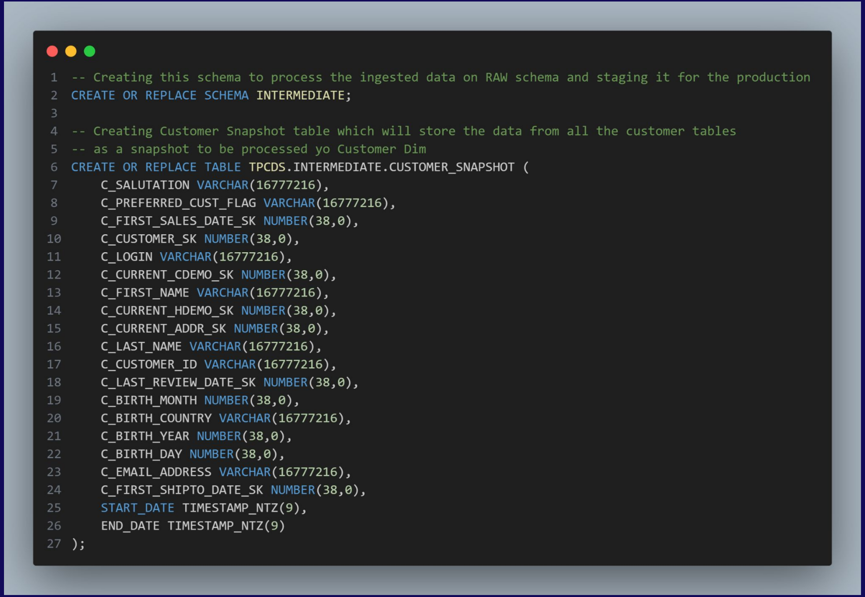
The tasks of the 2nd week is all about the modeling of our data to be ready for the ETL process, and answering the business requirements for it it to be ready for analytics with Metabase.

Data Modeling (cont’d)

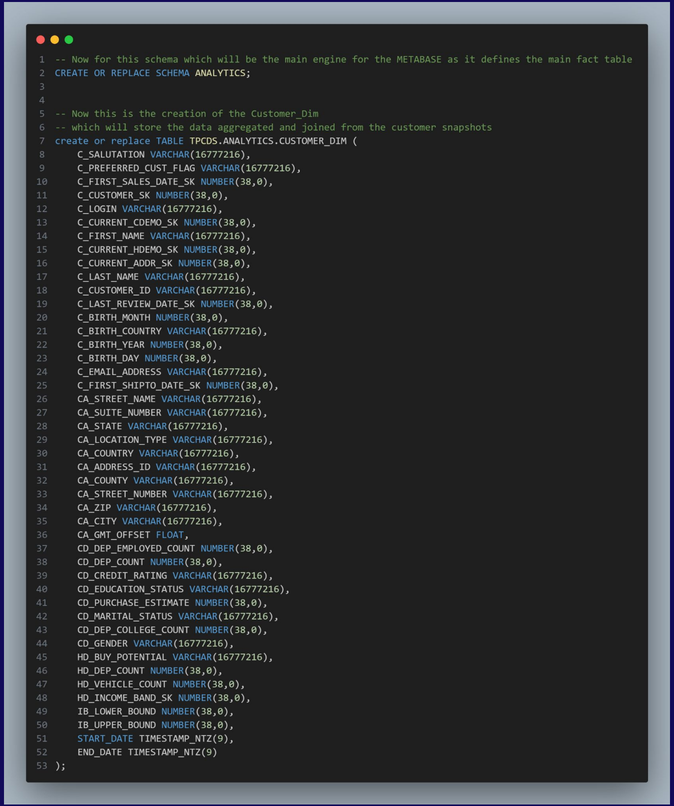
WeCloud**Data**

[Week 2]

Code snippet for data modeling



```sql
-- Creating this schema to process the ingested data on RAW schema and staging it for the production
CREATE OR REPLACE SCHEMA INTERMEDIATE;

-- Creating Customer Snapshot table which will store the data from all the customer tables
-- as a snapshot to be processed yo Customer Dim
CREATE OR REPLACE TABLE TPCDS.INTERMEDIATE.CUSTOMER_SNAPSHOT (
    C_SALUTATION VARCHAR(16777216),
    C_PREFERRED_CUST_FLAG VARCHAR(16777216),
    C_FIRST_SALES_DATE_SK NUMBER(38,0),
    C_CUSTOMER_SK NUMBER(38,0),
    C_LOGIN VARCHAR(16777216),
    C_CURRENT_CDEMO_SK NUMBER(38,0),
    C_FIRST_NAME VARCHAR(16777216),
    C_CURRENT_HDEMO_SK NUMBER(38,0),
    C_CURRENT_ADDR_SK NUMBER(38,0),
    C_LAST_NAME VARCHAR(16777216),
    C_CUSTOMER_ID VARCHAR(16777216),
    C_LAST_REVIEW_DATE_SK NUMBER(38,0),
    C_BIRTH_MONTH NUMBER(38,0),
    C_BIRTH_COUNTRY VARCHAR(16777216),
    C_BIRTH_YEAR NUMBER(38,0),
    C_BIRTH_DAY NUMBER(38,0),
    C_EMAIL_ADDRESS VARCHAR(16777216),
    C_FIRST_SHIPTO_DATE_SK NUMBER(38,0),
    START_DATE TIMESTAMP_NTZ(9),
    END_DATE TIMESTAMP_NTZ(9)
);
```

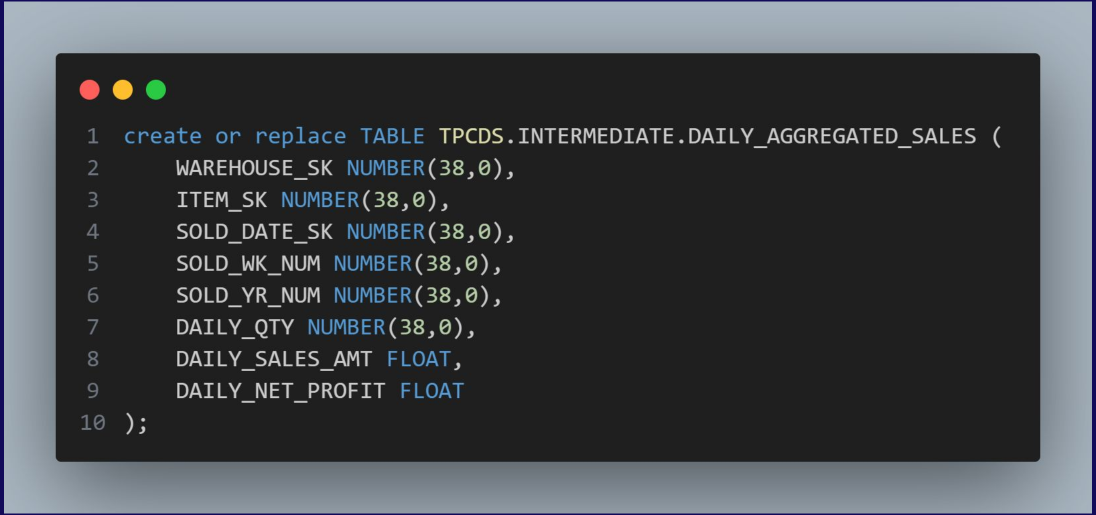


```sql
-- Now for this schema which will be the main engine for the METABASE as it defines the main fact table
CREATE OR REPLACE SCHEMA ANALYTICS;


-- Now this is the creation of the Customer_Dim
-- which will store the data aggregated and joined from the customer snapshots
create or replace TABLE TPCDS.ANALYTICS.CUSTOMER_DIM (
    C_SALUTATION VARCHAR(16777216),
    C_PREFERRED_CUST_FLAG VARCHAR(16777216),
    C_FIRST_SALES_DATE_SK NUMBER(38,0),
    C_CUSTOMER_SK NUMBER(38,0),
    C_LOGIN VARCHAR(16777216),
    C_CURRENT_CDEMO_SK NUMBER(38,0),
    C_FIRST_NAME VARCHAR(16777216),
    C_CURRENT_HDEMO_SK NUMBER(38,0),
    C_CURRENT_ADDR_SK NUMBER(38,0),
    C_LAST_NAME VARCHAR(16777216),
    C_CUSTOMER_ID VARCHAR(16777216),
    C_LAST_REVIEW_DATE_SK NUMBER(38,0),
    C_BIRTH_MONTH NUMBER(38,0),
    C_BIRTH_COUNTRY VARCHAR(16777216),
    C_BIRTH_YEAR NUMBER(38,0),
    C_BIRTH_DAY NUMBER(38,0),
    C_EMAIL_ADDRESS VARCHAR(16777216),
    C_FIRST_SHIPTO_DATE_SK NUMBER(38,0),
    CA_STREET_NAME VARCHAR(16777216),
    CA_SUITE_NUMBER VARCHAR(16777216),
    CA_STATE VARCHAR(16777216),
    CA_LOCATION_TYPE VARCHAR(16777216),
    CA_COUNTRY VARCHAR(16777216),
    CA_ADDRESS_ID VARCHAR(16777216),
    CA_COUNTY VARCHAR(16777216),
    CA_STREET_NUMBER VARCHAR(16777216),
    CA_ZIP VARCHAR(16777216),
    CA_CITY VARCHAR(16777216),
    CA_GMT_OFFSET FLOAT,
    CD_DEP_EMPLOYED_COUNT NUMBER(38,0),
    CD_DEP_COUNT NUMBER(38,0),
    CD_CREDIT_RATING VARCHAR(16777216),
    CD_EDUCATION_STATUS VARCHAR(16777216),
    CD_PURCHASE_ESTIMATE NUMBER(38,0),
    CD_MARITAL_STATUS VARCHAR(16777216),
    CD_DEP_COLLEGE_COUNT NUMBER(38,0),
    CD_GENDER VARCHAR(16777216),
    HD_BUY_POTENTIAL VARCHAR(16777216),
    HD_DEP_COUNT NUMBER(38,0),
    HD_VEHICLE_COUNT NUMBER(38,0),
    HD_INCOME_BAND_SK NUMBER(38,0),
    IB_LOWER_BOUND NUMBER(38,0),
    IB_UPPER_BOUND NUMBER(38,0),
    START_DATE TIMESTAMP_NTZ(9),
    END_DATE TIMESTAMP_NTZ(9)
);
```

Data Modeling (cont’d)

[Week 2]

Code snippet for data modeling

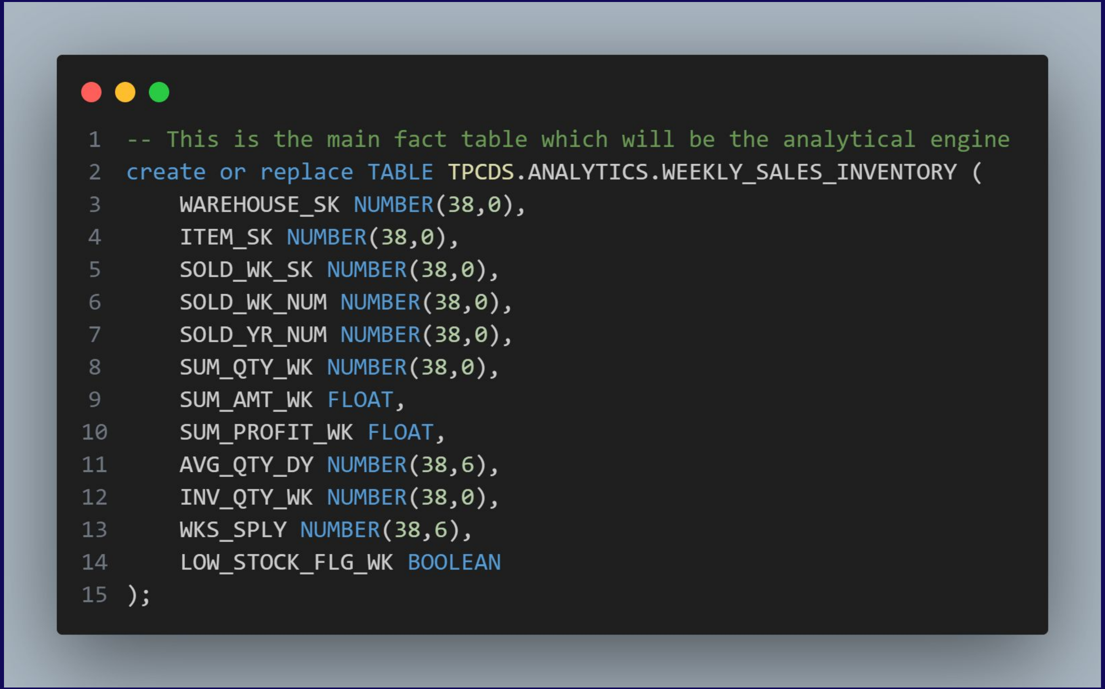


```sql
create or replace TABLE TPCDS.INTERMEDIATE.DAILY_AGGREGATED_SALES (
    WAREHOUSE_SK NUMBER(38,0),
    ITEM_SK NUMBER(38,0),
    SOLD_DATE_SK NUMBER(38,0),
    SOLD_WK_NUM NUMBER(38,0),
    SOLD_YR_NUM NUMBER(38,0),
    DAILY_QTY NUMBER(38,0),
    DAILY_SALES_AMT FLOAT,
    DAILY_NET_PROFIT FLOAT
);
```

Data Modeling (cont’d)

[Week 2]

Code snippet for data modeling



```sql
-- This is the main fact table which will be the analytical engine
create or replace TABLE TPCDS.ANALYTICS.WEEKLY_SALES_INVENTORY (
    WAREHOUSE_SK NUMBER(38,0),
    ITEM_SK NUMBER(38,0),
    SOLD_WK_SK NUMBER(38,0),
    SOLD_WK_NUM NUMBER(38,0),
    SOLD_YR_NUM NUMBER(38,0),
    SUM_QTY_WK NUMBER(38,0),
    SUM_AMT_WK FLOAT,
    SUM_PROFIT_WK FLOAT,
    AVG_QTY_DY NUMBER(38,6),
    INV_QTY_WK NUMBER(38,0),
    WKS_SPLY NUMBER(38,6),
    LOW_STOCK_FLG_WK BOOLEAN
);
```

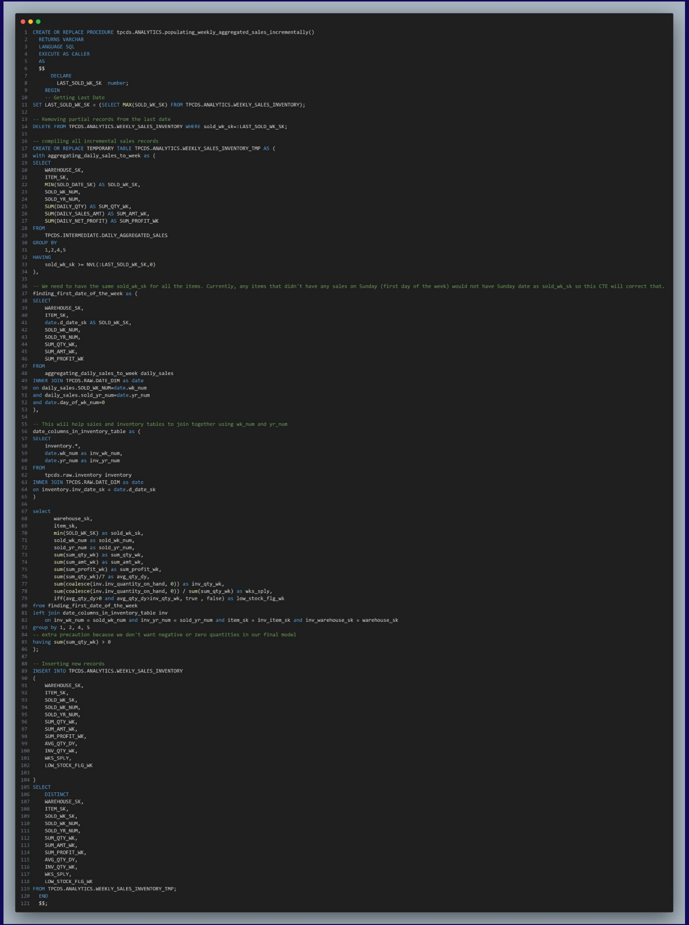
Project Steps: ETL

[Week 2]
The ETL is achieved through creating **stored procedures** and applying tasks to those procedures so data gets updated every time new data added to the Snowflake.

ETL (cont’d)

[Week 2]

Code snippet for creating a stored procedure and that one for the **weekly sales** that is crucial and the engine of the analytics



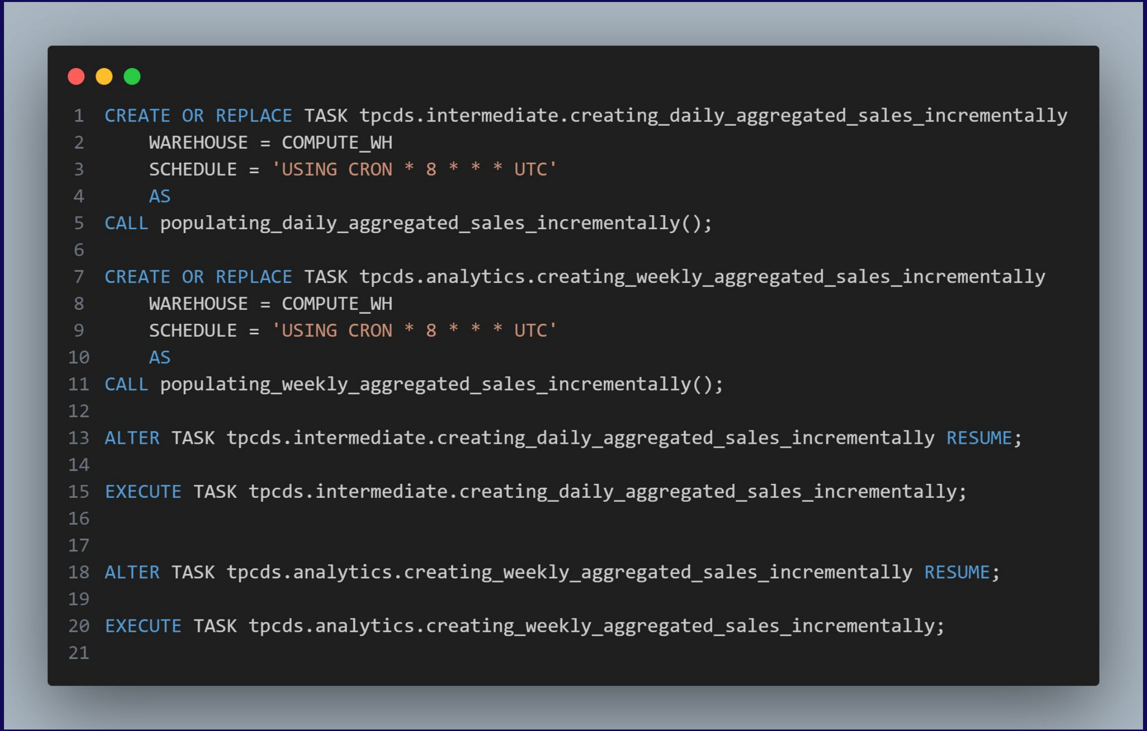
```sql
CREATE OR REPLACE PROCEDURE tpcds.ANALYTICS.populating_weekly_aggregated_sales_incrementally()
  RETURNS VARCHAR
  LANGUAGE SQL
  EXECUTE AS CALLER
  AS
  $$
      DECLARE
        LAST_SOLD_WK_SK  number;
    BEGIN
    -- Getting Last Date
SET LAST_SOLD_WK_SK = (SELECT MAX(SOLD_WK_SK) FROM TPCDS.ANALYTICS.WEEKLY_SALES_INVENTORY);

-- Removing partial records from the last date
DELETE FROM TPCDS.ANALYTICS.WEEKLY_SALES_INVENTORY WHERE sold_wk_sk=:LAST_SOLD_WK_SK;

-- compiling all incremental sales records
CREATE OR REPLACE TEMPORARY TABLE TPCDS.ANALYTICS.WEEKLY_SALES_INVENTORY_TMP AS (
with aggregating_daily_sales_to_week as (
SELECT
    WAREHOUSE_SK,
    ITEM_SK,
    MIN(SOLD_DATE_SK) AS SOLD_WK_SK,
    SOLD_WK_NUM,
    SOLD_YR_NUM,
    SUM(DAILY_QTY) AS SUM_QTY_WK,
    SUM(DAILY_SALES_AMT) AS SUM_AMT_WK,
    SUM(DAILY_NET_PROFIT) AS SUM_PROFIT_WK
FROM
    TPCDS.INTERMEDIATE.DAILY_AGGREGATED_SALES
GROUP BY
    1,2,4,5
HAVING
    sold_wk_sk >= NVL(:LAST_SOLD_WK_SK,0)
),

-- We need to have the same sold_wk_sk for all the items. Currently, any items that didn't have any sales on Sunday (first day of the week) would not have Sunday date as sold_wk_sk so this CTE will correct that.
finding_first_date_of_the_week as (
SELECT
    WAREHOUSE_SK,
    ITEM_SK,
    date.d_date_sk AS SOLD_WK_SK,
    SOLD_WK_NUM,
    SOLD_YR_NUM,
    SUM_QTY_WK,
    SUM_AMT_WK,
    SUM_PROFIT_WK
FROM
    aggregating_daily_sales_to_week daily_sales
INNER JOIN TPCDS.RAW.DATE_DIM as date
on daily_sales.SOLD_WK_NUM=date.wk_num
and daily_sales.sold_yr_num=date.yr_num
and date.day_of_wk_num=0
),

-- This will help sales and inventory tables to join together using wk_num and yr_num
date_columns_in_inventory_table as (
SELECT
    inventory.*,
    date.wk_num as inv_wk_num,
    date.yr_num as inv_yr_num
FROM
    tpcds.raw.inventory inventory
INNER JOIN TPCDS.RAW.DATE_DIM as date
on inventory.inv_date_sk = date.d_date_sk
)

select
        warehouse_sk,
        item_sk,
        min(SOLD_WK_SK) as sold_wk_sk,
        sold_wk_num as sold_wk_num,
        sold_yr_num as sold_yr_num,
        sum(sum_qty_wk) as sum_qty_wk,
        sum(sum_amt_wk) as sum_amt_wk,
        sum(sum_profit_wk) as sum_profit_wk,
        sum(sum_qty_wk)/7 as avg_qty_dy,
        sum(coalesce(inv.inv_quantity_on_hand, 0)) as inv_qty_wk,
        sum(coalesce(inv.inv_quantity_on_hand, 0)) / sum(sum_qty_wk) as wks_sply,
        iff(avg_qty_dy>0 and avg_qty_dy>inv_qty_wk, true , false) as low_stock_flg_wk
from finding_first_date_of_the_week
left join date_columns_in_inventory_table inv
    on inv_wk_num = sold_wk_num and inv_yr_num = sold_yr_num and item_sk = inv_item_sk and inv_warehouse_sk = warehouse_sk
group by 1, 2, 4, 5
-- extra precaution because we don't want negative or zero quantities in our final model
having sum(sum_qty_wk) > 0
);

-- Inserting new records
INSERT INTO TPCDS.ANALYTICS.WEEKLY_SALES_INVENTORY
(
    WAREHOUSE_SK,
    ITEM_SK,
    SOLD_WK_SK,
    SOLD_WK_NUM,
    SOLD_YR_NUM,
    SUM_QTY_WK,
    SUM_AMT_WK,
    SUM_PROFIT_WK,
    AVG_QTY_DY,
    INV_QTY_WK,
    WKS_SPLY,
    LOW_STOCK_FLG_WK

)
SELECT
    DISTINCT
    WAREHOUSE_SK,
    ITEM_SK,
    SOLD_WK_SK,
    SOLD_WK_NUM,
    SOLD_YR_NUM,
    SUM_QTY_WK,
    SUM_AMT_WK,
    SUM_PROFIT_WK,
    AVG_QTY_DY,
    INV_QTY_WK,
    WKS_SPLY,
    LOW_STOCK_FLG_WK
FROM TPCDS.ANALYTICS.WEEKLY_SALES_INVENTORY_TMP;
  END
  $$;
```

WeCloud**Data**

ETL (cont’d)

[Week 2]

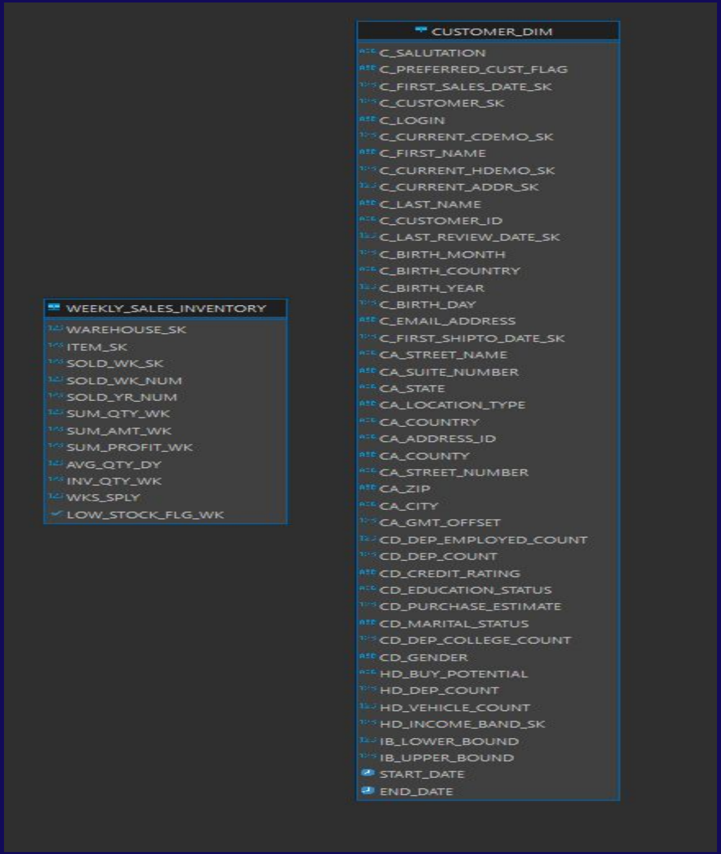
Code snippet for the tasks which will be the key to automating the performing of ETL by calling the stored procedures and updating the tables based on the CRON time given



```
CREATE OR REPLACE TASK tpcds.intermediate.creating_daily_aggregated_sales_incrementally
    WAREHOUSE = COMPUTE_WH
    SCHEDULE = 'USING CRON * 8 * * * UTC'
    AS
CALL populating_daily_aggregated_sales_incrementally();

CREATE OR REPLACE TASK tpcds.analytics.creating_weekly_aggregated_sales_incrementally
    WAREHOUSE = COMPUTE_WH
    SCHEDULE = 'USING CRON * 8 * * * UTC'
    AS
CALL populating_weekly_aggregated_sales_incrementally();

ALTER TASK tpcds.intermediate.creating_daily_aggregated_sales_incrementally RESUME;

EXECUTE TASK tpcds.intermediate.creating_daily_aggregated_sales_incrementally;


ALTER TASK tpcds.analytics.creating_weekly_aggregated_sales_incrementally RESUME;

EXECUTE TASK tpcds.analytics.creating_weekly_aggregated_sales_incrementally;

```

Project Steps:

Final Schema



WEEKLY_SALES_INVENTORY

WAREHOUSE_SK
ITEM_SK
SOLD_WK_SK
SOLD_WK_NUM
SOLD_YR_NUM
SUM_QTY_WK
SUM_AMT_WK
SUM_PROFIT_WK
AVG_QTY_DY
INV_QTY_WK
WKS_SPLY
LOW_STOCK_FLG_WK

CUSTOMER_DIM

C_SALUTATION
C_PREFERRED_CUST_FLAG
C_FIRST_SALES_DATE_SK
C_CUSTOMER_SK
C_LOGIN
C_CURRENT_CDEMO_SK
C_FIRST_NAME
C_CURRENT_HDEMO_SK
C_CURRENT_ADDR_SK
C_LAST_NAME
C_CUSTOMER_ID
C_LAST_REVIEW_DATE_SK
C_BIRTH_MONTH
C_BIRTH_COUNTRY
C_BIRTH_YEAR
C_BIRTH_DAY
C_EMAIL_ADDRESS
C_FIRST_SHIPTO_DATE_SK
CA_STREET_NAME
CA_SUITE_NUMBER
CA_STATE
CA_LOCATION_TYPE
CA_COUNTRY
CA_ADDRESS_ID
CA_COUNTY
CA_STREET_NUMBER
CA_ZIP
CA_CITY
CA_GMT_OFFSET
CD_DEP_EMPLOYED_COUNT
CD_DEP_COUNT
CD_CREDIT_RATING
CD_EDUCATION_STATUS
CD_PURCHASE_ESTIMATE
CD_MARITAL_STATUS
CD_DEP_COLLEGE_COUNT
CD_GENDER
HD_BUY_POTENTIAL
HD_DEP_COUNT
HD_VEHICLE_COUNT
HD_INCOME_BAND_SK
IB_LOWER_BOUND
IB_UPPER_BOUND
START_DATE
END_DATE

WeCloudData

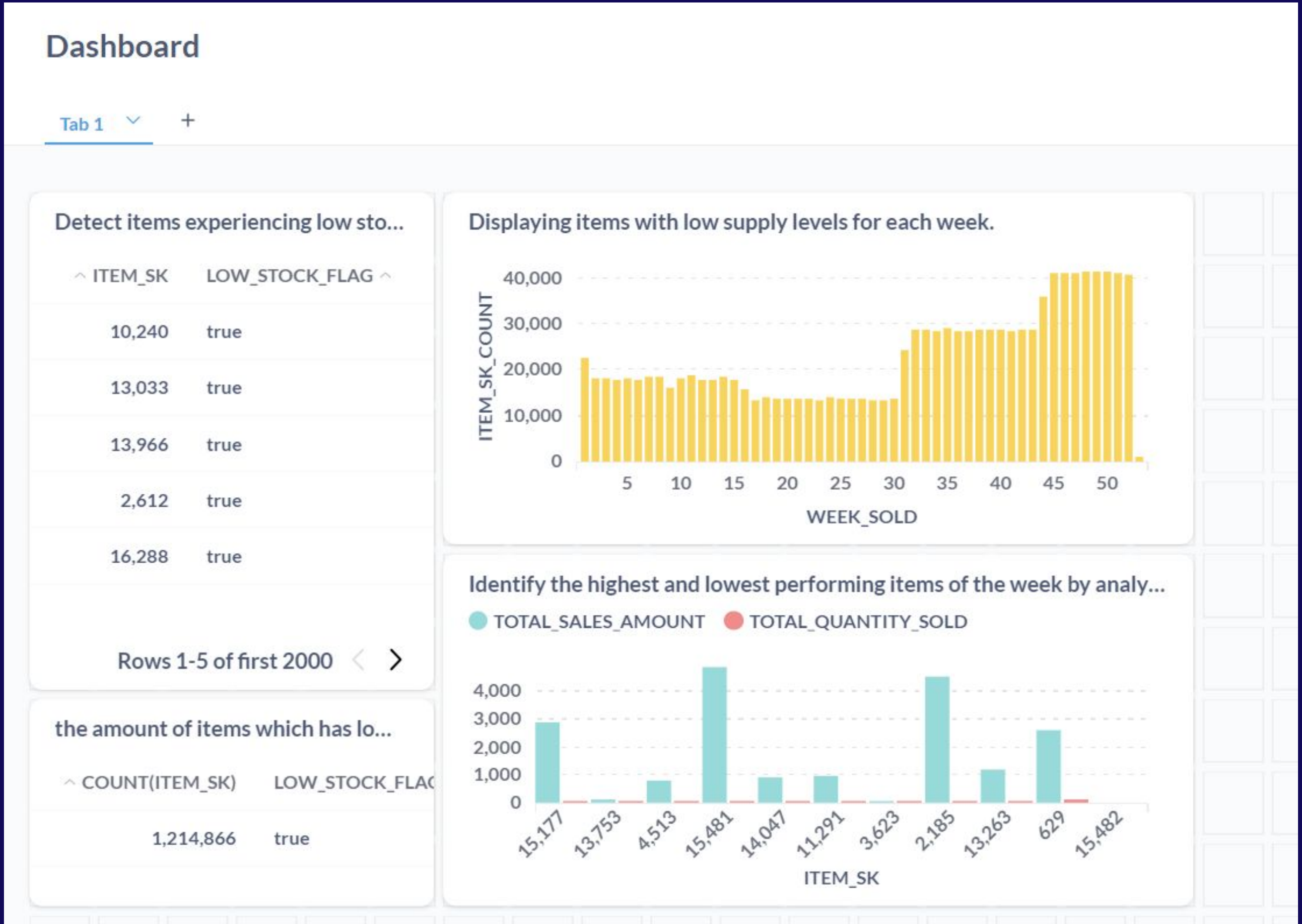
Project Steps: Metabase

[Week 2]

For the ***Metabase*** and the final task of the capstone project, the EC2 instance needed for the analytics tool that would allow the visual representation of the data finding, we used the created ***t3.small*** instance, as ***Docker*** is installed and configured to run *Metabase* as a container, and then connecting it to our Snowflake warehouse to sync the data and start answering the analytical tool.

Project Achievement

DASHBOARD



Detect items experiencing low sto...

ITEM_SK	LOW_STOCK_FLAG
10,240	true
13,033	true
13,966	true
2,612	true
16,288	true

Rows 1-5 of first 2000

the amount of items which has lo...

COUNT(ITEM_SK)	LOW_STOCK_FLAG
1,214,866	true

Displaying items with low supply levels for each week.

Identify the highest and lowest performing items of the week by analy...

Future

- If we had more time we will clean data to find perfect result(especially customer information)
- For further develop the project we are thinking about making Predictive Analytics Model , include predictive analytics by building machine learning models
- For further research that we want to do on the project in the future is Data Governance compliance , such as data protection mechanisms

Thank you!

WeCloud**Data**