Big Data Hadoop and Spark Developer

Course-End Project Problem Statement

Retail Business Analytics

Problem Statement:

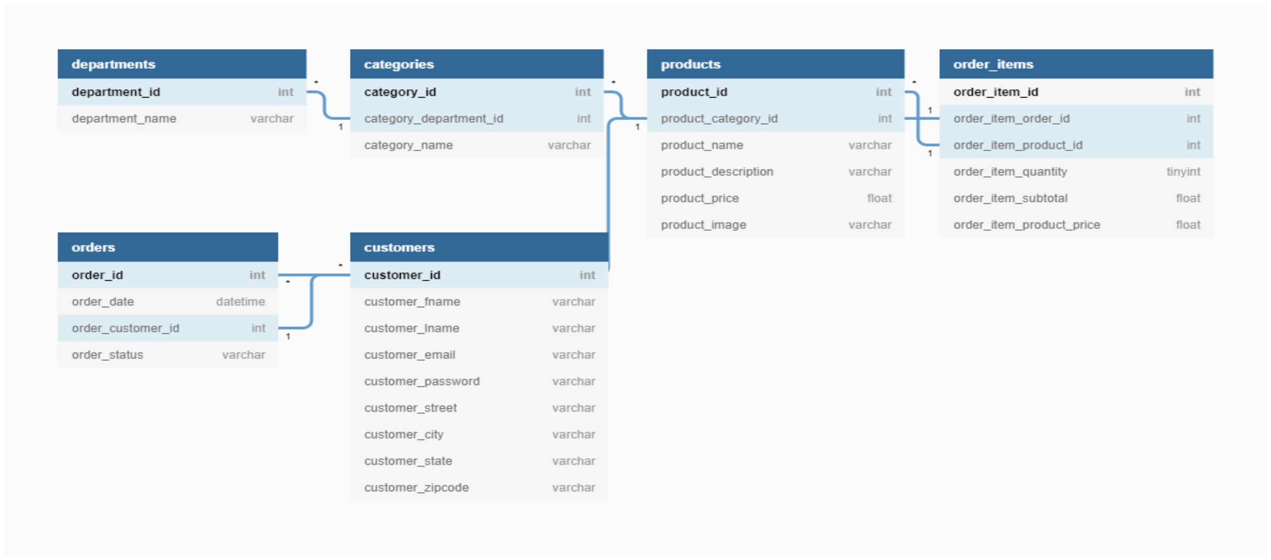
Customers can purchase products or services from Amazon for consumption and usage. Amazon usually sells products and services in-store; however, some may be sold online or over the phone and shipped to the customer. Clothing, medicine, supermarket, and convenience stores are examples of their retail operations.

Objective:

The objective is to analyze the **retail_db** dataset, provide reports on the total completed orders, and perform customer and product analytics.

Dataset to be Used: data_files

ER Diagram:



Understanding the Data Model

RETAIL_DB database contains the following tables:

- Department
- Customer
- Categories
- Products
- Orders
- Orders items

Steps Overview:

Step 1: Upload the **data-files** file to HDFS through FTP

1.1 Download the relevant dataset from the **Reference Materials** section or the project description

1.2 Upload the dataset to the **FTP** lab from your local system

1.3 Upload the dataset to **HDFS** from **Terminal** using the put command

Syntax of put command for reference:

```
hdfs dfs -put <FTP_folder _name_source> <hdfs_path_destination>
```

Example:

```
hdfs dfs -put data-files /user/abcsimplilearn/data-files-project
```

Step 2: Perform the tasks on the uploaded dataset using PySpark

Tasks to be Performed:

Task 2.1:

Log in to PySpark shell

Task 2.2:

Explore the customer records saved in the **customers-tab-delimited** directory on HDFS

Requirement:

2.2.1 Show the client information for those who live in California

2.2.2 The final output must be in text format

2.2.3 Save the results in the result/scenario1/solution folder

2.2.4 Only records with the state value **CA** should be included in the result

2.2.5 Only the customer's entire name should be included in the output

Example: **Robert Hudson**

Task 2.3:

Explore the order records saved in the **orders parquet** directory on HDFS

Requirement:

2.3.1 Show all orders with the order status value **COMPLETE**

2.3.2 The output should be in JSON format

2.3.3 Save the data in the **result/scenario2/solution** directory on HDFS

2.3.4 The **order date** column should be in the **YYYY-MM-DD** format

2.3.5 Use GZIP compression to compress the output

2.3.6 Only the column names listed below should be included in the output:

2.3.6.1 Order number

2.3.6.2 Order date

2.3.6.3 Current situation

Task 2.4

Explore the customer records saved in the **customers-tab-delimited** directory on HDFS

Requirement:

2.4.1 Produce a list of all consumers who live in the city of **Caguas**

2.4.2 Save the data in the result/scenario3/solution directory on HDFS

2.4.3 The result should only contain records with the value **Caguas** for the customer city

2.4.4 Use snappy compression to compress the output

2.4.5 Save the file in the orc format

Task 2.5

Explore all the category records stored in the **categories** directory on HDFS

Requirement:

2.5.1 Save the result files in CSV format

2.5.2 Save the data in the result/scenario4/solution directory on HDFS

2.5.3 Use lz4 compression to compress the output

Task 2.6

Explore all product records that are saved in the **products_avro** database

Requirement:

2.6.1 Only products with a price of more than 1000.0 should be included in the output

2.6.2 Save the output files in parquet format

2.6.3 Remove data from the table if the product price is lesser than 1000.0

2.6.4 Save the data in the result/scenario5/solution directory on HDFS

2.6.5 Use snappy compression to compress the output

Task 2.7

Explore the **products_avro** stored in product records

Requirement:

2.7.1 Only products with a price of more than 1000.0 should be in the output

2.7.2 The pattern **Treadmill** appears in the product name

2.7.3 Save the output files in parquet format

2.7.4 Save the data in the result/scenario6/solution directory on HDFS

2.7.5 Use GZIP compression to compress the output

Task 2.8

Explore the order records that are saved in the **orders parquet** table on HDFS

Requirement:

2.8.1 Output all PENDING orders in July 2013

2.8.2 Output files should be in JSON format

2.8.3 Save the data in the result/scenario7/solution directory on HDFS

2.8.4 Only entries with the order status value of **PENDING** should be included in the result

2.8.5 Order date should be in the YYY-MM-DD format

2.8.6 Use snappy compression to compress the output, which should just contain the order date and order status