

## Contents

### PART A

1. Introduction .....	3
2. Business Requirements.....	3
3. Reporting Requirement .....	5
4. Understanding the Business Data.....	5
5. Schema and Granularity.....	6
5.1. Identify Grain Level .....	6
5.2. Grain Declaration .....	7
5.3. Identify Dimensions and their Attributes .....	7
5.4. Identify Dimensions and Measures of the Fact Tables .....	8
5.4.1. Sending Purchase Orders.....	8
5.4.2. Receiving Purchase Orders .....	9
5.4.3. Controlling and Maintaining Stock.....	9
6. Data Warehouse Schema and Design.....	9
7. Further Works .....	12

### PART B

1. Introduction .....	13
2. Analysis and Design of a data dimension data model .....	13
3. Implementation and Testing of Data Warehouse .....	14
3.1. Creating Database.....	15
3.2. Create Dimension tables.....	15
3.3. Create Fact tables .....	16
3.4. Add appropriate primary key constraints.....	16
3.5. Add appropriate foreign key constraints.....	17
4. Populating the Data Warehouse.....	17
4.1. Populate Dimensions .....	17
4.2. Populate Fact Tables .....	17
4.2.1. Creating Lookup tables in the Staging Area .....	18
4.2.2. Populating Lookup tables.....	18
4.2.3. Populating the transaction lookup table in the Staging Area .....	19

4.2.4.	Populating the Fact tables .....	19
5.	Test implementation.....	20
6.	Testing some of the Business Requirements .....	22
6.1.	Analysing stock levels by brand or product type or supplier.....	22
7.	Exporting the data warehouse to a flat file destination.....	23
8.	Migrating Data file to Apache HDFS.....	24
9.	Creating a data structure in hive.....	24
10.	Demonstrating the use of Pig .....	25
10.1.	DUMP .....	25
10.2.	FILTER and GROUP.....	26
10.3.	Join Data.....	27
10.4.	SORT .....	29
11.	REFLECTIVE CONCLUSION .....	30

## 1. Introduction

ABC Consumer Electronics Outlet Ltd is a multi-channel retailer of consumer electronics that has six stores operated in different locations in and around London. The Consumer Electronics retail industry is very competitive, hence the need to gain access to innovative information and communication technology to gain more advantage in this competitive business. ABC Consumer Electronics Outlet Ltd has already invested in information and communication technologies to support its business processes – retail activities, online business automation, and finance department support using Vend, Linnworks, and Xerox respectively which generate huge transactional data every day. However, the managers are still having a difficult time producing reports and performing analysis to gain insights for decision-making.

Several business requirements have been identified by the business analyst. This report is centered only on proffering a data warehousing solution for the inventory management business process. Simple Star schema dimensional data model design was adopted in this coursework with three (3) fact tables to cover the business activities of the company and four (4) dimensions to answer to the requested report and analysis needed for inventory management.

Daily, weekly and monthly reports demands are expected to be effectively generated from the warehouse system. This in turn will help the management to achieve the aim of increased revenue and customer satisfaction.

## 2. Business Requirements

Data warehousing business requirements help describe the solution needed in detail and business terms. They are usually deduced from the outcome of function user's requirement gathering conducted by business analysts. This will form the basis for developing a good data warehousing solution to meet all users' reporting and analysis needs. The aspect of business requirements in the development and design process of a data warehousing project cannot be overemphasized as it determines the success of the data warehousing solution. The scope of the business requirement for ABC Consumer Electronics Outlet Ltd is on Inventory Management which focuses on how managers can control and manage product stock levels. Table 1 and 2 below gives a more detailed description of the business requirement and the processes involved in developing the data warehousing solution.

Business Requirement	
Business Requirement Code	01
Business Requirement Name	Inventory Management
Priority	High
Description	

This requirement is focused on controlling and managing inventory or stocks. The main goal is to produce a good purchase order plan to ensure that products do not run out-of-stock and to also reduce overstocking of products to maximize profit and increase customer satisfaction.

*Table 1: Business Requirement for Inventory Management*

BUSINESS PROCESSES	
Business Process Code	01
Business Process Name	Sending Purchase Orders
Priority	High
Description	
This process will keep a log of purchase orders sent, ordered quantity, sent date, product details, and supplier details for products with low/minimum levels of inventory.	
Process Events	Active, Discontinued
Measured By:	Ordered Quantity
Related To:	Product, Supplier, Date, Outlet
Business Process Code	02
Business Process Name	Receiving Purchase Orders
Priority	High
Description	
This process will manage all received orders while noting the destination outlet for storage. It will keep track of the received purchase order, quantity ordered, quantity received, received date, product details and supplier's details.	
Process Events	Active, Discontinued
Measured By:	Ordered Quantity, Received Quantity
Related To:	Product, Supplier, Date, Outlet
Business Process Code	03
Business Process Name	Controlling and Maintaining Stock
Priority	High
Description	
This process will manage stock levels for all products ensuring that new products are added while the stock levels is being adjusted for all existing products.	
Process Events	Active, Discontinued
Measured By:	StockLevel
Related To:	Product, Supplier, Date

*Table 2: Business Processes for Inventory Management Requirements*

### 3. Reporting Requirement

Inventory management is one of the most important business processes of a company. The main goal is to produce a good purchase order plan to ensure that product items are available when they are needed. The data warehousing solution is expected to meet the following reporting and analysis requirements:

- **A daily stock levels of all products for the last month:** This involves all available stocks of all products in the last month (a monthly report)
- **A weekly report of all products with minimum stock levels:** This is a report of all products that fall within the limit of the stock level (a weekly report). This helps the company to keep tabs on their stocks to know when they are over stock or understocked.
- **Analyzing stock levels by brand or product type or supplier:** This report involves segregation and analysis of stock levels by brand or product type or supplier. This report helps the company to have information on their stocks based on the brand or product type or supplier.
- **Daily and weekly sent and received stock orders for the last four weeks:** This report helps to monitor stock orders daily and weekly so the company can have information on their level of demand and supplies.
- **Analyzing received stock orders by the supplier and by the month:** This report helps to monitor who the suppliers are and how many orders they received from them every month.

Based on the problems the company is facing in controlling over-stocks and under-stocks of product items. The above report requirement will give an overview of the company's well-being and in turn reduce over-stocks and under-stocks.

The report when properly utilized will increase revenue and customer satisfaction.

### 4. Understanding the Business Data

This coursework utilizes the data sourced from Vend which is a cloud-based software used by the company to support its retail activities. The data contains details of the products, suppliers, and stock movements from suppliers to warehouse locations. There are five (5) data flat files provided, which are as follows:

- Generated Date and Time
- Products Details.
- Suppliers Details.
- Sent Purchase Orders.
- Received Purchase Orders.

The generated date and time are in a .txt flat-file format and it is within the year range 01/01/2006 - 01/01/2027. This data will be used to generate date and time dimensions. It contains the following columns:

Column Names	Data Types	Description
FullDate	DateTime	Stores date in (dd/mm/yyyy) format
DateName	DateTime	Stores date in (yyyy/mm/dd) format
DayOfWeek	Int	Stores day of the week- 7 days (values from 1-7)
DayNameOfWeek	Str	Stores day name of the week (Sunday to Monday)
DayOfYear	Int	Stores day of the year – 365 days a year
WeekOfYear	Int	Stores week of the year – 52 weeks of the year
MonthName	Str	Stores month Name – January to December
MonthOfYear	Int	Stores month number – 12 months a year
CalendarQuarter	Int	Stores quarter of the year.
CalendarYear	Str	Stores year in (yyyy) format.

*Table 3: Generated Date Attributes and Description*

The supplier data file include SupplierName, Description, Phone, Email, Fax, FirstLineAddress, PostCode, City, State, and CountryID. The solution will use the attributes to identify and differentiate each supplier.

The product data file is SKU, ProductName, Description, Condition, ProductType, Brand, SupplierName, Tags, CostPrice, RetailPrice, CurrentStockLevel, DateCreatedAt, DateDiscontinuedAt, IsActive. The CurrentStockLevel is used for observing and maintaining stock trends.

For stock movements, the tables SentPurchaseOrders and ReceivedPurchaseOrders are used.

The ReceivedPurchaseOrder has PurchaseOrderCode, ProductSKU, SupplierName, DestinationOutletID, SentDate, ReceivedDate, ReceivedQty, and OrderedQty attributes.

The SentPurchaseOrders include PurchaseOrderCode, ProductSKU, SupplierName, DestinationOutlet, SentDate, and OrderedQty.

## 5. Schema and Granularity

### 5.1. Identify Grain Level

The grain of a fact table **represents the atomic level by which the facts may be defined**. The grain of a sales fact table might be stated as "sales volume by day by product by store". Each record in this fact table is therefore uniquely defined by a day, product, and store.

## 5.2. Grain Declaration

The records in the fact table are on the day level, outlet level, and on product level. This granularity defines each record as uniquely as possible and all requirements from the managers can be fulfilled. The following identifies the grain of the three different central fact tables:

1. Sending Purchase Orders: A line item on a purchase order sent to suppliers daily
2. Receiving Purchase Orders: A line item on a purchase order received from suppliers and stored in a location daily.
3. Controlling and Maintaining Stocks: An individual line item is added to products while updating stock levels daily.

## 5.3. Identify Dimensions and their Attributes

This data warehousing solution identified five (5) Dimensions. These dimensions are briefly described and justified below:

- **DATE DIMENSION:** This dimension is very important as it will be useful in the daily, weekly, and monthly report generation and analysis to meet the reporting analysis. The attributes identified based on the available data source include:  
FullDate(PK)  
DateName  
DayOfWeek  
DayNameOfWeek  
DayOfYear  
WeekOfYear  
MonthName  
MonthOfYear  
CalendarQuarter  
CalendarYear
- **PRODUCT DIMENSION:** This is a slowly changing dimension that contains the product details. The attributes identified based on the available data source include:  
SKU (PK)  
ProductName  
Description  
Condition  
ProductType  
Brand  
Tags  
CostPrice  
RetailPrice  
CurrentStockLevel  
DateCreatedAt  
DateDiscontinuedAt

IsActive

- **SUPPLIER DIMENSION:** This is a slowly changing dimension that contains the supplier's details. The attributes identified based on the available data source include:
  - SupplierID (PK)
  - SupplierName
  - Description
  - Phone
  - Email
  - Fax
  - FirstLineAddress
  - PostCode
  - City
  - State
  - CountryID
- **OUTLET DIMENSION:** This is a slowly changing dimension that contains the Outlet or storage location details. The available data source contains only the OutletName and does not include detailed information on the Outlet.
  - OutletID (PK)
  - OutletCode
  - OutletName
  - OutletAddress
  - OutletPhone
  - OutletEmail
  - City
  - State
  - CountryID
- **PURCHASEORDER DIMENSION:** This is a degenerate dimension that stores the details of the purchase order.
  - PurchaseOrderCode (PK)

#### 5.4. Identify Dimensions and Measures of the Fact Tables

The dimensions and measures identified for each of the fact tables are as follows:

##### 5.4.1. Sending Purchase Orders

The dimensions identified for the SendingPurchaseOrder Fact table are as follows:

- Date Dimension
- Product Dimension
- Supplier Dimension

The measure identified is

- Ordered Quantity

#### 5.4.2. Receiving Purchase Orders

The dimensions identified for the ReceivingPurchaseOrder Fact table are as follows:

- Date Dimension
- Product Dimension
- Supplier Dimension
- Outlet Dimension

The measures identified are:

- Ordered Quantity
- Received Quantity

#### 5.4.3. Controlling and Maintaining Stock

The dimensions identified for the ControllingandMantainingStocks Fact table are as follows:

- Date Dimension
- Product Dimension
- Supplier Dimension

The measure identified is

- CurrentStockLevel

## 6. Data Warehouse Schema and Design

A simple star schema dimensional data model design was used to design the models. Fig 1, 2 and 3 show a simple star schema for Controlling and Maintaining Stocks, Receiving Purchase Orders, and Sending Purchase Orders fact tables respectively. Fig 4 shows a combined view of all the schemas on a page.

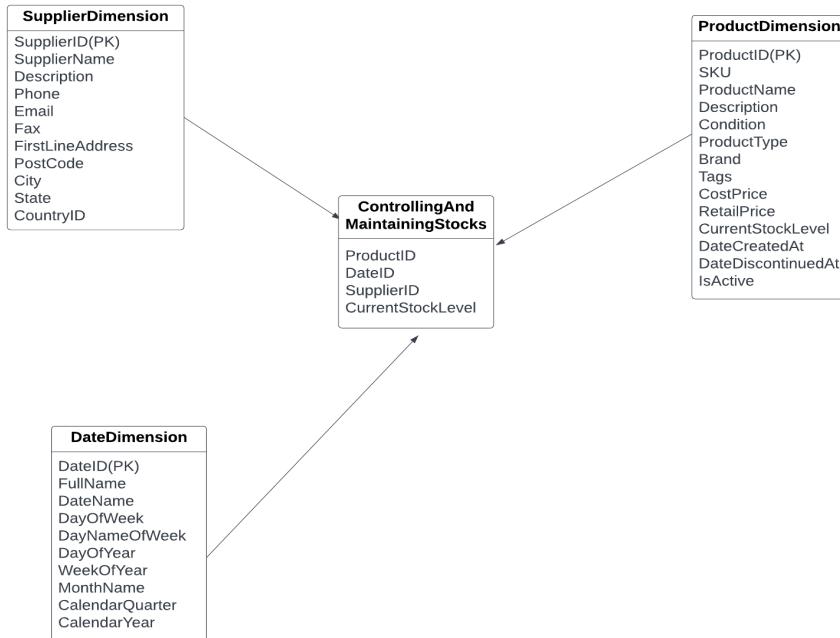


Figure 1: A simple Star Schema for Controlling and Maintaining Stocks

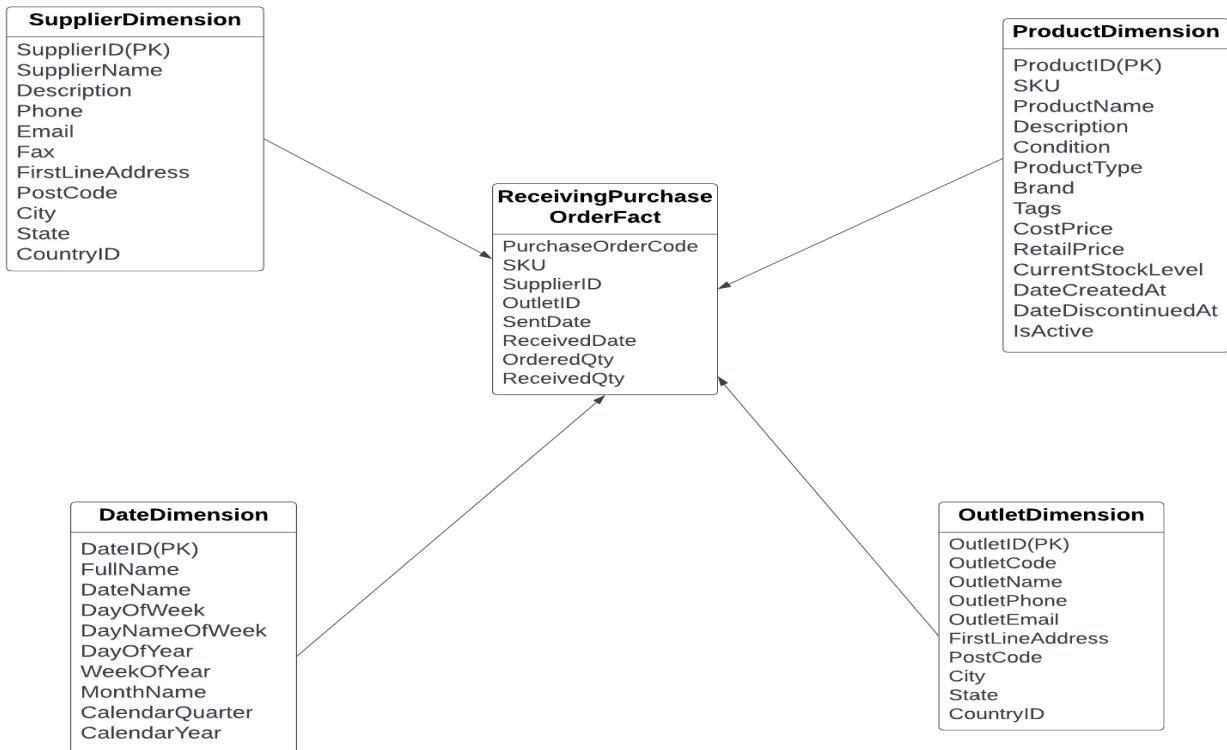


Figure 2: A simple Star Schema for Receiving Purchase Orders fact table

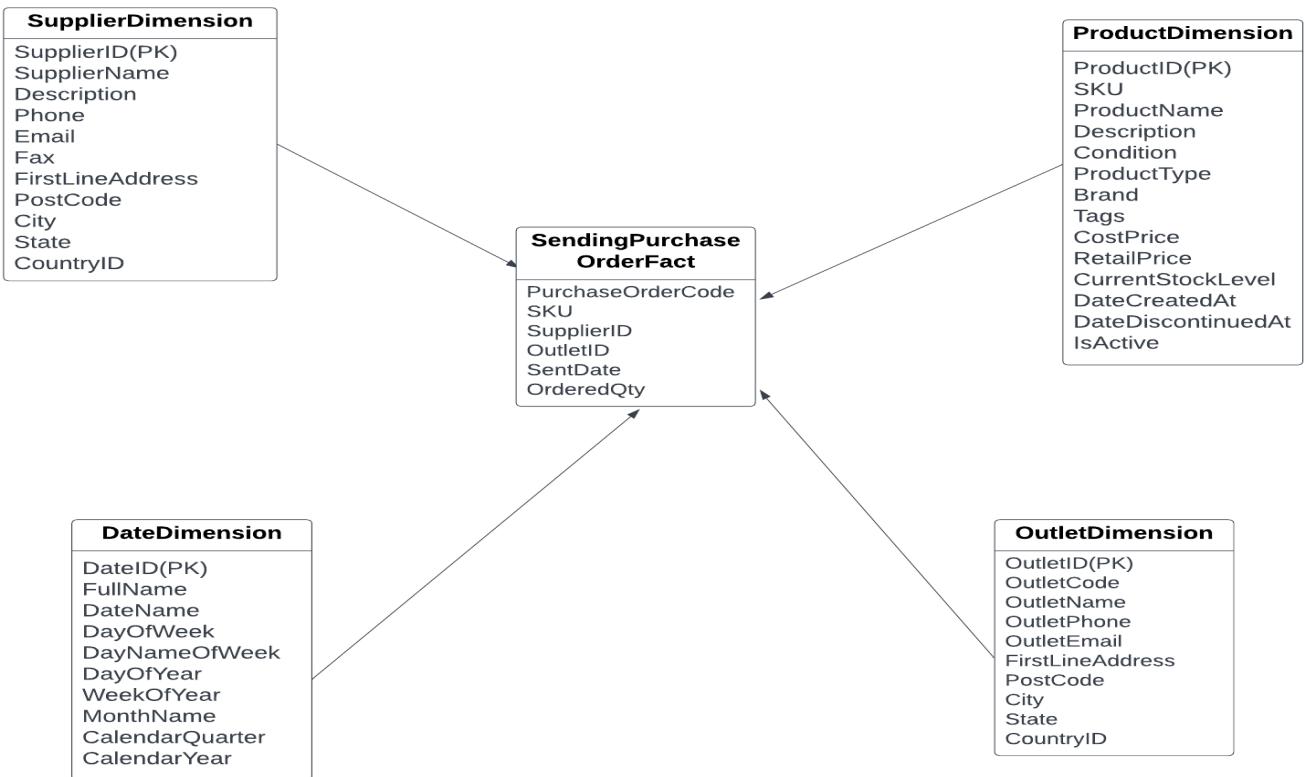


Figure 3: A simple Star Schema for Receiving Purchase Orders fact table

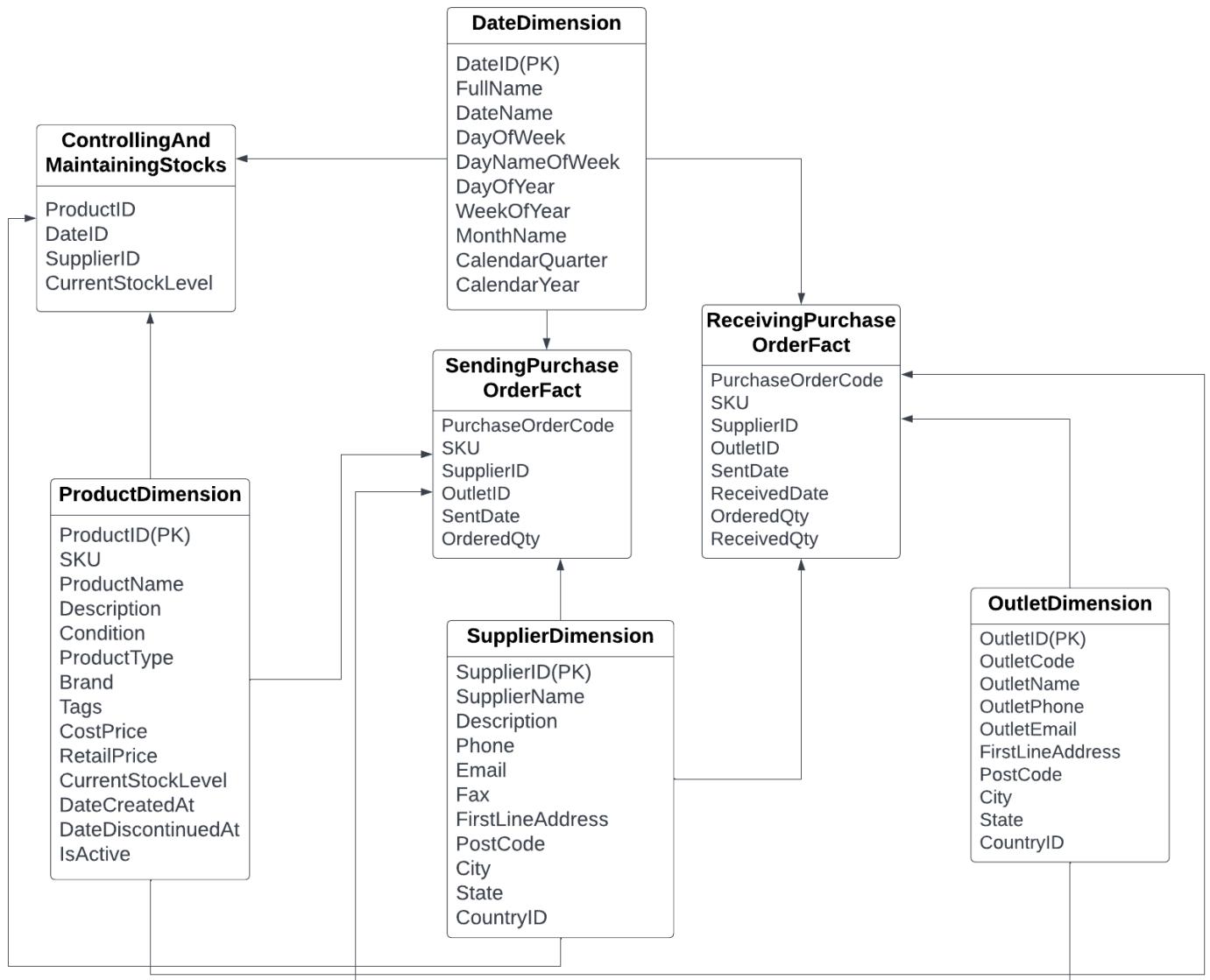


Figure 4: Overall Graphical Representation of all the schemas

## 7. Further Works

The design of the star schema will be implemented on the SQL server. The business and reporting requirements will be checked at all times during the implementation phase to ensure that the data warehouse conforms to the requirements for reporting and analysis. If the requirements are not being met, the design phase will be updated to meet the requirements.

## PART B

### 1. Introduction

The business case scenario of this coursework is based on ABC Consumer Electronics Outlet Ltd- which is in the retail industry for consumer electronics in London and Europe. The company has invested in Vend, Linnworks, and Xerox to support its business processes which generate huge transactional data daily. Despite their investments in information and communication technologies, the managers still have difficulty in producing and performing analysis to gain insights for decision-making.

This coursework aims to design, implement, and test a data warehouse solution for the inventory management process of the ABC Consumer Electronics Outlet Ltd business case scenario, to export data from it and ingest it for further processing on a Big Data platform. The Objectives are as follows:

- Analyse and design of the data dimensional model.
- Create a Relational Database to store dimensions and fact tables using Microsoft SQL Server Management Studio.
- Migrate test data from the data warehouse to an Apache Hadoop platform (Hortonworks Data Platform) for further analysis of Big Data.
- Implement and test the Big Data Storage on Apache HDFS
- Implement a suitable data structure to load data into HIVE using Data Analysis Studio (DAS)
- Implement and manipulate the loaded data using Pig

### 2. Analysis and Design of a data dimension data model

The first step in the data dimensional modeling is business and reporting requirement gathering and grain declaration. The scope of the business requirement for ABC Consumer Electronics Outlet Ltd is on Inventory Management which focuses on how managers can control and manage product stock levels. Table 1 and 2 below gives a more detailed description on the business requirement and the processes involved in developing the data warehousing solution.

Business Requirement	
Business Requirement Code	01
Business Requirement Name	Inventory Management
Priority	High
Description	
This requirement is focused on controlling and managing inventory or stocks. The main goal is to produce a good purchase order plan to ensure that products do not run out-of-stock and to also reduce overstocking of products to maximize profit and increase customer satisfaction.	

Table 4: Business Requirement for Inventory Management

BUSINESS PROCESSES			
<b>Business Process Code</b>	01	02	03
<b>Business Process Name</b>	Sending Purchase Orders	Receiving Purchase Orders	Controlling
<b>Priority</b>	High	High	High
<b>Description</b>	This process will keep a log of purchase orders sent, ordered quantity, sent date, product details, and supplier details for products with low/minimum levels of inventory.	This process will manage all received orders while noting the destination outlet for storage. It will keep track of the received purchase order, quantity ordered, quantity received, received date, product details and supplier's details.	This process for all products are at levels is being products.
<b>Process Events</b>	Active, Discontinued	Active, Discontinued	
<b>Measured By:</b>	Ordered Quantity	Ordered Quantity, Received Quantity	Stock Level
<b>Related To:</b>	Product, Supplier, Date	Product, Supplier, Date, Outlet	Product, Sup

*Table 5: Business Processes for Inventory Management Requirements*

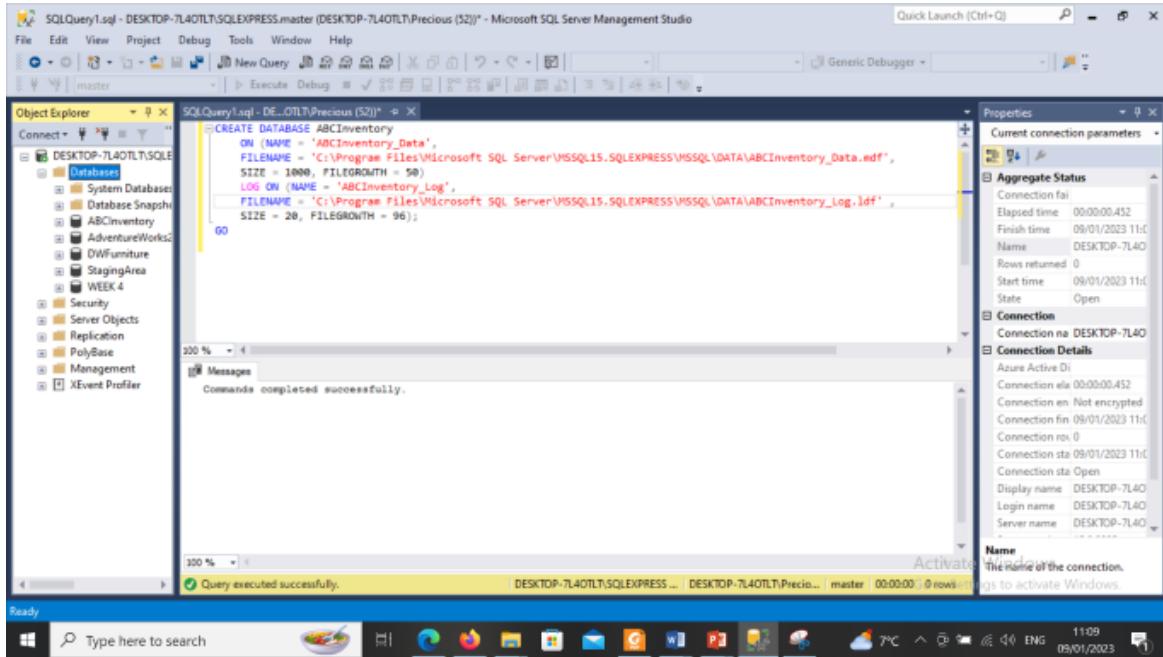
The reporting requirement required are as follows:

- A daily stock levels of all products for the last month
- A weekly report of all products with minimum stock levels
- Analyzing stock levels by brand or product type or supplier
- Daily and weekly sent and received stock orders for the last four weeks
- Analyzing received stock orders by supplier and by month

### 3. Implementation and Testing of Data Warehouse

The data warehouse was implemented using the SQL Server Management Studio (SSMS).

### 3.1. Creating Database



```
CREATE DATABASE ABCInventory
ON (NAME = 'ABCInventory_Data',
    FILENAME = 'C:\Program Files\Microsoft SQL Server\MSSQL15.SQLEXPRESS\MSSQL\DATA\ABCInventory_Data.mdf',
    SIZE = 10MB, FILEGROWTH = 50)
LOG ON (NAME = 'ABCInventory_Log',
    FILENAME = 'C:\Program Files\Microsoft SQL Server\MSSQL15.SQLEXPRESS\MSSQL\DATA\ABCInventory_Log.ldf',
    SIZE = 2MB, FILEGROWTH = 96);
```

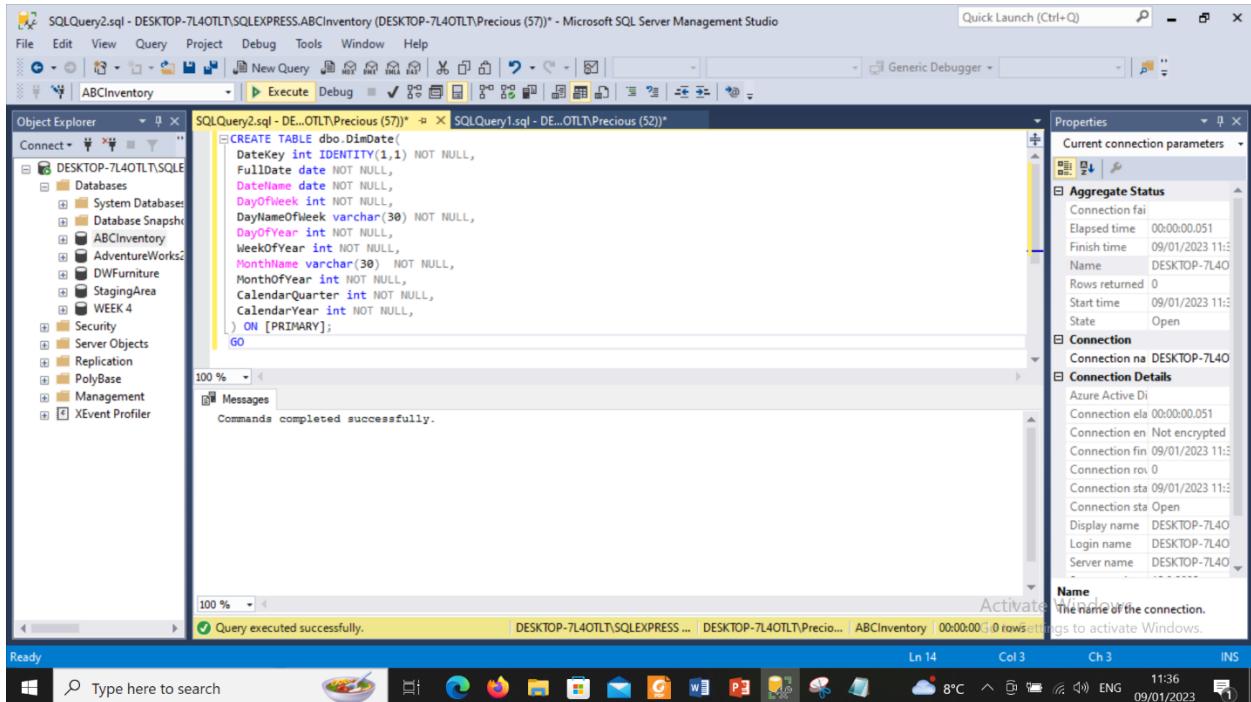
Messages  
Commands completed successfully.

Query executed successfully.

DESKTOP-7L40TLT\SQLEXPRESS ... DESKTOP-7L40TLT\Preciou... master 00:00:00 0 rows

Ready Type here to search 11:09 7°C ENG 09/01/2023

### 3.2. Create Dimension tables



```
CREATE TABLE dbo.DimDate(
    DateKey int IDENTITY(1,1) NOT NULL,
    FullDate date NOT NULL,
    DateName date NOT NULL,
    DayOfWeek int NOT NULL,
    DayNameOfWeek varchar(30) NOT NULL,
    DayOfYear int NOT NULL,
    WeekOfYear int NOT NULL,
    MonthName varchar(30) NOT NULL,
    MonthOfYear int NOT NULL,
    CalendarQuarter int NOT NULL,
    CalendarYear int NOT NULL,
    ) ON [PRIMARY];
GO
```

Messages  
Commands completed successfully.

Query executed successfully.

DESKTOP-7L40TLT\SQLEXPRESS ... DESKTOP-7L40TLT\Preciou... ABCInventory 00:00:00 0 rows

Activate Windows  
The name of the connection.

Ready Type here to search 11:36 8°C ENG 09/01/2023

### 3.3. Create Fact tables

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the database 'master' is selected. In the center pane, two CREATE TABLE statements are displayed:

```
CREATE TABLE dbo.SendingPurchaseOrderFact(
    SentDateID int NOT NULL,
    PurchaseOrderCode varchar(30) NOT NULL,
    ProductID int NOT NULL,
    SupplierID int NOT NULL,
    OutletID int NOT NULL,
    OrderedQty int NOT NULL
) ON [PRIMARY];
GO

CREATE TABLE dbo.ReceivingPurchaseOrderFact(
    SentDateID int NOT NULL,
    ReceivedDateID int NOT NULL,
    PurchaseOrderCode varchar(30) NOT NULL,
    ProductID int NOT NULL,
    SupplierID int NOT NULL,
    OutletID int NOT NULL,
    OrderedQty int NOT NULL,
    ReceivedQty int NOT NULL
) ON [PRIMARY];
GO
```

The 'Messages' pane at the bottom shows the message "Commands completed successfully." and "Query executed successfully." The status bar at the bottom right indicates "DESKTOP-7L40TLT\SQLEXPRESS... DESKTOP-7L40TLT\Precios... master 00:00:00 0 rows".

### 3.4. Add appropriate primary key constraints

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the database 'ABCInventory' is selected. In the center pane, several ALTER TABLE statements are displayed, each adding a primary key constraint:

```
ALTER TABLE dbo.DimDate ADD
CONSTRAINT PK_DimDate PRIMARY KEY (DateID) ON [PRIMARY];
ALTER TABLE dbo.DimProduct ADD
CONSTRAINT PK_DimProduct PRIMARY KEY (ProductID) ON [PRIMARY];
ALTER TABLE dbo.DimSupplier ADD
CONSTRAINT PK_DimSupplier PRIMARY KEY (SupplierID) ON [PRIMARY];
ALTER TABLE dbo.DimOutlet ADD
CONSTRAINT PK_DimOutlet PRIMARY KEY (OutletID) ON [PRIMARY];
ALTER TABLE dbo.ReceivingPurchaseOrderFact ADD
CONSTRAINT PK_ReceivingPurchaseOrderFact PRIMARY KEY
(SentDateID,ReceivedDateID,PurchaseOrderCode,ProductID,SupplierID,OutletID) ON [PRIMARY];
ALTER TABLE dbo.SendingPurchaseOrderFact ADD
CONSTRAINT PK_SendingPurchaseOrderFact PRIMARY KEY
(SentDateID,PurchaseOrderCode,ProductID,SupplierID,OutletID) ON [PRIMARY];
ALTER TABLE dbo.StockControlFact ADD
CONSTRAINT PK_StockControlFact PRIMARY KEY
(DateID,ProductID,SupplierID) ON [PRIMARY];
```

The 'Messages' pane at the bottom shows the message "Commands completed successfully." and "Query executed successfully." The status bar at the bottom right indicates "DESKTOP-7L40TLT\SQLEXPRESS... DESKTOP-7L40TLT\Precios... ABCInventory 00:00:00 0 rows".

### 3.5. Add appropriate foreign key constraints

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the database 'ABCInventory' is selected. In the center pane, several T-SQL commands are being run to create foreign key constraints:

```
ALTER TABLE dbo.ReceivingPurchaseOrderFact ADD  
CONSTRAINT FK_SentDateID FOREIGN KEY (SentDateID)  
REFERENCES dbo.DimDate (DateID);  
  
ALTER TABLE dbo.SendingPurchaseOrderFact ADD  
CONSTRAINT FK_ReceivedDateID FOREIGN KEY (ReceivedDateID)  
REFERENCES dbo.DimDate (DateID);  
  
ALTER TABLE dbo.ReceivingPurchaseOrderFact ADD  
CONSTRAINT FK_ProductID FOREIGN KEY (ProductID)  
REFERENCES dbo.DimProduct (ProductID);  
  
ALTER TABLE dbo.SendingPurchaseOrderFact ADD  
CONSTRAINT FK_SupplierID FOREIGN KEY (SupplierID)  
REFERENCES dbo.DimSupplier (SupplierID);  
  
ALTER TABLE dbo.SendingPurchaseOrderFact ADD  
CONSTRAINT FK_OutletID FOREIGN KEY (OutletID)  
REFERENCES dbo.DimOutlet (OutletID);
```

The status bar at the bottom right shows the command count (Ln 139), column count (Col 3), character count (Ch 3), and the word 'INS'.

## 4. Populating the Data Warehouse

The SQL Server Import and Export Wizard was used to import all the test data into the datawarehouse

### 4.1. Populate Dimensions

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the database 'ABCInventory' is selected. In the center pane, a query is running to insert data into the 'DimDate' table:

```
SELECT * FROM dbo.DimDate
```

The results grid shows 19 rows of data, representing dates from January 1, 2006, to January 1, 2007. The columns include DateID, Date, DateName, DayofWeek, DayofYear, WeekofYear, MonthName, MonthofYear, and CalendarQuarter.

### 4.2. Populate Fact Tables

Inorder to populate the fact tables with test data, a staging area named "StagingArea\_ABC" was created to house the lookup tables.

#### 4.2.1. Creating Lookup tables in the Staging Area

```
Coursework_ETL.sql - DESKTOP-7L40TLT\SQLEXPRESS.master (DESKTOP-7L40TLT\Precious (64)) - Microsoft SQL Server Management Studio
File Edit View Query Project Debug Tools Window Help
master New Query Execute Debug
Object Explorer Properties Current conn Aggregate Conn Elaps Final Nam DESK Row 0 Start State Open Connection Conn DESK Connection Conn Azan Conn Conn Not e Conn Conn 0 Conn Conn Open Disp DESK Logi DESK Serv DESK Name The name of the connection
Recovered-Coursework...LT(Precious (65)) Coursework_ETL.sql,LT(Precious (64))*
CREATE TABLE dbo.SupplierIDLookup(
SupplierName varchar(50) NOT NULL PRIMARY KEY ,
SupplierID int NOT NULL
)
CREATE TABLE dbo.ProductIDLookup(
SKU varchar(50) NOT NULL PRIMARY KEY ,
ProductID int NOT NULL,
ProductName varchar(100) NOT NULL,
SupplierName varchar(50) NOT NULL,
CostPrice decimal(10,2) NOT NULL,
RetailPrice decimal(10,2) NOT NULL,
CurrentStockLevel int NOT NULL
)
CREATE TABLE dbo.OutletIDLookup(
OutletName varchar(50) NOT NULL PRIMARY KEY ,
OutletID int NOT NULL
)
SELECT * FROM [dbo].[ProductIDLookup]
SELECT * FROM [dbo].[SentPurchaseOrders]
```

Activate Windows

Connected. (1/1)

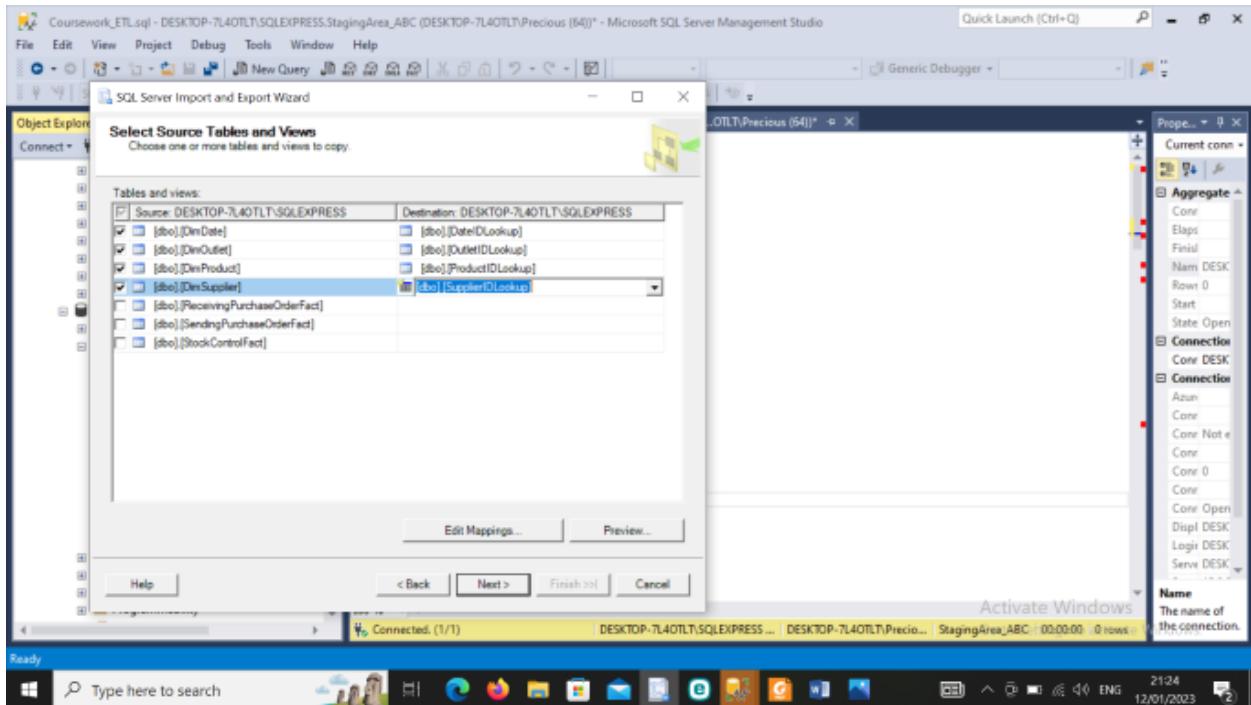
DESKTOP-7L40TLT\SQLEXPRESS ... DESKTOP-7L40TLT\Precious (64) master 00:00:00 0 rows

Ready

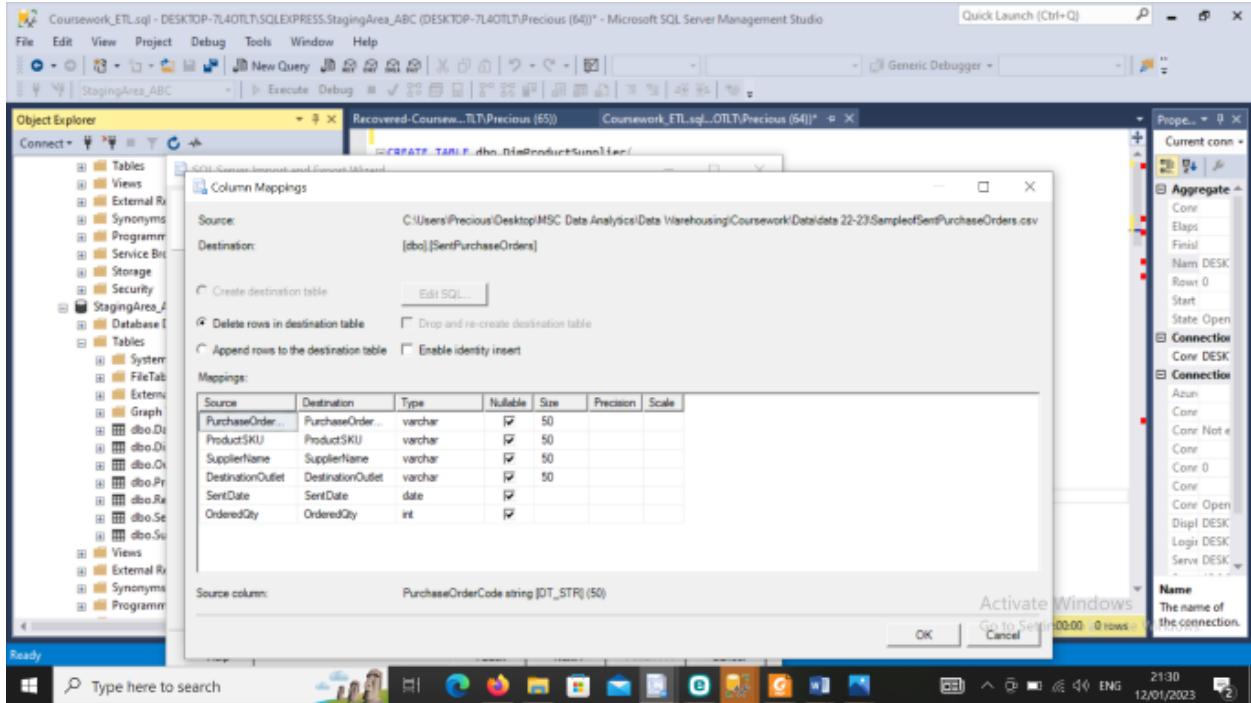
Line 24 Col 1 Ch 1 INS

Type here to search

#### 4.2.2. Populating Lookup tables

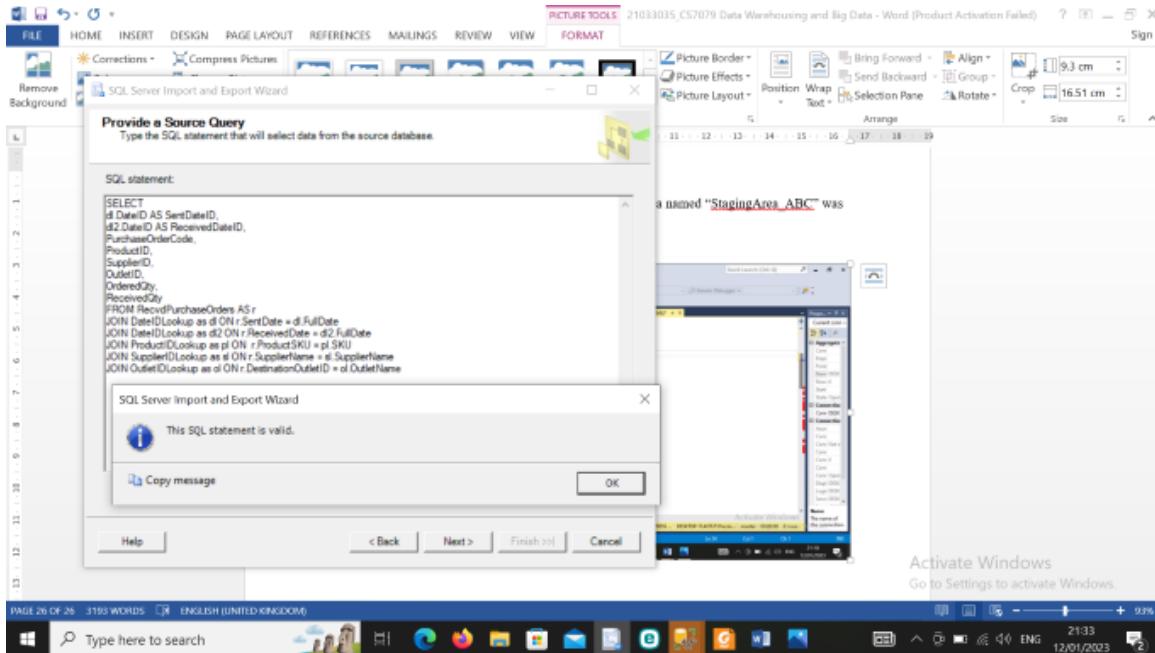


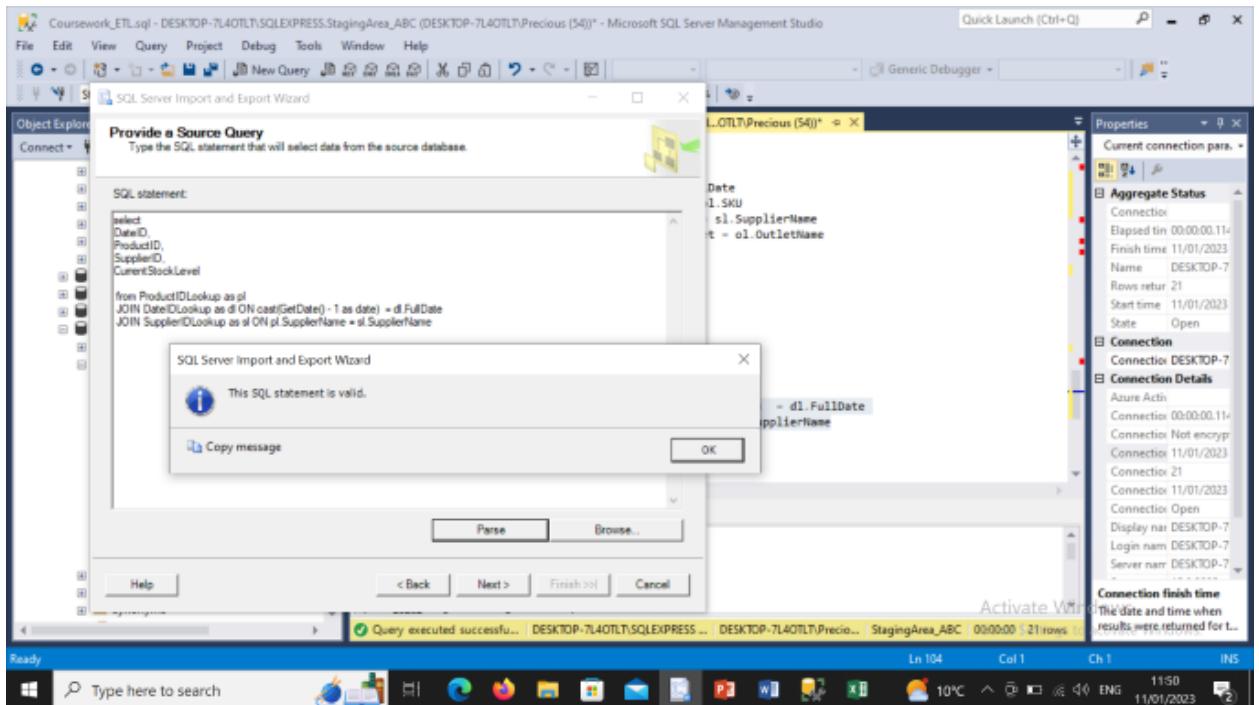
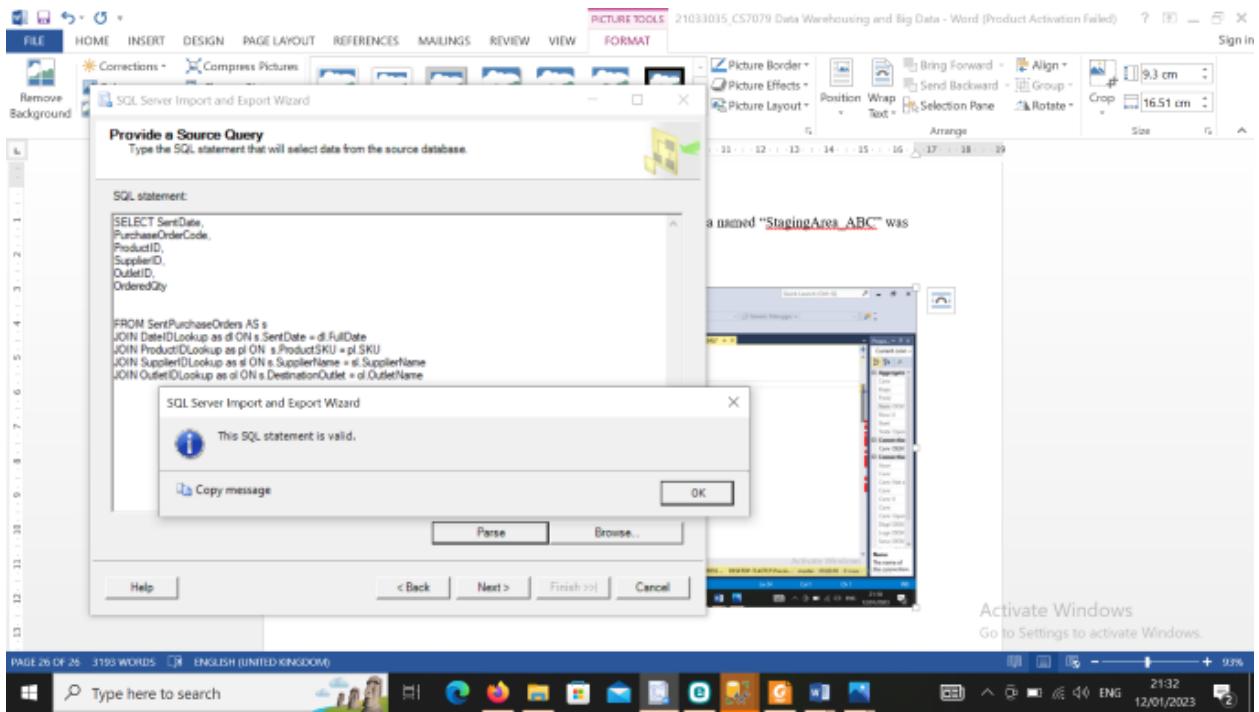
#### 4.2.3. Populating the transaction lookup table in the Staging Area



#### 4.2.4. Populating the Fact tables

Once all the lookup tables in the staging area have been populated. The test data now needs to be transferred from the staging area to the main ABCInventory database.





## 5. Test implementation

To test that the tables were fully populated, the figure below shows the results of the queries from all the tables.

Recovered-Coursework.sql - DESKTOP-TL40TLT\SQLEXPRESS.ABCInventory (DESKTOP-TL40TLT\Precious (65)) - Microsoft SQL Server Management Studio

```

File Edit View Query Project Debug Tools Window Help
New Query Execute Debug
ABCInventory
Object Explorer
Connect Current conn
Current conn
Aggregate
Conn
Elapsed 00:00
Finish 12/01
Name DESK
Row 10
Start 12/01
State Open
Connection
Conn DESK
Connection
Azur
Conn 00:00
Conn Note
Conn 12/01
Conn 10
Conn 12/01
Conn Open
Displ DESK
Login DESK
Serve DESK
Name
The name of the connection.

Recovered-Coursework..TLT\Precious (65))" < Coursework_ETL.sql..OLT\Precious (64)
JOIN OutletIDLookup as ol ON s.DestinationOutlet = ol.OutletName
select * from DimProduct
select * from [dbo].[ReceivingPurchaseOrderFact]

```

Results Messages

SendDateID	ReceivedDateID	PurchaseOrderCode	ProductID	SupplierID	OutletID	OrderedQty	ReceivedQty
1	26970	SA311016 JVC PRO	3	3	1	6	6
2	26970	SA311016 SENNHEISER PRO	1	1	1	4	4
3	26970	SA311016 SENNHEISER PRO	4	1	1	5	5
4	26970	SA311016 SONY PRO	2	2	1	2	0
5	26970	SA311016 TOSHIBA PRO	5	6	1	2	0
6	26977	SA071116 SOLOCO	9	7	1	4	0
7	26977	SA071116 SOLOCO	10	7	1	1	1
8	26987	PM0811COMPUB	8	7	1	1	1
9	26986	SA1406	6	7	1	1	1
10	26986	PM1511COMPUBA02	7	6	1	1	1

Query executed successfully.

DESKTOP-TL40TLT\SQLEXPRESS... DESKTOP-TL40TLT\Preci... ABCInventory | 00:00:00 | 10 rows

Ready Type here to search

Ln 232 Col 1 Ch 1 INS

2147 12/01/2023

Recovered-Coursework.sql - DESKTOP-TL40TLT\SQLEXPRESS.ABCInventory (DESKTOP-TL40TLT\Precious (65)) - Microsoft SQL Server Management Studio

```

File Edit View Query Project Debug Tools Window Help
New Query Execute Debug
ABCInventory
Object Explorer
Connect Current conn
Current conn
Aggregate
Conn
Elapsed 00:00
Finish 12/01
Name DESK
Row 11
Start 12/01
State Open
Connection
Conn DESK
Connection
Azur
Conn 00:00
Conn Note
Conn 12/01
Conn 11
Conn 12/01
Conn Open
Displ DESK
Login DESK
Serve DESK
Name
The name of the connection.

Recovered-Coursework..TLT\Precious (65))" < Coursework_ETL.sql..OLT\Precious (64)
JOIN OutletIDLookup as ol ON s.DestinationOutlet = ol.OutletName
select * from DimProduct
select * from [dbo].[ReceivingPurchaseOrderFact]
select * from [dbo].[SendingPurchaseOrderFact]

```

Results Messages

SendDateID	PurchaseOrderCode	ProductID	SupplierID	OutletID	OrderedQty
1	SA301015 CANON CS PO 1	16	9	1	1
2	SA301015 JVC CS PO 1	13	3	1	1
3	SA301015 JVC CS PO 1	17	3	1	1
4	SA301015 SAMSUNG CS PO 1	19	10	1	1
5	SA301015 SONY CS PO 1	20	2	1	1
6	SA301015 SONY CS PO 1	21	2	1	2
7	SA301015 SONY PO 1	12	2	1	2
8	SA301015 SAMSUNG CS PO 1	14	10	1	1
9	SA301015 TOSHIBA CS PO 1	11	6	1	3
10	SA301015 TOSHIBA CS PO 1	15	6	1	1

Query executed successfully.

DESKTOP-TL40TLT\SQLEXPRESS... DESKTOP-TL40TLT\Preci... ABCInventory | 00:00:00 | 11 rows

Ready Type here to search

Ln 234 Col 1 Ch 1 INS

2148 12/01/2023

Recovered-Coursework.sql - DESKTOP-7L40TLT\SQLEXPRESS.ABCInventory (DESKTOP-7L40TLT\Precious (65)) - Microsoft SQL Server Management Studio

```

    JOIN OutletIDLookup AS ol ON s.DestinationOutlet = ol.OutletName
    select * from DimProduct
    select * from [dbo].[ReceivingPurchaseOrderFact]
    select * from [dbo].[SendingPurchaseOrderFact]
    select * from [dbo].[StockControlFact]
  
```

DateID	ProductID	SupplierID	StockQty
1	29232	1	0
2	29232	2	3
3	29232	3	4
4	29232	4	10
5	29232	5	4
6	29232	6	1
7	29232	7	8
8	29232	8	3
9	29232	9	4
10	29232	10	3

Activate Windows

Ready Type here to search

## 6. Testing some of the Business Requirements

To ensure the functionality of the data warehouse and its alignment with business and reporting needs, SQL queries were formulated and executed to verify compliance with these requirements.

### 6.1. Analysing stock levels by brand or product type or supplier

Recovered-Coursework.sql - DESKTOP-7L40TLT\SQLEXPRESS.ABCInventory (DESKTOP-7L40TLT\Precious (65)) - Microsoft SQL Server Management Studio

```

WHERE d.FullDate >= DATEADD(m, datediff (m, 0, current_timestamp)-1, 0)
and d.FullDate < DATEADD(m, datediff (m, 0, current_timestamp)-1, 0)

Select
p.Brand,
sum(StockQty) AS StockLevel
FROM [dbo].[StockControlFact] AS s
JOIN [dbo].[DimProduct] AS p ON s.ProductID = p.ProductID
GROUP BY p.Brand
  
```

Brand	StockLevel
Hoya	13
JVC	4
MSCS	11
Rycot	34
SENNHEISER	10
SONY	3
TOSHIBA	12

Activate Windows

Ready Type here to search

```

Recovered-Coursework.sql - DESKTOP-7L40TLT\SQLEXPRESS.ABCInventory (DESKTOP-7L40TLT\Precious (65)) - Microsoft SQL Server Management Studio
File Edit View Query Project Debug Tools Window Help
ABCInventory | Execute Debug | Quick Launch (Ctrl+Q) | Generic Debugger |
Object Explorer | Properties | Current conn |
Connect | ABCInventory | Aggregate | Conn | Elaps 0:00 | Final 12/01 | Nam DESK | Row 0 | Start 12/01 | State Open |
Databases | ABCInventory | Connection | Conn DESK | Conn Not e | Conn 12/01 | Conn 0 | Conn 12/01 | Conn Open | Displ DESK | Logon DESK | Serve DESK |
Tables | ABCInventory | Connection | Conn Azur | Conn 0:00 | Conn Not e | Conn 12/01 | Conn 0 | Conn 12/01 | Conn Open | Displ DESK | Logon DESK | Serve DESK |
Views | ABCInventory | Connection | Conn 12/01 | Conn Open | Displ DESK | Logon DESK | Serve DESK |
External Resources | ABCInventory | Connection | Conn 12/01 | Conn Open | Displ DESK | Logon DESK | Serve DESK |
Synonyms | ABCInventory | Connection | Conn 12/01 | Conn Open | Displ DESK | Logon DESK | Serve DESK |
Programmability | ABCInventory | Connection | Conn 12/01 | Conn Open | Displ DESK | Logon DESK | Serve DESK |
Service Broker | ABCInventory | Connection | Conn 12/01 | Conn Open | Displ DESK | Logon DESK | Serve DESK |
Storage | ABCInventory | Connection | Conn 12/01 | Conn Open | Displ DESK | Logon DESK | Serve DESK |
Security | ABCInventory | Connection | Conn 12/01 | Conn Open | Displ DESK | Logon DESK | Serve DESK |
AdventureWorks2014 | ABCInventory | Connection | Conn 12/01 | Conn Open | Displ DESK | Logon DESK | Serve DESK |
Ready
Type here to search
DESKTOP-7L40TLT\SQLEXPRESS... DESKTOP-7L40TLT\Precious (65) 0:00:00 0 rows
Ln 249 Col 69 Ch 69 INS
Activate Windows
100 %
Results Messages
DateID ProductID StockQty
Query executed successfully.

```

## 7. Exporting the data warehouse to a flat file destination.

The SQL Server import and export wizard was used to export the datawarehouse tables to external files. The Datasource is the ABCInventory Database while the destination is Flat File Destination

Choose a Destination

Specify where to copy data to.

Destination: Flat File Destination

Select a file and specify the file properties and the file format.

File name: Data Analytics\Coursework\Data\data 22-23\DimOutlet\_DW.csv

Locale: English (United Kingdom)

Code page: 1252 (ANSI - Latin I)

Format: Delimited

Text qualifier: <none>

Column names in the first data row

Excel (Product Activation Failed)

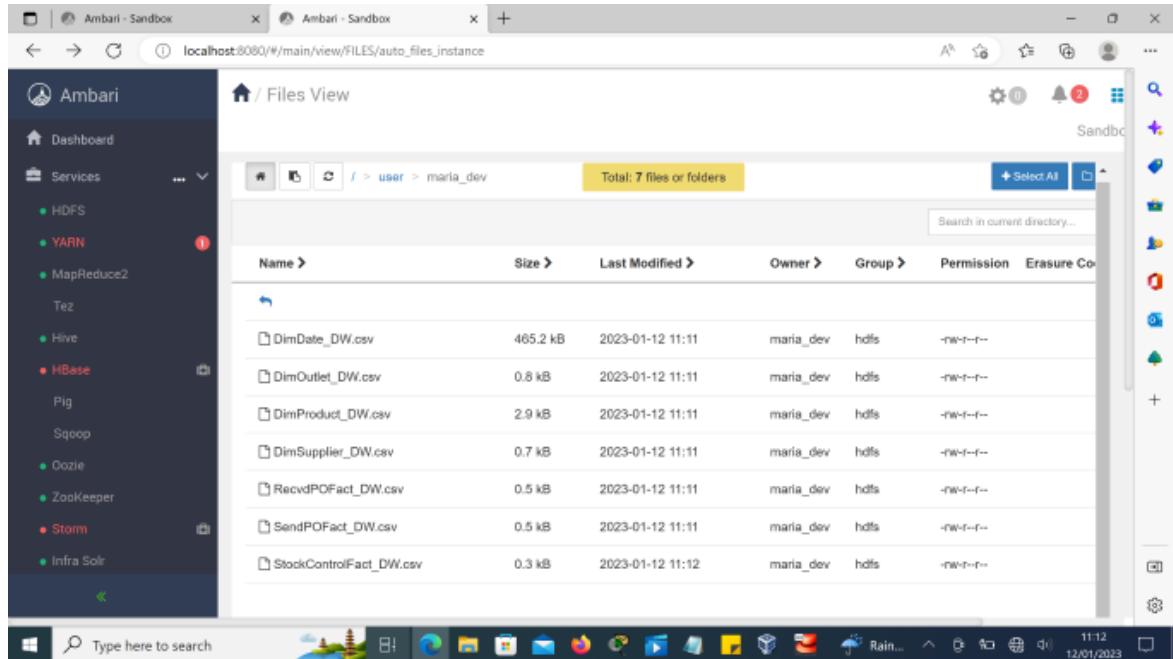
scriptioPhone	Email	Fax	FirstLineAddress	PostCode	City	State	Country
UNHEIS	UNKN	UNKN	UNKN	UNKN	UNKN	UNKN	U
NY	UNKN	UNKN	UNKN	UNKN	UNKN	UNKN	U
	UNKN	UNKN	UNKN	UNKN	UNKN	UNKN	U
MA	UNKN	UNKN	UNKN	UNKN	UNKN	UNKN	U
SHIBA	UNKN	UNKN	UNKN	UNKN	UNKN	UNKN	U
CS	UNKN	UNKN	UNKN	UNKN	UNKN	UNKN	U
E	UNKN	UNKN	UNKN	UNKN	UNKN	UNKN	U
NON	UNKN	UNKN	UNKN	UNKN	UNKN	UNKN	U
Samsung	UNKN	UNKN	UNKN	UNKN	UNKN	UNKN	U

SQL Server Writing to Text File 4:38

24K views • 8 years ago

## 8. Migrating Data file to Apache HDFS

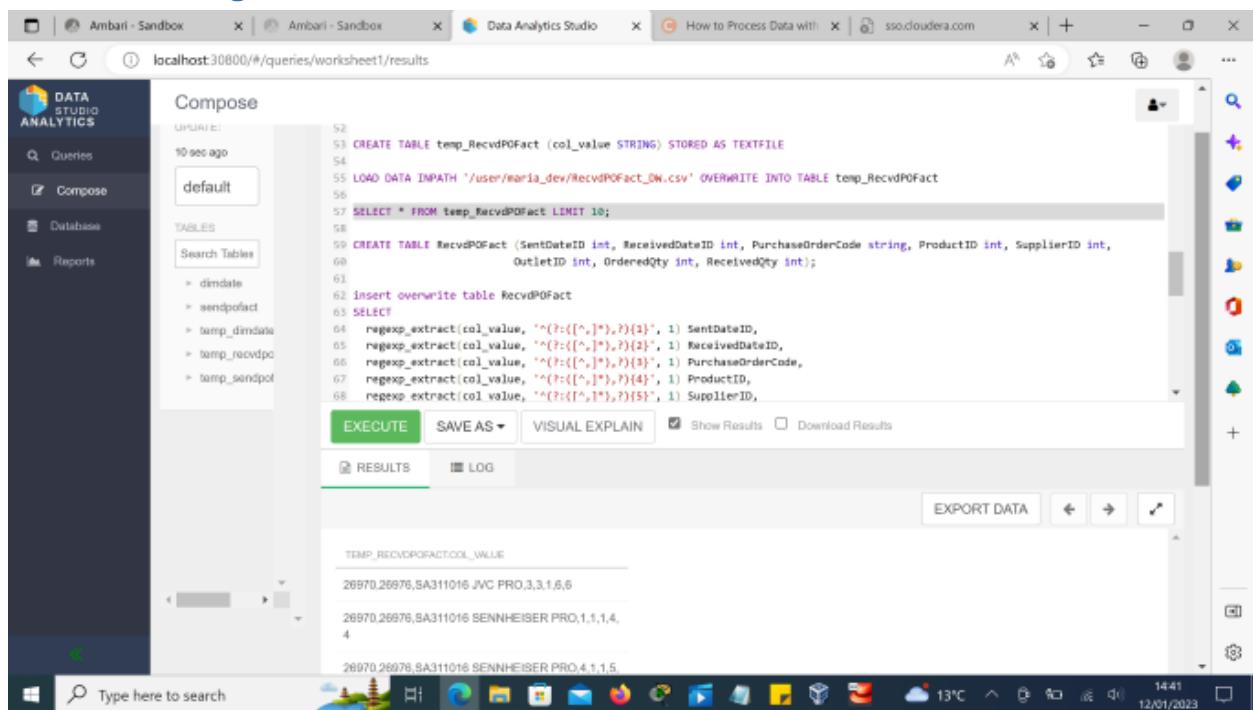
This was done by using the Files View and browsing to a preferred location or folder to upload the data files to. In this case, the path ‘/user/maria\_dev’ was navigated to and write permission access was granted to the maria\_dev user.



The screenshot shows the Ambari interface with the 'Files View' selected. The left sidebar lists various services like HDFS, YARN, and Tez. The main area shows a list of files in the '/user/maria\_dev' directory. The files are:

Name	Size	Last Modified	Owner	Group	Permission
DimDate_DW.csv	465.2 kB	2023-01-12 11:11	maria_dev	hdfs	-rw-r--r--
DimOutlet_DW.csv	0.8 kB	2023-01-12 11:11	maria_dev	hdfs	-rw-r--r--
DimProduct_DW.csv	2.9 kB	2023-01-12 11:11	maria_dev	hdfs	-rw-r--r--
DimSupplier_DW.csv	0.7 kB	2023-01-12 11:11	maria_dev	hdfs	-rw-r--r--
RevcvdPOFact_DW.csv	0.5 kB	2023-01-12 11:11	maria_dev	hdfs	-rw-r--r--
SendPOFact_DW.csv	0.5 kB	2023-01-12 11:11	maria_dev	hdfs	-rw-r--r--
StockControlFact_DW.csv	0.3 kB	2023-01-12 11:12	maria_dev	hdfs	-rw-r--r--

## 9. Creating a data structure in hive



The screenshot shows the Data Analytics Studio interface with a query being composed. The code is:

```
52 CREATE TABLE temp_RevcvdPOFact (col_value STRING) STORED AS TEXTFILE
53
54 LOAD DATA INPATH '/user/maria_dev/RevcvdPOFact_DW.csv' OVERWRITE INTO TABLE temp_RevcvdPOFact
55
56 SELECT * FROM temp_RevcvdPOFact LIMIT 10;
```

The results pane shows the output of the query:

TEMP_RECVDPFACT.COL_VALUE
26970.26976,SA311016 JVC PRO.3,3,1,6,6
26970.26976,SA311016 SENNHEISER PRO,1,1,1,4,
4
26970.26976,SA311016 SENNHEISER PRO.4,1,1,5,

The screenshot shows the Data Analytics Studio interface. In the left sidebar, under the 'Compose' section, there is a list of tables: dimdate, recvdpofact, sendpofact, temp\_dimdate, temp\_recvdpo, and temp\_sendpo. The main area displays a Pig script:

```
69  regexp_extract(col_value, '^([^\"]*)(\".*\")', 1) outletID,
70  regexp_extract(col_value, '^([^\"]*)(\".*\")', 1) orderedQty,
71  regexp_extract(col_value, '^([^\"]*)(\".*\")', 1) ReceivedQty
72
73 from temp_RecvdPOFact
74
75 select * from RecvdPOFact;
76
77
```

Below the script are buttons for 'EXECUTE', 'SAVE AS', 'VISUAL EXPLAIN', and checkboxes for 'Show Results' and 'Download Results'. The 'RESULTS' tab is selected, showing the following data:

RECDVDOFACT.SENTDATEID	RECDVDOFACT.RECEIVEDATEID	RECDVDOFACT.PURCHASEORDERCODE	RECDVDOFACT.PRODUCTID	RECDVDOFACT.SUPPLIERID
26970	26976	SA311016 JVC PRO	3	3
26970	26976	SA311016 SENNHEISER PRO	1	1
26970	26976	SA311016 SENNHEISER PRO	4	1
26970	26976	SA311016 SONY PRO	2	2
26970	26976	SA311016 TOSHIBA PRO	5	6
26977	26988	SA071116 SOLOCO	9	7
26977	26988	SA071116 SOLOCO	10	7

## 10. Demonstrating the use of Pig

The figure below shows how to load data and create data structure in pig and also using the DUMP command to view data.

### 10.1. DUMP

The screenshot shows a browser window with multiple tabs open. The active tab is titled 'maria\_dev' and shows the following Pig script:

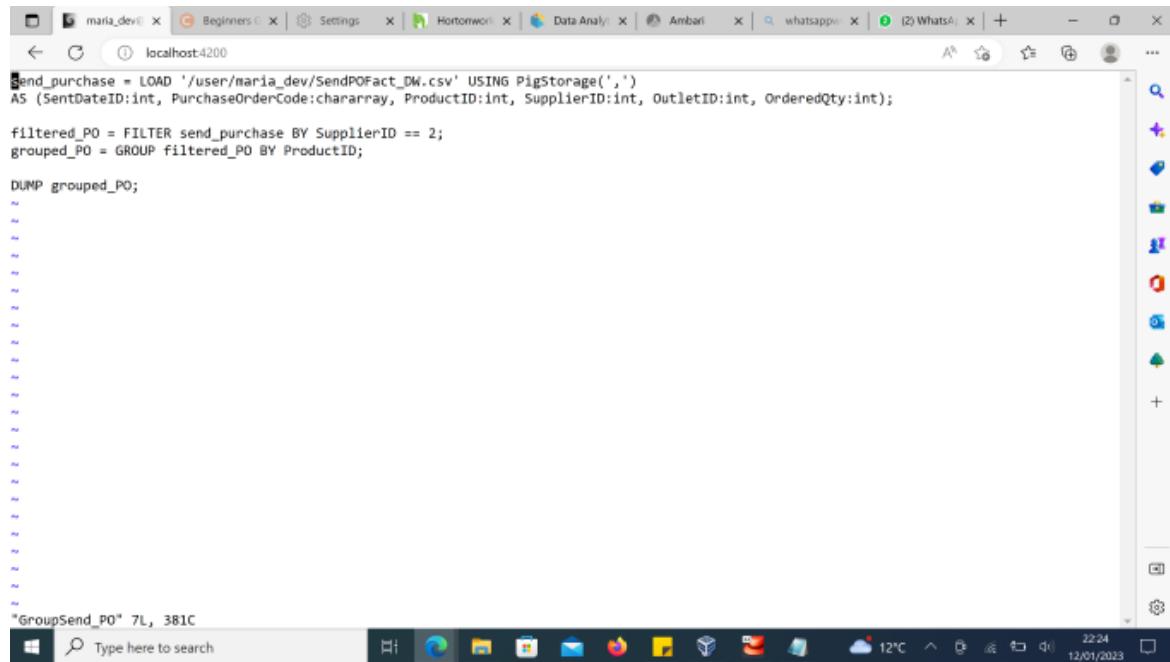
```
send_purchase = LOAD '/user/maria_dev/SendPOFact_DW.csv' USING PigStorage(',');
AS (SendDateID:int, PurchaseOrderCode:chararray, ProductID:int, SupplierID:int, OutletID:int, OrderedQty:int);
DESCRIBE send_purchase;
DUMP send_purchase;
```

The output of the DUMP command is displayed below the script:

```
"Send_PO" 4L, 234C
```

## 10.2. FILTER and GROUP.

The below script shows how the loaded data was filtered by SupplierID and grouped by the ProductID.



```
send_purchase = LOAD '/user/maria_dev/SendPOFact_DW.csv' USING PigStorage(',') AS (SentDateID:int, PurchaseOrderCode:chararray, ProductID:int, SupplierID:int, OutletID:int, OrderedQty:int);
filtered_PO = FILTER send_purchase BY SupplierID == 2;
grouped_PO = GROUP filtered_PO BY ProductID;
DUMP grouped_PO;
```

"GroupSend\_PO" 7L, 381C

The result of the above script is as follows showing the result of filtering only the supplier whose ID is 2 and grouped all the products.

DAG Plan:

```

Tez vertex scope-29 -> Tez vertex scope-30,
Tez vertex scope-30

```

Vertex Stats:

VertexId	Parallelism	TotalTasks	InputRecords	ReduceInputRecords	OutputRecords	FileBytesRead	FileBytesWritten	HdfsBytesRead	HdfsBytesWritten
scope-29	1	1	11	0	3	64	127	431	
		0	filtered_PO,grouped_PO,send_purchase						
scope-30	1	1	0	3	3	127	0	0	
	125	GROUP_BY	hdfs://sandbox-hdp.hortonworks.com:8020/tmp/temp-1439952442/tmp1927856432,						

Input(s):

```
Successfully read 11 records (431 bytes) from: "/user/maria_dev/SendPOFact_DW.csv"
```

Output(s):

```
Successfully stored 3 records (125 bytes) in: "hdfs://sandbox-hdp.hortonworks.com:8020/tmp/temp-1439952442/tmp1927856432"
```

```

2023-01-12 18:15:58,591 [main] INFO org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input files to process : 1
2023-01-12 18:15:58,592 [main] INFO org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
(12,{{(26603,SA301015 SONY PO 1,12,2,1,2)})
(20,{{(26603,SA301015 SONY CS PO 1,20,2,1,1)})
(21,{{(26603,SA301015 SONY CS PO 1,21,2,1,2)})
```

2023-01-12 18:15:58,881 [main] INFO org.apache.pig.Main - Pig script completed in 52 seconds and 500 milliseconds (52500 ms)

2023-01-12 18:15:58,901 [main] INFO org.apache.pig.backend.hadoop.executionengine.tez.TezLauncher - Shutting down thread pool

2023-01-12 18:15:58,926 [shutdown-hook-0] INFO org.apache.pig.backend.hadoop.executionengine.tez.TezSessionManager - Shutting down Tez session org.apache.tez.client.TezClient@f1fa926e

2023-01-12 18:15:58,927 [shutdown-hook-0] INFO org.apache.tez.client.TezClient - Shutting down Tez Session, sessionName=PigLatin:GroupSend\_PO, applicationId=application\_1673519522781\_0027

[maria\_dev@sandbox-hdp ~]\$ vi SortSend\_PO

### 10.3. Join Data

The script below shows how to join two tables together and display the result

```

send_purchase = LOAD '/user/maria_dev/SendPOFact_DW.csv' USING PigStorage(',');
AS (SentDateID:int, PurchaseOrderCode:chararray, ProductID:int, SupplierID:int, OutletID:int, OrderedQty:int);

dimdate = LOAD '/user/maria_dev/DimDate_DW.csv' USING PigStorage(',');
AS (DateID:int, FullDate:chararray, DateName:chararray, DayOfWeek:int, DayNameOfWeek:chararray, DayOfYear:int, WeekOfYear:int, MonthName:chararray, MonthOfYear:int, CalendarQuarter:int,CalendarYear:int);

dimdate_convert = FOREACH dimdate GENERATE DateID, ToDate(FullDate) AS FullDate, ToDate(DateName)AS DateName, DayOfWeek, DayNameOfWeek , DayOfYear, WeekOfYear, MonthName, MonthOfYear, CalendarQuarter,CalendarYear;

join_data = JOIN send_purchase BY (SentDateID), dimdate_convert BY (DateID);
DESCRIBE join_data;
DUMP join_data;

```

"JoinDateSendPO" 13L, 980C

maria\_dev@sandbox-hdp:~ - Sh | Beginners Guide to Apache Pig | Settings | New tab | +

localhost:4200

```
Details at logfile: /home/maria_dev/pig_1673543907487.log
2023-01-12 17:18:32,034 [main] INFO org.apache.pig.Main - Pig script completed in 4 seconds and 759 milliseconds (4759 ms)
[maria_dev@sandbox-hdp ~]$ vi joinDateSendPO
[maria_dev@sandbox-hdp ~]$ pig -f JoinDateSendPO | tee -a joinAttributes.txt
WARNING: HADOOP_PREFIX has been replaced by HADOOP_HOME. Using value of HADOOP_PREFIX.
23/01/12 17:19:46 INFO pig.ExecTypeProvider: Trying ExecType : LOCAL
23/01/12 17:19:46 INFO pig.ExecTypeProvider: Trying ExecType : MAPREDUCE
23/01/12 17:19:46 INFO pig.ExecTypeProvider: Trying ExecType : TEZ_LOCAL
23/01/12 17:19:46 INFO pig.ExecTypeProvider: Trying ExecType : TEZ
23/01/12 17:19:46 INFO pig.ExecTypeProvider: Picked TEZ as the ExecType
2023-01-12 17:19:46,348 [main] INFO org.apache.pig.Main - Apache Pig version 0.16.0.3.0.1.0-187 (rUnversioned directory) compiled Sep 19 2018, 10:13:33
2023-01-12 17:19:46,348 [main] INFO org.apache.pig.Main - Logging error messages to: /home/maria_dev/pig_1673543986345.log
2023-01-12 17:19:47,069 [main] INFO org.apache.pig.impl.util.Utils - Default bootup file /home/maria_dev/.pigbootup not found
2023-01-12 17:19:47,297 [main] INFO org.apache.pig.backend.hadoop.executionengine.HExecutionEngine - Connecting to hadoop file system at: hdfs://sandbox-hdp.hortonworks.com:8020
2023-01-12 17:19:48,074 [main] INFO org.apache.pig.PigServer - Pig Script ID for the session: PIG-JoinDateSendPO-284a2fe1-244d-46d7-8967-10b9c3c578a2
2023-01-12 17:19:48,074 [main] WARN org.apache.pig.PigServer - ATS is disabled since yarn.timeline-service.enabled set to false
join_data: {send_purchase::SendPurchaseID: int, send_purchase::PurchaseOrderCode: chararray, send_purchase::ProductID: int, send_purchase::SupplierID: int, send_purchase::OutletID: int, send_purchase::OrderedQty: int, dimdate_convert::DateID: int, dimdate_convert::FullDate: date, dimdate_convert::DateName: datetime, dimdate_convert::DayOfWeek: int, dimdate_convert::DayNameOfWeek: chararray, dimdate_convert::DayOfYear: int, dimdate_convert::WeekOfYear: int, dimdate_convert::MonthName: chararray, dimdate_convert::MonthOfYear: int, dimdate_convert::CalendarQuarter: int, dimdate_convert::CalendarYear: int}
2023-01-12 17:19:49,589 [main] INFO org.apache.pig.Main - Pig script completed in 3 seconds and 479 milliseconds (3479 ms)
[maria_dev@sandbox-hdp ~]$ cat joinAttributes.txt
join_data: {send_purchase::SendPurchaseID: int, send_purchase::PurchaseOrderCode: chararray, send_purchase::ProductID: int, send_purchase::SupplierID: int, send_purchase::OutletID: int, send_purchase::OrderedQty: int, dimdate_convert::DateID: int, dimdate_convert::FullDate: date, dimdate_convert::DateName: datetime, dimdate_convert::DayOfWeek: int, dimdate_convert::DayNameOfWeek: chararray, dimdate_convert::DayOfYear: int, dimdate_convert::WeekOfYear: int, dimdate_convert::MonthName: chararray, dimdate_convert::MonthOfYear: int, dimdate_convert::CalendarQuarter: int, dimdate_convert::CalendarYear: int}
```

17:37 12/01/2023

The result of the join tables – Dimdate and SendPurchaseOrder is as follows:

maria\_dev@sandbox-hdp:~ - Sh | Beginners Guide to Apache Pig | Settings | Hortonworks Sandbox with Ambari - Sandbox | Ambari - Sandbox | - | +

localhost:4200

```
scope-70          1      1      7671          0      7671      112     173608    476237
      0 dimdate,dimdate_convert,join_data
scope-71          2      1      0      7682      11     218162      0      0
      1071 join_data HASH_JOIN      hdfs://sandbox-hdp.hortonworks.com:8020/tmp/temp1677549545/tmp-1729501416,
```

**Input(s):**  
Successfully read 7671 records (476237 bytes) from: "/user/maria\_dev/DimDate\_DW.csv"  
Successfully read 11 records (431 bytes) from: "/user/maria\_dev/SendPOFact\_DW.csv"

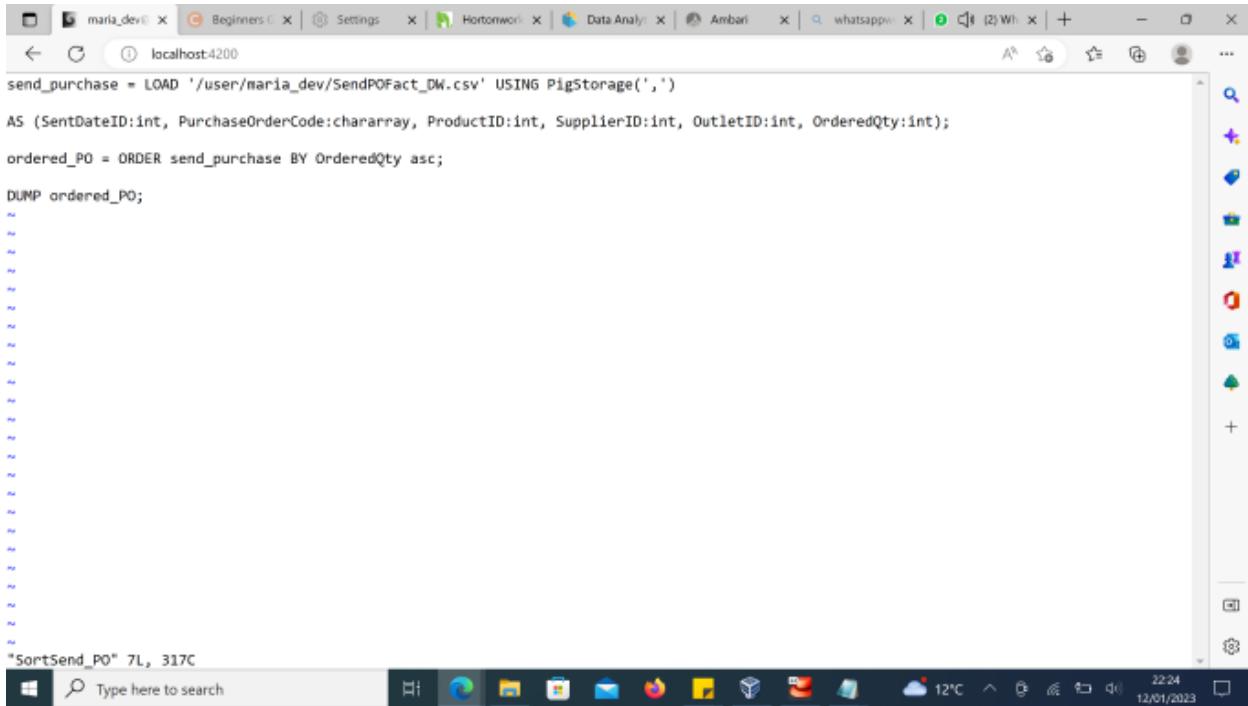
**Output(s):**  
Successfully stored 11 records (1071 bytes) in: "hdfs://sandbox-hdp.hortonworks.com:8020/tmp/temp1677549545/tmp-1729501416"

2023-01-12 17:48:22,771 [main] INFO org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input files to process : 1
2023-01-12 17:48:22,771 [main] INFO org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
(26603,SA301015 CANON CS PO 1,16,9,1,26603,2015-10-30T00:00:00.000Z,2015-10-30T00:00:00.000Z,6,Friday,303,44,October,10,4,2015)
(26603,SA301015 JVC CS PO 1,13,3,1,26603,2015-10-30T00:00:00.000Z,2015-10-30T00:00:00.000Z,6,Friday,303,44,October,10,4,2015)
(26603,SA301015 JVC CS PO 1,17,3,1,26603,2015-10-30T00:00:00.000Z,2015-10-30T00:00:00.000Z,6,Friday,303,44,October,10,4,2015)
(26603,SA301015 SAMSUNG CS PO 1,19,10,1,26603,2015-10-30T00:00:00.000Z,2015-10-30T00:00:00.000Z,6,Friday,303,44,October,10,4,2015)
(26603,SA301015 SONY CS PO 1,20,2,1,26603,2015-10-30T00:00:00.000Z,2015-10-30T00:00:00.000Z,6,Friday,303,44,October,10,4,2015)
(26603,SA301015 SONY CS PO 1,21,2,1,2,26603,2015-10-30T00:00:00.000Z,2015-10-30T00:00:00.000Z,6,Friday,303,44,October,10,4,2015)
(26603,SA301015 SONY PO 1,12,2,1,2,26603,2015-10-30T00:00:00.000Z,2015-10-30T00:00:00.000Z,6,Friday,303,44,October,10,4,2015)
(26603,SA301015 SAMSUNG CS PO 1,14,10,1,1,26603,2015-10-30T00:00:00.000Z,2015-10-30T00:00:00.000Z,6,Friday,303,44,October,10,4,2015)
(26603,SA301015 TOSHIBA CS PO 1,11,6,1,3,26603,2015-10-30T00:00:00.000Z,2015-10-30T00:00:00.000Z,6,Friday,303,44,October,10,4,2015)
(26603,SA301015 TOSHIBA CS PO 1,15,6,1,1,26603,2015-10-30T00:00:00.000Z,2015-10-30T00:00:00.000Z,6,Friday,303,44,October,10,4,2015)
(26603,SA301015 TOSHIBA CS PO 1,18,6,1,5,26603,2015-10-30T00:00:00.000Z,2015-10-30T00:00:00.000Z,6,Friday,303,44,October,10,4,2015)
2023-01-12 17:48:23,164 [main] INFO org.apache.pig.Main - Pig script completed in 58 seconds and 284 milliseconds (58284 ms)
2023-01-12 17:48:23,180 [main] INFO org.apache.pig.backend.hadoop.executionengine.tez.TezLauncher - Shutting down thread pool
2023-01-12 17:48:23,219 [shutdown-hook-0] INFO org.apache.pig.backend.executionengine.tez.TezSessionManager - Shutting down Tez session org.apache.tez.client.TezClient@408bb08f
2023-01-12 17:48:23,220 [shutdown-hook-0] INFO org.apache.tez.client.TezClient - Shutting down Tez Session, sessionName=PigLatin:JoinDateSendPO, applicationId=application\_1673519522781\_0026
[maria\_dev@sandbox-hdp ~]\$ 3-

18:04 12/01/2023

## 10.4. SORT

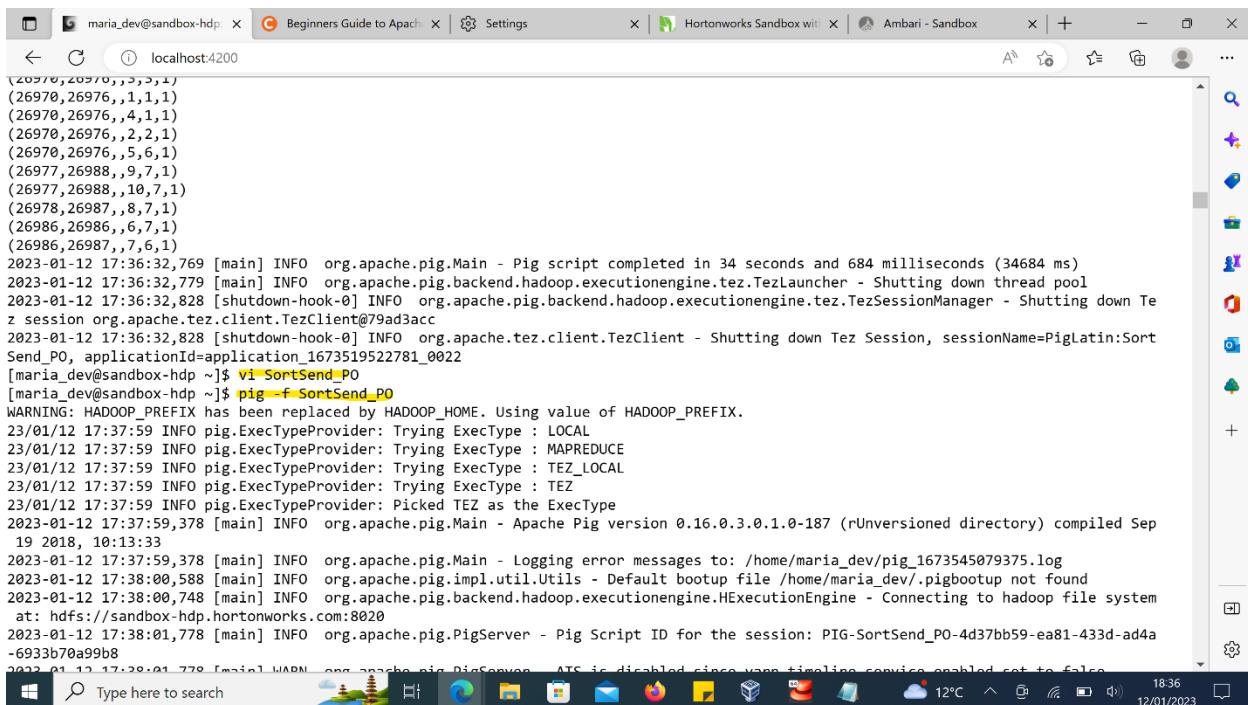
Loaded data can be ordered or sorted by using the ORDER command. The script below shows how to order a table.



A screenshot of a Microsoft Edge browser window. The address bar shows 'localhost:4200'. The page content displays a Pig Latin script:

```
send_purchase = LOAD '/user/maria_dev/SendPOFact_DW.csv' USING PigStorage(',');
AS (SentDateID:int, PurchaseOrderCode:chararray, ProductID:int, SupplierID:int, OutletID:int, OrderedQty:int);
ordered_PO = ORDER send_purchase BY OrderedQty asc;
DUMP ordered_PO;
```

The status bar at the bottom right indicates the date as 12/01/2023 and the time as 22:24.



A screenshot of a Microsoft Edge browser window. The address bar shows 'localhost:4200'. The page content displays the output of the Pig Latin script, showing a list of tuples:

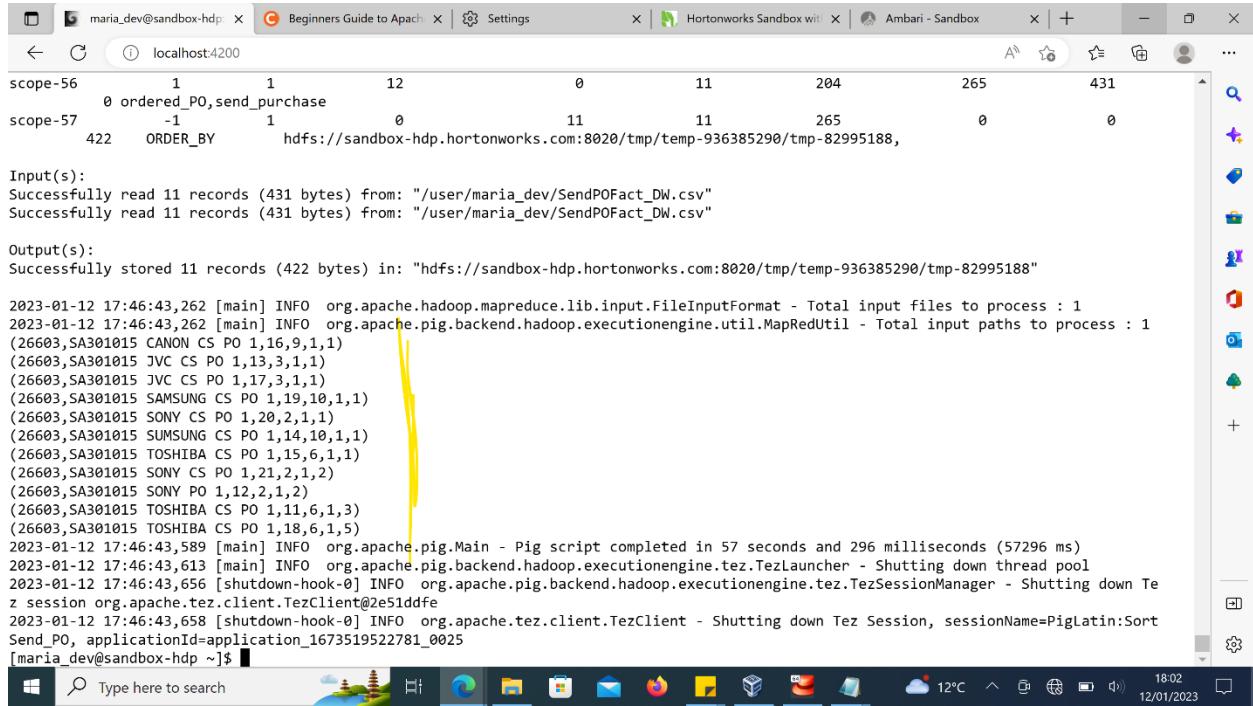
```
(26970,26976,,1,1,1)
(26970,26976,,4,1,1)
(26970,26976,,2,2,1)
(26970,26976,,5,6,1)
(26977,26988,,9,7,1)
(26977,26988,,10,7,1)
(26978,26987,,8,7,1)
(26986,26986,,6,7,1)
(26986,26987,,7,6,1)
```

Below the data, the terminal logs show the completion of the script and the shutdown of the Tez session:

```
2023-01-12 17:36:32,769 [main] INFO org.apache.pig.Main - Pig script completed in 34 seconds and 684 milliseconds (34684 ms)
2023-01-12 17:36:32,779 [main] INFO org.apache.pig.backend.hadoop.executionengine.tez.TezLauncher - Shutting down thread pool
2023-01-12 17:36:32,828 [shutdown-hook-0] INFO org.apache.pig.backend.hadoop.executionengine.tez.TezSessionManager - Shutting down Tez session org.apache.tez.client.TezClient@79ad3acc
2023-01-12 17:36:32,828 [shutdown-hook-0] INFO org.apache.tez.client.TezClient - Shutting down Tez Session, sessionName=PigLatin:SortSend_PO, applicationId=application_1673519522781_0022
[maria_dev@sandbox-hdp ~]$ vi SortSend_PO
[maria_dev@sandbox-hdp ~]$ pig -f SortSend_PO
WARNING: HADOOP_PREFIX has been replaced by HADOOP_HOME. Using value of HADOOP_PREFIX.
23/01/12 17:37:59 INFO pig.ExecTypeProvider: Trying ExecType : LOCAL
23/01/12 17:37:59 INFO pig.ExecTypeProvider: Trying ExecType : MAPREDUCE
23/01/12 17:37:59 INFO pig.ExecTypeProvider: Trying ExecType : TEZ_LOCAL
23/01/12 17:37:59 INFO pig.ExecTypeProvider: Trying ExecType : TEZ
23/01/12 17:37:59 INFO pig.ExecTypeProvider: Picked TEZ as the ExecType
2023-01-12 17:37:59,378 [main] INFO org.apache.pig.Main - Apache Pig version 0.16.0.3.0.1.0-187 (rUnversioned directory) compiled Sep 19 2018, 10:13:33
2023-01-12 17:37:59,378 [main] INFO org.apache.pig.Main - Logging error messages to: /home/maria_dev/pig_1673545079375.log
2023-01-12 17:38:00,588 [main] INFO org.apache.pig.impl.util.Utils - Default bootup file /home/maria_dev/.pigbootup not found
2023-01-12 17:38:00,748 [main] INFO org.apache.pig.backend.hadoop.executionengine.HExecutionEngine - Connecting to hadoop file system at: hdfs://sandbox-hdp.hortonworks.com:8020
2023-01-12 17:38:01,778 [main] INFO org.apache.pig.PigServer - Pig Script ID for the session: PIG-SortSend_PO-4d37bb59-ea81-433d-ad4a-693b70a99b8
2023-01-12 17:38:01,778 [main] WARN org.apache.pig.PigServer - ATS is disabled since yarn.timeline.service.enabled set to false
```

The status bar at the bottom right indicates the date as 12/01/2023 and the time as 18:36.

The result below shows an ordered table by OrderedQty which is the last column in ascending order.



```

maria_dev@sandbox-hdp ~$ C Beginners Guide to Apache Pig | Settings | Hortonworks Sandbox with Ambari - Sandbox | + | - | X
localhost:4200
Input(s):
Successfully read 11 records (431 bytes) from: "/user/maria_dev/SendPOFact_DW.csv"
Successfully read 11 records (431 bytes) from: "/user/maria_dev/SendPOFact_DW.csv"

Output(s):
Successfully stored 11 records (422 bytes) in: "hdfs://sandbox-hdp.hortonworks.com:8020/tmp/temp-936385290/tmp-82995188"

2023-01-12 17:46:43,262 [main] INFO org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input files to process : 1
2023-01-12 17:46:43,262 [main] INFO org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
(26603,SA301015 CANON CS PO 1,16,9,1,1)
(26603,SA301015 JVC CS PO 1,13,3,1,1)
(26603,SA301015 JVC CS PO 1,17,3,1,1)
(26603,SA301015 SAMSUNG CS PO 1,19,10,1,1)
(26603,SA301015 SONY CS PO 1,20,2,1,1)
(26603,SA301015 SAMSUNG CS PO 1,14,10,1,1)
(26603,SA301015 TOSHIBA CS PO 1,15,6,1,1)
(26603,SA301015 SONY CS PO 1,21,2,1,2)
(26603,SA301015 SONY PO 1,12,21,1,2)
(26603,SA301015 TOSHIBA CS PO 1,11,6,1,3)
(26603,SA301015 TOSHIBA CS PO 1,18,6,1,5)
2023-01-12 17:46:43,589 [main] INFO org.apache.pig.Main - Pig script completed in 57 seconds and 296 milliseconds (57296 ms)
2023-01-12 17:46:43,613 [main] INFO org.apache.pig.backend.hadoop.executionengine.tez.TezLauncher - Shutting down thread pool
2023-01-12 17:46:43,656 [shutdown-hook-0] INFO org.apache.pig.backend.hadoop.executionengine.tez.TezSessionManager - Shutting down Tez session org.apache.tez.client.TezClient@2e51ddfe
2023-01-12 17:46:43,658 [shutdown-hook-0] INFO org.apache.tez.client.TezClient - Shutting down Tez Session, sessionName=PigLatin:Sort Send_P0, applicationId=application_1673519522781_0025
[maria_dev@sandbox-hdp ~]$ 

```

## 11. REFLECTIVE CONCLUSION

Overall, I have learnt the following:

- How to effectively define and translate business and reporting requirements.
- How to identify and define the grain level of a data warehouse project.
- How to define dimensions and their attribute.
- How to define measures of a fact table.
- How to design a data warehouse schema to conform to the identified business requirements.
- How to effectively use SSMS to create database and tables.
- How to effectively populate dimension tables with test data using SSMS.
- How to effectively create lookup tables and populate fact tables using SSMS.
- How to use Apache HDFS to upload external file.
- How to use the Data Analytics Studio to create data structure and load data in Hive
- How to use Pig to load and manipulate data.