**Title**: Integrate Hadoop eco system with My sql database

**Autho**r: Rajendra Prasad T

**Track Category**: Design

**Subcategory**: Hadoop,Mysql database

**Tags**: Hadoop,Mysql database

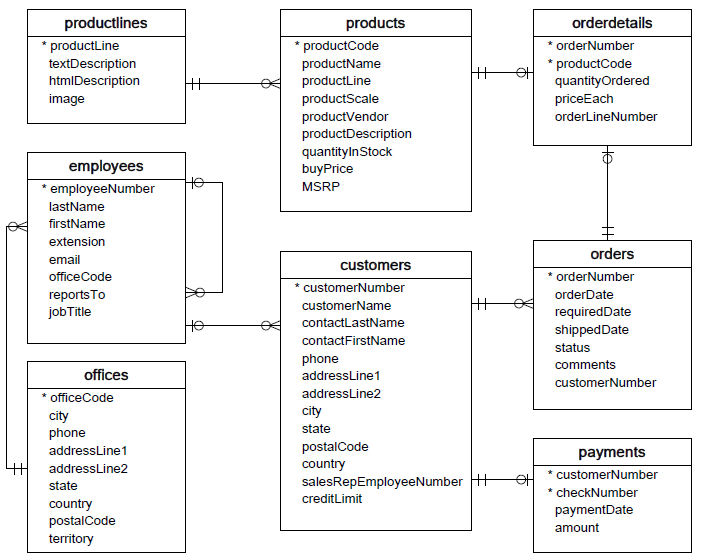
**Read Time**: 8-10 min

**Banner Image**: https://lh3.googleusercontent.com/hremWFE1HIUSdeFgBJFWDEICZQDr\_zsxOKqaKgvpp4OdR19SMrW16bj-JPYp90CWQ-wom5I=s170

**Objectives**

The objective is enable participants to get a feel of working with production datasets and analyzing data using tools in the Hadoop ecosystem. This POC requires extracting required data from MySQL database using Sqoop, loading into HDFS, Hive Tables, and executing queries in spark to generate summary tables.

**Classic Model Schema:**



**Assignment:**

a. Find the total sales generated by sales representatives by month

The output should have the following columns:

Sales Rep Code, Sales Rep Name, Year, Month, Sales

b. Find the total sales generated by each product by month

The output should have the following columns:

Product Code, Product Name, Product Line, year, month, sales

c. Find the total order amount and payment amount for each customer

The output should have the following columns:

Customer Number, Customer Name, Order Amount, payment amount

Order Amount = quantity Ordered \* priceEach

Payment Amount = Payment. Amount)

**Technology Used: HDFS, Sqoop, Hive, MapReduce, Spark**

**Steps:**

**Step1: Mysql Database Setup**

Login to MySql database and execute the script

Mysql> source /file\_path/classicModels.txt

This will create the classic model's database in MySQL

Mysql> source /home/cloudera/trg1709/classicmodels.txt

**Step2: Extract data from MySQL and load into HDFS**

           Use Sqoop to load below tables from classic models into HDFS.

Products ---------- sqoop import --connect jdbc:mysql://quickstart.cloudera/classicmodels -- username root --password cloudera --table products --target-dir /user/cloudera/trg1811/classicmodels/products --m 2

Orders ----------- sqoop import --connect jdbc:mysql://quickstart.cloudera/classicmodels -- username root --password cloudera --table orders --target-dir /user/cloudera/trg1811/classicmodels/orders --m 2

OrderDetails --------- sqoop import --connect jdbc:mysql://quickstart.cloudera/classicmodels -- username root --password cloudera --table orderdetails --target-dir /user/cloudera/trg1811/classicmodels/orderdetails --m 2

Employees ---------- sqoop import --connect jdbc:mysql://quickstart.cloudera/classicmodels -- username root --password cloudera --table Employees --target-dir /user/cloudera/trg1811/classicmodels/Employes --m 2

Customers ---------- sqoop import --connect jdbc:mysql://quickstart.cloudera/classicmodels -- username root --password cloudera --table customers --target-dir /user/cloudera/trg1811/classicmodels/customers --m 2

Payments ----------- sqoop import --connect jdbc:mysql://quickstart.cloudera/classicmodels -- username root --password cloudera --table payments --target-dir /user/cloudera/trg1811/classicmodels/payments --m 2

**Step3: Create Hive External Tables in Hive**

External Tables to be created for:

            Products

            Orders

            OrderDetails

            Employees

            Customers

            Payments

create external table customers (

  customerNumber int,

  customerName varchar(50),

  city varchar(50),

  state varchar(50),

  postalCode varchar(15),

  country varchar(50),

  salesRepEmployeeNumber int,

  creditLimit decimal(10,2)

) row format delimited

 fields terminated by ","

 stored as textfile

 location '/user/cloudera/trg1709/classicmodels/customers';

create external table employees (

  employeeNumber int,

  lastName varchar(50),

  firstName varchar(50),

  jobTitle varchar(50)

) row format delimited

 fields terminated by ","

 stored as textfile

 location '/user/cloudera/trg1709/classicmodels/employees';

drop table orders;

create external table orders (

 orderNumber int,

 orderDate date,

 requiredDate date,

 shippedDate date,

 status varchar(15),

 comments varchar(100),

 customerNumber int

) row format delimited

 fields terminated by ","

 stored as textfile

 location '/user/cloudera/trg1709/classicmodels/orders';

create external table orderdetails (

 orderNumber int,

 productCode varchar(15),

 quantityOrdered int,

 priceEach decimal(10,2),

 orderLineNumber smallint

) row format delimited

 fields terminated by ","

 stored as textfile

 location '/user/cloudera/trg1709/classicmodels/orderdetails';

create external table payments (

 customerNumber int,

 checkNumber varchar(50),

 paymentDate date,

 amount decimal(10,2)

) row format delimited

 fields terminated by ","

 stored as textfile

 location '/user/cloudera/trg1709/classicmodels/payments';

create external table products (

 productCode varchar(15),

 productName varchar(70),

 productLine varchar(50),

 productScale varchar(10),

 productVendor varchar(50),

 productDescription varchar(100),

 quantityInStock smallint,

 buyPrice decimal(10,2),

 MSRP decimal(10,2)

) row format delimited

 fields terminated by "|"

 stored as textfile

 location '/user/cloudera/trg1709/classicmodels/products';

**Step4: Create Hive Partitioned Tables**

a. Create a Hive Partitioned table in Parquet format by joining Orders, OrderDetails, and Customers tables partitioned by year and month:

**Table Name: OrderCustomerDetail**

**Columns:**

                      OrderNumber

                        OrderDate

                        ShippedDate

                        CustomerNumber

            CustomerName

                       OrderAmount

                        Status

                        OrderYear (Partition Column)

                        OrderMonth (Partition Column)

create table ordercustomerdetail(ordernumber int,

orderdate timestamp,

shippeddate timestamp,

customernumber int,

customername varchar(50),

OrderAmount Decimal(10,2),

status varchar(15)

)partitioned by (orderyear int,ordermonth int)

stored as PARQUET;

-----------------load using dynamic partition--------------------------

insert into table ordercustomerdetail

partition(orderyear ,ordermonth)

select o.ordernumber,o.orderdate,o.shippeddate,o.customernumber,c.customername,(od.quantityordered \* od.priceeach) as OrderAmount,o.status,year(o.orderdate) as orderyear,month(o.orderdate) as ordermonth from orders o join orderdetails od on (o.ordernumber=od.ordernumber) join customers c on (o.customernumber=c.customernumber);

Create a Hive Partitioned table in Parquet format by joining Orders, OrderDetails, and Products tables partitioned by year and month:

**Table Name: OrderProductDetail**

**Columns:**

                      OrderNumber

            OrderDate

                       ProductCode

ProductName

                        ProductLine

                        OrderAmount

                        OrderYear (Partition Column)

                        OrderMonth (Partition Column)

create table orderproductdetail(ordernumber int,

orderdate timestamp,

productcode int,

productname varchar(70),

productline varchar(50),

orderAmount Decimal(10,2)

)partitioned by (orderyear int,ordermonth int)

stored as PARQUET;

insert into table orderproductdetail

partition(orderyear ,ordermonth)

select o.ordernumber,o.orderdate,p.productcode,p.productname,(od.quantityordered \* od.priceeach) as OrderAmount,o.status,year(o.orderdate) as orderyear,month(o.orderdate) as ordermonth from orders o join orderdetails od on (o.ordernumber=od.ordernumber) join products p on (od.productcode=p.productcode);

**Step5: Execute Queries in Spark against partitioned tables and load the output into Hive target Tables**

d. Find the total sales generated by sales representatives by month

Target TableName: SalesSummary

The output should have the following columns:

Sales Rep Code, Sales Rep Name, Year, Month, Sales

create table salessummary

stored as PARQUET

as

select c.salesRepEmployeeNumber, e.employeenumber, ocd.orderyear, ocd.ordermonth, sum(orderamount) as Sales from ordercustomerdetail ocd join customers c on (ocd.customernumber = c.customernumber)

                                  join employees e on (c.salesRepEmployeeNumber = e.employeenumber)

group by c.salesRepEmployeeNumber, e.employeenumber, ocd.orderyear, ocd.ordermonth;

e. Find the total sales generated by each product by month

Target TableName: ProductSummary

The output should have the following columns:

Product Code, Product Name, Product Line, year, month, sales

create table productSummary

stored as PARQUET

as

select p.productcode, p.productname, p.productline, opd.orderyear, opd.ordermonth, sum(orderamount) as sales from orderproductdetail opd join products p on

(opd.productcode = p.productcode)

group by p.productname, p.productline, p.productname, p.productcode, opd.orderyear, opd.ordermonth;

f. Find the total order amount, and payment amount for each customer by joining OrderCustomerDetail and Payments tables

Target TableName: CustomerSummary

The output should have the following columns:

CustomerNumber, CustomerName, OrderAmount, payment amount

             create table customersummary

             stored as PARQUET

as

select ocd.customernumber, ocd.customername, ocd.orderamount, p.amount, sum(orderamount) as sales from ordercustomerdetail ocd join payments p

on(ocd.customernumber = p.customernumber)

group by ocd.customernumber, ocd.customername, ocd.orderamount, p.amount;