Step i: Create and Load Table with Online Retail Data in Hive

Assuming your data is in a CSV format and has been uploaded to HDFS:

sql

CopyEdit

```
CREATE TABLE online_retail_raw (

    InvoiceNo STRING,

    StockCode STRING,

    Description STRING,

    Quantity INT,

    InvoiceDate STRING,

    UnitPrice DOUBLE,

    CustomerID STRING,

    Country STRING

)

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

STORED AS TEXTFILE;


-- Load data into Hive table

LOAD DATA INPATH '/user/hive/warehouse/online_retail.csv' INTO
TABLE online_retail_raw;
```

Step j: Create Index on Online Retail Table in Hive

Hive supports indexing, though it is less commonly used due to performance overhead in newer versions. Still, for academic/demo purposes:

sql

CopyEdit

```
CREATE INDEX idx_customerid

ON TABLE online_retail_raw (CustomerID)

AS 'COMPACT'

WITH DEFERRED REBUILD;



ALTER INDEX idx_customerid ON online_retail_raw REBUILD;
```

> 📝 Note: Indexes in Hive can help in certain query optimizations but are not widely used in modern big data processing where partitioning and bucketing are more efficient.

Step k: Find the Total, Average Sales in Hive

To compute sales, we'll use the formula:
`Sales = Quantity * UnitPrice`

sql

CopyEdit

```
SELECT

    SUM(Quantity * UnitPrice) AS total_sales,

    AVG(Quantity * UnitPrice) AS average_sales

FROM online_retail_raw;
```

Step l: Find Order Details with Maximum Cost

sql

CopyEdit

```
SELECT InvoiceNo, SUM(Quantity * UnitPrice) AS order_total

FROM online_retail_raw

GROUP BY InvoiceNo

ORDER BY order_total DESC

LIMIT 1;
```

Step m: Find Customer Details with Maximum Order Total

sql

CopyEdit

```
SELECT CustomerID, SUM(Quantity * UnitPrice) AS customer_total

FROM online_retail_raw
```

```
WHERE CustomerID IS NOT NULL

GROUP BY CustomerID

ORDER BY customer_total DESC

LIMIT 1;
```

Optional: Integrate with HBase (for advanced use)

If you want to store Hive table data in HBase for fast key-based access, you can create an external Hive table backed by HBase:

sql

CopyEdit

```
CREATE TABLE hbase_online_retail(

    rowkey STRING,

    InvoiceNo STRING,

    StockCode STRING,

    Description STRING,

    Quantity INT,

    InvoiceDate STRING,

    UnitPrice DOUBLE,

    CustomerID STRING,

    Country STRING

)
```

```
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'

WITH SERDEPROPERTIES ("hbase.columns.mapping" =


":key,cf1:InvoiceNo,cf1:StockCode,cf1:Description,cf1:Quantity,cf1:InvoiceDate,cf1:UnitPrice,cf1:CustomerID,cf1:Country")

TBLPROPERTIES ("hbase.table.name" = "hbase_online_retail");
```

Then use Hive `INSERT INTO hbase_online_retail SELECT ...` to populate the HBase-backed table.

Let's go step-by-step from the basics and walk through the remaining parts of your Online Retail application using Hive and HBase.

Assumptions:

- You already have Hadoop and Hive installed.
- You have HBase integrated with Hive via HBaseStorageHandler.
- Your dataset (CSV) is uploaded to HDFS at /user/hive/warehouse/online_retail.csv

Dataset Fields (Standard Online Retail):

- InvoiceNo, StockCode, Description, Quantity, InvoiceDate, UnitPrice, CustomerID, Country

Now let's continue:

🟩 Step n. Find the Country with Maximum and Minimum Sales

First, compute sales per country (Sales = Quantity × UnitPrice):

Query to get sales per country:

sql

CopyEdit

```
SELECT Country, SUM(Quantity * UnitPrice) AS total_sales

FROM online_retail_raw

GROUP BY Country;
```

Now get the country with maximum and minimum sales:

Maximum sales:

sql

CopyEdit

```
SELECT Country, SUM(Quantity * UnitPrice) AS total_sales

FROM online_retail_raw

GROUP BY Country

ORDER BY total_sales DESC

LIMIT 1;
```

Minimum sales:

sql

CopyEdit

```
SELECT Country, SUM(Quantity * UnitPrice) AS total_sales

FROM online_retail_raw
```

```
GROUP BY Country

ORDER BY total_sales ASC

LIMIT 1;
```

🟩 Step o. Create an External Hive Table Connected to HBase

We will create an HBase table and a corresponding external Hive table.

First, create the HBase table:

From HBase shell:

bash

CopyEdit

```
hbase shell
```

Inside HBase shell:

bash

CopyEdit

```
create 'online_retail_hbase', 'cf1'
```

Now, create an external Hive table pointing to this HBase table:

sql

CopyEdit

```
CREATE EXTERNAL TABLE online_retail_hbase (
    rowkey STRING,

    InvoiceNo STRING,

    StockCode STRING,

    Description STRING,

    Quantity INT,

    InvoiceDate STRING,

    UnitPrice DOUBLE,

    CustomerID STRING,

    Country STRING

)

STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'

WITH SERDEPROPERTIES (

  "hbase.columns.mapping" =
":key,cf1:InvoiceNo,cf1:StockCode,cf1:Description,cf1:Quantity,c
f1:InvoiceDate,cf1:UnitPrice,cf1:CustomerID,cf1:Country"

)

TBLPROPERTIES("hbase.table.name" = "online_retail_hbase");
```

Here:

- rowkey = unique identifier (e.g., InvoiceNo+CustomerID+StockCode).
- cf1 = column family used in HBase.

🟩 Step p. Display Records of OnlineRetail Table in HBase

After loading data into the HBase-backed Hive table, you can simply run:

sql

CopyEdit

```
SELECT * FROM online_retail_hbase LIMIT 10;
```

If you haven't loaded data yet, you can insert from your raw table:

sql

CopyEdit

```
INSERT INTO TABLE online_retail_hbase

SELECT

    concat(InvoiceNo, '_', CustomerID, '_', StockCode) AS
rowkey,

    InvoiceNo,

    StockCode,

    Description,

    Quantity,

    InvoiceDate,

    UnitPrice,

    CustomerID,

    Country
```

```
FROM online_retail_raw;
```

Then rerun the SELECT:

sql

CopyEdit

```
SELECT * FROM online_retail_hbase LIMIT 10;
```

✅ Summary of What You've Learned:

- How to query sales stats by country in Hive
- How to connect Hive with HBase using external table
- How to populate and query records from HBase through Hive