Write an application using HBase and HiveQL for flight information system which will include a. Create Flight Info Hbase Table(with Flight information, schedule, and delay) b. Demonstrate Creating, Dropping, and altering Database tables in Hbase c. Creating an external Hive table to connect to the HBase for Flight Information Table d. Find the total departure delay in Hive e. Find the average departure delay in Hive f. Create index on Flight information Table I am using cloudera vm tfor this expalin detailed steps

ChatGPT said:

Here's a detailed step-by-step guide to building a **Flight Information System using HBase and HiveQL** on your **Cloudera VM**. The system will include table creation, connection between HBase and Hive, queries, and indexing.

⚙️ **Prerequisites:**

Ensure these services are running in Cloudera VM:

- HBase
- Hive
- HDFS
- Hadoop MapReduce or YARN
- HCatalog (for Hive-HBase integration)

Step A: Create Flight Info HBase Table

Open HBase Shell in Cloudera VM:

bash

CopyEdit

```
hbase shell
```

Create HBase Table flight_info:

hbase

CopyEdit

```
create 'flight_info', 'info'
```

This creates an HBase table named flight_info with a column family info.

Add Sample Data:

hbase

CopyEdit

```
put 'flight_info', 'FL001', 'info:flight_number', 'AI202'
put 'flight_info', 'FL001', 'info:origin', 'DEL'
put 'flight_info', 'FL001', 'info:destination', 'BOM'
put 'flight_info', 'FL001', 'info:schedule', '2025-05-07 10:00'
put 'flight_info', 'FL001', 'info:delay', '15'

put 'flight_info', 'FL002', 'info:flight_number', 'AI203'
put 'flight_info', 'FL002', 'info:origin', 'BLR'
put 'flight_info', 'FL002', 'info:destination', 'DEL'
put 'flight_info', 'FL002', 'info:schedule', '2025-05-07 12:00'
put 'flight_info', 'FL002', 'info:delay', '0'
```

Step B: HBase Table Operations

Drop Table:

hbase

CopyEdit

```
disable 'flight_info'
drop 'flight_info'
```

Alter Table (Add Column Family):

hbase

CopyEdit

```
alter 'flight_info', NAME => 'status'
```

Step C: Create External Hive Table for HBase Table

Launch Hive Shell:

bash

CopyEdit

```
hive
```

Create Hive External Table Mapped to HBase:

sql

CopyEdit

```
CREATE EXTERNAL TABLE flight_info_hive(
    rowkey STRING,
    flight_number STRING,
    origin STRING,
    destination STRING,
    schedule STRING,
    delay INT
)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
WITH SERDEPROPERTIES (
  "hbase.columns.mapping" = ":key,info:flight_number,info:origin,info:destination,info:schedule,info:delay"
)
TBLPROPERTIES (
  "hbase.table.name" = "flight_info"
);
```

✅ Now your Hive table flight_info_hive is connected to HBase table flight_info.

Step D: Find Total Departure Delay in Hive

sql

CopyEdit

```
SELECT SUM(delay) AS total_delay FROM flight_info_hive;
```

Step E: Find Average Departure Delay in Hive

sql

CopyEdit

```
SELECT AVG(delay) AS average_delay FROM flight_info_hive;
```

Step F: Create Index on Flight Number (Hive Table)

Note: Hive indexes are largely deprecated in modern usage, but if you're on older Hive version:

sql

CopyEdit

```
CREATE INDEX idx_flight_number

ON TABLE flight_info_hive (flight_number)

AS 'COMPACT'

WITH DEFERRED REBUILD;


ALTER INDEX idx_flight_number ON flight_info_hive REBUILD;
```

📌 **Note:** Hive indexing is often not supported or useful with external tables, especially ones backed by HBase. For performance, prefer **partitioning**, **bucketing**, or **materialized views** instead in production.

✅ **Summary:**

Step	**Task**
A	Created flight_info HBase table
B	Performed table operations: create, alter, drop
C	Linked HBase with Hive using external table
D	Queried total delay
E	Queried average delay

Step	Task
F	Created index (optional / legacy feature)

Would you like a script file combining all commands to run on your Cloudera VM?

2/2

4o

You said:

Write an application using HBase and HiveQL for Customer information system which will include a. Creation of –Cutomer_info(Cust-ID,Cust-Name,orderID), order_info(OrderID,ItemID,Quantity), item_info(Item-ID,Item-Name,ItemPrice) tables in Hive b. Load table with data from local storage in Hive. c. Perform Join tables with Hive d. Create Index on Customer information system in Hive. e. Find the total, average sales in Hive f. Find Order details with maximum cost. g. Creating an external Hive table to connect to the HBase for Customer Information System. h. Display records of Customer Information Table in Hbase.

ChatGPT said:

Here is a **complete Hive and HBase-based application** for a **Customer Information System** on **Cloudera VM**, covering table creation, data loading, queries, joins, indexing, and integration with HBase.

🔧 Prerequisites

Make sure the following services are **running** on your Cloudera VM:

- Hive
- HBase
- HDFS
- YARN
- HCatalog (for Hive-HBase integration)

🅰️ a. Create Tables in Hive

Launch Hive:

bash

CopyEdit

```
hive
```

Create customer_info Table:

sql

CopyEdit

```
CREATE TABLE customer_info (
  cust_id INT,
  cust_name STRING,
  order_id INT
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE;
```

Create order_info Table:

sql

CopyEdit

```
CREATE TABLE order_info (
  order_id INT,
  item_id INT,
  quantity INT
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE;
```

Create item_info Table:

sql

CopyEdit

```
CREATE TABLE item_info (
  item_id INT,
  item_name STRING,
  item_price FLOAT
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE;
```

🅱️ b. Load Table with Data from Local Storage

Example data files:

Place these files under /home/cloudera/customer_data/ in Cloudera VM:

- customer_info.txt

CopyEdit

```
1,John,1001
2,Alice,1002
3,Bob,1003
```

- order_info.txt

yaml

CopyEdit

```
1001,2001,2
1002,2002,1
1003,2001,3
```

- item_info.txt

yaml

CopyEdit

```
2001,Laptop,50000
2002,Phone,20000
```

Load Data into Tables:

sql

CopyEdit

```
LOAD DATA LOCAL INPATH '/home/cloudera/customer_data/customer_info.txt' INTO TABLE customer_info;

LOAD DATA LOCAL INPATH '/home/cloudera/customer_data/order_info.txt' INTO TABLE order_info;

LOAD DATA LOCAL INPATH '/home/cloudera/customer_data/item_info.txt' INTO TABLE item_info;
```

🆑 c. Perform Join Tables in Hive

Join to Get Full Order Details with Cost:

sql

CopyEdit

```
SELECT
  c.cust_id,
  c.cust_name,
  i.item_name,
  o.quantity,
  i.item_price,
  (o.quantity * i.item_price) AS total_cost
FROM
  customer_info c
JOIN
  order_info o ON c.order_id = o.order_id
JOIN
  item_info i ON o.item_id = i.item_id;
```

🅳 d. Create Index on Customer Table

Hive indexing is rarely used now, but here's the legacy syntax:

sql

CopyEdit

```
CREATE INDEX idx_customer_name
ON TABLE customer_info (cust_name)
AS 'COMPACT'
WITH DEFERRED REBUILD;

ALTER INDEX idx_customer_name ON customer_info REBUILD;
```

🇪 e. Find Total and Average Sales

sql

CopyEdit

```
SELECT
  SUM(o.quantity * i.item_price) AS total_sales,
  AVG(o.quantity * i.item_price) AS average_sales
FROM order_info o
JOIN item_info i ON o.item_id = i.item_id;
```

🇫 f. Find Order Details with Maximum Cost

sql

CopyEdit

```
SELECT
  o.order_id,
  i.item_name,
  o.quantity,
  i.item_price,
  (o.quantity * i.item_price) AS total_cost
FROM order_info o
```

```
JOIN item_info i ON o.item_id = i.item_id
ORDER BY total_cost DESC
LIMIT 1;
```

🅖 g. Create External Hive Table Connected to HBase

Step 1: Create HBase Table

bash

CopyEdit

```
hbase shell
```

hbase

CopyEdit

```
create 'customer_info_hbase', 'info'
put '1', 'info:cust_name', 'John'
put '1', 'info:order_id', '1001'

put '2', 'info:cust_name', 'Alice'
put '2', 'info:order_id', '1002'
```

Step 2: Create Hive Table Linked to HBase Table

sql

CopyEdit

```
CREATE EXTERNAL TABLE customer_info_ext (
  key STRING,
  cust_name STRING,
  order_id STRING
)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
WITH SERDEPROPERTIES (
  "hbase.columns.mapping" = ":key,info:cust_name,info:order_id"
)
TBLPROPERTIES (
  "hbase.table.name" = "customer_info_hbase"
);
```

---

h. Display Records of Customer Information from HBase

sql

CopyEdit

```
SELECT * FROM customer_info_ext;
```

✅ Summary

Task	**Operation**
a	Created Hive tables for customer, order, and item info
b	Loaded data from local files
c	Performed joins to fetch complete order info
d	Created index on customer table
e	Calculated total and average sales
f	Found order with maximum cost
g	Created external Hive table connected to HBase
h	Displayed HBase table data in Hive

Would you like me to generate a script file or all commands in .hql and .sh format for easier execution?

4o

You said:

Write an application using HBase and HiveQL for OnlineRetail Dataset which will include i. Create and Load table with Online Retail data in Hive. j. Create Index on Online Retail Table in Hive. k. Find the total, average sales in Hive l. Find Order details with maximum cost. m. Find Customer details with maximum order total. n. Find the Country with maximum and minimum sale. o. Creating an external Hive table to connect to the HBase for OnlineRetail. p. Display records of OnlineRetail Table in Hbase.

ChatGPT said:

Here is a **step-by-step Hive + HBase application** using the **OnlineRetail dataset**. This includes Hive table creation, loading data, running sales analytics, connecting Hive with HBase, and querying HBase through Hive.

🗂️ Dataset: *Online Retail (UCI Repository)*

Expected columns (typical format):

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

✅ i. Create and Load Hive Table for Online Retail Data

Step 1: Launch Hive shell

bash

CopyEdit

```
hive
```

Step 2: Create Hive Table

sql

CopyEdit

```
CREATE TABLE online_retail (
  invoice_no STRING,
  stock_code STRING,
  description STRING,
  quantity INT,
  invoice_date STRING,
  unit_price FLOAT,
  customer_id STRING,
  country STRING
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE;
```

Step 3: Load Data from Local File

Place file in VM:

bash

CopyEdit

```
# Move file to local folder
mv OnlineRetail.csv /home/cloudera/data/
```

Load it into Hive:

sql

CopyEdit

```
LOAD DATA LOCAL INPATH '/home/cloudera/data/OnlineRetail.csv' INTO TABLE online_retail;
```

✅ j. Create Index on Online Retail Table

sql

CopyEdit

```
CREATE INDEX idx_customer_id
ON TABLE online_retail (customer_id)
AS 'COMPACT'
WITH DEFERRED REBUILD;
```

```
ALTER INDEX idx_customer_id ON online_retail REBUILD;
```

ⓘ Hive indexing is deprecated in modern versions; better alternatives: **partitioning** or **materialized views**.

✅ k. Find Total, Average Sales in Hive

Sales = quantity * unit_price

sql

CopyEdit

```
SELECT
  SUM(quantity * unit_price) AS total_sales,
  AVG(quantity * unit_price) AS average_sales
FROM online_retail;
```

✅ l. Find Order Details with Maximum Cost

Assuming invoice_no is the order ID:

sql

CopyEdit

```
SELECT
  invoice_no,
  SUM(quantity * unit_price) AS order_total
FROM online_retail
GROUP BY invoice_no
ORDER BY order_total DESC
LIMIT 1;
```

✅ m. Find Customer with Maximum Order Total

sql

CopyEdit

```sql
SELECT
  customer_id,
  SUM(quantity * unit_price) AS total_spent
FROM online_retail
GROUP BY customer_id
ORDER BY total_spent DESC
LIMIT 1;
```

✅ n. Find Country with Maximum and Minimum Sale

sql

CopyEdit

```sql
-- Maximum Sale
SELECT
  country,
  SUM(quantity * unit_price) AS total_sales
FROM online_retail
GROUP BY country
ORDER BY total_sales DESC
LIMIT 1;

-- Minimum Sale
SELECT
  country,
  SUM(quantity * unit_price) AS total_sales
FROM online_retail
GROUP BY country
ORDER BY total_sales ASC
```

```
LIMIT 1;
```

✅ o. Create External Hive Table Connected to HBase

Step 1: Create HBase Table

bash

CopyEdit

```
hbase shell
```

hbase

CopyEdit

```
create 'online_retail_hbase', 'info'
put '1', 'info:invoice_no', '536365'
put '1', 'info:description', 'WHITE HANGING HEART T-LIGHT HOLDER'
put '1', 'info:quantity', '6'
put '1', 'info:unit_price', '2.55'
put '1', 'info:customer_id', '17850'
put '1', 'info:country', 'United Kingdom'
```

Step 2: Create External Hive Table for HBase Table

```
CREATE EXTERNAL TABLE online_retail_hbase (
  rowkey STRING,
  invoice_no STRING,
  description STRING,
  quantity INT,
  unit_price FLOAT,
  customer_id STRING,
  country STRING
)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
```

```
WITH SERDEPROPERTIES (
  "hbase.columns.mapping" = ":key,info:invoice_no,info:description,info:quantity,info:unit_price,info:customer_id,info:country"
)
TBLPROPERTIES (
  "hbase.table.name" = "online_retail_hbase"
);
```

✅ p. Display Records from HBase in Hive

sql

CopyEdit

```
SELECT * FROM online_retail_hbase;
```

📘 Summary

Task	Action
i	Created and loaded Hive table for OnlineRetail
j	Created index on customer_id
k	Calculated total and average sales
l	Found order with maximum cost
m	Found customer with highest spending
n	Identified countries with max/min sales
o	Created Hive-HBase external table
p	Displayed HBase records via Hive

Would you like me to generate a .hql script or bash automation for this entire setup?

4o