

Creating Lookup Table

Command to create the Lookup Table

Dataframe Name: look_up_table

Hbase Table: look_up_table

```
import happybase

# Establish connection to HBase
connection = happybase.Connection('localhost', port=9090, autoconnect=False)

# Function to open the connection
def open_connection():
    connection.open()

# Function to close the connection
def close_connection():
    connection.close()

# Function to list all tables in HBase
def list_tables():
    print("Fetching all tables")
    open_connection()
    tables = connection.tables()
    close_connection()
    print("All tables fetched")
    return tables

# Function to create a new HBase table
def create_table(name, cf):
    print("Creating table " + name)
    tables = list_tables()
    if name not in tables:
        open_connection()
        connection.create_table(name, cf)
        close_connection()
        print("Table created")
    else:
        print("Table already present")

# Function to get a table object
def get_table(name):
    open_connection()
    table = connection.table(name)
    close_connection()
    return table

# Create the lookup table with the specified column family
create_table('look_up_table', {'info': dict(max_versions=5)})
```

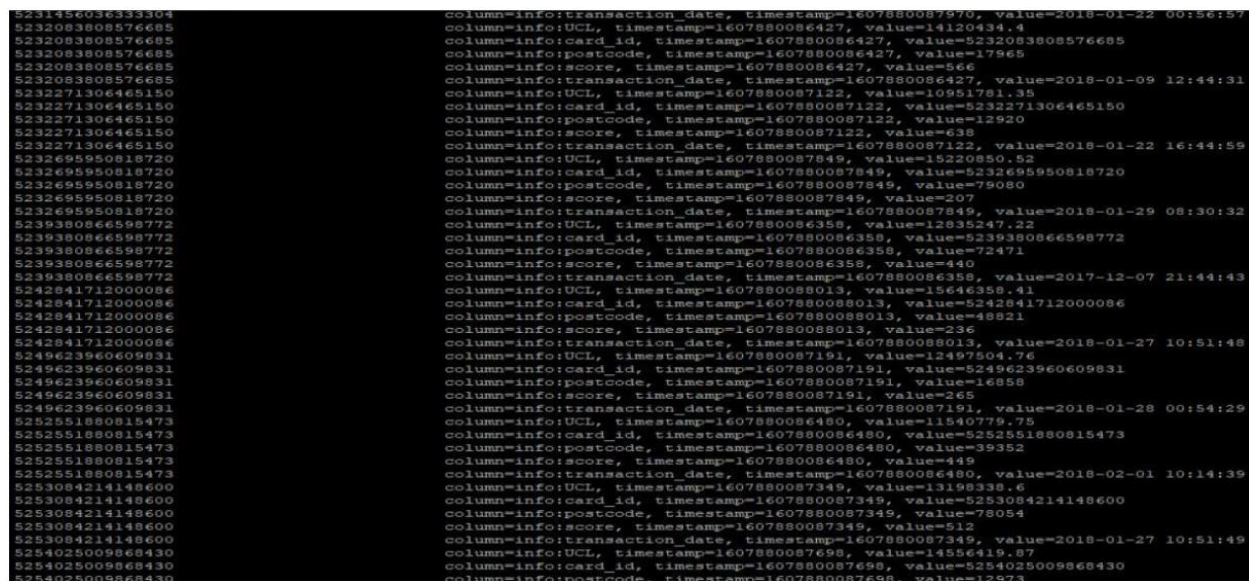
```
# Function to batch insert data into the HBase table
def batch_insert_data(df, tableName):
    print("Starting batch insert of events")
    table = get_table(tableName)
    open_connection()
    with table.batch(batch_size=4) as b:
        for row in df.rdd.collect():
            b.put(
                bytes(row.card_id),
                {
                    'info:card_id': bytes(row.card_id),
                    'info:transaction_date': bytes(row.transaction_date),
                    'info:score': bytes(row.score),
                    'info:postcode': bytes(row.postcode),
                    'info:UCL': bytes(row.UCL)
                }
            )
    print("Batch insert done")
    close_connection()

# Insert data into HBase table from the DataFrame
batch_insert_data(look_up_table, 'look_up_table')
```

<Command to see the table created>

- `list` command is used to view the created table
- `scan 'look_up_table'` command to view data inserted into table.

Screenshot of the created table



The screenshot displays a list of data rows from an HBase table. Each row is represented by a key-value pair. The key is a long hexadecimal string (e.g., 5231456036339304), and the value is a JSON-like structure containing transaction details. The values are truncated in the image. The structure of the values is as follows:

```
{
  "column": "info:transaction_date", "timestamp": "1607880087970", "value": "2018-01-22 00:15:17",
  "column": "info:UCL", "timestamp": "1607880086427", "value": "14120434.4",
  "column": "info:card_id", "timestamp": "1607880086427", "value": "5232053808576685",
  "column": "info:postcode", "timestamp": "1607880086427", "value": "17965",
  "column": "info:score", "timestamp": "1607880086427", "value": "566",
  "column": "info:transaction_date", "timestamp": "1607880086427", "value": "2018-01-09 12:44:31",
  "column": "info:UCL", "timestamp": "1607880087122", "value": "10951781.35",
  "column": "info:card_id", "timestamp": "1607880087122", "value": "5232271306465150",
  "column": "info:postcode", "timestamp": "1607880087122", "value": "12920",
  "column": "info:score", "timestamp": "1607880087122", "value": "638",
  "column": "info:transaction_date", "timestamp": "1607880087122", "value": "2018-01-22 16:44:59",
  "column": "info:UCL", "timestamp": "1607880087849", "value": "15220950.52",
  "column": "info:card_id", "timestamp": "1607880087849", "value": "5232695950818720",
  "column": "info:postcode", "timestamp": "1607880087849", "value": "79080",
  "column": "info:score", "timestamp": "1607880087849", "value": "207",
  "column": "info:transaction_date", "timestamp": "1607880087849", "value": "2018-01-29 08:30:32",
  "column": "info:UCL", "timestamp": "1607880086358", "value": "12835247.22",
  "column": "info:card_id", "timestamp": "1607880086358", "value": "5239380866598772",
  "column": "info:postcode", "timestamp": "1607880086358", "value": "72471",
  "column": "info:score", "timestamp": "1607880086358", "value": "440",
  "column": "info:transaction_date", "timestamp": "1607880086358", "value": "2017-12-07 21:44:43",
  "column": "info:UCL", "timestamp": "1607880088013", "value": "15646358.41",
  "column": "info:card_id", "timestamp": "1607880088013", "value": "5242841712000086",
  "column": "info:postcode", "timestamp": "1607880088013", "value": "49821",
  "column": "info:score", "timestamp": "1607880088013", "value": "236",
  "column": "info:transaction_date", "timestamp": "1607880088013", "value": "2018-01-27 10:51:48",
  "column": "info:UCL", "timestamp": "1607880087191", "value": "12497504.76",
  "column": "info:card_id", "timestamp": "1607880087191", "value": "5249623960609831",
  "column": "info:postcode", "timestamp": "1607880087191", "value": "16858",
  "column": "info:score", "timestamp": "1607880087191", "value": "265",
  "column": "info:transaction_date", "timestamp": "1607880087191", "value": "2018-01-28 00:54:29",
  "column": "info:UCL", "timestamp": "1607880086480", "value": "11540779.75",
  "column": "info:card_id", "timestamp": "1607880086480", "value": "5252551880815473",
  "column": "info:score", "timestamp": "1607880086480", "value": "39352",
  "column": "info:transaction_date", "timestamp": "1607880086480", "value": "2018-02-01 10:14:39",
  "column": "info:UCL", "timestamp": "1607880087349", "value": "13198338.6",
  "column": "info:card_id", "timestamp": "1607880087349", "value": "5253084214148600",
  "column": "info:postcode", "timestamp": "1607880087349", "value": "78054",
  "column": "info:score", "timestamp": "1607880087349", "value": "512",
  "column": "info:transaction_date", "timestamp": "1607880087349", "value": "2018-01-27 10:51:49",
  "column": "info:UCL", "timestamp": "1607880087698", "value": "14556419.87",
  "column": "info:card_id", "timestamp": "1607880087698", "value": "5254025009868430",
  "column": "info:postcode", "timestamp": "1607880087698", "value": "12873"
}
```