EXPERIMENT NUMBER 2 : Use of Sqoop Tool to transfer data between Hadoop and relational database servers

Date of Performance:

Date of Submission:

1. Statement of Problem:

Use of Sqoop tool to transfer data between Hadoop and relational database servers.

a. Sqoop - Installation.

b. To execute basic commands of Hadoop eco system component Sqoop.

Aim: To install and configure **Apache Sqoop** and demonstrate how to **import and export data** between **Hadoop Distributed File System (HDFS)** and a **MySQL relational database**.

Objectives:

- To understand the architecture and purpose of Sqoop.
- To install and configure Sqoop on a Hadoop-enabled environment.
- To connect Sqoop with Hadoop and MySQL.
- To perform basic import/export operations using Sqoop. • To observe how data flows between RDBMS and Hadoop systems.

Software Used:

- **Operating System:** Ubuntu/Linux or Windows + WSL
- **Hadoop:** Version 3.x or compatible
- **Sqoop:** Version 1.4.7 or latest
- **MySQL:** Version 5.7 or latest compatible
- **Java:** JDK 8 or above
- **Apache Hive (Optional):** For advanced integration

2. Theory:

2.1 What is Sqoop?

Apache Sqoop is a command-line interface application designed for efficiently **transferring bulk data** between **Apache Hadoop** and **structured data stores** such as **relational databases** (e.g., MySQL, Oracle, SQL Server).

Key Features:

- High-performance data import/export.
- Supports **parallel processing**.
- Easily integrates with **HDFS**, **Hive**, and **HBase**.
- Works well with **batch processing** pipelines. **2.2**

Sqoop Architecture Overview

Sqoop works by:

1. **Connecting** to the relational database using JDBC.
2. **Generating MapReduce jobs** to import/export data.
3. **Storing** the imported data in HDFS (or Hive/HBase).
4. **Exporting** processed data back to RDBMS if required.

3. Implementation

3.1 Environment Setup

1. **Install Hadoop** and configure it (set `JAVA_HOME`, `HADOOP_HOME`).
2. **Install MySQL** and create a database for testing.
3. **Download and Extract Sqoop** from Apache Sqoop Website.

```
tar -xvf sqoop-1.4.7.bin
hadoop-2.6.0.tar.gz mv sqoop-
1.4.7.bin ___________ hadoop-
2.6.0

/usr/local/sqoop
```

4. **Set Environment Variables** (`~/.bashrc` or `.profile`): `export`

```
SQOOP_HOME=/usr/local/sqoop
export
PATH=$PATH:$SQOOP_HOME/bin
export
```

```
HADOOP_COMMON_HOME=/usr/loca
l/hadoop export
HADOOP_MAPRED_HOME=/usr/loca
l/hadoop
```

5. Add MySQL JDBC Connector:
   - Download `mysql-connector-java-x.x.xx.jar`.
   - Place it inside:`/usr/local/sqoop/lib/`

How to Connect Sqoop with Hadoop and MySQL

- **Hadoop** must be properly installed and running (`start-dfs.sh`, `start-yarn.sh`).
- **MySQL** should be running with a database and user created.
- Verify connectivity using:

```
sqoop list-databases \
--connect jdbc:mysql://localhost:3306/ \
--username root \
--password yourpassword
```

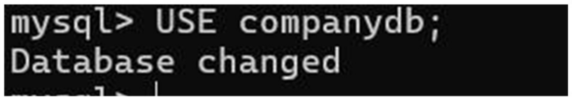
1.1 Problem Explanation

The aim of the experiment was to demonstrate how to transfer data between a relational database (MySQL) and Hadoop Distributed File System (HDFS) using Apache Sqoop. This process simulates a real-world ETL (Extract, Transform, Load) scenario where structured data is ingested into the Hadoop ecosystem for analysis and later exported back for reporting or operational use.

1.2 Execution and Output

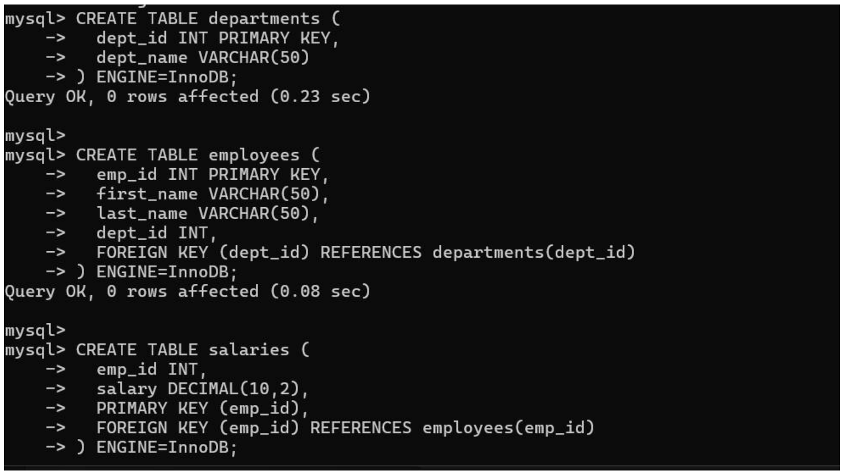
Creating Database on MySQL

Create database:



```
mysql> USE companydb;
Database changed
```

Create Tables:

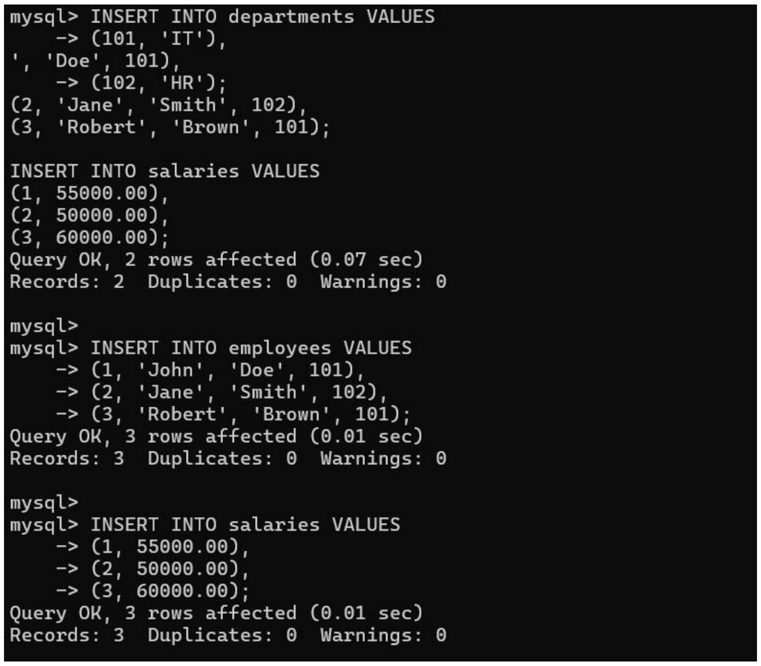


```
mysql> CREATE TABLE departments (
    ->   dept_id INT PRIMARY KEY,
    ->   dept_name VARCHAR(50)
    -> ) ENGINE=InnoDB;
Query OK, 0 rows affected (0.23 sec)

mysql>
mysql> CREATE TABLE employees (
    ->   emp_id INT PRIMARY KEY,
    ->   first_name VARCHAR(50),
    ->   last_name VARCHAR(50),
    ->   dept_id INT,
    ->   FOREIGN KEY (dept_id) REFERENCES departments(dept_id)
    -> ) ENGINE=InnoDB;
Query OK, 0 rows affected (0.08 sec)

mysql>
mysql> CREATE TABLE salaries (
    ->   emp_id INT,
    ->   salary DECIMAL(10,2),
    ->   PRIMARY KEY (emp_id),
    ->   FOREIGN KEY (emp_id) REFERENCES employees(emp_id)
    -> ) ENGINE=InnoDB;
```

Insert Value:



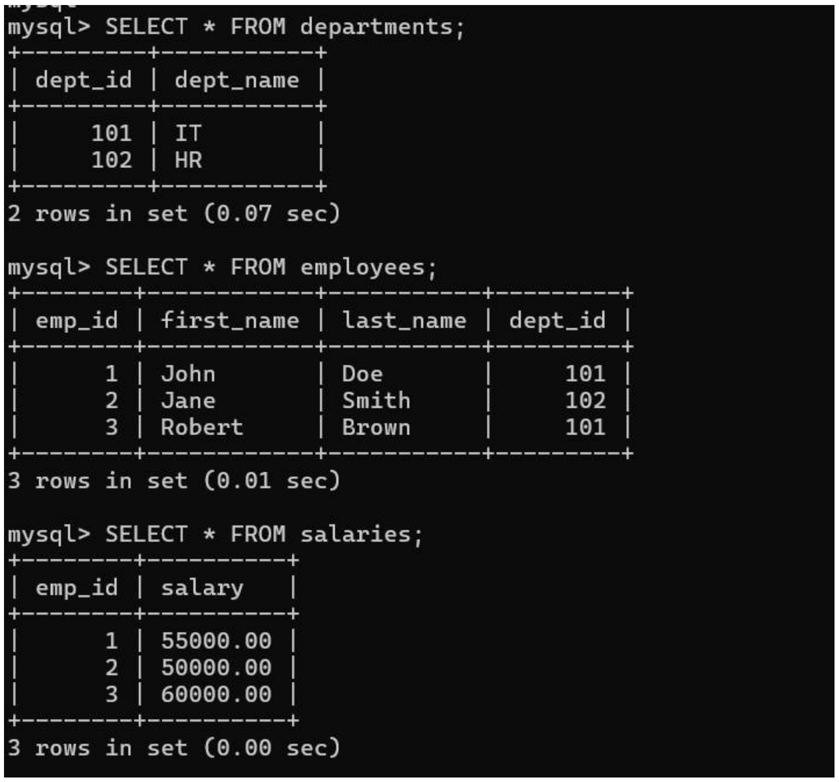
```
mysql> INSERT INTO departments VALUES
    -> (101, 'IT'),
', 'Doe', 101),
    -> (102, 'HR');
(2, 'Jane', 'Smith', 102),
(3, 'Robert', 'Brown', 101);

INSERT INTO salaries VALUES
(1, 55000.00),
(2, 50000.00),
(3, 60000.00);
Query OK, 2 rows affected (0.07 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql>
mysql> INSERT INTO employees VALUES
    -> (1, 'John', 'Doe', 101),
    -> (2, 'Jane', 'Smith', 102),
    -> (3, 'Robert', 'Brown', 101);
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql>
mysql> INSERT INTO salaries VALUES
    -> (1, 55000.00),
    -> (2, 50000.00),
    -> (3, 60000.00);
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0
```

View Database:



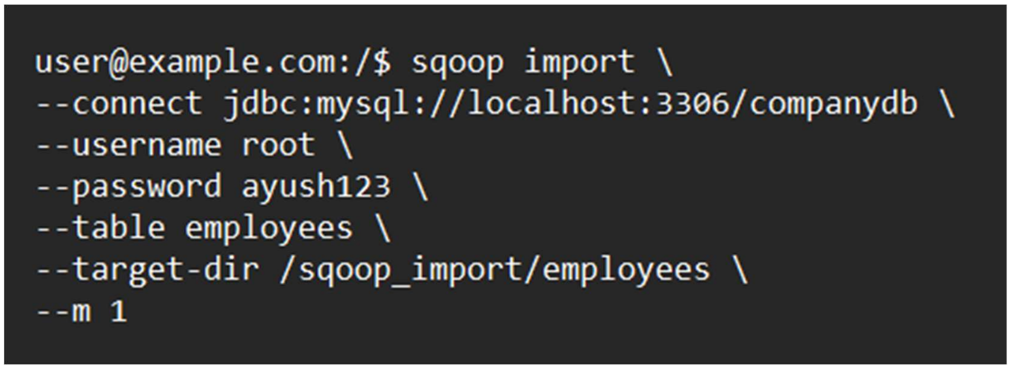
```
mysql> SELECT * FROM departments;
+---------+-----------+
| dept_id | dept_name |
+---------+-----------+
|     101 | IT        |
|     102 | HR        |
+---------+-----------+
2 rows in set (0.07 sec)

mysql> SELECT * FROM employees;
+--------+------------+-----------+---------+
| emp_id | first_name | last_name | dept_id |
+--------+------------+-----------+---------+
|      1 | John       | Doe       |     101 |
|      2 | Jane       | Smith     |     102 |
|      3 | Robert     | Brown     |     101 |
+--------+------------+-----------+---------+
3 rows in set (0.01 sec)

mysql> SELECT * FROM salaries;
+--------+----------+
| emp_id | salary   |
+--------+----------+
|      1 | 55000.00 |
|      2 | 50000.00 |
|      3 | 60000.00 |
+--------+----------+
3 rows in set (0.00 sec)
```

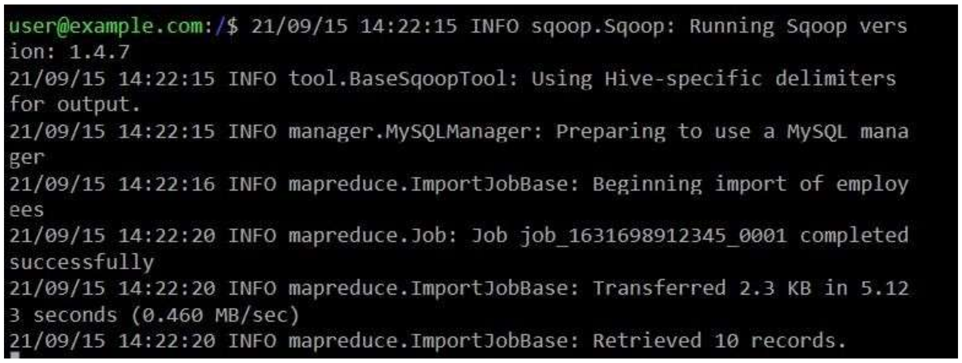
Data Transfer Using Sqoop

1. Import Data from MySQL to HDFS



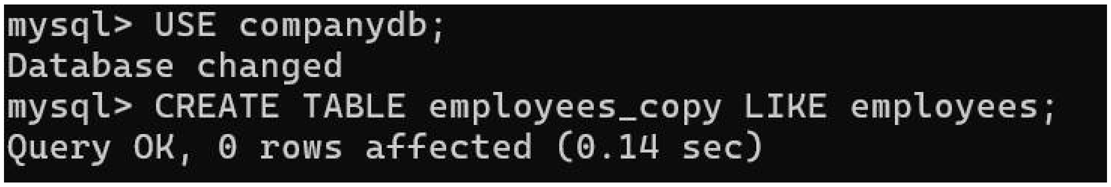
```
user@example.com:/$ sqoop import \
--connect jdbc:mysql://localhost:3306/companydb \
--username root \
--password ayush123 \
--table employees \
--target-dir /sqoop_import/employees \
--m 1
```

Output:



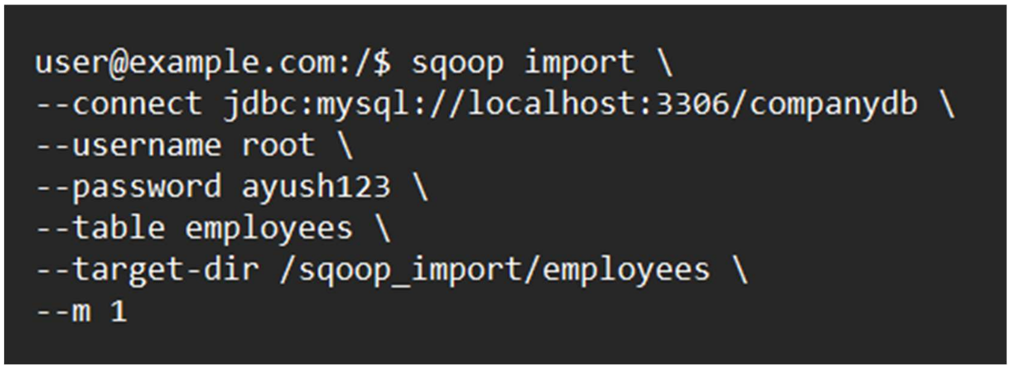
```
user@example.com:/$ 21/09/15 14:22:15 INFO sqoop.Sqoop: Running Sqoop vers
ion: 1.4.7
21/09/15 14:22:15 INFO tool.BaseSqoopTool: Using Hive-specific delimiters
for output.
21/09/15 14:22:15 INFO manager.MySQLManager: Preparing to use a MySQL mana
ger
21/09/15 14:22:16 INFO mapreduce.ImportJobBase: Beginning import of employ
ees
21/09/15 14:22:20 INFO mapreduce.Job: Job job_1631698912345_0001 completed
successfully
21/09/15 14:22:20 INFO mapreduce.ImportJobBase: Transferred 2.3 KB in 5.12
3 seconds (0.460 MB/sec)
21/09/15 14:22:20 INFO mapreduce.ImportJobBase: Retrieved 10 records.
```

2. Export Data from HDFS to MySQL



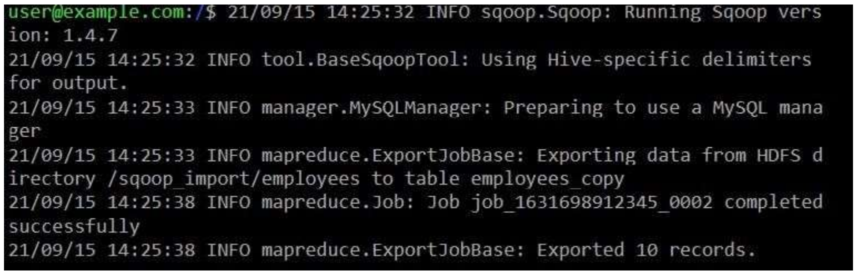
```
mysql> USE companydb;
Database changed
mysql> CREATE TABLE employees_copy LIKE employees;
Query OK, 0 rows affected (0.14 sec)
```

Then run:



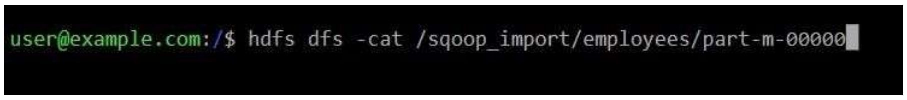
```
user@example.com:/$ sqoop import \
--connect jdbc:mysql://localhost:3306/companydb \
--username root \
--password ayush123 \
--table employees \
--target-dir /sqoop_import/employees \
--m 1
```

Output:



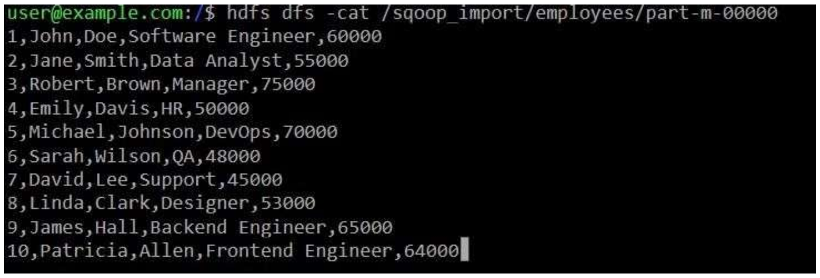
```
user@example.com:/$ 21/09/15 14:25:32 INFO sqoop.Sqoop: Running Sqoop vers
ion: 1.4.7
21/09/15 14:25:32 INFO tool.BaseSqoopTool: Using Hive-specific delimiters
for output.
21/09/15 14:25:33 INFO manager.MySQLManager: Preparing to use a MySQL mana
ger
21/09/15 14:25:33 INFO mapreduce.ExportJobBase: Exporting data from HDFS d
irectory /sqoop_import/employees to table employees_copy
21/09/15 14:25:38 INFO mapreduce.Job: Job job_1631698912345_0002 completed
successfully
21/09/15 14:25:38 INFO mapreduce.ExportJobBase: Exported 10 records.
```

3. View Imported Data in HDFS



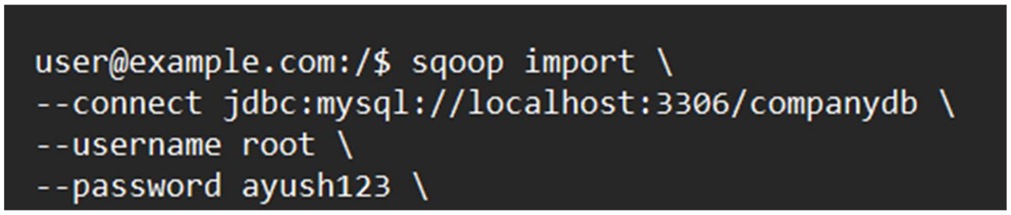
```
user@example.com:/$ hdfs dfs -cat /sqoop_import/employees/part-m-00000
```

Output:



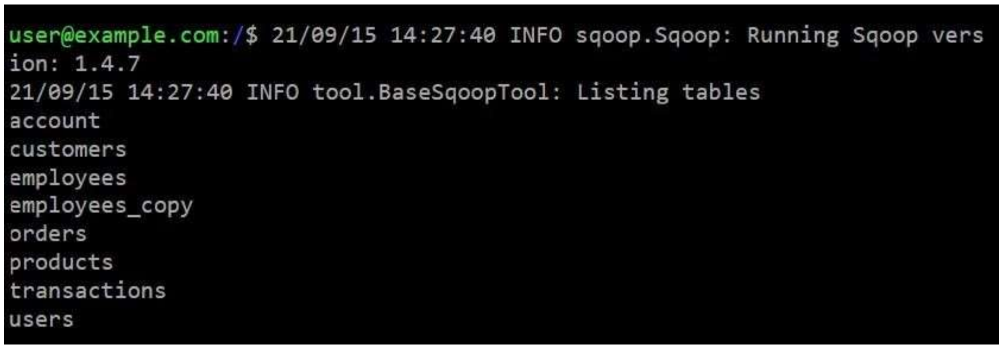
```
user@example.com:/$ hdfs dfs -cat /sqoop_import/employees/part-m-00000
1,John,Doe,Software Engineer,60000
2,Jane,Smith,Data Analyst,55000
3,Robert,Brown,Manager,75000
4,Emily,Davis,HR,50000
5,Michael,Johnson,DevOps,70000
6,Sarah,Wilson,QA,48000
7,David,Lee,Support,45000
8,Linda,Clark,Designer,53000
9,James,Hall,Backend Engineer,65000
10,Patricia,Allen,Frontend Engineer,64000
```

4. List Tables from MySQL



```
user@example.com:/$ sqoop import \
--connect jdbc:mysql://localhost:3306/companydb \
--username root \
--password ayush123 \
```

Output:



```
user@example.com:/$ 21/09/15 14:27:40 INFO sqoop.Sqoop: Running Sqoop vers
ion: 1.4.7
21/09/15 14:27:40 INFO tool.BaseSqoopTool: Listing tables
account
customers
employees
employees_copy
orders
products
transactions
users
```

4. Conclusion

In this experiment, Apache Sqoop was successfully installed and configured in a Hadoop- enabled environment, with proper connectivity established between Sqoop, MySQL, and HDFS. Environment variables were configured, the MySQL JDBC connector was added, and both Hadoop and MySQL services were verified to be running.

We performed import and export operations to transfer data between MySQL and HDFS, confirming Sqoop's bidirectional data transfer capability. Data verification in both environments ensured accuracy and consistency.

This exercise provided hands-on experience with Sqoop's role in ETL workflows and highlighted its importance in efficient large-scale data migration. It enhanced both practical skills and conceptual understanding of data movement in the Big Data ecosystem, laying a foundation for advanced integrations with Hive, HBase, or enterprise pipelines.

References:

1. Apache Sqoop Documentation. (2024). User Guide. Retrieved from https://sqoop.apache.org/
2. White, T. (2015). Hadoop: The Definitive Guide. O'Reilly Media.
3. MySQL Documentation. (2024). MySQL 5.7 Reference Manual. Oracle Corporation.

Sign and Remark:

R1 (4 Marks)	**R2 (4 Marks)**	**R3 (4 Marks)**	**R4 (3 Marks)**	**Total Marks (15 Marks)**	**Signature**