# **HDFS**

HDFS (Hadoop Distributed File System) is designed to store large datasets across multiple machines in a distributed fashion. It provides high throughput and fault tolerance, which makes it suitable for big data applications.

#### **2. HDFS Architecture Components**

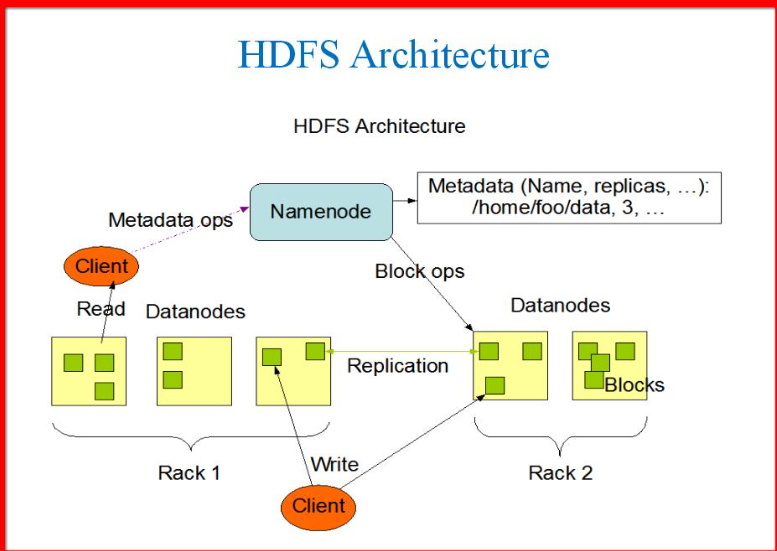
**NameNode**: The NameNode is the master server that manages the metadata (directory structure, file system operations, file locations, etc.). It knows where the data is stored on the DataNodes but doesn’t store the actual data itself.

**DataNode**: The DataNodes are the worker nodes in HDFS that store the actual data. Each DataNode is responsible for serving read/write requests for the data blocks it manages.

**Secondary NameNode**: The Secondary NameNode periodically checkpoints the metadata from the NameNode. It doesn’t provide redundancy for the NameNode but helps to ensure the metadata is periodically saved to prevent data loss in case of a crash.

**Client**: The Client interacts with the HDFS system, sending requests to the NameNode and DataNodes to perform read/write operations.

**Block**: HDFS splits files into fixed-size blocks (default is 128 MB) and distributes these blocks across multiple DataNodes for fault tolerance and parallel processing.



#### **HDFS Data Flow**

**Write Operation**:

* 1. The client requests the NameNode to create a new file.
  2. The NameNode responds with a list of DataNodes where blocks of the file will be stored.
  3. The client writes data to the corresponding DataNodes.

**Read Operation**:

* 1. The client requests the NameNode for the locations of blocks for the file.
  2. The NameNode returns the DataNodes storing the blocks.
  3. The client reads the file data directly from the DataNodes.

#### **4. HDFS Storage**

* **Block Replication**: By default, HDFS replicates blocks three times (this can be configured) to ensure fault tolerance. If one replica is lost, the system can still access the data from another replica.

#### **5. HDFS Command Structure**

HDFS provides several commands for managing files and directories. These commands resemble Linux shell commands, but they operate on HDFS. Below are some of the most commonly used commands:

#### **Basic HDFS Commands**

**Making Directories**

* + **Command**: hdfs dfs -mkdir /path/to/directory
  + **E**

hdfs dfs -mkdir /user/tfl\_data/underground\_status/

* + This command creates a new directory in HDFS.

**Listing Files and Directories**

* + **Command**: hdfs dfs -ls /path/to/directory
  + **Example**:

hdfs dfs -ls /user/tfl\_data/

* + This command lists files and directories in HDFS at the specified path.

**Copying Files to HDFS**

* + **Command**: hdfs dfs -copyFromLocal /local/file /path/in/hdfs

hdfs dfs -copyFromLocal /local\_path/status.csv /user/tfl\_data/underground\_status/

* + This command copies a local file to HDFS.

**Copying Files from HDFS to Local System**

* + **Command**: hdfs dfs -copyToLocal /path/in/hdfs /local/directory

**e**

hdfs dfs -copyToLocal /user/tfl\_data/underground\_status/status.csv /local\_path/

**Removing Files or Directories**

* + **Command**: hdfs dfs -rm /path/to/file

**Example**:

hdfs dfs -rm /user/tfl\_data/underground\_status/status.csv

* + This command removes a file from HDFS.

**Changing Permissions**

* + **Command**: hdfs dfs -chmod permissions /path/to/file

hdfs dfs -chmod 755 /user/tfl\_data/underground\_status/

**Changing Owner**

* + **Command**: hdfs dfs -chown owner:group /path/to/file

hdfs dfs -chown tfl\_user:tfl\_group /user/tfl\_data/underground\_status/

**Displaying File Contents**

* + **Command**: hdfs dfs -cat /path/to/file

hdfs dfs -cat /user/tfl\_data/underground\_status/status.csv

**Getting File Information**

* + **Command**: hdfs dfs -du -h /path/to/directory

hdfs dfs -du -h /user/tfl\_data/underground\_status/

* + This command shows the size of a file or directory in HDFS.

**Appending Data to a File**

* + **Command**: hdfs dfs -appendToFile /local/file /path/in/hdfs

**Example**:

hdfs dfs -appendToFile /local\_path/new\_data.csv /user/tfl\_data/underground\_status/status.csv

**Checking Disk Usage**

* + **Command**: hdfs dfs -df /path

hdfs dfs -df /user/tfl\_data/

**Viewing the HDFS File System**

* + **Command**: hdfs dfsadmin -report
  + This command provides a summary of HDFS health, including the number of DataNodes and the disk space usage.

#### **6. Important HDFS Terms to Remember**

* **NameNode**: The master node that manages metadata (directories, files, and blocks).
* **DataNode**: The worker node that stores data.
* **Block**: A fixed-size unit of data stored in HDFS (typically 128 MB).
* **Replication**: The number of copies of each data block (default is 3).
* **HDFS URI**: The URI used to reference HDFS files, typically starting with hdfs://hostname:port.
* **HDFS Permissions**: Similar to UNIX file permissions (read, write, execute).
* **Checkpoint**: A snapshot of HDFS metadata that is periodically saved by the Secondary NameNode to prevent data loss.

#### **7. Fault Tolerance in HDFS**

* **Replication Factor**: The default replication factor in HDFS is 3, meaning each block is stored on 3 different DataNodes. If one DataNode fails, the data can still be accessed from another replica.
* **Block Placement Policy**: HDFS places the first replica of a block on the local DataNode and the second and third replicas on different nodes (possibly on different racks) to improve fault tolerance.

#### **8. HDFS Advantages**

* **Fault Tolerance**: Data replication ensures that even if one DataNode fails, the data is still available.
* **High Throughput**: Designed for large-scale data processing, it can handle large files and provide high throughput.
* **Scalability**: HDFS is highly scalable and can store petabytes of data by simply adding more DataNodes to the cluster.

# **MobXterm**

**CONTENT**

### **What is MobaXterm?**

**MobaXterm** is a software tool that provides a **remote computing environment**, combining features like an **SSH client**, **X11 server**, **RDP**, and **SFTP** into a single application. It allows users to connect to and manage remote systems securely, often used by developers and IT professionals.

### **Key Features:**

1. **SSH Client**: Securely access remote systems via SSH.
2. **X11 Server**: Run remote graphical applications locally.
3. **RDP/VNC Support**: Remotely control desktop systems.
4. **SFTP Browser**: Secure file transfers between local and remote systems.
5. **Tabbed Interface**: Manage multiple remote sessions in tabs.
6. **Portable Version**: Can be run from a USB stick without installation.

### **Uses:**

* Remote **server management** and secure file transfer.
* Running **Linux/Unix commands** on Windows.
* Connecting to **multiple types of remote servers** (SSH, RDP, VNC, etc.).

### **History:**

* Released in **2007** by **Mobatek**, MobaXterm was designed as a powerful tool for **remote administration** and **network management**, combining various network tools into one easy-to-use interface.

### **More about MobaXterm:**

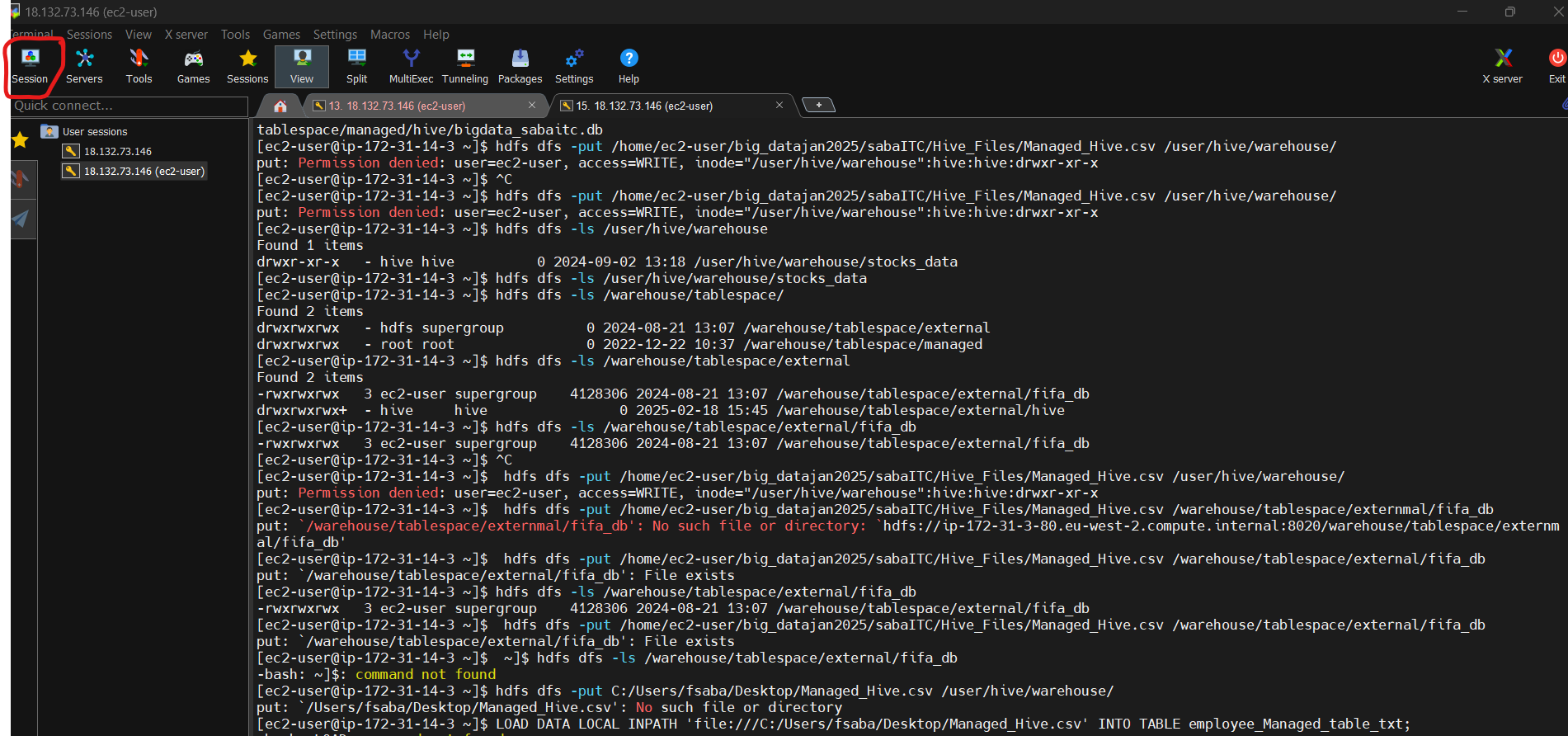
* **Unix Commands on Windows**: Provides an embedded terminal with Linux commands like ls, grep, etc.
* **SSH Tunneling & Port Forwarding**: Secure connections to remote services through SSH.
* **Multi-session Support**: Open multiple sessions in tabs (SSH, RDP, etc.) for easy multitasking.
* **X11 Forwarding**: Run graphical apps from remote servers on your local machine.
* **Customizable Profiles**: Create and save custom settings for different sessions.
* **Security**: Supports **SSH public key authentication** for secure connections.

In short, MobaXterm is a powerful, all-in-one tool for managing remote systems and working with multiple protocols like SSH, RDP, and X11.

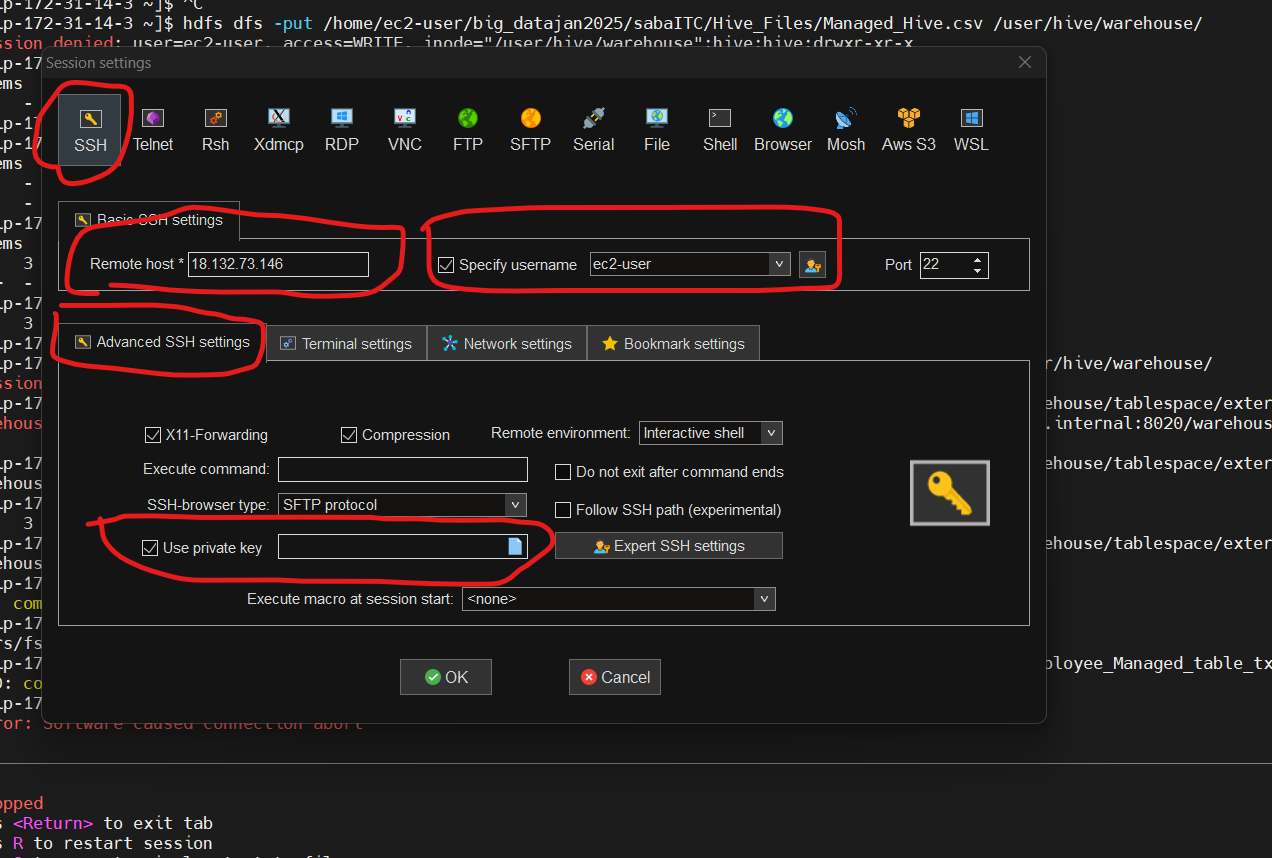
***CODE PROCESS IN MobaXterm***

Open Control Process:

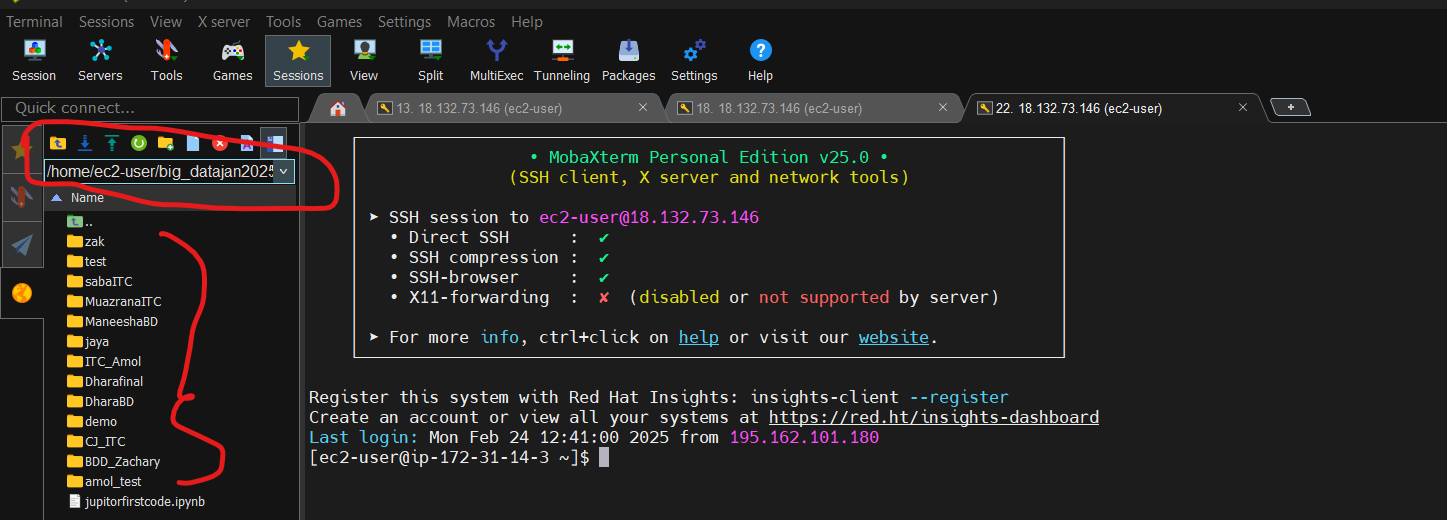
* Configure the MobaXterm, download the MobaXterm.exe file from the internet.
* File is at this location in my system **(C:\Users\fsaba\Documents\Software-Download\MobaXterm\_Portable\_v25.0)**
* Open the MobaXterm\_Personal\_25.0 app



* Open a new session, on click session. It will open a window where we have to giving the details as mentioned in screenshot.
* Use private key (you have to add the key, which need to give for access of the server) Key location path is: (**C:\Users\fsaba\Documents\Software-Download\MobaXterm\_Portable\_v25.0\test\_key.pem**)



* On right side, we can see the folder which we create. Using some command line code.



**Basic Launch:** To start MobaXterm from the command prompt (on Windows), navigate to the MobaXterm directory or use the full path:

* **MobaXterm.exe**

**Open a Specific Session:** If you have predefined sessions in MobaXterm (e.g., SSH, RDP, or others), you can launch a specific session directly using:

* **MobaXterm.exe -m "SessionName"**

### **SFTP Commands for File Transfer:**

MobaXterm includes an **SFTP** client that allows you to transfer files securely over SSH:

* **Connect to a Server via SFTP:**
* sftp username@hostname

This opens an SFTP session, where you can use commands like:

* get filename to download files.
* put filename to upload files.
* ls to list files in the current directory.
* cd path to change directories on the remote server.

**Start an SFTP Session from the Command Line:** You can directly launch **MobaXterm’s** embedded SFTP client in a new tab with:

* **MobaXterm.exe -newtab "sftp username@hostname"**

you can run typical **Unix/Linux commands**. Some commonly used commands:

* ls – List directory contents.
* cd – Change directory.
* cat filename – Display the contents of a file.
* grep 'pattern' filename – Search for a pattern in a file.
* top – Display running processes and system resource usage.
* ps aux – List all running processes.
* chmod +x script.sh – Change permissions to make a script executable.

**Common MobaXterm Command Lines:**

For connecting to server, can be done in multiple ways:

1. **SSH Connection (ssh [john\_doe@192.168.1.100](mailto:john_doe@192.168.1.100))**
2. **SFTP (Secure File Transfer Protocol) (sftp username@hostname\_or\_ip)**

* put local\_file\_path — To upload a file from your local machine to the remote server.
* get remote\_file\_path — To download a file from the remote server to your local machine.

Like we have the command:

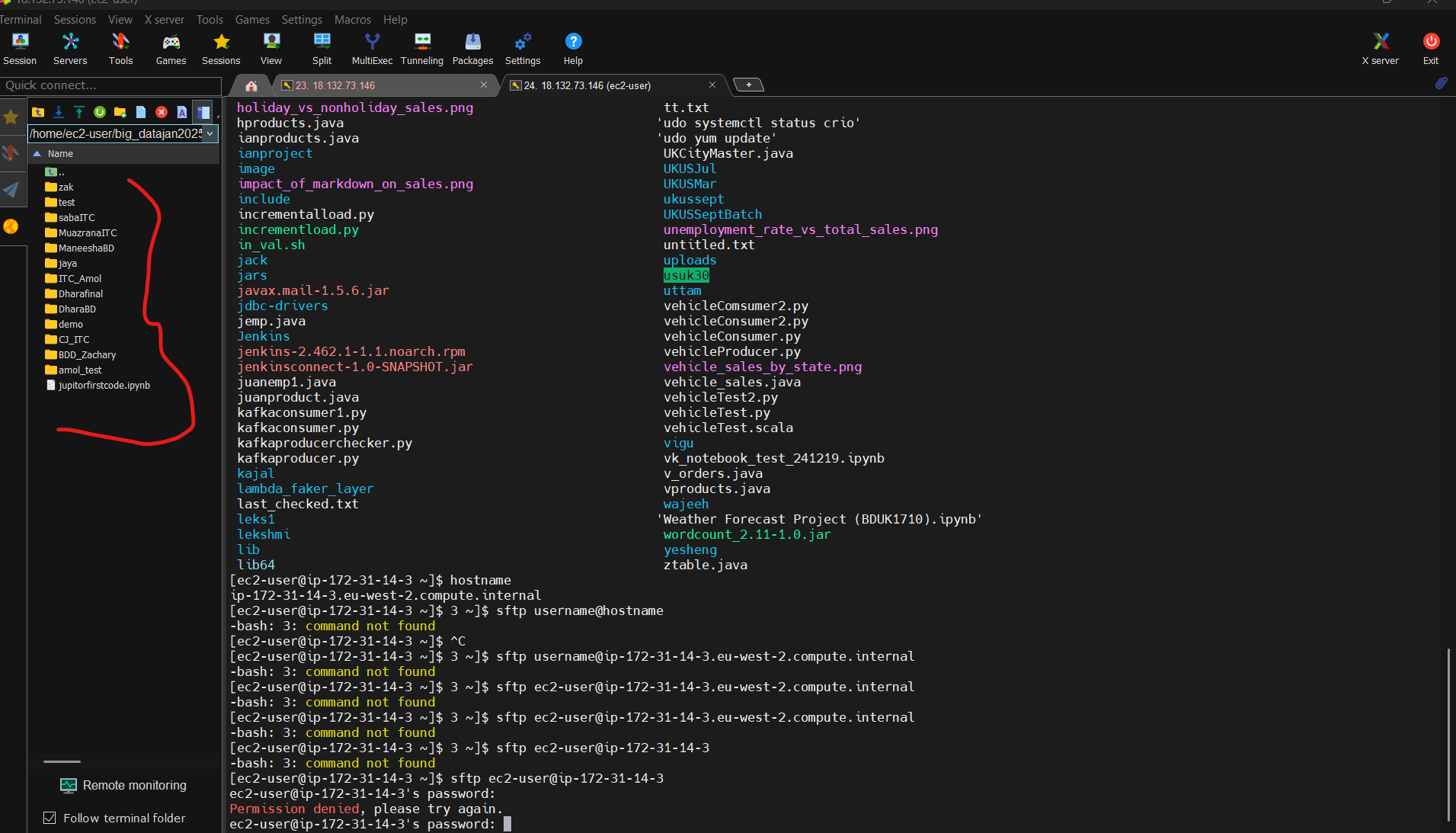
* ls,
* cd directory\_name,
* mkdir new\_directory\_name,
* get /path/to/remote/file /path/to/local/destination,
* put C:/Users/John/Documents/myfile.txt /home/john\_doe/uploads/
* This will upload/ get myfile.txt from your local machine to the /home/john\_sa/uploads/ directory on the remote server.

1. **scp Command**

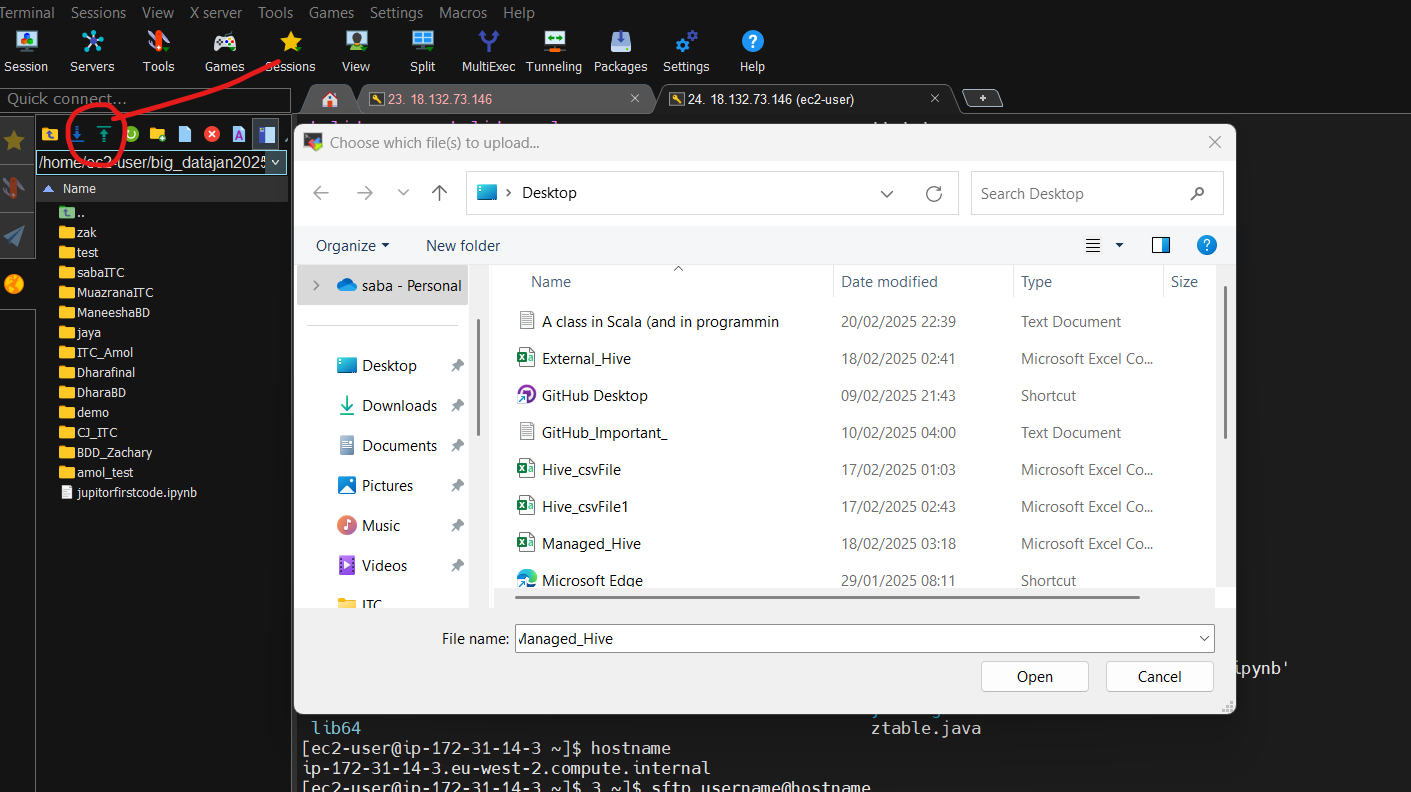
**Uploading the file:**

scp C:/Users/John/Documents/myfile.txt [john\_doe@192.168.1.100:/home/john\_doe/uploads/](mailto:john_doe@192.168.1.100:/home/john_doe/uploads/)

Here is the screenshot, you can see the folder in right hand side and you can check the file which you have uploaded at your location:



Manually, you can upload the file in this way:



### **File and Directory Commands:**

* **pwd** – Print the current working directory.
* **ls** – List the contents of the current directory.
* **cd** – Change the current directory.
* **mkdir** – Create a new directory.
* **rmdir** – Remove an empty directory.
* **rm** – Remove files or directories.
  + Example: rm filename
  + Example (recursive): rm -r directory\_name
* **cp** – Copy files or directories.
  + Example: cp source\_file destination
* **mv** – Move or rename files or directories.
  + Example: mv source destination

### **File Viewing and Editing Commands:**

* **cat** – View the contents of a file.
* **less** – View the contents of a file with scroll functionality. (press q button to exit)
* **nano** – Simple text editor for file editing.
* **vim** or **vi** – Advanced text editor for file editing.

### **System Commands:**

* **top** – Display system processes (Task Manager-like).
* **ps** – List running processes.
  + Example: ps aux
* **df** – Display disk space usage.
  + Example: df -h
* **free** – Display memory usage.
  + Example: free -h
* **uptime** – Show system uptime (how long it has been running).
* **hostname** – Display the system's hostname.
* **whoami** – Show the current logged-in user's username.
* **shutdown** – Shutdown the server.
  + Example: sudo shutdown now
  + Example (schedule): sudo shutdown +10 (in 10 minutes)
* **reboot** – Reboot the server.

### **Networking Commands:**

* **ping** – Ping a host to check connectivity.
  + Example: ping hostname\_or\_ip
* **ifconfig** – Show network interface configuration.
* **netstat** – Show network connections and statistics.
  + Example: netstat -tuln

### **File Transfer Commands:**

* **scp** – Securely copy files between hosts.
  + Example: scp local\_file username@hostname:/remote/destination

Uploading the file from local server to MobaXterm:  
  
File location is: **C:\Users\fsaba\Documents\Files\_Import\ Student\_Details\_MbxTerm.csv**

Practice File is also at same location: **C:\Users\fsaba\Documents\Files\_Import\ MbaXterm\_Basic\_Commands.txt**

**You can save the file in HDFS (Hadoop Distributed File System) using this mobaXterm.**

These commands allow you to interact with files stored in HDFS and perform various operations such as uploading, downloading, viewing, and modifying files.

* hdfs dfs -ls /user/hadoop/
* hdfs dfs -mkdir /user/hadoop/new\_directory
* hdfs dfs -put localfile.txt /user/hadoop/
* hdfs dfs -get /user/hadoop/file.txt /home/localuser/
* hdfs dfs -copyToLocal /path/to/hdfs/file local\_directory
* Similar to -get, but overwrites the file if it exists in the local directory.
* Example: hdfs dfs -copyToLocal /user/hadoop/file.txt /home/localuser/
* hdfs dfs -rm /user/hadoop/file.txt
* hdfs dfs -rm -r /user/hadoop/directory\_name
* hdfs dfs -cat /user/hadoop/file.txt
* hdfs dfs -copyFromLocal localfile.txt /user/hadoop/
* hdfs dfs -du /user/hadoop/(Show summarized disk usage (only total size of the directory).
* hdfs dfs -tail /user/hadoop/file.txt
* hdfs dfs -chmod 755 /user/hadoop/file.txt
* hdfs dfs -chown hadoop:hadoop /user/hadoop/file.txt (Change the ownership of a file or directory in HDFS (run as the HDFS user).
* hdfs dfs -stat /user/hadoop/file.txt
* **SUDO:**

**Sudo** stands for **"SuperUser Do"**. It is a command in Linux/Unix that allows a user to execute commands with elevated (administrator or root) privileges. This is useful for performing tasks that require higher permissions, like installing software or modifying system files.

* sudo -u hdfs hdfs dfs -ls /user/hadoop/  
    
  just adding **sudo –u** **hdfs** to all the hadoop command.

**Important command:**

* sudo -u hdfs hdfs dfs -setrep -R 3 /path/to/hdfs/directory

Set the replication factor of a directory or file in HDFS (run as the HDFS user).

* Example: **sudo -u hdfs hdfs dfs -setrep -R 3 /user/hadoop/**

Get the count, by giving the file name:

**sudo -u hdfs hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop-mapreduce/hadoop-mapreduce-examples.jar wordcount /tmp/big\_datajan2025/saba/text1.txt /tmp/big\_datajan2025/saba/count**