Advanced Database Management System Lab

Subject Code: MCAL13

A Practical Journal Submitted in Fulfilment of

the Degree of

MASTER

In

COMPUTER APPLICATION

Year 2023-2024

By

Mr. Vishwakarma Dheeraj Kumar Jaynath

(Application Id: - 78383)

Semester- 1

Under the Guidance of

Prof.

Institute of Distance and Open Learning

Vidya Nagari, Kalina, Santacruz East – 400098.

University of Mumbai

PCP Centre

[Vidyavardhini's College of Technology – Vasai Road, Palghar 401202]

Institute of Distance and Open Learning,

Vidya Nagari, Kalina, Santacruz (E) -400098

CERTIFICATE

This to certify that, **Mr. Vishwakarma Dheeraj Kumar Jaynath** appearing **Master in Computer Application (Semester I) Application ID: 78383** has satisfactory completed the prescribed practical of **MCAL13-Advanced Database Management System Lab** as laid down by the University of Mumbai for the academic year 2023-24

Teacher in charge

Examiners

Coordinator
IDOL, MCA
University of Mumbai

Date: -31/01/2024

Place: - Vasai

Index

Sr. No.	Practical	Signature
1.	Implementation of Data partitioning through Range	
2.	Implementation of Analytical queries like Roll_UP, CUBE, First, Last, Rank AND Dense Rank	
3.	Implementation of Abstract Data Type & Reference	
4.	To study ETL process	
5.	Installation of R datatype in R programming Reading and Writing data to and from R	
6.	To study Linear Regression.	
7.	To study Analysis of Regression	
8.	To study Logistic Regression	
9.	To study support Vector Machine	
10	To study varied Algorithm	

Experiment No: 01

RANGE Partitioning in MySQL

Aim: Implementation of Data partitioning through Range

```
C:\Program Files (x86)\MySQL\MySQL Server 5.1\bin\mysql.exe
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 9
Server version: 5.1.28-rc-community MySQL Community Server (GPL)

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> use mca;
Database changed
mysql> CREATE TABLE tr1 (id INT, name VARCHAR(50), purchased
    -> DATE)
    -> PARTITION BY RANGE( YEAR(purchased) ) (
    -> PARTITION p0 VALUES LESS THAN (1990),
    -> PARTITION p1 VALUES LESS THAN (1995),
    -> PARTITION p2 VALUES LESS THAN (2000),
    -> PARTITION p3 VALUES LESS THAN (2005),
    -> PARTITION p4 VALUES LESS THAN (2010),
    -> PARTITION p5 VALUES LESS THAN (2015)
    -> );
Query OK, 0 rows affected (0.30 sec)
```

```
mysql>
mysql>  INSERT INTO tr1 VALUES
    -> (1, 'desk organiser', '2003-10-15'),
    -> (2, 'alarm clock', '1997-11-05'),
    -> (3, 'chair', '2009-03-10'),
    -> (4, 'bookcase', '1989-01-10'),
    -> (5, 'exercise bike', '2014-05-09'),
    -> (6, 'sofa', '1987-06-05'),
    -> (7, 'espresso maker', '2011-11-22'),
    -> (8, 'aquarium', '1992-08-04'),
    -> (9, 'study desk', '2006-09-16'),
    -> (10, 'lava lamp', '1998-12-25');
Query OK, 10 rows affected (0.05 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

```
mysql> select * from tr1;
+------+---------------+------------+
| id   | name          | purchased  |
+------+---------------+------------+
|    4 | bookcase      | 1989-01-10 |
|    6 | sofa          | 1987-06-05 |
|    8 | aquarium      | 1992-08-04 |
|    2 | alarm clock   | 1997-11-05 |
|   10 | lava lamp     | 1998-12-25 |
|    1 | desk organiser | 2003-10-15 |
|    3 | chair         | 2009-03-10 |
|    9 | study desk    | 2006-09-16 |
|    5 | exercise bike | 2014-05-09 |
|    7 | espresso maker | 2011-11-22 |
+------+---------------+------------+
10 rows in set (0.02 sec)

mysql>
```

```
mysql> SELECT PARTITION_NAME, TABLE_ROWS FROM
    -> INFORMATION_SCHEMA.PARTITIONS WHERE
    -> TABLE_NAME='tr1';
+----------------+------------+
| PARTITION_NAME | TABLE_ROWS |
+----------------+------------+
| p0             |          2 |
| p1             |          1 |
| p2             |          2 |
| p3             |          1 |
| p4             |          2 |
| p5             |          2 |
+----------------+------------+
6 rows in set (0.27 sec)

mysql>
```

Experiment No: 02

ANALYTICAL QUERIES

Aim: Implementation of Analytical queries like Roll_UP, CUBE, First, Last, Rank AND Dense Rank.

```
SQL> CREATE TABLE emp(
  2  empno NUMBER(4) CONSTRAINT pk_emp PRIMARY KEY,
  3  ename VARCHAR2(10),
  4  job VARCHAR2(10),
  5  mgr NUMBER(4),
  6  hiredate DATE,
  7  sal NUMBER(7,2),
  8  comm NUMBER(7,2),
  9  deptno NUMBER(2));

Table created.

SQL> INSERT INTO emp VALUES(1,'Hema','Developer',2,'22-May-2020',25000,2000,4)
  2  ;

1 row created.

SQL> INSERT INTO emp VALUES(2,'Ram','Developer',2,'20-April-2019',45000,2300,4);

1 row created.

SQL> INSERT INTO emp VALUES(3,'Vrudhi','Tester',4,'20-April-2019',30000,8000,5);

1 row created.
```

```
SQL> INSERT INTO emp VALUES(4,'Rahul','Tester',4,'5-November-2018',50000,8000,5);

1 row created.
```

```
SQL> SELECT * FROM emp;

     EMPNO ENAME      JOB              MGR HIREDATE          SAL       COMM
---------- ---------- ---------- ---------- --------- ---------- ----------
    DEPTNO
----------
         1 Hema       Developer            2 22-MAY-20      25000       2000
         4

         2 Ram        Developer            2 20-APR-19      45000       2300
         4

         3 Vrudhi     Tester               4 20-APR-19      30000       8000
         5


     EMPNO ENAME      JOB              MGR HIREDATE          SAL       COMM
---------- ---------- ---------- ---------- --------- ---------- ----------
    DEPTNO
----------
         4 Rahul      Tester               4 05-NOV-18      50000       8000
         5
```

```
SQL> SELECT empno,sal,sum(sal) as Totalsal from emp group by rollup(empno,sal);

     EMPNO         SAL   TOTALSAL
---------- ---------- ----------
         1      25000      25000
         1                 25000
         2      45000      45000
         2                 45000
         3      30000      30000
         3                 30000
         4      50000      50000
         4                 50000
                          150000

9 rows selected.

SQL> _
```

```
SQL> SELECT empno,sal,sum(sal) as Totalsal from emp group by cube(empno,sal);

      EMPNO          SAL    TOTALSAL
---------- ----------- -----------
                         150000
                 30000       30000
                 50000       50000
                 25000       25000
                 45000       45000
          1                  25000
          1       25000      25000
          2                  45000
          2       45000      45000
          3                  30000
          3       30000      30000

      EMPNO          SAL    TOTALSAL
---------- ----------- -----------
          4                  50000
          4       50000      50000

13 rows selected.
```

RANK

```
SQL> SELECT empno,deptno,sal,RANK() OVER (PARTITION BY deptno ORDER BY sal) AS myrank FROM emp;

      EMPNO      DEPTNO          SAL      MYRANK
---------- ---------- ---------- ----------
          1          4      25000          1
          2          4      45000          2
          3          5      30000          1
          4          5      50000          2
```

```
SQL> SELECT empno,deptno,sal, DENSE_RANK() OVER (Partition By deptno ORDER By sal) as myrank FROM emp;

      EMPNO      DEPTNO          SAL      MYRANK
---------- ---------- ---------- ----------
          1          4      25000          1
          2          4      45000          2
          3          5      30000          1
          4          5      50000          2
```

DENSE_RANK

```
SQL> SELECT * FROM (SELECT empno,deptno,sal, DENSE_RANK() OVER (Partition By deptno ORDER By sal DESC) as myrank FROM emp)WHERE myrank<=2;

     EMPNO      DEPTNO         SAL      MYRANK
---------- ---------- ---------- ----------
         2          4      45000          1
         1          4      25000          2
         4          5      50000          1
         3          5      30000          2

SQL>
```

FIRST AND LAST

```
SQL> SELECT empno,deptno,sal, MIN(sal) KEEP (DENSE_RANK FIRST ORDER By sal) OVER (Partition By deptno) as lowest,MAX(sal) KEEP (DENSE_RANK LAST ORDER By sal) OVER (Part
ition By deptno) As highest FROM emp ORDER By deptno,sal;

     EMPNO     DEPTNO        SAL     LOWEST    HIGHEST
---------- ---------- ---------- ---------- ----------
         1          4      25000      25000      45000
         2          4      45000      25000      45000
         3          5      30000      30000      50000
         4          5      50000      30000      50000

SQL>
```

LAG

```
SQL> SELECT deptno,empno,ename,job,sal, LAG(sal,1,0) OVER (PARTITIOn By deptno ORDER BY sal) As sal_prev FROM emp;

    DEPTNO      EMPNO ENAME      JOB               SAL   SAL_PREV
---------- ---------- ---------- ---------- ---------- ----------
         4          1 Hema       Developer       25000          0
         4          2 Ram        Developer       45000      25000
         5          3 Vrudhi     Tester          30000          0
         5          4 Rahul      Tester          50000      30000

SQL>
```

LEAD

```
SQL> SELECT empno,ename,job,sal, LEAD(sal,1,0) OVER (ORDER By sal) As sal_next, LEAD(sal,1,0) OVER (ORDER By sal)-sal As sal_diff FROM emp;

     EMPNO ENAME      JOB              SAL   SAL_NEXT   SAL_DIFF
---------- ---------- ---------- ---------- ---------- ----------
         1 Hema       Developer       25000      30000       5000
         3 Vrudhi     Tester          30000      45000      15000
         2 Ram        Developer       45000      50000       5000
         4 Rahul      Tester          50000          0     -50000
```

Experiment No: 03

Aim: Implementation of Abstract Data Type & Reference

Customer_reltab:

CREATE TABLE Customer_reltab (CustNo NUMBER NOT NULL, CustName VARCHAR2(200) NOT NULL, Street VARCHAR2(200) NOT NULL, City VARCHAR2(200) NOT NULL,
State CHAR(2) NOT NULL, Zip VARCHAR2(20) NOT NULL, Phone1 VARCHAR2(20), Phone2 VARCHAR2(20), Phone3 VARCHAR2(20), PRIMARY KEY (CustNo));

```
SQL> CREATE TABLE Customer_reltab (
  2  CustNo NUMBER NOT NULL,
  3  CustName VARCHAR2(200) NOT NULL,
  4  Street VARCHAR2(200) NOT NULL,
  5  City VARCHAR2(200) NOT NULL,
  6  State CHAR(2) NOT NULL,
  7  Zip VARCHAR2(20) NOT NULL,
  8  Phone1 VARCHAR2(20),
  9  Phone2 VARCHAR2(20),
 10  Phone3 VARCHAR2(20),
 11  PRIMARY KEY (CustNo));

Table created.

SQL> desc customer_reltab;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 CUSTNO                                    NOT NULL NUMBER
 CUSTNAME                                  NOT NULL VARCHAR2(200)
 STREET                                    NOT NULL VARCHAR2(200)
 CITY                                      NOT NULL VARCHAR2(200)
 STATE                                     NOT NULL CHAR(2)
 ZIP                                       NOT NULL VARCHAR2(20)
 PHONE1                                             VARCHAR2(20)
 PHONE2                                             VARCHAR2(20)
 PHONE3                                             VARCHAR2(20)

SQL>
```

PurchaseOrder_reltab:

CREATE TABLE PurchaseOrder_reltab (PONo NUMBER, /* purchase order no */
Custno NUMBER references Customer_reltab, /* Foreign KEY referencing
customer */ OrderDate DATE, /* date of order */ ShipDate DATE, /* date to be shipped */
ToStreet VARCHAR2(200), /* shipto address */ ToCity VARCHAR2(200), ToState CHAR(2),
ToZip VARCHAR2(20), PRIMARY KEY(PONo));

```
SQL> CREATE TABLE PurchaseOrder_reltab (
  2  PONo NUMBER, /* purchase order no */
  3  Custno NUMBER references Customer_reltab, /* Foreign KEY
  4  referencing
  5  customer */
  6  OrderDate DATE, /* date of order */
  7  ShipDate DATE, /* date to be shipped */
  8  ToStreet VARCHAR2(200), /* shipto address */
  9  ToCity VARCHAR2(200),
 10  ToState CHAR(2),
 11  ToZip VARCHAR2(20),
 12  PRIMARY KEY(PONo));

Table created.

SQL>
```

Stock_reltab:

CREATE TABLE Stock_reltab (StockNo NUMBER PRIMARY KEY, Price NUMBER, TaxRate NUMBER);

```
SQL> CREATE TABLE Stock_reltab (
  2  StockNo NUMBER PRIMARY KEY,
  3  Price NUMBER,
  4  TaxRate NUMBER);

Table created.

SQL>
```

LineItems_reltab:

CREATE TABLE LineItems_reltab (LineItemNo NUMBER, PONo NUMBER REFERENCES PurchaseOrder_reltab, StockNo NUMBER REFERENCES Stock_reltab, Quantity NUMBER, Discount NUMBER, PRIMARY KEY (PONo, LineItemNo));

```
SQL> CREATE TABLE LineItems_reltab (
  2  LineItemNo NUMBER,
  3  PONo NUMBER REFERENCES PurchaseOrder_reltab,
  4  StockNo NUMBER REFERENCES Stock_reltab,
  5  Quantity NUMBER,
  6  Discount NUMBER,
  7  PRIMARY KEY (PONo, LineItemNo));

Table created.

SQL>
```

Inserting Values Under the Relational Model:

INSERT INTO Stock_reltab VALUES(1004, 6750.00, 2);
INSERT INTO Stock_reltab VALUES(1011, 4500.23, 2);
INSERT INTO Stock_reltab VALUES(1534, 2234.00, 2);
INSERT INTO Stock_reltab VALUES(1535, 3456.23, 2);
INSERT INTO Customer_reltab VALUES
(1, 'Jean Nance', '2 Avocet Drive', 'Redwood Shores', 'CA', '95054', '415-555-1212', NULL, NULL);
INSERT INTO Customer_reltab VALUES (2, 'John Nike', '323 College Drive', 'Edison', 'NJ', '08820', '609-555-1212', '201-555-1212', NULL);
INSERT INTO PurchaseOrder_reltab VALUES (1001, 1, SYSDATE, '10-MAY-1997', NULL, NULL, NULL, NULL);
INSERT INTO PurchaseOrder_reltab VALUES (2001, 2, SYSDATE, '20-MAY-1997',
'55 Madison Ave', 'Madison', 'WI', '53715');

Querying Data Under the Relational Model:

SELECT C.CustNo, C.CustName, C.Street, C.City, C.State, C.Zip, C.phone1, C.phone2, C.phone3, P.PONo, P.OrderDate, L.StockNo, L.LineItemNo, L.Quantity, L.Discount FROM Customer_reltab C, PurchaseOrder_reltab P, LineItems_reltab L Advanced Database Management System Lab
WHERE C.CustNo = P.CustNo AND P.PONo = L.PONo AND P.PONo = 1001;

Get the Total Value of Purchase Orders

SELECT P.PONo, SUM(S.Price * L.Quantity) FROM PurchaseOrder_reltab P, LineItems_reltab L, Stock_reltab S WHERE P.PONo= L.PONo AND L.StockNo = S.StockNo GROUP BYP.PONo;

Get the Purchase Order and Line Item Data for Stock Item 1004

SELECT P.PONo, P.CustNo, L.StockNo, L.LineItemNo, L.Quantity, L.Discount FROM PurchaseOrder_reltab P, LineItems_reltab L WHERE P.PONo = L.PONo AND L.StockNo = 1004;

Updating Data Under the Relational Model:

UPDATE LineItems_reltab SET Quantity = 20 WHERE PONo = 1001 AND StockNo = 1534;

Deleting Data Under the Relational Model:

DELETE FROM LineItems_reltab WHERE PONo = 1001;
DELETE FROM PurchaseOrder_reltab WHERE PONo = 1001;

Experiment No: 04

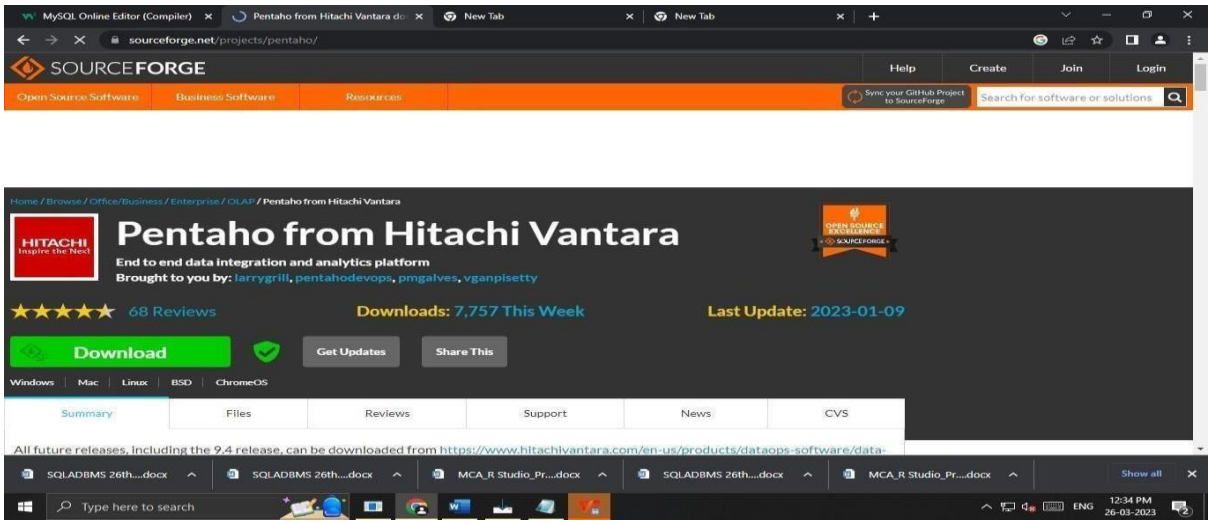
Aim: To study ETL process

* Installation steps for Pentaho Data Integration Software

Step 1: Download Pentaho Data Integration Software. The first thing we need is the Pentaho

Data Integration software that we'll be working with

You can download the set up file from link https://sourceforge.net/projects/pentaho/.



Press the "Download" button.

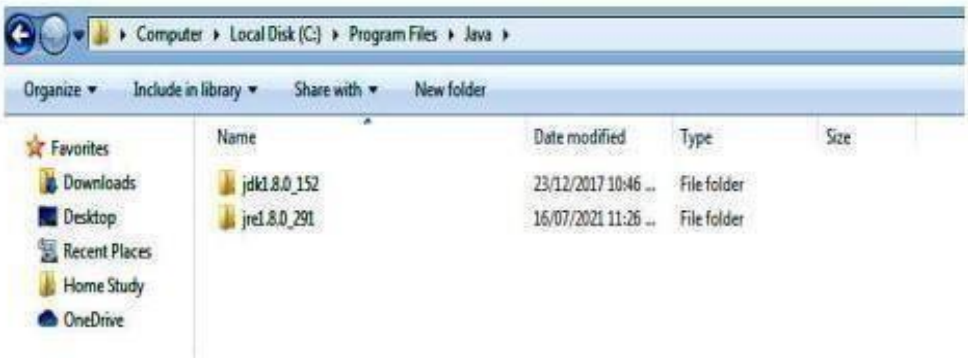
It will start downloading zip file on your computer. Once the downloading is finished, extract the files into a folder you want to.

Your folder should look something like this:

Step 2: Install the Java Dependencies, if Required.

To run Pentaho Data Integration, Java Runtime Environment and Java Development Kit are required. To check if you already have these installed, go to this path in your file explorer:

C:\Program Files\Java Or: C:\Program Files (x86)\Java If this folder exists and you see folders that look like:



Then you have the required files. If this folder doesn't exist or you don't see one or both of these folders, then you need to download JRE and/or JDK. To download JRE, go to this link https://java.com/en/download/ and press "Download."

Your page should look like this:

The installation window will look something like this:

Follow the instructions until finished.

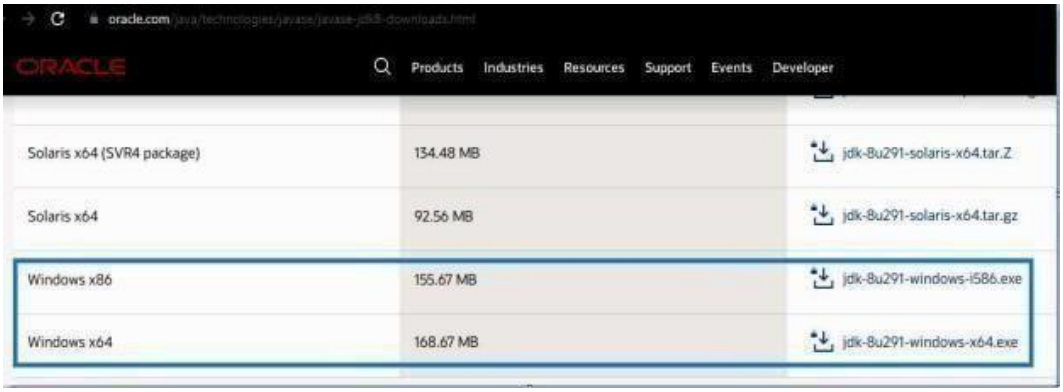
Next, download the JDK from this link

https://www.oracle.com/java/technologies/javase/javase-jdk8- downloads.html.

Please note that there have been substantial changes to the Oracle JDK licensing agreement. Details are available at Oracle Technology Network License Agreement for Oracle Java SE.

There will be a list of different operating systems to choose from. Scroll until you find Windows.

If you're unsure about which version (x64 or x86) your Windows is, select x86.

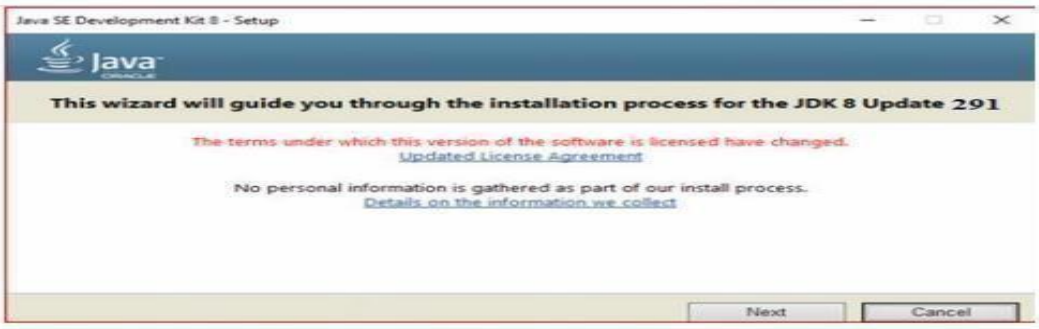


It will open following window

Press "Download".

If you're not logged in to Oracle, then you will be prompted to log in.
If you don't have an Oracle account, you need to create one in order to download the JDK.

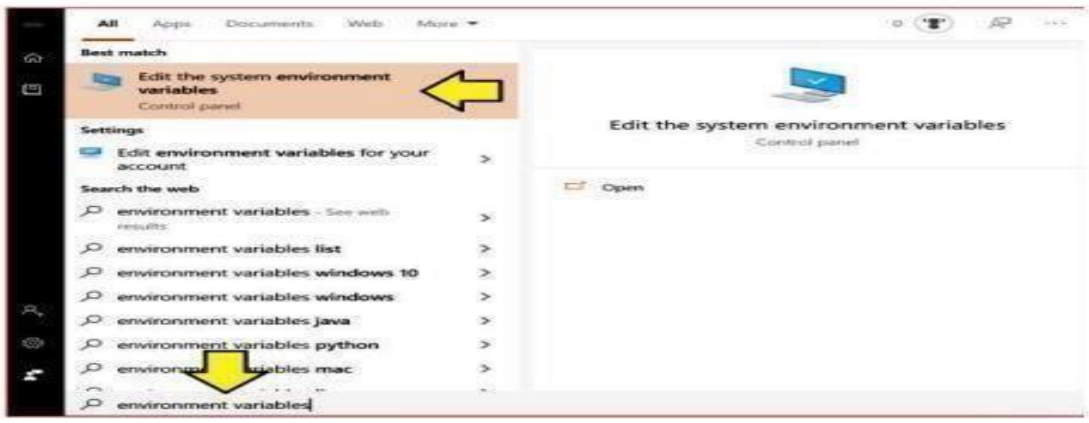
The installation setup will look like this:



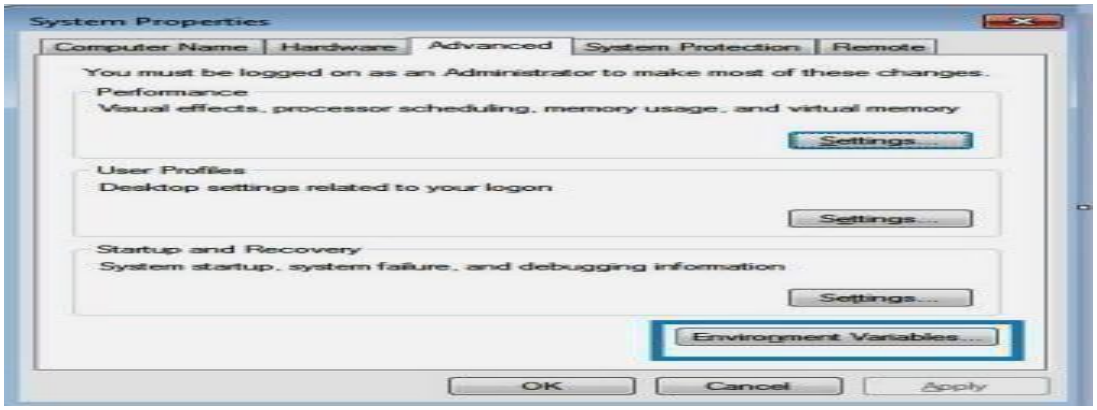
Graphics:

Hitachi Video Management Platform (VMP) has been designed from the ground up to meet the challenges of data storage and processing that new video systems present.

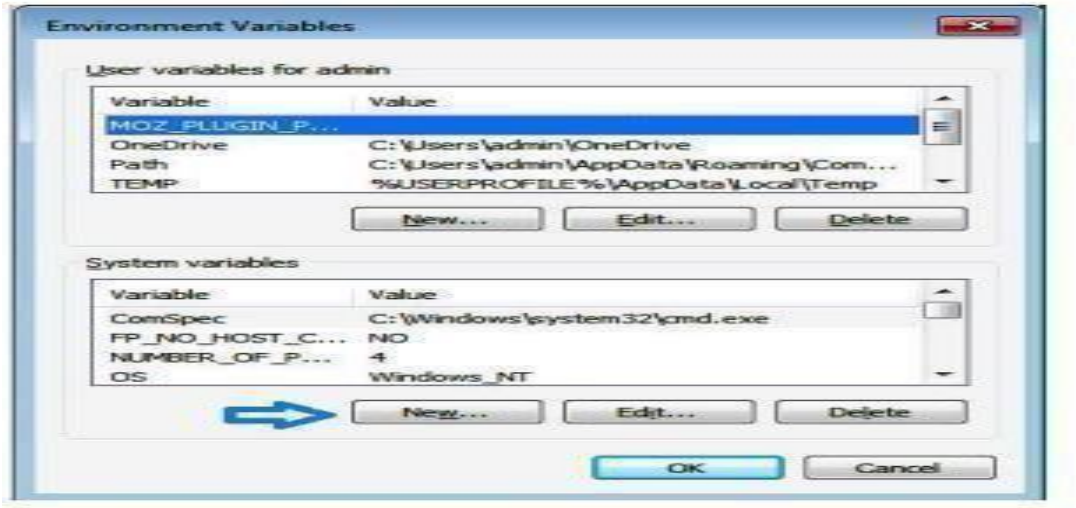
Step 3: Set Up the Environment Variables There are three environment variables that need to be set up. To open the environment variables menu type in "environment variables" in the Windows search bar like this:



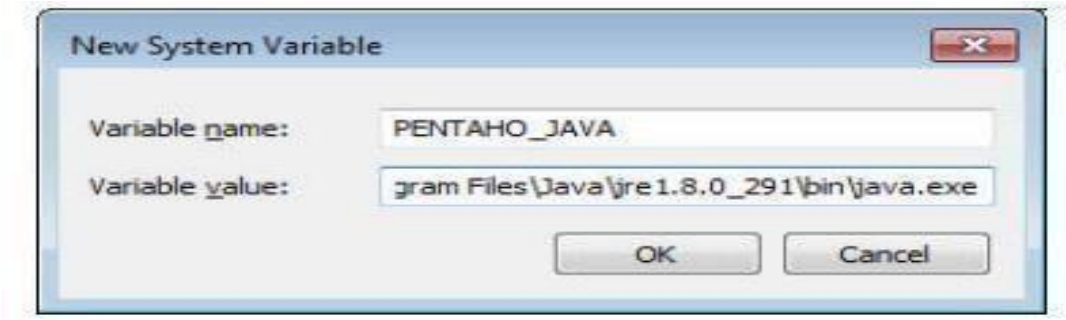
Click the “Edit the system environment variables” option.
That will open the “System Properties” window. Under Advanced tab …Click the “Environment Variables.” button at the bottom.



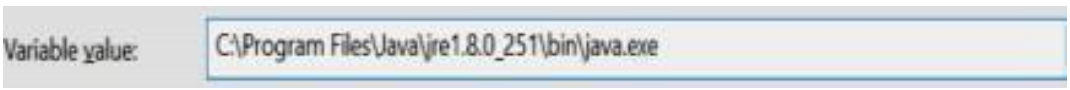
That will open a window that looks like this:



We need to add three new System variables.
Click the “New…” button under “System variables” and enter the following:

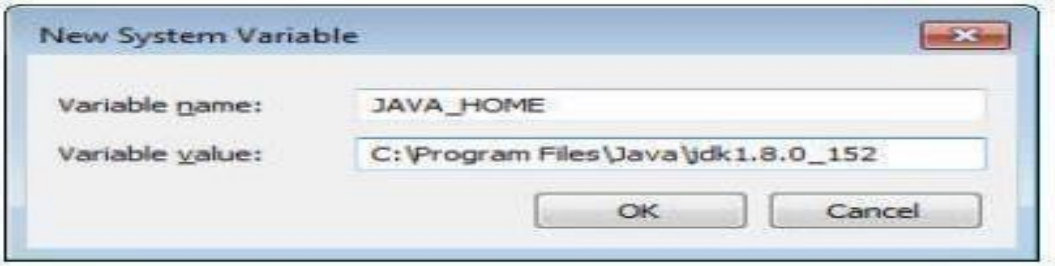


Make sure your variable value file path is the same one on your computer.



Press “OK” and then enter two more.

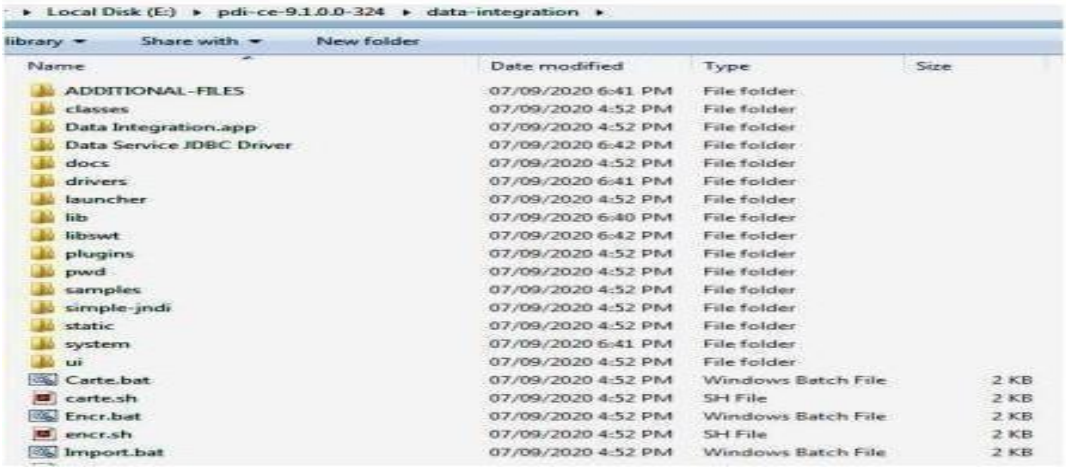
Press ‘’OK’’.



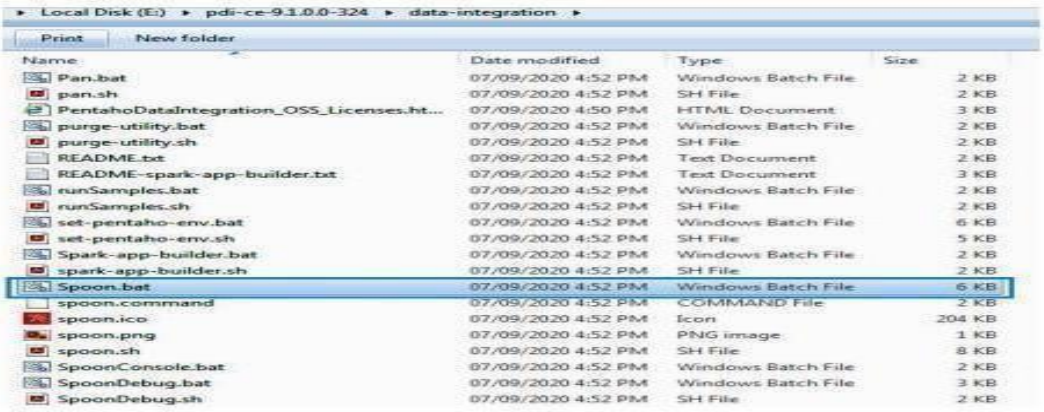
Press ‘’OK’’ and close all the previous windows by pressing “OK.”

Step 4: Open the Pentaho Data Integration App Now that Java is installed successfully and the environment variables are also set, we can start running the Pentaho Data Integrationapp.

The data integration folder that you downloaded earlier will looklike this:



The file that runs the app is called “Spoon.bat”.



Double click this file to open the Pentaho Data Integration app.

Now you can start using this app by pressing “New transformation” or “New job.”

Experiment No: 05

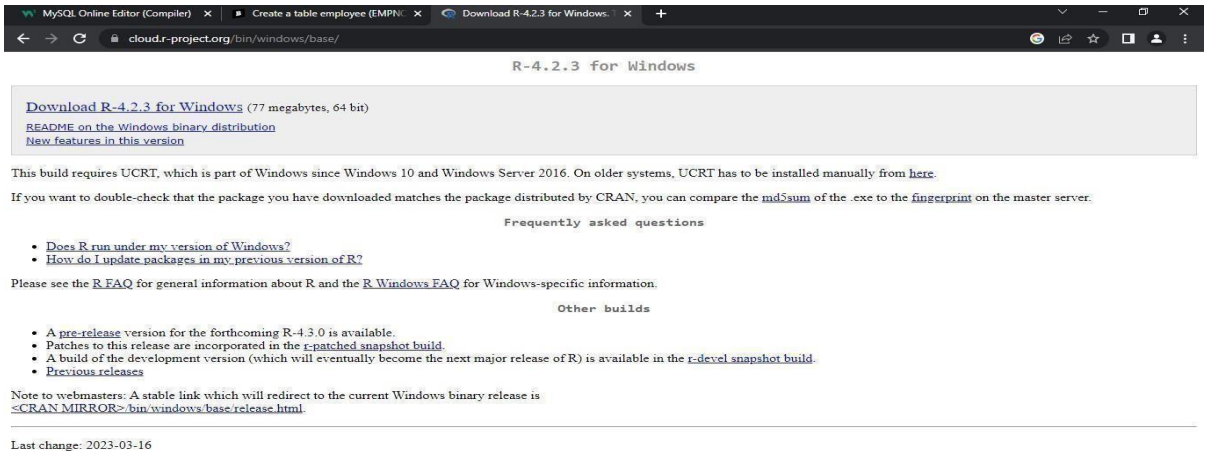
Aim:

1. **installation of R**
2. **datatype in R programming**
3. **Reading and Writing data to and from R.**

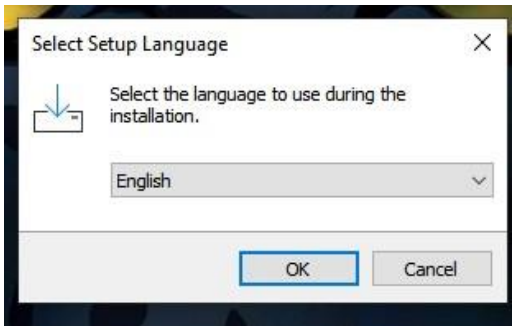
Aim: installation of R

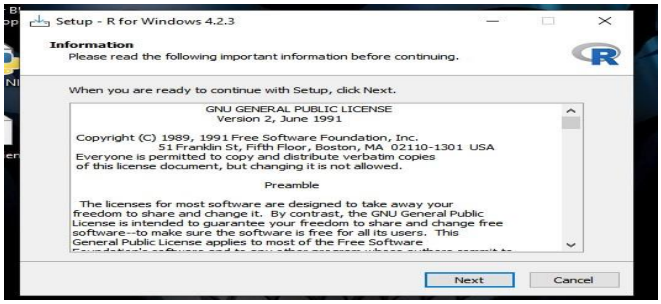
*** R Installation in Windows**

Steps used to install the R in Windows are as follows:
Step 1: First, we have to download the R setup from https://cloud.rproject.org/bin/windows/base/.

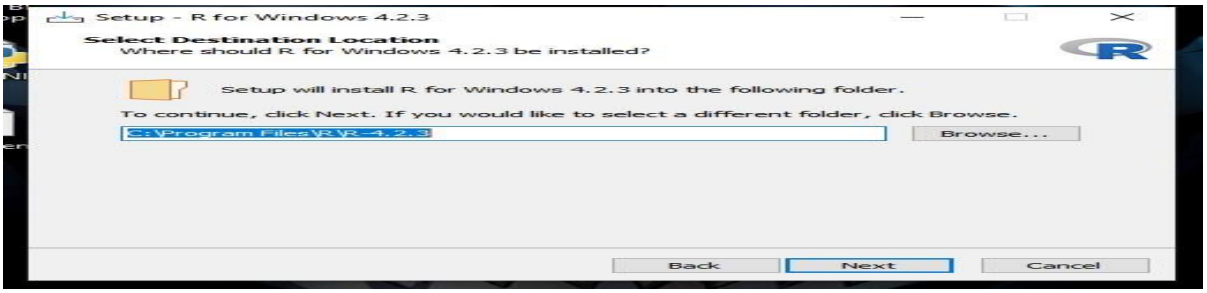


Step 2: When we click on Download R- 4.1.0 for windows, our downloading will start. Once the downloading is finished, we have to run the setup of R as follows:

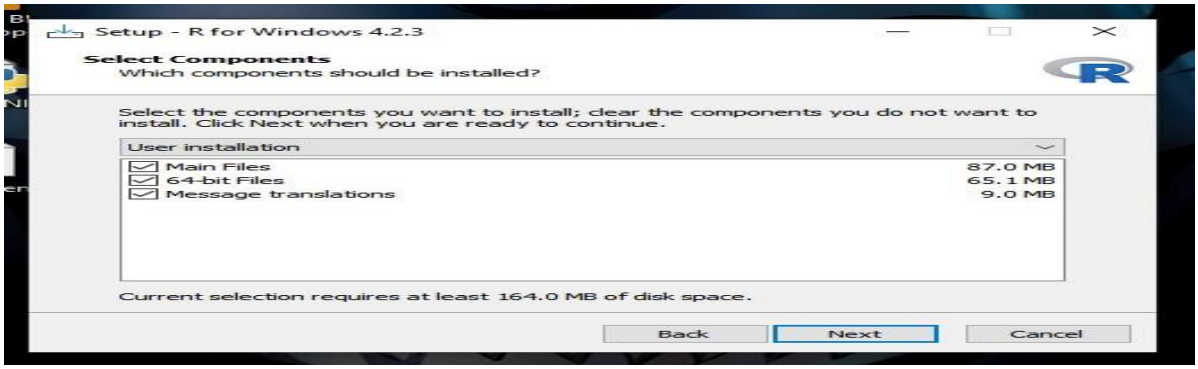




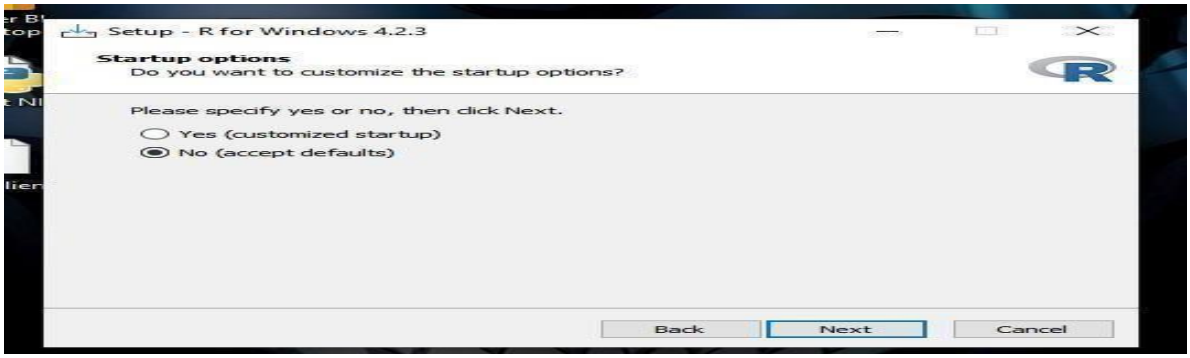
1) Select the path where we want to download the R and click Next



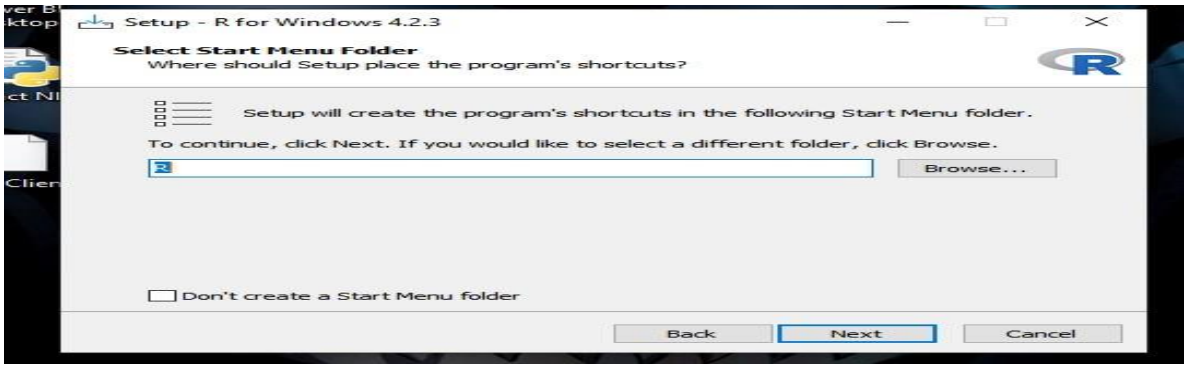
2) Select all components which we want to install, and then click Next.



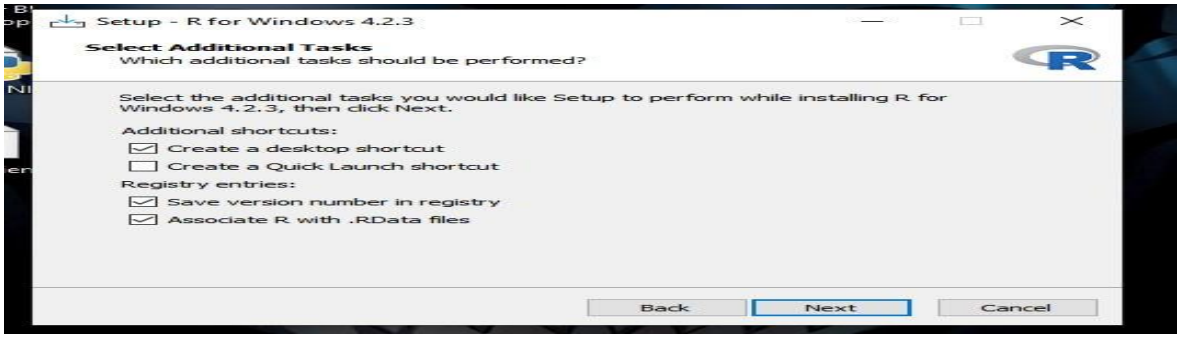
3) Now, we have to select either (customized startup) or (accept the default), and then click Next.



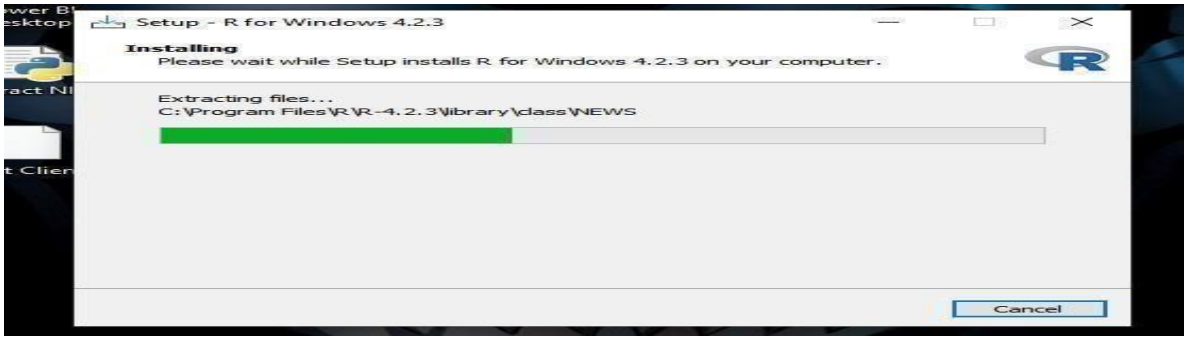
4) Now Select Start Menu Folder window will appear, click Next



5) Click Next



6) When we proceed to Next, installation of R will get started:



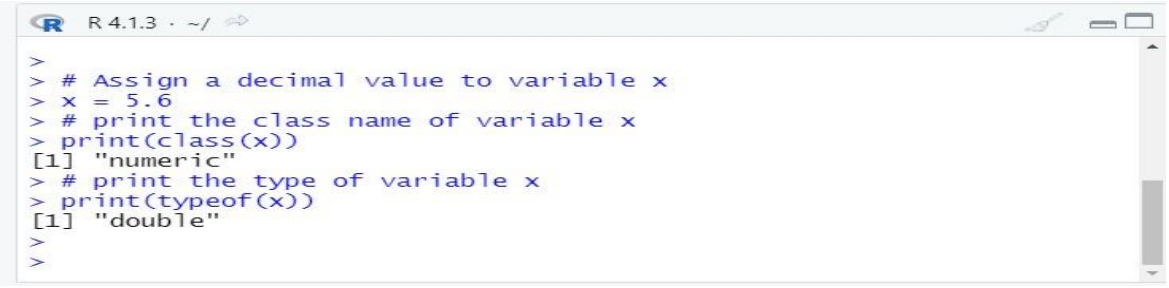
7) Finally, we will click on Finish.

R has been successfully installed.

Aim: Datatype in R programming

1. **R program to illustrate Numeric data** # Assign a decimal value to variable x

x = 5.6
# print the class name of variable x print(class(x))
# print the type of variable x print(typeof(x))

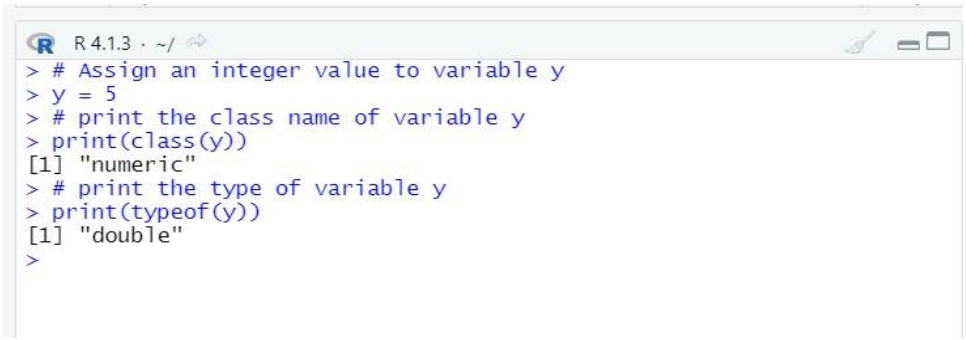


Output:

[1] "numeric"
[1] "double"

2. R program to illustrate Numeric datatype

# Assign an integer value to variable y
y = 5
# print the class name of variable y print(class(y))
# print the type of variable y print(typeof(y))

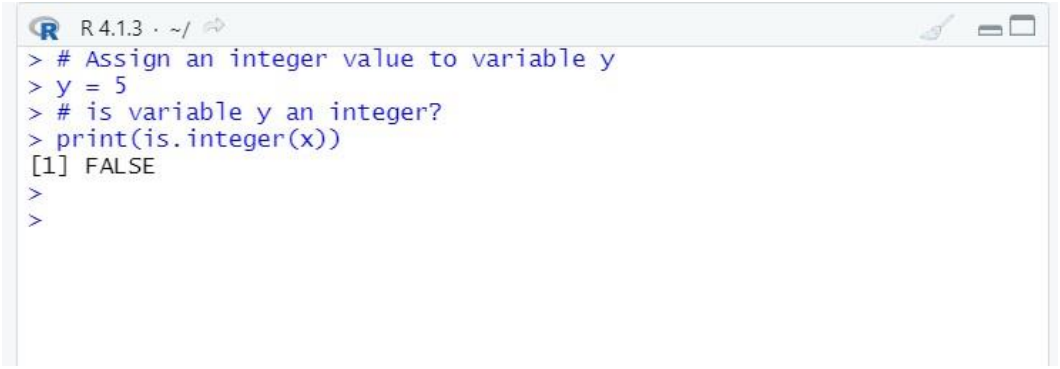


Output:

[1] "numeric"
[1] "double"

3. R program to illustrate Numeric datatype

Assign an integer value to variable y y = 5
is variable y an integer? print(is.integer(x))



Output:

[1] FALSE

3. R program to illustrate integer data type # Create an integer variable x = as.integer(5)

print the class name of variable x print(class(x))
print the type of variable x print(typeof(x))
Declare an integer by appending 'L' as suffix. y = 5L

# print the class name of y
print(class(y)) # print the type of y print(typeof(y))

```
R 4.1.3 · ~/
> # Create an integer variable
> x = as.integer(5)
> # print the class name of variable x
> print(class(x))
[1] "integer"
> # print the type of variable x
> print(typeof(x))
[1] "integer"
> # Declare an integer by appending 'L' as suffix.
> y = 5L
> # print the class name of y
> print(class(y))
[1] "integer"
> # print the type of y
> print(typeof(y))
[1] "integer"
>
>
```

Output:

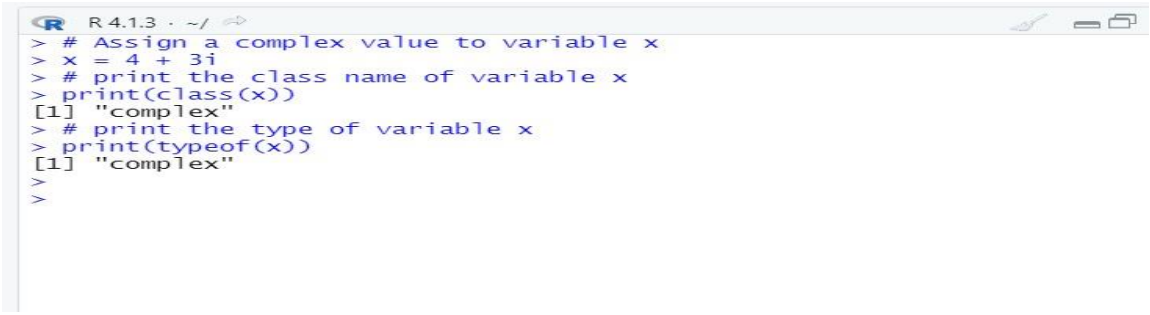
[1] "integer"
[1] "integer"
[1] "integer"
[1] "integer"

5. R program to illustrate logical data type # Two variables x = 4
y = 3
# Comparing two values z = x > y # print the logical value print(z)
# print the class name of z print(class(z))
# print the type of z
print(typeof(z))

```
R 4.1.3 · ~/
> # Two variables
> x = 4
> y = 3
> # Comparing two values
> z = x > y
> # print the logical value
> print(z)
[1] TRUE
> # print the class name of z
> print(class(z))
[1] "logical"
> # print the type of z
> print(typeof(z))
[1] "logical"
>
>
```

Output:
[1] TRUE
[1] "logical"
[1] "logical"

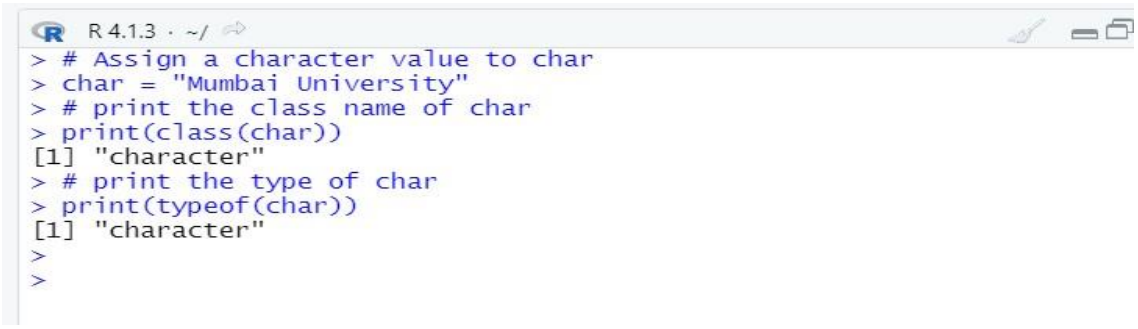
6. R program to illustrate complex datatype # Assign a complex value to variable x x = 4 + 3i
print the class name of variable x print(class(x))
print the type of variable x print(typeof(x))



Output:
[1] "complex"
[1] "complex"

7. R program to illustrate character data type

# Assign a character value to char char = "Mumbai University"
# print the class name of char
print(class(char))
# print the type of char
print(typeof(char))



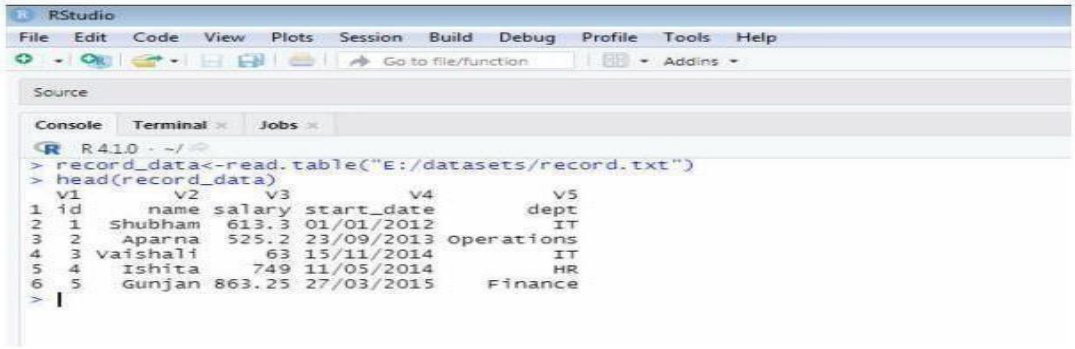
Output:
[1] "character"
[1] "character"

Aim: Reading and Writing data to and from R.

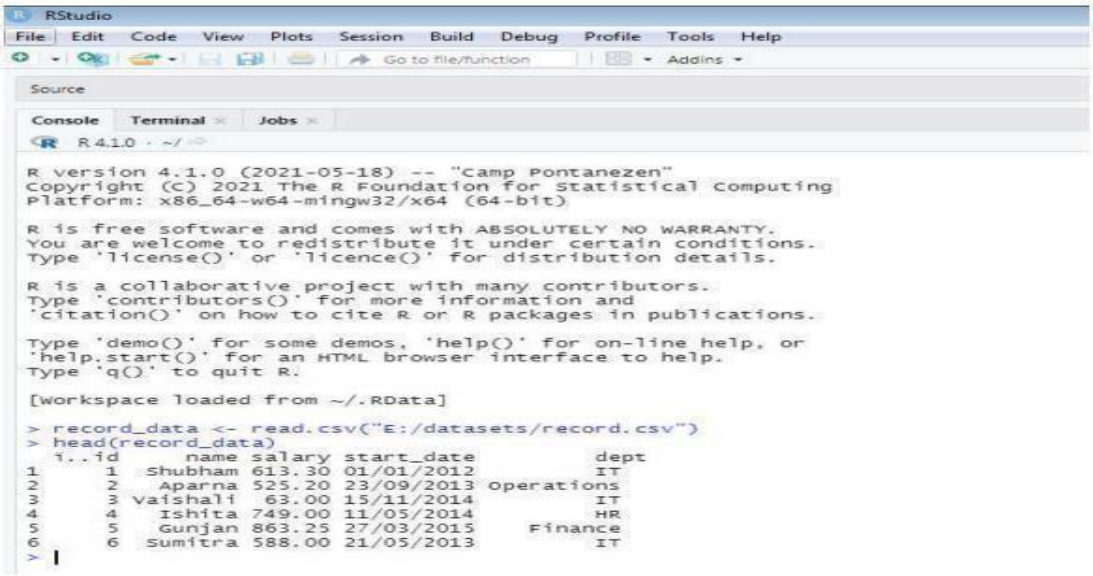
Reading data files with read.table()
The read.table() function is one of the most common used functions for readingdata into R. It has following arguments.
The function read.table() can be used to read the data frame.

We have kept record.txt and record.csv files under datasets folder inside E: drive.

>record_data<- read.table("E:/datasets/record.txt")
>head(record_data)
#returns first n rows of the data



Similarly, read.csv() function can be used to read data from csv files.
>record_data<- read.csv("E:/datasets/record.csv")
>head(record_data) #returns first n rows of thedata

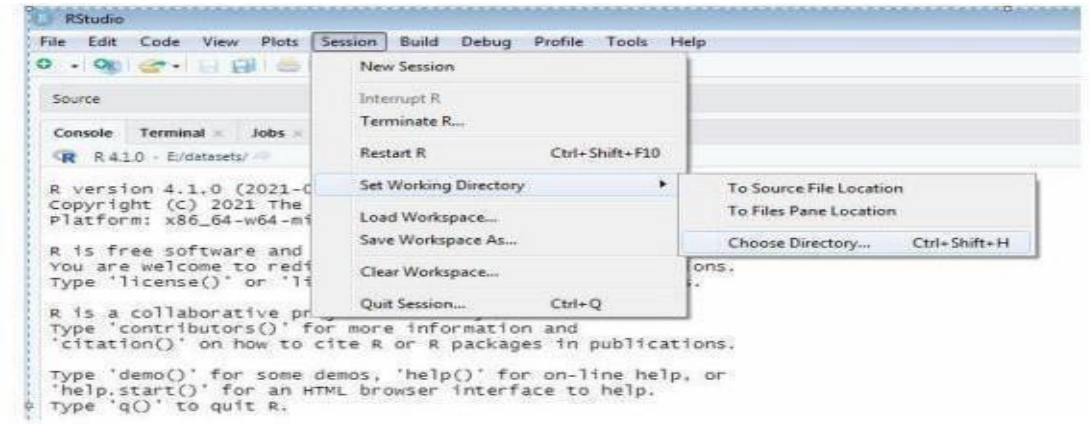


***Writing Data to a File**

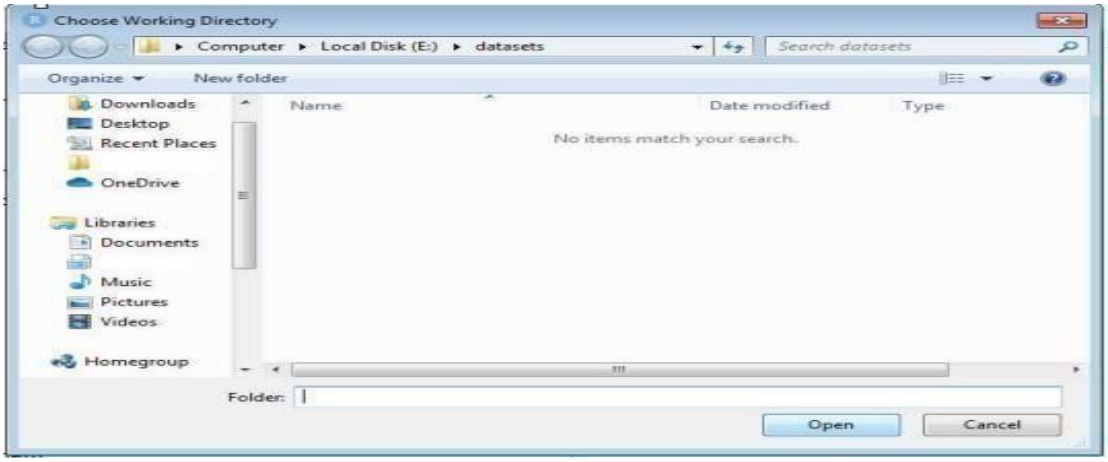
After working with a dataset, we might like to save it for future use. Before we do this, let's first set up a working directory so we know where we can find all our data sets and files later.

Setting up a Directory

From RStudio, use the menu to change your working directory under Session > Set Working Directory > Choose Directory

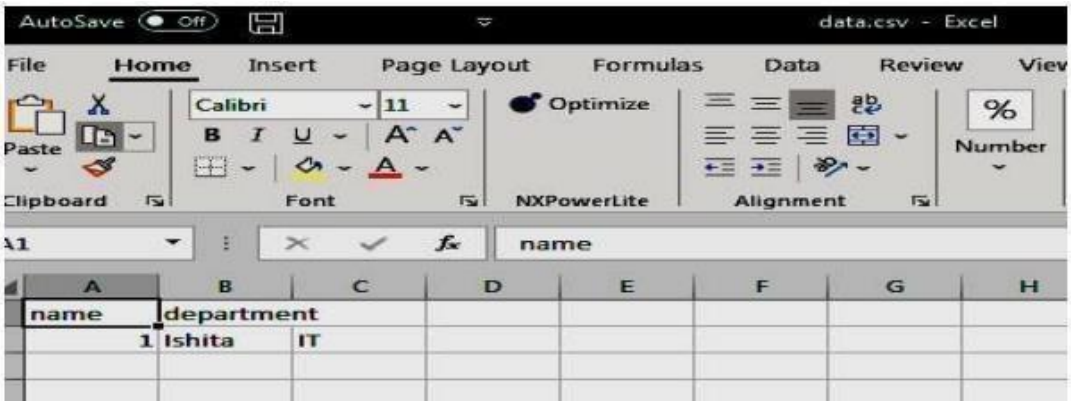


Click Open.



Alternatively, you can use the setwd() function to assign working directory.
> setwd("E:/datasets")
To check your current working directory, type
> getwd()
In R, we can write data easily to a file, using the write.table() command.
x<-data.frame(name ="Ishita", department = "IT") write.table(x, file ="data.csv", sep = ",")

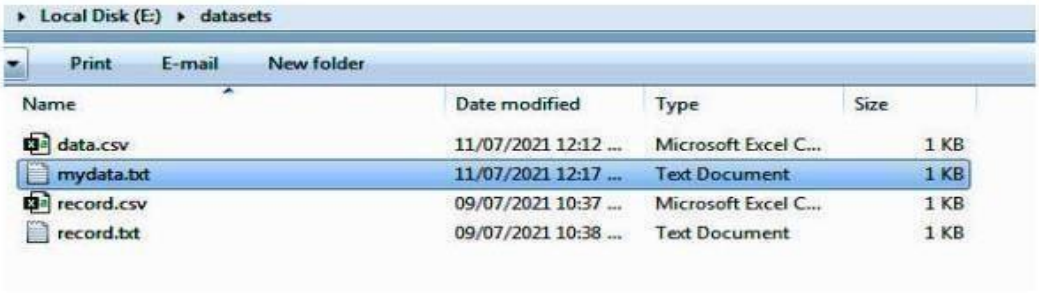
By going to this location E:/datasets,you should see a data.csv file.



y<-data.frame(name ="Ankit", department = "HR") write.table(y,"E:/datasets/mydata.txt", sep = "\t")

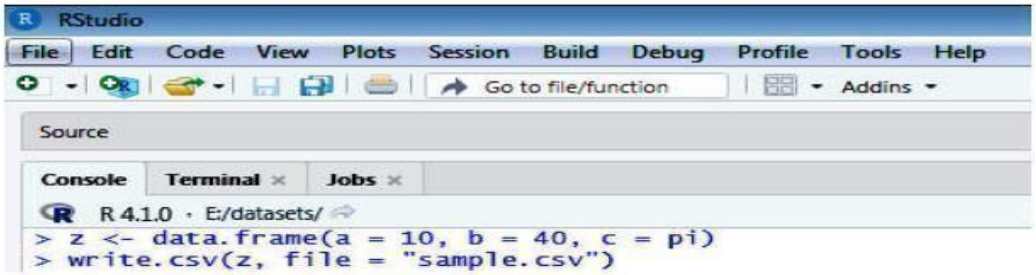
```
> y<-data.frame(name ="Ankit", department = "HR")
> write.table(y, "E:/datasets/mydata.txt", sep = "\t")
>
```

Now, let's check whether R created the file mydata.txt under E:/datasets folder or not.

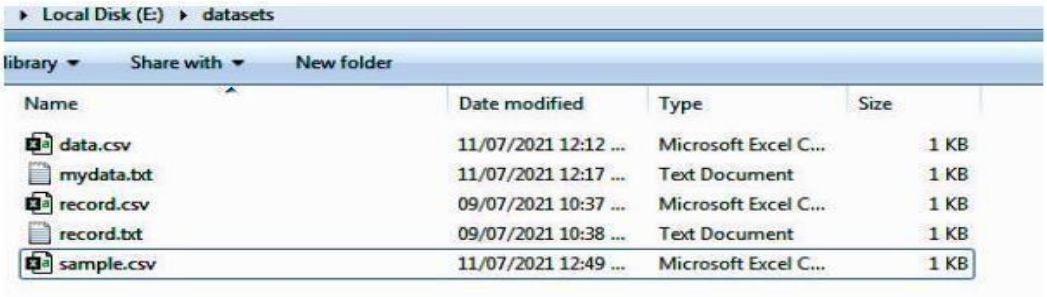


By going to this location E:/datasets, you should see a mydata.txt file.

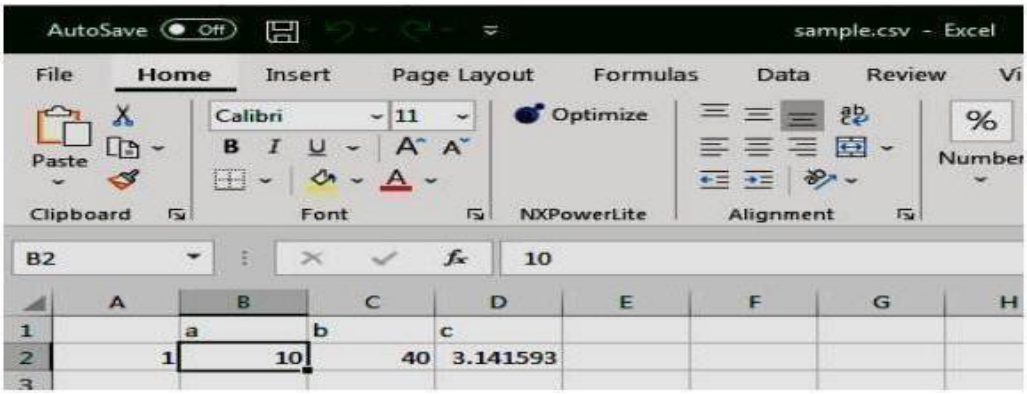
z <- data.frame(a = 10, b = 40, c = pi) write.csv(z, file = "sample.csv")



Now, let's check whether R created the file sample.csv under E:/datasets folder or not.



By going to this location E:/datasets,you should see a sample.csv file.



Experiment No: 06

Aim: To study Linear Regression

program: -

```
import numpy as np

class LinearRegression:
    def __init__(self):
        self.b0 = 0
        self.b1 = 1

    def fit(self, X, Y):
        X_mean = np.mean(X)
        Y_mean = np.mean(Y)
        numerator, denominator = 0, 0
        for _ in range(len(X)):
            numerator += (X[_] - X_mean) * (Y[_] - Y_mean)
            denominator += (X[_] - X_mean) ** 2
        self.b1 = numerator / denominator
        self.b0 = Y_mean - (X_mean * self.b1)
        return self.b0, self.b1

    def predict(self, X):
        y_hat = self.b0 + (self.b1 * X)
        return y_hat

if __name__ == '__main__':
    X = np.array([173, 160, 154, 188, 168], ndmin=2)
    X = X.reshape(5, 1)
    y = np.array([73, 65, 54, 80, 70])

    model = LinearRegression()
    b0, b1 = model.fit(X, y)
    print(b0, b1)

    y_pred = model.predict([165])
    print(y_pred)
```

Output:

```
dj@dheeraj MINGW64 ~/Desktop/mca-practical/python-practical (main)
$ python linear-regression.py
[-50.99209602] [0.70813817]
[65.85070258]
```

Experiment No: 07

Aim: To study Analysis of Regression
Program:

```
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
from sklearn.metrics import r2_score

if __name__ == '__main__':
    X = np.array([173, 160, 154, 188, 168], ndmin=2)
    X = X.reshape(5, 1)
    y = np.array([73, 65, 54, 80, 70])

    model = LinearRegression()
    model.fit(X, y)
    y_hat = model.predict(X)
    print(y_hat)

    mse = mean_squared_error(y_true=y, y_pred=y_hat)
    print(f'Loss Calculated : {mse}')

    r2 = r2_score(y_true=y, y_pred=y_hat)
    print(f'Goodness of Fit : {r2}')
```

Output:

```
dj@dheeraj MINGW64 ~/Desktop/mca-practical/python-practical (main)
$ python analysis-regression.py
[71.51580796 62.31001171 58.06118267 82.13788056 67.9751171 ]
Loss Calculated : 6.920550351288062
Goodness of Fit : 0.9082641788005293
```

Experiment No: 08

Aim: To study Logistic Regression
Program:

```
import numpy as np
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import log_loss, confusion_matrix, precision_score, recall_score, f1_score

if __name__ == '__main__':
    X = np.array([6, 2, 5, 9, 1], ndmin=2)
    X = X.reshape(5, 1)
    y = np.array([1, 0, 1, 1, 0])

    model = LogisticRegression()
    model.fit(X, y)
    y_hat = model.predict(X)
    print(y_hat)

    loss = log_loss(y_true=y, y_pred=y_hat)
    print(f'Logarithmic Loss: {loss}')

    cm = confusion_matrix(y_true=y, y_pred=y_hat)
    precision = precision_score(y_true=y, y_pred=y_hat)
    recall = recall_score(y_true=y, y_pred=y_hat)
    f1 = f1_score(y_true=y, y_pred=y_hat)

    print(f'Confusion Matrix:\n{cm}')
    print(f'Precision: {precision}')
    print(f'Recall: {recall}')
    print(f'F1-Score: {f1}')
```

Output:

```
dj@dheeraj MINGW64 ~/Desktop/mca-practical/python-practical (main)
$ python logistic-regression.py
[1 0 1 1 0]
Logarithmic Loss: 2.220446049250313e-16
Confusion Matrix:
[[2 0]
 [0 3]]
Precision: 1.0
Recall: 1.0
F1-Score: 1.0
```

Experiment No: 09

Aim: To study support Vector Machine
Program:

```
from sklearn.datasets import load_iris
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import precision_score, recall_score, f1_score

if __name__ == '__main__':
    iris = load_iris ()
    X = np.array (iris.data)
    y = np.array (iris.target)
    X_train, X_test, y_train, y_test = train_test_split (X, y, shuffle=True, test_size=0.1)

    model = SVC ()
    model.fit (X_train, y_train)
    y_hat = model.predict (X_test)

    precision = precision_score (y_true=y_test, y_pred=y_hat, average='micro')
    recall = recall_score (y_true=y_test, y_pred=y_hat, average='micro')
    f1 = f1_score (y_true=y_test, y_pred=y_hat, average='micro')

    print (f'Precision : {precision}')
    print (f'Recall : {recall}')
    print (f'F1-Score : {f1}')
```

Output:

```
dj@dheeraj MINGW64 ~/Desktop/mca-practical/python-practical (main)
$ python vector-machine.py
Precision : 0.9333333333333333
Recall : 0.9333333333333333
F1-Score : 0.9333333333333333
```

Experiment No: 10

Aim: To study varied Algorithm
Program:

```
from sklearn.datasets import load_iris
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB

if __name__ == '__main__':
    iris = load_iris ()
    X = np.array (iris.data)
    y = np.array (iris.target)
    X_train, X_test, y_train, y_test = train_test_split (X, y, shuffle=True, test_size=0.1)

    dt = DecisionTreeClassifier ()
    dt.fit (X_train, y_train)
    dt_pred = dt.predict ([[1, 2, 3, 4]])
    print (f'Decision Tree : {dt_pred}')

    rf = RandomForestClassifier ()
    rf.fit (X_train, y_train)
    rf_pred = rf.predict ([[1, 2, 3, 4]])
    print (f'Random Forest : {rf_pred}')

    knn = KNeighborsClassifier ()
    knn.fit (X_train, y_train)
    knn_pred = knn.predict ([[1, 2, 3, 4]])
    print (f'KNN : {knn_pred}')

    nb = GaussianNB ()
    nb.fit (X_train, y_train)
    nb_pred = nb.predict ([[1, 2, 3, 4]])
    print (f'Naive Bayes : {nb_pred}')
```

Output:

```
dj@dheeraj MINGW64 ~/Desktop/mca-practical/python-practical (main)
$ python varied-algorithm.py
Decision Tree : [2]
Random Forest : [2]
KNN : [1]
Naive Bayes : [2]
```