4 Importing/Exporting Data between CSV Files/My SQL and Pandas

Fastrack Revision

- **Introduction:** In this chapter, we shall learn how to transfer data to/from a .CSV file (.CSV is a format that stores data in common separated form – Comma Separated Values) from/into a DataFrame and also to/from a database table from/into a DataFrame.
- **Importing and Exporting Data between CSV Files and DataFrames:** We can create a DataFrame by importing data from CSV files where values are separated by commas. Similarly, we can also store or export data in a DataFrame as a .csv file.
- **Importing a CSV File to a DataFrame:** Let us assume that we have the following data in a csv file named ResultData.csv stored in the folder C:/Blueprint.

RollNo.	Name	Eco	Maths
1	Rohit	28	57
2	Aditi	23	45
3	Sanyam	31	47
4	Gagan	30	30
5	Deepak	28	49

- We can load the data from the ResultData.csv file into a DataFrame, say marks using Pandas read_csv() function as shown below:

```
>>> marks = pd.read_csv("C:/Blueprint/ResultData.csv", sep =",", header=0)
>>> marks
```

RollNo .	Name	Eco	Maths
1	Rohit	28	57
2	Aditi	23	45
3	Sanyam	31	47
4	Gagan	30	30
5	Deepak	28	49

  - The first parameter to the read_csv() is the name of the comma separated data file along with its path.
  - The parameter sep specifies whether the values are separated by comma, semicolon, tab or any other character. The default value for sep is a space.
  - The parameter header specifies the number of the row whose values are to be used as the column names. It also marks the start of the data to be fetched. Header=0 implies that column names are inferred from the first line of the file. By default, header=0.
- **Exporting a DataFrame to a CSV File:** We can use the to_csv() function to save a DataFrame to a text or csv file. For example, to save the DataFrame ResultDF created in the previous section; we can use the following statement:

```
>>> ResultDF
```

Sub	Gaurav	Rahul	Priya	Aksansha	Ayush
English	94	72	79	88	97
Science	71	91	81	77	96
Maths	67	76	98	76	95

```
>>> ResultDF.to_csv(path_or_buf='C:/Blueprint/resultout.csv', sep=',')
```

- This creates a file by the name resultout.csv in the folder C:/Blueprint on the hard disk. When we open this file in any text editor or a spreadsheet, we will find the above data along with the row labels and the column headers, separated by comma.
- **Import and Export of Data between Pandas and MySQL:** In real-world scenarios, we will be required to bring data directly from a database and load to a DataFrame. This is called **importing data from a database**. Likewise, after analysis, we will be required to store data back to a database. This is called **exporting data to a database**.
- Data from DataFrame can be read from and written to MySQL database. To do this, a connection is required with the MySQL database using the pymysql database driver. And for this, the driver should be installed in the Python environment using the following command:

```
pip install pymysql
```

- **Sqlalchemy** is a library used to interact with the MySQL database by providing the required credentials. This library can be installed using the following command:

```
pip install sqlalchemy
```

- **Sqlalchemy** provides a function create_engine() that enables this connection to be established. The string inside the function is known as **connection string**. The connection string is composed of multiple parameters like the name of the database with which we want to establish the connection, username, password, host, port number and finally the name of the database.

  Syntax: engine=create_engine('driver://username: password@host:port/name_of_database',index=false)
- **Importing Data from MySQL to Pandas:** Importing data from MySQL to pandas basically refers to the process of reading a table from MySQL database and loading it to a pandas DataFrame. After establishing the connection, in order to fetch data from the table of the database we have the following three functions:

- **Pandas.read_sql_query(query,sql_conn):** It is used to read an sql query (query) into a DataFrame using the connection identifier (sql_conn) returned from the create_engine ( ).
- **Pandas.read_sql_table(table_name,sql_conn):** It is used to read an sql table (table_name) into a DataFrame using the connection identifier (sql_conn).
- **Pandas.read_sql(sql, sql_conn):** It is used to read either an sql query or an sqltable (sql) into a DataFrame using the connection identifier (sql_conn).

- **Exporting Data from Pandas to MySQL:** Exporting data from Pandas to MySQL basically refers to the process of writing a pandas DataFrame to a table of MySQL database.

  Syntax: pandas.DataFrame.to_sql(table,sql_conn, if_exists="fail", index=False/True)

Practice Exercise

Multiple Choice Questions

Q 1. CSV stands for: *[CBSE SQP 2023-24]*

a. Column Separated Value
b. Class Separated Value
c. Common Separated Value
d. None of the above

Q 2. A CSV file can take ..... character as separator.

a. , b. |
c. \t d. All of these

Q 3. The correct statement to read from a CSV file in a DataFrame is:

a. <DF>.read_csv(<file>)
b. <File>.read_csv(<DF>)
c. <DF>pandas.read(<file>)
d. <DF>pandas.read_csv(<files>)

Q 4. Which argument do you specify with read_csv( ) to specify a separator character?

a. Character b. Char
c. Separator d. Sep

Q 5. To suppress first row as header, which of the following arguments is to be given in read_csv( )?

a. noheader = True b. header = None
c. skipheader = True d. header = Null

Q 6. To read specific number of rows from a CSV file, which argument is to be given in read_csv( )?

a. rows = <n> b. nrows = <n>
c. n_rows = <n> d. number_rows = <n>

Q 7. To skip first 5 rows of CSV file, which argument will you give in read_csv( )?

a. skiprows = 5 b. skip_rows = 5
c. skip = 5 d. noread = 5

Q 8. To skip 1st, 3rd and 5th row of CSV file, which argument will you give in read_csv( )?

a. skiprows = 1 | 3 | 5 b. skiprows = [1, 5, 1]
c. skiprows = [1, 3, 5] b. Any of these

Q 9. While reading from a CSV file, to use a column's values as index labels, argument given in read_CSV( ) is:

a. Index b. Index_col
c. Index_values d. Index_label

Q 10. While writing a DataFrame onto a CSV file, which argument would you use in to_sql( ) for NaN values' representation as NULL?

a. NaN = NULL b. na_rep = NULL
c. na = NULL d. na_value = NULL

Q 11. Which of the following libraries let you establish a connection with a MySQL database from within Python?

a. mysqlconnector b. pymysql SQLALchemy
c. numpy d. Either a. or b.

Q 12. In Pandas.read_sql (<A>, <B>), <A> is:

a. connection name b. table name
c. SQL query string d. database name

Q 13. In pandas.read_sql (<A>, <B>), <B> is:

a. connection name b. table name
c. SQL query string d. database name

Q 14. To suppress the creation of a column for index labels of a DataFrame, ... argument is specified in to_sql( ).

a. if_exists = False b. Index = False
c. Index = True d. if_exists = True

Q 15. To append the content of DataFrame in a table of MySQL, ... argument is used in to_sql( ).

a. if_exists = "Add" b. if_exists = "append"
c. if_exists = Add d. if_exists = append

Fill in the Blanks Type Questions

Q 16. Default separator of CSV files is .....

Q 17. The default value for parameter sep is a .....

Q 18. To load data of a CSV file in a DataFrame ..... function is used.

Q 19. To write data of a DataFrame in a CSV file, ... function is used.

Q 20. To specify a separator other than comma in a CSV file, ... argument is used.

Q 21. To specify the string to represent NaN values in a CSV file, ... argument in to-sql() is used.

Q 22. To load data in a DataFrame from mysql table, ... function is used.

Q 23. To write data of a DataFrame in a mysql table, ... function is used.

Assertion & Reason Type Questions

Directions (Q. Nos. 24-26): *In the questions given below, there are two statements marked as Assertion (A) and Reason (R). Read the statements and choose the correct option.*

a. Both Assertion (A) and Reason (R) are true and Reason (R) is the correct explanation of Assertion (A).
b. Both Assertion (A) and Reason (R) are true, but Reason (R) is not the correct explanation of Assertion (A).
c. Assertion (A) is true, but Reason (R) is false.
d. Assertion (A) is false, but Reason (R) is true.

Q 24. Assertion (A): The parameter header specifies the number of the row whose values are to be used as the row names.
Reason (R): Importing data from MySQL to pandas basically refers to the process of reading a table from MySQL database and loading it to a pandas DataFrame.

Q 25. Assertion (A): pandas.read_sql_query(query, sql_conn) is used to read and sql query (query) into a DataFrame using the connection identifier (sql_conn) returned from the create_engine( ).
Reason (R): pandas.read_sql(sql, sql_conn) is used to read either an sql query or an sqltable (sql) into a DataFrame using the connection identifiers (sql_conn).

Q 26. Assertion (A): sqlalchemy is a library used to interact with the MySQL database by providing the required credentials.
Reason (R): In realworld scenarios, we will be required to bring data directly from a database and load to a DataFrame.

Answers

1. (c)	2. (d)	3. (d)	4. (d)	5. (b)
6. (b)	7. (a)	8. (c)	9. (b)	10. (b)
11. (d)	12. (c)	13. (a)	14. (b)	15. (b)

16. (comma) 17. space
18. read_csv 19. to_csv
20. sep 21. na_rep
22. read_sql( ) 23. to_sql( )
24. (d) 25. (b) 26. (c)

Case Study Based Questions

Case Study 1

Exporting Data from Pandas to MySQL: Exporting data from Pandas to MySQL basically refers to the process of writing a Pandas DataFrame to a table of MySQL database.

Syntax: Pandas.DataFrame.to_sql(table, sql_conn,if_exists="fail", index=False/True)

Q 1. Table specifies the ..... of the table in which we want to create or append DataFrame values.
a. address b. name
c. index d. content

Q 2. The parameter if_exists specifies "the way data from the DataFrame should be entered in the .....".
a. table b. row
c. column d. record

Q 3. ..... is the default value that indicates a ValueError if the table already exists in the database.
a. True b. False c. Fail d. Pass

Q 4. ..... specifies that the contents of the DataFrame should be appended to the existing table and when updated the format must be the same (column name sequences).
a. Replace b. Append
c. Content d. None of these

Q 5. Index – By default index is ..... means DataFrame index will be copied to MySQL table. If....., then it will ignore the DataFrame indexing.
a. true, false b. false, true
c. pass, fail d. fail, pass

Answers

1. (b) 2. (a) 3. (c) 4. (b) 5. (a)

Case Study 2

The so-called CSV (Comma Separated Values) format is the most common import and export format for spreadsheets and databases. CSV format was used for many years prior to attempts to describe the format in a standardised way in RFC 4180. The lack of a well-defined standard means that subtle differences often exist in the data produces and consumed by different applications. These differences can make it annoying to process CSV files from multiple sources. Still, while the delimiters and quoting characters vary, the overall format is similar enough that it is possible to write a single module which can efficiently manipulate such data, hiding the details of reading and writing the data from the programmer.

The CSV module implements classes to read and write tabular data in CSV format. It allows programmers to say, "write this data in the format preferred by Excel," or "read data from this file which was generated by Excel," without knowing the precise details of the CSV format used by Excel. Programmers can also describe the CSV formats understood by other applications or define their own special-purpose CSV formats.

Q 1. Observe the following code and fill the blank in statement1:

```
import csv
with ..... as f: #statement1
r = csv. ..... (f) #statement2
for row in ..... : #statement3
print(.....) #statement4
```

a. open('data.csv')
b. f=open('data.csv')
c. Both a. and b. are correct
d. Both a. and b. are incorrect

Q 2. Observe the following code and fill the blank in statement2:

```
import csv
with ..... as f: #statement1
r = csv. ..... (f) #statement2
for row in ..... : #statement3
print(.....) #statement4
```

a. load( ) b. read( )
c. reader( ) d. readliness( )

Q 3. Observe the following code and fill the blank in statement3:

```
import csv
with ..... as f: #statement1
r = csv. ..... (f) #statement2
for row in ..... : #statement3
print(.....) #statement4
```

a. F b. r
c. r, f d. None of these

Q 4. Observe the following code and fill the blank in statement4:

```
import csv
with ..... as f: #statement1
r = csv. ..... (f) #statement2
for row in ..... : #statement3
print(.....) #statement4
```

a. r b. row
c. f d. csv

Q 5. Every record in a CSV file is stored in reader object in the form of a list using which method?

a. writer( ) b. append( )
c. reader( ) d. list( )

Answers

1. (a) 2. (c) 3. (b) 4. (b) 5. (c)

Case Study 3

Deepesh works as a programmer with Delta Technologies. He has been assigned the job of generating the salary of all employees using the file "employee.csv". He has written a program to read the CSV file "employee.csv" which will contain details of all the employees. He has written the following code. As a programmer, help him to successfully execute the given task.

```
import ..... # Line 1
def readcsvEmp( ): # to read data from the CSV file
with ..... ('employees.csv', newline='') as f: #  Line 2
reader=csv. ..... (f) # Line 3
data_list = ..... (reader) # Line 4
..... (data_list) # Line 5
```

Q 1. Name the module he should import in Line 1.
Q 2. Write the method that he should use to open the file to read data from it.
Q 3. Fill in the blank in Line 3 to read the data from a csv file.
Q 4. Fill in the blank in Line 4 with the method to convert the data read from the file into list.
Q 5. Define connection string.

Answers

1. import csv is the module he should import in line 1.
2. open method, he should use to open the file to read data from it.
3. reader
4. list
5. SQLALchemy provides a function create_engine() that enables this connection to be established. The string inside the function is known as connection string.

Very Short Answer Type Questions

Q 1. Define CSV File.

Ans. A Comma-Separated Value (CSV) file is a text file where values are separated by comma. Each line represents a record (row). Each row consists of one or more fields (columns).

Knowledge BOOSTER

CSV files can be easily handled through a spreadsheet application.

Q 2. What are advantages of CSV file formats?

Ans. Advantages of CSV file formats are as follows:

(i) CSV files are plain-text files, making them easier to create.
(ii) They're easier to import into a spreadsheet or another storage database.
(iii) To better organise large amounts of data.

Q 3. What all libraries do you require in order to bring data from a CSV file into a DataFrame?

Ans. Pandas Library is required.

Q 4. You want to read data from a CSV file in a DataFrame but you want to provide your own column names to DataFrame. What additional argument would you specify in read_csv( )?

Ans. name argument.
Example: pd.read_csv('file.csv',name=['name', 'city'])

Q 5. By default, read_csv() uses the values of first row as column headers in DataFrames. Which argument will you give to ensure that the top/first row's data is used as data and not as column headers?

Ans. Header argument will be given.
Example: pd.read_csv('file.csv', header = None)

Q 6. Which argument would you give to read.csv() if you only want to read top 10 rows of data?

Ans. nrows argument.

Example: pd.read_csv('file.csv', name = ['name', 'city'], nrows = 10)

Q 7. Write command to store data of DataFrame mdf into a CSV file Mydata.csv, with separator character as '@'.

Ans. Mdf.to_csv('mydata.csv', sep = '@')

Q 8. Write Python statement to export the DataFrame to a CSV file named data.csv stored at D: drive. *[CBSE SQP 2023-24]*

Ans. df . to_csv ("D:\data . csv")

Q 9. What is PyMySQL library of Python?

Ans. PyMySQL is an interface for connecting to a MySQL database server from Python.

Q 10. What all libraries do you require in order to interact with MySQL databases (and DataFrame) from within Python?

Ans. (i) Mysqlconnector (ii) Pymysql

(iii) Pandas

Q 11. What additional argument do you need to specify in to_sql( ) so that old data of MySQL table is retained?

Ans. If_exists argument.

Short Answer Type-I Questions

Q 1. Explain briefly the CSV format of storing files.

Ans. The acronym CSV is short for Comma-Separated Values, which refers to a tabular data saved as plaintext where data values are separated by commas.

In CSV format:

(i) Each row of the table is stored in one row i.e., the number of rows in a CSV file are equal to number of rows in the table (or sheet or database, table, etc.).

(ii) The field-values of a row are stored together with commas after every field value, but after the last field's value, no comma is given, just the end of line.

Q 2. If query is a string storing an SQL statement, write statements so that the data is fetched based on query from SQL database Mydata.

Ans. import mysqlconnector as a

db = a.connect(user = 'root', passwd = '0000000000', host = 'localhost', database = 'company')

import pandas as pd

df = pd.read_sql (f'select * from sales where sales> 50000', db)

print(df)s

Q 3. Are the following two statements same? Why/Why not?

(i) pd.read_csv('zoo.csv', sep = ',')

(ii) pd.read_csv('zoo.csv')

Ans. Yes, these two statements are same. These two statements are reading a csv file-first statement reading a csv file with a separator ',' comma and second statement reading a csv file with a default separator ',' comma.

Q 4. How are following two codes similar or different? What output will they produce?

(i) df = pd.read_csv("data.csv", nrows = 5)
print(df)

(ii) df = pd.read_csv("data.csv")
print(df)

Ans. First statement reads 5 rows of csv file but second statement reads whole csv file.

Q 5. What is the difference between following two statements?

(i) df.to_sql('houses', con = conn, if_exists = 'replace')

(ii) df.to_sql('houses', con = conn, if_exists = 'replace', index = False)

Ans. (i) (a) It converts a DataFrame to SQL table.
(b) It creates SQL table 'houses'.
(c) If 'houses' table exists, then this code statement replaces the table, this statement creates index column.

(ii) (a) It also converts a DataFrame to SQL table.
(b) It creates SQL table 'houses'.
(c) If 'houses' table exists, then this code statement replace the table, this statement does not create index column.

Q 6. Consider following code when conn is the name of established connection to MySQL database.

Cars = {'Brand': ['Alto', 'Zen', 'City', 'Kia'], 'Price': [22000, 25000, 27000, 35000]}
df = DataFrame (Cars, columns = ['Brand', 'Price'])
df.to_sql ('CARS', conn, if_exists = 'replace', index = False)

What will be the output of following query if executed on MySQL : SELECT * from CARS?

Ans. **Output:** select * from cars:

Brand	Price
Alto	22000
Zen	25000
City	27000
Kia	35000

Q 7. Consider following code when conn is the name of established connection to MySQL database.

sql SELECT from Sales where zone = "central"
df = pandas.read_sql(sql, conn)
df.head()

What will be stored in df?

Ans. This code fragment creates a DataFrame after reading SQL table 'sales'. Of contains only those records where zone = 'central

Q 8. Write a program to read data from a CSV file where separator character is '@'. Make sure that:

*** the top row is used as data, not as column headers.**
*** only 20 rows are read into DataFrame.**

Ans.
```
import pandas as pd
file='csv file.csv'
dframe=pd.read_csv( file.sep='@',header=
None,nrows=20)
print(dframe)
```

Q 9. The sales table of company database of MySQL stores the sales records of 100 salesmen. Write a program to load only those records in DataFrame which have made sales more than of ₹ 50,000/-.

Ans.
```
import mysql.connector as a
db=a.connect(user='root',passwd='0000000000',
host='localhost',database='company')
import pandas as pd
df=pd.read_sql(f'select * from sales where
sales>50000',db)
print(df)
```

Q 10. Following code is reading from employee.csv as shown here and intends to use column Empno's values as the index label. Why is the given code giving error? Suggest solution.

Employee.csv

Empno	Name	Designation	Salary
1001	Trupti	Manager	56000
1002	Raziya	Manager	55900
1003	Simran	Analyst	35000
1004	Silviya	Clerk	25000
1005	Suji	PR officer	31000

```
import pandas as pd
edf = pd.read_csv("Employee.csv", index_col =
"Empno.")
print(edf)
value Error: Index Empno. invalid
```

Ans. The above code is giving error because the column name is given as Empno. (a dot in the end) whereas the column name in the csv file is Empno (without dot), not Empno.

To correct this error, we just need to change the column name as **Empno** for **Index_col** argument, *i.e.*, the 2nd line of the code should be:

```
edf = pd.read_csv("Employee.csv", index_col =
"Empno")
```

Q 11. Write a program that reads from a CSV file where the separator character is '$'. Read only first 5 rows in your DataFrame.

- **Give column headings as ItemName, Quantity, Price.**
- **Make sure to read first row as data and not as column headers.**

Ans.
```
import pandas as pd
df = pd.read_csv("c:\\data\\data.csv", sep = "$", \
  names = ["ItemName", "Quantity", "Price"], header
  = None, nrows = 5)
print(df)
```

Short Answer Type-II Questions

Q 1. Predict the output of following code fragments one by one. For every next code fragment, consider that the changes by previous code fragment are in place. That is, for code fragment (b), changes made by code fragment (a) are persisting; for (c), changes by (a) and (b) are persisting and so on.

(i)
```
import pandas as pd
columns=['2015','2016','2017','2018']
index=['Messi','Ronaldo','Neymar','Hazard']
df = pd.DataFrame (columns = columns, index
= index)
print (df)
df.to_csv("c:\one.csv")
```

(ii)
```
df['2015']['Messi'] = 12
df['2016']['Ronaldo'] = 11
df['2017']['Neymar'] = 8
df['2018']['Hazard'] = 16
print (df)
df.to_csv("c:\two.csv", sep= '@')
```

(iii)
```
new_df = pd.read_csv('c:\one.csv',index_col
= 0)
print(new_df)
```

Ans. (i) Output:

	2015	2016	2017	2018
messi	NaN	NaN	NaN	NaN
ronaldo	NaN	NaN	NaN	NaN
neymar	NaN	NaN	NaN	NaN
hazard	NaN	NaN	NaN	NaN

This code fragment creates a DataFrame df and saving the DataFrame data into csv file.

(ii) Output:

	2015	2016	2017	2018
messi	12	NaN	NaN	NaN
ronaldo	NaN	11	NaN	NaN
neymar	NaN	NaN	8	NaN
hazard	NaN	NaN	NaN	16

This code fragment is inserting some values to DataFrame df and saving the DataFrame data into csv file with separator '@'.

(iii) Output: This code fragment creates in new DataFrame from csv file.

Q 2. Write a program to get following data in two DataFrames:

Df1

Roll No.	Name
1	ABC
2	DEF
:	:

Df2

Roll No.	Marks 1	Marks 2	Marks 3
1	70	80	75
2	60	65	70
:	:	:	:

Store these DataFrames as two separate tables in the same database.

Ans.
```
import pandas as pd
import pymysql
from sqlalchemy import create_engine
engine=create_engine('mysql+pymysql://
root:0000000000@localhost/pathwalla')
con=engine.connect()
d1={'Roll no':[1,2,3],'Name':['blue','print','Incredible']}
d2={'Roll no':[1,2,3],'Mark1':[12,15,18],'Mark2':[12,105,158],
'Mark3':[12,105,158]}
df=pd.DataFrame(d1)
df1=pd.DataFrame(d2)
df1.to_sql('new',con)
df.to_sql('new table',con)
```

Q 3. You have a database on MySQL namely school having three tables in it - Student, Subject, Teacher. Write a program to store these tables in three DataFrames.

Ans.
```
import mysql.connector as a
db=a.connect(user='root',passwd='0000000000',
host='localhost',database='school')
import pandas as pd
tables=['student','subject','teacher']
df=pd.read_sql(f'select * from {tables[0]}',db)
df1=pd.read_sql(f'select * from {tables[1]}',db)
df2=pd.read_sql(f'select * from {tables[2]}',db)
print(df)
print(df1)
print(df2)
```

Q 4. Following code is reading from file sport.csv. It displays the whole DataFrame but when it tries to print the Competitions column of the DataFrame, the code raises error.

Sport.csv

Sports	Competitions	Prizes won
Tennis	14	9
Football	22	16
Chess	25	15

```
import pandas as pd
sdf = pd.read_csv("sport.csv")
print(sdf)
print(sdf.Competitions)
```

Output generated is:

```
Sport\tcompetitions\tprizes won
0    Tennis\t14\t9
1    Football\t22\t16
2    Chess\t25\t15
```

AttributeError: 'DataFrame' object has no attribute 'competitions'

Why is above code giving this error? Suggest a solution.

Ans. The given file sport.csv has a separator as '\t', which is not specified while reading from it.

Thus, the code has read the entire row as one column and thus there is no separate column by the name **Competitions**.

To correct the above problem, all we need to do is to specify the separator as'\t'. That is, the 2nd line of the code should be:

```
sdf = pd.read_csv("sport.csv", sep = '\t')
```

Now, it will display the column **Competitions**' values as well.

COMMON ERROR

There may be confusion in finding error from the code so it must be read properly.

Q 5. DataFrame saleDf stores about 50 rows in it. Write a program to store its rows from 10 to 15 in a table random on MySQL database namely 'world'. Append the rows if the table already exists.

Ans.
```
import pandas as pd
from sqlalchemy import create_engine
import pymysql
engine = create_engine('mysql+pymysql ://
root:MyPass@localhost/world')
mycon = engine.connect( )
    : # statements to create or load saleDf
saleDf.iloc[10:15, : ].to_sql('random', mycon, index =
False,    if_exists = 'replace')
```

Long Answer Type Questions

Q 1. Write a program to read details such as item, sales made in a DataFrame and then store this data in a CSV file.

Ans.
```
import pandas as pd
n=int(input('enter no of item'))
dic={ }
values_sales=[ ]
values_name=[ ]
for i in range(n):
    item_n=input('enter name of item')
    sales=int(input('enter sales made of item'))
    values_sales.append(sales)
    values_name.append(item_n)
dic['Name']=values_name
dic['sales']=values_sales
dframe=pd.DataFrame(dic)
print(dframe)
dframe.to_csv('csv file',sep='|')
```

Q 2. The DataFrame SDF stores the sales records of 100 salesmen. Write a program to store this as a table in database namely "company" on MySQL.

Ans.
```
import pandas as pd
import pymysql
from sqlalchemy import create_engine
engine=create_engine('mysql+pymysql://root:
0000000000@localhost/company')
conn=engine.connect()
d1={'sales_record':[(1,2,3,4,5,6,7,8,9,10,11,) 12, 13, 14, 15,
16, 17, 18, 19, \
20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, \
```

```
33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, \
46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59,
60, 61, \
62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76,
77, 78, \
79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, \
93, 94, 95, 96, 97, 98, 99, 100])
SDF=pd.DataFrame(d1)
SDF.to_sql('sales_record',conn)
```

Q 3. Consider the SDF DataFrame storing the sales records of 100 salesmen, write a program that stores only the first 25 rows of the DataFrame in a table on MySQL database.

Ans.

```
import pandas as pd
import pymysql
from sqlalchemy import create_engine
engine=create_engine('mysql+pymysql://root:
0000000000@localhost/company')conn=engine.
connect()
d1={'sales_record':[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14,
15, 16, 17, 18, 19, \
20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, \
33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, \
46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59,
60, 61, \
62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76,
77, 78, \
79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, \
93, 94, 95, 96, 97, 98, 99, 100])
SDF=pd.DataFrame(d1)
sdf25only=SDF.head(25)
sdf25only.to_sql('sales_record_25',conn)
```

Q 4. Consider the following code and figure out what these are trying to do. The pandas library has been imported as pd.

(i) pd.read_csv("data.csv", nrows = 20)
(ii) pd.read_csv("data.csv", skiprows = [1, 2, 3, 4])
(iii) pd.read_csv("data.csv", header = None)
(iv) pd.read_csv("data.csv", sep = '@')
(v) pd.read_csv("data.csv", index_col = 'Ename')

Ans. (i) The given code is reading a csv file namely data.csv and it will read only the first 20 rows from it.

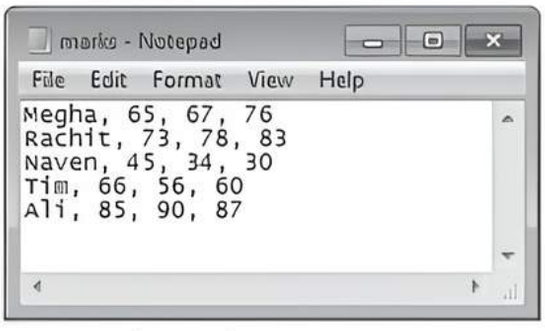
(ii) The given code is reading a csv file namely data.csv and it will skip the mentioned row numbers given in the list for **skiprows** argument.

(iii) The given code is reading a csv file namely data.csv and it will not read column header row as DataFrame column headings.

(iv) The given code is reading a csv file namely data.csv which has separator character as '@'.

(v) The given code is reading a csv file namely data.csv and will create DataFrame's index labels from column Ename's values.

Q 5. Write a program that reads from a csv file (marks.csv stored in data folder of C: drive having data as shown here: Name and marks in 3 subjects) in a DataFrame. Then the program should add a column 'Total' storing total of marks in three subjects and another column storing average marks. Print the DataFrame after adding these columns.



Ans.

```
import pandas as pd
df = pd.read_csv("c:\\data\\marks.csv",names =
("Name", "Marks1", "Marks2", "Marks3"))
print("DataFrame after fetching data from CSV file")
print(df)
df['Total'] = df ['Marks1']+df ['Marks2'] + df ['Marks3']
# 'Total' column
df ['AvgMarks'] = df ['Total'] / 3
print("DataFrame after all the calculations")
print(df)
```

Output:

DataFrame after fetching data from CSV file

	Name	Marks1	Marks2	Marks3
0	Megha	65	67	76
1	Rachit	73	78	83
2	Nevan	45	34	30
3	Tim	66	56	60
4	Ali	85	90	87

DataFrame after all the calculations

	Name	Marks1	Marks2	Marks3	Total	AvgMarks
0	Megha	65	67	76	208	69.333333
1	Rachit	73	78	83	234	78.000000
2	Nevan	45	34	30	109	36.333333
3	Tim	66	56	60	182	60.666667
4	Ali	85	90	87	262	87.333333

Q 6. The students table of test database of MySQL stores student details as (Roll no., Name, Marks, Grade, Section, Project). Write a program to load the data of this Student table in a DataFrame. Display the DataFrame and also display the details of the topper students (the students having the maximum marks).

Ans.

```
import pandas as pd
import mysql.connector as sqltor
mycon = sqltor.connect(host = "localhost",\ user =
"root", passwd = "MyPass", database = "test")
If mycon.is_connected( ):
    qry = "Select * from student;"
    sdf = pd. read__sql(qry, mycon)
    print("DataFrame contains:")
    print(sdf)
```

```
    #index of maximum marks
    topper_index = sdf.Marks.idxmax( )
    print("Topper student:" )
    print(sdf. iloc[topper_index, :])
else:
    print("MySQL Connection problem")
```

Output:

DataFrame contains:

	Roll No.	Name	Marks	Grade	Section	Project
0	101	Ruhani	76.8	A	A	Pending
1	102	George	71.2	B	A	Submitted
2	103	Simran	81.2	A	B	Evaluated
3	104	Ali	61.2	B	C	Assigned
4	105	Kushal	51.6	C	C	Evaluated
5	106	Arsiya	91.6	A+	B	Submitted
6	107	Raunaq	32.5	F	B	Submitted

Topper student:

Rollno	106
Name	Arsiya
Marks	91.6
Grade	A+
Section	B
Project	submitted

Chapter Test

Multiple Choice Questions

Q 1. To open a file c:\scores.csv for reading, we use ..... command.

a. infile = open("c:\scores.csv", "r")
b. infile = open("c:\\scores.csv", "r")
c. infile = open(file='c:\scores.csv', 'r')
d. infile = open(file='c:\\scores.csv', 'r')

Q 2. Which of the following statement(s) are true for csv files?

a. When you open a file for reading, if the file does not exist, an error occurs
b. When you open a file for writing, if the file does not exist, a new file is created
c. When you open a file for writing, if the file exists, the existing file is overwritten with the new file
d. All of the above

Q 3. To read the entire content of the CSV file as a nested list from a file object infile, we use ..... command.

a. infile.read() b. infile.reader()
c. csv.reader(infile) d. infile.readlines()

Q 4. EOL character used in windows operating system in CSV file is:

a. \r b. \n c. \r\n d. \0

Q 5. The CSV files are popular because they are:

a. Capable of storing large amount of data
b. Easier to create
c. Preferred export and import format for databases and spread sheets
d. All of the above

Fill in the Blanks

Q 6. The default delimiter character of CSV file is .....

Q 7. The valid mode to open CSV file are .....

Q 8. The file mode to open a CSV file for appending as well as reading is .....

Assertion & Reason Type Questions

Directions (Q. Nos. 9-10): *In the questions given below, there are two statements marked as Assertion (A) and Reason (R). Read the statements and choose the correct option.*

a. Both Assertion (A) and Reason (R) are true and Reason (R) is the correct explanation of Assertion (A).
b. Both Assertion (A) and Reason (R) are true, but Reason (R) is not the correct explanation of Assertion (A).
c. Assertion (A) is true, but Reason (R) is false.
d. Assertion (A) is false, but Reason (R) is true.

Q 9. Assertion (A): We can create a DataFrame by importing data from CSV files where values are separated by commas.
Reason (R): The first parameter to the read_csv() is the name of the comma separated data file along with its path.

Q 10. Assertion (A): The parameter sep specifies whether the values are separated by comma, semicolon, tab or any other character.
Reason (R): The default value for sep is a zero.

Case Study Based Question

Q 11. Legend sports wanted to store the number of prizes for each sport as a SPORTS.CSV file. As a programmer help them to complete the task successfully.

```
import ..... #Line 1
fh= ..... #Line 2
swriter = ..... (fh) #Line 3
ans="y"
i=1
while ans=="y"
print("Record",i)
sport=input("Sport name")
prizes=int(input("Enter prizes won"))
..... #Line 4
i=i+1
```

```
ans=input("Want to enter records")
fh. ..... #Line 5
```

(i) The module to be imported in Line 1.
   a. .tsv b. .csv c. .py d. .bin

(ii) Fill in line 2 to open the CSV file.
   a. fh=open("sports.csv", "w")
   b. fh=read("sports.csv", "w")
   c. fh=file("sports.csv", "w")
   d. fh=append("sports.csv", "w")

(iii) Write the correct statement to write the data into.file in line 3.
   a. writerrows( )
   b. writerow( )
   c. writer( )
   d. swriter=csv.csvwriter(fh)

(iv) Write the statement to write the records given as input from user in line 4.
   a. swriter([sport,prizes])
   b. swriter.writrrow([sport,prizes])
   c. swriter_writrrow([sport,prizes])
   d. swriterwritrrow([sport,prizes])

(v) To specify a different delimiter while writing into csv file, ... argument is used with csv.writer().
   a. delimit b. delimiter
   c. delimited d. delimits

Very Short Answer Type Questions

Q 12. What is the use of quoting parameter?

Q 13. Define csv.QUOTE_MINIMAL Parameter.

Q 14. Name some softwares in which CSV can be operated.

Short Answer Type-I Questions

Q 15. Write a program to read entire data from file data.csv.

Q 16. Write a program to copy the data from "data.csv" to temp.csv".

Short Answer Type-II Question

Q 17. Mohan has written following program to create a CSV file "File_extent.csv" which will contain fill types and file extensions for some records. As a programmer, help him to successfully execute the given task:

```
import ..... # Statement 1
def adddata(filetype,extension): #To write / add data into the file
    f=open (....., ....., newline="") # Statement 2
    newFileWriter = csv.writer(f)
    newFileWriter.writerow([filetype,extension])
    f.close()
#csv file reading code
def readdata(filename): # To read data
    with open(filename,'r') as f:
        filereader=csv. ..... (f) # Statement 3
        for row in filereader:
            print (row[0], row[1])
        f. ..... # Statement 4
adddata("Notepad", "txt")
adddata("Word", "docx")
adddata("Excel", "xlsx")
adddata("PowerPoint", "pptx")
readdata("....") # Statement5
```

(i) Which module he should import for Statement 1?

(ii) What is the correct option for Statement 2 to open file name and mode. (Suresh should open the file to add data into the file)?

(iii) What is the correct option for Statement 3 to read the data from a csv file?

Long Answer Type Question

Q 18. Sumit is making a software on "Countries and their Capitals" in which various records are to be stored/ retrieved in "CAPITAL.CSV" data file. It consists of few records of Countries and their Capitals. He has written the following code in Python. As a programmer, you have to help him to successfully execute the program.

```
import csv
    # Fn. to add a new record in CSV file
    def .....(Country, Capital): #Statement 1
    f=open("CAPITAL.CSV",".....") #Statement 2
    fwriter=csv.writer(f)
    fwriter.writerow([.....]) # Statement 3
    f.close()
def ShowRec(): #Fn. to display all records from CSV file
    with open("CAPITAL.CSV", "r") as NF:
        NewReader=csv. ..... (NF) # Statement 4
    for rec in NewReader:
        if len(rec)!=0:
            print(rec[0],rec[1])
AddNewRec("INDIA", "NEW DELHI")
AddNewRec("CHINA", "BEIJING")
ShowRec() # Statement 5
```

(i) What should be the Name of the function in Statement 1?

(ii) Which file mode to be passed to add new records in Statement 2?

(iii) What is the correct variables in Statement 3 to store data to the file?

(iv) What is the function for Statement 4 to read the data from a csv file?

(v) Write the correct output after the execution of Statement 5.