

LAST-HOUR CRAM SHEET · Informatics Practices

CBSE Class 12 Informatics Practices (065) · Theory
70 + Practical 30 · Read this in the 60 minutes
before you walk in

A. The Paper at a Glance

Unit	Name	Marks	Nature
1	Data Handling using Pandas + Data Visualization	25	Code writing + predict output
2	Database Query using SQL	25	Write queries + predict output
3	Introduction to Computer Networks	10	Theory + case study
4	Societal Impacts	10	Theory + case study

STRATEGY Units 1 and 2 are 50 of 70.

Both are *code*, which means they are fully learnable — there is no interpretation, only syntax. Units 3 and 4 are 20 marks of pure theory and are the easiest to finish quickly.

B. Every Import and Syntax Pattern

IMPORTS — write these first, always

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

SERIES

```
pd.Series([10,20,30])           # from
list
pd.Series(arr, index=['a','b','c']) # from
ndarray
pd.Series({'a':1,'b':2})         # keys
become index
pd.Series(5, index=[1,2,3])      #
scalar - index COMPULSORY

s.head(3)  s.tail(2)  s.size  s.count()  s.ndim
s['a':'c']  # LABEL slice - END INCLUDED
s[0:2]      # POSITION slice - END EXCLUDED
```

DATAFRAME

```

pd.DataFrame({'Name':[...], 'Marks':[...]})

df.shape      # (rows, cols)
df.size       # rows x cols
df.ndim       # always 2
df.T          # transpose

df.loc[0:2]    # LABEL - END INCLUDED
df.iloc[0:2]   # INTEGER - END EXCLUDED
df[df['Marks'] > 80]

df['New'] = values      # add
                        column
df.drop('col', axis=1, inplace=True)  #
                        delete column
df.drop(2, axis=0, inplace=True)      #
                        delete row
df.rename(columns={'a':'b'}, inplace=True)

pd.read_csv("f.csv")
df.to_csv("f.csv", index=False)

```

PLOTTING — the five-line answer

```

plt.plot(x, y, 'r--o', label='Sales')  # line
plt.bar(x, y)      plt.barh(x, y)      # bar
plt.hist(data, bins=5)                  #
histogram

plt.xlabel("X")
plt.ylabel("Y")
plt.title("Title")
plt.legend()
plt.savefig("c.png")  # BEFORE show()
plt.show()           # always LAST

```

SQL — the clause order never changes

```

SELECT  col, AGGREGATE(col)
FROM    table
WHERE   condition          -- rows, BEFORE
grouping
GROUP BY col
HAVING  condition          -- groups, AFTER
grouping
ORDER BY col DESC;         -- ASC is default

SELECT E.Name, D.DeptName          -- equi-join
FROM   Emp E, Dept D
WHERE  E.DeptID = D.DeptID;

```

C. Every SQL Function

Family	Functions and results
Math	POWER(2,3) → 8 · ROUND(15.678,2) → 15.68 · ROUND(4567,-2) → 4600 · MOD(10,3) → 1
Text	UCASE / UPPER · LCASE / LOWER · MID(s,3,4) (starts at 1) · LENGTH (counts spaces) · LEFT · RIGHT · INSTR (0 if absent) · LTRIM RTRIM TRIM
Date	NOW() · DATE() · YEAR() · MONTH() (number) · MONTHNAME() (word) · DAY() (number) · DAYNAME() (word)
Aggregate	MAX MIN AVG SUM COUNT — all ignore NULL. COUNT(*) counts all rows including NULL.

D. Rules You Must Not Get Wrong

MEMORISE EXACTLY

- Label slicing **INCLUDES** the end. Position slicing **EXCLUDES** it. Applies to Series `['a':'c']` and to `loc` vs `iloc`.
- Series + Series aligns on **INDEX**, not position. Unmatched labels give **NaN** and the dtype becomes **float64**.
- **axis=0** → **ROWS**. **axis=1** → **COLUMNS**.
`drop()` defaults to **axis=0**.
- Without `inplace=True` nothing changes — `drop` and `rename` return a copy.
- **Scalar Series** needs an index. Without it, error.
- `size` counts everything, `count()` skips **NaN**.
- **Attributes** have no brackets (`df.shape`); **methods** do (`df.head()`).
- `savefig()` **BEFORE** `show()`, or the file is blank.
- `legend()` needs `label=` in the plot call.
- **Bar graph** = categories with gaps. **Histogram** = continuous bins, bars touch.

- **WHERE** filters rows before grouping; **HAVING** filters groups after. Aggregates can only go in **HAVING**.
- **COUNT(*)** vs **COUNT(col)** — the second skips **NULL**.
- **SQL string positions start at 1, not 0.**
- **Never write `= NULL` — use `IS NULL`.**
- **Cartesian product** = $\text{rows}(A) \times \text{rows}(B)$
 $\text{rows, cols}(A) + \text{cols}(B)$ columns.
- **Hub broadcasts to all; Switch sends only to the target.**
- **Router joins similar networks; Gateway joins dissimilar ones.**
- **Mesh = most reliable. Bus = least cable. Star = easiest to troubleshoot.**
- **Freeware is NOT open source — free of cost, source code hidden.**
- **Plagiarism = no credit (ethical).
Copyright infringement = no permission (legal).**
- **IT Act 2000, amended 2008. 20-20-20 rule for eye strain.**

E. Master Task Triggers

"create a Series from a dictionary"	Keys become the index	1
"first 3 records"	<code>head(3)</code>	1
"how many non-null values"	<code>count()</code> not <code>size</code>	1
"delete the column permanently"	<code>drop(..., axis=1, inplace=True)</code>	2
"rows 1 to 3 by label"	<code>df.loc[1:3]</code>	2
"write to CSV without index"	<code>to_csv(..., index=False)</code>	2
"trend over 12 months"	Line plot	3
"compare 5 students"	Bar graph	3
"how many fall in each range"	Histogram	3
"department-wise average"	<code>GROUP BY + AVG()</code>	4
"departments with more than 5 staff"	HAVING <code>COUNT(*) > 5</code>	4
"names starting with A"	<code>LIKE 'A%'</code>	4
"second character is a"	<code>LIKE '_a%'</code>	4
"month name of joining"	<code>MONTHNAME(DOJ)</code>	4
"data from two tables"	Equi-join with matching condition	4
"computers in one lab"	LAN	5
"signal weakens over distance"	Repeater	5
"sends only to intended device"	Switch	5
"connects different protocols"	Gateway	5

"voice call over Internet"	VoIP	5
"content differs per user"	Dynamic web page	5
"website logged IP silently"	Passive digital footprint	6
"fake bank email asking for OTP"	Phishing	6
"copied without citing"	Plagiarism	6
"used copyrighted work with credit, no permission"	Copyright infringement	6
"protect a new invention"	Patent	6
"free but source code hidden"	Freeware	6
"eyes hurt after screen use"	20-20-20 rule	6

F. How to Write for Marks

Marks	What the examiner wants
1	One line: a full form, an output value, or a function name.
2	A short code snippet, or two points of difference in a table .
3–4	Complete code with the import line , or 3–4 SQL queries, or a network case study with reasons.
5	Full program or query set, plus labels/output as asked.
Output	Write the index column too for a Series or DataFrame. Preserve exact spacing where you can.
Case study	Name → define → what should be done . The last part is half the mark.

THE THREE CHECKS THAT SAVE MARKS — 1. Every Python answer starts with `import pandas as pd`. 2. Every

SQL query ends with a **semicolon** and follows

SELECT→FROM→WHERE→GROUP BY→HAVING→ORDER BY. **3.** Every plot has **xlabel, ylabel, title** and ends with `plt.show()`.

Chapter 1 · Pandas — Series

CBSE Class 12 Informatics Practices (065) · Unit 1

= 25 marks (largest unit) · Ultimate Revision Sheet

60-SECOND RECALL

Import	<code>import pandas as pd</code> · <code>import numpy as np</code>
Series	One-dimensional labelled array, holds any data type, size immutable
Default index	0, 1, 2 ... unless you pass <code>index=</code>
head() / tail()	First / last 5 rows by default
Label slicing	<code>s['a':'c']</code> → INCLUDES the end label
Position slicing	<code>s[0:3]</code> → EXCLUDES the end position

1. Pandas and NumPy — the Basics

DEFINITIONS Pandas (Panel Data) is an

open-source Python library for **data manipulation and analysis**. Its two main data structures are **Series** (1-D) and **DataFrame** (2-D).

NumPy (Numerical Python) provides the **ndarray**, a fast *n*-dimensional array that Pandas is built on.

Matplotlib is the library used for **plotting**.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

Basis	SERIES	DATAFRAME
Dimension	1-D	2-D
Structure	A single column with an index	Rows and columns, like a table
Data types	Homogeneous (one dtype)	Heterogeneous (different dtype per column)
Size	Size immutable	Size mutable
Value mutability	Both are value mutable — you can change the data	

SERIES — 1 dimension

index	values
a	10
b	20
c	30
d	40

one column of data + an index
.shape → (4,)

DATAFRAME — 2 dimensions

index	Name	Marks	City
0	Asha	91	Pune
1	Bala	78	Delhi
2	Chen	85	Kochi

rows + named columns
.shape → (3, 3)

Pull ONE column out of a DataFrame and what you get back is a Series.

df['Marks'] → Series · df[['Marks']] → still a DataFrame. One bracket or two decides the type.

A DataFrame is a dictionary of Series that all share one index.

2. Creating a Series

SYNTAX

```
pd.Series(data, index=None, dtype=None)
```

THE FOUR WAYS — every exam uses one of these 1.

From a LIST

```
s1 = pd.Series([10, 20, 30, 40])
```

```
OUTPUT 0    10  
1    20  
2    30  
3    40  
dtype: int64
```

2. From an ndarray, with a custom index

```
arr = np.array([1, 2, 3])  
s2 = pd.Series(arr, index=['a', 'b', 'c'])
```

```
OUTPUT a    1  
b    2  
c    3  
dtype: int32
```

3. From a DICTIONARY — keys become the index

```
d = {'Maths': 95, 'Science': 88, 'English': 72}  
s3 = pd.Series(d)
```

```
OUTPUT Maths    95  
Science    88  
English    72  
dtype: int64
```

4. From a SCALAR value — index is compulsory, value is repeated

```
s4 = pd.Series(5, index=[1, 2, 3, 4])
```

OUTPUT 1 5

2 5

3 5

4 5

dtype: int64

TRAP When creating from a *scalar*, the *index argument is mandatory* — without it Python raises an error, because Pandas cannot know how many times to repeat the value.

3. Series Attributes

Attribute	Returns
<code>s.index</code>	The index labels
<code>s.values</code>	The data as an ndarray
<code>s.dtype</code>	Data type of the elements
<code>s.size</code>	Number of elements
<code>s.shape</code>	Tuple, e.g. (4,)
<code>s.ndim</code>	1 — always, for a Series
<code>s.empty</code>	True if the Series has no element
<code>s.hasnans</code>	True if any value is NaN
<code>s.name</code>	Name given to the Series

TRAP *Attributes take no brackets* (`s.size`), but *methods do* (`s.head()`).
Writing `s.size()` is an error.

4. head(), tail() and Mathematical Operations

```
s = pd.Series([10, 20, 30, 40, 50, 60, 70])
```

```
s.head()      # first 5 by default  
s.head(3)     # first 3  
s.tail(2)     # last 2
```

OUTPUT of s.head(3)

```
0    10  
1    20  
2    30  
dtype: int64
```

VECTORISED OPERATIONS

```
s = pd.Series([10, 20, 30])  
s + 5      # adds 5 to every element  
s * 2      # multiplies every element  
s > 15     # returns a Boolean Series
```

Series + Series aligns on the INDEX, not position:

```
a = pd.Series([1, 2, 3], index=['x', 'y', 'z'])  
b = pd.Series([10, 20, 30], index=['y', 'z', 'w'])  
print(a + b)
```

OUTPUT

```
      w    NaN  
x     NaN  
y    12.0  
z    23.0  
dtype: float64
```

THE MOST-ASKED OUTPUT QUESTION

When two Series are added, Pandas **matches labels**. Any label present in one but not the other gives **NaN**, and the result becomes **float64** because NaN is a float. Expect this exact question.

5. Selection, Indexing and Slicing

THE CRITICAL DIFFERENCE

```
s = pd.Series([10, 20, 30, 40], index=['a', 'b', 'c', 'd'])
```

```
s['b']           # 20 - single label
```

```
s[1]            # 20 - positional
```

```
s[['a', 'c']]   # multiple labels
```

```
s['a':'c']      # LABEL slice - INCLUDES 'c'
```

```
s[0:2]         # POSITION slice - EXCLUDES
```

```
index 2
```

s['a':'c'] gives THREE items

a 10

b 20

c 30

s[0:2] gives TWO items

a 10

b 20

REMEMBER THIS ONE LINE *Label-based slicing INCLUDES the end. Position-based slicing EXCLUDES the end.*

This single rule is worth a mark almost every year.

```
s['b'] = 25          # modify a value
s[s > 20]           # Boolean filtering
```

6. Common Series Methods

Method	What it does
s.head(n) / s.tail(n)	First / last n elements
s.count()	Number of non-null values
s.sum() s.mean() s.max() s.min()	Aggregations, skipping NaN
s.sort_values()	Sort by value
s.sort_index()	Sort by index
s.drop('b')	Remove the element with that label
s.rename()	Change the index labels or the Series name

TRAP `count()` excludes NaN, but `size` includes it. For a Series with one NaN among four values, `size` is 4 and `count()` is 3.

7. Predict the Output

the highest-frequency question type

```
Q1 s = pd.Series([1,2,3,4], index=
['a','b','c','d'])
print(s['a':'c'])
```

```
Q2 s = pd.Series([1,2,3,4])
print(s[1:3])
```

```
Q3 a = pd.Series([1,2], index=['x','y'])
b = pd.Series([10,20], index=['y','z'])
print(a+b)
```

```
Q4 s = pd.Series(5, index=[1,2,3])
print(s.size, s.ndim)
```

```
Q5 s = pd.Series([10,None,30])
print(s.size, s.count())
```

8. Task Triggers wording → which method?

If the question says...	Use
"display the first 3 records"	<code>s.head(3)</code>
"display the last 2 records"	<code>s.tail(2)</code>
"create a Series from a dictionary"	Keys become the index

"create a Series with the same value repeated"	Scalar + index is compulsory
"how many non-null values?"	<code>s.count()</code>
"how many elements in total?"	<code>s.size</code>
"arrange in ascending order of marks"	<code>s.sort_values()</code>
"arrange alphabetically by name"	<code>s.sort_index()</code>
"show only values greater than 50"	<code>s[s > 50]</code>
"remove the entry for 'b'"	<code>s.drop('b')</code>
"check if the Series has any missing value"	<code>s.hasnans</code>

MOST-ASKED BOARD QUESTIONS

1. **[2-3M]** Write code to create a Series from a list / dictionary / ndarray with a given index.
2. **[2M]** Predict the output of the given Series code.
3. **[2M]** Differentiate between a Series and a DataFrame.
4. **[1M]** Difference between `size` and `count()`.
5. **[1M]** What does `s['a':'c']` return? → Includes 'c'

THE HABIT THAT SCORES — In code answers, always write `import pandas as pd` on the first line. Examiners deduct for a missing import even when the rest is perfect. And when asked to predict output, **write the index column too** — the output of a Series is two columns, not one.

Chapter 2 · Pandas — DataFrame and Import/Export

CBSE Class 12 Informatics Practices (065) · Unit 1

= 25 marks · Ultimate Revision Sheet

60-SECOND RECALL

DataFrame	2-D labelled structure — rows and columns, heterogeneous , size mutable
loc vs iloc	loc = LABEL based, end INCLUDED · iloc = INTEGER based, end EXCLUDED
Add column	<code>df['new'] = values</code>
Delete column	<code>df.drop('col', axis=1)</code> or <code>del df['col']</code>
Delete row	<code>df.drop(index, axis=0)</code>
axis rule	axis=0 → ROWS · axis=1 → COLUMNS

1. Creating a DataFrame

SYNTAX

```
pd.DataFrame(data, index=None, columns=None)
```

THE FOUR WAYS

1. From a dictionary of **LISTS** — keys become column names

```
d = {'Name': ['Amit', 'Bina', 'Chetan'],  
     'Marks': [85, 92, 78]}  
df = pd.DataFrame(d)
```

OUTPUT

	Name	Marks
0	Amit	85
1	Bina	92
2	Chetan	78

2. From a LIST OF DICTIONARIES — each dict is a row; missing keys become NaN

```
data = [{'a': 1, 'b': 2}, {'a': 5, 'b': 10, 'c': 20}]  
df = pd.DataFrame(data)
```

OUTPUT

	a	b	c
0	1	2	NaN
1	5	10	20.0

3. From a dictionary of SERIES — indexes are aligned

```
d = {'one': pd.Series([1,2,3], index= ['a', 'b', 'c']),  
     'two': pd.Series([4,5,6,7], index= ['a', 'b', 'c', 'd'])}  
df = pd.DataFrame(d)
```

OUTPUT

	one	two
a	1.0	4
b	2.0	5
c	3.0	6
d	NaN	7

4. From a 2-D ndarray / list of lists

```
df = pd.DataFrame([[1,2],[3,4]], columns= ['A', 'B'])
```

2. DataFrame Attributes

Attribute	Returns
<code>df.index</code>	Row labels
<code>df.columns</code>	Column labels
<code>df.values</code>	Data as a 2-D ndarray
<code>df.dtypes</code>	Data type of each column
<code>df.shape</code>	Tuple (rows, columns)
<code>df.size</code>	rows × columns — total number of cells
<code>df.ndim</code>	2 — always, for a DataFrame
<code>df.T</code>	Transpose — rows become columns
<code>df.empty</code>	True if the DataFrame has no element

TRAP For a 3×4 DataFrame: `shape` is (3, 4), `size` is 12, `ndim` is 2. Students often give `size` as the number of rows.

3. Selecting Data — `loc` and `iloc`

One DataFrame, two ways to point at the same rows.

posn / label

0 0
1 1
2 2

Name	Marks
Asha	91
Bala	78
Chen	85

`df.loc[0:2]`

→ 3 rows — end INCLUDED
counts by the LABEL written in the index

`df.iloc[0:2]`

→ 2 rows — end EXCLUDED
counts by POSITION, like a normal Python slice

Same numbers, different answers. `loc` includes the end, `iloc` does not.

The `i` in `iloc` stands for integer position — and Python slices never include their end.

With a default 0,1,2 index the label and the position match — which is why the trap works.

If the index were 'a','b','c', `loc` would need 'a':'c' while `iloc` would still use 0:3.

THE MOST IMPORTANT DISTINCTION IN THE UNIT

Basis	<code>loc[]</code>	<code>iloc[]</code>
Based on	LABEL (name)	INTEGER position
End of slice	INCLUDED	EXCLUDED
Example	<code>df.loc['a':'c']</code> → a, b, c	<code>df.iloc[0:3]</code> → positions 0, 1, 2

```

df['Marks']           # one column, as a
Series
df[['Name', 'Marks']] # multiple columns, as a
DataFrame

df.loc[1]             # row with label 1
df.loc[0:2]           # rows 0,1,2 - END
INCLUDED
df.iloc[0:2]          # rows 0,1 - END
EXCLUDED

df.loc[0:2, 'Name']   # rows and a column
together
df.iloc[0:2, 0:2]     # by position

df[df['Marks'] > 80]   # Boolean filtering

```

**Given df with 3 rows: `df.loc[0:2]` gives 3 rows,
`df.iloc[0:2]` gives 2 rows**

4. Adding, Modifying and Deleting

COLUMNS

```
df['Grade'] = ['A', 'A+', 'B']          # ADD a new
column
df['Marks'] = df['Marks'] + 5           # MODIFY an
existing column

df.drop('Grade', axis=1)                # DELETE -
returns a copy
df.drop('Grade', axis=1, inplace=True)  #
DELETE permanently
del df['Grade']                         # DELETE
permanently

df.rename(columns={'Marks': 'Score'},
inplace=True)
```

ROWS

```
df.loc[3] = ['Deepa', 88]              # ADD a row
by label
df.drop(1, axis=0, inplace=True)        # DELETE
row with label 1
df.rename(index={0: 'first'}, inplace=True)
```

THE TWO TRAPS 1. **axis:** `axis=0` means **rows**,

`axis=1` means **columns**. The default for `drop()` is **`axis=0`**, so dropping a column requires `axis=1`.

2. **inplace:** without `inplace=True`, `drop()` and `rename()` return a **new** DataFrame and the original is **unchanged**. This is the most common silent mistake.

5. Iteration

```
for col in df:                                # iterates
over COLUMN NAMES
    print(col)

for index, row in df.iterrows():              # row by row
    print(index, row['Name'], row['Marks'])

for label, series in df.iteritems():          # column by
column
    print(label, series.values)
```

TRAP `iterrows()` gives (index, row) where the row is a Series.

`iteritems()` gives (column name, column) where the column is a Series.

Confusing the two is a common output-prediction error.

6. Head, Tail and Descriptive Functions

```
df.head()          # first 5 rows
df.tail(3)         # last 3 rows
df.info()          # summary: dtypes, non-null
counts
df.describe()      # count, mean, std, min, 25%,
50%, 75%, max
```

Function	Returns
<code>df.max()</code> <code>df.min()</code>	Column-wise maximum / minimum
<code>df.sum()</code> <code>df.mean()</code>	Column-wise total / average
<code>df.count()</code>	Non-null count per column

<code>df.median()</code> <code>df.mode()</code>	Median / most frequent value
<code>df.quantile()</code>	Quantile values
<code>df.var()</code> <code>df.std()</code>	Variance / standard deviation

Add `axis=1` to any of these to compute **row-wise** instead of column-wise.

7. Importing and Exporting CSV

THE TWO COMMANDS — guaranteed in the paper

Import (read from CSV into a DataFrame):

```
df = pd.read_csv("marks.csv")

df = pd.read_csv("marks.csv", sep=";",
                 header=0,
                 names=['Name', 'Marks'],
                 index_col='Name')
```

Export (write a DataFrame to CSV):

```
df.to_csv("output.csv")

df.to_csv("output.csv", index=False,
         header=True)
```

Parameter	Meaning
<code>sep</code>	The separator; default is a comma
<code>header=0</code>	Use row 0 as column names. <code>header=None</code> means the file has no header row.
<code>names=[...]</code>	Supply your own column names

<code>index_col</code>	Which column to use as the row index
<code>index=False</code>	In <code>to_csv</code> , do not write the index column into the file

TRAP In `to_csv()` the default is `index=True`, so an extra unnamed column of 0,1,2 appears in the file. If the question says “without the index”, you **must** write `index=False`.

8. Predict the Output

highest-frequency question type

Q1 `df = pd.DataFrame({'A':[1,2,3],'B':[4,5,6]})`
`print(df.shape, df.size, df.ndim)`

Q2 `print(df.loc[0:1])` # 2 rows (labels 0 and 1)
`print(df.iloc[0:1])` # 1 row (position 0 only)

Q3 `data = [{'a':1,'b':2},{'a':5,'b':10,'c':20}]`
`df = pd.DataFrame(data)`
`print(df)`

Q4 `df.drop('B', axis=1)`
`print(df.columns)`

9. Task Triggers

wording → which method?

If the question says...

Use

"add a new column Total"	<code>df['Total'] = ...</code>
"delete the column Grade permanently"	<code>df.drop('Grade', axis=1, inplace=True)</code>
"delete the row with index 2"	<code>df.drop(2, axis=0, inplace=True)</code>
"display rows 1 to 3 by label"	<code>df.loc[1:3]</code> — end included
"display the first two rows by position"	<code>df.iloc[0:2]</code>
"show students scoring above 80"	<code>df[df['Marks'] > 80]</code>
"rename the column Marks to Score"	<code>df.rename(columns={...}, inplace=True)</code>
"display a statistical summary"	<code>df.describe()</code>
"show the data types of columns"	<code>df.dtypes</code> or <code>df.info()</code>
"read data from a CSV file"	<code>pd.read_csv("file.csv")</code>
"write to CSV without the index"	<code>df.to_csv("f.csv", index=False)</code>
"interchange rows and columns"	<code>df.T</code>
"total number of cells"	<code>df.size</code>

MOST-ASKED BOARD QUESTIONS

1. **[3-4M]** Create a DataFrame from the given data and perform the stated operations.
2. **[2-3M]** Predict the output of the given DataFrame code.

3. [2M] Differentiate between `loc` and `iloc`.
4. [2M] Write code to import from / export to a CSV file.
5. [1M] What does `axis=1` mean? →
Columns
6. [1M] Difference between `shape` and `size`.

THE HABIT THAT SCORES — Whenever a question says “permanently” or “from the DataFrame”, add `inplace=True`. Whenever it involves a column, add `axis=1`. Those two keywords account for most of the lost marks in this chapter.

Chapter 3 · Data Visualization with Matplotlib

CBSE Class 12 Informatics Practices (065) · Unit 1

= 25 marks · Ultimate Revision Sheet

60-SECOND RECALL

Import	<code>import matplotlib.pyplot as plt</code>
Only 3 plots prescribed	Line plot · Bar graph · Histogram
The three commands	<code>plt.plot()</code> · <code>plt.bar()</code> · <code>plt.hist()</code>
Customising	<code>xlabel()</code> <code>ylabel()</code> <code>title()</code> <code>legend()</code>
Must end with	<code>plt.show()</code>
Save	<code>plt.savefig("chart.png")</code> — before <code>show()</code>

1. Purpose of Plotting

WHY VISUALISE

Data visualisation presents data in a **graphical form** so that patterns, trends and outliers become immediately visible. Benefits: easier and faster **comparison**, quick identification of **trends**, better **decision making**, and clearer **communication** of results than raw numbers allow.

```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
```

2. Line Plot

WHEN TO USE A line plot shows how a value changes over a continuous range — typically over time. Use it for trends.

```
x = [1, 2, 3, 4, 5]
y = [10, 25, 18, 40, 35]

plt.plot(x, y)
plt.xlabel("Month")
plt.ylabel("Sales")
plt.title("Monthly Sales")
plt.show()
```

CUSTOMISING A LINE

```
plt.plot(x, y,
         color='red',           # or 'r',
         '#FF0000'
         linewidth=2,
         linestyle='dashed', # 'solid' 'dotted'
         'dashdot'
         marker='o',           # point marker
         markersize=8,
         label='Sales 2026')

plt.legend()                  # shows the label
plt.grid(True)
```

Shorthand: `plt.plot(x, y, 'r--o')` means red, dashed, circle markers.

Marker	Symbol	Linestyle	Symbol
Circle	'o'	Solid	'-'
Point	'.'	Dashed	'--'
Star	'*'	Dotted	'::'
Plus	'+'	Dash-dot	'-.'
Cross	'x'	Colours: r g b c m y k w	

3. Bar Graph

WHEN TO USE A bar graph compares values across discrete categories. Each bar's height (or length) represents the value.

```
subjects = ['IP', 'Maths', 'English', 'Eco']
marks = [92, 78, 85, 70]
```

```
plt.bar(subjects, marks, color='skyblue',
width=0.5)
plt.xlabel("Subject")
plt.ylabel("Marks")
plt.title("Marks by Subject")
plt.show()
```

VARIATIONS

```
plt.barh(subjects, marks)           # HORIZONTAL
bar graph

# Multiple bars side by side
x = np.arange(len(subjects))
plt.bar(x - 0.2, marks1, width=0.4, label='Term
1')
plt.bar(x + 0.2, marks2, width=0.4, label='Term
2')
plt.xticks(x, subjects)
plt.legend()
plt.show()
```

TRAP `plt.bar()` gives *vertical* bars;

`plt.barh()` gives *horizontal* bars. Note also that `width` controls bar thickness in `bar()`, but the equivalent in `barh()` is `height`.

4. Histogram

WHEN TO USE A histogram shows the frequency distribution of continuous data by grouping values into *bins* (intervals). It answers “how many values fall in each range?”

```
data = [12, 15, 18, 22, 25, 28, 30, 33, 35, 38, 41, 45]
```

```
plt.hist(data, bins=5, color='green',  
edgecolor='black')  
plt.xlabel("Marks")  
plt.ylabel("Number of Students")  
plt.title("Marks Distribution")  
plt.show()
```

USEFUL PARAMETERS

```
plt.hist(data,  
         bins=[0,20,40,60,80,100], # explicit  
         bin edges  
         color='green',  
         edgecolor='black',  
         cumulative=False,  
         histtype='bar') # 'step'  
'stepfilled'
```

BAR GRAPH vs HISTOGRAM — asked almost every year

Basis	BAR GRAPH	HISTOGRAM
Data type	Categorical / discrete	Continuous / numerical
X-axis	Categories (names)	Bins (ranges)
Gaps between bars	YES — bars are separated	NO — bars touch each other
Bars can be reordered	Yes — order is arbitrary	No — bins follow numeric order
Shows	Comparison between categories	Frequency distribution

5. Customising Any Plot

THE STANDARD BLOCK — write all of these

```
plt.xlabel("X axis label")
plt.ylabel("Y axis label")
plt.title("Chart Title")
plt.legend() # needs
label= in the plot call
plt.grid(True)
plt.xticks([1,2,3], ['A','B','C']) # custom
tick labels
plt.xlim(0, 10) # axis range
plt.ylim(0, 100)

plt.savefig("chart.png") # SAVE
before show()
plt.show() # DISPLAY -
always last
```

THE TWO ORDER TRAPS 1. `plt.savefig()` must come **BEFORE** `plt.show()`. After `show()` the figure is cleared and the saved file will be blank.

2. `plt.legend()` only works if you passed `label=` inside the plotting call. A legend with no labels shows nothing.

6. Plotting Directly from a DataFrame

```
import pandas as pd
df = pd.DataFrame({'Subject':
['IP', 'Maths', 'Eco'],
                  'Marks':   [92, 78, 70]})

plt.bar(df['Subject'], df['Marks'])
plt.show()

# or the built-in Pandas plot
df.plot(kind='bar', x='Subject', y='Marks')
plt.show()
```

kind can be 'line', 'bar', 'barh', 'hist'.

7. Predict the Output / Identify the Error

Q1 - what does this produce? `plt.plot([1,2,3], [2,4,6], 'g--*')`

Q2 - why is the saved image blank? `plt.plot(x,y)`
`plt.show()`
`plt.savefig("a.png")`

Q3 - why does no legend appear? `plt.plot(x, y)`
`plt.legend()`

Q4 - which chart for this data? Heights of 100 students grouped into ranges

8. Task Triggers

wording → which chart / command?

If the question says...	Use
"show the trend of sales over 12 months"	Line plot — <code>plt.plot()</code>
"compare marks of 5 students"	Bar graph — <code>plt.bar()</code>
"show how many students fall in each marks range"	Histogram — <code>plt.hist()</code>
"frequency distribution"	Histogram
"horizontal bars"	<code>plt.barh()</code>
"add a heading to the chart"	<code>plt.title()</code>
"label the axes"	<code>plt.xlabel()</code> and <code>plt.ylabel()</code>
"show which line is which"	<code>label=</code> + <code>plt.legend()</code>
"dotted red line with circle markers"	<code>plt.plot(x,y,'r:o')</code>
"save the chart as an image"	<code>plt.savefig()</code> before <code>plt.show()</code>
"display the chart"	<code>plt.show()</code>
"divide the data into 6 intervals"	<code>plt.hist(data, bins=6)</code>

MOST-ASKED BOARD QUESTIONS

1. [3-4M] Write code to plot the given data as a line / bar / histogram with labels, title and legend.
2. [2M] Differentiate between a bar graph and a histogram.

3. [2M] Identify and correct the error in the given plotting code.
4. [1-2M] Which chart is suitable for the given data, and why?
5. [1M] Which function saves a chart to a file? → `plt.savefig()`
6. [1M] Which library is used for plotting? → **Matplotlib (pyplot)**

THE HABIT THAT SCORES — Every plotting answer must contain **five lines**: the `import`, the plot call, `xlabel`, `ylabel`, `title`, and `show()`. Marks are allotted separately for the labels and title — a correct chart with no labels loses half the question.

Chapter 4 · Database Query using SQL

CBSE Class 12 Informatics Practices (065) · Unit 2

= 25 marks · Ultimate Revision Sheet

60-SECOND RECALL

Function families (4)	Math · Text · Date · Aggregate
5 aggregate functions	MAX() MIN() AVG() SUM() COUNT()
WHERE vs HAVING	WHERE filters rows BEFORE grouping · HAVING filters groups AFTER
COUNT(*) vs COUNT(col)	COUNT(*) counts all rows · COUNT(col) skips NULL
Clause order	SELECT → FROM → WHERE → GROUP BY → HAVING → ORDER BY
Equi-join	Join two tables on a common column using =

1. Math Functions

THE THREE PRESCRIBED

Function	What it does	Example → Result
POWER(x, y)	x raised to the power y	POWER(2, 3) → 8
ROUND(x, d)	Rounds x to d decimal places. A negative d rounds to the left of the decimal point.	ROUND(15.678, 2) → 15.68 ROUND(15.678) → 16 ROUND(158, -1) → 160

MOD(x, y)

Remainder when x
is divided by y

MOD(10,3) → 1

TRAP

ROUND

with a *negative* second
argument rounds to tens, hundreds and so
on. ROUND(4567, -2) gives **4600**. This exact
case is a favourite 1-marker.

2. Text (String) Functions

ALL PRESCRIBED TEXT FUNCTIONS

Function	What it does	Example → Result
UCASE() / UPPER()	Converts to upper case	UCASE('india') → INDIA
LCASE() / LOWER()	Converts to lower case	LCASE('INDIA') → india
MID() / SUBSTRING() / SUBSTR()	Extracts a substring: (string, start, length). Position starts at 1, not 0.	MID('Informatics',3,4) → form
LENGTH()	Number of characters, including spaces	LENGTH('Data Base') → 9
LEFT(str, n)	Leftmost n characters	LEFT('Computer',4) → Comp

<code>RIGHT(str, n)</code>	Rightmost n characters	<code>RIGHT('Computer', 3) → ter</code>
<code>INSTR(str, sub)</code>	Position of the first occurrence; 0 if not found	<code>INSTR('Informatics', 'mat') → 5</code>
<code>LTRIM()</code>	Removes leading spaces	<code>LTRIM(' Hi ') → Hi</code>
<code>RTRIM()</code>	Removes trailing spaces	<code>RTRIM('Hi ') → Hi</code>
<code>TRIM()</code>	Removes both leading and trailing spaces	<code>TRIM(' Hi ') → Hi</code>

- THE TWO TEXT TRAPS**
1. *SQL string positions start at 1, not 0.* `MID('ABCDEF', 1, 3)` gives **ABC**.
 2. `LENGTH()` *counts spaces too.* `LENGTH('A B')` is **3**.

3. Date Functions

ALL PRESCRIBED DATE FUNCTIONS

Function	What it does	Example (date = 2026-09-11)
----------	--------------	-----------------------------

NOW()	Current date AND time	2026-09-11 05:25:00
DATE()	Extracts the date part only	DATE(NOW()) → 2026-09-11
YEAR()	Year as a number	2026
MONTH()	Month as a number 1–12	9
MONTHNAME()	Month as a name	September
DAY()	Day of month as a number	11
DAYNAME()	Day as a name	Friday

TRAP MONTH() returns a *number*,
 MONTHNAME() returns a *word*. Same for DAY()
 and DAYNAME(). Read the question wording
 carefully — “month name” means
 MONTHNAME().

4. Aggregate Functions

THE FIVE

Function	Returns	NULL handling
MAX(col)	Largest value	Ignores NULL
MIN(col)	Smallest value	Ignores NULL
AVG(col)	Average	Ignores NULL
SUM(col)	Total	Ignores NULL
COUNT(col)	Number of non-NULL values	Ignores NULL

COUNT(*)

Total number of rows

Includes NULL
rows

THE GUARANTEED 1-MARKER

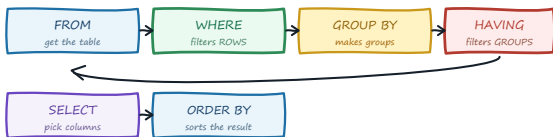
COUNT(*) counts every row. COUNT(column) skips NULLs.

If a table has 10 rows and 3 have NULL in Grade, then COUNT(*) = 10 but COUNT(Grade) = 7.

Also: all aggregate functions except COUNT(*) ignore NULL.

5. GROUP BY, HAVING and ORDER BY

You WRITE the clauses in one order. The database RUNS them in another.



WHERE runs BEFORE the grouping, HAVING runs AFTER it. That single fact decides which one the question wants.

• A condition on a plain column → WHERE e.g. Salary > 20000

• A condition on an AGGREGATE → HAVING e.g. COUNT(*) > 2

An aggregate inside WHERE is always an error — the groups do not exist yet at that point.

Written order is SELECT first; executed order is FROM first. HAVING can see aggregates only because it runs last.

THE CLAUSE ORDER — never change it

```
SELECT    column, AGGREGATE(column)
FROM      table
WHERE     condition           -- filters ROWS,
before grouping
GROUP BY  column             -- makes the groups
HAVING    condition          -- filters GROUPS,
after grouping
ORDER BY  column ASC|DESC;    -- sorts the result
```

WORKED EXAMPLE

```
SELECT Dept, COUNT(*), AVG(Salary)
FROM   Employee
WHERE  Salary > 20000
GROUP BY Dept
HAVING COUNT(*) > 2
ORDER BY AVG(Salary) DESC;
```

Reads as: take employees earning over 20000 → group them by department → keep only departments with more than 2 such employees → sort by average salary, highest first.

WHERE vs HAVING — asked every year

Basis	WHERE	HAVING
Filters	Individual rows	Groups
When	BEFORE grouping	AFTER grouping
Aggregate functions	CANNOT be used	CAN be used
Needs GROUP BY	No	Usually yes

Rule: if the condition uses COUNT , SUM , AVG , MAX or MIN , it must go in HAVING.

```
ORDER BY Marks;           -- ASC is the DEFAULT
ORDER BY Marks DESC;
ORDER BY Dept, Marks DESC; -- sort by Dept, then
                             Marks within it
```

6. Equi-Join — Working with Two Tables

DEFINITION An *equi-join* combines rows from two tables by matching a **common column** using the **equality (=)** operator. The common column is usually a **primary key** in one table and a **foreign key** in the other.

```
-- Method 1: WHERE clause
SELECT E.Name, D.DeptName
FROM   Employee E, Dept D
WHERE  E.DeptID = D.DeptID;

-- Method 2: JOIN ... ON (same result)
SELECT E.Name, D.DeptName
FROM   Employee E JOIN Dept D
ON     E.DeptID = D.DeptID;
```

THE RULES

1. Always state the *join condition*, or you get a *Cartesian product* (every row of A paired with every row of B).
2. If a column name exists in *both* tables, *qualify it*: E.DeptID, not just DeptID.
3. Cartesian product row count = *rows in A* × *rows in B*; column count = *columns in A* + *columns in B*.

TRAP

If Employee has 5 rows / 4 columns and Dept has 3 rows / 2 columns, then a *Cartesian product* gives 15 rows and 6 columns. This calculation is a common 1–2 marker.

7. Revision — Core SQL from Class XI

DDL / DML / DQL

```

-- DDL: structure
CREATE DATABASE school;
USE school;
CREATE TABLE Student (RollNo INT PRIMARY KEY,
                        Name VARCHAR(30),
                        Marks DECIMAL(5,2));
ALTER TABLE Student ADD Grade CHAR(2);
DROP TABLE Student;

-- DML: data
INSERT INTO Student VALUES (1, 'Amit', 92.5);
UPDATE Student SET Marks = 95 WHERE RollNo = 1;
DELETE FROM Student WHERE RollNo = 1;

-- DQL: querying
SELECT * FROM Student;
SELECT DISTINCT Dept FROM Employee;
SELECT * FROM Student WHERE Marks BETWEEN 60
AND 90;
SELECT * FROM Student WHERE Name LIKE 'A%';
SELECT * FROM Student WHERE Grade IS NULL;
SELECT * FROM Student WHERE Dept IN
('Sales', 'HR');

```

Operator	Meaning
BETWEEN a AND b	Range, both ends INCLUSIVE
IN (list)	Matches any value in the list
LIKE '%'	% = any number of characters; _ = exactly one character
IS NULL / IS NOT NULL	Tests for NULL. Never use = NULL — it never matches.

TRAP *NULL is not zero and not an empty string. It means "unknown". You must use IS NULL ; WHERE Grade = NULL returns no rows, always.*

8. Predict the Output

Q1 SELECT ROUND(4567.789, -2); →
SELECT MOD(17, 5); →
SELECT POWER(3, 3); →

Q2 SELECT MID('Informatics', 3, 4); →
SELECT INSTR('Informatics','mat'); →
SELECT LENGTH('Data Base'); →

Q3 - table has 8 rows, 2 NULLs in Grade SELECT
COUNT(*) →
SELECT COUNT(Grade) →

Q4 - A has 4 rows/3 cols, B has 2 rows/2 cols SELECT *
FROM A, B; → (Cartesian product)

9. Task Triggers wording → which clause / function?

If the question says...	Use
"total number of records"	COUNT(*)
"how many employees have a grade recorded"	COUNT(Grade)
"department-wise average salary"	GROUP BY Dept + AVG()
"departments having more than 5 employees"	HAVING COUNT(*) > 5

"employees earning more than 30000"	WHERE Salary > 30000
"in descending order of marks"	ORDER BY Marks DESC
"names starting with A"	LIKE 'A%'
"second letter is a"	LIKE '_a%'
"salary between 20000 and 40000"	BETWEEN 20000 AND 40000
"where grade is not assigned"	IS NULL
"remove duplicate department names"	SELECT DISTINCT
"display name in capitals"	UCASE (Name)
"first 3 characters of the name"	LEFT (Name,3) or MID (Name,1,3)
"month name of joining date"	MONTHNAME (DOJ)
"data from two tables together"	Equi-join with a matching condition
"add a new column to the table"	ALTER TABLE ... ADD
"change the marks of roll no 5"	UPDATE ... SET ... WHERE

MOST-ASKED BOARD QUESTIONS

1. **[4-5M]** Write SQL queries for the given table (usually 5 queries of 1 mark each).
2. **[2-3M]** Predict the output of the given SQL statements.
3. **[2M]** Differentiate between **WHERE** and **HAVING**.

4. [2M] Differentiate between `COUNT(*)` and `COUNT(column)`.
5. [2M] Write a query using an equi-join on two tables.
6. [1M] What is the default sort order of `ORDER BY`? → **ASC**

THE HABIT THAT SCORES — Write every query on separate lines by clause and end with a **semicolon**. Then check the order: `SELECT` → `FROM` → `WHERE` → `GROUP BY` → `HAVING` → `ORDER BY`. A clause out of order is marked wrong even if the logic is right.

Chapter 5 · Introduction to Computer Networks

CBSE Class 12 Informatics Practices (065) · Unit 3

= 10 marks · Ultimate Revision Sheet

60-SECOND RECALL

4 network types	PAN (few m) → LAN (building) → MAN (city) → WAN (country/world)
6 devices	Modem · Hub · Switch · Repeater · Router · Gateway
4 topologies	Star · Bus · Tree · Mesh
Hub vs Switch	Hub broadcasts to ALL · Switch sends only to the intended device
Router vs Gateway	Router joins similar networks · Gateway joins dissimilar ones
Website vs Webpage	Website = collection of webpages · Webpage = a single document

1. Network and Its Types

DEFINITION A computer network is a collection of computers and devices connected together to share data, hardware and software resources.

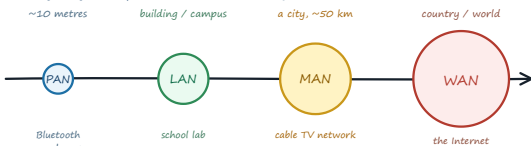
THE FOUR TYPES — by geographical spread

Type	Full form	Range	Example
------	-----------	-------	---------

PAN	Personal Area Network	A few metres (up to ~10 m)	Bluetooth between a phone and earphones; a phone hotspot
LAN	Local Area Network	A room, building or campus (up to a few km)	A school computer lab; an office network
MAN	Metropolitan Area Network	A city (up to ~50 km)	Cable TV network; all branches of a bank in one city
WAN	Wide Area Network	A country or the world	The Internet — the largest WAN

As range increases: speed decreases, cost increases, and error rate increases.

The only thing that separates them is how far they reach.



As the range grows: speed falls, cost rises, error rate rises.

The Internet is the largest WAN — a 1-mark answer that appears almost every year.

2. Network Devices

ALL SIX PRESCRIBED DEVICES

Device	What it does
--------	--------------

Modem	MO dulator- DE Modulator. Converts digital signals to analog for transmission over telephone lines, and back again. Needed to connect to the Internet over a phone line.
Hub	Connects multiple devices in a LAN. It broadcasts incoming data to ALL connected ports — it cannot tell which device the data is for. Result: more collisions, less secure, slower.
Switch	Like a hub, but intelligent . It reads the MAC address and forwards data only to the intended device. Faster, more secure, fewer collisions.
Repeater	Regenerates and amplifies a weak signal so it can travel a longer distance. Used when cable length exceeds the medium's limit.
Router	Connects two or more networks of the same type and chooses the best path for data. Works with IP addresses .
Gateway	Connects two DISSIMILAR networks that use different protocols. It acts as a protocol converter — the entry and exit point of a network.

THE TWO PAIRS EXAMINERS TEST *Hub vs*

Switch: Hub broadcasts to all (dumb);

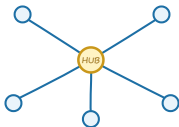
Switch sends only to the intended recipient (intelligent, uses MAC address).

Router vs Gateway: Router connects networks using the same protocol;

Gateway connects networks using different protocols and converts between them.

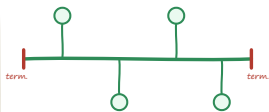
3. Network Topologies

STAR



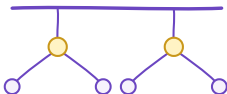
hub dies → whole network dies

BUS



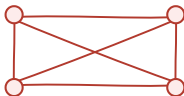
one backbone — cheapest, but it breaks

TREE



stars hung off a bus backbone

MESH



everyone to everyone — links = $n(n-1)/2$

Most reliable → **Mesh**. Cheapest → **Bus**. Easiest to troubleshoot → **Star**. That trio answers most 1-markers.

THE FOUR PRESCRIBED TOPOLOGIES

Topology	Structure	Advantages	Disadvantages
STAR	All nodes connect to a central hub/switch	Easy to install and manage · failure of one node does not affect others · easy fault detection · easy to add nodes	Central device failure brings down the whole network · more cable needed · costly

BUS	All nodes connect to a single backbone cable ; terminators at both ends	Least cable , so cheapest · easy to extend · simple to set up	Backbone failure breaks the whole network · difficult fault detection · performance drops as nodes increase · collisions
TREE	Hierarchical — groups of star networks joined to a bus backbone	Scalable · easy to expand · good for large networks · supports grouping	Heavy cabling · backbone failure affects everything · costly and complex
MESH	Every node connects to every other node	Most reliable — multiple paths, failure of one link does not stop communication · very secure · no traffic problem	Most expensive — huge cabling · very complex to install and maintain

Mesh links for n nodes = $n(n - 1) / 2$

TRAP *Star fails if the central device fails. Bus fails if the backbone cable fails. Mesh is the most reliable but the most expensive. If a question asks for the most reliable topology the answer is Mesh; for the cheapest it is Bus; for the easiest to troubleshoot it is Star.*

4. Internet — Key Terms

Term	Meaning
Internet	A global network of networks connected using TCP/IP. The largest WAN.
WWW	World Wide Web — the collection of interlinked hypertext documents accessed over the Internet. The Web is a service ON the Internet, not the Internet itself.
URL	Uniform Resource Locator — the unique address of a resource on the web. <code>https://www.cbse.gov.in/notes/ip.html</code> Parts: protocol (<code>https</code>) · domain name (<code>www.cbse.gov.in</code>) · path (<code>/notes/</code>) · file name (<code>ip.html</code>)
Protocol	A set of rules for communication. HTTP, HTTPS, FTP, TCP/IP, SMTP.
IP address	A unique numeric address of a device on a network.
Domain name	The human-readable name of a website; translated to an IP address by DNS .

Applications of the Internet

Application	Meaning
Web	Browsing and accessing information through websites
Email	Electronic mail — sending messages and attachments
Chat	Real-time text conversation between users
VoIP	Voice over Internet Protocol — making voice calls over the Internet instead of a telephone line. <i>Eg: WhatsApp call, Skype, Google Meet.</i> Cheaper, but depends on connection quality.

5. Website, Webpage and Hosting

THE DEFINITIONS

Term	Meaning
Webpage	A single document written in HTML, viewable in a browser
Website	A collection of related webpages under one domain name
Home page	The first / main page of a website, usually <code>index.html</code>
Web server	A computer that stores websites and delivers them to browsers on request
Web hosting	The service of storing a website on a web server so it is available on the Internet

STATIC vs DYNAMIC WEB PAGE — a common 2-marker

Basis	STATIC	DYNAMIC
Content	Fixed — same for every user	Changes with user, time or input
Built with	HTML, CSS only	HTML + server-side scripting (PHP, ASP) + database
Speed	Faster to load	Slower — content is generated on request
Cost / complexity	Cheap and simple	Costly and complex
Example	A simple "About Us" page	Gmail inbox, a shopping site, a news feed

6. Web Browsers

DEFINITION A web browser is application software used to access, retrieve and display content from the World Wide Web. Examples: Google Chrome, Mozilla Firefox, Microsoft Edge, Safari, Opera.

Term	Meaning
Browser settings	Options to control the browser — home page, default search engine, privacy and security, downloads, clearing history and cache.
Add-on	A small program that extends the browser's own functionality — ad blockers, password managers, grammar checkers. It becomes part of the browser .
Plug-in	A third-party program that lets the browser display or handle a specific type of content it could not otherwise — PDF viewers, Flash, media players. It is a separate program that works with the browser.
Cookie	A small text file stored on the user's computer by a website, holding information such as login details, preferences and items in a cart. Used to remember the user on the next visit.
Cache	Temporarily stored copies of web content so pages load faster on the next visit.

ADD-ON vs PLUG-IN Add-on = enhances the browser itself, becomes part of it.

Plug-in = a separate third-party program that lets the browser handle specific content.

Cookies — the privacy angle: they improve convenience (staying logged in) but can track browsing behaviour, which is a privacy concern. This link to Unit 4 is often asked.

7. Question Triggers identify the device / type

If the question says...	The answer is
"connects a phone to earphones wirelessly"	PAN (Bluetooth)
"computers in one school lab"	LAN
"all branches of a bank within one city"	MAN
"offices in Delhi, Mumbai and London"	WAN
"signal weakens over a long cable"	Repeater
"sends data only to the intended computer"	Switch
"broadcasts data to every connected device"	Hub
"connects two networks using different protocols"	Gateway

"chooses the best path between networks"	Router
"converts digital to analog signals"	Modem
"one central device, all nodes connect to it"	Star topology
"single backbone cable with terminators"	Bus topology
"every node connected to every other node"	Mesh topology
"which topology is most reliable?"	Mesh
"which topology needs least cable?"	Bus
"voice call over the Internet"	VoIP
"a small file that remembers your login"	Cookie
"content changes for each user"	Dynamic web page
"storing a website on a server for public access"	Web hosting

MOST-ASKED BOARD QUESTIONS

1. **[3-4M]** *Case study:* an organisation has four buildings — suggest the topology, the device, the cable layout, and where to place the server.
2. **[2M]** *Differentiate between hub and switch / router and gateway.*

3. [2M] Differentiate between a static and a dynamic web page.
4. [2M] Differentiate between a website and a webpage / add-on and plug-in.
5. [1M] Expand VoIP / URL / PAN.
6. [1M] Which topology is the most reliable? → Mesh

THE HABIT THAT SCORES — The case-study question is worth 3–4 marks and follows the same shape every year: name the **topology** (usually Star or Bus), name the **device** (usually Switch for within a building, Router between buildings), and put the **server in the building with the most computers**. Give a one-line **reason** for each — the reason is the mark.

Chapter 6 · Societal Impacts

CBSE Class 12 Informatics Practices (065) · Unit 4

= 10 marks · Ultimate Revision Sheet

60-SECOND RECALL

Digital footprint	Active = you post it · Passive = collected without you knowing
3 IPR types	Copyright (creative works) · Patent (inventions) · Trademark (brand identity)
FOSS	Free and Open Source Software — “free” means freedom , not price
4 cybercrimes	Hacking · Phishing · Cyber bullying · Identity theft
Indian law	IT Act 2000 , amended 2008
E-waste rule	3 Rs — Reduce, Reuse, Recycle

1. Digital Footprint

DEFINITION A *digital footprint* is the trail of data you leave behind while using the Internet — websites visited, emails sent, posts, searches, and data submitted online. It is *permanent* and can be traced back to you.

Type

Meaning

Examples

Active	Data you deliberately share	Social media posts, comments, emails sent, filling an online form, uploading a photo
Passive	Data collected without your knowledge	IP address logged by a website, browsing history, cookies, location tracking, search history

Why it matters: a digital footprint is permanent and public; employers and colleges check it; it can be misused for identity theft or targeted advertising. Manage it by thinking before posting, adjusting privacy settings, using strong passwords, logging out of shared devices, and periodically searching your own name.

2. Net and Communication Etiquette

THE THREE ETIQUETTES

Type	Rules
Net etiquette <i>netiquette</i>	Be ethical — no copyright violation, no hacking · Be respectful — respect privacy and diversity · Be responsible — avoid cyber bullying, do not spread rumours
Communication etiquette	Be precise — do not waste others' time · Be polite — no abusive language · Be credible — do not post false information · Avoid ALL CAPS , which reads as shouting

Social media etiquette

Choose your **friends** carefully · Do not **overshare** personal information · **Beware of fake news** · Respect others' sentiments · Keep personal and professional profiles separate

3. Data Protection

CONCEPT Data protection means safeguarding data from **unauthorised access, loss, corruption or misuse**. Sensitive personal data includes bank details, passwords, biometric data, medical records and Aadhaar numbers.

Ways to protect data: strong and unique passwords · two-factor authentication · encryption · firewalls and antivirus · regular backups · avoiding public Wi-Fi for sensitive work · checking for **HTTPS** before entering data · logging out of shared computers.

4. Intellectual Property Rights (IPR)

DEFINITION Intellectual Property Rights are the legal rights given to a person over the **creations of their mind** — inventions, literary and artistic works, designs, symbols and names.

THE THREE TYPES — know exactly what each protects

Type	Protects	Duration	Example
Copyright	Creative and literary works — books, music, films, software code, paintings, photographs	Life of author + 60 years (India)	© symbol on a book or song
Patent	Inventions — a new product or process that is novel, useful and non-obvious	20 years	A new drug formula; a new machine design
Trademark	Brand identity — names, logos, symbols, slogans that distinguish goods	10 years, renewable	™ and ® symbols; the Nike swoosh

THE ONE-LINE TEST *Copyright* → something

created (a work of expression).

Patent → something *invented* (a new product or process).

Trademark → something that *identifies a brand* (name, logo, slogan).

5. Plagiarism, Licensing and Copyright

Term	Meaning
------	---------

Plagiarism	Presenting someone else's work or ideas as your own without giving credit. It is academic theft , whether intentional or not.
Copyright infringement	Using copyrighted material without permission of the owner — even with credit. Illegal .
Licence	The permission given by the owner stating how the work may be used, modified and distributed.
Public domain	Works whose copyright has expired or been waived — free for anyone to use.
Creative Commons (CC)	A set of licences that let creators allow certain uses while keeping some rights.

PLAGIARISM vs COPYRIGHT INFRINGEMENT — a frequent 2-marker

Plagiarism = using someone's work WITHOUT giving credit → an ethical / academic offence.

Copyright infringement = using copyrighted work WITHOUT permission, even if you credit the author → a legal offence.

Citing the source fixes plagiarism but not copyright infringement — for that you need permission.

Avoid plagiarism by: citing all sources · using quotation marks for exact text · paraphrasing in your own words and still

citing · adding a bibliography · using plagiarism-check tools.

6. Free and Open Source Software (FOSS)

THE CATEGORIES

Type	Meaning	Examples
Free Software	Freedom to run, study, modify and redistribute . "Free" refers to freedom , not price.	GNU tools
Open Source Software	Source code is available and may be modified and redistributed	Linux, Firefox, Python, LibreOffice, MySQL
FOSS	Free and open source — both freedoms together, usually free of cost	Ubuntu, GIMP, Apache
Proprietary Software	Source code is closed ; usage restricted by licence; usually paid	Microsoft Windows, MS Office, Photoshop
Freeware	Free of cost but the source code is NOT available	Adobe Reader, Skype, Zoom (basic)
Shareware	Free to use for a trial period , then payment is required	Trial versions of WinZip, WinRAR

THE CLASSIC TRAP

Freeware is NOT open source. Freeware costs nothing but its **source code is hidden**. Open source gives you the **code**, whether or not it costs money. “Free” in FOSS means **freedom**, not price — “free as in speech, not free as in beer”.

7. Cybercrime and Cyber Law

THE MAIN CYBERCRIMES

Crime	Meaning
Hacking	Gaining unauthorised access to a computer system or network. White hat = ethical hacker, tests security with permission. Black hat = malicious hacker, illegal. Grey hat = in between; breaks in without permission but without malicious intent.
Phishing	Sending fake emails, messages or websites that look genuine to trick users into revealing passwords, OTPs or bank details .
Cyber bullying	Harassing, threatening, humiliating or targeting a person using digital technology — abusive messages, spreading rumours, posting embarrassing content.
Cyber trolling	Deliberately posting provocative or offensive comments online to upset people.
Identity theft	Stealing someone's personal information to impersonate them, usually for financial gain.

Ransomware

Malware that **encrypts the victim's data** and demands payment to release it.

Cyber stalking

Repeatedly following or monitoring a person online, causing fear.

THE INFORMATION TECHNOLOGY ACT, 2000

India's primary law on cybercrime and electronic commerce, *amended in 2008*.

It:

- gives legal recognition to electronic records and digital signatures
- defines cyber offences and their penalties — hacking, identity theft, publishing obscene material, data theft
- provides for the **Cyber Appellate Tribunal**
- makes provision for data protection and privacy

Reporting: cybercrime can be reported at a cyber cell or on the national portal cybercrime.gov.in.

8. E-Waste — Hazards and Management

DEFINITION E-waste is discarded electrical and electronic equipment — old computers, mobile phones, printers, batteries, televisions, cables.

Hazards

- Contains **toxic metals** — lead, mercury, cadmium, arsenic
- **Soil and water pollution** when dumped in landfills
- **Air pollution** from open burning of plastics
- Serious **health effects** — damage to the nervous system, kidneys, lungs and skin
- Harms workers in the informal recycling sector, often children

Management — the 3 Rs and more

- **Reduce** — buy only what is needed; extend device life
- **Reuse** — donate or resell working equipment
- **Recycle** — hand over to authorised e-waste recyclers

- Follow the **E-Waste (Management) Rules**; use producer take-back schemes
- **Never** dump electronics with household waste

9. Health Concerns from Technology Use

Problem	Cause and prevention
Eye strain Computer Vision Syndrome	Long screen exposure. Prevention: the 20-20-20 rule — every 20 minutes , look at something 20 feet away for 20 seconds . Adjust brightness; blink often.
Repetitive Strain Injury (RSI)	Repeated typing and mouse use. Prevention: take breaks, stretch, use ergonomic keyboard and mouse.
Back and neck pain	Poor posture. Prevention: ergonomic chair, screen at eye level, feet flat on the floor.
Internet addiction	Excessive use affecting daily life. Prevention: set screen-time limits, take digital detox breaks.
Sleep disturbance	Blue light before bed. Prevention: avoid screens an hour before sleeping; use night mode.
Obesity	Sedentary screen time. Prevention: regular physical activity.

Ergonomics is the science of designing the workplace to fit the user, reducing strain and injury.

10. Question Triggers

identify the concept

If the case says...

The answer is

"posted a photo on social media"	Active digital footprint
"website recorded her IP address silently"	Passive digital footprint
"copied an essay without citing the source"	Plagiarism
"used a copyrighted song with credit but no permission"	Copyright infringement
"invented a new machine and wants protection"	Patent
"wants to protect the company logo"	Trademark
"wrote a book and wants protection"	Copyright
"received a fake bank email asking for OTP"	Phishing
"someone repeatedly sent threatening messages"	Cyber bullying
"gained access to a server without permission"	Hacking
"tested security with the company's permission"	White hat / ethical hacking
"data encrypted, money demanded"	Ransomware
"free to use but source code hidden"	Freeware (not open source)
"source code available to modify"	Open source
"free for 30 days then pay"	Shareware
"old laptops dumped in a landfill"	E-waste hazard
"eyes hurt after long screen use"	Computer Vision Syndrome → 20-20-20 rule

"wrist pain from constant typing"

RSI

"which law covers cybercrime in India?"

IT Act 2000 (amended 2008)

MOST-ASKED BOARD QUESTIONS

1. **[2-3M]** *Case study: identify the cybercrime/IPR issue and suggest what the person should do.*
2. **[2M]** *Differentiate between plagiarism and copyright infringement.*
3. **[2M]** *Differentiate between active and passive digital footprint / freeware and open source.*
4. **[2-3M]** *What is e-waste? Explain its hazards and management.*
5. **[2M]** *Explain any two health concerns from technology and how to prevent them.*
6. **[1M]** *Expand FOSS / IPR. Which Act deals with cybercrime in India?*

THE HABIT THAT SCORES — This unit is examined almost entirely through **short case studies**. Answer in three steps: **name the concept → define it in one line →**

state what the person should do. The “what to do” step is where half the marks sit, and it is the one students skip.