

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()`.