CUET UG Information Practices Sample Paper - 9

Duration: 1 Hour **Maximum Marks: 250**

Instructions

- This paper contains a total of 50 Multiple Choice Questions.
- Each correct answer carries **+5 marks**.
- Each incorrect answer carries **-1 mark**.
- No negative marking for unattempted questions.

Q1. A DataFrame `df` contains 50 rows. A user executes `df.iloc[10:20, :].shape`. What will be the output?

(A) (10, 50)

(B) (11, 50)

(C) (10, number of columns)

(D) (20, number of columns)

Q2. Which SQL function is used to return the position of the first occurrence of a substring `'at'` in the string `'Informatics'`?

(A) `SEARCH('at', 'Informatics')`

(B) `INSTR('Informatics', 'at')`

(C) `SUBSTR('Informatics', 'at')`

(D) `LOCATE('Informatics', 'at')`

Q3. Observe the following Matplotlib code:

```
plt.bar([1, 2, 3], [10, 20, 30], width=[0.1, 0.2, 0.5])
```

What is the effect of passing a list to the `width` parameter?

(A) It changes the color of each bar.

(B) It assigns different thicknesses to each individual bar.

(C) It specifies the distance between the x-axis and the bars.

(D) It results in a `ValueError`.

Q4. In Networking, which device operates at the Physical Layer of the OSI model and is used to connect multiple segments of a network by broadcasting incoming data to all ports?

(A) Switch

(B) Router

(C) Hub

(D) Gateway

Q5. Consider the following SQL query:

```
SELECT ROUND(156.78, -2);
```

The output will be:

(A) 156.8

(B) 160

(C) 200

(D) 100

Q6. Which property of a Pandas Series returns the underlying data as a NumPy ndarray?

(A) `S.data`

(B) `S.values`

(C) `S.array`

(D) `S.tolist()`

Q7. Identify the correct SQL clause to arrange the results of a query based on the descending order of `Salary` and ascending order of `Name`.

(A) `ORDER BY Salary DESC, Name ASC`

(B) `SORT BY Salary DESC, Name`

(C) `ORDER BY Salary, Name DESC`

(D) `GROUP BY Salary DESC, Name`

Q8. A user wants to separate the 'Health' slice from a Pie Chart by 20% of the radius. Which value should be placed in the `explode` list for that index?

(A) 20

(B) 0.2

(C) 2.0

(D) 0.02

Q9. Which of the following describes the 'Selection' (σ) operation in Relational Algebra?

(A) Extracts specific columns from a table.

(B) Combines two tables based on a common attribute.

(C) Extracts rows that satisfy a specified condition.

(D) Removes duplicate rows from a dataset.

Q10. What is the purpose of the `df.info()` method in Pandas?

(A) To display the first 5 rows of the DataFrame.

(B) To show the statistical summary like mean and median.

(C) To provide a concise summary including non-null counts and memory usage.

(D) To list only the column names.

Q11. Which type of Cyber Crime involves redirecting a user to a fraudulent website even if they typed the correct URL in the browser?

(A) Phishing

(B) Pharming

(C) Spimming

(D) Salami Slicing

Q12. Match the following SQL aggregate functions with their descriptions:

(i) `COUNT(*)`	(a) Returns the average of non-null values
(ii) `COUNT(col)`	(b) Returns total number of rows including NULLs
(iii) `AVG()`	(c) Returns the sum of values
(iv) `SUM()`	(d) Returns number of non-null values in a column

(A) (i)-b, (ii)-d, (iii)-a, (iv)-c

(B) (i)-d, (ii)-b, (iii)-a, (iv)-c

(C) (i)-b, (ii)-a, (iii)-d, (iv)-c

(D) (i)-a, (ii)-d, (iii)-b, (iv)-c

Q13. To add a label to the Y-axis in a Matplotlib histogram, which function is used?

(A) `plt.y_axis("Frequency")`

(B) `plt.ylabel("Frequency")`

(C) `plt.label_y("Frequency")`

(D) `plt.set_y("Frequency")`

Q14. Which Pandas method is used to fill all missing (NaN) values in a DataFrame with a constant value 0?

(A) `df.dropna(0)`

(B) `df.fillna(0)`

(C) `df.replace(0)`

(D) `df.nullfill(0)`

Q15. What is the output of the SQL query: `SELECT LENGTH(TRIM(' CUET '));`?

(A) 8

(B) 6

(C) 4

(D) 10

Q16. Given a Series `S = pd.Series([10, 20, 30, 40, 50])`. What will be the result of `S[S > 30] * 2`?

(A) `3: 80, 4: 100`

(B) `0: 20, 1: 40`

(C) `40, 50`

(D) `80, 100`

Q17. Which property of a DataFrame is used to transpose rows and columns?

(A) `df.swap()`

(B) `df.T`

(C) `df.transpose`

(D) `df.pivot()`

Q18. What will be the output of the SQL query: `SELECT SUBSTR("INFORMATICS", 3, 4);`?

(A) `INFO`

(B) `FORM`

(C) `ORMA`

(D) `MATI`

Q19. In Matplotlib, if you want to create a Histogram with 10 bins but only for the data range between 50 and 100, which parameter combination is correct?

(A) `plt.hist(data, bins=10, limit=(50, 100))`

(B) `plt.hist(data, bins=10, range=(50, 100))`

(C) `plt.hist(data, intervals=10, range=(50, 100))`

(D) `plt.hist(data, 10, scale=(50, 100))`

Q20. A network that connects devices over a very short distance, typically within a range of an individual person (like Bluetooth), is called:

(A) LAN

(B) MAN

(C) PAN

(D) WAN

Q21. Consider the table `EXAM` with columns (`RollNo, Marks, Subject`). Which query will display the average marks of each subject where the average is more than 75?

(A) `SELECT Subject, AVG(Marks) FROM EXAM WHERE AVG(Marks) > 75 GROUP BY Subject;`

(B) `SELECT Subject, AVG(Marks) FROM EXAM GROUP BY Subject HAVING AVG(Marks) > 75;`

(C) `SELECT Subject, AVG(Marks) FROM EXAM GROUP BY Subject WHERE AVG(Marks) > 75;`

(D) `SELECT Subject, AVG(Marks) FROM EXAM HAVING AVG(Marks) > 75;`

Q22. Which Pandas method is used to combine two DataFrames side-by-side based on a common key (similar to a SQL Join)?

(A) `pd.concat()`

(B) `pd.append()`

(C) `pd.merge()`

(D) `pd.combine()`

Q23. Match the following Pandas attributes with their meanings:

(i) `df.shape`	(a) Total number of elements
(ii) `df.size`	(b) Row and Column count as a tuple
(iii) `df.ndim`	(c) Data types of columns
(iv) `df.dtypes`	(d) Number of dimensions (always 2 for DF)

(A) (i)-b, (ii)-a, (iii)-d, (iv)-c

(B) (i)-a, (ii)-b, (iii)-d, (iv)-c

(C) (i)-b, (ii)-d, (iii)-a, (iv)-c

(D) (i)-c, (ii)-a, (iii)-d, (iv)-b

Q24. What does the SQL function `MOD(11, 4)` return?

(A) 2.75

(B) 2

(C) 3

(D) 1

Q25. To display a chart with a title ”Growth Analysis”, which Matplotlib command is correct?

(A) `plt.set_title("Growth Analysis")`

(B) `plt.title("Growth Analysis")`

(C) `plt.header("Growth Analysis")`

(D) `plt.chart_title("Growth Analysis")`

Q26. Which of the following is an example of Intellectual Property Rights (IPR) violation?

(A) Plagiarism

(B) Trademark Infringement

(C) Copyright Infringement

(D) All of the above

Q27. In SQL, which wildcard character is used to match exactly one character in a `LIKE` pattern?

(A) `%`

(B) `*`

(C) `_`

(D) `?`

Q28. A DataFrame `df` has columns 'Name' and 'Score'. Which command will add a new column 'Grade' with a default value 'A' for all rows?

(A) `df['Grade'] = 'A'`

(B) `df.add_column('Grade', 'A')`

(C) `df.Grade = 'A'`

(D) `df.insert('Grade', 'A')`

Q29. Which networking topology uses a central controller or Hub to which all other nodes are connected?

(A) Bus

(B) Ring

(C) Star

(D) Mesh

Q30. Observe the Pie Chart command: `plt.pie(x, labels=y, autopct='%1.1f%%')`. What does `autopct` do?

(A) It sets the color of the slices automatically.

(B) It displays the percentage value inside the wedges.

(C) It separates the slices from the center.

(D) It rotates the chart automatically.

Q31. Which SQL function returns the current system date and time?

(A) `DATE()`

(B) `CURDATE()`

(C) `NOW()`

(D) `SYSDATE`

Q32. The `read_csv()` function in Pandas by default considers the first row of the CSV file as:

(A) Data

(B) Index

(C) Column Names (Header)

(D) Metadata

Q33. Which of the following describes 'Cyber Stalking'?

(A) Stealing someone's identity online.

(B) Harassing or threatening a person through digital means repeatedly.

(C) Sending unwanted promotional emails.

(D) Spreading viruses to damage a computer.

Q34. What will be the output of `SELECT POWER(3, 2) + ROUND(1.5);`

(A) 11

(B) 10.5

(C) 10

(D) 12

Q35. Which method is used to rename existing row indices in a DataFrame?

(A) `df.rename_index()`

(B) `df.reindex()`

(C) `df.rename()`

(D) `df.index_name()`

Q36. In a Histogram, the height of each bar represents:

(A) The mean of the data in that bin.

(B) The frequency (count) of data points in that bin.

(C) The width of the interval.

(D) The total sum of the values in that bin.

Q37. Which SQL command is used to remove all records from a table but keep the table structure intact?

(A) `DROP TABLE`

(B) `DELETE FROM`

(C) `TRUNCATE TABLE`

(D) `REMOVE TABLE`

Q38. The process of converting a readable text into an unreadable format to prevent unauthorized access is called:

(A) Decryption

(B) Encryption

(C) Masking

(D) Firewalls

Q39. If `df` is a DataFrame, `df.iloc[:, -1]` will return:

(A) The first column.

(B) The first row.

(C) The last column.

(D) The last row.

Q40. Which function is used to add a legend to a Matplotlib plot?

(A) `plt.show_legend()`

(B) `plt.legend()`

(C) `plt.label()`

(D) `plt.info()`

Q41. Consider the following table `SALES`:

S_ID	Region	Amount	Year
1	North	5000	2022
2	South	7000	2022
3	North	3000	2023
4	East	4000	2023

What will be the output of: `SELECT Region, SUM(Amount) FROM SALES GROUP BY Region ORDER BY SUM(Amount) DESC;`

(A) North 8000, South 7000, East 4000

(B) North 5000, South 7000, North 3000, East 4000

(C) South 7000, North 8000, East 4000

(D) North 8000, South 7000, East 4000 (sorted randomly)

Q42. Observe the code: `plt.hist(data, bins=[0, 10, 25, 50, 100])`. What does passing a list to the `bins` parameter indicate?

(A) It creates 5 equal-sized bins.

(B) It defines custom, unequal width intervals (edges) for the bins.

(C) It sets the colors for the 5 bars.

(D) It limits the data to only these five specific numbers.

Q43. Which property of a Pandas DataFrame returns the column labels as an Index object?

(A) `df.colnames`

(B) `df.columns`

(C) `df.keys()`

(D) `df.index`

Q44. In networking, which protocol is responsible for transferring web pages from a server to a client's browser?

(A) FTP

(B) SMTP

(C) HTTP

(D) TCP

Q45. What will be the output of `SELECT INSTR('Computer Science', 'e');` in SQL?

(A) 8

(B) 9

(C) 7

(D) 16

Q46. A user wants to find the number of unique entries in a column 'Subject' of a DataFrame `df`. Which method is most efficient?

(A) `df['Subject'].unique()`

(B) `df['Subject'].nunique()`

(C) `df['Subject'].count()`

(D) `df['Subject'].value_counts()`

Q47. Which type of software license allows you to view the source code, modify it, and distribute it, but usually requires you to keep the same license for the modified version (Copyleft)?

(A) Proprietary

(B) Freeware

(C) GPL (General Public License)

(D) Shareware

Q48. What is the correct way to plot a line chart with a green line and 'square' markers at each data point?

(A) `plt.plot(x, y, color='green', marker='s')`

(B) `plt.plot(x, y, 'g-s')`

(C) `plt.plot(x, y, line='green', points='square')`

(D) `plt.plot(x, y, color='g', shape='s')`

Q49. In the Relational Algebra operation Projection (Π), if a table has 10 rows but the projected column has 3 duplicate values, how many rows will be in the output according to standard relational theory?

(A) 10

(B) 7

(C) 13

(D) 3

Q50. Which of the following describes the Primary Key constraint in SQL?

(A) Unique only

(B) Not Null only

(C) Both Unique and Not Null

(D) Neither Unique nor Not Null

Detailed Solutions

Q1.

Solution

Concept:

The `.iloc` indexer in Pandas is a vital tool for data extraction based on integer positions. Unlike `.loc`, which uses labels, `.iloc` ignores the names of your rows and columns and looks only at their numerical sequence. A key rule in Python and Pandas is that slicing `[a:b]` is inclusive of the start index (`a`) but exclusive of the stop index (`b`). Furthermore, the `.shape` attribute is used to check the dimensions of the resulting object, returning a tuple in the format (number of rows, number of columns). Mastering this is essential for data preprocessing where you must isolate specific chunks of a dataset for analysis or training.

Solution:

1. **Analyzing the Row Slice:** The command uses `10:20`. Following the exclusive-stop rule, this selects rows with index positions 10, 11, 12, 13, 14, 15, 16, 17, 18, and 19. If you count these, it totals exactly 10 rows ($20 - 10 = 10$). 2. **Analyzing the Column Slice:** The second part of the indexer is a single colon (`:`). In slicing notation, this is a wildcard that instructs Pandas to select all available columns from the start of the DataFrame to the very end. 3. **Determining the Shape:** Since the operation successfully isolated 10 rows and kept all original columns, the shape property will report the row count first, followed by the column count. 4. **Final Result:** If the DataFrame has N columns, the shape will be $(10, N)$. This is a standard output format used to verify that a slicing operation has worked as intended before proceeding with further data computation.

Final Answer: The shape of the resulting DataFrame is (10, number of columns).

Answer: (C)

Q2.

Solution

Concept:

String manipulation is a core component of SQL, particularly when searching for patterns or specific data within text fields. Functions like `INSTR()` (In-String) allow a database user to locate the exact starting position of a substring within a larger string. This is useful for parsing data, such as finding a specific domain in an email address or identifying a prefix in a product code. A critical distinction in SQL is that string indexing is **one-based**, meaning the first character is at position 1, not position 0 as in Python. Understanding these nuances prevents ”off-by-one” errors which are common in competitive exams like CUET.

Solution:

1. **Function Selection:** To find a position, we use `INSTR(string, substring)`. Other functions like `SUBSTR` are used for extraction, not for finding a location, while `LOCATE` is a variation but `INSTR` is the standard in the CUET curriculum. 2. **Mapping the Search:** We are searching for ’at’ within ’Informatics’. We count the characters one by one: I(1), n(2), f(3), o(4), r(5), m(6), a(7), t(8). 3. **Identifying the Start:** The substring ’at’ begins with the letter ’a’. Since ’a’ is at the 7th position, the function will return the integer 7. 4. **Handling Cases:** If the substring occurred multiple times, `INSTR()` would return only the position of the very first occurrence. If the substring was not found, it would return 0. In this specific query, the logic is straightforward: find the first ’a’ followed by ’t’ and return its starting index.

Final Answer: The function to find the position is `INSTR('Informatics', 'at')`.

Answer: (B)

Q3.

Solution

Concept:

The `plt.bar()` function in Matplotlib is the primary tool for creating vertical bar charts, which are ideal for comparing categorical variables. While most beginners use a single numeric value for the `width` parameter (applying the same thickness to all bars), Matplotlib allows for advanced customization by passing a list or an array to this parameter. This allows each bar to have a unique width relative to its neighbors on the X-axis. This technique is often used in specialized visualizations where the width of a bar might represent a second dimension of data, such as the sample size or the duration of an event.

Solution:

1. **Parameter Breakdown:** In the code `plt.bar([1, 2, 3], [10, 20, 30], width=[0.1, 0.2, 0.5])`, the first list defines the X-coordinates, the second list defines the heights, and the third defines the widths. 2. **Mapping the Widths:** The widths are applied sequentially. The bar at $x = 1$ will be very thin (0.1 units wide), the bar at $x = 2$ will be slightly thicker (0.2 units), and the bar at $x = 3$ will be the widest (0.5 units). 3. **Visual Impact:** This results in a chart where the bars do not look uniform. It is important to ensure that the widths do not cause bars to overlap, which can happen if the X-coordinates are too close together and the widths are too large. 4. **Comparison with other charts:** This is distinct from histograms where bar width usually represents an interval. In a bar chart, width is purely a stylistic or secondary data-driven property. Therefore, passing a list allows for individual thickness control.

Final Answer: Passing a list to the `width` parameter assigns different thicknesses to each individual bar.

Answer: (B)

Q4.

Solution

Concept:

In networking, the OSI (Open Systems Interconnection) model divides the complex process of data transmission into seven distinct layers. The Physical Layer (Layer 1) is the lowest layer, dealing with the actual physical medium, such as copper wires or fiber optics, and the raw transmission of bits. Devices operating at this layer lack the ”intelligence” to read data addresses like MAC or IP addresses. Instead, they handle electrical signals or light pulses. A Hub is a classic example of a Layer 1 device. Understanding the difference between Layer 1 devices (like Hubs) and Layer 2 devices (like Switches) is a fundamental topic in Informatics Practices.

Solution:

1. **The Role of a Hub:** A Hub is a non-intelligent network device. When a signal (data) arrives at one of its ports, the Hub does not inspect the destination. Instead, it simply regenerates the signal and broadcasts it to every other port on the device. 2. **Broadcasting Mechanism:** Because it sends data to everyone, it creates heavy network traffic and ”collisions.” Every connected computer receives the data, but only the one with the matching address accepts it; the others discard it. 3. **Comparison with Other Devices:** A **Switch** (Layer 2) is smarter; it learns which MAC address is on which port and sends data only to the specific recipient. A **Router** (Layer 3) works with IP addresses to connect different networks. 4. **Conclusion:** The device that operates at the Physical Layer and performs broadcasting to all ports is the Hub. While mostly replaced by switches in modern networks, they remain an important conceptual topic for understanding basic network topology and data flow.

Final Answer: The device described is a Hub.

Answer: (C)

Q5.

Solution

Concept:

The `ROUND()` function in SQL is used to modify the precision of numerical values. While most users are familiar with rounding to decimal places (e.g., rounding 10.55 to 10.6), SQL allows for rounding to the ”left” of the decimal point by using negative integers as the second argument. This is extremely useful for generating high-level reports where exact figures are less important than ”ballpark” numbers, such as rounding sales to the nearest thousand or populations to the nearest million. The standard mathematical rule applies: if the digit being rounded is 5 or greater, you round up; otherwise, you round down.

Solution:

1. **Evaluating the Arguments:** We have the number 156.78 and the rounding factor -2. The negative sign indicates we are moving to the left of the decimal. 2. **Identifying the Place Value:** A factor of -1 would round to the nearest tens place. A factor of -2 rounds to the nearest **hundreds** place. 3. **Applying the Rounding Rule:** To round to the nearest hundred, we look at the tens digit. In the number 156.78, the tens digit is 5. According to standard rounding rules, since the digit is 5 or greater, we increment the hundreds place. 4. **Final Calculation:** The digit in the hundreds place is 1. Incrementing it by 1 gives us 2. The remaining digits to the right are replaced with zeros. Therefore, 156.78 rounded to the nearest hundred results in 200. This technique simplifies large datasets for executive summaries and dashboard visualizations.

Final Answer: The output of the query is 200.

Answer: (C)

Q6.

Solution

Concept:

Pandas is built on top of the NumPy library, which provides the high-performance multidimensional array objects that allow Pandas to perform complex mathematical operations efficiently. While a Pandas Series provides extra functionality like index labels, alignment, and handling of missing data, there are many scenarios—such as passing data to a machine learning model or performing linear algebra—where you need the ”raw” data without the Pandas metadata. Understanding how to ”unwrap” a Series to get the underlying NumPy array is a fundamental skill. Python programmers must distinguish between attributes that return views of the data and methods that create copies or convert the data into different formats like lists.

Solution:

1. **The `.values` Attribute:** Historically, `.values` has been the primary way to access the underlying data of a Series as a NumPy `ndarray`. It returns the data exactly as it is stored in memory, stripped of its index. 2. **The `.array` Attribute:** Introduced in more recent versions of Pandas, `.array` returns a `PandasArray` or `ExtensionArray`. While similar to NumPy, it is designed to handle Pandas-specific data types (like nullable integers) better than standard NumPy arrays. 3. **The `.tolist()` Method:** This is a function call that converts the Series into a standard Python list. While this makes the data compatible with basic Python functions, it loses the performance benefits of vectorized NumPy operations. 4. **Deprecated Attributes:** The `.data` attribute was used in very early versions of Pandas but has been deprecated for years and should not be used in modern code. 5. **Conclusion:** For the CUET curriculum, `.values` remains the standard answer for retrieving data as a NumPy ndarray.

Final Answer: The property is `S.values`.

Answer: (B)

Q7.

Solution

Concept:

The `ORDER BY` clause is the final step in an SQL query's logical execution, used to sort the result-set based on one or more columns. Sorting is essential for making data readable, such as ranking students by marks or organizing employees by joining date. A critical feature of `ORDER BY` is the ability to perform **multi-level sorting**. This means if two records have the same value in the first sorting column, the database will use the second column to break the tie. By default, SQL sorts in ascending order (`ASC`), so the `DESC` keyword must be explicitly mentioned for descending order.

Solution:

1. **Primary Sort:** The requirement is to sort by `Salary` in descending order (highest to lowest). The correct syntax for this is `Salary DESC`. 2. **Secondary Sort:** If two employees have the same salary, they should be sorted by `Name` in ascending order (A to Z). Since ascending is the default, we can write either `Name ASC` or simply `Name`. 3. **Syntax Construction:** These criteria are combined in the `ORDER BY` clause, separated by a comma. The order matters: the column listed first is the primary sort. 4. **Incorrect Keywords:** Commands like `SORT BY` are used in other environments (like Hive or MapReduce) but are not valid standard SQL. `GROUP BY` is used for aggregation, not for simple sorting of individual rows. 5. **Final Logic:** Combining these rules, the expression `ORDER BY Salary DESC, Name ASC` perfectly addresses the prompt.

Final Answer: The correct clause is `ORDER BY Salary DESC, Name ASC`.

Answer: (A)

Q8.

Solution

Concept:

Matplotlib's pie charts are excellent for showing the relationship of parts to a whole. However, when all slices are joined at the center, it can be difficult to highlight a specific data point. The `explode` parameter solves this by "pulling" a slice away from the center of the pie. This creates a visual emphasis, often used in presentations to draw attention to a "winning" category or a "problem" area. The value provided in the `explode` list is a numerical representation of the distance of the offset. Understanding the scale of this distance is key to preventing the chart from looking distorted or messy.

Solution:

1. **Understanding the Scale:** In Matplotlib, the radius of the pie chart is considered to be 1.0 unit. The values in the `explode` list represent a fraction of this radius. 2. **Calculating the Fraction:** To separate a slice by 20%, we must convert the percentage into a decimal value relative to the whole radius. Calculation: $20/100 = 0.2$. 3. **List Implementation:** If the pie chart has three slices and the user wants to explode only the first one, the parameter would look like `explode=[0.2, 0, 0]`. 4. **Common Errors:** Using a value of 20 would move the slice incredibly far away (20 times the radius of the circle), likely moving it off the screen entirely. Using 0.02 would move it by only 2%, which is barely noticeable to the naked eye. 5. **Conclusion:** A value of 0.2 provides exactly the 20% offset required to achieve the desired visual pop-out effect.

Final Answer: The value should be 0.2.

Answer: (B)

Q9.

Solution

Concept:

Relational Algebra is the theoretical foundation upon which SQL is built. It consists of a set of operations that take one or more relations (tables) as input and produce a new relation as output. The **Selection** operation is one of the most fundamental unary operators. While SQL uses the `WHERE` clause to filter data, Relational Algebra uses the Greek letter sigma (σ) to denote this operation. It is important to distinguish Selection (which filters rows) from Projection (which filters columns), as these two operations form the ”horizontal” and ”vertical” slicing tools of database management.

Solution:

1. **Defining Selection (σ):** Selection identifies a subset of tuples (rows) in a relation that satisfy a specific logic or predicate. It does not change the ”degree” (number of columns) of the table, but it likely reduces the ”cardinality” (number of rows). 2. **Mathematical Notation:** An example would be $\sigma_{Salary>50000}(Employees)$, which returns all employee records where the salary exceeds 50,000. 3. **Horizontal Filtering:** Because it looks at the data row-by-row and decides whether to include the entire row based on a condition, it is known as a horizontal filtering operation. 4. **Contrast with Projection (π):** Projection extracts specific columns (attributes) and ignores others. It changes the degree of the table. 5. **Conclusion:** In the context of the question, Selection is the process of extracting rows that satisfy a specified condition.

Final Answer: Selection extracts rows that satisfy a specified condition.

Answer: (C)

Q10.

Solution

Concept:

Before performing any analysis or machine learning task, a data scientist must understand the structure of the dataset. Pandas provides several inspection methods for this purpose. While `df.head()` shows the data and `df.describe()` provides statistics (like mean and standard deviation), the `df.info()` method is unique because it provides a ”technical” summary of the DataFrame object itself. This is critical for identifying memory constraints, checking if data types are correct (e.g., ensuring a ”Date” column isn't being read as a ”String”), and pinpointing exactly where missing data (NaN) exists in the system.

Solution:

1. **Contents of the Summary:** When `df.info()` is called, it prints the class type, the total number of entries (the index range), and the total number of columns. 2. **Column Metadata:** For every column, it lists the column name, the count of **non-null** (valid) values, and the data type (`int64`, `float64`, `object`, etc.). 3. **Missing Data Identification:** By comparing the ”non-null count” to the ”total entries,” a user can instantly see which columns have missing values that need cleaning. 4. **Memory Usage:** At the very bottom, it displays the total memory footprint of the DataFrame, which is vital when working with large-scale datasets that might exceed the available RAM. 5. **Final Logic:** Unlike `describe()`, which summarizes the values *inside* the columns, `info()` summarizes the *structure* and *integrity* of the columns.

Final Answer: The purpose is to provide a concise summary including non-null counts and memory usage.

Answer: (C)

Q11.

Solution

Concept:

The digital landscape has given rise to sophisticated forms of Cyber Crime designed to exploit technical vulnerabilities rather than human error alone. **Pharming** is a major security threat that falls under the category of ”DNS Poisoning” or ”DNS Spoofing.” Unlike traditional phishing, which requires a user to make a mistake (like clicking a suspicious link), pharming is much more dangerous because it can affect users who are following all the ”correct” steps. It manipulates the fundamental way the internet translates human-readable addresses (URLs) into machine-readable IP addresses, making it a critical topic for students studying network security and societal impacts.

Solution:

1. **The Mechanism:** Pharming works by either installing malicious code on a personal computer (modifying the local ‘hosts‘ file) or by ”poisoning” a DNS server. 2. **The Result:** When a victim types a legitimate website address—such as ‘www.yourbank.com‘—into their browser, the poisoned DNS system redirects the request to a fraudulent IP address controlled by the attacker. 3. **The Deception:** The user sees the correct URL in their address bar, but they are actually interacting with a fake website designed to harvest their login credentials or financial data. 4. **Phishing vs. Pharming:** While Phishing uses ”bait” (like emails) to lure you to a fake site, Pharming ”redirects” you there without your knowledge, even when you type the address yourself. 5. **Conclusion:** Therefore, the redirection to a fraudulent site despite typing the correct URL is the defining characteristic of Pharming.

Final Answer: The type of crime is Pharming.

Answer: (B)

Q12.

Solution

Concept:

SQL aggregate functions are used to perform mathematical calculations on a group of values, resulting in a single summary value. These functions are indispensable for data analysis, allowing users to find totals, averages, and counts across thousands of rows. However, a major point of confusion—and a frequent exam topic—is how these functions handle **NULL values**. NULL represents missing or unknown data. Some functions ignore NULLs entirely, while others include them depending on the syntax used. Mastering these distinctions is vital for ensuring that database reports reflect the true state of the data.

Solution:

1. `COUNT(*):` This is a special version of the count function. It counts every single row in the table, regardless of whether the columns contain NULL values or duplicates. If a row exists, it is counted. This matches description (b). 2. `COUNT(column_name):` Unlike the asterisk version, this specific call counts only the rows where the named column has a non-null value. It effectively measures the ”completeness” of that column. This matches description (d). 3. `AVG():` This function calculates the mean. Crucially, it ignores NULL values in both the sum and the count of items, which avoids skewing the average. This matches description (a). 4. `SUM():` This simply adds up all the numerical values in a column, ignoring any NULL entries. This matches description (c). 5. **Matching Sequence:** (i)-b, (ii)-d, (iii)-a, (iv)-c.

Final Answer: The correct match is (i)-b, (ii)-d, (iii)-a, (iv)-c.

Answer: (A)

Q13.

Solution

Concept:

Matplotlib is designed to be highly customizable, following a philosophy that every part of a chart should be accessible and modifiable by the programmer. Axis labels are one of the most important components of any visualization, as they provide the necessary context to understand what the data represents. Without clear labels, a histogram is just a series of bars without meaning. In Matplotlib's 'pyplot' interface, labeling functions follow a simple and intuitive naming convention: 'set' or 'label' followed by the axis name. For students, remembering these specific function names is key to writing bug-free visualization code.

Solution:

1. **The Correct Function:** The standard function provided by the 'matplotlib.pyplot' module to label the vertical axis is `plt.ylabel()`. 2. **Usage:** It takes a string as an argument, which is then rendered next to the Y-axis. For example, `plt.ylabel("Frequency")` will place that word vertically along the left side of the chart. 3. **X-Axis Counterpart:** Similarly, `plt.xlabel()` is used for the horizontal axis to describe the categories or intervals being measured. 4. **Incorrect Alternatives:** Functions like `plt.y_axis()` or `plt.label_y()` do not exist in the standard Matplotlib library. While the Object-Oriented interface uses `ax.set_ylabel()`, the question asks for the standard 'plt' approach used in basic scripts. 5. **Conclusion:** Therefore, `plt.ylabel()` is the only correct command for adding a Y-axis label.

Final Answer: The function used is `plt.ylabel("Frequency")`.

Answer: (B)

Q14.

Solution

Concept:

Data cleaning is often the most time-consuming part of data science. Real-world datasets are rarely perfect and frequently contain ”missing values,” represented in Pandas as ‘NaN‘ (Not a Number). These gaps can break mathematical calculations or cause errors in machine learning models. There are two primary strategies for handling missing data: ”Deletion” (removing the rows) or ”Imputation” (filling the gaps with a logical value). Pandas provides specific methods for both. The `fillna()` method is the go-to tool for imputation, allowing a user to maintain the size of their dataset while making it computationally ”complete.”

Solution:

1. **The `fillna()` Method:** This function is specifically designed to locate all occurrences of ‘NaN‘ within a DataFrame or Series and replace them with a value provided by the user. 2. **Implementation:** By executing `df.fillna(0)`, every missing cell in the DataFrame is instantly populated with the integer 0. This is useful when a missing value logically implies a zero count or neutral state. 3. **The `inplace` Parameter:** Like many Pandas methods, `fillna()` returns a copy of the DataFrame by default. To modify the original DataFrame permanently, one must use `df.fillna(0, inplace=True)`. 4. **Contrast with `dropna()`:** This method would delete any row containing a NULL value. If you have a dataset of 100 rows and 10 have NULLs, `dropna()` leaves you with 90 rows, whereas `fillna()` keeps all 100. 5. **Conclusion:** For filling missing values with a constant like 0, `fillna(0)` is the correct and most efficient approach.

Final Answer: The correct method is `df.fillna(0)`.

Answer: (B)

Q15.

Solution

Concept:

In SQL, string processing often involves ”cleaning” data before measuring it. Leading and trailing spaces are a common source of error in database management; they might be accidentally added during data entry and can interfere with string comparisons (e.g., ‘’ CUET ’‘ does not equal ‘’CUET’‘). To handle this, SQL provides the `TRIM()` function. Additionally, the `LENGTH()` function is used to calculate the number of characters in a string. When these functions are ”nested” (one placed inside another), SQL follows a specific order of operations: it evaluates the innermost function first and passes that result to the outer function.

Solution:

1. **Inner Evaluation:** The first step is to evaluate `TRIM(' CUET ')`. The original string contains two leading spaces, the four letters of ”CUET”, and two trailing spaces, for a total of 8 characters. 2. **Trimming Action:** The `TRIM()` function identifies and removes the two spaces at the start and the two spaces at the end. The result of this inner operation is the clean string `'CUET'`. 3. **Outer Evaluation:** Now, the `LENGTH()` function is applied to the result of the first step. It receives the string `'CUET'`. 4. **Counting Characters:** The function counts the characters: C(1), U(2), E(3), T(4). 5. **Final Result:** The count is 4. This demonstrates how nesting functions can be used to perform complex data cleaning and validation in a single SQL statement.

Final Answer: The output of the query is 4.

Answer: (C)

Q16.

Solution

Concept:

Pandas leverages **Vectorized Operations** and **Boolean Indexing** to perform data filtering and arithmetic in a single, highly efficient step. Boolean indexing works by applying a logical condition to every element in a Series, creating a "Mask" of True and False values. When this mask is applied back to the Series, only the True entries remain. Any subsequent mathematical operations are then applied only to those selected entries. This "filtering-then-calculating" workflow is much faster than using standard Python loops and is a core part of the Informatics Practices syllabus.

Solution:

1. **The Filter:** The condition `S > 30` is applied to the Series `[10, 20, 30, 40, 50]`. This returns `False` for the first three values and `True` for the values 40 and 50. 2. **The Selection:** Applying `S[S > 30]` extracts only the elements at indices 3 and 4. The resulting temporary Series is:

- Index 3: 40
- Index 4: 50

3. **The Operation:** The instruction `* 2` is then applied to this subset.

- $40 \times 2 = 80$
- $50 \times 2 = 100$

4. **Final Output:** Pandas preserves the original indices during this operation. Therefore, the result shows index 3 with 80 and index 4 with 100. Options that only show values without indices are technically incorrect for a Series output.

Final Answer: The result is `3: 80, 4: 100`.

Answer: (A)

Q17.

Solution

Concept:

Data transposition is the geometric rotation of a dataset where the rows become columns and the columns become rows. This is a common requirement in data analysis when the current orientation of a DataFrame makes it difficult to apply certain functions (like `describe()` or `plot()`). In Pandas, a DataFrame is conceptually a 2D table; transposing it essentially flips the table over its main diagonal. This is a "Metadata" change, meaning the underlying data points remain the same, but their associated labels and positions are swapped.

Solution:

1. **The `.T` Property:** Pandas provides a very convenient, single-character property called `.T` (short for Transpose). It is an attribute, not a function, so it does not require parentheses. 2. **Usage:** If you have a DataFrame `df` with 5 rows and 2 columns, executing `df.T` will instantly yield a new DataFrame with 2 rows and 5 columns. 3. **Alternative Method:** There is also a full method called `df.transpose()`, which does the exact same thing but allows for more complex parameters (though these are rarely used in basic data science). 4. **Comparison:** `df.pivot()` is used for reshaping data into a summary format, and `df.swaplevel()` is used for multi-index DataFrames. Neither performs a simple global transposition. 5. **Conclusion:** For a quick row-to-column swap, the property `df.T` is the standard and most concise solution.

Final Answer: The property used is `df.T`.

Answer: (B)

Q18.

Solution

Concept:

The `SUBSTR()` (or `SUBSTRING()`) function in SQL is used to slice a piece of text from a larger string. This is vital for data extraction tasks, such as pulling a specific year from a date string or an area code from a phone number. To use it, you must provide the string, the starting index, and the number of characters to extract. A critical distinction for students is that SQL uses **1-based indexing**. Unlike Python, where the first character is at index 0, SQL considers the first character to be at index 1.

Solution:

1. **Identify the Input:** The source string is ”INFORMATICS”. 2. **Identify the Start Position:** The second argument is 3. We count characters: I(1), N(2), F(3). So, the extraction begins at the character ’F’. 3. **Identify the Length:** The third argument is 4, which means we extract four characters starting from ’F’. 4. **The Extraction:**

- 1st char: F
- 2nd char: O
- 3rd char: R
- 4th char: M

5. **Conclusion:** The resulting substring is ”FORM”. If the length argument were omitted, the function would return everything from the 3rd character to the end of the string.

Final Answer: The output is `FORM`.

Answer: (B)

Q19.

Solution

Concept:

A **Histogram** is a visualization of the frequency distribution of a numeric variable. By default, Matplotlib looks at your entire dataset and automatically creates bins that cover the full range from the minimum value to the maximum value. However, in many real-world scenarios, you might only be interested in a specific portion of the data (e.g., viewing the distribution of "Passing Scores" only). The `plt.hist()` function provides parameters to manually override the automatic scaling and binning behavior, allowing for highly targeted data visualization.

Solution:

1. **Defining Intervals:** The `bins` parameter specifies how many vertical bars should appear in the chart. Setting `bins=10` creates ten intervals. 2. **Setting the Data Filter:** The `range` parameter is used to define the lower and upper limits of the bins. It accepts a tuple in the format `(min, max)`. 3. **Parameter Combination:** To show 10 bins between the values of 50 and 100, the syntax is `plt.hist(data, bins=10, range=(50, 100))`. 4. **Execution Details:** Any data points in your dataset that are less than 50 or greater than 100 will be ignored by this specific plot. 5. **Incorrect Options:** `limit` and `scale` are not valid parameters for `plt.hist()`. Standardizing on the term `range` is essential for accurate Matplotlib programming.

Final Answer: The correct combination is `plt.hist(data, bins=10, range=(50, 100))`.

Answer: (B)

Q20.

Solution

Concept:

In computer networking, a PAN (Personal Area Network) is the smallest category of network in terms of geographic scale. It is designed for communication between devices belonging to a single individual, typically within a very small radius (usually less than 10 meters). The goal of a PAN is to eliminate the need for wires when connecting personal peripherals. Modern life is heavily reliant on PANs, primarily through the use of Bluetooth technology. Understanding the hierarchy of networks—from PAN to LAN, MAN, and WAN—is a foundational requirement for the networking unit of Informatics Practices.

Solution:

Scale of PAN: A PAN connects devices like smartphones, tablets, smartwatches, and wireless headphones. Because these devices are usually on the person or on a desk nearby, the range is strictly ”Personal.”

Primary Technology: While USB cables can create a wired PAN, wireless PANs (WPANs) are far more common, utilizing Bluetooth or Zigbee.

Comparison with Others:

- **LAN:** Covers a home or office building.
- **MAN:** Covers a metropolitan area or city.
- **WAN:** Covers global distances (the Internet).

Conclusion: Since the question specifies a very short distance ”within a range of an individual person,” the network is classified as a PAN.

Final Answer: The network is a PAN.

Answer: (C)

Q21.

Solution

Concept:

Filtering aggregated data is a common requirement in data analysis. In SQL, it is critical to understand that the `WHERE` clause and the `HAVING` clause serve two different purposes in the logical order of query execution. The `WHERE` clause filters individual rows before any grouping occurs. Conversely, the `HAVING` clause is used specifically to filter groups after the `GROUP BY` clause has been processed. Because aggregate functions like `AVG()`, `SUM()`, or `COUNT()` operate on groups rather than single rows, they cannot be placed inside a `WHERE` clause. This distinction is one of the most frequently tested logic points in the CUET IP syllabus.

Solution:

1. **Grouping Strategy:** To find the average marks "of each subject," we must first group the data using `GROUP BY Subject`. 2. **Aggregation:** Within each created group, we calculate the average using `AVG(Marks)`. 3. **Applying the Filter:** The condition is that this average must be greater than 75. Since `AVG(Marks)` is an aggregate result, we must use the `HAVING` keyword to apply this filter. 4. **Syntax Check:** The correct SQL structure is: `SELECT [Columns] FROM [Table] GROUP BY [Column] HAVING [Aggregate Condition]`. 5. **Incorrect Options:** Option A is invalid because it tries to use an aggregate function in the `WHERE` clause. Option C is invalid because `WHERE` cannot follow `GROUP BY`. 6. **Conclusion:** Option B provides the syntactically correct and logical approach for grouping by subject and then filtering based on the group's average.

Final Answer: The correct query is `SELECT Subject, AVG(Marks) FROM EXAM GROUP BY Subject HAVING AVG(Marks) > 75;`.

Answer: (B)

Q22.

Solution

Concept: In Pandas, different methods are used for combining datasets: * **pd.merge():** Joins DataFrames on common columns or indices (similar to SQL JOIN). * **pd.concat():** Stacks DataFrames along a particular axis (rows or columns). * **pd.append():** Specifically adds rows to the end of a DataFrame (deprecated in newer versions).

Solution: To combine two DataFrames side-by-side based on a common key (like an ID or Name), the `merge()` function is used. It supports "inner", "left", "right", and "outer" joins.

Syntax: pd.merge(df1, df2, on='key_column')

Thus, the method that replicates SQL-like joining logic is `pd.merge()`.

Answer: (C)

Q23.

Solution

Concept:

Pandas DataFrames come with built-in attributes that provide essential metadata about the object's structure. These attributes are "properties," meaning they do not require parentheses to call and they return a snapshot of the DataFrame's dimensions, data volume, or types. For students, it is vital to distinguish between the **shape** (which describes the layout) and the **size** (which describes the quantity of data). Mistakes here lead to errors in loop logic or memory management during large-scale data processing.

Solution:

1. `df.shape:` Returns a tuple representing the number of rows and columns (e.g., `(100, 5)`). This is the "blueprint" of the DataFrame. Matches (b). 2. `df.size:` Returns the total number of individual elements or cells in the DataFrame. It is calculated as $Rows \times Columns$. Matches (a). 3. `df.ndim:` Returns the number of dimensions. For a DataFrame, this is always 2 (representing a table), whereas a Series is 1. Matches (d). 4. `df.dtypes:` Returns the data type (`int`, `float`, `object`) of every column in the DataFrame. Matches (c). 5. **Logic Sequence:** Following these definitions, we find that (i)-b, (ii)-a, (iii)-d, and (iv)-c is the correct logical pairing.

Final Answer: The correct match is (i)-b, (ii)-a, (iii)-d, (iv)-c.

Answer: (A)

Q24.

Solution

Concept:

The `MOD()` function in SQL (which stands for Modulo) is used to find the remainder of a division operation between two numbers. While standard division (using the `/` operator) returns the quotient (how many times a number fits), the modulo operator tells us what is "left over." This is a foundational concept in computer science used for various logic checks, such as determining if a number is even or odd (checking `MOD(n, 2)`), cyclically iterating through lists, or managing time and date conversions.

Solution:

1. **Mathematical Calculation:** We are dividing the dividend (11) by the divisor (4). 2. **Finding the Quotient:** 4 fits into 11 exactly twice ($4 \times 2 = 8$). It cannot fit a third time because $4 \times 3 = 12$, which exceeds 11. 3. **Subtracting to find Remainder:** We subtract the product of the quotient and the divisor from the original number: $11 - 8 = 3$. 4. **Result:** The value "left over" after the division is 3. 5. **SQL Syntax:** In many SQL versions, this can also be written as `11 % 4`. However, `MOD(11, 4)` is the standard function call taught in the Informatics Practices syllabus.

Final Answer: The function returns 3.

Answer: (C)

Q25.

Solution

Concept:

Titles are essential for making charts informative and self-explanatory. In Matplotlib, the `pyplot` module provides simple, high-level functions to add text elements to specific parts of a figure. The title typically appears centered at the top of the chart and summarizes the purpose of the visualization. Because Matplotlib has both a "State-based" interface (`pyplot`) and an "Object-Oriented" interface, students must be careful with syntax. While the Object-Oriented approach uses `set_title()`, the standard `plt` interface—which is the focus of the CUET curriculum—uses a simpler function name.

Solution:

1. **The Correct Function:** In the `pyplot` module, the standard command to add a main title to the current active chart is `plt.title()`. 2. **Syntax:** The function takes a string as an argument, for example: `plt.title("Growth Analysis")`. 3. **Customization:** You can pass additional arguments to this function to change the font size, color, or alignment, such as `plt.title("Growth Analysis", fontsize=14, color='red')`. 4. **Incorrect Alternatives:** `plt.set_title()` is a common distractor; it is a method used on an *Axes* object, not a direct call on `plt`. `header` and `chart_title` are not valid Matplotlib functions. 5. **Conclusion:** `plt.title()` is the industry-standard way to label the top of your plots in Matplotlib scripts.

Final Answer: The correct command is `plt.title("Growth Analysis")`.

Answer: (B)

Q26.

Solution

Concept:

Intellectual Property Rights (IPR) are the legal protections granted to creators for their ”creations of the mind.” In the digital age, IPR has become a central topic in ethics and law because digital content is incredibly easy to copy, modify, and distribute without permission. IPR covers a broad spectrum, including inventions (patents), literary and artistic works (copyright), and symbols or names used in commerce (trademarks). Understanding the different ways these rights can be violated is essential for the ”Societal Impacts” unit, which emphasizes the legal and ethical responsibilities of digital citizens.

Solution:

1. **Plagiarism:** This is the act of taking someone else's ideas, words, or creative work and presenting them as your own. While it is often an academic violation, it is fundamentally an infringement on the creator's intellectual rights to their original thought. 2. **Trademark Infringement:** This occurs when a person or business uses a logo, name, or design that is identical or confusingly similar to a registered trademark owned by someone else. This is a violation of commercial IPR. 3. **Copyright Infringement:** This is the unauthorized use of works protected by copyright law, such as downloading movies illegally, using copyrighted music in a video without a license, or distributing software (piracy). 4. **Conclusion:** Since each of these represents a specific way in which intellectual property is misused or stolen, they are all classified as IPR violations. This makes ”All of the above” the logically correct answer.

Final Answer: All of the above.

Answer: (D)

Q27.

Solution

Concept:

The `LIKE` operator in SQL is a powerful tool for pattern matching, allowing users to search for data that matches a specific "shape" rather than a precise value. This is useful for finding names starting with a certain letter or identifying codes with a specific format. To define these patterns, SQL uses two primary wildcard characters: the percent sign (%) and the underscore (_). Mastering these wildcards is critical because it allows for flexible querying of large databases where exact string matches are not always known or practical.

Solution:

1. **The Percent Wildcard (%):** This character represents a substitute for zero, one, or multiple characters. For example, `'A%'` matches "Apple", "Ant", or just "A". 2. **The Underscore Wildcard (_):** Unlike the percent sign, the underscore is a "placeholder" for exactly one single character. It is the SQL equivalent of a single empty slot that must be filled. 3. **Example Walkthrough:** If you search for `WHERE Name LIKE 'S_m'`, the query will return "Sam" or "Sim", but it will not return "Slim" (because 'li' is two characters) or "S" (because there is no character to fill the underscore). 4. **Incorrect Options:** In many programming languages or operating systems, `*` or `?` are used as wildcards, but in standard SQL pattern matching with `LIKE`, they have no special meaning and are treated as literal characters. 5. **Conclusion:** For matching exactly one character, the underscore (_) is the only correct wildcard.

Final Answer: The wildcard is the underscore (_).

Answer: (C)

Q28.

Solution

Concept:

A key feature of Pandas DataFrames is their dynamic nature, allowing users to add, modify, or delete columns with ease. When adding a new column, the most common method is using the "dictionary-style" bracket notation. A powerful aspect of this operation is "Broadcasting." In Pandas, if you assign a single scalar value (like a number or a string) to a column, Pandas automatically "broadcasts" or repeats that value across every single row in the DataFrame. This is an efficient way to initialize data or set default categories without manually creating a list of matching length.

Solution:

1. **The Syntax:** By writing `df['Grade'] = 'A'`, you are telling Pandas to create a new column named 'Grade'. If that column already exists, Pandas will overwrite its data; if it doesn't exist, Pandas adds it to the end of the DataFrame. 2. **Broadcasting in Action:** Since 'A' is a single string, and the DataFrame has multiple rows, Pandas understands that every row should receive the value 'A'. This eliminates the need for loops or complex logic. 3. **Comparison with Attribute Access:** While `df.Grade = 'A'` might work for updating an existing column, it is generally discouraged for creating new columns because it can lead to ambiguity (e.g., if 'Grade' is also the name of a built-in DataFrame method). 4. **Advanced Methods:** `df.insert()` can be used to add a column at a specific position (like at index 0), but it requires more arguments and is less concise for simple additions. 5. **Conclusion:** For adding a new column with a uniform default value, the bracket notation used in Option A is the standard industry practice.

Final Answer: The correct command is `df['Grade'] = 'A'`.

Answer: (A)

Q29.

Solution

Concept:

Network topology refers to the geometric arrangement of computers and cables in a network. Choosing the right topology is critical because it affects the network's cost, reliability, and ease of maintenance. The **Star Topology** is currently the most popular choice for modern Local Area Networks (LANs). In this setup, every individual node (computer, printer, etc.) is connected directly to a central connecting point. This centralized design offers significant advantages in troubleshooting and scalability compared to older topologies like Bus or Ring, which are more susceptible to complete network failure.

[Image of star network topology]

Solution:

1. **Central Controller:** The defining feature of a Star topology is the central Hub or Switch. All data transmitted between nodes must first pass through this central device, which then directs it to the correct destination. 2. **Fault Tolerance:** A major benefit of this arrangement is that if one cable or one computer fails, it does not affect the rest of the network. Only the connection between that specific node and the hub is broken. 3. **Comparison with Others:**

- **Bus Topology:** All nodes share a single main cable. If the cable breaks, the whole network dies.
- **Ring Topology:** Data travels in a circle. One failure stops the entire loop.
- **Mesh Topology:** Every node is connected to every other node. It is very reliable but extremely expensive and complex to wire.

4. **Conclusion:** Because the question describes a setup with a "central controller or Hub," the answer is the Star topology.

Final Answer: The topology is the Star topology.

Answer: (C)

Q30.

Solution

Concept:

In Matplotlib, a pie chart without numeric values can be difficult to interpret, as the human eye is not always good at judging the relative area of slices. To make a pie chart data-driven and precise, we use the `autopct` parameter. This parameter is short for ”automatic percentage.” It allows Matplotlib to calculate the percentage contribution of each slice relative to the total and then display that number directly on top of the corresponding wedge. This is achieved using a specialized string formatting syntax that determines the precision and style of the displayed numbers.

Solution:

1. **How it Works:** When you pass data to `plt.pie(x)`, Matplotlib sums the values in `x`. It then calculates what percent of that sum each individual value represents. 2. **String Formatting:** The argument `'%1.1f%%'` is a format string. The `1.1f` tells Python to display the number as a float with one decimal place. The double percent sign `%%` at the end is used to render a literal ’3. **Visual Presentation:** The resulting percentages are automatically placed in the middle of each wedge, though their position can be further adjusted with parameters like `pctdistance`. 4. **Exclusion of Other Effects:** `autopct` does not change the color of the slices (handled by `colors`) or pull the slices apart (handled by `explode`). 5. **Conclusion:** The primary and only role of `autopct` is to display the calculated percentage values within the wedges of the pie chart.

Final Answer: It displays the percentage value inside the wedges.

Answer: (B)

Q31.

Solution

Concept:

Handling temporal data is a core requirement for any database management system. SQL provides several built-in functions to capture the current state of the system's clock. These functions are essential for "logging" purposes, such as recording the exact second a transaction was made or calculating the age of a record. While functions like `CURDATE()` focus only on the calendar date, and `CURTIME()` focuses only on the time of day, the most frequently used function in the CUET IP syllabus for capturing the full picture is `NOW()`. Understanding how this function outputs data (usually in the format 'YYYY-MM-DD HH:MM:SS') is vital for database developers.

Solution:

1. **The `NOW()` Function:** This function is designed to return the current system date and time simultaneously. It is highly efficient because it allows a single function call to populate both date and time fields in a database table. 2. **The `SYSDATE()` Alternative:** In many SQL systems (like MySQL), `SYSDATE()` also returns the current date and time. The subtle difference is that `NOW()` returns the time at which the query started, while `SYSDATE()` returns the exact time the function was executed. For the purposes of the CUET curriculum, `NOW()` is the standard answer.
3. **Comparison with Others:**

- **CURDATE()**: Returns only the date (e.g., '2023-10-25').
- **DATE()**: This is a conversion function used to extract the date portion from an existing datetime expression; it doesn't fetch the system clock itself.

4. **Conclusion:** Because the question explicitly asks for both date and time, `NOW()` is the correct and most complete function.

Final Answer: The SQL function is `NOW()`.

Answer: (C)

Q32.

Solution

Concept:

The `read_csv()` function is perhaps the most utilized function in the Pandas library, serving as the gateway for importing external data into Python for analysis. CSV (Comma Separated Values) files are the industry standard for data exchange because they are simple, text-based, and readable by almost any software. However, because CSV files don't contain metadata about their own structure, Pandas has to make certain "default assumptions" when it reads a file. One of the most important assumptions is how to identify the names of the columns. Understanding these defaults is key to avoiding errors where data rows are accidentally treated as column headers.

Solution:

1. **Default Header Behavior:** By default, `read_csv()` assumes that the first row (line 0) of your file contains the labels for the columns. This is technically controlled by the parameter `header=0`.
2. **Visual Impact:** When you load the data, Pandas will take the strings found in that first row and use them as the DataFrame's column names. The actual numerical data begins from the second row of the file. 3. **Overriding the Default:** If your CSV file does not have a header row and starts with data immediately, you must specify `header=None`. If you don't, Pandas will incorrectly turn your first row of data into column titles, effectively losing one record of data. 4. **Comparison:** The function does not automatically assume the first row is an index (this must be set with `index_col`) or metadata. 5. **Conclusion:** Therefore, the default interpretation for the first row of a CSV file is that it contains the Column Names.

Final Answer: The first row is considered as the Column Names (Header).

Answer: (C)

Q33.

Solution

Concept:

As our lives move increasingly online, the nature of harassment and crime has evolved. ”Cyber Stalking” is a significant digital threat covered in the ”Societal Impacts” unit of Informatics Practices. It is characterized by the use of technology to harass, intimidate, or monitor a person. Unlike a one-time incident of rudeness or spam, cyber stalking is defined by its **persistence** and **intent to cause distress**. Laws regarding cyber stalking have become much stricter globally, as the psychological impact on victims can be devastating. Students must be able to distinguish this targeted harassment from more general crimes like identity theft or electronic fraud.

Solution:

1. **Defining Characteristics:** Cyber stalking involves a pattern of behavior. This might include sending threatening emails, monitoring a victim’s social media activity without their consent, spreading false rumors online, or ”doxxing” (releasing private information). 2. **Digital Means:** The stalker uses the anonymity and reach of the internet (social media, messaging apps, emails) to maintain contact with the victim, often bypassing traditional physical boundaries. 3. **Distinction from Other Crimes:**

- **Identity Theft**: Focuses on financial gain by using someone’s personal details.
- **Spamming**: Focuses on mass-sending unwanted commercial content, not targeting an individual for emotional distress.
- **Phishing**: A deceptive technique to steal passwords.

4. **Conclusion:** Therefore, the repeated harassment or threatening of a person through digital means is the correct definition of Cyber Stalking.

Final Answer: It is harassing or threatening a person through digital means repeatedly.

Answer: (B)

Q34.

Solution

Concept:

SQL queries often require complex arithmetic that goes beyond simple addition or subtraction. To handle this, SQL provides advanced mathematical functions like `POWER()` and `ROUND()`. A crucial skill for the CUET IP exam is the ability to solve "nested" functions, where the result of one function serves as the input to another, or multiple functions are combined in a single arithmetic expression. This requires a clear understanding of the "Order of Operations" and how specific functions handle rounding—particularly the "rounding up" behavior for values ending in .5.

Solution:

1. **Evaluating the First Term:** The expression starts with `POWER(3, 2)`. This function raises the first argument to the power of the second. Calculation: $3^2 = 3 \times 3 = 9$. 2. **Evaluating the Second Term:** Next, we have `ROUND(1.5)`. In SQL, if the second argument for decimal places is missing, it defaults to rounding to the nearest integer. Since the value is exactly 1.5, the function follows the standard mathematical rule and rounds **up** to the next whole number. Calculation: $1.5 \rightarrow 2$. 3. **Final Addition:** Now we combine the results of the two functions as instructed by the plus sign in the query. Calculation: $9 + 2 = 11$. 4. **Why not 10?** A common mistake is to round 1.5 down to 1 or to confuse the exponent with multiplication ($3 \times 2 = 6$). Precision in these basic steps is key to solving SQL math problems.

Final Answer: The output of the query is 11.

Answer: (A)

Q35.

Solution

Concept:

DataFrames are often created with default labels (like integers 0, 1, 2...) or with raw names from a database that might not be user-friendly. To make data more readable and professional, we often need to ”rename” these labels. Pandas offers several ways to modify these identifiers. It is important for students to differentiate between the **labels** (the names of the rows/columns) and the **data** itself. The `rename()` method is the industry standard because it allows for granular control—you can rename just one specific row without having to redefine the entire index.

Solution:

1. **The `rename()` Method:** This function is designed specifically to change the labels of the axes. It uses a ”dictionary-based” mapping approach, where you provide the old name as the key and the new name as the value: `{'old_name': 'new_name'}`. 2. **Targeting the Index:** To change row names, you use the `index` parameter within the function: `df.rename(index={0: 'Row1'})`. 3. **Immutability:** By default, `rename()` returns a new, modified DataFrame and leaves the original one unchanged. To apply the change to the original DataFrame directly, the user must set `inplace=True`. 4. **Comparison with `reindex()`:** A common point of confusion is `reindex()`, but this is used to change the **order** of existing indices or to add missing ones; it cannot change the text of an existing label while keeping its data. 5. **Conclusion:** For the task of renaming row labels, `df.rename()` is the correct and most flexible method.

Final Answer: The method is `df.rename()`.

Answer: (C)

Q36.

Solution

Concept:

A **Histogram** is the standard tool in data science for visualizing the ”shape” of a continuous numerical variable. While it looks similar to a bar chart, its logic is entirely different. In a bar chart, each bar represents a distinct category (like ”Apple” or ”Orange”). In a histogram, each bar represents a numeric interval called a **bin**. The chart aggregates data by counting how many data points fall into each interval. This provides a visual representation of data density, allowing a researcher to identify where most values are concentrated (the mode), the range of the data, and whether the distribution is symmetric or skewed.

Solution:

1. **X-Axis (Horizontal):** This axis represents the bins or ranges (e.g., ages 0–10, 10–20, etc.). It represents the value of the variable being measured. 2. **Y-Axis (Vertical):** This axis represents the **Frequency**. The height of each rectangular bar corresponds directly to the count of data points—the number of times a value from the dataset falls into that specific bin. 3. **Visualization Logic:** If a bar for the ”60–70” score range has a height of 15, it implies that 15 students in the class scored between 60 and 70. 4. **Incorrect Alternatives:** The height does not represent the ”mean” (average) or the ”sum” of values; it is strictly a headcount (frequency) of occurrences within the interval. 5. **Conclusion:** Understanding that height equals frequency is fundamental to correctly interpreting histograms in statistical analysis.

Final Answer: The height represents the frequency (count) of data points in that bin.

Answer: (B)

Q37.

Solution

Concept: Understanding Data Manipulation Language (DML) vs. Data Definition Language (DDL) commands for data removal: * **DROP:** Deletes the entire table structure and all data from the database. * **DELETE:** A DML command used to remove specific rows (with a WHERE clause) or all rows; it is slower as it logs individual row deletions. * **TRUNCATE:** A DDL command that removes all rows from a table by deallocating the data pages, making it faster than DELETE. It keeps the table structure for future use.

Solution: The command used to remove all records while keeping the table structure intact is `TRUNCATE TABLE`. Unlike `DELETE`, it does not trigger table delete triggers and usually resets identity columns.

Command: TRUNCATE TABLE table_name;

Hence, option (C) is the correct method for this specific requirement.

Answer: (C)

Q38.

Solution

Concept: Data security involves several processes to protect information from unauthorized access:

- **Encryption:** The process of encoding information so that only authorized parties can access it. It converts *Plaintext* into *Ciphertext*.
- **Decryption:** The reverse process of encryption; converting unreadable ciphertext back into readable plaintext.
- **Masking:** Hiding specific data within a database (like the middle digits of a credit card) to protect privacy while maintaining the data format.
- **Firewalls:** Network security systems that monitor and control incoming and outgoing network traffic based on predetermined security rules.

Solution: The transformation of data into an unreadable format is specifically defined as encryption. This ensures that even if an unauthorized person intercepts the data, they cannot understand its content without the correct cryptographic key.
Conclusion: Since the objective is to prevent unauthorized access by making text unreadable, the process is called Encryption.
Final Answer: The process is called Encryption.

Answer: (B)

Q39.

Solution

Concept:

The `.iloc` indexer in Pandas is one of the most powerful tools for data selection. It stands for "Integer Location" and allows you to slice a DataFrame based on the numerical position of the rows and columns, starting from zero. A critical feature of `.iloc` is its support for Negative Indexing. Just like standard Python lists, Pandas allows you to count from the end of the DataFrame using negative numbers. This is extremely useful when you are working with datasets of unknown size and need to grab the "very last" piece of information.

Solution:

The Slicing Syntax: The indexer uses the format `[row_range, column_range]`.

Row Range (`:`): Using a colon by itself indicates "all rows." This means the operation will return data for every single record in the DataFrame.

Column Index (`-1`): In Python, the index `-1` refers to the last element in a sequence. By using `-1` in the column spot, we are targeting the final column of the DataFrame.

Resulting Object: The combined command `df.iloc[:, -1]` results in a Series containing every row for the last column. This is a common step in machine learning for isolating the "labels" or "target" column from the rest of the data.

Conclusion: Because we are selecting all rows for the very last integer position in the columns, the output is the last column.

Final Answer: The command returns the last column.

Answer: (C)

Q40.

Solution

Concept:

Data visualization is as much about communication as it is about math. When a chart contains multiple data series (for example, a line showing ”Revenue” and another showing ”Expenses”), the viewer needs a ”Key” to understand which line represents which category. In Matplotlib, this key is called a Legend. Adding a legend is a two-step process: first, assigning a label to the data when it is plotted, and second, calling the legend function to draw the box on the figure. This ensures that the chart is self-explanatory and professional.

Solution:

Creating Labels: When you call a plotting function like `plt.plot(x, y)`, you include a string in the `label` parameter (e.g., `label="Sales"`).

Activating the Box: Simply adding a label is not enough; the legend box will not appear on the screen unless you explicitly run the `plt.legend()` command.

Functionality: This command automatically searches for any plotted elements that have a label and creates a small descriptive box on the plot area.

Placement: By default, Matplotlib tries to find the ”best” location to avoid overlapping with data points, but you can manually set the location using the `loc` parameter.

Conclusion: Therefore, `plt.legend()` is the specific function required to add this explanatory guide to your chart.

Final Answer: The function is `plt.legend()`.

Answer: (B)

Q41.

Solution

Concept:

Advanced data retrieval in SQL often requires combining several clauses to transform raw data into a summarized report. The logic follows a strict order of operations: **Grouping**, **Aggregation**, and **Sorting**. The `GROUP BY` clause organizes rows into summary rows based on a specific column (the "Region"). Once grouped, an aggregate function like `SUM()` calculates a total for each group. Finally, the `ORDER BY` clause is applied to these results to arrange them in a specific sequence. For competitive exams, understanding that the sorting must happen *after* the calculation is key to choosing the right query result.

Solution:

1. **Grouping and Summing:** The query identifies three distinct regions in the table: North, South, and East.

- **North:** Contains two entries (5000 and 3000). The sum is 8000.
- **South:** Contains one entry (7000). The sum is 7000.
- **East:** Contains one entry (4000). The sum is 4000.

2. **Sorting (`DESC`):** The `ORDER BY SUM(Amount) DESC` clause sorts these totals from highest to lowest. 3. **Result Sequence:** The highest total is 8000 (North), followed by 7000 (South), and finally 4000 (East). 4. **Incorrect Options:** Option B fails to group the data. Option C provides the correct groups but in the wrong order (South before North). 5. **Conclusion:** The query successfully aggregates the sales by region and presents them in descending order of total revenue.

Final Answer: The output is North 8000, South 7000, East 4000.

Answer: (A)

Q42.

Solution

Concept:

In Matplotlib, a **Histogram** is designed to visualize the distribution of a dataset. By default, the `bins` parameter takes an integer, which instructs the library to divide the range of data into that many equal-width segments. However, real-world data distribution is often non-uniform, and a researcher may want to look at specific "interest zones" with more or less detail. To achieve this, Matplotlib allows the `bins` parameter to accept a **list of edges**. This provides complete control over where each interval starts and ends, essentially allowing for the creation of custom, unequal-width bins.

Solution:

1. **Defining the Edges:** When `bins=[0, 10, 25, 50, 100]` is used, the intervals are calculated between the adjacent numbers in the list. 2. **Resulting Intervals:** The data is grouped into four distinct bins: $[0 - 10]$, $[10 - 25]$, $[25 - 50]$, and $[50 - 100]$. 3. **Unequal Widths:** Note the distance between the edges. The first bin has a width of 10 units, while the last bin has a width of 50 units. 4. **Comparison:** If you had passed the integer `5` instead of a list, Matplotlib would have created 5 bars of exactly the same width covering the entire data range. 5. **Conclusion:** Passing a list allows the user to define specific, unequal-width intervals for the data distribution, which is useful for grouping data that spans several orders of magnitude or unevenly distributed categories.

Final Answer: It defines custom, unequal width intervals (edges) for the bins.

Answer: (B)

Q43.

Solution

Concept:

A Pandas DataFrame is a two-dimensional labeled data structure. To programmatically interact with a DataFrame, you often need to know the names of the columns to perform operations like filtering, renaming, or looping. Pandas stores these labels in a specialized object called a **Pandas Index**. Accessing this metadata is done through attributes. It is important to distinguish between "attributes" (which describe the structure) and "methods" (which perform actions on the data). In the CUET IP curriculum, identifying the correct property to retrieve column names is a foundational requirement.

Solution:

1. **The `columns` Property:** The `df.columns` attribute returns an Index object containing the labels of all columns in the DataFrame. 2. **Data Type:** The returned object is not a simple list, but a `pandas.Index` object, which supports set-like operations (such as finding common columns between two DataFrames). 3. **Comparison with Alternatives:**

- `df.index`: This returns the row labels, not column labels.
- `df.keys()`: In a DataFrame, this is a method that functions identically to `df.columns`, but `.columns` is the preferred attribute for structural identification.
- `df.colnames`: This is a common property in other languages (like R) but does not exist in Pandas.

4. **Conclusion:** To see or manipulate the column headers of a DataFrame, `df.columns` is the standard property used by Python developers.

Final Answer: The property is `df.columns`.

Answer: (B)

Q44.

Solution

Concept:

The modern internet operates on a suite of "Protocols," which are standardized sets of rules determining how data is exchanged between computers. Different tasks require different protocols. For example, sending an email uses one set of rules, while transferring a file uses another. For the general user, the most important protocol is the one that manages the World Wide Web. This protocol operates at the **Application Layer** of the network model and facilitates the communication between a web client (like Google Chrome) and a web server. Understanding these protocols is a core part of the Networking unit.

Solution:

1. **HTTP (Hypertext Transfer Protocol):** This is the specific protocol used to transmit "Hypertext" (HTML documents) over the internet. When you type a URL in your browser, an HTTP request is sent to the server, which responds by sending back the web page's code. 2. **Secure Version (HTTPS):** An encrypted version of this protocol is used for sensitive transactions like banking or online shopping. 3. **Comparison with Others:**

- **FTP (File Transfer Protocol):** Used for moving large files between computers, not for viewing web pages.
- **SMTP (Simple Mail Transfer Protocol):** Used specifically for outgoing emails.
- **TCP (Transmission Control Protocol):** A lower-level protocol that ensures data packets are delivered accurately; HTTP actually runs on top of TCP.

4. **Conclusion:** Because the question asks specifically for the protocol that transfers web pages to a browser, the answer is HTTP.

Final Answer: The protocol is HTTP.

Answer: (C)

Q45.

Solution

Concept:

The `INSTR()` function is an essential tool for string searching in SQL. It returns the position of a specific character or substring within a larger string. This function is frequently used in data cleaning and verification, such as checking if a specific character exists in an email address or locating delimiters in a text field. A key rule to remember for any SQL string function is that indexing starts at **1**. Additionally, if the character you are searching for appears multiple times in the string, the standard `INSTR()` function will only return the position of the **very first occurrence**.

Solution:

1. **Identify the Search Target:** We are looking for the character 'e' within the string 'Computer Science'. 2. **Count the Positions:**

- C(1), o(2), m(3), p(4), u(5), t(6), **e(7)**, r(8), space(9), S(10), c(11), i(12), **e(13)**...

3. **Selecting the First Match:** The letter 'e' occurs twice: once at position 7 (in "Computer") and once at position 13 (at the end of "Science"). 4. **Logic of INSTR:** Since the function always stops at the first successful match, it will return the value 7. 5. **Conclusion:** By counting manually and applying the "first occurrence" rule, we arrive at the correct position, which is 7.

Final Answer: The output of the query is 7.

Answer: (C)

Q46.

Solution

Concept:

Data exploration often requires determining the variety of data within a specific category. For instance, in a column containing "City" names, you might need to know how many distinct cities are represented. Pandas provides two primary tools for this: `unique()` and `nunique()`. While they sound similar, their outputs are fundamentally different. `unique()` returns an array containing the actual names of the unique items, whereas `nunique()` is a summary function that returns a single integer representing the count. Understanding which one to use is vital for writing clean, efficient code that doesn't require extra steps like calculating the length of an array manually.

Solution:

1. **The `nunique()` Method:** This is the most direct way to find the "number of unique" entries. When called on a Series, it iterates through the data and returns the total count of distinct values, excluding `NaN` (missing values) by default. 2. **Efficiency:** Using `df['Subject'].nunique()` is more efficient than the alternative `len(df['Subject'].unique())` because it is a vectorized operation designed specifically for counting. 3. **Comparison with `value_counts()`:** While `value_counts()` also identifies unique entries, its primary purpose is to show the frequency of each entry (e.g., Math: 10, Science: 8). It returns a Series, which then requires more processing to get a single count. 4. **Conclusion:** For a fast and direct integer count of unique items in a column, `nunique()` is the industry-standard method in Pandas.

Final Answer: The most efficient method is `df['Subject'].nunique()`.

Answer: (B)

Q47.

Solution

Concept:

Software licensing is a critical part of the ”Societal Impacts” unit, as it governs how technology is shared and developed globally. There are two major philosophies: **Proprietary** (closed source) and **Open Source** (free to modify). Within open source, the **GPL (General Public License)** is unique due to its ”Copyleft” clause. This ensures that the software remains free for everyone forever. It prevents people from taking a free piece of code, modifying it slightly, and then selling it as proprietary software. Understanding these licenses helps students navigate the legalities of using third-party code in their own projects.

Solution:

1. **GPL Logic:** The GPL allows users to view the source code, change it, and distribute it. However, it has a ”viral” nature: if you distribute a modified version, you **must** also release it under the GPL. This is known as ”Copyleft.” 2. **Freeware vs. GPL:** Freeware is free to use (like Adobe Reader), but you are usually not allowed to see or modify the source code. 3. **Shareware:** This is ”trialware” that you can use for free for a limited time before you are asked to pay. Like freeware, it is closed source. 4. **Proprietary:** This is commercial software (like Microsoft Windows) where the source code is a trade secret and you are only buying a license to use it, not own or change it. 5. **Conclusion:** The only license in the list that combines source code access with the ”Copyleft” requirement is the GPL.

Final Answer: The software license is GPL (General Public License).

Answer: (C)

Q48.

Solution

Concept:

Matplotlib's `plt.plot()` is a versatile function that can create basic line charts or complex, stylized visualizations. To make a plot readable, it is common practice to use "Markers" (symbols at data points) and specific "Colors." Matplotlib supports two ways of setting these styles: using explicit keyword arguments or a "Format String" shortcut. For the CUET IP exam, students must be familiar with the character codes for different colors (e.g., 'g' for green, 'r' for red) and markers (e.g., 's' for square, 'o' for circle, " for triangle).

Solution:

1. **Explicit Keywords:** The most readable way to plot is by using the parameters `color` and `marker`. For a green line, we use `color='green'` (or `'g'`). For square markers, the code is `marker='s'`. 2. **Evaluating Option A:** `plt.plot(x, y, color='green', marker='s')` uses correct, officially supported parameter names and values. 3. **Evaluating Option B:** The shorthand format string `'g-s'` would also work (green color, solid line, square marker), but the option provided is `'g-sq'`, which is invalid as 'sq' is not a recognized marker code. 4. **Incorrect Parameters:** Options C and D use `line`, `points`, and `shape`. These are not valid Matplotlib keywords and would cause a `TypeError`. 5. **Conclusion:** Option A is the correct and syntactically sound way to achieve the requested visual style.

Final Answer: The correct command is `plt.plot(x, y, color='green', marker='s')`.

Answer: (A)

Q49.

Solution

Concept:

Relational Algebra is a theoretical language used to describe how data should be manipulated in a database. One of its core operations is **Projection** (denoted by the Greek letter Pi, Π). In standard mathematical set theory, a "Set" cannot contain duplicate elements. This is a major point of difference between Relational Algebra and SQL. While SQL allows duplicate rows in its results by default (unless you use the `DISTINCT` keyword), theoretical Relational Algebra automatically removes any duplicate rows resulting from a projection.

Solution:

1. **The Scenario:** We have a table with 10 rows. We perform a projection on a column that has 3 duplicate values. 2. **Unique Count Calculation:** If 3 values are duplicates of others, it means there are only $10 - 3 = 7$ unique values in that column. 3. **Set Theory Logic:** In Relational Algebra, the output of a projection is a set. Since sets only keep unique elements, the three duplicate rows are "collapsed" into their unique counterparts. 4. **Result:** The final relation produced by the projection will contain only 7 rows. 5. **Conclusion:** This theoretical rule ensures that every relation is always a valid set, a fundamental principle of the relational model proposed by E.F. Codd.

Final Answer: In relational theory, the output will have 7 rows.

Answer: (B)

Q50.

Solution

Concept:

A **Primary Key** is a constraint used to uniquely identify each record in a database table. It acts as the "ID" for a row. Because its purpose is identification, it must follow strict rules to ensure that the database remains organized and that relationships between tables (Foreign Keys) remain valid. These rules are built into the database engine and are checked every time a user tries to insert or update data. Understanding the properties of a Primary Key is the most fundamental requirement for the "Database Concepts" unit in Informatics Practices.

Solution:

1. **Uniqueness:** A Primary Key must be unique. No two rows in the same table can have the same value in the Primary Key column. This prevents data duplication and confusion. 2. **Nullability:** A Primary Key must be **Not Null**. It cannot be empty because a "NULL" value (which represents an unknown) cannot reliably identify a specific row. 3. **The Combination:** Therefore, logically, a Primary Key is the combination of the `UNIQUE` constraint and the `NOT NULL` constraint. 4. **SQL Implementation:** When you define a column as `PRIMARY KEY`, the database automatically enforces both unique values and the presence of data in every row for that column. 5. **Conclusion:** A Primary Key must be both Unique and Not Null.

Final Answer: The Primary Key constraint is both Unique and Not Null.

Answer: (C)

Answer Key

Q	Ans	Q	Ans	Q	Ans	Q	Ans	Q	Ans
1	C	2	B	3	B	4	C	5	C
6	B	7	A	8	B	9	C	10	C
11	B	12	A	13	B	14	B	15	C
16	A	17	B	18	B	19	B	20	C
21	B	22	C	23	A	24	C	25	B
26	D	27	C	28	A	29	C	30	B
31	C	32	C	33	B	34	A	35	C
36	B	37	C	38	B	39	C	40	B
41	A	42	B	43	B	44	C	45	C
46	B	47	C	48	A	49	B	50	C