CUET-UG Information Practices Sample Paper-8

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. What will be the output of `SELECT CEIL(45.01);` ?

(A) 45

(B) 46

(C) 45.01

(D) 44

Q2. Which SQL function converts all characters of a string into lowercase letters?

(A) LOWER()

(B) LCASE()

(C) Both LOWER() and LCASE()

(D) SMALL()

Q3. What will be the output of `SELECT MID('DATABASE',3,4);` ?

(A) TABA

(B) TABS

(C) TABA

(D) TABS

Q4. Which SQL function returns the current date only?

(A) TODAY()

(B) CURDATE()

(C) NOWDATE()

(D) DATE()

Q5. What is the output of `SELECT LENGTH('INFORMATICS');` ?

(A) 10

(B) 11

(C) 12

(D) 9

Q6. Which constraint ensures that a column cannot have NULL values?

(A) UNIQUE

(B) PRIMARY KEY

(C) NOT NULL

(D) CHECK

Q7. Which relational algebra operation combines tuples of two compatible relations and removes duplicates?

(A) Difference

(B) Cartesian Product

(C) Union

(D) Selection

Q8. Which key is used to uniquely identify each tuple in a relation?

(A) Candidate Key

(B) Foreign Key

(C) Primary Key

(D) Alternate Key

Q9. Which topology requires the least amount of cable among all LAN topologies?

(A) Mesh

(B) Bus

(C) Star

(D) Tree

Q10. Which device regenerates weak signals in a network?

(A) Switch

(B) Router

(C) Repeater

(D) Gateway

Q11. Which protocol is commonly used for secure web browsing?

(A) HTTP

(B) HTTPS

(C) FTP

(D) SMTP

Q12. What will be returned by `df.shape` for a DataFrame containing 20 rows and 4 columns?

(A) (20,4)

(B) (4,20)

20,4

(C) 80

Q13. Which Pandas method is used to display the last five rows of a DataFrame?

(A) tail()

(B) end()

(C) bottom()

(D) last()

Q14. Which function is used to create a Series object in Pandas?

(A) pd.Array()

(B) pd.Series()

(C) pd.DataFrame()

(D) pd.List()

Q15. Which attribute returns the dimensions of a DataFrame?

(A) ndim

(B) shape

(C) size

(D) axes

Q16. What will be the output of `df.iloc[:,1]` ?

(A) First row

(B) Second column

(C) Second row

(D) Entire DataFrame

Q17. Which method replaces missing values with a specified value?

(A) dropna()

(B) fillna()

(C) replacena()

(D) insertna()

Q18. Which function is used to combine DataFrames horizontally using a common key?

(A) concat()

(B) append()

(C) merge()

(D) stack()

Q19. What is the output of `len(df.index)` ?

(A) Number of columns

(B) Number of rows

(C) Number of null values

(D) Total elements

Q20. Which parameter in `to_csv()` suppresses column names while writing data?

(A) columns=False

(B) names=False

(C) header=False

(D) label=False

Q21. Which indexing function selects data using row and column labels?

(A) iloc

(B) loc

(C) at

(D) iat

Q22. Which SQL function returns the average value of a numeric column?

(A) AVG()

(B) MEAN()

(C) AVERAGE()

(D) MID()

Q23. Which clause is used to filter rows before grouping takes place?

(A) HAVING

(B) ORDER BY

(C) WHERE

(D) GROUP BY

Q24. Which SQL query correctly displays all unique department names from table `employee`?

(A) `SELECT UNIQUE dept FROM employee;`

(B) `SELECT DISTINCT dept FROM employee;`

(C) `SELECT DIFFERENT dept FROM employee;`

(D) `SELECT dept UNIQUE FROM employee;`

Q25. Which join displays records having matching values in both tables only?

(A) FULL JOIN

(B) LEFT JOIN

(C) INNER JOIN

(D) RIGHT JOIN

Q26. What will be the default order of records in `ORDER BY salary`?

(A) Descending

(B) Ascending

(C) Random

(D) Reverse

Q27. Which clause is mandatory when aggregate functions are used with non-aggregate columns?

(A) ORDER BY

(B) HAVING

(C) GROUP BY

(D) DISTINCT

Q28. Which aggregate function counts all rows including NULL values?

(A) COUNT(column)

(B) COUNT(*)

(C) SUM()

(D) AVG()

Q29. Which SQL statement is used to display the highest salary from a table?

(A) `SELECT HIGH(salary) FROM emp;`

(B) `SELECT MAX(salary) FROM emp;`

(C) `SELECT TOP(salary) FROM emp;`

(D) `SELECT UPPER(salary) FROM emp;`

Q30. Which query correctly sorts employee names alphabetically in descending order?

(A) `SELECT * FROM emp ORDER BY name DESC;`

(B) `SELECT * FROM emp SORT name DESC;`

(C) `SELECT * FROM emp GROUP BY name DESC;`

(D) `SELECT * FROM emp ORDER DESC name;`

Q31. Which Matplotlib function is used to create a bar graph?

(A) plt.bar()

(B) plt.hist()

(C) plt.plot()

(D) plt.pie()

Q32. Which parameter in `plt.plot()` changes the line style?

(A) marker

(B) linestyle

(C) xlabel

(D) title

Q33. Which function is used to set the title of a graph in Matplotlib?

(A) plt.heading()

(B) plt.caption()

(C) plt.title()

(D) plt.name()

Q34. Which chart is most suitable for showing percentage contribution of categories?

(A) Histogram

(B) Scatter Plot

(C) Pie Chart

(D) Line Graph

Q35. Which Matplotlib function is used to label the Y-axis?

(A) plt.ylabel()

(B) plt.yaxis()

(C) plt.yname()

(D) plt.labely()

Q36. Which cybercrime involves creating fake websites to deceive users into sharing confidential information?

(A) Hacking

(B) Phishing

(C) Cracking

(D) Spamming

Q37. Which term refers to malicious software designed to damage or disrupt systems?

(A) Firmware

(B) Malware

(C) Shareware

(D) Freeware

Q38. Which of the following is an example of biometric authentication?

(A) Password

(B) OTP

(C) Fingerprint scan

(D) PIN

Q39. Which law protects inventions and technological innovations?

(A) Copyright

(B) Patent

(C) Trademark

(D) Cyber Law

Q40. Which of the following is considered a strong password?

(A) 12345678

(B) password

(C) Aryan123

(D) A$7kP!29

Q41. Which practice helps protect systems from unauthorized access?

(A) Disabling antivirus

(B) Using firewalls

(C) Sharing passwords

(D) Opening unknown links

Q42. Which type of software license allows users to modify and distribute the source code freely?

(A) Proprietary License

(B) Closed Source License

(C) Open Source License

(D) Trial License

Q43. Which device stores and forwards packets between different networks?

(A) Hub

(B) Switch

(C) Router

(D) NIC

Q44. Which protocol is mainly used for sending emails over the internet?

(A) FTP

(B) SMTP

(C) HTTP

(D) TCP

Q45. What is the full form of URL?

(A) Uniform Resource Locator

(B) Universal Resource Link

(C) Uniform Reference Link

(D) Universal Retrieval Locator

Q46. Which Pandas method returns the maximum value from a Series?

(A) high()

(B) maximum()

(C) max()

(D) top()

Q47. Which Pandas method is used to sort a Series in ascending order?

(A) sort()

(B) order()

(C) sort_values()

(D) arrange()

Q48. Which SQL operator is used to search for a pattern in a column?

(A) BETWEEN

(B) LIKE

(C) IN

(D) EXISTS

Q49. Which wildcard character represents exactly one character in SQL pattern matching?

(A) %

(B) *

(C) _

(D) #

Q50. Which SQL command is used to remove all records from a table without deleting its structure?

(A) DROP

(B) DELETE

(C) REMOVE

(D) TRUNCATE

Detailed Solutions

Q1.

Solution

Concept: The 'CEIL()' function, an abbreviation for 'ceiling', is a mathematical function available in SQL. It returns the smallest integer value that is greater than or equal to the given numeric expression. This means it always rounds a number *up* to the next whole number if it has any decimal component, or returns the number itself if it's already an integer.

Solution: The SQL query provided is 'SELECT CEIL(45.01);'.

1. Understand 'CEIL()': The function 'CEIL()' takes a numeric value as input. Its purpose is to find the smallest integer that is greater than or equal to that input value.
If the input is an integer (e.g., 'CEIL(45)'), the output is that same integer (45).
If the input is a decimal number (e.g., 'CEIL(45.01)' or 'CEIL(45.99)'), the output will be the next whole integer greater than the input.

2. Apply to the given value: The input value is 45.01.
Since 45.01 is not an integer, 'CEIL()' must round it up to the next greater whole number. The integers around 45.01 are 45 and 46.
The smallest integer that is greater than or equal to 45.01 is 46.
Therefore, the output of 'SELECT CEIL(45.01);' will be 46.

Final Answer : "46"

Answer: (B)

Q2.

Solution

Concept: SQL provides several built-in string functions designed to manipulate character data. Among these are functions for case conversion, allowing you to change the case of letters within a string. These functions are crucial for data standardization and comparison.

Solution: The question asks for the SQL function that converts all characters of a string into lowercase letters. Let's evaluate the given options:

1. **'LOWER()':**
This is a standard SQL function widely supported across almost all relational database management systems (RDBMS) like MySQL, PostgreSQL, SQL Server, Oracle, etc.
Its purpose is to convert every uppercase character in a given string to its corresponding lowercase character. Any characters that are already lowercase or are not letters (e.g., numbers, symbols) remain unchanged.
Example: 'SELECT LOWER('SQL Functions');' would return ''sql functions''.

2. **'LCASE()':**
This function serves the exact same purpose as 'LOWER()'. In many SQL dialects, particularly MySQL and H2, 'LCASE()' is provided as an alias or an alternative to 'LOWER()'. They perform identically. Example: 'SELECT LCASE('DATABASE');' would return ''database''.

3. **'Both LOWER() and LCASE()':**
Since both 'LOWER()' and 'LCASE()' achieve the desired result of converting a string to lowercase, and 'LCASE()' is a common alternative/alias for 'LOWER()' in several popular database systems, this option accurately encompasses both possibilities.

4. **'SMALL()':**
This is not a recognized standard SQL function for case conversion. There is no 'SMALL()' function in SQL that converts a string to lowercase.

Given that both 'LOWER()' and 'LCASE()' are valid functions (with 'LCASE()' often being an alias for 'LOWER()') for converting strings to lowercase, the most comprehensive and correct answer is that both can be used.

Final Answer : "Both LOWER() and LCASE()"

Answer: (C)

Q3.

Solution

Concept: The ‘MID()‘ function (often known as ‘SUBSTRING()‘ or ‘SUBSTR()‘ in other SQL dialects) is a string function used to extract a portion (a substring) from a larger string. It requires three pieces of information: the original string, the starting position for the extraction, and the length of the substring to be extracted. It’s important to note that string positions in SQL functions are typically 1-based, meaning the first character is at position 1.

Solution: The SQL query provided is ‘SELECT MID(’DATABASE’,3,4);‘. Let’s break down its components:
1. Original String: ‘’DATABASE’‘
2. Starting Position: ‘3‘ (This means the extraction should begin from the 3rd character of the string).
3. Length of Substring: ‘4‘ (This means 4 characters should be extracted starting from the specified position).

Now, let’s locate the characters in the string ’DATABASE’ with their 1-based positions:
D - Position 1
A - Position 2
T - Position 3 (This is our starting point)
A - Position 4
B - Position 5
A - Position 6
S - Position 7
E - Position 8

Starting from the 3rd character (’T’), we need to extract 4 characters:
1. The 1st character of the substring is ’T’ (from position 3).
2. The 2nd character of the substring is ’A’ (from position 4).
3. The 3rd character of the substring is ’B’ (from position 5).
4. The 4th character of the substring is ’A’ (from position 6).

Combining these extracted characters, the resulting substring is ’TABA’.

Final Answer : “TABA”

Answer: (A)

Q4.

Solution

Concept: SQL provides various functions to retrieve current date and time information from the database server. These functions are essential for logging events, timestamping records, or displaying current date-related information. Different functions may return only the date, only the time, or a combination of both (datetime).

Solution: The question asks for the SQL function that returns the current date only. Let's examine each option:

1. **'TODAY()':**
This is generally not a standard SQL function for retrieving the current date. While it might be used in some programming languages or specific non-SQL environments, it is not recognized as a native SQL function for this purpose across major RDBMS.

2. **'CURDATE()':**
This function is commonly found in SQL dialects like MySQL and H2, and it is specifically designed to return the current date *only*, without any time component. The output format is typically 'YYYY-MM-DD'.
Example: If today is October 26, 2023, 'SELECT CURDATE();' would return ''2023-10-26''.

3. **'NOWDATE()':**
This is not a standard SQL function. While 'NOW()' is a common SQL function, it typically returns the current date *and time* (e.g., 'YYYY-MM-DD HH:MM:SS'). 'NOWDATE()' is not a standard variant.

4. **'DATE()':**
The 'DATE()' keyword can refer to a data type (e.g., 'CREATE TABLE events (event_date DATE);'). As a function, it is typically used to extract the date part from a datetime expression (e.g., 'DATE('2023-10-26 14:30:00')' would return ''2023-10-26''). It is not generally used as a standalone function 'DATE()' to get the current date directly. To get the current date using 'DATE()', you would usually combine it with a datetime function, e.g., 'DATE(NOW())'.

Based on the options, 'CURDATE()' is the function specifically designed and widely used in relevant SQL systems to return only the current date. The ANSI SQL standard equivalent for this is 'CURRENT_DATE'.

Final Answer : "CURDATE()"

Answer: (B)

Q5.

Solution

Concept: The 'LENGTH()' function (or variations like 'LEN()' in SQL Server, or 'CHAR_LENGTH()' as a standard ANSI SQL function) is a common string function in SQL. Its purpose is to return the number of characters in a given string. It's useful for data validation, formatting, or analysis where the size of a string is important.

Solution: The SQL query is 'SELECT LENGTH('INFORMATICS');'.

1. Understand 'LENGTH()': This function takes a string as its argument and calculates the total number of characters present in that string.

2. Apply to the given string: The string provided is ''INFORMATICS''. To find its length, we simply need to count each character within the single quotes.
I - 1st character
N - 2nd character
F - 3rd character
O - 4th character
R - 5th character
M - 6th character
A - 7th character
T - 8th character
I - 9th character
C - 10th character
S - 11th character

By counting each letter, we find that the string 'INFORMATICS' consists of 11 characters. Therefore, the output of 'SELECT LENGTH('INFORMATICS');' will be 11.

Final Answer : "11"

Answer: (B)

Q6.

Solution

Concept: Database constraints are rules applied to table columns to maintain data accuracy and integrity. They restrict the type of values that can be inserted into a database.

Solution: The question asks which constraint ensures that a column cannot contain 'NULL' values.
1. **'UNIQUE':** Ensures that all values in a column are different, but it generally allows 'NULL' values.
2. **'PRIMARY KEY':** Uniquely identifies each record in a table. A primary key automatically includes both 'UNIQUE' and 'NOT NULL' properties.
3. **'NOT NULL':** Explicitly prevents a column from storing 'NULL' values. Every record must contain a value for that column.
4. **'CHECK':** Ensures that values satisfy a specific condition, but it does not automatically prevent 'NULL' values.

Thus, the constraint specifically designed to disallow 'NULL' values is 'NOT NULL'.

Final Answer : "NOT NULL"

Answer: (C)

Q7.

Solution

Concept: Relational algebra is a procedural query language used in relational databases. Operations in relational algebra work on relations (tables) and produce a new relation as output.

Solution: The question asks which operation combines tuples of two compatible relations and removes duplicates.
1. **'Difference':** Returns tuples present in the first relation but absent in the second relation.
2. **'Cartesian Product':** Combines every tuple of one relation with every tuple of another relation. It does not remove duplicates.
3. **'Union':** Combines tuples from both relations into a single relation and automatically removes duplicate tuples. It requires the relations to be union-compatible.
4. **'Selection':** Selects tuples from a single relation based on a condition. It does not combine relations.

Therefore, the operation that combines tuples and removes duplicates is 'Union'.

Final Answer : "Union"

Answer: (C)

Q8.

Solution

Concept: Keys are important in relational databases because they help uniquely identify records and establish relationships between tables.

Solution: The question asks which key is used to uniquely identify each tuple in a relation.
1. **'Candidate Key':** Any attribute or set of attributes that can uniquely identify a tuple in a table.
2. **'Foreign Key':** Used to create relationships between two tables by referring to the primary key of another table.
3. **'Primary Key':** The selected candidate key used as the main identifier for records in a table. It must contain unique and non-'NULL' values.
4. **'Alternate Key':** Candidate keys that are not chosen as the primary key.

Hence, the 'Primary Key' is used to uniquely identify each tuple in a relation.

Final Answer : "Primary Key"

Answer: (C)

Q9.

Solution

Concept: Understanding different network topologies and their implications for cable installation.

Solution: Among the common LAN topologies:

- **Bus topology** uses a single main cable, often called a backbone, to which all workstations and servers are connected. Data travels along this single cable. This design inherently requires the least amount of cable because there isn't a dedicated cable running from each device to a central point, nor are there connections between every pair of devices.
- **Star topology** requires each device to have its own dedicated cable segment connecting it to a central hub, switch, or router. While robust, this means a significant amount of cabling, especially in larger networks.
- **Mesh topology** requires a dedicated point-to-point link to every other device in the network. For 'n' devices, this means 'n * (n-1) / 2' cables, making it the most cable-intensive topology.
- **Tree topology** is a hybrid that combines elements of bus and star topologies. It extends a star bus, meaning it typically requires more cable than a simple bus structure.

Given these characteristics, the Bus topology is the most efficient in terms of cable usage.

Final Answer : "Bus"

Answer: (B)

Q10.

Solution

Concept: Identifying the functions of various network hardware devices.

Solution: Let's analyze the function of each device:

- A **Switch** operates at Layer 2 (Data Link Layer) of the OSI model. It connects multiple devices within a local area network (LAN) and intelligently forwards data frames only to the intended recipient, based on MAC addresses. It does not primarily regenerate weak signals.
- A **Router** operates at Layer 3 (Network Layer). It connects different networks (e.g., your home network to the internet) and forwards data packets between them based on IP addresses, determining the best path for data. It does not primarily regenerate weak signals.
- A **Repeater** operates at Layer 1 (Physical Layer). Its sole purpose is to receive a weak or degraded signal, regenerate it to its original strength and form, and then retransmit it. This extends the effective range of a network segment by overcoming signal attenuation (weakening) over distance.
- A **Gateway** is a network node that connects two networks with different transmission protocols. It translates protocols to allow communication between disparate networks. It typically involves more complex protocol conversions than simple signal regeneration.

Therefore, the device specifically designed to regenerate weak signals is a Repeater.

Final Answer : "Repeater"

Answer: (C)

Q11.

Solution

Concept: Understanding secure communication protocols on the internet.

Solution: Let's examine the protocols listed:

- **HTTP (Hypertext Transfer Protocol)** is the foundational protocol for data communication on the World Wide Web. It is used to send and receive web pages, but the data transferred via HTTP is unencrypted, meaning it can be intercepted and read by malicious actors.
- **HTTPS (Hypertext Transfer Protocol Secure)** is the secure version of HTTP. It uses SSL/TLS (Secure Sockets Layer/Transport Layer Security) to encrypt the communication between a web browser and a website. This encryption protects the integrity and confidentiality of data, making it suitable for sensitive transactions like online banking, e-commerce, and any interaction where privacy is critical.
- **FTP (File Transfer Protocol)** is a standard network protocol used for transferring computer files from a server to a client on a computer network. While it can be secured with FTPS or SFTP, basic FTP itself is not inherently secure and does not encrypt data.
- **SMTP (Simple Mail Transfer Protocol)** is an internet standard for electronic mail (email) transmission. It is primarily used for sending email messages and, in its basic form, does not provide encryption for the email content or login credentials.

Therefore, HTTPS is the protocol commonly used for secure web browsing due to its built-in encryption capabilities.

Final Answer : "HTTPS"

Answer: (B)

Q12.

Solution

Concept: Understanding the ‘.shape‘ attribute in Pandas DataFrames.

Solution: In the Pandas library, a DataFrame is a two-dimensional labeled data structure with columns of potentially different types. The ‘.shape‘ attribute is a fundamental property of a DataFrame (and also Series) that provides its dimensions.

- When ‘df.shape‘ is called on a DataFrame, it returns a tuple.
- The first element of this tuple represents the number of rows in the DataFrame.
- The second element of this tuple represents the number of columns in the DataFrame.

Given a DataFrame containing 20 rows and 4 columns, ‘df.shape‘ will return a tuple where the first value is 20 and the second value is 4. Therefore, the output will be ‘(20, 4)‘.

Final Answer : “(20,4)”

Answer: (A)

Q13.

Solution

Concept: Pandas DataFrame methods for inspecting data.

Solution: Pandas provides convenient methods to quickly inspect parts of a DataFrame:

- The ‘head()‘ method is used to display the first ’n’ rows of a DataFrame. By default, if no argument is specified, it displays the first five rows.
- The ‘tail()‘ method is used to display the last ’n’ rows of a DataFrame. By default, if no argument is specified, it displays the last five rows.
- ‘end()‘, ‘bottom()‘, and ‘last()‘ are not standard Pandas DataFrame methods for this specific purpose. While one might intuitively associate them with the end of a DataFrame, the correct method name in Pandas is ‘tail()‘.

Therefore, to display the last five rows of a DataFrame, the ‘tail()‘ method is used.

Final Answer : “tail()”

Answer: (A)

Q14.

Solution

Concept: Creating Pandas Series objects.

Solution: The Pandas library offers specific constructors for its core data structures:

- A **Series** in Pandas is a one-dimensional labeled array capable of holding any data type (integers, strings, floating point numbers, Python objects, etc.). It is essentially a column of a DataFrame. To create a Series object, the constructor provided by the Pandas library is ‘pd.Series()‘.
- A **DataFrame** is a two-dimensional labeled data structure with columns of potentially different types. To create a DataFrame object, the constructor is ‘pd.DataFrame()‘.
- ‘pd.Array()‘ and ‘pd.List()‘ are not standard constructors for creating Series or DataFrame objects directly in Pandas. While ‘pd.array‘ exists in a specific context (for extension arrays), ‘pd.Array()‘ is not the general function for creating a Series. Python’s built-in ‘list()‘ creates a list, not a Pandas object.

Hence, to create a Series object in Pandas, ‘pd.Series()‘ is the correct function.

Final Answer : “pd.Series()”

Answer: (B)

Q15.

Solution

Concept: Pandas DataFrame attributes for obtaining structural information.

Solution: Pandas DataFrames have several attributes that provide information about their structure and dimensions:

- **‘ndim‘**: This attribute returns the number of array dimensions. For a Series, ‘ndim‘ is 1. For a DataFrame, ‘ndim‘ is always 2.
- **‘shape‘**: This attribute returns a tuple representing the dimensions of the DataFrame, specifically ‘(number_of_rows, number_of_columns)‘. It’s the most direct way to get the row and column counts.
- **‘size‘**: This attribute returns the total number of elements in the DataFrame, which is equivalent to ‘rows * columns‘.
- **‘axes‘**: This attribute returns a list of the row and column labels (index and columns objects).

The question asks for the attribute that returns the *dimensions* of a DataFrame, which is precisely what ‘shape‘ provides as a ‘(rows, columns)‘ tuple.

Final Answer : “shape”

Answer: (B)

Q16.

Solution

Concept: Understanding integer-location based indexing with ‘iloc‘ in Pandas.

Solution: The ‘iloc‘ indexer in Pandas is used for integer-location based indexing, meaning you select data based on the integer positions of rows and columns (from 0 to ‘length-1‘). The syntax is generally ‘df.iloc[row_indexer, column_indexer]‘. Let’s break down ‘df.iloc[:,1]‘:

- The first part, ‘:‘, is the slice operator for rows. When used alone, it means "select all rows".
- The second part, ‘1‘, is the integer index for the column. In Python and Pandas, indexing is 0-based. So, index ‘0‘ refers to the first element/column, index ‘1‘ refers to the second element/column, and so on.

Therefore, ‘df.iloc[:,1]‘ selects all rows (‘:‘) and the column at integer position ‘1‘, which is the second column of the DataFrame.

Final Answer : “Second column”

Answer: (B)

Q17.

Solution

Concept: Handling missing data in Pandas DataFrames.

Solution: Missing data, often represented as ‘NaN‘ (Not a Number) in Pandas, is a common issue in real-world datasets. Pandas provides specific methods to manage these missing values:

- **‘dropna()‘**: This method is used to *remove* rows or columns that contain missing values (NaN). You can specify whether to drop rows (‘axis=0‘) or columns (‘axis=1‘) and how stringent the dropping condition should be (e.g., ‘how=’any’‘ or ‘how=’all’‘).
- **‘fillna()‘**: This method is specifically designed to *replace* missing values with a specified value or by using a particular filling strategy. For example, ‘df.fillna(0)‘ would replace all ‘NaN‘s with ‘0‘, ‘df.fillna(df.mean())‘ would fill with column means, and ‘df.fillna(method=’ffill’)‘ would forward-fill values.
- **‘replacena()‘** and **‘insertna()‘** are not standard Pandas methods for handling missing values. While you can use ‘replace()‘ to substitute values, ‘fillna()‘ is the dedicated and most efficient method for handling ‘NaN‘s specifically.

Thus, ‘fillna()‘ is the correct method for replacing missing values with a specified value.

Final Answer : “fillna()”

Answer: (B)

Q18.

Solution

Concept: Methods for combining DataFrames in Pandas.

Solution: Pandas offers several powerful functions to combine DataFrames:

- **‘concat()‘**: This function is used to concatenate Pandas objects along a particular axis (row-wise or column-wise). It essentially "stacks" DataFrames either on top of each other (vertically, ‘axis=0‘) or side-by-side (horizontally, ‘axis=1‘). It’s useful for combining DataFrames that have the same columns or same index.
- **‘append()‘**: This method was used to append rows of one DataFrame to another. It is now deprecated in favor of ‘concat()‘.
- **‘merge()‘**: This function is used to combine two DataFrames based on a common column or index, similar to SQL JOIN operations. It’s ideal for combining DataFrames horizontally when there’s a logical relationship (a common "key") between them, allowing for inner, outer, left, and right joins. This is exactly what is meant by "combining DataFrames horizontally using a common key."
- **‘stack()‘**: This method reshapes a DataFrame by transforming columns into rows, effectively moving the innermost column index to become the innermost row index. It’s for reshaping, not directly for combining two separate DataFrames based on a key.

Therefore, ‘merge()‘ is the function specifically designed for combining DataFrames horizontally using a common key.

Final Answer : “merge()”

Answer: (C)

Q19.

Solution

Concept: Understanding DataFrame index and using the ‘len()‘ function.

Solution: In a Pandas DataFrame (‘df‘):

- ‘df.index‘ is an attribute that returns the ‘Index‘ object of the DataFrame. This ‘Index‘ object contains the labels for each row.
- The built-in Python ‘len()‘ function, when applied to any sequence or collection, returns the number of items in that sequence or collection.

When you execute ‘len(df.index)‘, you are asking for the number of labels present in the DataFrame’s index. Since each row in a DataFrame has a unique label (or at least a distinct position), the number of labels in the index directly corresponds to the total number of rows in the DataFrame.

Final Answer : “Number of rows”

Answer: (B)

Q20.

Solution

Concept: Controlling output format when writing DataFrames to CSV files.

Solution: The ‘to_csv()‘ method in Pandas is used to write a DataFrame to a comma-separated values (CSV) file. It has several parameters to control the output format:

- ‘index‘: A boolean parameter (‘True‘ by default) that controls whether the DataFrame’s index is written as a column in the CSV file.
- ‘header‘: A boolean parameter (‘True‘ by default) that controls whether the column names (the header row) are written to the CSV file. Setting ‘header=False‘ will suppress the writing of the column names.
- ‘columns‘: A list of column labels to write. If specified, only these columns will be written. This doesn’t suppress the header entirely but selects which columns appear.
- ‘names‘ and ‘label‘ are not standard parameters for this purpose in ‘to_csv()‘.

Therefore, to suppress column names while writing data to a CSV file, the ‘header=False‘ parameter should be used.

Final Answer : “header=False”

Answer: (C)

Q21.

Solution

Concept: Understanding different Pandas indexing methods.

Solution: Pandas provides powerful and flexible indexing methods to select data from DataFrames and Series:

- **‘iloc‘**: This is used for *integer-location based* indexing. You specify rows and columns using their integer positions (0-based indexing), not their labels. For example, ‘df.iloc[0, 1]‘ selects the value at the first row and second column by their position.
- **‘loc‘**: This is used for *label-based* indexing. You specify rows and columns using their actual labels (names), not their integer positions. For example, ‘df.loc[’row_label’, ’col_label’]‘ selects the value at the specified row and column labels.
- **‘at‘**: This is a fast label-based scalar accessor. It is similar to ‘loc‘ but optimized for getting or setting a *single value* by its row and column labels.
- **‘iat‘**: This is a fast integer-location based scalar accessor. Similar to ‘iloc‘ but optimized for getting or setting a *single value* by its integer row and column positions.

The question asks for the indexing function that selects data using *row and column labels*, which is precisely the purpose of ‘loc‘.

Final Answer : “loc”

Answer: (B)

Q22.

Solution

Concept: Understanding SQL aggregate functions.

Solution: SQL (Structured Query Language) provides several aggregate functions to perform calculations on a set of rows and return a single summary value.

- **‘AVG()‘**: This is the standard SQL aggregate function used to calculate the average value of a numeric column. It calculates the sum of the values in the specified column and divides it by the count of non-NULL values in that column. For example, ‘SELECT AVG(salary) FROM employees;‘ would return the average salary of all employees.
- **‘MEAN()‘**: While "mean" is a common statistical term for average, ‘MEAN()‘ is not a standard SQL aggregate function. Some database systems might offer it as an extension, but it’s not universally recognized.
- **‘AVERAGE()‘**: Similar to ‘MEAN()‘, ‘AVERAGE()‘ is not a standard SQL aggregate function for calculating the average.
- **‘MID()‘**: This is often a string function (e.g., in MS Access or some versions of SQL Server, often called ‘SUBSTRING‘ in standard SQL) used to extract a substring from a string. It has no relation to calculating an average.

Therefore, the correct SQL function to return the average value of a numeric column is ‘AVG()‘.

Final Answer : “AVG()”

Answer: (A)

Q23.

Solution

Concept: Understanding the order of execution of SQL clauses and their filtering purposes.

Solution: In a SQL query, clauses are processed in a specific logical order, which is not necessarily the order in which they appear in the query:
1. **'FROM'/**'JOIN' clauses: Determine the tables involved and how they are joined.
2. **'WHERE' clause**: Filters individual rows from the tables based on specified conditions. This filtering happens before any grouping occurs. Rows that do not meet the 'WHERE' condition are discarded.
3. **'GROUP BY' clause**: Groups the remaining rows that have identical values in one or more specified columns into summary rows.
4. **'HAVING' clause**: Filters these *groups* based on specified conditions. This filtering happens after grouping and after aggregate functions have been applied to each group.
5. **'SELECT' clause**: Specifies which columns/expressions to retrieve.
6. **'ORDER BY' clause**: Sorts the final result set.

The question specifically asks about filtering rows before grouping takes place. Based on the logical order of execution, the 'WHERE' clause is responsible for this pre-grouping filtering.

Final Answer : "WHERE"

Answer: (C)

Q24.

Solution

Concept: Retrieving unique values in SQL.

Solution: In SQL, to display only the unique (non-duplicate) values from a specific column, the 'DISTINCT' keyword is used in conjunction with the 'SELECT' statement. Let's analyze the given options:

- `SELECT UNIQUE dept FROM employee;`: 'UNIQUE' is not the standard SQL keyword for this purpose. While it might be supported as a synonym in some specific database systems (like Oracle), 'DISTINCT' is the ANSI SQL standard.
- `SELECT DISTINCT dept FROM employee;`: This is the correct and standard SQL syntax to retrieve all unique department names from the 'employee' table. The 'DISTINCT' keyword ensures that each 'dept' value appears only once in the result set.
- `SELECT DIFFERENT dept FROM employee;`: 'DIFFERENT' is not a valid SQL keyword for selecting unique values.
- `SELECT dept UNIQUE FROM employee;`: The 'UNIQUE' keyword is misplaced here and used incorrectly as per standard SQL syntax for selecting distinct values. It's often used in constraint definitions (e.g., 'UNIQUE' constraint).

Therefore, 'SELECT DISTINCT dept FROM employee;' is the correct SQL query.

Final Answer : "SELECT DISTINCT dept FROM employee;"

Answer: (B)

Q25.

Solution

Concept: Understanding different types of SQL JOIN operations and their data retrieval behavior.

Solution: SQL JOIN clauses are used to combine rows from two or more tables based on a related column between them.

- **INNER JOIN**: This is the most common type of join. It returns only the rows that have matching values in *both* tables based on the specified join condition. Rows from either table that do not have a match in the other table are excluded from the result set.
- **LEFT JOIN (or LEFT OUTER JOIN)**: This join returns all rows from the "left" table (the first table specified in the ‘FROM‘ clause) and the matching rows from the "right" table. If there is no match in the right table, ‘NULL‘ values are returned for the columns of the right table.
- **RIGHT JOIN (or RIGHT OUTER JOIN)**: This join returns all rows from the "right" table and the matching rows from the "left" table. If there is no match in the left table, ‘NULL‘ values are returned for the columns of the left table.
- **FULL JOIN (or FULL OUTER JOIN)**: This join returns all rows when there is a match in either the left or the right table. It effectively combines the results of a LEFT JOIN and a RIGHT JOIN. If there is no match, ‘NULL‘ values are returned for the columns of the non-matching side.

The question asks for the join that displays records having matching values in both tables *only*. This precisely describes the behavior of an INNER JOIN.

Final Answer : “INNER JOIN”

Answer: (C)

Q26.

Solution

Concept: Understanding the default sorting order of the ‘ORDER BY‘ clause in SQL.

Solution: The ‘ORDER BY‘ clause in SQL is used to sort the result set of a query in ascending or descending order based on one or more columns.

- If you want to sort in ascending order (from lowest to highest, or A-Z for strings), you can explicitly use the ‘ASC‘ keyword (e.g., ‘ORDER BY salary ASC‘).
- If you want to sort in descending order (from highest to lowest, or Z-A for strings), you must explicitly use the ‘DESC‘ keyword (e.g., ‘ORDER BY salary DESC‘).
- When neither ‘ASC‘ nor ‘DESC‘ is specified after the column name in an ‘ORDER BY‘ clause, SQL databases universally default to sorting in **ascending** order.

Therefore, ‘ORDER BY salary‘ without any explicit keyword will sort the records by salary in ascending order.

Final Answer : “Ascending”

Answer: (B)

Q27.

Solution

Concept: Rules for using aggregate functions in SQL, specifically with non-aggregate columns.

Solution: In SQL, aggregate functions (like ‘COUNT()‘, ‘SUM()‘, ‘AVG()‘, ‘MAX()‘, ‘MIN()‘) perform calculations on a set of rows and return a single summary value. When you include both aggregate functions and non-aggregate columns in your ‘SELECT‘ statement, SQL needs to know how to group the rows before applying the aggregate function.

- If you select a non-aggregate column (e.g., ‘dept_name‘) along with an aggregate function (e.g., ‘COUNT(employee_id)‘), you must use the ‘GROUP BY‘ clause.
- The ‘GROUP BY‘ clause specifies the columns by which the rows should be grouped. All non-aggregate columns in the ‘SELECT‘ list must also appear in the ‘GROUP BY‘ clause (or be functionally dependent on the ‘GROUP BY‘ columns).
- ‘ORDER BY‘ is for sorting the final result set.
- ‘HAVING‘ is for filtering groups after they have been formed and aggregates calculated.
- ‘DISTINCT‘ is for selecting unique values.

For example, to count the number of employees in each department, you would write ‘SELECT dept_name, COUNT(employee_id) FROM employee GROUP BY dept_name;‘. Without ‘GROUP BY dept_name‘, most SQL databases would raise an error because they wouldn’t know how to relate a single department name to multiple counts.

Final Answer : “GROUP BY”

Answer: (C)

Q28.

Solution

Concept: Understanding the behavior of the ‘COUNT()‘ aggregate function in SQL with respect to NULL values.

Solution: The ‘COUNT()‘ aggregate function in SQL is used to count the number of rows or non-NULL values. Its behavior differs based on its argument:

- **‘COUNT(column_name)‘**: This counts the number of non-NULL values in the specified ‘column_name‘. If a row has a ‘NULL‘ value in that particular column, that row is not included in the count for that column.
- **‘COUNT(*)‘**: This counts the total number of rows in the result set, irrespective of whether any of the columns in those rows contain ‘NULL‘ values. It counts *all* rows that satisfy the ‘WHERE‘ clause (if any). This is the function to use when you want to count every single record.
- **‘SUM()‘**: This function calculates the sum of all numeric values in a column, ignoring ‘NULL‘ values.
- **‘AVG()‘**: This function calculates the average of all numeric values in a column, ignoring ‘NULL‘ values.

Therefore, ‘COUNT(*)‘ is the aggregate function that counts all rows, including those that might have ‘NULL‘ values in other columns.

Final Answer : “COUNT(*)”

Answer: (B)

Q29.

Solution

Concept: Using SQL aggregate functions to find maximum values.

Solution: SQL provides a set of aggregate functions to perform calculations on a group of rows. To find the highest value within a numeric column, a specific aggregate function is used:

- **‘MAX(column_name)‘**: This is the standard SQL aggregate function used to retrieve the maximum value from a specified column. It works for numeric, string, and date/time columns. For example, ‘SELECT MAX(salary) FROM emp;‘ will return the single highest salary found in the ‘salary‘ column of the ‘emp‘ table.
- **‘HIGH()‘**: This is not a standard SQL aggregate function for finding the maximum value.
- **‘TOP()‘**: In some SQL dialects (like SQL Server), ‘TOP‘ is used with ‘SELECT‘ to limit the number of rows returned by a query (e.g., ‘SELECT TOP 1 salary FROM emp ORDER BY salary DESC;‘). It doesn’t directly return the maximum value as an aggregate function.
- **‘UPPER()‘**: This is a string function that converts all characters in a string to uppercase. It is unrelated to finding a maximum numeric value.

Thus, the correct SQL statement to display the highest salary is ‘SELECT MAX(salary) FROM emp;‘.

Final Answer : “SELECT MAX(salary) FROM emp;”

Answer: (B)

Q30.

Solution

Concept: Correct syntax for sorting query results in SQL using the ‘ORDER BY‘ clause.

Solution: To sort the records in a SQL query result, the ‘ORDER BY‘ clause is used. This clause is placed after the ‘FROM‘ and ‘WHERE‘ clauses (and ‘GROUP BY‘ and ‘HAVING‘ if present).

- The basic syntax for sorting is ‘ORDER BY column_name [ASC | DESC]‘.
- ‘ASC‘ (Ascending) sorts from smallest to largest (A-Z for text, 0-9 for numbers, earliest to latest for dates). This is the default if neither ‘ASC‘ nor ‘DESC‘ is specified.
- ‘DESC‘ (Descending) sorts from largest to smallest (Z-A for text, 9-0 for numbers, latest to earliest for dates).

Let’s evaluate the given options:

(a) `SELECT * FROM emp ORDER BY name DESC;`: This correctly uses the ‘ORDER BY‘ clause, specifies the ‘name‘ column, and uses ‘DESC‘ for descending order. This will sort employee names alphabetically in reverse order (Z-A).

(b) `SELECT * FROM emp SORT name DESC;`: ‘SORT‘ is not a standard SQL keyword for ordering results; ‘ORDER BY‘ must be used.

(c) `SELECT * FROM emp GROUP BY name DESC;`: ‘GROUP BY‘ is used for grouping rows, not for sorting the final display order. While ‘GROUP BY‘ can implicitly order results in some databases, it’s not its primary purpose and ‘DESC‘ here is syntactically incorrect for ‘GROUP BY‘ in most standard SQL contexts.

(d) `SELECT * FROM emp ORDER DESC name;`: The ‘DESC‘ keyword must come after the column name it applies to. The correct placement is ‘ORDER BY name DESC‘.

Therefore, the query ‘SELECT * FROM emp ORDER BY name DESC;‘ correctly sorts employee names alphabetically in descending order.

Final Answer : “SELECT * FROM emp ORDER BY name DESC;”

Answer: (A)

Q31.

Solution

Concept: Identifying the correct Matplotlib function for creating specific types of plots.

Solution: Matplotlib's 'pyplot' module (commonly imported as 'plt') provides a wide range of functions for creating various types of plots:

- **'plt.bar()'**: This function is specifically designed to create vertical bar graphs. It takes x-coordinates and corresponding heights for the bars as input.
- **'plt.hist()'**: This function creates a histogram, which is used to display the distribution of a single numerical variable by dividing data into bins and counting observations in each bin.
- **'plt.plot()'**: This is a versatile function primarily used to create 2D line plots and scatter plots. It can display data points connected by lines or just markers.
- **'plt.pie()'**: This function is used to create pie charts, which show the proportional distribution of categories within a whole.

To create a bar graph, the 'plt.bar()' function is the appropriate choice in Matplotlib.

Final Answer : "plt.bar()"

Answer: (A)

Q32.

Solution

Concept: Customizing line properties in Matplotlib's 'plt.plot()' function.

Solution: The 'plt.plot()' function is highly customizable, allowing control over various aspects of the line:

- **'marker'**: This parameter specifies the style of markers to be displayed at data points (e.g., 'o' for circles, 'x' for 'x' marks, '*' for stars).
- **'linestyle' (or 'ls')**: This parameter controls the style of the line connecting the data points. Common values include ''-'' (solid line), ''–'' (dashed line), '':'' (dotted line), and ''-.'' (dash-dot line).
- **'xlabel'**: This is a separate function ('plt.xlabel()') used to set the label for the x-axis, not a parameter within 'plt.plot()'.
- **'title'**: This is a separate function ('plt.title()') used to set the title of the plot, not a parameter within 'plt.plot()'.

Therefore, to change the line style in 'plt.plot()', the 'linestyle' (or its shorthand 'ls') parameter is used.

Final Answer : "linestyle"

Answer: (B)

Q33.

Solution

Concept: Setting various textual elements for a graph in Matplotlib.

Solution: Matplotlib's 'pyplot' interface provides dedicated functions for adding descriptive text elements to plots to enhance their readability and context:

- **'plt.title()'**: This is the standard function used to set the main title of the current plot. It accepts a string argument for the title text.
- **'plt.xlabel()'** and 'plt.ylabel()': Used to set labels for the x and y axes, respectively.
- **'plt.text()'**: Used to add arbitrary text at any location on the plot.
- **'plt.heading()'**, **'plt.caption()'**, and **'plt.name()'** are not standard Matplotlib functions for setting the plot's title. While 'caption' might be used in other contexts (like HTML), 'title' is the specific term and function used in Matplotlib for this purpose.

Hence, to set the title of a graph in Matplotlib, the 'plt.title()' function is the correct choice.

Final Answer : "plt.title()"

Answer: (C)

Q34.

Solution

Concept: Choosing the most appropriate data visualization chart type for specific data representation goals.

Solution: Different types of charts are best suited for different data visualization tasks:

- **Histogram**: Used to display the distribution of a single numerical variable. It shows how frequently different value ranges occur, but not the contribution of categories to a whole.
- **Scatter Plot**: Used to show the relationship or correlation between two numerical variables. Each point represents an observation, and its position indicates the values of the two variables. It's not suitable for showing proportional contributions.
- **Pie Chart**: This chart is specifically designed to represent proportions or percentages of a whole. A circle is divided into sectors (slices), where the area of each sector is proportional to the percentage it represents out of the total. It is highly effective for illustrating how different categories contribute to a sum.
- **Line Graph**: Primarily used to display trends over a continuous variable, often time. It shows how one or more variables change over a period, but it's not ideal for showing percentage contributions of distinct categories at a single point in time.

Therefore, a Pie Chart is the most suitable chart for showing the percentage contribution of different categories to a total.

Final Answer : "Pie Chart"

Answer: (C)

Q35.

Solution

Concept: Labeling axes in Matplotlib plots for clarity.

Solution: Matplotlib provides dedicated functions for labeling the axes of a plot, which is crucial for making visualizations informative and understandable:

- **‘plt.xlabel()‘**: This function is used to set the label for the X-axis.
- **‘plt.ylabel()‘**: This function is used to set the label for the Y-axis. It takes a string as an argument, which will be displayed as the label along the Y-axis.
- **‘plt.yaxis()‘**: This is not a standard Matplotlib function for setting the Y-axis label. While there are methods like ‘ax.yaxis.set_label_text()‘, ‘plt.ylabel()‘ is the direct and commonly used function in the ‘pyplot‘ interface.
- **‘plt.yname()‘** and **‘plt.labely()‘** are not standard Matplotlib functions for this purpose.

Therefore, to label the Y-axis in Matplotlib, ‘plt.ylabel()‘ is the correct function.

Final Answer : “plt.ylabel()”

Answer: (A)

Q36.

Solution

Concept: Understanding common types of cybercrime.

Solution: Let's define the cybercrimes listed:

- **Hacking**: Refers to gaining unauthorized access to computer systems or networks. It's a broad term for illicit access.
- **Phishing**: This is a deceptive cybercrime tactic where attackers attempt to trick individuals into revealing sensitive information (like usernames, passwords, credit card details) by disguising themselves as trustworthy entities in electronic communication. This often involves creating fake websites that mimic legitimate ones (e.g., banking sites, social media platforms) to capture login credentials.
- **Cracking**: Often used interchangeably with hacking, but sometimes specifically refers to breaking into systems with malicious intent, such as breaking software copy protection or password encryption.
- **Spamming**: Refers to sending unsolicited messages, typically emails, in bulk to a large number of recipients, often for commercial advertising or malicious purposes. While it can be part of a phishing campaign, spamming itself doesn't inherently involve fake websites for credential harvesting.

The description of creating fake websites to deceive users into sharing confidential information perfectly matches the definition of Phishing.

Final Answer : "Phishing"

Answer: (B)

Q37.

Solution

Concept: Classifying different types of software.

Solution: Let's differentiate the terms:

- **Firmware**: Software permanently embedded in hardware devices (e.g., BIOS, router firmware). It's essential for the hardware's basic operation but is not inherently malicious.
- **Malware**: A portmanteau of "malicious software." This is a broad term that encompasses any software intentionally designed to cause damage to a computer, server, client, or computer network, or to gain unauthorized access to systems, steal data, or disrupt operations. Examples include viruses, worms, Trojans, ransomware, spyware, and adware.
- **Shareware**: Software that is distributed free of charge, typically for evaluation purposes, with the understanding that users may need to pay a fee to continue using it or to access full features after a trial period. It is not inherently malicious.
- **Freeware**: Software that is available for use at no cost, typically with proprietary licenses that restrict modification or redistribution. It is not inherently malicious.

The term that refers to malicious software designed to damage or disrupt systems is Malware.

Final Answer : "Malware"

Answer: (B)

Q38.

Solution

Concept: Understanding different methods of user authentication.

Solution: Authentication is the process of verifying the identity of a user or system. There are typically three main categories of authentication factors:

- **Something you know**: This includes passwords, PINs (Personal Identification Numbers), and answers to security questions.
- **Something you have**: This includes physical tokens, smart cards, and one-time passwords (OTPs) sent to a registered device.
- **Something you are**: This refers to biometric data, which are unique biological or behavioral characteristics of an individual.

Let's look at the options:

- **Password**: Something you know.
- **OTP (One-Time Password)**: Something you have (usually a code sent to your phone or generated by an app on a device you own).
- **Fingerprint scan**: This is a direct measurement of a unique physical characteristic (your fingerprint pattern) for identification. It falls under "something you are," making it an example of biometric authentication.
- **PIN**: Something you know.

Therefore, a Fingerprint scan is an example of biometric authentication.

Final Answer : "Fingerprint scan"

Answer: (C)

Q39.

Solution

Concept: Understanding different types of Intellectual Property (IP) laws.

Solution: Intellectual Property (IP) laws protect creations of the mind. Different types of IP laws protect different kinds of creations:

- **Copyright**: Protects original works of authorship, such as literary, dramatic, musical, and artistic works (including software code, books, movies, music, and art). It grants the creator exclusive rights to reproduce, distribute, and display their work.
- **Patent**: Protects inventions. A patent grants the inventor exclusive rights to make, use, and sell their invention for a period, typically 20 years. This applies to new and useful processes, machines, articles of manufacture, compositions of matter, or any new and useful improvement thereof. This directly covers technological innovations.
- **Trademark**: Protects words, names, symbols, sounds, or colors that distinguish goods and services from those of others. It's about brand identity (e.g., company logos, product names).
- **Cyber Law**: This is a broad term encompassing legal issues related to the internet and cyberspace. It covers various aspects like e-commerce, data privacy, cybercrime, and intellectual property in the digital realm, but it's not a specific type of IP protection itself like patent or copyright.

The law that specifically protects inventions and technological innovations is a Patent.

Final Answer : "Patent"

Answer: (B)

Q40.

Solution

Concept: Characteristics of a strong password in cybersecurity.

Solution: A strong password helps protect accounts and systems from unauthorized access. Such passwords are difficult to guess or crack because they contain a combination of different character types and avoid predictable patterns.
The important characteristics of a strong password are:

- It should contain uppercase and lowercase letters.
- It should include numbers and special symbols.
- It should avoid common words or predictable sequences.
- It should not contain personal information such as names or birthdays.
- Greater length and complexity improve password security.

Now, examine the given options:

- `12345678` is a very common sequential password and can be guessed easily.
- `password` is one of the weakest and most commonly used passwords.
- `Aryan123` contains a predictable name and simple numeric pattern.
- `A$7kP!29` contains uppercase letters, lowercase letters, numbers, and special characters, making it strong and difficult to crack.

Hence, the strongest password among the given options is `A$7kP!29`.

Final Answer : "A$7kP!29"

Answer: (D)

Q41.

Solution

Concept: Best practices for cybersecurity and protecting systems from unauthorized access.

Solution: Protecting systems from unauthorized access is a core goal of cybersecurity. Let's analyze the options:

- **Disabling antivirus**: Antivirus software is designed to detect, prevent, and remove malicious software. Disabling it leaves the system vulnerable to various threats, increasing the risk of unauthorized access or infection. This practice *harms* protection.
- **Using firewalls**: A firewall acts as a barrier between a trusted internal network and untrusted external networks (like the internet). It monitors and controls incoming and outgoing network traffic based on predetermined security rules, blocking unauthorized access attempts. This practice *helps* protect systems.
- **Sharing passwords**: Sharing passwords compromises the security of accounts and systems. It allows multiple individuals to access resources, making it difficult to track who did what and increasing the risk of credentials falling into the wrong hands. This practice *harms* protection.
- **Opening unknown links**: Clicking on unknown or suspicious links (especially in emails or messages) can lead to phishing websites, malware downloads, or other cyber attacks that grant unauthorized access or compromise data. This practice *harms* protection.

Therefore, using firewalls is a crucial practice that helps protect systems from unauthorized access.

Final Answer : "Using firewalls"

Answer: (B)

Q42.

Solution

Concept: Understanding different types of software licenses.

Solution: Software licenses define the rights and restrictions for using, modifying, and distributing software:

- **Proprietary License (or Closed Source License)**: This type of license restricts users from modifying, redistributing, or reverse-engineering the software. The source code is typically kept secret by the copyright holder, and users usually pay for the right to use the software under specific terms.
- **Closed Source License**: This is another term for a proprietary license, emphasizing that the source code is not publicly available.
- **Open Source License**: This type of license grants users the freedom to run, study, change, and distribute the software and its source code. It encourages collaboration and transparency, allowing users to modify and distribute the source code freely (often with specific conditions, such as retaining the original license). Examples include the GNU General Public License (GPL) and MIT License.
- **Trial License**: This allows users to use software for a limited period or with limited features for evaluation purposes before purchasing a full license. It doesn't grant rights to modify or freely distribute the source code.

The license type that specifically allows users to modify and distribute the source code freely is an Open Source License.

Final Answer : "Open Source License"

Answer: (C)

Q43.

Solution

Concept: Identifying the function of different network devices.

Solution: Let's review the roles of the listed network devices:

- **Hub**: Operates at Layer 1 (Physical Layer). It broadcasts incoming data packets to all connected devices in a local area network (LAN). It does not distinguish between different networks or intelligent forwarding.
- **Switch**: Operates at Layer 2 (Data Link Layer). It connects multiple devices within a single LAN and intelligently forwards data frames to the specific destination device based on MAC addresses. It still operates within one network segment.
- **Router**: Operates at Layer 3 (Network Layer). Its primary function is to connect different networks (e.g., a local area network to the internet, or different subnets within an organization). It inspects the destination IP address of incoming data packets and uses routing tables to determine the best path to forward those packets to their intended destination network.
- **NIC (Network Interface Card)**: This is a hardware component that allows a computer to connect to a network. It's the interface, not a device that forwards packets between networks.

The device that stores and forwards packets between *different networks* is a Router.

Final Answer : "Router"

Answer: (C)

Q44.

Solution

Concept: Understanding common internet protocols and their specific uses.

Solution: The internet relies on various protocols for different communication tasks:

- **FTP (File Transfer Protocol)**: Used for transferring computer files between a client and server on a computer network.
- **SMTP (Simple Mail Transfer Protocol)**: This is the standard protocol for sending (and often receiving, though POP3/IMAP are more common for client-side reception) electronic mail messages over the Internet. When you send an email, it typically uses SMTP to travel from your email client to your email server, and then from your email server to the recipient's email server.
- **HTTP (Hypertext Transfer Protocol)**: Used for accessing web pages and other web resources on the World Wide Web. **TCP (Transmission Control Protocol)**: A core protocol of the Internet Protocol Suite, lying beneath application-layer protocols like HTTP, SMTP, and FTP. It provides reliable, ordered, and error-checked delivery of a stream of octets between applications running on hosts. It's a transport layer protocol, not an application layer protocol for emails directly.

Therefore, SMTP is the protocol mainly used for sending emails over the internet.

Final Answer : "SMTP"

Answer: (B)

Q45.

Solution

Concept: Understanding common acronyms in web and networking technologies.

Solution: URL stands for Uniform Resource Locator.

- A **URL** is a specific type of Uniform Resource Identifier (URI) that provides a means of locating resources on the internet. It specifies the location of a resource (like a web page, image, or video) and the mechanism for retrieving it (e.g., HTTP, FTP).
- It consists of several parts, including the protocol (e.g., 'http://'), the domain name (e.g., 'www.example.com'), and often a path to a specific resource (e.g., '/folder/page.html').

The other options are incorrect interpretations of the acronym.

Final Answer : "Uniform Resource Locator"

Answer: (A)

Q46.

Solution

Concept: Pandas Series methods for aggregation.

Solution: Pandas Series (and DataFrames) provide convenient methods for performing common aggregation operations:

- **‘max()‘**: This is the standard Pandas method used to return the maximum value from a Series. For example, if ‘s‘ is a Series of numbers, ‘s.max()‘ will return the single largest number in that Series.
- **‘high()‘**: This is not a standard Pandas method for finding the maximum value.
- **‘maximum()‘**: This is also not a standard Pandas method for finding the maximum value. (Note: ‘numpy.maximum()‘ exists for element-wise maximum of two arrays, but not for a Series’ aggregate maximum.)
- **‘top()‘**: This is not a standard Pandas method for finding the maximum value. While ‘nlargest()‘ can find the top ’n’ values, ‘top()‘ itself is not the correct function for the single maximum.

Therefore, ‘max()‘ is the correct Pandas method to return the maximum value from a Series.

Final Answer : “max()”

Answer: (C)

Q47.

Solution

Concept: Pandas Series methods for sorting.

Solution: Pandas provides methods to sort Series and DataFrames based on their values or index:

- **‘sort_values()‘**: This is the primary method used to sort a Series (or DataFrame) by its values. By default, it sorts in ascending order. You can specify ‘ascending=False‘ for descending order. For example, ‘s.sort_values()‘ will return a new Series with values sorted in ascending order.
- **‘sort()‘**: While a generic ‘sort()‘ function exists in Python for lists, ‘sort()‘ is not the standard method for a Pandas Series or DataFrame.
- **‘order()‘**: This is not a standard Pandas method for sorting values.
- **‘arrange()‘**: This is not a standard Pandas method for sorting values. (Note: ‘arrange‘ is a function in R’s ‘dplyr‘ package for reordering rows.)

Therefore, ‘sort_values()‘ is the correct Pandas method used to sort a Series in ascending order.

Final Answer : “sort_values()”

Answer: (C)

Q48.

Solution

Concept: SQL operators for pattern matching.

Solution: SQL offers specific operators for searching for patterns within string data in a column:

- **'BETWEEN'**: This operator is used to select values within a given range (inclusive). It's typically used with numeric or date data, not for string patterns.
- **'LIKE'**: This is the SQL operator specifically used to search for a specified pattern in a column. It is often used with wildcard characters ('%' and '_') to define the pattern. For example, 'WHERE name LIKE 'A%'' would find all names starting with 'A'.
- **'IN'**: This operator is used to specify multiple possible values for a column in a 'WHERE' clause. It checks if a value matches any value in a list (e.g., 'WHERE city IN ('New York', 'London')').
- **'EXISTS'**: This operator is used in conjunction with a subquery to test for the existence of any rows in the subquery's result. It returns 'TRUE' if the subquery returns one or more rows.

Therefore, the 'LIKE' operator is used to search for a pattern in a column.

Final Answer : "LIKE"

Answer: (B)

Q49.

Solution

Concept: Understanding wildcard characters in SQL pattern matching with the ‘LIKE‘ operator.

Solution: When using the ‘LIKE‘ operator for pattern matching in SQL, two main wildcard characters are used:

- **‘%‘ (percent sign)**: This wildcard represents zero, one, or multiple characters. For example, ‘WHERE name LIKE ’A%’‘ matches any name that starts with ’A’, regardless of how many characters follow. ‘WHERE name LIKE ’%son%’‘ matches any name containing ’son’.
- **‘_‘ (underscore)**: This wildcard represents *exactly one* single character. For example, ‘WHERE name LIKE ’J_n’‘ would match ’Jan’, ’Jen’, ’Jon’, etc., but not ’Joan’ or ’John’ (if only one character is expected between ’J’ and ’n’).
- ‘*‘: This is a wildcard character used in file systems (e.g., in command line interfaces) or some programming languages for "zero or more" characters, but it is *not* a standard SQL wildcard for pattern matching with ‘LIKE‘.
- ‘#‘: This is not a standard SQL wildcard character for pattern matching with ‘LIKE‘. It’s sometimes used for comments or specific database features.

The question asks for the wildcard character that represents exactly one character. This is the underscore (‘_‘).

Final Answer : “_”

Answer: (C)

Q50.

Solution

Concept: SQL DDL (Data Definition Language) and DML (Data Manipulation Language) commands for deleting data and tables.

Solution: SQL has different commands for manipulating data and managing table structures:

- **'DROP TABLE'**: This is a DDL command that completely removes a table (including its structure, data, indexes, and constraints) from the database.
- **'DELETE FROM'**: This is a DML command used to remove rows (records) from a table based on a 'WHERE' clause. If no 'WHERE' clause is specified, it deletes all rows. However, it removes rows one by one, logs each deletion, and generates rollback information, making it slower for large tables. The table structure remains intact.
- **'REMOVE'**: This is not a standard SQL command for deleting records or tables.
- **'TRUNCATE TABLE'**: This is a DDL command (though it primarily affects data). It removes all rows from a table much faster and more efficiently than 'DELETE FROM' because it deallocates the data pages used by the table. It does not log individual row deletions, making it non-rollbackable in some systems. Crucially, it leaves the table's structure intact, unlike 'DROP TABLE'.

The question specifies removing all records *without deleting its structure*. Both 'DELETE FROM' (without a 'WHERE' clause) and 'TRUNCATE TABLE' achieve this. However, 'TRUNCATE TABLE' is generally preferred for completely emptying a table rapidly while retaining its definition, as it is a DDL command that resets the table. 'DELETE FROM' without a 'WHERE' clause also works, but 'TRUNCATE TABLE' is specifically designed for this high-performance task of emptying a table. In many contexts, 'TRUNCATE' is considered the more appropriate command for this specific scenario.

Final Answer : "TRUNCATE"

Answer: (D)

Answer Key

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