SQL Test

Section A :-Theory

1.What are Constraints in SQL ? list and briefly explain any three types of constraints.

Ans:- Constraint is a rule or condition that is applied to a table column or a set of columns to maintain the integrity, accuracy, and consistency of the data stored in the database.  Constraint type is not null, check, default, unique, primary key, foreign key.

Type of constraints:-

* NOT NULL - Restricts NULL value from being inserted into a column.
* CHECK - Verifies that all values in a field satisfy a condition.
* DEFAULT - Automatically assigns a default value if no value has been specified for the field.

2.Explain the Difference Between Delete and Truncate Commands in SQL

Ans:-

**Delete:-**

It is used to remove specific rows from a table based on a specified condition.

* DELETE is a logged operation, meaning it is recorded in the transaction log and can be rolled back if necessary.
* It can be slower when dealing with a large number of rows.
* DELETE does not reset identity or auto-increment columns.

**Truncate:-**

* TRUNCATE command is used to remove all rows from a table in a more efficient way compared to DELETE without the need for a WHERE condition.
* TRUNCATE is not as logged as DELETE, means data cannot be rolled back.
* TRUNCATE resets identity or auto-increment columns, effectively starting from the initial value.

3.what are aggregate functions in SQL? List any four with a brief explaination of each.

Ans:-Aggregate Function:-

* Aggregate functions perform calculations on a set of values and return a single value that summarizes the entire set.
* They are often used in conjunction with the GROUP BY clause to perform calculations on groups of rows.
* Common aggregate functions include SUM, AVG, COUNT, MAX, and MIN.

SUM:- Calculates the sum of values in a group.

AVG:- Calculates the average of values in a group.

COUNT:- Returns the number of rows in a group

MAX:-Returns the maximum value in a group.

4.Explain the Concept of Normalization and Why It Is Important in database Design

Ans:-Normalization is a process in database design aimed at organizing data efficiently, minimizing redundancy, and avoiding undesirable characteristics such as update anomalies, insert anomalies, and delete anomalies. The goal is to ensure that the database is logically structured so that it can store data in a way that preserves data integrity and optimizes database performance.

Normalization involves dividing a database into smaller, related tables and ensuring that data dependencies make sense. It applies a set of rules that define how tables should be structured. Each normal forms addresses specific types of redundancy and dependency issues.

There are several normal forms, each addressing different levels of redundancy and dependency problems:

1. **First Normal Form (1NF)**: Ensures that each column in a table contains atomic values.

**2.Second Normal Form (2NF)**: Ensures that the table is in 1NF and that all non-key attributes are fully functionally dependent on the primary key .

**3.Third Normal Form (3NF)**: Ensures that the table is in 2NF and that there are no transitive dependencies .

**Why Normalization Is Important**

1. **Reduces Data Redundancy**:
   * By organizing data into multiple tables and ensuring that each piece of data is stored only once, normalization eliminates the need for repeated data across the database. This helps to save storage space and reduces the chance of inconsistency in data.
2. **Prevents Data Anomalies**:
   * **Insert Anomaly**: Without normalization, inserting new records could lead to inconsistent data if certain attributes must be repeated for every new record.
   * **Update Anomaly**: If the same data appears in multiple places, updating it in one place and forgetting to update in another can result in inconsistencies.
   * **Delete Anomaly**: Deleting a record may inadvertently remove important information if that data is stored across multiple records.
3. **Improves Data Integrity**:
   * Normalization helps maintain consistency and ensures that data is logically related. By eliminating redundant data and enforcing dependencies, it makes the database less prone to errors or inconsistencies.
4. **Simplifies Database Maintenance**:
   * When data is well-organized, it is easier to update, delete, and insert data. Changes to data are localized in fewer places, reducing maintenance efforts.
5. **Enhances Query Efficiency**:
   * Properly normalized databases tend to be more efficient for querying because the structure avoids complex joins due to unnecessary duplicated data. Also, index management becomes simpler when the data is clean and well-structured.
6. **Supports Scalability**:
   * A normalized database is generally easier to scale and adapt to future requirements because it’s more flexible. New attributes can be added to existing tables without creating redundant data structures.

4.Differentiate between WHERE and HAVING clauses with Examples.

Ans:-

**1. WHERE Clause**

* **Purpose**: The WHERE clause is used to filter rows before any grouping or aggregation is done (i.e., it filters individual rows in the table).
* **Used With**: WHERE can be used in any type of SQL query, including SELECT, UPDATE, DELETE, etc. It works on individual rows of the table.
* **When to Use**: It is used when you need to filter data based on conditions applied to columns that are not aggregated (i.e., non-aggregated data).

**2. HAVING Clause**

* **Purpose**: The HAVING clause is used to filter rows after aggregation (i.e., it filters grouped data). It is typically used with aggregate functions like COUNT(), SUM(), AVG(), MAX(), and MIN().
* **Used With**: The HAVING clause is used exclusively in queries that include a GROUP BY clause or have aggregate functions.
* **When to Use**: It is used when you need to filter aggregated results, such as finding groups of data that meet certain conditions.

**WHERE CLAUSE EXAMPLE:-**

SELECT \* FROM employees

WHERE age > 30;

**HAVING CLAUSE EXAMPLE:-**

SELECT department\_id, COUNT(\*) AS employee\_count

FROM employees GROUP BY department\_id

HAVING COUNT(\*) > 5;

6.Explain the Difference Between INNER JOIN and LEFT JOIN with A scenario where each would be used

Ans:- In SQL, **INNER JOIN** and **LEFT JOIN** are types of joins that are used to combine rows from two or more tables based on a related column. They have key differences in how they handle unmatched rows:

**1. INNER JOIN**

* **Definition**: An INNER JOIN returns only the rows where there is a match in both tables based on the join condition.
* **When to Use**: Use an INNER JOIN when you only want to retrieve rows that have corresponding records in both tables.

**2. LEFT JOIN (or LEFT OUTER JOIN)**

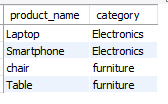
* **Definition**: A LEFT JOIN returns all rows from the left table (the first table), and the matching rows from the right table (the second table). If there is no match, NULL values are returned for columns from the right table.
* **When to Use**: Use a LEFT JOIN when you want to include all rows from the left table, regardless of whether there is a match in the right table.

Section B :-pratical

1.write a query to display all product name along with their categories.

Ans;- input:- 

Output:-



2.write a quries to find the total revenue genetrated from all producr in the order table

Ans;- input:-



Output:-

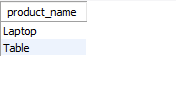


3.write a query to list the names of products where the available quantity less than 20

Ans:- input:-



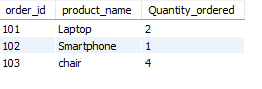
Output:-



4.write a query to display the order details using a join between product and order table

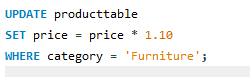
Ans;-input:- 

Output



5.write a query to update the price of all furnitute products by increasing it by 10%

input:-



Output:-

6.write a query to delete all records from the order table where the orderdate is earlier than November 2,2024

input:-



Output:-