assignment=12

July 27, 2023

1 Q1. What is a database? Differentiate between SQL and NoSQL databases.

A database is a collection of structured data that is organized in a way that enables efficient storage, retrieval, and manipulation of information. Databases are commonly used in software applications, websites, and other information systems to store and manage large amounts of data.

SQL (Structured Query Language) and NoSQL (Not Only SQL) are two different types of databases that have different architectures, data models, and query languages.

SQL databases are relational databases that store data in tables with rows and columns. They use a standardized query language, SQL, to manipulate and retrieve data. SQL databases are ideal for structured data with well-defined relationships between different data points. Some examples of SQL databases include MySQL, PostgreSQL, and Oracle.

NoSQL databases, on the other hand, are non-relational databases that store data in a more flexible way. They can handle unstructured, semi-structured, and structured data, and they do not require a fixed schema or data model. NoSQL databases use different data models, such as key-value, document, graph, or column-family. Examples of NoSQL databases include MongoDB, Cassandra, and Neo4j.

The choice between SQL and NoSQL databases depends on the specific needs of the application or system. SQL databases are good for handling structured data that follows a specific schema, while NoSQL databases are more flexible and scalable for handling unstructured or semi-structured data. NoSQL databases are often preferred for web-scale applications and big data projects, while SQL databases are commonly used in traditional enterprise applications.

2 Q2. What is DDL? Explain why CREATE, DROP, ALTER, and TRUNCATE are used with an example.

DDL stands for Data Definition Language, which is a set of SQL commands that are used to define and manage the structure of database objects such as tables, indexes, views, and constraints. DDL statements are used to create, modify, and delete database objects, as well as to grant or revoke permissions on those objects.

Here are some examples of DDL statements and their uses:

CREATE: The CREATE statement is used to create new database objects, such as tables, views, indexes, and stored procedures. For example, the following SQL statement creates a new table called “users” with three columns: “id”, “name”, and “email”:

[ ]:
```
CREATE TABLE users (
  id INT PRIMARY KEY,
  name VARCHAR(50),
  email VARCHAR(100)
);
```

DROP: The DROP statement is used to remove an existing database object. For example, the following SQL statement drops the “users” table that was created in the previous example:

[ ]:
```
DROP TABLE users;
```

ALTER: The ALTER statement is used to modify an existing database object, such as a table or a view. For example, the following SQL statement adds a new column called “age” to the “users” table:

ALTER TABLE users ADD COLUMN age INT;

TRUNCATE: The TRUNCATE statement is used to delete all rows from a table, but the table structure remains intact. For example, the following SQL statement deletes all rows from the “users” table:

[ ]:
```
TRUNCATE TABLE users;
```

3 Q3. What is DML? Explain INSERT, UPDATE, and DELETE with an example.

DML stands for Data Manipulation Language, which is a set of SQL commands used to manipulate and modify the data stored in a database. DML statements are used to insert, update, delete, and retrieve data from a database. Here are some examples of DML statements and their uses:

INSERT: The INSERT statement is used to add new rows to a table. For example, the following SQL statement inserts a new record into a table named “users” with the values “sonal jinjala” for the name column, “sonaljinjala@gmail.com” for the email column, and 21 for the age column:

[ ]:
```
INSERT INTO users (name, email, age) VALUES ('sonal jinjala',␣
 ↪'sonaljinjala@gmail.com', 21);
```

UPDATE: The UPDATE statement is used to modify existing data in a table. For example, the following SQL statement updates the email of the user with the name “sonal jinjala” to “sonaljinjala@gmail.com”:

[ ]:
```
UPDATE users SET email = 'sonaljinjala@gmail.com' WHERE name = 'sonal jinjala';
```

DML stands for Data Manipulation Language, which is a set of SQL commands used to manipulate and modify the data stored in a database. DML statements are used to insert, update, delete, and retrieve data from a database. Here are some examples of DML statements and their uses:

INSERT: The INSERT statement is used to add new rows to a table. For example, the following SQL statement inserts a new record into a table named “users” with the values “sonal jinjala” for the name column, “sonaljinjala@gmail.com” for the email column, and 21 for the age column: sql Copy

code INSERT INTO users (name, email, age) VALUES (‘sonal jinjala’, ‘sonaljinjala@gmail.com’, 21); UPDATE: The UPDATE statement is used to modify existing data in a table. For example, the following SQL statement updates the email of the user with the name “sonal jinjala” to “sonaljinjala@gmail.com”: sql Copy code UPDATE users SET email = ‘sonaljinjala@gmail.com’ WHERE name = ‘sonal jinjala’; DELETE: The DELETE statement is used to remove existing data from a table. For example, the following SQL statement deletes the user with the name “sonal jinjala” from the “users” table:

[ ]:
```
DELETE FROM users WHERE name = 'sonal jinjala';
```

4 Q4. What is DQL? Explain SELECT with an example.

DQL stands for Data Query Language, which is a subset of SQL used to retrieve data from a database. DQL statements are used to query the database and retrieve specific data based on specified criteria. The most commonly used DQL statement is SELECT.

SELECT is used to retrieve data from one or more tables in a database. It allows you to specify the columns to retrieve, the table or tables to retrieve them from, and any conditions that the retrieved data must meet. Here’s an example of a SELECT statement:

[ ]:
```
SELECT name, email FROM users WHERE age > 21;
```

5 Q5. Explain Primary Key and Foreign Key.

A primary key is a column or set of columns in a table that uniquely identifies each row or record in that table. It ensures that each row in the table is uniquely identifiable and is used to enforce data integrity constraints. A primary key column cannot contain null values and must have a unique value for each row.

Here’s an example of a table with a primary key:

sql

[ ]:
```
CREATE TABLE students (
    id INT PRIMARY KEY,
    name utran(50),
    email utran(100),
    age INT
);
```

A foreign key is a column or set of columns in a table that refers to the primary key of another table. It establishes a link between two tables, which allows data to be retrieved from multiple tables using a single query. A foreign key is used to enforce referential integrity constraints, which ensure that the data in the foreign key column matches the data in the primary key column of the other table.

Here’s an example of a table with a foreign key:

[ ]:
```
CREATE TABLE courses (
    id INT PRIMARY KEY,
    name utran(50),
    instructor_id INT,
    FOREIGN KEY (instructor_id) REFERENCES instructors(id)
);
```

6 Q6. Write a python code to connect MySQL to python. Explain the cursor() and execute() method.

To connect MySQL to Python, we need to have the mysql-connector-python library installed. We can install it using pip, like this:

[2]:
```
pip install mysql-connector-python
```

```
Requirement already satisfied: mysql-connector-python in
/opt/conda/lib/python3.10/site-packages (8.1.0)
Requirement already satisfied: protobuf<=4.21.12,>=4.21.1 in
/opt/conda/lib/python3.10/site-packages (from mysql-connector-python) (4.21.8)
Note: you may need to restart the kernel to use updated packages.
```

Once we have installed the library, we can connect to a MySQL database using the following Python code:

[ ]:
```
import mysql.connector

mydb = mysql.connector.connect(
  host="localhost",
  user="username",
  password="password",
  database="database"
)

mycursor = mydb.cursor()
```

In this code, we first import the mysql.connector module and use the connect() method to establish a connection to the MySQL database. We need to provide the database connection details, such as the host, username, password, and database name.

After establishing the connection, we create a cursor object using the cursor() method of the database connection object. The cursor object allows us to execute SQL statements on the database.

The execute() method is used to execute an SQL statement on the database using the cursor object. It takes an SQL statement as its parameter and returns the result of the statement, if any. Here's an example of how to use the execute() method to execute a SELECT statement:

[ ]:
```
sql = "SELECT * FROM customers"
```

```
mycursor.execute(sql)

myresult = mycursor.fetchall()

for x in myresult:
  print(x)
```

In this code, you first define an SQL SELECT statement and store it in the sql variable. You then call the execute() method on the cursor object, passing the sql variable as its parameter. This executes the SELECT statement on the database and returns the result.

The fetchall() method is used to fetch all the rows returned by the SELECT statement. It returns a list of tuples, where each tuple represents a row in the result set. You can then loop through the result set and print each row, as shown in the code above.

In summary, the cursor() method is used to create a cursor object, which allows you to execute SQL statements on the database. The execute() method is used to execute an SQL statement on the database using the cursor object, and the fetchall() method is used to retrieve the results of the SQL statement.

7 Q7. Give the order of execution of SQL clauses in an SQL query.

The order of execution of SQL clauses in an SQL query can be summarized as follows:

FROM clause: The FROM clause specifies the table or tables from which the data will be retrieved. If multiple tables are used, they may be joined using the JOIN clause.

WHERE clause: The WHERE clause filters the data based on a specified condition or set of conditions. It is used to retrieve only the rows that meet the specified criteria.

GROUP BY clause: The GROUP BY clause is used to group the result set by one or more columns. It is often used in conjunction with aggregate functions, such as COUNT, SUM, AVG, etc.

HAVING clause: The HAVING clause is used to filter the result set based on a specified condition or set of conditions. It is similar to the WHERE clause, but it is applied after the data has been grouped by the GROUP BY clause.

SELECT clause: The SELECT clause specifies the columns that will be included in the result set. It can also be used to apply functions or expressions to the data, such as CONCAT, DATE_FORMAT, etc.

ORDER BY clause: The ORDER BY clause is used to sort the result set based on one or more columns. It can sort the data in ascending or descending order.

LIMIT clause: The LIMIT clause is used to limit the number of rows returned by the query. It is often used in conjunction with the ORDER BY clause to retrieve the top N rows based on a specified criteria.

It's important to note that not all clauses are required in every query. The basic syntax of an SQL query is:

[ ]:
```
SELECT column1, column2, ...
FROM table_name
WHERE condition;
```