### **Week 6: Introduction to SQL**

#### **Overview**

This week, we will focus on SQL (Structured Query Language), a powerful tool for managing and manipulating relational databases. SQL is essential for data analytics as it allows you to efficiently query, update, and analyze data stored in relational databases. By the end of this week, you will be able to write basic SQL queries, understand the structure of relational databases, and integrate SQL with Python.

#### **Learning Objectives**

By the end of this module, students will be able to:

1. Understand the basics of SQL and relational databases.
2. Write SQL queries to retrieve, insert, update, and delete data.
3. Use SQL joins and subqueries to manipulate data from multiple tables.
4. Integrate SQL with Python for data analysis.

#### **Basics of SQL and Relational Databases**

**SQL**: A standard language for accessing and manipulating databases. SQL is used to perform tasks such as querying data, updating records, and managing database structures.

**Relational Database**: A type of database that stores data in tables with rows and columns. Each table represents a different entity, and relationships between tables are established through foreign keys.

**Key SQL Concepts**:

* **Table**: A collection of related data organized in rows and columns.
* **Row**: A single record in a table.
* **Column**: A single field in a table.
* **Primary Key**: A unique identifier for a record in a table.
* **Foreign Key**: A field in one table that uniquely identifies a row in another table, establishing a relationship between the two tables.

#### **Writing SQL Queries**

**SELECT Statement**: Retrieves data from one or more tables.

SELECT column1, column2

FROM table\_name

WHERE condition;

**INSERT Statement**: Adds new records to a table.

INSERT INTO table\_name (column1, column2)

VALUES (value1, value2);

**UPDATE Statement**: Modifies existing records in a table.

UPDATE table\_name

SET column1 = value1, column2 = value2

WHERE condition;

**DELETE Statement**: Removes records from a table.

DELETE FROM table\_name

WHERE condition;

#### **SQL Joins and Subqueries**

**Joins**: Combine rows from two or more tables based on a related column.

* **INNER JOIN**: Returns records with matching values in both tables.

SELECT table1.column1, table2.column2

FROM table1

INNER JOIN table2

ON table1.common\_column = table2.common\_column;

* **LEFT JOIN**: Returns all records from the left table and matched records from the right table.

SELECT table1.column1, table2.column2

FROM table1

LEFT JOIN table2

ON table1.common\_column = table2.common\_column;

* **RIGHT JOIN**: Returns all records from the right table and matched records from the left table.

SELECT table1.column1, table2.column2

FROM table1

RIGHT JOIN table2

ON table1.common\_column = table2.common\_column;

* **FULL JOIN**: Returns all records when there is a match in either table.

SELECT table1.column1, table2.column2

FROM table1

FULL JOIN table2

ON table1.common\_column = table2.common\_column;

**Subqueries**: Nested queries used to perform operations that need multiple steps.

Example:

SELECT column1

FROM table\_name

WHERE column2 = (SELECT column2 FROM table\_name WHERE condition);

#### **Integrating SQL with Python**

Using the sqlite3 module, you can integrate SQL queries within Python to manage and analyze databases.

**Example**:

import sqlite3

# Connect to the database

conn = sqlite3.connect('example.db')

# Create a cursor object

cursor = conn.cursor()

# Execute a SQL query

cursor.execute('SELECT \* FROM table\_name')

# Fetch and print all results

results = cursor.fetchall()

for row in results:

print(row)

# Close the connection

conn.close()

#### **Learning Activities**

To reinforce your understanding of this week's content, complete the following activities:

1. **Reading Assignment**: Read Chapter 6 of "Data Analytics Made Accessible" by Anil Maheshwari, focusing on SQL and relational databases.
2. **Video Lecture**: Watch the video "Introduction to SQL" on YouTube to see practical examples of SQL queries.
3. **Hands-On Exercise**: Download a sample database and write SQL queries to perform various data manipulation tasks.

#### **Discussion Questions**

Participate in the class discussion by answering the following questions:

1. What are the advantages of using SQL for data manipulation?
2. How can SQL joins help in combining data from multiple tables?
3. What are some common challenges you might face when integrating SQL with Python?

#### **Summary**

This week, we explored the basics of SQL and relational databases. We learned how to write SQL queries to retrieve, insert, update, and delete data. We also covered SQL joins and subqueries for manipulating data from multiple tables and integrated SQL with Python for data analysis. These skills are essential for managing and analyzing data stored in relational databases.

#### **Additional Resources**

* **Book**: "SQL for Data Scientists" by Renee M. P. Teate.
* **Website**: Visit W3Schools SQL Tutorial for interactive tutorials on SQL.
* **Tutorial**: Follow the tutorial "SQL for Data Analysis" on [Kaggle](https://www.kaggle.com/) to practice with real datasets.

#### **Homework**

1. Write a short essay (300-500 words) on the importance of SQL in data analytics.
2. Complete the hands-on exercise and submit your SQL queries and results through the course portal.