1. **Write a Python program to create a list, a dictionary, and a set. Perform basic operations like adding, removing, and modifying elements**

**# Creating and manipulating a list**

**my\_list = [1, 2, 3, 4, 5]**

**print("Initial list:", my\_list)**

**# Adding an element to the list**

**my\_list.append(6)**

**print("List after adding an element:", my\_list)**

**# Removing an element from the list**

**my\_list.remove(3)**

**print("List after removing an element:", my\_list)**

**# Modifying an element in the list**

**my\_list[1] = 10**

**print("List after modifying an element:", my\_list)**

**# Creating and manipulating a dictionary**

**my\_dict = {'name': 'Alice', 'age': 25, 'city': 'New York'}**

**print("\nInitial dictionary:", my\_dict)**

**# Adding a new key-value pair to the dictionary**

**my\_dict['job'] = 'Engineer'**

**print("Dictionary after adding a new key-value pair:", my\_dict)**

**# Removing a key-value pair from the dictionary**

**del my\_dict['age']**

**print("Dictionary after removing a key-value pair:", my\_dict)**

**# Modifying a value in the dictionary**

**my\_dict['city'] = 'San Francisco'**

**print("Dictionary after modifying a value:", my\_dict)**

**# Creating and manipulating a set**

**my\_set = {1, 2, 3, 4, 5}**

**print("\nInitial set:", my\_set)**

**# Adding an element to the set**

**my\_set.add(6)**

**print("Set after adding an element:", my\_set)**

**# Removing an element from the set**

**my\_set.discard(3)**

**print("Set after removing an element:", my\_set)**

**# Attempting to remove a non-existing element**

**my\_set.discard(10)**

**print("Set after attempting to remove a non-existing element:", my\_set)**

**Output**

**Initial list: [1, 2, 3, 4, 5]**

**List after adding an element: [1, 2, 3, 4, 5, 6]**

**List after removing an element: [1, 2, 4, 5, 6]**

**List after modifying an element: [1, 10, 4, 5, 6]**

**Initial dictionary: {'name': 'Alice', 'age': 25, 'city': 'New York'}**

**Dictionary after adding a new key-value pair: {'name': 'Alice', 'age': 25, 'city': 'New York', 'job': 'Engineer'}**

**Dictionary after removing a key-value pair: {'name': 'Alice', 'city': 'New York', 'job': 'Engineer'}**

**Dictionary after modifying a value: {'name': 'Alice', 'city': 'San Francisco', 'job': 'Engineer'}**

**Initial set: {1, 2, 3, 4, 5}**

**Set after adding an element: {1, 2, 3, 4, 5, 6}**

**Set after removing an element: {1, 2, 4, 5, 6}**

**Set after attempting to remove a non-existing element: {1, 2, 4, 5, 6}**

1. **The intern will learn foundational Python concepts such as variables, data types, loops, and functions.**

**# Loop through a list**

**colors = ["red", "green", "blue"]**

**for color in colors:**

**print(color)**

**# Loop through a range of numbers**

**for i in range(5):**

**print(i)**

**Output**

**red**

**green**

**blue**

**0**

**1**

**2**

**3**

**4**

**function**

**# Define a function**

**def greet(name):**

**return f"Hello, {name}!"**

**# Call the function**

**message = greet("Alice")**

**print(message) # Output: Hello, Alice!**

**# Function with default argument**

**def greet(name="Guest"):**

**return f"Hello, {name}!"**

**print(greet()) # Output: Hello, Guest!**

**# Output:**

**Hello, Guest!**

**1. Literals**

**Concept: Literals are fixed values used in SQL statements. They represent constant values and can be of various types:**

* **String Literals: Enclosed in single quotes '**

**Sql**

**SELECT \* FROM employees WHERE name = 'Alice';**

**Numeric Literals: Numeric values can be integers or floating-point numbers.**

**Sql**

**SELECT \* FROM products WHERE price > 50;**

**Date Literals: Dates are enclosed in single quotes and follow the YYYY-MM-DD format.**

**Sql**

**SELECT \* FROM orders WHERE order\_date = '2024-08-11';**

**2. Data Types**

**Concept: Data types define the kind of data that can be stored in a column. Here are some common MySQL data types:**

* **Numeric Data Types:**
  + **INT: Integer values.**
  + **FLOAT: Floating-point numbers.**
  + **DOUBLE: Double precision floating-point numbers.**
  + **DECIMAL: Exact numeric values.**

**CREATE TABLE products (**

**product\_id INT,**

**price DECIMAL(10, 2)**

**);**

**String Data Types:**

* **CHAR: Fixed-length string.**
* **VARCHAR: Variable-length string.**
* **TEXT: Large text data.**

**CREATE TABLE users (**

**username VARCHAR(50),**

**bio TEXT**

**);**

**Date and Time Data Types:**

* **DATE: Date values.**
* **DATETIME: Date and time values.**
* **TIMESTAMP: Date and time values with timezone.**

**CREATE TABLE events (**

**event\_date DATE,**

**event\_timestamp TIMESTAMP**

**);**

**Boolean Data Types:**

**BOOLEAN: Stores TRUE or FALSE values**

**CREATE TABLE features (**

**is\_active BOOLEAN**

**);**

**3. NULLs**

**Concept: NULL represents the absence of a value. It's not the same as an empty string or zero.**

* **Inserting NULL Values:**

**INSERT INTO users (username, bio) VALUES ('JohnDoe', NULL);**

**Checking for NULL Values: Use IS NULL or IS NOT NULL in queries.**

**SELECT \* FROM users WHERE bio IS NULL;**

**Handling NULLs: Use functions like COALESCE to provide default values for NULL.**

**SELECT username, COALESCE(bio, 'No bio available') AS bio FROM users;**

**4. Comments**

**Concept: Comments are used to annotate SQL code. They are ignored by the SQL engine.**

* **Single-Line Comments: Use -- or # for single-line comments.**

**-- This is a single-line comment**

**SELECT \* FROM employees; -- This comment is at the end of a line**

**# This is another single-line comment**

**SELECT \* FROM departments;**

**Multi-Line Comments: Use /\* \*/ for multi-line comments.**

**/\***

**This is a multi-line comment.**

**It can span multiple lines.**

**\*/**

**SELECT \* FROM orders;**

**5. SQL Syntax and Structure**

**SQL Syntax and Structure**

SQL (Structured Query Language) is used to communicate with databases. It allows users to perform various operations such as querying data, updating data, and managing the database schema.

**Basic SQL Syntax:**

**SELECT column1, column2**

**FROM table\_name**

**WHERE condition;**

SELECT: Specifies which columns to retrieve.

FROM: Specifies the table to retrieve data from.

WHERE: Filters the results based on a condition.

**2. Literals**

Literals are fixed values used in SQL queries. They can be of various types:

* **String Literals:** Enclosed in single quotes (')

SELECT \* FROM employees WHERE name = 'John Doe';

**Numeric Literals:** Represent numbers directly.

SELECT \* FROM products WHERE price > 100;

**Date Literals:** Represent date values enclosed in single quotes.

SELECT \* FROM orders WHERE order\_date = '2024-08-11';

**3. Datatypes**

Understanding datatypes is crucial for defining the nature of data that can be stored in each column.

* **Numeric Types:**
  + INT: Integer values.
  + FLOAT / DOUBLE: Floating-point numbers.
  + DECIMAL: Exact numeric values with a fixed decimal point.
* **String Types:**
  + VARCHAR(n): Variable-length string, where n is the maximum length.
  + CHAR(n): Fixed-length string.
* **Date and Time Types:**
  + DATE: Date values (YYYY-MM-DD).
  + DATETIME: Date and time values (YYYY-MM-DD HH:MM

).

* + TIMESTAMP: Date and time with automatic updating.
* **Other Types:**
  + BOOLEAN: TRUE or FALSE values.
  + BLOB: Binary Large Object, used for storing binary data.

**4. NULLs**

In SQL, NULL represents missing or undefined values. It is different from an empty string or zero.

**Checking for NULL:**

SELECT \* FROM employees WHERE manager IS NULL;

**Checking NOT NULL:**

SELECT \* FROM employees WHERE manager IS NOT NULL;

**Handling NULLs in Functions:**

* IFNULL(value, alternative): Returns alternative if value is NULL.
* COALESCE(value1, value2, ...): Returns the first non-NULL value.

sql

SELECT IFNULL(commission, 0) AS commission FROM employees;

**5. Comments in SQL**

Comments are used to annotate SQL code. They can be used for documentation or to temporarily disable parts of the code.

* **Single-Line Comments:**
  + Using --:

-- This is a single-line comment

SELECT \* FROM employees;

Using # (MySQL-specific):

# This is also a single-line comment

SELECT \* FROM employees;

**Multi-Line Comments:**

/\* This is a multi-line comment

It can span multiple lines \*/

SELECT \* FROM employees;

**Example Queries**

Here’s how you might use these elements in practice:

* **Query with Literals:**

*SELECT name, salary*

*FROM employees*

*WHERE department = 'Sales' AND salary > 50000;*

**Creating a Table with Different Datatypes:**

*CREATE TABLE employees (*

*id INT AUTO\_INCREMENT PRIMARY KEY,*

*name VARCHAR(100) NOT NULL,*

*salary DECIMAL(10, 2),*

*hire\_date DATE,*

*manager\_id INT,*

*is\_active BOOLEAN DEFAULT TRUE*

*);*

**Inserting Data with NULL Values:**

*INSERT INTO employees (name, salary, hire\_date, manager\_id)*

*VALUES ('Jane Doe', 60000.00, '2024-08-11', NULL);*

By familiarizing themselves with these SQL elements, the intern will build a solid foundation for working with MySQL databases and writing effective SQL queries.

**5. Basic Level HTML structure and styling.**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="UTF-8">**

**<meta name="viewport" content="width=device-width, initial-scale=1.0">**

**<title>Simple Webpage</title>**

**<link rel="stylesheet" href="styles.css">**

**</head>**

**<body>**

**<header>**

**<h1>My Simple Webpage</h1>**

**<nav>**

**<ul>**

**<li><a href="#home">Home</a></li>**

**<li><a href="#about">About</a></li>**

**<li><a href="#contact">Contact</a></li>**

**</ul>**

**</nav>**

**</header>**

**<main id="home">**

**<section>**

**<h2>Welcome to My Webpage</h2>**

**<p>This is a simple webpage designed to demonstrate HTML and CSS.</p>**

**</section>**

**<section id="about">**

**<h2>About Me</h2>**

**<p>Hello! I am learning web development and this is my first webpage.</p>**

**</section>**

**</main>**

**<footer id="contact">**

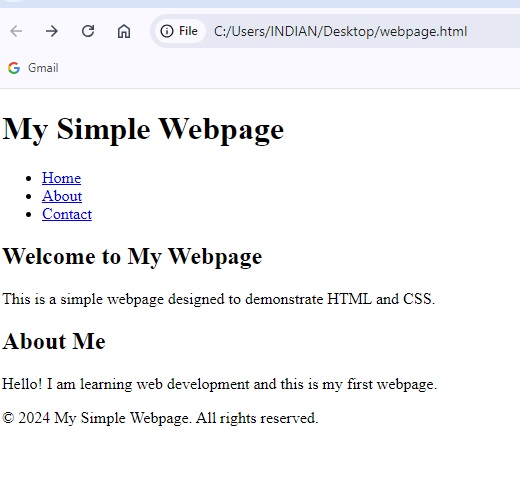
**<p>&copy; 2024 My Simple Webpage. All rights reserved.</p>**

**</footer>**

**</body>**

**</html>**

**Output:**



**Css**

**/\* Reset some default browser styles \*/**

**body, h1, h2, p, ul {**

**margin: 0;**

**padding: 0;**

**}**

**body {**

**font-family: Arial, sans-serif;**

**line-height: 1.6;**

**background-color: #f4f4f4;**

**color: #333;**

**}**

**header {**

**background: #333;**

**color: #fff;**

**padding: 10px 0;**

**text-align: center;**

**}**

**header h1 {**

**margin-bottom: 10px;**

**}**

**nav ul {**

**list-style: none;**

**padding: 0;**

**}**

**nav ul li {**

**display: inline;**

**margin: 0 10px;**

**}**

**nav ul li a {**

**color: #fff;**

**text-decoration: none;**

**}**

**nav ul li a:hover {**

**text-decoration: underline;**

**}**

**main {**

**padding: 20px;**

**}**

**section {**

**margin-bottom: 20px;**

**}**

**footer {**

**background: #333;**

**color: #fff;**

**text-align: center;**

**padding: 10px 0;**

**position: fixed;**

**width: 100%;**

**bottom: 0;**

**}**

**Explanation:**

1. **HTML Structure:**
   * **The <header> contains the title and navigation links.**
   * **The <main> section holds different sections of content (home, about, contact).**
   * **The <footer> displays a copyright notice.**
2. **CSS Styling:**
   * **Resetting margins and paddings for some elements to ensure consistent styling across browsers.**
   * **Body has a background color and default text color.**
   * **Header has a dark background with white text, and the navigation links are styled to be inline with hover effects.**
   * **Main has padding for spacing.**
   * **Footer is fixed at the bottom of the page with a matching background color to the header.**