**Week 1: Introduction & Foundational Skills (Focus on Project Relevance)**

This week, we'll be diving into the exciting world of SQL and databases! We'll explore what SQL is used for, how it benefits web applications, and the building blocks of databases: tables, columns, and data types. But most importantly, we'll get our hands dirty by creating a basic database structure for our upcoming Expense Tracker project!

**Learning Objectives:**

Understand the purpose and applications of SQL, particularly for web applications. Identify the fundamental components of a database: tables, columns, and data types. Design a basic database schema for our Expense Tracker project. Instructions This assignment is designed to be completed in approximately 2 hours.

**What you'll need:**

Access to a computer with internet access A text editor (Microsoft Word document) Drawing software (e.g. Draw.io, visual paradigm) for the bonus question.

**Submission:**

Save your completed assignment as a document (e.g., .docx, pdf) Submit your document through the designated course platform.

**Part 1: Understanding SQL (30 minutes)**

**Question 1. Research**

Use online resources like websites or PowerPoint slides.

**1.1.** In a single Word document, summarize your findings in a short paragraph (3-5 sentences). Web Applications:

Imagine a dynamic website like an online store. How do you think SQL plays a role in managing data behind the scenes? Consider how product information, user accounts, and order details might be stored and accessed.

In an online store, SQL plays a critical role in managing data related to products, users, and orders. It stores product details (e.g., name, price, inventory), user accounts (e.g., emails, passwords), and order information (e.g., items purchased, order status). SQL queries enable product searches, user authentication, and the processing of orders. It supports key database operations like creating, reading, updating, and deleting data (CRUD) and ensures data integrity through transactions. SQL allows dynamic management and scaling of the store as the website grows in complexity.

**1.2.** Write a short explanation (3-5 sentences) in your document about the role of SQL in web applications.

SQL plays a fundamental role in web applications by managing the storage, retrieval, and manipulation of data. It powers key operations such as handling user accounts, storing product information, processing orders, and ensuring the integrity of transactions. SQL queries enable efficient data access, allowing web applications to deliver dynamic content, process user inputs, and update information in real-time. Through SQL, web apps maintain structured data management and scalability, supporting seamless user experiences.

**1.3.** List 3 benefits of using SQL for web applications.

1. **Efficient Data Management**: SQL allows structured storage, retrieval, and manipulation of large datasets, making it ideal for handling user accounts, products, and orders in web applications.
2. **Data Integrity and Security**: SQL supports features like constraints and transactions, ensuring data accuracy and consistency, while also enabling secure access control for sensitive information like user credentials.
3. **Scalability and Flexibility**: SQL databases are designed to scale with increasing amounts of data and traffic, while offering powerful querying capabilities to dynamically filter, sort, and retrieve data as needed by the web application.

**1.4.** Think about efficiency, data organization, and data retrieval capabilities. Briefly explain each benefit in your document (1-2 sentences per benefit).

**Efficiency**: Streamlining processes and workflows can reduce time spent on repetitive tasks, leading to quicker decision-making and improved productivity.

**Data Organization**: Properly structured data ensures that information is easy to navigate, categorize, and maintain, preventing redundancy and confusion.

**Data Retrieval Capabilities**: Optimized retrieval systems enable fast access to critical data, enhancing responsiveness and reducing the time needed to locate necessary information.

**1.5.** List any 3 Database Management Systems.

1. **MySQL**: An open-source relational database management system widely used for web applications and data storage.
2. **PostgreSQL**: A powerful, open-source object-relational database system known for its robustness, scalability, and support for complex queries.
3. **Microsoft SQL Server**: A relational database management system developed by Microsoft, offering a wide range of data management, analytics, and integration tools.

**Part 2: Database Fundamentals (45 minutes)**

**Question 2.1: Tables**

Think about how data is organized in rows and columns. In your document, define a database table and explain its similarity to a spreadsheet (2-3 sentences).

A **database table** is a structured format used to store data, organized into rows (records) and columns (fields), where each column represents a specific attribute, and each row contains a set of related values for those attributes. It is similar to a **spreadsheet**, where data is also arranged in rows and columns, with each cell holding individual data, allowing for easy data organization and retrieval. The key difference is that database tables are often part of a larger relational database, optimized for querying and data integrity.

**Question 2.2: Columns**

Consider different types of data like text, numbers, and dates. Define "columns" and provide an example with an explanation (2-3 sentences) in your document. Data Types: Why are data types important in a database? Briefly explain 3 common data types (e.g., Text, Number, Date).

In a database, **columns** are the vertical divisions within a table that hold data of the same type. Each column represents a specific attribute or field for the records in that table. For example, in a table of employee information, there might be columns for "Employee Name" (text), "Age" (number), and "Hire Date" (date).

#### Example:

| **Employee Name** | **Age** | **Hire Date** |
| --- | --- | --- |
| John Doe | 30 | 2022-01-15 |
| Jane Smith | 25 | 2023-03-10 |

Here, "Employee Name" is a text column, "Age" is a number column, and "Hire Date" is a date column.

### Importance of Data Types

Data types are crucial in a database because they define the kind of data that can be stored in each column, ensuring consistency and preventing errors. Proper use of data types improves database performance and integrity.

#### 3 Common Data Types:

1. **Text**: Stores alphanumeric characters like names or descriptions (e.g., "John Doe").
2. **Number**: Stores numeric values, which can be used for calculations (e.g., "25" for age).
3. **Date**: Stores dates and times, which can be used for sorting or filtering by time periods (e.g., "2023-03-10").

**Question 2.3: Data Types**

Think about how data types ensure data integrity and efficient storage. Explain the importance of data types and provide brief explanations of 3 common types (2-3 sentences each) in your document.

Data types play a critical role in programming by ensuring that data is stored efficiently and that operations on the data are valid and meaningful. By specifying a data type, you define the kind of data a variable can hold, which helps prevent errors and optimizes memory usage. This also contributes to data integrity, ensuring that data behaves in predictable ways according to its type.

### 3 Common Data Types:

1. **Integer (int)**: This data type stores whole numbers, both positive and negative. Integers are typically used when precise counting or calculations involving whole numbers are needed. They are efficient in terms of memory usage, as they occupy a fixed size in memory.
2. **Float (floating-point)**: Floats represent numbers with decimals, allowing for the representation of real numbers. They are essential when precision is needed in calculations, such as in scientific computations or financial applications, though they can be more memory-intensive than integers.
3. **String**: A string is a sequence of characters used to represent text. It is useful for storing and manipulating human-readable data like names, addresses, or sentences. Strings are more complex than numbers and often take up more memory due to their variable length.

**Part 3: Expense Tracker Database Design (45 minutes)**

**3.1. Planning:** We'll be building an Expense Tracker application. What kind of data do you think we'll need to track? List at least 5 data points relevant to our project.

* Consider information like expense amount, date, and category.
* List your identified data points in your document.

For an Expense Tracker application, we'll need to track the following key data points:

1. **Expense Amount** – The monetary value of the expense.
2. **Expense Category** – The type of expense (e.g., groceries, entertainment, rent, utilities, transportation).
3. **Date of Expense** – When the expense occurred.
4. **Payment Method** – The method used to pay for the expense (e.g., credit card, cash, bank transfer).
5. **Expense Description** – A brief note or description of the expense (e.g., "Dinner at restaurant," "Monthly rent").

Other potential data points could include the location of the expense, currency, and whether the expense is recurring.

**3.2. Tables:** Considering the data points you listed, design a basic database schema with one main table (likely named "Expenses").

* Define the columns needed for this table.
* Assign appropriate data types to each column based on the kind of data it will hold. (e.g., amount: number, date: date, category: text)

In your document, create a table structure that includes:

* Table name (e.g., Expenses)
* Column names (e.g., expense\_id, amount, date, category)
* Data type for each column (e.g., INT, DECIMAL, DATE, TEXT)

Here's a basic database schema for the "Expenses" table:

### Table Name: Expenses

| **Column Name** | **Data Type** | **Description** |
| --- | --- | --- |
| expense\_id | INT | Unique identifier for each expense. |
| amount | DECIMAL(10,2) | The amount of the expense. |
| category | TEXT | The category of the expense. |
| date | DATE | The date when the expense occurred. |
| payment\_method | TEXT | The method used to pay for the expense. |
| description | TEXT | A brief description of the expense. |

#### Notes:

* **expense\_id**: Should be set as the primary key and auto-incremented if using SQL databases.
* **amount**: DECIMAL type with precision and scale, allowing for monetary values.
* **category**: TEXT type to accommodate various expense categories.
* **date**: DATE type to store the date of the expense.
* **payment\_method**: TEXT type to store the payment method.
* **description**: TEXT type to provide a brief description of the expense.

**Bonus:**

Sketch a simple Entity Relational Diagram (ERD) of your table structure, including column names and data types.

Use drawing software or a simple table format to visually represent your schema.

\*\* Remember: There might be multiple ways to design your database schema. The goal is to understand the concepts and create a logical structure to store our expense tracking data.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *WELLDONE* \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_