**What is SQL?**

SQL (pronounced "ess-cue-el" or sometimes "sequel") stands for Structured Query Language. It’s a standard programming language used to communicate with and manage data in relational databases.

**A screenshot of a computer

AI-generated content may be incorrect.What You Can Do With SQL:**

**Why It’s Important:**

* Used in systems like MySQL, SQL Server, PostgreSQL, SQLite, and Oracle.
* Essential for backend development, data analysis, business intelligence, and more.
* Forms the foundation for most data-driven web applications.

**What is Entity Relationship Model?**

An **Entity-Relationship Model (ER Model)** is a **conceptual framework** used in database design to visually represent data and its relationships within a system. It helps in structuring and organizing data before creating a database.

**Key Components of an ER Model:**

1. **Entities** – Objects or things in the system (e.g., Student, Car, Employee).
2. **Attributes** – Characteristics of an entity (e.g., Student **has** Name, Age, Student\_ID).
3. **Primary Key** – A unique identifier for an entity (e.g., Student\_ID uniquely identifies a student).
4. **Relationships** – Associations between entities (e.g., Student **enrolls in** Course).
5. **Cardinality** – Defines the number of associations (e.g., One-to-One, One-to-Many, Many-to-Many).

Each Entity is a table (or relation) and each relationship is represented by a Primary Key - Foreign key link between the relations.

**Relation Rules**

Below is an example of an Order table explaining relations rules:

A diagram of a number

AI-generated content may be incorrect.

**Data Integrity**

It refers to the accuracy, consistency, and reliability of data throughout its lifecycle. It ensures that data remains correct, complete, and unaltered during storage, retrieval, and processing.

Using a primary key enforces Entity integrity i.e a ‘row’ can only appear once in a table.

A close-up of a number

AI-generated content may be incorrect.

Each row has a unique set of values due to the uniqueness of the primary key.

A foreign key references another value in a different table.

A diagram of a customer and foreign key

AI-generated content may be incorrect.

**Creating Tables**

In SQL we use the following syntax in creating a table:

***CREATE TABLE table\_name***

NB: The syntax is not case sensitive, but it is more professional to use the upper case while writing the syntax.

A purple dotted line with a white background

AI-generated content may be incorrect.

**Example:**

*CREATE TABLE Course,*

*(CourseCode VARCHAR (8) NOT NULL,*

*CourseDesc VARCHAR (20) NOT NULL,*

*CONSTANT Course\_CourseCode\_pk PRIMARY KEY (CourseCode));*

The attribute CourseCode is the primary key. Both CourseCodeand CourseDesc are not allowing Null values. A Null value is not the same as a zero numeric value or a string filled with spaces. Zero and spaces are values. Null represents the absence of a value. They are used to deal with incomplete or exceptional data.

**Types of Data types in SQL**

There are various data types you can use. The most common are:

* CHAR / VARCHAR (): This is used for string text
* NUMBER: This is used for numeric data
* DATE: This is used for dates
* BIT: This is used for Boolean
* CURRENCY: This is used for currency. (Add a £ front for display purposes).

The diagram below shows and explains the CourseCodeused as a primary key.

A diagram of course

AI-generated content may be incorrect.

**Creating a table using two constraints**

*CREATE TABLE Student*

*(StudentNo VARCHAR (10) NOT NULL,*

*Name VARCHAR (20) NOT NULL,*

*CourseCode VARCHAR (8) NOT NULL,*

*CONSTRAINT Student\_StudentNo\_pk PRIMARY KEY (StudentNo),*

*CONSTRAINT Student\_CourseCode\_fk FOREIGN KEY (CourseCode) REFERENCES Course(CourseCode));*

One to define the Primary Key

Another one to define the Foreign Key that links it to Course.

NB: When creating tables in SQL the foreign key is ALWAYS placed in the member table (the many ends of the relationship), and you only need one Foreign Key for each relationship.

A close-up of a computer code

AI-generated content may be incorrect.

**Relationships in an Entity-Relationship Diagram (ERD)**

In an **Entity-Relationship Diagram (ERD)**, a **relationship** represents the **association between two or more entities**. It shows how data in different tables (entities) are related to each other in a database.

**Types of Relationships in ERD:**

* One-to-one (1:1): Each entity in **A** is related to only one entity in **B**, and vice versa.
* One-to-Many (1:M): One entity in **A** can be associated with **many** entities in **B**, but each entity in **B** is linked to only **one** entity in **A**
* Many-to-Many (M: N): Many entities in **A** can relate to **many** entities in **B**.

**A diagram of a computer

AI-generated content may be incorrect.Examples:A diagram of a lecture

AI-generated content may be incorrect.**

**Inserting data into a table**

To insert data into tables, we can use the SQL **INSERT** statement. It is possible to write the **INSERT INTO** statement in two ways:

1. Specify both the column names and the values to be inserted:

**INSERT INTO table\_name (column1, column2, column3,..)VALUES (value1, value2, value3, ...);**

1. If you are adding values for all the columns of the table, you do not need to specify the column names in the SQL query. However, make sure the order of the values is in the same order as the columns in the table. Here, the **INSERT INTO** syntax would be as follows:

**INSERT INTO table\_name VALUES (value1, value2, value3, ...);**

**Rules in inserting data**

* You must enter a value into a required or NOT NULL attribute (field).
* Once you have set up relationships, your data must conform to the requirements of referential integrity.