SQL – STRUCTURED QUERY LANGUAGE

- KEY LINKS:
    a. SQL functions - here
    b. SQL datatypes - here
- This refers to a Structured Query Language
- SQL is an RDBMS
- There are multiple types of databases:
    a. Hierarchical
    b. Network
    c. Object-Oriented
    d. Relational
    e. NoSQL
- The building block of SQL is **database tables** - a collection of related data entries, consisting of columns and rows.
- **Record (row)** - individual entry in a table
- **Column** - Holds specific information about every record in a table
- SQL is used to perform **CRUD** operations
    a. Create - Insert
    b. Read - Search
    c. Update - Update
    d. Delete - Delete

- ## SQL COMMANDS
    a. **`SELECT` - extracts data from a database**
        - General syntax:

          ```
          SELECT <column-name> or *
          FROM <table-name>
          ```

        - To get **distinct** results (no duplicates):

          ```
          SELECT DISTINCT <column-name> …
          ```

        - You can limit the number of row results that you want

          ```
          SELECT column_name(s)
          FROM table_name
          WHERE condition
          LIMIT number;
          ```

    b. **`UPDATE` - updates data in a database**

      ```
      UPDATE table_name
      SET column1 = value1, column2 = value2, ...
      WHERE condition;
      ```

c. **`DELETE` - deletes data from a database**

- To delete a specific row/record in the database
  ```
  DELETE FROM table_name WHERE condition;
  ```

- To delete all the records of a table but then leave the table structure intact:
  ```
  DELETE FROM table_name;
  ```

d. **`INSERT INTO` - inserts new data into a database**

- Inserting into specific columns:
  - NB: The order of values must match

  ```
  INSERT INTO table_name (column1, …)
  VALUES (value1, …);
  ```

- Inserting into all the table columns

  ```
  INSERT INTO table_name
  VALUES (value1, …);
  ```

e. **`CREATE DATABASE` - creates a new database**
e. **`ALTER DATABASE` - modifies a database**
e. **`CREATE TABLE` - creates a new table**
e. **`ALTER TABLE` - modifies a table**
e. **`DROP TABLE` - deletes a table**
e. **`CREATE INDEX` - creates an index (search key)**
e. **`DROP INDEX` - deletes an index**

- **<u>SQL JOINS</u>**

a. A **join** is where we combine rows from two or more tables, based on a related column between them.
b. Syntax:

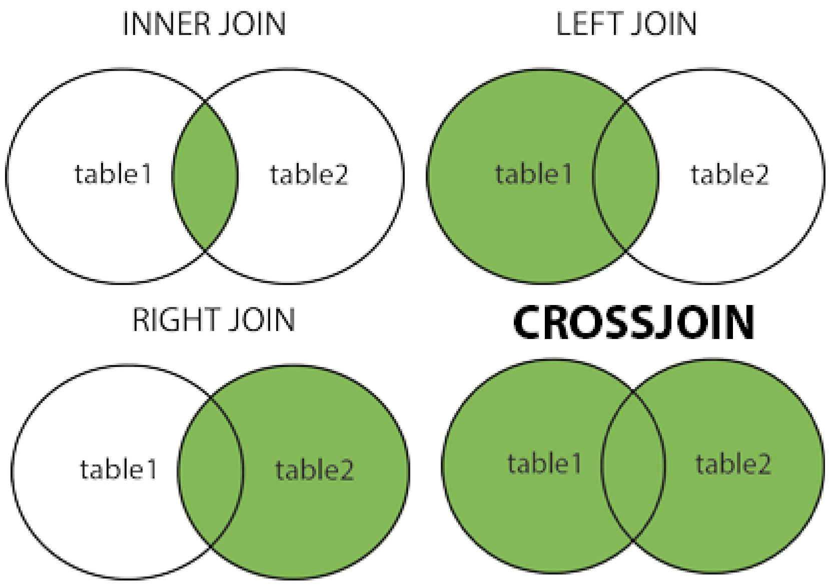
```
SELECT <column-name>, …
FROM <table-name-1>
INNER JOIN <table-name-2> ON
<table-1-unique column> = <table-2-unique column>;

EG:

SELECT Orders.OrderID, Customers.CustomerName,
Orders.OrderDate
FROM Orders
INNER JOIN Customers ON
Orders.CustomerID=Customers.CustomerID;
```

c. There are multiple types of table joins:

- `INNER JOIN`: Returns records that have matching values in both tables
- `LEFT JOIN`: Returns all records from the left table, and the matched records from the right table
- `RIGHT JOIN`: Returns all records from the right table, and the matched records from the left table
- `CROSS JOIN`: Returns all records from both tables
- `SELF JOIN`: The table is just joined with itself



- **<u>ORDER OF SQL QUERIES</u>**

  - `SELECT` *column data and functions (count, sum, avg)*
  - `FROM` *table data*
  - `JOIN` *to join tables by their unique columns*
  - `WHERE` conditionals
  - `GROUP BY` *categorises data*
  - `HAVING` *conditions on grouped data*

- **<u>SQL CONSTRAINTS</u>**
  - `NOT NULL` - Ensures that a column cannot have a NULL value
  - `UNIQUE` - Ensures that all values in a column are different
  - `PRIMARY KEY` - A combination of a `NOT NULL` and `UNIQUE`. Uniquely identifies each row in a table
  - `FOREIGN KEY` - Prevents actions that would destroy links between tables
  - `CHECK` - Ensures that the values in a column satisfy a specific condition

- `DEFAULT` - Sets a default value for a column if no value is specified
- `CREATE INDEX` - Used to create and retrieve data from the database very quickly

  SQL Creating a table with constraints:

```
CREATE TABLE Orders (
     OrderID int NOT NULL,
     OrderNumber int NOT NULL,
     PersonID int,
    PRIMARY KEY (OrderID),
    CONSTRAINT FK_PersonOrder FOREIGN KEY (PersonID)
    REFERENCES Persons(PersonID)
);
```

- **SQL Date Data Types**

  - These are the data types for storing a date or a date/time value
    - `DATE` - format YYYY-MM-DD
    - `DATETIME` - format: YYYY-MM-DD HH:MI:SS
    - `TIMESTAMP` - format: YYYY-MM-DD HH:MI:SS
    - `YEAR` - format YYYY or YY