Introduction and Revision

Why SQL is important?

- **Standardised Language:** universally accepted for managing and manipulating relational databases.
- **Data Retrieval:** efficiently retrieves data using SELECT statements and complex queries.
- **Data Manipulation:** supports INSERT, UPDATE, DELETE operations to modify data.
- **Data Definition:** allows creation and modification of database structures with CREATE, ALTER, DROP statements.
- **Data Control:** manages access permissions and controls data security using GRANT and REVOKE statements.
- **Transactional Control:** ensures data integrity and consistency with transaction commands like COMMIT, ROLLBACK.
- **Performance Optimisation:** enhances query performance through indexing and execution plans.
- **Widely Supported:** supported by major RDBMS platforms such as MySQL, PostgreSQL, SQL Server and Oracle.
- **Scalability:** handles large volumes of data and supports complex transactions in enterprise environments.
- **Interoperability:** facilitates integration with various applications and programming languages.
- **Data Integrity:** enforces data integrity through constraints like PRIMARY KEY, FOREIGN KEY, UNIQUE and CHECK.
- **Analytical Capabilities:** provides powerful analytical functions such as aggregation, grouping and window functions for data analysis.

- **Foundation of Advanced Databases:** fundamental knowledge required for understanding and working with advanced database technologies and NoSQL systems.

SQL Statements

> Basic SELECT Statements

- **SELECT:**
  - Begins the query and specifies the columns to retrieve.

- **FROM:**
  - Specifies the table from which to retrieve the data.

- **WHERE:** (optional)
  - Filters the results based on a condition.

- **ORDER BY:** (optional)
  - Sorts the results based on one or more columns. (asc / desc)

- **GROUP BY:** (optional)
  - Groups rows that have the same value in specified columns into summary rows.
  - Often used with aggregate functions (COUNT, SUM, AVG, ...).

- **HAVING:** (optional)
  - Filters record that work on summarized GROUP BY results.

    - Used with GROUP BY to apply conditions to the groups (similar to WHERE).

- **LIMIT:** (optional)
    - Limits the number of rows returned

```
select column1, column2, ...
    from table_name
    where condition
    order by column1 [asc/desc]
    group by column1
    having condition
    limit number

// * is used as a wildcard for all -> SELECT * FROM TABLE
```

> JOINS

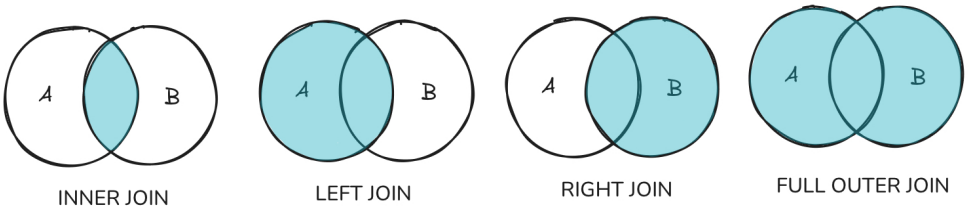
```
select * from table1
    join table2 on table1.id = table2.id
```

- **INNER JOIN:** retrieves record that have matching values in both tables (intersection).

- **LEFT JOIN:** retrieves all records from the left table and matched records from the right table. Records with no match in the right table will contain NULL.

- **RIGHT JOIN:** retrieves all records from the right table and matched records from the left table. Records with no match in the left table will contain NULL.

- **FULL OUTER JOIN:** retrieves records that appear when there is a match in either left or right table. Records with no match in either table will contain NULL.



> Aggregates

```
select
    count(column_name) as cnt     // AS used for alias names
from table_name
```

- **COUNT:** counts the number of rows in a result set.
- **SUM:** adds up the values in a specified column.
- **AVG:** calculates the average value of a specified column.
- **MIN:** retrieves the minimum value from a specified column.
- **MAX:** retrieves the maximum value from a specified column.

> Subqueries

- **Nested Queries:** a query within another query.
- **Correlated Subqueries:** a subquery that uses values from the outer query.
- **IN Clause:** used to filter the main query results based on the subquery results.

> WINDOW Functions

- **ROW_NUMBER:** assigns a unique sequential integer to rows within a partition of a result set.
- **RANK:** assigns a rank to each row within a partition, with gaps for ties.
- **DENSE_RANK:** same to gap but without gaps.
- **OVER Clause:** defines a window or user-specified set of rows within a query result.

> Common Table Expressions (CTEs)

- Temporary result set can be referenced within a SELECT, INSERT, UPDATE or DELETE statement.
- Defined using the WITH clause.

> UNION and UNION ALL

- **UNION:** combines the results of two or more SELECT statements and removes duplicates.

- **UNION ALL:** combines the results of two or more SELECT statements without removing duplicates.

Key concepts from the lecture:

- **ERD:** Entity Relationship Diagram. Tool for the data model which facilitates the representation of entities from a database.
  - **Conceptual ERD:** establishes the entities, their attributes and their relationships.
  - **Logical ERD:** defines the structure of the data elements and sets of relationships between them.
  - **Physical ERD:** represents the actual design of a relational database, describes the database-specific implementation of the data model.

Last year's DB notes:

notes_ERRM.pdf