SQL BOOTCAMP: COMBINE, RENAME & OPTIMIZE

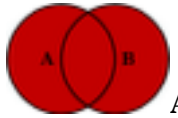
JOINS

Joins combine records from two or more tables in a database where a specific condition is met. The general syntax for a join is

```
SELECT columns FROM A
join_type B
ON condition;
```

Where columns is the list of columns to return, condition is the condition to match and join_type is one of **INNER JOIN**, **OUTER JOIN**, **LEFT JOIN** or **RIGHT JOIN**

OUTER JOIN

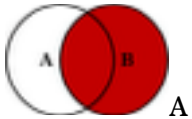


An outer join returns all rows where either table meets the condition, and fills in the remainder with **NULL**

INNER JOIN

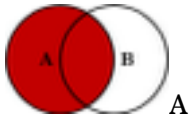
An inner join returns all rows where both tables match on the condition.

RIGHT JOIN



A right join returns all rows where B matches the condition.

LEFT JOIN



A left join returns all rows where A matches the condition.

UNION

UNION operator combines the result of two or more **SELECT** statements into one single table.

```
[query 1] UNION [query 2];
```

Note that each **SELECT** statement must have the same number of columns. The columns must also have similar data types and must be in the same order.

SUBQUERIES

A subquery is a query that is nested inside a SQL statement, or inside another subquery.

- a correlated subquery is a subquery that uses values from the outer query.
- uncorrelated subqueries have no relationship with the outer query.

ALIASING

SQL aliases are used to give a database table, or a column in a table, a temporary name. Their main use is to make tables more readable and useful. You can alias column names

```
SELECT count(*) AS count_rows
FROM table_name;
```

And also table names

```
SELECT column_list
FROM table_name AS table_alias;
```

Aliasing a table like this is useful for making long **JOIN**s and subqueries more readable

SELF JOINS

A self-join is a query in which a table is joined (compared) to itself. Self-joins are used to compare values in a column with other values in the same column in the same table. When you do a self join you need to alias at least one of the tables, like this

```
SELECT a.col, b.col
FROM table1 AS a, table1 AS b
WHERE a.common = b.common;
```

INDEXES

Indexes allow the database application to find data fast; without reading the whole table. The **CREATE INDEX** statement is used to create indexes in tables.