

- Pay attention to the **SELECT** clause and only include the specified columns in the result.
- While SQL keywords are not case sensitive, table names and column names are.
- Be careful not to mix up **JOIN** and **LEFT JOIN** — **JOIN** includes only those rows of the left table that have matches in the right table, while **LEFT JOIN** includes all rows of the left table, either paired with the matches in the right table or with a **NULL** row if there aren't any matches.
- Joins produces one result row for every combination of a left-table row and a matching right-table row. If a left-table row matches three rows on the right, that left-table row's data gets repeated three times in the result, once per match.
- Which table is on the left matters for **LEFT JOIN** — **R LEFT JOIN S ON ...** does not necessarily produce the same result as **S LEFT JOIN R ON ...**. Every row in the left table appears in the result; only those rows from the right table that have a match in the left table are included.
- **JOIN ... ON** does *not* remove the duplicate column — the result has all the columns of both tables. Only **JOIN ... USING** and **NATURAL JOIN** remove the duplicate columns. This is to be consistent within a particular form of **JOIN** — with **JOIN ... ON**, the join condition doesn't have to be an equality condition and doesn't have to involve columns of the same name so it would be confusing to remove duplicate columns only with certain join conditions. With **USING** and **NATURAL JOIN** the join condition is always an equality condition on columns of the same name, so there is always a duplicated column and thus it is reasonable to omit the redundant copy.
- While it is legal in MySQL to use **JOIN** without **ON** or **USING** (the behavior defaults to **CROSS JOIN**, it is better practice to explicitly use **CROSS JOIN** instead — it's clearer (you don't have to wonder if **ON** was omitted accidentally or on purpose) and is required by the SQL standard.
- The SQL should reflect the English — if a query asks for Geneva Public Library by name, don't look up the branch ID associated with Geneva Public Library in the database and hardcode that ID into the query. Such a query won't work correctly for a different data set.
- Be careful with syntax details e.g. parens around the column name in a **USING** clause and single quotes around string values.