Structured Query Language (SQL)

What is SQL?

- It is a standard programming language for accessing a relational database. It has been designed for managing data in Relational Database Management Systems (RDBMS) like Oracle, MySQL, MS SQL Server, IBM DB2.
- SQL has been widely used to retrieve data, merge data, perform group and nested case queries for decades. Even for data science, SQL has been widely adopted.

Properties of Databases

- A database transaction, however, must be ACID compliant. ACID stands for Atomic, Consistent, Isolated and Durable as explained below:
- Atomic : A transaction must be either completed with all its data modifications or may not.
- Consistent : At the end of the transaction, all data must be left consistent.
- Isolated : Data modifications performed by a transaction must be independent of other transactions.
- Durable : At the end of transaction, effects of modifications performed by the transaction must be permanent in system.

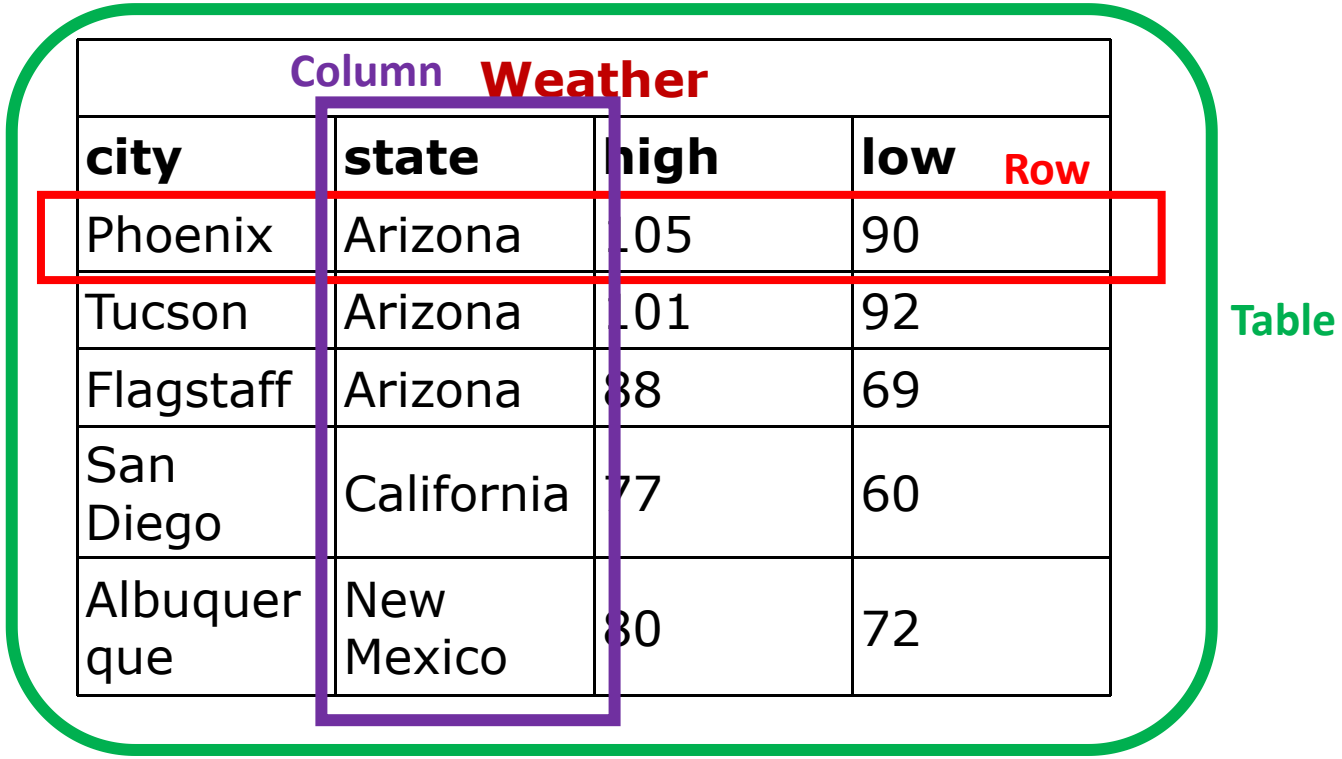
Tools

https://sqlzoo.net/
https://www.w3schools.com/sql/default.asp
http://www.sqlcourse2.com/select2.html
https://leetcode.com/problemset/database/
http://sqlfiddle.com/

Table Basics

- A relational database system contains one or more objects called tables.
- The data or information for the database are stored in these tables. Tables are uniquely identified by their names and are comprised of columns and rows.
- Columns contain the column name, data type, and any other attributes for the column. Rows contain the records or data for the columns.
- Here is a sample table called "weather".
- city, state, high, and low are the columns. The rows contain the data for this table:

Weather			
city	**state**	**high**	**low**
Phoenix	Arizona	105	90
Tucson	Arizona	101	92
Flagstaff	Arizona	88	69
San Diego	California	77	60
Albuquerque	New Mexico	80	72


Column
Weather
city
state
high
low
Row
Phoenix
Arizona
105
90
Tucson
Arizona
101
92
Flagstaff
Arizona
88
69
San Diego
California
77
60
Albuquerque
New Mexico
80
72
Table

CREATE Table

- The CREATE TABLE statement is used to create a new table in a database.
  - **CREATE TABLE *table_name* (**
    column1 datatype,
    column2 datatype,
    column3 datatype,
    ....
    );
  - https://sqlzoo.net/wiki/CREATE_TABLE

INSERT values

- The INSERT INTO statement is used to insert new records in a table.
  - **INSERT INTO *table_name* (*column1*, *column2*, *column3*, ...)**
    VALUES (*value1*, *value2*, *value3*, ...);
  - https://sqlzoo.net/wiki/INSERT_.._VALUES

UPDATE Table

- The UPDATE statement can be used to change a values in rows that already exists.
  - **UPDATE *table_name***
    SET *column1* = *value1*, *column2* = *value2*, ...
    WHERE *condition*;
  - https://sqlzoo.net/wiki/UPDATE

ALTER TABLE

- The ALTER TABLE statement is used to add, delete, or modify columns in an existing table.
- The ALTER TABLE statement is also used to add and drop various constraints on an existing table.
  - **ALTER TABLE *table_name***
    ADD *column_name datatype*;
  - https://sqlzoo.net/wiki/ALTER

DELETE

- The DELETE statement is used to delete existing records in a table.
  - **DELETE FROM *table_name* WHERE *condition*;**
  - https://sqlzoo.net/wiki/DELETE

DROP Table

- The DROP TABLE statement is used to drop an existing table in a database.
  - **DROP TABLE *table_name*;**
  - https://sqlzoo.net/wiki/DROP

SELECT Basics

- The select statement is used to query the database and retrieve selected data that match the criteria that you specify.
- Here is the format of a simple select statement:
  - **select "column1" [,"column2",etc] from "tablename" [where "condition"]; [] = optional**
- The column names that follow the **select** keyword determine which columns will be returned in the results. You can select as many column names that you'd like, or you can use a **"*"** to select all columns.
- The table name that follows the keyword **from** specifies the table that will be queried to retrieve the desired results.
- The **where** clause (optional) specifies which data values or rows will be returned or displayed, based on the criteria described after the keyword where.
- Conditional selections used in the where clause:

=	Equal
>	Greater than
<	Less than
>=	Greater than or equal
<=	Less than or equal
<>	Not equal to
LIKE	Similar values
IN	Set of values
BETWEEN AND	Range

Practice

https://sqlzoo.net/wiki/SELECT_basics

https://sqlzoo.net/wiki/SELECT_names

https://sqlzoo.net/wiki/SELECT_from_WORLD_Tutorial

https://sqlzoo.net/wiki/SELECT_from_Nobel_Tutorial

Quiz

https://sqlzoo.net/wiki/SELECT_Quiz

https://sqlzoo.net/wiki/BBC_QUIZ

https://sqlzoo.net/wiki/Nobel_Quiz

https://leetcode.com/problems/big-countries/

Inner Queries

- we can use SELECT statements within SELECT statements to perform more complex queries.
- https://sqlzoo.net/wiki/SELECT_.._SELECT

Aggregate Functions

- Aggregate functions are used to compute against a "returned column of numeric data" from your SELECT statement.
- They basically summarize the results of a particular column of selected data.
- These functions can be used without the "GROUP BY" clause.
- https://sqlzoo.net/wiki/COUNT
- https://sqlzoo.net/wiki/AVG
- https://sqlzoo.net/wiki/SUM
- https://sqlzoo.net/wiki/MAX
- https://sqlzoo.net/wiki/SUM_and_COUNT

MIN	returns the smallest value in a given column
MAX	returns the largest value in a given column
SUM	returns the sum of the numeric values in a given column
AVG	returns the average value of a given column
COUNT	returns the total number of values in a given column
COUNT(*)	returns the number of rows in a table

GROUP BY clause

- The GROUP BY clause will gather all of the rows together that contain data in the specified column(s) and will allow aggregate functions to be performed on the one or more columns.
  - **SELECT column1, SUM(column2) FROM "list-of-tables" GROUP BY "column-list";**
  - https://sqlzoo.net/wiki/SELECT_.._GROUP_BY

HAVING clause

- The HAVING clause allows you to specify conditions on the rows for each group - in other words, which rows should be selected will be based on the conditions you specify.
- The HAVING clause should follow the GROUP BY clause if you are going to use it.
  - **SELECT column1, SUM(column2) FROM "list-of-tables" GROUP BY "column-list" HAVING "condition";**
  - https://sqlzoo.net/wiki/SELECT_.._GROUP_BY

ORDER BY clause

- ORDER BY is an optional clause which will allow you to display the results of your query in a sorted order (either ascending order or descending order) based on the columns that you specify to order by.
  - **SELECT column1, SUM(column2) FROM "list-of-tables" ORDER BY "column-list" [ASC | DESC];**
    - [ ] = optional
  - https://sqlzoo.net/wiki/SELECT_.._GROUP_BY
  - https://sqlzoo.net/wiki/Using_SUM,_Count,_MAX,_DISTINCT_and_ORDER_BY

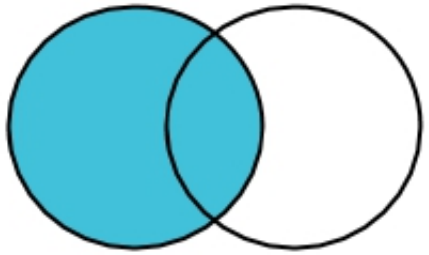
Quiz

https://sqlzoo.net/wiki/SELECT_within_SELECT_Tutorial

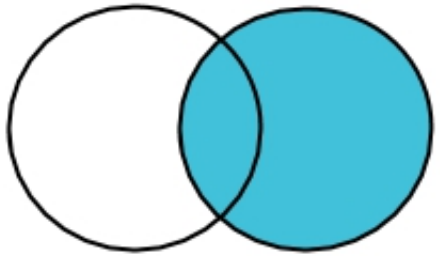
https://sqlzoo.net/wiki/SUM_and_COUNT_Quiz

SQL JOINS

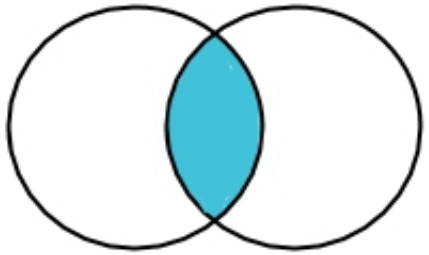
- A JOIN clause is used to combine rows from two or more tables, based on a related column between them.
- Joins allow you to link data from two or more tables together into a single query result



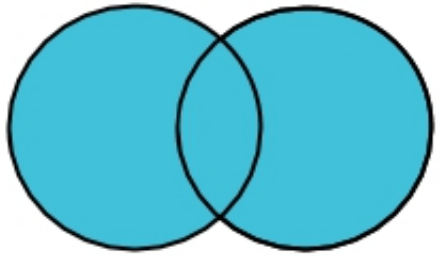
Left Join



Right Join



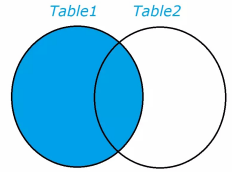
Inner Join



Full Outer Join

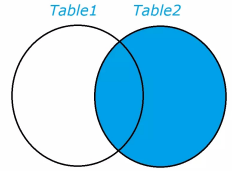
SQL Joins | Cheat Sheet

LEFT JOIN:



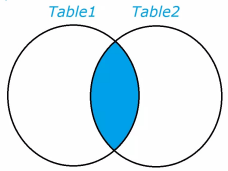
SELECT *column_name(s)* FROM *Table1* **LEFT JOIN** *Table2*
ON *Table1.column_name=Table2.column_name*

RIGHT JOIN:



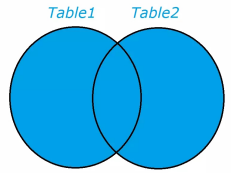
SELECT *column_name(s)* FROM *Table1* **RIGHT JOIN** *Table2*
ON *Table1.column_name=Table2.column_name*

INNER JOIN:



SELECT *column_name(s)* FROM *Table1* **INNER JOIN** *Table2*
ON *Table1.column_name=Table2.column_name*

FULL JOIN:



SELECT *column_name(s)* FROM *Table1* **FULL JOIN** *Table2*
ON *Table1.column_name=Table2.column_name*

Primary and Foreign Keys

- In each table there exist a column that is the **primary key** which is a column where each entry uniquely represents a single row in that table.
- This is usually the ID (short for identifier) column.
- A column in a table that establishes an association with another table's primary key via shared values is called a **foreign key**.
- Foreign keys are also typically titled IDs but prepended with the name of the referenced table.

User Table – Table 1

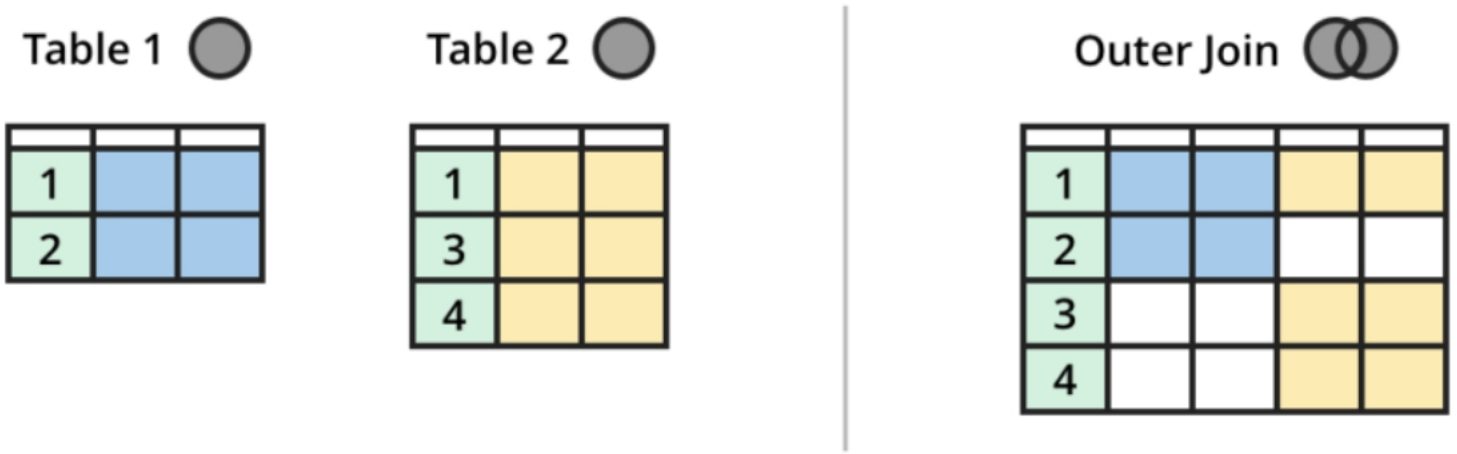
ID (Primary Key)	Name	Address
1	Sally Select	123 Join Dr
2	Frank From	25 Where St

Event Table – Table 2

User_ID (Foreign Key)	ID (Primary Key)	Action
1	A	LOGIN
3	B	VIEW PAGE
4	C	LOGIN

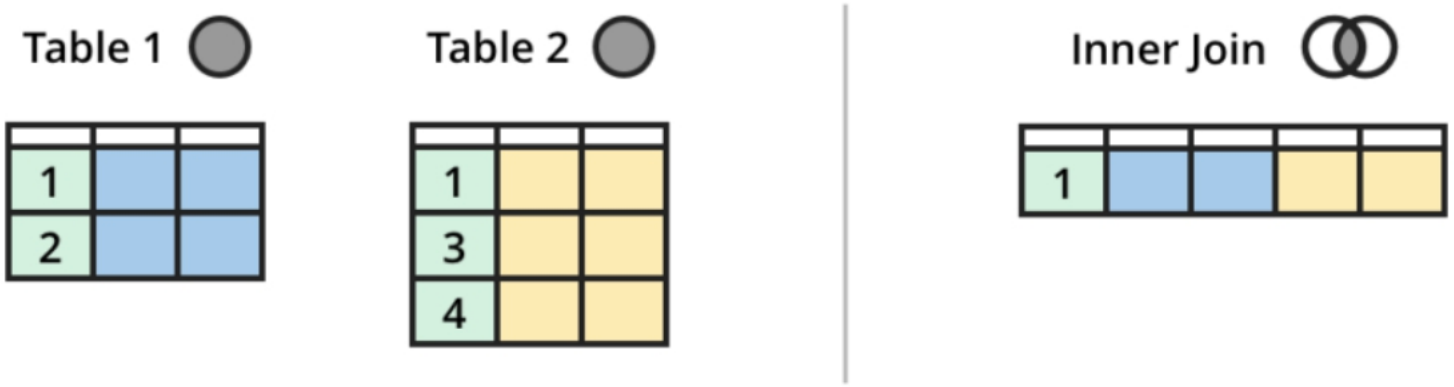
Outer Join

- Let’s say you want to have a table that contains all your user and event table data together.
- You would use an **Outer Join** to join the tables together.
- An outer join combines the columns from all tables on one or more common dimension when possible, and includes all data from all tables.



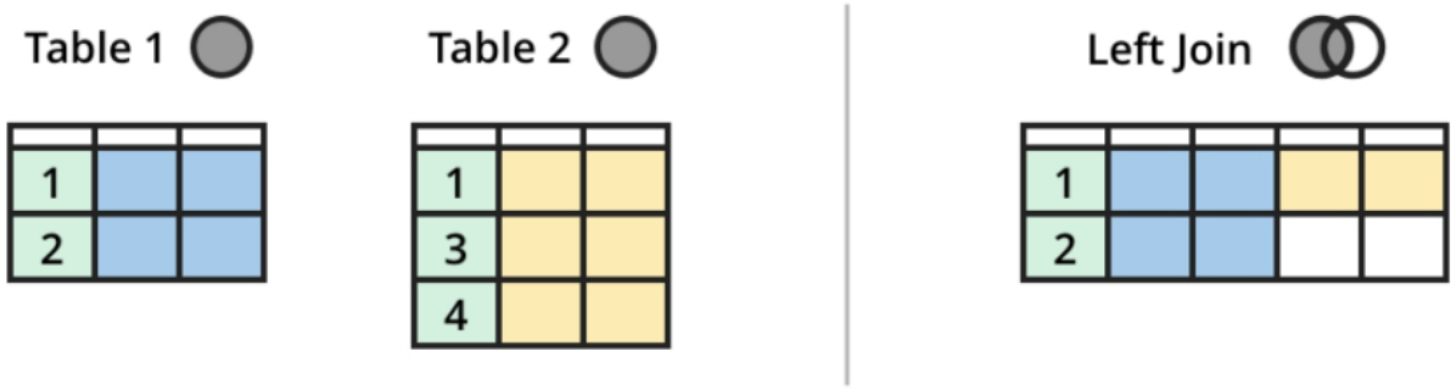
Inner Join

- What if you want to have a table that contains only users that have done an action?
- You would use an **Inner Join** to join the tables together.
- An inner join combines the columns on a common dimension (the first N columns) when possible, and only includes data for the columns that share the same values in the common N column(s).
- In the example, the User ID would be the common dimension used for the inner join.



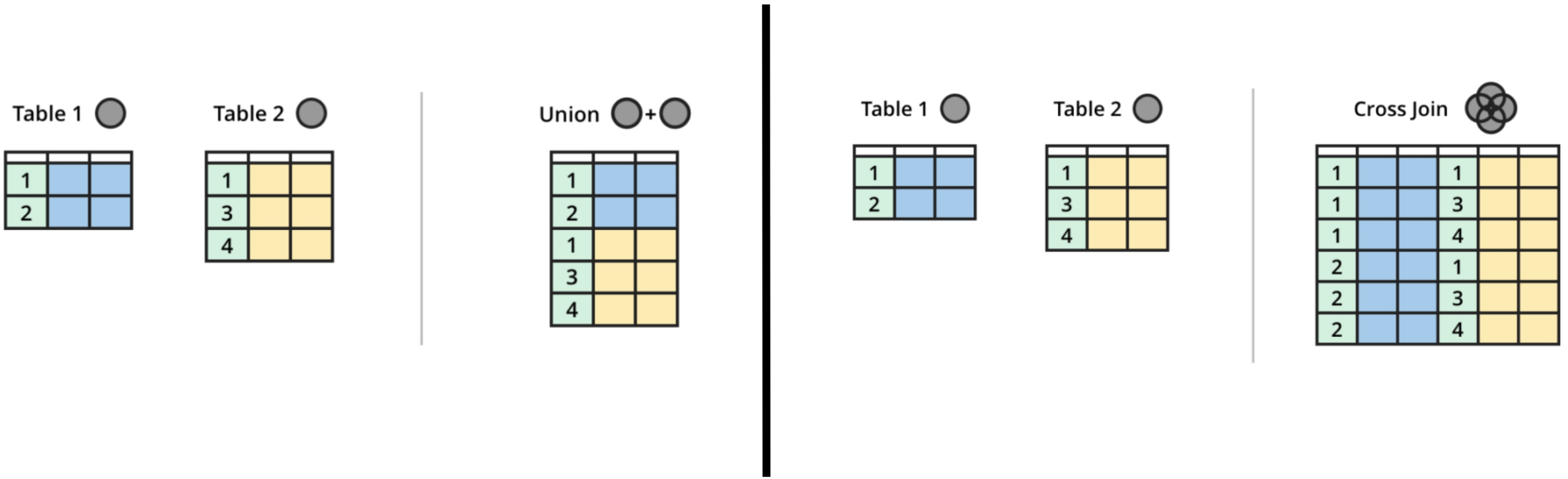
Left Join

- Now, what if you want to have a table that contains all the users' data and only actions that those users have done?
- Actions performed by other users not in the users table should not be included?
- You would use a **Left Join** to join the tables together.
- A left join combines the columns on a common dimension (the first N columns) when possible, returning all rows from the first table with the matching rows in the consecutive tables.
- The result is NULL in the consecutive tables when there is no match.
- In this case, we would make the User Table the first (left table) to use for the left join.



Union and Cross Join

- In addition to these common join types, there are some methods which will result in additional rows in your output table as well as more columns.
- Two of these join types are called **Union** and **Cross Join**.
- A **Union Join** will stack tables on top of each other resulting in new rows.
- A good use case for this would be if you're looking to combine two tables by appending them rather than joining them.
- A **Cross Join** would result in a table with all possible combinations of your tables' rows together.
- This can result in enormous tables and should be used with caution.
- Cross Joins will likely only be used when your tables contain single values that you want to join together without a common dimension.



Combining Data Tables – SQL Joins Explained

A JOIN clause in SQL is used to combine rows from two or more tables, based on a related column between them.

Table 1

1		
2		

Table 2

1		
3		
4		

Outer Join

1				
2				
3				
4				

Inner Join

1				

Left Join

1				
2				

Union

1		
2		
1		
3		
4		

Cross Join

1			1		
1			3		
1			4		
2			1		
2			3		
2			4		

Left outer

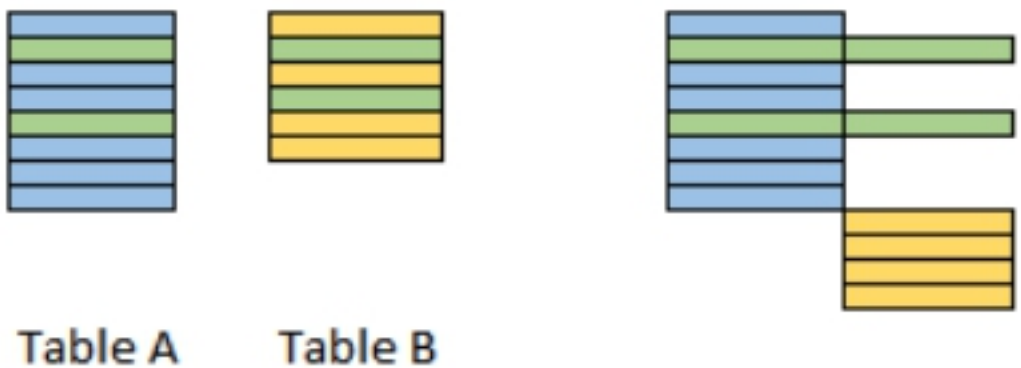
All from A and Matching from B

Right Outer

All from B and matching from A

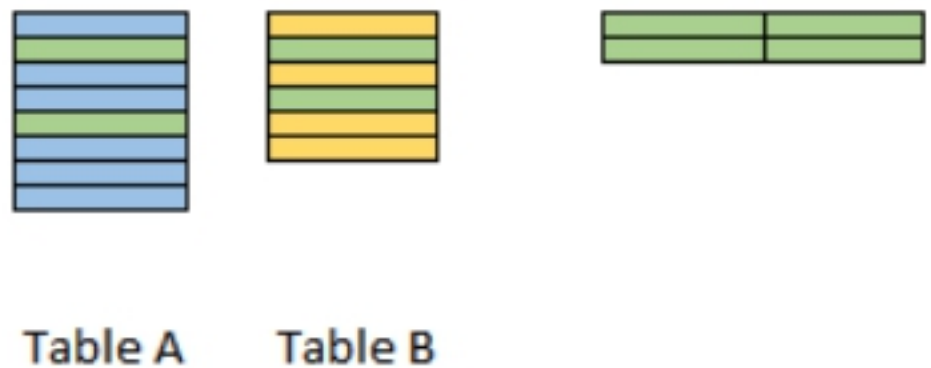
Full Outer

All from A and B



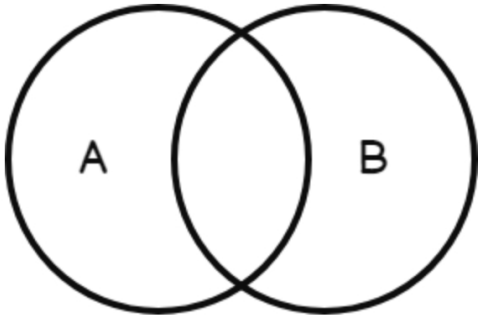
Inner

Only Matching rows

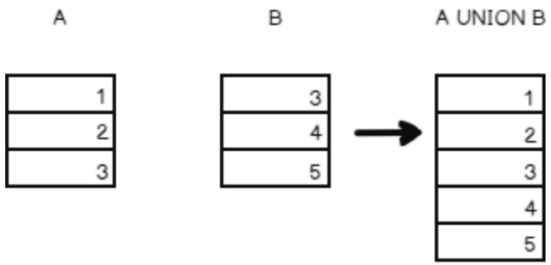


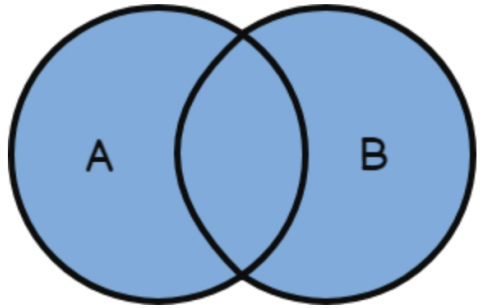
Union and Union All

- The **Union** operator combines the results of two or more queries into a distinct single result set that includes all the rows that belong to all queries in the Union.
- In this operation, it combines two more queries and removes the duplicates.
- When looking at Union vs Union, All we find they are quite similar, but they have some important differences from a performance results perspective.
- The **Union All** operator combines the results of two or more queries into a single result set that includes all the rows that belong to all queries in the Union.
- In simple terms, it combines the two or more row sets and keeps duplicates.

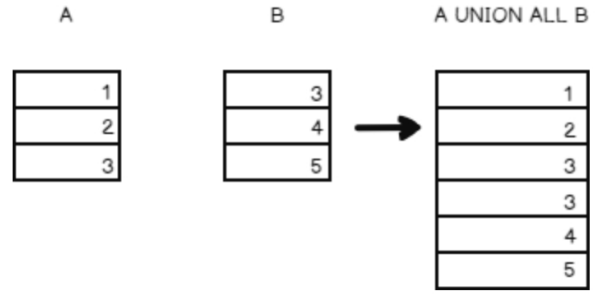


For example, the table 'A' has 1,2, and 3 and the table 'B' has 3,4,5.





For example, the table 'A' has 1,2, and 3 and the table 'B' has 3,4,5.



Practice

https://www.w3schools.com/sql/sql_join.asp

https://www.w3schools.com/sql/sql_union.asp

https://sqlzoo.net/wiki/The_JOIN_operation

Quiz

https://sqlzoo.net/wiki/JOIN_Quiz