SQL CHEATSHEET

@yosracodes

Create

used to create a new database or table

```
CREATE DATABASE <DATABASE NAME>
CREATE TABLE <TABLE NAME>
```

Drop

used to delete an existing database or table

```
DROP DATABASE <DATABASE NAME>
DROP TABLE <TABLE NAME>
```

Truncate

used to delete information in the table but doesn't delete the table itself

```
TRUNCATE TABLE <TABLE NAME>
```

Alter

used to delete, add or modify constraints or columns in a table

```
ALTER TABLE <TABLE NAME>
ADD <COLUMN NAME> <DATA TYPE>

ALTER TABLE <TABLE NAME>
DROP COLUMN <COLUMN NAME>

ALTER TABLE <TABLE NAME>
ALTER COLUMN <COLUMN NAME> <DATA TYPE>
```

Backup

used to create a backup on an existing database

```
BACKUP DATABASE <DATABASE NAME>
TO DISK = '<PATH>'
```

Insert

used to insert new tuples (rows) in a table

```
INSERT INTO <TABLE NAME> (<COLUMN1>, ....)
VALUES (<VALUE1>, ....)
```

*you do not need to specify all columns if you will add values for all the columns

Delete

used to delete tuples (rows) from a table

```
DELETE FROM <TABLE NAME>
WHERE <CONDITION>
```

*if you don't add the WHERE clause, all rows will be deleted

Update

used to modify existing records in a table

```
UPDATE <TABLE NAME>
SET <COLUMN NAME> = <NEW VALUE>
WHERE <CONDITION>
```

Select

used to select data from a table

```
SELECT <ATTRIBUTE LIST>
FROM <TABLE NAME>
WHERE <CONDITION>
```

*if you want all attributes of a table use (*)

Union, Intersect, Except

equivalent to the set operations: union, intersection and difference.

```
<FIRST SELECT STATEMENT>
UNION / INTERSECT / EXCEPT
<SECOND SELECT STATEMENT>
```

In

compares a value with a set of values, returns true if the value is one of the elements of the set.

```
SELECT <ATTRIBUTE LIST>
FROM <TABLE NAME>
WHERE <VALUE> IN <ANOTHER SELECT QUERY>
```

Null

used to check whether a value is NULL

```
<ATTRIBUTE NAME> IS (NOT) NULL
```

Join

used to join two tables based on a related column between them

```
SELECT <ATTRIBUTES LIST>
FROM <TABLE 1> JOIN <TABLE 2>
     ON <JOIN CONDITION>
WHERE <SELECTION CONDITION>
```

Assertion

used to ensure a certain condition is always met in the database

```
CREATE ASSERTION <ASSERTION NAME>
CHECK (<CONDITION>)
```

Trigger

Triggers are activated when a defined action is executed for the table

```
CREATE TRIGGER <TRIGGER NAME>
BEFORE / AFTER
INSERT / UPDATE / DELETE
ON <TABLE NAME>
FOR EACH ROW
<TRIGGER BODY>
```

Data Types

Numeric	-	INT, SMALLINT, DECIMAL(i, j)
String	-	CHAR, CHAR(n), VARCHAR(n)
Bit String	-	BIT, BIT(n)
Date and time	-	DATE, TIME, TIME(i)
Timestamp	-	TIMESTAMP

Referential Triggered Action

used to set what happens on updating or deleting a tuple (row) in the database that references another row

```
ON DELETE <OPTION>
ON UPDATE <OPTION>
```

OPTIONS:
SET NULL
SET DEFAULT
CASCADE

Renaming (Aliasing)

Relation and attribute names can be renamed for conenience or to remove ambiguity using the keyword **AS**

```
<TABLE NAME> AS <NEW TABLE NAME>
(<NEW ATTRIBUTE 1 NAME>, .....)
```

Cross Product (,)

used to produce a result table that has the number of rows of the first table multiplied by the number of rows of the second table

```
SELECT <ATTRIBUTE LIST>
FROM <TABLE 1>, <TABLE 2>
```

Duplicates

- **DISTINCT** is used to eliminate duplicates
- **ALL** is used to allow duplicates

```
SELECT ALL <ATTRIBUTE LIST>
FROM <TABLE NAME>

SELECT DISTINCT <ATTRIBUTE LIST>
FROM <TABLE NAME>
```

*SELECT without ALL or DISTINCT is equivalent to ALL

String Comparisons

- **LIKE** is used for string comparison
- (%) replaces an arbitary number of characters
- (_) replaces one character

```
<ATTRIBUTE> LIKE <PATTERN>
```

Arithmetic Operators

- (+) add
- (-) subtract
- (*) multiply
- (/) divide

Ordering

- **ORDER BY** is used to order the resulting tuples
- The keyword **ASC** (ascending) and **DESC** can be used.

```
<SELECT STATEMENT>
ORDER BY <ATTRIBUTE> <ASC / DESC>
```

*The default is ASC (ascending)

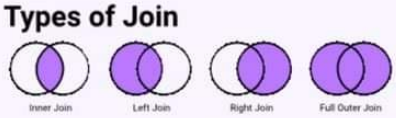
Set Comparisons

ANY and **ALL** can be used with (=, >, >=, <, <=, <>) to compare a value with a set

```
SELECT <ATTRIBUTE LIST>
FROM <TABLE NAME>
WHERE <VALUE> > ALL / ANY <ANOTHER SELECT QUERY>
```

- **CONTAINS** - Compares two sets and returns true if one set contains the other
- **EXISTS** - It checks whether the result of a nested query is empty or not
- **UNIQUE** - checks if the table has duplicates

Types of Join



Aggregate Functions

- **COUNT** - Counts how many rows in a particular column
- **SUM** - adds together all the values in a particular column
- **MIN** - returns the minumum value in a column
- **MAX** - returns the maximum value in a column
- **AVG** - returns the average of a group of selected values