COMMANDS

COMMAND	CODE	DESCRIPTION

Create	CREATE DATABASE <DATABASE NAME>
	CREATE TABLE <TABLE NAME>
		used to create a new database or table

Drop	DROP DATABASE <DATABASE NAME> DROP TABLE <TABLE NAME>	used to delete an existing database or table
Truncate	TRUNCATE TABLE <TABLE NAME>	used to delete information in the table but doesn t delete the table itself
Alter	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>	used to delete, add or modify constraints or columns in a table
Backup	BACKUP DATABASE <DATABASE NAME> TO DISK = ‘<PATH>’	used to create a backup on an existing database

Insert	INSERT INTO <TABLE NAME> (<COLUMN1>, ...)
	VALUES (<VALUE1>, ...)
		used to insert new tuples (rows) in a table
		**you do not need to specify all columns if you will add values for all the columns*

Delete	DELETE FROM <TABLE NAME>
	WHERE <CONDITION>
		used to delete tuples (rows) from a table
		**if you don t add the WHERE clause, all rows will be deleted*

Update	UPDATE <TABLE NAME>
	SET <COLUMN NAME> = <NEW VALUE>
	WHERE <CONDITION>
		used to modify existing records in a table

Select	SELECT <ATTRIBUTE LIST>
	FROM <TABLE NAME>
	WHERE <CONDITION>
		used to select data from a table
		**if you want all attributes of a table use (*)*

Union, Intersect, Except	<FIRST SELECT STATEMENT> UNION / INTERSECT / EXCEPT <SECOND SELECT STATEMENT>	equivalent to the set operations: union, intersection and difference.

In	SELECT <ATTRIBUTE LIST>
	FROM <TABLE NAME>
	WHERE <VALUE> IN <ANOTHER SELECT QUERY>
		compares a value with a set of values, returns true if the value is one of the elements of the set.

Null	<ATTRIBUTE NAME> IS (NOT) NULL	used to check whether a value is NULL

Join	SELECT <ATTRIBUTES LIST>
	FROM <TABLE 1> JOIN <TABLE 2>
	ON <JOIN CONDITION>
	WHERE <SELECTION CONDITION>
		used to join two tables based on a related column between them

COMMANDS (cont)

Assertion	CREATE ASSERTION <ASSERTION NAME> CHECK (<CONDITION>)	used to ensure a certain condition is always met in the database
Trigger	CREATE TRIGGER <TRIGGER NAME> BEFORE / AFTER INSERT / UPDATE / DELETE ON <TABLE NAME> FOR EACH ROW <TRIGGER BODY>	Triggers are activated when a defined action is executed for the table
Data Types	**Numeric** - INT, SMALLINT, DECIMAL(i, j) **String** - CHAR, CHAR(n), VARCHAR(n) **Bit Sring** - BIT, BIT(n) **Date and Time** - DATE, TIME, TIME(i) **Timestamp** -TIMESTAMP	
Referential Triggered Action	ON DELETE <OPTION> ON UPDATE <OPTION>	used to set what happens on updating or deleting a tuple (row) in the database that references another row **OPTIONS:** SET NULL SET DEFAULT CASCADE
Renaming (Aliasing)	<TABLE NAME> AS <NEW TABLE NAME> (<NEW ATTRIBUTE 1 NAME>, ...)	Relation and attribute names can be renamed for conenience or to remove ambiguity using the keyword AS

Cross Product (,)	SELECT <ATTRIBUTE LIST>
	FROM <TABLE 1>, <TABLE 2>
		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

Duplicates	SELECT ALL <ATTRIBUTE LIST>
	FROM <TABLE NAME> <ATTRIBUTE> LIKE <PATTERN> <ATTRIBUTE NAME> IS (NOT) NULL
	SELECT <ATTRIBUTES LIST>
	FROM <TABLE 1> JOIN <TABLE 2> ON <JOIN CONDITION>
	WHERE <SELECTION CONDITION> SELECT DISTINCT <ATTRIBUTE LIST> FROM <TABLE NAME>
		DISTINCT is used to eliminate duplicates
		ALL is used to allow duplicates _*SELECT without ALL or DISTINCT is equivalent to ALL_

String Comparisons	<ATTRIBUTE> LIKE <PATTERN>	**LIKE** is used for string comparisoJ (%) replaces an arbitary number of characters (_) replaces one character

COMMANDS (cont)

Arithmetic Operators	# (+) add # (*) multiply # ( ) subtract # (/) divide	
Ordering	<SELECT STATEMENT> ORDER BY <ATTRIBUTE> <ASC / DESC>	**ORDER BY** is used to order the resulting tuples The keyword **ASC** (ascending) and **DESC** can be used. _*The default is ASC (ascending)_

Set Comparisons	SELECT <ATTRIBUTE LIST>
	FROM <TABLE NAME>
	WHERE <VALUE> > ALL / ANY <ANOTHER SELECT QUERY>
		ANY and **ALL** can be used with (=, >, >=, <, <=, <>) to compare a value with a set
		#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

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
Types of Join		Inner join Left Join Right Join Full Outer Join

Credit: @yosracodes