SQL Quick Reference

The examples below use the following tables:

LECTURER (id, LName) and STUDENT (id, StName, StAdvisor) where StAdvisor is the foreign key for a one-to-many SUPERVISING relationship between the lecturers and the students.

SQL Statement	**Syntax**	**Example**

AND / OR	SELECT column_name(s)
	FROM table_name
	WHERE condition
	AND|OR condition
		SELECT Id, StName
		FROM Student
		WHERE ID>1 AND StAdvisor > 2;

ALTER TABLE	ALTER TABLE table_name
	ADD column_name datatype
	or
	ALTER TABLE table_name
	DROP COLUMN column_name
		ALTER TABLE Student
		ADD Age Number;
		
		ALTER TABLE Student
		DROP COLUMN Age;

AS (alias)	SELECT column_name AS column_alias
	FROM table_name
	or
	SELECT column_name
	FROM table_name AS table_alias
		SELECT StName AS StudentName
		FROM Student;
		
		SELECT StName
		FROM Student AS KSUStudent;

BETWEEN	SELECT column_name(s)
	FROM table_name
	WHERE column_name
	BETWEEN value1 AND value2
		SELECT Id, LName
		FROM Lecturer
		WHERE ID
		BETWEEN 2 AND 4;

CREATE DATABASE	CREATE DATABASE database_name	

CREATE TABLE	CREATE TABLE table_name
	(
	column_name1 data_type,
	column_name2 data_type,
	...
	)
		CREATE TABLE Room(
		RNO Number,
		RDescription Text);

CREATE INDEX	CREATE INDEX index_name ON table_name (column_name) or CREATE UNIQUE INDEX index_name ON table_name (column_name)	

CREATE VIEW	CREATE VIEW view_name AS
	SELECT column_name(s)
	FROM table_name
	WHERE condition	

DELETE	DELETE FROM table_name
	WHERE some_column=some_value
	or
	DELETE FROM table_name
	(**Note:** Deletes the entire table!!)
	DELETE * FROM table_name
	(**Note:** Deletes the entire table!!)
		DELETE FROM Room
		WHERE RNO =2;
		
		DELETE FROM Room
		WHERE RNO
		BETWEEN 2 AND 4;

DROP DATABASE	DROP DATABASE database_name	

DROP INDEX	DROP INDEX table_name.index_name (SQL Server) DROP INDEX index_name ON table_name (MS Access) DROP INDEX index_name (DB2/Oracle) ALTER TABLE table_name DROP INDEX index_name (MySQL)	
DROP TABLE	DROP TABLE table_name	DROP TABLE Room;

GROUP BY	SELECT column_name, aggregate_function(column_name)
	FROM table_name
	WHERE column_name operator value
	GROUP BY column_name
		SELECT StAdvisor, COUNT(ID)
		FROM Student
		GROUP BY StAdvisor;
		SELECT StAdvisor, Avg(Age)
		FROM Student
		GROUP BY StAdvisor;
		SELECT StAdvisor, Avg(Age)
		FROM Student
		WHERE StAdvisor > 2
		GROUP BY StAdvisor;

HAVING	SELECT column_name, aggregate_function(column_name)
	FROM table_name
	WHERE column_name operator value
	GROUP BY column_name
	HAVING aggregate_function(column_name) operator value
		SELECT StAdvisor, Avg(Age)
		FROM Student
		WHERE StAdvisor > 1
		GROUP BY StAdvisor
		HAVING Avg(Age)>10;

IN	SELECT column_name(s)
	FROM table_name
	WHERE column_name
	IN (value1,value2,..)	

INSERT INTO	INSERT INTO table_name
	VALUES (value1, value2, value3,....)
	or
	INSERT INTO table_name
	(column1, column2, column3,...)
	VALUES (value1, value2, value3,....)
		INSERT INTO Student
		VALUES (10, ‘Mohamed’, 3, 15);
		or
		INSERT INTO Student
		(ID,Age)
		VALUES (11, 15);

LIKE	SELECT column_name(s)
	FROM table_name
	WHERE column_name LIKE pattern
		SELECT Id, LName
		FROM Lecturer
		WHERE LName LIKE ‘L*’;
		SELECT Id, LName
		FROM Lecturer
		WHERE LName LIKE ‘L?’;

ORDER BY	SELECT column_name(s)
	FROM table_name
	ORDER BY column_name [ASC|DESC]
		SELECT Id, LName
		FROM Lecturer
		ORDER BY LName DESC ;

SELECT	SELECT column_name(s)
	FROM table_name
		SELECT LName
		FROM Lecturer

SELECT *	SELECT *
	FROM table_name
		SELECT *
		FROM Lecturer

SELECT DISTINCT	SELECT DISTINCT column_name(s)
	FROM table_name
		SELECT DISTINCT LName
		FROM Lecturer

UPDATE	UPDATE table_name
	SET column1=value, column2=value,...
	WHERE some_column=some_value
		UPDATE Student
		SET StName=’Ali’, StAdvisor=’2’
		WHERE ID=1;

WHERE	SELECT column_name(s)
	FROM table_name
	WHERE column_name operator value
		Most of the examples above!

INNER JOIN	SELECT column_name(s)
	FROM table_name1
	INNER JOIN table_name2
	ON
	table_name1.column_name=table_name2.column_name
		SELECT StName, LName
		FROM Student, Lecturer
		WHERE Lecturer.ID = StAdvisor;
		
		SELECT StName, LName
		FROM Lecturer INNER JOIN Student ON
		Lecturer.ID = Student.StAdvisor;

LEFT JOIN	SELECT column_name(s)
	FROM table_name1
	LEFT JOIN table_name2
	ON
	table_name1.column_name=table_name2.column_name	

RIGHT JOIN	SELECT column_name(s)
	FROM table_name1
	RIGHT JOIN table_name2
	ON
	table_name1.column_name=table_name2.column_name	

FULL JOIN	SELECT column_name(s)
	FROM table_name1
	FULL JOIN table_name2
	ON
	table_name1.column_name=table_name2.column_name	

SELECT INTO	SELECT *
	INTO new_table_name [IN externaldatabase]
	FROM old_table_name
	or
	SELECT column_name(s)
	INTO new_table_name [IN externaldatabase]
	FROM old_table_name	

SELECT TOP	SELECT TOP number|percent column_name(s)
	FROM table_name	

TRUNCATE TABLE	TRUNCATE TABLE table_name	

UNION	SELECT column_name(s) FROM table_name1
	UNION
	SELECT column_name(s) FROM table_name2	

UNION ALL	SELECT column_name(s) FROM table_name1
	UNION ALL
	SELECT column_name(s) FROM table_name2	

SQL aggregate functions are: MIN, MAX,SUM, AVG, COUNT

Group By Example 1: What is the average mark for each course?

```
SELECT CourseID, Avg(Marks)
FROM Marks
GROUP BY CourseID
```

Course ID	Marks
CC151	73.9
CC152	55.5
CC153	55.7
EC111	71.0
EC112	43.5

If there's a GROUP BY clause, only grouped attributes and aggregates may appear in the SELECT clause.

Group By Example 2: How many students are taking each course?

```
SELECT CourseID, Count (ID),
FROM Marks
GROUP BY CourseID
```

Course ID	Count (ID)
CC151	3
CC152	2
CC153	1
EC111	1
EC112	1

HAVING Clause: How many students are taking each course (only report courses with more than one student)?

```
SELECT Course ID, Count (ID),
FROM Marks
GROUP BY Course ID
HAVING Count(ID) > 1
```

Course ID	Count (ID)
CC151	3
CC152	2

Comments: WHERE clauses perform filtering *before* the grouping. HAVING clauses perform filtering *after* the grouping.