Structured Query Language

SELECT

SELECT [DISTINCT] * | LIST OF COLUMNS, FUNCTIONS, CONSTANTS
FROM LIST OF TABLES OR VIEWS
[**WHERE** CONDITION(S)]
[**ORDER BY** ORDERING COLUMN(S) [**ASC** | **DESC**]]
[**GROUP BY** GROUPING COLUMN(S)]
[**HAVING** CONDITION(S)]

Usage: Returns records from a database. The SELECT statement can be written to; return records where conditions have been met, return records that are grouped (GROUPED BY) or sorted (ORDER BY), return aggregate information about the records (Count, Min, Max, Sum, Avg), or return unique records (Add the 'Distinct' keyword to the statement).

Examples: (*reference Sample Table*)

- Select all records from the Products table (tblProducts)
  SELECT * **FROM** tblProducts
  Returns: All Records

- Select two fields and sort in descending order by the products name
  SELECT fldAmount, fldName **FROM** tblProducts
  ORDER BY fldName **DESC**
  Returns: All records in descending order

- Return only unique products
  SELECT DISTINCT fldName **FROM** tblProducts
  Returns:

fldName
Chair
Desk
Speakers
Computer Desk

- Return the total of items with a stock quantity greater than zero
  SELECT COUNT(*) FROM tblProducts **WHERE** fldQty > 0
  Returns: 4

- Return the total amount of each product
  SELECT fldName, **SUM**(fldAmount)
  FROM tblProducts **GROUP BY** fldName
  Returns:

fldName	*fldAmount*
Chair	56
Desk	102
Speakers	0
Computer Desk	1

- Return the total amount of each product having an amount greater than 100
  SELECT fldName, **SUM**(fldAmount)
  FROM tblProducts
  GROUP BY fldName **HAVING SUM**(fldAmount) > 100
  Returns:

fldName	*fldAmount*
Desk	102

**** - Any field name with special characters (spaces) or reserved words will need to be enclosed in square brackets [ ]***

Table: tblProducts

fldName	*fldAmount*
Chair	0
Chair	56
Desk	42
Desk	60
Speakers	0
Computer Desk	1

Information

When passing values these formatting rules may need to be applied

Numerical: No special formatting required
String: Requires a single quote (') on each side of the value
Dates: Requires a pound sign (#) on each side of the value (Access). MySQL uses a single quote (').

WHERE

WHERE CONDITION(S)

Usage: Use the WHERE statement to selectively query a table in a database and only affect the records that meet your conditions.

Examples: (*reference Sample Table*)

- Affect only products named Chair
  WHERE fldName = 'Chair'
- Affect only products named Chair and amount is zero
  WHERE fldName = 'Chair' **AND** fldAmount = 0
- Affect only product names that start with C*
  WHERE fldName **Like** 'C%'
- Affect only products that have desk somewhere in the name
  WHERE fldName **Like** '%Desk%'
- Affect only products where the amount is null
  WHERE fldAmount **Is Null**
- Affect only chairs and desks
  WHERE fldName **IN** ('Chair','Desk')
- Affect products with an amount between 0 and 50
  WHERE fldAmount **BETWEEN** 0 And 50

* - It should be noted that Access uses the perctange (%) symbol for a wildcard while other databases may use an asterisk (*).

UPDATE

UPDATE TABLE_NAME
SET COLUMN NAME = VALUE
[**WHERE** CONDITION(S)]

Usage: Updates any or all fields in a table. Used to change the value of a single field, multiple fields, or all fields where a condition (if supplied) has been satisfied.

Examples: (*reference Sample Table*)

- Modify all records in the Products table (tblProducts)
  UPDATE tblProducts
  SET fldName = 'Pencil', fldAmount = 6
- Change all chairs to a zero amount
  UPDATE tblProducts
  SET fldAmount = 0
  WHERE fldName = 'Chair

INSERT INTO

INSERT INTO TABLE_NAME
  [COLUMN LIST]
 VALUE (VALUE LIST)

Usage: Adds a new record into an existing table. Used to add a record into a table by explicitly defining the columns or by passing values in the order the columns appear in the table

Examples: (*reference Sample Table*)

- Add a record into the table (values added in the same order as the columns)
  INSERT INTO tblProducts
   VALUES ('Book',10)
- Change all chairs to a zero amount
  INSERT INTO tblProducts (fldName, fldAmount)
   VALUES ('Book',10)

** - Any columns that are not defined will be given either a Null value or the default value as set in the columns default value property. Autonumber columns should not be passed as the database will automatically assign a value when the record is added.*

SELECT INTO

SELECT COLUMN_NAME(S) **INTO** NEW_TABLE_NAME
  [**IN** EXTERNAL_DATABASE]
FROM SOURCE_TABLE_NAME
  [**WHERE** CONDITION(S)]

Usage: Used to create a make-table query. The most common use for this statement is for making backup copies of tables. The **SELECT...INTO** statement doesn't define a primary key for the new table.

Examples:

- Make a complete backup of a table*
  SELECT * INTO BackupTable **FROM** SourceTable
- Make a backup of select columns inside of a table
  SELECT fldOne, fldTwo **INTO** BackupTable **FROM** SourceTable
- Make a complete backup of a table into a different database*
  SELECT fldOne, fldTwo **INTO** BackupTable
   IN 'Backup.mdb'
  FROM SourceTable

** - The asterisk should not be used and the field names should be explicitly listed out. This was done for demonstration purposes only.*

DELETE

DELETE FROM TABLE_NAME
  [**WHERE** CONDITION(S)]

Usage: Deletes a single or multiple records from a table. Can be used to delete a single record, multiple records (using the WHERE clause), or all records.

Examples: (*reference Sample Table*)

- Delete all chairs from the products table
  DELETE FROM tblProducts **WHERE** fldName = 'Chair'
- Delete all products
  DELETE FROM tblProducts

DROP TABLE

DROP TABLE TABLE_NAME(s)

Usage: Deletes an entire table from a database. To delete multiple tables separate table names by commas

Examples: (*reference Sample Table*)

- Delete the products table from the database
  DROP TABLE tblProducts

ALTER TABLE

ALTER TABLE TABLE_NAME
{**ADD**
  {**COLUMN** field type[(size)] [**NOT NULL**] [**CONSTRAINT** index] |
  CONSTRAINT multifieldindex} |
DROP {**COLUMN** field | **CONSTRAINT** indexname} }

Usage: Modifies an existing table in a database.

Examples:

- Delete the fldAmount column from the tblProducts table
  ALTER TABLE tblProducts **DROP COLUMN** fldAmount
- Add a new Date column to the tblProducts table
  ALTER TABLE tblProducts **ADD COLUMN** fldDate DateTime

CREATE TABLE

CREATE TABLE TABLE_NAME
  (COLUMN_NAME DATA_TYPE [(SIZE)] [NOT NULL]
 COLUMN_CONSTRAINT,
   [, other column definitions,…]
   [, primary key constraint]
)

Usage: Creates a new table in an existing database. When creating the table field names and the data types must be specified.

Examples:

- Create the products table from code
  CREATE TABLE tblProducts (fldName varchar(50),
   fldAmount Integer)
- Create a table and force a field to require a value
  CREATE TABLE tblMyTable (TableID Long NOT NULL,
   fldName varchar(25))

Aggregate Functions

Operate against a collection of values, but return a single, summarizing value.

AVG (COLUMN_NAME) - Returns the average value of a column
COUNT (COLUMN_NAME) - Returns the row number for any row not containing a null value for the column
COUNT (*****) - Returns the number of selected rows
FIRST (COLUMN_NAME) - Returns the value of the first record for the specified field
LAST (COLUMN_NAME) - Returns the value of the last record for the specified field
MAX (COLUMN_NAME) - Returns the maximum value of a column
MIN (COLUMN_NAME) - Returns the minimum value of a column
SUM (COLUMN_NAME) - Return the total sum of a column

COUNT (**DISTINCT** COLUMN_NAME) - Returns the count for all unique column values*

** - Access does not support this aggregate function*

Examples: (*reference Sample Table*)

- Determine the greatest quantity of any product
  SELECT MAX(fldAmount) **FROM** tblProducts
   Returns: 60
- Determine how many unique products (Access)
  SELECT DISTINCT COUNT(fldName) **FROM** tblProducts
   Returns: 4

Scalar Functions

Operate against a single value, and return a single value based on the input value.

Sample of the common Scalar Functions:
AVG (COLUMN_NAME) - Returns the average value of a column
UCASE (COLUMN_NAME) - Converts a field to upper case
LCASE (COLUMN_NAME) - Converts a field to lower case
MID (COLUMN_NAME, start [,end]) - Extract characters from a text field
LEN (COLUMN_NAME) - Returns the length of a text field
INSTR (COLUMN_NAME) - Returns the numeric position of a named character within a text field
LEFT (COLUMN_NAME, number_of_char) - Return the left part of a text field requested
RIGHT (COLUMN_NAME, number_of_char) - Return the right part of a text field requested
ROUND (COLUMN_NAME, decimals) - Rounds a numeric field to the number of decimals specified
MOD (x,y) - Returns the remainder of a division operation
NOW () - Returns the current system date
FORMAT (COLUMN_NAME ,format) - Changes the way a field is displayed
DATEDIFF (d,date1,date2) - Used to perform date calculations