

Structured Query Language

SQL (pronounced "ess-que-el") stands for Structured Query Language. SQL is used to communicate with a database. It is the standard language for relational database management systems. SQL statements are used to perform tasks such as update data on a database, or retrieve data from a database.

Some common relational database management systems that use SQL are: Oracle, MS SQL Server, Access, Paradox & etc. Although most database systems use SQL, most of them also have their own additional proprietary extensions that are usually only used on their system. However, the standard SQL commands such as "Select", "Insert", "Update", "Delete", "Create", and "Drop" can be used to accomplish almost everything that one needs to do with a database.

Features of SQL

Main features of SQL are as follows:

- SQL provides an interface between relational database such as Oracle, Access, MS SQL Server and User. All SQL statements are instructions to the database.
- All programs written in SQL are portable, means they can often be moved from one database to another with very little modification.
- SQL is a non-procedural language, means access method of data is not required to be specified.
- SQL processes sets of data as groups rather than as individual units. For example SQL will retrieve all the rows from table, which meet the conditions and can use this result set into another operation or query, user is not required to deal with each row individually.
- SQL provides automatic navigation to the data, means user is not required to specify where the data is physically stored and how to retrieve it SQL does it automatically.

Components of SQL

In database, stores information in the form of tables, consisting of rows and columns. A row of a table represents a record and a column of a table represent an attribute. A table represents an entity while a record gives the information about one instance of entity. As we know SQL is an interface between the database and User. Now we will learn about how to create the tables to store the information, insert the information into the tables, retrieve the information from the tables via SQL. A specific instance of valid SQL commands is called an SQL statement.

SQL statements are broadly divided into two categories. Here we are giving a brief overview of each of these categories.

1. Data Definition Language (DDL) statements:

Data Definition Language (DDL) for creating, altering, and dropping tables. Main DDL statements are CREATE, ALTER and DROP.

2. Data Manipulation Language (DML) statements:

Data Manipulation Language statements are used to manipulate the data from database. These are the most frequently used commands. These statements are used for

Nottingham College of London

No: 36 1/1, Station Road, Colombo – 06, Sri Lanka
Tel +94 (0) 11 2598059, Fax: +94 (0) 11 2552559
Email: ncl@nclworld.net, www.nclworld.net

insert/ delete records or rows in the table, to perform queries on the table to see its contents, changing data values in the existing rows of the table.

Main DML statements are INSERT (to insert rows in the existing table), SELECT (to retrieve data from an existing table), DELETE (to delete rows from an existing table) and UPDATE (to update records of an existing table).

Select statement:

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 [distinct column1] [*] column1[, column2, etc] from TableName
[where condition] [order by column1 asc or desc];
```

Note : []- 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**.

The **LIKE** pattern matching operator can also be used in the conditional selection of the where clause. Like is a very powerful operator that allows you to select only rows that are "like" what you specify. The percent sign "%" can be used as a wild card to match any possible character that might appear before or after the characters specified. For example:

```
select first, last, city from Empinfo where first LIKE 'selva%';
```

This SQL statement will match any first names that start with 'selva'.

```
select first, last from Empinfo where last LIKE '%s';
```

This statement will match any last names that end in a 's'.

```
select * from Empinfo where first = 'selva';
```

This will only select rows where the first name equals 'selva' exactly.

Note : Strings & dates must be in single quotes.

Example : Table name – Employee

Field - name, empno ,salary, age

```
SELECT * from employee;
```

```
SELECT name, salary from employee;
```

```
SELECT * from employee where age = 30;
```

```
SELECT * from employee where age in (35,50,60,25);
```

```
SELECT * from employee where age != 30;
```

```
SELECT * from employee where age >= 30;
```

```
SELECT * from employee where name like ' %kumar% ';
```

```
SELECT * from employee where age = 30 or age = 40;
```

```
SELECT * from employee where age >= 30 or age <= 40;
```

```
SELECT * from employee where salary between 15000 and 40000 =30;
```

Select statement exercises:

Nottingham College of London

No: 36 1/1, Station Road, Colombo – 06, Sri Lanka
Tel +94 (0) 11 2598059, Fax: +94 (0) 11 2552559
Email: ncl@nclworld.net, www.nclworld.net

- 1 Display the first name and age for everyone that's in the table.
- 2 Display the first name, last name, and city for everyone that's not from Payson.
- 3 Display all columns for everyone that is over 40 years old.
- 4 Display the first and last names for everyone whose last name ends in an "ay".
- 5 Display all columns for everyone whose first name equals "Mary".
- 6 Display all columns for everyone whose first name contains "Mary".

Insert statement:

The **insert** statement is used to insert or add a row of data into the table. To insert records into a table, enter the key words **insert into** followed by the table name, followed by an open parenthesis, followed by a list of column names separated by commas, followed by a closing parenthesis, followed by the keyword **values**, followed by the list of values enclosed in parenthesis. The values that you enter will be held in the rows and they will match up with the column names that you specify.

```
insert into TableName (first_column,...last_column) values (first_value,...last_value);
```

Example:

```
insert into Employee (first, last, age, address, city, state)
values ('Selva', 'Kumar', 28, 'Colombo-3','Colombo','South');
```

Note: All strings should be enclosed between **single** quotes: 'string'

In the example above, the column name **first** will match up with the value 'Selva', and the column name **state** will match up with the value 'South'.

Insert statement exercises

To insert this data into your new employee table.

Jonie Weber,	Secretary,	28,	19500.00
Potsy Weber,	Programmer,	32,	45300.00
Dirk Smith,	Programmer II,	45,	75020.00

Update statement:

The update statement is used to update or change records that match a specified criteria. This is accomplished by carefully constructing a where clause.

```
update TableName
set "columnname" = "newvalue"[,"nextcolumn" = "newvalue2"...]
where "columnname" OPERATOR "value" [and|or "column" OPERATOR "value"];
```

```
update phone_book set area_code = 623 where prefix = 979;
```

```
update phone_book set last_name='Smith', prefix=555, suffix=9292 where last_name='Jones';
```

```
update employee set age = age+1 where first_name='Mary' and last_name='Williams';
```

Update statement exercises:

- 1 Jonie Weber just got married to Bob Williams. She has requested that her last name be updated to Weber-Williams.
- 2 Dirk Smith's birthday is today, add 1 to his age.
- 3 All secretaries are now called "Administrative Assistant". Update all titles accordingly.
- 4 Everyone that's making under 30000 are to receive a 3500 a year raise.
- 5 Everyone that's making over 33500 are to receive a 4500 a year raise.

Nottingham College of London

No: 36 1/1, Station Road, Colombo – 06, Sri Lanka
 Tel +94 (0) 11 2598059, Fax: +94 (0) 11 2552559
 Email: ncl@nclworld.net, www.nclworld.net

6 All "Programmer II" titles are now promoted to "Programmer III".

7 All "Programmer" titles are now promoted to "Programmer II".

Delete statement:

The delete statement is used to delete records or rows from the table. To delete an entire record/row from a table, enter "delete from" followed by the table name, followed by the where clause which contains the conditions to delete. If you leave off the where clause, all records will be deleted.

```
delete from tablename where columnname OPERATOR "value" [and|or "column" OPERATOR "value"];
```

Examples:

```
delete from employee;
```

Note: if you leave off the where clause, all records will be deleted!

```
delete from employee where lastname = 'May';
```

```
delete from employee where firstname = 'Mike' or firstname = 'Eric';
```

Delete statement exercises:

1 Praba just quit, remove her record from the table;

2 It's time for budget cuts. Remove all employees who are making over 5000 rupees.

Note:

SELECT, for retrieving existing data

INSERT, for adding new data to a table

UPDATE, for modifying existing data

DELETE, for removing existing data from a table

Create statement :

For example, the following statement creates a Access table

```
create table employee
(
  last_name text(20),
  first_name text(15),
  salary currency(10,2),
  age number,
)
```

Local SQL supports the following subset of the ANSI-standard ALTER TABLE statement.

You can add new columns to an existing table using this ALTER TABLE syntax:

```
ALTER TABLE table ADD column_name data_type [, ADD column_name data_type ...]
```

You can delete existing columns from a table using the following ALTER TABLE syntax:

```
ALTER TABLE table DROP column_name [, DROP column_name ...]
```

For example, the next statement drops two columns from a table:

```
ALTER TABLE "employee.dbf" DROP LAST_NAME, DROP FIRST_NAME
```

ADD and DROP operations can be combined in a single statement. For example, the following statement drops two columns and adds one:

```
ALTER TABLE "employee.dbf" DROP LAST_NAME, DROP FIRST_NAME, ADD
FULL_NAME CHAR[30]
```

Nottingham College of London

No: 36 1/1, Station Road, Colombo – 06, Sri Lanka
Tel +94 (0) 11 2598059, Fax: +94 (0) 11 2552559
Email: ncl@nclworld.net, www.nclworld.net

DROP TABLE deletes a table.

For example, the following statement drops a Paradox table:

```
DROP TABLE "employee.db"
```

Nottingham College of London

No: 36 1/1, Station Road, Colombo – 06, Sri Lanka
Tel +94 (0) 11 2598059, Fax: +94 (0) 11 2552559
Email: ncl@nclworld.net, www.nclworld.net

NCL is a Private Limited Liability Company Registered in Sri Lanka