

# THE SQL SELECT STATEMENT

## Retrieving data using the SQL Select Statement

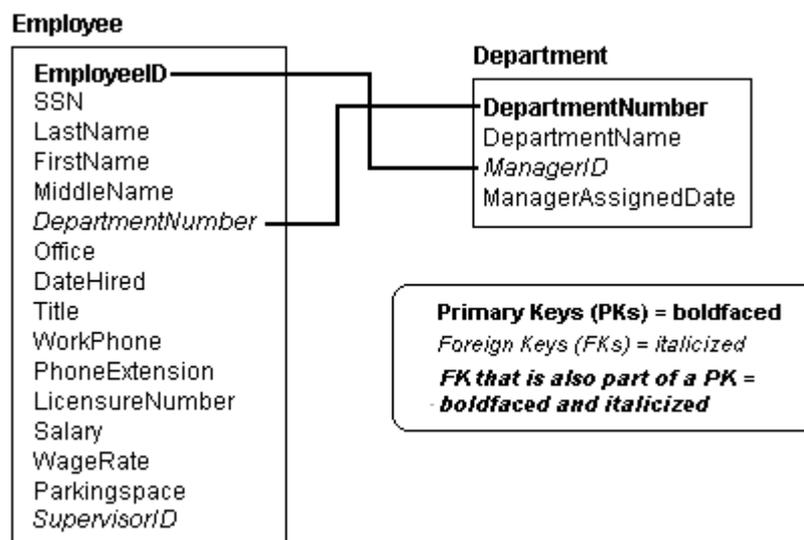
SQL is a comprehensive database language. SQL, pronounced Sequel or simply S-Q-L, is a computer programming language used for querying relational databases following a nonprocedural approach. When you extract information from a database using SQL, this is termed querying the database.

A relational database is implemented through the use of a Relational Database Management System *RDBMS*. An RDBMS performs all the basic functions of the DBMS software mentioned above along with a multitude of other functions that make the relational model easier to understand and to implement. RDBMS users manipulate data through the use of a special data manipulation language. Database structures are defined through the use of a data definition language. The commands that system users execute in order to store and retrieve data can be entered at a terminal with an RDBMS interface by typing the commands, or entered through use of some type of graphical interface. The DBMS then processes the commands.

## Capabilities of the SELECT Statement

Data retrieval from data base is done through appropriate and efficient use of SQL. Three concepts from relational theory encompass the capability of the SELECT statement: projection, selection, and joining.

- Projection: A project operation selects only certain columns *fields* from a table. The result table has a subset of the available columns and can include anything from a single column to all available columns.
- Selection: A select operation selects a subset of rows *records* in a table *relation* that satisfy a selection condition. The ability to select rows from out of complete result set is called Selection. It involves conditional filtering and data staging. The subset can range from no rows, if none of the rows satisfy the selection condition, to all rows in a table.
- Joining: A join operation combines data from two or more tables based on one or more common column values. A join operation enables an information system user to process the relationships that exist between tables. The join operation is very powerful because it allows system users to investigate relationships among data elements that might not be anticipated at the time that a database is designed.



Consider the above table structures. Fetching first\_name name, department\_id and salary for a single employee from EMPLOYEES table is Projection. Fetching employee details whose salary is less than 5000, from EMPLOYEES table is Selection. Fetching employee's first name, department name by joining EMPLOYEES and DEPARTMENTS is Joining.

## Basic SELECT statement

The basic syntax for a SELECT statement is presented below.

```
SELECT [DISTINCT | ALL] { * | select_list }
FROM { table_name [alias] | view_name }
    [{ table_name [alias] | view_name } ] ...
[WHERE condition]
[GROUP BY condition_list]
[HAVING condition]
[ORDER BY { column_name | column_# [ASC | DESC] } ...]
```

The SELECT clause is mandatory and carries out the relational project operation.

The FROM clause is also mandatory. It identifies one or more tables and/or views from which to retrieve the column data displayed in a result table.

The WHERE clause is optional and carries out the relational select operation. It specifies which rows are to be selected.

The GROUP BY clause is optional. It organizes data into groups by one or more column names listed in the SELECT clause.

The optional HAVING clause sets conditions regarding which groups to include in a result table. The groups are specified by the GROUP BY clause.

The ORDER BY clause is optional. It sorts query results by one or more columns in ascending or descending order.

## Arithmetic expressions and NULL values in the SELECT statement

An arithmetic expression can be created using the column names, operators and constant values to embed an expression in a SELECT statement. The operator applicable to a column depends on column's data type. For example, arithmetic operators will not fit for character literal values. For example,

```
SELECT employee_id, sal * 12 ANNUAL_SAL
FROM employees;
```

The above query contains the arithmetic expression  $sal * 12$  to calculate annual salary of each employee.

## Arithmetic operators

Operators act upon the columns *known as operands* to result into a different result. In case of multiple operators in an expression, the order of evaluation is decided by the operator precedence. Here are the elementary rules of precedence -

- Multiplication and division occur before multiplication and division.
- Operators on the same priority are evaluated from left to right.
- Use parentheses to override the default behavior of the operators.

Below table shows the precedence of the operators, in such cases. Precedence Level Operator Symbol Operation

Description	Operator	Precedence
Addition	+	Lowest
Subtraction	-	Lowest
Multiplication	*	Medium
Division	/	Medium
Brackets	( )	Highest

Examine the below queries *a*, *b*, and *c*

```
SQL> SELECT 2*35 FROM DUAL;
```

```
SQL> SELECT salary + 1500 FROM employees;
```

```
SQL> SELECT first_name, salary, salary + (commission_pct* salary) FROM employees;
```

Query *a* multiplies two numbers, while *b* shows addition of \$1500 to salaries of all employees. Query (c) shows the addition of commission component to employee's salary. As per the precedence, first commission would be calculated on the salary, and then added to the salary.

## Column Alias

An alias is used to rename a column or an expression during display. The alias to a column or an expression appears as the heading in the output of a query. It is useful in providing a meaningful heading to long expressions in the SELECT query. By default, the alias appears in uppercase in the query output without spaces. To override this behavior, alias must be enclosed within double quotes to preserve the case and spaces in the alias name.

```
SELECT price * 2 as DOUBLE_PRICE, price * 10 "Double Price"  
FROM products;
```

```
DOUBLE_PRICE Double Price
```

```
-----  
39.9      39.9  
60        60  
51.98     51.98
```

## Concatenation operators

Concatenation operator can be used to join two string values or expressions in a SELECT query. The double vertical bar symbol is used as string concatenation operator. It is applicable only for character and string column values resulting into a new character expression. Example

```
SQL> SELECT 'ORACLE' || ' CERTIFICATION' FROM dual;
```

The above query shows concatenation of two character literals values.

## Literals

Any hard coded value, which is not stored in database, in the SELECT clause, is known s Literal. It can be number, character, or date value. Character and date values must be enclosed within quotes. Consider the below SQL queries.examples of using literals of different data types in SQL queries.

The query below uses two character literals to join them together.

```
SQL> SELECT 'ORACLE' || ' CERTIFICATION' FROM DUAL
```

The query below uses character literals to pretty print the employee's salary.

```
SQL> SELECT first_name || 'earns' || salary || ' as of ' || sysdate  
FROM employees
```

## Quote Operator

The quote operator is used to specify the quotation mark delimiter of your own. You can chose a convenient delimiter, dedepnding on the data.

```
SELECT department_name || ' Department' || q['s Manager Id: ']' || manager_id  
FROM departments;
```

## NULL

If a column doesn't have a definite value, it is considered as NULL. NULL value denotes unknown or unavailable. It is not zero for numeric values, not blank space for character values.

Columns with NULL value can be selected in a SELECT query and can be the part of an arithmetic expression. Any arithmetic expression using NULL values results into NULL. For this reason, columns with NULL value must be handled differently by specifying their alternate values using Oracle supplied functions like NVL or NULLIF.

```
SQL> SELECT NULL + 1000 NUM
FROM DUAL;

NUM
-----
```

## DISTINCT Keyword

If the data is expected to have duplicate results, use DISTINCT keyword to eliminate duplicates and display only the unique results in the query output. Only the selected columns are validated for duplication and the rows will be logically eliminated from the query output. To be noted, the DISTINCT keyword must appear just after the SELECT clause.

The simple query below demonstrates the use of DISTINCT to display unique department ids from EMPLOYEES table.

```
SQL> SELECT DISTINCT DEPARTMENT_ID
FROM employees;

DEPARTMENT_ID
-----
10
20
30
40
```

## DESCRIBE command

The structural metadata of a table may be obtained by querying the database for the list of columns that comprise it using the DESCRIBE command. It will list the used column names, their null property and data type.

### Syntax:

```
DESC[RIBE] [SCHEMA].object name
```

For example,

```
DESC EMPLOYEE
```

will display the EMPLOYEE table structure i.e. columns, their data types, precision and nullable property.

Loading [MathJax]/jax/output/HTML-CSS/jax.js