**Computer Lab: SQL Server-2**

# Data Manipulation

This section looks at the SQL DML statements, namely:

1. SELECT – to query data in the database.
2. INSERT – to insert data into a table.
3. UPDATE – to update data in a table.
4. DELETE – to delete data from a table.

## SQL SELECT Statement

The purpose of the SELECT statement is to retrieve and display data from one or more database tables. SELECT is the most frequently used SQL command and has the following general form:

```
SELECT [DISTINCT | ALL] {*| [columnExpression [AS newName]] [, . . ]}
FROM TableName [alias] [, . . . ]
[WHERE condition]
[GROUP BY columnList] [HAVING condition]
[ORDER BY columnList]
```

- *columnExpression* represents a column name or an expression, *TableName* is the name of an existing database table or view that you have access to, and *alias* is an optional abbreviation for *TableName*. The sequence of processing in a SELECT statement is:
    - FROM specifies the table or tables to be used
    - WHERE filters the rows subject to some condition
    - GROUP BY forms groups of rows with the same column value
    - HAVING filters the groups subject to some condition.
    - SELECT specifies which columns are to appear in the output.
    - ORDER BY specifies the order of the output.

- The order of the clauses in the SELECT statement *cannot* be changed.
- The only two mandatory clauses are the first two: SELECT and FROM; the remainder are optional.
- The SELECT operation is **closed**

- The result of a query on a table **is another table**.

---

**We illustrate the SQL statements using the instance of the following tables:**
**Staff (staffNo, fName, lName, city, position, salary, branchNo)**

Results Messages

| | staffNO | fName | lName | city | position | salary | branchNo |
|---|---|---|---|---|---|---|---|
| 1 | s1 | Gad | Ahmed | Tanta | Manager | 60000 | B02 |
| 2 | s2 | Samy | Tamer | Cairo | Assistant | 30000 | B02 |
| 3 | s3 | Samy | Ahmed | Tanta | Supervisor | 50000 | B01 |
| 4 | s4 | Tamer | Samir | Giza | Manager | 40000 | B03 |

### Example 1

Retrieve all columns all rows….. or
List full details of all staff.

```
SELECT staffNo, fName, lName, city, position, salary, branchNo
FROM Staff;
```

- Instead of writing all column names, there is a quick way:
  use an asterisk (*) in place of the column names.

The following statement is an equivalent and shorter way of expressing this query:

```
SELECT *
FROM Staff;
```

---

The result table in either case is shown in Table 7.1.

**Table 1 Result table for Example 1**

Results | Messages

| | staffNO | fName | lName | city | position | salary | branchNo |
|---|---|---|---|---|---|---|---|
| 1 | s1 | Gad | Ahmed | Tanta | Manager | 60000 | B02 |
| 2 | s2 | Samy | Tamer | Cairo | Assistant | 30000 | B02 |
| 3 | s3 | Samy | Ahmed | Tanta | Supervisor | 50000 | B01 |
| 4 | s4 | Tamer | Samir | Giza | Manager | 40000 | B03 |

---

**Example 2**

Retrieve specific columns, all rows….. or
Produce a list of salaries for all staff, showing only the staff number, the first and last names, and the salary details.

```
SELECT staffNo, fName, lName, salary
FROM Staff;
```

In this example a new table is created from Staff containing only the designated columns staffNo, fName, lName, and salary, in the specified order. The result of this operation is shown in Table 7.2.

**Table 2** Result table for Example 2

Results | Messages

| | staffNo | fName | lName | salary |
|---|---|---|---|---|
| 1 | s1 | Gad | Ahmed | 60000 |
| 2 | s2 | Samy | Tamer | 30000 |
| 3 | s3 | Samy | Ahmed | 50000 |
| 4 | s4 | Tamer | Samir | 40000 |

---

**Example 3**

Use of DISTINCT…… or
List all cities of staff.

```
SELECT City
FROM staff;
```

Table 3

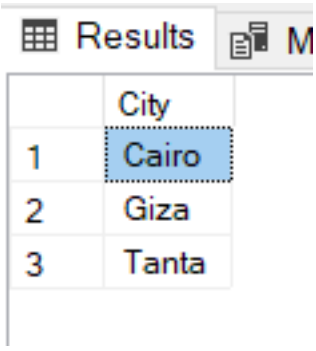

The result table is shown in Table 3.

- Notice that there are several duplicates.
- SELECT does not eliminate duplicates when it projects over one or more columns.
- To eliminate the duplicates, we use the DISTINCT keyword. Rewriting the query as:

```
SELECT DISTINCT City
FROM Staff;
```

we get the result table shown in Table 3 with the duplicates eliminated.

Table 3 result of example 3



**Example 4**

Calculated fields…… or
Produce a list of monthly salaries for all staff, showing the staff number, the first and last names, and the salary details.

```
SELECT staffNo, fName, lName, salary/12
FROM Staff;
```

In this case, the desired result can be obtained by simply dividing the salary by 12, giving the result table shown in Table 4.

- This is an example of the use of a **calculated field** (sometimes called a **computed** or **derived field**).
- In general, to use a calculated field you specify an SQL expression in the SELECT list.
- An SQL expression can involve addition, subtraction, multiplication, and division, and parentheses can be used to build complex expressions.

**Table 4** Result table for Example 4

Results | Messages

| | staffNo | fName | lName | (No column name) |
|---|---|---|---|---|
| 1 | s1 | Gad | Ahmed | 5000 |
| 2 | s2 | Samy | Tamer | 2500 |
| 3 | s3 | Samy | Ahmed | 4166 |
| 4 | s4 | Tamer | Samir | 3333 |

Note (in older versions):

- The fourth column of this result table has been output as *col4*.
- Normally, a column in the result table takes its name from the corresponding column of the database table from which it has been retrieved.
- The ISO standard allows the column to be named using an AS clause.

In the previous example, we could have written:

```
SELECT staffNo, fName, lName, salary/12 AS monthlySalary
FROM Staff;
```

In this case the column heading of the result table would be monthlySalary rather than col4.