

## EXERCISE-4

### Writing Basic SQL SELECT Statements

#### OBJECTIVES

After the completion of this exercise, the students will be able to do the following:

- List the capabilities of SQL SELECT Statement
- Execute a basic SELECT statement

#### Capabilities of SQL SELECT statement

A SELECT statement retrieves information from the database. Using a select statement, we can perform

- ✓ Projection: To choose the columns in a table
- ✓ Selection: To choose the rows in a table
- ✓ Joining: To bring together the data that is stored in different tables

#### Basic SELECT Statement

##### Syntax

```
SELECT *|DISTINCT Column_name| alias  
      FROM table_name;
```

##### NOTE:

DISTINCT—Supr  
ess the duplicates.  
Alias—gives selected columns different headings.

##### Example: 1

```
SELECT * FROM departments;
```

##### Example: 2

```
SELECT location_id, department_id FROM departments;
```

#### Writing SQL Statements

- SQL statements are not case sensitive
- SQL statements can be on one or more lines.
- Keywords cannot be abbreviated or split across lines
- Clauses are usually placed on separate lines
- Indents are used to enhance readability

#### Using Arithmetic Expressions

Basic Arithmetic operators like \*, /, +, -can be used

##### Example:1

```
SELECT last_name, salary, salary+300 FROM employees;
```

##### Example:2

```
SELECT last_name, salary, 12*salary+100 FROM employees;
```

The statement is not same as

```
SELECT last_name, salary, 12*(salary+100) FROM employees;
```

### Example:3

```
SELECT last_name, job_id, salary, commission_pct FROM employees;
```

### Example:4

```
SELECT last_name, job_id, salary, 12*salary*commission_pct FROM employees;
```

### Using Column Alias

- To rename a column heading with or without AS keyword.

#### Example:1

```
SELECT last_name AS Name  
FROM employees;
```

#### Example: 2

```
SELECT last_name "Name" salary*12 "Annual Salary"  
FROM employees;
```

### Concatenation Operator

- Concatenates columns or character strings to other columns
- Represented by two vertical bars (||)
- Creates a resultant column that is a character expression

#### Example:

```
SELECT last_name||job_id AS "EMPLOYEES JOB" FROM employees;
```

### Using Literal Character String

- A literal is a character, a number, or a date included in the SELECT list.
- Date and character literal values must be enclosed within single quotation marks.

#### Example:

```
SELECT last_name||'s'||job_id AS "EMPLOYEES JOB" FROM employees;
```

### Eliminating Duplicate Rows

- Using DISTINCT keyword.

#### Example:

```
SELECT DISTINCT department_id FROM employees;
```

### Displaying Table Structure

- Using DESC keyword.

### Syntax

```
DESC table_name;
```

#### Example:

```
DESC employees;
```

### Find the Solution for the following:

True OR False

1. The following statement executes successfully.

### Identify the Errors

```
SELECT employee_id, last_name  
sal*12 ANNUAL SALARY
```

FROM employees;

### Queries

2. Show the structure of departments the table. Select all the data from it.

des < department;

3. Create a query to display the last name, job code, hire date, and employee number for each employee, with employee number appearing first.

Select employee-id, last-name, job-id, hire-date from employee

4. Provide an alias STARTDATE for the hire date.

Select hire-date as STARTDATE from employees

5. Create a query to display unique job codes from the employee table.

Select distinct job-id from employees;

6. Display the last name concatenated with the job ID , separated by a comma and space, and name the column EMPLOYEE and TITLE.

Select CONCAT\_WS(' ', l-name, job-code) AS = EMPLOYEE AND  
TITLE " from employee;

7. Create a query to display all the data from the employees table. Separate each column by a comma. Name the column THE\_OUTPUT.

Select CONCAT\_WS(' ', employee-number, f-name, l-name, job-code,  
hire-date ) AS THE\_OUTPUT  
FROM employee;

Evaluation Procedure	Marks awarded
Query(5)	5
Execution (5)	5
Viva(5)	5
Total (15)	15
Faculty Signature	BPL