**National University of Computer & Emerging Sciences, Karachi**

**Computer Science Department Fall 2025, Lab Manual - 01**

|  |  |
| --- | --- |
| **Course Code: CL-2005** | **Course: Database Systems Lab** |
| **Instructor(s):** | **Fatima Gado** |

**Contents:**

1. Database
2. SQL
3. Basic SQL Concepts

**Database**

A database is a systematic collection of data. They support electronic storage and manipulation of data. Databases make data management easy.

**Example #1**

An online telephone directory uses a database to store data of people, phone numbers, and other contact details. Your electricity service provider uses a database to manage billing, client-related issues, handle fault data, etc.

**Example #2**

Facebook needs to store, manipulate, and present data related to members, their friends, member activities, messages, advertisements, and a lot more. We can provide a countless number of examples for the usage of databases.

**SQL**

**SQL** is the standard language for dealing with Relational Databases. SQL can be used to insert, search, update, and delete database records. SQL can do lots of other operations, including optimizing and maintenance of databases. SQL stands for Structured Query language, pronounced as "S-Q-L" or sometimes as "See-Quel"... Relational databases like MySQL Database, Oracle, MS SQL Server, Sybase, etc. use ANSI SQL.

**Basic SQL Concepts**

1. **Data Types**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| bigint | decimal | real | char | nvarchar |
| Int | numeric | datetime | varchar | nvarchar(max) |
| smallint | money | smalldatetime | varchar(max) | ntext |
| tinyint | smallmoney | date | text | binary |
| Bit | float | time | nchar | varbinary |
| varbinary(max) | image |  |  |  |

1. **Arithmetic operators**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Addition | Subtraction | Multiplication | Division | Modulus |
| **+** | **-** | **\*** | **/** | **%** |

1. **SQL Comparison Operators**

|  |  |
| --- | --- |
| **=** | Checks if the values of two operands are equal or not, if yes then condition becomes true. |
| **!=** | Checks if the values of two operands are equal or not, if values are not equal then condition becomes true. |
| **<>** | Checks if the values of two operands are equal or not, if values are not equal then condition becomes true. |
| **>** | Checks if the value of left operand is greater than the value of right operand, if yes then condition becomes true. |
| **<** | Checks if the value of left operand is less than the value of right operand, if yes then condition becomes true. |
| **>=** | Checks if the value of left operand is greater than or equal to the value of right operand, if yes then condition becomes true. |
| **<=** | Checks if the value of left operand is less than or equal to the value of right operand, if yes then condition becomes true. |
| **!<** | Checks if the value of left operand is not less than the value of right operand, if yes then condition becomes true. ***(It’s a non-standard comparison operator, it is equivalent to >= )*** |
| **!>** | Checks if the value of left operand is not greater than the value of right operand, if yes then condition becomes true. ***(It’s a non-standard comparison operator, it is equivalent to <= )*** |

1. **SQL Logical Operators**

|  |  |
| --- | --- |
| **ALL** | The ALL operator is used to compare a value to all values in another value set. |
| **AND** | The AND operator allows the existence of multiple conditions in an SQL statement's WHERE clause |
| **ANY** | The ANY operator is used to compare a value to any applicable value in the list as per the condition. |
| **BETWEEN** | The BETWEEN operator is used to search for values that are within a set of values, given the minimum value and the maximum value. |
| **EXISTS** | The EXISTS operator is used to search for the presence of a row in a specified table that meets a certain criterion. |
| **IN** | The IN operator is used to compare a value to a list of literal values that have been specified. |
| **LIKE** | The LIKE operator is used to compare a value to similar values using wildcard operators. |
| **NOT** | The NOT operator reverses the meaning of the logical operator with which it is used. Eg: NOT EXISTS, NOT BETWEEN, NOT IN, etc. **This is a negate operator.** |
| **OR** | The OR operator is used to combine multiple conditions in an SQL statement's WHERE clause. |
| **NULL** | The NULL operator is used to compare a value with a NULL value. |
| **UNIQUE** | The UNIQUE operator searches every row of a specified table for uniqueness (no duplicates). |

1. **Basic SQL Queries**

**Note: Connect the HR Database in SqlDeveloper**

Select \* from EMPLOYEES

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **EMPLOYEE\_I D** | **FIRST\_NAME** | **LAST\_NAME** | **EMAIL** | **PHONE\_NUMBER** | **HIRE\_DAT E** | **JOB\_ID** |
| 100 | Steven | King | SKING | 515.123.4567 | 17-Jun-03 | AD\_PRES |
| 101 | Neena | Kochhar | NKOCHHAR | 515.123.4568 | 21-Sep-05 | AD\_VP |
| 102 | Lex | De Haan | LDEHAAN | 515.123.4569 | 13-Jan-01 | AD\_VP |
|  |  |  |  |  |  |  |
| 1023 | Lex3 | De Haanas | LDEsdaHAA N | 515.123.4569 | 13-Jan-01 | AD\_VPP |

Select EMPLOYEE\_ID, FIRST\_NAME, SALARY from EMPLOYEES

|  |  |  |
| --- | --- | --- |
| **EMPLOYEE\_ID** | **FIRST\_NAME** | **SALARY** |
| 100 | Steven | 24000 |
| 101 | Neena | 17000 |
| 102 | Lex | 17000 |
| - |  |  |
| 1023 | Lex3 | 12000 |

Select EMPLOYEE\_ID, FIRST\_NAME, SALARY from EMPLOYEES where salary>2300

|  |  |  |
| --- | --- | --- |
| **EMPLOYEE\_ID** | **FIRST\_NAME** | **SALARY** |
| 100 | Steven | 24000 |

Select EMPLOYEE\_ID, FIRST\_NAME, SALARY from EMPLOYEES where salary greater than or equal to 10000 and less than or equal to 12000

|  |  |  |
| --- | --- | --- |
| **EMPLOYEE\_ID** | **FIRST\_NAME** | **SALARY** |
| 114 | Den | 11000 |
| 147 | Alberto | 12000 |
| 148 | Gerald | 11000 |
| 149 | Eleni | 10500 |
| 114 | Den | 11000 |

**Let’s Practice More!!**

**Comparison operator:**

* SELECT \* FROM EMPLOYEES WHERE MANAGER\_ID = 101;
* SELECT \* FROM EMPLOYEES WHERE MANAGER\_ID < 110;
* SELECT \* FROM EMPLOYEES WHERE MANAGER\_ID > 200;
* SELECT \* FROM EMPLOYEES WHERE MANAGER\_ID >= 200;
* SELECT \* FROM EMPLOYEES WHERE MANAGER\_ID <= 150;
* SELECT \* FROM EMPLOYEES WHERE MANAGER\_ID <> 114;

**Logical Operators:**

* SELECT FIRST\_NAME,SALARY,JOB\_ID,DEPARTMENT\_ID FROM EMPLOYEES WHERE JOB\_ID = 'AD\_VP' AND DEPARTMENT\_ID = 90;
* SELECT FIRST\_NAME,SALARY,JOB\_ID,DEPARTMENT\_ID FROM EMPLOYEES WHERE JOB\_ID = 'AD\_VP' OR DEPARTMENT\_ID = 90;
* SELECT FIRST\_NAME,SALARY,JOB\_ID,DEPARTMENT\_ID FROM EMPLOYEES WHERE Not JOB\_ID = 'AD\_VP';
* SELECT EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, DEPARTMENT\_ID FROM EMPLOYEES WHERE DEPARTMENT\_ID IN (10, 20, 30);
* SELECT EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, DEPARTMENT\_ID FROM EMPLOYEES WHERE DEPARTMENT\_ID NOT IN (90);
* SELECT EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, MANAGER\_ID FROM EMPLOYEES WHERE MANAGER\_ID IS NULL;

**LAB TASKS:**

* 1. Write a SQL statement to display all the information of table **Jobs**.
  2. Write a SQL query to find min and max salary of the Job table with Job title ‘President’ from **Jobs**

table.

* 1. Write a SQL query to find those employees whose Salaries is greater than 20000 from **Employees**

table.

* 1. Write a SQL query to find the Jobs whose salary are higher than or equal to $15000 from **Employees**

table.

* 1. Write a SQL query to find the details of employees whose last name is 'Snares'. Return employee ID, employee first name, employee last name and employee dept ID.
  2. Write a SQL query to find the details of the employees who work in the department 57. Return employee ID, employee first name, employee last name and employee dept ID.
  3. Write a query to find the PHONE\_NUMBER of the DEPARTMENT\_ID=80 and MANAGER\_ID=100 of **Employees** table.
  4. write a SQL query to find the Employees with the First name “John” “NEENA” and “Lency”
  5. Write a query to find the list of cities with country ID ‘IT’ from **locations** table.
  6. Write a query to find the list of city except country ID ‘IN’ and ‘CH’ from **locations** table.
  7. Write a query to find the list of jobs whose min salary is greater than 8000 and less than 15,000 from

**job** table.

* 1. Write a query to find list of phone with DEPARTMENT\_ID ‘90’ but not with job\_id ‘IT\_PROG’ from **Employees** table.
  2. Write a query to find the list of employees who are hired after '12-Dec-07' from employee table.
  3. Write a query to find the list of employees who are hired after '12-Dec-07' in Department with DEPARTMENT\_ID=100 from employee table.
  4. Write a query to find the list of employees who are hired after '12-Dec-07' but not in Department with DEPARTMENT\_ID=100 from employee table.
  5. Write a query to find the list of employees whose COMMISSION\_PCT=0 and they do not belong to DEPARTMENT\_ID 90 or 100 from **Employees** table
  6. Write a query to find the employees who are hired in year 2010 from **Employees** table.
  7. Write a query to find the list of jobs whose min salary is greater than 8000 and less than 15,000 from

**job** table.

* 1. Write a query to find employee whose ID are greater than 100 and less than 150 and their department\_id is greater than 90 and less than 100 along with their F\_name, Last\_name & Job ID.
  2. Write a query to find total salary along with salary & commission\_pct Total salary formula = commission\_pct, salary+(commission\_pct\*salary)

**Lab Practice Problems:**

1. Write a query to find the email addresses of employees with MANAGER\_ID=101 and DEPARTMENT\_ID=50 from the Employees table.
2. Write a SQL statement to display all the information from the Departments table where the department name is 'Finance/Sales/SA\_RE'.
3. Write a SQL query to find the average salary of the job title 'Manager' from the Jobs table.
4. Write a SQL query to find those employees whose salaries are less than 10,000 from the Employees table.
5. Write a SQL query to find the jobs whose maximum salary is less than or equal to 20,000 from the Jobs table.
6. rite a SQL query to find the details of employees whose last name is 'Jones'. Return employee ID, employee first name, employee last name, and employee department ID.
7. Write a query to find the list of cities with country ID 'IT' from the locations table.
8. Write a query to find the phone numbers of employees who work in department 80 and have a manager ID of 100.
9. Retrieve employees who do not work in the 'Sales' department.
10. Find employees whose names start with the letter 'A'.
11. Retrieve employees who work in the 'Sales' department and have a salary greater than 50,000.