Lab – 03

# Basic SQL Select Statements

## Objective:

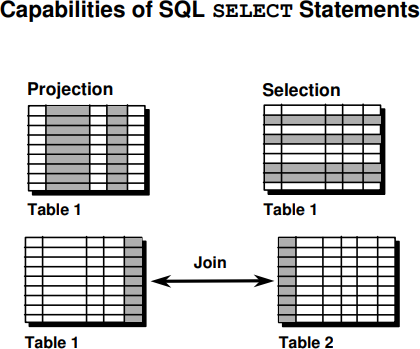
* Students will be able to list the basic capabilities of SELECT statements, execute basic SELECT statements.

# Capabilities of SQL SELECT Statements

A SELECT statement retrieves information from the database. Using a SELECT statement, you can do

the following:

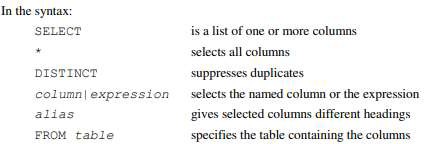
* **Projection**: You can use the projection capability in SQL to choose the columns in a table that you want returned by your query. You can choose as few or as many columns of the table as you require.
* **Selection:** You can use the selection capability in SQL to choose the rows in a table that you want returned by a query. You can use various criteria to restrict the rows that you see.
* **Joining:** You can use the join capability in SQL to bring together data that is stored in different tables by creating a link between them. You learn more about joins in a later lesson.

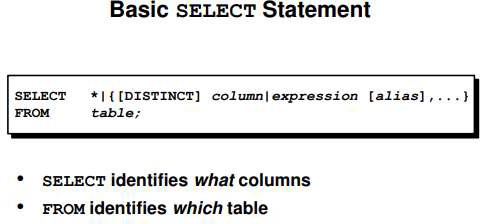


# Basic SELECT Statement

In its simplest form, a SELECT statement must include the following:

* + A SELECT clause, which specifies the columns to be displayed
  + A FROM clause, which specifies the table containing the columns listed in the SELECT clause





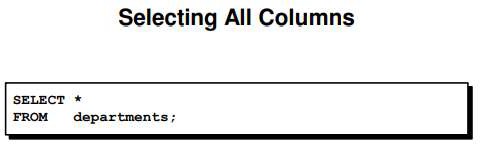
# Selecting All Columns of All Rows

You can display all columns of data in a table by following the SELECT keyword with an asterisk (\*). In the example on the slide, the department table contains four columns: DEPARTMENT\_ID, DEPARTMENT\_NAME, MANAGER\_ID, and LOCATION\_ID. The

table contains seven rows, one for each department.

You can also display all columns in the table by listing all the columns after the SELECT keyword. For example, the following SQL statement, like the example on the slide, displays all columns and all rows of the DEPARTMENTS table:

SELECT department\_id, department\_name, manager\_id, location\_ id FROM departments;

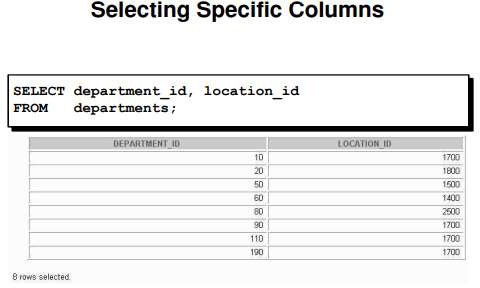


# Selecting Specific Columns of All Rows

You can use the SELECT statement to display specific columns of the table by specifying the column names, separated by commas. The example on the slide displays all the department numbers and location numbers from the DEPARTMENTS table.

In the SELECT clause, specify the columns that you want, in the order in which you want them to appear in the output. For example, to display location before department number going from left to right, you use the following statement:

SELECT location\_id, department\_id FROM departments;



# Writing SQL Statements

Using the following simple rules and guidelines, you can construct valid statements that are both easy to read and easy to edit:

* + SQL statements are not case sensitive, unless indicated.
  + SQL statements can be entered on one or many lines.
  + Keywords cannot be split across lines or abbreviated.
  + Clauses are usually placed on separate lines for readability and ease of editing.
  + Indents should be used to make code more readable.
  + Keywords typically are entered in uppercase; all other words, such as table names and columns, are entered in lowercase.

# Executing SQL Statements

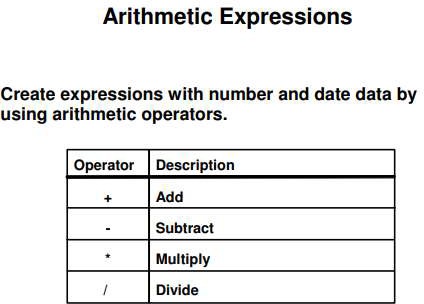
Using iSQL\*Plus, click the Execute button to run the command or commands in the editing window.

# Arithmetic Expressions

You may need to modify the way in which data is displayed, perform calculations, or look at what- if scenarios. These are all possible using arithmetic expressions. An arithmetic expression can contain column names, constant numeric values, and the arithmetic operators.

# Arithmetic Operators

The slide lists the arithmetic operators available in SQL. You can use arithmetic operators in any clause of a SQL statement except in the FROM clause.

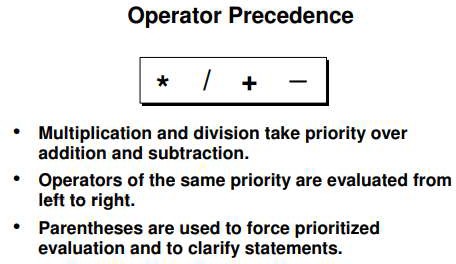


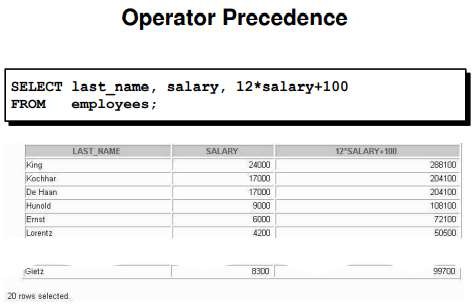
# Using Arithmetic Operators

The example in the slide uses the addition operator to calculate a salary increase of $300 for all employees and displays a new SALARY+300 columns in the output.

Note that the resultant calculated column SALARY+300 is not a new column in the EMPLOYEES table; it is for display only. By default, the name of a new column comes from the calculation that generated it—in this case, salary+300.

# Operator Precedence

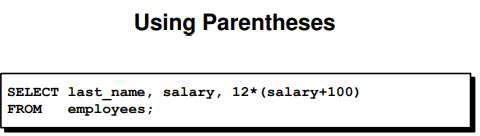




**Using Parentheses**

You can override the rules of precedence by using parentheses to specify the order in which operators are executed.

The example in the slide displays the last name, salary, and annual compensation of each employee. It calculates the annual compensation as monthly salary plus a monthly bonus of $100, multiplied by 12. Because of the parentheses, addition takes priority over multiplication.



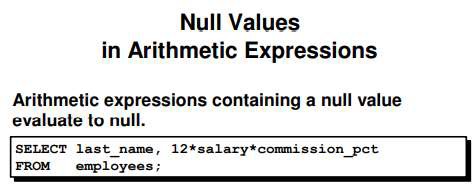
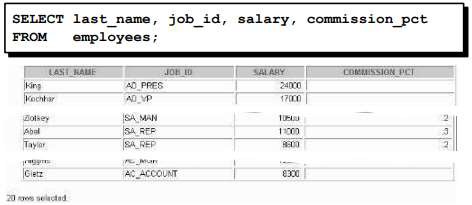
# Null Values

If a row lacks the data value for a particular column, that value is said to be null, or to contain a null.

A null is a value that is unavailable, unassigned, unknown, or inapplicable. A null is not the same as zero or a space. Zero is a number, and a space is a character.

Columns of any data type can contain nulls. However, some constraints, NOT NULL and PRIMARY KEY, prevent nulls from being used in the column.

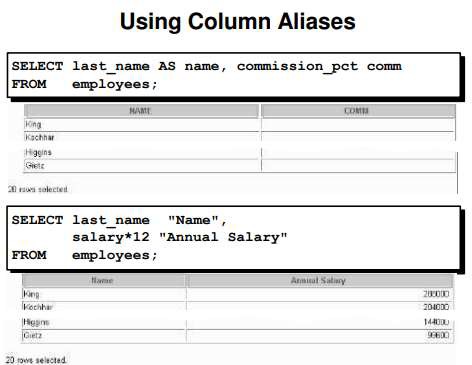
In the COMMISSION\_PCT column in the EMPLOYEES table, notice that only a sales manager or sales representative can earn a commission. Other employees are not entitled to earn commissions. A null represents that fact.



# Column Aliases

When displaying the result of a query, iSQL\*Plus normally uses the name of the selected column as the column heading. This heading may not be descriptive and hence may be difficult to understand. You can change a column heading by using a column alias.

Specify the alias after the column in the SELECT list using a space as a separator. By default, alias headings appear in uppercase. If the alias contains spaces or special characters (such as # or $), or is case sensitive, enclose the alias in double quotation marks (" ").

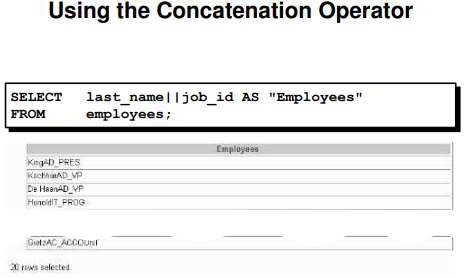


# Concatenation Operator

You can link columns to other columns, arithmetic expressions, or constant values to create a character expression by using the concatenation operator (||). Columns on either side of the operator are combined to make a single output column.

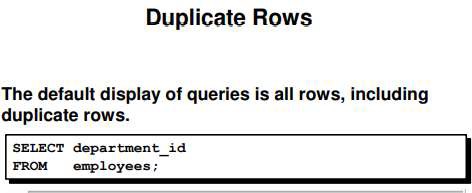
# Using the Concatenation Operator

In the example, LAST\_NAME and JOB\_ID are concatenated, and they are given the alias Employees. Notice that the employee last name and job code are combined to make a single output column. The AS keyword before the alias name makes the SELECT clause easier to read



# Duplicate Rows

Unless you indicate otherwise, iSQL\*Plus displays the results of a query without eliminating duplicate rows. The example on the slide displays all the department numbers from the EMPLOYEES table. Notice that the department numbers are repeated.



# Eliminating Duplicate Rows

To eliminate duplicate rows in the result, include the DISTINCT keyword in the SELECT clause immediately after the SELECT keyword. In the example on the slide, the EMPLOYEES table actually contains 20 rows but there are only seven unique department numbers in the table.

You can specify multiple columns after the DISTINCT qualifier. The DISTINCT qualifier affects all the selected columns, and the result is every distinct combination of the columns. SELECT DISTINCT department\_id, job\_id FROM employees;

## Lab Tasks:

1. Initiate an iSQL\*Plus session using the user ID and password provided by the instructor
2. iSQL\*Plus commands access the database. True/False
3. The following SELECT statement executes successfully:



True/False

1. The following SELECT statement executes successfully

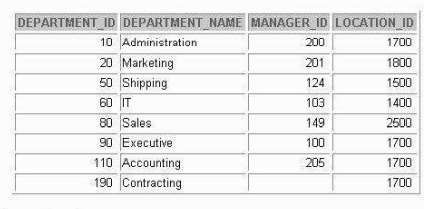
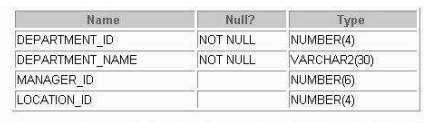


True/False

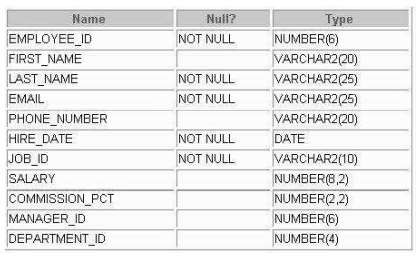
1. There are four coding errors in this statement. Can you identify them?

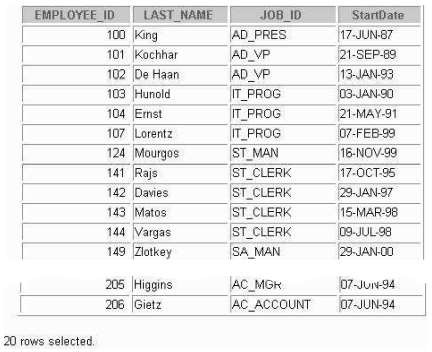


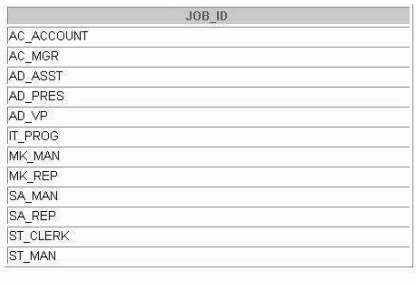
1. Show the structure of the DEPARTMENTS table. Select all data from the table



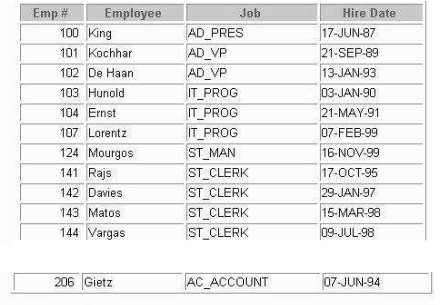
1. Show the structure of the EMPLOYEES table. Create a query to display the last name, job code, hire date, and employee number for each employee, with employee number appearing first. Save your SQL statement to a file named lab1\_7.sql



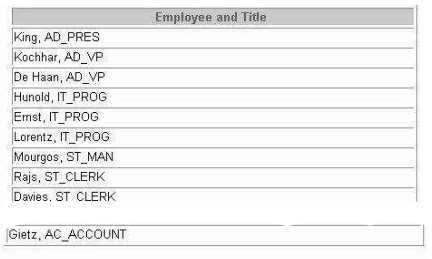
1. Run your query in the file lab1\_7.sql
2. Create a query to display unique job codes from the EMPLOYEES table



1. Copy the statement from lab1\_7.sql into the iSQL\*Plus Edit window. Name the column headings Emp #, Employee, Job, and Hire Date, respectively. Run your query again



1. Display the last name concatenated with the job ID, separated by a comma and space, and name the column Employee and Title.



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

**Table**

CREATE TABLE Employees(

ID int,

FirstName varchar(255),

LastName varchar(255),

JoiningDate varchar(255),

Salary varchar(255)

);

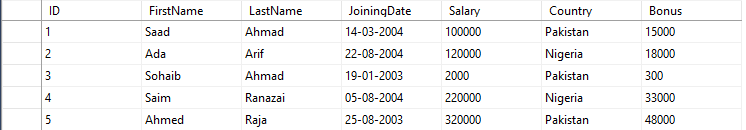
INSERT INTO Employees VALUES(1,'Saad','Ahmad','14-03-2004','20000'),

(2,'Ada','Arif','22-08-2004','120000'),

(3,'Sohaib','Ahmad','19-01-2003','2000'),

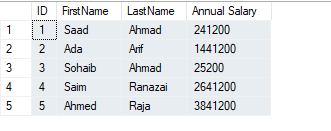
(4,'Saim','Ranazai','05-08-2004','220000'),

(5,'Ahmed','Raja','25-08-2003','320000');

****

**Arithmetic Operators & AS**

SELECT ID,FirstName,LastName,(Salary+100)\*12 AS "Annual Salary" FROM Employees;

****

**Concatenation**

SELECT CONCAT(FirstName, ' ',LastName) AS "Full Name" FROM Employees;

****

**Relational Operators**

SELECT \* FROM Employees WHERE Salary >= 100000;

****

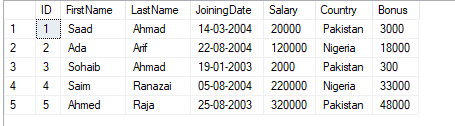
**Distinct**

SELECT DISTINCT Country FROM Employees;

****

**Update**

UPDATE Employees SET Bonus = (Salary \*15/100) WHERE Bonus IS NULL;

****

**QUIZ 1**

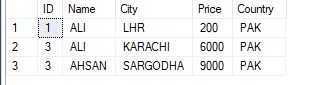
1. Find the customer names that lie in the range of purchasing price of 100 to 5000.

Ans: SELECT Name FROM Transactions WHERE Price<=5000 AND Price >=100;



1. Find the customers that belong to Pakistan.

Ans: SELECT \* FROM Transactions WHERE Country = 'PAK';



1. Search the table for a null city name by writing the query and show the results.

Ans: SELECT \* FROM Transactions WHERE City IS NULL;



1. Write the customer names whose order price is greater than 200.

Ans: SELECT Name FROM Transactions WHERE Price > 200;



1. Find the customer records that belong to Pakistan and their payment is less than 7000.

Ans: SELECT \* FROM Transactions WHERE Country = 'PAK' AND Price < 7000;

