**1Writing Basic**

**SQL SELECT Statements**

**Objectives**

**After completing this lesson, you should be able to do the following:**

• **List the capabilities of SQL SELECT statements** • **Execute a basic SELECT statement**

• **Differentiate between SQL statements and *i*SQL\*Plus commands**

**1-2** Badgujar Dipak D

**Capabilities of SQL SELECT Statements Projection Selection**

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Table 1 Table 1**

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Join**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Table 1 Table 2**

**1-3** Badgujar Dipak D

**Basic SELECT Statement**

**SELECT \*|{[DISTINCT] *column*|*expression* [*alias*],...} FROM *table;*** 

• **SELECT identifies *what* columns**

• **FROM identifies *which* table**

**1-4** Badgujar Dipak D

**Selecting All Columns**

**SELECT \***

**FROM departments; **

**1-5** Badgujar Dipak D

**Selecting Specific Columns**

**SELECT department\_id, location\_id** 

**FROM departments;**

**1-6** Badgujar Dipak D

**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.**

**1-7** Badgujar Dipak D

**Column Heading Defaults**

• ***i*SQL\*Plus:**

– **Default heading justification: Center**

– **Default heading display: Uppercase**

• **SQL\*Plus:**

– **Character and Date column headings are left justified**

– **Number column headings are right-justified** – **Default heading display: Uppercase**

**1-8** Badgujar Dipak D

**Arithmetic Expressions**

**Create expressions with number and date data by using arithmetic operators.**

**Operator Description**

**+ Add**

**- Subtract**

**\* Multiply**

**/ Divide**

**1-9** Badgujar Dipak D

**Using Arithmetic Operators**

**SELECT last\_name, salary, salary + 300** 

**FROM employees;**

**…**

**1-10** Badgujar Dipak D

**Operator Precedence**

**\*/ +\_**

• **Multiplication and division take priority over addition and subtraction.**

• **Operators of the same priority are evaluated from left to right.**

• **Parentheses are used to force prioritized evaluation and to clarify statements.**

**1-11** Badgujar Dipak D

**Operator Precedence**

**SELECT last\_name, salary, 12\*salary+100** 

**FROM employees;**

**…**

**1-12** Badgujar Dipak D

**Using Parentheses**

**SELECT last\_name, salary, 12\*(salary+100) FROM employees;** 

**…**

**1-13** Badgujar Dipak D

**Defining a Null Value**

• **A null is a value that is unavailable, unassigned, unknown, or inapplicable.**

• **A null is not the same as zero or a blank space.**

**SELECT last\_name, job\_id, salary, FROM employees;** 

| **commission\_pct** |
| --- |

**…**

**…**

**1-14** Badgujar Dipak D

**Null Values**

**in Arithmetic Expressions**

**Arithmetic expressions containing a null value evaluate to null.**

**SELECT last\_name, 12\*salary\*commission\_pct FROM employees;** 

**… …**

**1-15** Badgujar Dipak D

**Defining a Column Alias**

**A column alias:**

• **Renames a column heading**

• **Is useful with calculations**

• **Immediately follows the column name - there can also be the optional AS keyword between the column name and alias**

• **Requires double quotation marks if it contains spaces or special characters or is case sensitive**

**1-16** Badgujar Dipak D

**Using Column Aliases**

**SELECT last\_name AS name, commission\_pct comm FROM employees;** 

**… **

**SELECT last\_name "Name", salary\*12 "Annual Salary" FROM employees;** 

**…**

**1-17** Badgujar Dipak D

**Concatenation Operator**

**A concatenation operator:**

• **Concatenates columns or character strings to other columns**

• **Is represented by two vertical bars (||)** • **Creates a resultant column that is a character expression**

**1-18** Badgujar Dipak D

**Using the Concatenation Operator**

**SELECT last\_name||job\_id AS "Employees"** 

**FROM employees;**

**…**

**1-19** Badgujar Dipak D

**Literal Character Strings**

• **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.**

• **Each character string is output once for each row returned.**

**1-20** Badgujar Dipak D

**Using Literal Character Strings**

**SELECT last\_name || ' is a '||job\_id** 

**AS "Employee Details"**

**FROM employees;**

**…**

**1-21** Badgujar Dipak D

**Duplicate Rows**

**The default display of queries is all rows, including duplicate rows.**

**SELECT department\_id** 

**FROM employees;**

**…**

**1-22** Badgujar Dipak D

**Eliminating Duplicate Rows**

**Eliminate duplicate rows by using the DISTINCT keyword in the SELECT clause.**

**SELECT department\_id** 

| **DISTINCT** |
| --- |

**FROM employees;**

**1-23** Badgujar Dipak D

**Displaying Table Structure**

**Use the *i*SQL\*Plus DESCRIBE command to display the structure of a table.**

**DESC[RIBE] *tablename***

**1-29** Badgujar Dipak D

**Displaying Table Structure**

**DESCRIBE employees**

**1-30** Badgujar Dipak D

**Summary**

**In this lesson, you should have learned how to:** • **Write a SELECT statement that:**

– **Returns all rows and columns from a table** – **Returns specified columns from a table**

– **Uses column aliases to give descriptive column headings**

• **Use the *i*SQL\*Plus environment to write, save, and execute SQL statements and *i*SQL\*Plus commands.**

**SELECT \*|{[DISTINCT] *column|expression* [*alias*],...} FROM *table;*** 

**1-34** Badgujar Dipak D

**1**

**Restricting and Sorting Data**

Badgujar Dipak D

**Objectives**

**After completing this lesson, you should be able to do the following:**

• **Limit the rows retrieved by a query**

• **Sort the rows retrieved by a query**

**1-36** Badgujar Dipak D

**Limiting Rows Using a Selection**

**EMPLOYEES**

**… “retrieve all**

**employees**

**in department 90”**

**1-37** Badgujar Dipak D

**Limiting the Rows Selected**

• **Restrict the rows returned by using the WHERE clause.**

**SELECT \*|{[DISTINCT] *column|expression* [*alias*],...} FROM *table***

| **[WHERE *condition(s)*];** |
| --- |

• **The WHERE clause follows the FROM clause.**

**1-38** Badgujar Dipak D

**Using the WHERE Clause**

**SELECT employee\_id, last\_name, job\_id, department\_id FROM employees** 

**WHERE department\_id = 90 ;**

**1-39** Badgujar Dipak D

**Character Strings and Dates**

• **Character strings and date values are enclosed in single quotation marks.**

• **Character values are case sensitive, and date values are format sensitive.**

• **The default date format is DD-MON-RR. SELECT last\_name, job\_id, department\_id FROM employees** 

**WHERE last\_name = 'Whalen' ;**

**1-40** Badgujar Dipak D

**Comparison Conditions**

**Operator Meaning**

**= Equal to**

> **Greaterthan**

**>= <**

**<= <>**

**Greaterthan or equal to Less than**

**Less than or equal to Not equal to**

**1-41** Badgujar Dipak D

**Using Comparison Conditions**

**SELECT last\_name, salary** 

**FROM employees**

**WHERE salary**

| **<= 3000 ;** |
| --- |

**1-42** Badgujar Dipak D

**Other Comparison Conditions**

**Operator Meaning**

**BETWEEN Betweentwo values (inclusive), ...AND...**

**IN(set) Match any of a list of values LIKE Match a characterpattern IS NULL Is a null value**

**1-43** Badgujar Dipak D

**Using the BETWEEN Condition**

**Use the BETWEEN condition to display rows based on a range of values.**

**SELECT last\_name, salary** 

**FROM employees**

**WHERE salary BETWEEN 2500 AND 3500 ;**

**Lower limit Upperlimit**

**1-44** Badgujar Dipak D

**Using the IN Condition**

**Use the IN membership condition to test for values in a list.**

**SELECT employee\_id, last\_name, salary, manager\_id FROM employees** 

**WHERE manager\_id IN (100, 101, 201) ;**

**1-45** Badgujar Dipak D

**Using the LIKE Condition**

• **Use the LIKE condition to perform wildcard searches of valid search string values.** • **Search conditions can contain either literal characters or numbers:**

– **% denotes zero or many characters.**

– **\_ denotes one character.**

**SELECT first\_name** 

**FROM employees**

**WHERE first\_name**

| **LIKE 'S%' ;** |
| --- |

**1-46** Badgujar Dipak D

**Using the LIKE Condition**

• **You can combine pattern-matching characters. SELECT last\_name** 

**FROM employees**

**WHERE last\_name LIKE '\_o%' ;**

****• **You can use the ESCAPE identifier to search for the actual *%* and *\_* symbols.**

**1-47** Badgujar Dipak D

**Using the NULL Conditions**

**Test for nulls with the IS NULL operator.**

**SELECT last\_name, manager\_id** 

**FROM employees**

**WHERE ;**

| **manager\_id IS NULL** |
| --- |

**1-48** Badgujar Dipak D

**Operator AND**

**OR**

**NOT**

**Logical Conditions**

**Meaning**

**Returns TRUEif *both* component conditions are true**

**Returns TRUEif *either* component condition is true**

**Returns TRUEif the following condition is false**

**1-49** Badgujar Dipak D

**Using the AND Operator**

**AND requires both conditions to be true.**

**SELECT employee\_id, last\_name, job\_id, salary FROM employees**

**WHERE**

| **salary >=10000**  **job\_id LIKE '%MAN%'** |
| --- |

**AND;**

**1-50** Badgujar Dipak D

**Using the OR Operator**

**OR requires either condition to be true.** 

**SELECT employee\_id, last\_name, job\_id, salary FROM employees**

**WHERE**

| **salary >= 10000**  **job\_id LIKE '%MAN%'** |
| --- |

**OR;**

**1-51** Badgujar Dipak D

**Using the NOT Operator**

**SELECT last\_name, job\_id** 

**FROM employees**

**WHERE job\_id**

**NOT IN ('IT\_PROG', 'ST\_CLERK', 'SA\_REP'); 1-52** Badgujar Dipak D

**Rules of Precedence**

**Order Evaluated Operator**

**1 Arithmetic operators**

**2 Concatenation operator**

**3 Comparison conditions**

**4 IS [NOT] NULL, LIKE, [NOT] IN 5 [NOT] BETWEEN**

**6 NOT logical condition**

**7 AND logical condition**

**8 OR logical condition**

**Override rules of precedence by using parentheses.**

**1-53** Badgujar Dipak D

**Rules of Precedence**

**SELECT last\_name, job\_id, salary** 

**FROM employees**

**WHERE job\_id = 'SA\_REP'**

**OR job\_id = 'AD\_PRES'** 

**AND salary > 15000;**

**1-54** Badgujar Dipak D

**Rules of Precedence**

**Use parentheses to force priority.**

**SELECT last\_name, job\_id, salary** 

**FROM employees**

**WHERE (job\_id = 'SA\_REP'** 

**OR job\_id = 'AD\_PRES')**

**AND salary > 15000;**

**1-55** Badgujar Dipak D

**ORDER BY Clause**

• **Sort rows with the ORDER BY clause** – **ASC: ascending order, default**

– **DESC: descending order**

• **The ORDER BY clause comes last in the SELECT statement.**

**SELECT last\_name, job\_id, department\_id, hire\_date FROM employees** 

**ORDER BY hire\_date ;**

**…**

**1-56** Badgujar Dipak D

**Sorting in Descending Order**

**SELECT last\_name, job\_id, department\_id, hire\_date FROM employees** 

| **DESC** |
| --- |

**ORDER BY hire\_date ;**

**…**

**1-57** Badgujar Dipak D

**Sorting by Column Alias**

**SELECT employee\_id, last\_name, salary\*12 FROM employees** 

**ORDER BY annsal;**

| **annsal** |
| --- |

**…**

**1-58** Badgujar Dipak D

**Sorting by Multiple Columns**

• **The order of ORDER BY list is the order of sort. SELECT last\_name, department\_id, salary** 

**FROM employees**

**ORDER BY department\_id, salary DESC ;**

**…**

****• **You can sort by a column that is not in the SELECT list.**

**1-59** Badgujar Dipak D

**Summary**

**In this lesson, you should have learned how to:** • **Use the WHERE clause to restrict rows of output**

– **Use the comparison conditions**

– **Use the BETWEEN, IN, LIKE, and NULL conditions** – **Apply the logical AND, OR, and NOT operators** • **Use the ORDER BY clause to sort rows of output**

**SELECT FROM**

**\*|{[DISTINCT] *column|expression* [*alias*],...} *table***

**[WHERE *condition(s)*]** 

**[ORDER BY {*column, expr, alias*} [ASC|DESC]];**

**1-60** Badgujar Dipak D