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 iSQL*Plus commands
Capabilities of SQL SELECT Statements

Basic SELECT Statement

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

- SELECT identifies *what* columns
- FROM identifies *which* table
### Selecting All Columns

**SELECT**

```
SELECT * FROM departments;
```

**Result:**

<table>
<thead>
<tr>
<th>DEPARTMENT_ID</th>
<th>DEPARTMENT_NAME</th>
<th>MANAGER_ID</th>
<th>LOCATION_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>Administration</td>
<td>200</td>
<td>1700</td>
</tr>
<tr>
<td>20</td>
<td>Marketing</td>
<td>201</td>
<td>1800</td>
</tr>
<tr>
<td>50</td>
<td>Shipping</td>
<td>124</td>
<td>1500</td>
</tr>
<tr>
<td>60</td>
<td>IT</td>
<td>113</td>
<td>1400</td>
</tr>
<tr>
<td>90</td>
<td>Sales</td>
<td>148</td>
<td>2500</td>
</tr>
<tr>
<td>99</td>
<td>Executive</td>
<td>100</td>
<td>1700</td>
</tr>
<tr>
<td>110</td>
<td>Accounting</td>
<td>205</td>
<td>1700</td>
</tr>
<tr>
<td>150</td>
<td>Contracting</td>
<td></td>
<td>1700</td>
</tr>
</tbody>
</table>

8 rows selected.

---

### Selecting Specific Columns

**SELECT**

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

**Result:**

<table>
<thead>
<tr>
<th>DEPARTMENT_ID</th>
<th>LOCATION_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>1709</td>
</tr>
<tr>
<td>20</td>
<td>1809</td>
</tr>
<tr>
<td>50</td>
<td>1509</td>
</tr>
<tr>
<td>60</td>
<td>1400</td>
</tr>
<tr>
<td>90</td>
<td>2500</td>
</tr>
<tr>
<td>99</td>
<td>1700</td>
</tr>
<tr>
<td>110</td>
<td>1709</td>
</tr>
<tr>
<td>150</td>
<td>1709</td>
</tr>
</tbody>
</table>

9 rows selected.
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.

Column Heading Defaults

- **iSQL*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
# Arithmetic Expressions

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

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>Add</td>
</tr>
<tr>
<td>-</td>
<td>Subtract</td>
</tr>
<tr>
<td>*</td>
<td>Multiply</td>
</tr>
<tr>
<td>/</td>
<td>Divide</td>
</tr>
</tbody>
</table>

---

## Using Arithmetic Operators

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

<table>
<thead>
<tr>
<th>LAST_NAME</th>
<th>SALARY</th>
<th>SALARY+300</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jones</td>
<td>3000</td>
<td>3300</td>
</tr>
<tr>
<td>Smith</td>
<td>17000</td>
<td>17300</td>
</tr>
<tr>
<td>Davis</td>
<td>17000</td>
<td>17300</td>
</tr>
<tr>
<td>Lillard</td>
<td>9000</td>
<td>9300</td>
</tr>
<tr>
<td>...</td>
<td>1111</td>
<td>1411</td>
</tr>
</tbody>
</table>

---

<DBMS - Lab> <Notes -1>
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.

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

<table>
<thead>
<tr>
<th>LAST_NAME</th>
<th>SALARY</th>
<th>12*SALARY+100</th>
</tr>
</thead>
<tbody>
<tr>
<td>King</td>
<td>7600</td>
<td>91600</td>
</tr>
<tr>
<td>Smiley</td>
<td>114000</td>
<td>136400</td>
</tr>
<tr>
<td>SDM</td>
<td>17000</td>
<td>204100</td>
</tr>
<tr>
<td>T-Hold</td>
<td>9000</td>
<td>109000</td>
</tr>
<tr>
<td>Sw</td>
<td>6000</td>
<td>72000</td>
</tr>
<tr>
<td>Harper</td>
<td>15000</td>
<td>180100</td>
</tr>
<tr>
<td>Pii</td>
<td>3000</td>
<td>36100</td>
</tr>
<tr>
<td>Hargen</td>
<td>17000</td>
<td>194100</td>
</tr>
<tr>
<td>Dr.</td>
<td>31000</td>
<td>370100</td>
</tr>
</tbody>
</table>

---

<DBMS - Lab> <Notes -1>
Using Parentheses

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

<table>
<thead>
<tr>
<th>LAST_NAME</th>
<th>SALARY</th>
<th>12*(SALARY+100)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jan</td>
<td>9000</td>
<td>108000</td>
</tr>
<tr>
<td>Fred</td>
<td>10000</td>
<td>120000</td>
</tr>
<tr>
<td>Dirksen</td>
<td>17000</td>
<td>204000</td>
</tr>
<tr>
<td>Hands</td>
<td>9000</td>
<td>108000</td>
</tr>
<tr>
<td>Ira</td>
<td>1211</td>
<td>14532</td>
</tr>
</tbody>
</table>

(1 row selected)

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, commission_pct
FROM employees;
```

<table>
<thead>
<tr>
<th>LAST_NAME</th>
<th>JOB_ID</th>
<th>SALARY</th>
<th>COMMISSION_PCT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
<td>AD_PRES</td>
<td></td>
<td>24000</td>
</tr>
<tr>
<td>Novak</td>
<td>AD_VP</td>
<td></td>
<td>17000</td>
</tr>
</tbody>
</table>

(2 rows selected)

| Last | DA_481 |          | 2              |
|      | DA_911 |          | 2              |

(2 rows selected)

| Last | AC_ACCOUNT |          | 8000           |

(1 row selected)
Null Values in Arithmetic Expressions

Arithmetic expressions containing a null value evaluate to null.

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

---

<table>
<thead>
<tr>
<th>NAME</th>
<th>12<em>salary</em>commission_pct</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

---

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
Using Column Aliases

```
SELECT last_name AS name, commission_pct AS comm
FROM employees;
```

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

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
Using the Concatenation Operator

```sql
SELECT last_name||job_id AS "Employees"
FROM employees;
```

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.
Using Literal Character Strings

```
SELECT last_name || ' is a ' || job_id AS "Employee Details"
FROM employees;
```

Duplicate Rows

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

```
SELECT department_id
FROM employees;
```
Eliminating Duplicate Rows

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

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

SQL and `iSQL*Plus` Interaction

SQL statements

Oracle server

Internet Browser

`iSQL*Plus`

Query results

`iSQL*Plus` commands

Formatted report

Client
SQL Statements Versus iSQL*Plus Commands

**SQL**
- A language
- ANSI standard
- Keyword cannot be abbreviated
- Statements manipulate data and table definitions in the database

**iSQL*Plus**
- An environment
- Oracle proprietary
- Keywords can be abbreviated
- Commands do not allow manipulation of values in the database
- Runs on a browser
- Centrally loaded, does not have to be implemented on each machine

Overview of iSQL*Plus

After you log into iSQL*Plus, you can:
- Describe the table structure
- Edit your SQL statement
- Execute SQL from iSQL*Plus
- Save SQL statements to files and append SQL statements to files
- Execute statements stored in saved files
- Load commands from a text file into the iSQL*Plus Edit window
Logging In to iSQL*Plus

From your Windows browser environment:

The iSQL*Plus Environment

<DBMS - Lab> <Notes -1>
Displaying Table Structure

Use the \texttt{SQL*Plus DESCRIBE} command to display the structure of a table.

\texttt{DESCRIBE tablename}

\begin{tabular}{|l|l|l|}
\hline
Name & Null? & Type \\
\hline
EMPLOYEE_ID & NOT NULL & NUMBER(30) \\
FIRST_NAME & & VARCHAR2(20) \\
LAST_NAME & NOT NULL & VARCHAR2(25) \\
EMAIL & NOT NULL & VARCHAR2(25) \\
PHONE & NULL & VARCHAR2(15) \\
HIRE_DATE & NOT NULL & DATE \\
JOB & NOT NULL & VARCHAR2 (10) \\
SALARY & NOT NULL & NUMBER(9,2) \\
COMMISSION_PCT & & NUMBER(2,2) \\
MANAGER_ID & & NUMBER(3) \\
DEPARTMENT_ID & & NUMBER(4) \\
\hline
\end{tabular}
Interacting with Script Files

SELECT last_name, hire_date, salary
FROM employees;

Interacting with Script Files

SELECT last_name, hire_date, salary
FROM employees;
Interacting with Script Files

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 `iSQL*Plus` environment to write, save, and execute SQL statements and `iSQL*Plus` commands.

```
SELECT * | ([DISTINCT] column | expression [alias], ...) 
FROM     table;
```
Practice 1 Overview

This practice covers the following topics:

• Selecting all data from different tables
• Describing the structure of tables
• Performing arithmetic calculations and specifying column names
• Using iSQL*Plus