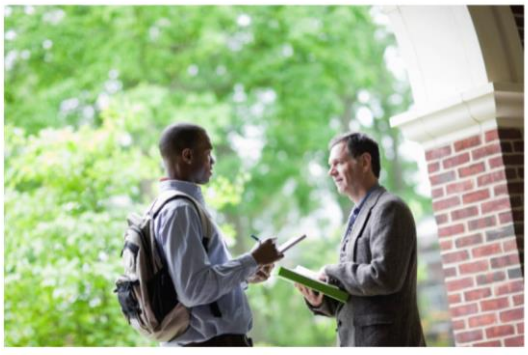

ORACLE
Academy

Database Foundations

6-6

Retrieving Data Using SELECT



ORACLE
Academy

Objectives

- This lesson covers the following objectives:
  - List the capabilities of SQL SELECT statements
  - Write and execute a SELECT statement that:
    - Returns all rows and columns from a table
    - Returns specific columns from a table
    - Uses column aliases to display descriptive column headings

ORACLE Academy

Objectives

- This lesson covers the following objectives:
  - Write and execute a SELECT statement that:
    - Uses arithmetic and concatenation operators
    - Uses literal character strings
    - Eliminates duplicate rows
  - Describe the structure of a table

Basic SELECT Statement

- SELECT identifies the columns to be displayed
- FROM identifies the table that contains those columns

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

ORACLE Academy

DFo 6-6
Retrieving Data Using SELECT

 5

In the syntax:

- SELECT is a list of one or more columns.
- * selects all columns.
- DISTINCT suppresses duplicates.
- column|expression selects the named column or the expression.
- alias gives different headings to the selected columns.
- FROM *table* specifies the table that contains the columns.

Note: Throughout this lesson, the words *keyword, clause,* and *statement* are used as follows:

- A *keyword* refers to an individual SQL element; for example, SELECT and FROM are keywords.
- A *clause* is a part of a SQL statement; for example, SELECT employee_id, last_name.
- A *statement* is a combination of two or more clauses; for example, SELECT * FROM employees.

Selecting All Columns

- All columns of a table can be displayed by placing an * after keyword SELECT

```
SELECT *
FROM   departments;
```

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
10	Administration	200	1700
20	Marketing	201	1800
50	Shipping	124	1500
60	IT	103	1400
80	Sales	149	2500
90	Executive	100	1700
110	Accounting	205	1700

Selecting Specific Columns - Projection

- You can use the SELECT statement to display specific columns of the table by indicating the column names in the order you would like to see them, separated by commas

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

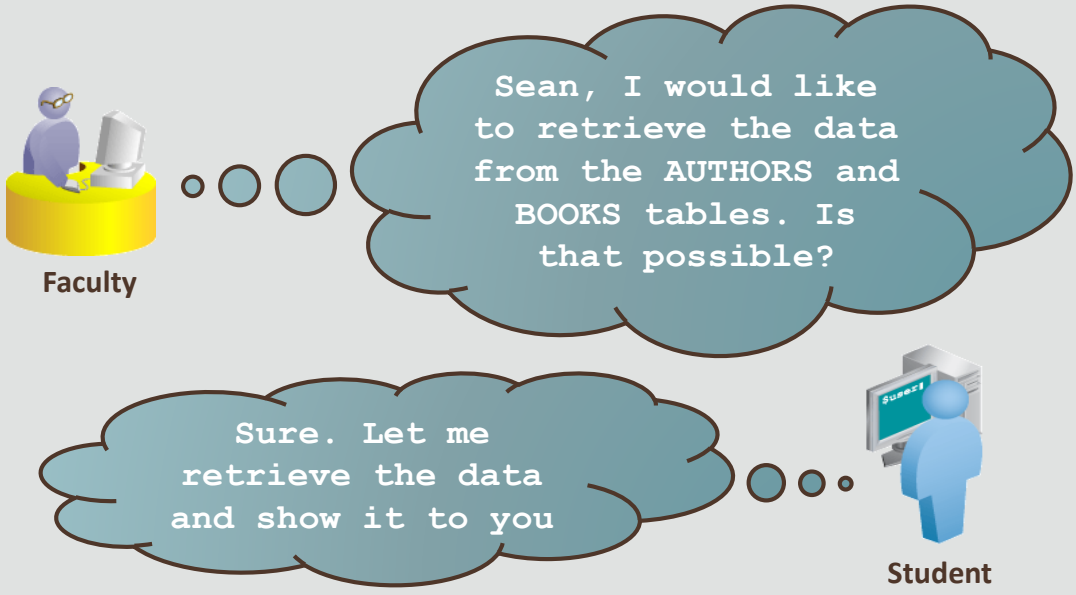
DEPARTMENT_ID	LOCATION_ID
10	1700
20	1800
50	1500
60	1400
80	2500

…

Writing SQL Statements

- By using the simple rules and guidelines shown below, you can construct valid statements that are both easy to read and edit
  - SQL statements are not case-sensitive
  - SQL statements can be entered on one or more lines
  - Keywords cannot be abbreviated or split across lines and are typically spelled with uppercase letters
  - Clauses are usually placed on separate lines
  - Indents are used to enhance readability
  - In Application Express, SQL statements can be terminated by a semicolon (;) but it is not mandatory

Case Scenario: Retrieving Data



ORACLE Academy

Case Scenario: Retrieving Data

```
SELECT id, name
FROM   authors
```

ID	NAME
200	P.G. Wodehouse
300	George Bernard Shaw
100	Leo Tolstoy

Here is the information

```
SELECT *
FROM   books
```

ID	TITLE	PUBLISHER_ID	AUTHOR_ID
3	An Unsocial Socialist	30	300
1	War and Peace	10	100
2	The Clicking of Cuthbert	20	200

ORACLE Academy

Project Exercise 1

- DFo_6_6_1_Project
  - Oracle Baseball League Store Database
  - Write and Execute SELECT statements :
    - Retrieving all columns
    - Selecting specific columns

ORACLE Academy

Arithmetic Expressions

- Create expressions with number and date data by using arithmetic operators
- Column names, numeric constants and arithmetic operators can be used in an arithmetic expression

Operator	Description
+	Add
-	Subtract
*	Multiply
/	Divide

- Arithmetic operators can be used in any clause of a SQL statement except FROM

Note: With the DATE and TIMESTAMP data types, you can only use the addition and subtraction operators.

Using Arithmetic Operators

- Here the addition operator is used to calculate a salary increase of $300 for all employees
- SALARY + 300 is displayed as the column heading

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

LAST_NAME	SALARY	SALARY+300
King	24000	24300
Kochhar	17000	17300
De Haan	17000	17300
Whalen	4400	4700

...

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).

Note: The Oracle server ignores blank spaces before and after the arithmetic operator.

Rules of Precedence

- Multiplication and division are evaluated before addition and subtraction.
- Operators of the same priority are evaluated from left to right.
- Parentheses are used to override the default precedence or to clarify the statement.

Operator Precedence

- Use parentheses to reinforce the standard order of precedence and to improve clarity

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

LAST_NAME	SALARY	12*SALARY+100
King	24000	288100
Kochhar	17000	204100
De Haan	17000	204100
Whalen	4400	52900
Higgins	12000	144100
Gietz	8300	99700

...

The first slide example displays the last name, salary, and annual compensation of employees. It calculates the annual compensation by multiplying the monthly salary by 12, plus a one-time bonus of $100. Multiplication is performed before addition.

The second slide example displays the last name, salary, and annual compensation of employees. It calculates the annual compensation by adding a monthly bonus of $100 to the monthly salary, and then multiplying that subtotal by 12.

Because of the parentheses, addition takes priority over multiplication.

Operator Precedence

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

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

LAST_NAME	SALARY	12*(SALARY+100)
King	24000	289200
Kochhar	17000	205200
De Haan	17000	205200
Whalen	4400	54000
Higgins	12000	145200
Gietz	8300	100800
Zlotkey	10500	127200

…

The first slide example displays the last name, salary, and annual compensation of employees. It calculates the annual compensation by multiplying the monthly salary by 12, plus a one-time bonus of $100. Multiplication is performed before addition.

The second slide example displays the last name, salary, and annual compensation of employees. It calculates the annual compensation by adding a monthly bonus of $100 to the monthly salary, and then multiplying that subtotal by 12.

Because of the parentheses, addition takes priority over multiplication.

Defining a Null Value

- Null is a value that is unavailable, unassigned, unknown, or inapplicable
- Null is not the same as zero or a blank space

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

LAST_NAME	JOB_ID	SALARY	COMMISSION_PCT
King	AD_PRES	24000	-
Gietz	AC_ACCOUNT	8300	-
Zlotkey	SA_MAN	10500	.2
Abel	SA_REP	11000	.3
Taylor	SA_REP	8600	.2
Grant	SA_REP	7000	.15
Mourgos	ST_MAN	5800	-

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

Null Values in Arithmetic Expressions

- Any arithmetic expression containing a null value will evaluate to null

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

LAST_NAME	12*SALARY*COMMISSION_PCT
King	-
Gietz	-
Zlotkey	25200
Abel	39600
Taylor	20640
Grant	12600
Mourgos	-

...

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 the alias.)
  - Requires double quotation marks if it contains spaces or special characters or if it is case-sensitive, the default is all uppercase

Using Column Aliases

- Keyword AS is optional
- Column names appear uppercase by default

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

NAME	COMM
King	-
Kochhar	-
Whalen	-
Higgins	-
…….	

- Column names enclosed in parenthesis will appear as entered

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

Name	Annual Salary
King	288000
Kochhar	204000
Whalen	204000
Higgins	52800
…….	

ORACLE Academy

Note: An alias cannot be referenced in the column list that contains the alias definition. For example, writing a SELECT as follows returns an error:

```
SELECT last_name "Name" , salary*12 "Annual Salary", Annual Salary +100
FROM   employees;
ORA-00904: "ANNUAL SALARY": invalid identifier
```

Concatenation Operator

- Links columns or character strings to other columns
- Is represented by two vertical bars (||)
- Creates a column that is a character expression

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

- Concatenating a NULL with a character results in a character string

Employees
KingAD_PRES
KochharAD_VP
De HaanAD_VP
WhalenAD_ASST

Literal Character Strings

- A literal is a character, a number, or a date that is included in the SELECT statement
- 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

- In the example, the last name and job_id for each employee are concatenated with a literal to give the returned rows more meaning

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

Employee Details
King is a AD_PRES
Kochhar is a AD_VP
De Haan is a AD_VP
Whalen is a AD_ASST
Higgins is a AC_MGR
Gietz is a AC_ACCOUNT

Alternative Quote (q) Operator

- Many SQL statements use character literals in expressions or conditions. If the literal itself contains a single quotation mark, you can use the quote (q) operator and select your own quotation mark delimiter – in this case brackets []

```
SELECT department_name || q'[ Department's Manager Id: ]'
       || manager_id
       AS "Department and Manager"
FROM departments;
```

Department and Manager
Administration Department's Manager Id: 200
Marketing Department's Manager Id: 201
Shipping Department's Manager Id: 124

...

You can choose any convenient delimiter, single-byte or multibyte, or any of the following character pairs: [ ], { }, ( ), or < >.

Case Scenario: Using the Column Alias

Faculty

Sean, I would like to see the different locations where the members are located

I can create a simple query using the SELECT statement and display that information

Student

ORACLE Academy

Case Scenario: Using the SELECT Statement

Here the concatenation operator as well as the column alias has been used

```
SELECT last_name || ' ' || first_name || ' IS LOCATED IN '
       || city AS "Member Location"
FROM   members;
```

Successful retrieval of data

Member Location
Urguhart Molly IS LOCATED IN Quebec
Biri Ben IS LOCATED IN Columbus
Valasquez Carmen IS LOCATED IN Seattle
Menchu Roberta IS LOCATED IN Brussels

ORACLE Academy

Project Exercise 2

- DFo_6_6_2_Project
  - Oracle Baseball League Store Database
  - Write and Execute SELECT statements :
    - Using Arithmetic Operators
    - Using Column Aliases
    - Using Literal Character Strings

ORACLE Academy

Duplicate Rows

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

```
SELECT department_id
FROM   employees;
```

DEPARTMENT_ID
90
90
90
10
110
110
80
80
80

You can specify multiple columns after the DISTINCT qualifier. That qualifier affects all selected columns, and the result displayed is a distinct combination of the columns.

```
SELECT  DISTINCT department_id, job_id
FROM    employees;
```

Note: You can also specify the UNIQUE keyword, which is a synonym for the DISTINCT keyword.

Duplicate Rows

- To eliminate duplicate rows in the result, include the DISTINCT keyword in the SELECT clause immediately after the SELECT keyword

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

DEPARTMENT_ID
-
90
20
110
80
50

...

You can specify multiple columns after the DISTINCT qualifier. That qualifier affects all selected columns, and the result displayed is a distinct combination of the columns.

```
SELECT  DISTINCT department_id, job_id
FROM    employees;
```

Note: You can also specify the UNIQUE keyword, which is a synonym for the DISTINCT keyword.

Displaying the Table Structure

- Use the DESCRIBE command to display the structure of a table including column name, datatype and nullability
- Or, select the APEX Object Browser in SQL Workshop to view table structure
- You can click the Find Tables button in APEX SQL Commands to view table structures as well

```
DESC[RIBE] tablename
```

Using the DESCRIBE Command

```
DESCRIBE employees
```

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
EMPLOYEES	EMPLOYEE_ID	NUMBER	-	6	0	1	-	-	-
	FIRST_NAME	VARCHAR2	30	-	-	-	✓	-	-
	LAST_NAME	VARCHAR2	25	-	-	-	-	-	-
	EMAIL	VARCHAR2	25	-	-	-	-	-	-
	PHONE_NUMBER	VARCHAR2	20	-	-	-	✓	-	-
	HIRE_DATE	DATE	7	-	-	-	-	-	-
	JOB_ID	VARCHAR2	10	-	-	-	-	-	-
	SALARY	NUMBER	-	8	2	-	✓	-	-
	COMMISSION_PCT	NUMBER	-	2	2	-	✓	-	-
	MANAGER_ID	NUMBER	-	6	0	-	✓	-	-
	DEPARTMENT_ID	NUMBER	-	4	0	-	✓	-	-
	BONUS	VARCHAR2	5	-	-	-	✓	-	-

Summary

- In this lesson, you should have learned how to:
  - List the capabilities of SQL SELECT statements
  - Write and execute a SELECT statement that:
    - Returns all rows and columns from a table
    - Returns specific columns from a table
    - Uses column aliases to display descriptive column headings

Summary

- In this lesson, you should have learned how to:
  - Write and execute a SELECT statement that:
    - Uses arithmetic and concatenation operators
    - Uses literal character strings
    - Eliminates duplicate rows
  - Describe the structure of a table


ORACLE
Academy