Chapter 2
Retrieving Data Using the SQL SELECT Statement

Issue No./ Date : 01 / August. 06, 2023
Revision No. & Date : 00 / August. 06, 2023

seed
beyond the obvious

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

seed
beyond the obvious

Basic SELECT Statement

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

- SELECT identifies the columns to be displayed.
- FROM identifies the table containing those columns.

seed
beyond the obvious

Selecting All Columns

```
SELECT * FROM departments;
```

	DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
1	10	Administration	200	1700
2	20	Marketing	201	1800
3	50	Shipping	124	1500
4	60	IT	103	1400
5	80	Sales	149	2500
6	90	Executive	100	1700
7	110	Accounting	205	1700
8	190	Contracting	(null)	1700

seed
beyond the obvious

Selecting Specific Columns

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

	DEPARTMENT_ID	LOCATION_ID
1	10	1700
2	20	1800
3	50	1500
4	60	1400
5	80	2500
6	90	1700
7	110	1700
8	190	1700

seed
beyond the obvious

Writing SQL Statements

- NOT case-sensitive
- Can be entered on one or more lines
  - keywords CANNOT be abbreviated or split across lines
  - clauses usually placed on separate lines
  - indents used to enhance readability
- In SQL Developer - SQL statements can be optionally terminated by a semicolon (;)
  - Semicolons are required when you execute multiple SQL statements
- In SQL*Plus - SQL statement are required to be terminated with a semicolon (;)

seed
beyond the obvious

Column Heading Defaults

- SQL Developer
  - Default heading alignment: Left-aligned
  - Default heading display: Uppercase
- SQL*Plus
  - Character and Date column headings are left-aligned.
  - Number column headings are right-aligned.
  - Default heading display: Uppercase

seed
beyond the obvious

Arithmetic Expressions

- Expressions – created with number and date data using arithmetic operators.

Operator	Description
+	Add
-	Subtract
*	Multiply
/	Divide

seed

Using Arithmetic Operators

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

	LAST_NAME	SALARY	SALARY+300
1	King	24000	24300
2	Kochhar	17000	17300
3	De Haan	17000	17300
4	Hunold	9000	9300
5	Ernst	6000	6300
6	Lorentz	4200	4500
7	Mourgos	5800	6100
8	Rajs	3500	3800
9	Davies	3100	3400
10	Matos	2600	2900

...

Operator Precedence

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

1

	LAST_NAME	SALARY	12*SALARY+100
1	King	24000	288100
2	Kochhar	17000	204100
3	De Haan	17000	204100
4	Hunold	9000	108100

...

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

2

	LAST_NAME	SALARY	12*(SALARY+100)
1	King	24000	289200
2	Kochhar	17000	205200
3	De Haan	17000	205200
4	Hunold	9000	109200

...

What is a NULL Value?

- A value that is **unavailable, unassigned, unknown, or inapplicable**
- It is **NOT** same as zero or a blank space.

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

	LAST_NAME	JOB_ID	SALARY	COMMISSION_PCT
1	King	AD_PRES	24000	(null)
2	Kochhar	AD_VP	17000	(null)
3	De Haan	AD_VP	17000	(null)

...

12	Zlotkey	SA_MAN	10500	0.2
13	Abel	SA_REP	11000	0.3
14	Taylor	SA_REP	8600	0.2
15	Grant	SA_REP	7000	0.15

...

18	Fay	MK_REP	6000	(null)
19	Higgins	AC_MGR	12008	(null)
20	Gietz	AC_ACCOUNT	8300	(null)

seed
beyond the obvious

NULL Values in Arithmetic Expressions

- Arithmetic expressions containing a NULL value **evaluate to NULL.**

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

	LAST_NAME	12*SALARY*COMMISSION_PCT
1	King	(null)
2	Kochhar	(null)
3	De Haan	(null)

...

12	Zlotkey	25200
13	Abel	39600
14	Taylor	20640
15	Grant	12600

...

17	Hartstein	(null)
18	Fay	(null)
19	Higgins	(null)
20	Gietz	(null)

seed
beyond the obvious

Defining a Column Alias

- A column alias:
  - Renames a column heading
  - Is useful with calculations
  - Immediately follows the column name
    - optional `AS` keyword between the column name and the alias can be placed
  - Requires double quotation marks if it contains spaces or special characters, or if it is case-sensitive

seed
beyond the obvious

Using Column Aliases

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

	NAME	COMM
1	King	(null)
2	Kochhar	(null)
3	De Haan	(null)
4	Hunold	(null)

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

	Name	Annual Salary
1	King	288000
2	Kochhar	204000
3	De Haan	204000
4	Hunold	108000

...

seed
beyond the obvious

Concatenation Operator

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

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

	Employees
1	AbelSA_REP
2	DaviesST_CLERK
3	De HaanAD_VP
4	ErnstIT_PROG
5	FayMK_REP
6	GietzAC_ACCOUNT
7	GrantSA_REP
8	HartsteinMK_MAN

...

Literal Character Strings

What is a literal?

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

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

	Employee Details
1	Abel is a SA_REP
2	Davies is a ST_CLERK
3	De Haan is a AD_VP
4	Ernst is a IT_PROG
5	Fay is a MK_REP
6	Gietz is a AC_ACCOUNT
7	Grant is a SA_REP
8	Hartstein is a MK_MAN
9	Higgins is a AC_MGR
10	Hunold is a IT_PROG
11	King is a AD_PRES

...

Alternative Quote (q) Operator

- Specify your own quotation mark delimiter
- Select any delimiter
- Increase readability and usability

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

	Department and Manager
1	Administration Department's Manager Id: 200
2	Marketing Department's Manager Id: 201
3	Shipping Department's Manager Id: 124
4	IT Department's Manager Id: 103
5	Sales Department's Manager Id: 149
6	Executive Department's Manager Id: 100
7	Accounting Department's Manager Id: 205
8	Contracting Department's Manager Id:

Duplicate Rows

Default display of queries - is ALL rows, including duplicate rows

DISTINCT – to eliminate duplicate rows in the result

1

```
SELECT department_id
FROM   employees;
```

	DEPARTMENT_ID
1	90
2	90
3	90
4	60
5	60
6	60
7	50
8	50

...

2

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

	DEPARTMENT_ID
1	(null)
2	90
3	20
4	110
5	50
6	80
7	60
8	10

seed beyond the obvious

Displaying Table Structure

- `DESCRIBE` command – used to display the structure of a table

  OR

- Select the table in the Connections tree and use the Columns tab to view the table structure.

```
DESC[RIBE] tablename
```


seed beyond the obvious

DESCRIBE Command

```
DESCRIBE employees
```

```
DESCRIBE Employees
Name           Null     Type
-------------- -------- ------------
EMPLOYEE_ID    NOT NULL NUMBER(6)
FIRST_NAME              VARCHAR2(20)
LAST_NAME      NOT NULL VARCHAR2(25)
EMAIL          NOT NULL VARCHAR2(25)
PHONE_NUMBER            VARCHAR2(20)
HIRE_DATE      NOT NULL DATE
JOB_ID         NOT NULL VARCHAR2(10)
SALARY                  NUMBER(8,2)
COMMISSION_PCT          NUMBER(2,2)
MANAGER_ID              NUMBER(6)
DEPARTMENT_ID           NUMBER(4)
```


Summary

- Simple `SELECT` statement
- Arithmetic operators
- `NULL` value
- Concatenation operator
- Literals
- `DISTINCT` clause
- `DESCRIBE` command

seed
beyond the obvious