1

Retrieving Data Using the SQL SELECT Statement

Objectives

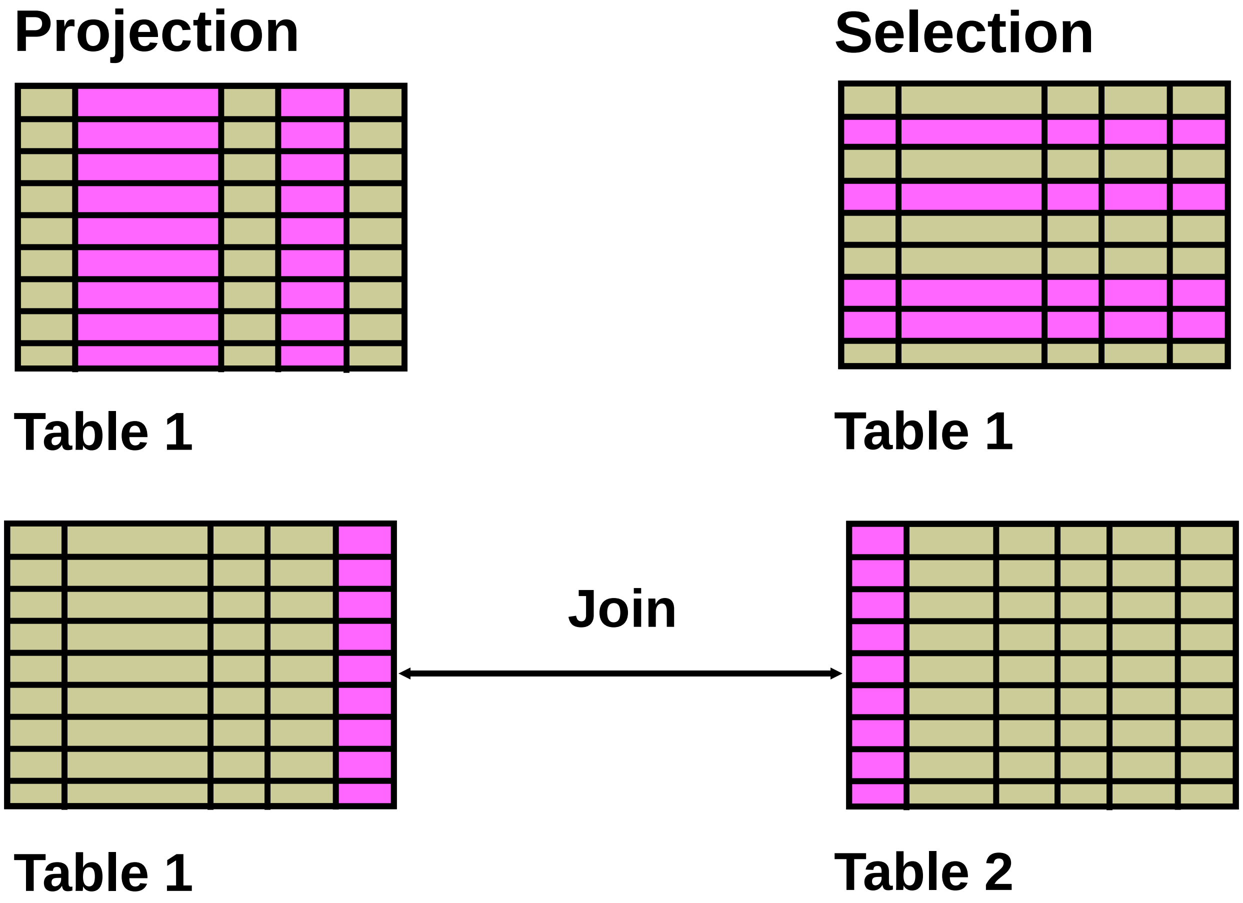
After completing this lesson, you should be able to do the following:

- List the capabilities of SQL `SELECT` statements
- Execute a basic `SELECT` statement

Lesson Agenda

- Basic SELECT statement
- Arithmetic expressions and NULL values in the SELECT statement
- Column aliases
- Use of concatenation operator, literal character strings, alternative quote operator, and the DISTINCT keyword
- DESCRIBE command

Capabilities of SQL `SELECT` Statements



Projection

Table 1

Selection

Table 1

Join

Table 1

Table 2

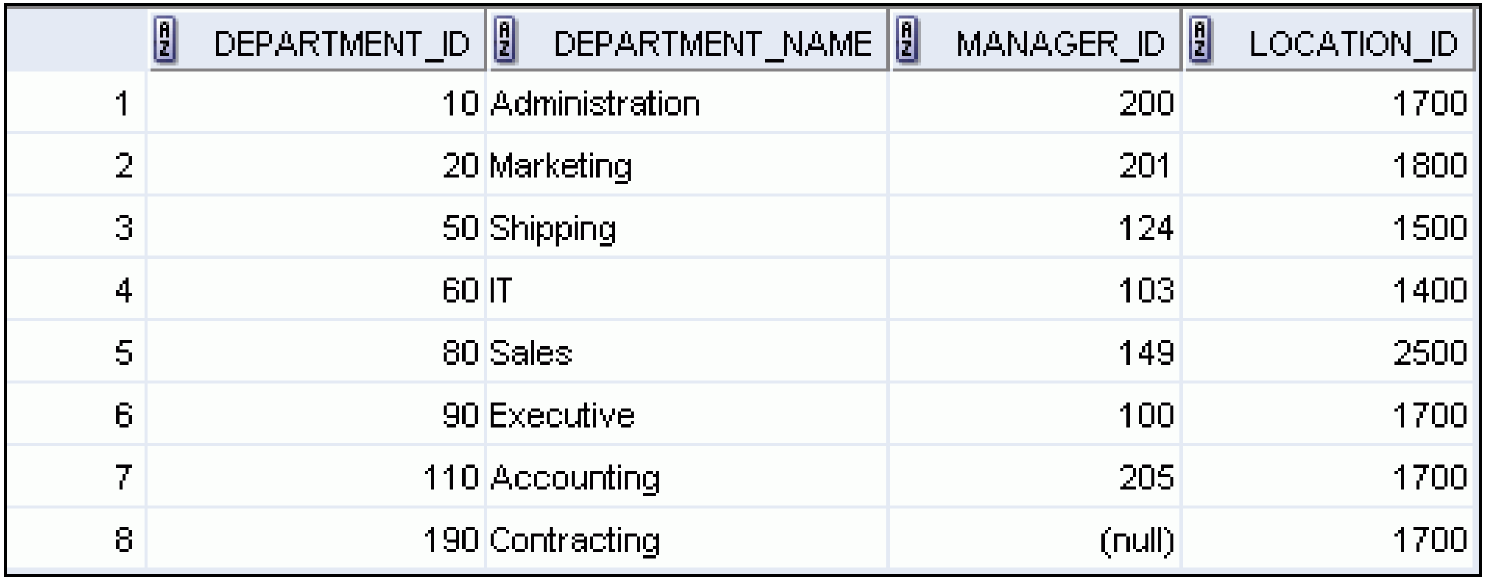
Basic `SELECT` Statement

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

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

Selecting All Columns

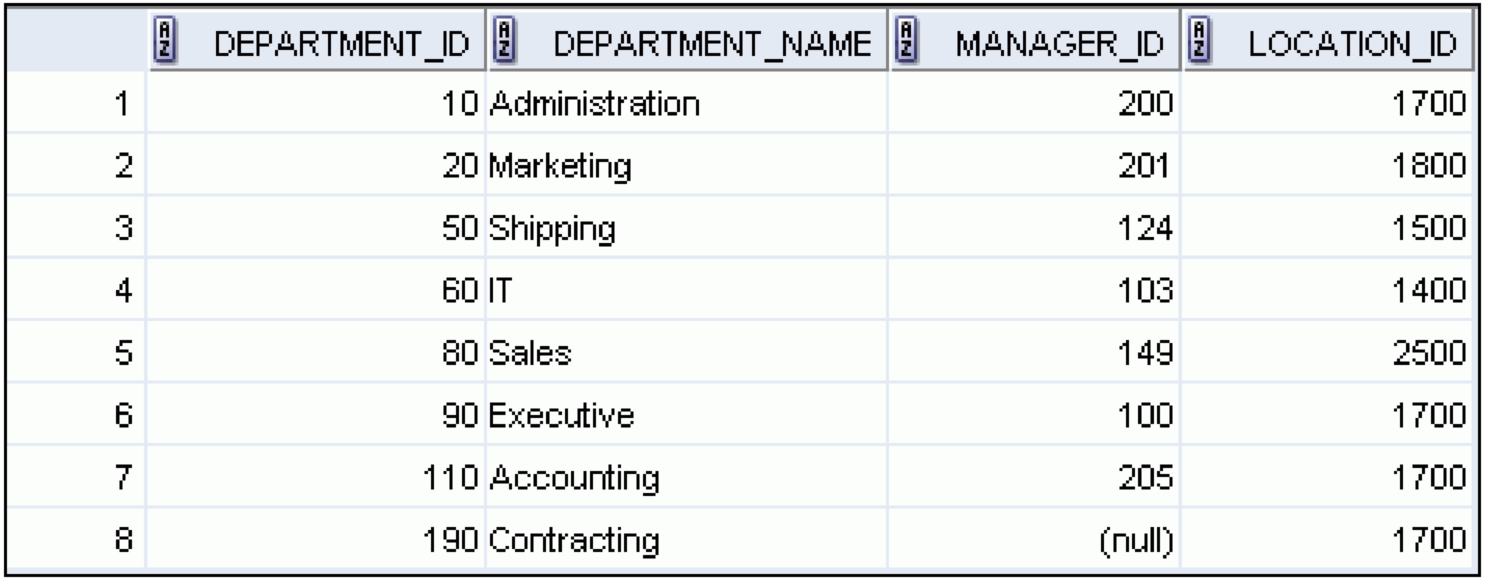
```
SELECT *
FROM   departments;
```



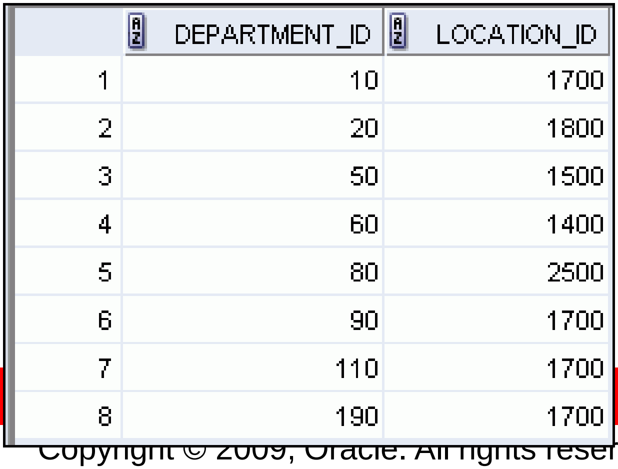
	DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
1	10	Administration	200	1700
2	20	Marketing	201	1800
3	50	Shipping	124	1500
4	60	IT	103	1400
5	80	Sales	149	2500
6	90	Executive	100	1700
7	110	Accounting	205	1700
8	190	Contracting	(null)	1700

Selecting Specific Columns

```
SELECT department_id, location_id
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



	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

Writing SQL Statements

- SQL statements are not case-sensitive.
- SQL statements can be entered 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.
- In many DBMS, SQL statements can optionally be terminated by a semicolon (;) Semicolons are required when you execute multiple SQL statements.

Lesson Agenda

- Basic SELECT statement
- Arithmetic expressions and NULL values in the SELECT statement
- Column Aliases
- Use of concatenation operator, literal character strings, alternative quote operator, and the DISTINCT keyword
- DESCRIBE command

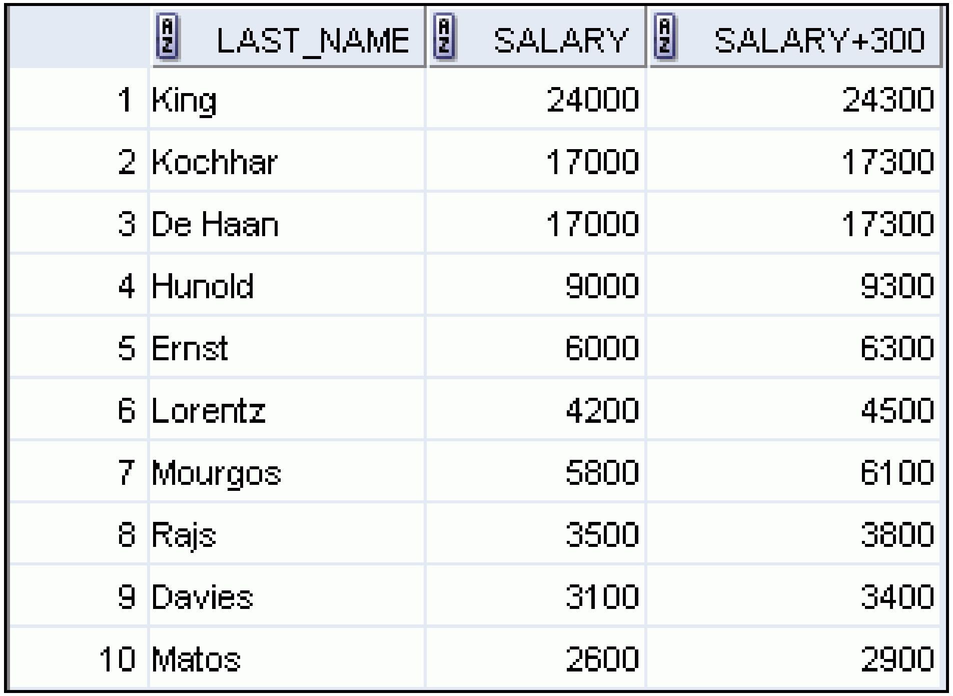
Arithmetic Expressions

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

Operator	Description
+	Add
-	Subtract
*	Multiply
/	Divide

Using Arithmetic Operators

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

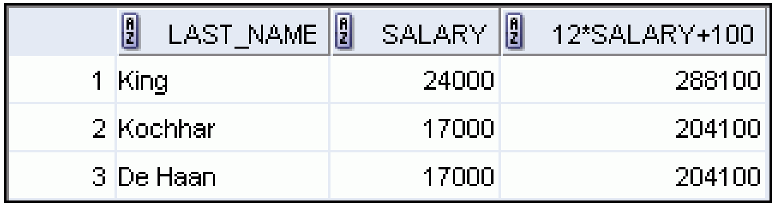


	LAST_NAME	SALARY	SALARY+300
1	King	24000	24300
2	Kochhar	17000	17300
3	De Haan	17000	17300
4	Hunold	9000	9300
5	Ernst	6000	6300
6	Lorentz	4200	4500
7	Mourgos	5800	6100
8	Rajs	3500	3800
9	Davies	3100	3400
10	Matos	2600	2900

Operator Precedence

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

1

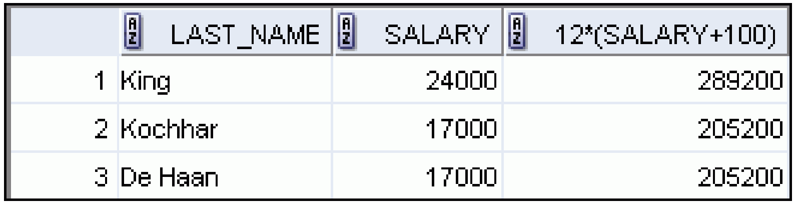


	LAST_NAME	SALARY	12*SALARY+100
1	King	24000	288100
2	Kochhar	17000	204100
3	De Haan	17000	204100

…

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

2



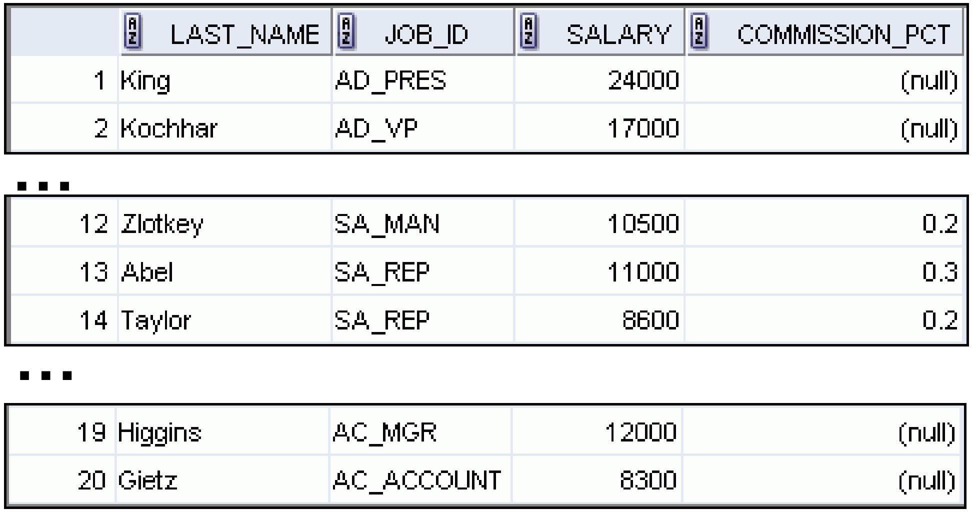
	LAST_NAME	SALARY	12*(SALARY+100)
1	King	24000	289200
2	Kochhar	17000	205200
3	De Haan	17000	205200

…

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
1	King	AD_PRES	24000	(null)
2	Kochhar	AD_VP	17000	(null)

…

	LAST_NAME	JOB_ID	SALARY	COMMISSION_PCT
12	Zlotkey	SA_MAN	10500	0.2
13	Abel	SA_REP	11000	0.3
14	Taylor	SA_REP	8600	0.2

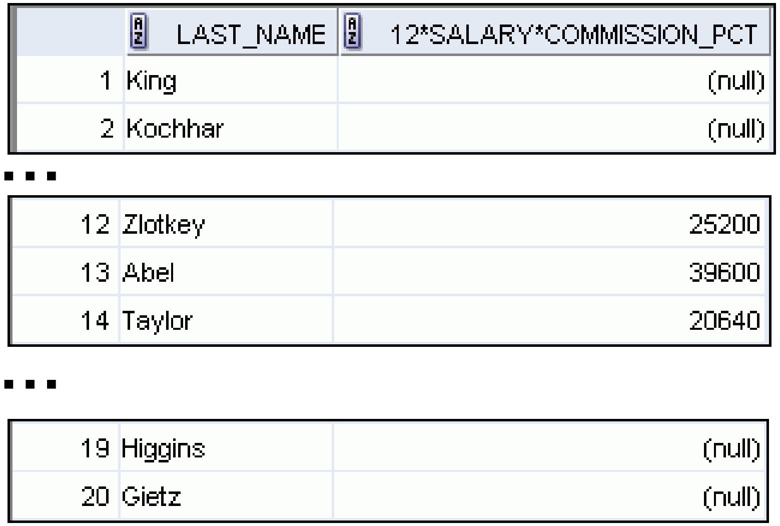
…

	LAST_NAME	JOB_ID	SALARY	COMMISSION_PCT
19	Higgins	AC_MGR	12000	(null)
20	Gietz	AC_ACCOUNT	8300	(null)

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)
...		
12	Zlotkey	25200
13	Abel	39600
14	Taylor	20640
...		
19	Higgins	(null)
20	Gietz	(null)

Lesson Agenda

- Basic SELECT statement
- Arithmetic expressions and NULL values in the SELECT statement
- Column aliases
- Use of concatenation operator, literal character strings, alternative quote operator, and the DISTINCT keyword
- DESCRIBE command

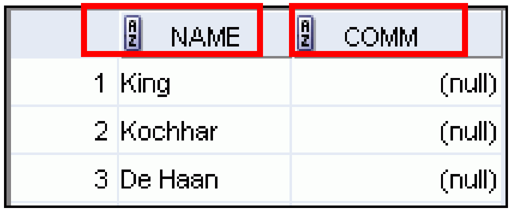
Defining a Column Alias

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 if it is case-sensitive

Using Column Aliases

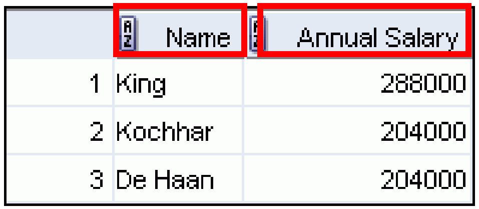
```
SELECT last_name AS name, commission_pct comm
FROM   employees;
```



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

...

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



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

...

Lesson Agenda

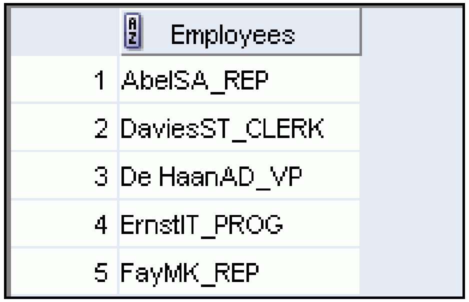
- Basic `SELECT` Statement
- Arithmetic Expressions and NULL values in `SELECT` statement
- Column Aliases
- Use of concatenation operator, literal character strings, alternative quote operator, and the `DISTINCT` keyword
- `DESCRIBE` command

Concatenation Operator in Oracle

A concatenation operator:

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

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



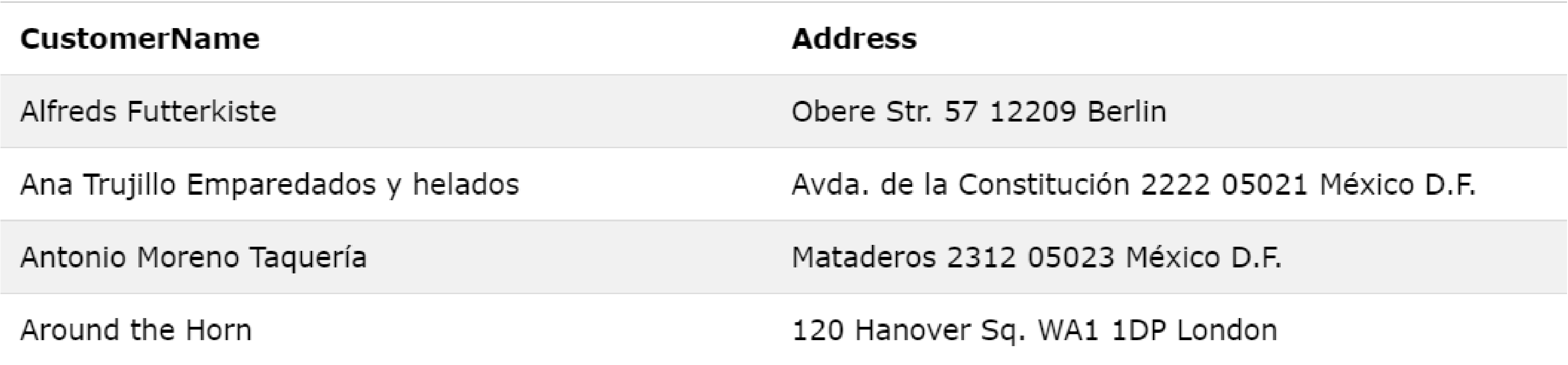
	Employees
1	AbelSA_REP
2	DaviesST_CLERK
3	De HaanAD_VP
4	ErnstIT_PROG
5	FayMK_REP

...

Concatenation Operator in MySQL

Uses concat_ws with a space.

```
SELECT CONCAT_WS(" ", Address, PostalCode,
City) AS Address
FROM Customers;
```



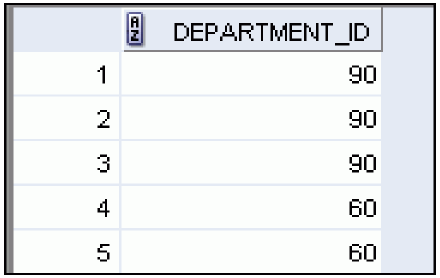
CustomerName	Address
Alfreds Futterkiste	Obere Str. 57 12209 Berlin
Ana Trujillo Emparedados y helados	Avda. de la Constitución 2222 05021 México D.F.
Antonio Moreno Taquería	Mataderos 2312 05023 México D.F.
Around the Horn	120 Hanover Sq. WA1 1DP London

Duplicate Rows

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

```
SELECT department_id
FROM   employees;
```

1

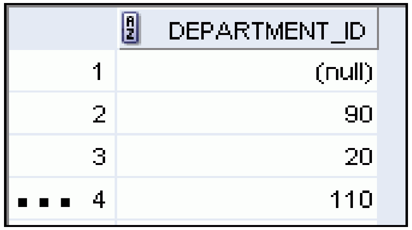


	DEPARTMENT_ID
1	90
2	90
3	90
4	60
5	60

…

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

2



	DEPARTMENT_ID
1	(null)
2	90
3	20
4	110

…

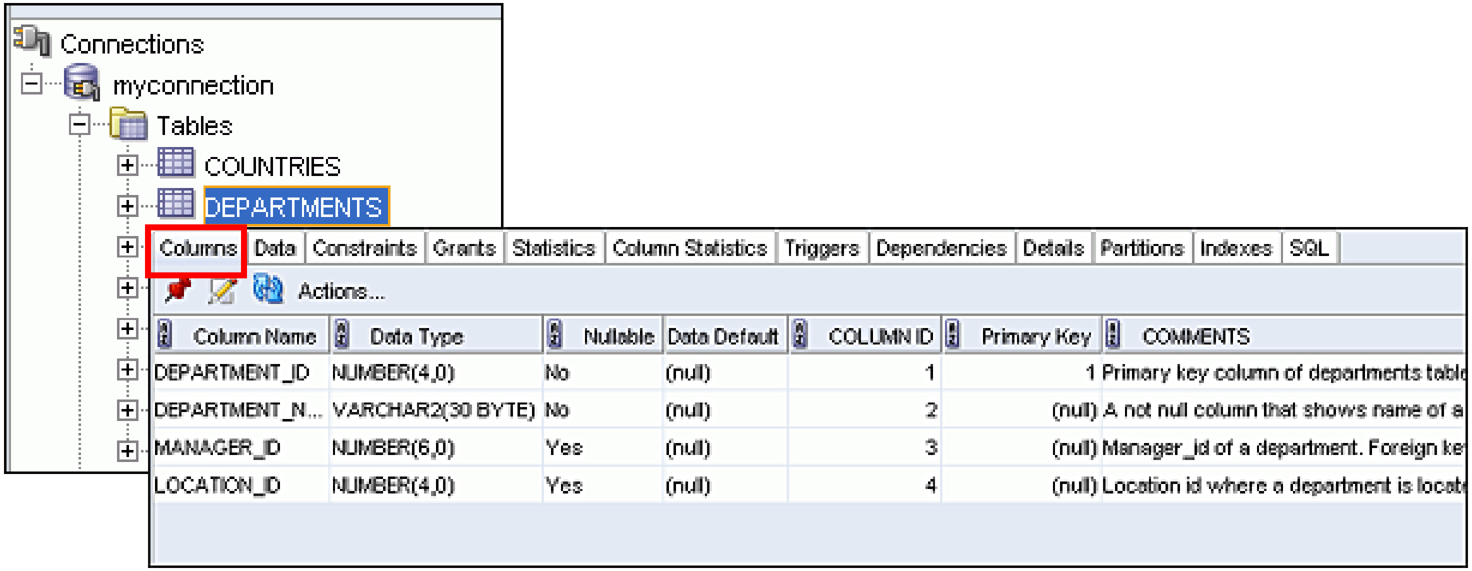
Lesson Agenda

- Basic SELECT statement
- Arithmetic expressions and NULL values in the SELECT statement
- Column aliases
- Use of concatenation operator, literal character strings, alternative quote operator, and the DISTINCT keyword
- DESCRIBE command

Displaying the Table Structure

- Use the `DESCRIBE` command 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.

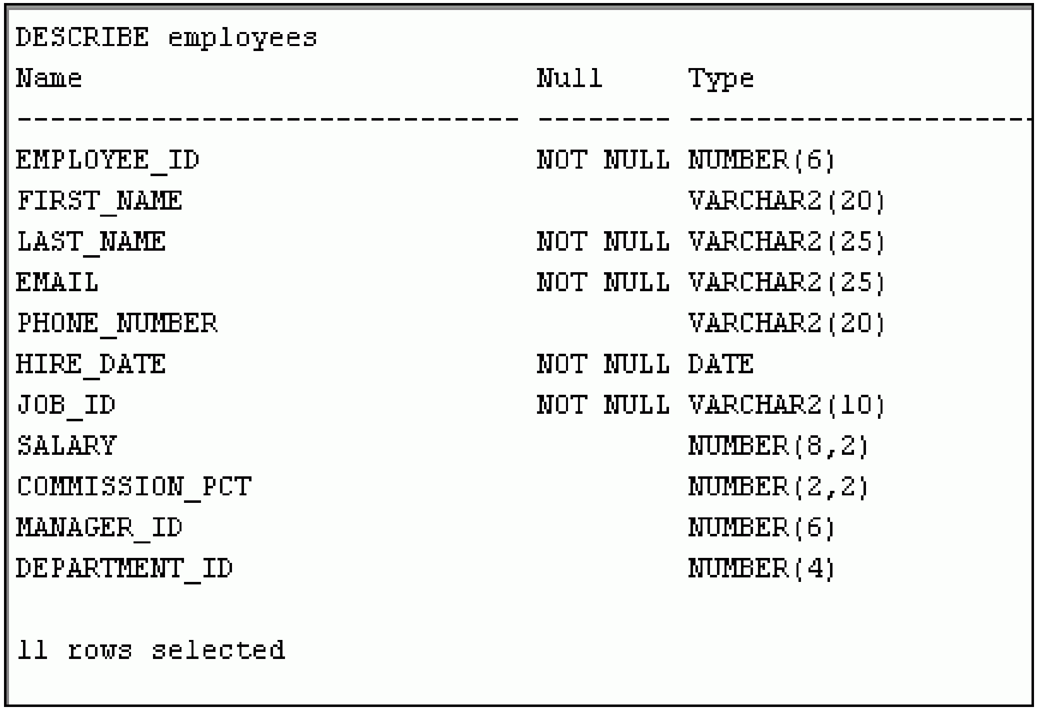
```
DESC[RIBE] tablename
```



Column Name	Data Type	Nullable	Data Default	COLUMN ID	Primary Key	COMMENTS
DEPARTMENT_ID	NUMBER(4,0)	No	(null)	1	1	Primary key column of departments tabl
DEPARTMENT_N...	VARCHAR2(30 BYTE)	No	(null)	2	(null)	A not null column that shows name of a
MANAGER_ID	NUMBER(6,0)	Yes	(null)	3	(null)	Manager_id of a department. Foreign ke
LOCATION_ID	NUMBER(4,0)	Yes	(null)	4	(null)	Location id where a department is locat

Using the `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)

11 rows selected
```

Quiz

Identify the `SELECT` statements that execute successfully.

1.

```
SELECT first_name, last_name, job_id, salary*12
 AS “Yearly Sal”
FROM   employees;
```

2.

```
SELECT first_name, last_name, job_id, salary*12
 yearly sal
FROM   employees;
```

3.

```
SELECT first_name, last_name, job_id, salary AS
 yearly sal
FROM   employees;
```

4.

```
SELECT first_name+last_name AS name, job_Id,
 salary*12 yearly sal
FROM   employees;
```

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 display more descriptive column headings

```
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

W3C SQL Tutorial

Please look at more examples regarding select statements from

https://www.w3schools.com/MySQL/mysql_select.asp