ORACLE

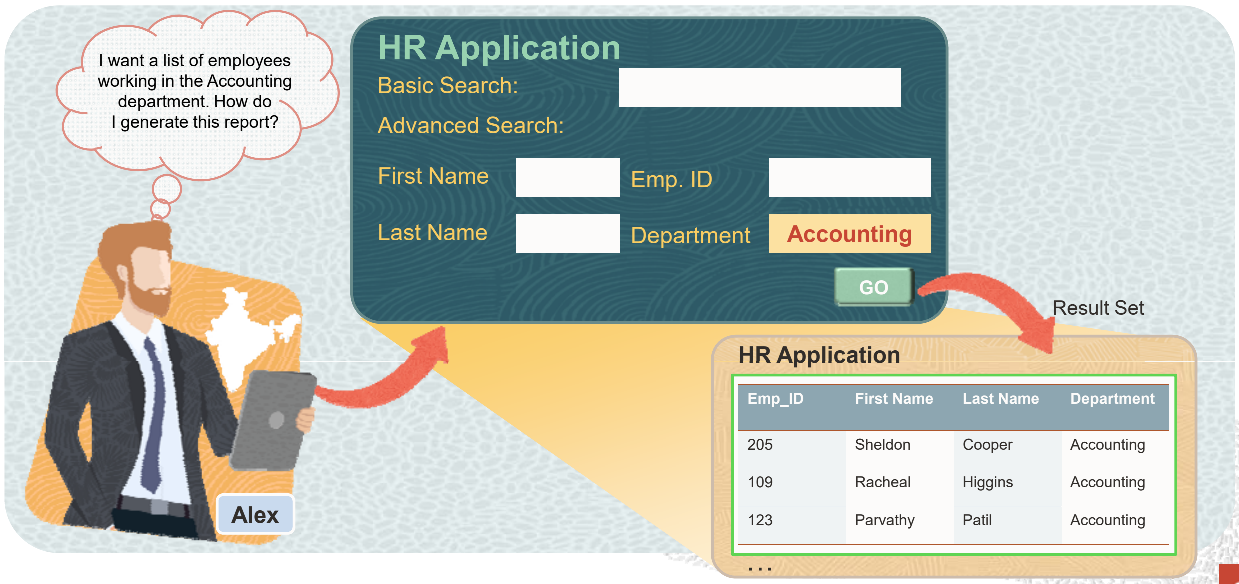
2

Retrieving Data Using the SQL SELECT Statement

Lesson Agenda

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

HR Application Scenario



Emp_ID	First Name	Last Name	Department
205	Sheldon	Cooper	Accounting
109	Racheal	Higgins	Accounting
123	Parvathy	Patil	Accounting

. . .

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.

Basic SELECT Statement

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

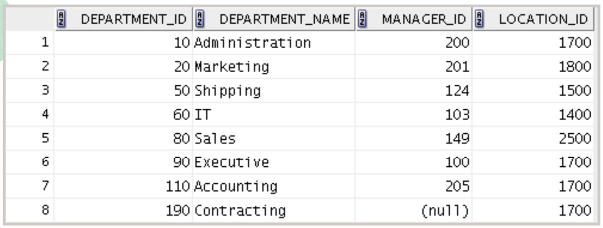
```
SELECT  *|{[DISTINCT] column [alias],...}
FROM    table;
```

Selecting from a table

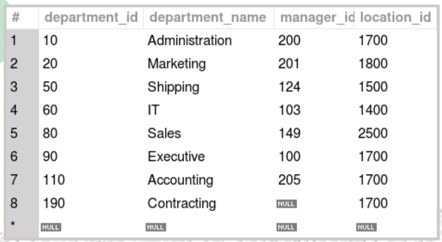
Selecting All Columns

Oracle SQL Developer: MySQL Workbench:

```
SELECT *
FROM   departments;
```



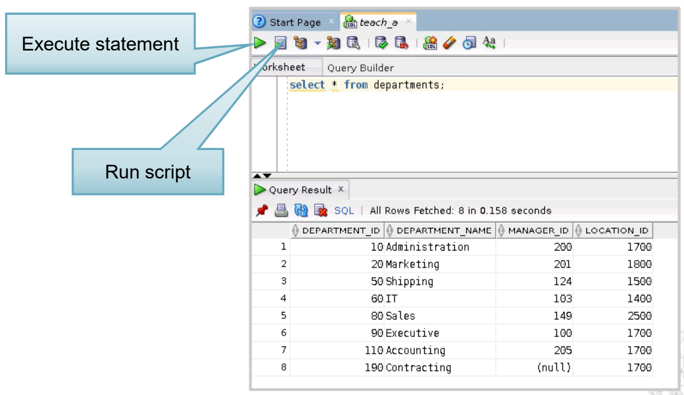
	DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
1	10	Administration	200	1700
2	20	Marketing	201	1800
3	50	Shipping	124	1500
4	60	IT	103	1400
5	80	Sales	149	2500
6	90	Executive	100	1700
7	110	Accounting	205	1700
8	190	Contracting	(null)	1700



#	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
*	NULL	NULL	NULL	NULL

Executing SQL Statements with Oracle SQL Developer and SQL*Plus

SQL Developer

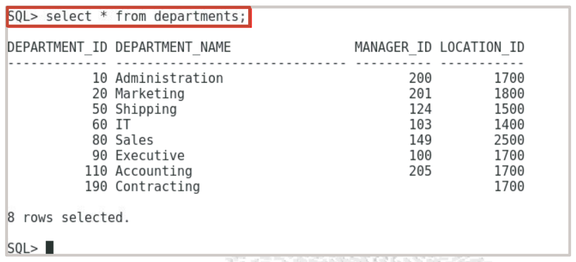


```
select * from departments;
```

All Rows Fetched: 8 in 0.158 seconds

	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

SQL * Plus



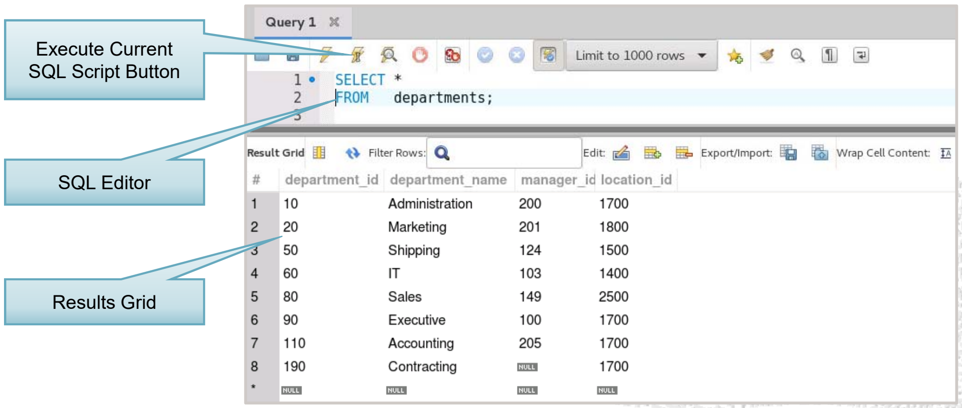
```
SQL> select * from departments;

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
          190 Contracting                                      1700

8 rows selected.

SQL> 
```

Column Heading Defaults in SQL Developer and SQL*Plus

- 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

Executing SQL Statements in MySQL Workbench

Enter statements in the SQL Editor. To execute a single statement, place the cursor anywhere in the statement and click the **Execute Current SQL Script** button or press **Ctrl+Enter**. The results display in the Results Grid.

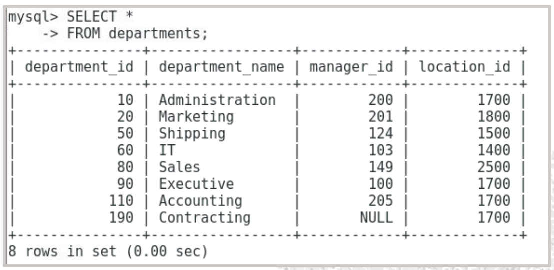


```
SELECT *
FROM   departments;
```

#	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
*	NULL	NULL	NULL	NULL

Executing SQL Statements in mysql Command-line Client

Enter statements in the `mysql` command-line client. Press **Enter** to continue a statement to another line. Terminate a statement with semicolon (`;`) and press **Enter** to execute the statement. Results display in a text table.

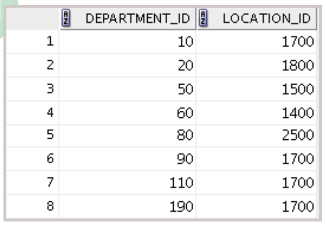


```
mysql> SELECT *
    -> FROM departments;
+---------------+-----------------+------------+-------------+
| department_id | department_name | manager_id | location_id |
+---------------+-----------------+------------+-------------+
|            10 | Administration  |        200 |        1700 |
|            20 | Marketing       |        201 |        1800 |
|            50 | Shipping        |        124 |        1500 |
|            60 | IT              |        103 |        1400 |
|            80 | Sales           |        149 |        2500 |
|            90 | Executive       |        100 |        1700 |
|           110 | Accounting      |        205 |        1700 |
|           190 | Contracting     |       NULL |        1700 |
+---------------+-----------------+------------+-------------+
8 rows in set (0.00 sec)
```

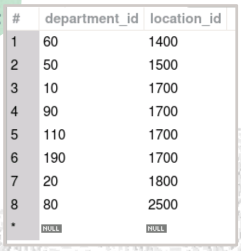
Selecting Specific Columns

Oracle SQL Developer: MySQL Workbench:

```
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

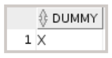


#	department_id	location_id
1	60	1400
2	50	1500
3	10	1700
4	90	1700
5	110	1700
6	190	1700
7	20	1800
8	80	2500
*	NULL	NULL

Selecting from `dual` with Oracle Database

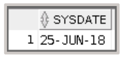
- `dual` is a table automatically created by Oracle Database.
- `dual` has one column called `DUMMY`, of data type `VARCHAR(1)`, and contains one row with a value `X`.

```
SELECT *
FROM   dual;
```



	DUMMY
1	X

```
SELECT SYSDATE
FROM   dual;
```



	SYSDATE
1	25-JUN-18

Selecting Constant Expressions in MySQL

MySQL accepts the `FROM DUAL` clause but ignores it. The following statements are equivalent:

```
SELECT SYSDATE();
```

#	SYSDATE()
1	2018-08-17 18:24:44

```
SELECT SYSDATE()
FROM   DUAL;
```

#	SYSDATE()
1	2018-08-17 18:24:44

Lesson Agenda

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

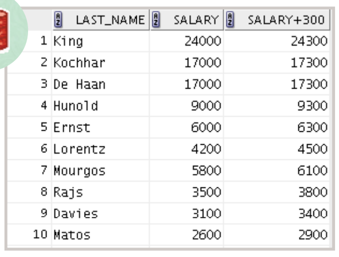
Arithmetic Expressions

You can create expressions with number and date data by using arithmetic operators.

Operator	Description
+	Add
-	Subtract
*	Multiply
/	Divide

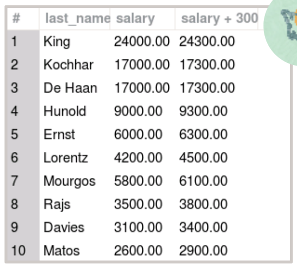
Using Arithmetic Operators

```
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

. . .



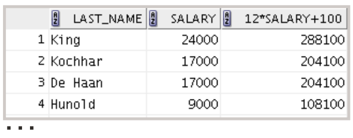
#	last_name	salary	salary + 300
1	King	24000.00	24300.00
2	Kochhar	17000.00	17300.00
3	De Haan	17000.00	17300.00
4	Hunold	9000.00	9300.00
5	Ernst	6000.00	6300.00
6	Lorentz	4200.00	4500.00
7	Mourgos	5800.00	6100.00
8	Rajs	3500.00	3800.00
9	Davies	3100.00	3400.00
10	Matos	2600.00	2900.00

. . .

Operator Precedence

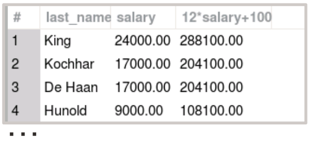
1

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



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

...

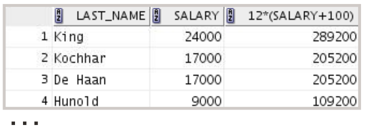


#	last_name	salary	12*salary+100
1	King	24000.00	288100.00
2	Kochhar	17000.00	204100.00
3	De Haan	17000.00	204100.00
4	Hunold	9000.00	108100.00

...

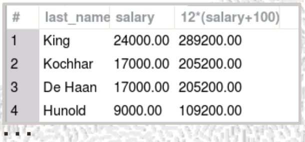
2

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



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

...



#	last_name	salary	12*(salary+100)
1	King	24000.00	289200.00
2	Kochhar	17000.00	205200.00
3	De Haan	17000.00	205200.00
4	Hunold	9000.00	109200.00

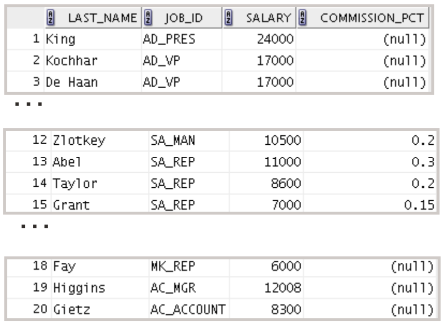
...

17

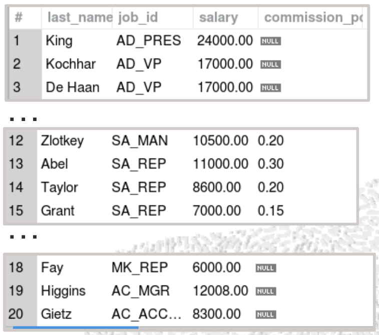
Defining a Null Value

- 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)
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)



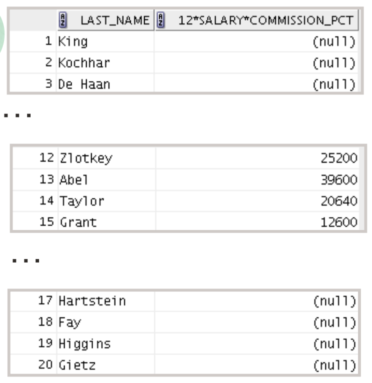
#	last_name	job_id	salary	commission_pc
1	King	AD_PRES	24000.00	NULL
2	Kochhar	AD_VP	17000.00	NULL
3	De Haan	AD_VP	17000.00	NULL
...				
12	Zlotkey	SA_MAN	10500.00	0.20
13	Abel	SA_REP	11000.00	0.30
14	Taylor	SA_REP	8600.00	0.20
15	Grant	SA_REP	7000.00	0.15
...				
18	Fay	MK_REP	6000.00	NULL
19	Higgins	AC_MGR	12008.00	NULL
20	Gietz	AC_ACC...	8300.00	NULL

18

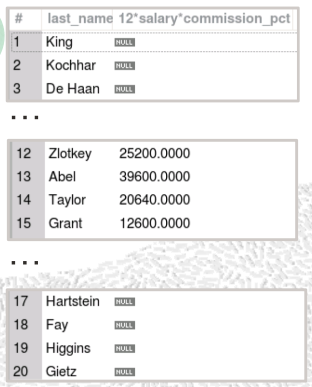
Null Values in Arithmetic Expressions

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)



#	last_name	12*salary*commission_pct
1	King	NULL
2	Kochhar	NULL
3	De Haan	NULL
...		
12	Zlotkey	25200.0000
13	Abel	39600.0000
14	Taylor	20640.0000
15	Grant	12600.0000
...		
17	Hartstein	NULL
18	Fay	NULL
19	Higgins	NULL
20	Gietz	NULL

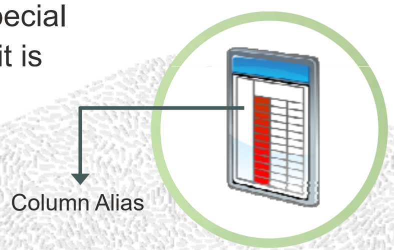
Lesson Agenda

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

Defining a Column Alias

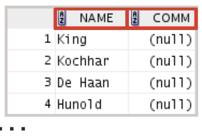
A column alias:

- 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 the alias)
- Requires double quotation marks if it contains spaces or special characters. In Oracle, it requires double quotation marks if it is case-sensitive.



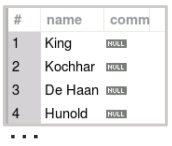
Using Column Aliases

```sql
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)

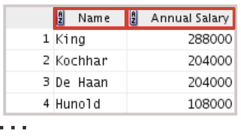
...



#	name	comm
1	King	NULL
2	Kochhar	NULL
3	De Haan	NULL
4	Hunold	NULL

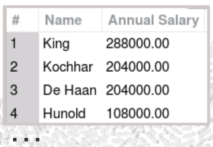
...

```sql
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

...



#	Name	Annual Salary
1	King	288000.00
2	Kochhar	204000.00
3	De Haan	204000.00
4	Hunold	108000.00

...

Lesson Agenda

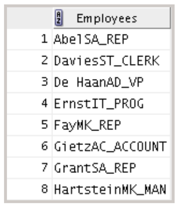
- Capabilities of SQL `SELECT` statements
- Arithmetic expressions and `NULL` values in the `SELECT` statement
- Column aliases
- Use of the concatenation operator, literal character strings, the alternative quote operator, and the `DISTINCT` keyword
- `DESCRIBE` command

Concatenation Operator in Oracle

The 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
6	GietzAC_ACCOUNT
7	GrantSA_REP
8	HartsteinMK_MAN

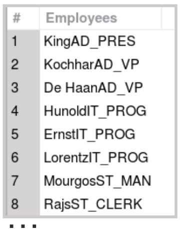
...

Concatenation Function in MySQL – CONCAT( )

The `CONCAT( )`function:

- Links columns or character strings to other columns
- Is a function that concatenates the values provided to it
- Creates a resultant column that is a character expression

```
SELECT    CONCAT(last_name, job_id) AS "Employees"
FROM employees;
```



#	Employees
1	KingAD_PRES
2	KochharAD_VP
3	De HaanAD_VP
4	HunoldIT_PROG
5	ErnstIT_PROG
6	LorentzIT_PROG
7	MourgosST_MAN
8	RajsST_CLERK

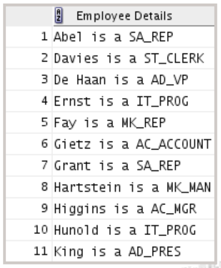
...

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 Oracle

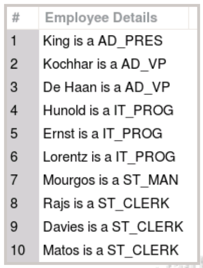
```
SELECT last_name ||' is a '||job_id
       AS "Employee Details"
FROM   employees;
```



	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

Using Literal Character Strings in MySQL

```
SELECT CONCAT(last_name,' is a ', job_id)
       AS 'Employee Details'
FROM   employees;
```



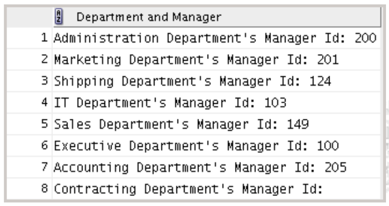
#	Employee Details
1	King is a AD_PRES
2	Kochhar is a AD_VP
3	De Haan is a AD_VP
4	Hunold is a IT_PROG
5	Ernst is a IT_PROG
6	Lorentz is a IT_PROG
7	Mourgos is a ST_MAN
8	Rajs is a ST_CLERK
9	Davies is a ST_CLERK
10	Matos is a ST_CLERK

...

Alternative Quote (q) Operator in Oracle

- 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;
```

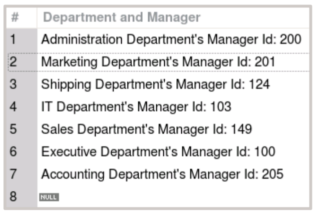


	Department and Manager
1	Administration Department's Manager Id: 200
2	Marketing Department's Manager Id: 201
3	Shipping Department's Manager Id: 124
4	IT Department's Manager Id: 103
5	Sales Department's Manager Id: 149
6	Executive Department's Manager Id: 100
7	Accounting Department's Manager Id: 205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

Including a Single Quotation Mark in a String with an Escape Sequence in MySQL

- To indicate a quotation mark is to be included in a string, use the `\'` escape sequence.

```
SELECT CONCAT(department_name,
  ' Department\'s Manager Id: ',    manager_id)
  AS "Department and Manager"
FROM departments;
```



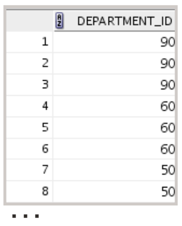
#	Department and Manager
1	Administration Department's Manager Id: 200
2	Marketing Department's Manager Id: 201
3	Shipping Department's Manager Id: 124
4	IT Department's Manager Id: 103
5	Sales Department's Manager Id: 149
6	Executive Department's Manager Id: 100
7	Accounting Department's Manager Id: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

Duplicate Rows

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

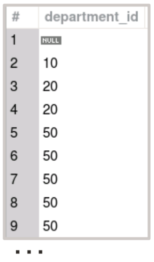
1

```
SELECT department_id
FROM   employees;
```



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

...

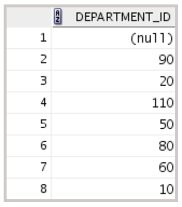


#	department_id
1	NULL
2	10
3	20
4	20
5	50
6	50
7	50
8	50
9	50

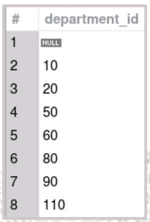
...

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



#	department_id
1	NULL
2	10
3	20
4	50
5	60
6	80
7	90
8	110

Lesson Agenda

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

Displaying Table Structure by Using the DESCRIBE Command

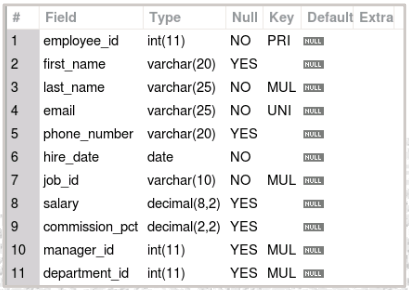
Syntax:

```
DESCRIBE tablename
```

Example:

```
DESCRIBE employees
```

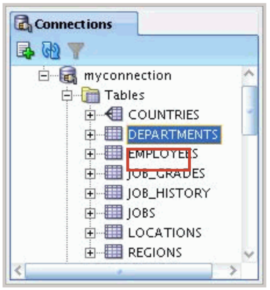
```
DESCRIBE Employees
Name           Null     Type
-------------- -------- ------------
EMPLOYEE_ID    NOT NULL NUMBER(6)
FIRST_NAME              VARCHAR2(20)
LAST_NAME      NOT NULL VARCHAR2(25)
EMAIL          NOT NULL VARCHAR2(25)
PHONE_NUMBER            VARCHAR2(20)
HIRE_DATE      NOT NULL DATE
JOB_ID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

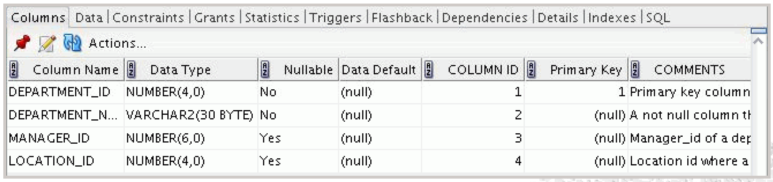


#	Field	Type	Null	Key	Default	Extra
1	employee_id	int(11)	NO	PRI	NULL	
2	first_name	varchar(20)	YES		NULL	
3	last_name	varchar(25)	NO	MUL	NULL	
4	email	varchar(25)	NO	UNI	NULL	
5	phone_number	varchar(20)	YES		NULL	
6	hire_date	date	NO		NULL	
7	job_id	varchar(10)	NO	MUL	NULL	
8	salary	decimal(8,2)	YES		NULL	
9	commission_pct	decimal(2,2)	YES		NULL	
10	manager_id	int(11)	YES	MUL	NULL	
11	department_id	int(11)	YES	MUL	NULL	

Displaying Table Structure by Using Oracle SQL Developer

- Use the DESCRIBE command to display the structure of a table.
- Alternatively, select the table in the Connections tree and use the Columns tab to view the table structure.

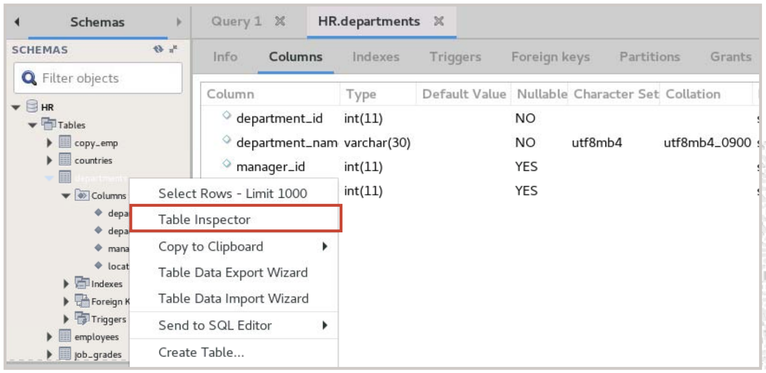



Columns | Data | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Indexes | SQL
Actions...

Column Name	Data Type	Nullable	Data Default	COLUMN ID	Primary Key	COMMENTS
DEPARTMENT_ID	NUMBER(4,0)	No	(null)	1	1	Primary key column
DEPARTMENT_N...	VARCHAR2(30 BYTE)	No	(null)	2	(null)	A not null column th
MANAGER_ID	NUMBER(6,0)	Yes	(null)	3	(null)	Manager_id of a dep
LOCATION_ID	NUMBER(4,0)	Yes	(null)	4	(null)	Location id where a

Displaying Table Structure by Using MySQL Workbench

- Use the `DESCRIBE` command to display the structure of a table.
- Alternatively, right-click the table in the Navigator and select **Table Inspector** from the menu. Select the **Columns** tab.



Column	Type	Default Value	Nullable	Character Set	Collation
department_id	int(11)		NO		
department_nam	varchar(30)		NO	utf8mb4	utf8mb4_0900
manager_id	int(11)		YES		
	int(11)		YES		

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
- Describes the structure of a table