# Database Management Systems

Subject Teacher: Zartasha Baloch

# Restricting and sorting data

Lecture # 13 & 14

**Disclaimer:** The material used in this presentation to deliver the lecture i.e., definitions/text and pictures/graphs etc. does not solely belong to the author/presenter. The presenter has gathered this lecture material from various sources on web/textbooks. Following sources are especially acknowledged:

- 1. Connolly, Thomas M., and Carolyn E. Begg. Database systems: a practical approach to design, implementation, and management. Pearson Education, 2005.
- 2. Gorman, Tim, Inger Jorgensen, Melanie Caffrey, and Lex de Haan. Beginning Oracle SQL: For Oracle Database 12c. Apress, 2014.
- 3. Greenberg, Nancy, and Instructor Guide PriyaNathan. "Introduction to Oracle9i: SQL." ORACLE, USA (2001).

## Objectives

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

- ► Limit the rows retrieved by a query
- Sort the rows retrieved by a query

## Limiting Rows Using a Selection

#### **EMPLOYEES**

EMPLOYEE_ID	LAST_NAME	JOB_ID	DEPARTMENT_ID
100	King	AD_PRES	90
101	Kochhar	AD_VP	90
102	De Haan	AD_VP	90
103	Hunold	IT_PROG	60
104	Ernst	IT_PROG	60
107	Lorentz	IT_PROG	60
124	Mourgos	ST_MAN	50

20 rows selected.

"retrieve all employees in department 90"

EMPLOYEE_ID	LAST_NAME	JOB_ID	DEPARTMENT_ID
100	King	AD_PRES	90
101	Kochhar	AD_VP	90
102	De Haan	AD_VP	90

#### Limiting the Rows Selected

▶ Restrict the rows returned by using the WHERE clause.

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

FROM table
[WHERE condition(s)];
```

#### Using the WHERE Clause

```
SELECT employee_id, last_name, job_id, department_id
FROM employees
WHERE department_id = 90;
```

EMPLOYEE_ID	LAST_NAME	JOB_ID	DEPARTMENT_ID
100	King	AD_PRES	90
101	Kochhar	AD_VP	90
102	De Haan	AD_VP	90

#### Character Strings and Dates

- Character strings and date values are enclosed in single quotation marks.
- Character values are case sensitive, and date values are format sensitive.
- ▶ The default date format is DD-MON-RR.

```
SELECT last_name, job_id, department_id
FROM employees
WHERE last_name = 'Whalen';
```

# Comparison Conditions

Operator	Meaning
=	Equal to
>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to
<b>&lt;&gt;</b>	Not equal to

## Using Comparison Conditions

```
SELECT last_name, salary
FROM employees
WHERE salary <= 3000;</pre>
```

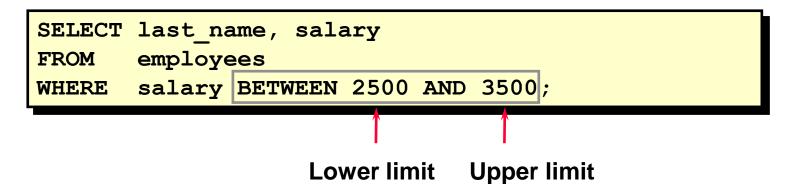
LAST_NAME	SALARY	
Matos	2600	
Vargas	2500	

## Other Comparison Conditions

Operator	Meaning
BETWEENAND	Between two values (inclusive),
IN(set)	Match any of a list of values
LIKE	Match a character pattern
IS NULL	Is a null value

#### Using the BETWEEN Condition

Use the BETWEEN condition to display rows based on a range of values.



LAST_NAME	SALARY	
Rajs	3500	
Davies	3100	
Matos	2600	
Vargas	2500	

#### Using the IN Condition

Use the IN operator to test for values in a list.

```
SELECT employee_id, last_name, salary, manager_id FROM employees
WHERE manager_id IN (100, 101, 201);
```

EMPLOYEE_ID	LAST_NAME	SALARY	MANAGER_ID
202	Fay	6000	201
200	Whalen	4400	101
205	Higgins	12000	101
101	Kochhar	17000	100
102	De Haan	17000	100
124	Mourgos	5800	100
149	Zlotkey	10500	100
201	Hartstein	13000	100

8 rows selected.

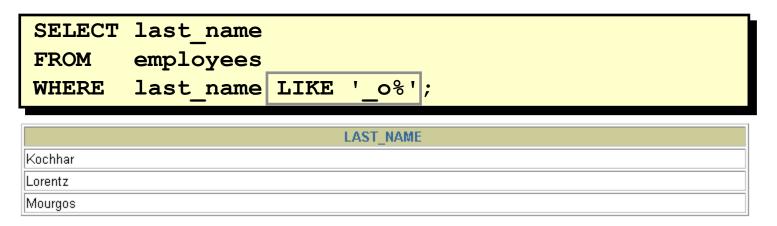
#### Using the LIKE Condition

- ▶ Use the LIKE condition to perform wildcard searches of valid search string values.
- Search conditions can contain either literal characters or numbers:
  - ▶ % denotes zero or many characters.
  - denotes one character.

```
SELECT first_name
FROM employees
WHERE first_name LIKE 'S%';
```

#### Using the LIKE Condition

▶ You can combine pattern-matching characters.



► You can use the ESCAPE identifier to search for the actual % and \_ symbols.

#### Using the NULL Conditions

Test for nulls with the IS NULL operator.

```
SELECT last_name, manager_id
FROM employees
WHERE manager_id IS NULL;
```

LAST_NAME	MANAGER_ID	
King		

# Logical Conditions

Operator	Meaning
AND	Returns TRUE if both component conditions are true
OR	Returns TRUE if either component condition is true
NOT	Returns TRUE if the following condition is false

#### Using the AND Operator

#### AND requires both conditions to be true.

```
SELECT employee_id, last_name, job_id, salary
FROM employees
WHERE salary >=10000
AND job_id LIKE '%MAN%';
```

EMPLOYEE_ID	LAST_NAME	JOB_ID	SALARY
149	Zlotkey	SA_MAN	10500
201	Hartstein	MK_MAN	13000

#### Using the OR Operator

#### OR requires either conditions to be true.

```
SELECT employee_id, last_name, job_id, salary
FROM employees
WHERE salary >= 10000
OR job_id LIKE '%MAN%';
```

EMPLOYEE_ID	LAST_NAME	JOB_ID	SALARY
100	King	AD_PRES	24000
101	Kochhar	AD_VP	17000
102	De Haan	AD_VP	17000
124	Mourgos	ST_MAN	5800
149	Zlotkey	SA_MAN	10500
174	Abel	SA_REP	11000
201	Hartstein	MK_MAN	13000
205	Higgins	AC_MGR	12000

8 rows selected.

## Using the NOT Operator

```
SELECT last_name, job_id
FROM employees
WHERE job_id
NOT IN ('IT_PROG', 'ST_CLERK', 'SA_REP');
```

LAST_NAME	JOB_ID
King	AD_PRES
Kochhar	AD_VP
De Haan	AD_VP
Mourgos	ST_MAN
Zlotkey	SA_MAN
Whalen	AD_ASST
Hartstein	MK_MAN
Fay	MK_REP
Higgins	AC_MGR
Gietz	AC_ACCOUNT

10 rows selected.

#### Rules of Precedence

Order Evaluated	Operator
1	Arithmetic operators
2	Concatenation operator
3	Comparison conditions
4	IS [NOT] NULL, LIKE, [NOT] IN
5	[NOT] BETWEEN
6	NOT logical condition
7	AND logical condition
8	OR logical condition

Override rules of precedence by using parentheses.

#### Rules of Precedence

```
SELECT last_name, job_id, salary

FROM employees

WHERE job_id = 'SA_REP'

OR job_id = 'AD_PRES'

AND salary > 15000;
```

LAST_NAME	JOB_ID	SALARY
King	AD_PRES	24000
Abel	SA_REP	11000
Taylor	SA_REP	8600
Grant	SA_REP	7000

#### Rules of Precedence

#### Use parenthesis to force priority.

```
SELECT last_name, job_id, salary

FROM employees

WHERE (job_id = 'SA_REP'

OR job_id = 'AD_PRES')

AND salary > 15000;
```

	LAST_NAME	JOB_ID	SALARY
King		AD_PRES	24000

#### ORDER BY Clause

- ▶ Sort rows with the ORDER BY clause
  - ► ASC: ascending order, default
  - ▶ DESC: descending order
- ▶ The ORDER BY clause comes last in the SELECT statement.

```
SELECT last_name, job_id, department_id, hire_date
FROM employees
ORDER BY hire_date;
```

LAST_NAME	JOB_ID	DEPARTMENT_ID	HIRE_DATE
King	AD_PRES	90	17-JUN-87
Whalen	AD_ASST	10	17-SEP-87
Kochhar	AD_VP	90	21-SEP-89
Hunold	IT_PROG	60	03-JAN-90
Ernst	IT_PROG	60	21-MAY-91

. . .

## Sorting in Descending Order

```
SELECT last_name, job_id, department_id, hire_date
FROM employees
ORDER BY hire_date DESC;
```

LAST_NAME	JOB_ID	DEPARTMENT_ID	HIRE_DATE
Zlotkey	SA_MAN	80	29-JAN-00
Mourgos	ST_MAN	50	16-NOV-99
Grant	SA_REP		24-MAY-99
Lorentz	IT_PROG	60	07-FEB-99
Vargas	ST_CLERK	50	09-JUL-98
Taylor	SA_REP	80	24-MAR-98
Matos	ST_CLERK	50	15-MAR-98
Fay	MK_REP	20	17-AUG-97
Davies	ST_CLERK	50	29-JAN-97

. . .

20 rows selected.

## Sorting by Column Alias

```
SELECT employee_id, last_name, salary*12 annsal FROM employees
ORDER BY annsal;
```

EMPLOYEE_ID	LAST_NAME	ANNSAL
144	Vargas	30000
143	Matos	31200
142	Davies	37200
141	Rajs	42000
107	Lorentz	50400
200	Whalen	52800
124	Mourgos	69600
104	Ernst	72000
202	Fay	72000
178	Grant	84000

- - -

20 rows selected.

#### Sorting by Multiple Columns

The order of ORDER BY list is the order of sort.

```
SELECT last_name, department_id, salary
FROM employees
ORDER BY department_id, salary DESC;
```

LAST_NAME	DEPARTMENT_ID	SALARY
Whalen	10	4400
Hartstein	20	13000
Fay	20	6000
Mourgos	50	5800
Rajs	50	3500
Davies	50	3100
Matos	50	2600
Vargas	50	2500

- -

20 rows selected.

You can sort by a column that is not in the SELECT list.

#### Summary

#### In this lesson, you should have learned how to:

- Use the WHERE clause to restrict rows of output
  - Use the comparison conditions
  - Use the BETWEEN, IN, LIKE, and NULL conditions
  - Apply the logical AND, OR, and NOT operators
- Use the ORDER BY clause to sort rows of output

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

FROM table

[WHERE condition(s)]

[ORDER BY {column, expr, alias} [ASC|DESC]];
```