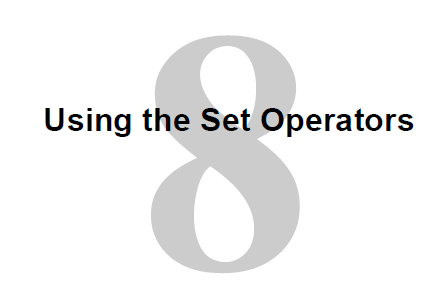
Les05 - Set Operators



Objectives

# 1 Describe Set Operators

# 2 Use set operators to combine multiple queries into a single query

# 3 Control order of rows returned

Topics Covered

Set Operator Types and rules

Tables Used in the lesson 🡪 Employee and Job\_history

UNION

UNION ALL

INTERSECT

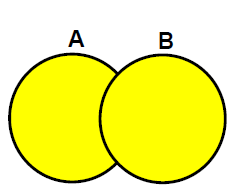
MINUS

Matching SELECT statements

ORDER BY with set operators

Types - Union

UNION



UNION of all the rows in A

With ALL the rows in B

With NO DUPLICATES

RESULT is the Yellow- but duplicates not showing twice

EXAMPLE:

JOB\_HISTORY 🡺Table keeps history of when an employee changes jobs

Records start date and end date of employees that switch jobs

Employees who are still in the same job will not appear here

The current job is shown in the EMPLOYEE table. Again, this shows history.

SELECT \* FROM job\_history;

EMPLOYEE\_ID START\_DATE END\_DATE JOB\_ID DEPARTMENT\_ID ------------------ ------------------------- ------------------ ---------- ---

102 13-JAN-93 24-JUL-98 IT\_PROG 60

101 21-SEP-89 27-OCT-93 AC\_ACCOUNT 110

101 28-OCT-93 15-MAR-97 AC\_MGR 110

201 17-FEB-96 19-DEC-99 MK\_REP 20

114 24-MAR-98 31-DEC-99 ST\_CLERK 50

122 01-JAN-99 31-DEC-99 ST\_CLERK 50

200 17-SEP-87 17-JUN-93 AD\_ASST 90

176 24-MAR-98 31-DEC-98 SA\_REP 80

176 01-JAN-99 31-DEC-99 SA\_MAN 80

200 01-JUL-94 31-DEC-98 AC\_ACCOUNT 90

10 rows selected

EMPLOYEE table contains employee information.

This example, only show the employee\_id, job\_id, department\_id 🡺 the common attributes.

SELECT employee\_id, job\_id, department\_id

FROM employee;

EMPLOYEE\_ID JOB\_ID DEPARTMENT\_ID

----------- ---------- -------------

100 AD\_PRES 90

101 AD\_VP 90

102 AD\_VP 90

103 IT\_PROG 60

104 IT\_PROG 60

107 IT\_PROG 60

124 ST\_MAN 50

141 ST\_CLERK 50

142 ST\_CLERK 50

143 ST\_CLERK 50

144 ST\_CLERK 50

149 SA\_MAN 80

174 SA\_REP 80

176 SA\_REP 80

178 SA\_REP

200 AD\_ASST 10

201 MK\_MAN 20

202 MK\_REP 20

205 AC\_MGR 110

206 AC\_ACCOUNT 110

20 rows selected.

Result of the UNION of both tables using just employee\_id and job\_id

**SELECT employee\_id, job\_id**

**FROM employee**

**UNION**

**SELECT employee\_id, job\_id**

**FROM job\_history**

**ORDER BY job\_id; 🡸 added order by for readability**

EMPLOYEE\_ID JOB\_ID

---------------------- ----------

100 AD\_PRES

101 AC\_ACCOUNT

101 AC\_MGR

101 AD\_VP

102 AD\_VP

102 IT\_PROG

103 IT\_PROG

104 IT\_PROG

107 IT\_PROG

114 ST\_CLERK

122 ST\_CLERK

124 ST\_MAN

141 ST\_CLERK

142 ST\_CLERK

143 ST\_CLERK

144 ST\_CLERK

149 SA\_MAN

174 SA\_REP

176 SA\_MAN

176 SA\_REP

178 SA\_REP

200 AC\_ACCOUNT

200 AD\_ASST

201 MK\_MAN

201 MK\_REP

202 MK\_REP

205 AC\_MGR

206 AC\_ACCOUNT

28 rows selected 🡸 Notice the 28 rows

Employee 20

Job\_history 10

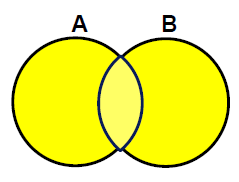
TOTAL 30 rows

Since only produced 28 rows, then 2 rows must be duplicates and not shown.

WHERE ARE THE DUPLICATES ?

Types – Union All

8-4



UNION of ALL the rows in A and B including duplicates

RESULT is all the Yellow and duplicates showing twice

ENTER the code using the same 2 tables to see the result.

SELECT employee\_id, job\_id

FROM employee

UNION ALL

SELECT employee\_id, job\_id

FROM job\_history

ORDER BY employee\_id;

EMPLOYEE\_ID JOB\_ID

---------------------- ----------

100 AD\_PRES

101 AD\_VP

101 AC\_ACCOUNT

101 AC\_MGR

102 IT\_PROG

102 AD\_VP

103 IT\_PROG

104 IT\_PROG

107 IT\_PROG

114 ST\_CLERK

122 ST\_CLERK

124 ST\_MAN

141 ST\_CLERK

142 ST\_CLERK

143 ST\_CLERK

144 ST\_CLERK

149 SA\_MAN

174 SA\_REP

176 SA\_REP Was a Sales Representative

176 SA\_MAN Became a Sales Manager

176 SA\_REP Went back to a Sales Rep

178 SA\_REP

200 AD\_ASST Looks like the same here

200 AD\_ASST

200 AC\_ACCOUNT

201 MK\_REP

201 MK\_MAN

202 MK\_REP

205 AC\_MGR

206 AC\_ACCOUNT

30 rows selected

Change the code and add in DEPARTMENT\_ID

**SELECT employee\_id, job\_id, department\_id**

**FROM employee**

**UNION**

**SELECT employee\_id, job\_id, department\_id**

**FROM job\_history**

**ORDER BY employee\_id;**

What was the result?

How many duplicates, if any?

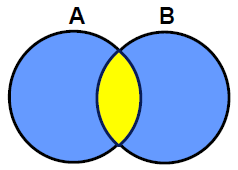
Why?

ANSWER:

Look at the former duplicate 200 – notice different department\_id

Types – Intersect

INTERSECT



The rows in common to both tables only

A intersect B

same as

B intersect A

RESULT is the Yellow

Change the previous SQL to an INTERSECT

**- finds the common rows (or duplicates)**

**SELECT employee\_id, job\_id, department\_id**

**FROM employee**

**INTERSECT**

**SELECT employee\_id, job\_id, department\_id**

**FROM job\_history**

**ORDER BY employee\_id;**

EMPLOYEE\_ID JOB\_ID DEPARTMENT\_ID

----------- ---------- -------------

176 SA\_REP 80

Do the same but without department\_id

**SELECT employee\_id, job\_id**

**FROM employee**

**INTERSECT**

**SELECT employee\_id, job\_id**

**FROM job\_history;**

EMPLOYEE\_ID JOB\_ID

----------- ----------

176 SA\_REP

200 AD\_ASST

What did this tell you?

SELECT \* FROM JOB\_HISTORY;

EMPLOYEE\_ID START\_DAT END\_DATE JOB\_ID DEPARTMENT\_ID

----------- --------- --------- ---------- -------------

102 13-JAN-93 24-JUL-98 IT\_PROG 60

101 21-SEP-89 27-OCT-93 AC\_ACCOUNT 110

101 28-OCT-93 15-MAR-97 AC\_MGR 110

201 17-FEB-96 19-DEC-99 MK\_REP 20

114 24-MAR-98 31-DEC-99 ST\_CLERK 50

122 01-JAN-99 31-DEC-99 ST\_CLERK 50

200 17-SEP-87 17-JUN-93 AD\_ASST 90

176 24-MAR-98 31-DEC-98 SA\_REP 80

176 01-JAN-99 31-DEC-99 SA\_MAN 80

200 01-JUL-94 31-DEC-98 AC\_ACCOUNT 90

10 rows selected.

TITLES and ORDER BY

SELECT employee\_id as "Emp#", job\_id as "Job Title"

FROM employee

UNION ALL

SELECT employee\_id, job\_id

FROM job\_history

ORDER BY 1, 2

Emp# Job Title

---------- ----------

100 AD\_PRES

101 AC\_ACCOUNT

101 AC\_MGR …. Etc for 30 rows

# What if use 3 columns in table 1 and 2 in table 2?

SELECT employee\_id as "Emp#", job\_id as "Job Title", department\_id

FROM employee

UNION ALL

SELECT employee\_id, job\_id

FROM job\_history

ORDER BY 1, 2

Can’t make the comparison properly

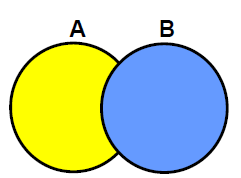
SQL Error: ORA-01789: query block has incorrect number of result columns

01789. 00000 - "**query block has incorrect number of result columns**"

Types – Minus

8-4

MINUS



Rows in the first query A

That are not in second query B

RESULT is the Yellow

PRECEDENCE – equal – evaluated left to right

Caution recommended. Use brackets with INTERSECT

Alternate way of saying it is 🡺 Those rows that are unique to the first query

**SELECT employee\_id, job\_id**

**FROM employee**

**MINUS**

**SELECT employee\_id, job\_id**

**FROM job\_history**

**ORDER BY 1, 2**

Predict how many rows?

Table A or employees has 20-0 🡺 means all 20 rows with no duplicates

The intersect of A and B is 2 rows duplicated 🡺 result so far is 20 – 0 – 2 = 18

A bit more

Give a list of department\_id, location\_id, hire\_date.

That requires 2 tables, EMPLOYEES and DEPARTMENT

Using a JOIN

SELECT E.department\_id, location\_id, hire\_date

FROM employee E, department D

WHERE E.department\_id = D.department\_id

DEPARTMENT\_ID LOCATION\_ID HIRE\_DATE

------------- ----------- ---------

10 1700 17-SEP-87

20 1800 17-FEB-96

20 1800 17-AUG-97

50 1500 16-NOV-99

50 1500 17-OCT-95

50 1500 29-JAN-97

50 1500 15-MAR-98

50 1500 09-JUL-98

60 1400 03-JAN-90

60 1400 21-MAY-91

60 1400 07-FEB-99

80 2500 29-JAN-00

80 2500 11-MAY-96

80 2500 24-MAR-98

90 1700 17-JUN-87

90 1700 21-SEP-89

90 1700 13-JAN-93

110 1700 07-JUN-94

110 1700 07-JUN-94

19 rows selected

SAME EXAMPLE but using UNION

Display department ID, location ID and hire date for all members

To use SET operators you need the same number of columns

Need 3 columns in employees

Need same 3 columns in departments

**PROBLEM:**

* Need hire\_date from employees but it doesn't have a location\_id in employees
* Need location\_id from departments but it doesn't have a date to match with

**SOLUTION**

***Because the expressions in the SELECT lists of the queries must match in number***,

- use the dummy columns and the data type conversion functions to comply with this rule.

You must match the data type when columns do not exist in one or the other table

- use the TO\_CHAR or any other conversion function to get the same data type

SELECT department\_id, TO\_NUMBER (null) as location, hire\_date

FROM employee

UNION

Note the location because TO\_NUMBER (null) does not make a good column heading

SELECT department\_id, location\_id, TO\_DATE (null)

FROM department;

DEPARTMENT\_ID LOCATION HIRE\_DATE

------------- ---------- ---------

10 1700

10 17-SEP-87

20 1800

20 17-FEB-96

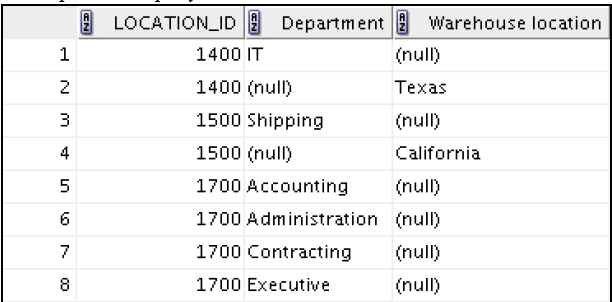
20 17-AUG-97

50 1500

50 17-OCT-95

50 29-JAN-97

From a diffeent example. Notice that the blanks are nulls



50 15-MAR-98

50 09-JUL-98

50 16-NOV-99

60 1400

60 03-JAN-90

60 21-MAY-91

60 07-FEB-99

80 2500

80 11-MAY-96

80 24-MAR-98

80 29-JAN-00

90 1700

90 17-JUN-87

90 21-SEP-89

90 13-JAN-93

110 1700

110 07-JUN-94

190 1700

24-MAY-99

27 rows selected.

Matching SELECT statements

8-25

EXAMPLE 1:

**Display all employees their job id and salary.**

What are the problems?

Employees have several jobs and to display all the jobs requires a join to the job\_history table

But … the job\_history table does not have salary

SELECT employee\_id, job\_id, salary

Matching columns

If no salary will show 0

FROM employee

UNION

SELECT employee\_id, job\_id, 0

FROM job\_history;

EMPLOYEE\_ID JOB\_ID SALARY

----------- ---------- ----------

100 AD\_PRES 24000

101 AC\_ACCOUNT 0

101 AC\_MGR 0

101 AD\_VP 17000

102 AD\_VP 17000

102 IT\_PROG 0 🡸 means, no record of salary of previous job

103 IT\_PROG 9000

104 IT\_PROG 6000

107 IT\_PROG 4200

114 ST\_CLERK 0 **🡸 what does this mean?**

122 ST\_CLERK 0

124 ST\_MAN 5800

141 ST\_CLERK 3500

142 ST\_CLERK 3100

143 ST\_CLERK 2600

144 ST\_CLERK 2500

149 SA\_MAN 10500

174 SA\_REP 11000

176 SA\_MAN 0 🡸 means, no record of salary of previous job

176 SA\_REP 0

176 SA\_REP 8600

178 SA\_REP 7000

200 AC\_ACCOUNT 0

200 AD\_ASST 0

200 AD\_ASST 4400

201 MK\_MAN 13000

201 MK\_REP 0

202 MK\_REP 6000

205 AC\_MGR 12000

206 AC\_ACCOUNT 8300

30 rows 🡸 20 + 10 - 0

Rules or Guidelines

8-5

- The expressions in the SELECT lists must match in number.

- If you select 3 columns in A, then must have 3 columns in B

- The data type of each column in the second query must match the data type of its corresponding column in the first query.

- Parentheses can be used to alter the sequence of execution.

- ORDER BY clause can appear only at the very end of the statement.

Other

• Duplicate rows are automatically eliminated except in UNION ALL.

• Column names from the first query are the ones that appear in the result.

• The output is sorted in ascending order by default except in UNION ALL.