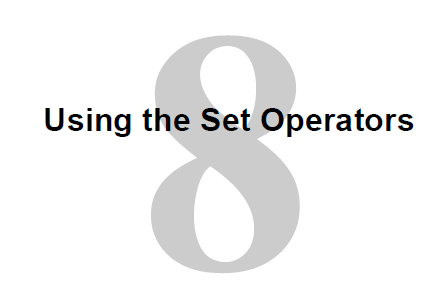
Les08-Set Operators



Objectives

# Describe Set Operators

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

# Control order of rows returned

Topics Covered

Set Operator Types and rules

Tables Used in the lesson

UNION

UNION ALL

INTERSECT

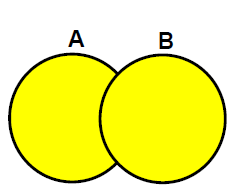
MINUS

Matching SELECT statements

ORDER BY with set operators

Types - Union

UNION



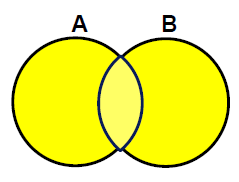
UNION of all the rows in A

With ALL the rows in B

With NO DUPLICATES

Types – Union All

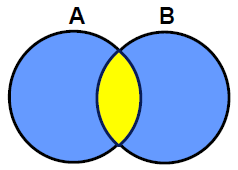
8-4



UNION of ALL the rows in A and B including duplicates

Types – Intersect

INTERSECT



The rows in common to both tables only

A intersect B

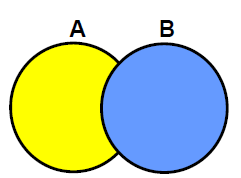
same as

B intersect A

Types – Minus

8-4

MINUS



Rows in the first query A

That are not in second query B

PRECEDENCE – equal – evaluated left to right

Caution recommended. Use brackets with INTERSECT

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.

Tables Used

8-8

Job\_History

SQL> 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**. 🡸🡸

Records start date and end date of employees that switch jobs

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

Employees

SQL> SELECT employee\_id, job\_id, department\_id

2 FROM employees;

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**.🡸 🡸

UNION

8-14

SELECT employee\_id, job\_id

FROM employees

UNION

SELECT employee\_id, job\_id

FROM job\_history;

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**. 🡸 Why 28 rows

There are duplicates and were removed

20 rows + 10 rows – less 2 duplicates 🡺 28

To see the duplicates have to use UNION ALL

UNION ALL

8-16

Duplicate rows not eliminated

Output not sorted

Fasted – since no sorting

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

**FROM employees**

**UNION ALL**

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

**FROM job\_history**

**ORDER BY employee\_id;**

EMPLOYEE\_ID JOB\_ID

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

100 AD\_PRES

101 AD\_VP

102 AD\_VP

103 IT\_PROG

104 IT\_PROG

107 IT\_PROG

124 ST\_MAN

141 ST\_CLERK

142 ST\_CLERK

143 ST\_CLERK

144 ST\_CLERK

149 SA\_MAN

174 SA\_REP

176 SA\_REP

178 SA\_REP

200 AD\_ASST

201 MK\_MAN

202 MK\_REP

205 AC\_MGR

206 AC\_ACCOUNT

102 IT\_PROG

101 AC\_ACCOUNT

101 AC\_MGR

201 MK\_REP

114 ST\_CLERK

122 ST\_CLERK

200 AD\_ASST

176 SA\_REP

176 SA\_MAN

200 AC\_ACCOUNT

30 rows selected. 🡸🡸 **20 rows + 10 rows**

Shows 2 duplicates

What if a big file 🡺 need to ORDER BY to make it easier to see

Try this

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

FROM employees

UNION ALL

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

FROM job\_history

ORDER BY employee\_id;

-- 30 rows

VS

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

FROM employees

UNION

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

FROM job\_history

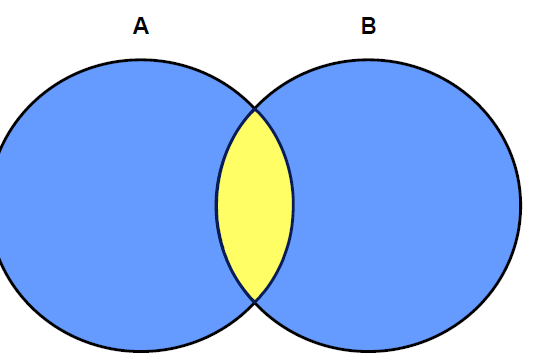
ORDER BY employee\_id;

-- 29 rows – eliminated a duplicate

-- see employee 176 which is the duplicate

INTERSECT

8-19



SELECT statements must be identical in number and data type

They do not have to be the same name

Reversing the order of the tables has no effect on the result

Finds the intersection of the two queries – what they have in common

EXAMPLE:

Display employee and job ids for those employees who currently hold the same job as when they were initially hired

i.e. Have gone back to their original job

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

**FROM employees**

**INTERSECT**

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

**FROM job\_history;**

Results

Emp# Job Title

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

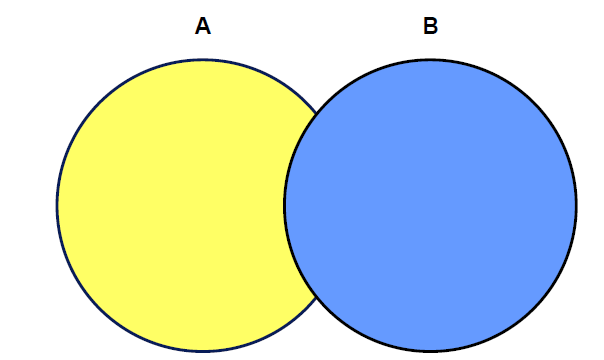
176 SA\_REP

200 AD\_ASST

NOTE: Add the column department\_id and 200 won't appear because department is different

MINUS OPERATOR

8-22



The order of SELECTs does make a difference

EXAMPLE:

Show all employees in the company (A) that - have never held a different job (stored in B).

- i.e. stayed in the same job

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

FROM employees

MINUS

SELECT employee\_id, job\_id

FROM job\_history

ORDER BY 1, 2

Emp# Job Title

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

100 AD\_PRES

101 AD\_VP

102 AD\_VP

103 IT\_PROG

104 IT\_PROG

107 IT\_PROG

124 ST\_MAN

141 ST\_CLERK

142 ST\_CLERK

143 ST\_CLERK

144 ST\_CLERK

149 SA\_MAN

174 SA\_REP

178 SA\_REP

201 MK\_MAN

202 MK\_REP

205 AC\_MGR

206 AC\_ACCOUNT

18 rows selected. 🡸 **20 rows in employees, less the 0 duplicates -2 in the intersect**

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 employees

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

103 IT\_PROG 9000

104 IT\_PROG 6000

107 IT\_PROG 4200

114 ST\_CLERK 0

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

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

EXAMPLE 2:

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

Need location\_id from departments but it doesn't have a date to match with

Need hire\_date from employees but it doesn't have a location\_id in employees

***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 employees

UNION

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

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

FROM departments;

DEPARTMENT\_ID LOCATION HIRE\_DATE

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

10 1700

10 17-SEP-87

20 1800

20 17-FEB-96

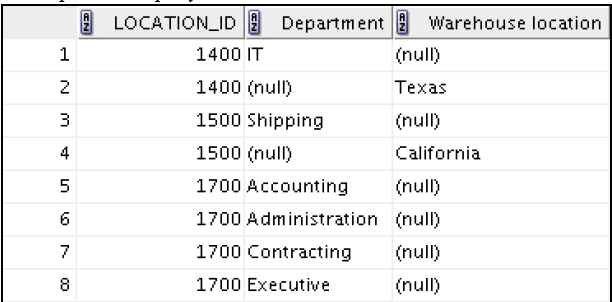
20 17-AUG-97

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.

Rules for ORDER BY

8-28

• The ORDER BY clause can appear only once at the end of the compound query. Same as before – at the end

• Component queries cannot have individual ORDER BY clauses.

Follows the only once rule above

• The ORDER BY clause recognizes only the columns of the first SELECT query.

• By default, the first column of the first SELECT query is used to sort the output in an ascending order.

Need an ORDER BY to force a different sort

SELECT employee\_id, job\_id, salary

FROM employees

UNION

SELECT employee\_id, job\_id, 0

FROM job\_history

ORDER BY 2; 🡸 to change default of sorting on employee-Id