



# ORACLE

## Academy



# Database Programming with SQL

9-3

Usando Operadores de Conjunto

**ORACLE**  
Academy



# Objetivos

- Esta lição abrange os seguintes objetivos:
  - Definir e explicar o propósito dos Operadores de Conjunto
  - Usar um operador de conjunto para combinar várias consultas em uma só
  - Controlar a ordem das linhas retornadas usando operadores de conjunto

# Finalidade

- Os operadores de conjunto são usados para combinar os resultados de instruções SELECT diferentes em uma única saída
- Às vezes, você quer uma única saída de mais de uma tabela
- Se você juntar as tabelas, as linhas que corresponderem aos critérios da junção serão retornados, mas e se for retornado um conjunto de resultados que não atendam às suas necessidades?
- É aí que entram os operadores de conjunto
- Eles conseguem retornar as linhas encontradas em várias instruções SELECT; as linhas que estão em uma tabela, mas não em outra; ou as linhas comuns a ambas as instruções

# Definindo o Cenário

- Para explicar os operadores de conjunto, as duas listas a seguir serão mencionadas nesta lição:
- $A = \{1, 2, 3, 4, 5\}$
- $B = \{4, 5, 6, 7, 8\}$
- Ou, na realidade: duas tabelas, uma chamada A e outra chamada B
- 

A

A_ID
1
2
3
4
5

B

B_ID
4
5
6
7
8

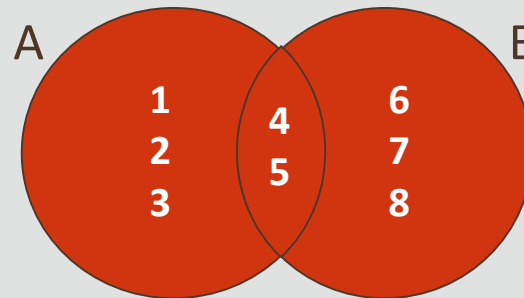
# Regras a Lembrar

- Há algumas regras a lembrar quando se usa operadores de conjunto:
  - O número de colunas e os tipos de dados das colunas devem ser idênticos em todas as instruções SELECT usadas na consulta
  - Os nomes das colunas não precisam ser idênticos
  - Os nomes das colunas na saída são tirados dos nomes das colunas na primeira instrução SELECT
- Portanto, os aliases de coluna devem ser incluídos na primeira instrução, já que você vai querer vê-los no relatório final

# UNION

- O operador UNION retorna todas as linhas de ambas as tabelas, após eliminar as duplicatas

```
SELECT a_id  
FROM a  
  UNION  
SELECT b_id  
FROM b;
```

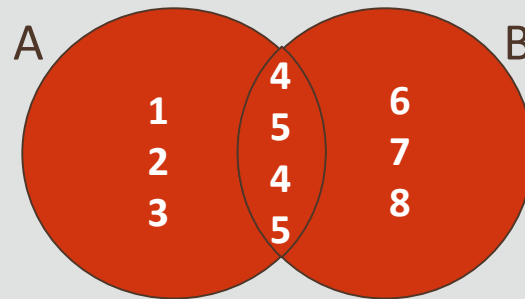


- O resultado da listagem de todos os elementos em A e B após a eliminação das duplicatas é {1, 2, 3, 4, 5, 6, 7, 8}
- Se você juntasse A e B, obteria apenas {4, 5}. Seria necessário fazer uma junção externa completa para obter uma lista como a que se encontra acima

# UNION ALL

- O operador UNION ALL retorna todas as linhas de ambas as tabelas, sem eliminar as duplicatas

```
SELECT a_id  
FROM a  
  UNION  ALL  
SELECT b_id  
FROM b;
```



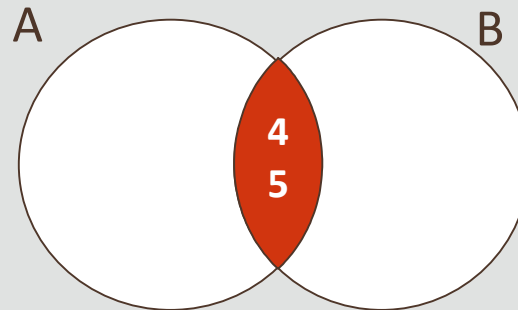
- O resultado da listagem de todos os elementos em A e B sem eliminar as duplicatas é {1, 2, 3, 4, 5, 4, 5, 6, 7, 8}



# INTERSECT

- O operador INTERSECT retorna todas as linhas comuns a ambas as tabelas

```
SELECT a_id  
FROM a  
  INTERSECT  
SELECT b_id  
FROM b;
```

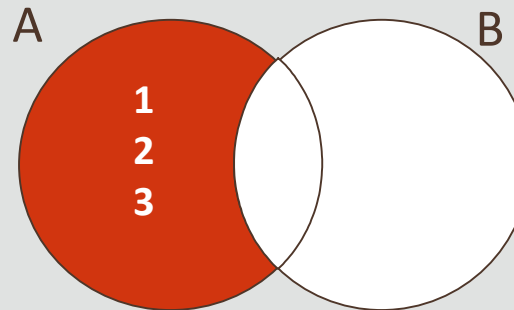


- O resultado da listagem de todos os elementos encontrados em A e B é {4, 5}

# MINUS

- O operador MINUS retorna todas as linhas encontradas em uma tabela, mas não na outra

```
SELECT a_id  
FROM a  
  MINUS  
SELECT b_id  
FROM b;
```



- O resultado da listagem de todos os elementos encontrados em A, mas não em B é {1, 2, 3}
- O resultado de B MINUS A seria {6, 7, 8}

# Exemplos de Operadores de Conjunto

- Às vezes, se estiver selecionando linhas de tabelas que não tenham colunas em comum, você terá que criar suas próprias colunas para igualar o número de colunas nas consultas
- A maneira mais fácil de fazer isso é incluir um ou mais valores nulos na lista SELECT
- Não se esqueça de dar um alias adequado e um tipo de dados correspondente a cada um deles

# Exemplos de Operadores de Conjunto

- Por exemplo:
  - A tabela de funcionários contém data de contratação, id do funcionário e id do cargo.
  - A tabela de histórico de cargos contém id do funcionário e id do cargo, mas não uma coluna de datas de contratação
  - As duas tabelas têm id do funcionário e id do cargo em comum, mas o histórico de cargos não tem uma data inicial
- Você pode usar a função `TO_CHAR(NULL)` para criar colunas correspondentes, como no próximo slide

# Exemplos de Operadores de Conjunto

```
SELECT hire_date, employee_id, job_id
FROM employees
UNION
SELECT TO_DATE(NULL), employee_id,
job_id
FROM job_history;
```

HIRE_DATE	EMPLOYEE_ID	JOB_ID
17-Jun-1987	100	AD_PRES
17-Sep-1987	200	AD_ASST
21-Sep-1989	101	AD_VP
03-Jan-1990	103	IT_PROG
21-May-1991	104	IT_PROG
13-Jan-1993	102	AD_VP
07-Jun-1994	205	AC_MGR
07-Jun-1994	206	AC_ACCOUNT
17-Oct-1995	141	ST_CLERK
17-Feb-1996	201	MK_MAN
11-May-1996	174	SA_REP
29-Jan-1997	142	ST_CLERK
17-Aug-1997	202	MK_REP
15-Mar-1998	143	ST_CLERK
24-Mar-1998	176	SA_REP
09-Jul-1998	144	ST_CLERK
07-Feb-1999	107	IT_PROG
24-May-1999	178	SA_REP
16-Nov-1999	124	ST_MAN
29-Jan-2000	149	SA_MAN
-	101	AC_ACCOUNT
-	101	AC_MGR
-	102	IT_PROG
-	114	ST_CLERK
-	122	ST_CLERK
-	176	SA_MAN
-	176	SA_REP
-	200	AC_ACCOUNT
-	200	AD_ASST
-	201	MK_REP

# Exemplos de Operadores de Conjunto

- A palavra-chave NULL pode ser usada para igualar colunas em uma lista SELECT
- Um valor nulo é incluído para cada coluna ausente
- Além disso, o valor nulo é formatado para ser correspondente ao tipo de dados da coluna que está representando, portanto, as funções TO\_CHAR, TO\_DATE ou TO\_NUMBER são usadas para se obter listas SELECT idênticas

# Operações de Conjunto: ORDER BY

- Se você quiser controlar a ordem das linhas retornadas quando estiver usando operadores de conjunto na sua consulta, a instrução ORDER BY deverá ser usada somente uma vez, na última instrução SELECT da consulta
- Usando o exemplo de consulta anterior, poderíamos ordenar por id\_funcionário para ver os cargos que cada funcionário já teve

```
SELECT hire_date, employee_id, job_id
FROM employees
UNION
SELECT TO_DATE(NULL), employee_id, job_id
FROM job_history
ORDER BY employee_id;
```

# Operações de Conjunto: ORDER BY

```
SELECT hire_date, employee_id, job_id
FROM employees
UNION
SELECT TO_DATE(NULL), employee_id, job_id
FROM job_history
ORDER BY employee_id;
```

HIRE_DATE	EMPLOYEE_ID	JOB_ID
17-Jun-1987	100	AD_PRES
21-Sep-1989	101	AD_VP
-	101	AC_ACCOUNT
-	101	AC_MGR
13-Jan-1993	102	AD_VP
-	102	IT_PROG
03-Jan-1990	103	IT_PROG
21-May-1991	104	IT_PROG
07-Feb-1999	107	IT_PROG
-	114	ST_CLERK
...	...	...



# Operações de Conjunto: ORDER BY

- Poderíamos melhorar a legibilidade da saída incluindo as colunas de datas iniciais e datas finais da tabela de histórico de cargos. Para fazer isso, precisaríamos igualar as colunas em ambas as consultas adicionando mais duas colunas `TO_DATE(NULL)` à primeira consulta

```
SELECT hire_date, employee_id, TO_DATE(null) start_date,  
       TO_DATE(null) end_date, job_id, department_id  
FROM   employees  
       UNION  
SELECT TO_DATE(null), employee_id, start_date, end_date,  
       job_id, department_id  
FROM   job_history  
ORDER BY employee_id;
```

# Operações de Conjunto: ORDER BY

HIRE_DATE	EMPLOYEE_ID	START_DATE	END_DATE	JOB_ID	DEPARTMENT_ID
17-Jun-1987	100	-	-	AD_PRES	90
21-Sep-1989	101	-	-	AD_VP	90
-	101	21-Sep-1989	27-Oct-1993	AC_ACCOUNT	110
-	101	28-Oct-1993	15-Mar-1997	AC_MGR	110
13-Jan-1993	102	-	-	AD_VP	90
-	102	13-Jan-1993	24-Jul-1998	IT_PROG	60
03-Jan-1990	103	-	-	IT_PROG	60
21-May-1991	104	-	-	IT_PROG	60
07-Feb-1999	107	-	-	IT_PROG	60
-	114	24-Mar-1998	31-Dec-1999	ST_CLERK	50
-	122	01-Jan-1999	31-Dec-1999	ST_CLERK	50
16-Nov-1999	124	-	-	ST_MAN	50
17-Oct-1995	141	-	-	ST_CLERK	50
29-Jan-1997	142	-	-	ST_CLERK	50
15-Mar-1998	143	-	-	ST_CLERK	50
...	...	...	...	...	...

# Terminologia

- Estes são os principais termos usados nesta lição:
  - INTERSECT
  - MINUS
  - Operadores de conjunto
  - TO\_CHAR(null) – igualando a lista SELECT
  - UNION
  - UNION ALL

# Resumo

- Nesta lição, você deverá ter aprendido a:
  - Definir e explicar o propósito dos Operadores de Conjunto
  - Usar um operador de conjunto para combinar várias consultas em uma só
  - Controlar a ordem das linhas retornadas usando operadores de conjunto



# ORACLE

## Academy

