Banco de Dados

Rebeca Barros

Consultando múltiplas tabelas

MySQL

Junção (JOIN)

- Durante a normalização, decompomos uma tabela em várias outras tabelas para resolver possíveis problemas. No entanto, quase sempre será necessário gerar algum tipo de resultado que envolva dados de várias tabelas.
- É preciso um mecanismo para juntar esses dados, a qual é dado o nome de junção (**join**).

Produto Cartesiano (Cross-Join)

- O jeito mais simples de juntar dados de duas tabelas. Acontece quando:
  - A condição de união entre as tabelas for omitida;
  - A condição de união entre as tabelas for inválida;
  - Deseja-se ter todas as linhas da primeira tabela unidas com todas as linhas da segunda tabela.

Produto Cartesiano (Cross-Join)

```
SELECT coluna1, [coluna2...colunan] FROM tabela1 JOIN tabela2;
```

- Onde:
  - **coluna1...colunan** são as colunas que serão recuperadas;
  - **tabela1 e tabela2** são as tabelas em que será realizada a busca.

Produto Cartesiano (Cross-Join)

```
-- produto cartesiano entre as tabelas cd e gravadora
SELECT cd.idCD, cd.nome, gravadora.nomeGravadora FROM
cd JOIN gravadora;
```

União Regular (Inner-Join)

- Permite obter registros com dados provenientes de duas ou mais tabelas relacionadas do banco de dados. É o tipo mais usado de junção.

União Regular (Inner-Join)

```
SELECT coluna1, [coluna2..colunan] FROM tabela1 INNER JOIN
tabela2 ON tabela1.coluna = tabela2.coluna
```

- Onde:
  - **coluna1...colunan** são as colunas que serão recuperadas;
  - **tabela1 e tabela2** são as tabelas em que será realizada a busca.
  - **tabela1.coluna** é o nome da primeira tabela concatenado com o nome da coluna que é chave primária ou estrangeira na tabela.
  - **tabela2.coluna** é o nome da primeira tabela concatenado com o nome da coluna que é chave primária ou estrangeira na tabela.

União Regular (Inner-Join)

```
/* Consulta para saber o id do Cd, o nome do Cd
e o nome da Gravadora */
SELECT cd.idCd, cd.nome, gravadora.NomeGravadora
FROM cd INNER JOIN gravadora ON
cd.Gravadora_idGravadora = gravadora.idGravadora;
```

União Regular (Inner-Join)

- Sintaxe antiga para junção (anterior ao SQL-92)

```
-- sintaxe de junção antiga
SELECT cd.idCd, cd.nome, gravadora.NomeGravadora
FROM cd, gravadora WHERE
cd.Gravadora_idGravadora = gravadora.idGravadora;
```

União Regular (Inner-Join)

- Algumas vantagens de usar a notação do SQL92 com a cláusula JOIN e ON:
  - condições de junção e condições de filtragem são separadas em duas cláusulas diferentes (**ON** e **WHERE**), tornando a consulta mais clara de entender;

União Regular (Inner-Join)

- Algumas vantagens de usar a notação do SQL92 com a cláusula JOIN e ON:
  - As condições de junção para cada par de tabelas são contidas em suas correspondentes cláusula ON, tornando mais improvável que parte da condição seja omitida;
  - Consultas que usam a sintaxe SQL92 são portáteis entre diferentes serviços de banco de dados, enquanto que a sintaxe antiga pode ser ligeiramente diferente entre eles.

Apelidando tabelas (Alias)

- Quando múltiplas tabelas são agrupadas em uma consulta é necessário uma forma de identificar de qual tabela é a coluna que você está referenciando. Isso pode ser feito de duas formas:
- - Usando o nome da tabela inteiro, como em **gravadora.nomeGravadora**;
- - Criando um apelido para a tabela e usar esse apelido na consulta.

Apelidando tabelas (Alias)

```sql
-- usando apelido de tabelas
SELECT c.idCd, c.nome, g.NomeGravadora
FROM cd c INNER JOIN gravadora g ON
c.Gravadora_idGravadora = g.idGravadora;
```

```sql
-- outra forma de declarar o apelido
SELECT c.idCd, c.nome, g.NomeGravadora
FROM cd AS c INNER JOIN gravadora AS g ON
c.Gravadora_idGravadora = g.idGravadora;
```

União de mais de duas tabelas

- Similar ao comando de unir duas tabelas, a diferença é que serão adicionados novas cláusulas JOIN-ON para atender as novas tabelas envolvidas.

- **Como seria uma consulta para buscar o nome da música, número da faixa e nome do Cd onde a música se encontra?**

União de mais de duas tabelas

```
SELECT m.nome, f.numero_faixa, c.nome FROM
musica m INNER JOIN faixa f
ON m.idMusica = f.Musica_idMusica
INNER JOIN cd c
ON f.CD_idCd = c.idCd;
```

União de mais de duas tabelas

- Buscando as músicas de um cd específico.

```
SELECT m.nome, f.numero_faixa, c.nome FROM
musica m INNER JOIN faixa f
ON m.idMusica = f.Musica_idMusica
INNER JOIN cd c
ON f.CD_idCd = c.idCd WHERE c.nome = 'Mais do Mesmo';
```

União de mais de duas tabelas

- Buscando as músicas de um cd específico e criando apelidos para as colunas do resultado..

```
SELECT m.nome AS 'Nome da Musica', f.numero_faixa,
c.nome AS 'Nome do cd' FROM
musica m INNER JOIN faixa f
ON m.idMusica = f.Musica_idMusica
INNER JOIN cd c
ON f.CD_idCd = c.idCd WHERE c.nome = 'Mais do Mesmo';
```

União de uma tabela com ela mesma (Self-Join)

- Algumas vezes é necessário unir uma tabela com ela mesma. Como por exemplo em casos de auto-relacionamento.
- Para tal basta-se fazer uso dos apelidos de tabela declarando a mesma tabela com dois apelidos diferentes.

União de uma tabela com ela mesma (Self-Join)

```sql
-- exemplo de self-join. Consulta para saber o nome do cd indicado
SELECT c1.nome AS 'Cd Principal', c2.nome AS 'Cd Indicado'
FROM cd c1 INNER JOIN cd c2 ON c1.CD_indicado = c2.idCd;
```

União de tabelas sem colunas em comum (Non-Equi-Join)

- Todos os exemplos mostrados até o momento aplicam a chamada **equi-join**, que significa que valores provenientes de duas tabelas devem ser iguais para a junção acontecer.
- A maioria das consultas será do tipo equi-join mas é possível juntar tabelas através de um intervalo de valores, a chamada **non-equi-join**.
- Nesse tipo de junção não é definido uma igualdade (=) e sim um intervalo.

União de tabelas sem colunas em comum (Non-Equi-Join)

```sql
-- Consulta para checar a categoria de preço de um cd
SELECT c.nome, c.preco_venda, cg.idCD_Categoria
FROM cd c INNER JOIN cd_categoria cg ON
c.preco_venda BETWEEN cg.menor_preco AND cg.maior_preco;
```

União Externa (Outer Join)

- Quando uma linha não satisfaz a condição de união entre as tabelas, ela não será mostrada no resultado.
- A **união externa** é aquela que inclui linhas no resultado mesmo que não haja relação entre as duas tabelas que estão sendo unidas.

União Externa à Esquerda (Left Outer Join)

- A união pela **esquerda** irá incluir todas as linhas da primeira tabela (a esquerda) e incluirá dados da segunda tabela apenas quando existir igualdade.

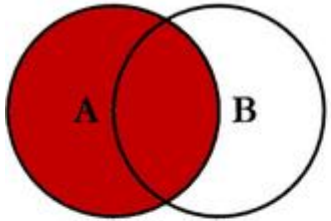
```
-- exemplo de left outer join
SELECT cd.idCd, cd.nome, gravadora.NomeGravadora
FROM cd LEFT OUTER JOIN gravadora ON
cd.Gravadora_idGravadora = gravadora.idGravadora;
```

União Externa à Direita (Right Outer Join)

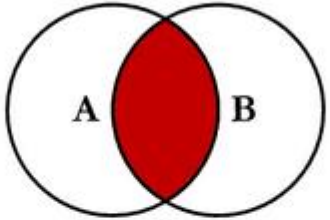
- A união pela **direita** irá incluir todas as linhas da segunda tabela (à direita) e incluirá dados da primeira tabela apenas quando existir igualdade.

```
-- exemplo right outer join
SELECT cd.idCd, cd.nome, gravadora.NomeGravadora
FROM cd RIGHT OUTER JOIN gravadora ON
cd.Gravadora_idGravadora = gravadora.idGravadora;
```

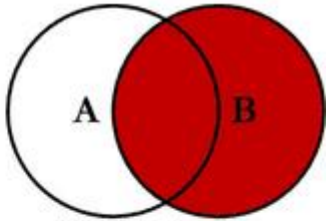
SQL JOINS



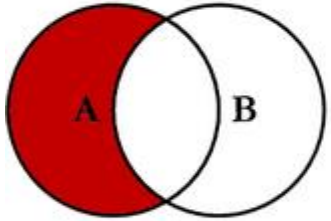

```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
```



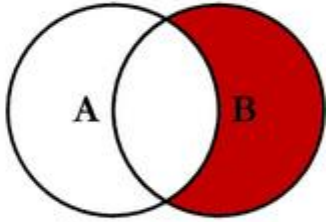

```
SELECT <select_list>
FROM TableA A
INNER JOIN TableB B
ON A.Key = B.Key
```



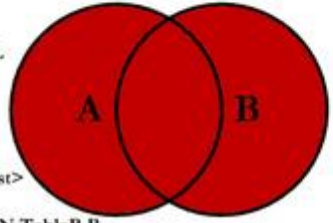

```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
```



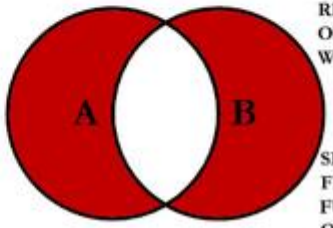

```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
WHERE B.Key IS NULL
```




```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
```




```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
```




```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
OR B.Key IS NULL
```

Referências

- Beaulieu, A. (2009). *Learning SQL: Master SQL Fundamentals*. " O'Reilly Media, Inc.".
- de OLIVEIRA, C. H. P. (2002). *SQL: curso prático*. Novatec.
- Documentação oficial do MySQL. Disponível em: https://dev.mysql.com/doc/refman/8.0/en/
- Curso Completo de MySQL (Bóson Treinamentos). Disponível em: http://www.bosontreinamentos.com.br/curso-completo-de-mysql/