Implementação de banco de dados

Aula 07: Linguagem SQL – Subconsulta e Operadores de Conjunto

Apresentação

Na última aula, começamos a estudar as junções; vimos a junção cruzada (cross join) e a interior (inner join).

Agora, vamos estudar os outros tipos de junção, como a exterior (outer join) e a autojunção.

Objetivos

- Elaborar comandos de junção exterior e de autojunção.

Banco de dados de exemplo

Nesta aula, continuaremos utilizando o banco de dados da empresa para os exemplos.

Modelo lógico:

As tabelas têm os seguintes dados:

Clique nos botões para ver as informações.

Região

Departamento

Empregado

Cliente

Tendo este banco em mente, é extremante recomendável que você execute os comandos de exemplo no PostGreSQL.

Comentário

Foi escolhido como base o PostgreSQL por ser um SGBD mais leve e fácil de instalar, porém você pode usar o SQLServer ou o Oracle; quando houver diferença entre os SGBDs, você receberá um alerta.

Junção exterior (outer join)

Este tipo de junção retorna as linhas que estão relacionadas, como no INNER JOIN, e as não relacionadas de uma ou mais tabelas.

Vejamos um exemplo:

Exemplo

Queremos retornar todos os empregados e os clientes que eles atendem. Este comando seria um comando de junção interior normal, no qual a coluna Vendedor deve ser igualada à coluna ID de empregado. Observe que somente os empregados de ID 5 e 6 atendem algum cliente, portanto apenas eles podem retornar na consulta.

EMPREGADO

VENDEDOR = ID

CLIENTE

O comando seria então:

```
SELECT C.ID, C.NOME, E.ID, E.ULT_NOME, E. CARGO

FROM EMPREGADO E INNER JOIN CLIENTE C ON C.

VENDEDOR = E. ID
```

Resultando na tabela da figura 1, que geraria o seguinte resultado:

Figura 1: resultado do comando.

Note que no resultado voltam apenas os empregados de id 5 e 6, pois estão relacionados como clientes; os demais não aparecem no resultado (id 1, 2, 3 e 4).

E se desejássemos retornar todos os empregados e para os vendedores os dados dos clientes?

Comandaríamos então uma junção exterior. Neste caso, substituiríamos a expressão **inner** por **left** ou **right** caso a tabela que desejamos retornar esteja do lado esquerdo ou direito do comando.

No caso, como empregado está à esquerda, o comando seria:

```
SELECT C.ID, C.NOME, E.ID, E.ULT_NOME, E. CARGO

FROM EMPREGADO E LEFT JOIN CLIENTE C ON C. VENDEDOR = E. ID
```

Resultando na tabela da figura 2:

Figura 2: resultado do comando.

Note que as colunas de clientes ficam nulas nas linhas que correspondem aos empregados que não se relacionam com clientes.

E se o comando fosse de right join, o que aconteceria?

```
SELECT C. ID, C. NOME, E. ID, E. ULT_NOME, E. CARGO

FROM EMPREGADO E RIGHT JOIN CLIENTE C ON C.

VENDEDOR = E. ID
```

Figura 3: resultado do comando.

Retornaríamos todos os clientes e, para os que não se relacionam com vendedores (Casa Desconto), as colunas de empregado seriam nulas.

Uma terceira forma de fazer a junção exterior é com **full join**, conforme mostra a figura 4:

Figura 4: resultado do comando.

O **full join** retorna todos os relacionados e os não relacionados das duas tabelas, preenchendo com nulo as colunas da outra tabela para as linhas não relacionadas. Ou seja, é um **left** e um **right** realizados em conjunto.

Você pode estar se perguntando :

Para que serve isso? Por que eu desejaria retornar os relacionados e os não relacionados?

Normalmente isso ocorre:

1

Quando você deseja saber quais elementos de uma tabela não estão relacionados com os da outra tabela.

2

Quando você precisa fazer uma junção e necessita garantir que retornem todos os elementos de uma determinada tabela.

Vamos discutir cada um destes casos.

- **Determinar os elementos não relacionados:**

Para exemplificar esta situação, vamos incluir mais uma linha na tabela de Região.

```
INSERT INTO REGIAO VALUES (3,'Centro')
```

A tabela então ficaria como está na figura 5:

Figura 5: resultado do comando.

Se olharmos na tabela de Departamentos, veremos que não existe departamento ligado à região Centro (id_regiao = 3).

Figura 6: tabela departamento.

Vamos detalhar a primeira situação: quando você deseja saber quais elementos de uma tabela não estão relacionados com os da outra tabela. Portanto, se você desejasse retornar todos os dados das regiões que não têm departamentos, teria de comandar uma junção exterior.

Façamos passo a passo:

1. Fazer a junção exterior:

Primeiro, temos que determinar como fazer a junção exterior. Para isso, escrevemos o comando retornando todas as colunas das duas tabelas, conforme você pode ver na figura 7.

Figura 7: resultado do comando.

2. Isolar a região não ligada a departamentos:

Analisando o retorno (figura 7), podemos notar que a PK da tabela Departamento (coluna ID) ligada à região Centro é nula. Como vimos, isso acontece devido ao fato de a junção exterior acrescentar ao retorno uma linha toda nula nas colunas da tabela de departamento.

Desta forma, basta filtrar o resultado por esta coluna para isolar a região Centro, conforme você pode ver na figura 8:

Figura 8: resultado do comando.

3. Retornar apenas os dados da Região

Se você observar a figura 6, vai notar que estão retornando as colunas de departamento (todas nulas). Para retornarem apenas as da Região, basta fazer a projeção, conforme você pode ver na figura 9:

Figura 9: resultado do comando.

Duas observações importantes:

Note o uso do alias com o * no select (R.*). É uma forma otimizada de pedir para retornar todas as colunas de uma das tabelas, sem ter que listá-las na cláusula.

O passo a passo mostrado é meramente didático; o comando da figura 9 funciona e gera o resultado esperado.

Vejamos agora um exemplo da segunda situação:

Exemplo

Quando você precisa fazer uma junção e necessita garantir que retornem todos os elementos de uma determinada tabela. Você deseja retornar o nome da região e o do departamento que fica na região. Para regiões que não têm departamento, retornar o nome da região e o texto "**não tem**".

Conforme vimos no caso anterior, a Região 3, Centro, não está ligada a nenhum departamento, portanto se fizéssemos um inner join, ela não retornaria (figura 10).

Figura 10: resultado do comando.

Teremos então que fazer um outer join. Vamos novamente passo a passo:

1. Fazer a junção exterior:

Primeiro, temos que determinar como fazer a junção exterior; para isto, escrevemos o comando retornando todas as colunas das duas tabelas, conforme você pode ver na figura 11:

Figura 11: resultado do comando.

2. Projetar as colunas desejadas:

Como desejamos apenas as colunas Nome da região e Nome do departamento, devemos fazer a sua especificação na cláusula select, conforme você pode ver na figura 12.

Figura 12: resultado do comando.

3. Substituir o nulo no nome do Departamento pela mensagem “Não tem”:

Para fazer esta substituição, vamos utilizar uma função chamada COALESCE. Se o primeiro valor nos parênteses for nulo, a função o substitui pelo segundo valor, conforme podemos ver na figura 13:

Figura 13: resultado do comando.

Duas observações importantes:

Este comando funciona exatamente da mesma forma no Oracle e no SQLServer.

O passo a passo mostrado é meramente didático; o comando da figura 13 funciona e gera o resultado esperado.

Finalmente, você pode estar se perguntando: onde está o outer no comando de junção exterior, já que em todos os comandos que demos ele não aparece, ao contrário da junção interior, na qual sempre escrevemos inner?

Na realidade, tanto inner como outer são opcionais. Na figura 14, podemos ver o comando utilizando outer, e na 15, o de inner join somente com join.

Figura 14: resultado do comando.

Figura 15: resultado do comando.

Qual seria o comando para mostrarmos uma lista de todos os gerentes, cada um com seus subordinados?

Junção exterior – sintaxe tradicional do Oracle

Ao contrário da sintaxe tradicional de inner join, que é igual em todos os SGBDs, existe no Oracle uma sintaxe de outer join proprietária cuja sintaxe tradicional é:

```
SELECT nome da tabela1.nome da coluna, nome da tabela2.nome da coluna ....

FROM nome da tabela1, nome da tabela2

WHERE nome da tabela1.nome da coluna (+) = nome da tabela2.nome da coluna
```

Onde:

(+) é o símbolo do outer join, que pode ser colocado em quaisquer dos lados da cláusula where, mas não em ambos os lados. Este símbolo deve ser colocado seguindo o nome da coluna, que pode não ter correspondente.

No caso do nosso exemplo, envolvendo clientes e vendedores, no qual desejamos retornar todos os empregados e para os vendedores os dados dos clientes, o comando na sintaxe tradicional seria:

```
SELECT C.ID, C.NOME, E.ID, E.ULT_NOME, E. CARGO

FROM CLIENTE C, EMPREGADO E

WHERE C. VENDEDOR (+) = E.ID
```

O resultado pode ser visto na figura 16 e é o equivalente a um right join na sintaxe ANSI.

Figura 16: resultado do comando.

O operador de outer join (+) não tem ligação com o lado da tabela na cláusula from.

Observe na figura 17 o comando invertendo a ordem das tabelas na from e veja que o resultado é o mesmo, equivalendo agora a um left join.

Figura 17: resultado do comando.

Na sintaxe tradicional do Oracle, não é possível fazer um full join; o comando gera erro, conforme podemos ver na figura 18:

Figura 18: resultado do comando.

Autojunção

Uma autojunção [self join] é uma junção da tabela com ela mesma. Na tabela Empregado, por exemplo, cada empregado está subordinado a outro. A coluna 'id_gerente' indica o código do gerente do empregado (figura 19).

Figura 19: resultado do comando.

Para mostrarmos uma lista de todos os gerentes, cada um com seus subordinados, podemos comandar na sintaxe tradicional:

```
SELECT G.ID, G.ULT_NOME, G.CARGO, S.ID, S.ULT_NOME, S. CARGO

FROM EMPREGADO G, EMPREGADO S

WHERE S.ID_GERENTE = G.ID
```

Nesse caso, é obrigatório usar um apelido de tabela para distinguir as duas "cópias" da tabela que estão sendo relacionadas: 'G', no , representa uma linha da tabela Empregado, enquanto Gerente e 'S' representam outra linha, de um subordinado, que estão sendo comparadas entre si.

Podemos ver o resultado dessa consulta na figura 20:

Figura 20: resultado do comando.

Na sintaxe ANSI, o mesmo comando seria:

```
SELECT G.ID, G._ULT_NOME, G.CARGO, S.ID, S.ULT_NOME, S. CARGO

FROM EMPREGADO G INNER JOIN EMPREGADOS ON S.ID_GERENTE = G.ID
```

Obteríamos o mesmo resultado da consulta anterior. Observe a figura 21:

Figura 21: resultado do comando.

Note que na realidade a autojunção não é um tipo de comando de junção; o comando é de inner join; é uma forma de raciocinar para fazer a junção de uma tabela com ela mesma.

Junção using

Existe uma outra terceira forma de se escrever um comando de junção interior, a junção using, mas o que vem a ser isso?

Repare que, tanto na tabela Departamento quanto na Região, temos uma coluna chamada id_regiao, FK em Departamento e PK em Região, que seria utilizada na cláusula de junção no comando (figura 22).

Figura 22: tabelas Departamento e Região.

O comando de junção normal seria o que vemos na figura 23:

Figura 23: inner join on.

Quando esta situação ocorre, em vez de comandarmos inner join on, podemos comandar **inner join using**, pondo entre parênteses a coluna comum às duas tabelas. Veja o comando e seu retorno:

Figura 24: inner join using.

Compare agora os dois retornos: as linhas que voltam são as mesmas, mas a coluna ID_REGIAO retorna apenas uma vez no using.

Você pode utilizar using para comandos de outer join. Veja o exemplo da figura 25:

Figura 25: outer join using.

A junção using, além do PostgreSQL, funciona no Oracle, mas no SQLServer dá erro; veja a figura 26.

Figura 26: inner join using.

Natural join

A junção natural é um tipo de junção que você realiza pelas colunas de mesmo nome de duas tabelas. Embora pareça similar ao using, temos que tomar alguns cuidados. Vamos repetir a junção de Departamento e Região usando natural join:

Figura 27: natural join.

Note que o comando não deu erro, mas não retornaram linhas.O que aconteceu?Observe a figura28.

Existem duas colunas de mesmo nome das tabelas (ID_REGIAO e NOME), mas isso não atrapalhou no using, pois especificamos a coluna a ser usada.

Figura 28: tabelas Departamento e Região.

Para podermos fazer a junção natural aqui, somente ID_REGIAO poderia ser comum.Vamos então criar uma nova tabela, cópia de Região,mas chamando a coluna nome com NOMEREG,conforme mostra a figura 29:

Figura 29: cópia da tabela Região.

Para a criação da tabela, utilizamos o comando select into from, que tem a seguinte sintaxe:

nome_do_arquivo.html

```
Select <colunas a serem copiadas>
Into <tabela a ser criada>
From <tabela de origem dos dados>
```

No caso, copiamos as colunas (SELECT) ID_REGIAO e NOME, alterando o nome desta para NOMEREG para dentro (INTO) da tabela CREG a partir (FROM) da tabela Região.Se olharmos agora a estrutura das tabelas Departamento e Região (figura 30), poderemos notar que existe apenas uma coluna com o mesmo nome entre elas (ID_REGIAO).

Figura 30: tabelas Departamento e Região.

Agora o comando de natural join poderá ser executado (figura 31).

Figura 31: resultado do comando.

Note que, como aconteceu na junção USING ID_REGIAO, retornou uma única vez.

Este comando funciona no Oracle e no SQLServer?Vamos ver; primeiro criaremos a tabela CREG nos dois SGBDs.

O Comando no SQLServer é o mesmo do PostgreSQL, mas no Oracle é diferente: é o comando create table < nome da tabela> as Select o que você deseja copiar (figura 32).

Figura 32: resultado do comando.

Vamos comandar agora o natural join nos dois:

Figura 33: resultado do comando.

No Oracle, o comando funciona, mas no SQLServer dá erro, como acontece com o using.

Noequijoin

Todos os comandos de junção que demos até agora foram feitos com a utilização do sinal de igual (=) na cláusula de junção.Este tipo de junção é chamado de equijoin, ou seja, junção baseada em igualdade.Mas nós podemos dar comando utilizando outros tipos de comparação.Para podemos exemplificar, vamos criar uma tabela chamada Tipo_Salarial, com o seguinte comando:

```
CREATE TABLE TIPO_SALARIO
(TIPO VARCHAR(20) PRIMARY KEY,
LIMITE_INFERIOR INTEGER,
LIMITE_SUPERIOR INTEGER);
```

A seguir, vamos inserir três linhas:

```
INSERT INTO TIPO_SALARIO VALUES ('BAIXO',1000,10000);
INSERT INTO TIPO_SALARIO VALUES ('MEDIO',10001,20000);
INSERT INTO TIPO_SALARIO VALUES ('ALTO',20001,50000);
```

Se você desejasse retornar apenas o **nome** do departamento e o **nome** da região, poderia comandar:

Comentário

Se você estiver usando Oracle, lembre-se de dar o comando de commit após o último insert.

Ao término dos comandos, a tabela ficará com os seguintes dados:

Figura 34: tabela Tipo_Salario.

Agora imagine que você deseja recuperar o ID, o sobrenome e o salário do empregado e o seu tipo salarial.Os três primeiros dados estão na tabela de empregados, mas o tipo está na tabela Tipo_Salario.

Temos então que fazer uma junção, mas como, se não temos uma chave que ligue as tabelas?

Observe a figura 35. Queiroz, o empregado de ID 4, ganha 8000.Se você comparar este valor com os limites de cada tipo da tabela Tipo_Salario (figura 34), notará que ele tem um salário do tipo baixo, pois está entre o limite inferior e o superior deste tipo.

Figura 35: dados de empregados.

A forma de fazer a junção, então, é estabelecer uma cláusula de junção que verifique em qual faixa o salário do empregado se encaixa, conforme podemos ver na figura 36:

Figura 36: resultado do comando.

Comentário

Este comando funciona exatamenteda mesma forma no Oracle e no SQLServer.

Figura 37: resultado do comando.

Para podermos visualizar melhor estes comandos, vamos inserir mais uma linha na tabela de Veículos que não terá um proprietário.

Observe que o veículo inserido tem Proprietário nulo, ou seja, não está associado a nenhum proprietário.

Conforme já tínhamos visto, também temos o proprietário MARIANA ROSA, que não está associado a nenhum veículo.Desta forma, ao fazer a junção interior, não retornará nem a Mariana nem o veículo de placa TTZ0156.

Vamos ver agora como fazer a junção exterior.Neste tipo de junção, temos o conceito de lado.Repare no comando: a tabela Proprietário está à esquerda na cláusula from, e a tabela Veículo à direita.

Se você quiser retornar todos os proprietários,quando apenas os veículos associados como Proprietário estiverem à esquerda, você deverá comandar um left join.

Note que:

- Retornou uma linha com a Mariana Rosa.
- Como não existe um veículo para se associar a Mariana, as colunas de Veículo na linha ficam nulas; isto sempre acontece na junção anterior.

Agora, para voltarem todos os veículos e apenas os proprietários associados, basta trocar left por right.

Agora retornou o veículo TTZ0156, e as colunas de Proprietário na linha são nulas.

Para você voltar todas as linhas associadas e as não associadas das duas tabelas, basta escrever full join.

As utilidades disso são basicamente duas:

1	Suponha que você tenha uma tabela de alunos e uma de fotos, e na de fotos exista uma FK para aluno. Se você deseja retornar todos os alunos para os que têm foto, a única forma é comandar uma junção exterior, pois, na interior, retornariam apenas os alunos que têm foto.
2	Selecionar, por exemplo, o proprietário que não possui veículos. Para obter este resultado, você deve comandar o outer join apropriado (left ou right) e, a seguir, filtrar as linhas cuja chave primária da outra tabela seja nula.

Veja o comando:

O comando funciona devido ao fato de uma chave primária nunca poder ser nula, e o nulo, no caso, aparece devido à junção exterior.

Atividade

Questão 1

Para esta atividade, utilize o banco de dados da seguradora, que conta com o seguinte modelo lógico:

As tabelas contêm os seguintes dados:

proprietário

Modelo

Veículo

1. Insira uma linha na tabela de proprietários com os seguintes dados:

Observe que o veículo inserido deverá ter Proprietário nulo, ou seja, não estará associado a nenhum proprietário.

2. Insira uma linha na tabela de proprietários com os seguintes dados:

Dê um comando de junção que produza o resultado da figura:

3. Dê um comando de junção que produza o resultado da figura:

4. Dê um comando de junção que produza o resultado da figura:

Questão 2

Utilizando o SGBD.

1. Crie um banco de dados chamado Supersport.

Utilizando o SGBD.

2. Utilizando o script disponível aqui <galeria/aula7/anexo/SUPERSPORT.sql.zip> , crie e popule as tabelas da Supersport.

Questão 3

Utilizando o SGBD.

Utilizando o banco de dados da Supersport <galeria/aula7/anexo/MODELOSUPERSPORT.doc> , escreva os comandos de junção exterior solicitados.

1. Mostrar os nomes de todos os clientes e a identificação e o último nome do representante de vendas que atende cada um, ordenados pelo nome do cliente. Existem clientes que não têm um representante de vendas os atendendo.

Retorno esperado:

2. Mostrar os nomes de todos os clientes e os números de suas faturas. Existem clientes que não têm faturas.

Retorno esperado:

3. Mostrar os nomes e as situações de crédito dos clientes que não têm faturas.

Retorno esperado:

Questão 4

Utilizando o banco de dados da Supersport, escreva os comandos de autojunção solicitados.

1. Mostrar a hierarquia da SuperSports, apresentando o último nome de cada gerente e de seu subordinado direto, incluindo os cabeçalhos Gerente e Subordinado, utilizando a sintaxe ANSI.

Retorno esperado:

2. Produzir o mesmo resultado da atividade 1, utilizando a sintaxe tradicional.

3. Mostrar o último nome de cada empregado e de seu gerente, ordenados pelo último nome dos empregados, incluindo os cabeçalhos Empregado e Empregado-Gerente, utilizando a sintaxe ANSI.

Retorno esperado:

4. Produzir o mesmo resultado da atividade 3, utilizando a sintaxe tradicional.

5. Mostrar o último nome de todos empregados e de seus gerentes, ordenados pelo último nome dos empregados, incluindo os cabeçalhos Empregado e Empregado-Gerente, utilizando a sintaxe ANSI. Um empregado pode não ter gerente.

Retorno esperado:

Questão 5

Utilizando o banco de dados da Supersport, escreva os comandos de junção solicitados, nas sintaxes ANSI e tradicional.

1. Mostrar identificador, último nome, código do departamento e nome do departamento de todos os empregados.

Retorno esperado:

Utilizando o banco de dados da Supersport, escreva os comandos de junção solicitados, nas sintaxes ANSI e tradicional.

2. Mostrar código do departamento, código da região e nome da região de todos os departamentos. Chamar as colunas de Departamento, Região e Nome da região.

Retorno esperado:

3. Mostrar identificador, último nome, código e nome do departamento do empregado com último nome Pires.

Retorno esperado:

4. Mostrar identificador e nome de todos os departamentos situados na região de nome América do Norte.

Retorno esperado:

5. Mostrar nome do cliente, código da região e nome da região de todos os clientes das regiões 4 e 5. Atribuir apelidos às tabelas referenciadas.

Retorno esperado:

6. Mostrar o último nome, nome do departamento e nome da região de todos os empregados que recebem comissão.

Retorno esperado:

7. Mostrar nome do produto, código do produto e quantidade faturada dos itens da fatura de número 101.

Retorno esperado:

8. Mostrar nome do cliente, código do produto e quantidade faturada dos clientes cujas faturas totalizaram mais que 100000.

Retorno esperado:

Notas

Título modal [1]

Lorem Ipsum é simplesmente uma simulação de texto da indústria tipográfica e de impressos. Lorem Ipsum é simplesmente uma simulação de texto da indústria tipográfica e de impressos. Lorem Ipsum é simplesmente uma simulação de texto da indústria tipográfica e de impressos.

Título modal [1]

Lorem Ipsum é simplesmente uma simulação de texto da indústria tipográfica e de impressos. Lorem Ipsum é simplesmente uma simulação de texto da indústria tipográfica e de impressos. Lorem Ipsum é simplesmente uma simulação de texto da indústria tipográfica e de impressos.

Referências

DATE, C. J. **Introdução a sistemas de banco de dados**. 7. ed. Rio de Janeiro: Campus, 2000.

ELMASRI, R.; NAVATHE, S. B. **Sistemas de banco de dados**. 7. ed. São Paulo: Pearson Addison Wesley, 2015.

SILBERSCHATZ, A.; KORTH, H. F.; SUDARSHAN, S. **Sistemas de banco de dados**. 5. ed. Rio de Janeiro: Campus, 2006.

Próxima aula

- Subconsultas;
- Operadores de conjunto.

Explore mais

Veja:

Veja:

- Junções entre tabelas
- Introdução ao SQL/ Junções
- Left join e inner join: junção em consultas SQL
- Inner, cross, left, rigth e full joins
- Joins em SQL
- PostgreSQL Prático/DML/Consultas Join