Fundamentos de Bacos de Dados

Join e Group by

- JOIN lista (recupera) registros de duas ou mais tabelas de um banco de dados, se baseando no relacionamento entre certas colunas destas tabelas

- Para utilizar o JOIN, as tabelas ***precisam*** ser relacionadas

- Em Bancos de Dados, as tabelas são relacionadas umas com as outras através de ***chaves primárias e estrangeiras***

- Uma chave primária é uma coluna (ou combinação de colunas) com um ***valor único*** em cada linha. Cada valor de uma chave primária precisa ser único para toda a tabela. O propósito é buscar, de uma única vez, os dados de forma conjunta, que estão divididos entre várias tabelas

- Veja a tabela “pessoa”:

id	sobrenome	nome	endereco	cidade
1	Silva	Luis	Rua Lauro	Florianopolis
2	Souza	Pedro	Rua Schmidt	Rio de Janeiro
3	Santos	João	Rua Nono	São Paulo
4	Alves	Silvano	Rua Margarida	Salvador
5	Souza	Jack	Rua das Araras	Florianopolis

- A coluna ‘id’ é a ***chave primária*** da tabela ‘pessoa’. Isto significa que duas linhas **NÃO** podem ter o mesmo valor de id
- O id distingue duas pessoas, mesmo que elas tenham o mesmo nome

- Agora, temos também a tabela de “compra”:

id_compra	numero	id_pessoa
1	77895	3
2	44678	3
3	22456	1
4	24562	1
5	34764	5

- A coluna ‘id_compra’ é a chave primária da tabela ‘compra’, e a coluna ‘id_pessoa’ se refere a tabela de pessoa, sem utilizar o nome da pessoa em si, mas sim o seu valor de id
- **Perceba que o relacionamento entre as duas tabelas se faz através da coluna ‘id_pessoa’**
- É através deste relacionamento que podemos “juntar” duas tabelas e buscar valores que estão em duas (ou mais) ao mesmo tempo, com apenas uma SQL Esta junção se faz através do INNER JOIN (ou simplesmente JOIN)

id	sobrenome	nome	endereco	cidade
1	Silva	Luis	Rua Lauro	Florianopolis
2	Souza	Pedro	Rua Schmidt	Rio de Janeiro
3	Santos	João	Rua Nono	São Paulo
4	Alves	Silvano	Rua Margarida	Salvador
5	Souza	Jack	Rua das Araras	Florianopolis

id_compra	numero	id_pessoa
1	77895	3
2	44678	3
3	22456	1
4	24562	1
5	34764	5

- Para o nosso exemplo, então, queremos buscar o ***nome*** e o ***sobrenome*** de pessoas que possuem compra cadastrada, junto com o número da **compra**. Devemos então, utilizar a seguinte instrução:

```
SELECT pessoa.nome, pessoa.sobrenome, compra.numero
FROM pessoa
INNER JOIN compra ON pessoa.id=compra.id_pessoa
                          OU
SELECT pessoa.nome, pessoa.sobrenome, compra.numero
FROM pessoa
JOIN compra ON pessoa.id=compra.id_pessoa
```

»

Nota: A instrução JOIN (sem especificar seu tipo) é, implicitamente, um INNER JOIN.

- RESULTADO:

Nome	Sobrenome	Numero
João	Santos	77895
João	Santos	44678
Luis	Silva	22456
Luis	Silva	24562
Jack	Souza	34764

- A palavra reservada INNER JOIN (ou simplesmente JOIN) retorna linhas onde exista o valor do seu relacionamento (no caso, o id da pessoa) nas duas tabelas. Se existem linhas na tabela de pessoas que não tem registro na tabela de compra, estas linhas NÃO SERÃO LISTADAS.

- As funções de agregação, em SQL, podem ser utilizadas com a instrução GROUP BY para agruparmos seus resultados
- O GROUP BY é utilizado em conjunto com as funções de agregação para agrupar seu Resulto por uma ou mais colunas.
- Sintaxe:

```
SELECT nome_coluna, funcao_de_agregacao(nome_coluna2)
FROM tabela
WHERE nome_coluna operador valor
GROUP BY nome_coluna
```

- Exemplo: Temos a seguinte tabela de ‘compra’:

id	data	preco	cliente
1	2011/11/12	1000	Hallan
2	2011/10/23	1600	João
3	2011/09/02	700	Hallan
4	2011/09/03	300	Hallan
5	2011/08/30	2000	Pedro
6	2011/10/04	100	João

- Agora, queremos a soma das compras (coluna preco) de cada cliente.Devemos utilizar o GROUP BY para agrupar os clientes.Utilizaremos, então, a seguinte instrução:
- SELECT cliente, SUM(preco) FROM compra GROUP BY cliente;
- O ResultSet, para a instrução acima, será:

cliente	SUM(preco)
Hallan	2000
João	1700
Pedro	2000

- Percebemos então que o GROUP BY deve ser utilizado sempre que queremos utilizar uma função de agregação para trazer um valor (no exemplo, a coluna preco) com alguma outra coluna onde não efetuamos nenhuma operação (no exemplo, a coluna cliente).

cliente	SUM(preco)
Hallan	2000
João	1700
Pedro	2000

- Exemplo: Veja a tabela ‘pessoa’:

id	sobrenome	nome	endereco	cidade
1	Silva	Luis	Rua Lauro	Florianopolis
2	Souza	Pedro	Rua Schmidt	Rio de Janeiro
3	Santos	João	Rua Nono	São Paulo
4	Alves	Silvano	Rua Margarida	Salvador
5	Souza	Jack	Rua das Araras	Florianopolis

- Temos também uma outra tabela, chamada ‘venda’:

id	data	id_pessoa
1	2011/11/12	3
2	2011/10/10	3
3	2011/09/08	1
4	2011/07/22	1
5	2011/08/30	5

- Lembrando que a coluna id_pessoa da tabela venda é chave estrangeira para a a coluna id da tabela ‘pessoa’.

- Temos também uma terceira tabela, chamada ‘venda_produtos’:

id	produto	valor	id_venda
1	Agua	1000	1
2	Arroz	2000	1
3	Agua	1500	2
4	Fruta	3000	3
5	Feijão	500	5

- A coluna id_venda é chave estrangeira para a a coluna id da tabela ‘venda’.
- Desta forma, temos um relacionamento entre a tabela de venda_produto e a tabela de venda.

- Questão: Queremos, agora, recuperar o nome e o sobrenome de cada pessoa e a soma (do valor) das vendas relacionadas a ela.
- Devemos, então, utilizar a seguinte instrução:

```
SELECT pessoa.nome, pessoa.sobrenome, SUM(venda_produto.valor)
FROM pessoa
JOIN venda on venda.id_pessoa = pessoa.id
JOIN venda_produto on venda_produto.id_venda = venda.id
GROUP BY pessoa.nome, pessoa.sobrenome;
```

Nome	Sobrenome	SUM
João	Santos	4500
Luis	Silva	3000
Jack	Souza	500