
Campus:  Polo Centro - Jequié (BA)
Curso: Desenvolvimento fullstack

Disciplina: Nível 1 Iniciando caminho pelo Java
turma: 2023.1
Semestre: 2024.1

Aluno: Israel dos Santos Hamdan D’Araujo

Título da prática: Vamos manter informações
**Objetivos** da prática

1. Identificar os requisitos de um sistema e transformá-los no modelo adequado.
2. Utilizar ferramentas de modelagem para bases de dados relacionais.
3. Explorar a sintaxe SQL na criação das estruturas do banco (DDL).
4. Explorar a sintaxe SQL na consulta e manipulação de dados (DML)
7. SQL, na plataforma do SQL Server.

**Códigos do roteiro de aula**

1. Alimentando as tabelas de pessoas jurídicas com CNPJ e pessoas físicas com CPF fazendo a relação dos ids de cada uma

```
INSERT INTO PessoasFisicas (idPessoaFisica , CPF)
VALUES
  (1, '41412651561'),
  (2, '51651151516'),
  (3,'7784116516');


 --inserindo pessoas juridicas
INSERT INTO PessoasJuridicas (idPJ, CNPJ)

VALUES
  (4, '2151321531'),
  (5, '1232151546'),
  (6, '1516541351'),
  (7, '5531546535'),
  (8, '58796413561');
```

2. Alimentando a tabela de usuários

```
INSERT INTO Usuarios (idUser, loginUser, Senha)

Values
    (1, 'Op1', 'op1'),
    (2, 'Op2', 'op2')
```

3. Alimentando a tabela de produtos

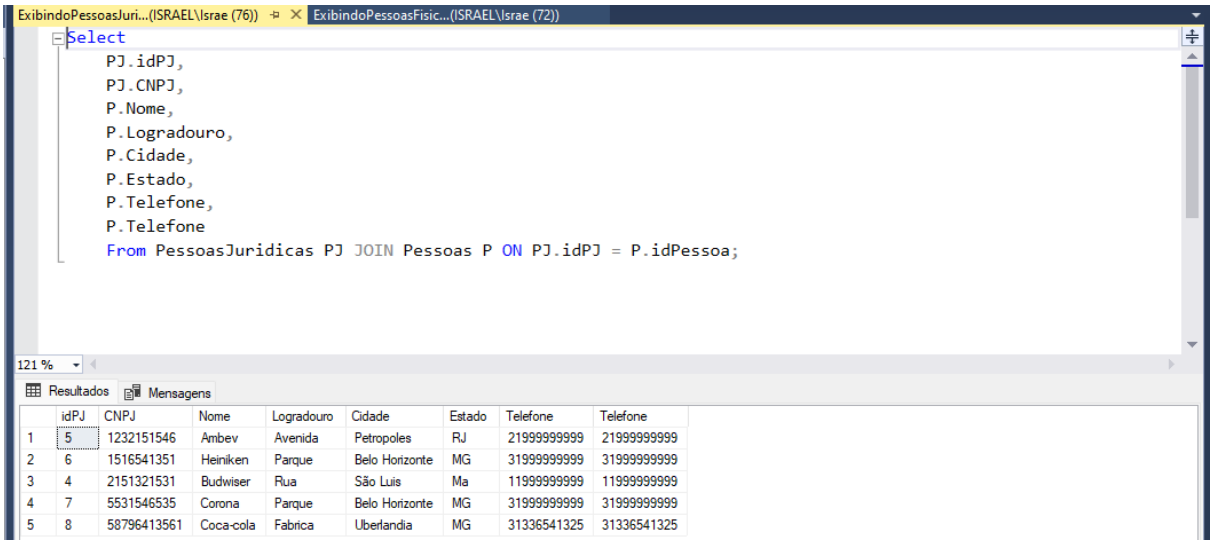
```
INSERT INTO Produtos (idProduto, nomeProduto, quantidade, preco)

Values
    (1, 'Cerveja Budwiser', 800, 5.00),
    (2, 'Cerveja Heiniken', 800,7.80),
    (3, 'Cerveja Heiniken zero álcool', 800, 7.80),
    (4, 'Cerveja Corona', 800, 8.00),
```

No nosso caso é uma distribuidora de bebidas

4. Alimentando a tabela de Movimentos

```
INSERT INTO Movimentos (idMovimento, tipo, quantidade, precoUnitario, FK_idUsuario, FK_idPessoa, FK_idProduto)
VALUES
    (1, 'S', 10, 5.00, 1, 1, 1),
    (2, 'E', 100, 2.00, 1, 7, 4);
```

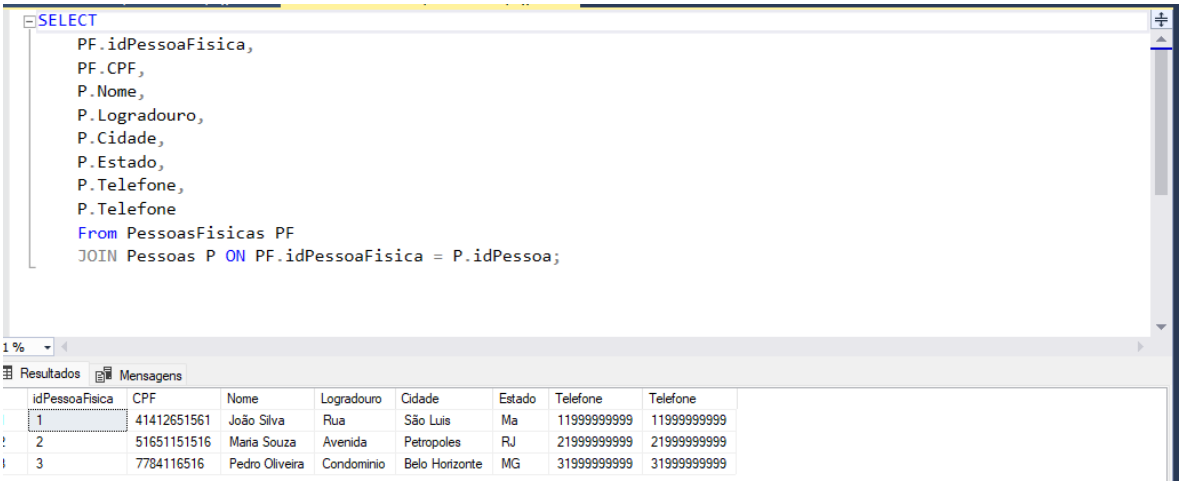
5. Exibindo tabela de pessoas jurídicas



```
Select
    PJ.idPJ,
    PJ.CNPJ,
    P.Nome,
    P.Logradouro,
    P.Cidade,
    P.Estado,
    P.Telefone,
    P.Telefone
    From PessoasJuridicas PJ JOIN Pessoas P ON PJ.idPJ = P.idPessoa;
```

| | idPJ | CNPJ | Nome | Logradouro | Cidade | Estado | Telefone | Telefone |
|---|---|---|---|---|---|---|---|---|
| 1 | 5 | 1232151546 | Ambev | Avenida | Petropoles | RJ | 21999999999 | 21999999999 |
| 2 | 6 | 1516541351 | Heiniken | Parque | Belo Horizonte | MG | 31999999999 | 31999999999 |
| 3 | 4 | 2151321531 | Budwiser | Rua | São Luis | Ma | 11999999999 | 11999999999 |
| 4 | 7 | 5531546535 | Corona | Parque | Belo Horizonte | MG | 31999999999 | 31999999999 |
| 5 | 8 | 58796413561 | Coca-cola | Fabrica | Uberlandia | MG | 31336541325 | 31336541325 |

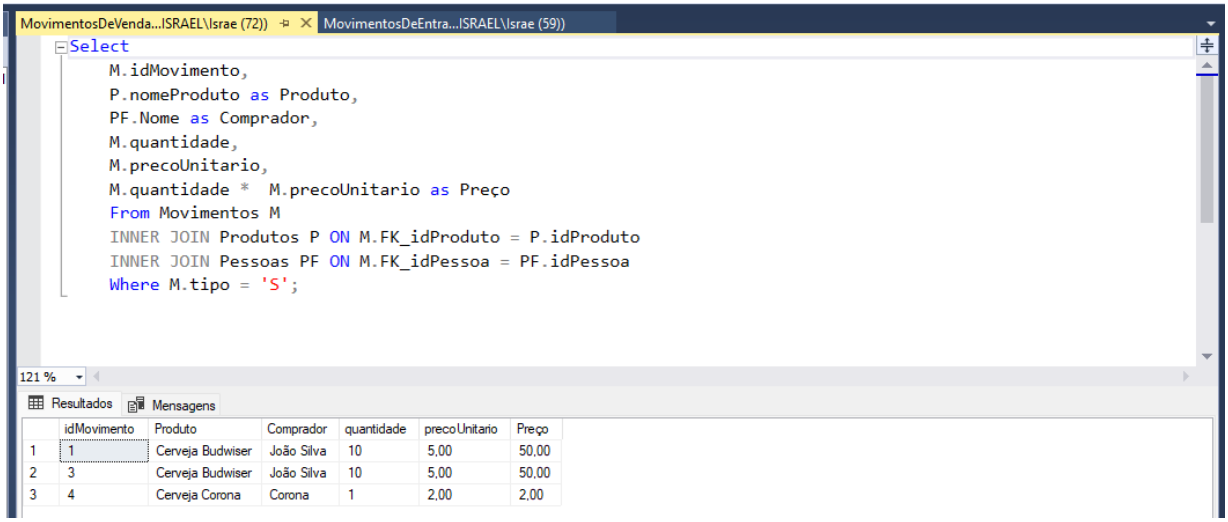
6. Exibindo pessoas físicas



```
SELECT
    PF.idPessoaFisica,
    PF.CPF,
    P.Nome,
    P.Logradouro,
    P.Cidade,
    P.Estado,
    P.Telefone,
    P.Telefone
    From PessoasFisicas PF
    JOIN Pessoas P ON PF.idPessoaFisica = P.idPessoa;
```

| | idPessoaFisica | CPF | Nome | Logradouro | Cidade | Estado | Telefone | Telefone |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 41412651561 | João Silva | Rua | São Luis | Ma | 11999999999 | 11999999999 |
| 2 | 2 | 51651151516 | Maria Souza | Avenida | Petropoles | RJ | 21999999999 | 21999999999 |
| 3 | 3 | 7784116516 | Pedro Oliveira | Condominio | Belo Horizonte | MG | 31999999999 | 31999999999 |

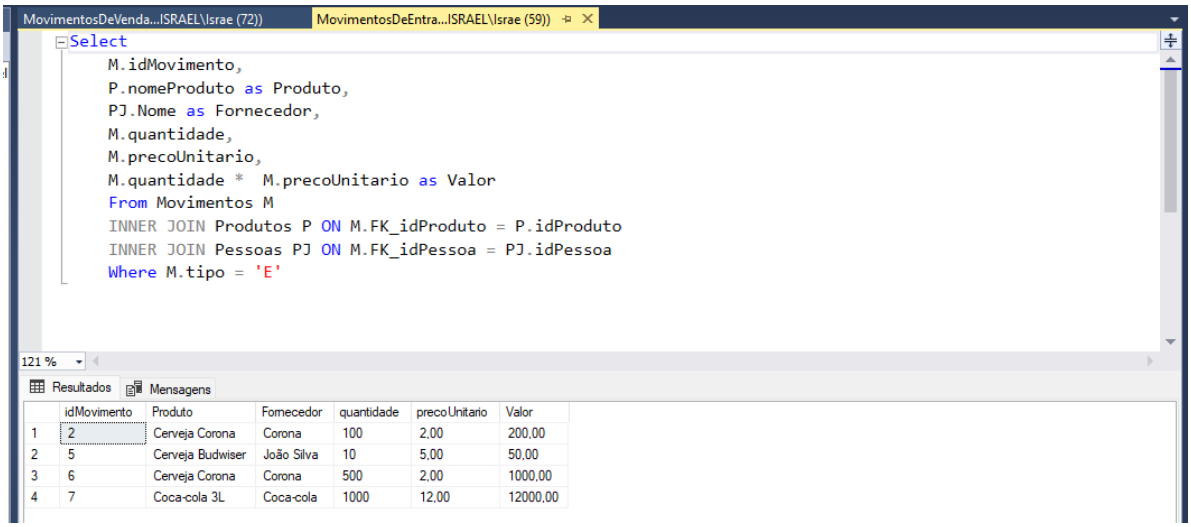
7. Exibindo movimentos de saída:



```
Select
    M.idMovimento,
    P.nomeProduto as Produto,
    PF.Nome as Comprador,
    M.quantidade,
    M.precoUnitario,
    M.quantidade *  M.precoUnitario as Preço
    From Movimentos M
    INNER JOIN Produtos P ON M.FK_idProduto = P.idProduto
    INNER JOIN Pessoas PF ON M.FK_idPessoa = PF.idPessoa
    Where M.tipo = 'S';
```

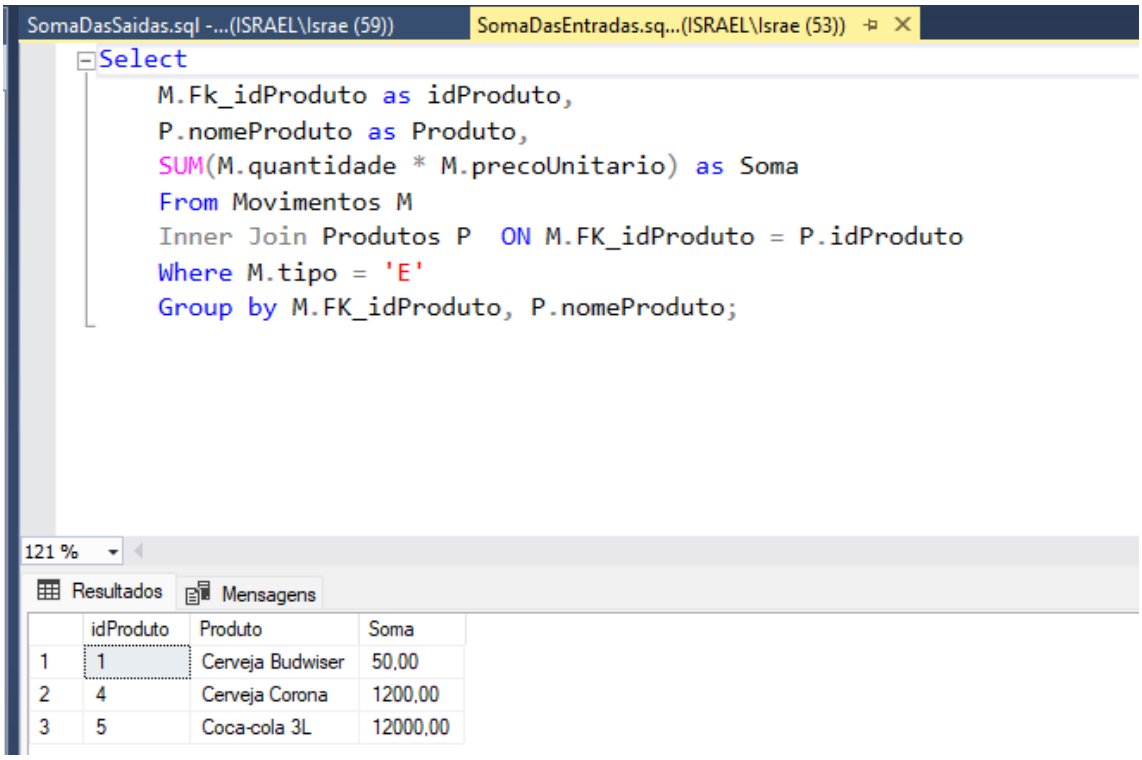
| | idMovimento | Produto | Comprador | quantidade | precoUnitario | Preço |
|---|---|---|---|---|---|---|
| 1 | 1 | Cerveja Budwiser | João Silva | 10 | 5,00 | 50,00 |
| 2 | 3 | Cerveja Budwiser | João Silva | 10 | 5,00 | 50,00 |
| 3 | 4 | Cerveja Corona | Corona | 1 | 2,00 | 2,00 |

8. Exibindo movimentos de entrada:




| | idMovimento | Produto | Fornecedor | quantidade | precoUnitario | Valor |
|---|---|---|---|---|---|---|
| 1 | 2 | Cerveja Corona | Corona | 100 | 2,00 | 200,00 |
| 2 | 5 | Cerveja Budwiser | João Silva | 10 | 5,00 | 50,00 |
| 3 | 6 | Cerveja Corona | Corona | 500 | 2,00 | 1000,00 |
| 4 | 7 | Coca-cola 3L | Coca-cola | 1000 | 12,00 | 12000,00 |

9. Valor das entradas por produto:




| | idProduto | Produto | Soma |
|---|---|---|---|
| 1 | 1 | Cerveja Budwiser | 50,00 |
| 2 | 4 | Cerveja Corona | 1200,00 |
| 3 | 5 | Coca-cola 3L | 12000,00 |

10. Valor das saídas por produtos

SomaDasSaidas.sql -...(ISRAEL\Israe (59)) | SomaDasEntradas.sq...(ISRAEL\Israe (53))

```
Select
    M.Fk_idProduto as idProduto,
    P.nomeProduto as Produto,
    SUM(M.quantidade * M.precoUnitario) as Soma
    From Movimentos M
    Inner Join Produtos P  ON M.FK_idProduto = P.idProduto
    Where M.tipo = 'S'
    Group by M.FK_idProduto, P.nomeProduto;
```

121 %

Resultados | Mensagens

| | idProduto | Produto | Soma |
|---|---|---|---|
| 1 | 1 | Cerveja Budwiser | 100,00 |
| 2 | 4 | Cerveja Corona | 2,00 |

11. Exibindo operadores que não compraram

NaoCompraram.sql -...ISRAEL\Israe (66)) | SomaDasSaidas.sql -...(ISRAEL\Israe (59)) | SomaDasEntradas.sq...(ISRAEL\Israe (53))

```
Select
    U.idUser,
    U.loginUser
    From Usuarios U
    Left join Movimentos M  on U.idUser = M.FK_idUsuario
    Where
        M.idMovimento IS NULL OR M.tipo <> 'E';
```

121 %

Resultados | Mensagens

| | idUser | loginUser |
|---|---|---|
| 1 | 1 | Op1 |
| 2 | 1 | Op1 |
| 3 | 1 | Op1 |
| 4 | 2 | Op2 |
| 5 | 2 | Op2 |
| 6 | 2 | Op2 |

12. Valor de saídas por operador

SomaDeVendasPorUs...ISRAEL\Israe (53)) | SomaDeEntradasPor...ISRAEL\Israe (52))

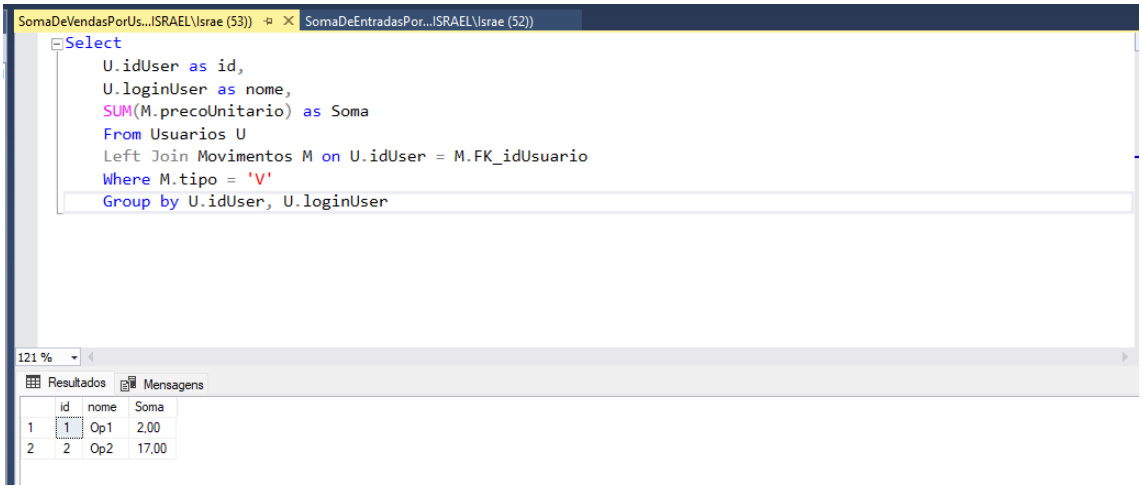
```
Select
    U.idUser as id,
    U.loginUser as nome,
    SUM(M.precoUnitario) as Soma
    From Usuarios U
    Left Join Movimentos M on U.idUser = M.FK_idUsuario
    Where M.tipo = 'E'
    Group by U.idUser, U.loginUser
```

121 %

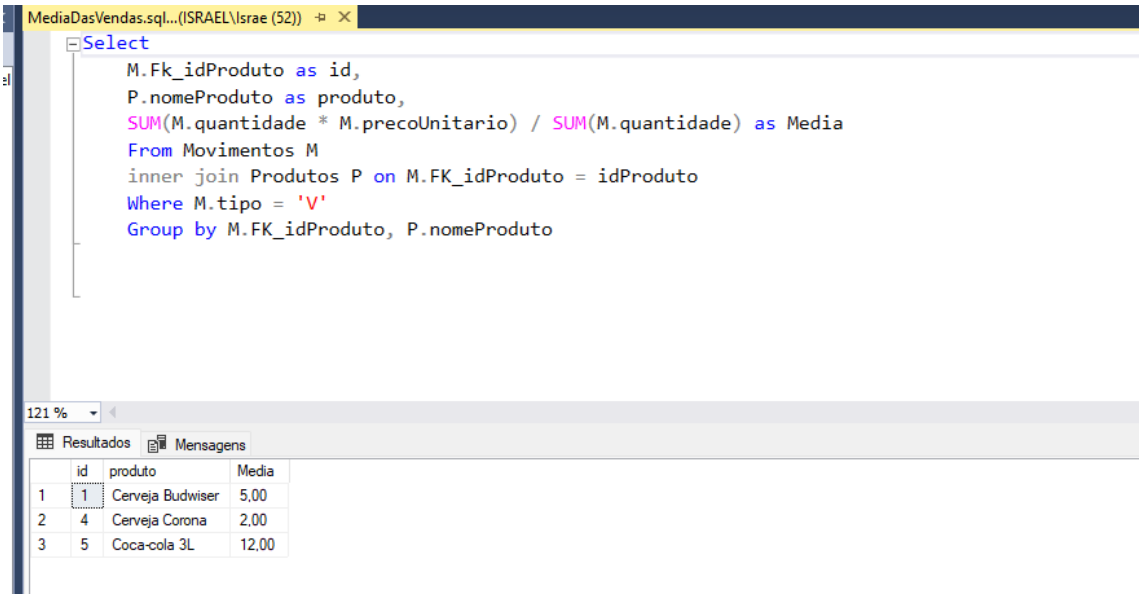
Resultados | Mensagens

| | id | nome | Soma |
|---|---|---|---|
| 1 | 1 | Op1 | 7,00 |
| 2 | 2 | Op2 | 14,00 |

## 13. Saida por operador




## 14. Media ponderada



Análise e conclusão:

1. Diferença entre sequence e identity
    1. *Sequence*: é um objeto no banco de dados, que gera uma sequência de números exclusivos, não é necessário usar tabelas para gerar valores para colunas em várias tabelas, é mais flexível e pode ser compartilhado entre varias tabelas
    2. *Identity*: é uma propriedade passada quando a tabela é criada no MSSQL e é usada para gerar valores automaticamente para uma coluna e a esta coluna ela é associada. Os valores são exclusivos para aquela coluna especifica na tabela
2. Importância das chaves estrangeiras
    1. Elas são essenciais pois mantem a integridade referencial e a consistência em um banco de dados relacional (SQL), garantindo que cada valor em uma coluna de chave estrangeiras correspondem a um valor existente na tabela relacionada
3. Operadores SQL de álgebra relacional e cálculo relacional:
    1. Álgebra relacional: No SQL existem algumas palavras-chave que fazem o relacionamento por exemplo o *Where*, que diz aonde especificamente aquela instrução deve ocorrer, o Join para juntar tabelas, o *Union* que une os dados de duas ou mais consultas e o *Intersect* é usado para retornar apenas registros que são comuns a duas ou mais consultas.
    2. Cálculo relacional: Os operadores incluem proteção (π), seleção (σ), junção natural (⋈), junção externa (⟕, ⟖, ⟗), porém o SQL padrão não adota explicitamente o cálculo relacional, mas muitos de seus

operadores tem seus equivalentes na álgebra relacional

4. Agrupamento em consultas e requisitos obrigatórios
    1. É feito usando o *GROUP BY*, seguido das colunas que precisam ser agrupadas, por exemplo *GROUP BY P.idPessoa, PO.idProduto*. É obrigatório para usar o *Group by* e que todas as colunas selecionadas na lista que não são agregadas devem estar presentes na clausula *GROUP BY*, garantindo que cada linha esteja claramente associada