JUNÇÃO DE TABELAS
FUNÇÕES DE AGREGAÇÃO

1. FUNÇÕES DE AGREGAÇÃO

Uma função de agregação processa um conjunto de valores contidos em uma única coluna de uma tabela e retorna um único valor como resultado**.**

Sintaxe:
```
nome_da_funcao(atributo)
```

1.1 Função MAX
A função MAX avalia um conjunto de valores e retorna o maior entre eles.

1.2 Função MIN
A função MIN verifica em um conjunto de valores qual o menor valor entre eles.

1.3 Função SUM
A função SUM realiza a soma dos valores de um atributo numérico.

1.4 Função AVG
A função AVG calcula a média aritmética dos valores em um atributo.

1.5 Função COUNT
A função COUNT retorna o total de linhas selecionadas. Ela pode receber como parâmetro, o nome do atributo ou um asterisco.

Exemplo 1: Qual é a data de nascimento do aluno mais jovem?

```
SELECT MAX (dt_nascimento)
FROM aluno;
```

Exemplo 2: Qual é o valor da menor mensalidade?

```
SELECT MIN (mensalidade)
FROM curso;
```

Exemplo 3: Qual o total de alunos cadastrados?

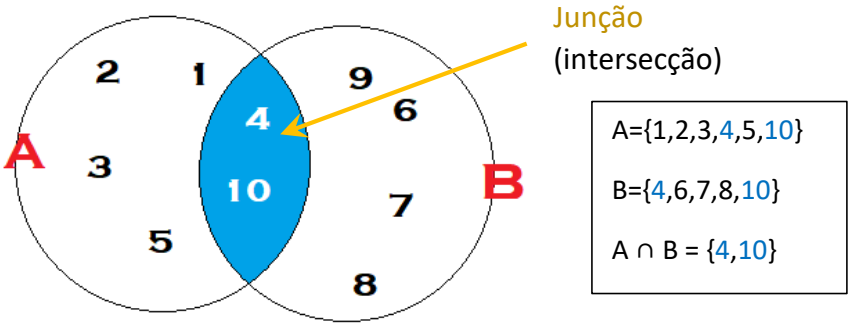
```
SELECT COUNT (*) AS "Total de alunos"
FROM aluno;
```

Exemplo 4: Qual é a data de nascimento do aluno mais jovem e a data do mais velho?

```
SELECT MAX (dt_nascimento), MIN (dt_nascimento)
FROM aluno;
```

2. JUNÇÃO DE TABELAS

Para obter informações de um banco de dados, muitas vezes é necessário acessar simultaneamente várias tabelas. Esse processo leva à realização de **junções** entre as tabelas, para extração das informações necessárias à consulta formulada.



Assim, **junção** é uma condição necessária para se obter dados provenientes de mais de uma tabela.

Os registros de uma tabela podem ser ligados a outros registros de outra tabela por meio de **valores em comum** que possam existir nos atributos correspondentes, normalmente atributos que servem de chave primária e chave estrangeira.

2.1 Condição de Junção

Para se obter dados provenientes de duas ou mais tabelas, deve ser escrita uma **condição de junção** na cláusula **WHERE**.

```
SELECT tabela1.coluna1, tabela2.coluna2
FROM tabela1, tabela2
WHERE tabela1.coluna1 = tabela2.coluna2;
```

Condição de junção

Observação: Podem-se utilizar os operadores LIKE, NOT LIKE, IN, NOT IN, NULL, NOT NULL, os operadores relacionais e operadores AND, OR e NOT, na cláusula WHERE de uma junção de tabelas.

2.2 Qualificadores de Nomes

Um **qualificador de nome** consiste do **nome da tabela**, seguido de um **ponto**, seguido por um **nome de um atributo da tabela**. Por exemplo, o qualificador do atributo NOME da tabela ALUNO será ALUNO.NOME. Os qualificadores de nome são utilizados em uma consulta com junção para estabelecer de qual tabela provém um determinado atributo.

Exemplo:

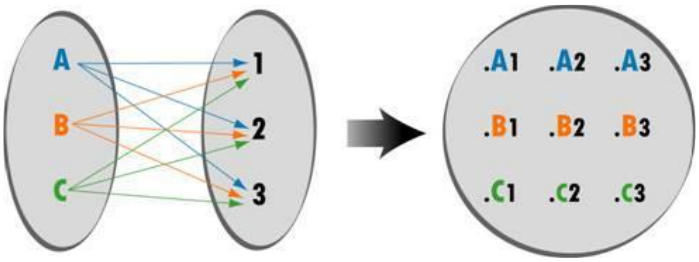
```
SELECT curso.nm_curso, aluno.nome
FROM curso, aluno
WHERE curso.cd_curso=aluno.cod_curso;
```

2.3 Produto Cartesiano

Se a **condição de junção** é inválida ou omitida, o resultado é o **produto cartesiano** em que a combinação de todos os registros das tabelas especificadas no SELECT é exibida.

O produto cartesiano tende a gerar um grande número de registros e o seu resultado raramente é utilizado.

Utiliza-se **produto cartesiano** quando é necessário unir todos os registros das tabelas.



Exemplo:

emp (130 registros)

Kcd_emp	nm_emp	. . .	cd_dept
1073	Jacson C.	. . .	10
1118	Pedro F.	. . .	30
1231	Ramiro A.	. . .	100
⋮	⋮	⋮	⋮
1781	Paula J.	. . .	30

dept (10 registros)

Kcd_dept	nm_dept	nm_loc
10	Marketing	Pelotas
20	GRH	POA
30	Contabilidade	POA
⋮	⋮	⋮
100	Almoxarifado	POA

```
SELECT nm_emp, nm_dept, nm_loc
FROM emp, dept;
```

resultado (1300 registros)

nm_emp	nm_dept	nm_loc
Jacson C.	Marketing	Pelotas
Pedro F.	Marketing	Pelotas
Ramiro A.	Marketing	Pelotas
⋮	⋮	⋮
Jacson C.	GRH	POA
Pedro F.	GRH	POA
⋮	⋮	⋮

Nesse exemplo foi executado um **produto cartesiano** das tabelas EMP e DEPT. Neste caso, poucas informações úteis podem ser extraídas da tabela resultante. Devem ser aplicada **condição de junção** à consulta para um resultado mais preciso.

2.4 Junção

Uma **junção** onde há uma condição de igualdade entre os atributos das tabelas (normalmente entre a chave primária de uma tabela e a chave estrangeira de outra tabela).

emp (130 registros)

Kcd_emp	nm_emp	. . .	cd_dept
1073	Jacson C.	. . .	10
1118	Pedro F.	. . .	30
1231	Ramiro A.	. . .	100
⋮	⋮	⋮	⋮
1781	Paula J.	. . .	30

Chave estrangeira

dept (10 registros)

Kcd_dept	nm_dept	nm_loc
10	Marketing	Pelotas
20	GRH	POA
30	Contabilidade	POA
⋮	⋮	⋮
100	Almoxarifado	POA

Chave primária

```
SELECT emp.nm_emp, dept.nm_dept,
nm_loc
  FROM emp, dept
WHERE emp.cd_dept =
dept.Kcd_dept;
```

resultado (130 registros)

nm_emp	nm_dept	nm_loc
Jacson C.	Marketing	Pelotas
Pedro F.	Contabilidade	POA
Ramiro A.	Almoxarifado	POA
⋮	⋮	⋮
Paula J.	Contabilidade	POA

2.5 Sinônimos

Para não ser necessário escrever o nome da tabela nas qualificações de nomes, é possível utilizar ALIASES. Sua definição é feita na cláusula FROM e utilizada nas outras cláusulas que compõe a consulta (WHERE, ORDER BY, GROUP BY, HAVING e SELECT).

Exemplo:

```
SELECT c.nome, a.nome
FROM curso c, aluno a
WHERE c.cd_curso=a.cod_curso;
```

2.6 Consultas Aninhadas (Subqueries)

Consultas aninhadas são consultas cujo o resultado é utilizado por outra consulta, de forma encadeada e contida no mesmo comando SQL.

Exemplo1: Qual o nome do curso que possui maior mensalidade?

```
SELECT nome
FROM curso
WHERE mensalidade = (SELECT MAX (mensalidade)
                     FROM curso);
```

Exemplo2: Quais os nomes dos alunos que estão no curso de menor mensalidade?

```
SELECT nome
FROM aluno
WHERE cod_curso IN (SELECT cd_curso
                    FROM curso
                   WHERE mensalidade = (SELECT MIN (mensalidade)
                                        FROM curso));
```

Exemplo 3: Quais os nomes dos alunos que estão no curso de menor mensalidade? Informar, também, o nome do curso na listagem.

```
SELECT aluno.nome AS "Nome do aluno?", curso.nm_curso AS "Nome do
curso"
FROM aluno, curso
WHERE aluno.cod_curso = curso.cd_curso AND
      cod_curso IN (SELECT cd_curso
                    FROM curso
                   WHERE mensalidade = (SELECT MIN (mensalidade)
                                        FROM curso));
```

Exemplo 4: Com exceção do curso de maior valor de mensalidade, quais os nomes dos cursos e os valores de suas mensalidade?

```
SELECT nm_curso, mensalidade
FROM curso
WHERE mensalidade < (SELECT MAX (mensalidade)
                     FROM curso);
```

REFERÊNCIAS

ELMASRI, Ramez; NAVATHE, Shamkant B. **Sistemas de banco de dados**. 7. ed. São Paulo: Pearson, 2018.

MACHADO, Felipe; ABREU, Maurício. **Projeto de Banco de Dados**: uma visão prática. São Paulo: Ética, 2009.