BANCO DE DADOS RELACIONAL

PROF. NILTON

Aula de hoje

- Revisão
- Tipos de Junções (*join*).

REVISÃO

FUNÇÕES

Funções de Agregação

AVG – A função AVG calcula o valor médio de um conjunto de valores. Os valores NULL são ignorados.

```
Select AVG(preco) as precoMedio
From Produtos;
```

COUNT – A função COUNT retorna o número de linhas de uma tabela.

```
Select COUNT(*) as Total
From Produtos;
```

FUNÇÕES

SUM – A função SUM retorna a soma de um conjunto de valores. Os valores NULL são ignorados.

```
Select IdProduto, SUM(preco * quantidade) as precoTotal
From Itens;
```

MAX – A função MAX retorna o valor máximo de um conjunto de valores.

```
Select MAX(preco) as maiorPreco
From Produtos;
```

FUNÇÕES

MIN – A função MIN retorna o valor mínimo de um conjunto de valores.

```
Select MIN(preco) as menorPreco
From Produtos;
```

Select - Funções

Distinct

Trata-se de uma cláusula para eliminar repetições em consultas, considerando as colunas informadas na listagem de colunas do comando SELECT que contenham valores iguais como o mesmo registro.

SELECT DISTINCT email **FROM** cliente

Select - Funções

Limit

LIMIT é uma cláusula SQL que especifica o número de linhas que devem ser retornadas no resultado de uma consulta.

Este recurso não está disponível em todos os SGBDS, como alternativa podem ser utilizadas as cláusulas TOP para SQL Server, ROWNUM para Oracle e FIRST para Firebird.

```
SELECT *
FROM clientes
LIMIT 10
```

Select - Funções

Having

O HAVING é uma cláusula SQL utilizada para filtrar resultados, assim como WHERE, com a diferença que ele suporta as **funções de agregação**.

SELECT [coluna1, coluna2, ... | *] **FROM** [tabela1, tabela2, ...] **GROUP BY** [campo1, campo2, ...] **HAVING** [condicao1, condicao2, ...]

SELECT P.nome, **SUM**(V.valor) **as** RECEBIDO **FROM** produto P, venda_produto V **WHERE** **YEAR**(v.Data) = 2018 **AND** P.id = V.id_produto **GROUP BY** P.id **HAVING SUM**(V.valor) > 300

FUNÇÕES

Funções de String

CONCAT – Concatena uma ou mais strings.

A função CONCAT converte todos os argumentos para String antes de concatenar. Se qualquer argumento for NULL, a função retornará NULL.

```
Select CONCAT('MYSQL', 'CONCAT');
```

```
Select CONCAT (Nome, ' ' , Sobrenome) as NomeCompleto
From Pessoas;
```

FUNÇÕES

LENGTH e CHAR_LENGTH – Essas funções retornam a quantidade de caracteres de um String.

```
Select CHAR_LENGTH(nome) as qtdeCaracter
From Pessoa;
```

```
Select IdPessoa,
        IF(CHAR_LENGTH(nome) > 20,
           CONCAT (LEFT(nome, 20), '...'),
           nome) soma
From Pessoa;
```

FUNÇÕES

REPLACE – Essa função permite alterar o conteúdo de uma string, substituindo os caracteres desejados. Retorna uma String “str” com todas as ocorrências de “from_str” em uma nova String “to_str”. A função REPLACE é case-sensitive.

SELECT REPLACE(‘www.fatecararas.com.br’, ‘w’, ‘Ww’);

SELECT REPLACE(nome, ‘ ’, ‘_’)

From Pessoa;

FUNÇÕES

Data e Hora

O SQL nos fornece funções pré-definidas para trabalharmos com informações relacionadas ao tipo DATE, podendo selecionar apenas o dia, mes ou ano de um campo com este tipo.

SELECT [**DAY** | **MONTH** | **YEAR** (tabela1.coluna), ... | *] **FROM** [tabela1]

SELECT MONTH(V.data) **as** MES, **SUM**(V.valor) **as** TOTAL_RECEBIDO

FROM venda_produto V

GROUP BY MES

ORDER BY TOTAL_RECEBIDO **DESC LIMIT** 0, 2

LPAD

Retorna a string **str**, preenchida à esquerda com o **padstr** de string para um comprimento de caracteres **len**. Se **str** for maior que **len**, o valor de retorno é encurtado para **len characters**. Se **padstr** for omitido, a função LPAD preencherá os espaços (versão 10.3).

Sintaxe:

LPAD(str, len ,padstr)

RPAD

Retorna a string **str**, preenchida à direita com a cadeia de caracteres **padstr** para um comprimento de caracteres **len**. Se **str** for maior que **len**, o valor de retorno é encurtado para **len characters**. Se **padstr** for omitido, a função RPAD preencherá os espaços (versão 10.3).

Sintaxe:

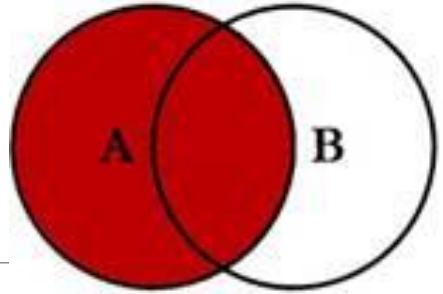
RAPD(str, len, padstr)

TIPOS DE JUNÇÕES (JOIN)

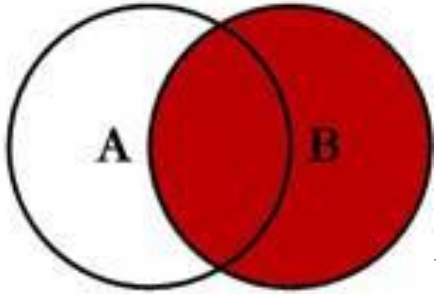
TIPOS DE JUNÇÃO (JOIN)

A figura a seguir traz uma representação gráfica, baseada na Teoria dos Conjuntos, muito conhecida na matemática. Nessa imagem, temos a representação de duas tabelas (A e B) e o resultado esperado por cada tipo de join (a área em vermelho representa os registros retornados pela consulta).

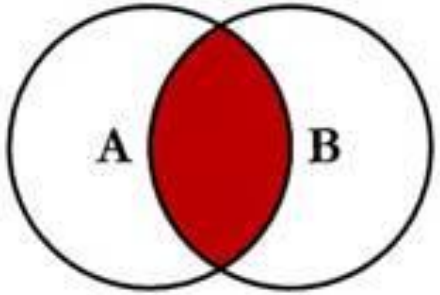
SQL JOINS



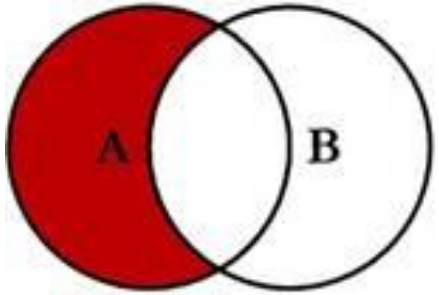

```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
```



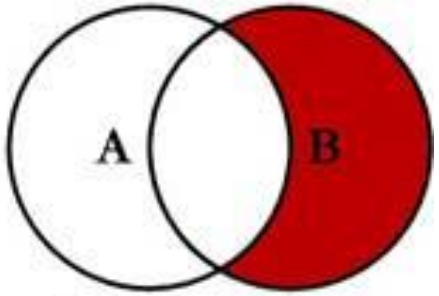

```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
```



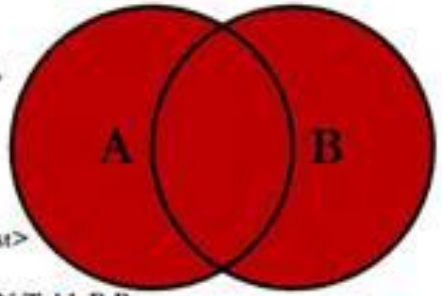

```
SELECT <select_list>
FROM TableA A
INNER JOIN TableB B
ON A.Key = B.Key
```



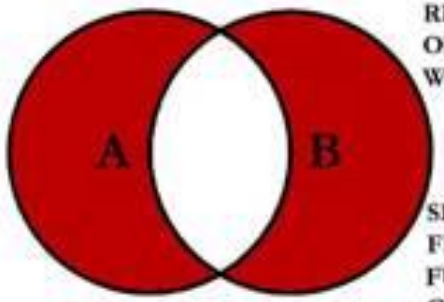

```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
WHERE B.Key IS NULL
```




```
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
```

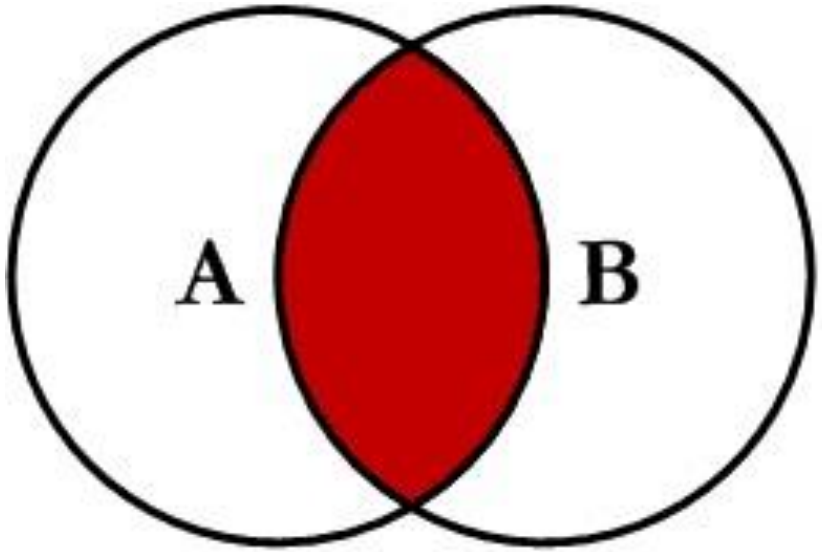
Modelo

```
CREATE TABLE TabelaA(
	Nome varchar(50) NULL
)

CREATE TABLE TabelaB(
	Nome varchar(50) NULL
)
```

Inner Join

O Inner Join é o método de junção mais conhecido e, como ilustra abaixo, retorna os registros que são comuns às duas tabelas.

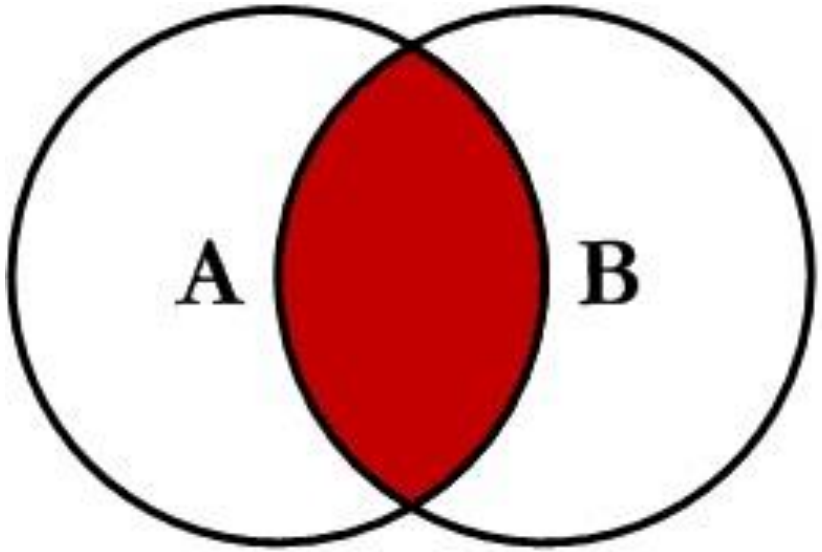
```
SELECT a.Nome, b.Nome
FROM TabelaA as A
INNER JOIN TabelaB as B
            on a.Nome = b.Nome
```



Natural Join

O Natural Join é o método de junção que utiliza os pares de tuplas que possuem os mesmo valores nos atributos que aparecem das relações, como ilustra abaixo, retorna os registros que são comuns às duas tabelas.

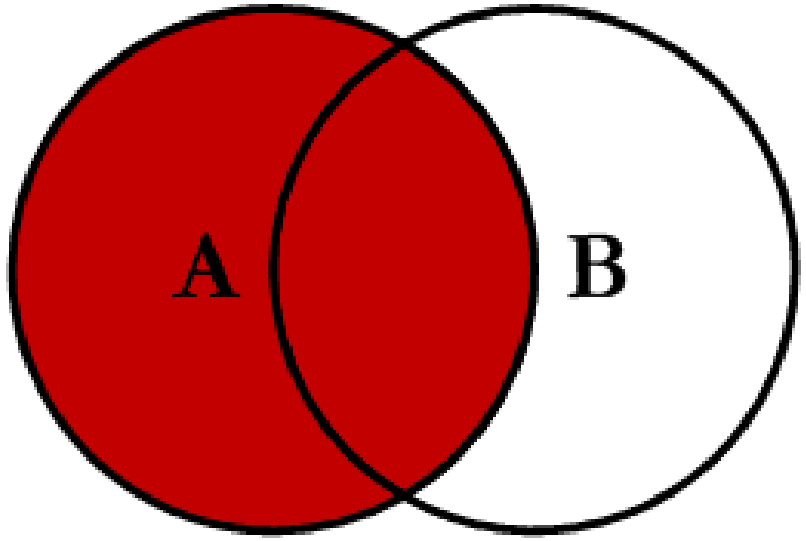
```
SELECT a.Nome, b.Nome
FROM TabelaA as A
NATURAL JOIN TabelaB as B
```



Left Join

O Left Join, cujo funcionamento é ilustrado na figura abaixo, tem como resultado todos os registros que estão na tabela A (mesmo que não estejam na tabela B) e os registros da tabela B que são comuns à tabela A.

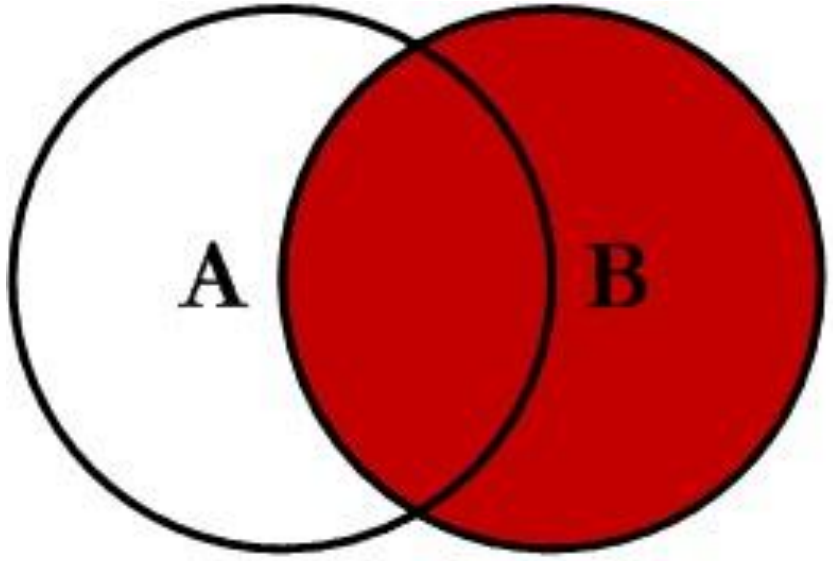
```
SELECT a.Nome, b.Nome
FROM TabelaA as A
LEFT JOIN TabelaB as B
            on a.Nome = b.Nome
```



Right Join

Usando o Right Join, conforme mostra a figura abaixo, teremos como resultado todos os registros que estão na tabela B (mesmo que não estejam na tabela A) e os registros da tabela A que são comuns à tabela B.

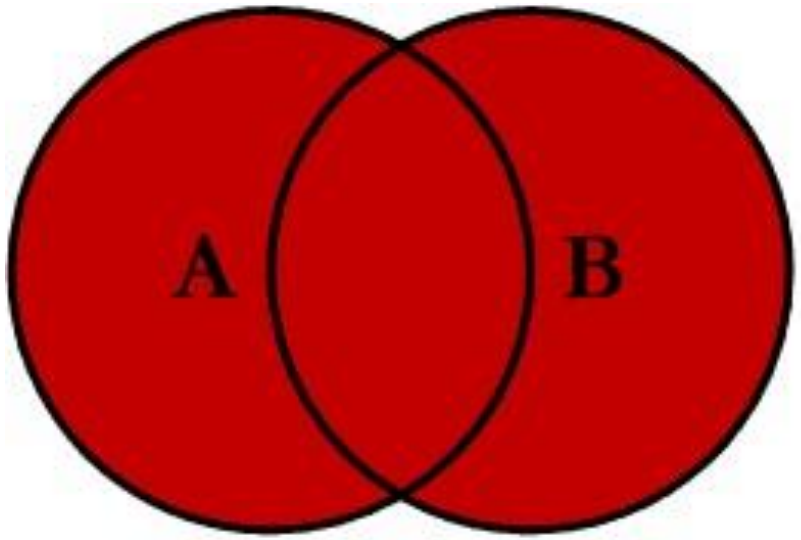
```
SELECT a.Nome, b.Nome
FROM TabelaA as A
RIGHT JOIN TabelaB as B
            on a.Nome = b.Nome
```



Outer Join

O Outer Join (também conhecido por Full Outer Join ou Full Join), conforme mostra abaixo, tem como resultado todos os registros que estão na tabela A e todos os registros da tabela B.

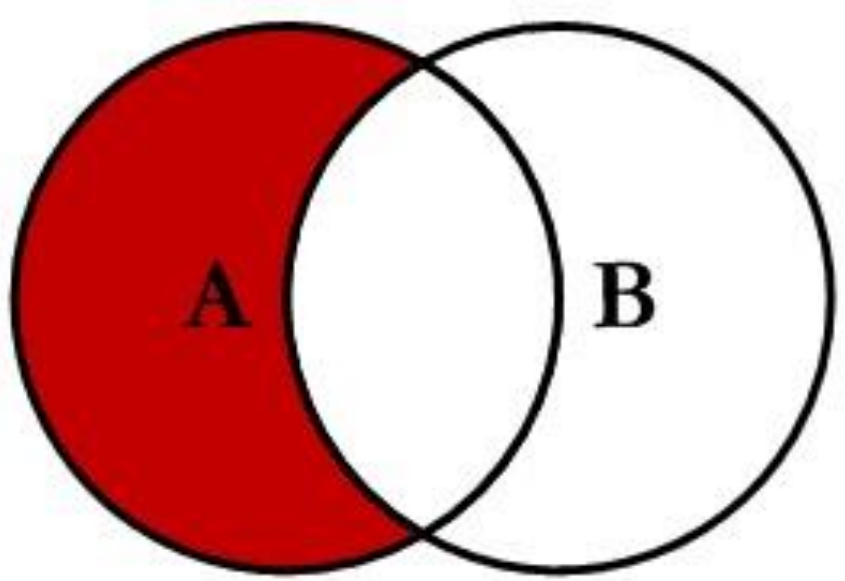
```
SELECT a.Nome, b.Nome
FROM TabelaA as A LEFT JOIN
TabelaB as B on a.Nome = b.Nome

UNION

SELECT a.Nome, b.Nome
FROM TabelaA as A RIGHT JOIN
TabelaB as B on a.Nome = b.Nome
```



Left Excluding Join

Na Figura abaixo temos a representação gráfica do Left Excluding Join, que retorna como resultado todos os registros que estão na tabela A e que não estejam na tabela B.

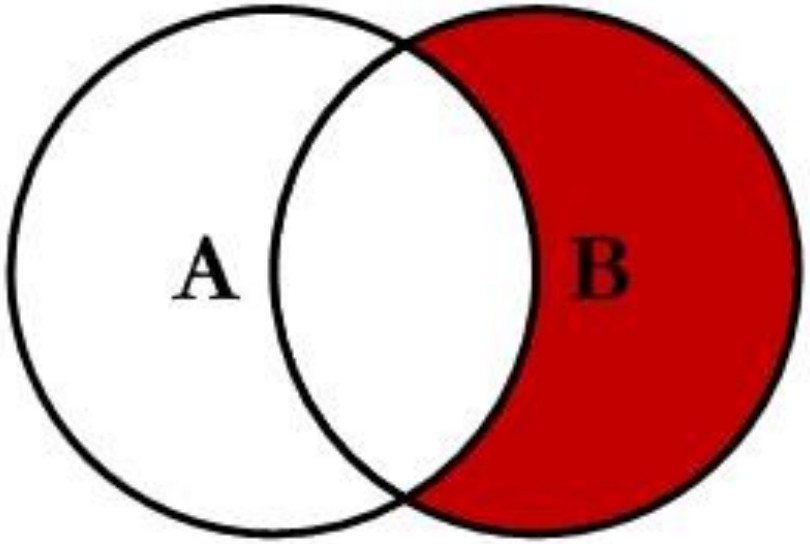
```
SELECT a.Nome, b.Nome
FROM TabelaA as A
LEFT JOIN TabelaB as B
            on a.Nome = b.Nome
WHERE b.Nome is null
```



Right Excluding Join

O Right Excluding Join, como ilustra a **abaixo**, retorna como resultado todos os registros que estão na tabela B e que não estejam na tabela A.

```
SELECT a.Nome, b.Nome
FROM TabelaA as A
RIGHT JOIN TabelaB as B
            on a.Nome = b.Nome
WHERE a.Nome is null
```



Outer Excluding Join

Usando o Outer Excluding Join, conforme mostra a Figura abaixo, teremos como resultado todos os registros que estão na tabela B, mas que não estejam na tabela A, e todos os registros que estão na tabela A, mas que não estejam na tabela B.

```
SELECT a.Nome, b.Nome
FROM TabelaA as A LEFT JOIN
TabelaB as B on a.Nome = b.Nome
WHERE b.Nome is null

UNION

SELECT a.Nome, b.Nome
FROM TabelaA as A RIGHT JOIN
TabelaB as B on a.Nome = b.Nome
WHERE a.Nome is null
```

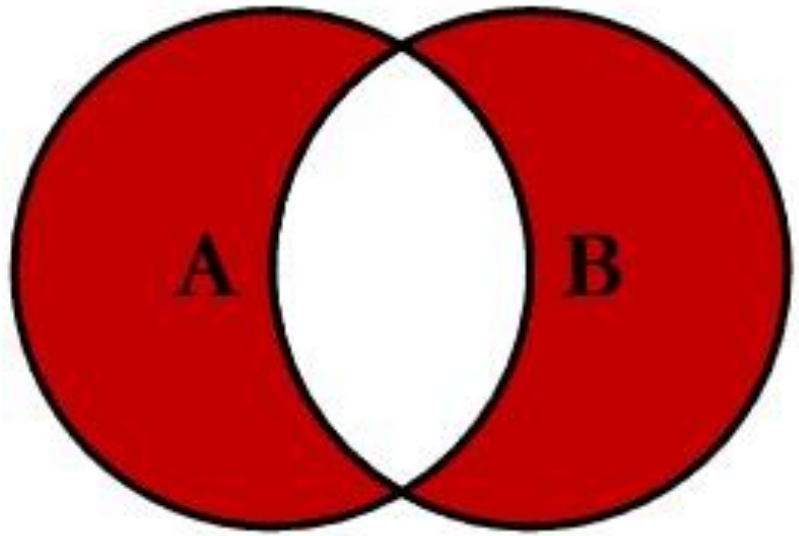


Diagrama de Venn

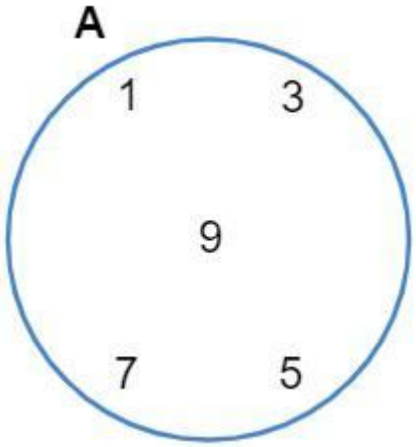
O diagrama de Venn, também conhecido como diagrama de Venn-Euler, é uma maneira de representar graficamente um conjunto, para isso utilizamos uma linha fechada que não possui auto intersecção e representamos os elementos do conjunto no interior dessa linha. A ideia do diagrama é facilitar o entendimento nas operações básicas de conjuntos, como: relação inclusão e pertinência, união e intersecção, diferença e conjunto complementar.

Representações

Como apresentado, o diagrama de Venn consiste em uma linha fechada (que não se entrelaça) na qual “colocamos” os elementos do conjunto em questão, logo, podemos representar um ou vários conjuntos de maneira simultânea. Veja os exemplos:

- Conjunto único

Podemos representá-lo utilizando uma única linha fechada, por exemplo, vamos representar o conjunto A = {1, 3, 5, 7, 9}:



Representações

- **Entre dois conjuntos**

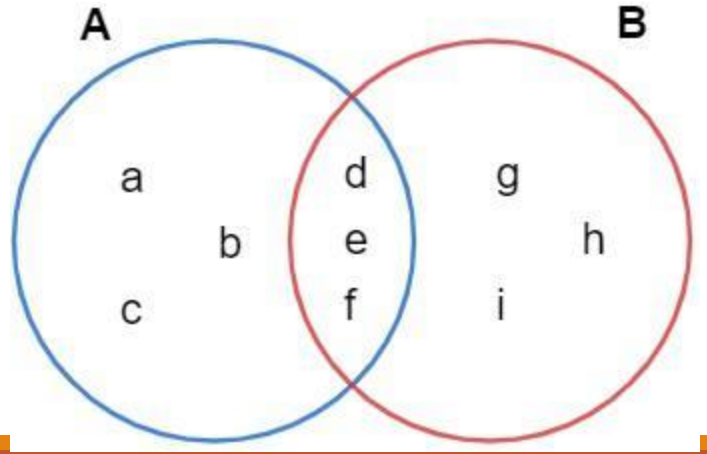
Devemos fazer dois gráficos como o da representação do conjunto único. Entretanto, das operações com conjuntos sabemos que: dado dois conjuntos, eles podem ter intersecção ou não. Caso os dois conjuntos não possuam intersecção, eles recebem o nome de **conjuntos disjuntos**.

Exemplo 1

Represente, utilizando o diagrama de Venn, os conjuntos **A = {a, b, c, d, e, f} e B = {d, e f, g, h, i}.**

Observe que a intersecção é a parte do diagrama que pertence aos dois conjuntos, assim como na definição.

$A \cap B = \{d, e, f\}$



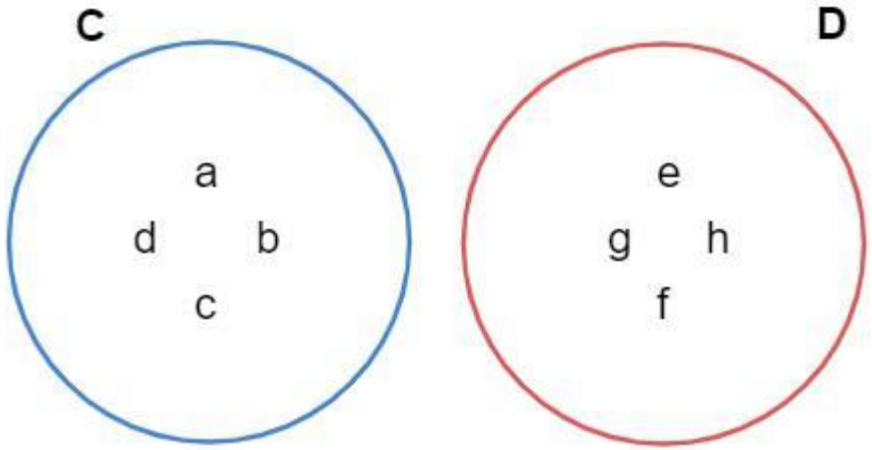
Representações

Exemplo 2

Represente os conjuntos C = {a, b, c, d}e D = {e, f, g, h}.

Observe que a intersecção desses conjuntos é vazia, pois não possui nenhum elemento que pertença simultaneamente a ambos, ou seja:

$C \cap D = \{\ \}$

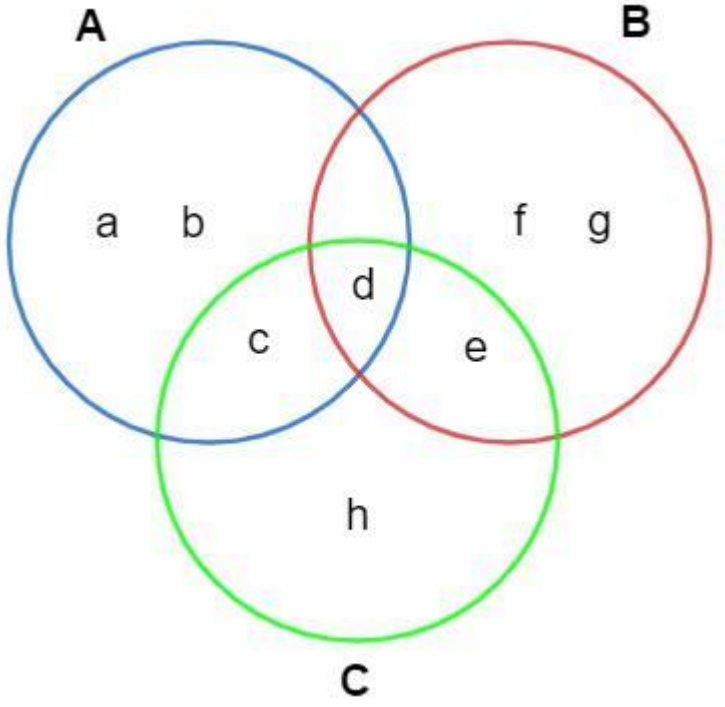


Representações

- Entre três conjuntos

A ideia por trás da representação utilizando o diagrama de Venn para três conjuntos é semelhante à da representação entre dois conjuntos. Nesse sentido, os conjuntos podem ser disjuntos um a um, isto é, não possuem nenhuma intersecção; ou podem ser disjuntos dois a dois, ou seja, somente dois deles possuem intersecção; ou todos possuem intersecção.

Exemplo
Representação, utilizando o diagrama de Venn,
dos conjuntos A = {a, b, c, d}, B = {d, e, f, g} e C = {d, e, c, h}.



REFERÊNCIAS

http://www.mysqltutorial.org/basic-mysql-tutorial.aspx

https://www.tutorialspoint.com/mariadb/

https://www.tecmint.com/learn-mysql-mariadb-for-beginners/

http://ronaldodba.blogspot.com/2015/03/entendendo-os-deadlocks.html

https://www.devmedia.com.br/transacoes-no-mysql/21050

http://sqlparatodos.com.br/transacoes-no-mysql/

https://www.upinside.com.br/blog/aprenda-a-trabalhar-com-transacao-no-mariadb

https://mariadb.com/kb/pt-br/isolation/