BANCO DE DADOS

UniCesumar

PRINCIPAIS CARACTERÍSTICAS

1. Podemos definir banco de dados como uma coleção de dados relacionados.

2. Existe uma categoria de software especializado que é desenvolvido especificamente com o propósito de se gerenciar essas coleções de dados: os sistemas gerenciadores de banco de dados – popularmente reconhecidos pela sigla SGBD.

3. Consideramos como transação um conjunto de uma ou mais operações que compõem uma única tarefa ou unidade lógica de trabalho a ser executada.

4. Para garantir a integridade dos dados, deve-se manter as propriedades que chamamos de ACID: Atomicidade, Consistência, Isolamento e Durabilidade.

5. Atomicidade: seria o conceito do "tudo ou nada", ou seja, as operações realizadas numa transação sejam todas realizadas por completo ou que nenhuma seja realizada.

6. Consistência: assegura que a execução de qualquer transação trará o banco de dados de um estado consistente para outro estado também consistente.

7. Isolamento: determina que o resultado da execução concorrente de um conjunto de transações terá o mesmo resultado de sua execução em série (uma após a outra).

8. Durabilidade: garante que uma vez que uma transação tenha sido finalizada com sucesso, os dados terão a garantia de terem sido armazenados corretamente.

Essas são algumas características importantes de um Banco de Dados.

UniCesumar

MODELAGEM DE DADOS

DER - DIAGRAMA ENTIDADE E RELACIONAMENTO

MODELO RELACIONAL

O modelo relacional permite a representação da estrutura lógica do projeto com uma visão genérica. Sua estrutura é feita de forma clara e simples, possibilitando representar os dados do mundo real como objetos denominados entidades ou conjunto de entidade.

DER - DIAGRAMA ENTIDADE E RELACIONAMENTO

O DER é a representação gráfica de situações práticas em um banco de dados. Os principais componentes de um DER são: Entidades, Atributos e Relacionamentos.

ENTIDADES

Entidades são tabelas.

São representadas graficamente por um retângulo com o nome da entidade dentro.

Para identificarmos uma entidade, devemos considerar os objetos, coisas ou algo que seja relevante no levantamento dos dados.

ATRIBUTOS

Os atributos são propriedades utilizadas para descrever uma entidade, podemos afirmar que os Atributos são as características contidas nas Entidades.

Normalmente representados por um círculo.

RELACIONAMENTOS

No Modelo Entidade Relacionamento (MER), não é permitido ligar uma entidade diretamente à outra. Quando há uma associação, ela é representada por um relacionamento.

Relacionamentos são representados por um losango.

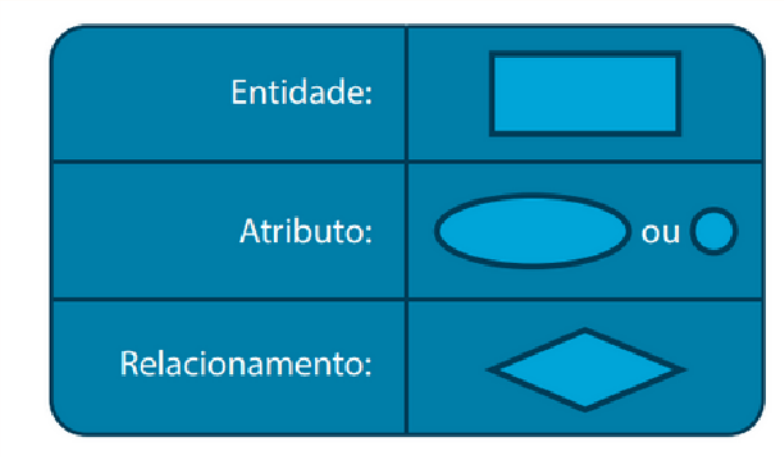
CARDINALIDADE

É o número máximo e mínimo de ocorrências de uma entidade que estão associadas às ocorrências de outra entidade que participa do relacionamento. Vejamos alguns tipos de cardinalidade: (0) – é quando uma ocorrência se relaciona com (no mínimo) nenhuma de outra entidade.
(0:1) – No mínimo nenhuma ocorrência em uma entidade para no máximo uma ocorrência na outra entidade.
Um para Um (1:1) Acontece quando a ocorrência de uma entidade se relaciona com (no máximo) uma ocorrência de outra e vice-versa.

MODELO RELACIONAL

O modelo relacional a seguir foi criado em 1976, pelo Dr. Peter Pin-Shan Chen, que é um cientista da computação americano e professor de ciência da computação na Louisiana State University, conhecido como criador do modelo entidade relacionamento.

Na figura a seguir podemos verificar os componentes de um DER (Diagrama Entidade Relacionamento).



ENTIDADES: A entidade ou tabela trata-se de uma representação gráfica de um conjunto, conjunto este cuja representação física ou gráfica padrão é feita por meio de um **retângulo** com o nome da entidade dentro dele. Para identificarmos uma entidade, devemos considerar os objetos, coisas ou algo que seja relevante no levantamento dos dados.

ATRIBUTOS: Os atributos são propriedades utilizadas para descrever uma entidade, podemos afirmar que os Atributos são as características contidas nas Entidades, por exemplo, em uma Entidade Cliente, podemos relacionar os atributos CPF, NOME, IDADE, ENDEREÇO, BAIRRO, CIDADE etc.

RELACIONAMENTOS: Para essa organização sem perda de conteúdo, as entidades devem estar associadas, ligadas entre si. No Modelo Entidade Relacionamento (MER), não é permitido ligar uma entidade diretamente à outra. Quando há uma associação, ela é representada por um relacionamento. Quando há uma associação, ela é representada por um relacionamento e o relacionamento é apresentado na forma de um **losango** e, para a associação entre entidades, deve seguir a notação básica, que são entidades ligadas ao relacionamento por **linhas retas**.

RESTRIÇÕES E DOMÍNIO

As restrições de domínio especificam que o valor de cada atributo deve ser um valor atômico que, para cada atributo criado, devemos associar um tipo a ele. A essa associação, damos o nome de domínio.

Exemplos

Chave Primária

Para especificar uma chave primária, utiliza-se a cláusula PRIMARY KEY.

Exemplo 1:
id INT PRIMARY KEY

Exemplo 2:
PRIMARY KEY (grupo_fk,
contato_fk)

Chave Estrangeira

Para especificar uma chave estrangeira utiliza-se a cláusula FOREIGN KEY.

Exemplo:

```
CREATE TABLE email (
id INT PRIMARY KEY,
email VARCHAR(60) NOT NULL,
contato_fk INT,
FOREIGN KEY (contato_fk)
REFERENCES contato(id)
);
```

Chave Primária - Primary Key

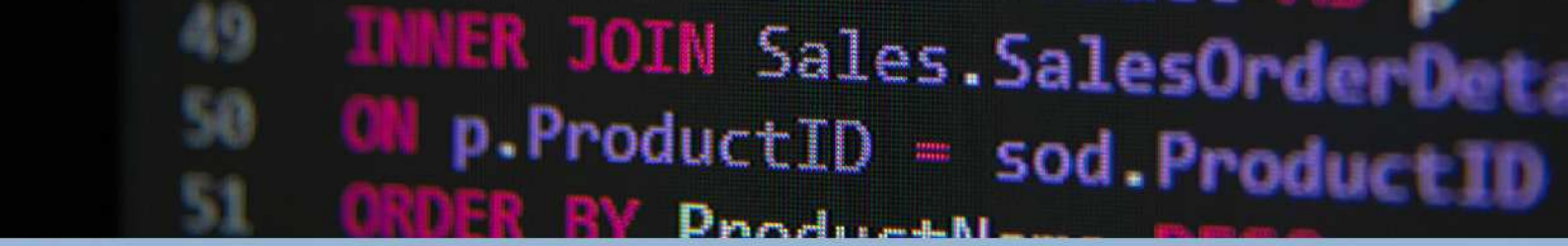
Toda entidade/Tabela necessita de um atributo que a identifique como única, a este campo damos o nome de Chave Primária ou Primary key.

É importante lembrar que este atributo deve ser único, ou seja, não pode permitir repetições.

Chave Estrangeria - Foreign Key

A chave estrangeira trata-se de um campo que aponta para a chave primária de outra tabela. Esta relação é importante para garantir a integridade de dados referenciais, pois, nesses casos, serão permitidos apenas valores que estiverem na base de dados.

É importante lembrar que, após estabelecer uma chave estrangeira, o atributo marcado não permitirá a exclusão, inserção ou modificação de dados em tabelas que estejam dependentes uma das outras.



SQL BÁSICO

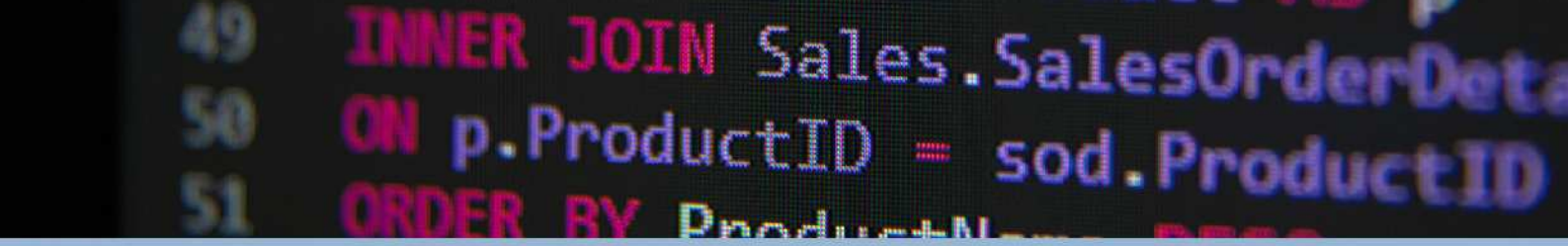
SQL é uma linguagem diferente das linguagens de programação que você provavelmente aprendeu até agora. Em qualquer curso de programação, costuma-se ensinar inicialmente linguagens de programação imperativas (como C, Pascal, Java ou Python), em que você é responsável por escrever os comandos na ordem de execução esperada.

A SQL é uma linguagem declarativa, pois nela define-se o que deve ser retornado como resultado do processamento, sem especificar o como isso será feito. Permita uma reflexão sobre a natureza declarativa da SQL. Na SQL, ao definirmos somente o que esperamos de resultado ao invés do como, permitimos que o SGBD decida como é que ele deve executar as instruções.

A SQL possui comandos tanto para a criação de definições de dados (criação de schemas) quanto para a execução de comandos de manipulação de banco de dados (consultas e atualizações). É uma linguagem bastante abrangente.

PRINCIPAIS COMANDOS SQL

Comando	Função
INSERT	Adicionar registros em uma tabela.
UPDATE	Atualizar registros já inseridos em uma tabela.
DELETE	Excluir registros de uma tabela.
SELECT	Selecionar registros de uma tabela (usado para consultas).
CREATE	Criar novas tabelas ou novos Bancos de Dados.
ALTER	Alterar um objeto já criado no Banco de Dados.
DROP	Excluir uma tabela ou um Banco de Dados.



PRINCIPAIS COMANDOS SQL

EXEMPLOS DE UTILIZAÇÃO

1) INSERT

- Usado para inserir ou adicionar dados (registros) em uma tabela do Banco de Dados.
- Exemplo: Inserção de um novo registro na tabela "Estudantes", sendo os termos "id", "nome" e "curso" os campos da tabela que vão receber essas novas informações.

```
INSERT into Estudantes (id, nome, curso) values (15, 'João', 'Análise e Desenvolvimento de Sistemas');
```

2) UPDATE

- Usado para fazer edições ou alterações em registros que já constam em uma tabela do BD.
- Permite corrigir ou complementar os dados, garantindo que o BD tenha sempre informações atualizadas.
- Exemplo: Atualização do valor do atributo "nome" da tabela "Estudantes", que antes era "João" e depois da atualização será "João da Silva".

```
UPDATE Estudantes SET nome = 'João da Silva' WHERE id = 15;
```

3) DELETE

- Usado para excluir dados ou registros de uma tabela.
- Exemplo: Exclusão do registro com "id" igual a 15 da tabela "Estudantes".

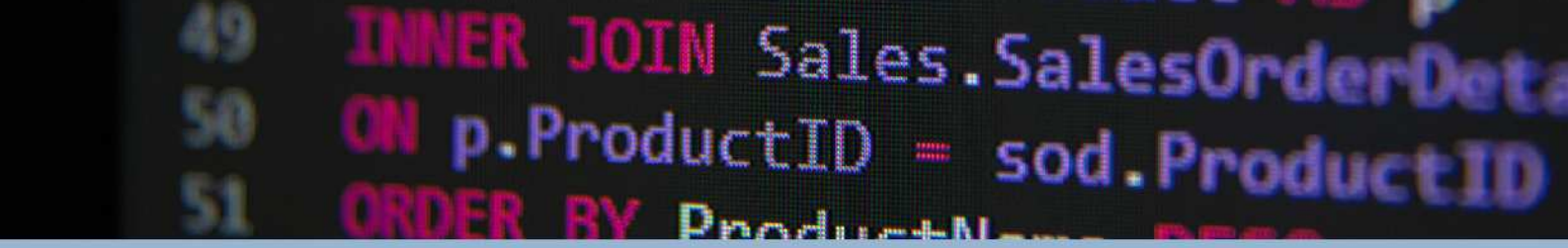
```
DELETE FROM Estudantes WHERE id = 15;
```

4) SELECT

- Usado para elaborar diversas consultas aos registros de uma tabela.
- Exemplo 1: pesquisa que retorna todos os campos de uma tabela:

```
SELECT * FROM estudantes;
```

- Exemplo 2: pesquisa que retorna apenas os campos da tabela "Estudantes" dos alunos que estão matriculados no curso de ADS.



PRINCIPAIS COMANDOS SQL

5) CREATE

- Usado para a criação de novas tabelas ou Banco de Dados.
- Exemplo 1: criação um BD chamado "Faculdade":

```
CREATE DATABASE Faculdade;
```

- Exemplo 2: criação uma tabela chamada "Estudantes" dentro do BD "Faculdade":

```
CREATE TABLE Estudantes (id INT PRIMARY KEY, nome VARCHAR(50), curso VARCHAR (100));
```

Observações:

- Ao criar a tabela também especificamos quais colunas ela terá e os tipos de dados que cada coluna pode receber.
- Nas colunas de tipo VARCHAR, também foi especificada a quantidade máxima de caracteres que cada campo pode comportar.
- A utilização de PRIMARY KEY ao definir o campo "id" indica que essa coluna é uma chave-primária, ou seja, o seu conteúdo é único e representa a identificação do registro.

6) ALTER

- Usado para alterar a estrutura de um objeto que já existe no BD.
- Exemplo: alteração da tabela "Estudantes" com a inserção de um novo campo (coluna) chamado "idade".

```
ALTER TABLE Estudantes ADD idade INT;
```

7) DROP

- Usado para deletar objetos de um Banco de dados.
- Exemplo 1: exclusão da tabela "Estudantes":

```
DROP TABLE Estudantes;
```

- Exemplo 2: exclusão do Banco de Dados "Faculdade":

```
DROP DATABASE Faculdade;
```


UniCesumar