**Bancos de Dados Relacionais**

**O que é um Banco de Dados?**

Um banco de dados é uma coleção organizada de informações ou dados. É estruturada e normalmente é armazenada de forma eletrônica em um computador.

**Tipos de Banco de Dados:**

* Relacionais/SQL
* Não Relacionais/NoSQL (Not OnlySQL)
* Orientado a Objetos
* Hierárquicos

**SGBD:**

Um banco de dados por si só não é auto controlável. Então temos Sistemas de Gerenciamento de Banco de Dados (SGBDs).

Os SGBDs fornecem para a gente um conjunto de ferramentas e recursos para que possamos criar nossos bancos, manipular nossos dados e administrar os nossos de banco de dados.

As funcionalidades básicas de um SGBD são:

* Create
* Read
* Update
* Delete

**Estrutura de um BD Relacional:**

**Diagrama

O conteúdo gerado por IA pode estar incorreto.**

O nosso banco de dados é formado por várias tabelas que podem se relacionar entre si. As tabelas, por sua vez, são compostas por colunas, que são onde colocamos as definições dos nossos dados. O registro (também conhecido como linha / tupla) são as informações armazenadas em si.

Os relacionamentos dependem de dois conceitos básicos: as chaves primárias, que são os identificadores únicos dos registros de nossa tabela e as chaves estrangeiras, que sinalizam que ali existe um relacionamento entre as tabelas.

**Características de um BD Relacional:**

* Relacionamento entre tabelas;
* Linguagem de Consulta Estruturada (SQL);
* Integridade Referencial;
* Normalização de dados;
* Segurança;
* Flexibilidade e extensibilidade;
* Suporte a transações ACID.

**O que é ACID:**

As transações ACID são um conjunto de propriedades de um banco de dados relacional que garante que as operações sejam executadas de forma segura e confiável, para que a gente tenha as informações totalmente consistentes.

* Atomicidade
* Consistência
* Isolamento
* Durabilidade

A atomicidade garante que todas as operações de uma transação sejam executadas com sucesso, ou que nenhuma delas seja executada. A consistência refere-se a capacidade de alterar de um estado consistente para outro estado consistente no banco de dados. O isolamento diz que cada transação ocorre de forma isolada, sem interferir nas transações concorrentes. Por fim, a durabilidade diz que uma vez que a transação é confirmada, essa alteração de informações seja permanente.

**SQL:**

O SQL é uma linguagem amplamente utilizada para interação com o banco de dados. Ele permite que nós, desenvolvedores, escrevemos consultas e comandos para poder criar nossos schemas, recuperar, inserir informação, atualizar informação ou até mesmo excluir informação do banco de dados.

Organização do SQL:

* DQL – Linguagem de Consulta de Dados (Data Query Language)
  + SELECT;
* DML – Linguagem de Manipulação de Dados (Data Manipulation Language)
  + INSERT, UPDATE, DELETE;
* DDL – Linguagem de Definição de Dados (Data Definition Language)
  + CREATE, ALTER, DROP;

Além dessas 3 principais, também temos:

* DCL – Linguagem de Controle de Dados (Data Control Language)
  + GRANT, REVOKE;
* DTL – Linguagem de Transação de Dados (Data Transaction Language)
  + BEGIN, COMMIT, ROLLBACK;

Sintaxe Básica: Nomenclatura

Essas regras de sintaxe vão depender de cada SGBD, porém de forma geral use:

* Os nomes devem começar com uma letra ou com um caractere de sublinhado (\_);
* Sensibilidade a letras maiúsculas e minúsculas.

**Modelo de Entidade-Relacionamento (MER) e Diagramas de Entidade-Relacionamento (DER):**

O modelo de entidade-relacionamento (MER) é representado através de diagramas chamados de diagrama de entidade-relacionamento (DER).

Entidades:

As entidades são a representação das nossas tabelas. Elas são nomeadas com substantivos concretos ou abstratos que representem de forma clara sua função dentro do domínio.

Atributos:

Os atributos são as características ou propriedades das entidades. Eles descrevem informações específicas sobre uma entidade.

Relacionamentos:

Os relacionamentos representam as associações entre entidades e descrevem como elas se relacionam entre si.

Cardinalidade:

A cardinalidade se refere a forma como as entidades se relacionam uma com as outras. Ela indica o número máximo de instâncias ou ocorrências que a gente pode ter de uma entidade associada a outra entidade.

Exemplos:

* Relacionamento 1..1 (um para um);
* Relacionamento 1..n (um para muitos);
* Relacionamento n..n (muitos para muitos).

**Normalização de Dados:**

A normalização de dados é um processo no qual se organiza e estrutura um banco de dados relacional de forma a eliminar redundâncias e anomalias, garantindo a consistência e integridade dos dados.

Formas Normais:

* 1FN: Atomicidade de dados
  + A 1FN estabelece que cada valor em uma tabela deve ser atômico, ou seja, indivisível. Nenhum campo deve conter múltiplos valores ou listas.
* 2FN
  + Para uma tabela estar na 2FN deve estar na 1FN.
  + A 2FN estabelece que todos os atributos não-chave devem depender totalmente da chave primária.
  + Se a sua tabela tem uma chave primária simples, não existe a possibilidade de termos dependência parcial e, portanto, já está na 2FN.
* 3FN
  + Para estar na 3FN, deve estar na 2FN.
  + A 3FN estabelece que nenhuma coluna não-chave deve depender de outra coluna não-chave.

**Consultas com Junções (Joins):**

As consultas Join são usadas para combinar dados de duas ou mais tabelas relacionadas em uma única consulta.

Tipos:

* INNER JOIN;
* LEFT JOIN ou LEFT OUTER JOIN;
* RIGHT JOIN ou RIGHT OUTER JOIN;
* FULL JOIN ou FULL OUTER JOIN.

INNER JOIN:

Retorna apenas as linhas que têm correspondência em ambas as tabelas envolvidas na junção. A junção é feita com base em uma condição de igualdade especificada na clausula ON.

Exemplo:

SELECT \*

FROM tabela1

INNER JOIN tabela2

ON tabela1.coluna = tabela2.coluna;

LEFT JOIN:

Retorna todas as linhas da tabela à esquerda da junção e as linhas correspondentes da tabela à direita. Se não houver correspondência, os valores da tabela à direita serão NULL.

Exemplo:

SELECT \*

FROM tabela1

LEFT JOIN tabela2

ON tabela1.coluna = tabela2.coluna;

RIGHT JOIN:

Retorna todas as linhas da tabela à direita da junção e as linhas correspondentes da tabela à esquerda. Se não houver correspondência, os valores da tabela à esquerda serão NULL.

Exemplo:

SELECT \*

FROM tabela1

RIGHT JOIN tabela2

ON tabela1.coluna = tabela2.coluna;

FULL JOIN:

Retorna todas as linhas de ambas as tabelas envolvidas na junção, combinando-as com base em uma condição de igualdade. Se não houver correspondência, os valores ausentes serão preenchidos com NULL.

Exemplo:

SELECT \*

FROM tabela1

FULL JOIN tabela2

ON tabela1.coluna = tabela2.coluna;

**Subconsultas ou Consultas Aninhadas:**

Elas permitem realizar consultas mais complexas permitindo que você use o resultado de uma consulta como entrada para outra consulta.

As subconsultas podem ser utilizadas em várias partes da consulta, como por exemplo dentro do SELECT, FROM, WHERE, HAVING e JOIN.

**Funções Agregadas:**

São funções que realizam algum tipo de pré-processamento ou cálculo em nossas colunas retornando algum valor.

Exemplos:

* COUNT: conta o número de registros;
* SUM: soma os valores de uma coluna numérica;
* AVG: calcula a média dos valores de uma coluna numérica;
* MIN: retorna o valor mínimo de uma coluna;
* MAX: retorna o valor máximo de uma coluna.

**Índices de Busca:**

Os índices de busca são estruturas de dados que aceleram as pesquisas e a recuperação de informações nos bancos de dados. Podem ser criados em uma ou várias colunas.

Análise do Plano de Execução:

Ela nos permite examinar as operações realizadas, as tabelas acessadas, os índices utilizados e outras informações importantes para identificar possíveis melhorias de desempenho.

Estrutura:

EXPLAIN

SELECT \*

FROM tabela

O comando EXPLAIN é responsável por retornar os dados da execução da query.

Possíveis retornos:

* select\_type: “SIMPLE”, “SUBQUERY”, “JOIN”.
* Table.
* type: “ALL”, “INDEX”, etc.
* possible\_keys: Os índices possíveis que podem ser utilizados na operação.
* key: O índice utilizado na operação, se aplicável.
* key\_len: O comprimento do índice utilizado.
* ref: As colunas ou constantes usadas para acessar o índice.
* rows: Diz quantas linhas o banco precisou processar para chegar no resultado.

Índice de Busca:

CREATE INDEX idx\_nome

ON tabela\_nome(coluna\_nome);