***DDL (Data DefinitionLanguage)***

Linguagem de Definição de Dados, define a estrutura dos dados e tabelas. Os comandos que são mais usados no DDL são os: *CREATE, ALTER, DROP, RENAME e TRUNCATE.*

***CREATE*** - É um comando usado para criar um Banco (*CREATE DB* – cria um banco de dados) e uma tabela (*CREATE TABLE* – cria uma tabela).

***ALTER TABLE*** – É um comando usado para adicionar, excluir ou modificar as colunas de uma tabela existente.

***DROP*** – É um comando usado para excluir dados. Ele também pode ser utilizado de três formas.

* *DROP INDEX* – Excluir um índice em uma tabela
* *DROP TABLE* – Excluir uma tabela
* *DROP DATABASE* – Excluir um banco de dados

***RENAME*** – São usadas para renome ar as tabelas.

***TRUNCATE*** – É usado para excluir dados de uma tabela, sem excluir a tabela em si.

***DML (Data Manipulation Language)***

Linguagem de Manipulação de Dados, usado para modificar o conteúdo das tabelas. Os comandos mais utilizados são os: *INSERT, UPTADE e DELETE*.

***INSERT*** – Insere linhas de dados em uma coluna

***UPTADE*** – Atualiza os dados de uma tabela.

***DELETE*** – Exclui dados de uma tabela.

***DCL (Data Control Language)***

Linguagem de Controle de Dados serve para controlar a parte de segurança de um banco de dados. ( dar e retirar permissões). Os comandos que são mais utilizados são os: *GRANT, REVOKE e DENY.*

***SQL GRANT*** – É usado para fornecer acesso ou privilégios sobre os objetivos de banco de dados para os usuários.

***REVOKE*** – É usado para remover direitos de acesso do usuário ou privilégios para os objetos de banco de dados.

***DENY*** – Nega explicitamente uma ou mais permissões à entidade de segurança.

***TCL (Transaction Control Language)***

Linguagem de Transação de Dados, são usados para gerenciar as mudanças feitas por instruções DML. Com isso ele permite que as declarações sejam agrupadas em transações lógicas. Os comandos mais usados são os: *COMMIT, SAVEPOINT e ROLLBACK.*

***COMMIT*** - É usado para salvar o seu trabalho feito.

***SAVEPOINT*** – Usado para identificar um ponto em uma transação, que com isso você possa efetuar um ROLLBACK mais tarde.

***ROLLBACK*** - É usado para restaurar um banco de dados ao original desde o último COMMIT.

***Banco de Dados Relacionais e Não Relacionais***

A maior e principal diferença entre um banco de dados relacional e não relacional é que:

O banco de dados relacional oferece maior consistência e confiabilidade, mas ele exige o relacionamento entre várias tabelas para que tenha acesso à informação.

O banco de dados não relacional, tem como sua vantagem uma escalabilidade maior, com a informação armazenada no mesmo registro.

***Suas características:***

*Banco de dados relacional*

A forma mais popular de armazenamento de dados no mercado. Foi criada por Edgar Frank Codd, em 1970.

O banco de dados relacional utilizada tabelas para a sua organização. Com ajuda da linguagem SQL, sua informações são guardadas nas suas linhas e colunas. Mas, para que seja possível inserir as coisas nas tabelas, é preciso projetar cada estrutura.

*Exemplos de banco de dados relacionais:*

* Oracle;
* MySQL;
* PostgreSQL;
* SQLite;
* DQL Server.

*Empresas que utilizam*

* Twitter (MySQL, FlockDB, Memcache, Cassandra, Gizzar);
* Quora (MySQL);
* Facebook (MySQL, Cassandra, Memcached;
* Google (BitTable, MySQL, MegaStore).

*Pontos positivos*

* Otimização em grandes conjuntos de ferramentas de informação;
* Acesso a Structured Query Language (SQL) ;
* Permitem que os dados sejam claros e organizados;

*Pontos negativos*

* Os bancos de dados relacionais vêm com restrições quanto ao que tipos de dados que podem armazenar.
* Complexiblidade quando ele é criado pela primeira vez;
* Podem rapidamente tornar-se complexa quando a quantidade de dados cresce e as relações entre as partes de dados tornam-se mais complicado

***Banco de dados não relacional***

Garante um gerencionamento mais eficiente, com a linguagem NoSQL.

Não exige a elaboração de um esquema antes de sua implementação, pois todas as informações ficam agrupadas em um único registro. Com isso, é necessário fazer cada um deles de modo que o banco de dados consiga entender.

*Exemplos de banco de dados não relacionais:*

* MongoDB;
* Cassandra;
* Voldemort;
* CouchDB;
* Riak.

*Empresas que utilizam:*

* Facebook (MySQL, Cassandra, Memcached);
* FourSquare (MongoDB);
* Twitter (MySQL, FlockDB, Memcache, Cassandra, Gizzar);

*Pontos positivos*

* Tem maior flexibilidade, sendo mais simples de gerir;
* Escabilidade mais simples;
* Não é necessário desenvolver um modelo de dados tão detalhado como no relacional – o que nos poupa mais tempo no desenvolvimento.

*Pontos negativos*

* Não tem uma comunidade ainda bem definida apesar de estar a crescer;
* Falta de padronização – ainda não existe uma linguagem padrão como o SQL;
* Faltam ferramentas de relatório para análises e testes de desempenho.