**Bancos de dados** são sistemas que permitem o armazenamento, organização e recuperação de informações de forma estruturada e eficiente. Eles são amplamente utilizados em muitas áreas, desde gerenciamento de negócios à pesquisa científica. Um banco de dados bem projetado pode melhorar significativamente a eficiência e a precisão das operações de uma organização.

Existem bancos de dados **relacionais** e **não-relacionais**. Os bancos de dados relacionais representam e armazenam dados em tabelas. Já os não-relacionais, também conhecidos como bancos de dados NoSQL (*Not Only SQL* - Não SQL), usam uma variedade de estruturas de dados, como documentos, grafos ou pares de chave-valor.

Python oferece vários pacotes e bibliotecas para trabalhar com bancos de dados, incluindo SQLite, MySQL, PostgreSQL, Oracle, MongoDB, entre outros. Um dos pacotes mais comuns usados para trabalhar com bancos de dados relacionais em Python é o pacote **sqlite3** que oferece suporte a bancos de dados **SQLite**. Este banco de dados é leve e incorporado que não exige um servidor separado para ser executado que já vem nativamente instalado no Google Colab.

Para trabalhar com esse banco podemos utilizar a **SQLAlchemy**, uma biblioteca de mapeamento objeto-relacional (ORM), que possibilita interagir com bancos de dados relacionais usando código Python. Ela fornece uma camada de abstração que permite aos desenvolvedores trabalhar com objetos Python em vez de lidar diretamente com as complexidades da linguagem **SQL** (*Structured Query Language* - Linguagem de consulta estruturada).

Uma das principais vantagens do uso de SQLAchemy é a capacidade de criar código mais legível e fácil de manter. Com SQLAlchemy, as operações do banco de dados são executadas usando métodos em objetos Python, tornando o código mais claro e menos propenso a erros.

Além disso, a SQLAlchemy oferece suporte a consultas complexas em bancos de dados, permitindo que pessoas desenvolvedoras extraiam facilmente informações relevantes de grandes conjuntos de dados. Isso é especialmente útil em aplicações que precisam lidar com grandes quantidades de dados.

Você pode encontrar mais detalhes sobre a biblioteca SQLAchemy na [documentação](https://www.sqlalchemy.org/).

**SQL** (*Structured Query Language* - Linguagem de consulta estruturada) é uma linguagem de consulta utilizada em bancos de dados relacionais para inserir, atualizar, consultar e gerenciar dados.

Para fazer tudo isso,existem as cláusulas SQL, que são componentes fundamentais das instruções SQL, permitindo especificar detalhes sobre como a consulta ou a operação de banco de dados devem ser executadas. As cláusulas são usadas para filtrar, classificar, agrupar e limitar os resultados das consultas.

As instruções SQL podem ser compostas por uma ou mais cláusulas que fornecem informações adicionais sobre o que a consulta deve fazer. As cláusulas mais comuns são:

* **SELECT**: especifica quais colunas devem ser selecionadas na consulta.
* **FROM**: especifica as tabelas do banco de dados que devem ser consultadas.
* **WHERE**: filtra os resultados da consulta com base em uma ou mais condições especificadas.
* **ORDER BY**: classifica os resultados da consulta em ordem crescente ou decrescente com base em uma ou mais colunas.
* **GROUP BY**: agrupa os resultados da consulta com base em uma ou mais colunas.
* **LIMIT**: limita o número de linhas retornadas pelos resultados da consulta.

Temos um exemplo simples de sintaxe SQL que usa o comando SELECT para consultar dados de uma tabela de colaboradores(as) em um banco de dados:

SELECT nome, sobrenome, salario

FROM colaboradores

WHERE departamento = 'vendas'

Nessa instrução SQL, a cláusula **SELECT** é usada para especificar as **colunas** que se deseja consultar da tabela "colaboradores", incluindo "nome", "sobrenome" e "salario". A cláusula **FROM** é usada para especificar a **tabela** que se deseja consultar, que é "colaboradores" neste caso.

A cláusula **WHERE** é usada para **filtrar** os **resultados** da consulta com base em uma condição especificada. Neste exemplo, a condição é departamento = 'vendas', o que significa que a consulta retornará apenas as pessoas colaboradoras que trabalham no departamento de vendas.

No próximo vídeo vamos aprender a usar algumas dessas cláusulas para realizar as primeiras consultas SQL na tabela que contém os dados de clientes de uma instituição financeira.