Diferentes fornecedores de software de banco de dados, como Oracle e Microsoft, adaptaram o SQL com suas extensões e modificações exclusivas;
Existem um núcleo comum de comandos SQL que é padrão em todos os sistemas;
O SQL oferece funcionalidades avançadas, como agregações, junções, subconsultas e transações, que permitem consultas complexas e a manipulação eficaz de dados

DDL (Data Definition Language - Linguagem de Definição de Dados): criação, modificação e exclusão de banco de dados e tabelas;

DML (Data Manipulation Language - Linguagem de Manipulação de Dados): recuperar, atualizar, inserir e excluir dados no banco de dados;

DCL (Data Control Language - Linguagem de Controle de Dados): Segurança e a autorização de acesso aos dados no banco de dados (GRANT e REVOKE). Comandos como GRANT (para conceder privilégios) e REVOKE (para revogar privilégios) são usados para garantir que apenas usuários autorizados possam acessar e modificar os dados.

A conexão perfeita
Primeiro passo para utilizar SQL é a conexão bem sucedida com SGBD.
Tanto o ODBC quanto o JDBC são tecnologias que proporcionam uma interface padronizada para que aplicativos possam interagir com bancos de dados de forma eficiente.
Uma das grandes vantagens dessas tecnologias é que permite que uma aplicação acesse diferentes Sistemas Gerenciadores de Banco de Dados sem a necessidade de recompilar o código.
No contexto de Python, para se comunicar com um SGBD, podemos usar bibliotecas específicas que incorporam os drivers de fornecedores. Isso permite a conexão e a execução de comandos SQL no banco de dados.

Para viabilizar essa comunicação entre a linguagem de programação e o RDBMS, fazemos uso de tecnologias como Open Database Connectivity (ODBC) e Java Database Connectivity (JDBC).
Tanto o ODBC quanto o JDBC oferecem uma maneira padronizada de os programadores acessarem os recursos do banco de dados a partir de uma Interface de Programação de Aplicativos (API). Uma das grandes vantagens dessas tecnologias é a possibilidade de que uma aplicação acesse diferentes Sistemas Gerenciadores de Banco de Dados sem a necessidade de recompilar o código. Isso se torna viável porque a comunicação direta com o RDBMS é realizada por meio de um software específico, chamado “driver”, responsável por traduzir as chamadas ODBC e JDBC para a linguagem compreendida pelo RDBMS.

SQlite

Uma poderosa biblioteca de banco de dados escrita em linguagem C, que oferece um mecanismo de banco de dados SQL Completo, embora compacto e de alta confiabilidade.
Diferentemente da maioria dos sistemas de gerenciamento de bancos de dados SQL, o SQLite opera sem a necessidade de um servidor separado.
Para os desenvolvedores que utilizam o Python, a linguagem possui um módulo integrado chamado "sqlite3" que permite a interação com o mecanismo do banco de dados SQLite.

Podemos inserir informações (create), ler (read), atualizar (update) e apagar (delete).
(i) estabelecer a conexão com um banco;
(ii) criar um cursor e executar o comando;
(iii) gravar a operação;
(iv) fechar o cursor e a conexão.

Para conhecer mais detalhes sobre SQL, faça a leitura do livro *Linguagem SQL: fundamentos e práticas*, cujo link de acesso está disponível a seguir.

Pandas I
DataFrames e Series: Estruturas de dados flexíveis, semelhante a tabelas(DF) e lista
Manipulação de dados: filtragem, seleção, ordenação, agrupamento e agregação
Leitura e Escrita de dados: CSV, Excel, SQL, HDF5 e muito outros, tornando-o uma ferramenta versátil para lidar com dados de diferentes fontes

Pandas II
Tratamento de Dados Ausentes: Simplifica o tratamento de dados faltantes
Visualização de Dados: integrado com bibliotecas de visualização, como Matplotlib e Seaborn
Integração com NumPy: construído sobre a biblioteca NumPy, combinação de cálculos e manipulação
Comunidade Ativa: tem uma comunidade de usuários e desenvolvedores ativa.

O principal parâmetro é "data", que pode conter um único valor, uma lista de valores ou um dicionário

Os outros parâmetros como "index", "dtype", "name", têm valores padrão predefinidos, tornando sua especificação opcional

Um recurso poderoso do pandas é a capacidade de ler dados estruturados e armazená-los em um DataFrame

1. **DataFrames e Series:** o DataFrame é uma estrutura bidimensional semelhante a uma tabela, enquanto a Series é uma estrutura unidimensional semelhante a uma lista ou matriz. Ambas as estruturas são altamente flexíveis e podem acomodar diversos tipos de dados.
2. **Manipulação de dados:** o pandas oferece uma ampla gama de funções e métodos para realizar tarefas comuns de manipulação de dados, como filtragem, seleção, ordenação, agrupamento e agregação.
3. **Leitura e escrita de dados:** o pandas suporta a leitura e escrita de dados em vários formatos, incluindo CSV, Excel, SQL, HDF5 e muitos outros, tornando-se uma ferramenta versátil para lidar com dados de diferentes fontes.
4. **Tratamento de dados ausentes:** a biblioteca simplifica o tratamento de dados faltantes, permitindo que os desenvolvedores preencham ou removam valores ausentes de forma eficaz.
5. **Visualização de dados:** embora o pandas seja mais conhecido por sua capacidade de manipular dados, também pode ser integrado a outras bibliotecas de visualização, como Matplotlib e Seaborn, para criar gráficos e visualizações informativas.
6. **Integração com NumPy:** o pandas é construído sobre a biblioteca NumPy, o que significa que você pode facilmente combinar as capacidades de NumPy para cálculos numéricos com as funcionalidades do pandas para manipulação de dados.
7. **Comunidade ativa:** o pandas tem uma comunidade de usuários e desenvolvedores ativa, o que resulta em suporte contínuo e atualizações regulares.

1. Para entender mais detalhes sobre o uso do pandas, sugiro que você visite a seguinte página: pandas. O site descreve a história e as funcionalidades dessa biblioteca.

2. Agora, para aprender mais sobre DataFrame, acesse o endereço a seguir: pandas.

Tipo de Dado	Descrição do Dado	Método para Leitura	Método para Escrita
Texto	CSV	read_csv	to_csv
Texto	Fixe-width texto file	read_fwf	
Texto	JSON	read_json	to_json

Texto	HTML	read_html	to_html
Texto	Latex		styler.to_latex
Texto	XML	read_xml	to_xml
Texto	Local Clipboard	read_clipboard	to_clipboard
Binário	MS Excel	read_excel	to_excel
Binário	OpenDocument	read_excel	
Binário	HDF5 Format	read_hdf	to_hdf
Binário	Feather Fomart	read_feather	to_feather
Binário	Parquet Format	read_parquet	to_parquet
Binário	ORC Format	read_orc	
Binário	MsgPack	read_msgpack	to_msgpack
Binário	Stata	read_stata	to_stata

Binário	SAS	read_sas	
Binário	SPSS	read_spss	
Binário	Python Pickle Format	read_pickle	to_picke
SQL	SQL	read_sql	to_sql
SQL	Google BigQuery	read_gbq	to_gbq

Matplotlib
A biblioteca Matplotlib desempenha um papel central na criação de gráficos em Python, sendo amplamente adotada em projetos de visualização de dados
Anteriormente, os cientistas tinham que gerar gráficos em outros softwares após extrair resultados de suas análises, tornando o processo incômodo
Assim, a biblioteca Matplotlib surgiu como uma solução eficiente para criar visualização em Python

Os gráficos desempenham o papel de narradores visuais, permitindo contar histórias por meio dos dados

import seaborn as sns
Uma característica notável da biblioteca Seaborn é seu repositório de conjuntos de dados prontos para uso, facilitando a exploração das funcionalidades
https://github.com/mwaskom/seaborn-data

TIMELINE

INTRODUÇÃO À ANÁLISE DE DADOS COM PYTHON