**CENTRO UNIVERSITÁRIO FACENS**

**CURSOS DE TECNOLOGIA – SPAS034TSN1**

BANCO DE DADOS

# TÓPICOS DA AULA

AGREGAÇÃO

* FUNÇÕES

AGRUPAMENTO

* FUNÇÕES
* HAVING E WHERE

EXERCÍCIOS

FERRAMENTAS

* WAMP
* ORACLE WORKBENCH

JUNÇÕES

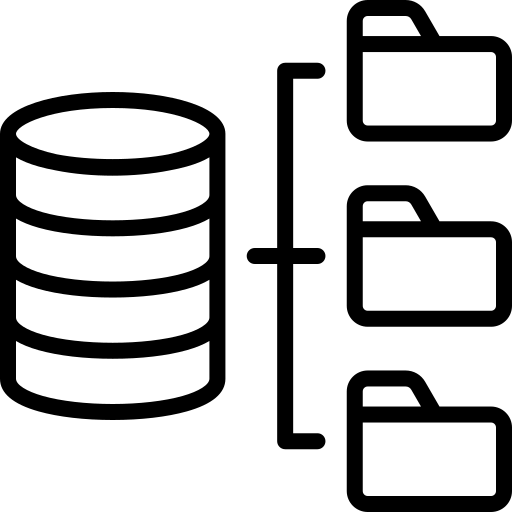
* INTRODUÇÃO
* CLÁUSULA JOIN

ALIASES

EXERCÍCIO

# AGREGAÇÃO

Em bancos de dados relacionais e na linguagem SQL, a agregação refere-se ao processo de calcular um único valor resumido a partir de um conjunto de valores de uma coluna ou expressão em um grupo de linhas. Esses valores resumidos são frequentemente usados para entender melhor os dados e realizar análises.



As funções de agregação resumem os dados de um conjunto em um único valor

escalar. A sintaxe é construída da seguinte forma: FUNCAO\_DE\_AGREGACAO(conjunto)

Exemplos:

## SELECT MAX (preco) FROM produtos; SELECT COUNT(id) FROM clientes; SELECT SUM(peso) FROM pacientes;

As funções de agregação resumem os dados de um conjunto em um único valor

escalar. A sintaxe é construída da seguinte forma: FUNCAO\_DE\_AGREGACAO(conjunto)

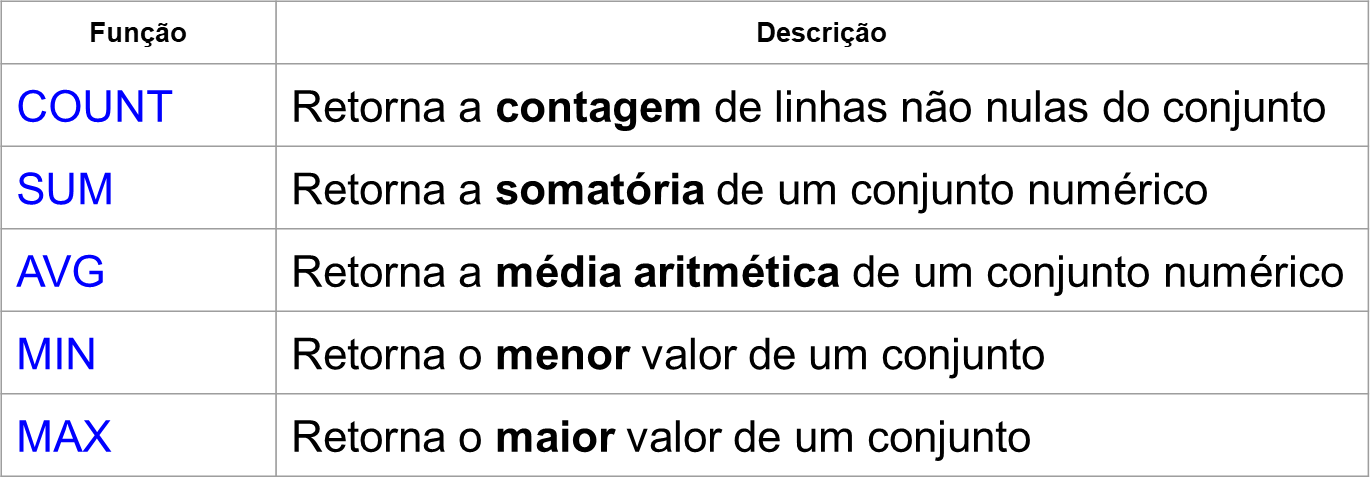
Exemplos:

## SELECT COUNT(id) FROM clientes;

**SELECT MAX (preco) FROM produtos;**

## SELECT AVG(idade) as idade\_media FROM pessoas; SELECT MIN(preco) as preco\_mínimo FROM produtos; SELECT SUM(peso) FROM pacientes;

As principais funções de agregação são:



A **agregação** é frequentemente usada em conjunto com a cláusula **GROUP BY**, como mencionado na resposta anterior, para calcular esses valores resumidos em grupos de dados com base em uma ou mais colunas.

A combinação de **agrupamento** e **agregação** permite a análise de dados de maneira mais granular e fornece informações valiosas sobre o conjunto de dados. Isso é fundamental para a tomada de decisões informadas e a geração de relatórios em bancos de dados relacionais.

O **agrupamento** em bancos de dados relacionais refere-se à capacidade de organizar registros ou dados com base em critérios específicos, a fim de realizar operações como **agregação**, **sumarização** e **análise de dados** de maneira eficiente. Isso geralmente é feito usando a cláusula **GROUP BY** em SQL, que permite agrupar linhas de uma tabela com base em uma ou mais colunas.

Muitas vezes é preciso obter informações organizadas em grupos. Por exemplo, agrupar todos os clientes por cidade, onde cada grupo é um resumo daqueles dados. Com isso, é possível saber quantos clientes há em cada cidade ou a média de suas idades, por exemplo.

A cláusula **GROUP BY** organiza linhas **agrupando** dados semelhantes, de maneira que o resultado é um resumo dessas linhas. Ela pode ser usada com funções de **agregação** como **SUM**, **COUNT** e **MAX**. São as funções de agregação que irão resumir os dados de cada grupo. Em outras palavras, **GROUP BY** permite segmentar os resultados das funções de **agregação**.

A sintaxe da cláusula GROUP BY é:

## SELECT COLUNAS FROM TABELAS GROUP BY COLUNAS;

* É usada em conjunto com o **SELECT**;
* Deve ser colocada após o **WHERE** (se houver) e
* Antes de **ORDER BY** (se houver)
* Todas as colunas da seleção que não são valores agregados devem constar na

cláusula **GROUP BY**.



## SELECT CIDADE, COUNT(ID) FROM CLIENTES GROUP BY CIDADE;

Resultado:



## SELECT COR, AVG(PESO) FROM ANIMAIS GROUP BY COR;

Resultado:

É possível filtrar os resultados do agrupamento por meio da cláusula HAVING. Exemplo:

## SELECT COR, AVG(PESO) FROM ANIMAIS GROUP BY COR HAVING

**AVG(PESO) > 15**

Resultado:



O comando **WHERE** é usado na cláusula **SELECT** para filtrar as linhas antes que elas sejam **agrupadas** (no caso de uso com **GROUP BY**) ou antes que as operações de **agregação** sejam aplicadas (no caso de uso com funções de **agregação**).

Ele é usado para filtrar as linhas individuais da tabela de origem antes que qualquer **agrupamento** ou **agregação** ocorra. Pode ser usado em consultas que não envolvem **agrupamento** ou **agregação**, simplesmente para filtrar linhas com base em condições específicas.

## SELECT nome, idade FROM pessoas WHERE idade >= 18;

Neste exemplo, o **WHERE** filtra as linhas onde a idade é maior ou igual a 18 antes

que qualquer outra operação seja realizada.

O comando **HAVING** é usado após o agrupamento (**GROUP BY**) e a aplicação de funções de **agregação**, para filtrar os grupos resultantes com base em condições que envolvem os resultados das funções de **agregação**. Ele é usado para filtrar grupos de linhas (resultantes do **agrupamento**) com base em critérios de **agregação**.

Geralmente é usado em consultas que envolvem **agrupamento** e **agregação**, onde você deseja filtrar grupos de resultados em vez de linhas individuais.

## SELECT departamento, AVG(salario) as media\_salario FROM funcionários GROUP BY departamento HAVING AVG(salario) > 50000;

Neste exemplo, o **HAVING** filtra os grupos de departamentos cuja média de salário é superior a $50,000, após a aplicação do **GROUP BY** e do cálculo da média de salário.

Crie uma tabela Produtos com os seguintes itens id\_produto; nome; preço; estoque; perecível; marca; nacionalidade;

Atribua a cada campo seu respectivo tipo;

Os itens: nome; preço; estoque; perecível não poderão receber valores nulos; O id\_produto deve ser utilizado como chave-primaria;

Após a criação insira cinco produtos, todos devem ter seus respectivos campos preenchidos;

Verifique se todos os dados foram inseridos;

Ao término inicie a segunda etapa.

Gere um relatório informando quantos produtos estão cadastrados; Gere um relatório informando o preço médio dos produtos;

Selecione a média dos preços dos produtos em 2 grupos: perecíveis e não

perecíveis;

Selecione a média dos preços dos produtos agrupados pelo nome do produto; Selecione a média dos preços e total em estoque dos produtos;

Selecione o nome, marca e quantidade em estoque do produto mais caro;

Selecione os produtos com preço acima da média;

Selecione a quantidade de produtos de cada nacionalidade.

Execute toda a atividade no WorkBench da Oracle;

Comente o código esclarecendo o que está sendo feito no bloco de código;

Crie um repositório remoto (GITHUB);

Envie somente o script SQL para o repositório;

Crie o ReadMe com base no modelo disponibilizado em aula; Tire um print da tela do WorkBench e insira no ReadMe; Envie na atividade somente o link do repositório criado.

Está atividade corresponde as atividades complementares 2; A data de entrega será para próxima aula;

Utilize o tempo disponível durante a aula para execução da atividade;

Não será aceito o exercício após a data de entrega;

Execute cada uma das atividades solicitadas, a falta de algum dos itens será descontado em nota;

Caso execute em seu computador pessoal, realize a instalação dos itens a seguir

para conclusão da atividade.

O WAMP é uma plataforma de desenvolvimento web popular que simplifica a configuração de um ambiente de desenvolvimento web local no Windows. O nome "WAMP" é um acrônimo que representa os principais componentes do ambiente: Windows, Apache, MySQL (ou MariaDB), e PHP.



**Apache**: O WAMP inclui o servidor web Apache, que é um dos servidores web mais populares do mundo. O Apache é responsável por servir páginas da web e processar solicitações HTTP.

**MySQL ou MariaDB**: O WAMP instala um sistema de gerenciamento de banco de dados (SGBD) MySQL ou MariaDB, que é usado para armazenar dados em bancos de dados relacionais. Você pode escolher entre MySQL ou MariaDB durante a instalação.

**PHP**: O WAMP também inclui uma versão do PHP, que é uma linguagem de programação amplamente usada para desenvolvimento web. O PHP é usado para processar dados no servidor e gerar conteúdo dinâmico para páginas da web.

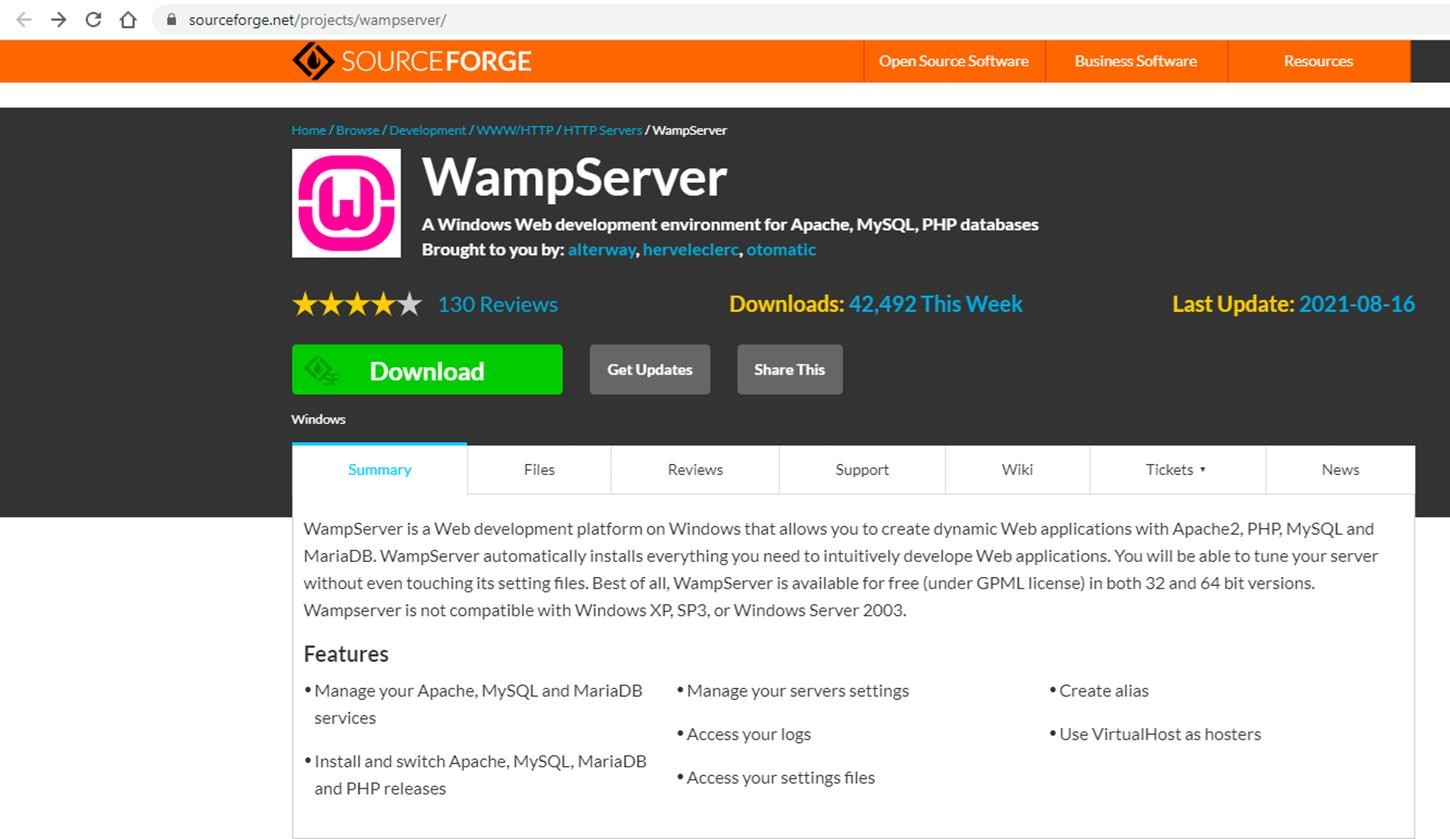
**Desenvolvimento Web Local:** O principal objetivo do WAMP é permitir que os desenvolvedores web criem, testem e depurem seus sites e aplicativos em um ambiente local antes de implantá-los em servidores de produção. Isso ajuda a evitar problemas e erros em produção, economizando tempo e recursos.

**Configuração Simples:** O WAMP facilita a configuração de um ambiente de desenvolvimento web no Windows, eliminando a necessidade de instalar e configurar manualmente o Apache, MySQL (ou MariaDB), PHP e outras ferramentas relacionadas.

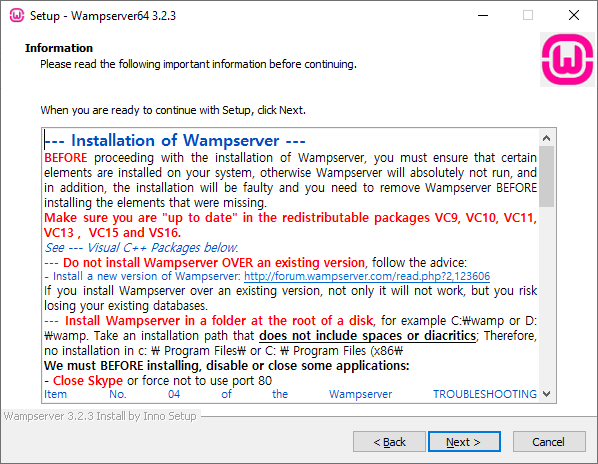
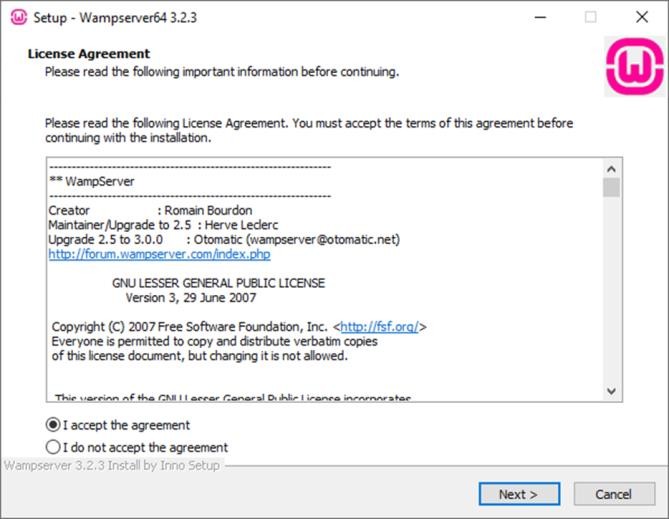
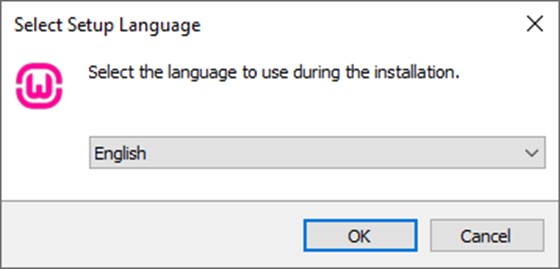
**Compatibilidade com WordPress e Outros CMS:** Muitos sistemas de gerenciamento de conteúdo (CMS), como o WordPress, Joomla e Drupal, são compatíveis com o WAMP, tornando-o uma escolha popular para desenvolver sites baseados em CMS.

**Teste de Aplicativos Web:** Os desenvolvedores podem usar o WAMP para testar aplicativos da web localmente, simular cenários de produção e garantir que seus aplicativos funcionem conforme o esperado antes de implantá-los em servidores web ao vivo.

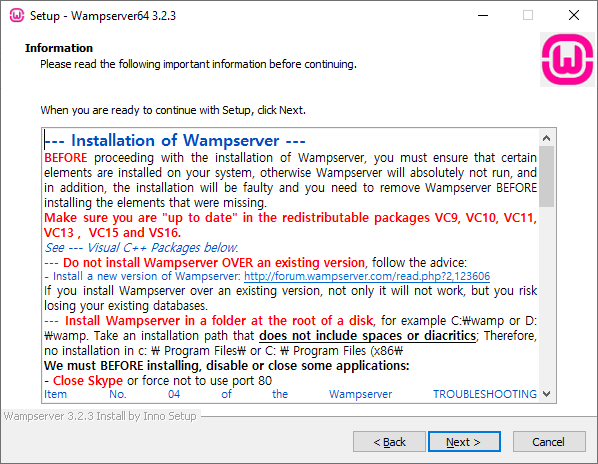
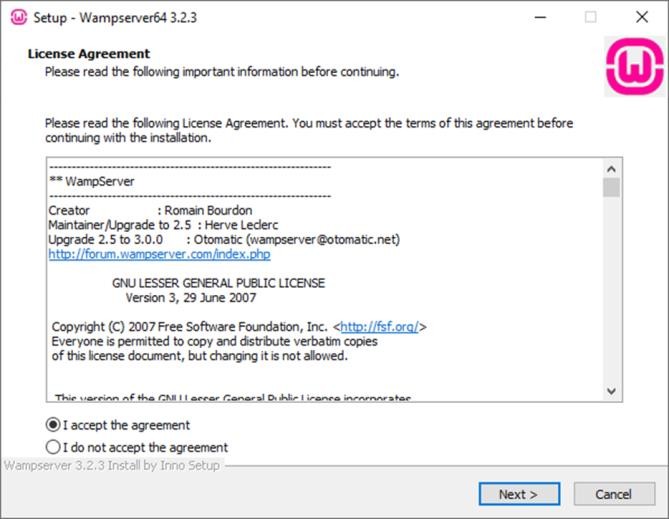
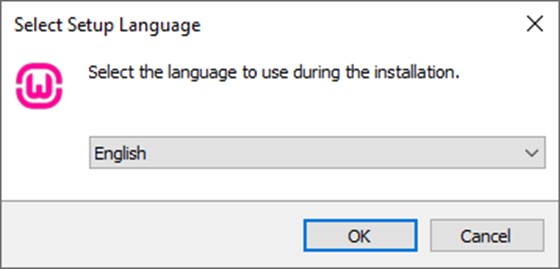
Faça o download no link a seguir: https://sourceforge.net/projects/wampserver/



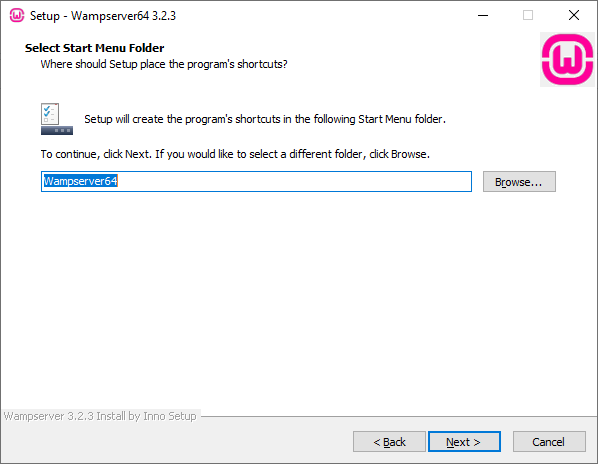
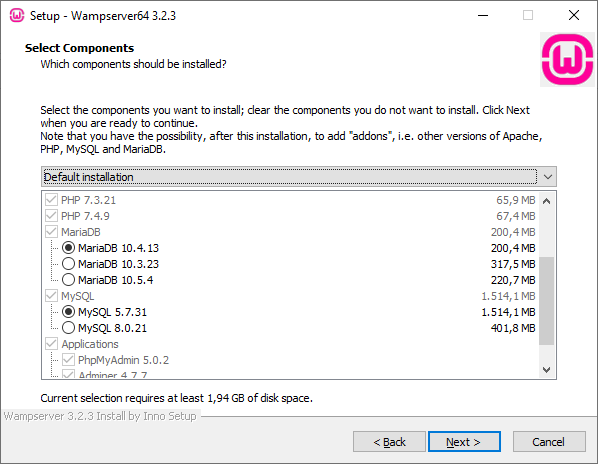
Siga a sequencia de telas, conforme abaixo:



Siga a sequencia de telas, conforme abaixo da esquerda para direita:



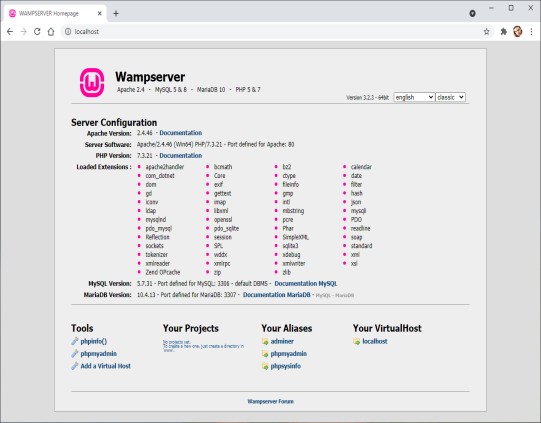
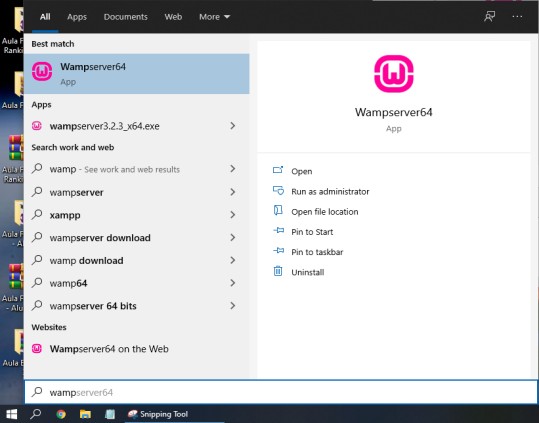
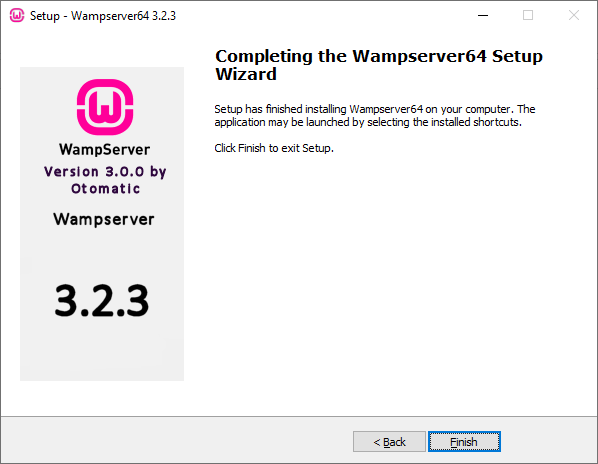
Siga a sequencia de telas, conforme abaixo da esquerda para direita:



A sequencia corresponde ao local de instalação; Itens a serem instalados; e pasta

de inicialização do WAMP;

Siga a sequencia de telas, conforme abaixo da esquerda para direita:



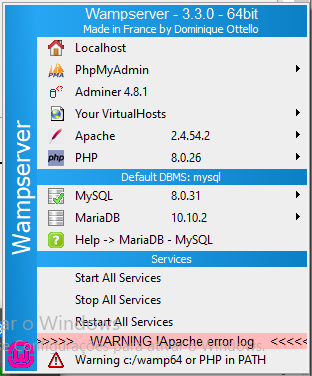
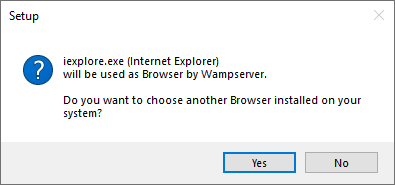
Ao término é informado a versão instalada; para acessar utilize a busca do

Windows ou botão Iniciar do S.O.; para testar digite localhost no seu navegador.

Durante a instalação o WAMP, poderá informar que utilizara o navegador Edge para testes;

Para saber se o WAMP está funcionando verifique sempre a caixa ao lado do

relógio do Windows;

Ao acessar pelo ícone (figura meio) você terá acesso aos serviços rodando (figura direita).

O Oracle SQL Developer (também conhecido como Oracle SQL Developer Data Modeler quando se trata da parte de modelagem de dados) é uma ferramenta de desenvolvimento de banco de dados fornecida pela Oracle Corporation. Não deve ser confundido com o MySQL Workbench, que é uma ferramenta separada para o sistema de gerenciamento de banco de dados MySQL.



**Desenvolvimento SQL**: O Oracle SQL Developer permite que os desenvolvedores escrevam e execute consultas SQL em bancos de dados Oracle. Ele fornece um ambiente de desenvolvimento interativo para criar, testar e otimizar consultas SQL e instruções PL/SQL.

**Modelagem de Dados**: A parte de modelagem de dados do SQL Developer, chamada Oracle SQL Developer Data Modeler, permite que os desenvolvedores criem e mantenham modelos de dados relacionais e dimensionais. Isso é útil para projetar e documentar a estrutura de banco de dados antes de implementá-la.

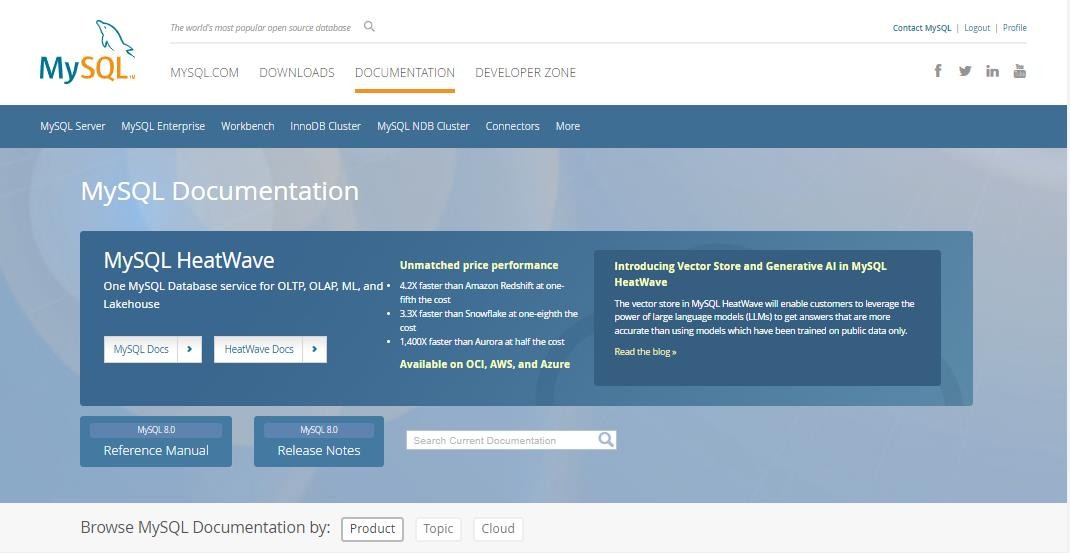
**Administração de Banco de Dados**: O SQL Developer inclui recursos de administração de banco de dados que permitem aos administradores de banco de dados realizar tarefas de gerenciamento, como criar e gerenciar usuários, tabelas, índices, backups e muito mais.

**Integração com o Oracle Cloud**: O SQL Developer oferece integração com serviços em nuvem da Oracle, como o Oracle Cloud Autonomous Database, permitindo que os desenvolvedores gerenciem e acessem bancos de dados Oracle na nuvem.

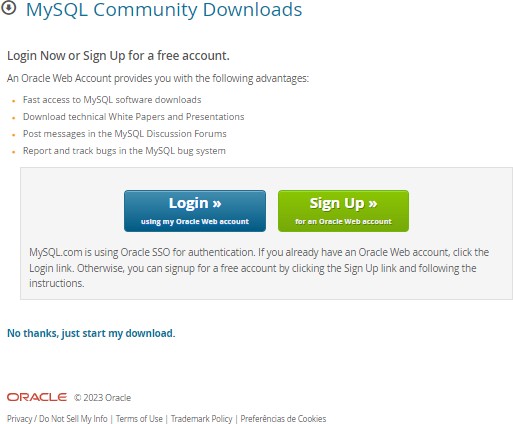
**Ferramentas de Migração**: Ele inclui ferramentas de migração que ajudam a migrar dados de outras fontes para o banco de dados Oracle.

**Controle de Versão**: O SQL Developer possui recursos de controle de versão que permitem que os desenvolvedores trabalhem com sistemas de controle de versão, como o Git, para gerenciar código SQL e objetos de banco de dados.

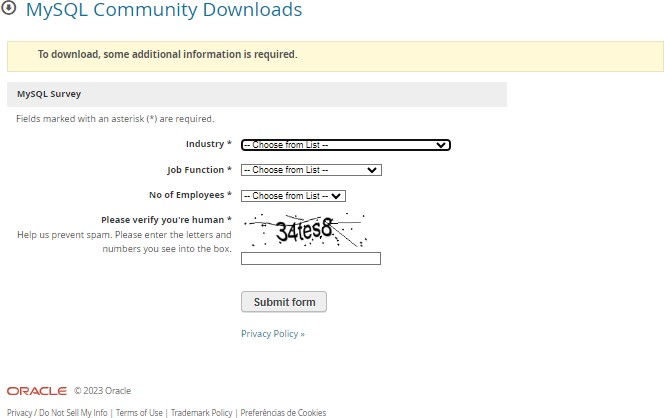
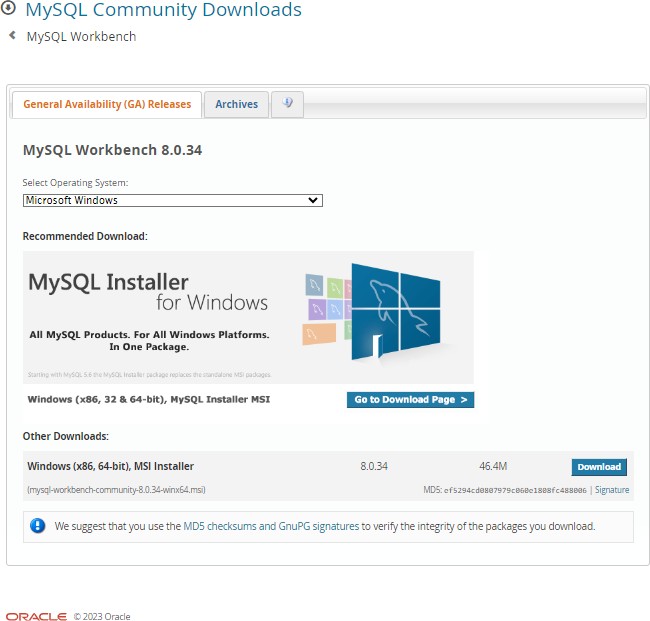
**Suporte a Diversos Bancos de Dados**: Embora seja desenvolvido pela Oracle para bancos de dados Oracle, o SQL Developer também oferece suporte para outros sistemas de gerenciamento de banco de dados, como o Microsoft SQL Server e o MySQL, tornando-o uma ferramenta versátil para desenvolvedores que trabalham com diferentes sistemas.

O primeiro passo é acessar o website da Oracle, destinado para o MySQL, conforme a figura abaixo:

Após o acesso faça o login ou crie uma conta na Oracle, o processo é gratuito;



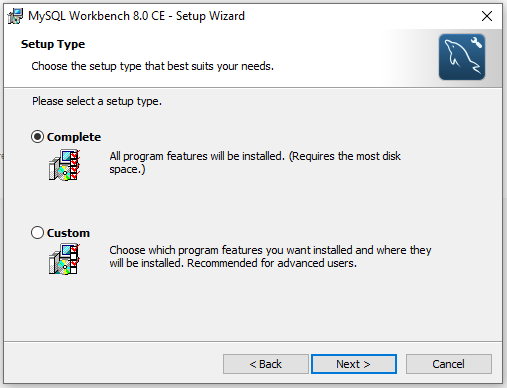
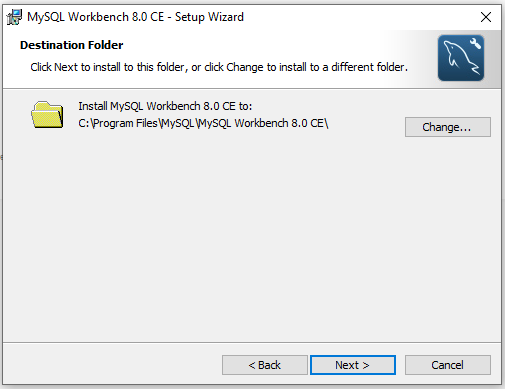
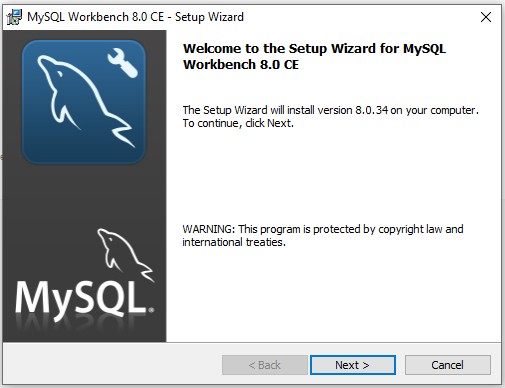
O processo é necessário para realizar o download da IDE;

Devidamente logado ao sistema da Oracle, siga os passos abaixo (esquerda para direita);

Busque na área de download pelo MySql WorkBench; selecione a versão para o

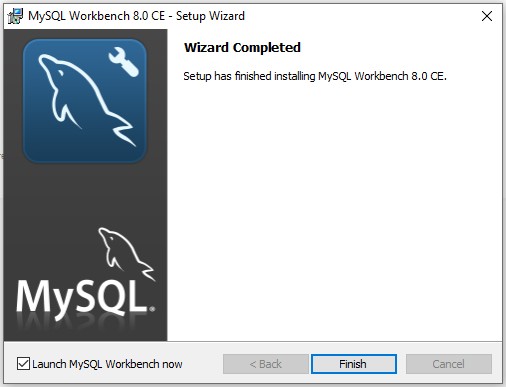
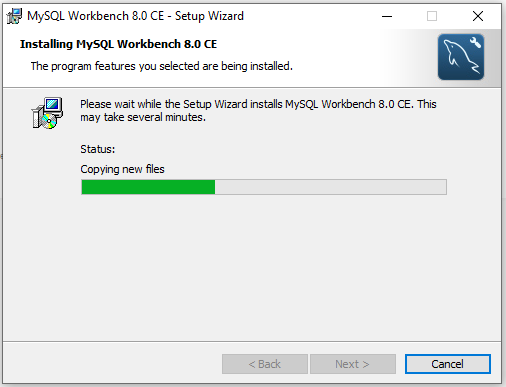
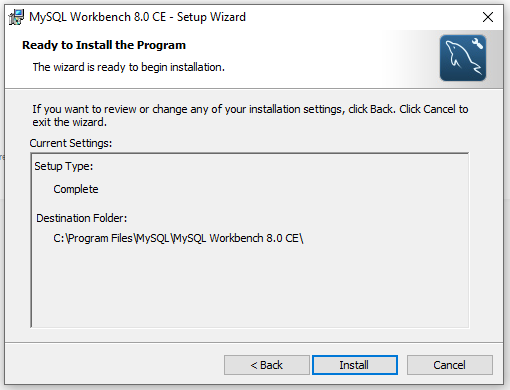
seu S.O.; Faça o download e Explique o motivo para o download (fins estudantis).

Siga a sequencia de telas da esquerda para direita, conforme abaixo:

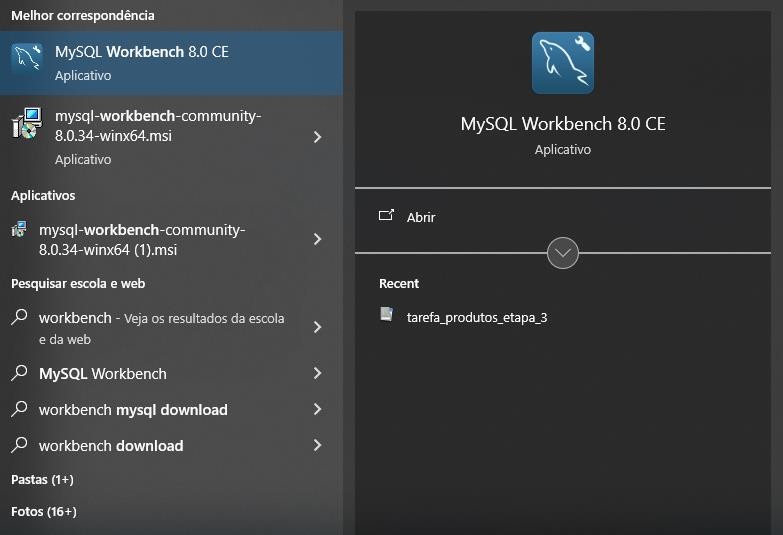


Ao término do download, acesso o ícone baixado; Passe pela tela inicial; verifique a pasta ser criada para o programa de instalação; escolha a opção complete para instalar.

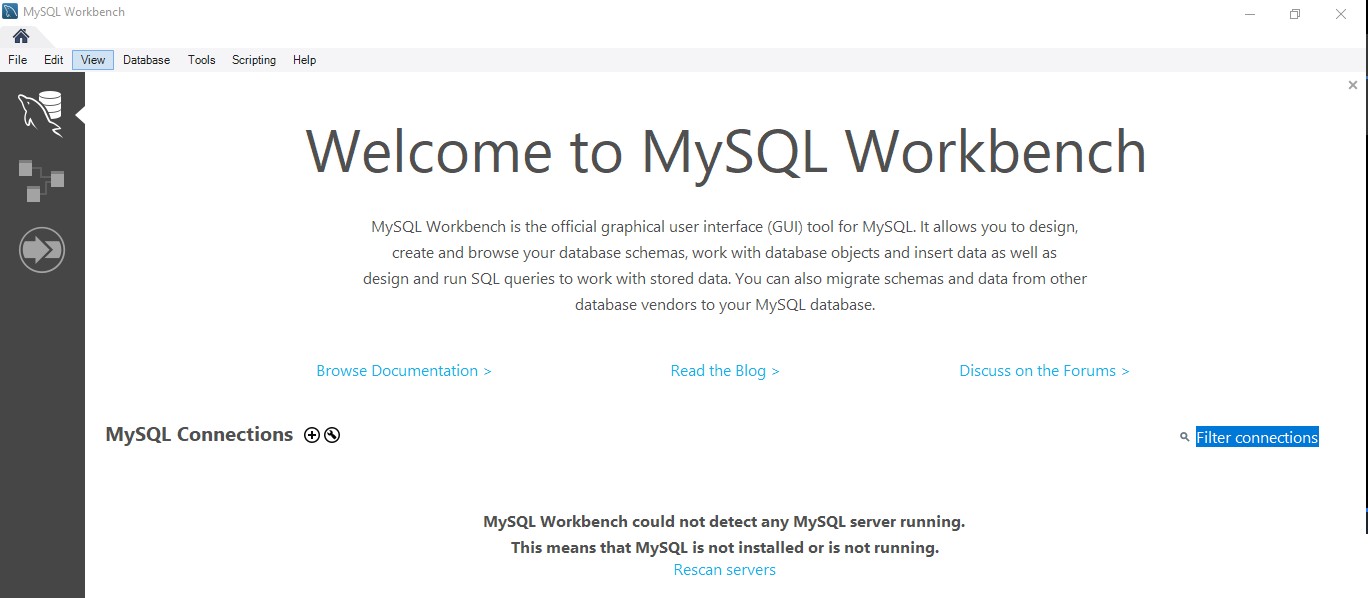
Siga a sequencia de telas da esquerda para direita, conforme abaixo:



Veja as informações da instalação; aguarde para que a instalação seja concluída e finalize a instação.

Após a instalação, acesse a IDE por meio da busca do Windows ou pelo botão iniciar:

O ambiente irá carregar conforme a imagem abaixo:



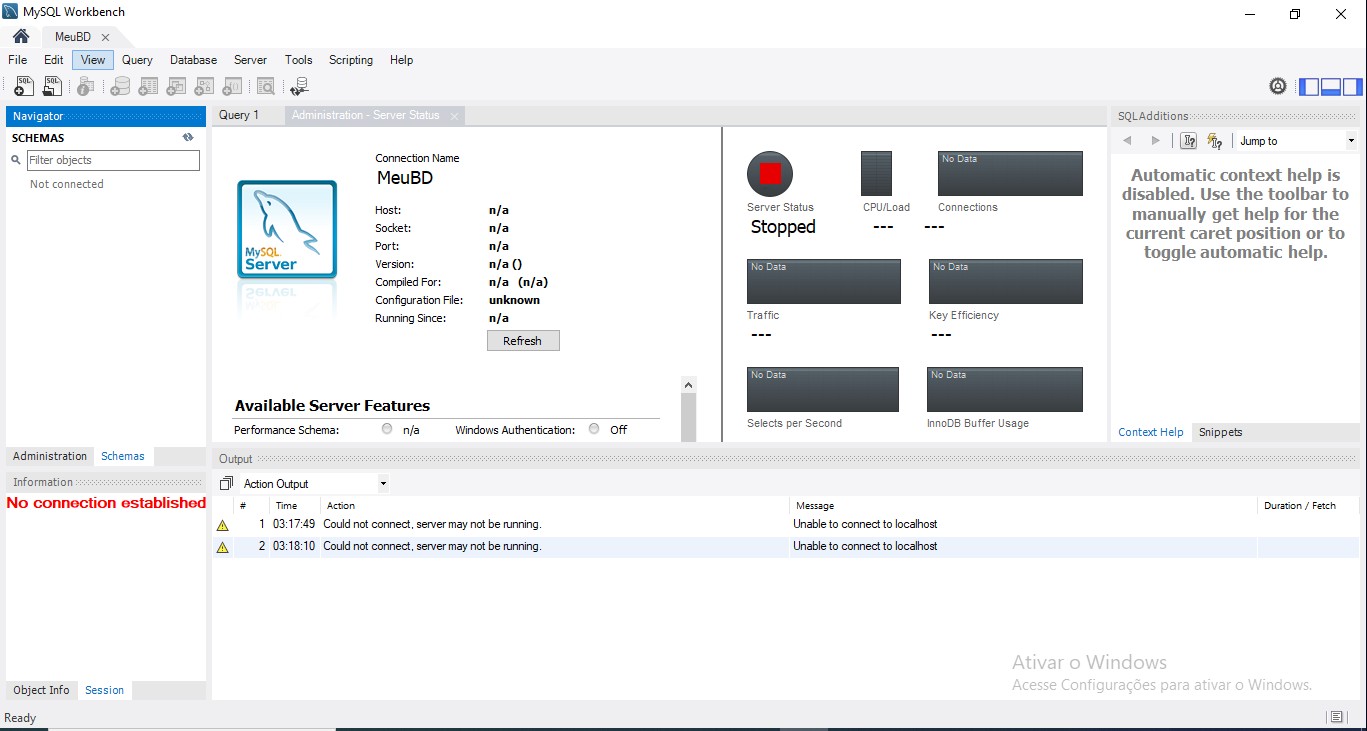
**4**

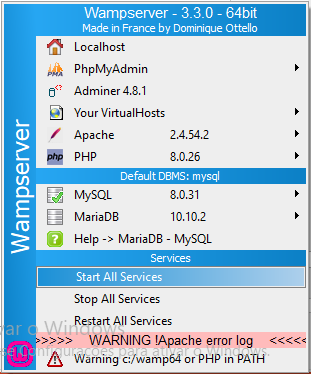
**3**

**2**

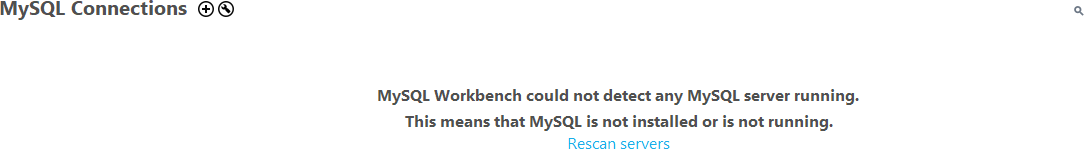
**1**

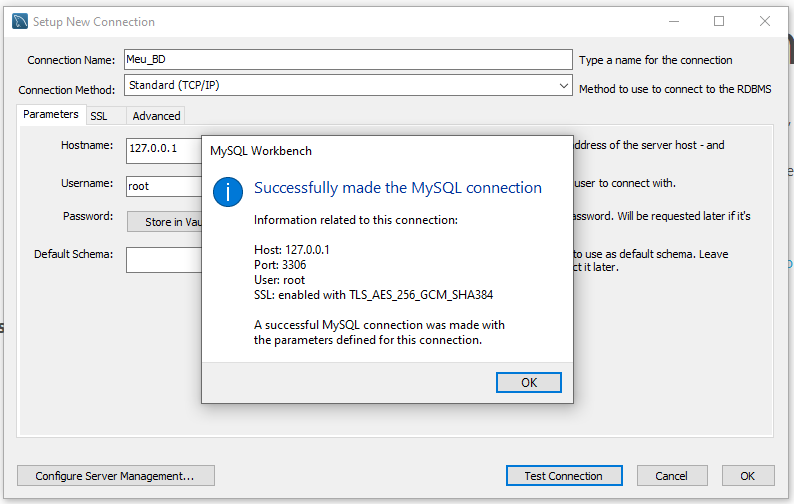
1) Barra de Menu; 2) Barra Lateral; 3) Área de Trabalho; 4) Área de Conexões

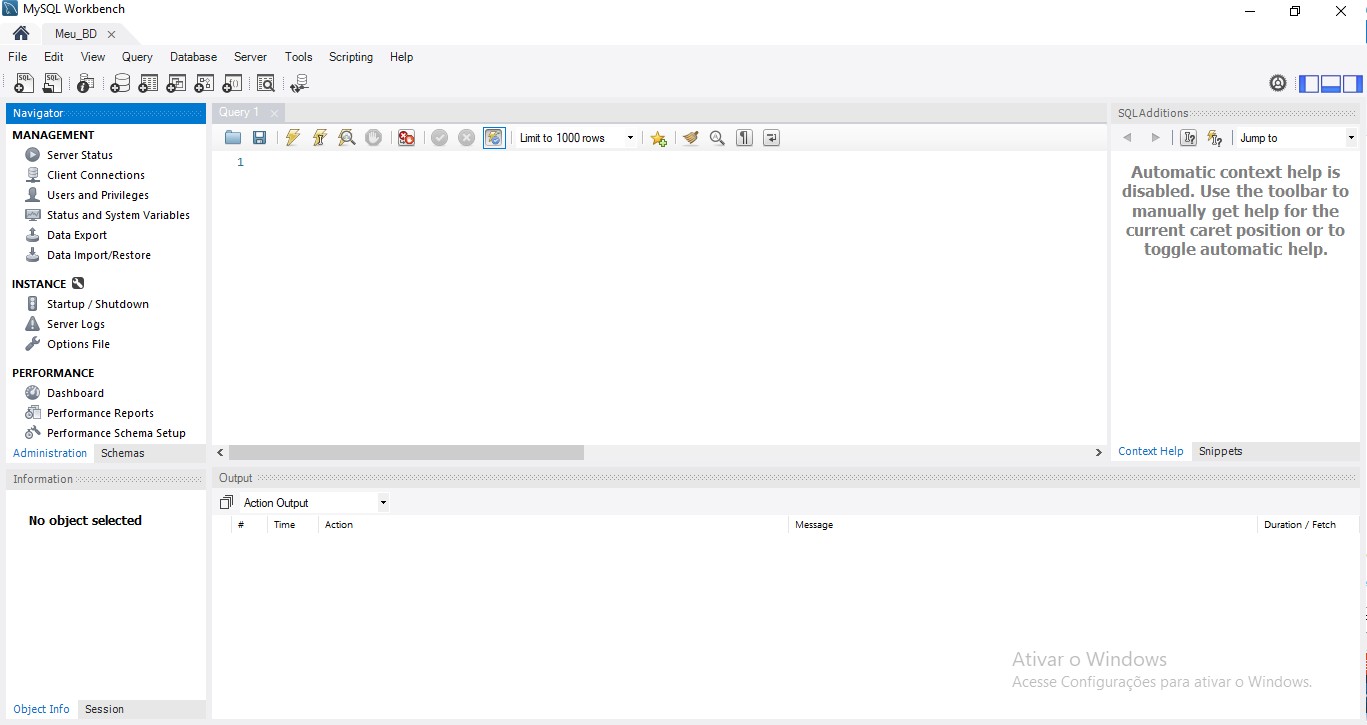
Caso, você não tenha instalado o WAMP, ou não esteja iniciado o funcionamento do mesmo, ao criar um modelo não teremos uma conexão a uma base de dados, conforme a figura abaixo:

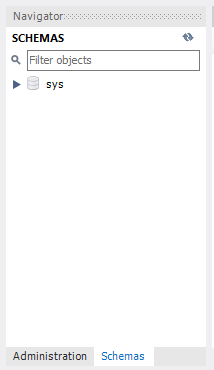
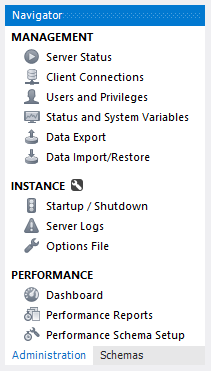
Para criar uma conexão, primeiro certifique que o WAMP esteja em funcionamento (figura 1) e acesse o ícone de + (figura 2).

**Figura 1**

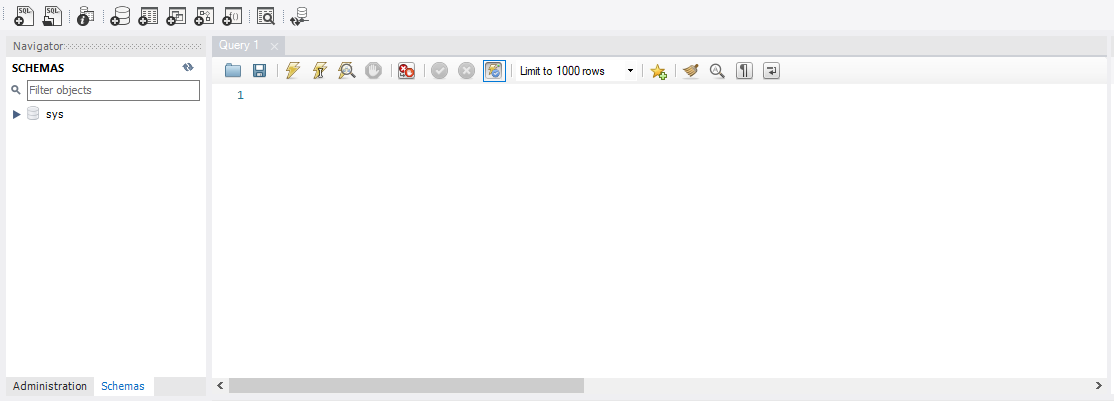


Coloque um nome (**Connection Name**) para sua conexão e teste seu funcionamento com o botão **Test Connection**. Caso esteja correto irá aparecer a mensagem conforme a figura abaixo:

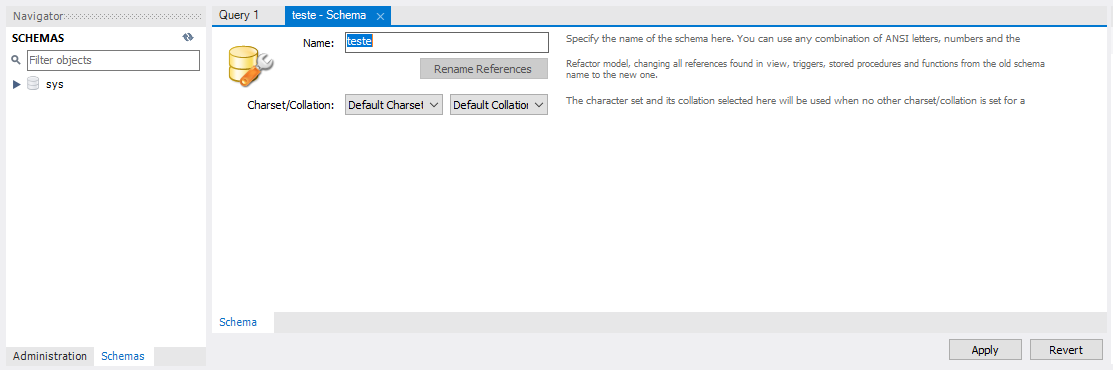
Ao acessar, você estará dentro da conexão criada e estará no ambiente abaixo:

Para desenvolver o exercício, acesse a barra **Navigator** (figura a esquerda) e selecione o item no rodapé da barra com o nome **Schemas**. Ao selecionar será apresentado os itens existentes (figura a direita).

Crie um novo Schemas, para isso clique no ícone em destaque na figura abaixo:

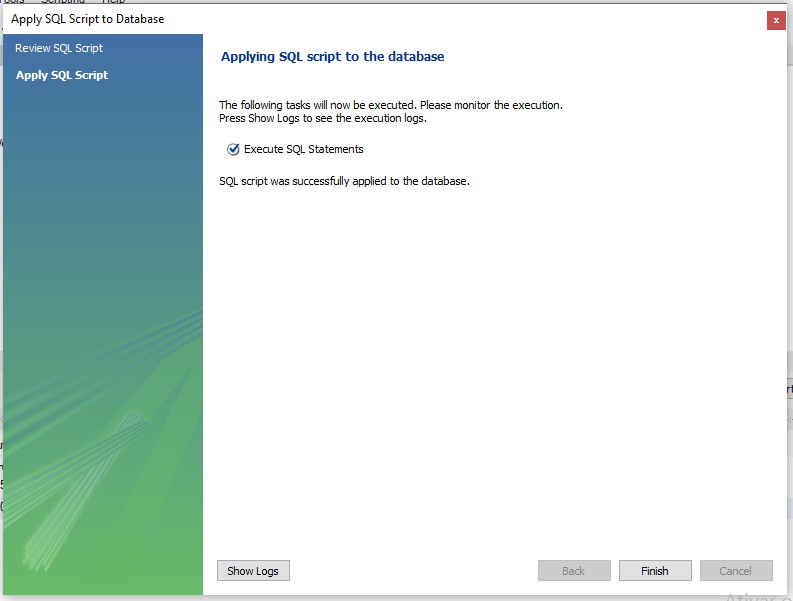
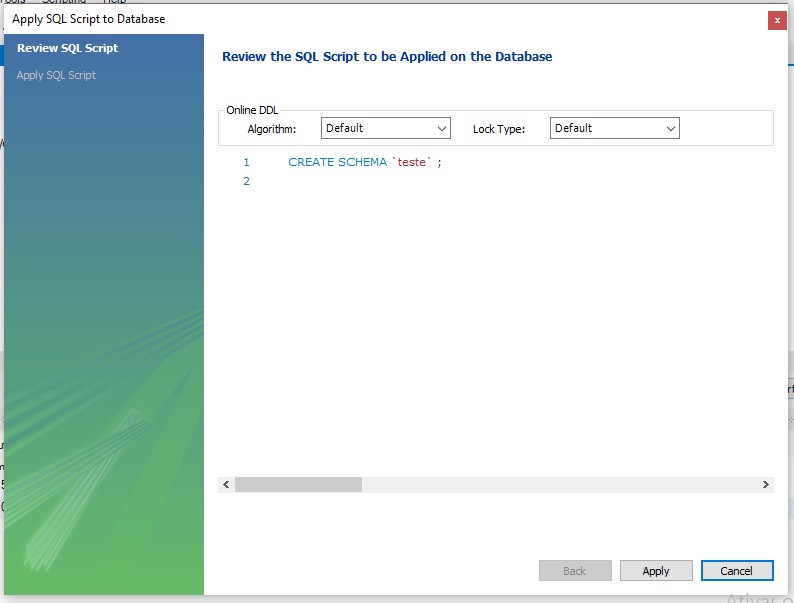


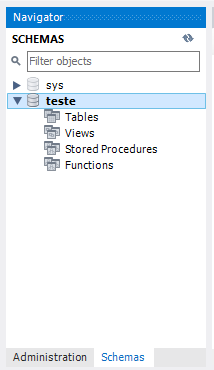
Coloque um nome para o seu SCHEMA e clique em Apply.

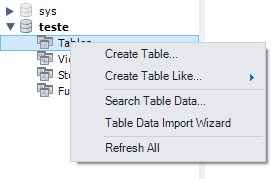


O script em SQL será criado, mas basta confirmar o procedimento com Apply.

Finalize o procedimento apertando Finish.

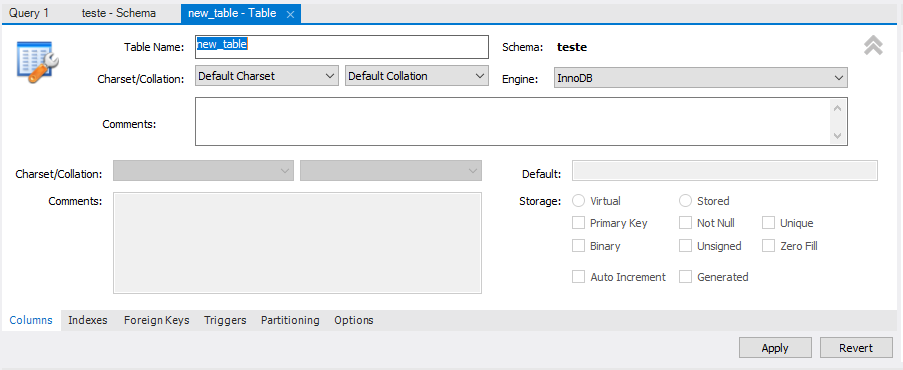


Clique duas vezes em cima do SCHEMA, recém criado e desta forma sua identificação (nome), estará em negrito. Agora, você poderá utilizar a interface para criar as tabelas necessárias seja por meio da interface ou por Script SQL. Com o botão da direita do mouse em cima da opção table, você poderá adicionar uma tabela (figura abaixo).

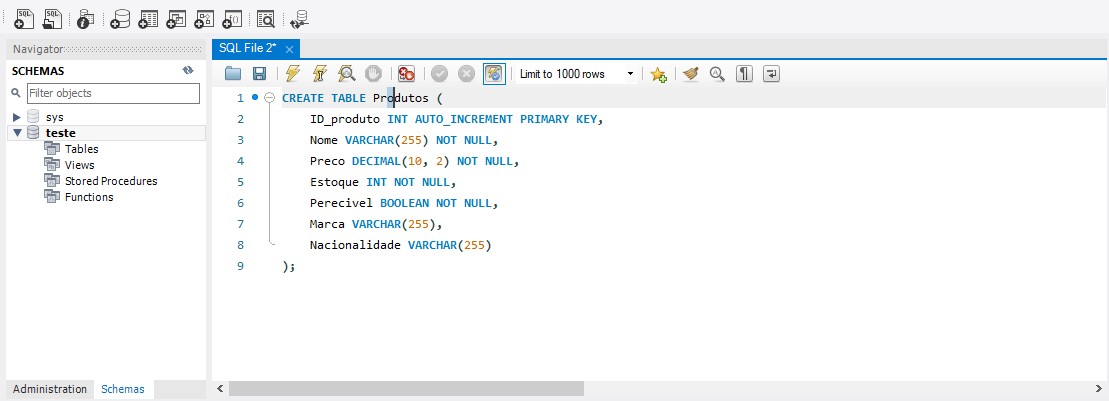


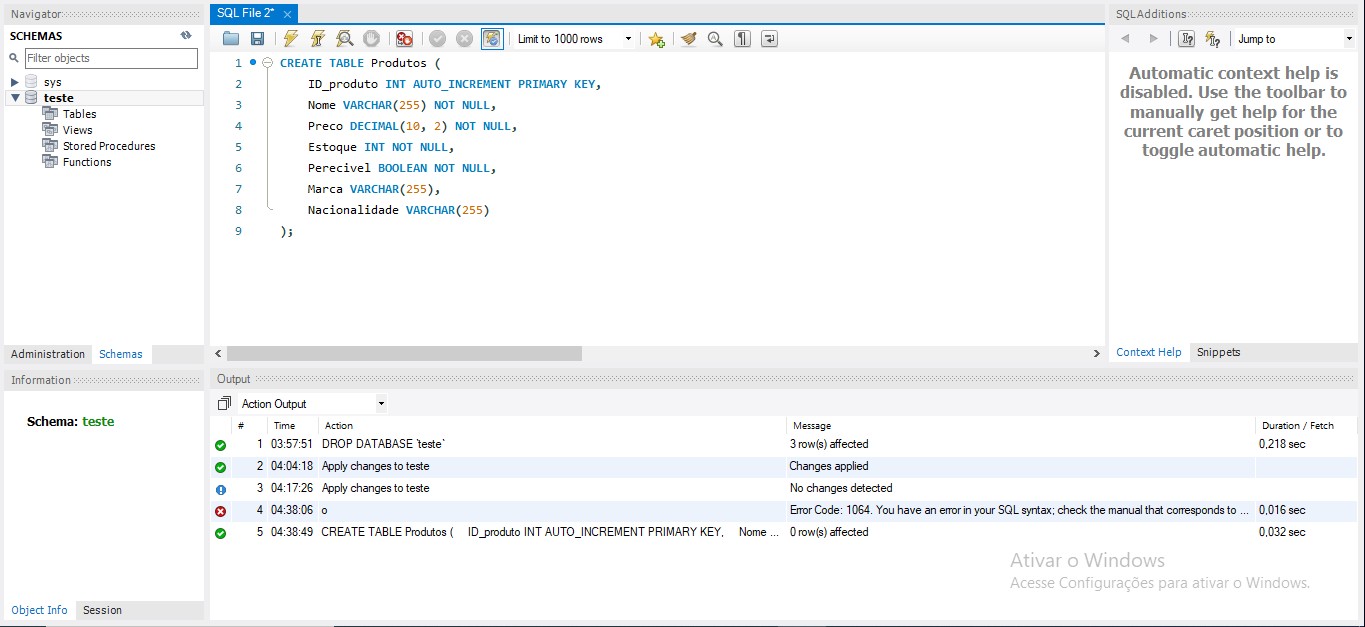
Será aberto um editor para inserir cada campo da tabela a ser criada conforme a

figura abaixo.

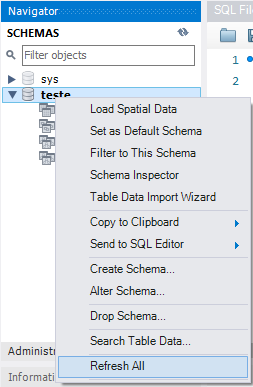
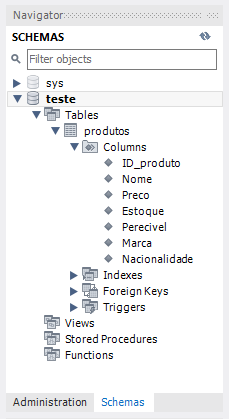


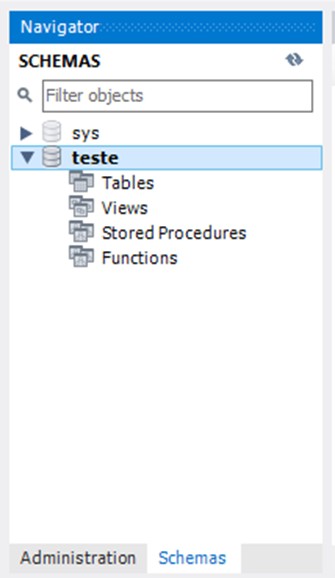
Para criar uma tabela por meio de script, acione o itens em destaque na figura abaixo, digite o código no local indicado e execute, por meio do ícone semelhante a um trovão.



Após executar, observe que a cada etapa realizada será adicionado no log de informações conhecido como **Output**. Ao término, será necessário atualizar a SCHEMA para aparecer a tabela criada.

Ao término, será necessário atualizar a SCHEMA para aparecer a tabela criada. Clique com o botão da direita em cima do SCHEMA teste e de um **Refresh All.** Logo irá aparecer a tabela criada.

No **MySQL Workbench**, um "**schema**" é um conceito que se refere a um espaço de nome lógico que organiza e contém objetos de banco de dados, como tabelas, visões, procedimentos armazenados e outros elementos. Em essência, um **schema** é uma estrutura de contêiner que ajuda a organizar e gerenciar objetos de banco de dados relacionados.

**Organização Lógica:** Um schema é uma maneira de organizar objetos de banco de dados de forma lógica e agrupá-los com base em sua função, aplicação ou relacionamento. Isso ajuda a manter a estrutura do banco de dados mais organizada e legível.

**Isolamento de Objetos:** Os objetos dentro de um schema são isolados de objetos em outros schemas. Isso significa que você pode ter tabelas com o mesmo nome em schemas diferentes sem conflitos, pois cada tabela é única dentro do contexto de seu schema.

**Permissões e Controle de Acesso:** Schemas também são usados para controlar o acesso e as permissões. Você pode conceder ou negar privilégios de acesso a um schema específico para usuários ou funções de banco de dados.

**Utilização em Múltiplos Bancos de Dados:** No MySQL Workbench, você pode criar schemas em vários bancos de dados. Isso é útil quando você deseja organizar objetos em diferentes bancos de dados em grupos lógicos separados.

**Padrão de Nomenclatura:** É comum usar schemas como parte de um padrão de nomenclatura para nomear objetos de banco de dados. Por exemplo, você pode criar um schema chamado "financeiro" para conter tabelas relacionadas a finanças.

**Facilita a Manutenção:** A utilização de schemas pode facilitar a manutenção e a gestão de bancos de dados complexos, pois torna mais claro onde cada objeto pertence e como eles estão relacionados.

**Consultas Qualificadas por Schema:** Você pode qualificar consultas SQL usando o nome do schema para especificar a qual schema os objetos pertencem. Por exemplo: SELECT \* FROM schema\_nome.tabela\_nome.

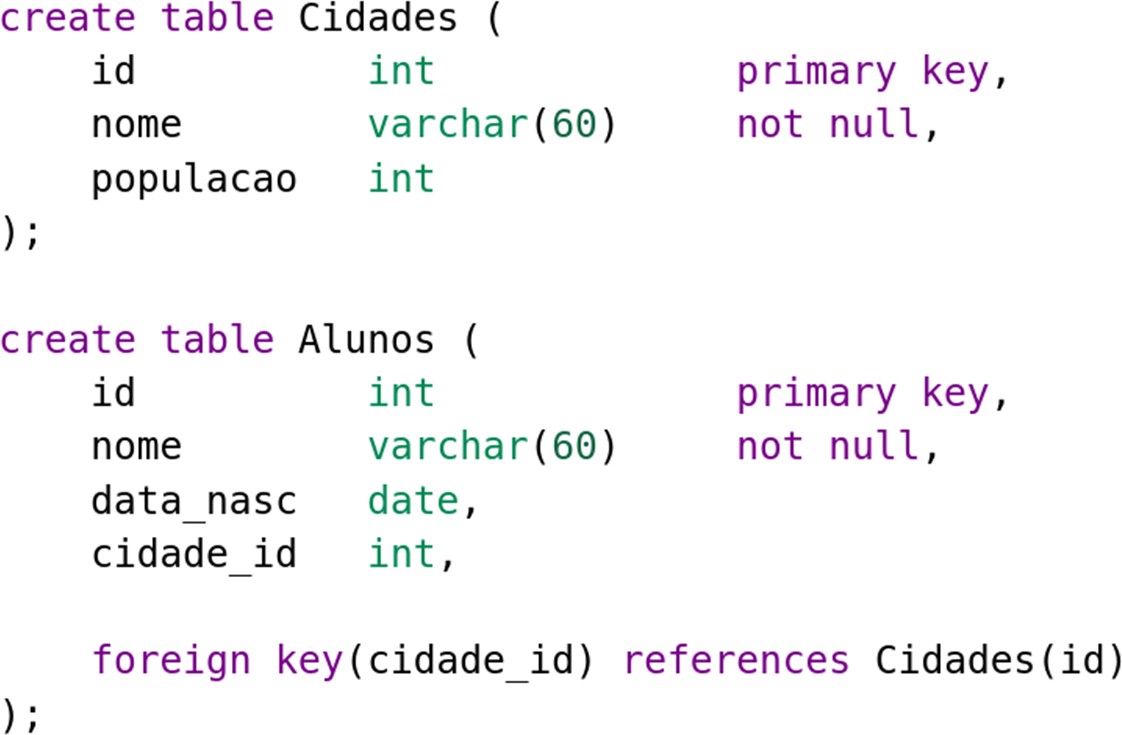
No **modelo relacional**, as **entidades** tornar-se-ão **tabelas**. Onde cada tabela

armazena dados distintos dos demais.

Assim, um banco de dados relacional pode ter **várias tabelas**, de modo que os

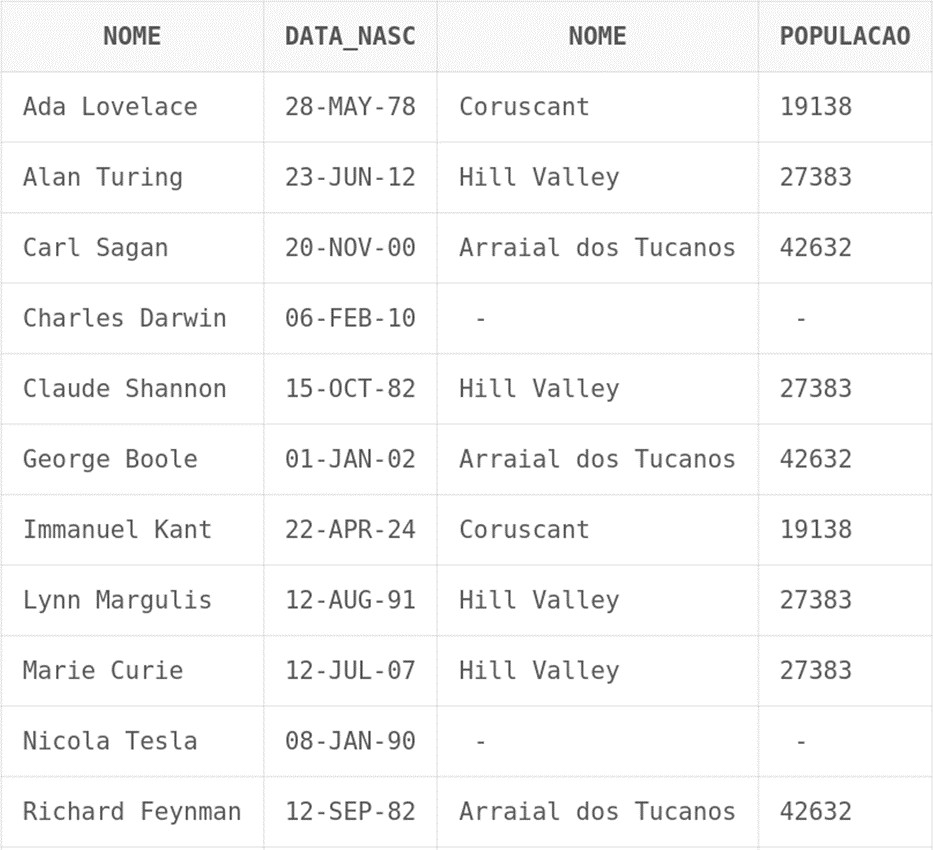
**dados** de um mesmo contexto ficam **esparsos.**

Acompanhe o exemplo no próximos slides:





Embora os dados devam estar em tabelas separadas (Alunos e Cidades), o

usuário final precisa visualizar dados consolidados.

A cláusula JOIN em SQL é usada para combinar dados de duas ou mais tabelas

em um único conjunto de resultados com base em uma condição especificada.

Ela é uma parte fundamental da linguagem SQL e permite que você relacione registros em diferentes tabelas, criando relações entre elas.

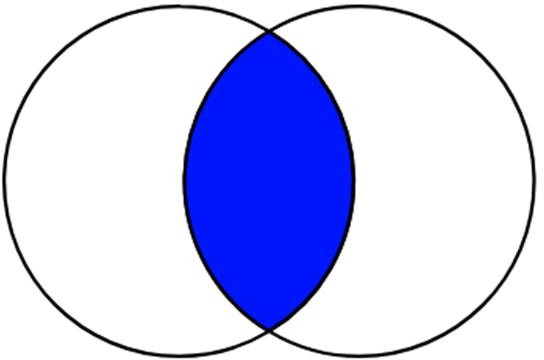
A cláusula JOIN é especialmente útil quando você deseja recuperar informações

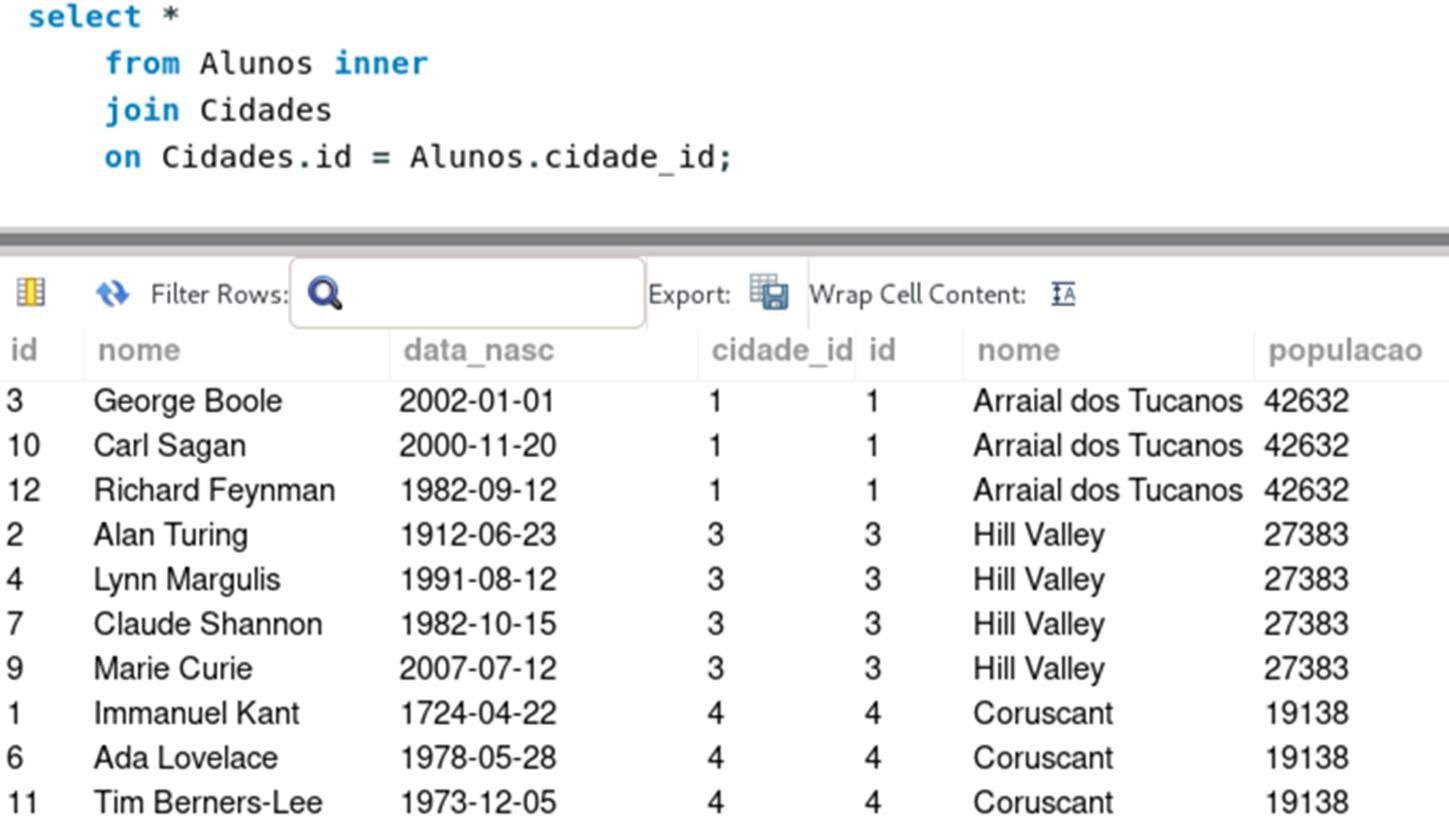
de tabelas relacionadas e apresentá-las em um único resultado.

**Tabelas:** Em SQL, você pode ter várias tabelas em um banco de dados, e cada tabela contém dados específicos. As tabelas podem estar relacionadas de várias maneiras, e a cláusula JOIN é usada para unir essas tabelas com base nas colunas que elas têm em comum.

**Chave de Junção:** Uma chave de junção é a coluna ou conjunto de colunas usadas para combinar as linhas de duas tabelas. Geralmente, as chaves de junção têm valores correspondentes nas tabelas que estão sendo unidas.

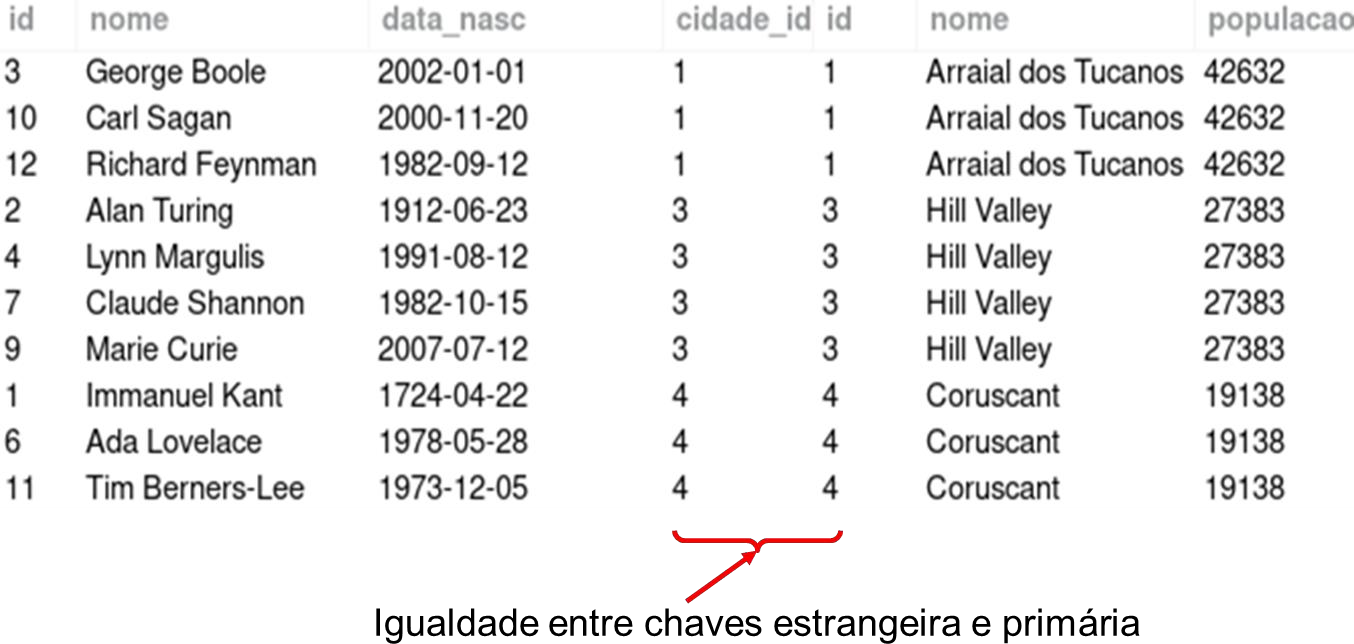
**INNER JOIN:** Retorna apenas os registros que têm correspondências em ambas as tabelas. É o tipo de JOIN mais comum. Por padrão, o resultado da seleção é a intersecção entre os registros conforme o critério estabelecido. Retorna somente as linhas que casam em ambas as tabelas



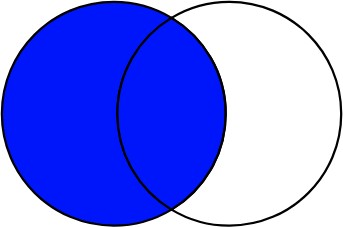


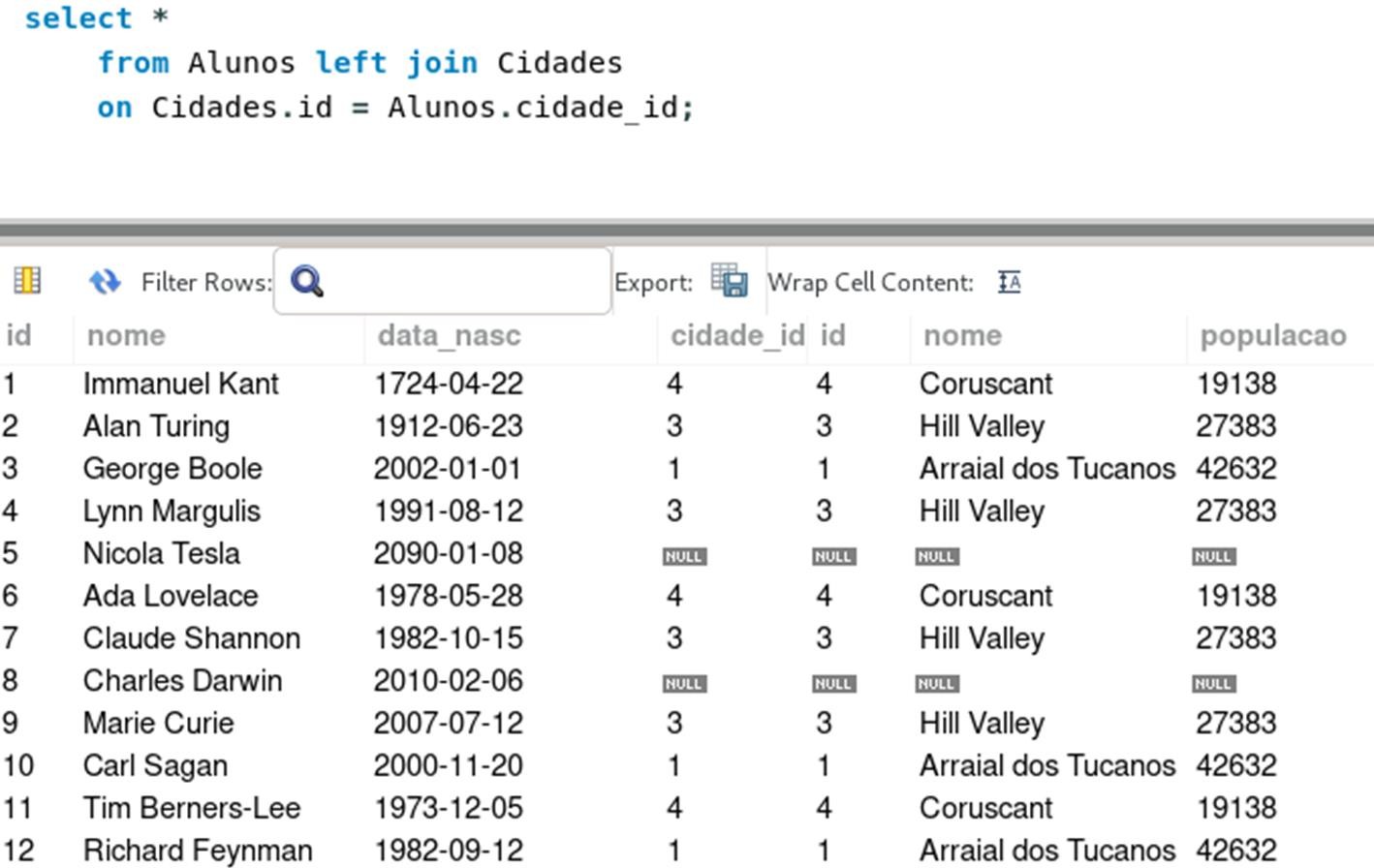
Também é possível utilizar com a seguinte sintaxe:

## SELECT \* FROM Alunos JOIN Cidades ON Cidades.id = Alunos.cidade\_id;



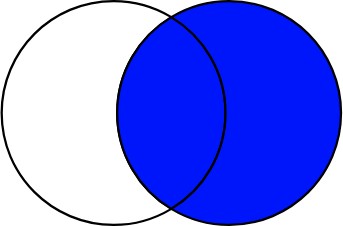
**LEFT JOIN (ou LEFT OUTER JOIN):** Retorna todos os registros da tabela à esquerda (tabela principal) e os registros correspondentes da tabela à direita (ou NULL se não houver correspondência).

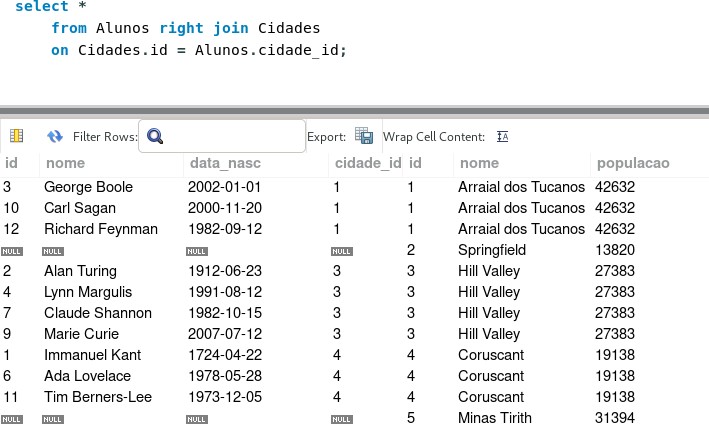




**RIGHT JOIN (ou RIGHT OUTER JOIN):** Retorna todos os registros da tabela à

direita e os registros correspondentes da tabela à esquerda.





Em SQL, um "alias" é um nome temporário ou alternativo atribuído a uma tabela ou a uma coluna de tabela, a fim de tornar uma consulta mais legível ou para evitar conflitos de nomeação.

O uso de aliases pode tornar o código SQL mais claro e conciso, além de ser útil em várias situações. Existem dois tipos principais de aliases em SQL: aliases de tabela e aliases de coluna.

Um bom exemplo é o resultado de um select por padrão, identifica cada coluna

com o nome do campo definido na tabela.

Podemos alterar esse comportamento usando aliases, ou apelidos. A sintaxe para

apelidar uma coluna é: nome\_da\_coluna as apelido



**Alias de coluna** é um nome temporário ou alternativo atribuído a uma coluna em

uma consulta SQL.

SELECT Nome, Salario AS Salario\_Anual FROM Funcionarios;

Neste exemplo, a coluna "Salario" é renomeada como "Salario\_Anual" no resultado

da consulta.

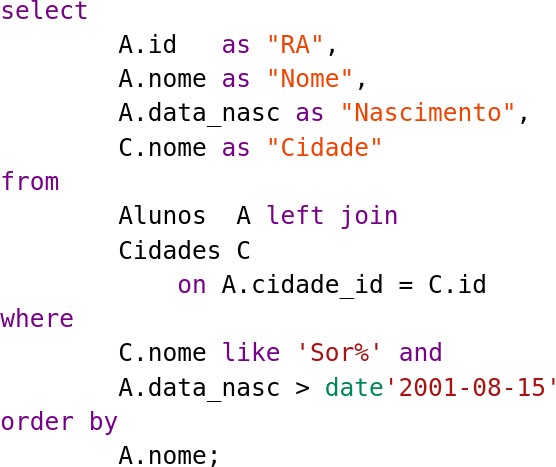
Outro exemplo pode ser o usado no código abaixo:

## SELECT e.Nome AS Nome\_Empregado, d.Nome AS Nome\_Departamento FROM Empregados AS e JOIN Departamentos AS d ON e.DepartamentoID = d.ID;

Neste exemplo, "Empregados" é renomeado para "e" e "Departamentos" é renomeado para "d". Isso torna a consulta mais legível e evita ambiguidades quando ambas as tabelas têm uma coluna chamada "Nome".

Podemos ainda dar nomes alternativos para tabelas. A sintaxe para apelidar uma

tabela é: nome\_da\_tabela apelido



Crie duas tabelas conforme o modelo apresentado nos slides 61 e 62, deste material;

Preste atenção aos campos que estão no exemplo;

Insira os valores conforme os slides;

Aplique o exemplo sobre inner join dado no slide 67, deste material; Execute essas atividades dentro o Oracle Workbench;

Crie um repositório remoto e envie o script em SQL;

Produza o Readme do repositório remoto e tire um print da tela após o término da

atividade;

Coloque comentários em seu código.

Está atividade corresponde as atividades complementares 2; A data de entrega será para próxima aula;

Utilize o tempo disponível durante a aula para execução da atividade;

Não será aceito o exercício após a data de entrega;

Execute cada uma das atividades solicitadas, a falta de algum dos itens será descontado em nota;

Caso execute em seu computador pessoal, realize a instalação dos itens a seguir

para conclusão da atividade.



# MUITO OBRIGADO!!!!



[**daniel.ohata@facens.br**](mailto:danthyl@hotmail.com)