SQL e MySQL

SQL = desenvolvido em 1970, pela IBM.
Vantagens = aprendizado, portabilidade, longevidade, comunicação e liberdade de escolha.
Desvantagens = falta de criatividade e estruturação.
Comandos = DDL (estruturação do banco de dados), DML (manipulação dos dados) e DCL (administração do banco de dados).
MySQL = criado por David Axmark, Allan Larsson e Michael Widenius com C++.
Site = W3Schools. Explica bastante todas as possibilidades do MySQL.
IDE = o SQL permite fazer tudo por linhas de comando, como na programação. Mas pode ser usado também o IDE, que coloca as coisas de uma forma mais ilustrativa. O IDE que o MySQL usa é o Workbench.
Tipos de Dados = há diversos tipos de dados possíveis de se armazenar, de acordo com o que é especificado na criação da tabela. Quanto menor o intervalo permitido, mais fácil armazenar.
Tinyint = armazena 255 números inteiros. Do -127 ao +127 se for signed, do 0 ao 255 se for unsigned.
Smallint = armazena 65.535 números inteiros. Do -32.767 ao +32.767 se for signed, do 0 ao 65.535 se for unsigned.
Mediumint = armazena 16.777.215 números inteiros. Do -8.388.607 ao +8.388.607 se for signed, do 0 ao 16.777.215 se for unsigned.
Int = armazena 4.294.967.295 números inteiros. Do -2.147.483.647 ao +2.147.483.647 se for signed, do 0 ao 4.294.967.295 se for unsigned.
Bigint = armazena muitos números inteiros, mesmo.
Float = armazena números decimais. Para declarar é float(numero_caracteres_gerais,numero_caracteres_decimais).
Double = armazena números decimais que precisam ser extremamente precisos. Para declarar é double(numero_caracteres_gerais,numero_caracteres_decimais).
Bit = armazena 0 e 1.
Zerofill = preenche com 0 os espaços vazios no número, para que fique igual em todo o campo.
Auto-Increment = preenche o campo com números crescentes.
Year = armazena ano.
Date = armazena ano-mês-dia.
Time = armazena horas:minutos:segundos.
Datetime = armazena ano-mês-dia horas:minutos:segundos.
Timestamp = armazena ano-mês-dia horas:minutos:segundos fuso horário.
Varchar = armazena strings.
Char = armazena strings, e preenche com espaços vazios os caracteres não usados, como um zerofill de letras.
Enum = armazena opções de preenchimento. Para declarar é enum("opção 1", "opção 2", "opção 3").
Banco de Dados = contém várias entidades, sendo a mais comum a tabela. As tabelas são agrupadas dentro de esquemas, para melhor organização.
Tabela = formada por campos (colunas) e registros (linhas). A quantidade de campos é informada na criação da tabela, mas pode haver quantos registros precisar.

Chave Primária = campo opcional. Diz que os valores não podem se repetir nos registros. Funciona como identificador.
Chave Estrangeira = campo vindo de outra tabela, meio "emprestado" pra essa.
Índice = ferramenta que facilita a busca por informações.
View = resultado de uma busca em várias tabelas. Mostra como se fosse uma tabela com os resultados, mas está apenas consultando as outras, os dados não estão ali de fato.
Procedures = como um mini código que pede ao banco de dados um resultado mais elaborado, com while, for, if...
Trigger = aviso que pode ser programado caso algo aconteça, como inserção ou exclusão de dados.
Importante = sempre selecionar o banco de dados, para acessar a tabela. O símbolo de diferente no banco de dados é <>. Se pedir consultas com > ou < referente a strings, ele considera a ordem alfabética para filtrar.
Raio = executa o pedido feito.
Seleção = se selecionar com o mouse parte do pedido, só o selecionado vai ser executado.
CREATE DATABASE banco_de_dados; = cria um banco de dados.
USE banco_de_dados = seleciona o banco de dados.
DROP DATABASE banco_de_dados; = deleta o banco de dados.
CREATE TABLE tabela (primeira_coluna tipo_primeiro_dado (numero_caracteres), (segunda_coluna tipo_segundo_dado (numero_caracteres)) = cria uma tabela.
SELECT * FROM tabela = mostra tudo da tabela.
SELECT coluna AS novo_nome FROM tabela; = mostra a coluna da tabela com um novo nome, na visualização.
SELECT coluna FROM tabela LIMIT numero; = mostra a quantidade de registros pedidos no número, apenas.
SELECT * FROM tabela WHERE coluna condição valor; = mostra o resultado apenas do que atender à condição.
SELECT * FROM tabela WHERE coluna condição valor AND coluna condição valor; = mostra o resultado apenas do que atender a ambas condições.
SELECT * FROM tabela WHERE coluna condição valor OR coluna condição valor; = mostra o resultado do que atender a qualquer das condições.
SELECT * FROM tabela WHERE coluna IN ("valor 1", "valor 2"); = mostra os resultados que tiverem esses valores.
SELECT * FROM tabela WHERE coluna LIKE "%valor%"; = mostra os resultados que tiverem o valor escrito, independente da posição.
SELECT * FROM tabela WHERE coluna LIKE "valor%"; = mostra os resultados que tiverem o valor escrito no início.
SELECT * FROM tabela WHERE coluna LIKE "%valor"; = mostra os resultados que tiverem o valor escrito no final.
SELECT DISTINCT * FROM tabela; = não mostra resultados repetidos. Se tiver várias linhas iguais, mostra apenas uma delas.
SELECT * FROM tabela ORDER BY coluna ASC; = mostra os resultados na ordem crescente da coluna.
SELECT * FROM tabela ORDER BY coluna DESC; = mostra os resultados na ordem decrescente da coluna.
SELECT primeira_coluna, SUM(segunda_coluna) AS novo_nome_segunda_coluna FROM tabela GROUP BY primeira_coluna; = mostra o resultado das duas colunas, agrupando a primeira e somando a segunda.
SELECT primeira_coluna, MAX(segunda_coluna) AS novo_nome_segunda_coluna FROM tabela GROUP BY primeira_coluna; = mostra o resultado das duas colunas, agrupando a primeira e mostrando o maior valor da segunda.
SELECT primeira_coluna, MIN(segunda_coluna) AS novo_nome_segunda_coluna FROM tabela GROUP BY primeira_coluna; =

mostra o resultado das duas colunas, agrupando a primeira e mostrando o menor valor da segunda.
SELECT primeira_coluna, AVG(segunda_coluna) AS novo_nome_segunda_coluna FROM tabela GROUP BY primeira_coluna; = mostra o resultado das duas colunas, agrupando a primeira e mostrando a média de valores da segunda.
SELECT primeira_coluna, COUNT(segunda_coluna) AS novo_nome_segunda_coluna FROM tabela GROUP BY primeira_coluna; = mostra o resultado das duas colunas, mostrando a quantidade de ocorrências da primeira.
SELECT primeira_coluna, AGRUPADOR(segunda_coluna) AS novo_nome_segunda_coluna FROM tabela GROUP BY primeira_coluna HAVING AGRUPADOR (segunda_coluna) condição valor; = mostra o resultado das duas colunas, agrupando a primeira, fazendo o necessário com a segunda e respeitando a condição do having. Ele funciona como um where, mas para agrupamentos.
SELECT primeira_coluna, segunda_coluna, CASE WHEN segunda_coluna condição_1 valor_1 THEN "característica 1" WHEN segunda_coluna condição_2 valor_2 THEN "característica 2" ELSE "característica 3" END AS novo_nome_segunda_coluna FROM tabela; = mostra o resultado das duas colunas, classificando a segunda de acordo com as condições em, nesse caso, três grupos.
SELECT A.coluna_primeira_tabela, B.coluna_segunda_tabela FROM primeira_tabela A INNER JOIN segunda_tabela B ON A.coluna_igual_primeitra_tabela = B.coluna_igual_segunda_tabela; = seleciona uma coluna de uma tabela e uma coluna de outra tabela e mostra os dados dos registros em comum, que possuem os mesmos valores nas colunas iguais.
SELECT A.coluna_primeira_tabela, B.coluna_segunda_tabela FROM primeira_tabela A LEFT JOIN segunda_tabela B ON A.coluna_igual_primeitra_tabela = B.coluna_igual_segunda_tabela; = seleciona uma coluna de uma tabela e uma coluna de outra tabela e mostra todos os dados da coluna da primeira tabela e seus correspondentes na segunda tabela, preenchendo com null o que estiver faltando.
SELECT A.coluna_primeira_tabela, B.coluna_segunda_tabela FROM primeira_tabela A RIGHT JOIN segunda_tabela B ON A.coluna_igual_primeitra_tabela = B.coluna_igual_segunda_tabela; = seleciona uma coluna de uma tabela e uma coluna de outra tabela e mostra todos os dados da coluna da segunda tabela e seus correspondentes na primeira tabela, preenchendo com null o que estiver faltando.
SELECT A.coluna_primeira_tabela, B.coluna_segunda_tabela FROM primeira_tabela A, segunda_tabela; = seleciona uma coluna de uma tabela e uma coluna de outra tabela e mostra todas as combinações possíveis entre os dados de ambas as colunas.
SELECT A.coluna_primeira_tabela, B.coluna_segunda_tabela FROM primeira_tabela A LEFT JOIN segunda_tabela B ON A.coluna_igual_primeitra_tabela = B.coluna_igual_segunda_tabela; = seleciona uma coluna de uma tabela e uma coluna de outra tabela e mostra todos os dados da coluna da primeira tabela e seus correspondentes na segunda tabela, preenchendo com null o que estiver faltando.
SELECT primeira_coluna FROM primeira_tabela UNION SELECT segunda_coluna FROM segunda_tabela; = mostra em uma tabela todos os dados de ambas as colunas, sem repeti-los, tem como um distinct embutido.
SELECT primeira_coluna FROM primeira_tabela UNION ALL SELECT segunda_coluna FROM segunda_tabela; = mostra em uma tabela todos os dados de ambas as colunas, repetindo-os.
Agrupadores = Pode ser sum (soma), max (maior número), min (menor número), avg (média) ou count (contador).
Condições = Pode ser >, <, >=, <= ou <>. Se forem postas em strings, respeitam a ordem alfabética. Em datas, o calendário.
INSERT INTO tabela (primeira_coluna, segunda_coluna) VALUES ("valor 1", "valor 2"); = insere os dados em uma tabela.
UPDATE tabela SET primeira_coluna = "novo_valor", segunda_coluna = "novo_valor" WHERE coluna = "valor"; = atualiza a

tabela com os novos valores.

ALTER TABLE tabela ADD PRIMARY KEY (coluna); = transforma uma coluna em coluna de chave primária.

ALTER TABLE tabela ADD COLUMN (coluna tipo_dado); = adiciona uma nova coluna à tabela.

DROP TABLE banco_de_dados; = deleta a tabela.

DELETE FROM tabela WHERE coluna = "valor"; = deleta um registro da tabela, uma linha.

CREATE VIEW "vw_nome" AS select; = cria uma visualização parcial dos dados, salva aquele select para ser mostrado.

database → reverse engineer → next → next → seleciona o banco de dados → next → next → execute = no MySQL, mostra um esquema das tabelas que possuem chaves estrangeiras, simplificando as relações entre elas.

Funções = há várias funções que podem ser usadas no banco de dados, para strings, números e datas. Um bom site para consultá-las é o W3Schools.

Conversões = há vários modos de converter o tipo de dado exibido na coluna, no momento da exibição. Um bom site para consultá-los é o W3Schools.