# CADERNO MySQL – MARCOS VALENTE

**SUMÁRIO**

[CADERNO MySQL – MARCOS VALENTE 1](#_Toc131783925)

[SOBRE BANCO DE DADOS ESTRUTURADO 2](#_Toc131783926)

[SOBRE MySQL 3](#_Toc131783927)

[PRIMEIRO BANCO DE DADOS 3](#_Toc131783928)

[Criando primeiro banco de dados: 4](#_Toc131783929)

[Tipos primitivos do MySQL: 4](#_Toc131783930)

[Criando tabelas dentro do banco de dados: 6](#_Toc131783931)

[MELHORANDO A ESTRUTURA DO BANCO DE DADOS 6](#_Toc131783932)

[INSERINDO DADOS NA TABELA (INSERT INTO) 8](#_Toc131783933)

[ALTERANDO A ESTRUTURA DA TABELA (ALTER TABLE E DROP TABLE) 10](#_Toc131783934)

[MANIPULANDO LINHAS/ REGISTROS/ TUPLAS (UPDATE, DELETE e TRUNCATE) 12](#_Toc131783935)

[GERENCIANDO CÓPIAS DE SEGURANÇA MySQL (BACKUPS) 15](#_Toc131783936)

[FAZENDO BACKUPS PELO WORKBENCH 15](#_Toc131783937)

[FAZENDO BACKUPS POR COMANDO 16](#_Toc131783938)

[IMPORTANDO BACKUP 16](#_Toc131783939)

[SELECT (PARTE 1) 16](#_Toc131783940)

[SELECIONANDO COLUNAS 16](#_Toc131783941)

[SELECIONANDO LINHAS 18](#_Toc131783942)

[SELECT (PARTE 2) 21](#_Toc131783943)

[OPERADOR “LIKE” 21](#_Toc131783944)

[SELECT DISTINCT 22](#_Toc131783945)

[FUNÇÕES DE AGREGAÇÃO 23](#_Toc131783946)

[SELECT (PARTE 3) 25](#_Toc131783947)

[AGRUPANDO REGISTROS 25](#_Toc131783948)

[SELECT (PARTE 4) 26](#_Toc131783949)

[FILTROS/ CONDIÇÕES (SELECT... CASE) 26](#_Toc131783950)

[CONCAT 27](#_Toc131783951)

[MODELO RELACIONAL 28](#_Toc131783952)

[CHAVES ESTRANGEIRAS E JOIN 32](#_Toc131783953)

[INNER JOIN COM VÁRIAS TABELAS 36](#_Toc131783954)

[SUBQUERIES (SUBCONSULTAS) 38](#_Toc131783955)

[OUTROS COMANDOS 38](#_Toc131783956)

[COMANDOS RELACIONAIS 38](#_Toc131783957)

[FORMATAÇÃO 39](#_Toc131783958)

[RESUMO COMANDOS MySQL 40](#_Toc131783959)

[RESUMO JOINS 40](#_Toc131783960)

# SOBRE BANCO DE DADOS ESTRUTURADO

Imagem de vídeo game

Descrição gerada automaticamente com confiança média

* Arquivos guardam tabelas que guardam registros

Arquivos de acesso sequencial são aqueles em que os registros são acessados sequencialmente

Arquivos de acesso direto são aqueles em que os registros são armazenados e acessados de maneira direta

Essas tecnologias foram substituídas pelos bancos de dados. Compõe um banco de dados:

* Banco de dados
* Sistema gerenciador (SGBD, em português; DNS, em inglês) gerencia os dados
* Linguagem de exploração
  + Structured Query Language (SQL)
    - Linguagem que permite dar comandos ao meio ambiente do banco de dados, para ter como retorno uma resposta
    - A ideia era que a linguagem SQL fosse universal (padronização)
* Programas adicionais

Grandes bancos de dados:

* Oracle (pago)
* IBM (pago)
* SQL Server – Microsoft (pago)
* dBase (pago - em desuso)
* MySQL (gratuito)
* MariaDB (gratuito)
* Firebird (gratuito)
* PostgreSQL (gratuito)

# SOBRE MySQL

Instruções/ linguagens do MySQL

* DDL: linguagem de definição, que permite alterar a estrutura da base de dados
* DML: linguagem de manipulação, que permite manipular (adicionar, alterar, excluir) dados
* DQL: linguagem de solicitações (select)
* DCL: linguagem de controle
* DTL: linguagem de transações

# PRIMEIRO BANCO DE DADOS

Objetivo do banco de dados: registrar instâncias separadas de coisas que possuem características semelhantes

* Exemplos de características de pessoas: nome, nacionalidade, altura, peso...
  + As características são semelhantes, o que muda são os valores

Banco de dados são coleções de dados que estão organizados em locais específicos (tabelas, as quais guardam dados de coisas que possuem características semelhantes).

Desenho de um barco

Descrição gerada automaticamente com confiança baixa

* Banco de dados: conjunto de tabelas
* Tabelas: conjunto de registros
* Registros são compostos por campos
* A primeira linha da tabela contém os atributos da tabela, chamados de esquema
* As outras linhas são chamadas de tuplas ou registros; o conjunto delas é chamado de instâncias

## Criando primeiro banco de dados:

* Utilizar o comando “create database *nome do banco de dados*;”
  + Depois executar o comando

## Tipos primitivos do MySQL:

Diagrama

Descrição gerada automaticamente

* + Utilizar varchar quando não souber (ou variar) o tamanho da string armazenada
  + Utilizar char quando souber exatamente o tamanho da string (economia de processamento da máquina)
  + Ao apontar um número int, pode-se passar em um parêntese a quantidade de dígitos do número
    - Exemplo: id\_livro INT(10)
    - Observação: com a opção ZEROFILL habilitada, o MySQL irá cadastrar ‘00065’. Sem a opção ZEROFILL habilitada, o MySQL irá cadastrar ’65’.
  + Ao apontar um campo float, passar um parêntese indicando o número de dígitos (incluindo as casas decimais) e o número de casas decimais
    - Exemplo: quilometragem FLOAT(5,2)
      * Isso significa que haverá 5 dígitos no total, sendo dois deles reservados para casas decimais (ex: 100,50)

Texto

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente

## Criando tabelas dentro do banco de dados:

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

* + Observação: esse é um único comando (o símbolo “;” indica o fim do comando)
  + Observação 2: na primeira linha, escreveu-se “use cadastro” pois, nesse caso, o nome do banco de dados era “cadastro”
* Utilize o comando “decribe *nome da tabela*” para que o MySQL descreva a tabela para você

# MELHORANDO A ESTRUTURA DO BANCO DE DADOS

* Comando para excluir banco de dados:



* Criando bancos de dados com **constraints** (parâmetros):

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

* + Dessa forma, o banco de dados terá uma codificação de caracteres por padrão e um collation padrão

**Criando tabelas mais bem estruturadas, otimizadas**

* Com isso, economiza-se espaço (bytes alocados) no banco de dados

**Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente**

* O constraint “not null” obriga o usuário a cadastrar esse campo
* O tipo primitivo “enum” (colocando os valores entre parênteses e separados por vírgula) determina quais valores serão aceitos (torna a estrutura mais rígida)
* O tipo decimal acompanhado de um parênteses com dois números indica a quantidade de casas decimais (5 nesse caso) e de casas decimais (2 nesse caso)
* A constraint “default” faz com que o valor padrão seja Brasil (será Brasil até que alguém digite outra coisa)
* Repare que também foi determinado um charset após determinar os campos

**Observação:** é possível cadastrar o nome dos campos entre crases

Uma imagem contendo Texto

Descrição gerada automaticamente

* Isso permite que se utilize campos com acentos ou espaços

**Chave primária**

* Para que um mesmo registro não seja cadastrado duas ou mais vezes (ocorrência de tuplas), é importante definir um campo como sendo uma **chave primária**
  + O valor desse campo não poderá ser repetido
  + Exemplos: CPF, matrículas, etc.
  + Nomes não são boas chaves primarias, pois pode haver homônimos
* Cadastro de chave primária

**Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente**

* Nesse caso, o campo ID é a chave primária
* Constraint “auto\_increment” faz com que o cadastro seguinte tenha seja o próximo número do cadastro anterior

**Criando tabela com novos constraints**

**Texto

Descrição gerada automaticamente com confiança média**

* IF NOT EXISTS: só permite criação de tabela se uma tabela com esse nome ainda não existir
  + Se não utilizar esse comando e houver uma tabela com mesmo nome, aquela será apagada e substituída pela nova
* UNIQUE: não permite que dois valores iguais sejam cadastrados (exemplo: não pode haver dois cursos com o mesmo nome, não pode haver duas pessoas com mesmo CPF)
* UNSIGNED: sem sinal (economia de um byte, mas também impede que sejam cadastrados números negativos)

# INSERINDO DADOS NA TABELA (INSERT INTO)

Texto

Descrição gerada automaticamente

Comando para inserir registros/ dados dentro da tabela (no exemplo, tabela “pessoas”):

Uma imagem contendo Word

Descrição gerada automaticamente

Datas no MySQL: ‘aaaa-mm-dd’

* Não era necessário informar o “id”, haja visto que esse campo é autoincrementável
* É possível, ao invés de inserir um valor, pedir para atribuir o valor padrão (default)

Texto

Descrição gerada automaticamente



Observação:

* Se os dados inseridos estiverem na mesma ordem dos campos cadastrados na tabela, não é necessário informar os campos no código
* Exemplo:

Uma imagem contendo Gráfico

Descrição gerada automaticamente

**Inserir múltiplos dados ao mesmo tempo:**

**Uma imagem contendo Texto

Descrição gerada automaticamente**

* Repare que os registros devem ser separados por vírgulas (somente o último acompanha ponto-e-vírgula)

# ALTERANDO A ESTRUTURA DA TABELA (ALTER TABLE E DROP TABLE)

* Adicionar nova “coluna” (os campos no MySQL são chamados de colunas)

Texto

Descrição gerada automaticamente com confiança média

* + Observação: os cadastros já inseridos receberão valores “null” nesse campo
  + O termo “column” nesse código é opcional
* Adicionar nova coluna definindo seu posicionamento na tabela

Uma imagem contendo Texto

Descrição gerada automaticamente

* + Também é possível indicar o posicionamento da coluna da seguinte forma:

Texto

Descrição gerada automaticamente

* + Se não for definida a localização dessa coluna, ela será alocada ao final da tabela
* Remover coluna da tabela (eliminando-a)

Interface gráfica do usuário

Descrição gerada automaticamente com confiança baixa

* Alterar estrutura do campo (alterar seu tipo primitivo e suas constraints)
  + Só não é possível mudar o nome do campo

Logotipo

Descrição gerada automaticamente

* + Nesse caso, foi obrigatório definir uma constraint DEFAULT com varchar vazio, pois a tabela já continha registros e nós determinamos que a coluna profissão não poderia ser nula (NOT NULL)
* Modificar nome da coluna (bem como tipos primitivos e constraints)

Logotipo

Descrição gerada automaticamente com confiança média

* + Nome antigo: profissao
  + Nome novo: prof
  + Repare que mantive os mesmos constraints (se não repetisse eles, eles seriam eliminados)
  + Não indicado utilizar esse comando para mudar tipos primitivos e constraints sem modificar o nome
* Renomear a tabela inteira

Texto

Descrição gerada automaticamente com confiança média

* Adicionar uma chave primária a uma tabela
  + Observação: a coluna já deve existir (não é possível cria-la e, no mesmo comando, torná-la chave primária)

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

* Apagar tabelas
  + Com esse comando, a tabela e os dados contidos nela são apagados (CUIDADO)



# MANIPULANDO LINHAS/ REGISTROS/ TUPLAS (UPDATE, DELETE e TRUNCATE)

Interface gráfica do usuário, Tabela, Excel

Descrição gerada automaticamente

* Linhas são tuplas ou registros
* Colunas são colunos ou atributos

**Modificando linhas incorretas:**

* Observação: um comando modifica somente uma linha
* Ao sinalizar a linha que será modificada, é interessante valer-se da chave primária, pois ela garante que somente aquela linha será manipulada

*Exemplo*

Tabela utilizada (“CURSOS”):

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Comando:

Uma imagem contendo Tabela

Descrição gerada automaticamente

* Objetivo do comando: corrigir nome “HTML4” para “HTML5”

Corrigindo mais de um campo por linha:

Uma imagem contendo Interface gráfica do usuário

Descrição gerada automaticamente

Comando para certificar-se que somente uma linha será modificada:

* Para além da chave primária (que já é uma garantia)

Uma imagem contendo Gráfico

Descrição gerada automaticamente

* Repare que o comando **limit** **1**, obriga o MySQL a somente modificar uma linha

É necessário ter cuidado, pois escolher um campo cujo valor é compartilhado por outras linhas para servir de localização para mudança pode alterar todas as linhas que o compartilham

*Exemplo:*

*Interface gráfica do usuário, Tabela

Descrição gerada automaticamente*

* Repare que esses três últimos cursos compartilham o valor 2018 no campo “ano”
* Se o usuário decidir localizar a modificação dos valores com base no valor “2018” no campo “ano”, ele modificará essas três linhas

Interface gráfica do usuário

Descrição gerada automaticamente com confiança média

Interface gráfica do usuário, Texto

Descrição gerada automaticamente

* Continuando nesse exemplo, se o usuário empreender uma nova modificação, mas com o comando “**limit 1**”, ele modificará somente a primeira linha que se adequa nos parâmetros selecionados

Uma imagem contendo Interface gráfica do usuário

Descrição gerada automaticamente

Interface gráfica do usuário, Texto

Descrição gerada automaticamente

**Apagar linhas**

**Uma imagem contendo Texto

Descrição gerada automaticamente**

* Cuidado, se não houver backup, a linha será deletada DEFINITIVAMENTE

**Remover todas as linhas de uma tabela**

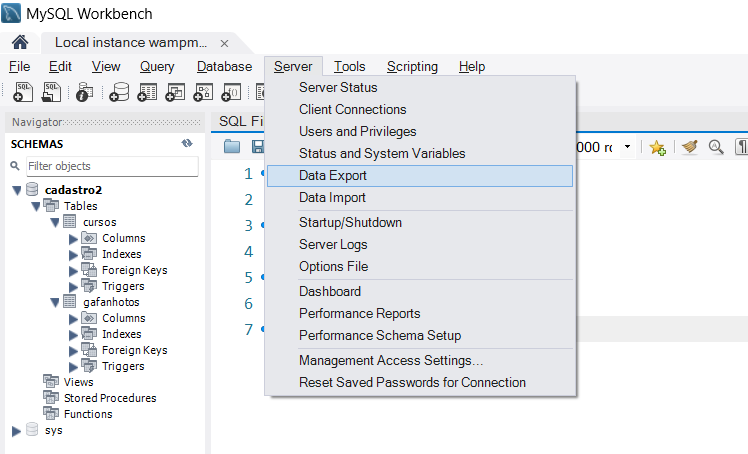
****

* Obs: comando “**truncate** *nome\_da\_tabela*” também é válido (não é necessário indicar que é uma tabela)
* Truncar uma tabela equivale a deletar todos os registros

# GERENCIANDO CÓPIAS DE SEGURANÇA MySQL (BACKUPS)

Para mais informações: [Curso MySQL #08 - Gerenciando Cópias de Segurança MySQL - YouTube](https://www.youtube.com/watch?v=w6OYS_M7hTM)

## FAZENDO BACKUPS PELO WORKBENCH



* Selecionar o banco de dados (nesse caso “Cadastro 2”) e os objetos desejados (as tabelas):

Interface gráfica do usuário, Aplicativo

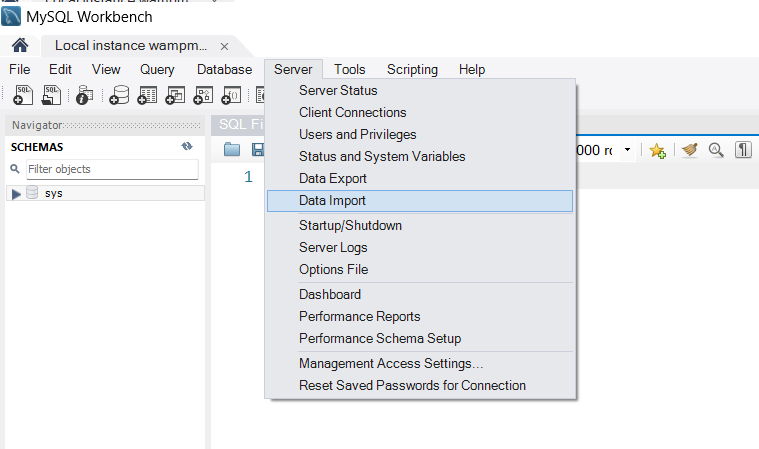
Descrição gerada automaticamente

* Observação: o backup de um banco de dados se chama “dump”
* Dump Structure and Data: faz backup da estrutura e dos dados
* Dump Structure: backup somente da estrutura do banco de dados
* Dump Data: somente os dados
* O dump, quando se é exportado como um “self-contained file” é basicamente uma lista de comandos, a qual será aplicada pelo MySQL

## FAZENDO BACKUPS POR COMANDO



## IMPORTANDO BACKUP



# SELECT (PARTE 1)

## SELECIONANDO COLUNAS

Texto

Descrição gerada automaticamente

* Seleciona todos os campos (colunas) da tabela “cursos”
  + Asterisco significa todas as colunas
  + Dessa forma, mostra todos os registros inseridos em todas as colunas da tabela
* É possível aplicar uma ordem para mostrar os registros (com base na coluna indicada):

Texto

Descrição gerada automaticamente com confiança baixa

* + Dessa forma, ordena a apresentação pela coluna nome
  + De outra forma, seria apresentado com base no id
  + Ainda é possível indicar a ordem com que os dados serão apresentados:

Texto

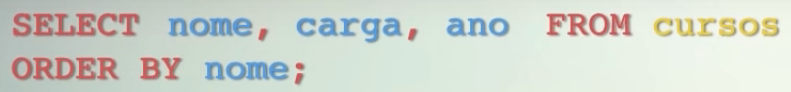
Descrição gerada automaticamente com confiança baixa

* + - Apresenta os registros de maneira decrescente (se forem caracteres, de Z até A)

Uma imagem contendo Texto

Descrição gerada automaticamente

* + - Apresenta os registros de maneira crescente (mas não é necessário, pois essa já é a forma padrão)
* Selecionar colunas específicas
  + Selecionam-se todos os registros, mas não são apresentadas todas as colunas



Tabela

Descrição gerada automaticamente

* + É possível inclusive mudar a ordem (a ordem depende da apresentação das colunas no comando). Exemplo:

Uma imagem contendo Texto

Descrição gerada automaticamente

Tabela

Descrição gerada automaticamente

* + - Repare que no ano de 2014 houve três cursos diferentes. É possível ordená-los por nome também da seguinte maneira:

Tela de celular com aplicativo aberto

Descrição gerada automaticamente com confiança média

Tabela

Descrição gerada automaticamente

## SELECIONANDO LINHAS

Texto

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

**Selecionar linhas e colunas ao mesmo tempo:**

Interface gráfica do usuário, Texto

Descrição gerada automaticamente com confiança média

Tabela

Descrição gerada automaticamente

* Repare que, nesse caso, a coluna ano (responsável por filtrar os registros) não foi apresentada (“result set”)

**Selecionando linhas com base em operadores relacionais**

* Na querry, a cláusula where pode ser seguida de um operador relacional:

**Uma imagem contendo Interface gráfica do usuário

Descrição gerada automaticamente**

**Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente**

Operadores relacionais:

|  |  |
| --- | --- |
| Operador | Símbolo |
| Igual | = |
| Diferente | != / <> |
| Maior | > |
| Maior/igual | >= |
| Menor | < |
| Menor/igual | <= |

**Selecionando intervalos (BETWEEN)**

**Interface gráfica do usuário, Texto

Descrição gerada automaticamente com confiança média**

**Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente**

**Selecionando valores (IN)**

**Texto

Descrição gerada automaticamente com confiança média**

**Interface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamente**

**Combinando testes (AND/ OR)**

**Tela de celular com texto preto sobre fundo branco

Descrição gerada automaticamente com confiança média**

**Texto

Descrição gerada automaticamente**

# SELECT (PARTE 2)

## OPERADOR “LIKE”

* Mostrar linhas cujo ‘nome’ começam com ‘p’

Uma imagem contendo Interface gráfica do usuário

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamente

* + Repare que o % é um caractere “coringa”
    - Substitui nenhum ou todos os demais caracteres (troca por qualquer conjunto de caracteres)
  + Atenção: o operador “like” não case-sensitive (não faz diferença escrever a primeira letra em maiúscula ou minúscula)
* É possível trocar a posição do caractere “coringa”

Uma imagem contendo Texto

Descrição gerada automaticamente

* + Nesse caso, ele mostrará todas as linhas cujo campo nome terminam com a letra “a”
* Selecionar linhas cujo campo selecionado apresentem determinados caracteres ou trechos

Uma imagem contendo Texto

Descrição gerada automaticamente

* + O caractere “coringa” indica que poderá haver qualquer coisa à frente e qualquer coisa atrás (inclusive nada)
* Selecionar linhas cujo campo selecionado NÃO apresentem determinados caracteres ou trechos

Texto

Descrição gerada automaticamente com confiança média

* Exigir que sejam selecionadas linhas que possuam um caractere naquela posição, ainda que não apontado qual (**caractere \_**)
  + Compare a diferença nos resultados:

Uma imagem contendo Interface gráfica do usuário

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente com confiança baixa

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

## SELECT DISTINCT

* Seleciona linhas que possuem valores distintos de um determinado campo (não mostra linhas que possuam o mesmo valor no campo selecionado)
  + Exemplo: descobrir quais nacionalidades já fizeram curso com o Prof. Gustavo Guanabara (ele apelida os seus alunos de “gafanhotos”)



Tabela

Descrição gerada automaticamente com confiança média

* + Repare que se não houvesse utilizado o comando “select disctinct”, todas as linhas da tabela seriam mostradas, apresentando muitas ocorrências repetidas (haveria, por exemplo, um grande número de registros “Brasil”)

## FUNÇÕES DE AGREGAÇÃO

**Select count()**

* Função que conta o número de registros:

****

**Uma imagem contendo Forma

Descrição gerada automaticamente**

* + Nesse caso, será apresentado o número de registros na tabela gafanhotos (na prática, apresenta-se o número alunos do prof. Gustavo Guanabara, chamados “gafanhotos”)
* Filtrando o número de registros:

****

* + Nesse caso, será apresentado o número de cursos que possuem carga horária maior que 40 horas
* Selecionando quantidade de registros distintos



**Select max()**

* Função que apresenta maior valor em um determinado campo
  + Exemplo:

****

**Uma imagem contendo Texto

Descrição gerada automaticamente**

* Filtrando seleção:

****

**Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente com confiança média**

* + Nesse caso, o MySQL apresentará o maior número de aulas dado em um curso de 2016

**Select min()**

* Função que apresenta menor valor em um determinado campo
  + Exemplo:

****

**Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente**

* + Repare que, nesse caso, será apresentado também o nome do curso (embora a função “min” esteja atribuída sobre ao campo “totaulas”)

**Select sum()**

* Função que soma os valores de um determinado campo



**Select avg()**

* Função que apresenta a média dos valores de um determinado campo



* + Nesse caso, o MySQL apresentará a média de aulas por curso apresentado em 2016 pelo prof. Gustavo Guanabara

# SELECT (PARTE 3)

## AGRUPANDO REGISTROS

* Atenção: esse comando é diferente do comando Select distinct
  + Embora muitas vezes ambos mostrem os resultados, utilizar o “GROUP BY” permite incrementar a querry com outras informações
* Exemplo:

Texto

Descrição gerada automaticamente

Tabela

Descrição gerada automaticamente

* + Querry apresenta número de cursos agrupados por número de aulas
* Exemplo 2:

Uma imagem contendo Texto

Descrição gerada automaticamente

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

* + Querry apresenta número de cursos que possuem carga acima de 30 horas, agrupados por carga

**Filtrar agrupamentos**

**Texto

Descrição gerada automaticamente com confiança baixa**

**Interface gráfica do usuário, Tabela

Descrição gerada automaticamente**

* Repare que, dessa forma, só serão mostradas os anos em que foram produzidos mais que 5 cursos
* Observação: só é possível utilizar o filtro “having” com campo demandado na querry
  + Para valer-se de outros filtros, pode-se utilizar o comando “where”. Exemplo:

**Aplicativo

Descrição gerada automaticamente com confiança baixa**

* Ou seja, a seleção foi filtrada, agrupada e ainda se delimitou, dentro desse agrupamento, quais dados exibir (pelo comando “having”)
* É possível filtrar valendo-se de outros agregadores. Exemplo:

Uma imagem contendo Texto

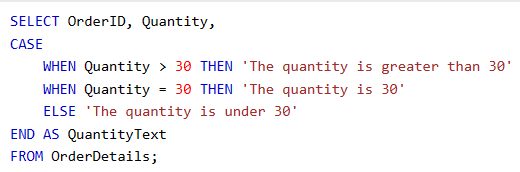
Descrição gerada automaticamente

* + Repare que foram utilizados dois comandos select
    - O segundo gera uma tabela apresentando a média da carga horária da tabela cursos

# SELECT (PARTE 4)

## FILTROS/ CONDIÇÕES (SELECT... CASE)

* Utiliza-se a estrutura “CASE” para retornar um valor quando uma condição for cumprida
* Exemplo:



* + Nesse caso, será criada uma nova coluna, chamada “QuantityText”, que trará o valor que corresponderá à condicional
    - O nome dessa coluna deve ser apresentado após “END AS”, que por sua vez deverá estar após todas as condições
  + Repare que as condições são apresentadas utilizando-se “WHEN... THEN”, podendo ser apresentado também o que acontecerá quando nenhuma delas for cumprida (“ELSE”)
  + A tabela referenciada (“FROM”) está ao final do comando

**Outros métodos de filtragem/ utilização de condições**

* Utilizando condicionantes com ORDER BY:

**Texto

Descrição gerada automaticamente com confiança média**

* + Repare que os registros serão ordenados por cidade, salvo os registros cujo campo “City” for nulo, os quais serão então ordenados por país

## CONCAT

* Método de formatação de strings
* Sintaxe:



* + Observação: se alguma expressão possuir valor nulo, ela retornará “NULL”
* Exemplo:

Gráfico, Gráfico de dispersão

Descrição gerada automaticamente

# MODELO RELACIONAL

* Atributos são inseridos em entidades
  + Toda entidade possui uma coleção de atributos definidos
  + Esses atributos compõem os dados que compõem cada um dos dados que estão dentro dessa entidade

Interface gráfica do usuário

Descrição gerada automaticamente com confiança média

* A diferença do modelo relacional é que é possível fazer relações entre entidades
  + Existe uma entidade dominante e uma entidade dominada

**Diagrama entidade-relacionamento (DER)**

**Diagrama

Descrição gerada automaticamente**

* A partir do modelo relacional, é possível traçar relações entre as tuplas das entidades

**Diagrama

Descrição gerada automaticamente**

* Sentidos de leitura:
  + Cada gafanhoto assiste N cursos;
  + Cada um dos cursos pode ser assistido por N gafanhotos.

**Uma imagem contendo objeto, relógio

Descrição gerada automaticamente**

* A cardinalidade pode ser simples (um) ou múltipla (N)
* A partir da cardinalidade, é possível classificar o relacionamento
  + Nesse caso, o relacionamento é classificado como **“muitos-para-muitos”**, ou “N para N”.
  + Exemplo de relacionamento cuja cardinalidade é **“um para um”**

**Diagrama

Descrição gerada automaticamente**

* + - Cada marido se casa apenas com uma esposa
    - Cada esposa se casa apenas com um marido
    - Observação: repare que existe um indivíduo solteiro. Uma tupla não tem que necessariamente se relacionar com outra.
  + Cardinalidade **“um-para-muitos”**

**Diagrama

Descrição gerada automaticamente**

* + - Repare que um funcionário pode ter um, vários ou nenhum dependente
    - Cada funcionário pode cuidar de N dependentes
    - Cada dependente pode ser cuidado por apenas um funcionário
* Passos para construção do relacionamento em um banco de dados:
  + Modelar as entidades com seus atributos e relacionamentos
  + Classificar cada um dos relacionamentos
* Na prática, o relacionamento entre entidades se dá por meio das chaves-primárias (troca de chaves)
  + **Chave-estrangeira** é a chave-primária de uma entidade após ser transportada para outra
* Regras de transporte de chaves depende da cardinalidade da relação:
  + Relacionamento 1 para 1:
    - Primeira regra: quando o relacionamento é 1 para 1, é possível juntar as duas tabelas em uma só
      * Contudo, é necessário averiguar se faz sentido manter essas tabelas separadas (ou não)
    - Segunda regra (quando se pretende manter as tabelas separadas): escolher uma entidade para ser dominante
    - Terceira regra: Pegar a chave primária da entidade dominada e transportar para a entidade dominante

**Diagrama

Descrição gerada automaticamente**

* + - * Em todos os casos, a chave estrangeira cpf-esposa deve conter atributos compatíveis com a chave primária cpf-esposa (mas não precisam ter o mesmo nome)
  + Relacionamento 1 para muitos (1 para N)
    - Regra: Pegar a chave primário do lado 1 e jogar para o lado N

**Diagrama

Descrição gerada automaticamente**

* + Relacionamento muitos para muitos (N para N)
    - Primeira regra: transformar o relacionamento em uma entidade
      * Definir atributos para essa entidade
    - Segunda regra: Criar dois relacionamentos (ligando as entidades primárias à nova entidade, oriunda do relacionamento)
    - Levar a chave primária dessas duas primeiras entidades para a nova entidade (como acontece no relacionamento 1 para muitos)

**Tela de celular

Descrição gerada automaticamente com confiança média**

# CHAVES ESTRANGEIRAS E JOIN

* Chaves estrangeiras não precisam ter o mesmo nome da chave primária que a originou
  + Elas devem possuir o mesmo tipo e o mesmo tamanho (também devem ter as mesmas constraints)
  + Exemplo: varchar(30)
* Passo zero: estudar o relacionamento entre as tabelas

Diagrama

Descrição gerada automaticamente

* + Nesse caso, trata-se de uma relação de preferência (cada aluno pode preferir somente um curso)
* Primeiro passo: adicionar um campo para receber a chave estrangeira

Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente com confiança média

* + cursopreferido será o campo que receberá a chave estrangeira (é do tipo int, assim como a chave primária da tabela cursos (idcurso))
* Segundo passo: designar o campo como chave estrangeira

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

* Terceiro passo: atribuir valor à coluna “cursopreferido” (chave-estrangeira)
  + O valor será o mesmo da chave primária da outra tabela
  + Exemplo: imagine que o curso preferido de um determinado aluno seja “MySQL”
    - Na tabela de cursos, MySQL possui a chave primária “6”
    - Logo, o aluno, registrado na tabela gafanhotos, deverá ter como valor na coluna “cursopreferido” o valor “6” também





**Junções do MySQL**

* Até agora, a coluna cursopreferido mostra somente a chave estrangeira
* O comando JOIN permite visualizar a relação entre as tabelas mais explicitamente
  + Deve vir acompanhado da cláusula “on”

**Tela de celular com texto preto sobre fundo branco

Descrição gerada automaticamente com confiança média**

**Tabela

Descrição gerada automaticamente**

* + Repare que, quando tabelas são relacionadas, os campos devem explicitar a tabela de origem (*tabela*.*coluna*)
    - É possível atribuir apelidos às tabelas, para se poupar espaço no comando

**Interface gráfica do usuário, Texto

Descrição gerada automaticamente**

* + Nesse caso, o comando JOIN equivale a um INNER JOIN (ambos podem ser escritos)
    - Com esse comando, somente serão mostrados os campos que possuem relação (tuplas sobre as quais não se designou uma chave estrangeira não aparecerão no resultado dessa querry)

**Outer join**

* Comando mostrará campos que não receberam chave estrangeira também
  + Nesse caso, mostrará nomes de alunos que não possuem curso preferido

Interface gráfica do usuário, Texto

Descrição gerada automaticamente

Tabela

Descrição gerada automaticamente

* + Repare que o **LEFT JOIN** priorizará a tabela da esquerda (alunos)
  + O comando **RIGHT JOIN** priorizará a tabela da direita (cursos)

Texto

Descrição gerada automaticamente

Tabela

Descrição gerada automaticamente

Diagrama

Descrição gerada automaticamente

# INNER JOIN COM VÁRIAS TABELAS

**Relações muitos-para-muitos (N para N)**

Diagrama

Descrição gerada automaticamente

* Necessário transformar a ação em uma entidade, criando novos relacionamentos
  + Essa nova entidade possuirá atributos próprios (bem como uma chave-primária)

Uma imagem contendo objeto, relógio

Descrição gerada automaticamente

* Primeiro passo: criar nova entidade com duas chaves estrangeiras

Interface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamente

* Segundo passo: inserir dados na tabela
* Terceiro passo: estabelecer junções (utilizar dois comandos “join” na mesma querry)

Interface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, Tabela

Descrição gerada automaticamente

# SUBQUERIES (SUBCONSULTAS)

* Pode-se utilizar as subqueries em 3 situações:

1. Dentro da cláusula WHERE, como um filtro;
2. Dentro da cláusula SELECT, como uma nova coluna;
3. Dentro da cláusula FROM, como uma nova tabela.

**Subquery dentro da cláusula WHERE**

Texto

Descrição gerada automaticamente com confiança baixa

Tabela

Descrição gerada automaticamente

# OUTROS COMANDOS

## COMANDOS RELACIONAIS

* Por padrão, é impossível remover registros que são referenciados em outras tabelas (por meio de chaves estrangeiras)
  + Por isso, pode ser interessante, na hora de criar a tabela mãe (aquela que possuirá as informações que serão referenciadas) introduzir os comandos:
    - "ON UPDATE CASCADE,
    - “ON DELETE CASCADE;”
  + Dessa forma, se uma linha for deletada na tabela mãe, as linhas das tabelas filhas que fizerem referência a ela também serão deletadas

## FORMATAÇÃO

* Para formatar número de casas decimais apresentadas, utiliza-se “SELECT FORMAT (parâmetro, *número\_casas*) FROM *tabela*”
  + Exemplo:



* Para arredondar números, utiliza-se “SELECT ROUND(parâmetro, número\_casas) FROM tabela”
  + Observação: o número de casas decimais é opcional (se não for especificado, o MySQL entende que se trata de zero)
  + Exemplo:



# RESUMO COMANDOS MySQL

Tabela

Descrição gerada automaticamente

# RESUMO JOINS

Diagrama

Descrição gerada automaticamente