

Objetivos da Aula:

- SQL - (Strutured Query Language)
- DDL

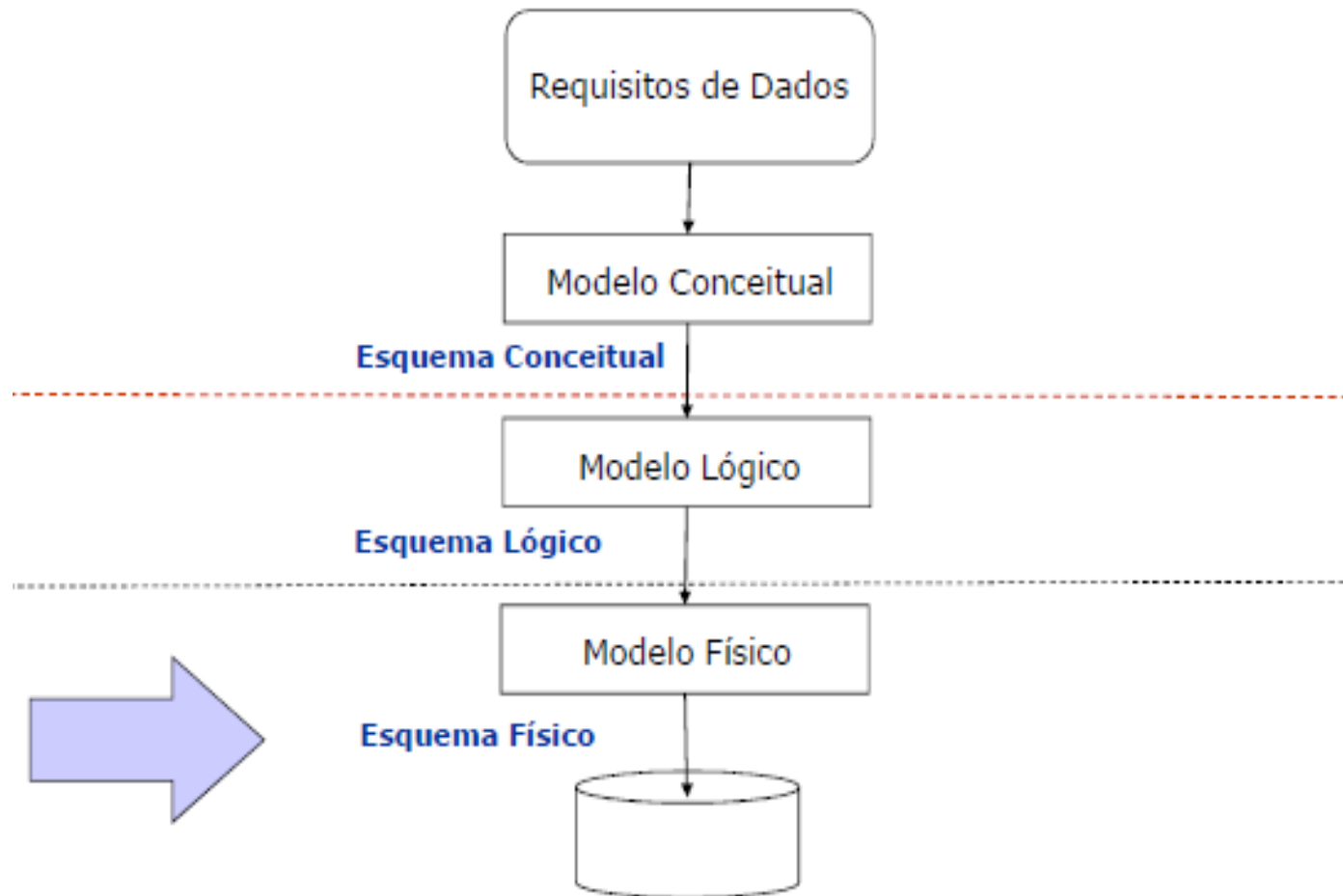
Professora: Juliana

16/8/2024

Roteiro

- ❑ Projeto de Banco de Dados
- ❑ Descrição e Tipos de instruções SQL
 - Instruções DDL
 - Instruções DML
 - Instruções DCL
- ❑ Introdução ao MySQL Workbench
- ❑ Introdução aos comandos DDL: Parte 1

Projeto de Banco de Dados



Descrição da SQL

- SQL - *Structured Query Language*- *Linguagem Estruturada de Consulta*
- Linguagem para acesso a Sistemas de Bases de Dados Relacionais (padrão para diversos produtos comerciais).
- Uma base de dados é como uma coleção de tabelas (Modelo Relacional).
- ***Características da linguagem SQL:***
 - Independente do fabricante do SGBD
 - Portabilidade entre plataformas
 - Redução de custos com treinamento
 - Inglês estruturado de alto nível
 - Definição de múltiplas visões dos dados

Tipos de Instruções SQL

- **Comandos DDL (*Data Definition Language*):**
 - comandos destinados a manutenção do esquema do BD;
 - Manutenção de objetos (tabelas, índices, colunas, etc);
 - Especificação de restrições de integridade.

Principais comandos:

- **CREATE** ➡ Cria objetos no esquema
- **ALTER** ➡ Altera objetos do esquema
- **DROP** ➡ Exclui objetos do esquema

Tipos de Instruções SQL (cont...)

- ❑ **Comandos DML (*Data Manipulation Language*):**
comandos destinados a manipulação dos dados do banco de dados.

Principais comandos:

- SELECT Seleciona linhas de dados
- INSERT Inclusão de dados
- UPDATE Alteração de dados
- DELETE Exclusão de dados
- COMMIT Confirma operações com dados
- ROLLBACK Cancela operações com dados

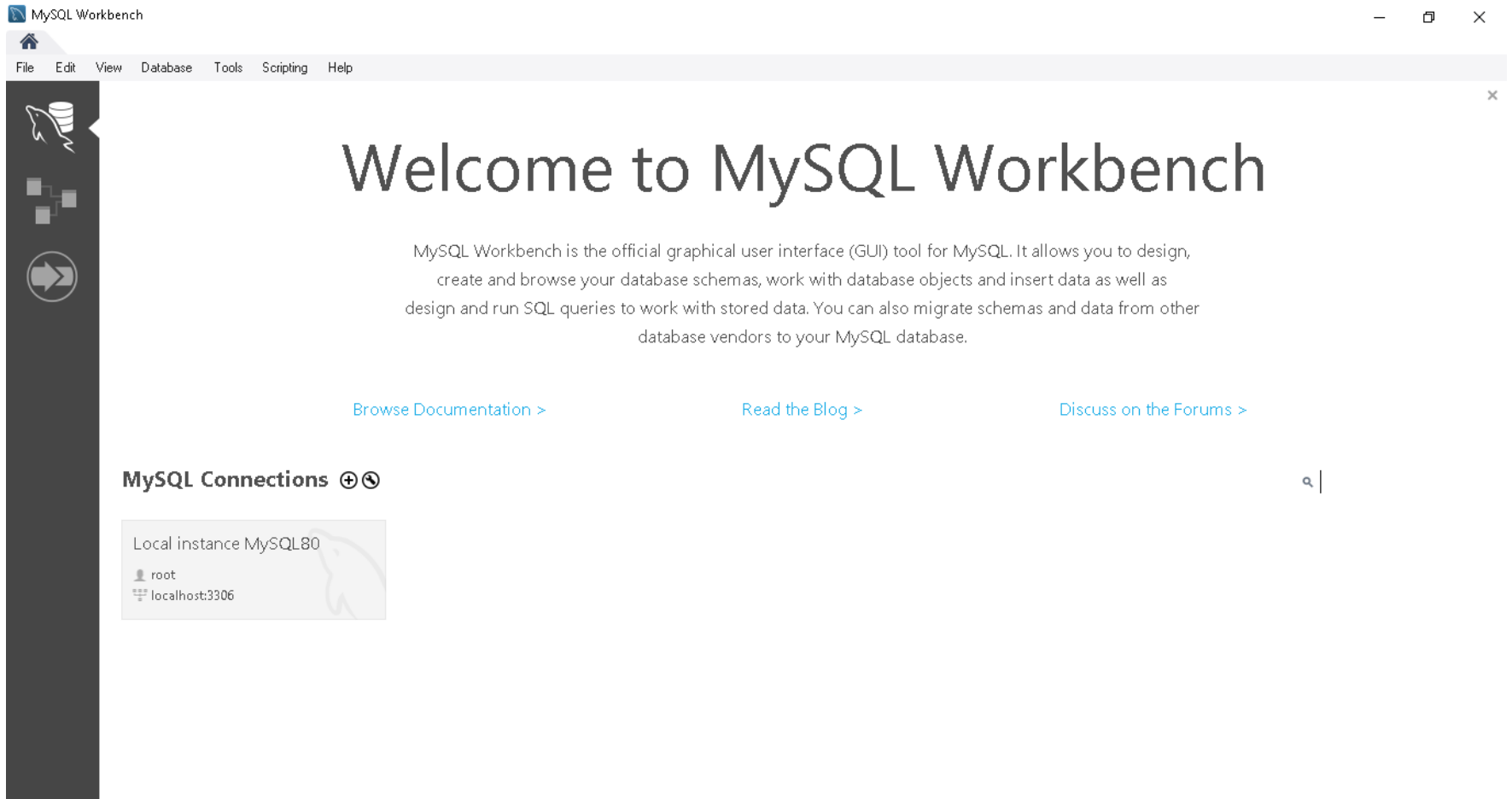
Tipos de Instruções SQL (cont...)

- ❑ **Comandos DCL (*Data Control Language*): comandos**
destinados ao controle de acesso aos dados, ou
seja, definição dos privilégios dos usuários.

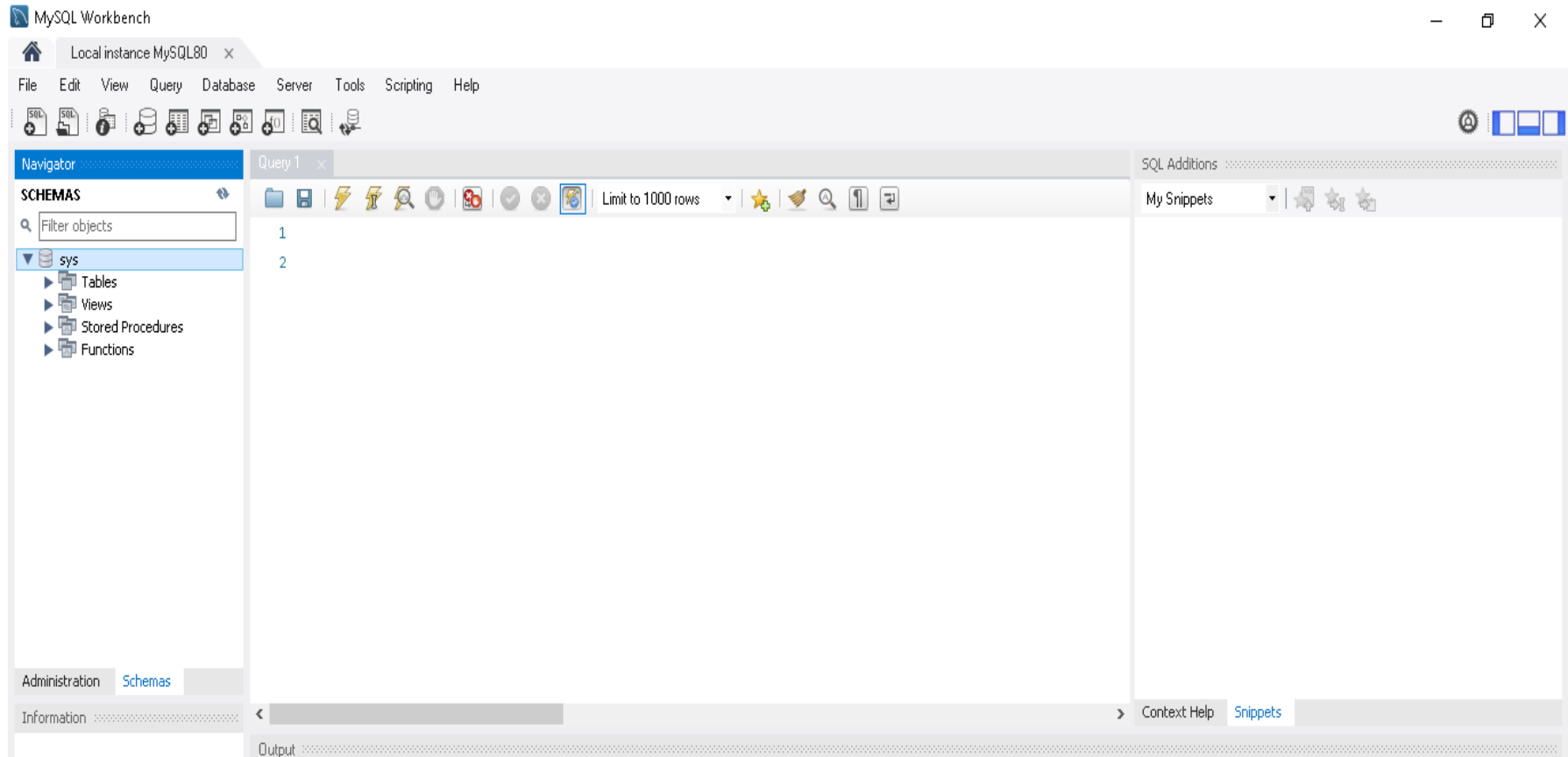
Principais comandos:

- GRANT → Concede privilégios aos usuários
- REVOKE → Revoga privilégios dos usuários

Introdução ao MySQL Workbench



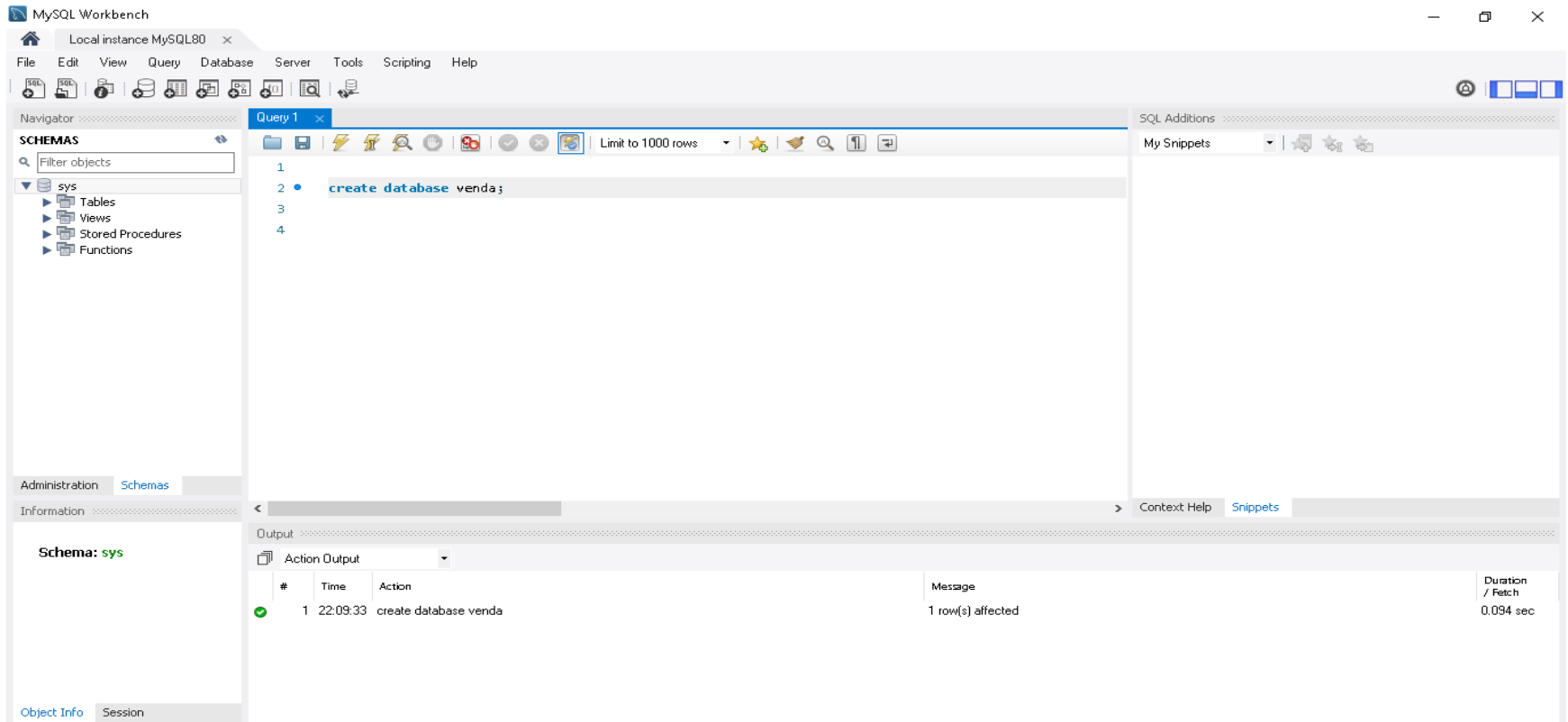
Introdução ao MySQL Workbench



Introdução ao MySQL Workbench

Criando um Banco de Dados

Sintaxe: create database <nome do Banco de dados>;



Introdução ao MySQL Workbench

Criando um Banco de Dados

Clicar para atualizar visualização de SCHEMAS.

The screenshot shows the MySQL Workbench interface. The left sidebar contains the 'Navigator' pane with a tree view of schemas. The 'sys' schema is expanded, showing its internal structure. The 'venda' schema is also visible. The main editor pane shows a SQL query: `create database venda;`. The bottom pane shows the 'Output' tab with a table of results.

Navigator

SCHEMAS

Filter objects

- sys
 - Tables
 - sys_config
 - Columns
 - Indexes
 - Foreign Keys
 - Triggers
 - Views
 - Stored Procedures
 - Functions
 - venda
 - Tables
 - Views
 - Stored Procedures
 - Functions

Query 1

```
1  
2 create database venda;  
3  
4
```

Output

#	Time	Action	Message	Duration / Fetch
✓ 1	22:09:33	create database venda	1 row(s) affected	0.094 sec



Comandos DDL

Parte 1

Create Table

□ Sintaxe:

CREATE TABLE TABELA

(

COLUNA TIPO [DEFAULT *exp*] [NULL | NOT NULL]

[CONSTRAINT da coluna],

...,

[CONSTRAINT da tabela]

);

Create Table (cont...)

```
CREATE TABLE TABELA  
(  
  COLUNA TIPO [DEFAULT exp]  
  [NULL | NOT NULL]  
  [CONSTRAINT da coluna],  
  ...,  
  [CONSTRAINT da tabela]  
);
```

□ Descrição da sintaxe:

- **TABELA:** é o nome da tabela
- **COLUNA:** é o nome da coluna
- **TIPO:** tipo de dados da coluna + tamanho
- **[DEFAULT exp]:** *especifica o valor que será utilizado quando um dado for omitido durante uma inclusão*
- **[NULL | NOT NULL]:** define se a coluna aceitará ou não valores nulos.
- **[CONSTRAINT]:** que especifica as restrições para uma coluna ou para a tabela.

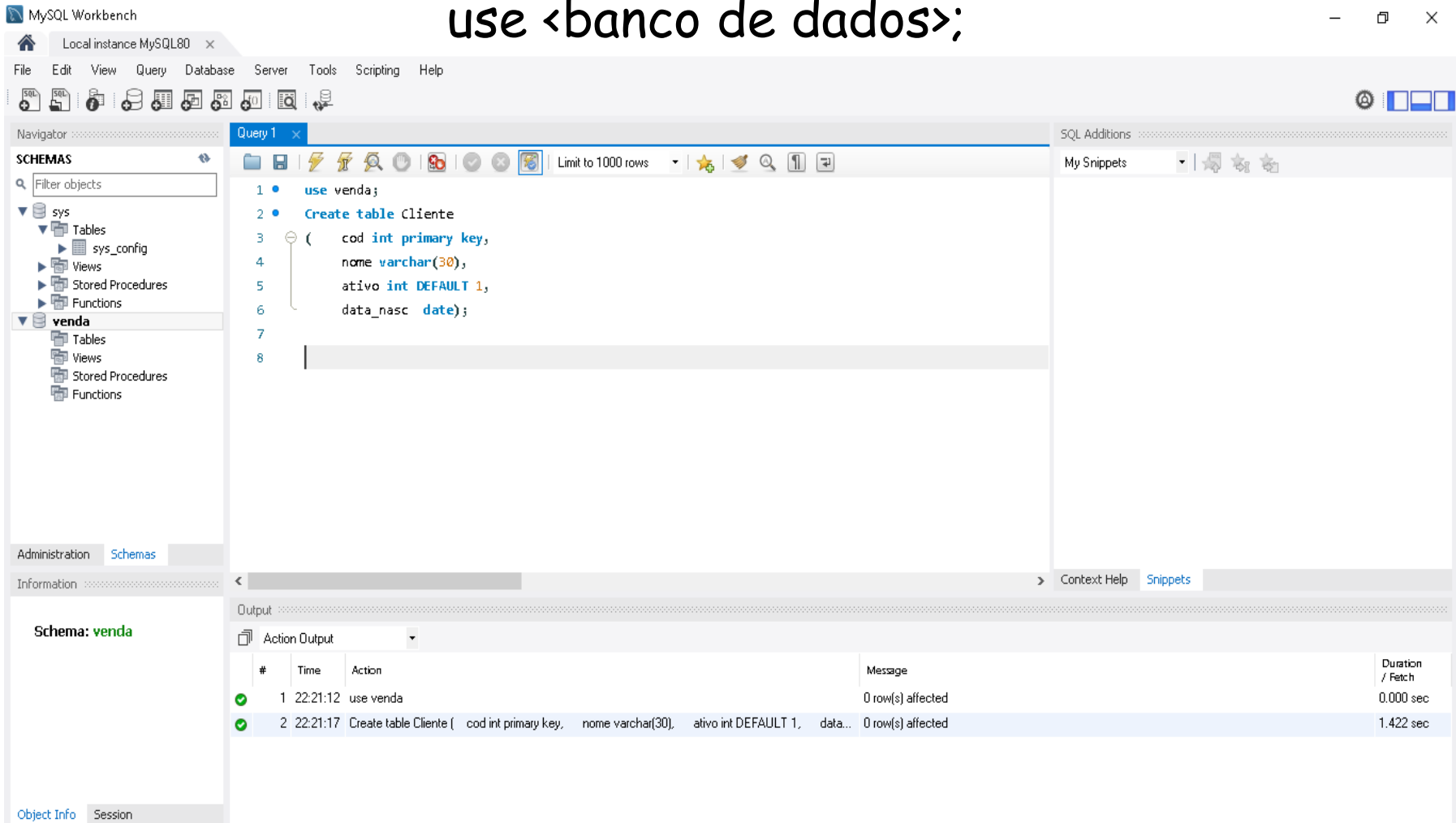
Exemplo: Create Table

Create table Cliente

```
( cli_cod int primary key,  
  cli_nome varchar(30),  
  cli_ativo int DEFAULT 1,  
  cli_data_nasc date);
```

Exemplo: Create Table

Sintaxe para selecionar um banco de dados:
`use <banco de dados>;`



The screenshot displays the MySQL Workbench interface. The 'Query 1' editor contains the following SQL code:

```
1 • use venda;  
2 • Create table Cliente  
3 • (  
4 •     cod int primary key,  
5 •     nome varchar(30),  
6 •     ativo int DEFAULT 1,  
7 •     data_nasc date);  
8 •
```

The 'Navigator' pane on the left shows the database structure, with the 'venda' database selected. The 'Output' pane at the bottom shows the execution results:

#	Time	Action	Message	Duration / Fetch
✓ 1	22:21:12	use venda	0 row(s) affected	0.000 sec
✓ 2	22:21:17	Create table Cliente (cod int primary key, nome varchar(30), ativo int DEFAULT 1, data...	0 row(s) affected	1.422 sec

Verificando as restrições (constraints)

❑ `select * from information_schema.table_constraints;`

MySQL Workbench

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

sys

venda

Tables

cliente

Views

Stored Procedures

Functions

Query 1 x

Limit to 1000 rows

```
1 • select * from information_schema.table_constraints;  
2  
3
```

Result Grid

	CONSTRAINT_CATALOG	CONSTRAINT_SCHEMA	CONSTRAINT_NAME	TABLE_SCHEMA	TABLE_NAME	CONSTRAINT_TYPE
def	performance_schema	PRIMARY		performance_s...	user_defined...	PRIMARY KEY
def	performance_schema	PRIMARY		performance_s...	user_variable...	PRIMARY KEY
def	performance_schema	USER		performance_s...	users	UNIQUE
def	performance_schema	PRIMARY		performance_s...	variables_by...	PRIMARY KEY
def	sys	PRIMARY		sys	sys_config	PRIMARY KEY
def	venda	PRIMARY		venda	cliente	PRIMARY KEY

Administration Schemas

Information

_constraints 1 x

Read Only Context Help Snippets

Output

Action Output

#	Time	Action	Message	Duration / Fetch
✓ 1	23:12:30	use venda	0 row(s) affected	0.000 sec
✓ 2	23:12:39	select * from information_schema.table_constraints LIMIT 0, 1000	135 row(s) returned	0.719 sec / 0.000 se

Verificando as restrições (constraints)

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

sys

venda

Tables

cliente

Views

Stored Procedures

Functions

Query 1

```
1 select * from information_schema.table_constraints
2 where table_name='cliente';
3
4
```

Limit to 1000 rows

Result Grid

	CONSTRAINT_CATALOG	CONSTRAINT_SCHEMA	CONSTRAINT_NAME	TABLE_SCHEMA	TABLE_NAME	CONSTRAINT_TYPE	EN
▶	def	venda	PRIMARY	venda	cliente	PRIMARY KEY	YES

Administration Schemas

Information

Schema: venda

table_constraints 2

Read Only Context Help Snippets

Output

Action Output

#	Time	Action	Message	Duration / Fetch
✓ 1	23:12:30	use venda	0 row(s) affected	0.000 sec
✓ 2	23:12:39	select * from information_schema.table_constraints LIMIT 0, 1000	135 row(s) returned	0.719 sec / 0.000 sec
✓ 3	23:14:22	select * from information_schema.table_constraints where table_name='cliente' LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

Inserindo linhas na tabela

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

sys

venda

Tables

cliente

Views

Stored Procedures

Functions

Query 1

```
1 • Create table cliente
2 ( cli_cod int primary key,
3   cli_nome varchar(30),
4   cli_ativo int DEFAULT 1,
5   cli_data_nasc date);
6
7 • insert into cliente values (1,'João',0,'1978-10-30');
8 • insert into cliente(cli_cod,cli_nome,cli_data_nasc) values (2,'Maria','1978-10-30');
9
```

Result Grid

	cli_cod	cli_nome	cli_ativo	cli_data_nasc
1	João	0	1978-10-30	
2	Maria	1	1978-10-30	
*	NULL	NULL	NULL	NULL

Administration Schemas

Information

Schema: venda

cliente 5

Apply Revert Context Help Snippets

Output

Action Output

#	Time	Action	Message	Duration / Fetch
---	------	--------	---------	------------------

Usando AUTO_INCREMENT

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

sys

venda

Tables

Views

Stored Procedures

Functions

Query 1

```
1 • Create table cliente
2 ( cli_cod int primary key AUTO_INCREMENT,
3   cli_nome varchar(30),
4   cli_ativo int DEFAULT 1,
5   cli_data_nasc date);
6
7 • insert into cliente(cli_nome,cli_data_nasc) values ('Maria','1978-10-30');
8
9 • select * from cliente;
```

Result Grid

	cli_cod	cli_nome	cli_ativo	cli_data_nasc
▶	1	Maria	1	1978-10-30
*	NULL	NULL	NULL	NULL

Administration Schemas

Information

Schema: venda

cliente 7

Apply Revert Context Help Snippets

Output

Action Output

#	Time	Action	Message	Duration / Fetch
✓ 1	00:25:09	insert into cliente(cli_nome,cli_data_nasc) values ('Maria','1978-10-30')	1 row(s) affected	0.109 sec
✓ 2	00:25:13	select * from cliente LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

Tipos de Dados do MySQL

Verificar em: <https://dev.mysql.com/doc/refman/8.0/en/data-types.html>

The screenshot shows a web browser displaying the MySQL 8.0 Reference Manual page for Chapter 11 Data Types. The browser's address bar shows the URL <https://dev.mysql.com/doc/refman/8.0/en/data-types.html>. The MySQL logo and navigation links (MYSQL.COM, DOWNLOADS, DOCUMENTATION, DEVELOPER ZONE) are at the top. Below the navigation bar, there's a blue header with links to MySQL Server, MySQL Enterprise, Workbench, InnoDB Cluster, MySQL NDB Cluster, Connectors, and More. The main content area is titled "Chapter 11 Data Types" and includes a "Table of Contents" section with links to various data types: 11.1 Numeric Data Types, 11.2 Date and Time Data Types, 11.3 String Data Types, 11.4 Spatial Data Types, 11.5 The JSON Data Type, 11.6 Data Type Default Values, 11.7 Data Type Storage Requirements, 11.8 Choosing the Right Type for a Column, and 11.9 Using Data Types from Other Database Engines. A sidebar on the left contains a search bar and a list of links for the MySQL 8.0 Reference Manual, including Preface and Legal Notices, General Information, Installing and Upgrading MySQL, Tutorial, MySQL Programs, MySQL Server Administration, Security, Backup and Recovery, Optimization, Language Structure, Character Sets, Collations, Unicode, and Data Types. A dropdown menu for "version 8.0" is visible in the top right corner.

MySQL 8.0 Reference Manual / Data Types

version 8.0

Chapter 11 Data Types

Table of Contents

- 11.1 Numeric Data Types
- 11.2 Date and Time Data Types
- 11.3 String Data Types
- 11.4 Spatial Data Types
- 11.5 The JSON Data Type
- 11.6 Data Type Default Values
- 11.7 Data Type Storage Requirements
- 11.8 Choosing the Right Type for a Column
- 11.9 Using Data Types from Other Database Engines

MySQL supports [SQL](#) data types in several categories: numeric types, date and time types, string (character and byte) types, spatial types, and the [JSON](#) data type. This chapter provides an overview and more detailed description of the properties of the types in each category, and a summary of the data type storage requirements. The initial overviews are intentionally brief. Consult the more detailed descriptions for additional information about particular data types, such as the permissible formats in which you can

Tipos de Dados do MySQL

Type	Storage (Bytes)	Minimum Value Signed	Minimum Value Unsigned	Maximum Value Signed	Maximum Value Unsigned
TINYINT	1	-128	0	127	255
SMALLINT	2	-32768	0	32767	65535
MEDIUMINT	3	-8388608	0	8388607	16777215
INT	4	-2147483648	0	2147483647	4294967295
BIGINT	8	-2^{63}	0	$2^{63}-1$	$2^{64}-1$

Restrições (*Constraints*)

- As restrições impõem regras ao banco de dados.
- Devem ser definidas para evitar que dados incorretos (inválidos) sejam inseridos nas tabelas, garante a consistência dos dados.
- São válidos os seguintes tipos de restrições:
 - **NOT NULL** → Obrigatoriedade de valor no campo
 - **UNIQUE** → Unicidade de valor no campo
 - **PRIMARY KEY** → Definição de chave primária
 - **FOREIGN KEY** → Definição de chave estrangeira
 - **CHECK** → Validação de valor dentro de domínio

Constraint *UNIQUE KEY* (UK)

- Define que cada valor da coluna, ou conjunto de colunas, seja sempre único dentro da tabela.
- Uma coluna especificada com uma restrição *UNIQUE KEY* é chamada de chave exclusiva.
- Se a restrição *UNIQUE KEY* for formada por mais de uma coluna, o grupo de colunas é chamado de chave exclusiva composta.
- Pode ser definida tanto a nível de coluna quanto a nível de tabela.

Exemplo: UNIQUE KEY a nível de tabela

```
CREATE TABLE ALUNO  
(COD_ALUNO int primary key,  
MATRICULA int NOT NULL,  
NOME varchar(40),  
CPF varchar(11),  
Constraint uk_cpf Unique(CPF));
```

Exemplo: UNIQUE KEY a nível de coluna

```
CREATE TABLE ALUNO  
(  
    COD_ALUNO int NOT NULL,  
    MATRICULA int NOT NULL,  
    NOME varchar(40),  
    CPF int Unique  
);
```

Constraint PRIMARY KEY (PK)

- Define uma ou mais colunas como chave primária da coluna.
- Portanto, essa restrição identifica exclusivamente cada linha de dados em uma tabela.
- Nenhuma coluna com definição de uma constraint PRIMARY KEY aceitará valores nulos.
- Uma constraint UNIQUE é automaticamente criada para uma coluna (ou conjunto de colunas) PRIMARY KEY.
- Pode ser definida a nível de coluna e tabela.

Exemplo: PRIMARY KEY a nível de tabela

```
CREATE TABLE ALUNO  
(COD_ALUNO int,  
  MATRICULA int NOT NULL,  
  NOME varchar(40),  
  CPF int,  
  Primary Key(COD_ALUNO));
```

Exemplo: PRIMARY KEY a nível de coluna


```
CREATE TABLE ALUNO  
( COD_ALUNO int Primary Key,  
  MATRICULA int NOT NULL,  
  NOME varchar(40),  
  CPF int  
);
```

OBS: A definição de uma chave primária composta só pode ser realizada a nível de tabela!

Exemplo: PRIMARY KEY COMPOSTA

```
create table itempedido (  
    ped_cod int,  
    pro_cod int,  
    ite_quantidade int,  
    ite_valor decimal(7,2),  
    Foreign Key(ped_cod) references pedido(ped_cod),  
    Foreign Key(pro_cod) references produto(prod_cod),  
    primary key (ped_cod,pro_cod));
```

```
create table itempedido (    ped_cod int,    pro_cod
int,    ite_quantidade int,    ite_valor
decimal(7,2),    constraint fk_ped_cod Foreign
Key(ped_cod) references pedido(ped_cod),
constraint fk_pro_cod Foreign Key(pro_cod)
references produto(prod_cod), primary key
(ped_cod,pro_cod));
```



```
CREATE TABLE PRODUTO
(  prod_cod int primary key,
   pro_nome varchar(30) not null,
   pro_quantidade int,
   pro_valor decimal(9,2)
);
```




```
CREATE TABLE PEDIDO
```

```
( ped_cod int primary key,  
  ped_data date,  
  ped_valor_total decimal(9,2)  
);
```

Constraint *FOREIGN KEY* (FK)

- Define uma restrição de integridade referencial, designando uma ou mais colunas como chave estrangeira.
- Um valor de chave estrangeira deve corresponder a um valor existente na tabela referida ou valor NULL.
- Pode ser definida a nível de tabela

Exemplo: FOREIGN KEY a nível de tabela

```
CREATE TABLE ALUNO
```

```
(COD_ALUNO int,
```

```
  MATRICULA int NOT NULL,
```

```
  COD_CURSO int,
```

```
  CPF varchar(11),
```

```
  NOME varchar(40),
```

```
Foreign Key(COD_CURSO)References CURSO (COD_CURSO));
```

Constraint CHECK (CK)

- Define uma condição (regra) que cada valor da coluna de obedecer.

Exemplos:

- Restrição de valor mínimo para salário, ou seja, só aceitar valores maiores que 500,00.
 - Restrição de campo Estado Civil, permitindo apenas valores dentro do domínio ('S','C','V','D','S','U')
- Pode ser definida a nível de coluna e tabela

Exemplo: CHECK a nível de tabela

CREATE TABLE ALUNO

(COD_ALUNO int PRIMARY KEY,

MATRICULA int UNIQUE,

CPF varchar(11),

NOME varchar(40),

ESTADO_CIVIL char(1) ,

Constraint CK_ ESTADO_CIVIL

Check(ESTADO_CIVIL in ('S','C','V','D','S','U'))

Exemplo: CHECK a nível de coluna

```
CREATE TABLE ALUNO  
( COD_ALUNO int PRIMARY KEY,  
  MATRICULA int UNIQUE,  
  CPF varchar(11),  
  NOME varchar(40),  
  SEXO char(1) Constraint CK_SEXO Check (ESTADO_CIVIL  
in ('S','C','V','D','S','U'))
```