



Quem se prepara, não para.



Banco de Dados

2º período

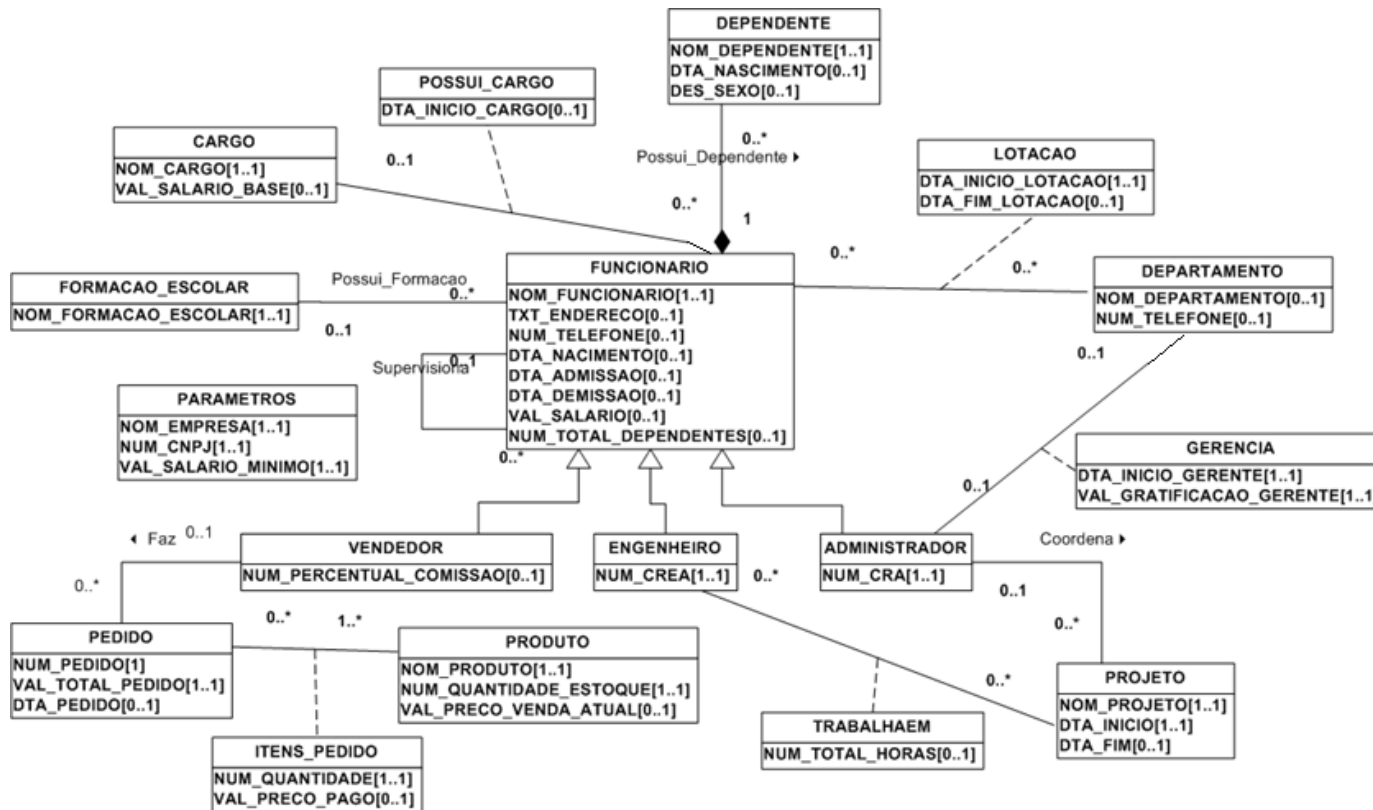
Prof. Dr. João Paulo Aramuni



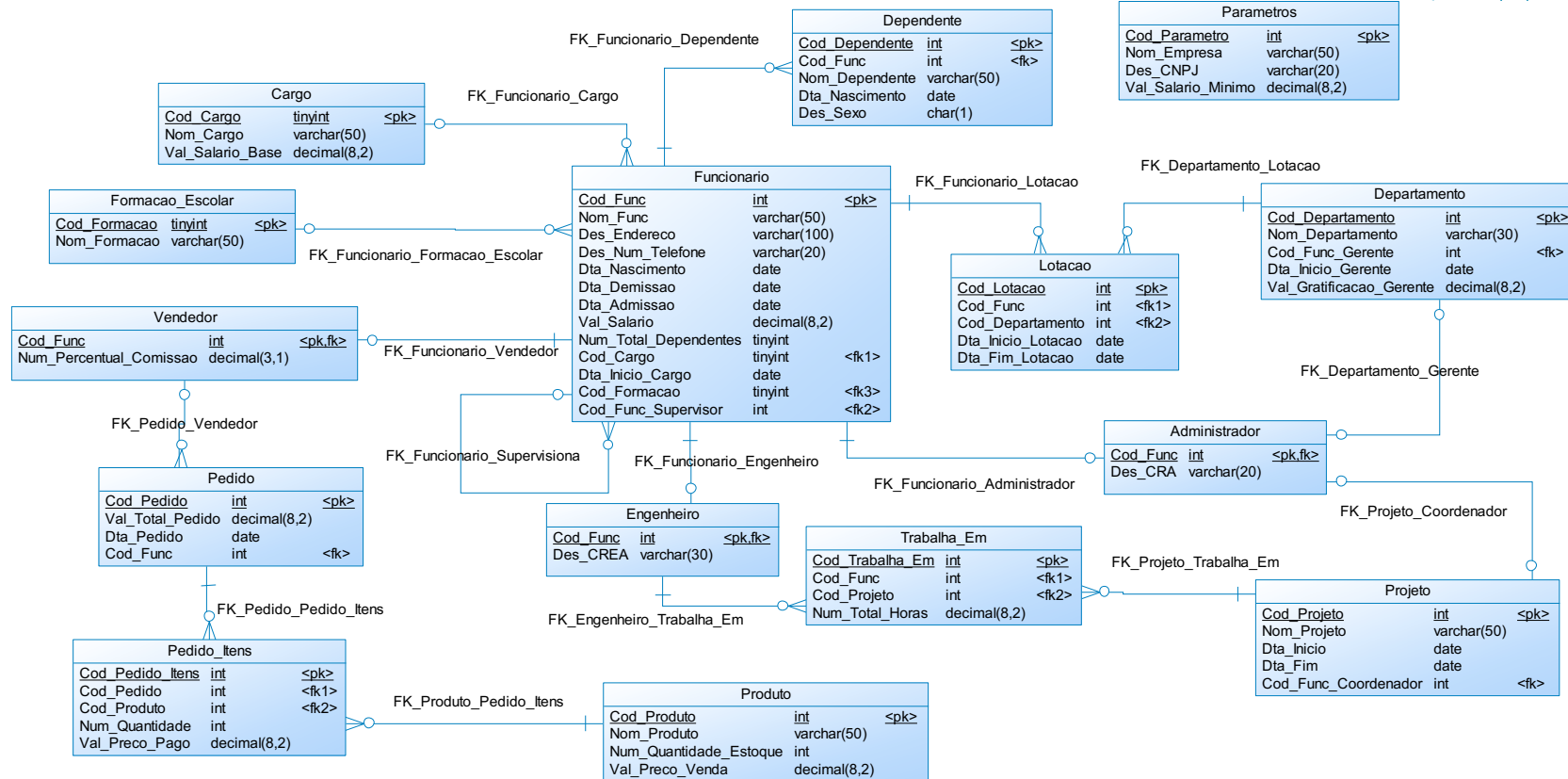
Projeto Físico de Bando de Dados

Aula 08

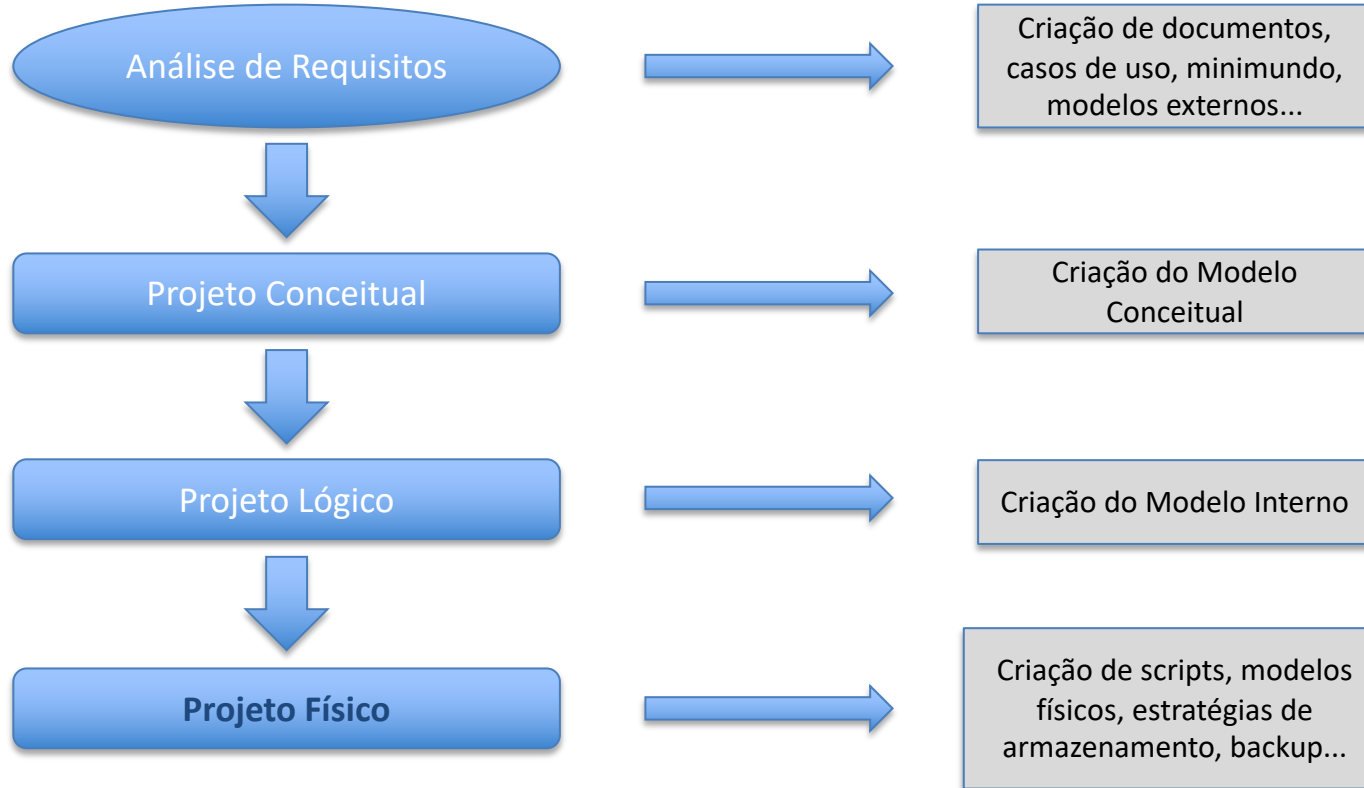
Projeto Lógico de BD - DCP Empresa



Projeto Lógico de BD - DER Empresa



Projeto Físico de BD



DB-Engines

- O DB-Engines Ranking classifica os sistemas de gerenciamento de banco de dados de acordo com sua popularidade. O ranking é atualizado mensalmente.

419 systems in ranking, July 2023

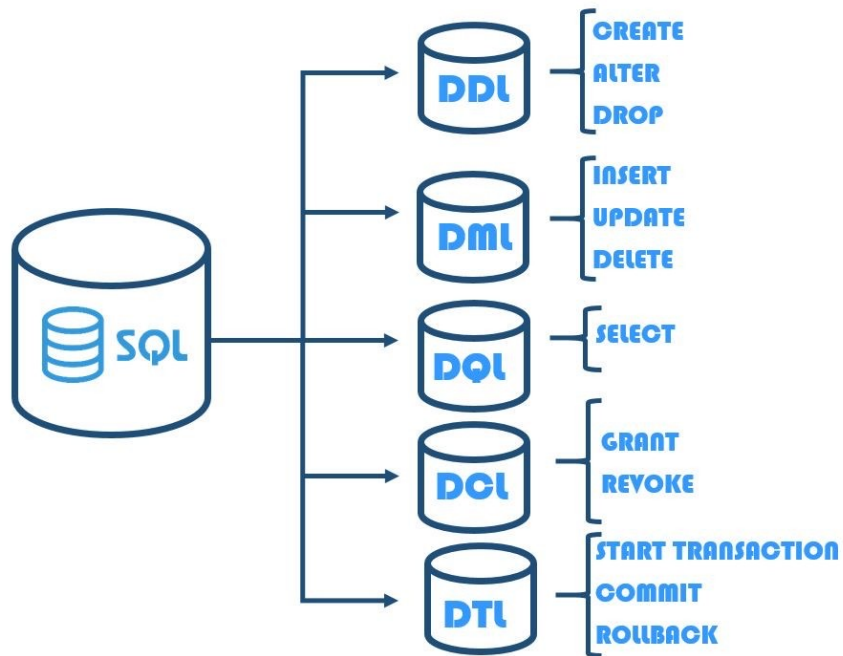
Rank			DBMS	Database Model	Score		
Jul 2023	Jun 2023	Jul 2022			Jul 2023	Jun 2023	Jul 2022
1.	1.	1.	Oracle +	Relational, Multi-model ⓘ	1256.01	+24.54	-24.28
2.	2.	2.	MySQL +	Relational, Multi-model ⓘ	1150.35	-13.59	-44.53
3.	3.	3.	Microsoft SQL Server +	Relational, Multi-model ⓘ	921.60	-8.47	-20.53
4.	4.	4.	PostgreSQL +	Relational, Multi-model ⓘ	617.83	+5.01	+1.96
5.	5.	5.	MongoDB +	Document, Multi-model ⓘ	435.49	+10.13	-37.49
6.	6.	6.	Redis +	Key-value, Multi-model ⓘ	163.76	-3.59	-9.86
7.	7.	7.	IBM Db2	Relational, Multi-model ⓘ	139.81	-5.07	-21.40
8.	8.	8.	Elasticsearch	Search engine, Multi-model ⓘ	139.59	-4.16	-14.74
9.	9.	9.	Microsoft Access	Relational	130.72	-3.73	-14.37
10.	10.	10.	SQLite +	Relational	130.20	-1.02	-6.48
11.	11.	↑ 13.	Snowflake +	Relational	117.69	+3.55	+18.53
12.	12.	↓ 11.	Cassandra +	Wide column	106.53	-2.03	-7.88
13.	13.	↓ 12.	MariaDB +	Relational, Multi-model ⓘ	96.10	-1.21	-16.42
14.	14.	14.	Splunk	Search engine	87.12	-2.34	-11.09
15.	↑ 16.	15.	Microsoft Azure SQL Database	Relational, Multi-model ⓘ	78.96	-0.01	-5.94
16.	↓ 15.	16.	Amazon DynamoDB +	Multi-model ⓘ	78.81	-1.10	-5.13
17.	17.	17.	Hive	Relational	72.87	-2.65	-6.61
18.	18.	↑ 22.	Databricks	Multi-model ⓘ	68.47	+2.65	+17.25
19.	19.	↓ 18.	Teradata	Relational, Multi-model ⓘ	60.25	-2.39	-10.67
20.	20.	↑ 24.	Google BigQuery +	Relational	55.42	+0.78	+6.53

Linguagem SQL - *Structured Query Language*

- SQL é uma linguagem padrão de alto nível (não procedural, como HTML e CSS) para criar e manipular dados em tabelas em um **SGBD relacional**.
- O padrão da linguagem é mantido pela **ISO** (*International Organization for Standardization*).
- Versões SQL lançadas: SQL86, SQL89, SQL92, SQL99, SQL2003, SQL2008, SQL2016.

Projeto Físico de BD

Divisão da Linguagem SQL

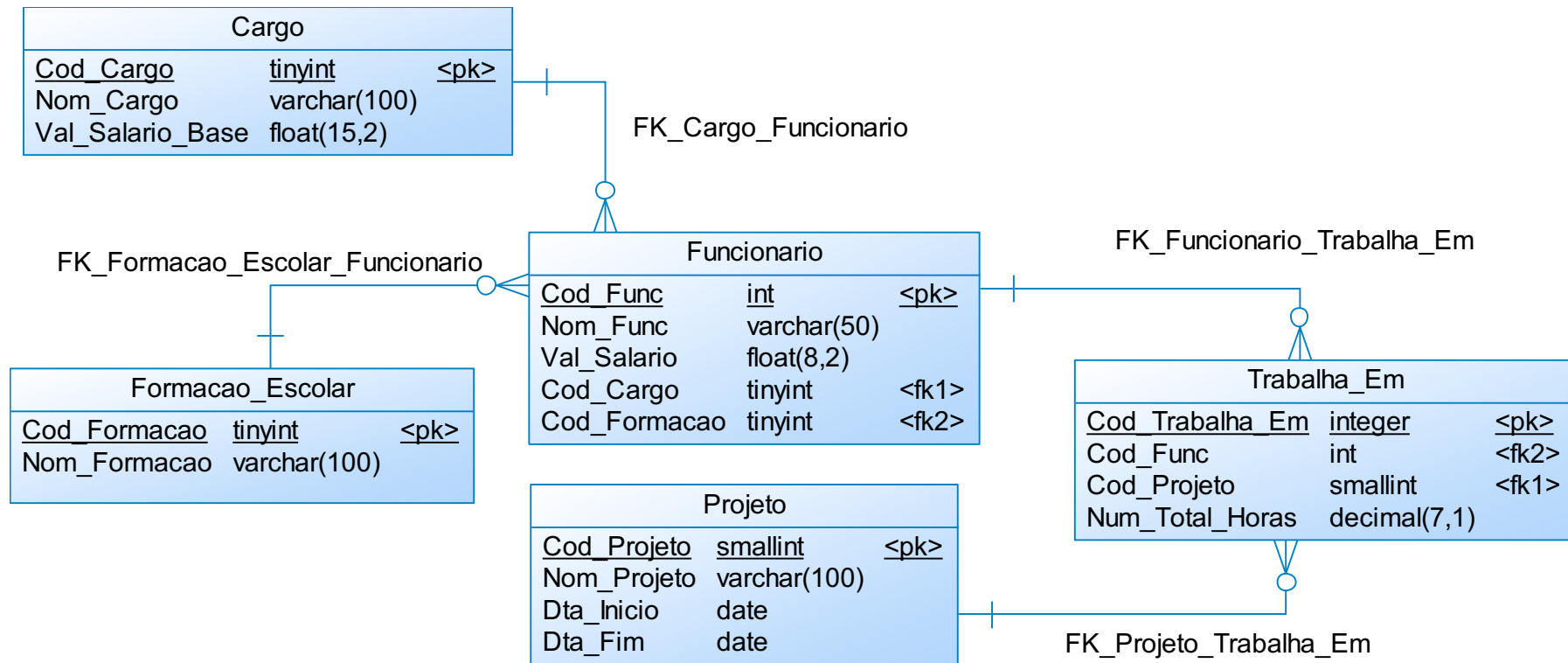


Projeto Físico de BD

- **DDL** (*Data Definition Language*): Linguagem de Definição de Dados. Essa parte do SQL é responsável por definir e gerenciar a estrutura dos objetos de banco de dados, como tabelas, índices, visões e outras estruturas. Os comandos DDL mais comuns são CREATE, ALTER e DROP.
- **DML** (*Data Manipulation Language*): Linguagem de Manipulação de Dados. Essa parte do SQL é usada para manipular os dados dentro das tabelas do banco de dados. Os comandos DML mais comuns são INSERT, UPDATE e DELETE.
- **DQL** (*Data Query Language*): Linguagem de Consulta de Dados. Essa parte do SQL é usada para realizar consultas e recuperar informações específicas do banco de dados. O comando DQL mais comum é o SELECT.
- **DCL** (*Data Control Language*): Linguagem de Controle de Dados. Essa parte do SQL é usada para gerenciar os direitos de acesso e permissões no banco de dados. Os comandos DCL mais comuns são GRANT e REVOKE.
- **DTL** (*Data Transaction Language*): Linguagem de Transação de Dados. A DTL é uma parte importante do SQL que permite ao programador controlar transações no banco de dados, garantindo a integridade dos dados e a consistência dos resultados. Os comandos DCL mais comuns são COMMIT, ROLLBACK e SAVEPOINT.

Projeto Físico de BD

Considere o seguinte DER relacional



Projeto Físico de BD

- **DDL** (*Data Definition Language*): Linguagem de Definição de Dados.

-- Comentário SQL

-- Para criar um DATABASE:

CREATE DATABASE DB_TESTE;

-- Para usar esse database:

USE DB_TESTE;

Obs: A linguagem SQL não é *case sensitive*, ou seja, tanto faz escrevermos maiúsculo ou minúsculo.

Projeto Físico de BD

- **DDL** (*Data Definition Language*): Linguagem de Definição de Dados.

-- Exemplo de criação de tabela

CREATE TABLE Cargo

```
(  
  Cod_Cargo          tinyint not null Primary Key,  
  Nom_Cargo          varchar(100) not null,  
  Val_Salario_Base   float(15,2)  
);
```

Obs: Neste float, o número 15 representa a precisão total do número, ou seja, o número máximo de dígitos que o valor pode conter, incluindo os dígitos à esquerda e à direita do ponto decimal. O número 2 representa a escala, ou seja, o número máximo de dígitos que podem ser armazenados à direita do ponto decimal.

Projeto Físico de BD

- **DDL** (*Data Definition Language*): Linguagem de Definição de Dados.

-- Exemplo de criação de tabela

CREATE TABLE Formacao_Escolar

(

Cod_Formacao **tinyint not null Primary Key,**

Nom_Formacao **varchar(100) not null**

);

Dica: Você pode salvar seus scripts SQL usando a extensão de arquivo .sql.

Projeto Físico de BD

- **DDL** (*Data Definition Language*): Linguagem de Definição de Dados.

```
-- Criando a tabela Livro
CREATE TABLE [Livro]
(
    [LivroId] INTEGER NOT NULL,
    [Titulo] NVARCHAR(160) NOT NULL,
    [Autor] NVARCHAR(160) NOT NULL,
    [ISBN] NVARCHAR(160) NOT NULL,
    [Quantidade] INTEGER NOT NULL,
    CONSTRAINT [PK_Livro] PRIMARY KEY ([LivroId])
);

CREATE UNIQUE INDEX [IPK_Livro] ON [Livro]([LivroId]);

-- Criando a tabela Aluno
CREATE TABLE [Aluno]
(
    [Matricula] INTEGER NOT NULL,
    [Nome] NVARCHAR(160) NOT NULL,
    CONSTRAINT [PK_Aluno] PRIMARY KEY ([Matricula])
);

CREATE UNIQUE INDEX [IPK_Aluno] ON [Aluno]([Matricula]);

-- Criando a tabela Emprestimo
CREATE TABLE [Emprestimo]
(
    [EmprestimoId] INTEGER NOT NULL,
    [LivroId] INTEGER NOT NULL,
    [Matricula] INTEGER NOT NULL,
    [DataHora_Emprestimo] DATETIME NOT NULL,
    CONSTRAINT [PK_Emprestimo] PRIMARY KEY ([EmprestimoId]),
    CONSTRAINT [FK_Emprestimo_Livro] FOREIGN KEY ([LivroId]) REFERENCES Livro(LivroId),
    CONSTRAINT [FK_Emprestimo_Aluno] FOREIGN KEY ([Matricula]) REFERENCES Aluno(Matricula)
);

CREATE UNIQUE INDEX [IPK_Emprestimo] ON [Emprestimo]([EmprestimoId]);
```

Um exemplo de criação de tabelas para o BD **SQLite**.

Datatypes no SQLite:

<https://www.sqlite.org/datatype3.html>



```
-- Para deletar as tabelas:
DROP TABLE Livro;
DROP TABLE Aluno;
DROP TABLE Emprestimo;
```

MariaDB - Datatypes

Dados Numéricos Inteiros		
Tipo	Escopo com sinal	Escopo sem sinal
Tinyint	-128 a 127	0 a 255
Smallint	-32.768 a 32.767	0 a 65.535
Mediumint	-8.388.608 a 8.388.607	0 a 16.777.215
Int	-2.147.483.648 a 2.147.483.647	0 a 4.294.967.295
Bigint	-9.223.372.036.854.775.808 a 9.223.372.036.854.775.807	0 a 18.446.744.073.709.551.615

Dados Numéricos (Bit e Boolean)	
Tipo	Numero máximo de bytes
bit	1
bool ou boolean	1

Dados de Texto Não-Binário	
Tipo de texto	Numero máximo de bytes
Tinytext	255
Text	65.535
MediumText	16.777.215
LongText	4.294.967.295
Varchar	65.535
Char	255

Dados Temporais		
Tipo	Formato padrão	Valores permitidos
Date	AAAA-MM-DD	1000-01-01 a 9999-12-31
Datetime	AAAA-MM-DD HH:MI:SS	1000-01-01 00:00:00 a 9999-12-31 23:59:00
Timestamp	AAAA-MM-DD HH:MI:SS	1970-01-01 00:00:00 a 2037-12-31 23:59:00
Year	AAAA	1901 a 2155
Time	HHH:MI:SS	-838:59:59 a 838:59:59

Dados de Texto Binário	
Tipo de texto	Numero máximo de bytes
Tinyblob	255
Blob	65.535
Mediumblob	16.777.215
Longblob	4.294.967.295
Varbinary	65.535
Binary	255

Dados Numéricos de Ponto Flutuante e Ponto Fixo	
Tipo	Escopo numérico
Float(p,e)	-3,402823466E+38 a -1,175494351E-38 e de 1.175494351E-38 a 3,402823466E+38
Double(p,e)	-1,7976931348623157E+308 a -2,2250738585072014E-308 e de 2,2250738585072014E-308 a 1.7976931348623157E+308
Decimal(p,e)	-1,7976931348623157E+308 a -2,2250738585072014E-308 e de 2,2250738585072014E-308 a 1.7976931348623157E+308



Obrigado!

joao.aramuni@newtonpaiva.br