

Banco de Dados

ATIVIDADES DE HOJE

Horário das atividades

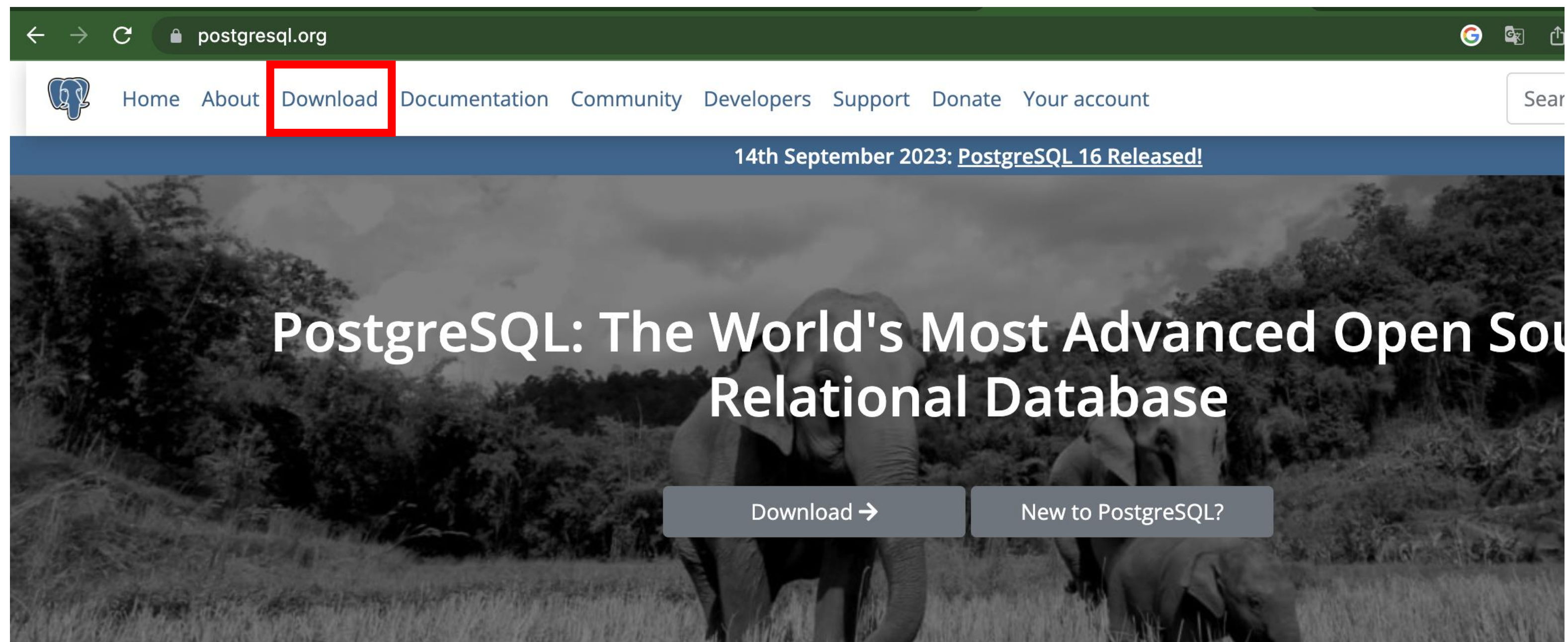
- 13h30 Faísca
- 13h45 Banco de dados
- 15h00 Intervalo
- 15h30 Palestra



TUTORIAL DE INSTALAÇÃO POSTGRES NO MAC OS

INSTALAÇÃO POSTGRES

ACESSE: <https://www.postgresql.org/>



New to PostgreSQL?

PostgreSQL is a powerful, open source object-relational database system with over 35 years of active development that has earned it a strong reputation for reliability, feature robustness, and performance.



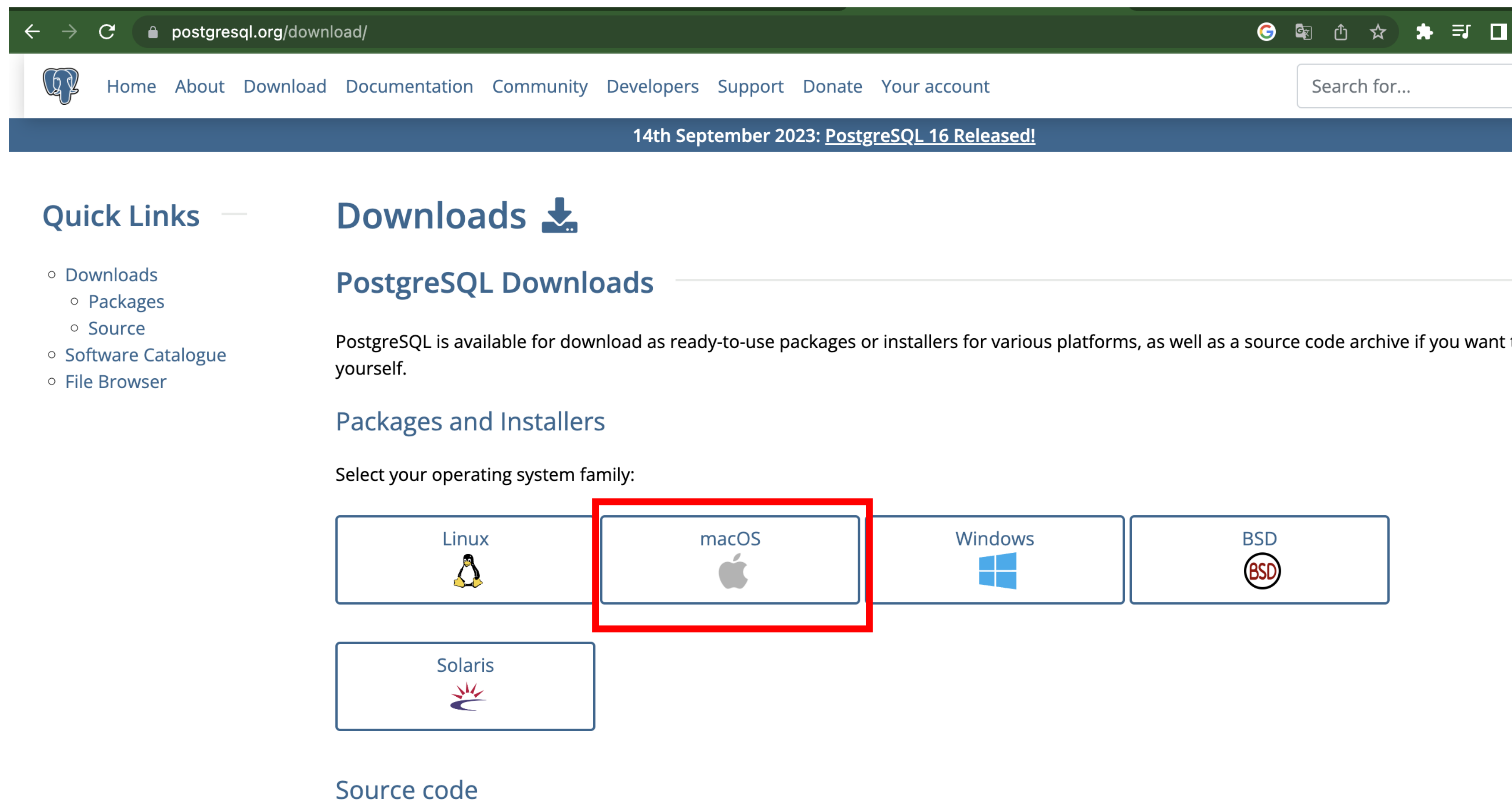
Latest Releases

2023-09-14 - PostgreSQL 16 Released!

The PostgreSQL Global Development Group today announce

INSTALAÇÃO POSTGRES

ACESSE: <https://www.postgresql.org/>



The screenshot shows the PostgreSQL website's download page. The browser's address bar displays 'postgresql.org/download/'. The navigation menu includes links for Home, About, Download, Documentation, Community, Developers, Support, Donate, and Your account. A search bar is located on the right. A blue banner at the top of the main content area reads '14th September 2023: PostgreSQL 16 Released!'. On the left, a 'Quick Links' sidebar lists: Downloads, Packages, Source, Software Catalogue, and File Browser. The main section is titled 'Downloads' with a download icon. Below this is 'PostgreSQL Downloads', followed by a paragraph stating that PostgreSQL is available for download as ready-to-use packages or installers for various platforms, as well as a source code archive. The 'Packages and Installers' section prompts the user to 'Select your operating system family:' and displays five buttons: Linux (with a penguin icon), macOS (with an Apple icon and highlighted by a red border), Windows (with a Windows logo icon), BSD (with a BSD logo icon), and Solaris (with a Solaris logo icon). At the bottom of this section is a link for 'Source code'.

Quick Links —

- Downloads
- Packages
- Source
- Software Catalogue
- File Browser


Downloads


PostgreSQL Downloads


PostgreSQL is available for download as ready-to-use packages or installers for various platforms, as well as a source code archive if you want to yourself.


Packages and Installers


Select your operating system family:

Linux


macOS


Windows


BSD


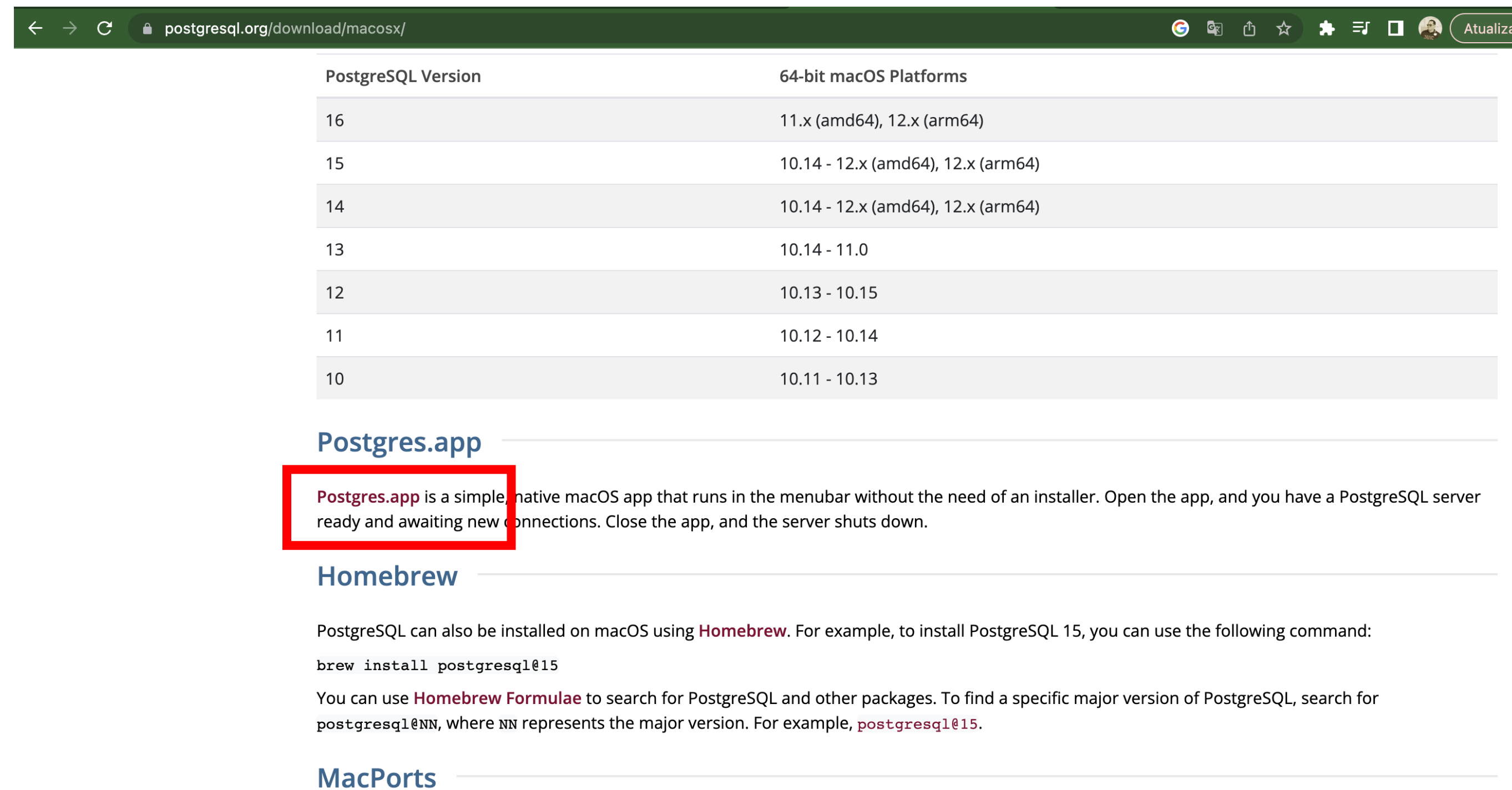
Solaris


Source code

INSTALAÇÃO POSTGRES

Para baixar o app basta clicar em “postgres.app”: <https://postgresapp.com/>

- postgres.app: é um app do macos que não precisa de instalação. com ele aberto é possível ter um servidor postgresql pronto para novas conexões.



The screenshot shows the PostgreSQL download page for macOS. It features a table of PostgreSQL versions and their supported macOS platforms. Below the table, there is a section for Postgres.app, which is highlighted with a red box. The text in the red box states: "Postgres.app is a simple native macOS app that runs in the menubar without the need of an installer. Open the app, and you have a PostgreSQL server ready and awaiting new connections. Close the app, and the server shuts down."

PostgreSQL Version	64-bit macOS Platforms
16	11.x (amd64), 12.x (arm64)
15	10.14 - 12.x (amd64), 12.x (arm64)
14	10.14 - 12.x (amd64), 12.x (arm64)
13	10.14 - 11.0
12	10.13 - 10.15
11	10.12 - 10.14
10	10.11 - 10.13

Postgres.app

Postgres.app is a simple native macOS app that runs in the menubar without the need of an installer. Open the app, and you have a PostgreSQL server ready and awaiting new connections. Close the app, and the server shuts down.

Homebrew

PostgreSQL can also be installed on macOS using Homebrew. For example, to install PostgreSQL 15, you can use the following command:

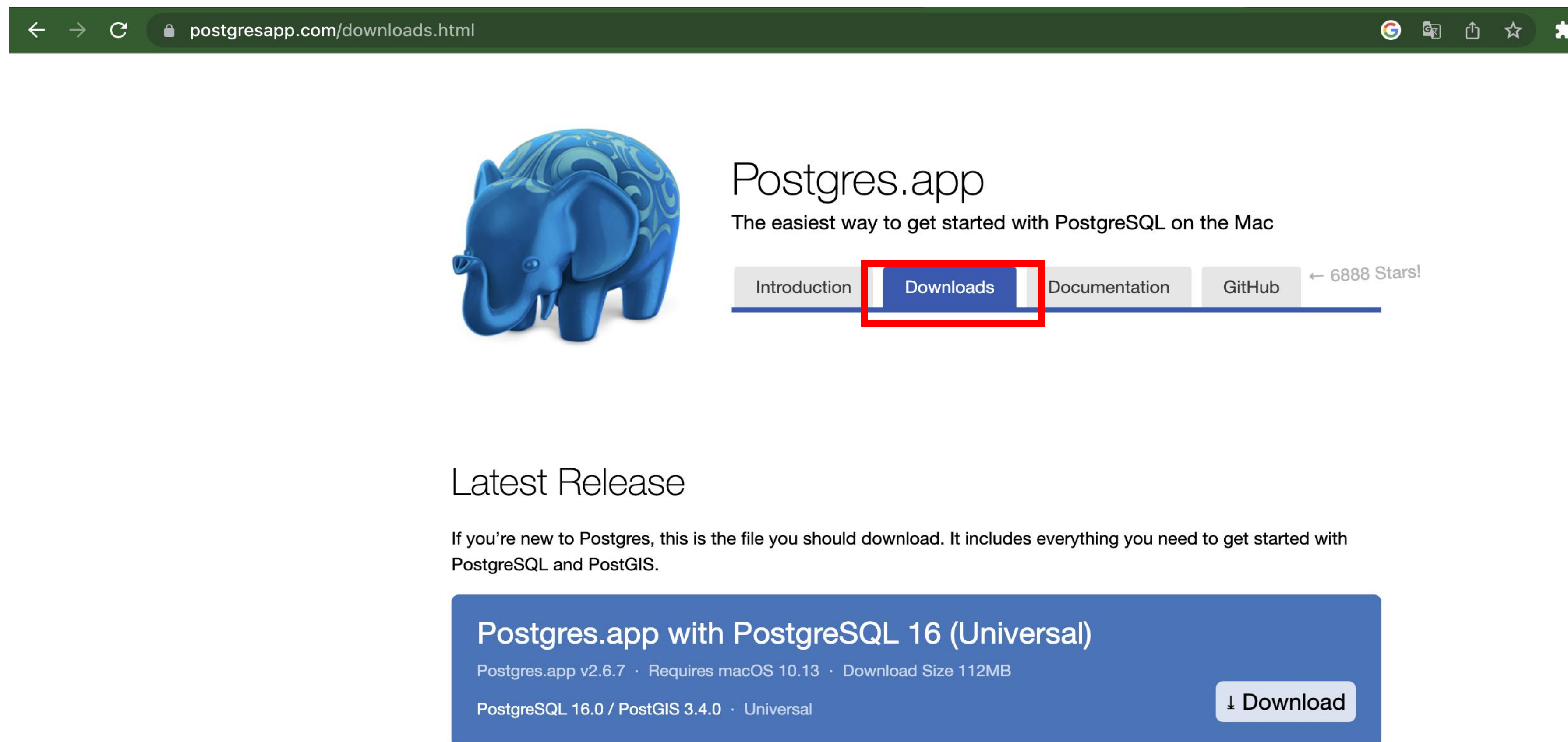
```
brew install postgresql@15
```

You can use Homebrew Formulae to search for PostgreSQL and other packages. To find a specific major version of PostgreSQL, search for postgresql@NN, where NN represents the major version. For example, postgresql@15.


MacPorts

INSTALAÇÃO POSTGRES

Na aba downloads clique no botão “download” da opção mais recente (latest release)



← → ↻ postgresapp.com/downloads.html



Postgres.app

The easiest way to get started with PostgreSQL on the Mac

Introduction Downloads Documentation GitHub ← 6888 Stars!

Latest Release

If you're new to Postgres, this is the file you should download. It includes everything you need to get started with PostgreSQL and PostGIS.

Postgres.app with PostgreSQL 16 (Universal)

Postgres.app v2.6.7 · Requires macOS 10.13 · Download Size 112MB

PostgreSQL 16.0 / PostGIS 3.4.0 · Universal

↓ Download

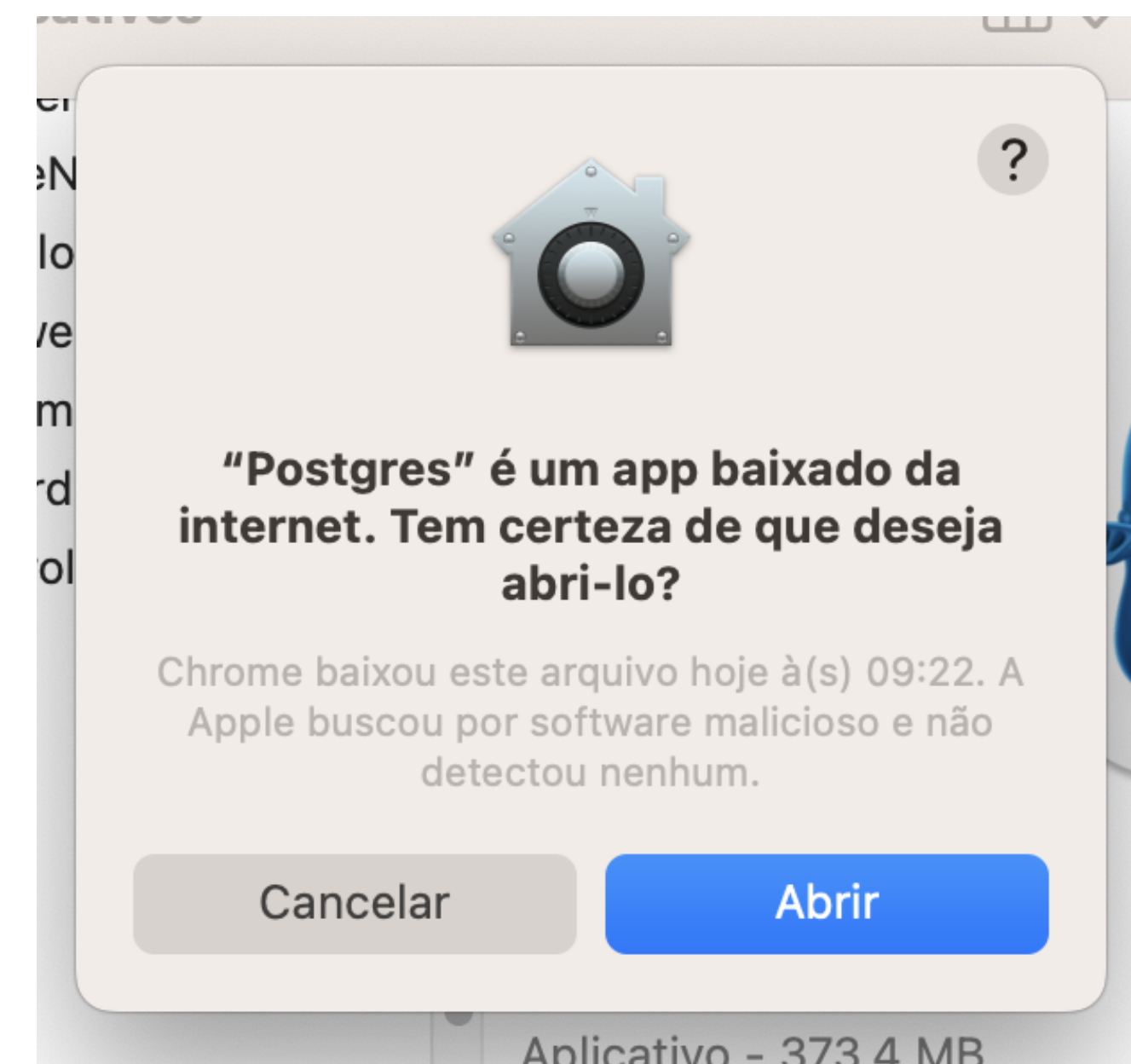
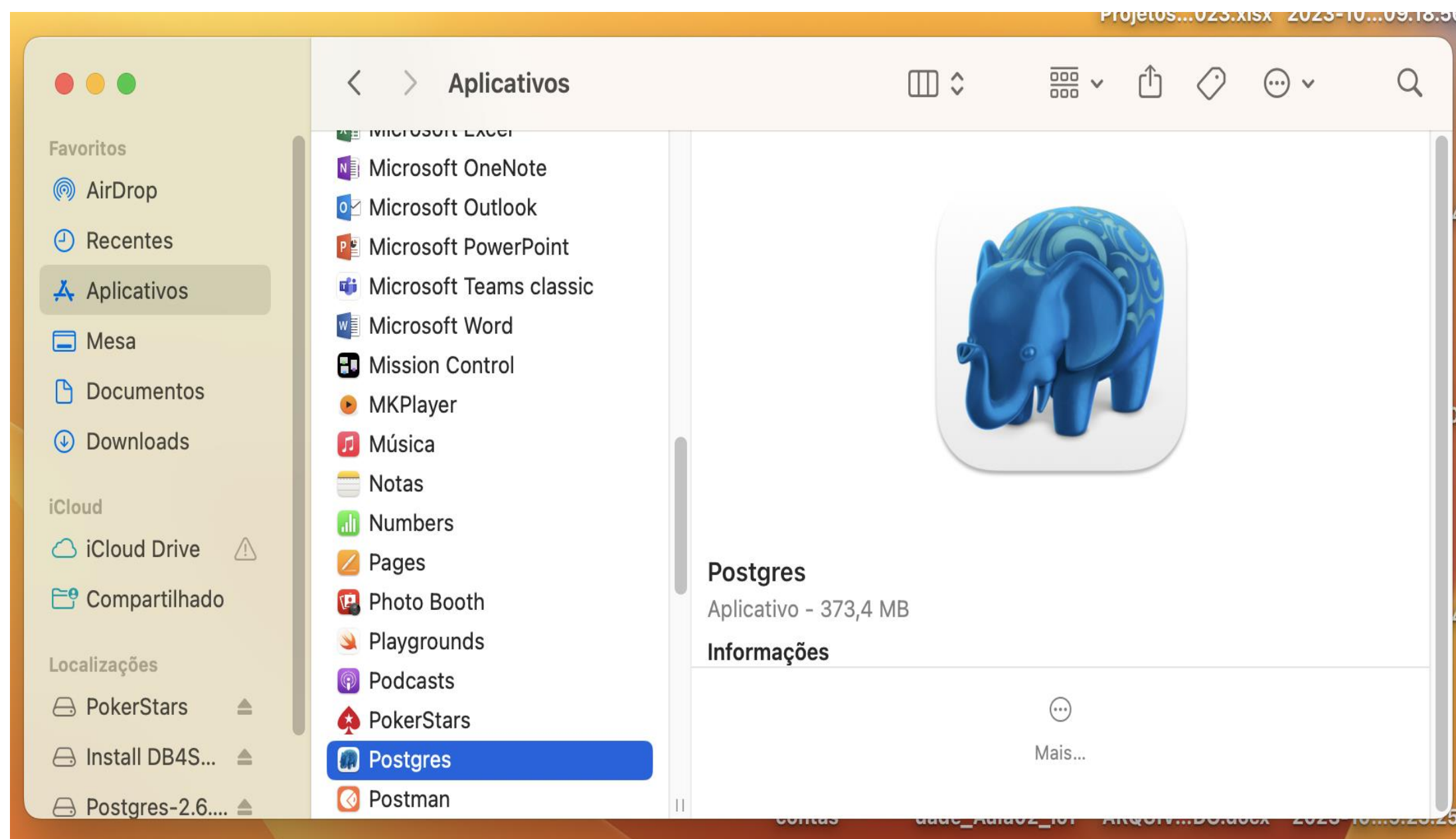
INSTALAÇÃO POSTGRES

- Após o download “arraste o ícone do postgres” para a pasta “aplicativos”. obs.: caso você não arraste o ícone para a pasta aplicativos pode ser que alguns comandos não funcionem.



INSTALAÇÃO POSTGRES

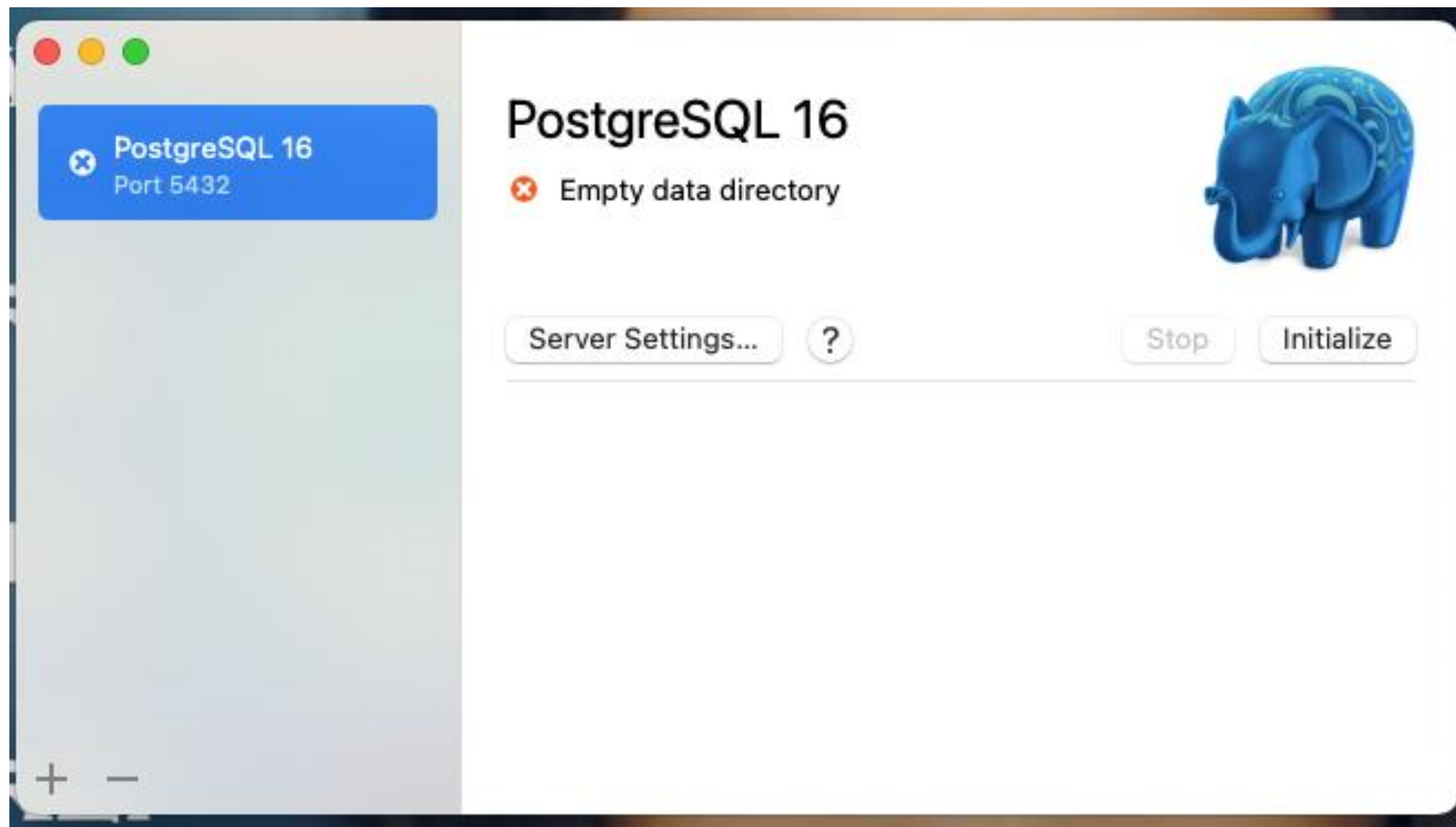
- Abra a pasta aplicativos e de um clique duplo no ícone postgres



- Clique em abrir

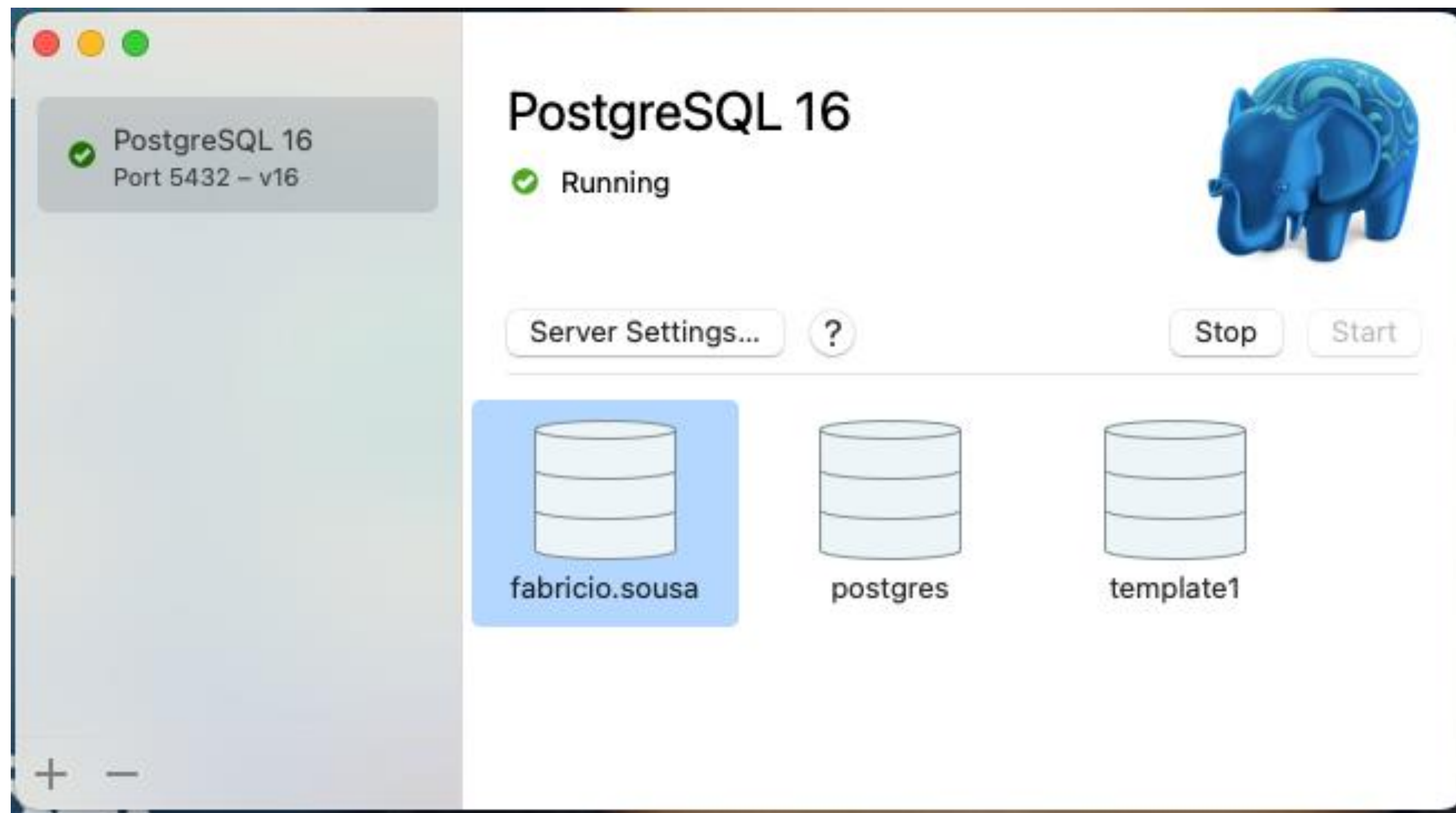
INSTALAÇÃO POSTGRES

- Clique em “initialize” para iniciar o servidor.



INSTALAÇÃO POSTGRES

- o servidor vai exibir as base de dados criadas por padrão. clique em uma das bases para ter acesso ao terminal.



CLIQUE EM "OK"



INSTALAÇÃO POSTGRES

- Para configurar o “path” no seu terminal digite as seguintes linhas de comando conforme a imagem abaixo:

A screenshot of a macOS terminal window. The title bar reads 'fabricio.sousa — psql -p5432 fabricio.sousa — 126x24'. The terminal shows the following text: 'Last login: Thu Oct 26 18:35:05 on ttys009', 'fabricio.sousa@MacBook-Pro-de-Fabrcio ~ % "/Applications/Postgres.app/Contents/Versions/16/bin/psql" -p5432 "fabricio.sousa"', 'psql (16.0)', 'Type "help" for help.', 'fabricio.sousa=# sudo mkdir -p /etc/paths.d &&', '[fabricio.sousa-# echo /Applications/Postgres.app/Contents/Versions/bin | sudo tee /etc/paths.d/postgresapp', and 'fabricio.sousa-#'.

```
fabricio.sousa — psql -p5432 fabricio.sousa — 126x24
Last login: Thu Oct 26 18:35:05 on ttys009
fabricio.sousa@MacBook-Pro-de-Fabrcio ~ % "/Applications/Postgres.app/Contents/Versions/16/bin/psql" -p5432 "fabricio.sousa"
psql (16.0)
Type "help" for help.

fabricio.sousa=# sudo mkdir -p /etc/paths.d &&
[fabricio.sousa-# echo /Applications/Postgres.app/Contents/Versions/bin | sudo tee /etc/paths.d/postgresapp
fabricio.sousa-#
```

```
sudo mkdir -p /etc/paths.d &&
```

```
echo /Applications/Postgres.app/Contents/Versions/latest/bin | sudo tee
/etc/paths.d/postgresapp
```

Banco de dados

O fantástico mundo do **Banco de Dados**

- Um banco de dados é um conjunto de tabelas que armazena informações.
- Seu princípio básico é armazenar informações de um sistema.



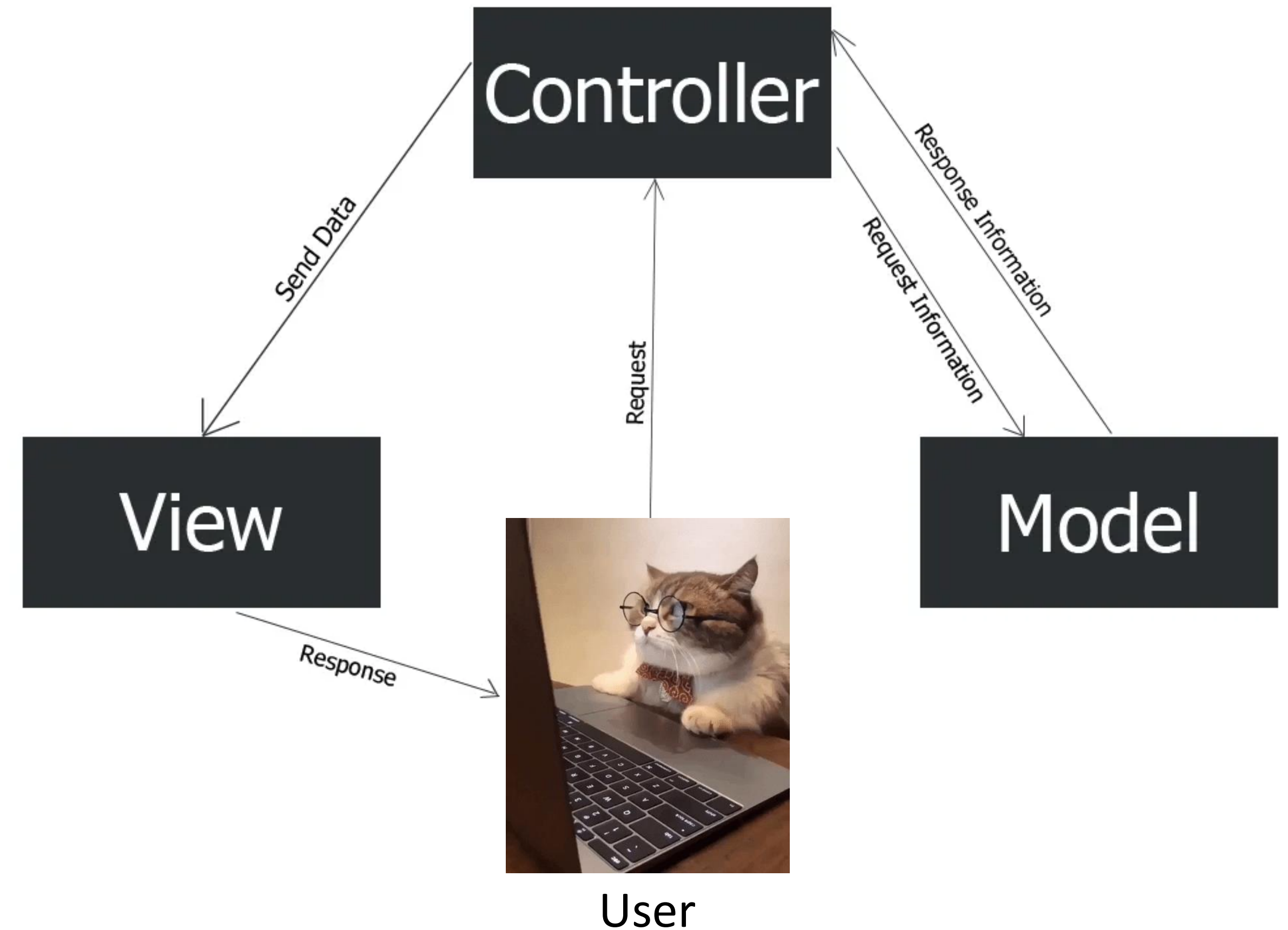
- Ao logar em um sistema, uma solicitação é feita ao banco de dados para verificar as credenciais.
- Se a resposta for positiva, então o usuário conseguirá acessar o sistema, caso não, ficará preso no limbo sem conseguir ir para lugar nenhum.

Banco de Dados

Arquitetura MVC

- Dividida em três camadas: Model, View e Controller.
- **View** lida com a interface do usuário.
- **Controller** processa regras de negócios.
- **Model**, classes simples(modelo) que representa objetos como pessoa, cliente, etc... Representadas por tabelas no banco de dados.

Model-View-Controller



Banco de Dados

Tipos de banco de dados

- **Banco de dados relacionais** (SQL é a linguagem usada para operar o banco de dado relacional — ou *Structured Query Language*, em inglês).
- **Banco de dados NoSQL** é um banco de dados não-relacional, que permite que os dados não estruturados e semiestruturados (mencionados anteriormente), possam ser armazenados e manipulados, usando valores-chave dinâmicos (em contraste com um banco de dados relacional, que usa linhas e colunas fixas).
- **Banco de dados orientados a objetos** (BDOOs): sua representação se dá na forma de objetos.
- **Data warehouse:** é um repositório de dados.
- **Bancos de dados OLTP** (ou em inglês, *Online Transactional Processing*): um banco de dados analítico projetado para um grande número de transações realizados por vários usuários ao mesmo tempo.

Banco de Dados

Relacional (SQL) e Não Relacional (NOSQL)

- Para ilustrar, pense em duas cidades:
- **Cidade A** - onde todos falam a mesma língua. Todas as empresas são criadas em torno dela, toda forma de comunicação a utiliza essa língua. Mudar essa língua em um lugar seria confuso para todos.
- **Cidade B** - onde cada casa pode falar uma língua diferente. Todos interagem com o mundo de forma diferente e não há um entendimento “universal” ou uma organização definida. Se uma casa é diferente, ela não afeta mais ninguém de forma alguma.



Banco de Dados

Relacional (SQL) e Não Relacional (NOSQL)

- As bases de dados SQL são estruturadas em **linguagem de consulta** (SQL) para definição e manipulação de dados. Sendo uma escolha segura e especialmente ótima para consultas complexas.
- Por outro lado, ele pode ser **restritivo**. O SQL exige que você use esquemas (arquiteturas visuais e lógicas de uma base de dados) pré-definidos para determinar a estrutura dos seus dados antes de trabalhar com eles.
- Além disso, todos os seus dados devem seguir a mesma estrutura.
- Isso pode requerer uma preparação inicial significativa e, assim como na Cidade A, pode ser que uma mudança na estrutura seja tanto difícil para todo o seu sistema.

Banco de Dados

Relacional (SQL) e Não Relacional (NoSQL)

- Uma base de dados NoSQL, por outro lado, tem um esquema **dinâmico** para dados não estruturados, e o dado é armazenado em várias formas: pode ser orientado a coluna, orientado a documento, baseado em grafos ou organizado como chave-valor. Essa flexibilidade permite que:
- Você crie documentos sem ter que definir sua estrutura primeiro;
- Cada documento tenha sua própria estrutura única;
- A sintaxe varie de base de dados para base de dados;
- Você adicione campos sempre que precisar.

Banco de Dados

Escalabilidade

- Na maioria das situações, as bases de dados SQL são **verticalmente escaláveis**, o que significa que você pode aumentar o carregamento em um servidor melhorando coisas como CPU, RAM ou SSD.
- As bases de dados NoSQL, por sua vez, são **horizontalmente escaláveis**.
- Isso quer dizer que você suporta muito mais tráfego por *sharding*(particionamento de dados), ou seja, adicionando mais servidores na sua base de dados NoSQL.

É como se comparássemos entre adicionar mais andares no mesmo prédio e adicionar mais prédios na vizinhança.



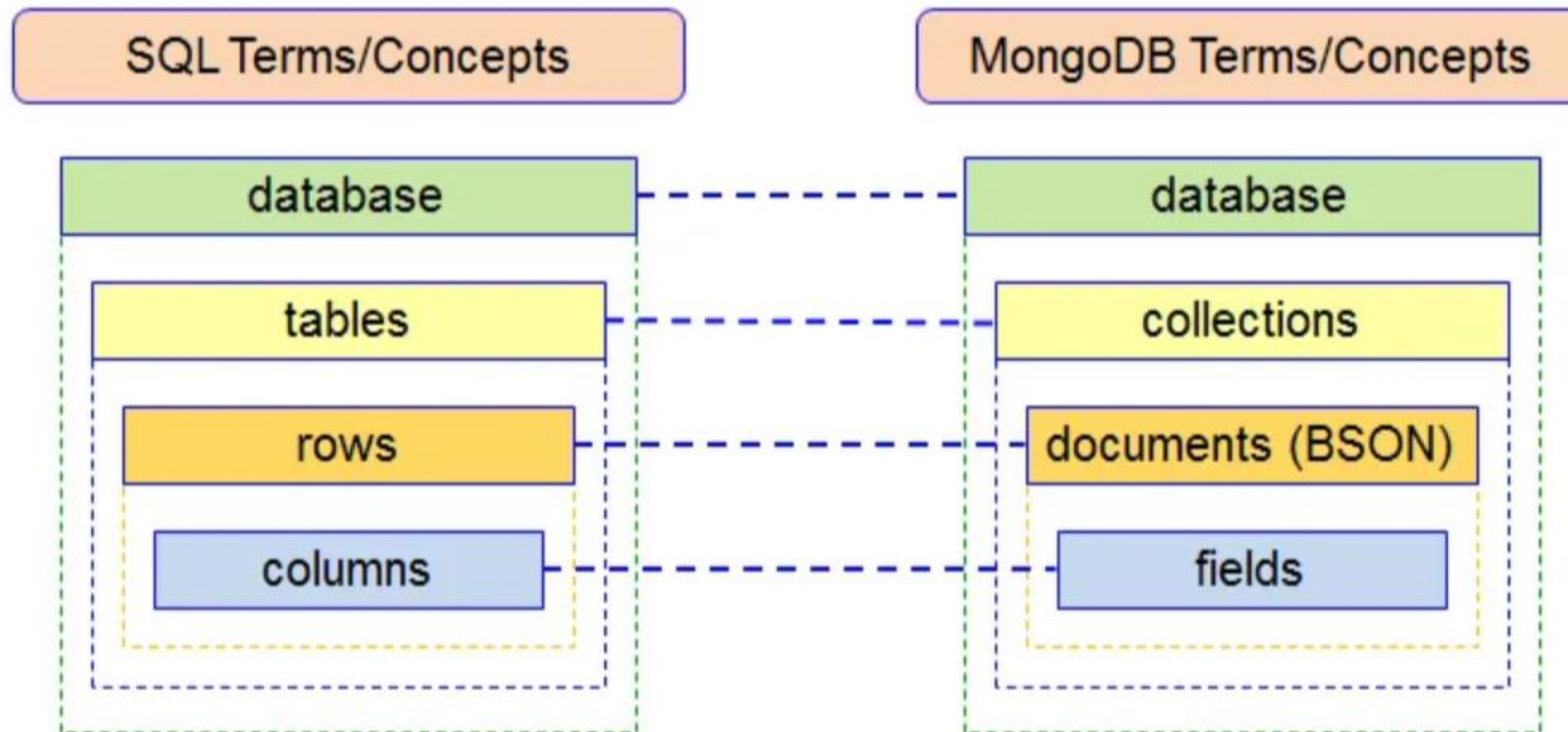
Banco de Dados

Estrutura

- As bases de **dados SQL** são baseadas em tabelas
- As bases de **dados NoSQL** podem ser baseadas em documentos, pares de chave-valor, grafos ou orientados a colunas.
- Isso torna as bases de dados SQL relacionais opções melhores para aplicações que requerem transações retornando várias colunas — como um sistema de contabilidade — ou para sistemas legados que foram criados em uma estrutura relacional.
- Exemplos de bases de **dados SQL** incluem o MySQL, Oracle, PostgreSQL e Microsoft SQL Server.
- Exemplos de bases de **dados NoSQL** incluem MongoDB, BigTable, Redis, RavenDB, Cassandra, HBase, Neo4j e CouchDB.

Banco de Dados

Comparativo de Objetos e Termos



Banco de Dados

MongoDB: O banco de dados NoSQL não-relacional



- **Esquema dinâmico.** Flexibilidade para mudar seu esquema de dados sem modificar nenhum outro dado existente.
- **Escalabilidade.** Horizontalmente escalável, o que ajuda a reduzir a carga de trabalho e escalar com facilidade.
- **Gerenciamento.** A base de dados não requer um administrador. Ele pode ser usado tanto pelos desenvolvedores quanto administradores.
- **Velocidade.** Tem performance alta para consultas simples.
- **Flexibilidade.** Você pode adicionar novas colunas e campos sem afetar as colunas existentes nem a performance da aplicação.

É uma boa escolha para negócios com crescimento rápido ou bases de dados sem definições claras de esquemas. Mais especificamente, se você não conseguir definir um esquema para o seu banco de dados, se perceber que está sempre "desnormalizando" esquemas de dados ou se o seu esquema passa constantemente por mudanças

Banco de Dados

MySQL: O banco de dados SQL relacional



- Ele é conhecido por sua velocidade e facilidade de uso. É frequentemente usado em aplicativos da web e é popular entre desenvolvedores.
- O MySQL suporta recursos como consultas SQL, chaves primárias e estrangeiras, procedimentos armazenados e visões.
- Existem duas variantes principais do MySQL: o MySQL Community Server (versão de código aberto gratuita) e o MySQL Enterprise Edition (que oferece recursos avançados e suporte comercial).
- É frequentemente usado em pequenos e médios aplicativos, como sites, blogs e sistemas de gerenciamento de conteúdo.

Banco de Dados

PostgreSQL : banco de dados SQL relacional



- O PostgreSQL, frequentemente chamado de "Postgres", é um sistema de gerenciamento de banco de dados de código aberto e de alta qualidade.
- O PostgreSQL suporta muitos recursos avançados, como consultas complexas, chaves estrangeiras, gatilhos, procedimentos armazenados, georreferenciamento (usando a extensão PostGIS), JSON nativo, entre outros.
- É altamente escalável e adequado para uma ampla variedade de aplicativos, desde pequenas aplicações até sistemas empresariais de grande porte.
- O PostgreSQL é de código aberto e está disponível gratuitamente. Ele é amplamente utilizado em aplicativos da web, aplicações empresariais e muito mais.

Modelagem em Banco Dados relacional

Banco de Dados

Decisão de Construir um Banco de Dados

- Surge da necessidade de armazenar dados específicos.
- Pode ocorrer quando nenhum software pronto atende às necessidades.
- Então daí surge a necessidade de **modelar um sistema** e criá-lo. E todo sistema, por mais simples que pareça, precisa de um banco de dados.
- **Modelar um banco de dados** de acordo com as informações desse sistema.

Banco de Dados

Requisitos

- Agora a parte **Importante**, não ache que vamos criar um banco de dados do nada por que não é bem assim. Começamos com algo chamado definição de **Requisitos**.
- **Documento de requisitos** é criado com as necessidades do sistema.
- Do que se trata o projeto? e qual finalidade? quem vai utilizar? quais são os requisitos técnicos, de software e hardware?



Ninguém define um banco de dados do nada!

Banco de Dados

Modelagem Conceitual

O que fazer?

- O objetivo aqui é criar um modelo de forma gráfica, sendo este chamado de **Diagrama Entidade e Relacionamento (DER)**, que identificará todas as entidades e relacionamentos de uma forma global.
- Aqui é evitado qualquer detalhamento específico do modelo de BD.
- Sua principal finalidade é capturar os requisitos de informação e regras de negócio sob o ponto de vista do negócio.
- Feito geralmente pelo Gestor de Dados de Negócio ou outro profissional acompanhado de sua supervisão/orientação.

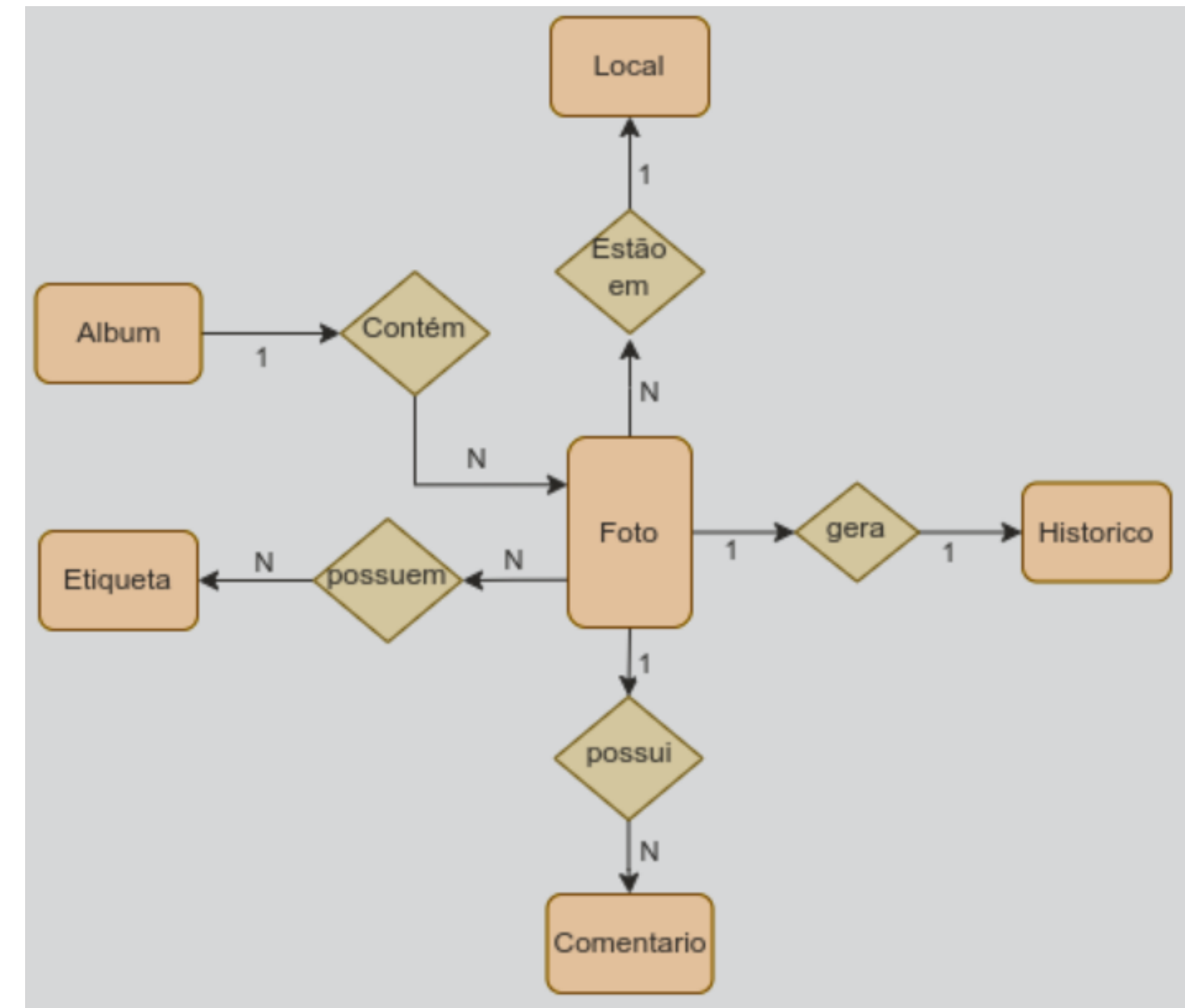
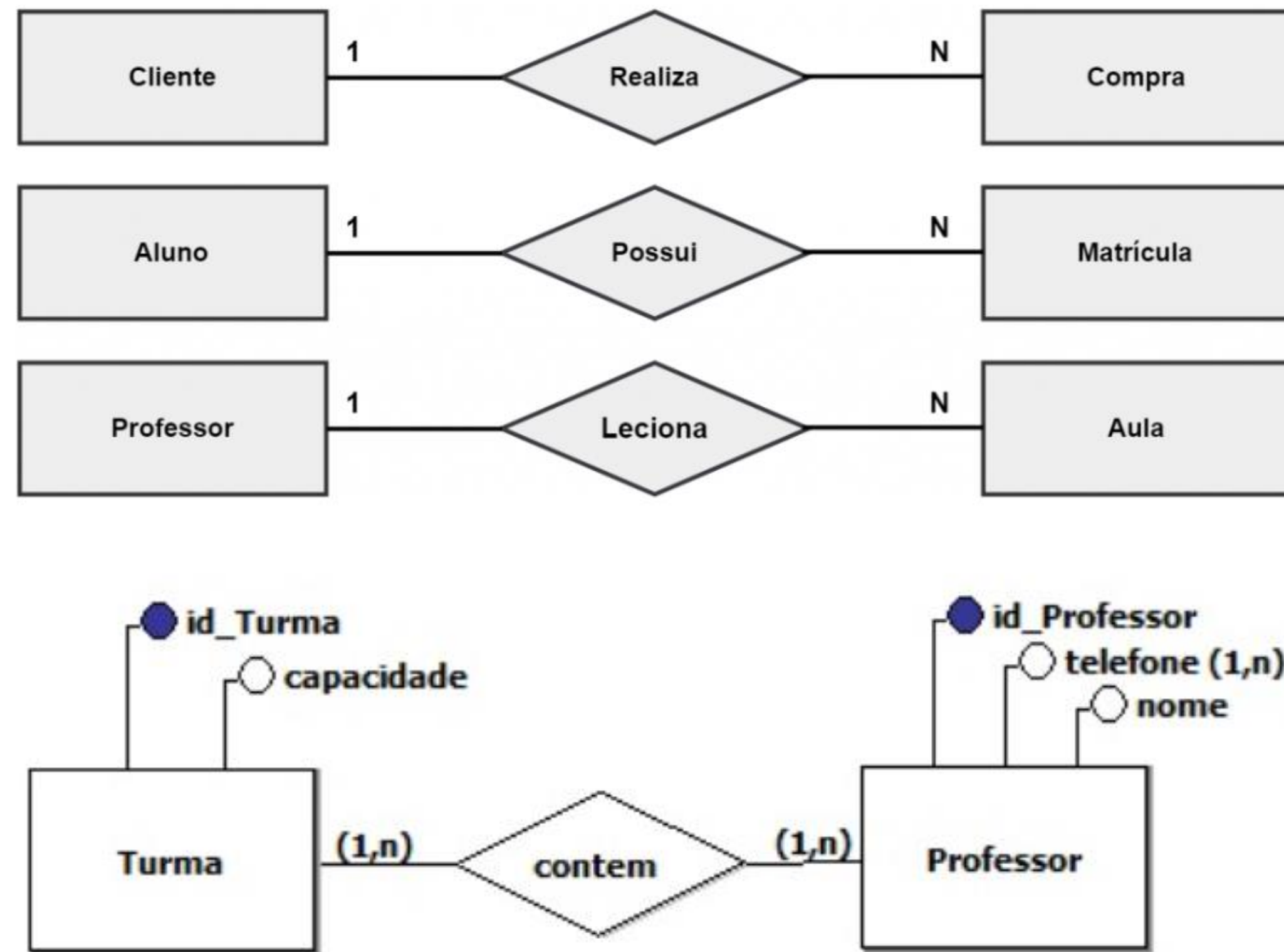
Banco de Dados

Exemplos de Modelagem Conceitual

DER – Diagrama de Entidade e Relacionamento



modelo conceitual



Banco de Dados

Modelagem Logica

Como fazer?

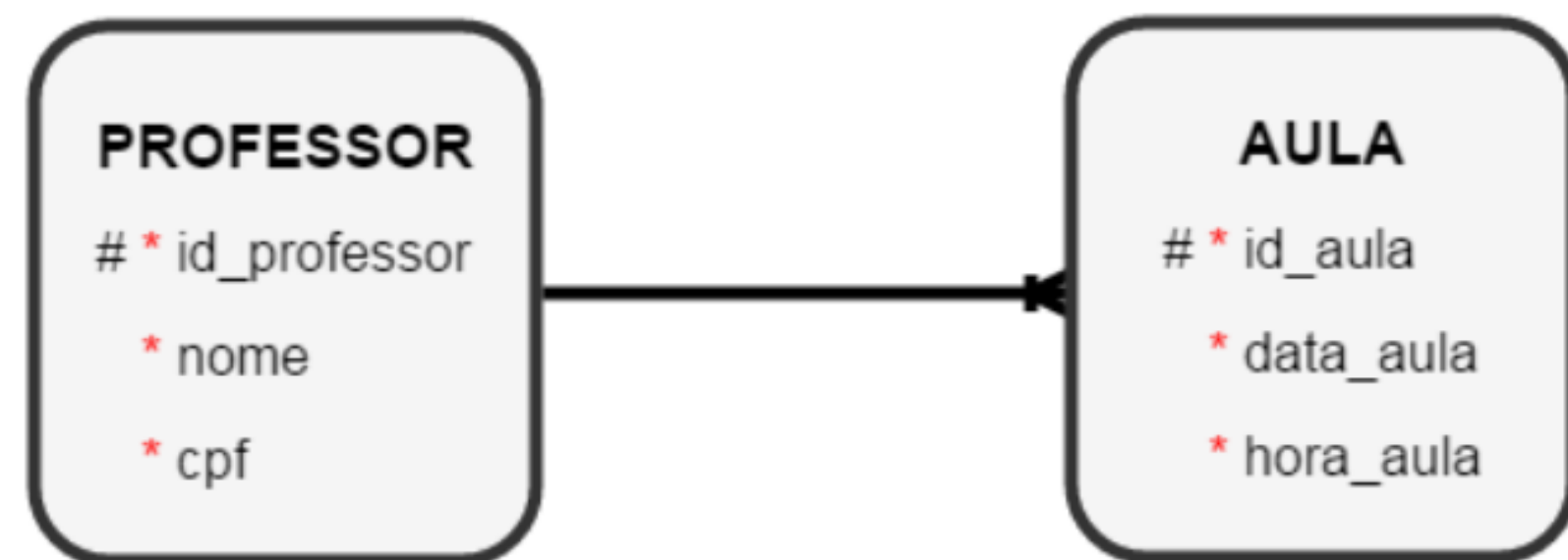
- Está principalmente associada à coleta de necessidades de negócios, e não ao design do banco de dados.
- Os modelos lógicos basicamente determinam se todos os requisitos do negócio foram reunidos.
- Ele é revisado pelos desenvolvedores, pelo gerenciamento e, por fim, pelos usuários finais para ver se é necessário coletar mais informações antes do início da modelagem física.
- O DER lógico também modela as informações coletadas dos requisitos de negócios.
- Observe que a configuração dos tipos de coluna é opcional e, se você fizer isso, deverá fazer isso para auxiliar na análise de negócios.
- Não tem nada a ver com a criação de banco de dados ainda.

Banco de Dados

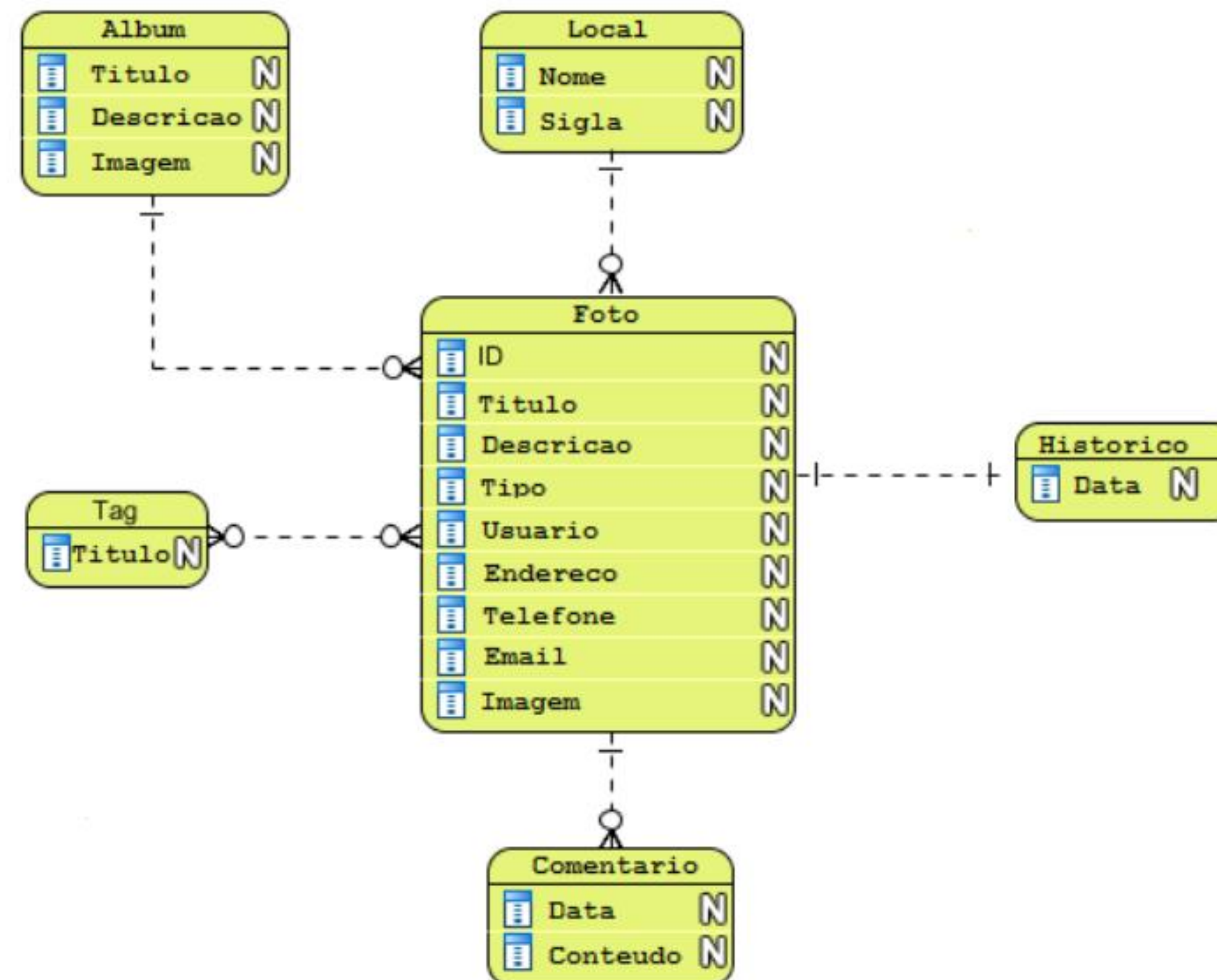
Exemplos de Modelagem Logica

DER – Diagrama de Entidade e Relacionamento

modelo lógico



DER - modelo lógico



Banco de Dados

Modelagem Física

- É o design do banco de dados real com base nos requisitos reunidos durante a modelagem lógica do banco de dados.
- Durante a modelagem física, os objetos são definidos em um nível denominado nível de esquema.
- Um esquema é considerado um grupo de objetos que estão relacionados entre si em um banco de dados.
- Tabelas e colunas são feitas de acordo com as informações fornecidas durante a modelagem lógica.
- Chaves primárias, chaves exclusivas e chaves estrangeiras são definidas para fornecer restrições. Índices são definidos.
- A modelagem física depende do software que já está sendo usado na organização.
- É específica ao software. [Sql Server, Oracle, MySql, Postgresql, etc]

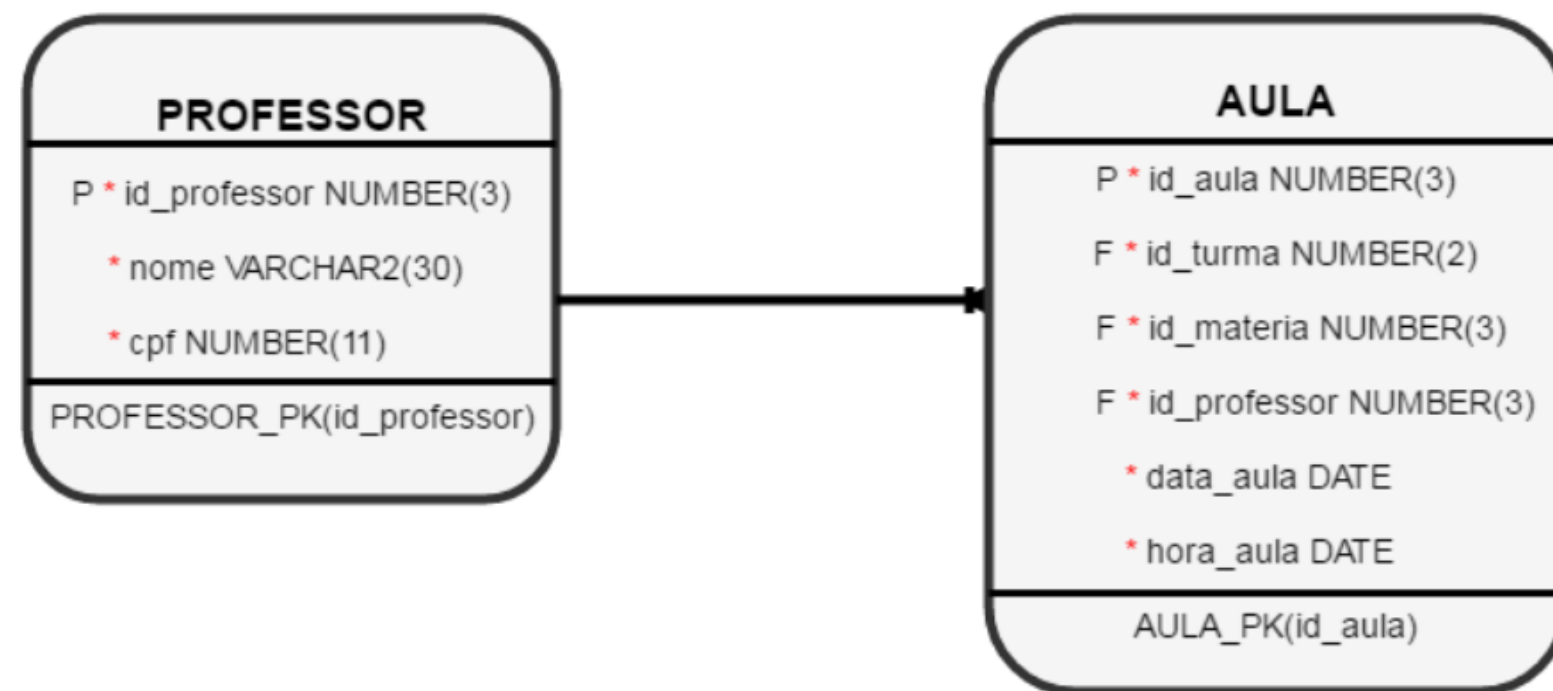


Banco de Dados

Exemplos de Modelagem Física

DER – Diagrama de Entidade e Relacionamento

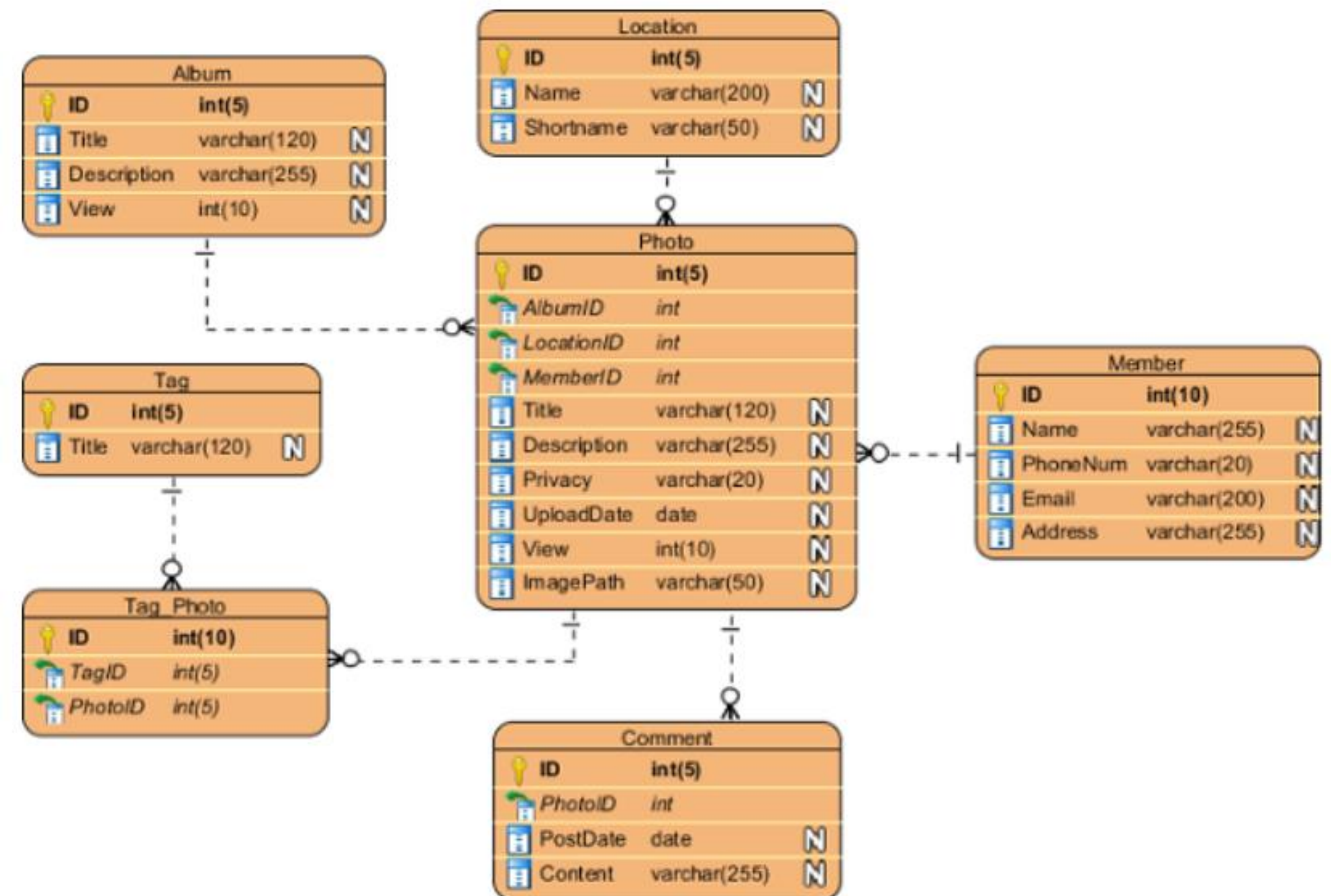
modelo físico



Um modelo físico pode ser constituído de código SQL para criação de objetos no banco

```
CREATE TABLE turma (  
  idturma INTEGER(4) NOT NULL AUTO_INCREMENT,  
  capacidade INTEGER(2) NOT NULL,  
  idProfessor INTEGER(4) NOT NULL,  
  PRIMARY KEY (idturma),  
  FOREIGN KEY (idProfessor) REFERENCES professor(idProfessor),  
  UNIQUE KEY idturma (idturma)  
)  
  
CREATE TABLE professor (  
  idProfessor INTEGER(4) NOT NULL AUTO_INCREMENT,  
  telefone INTEGER(10) NOT NULL,  
  nome CHAR(80) COLLATE NOT NULL DEFAULT '',  
  PRIMARY KEY (idProfessor),  
  FOREIGN KEY (idTurma) REFERENCES turma(idturma),  
  UNIQUE KEY idProfessor (idProfessor)  
)
```

DER - modelo físico



Banco de Dados

Tabela comparativa das características dos modelos

- O **modelo conceitual** é para conversar com o pessoal de negócios, que nada entendem de tecnologia.
- O **modelo lógico** tem dados mais técnicos, por exemplo o tipo do dado, o tamanho que ele comporta, e eventualmente alguma restrição de como o dado pode ser.
- No **modelo físico** colocará tudo o que é necessário para criar as tabelas no SGDB, não só das colunas, mas sobre [chaves](#), [índices](#), [gatilhos](#) e restrições mais técnicas, inclusive de permissões de acesso.

Característica	Conceitual	Lógico	Físico
Nome de Entidade	✓	✓	
Relacionamentos de Entidade	✓	✓	
Atributos	✓	✓	
Chave Primária		✓	✓
Chave Estrangeira		✓	✓
Nome das Tabelas			✓
Nome das Colunas			✓
Tipo das Colunas			✓



Chave Primária - PK
Chave Estrangeira - FK

Banco de Dados

CHAVE PRIMÁRIA - PK

- Corresponde a um identificador único da tabela em questão, pode ser representado por um campo (chave simples) ou mais (chave composta).
- As PKs não podem ser nulas, só deve existir uma na tabela.
- Se tivermos uma tabela de Pessoas, com os campos CPF, Nome, Idade e Nacionalidade, qual seria a PK?
- Existe uma outra abordagem que podemos usar com PKs, que é usar um ID como PK nessa tabela acima pois:- PKs podem ser incrementadas automaticamente pelo nosso BD.

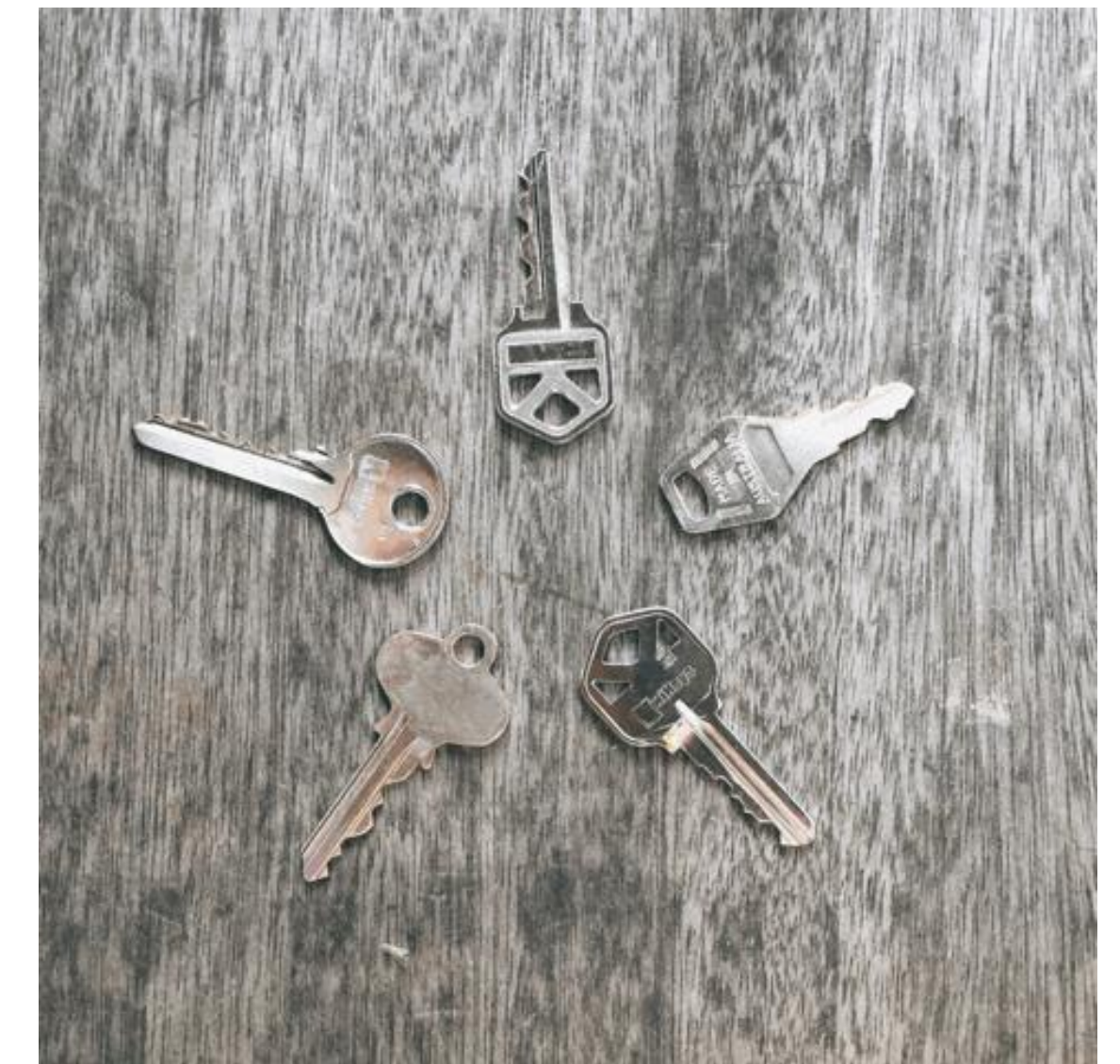
```
CREATE TABLE Pessoas (  
    CPF VARCHAR(14) PRIMARY KEY,  
    Nome VARCHAR(255) NOT NULL,  
    Idade INT,  
    Nacionalidade VARCHAR(50)  
);
```

```
CREATE TABLE Pessoas (  
    ID INT AUTO_INCREMENT PRIMARY KEY,  
    CPF VARCHAR(14),  
    Nome VARCHAR(255) NOT NULL,  
    Idade INT,  
    Nacionalidade VARCHAR(50)  
);
```

Banco de Dados

CHAVE PRIMÁRIA - PK

- Por exemplo, se tivéssemos uma tabela com informações de alunos em uma escola, a chave primária poderia ser o número de matrícula de cada aluno.
- Cada aluno tem um número de matrícula diferente, e é usado para garantir que cada linha da tabela represente um aluno único.
- Não pode haver dois alunos com o mesmo número de matrícula na mesma tabela.



Banco de Dados

CHAVE ESTRANGEIRA - FK

- Agora, pense em outra tabela em nosso banco de dados, talvez uma que armazene informações sobre os cursos que os alunos estão matriculados.
- Nessa tabela, podemos ter uma coluna que se relaciona com a tabela de alunos, usando o número de matrícula.
- Essa coluna, que se relaciona com a chave primária da tabela de alunos, é chamada de **chave estrangeira**.



Banco de Dados

CHAVE ESTRANGEIRA - FK

- Usamos para criar uma relação entre tabelas (uma ponte que conecta as informações em diferentes tabelas), resumidamente a FK é uma referência em uma tabela a uma chave primária de outra tabela.

Seguindo a ideia do slide anterior, vamos mudar a tabela Pessoas para essa:

```
CREATE TABLE Pessoas (  
  ID INT AUTO_INCREMENT PRIMARY KEY,  
  Nome VARCHAR(255) NOT NULL  
);
```

e vamos criar a tabela Enderecos dessa maneira:

```
CREATE TABLE Enderecos (  
  EnderecoID INT AUTO_INCREMENT PRIMARY KEY,  
  PessoaID INT,  
  Endereco VARCHAR(255),  
  FOREIGN KEY (PessoaID) REFERENCES Pessoas(ID)  
);
```

Banco de Dados

CHAVE ESTRANGEIRA - FK

- - A tabela "Enderecos" possui um campo "PessoalD" que será usado como a chave estrangeira;
- - FOREIGN KEY (PessoalD) REFERENCES Pessoas(ID): Isso define a chave estrangeira "PessoalD" na tabela "Enderecos" e a relaciona com a chave primária "ID" na tabela "Pessoas".
- Isso significa que o valor de "PessoalD" em cada registro na tabela "Enderecos" deve corresponder a um valor válido em "ID" na tabela "Pessoas".
- Agora, sempre que você inserir um novo registro na tabela "Enderecos", você deve garantir que o valor em "PessoalD" corresponda a um registro existente na tabela "Pessoas".
- Isso ajuda a manter a integridade referencial entre as duas tabelas e a estabelecer a relação entre pessoas e endereços.

Banco de Dados

Resumindo

- A chave primária é como a identidade única em uma tabela, enquanto a chave estrangeira é como um link que conecta informações em tabelas diferentes, permitindo-nos relacionar dados entre elas.

FIM



A large, stylized pink star with five points, positioned on the left side of the slide. The star is partially cut off by the left edge.

Agradecemos a sua atenção!