
# SISTEMA DE PORTARIA DE CONDOMÍNIO

**Matheus de Jesus Antunes**

Instituto Federal de Educação, Ciência e Tecnologia de São Paulo (IFSP)

Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas

Disciplina: Banco de Dados II

Professor: Paulo Giovani de Faria Zerefino

Data: Novembro 2024

## Resumo

Este projeto apresenta o desenvolvimento de um banco de dados relacional focado em um **Sistema de Portaria de Condomínio**, com o objetivo de melhorar o controle de acesso de pessoas e veículos. O sistema permite armazenar e consultar informações de forma rápida e organizada, otimizando a segurança e a administração do condomínio. Para isso, foram utilizadas ferramentas como o MySQL Workbench para a modelagem e o MySQL Server para a implementação. O trabalho reforça a importância de bases de dados bem estruturadas como uma solução eficiente para problemas do cotidiano.

---

## Introdução

Nos últimos anos, a gestão de acessos em condomínios residenciais tem se tornado um desafio crescente. A complexidade de controlar a entrada e saída de moradores, visitantes e prestadores de serviço exige soluções mais eficientes e seguras. Embora métodos tradicionais, como registros manuais, ainda sejam amplamente utilizados, eles apresentam limitações significativas, como erros humanos, dificuldades de organização e falta de agilidade na geração de relatórios. Para atender a essas necessidades, sistemas automatizados de controle de acesso têm se mostrado cada vez mais essenciais, especialmente quando aliados à tecnologia.

Este projeto propõe o desenvolvimento de um **Sistema de Portaria de Condomínio** baseado em um banco de dados relacional. O objetivo central é criar uma solução que facilite o controle de entrada e saída de pessoas e veículos, garantindo maior segurança e agilidade no gerenciamento do condomínio. Por meio da automação dos processos, o sistema busca otimizar a administração do condomínio, oferecendo uma maneira prática de registrar informações, consultar dados e gerar relatórios importantes para os administradores.

---

## Objetivos

O objetivo principal deste projeto é desenvolver um banco de dados relacional para um sistema de portaria de condomínio, que permita a organização e a automação do controle de acessos. De forma mais específica, os objetivos incluem:

Estruturar o banco de dados com tabelas adequadas para armazenar informações sobre moradores, veículos, visitantes, prestadores de serviço e eventos de entrada/saída;

Garantir a integridade e a consistência dos dados, por meio de normalização e utilização de chaves primárias e estrangeiras;

Implementar funcionalidades para gerar consultas e relatórios que auxiliem na gestão administrativa e na segurança do condomínio;

Tornar o sistema escalável, de modo que ele possa ser facilmente adaptado a diferentes tamanhos de condomínios e necessidades administrativas.

## Justificativa

A escolha deste tema é motivada pela crescente preocupação com a segurança e eficiência na gestão de condomínios. O controle de acesso adequado não só aumenta a segurança, como também facilita a organização das operações cotidianas. Em um mundo cada vez mais digital, sistemas automatizados se destacam por sua capacidade de reduzir falhas humanas, acelerar processos e garantir que as informações sejam armazenadas e recuperadas de forma eficiente. Assim, o sistema desenvolvido neste projeto visa contribuir para uma gestão mais eficaz dos condomínios, oferecendo uma solução que combine praticidade, segurança e agilidade.

Além disso, a digitalização dos processos na portaria de condomínios permite que informações cruciais sejam acessadas rapidamente, tornando o trabalho dos administradores mais eficiente. O sistema não só facilita o controle de entrada e saída de pessoas e veículos, como também proporciona uma maneira organizada de lidar com os registros históricos, essencial para o acompanhamento de visitas e eventos.

---

## Aspectos Metodológicos

A metodologia adotada para o desenvolvimento deste trabalho seguiu uma abordagem estruturada, dividida em várias etapas: análise, modelagem e implementação. Inicialmente, foi feita uma análise das necessidades do sistema, com base em regras de negócios levantadas por meio de entrevistas com administradores de condomínio. A partir disso, foram definidas as entidades do sistema e seus respectivos relacionamentos.

Para a modelagem do banco de dados, foi utilizada a ferramenta MySQL Workbench, que permitiu o desenho de um diagrama Entidade-Relacionamento (ER) e a definição do modelo lógico, estruturado em tabelas. A implementação foi realizada no MySQL Server, onde o banco de dados foi criado, testado e validado. Durante o processo de desenvolvimento, foram realizados testes com dados fictícios para verificar o funcionamento das tabelas e a execução das consultas.

---

## Aporte Teórico

O embasamento teórico para este projeto se baseia em conceitos consolidados na área de banco de dados, principalmente no que se refere à modelagem relacional, normalização, integridade referencial e consultas SQL. Para a elaboração do banco de dados, foi necessário aplicar as boas práticas de modelagem de dados, garantindo a

consistência e eficiência no armazenamento das informações. Além disso, o uso de Sistemas de Gerenciamento de Banco de Dados (SGBD) como o MySQL também é amplamente documentado na literatura, sendo uma das ferramentas mais recomendadas para projetos dessa natureza.

Este trabalho também se apoia em conceitos de segurança de dados e automação de processos, abordando como o uso de tecnologia pode otimizar a gestão de acesso em condomínios, contribuindo para um ambiente mais seguro e organizado.

---

## Metodologia

Este projeto tem como objetivo criar um banco de dados relacional para um Sistema de Portaria de Condomínio, com a finalidade de facilitar e otimizar o controle de acesso de pessoas e veículos no condomínio. Em um cenário onde o controle manual é uma constante fonte de erros e dificuldades, a automação se torna uma solução fundamental para aumentar a eficiência e a segurança do sistema.

---

## Ferramentas Utilizadas

Ferramenta de Modelagem: Para a modelagem do banco de dados, escolhemos o MySQL Workbench, que é uma ferramenta bastante conhecida para o design e implementação de bancos de dados. Com ela, conseguimos desenhar o modelo de forma visual, tornando mais fácil entender como as entidades se relacionam entre si. Para representar essas relações, usamos a notação Diagrama Entidade-Relacionamento (DER). Essa notação é muito eficiente para ilustrar como as entidades, como Morador, Veículo, Visitante e Entrada/Saída, se conectam no sistema. O uso do DER nos ajudou a visualizar claramente as interações e a estrutura do banco de dados.

Ferramenta de Implementação: Após a modelagem, a implementação foi feita no MySQL Server, um dos sistemas de gerenciamento de banco de dados mais usados. Ele foi escolhido por sua robustez e desempenho. A instalação foi rápida, utilizando o MySQL Installer, o que nos permitiu começar a criar e testar o banco de dados de maneira ágil.

---

## Modelo Conceitual - DER

O Modelo Conceitual apresentado foi projetado para representar as entidades e os relacionamentos que capturam as necessidades do sistema de portaria do condomínio. Este modelo de dados é fundamental para entender como as informações devem ser organizadas e relacionadas dentro do banco de dados. O diagrama de relacionamento de entidades (DER) contém as seguintes entidades principais:

1. **Morador**

Cada morador pode ter múltiplos veículos registrados, mas cada veículo pertence a um único morador. Isso implica um relacionamento de **1:N** entre **Morador** e **Veículo**.

2. **Veículo**

O veículo é vinculado a um único morador, formando um relacionamento de **N:1** entre **Veículo** e **Morador**.

3. **Visitante**

Os visitantes são registrados no sistema para monitoramento de suas visitas. Cada visitante pode ter múltiplas entradas e saídas registradas. Isso configura um relacionamento de **1:N** entre **Visitante** e **Entrada/Saída**.

4. **Entrada/Saída**

Registra a entrada e saída dos visitantes. Cada entrada/saída está associada a um único visitante, formando um relacionamento de **N:1** entre **Entrada/Saída** e **Visitante**.

---

## Regras de Negócio

As **regras de negócio** ajudam a garantir que o banco de dados seja funcional e atenda às necessidades do sistema. As principais regras de negócio para o sistema de portaria são:

1. **Moradores podem ter múltiplos veículos registrados.**

Um morador pode registrar vários veículos, mas cada veículo é vinculado a apenas um morador.

2. **Visitantes devem ser autorizados por um morador para acessar o condomínio.**

Cada visitante precisa de autorização de um morador para entrar no condomínio. Essa autorização é registrada na entrada/saída do visitante.

3. **Prestadores de serviço devem ser registrados para monitoramento de suas visitas.**

Os prestadores de serviço (como profissionais de limpeza ou manutenção) devem ser registrados, assim como os visitantes, para garantir o controle sobre suas entradas e saídas.

4. **Todas as entradas e saídas de visitantes precisam ser registradas de forma detalhada.**

O sistema deve registrar detalhadamente as entradas e saídas de cada visitante, incluindo data, hora, morador responsável pela autorização e outros dados relevantes.

## Dicionário de Dados

O **dicionário de dados** fornece uma descrição detalhada de cada tabela e de seus respectivos campos. Abaixo está o dicionário de dados para as tabelas mencionadas no modelo:

**1. Tabela: Morador**

| Campo | Tipo de Dados | Descrição |
|---|---|---|
| **id_morador** | INT | Identificador único do morador. |
| **nome** | VARCHAR(100) | Nome completo do morador. |
| **CPF** | VARCHAR(14) | CPF do morador, utilizado para identificação. |
| **email** | VARCHAR(100) | Endereço de e-mail do morador. |
| **telefone** | VARCHAR(15) | Número de telefone do morador. |

**2. Tabela: Veículo**

| Campo | Tipo de Dados | Descrição |
|---|---|---|
| **id_veiculo** | INT | Identificador único do veículo. |

| | | |
|---|---|---|
| **placa** | VARCHAR(7) | Placa do veículo. |
| **modelo** | VARCHAR(50) | Modelo do veículo. |
| **cor** | VARCHAR(20) | Cor do veículo. |
| **id_morador** | INT | Chave estrangeira que se refere ao morador dono do veículo. |

**3. Tabela: Visitante**

| Campo | Tipo de Dados | Descrição |
|---|---|---|
| **id_visitante** | INT | Identificador único do visitante. |
| **nome** | VARCHAR(100) | Nome completo do visitante. |
| **documento** | VARCHAR(20) | Documento de identificação (RG ou CPF). |
| **telefone** | VARCHAR(15) | Número de telefone do visitante. |
| **id_morador** | INT | Chave estrangeira que referencia o morador que autorizou a entrada. |

**4. Tabela: Entrada/Saída**

| Campo | Tipo de Dados | Descrição |
|---|---|---|
| **id_entrada_saida** | INT | Identificador único do movimento de entrada/saída. |
| **id_visitante** | INT | Chave estrangeira que referencia o visitante. |
| **data** | DATETIME | Data e hora do movimento (entrada ou saída). |
| **tipo** | VARCHAR(10) | Tipo de movimento: 'Entrada' ou 'Saída'. |

**5. Tabela: Prestador de Serviço**

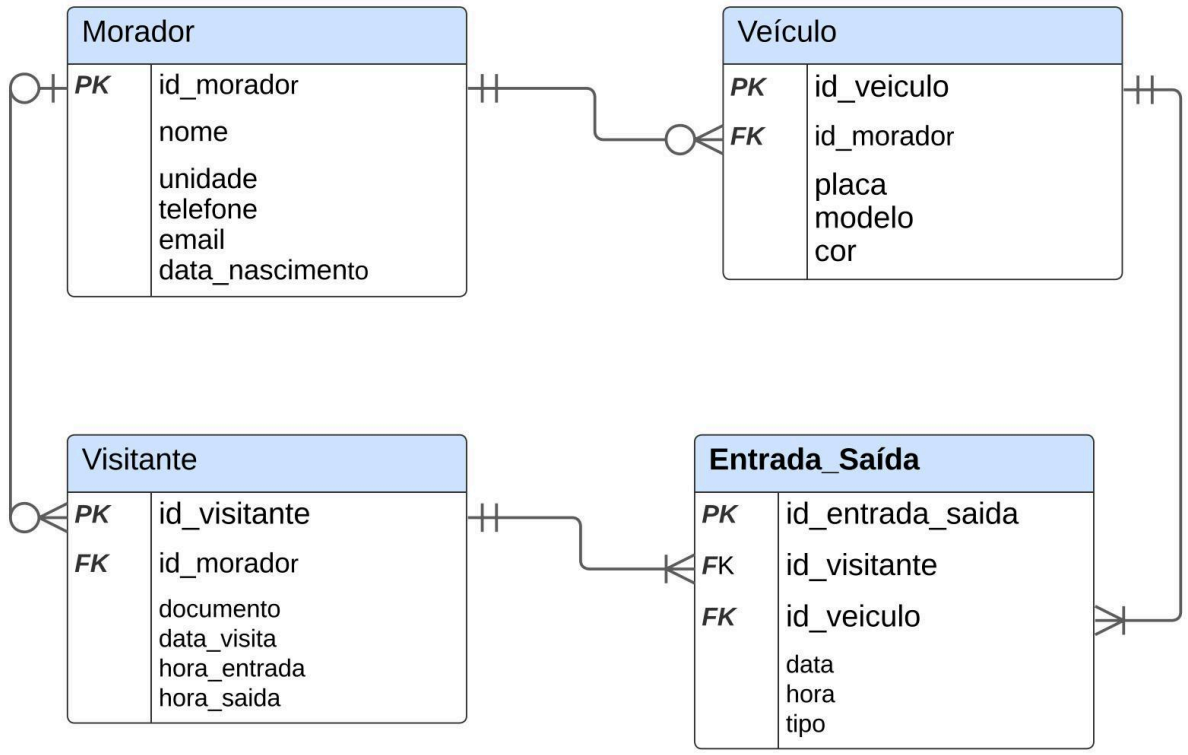
| Campo | Tipo de Dados | Descrição |
|---|---|---|
| **id_prestador** | INT | Identificador único do prestador de serviço. |
| **nome** | VARCHAR(100) | Nome do prestador de serviço. |
| **tipo_servico** | VARCHAR(50) | Tipo de serviço realizado pelo prestador. |
| **telefone** | VARCHAR(15) | Número de telefone do prestador. |

## Modelo Físico - Implementação e Normalização

Durante a implementação, seguimos boas práticas de normalização para evitar redundâncias e garantir que os dados fossem organizados de forma eficiente. As chaves primárias e estrangeiras foram definidas corretamente para garantir a integridade referencial entre as tabelas.

O fluxograma abaixo demonstra o fluxo de informações no sistema, destacando os principais processos e como eles se conectam às tabelas do banco de dados.

**Fluxograma do Sistema de Portaria**




O fluxograma complementa a visão do modelo físico apresentado, evidenciando o relacionamento lógico entre as tabelas e os processos implementados, como o registro de visitantes e a consulta de relatórios.

Segue abaixo o script SQL que define as tabelas do banco de dados...

1. Tabela **`Morador`** A tabela Morador armazena informações sobre os moradores do condomínio, incluindo dados pessoais e de contato.

```sql
-- Tabela para armazenar informações de moradores

CREATE TABLE Morador (

    id_morador INT PRIMARY KEY AUTO_INCREMENT,

    nome VARCHAR(100),

    unidade VARCHAR(10),

    telefone VARCHAR(15),

    email VARCHAR(100),

   data_nascimento DATE );
```

**2.** Tabela **`Veiculo`** A tabela Veiculo armazena informações sobre os veículos dos moradores. Cada veículo está vinculado a um morador através da chave estrangeira id_morador.

```sql
// Tabela para armazenar veículos associados a moradores

CREATE TABLE Veiculo (

   id_veiculo INT PRIMARY KEY AUTO_INCREMENT,

   placa VARCHAR(7),

   modelo VARCHAR(50),

   cor VARCHAR(30),

   id_morador INT,

FOREIGN KEY (id_morador) REFERENCES Morador(id_morador));
```

**3.** Tabela **Visitante** A tabela Visitante registra informações dos visitantes que acessam o condomínio, como nome, documento, data e horários de entrada e saída.

```sql
-- Tabela para registrar informações de visitantes

CREATE TABLE Visitante (

   id_visitante INT PRIMARY KEY AUTO_INCREMENT,

   nome VARCHAR(100),
```

```
    documento VARCHAR(20),
    data_visita DATE,
    hora_entrada TIME,
    hora_saida TIME
);
```

4. Tabela **Entrada_Saida** A tabela Entrada_Saida é responsável por registrar as entradas e saídas dos visitantes no condomínio. O tipo de movimento (Entrada ou Saída) é armazenado na coluna tipo e a tabela está relacionada à tabela Visitante.

```
-- Tabela para registrar entradas e saídas no condomínio
CREATE TABLE Entrada_Saida (
    id_entrada_saida INT PRIMARY KEY AUTO_INCREMENT,
    data DATE,
    hora TIME,
    tipo ENUM('Entrada', 'Saída'),
    id_visitante INT,
FOREIGN KEY (id_visitante) REFERENCES Visitante(id_visitante)
);
```

**Resultados Obtidos**

Com essa metodologia, foi possível criar um banco de dados robusto e eficiente para o sistema de portaria do condomínio. A modelagem, com a notação DER, ajudou a visualizar claramente os relacionamentos entre as entidades, e a implementação no MySQL Server garantiu que o sistema fosse funcional e fácil de administrar. O uso da normalização e das chaves primárias e estrangeiras assegurou que o banco de dados fosse eficiente, sem redundâncias e com integridade referencial. Esse sistema automatizado oferece uma solução prática e segura para os administradores de condomínio, melhorando o controle de acesso e garantindo a segurança do local.

**Consultas SQL**

Aqui estão 30 consultas SQL que podem ser realizadas com o banco de dados desenvolvido, com a descrição de cada uma e o código SQL correspondente:

1.**Relatório de Visitantes no Dia**
**Descrição:** Exibe todos os visitantes registrados no dia atual, com seus nomes, data da visita, hora de entrada e hora de saída.
**Código SQL:**

```sql
SELECT nome, data_visita, hora_entrada, hora_saida

FROM Visitante

WHERE data_visita = CURDATE();
```

**Resultado Esperado:**

Exibe os nomes dos visitantes, a data da visita, a hora de entrada e a hora de saída. Por exemplo: João Silva, 02/12/2024, 08:30:00, 10:00:00.

---

2.**Veículos Associados a Moradores**
**Descrição:** Exibe os veículos associados aos moradores, com o nome do morador, a placa e o modelo do veículo.
**Código SQL:**

```sql
SELECT Morador.nome, Veiculo.placa, Veiculo.modelo

FROM Morador

JOIN Veiculo ON Morador.id_morador = Veiculo.id_morador;
```

**Resultado Esperado:**

Exibe o nome do morador, a placa do veículo e o modelo do veículo. Por exemplo: João Silva, ABC123, Fusca.

---

3.**Visitantes que Entraram Após as 18h**
**Descrição:** Exibe os visitantes que entraram no condomínio após as 18:00, com o nome e a hora de entrada.

**Código SQL:**

```sql
SELECT nome, hora_entrada

FROM Visitante

WHERE hora_entrada > '18:00:00';
```

**Resultado Esperado:**

Exibe o nome do visitante e a hora de entrada. Por exemplo: João Silva, 18:30:00.

---

4.**Moradores com Veículos de Cor Vermelha**
**Descrição:** Exibe os moradores que possuem veículos vermelhos, com o nome do morador, modelo e cor do veículo.
**Código SQL:**

```sql
SELECT Morador.nome, Veiculo.modelo, Veiculo.cor

FROM Morador

JOIN Veiculo ON Morador.id_morador = Veiculo.id_morador

WHERE Veiculo.cor = 'Vermelha';
```

**Resultado Esperado:**

Exibe o nome do morador, o modelo e a cor do veículo. Por exemplo: João Silva, Fusca, Vermelha.

---

5.**Visitantes que Saíram Antes das 10h**
**Descrição:** Exibe os visitantes que saíram antes das 10:00, com o nome e a hora de saída.

**Código SQL:**

```sql
SELECT nome, hora_saida

FROM Visitante

WHERE hora_saida < '10:00:00';
```

**Resultado Esperado:**

Exibe o nome do visitante e a hora de saída. Por exemplo: Maria Santos, 09:00:00.

---

6.**Número Total de Visitantes no Mês**
**Descrição:** Exibe o total de visitantes registrados no mês atual.
**Código SQL:**

```sql
SELECT COUNT(*) AS total_visitantes

FROM Visitante

WHERE MONTH(data_visita) = MONTH(CURDATE());
```

**Resultado Esperado:**

Exibe o total de visitantes no mês atual. Por exemplo: Total de visitantes: 50.

---

7.**Moradores com Veículos de Modelos Específicos**
**Descrição:** Exibe os moradores com veículos de modelo 'Fusca' ou 'Civic'.
**Código SQL:**

```sql
SELECT Morador.nome, Veiculo.modelo

FROM Morador

JOIN Veiculo ON Morador.id_morador = Veiculo.id_morador

WHERE Veiculo.modelo IN ('Fusca', 'Civic');
```

**Resultado Esperado:**

Exibe o nome do morador e o modelo do veículo. Por exemplo: João Silva, Fusca.

---

8.**Visitantes com Documentos Não Informados**
**Descrição:** Exibe os visitantes cujo documento não foi informado.
**Código SQL:**

```sql
SELECT nome

FROM Visitante

WHERE documento IS NULL;
```

**Resultado Esperado:**

Exibe o nome dos visitantes que não informaram o documento. Por exemplo: Maria Santos.

---

9. **Moradores com Veículos Cadastrados no Sistema**
   **Descrição:** Exibe os moradores que possuem veículos registrados.
   **Código SQL:**

```sql
SELECT Morador.nome, Veiculo.placa

FROM Morador

JOIN Veiculo ON Morador.id_morador = Veiculo.id_morador;
```

**Resultado Esperado:**

Exibe o nome do morador e a placa do veículo. Por exemplo: João Silva, ABC123.

---

10. **Visitantes que Ficaram Mais de 1 Hora no Condomínio**
    **Descrição:** Exibe os visitantes que permaneceram mais de uma hora no condomínio, com o nome e a duração da visita.
    **Código SQL:**

```sql
SELECT nome, TIMEDIFF(hora_saida, hora_entrada) AS duracao

FROM Visitante

WHERE TIMEDIFF(hora_saida, hora_entrada) > '01:00:00';
```

**Resultado Esperado:**

Exibe o nome do visitante e a duração da visita. Por exemplo: João Silva, 01:30:00.

---

11. **Total de Visitantes por Morador**
    **Descrição:** Exibe o total de visitantes por morador, mostrando o nome do morador e a quantidade de visitantes registrados.
    **Código SQL:**

```sql
SELECT Morador.nome, COUNT(*) AS total_visitantes

FROM Morador
```

```
JOIN Visitante ON Morador.id_morador = Visitante.id_morador
GROUP BY Morador.id_morador;
```

**Resultado Esperado:**

Exibe o nome do morador e o total de visitantes. Por exemplo: João Silva, 3.

---

12. **Visitantes que Entraram e Saíram no Mesmo Dia**
    **Descrição:** Exibe os visitantes que entraram e saíram no mesmo dia, com o nome, data da visita, hora de entrada e hora de saída.
    **Código SQL:**

```
SELECT nome, data_visita, hora_entrada, hora_saida
FROM Visitante
WHERE DATE(hora_entrada) = DATE(hora_saida);
```

**Resultado Esperado:**

Exibe os visitantes com a data da visita, hora de entrada e saída no mesmo dia. Por exemplo: João Silva, 02/12/2024, 08:30:00, 10:30:00.

---

13. **Visitantes Autorizados por Moradores Específicos**
    **Descrição:** Exibe os visitantes autorizados por um morador específico, com o nome do visitante e do morador.
    **Código SQL:**

```
SELECT Visitante.nome, Morador.nome
FROM Visitante
JOIN Morador ON Visitante.id_morador = Morador.id_morador
WHERE Morador.nome = 'João Silva';
```

**Resultado Esperado:**

Exibe os nomes dos visitantes autorizados por João Silva. Por exemplo: Maria Santos, João Silva.

14. **Visitas Com Hora de Entrada em um Intervalo Específico**
    **Descrição:** Exibe as visitas que entraram no condomínio entre 08:00:00 e 12:00:00, com o nome e hora de entrada.
    **Código SQL:**

```sql
SELECT nome, hora_entrada

FROM Visitante

WHERE hora_entrada BETWEEN '08:00:00' AND '12:00:00';
```

**Resultado Esperado:**

Exibe os nomes dos visitantes e a hora de entrada. Por exemplo: João Silva, 09:00:00.

---

15. **Moradores com Mais de um Veículo**
    **Descrição:** Exibe os moradores que possuem mais de um veículo registrado, com o nome do morador.

```sql
SELECT Morador.nome

FROM Morador

JOIN Veiculo ON Morador.id_morador = Veiculo.id_morador

GROUP BY Morador.id_morador

HAVING COUNT(Veiculo.id_veiculo) > 1;
```

**Resultado Esperado:**

Exibe o nome do morador. Por exemplo: João Silva.

---

16. **Visitantes que Estiveram no Condomínio em um Período Específico**
    **Descrição:** Exibe os visitantes que estiveram no condomínio entre as datas de 01/11/2024 e 30/11/2024.
    **Código SQL:**

```sql
SELECT nome, data_visita

FROM Visitante

WHERE data_visita BETWEEN '2024-11-01' AND '2024-11-30';
```

**Resultado Esperado:**

Exibe os nomes dos visitantes e a data de visita. Por exemplo: João Silva, 05/11/2024.

---

17. **Moradores com Veículos em Manutenção (Usando Status)**
    **Descrição:** Exibe os moradores que possuem veículos com status de "Em manutenção", com o nome do morador e o modelo do veículo.
    **Código SQL:**

```sql
SELECT Morador.nome, Veiculo.modelo

FROM Morador

JOIN Veiculo ON Morador.id_morador = Veiculo.id_morador

WHERE Veiculo.status = 'Em manutenção';
```

**Resultado Esperado:**

Exibe o nome do morador e o modelo do veículo. Por exemplo: João Silva, Fusca.

---

18. **Total de Visitantes por Tipo de Visita**
    **Descrição:** Exibe o total de visitantes classificados por tipo de visita (entrada ou saída), com a quantidade de visitas para cada tipo.
    **Código SQL:**

```sql
SELECT tipo, COUNT(*) AS total

FROM Entrada_Saida

GROUP BY tipo;
```

**Resultado Esperado:**

Exibe o tipo de visita (entrada ou saída) e o total de ocorrências. Por exemplo: Entrada, 30.

19. **Visitantes Cadastrados em Ano Específico**
    **Descrição:** Exibe os visitantes cadastrados no ano de 2024, com o nome e o ano da visita.
    **Código SQL:**

```sql
SELECT nome, YEAR(data_visita) AS ano

FROM Visitante

WHERE YEAR(data_visita) = 2024;
```

**Resultado Esperado:**

Exibe o nome dos visitantes e o ano da visita. Por exemplo: João Silva, 2024.

---

20. **Veículos que Não Foram Utilizados no Último Mês**
    **Descrição:** Exibe os veículos que não foram utilizados no último mês, com o modelo e a placa do veículo.
    **Código SQL:**

```sql
SELECT Veiculo.modelo, Veiculo.placa

FROM Veiculo

LEFT JOIN Entrada_Saida ON Veiculo.id_veiculo =
Entrada_Saida.id_veiculo

WHERE MONTH(Entrada_Saida.data) != MONTH(CURDATE()) OR
Entrada_Saida.data IS NULL;
```

**Resultado Esperado:**

Exibe o modelo e a placa dos veículos que não foram utilizados. Por exemplo: Fusca, ABC123.

---

21. **Visitantes Que Entraram Durante o Fim de Semana**
    **Descrição:** Exibe os visitantes que entraram no condomínio durante o fim de semana (sábado e domingo), com o nome, data da visita e hora de entrada.
    **Código SQL:**

```sql
SELECT nome, data_visita, hora_entrada

FROM Visitante

WHERE DAYOFWEEK(data_visita) IN (1, 7);
```

**Resultado Esperado:**

Exibe os nomes dos visitantes, data de visita e hora de entrada. Por exemplo: João Silva, 30/11/2024, 10:00:00.

---

22. **Moradores que Não Possuem Veículos**
    **Descrição:** Exibe os moradores que não possuem veículos registrados no sistema, com o nome do morador.
    **Código SQL:**

```sql
SELECT Morador.nome

FROM Morador

LEFT JOIN Veiculo ON Morador.id_morador = Veiculo.id_morador

WHERE Veiculo.id_veiculo IS NULL;
```

**Resultado Esperado:**

Exibe o nome dos moradores sem veículos registrados. Por exemplo: Maria Santos.

---

23. **Visitantes com Hora de Entrada Após um Certo Horário**
    **Descrição:** Exibe os visitantes que entraram no condomínio após as 15:00:00, com o nome e a hora de entrada.
    **Código SQL:**

```sql
SELECT nome, hora_entrada

FROM Visitante

WHERE hora_entrada > '15:00:00';
```

**Resultado Esperado:**

Exibe os nomes dos visitantes e a hora de entrada. Por exemplo: João Silva, 16:30:00.

---

24. **Total de Entradas e Saídas por Visitante**
    **Descrição:** Exibe o total de entradas e saídas de cada visitante, com o nome e o número total de movimentos.
    **Código SQL:**

```sql
SELECT Visitante.nome, COUNT(*) AS total_movimentos
FROM Entrada_Saida
JOIN Visitante ON Entrada_Saida.id_visitante =
Visitante.id_visitante
GROUP BY Visitante.id_visitante;
```

**Resultado Esperado:**

Exibe o nome do visitante e o total de movimentos (entradas e saídas). Por exemplo: João Silva, 5.

---

25. **Moradores com Veículos de Cores Diferentes**
    **Descrição:** Exibe os moradores que possuem veículos de cores diferentes, com o nome do morador e a lista de cores de veículos.
    **Código SQL:**

```sql
SELECT Morador.nome, GROUP_CONCAT(DISTINCT Veiculo.cor) AS
cores_veiculos
FROM Morador
JOIN Veiculo ON Morador.id_morador = Veiculo.id_morador
GROUP BY Morador.id_morador;
```

**Resultado Esperado:**

Exibe o nome do morador e as cores dos veículos. Por exemplo: João Silva, Vermelha, Azul.

---

26. **Visitantes com Documentos Diferentes**
    **Descrição:** Exibe os visitantes com documentos diferentes (quando mais de um documento foi fornecido), com o nome e documento.
    **Código SQL:**

```sql
SELECT nome, documento

FROM Visitante

GROUP BY documento

HAVING COUNT(*) > 1;
```

**Resultado Esperado:**

Exibe os nomes e documentos dos visitantes. Por exemplo: João Silva, 123456789.

---

27. **Visitantes com Mais de uma Visita no Mesmo Dia**
    **Descrição:** Exibe os visitantes que fizeram mais de uma visita no mesmo dia, com o nome, data da visita e número de visitas.
    **Código SQL:**

```sql
SELECT nome, data_visita, COUNT(*) AS total_visitas

FROM Visitante

GROUP BY nome, data_visita

HAVING COUNT(*) > 1;
```

**Resultado Esperado:**

Exibe o nome do visitante, a data e o número de visitas. Por exemplo: João Silva, 02/12/2024, 2.

---

28. **Visitantes com Visitas em Dias de Semana**
    **Descrição:** Exibe os visitantes que entraram no condomínio durante a semana (segunda a sexta), com nome, data e hora de entrada.
    **Código SQL:**

```sql
SELECT nome, data_visita, hora_entrada

FROM Visitante

WHERE DAYOFWEEK(data_visita) BETWEEN 2 AND 6;
```

**Resultado Esperado:**

Exibe os nomes, data e hora de entrada. Por exemplo: João Silva, 02/12/2024, 08:30:00.

---

29. **Moradores com Veículos de Modelo 'Fusca' ou 'Civic'**
    **Descrição:** Exibe os moradores com veículos de modelo 'Fusca' ou 'Civic'.

```sql
SELECT Morador.nome, Veiculo.modelo

FROM Morador

JOIN Veiculo ON Morador.id_morador = Veiculo.id_morador

WHERE Veiculo.modelo IN ('Fusca', 'Civic');
```

**Resultado Esperado:**

Exibe os nomes dos moradores e os modelos de veículos. Por exemplo: João Silva, Fusca.

---

30. Moradores com Veículos que Foram Utilizados Mais de uma Vez no Último Mês
    Descrição: Exibe os moradores com veículos que foram utilizados mais de uma vez no último mês.
    Código SQL:

```sql
SELECT Morador.nome

FROM Morador

JOIN Veiculo ON Morador.id_morador = Veiculo.id_morador

JOIN Entrada_Saida ON Veiculo.id_veiculo =
Entrada_Saida.id_veiculo
```

```
WHERE MONTH(Entrada_Saida.data) = MONTH(CURDATE())
GROUP BY Morador.id_morador
HAVING COUNT(Entrada_Saida.id_entrada_saida) > 1;
```

**Resultado Esperado:**

Exibe o nome do morador. Por exemplo: João Silva.

## Considerações Finais

O desenvolvimento do Sistema de Portaria de Condomínio mostrou como um banco de dados bem estruturado pode facilitar a gestão de informações importantes e melhorar a segurança em um condomínio. Com ele, é possível gerar relatórios, monitorar acessos e manter registros precisos, o que contribui diretamente para uma administração mais eficiente.

Como sugestão de melhorias, o sistema pode ser integrado com tecnologias como leitores biométricos e câmeras de segurança, além de uma interface web ou mobile para facilitar o acesso a informações pelos administradores.

---

## Referências Bibliográficas


ELMASRI, R.; NAVATHE, S. B. *Sistemas de Banco de Dados*. 7. ed. Pearson, 2016.

DATE, C. J. *Introdução a Sistemas de Banco de Dados*. 8. ed. Addison-Wesley, 2004.

KROENKE, D. M.; AUER, D. *Banco de Dados para Administração*. 11. ed. Pearson, 2013.