



Desenvolvimento de um
projeto de **BASE DE DADOS**
PARA UM STAND DE AUTOMÓVEIS

Edgar Sousa (28557) | Renato Azevedo (28549) | Ricardo Rocha (28570)

Armazenamento e Acesso a Dados – 2º Semestre | CTESP - DWM

Vila Nova de Famalicão, 18 de junho de 2024

EPÍGRAFE

*"Dados são o novo petróleo. É valioso, mas se não for refinado,
não pode realmente ser usado."*

Clive Humby

ÍNDICE DE CONTEÚDOS

ÍNDICE DE FIGURAS.....	4
INTRODUÇÃO AO TEMA.....	6
DESCRIÇÃO DO PROBLEMA	7
MODELAÇÃO E NORMALIZAÇÃO DE TABELAS E ESTRUTURAS DA BASE DE DADOS.....	8
a. MODELO ER.....	8
b. MODELO LÓGICO / NORMALIZAÇÃO.....	9
IMPLEMENTAÇÃO DO MODELO FÍSICO	11
a. CRIAÇÃO DA BASE DE DADOS	11
b. CRIAÇÃO DAS TABELAS.....	11
c. INSERIR DADOS NAS TABELAS.....	16
d. ALTERAR OS DADOS DAS TABELAS.....	19
e. CONSULTAS À BASE DE DADOS	22
CRIAÇÃO DE VIEWS.....	25
PROCEDIMENTOS	27
FUNÇÕES	29
TRIGGERS.....	31
CONCLUSÃO	34
BIBLIOGRAFIA.....	35
ANEXOS.....	36
ANEXO 1: FICHA DE IDENTIFICAÇÃO DO GRUPO.....	37

ÍNDICE DE FIGURAS

Figura 1- Logotipo	7
Figura 2- Modelo ER	8
Figura 3- Modelo Lógico	9
Figura 4- Criação da Base de dados.....	11
Figura 5- Comando em SQL para a criação da tabela 'Marca'	11
Figura 6 - Criação da tabela 'Fornecedor' em SQL.....	12
Figura 7- Criação da tabela auxiliar.....	12
Figura 8 - Comando em SQL para criação da tabela 'Modelo'	13
Figura 9 - Comando em SQL para criação da tabela 'Matricula'	13
Figura 10 - Comando em SQL para criação da tabela 'Proprietario'	13
Figura 11- Comando em SQL para criação da tabela 'Automóvel'	14
Figura 12 - Visão Geral da estrutura da tabela 'automovel' no PHPmyAdmin	14
Figura 13 - Tabelas e Relações - Desenhador PHPmyAdmin	15
Figura 14 - Dados inseridos na tabela 'Marca'.....	16
Figura 15- Dados inseridos na tabela 'Fornecedor'	17
Figura 16- Dados inseridos na tabela 'Modelo'	17
Figura 17- Dados inseridos na tabela 'Proprietario'	17
Figura 18- Dados inseridos na tabela 'Matricula'	17
Figura 19 - Dados inseridos na tabela 'Automovel'	18
Figura 20 - Dados inseridos na tabela 'Marca-Fornecedor'	18
Figura 21 - Adicionar coluna 'telefone' à tabela fornecedor	19
Figura 22- Alterações à tabela 'Fornecedor'	19
Figura 23 - Comando SQL para eliminar coluna	20
Figura 24 - Alterar tipo de dados	20
Figura 25 - Estrutura da tabela 'proprietario'	20
Figura 26 - Alterar uma linha da tabela 'matricula'.....	21
Figura 27 - Consultar todos os veículos presentes na tabela 'automóvel'	22
Figura 28 - Soma do valor de todos os carros do stand	22
Figura 29 - Consulta da média de preços dos automóveis.....	23
Figura 30- Ordenar proprietários por ordem alfabética.....	23
Figura 31 - Consulta a 2 tabelas	24
Figura 32 - Criação e Resultado da view 1	25
Figura 33 - SQL para criação da View e Resultado.....	25

Figura 34- View Automóveis e Proprietários	26
Figura 35 - Procedimento para atualizar a cor do veículo	27
Figura 36- Resultado da aplicação do procedimento 1.....	27
Figura 37 - Procedimento para atualizar o preço de um automóvel.....	28
Figura 38 - Função para calcular a média do preço dos veículos.....	29
Figura 39 - Obter o número de automóveis por proprietário	30
Figura 40 - Obter o número de automóveis por proprietário (continuação).....	31
Figura 41 - Trigger Validação de Preço para novos carros que sejam adicionados ...	31
Figura 42 - Teste trigger 1	32
Figura 43 - Trigger para impedir a redução do preço do veículo	32
Figura 44 - Teste trigger 2.....	33

INTRODUÇÃO AO TEMA

A gestão eficiente de dados é um pilar fundamental para o sucesso de qualquer empresa no mundo atual, especialmente no setor automóvel. Este trabalho final, desenvolvido no âmbito da unidade curricular de Armazenamento e Acesso a Dados, visa apresentar a criação e configuração de uma base de dados SQL para a *Supremo Motors*, um stand de automóveis fictício. A *Supremo Motors* enfrenta o desafio de gerir uma vasta gama de informações, desde marcas e modelos de automóveis até fornecedores, carros, proprietários e matrículas. Para otimizar a gestão destas informações e melhorar o atendimento aos clientes, torna-se imprescindível uma base de dados integrada e bem estruturada.

Para alcançar os objetivos deste projeto, utilizámos o phpMyAdmin, uma ferramenta eficaz para a administração de bases de dados *MySQL*, que nos permite gerir e configurar a base de dados de forma intuitiva. Este relatório detalha todo o processo de desenvolvimento da base de dados da *Supremo Motors*, desde a definição inicial dos requisitos até à implementação prática das tabelas e dos relacionamentos entre elas. Serão abordados os passos essenciais para a criação da base de dados, incluindo a construção de um modelo ER (Entidade-Relacionamento), um modelo lógico e a sua respetiva normalização, de forma a garantir uma estrutura coerente e eficiente no modelo físico. Apresentamos exemplos de comandos SQL, como INSERT, CREATE, SELECT, TRIGGER, PROCEDURE e FUNCTION, para ilustrar a manipulação dos dados. Este trabalho não se limita à aplicação dos conhecimentos teóricos adquiridos na unidade curricular; pretende também oferecer uma perspetiva prática sobre como as bases de dados podem ser utilizadas no contexto empresarial real. A nossa abordagem procura demonstrar como uma gestão rigorosa e segura da informação pode contribuir significativamente para a eficiência operacional e a qualidade do atendimento no setor automóvel.

O presente relatório serve de suporte ao ficheiro da base de dados criado no phpMyAdmin, que se encontra no mesmo diretório deste documento. O ficheiro, em formato '.sql', intitula-se 'supremo_db.sql'. Este documento foi estruturado de forma sequencial, abordando os temas de modo a culminar na implementação da base de dados e na explicação dos aspetos mais relevantes das decisões tomadas.

Estamos prontos para começar esta jornada. Acompanhe-nos nas próximas páginas e descubra como a *Supremo* pode beneficiar de uma gestão de dados eficiente e segura, potenciando a melhoria da gestão do seu negócio. **Seja bem-vindo!**

DESCRIÇÃO DO PROBLEMA

A 'Supremo Motors', uma empresa fictícia de renome no setor automóvel, procura otimizar a gestão do seu stand de automóveis. Para melhorar a eficiência operacional e proporcionar um melhor atendimento aos seus clientes, a 'Supremo Motors' necessita de uma base de dados integrada que facilite a gestão de todas as informações relacionadas aos automóveis, matrículas, fornecedores, marcas e modelos. Para além disso, a 'Supremo Motors' também efetua a venda de automóveis, que não pertencem ao stand, surgindo a necessidade de associar o automóvel a um proprietário, caso exista.

Para tornar a "Supremo Motors" mais real, foi criado por inteligência artificial (<https://ideogram.ai/>) um logótipo original (figura 1).

Figura 1- Logotipo



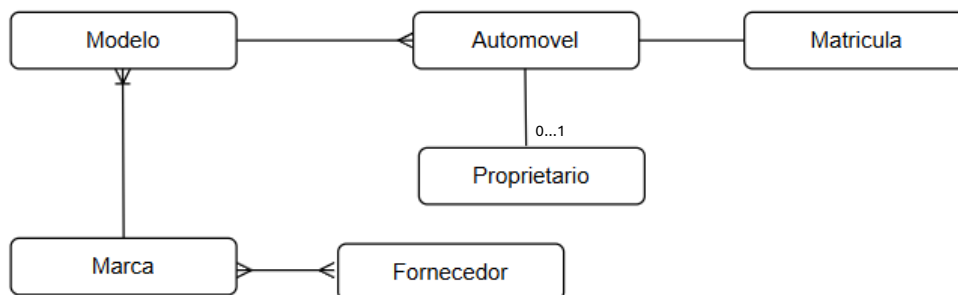
Posto isto, pretende-se desenvolver uma base de dados em *MySQL* para gerir o inventário de automóveis do stand, a matrículas associada a cada veículo, as informações dos fornecedores e as marcas que comercializam, os modelos de automóveis disponíveis e o respetivo preço e o proprietário de cada veículo (caso exista).

MODELAÇÃO E NORMALIZAÇÃO DE TABELAS E ESTRUTURAS DA BASE DE DADOS

a. MODELO ER

- **Entidades:** Modelo, Automovel, Matricula, Proprietario, Marca, Fornecedor.
- **Relacionamentos:** Um-para-um (Automóvel-Matrícula), um-para-muitos (Modelo-Automóvel; Marca-Modelo) e Muitos-para-muitos (Fornecedores-Marca)
- **Cardinalidades:**
 - Um automóvel só pode estar associado a uma matrícula e uma matrícula apenas pode estar associado a um automóvel (1,1);
 - Um automóvel pode ou não estar associado a um proprietário (1 , 0...1);
 - Um modelo de automóvel pode ter vários veículos no stand (1,n), mas um automóvel só pode pertencer a um determinado modelo (1,1);
 - Uma marca pode ter vários modelos de automóveis (1,n), contudo um modelo de automóvel só pode pertencer a uma marca (1,1);
 - Uma marca pode ter vários fornecedores (1,n) e vários fornecedores podem vender marcas (n,n)

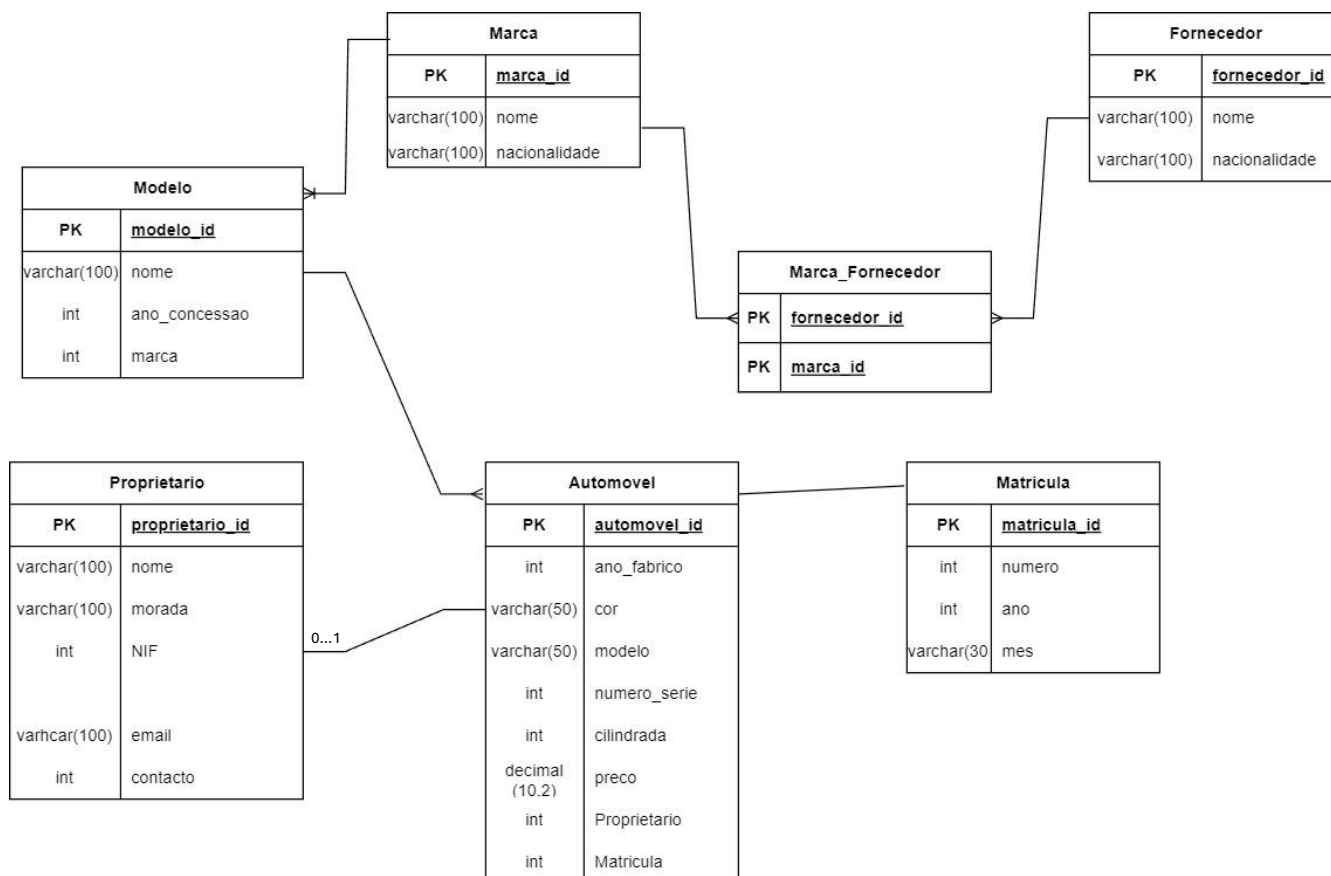
Figura 2- Modelo ER



b. MODELO LÓGICO / NORMALIZAÇÃO

Assim sendo, vamos explorar detalhadamente o processo de normalização aplicado ao modelo lógico do stand de automóveis (figura 3), para cada tabela ser projetada para atender aos requisitos das formas normais e assegurar uma estrutura de dados coerente e eficiente (figura 3).

Figura 3- Modelo Lógico



Primeira Forma Normal

A primeira forma normal, assegura que os dados sejam armazenados em uma tabela, onde cada coluna contém apenas valores atômicos, ou seja, um só dado por cada registo e cada campo contém um único valor. Se olharmos para o nosso modelo, vemos que:

- Todas as tabelas possuem colunas com valores atômicos.
- Não há colunas repetitivas ou multi-valores.
- Cada tabela tem uma chave primária claramente definida.

Segunda Forma Normal

A segunda forma normal implica que o modelo cumpra a primeira forma normal, e que cada atributo *não chave* tem de ser funcionalmente dependente da totalidade da chave primária e não apenas de uma parte dessa chave. Posto isto no nosso modelo, vemos que:

- Tabela **Marca**: 'nome' e 'nacionalidade' dependem totalmente da chave primária 'marca_id';
- Tabela **Fornecedor**: 'nome' e 'nacionalidade' dependem totalmente da chave primária fornecedor_id;
- Tabela **Marca_Fornecedor**: Esta é uma tabela auxiliar que não possui atributos adicionais além das chaves primárias;
- Tabela **Modelo**: 'nome' e 'ano_concessao' dependem totalmente da chave primária 'modelo_id';
- Tabela **Automovel**: Todos os atributos dependem totalmente da chave primária 'automovel_id';
- Tabela **Proprietario**: Todos os atributos dependem totalmente da chave primária 'proprietario_id';
- Tabela **Matricula**: Todos os atributos dependem totalmente da chave primária 'matricula_id'.

Terceira Forma Normal

A terceira forma normal, tem de estar na segunda forma normal e que nenhum atributo *não chave* pode depender funcionalmente de algum outro atributo que não seja a chave primária. Vamos analisar o nosso modelo, a tabela:

- **Marca**: 'nome' e 'nacionalidade' dependem diretamente de 'marca id'.
- **Fornecedor**: 'nome' e 'nacionalidade' dependem diretamente de 'fornecedor_id'.
- **Marca_Fornecedor**: Tabela auxiliar sem atributos adicionais, sem dependências transitivas.
- **Modelo**: 'nome' e 'ano_concessao' dependem diretamente de 'modelo_id'.
- **Automóvel**: 'ano_fabrico', 'cor', 'modelo', 'numero_serie' e 'cilindrada' dependem diretamente de 'automovel_id'.
- **Proprietário**: 'nome', 'morada', 'NIF', 'email' e 'contacto' dependem diretamente de 'proprietario_id'.
- **Matrícula**: 'numero', 'ano' e 'mes' dependem diretamente de 'matricula_id'.

Concluimos assim que, que o modelo lógico criado encontra-se na 3ª forma normal, e está, portanto, normalizado.

IMPLEMENTAÇÃO DO MODELO FÍSICO

O procedimento começa com a criação da base de dados, das tabelas nela presentes e das chaves estrangeiras. Para cada passo, apresentamos imagens correspondentes de como foi realizado. Em vez de criarmos a base de dados através da linha de comandos, utilizámos o software phpMyAdmin para facilitar a visualização e a criação das bases de dados.

a. CRIAÇÃO DA BASE DE DADOS

Vamos começar por criar a base de dados da Supremo Motors (figura 4).

Figura 4- Criação da Base de dados

```
1 CREATE DATABASE supremo_db;
```

b. CRIAÇÃO DAS TABELAS

Após a criação da base de dados, criamos as tabelas necessárias. Cada tabela representa uma entidade do stand de automóveis e contém atributos específicos dessa entidade.

Tabela 'Marca'

A tabela 'Marca' armazena dados sobre as marcas dos automóveis do stand. Possui 3 colunas, a coluna 'marca_id', que corresponde à chave primária desta tabela do tipo inteiro e as colunas nome e nacionalidade são do tipo *varchar* (figura 5).

Figura 5- Comando em SQL para a criação da tabela 'Marca'

```
CREATE TABLE Marca (  
    marca_id INT AUTO_INCREMENT PRIMARY KEY,  
    nome VARCHAR(100),  
    nacionalidade VARCHAR(100)  
);
```

Tabela 'Fornecedor'

A tabela 'Fornecedor' armazena informações sobre os fornecedores dos veículos. Composta por 3 colunas: 'fornecedor_id' que é a chave primária do tipo inteiro e 'nome' e 'nacionalidade' que são do tipo varchar (figura 6).

Figura 6 - Criação da tabela 'Fornecedor' em SQL

```
CREATE TABLE Fornecedor (  
    fornecedor_id INT AUTO_INCREMENT PRIMARY KEY,  
    nome VARCHAR(100),  
    nacionalidade VARCHAR(100)  
);
```

Tabela 'Marca_Fornecedor'

A tabela 'Marca_fornecedor' é uma tabela intermédia que guarda os relacionamentos entre as tabelas marcas e fornecedores, consequência de um relacionamento muitos-para-muitos entre estas duas tabelas. Possui 2 colunas com chaves estrangeiras, a 'fornecedor_id' e 'marca_id' do tipo inteiro (figura 7).

Figura 7- Criação da tabela auxiliar

```
CREATE TABLE Marca_Fornecedor (  
    fornecedor_id INT,  
    marca_id INT,  
    FOREIGN KEY (fornecedor_id) REFERENCES Fornecedor(fornecedor_id),  
    FOREIGN KEY (marca_id) REFERENCES Marca(marca_id)  
);
```

Tabela 'Modelo'

A tabela Modelo guarda dados sobre os modelos dos automóveis. É composta por 4 colunas. A coluna 'modelo_id' é a chave primária do tipo inteiro. As colunas 'nome' e 'ano_concessao' são do tipo varchar. A coluna 'marca' é uma chave estrangeira que referencia à tabela 'Marca', assim sendo, estabelece-se um relacionamento entre o modelo e a marca.

Figura 8 - Comando em SQL para criação da tabela 'Modelo'

```
CREATE TABLE Modelo (  
    modelo_id INT AUTO_INCREMENT PRIMARY KEY,  
    nome VARCHAR(100),  
    ano_concessao INT  
);
```

```
1 ALTER TABLE Modelo  
2 ADD COLUMN marca INT,  
3 ADD FOREIGN KEY (marca) REFERENCES Marca(marca_id);
```

Tabela 'Matricula'

A tabela 'Matricula' armazena dados da matrícula do veículo. Possui 4 colunas. A coluna 'matricula_id' é a chave primária. As colunas 'número', 'ano' são do tipo inteiro e a coluna 'mês' é do tipo varchar.

Figura 9 - Comando em SQL para criação da tabela 'Matricula'

```
CREATE TABLE Matricula (  
    matricula_id INT AUTO_INCREMENT PRIMARY KEY,  
    numero INT,  
    ano INT,  
    mes VARCHAR(30)  
);
```

Tabela 'Proprietario'

A tabela 'Proprietario' guarda dados dos proprietários dos veículos. A tabela possui 6 colunas. A coluna 'proprietario_id' é a chave primária. As colunas 'nome', 'morada', 'email' são do tipo varchar e o 'nif' e o 'contacto' são do tipo inteiro.

Figura 10 - Comando em SQL para criação da tabela 'Proprietario'

```
CREATE TABLE Proprietario (  
    proprietario_id INT AUTO_INCREMENT PRIMARY KEY,  
    nome VARCHAR(100),  
    morada VARCHAR(100),  
    nif INT,  
    email VARCHAR(100),  
    contacto INT  
);
```

Tabela Automovel

A tabela 'automovel' guarda dados sobre os automóveis. É a maior tabela da base de dados possuindo 9 colunas (figura 11). A coluna 'automovel_id' é a chave primária. A coluna 'preco' é do tipo decimal(10,2) isto significa que o preço pode ter 10 dígitos e 2 depois da vírgula. As colunas 'ano_fabrico', 'numero_serie', 'cilindrada', são do tipo inteiro e a coluna 'cor' é do tipo varchar. As colunas 'matricula', 'modelo', 'proprietario' referenciam as tabelas Matricula, Modelo e Proprietario.

Figura 11- Comando em SQL para criação da tabela 'Automóvel'

```
CREATE TABLE Automovel (
  automovel_id INT AUTO_INCREMENT PRIMARY KEY,
  ano_fabrico INT,
  cor VARCHAR(50),
  modelo VARCHAR(50),
  numero_serie INT,
  cilindrada INT
);

1 ALTER TABLE Automovel
2 ADD matricula INT,
3 ADD FOREIGN KEY (matricula) REFERENCES Matricula(matricula_id);

1 ALTER TABLE Automovel
2 ADD COLUMN Preco DECIMAL(10,2);

1 ALTER TABLE Automovel
2 ADD COLUMN Proprietario INT,
3 ADD FOREIGN KEY (Proprietario) REFERENCES Proprietario(proprietario_id);

1 ALTER TABLE Automovel
2 ADD COLUMN modelo INT,
3 ADD FOREIGN KEY (modelo) REFERENCES Modelo(modelo_id);
```

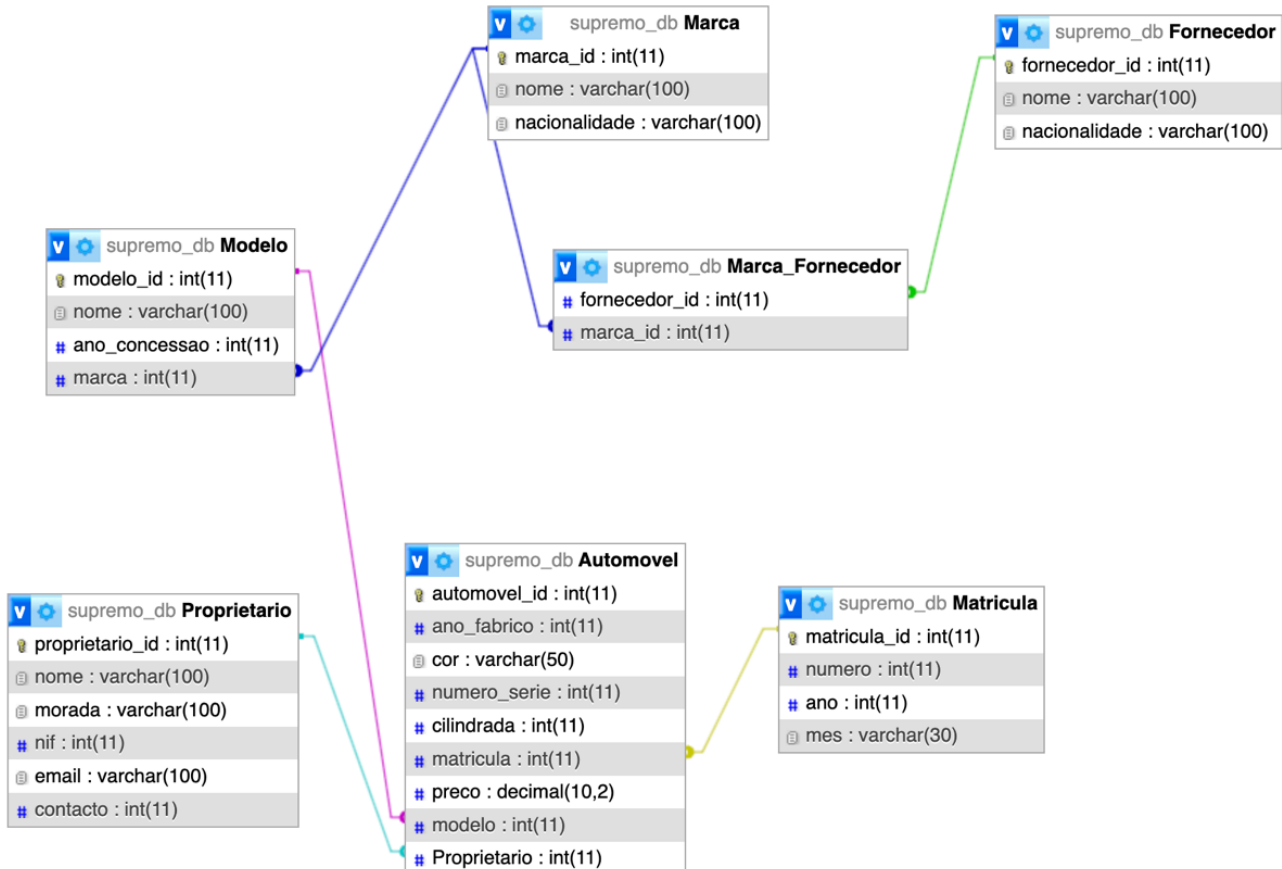
Figura 12 - Visão Geral da estrutura da tabela 'automovel' no PHPmyAdmin

#	Nome	Tipo	Agrupamento (Collation)	Atributos	Nulo	Predefinido	Comentários	Extra	Ação
<input type="checkbox"/>	1 automovel_id	int(11)			Não	Nenhum		AUTO_INCREMENT	
<input type="checkbox"/>	2 ano_fabrico	int(11)			Sim	NULL			
<input type="checkbox"/>	3 cor	varchar(50)	utf8mb4_general_ci		Sim	NULL			
<input type="checkbox"/>	4 numero_serie	int(11)			Sim	NULL			
<input type="checkbox"/>	5 cilindrada	int(11)			Sim	NULL			
<input type="checkbox"/>	6 matricula	int(11)			Sim	NULL			
<input type="checkbox"/>	7 preco	decimal(10,2)			Sim	NULL			
<input type="checkbox"/>	8 modelo	int(11)			Sim	NULL			
<input type="checkbox"/>	9 Proprietario	int(11)			Sim	NULL			

☐ Marcar todos Com os seleccionados:

Adiante, vamos conferir o diagrama relacional que ajuda a visualizar os relacionamentos entre as tabelas (figura 13). É uma representação gráfica que mostra como as tabelas estão “ligadas” através de chaves primárias e estrangeiras. Para visualizar o diagrama utilizamos uma opção chamada de desenhador no phpmyadmin.

Figura 13 - Tabelas e Relações - Desenhador PHPmyAdmin



c. INSERIR DADOS NAS TABELAS

Para garantir a integridade referencial na nossa base de dados, os dados foram inseridos na ordem correta, começando pelas tabelas que não possuem dependências de chaves estrangeiras. Desta forma, conseguimos assegurar que todas as relações entre as tabelas estão corretamente estabelecidas. Por ordem de inserção de dados nas tabelas, temos:

- **Tabela 'Marca'**: Não possui chaves estrangeiras, então será a primeira a receber dados;
- **Tabela 'Fornecedor'**: Não possui chaves estrangeiras, então será a segunda tabela a receber dados;
- **Tabela 'Modelo'**: Esta tabela possui uma chave estrangeira que referencia a tabela 'Marca'. Daí, só fazer sentido agora inserir os dados nesta tabela;
- **Tabela 'Proprietario'**: Não possui chaves estrangeiras. Foi a quarta tabela onde inserimos os dados;
- **Tabela 'Matricula'**: Não possui chaves estrangeiras. Foi a quinta tabela onde inserimos os dados;
- **Tabela 'Automovel'**: Esta tabela possui chaves estrangeiras que fazem referência às tabelas Matricula, Modelo e Proprietario. Por isso, só faz sentido neste momento inserir os dados nesta tabela;
- **Tabela 'Marca_Fornecedor'**: Esta é uma tabela intermédia que faz referência à tabela 'Marca' e 'Fornecedor', então só faz sentido abordá-la depois das mesmas estarem com dados inseridos.

Depois desta tomada de consciência, estamos prontos para inserir os dados, para isso usou-se o ambiente visual do PHPmyAdmin. O resultado encontra-se nas figuras seguintes (figuras 14, 15, 16, 17, 18, 19 e 20).

Figura 14 - Dados inseridos na tabela 'Marca'
















				marca_id	nome	nacionalidade
<input type="checkbox"/>	 Editar	 Copiar	 Apagar	1	Fiat	Italiana
<input type="checkbox"/>	 Editar	 Copiar	 Apagar	2	Ford	Americana
<input type="checkbox"/>	 Editar	 Copiar	 Apagar	3	Renault	Francesa
<input type="checkbox"/>	 Editar	 Copiar	 Apagar	4	Volkswagen	Alemã
<input type="checkbox"/>	 Editar	 Copiar	 Apagar	5	Audi	Alemã

Figura 15- Dados inseridos na tabela 'Fornecedor'

















		fornecedor_id	nome	nacionalidade
<input type="checkbox"/>	 Editar  Copiar  Apagar	1	Bosch	Alemã
<input type="checkbox"/>	 Editar  Copiar  Apagar	2	Continental	Alemã
<input type="checkbox"/>	 Editar  Copiar  Apagar	3	TMG Automotive	Portuguesa
<input type="checkbox"/>	 Editar  Copiar  Apagar	4	AUTODOC	Alemã
<input type="checkbox"/>	 Editar  Copiar  Apagar	5	Delphi Technologies	Britânica

Figura 16- Dados inseridos na tabela 'Modelo'

















		modelo_id	nome	ano_concessao	marca
<input type="checkbox"/>	 Editar  Copiar  Apagar	1	Multipla	1998	1
<input type="checkbox"/>	 Editar  Copiar  Apagar	2	Puma	2024	2
<input type="checkbox"/>	 Editar  Copiar  Apagar	3	Clio	2001	3
<input type="checkbox"/>	 Editar  Copiar  Apagar	4	Polo	2012	4
<input type="checkbox"/>	 Editar  Copiar  Apagar	5	Megane	2017	3

Figura 17- Dados inseridos na tabela 'Proprietario'

		proprietario_id	nome	morada	nif	email	contacto
<input type="checkbox"/>	 Editar  Copiar  Apagar	1	Asdrubal Pinheiro	Rua do Quarto 22	234987456	asdrubalpinheiro@gmail.com	915681616
<input type="checkbox"/>	 Editar  Copiar  Apagar	2	Marta Maria	Rua da Ponta 32	334927456	martamaria@gmail.com	915581515
<input type="checkbox"/>	 Editar  Copiar  Apagar	3	Caio Pinto	Rua do Brasil 52	534987456	caiopinto@gmail.com	915681313
<input type="checkbox"/>	 Editar  Copiar  Apagar	4	Marcelo Rebelo de Sousa	Palácio de Cristal 25	734987456	marcelopresidente@portugal.pt	252875561
<input type="checkbox"/>	 Editar  Copiar  Apagar	5	Jacinto Pinto	Rua do Brasil 51	246987456	jacintopinto@gmail.com	915611616

Figura 18- Dados inseridos na tabela 'Matricula'
















		matricula_id	numero	ano	mes
<input type="checkbox"/>	 Editar  Copiar  Apagar	1	49-51-BF	1998	janeiro
<input type="checkbox"/>	 Editar  Copiar  Apagar	2	40-45-FB	2024	fevereiro
<input type="checkbox"/>	 Editar  Copiar  Apagar	3	31-69-CD	2001	dezembro
<input type="checkbox"/>	 Editar  Copiar  Apagar	4	48-50-BF	2012	junho
<input type="checkbox"/>	 Editar  Copiar  Apagar	5	56-12-VF	2017	outubro

Figura 19 - Dados inseridos na tabela 'Automovel'

Lembrete: quando existir campos 'Null' no campo 'Proprietario', significa que é o stand o dono do automóvel.

<div><div></div><div></div></div>				automovel_id	ano_fabrico	cor	numero_serie	cilindrada	matricula	preco	modelo	Proprietario
<div><div></div><div></div></div>	<div><div></div><div></div></div>	<div><div></div><div></div></div>	<div><div></div><div></div></div>	1	1998	Amarelo	681279137	1000	1	3999.99	1	4
<div><div></div><div></div></div>	<div><div></div><div></div></div>	<div><div></div><div></div></div>	<div><div></div><div></div></div>	2	2024	Preto	751279137	1500	2	19999.99	2	5
<div><div></div><div></div></div>	<div><div></div><div></div></div>	<div><div></div><div></div></div>	<div><div></div><div></div></div>	3	2001	Branco	164278137	1200	3	1999.99	3	3
<div><div></div><div></div></div>	<div><div></div><div></div></div>	<div><div></div><div></div></div>	<div><div></div><div></div></div>	4	2012	Azul	561279137	1500	4	5999.99	4	2
<div><div></div><div></div></div>	<div><div></div><div></div></div>	<div><div></div><div></div></div>	<div><div></div><div></div></div>	5	2017	Verde	891278137	1500	5	9999.99	5	NULL

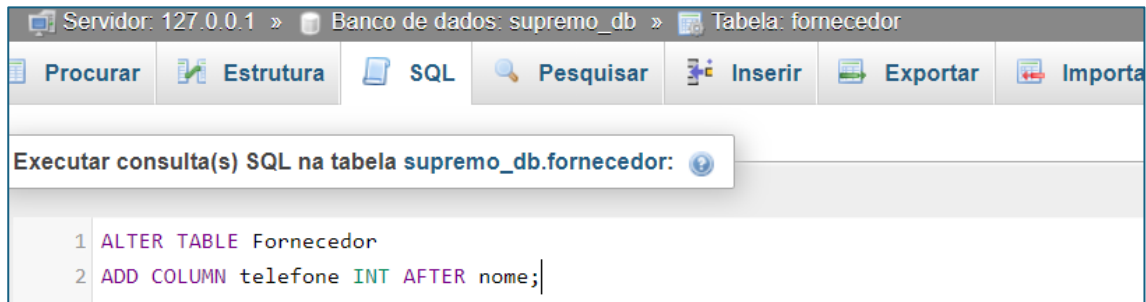
Figura 20 - Dados inseridos na tabela 'Marca-Fornecedor'

fornecedor_id	marca_id
1	1
2	2
3	3
4	4
5	5

d. ALTERAR OS DADOS DAS TABELAS

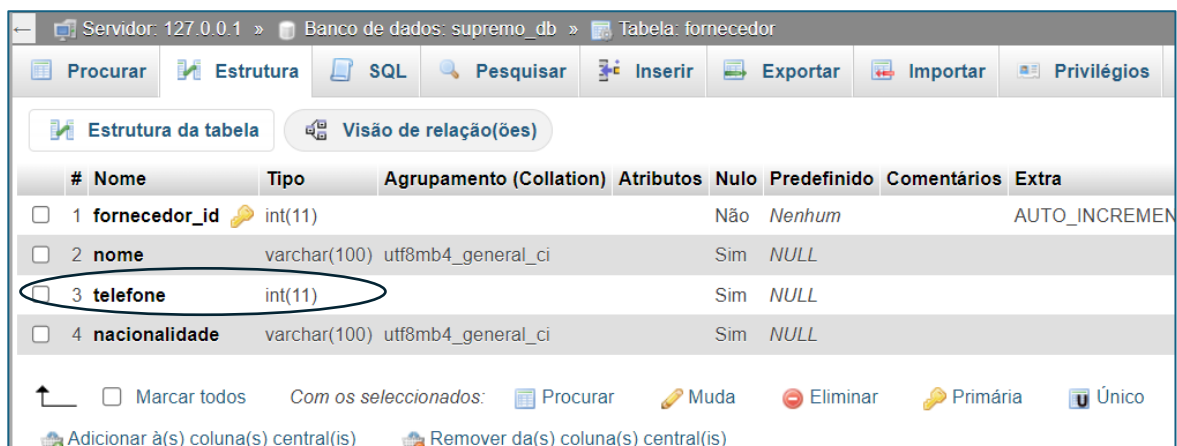
Vamos de seguida, tentar alterar os dados de algumas tabelas. Depois da base de dados criada, o cliente sinalizou-nos que pretendia um campo para armazenar o telefone do fornecedor. Decidimos colocar a coluna 'telefone' logo a seguir à coluna 'nome' usando o comando ALTER TABLE e AFTER (figura 21)

Figura 21 - Adicionar coluna 'telefone' à tabela fornecedor

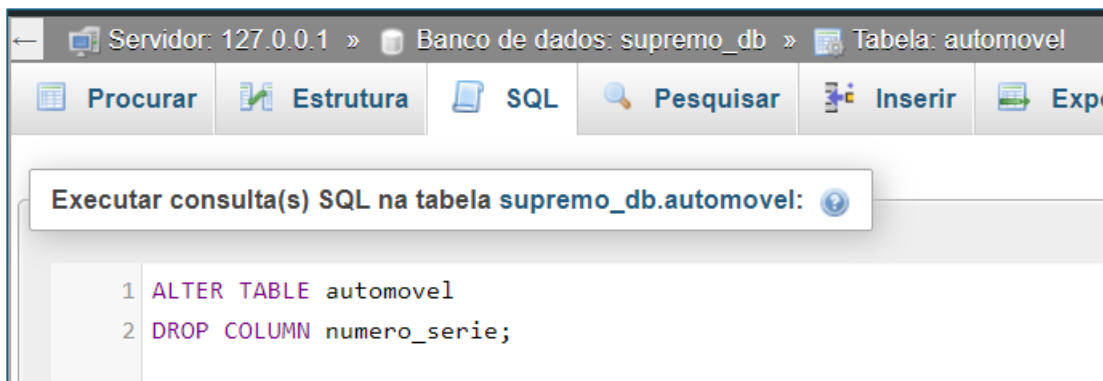


Depois de efetivar o código acima, vamos conferir se realmente funcionou. Pela figura abaixo, vemos que foi adicionada uma nova coluna 'telefone' logo a seguir ao 'nome', o que confirma o sucesso da nossa operação (figura 22).

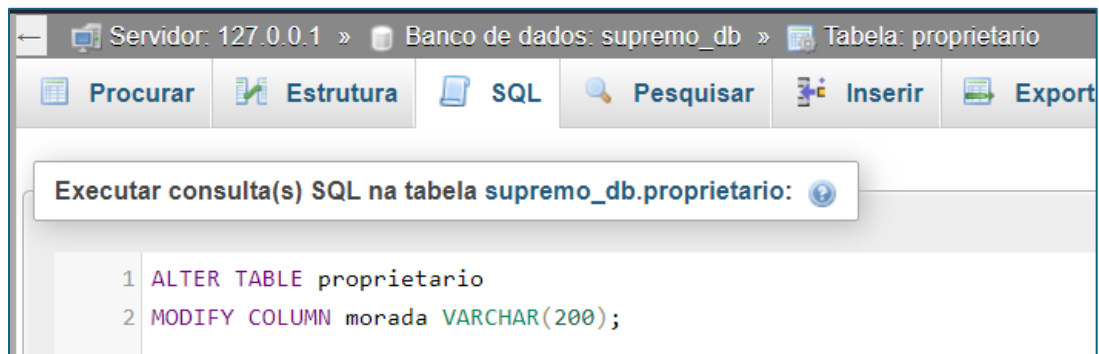
Figura 22- Alterações à tabela 'Fornecedor'



Deixamos de seguida, apenas a título de exemplo e na continuidade do código anterior de acrescentar uma coluna, o código SQL para apagar uma coluna de uma tabela. No exemplo abaixo, elimina da tabela 'automovel' a coluna 'numero_serie'. (figura 23).

Figura 23 - Comando SQL para eliminar coluna

Reparamos que o campo '*morada*' na tabela '*proprietario*' permitia apenas a introdução de 100 caracteres, o que nos pareceu insuficiente dado que este campo inclui nomes de ruas, cidade, localidade e o código-postal. Vamos alterar o número de caracteres permitidos de 100 para 200 (figura 24).

Figura 24 - Alterar tipo de dados

Vamos verificar então, se de facto a nossa alteração surtiu efeito. Pela figura 25, confirmamos o sucesso da operação.

Figura 25 - Estrutura da tabela 'proprietario'

The screenshot shows the 'Estrutura da tabela' (Table Structure) view for the 'proprietario' table. The table has the following columns:

#	Nome	Tipo	Agrupamento (Collation)	Atributos	Nulo	Predefinido	Comentários	Extra
1	proprietario_id	int(11)			Não	Nenhum		AUTO_INCREMENT
2	nome	varchar(100)	utf8mb4_general_ci		Sim	NULL		
3	morada	varchar(200)	utf8mb4_general_ci		Sim	NULL		
4	nif	int(11)			Sim	NULL		
5	email	varchar(100)	utf8mb4_general_ci		Sim	NULL		
6	contacto	int(11)			Sim	NULL		

The 'morada' column is circled in the original image, highlighting the successful change from varchar(100) to varchar(200).

Agora queremos alterar uma linha da tabela 'matricula', nomeadamente os campos 'ano' e 'mês' da matrícula com id=1 (figura 26)

Figura 26 - Alterar uma linha da tabela 'matricula'

The screenshot shows a database management interface. At the top, the breadcrumb navigation indicates the path: Servidor: 127.0.0.1 » Banco de dados: supremo_db » Tabela: matricula. Below this is a toolbar with buttons for 'Procurar', 'Estrutura', 'SQL', 'Pesquisar', 'Inserir', and 'Exportar'. A message box states 'Executar consulta(s) SQL na tabela supremo_db.matricula:'. The SQL editor contains the following query:

```
1 UPDATE matricula
2 SET ano = 1995, mes = 'abril'
3 WHERE matricula_id = 1;
```

An arrow points from the SQL editor to a table view below. The table view shows the 'matricula' table with columns: matricula_id, numero, ano, and mes. The first row (id=1) is highlighted, and its 'ano' and 'mes' values (1995 and abril) are circled, indicating the successful update.

	matricula_id	numero	ano	mes
<input type="checkbox"/> Editar Copiar Apagar	1	49-51-BF	1995	abril
<input type="checkbox"/> Editar Copiar Apagar	2	40-45-FB	2024	fevereiro
<input type="checkbox"/> Editar Copiar Apagar	3	31-69-CD	2001	dezembro
<input type="checkbox"/> Editar Copiar Apagar	4	48-50-BF	2012	junho
<input type="checkbox"/> Editar Copiar Apagar	5	56-12-VF	2017	outubro

e. CONSULTAS À BASE DE DADOS

Nesta secção, mostramos as várias consultas SQL para analisar a base de dados do stand. As consultas que se apresentam abaixo, são acompanhadas com a descrição e os resultados obtidos.

Listar todos os veículos disponíveis no stand (figura 27)

Esta consulta mostra-nos todos os automóveis disponíveis no stand. É útil para fazer o inventário completo dos carros que temos.

Figura 27 - Consultar todos os veículos presentes na tabela 'automóvel'

1 `SELECT * FROM automovel;`

✓ A mostrar registos de 0 - 4 (5 total, A consulta demorou 0.0004 segundos.)

`SELECT * FROM automovel;`

☐ Perfil [\[Editar em linha \]](#) [\[Editar \]](#) [\[Explicar SQL \]](#) [\[Criar código PHP \]](#) [\[Atualizar \]](#)

☐ Mostrar tudo | Número de registos: 25 | Filtrar registos: Ordenar pela chave: Nenhum

Opções extra

	automovel_id	ano_fabrico	cor	numero_serie	cilindrada	matricula	preco	modelo	Proprietario
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Apagar	1	1998	Amarelo	681279137	1000	1	3999.99	1	4
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Apagar	2	2024	Preto	751279137	1500	2	19999.99	2	5
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Apagar	3	2001	Branco	164278137	1200	3	1999.99	3	3
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Apagar	4	2012	Azul	561279137	1500	4	5999.99	4	2
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Apagar	5	2017	Verde	891278137	1500	5	9999.99	5	NULL

Consultar o valor total dos veículos no stand (figura 28)

Esta consulta permite-nos obter o preço total de todos os automóveis que temos no stand e apresenta-nos o resultado com o nome de 'preco_total'

Figura 28 - Soma do valor de todos os carros do stand

1 `SELECT SUM(preco) AS preco_total FROM automovel;`

✓ A mostrar registos de 0 - 0 (1 total, A consulta demorou 0.0013 segundos.)

`SELECT SUM(preco) AS preco_total FROM automovel;`

☐ Perfil [\[Editar em linha \]](#) [\[Editar \]](#) [\[Explicar SQL \]](#) [\[Criar código PHP \]](#) [\[Atualizar \]](#)

☐ Mostrar tudo | Número de registos: 25 | Filtrar registos:

Opções extra

preco_total
41999.95

Conhecer a média dos preços dos veículos (figura 29)

Esta consulta diz-nos a média de preços dos veículos do stand.

Figura 29 - Consulta da média de preços dos automóveis

The screenshot shows a database query interface. At the top, a SQL query is entered: `1 SELECT AVG(preco) AS media_preco FROM automovel;`. Below the query, a green status bar indicates: "A mostrar registos de 0 - 0 (1 total, A consulta demorou 0.0038 segundos.)". The query is repeated: `SELECT AVG(preco) AS media_preco FROM automovel;`. Below the query, there are links: "Perfil", "Editar em linha", "Editar", "Explicar SQL", "Criar código PHP", and "Actualizar". A control bar shows "Mostrar tudo" (unchecked), "Número de registos: 25", and "Filtrar registos: P". An "Opções extra" button is present. The result is displayed in a table with one column, **media_preco**, and one row with the value **8399.990000**.

Ordenar os proprietários por ordem alfabética (figura 30)

Devolve o nome dos proprietários por ordem alfabética.

Figura 30- Ordenar proprietários por ordem alfabética

The screenshot shows a database query interface. At the top, a SQL query is entered: `1 SELECT * FROM proprietario ORDER BY nome;`. Below the query, a green status bar indicates: "A mostrar registos de 0 - 4 (5 total, A consulta demorou 0.0004 segundos.)". The query is repeated: `SELECT nome FROM proprietario ORDER BY nome;`. Below the query, there are links: "Perfil", "Editar em linha", "Editar", "Explicar SQL", "Criar código PHP", and "Actualizar". A control bar shows "Mostrar tudo" (unchecked), "Número de registos: 25", and "Filtrar registos: P". An "Opções extra" button is present. The result is displayed in a table with one column, **nome**, and five rows. Each row has a checkbox, an "Editar" button, a "Copiar" button, and an "Apagar" button. The names are: Asdrubal Pinheiro, Caio Pinto, Jacinto Pinto, Marcelo Rebelo de Sousa, and Marta Maria.

Descobrir se tem um carro com o nome do proprietário começado pela letra 'A' (figura 31)

Figura 31 - Consulta a 2 tabelas

```
1 SELECT * FROM automovel a JOIN proprietario p ON a.automovel_id = p.proprietario_id WHERE p.nome LIKE 'A%';
```

✓ A mostrar registos de 0 - 0 (1 total, A consulta demorou 0.0005 segundos.)

`SELECT * FROM automovel a JOIN proprietario p ON a.automovel_id = p.proprietario_id WHERE p.nome LIKE 'A%';`

☐ Perfil [Editar em linha] [Editar] [Explicar SQL] [Criar código PHP] [Actualizar]

☐ Mostrar tudo | Número de registos: 25 | Filtrar registos:

Opções extra

automovel_id	ano_fabrico	cor	numero_serie	cilindrada	matricula	preco	modelo	Proprietario	proprietario_id	nome	morada	nif	email	conta
1	1998	Amarelo	681279137	1000	1	3999.99	1	4	1	Asdrubal Pinheiro	Rua do Quarto 22	234987456	asdrubalpinheiro@gmail.com	91564

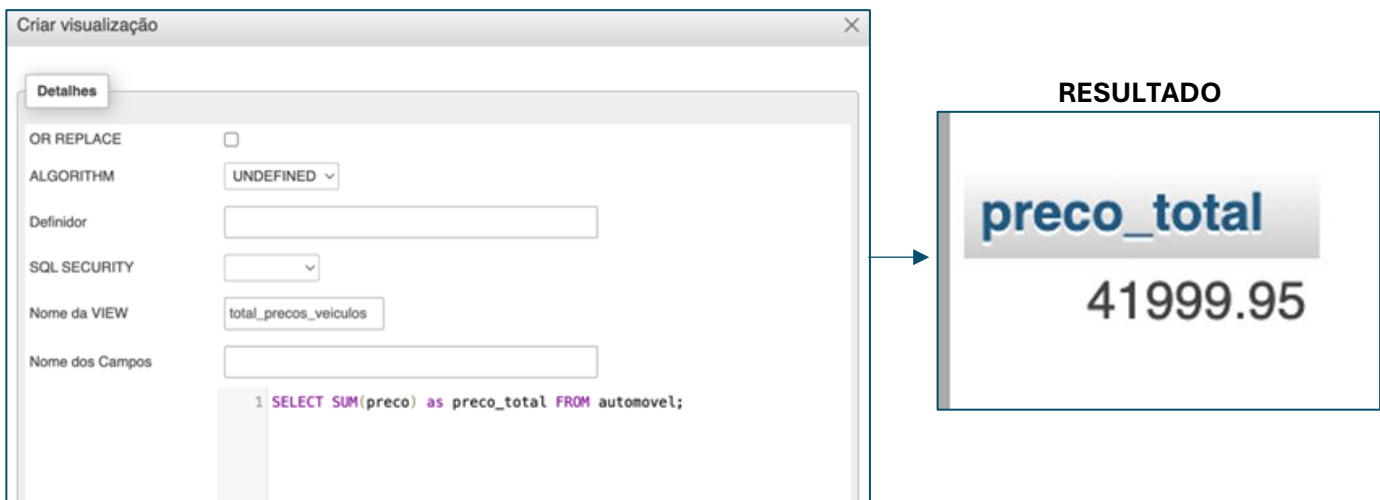
☐ Mostrar tudo | Número de registos: 25 | Filtrar registos:

CRIAÇÃO DE VIEWS

As *views* servem para simplificar consultas complexas, melhorando a gestão e a segurança dos dados. Permitem criar uma "janela" virtual para visualizar e consultar dados específicos sem alterar as tabelas originais. Assim, podemos aceder a informações importantes de forma rápida e organizada. Vamos utilizar o ambiente amigável do phpMyAdmin para criar a *view*.

VIEW 1 - consultar o preço total dos veículos (figura 32)

Figura 32 - Criação e Resultado da view 1



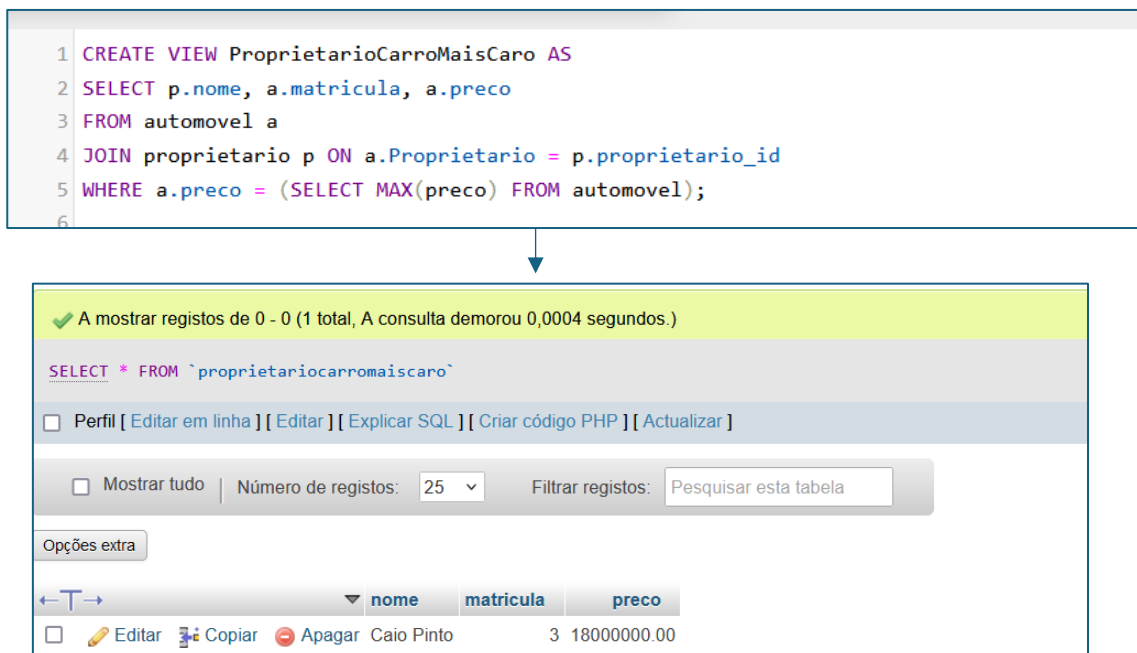
The screenshot shows the 'Criar visualização' (Create visualization) window in phpMyAdmin. The 'Detalhes' (Details) tab is active. The 'Nome da VIEW' (View name) is set to 'total_precos_veiculos'. The SQL query is: `1 SELECT SUM(preco) as preco_total FROM automovel;`. To the right, the 'RESULTADO' (Result) is displayed as a table with one row:

preco_total
41999.95

.

VIEW 2 - Proprietário que possui o automóvel mais caro (figura 33)

Figura 33 - SQL para criação da View e Resultado



The screenshot shows the SQL query for creating View 2:

```
1 CREATE VIEW ProprietarioCarroMaisCaro AS
2 SELECT p.nome, a.matricula, a.preco
3 FROM automovel a
4 JOIN proprietario p ON a.Proprietario = p.proprietario_id
5 WHERE a.preco = (SELECT MAX(preco) FROM automovel);
6
```

 Below the query, the result is displayed in a table with columns 'nome', 'matricula', and 'preco'. The result shows one row:

nome	matricula	preco
Caio Pinto	3	18000000.00

.

Esta *view* foi criada para fornecer uma visão rápida e eficiente sobre qual proprietário possui o automóvel de maior valor. Isso pode ser útil para diversas análises, como avaliar a concentração de valor na base de clientes ou identificar clientes prioritários para estratégias de marketing.

VIEW 3 - Automóveis por Proprietário (figura 34)

Esta *view* apresenta todos os automóveis e o seu respetivo proprietário.

Figura 34- View Automóveis e Proprietários

The screenshot shows a database management interface. The top panel displays the SQL command to create a view named 'AutomoveisPorProprietario'. The command is as follows:

```
1 CREATE VIEW AutomoveisPorProprietario AS
2 SELECT p.nome, a.matricula, a.preco
3 FROM automovel a
4 JOIN proprietario p ON a.Proprietario = p.proprietario_id;
```

The bottom panel shows the result of the query, displaying a table with 4 records. The table has columns for 'nome', 'matricula', and 'preco'. The records are:

	nome	matricula	preco
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Apagar	Marcelo Rebelo de Sousa	1	3999.99
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Apagar	Jacinto Pinto	2	19999.99
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Apagar	Caio Pinto	3	18000000.00
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Apagar	Marta Maria	4	5999.99

PROCEDIMENTOS

PROCEDIMENTO 1 – Atualizar a cor do automóvel (figura 35 e 36)

Permite a atualização da cor de um automóvel específico, identificado pelo seu número de série.

Figura 35 - Procedimento para atualizar a cor do veículo

```

4 CREATE PROCEDURE AtualizarCorAutomovel(
5     IN num_serie INT,
6     IN nova_cor VARCHAR(50)
7 )
8 BEGIN
9     DECLARE automovel_existente INT;
10
11     SET automovel_existente = (SELECT COUNT(*) FROM automovel WHERE numero_serie = num_serie);
12
13     IF automovel_existente = 0 THEN
14         SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Automóvel não encontrado.';
15     ELSE
16         UPDATE automovel
17         SET cor = nova_cor
18         WHERE numero_serie = num_serie;
19     END IF;
20
21
22

```

Vamos atualizar a cor de um veículo através do número de serie para '*marron*' para efeitos de teste (figura 36)

Figura 36- Resultado da aplicação do procedimento 1

```
SET @p0='681279137'; SET @p1='Marron'; CALL `AtualizarCorAutomovel`(@p0, @p1);
```

Resultados da execução da rotina 'AtualizarCorAutomovel'

	automovel_id	ano_fabrico	cor	numero_serie	cilindrada	matricula	preco	modelo	Proprietario
	1	1998	Marron	681279137	1000	1	3999.99	1	4
	2	2024	vermelho	751279137	1500	2	19999.99	2	5
	3	2001	vermelho	164278137	1200	3	18000000.00	3	3
	4	2012	vermelho	561279137	1500	4	5999.99	4	2
	5	2017	vermelho	891278137	1500	5	9999.99	5	NULL

☐ Marcar todos
 Com os seleccionados:
 Editar
 Copiar
 Apagar
 Exportar

A escolha de implementar este procedimento foi baseada na necessidade de permitir atualizações de dados de forma segura e controlada. Atualizações diretas nas tabelas podem ser propensas a erros e inconsistências, enquanto uma rotina armazenada

encapsula a lógica de negócio e garante que as verificações necessárias sejam realizadas antes de qualquer alteração.

PROCEDIMENTO 2 – Atualizar o preço de um automóvel (figura 37)

Permite a atualização do preço de um automóvel específico, identificado pelo seu ID.

Figura 37 - Procedimento para atualizar o preço de um automóvel

The diagram illustrates the process of updating a car's price using a stored procedure. It consists of four main steps connected by arrows:

- SQL Command Execution:** A screenshot of a database interface showing the execution of a SQL command to create a stored procedure named `AtualizarPrecoAutomovel`. The command is as follows:


```
1 DELIMITER //
2
3 CREATE PROCEDURE AtualizarPrecoAutomovel(
4     IN automovelId INT,
5     IN novoPreco DECIMAL(10,2)
6 )
7 BEGIN
8     UPDATE automovel
9     SET preco = novoPreco
10    WHERE automovel_id = automovelId;
11 END //
12
13 DELIMITER ;
```
- Procedure Execution:** A screenshot of the 'Rotinas' (Routines) section in the database interface. A dialog box titled 'Executar rotina `AtualizarPrecoAutomovel`' is shown, allowing the user to input parameters:

Nome	Tipo	Funções	Valor
automovelId	INT		3
novoPreco	DECIMAL		18000000

 Buttons for 'Continuar' (Continue) and 'Fechar' (Close) are visible at the bottom.
- Execution Result:** A screenshot of the database interface showing a success message:

✓ A sua consulta SQL foi executada com êxito.
0 linha afetada pela última instrução dentro do procedimento.

 Below the message, the executed SQL command is displayed:


```
SET @p0='3'; SET @p1='18000000'; CALL `AtualizarPrecoAutomovel`(@p0, @p1);
```
- Data Table Update:** A screenshot of the 'automovel' table in the database interface. The table has columns: `automovel_id`, `ano_fabrico`, `cor`, `numero_serie`, `cilindrada`, `matricula`, `preco`, `modelo`, and `Proprietario`. The data is as follows:

automovel_id	ano_fabrico	cor	numero_serie	cilindrada	matricula	preco	modelo	Proprietario
1	1998	Amarelo	681279137	1000	1	3999.99	1	4
2	2024	Preto	751279137	1500	2	10000.00	2	5
3	2001	Branco	164278137	1200	1	18000000.00	3	3

 The price for the car with ID 3 is highlighted with a red circle, showing the updated value of 18,000,000.00.

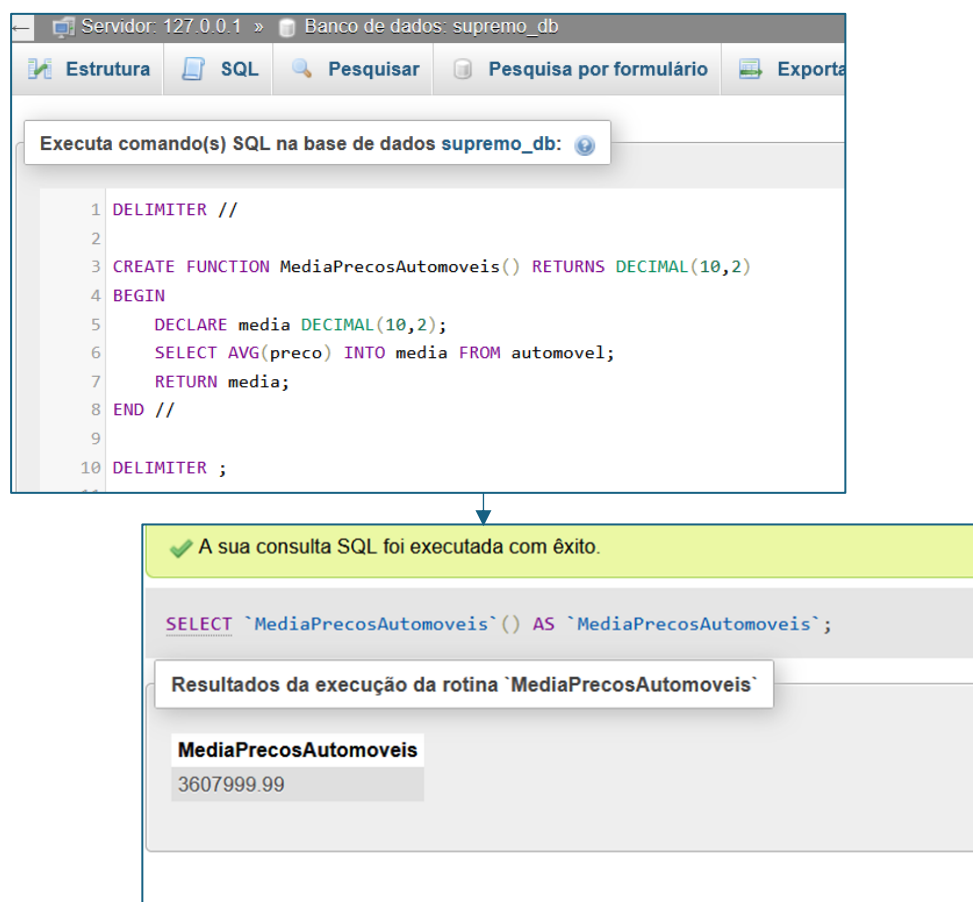
Verificamos que o valor foi corretamente atualizado. Este procedimento foi criado para facilitar a atualização do preço dos automóveis de maneira controlada. Ele garante que apenas os automóveis existentes sejam atualizados e previne erros comuns em operações de atualização direta.

FUNÇÕES

Função 1 – Calcular Média do Preço dos Automóveis (figura 38)

Devolve a média de preços de todos os automóveis na base de dados.

Figura 38 - Função para calcular a média do preço dos veículos



Esta função foi criada para fornecer uma forma rápida de obter a média de preços dos automóveis na base de dados. Funções desse tipo são úteis para relatórios e análises financeiras.

Função 2 –Calcular o total de automóveis por proprietário (figura 39 e 40)

Figura 39 - Obter o número de automóveis por proprietário

The figure consists of three screenshots illustrating the process of creating and executing a SQL function to count cars by owner.

Top Screenshot: A screenshot of a SQL editor window titled 'Servidor: 127.0.0.1 » Banco de dados: supremo_db'. The toolbar includes 'Estrutura', 'SQL', 'Pesquisar', 'Pesquisa por formulário', 'Exportar', 'Importar', and 'Operações'. The main area shows the following SQL code:

```

1 DELIMITER //
2
3 CREATE FUNCTION ContarAutomoveisPorProprietario(proprietarioId INT) RETURNS INT
4 BEGIN
5     DECLARE contador INT;
6     SELECT COUNT(*) INTO contador FROM automovel WHERE Proprietario = proprietarioId;
7     RETURN contador;
8 END //
9
10 DELIMITER ;
11

```

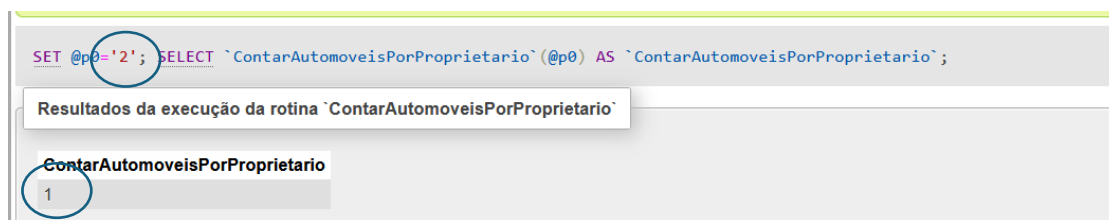
Middle Screenshot: A screenshot of a 'Rotinas' (Routines) window. A dialog box titled 'Executar rotina 'ContarAutomoveisPorProprietario'' is open. It shows the parameters for the function:

Nome	Tipo	Funções	Valor
proprietarioId	INT		1

Buttons 'Continuar' and 'Fechar' are at the bottom of the dialog.

Bottom Screenshot: A screenshot of the SQL editor showing the execution of the function. A green message bar at the top states: 'A sua consulta SQL foi executada com êxito.' Below it, the SQL command is shown: `SET @p0='1'; SELECT `ContarAutomoveisPorProprietario`(@p0) AS `ContarAutomoveisPorProprietario`;`. The results section, titled 'Resultados da execução da rotina 'ContarAutomoveisPorProprietario'', shows a single row with the value 0, which is circled in red.

Neste caso, vemos que apesar de o proprietário com ID =1 constar na tabela 'Proprietario', comprovamos que não possui nenhum carro no nosso stand. Contudo, o proprietário com ID=2 possui um carro no stand (figura 40)

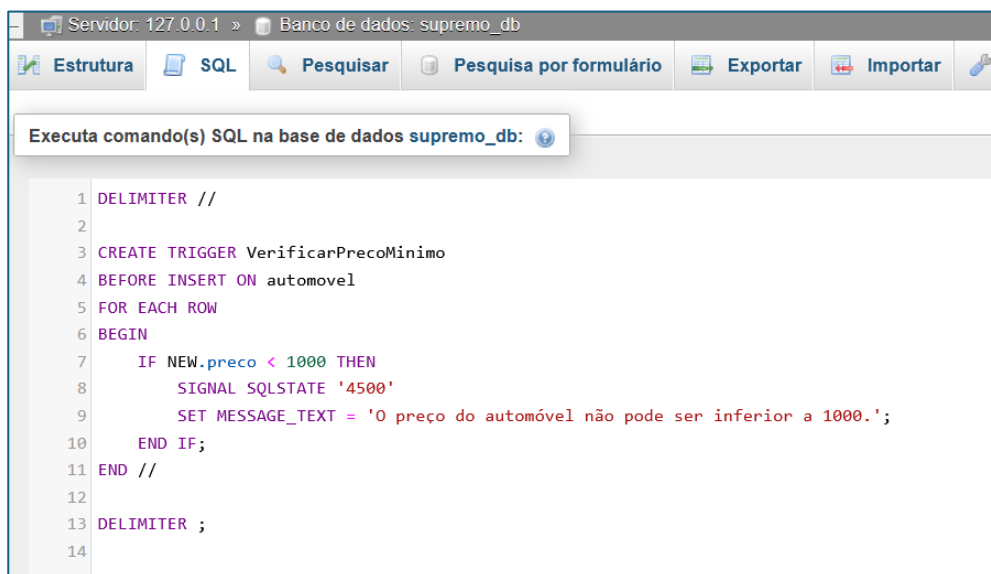
Figura 40 - Obter o número de automóveis por proprietário (continuação)

Esta função permite obter rapidamente o número de automóveis pertencentes a um determinado proprietário. É especialmente útil para gerar relatórios de propriedade e para análise de clientes.

TRIGGERS

Trigger 1

Este *trigger* impede a inserção de um preço de automóvel abaixo de 1000€ (figura 41).

Figura 41 - Trigger Validação de Preço para novos carros que sejam adicionados

Esses triggers garantem que o preço de um novo automóvel nunca seja definido abaixo de um valor mínimo (1 000€). Isso ajuda a manter a integridade dos dados. Quando tentamos inserir um veículo com preço abaixo do valor mínimo, o sistema não permite (figura 42).

Figura 42 - Teste trigger 1

The screenshot displays a database management interface with a form for inserting a new vehicle record. The form includes fields for 'ano_fabrico' (int(11)), 'cor' (varchar(50)), 'numero_serie' (int(11)), 'cilindrada' (int(11)), 'matricula' (int(11)), 'preco' (decimal(10,2)), 'modelo' (int(11)), and 'Proprietario' (int(11)). A modal error dialog is open, titled 'Erro', showing the SQL command and the MySQL error message: '#1644 - O preço do automóvel não pode ser inferior a 1000.' Below the error dialog, there are buttons for 'Pré-visualizar SQL', 'Reiniciar', and 'Continuar'. At the bottom, it indicates 'Continuar a inserção com 2 linhas'.

Trigger 2

Este trigger impede a redução do preço original de veículos existentes na base de dados. Apenas permite o aumento do preço do mesmo (figura 43)

Figura 43 - Trigger para impedir a redução do preço do veículo

The screenshot shows the 'Detalhes' (Details) configuration for a trigger named 'ImpedirReducaoPreco'. The trigger is configured to fire 'BEFORE' an 'UPDATE' event on the 'automovel' table. The trigger code is as follows:

```

1 BEGIN
2     IF NEW.preco < OLD.preco THEN
3         SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'O
preço do automóvel não pode ser reduzido.';
4     END IF;
5 END

```


Quando tentamos atualizar o preço de um carro para um valor inferior ao inicial, o sistema impede (figura 44)

Figura 44 - Teste trigger 2

Form fields:

- automovel_id: int(11) [dropdown] 4
- ano_fabrico: int(11) [dropdown] 2012
- cor: varchar(50) [dropdown] vermelho
- numero_serie: int(11) [text]
- cilindrada: int(11) [text]
- matricula: int(11) [text]
- preco: decimal(10,2) [text]
- modelo: int(11) [text]
- Proprietario: int(11) [text]

Modal dialog box:

Erro

Comando SQL: [Copiar](#) [Editar](#)

```
UPDATE `automovel` SET `preco` = '900' WHERE
```

Mensagens do MySQL: [🔗](#)

```
#1644 - O preço do automóvel não pode ser reduzido.
```

Form controls:

Guarda [dropdown] e então Voltar atrás [dropdown]

[🔗](#) [Pré-visualizar SQL](#) [Reiniciar](#) [Continuar](#)

Bottom modal dialog box:

Erro

Comando SQL: [Copiar](#) [Editar](#)

```
UPDATE `automovel` SET `preco` = '900' WHERE `automovel`.`automovel_id` = 4
```

Mensagens do MySQL: [🔗](#)

```
#1644 - O preço do automóvel não pode ser reduzido.
```

CONCLUSÃO

O presente trabalho proporcionou-nos uma compreensão prática e profunda sobre a criação e configuração de uma base de dados, desenvolvida no âmbito do trabalho final da unidade curricular de Armazenamento e Acesso a Dados. Desde a definição inicial dos requisitos até à implementação das estruturas de tabelas e relacionamentos, percorremos os passos essenciais para uma gestão eficiente de dados.

Concluímos que a criação e gestão adequada de uma base de dados SQL não só facilita as operações internas, mas também potencia a capacidade da empresa em funcionar de forma eficiente e personalizada. A escolha ambiente amigável do *phpMyAdmin* revelou-se crucial para garantir uma configuração precisa e segura, permitindo-nos aplicar conhecimentos teóricos de maneira prática e estratégica. Este trabalho foi além da aplicação de teorias, proporcionando uma visão prática sobre a importância estratégica das bases de dados. Destacamos que a gestão eficiente de uma base de dados é vital para o sucesso de qualquer organização.

Em jeito de conclusão, conseguimos cumprir os nossos objetivos iniciais e dar uma resposta eficaz às necessidades da *Supremo Automóveis*. A estrutura de base de dados desenvolvida permite uma gestão de dados robusta e adaptada-se às especificidades da empresa, proporcionando um suporte sólido para as suas operações e decisões estratégicas. Este projeto contribuiu positivamente para o nosso crescimento pessoal e profissional. Permitiu-nos delegar, estabelecer metas, prazos e a desenvolver confiança e sentido de compromisso para com o outro. Aspectos de carácter pessoal também são cruciais e tidos em conta pelas empresas nos dias que vivemos.

BIBLIOGRAFIA

Acedido a 12 de junho de 2024.

- <https://spaceprogrammer.com/bd/normalizando-um-banco-de-dados-por-meio-das-3-principais-formas/>

Acedido a 14 de junho de 2024.

- https://www.w3schools.com/MySQL/mysql_select.asp
- https://www.w3schools.com/MySQL/mysql_join_inner.asp
- https://www.w3schools.com/MySQL/mysql_count_avg_sum.asp

ANEXOS

ANEXO 1: FICHA DE IDENTIFICAÇÃO DO GRUPO



Edgar Rafael Stratulat de Sousa

28557



Renato Filipe Carneiro Azevedo

28549



Ricardo Emanuel Costa Rocha

28570