📄 Relatório Final – Projeto de Base de Dados: Aluguer de Veículos

1. Introdução

Este trabalho teve como objetivo o desenvolvimento de uma base de dados relacional para uma empresa fictícia de aluguer de veículos. Com base nos requisitos propostos, foi feita a modelação E-R, a conversão para esquema relacional, a normalização até à 3ª Forma Normal (3FN) e a implementação em SQL. O objetivo principal foi garantir uma estrutura de dados eficiente, sem redundâncias, com integridade referencial e capacidade de expansão futura.

---

2. Modelo Entidade-Relacionamento (E-R)

O diagrama E-R construído representa sete entidades principais:

- **Clientes**
- **Agencias**
- **Funcionarios**
- **Veiculos**
- **Alugueres**
- **Estados** (ligado aos veículos)
- **Categorias** (ligado aos veículos)

Cada entidade foi ligada pelas suas relações naturais:

- Um **Funcionário** está associado a uma **Agência**.
- Um **Veículo** pertence a uma **Agência** e está associado a um **Estado** e a uma **Categoria**.
- Um **Aluguer** liga um **Cliente**, um **Veículo** e um **Funcionário responsável**, com as respetivas datas e preço final.

O diagrama representa corretamente os **tipos de relacionamentos** (1:N e N:1) e as **chaves primárias e estrangeiras**.

---

3. Esquema Relacional

A conversão do diagrama E-R para o modelo relacional seguiu o formato correto. Cada entidade transformou-se numa tabela com uma chave primária (`PK`) e, quando necessário, chaves estrangeiras (`FK`):

- `Clientes(Cliente_ID PK, Nome, NIF, Morada, Telefone, Email)`
- `Agencia(Agencia_ID PK, Nome, Localizacao, Contacto)`
- `Funcionarios(Funcionario_ID PK, Nome, Telefone, Email, Agencia_ID FK)`
- `Estados(Estado_ID PK, Designacao)`
- `Categorias(Categoria_ID PK, Designacao)`
- `Veiculos(Veiculo_ID PK, ..., Estado_ID FK, Categoria_ID FK, Agencia_ID FK)`
- `Alugueres(Aluguer_ID PK, DataInicio, DataFim, PrecoTotal, Cliente_ID FK, Veiculo_ID FK, Funcionario_ID FK)`

Todos os relacionamentos foram corretamente implementados através de `FOREIGN KEY`, garantindo a integridade referencial entre entidades.

---

4. Normalização

Foi realizada a análise das três formas normais:

- **1FN (Primeira Forma Normal):** Todos os atributos são atómicos e não existem listas, campos compostos ou grupos repetidos. O problema dos atributos multivalorados em `Estado` e `Categoria` foi resolvido através da criação das tabelas `Estados` e `Categorias`.
- **2FN (Segunda Forma Normal):** Todas as tabelas têm chave primária simples, portanto não existem dependências parciais. Cada atributo não-chave depende diretamente da chave primária.
- **3FN (Terceira Forma Normal):** Verificou-se que não existem dependências transitivas. Por exemplo, `Marca`, `Modelo`, `Ano` e `Preco` em `Veiculos` dependem diretamente de `Veiculo_ID`, e `Categoria`/`Estado` estão normalizados em tabelas

separadas.

Resultado: **todas as tabelas estão corretamente normalizadas até à 3FN**.

---

5. Implementação SQL

Foi criado um **script SQL completo** que inclui:

- Criação da base de dados: `CREATE DATABASE AluguerVeiculos;`
- Criação de tabelas com restrições de integridade (`PRIMARY KEY`, `FOREIGN KEY`)
- Inserção de dados de exemplo para simular um cenário real
- Os campos obrigatórios estão bem definidos (`NOT NULL` em campos essenciais como nomes e NIF)
- Todas as relações foram validadas com sucesso (ex. associação correta de `Funcionario_ID` na tabela `Alugueres`)

---

6. Validação e Testes

O script SQL foi testado com sucesso. A base de dados cria todas as tabelas sem erros e os dados de exemplo são inseridos corretamente. A integridade referencial está assegurada – por exemplo, não é possível registar um aluguer com um veículo ou funcionário inexistente.

---

7. Potenciais Melhorias Futuras

Embora a base de dados esteja funcional e completa, há sempre oportunidades para melhorias futuras:

- **Tabela de pagamentos:** para registar o estado do pagamento (pago, pendente, modo de pagamento).
- **Histórico de manutenção de veículos:** incluir uma tabela `Manutencoes(Veiculo_ID, Data, Tipo, Custo)` para registar manutenções.
- **Login para funcionários/clientes:** permitir autenticação e controlo de acessos (ideal para uma aplicação web associada).

- **Melhor gestão de disponibilidade:** permitir calcular automaticamente a disponibilidade de veículos com base nas datas de aluguer.

---

8. Conclusão

O projeto cumpriu integralmente os objetivos propostos. Foi feita uma modelação sólida com base na análise do problema, e todas as regras de normalização foram corretamente aplicadas. O resultado é uma base de dados eficiente, funcional e preparada para crescer em futuras versões. Este exercício foi fundamental para aplicar conhecimentos de modelação, SQL e boas práticas de desenvolvimento de bases de dados relacionais.