

Bases de Dados - MotoShop

Departamento de Eletrónica, Telecomunicações e
Informática
Universidade de Aveiro

Fábio Alves 84794, Marcelo Fraga 84839
fabioalves98@ua.pt, fraga16@ua.pt



Conteúdo

1	Introdução	1
2	Diagrama Entidade-Relação	2
2.1	Alterações	3
2.1.1	Esquema Relacional	4
3	Implementação	5
3.1	Base de Dados	5
3.1.1	Views	5
3.1.2	Stored Procedures	5
3.1.3	User-Defined Functions	6
3.1.4	Triggers	6
3.2	Aplicação	6
4	Conclusão	8

Capítulo 1

Introdução

Surge, no âmbito da disciplina de Bases de Dados, a criação de uma Base de Dados e seu respetivo modelo para a gestão de informação de uma loja de motos. Neste caso particular, tem-se como objetivo criar também uma aplicação de interação com a respetiva BD onde deverá ser possível manipular e visualizar dados de forma simples e rápida.

Tendo em conta alguns requisitos previamente retirados acerca das necessidades e capacidades do modelo de dados, efetuaram-se os primeiros diagramas relativos às relações entre as entidades existentes no sistema.

Seria necessário gerir não só qualquer mota envolvida em todos os processos da loja, mas também todos os clientes e staff da mesma. Seria ainda necessário também administrar os diferentes stands e oficinas, e por fim emitir novas vendas e revisões para o sistema.

Capítulo 2

Diagrama Entidade-Relação

Esclarecido assim o contexto principal do nosso problema, bem como o método de trabalho a adotar para a resolução do mesmo, mostramos agora as principais relações entre as diversas entidades presentes no DER.

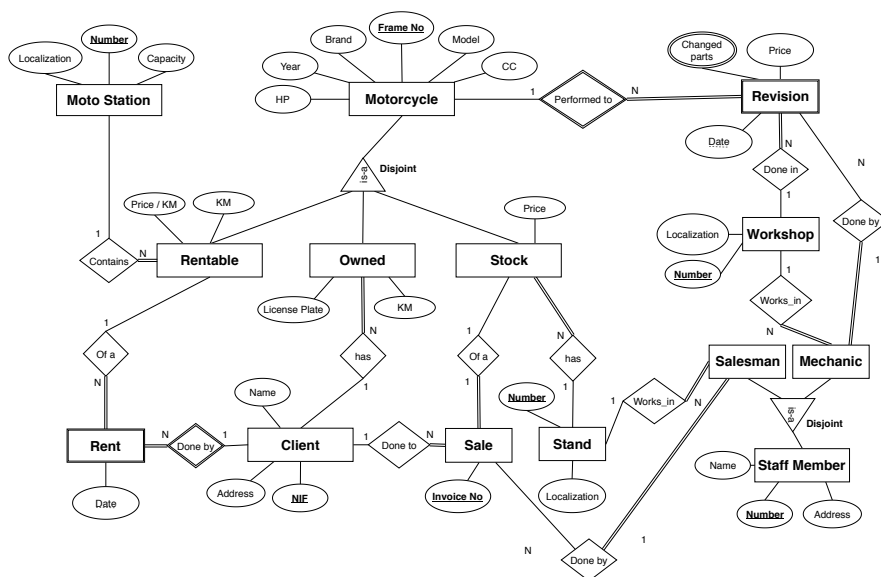


Figura 2.1: Diagrama Entidade-Relação

A nível das Entidades, podemos considerar a entidade **Motorcycle** como uma das mais importantes do sistema. Esta classe está presente no topo de uma hierarquia que tem como principal objetivo dividir as motos do sistema em três tipos distintos: **Rentable**, **Owned** e **Stock** cujos nomes tornam per-

cetível aquilo que elas tentam representar. Neste caso, as motos **Rentable** e **Stock** pertencem à loja estando disponíveis em Moto Stations ou em Stands, respetivamente. Já as **Owned** representam motos compradas por um cliente.

Do lado contrário do diagrama podemos observar não só mais uma relação hierárquica que, neste caso, representa os diferentes tipos de **Owned** existentes no sistema, mas também o último serviço disponibilizado pelas lojas MotoShop, as revisões.

2.1 Alterações

O DER acima descrito representa uma versão mais atualizada, em comparação com o primeiro DER desenvolvido, daquilo que o modelo da Base de Dados se tornou. Um exemplo de uma alteração efetuada ao DER diretamente, é o facto de existir uma ligação mais direta entre o membro de Staff vendedor e a venda propriamente dita, permitindo assim identificar o vendedor de uma moto na própria venda. Este mesmo problema acontece igualmente no caso das revisões. Para podermos indicar qual o mecânico que realizou uma certa revisão, necessitamos de criar uma relação direta entre ambas estas duas entidades: **Revision** e **Mechanic**.

Outras alterações mais subtis passam pela alteração de como as chaves primárias funcionam em algumas das entidades do diagrama. Por exemplo, no caso das revisões onde usávamos a data da revisão e o número do quadro da moto (Frame_No) como chave primária conjunta da entidade, decidiu-se fazer a troca para uma chave primária auto-incrementada única da revisão, tornando-a uma entidade normal em vez de uma entidade fraca, como é representada.

Finalmente, e falando um pouco nas alterações mais concretas no modelo da base de dados, foi adicionada uma nova tabela **Part** com o seguinte formato:

Part

<u>Part_No</u>	Part_Name
----------------	-----------

Figura 2.2: Tabela "Part"

Esta adição permite adicionar nomes às partes modificadas nas revisões, visto que previamente, a tabela **Changed Parts** que surgia de um atributo multi-valor da entidade **Revision**, continha apenas um pseudo-número de uma peça alterada.

2.1.1 Esquema Relacional

O esquema relacional obtido no final da criação de tabelas na base de dados é o seguinte:

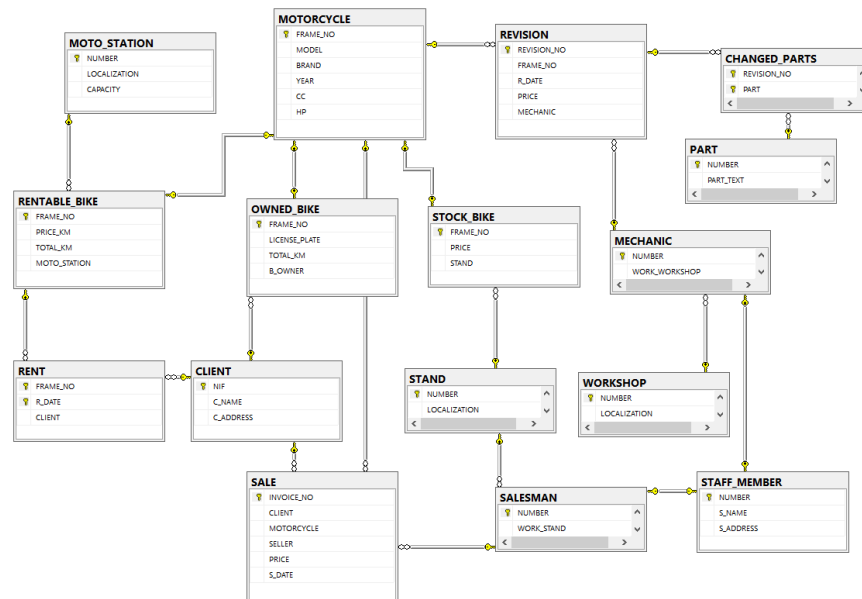


Figura 2.3: Esquema relacional

Capítulo 3

Implementação

Passando agora para a implementação propriamente dita, e explicando todas as ferramentas utilizadas para o desenvolvimento da Base de Dados, que passa pela utilização de features como as Views, Stored Procedures ou Triggers, iremos explicar a sua aplicação em alguns dos casos mais específicos do sistema.

3.1 Base de Dados

3.1.1 Views

No caso das Views, estas são utilizadas como um primeiro nível de abstração ao modelo de dados. Por exemplo, no caso das motos que são representadas, tal como explicado anteriormente, através de uma relação de herança, onde a classe pai tem todos os atributos base, e as entidades filho têm as restantes propriedades individuais de cada tipo, as views tornam possível fazer a junção de ambas as tabelas da entidade pai e filho para podermos aceder à informação total da moto sem ter de juntar constantemente ambas as tabelas em cada query efetuada.

3.1.2 Stored Procedures

Em relação às Stored Procedures, estas são usadas para parametrizar e abstrair todas as operações de escrita - Insert ou Update - bem como operações de remoção de dados. Para além de algumas verificações na inserção ou update de alguns dados, como por exemplo no update de o número total de quilómetros de uma moto não poder ser inferior ao valor anteriormente presente na entrada do tuplo a atualizar, as stored procedures ajudam também a melhorar a performance da própria base de dados.

3.1.3 User-Defined Functions

Em contraste com as Stored Procedures, as UDFs presentes no sistema final são utilizadas apenas para a leitura de dados da Base de Dados visto terem o benefício de poderem ser utilizadas em sub-queries. Neste caso em particular, são utilizadas para parametrizar algumas leituras, por exemplo obter as revisões feitas por um mecânico em específico, ou as motas presentes num determinado stand. Finalmente têm também o propósito de criar abstrações concretas a nível de dados estatísticos em algumas tabelas. Por exemplo, para obter a mota com mais arrendamentos efetuados, ou o vendedor com maior número de vendas.

3.1.4 Triggers

Por fim, os Triggers existentes no modelo final permitem garantir a propriedade de Disjoint nas relações hierárquicas presentes na base de dados. Além disso, existem outros como por exemplo o trigger - COMPLETE_SALE - que atua como um finalizador de uma venda de um motociclo, removendo a entrada da mota vendida da tabela STOCK_BIKE, e colocando essa mesma mota na tabela de motas usadas com a respetiva entrada do dono correspondente ao comprador.

3.2 Aplicação

Em relação à aplicação, e tal como já mencionado anteriormente, esta é utilizada não só como uma maneira de visualizar e alterar dados da base de dados mas como um ponto de venda de uma loja de motas. A ideia principal é esta servir não só como interface de gestão para com a base de dados mas também com os serviços da própria loja.

Assim sendo, podemos contar com features como a gestão de staff, clientes e motas bem como a administração de stands e oficinas(Fig. 3.1). É possível também realizar a emissão de novas vendas, revisões e arrendamentos através desta mesma aplicação(Fig. 3.2).

Store

Motorcycles

Staff

Clients

Logged in as: Duarte Pinto

NIF	NAME	ADDRESS
104260343	Inês Silva	Calçada do Poço dos ...
129486690	Francisca Rodrigues	Avenida Resano Garc...
200498521	Tomás Almeida	Rua da Madalena, N.º 2...
302421035	Tomás Fernandes	Rua dos Fanqueiros, N.º...
333808024	Rodrigo Nunes	Rua de São João, N.º...
399567190	Diogo Nunes	Avenida Resano Garc...
430803428	Diogo Pinto	Rua dos Bacalhoeiros, ...
46285303	Ana Pinto	Rua dos Bacalhoeiros, ...
614213351	Afonso Rodrigues	Rua Roberto Duarte Sil...
918985398	João Almeida	Calçada Marquês de A...

Remove

Motorcycles

Motorcycle	License_plate	Total_km
Ducati MX	175582	0
Yamaha Multistrada	600309	42261
BMW XT	108371	19090
Kawasaki RX	334954	43426

Revisions

Number	Revision_Date	Motorcycle	Part
1004	22/10/2018	Yamaha Mult...	Pneus
1008	15/03/2018	Ducati MX	Pastilhas travão
1010	14/07/2018	BMW XT	Pneus
1013	11/08/2018	Kawasaki RX	

Rents

Rent_Date	Motorcycle
03/03/2018	Yamaha Resano
26/12/2018	Triumph RS
10/06/2018	Kawasaki XT

Insert new client

Client

NIF

Name

Address

Insert

Figura 3.1: Gestão de clientes

MotoShop

Store

Motorcycles

Staff

Clients

Logged in as: Francisca Oliveira

Sales

Invoice	Date	Motorcycle
1000	08-Jun-15	Yamaha TRK
1001	15-Apr-17	Ducati Titan
1002	04-Feb-17	Benelli T-Max
1003	23-Jun-16	Kawasaki X-Max
1004	04-Oct-17	Suzuki SSR
1005	12-Jun-16	Ducati Star

Add Remove

Details

Motorcycle

CC

HP

Yamaha TRK

750

82

Client

Martin Gonçalves

Seller

Francisca Gonçalves

Stand

Cerinas

Price

18966

Statistics

Best Salesman -

Francisca Gonçalves

Most Expensive Motorcycle -

Ducati T-Max

Revisions

Number	Date	Motorcycle
1000	16-Jan-16	Honda Turbo
1001	04-Apr-15	Yamaha TRK
1002	04-Jul-15	Ducati Titan
1003	16-Sep-15	Ducati SX
1004	16-Apr-16	Benelli T-Max
1005	13-Mar-17	Kawasaki X-Max
1006	10-Jun-18	BMW GS
1007	28-Oct-16	Suzuki SSR
1008	15-May-18	Ducati Star

Add Remove

Details

Motorcycle

CC

HP

BMW GS

200

28

Owner

Martin Martins

Mechanic

Martin Martins

Workshop

Lounhã

Price

353

NP Parts

2

Statistics

Best Mechanic -

João Ribeiro

Average Number of Parts -

1.55

Rents

Date	Motorcycle
06-Mar-16	Yamaha MT
13-May-15	Honda Turbo
20-Oct-17	Ducati SX
23-Feb-15	Ducati MT
11-Feb-17	Ducati MT
04-Jun-18	Suzuki Bolt
08-Sep-18	Suzuki Bolt
13-Dec-15	Harley RC
13-Feb-16	Harley RC

Add Remove

Details

Motorcycle

CC

HP

Ducati SX

1200

128

Client

Inês Jesus

MechStation

Sobral de Monte Agrajo

Price per KM

7

Statistics

Most Active Client -

Rodrigo Oliveira

Most Rented Motorcycle -

Harley RC

Figura 3.2: POS

Capítulo 4

Conclusão

No decorrer do projeto vários desafios surgiram. Sendo esta a primeira vez a trabalhar com um modelo de dados maior e com mais rigor, podemos dizer que a versão final do sistema se adequa àquilo que havia sido projetado nas primeiras iterações do projeto.

Os requisitos foram cumpridos e grande parte do conteúdo lecionado durante o semestre foi estudado e adaptado ao nosso problema em particular.

Assim, é de valor referir que apesar do produto final estar de acordo com as especificações iniciais, existem ainda alguns detalhes que seriam importantes completar e que não foram finalizados no âmbito deste projeto. Seriam estes, por exemplo, um conjunto mais abrangente de queries e udf's ou stored procedures ou até mesmo a variedade destas.