Fundamentos de Base de Dados, 2020

1ª Fase do Projecto

Índice

1 – Introdução

2 – Justificativa

3 – Objectiva

4 – Desenvolvimento

4.1 – Definição

4.2 – Fases do processo de construção da base de dados

4.2.1 – Análise de requisitos

4.2.2 – Modelo conceitual

4.2.3 – Modelo lógico

4.2.4 – Restrição de integridade

4.2.5 – Modelo físico

4.3 – Scripts

4.3.1 – SQL - DDL

4.3.2 – SQL – DML

5 – Conclusão

6 - Bibliografia

Introdução

Este trabalho insere-se na senda de modelagem e implementação de uma base de dados, que contempla a estrutura organizacional de uma instituição, neste caso, a centralidade DEV-MOD.

Justificativa

As organizações públicas e privadas começaram a perceber o valor dos dados que têm em sua disposição, e a considera-los um bem importante no aumento de produtividade, eficiência e competitividade. Como consequência, exploração de enormes volumes de dados assumem papéis cada vez mais importante.

Portanto, esforços realizados para um gerenciamento eficiente das informações são cada vez mais justificados, principalmente em casos que, as informações devem estar cada vez mais integradas fornecendo assim um apoio eficiente na tomada de decisões.

Levando em consideração todo esses parâmetros, é que surge esta pesquisa no sentido de propor um modelo de projecto e implementação para um tipo de base de dados, neste caso o Oracle.

O Oracle é um Sistema Gerenciador de Banco de Dados (SGBD), sendo ele mais utilizados em aplicações corporativas. Robusto, confiável e seguro, estatísticas aponta como sendo o SGBD mais utilizado mundialmente.

Objectiva

O objectivo inicial é propor um modelo, implementação de uma base de dados confiável que destaca a eficiência e produtividade, desacelerando, consequentemente, o processo burocráticos nas suas diversas facetas.

Desenvolvimento

Os sistemas de base de dados são projectados para administrar grandes quantidades de informações, sem redundâncias, sobre uma determinada informação, criando assim um ambiente adequado eficiente para armazenar e recuperar as informações.

Quatros fases foram seguidas no processo de construção da base de dados em questão, sendo elas:

* Análise de requisitos
* Construção do modelo conceitual
* Construção do modelo lógico
* Construção do modelo físico

Análise de requisitos

Considerando o universo do discurso elaborado pelo Administrador, conseguiu-se apurar todos os dados necessários na perspectiva do usuário.

Construção do modelo conceitual

Levando em conta cada ponto abordado na 1ª fase, obteve-se o seguinte diagrama conceitual:

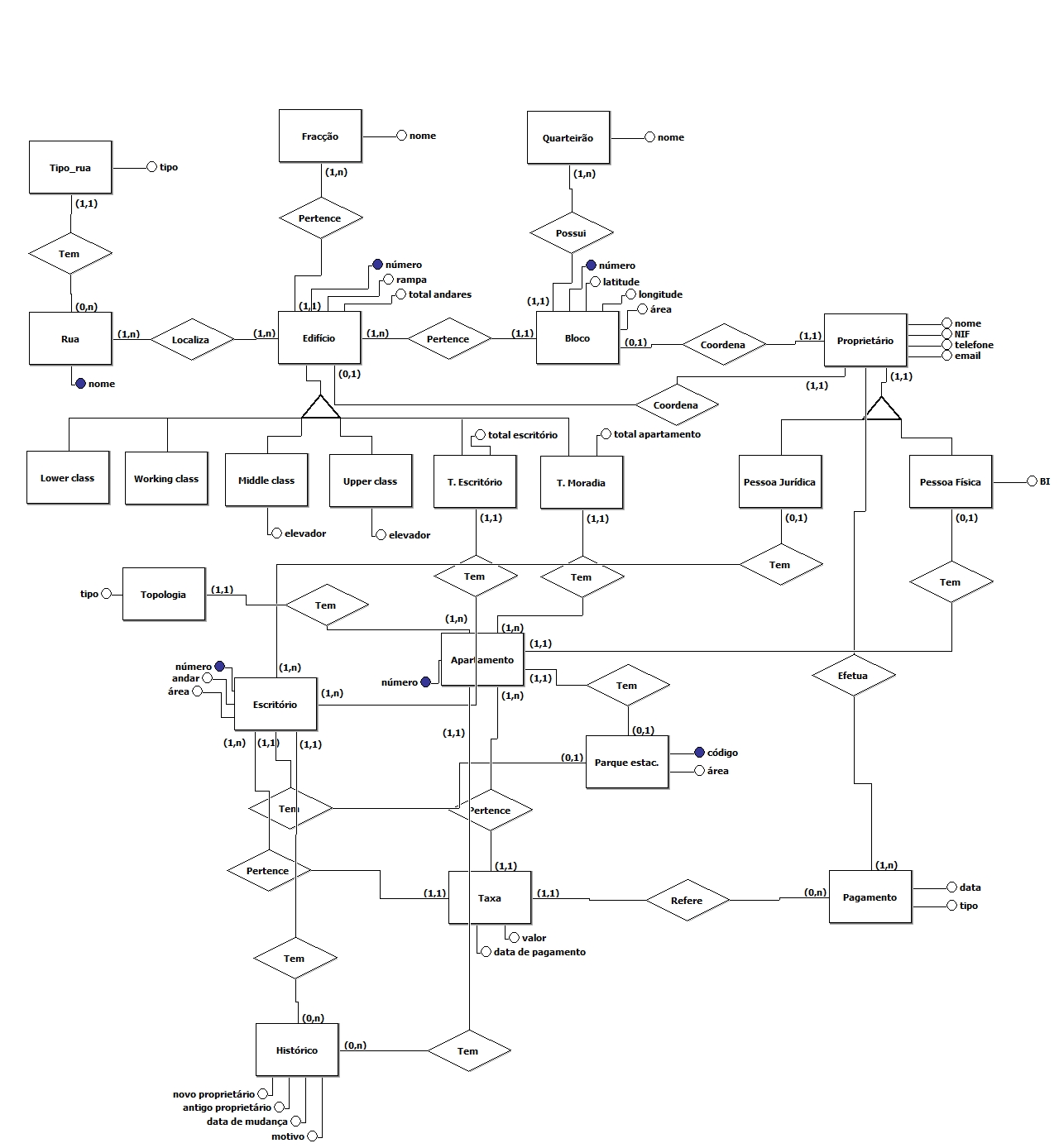


Foto 1. Modelo conceitual

Construção do modelo lógico

Proprietário (**NIF**, nome, email, telefone)

Bloco (**num\_bloco**, latitude, longitude, area, **NIF\_proprietario**)

Quarteirao (designacao, **num\_bloco**)

Edificio (**num\_edificio**, is\_rampa, total\_andares, **num\_bloco,** modelo, utilidade, **NIF\_proprietario**)

Fraccao (**cod\_fraccao**, designacao, **num\_edifício**)

Rua (**cod\_rua**, nome )

Tipo\_Rua (**cod\_tipo\_rua**, tipo)

Rua\_Tipo\_Rua (**cod\_rua\_tipo\_rua , cod\_rua**, **cod\_tipo\_rua,**)

Edifício\_Rua (**num\_edifício**, **cod\_rua, cod\_edificio\_rua** )

Lower\_Class (**cod\_lower\_class**, **num\_edificio**)

Working\_Class (**cod\_working\_class**, **num\_edificio**)

Middle\_Class (**cod\_middle\_class**, is\_elevador, **num\_edificio**)

Upper\_Class (**cod\_upper\_class**, is\_elevador, **num\_edificio**)

Tipo\_Moradia (**cod\_tipo\_moradia**, total\_apartamentos, **num\_edificio**)

Tipo\_Escritório (**cod\_tipo\_escritorio**, total\_escritórios, **num\_edificio**)

Topologia (**cod\_topologia**, tipo)

Taxa (**cod\_taxa**, valor, data\_pagamento)

Apartamento (**num\_apartamento**, num\_andar, cod\_topologia, cod\_tipo\_moradia, cod\_taxa)

Parque\_Estacionamento (**cod\_parque\_estacionamento**, area, **num\_apartamento**, **num\_edificio**)

Pessoa\_Fisica ( **cod\_pessoa\_fisica**, BI, **NIF\_proprietario**, **num\_ apartamento**, **num\_andar**)

Pessoa\_Juridica (**cod\_pessoa\_juridica**, **NIF\_proprietario**)

Escritorio (**cod\_escritorio**, andar, area, num\_escritorio, **cod\_proprietario\_juridica**, **cod\_taxa**, **cod\_tipo\_escritorio**)

Pagamento (**cod\_pagamento**, tipo, data\_pagamento, **cod\_proprietario**)

Pagamento-taxa (**cod\_pagamento\_taxa**, **cod\_pagamento**, **cod\_taxa**)

Historico (**cod\_historico, cod\_novo\_proprietario**, **cod\_antigo\_proprietario**, data\_mudanca, motivo)

Escritorio\_Historico (**cod\_escritorio\_historico, cod\_escritorio, cod\_historico**)

Apartamento\_Historico (**cod\_apartamento\_historico,cod\_apartamento, cod\_historico**)

Dependendo do relacionamento, podemos afirmar que os atributos contidos nas entidades envolvidas podem gerar outras relações ou tabelas conforme é visto no modelo lógico do nosso projecto.

Restrições de Integridade

As restrições de integridades, são condições que devem ser verificadas para que um elemento do produto cartesiano dos domínios dos atributos possa pertencer à relação.

Temos várias definições acerca de integridade mas, decidimos escolheres algumas para podermos explicar.

Integridade de Domínio: é a integridade que é imposta pelo domínio de cada atributo em cada relação.

Integridade de Referência: Diz que o valor de qualquer chave estrangeira de uma relação é um valor da chave primária que se refere.

Integridade de Entidade: Diz que não existem duas instâncias de uma relação cujas as chaves tenham o mesmo valor e, o valor de uma chave primária nunca deve ser null

Construção do modelo físico

O Modelo Físico irá partir do Modelo Lógico e descreve as estruturas físicas do armazenamento de dados, tais como: tamanho de campos, índices, tipo de preenchimento destes campos, nomenclaturas, etc.

Script

SQL-DDL (Data Definition Language) - conjunto de comandos que lidam com os objectos permitindo assim a implementação da base de dados.

Em anexo encontram-se os scripts usados para a criação das tabelas:

create table proprietario(

nome\_proprietario varchar2(100) not null,

nif varchar2(14),

email varchar2(100),

telefone varchar2(9) not null,

constraint pk\_proprietario primary key(nif)

)

create table bloco(

num\_bloco number(3),

latitude number(7,3) not null,

longitude number(7,3)not null,

area number(7,3) not null,

nif\_coordenador varchar2(14),

foreign key(nif\_coordenador) references proprietario(nif),

constraint pk\_bloco primary key(num\_bloco)

)

create table quarteirao(

designacao varchar2(5) not null,

num\_bloco number(3),

foreign key(num\_bloco) references bloco(num\_bloco),

constraint pk\_quarteirao primary key(designacao)

)

create table edificio(

num\_edificio number(5),

total\_andares number(3) not null,

modelo varchar2(20) not null check (modelo = 'lower-class' or modelo = 'working-class' or modelo = 'middle-class' or modelo = 'upper-class'),

utilidade varchar2(20) not null check (utilidade = 'moradia' or utilidade = 'escritorio'),

is\_rampa number(1) not null check (is\_rampa = 1 or is\_rampa = 0),

nif\_coordenador varchar2(14),

num\_bloco number(3),

foreign key(num\_bloco) references bloco(num\_bloco),

foreign key(nif\_coordenador) references proprietario(nif),

constraint pk\_edificio primary key(num\_edificio)

)

create table fraccao(

cod\_fraccao number(3),

designacao varchar2(1),

num\_edificio number(5),

foreign key(num\_edificio) references edificio(num\_edificio),

constraint pk\_fraccao primary key(cod\_fraccao))

create table rua(

cod\_rua number(3),

nome varchar2(30) not null unique,

constraint pk\_rua primary key(cod\_rua)

)

create table tipo\_rua(

cod\_tipo\_rua number(3),

tipo varchar2(15) not null check (tipo = 'principal' or tipo = 'secundaria'),

constraint pk\_tipo\_rua primary key(cod\_tipo\_rua)

)

create table rua\_tipo\_rua(

cod\_rua\_tipo\_rua number(3),

cod\_rua number(3),

cod\_tipo\_rua number(3),

foreign key(cod\_rua) references rua(cod\_rua),

foreign key(cod\_tipo\_rua) references tipo\_rua(cod\_tipo\_rua),

constraint pk\_rua\_tipo\_rua primary key(cod\_rua\_tipo\_rua)

)

create table edificio\_rua(

cod\_edificio\_rua number(3),

cod\_rua number(3),

num\_edificio number(5),

foreign key(num\_edificio) references edificio(num\_edificio),

foreign key(cod\_rua) references rua(cod\_rua),

constraint pk\_edificio\_rua primary key(cod\_edificio\_rua)

)

create table lower\_class(

cod\_lower\_class number(3),

num\_edificio number(5),

foreign key(num\_edificio) references edificio(num\_edificio),

constraint pk\_lower\_class primary key(cod\_lower\_class)

)

create table working\_class(

cod\_working\_class number(3),

num\_edificio number(5),

foreign key(num\_edificio) references edificio(num\_edificio),

constraint pk\_working\_class primary key(cod\_working\_class)

)

create table middle\_class(

cod\_middle\_class number(3),

is\_elevador number(1) not null check (is\_elevador = 1 or is\_elevador = 0),

num\_edificio number(5),

foreign key(num\_edificio) references edificio(num\_edificio),

constraint pk\_middle\_class primary key(cod\_middle\_class)

)

create table upper\_class(

cod\_upper\_class number(3),

is\_elevador number(1) not null check (is\_elevador = 1 or is\_elevador = 0),

num\_edificio number(5),

foreign key(num\_edificio) references edificio(num\_edificio),

constraint pk\_upper\_class primary key(cod\_upper\_class)

)

create table tipo\_moradia(

cod\_tipo\_moradia number(3),

total\_apartamentos number(3) not null,

num\_edificio number(5),

foreign key(num\_edificio) references edificio(num\_edificio),

constraint pk\_tipo\_moradia primary key(cod\_tipo\_moradia)

)

create table tipo\_escritorio(

cod\_tipo\_escritorio number(3),

total\_escritorios number(3) not null,

num\_edificio number(5),

foreign key(num\_edificio) references edificio(num\_edificio),

constraint pk\_tipo\_escritorio primary key(cod\_tipo\_escritorio)

)

create table topologia(

cod\_topologia number(3),

tipo varchar2(5) not null check (tipo = 't2' or tipo = 't3+1' or tipo = 't4' or tipo = 't5'),

constraint pk\_topologia primary key(cod\_topologia)

)

create table taxa(

cod\_taxa number(3),

valor number(10,2) not null,

data\_pagamento varchar2(6) not null check (tipo = 'mensal' or tipo = 'anual'),

constraint pk\_taxa primary key(cod\_taxa)

)

create table apartamento(

num\_apartamento number(3),

num\_andar number(3) not null,

cod\_taxa number(3),

cod\_topologia number(3),

cod\_tipo\_moradia number(3),

foreign key(cod\_topologia) references topologia(cod\_topologia),

foreign key(cod\_taxa) references taxa(cod\_taxa),

foreign key(cod\_tipo\_moradia) references tipo\_moradia(cod\_tipo\_moradia),

constraint pk\_apartamento primary key(num\_apartamento, num\_andar)

)

create table parque\_estacionamento(

cod\_parque\_estacionamento number(3),

area number(7,2) not null,

num\_apartamento number(3) not null,

num\_edificio number(5),

foreign key(num\_edificio) references edificio(num\_edificio),

constraint pk\_parque\_estacionamento primary key(num\_edificio, num\_apartamento, cod\_parque\_estacionamento))

create table pessoa\_fisica(

cod\_pessoa\_fisica number(3),

bi varchar2(14) not null unique,

num\_apartamento number(3),

num\_andar number(3),

cod\_proprietario varchar2(14),

foreign key(cod\_proprietario) references proprietario(nif),

foreign key(num\_apartamento, num\_andar) references apartamento(num\_apartamento, num\_andar),

constraint pk\_pessoa\_fisica primary key(cod\_pessoa\_fisica)

)

create table pessoa\_juridica(

cod\_pessoa\_juridica number(3),

cod\_proprietario varchar2(14),

foreign key(cod\_proprietario) references proprietario(nif),

constraint pk\_pessoa\_juridica primary key(cod\_pessoa\_juridica)

)

create table escritorio(

cod\_escritorio number(3),

num\_escritorio number(3) not null,

andar number(3) not null,

area number(7,5) not null,

cod\_taxa number(3),

cod\_tipo\_escritorio number(3),

foreign key(cod\_taxa) references taxa(cod\_taxa),

cod\_proprietario\_juridica number(3),

foreign key(cod\_tipo\_escritorio) references tipo\_escritorio(cod\_tipo\_escritorio),

foreign key(cod\_proprietario\_juridica) references pessoa\_juridica(cod\_pessoa\_juridica),

constraint pk\_escritorio primary key(cod\_escritorio)

)

create table pagamento(

cod\_pagamento number(3),

tipo varchar2(6) not null check (tipo = 'mensal' or tipo = 'anual'),

data\_pagamento date not null,

cod\_proprietario varchar2(14),

foreign key(cod\_proprietario) references proprietario(nif),

constraint pk\_pagamento primary key(cod\_pagamento)

)

create table pagamento\_taxa(

cod\_pagamento\_taxa number(3),

cod\_pagamento number(3),

cod\_taxa number(3),

foreign key(cod\_taxa) references taxa(cod\_taxa),

foreign key(cod\_pagamento) references pagamento(cod\_pagamento),

constraint pk\_pagamento\_taxa primary key(cod\_pagamento\_taxa)

)

create table historico(

cod\_historico number(3),

motivo varchar2(8) not null check (motivo = 'venda' or motivo = 'juridico'),

data\_mudanca date not null,

cod\_novo\_proprietario varchar2(14),

cod\_antigo\_proprietario varchar2(14),

foreign key(cod\_novo\_proprietario) references proprietario(nif),

foreign key(cod\_antigo\_proprietario) references proprietario(nif),

constraint pk\_historico primary key(cod\_historico)

)

create table escritorio\_historico(

cod\_escritorio\_historico number(3),

cod\_historico number(3),

cod\_escritorio number(3),

foreign key(cod\_historico) references historico(cod\_historico),

foreign key(cod\_escritorio) references escritorio(cod\_escritorio),

constraint pk\_escritorio\_historico primary key(cod\_escritorio\_historico)

)

create table apartamento\_historico(

cod\_apartamento\_historico number(3),

cod\_historico number(3),

num\_apartamento number(3),

num\_andar number(3),

foreign key(cod\_historico) references historico(cod\_historico),

foreign key(num\_apartamento, num\_andar) references apartamento(num\_apartamento, num\_andar),

constraint pk\_apartamento\_historico primary key(cod\_apartamento\_historico))

SQL-DML (data manipulation language) - os comandos que manipulam as informações contidas nos dados.

Em anexo encontram-se os scripts utilizados para a inserção dos dados:

insert into proprietario( nome\_proprietario, nif, telefone, email ) values('antónio kadiamossiko', '007120555la045', '938709693', 'antonio-kadiamossiko@gmail.com');

insert into bloco( num\_bloco, longitude, latitude, area, nif\_coordenador ) values( 5, 100.3, 100.4, 24.5, '007120555la045');

insert into quarteirao( num\_bloco, designacao) values( 5, 'f');

insert into edificio( num\_edificio, num\_bloco, nif\_coordenador, total\_andares, modelo, utilidade, is\_rampa ) values(12, 5, '007120555la045', 3, 'upper-class', 'moradia', 1 );

insert into fraccao( cod\_fraccao, designacao, num\_edificio ) values( 1, 'a', 12 );

insert into rua( cod\_rua, nome) values(1, 'rua-01');

insert into tipo\_rua( cod\_tipo\_rua, tipo) values(1, 'principal');

insert into tipo\_rua( cod\_tipo\_rua, tipo) values(2, 'secundaria');

insert into rua\_tipo\_rua( cod\_rua\_tipo\_rua, cod\_rua, cod\_tipo\_rua ) values(1,1, 1 );

insert into edificio\_rua( cod\_edificio\_rua, cod\_rua, num\_edificio) values(1,1, 12 );

insert into upper\_class( cod\_upper\_class,num\_edificio, is\_elevador) values(1, 12, 1 );

insert into tipo\_moradia( cod\_tipo\_moradia, num\_edificio, total\_apartamentos ) values(1, 12, 8 );

insert into topologia( cod\_topologia, tipo, num\_edificio) values(1, 't5', 12 );

insert into taxa( cod\_taxa, valor, data\_pagamento) values(1, '50000', to\_date('21/02/2021','dd/mm/yyyy') );

insert into apartamento( num\_apartamento,num\_andar,cod\_tipo\_moradia,cod\_topologia,cod\_taxa) values(101, 1, 1, 1, 1);

insert into parque\_estacionamento( cod\_parque\_estacionamento,area,apt\_correspondente,num\_edificio) values(1, 10.2, 101, 12);

insert into pessoa\_fisica( cod\_pessoa\_fisica, cod\_proprietario, bi,num\_andar, num\_apartamento ) values(1, '007120555la045', '007120555la045', 1, 101 );

insert into pagamento( cod\_pagamento, cod\_proprietario, tipo, data\_pagamento) values(1, '007120555la045', 'anual', to\_date('21/02/2021', 'dd/mm/yyyy') );

insert into pagamento\_taxa( cod\_pagamento\_taxa, cod\_pagamento, cod\_taxa) values(1, 1, 1 );

Conclusão

De acordo com o trabalho desenvolvido, chegamos a seguinte conclusão:

O sucesso de uma organização, depende das organizações singulares dos seus departamentos. Uma das consequências da organização é a facilidade em manipular e divulgar acesso as informações em sua disposição. Isso tem influência directa na tomada de decisões e aumento de produtividade.

O Oracle é principal sistema de gerenciamento de base de dados por transmitir confiabilidade e segurança aos seus utilizadores.

Bibliografia

# <https://www.teses.usp.br/teses/disponiveis/3/3141/tde-31032008-094801/publico/Alex_Poletto.pdf>

# slide da cadeira do 3º ano - Faculdade de Ciências da Universidade Agostinho Neto Ano Lectivo 2009 Mateus Padoca Calado

# https://www.portalgsti.com.br/oracle-database/sobre/