Instituto Superior de Engenharia de Coimbra

Curso Técnico Superior e Profissional –

Tecnologias e Programação de Sistemas de Informação

Leandro Adão Fidalgo | a21270093

Pedro dos Santos Alves | a21270246

Base de dados

Trabalho Prático

Coimbra, 20 de janeiro de 2019

Índice

1. Introdução

O presente relatório descreve o projeto desenvolvido pelos alunos: Leandro Fidalgo e Pedro Alves, no âmbito da disciplina de Base de Dados, no Curso Técnico Superior Profissional de Tecnologias e Programação de Sistemas de Informação do Instituto Superior de Engenharia de Coimbra. O propósito deste trabalho é consolidar todos os conhecimentos adquiridos no decorrer do semestre.

O objetivo do trabalho consiste na criação de uma base de dados, na qual o objetivo é registar pelos menos os detalhes das equipas, jogadores, treinadores (principais e adjuntos), sanções disciplinares, resultados, classificação, melhores marcadores, guarda redes com mais golos sofridos, e outras informações pertinentes.

Para esse processo foi necessário a realização de um Diagrama do modelo E/R. Durante o processo foram surgindo algumas dúvidas, que conseguiram ser superadas através do auxílio do professor da disciplina e das fichas de trabalho por ele disponibilizadas.

Este processo foi constituído por várias etapas e tarefas demonstradas nos tópicos em baixo descritos.

2. Tabelas

Tabela época:

```
CREATE TABLE epoca (
    id_epoca     NUMBER,
    nome     VARCHAR2(100) UNIQUE NOT NULL,
    data_inicio DATE NOT NULL,
    data_fim     DATE NOT NULL,
    PRIMARY KEY(id_epoca)
);
```

Tabela liga:

```
CREATE TABLE liga (
    id_liga  NUMBER,
    nome         VARCHAR2(100) NOT NULL,
    epoca_id_epoca NUMBER NOT NULL,
    PRIMARY KEY(id_liga)
);
```

Tabela patrocinador:

```
CREATE TABLE patrocinador (
    id_patrocinador NUMBER,
    nome         VARCHAR2(100) UNIQUE NOT NULL,
    PRIMARY KEY(id_patrocinador)
);
```

Tabela Equipa e estádio:

```
CREATE TABLE equipa_estadio (
    id_equipa        NUMBER,
    nome         VARCHAR2(100) UNIQUE NOT NULL,
    fundacao         DATE NOT NULL,
    material_desportivo VARCHAR2(100) NOT NULL,
    alcunha      VARCHAR2(100),
    estadio_nome     VARCHAR2(100) UNIQUE NOT NULL,
    estadio_capacidade   NUMBER(5,0) NOT NULL,
    estadio_localizacao VARCHAR2(100),
    liga_id_liga     NUMBER NOT NULL,
    PRIMARY KEY(id_equipa)
);
```

Tabela Staff:

```
CREATE TABLE staff (
    id_staff     NUMBER,
    nacionalidade VARCHAR2(100) NOT NULL,
    nome          VARCHAR2(100) NOT NULL,
    tipo          VARCHAR2(100) NOT NULL,
    PRIMARY KEY(id_staff)
);
```

Tabela Liga_Equipa_Estadio - classificação:

```
CREATE TABLE liga_equipa_estadio (
    liga_id_liga         NUMBER,
    equipa_estadio_id_equipa NUMBER,
    numvitorias  NUMBER(3,0),
    numderrotas  NUMBER(3,0),
    numempates   NUMBER(3,0),
    numvitoriascons  NUMBER(3,0),
    pontos       NUMBER(3,0) NOT NULL,
    PRIMARY KEY(liga_id_liga,equipa_estadio_id_equipa)
);
```

Tabela Jogo:

```
CREATE TABLE jogo (
    id_jogo          NUMBER,
    data_jogo        DATE NOT NULL,
    res_casa             NUMBER(2,0),
    res_fora             NUMBER(2,0),
    jornada          NUMBER(3,0) NOT NULL,
    equipa_estadio_id_equipa     NUMBER NOT NULL,
    equipa_estadio_id_equipal NUMBER NOT NULL,
    PRIMARY KEY(id_jogo)
);
```

Tabela Jogador:

```
CREATE TABLE jogador (
    id_jogador   NUMBER,
    nome         VARCHAR2(100) NOT NULL,
    naturalidade     VARCHAR2(100) NOT NULL,
    nacionalidade VARCHAR2(100) NOT NULL,
    data_nasc    DATE NOT NULL,
    altura   NUMBER(3,0) NOT NULL,
    salario  NUMBER(10,0) NOT NULL,
    posicao  VARCHAR2(100),
    pe       VARCHAR2(100) NOT NULL,
    golos    NUMBER(5,0) NOT NULL,
    golos_suf NUMBER(5,0) NOT NULL,
    PRIMARY KEY(id_jogador)
);
```

Tabela Sancao_Disc - sanção disciplinar:

```
CREATE TABLE sancao_disc (
    id_sancao NUMBER,
    jogo_id_jogo NUMBER NOT NULL,
    cartao   VARCHAR2(8),
    coima    NUMBER(10,0) NOT NULL,
    data_sanc DATE NOT NULL,
    PRIMARY KEY(id_sancao)
);
```

Tabela Liga+Equipa+Estadio - equipa vencedora:

```
CREATE TABLE equipa_estadio_liga (
    equipa_estadio_id_equipa NUMBER NOT NULL,
    liga_id_liga         NUMBER,
    datav    DATE NOT NULL,
    PRIMARY KEY(liga_id_liga)
);
```

Tabela Jogador+Jogo - jogadores que marcam golos e seus golos no jogo:

```
CREATE TABLE jogador_jogo (
    jogador_id_jogador NUMBER,
    jogo_id_jogo     NUMBER,
    golos_marc NUMBER(5,0) NOT NULL,
    PRIMARY KEY(jogador_id_jogador,jogo_id_jogo)
);
```

Tabela Jogador_Sancao_Disc - jogadores que levaram sanções disciplinares:

```
CREATE TABLE jogador_sancao_disc (
    jogador_id_jogador   NUMBER,
    sancao_disc_id_sancao NUMBER,
    PRIMARY KEY(jogador_id_jogador,sancao_disc_id_sancao)
);
```

Tabela Staff_Sancao_Disc - staff que levaram sanções disciplinares:

```
CREATE TABLE staff_sancao_disc (
    staff_id_staff   NUMBER,
    sancao_disc_id_sancao NUMBER,
    PRIMARY KEY(staff_id_staff,sancao_disc_id_sancao)
);
```

Tabela Liga_Jogador - GR_Mais_GS:

```
CREATE TABLE GR_Mais_GS (
    golos    NUMBER,
    jogador_id_jogador NUMBER,
    PRIMARY KEY(jogador_id_jogador)
);
```

Tabela Jogador_Liga - Melhor_Marc:

```
CREATE TABLE melhor_marc (
    jogador_id_jogador NUMBER,
    golos    NUMBER,
    PRIMARY KEY(jogador_id_jogador)
);
```

Tabela Jogador_Equipa_Estadio:

```
CREATE TABLE jogador_equipa_estadio (
    jogador_id_jogador   NUMBER,
    equipa_estadio_id_equipa NUMBER NOT NULL,
    PRIMARY KEY(jogador_id_jogador)
);
```

Tabela Equipa_Estadio_Staff:

```
CREATE TABLE equipa_estadio_staff (
    equipa_estadio_id_equipa NUMBER NOT NULL,
    staff_id_staff       NUMBER,
    PRIMARY KEY(staff_id_staff)
);
```

Tabela Patrocinador_Equipa_Estadio:

```
CREATE TABLE patrocinador_equipa_estadio (
    patrocinador_id_patrocinador NUMBER,
    equipa_estadio_id_equipa     NUMBER,
    PRIMARY KEY(patrocinador_id_patrocinador,equipa_estadio_id_equipa)
);
```

Tabela Substituicoes - substuições num determinado jogo:

```
CREATE TABLE substituicoes (
    jogo_id_jogo NUMBER,
    data_sub DATE NOT NULL,
    jogador_id_jogador1 NUMBER NOT NULL,
    jogador_id_jogador2 NUMBER NOT NULL,
    PRIMARY KEY(jogo_id_jogo, jogador_id_jogador1, jogador_id_jogador2)
);
```

3. Vistas

Vista A:

No que respeita às equipas, apresente: a equipa que ganhou o penúltimo campeonato, a equipa que teve mais cartões amarelos (jogadores e treinadores) nesta época, a equipa que sofreu mais golos, em casa, no ultimo mês.

```
create view VISTA_A as (
    select x.NOM as "Ganharam penúltimo campeonato", yz.NOMY as "Mais cartões amarelos", z.NOM as "Sofreu mais golos"
    from
        (select ee.nome NOM
        from equipa_estadio ee, equipa_estadio_liga eel, epoca e, liga l
        where ee.liga_id_liga = l.id_liga and e.id_epoca = l.epoca_id_epoca and ee.id_equipa = eel.equipa_estadio_id_equipa and
        (to_char(eel.DATAV, 'YYYY') = to_char(sysdate, 'YYYY') - 2) and l.NOME = 'Primeira Divisão'
        ) x, (
            select YY.NOM NOMY
            from (
                select ee.nome NOM, max(y.co + z.co)
                from (
                    select distinct ee.ID_EQUIPA, count(ee.ID_EQUIPA) co
                    from equipa_estadio ee, staff_sancao_disc ssd, staff s, equipa_estadio_staff ees
                    where ee.ID_EQUIPA = ees.EQUIPA_ESTADIO_ID_EQUIPA and ees.STAFF_ID_STAFF = s.ID_STAFF and ssd.STAFF_ID_STAFF = s.ID_STAFF
                    group by ee.ID_EQUIPA
                    ) y, (
                    select distinct ee.ID_EQUIPA, count(ee.ID_EQUIPA) co
                    from equipa_estadio ee, jogador_sancao_disc jsd, jogador p, jogador_equipa_estadio jee
                    where ee.ID_EQUIPA = jee.EQUIPA_ESTADIO_ID_EQUIPA and jee.JOGADOR_ID_JOGADOR = p.ID_JOGADOR and jsd.JOGADOR_ID_JOGADOR = p.ID_JOGADOR
                    group by ee.ID_EQUIPA
                    ) z, equipa_estadio ee
                where y.ID_EQUIPA = z.ID_EQUIPA and ee.id_equipa = z.id_equipa and (ROWNUM = 1)
                group by ee.nome
                order by max(y.co + z.co) desc) YY
        ) yz, (
            select zz.nom
            from (
                select ee.nome nom, max(x.soma) som
                from equipa_estadio ee, (
                    select j.EQUIPA_ESTADIO_ID_EQUIPA equip, sum(j.RES_FORA) soma
                    from jogo j
                    group by j.EQUIPA_ESTADIO_ID_EQUIPA
                    ) x
                where ee.ID_EQUIPA = x.equip
                group by ee.nome
                order by som desc) ZZ
            where rownum = 1
        ) z
);
```

Vista B:

Mostre quantos jogos foram realizados por equipas com treinadores de nacionalidade portuguesa, na presente época.

```
create view VISTA_B as (
    select count(j.ID_Jogo) as NumJogos
    from jogo j, staff s, epoca e, equipa_estadio ee, equipa_estadio_staff ees, liga l
    where ees.equipa_estadio_id_equipa = ee.id_equipa and ee.liga_id_liga = l.id_liga and l.epoca_id_epoca = e.id_epoca and
    s.id_staff = ees.staff_id_staff and (j.equipa_estadio_id_equipa = ee.id_equipa or j.equipa_estadio_id_equipa1 = ee.id_equipa) and
    s.tipo = 'Treinador' and to_char(e.DATA_FIM, 'YYYY') = to_char(sysdate, 'YYYY') and s.NACIONALIDADE = 'Portuguese'
);
```

Vista C:

Apresente a lista de jogadores portugueses que jogam em equipas da segunda divisão esta época e que nunca receberam uma sanção disciplinar.

```
create or replace view VISTA_C as (
    select p.nome NOME
    from jogador p, EQUIPA_ESTADIO ee, liga l, epoca e, jogador_equipa_estadio jee
    where jee.equipa_estadio_id_equipa = ee.id_equipa and jee.jogador_id_jogador = p.id_jogador and ee.liga_id_liga = l.id_liga and
    l.epoca_id_epoca = e.id_epoca and
    (select count(jsdd.jogador_id_jogador) from jogador_sancao_disc jsdd where jsdd.jogador_id_jogador = p.id_jogador) = 0 and
    (to_char(e.Data_Fim, 'YYYY') = '2018') and
    l.NOME = 'Segunda Divisão' and p.nacionalidade = 'Portuguese'
);
```

Vista D:

Lista de jogadores, que jogam na posição “extremo”, que no último mês marcaram pelo menos dois golos.

```
create or replace view VISTA_D as (
    select distinct p.nome
    from jogador p, jogo j, jogador_jogo pj
    where p.ID_JOGADOR = pj.JOGADOR_ID_JOGADOR and pj.JOGO_ID_JOGO = j.ID_JOGO and
    p.POSICAO = 'Avançado' and (select count(jogador_jogo.jogador_id_jogador) from jogador_jogo, jogador where p.id_jogador = jogador.id_jogador
    and jogador_jogo.jogador_id_jogador = p.id_jogador and jogador_jogo.jogador_id_jogador = jogador.id_jogador
    group by jogador_jogo.jogador_id_jogador) >= 2 and ('2018' - to_char(j.Data_Jogo, 'YYYYMM') <= 1)
);
```

Vista E:

Apresente o número total de golos marcados por equipas de Lisboa e do Porto na última época.

```
create view VISTA_E as (
    select count(j.jogador_id_jogador) NUMGOLS
    from jogador_jogo j, epoca e, equipa_estadio ee, jogador p, liga l, jogador_equipa_estadio jee
    where j.JOGADOR_ID_JOGADOR = p.id_jogador and jee.JOGADOR_ID_JOGADOR = p.id_jogador and jee.EQUIPA_ESTADIO_ID_EQUIPA = ee.id_equipa and ee.LIGA_ID_LIGA = l.id_liga and
    l.EPOCA_ID_EPOCA = e.id_epoca and ((to_char(e.Data_Fim, 'YYYY') = to_char(SYSDATE, 'YYYY') - 1)) and
    (ee.ESTADIO_LOCALIZACAO = 'Lisboa' or ee.ESTADIO_LOCALIZACAO = 'Porto')
);
```

Vista F:

Apresente o clube e os nomes dos jogadores que receberam cartões vermelhos nos últimos três meses. Considere apenas os jogadores a jogar na posição “guarda-redes” e que tenham sido utilizados em mais do que três jogos. (apresente o resultado por ordem alfabética)

```
create view VISTA_F as (
    select Jogador JOGADOR,
    Equipa EQUIPA from
        (select distinct (p.nome) Jogador, (ee.nome) Equipa
        from jogador p, equipa_estadio ee, jogador_sancao_disc jsd, sancao_disc sd, jogador_equipa_estadio jee
        where p.ID_JOGADOR = jsd.JOGADOR_ID_JOGADOR and jsd.SANCAO_DISC_ID_SANCAO = sd.ID_SANCAO and jee.EQUIPA_ESTADIO_ID_EQUIPA = ee.ID_EQUIPA and
        jee.JOGADOR_ID_JOGADOR = p.ID_JOGADOR and p.POSICAO = 'Guarda-Redes' and sd.CARTAO = 'Vermelho' and ((select to_char(data_jogo, 'YYYYMM') from jogo where sd.jogo_id_jogo = jogo.id_jogo) <= '3')
        order by p.nome, ee.nome)
);
```

Vista G:

Apresente os quatros melhores goleadores, fora de casa, das últimas duas épocas.

```
create view VISTA_G as (
    select co as Jogador
    from (
        select distinct p.nome co,  count(jj.jogador_id_jogador)
        from jogador p, jogador_jogo jj, equipa_estadio ee, liga l, epoca e, jogador_equipa_estadio jee
        where jj.jogador_id_jogador = p.id_jogador and jee.EQUIPA_ESTADIO_ID_EQUIPA = ee.id_equipa and jee.JOGADOR_ID_JOGADOR = p.id_jogador and
        ee.LIGA_ID_LIGA = l.id_liga and l.EPOCA_ID_EPOCA = e.id_epoca and (to_char(e.Data_Fim, 'YYYY') = to_char(SYSDATE, 'YYYY') - '1' or to_char(e.Data_Fim, 'YYYY') = to_char(SYSDATE, 'YYYY') - '2')
        group by p.nome
        order by count(jj.jogador_id_jogador) desc) x
    where (ROWNUM <= 4)
);
```

Vista H:

Obtenha uma lista de treinadores que tenham ganho mais do que três jogos consecutivos e menos do que duas derrotas no campeonato. Ordene por ordem descente pelo número de vitorias.

```
create view VISTA_H as (
    select x.nom NOME
    from (
        select s.nome nom
        from staff s, equipa_estadio ee, liga_equipa_estadio lee, liga l, equipa_estadio_staff ees
        where s.ID_STAFF = ees.staff_id_staff and ees.EQUIPA_ESTADIO_ID_EQUIPA = ee.id_equipa and
        ee.id_equipa = lee.equipa_estadio_id_equipa and l.id_liga = lee.liga_id_liga and lee.numvitoriascons > 3 and
        lee.NUMDERROTAS < 2  and s.tipo like '%Treinador%'
        order by lee.numvitorias desc) x
);
```

Vista J:

Para todos os jogos da última jornada, apresente uma listagem com o nome do estádio onde se realizou o jogo, nome do treinador, resultado final, quantidade de golos sofridos e marcados. Ordene descendentemente pelo nome da equipa.

```
create view VISTA_J as (
    select * from (
        select distinct ee.estadio_nome as "Nome Estadio", s.nome as Treinador, 'Resultado: ' || j.res_casa || ' - ' || j.res_fora as Resultado, j.res_casa as "Golos Marcados", j.res_fora as "Golos Sufridos"
        from equipa_estadio ee, jogo j, staff s, equipa_estadio_staff ees
        where j.EQUIPA_ESTADIO_ID_EQUIPA = ee.ID_EQUIPA and ees.STAFF_ID_STAFF = s.ID_STAFF and ees.EQUIPA_ESTADIO_ID_EQUIPA = ee.ID_EQUIPA and
        s.TIPO = 'Treinador' and j.jornada = (select max(jogo.jornada) from jogo)
        order by ee.estadio_nome desc)
);
```

Vista L:

Para cada clube, apresente a lista de resultados por jogo. Apenas considere jogos com mais de 4 golos, realizados entre as 17:00 e as 23:00. Ordene descendentemente pelo número de golos.

```
create view VISTA_L as (
    select * from (
        select j.RES_CASA || ' - ' || j.res_fora as "Resultado", ee.NOME as "Nome da Equipa"
        from jogo j, equipa_estadio ee
        where (j.EQUIPA_ESTADIO_ID_EQUIPA = ee.ID_EQUIPA or j.EQUIPA_ESTADIO_ID_EQUIPA1 = ee.ID_EQUIPA)
        and (j.res_casa + j.res_fora) > 4 and (to_char(j.data_jogo, 'HH24MI') >= '1700'
        and to_char(j.data_jogo, 'HH24MI') <= '2300') order by (j.res_casa + j.res_fora) desc)
);
```

Vista M e N:

Pelo menos duas consultas adicionais, que o grupo considere relevantes, justificando a sua relevância. A relevância e o nível de complexidade das mesmas influenciará fortemente a sua avaliação: a. VISTA_M que inclua um SELECT com GROUP BY. b. VISTA_N que inclua um SELECT encadeado.

```
create view VISTA_M as (
    select x.NomeEquipa, x.NumPatrocinadores
    from (
        select ee.nome NomeEquipa, count(pa.id_patrocinador) NumPatrocinadores
        from equipa_estadio ee, patrocinador_equipa_estadio pee, patrocinador pa
        where ee.id_equipa = pee.equipa_estadio_id_equipa and pee.patrocinador_id_patrocinador = pa.id_patrocinador
        group by ee.nome
        order by NumPatrocinadores desc) x
);
```

4. Funções, Procedimentos e Triggers

Exercício A:

Implementar os mecanismos necessários a garantir a integridade dos dados que não seja assegurada por restrições da BD, incluindo as restrições do numero de cartões por jogo/jogador (garantir que um jogador só pode ter 2 cartões amarelos ou/ e um vermelho), que nenhum jogador/treinador pode pertencer em simultâneo a duas equipas e que uma equipa não pode ter mais do que 5 jogadores estrangeiros.

```
create or replace trigger rest_num_cartoes
before insert
on jogador_sancao_disc
for each row
declare
    numAm number := 0;
    numVe number := 0;

begin
    declare
    begin
        select count(sd.id_sancao) into numAm
        from sancao_disc sd, jogador_sancao_disc jsd
        where sd.id_sancao = jsd.sancao_disc_id_sancao and jsd.jogador_id_jogador = :new.jogador_id_jogador and sd.cartao = 'Amarelo'
        group by sd.jogo_id_jogo;
    exception
        when no_data_found then
            numAm := 0;
    end;

    declare
    begin
        select count(sd.id_sancao) into numAm
        from sancao_disc sd, jogador_sancao_disc jsd
        where sd.id_sancao = jsd.sancao_disc_id_sancao and jsd.jogador_id_jogador = :new.jogador_id_jogador and sd.cartao = 'Vermelho'
        group by sd.jogo_id_jogo;
    exception
        when no_data_found then
            numVe := 0;
    end;

    if (numAm >= 2) then
        update sancao_disc set cartao = 'Vermelho' where id_sancao = :new.sancao_disc_id_sancao;
    end if;

    if (numVe >= 1) then
        insert into avisos(mensagem) values ('O jogador ' || (select nome from jogador where id_jogador = :new.jogador_id_jogador) || ' já tem um cartão vermelho.');
        -- raise_application_error cancela o insert?
        -- isto vai dar erro quando for dar insert...? não pode ser null...
        :new.jogador_id_jogador := null;
    end if;
end;
/


create trigger nacionais_cinco
before insert
on jogador_equipa_estadio
for each row
declare
    numJogEst number := 0;
begin
    select count(jee.jogador_id_jogador) into numJogEst
    from jogador jog, jogador_equipa_estadio jee
    where jog.id_jogador = jee.jogador_id_jogador and jog.naturalidade <> 'Portugal' and jee.equipa_estadio_id_equipa = :new.equipa_estadio_id_equipa;

    if (numJogEst > 5) then
        insert into avisos(mensagem) values ('Não pode ter mais que 5 jogadores estrangeiros na equipa...');
        :new.jogador_id_jogador := null;
    end if;
end;
/
```

```
create trigger jog_mesma_equip
before insert
on jogador_equipa_estadio
for each row
declare
    jogExiste number := 0;
begin
    select count(jogador_id_jogador) into jogExiste
    from jogador_equipa_estadio
    where jogador_id_jogador = :new.jogador_id_jogador;

    if (jogExiste > 0) then
        insert into avisos(mensagem) values ('Jogador já existe...');
        :new.jogador_id_jogador := null;
    end if;
end;
/
```

```
create trigger staf_mesma_equip
before insert
on equipa_estadio_staff
for each row
declare
    StafExiste number := 0;
begin
    select count(staff_id_staff) into StafExiste
    from equipa_estadio_staff
    where staff_id_staff = :new.staff_id_staff;

    if (StafExiste > 0) then
        insert into avisos(mensagem) values ('Staff já existe...');
        :new.staff_id_staff := null;
    end if;
end;
/
```

Exercício B:

Criar a função numero_golos que recebe como argumento um código de uma posição (ex: avançado, extremo) e retorna o numero de golos realizados por todos os jogadores dessa posição, nos últimos 3 meses. A função deve lançar exceções (ver exceções a baixo).

```
create or replace function numero_golos_func(pos varchar2) return number
is
    maquinadefazergols number := 0;
    dataFimEpo date;
begin
    if (LOWER(pos) <> 'avançado' and LOWER(pos) <> 'médio' and LOWER(pos) <> 'defesa' and LOWER(pos) <> 'guarda-redes') then
        raise_application_error(-20100, 'Código de posição ' || LOWER(pos) || ' inválido.');
    end if;

    select EPOCA.DATA_FIM into dataFimEpo
    from epoca
    where epoca.id_epoca = (select count(*) from epoca);

    select count(jj.jogador_id_jogador) into maquinadefazergols
    from jogador_jogo jj, jogo j, jogador jo
    where j.id_jogo = jj.jogo_id_jogo and jj.jogador_id_jogador = jo.id_jogador and
    (LOWER(jo.posicao) = LOWER(pos)) and (to_char(dataFimEpo, 'yyyymm') - to_char(j.data_jogo, 'yyyymm') <= '30');

    return maquinadefazergols;
end;
/

select numero_golos_func('avançado') from dual;
select numero_golos_func('guarda-redes') from dual;
select numero_golos_func('defesa') from dual;
select numero_golos_func('médio') from dual;

set serveroutput on;

declare
    var number := 0;
begin
select numero_golos_func('avbv') into var from dual;
exception
    when others then
        DBMS_OUTPUT.PUT_LINE('Cod: ' || SQLCODE || ' Message: ' || SQLERRM);
end;
/
```

Exercício C:

Criar a função numero_amarelos que recebe como argumento o nome de um jogador, uma data inicial e uma data final. Esta função deve retornar o somatório do número de cartões amarelos, recebidos apenas em jogos em casa, nesse intervalo de tempo. A função deve lançar exceções (ver exceções abaixo).

```
create function numero_amarelos(nomeJog varchar2, dataInit date, dataFin date) return number
is
    soma number := 0;
    jogadorExiste number := 0;
    epo1 number := 1;
    epo2 number := 4;
    dataInitEpo date;
    dataFinEpo date;
begin
    select count(*) into jogadorExiste from jogador where nome = nomeJog;
    if (jogadorExiste = 0) then
        raise_application_error(-20101, 'Nome do jogador ' || nomeJog || ' inválido.');
    end if;

    select count(*) into epo2 from epoca;

    select data_inicio into dataInitEpo from epoca where id_epoca = epo1;
    select data_fim into dataFinEpo from epoca where id_epoca = epo2;

    if (dataInit > dataInitEpo) then
        raise_application_error(-20102, 'Data ' || to_char(dataInit, 'yyyy-mm-dd HH24:MI') || ' inválida.');
    end if;

    if (dataFin < dataFinEpo) then
        raise_application_error(-20102, 'Data ' || to_char(dataFin, 'yyyy-mm-dd HH24:MI') || ' inválida.');
    end if;

    select count(sd.ID_SANCAO) into soma
    from sancao_disc sd, jogador jg, jogo j, jogador_sancao_disc jsd, equipa_estadio ee
    where sd.ID_SANCAO = jsd.SANCAO_DISC_ID_SANCAO and jsd.jogador_id_jogador = jg.ID_JOGADOR and jg.NOME = nomeJog and sd.cartao = 'Amarelo' and
    sd.jogo_id_jogo = j.ID_JOGO and j.EQUIPA_ESTADIO_ID_EQUIPA = ee.ID_EQUIPA and j.DATA_JOGO > dataInit and j.DATA_JOGO < dataFin;

    return soma;
end;
/
```

Exercício D:

Criar o trigger golos_marcados que, após cada golo marcado, atualize uma tabela

com os 10 melhores goleadores e atualize a tabela com os guarda-redes que mais golos sofreram. Deve ainda enviar um aviso de parabéns à equipa e ao treinador.

```
create or replace trigger golos_marcados
after insert or update
on jogador_jogo
for each row
declare
    totalGols number := 0;
    minGols number := 0;
    top number := 0;
    jogETop number := 0;
    idJogAntigo number;

    nomeEquipa equipa_estadio.nome%type;
    nomeTreinador staff.nome%type;

    idEquipa number;
    idEquipD number;

    GR number;

    GRtotalGols number := 0;
    GRminGols number := 0;
    GRtop number := 0;
    GRjogETop number := 0;
    GRidJogAntigo number;
begin
    -- receber o total de golos que do jogador que marcou
    select j.golos into totalGols
    from jogador j
    where j.ID_JOGADOR = :new.JOGADOR_ID_JOGADOR;

    -- receber o minimo de golos que existe na tabela de melhor marcador
    select min(golos) into minGols
    from melhor_marc;

    -- receber id do jogador com min golos
    select jogador_id_jogador into idJogAntigo
    from melhor_marc
    where golos = minGols;

    -- quantos jogadores estão no top 5
    select count(*) into top
    from melhor_marc;
```

```
    -- se top 5 já está cheio
    if (top = 5) then
        select count(*) into jogETop
        from melhor_marc mm
        where mm.jogador_id_jogador = :new.JOGADOR_ID_JOGADOR;

        -- se jogador não está no top 5
        if (jogETop = 0) then
            if (totalGols > minGols) then
                update melhor_marc set jogador_id_jogador = :new.JOGADOR_ID_JOGADOR, golos = totalGols where jogador_id_jogador = idJogAntigo;
            end if;
        else
            -- se já estiver no top 5 coloca os golos certos
            update melhor_marc set golos = totalGols where jogador_id_jogador = :new.JOGADOR_ID_JOGADOR;
        end if;
    else
        insert into melhor_marc(jogador_id_jogador, golos) values(:new.JOGADOR_ID_JOGADOR, totalGols);
    end if;

    -- Id da equipa que marcou
    select ee.id_equipa into idEquipa
    from equipa_estadio ee, jogador_jogo jj, jogador_equipa_estadio jee
    where ee.ID_EQUIPA = jee.equipa_estadio_id_equipa and jj.jogador_id_jogador = jee.jogador_id_jogador and jj.jogador_id_jogador = :new.JOGADOR_ID_JOGADOR;

    select j.equipa_estadio_id_equipa into idEquipD
    from jogo j
    where j.id_jogo = :new.jogo_id_jogo;

    if (idEquipD = idEquipa) then
        select j.equipa_estadio_id_equipa1 into idEquipD
        from jogo j
        where j.id_jogo = :new.jogo_id_jogo;
    end if;

    select jee.jogador_id_jogador into GR
    from jogador_equipa_estadio jee, jogador jog
    where jee.equipa_estadio_id_equipa = idEquipD and jee.jogador_id_jogador = jog.id_jogador and LOWER(jog.posicao) = 'guarda-redes';

    -- receber o total de golos sufridos
    select j.golos_suf into GRtotalGols
    from jogador j
    where j.ID_JOGADOR = GR;

    -- receber o minimo de golos que existe na tabela de GR_mais_GS
    select min(golos) into GRminGols
    from gr_mais_gs;

    -- receber id do jogador com min golos_suf
    select jogador_id_jogador into idJogAntigo
    from gr_mais_gs
    where golos = GRminGols;

    -- quantos jogadores estão no top 5
    select count(*) into GRtop
    from gr_mais_gs;
```

```
    -- se top 5 já está cheio
    if (top = 5) then
        select count(*) into GRjogETop
        from gr_mais_gs
        where jogador_id_jogador = GR;

        -- se jogador não está no top 5
        if (GRjogETop = 0) then
            if (GRtotalGols > GRminGols) then
                update gr_mais_gs set jogador_id_jogador = GR, golos = GRtotalGols where jogador_id_jogador = GRidJogAntigo;
            end if;
        else
            -- se já estiver no top 5 coloca os golos certos
            update gr_mais_gs set golos = GRtotalGols where jogador_id_jogador = GR;
        end if;
    else
        insert into gr_mais_gs(jogador_id_jogador, golos) values(GR, GRtotalGols);
    end if;

    -- Nome da equipa que marcou
    select ee.nome into nomeEquipa
    from equipa_estadio ee
    where ee.ID_EQUIPA = idEquipa;

    -- treinador da equipa que marcou
    select STAFF.NOME into nomeTreinador
    from equipa_estadio_staff ees, staff
    where idEquipa = ees.EQUIPA_ESTADIO_ID_EQUIPA and ees.staff_id_staff = staff.id_staff;

    insert into avisos(mensagem) values ('Parabéns á equipa ' || nomeEquipa || ' e ao treinador: ' || nomeTreinador || '.');
end;
/
```

Exercício E:

Criar uma função ultimo_jogo que recebe o nome de uma equipa e retorna a data do último ultimo jogo que tenham perdido fora de casa. A função deve lançar exceções (ver exceções abaixo).

```
create function ultimo_jogo(nomeEquip varchar2) return date
is
    equipExiste number := 0;
    idEquip equipa_estadio.id_equipa%type;
    dataJogo jogo.data_jogo%type;
begin
    select count(id_equipa) into equipExiste
    from equipa_estadio
    where nome = nomeEquip;

    if (equipExiste = 0) then
        raise_application_error(-20106, 'Nome/ID ' || nomeEquip || ' de equipa inválido.');
    end if;

    select id_equipa into idEquip
    from equipa_estadio
    where nome = nomeEquip;

    select d into dataJogo
    from (
        select data_jogo d
        from jogo
        where equipa_estadio_id_equipa1 = idEquip and res_casa > res_fora
        order by data_jogo desc)
    where ROWNUM = 1;

    return dataJogo;
end;
/
```

Exercício F:

Crie a função primeira_substituição que para um determinado jogo, determine quantos minutos jogou o jogador que foi substituído. A função deve retornar os minutos ou, NULL se nenhum potencial jogador tiver sido substituído.

```
create or replace function primeira_substituição(idJogo number, idJogador number) return number
is
    coJogo number := 0;
    coJoga number := 0;
    numMins number := 0;
begin
    select count(jogo_id_jogo) into coJogo
    from substituicoes
    where jogo_id_jogo = idJogo;

    if (coJogo = 0) then
        raise_application_error(-20110, 'Não existem jogos.');
    end if;

    select count(jogador_id_jogador1) into coJoga
    from substituicoes
    where jogador_id_jogador1 = idJogador;

    if (coJoga = 0) then
        raise_application_error(-20109, 'Não existem substituições.');
    end if;

    select min(s.minuto_sub) into numMins
    from substituicoes s, jogo j
    where s.jogo_id_jogo = idJogo and s.jogador_id_jogador1 = idJogador and j.id_jogo = s.jogo_id_jogo;

    return numMins;
end;
/
```

Exercicio G:

Implemente um trigger cartões que, ao inserir um cartão vermelho na ficha de um dado jogador na base de dados, submeta um aviso à equipa desse mesmo jogador a informar que não pode jogar na próxima jornada. Ao receber um cartão vermelho deve ainda garantir que ele se encontra no estado suspenso por 2 jogos.

```
create trigger cartoes
after insert
on jogador_sancao_disc
for each row
declare
    corcartao sancao_disc.cartao%type;

    nomeJog jogador.nome%type;
    nomeEqui equipa_estadio.nome%type;
begin
    select cartao into corcartao
    from sancao_disc
    where id_sancao = :new.sancao_disc_id_sancao;

    if (corcartao <> null and corcartao = 'Vermelho') then
        select nome into nomeJog
        from jogador
        where id_jogador = :new.jogador_id_jogador;

        select ee.nome into nomeEqui
        from equipa_estadio ee, jogador_equipa_estadio jee
        where ee.id_equipa = jee.equipa_estadio_id_equipa and jee.jogador_id_jogador = :new.jogador_id_jogador;

        insert into avisos(mensagem) values ('O jogador ' || nomeJog || ', da equipa ' || nomeEqui || ', estará suspenso durante dois jogos.');
    end if;
end;
/
```

Exercício H:

Criar a função clubes_cidade que recebe como argumento um Concelho (Ex Lisboa) e deve retornar a posição (avançados, guarda redes, etc) que mais jogadores, nacionais, tiveram inscritos na penúltima época desportiva. A função deve lançar exceções (ver exceções abaixo).

```
create or replace function clubes_cidade(cidade varchar2) return varchar2
is
    cidade_existe number := 0;

    numAv number := 0;
    numMe number := 0;
    numDe number := 0;
    numGr number := 0;
begin
    select count(id_equipa) into cidade_existe
    from equipa_estadio
    where LOWER(estadio_localizacao) = LOWER(cidade);

    if (cidade_existe = 0) then
        raise_application_error(-20104, 'Nome ' || cidade || ' de Concelho inválido.');
    end if;

    select count(j.id_jogador) into numAv
    from jogador j, equipa_estadio ee, jogador_equipa_estadio jee
    where j.id_jogador = jee.jogador_id_jogador and jee.equipa_estadio_id_equipa = ee.id_equipa and
    LOWER(ee.estadio_localizacao) = LOWER(cidade) and j.naturalidade = 'Portugal' and j.posicao = 'Avançado';

    select count(j.id_jogador) into numMe
    from jogador j, equipa_estadio ee, jogador_equipa_estadio jee
    where j.id_jogador = jee.jogador_id_jogador and jee.equipa_estadio_id_equipa = ee.id_equipa and
    LOWER(ee.estadio_localizacao) = LOWER(cidade) and j.naturalidade = 'Portugal' and j.posicao = 'Médio';

    select count(j.id_jogador) into numDe
    from jogador j, equipa_estadio ee, jogador_equipa_estadio jee
    where j.id_jogador = jee.jogador_id_jogador and jee.equipa_estadio_id_equipa = ee.id_equipa and
    LOWER(ee.estadio_localizacao) = LOWER(cidade) and j.naturalidade = 'Portugal' and j.posicao = 'Defesa';

    select count(j.id_jogador) into numGr
    from jogador j, equipa_estadio ee, jogador_equipa_estadio jee
    where j.id_jogador = jee.jogador_id_jogador and jee.equipa_estadio_id_equipa = ee.id_equipa and
    LOWER(ee.estadio_localizacao) = LOWER(cidade) and j.naturalidade = 'Portugal' and j.posicao = 'Guarda-Redes';

    if (numAv > numMe) then
        if (numAv > numDe) then
            if (numAv > numGr) then
                return 'Avançado';
            else
                return 'Guarda-Redes';
            end if;
        else
            if (numDe > numGr) then
                return 'Defesa';
            else
                return 'Guarda-Redes';
            end if;
        end if;
    else
        if (numMe > numDe) then
            if (numMe > numGr) then
                return 'Médio';
            else
                return 'Guarda-Redes';
            end if;
        else
            if (numDe > numGr) then
                return 'Defesa';
            else
                return 'Guarda-Redes';
            end if;
        end if;
    end if;
end;
/
```

Exercício I:

Criar o procedimento época_desportiva que ao receber uma época desportiva, que introduza numa tabela auxiliar, três melhores equipas e as três piores equipas.

```
create table aux(
    -- Posição...
    pos number,
    -- 0 = Melhor
    -- 1 = Pior
    mp number,
    -- id da equipa
    id_equipa number,
    primary key(id_equipa)
);
/

ALTER TABLE aux ADD CONSTRAINT aux_fk FOREIGN KEY (id_equipa) REFERENCES equipa_estadio(id_equipa);
/

create or replace procedure epoca_desportiva(idEpoca number)
is
    cursor melhores is select * from liga_equipa_estadio where liga_id_liga in (select id_liga from liga where epoca_id_epoca = idEpoca) order by pontos desc;
    cursor piores is select * from liga_equipa_estadio where liga_id_liga in (select id_liga from liga where epoca_id_epoca = idEpoca) order by pontos asc;

    x number := 0;
begin
    for c1 in melhores
    loop
        insert into aux(pos, mp, id_equipa) values(x, 0, c1.equipa_estadio_id_equipa);
        x := x + 1;
        exit when (x = 3);
    end loop;

    x := 0;

    for c2 in piores
    loop
        insert into aux(pos, mp, id_equipa) values(x, 1, c2.equipa_estadio_id_equipa);
        x := x + 1;
        exit when (x = 3);
    end loop;
end;
/
```

Exercício J:

Criar uma função idade que recebe uma idade, uma data inicial e uma data final. A função deve devolver o número de golos marcados por esses jogadores, apenas na presente época. Deve excluir jogadores que não estão no clube à menos de duas épocas. A função deve lançar exceções (ver exceções abaixo).

```
create or replace function idade(idade number, datInit date, datFin date) return number
is
    numGols number := 0;
    auxGols number := 0;
    cursor c1 is select * from jogador where idade = TRUNC(MONTHS_BETWEEN(SYSDATE, data_nasc))/12;
begin
    if (idade < 18 or idade > 100) then
        raise_application_error(-20105, 'Idade inválida');
    end if;

    if (datInit > sysdate or datInit > datFin) then
        raise_application_error(-20102, 'Data ' || to_char(datInit, 'dd-mm-yyyy') || ' inválida');
    end if;

    if (datFin > sysdate) then
        raise_application_error(-20102, 'Data ' || to_char(datFin, 'dd-mm-yyyy') || ' inválida');
    end if;

    for jo in c1
    loop
        select count(jj.jogador_id_jogador) into auxGols
        from jogador_jogo jj, jogo j
        where j.id_jogo = jj.jogo_id_jogo and jj.jogador_id_jogador = jo.id_jogador and j.data_jogo < datFin and j.data_jogo > datInit;

        numGols := numGols + auxGols;
    end loop;

    return numGols;
end;
/
```

Exercício K:

Implementar o procedimento alertas_treinador que recebe como argumento um id de um treinador e de seguida deve calcular, o número de jogos realizados, o número de jogos ganhos. o número de jogos perdidos, o número de golos marcados, por clube onde treinou e por época desportiva.

```
create procedure alertas_treinador(idStaff number)
is
    idEquipa equipa_estadio.id_equipa%type;
    numJ number := 0;
    numV number := 0;
    numD number := 0;
    numGols number := 0;
    nomStaff staff.nome%type;
begin
    select nome into nomStaff
    from staff
    where id_staff = idStaff;

    select equipa_estadio_id_equipa into idEquipa
    from equipa_estadio_staff
    where staff_id_staff = idStaff;

    select numvitorias into numV
    from liga_equipa_estadio
    where equipa_estadio_id_equipa = idEquipa;

    select numderrotas into numD
    from liga_equipa_estadio
    where equipa_estadio_id_equipa = idEquipa;

    select numempates into numJ
    from liga_equipa_estadio
    where equipa_estadio_id_equipa = idEquipa;

    -- numJ tem num de empates... + Vitorias + derrotas = total jogos...
    numJ := numJ + numV + numD;

    select count(jj.jogo_id_jogo) into numGols
    from jogador_jogo jj, jogador_equipa_estadio jee
    where jee.jogador_id_jogador = jj.jogador_id_jogador and jee.equipa_estadio_id_equipa = idEquipa;

    insert into avisos(mensagem) values ('Treinador: ' || nomStaff || ' Número de jogos: ' || numJ || ' Número de vitórias: ' || numV || ' Número de derrotas: ' || numD || ' Número de golos: ' || numGols);
exception
    when others then
        raise_application_error(SQLCODE, SQLERRM);
end;
/
```

Exercício M:

Crie a função tempo_medio_venda que recebe como argumento o id de uma equipa, e retorna a média, em minutos, que essa equipa necessitou para marcar o primeiro golo com o penúltimo treinador. Considere apenas jogos em casa e que não tenham cartões vermelhos associados nesse jogo.

```
create or replace function tempo_medio_venda(idEquipa number) return number
is
    EquipEx number := 0;

    media number := 0;
    cont number := 0;

    cursor c1 is select nvl(min(jj.minuto_marc), 0) mini, j.id_jogo jogi
    from jogo j, jogador_jogo jj, jogador_equipa_estadio jee
    where j.id_jogo = jj.JOGO_ID_JOGO and jj.jogador_id_jogador = jee.jogador_id_jogador and
    j.equipa_estadio_id_equipa = idEquipa and jee.equipa_estadio_id_equipa = idEquipa
    group by j.id_jogo;
begin
    select count(*) into EquipEx
    from equipa_estadio
    where id_equipa = idEquipa;

    if (EquipEx = 0) then
        raise_application_error(-20106, 'Nome/ID ' || idEquipa || ' de equipa inválido.');
    end if;

    for jog in c1
    loop
        declare
            testvar sancao_disc.cartao%type;
        begin
            select cartao into testvar
            from sancao_disc
            where jogo_id_jogo = jog.jogi;

            if (testvar <> 'Vermelho') then
                media := media + jog.mini;
                cont := cont + 1;
            end if;

        exception
            when no_data_found then
            -- não existe sanções neste jogo...
                media := media + jog.mini;
                cont := cont + 1;
        end;
    end loop;

    return media / cont;
end;
/
```

Exercício N:

Procedimento que adiciona um golo num jogo, num determinado minuto.

```
create procedure add_golo(idJogador number, idJogo number, minuto number)
is

begin
    declare
        tesvar number;
    begin
        select id_jogador into tesvar
        from jogador
        where idJogador = id_jogador;
    exception
        when no_data_found then
            raise_application_error(-20113, 'ID ' || idJogador || ' do jogador inválido.');
    end;

    declare
        tesvar number;
    begin
        select id_jogo into tesvar
        from jogo
        where id_jogo = idJogo;
    exception
        when no_data_found then
            raise_application_error(-20110, 'Não existem jogos.');
    end;

    if (minuto < 0 or minuto > 200) then
        raise_application_error(-20112, 'Minutos inválidos.');
    end if;

    insert into jogador_jogo values(idJogador, idJogo, minuto);
end;
/
```

Exercício P:

Trigger que atualiza os dados do jogador quando este marca um golo.

```
create or replace trigger add_golos_jog
before insert
on jogador_jogo
for each row
declare

    EquipGol number;
    EquipD number;

    GR number;

begin
    select jee.equipa_estadio_id_equipa into EquipGol
    from jogador_equipa_estadio jee
    where jee.jogador_id_jogador = :new.jogador_id_jogador;

    select j.equipa_estadio_id_equipal into EquipD
    from jogo j
    where j.id_jogo = :new.jogo_id_jogo;

    if (EquipD = EquipGol) then
        select j.equipa_estadio_id_equipa into EquipD
        from jogo j
        where j.id_jogo = :new.jogo_id_jogo;
    end if;

    update jogador set golos = golos + 1 where :new.jogador_id_jogador = id_jogador;

    select jee.jogador_id_jogador into GR
    from jogador_equipa_estadio jee, jogador jog
    where jee.equipa_estadio_id_equipa = EquipD and jee.jogador_id_jogador = jog.id_jogador and LOWER(jog.posicao) = 'guarda-redes';

    update jogador set golos_suf = golos_suf + 1 where GR = id_jogador;
end;
/
```

5. Conclusão

A realização deste projeto permitiu consolidar diversas competências nomeadamente: a criação de tabelas, a criação de vistas, a criação de procedimentos, a criação de triggers e a criação de funções.

As principais dificuldades existentes no desenvolvimento do projeto foi a implementação da base de dados de maneira a conseguir satisfazer todas os objetivos pretendidos. Foram surgindo algumas dificuldades que foram superadas com a ajuda dos professores da disciplina.