Universidade Federal de Mato Grosso
Campus Universitário do Araguaia

Trabalho de povoamento e consultas no banco de dados da Copa de 2014

Grupo:
Joathan Mareto
Lucas Ramos

Barra do Garças,
2021

1. Introdução

Para produzir o banco de dados, foi criado um arquivo sql para originar as tabelas e suas respectivas colunas. Todo o banco foi criado com base nas informações disponibilizadas pelo professor orientador da disciplina.

Devido a grande quantidade de dados que compõem o banco de dados da copa de 2014, foi necessário desenvolver um software para inserir as informações no banco. Esse programa em questão foi escrito utilizando a linguagem de programação JavaScript.

Todo o desenvolvimento até aqui foi pensado para o SGBD (Sistema de gerenciamento de banco de dados) MySQL. Com isso, todas as consultas solicitadas pelo professor foram escritas em sql, a fim então, de realizar buscas no banco para tornar o resultado da execução mais ativo para o trabalho.

2. Desenvolvimento

Esta foi a primeira etapa no processo de criação do banco de dados. Criamos um arquivo SQL (copa.sql) contendo a criação do banco, chamado copa, e a criação de todas as cinco tabelas sendo elas, *pais*, *jogadores*, *resultados_jogos*, *cartoes_jogadores*, *gols_assistencias_jogadores* e cada uma de suas colunas. Além disso, optamos por usar o *drop database copa* antes de qualquer comando, no arquivo SQL em questão, pois assim não precisamos escrever toda vez que quisermos remover o banco.

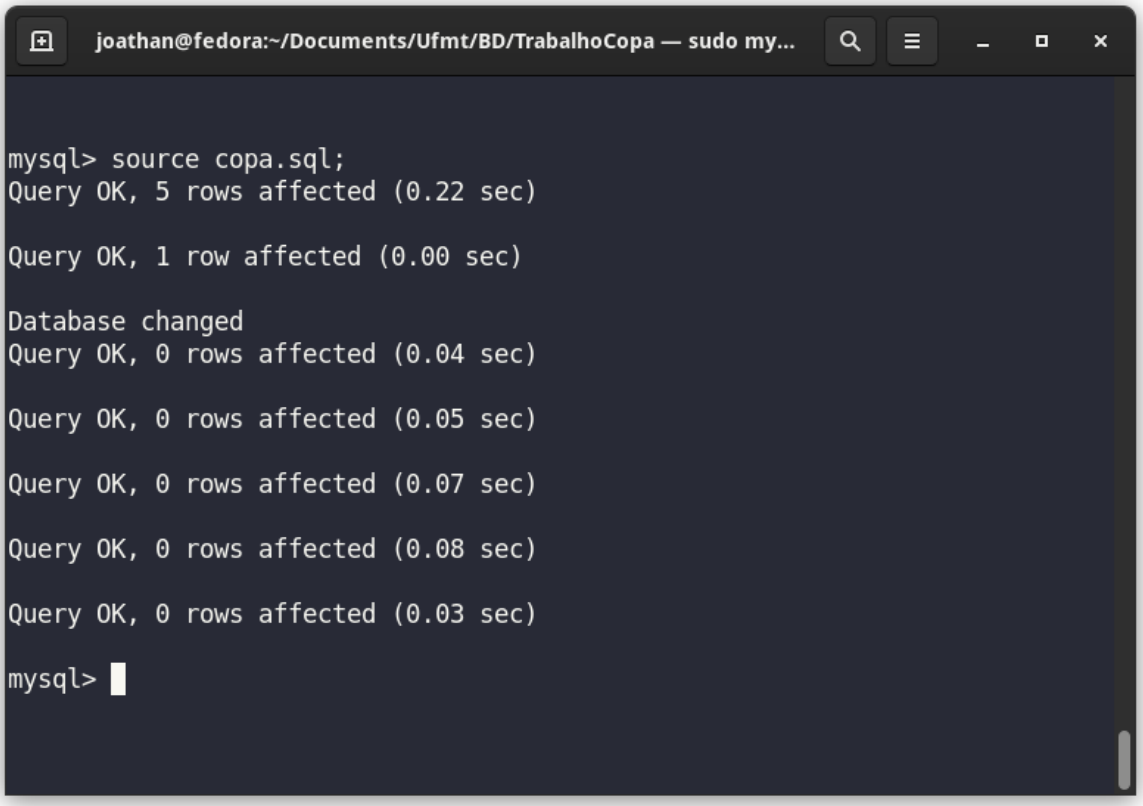


Figura 1 - Inclusão do arquivo SQL no SGBD

Para a segunda etapa utilizamos o código feito em JavaScript para inserção de dados. Nessa primeira inserção colocamos os países que participaram da copa onde continha as colunas de *nome_do_pais, populacao, numero_de_vitoria_em_copas, e tecnico.* A figura abaixo mostra os dados sendo inseridos no arquivo sql, logo, como podemos analisar também a forma de leitura de dados para inserção no banco de dados. O código *importPais.js* para povoamento de dados nos oferece também a quantidade de linhas e informações extras a respeito do que foi lido do arquivo de leitura, para este propósito, nos favorece em evitar possíveis erros para a criação de todo banco de dados.

É importante ressaltar que houve alteração dos arquivos CSV, onde neles, todas as linhas continha aspas simples para representar os campos de dados, e com isso, pensamos em retirar para padronizar todos os dados, pois, se mantivéssemos essas aspas os campos que contém números seria necessário alterar a lógica de todo código escrito em JavaScript, para que continuasse dando certo na parte do SQL. Contudo, o campo do tipo *booleano* no arquivo csv também sofreu alterações, onde, *false* e *true* foi alterado, para *false* 0 e o mesmo para *true* foi colocado 1 no lugar.

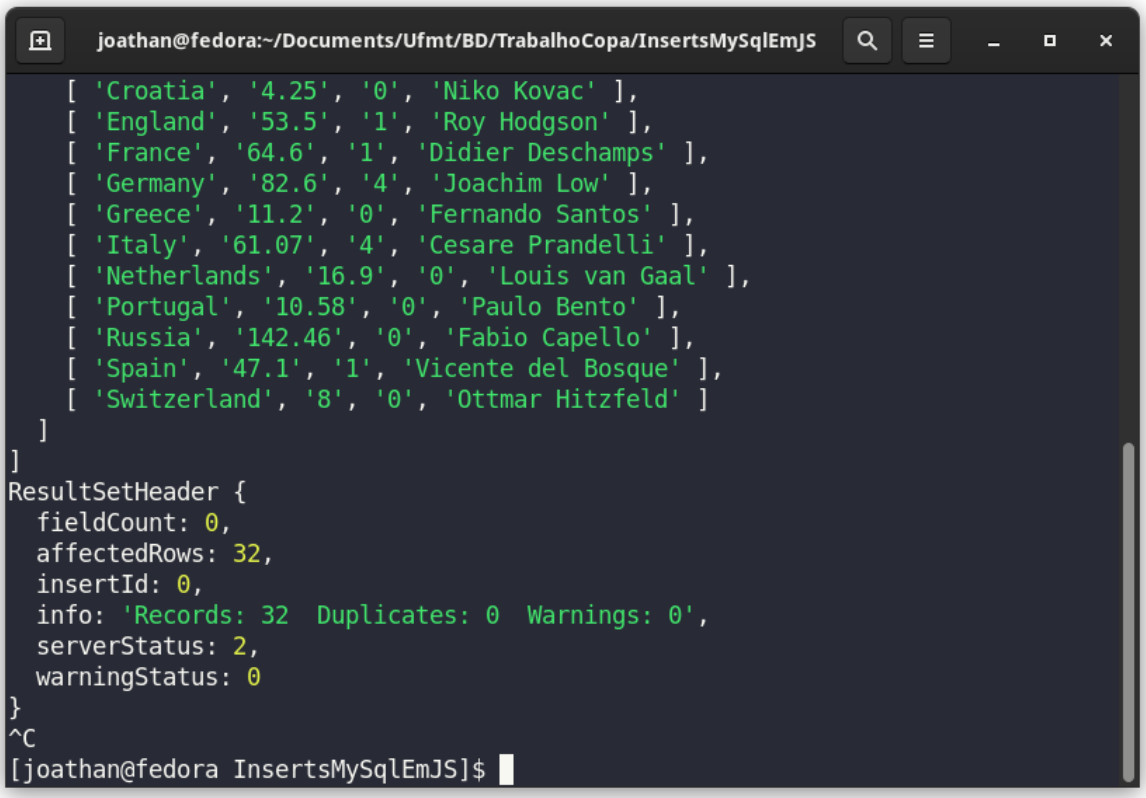


Figura 2 - Inserção dos países no banco utilizando JavaScript

Depois de inserir tudo no banco de dados, tínhamos que garantir que todos os dados estavam lá. Primeiro fizemos a consulta dos Países inseridos. Pudemos perceber que, todas as 32 linhas informadas, pelo script em JavaScript (importPais.js), realmente estão lá.

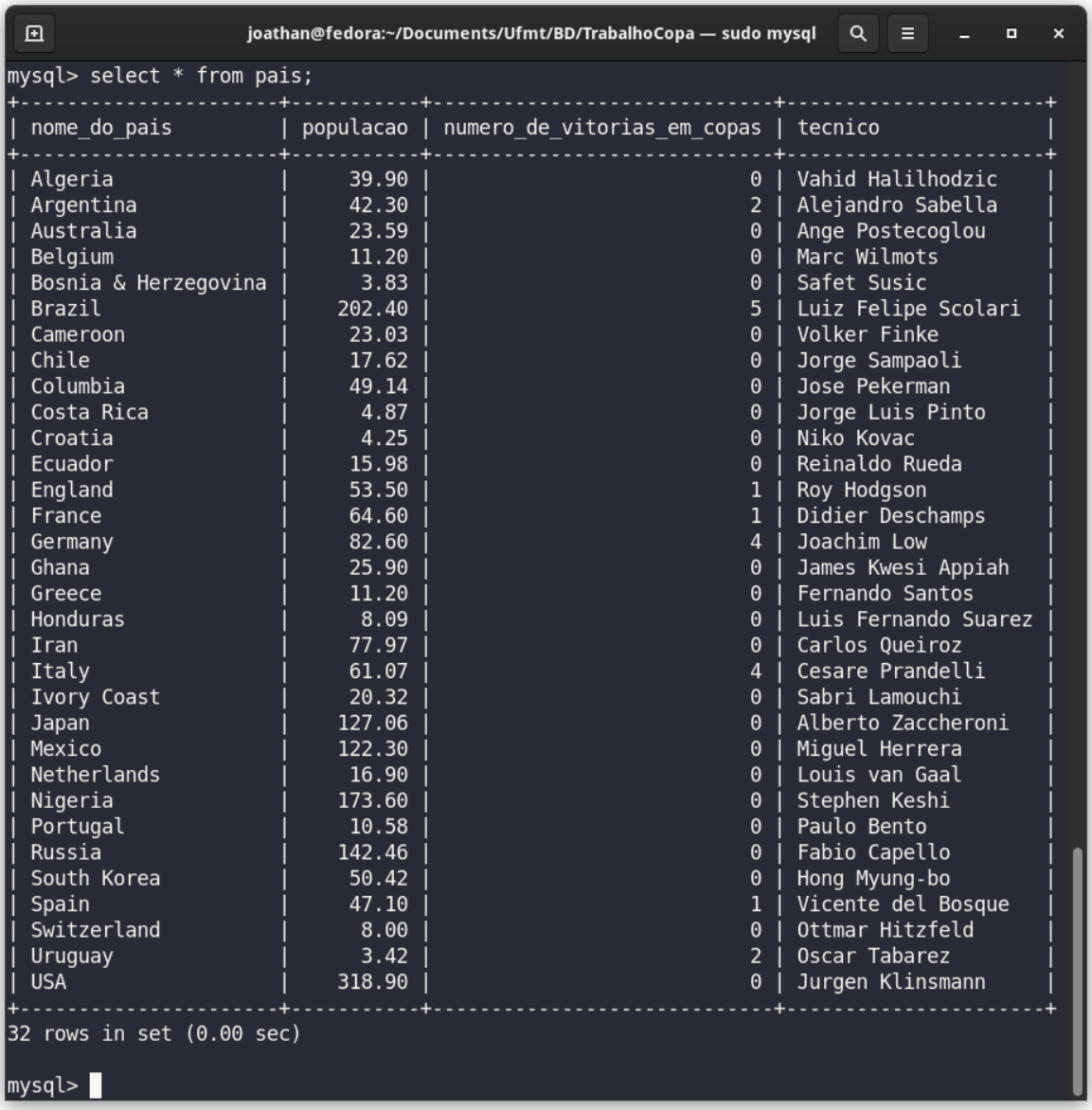
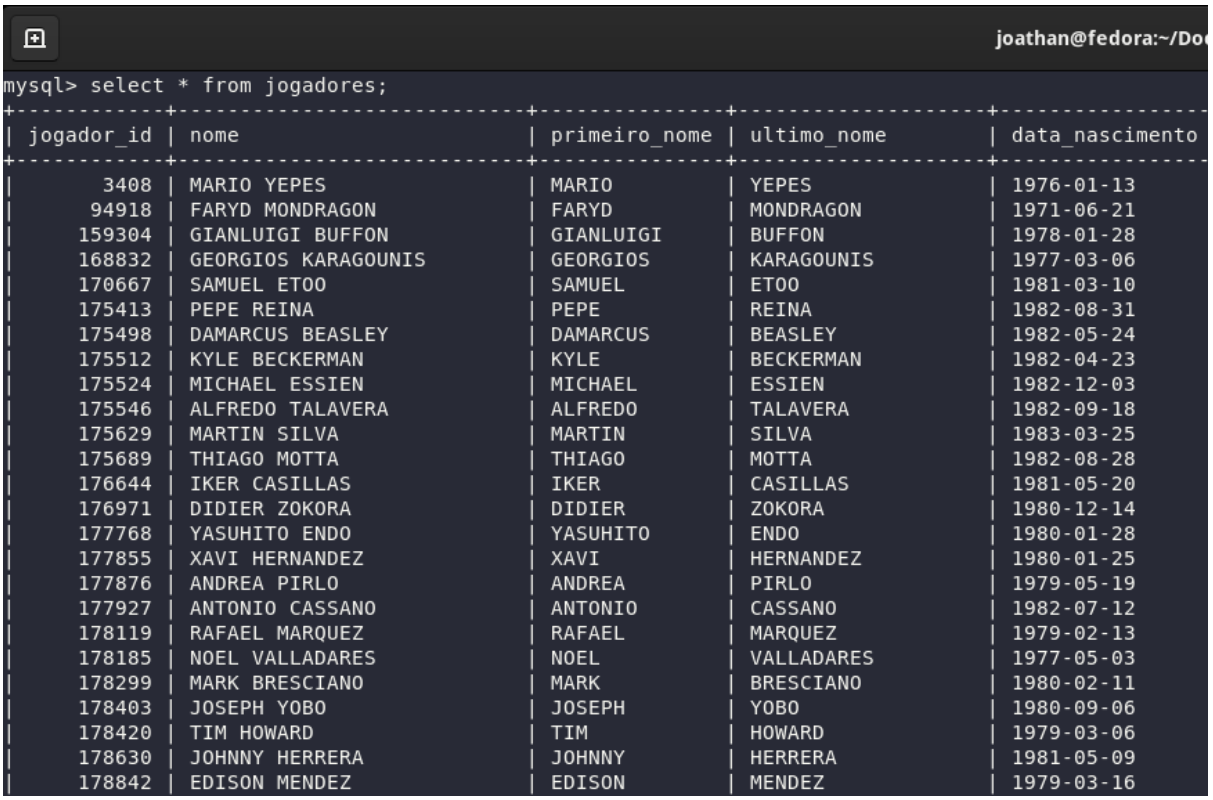


Figura 3 - Consulta de todos os países no banco

As imagens abaixo correspondentes a consulta de jogadores e resultados de jogos estão cortadas devido a quantidade de dados que continha na consulta do comando sql, portanto, colocamos a imagem completa no arquivo que compõem todos os dados do trabalho.

Após a execução do script *importJogadores.js*, realizamos a consulta dos dados dos jogadores, com o comando *select * from jogadores* para obter todos os jogadores da copa.

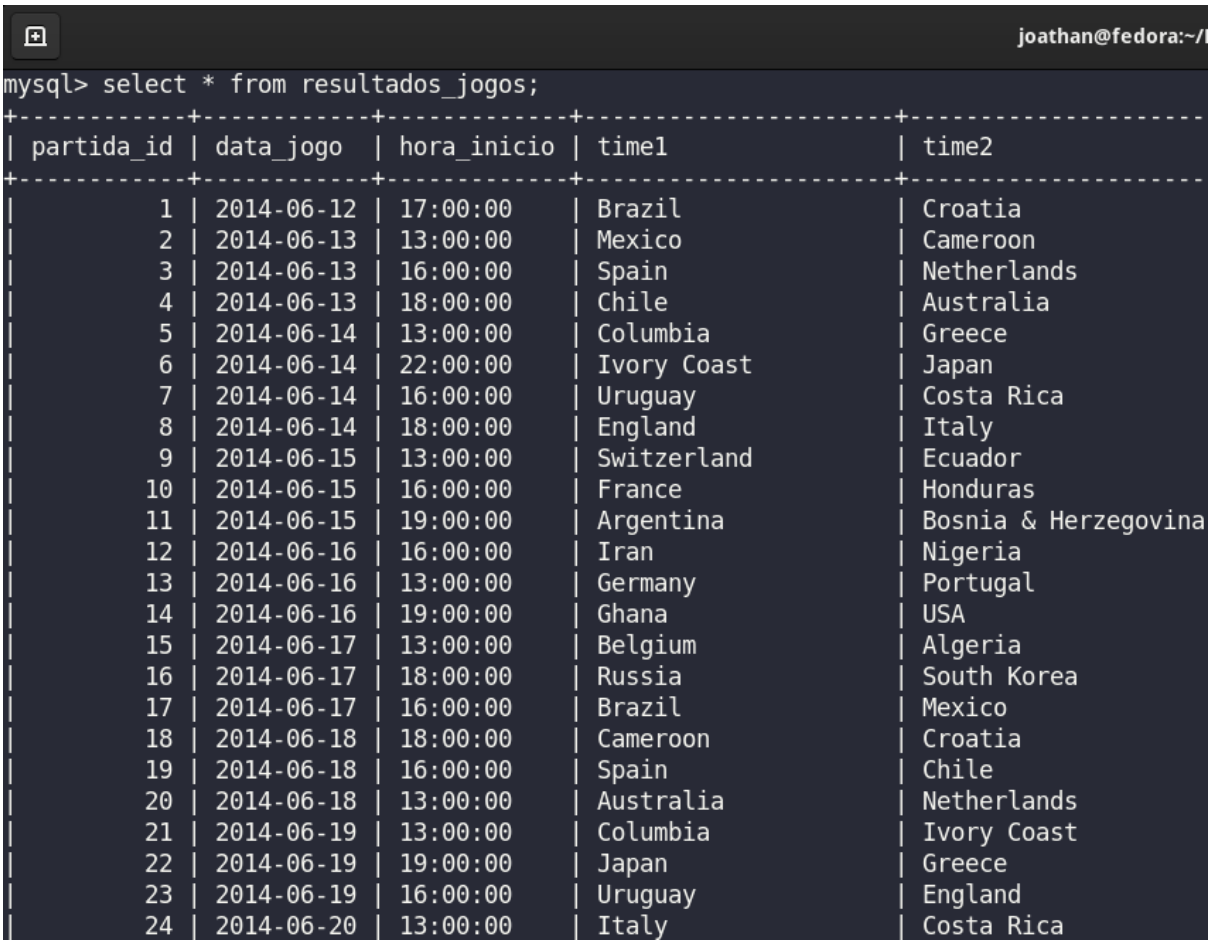


joathan@fedora:~/Do

```
mysql> select * from jogadores;
+------------+---------------------------+--------------+--------------------+-----------------
| jogador_id | nome                      | primeiro_nome | ultimo_nome        | data_nascimento
+------------+---------------------------+--------------+--------------------+-----------------
|       3408 | MARIO YEPES               | MARIO         | YEPES              | 1976-01-13
|      94918 | FARYD MONDRAGON           | FARYD         | MONDRAGON          | 1971-06-21
|     159304 | GIANLUIGI BUFFON          | GIANLUIGI     | BUFFON             | 1978-01-28
|     168832 | GEORGIOS KARAGOUNIS       | GEORGIOS      | KARAGOUNIS         | 1977-03-06
|     170667 | SAMUEL ETOO               | SAMUEL        | ETOO               | 1981-03-10
|     175413 | PEPE REINA                | PEPE          | REINA              | 1982-08-31
|     175498 | DAMARCUS BEASLEY          | DAMARCUS      | BEASLEY            | 1982-05-24
|     175512 | KYLE BECKERMAN            | KYLE          | BECKERMAN          | 1982-04-23
|     175524 | MICHAEL ESSIEN            | MICHAEL       | ESSIEN             | 1982-12-03
|     175546 | ALFREDO TALAVERA          | ALFREDO       | TALAVERA           | 1982-09-18
|     175629 | MARTIN SILVA              | MARTIN        | SILVA              | 1983-03-25
|     175689 | THIAGO MOTTA              | THIAGO        | MOTTA              | 1982-08-28
|     176644 | IKER CASILLAS             | IKER          | CASILLAS           | 1981-05-20
|     176971 | DIDIER ZOKORA             | DIDIER        | ZOKORA             | 1980-12-14
|     177768 | YASUHITO ENDO             | YASUHITO      | ENDO               | 1980-01-28
|     177855 | XAVI HERNANDEZ            | XAVI          | HERNANDEZ          | 1980-01-25
|     177876 | ANDREA PIRLO              | ANDREA        | PIRLO              | 1979-05-19
|     177927 | ANTONIO CASSANO           | ANTONIO       | CASSANO            | 1982-07-12
|     178119 | RAFAEL MARQUEZ            | RAFAEL        | MARQUEZ            | 1979-02-13
|     178185 | NOEL VALLADARES           | NOEL          | VALLADARES         | 1977-05-03
|     178299 | MARK BRESCIANO            | MARK          | BRESCIANO          | 1980-02-11
|     178403 | JOSEPH YOBO               | JOSEPH        | YOBO               | 1980-09-06
|     178420 | TIM HOWARD                | TIM           | HOWARD             | 1979-03-06
|     178630 | JOHNNY HERRERA            | JOHNNY        | HERRERA            | 1981-05-09
|     178842 | EDISON MENDEZ             | EDISON        | MENDEZ             | 1979-03-16
```

Figura 4 - Consulta dos jogadores

Nessa parte após povoar os resultados dos jogos da copa, fizemos a amostragem destes dados com o SQL.



```
joathan@fedora:~/
mysql> select * from resultados_jogos;
+------------+------------+-------------+----------------------+--------------------
| partida_id | data_jogo  | hora_inicio | time1                | time2
+------------+------------+-------------+----------------------+--------------------
|          1 | 2014-06-12 | 17:00:00    | Brazil               | Croatia
|          2 | 2014-06-13 | 13:00:00    | Mexico               | Cameroon
|          3 | 2014-06-13 | 16:00:00    | Spain                | Netherlands
|          4 | 2014-06-13 | 18:00:00    | Chile                | Australia
|          5 | 2014-06-14 | 13:00:00    | Columbia             | Greece
|          6 | 2014-06-14 | 22:00:00    | Ivory Coast          | Japan
|          7 | 2014-06-14 | 16:00:00    | Uruguay              | Costa Rica
|          8 | 2014-06-14 | 18:00:00    | England              | Italy
|          9 | 2014-06-15 | 13:00:00    | Switzerland          | Ecuador
|         10 | 2014-06-15 | 16:00:00    | France               | Honduras
|         11 | 2014-06-15 | 19:00:00    | Argentina            | Bosnia & Herzegovina
|         12 | 2014-06-16 | 16:00:00    | Iran                 | Nigeria
|         13 | 2014-06-16 | 13:00:00    | Germany              | Portugal
|         14 | 2014-06-16 | 19:00:00    | Ghana                | USA
|         15 | 2014-06-17 | 13:00:00    | Belgium              | Algeria
|         16 | 2014-06-17 | 18:00:00    | Russia               | South Korea
|         17 | 2014-06-17 | 16:00:00    | Brazil               | Mexico
|         18 | 2014-06-18 | 18:00:00    | Cameroon             | Croatia
|         19 | 2014-06-18 | 16:00:00    | Spain                | Chile
|         20 | 2014-06-18 | 13:00:00    | Australia            | Netherlands
|         21 | 2014-06-19 | 13:00:00    | Columbia             | Ivory Coast
|         22 | 2014-06-19 | 19:00:00    | Japan                | Greece
|         23 | 2014-06-19 | 16:00:00    | Uruguay              | England
|         24 | 2014-06-20 | 13:00:00    | Italy                | Costa Rica
```

Figura 5 - Consulta dos resultados dos jogos

Exibição de todos a tabela de cartões. Após a inserção pelo script de JS pudemos ver os dados solicitados

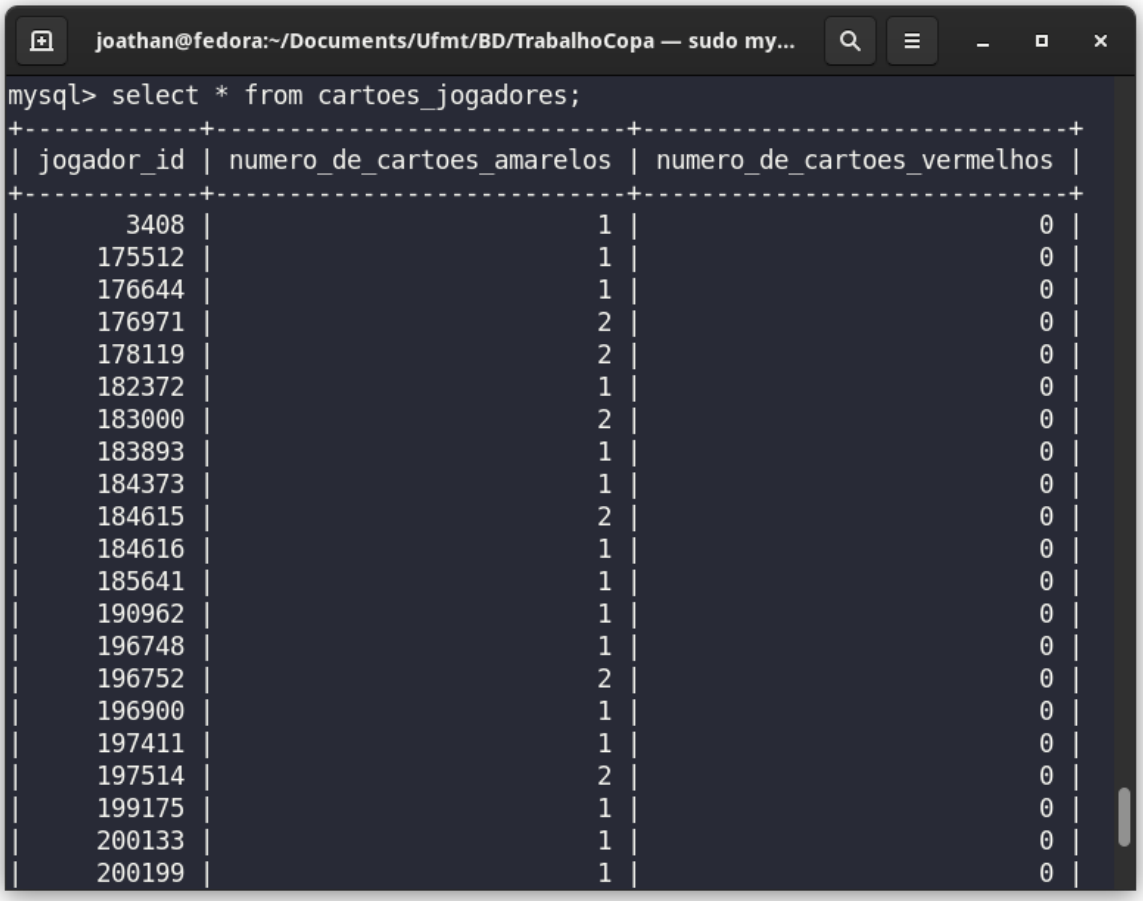


Figura 6 - Consulta dos cartões dos jogadores

Para a última inserção de dados, esta é a exibição de algumas assistências de gols dos jogadores da copa.

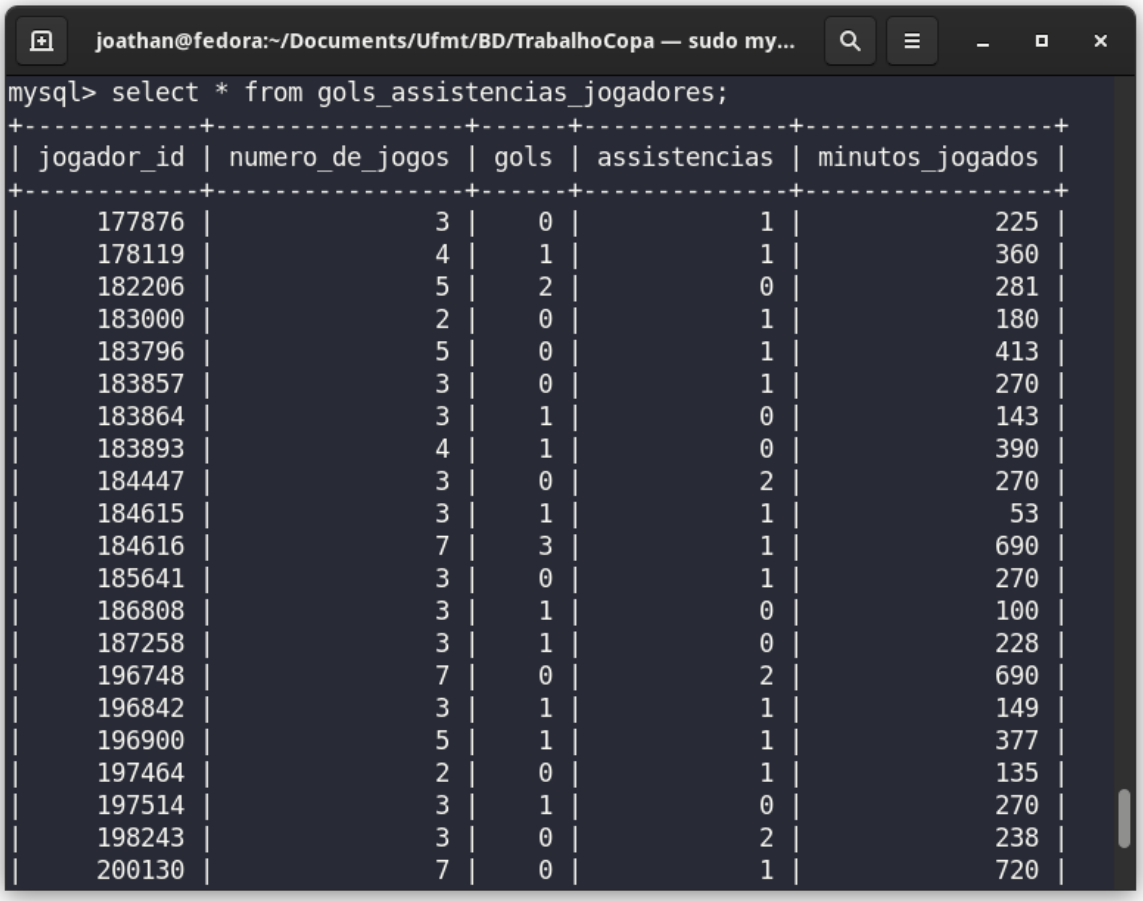


Figura 7 - Consulta dos gols e assistências dos jogadores

4. Consultas Solicitadas

Nesta etapa a seguir vamos expor as questões pedidas no trabalho e logo abaixo o código, por vias de tornar o conteúdo mais didático vamos colocar o código e o resultado em um outro arquivo contendo todas as linhas e os resultados, logo, se colocasse aqui teríamos grandes resultados em algumas questões pedidas no trabalho, o que tornaria o trabalho bem extenso e que possa dificultar a compreensão. Porém, vamos explicar o que se pede em tais questões e da mesma forma realizar a explicação por meio de como pensamos para tornar possível a resposta de cada pergunta levantada no trabalho.

1 - Recupere o nome, a posição e o clube dos jogadores do país 'USA'.

Portanto, para a primeira questão que era apenas realizar a busca recuperando o nome dos jogadores da copa do país USA, juntamente com os dados de posição e o clube dos jogadores, utilizamos para tal o cláusula *where* que visa filtrar uma condição de busca extraindo apenas os registros que atendem a condição pedida. Neste caso, selecionamos o nome, a posição e o clube filtrando apenas pelo país USA.

select nome, posicao, clube from jogadores where pais="USA";

2 - Recupere os nomes dos países participantes da copa do mundo de 2014 que ganharam a copa do mundo pelo menos uma vez.

Como explicado a questão de funcionamento da cláusula *where,* aqui vamos seguir a mesma linha de raciocínio, porém, vamos acrescentar na filtragem o número de vitórias que os países contém no mínimo uma vez, com isso, basta selecionar a coluna da tabela desejada e desfrutar da função matemática responsável por comparar se o valor é maior, ou igual.

select nome_do_pais from pais where numero_de_vitorias_em_copas>=1;

3 - Recupere os nomes dos países participantes da copa do mundo de 2014 que nunca ganharam uma copa do mundo.

Assim feito na questão passada aqui apenas requisitamos o número de vitória dos países que nunca ganharam uma copa, ou seja, realizar uma filtragem usando o *where* com a coluna de número de vitoria igual a zero.

select nome_do_pais from pais where numero_de_vitorias_em_copas=0;

4 - Recupere o nome e o país do jogador com o maior número de cartões amarelos na copa do mundo de 2014.

Aqui nesta etapa deparamos com um conteúdo novo das demais questões explicadas, para recuperar o nome e o país do jogador esses dois dados não estão armazenados na mesma tabela, o que implicaria em levar outro caminho para responder a atividade proposta, uma das formas de se fazer essa consulta seria realizar a junção de duas tabelas para conseguir recuperar o nome e o país. Com isso, usamos a cláusula *join,* mais especificamente a inner join, existem outras cláusulas como a *left join, right join, full join* e a que vamos utilizar aqui é a *inner join;* A cláusula *Inner Join* corresponde a uma operação de junção em álgebra relacional, ou seja, com essa cláusula conseguimos combinar a tabela *cartoes_jogadores* com a tabela de *jogadores,* essa junção pode ser feita quando ambas as tabelas de interesse mantém dados que compõem nas duas, nesse caso utilizamos o id que aparece em ambas tabelas, após a junção utilizamos a where para filtrar apenas os jogadores com cartões amarelos pedido na questão.

select j.nome, j.pais from cartoes_jogadores inner join jogadores j on cartoes_jogadores.jogador_id = j.jogador_id where numero_de_cartoes_amarelos = (select max(numero_de_cartoes_amarelos) from cartoes_jogadores);

5 - Para cada cidade sede, recupere a cidade sede e o número total de partidas que nela foram disputadas.

A estratégia usada foi utilizar a função *count*, com essa função retornamos o número de linhas que compõem a coluna especificada da tabela, ou como preferir utilizar a função, por assim dizer, para recuperar o número de partidas a instrução *count* proporcionou o resultado esperado e para ficar mais didático renomeamos a coluna utilizando o *as (apelido)* para *numero_total_partidas*. Portanto, seguindo a linha de raciocínio e para funcionar como esperado agrupamos os dados usufruindo da instrução *group by*, essa cláusula ainda não falada aqui serve para agrupar dados pela semelhança delas, o propósito de usá-la foi agrupar os dados em torno da coluna *cidade_sede* da tabela *resultados_jogos,* se não tivessemos agrupadas o retorno seria dado apenas a soma de linhas das partidas e o que temos objetivo seria agrupar os dados para exibir o número total mas se baseando no grupo que fizemos, ou seja, assim retorna os números totais de cada país que contém no grupo especificado. Para fins de utilização do agrupamento é necessário utilizar quando precisamos aplicar alguma função em algum grupo, para exemplo de funções temos algumas como exemplo count, avg, e sum.

```
select c.cidade_sede, count(c.cidade_sede) as numero_total_partidas from
resultados_jogos c group by c.cidade_sede;
```

6 - Para cada país, recupere o nome do país e o número de jogos que ele disputou como Time1 na tabela RESULTADOS_JOGOS, bem como o total de gols marcados (Soma de Gols_time1) e gols tomados (SUM de Gols_time2).

Como falado na questão anterior sobre o uso de agrupamento e os exemplos das funções que podem ser utilizadas com o *group by*, aqui recorremos ao uso do *sum,* a função *sum* apresentada retorna a soma total de uma coluna do tipo numérico. Logo, dado o funcionamento que utilizamos desta para somar os gols dos times da copa da tabela de *resultados_jogos*, vale lembrar que só foi possível com a finalidade do uso de agrupamento do *time1*.

select time1 as nome_do_pais, count(t.time1) as numero_de_jogos_pelo_pais, sum(t.gols_time1) as numero_gols_marcados_time1, sum(t.gols_time2) as gols_tomados_do_time2 from resultados_jogos t group by t.time1;

7 - Para cada país, recupere o nome do país e o número de jogos que ele disputou como Time2 na tabela RESULTADOS_JOGOS, bem como o total de gols marcados (Soma de Gols_time2) e gols tomados (SUM de Gols_time1).

De modo apresentado na questão acima, a alternativa aqui foi alterar para o time 2 e agrupar usando o mesmo.

select time2 as nome_do_pais, count(t.time2) as numero_de_jogos_pelo_pais, sum(t.gols_time2) as numero_gols_marcados_time2, sum(t.gols_time1) as gols_tomados_do_time1 from resultados_jogos t group by t.time2;

8 - Escreva uma consulta para obter o número total de partidas que cada país disputou (seja como Time1 ou como Time2), o número total de gols marcados e o número total de gols tomados. Crie uma view chamada SUMÁRIO_TIMES com os seguintes atributos para manter o resultado da consulta: NomePaís, NoDeJogos, TotalGolsMarcados, TotalGolsTomados. A saída deve estar ordenada em ordem decrescente do número de jogos disputados.

Para ficar mais claro o uso da view antes vou dar uma breve explicação sobre o uso dela, a view então pode ser definida como uma tabela virtual, com isso, pode ser composta por colunas e linhas de dados oriundos de tabelas relacionadas por uma query, ou seja, por exemplo um grupo de select, como foi o caso da solução da questão abaixo. Portanto, a view pode vir de outras views ou de tabelas o que proporciona diversas vantagens como uma delas a simplificação de códigos, reuso entre outros, e podemos notar bem claro a vantagem nessa solução da questão, onde, o código é extenso mas fica dentro da view, logo, quando vamos utilizar precisamos apenas chamar a view a fim de poupar menos trabalho na hora de codificar.

Agora com a explicação do uso e como funciona um pouco sobre a view vamos para a explicação do código escrito, criamos uma view com o nome proposto *sumário_times*, nessa view vamos obter três colunas, o *NomePaís*, a soma do número de jogos apelidada por

NoDeJogos, soma do total de gols marcados, apelidada também por *TotalGolsMarcados* e por fim o total de gols tomados, apelidada por *TotalGolsTomados*, com isso, utilizamos como estratégia o uso de da junção de duas tabelas criadas, primeiro criamos a tabela com o resultado referente ao time 1, como podemos observar o *group by time1*, logo após, fizemos o mesmo processo porém obtendo os dados do time 2, essa união foi feita pelo operador *union* que combina resultados de uma ou mais queries. Para finalizar, após a união de duas tabelas ficamos com uma "terceira" tabela que nesse caso será a junção das duas com os dados pedidos na questão, como a soma de número de jogos ordenado por ordem decrescente. Com essa terceira tabela conseguimos obter o resultado final que queríamos para a questão.

```
create view SUMÁRIO_TIMES as ( select NomePaís, sum(NoDeJogos) as
NoDeJogos, sum(TotalGolsMarcados) as TotalGolsMarcados,
sum(TotalGolsTomados) as TotalGolsTomados from ( select time1 as NomePaís,
count(*) as NoDeJogos, sum(gols_time1) as TotalGolsMarcados, sum(gols_time2)
as TotalGolsTomados from resultados_jogos group by time1 union select time2 as
NomePaís, count(*) as NoDeJogos, sum(gols_time2) as TotalGolsMarcados,
sum(gols_time1) as TotalGolsTomados from resultados_jogos group by time2) as
CAMPEONATO group by NomePaís order by sum(NoDeJogos) desc );
```

9 - Liste todas as partidas disputadas pelo país 'Brazil' como Time1 ou Time2.

Nessa, uma novidade das demais é o uso do *or (ou)*, ou seja, só referência mesmo a busca de primeiro time como Brazil ou como segundo time como Brazil, utilizamos assim conseguimos obter as partidas disputadas tanto como time 1 e como time 2 pelo Brazil.

```
select * from resultados_jogos where time1 = 'Brazil' or time2 = 'Brazil' ;
```

10 - Recupere os nomes dos jogadores que marcaram pelo menos um gol, o país desses jogadores, e o número de gols que cada um marcou. Ordene o resultado pelo número de gols marcados em ordem decrescente.

Visto as demais questões para essa utilizei o conhecimento juntado até então e recuperei os jogadores utilizando a união de tabelas com o *inner join* e filtrando com *where,*

assim, foi a forma mais fácil de chegar a resposta e só por final utilizando o order by para ordenar os resultados.

select j.nome, j.pais, gols_assistencias_jogadores.gols as num_de_gols_ordem_desc from gols_assistencias_jogadores inner join jogadores j on gols_assistencias_jogadores.jogador_id = j.jogador_id where gols >= 1 order by gols desc;

11 - Repita a consulta 10, mas somente para os jogadores que tiveram mais de 2 gols.

A única diferença foi no *where* no qual filtro utilizou o operador de maior para pegar os resultados dos jogadores acima de dois gols.

select j.nome, j.pais, gols_assistencias_jogadores.gols as num_de_gols_ordem_desc from gols_assistencias_jogadores inner join jogadores j on gols_assistencias_jogadores.jogador_id = j.jogador_id where gols > 2 order by gols desc;

12 - Liste os países participantes da copa de 2014 e a sua população, ordenando em ordem decrescente de população.

select nome_do_pais, populacao from pais order by populacao desc;

5. Restrição de Integridade

Primeira inserção: Inserindo, na tabela cartões_jogadores, uma tupla com jogador_id que não existe na tabela de jogadores.

insert into cartoes_jogadores values (999999, 10, 10);

```
ERROR 1452 (23000): Cannot add or update a child row:
a foreign key constraint fails (`copa`.`cartoes_jogadores`,
CONSTRAINT `cartoes_jogadores_ibfk_1` FOREIGN KEY
(`jogador_id`) REFERENCES `jogadores` (`jogador_id`))
```

Segunda inserção: Inserindo um valor que já existe na tabela dos países.

insert into pais values ('Brazil', 202.4 , 5, 'Luiz Felipe Scolari');

```
ERROR 1062 (23000): Duplicate entry
'Brazil' for key 'PRIMARY'
```

Terceira inserção: Inserção de uma nova tupla, com os cartões, amarelos e vermelhos, porém sem o identificador do jogador. O campo jogador_id da tabela em questão é *not null*.

insert into cartoes_jogadores (numero_de_cartoes_amarelos, numero_de_cartoes_vermelhos) values (10, 10);

```
ERROR 1364 (HY000): Field 'jogador_id' doesn't have a default
value
```

6. Restrição de Integridade Referencial

O SGBD não permite que algumas coisas sejam feitas. O exemplo abaixo trata-se de uma tentativa de deletar a tupla com o nome do país igual a *Brazil* da tabela dos países. O que acontece aqui é que existe outra tabela que faz referência à tabela *pais*, sendo ela *resultado_jogos*. Caso tivéssemos definido a chave estrangeira para deletar junto (on delete cascade) o erro a seguir não aconteceria, já que tanto a tupla com o nome do país igual a Brazil quanto a tupla que faz referência a ela seriam deletadas.

delete from pais where nome_do_pais='Brazil';

```
ERROR 1451 (23000): Cannot delete or update a parent row: a
foreign key constraint fails (`copa`.`jogadores`, CONSTRAINT
`jogadores_ibfk_1` FOREIGN KEY (`pais`) REFERENCES
`pais` (`nome_do_pais`))
```

7. Inserção sem Violação de Restrição

Cada inserção listada abaixo funcionou como esperado. Primeiro é inserido no banco mais um país para compor a lista de países da copa de 2014. A próxima inserção diz respeito a um dos jogadores da seleção de futebol masculino da Angola. Desconsiderando o ID do jogador que foi criado por nós, o restante dos dados é real, incluindo o país em questão. Por último o número de cartões vermelhos e amarelos referente a esse jogador recém adicionado. Como ele não participou da copa, já que o país Angola não esteve presente, decidimos colocar zero nos dois valores de cartão.

insert into pais values ('Angola', 31.83, 0, 'Pedro Gonçalves');

insert into jogadores values (111111, 'CARLOS FERNANDES', 'CARLOS', 'FERNANDES', '1979-12-8', 'Angola', 188, 'Moreirense FC', 'Goleiro', 37, 0);

insert into cartoes_jogadores values (111111, 0, 0);

8. Execução dos scripts

Já que utilizamos a linguagem JavaScript é necessário ter em sua máquina o Node.js, para assim ser possível executar os códigos em formato .js. A instalação é bem simples, porém vai depender de cada sistema operacional, vamos disponibilizar o site para que fique mais fácil a instalação em cada máquina. Segue o link para instalação mais detalhada do Node.js https://nodejs.org/en/download/.

Agora com sua máquina com suporte para rodar arquivos .js você pode executar com node "nome do arquivo", porém, vale ressaltar que para a parte de conexão de dados deve criar a conexão de acordo com seu usuário e senha do mysql. Essas alterações devem ser feitas em cada código de *InsertMysql.js*. Segue abaixo uma imagem onde deve realizar as alterações.

```
//Criando conexão com bd
const connection = mysql.createConnection({
  host: "localhost",
  user: "root",
  password: "ramos",
  database: "copa"
});
```