

O Campeonato Paulista de 2016 teve regras absolutamente esdrúxulas. Para se adequar a emissoras de tv, patrocinadores e jogadores, o campeonato foi organizado da seguinte maneira:

O "Paulistão" será disputado por 20 clubes, divididos em quatro grupos com cinco equipes cada. Os times que estão na mesma chave enfrentam apenas os clubes de outros grupos. No total, cada participante vai realizar 15 partidas na primeira fase. A primeira fase acontecerá a partir de 30 de janeiro. Os dois melhores classificados de cada chave avançam às quartas de final, que será disputada, entre eles, em jogo único - com o mando de campo ao clube de melhor campanha no somatório das fases anteriores. Em caso de empate no tempo regulamentar, o confronto será decidido através de pênaltis. A semifinal também será definida em apenas uma partida e com possibilidade de decisão por penais. Já a final acontecerá em dois jogos, e em caso de empate em pontos (uma vitória para cada time ou dois empates), o primeiro critério de desempate será o saldo de gols na fase final. Caso o empate persista, o confronto será decidido através de pênaltis. O gol marcado fora de casa não vale como critério de desempate.

Os quatro times que somarem menos pontos na primeira fase estarão rebaixados para a segunda divisão.

Fazer uma aplicação em Java (Web (Servlets) ou Desktop (Swing ou JavaFX)) com SQL Server para resolver os problemas, da seguinte maneira:

O sistema deve ter 3 tabelas principais:

- Times (Com todos os 20 times)(Não é necessário CRUD para ela)

Times (CodigoTime | NomeTime | Cidade | Estadio)

- Grupos

(Corinthians, Palmeiras, Santos e São Paulo NÃO PODEM estar no mesmo grupo)

(A coluna Grupo não pode aceitar nenhum valor diferente de A, B, C, D)

Grupos (Grupo | CodigoTime)

- Jogos

(A primeira fase ocorrerá em 15 datas seguidas, sempre rodada cheia (os 10 jogos), aos domingos e quartas)

Jogos (CodigoTimeA | CodigoTimeB | GolsTimeA | GolsTimeB | Data)

O sistema deve se comportar da seguinte maneira:

Uma tela deve chamar uma procedure que divide os times nos quatro grupos, preenchendo, aleatoriamente (com exceção da regra já exposta em Grupos).

Uma tela deve gerar as rodadas dos jogos, de acordo com as regras do campeonato, preenchendo a tabela jogos.

Lembre-se, cada rodada tem 10 jogos (todos os 20 times). Lembre-se também que, as rodadas vão acontecer de quarta e domingo, sucessivamente, sem pausas.

Uma tela deve mostrar 4 Tabelas com os 4 grupos formados.

Uma tela deve mostrar um Campo, onde o usuário digite a data e, em caso de ser uma data com rodada, mostre uma tabela com todos os jogos daquela rodada.

Tabela de Times: (Não é necessário fazer o CRUD)

Equipe	Cidade	Em 2015	Estádio	Capacidade	Títulos
Esporte Clube <b>Água Santa</b>	 Diadema	4° (A2)	Distrital do Inamar <sup>[3]</sup>	10 000	0
Grêmio Osasco <b>Audax</b>	 Osasco	9°	José Liberatti	17 780	0
<b>Botafogo</b> Futebol Clube	 Ribeirão Preto	7°	Santa Cruz	29 292	0
<b>Capivariano</b> Futebol Clube	 Capivari	14°	Arena Capivari	19 000	0
Sport Club <b>Corinthians</b> Paulista	 São Paulo	3°	Arena Corinthians	47 605	27 (último em 2013)
Associação <b>Ferrovária</b> de Esportes	 Araraquara	1° (A2)	Fonte Luminosa	20 000	0
<b>Ituano</b> Futebol Clube	 Itu	12°	Novelli Júnior	18 560	2 (último em 2014)
Clube Atlético <b>Linense</b>	 Lins	16°	Gilberto Siqueira Lopes	15 770	0
<b>Mogi Mirim</b> Esporte Clube	 Mogi Mirim	11°	Vail Chaves	19 900	0
Grêmio <b>Novorizontino</b>	 Novo Horizonte	2° (A2)	Jorge Ismael de Biasi	12 398	0
<b>Oeste</b> Futebol Clube	 Itápolis	3° (A2)	Amaros	10 000	0
Sociedade Esportiva <b>Palmeiras</b>	 São Paulo	2°	Allianz Parque	43 713	22 (último em 2008)
Associação Atlética <b>Ponte Preta</b>	 Campinas	5°	Moisés Lucarelli	19 728	0
<b>Red Bull Brasil</b>	 Campinas	6°	Moisés Lucarelli	19 728	0
<b>Rio Claro</b> Futebol Clube	 Rio Claro	15°	Augusto Schmidt Filho	6 284	0
<b>Santos</b> Futebol Clube	 Santos	1°	Vila Belmiro	16 798	22 (último em 2016)
Esporte Clube <b>São Bento</b>	 Sorocaba	10°	Walter Ribeiro	13 772	0
<b>São Bernardo</b> Futebol Clube	 São Bernardo do Campo	13°	Primeiro de Maio	15 789	0
<b>São Paulo</b> Futebol Clube	 São Paulo	4°	Morumbi	67 428	21 (último em 2005)
Esporte Clube <b>XV</b> de Novembro	 Piracicaba	8°	Barão de Serra Negra	18 000	0

Para conhecimento, sempre é possível utilizar *Alias* em consultas tipo JOIN em SQL.

Normalmente, usamos *Alias* no nome da coluna (Ex.: SELECT GETDATE() AS 'Hoje'). O "AS" determina o *Alias*.

No entanto, é possível se usar *Alias* no nome da tabela também. De acordo com a documentação Microsoft:

A legibilidade de uma instrução SELECT pode ser aprimorada, atribuindo um alias a uma tabela, que também é conhecido como nome de correlação ou variável de intervalo. Um alias de tabela pode ser atribuído com ou sem a palavra-chave AS:

- *table\_name AS table alias*
- *table\_name table\_alias*

No exemplo a seguir, o alias *c* é atribuído a *Customer* e o alias *s* é atribuído a *Store*.

```
USE AdventureWorks2008R2;  
GO  
SELECT c.CustomerID, s.Name  
FROM Sales.Customer AS c  
JOIN Sales.Store AS s  
ON c.CustomerID = s.BusinessEntityID ;
```

Se um alias for atribuído a uma tabela, todas as referências explícitas à tabela na instrução do Transact-SQL precisarão usar o alias; não o nome de tabela. Por exemplo, a instrução SELECT a seguir gera um erro de sintaxe porque utiliza o nome da tabela quando existe um alias atribuído:

```
SELECT Sales.Customer.CustomerID, /* Illegal reference to  
Sales.Customer. */  
       s.Name  
FROM Sales.Customer AS c  
JOIN Sales.Store AS s  
ON c.CustomerID = s.BusinessEntityID ;
```

Um exemplo da obrigatoriedade do uso de *Alias* no nome da tabela, em um Select com JOIN, é quando mais de uma coluna da consulta faz referência à mesma tabela e à mesma PK.

Exemplo:

```
create table pais(
id int identity not null primary key,
nome varchar(100))

create table filhos(
id int identity(1001,1) not null primary key,
nome varchar(100),
pai int,
mae int
foreign key (pai) references pais (id),
foreign key (mae) references pais (id))

select filhos.nome, pai.nome as 'Nome do pai', mae.nome as 'Nome da mãe'
from pais as pai
inner join filhos
on pai.id = filhos.pai
inner join pais as mae
on mae.id = filhos.mae
```

Tabela Pais			Tabela Filhos				Consulta com <i>Alias</i>				
	id	nome		id	nome	pai	mae		nome	Nome do pai	Nome da mãe
1	1	Pai 1	1	1001	Filho 1	1	2	1	Filho 1	Pai 1	Mae 1
2	2	Mae 1	2	1002	Filho 2	3	4	2	Filho 2	Pai 2	Mae 2
3	3	Pai 2	3	1003	Filho 3	5	6	3	Filho 3	Pai 3	Mae 3
4	4	Mae 2	4	1004	Filho 4	7	8	4	Filho 4	Pai 4	Mae 4
5	5	Pai 3	5	1005	Filho 5	9	10	5	Filho 5	Pai 5	Mae 5
6	6	Mae 3	6	1006	Filho 6	11	12	6	Filho 6	Pai 6	Mae 6
7	7	Pai 4	7	1007	Filho 7	13	14	7	Filho 7	Pai 7	Mae 7
8	8	Mae 4	8	1008	Filho 8	15	16	8	Filho 8	Pai 8	Mae 8
9	9	Pai 5	9	1009	Filho 9	17	18	9	Filho 9	Pai 9	Mae 9
10	10	Mae 5	10	1010	Filho 10	19	20	10	Filho 10	Pai 10	Mae 10
11	11	Pai 6									
12	12	Mae 6									
13	13	Pai 7									
14	14	Mae 7									
15	15	Pai 8									
16	16	Mae 8									
17	17	Pai 9									
18	18	Mae 9									
19	19	Pai 10									
20	20	Mae 10									

Verifique que demos o *alias* pai para a tabela pais e, também demos o *alias* mãe para a tabela pais. Com isso, conseguimos fazer um Inner Join com 2 colunas que referenciam a mesma tabela e a mesma PK.