TRABALHO PRÁTICO DE FUNDAMENTOS DE BANCO DE DADOS 2018/1

Integrantes: Gabriel Gomes, Otávio Mello, Ricardo Kunde

ETAPA 1- DADOS UTILIZADOS

Nome: Batalhas em *Game of Thrones*

Colunas	Descrição	Dado	Colunas	Descrição	Dado
name	Nome da Batalha	string	defender_4	Principal casa defensora 4	string
year	O ano da batalha	numérico	attacker_outcome	Resultado da perspectiva do atacante	string
battle_number	ID único da batalha	numérico	battle_type	Classificação do tipo de batalha	string
attacker_king	Reis atacantes	string	major_death	Principal personagem morto	string
defender_king	Reis defensores	string	major_capture	Principal personagem capturado	string
attacker_1	Principal casa atacante	string	attacker_size	Tamanho do exército do atacante	numérico
attacker_2	Principal casa atacante 2	string	defender_size	Tamanho do exército defensor	numérico
attacker_3	Principal casa atacante 3	string	attacker_commander	Comandantes do exército atacante	string
attacker_4	Principal casa atacante 4	string	defender_commander	Comandantes do exército defensor	string
defender_1	Principal casa defensora 1	string	summer	É verão?	numérico
defender_2	Principal casa defensora 2	string	location	Localização da batalha	string
defender_3	Principal casa defensora 3	string	região	Região da batalha	string
note	Observações	string			

ETAPA 2 - NORMALIZAÇÃO

Modelo desnormalizado ÑN

battles(name, year, battle_number, attacker_king, defender_king, attacker_1, attacker_2, attacker_3, attacker_4, defender_1, defender_2, defender_3, defender_4 , attacker_outcome, battle_type, major_death, major_capture, attacker_size, defender_size, attacker_commander, defender_commander, summer, location, region, note)

PRIMEIRA FORMA NORMAL

Modelo 1FN

- *"Diz-se que uma tabela está na primeira forma normal,* ***quando ela não contém tabelas aninhadas.****"*

- *"Todos os atributos* ***devem ser atômicos*** *ou seja a tabela* ***não deve conter grupos repetidos e nem atributos com mais de um valor.****"*

Durante essa etapa, na tabela, foram identificadas colunas com atributos multivalorados, assim desobedecendo a condição para estar na primeira forma normal.

battles(name, year, battle_number, attacker_king, defender_king, attacker_1, attacker_2, attacker_3, attacker_4, defender_1, defender_2, defender_3, defender_4 , attacker_outcome, battle_type, major_death, major_capture, attacker_size, defender_size, attacker_commander, defender_commander, summer, location, region, note)

COLUNAS COM ATRIBUTOS MULTIVALORADOS

attacker_commander	defender_commander
Jaime Lannister	Clement Piper, Vance
Gregor Clegane	Beric Dondarrion
Jaime Lannister, Andros Brax	Edmure Tully, Tytos Blackwood
Roose Bolton, Wylis Manderly, Medger Cerwyn, Harrion Karstark, Halys Hornwood	Tywin Lannister, Gregor Clegane, Kevan Lannister, Addam Marbrand
Robb Stark, Brynden Tully	Jaime Lannister
Robb Stark, Tytos Blackwood, Brynden Tully	Lord Andros Brax, Forley Prester
Gregor Clegane	Lyman Darry

attacker_king	defender_king
Joffrey/Tommen Baratheon	Robb Stark
Joffrey/Tommen Baratheon	Robb Stark
Joffrey/Tommen Baratheon	Robb Stark
Robb Stark	Joffrey/Tommen Baratheon
Robb Stark	Joffrey/Tommen Baratheon
Robb Stark	Joffrey/Tommen Baratheon

location
Golden Tooth
Mummer's Ford
Ryamsport, Vinetown, Starfish Harbor
Storm's End
Dragonstone

PRIMEIRA FORMA NORMAL

Solução adotada

battles(name, year, **battle_number**, , attacker_1, attacker_2, attacker_3, attacker_4, defender_1, defender_2, defender_3, defender_4, attacker_outcome, battle_type, major_death, major_capture, attacker_size, defender_size, summer, region, note)

attacker_king(**battle_number, id_king,** attacker_king)

battle_number referencia battles

defender_king(**battle_number, id_king,** defender_king)

battle_number referencia battles

attacker_commander(**battle_number, id_commander,** attacker_commander)

battle_number referencia battles

defender_commander(**battle_number, id_commander**, defender_commander)

battle_number referencia battles

location(**battle_number, id_location,** location)

battle_number referencia battles

SEGUNDA FORMA NORMAL

Modelo 2FN

- *"Uma tabela encontra-se na segunda forma normal, quando,* ***além de estar na 1FN, não contém dependências parciais.****"*
- *"Dependências parciais ocorrem* ***quando uma coluna depende apenas de parte de uma chave primária composta.****"*

battles(name, year, **battle_number**, , attacker_1, attacker_2, attacker_3, attacker_4, defender_1, defender_2, defender_3, defender_4, attacker_outcome, battle_type, major_death, major_capture, attacker_size, defender_size, summer, region, note)

Está na 2FN!

attacker_king(**battle_number**, **id_king**, attacker_king)

defender_king(**battle_number**, **id_king**, defender_king)

attacker_commander(**battle_number**, **id_commander**, attacker_commander)

defender_commander(**battle_number**, **id_commander**, defender_commander)

location(**battle_number**, **id_location**, location)

Não estão na 2FN!

SEGUNDA FORMA NORMAL

As tabelas abaixo geradas após a 1FN possuem dependência parcial.

attacker_king(**battle_number**, **id_king**, attacker_king)

id_king -> attacker_king

defender_king(**battle_number**, **id_king**, defender_king)

id_king -> defender_king

attacker_commander(**battle_number**, **id_commander**, attacker_commander)

id_commander -> attacker_commander

defender_commander(**battle_number**, **id_commander**, defender_commander)

id_commander -> defender_commander

location(**battle_number**, **id_location**, location)

id_location -> location

AJUSTES

attacker_king(**battle_number**, **id_king**, attacker_king)

king(**id_king**, king_name)

defender_king(**battle_number**, **id_king**, defender_king)

attacker_commander(**battle_number**, **id_commander**, attacker_commander)

commander(**id_commander**, commander_name)

defender_commander(**battle_number**, **id_commander**, defender_commander)

location(**battle_number**, **id_location**, location)

id_location -> location

location(**id_location**, location)

SEGUNDA FORMA NORMAL *Solução*

battles(name, year, **battle_number**, attacker_outcome, battle_type, major_death, major_capture, attacker_size, defender_size, summer,region, note)

king(**id_king**, king_name)

location(**id_location**, location)

commander(**id_comander**, commander)

attacker_king(**battle_number, id_king**)

battle_number referencia battles

id_king referencia king

defender_king(**battle_number, id_king**)

battle_number referencia battles

id_king referencia king

attacker_commander(**battle_number, id_commander**)

SEGUNDA FORMA NORMAL *Solução*

battle_number referencia battles

id_commander referencia commander

defender_commander(**battle_number, id_commander**)

battle_number referencia battles

id_commander referencia commander

location_battle(**battle_number**, id_location)

battle_number referencia battles

TERCEIRA FORMA NORMAL

Modelo 3FN

- *"Uma tabela encontra-se na terceira forma normal, quando* ***está na 1FN e na 2FN, e não contém dependências transitivas.****"*
- *"Dependências transitivas ocorrem* ***quando uma coluna que não seja chave primária depende de outra que também não seja.****"*

Year → Summer

Em uma observação inicial obteve-se a ideia de que Year determina Summer (analisando a tabela), porém, com uma pesquisa sobre como funcionam as estações no universo de Game of Thrones, descobriu-se que suas durações são imprevisíveis e, logo, um verão poderia durar décadas ou meses. Dessa forma, optou-se por não considera-la uma dependência transitiva.

TERCEIRA FORMA NORMAL

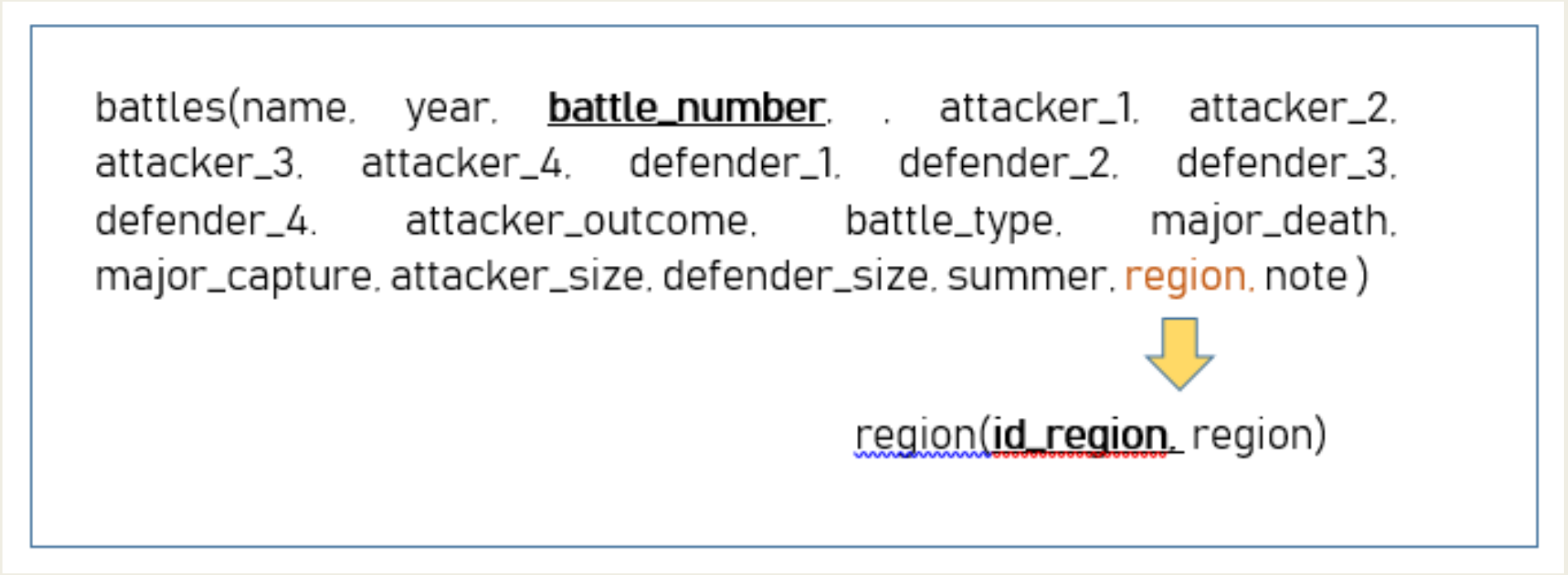
year	summer
298	1
298	1
298	1
298	1
298	1
299	1
299	1
299	1
299	1
299	1
299	1
299	1
299	1
300	0
300	0
300	0
300	0
300	0
300	0
300	0
300	0
300	0

No ano 298 será sempre Verão.

No ano 299 também será sempre Verão.

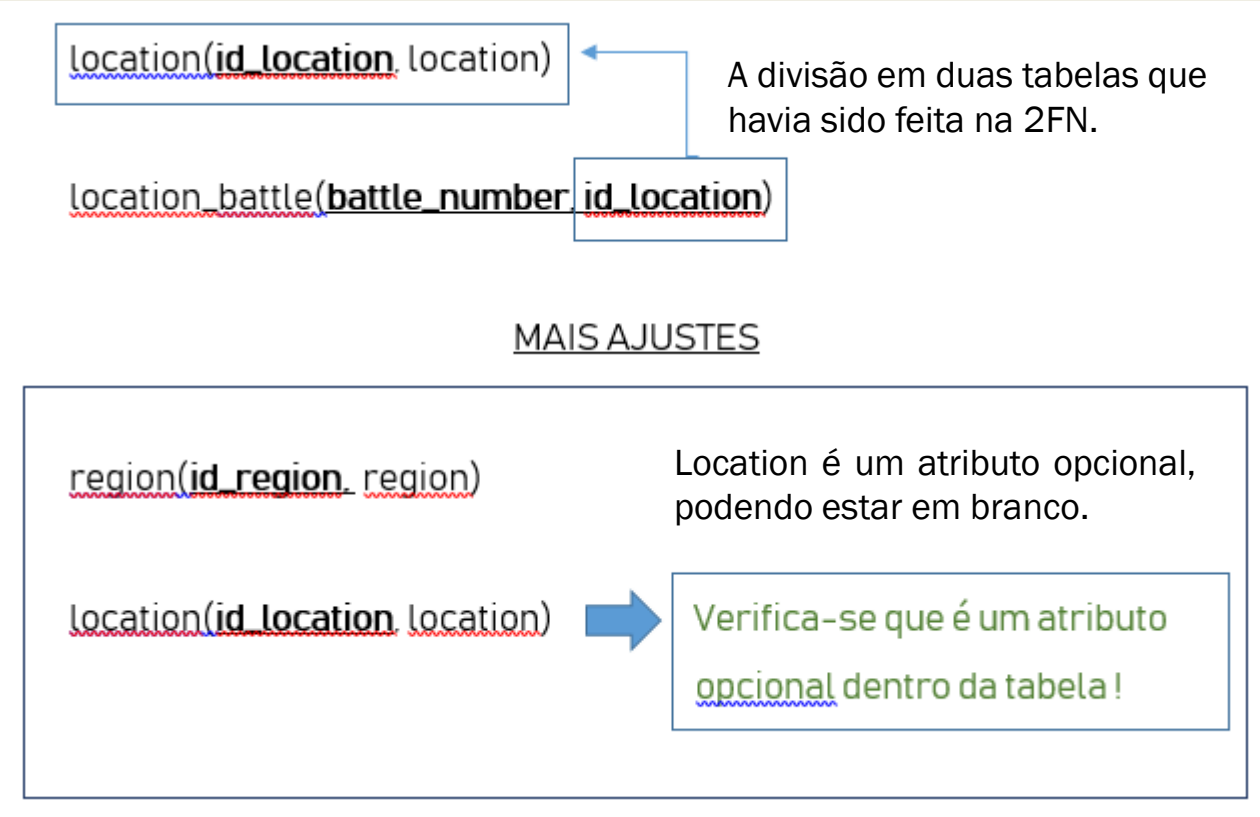
No ano 300 será inverno, sem previsão de fim do mesmo (podendo durar o ano de 300 inteiro ou alguns meses).

AJUSTES



Julgamos a necessidade de criar uma tabela região, visto que há muita repetição dentro da coluna.

AJUSTES



Criou-se duas tabelas para determinar em que região e que localização ocorreram as batalhas (lembrando que uma batalha pode ter ocorrido em duas localidades distintas).

As tabelas recebem referência de location e region.

location_battle(**battle_number**, id_location)

region_battle(**battle_number**, id_region)

AJUSTES

Remove region da tabela battles, pois criamos uma tabela única para ela.

battles(name, year, **battle_number**, , attacker_1, attacker_2, attacker_3, attacker_4, defender_1, defender_2, defender_3, defender_4, attacker_outcome, battle_type, major_death, major_capture, attacker_size, defender_size, summer, ~~region~~, note)

Verificamos também que as colunas attacker_1, attacker_2, attacker_3, attacker_4, defender_1, defender_2, defender_3, defender_4, se referem as casas de *Game of Thrones*. Portanto possuem o mesmo tipo de dado. **Optamos por transformar essas colunas em 3 tabelas.**

house(**id_house**, house_name)

attacker(**battle_number**, **id_house**)

defender(**battle_number**, **id_house**)

TERCEIRA FORMA NORMAL

Solução

battles(name, year, **battle_number**, attacker_outcome, battle_type, major_death, major_capture, attacker_size, defender_size, summer, note)

king(**id_king**, king_name)

house(**id_house**, house_name)

region(**id_region**, region)

location(**id_location**, location)

commander(**id_comander**, commander)

attacker_king(**battle_number**, **id_king**)

battle_number referencia battles

id_king referencia king

TERCEIRA FORMA NORMAL

Solução

defender_king(**battle_number, id_king**)

battle_number referencia battles

id_king referencia king

attacker_commander(**battle_number, id_commander**)

battle_number referencia battles

id_commander referencia commander

defender_commander(**battle_number, id_commander**.)

battle_number referencia battles

id_commander referencia commander

location_battle(**battle_number**, id_location)

battle_number referencia battles

id_location referencia location

TERCEIRA FORMA NORMAL

Solução

region_battle(**battle_number**, id_region)

battle_number referencia battles

id_region referencia region

attacker(**battle_number**, **id_house**)

battle_number referencia battles

id_house referencia house

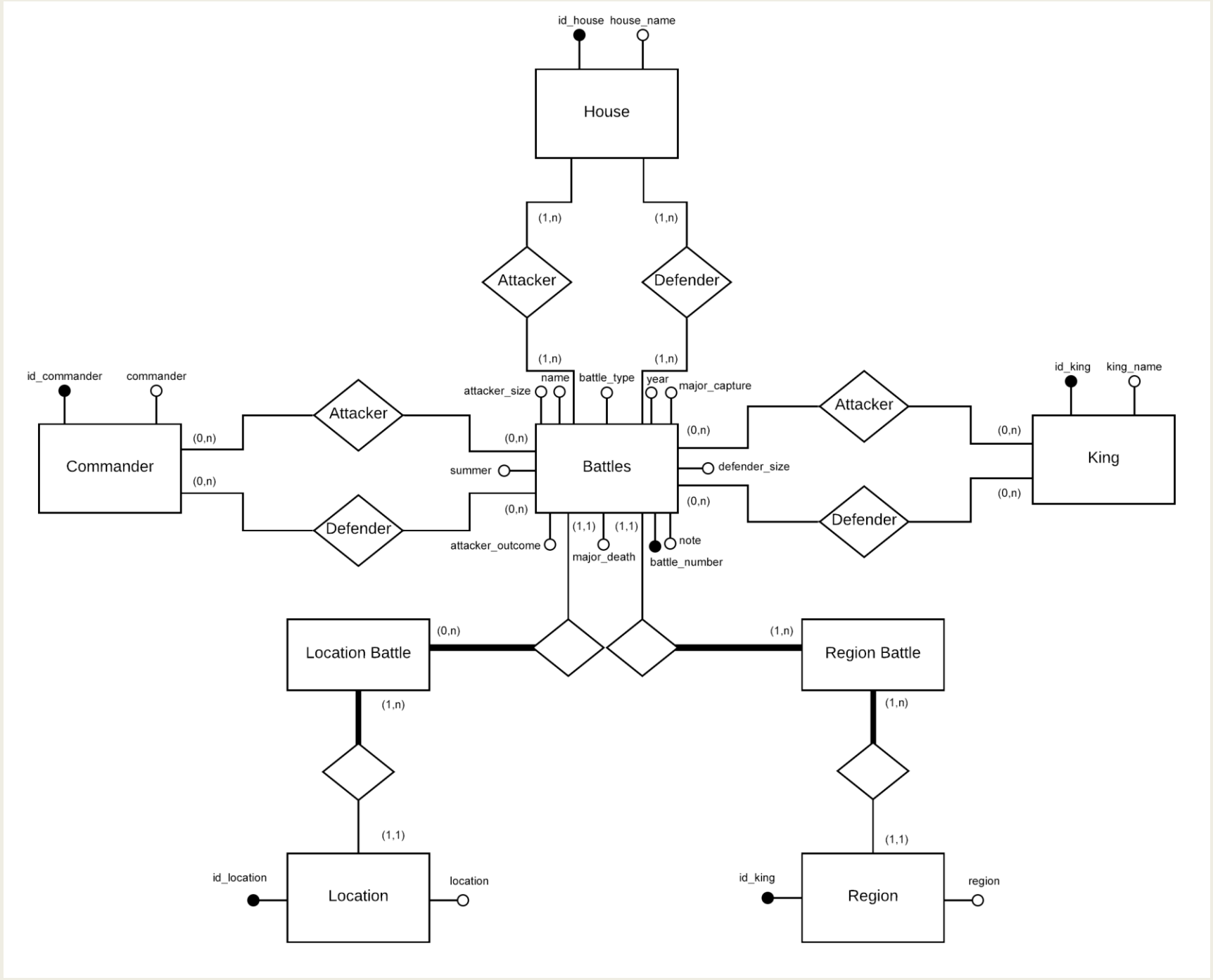
defender(**battle_number**, **id_house**)

battle_number referencia battles

id_house referencia house

ETAPA 3 – DIAGRAMA ER

Com base na tabela normalizada, elaborou-se o seguinte **Diagrama Entidade Relacional.**


id_house
house_name
House
(1,n)
(1,n)
Attacker
Defender
(1,n)
(1,n)
id_commander
commander
name
battle_type
year
major_capture
attacker_size
id_king
king_name
Attacker
Attacker
(0,n)
(0,n)
(0,n)
(0,n)
Commander
Battles
King
summer
defender_size
(0,n)
(0,n)
(0,n)
(0,n)
Defender
Defender
(1,1)
(1,1)
attacker_outcome
major_death
note
battle_number
(0,n)
(1,n)
Location Battle
Region Battle
(1,n)
(1,n)
(1,1)
(1,1)
id_location
location
Location
id_king
region
Region

ETAPA 4 – CARGA DE DADOS

```
USE gotbattles;

LOAD DATA LOCAL INFILE 'C:/Users//Desktop/Trabalho-Banco-de-Dados/Fundamentos-de-BD/Fundamentos-de-BD/
INTO TABLE battles_desnormalizado
CHARACTER SET utf8
FIELDS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\r\n'
IGNORE 1 LINES
(
    name,
    year,
    battle_number,
    attacker_king,
    defender_king,
    attacker_1,
    attacker_2,
    attacker_3,
    attacker_4,
    defender_1,
    defender_2,
    defender_3,
    defender_4,
    attacker_outcome,
    battle_type,
    major_death,
    major_capture,
    attacker_size,
    defender_size,
    attacker_commander,
    defender_commander,
    summer,
    location,
    region,
    note
)
```

Optou-se por criar um script SQL que levasse os dados carregados na tabela desnormalizada para a tabela normalizada.

Ao lado está o script que faz a leitura dos dados da tabela csv escolhida para a tabela desnormalizada.

DESNORMALIZADA PARA NORMALIZADA

```
INSERT INTO battle(name, year, battle_number, attacker_outcome, battle_type, major_death, major_capture, attacker_size, defender_size, summer, note)
SELECT DISTINCT name, year, battle_number, attacker_outcome, battle_type, major_death, major_capture, attacker_size, defender_size, summer, note
FROM battles_desnormalizado;

INSERT INTO region(region)
SELECT DISTINCT region
FROM battles_desnormalizado;
```

Para region e battle, não houve a necessidade de grandes manipulações para inserir os dados nas respectivas tabelas normalizadas, visto que suas colunas não possuíam atributos multivalorados.

```sql
INSERT INTO numbers VALUE (1), (2), (3), (4), (5), (6), (7);

/*Tabela auxiliar*/
INSERT INTO tab_aux_location(battle_number, location)
SELECT battle_number, SUBSTRING_INDEX(SUBSTRING_INDEX(battles_desnormalizado.location,',', numbers.n), ',', -1) location
FROM numbers INNER JOIN battles_desnormalizado ON CHAR_LENGTH(battles_desnormalizado.location)
    -CHAR_LENGTH(replace(battles_desnormalizado.location, ',', '')) >= numbers.n - 1
WHERE battles_desnormalizado.location != ""
ORDER BY battle_number, n;

/*carga das regiões onde ocorreram batalhas*/
INSERT INTO region_battle(battle_number, id_region)
SELECT bd.battle_number, r.id_region FROM region AS r INNER JOIN battles_desnormalizado AS bd ON r.region = bd.region;

/*Carga para location*/
INSERT INTO location(location)
SELECT DISTINCT location
FROM tab_aux_location;

/*Carrega os locais onde ocorreram batalhas*/
INSERT INTO location_battle(battle_number, id_location)
SELECT battle_number, l.id_location
FROM tab_aux_location AS tab INNER JOIN location AS l ON tab.location = l.location;

DROP TABLE tab_aux_location;
```

Já para location, a situação era outra. Com a existência de aributos multivalorados, necessitou-se criar uma tabela auxiliar na qual pudéssemos dividir essa única entrada multivalorada em várias linhas “monovaloradas” com o mesmo battle_number. Depois de fazer isso inserimos os locais dessa tabela auxiliar na nossa tabela location e os ids na nossa tabela location_battle e eliminamos a tabela, visto que não a usaríamos mais.

O mesmo caso ocorre nas tabelas king e commander, que possuem atributos multivalorados.

```sql
/*Table house*/

INSERT INTO house(house_name)
SELECT DISTINCT attacker_1
FROM battles_desnormalizado
WHERE attacker_1 != " "
UNION
SELECT DISTINCT attacker_2
FROM battles_desnormalizado
WHERE attacker_2 != " "
UNION
SELECT DISTINCT attacker_3
FROM battles_desnormalizado
WHERE attacker_3 != " "
UNION
SELECT DISTINCT attacker_4
FROM battles_desnormalizado
WHERE attacker_4 != " "
UNION
SELECT DISTINCT  defender_1
FROM battles_desnormalizado
WHERE defender_1 != " "
UNION
SELECT DISTINCT defender_2
FROM battles_desnormalizado
WHERE defender_2 != " "
UNION
SELECT DISTINCT defender_3
FROM battles_desnormalizado
WHERE defender_3 != " "
UNION
SELECT DISTINCT defender_4
FROM battles_desnormalizado
WHERE defender_4 != " ";
```

Carrega os dados de todas as tabelas de casas atacantes e defensoras em apenas uma tabela casas. Tornando, dessa forma, mais claras as tabelas. Em seguida, se carrega o id na tabela normalizada attacker e defender, juntamente com os nomes das casas participantes.

```sql
/*Carga para a tabela attacker*/
INSERT INTO attacker(battle_number, id_house)
SELECT battle_number, h.id_house FROM battles_desnormalizado AS bd INNER JOIN house AS h ON bd.attacker_1 = h.house_name
UNION
SELECT battle_number, h.id_house FROM battles_desnormalizado AS bd INNER JOIN house AS h ON bd.attacker_2 = h.house_name
UNION
SELECT battle_number, h.id_house FROM battles_desnormalizado AS bd INNER JOIN house AS h ON bd.attacker_3 = h.house_name
UNION
SELECT battle_number, h.id_house FROM battles_desnormalizado AS bd INNER JOIN house AS h ON bd.attacker_4 = h.house_name;

/*---------------------------------------------*/

/*Carga para a tabela defender*/
INSERT INTO defender(battle_number, id_house)
SELECT battle_number, h.id_house FROM battles_desnormalizado AS bd INNER JOIN house AS h ON bd.defender_1 = h.house_name
UNION
SELECT battle_number, h.id_house FROM battles_desnormalizado AS bd INNER JOIN house AS h ON bd.defender_2 = h.house_name
UNION
SELECT battle_number, h.id_house FROM battles_desnormalizado AS bd INNER JOIN house AS h ON bd.defender_3 = h.house_name
UNION
SELECT battle_number, h.id_house FROM battles_desnormalizado AS bd INNER JOIN house AS h ON bd.defender_4 = h.house_name;
```

ETAPA 5 - CONSULTAS

Foram realizados quatro tipos de consultas, cada uma envolvendo diferentes tipos de operações e construtores. Abaixo alguns utilizados:

- Count, Max;
- Natural Join, Left Join;
- Subconsulta;
- Union;

1º CONSULTA

Número de Batalhas que aconteceram em Riverrun

```
/* Número de batalhas que aconteceram em riverrun */
SELECT count(lb.battle_number)
FROM location as l natural join location_battle as lb
WHERE l.location = "Riverrun";
```

Resultado

count(lb.battle_number)
3

Validação (Verificado pelo filtro do Excel)

	name	location
4	Battle of Riverrun	Riverrun
7	Battle of the Camps	Riverrun
37	Siege of Riverrun	Riverrun

Ocorreram, realmente, somente três batalhas em Riverrun.

2º CONSULTA

Retorna o nome de todas as batalhas em que Robb Stark atuou como rei e/ou como comandante.

```
/*Retorna o nome das batalhas em que Robb Stark atuou como rei e/ou comandante*/
SELECT b.name
FROM battle as b join (
    SELECT list_k.battle_number /* Se extrai as batalhas com as especificações abai
    FROM king as k join (
        SELECT * /* Junta-se a lista de reis atacantes e defensor */
        FROM attacker_king
        UNION
        SELECT *
        FROM defender_king) as list_k
    WHERE k.id_king = list_k.id_king and k.king_name = "Robb Stark" /* Se isola ape
    UNION /* Junta-se a lista de reis com a lista de comandantes */
    SELECT list_c.battle_number
    FROM commander as c join (
        SELECT * /* Junta-se a lista de comandantes atacantes e defensor */
        FROM attacker_commander
        UNION
        SELECT *
        FROM defender_commander) as list_c
    WHERE c.id_commander = list_c.id_commander and c.commander_name = "RobbStark" /
) as robb_battles /* Retorna uma lista de batalhas (battle_numbers) em que Robb Sta
WHERE b.battle_number = robb_battles.battle_number
ORDER BY robb_battles.battle_number;
```

Resultado

	name
	Battle of the Golden Tooth
	Battle at the Mummer's Ford
	Battle of Riverrun
	Battle of the Green Fork
	Battle of the Whispering Wood
	Battle of the Camps
	Sack of Darry
	Battle of Moat Cailin
	Battle of Deepwood Motte
	Battle of the Stony Shore
	Battle of Torrhen's Square
	Battle of Winterfell
	Sack of Winterfell
	Battle of Oxcross
	Battle of the Fords
	Sack of Harrenhal
	Battle of the Crag
	Siege of Darry
	Battle of Duskendale
	Battle of the Ruby Ford
	The Red Wedding
	Siege of Seagard
	Siege of Riverrun
	Siege of Raventree

Validação (Verificado pelo filtro do Excel)

1	name	attacker_king	defender_king	attacker_commander	defender_commander
2	Battle of the Golden Tooth	Joffrey/Tommen Baratheor	Robb Stark	Jaime Lannister	Clement Piper, Vance
3	Battle at the Mummer's Ford	Joffrey/Tommen Baratheor	Robb Stark	Gregor Clegane	Beric Dondarrion
4	Battle of Riverrun	Joffrey/Tommen Baratheor	Robb Stark	Jaime Lannister, Andros Brax	Edmure Tully, Tytos Blackwo
5	Battle of the Green Fork	Robb Stark	Joffrey/Tommen Baratheor	Roose Bolton, Wylis Manderly, M	Tywin Lannister, Gregor Cleg
6	Battle of the Whispering Woc	Robb Stark	Joffrey/Tommen Baratheor	Robb Stark, Brynden Tully	Jaime Lannister
7	Battle of the Camps	Robb Stark	Joffrey/Tommen Baratheor	Robb Stark, Tytos Blackwood, B	Lord Andros Brax, Forley Pre
8	Sack of Darry	Joffrey/Tommen Baratheor	Robb Stark	Gregor Clegane	Lyman Darry
9	Battle of Moat Cailin	Balon/Euron Greyjoy	Robb Stark	Victarion Greyjoy	
10	Battle of Deepwood Motte	Balon/Euron Greyjoy	Robb Stark	Asha Greyjoy	
11	Battle of the Stony Shore	Balon/Euron Greyjoy	Robb Stark	Theon Greyjoy	
12	Battle of Torrhen's Square	Robb Stark	Balon/Euron Greyjoy	Rodrik Cassel, Cley Cerwyn	Dagmer Cleftjaw
13	Battle of Winterfell	Balon/Euron Greyjoy	Robb Stark	Theon Greyjoy	Bran Stark
15	Sack of Winterfell	Joffrey/Tommen Baratheor	Robb Stark	Ramsay Snow, Theon Greyjoy	Rodrik Cassel, Cley Cerwyn, L
16	Battle of Oxcross	Robb Stark	Joffrey/Tommen Baratheor	Robb Stark, Brynden Tully	Stafford Lannister, Roland Cı
18	Battle of the Fords	Joffrey/Tommen Baratheor	Robb Stark	Tywin Lannister, Flement Brax, C	Edmure Tully, Jason Mallister
19	Sack of Harrenhal	Robb Stark	Joffrey/Tommen Baratheor	Roose Bolton, Vargo Hoat, Robe	Amory Lorch
20	Battle of the Crag	Robb Stark	Joffrey/Tommen Baratheor	Robb Stark, Smalljon Umber, Bla	Rolph Spicer
22	Siege of Darry	Robb Stark	Joffrey/Tommen Baratheor	Helman Tallhart	
23	Battle of Duskendale	Robb Stark	Joffrey/Tommen Baratheor	Robertt Glover, Helman Tallhart	Randyll Tarly, Gregor Clegan
25	Battle of the Ruby Ford	Joffrey/Tommen Baratheor	Robb Stark	Gregor Clegane	Roose Bolton, Wylis Manderly
27	The Red Wedding	Joffrey/Tommen Baratheor	Robb Stark	Walder Frey, Roose Bolton, Wal	Robb Stark
28	Siege of Seagard	Robb Stark	Joffrey/Tommen Baratheor	Walder Frey	Jason Mallister
37	Siege of Riverrun	Joffrey/Tommen Baratheor	Robb Stark	Daven Lannister, Ryman Fey, Ja	Brynden Tully
38	Siege of Raventree	Joffrey/Tommen Baratheor	Robb Stark	Jonos Bracken, Jaime Lannister	Tytos Blackwood

3º CONSULTA

Seleciona o tamanho do maior exército que a casa Lannister já tenha atacado em uma batalha.

```
/* Seleciona o tamanho do maior exercito (attacker size) em que a casa Lannister atacou */
SELECT max(b.attacker_size)
FROM battle as b join (
    SELECT *
    FROM house as h natural join attacker as atk
    WHERE h.house_name = "Lannister") as ataques_lannister
WHERE ataques_lannister.battle_number = b.battle_number;
```

Resultado

	max(b.attacker_size)
	20000

Validação (Verificado pelo filtro do Excel)

1	attacker_1	attacke	attacke	attacke	attacker	HELPER
2	Lannister				15000	1
3	Lannister					1
4	Lannister				15000	1
8	Lannister					1
18	Lannister				20000	1
25	Lannister					1
26	Lannister					1
37	Lannister	Frey			3000	1
38	Bracken	Lannister			1500	2

4º CONSULTA

Retorna todas batalhas e os nomes dos comandantes que defenderam. Devem ser retornadas também aquelas batalhas que não tiveram comandante defensor.

```
/* Seleciona todas as batalhas e suas respectivas comandantes defensores, incluindo aqueles que não possuem um
SELECT b.name, commander_list.commander_name
FROM battle as b LEFT JOIN  (
    SELECT *
    FROM defender_commander natural join commander) as commander_list
ON commander_list.battle_number = b.battle_number;
```