**A5: Esquema relacional, validação e melhoramento**

Este artefacto contém o esquema relacional obtido através do mapeamento do modelo conceptual de dados. O esquema relacional inclui o esquema de relações, atributos, domínios, chaves primárias, chaves estrangeiras e outras regras de integridade: UNIQUE, DEFAULT, NOT NULL, CHECK.

**1. Esquema Relacional**

|  |  |
| --- | --- |
| R01 | User(id, username **UK NN**, password **NN**, email **UK NN**, regist\_date **NN DF** Today, first\_name **NN**, last\_name **NN**, image\_path, city\_id → City **NN**) |
| R02 | Event(id, name **NN**, date **NN** date > Today, description **NN**, owner\_id → User **NN**, localization\_id → Localization **NN**, image\_id → Image,  type **NN CK** type IN Types\_of\_event, category **NN CK** category IN Categories) |
| R03 | Done(event\_id → Event, rating **NN CK** rating > 0 AND rating < = 5) |
| R04 | Not\_done(event\_id → Event) |
| R05 | Paticipant(id, (user\_id → User**,** event\_id → Event) **UK** **NN**) |
| R06 | Owner(id (user\_id → User**,** event\_id → Event) **UK NN**) |
| R07 | Image(id, path **NN UK**) |
| R08 | Localization(id, name, address **NN**, latitude, longitude, city\_id → City **NN**) |
| R09 | City(id, name **UK NN**, country\_id → Country **NN**) |
| R10 | Country(id, name **UK NN**) |
| R11 | Post(id, description **NN**, date **NN DF** Today, event\_id → Event **NN**,  user\_id → User **NN**, image\_id → Image) |
| R12 | Poll(id, post\_id → Post **NN**) |
| R13 | Option(id, description **NN**, poll\_id → Poll **NN**) |
| R14 | Admin(id, username **UK NN**, password **NN**, email **UK NN**) |
| R15 | Friend\_request(id, answer, sender\_id → User **NN,** receiver\_id → User **NN**) |
| R16 | Friend\_activity(id, sender\_id → User **NN,** receiver\_id → User **NN**,  event\_id → Event **NN**) |
| R17 | Event\_invite(id, answer, event\_id → Not\_done **NN**, owner\_id → User **NN**,  receiver\_id → User **NN**) |

**2. Domínios**

|  |  |
| --- | --- |
| Today | DATE DEFAULT CURRENT\_DATE |
| Types\_of\_event | ENUM(‘Public’, Private’) |
| Categories | ENUM(‘Music’, ‘Sports’, ‘Entertainment’, ‘Educational’, ‘Business’, ‘Other’) |

**3. Dependências funcionais e validação do esquema**

|  |  |
| --- | --- |
| **Table R01** (User) | |
| **Keys:** {id}, {username}, {email} | |
| **Dependências Funcionais** | |
| FD0101 | {id} → {username, email, password, regist\_date, first\_name, last\_name, image, city\_id} |
| FD0102 | {username} → {id, email, password, regist\_date, first\_name, last\_name, image, city\_id} |
| FD0103 | {email} → {id, username, password, regist\_date, first\_name, last\_name, image, city\_id} |
| **Forma Normal** | BCNF |

|  |  |
| --- | --- |
| **Table R02** (Event) | |
| **Keys:** {id} | |
| **Dependências Funcionais** | |
| FD0201 | {id} → {name, date, type, category, description, localization\_id, owner\_id, image\_id} |
| **Forma Normal** | BCNF |

|  |  |
| --- | --- |
| **Table R03** (Done) | |
| **Keys:** {event\_id} | |
| **Dependências Funcionais** | |
| FD0301 | {event\_id} → {rating} |
| **Forma Normal** | BCNF |

|  |  |
| --- | --- |
| **Table R04** (Not\_done) | |
| **Keys:** {event\_id} | |
| **Dependências Funcionais** | |
| (none) | |
| **Forma Normal** | BCNF |

|  |  |
| --- | --- |
| **Table R05** (Participant) | |
| **Keys:** {id}, {user\_id, event\_id} | |
| **Dependências Funcionais** | |
| FD0501 | {id} → {user\_id, event\_id} |
| FD0502 | {user\_id, event\_id} → {id} |
| **Forma Normal** | BCNF |

|  |  |
| --- | --- |
| **Table R06** (Owner) | |
| **Keys:** {id}, {user\_id, event\_id} | |
| **Dependências Funcionais** | |
| FD0601 | {id} → {user\_id, event\_id} |
| FD0602 | {user\_id, event\_id} → {id} |
| **Forma Normal** | BCNF |

|  |  |
| --- | --- |
| **Table R07** (Image) | |
| **Keys:** {id} | |
| **Dependências Funcionais** | |
| FD0701 | {id} → {path} |
| **Forma Normal** | BCNF |

|  |  |
| --- | --- |
| **Table R08** (Localization) | |
| **Keys:** {id} | |
| **Dependências Funcionais** | |
| FD0801 | {id} → {name, address, latitude, longitude, city\_id} |
| **Forma Normal** | BCNF |

|  |  |
| --- | --- |
| **Table R09** (City) | |
| **Keys:** {id} | |
| **Dependências Funcionais** | |
| FD0901 | {id} → {name, country\_id} |
| **Forma Normal** | BCNF |

|  |  |
| --- | --- |
| **Table R10** (Country) | |
| **Keys:** {id} | |
| **Dependências Funcionais** | |
| FD1001 | {id} → {name} |
| **Forma Normal** | BCNF |

|  |  |
| --- | --- |
| **Table R11** (Post) | |
| **Keys:** {id} | |
| **Dependências Funcionais** | |
| FD1101 | {id} → {description, date, event\_id, user\_id, image\_id} |
| **Forma Normal** | BCNF |

|  |  |
| --- | --- |
| **Table R12** (Poll) | |
| **Keys:** {id} | |
| **Dependências Funcionais** | |
| FD1201 | {id} → {post\_id} |
| **Forma Normal** | BCNF |

|  |  |
| --- | --- |
| **Table R13** (Option) | |
| **Keys:** {id} | |
| **Dependências Funcionais** | |
| FD1301 | {id} → {description, poll\_id} |
| **Forma Normal** | BCNF |

|  |  |
| --- | --- |
| **Table R14** (Admin) | |
| **Keys:** {id}, {username}, {email} | |
| **Dependências Funcionais** | |
| FD1401 | {id} → {username, email, password} |
| FD1402 | {username} → {id, email, password} |
| FD1403 | {email} → {id, username, password} |
| **Forma Normal** | BCNF |

|  |  |
| --- | --- |
| **Table R15** (Friend\_request) | |
| **Keys:** {id} | |
| **Dependências Funcionais** | |
| FD1501 | {id} → {answer, sender\_id, receiver\_id} |
| **Forma Normal** | BCNF |

|  |  |
| --- | --- |
| **Table R16** (Friend\_activity) | |
| **Keys:** {id} | |
| **Dependências Funcionais** | |
| FD1601 | {id} → {sender\_id, receiver\_id, event\_id} |
| **Forma Normal** | BCNF |

|  |  |
| --- | --- |
| **Table R17** (Event\_invite) | |
| **Keys:** {id} | |
| **Dependências Funcionais** | |
| FD1701 | {id} → {answer, event\_id, owner\_id, receiver\_id } |
| **Forma Normal** | BCNF |

**4. SQL Código**

[Sql code](https://github.com/Quaresma1997/lbaw1765/blob/gh-pages/lbaw1765_db.sql)

--Types Enums

DROP TYPE IF EXISTS categories CASCADE;

CREATE TYPE categories AS ENUM(

'Music',

'Sports',

'Entertainment',

'Educational',

'Business',

'Other'

);

DROP TYPE IF EXISTS types\_of\_event CASCADE;

CREATE TYPE types\_of\_event AS ENUM(

'Public',

'Private'

);

--Tables

DROP TABLE IF EXISTS admins CASCADE;

CREATE TABLE admins (

id SERIAL NOT NULL,

username text NOT NULL,

password text NOT NULL,

email text NOT NULL,

CONSTRAINT admins\_pk PRIMARY KEY (id),

CONSTRAINT admins\_email\_uk UNIQUE (email)

);

DROP TABLE IF EXISTS cities CASCADE;

CREATE TABLE cities (

id SERIAL NOT NULL,

name text NOT NULL,

country\_id INTEGER NOT NULL,

CONSTRAINT cities\_pk PRIMARY KEY (id),

CONSTRAINT cities\_name\_uk UNIQUE (name)

);

DROP TABLE IF EXISTS countries CASCADE;

CREATE TABLE countries (

id SERIAL NOT NULL,

name text NOT NULL,

CONSTRAINT countries\_pk PRIMARY KEY (id),

CONSTRAINT countries\_name\_uk UNIQUE (name)

);

DROP TABLE IF EXISTS dones CASCADE;

CREATE TABLE dones (

event\_id INTEGER NOT NULL,

rating INTEGER NOT NULL,

CONSTRAINT dones\_pk PRIMARY KEY (event\_id),

CONSTRAINT rating\_ck CHECK (((rating >= 1) AND (rating <= 5)))

);

DROP TABLE IF EXISTS events CASCADE;

CREATE TABLE events (

id SERIAL NOT NULL,

name text NOT NULL,

date TIMESTAMP WITH TIME zone NOT NULL,

description text NOT NULL,

owner\_id INTEGER NOT NULL,

localization\_id INTEGER NOT NULL,

image\_id INTEGER NOT NULL,

type event\_type NOT NULL,

category categories NOT NULL,

CONSTRAINT events\_pk PRIMARY KEY (id),

CONSTRAINT date\_ck CHECK ((date > now()))

);

DROP TABLE IF EXISTS event\_invites CASCADE;

CREATE TABLE event\_invites (

id SERIAL NOT NULL,

answer text NOT NULL,

event\_id INTEGER NOT NULL,

owner\_id INTEGER NOT NULL,

receiver\_id INTEGER NOT NULL,

CONSTRAINT event\_invites\_pk PRIMARY KEY (id)

);

DROP TABLE IF EXISTS friend\_activities CASCADE;

CREATE TABLE friend\_activities (

id SERIAL NOT NULL,

sender\_id INTEGER NOT NULL,

receiver\_id INTEGER NOT NULL,

event\_id INTEGER NOT NULL,

CONSTRAINT friend\_activities\_pk PRIMARY KEY (id),

CONSTRAINT friend\_activities\_event\_id\_fk FOREIGN KEY (event\_id) REFERENCES

events(id) ON DELETE CASCADE

);

DROP TABLE IF EXISTS friend\_requests CASCADE;

CREATE TABLE friend\_requests (

id SERIAL NOT NULL,

answer text NOT NULL,

sender\_id INTEGER NOT NULL,

receiver\_id INTEGER NOT NULL,

event\_id INTEGER NOT NULL,

CONSTRAINT friend\_requests\_pk PRIMARY KEY (id)

);

DROP TABLE IF EXISTS images CASCADE;

CREATE TABLE images (

id SERIAL NOT NULL,

path text NOT NULL,

CONSTRAINT images\_pk PRIMARY KEY (id),

CONSTRAINT images\_path\_uk UNIQUE (path)

);

DROP TABLE IF EXISTS localizations CASCADE;

CREATE TABLE localizations (

id SERIAL NOT NULL,

name text NOT NULL,

address text NOT NULL,

latitude FLOAT,

longitude FLOAT,

city\_id INTEGER NOT NULL,

CONSTRAINT localizations\_pk PRIMARY KEY (id),

CONSTRAINT localizations\_city\_id\_fk FOREIGN KEY (city\_id) REFERENCES

cities(id) ON DELETE SET NULL

);

DROP TABLE IF EXISTS not\_dones CASCADE;

CREATE TABLE not\_dones (

event\_id INTEGER NOT NULL,

CONSTRAINT not\_dones\_pk PRIMARY KEY (event\_id),

CONSTRAINT not\_dones\_event\_id\_fk FOREIGN KEY (event\_id) REFERENCES

events(id) ON UPDATE CASCADE

);

DROP TABLE IF EXISTS options CASCADE;

CREATE TABLE options (

id SERIAL NOT NULL,

description text NOT NULL,

poll\_id INTEGER NOT NULL,

CONSTRAINT options\_pk PRIMARY KEY (id)

);

DROP TABLE IF EXISTS owners CASCADE;

CREATE TABLE owners (

id SERIAL NOT NULL,

user\_id INTEGER NOT NULL,

event\_id INTEGER NOT NULL,

CONSTRAINT owners\_pk PRIMARY KEY (id),

CONSTRAINT owners\_user\_id\_event\_id\_uk UNIQUE (user\_id, event\_id),

CONSTRAINT owners\_event\_id\_fk FOREIGN KEY (event\_id) REFERENCES

events(id) ON DELETE SET NULL

);

DROP TABLE IF EXISTS participants CASCADE;

CREATE TABLE participants (

id SERIAL NOT NULL,

user\_id INTEGER NOT NULL,

event\_id INTEGER NOT NULL,

CONSTRAINT participants\_pk PRIMARY KEY (id),

CONSTRAINT participants\_user\_id\_event\_id\_uk UNIQUE (user\_id, event\_id),

CONSTRAINT participants\_event\_id\_fk FOREIGN KEY (event\_id) REFERENCES

events(id) ON DELETE SET NULL

);

DROP TABLE IF EXISTS polls CASCADE;

CREATE TABLE polls (

id SERIAL NOT NULL,

post\_id INTEGER NOT NULL,

CONSTRAINT polls\_pk PRIMARY KEY (id)

);

DROP TABLE IF EXISTS posts CASCADE;

CREATE TABLE posts (

id SERIAL NOT NULL,

description text NOT NULL,

date TIMESTAMP WITH TIME zone NOT NULL,

event\_id INTEGER NOT NULL,

user\_id INTEGER NOT NULL,

image\_id INTEGER NOT NULL,

CONSTRAINT posts\_pk PRIMARY KEY (id),

CONSTRAINT posts\_event\_id\_fk FOREIGN KEY (event\_id) REFERENCES

events(id) ON DELETE CASCADE,

CONSTRAINT posts\_image\_id\_fk FOREIGN KEY (image\_id) REFERENCES

images(id) ON DELETE SET NULL

);

DROP TABLE IF EXISTS users CASCADE;

CREATE TABLE users (

id SERIAL NOT NULL,

username text NOT NULL,

password text NOT NULL,

email text NOT NULL,

regist\_date TIMESTAMP WITH TIME zone DEFAULT now() NOT NULL,

first\_name text NOT NULL,

last\_name text NOT NULL,

image\_path text NOT NULL,

city\_id INTEGER NOT NULL,

CONSTRAINT users\_pk PRIMARY KEY (id),

CONSTRAINT users\_name\_uk UNIQUE (username),

CONSTRAINT users\_email\_uk UNIQUE (email),

CONSTRAINT users\_city\_id\_fk FOREIGN KEY (city\_id) REFERENCES

cities(id) ON DELETE SET NULL

);

ALTER TABLE ONLY cities

ADD CONSTRAINT cities\_country\_id\_fk FOREIGN KEY (country\_id) REFERENCES

countries(id) ON DELETE SET NULL;

ALTER TABLE ONLY dones

ADD CONSTRAINT dones\_event\_id\_fk FOREIGN KEY (event\_id) REFERENCES

events(id) ON UPDATE CASCADE;

ALTER TABLE ONLY events

ADD CONSTRAINT events\_owner\_id\_fk FOREIGN KEY (owner\_id) REFERENCES

users(id) ON UPDATE CASCADE;

ALTER TABLE ONLY events

ADD CONSTRAINT events\_localization\_id\_fk FOREIGN KEY (localization\_id) REFERENCES

localizations(id) ON DELETE SET NULL;

ALTER TABLE ONLY events

ADD CONSTRAINT events\_image\_id\_fk FOREIGN KEY (image\_id) REFERENCES

images(id) ON DELETE SET NULL;

ALTER TABLE ONLY event\_invites

ADD CONSTRAINT event\_invites\_event\_id\_fk FOREIGN KEY (event\_id) REFERENCES

not\_dones(event\_id) ON DELETE CASCADE;

ALTER TABLE ONLY event\_invites

ADD CONSTRAINT event\_invites\_owner\_id\_fk FOREIGN KEY (owner\_id) REFERENCES

owners(id) ON DELETE SET NULL;

ALTER TABLE ONLY event\_invites

ADD CONSTRAINT event\_invites\_receiver\_id\_fk FOREIGN KEY (receiver\_id) REFERENCES

users(id) ON DELETE SET NULL;

ALTER TABLE ONLY friend\_activities

ADD CONSTRAINT friend\_activities\_sender\_id\_fk FOREIGN KEY (sender\_id) REFERENCES

users(id) ON DELETE SET NULL;

ALTER TABLE ONLY friend\_activities

ADD CONSTRAINT friend\_activities\_receiver\_id\_fk FOREIGN KEY (receiver\_id) REFERENCES

users(id) ON DELETE SET NULL;

ALTER TABLE ONLY friend\_requests

ADD CONSTRAINT friend\_requests\_sender\_id\_fk FOREIGN KEY (sender\_id) REFERENCES

participants(id) ON DELETE CASCADE;

ALTER TABLE ONLY friend\_requests

ADD CONSTRAINT friend\_requests\_receiver\_id\_fk FOREIGN KEY (receiver\_id) REFERENCES

users(id) ON DELETE CASCADE;

ALTER TABLE ONLY options

ADD CONSTRAINT options\_poll\_id\_fk FOREIGN KEY (poll\_id) REFERENCES

polls(id) ON DELETE CASCADE;

ALTER TABLE ONLY owners

ADD CONSTRAINT owners\_user\_id\_fk FOREIGN KEY (user\_id) REFERENCES

users(id) ON UPDATE CASCADE;

ALTER TABLE ONLY participants

ADD CONSTRAINT participants\_user\_id\_fk FOREIGN KEY (user\_id) REFERENCES

users(id) ON UPDATE CASCADE;

ALTER TABLE ONLY polls

ADD CONSTRAINT polls\_post\_id\_fk FOREIGN KEY (post\_id) REFERENCES

posts(id) ON DELETE CASCADE;

ALTER TABLE ONLY posts

ADD CONSTRAINT posts\_user\_id\_fk FOREIGN KEY (user\_id) REFERENCES

users(id) ON DELETE CASCADE;

**Histórico de Revisão**

**Alterações feitas à primeira submissão:**

1. **Correção das chaves do partcipant e owner**
2. **Correção de algumas dependências funcionais**
3. **Passagem de algumas chaves e restrições para dentro das classes;**

**GRUPO1765, 21/03/2018**

* Mariana Duarte Guimarães, [up201307777@fe.up.pt](mailto:up201307777@fe.up.pt)
* Rui Emanuel Cabral de Almeida Quaresma, [up201503005@fe.up.pt](mailto:up201503005@fe.up.pt)
* Rui Pedro Machado Araújo, [up201403263@fe.up.pt](mailto:up201403263@fe.up.pt)
* Tiago Duarte Carvalho, [up201504461@fe.up.pt](mailto:up201504461@fe.up.pt)