

# A5: Relational schema, validation and schema refinement

Our project, answerly, is a web application for collaborative questions and answers.

This artifact contains the Relational Schema obtained by mapping from the Conceptual Data Model. The Relational Schema includes the relation schema, attributes, domains, primary keys, foreign keys and other integrity rules: UNIQUE, DEFAULT, NOT NULL, CHECK

## 1. Relational Schema

Relation Reference	Relation Compact Notation
R01	user( <b>id</b> , first_name <i>NN</i> , last_name <i>NN</i> , email <i>UK NN</i> , bio, username <i>UK NN</i> , password <i>NN</i> , score <i>NN DF 0</i> )
R02	label( <b>id</b> , name <i>NN</i> )
R03	notification( <b>id</b> , content <i>NN</i> , date <i>DF Now NN</i> , viewed <i>DF False NN</i> , user_id → user <i>NN</i> )
R04	user_management( <b>id</b> , status <i>NN DF user CK status IN UserStatus</i> , date_last_changed <i>NN DF Now</i> , user_id → user <i>UK NN</i> )
R05	vote( <b>id</b> , vote <i>NN</i> , user_id → user <i>NN</i> , question_id → question, answer_id → answer <i>CK question_id = NN XOR answer_id = NN</i> )
R06	question( <b>id</b> , user_id → user <i>NN</i> , title <i>NN</i> , description <i>NN</i> , nr_likes <i>NN DF 0</i> , nr_dislikes <i>NN DF 0 CK nr_likes &gt;= 0 AND nr_dislikes &gt;= 0</i> , question_date <i>NN DF Now</i> )
R07	answer( <b>id</b> , user_id → user <i>NN</i> , question_id → question <i>NN</i> , answer_date <i>NN DF Now</i> , content <i>NN</i> , nr_likes <i>NN DF 0</i> , nr_dislikes <i>NN DF 0 CK nr_likes &gt;= 0 AND nr_dislikes &gt;= 0</i> , marked_answer <i>NN DF FALSE</i> )
R08	comment( <b>id</b> , user_id → user <i>NN</i> , question_id → question, answer_id → answer <i>CK question_id = NN XOR answer_id = NN</i> , content <i>NN</i> , comment_date <i>NN DF Now</i> )
R09	report( <b>id</b> , reporter_id → user <i>NN</i> , user_id → user, question_id → question, answer_id → answer, comment_id → comment <i>CK user_id = NN XOR question_id = NN XOR answer_id = NN XOR comment_id = NN</i> )
R10	report_status( <b>id</b> , report_id → report <i>ODC NN</i> , state <i>NN DF unresolved CK state IN ReportStates</i> , comment, responsible_user → user <i>NN ODC</i> )
R11	question_following( <b>user_id</b> → user, <b>question_id</b> → question)
R12	label_following( <b>user_id</b> → user, <b>label_id</b> → label)
R13	question_label( <b>question_id</b> → question, <b>label_id</b> → label)

- UK means UNIQUE KEY

- NN means NOT NULL
- DF means DEFAULT
- CK means CHECK
- ODC means ON DELETED CASCADE

## 2. Domains

Specification of additional domains:

Domain Name	Domain Specification
Now	DATE DEFAULT CURRENT_TIMESTAMP_
ReportStates	ENUM ('unresolved', 'reviewing', 'resolved')
UserStatus	ENUM ('user', 'moderator', 'administrator', 'banned')

## 3. Functional Dependencies and schema validation

In the following tables, all relations are in the Boyce-Codd Normal Form, since for each non trivial functional dependency  $A \rightarrow B$ , A is a (super)key of the relation.

**Table R01 (user)**

<b>Keys:</b> {id}, {username}, {email}	
<b>Functional Dependencies</b>	
FD0101	{id} $\rightarrow$ {first_name, last_name, email, bio, username, password, score}
FD0102	{username} $\rightarrow$ {id, first_name, last_name, email, bio, password, score}
FD0103	{email} $\rightarrow$ {id, first_name, last_name, bio, username, password, score}
<b>NORMAL FORM</b>	BCNF

**Table R02 (label)**

<b>Keys:</b> {id}	
<b>Functional Dependencies</b>	
FD0201	{id} $\rightarrow$ {name}
<b>NORMAL FORM</b>	BCNF

**Table R03 (notification)**

<b>Keys:</b> {id}	
<b>Functional Dependencies</b>	
FD0301	{id} $\rightarrow$ {content, date, viewed, user_id}
<b>NORMAL FORM</b>	BCNF

**Table R04 (user\_management)****Keys:** {id}**Functional Dependencies**

FD0401                      {id} → {status, date\_last\_changed, user\_id}

**NORMAL FORM**                      BCNF**Table R05 (vote)****Keys:** {id}**Functional Dependencies**

FD0501                      {id} → {vote, user\_id, question\_id, answer\_id}

**NORMAL FORM**                      BCNF**Table R06 (question)****Keys:** {id}**Functional Dependencies**

FD0601                      {id} → {user\_id, title, description, nr\_likes, nr\_dislikes, question\_date}

**NORMAL FORM**                      BCNF**Table R07 (answer)****Keys:** {id}**Functional  
Dependencies**FD0701                      {id} → {user\_id, question\_id, answer\_date, content, nr\_likes, nr\_dislikes,  
marked\_answer}**NORMAL FORM**                      BCNF**Table R08 (comment)****Keys:** {id}**Functional Dependencies**

FD0801                      {id} → {user\_id, question\_id, answer\_id, content, comment\_date}

**NORMAL FORM**                      BCNF**Table R09 (report)****Keys:** {id}

**Table R09 (report)****Functional Dependencies**

FD0901	$\{id\} \rightarrow \{reporter\_id, user\_id, question\_id, answer\_id, comment\_id\}$
--------	--

<b>NORMAL FORM</b>	BCNF
--------------------	------

**Table R10 (report\_status)****Keys:** {id}**Functional Dependencies**

FD1001	$\{id\} \rightarrow \{report\_id, state, comment, responsible\_user\}$
--------	--

<b>NORMAL FORM</b>	BCNF
--------------------	------

**Table R11 (question\_following)****Keys:** {user\_id, question\_id}**Functional Dependencies**

(none)

<b>NORMAL FORM</b>	BCNF
--------------------	------

**Table R12 (label\_following)****Keys:** {user\_id, label\_id}**Functional Dependencies**

(none)

<b>NORMAL FORM</b>	BCNF
--------------------	------

**Table R13 (question\_label)****Keys:** {question\_id, label\_id}**Functional Dependencies**

(none)

<b>NORMAL FORM</b>	BCNF
--------------------	------

## 4. SQL Code

```

-----
-- Drop old schmemma
-----

```

```

DROP TABLE IF EXISTS "user" CASCADE;
DROP TABLE IF EXISTS label CASCADE;
DROP TABLE IF EXISTS notification CASCADE;
DROP TABLE IF EXISTS user_management CASCADE;
DROP TABLE IF EXISTS question CASCADE;
DROP TABLE IF EXISTS answer CASCADE;
DROP TABLE IF EXISTS comment CASCADE;
DROP TABLE IF EXISTS vote CASCADE;
DROP TABLE IF EXISTS report CASCADE;
DROP TABLE IF EXISTS report_status CASCADE;
DROP TABLE IF EXISTS question_following CASCADE;
DROP TABLE IF EXISTS label_following CASCADE;
DROP TABLE IF EXISTS question_label CASCADE;

-----
-- Tables
-----

-- Table: user
CREATE TABLE "user" (
    id          SERIAL          PRIMARY KEY,
    first_name  TEXT            NOT NULL,
    last_name   TEXT            NOT NULL,
    email       TEXT            NOT NULL UNIQUE,
    bio         TEXT,
    username    TEXT            NOT NULL UNIQUE,
    password    TEXT            NOT NULL,
    score       INTEGER         NOT NULL DEFAULT 0
);

-- Table: label
CREATE TABLE label (
    id          SERIAL          PRIMARY KEY,
    name        TEXT            NOT NULL
);

-- Table: notification
CREATE TABLE notification (
    id          SERIAL          PRIMARY KEY,
    content     TEXT            NOT NULL,
    date        DATE            DEFAULT 'Now' NOT NULL,
    viewed      BOOLEAN         DEFAULT FALSE NOT NULL,
    user_id     INTEGER         REFERENCES "user" (id) NOT NULL
);

-- Table: user_management
CREATE TABLE user_management (
    id          SERIAL          PRIMARY KEY,
    status      TEXT            DEFAULT 'user' NOT NULL,
    date_last_changed DATE      DEFAULT 'Now' NOT NULL,
    user_id     INTEGER         REFERENCES "user" (id) NOT NULL
UNIQUE,
CHECK (
    status = 'user' OR status = 'moderator' OR status = 'administrator'

```

```
OR status = 'banned'
    )
);

-- Table: question
CREATE TABLE question (
    id SERIAL PRIMARY KEY,
    user_id INTEGER REFERENCES "user" (id) NOT NULL,
    title TEXT NOT NULL,
    description TEXT NOT NULL,
    nr_likes INTEGER DEFAULT 0 NOT NULL,
    nr_dislikes INTEGER DEFAULT 0 NOT NULL,
    question_date DATE DEFAULT 'Now' NOT NULL,
    CHECK (
        nr_likes >= 0 AND nr_dislikes >= 0
    )
);

-- Table: answer
CREATE TABLE answer (
    id SERIAL PRIMARY KEY,
    user_id INTEGER REFERENCES "user" (id) NOT NULL,
    question_id INTEGER REFERENCES "question" (id) NOT NULL,
    answer_date DATE DEFAULT 'Now' NOT NULL,
    content TEXT NOT NULL,
    nr_likes INTEGER DEFAULT 0 NOT NULL,
    nr_dislikes INTEGER DEFAULT 0 NOT NULL,
    marked_answer BOOLEAN DEFAULT FALSE NOT NULL,
    CHECK (
        nr_likes >= 0 AND nr_dislikes >= 0
    )
);

-- Table: comment
CREATE TABLE comment (
    id SERIAL PRIMARY KEY,
    user_id INTEGER REFERENCES "user" (id) NOT NULL,
    question_id INTEGER REFERENCES "question" (id),
    answer_id INTEGER REFERENCES "answer" (id),
    content TEXT NOT NULL,
    comment_date DATE DEFAULT 'Now' NOT NULL,
    CHECK (
        (question_id IS NOT NULL AND answer_id IS NULL) OR
        (question_id IS NULL AND answer_id IS NOT NULL)
    )
);

-- Table: vote
CREATE TABLE vote (
    id SERIAL PRIMARY KEY,
    "vote" BOOLEAN NOT NULL,
    user_id INTEGER REFERENCES "user" (id) NOT NULL,
    question_id INTEGER REFERENCES "question" (id),
    answer_id INTEGER REFERENCES "answer" (id),
```

```
        CHECK (
            (question_id IS NOT NULL AND answer_id IS NULL) OR
            (question_id IS NULL AND answer_id IS NOT NULL)
        )
    );

-- Table: report
CREATE TABLE report (
    id                SERIAL                PRIMARY KEY,
    reporter_id       INTEGER              REFERENCES "user" (id) NOT NULL,
    user_id           INTEGER              REFERENCES "user" (id),
    question_id       INTEGER              REFERENCES "question" (id),
    answer_id         INTEGER              REFERENCES "answer" (id),
    comment_id        INTEGER              REFERENCES "comment" (id),
    report_date       DATE                  DEFAULT 'Now' NOT NULL,
    description       TEXT                  NOT NULL,
    CHECK(
        (user_id IS NOT NULL AND question_id IS NULL AND answer_id IS NULL
        AND comment_id IS NULL) OR
        (user_id IS NULL AND question_id IS NOT NULL AND answer_id IS NULL
        AND comment_id IS NULL) OR
        (user_id IS NULL AND question_id IS NULL AND answer_id IS NOT NULL
        AND comment_id IS NULL) OR
        (user_id IS NULL AND question_id IS NULL AND answer_id IS NULL AND
        comment_id IS NOT NULL)
    )
);

-- Table: report_status
CREATE TABLE report_status (
    id                SERIAL                PRIMARY KEY,
    report_id         INTEGER              REFERENCES "report" (id) ON DELETE
    CASCADE NOT NULL,
    state             TEXT                  DEFAULT 'unresolved' NOT NULL,
    comment           TEXT,
    responsible_user  INTEGER              REFERENCES "user" (id) ON DELETE
    CASCADE NOT NULL,
    CHECK (
        state = 'unresolved' OR state = 'reviewing' OR state = 'resolved'
    )
);

-- Table: question_following
CREATE TABLE question_following (
    user_id           INTEGER              REFERENCES "user" (id) NOT NULL,
    question_id       INTEGER              REFERENCES "question" (id) NOT NULL,
    PRIMARY KEY (user_id, question_id)
);

-- Table: label_following
CREATE TABLE label_following (
    user_id           INTEGER              REFERENCES "user" (id) NOT NULL,
    label_id          INTEGER              REFERENCES "label" (id) NOT NULL,
    PRIMARY KEY (user_id, label_id)
```

```
);

-- Table: question_label
CREATE TABLE question_label (
  question_id      INTEGER      REFERENCES "question" (id) NOT NULL,
  label_id         INTEGER      REFERENCES "label" (id) NOT NULL,
  PRIMARY KEY (question_id, label_id)
);
```

## Revision history

1. First submission (23/03/2020).
2. Deleted Administrator and Moderator tables. Changed all id's to "id" and other minor changes (28/03/2020).
3. Tested sql code, fixed some errors (29/03/2020).

---

GROUP2064, 29/03/2020

- [Editor] Antonio Pedro Reis Ribeiro Sousa Dantas, up201703878@fe.up.pt
- Eduardo João Santana Macedo, up201703658@fe.up.pt
- Nuno Miguel Teixeira Cardoso, up201706162@fe.up.pt
- Paulo Roberto Dias Mourato, up201705616@fe.up.pt