1. Term Paper (CP1)

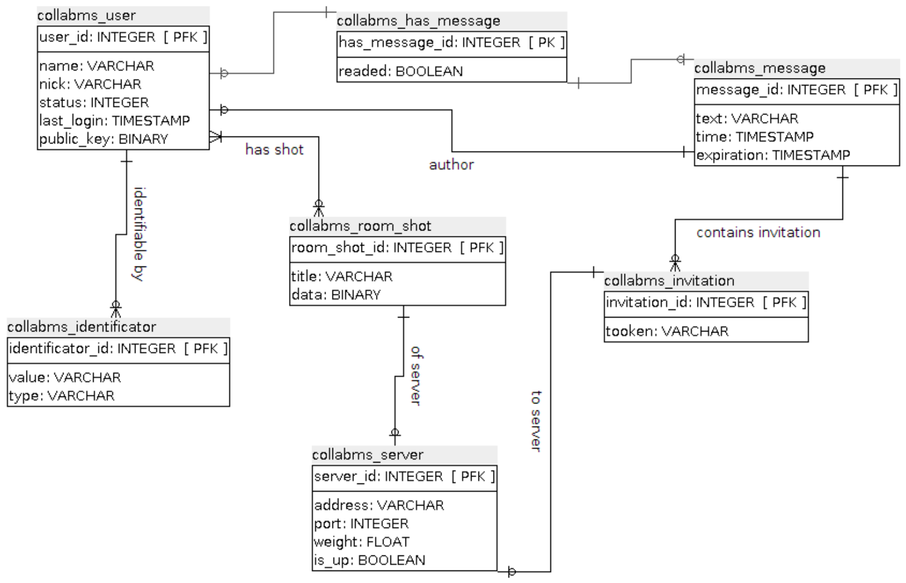
Databázové systémy

A4B33DS

Martin Indra
Czech Technical University in Prague
indrama1@fel.cvut.cz

1 Logical Modeling

Logical modeling deals with gathering purpose requirements and converting those requirements into a model. The logical model revolves around the needs of the function, not the database, although the needs of the purpose are used to establish the needs of the database. The diagrams produced should show the processes and data that exists, as well as the relationships between business processes and data.



Obrázek 1: Logical Modeling

2 SQL Script Creating Database

```
CREATE SEQUENCE collabms_has_message_id_seq;

CREATE TABLE collabms_has_message (
    has_message_id INTEGER NOT NULL
        DEFAULT nextval('collabms_has_message_id_seq'),
    readed BOOLEAN DEFAULT false NOT NULL,
    message_id INTEGER NOT NULL,
    user_id INTEGER NOT NULL,
    CONSTRAINT has_message_id PRIMARY KEY (has_message_id)
);
```

```
ALTER SEQUENCE collabms_has_message_id_seq
    OWNED BY collabms_has_message.has_message_id;

CREATE SEQUENCE collabms_message_id_seq;

CREATE TABLE collabms_message (
     message_id INTEGER NOT NULL
         DEFAULT nextval('collabms_message_id_seq'),
     text VARCHAR NOT NULL,
     time TIMESTAMP NOT NULL,
     expiration TIMESTAMP NOT NULL,
     author INTEGER NOT NULL,
     invitation INTEGER NOT NULL,
    CONSTRAINT message_id PRIMARY KEY (message_id)
);


ALTER SEQUENCE collabms_message_id_seq
    OWNED BY collabms_message.message_id;

CREATE SEQUENCE collabms_invitation_id_seq;

CREATE TABLE collabms_invitation (
     invitation_id INTEGER NOT NULL
         DEFAULT nextval('collabms_invitation_id_seq'),
     tooken VARCHAR NOT NULL,
    CONSTRAINT invitation_id PRIMARY KEY (invitation_id)
);


ALTER SEQUENCE collabms_invitation_id_seq
    OWNED BY collabms_invitation.invitation_id;

CREATE SEQUENCE collabms_user_id_seq;

CREATE TABLE collabms_user (
     user_id INTEGER NOT NULL
         DEFAULT nextval('collabms_user_id_seq'),
     name VARCHAR NOT NULL,
     nick VARCHAR NOT NULL,
     status INTEGER DEFAULT 0 NOT NULL,
     last_login TIMESTAMP NOT NULL,
     public_key BYTEA NOT NULL,
    CONSTRAINT user_id PRIMARY KEY (user_id)
);


ALTER SEQUENCE collabms_user_id_seq
    OWNED BY collabms_user.user_id;


CREATE SEQUENCE collabms_user_has_room_shot_id_seq;

CREATE TABLE collabms_user_has_room_shot (
     user_has_room_shot_id INTEGER NOT NULL
         DEFAULT nextval('collabms_user_has_room_shot_id_seq'),
     user_id INTEGER NOT NULL,
     room_shot INTEGER NOT NULL,
    CONSTRAINT user_has_room_shot_id PRIMARY KEY (user_has_room_shot_id)
);

ALTER SEQUENCE collabms_user_has_room_shot_id_seq
    OWNED BY collabms_user_has_room_shot.user_has_room_shot_id;


CREATE SEQUENCE collabms_room_shot_id_seq;

CREATE TABLE collabms_room_shot (
     room_shot_id INTEGER NOT NULL
         DEFAULT nextval('collabms_room_shot_id_seq'),
     title VARCHAR NOT NULL,
     data BYTEA NOT NULL,
     server INTEGER NOT NULL,
    CONSTRAINT room_shot_id PRIMARY KEY (room_shot_id)
);

ALTER SEQUENCE ccollabms_room_shot_id_seq
    OWNED BY collabms_room_shot.room_shot_id;

CREATE SEQUENCE collabms_server_id_seq;
CREATE TABLE collabms_server (
     server_id INTEGER NOT NULL
OWNEDDEFAULT nextval('collabms_server_id_seq'),
     address VARCHAR NOT NULL,
     port INTEGER DEFAULT 30125 NOT NULL,
     weight REAL DEFAULT 1 NOT NULL,
     is_up BOOLEAN DEFAULT true NOT NULL,
     CONSTRAINT server_id PRIMARY KEY (server_id)
);


ALTER SEQUENCE collabms_server_id_seq
     OWNED BY collabms_server.server_id;

CREATE SEQUENCE collabms_identificator_id_seq;

CREATE TABLE collabms_identificator (
     identificator_id INTEGER NOT NULL
          DEFAULT nextval('collabms_identificator_id_seq'),
     value VARCHAR NOT NULL,
     type VARCHAR DEFAULT PHONE_NUMBER NOT NULL,
     user_id INTEGER NOT NULL,
     CONSTRAINT identificator_id PRIMARY KEY (identificator_id)
);

ALTER SEQUENCE collabms_identificator_id_seq
     OWNED BY collabms_identificator.identificator_id;
```

3 Commented Referential Integrity

Table collabms_has_message is weak refferential type. If we delete user or message from database him messages should be deleted too.

```
ALTER TABLE collabms_has_message ADD CONSTRAINT user_id
FOREIGN KEY (user_id) REFERENCES collabms_user (user_id)
ON DELETE CASCADE ON UPDATE CASCADE;
```

```
ALTER TABLE collabms_has_message ADD CONSTRAINT message_id
FOREIGN KEY (user_id) REFERENCES collabms_user (user_id)
ON DELETE CASCADE ON UPDATE CASCADE;
```

Message should not be deleted after deleting user because it will be still useful for recipients.

```
ALTER TABLE collabms_message ADD CONSTRAINT author
FOREIGN KEY (author) REFERENCES collabms_user (user_id)
ON DELETE SET NULL UPDATE CASCADE;
```

Message should not be deleted after deleting invitation because it can be still useful for recipients (containing aditional message).

```
ALTER TABLE collabms_message ADD CONSTRAINT invitation
FOREIGN KEY (invitation) REFERENCES collabms_invitation (invitation_id)
ON DELETE SET NULL UPDATE CASCADE;
```

Delete user's room shots after deleting room shot or user.

```
ALTER TABLE collabms_user_has_room_shot ADD CONSTRAINT user_id
FOREIGN KEY (user_id) REFERENCES collabms_user (user_id)
ON DELETE CASCADE ON UPDATE CASCADE;
```

```
ALTER TABLE collabms_user_has_room_shot ADD CONSTRAINT room_shot
FOREIGN KEY (room_shot) REFERENCES collabms_room_shot (room_shot_id)
ON DELETE CASCADE ON UPDATE CASCADE;
```

Room shot can be still userful after server deletion.

```
ALTER TABLE collabms_room_shot ADD CONSTRAINT server
FOREIGN KEY (server) REFERENCES collabms_server (server_id)
ON DELETE SET NULL ON UPDATE CASCADE;
```

Identificator should be removed after deletion of user because it identificates just him.

```
ALTER TABLE collabms_identificator ADD CONSTRAINT user_id
FOREIGN KEY (user_id) REFERENCES collabms_user (user_id)
ON DELETE CASCADE ON UPDATE CASCADE;
```