Politechnika Wrocławska

Michael Abebe
Damian Jarek
Marta Kwolik
Piotr Luboch
Michał Mosion
Minura Nugmanova
Michał Rogóż
Adam Włodarczyk

Faculty Council Meetings

Advanced databases

Supervisor: Maciej Nikodem, PhD

Wrocław, 2016

Contents

List of Figures

List of Tables

Chapter 1

Database

1.1 Model details

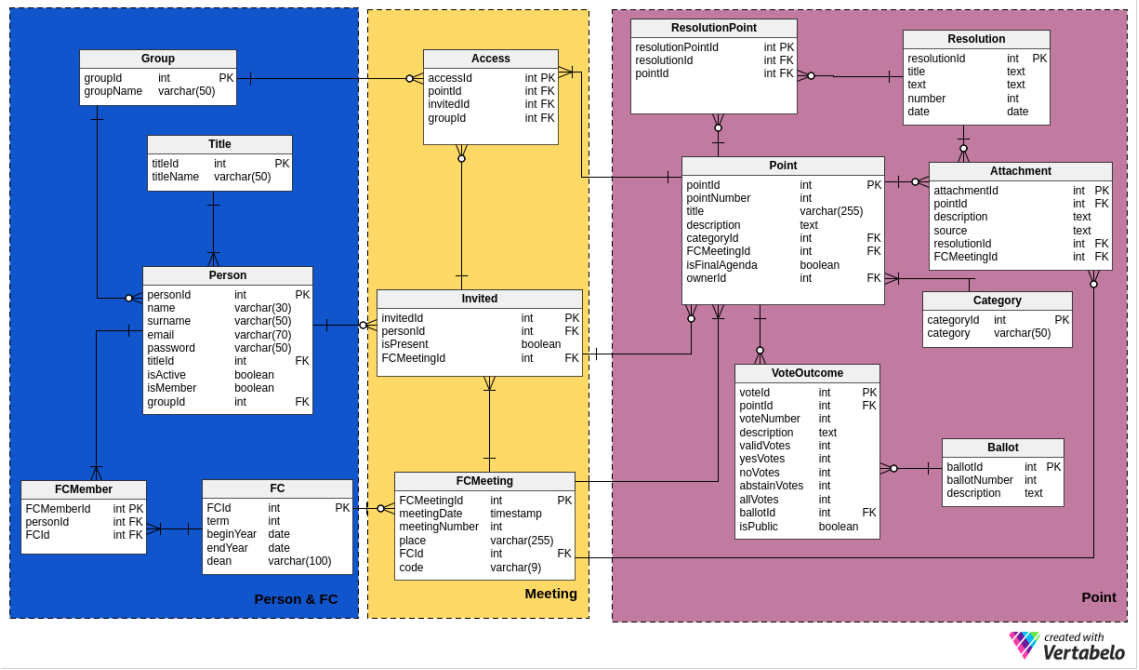

Figure 1.1: Schema of the database

Figure 1.1 shows schema of the database. Database is set in the PostrgreSQL environment and is designed to work with a high-level Python Web framework — Django.

On the left side of the figure (blue background), there are tables which contain data related with users and description of faculty council. In the middle of the figure (yellow background) there are tables describing meetings of the faculty council. At the right side, there are tables related to the point which is/was being discussed during the faculty council meeting. In this section, there are also tables which contain results of the voting and resolution of each point.

1.2 Tables

Legend used in tables properties:

- PK — Primary Key
- FK — Foreign Key
- NN — Not Null
- UQ — Unique
- AI — Auto Increment

1.2.1 Group

In `Group` table there are groups names to which each person belongs:

- smallQuorum – user belongs to the small quorum group. User is allowed to take part in small quorum voting and big quorum voting.
- bigQuorum – user belongs to the big quorum group. Is allowed to take part only in big quorum voting.
- guest – during the meeting, user takes part as a guest. Guest is not allowed to vote.
- supervisor – user who is responsible about managing with faculty council meetings. Supervisor can create votes, points, arranging meetings.

Table 1.1: Columns of Group table

Column name	**Type**	**Properties**	**Description**
groupId	int	PK, UQ, NN, AI	Titles identification number.
groupName	varchar(50)	NN, UQ	Name of the group where user belongs.

1.2.2 Title

In `Title` table, there are names of academic titles which people can achieve during their education career.

Table 1.2: Columns of Title table

Column name	**Type**	**Properties**	**Description**
titleId	int	PK, UQ, NN, AI	Titles identification number.
titleName	varchar(50)	NN, UQ	Name of the title (i.e. BSc, Eng, MSc, PhD).

1.2.3 FC

In `FC` table there are stored information about each Faculty Council term of office.

Table 1.3: Columns of FC table

Column name	Type	Properties	Description
FCId	int	PK, UQ, NN, AI	Faculty Council identification number.
term	int	NN	Defines the number of a term of office of Faculty Council.
beginYear	date	NN	Year, when Faculty Council started terms of office
endYear	date	NN	Year, when Faculty Council finished terms of office
dean	varchar(100)	NN	Information, who the dean is in each Faculty Council

1.2.4 FCMember

In `FCMember` table there are stored information about members of the Faculty Council during each cadence.

Table 1.4: Columns of FCMember table

Column name	Type	Properties	Description
FCMemberId	int	PK, UQ, NN, AI	Member of Faculty Council Meeting identification number.
personId	int	FK, NN	Foreign key from the „Person" table. Identifies . who is a member of faculty council referred by `FCId` field
FCId	int	FK, NN	Foreign key from the „FC" table. Identifies during which term of faculty council person was a member.

1.2.5 Person

In `Person` table information about users of the system are stored. It consists of details about person such as name, surname, title and information connected with login into the system (email, password). Email is stored for everyone. If there is no password (NULL), user is identified through LDAP system. If there is no user in LDAP system, it means that the user is a guest and needs a dedicated password. Due to the safety reasons and to prevent unauthorized access, password is protected with *hash()* function.

Table 1.5: Columns of Person table

Column name	Type	Properties	Description
personId	int	PK, UQ, NN, AI	Person identification number.
name	varchar(30)	NN	First name of the person.
surname	varchar(50)	NN	Last name of the person.
email	varchar(50)	UQ, NN	Unique name and email for identification of a person in the system and for login to the system.
password	varchar(50)		Encrypted password of a person.
titleId	int	FK, NN	Foreign key from the „Title" table. Identifies title of a person.
isActive	boolean	NN	Determines if the user is still member of faculty council. To keep historical data continuity, the member should not be removed but deactivated.
isMember	boolean	NN	Identifies whether the user is an administrator or not.
groupId	int	FK, NN	Foreign key from the `Group` table. Defines group membership during FCId (particular faculty council term) and consequently permisions of the person.

1.2.6 Access

In `Access` table there are stored information about access permissions.

Table 1.6: Columns of Access table

Column name	Type	Properties	Description
accessId	int	PK, UQ, NN, AI	Access identification number.
pointId	int	FK, NN	Foreign key from „Point" table. Identifiesto which point permissions are attached.
invitedId	int	FK	Foreign key from „Invited" table. Identifies people, mainly guests, who are invited for a specific point.
groupId	int	FK	Foreign key from „Group" table. Identifies which group has permissions for the following point.

1.2.7 Invited

In `Invited` table there are stored information about who is/was invited for each Faculty Council Meeting. Generally, the content of the table is imported from `FCMember` table. In case of invitation of a person, who is not a member of the Faculty Council, guest is added to the table, using `Person` table.

Table 1.7: Columns of Invited table

Column name	Type	Properties	Description
invitedId	int	PK, UQ, NN, AI	Invitation identification number.
personId	int	FK, NN	Foreign key from „Person" table. Identifies who was/is invited.
groupId	int	FK, NN	Foreign key from „Group" table. Identifies permissions of the invited person.
isPresent	boolean		Identifies if person was present during Faculty Council Meeting.
FCMeetingId	int	FK, NN	Foreign key from „FCMeeting" table. Identifies Faculty Council Meeting for which invitation was made.

1.2.8 FCMeeting

In `FCMeeting` table there are stored information about organized Faculty Council Meeting. It consists of information such as date, place and owner of that meeting.

Table 1.8: Columns of FCMeeting table

Column name	Type	Properties	Description
FCMeetingId	int	PK, UQ, NN, AI	FCMeeting identification number.
meetingDate	timestamp	UQ, NN	Date, when the faculty council meeting is.
meetingNumber	int	NN	Number of the meeting.
place	varchar(255)	NN	Place, where the meeting takes place.
FCId	int	FK, NN	Foreign key from „FC" table. Identifies during which Faculty Council the meeting took place.
code	varchar(9)	UQ	Unique identification code of the meeting. The format is as following: *meetingNumber*/*year*.

1.2.9 ResolutionPoint

In `ResolutionPoint` table there are stored information about points resolutions.

Table 1.9: Columns of Access table

Column name	Type	Properties	Description
resolutionPointId	int	PK, UQ, NN, AI	ResolutionPoint identification number.
resolutionId	int	FK, NN	Foreign key from „Resolution" table. Identifies resolution of the point.
pointId	int	FK, NN	Foreign key from „Point" table. Identifies which point resolution is about.

1.2.10 Resolution

In `Resolution` table there are stored information about confirmed solutions taken during Faculty Council Meeting.

Table 1.10: Columns of Resolution table

Column name	Type	Properties	Description
resoltuionId	int	PK, UQ, NN, AI	Resolution identification number.
title	text	NN	Title of the resolution.
text	text	NN	Content of the resolution.
number	int		Number of the resolution.
date	date	NN	Date, when the resolution was confirmed.

1.2.11 Category

In `Category` table there are stored names of the points categories (i.e. scholarship, renovations, students organisation). It is related with the `Point` table.

Table 1.11: Columns of Category table

Column name	Type	Properties	Description
categoryId	int	PK, UQ, NN, AI	Category identification number.
category	varchar(50)	NN	Name of the category.

1.2.12 Attachment

In `Attachment` table there are stored information about attachments. Table consists the source path of the file and short description about the file. Attachments can be linked to Point, Resolution and Faculty Council Meeting. It is possible, that the attachment can be include in `Point`, as well as in the `Resolution` and/or `FCMeeting` at the same time. If the attachment is not include in a section, it takes `NULL` value.

Table 1.12: Columns of Attachment table

Column name	Type	Properties	Description
attachmentId	int	PK, UQ, NN, AI	Attachment identification number.
pointId	int	FK	Foreign key from the „Point" table. Identifies to which point attachment belongs.
description	text		Describes what does the attachment contain.
source	text	NN	Identifies source path to the attachment on a server.
resolutionId	int	FK	Foreign key from the „ResolutionPoint" table. Identifies to which resolution of the point attachment belongs.
FCMeetingId	int	FK	Foreign key from the „FCMeeting" table. Identifies to which Faculty Council Meeting attachment belongs.

1.2.13 Ballot

In `Ballot` table there are stored information about voting that appears on one ballot.

Table 1.13: Columns of Ballot table

Column name	Type	Properties	Description
ballotId	int	PK, UQ, NN, AI	Ballot identification number.
ballotNumber	int	NN	Number of the voting for the point.
description	text		Header of the voting ballot.

1.2.14 Point

In `Point` table there are stored information about each point which is going to be or was discussed during faculty council meeting. It consists of information such as title, short description and who is the owner.

Table 1.14: Columns of Point table

Column name	Type	Properties	Description
pointId	int	PK, UQ, NN, AI	Point identification number.
pointNumber	int	NN	Number of the point to be discussed during faculty council meeting.
title	text	NN	Name of the point.
description	text		Description, what the point is about.
categoryId	int	FK, NN	Foreign key from the „Category" table.
FCMeetingId	int	FK, NN	Foreign key from the „FCMeeting" table. It describes on which FCMeeting the point is discuses.
isFinalAgenda	boolean		Identifies if the date is confirmed and final.
ownerId	int	FK, NN	Foreign key from „Person" table. Identifies who is the owner of the point.

1.2.15 VoteOutcome

In `VoteOutcome` there are stored information about results of the voting for each point. It stores information about ballot and if voting was public or not.

Table 1.15: Columns of VoteOutcome table

Column name	Type	Properties	Description
voteId	int	PK, UQ, NN, AI	Point identification number.
pointId	int	FK, NN	Foreign key from the „Point" table. Identifies which point was voted.
voteNumber	int	NN	Number of the vote for each point.
description	text		Description of the voting.
validVotes	int		How many votes were valid.
yesVotes	int		How many votes were „yes".
noVotes	int		How many votes were „no".
abstainVotes	int		How many people abstained.
allVotes	int		Number of all votes.
ballotId	int	FK	Foreign key from the „Ballot" table. Identifies generated ballot.
isPublic	boolean	NN	Identifies if the vote was public or not.

Chapter 2

Configuration

2.1 Environment

Recommended OS is Debian 8. System dependencies which are required for correctly working application:

- build-essential
- libxml2-dev
- libxslt1-dev
- postgresql
- python
- python-dev
- python-pip
- libpq-dev
- python-psycopg2
- nginx

```
apt-get install build-essential libxml2-dev libxslt1-dev postgresql python python-dev python-pip
    ↪ libpq-dev python-psycopg2 nginx
```

Application should be extracted into `/opt/fc-meeting-app/` directory. This directory should be allowed to be readable by www-data (default user for HTTP servers in Debian OS).

Requirements can be installed with following command:

```
cd /opt/fc-meeting-app && pip install -r requirements.txt
```

To ensure if everything was installed correctly:

```
python manage.py collectstatic --noinput
```

Configuration files:

```
cp nginx.conf /etc/nginx/nginx.conf && \
cp default-nginx /etc/nginx/sites-enabled/default-nginx.conf && \
cp fc-meeting-app-nginx.conf /etc/nginx/sites-enabled/fc-meeting-app-nginx.conf && \
cp fc-meeting-app.service /etc/systemd/system/fc-meeting-app.service
```

SSL keys and certifications should be copied into `/etc/ssl/cert/` directory with names `fc-meeting-app.*`.

2.2 Database

Database was designed for PostregsSQL environment. There is a need to create database variable file. To do so:

```
echo 'DATABASE_URL="postgres://fc_meeting_app_user@/fcmeetingapp"' > /opt/fc-meeting-app/env
```

After these steps, system should reload its units.

```
systemctl daemon-reload
```

There is a need to enable required daemons to run on startup. They should also be restarted to reload configuration files.

```
systemctl enable nginx.service postgresql.service fc-meeting-app.service && \
systemctl restart nginx.service nginx.service postgresql.service fc-meeting-app.service
```

To check if they started successfully:

```
systemctl status nginx.service nginx.service postgresql.service fc-meeting-app.service
```

If the database is empty, the administrator should be crated.

```
cd /opt/fc-meeting-app && python manage.py createsuperuser
```

Chapter 3

Database data

3.1 Backup

Attachments are stored in `/opt/fc-meeting-app/uploads` and they need to be secured manually.

Database contents can be dumped to the output-file:

```
pg_dump database_name > output-file
```

where `database_name` is name of the database.

3.2 Restore

To restore the data in database:

```
pg_dump fcmeetingapp > outfile
```

Attachments files should be copied back to the `/opt/fc-meeting-app/uploads`.

3.3 Migrations

Migrate generates a system version of tracking changes in the database structure by creating migrations scripts to provide backwards compatibility. It automatically detects changes in models and existing migrations scripts and translates them into changes in the database structure. More about migrations in django can be found on https://docs.djangoproject.com/en/1.8/topics/migrations.

Migration can be done with following command:

```
cd /opt/fc-meeting-app && python manage.py migrate
```