Garlic – Social Media

3EHIF
2015/2016

Team: Michael Bartl, Maximilian Meyer-Mölleringhof

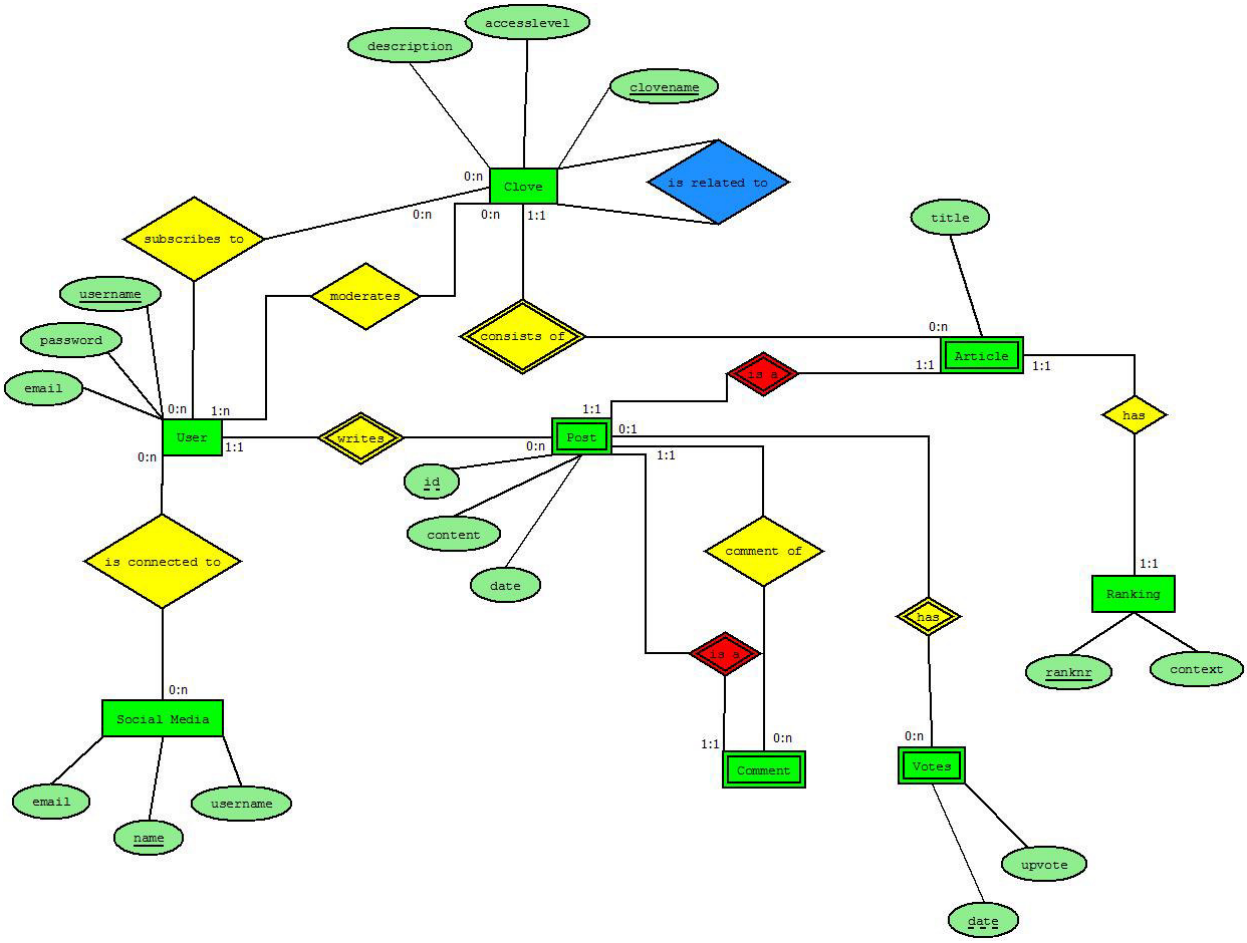
Description:

Garlic is a social network that can be accessed over a local application on your Windows Computer. A new user just needs to sign up with his or her e-mail address and password and can immediately start to write some content. The content being created is divided into categories (cloves = Knoblauchzehen) which contain articles that users can read and comment on. If a user really likes an article or a comment, he or she can easily upvote it. All upvotes are stored and in the end lead to a certain rank.
Users can subscribe as well as moderate a clove. Subscribing means that the user gets notifications as soon as there is new content being posted. Moderating means that the user has full access to the clove and can edit / delete / add content that a regular user cannot.
As Garlic is not the only social media out there, the user can connect his or her other social accounts with it. This is important for sharing content and can also be used for notifications.

Table of Contents

Michael Bartl
Maximilian Meyer-Mölleringhof

1. Entity Relationship Diagram – Extended Chen-Notation



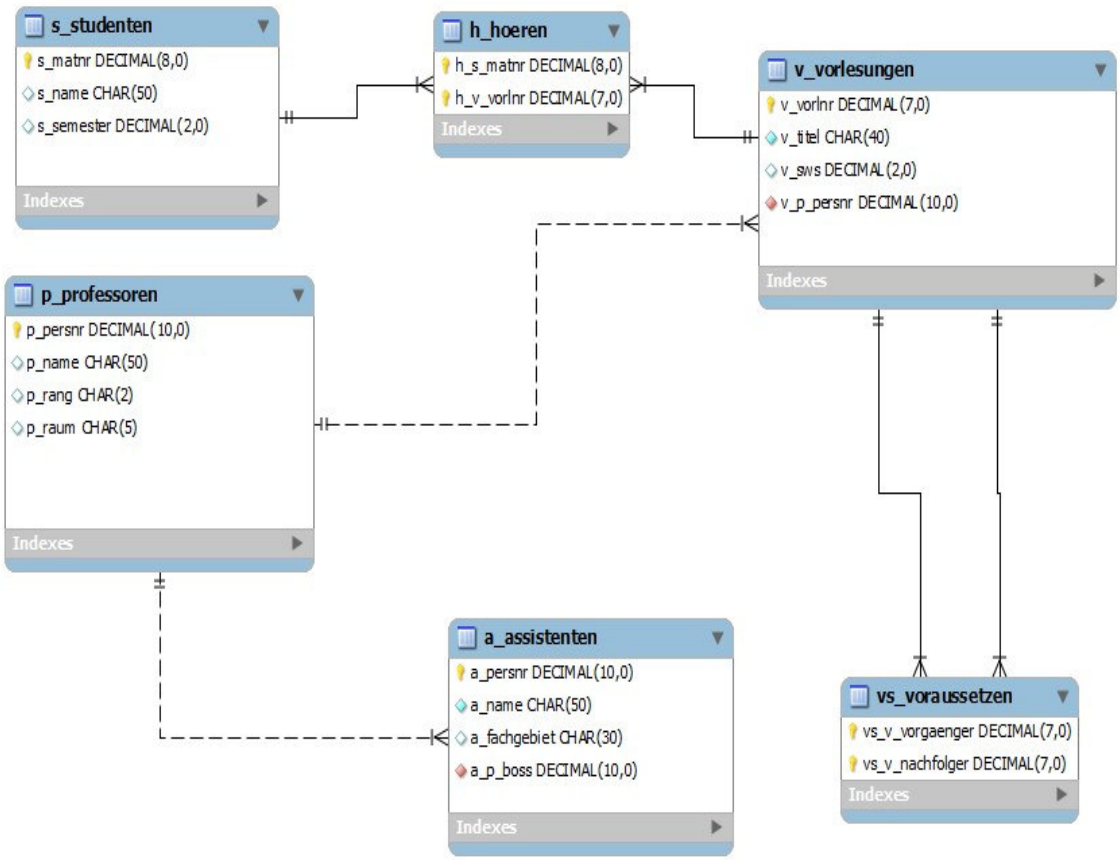
Michael Bartl
Maximilian Meyer-Mölleringhof

Description of tables and attributes

Entity-Set	Attributes	Description	Datatype	Constraint
u_users	username	Identifying username	varchar	Length: 20
	email	E-Mail of the user	varchar	Length: 50
	password	Password for the user	varchar	Length: 100
p_posts	id	Identifying ID for each post	int	
	content	The posts text	varchar	Length: 10000
	date	Date when the post has been posted	datetime	
a_articles	title	The title of the article	varchar	Length: 200
co_comments				
c_cloves	name	Name of the Clove	varchar	Length:50
	access	Public / private	bool	
	description	Description of the Clove	varchar	Length:1000
sm_socialmedias	name	Name of the social media	varchar	Length: 20
	email	Email the users uses for that social media	varchar	Length: 50
	username	Username the users uses for that social media	varchar	Length: 50
r_ranking	ranknr	The rank number of the article	int	
	context	The context of the rank (Recent, Rising, …)	varchar	Length: 50
v_votes	upvote	Whether the vote is up or down	bool	
	date	The date the vote was submitted	datetime	

Michael Bartl
Maximilian Meyer-Mölleringhof

2. Relational Model-MySQL Workbench



3. Data Description Language: Alle Befehle zum Erzeugen der Datenbank

```
-- MySQL Script generated by MySQL Workbench
-- 02/27/16 14:28:14
-- Model: New Model    Version: 1.0
-- MySQL Workbench Forward Engineering

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL,ALLOW_INVALID_DATES';

-- -----------------------------------------------------
-- Schema mydb
-- -----------------------------------------------------

-- -----------------------------------------------------
-- Schema mydb
-- -----------------------------------------------------
```

```
CREATE SCHEMA IF NOT EXISTS `mydb` DEFAULT CHARACTER SET utf8 COLLATE utf8_general_ci ;
USE `mydb` ;

-- -----------------------------------------------------
-- Table `mydb`.`u_users`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`u_users` (
  `u_username` VARCHAR(20) NOT NULL COMMENT '',
  `u_email` VARCHAR(50) NOT NULL COMMENT '',
  `u_password` VARCHAR(100) NOT NULL COMMENT '',
  PRIMARY KEY (`u_username`)  COMMENT '',
  UNIQUE INDEX `u_username_UNIQUE` (`u_username` ASC)  COMMENT '')
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`p_posts`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`p_posts` (
  `p_id` INT NOT NULL COMMENT '',
  `p_content` VARCHAR(10000) NOT NULL COMMENT '',
  `p_date` DATETIME NOT NULL COMMENT '',
  `p_u_username` VARCHAR(20) NOT NULL COMMENT '',
  PRIMARY KEY (`p_id`)  COMMENT '',
  INDEX `fk_p_posts_u_users_idx` (`p_u_username` ASC)  COMMENT '',
  CONSTRAINT `fk_p_posts_u_users`
    FOREIGN KEY (`p_u_username`)
    REFERENCES `mydb`.`u_users` (`u_username`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`co_comments`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`co_comments` (
  `co_p_id` INT NOT NULL COMMENT '',
  `co_p_commentof` INT NOT NULL COMMENT '',
  PRIMARY KEY (`co_p_id`)  COMMENT '',
  INDEX `fk_co_comments_p_posts2_idx` (`co_p_commentof` ASC)  COMMENT '',
  CONSTRAINT `fk_co_comments_p_posts1`
    FOREIGN KEY (`co_p_id`)
    REFERENCES `mydb`.`p_posts` (`p_id`)
    ON DELETE NO ACTION
```

```
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_co_comments_p_posts2`
    FOREIGN KEY (`co_p_commentof`)
    REFERENCES `mydb`.`p_posts` (`p_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`c_cloves`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`c_cloves` (
  `c_name` VARCHAR(50) NOT NULL COMMENT '',
  `c_access` TINYINT(1) NOT NULL COMMENT '',
  `c_description` VARCHAR(1000) NULL COMMENT '',
  PRIMARY KEY (`c_name`)  COMMENT '')
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`sm_socialmedias`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`sm_socialmedias` (
  `sm_name` VARCHAR(20) NOT NULL COMMENT '',
  `sm_username` VARCHAR(50) NOT NULL COMMENT '',
  `sm_email` VARCHAR(50) NOT NULL COMMENT '',
  PRIMARY KEY (`sm_name`)  COMMENT '')
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`r_rankings`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`r_rankings` (
  `r_rank` INT NOT NULL COMMENT '',
  `r_context` VARCHAR(50) NOT NULL COMMENT '',
  PRIMARY KEY (`r_rank`)  COMMENT '')
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`v_votes`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`v_votes` (
```

```
  `v_upvote` TINYINT(1) NOT NULL COMMENT '',
  `v_date` DATETIME NOT NULL COMMENT '',
  `v_p_id` INT NOT NULL COMMENT '',
  PRIMARY KEY (`v_date`, `v_p_id`)  COMMENT '',
  INDEX `fk_v_votes_p_posts1_idx` (`v_p_id` ASC)  COMMENT '',
  CONSTRAINT `fk_v_votes_p_posts1`
    FOREIGN KEY (`v_p_id`)
    REFERENCES `mydb`.`p_posts` (`p_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`a_articles`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`a_articles` (
  `a_p_id` INT NOT NULL COMMENT '',
  `a_c_clove` VARCHAR(50) NOT NULL COMMENT '',
  `a_title` VARCHAR(200) NOT NULL COMMENT '',
  `a_r_rank` INT NOT NULL COMMENT '',
  PRIMARY KEY (`a_p_id`, `a_c_clove`, `a_r_rank`)  COMMENT '',
  INDEX `fk_a_articles_c_cloves1_idx` (`a_c_clove` ASC)  COMMENT '',
  INDEX `fk_a_articles_r_rankings1_idx` (`a_r_rank` ASC)  COMMENT '',
  CONSTRAINT `fk_a_articles_p_posts1`
    FOREIGN KEY (`a_p_id`)
    REFERENCES `mydb`.`p_posts` (`p_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_a_articles_c_cloves1`
    FOREIGN KEY (`a_c_clove`)
    REFERENCES `mydb`.`c_cloves` (`c_name`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_a_articles_r_rankings1`
    FOREIGN KEY (`a_r_rank`)
    REFERENCES `mydb`.`r_rankings` (`r_rank`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`csm_connectedsocialmedia`
-- -----------------------------------------------------
```

```
CREATE TABLE IF NOT EXISTS `mydb`.`csm_connectedsocialmedia` (
 `csm_u_username` VARCHAR(20) NOT NULL COMMENT '',
 `csm_sm_name` VARCHAR(20) NOT NULL COMMENT '',
 PRIMARY KEY (`csm_u_username`, `csm_sm_name`)  COMMENT '',
 INDEX `fk_u_users_has_sm_socialmedias_sm_socialmedias1_idx` (`csm_sm_name` ASC)  COMMENT
'',
 INDEX `fk_u_users_has_sm_socialmedias_u_users1_idx` (`csm_u_username` ASC)  COMMENT '',
 CONSTRAINT `fk_u_users_has_sm_socialmedias_u_users1`
   FOREIGN KEY (`csm_u_username`)
   REFERENCES `mydb`.`u_users` (`u_username`)
   ON DELETE NO ACTION
   ON UPDATE NO ACTION,
 CONSTRAINT `fk_u_users_has_sm_socialmedias_sm_socialmedias1`
   FOREIGN KEY (`csm_sm_name`)
   REFERENCES `mydb`.`sm_socialmedias` (`sm_name`)
   ON DELETE NO ACTION
   ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`s_subscription`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`s_subscription` (
 `s_u_username` VARCHAR(20) NOT NULL COMMENT '',
 `s_c_clovename` VARCHAR(50) NOT NULL COMMENT '',
 PRIMARY KEY (`s_u_username`, `s_c_clovename`)  COMMENT '',
 INDEX `fk_u_users_has_c_cloves_c_cloves1_idx` (`s_c_clovename` ASC)  COMMENT '',
 INDEX `fk_u_users_has_c_cloves_u_users1_idx` (`s_u_username` ASC)  COMMENT '',
 CONSTRAINT `fk_u_users_has_c_cloves_u_users1`
   FOREIGN KEY (`s_u_username`)
   REFERENCES `mydb`.`u_users` (`u_username`)
   ON DELETE NO ACTION
   ON UPDATE NO ACTION,
 CONSTRAINT `fk_u_users_has_c_cloves_c_cloves1`
   FOREIGN KEY (`s_c_clovename`)
   REFERENCES `mydb`.`c_cloves` (`c_name`)
   ON DELETE NO ACTION
   ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`ad_admins`
-- -----------------------------------------------------
```

```
CREATE TABLE IF NOT EXISTS `mydb`.`ad_admins` (
  `ad_u_username` VARCHAR(20) NOT NULL COMMENT '',
  `ad_c_clovename` VARCHAR(50) NOT NULL COMMENT '',
  PRIMARY KEY (`ad_u_username`, `ad_c_clovename`)  COMMENT '',
  INDEX `fk_u_users_has_c_cloves_c_cloves2_idx` (`ad_c_clovename` ASC)  COMMENT '',
  INDEX `fk_u_users_has_c_cloves_u_users2_idx` (`ad_u_username` ASC)  COMMENT '',
  UNIQUE INDEX `ad_u_username_UNIQUE` (`ad_u_username` ASC)  COMMENT '',
  UNIQUE INDEX `c_cloves_c_name_UNIQUE` (`ad_c_clovename` ASC)  COMMENT '',
  CONSTRAINT `fk_u_users_has_c_cloves_u_users2`
    FOREIGN KEY (`ad_u_username`)
    REFERENCES `mydb`.`u_users` (`u_username`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_u_users_has_c_cloves_c_cloves2`
    FOREIGN KEY (`ad_c_clovename`)
    REFERENCES `mydb`.`c_cloves` (`c_name`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`re_realtedcloves`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`re_realtedcloves` (
  `re_c_clovename1` VARCHAR(50) NOT NULL COMMENT '',
  `re_c_clovename2` VARCHAR(50) NOT NULL COMMENT '',
  PRIMARY KEY (`re_c_clovename1`, `re_c_clovename2`)  COMMENT '',
  INDEX `fk_c_cloves_has_c_cloves_c_cloves2_idx` (`re_c_clovename2` ASC)  COMMENT '',
  INDEX `fk_c_cloves_has_c_cloves_c_cloves1_idx` (`re_c_clovename1` ASC)  COMMENT '',
  CONSTRAINT `fk_c_cloves_has_c_cloves_c_cloves1`
    FOREIGN KEY (`re_c_clovename1`)
    REFERENCES `mydb`.`c_cloves` (`c_name`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_c_cloves_has_c_cloves_c_cloves2`
    FOREIGN KEY (`re_c_clovename2`)
    REFERENCES `mydb`.`c_cloves` (`c_name`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
```

```
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

4. Views

```
# get all votes from articles / comments as well as the total of votes of a user
drop view if exists vUserVotes;
create view vUserVotes as
select u.u_username as uu_user,
        (
                select count(*)
                from a_articles a inner join (p_posts p inner join v_votes v
                                on p.p_id = v.v_p_post)
                        on a.a_p_id = p.p_id
                                where u.u_username like p.p_u_username
         ) as uu_articles_votes,
        (
        select count(*)
        from c_comments c inner join (p_posts p inner join v_votes v
                                on p.p_id = v.v_p_post)
                on c.c_p_id = p.p_id
                where u.u_username like p.p_u_username
         ) as uu_comments_votes,
        (
         select count(*)
         from v_votes v inner join p_posts p
                on v.v_p_post = p.p_id
                where u.u_username like p.p_u_username
         ) as uu_total_votes
from u_users u;


drop view if exists vUserRankings;
create view vUserRankings as
select u.u_username,
(
 select count(*)
 from p_posts p inner join a_articles a
                on p.p_id = a.a_p_id
        where a.a_r_rank between 1 and 499 and p.p_u_username like u.u_username
 ) as ur_superhot,
(
 select count(*)
 from p_posts p inner join a_articles a
                        on p.p_id = a.a_p_id
        where a.a_r_rank between 500 and 999 and p.p_u_username like u.u_username
 ) as ur_hot,
```

```
(
 select count(*)
 from p_posts p inner join a_articles a
                    on p.p_id = a.a_p_id
          where a.a_r_rank between 1000 and 1499 and p.p_u_username like u.u_username
 ) as ur_rising,
(
 select count(*)
 from p_posts p inner join a_articles a
                    on p.p_id = a.a_p_id
          where a.a_r_rank between 1500 and 2000 and p.p_u_username like u.u_username
 ) as ur_upcoming,
(
 select count(*)
 from p_posts p inner join a_articles a
                    on p.p_id = a.a_p_id
          where p.p_u_username like u.u_username
 ) as ur_total
from u_users u;


# get every post with the user who created it
# as well as all the votes the post has gotten so far
drop view if exists vPostInfo;
create view vPostInfo as
select p.p_id as pi_postID, p.p_date as pi_postDate, p.p_content as pi_postContent,
(
 select u.u_username
 from u_users u
 where u.u_username = p.p_u_username
 ) as pi_user,
(
 select count(*)
 from v_votes v
 where v.v_p_post = p.p_id
 ) as pi_votes,
 (
  select count(*)
  from c_comments c
  where c.c_p_commentOf = p.p_id
 ) as pi_comments,
 (
  select a.a_title
  from a_articles a
  where a.a_p_id = p.p_id
```

```
 ) as pi_postTitle
from p_posts p
order by pi_postID asc;
```

 # get the number of subscribers and admins per clove

```
drop view if exists vCloveInfo;
create view vCloveInfo as
select c.c_id as ci_cloveID, c.c_name as ci_cloveName,
(
 select count(*)
 from s_subscriptions s
 where s.s_c_clove = c.c_id
 ) as ci_subscribers,
(
 select count(*)
 from ad_admins ad
 where ad.ad_c_clove = c.c_id
 ) as ci_admins,
(
 select count(*)
 from a_articles a
 where a.a_c_clove = c.c_id
 ) as ci_articles
from c_clove c;
```

 # select all the data needed for the homepage of the asp.net client

```
drop view if exists vCloveArticles;
create view vCloveArticles as
select a.a_p_id, a.a_title, a.a_c_clove, p.p_u_username,
(
 select c.c_name
 from c_clove c
 where c.c_id = a.a_c_clove
 ) as cloveName,
(
 select c.c_description
 from c_clove c
 where c.c_id = a.a_c_clove
 ) as cloveDesc,
(
 select count(*)
 from c_comments co
 where co.c_p_commentOf = p.p_id
```

```
 ) as commentCount,
(
 select count(*)
 from v_votes v
 where v.v_p_post = p.p_id
 ) as voteCount
from p_posts p inner join a_articles a
            on p.p_id = a.a_p_id;
```

Michael Bartl
Maximilian Meyer-Mölleringhof

5. WPF Projekt