Modeling a Mobile Social Media Application

Matthew Flickner
Loyola Marymount University
CMSI486

December 15, 2015

Contents

Chapter 1

Description of Enterprise

Matthew Flickner wants build an application to help people manage their responsibilities within a group of friends. He wants to use it to make tasks like household chores easier and more convenient while holding each person accountable for their responsibilities. He needs a reliable database to support the application and store application and user data. Matt needs to store user information for login. There is also a need for keeping track of what users are in what groups and whom the users are friends with. He also needs to track what tasks are in what groups and which user is responsible for completing a given task. Tasks can also have deadlines for when they need to be completed by and users can get points for completing tasks. A user gets one point for completing a task if they are the responsible party and 2 points if they volunteer to complete the task. A user can pay 5 points to get out of doing a particular task. Each user has a their own points within a given group. When the task is completed, a new person responsible is assigned or the task is closed.
The following is an initial list of questions that the database for Matt's application will need to answer:

- What groups is user John Smith a member of?
- What tasks are in the group "House Stuff"?
- Who is the person responsible for completing the task, "Empty the Dishwasher," in the group, "House Stuff"?
- Is the "Wipe off the counters" a one-time task or a repeating task?
- How many points does user Logan Couture have allocated for completing the task "Take out the trash"?
- Does user Patrick Marleau have less or more points in the task "Take out the trash" than Logan Couture?
- Is Marshall Shady a friend of George Clooney?
- Who is in the group "Stuff to Do"?
- When does task "Clean Sink" in the group "Kitchen" need to be completed by?

Chapter 2

Definition of the Environment

2.1 Input and Report Forms

User Creation Form

- First Name
- Last Name
- Username
- Password
- Email Address

User Login View

- Username
- Password

UserGroup Creation Form

- GroupId
- UserId
- isGroupAdmin

Group Creation View

- Group Name
- Group Members (usernames)

Leave Group View

- Group Id
- Group Members
- User Group

Group Deletion View

- Group Name
- Group Members (usernames)
- Group Tasks
- Group Id

Task Creation Form

- Task Name
- Task Members
- Group That Task Belongs To
- Person Responsible For Completion

Task Completion View

- Task Name
- Task Members
- Group That Task Belongs To
- Person Who Completes Task
- Person Responsible for Next completion
- Points For Task

Volunteer for Task Completion View

- Task Name
- Task Members
- Group That Task Belongs To
- Person Responsible For Completion

Opt-Out of Task View

- Task Id
- User's Point in Task
- New Person Responsible For Completion

2.2 Assumptions

- Group Members and Task Members are not always the same list of people
- Person Responsible For Completion is one person in Task Members
- Usernames are unique
- Usernames contain must be between 4-16 characters and be composed only of letters and numbers
- Only one account is allowed per email address
- Passwords must be at least 6 characters
- 1 Point is awarded for the completion of the task when the user is the Person Responsible For Completion
- 2 Points are awarded for the completion of the task when the user volunteers to be the Person Responsible For Completion.

2.3 User-oriented data dictionary

Datum	Information Definition
user_id	The unique id given to a user
username	The unique identifier of a user
first_name	The first name of the user
last_name	The last name of the user
password	The password of the user
email	The email address of the user
user_group_id	The unique id given to a user within a group
is_group_admin	Boolean that tells whether a user is a admin of their group
group_id	The unique id given to a user
group_name	The name of a group
task_id	The unique id of a given task
task_name	The name of the task
user_group_task_id	The unique id of a user in a group for a particular task.
points	The number of points a user has for a given task within a group.
is_desi	Boolean that tells whether or not a person is responsible for a given task.
task_action_id	The unique id of an action a user performs in a group for a specific task.
completed_date	The date of the last time a task was completed.
volunteered_completion	The date task's volunteer completed.
opt_out	The date of the last time a user opted out of a given task in a given group.
friendship_id	The unique id of a friendship between two users

Cross-reference table

Datum	Form or Screen

	User Creation Form	User Login View	Group Creation View	Leave Group View	Group Deletion View	Task Creation Form	Task Completion View	Volunteer for Task Completion View	Opt-Out of Task View	Add Friend Form
user_id	x	x		x						x
username	x	x	x							
first_name	x									
last_name	x							x		
password	x	x								
email_address	x	x								
group_id			x	x	x					
group_name			x							
user_group_id			x	x	x					
is_group_admin				x	x					
task_id						x	x	x	x	
task_name						x				
user_group_task_id						x	x	x	x	
points						x	x	x	x	
is_desi						x	x	x	x	
friendship_id										x
task_action_id							x	x	x	
date_completed							x			
volunteer_completed								x		
opt_out									x	

Chapter 3

Enterprise Database Design

3.1 Logical Model of the Enterprise

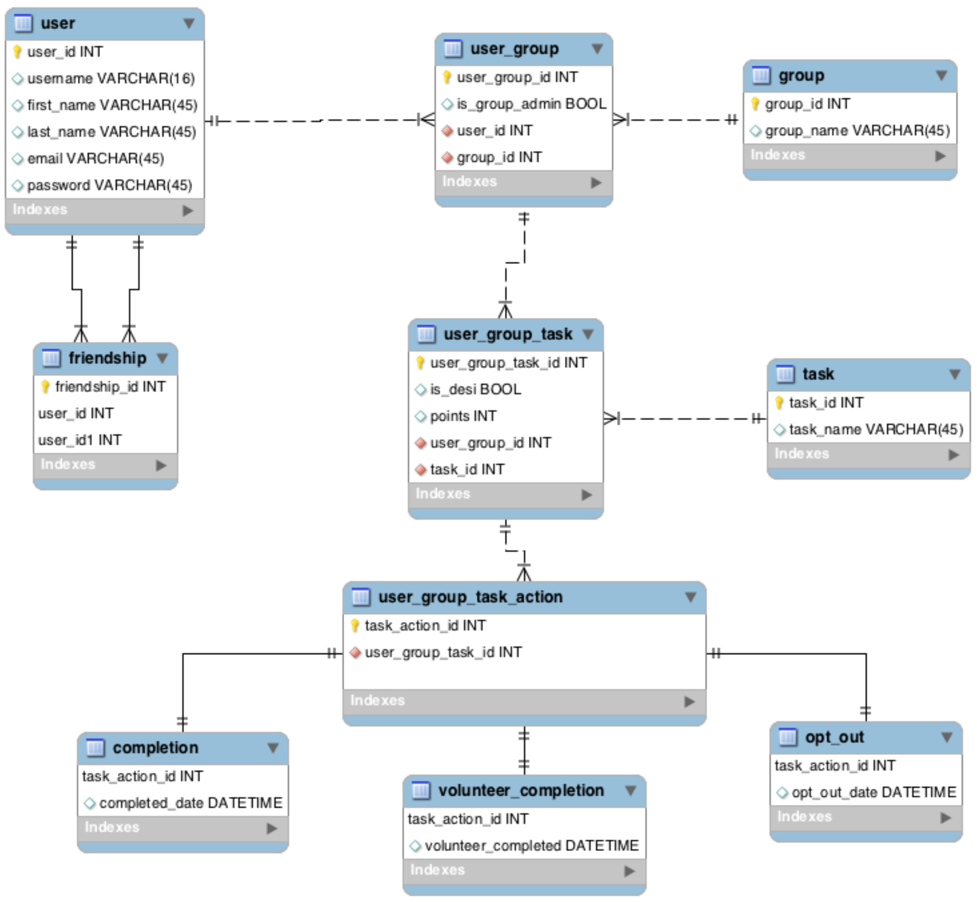
3.1.1 List of Entities and Attributes

Entity	Attributes
user	user_id(PK), username, first_name, last_name, password, email_address
friend	friend_id(PK), user_id(FK)
group	group_id(PK), group_name,
task	task_id(PK), group_id, desi(FK), desi_index
task_action	task_action_id(PK), user_group_task_id(FK)
completion	task_action_id(PK, FK), completed_date
volunteer_completion	task_action_id(PK, FK), volunteer_completed
opt_out	task_action_id(PK, FK), opt_out_date

3.1.2 List of Relationships and Attributes

Relationship	Attributes
user_group	user_group_id(PK), group_id(FK), user_id(FK), is_group_admin
user_group_task	user_group_task_id(PK), user_group_id(FK), task_id(FK), group_id(FK), is_desi, points
friendship	friendship_id(PK), user_id(FK), user1_id(FK)

3.1.3 Entity-Relationship Diagram of the Enterprise



3.2 Conceptual Model of the Enterprise

Table 3.1: User Entity

User	
user_id	PK, CK
username	CK
password	
first_name	
last_name	
email	CK

Table 3.2: Group Entity

Group	
group_id	PK, CK
group_name	

Table 3.3: Task Entity

Task	
task_id	PK, CK
task_name	

Table 3.4: User_Group Entity

User_Group	
user_group_id	PK, CK
user_id	FK references user.user_id
group_id	FK references group.group_id
is_group_admin	

Table 3.5: User_Group_Task Entity

User_Group_Task	
user_group_task_id	PK, CK
user_group_id	FK references user_group.user_group_id
task_id	FK references task.task_id
is_desi	
points	

Table 3.6: Friendship Entity

Friendship	
friendship_id	PK, CK
user_id	FK references user.user_id
user1_id	FK references user.user_id

3.3 Table Dictionary

Table	**Attributes**	**Description**
User	user_id (PK), username, first_name, last_name, password	A user of the application.
Friendship	friendship_id(PK), user_id (FK), user_id1 (FK)	A friendship between 2 users of the application.
Group	group_id (PK), group_name	A group of users that contains tasks.
Task	task_id (PK), task_name	A task to be completed by a given user in a group.
User_Group	user_group_id (PK), user_id (FK), group_id (FK), is_group_admin	Relationship between users and groups.
User_Group_Task	user_group_task_id (PK), user_group_id (FK), task_id (FK), is_desi, points	Relationship between usergroups and tasks.
Group	friendship_id (PK), user_id (FK), friend_id (FK)	Relationship between two users.
Task_Action	task_action_id (PK), user_group_task_id (FK)	The act within a task.
Completion	task_action_id (FK), completion_time	The completion a task by the desi.
Volunteer_Completion	task_action_id (FK), volunteer_completed	The completion a task by a volunteer.

3.4 Attribute Dictionary

Attribute	**Description**	**Tables Found In**
user_id	Primary key for users	User, User_Group (FK), Friendship (FK)
username	Unique identifier of a user	User
first_name	The first name of a user.	User
last_name	The last name of a user.	User
password	The password of a user.	User
email_address	The email of a user.	User
group_id	Primary key for groups.	Group, User_Group (FK), User_Group_Task (FK)
group_name	The name of a group.	Group
is_group_admin	Boolean that says whether a user is a group admin.	User_Group
user_group_id	Primary key for a user_group.	User_Group, User_Group_Task (FK)
user_group_task_id	Primary key for a user_group_task.	User_Group_Task
points	Number of points a user has for a given task.	User_Group_Task
is_desi	Boolean that say whether a user is the person responsible for completing a given task.	User_Group_Task
task_name	The name of a task.	Task
task_id	The primary key of a task.	Task, User_Group_Task (FK)
friendship_id	Primary key of a friendship.	Friendship

Chapter 4

Database and Query Definition

4.1 Database Definition

```
-- MySQL Script generated by MySQL Workbench
-- Sun Dec 13 16:12:58 2015
-- 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 cmsi486_mflickner
-- -----------------------------------------------------

-- -----------------------------------------------------
-- Schema cmsi486_mflickner
-- -----------------------------------------------------
CREATE SCHEMA IF NOT EXISTS `cmsi486_mflickner` DEFAULT CHARACTER
    SET utf8 ;
USE `cmsi486_mflickner` ;

-- -----------------------------------------------------
-- Table `cmsi486_mflickner`.`user`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `cmsi486_mflickner`.`user` (
   `user_id` INT NOT NULL AUTO_INCREMENT,
   `username` VARCHAR(16) NULL,
   `first_name` VARCHAR(45) NULL,
   `last_name` VARCHAR(45) NULL,
   `email` VARCHAR(45) NULL,
   `password` VARCHAR(45) NULL,
  PRIMARY KEY (`user_id`))
```

```
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `cmsi486_mflickner`.`group`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `cmsi486_mflickner`.`group` (
  `group_id` INT NOT NULL AUTO_INCREMENT,
  `group_name` VARCHAR(45) NULL,
  PRIMARY KEY (`group_id`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `cmsi486_mflickner`.`user_group`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `cmsi486_mflickner`.`user_group` (
  `user_group_id` INT NOT NULL AUTO_INCREMENT,
  `is_group_admin` TINYINT(1) NULL,
  `user_id` INT NOT NULL,
  `group_id` INT NOT NULL,
  PRIMARY KEY (`user_group_id`),
  INDEX `fk_user_group_user_idx` (`user_id` ASC),
  INDEX `fk_user_group_group1_idx` (`group_id` ASC),
  CONSTRAINT `fk_user_group_user`
    FOREIGN KEY (`user_id`)
    REFERENCES `cmsi486_mflickner`.`user` (`user_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_user_group_group1`
    FOREIGN KEY (`group_id`)
    REFERENCES `cmsi486_mflickner`.`group` (`group_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `cmsi486_mflickner`.`task`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `cmsi486_mflickner`.`task` (
  `task_id` INT NOT NULL AUTO_INCREMENT,
  `task_name` VARCHAR(45) NULL,
  PRIMARY KEY (`task_id`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `cmsi486_mflickner`.`friendship`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `cmsi486_mflickner`.`friendship` (
  `friendship_id` INT NOT NULL,
  `user_id` INT NOT NULL,
  `user_id1` INT NOT NULL,
  PRIMARY KEY (`friendship_id`, `user_id`, `user_id1`),
  INDEX `fk_friendship_user1_idx` (`user_id` ASC),
  INDEX `fk_friendship_user2_idx` (`user_id1` ASC),
  CONSTRAINT `fk_friendship_user1`
    FOREIGN KEY (`user_id`)
    REFERENCES `cmsi486_mflickner`.`user` (`user_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_friendship_user2`
    FOREIGN KEY (`user_id1`)
    REFERENCES `cmsi486_mflickner`.`user` (`user_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `cmsi486_mflickner`.`user_group_task`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `cmsi486_mflickner`.`user_group_task` (
  `user_group_task_id` INT UNSIGNED NOT NULL,
  `is_desi` TINYINT(1) NULL,
  `points` INT UNSIGNED NULL,
  `user_group_id` INT NOT NULL,
  `task_id` INT NOT NULL,
  PRIMARY KEY (`user_group_task_id`),
  INDEX `fk_user_group_task_user_group1_idx` (`user_group_id` ASC),
  INDEX `fk_user_group_task_task1_idx` (`task_id` ASC),
  CONSTRAINT `fk_user_group_task_user_group1`
    FOREIGN KEY (`user_group_id`)
    REFERENCES `cmsi486_mflickner`.`user_group` (`user_group_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_user_group_task_task1`
    FOREIGN KEY (`task_id`)
    REFERENCES `cmsi486_mflickner`.`task` (`task_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `cmsi486_mflickner`.`user_group_task_action`
-- -----------------------------------------------------
```

```
CREATE TABLE IF NOT EXISTS `cmsi486_mflickner`.`
    user_group_task_action` (
  `task_action_id` INT NOT NULL,
  `user_group_task_id` INT UNSIGNED NOT NULL,
  PRIMARY KEY (`task_action_id`),
  INDEX `fk_task_action_user_group_task1_idx` (`user_group_task_id`
      ASC),
  CONSTRAINT `fk_task_action_user_group_task1`
    FOREIGN KEY (`user_group_task_id`)
    REFERENCES `cmsi486_mflickner`.`user_group_task` (`
        user_group_task_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `cmsi486_mflickner`.`completion`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `cmsi486_mflickner`.`completion` (
  `task_action_id` INT NOT NULL,
  `completed_date` DATETIME NULL,
  PRIMARY KEY (`task_action_id`),
  INDEX `fk_completion_task_action1_idx` (`task_action_id` ASC),
  CONSTRAINT `fk_completion_task_action1`
    FOREIGN KEY (`task_action_id`)
    REFERENCES `cmsi486_mflickner`.`user_group_task_action` (`
        task_action_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `cmsi486_mflickner`.`volunteer_completion`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `cmsi486_mflickner`.`volunteer_completion
    ` (
  `task_action_id` INT NOT NULL,
  `volunteer_completed` DATETIME NULL,
  PRIMARY KEY (`task_action_id`),
  INDEX `fk_volunteer_completion_task_action1_idx` (`task_action_id`
       ASC),
  CONSTRAINT `fk_volunteer_completion_task_action1`
    FOREIGN KEY (`task_action_id`)
    REFERENCES `cmsi486_mflickner`.`user_group_task_action` (`
        task_action_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
```

```
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `cmsi486_mflickner`.`opt_out`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `cmsi486_mflickner`.`opt_out` (
  `task_action_id` INT NOT NULL,
  `opt_out_date` DATETIME NULL,
  PRIMARY KEY (`task_action_id`),
  CONSTRAINT `fk_opt_out_task_action1`
    FOREIGN KEY (`task_action_id`)
    REFERENCES `cmsi486_mflickner`.`user_group_task_action` (`
        task_action_id`)
    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;

-- -----------------------------------------------------
-- Data for table `cmsi486_mflickner`.`user`
-- -----------------------------------------------------
START TRANSACTION;
USE `cmsi486_mflickner`;
INSERT INTO `cmsi486_mflickner`.`user` (`user_id`, `username`, `
    first_name`, `last_name`, `email`, `password`) VALUES (1, '
    mwflickner', 'Matthew', 'Flickenr', 'mwflickner@gmail.com', '
    swagflick');
INSERT INTO `cmsi486_mflickner`.`user` (`user_id`, `username`, `
    first_name`, `last_name`, `email`, `password`) VALUES (2, '
    goobypls', 'Gooby', 'Pls', 'goobypls@gmail.com', 'meowswag');
INSERT INTO `cmsi486_mflickner`.`user` (`user_id`, `username`, `
    first_name`, `last_name`, `email`, `password`) VALUES (3, '
    dolanpls', 'Dolan', 'Pls', 'dolanpls@gmail.com', 'meowswag');
INSERT INTO `cmsi486_mflickner`.`user` (`user_id`, `username`, `
    first_name`, `last_name`, `email`, `password`) VALUES (4, 'jsmith
    ', 'John', 'Smith', 'jsmith@gmail.com', 'jsmitty');

COMMIT;


-- -----------------------------------------------------
-- Data for table `cmsi486_mflickner`.`group`
-- -----------------------------------------------------
START TRANSACTION;
```

```
USE `cmsi486_mflickner`;
INSERT INTO `cmsi486_mflickner`.`group` (`group_id`, `group_name`)
    VALUES (1, 'The House');
INSERT INTO `cmsi486_mflickner`.`group` (`group_id`, `group_name`)
    VALUES (2, 'Pls');
INSERT INTO `cmsi486_mflickner`.`group` (`group_id`, `group_name`)
    VALUES (3, 'Matt n Friends');

COMMIT;


-- -----------------------------------------------------
-- Data for table `cmsi486_mflickner`.`user_group`
-- -----------------------------------------------------
START TRANSACTION;
USE `cmsi486_mflickner`;
INSERT INTO `cmsi486_mflickner`.`user_group` (`user_group_id`, `
    is_group_admin`, `user_id`, `group_id`) VALUES (1, true, 1, 1);
INSERT INTO `cmsi486_mflickner`.`user_group` (`user_group_id`, `
    is_group_admin`, `user_id`, `group_id`) VALUES (2, false, 2, 1);
INSERT INTO `cmsi486_mflickner`.`user_group` (`user_group_id`, `
    is_group_admin`, `user_id`, `group_id`) VALUES (3, false, 3, 1);
INSERT INTO `cmsi486_mflickner`.`user_group` (`user_group_id`, `
    is_group_admin`, `user_id`, `group_id`) VALUES (4, true, 2, 2);
INSERT INTO `cmsi486_mflickner`.`user_group` (`user_group_id`, `
    is_group_admin`, `user_id`, `group_id`) VALUES (5, false, 3, 2);
INSERT INTO `cmsi486_mflickner`.`user_group` (`user_group_id`, `
    is_group_admin`, `user_id`, `group_id`) VALUES (6, true, 1, 3);
INSERT INTO `cmsi486_mflickner`.`user_group` (`user_group_id`, `
    is_group_admin`, `user_id`, `group_id`) VALUES (7, false, 2, 3);

COMMIT;


-- -----------------------------------------------------
-- Data for table `cmsi486_mflickner`.`task`
-- -----------------------------------------------------
START TRANSACTION;
USE `cmsi486_mflickner`;
INSERT INTO `cmsi486_mflickner`.`task` (`task_id`, `task_name`)
    VALUES (1, 'Empty Dishwasher');
INSERT INTO `cmsi486_mflickner`.`task` (`task_id`, `task_name`)
    VALUES (2, 'Take out trash');
INSERT INTO `cmsi486_mflickner`.`task` (`task_id`, `task_name`)
    VALUES (3, 'Designated Driving');
INSERT INTO `cmsi486_mflickner`.`task` (`task_id`, `task_name`)
    VALUES (4, 'Clean Sink');
INSERT INTO `cmsi486_mflickner`.`task` (`task_id`, `task_name`)
    VALUES (5, 'Mow Lawn');
```

```
INSERT INTO `cmsi486_mflickner`.`task` (`task_id`, `task_name`)
   VALUES (6, 'Feed Dog');
INSERT INTO `cmsi486_mflickner`.`task` (`task_id`, `task_name`)
   VALUES (7, 'Vaccum');

COMMIT;


-- -----------------------------------------------------
-- Data for table `cmsi486_mflickner`.`friendship`
-- -----------------------------------------------------
START TRANSACTION;
USE `cmsi486_mflickner`;
INSERT INTO `cmsi486_mflickner`.`friendship` (`friendship_id`, `
   user_id`, `user_id1`) VALUES (1, 1, 2);
INSERT INTO `cmsi486_mflickner`.`friendship` (`friendship_id`, `
   user_id`, `user_id1`) VALUES (2, 1, 3);
INSERT INTO `cmsi486_mflickner`.`friendship` (`friendship_id`, `
   user_id`, `user_id1`) VALUES (3, 1, 4);
INSERT INTO `cmsi486_mflickner`.`friendship` (`friendship_id`, `
   user_id`, `user_id1`) VALUES (4, 2, 1);
INSERT INTO `cmsi486_mflickner`.`friendship` (`friendship_id`, `
   user_id`, `user_id1`) VALUES (5, 2, 3);
INSERT INTO `cmsi486_mflickner`.`friendship` (`friendship_id`, `
   user_id`, `user_id1`) VALUES (6, 2, 4);
INSERT INTO `cmsi486_mflickner`.`friendship` (`friendship_id`, `
   user_id`, `user_id1`) VALUES (7, 3, 1);
INSERT INTO `cmsi486_mflickner`.`friendship` (`friendship_id`, `
   user_id`, `user_id1`) VALUES (8, 3, 2);
INSERT INTO `cmsi486_mflickner`.`friendship` (`friendship_id`, `
   user_id`, `user_id1`) VALUES (9, 3, 4);
INSERT INTO `cmsi486_mflickner`.`friendship` (`friendship_id`, `
   user_id`, `user_id1`) VALUES (10, 4, 1);
INSERT INTO `cmsi486_mflickner`.`friendship` (`friendship_id`, `
   user_id`, `user_id1`) VALUES (11, 4, 2);
INSERT INTO `cmsi486_mflickner`.`friendship` (`friendship_id`, `
   user_id`, `user_id1`) VALUES (12, 4, 3);

COMMIT;


-- -----------------------------------------------------
-- Data for table `cmsi486_mflickner`.`user_group_task`
-- -----------------------------------------------------
START TRANSACTION;
USE `cmsi486_mflickner`;
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
   user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
   task_id`) VALUES (1, true, 7, 1, 1);
```

```
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (2, false, 4, 2, 1);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (3, false, 12, 3, 1);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (4, false, 0, 1, 2);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (5, true, 2, 2, 2);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (6, false, 4, 3, 2);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (7, false, 6, 1, 3);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (8, false, 7, 2, 3);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (9, true, 1, 3, 3);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (10, true, 9, 4, 6);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (11, false, 2, 5, 6);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (12, false, 0, 4, 7);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (13, true, 2, 5, 7);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (14, false, 4, 6, 4);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (15, true, 5, 7, 4);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (16, false, 2, 6, 5);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (17, true, 1, 7, 5);
```

```
COMMIT;


-- -----------------------------------------------------
-- Data for table `cmsi486_mflickner`.`user_group_task_action`
-- -----------------------------------------------------
START TRANSACTION;
USE `cmsi486_mflickner`;
INSERT INTO `cmsi486_mflickner`.`user_group_task_action` (`
    task_action_id`, `user_group_task_id`) VALUES (1, 1);
INSERT INTO `cmsi486_mflickner`.`user_group_task_action` (`
    task_action_id`, `user_group_task_id`) VALUES (2, 2);
INSERT INTO `cmsi486_mflickner`.`user_group_task_action` (`
    task_action_id`, `user_group_task_id`) VALUES (3, 3);
INSERT INTO `cmsi486_mflickner`.`user_group_task_action` (`
    task_action_id`, `user_group_task_id`) VALUES (4, 4);
INSERT INTO `cmsi486_mflickner`.`user_group_task_action` (`
    task_action_id`, `user_group_task_id`) VALUES (5, 5);
INSERT INTO `cmsi486_mflickner`.`user_group_task_action` (`
    task_action_id`, `user_group_task_id`) VALUES (6, 6);
INSERT INTO `cmsi486_mflickner`.`user_group_task_action` (`
    task_action_id`, `user_group_task_id`) VALUES (7, 7);
INSERT INTO `cmsi486_mflickner`.`user_group_task_action` (`
    task_action_id`, `user_group_task_id`) VALUES (8, 8);
INSERT INTO `cmsi486_mflickner`.`user_group_task_action` (`
    task_action_id`, `user_group_task_id`) VALUES (9, 9);

COMMIT;


-- -----------------------------------------------------
-- Data for table `cmsi486_mflickner`.`completion`
-- -----------------------------------------------------
START TRANSACTION;
USE `cmsi486_mflickner`;
INSERT INTO `cmsi486_mflickner`.`completion` (`task_action_id`, `
    completed_date`) VALUES (1, '2015-12-10 1:00:00-08:00');
INSERT INTO `cmsi486_mflickner`.`completion` (`task_action_id`, `
    completed_date`) VALUES (3, '2015-12-05 13:57:00-08:00');
INSERT INTO `cmsi486_mflickner`.`completion` (`task_action_id`, `
    completed_date`) VALUES (5, '2015-12-09 22:34:00-08:00');
INSERT INTO `cmsi486_mflickner`.`completion` (`task_action_id`, `
    completed_date`) VALUES (7, '2015-12-08 11:45:00-08:00');
INSERT INTO `cmsi486_mflickner`.`completion` (`task_action_id`, `
    completed_date`) VALUES (9, '2015-12-06 9:20:00-08:00');

COMMIT;
```

```
-- ------------------------------------------------------------
-- Data for table `cmsi486_mflickner`.`volunteer_completion`
-- ------------------------------------------------------------
START TRANSACTION;
USE `cmsi486_mflickner`;
INSERT INTO `cmsi486_mflickner`.`volunteer_completion` (`
    task_action_id`, `volunteer_completed`) VALUES (2, '2015-12-06␣
    12:00:00-08:00');
INSERT INTO `cmsi486_mflickner`.`volunteer_completion` (`
    task_action_id`, `volunteer_completed`) VALUES (4, '2015-12-07␣
    18:30:00-08:00');
INSERT INTO `cmsi486_mflickner`.`volunteer_completion` (`
    task_action_id`, `volunteer_completed`) VALUES (6, '2015-12-05␣
    8:00:00-08:00');

COMMIT;


-- ------------------------------------------------------------
-- Data for table `cmsi486_mflickner`.`opt_out`
-- ------------------------------------------------------------
START TRANSACTION;
USE `cmsi486_mflickner`;
INSERT INTO `cmsi486_mflickner`.`opt_out` (`task_action_id`, `
    opt_out_date`) VALUES (8, '2015-12-05␣14:00:00-08:00');

COMMIT;
```

4.2 English version of 10+ database queries, and the SQL DML for each query

4.2.1 English version

1. How many users are in the group with group.id = 2?
2. Who is the desi for task Ëmpty Dishwasherïn the group with group.id = 1?
3. How many groups is user with username m̈wflicknerïn?
4. How many points does user m̈wflicknerḧave in task "Take Out Trash" in group with group.id = 1?
5. When was the last time the task "Empty Dishwasher" in group with group.id = 1 completed?
6. What is the first name and the last name of the user with user.id = 2?
7. How many points does user with user.user_id = 1 have among all of his tasks in group with group.id = 3?
8. How many points does user with user.user_id = 1 have among all of his tasks in all of his groups?

9. How many points are there total in this application?
10. Who are the users who have performed an opt out?

4.2.2 SQL DML

```
#1 How many users are in the group with group.id = 2?
SELECT COUNT(user_id)
    FROM `user_group`
        WHERE group_id = 2;

#2 Who is the desi for task "Empty_Dishwasher" in
#  the group with group.id = 1?

SELECT username
  FROM `user_group`
  NATURAL JOIN `user_group_task`
  NATURAL JOIN `task`
  NATURAL JOIN `user`
  WHERE group_id = 1
    AND task_name = "Empty_Dishwasher"
    AND is_desi = 1;

#3 How many groups is user with username "mwflickner" in?
SELECT COUNT(user_id)
  FROM `user_group` NATURAL JOIN `user`
    WHERE username = "mwflickner";

#4 How many points does user "mwflickner" have in task
#  "Take_Out_Trash" in group with group.id = 1?
SELECT points from `user_group_task`
  NATURAL JOIN `user_group`
  NATURAL JOIN `task`
  NATURAL JOIN `user`
  WHERE task_name = "Take_out_trash"
  AND group_id = 1
  AND username = "mwflickner";


#5 When has the task "Empty_Dishwasher"
#  in group with group.id = 1 been completed?
SELECT completed_date
  FROM `completion` NATURAL JOIN `user_group_task_action`
  NATURAL JOIN `user_group_task`
  NATURAL JOIN `task`
  NATURAL JOIN `user_group`
  WHERE group_id = 1
  AND task_name = "Empty_Dishwasher";
```

```
#6 What is the first name and the last name
#  of the user with user.id = 2?
SELECT user.first_name , user.last_name
  FROM `user`
    WHERE user.user_id = 2;


#7 How many points does user with user.user_id = 1
#  have among all of his tasks in group with
#  group.id = 3?
SELECT SUM(points) FROM `user_group_task`
  NATURAL JOIN `user_group`
  WHERE user_id = 1 AND group_id = 3;


#8 How many points does user with user.user_id = 1
#  have among all of his tasks in all of his groups?

SELECT SUM(points) FROM `user_group_task`
  NATURAL JOIN `user_group`
  WHERE user_id = 1;

#9 How many points are there
#  total in this application?
SELECT SUM(user_group_task.points)
  FROM `user_group_task`;

#10 Who are the users who have performed an opt out?
SELECT user_id , username FROM `user_group` NATURAL JOIN `user`
WHERE user_group_id = (SELECT
   user_group_id FROM `user_group_task`
  WHERE user_group_task_id = (SELECT
       user_group_task_id FROM `user_group_task_action`
      WHERE task_action_id = (SELECT
         task_action_id FROM `opt_out`
       )
     )
);
```

4.3 Review sign-off sheet

See attached review.

4.4 Design Limitations

One of the design limitations I ran into with Desi was being unable to track and log past actions for tasks. I revised my model and added the Task_Action along with Task_Completion, Volunteer_Completion, Opt_Out to track task actions and the types of actions. A limitation that I currently have is that points from a task are restricted to only those tasks and cannot be applied to other tasks.

Chapter 5

Database Integrity and Security

5.1 Functional Dependencies

5.1.1 Table: User

user_id → username, first_name, last_name, email_address, password
username → user_id, first_name, last_name, email_address, password
email_address → user_id, first_name, last_name, email_address, password

5.1.2 Table: Group

group_id → group_name

5.1.3 Table: UserGroup

user_group_id → is_group_admin, user_id (FK), group_id (FK)
user_id (FK), group_id (FK) → user_group_id, is_group_admin

5.1.4 Table: UserGroupTask

user_group_task_id → points, is_desi, user_group_id (FK), task_id (FK)
user_group_id (FK), task_id (FK) → user_group_task_id, points, is_desi

5.1.5 Table: Task

task_id → task_name

5.1.6 Table: Completion

user_group_task_action_id → completed_date

5.1.7 Table: VolunteerCompletion

user_group_task_action_id → volunteer_completed

5.1.8 Table: Opt-Out

user_group_task_action_id → opt_out_date

5.2 Adjustments for Normalization

My database design did not require any normalization adjustments as it is already normalized.

5.3 Integrity and Security

Users have the ability to write to other tables but only delete Friendships or themselves. GroupAdmins can delete anything group or task related. All emails, usernames, passwords need to be validated by a regex to make sure they fit the proper requirements.

Chapter 6

Implementation

6.1 Indices

Indices will be placed on the primary keys of each table.

6.2 Data

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

-- -----------------------------------------------------
-- Data for table `cmsi486_mflickner`.`user`
-- -----------------------------------------------------
START TRANSACTION;
USE `cmsi486_mflickner`;
INSERT INTO `cmsi486_mflickner`.`user` (`user_id`, `username`, `
    first_name`, `last_name`, `email`, `password`) VALUES (1, '
    mwflickner', 'Matthew', 'Flickenr', 'mwflickner@gmail.com', '
    swagflick');
INSERT INTO `cmsi486_mflickner`.`user` (`user_id`, `username`, `
    first_name`, `last_name`, `email`, `password`) VALUES (2, '
    goobypls', 'Gooby', 'Pls', 'goobypls@gmail.com', 'meowswag');
INSERT INTO `cmsi486_mflickner`.`user` (`user_id`, `username`, `
    first_name`, `last_name`, `email`, `password`) VALUES (3, '
    dolanpls', 'Dolan', 'Pls', 'dolanpls@gmail.com', 'meowswag');
INSERT INTO `cmsi486_mflickner`.`user` (`user_id`, `username`, `
    first_name`, `last_name`, `email`, `password`) VALUES (4, 'jsmith
    ', 'John', 'Smith', 'jsmith@gmail.com', 'jsmitty');

COMMIT;


-- -----------------------------------------------------
-- Data for table `cmsi486_mflickner`.`group`
-- -----------------------------------------------------
START TRANSACTION;
USE `cmsi486_mflickner`;
INSERT INTO `cmsi486_mflickner`.`group` (`group_id`, `group_name`)
    VALUES (1, 'The House');
INSERT INTO `cmsi486_mflickner`.`group` (`group_id`, `group_name`)
    VALUES (2, 'Pls');
INSERT INTO `cmsi486_mflickner`.`group` (`group_id`, `group_name`)
    VALUES (3, 'Matt n Friends');

COMMIT;


-- -----------------------------------------------------
-- Data for table `cmsi486_mflickner`.`user_group`
-- -----------------------------------------------------
START TRANSACTION;
USE `cmsi486_mflickner`;
INSERT INTO `cmsi486_mflickner`.`user_group` (`user_group_id`, `
    is_group_admin`, `user_id`, `group_id`) VALUES (1, true, 1, 1);
INSERT INTO `cmsi486_mflickner`.`user_group` (`user_group_id`, `
    is_group_admin`, `user_id`, `group_id`) VALUES (2, false, 2, 1);
INSERT INTO `cmsi486_mflickner`.`user_group` (`user_group_id`, `
    is_group_admin`, `user_id`, `group_id`) VALUES (3, false, 3, 1);
INSERT INTO `cmsi486_mflickner`.`user_group` (`user_group_id`, `
    is_group_admin`, `user_id`, `group_id`) VALUES (4, true, 2, 2);
INSERT INTO `cmsi486_mflickner`.`user_group` (`user_group_id`, `
    is_group_admin`, `user_id`, `group_id`) VALUES (5, false, 3, 2);
INSERT INTO `cmsi486_mflickner`.`user_group` (`user_group_id`, `
    is_group_admin`, `user_id`, `group_id`) VALUES (6, true, 1, 3);
INSERT INTO `cmsi486_mflickner`.`user_group` (`user_group_id`, `
    is_group_admin`, `user_id`, `group_id`) VALUES (7, false, 2, 3);

COMMIT;


-- -----------------------------------------------------
-- Data for table `cmsi486_mflickner`.`task`
-- -----------------------------------------------------
START TRANSACTION;
USE `cmsi486_mflickner`;
INSERT INTO `cmsi486_mflickner`.`task` (`task_id`, `task_name`)
    VALUES (1, 'Empty Dishwasher');
INSERT INTO `cmsi486_mflickner`.`task` (`task_id`, `task_name`)
    VALUES (2, 'Take out trash');
INSERT INTO `cmsi486_mflickner`.`task` (`task_id`, `task_name`)
    VALUES (3, 'Designated Driving');
INSERT INTO `cmsi486_mflickner`.`task` (`task_id`, `task_name`)
    VALUES (4, 'Clean Sink');
```

```
INSERT INTO `cmsi486_mflickner`.`task` (`task_id`, `task_name`)
    VALUES (5, 'Mow Lawn');
INSERT INTO `cmsi486_mflickner`.`task` (`task_id`, `task_name`)
    VALUES (6, 'Feed Dog');
INSERT INTO `cmsi486_mflickner`.`task` (`task_id`, `task_name`)
    VALUES (7, 'Vaccum');

COMMIT;


-- -----------------------------------------------------
-- Data for table `cmsi486_mflickner`.`friendship`
-- -----------------------------------------------------
START TRANSACTION;
USE `cmsi486_mflickner`;
INSERT INTO `cmsi486_mflickner`.`friendship` (`friendship_id`, `
    user_id`, `user_id1`) VALUES (1, 1, 2);
INSERT INTO `cmsi486_mflickner`.`friendship` (`friendship_id`, `
    user_id`, `user_id1`) VALUES (2, 1, 3);
INSERT INTO `cmsi486_mflickner`.`friendship` (`friendship_id`, `
    user_id`, `user_id1`) VALUES (3, 1, 4);
INSERT INTO `cmsi486_mflickner`.`friendship` (`friendship_id`, `
    user_id`, `user_id1`) VALUES (4, 2, 1);
INSERT INTO `cmsi486_mflickner`.`friendship` (`friendship_id`, `
    user_id`, `user_id1`) VALUES (5, 2, 3);
INSERT INTO `cmsi486_mflickner`.`friendship` (`friendship_id`, `
    user_id`, `user_id1`) VALUES (6, 2, 4);
INSERT INTO `cmsi486_mflickner`.`friendship` (`friendship_id`, `
    user_id`, `user_id1`) VALUES (7, 3, 1);
INSERT INTO `cmsi486_mflickner`.`friendship` (`friendship_id`, `
    user_id`, `user_id1`) VALUES (8, 3, 2);
INSERT INTO `cmsi486_mflickner`.`friendship` (`friendship_id`, `
    user_id`, `user_id1`) VALUES (9, 3, 4);
INSERT INTO `cmsi486_mflickner`.`friendship` (`friendship_id`, `
    user_id`, `user_id1`) VALUES (10, 4, 1);
INSERT INTO `cmsi486_mflickner`.`friendship` (`friendship_id`, `
    user_id`, `user_id1`) VALUES (11, 4, 2);
INSERT INTO `cmsi486_mflickner`.`friendship` (`friendship_id`, `
    user_id`, `user_id1`) VALUES (12, 4, 3);

COMMIT;


-- -----------------------------------------------------
-- Data for table `cmsi486_mflickner`.`user_group_task`
-- -----------------------------------------------------
START TRANSACTION;
USE `cmsi486_mflickner`;
```

```
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (1, true, 7, 1, 1);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (2, false, 4, 2, 1);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (3, false, 12, 3, 1);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (4, false, 0, 1, 2);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (5, true, 2, 2, 2);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (6, false, 4, 3, 2);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (7, false, 6, 1, 3);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (8, false, 7, 2, 3);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (9, true, 1, 3, 3);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (10, true, 9, 4, 6);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (11, false, 2, 5, 6);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (12, false, 0, 4, 7);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (13, true, 2, 5, 7);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (14, false, 4, 6, 4);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (15, true, 5, 7, 4);
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (16, false, 2, 6, 5);
```

```
INSERT INTO `cmsi486_mflickner`.`user_group_task` (`
    user_group_task_id`, `is_desi`, `points`, `user_group_id`, `
    task_id`) VALUES (17, true, 1, 7, 5);

COMMIT;


-- -----------------------------------------------------
-- Data for table `cmsi486_mflickner`.`user_group_task_action`
-- -----------------------------------------------------
START TRANSACTION;
USE `cmsi486_mflickner`;
INSERT INTO `cmsi486_mflickner`.`user_group_task_action` (`
    task_action_id`, `user_group_task_id`) VALUES (1, 1);
INSERT INTO `cmsi486_mflickner`.`user_group_task_action` (`
    task_action_id`, `user_group_task_id`) VALUES (2, 2);
INSERT INTO `cmsi486_mflickner`.`user_group_task_action` (`
    task_action_id`, `user_group_task_id`) VALUES (3, 3);
INSERT INTO `cmsi486_mflickner`.`user_group_task_action` (`
    task_action_id`, `user_group_task_id`) VALUES (4, 4);
INSERT INTO `cmsi486_mflickner`.`user_group_task_action` (`
    task_action_id`, `user_group_task_id`) VALUES (5, 5);
INSERT INTO `cmsi486_mflickner`.`user_group_task_action` (`
    task_action_id`, `user_group_task_id`) VALUES (6, 6);
INSERT INTO `cmsi486_mflickner`.`user_group_task_action` (`
    task_action_id`, `user_group_task_id`) VALUES (7, 7);
INSERT INTO `cmsi486_mflickner`.`user_group_task_action` (`
    task_action_id`, `user_group_task_id`) VALUES (8, 8);
INSERT INTO `cmsi486_mflickner`.`user_group_task_action` (`
    task_action_id`, `user_group_task_id`) VALUES (9, 9);

COMMIT;


-- -----------------------------------------------------
-- Data for table `cmsi486_mflickner`.`completion`
-- -----------------------------------------------------
START TRANSACTION;
USE `cmsi486_mflickner`;
INSERT INTO `cmsi486_mflickner`.`completion` (`task_action_id`, `
    completed_date`) VALUES (1, '2015-12-10 1:00:00-08:00');
INSERT INTO `cmsi486_mflickner`.`completion` (`task_action_id`, `
    completed_date`) VALUES (3, '2015-12-05 13:57:00-08:00');
INSERT INTO `cmsi486_mflickner`.`completion` (`task_action_id`, `
    completed_date`) VALUES (5, '2015-12-09 22:34:00-08:00');
INSERT INTO `cmsi486_mflickner`.`completion` (`task_action_id`, `
    completed_date`) VALUES (7, '2015-12-08 11:45:00-08:00');
INSERT INTO `cmsi486_mflickner`.`completion` (`task_action_id`, `
    completed_date`) VALUES (9, '2015-12-06 9:20:00-08:00');
```

```
COMMIT;


-- -----------------------------------------------------
-- Data for table `cmsi486_mflickner`.`volunteer_completion`
-- -----------------------------------------------------
START TRANSACTION;
USE `cmsi486_mflickner`;
INSERT INTO `cmsi486_mflickner`.`volunteer_completion` (`
    task_action_id`, `volunteer_completed`) VALUES (2, '2015-12-06␣
    12:00:00-08:00');
INSERT INTO `cmsi486_mflickner`.`volunteer_completion` (`
    task_action_id`, `volunteer_completed`) VALUES (4, '2015-12-07␣
    18:30:00-08:00');
INSERT INTO `cmsi486_mflickner`.`volunteer_completion` (`
    task_action_id`, `volunteer_completed`) VALUES (6, '2015-12-05␣
    8:00:00-08:00');

COMMIT;


-- -----------------------------------------------------
-- Data for table `cmsi486_mflickner`.`opt_out`
-- -----------------------------------------------------
START TRANSACTION;
USE `cmsi486_mflickner`;
INSERT INTO `cmsi486_mflickner`.`opt_out` (`task_action_id`, `
    opt_out_date`) VALUES (8, '2015-12-05␣14:00:00-08:00');

COMMIT;
```

6.3 Query Trace

Unfortunately, I could not get the query trace working. However I can verify that all of the queries ran and returned the correct results.

6.4 Implementation Assessment

All in all, I thought I implemented my design very well. At first there were a lot of kinks I need to work out, but as the semester progressed I felt very confident about all of the improvements I made. The biggest key to the whole project was honestly the ERD. Once I had that, everything else was easy. Every time I had to change the ERD, I had to change pretty much everything. MySQL Workbench was incredibly helpful with the ERD and being able to generate the SQL create file from my ERD was incredibly useful.

Chapter 7

Lessons Learned

I learned a great deal of things during this project. I obviously made some mistakes along the way but I definitely learned a lot of things and if I had to do this project again, I would go about it a lot differently. First of all, if I were to do this project again, the first thing I would do would be the ERD. The ERD was the core of the entire project and doing other things differently before that was pointless because if I changed one little thing about the ERD I would have to go back and redo all of the sections I previously did. It turned into a big waste of time doing that and I didn't get much out of it. Rather, next time I would do the ERD first and get it right. Once I got the ERD right I would proceed to the other steps.

The next thing I learned was that LATEXis the way to go. I initially started doing this project in Microsoft Word and let me tell you, it was terrible. Using LATEXcleaned up my project so much and made it much easier to do all of the write-ups. I was also much more organized when I used it. The LATEXpackages were also super useful when it came to building tables and getting them to look the way I wanted them too. I highly recommend using an online LATEXtable generator as well because it made life so much easier.

Next up was ER Studio and MySQL Workbench. Honestly, if i had to do the project again, I would scrap ER Studio and just use mySQL Workbench. The software seemed old and outdated and the user interface was very confusing. MySQL Workbench had a cleaner user interface and had better documentation online as far as how to user the software. It was very easy to generate both a create script in SQL and then a populate script. I would highly recommend MySQL Workbench.

I am not exactly sure how many hours I spent on the project but if I had guess I would say it would be around 45 total hours. Dividing that up it comes to around 3.5 hours a week. But a lot of that was clustered around the due dates of the deliverables. And there wasn't always a deliverable every week. Honestly, a lot of the time spend was going back and redoing work that I had already done, which as I mentioned earlier could have been avoided if the ERD had been completed first.

The project definitely taught me how to write and read SQL a lot better. It also taught me when I should actually write SQL and when I could not. For example I should not write the SQL create script for my database, rather I should let MySQL Workbench generate it from my ERD and then from there just write my queries. The project also helped me really understand functional dependency and normalization a lot better. Using MySQL was definitely the best call over other options. PHPmyAdmin was pretty cool as well, I liked having such a visual way of interacting with my database.