**SQL Query Challenge Site**

Chris Lansing

Tim Elam

Roland Heintze

Kevin McBrayer

#shaft @irc.freenode.net

**Table of Contents**

[Introduction](#h.33unusex2u1g)

[Team Roles](#h.gm89j0w0mfv2)

[Chris Lansing](#h.5sd53tryl7dn)

[Tim Elam](#h.arc6yuv2eo4w)

[Roland Heintze](#h.6vf45zakmax)

[Kevin McBrayer](#h.h8wko1sgwbyu)

[GUI Mockups](#h.eduxbp1ovp9z)

[Use Cases](#h.3hpk9g7i8001)

[SQL](#h.m0a6439m8oy3)

[Questions](#h.423khyiwob4c)

[Queries](#h.h399uuvizxix)

[Database Creation Script](#h.s2fb44qx2qff)

[Logical Model](#h.iqz6z1fma53v)

[Initial ER Diagram](#h.e6tn2l6yiaz3)

[Final ER Diagram](#h.lc0q43kyjos3)

[Physical Model](#h.ocgrt9vqnj5u)

[Physical Model](#h.2r7q8to14xm3)

[Data types and sizes](#h.l6hn0mgfskdp)

[Application Planning](#h.d7xt40m43rda)

[Transactions/Data Entry](#h.mzff8n9yf38g)

[Environments, DBA aspects, Backups, Security](#h.8updxupgnvje)

[Graphical User Interface](#h.wkbfcn4rgc40)

[Post Mortem](#h.he9ijt6jzxm)

[Chris Lansing](#h.lzn1xnv4hu5f)

[Tim Elam](#h.q37tl5qzia51)

[Roland Heintze](#h.evx5r83bzovx)

[Kevin McBrayer](#h.5oca8ytiozv)

# Introduction

The purpose of this site is to provide instructors with the means for evaluating student competency in SQL database querying. The site will function by running a student SQL query and comparing its answer with the proper answer. The site will eventually contain two means of evaluation. The first will start a timer when the player opens a challenge and then stop on a correct submission. The user will then be given a score based on the elapsed time and amount of attempts made. The other evaluation will be base on a quiz system where an administrator would grade the students’ work and then provide them with feedback. Users will possess a score for each completed challenge, a total score for each type of evaluation, and an overall score. The leaderboard will be able to show the scoreboard for each type of score to include scores for individual challenges.

# Team Roles

## Chris Lansing

Database testing and analysis.

## Tim Elam

Application back end and functionality testing.

## Roland Heintze

Server setup/maintenance and documentation.

## Kevin McBrayer

Application front end and documentation.

**Scope**

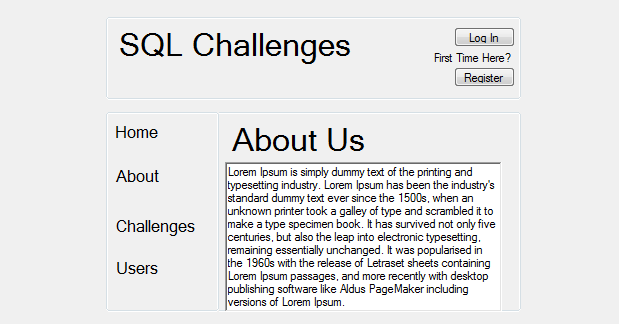
The premise of Challenge site is to create a safe and secure web application for user driven functional problems. The database keeps track of users, leagues, league members, challenge problems, valid entries to problems, challenge answers. There are plans for a leaderboard page that will contain statistics for users, leagues, and challenges.

**Requirements**

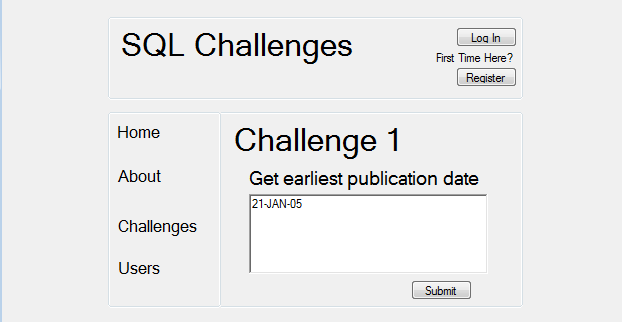
Users register and create leagues. Leagues are groups of users that will be able to respond to each others challenges. Users post challenges for other league members to answer and post valid entries to compare other user entries against. Other users post entries and are graded on time and correctness of their answer.

# GUI Mockups

About:



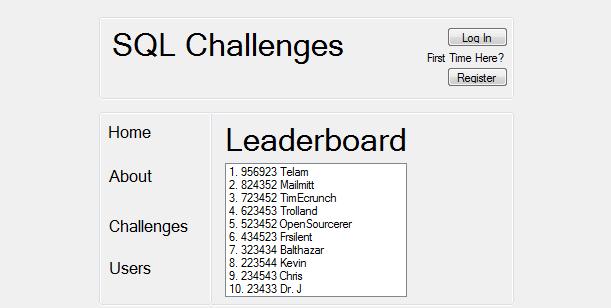
Chalenge Detail:



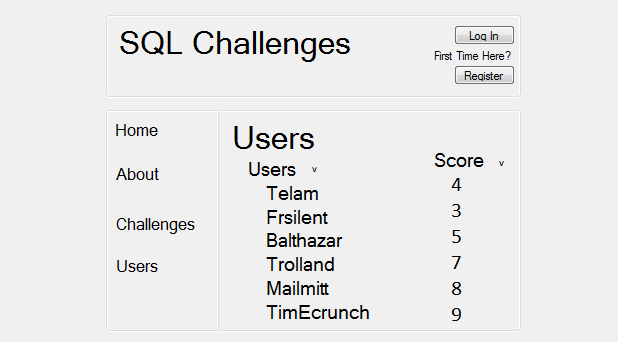
Challenges:



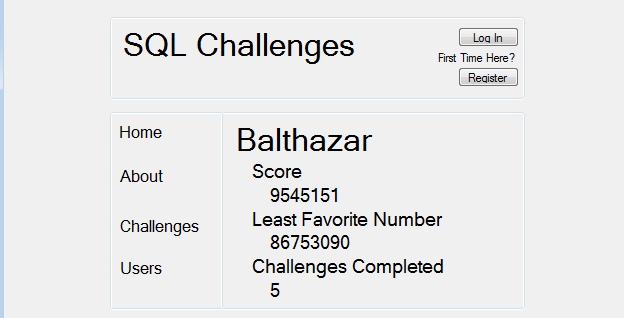
Leaderboard:



Users:



User Detail:



# Use Cases

|  |  |
| --- | --- |
| draft 1.0: | UC-1 |
| Title (Goal): | Register |
| Description: | A user registers an account on the site. |
| Primary Actor: | User |
| Main: | 1. User clicks the register button on the home page.  2. User enters desired username.  3. Username is checked in the database for availability.  4. User enters password.  5. User enters email address.  6. Email address is checked in the database for availability.  7. Email address is checked for validity.  8. User is sent a confirmation email to the registered address. |
| Alternate: | 3a. Username is unavailable.  -- 3a1. User is prompted to enter a new username and step 3 is repeated.  6a. Email address is unavailable.  -- 6a1. User is prompted to enter a new email address and step 6 is repeated.  7a. Email address is invalid.  -- 7a1. User is prompted to enter a new email address and step 7 is repeated. |

|  |  |
| --- | --- |
| draft 1.0: | UC-1 |
| Title (Goal): | Login |
| Description: | A user logs into a registered account. |
| Primary Actor: | User |
| Main: | 1. User clicks “login” button on homepage.  2. User is taken to the login page.  3. User enters username.  4. User enters password.  5. User clicks “login” again.  6. Username and password are checked against the database.  7. User is logged in. |
| Alternate: | 6a. Username or password is incorrect.  -- 6a1. User is prompted that username or password is incorrect and sent back to step 3. |

|  |  |
| --- | --- |
| draft 1.0: | UC-1 |
| Title (Goal): | Attempt Challenge |
| Description: | User attempts to complete one of the current challenges. |
| Primary Actor: | User |
| Main: | 1. User clicks on the Challenges submenu.  2. User is taken to the Challenges page.  3. User selects a challenge from the list of available challenges.  4. Challenge timer starts.  5. User enters answer into the textbox.  6. User clicks the submit button.  7. Timer is stopped.  8. Score is calculated.  9. Score is recorded in the database. |
| Alternate: | 6a. Users’ answer is incorrect.  6a1. User is asked to retry and is sent back to step 5. |

|  |  |
| --- | --- |
| draft 1.0: | UC-1 |
| Title (Goal): | Check Score |
| Description: | User checks a user's current score. |
| Primary Actor: | User |
| Main: | 1. User clicks the Users submenu from the home page.  2. User is taken to the Users page.  3. User clicks on a username from a list of usernames.  4. User is taken to the UserDetail page.  5. Score for the selected username is displayed on the page. |
| Alternate: | 5a. Selected username has no completed challenges.  5a1. User is told that selected username has no scores. |

|  |  |
| --- | --- |
| draft 1.0: | UC-1 |
| Title (Goal): | View Leaderboard |
| Description: | User views the Leaderboard page. |
| Primary Actor: | User |
| Main: | 1. User selects Leaderboard from the submenu.  2. User is taken to the Leaderboard page.  3. Top 10 usernames and their scores are displayed from the database, starting with the user with the highest score. |
| Alternate: |  |

# SQL

## Questions

1. How many active leagues are there?

2. How many active members are there in a particular league?

3. How many active members are there in all leagues?

4. How many active members are there on average through all leagues?

5. How many active members are there in the top 10 leagues?

6. What are the top 10 leagues based on user score?

7. Who are the top 10 users in a particular league?

8. Who are the top 10 users over all? (based on score)

9. How many challenges has a particular user posted?

10. How many challenges has a league posted?

11. How many challenges have all leagues posted?

12. How many challenges have been answered correctly?

13. How many challenges have not been answered correctly?

14. How many challenges have been answered correctly in a particular league?

15. How many challenges have not been answered correctly in a particular league?

16. How many challenges have been answered correctly for a particular user?

17. How many challenges have not been answered correctly for a particular user?

18. How many correct answers does a challenge have?

19. How many incorrect answers does a challenge have?

20. What was the top score for a challenge?

21. What was the lowest score for a challenge?

22. What was the average score for a challenge?

23. What was the fastest time for a challenge?

24. What was the slowest time for a challenge?

25. What was the average time for a challenge?

26. Who has the fastest challenge response average?

27. Who has the slowest challenge response average?

28. What league has the fastest average challenge response?

29. What league has the slowest average challenge response?

30. What is the average challenge response times for a particular league?

31. What league has the highest total user score?

32. What league has the lowest total user score?

33. What challenge has the fastest response time?

34. What challenge has the slowest response time?

35. What challenge has the fastest average response time?

36. What challenge has the slowest average response time?

## Queries

Chris Lansing

1. select count(name) from challenge\_site\_league;

2. select count(userprofile\_id) from challenge\_site\_league\_members where league\_id=1;

3. select count(userprofile\_id) from challenge\_site\_league\_members;

4. select avg(count(userprofile\_id)) from challenge\_site\_league\_members;

5. select league\_id, userprofile\_id from challenge\_site\_league\_members order by id where active=true and rownum <= 10;

6. select league\_id from challenge\_site\_league\_members order by score where rownum <= 10;

7. select userprofile\_id from challenge\_site\_league\_members order by score where rownum <= 10 and league\_id=1;

8. select userprofile\_id from challenge\_site\_league\_members order by score where rownum <= 10;

9. select count(name) from challenge\_site\_challenge where owner\_id=1;

Kevin McBrayer

10. select count(id) from challenge where league\_id=<any league id>

11. select count(id) from challenge

12. select count(distinct challenge) from challenge\_entry where correct=true

13. select count(distinct challenge) from challenge\_entry where correct=false

14. select count(distinct challenge) from challenge\_entry where correct=true and challenge in (select id from challenge where league=<any\_league\_id>)

15. select count(distinct challenge) from challenge\_entry where correct=false and challenge in (select id from challenge where league=<any\_league\_id>)

16. select count(id) from challenge\_entry where correct=true and owner=<any\_owner\_name>

17. select count(id) from challenge\_entry where correct=false and owner=<any\_owner\_name>

18. select count(id) from challenge\_entry where correct=true and challenge=<any\_challenge>

19. select count(id) from challenge\_entry where correct=false and challenge=<any\_challenge>

Roland Heintze

20. select max(score) from challenge\_entry;

21. select min(score) from challenge\_entry;

22. select avg(score) from challenge\_entry;

23. select min(time\_completed-time\_started) from challenge\_entry where challenge\_id=1;

24. select max(time\_completed-time\_started) from challenge\_entry where challenge\_id=1;

25. select avg(time\_completed-time\_started) from challenge\_entry where challenge\_id=1;

26. select min(sum(avg(time\_completed-time\_started))),owner\_id from (select \* from challenge\_entry where correct=True);

27. select max(avg(time\_completed-time\_started)),owner\_id from (select \* from challenge\_entry where correct=True);

28. select id, min(select sum(avg(time\_completed)time\_started)) from challenge\_entry) from league;

29. select id, max(select sum(avg(time\_completed)time\_started)) from challenge\_entry) from league;

30. select avg(time\_completed-time\_started) from challenge\_entry where league=1;

Tim Elam

30. What is the average challenge response times for a particular league?

select avg(time\_completed-time\_started), league\_id from challenge\_entry where league\_id=my\_league

31. What league has the highest total user score?

select

32. What league has the lowest total user score?

33. select min(time\_completed-time\_started), challenge\_id from challenge\_entry

34. select max(time\_completed-time\_started), challenge\_id from challenge\_entry

35. select min(time\_completed-time\_started), challenge\_id from challenge\_entry group by challenge\_id

36. max(time\_completed-time\_started), challenge\_id from challenge\_entry group by challenge\_id

37. select count(friends\_1\_id) from friends where friend\_1\_id=my\_id

38. select max(count(friends\_1\_id)) from friends where friend\_1\_id=my\_id

39. select min(count(friends\_1\_id)) from friends where friend\_1\_id=my\_id

40. select count(friends\_1\_id), count(friends\_2\_id), from friends where count(friends\_1\_id) > count(friends\_2\_id)

## Database Creation Script

BEGIN;

CREATE TABLE `challenge\_site\_userprofile\_friends` (

`id` integer AUTO\_INCREMENT NOT NULL PRIMARY KEY,

`from\_userprofile\_id` integer NOT NULL,

`to\_userprofile\_id` integer NOT NULL,

UNIQUE (`from\_userprofile\_id`, `to\_userprofile\_id`)

)

;

CREATE TABLE `challenge\_site\_userprofile` (

`id` integer AUTO\_INCREMENT NOT NULL PRIMARY KEY,

`dob` date,

`user\_id` integer NOT NULL UNIQUE

)

;

ALTER TABLE `challenge\_site\_userprofile` ADD CONSTRAINT `user\_id\_refs\_id\_68c33420` FOREIGN KEY (`user\_id`) REFERENCES `auth\_user` (`id`);

ALTER TABLE `challenge\_site\_userprofile\_friends` ADD CONSTRAINT `from\_userprofile\_id\_refs\_id\_3f3ff033` FOREIGN KEY (`from\_userprofile\_id`) REFERENCES `challenge\_site\_userprofile` (`id`);

ALTER TABLE `challenge\_site\_userprofile\_friends` ADD CONSTRAINT `to\_userprofile\_id\_refs\_id\_3f3ff033` FOREIGN KEY (`to\_userprofile\_id`) REFERENCES `challenge\_site\_userprofile` (`id`);

CREATE TABLE `challenge\_site\_league\_members` (

`id` integer AUTO\_INCREMENT NOT NULL PRIMARY KEY,

`league\_id` integer NOT NULL,

`userprofile\_id` integer NOT NULL,

UNIQUE (`league\_id`, `userprofile\_id`)

)

;

ALTER TABLE `challenge\_site\_league\_members` ADD CONSTRAINT `userprofile\_id\_refs\_id\_127824fb` FOREIGN KEY (`userprofile\_id`) REFERENCES `challenge\_site\_userprofile` (`id`);

CREATE TABLE `challenge\_site\_league` (

`id` integer AUTO\_INCREMENT NOT NULL PRIMARY KEY,

`name` varchar(200) NOT NULL,

`description` varchar(200) NOT NULL,

`owner\_id` integer NOT NULL

)

;

ALTER TABLE `challenge\_site\_league` ADD CONSTRAINT `owner\_id\_refs\_id\_66c5e44f` FOREIGN KEY (`owner\_id`) REFERENCES `challenge\_site\_userprofile` (`id`);

ALTER TABLE `challenge\_site\_league\_members` ADD CONSTRAINT `league\_id\_refs\_id\_76475f7f` FOREIGN KEY (`league\_id`) REFERENCES `challenge\_site\_league` (`id`);

CREATE TABLE `challenge\_site\_challenge` (

`id` integer AUTO\_INCREMENT NOT NULL PRIMARY KEY,

`name` varchar(200) NOT NULL,

`description` varchar(200) NOT NULL,

`score` integer NOT NULL,

`starting\_time` datetime NOT NULL,

`ending\_time` datetime NOT NULL,

`league\_id` integer NOT NULL,

`owner\_id` integer NOT NULL

)

;

ALTER TABLE `challenge\_site\_challenge` ADD CONSTRAINT `owner\_id\_refs\_id\_7d499392` FOREIGN KEY (`owner\_id`) REFERENCES `challenge\_site\_userprofile` (`id`);

ALTER TABLE `challenge\_site\_challenge` ADD CONSTRAINT `league\_id\_refs\_id\_60cf6e6a` FOREIGN KEY (`league\_id`) REFERENCES `challenge\_site\_league` (`id`);

CREATE TABLE `challenge\_site\_challengeentry` (

`id` integer AUTO\_INCREMENT NOT NULL PRIMARY KEY,

`response` varchar(2000) NOT NULL,

`time\_started` datetime NOT NULL,

`time\_completed` datetime NOT NULL,

`correct` bool NOT NULL,

`challenge\_id` integer NOT NULL,

`owner\_id` integer NOT NULL

)

;

ALTER TABLE `challenge\_site\_challengeentry` ADD CONSTRAINT `owner\_id\_refs\_id\_147f2aff` FOREIGN KEY (`owner\_id`) REFERENCES `challenge\_site\_userprofile` (`id`);

ALTER TABLE `challenge\_site\_challengeentry` ADD CONSTRAINT `challenge\_id\_refs\_id\_1927ae16` FOREIGN KEY (`challenge\_id`) REFERENCES `challenge\_site\_challenge` (`id`);

CREATE TABLE `challenge\_site\_validentry` (

`id` integer AUTO\_INCREMENT NOT NULL PRIMARY KEY,

`name` varchar(200) NOT NULL,

`description` varchar(200) NOT NULL,

`response` varchar(2000) NOT NULL,

`challenge\_id` integer NOT NULL,

`owner\_id` integer NOT NULL

)

;

ALTER TABLE `challenge\_site\_validentry` ADD CONSTRAINT `owner\_id\_refs\_id\_757c1b06` FOREIGN KEY (`owner\_id`) REFERENCES `challenge\_site\_userprofile` (`id`);

ALTER TABLE `challenge\_site\_validentry` ADD CONSTRAINT `challenge\_id\_refs\_id\_208e63eb` FOREIGN KEY (`challenge\_id`) REFERENCES `challenge\_site\_challenge` (`id`);

CREATE INDEX `challenge\_site\_league\_5d52dd10` ON `challenge\_site\_league` (`owner\_id`);

CREATE INDEX `challenge\_site\_challenge\_7cc33bab` ON `challenge\_site\_challenge` (`league\_id`);

CREATE INDEX `challenge\_site\_challenge\_5d52dd10` ON `challenge\_site\_challenge` (`owner\_id`);

CREATE INDEX `challenge\_site\_challengeentry\_22741432` ON `challenge\_site\_challengeentry` (`challenge\_id`);

CREATE INDEX `challenge\_site\_challengeentry\_5d52dd10` ON `challenge\_site\_challengeentry` (`owner\_id`);

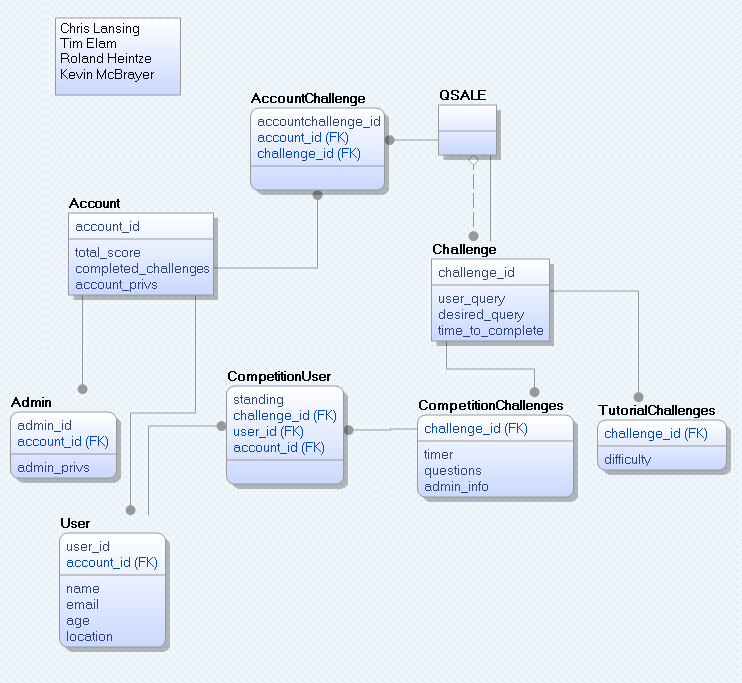
CREATE INDEX `challenge\_site\_validentry\_22741432` ON `challenge\_site\_validentry` (`challenge\_id`);

CREATE INDEX `challenge\_site\_validentry\_5d52dd10` ON `challenge\_site\_validentry` (`owner\_id`);

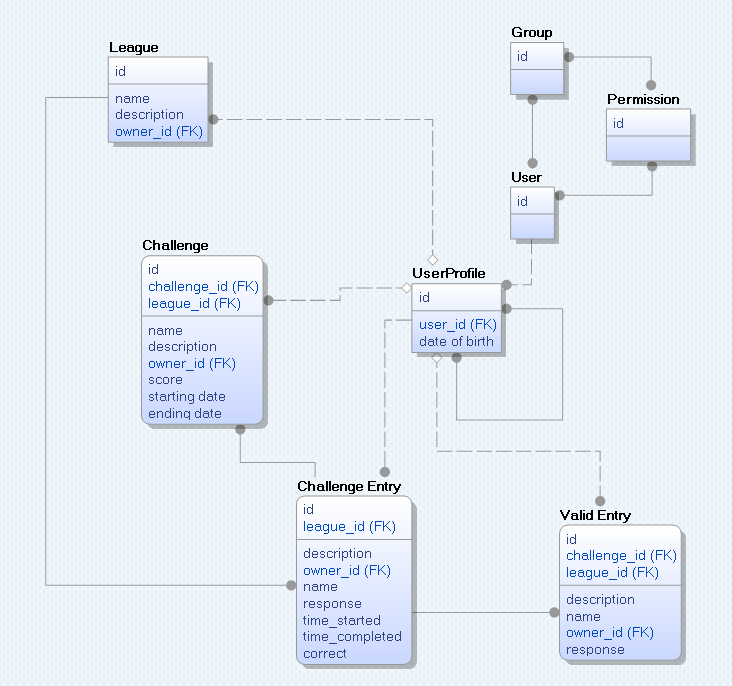
COMMIT;

# Logical Model

## Initial ER Diagram



## Final ER Diagram



# Physical Model

## Physical Model

DESCRIBE auth\_group;

Field Type Null Key Default Extra

id int(11) NO PRI NULL auto\_increment

name varchar(80) NO UNI NULL

DESCRIBE auth\_group\_permissions;

Field Type Null Key Default Extra

id int(11) NO PRI NULL auto\_increment

group\_id int(11) NO MUL NULL

permission\_id int(11) NO MUL NULL

DESCRIBE auth\_permission;

Field Type Null Key Default Extra

id int(11) NO PRI NULL auto\_increment

name varchar(50) NO NULL

content\_type\_id int(11) NO MUL NULL

codename varchar(100) NO NULL

DESCRIBE auth\_user;

Field Type Null Key Default Extra

id int(11) NO PRI NULL auto\_increment

username varchar(30) NO UNI NULL

first\_name varchar(30) NO NULL

last\_name varchar(30) NO NULL

email varchar(75) NO NULL

password varchar(128) NO NULL

is\_staff tinyint(1) NO NULL

is\_active tinyint(1) NO NULL

is\_superuser tinyint(1) NO NULL

last\_login datetime NO NULL

date\_joined datetime NO NULL

DESCRIBE auth\_user\_groups;

Field Type Null Key Default Extra

id int(11) NO PRI NULL auto\_increment

user\_id int(11) NO MUL NULL

group\_id int(11) NO MUL NULL

DESCRIBE auth\_user\_user\_permissions;

Field Type Null Key Default Extra

id int(11) NO PRI NULL auto\_increment

user\_id int(11) NO MUL NULL

permission\_id int(11) NO MUL NULL

DESCRIBE challenge\_site\_challenge;

Field Type Null Key Default Extra

id int(11) NO PRI NULL auto\_increment

name varchar(200) NO NULL

description varchar(200) NO NULL

score int(11) NO NULL

starting\_time datetime NO NULL

ending\_time datetime NO NULL

league\_id int(11) NO MUL NULL

owner\_id int(11) NO MUL NULL

DESCRIBE challenge\_site\_challengeentry;

Field Type Null Key Default Extra

id int(11) NO PRI NULL auto\_increment

response varchar(2000) NO NULL

time\_started datetime NO NULL

time\_completed datetime NO NULL

correct tinyint(1) NO NULL

challenge\_id int(11) NO MUL NULL

owner\_id int(11) NO MUL NULL

DESCRIBE challenge\_site\_league;

Field Type Null Key Default Extra

id int(11) NO PRI NULL auto\_increment

name varchar(200) NO NULL

description varchar(200) NO NULL

owner\_id int(11) NO MUL NULL

DESCRIBE challenge\_site\_league\_members;

Field Type Null Key Default Extra

id int(11) NO PRI NULL auto\_increment

league\_id int(11) NO MUL NULL

userprofile\_id int(11) NO MUL NULL

DESCRIBE challenge\_site\_userprofile;

Field Type Null Key Default Extra

id int(11) NO PRI NULL auto\_increment

dob date YES NULL

user\_id int(11) NO UNI NULL

DESCRIBE challenge\_site\_userprofile\_friends;

Field Type Null Key Default Extra

id int(11) NO PRI NULL auto\_increment

from\_userprofile\_id int(11) NO MUL NULL

to\_userprofile\_id int(11) NO MUL NULL

DESCRIBE challenge\_site\_validentry;

Field Type Null Key Default Extra

id int(11) NO PRI NULL auto\_increment

name varchar(200) NO NULL

description varchar(200) NO NULL

response varchar(2000) NO NULL

challenge\_id int(11) NO MUL NULL

owner\_id int(11) NO MUL NULL

DESCRIBE django\_admin\_log;

Field Type Null Key Default Extra

id int(11) NO PRI NULL auto\_increment

action\_time datetime NO NULL

user\_id int(11) NO MUL NULL

content\_type\_id int(11) YES MUL NULL

object\_id longtext YES NULL

object\_repr varchar(200) NO NULL

action\_flag smallint(5) unsigned NO NULL

change\_message longtext NO NULL

DESCRIBE django\_content\_type;

Field Type Null Key Default Extra

id int(11) NO PRI NULL auto\_increment

name varchar(100) NO NULL

app\_label varchar(100) NO MUL NULL

model varchar(100) NO NULL

DESCRIBE django\_session;

Field Type Null Key Default Extra

session\_key varchar(40) NO PRI NULL

session\_data longtext NO NULL

expire\_date datetime NO MUL NULL

## Data types and sizes

User:

date\_of\_birth: date type

friends: many to many field to User

username: char type length 20

password: encrypted char type length 20

League:

name: char type length 200

description: char type length 200

owner: foreign key to User

members: many to many field to User

Challenge:

name: char type length 200

description: char type length 200

score: int type

starting time: date time field

end time: date time field

league: foreign key to League

owner: foreign key to User

Challenge Entry:

response: char type length 2000

starting time: date time field

end time: date time field

correct: boolean

challenge: foreign key to Challenge

owner: foreign key to User

Valid Entry:

name: char type length 200

description: char type length 200

response: char type length 2000

challenge: foreign key to Challenge

owner: foreign key to User

# Application Planning

## Transactions/Data Entry

T1 - new USER and USER\_NAME and PASSWORD and/or DATE\_OF\_BIRTH and/or FRIENDS

- a new user is being created

T2 - new LEAGUE and/or LEAGUE\_NAME and/or DESCRIPTION and/or OWNER and/or MEMBERS

- a new league is being created

T3 - new CHALLENGE and CHALLENGE\_NAME and CHALLENGE\_DESCRIPTION and CHALLENGE\_SCORE and OWNER

- a new challenge is being posted its owner is automatically assigned to the user that created it.

T4 - new VALID\_ENTRY and/or NAME and/or DESCRIPTION and RESPONSE ( find CHALLENGE and OWNER)

- a new valid entry is being created

T5 - new CHALLENGE\_ENTRY and RESPONSE and TIME\_START update TIME\_COMPLETE and/or CORRECT (find OWNER)

- a new challenge entry is being created the owner is automatically set to user who created it, TIME\_START is set on creation TIME\_COMPLETE is updated on post.

T6 - update USER and/or USER\_NAME and/or DATE\_OF\_BIRTH and/or FRIENDS

- updating the user information except password(security)

T7 - update LEAGUE and/or LEAGUE\_NAME and/or description and/or OWNER and/or MEMBERS

-updating league information, adding or removing members and changing OWNER

T8 - update CHALLENGE and/or CHALLENGE\_NAME and/or CHALLENGE\_DESCRIPTION and (T9)

-updating or changing a challenge the user must create a new valid entry if they modify the challenge so we link this transaction to T9

T9 - update VALID\_ENTRY and/or NAME and DESCRIPTION (find CHALLENGE and OWNER)

- updating a valid entry

## Environments, DBA aspects, Backups

We will be able to backup and restore our MySQL database using the tool mysqldump. This command creates a .sql file with sql statements to drop and create tables, as well as perform inserts for all of the data of the source database. Restoring the database from the dump is easy; all you have to do is execute the .sql file in your destination database. A backup and restore of a database can be done in only 2 commands:

*backup*: # mysqldump -u root -p[password] [db name] > filename.sql

*restore*: # mysql -u root -p[password] [db name] < filename.sql

## Security

Our website uses the Django framework and a MySQL database. The two most common types of security risks are SQL injections and Cross-Site Scripting (XXS).

Our website is safe from SQL injection attacks because the Django database API automatically escapes all special SQL parameters according to the quoting conventions that our current database (MySQL) is using.

Django’s template system also takes care of this security risk by automatically escaping all variable value that a user inputs.

For more detailed information regarding security, refer to the official Django security documentation:

http://www.djangobook.com/en/2.0/chapter20.html

# Graphical User Interface

http://db.frsilent.com

# Post Mortem

## Chris Lansing

Doing this project was a valuable learning experience for me. The work that we did with SQL databases and websites will help me with my current and future job. Because of all the modeling and designing involved in creating the database for our project, I feel that I also have a much more robust understanding of databases and how they work. It was also nice working on a project that the entire team is interested in and one that may be worked on after the class is over as a hobby. I really enjoyed working with my team. Everyone brought a unique set of skills to the team as well as a strong general knowledge of what we were working on. I think that the only major improvement that could be made to the group is to start our documentation and final paper from the beginning of the class, in order to more easily assemble the project report.

## Tim Elam

I learned a lot by doing this project. Primarily I learned a few harsh lessons about managing complexity. For starters, using built-in alternatives to writing simple functions can add complexity to a project. Hiding implementation details deep in the source code of a framework is costly when you are not extremely familiar with the framework. Also, switching to a pre-made implementation for functions can lead to a costly refactoring process if the requirements change to necessitate the use of custom code in place of boilerplate. That being said, using a framework with an ORM has been fantastic. I have been able to focus on the logical model without worrying about the specific syntax of the underlying database. I still had to run custom SQL scripts to modify the database when I changed the schema, but I agree with the creators of the framework that this is a task best done by hand on the rare occasion that the database schema changes.

## Roland Heintze

It was great to work with a solid group in which work was accomplished quickly & efficiently. I believe the only shortcoming of our project was a changing list of requirements and functionality. In the future I would like to set up a working project first; with a solid backend but very few features. Then we could add the design and features to a system without facing the challenges of a poor design having to be changed in the end production. In the future I believe I would like to work with the same group but have more collaboration & involvement in the early design of the application & data model. Overall, it was a good experience with the project though and Django was a great platform for our purposes.

## Kevin McBrayer

It was great to experience a group project where all of the members brought a wealth of knowledge. Working on this project gave me an insight into building an application with rolling requirements and how to communicate with my team mates. Looking back I really enjoyed working on a dynamic database driven web application and even some of the peripheral technologies like CSS3 transitions and a new version of the framework Django. The only down side to a project like this was the documentation. I would have started on it much earlier or have made documentation points during the coding process. This project was a fun and educating experience.