

**BILKENT UNIVERSITY**

**DEPARTMENT OF COMPUTER ENGINEERING**

**CS353**

**Database Systems**

**Design Report**

Social Network and Forum for Researchers

**Group 11**

**Group Members**

Ahmet Ay 21200701

Berk Bozkurt 21101098

Cüneyt Erem 21202398

Mert Gurcan, 21101646

**Instructor**

A. Ercüment Çiçek

Table of Contents

1. Revised E/R Diagram.......................................................................................................... 3

2. Relations ............................................................................................................................ 4

2.1 User ...................................................................................................................... 4

2.2 Author ................................................................................................................... 4

2.3 Department ........................................................................................................... 4

2.4 Institutiton.............................................................................................................. 5

2.5 Post....................................................................................................................... 5

2.6 Article.................................................................................................................... 5

2.7 Question................................................................................................................ 6

2.8 Topic ..................................................................................................................... 6

2.9 Tag ....................................................................................................................... 6

2.10 Commend ........................................................................................................... 6

2.11 Writes ................................................................................................................. 7

2.12 Likes .................................................................................................................. 7

2.13 Sends ................................................................................................................. 7

2.14 Belongs............................................................................................................... 8

2.15 HasInst ............................................................................................................... 8

2.16 HasTop ............................................................................................................... 9

2.17 Favorite .............................................................................................................. 9

2.18 Hastag .................................................................................................................9

2.19 Follow ................................................................................................................10

2.20 Reference...........................................................................................................10

2.21 ForeignReference...............................................................................................10

2.22 HasForeignReference........................................................................................11

3. Functional Dependencies and Normalization of Tables ....................................................11

4. Functional Components ....................................................................................................11

4.1 Algorithms ...........................................................................................................11

4.2 Use Case Diagram...............................................................................................11

4.2.1User........................................................................................................12

4.2.Author.......................................................................................................12

4.3 Scenarios ........................................................................................................................13

4.4 Data Structures................................................................................................................13

5. SQL and GUI .....................................................................................................................14

5.1 Login Register......................................................................................................14

5.2 Question...............................................................................................................15

5.3 Publications..........................................................................................................17

5.4 Home....................................................................................................................19

5.5 CreatingArticle......................................................................................................20

6. Advanced Database Components ....................................................................................21

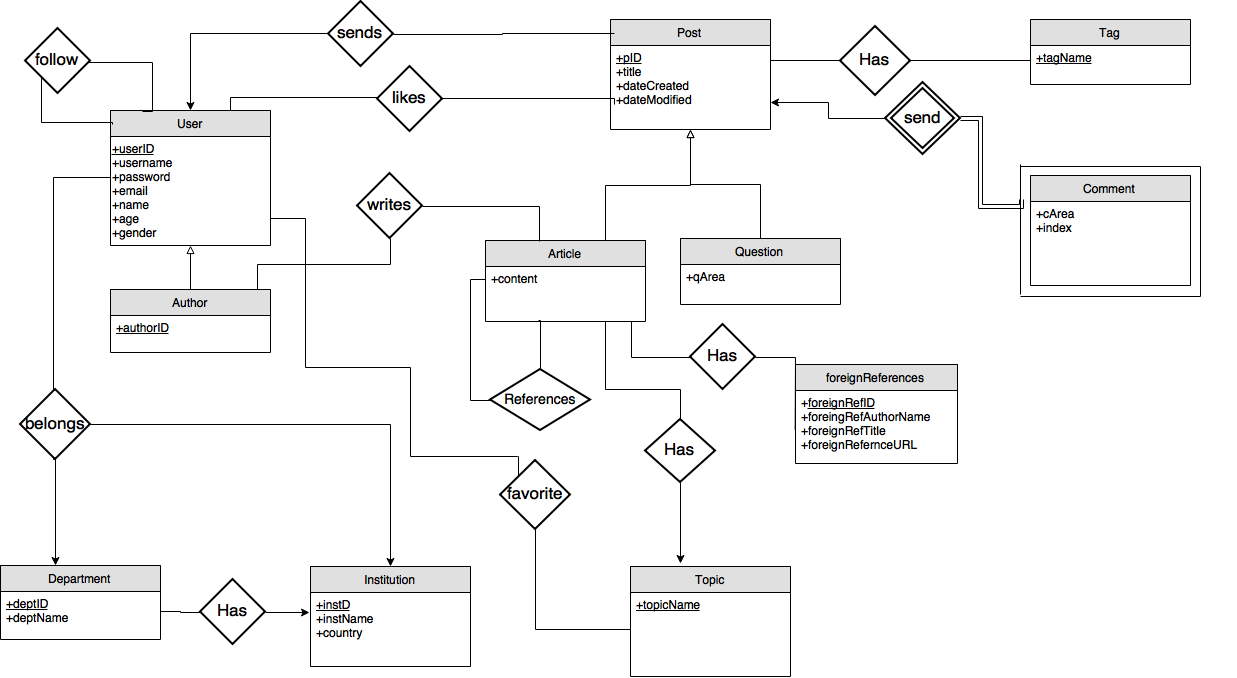
6.1 Triggers................................................................................................................21

6.2 Views....................................................................................................................21

6.3 Constraints ..........................................................................................................21

7. Conclusion ......................................................................................................................21

1. **Revised ER Diagram**



1. **Set of Relations**

There are attribute domains, candidate keys (and the selected, primary key), and foreign keys due to referential integrity was well as 3NF type.

* 1. **User**

**craete table** User(

userID **int** (8) **not null AUTO\_INCREMENT,**

username **varchar**(30) **not null**,

password **varchar**(30) **not null,**

email **varchar**(30) **not null,**

name **varchar**(30) **not null**,

age **int**(2) **not null,**

gender **varchar**(8) **not null,**

**primary key** (userID ),

)

**Functional Dependencies:**

userID -> username , password, email

**Candidate key:**

**(**username, password)

**Normal Form:** 3NF

---

* 1. **Author**

**create table** Author(

authorID **int**(8) **not null AUTO\_INCREMENT,**

userID **int** (8) **not null,**

**primary key** (authorID ),

**foreign key** (userID) **references** User

**);**

**Functional Dependencies:**

authorID -> userID

**Candidate key:**

userID

**Normal Form:** 3NF

---

* 1. **Department**

**create table** Department(

deptID **int** (8) **not null AUTO\_INCREMENT,**

deptName **varchar(**30) **not null,**

**primary key(**deptID **);**

);

**Functional Dependencies:**

deptID -> deptName

**Candidate key:**

**none**

**Normal Form:** 3NF

---

* 1. **Institution**

**create table** Institution(

instID **int** (8) **not null AUTO\_INCREMENT,**

instName **varchar**(30) **not null,**

country **varchar**(30) **not null,**

**primary key(**instID **);**

**);**

**Functional Dependencies:**

instID -> instName , country,

**Candidate key:**

**(**instID,instName),instID

**Normal Form:** 3NF

---

* 1. **Post**

**create table** Post(

pID **int** (8) **not null AUTO\_INCREMENT,**

title **varchar**(30) **not null,**

dateCreated **varchar**(30) **not null,**

dateModified **varchar**(30),

**primary key** (pID )

);

**Functional Dependencies:**

pID -> title, dateCreated, dateModified

**Candidate key:**

pID

**Normal Form:** 3NF

---

* 1. **Article**

**create table** Article(

content **blob** **not null,**

pID **int** (8) **not null,**

**primary key(**pID **);**

**foreign key** (pID) **references** Post

);

**Functional Dependencies:**

pID -> content

**Candidate key:**

pID

**Normal Form:** 3NF

---

* 1. **Question**

**create table** Question(

qArea **varchar**(30) **not null,**

pID **int** (8) **not null,**

**primary key(**pID **);**

**foreign key** (pID) **references** Post

);

**Functional Dependencies:**

pID -> qArea

**Candidate key:**

pID

**Normal Form:** 3NF

---

* 1. **Topic**

**create table** Topic(

topicName **varchar**(30) **not null,**

**primary key(**topicName **);**

);

**Functional Dependencies:**

none

**Candidate key:**

topicName

**Normal Form:** 3NF

---

* 1. **Tag**

**create table** Tag(

tagName **varchar**(30) **not null,**

**primary key(**tagName **);**

);

**Functional Dependencies:**

none

**Candidate key:**

tagName

**Normal Form:** 3NF

---

* 1. **Comment**

**create table** Comment(

pID **int** (8) **not null AUTO\_INCREMENT,**

cArea **varchar(1000) no null,**

index **int** (8) **not null,**

**primary key(**index, pID**);**

**foreign key** (pID) **references** Post

);

**Functional Dependencies:**

none

**Candidate key:**

index, pID

**Normal Form:** 1NF

---

* 1. **writes**

**create table** writes(

authorID **int** (8) **not null,**

pID **int** (8) **not null,**

**primary key(**authorID **);**

**foreign key** (authorID ) **references** Author

**foreign key** (pID ) **references** Post

);

**Functional Dependencies:**

none

**Candidate key:**

pID,authorID

**Normal Form:** 3NF

---

* 1. **likes**

**create table** likes(

userID **int** (8) **not null,**

pID **int** (8) **not null,**

**primary key(**userID **);**

**foreign key** (userID ) **references** User

**foreign key** (pID ) **references** Post

);

**Functional Dependencies:**

none

**Candidate key:**

userID,pID

**Normal Form:** 3NF

---

* 1. **sends**

**create table** sends(

userID **int** (8) **not null,**

pID **int** (8) **not null,**

**primary key(**userID **);**

**foreign key** (userID ) **references** User

**foreign key** (pID ) **references** Post

);

**Functional Dependencies:**

pID -> userID

**Candidate key:**

pID,userID

**Normal Form:** 3NF

---

* 1. **belongs**

**create table** belongs(

userID **int**(8) **not null,**

deptID **int** (8) **not null,**

instID **int** (8) **not null,**

**primary key**(username);

**foreign key** (userID ) **references** User

**foreign key** (deptID ) **references** Departmen

**foreign key** (instID ) **references** Institution

);

**Functional Dependencies:**

username -> deptID, instID

**Candidate key:**

username

**Normal Form:** 3NF

---

* 1. **hasInst**

**create table** hasInst(

instID **int** (8) **not null,**

deptID **int** (8) **not null,**

**primary key**(instID);

**foreign key** (deptID ) **references** Departmen

**foreign key** (instID ) **references** Institution

);

**Functional Dependencies:**

deptID -> instID

**Candidate key:**

none

**Normal Form:** 3NF

* 1. **hastop**

**create table** hastop(

pID **int** (8) **not null,**

topicName **varchar**(30) **not null**

**primary key**(pID);

**foreign key** (topicName ) **references** Topic

);

**Functional Dependencies:**

pID -> topicName

**Candidate key:**

pID

**Normal Form:** 3NF

---

* 1. **favorite**

**create table** favorite(

userID **int** (8) **not null,**

topicName **varchar**(30) **not null,**

**primary key**(userID);

**foreign key** (userID ) **references** User

**foreign key** (topicName ) **references** Topic

);

**Functional Dependencies:**

none

**Candidate key:**

userID

**Normal Form:** 3NF

---

* 1. **hastag**

**create table** hastag(

pID **int** (8) **not null,**

tagName **varchar**(30) **not null**

**primary key**(pID);

**foreign key** (tagName ) **references** Tag

);

**Functional Dependencies:**

none

**Candidate key:**

pID

**Normal Form:** 3NF

* 1. **follow**

**create table** follow(

followerID **int**(8) **not null,**

targetID **int(8) not null,**

**primary key**(followerID, targetID );

**foreign key** (targetID ) **references** User

**foreign key** (followerID ) **references** User

)

**Functional Dependencies:**

none

**Candidate key:**

(followerID, targetID )

**Normal Form:** 3NF

---

* 1. **reference**

**createn table** follow(

refID **int**(8) **not null,**

targetrefID **int(8) not null,**

**primary key**(refID, targetrefID );

**foreign key** (refID ) **references** Article

**foreign key** (targetrefID ) **references** Article

)

**Functional Dependencies:**

none

**Candidate key:**

(refID, targetrefID )

**Normal Form:** 3NF

---

* 1. **foreignReferences**

**createn table** foreignReferences(

foreignRefID **int**(8) **not null,**

foreignRefAuthorName **varchar**(30) **not null,**

foreignRefTitle **varchar**(30) **not null,**

foreignReferenceURL **varchar**(200) **not null,**

**primary key**(foreignRefID );

);

**Functional Dependencies:**

foreignRefID ->foreignRefAuthorName,foreignRefTitle,foreignReferenceURL

**Candidate key:**

foreignRefID,(foreignRefAuthorName,foreignRefTitle)

**Normal Form:** 3NF

* 1. **hasForeignReferences**

**create table** hasForeignReferences(

pID **int** (8) **not null,**

foreignRefID **int**(8) **not null,**

**primary key**(pID);

**foreign key** (foreignRefID ) **references** foreignReferences

);

**Functional Dependencies:**

none

**Candidate key:**

pID,foreignRefID

**Normal Form:** 3NF

**3. Functional Dependencies and Normalization of Tables**

Please see Section 2.

**4. Functional Components**

Functional components of our system are explained in detail below.

**4.1 Algorithms**

1. Once user likes an article or question its like count must be increased and like option of that entity must be blocked to restrict triggering like operation more than once.

2. Commenting on other articles and questions are limited to certain number. Commenting more than this limit will be evaluated as spam.

**4.2 Use Case Diagrams**

Our system has two different user types; user and foreign author. Users need to register to system to get additional features. Foreign authors can also use system by providing their names.



**4.2.1 User**

· Users can register with username, password, age, gender and email.

· User can login system with username and password.

· User can post questions.

· User can like articles and questions

· User can select his/her favorite topics.

· User can comment on articles and questions.

**4.2.2 Author**

In additon to User features

· Authors can post articles

**4.3. Scenarios**

**Sending Posts (User):** User Jimmy logs in to the system. He wants to post an article or a question. However in order to post an article he must be an author. Due to the fact that normal users cannot post article, they can only post question in the system. He registers as an author and writes an article with an interesting topic. His article also contains plenty of references references internal and external articles.

**Comment (User):** Michael reads Jimmy’s article and want to add his thoughts to his article. Michael logs in the system and post a comment on his article.

**Indicating Favourites (User):** After Michael register the system he indicates his favourite topics and areas in order to be informed pots related to his interests.

**Like Article (User):** A curious user Seren logs in to the system. She really like reading articles about different topics. She clicks on like button when she really like an article. She want to support and promote author of that article to write and post such writings.

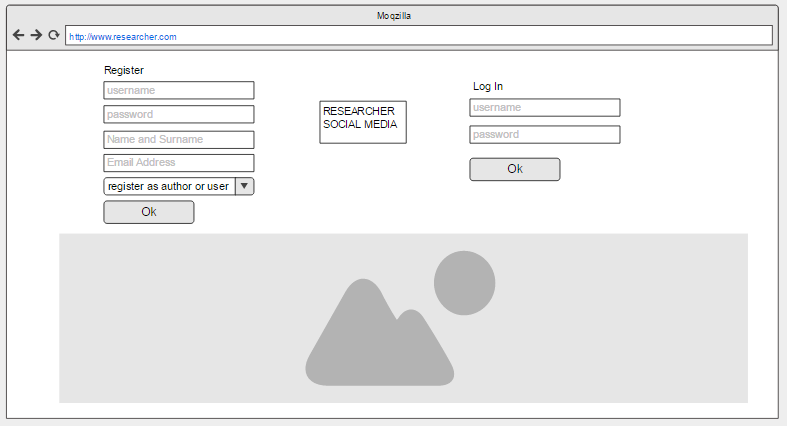
**Follow User (User):** Mert reads the writings of many different authors. However, he does not want to miss new writings of these writers because he likes the writings of some of them. For this reason, Mert follows these authors and becomes aware of new shares.

**4.4 Data Structures**

We can use lists for listing users, questions or articles. Database provides structural requirements like varchar, numeric and int. Any other structures apart from these would be unnecessary.

**5. SQL and GUI**

**5.1 Login/ Register**



**Register Explanation:**First, user enters username, password, full name, user type and email address. Then, user clicks register button. If username is allowed, then user will be registered.

**Inputs : $username , $password, $full\_name, $email**

**Duplicate username check:**

SELECT \*

FROM User

WHERE username=$username

**Registering User:**

INSERT INTO User(username, password, name, email) VALUES ($username, $password, $full\_name, $email)

**Registering Author:**

INSERT INTO User(username, password, name, email) VALUES ($username, $password, $full\_name, $email)

**Get $userID for new registered User**

SELECT userID

FROM User

WHERE username=$username

**Create Author Relation of User**

INSERT INTO Author(user\_id) VALUES ($userID)

**Login Explanation:** First, user enters username and password. Then, user clicks for login. If the login information is correct, then user will be directed to the main page.

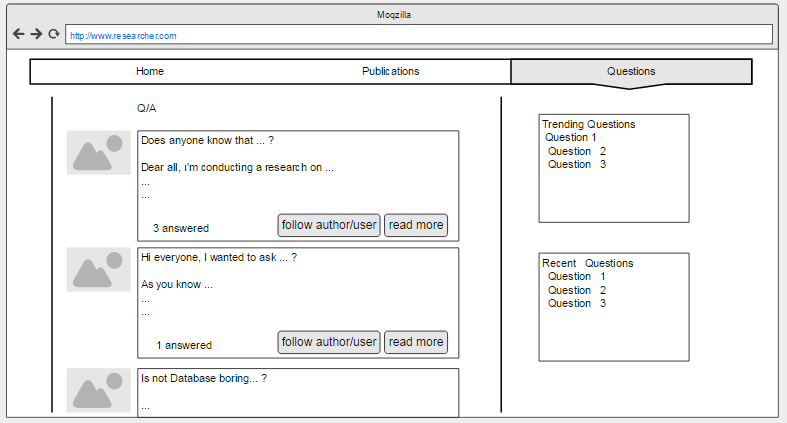
**Inputs : $username , $password**

SELECT \*

FROM User

WHERE user\_name=$user\_name AND password=$password

**5.2. Question**



**Questions Explanation:**In questions page user will see his/her followed users’ questions

**Page**

**Inputs: $userID**

SELECT Post.title, Post.dateCreated, Question.qArea

FROM Post

JOIN Question ON Post.pID = Question.pID

JOIN sends ON Post.pID = sends.pID

WHERE sends.userID IN

(

SELECT \*

FROM follow

WHERE refID = $userID)

ORDER BY Post.dateCreated

**Recent Questions**

SELECT Post.title

FROM Post

JOIN Question ON Post.pID = Question.pID

ORDER BY Post.dateCreated

LIMIT 3

**Trending Questions**

SELECT Post.title

FROM (SELECT Post.pID, COUNT ( Post.pID) as like\_count,

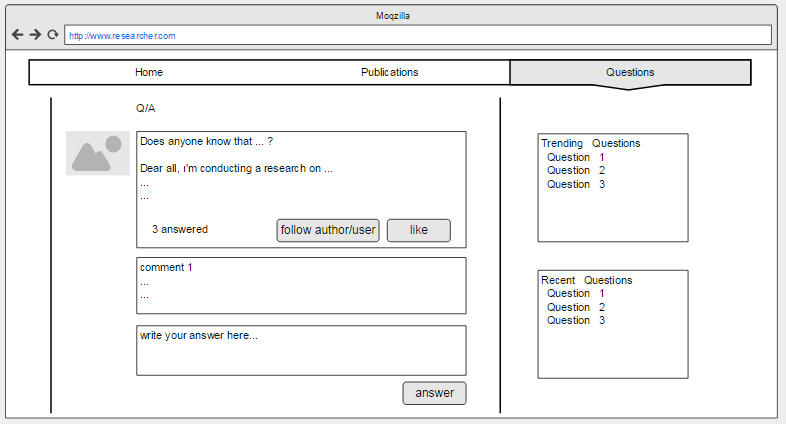
FROM like

JOIN Post ON Post.pID= like.pID

GROUP BY like.pID)

ORDER BY like\_count DESC

LIMIT 3



**Following User**

**Inputs: $userID $targetID**

INSERT INTO follow (followerID, targetrefID) VALUES ($userID, $targetID)

**Liking Post**

**Inputs: $userID $postID**

INSERT INTO likes (userID, pID) VALUES ($userID, $postID)

**Commenting a Post**

**Inputs: $userID $cArea $pID**

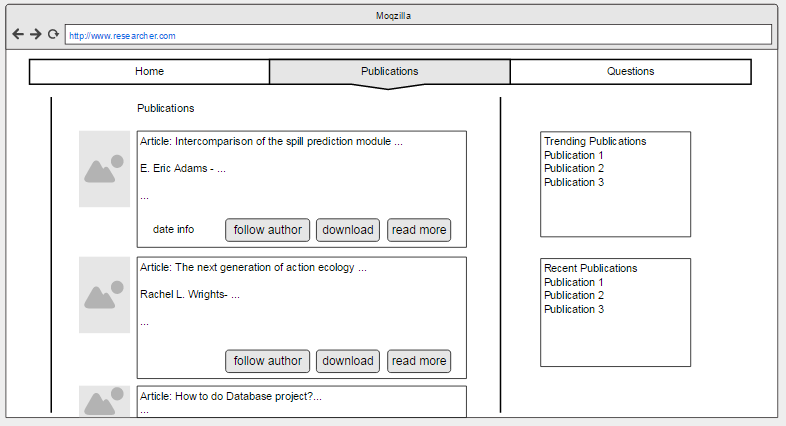
INSERT

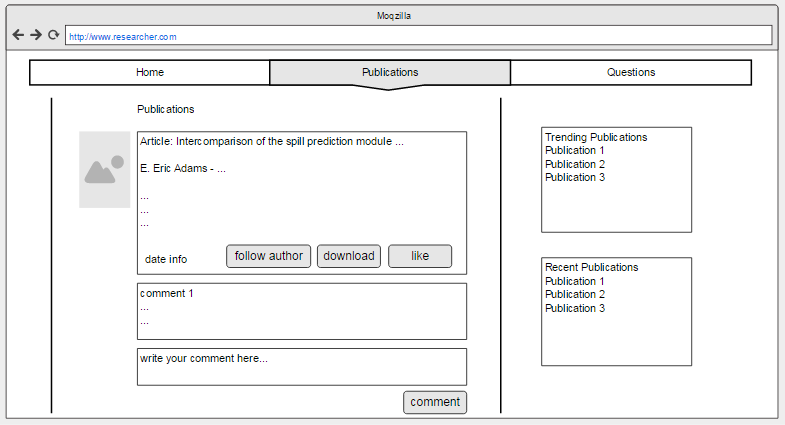
INTO Comment (pID, cArea, index)

VALUES($pID, $cArea,

(SELECT Max(index)+1 FROM Comment WHERE Comment.pID = $pID)

**5.3. Publications**





**Publications Explanation:**In publications page user will see his/her followed users’ publications

SELECT Post.title, Post.dateCreated, Article.content

FROM Post

JOIN Article ON Post.pID = Article .pID

JOIN writes ON Post.pID = writes.pID

WHERE writes.userID IN

(

SELECT \*

FROM follow

WHERE refID = $userID)

ORDER BY Post.dateCreated

**Recent Publications**

SELECT Post.title

FROM Post

JOIN Article ON Post.pID = Article .pID

ORDER BY Post.dateCreated

LIMIT 3

**Trending Publications**

SELECT Post.title

FROM (SELECT Post.pID, COUNT ( Post.pID) as like\_count,

FROM like

JOIN Post ON Post.pID= like.pID

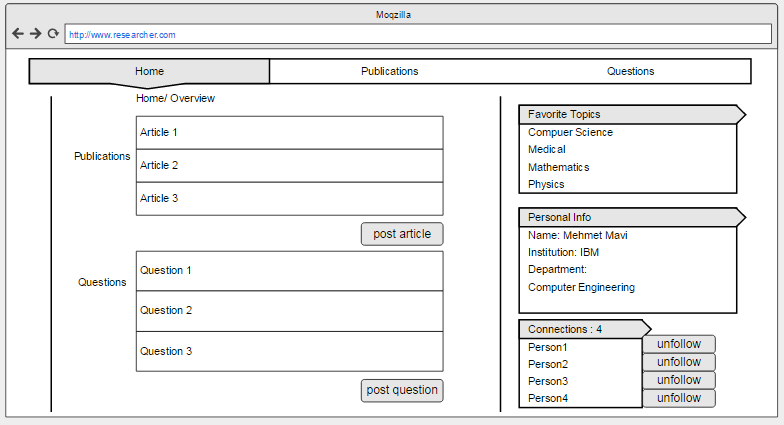
GROUP BY like.pID)

ORDER BY like\_count DESC

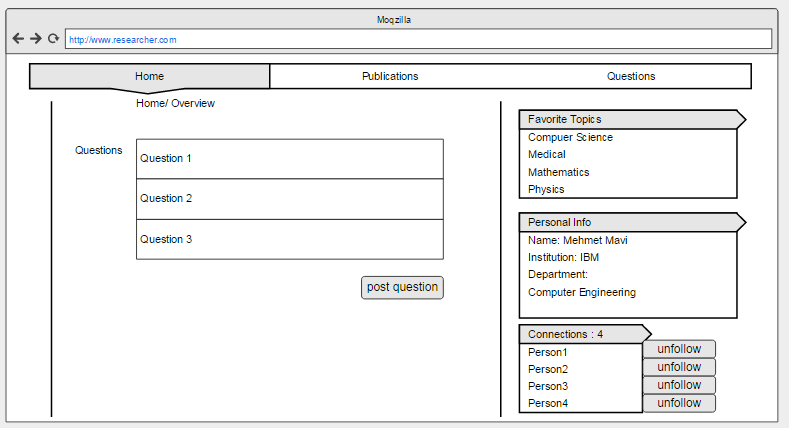
LIMIT 3

**5.4. Home**

**Author**



**User**



**Home, User Explanation:** User will see his/her favorite topics, personal informations, followed people and posted questions on detail page.

**Inputs: $userID**

**Getting Favorite Topics**

SELECT \* FROM favorite

JOIN User ON favorite.userID = $userID

**Getting Followers**

SELECT \*

FROM User

WHERE User.userID IN

(SELECT \* FROM follow WHERE follow.followerID = $userID)

**Getting Institution Information**

SELECT Institution.instName Department.deptName FROM Institution, Department

JOIN belongs ON

belongs.userID = $userID, AND

Department.deptID = belongs.deptID AND

Institution.instID = belongs.instID

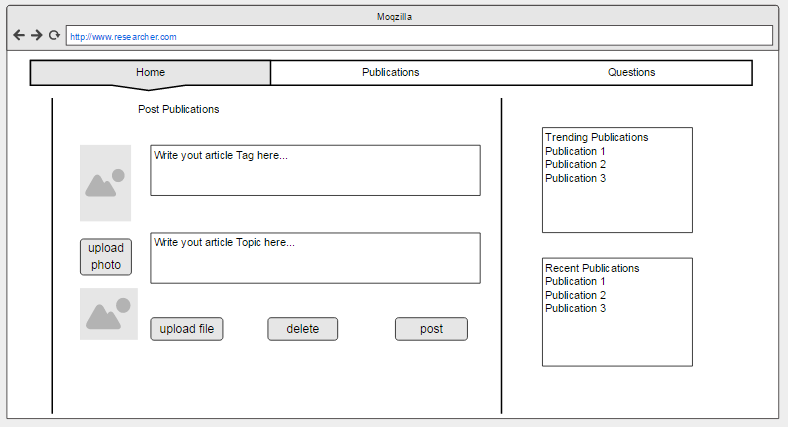
**Getting Asked Questions and Articles**

SELECT \*

FROM Post

JOIN sends ON sends.userID = $userID

**5.5. Creating Article**



**Explanation:** User creates a new article and sets a tag. If that tag is not already created we create new one.

**Inputs: $content, $dateCreated, $title, $tag**

INSERT INTO Post (title, dateCreated, dateModified) VALUES ($title, $dateCreated, NULL)

SELECT scope\_identity() as newID

// scope\_identity() returns last generated identity value for current scope which in this case Post.pID, We store it in $newID)

INSERT INTO Article(content, pID) VALUES ($newID, $content)

SELECT \* FROM tag WHERE tagName = $tag // If this returns no rows we insert new tag

INSERT INTO hastag(pID, tagName) VALUES ($newID, $tag)

**6. Advanced Database Components**

**6.1 Triggers**

**●** If a person post a comment on an article of a user, that user will be informed.

**●** If a share is made about the topic that the user is interested in, the user will be informed.

**●** If the user is followed by another user, the user will be notified.

**●** If a user's article refers to another user's article, the user to whom the article is referenced will be notified.

**6.2 Views**

**Article view in decending time order**

**create view** last\_posts **as**

**select** A.title, A.content, A.dateCreated

**from** Article A o**rders by** dateCreated **desc**;

**6.3 Constraints**

A person can read one question/article etc at the same time

Author can publish article, post question and define favorite topics

User can post question and define favorite topics except posting article

Because User cannot publish an article, other author can refer his artickle with user’s info

Both user and author can follow people and determine favorite topics

Both user and author can see recent publications and questions while they are looking at other publications or questions

**7. Conclusion**

In the design report, the project was evaluated by taking conceptual design as base and it coverred basic and significant structures. Relations, their keys and functional dependencies were determined. The user interface of the website is prepared and specific queries for different operations were written.