Leavy School of Business, Santa Clara University.

Final Project Report, Database Management Sysytems, MSIS 2603

FRIENDSPOST

Submitted to :

Professor Tejpal Chadha

Submitted by:

Megha Sharma

TABLE OF CONTENTS

Business Application Description

1.1 Introduction

This is the era of interaction between people to people through social platforms in the web. The tremendous popularity of using the social media networking could never have been realized before. This is the beginning of interweaving social media networking as a positive business process bringing two people together than merely two organizational entities. People can communicate their message to the suppliers or customers and convey any difference of opinion over the social networks. Social media networking is a more practical approach for dealing with the consumers. It may be for improving the products listening to the customers' views with critical appreciations. Take the example of activities the companies like IBM, Microsoft and Google performs to strengthen their networking platforms via social networks like Twitter and Facebook.

But with increasing number of users on social media, the volume of data is increasing too. There is competition in the market as to which platform is faster, which has the latest features, etc. The project tries to provide a solution to these challenges, how to increase user base without letting it affect the performance of the application.

1.2 Objective

- To provide a one stop communication platform- To provide a single platform to users where they can connect and communicate with other people, put up blog posts, share and discuss their opinions, do online promotions, share the commonalities with other people on the network, share some ideas, interests through their posts.
- To provide a storage solution to users- The users can upload any number of photos and videos, which will remain saved as long as the user's account is active. The user data will be safeguarded and will remain available to the user anytime and anywhere.
- To provide user data for analysis.

1.3 Scope

This project's purpose is to provide a solution against the increasing number of users and increasing amount of user data and letting it impact the performance. For this, the initial implementation of the project is being done in SQL as we need the user data to be organized and we would prefer the SQL schema since it would work well from a large to small number of users.

The NoSQL part would be used for the blog pages of the users where there will be large volume of data, so the flexible, scalable and denormalized structure of non-relational database would be preferred. Also, if any updates need to be made to the website or new features need to be added, it can be implemented quickly compare to SQL.

2.1 User Types-

1. **User-** they create their account on Friendpost by registering first. Once registered, they can access their profile which they can edit/update at any time. They users can add pictures, videos, put up blog posts, share URLs etc. The user content will remain saved as long as their account is active.

2. **Administrator**- The administrator manages the application. They have both read and write access and they maintain user profiles and user activities.

3. **Analysts**- they are the data analysts of friendspost who analyze the user base and user data to make business decisions

4. **Employee of survey portals**- The survey portal employees require data from the administrator to proof survey results and help business organization design promotion strategies based on target audience.

2.2 Tables

1. Registration Table

2. Profile Table

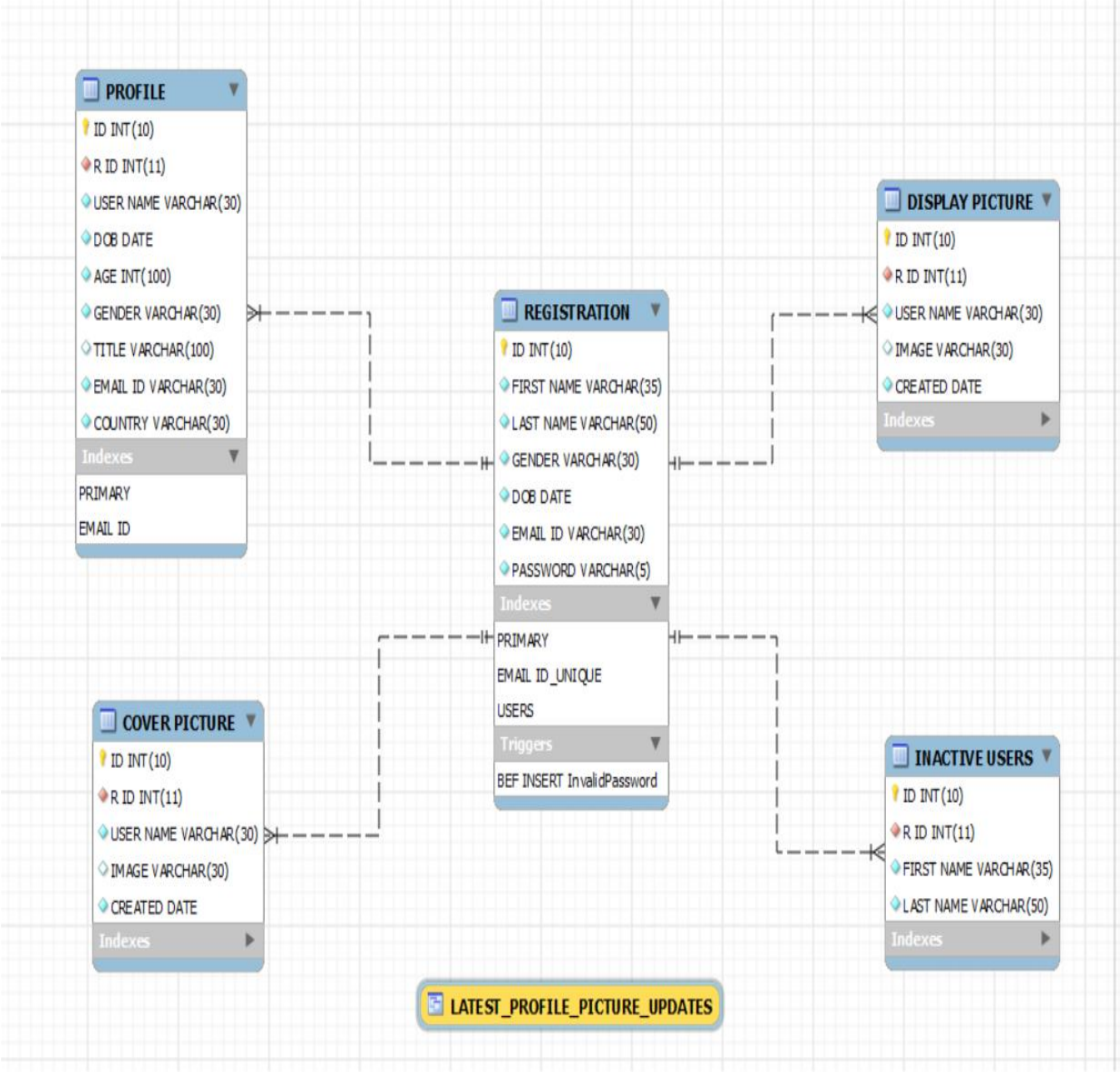
3. Display Picture

4. Cover Picture

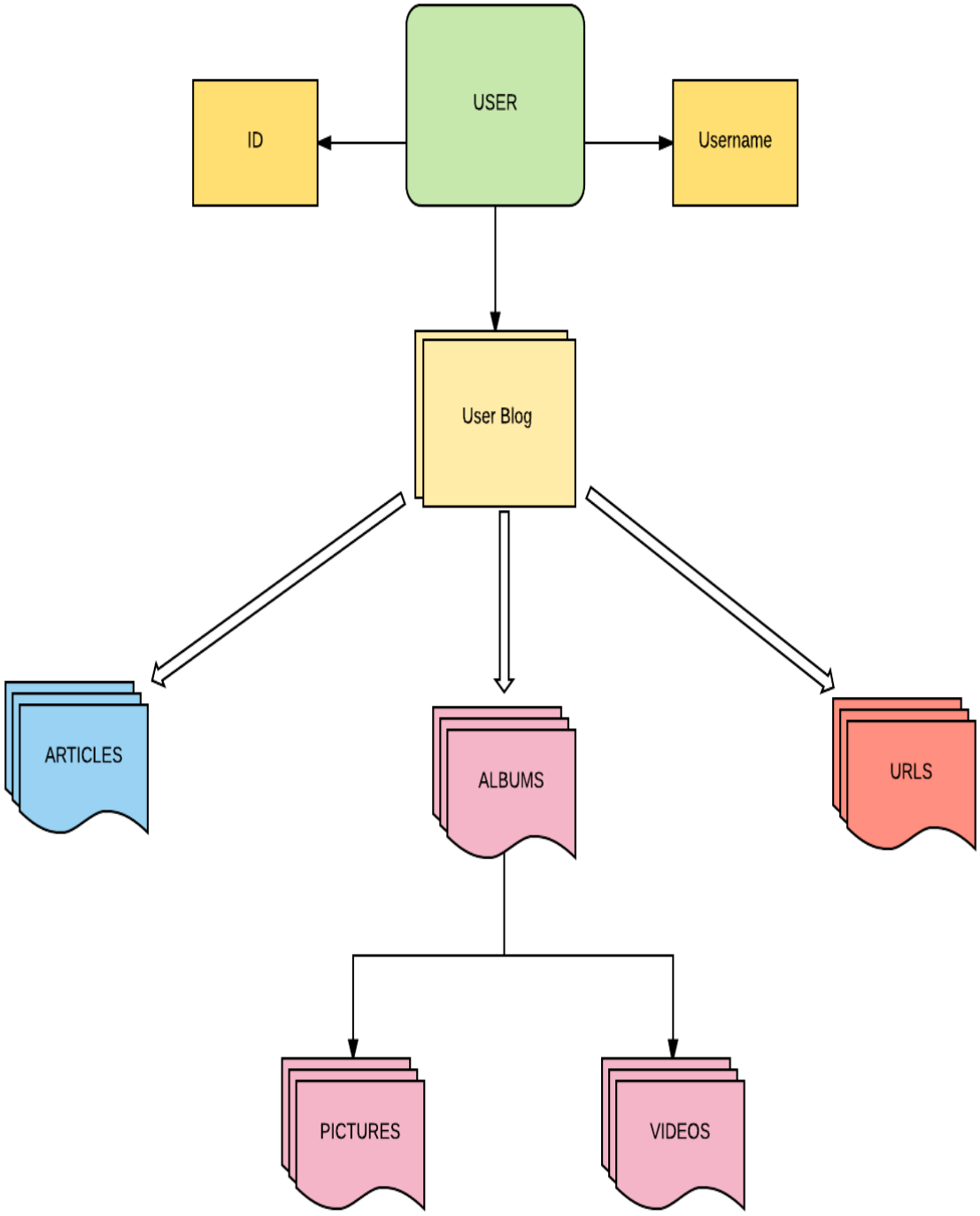
5. Inactive users

Logical Schema

UML Model



Mongo Data Structure



Use cases-

1. **User registers with the application**

    - User opens the web page of the application and clicks on ‘Sign Up’.
    - User enters the following details :
      -First Name
      -Last Name
      -Gender
      -DOB
      -Email Id
      -Password
    - User clicks on ‘Sign me up on Friendspost’.
    - User Account is created.

2. **User updates their profile details**

    - Userclicks on Login on the application's web page.
    - User types the email Id as User ID.
    - User types the user's password and clicks on ‘Login into my account’.
    - User selects the "edit my profile" option.
    - User updates their new user name.
    - User can give themselves a title.
    - User updates their country name.
    - User uploads their profile picture.
    - User updates their cover picture

3. **User uploads blog posts**

User can put the following posts-

- User uploads pictures on their profile.
- User uploads videos on their profile.
- User posts articles on their profile.
- User posts URLs on their profile.

4. Administrator

Administrator can perform the following functions-

- Administrator can fetch user data.
- Administrator manages the user account and user activities.
- Administrators retrieve the data required by the online survey employees and the data analysts of the application.
- Administrator can provide or revoke accesses.
- They are responsible for adding new features to the application.

5. Data Analysts

- The data analysts analyze the business metrics designed from the user data to take business decisions like increasing customer base in regions where the application is less popular, or how to make the application more popular in comparison to other applications in the market.
- They also analyze the user activities/posts to propose new additions or features to the application.

6. Survey Portal Employee

- They have read access to the website.
- They can ask for user data from website administrators and data analysts for their surveys. Based on the results of user data, they can make marketing decisions and business strategies.

Physical Schema – Database Dictionary

1. REGISTRATION TABLE

NAME	TYPE	CONSTRAINT
ID	INT (11)	NOT NULL, AUTO INCREMENT, **PRIMARY KEY**
FIRST_NAME	VARCHAR (35)	NOT NULL
LAST_NAME	VARCHAR (50)	NOT NULL
GENDER	VARCHAR (30)	NOT NULL
DOB	DATE	NOT NULL
EMAIL ID	VARCHAR (30)	NOT NULL, UNIQUE
PASSWORD	VARCHAR (5)	NOT NULL

2. PROFILE TABLE

NAME	TYPE	CONSTRAINT
ID	INT (11)	NOT NULL, AUTO INCREMENT, **PRIMARY KEY**
R_ID	INT (11)	NOT NULL, **FOREIGN KEY**
USER_NAME	VARCHAR (30)	NOT NULL
AGE	INT (100)	NOT NULL
DOB	DATE	NOT NULL
GENDER	VARCHAR (30)	NOT NULL
TITLE	VARCHAR (100)	
EMAIL ID	VARCHAR (30)	NOT NULL,**UNIQUE KEY**
COUNTRY	VARCHAR (30)	NOT NULL

3. DISPLAY PICTURE

NAME	TYPE	CONSTRAINT
ID	INT (11)	NOT NULL, AUTO INCREMENT, **PRIMARY KEY**
R_ID	INT (11)	NOT NULL, **FOREIGN KEY**
USER_NAME	VARCHAR (30)	NOT NULL
IMAGE	VARCHAR (30)	
CREATED	DATE	NOT NULL

4. COVER IMAGE

NAME	TYPE	CONSTRAINT
ID	INT (11)	NOT NULL, AUTO INCREMENT, **PRIMARY KEY**
R_ID	INT (11)	NOT NULL, **FOREIGN KEY**
USER_NAME	VARCHAR (30)	NOT NULL
IMAGE	VARCHAR (30)	
CREATED	DATE	NOT NULL

5. INACTIVE USERS

NAME	TYPE	CONSTRAINT
ID	INT (11)	NOT NULL, AUTO INCREMENT, **PRIMARY KEY**
R_ID	INT (11)	NOT NULL, **FOREIGN KEY**
FIRST_NAME	VARCHAR (35)	NOT NULL
LAST NAME	VARCHAR (50)	NOT NULL

QUERIES

1. **Users-** User creates an account-
   User named Samwell Tarly wants to create an account on Friendspost.

```
INSERT INTO REGISTRATION(FIRST_NAME,LAST_NAME,GENDER,DOB,EMAIL_ID,PASSWORD)
VALUES ('SAMWELL',’TARLY’,'MALE','1995/06/11','starly@got.com','KUIET');
```

Before insert-

```
mysql> select * from REGISTRATION;
+----+------------+-----------+--------+------------+--------------------+----------+
| ID | FIRST_NAME | LAST_NAME | GENDER | DOB        | EMAIL_ID           | PASSWORD |
+----+------------+-----------+--------+------------+--------------------+----------+
|  1 | OLENNA     | TYRELL    | FEMALE | 1957-12-11 | otyrell@got.com    | WERDT    |
|  2 | SANSA      | STARK     | FEMALE | 1994-05-24 | sstark@got.com     | NGTSB    |
|  3 | THEON      | GREYJOY   | MALE   | 1993-07-20 | tgreyjoy@got.com   | DHSUT    |
|  4 | JOHN       | SNOW      | MALE   | 1992-10-18 | jsnow@got.com      | DJDSG    |
|  5 | DAENERYS   | TARGARYEN | FEMALE | 1991-11-15 | dtargaryen@got.com | DFGTR    |
|  6 | ROB        | STARK     | MALE   | 1988-04-07 | rstark@got.com     | VDTHK    |
|  7 | CERSIE     | LANNISTER | FEMALE | 1967-09-19 | clannister@got.com | XDFGT    |
|  8 | RENLEY     | BARATHEON | MALE   | 1962-03-31 | rbaratheon@got.com | ASTRD    |
|  9 | KHAL       | DROGO     | MALE   | 1989-02-14 | kdrogo@got.com     | NVAXM    |
| 10 | OBERON     | MARTELL   | MALE   | 1977-01-01 | omartell@got.com   | MQPLZ    |
+----+------------+-----------+--------+------------+--------------------+----------+
10 rows in set (0.00 sec)
```

After Insert-

```
mysql> INSERT INTO REGISTRATION(FIRST_NAME,LAST_NAME,GENDER,DOB,EMAIL_ID,PASSWORD) VALUES ('SAMWELL','TARLY','MALE','1995/06/11','starly@got.com','KUIET');
Query OK, 1 row affected (0.00 sec)

mysql> select * from REGISTRATION;
+----+------------+-----------+--------+------------+--------------------+----------+
| ID | FIRST_NAME | LAST_NAME | GENDER | DOB        | EMAIL_ID           | PASSWORD |
+----+------------+-----------+--------+------------+--------------------+----------+
|  1 | OLENNA     | TYRELL    | FEMALE | 1957-12-11 | otyrell@got.com    | WERDT    |
|  2 | SANSA      | STARK     | FEMALE | 1994-05-24 | sstark@got.com     | NGTSB    |
|  3 | THEON      | GREYJOY   | MALE   | 1993-07-20 | tgreyjoy@got.com   | DHSUT    |
|  4 | JOHN       | SNOW      | MALE   | 1992-10-18 | jsnow@got.com      | DJDSG    |
|  5 | DAENERYS   | TARGARYEN | FEMALE | 1991-11-15 | dtargaryen@got.com | DFGTR    |
|  6 | ROB        | STARK     | MALE   | 1988-04-07 | rstark@got.com     | VDTHK    |
|  7 | CERSIE     | LANNISTER | FEMALE | 1967-09-19 | clannister@got.com | XDFGT    |
|  8 | RENLEY     | BARATHEON | MALE   | 1962-03-31 | rbaratheon@got.com | ASTRD    |
|  9 | KHAL       | DROGO     | MALE   | 1989-02-14 | kdrogo@got.com     | NVAXM    |
| 10 | OBERON     | MARTELL   | MALE   | 1977-01-01 | omartell@got.com   | MQPLZ    |
| 12 | SAMWELL    | TARLY     | MALE   | 1995-06-11 | starly@got.com     | KUIET    |
+----+------------+-----------+--------+------------+--------------------+----------+
11 rows in set (0.00 sec)
```

2. User updates their profile details-User Samwell Tarly updates his profile details but decides to not give himself a title as it is optional.

INSERT INTO PROFILE(R_ID,USER_NAME,DOB,AGE,GENDER,TITLE,EMAIL_ID,COUNTRY) VALUES (11,'SAMMY T','1995/06/11',22,'MALE',' ','styrell@got.com','UNITED KINGDOM');

```
mysql> SELECT * FROM PROFILE;
+----+------+-------------+------------+-----+--------+-------------------------+-------------------+---------------+
| ID | R_ID | USER_NAME   | DOB        | AGE | GENDER | TITLE                   | EMAIL_ID          | COUNTRY       |
+----+------+-------------+------------+-----+--------+-------------------------+-------------------+---------------+
|  1 |    1 | IMMORTAL OLY | 1957-12-11 |  60 | FEMALE | QUEEN OF THE HIGHGARDENS | otyrell@got.com   | AFGHANISTHAN  |
|  2 |    2 | LADY SANSA  | 1994-05-24 |  23 | FEMALE | QUEEN IN THE NORTH      | sstark@got.com    | UNITED KINGDOM |
|  3 |    3 | T GREY      | 1993-07-20 |  24 | MALE   | THE IRON BORN           | tgreyjoy@got.com  | NEW ZEALAND   |
|  4 |    4 | SNOW FLAKE  | 1992-10-18 |  25 | MALE   | YAY I AM A TARGARYEN    | jsnow@got.com     | UNITED KINGDOM |
|  5 |    5 | DENNY T     | 1991-11-15 |  26 | FEMALE | MOTHER OF DRAGONS       | dtargaryen@got.com | SYRIA        |
|  6 |    6 | ROBSTAR     | 1988-04-07 |  29 | MALE   | KING IN THE NORTH       | rstark@got.com    | UNITED KINGDOM |
|  7 |    7 | QUEEN CERSIE | 1967-09-19 |  50 | FEMALE | QUEEN MOTHER            | clannister@got.com | USA          |
|  8 |    8 | RENLEYBAR   | 1962-03-31 |  55 | MALE   | LORD OF STORMS END      | rbaratheon@got.com | USA          |
|  9 |    9 | KHALASAR    | 1989-02-14 |  28 | MALE   | THE GREAT RIDER         | kdrogo@got.com    | AFRICA        |
| 10 |   10 | MARTELLO    | 1977-01-01 |  30 | MALE   | FATHER OF THE SNAKES    | omartell@got.com  | UKRAINE       |
| 11 |   11 | SAMMY T     | 1995-06-11 |  22 | MALE   |                         | styrell@got.com   | UNITED KINGDOM |
+----+------+-------------+------------+-----+--------+-------------------------+-------------------+---------------+
11 rows in set (0.00 sec)
```

3. User uploads a Profile picture.

INSERT INTO DISPLAYPICTURE(R_ID,USER_NAME,IMAGE,CREATED) VALUES(11, 'SAMMY T ','66.jpg','2017-11-11 00:00:00');

Before Insert-

```
mysql> SELECT * FROM DISPLAYPICTURE;
+----+------+-------------+--------+------------+
| ID | R_ID | USER_NAME   | IMAGE  | CREATED    |
+----+------+-------------+--------+------------+
|  1 |    1 | IMMORTAL OLY | 20.jpg | 2015-11-01 |
|  2 |    2 | LADY SANSA  | 12.jpg | 2017-11-01 |
|  3 |    3 | T GREY      | 87.jpg | 2016-11-01 |
|  4 |    4 | SNOW FLAKE  | 34.jpg | 2017-11-01 |
|  5 |    5 | DENNY T     | 21.jpg | 2017-11-01 |
|  6 |    6 | ROBSTAR     | 49.JPG | 2017-11-01 |
|  7 |    7 | QUEEN CERSIE | 25.jpg | 2015-11-01 |
|  8 |    8 | RENLEYBAR   | 87.JPG | 2017-11-01 |
|  9 |    9 | KHALASAR    | 22.jpg | 2014-11-01 |
| 10 |   10 | MARTELLO    | 15.jpg | 2017-11-01 |
```

After Insert-

```
mysql> SELECT * FROM DISPLAYPICTURE;
+----+------+--------------+--------+------------+
| ID | R_ID | USER_NAME    | IMAGE  | CREATED    |
+----+------+--------------+--------+------------+
|  1 |    1 | IMMORTAL OLY | 20.jpg | 2015-11-01 |
|  2 |    2 | LADY SANSA   | 12.jpg | 2017-11-01 |
|  3 |    3 | T GREY       | 87.jpg | 2016-11-01 |
|  4 |    4 | SNOW FLAKE   | 34.jpg | 2017-11-01 |
|  5 |    5 | DENNY T      | 21.jpg | 2017-11-01 |
|  6 |    6 | ROBSTAR      | 49.JPG | 2017-11-01 |
|  7 |    7 | QUEEN CERSIE | 25.jpg | 2015-11-01 |
|  8 |    8 | RENLEYBAR    | 87.JPG | 2017-11-01 |
|  9 |    9 | KHALASAR     | 22.jpg | 2014-11-01 |
| 10 |   10 | MARTELLO     | 15.jpg | 2017-11-01 |
| 11 |   11 | SAMMY T      | 66.jpg | 2017-11-11 |
+----+------+--------------+--------+------------+
```

4.User uploads a CoverPicture.

INSERT INTO COVERPICTURE(R_ID,USER_NAME,IMAGE,CREATED) VALUES(11, 'SAMMY T ','99.jpg','2017-11-11 00:00:00');

```
+----+------+--------------+--------+------------+
| ID | R_ID | USER_NAME    | IMAGE  | CREATED    |
+----+------+--------------+--------+------------+
|  1 |    1 | IMMORTAL OLY | 11jpg  | 2015-11-01 |
|  2 |    2 | LADY SANSA   | 32.jpg | 2017-11-01 |
|  3 |    3 | T GREY       | 17.jpg | 2017-11-01 |
|  4 |    4 | SNOW FLAKE   | 14.jpg | 2016-11-01 |
|  5 |    5 | DENNY T      | 29.jpg | 2017-11-01 |
|  6 |    6 | ROBSTAR      | 30.jpg | 2017-11-01 |
|  7 |    7 | QUEEN CERSIE | 44.JPG | 2015-11-01 |
|  8 |    8 | RENLEYBAR    | 43.jpg | 2017-11-01 |
|  9 |    9 | KHALASAR     | 27.jpg | 2016-11-01 |
| 10 |   10 | MARTELLO     | 10.jpg | 2017-11-01 |
| 11 |   11 | SAMMY T      | 99.jpg | 2017-11-11 |
+----+------+--------------+--------+------------+
11 rows in set (0.00 sec)
```

5.User makes changes to their profile- User John Snow wants to update his Title.

```
UPDATE PROFILE SET TITLE = 'FIRE AND ICE' WHERE USER_NAME = 'SNOW FLAKE';
```

Before Update-

```
mysql> SELECT * FROM PROFILE;
+----+------+-------------+-----------+-----+--------+-----------------------+------------------+--------------+
| ID | R_ID | USER_NAME   | DOB       | AGE | GENDER | TITLE                 | EMAIL_ID         | COUNTRY      |
+----+------+-------------+-----------+-----+--------+-----------------------+------------------+--------------+
|  1 |    1 | IMMORTAL OLY | 1957-12-11 |  60 | FEMALE | QUEEN OF THE HIGHGARDENS | otyrell@got.com  | AFGHANISTHAN |
|  2 |    2 | LADY SANSA  | 1994-05-24 |  23 | FEMALE | QUEEN IN THE NORTH    | sstark@got.com   | UNITED KINGDOM |
|  3 |    3 | T GREY      | 1993-07-20 |  24 | MALE   | THE IRON BORN         | tgreyjoy@got.com | NEW ZEALAND  |
|  4 |    4 | SNOW FLAKE  | 1992-10-18 |  25 | MALE   | YAY I AM A TARGARYEN  | jsnow@got.com    | UNITED KINGDOM |
|  5 |    5 | DENNY T     | 1991-11-15 |  26 | FEMALE | MOTHER OF DRAGONS     | dtargaryen@got.com | SYRIA      |
```

After Update-

```
mysql> SELECT * FROM PROFILE;
+----+------+-------------+-----------+-----+--------+-----------------------+------------------+--------------+
| ID | R_ID | USER_NAME   | DOB       | AGE | GENDER | TITLE                 | EMAIL_ID         | COUNTRY      |
+----+------+-------------+-----------+-----+--------+-----------------------+------------------+--------------+
|  1 |    1 | IMMORTAL OLY | 1957-12-11 |  60 | FEMALE | QUEEN OF THE HIGHGARDENS | otyrell@got.com  | AFGHANISTHAN |
|  2 |    2 | LADY SANSA  | 1994-05-24 |  23 | FEMALE | QUEEN IN THE NORTH    | sstark@got.com   | UNITED KINGDOM |
|  3 |    3 | T GREY      | 1993-07-20 |  24 | MALE   | THE IRON BORN         | tgreyjoy@got.com | NEW ZEALAND  |
|  4 |    4 | SNOW FLAKE  | 1992-10-18 |  25 | MALE   | FIRE AND ICE          | jsnow@got.com    | UNITED KINGDOM |
|  5 |    5 | DENNY T     | 1991-11-15 |  26 | FEMALE | MOTHER OF DRAGONS     | dtargaryen@got.com | SYRIA      |
|  6 |    6 | ROBSTAR     | 1988-04-07 |  29 | MALE   | KING IN THE NORTH     | rstark@got.com   | UNITED KINGDOM |
|  7 |    7 | QUEEN CERSIE | 1967-09-19 |  50 | FEMALE | QUEEN MOTHER          | clannister@got.com | USA        |
|  8 |    8 | RENLEYBAR   | 1962-03-31 |  55 | MALE   | LORD OF STORMS END    | rbaratheon@got.com | USA        |
|  9 |    9 | KHALASAR    | 1989-02-14 |  28 | MALE   | THE GREAT RIDER       | kdrogo@got.com   | AFRICA       |
| 10 |   10 | MARTELLO    | 1977-01-01 |  30 | MALE   | FATHER OF THE SNAKES  | omartell@got.com | UKRAINE      |
| 11 |   11 | SAMMY T     | 1995-06-11 |  22 | MALE   |                       | styrell@got.com  | UNITED KINGDOM |
+----+------+-------------+-----------+-----+--------+-----------------------+------------------+--------------+
11 rows in set (0.00 sec)
```

6. **ADMINISTRATOR**- The administrator can perform the following functions-

Fetching user registration and profile table details as a single table

```
SELECT
REGISTRATION.ID,REGISTRATION.FIRST_NAME,REGISTRATION.LAST_NAME,REGISTRATION.GEN
DER,REGISTRATION.EMAIL_ID,PROFILE.USER_NAME,PROFILE.DOB,PROFILE.TITLE,PROFILE.COU
NTRY FROM REGISTRATION LEFT JOIN PROFILE ON REGISTRATION.ID = PROFILE.R_ID;
```

ID	FIRST_NAME	LAST_NAME	GENDER	EMAIL_ID	USER_NAME	DOB	TITLE	COUNTRY
1	OLENNA	TYRELL	FEMALE	otyrell@got.com	IMMORTAL OLY	1957-12-11	QUEEN OF THE HIGHGARDENS	AFGHANISTHAN
2	SANSA	STARK	FEMALE	sstark@got.com	LADY SANSA	1994-05-24	QUEEN IN THE NORTH	UNITED KINGDOM
3	THEON	GREYJOY	MALE	tgreyjoy@got.com	T GREY	1993-07-20	THE IRON BORN	NEW ZEALAND
4	JOHN	SNOW	MALE	jsnow@got.com	SNOW FLAKE	1992-10-18	FIRE AND ICE	UNITED KINGDOM
5	DAENERYS	TARGARYEN	FEMALE	dtargaryen@got.com	DENNY T	1991-11-15	MOTHER OF DRAGONS	SYRIA
6	ROB	STARK	MALE	rstark@got.com	ROBSTAR	1988-04-07	KING IN THE NORTH	UNITED KINGDOM
7	CERSIE	LANNISTER	FEMALE	clannister@got.com	QUEEN CERSIE	1967-09-19	QUEEN MOTHER	USA
8	RENLEY	BARATHEON	MALE	rbaratheon@got.com	RENLEYBAR	1962-03-31	LORD OF STORMS END	USA
9	KHAL	DROGO	MALE	kdrogo@got.com	KHALASAR	1989-02-14	THE GREAT RIDER	AFRICA
10	OBERON	MARTELL	MALE	omartell@got.com	MARTELLO	1977-01-01	FATHER OF THE SNAKES	UKRAINE

7. To see which user's profile does not have a title -

```
SELECT
REGISTRATION.ID,REGISTRATION.FIRST_NAME,REGISTRATION.LAST_NAME,PROFILE.TITLE
FROM REGISTRATION LEFT JOIN PROFILE ON REGISTRATION.ID = PROFILE.R_ID
WHERE PROFILE.TITLE IS NULL;
```

```
+----+------------+-----------+-------+
| ID | FIRST_NAME | LAST_NAME | TITLE |
+----+------------+-----------+-------+
| 12 | SAMWELL    | TARLY     |       |
+----+------------+-----------+-------+
1 row in set (0.00 sec)

mysql>
```

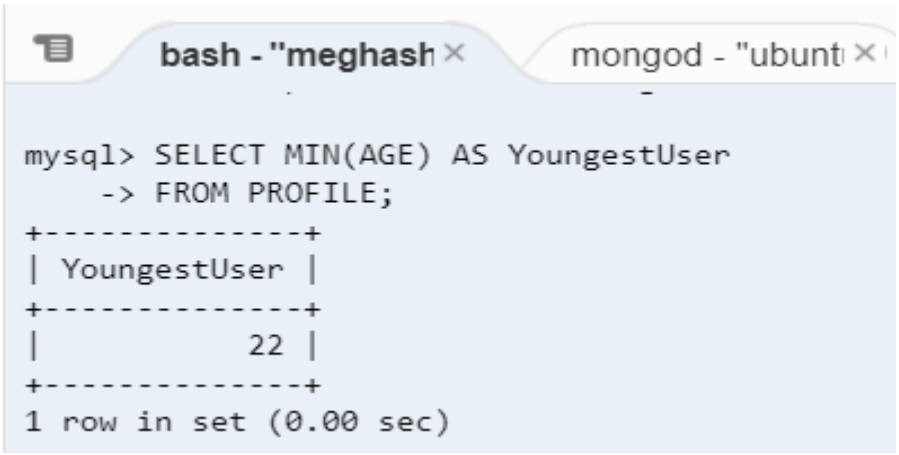
8. To display users in ascending order of their age-

```
SELECT
REGISTRATION.ID,REGISTRATION.FIRST_NAME,REGISTRATION.LAST_NAME,PROFILE.AGE
FROM REGISTRATION LEFT JOIN PROFILE ON REGISTRATION.ID = PROFILE.R_ID ORDER BY
PROFILE.AGE ASC;
```

```
+----+------------+-----------+------+
| ID | FIRST_NAME | LAST_NAME | AGE  |
+----+------------+-----------+------+
|  2 | SANSA      | STARK     |   23 |
|  3 | THEON      | GREYJOY   |   24 |
|  4 | JOHN       | SNOW      |   25 |
|  5 | DAENERYS   | TARGARYEN |   26 |
|  9 | KHAL       | DROGO     |   28 |
|  6 | ROB        | STARK     |   29 |
| 10 | OBERON     | MARTELL   |   30 |
|  7 | CERSIE     | LANNISTER |   50 |
|  8 | RENLEY     | BARATHEON |   55 |
|  1 | OLENNA     | TYRELL    |   60 |
+----+------------+-----------+------+
10 rows in set (0.00 sec)
```

9. To find out the minimum age of the users on friendspost-

```
SELECT MIN(AGE) AS YoungestUser FROM PROFILE;
```



10. To find out the maximum age of the users on friendspost-

SELECT MAX(AGE) AS YoungestUser FROM PROFILE;

```
+-------------+
| OldestUser |
+-------------+
|          60 |
+-------------+
1 row in set (0.00 sec)

mysql>
```

Immediate (Java × bash

```
meghasharma2:~/workspace $
```

11. To fetch male users of friendspost–

SELECT REGISTRATION.ID,REGISTRATION.FIRST_NAME, REGISTRATION.LAST_NAME,PROFILE.GENDER FROM REGISTRATION LEFT JOIN PROFILE ON REGISTRATION.ID = PROFILE.R_ID WHERE REGISTRATION.GENDER = 'MALE';

```
+----+------------+-----------+--------+
| ID | FIRST_NAME | LAST_NAME | GENDER |
+----+------------+-----------+--------+
|  3 | THEON      | GREYJOY   | MALE   |
|  4 | JOHN       | SNOW      | MALE   |
|  6 | ROB        | STARK     | MALE   |
|  8 | RENLEY     | BARATHEON | MALE   |
|  9 | KHAL       | DROGO     | MALE   |
| 10 | OBERON     | MARTELL   | MALE   |
+----+------------+-----------+--------+
6 rows in set (0.00 sec)

mysql>
```

bash - "ubuntu@ × bash - "meghash ×

```
meghasharma2:~/workspace $
```

12. To fetch female users of friendspost–

SELECT REGISTRATION.ID,REGISTRATION.FIRST_NAME, REGISTRATION.LAST_NAME,PROFILE.GENDER FROM REGISTRATION LEFT JOIN PROFILE ON REGISTRATION.ID = PROFILE.R_ID WHERE REGISTRATION.GENDER = 'FEMALE';

```
+----+------------+-----------+--------+
| ID | FIRST_NAME | LAST_NAME | GENDER |
+----+------------+-----------+--------+
|  1 | OLENNA     | TYRELL    | FEMALE |
|  2 | SANSA      | STARK     | FEMALE |
|  5 | DAENERYS   | TARGARYEN | FEMALE |
|  7 | CERSIE     | LANNISTER | FEMALE |
+----+------------+-----------+--------+
4 rows in set (0.00 sec)

mysql>
```

bash - "ubuntu@ × bash - "meghash ×

```
meghasharma2:~/workspace $
```

13. User deactivates their account

DELETE FROM REGISTRATION WHERE ID = 12;

Before-

```
mysql> select * from REGISTRATION;
+----+------------+-----------+--------+------------+--------------------+----------+
| ID | FIRST_NAME | LAST_NAME | GENDER | DOB        | EMAIL_ID           | PASSWORD |
+----+------------+-----------+--------+------------+--------------------+----------+
|  1 | OLENNA     | TYRELL    | FEMALE | 1957-12-11 | otyrell@got.com    | WERDT    |
|  2 | SANSA      | STARK     | FEMALE | 1994-05-24 | sstark@got.com     | NGTSB    |
|  3 | THEON      | GREYJOY   | MALE   | 1993-07-20 | tgreyjoy@got.com   | DHSUT    |
|  4 | JOHN       | SNOW      | MALE   | 1992-10-18 | jsnow@got.com      | DJDSG    |
|  5 | DAENERYS   | TARGARYEN | FEMALE | 1991-11-15 | dtargaryen@got.com | DFGTR    |
|  6 | ROB        | STARK     | MALE   | 1988-04-07 | rstark@got.com     | VDTHK    |
|  7 | CERSIE     | LANNISTER | FEMALE | 1967-09-19 | clannister@got.com | XDFGT    |
|  8 | RENLEY     | BARATHEON | MALE   | 1962-03-31 | rbaratheon@got.com | ASTRD    |
|  9 | KHAL       | DROGO     | MALE   | 1989-02-14 | kdrogo@got.com     | NVAXM    |
| 10 | OBERON     | MARTELL   | MALE   | 1977-01-01 | omartell@got.com   | MQPLZ    |
| 12 | SAMWELL    | TARLY     | MALE   | 1995-06-11 | starly@got.com     | KUIET    |
+----+------------+-----------+--------+------------+--------------------+----------+
11 rows in set (0.00 sec)
```

After deletion-

```
mysql> select * from REGISTRATION;
+----+------------+-----------+--------+------------+--------------------+----------+
| ID | FIRST_NAME | LAST_NAME | GENDER | DOB        | EMAIL_ID           | PASSWORD |
+----+------------+-----------+--------+------------+--------------------+----------+
|  1 | OLENNA     | TYRELL    | FEMALE | 1957-12-11 | otyrell@got.com    | WERDT    |
|  2 | SANSA      | STARK     | FEMALE | 1994-05-24 | sstark@got.com     | NGTSB    |
|  3 | THEON      | GREYJOY   | MALE   | 1993-07-20 | tgreyjoy@got.com   | DHSUT    |
|  4 | JOHN       | SNOW      | MALE   | 1992-10-18 | jsnow@got.com      | DJDSG    |
|  5 | DAENERYS   | TARGARYEN | FEMALE | 1991-11-15 | dtargaryen@got.com | DFGTR    |
|  6 | ROB        | STARK     | MALE   | 1988-04-07 | rstark@got.com     | VDTHK    |
|  7 | CERSIE     | LANNISTER | FEMALE | 1967-09-19 | clannister@got.com | XDFGT    |
|  8 | RENLEY     | BARATHEON | MALE   | 1962-03-31 | rbaratheon@got.com | ASTRD    |
|  9 | KHAL       | DROGO     | MALE   | 1989-02-14 | kdrogo@got.com     | NVAXM    |
| 10 | OBERON     | MARTELL   | MALE   | 1977-01-01 | omartell@got.com   | MQPLZ    |
+----+------------+-----------+--------+------------+--------------------+----------+
10 rows in set (0.00 sec)
```

14. The user who deactivated his account gets added to the Inactive users table

INSERT INTO INACTIVEUSERS(R_ID,FIRST_NAME,LAST_NAME) VALUES(12,'SAMWELL','TARLY');

Before user is inactive-

```
mysql> SELECT * FROM INACTIVEUSERS;
+----+------+------------+-----------+
| ID | R_ID | FIRST_NAME | LAST_NAME |
+----+------+------------+-----------+
|  1 |   13 | LORD       | BOLTON    |
|  2 |   14 | BENJAMIN   | FREY      |
|  3 |   15 | KATYLYN    | TULLY     |
+----+------+------------+-----------+
3 rows in set (0.00 sec)
```

After user is inactive-

```
mysql> SELECT * FROM INACTIVEUSERS;
+----+------+------------+-----------+
| ID | R_ID | FIRST_NAME | LAST_NAME |
+----+------+------------+-----------+
|  1 |   13 | LORD       | BOLTON    |
|  2 |   14 | BENJAMIN   | FREY      |
|  3 |   15 | KATYLYN    | TULLY     |
|  4 |   12 | SAMWELL    | TARLY     |
+----+------+------------+-----------+
4 rows in set (0.00 sec)
```

View

To create a view to see which users updated their profile pictures in 2017

CREATE VIEW LATEST_DISPLAY_PICTURE_UPDATES ASSELECT USER_NAME,CREATED

FROM DISPLAYPICTURE

WHERE DISPLAYPICTURE.CREATED<= 2017/12/12 && DISPLAYPICTURE.CREATED>= 2017/01/01;

```
mysql> CREATE VIEW LATEST_DISPLAY_PICTURE_UPDATES AS
    -> SELECT USER_NAME,CREATED
    -> FROM DISPLAYPICTURE
    -> WHERE DISPLAYPICTURE.CREATED<= 2017/12/12 && DISPLAYPICTURE.CREATED>= 2017/01/01;
```

SELECT * FROM LATEST_DISPLAY_PICTURE_UPDATES;

```
+-------------+-------------+
| USER_NAME   | CREATED     |
+-------------+-------------+
| LADY SANSA  | 2017-11-01  |
| SNOW FLAKE  | 2017-11-01  |
| DENNY T     | 2017-11-01  |
| ROBSTAR     | 2017-11-01  |
| RENLEYBAR   | 2017-11-01  |
| MARTELLO    | 2017-11-01  |
+-------------+-------------+
6 rows in set (0.00 sec)
```

Trigger

A trigger was created on the registration table to limit the password limit to be 5 and above-

DELIMITER $$ CREATE TRIGGER 'InvalidPassword' BEFORE INSERT ON 'REGISTRATION' FOR EACH ROW BEGIN IF CHAR_LENGTH(PASSWORD)<5 THEN SIGNAL SQLSTATE '12345' SET MESSAGE_TEXT='Invalid password, should be 5 characters long' END IF; END $$ DELIMITER;

INSERT INTO REGISTRATION(FIRST_NAME,LAST_NAME,GENDER,EMAIL_ID,PASSWORD) VALUES('MEGHA','SHARMA','FEMALE','MSHARMA2@SCU.EDU','2323');

```
mysql> DELIMITER $$ CREATE TRIGGER 'InvalidPassword' BEFORE INSERT ON 'REGISTRATION' FOR EACH ROW BEGIN IF CHAR_LENGTH(PASSWORD)<5 THEN SIGNAL SQLSTATE '12345' SET MESSAGE_TEXT='In
valid password, should be 5 characters long' END IF; END $$ DELIMITER;
mysql> INSERT INTO REGISTRATION(FIRST_NAME,LAST_NAME,GENDER,EMAIL_ID,PASSWORD) VALUES('MEGHA','SHARMA','FEMALE','MSHARMA2@SCU.EDU','2323');
```

Stored Procedure

The concept of stored procedures was applied on the registration table to ensure that two users cannot register with the same email id.

```
CREATE PROCEDURE DISTINCT_USER_ACCOUNT
EMAIL_ID VARCHAR(30) NOT NULL UNIQUE
AS
BEGIN
IF (SELECT * FROM REGISTRATION WHERE EMAIL_ID = EMAIL_ID) = 0
BEGIN
INSERT INTO REGISTRATION(EMAIL_ID)
VALUES(EMAIL_ID)
PRINT ‘ACCOUNT CREATED. WELCOME TO FRIENDSPOST!’
END
ELSE
BEGIN
PRINT ‘ACCOUNT ALREADY EXISTS WITH THIS EMAIL ID!’
END
END;
```

MONGO DB

To view the User data in Mongo-

- db.friendspost.find().pretty()


```
{
 "_id" : ObjectId("59d0b77117ee2c9c"),

        "userName":"Snowflake",
        "userId":4,
        "userCountry" : "United Kingdom",
        "pagePosts" : [
{
        "myAlbums": [
   {
        "title": "Camping with the wildlings",
        "createdOn": 02/11/2017,
        "description": { "text": "This is my very first album on Friendspost. I went camping with the wildlings in winters."} ,
        "numberOfPhotos" : 300,
        "nmberOfVideos" : 15,
        "likes" : 335,
        "comments" : 50,
      },
{
        "title": "Battle of Castle Black",
        "createdOn": 09/21/2017,
        "description": { "text": "I am posting my second album on Friendspost which is about my battle at the wall."} ,
        "numberOfPhotos" : 1000,
        "nmberOfVideos" : 45,
        "likes" : 905,
        "comments" : 90,
      }
   ],

      "articlesPosted" : [
  {

  "articleName" : "The myth that is the Whitewalkers",
  "articleLength" : "10pages",
   "likes" : 200,
   "comments" : 120
  }
```


```
]
}
},

{
 "_id" : ObjectId("89d0b89117ee2c5g"),

        "userName":"Lady Sansa",
        "userId" :2,
        "userCountry" : "United Kingdom",
        "pagePosts" : [
 {
        "articlesPosted" : [
  {
      "articleName" : "Life in Winterfell",
      "articleLength" : "100pages"
       "likes" : 150,
       "comments" : 97,
      },
  {
      "articleName" : "Journey to Kingslanding",
      "articleLength" : "1000pages",
       "likes" : 90,
       "comments" : 12
      },
  {
      "articleName" : "When I became Queen in the North",
      "articleLength" : "60pages",
       "likes" : 123,
       "comments" : 12
      }
   ]
  },
{
      "urlPosted" : [
   {
        "urlName" : "old songs",
       "urlsource": "soundcloud.com"
   },
  {
```


bash - "meghash
mongod - "ubunt
mongo - "ubuntu
"urlsource": "youtube.com"
}
]
}
]
},
{
"_id" : ObjectId("45d0b89667ee1j3g"),
"userName":"Denny T",
"userId" :5,
"userCountry" : "Syria",
"pagePosts" : [
"myAlbums":
{
"title": "My reign as Khaleesi",
"createdOn": 03/23/2016,
"description": { "text": "My is my first album on Friendspost is about my rule as the Queen of the Dothraaki."} ,
"numberOfPhotos" : 2000,
"nmberOfVideos" : 505,
"likes" : 4575,
"comments" : 980,
},
{
"videosUploaded" : [
{
"videoName" : "My wedding to Khal Drogo",
"videoLength" : "1 hour"
},
{
"videoName" : "My Dragons",
"videoLength" : "2 hours"
}
]
}
}
]

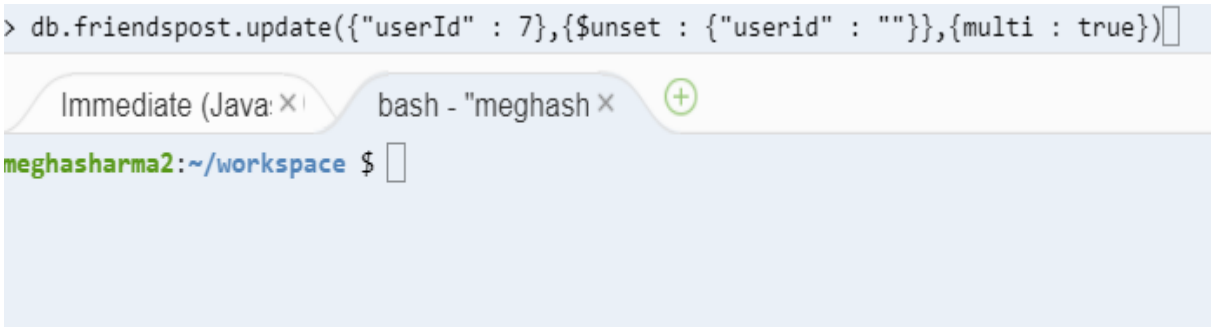
A new user registers on friendspost and uploads blog posts –

db.friendspost.insert({"_id" : ObjectId("54d0b76897ke1g3j"),"userName":"Queen Cersie","userId":7,"userCountry" : "USA","pagePosts" : [{ "myAlbums": {"title": "My home- Kingslanding","createdOn": 05/13/2017,"description": { "text": "A glipmspe of my palace"} ,"numberOfPhotos" : 600,"nmberOfVideos" : 85,"likes" : 175,"comments" : 80, }},{"articleName" : "A Queen's life","articleLength" : "700pages","likes" : 200,"comments" : 112}]})

db.friendspost.find({“userId: { $eq :7}}).pretty()

```
  "_id" : ObjectId("54d0b76897ke1g3j"),
"userName":"Queen Cersie",
"userId":7,
"userCountry" : "USA",
 "pagePosts" : [
{
 "myAlbums":
{
"title": "My home- Kingslanding",
"createdOn": 05/13/2017,
"description": { "text": "A glipmspe of my palace"} ,
"numberOfPhotos" : 600,
"nmberOfVideos" : 85,
"likes" : 175,
"comments" : 80,
 },{
      "articleName" : "A Queen's life",
      "articleLength" : "700pages",
       "likes" : 200,
       "comments" : 112
      }
}
]})
```

The user deactivates their account which results in the removal of their entire blog's data-



BONUS QUESTIONS

-Compare and improve the performance of Queries in SQL and MySQL.

-Using various techniques that you learned in the class improve the performance of SQL and MySQL queries.

The performance of the queries in SQL and Mongo can be improved by using the concepts of indexes. We can then compare the query return of using indexes or fetching the data using where or any other condition. We can then perform index range scans to measure the query return time.
Given below are the indexes created in SQL and Mongo DB respectively.

INDEXES IN SQL-One way to improve performance is to have a good index on relevant columns

CREATE UNIQUE INDEX USERS ON REGISTRATION (FIRST_NAME,LAST_NAME,EMAIL_ID);

CREATE UNIQUE INDEX USERS ON PROFILE (R_ID);

We can similarly create indexes in mongo on the collection-

CREATE INDEX –

```
> db.friendspost.createIndex({title:1})
{
        "createdCollectionAutomatically" : false,
        "numIndexesBefore" : 1,
        "numIndexesAfter" : 2,
        "ok" : 1
}
>
```

Immediate (Java × bash - "meghash ×

```
meghasharma2:~/workspace $
```

CREATING UNIQUE INDEX-

```
> db.friendspost.createIndex({isbn:1},{unique:true})
{
        "createdCollectionAutomatically" : false,
        "numIndexesBefore" : 2,
        "numIndexesAfter" : 3,
        "ok" : 1
}
>
```

Immediate (Java × bash - "meghash ×

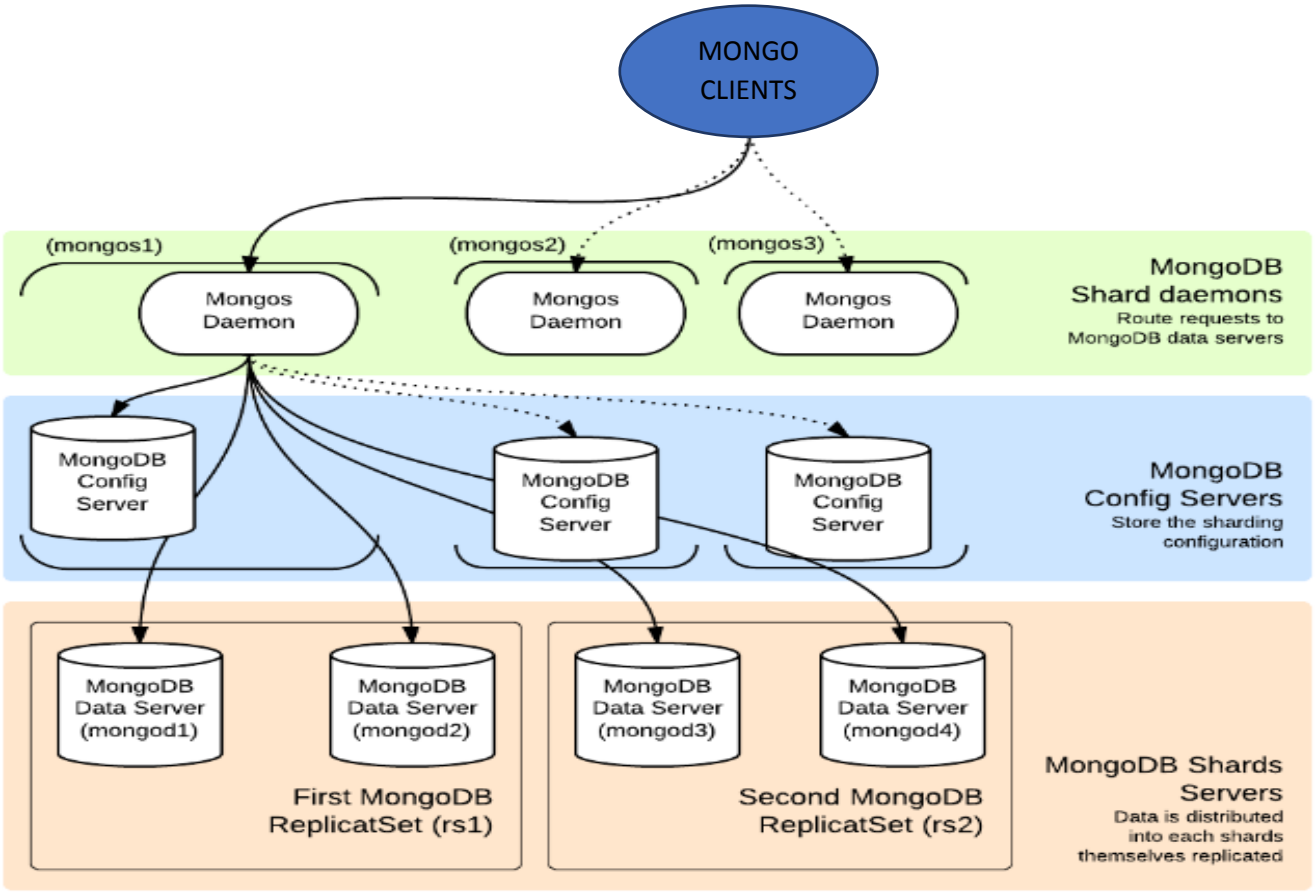
```
meghasharma2:~/workspace $
```

Show and demo benefits of Replica and Sharding to improve performance

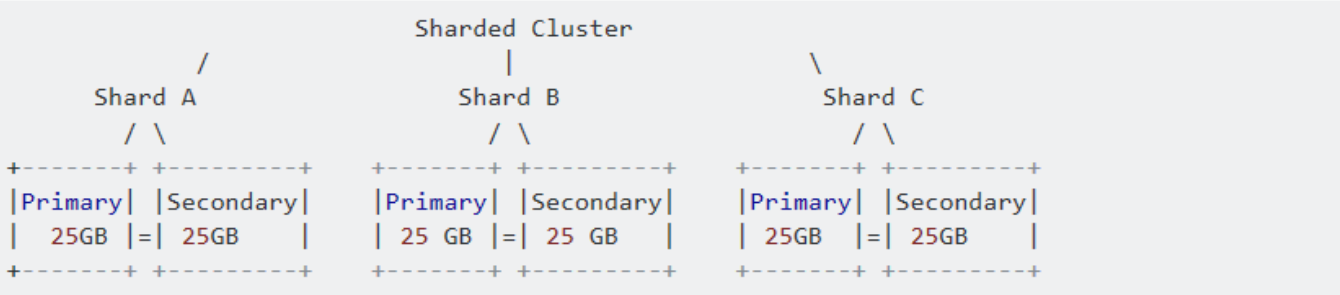
Creating aReplica-Setin MongoDB means creating multiple instances of MongoDB, where each of those instances mirror all the data of every other instance. A replica-set consists of one Master and one or more Slaves. Read-operations take place on any slave, so we can increase read-performance by adding more slaves to the replica-set. But write-operations always take place on the master of the replica-set and are then sent to the slaves, so writes won't get faster on adding more slaves.

Replica-sets offer fault-tolerance too. If one of the instances of the replica-set goes down, the other instances take over. When the master goes down, the slaves will elect a new master. So it is better for a productive deployment to always use MongoDB as a replica-set of at least three servers, two of them holding data, while the third one does not store any data but is a hands on available master available in case the other two masters go down.So replication is a mostly traditional master/slave setup, data is synced to backup members and if the primary fails one of them can take its place.

Sharding on the other hand is a method for storing data across multiple machines.Shards in MongoDB are replica sets with a router in front of them. So the application will be connected to the router, issue queries, and it will decide which shard to forward things to. The whole setup would look something like this-



For eg, from the uses' blog posts on friendspost, the current data size is 75GB, then replication will be used here and the data will be stored in 3 different servers- 75GB on Server-1, 75GB on server-2 and 75GB on server-3. So the data of 75GB will be split into 3 shards of 25GB each, for which at least 6 database servers organized in three replica-sets will be needed. Each replica-set consists of two servers who have the same 25GB of data as shown below-



It will also require servers for the arbiters of the three replica-sets as well as the mongos router and the config server for the cluster. The arbiters are very lightweight and are only needed when a replica-set member goes down, so they can usually share the same hardware with something else.

Sharding- Keeping theuser blog files in different folders

Replication- Syncing the collection to other drives

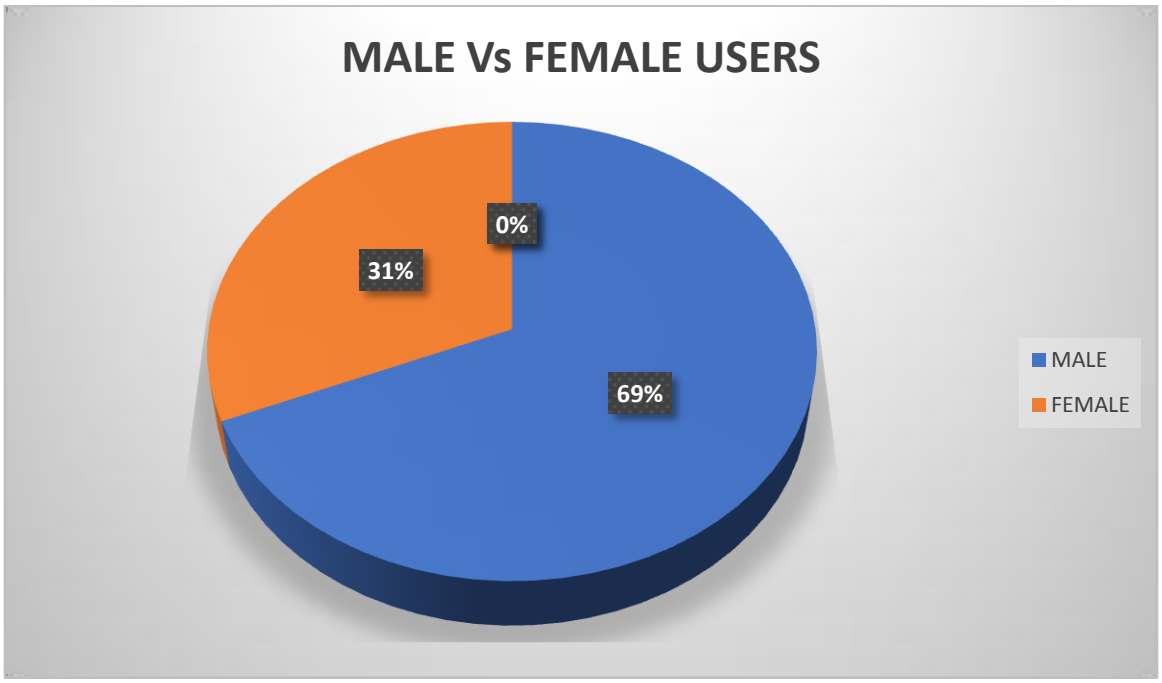
BUSINESS METRICS

1. Which gender group is more active on social media?

Query- SELECT COUNT(R_ID), GENDER FROM PROFILE GROUP BY GENDER;

```
mysql> SELECT COUNT(R_ID), GENDER FROM PROFILE GROUP BY GENDER;
+-------------+--------+
| COUNT(R_ID) | GENDER |
+-------------+--------+
|           4 | FEMALE |
|           7 | MALE   |
+-------------+--------+
2 rows in set (0.00 sec)

mysql>
```

```
bash - "ubuntu@ ×   bash - "meghash ×   Immediate ×
meghasharma2:~/workspace $
```



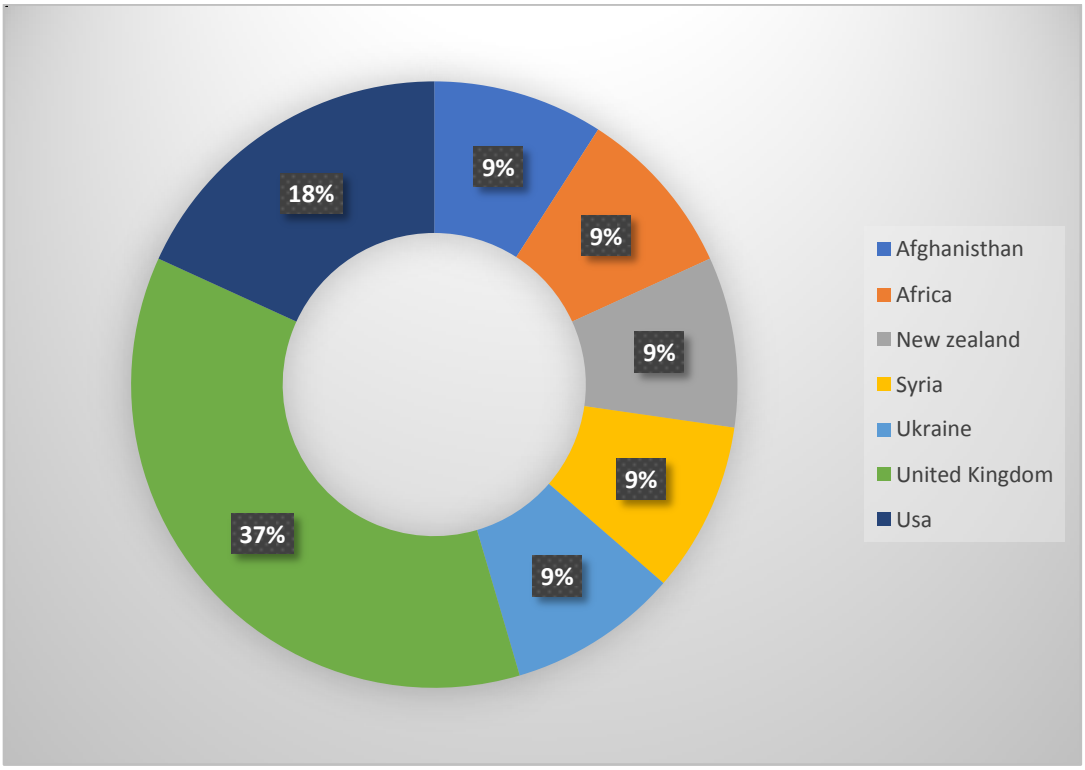
2. FRIENDSPOST COUNTRY BASIS USAGE AND GROWTH STATISTICS-

SELECT COUNT(R_ID), COUNTRY FROM PROFILEGROUP BY COUNTRY;

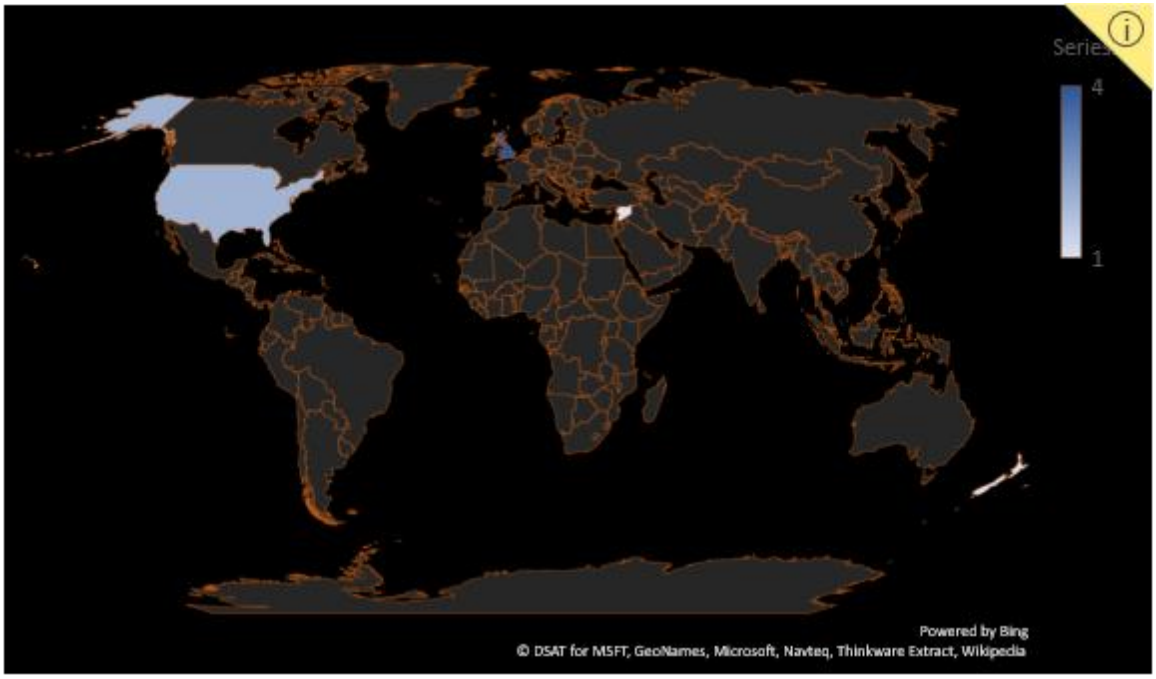
```
mysql> SELECT COUNT(R_ID), COUNTRY FROM PROFILE GROUP BY COUNTRY;
+-------------+----------------+
| COUNT(R_ID) | COUNTRY        |
+-------------+----------------+
|           1 | AFGHANISTHAN   |
|           1 | AFRICA         |
|           1 | NEW ZEALAND    |
|           1 | SYRIA          |
|           1 | UKRAINE        |
|           4 | UNITED KINGDOM |
|           2 | USA            |
+-------------+----------------+
7 rows in set (0.00 sec)

mysql>
```

bash - "ubuntu@ × bash - "meghash × Immediate ×

```
meghasharma2:~/workspace $
```



GEOGRAPHICAL VIEW OF THE RESULT-



3. WHICH AGE GROUP IS MORE SOCIAL MEDIA SAVY-

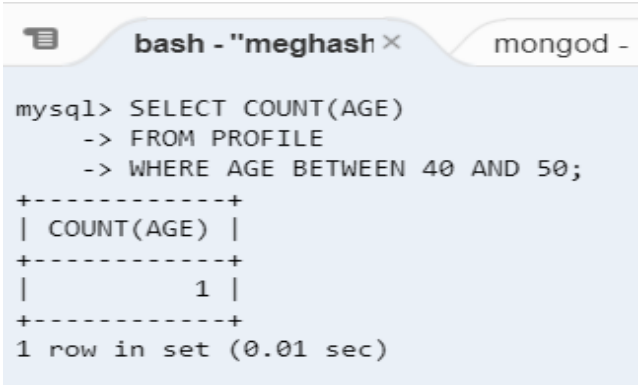
Query- SELECT COUNT(AGE) FROM PROFILE WHERE AGE BETWEEN 20 AND 30;

```
mysql> SELECT COUNT(AGE)
    -> FROM PROFILE
    -> WHERE AGE BETWEEN 20 AND 30;
+------------+
| COUNT(AGE) |
+------------+
|          8 |
+------------+
1 row in set (0.00 sec)
```

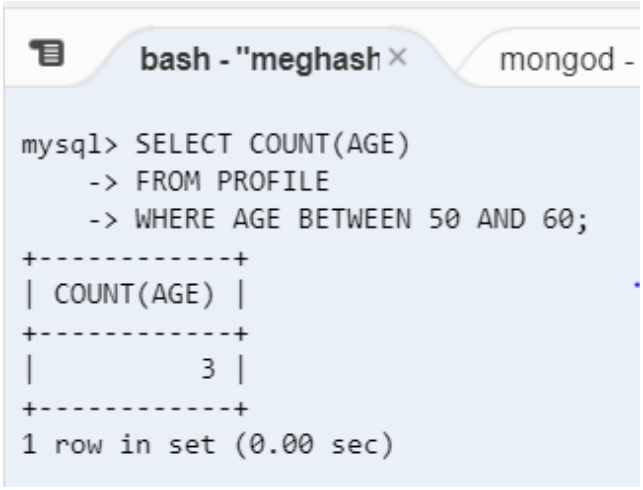
SELECT COUNT(AGE)
FROM PROFILE
WHERE AGE BETWEEN 30 AND 40;

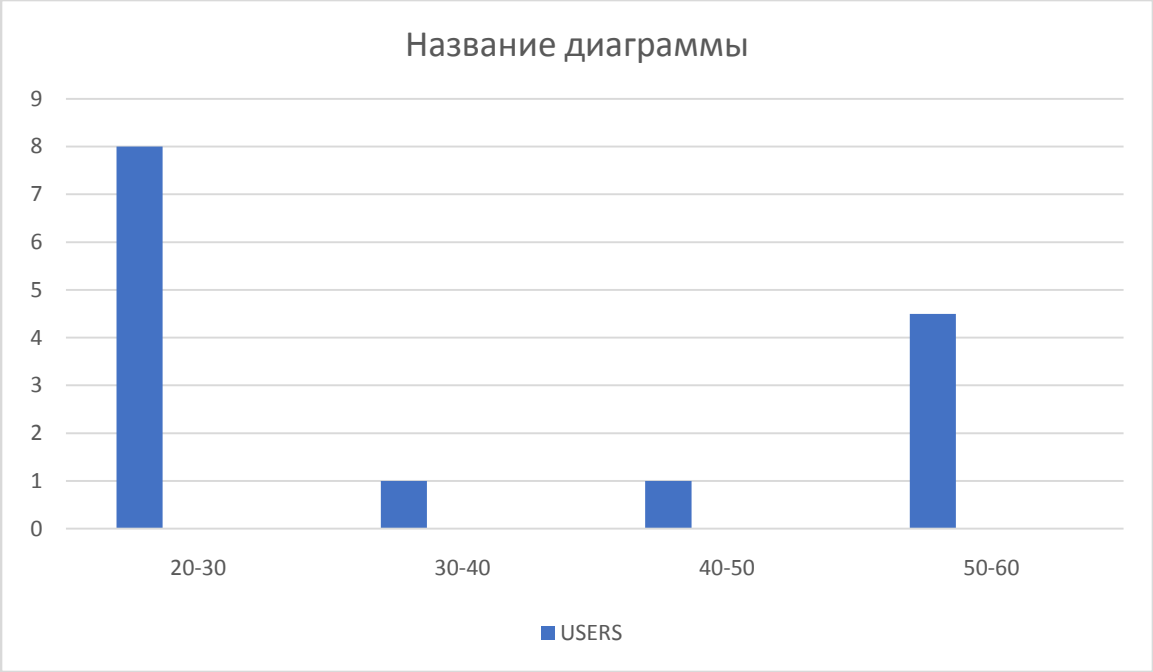
```
mysql> SELECT COUNT(AGE)
    -> FROM PROFILE
    -> WHERE AGE BETWEEN 30 AND 40;
+------------+
| COUNT(AGE) |
+------------+
|          1 |
+------------+
1 row in set (0.00 sec)
```

SELECT COUNT(AGE)FROM PROFILEWHERE AGE BETWEEN 40 AND 50;



SELECT COUNT(AGE)FROM PROFILEWHERE AGE BETWEEN 50 AND 60;




Название диаграммы
9
8
7
6
5
4
3
2
1
0
20-30
30-40
40-50
50-60
USERS

Project Summary

13.1 Summarize your experience with this exercise

- Working on a project all by yourself sounds intimidating at first, but while making this project, I felt more confident with each mistake I made and with each solution I found for that mistake. I strengthened my DBMS skills as making this project gave me more clarity on practical application and usage of SQL concepts like views, triggers and stored procedures.
- I have more clarity on how and when can the concepts of NoSQL be used to enhance application efficiency.
- This project made me think from a technical as well as a business perspective and gave me an understanding of both the domains.

13.2 What was the hardest part of this project?

The hardest part for me was querying the database and running into errors, then going back to improvise the database structure all over again. It was after multiple hits and trials that I was able to finalize my schema.

13.3 What problems did you run against in this project? How did you solve these problems?

- Designing the database was itself a complicated task, so drawing the UML diagrams and structure chart for mongo helped to map the right cardinalities for the entities.
- Eliminating redundancies and normalizing the database is time consuming, so doing the blog posts in denormalized mongo structure was helpful.

13.4 If you were to do this project again, what methodology would you follow?

- I would like to take up this project on a bigger level where I can will perform the task of introducing new features to the application to see and compare how much time does it take in SQL and NoSQL to implement those features.
- I would also like to enquire about some industry norms of database usage and plan the project accordingly.