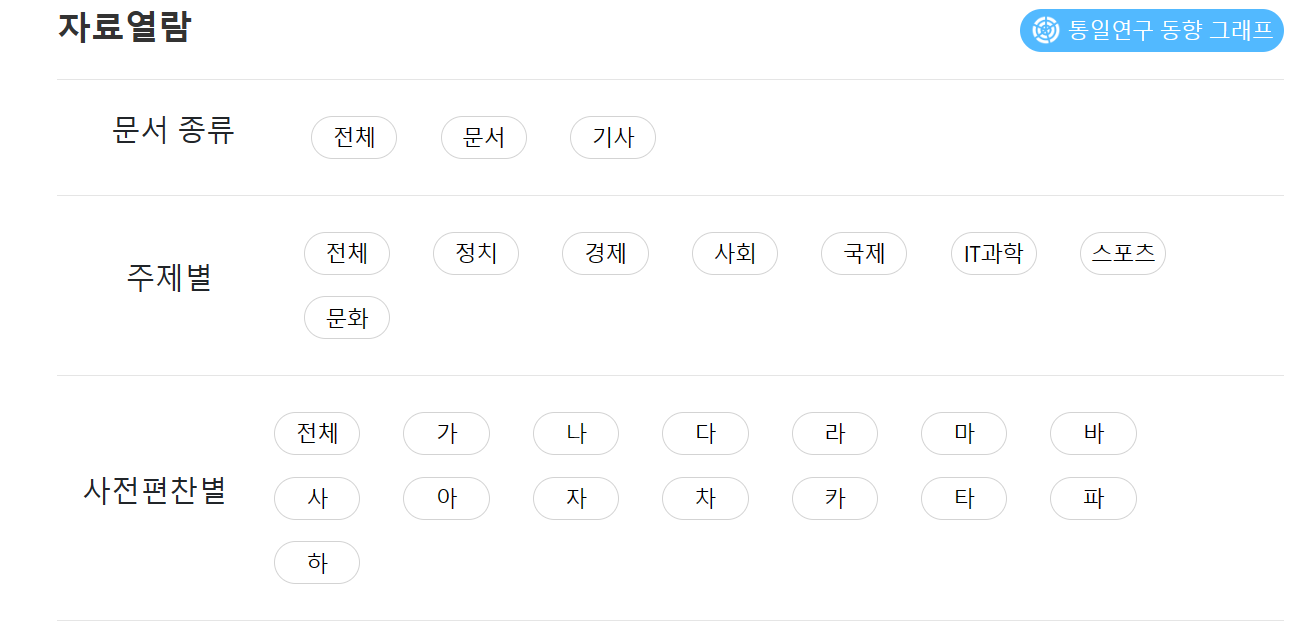
# **23-1 Database System Team Project1**

**21900156 김예준 / 22000796 함상훈 / 22100579 이진주**

**How to attack this problem?**

Our first step is to analyze the website, <https://kubic.handong.edu/>.



We are able to find attributes that allow us to find the document, such as doc\_type, title, post,title,first\_char, published\_institution.

We do a search through the search bar, and we get results like the one in the following image.

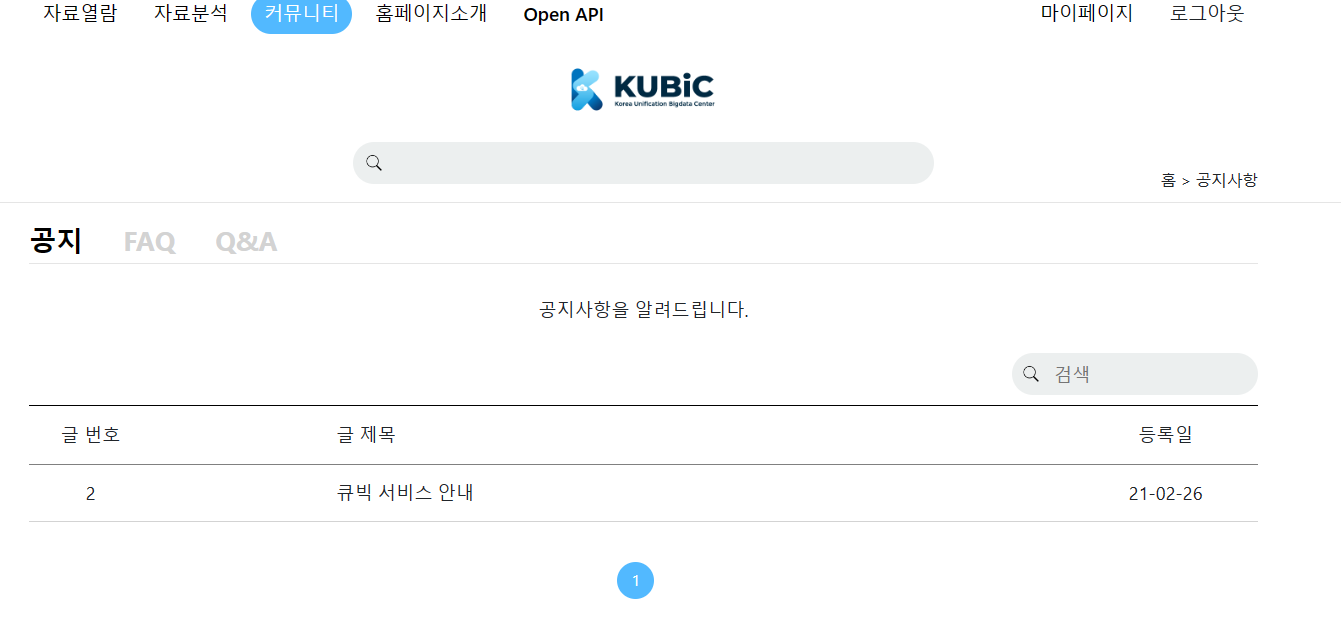


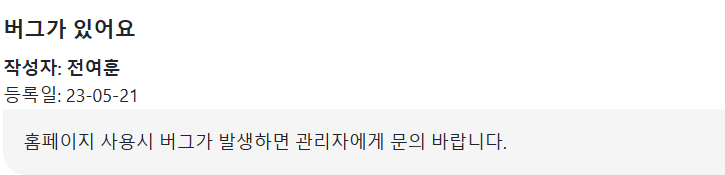
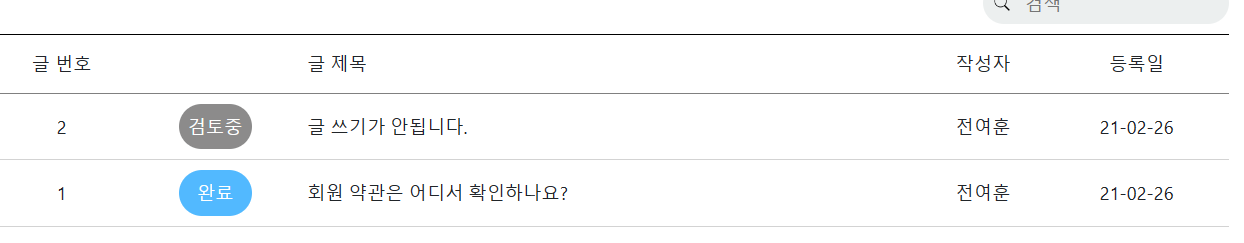
We were able to find attributes like post\_title, post\_date, post\_writer, and abstract that make up the documents.

And if we click on the documentation and look closely, we can find attributes like published\_institution, published\_institutuion\_url, original\_url, post\_body.When we sign up, it asks for the information such as occupation, institute, and email. The email also contains information about the name.

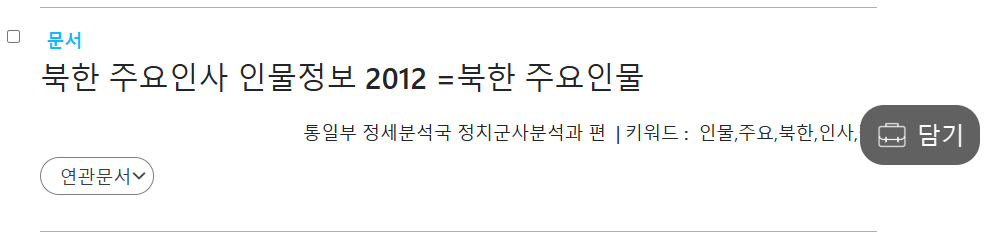
|  |  |
| --- | --- |
|  | |

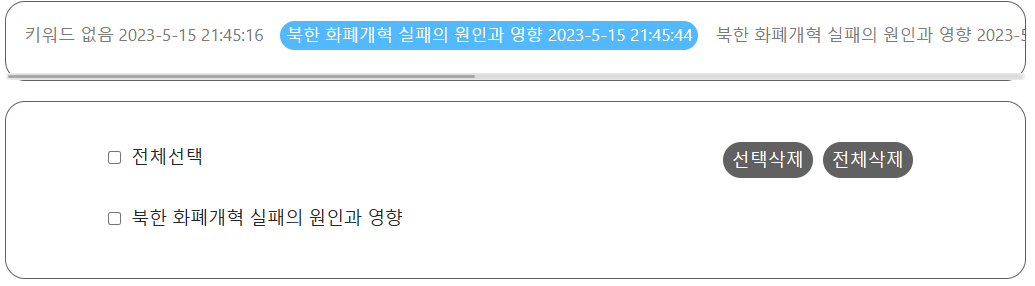
When we look at the Announcements, FAQs, and Q&As in the Community tab, we find the following properties: docId, category, title, content, userName, regDate and content.



ff

When we log in and search a document, we find the ability to save the document.



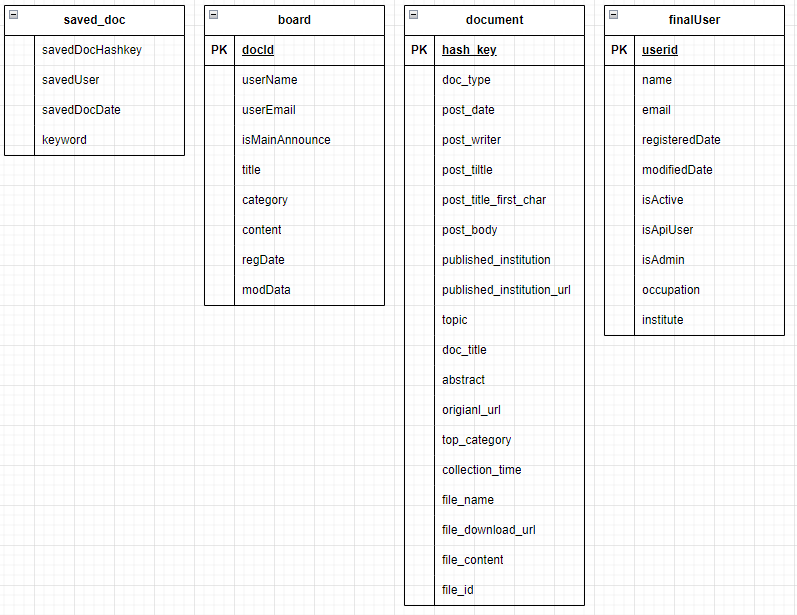


This shows us that to save the document use keyword, savedUser, and savedDocDate.

The second step is to find out if the attribute we haven't identified is associated with other attributes.After running the query and comparing the properties not found on the homepage.

Also, normalization principles were consulted to discover and reflect dependencies between columns.

According to the above steps, we are able to create the following table.



The next step is to find unnecessary duplicates.

We used various queries with ‘distinct’ to find duplicates like this.

| SELECT COUNT(DISTINCT [column name]) FROM [origin table]; |
| --- |

If the result of this kind of query is significantly less than the whole count, we made a new table for that column called ‘[column name]Mapping’.

The mapping tables consist of the de-duplicated target columns from the original table and the primary keys of smaller capacity (e.g., small int) assigned to each unique data of the columns . And the columns separated by the mapping from the origin table are replaced by the columns of the primary key of the mapping table. then, part of the size becomes like this example.

| origin table |  | origin table + mapping table |
| --- | --- | --- |
| 35000 \* varchar(255) | > | 35000 \* smallint + 100 \* (smallint + varchar(255)) |

We expect the mapping table to minimize the size of the overlapping parts, making the overall data smaller.

Finally, we adjust whole data types and sizes. in the row data, there were many columns defined with non-efficient domain type.

We changed the domain to the most appropriate data type, taking into account the maximum length of the data and the diversity of each data length in each column.

For example, for strings with large length variation within 255 characters, we chose the 'VARCHAR' type. But for strings where all the data is 22 characters long, we chose 'CHAR'. For strings longer than 255 characters, we chose the 'MEDIUM TEXT' type.

| hello  haha! i am JJ  wanna go home……  welcome to HGU!  null  .  . | VARCHAR(255) |
| --- | --- |
| 21800156  22000796  22100579  .  . | CHAR(8) |

And for numeric data types, we chose 'TINY INT', 'SMALLINT', etc. as appropriate, considering the maximum value. We were pleasantly surprised at how much space this process could save.

| 0~255 | TINY INT |
| --- | --- |
| 255~65535 | SMALL INT |
| 65535~4294967295 | INT |

This is the final result of our effort.

**DDL Query**

| ***document*** |
| --- |
| create table document as  select doc\_type, post\_date, post\_title, post\_body, hash\_key, doc\_title, abstract, original\_url, collection\_time, file\_name, file\_download\_url, file\_content, file\_id, post\_title\_first\_char, topic, top\_category, published\_institution, post\_writer from kubicdb  group by doc\_type, post\_date, post\_title, post\_body, hash\_key, doc\_title, abstract, original\_url, collection\_time, file\_name, file\_download\_url, file\_content, file\_id, post\_title\_first\_char, topic, top\_category, published\_institution, post\_writer;  alter table document add docType\_id tinyint;  alter table document add firstChar\_id tinyint unsigned;  alter table document add topic\_id tinyint;  alter table document add topCategory\_id tinyint unsigned;  alter table document add pubInst\_id tinyint;  alter table document add postWriter\_id smallint;  alter table document modify post\_data varchar(255);  alter table document modify post\_title varchar(255);  alter table document modify post\_body text;  alter table document modify doc\_title varchar(255);  alter table document modify original\_url text;  alter table document modify collection\_time char(22);  alter table document modify file\_name varchar(255);  alter table document modify file\_download\_url text;  alter table document modify file\_content text;  alter table document modify file\_id varchar(255);  #create a mapping table with them  alter table document drop doc\_type;  alter table document drop post\_title\_first\_char;  alter table document drop topic;  alter table document drop top\_category;  alter table document drop published\_institution;  alter table document drop post\_writer;  alter table document add PRIMARY KEY document (hash\_key);  alter table savedDoc add CONSTRAINT document\_PostfirstCharMapping\_firstChar\_id\_fk FOREIGN KEYS (firstChar\_id) REFERENCES PostfirstCharMapping(firstChar\_id);  alter table savedDoc add CONSTRAINT document\_topicMapping\_topic\_id\_fk FOREIGN KEYS (topic\_id) REFERENCES topicMapping(topic\_id);  alter table savedDoc add CONSTRAINT document\_topCategoryMapping\_topCategory\_id\_fk FOREIGN KEYS (topCategory\_id) REFERENCES topCategoryMapping(topCategory\_id);  alter table savedDoc add CONSTRAINT  document\_pubInstitutionMapping\_pubInst\_id\_fk FOREIGN KEYS (pubInst\_id) REFERENCES pubInstitutionMapping(pubInst\_id);  alter table savedDoc add CONSTRAINT  document\_postWriterMapping\_postWriter\_id\_fk FOREIGN KEYS (postWriter\_id) REFERENCES postWriterMapping(postWriter\_id); |

| ***savedDoc*** |
| --- |
| create table savedDoc as  select keyword, savedUser, savedDocDate, savedDocHashKey from kubicdb  group by keyword, savedUser, savedDocDate, savedDocHashKey;  alter table savedDoc add key\_id tinyint;  alter table savedDoc add eid tinyint;  alter table savedDoc add savedDocs\_id tinyint;  alter table savedDoc add PRIMARY KEY savedDoc (eid, key\_id, savedDocs\_id);  alter table savedDoc add CONSTRAINT savedDoc\_keyword\_mapping\_key\_id\_fk FOREIGN KEYS (key\_id) REFERENCES keyword\_mapping(key\_id);  alter table savedDoc add CONSTRAINT savedDoc\_finalUser\_eid\_fk FOREIGN KEYS (eid) REFERENCES finalUser(eid);  alter table savedDoc add CONSTRAINT savedDoc\_savedDocsMapping\_savedDocs\_id\_fk FOREIGN KEYS (savedDocs\_id) REFERENCES savedDocsMapping(savedDocs\_id);  #create a mapping table with them  alter table savedDoc drop keyword;  alter table savedDoc drop savedUser;  alter table savedDoc drop savedDocDate;  alter table savedDoc drop savedDocHashKey; |

| ***finalUser*** |
| --- |
| create table finalUser as  select distinct userId, name, email, institute, occupation, registeredDate, modifiedDate, isActive, isApiUser, isAdmin  from kubicdb  where userId is not null  group by userId, name, email, institute, occupation, registeredDate, modifiedDate, isActive, isApiUser, isAdmin;  alter table finalUser add eid tinyint;  alter table finalUser add occu\_id tinyint;  alter table finalUser add inst\_id tinyint;  alter table finalUser add PRIMARY KEY finalUser(eid);  alter table finalUser add CONSTRAINT finalUser\_occupationMapping\_occu\_id\_fk FOREIGN KEYS (occu\_id) REFERENCES occupationMapping(occu\_id);  alter table finalUser add CONSTRAINT finalUser\_instituteMapping\_inst\_id\_fk FOREIGN KEYS (inst\_id) REFERENCES instituteMapping(inst\_id);  #create a mapping table with them  alter table finalUser drop email;  alter table finalUser drop occupation;  alter table finalUser drop institute;  alter table finalUser modify userId char(24);  alter table finalUser modify name varchar(23);  alter table finalUser modify registeredDate bigint;  alter table finalUser modify modifiedDate bigint;  alter table finalUser modify isActive tinyint;  alter table finalUser modify isApiUser tinyint;  alter table finalUser modify isAdmin tinyint; |

| ***board*** |
| --- |
| create table board as  select title, content, userName, userEmail, isMainAnnounce, regDate, modDate, docId, category  from kubicdb  where docId is not null  group by title, content, userName, userEmail, userEmail, isMainAnnounce, regDate, modDate, docId, category;  alter table board add eid tinyint;  #create a mapping table with them  alter table board drop userName;  alter table board drop userEmail;  alter table board modify title varchar(255);  alter table board modify content varchar(255);  alter table board modify isMainAnnounce tinyint;  alter table board modify regDate bigint;  alter table board modify modDate bigint;  alter table board modify docId tinyint;  alter table board modify category varchar(255);  alter table board add PRIMARY KEY board(docId);  alter table board add CONSTRAINT board\_emailMapping\_eid\_fk FOREIGN KEYS (eid) REFERENCES emailMapping(eid); |

| ***doc\_type\_mapping*** |
| --- |
| create table doc\_type\_mapping(  docType\_id tinyint primary key auto\_increment,  doc\_type char(5)  ); |

| ***emailMapping*** |
| --- |
| create table emailMapping(  eid tinyint primary key auto\_increment,  email varchar(255)  ); |

| ***instituteMapping*** |
| --- |
| create table instituteMapping(  inst\_id tinyint primary key auto\_increment,  institute varchar(100)  ); |

| ***keyword\_mapping*** |
| --- |
| create table keyword\_mapping(  key\_id tinyint primary key auto\_increment,  keyword varchar(5)  ); |

| ***occupationMapping*** |
| --- |
| create table occupationMapping(  occu\_id tinyint primary key auto\_increment,  occupation varchar(255)  ); |

| ***PostfirstCharMapping*** |
| --- |
| create table PostfirstCharMapping(  firstChar\_id tinyint unsigned primary key auto\_increment,  post\_title\_first\_char(1)  ); |

| ***postWriterMapping*** |
| --- |
| create table postWriterMapping(  postWrite\_id smallint primary key auto\_increment,  post\_writer varchar(255)  ); |

| ***pubInstitutionMapping*** |
| --- |
| create table pubInstitutionMapping(  pubIinst\_id tinyintprimary key auto\_increment,  published\_institution varchar(255),  published\_institution\_url varchar(255)  ); |

| ***savedDocsMapping*** |
| --- |
| create table savedDocsMapping(  savedDocs\_id tinyint primary key auto\_increment,  savedDocDate date,  savedDocHashKey char(20)  ); |

| ***topCategoryMapping*** |
| --- |
| create table topCategoryMapping(  topCategoty\_id tinyint unsigned primary key auto\_increment,  top\_category varchar(255)  ); |

| ***topicMapping*** |
| --- |
| create table topicMapping(  topic\_id tinyint primary key auto\_increment,  topic varchar(255)  ); |

**Result for View instruction**

. ***#1. total\_volume***

| create view total\_volume as  SELECT table\_schema AS 'DB16',  ROUND(SUM(data\_length+index\_length)/1024, 1) AS 'Size(KB)'  FROM information\_schema.tables  WHERE table\_schema = 'DB16'  AND TABLE\_NAME <> 'kubicdb'; |
| --- |
|  |

***#2. userCount***

| create view userCount as  select COUNT(\*) from finalUser; |
| --- |
|  |

***#3. boardCount***

| create view boardCount as  select COUNT(\*) from board; |
| --- |
|  |

***#4. docCount***

| create view docCount as  select COUNT(\*) from document; |
| --- |
|  |

***#5. instInfo***

| create view instInfo as  select published\_institution, count(published\_institution) as CNT from document d join pubInstitutionMapping p on d.pubInst\_id = p.pubInst\_id  group by published\_institution  order by count(published\_institution) asc; |
| --- |
|  |

***#6. my\_docs\_2019\_keyword\_count***

| create view my\_docs\_2019\_keyword\_count as  select keyword, count(keyword) as keyword\_count from savedDoc d  join keyword\_mapping k on d.key\_id = k.key\_id  join DB16.savedDocsMapping sDM on d.savedDocs\_id = sDM.savedDocs\_id  where savedDocDate like '2019%'  group by keyword  order by count(keyword) desc; |
| --- |
|  |

***#7. policy\_writer\_count***

| create view policy\_writer\_count as  select post\_writer, count(post\_writer) as data\_count from document d  left join postWriterMapping pWM on d.postWriter\_id = pWM.postWriter\_id  left join savedDocsMapping sDM on d.hash\_key = sDM.savedDocHashKey  left join savedDoc sD on sD.savedDocs\_id = sDM.savedDocs\_id  left join keyword\_mapping k on sD.key\_id = k.key\_id  where keyword = '토론'  group by post\_writer  order by count(post\_writer) desc; |
| --- |
|  |

***#8. inst\_data\_max\_status***

| create view inst\_data\_max\_status as  with users(eid, institute, occupation) as (  select f.eid, institute, occupation from finalUser f  join instituteMapping iM on f.inst\_id = iM.inst\_id  join occupationMapping oM on f.occu\_id = oM.occu\_id  )  select institute,  (select occupation from users  where institute = u.institute  group by occupation  order by count(occupation) desc  limit 1) as max\_status,  (select count(sD.savedDocs\_id)  from savedDoc sD  left join users us on sD.eid = us.eid  left join occupationMapping o on us.occupation = o.occupation  where sD.eid = us.eid and us.institute = u.institute and us.occupation = (  select occupation from users  where institute = u.institute  group by occupation  order by count(occupation) desc  limit 1)) as max\_status\_count,  (select count(sD.savedDocs\_id)  from savedDoc sD left join users us on sD.eid = us.eid  where sD.eid = us.eid and us.institute = u.institute) as data\_count  from users u  left join savedDoc sD on u.eid = sD.eid  group by institute  order by count(institute) desc; |
| --- |
|  |

***#9. checkDoc***

| create view checkDoc as  select post\_title, post\_writer, published\_institution, post\_date, top\_category from document d  left join postWriterMapping pWM on d.postWriter\_id = pWM.postWriter\_id  left join pubInstitutionMapping pIM on d.pubInst\_id = pIM.pubInst\_id  left join topCategoryMapping tCM on d.topCategory\_id = tCM.topCategory\_id  order by post\_date desc  limit 5; |
| --- |
|  |

***#10. category\_Count***

| create view category\_Count as  select top\_category, count(top\_category) as category\_count, rank() over (order by count(top\_category) desc) as category\_rank  from document d  left join topCategoryMapping tCM on d.topCategory\_id = tCM.topCategory\_id  group by top\_category  order by count(top\_category) desc; |
| --- |
|  |

**ER-diagram**

|  |
| --- |

This is the ER-Diagram we draw.

**Summary of the database size and table sizes**

* database size is 216160.0 KB
* table sizes are below image
* 