**2021-2022 Fall Semester**

**Ege University Department of Computer Engineering**

**Database Management**

**Project Report**

EKİN ASLAN - 05180000097

ALİ İLMAN - 05200000010

ESRA DUMAN - 05190000076

Ingredients

[Intruduction 2](#_Toc94731130)

[ANALYSIS 3](#_Toc94731131)

[MOODLE 3](#_Toc94731132)

[**1. Write a brief explanation using your own words (in English) about these applications in terms of their scope.** 3](#_Toc94731133)

[**2. Write an analysis report for each web application:** 3](#_Toc94731134)

[LINKEDIN 6](#_Toc94731135)

[**1. Write a brief explanation using your own words (in English) about these applications in terms of their scope.** 6](#_Toc94731136)

[**2. Write an analysis report for each web application:** 7](#_Toc94731137)

[DESIGN-CONCEPTUAL MODEL 14](#_Toc94731138)

[DESIGN-LOGICAL MODEL 14](#_Toc94731139)

[5. Convert EER diagram into relational model using the methodology that will be introduced in your course. 14](#_Toc94731140)

[IMPLEMENTATION-PHYSICAL MODEL 17](#_Toc94731141)

[Type.txt 17](#_Toc94731142)

[Delection.sql 18](#_Toc94731143)

[Creation.sql 21](#_Toc94731144)

[Trigger and Assertion.sql 35](#_Toc94731145)

[İnsertion.sql 40](#_Toc94731146)

[Constraints.sql 46](#_Toc94731147)

[Views.Sql 46](#_Toc94731148)

[Queries.sql 47](#_Toc94731149)

# Intruduction

EER çizimi için <https://app.diagrams.net/> sitesi kullanılmıştır.

SQL için MYSQL uygulaması kullanılmıştır.

# ANALYSIS

## MOODLE

### **1. Write a brief explanation using your own words (in English) about these applications in terms of their scope.**

The general purpose of Moodle is to keep records of users education life. This system can be created by an university, an educational institution. The application allows the user to view the documents, projects, assignments and notes of his courses. You must specify a user name in the system before your user can login to the page.

Depending on the user name you have set, you can be an instructor or a student. When you log in as a student, there are the courses you have taken on the left side, the communication, announcements and contact us section in the upper block of the homepage.

As I go down on the homepage, you'll see the section of my lectures. In the lower headings of these courses, you will see the names of the course and the teachers who teach these courses. We just need to click on his name to get the information from any of these teachers. When we click here, we can see the teacher's e-mail address, certificates, lectures, and the last time he checked in.

You can select the language yourself when using this moodle page. There's also a message section to communicate and often receive important announcements via message. These messages can be private, group, or starry.

When we enter any of the courses you have taken, you can see that these courses are sorted by date. You can view video records of the courses that are processed every week and information about courses in the form of pdf, word, powerpoint. You can also upload a given assignment to the system as the instructor intended until the end of the site.

### **2. Write an analysis report for each web application:**

#### a. What is the aim of each application?

Moodle targets students and teachers enrolled in a school. The main purpose of this social platform is to enable students to follow the courses they are enrolled in, what is in the content of these courses, the assignments given and these assignments to the teacher through this platform. Due to this main purpose, lectures and assignments can be followed properly.

#### b. What are the main entities of them?

User, Location, Language, Organization, Student, Instructor, Course, Content, Massage, Ability, Media.

#### c. What are the characteristics of each entity?

User: They are the entities that people represent.

user\_id: Keeps a unique value for each user.

password: Allows a password to be kept for the user.

fname: Keeps the user's first name.

mname: Keeps the middle name of the user.

lname: Keeps the user's last name.

email: Allows the user to keep their e-mail address.

create\_date: Keeps the date the user record was created.

Last\_accees: Keeps the date of the user's last login to the site.

User\_type: It keeps the information that the user is a student or a teacher.

Location: Keeps the address information of individuals or institutions

City: Keeps the name of the city where the person or institutions are located.

Country: Keeps the country name of the person or institution.

Time period: Keeps the time zone of the person or institutions

Address: Keeps the address information of individuals or institutions.

Language: Keeps the language desired to be used on the site.

Language\_name: Keeps the language name selected by the user.

Organization: It is the title where the organization information is kept.

Organization\_id: It provides a unique value for each organization.

Organization\_name: Keeps name information for each organization

Phone: Keeps phone number information for each institution.

Email: Keeps Email information for each institution.

Website: Keeps web site url information for each institution.

Social: Keeps social media for each institution.

Media: Keeps users' image data.

Media\_id: Keeps unique value for each image.

Media\_url: The value that holds the image web address.

Student: User\_type keeps student information.

Student\_id: Keeps a unique value for each student.

Instructor: User\_type holds the trainers information.

Instructor\_id: Keeps unique value for each trainer

Course: Keeps course information.

Course\_id: Provides a unique value for each course.

Course\_name: Keeps name information for each course

Announcements: Keeps announcement information for each lesson

Complete percent: Completion percentage for each lesson.

Content: It is the title where the content information is kept.

Name: Keeps name information for each content.

Content\_type: It holds the content type for each content.

Document: Keeps file information for each content.

Text: Keeps text information for each content.

TODO: Keeps done information for each content.

Ability: It is the part where the abilities of the user are kept.

Certificate\_name: It holds the name of the certificate that the user has obtained from the abilities he has.

Badge: Keeps the badge information that the user has.

#### d. What relationships exists among the entities?

User has Location (1:N): Every user can have a location. A location can have multiple users. Location information is not required.

User has Organization (1: N): Each user is affiliated with an organization. Each institution must have at least 1 user.

User disjoin Student: Each user is either of the student type or must be of another type.

User disjoin Instructor: Each user is either of the instructor type or must be of another type.

User has Ability (N: M): Every user can have the ability. A user may have more than one skill or none at all. Each skill may be owned by more than one user or not at all.

User massage User (N: M): (date,text) Each user can send a message to another user. Message text and time are kept as data.

User has Media (1: N): Each user has a photo. More than one person can use the same photo.

Organization has Location (1: N): Every organization can have a location. There can be more than one institution in a location. Location information is not required.

Organization has Media (1: N): Every institution has a photo. More than one person can use the same photo.

Organization has Course (1: N): Each institution offers at least 1 course. Each course must be taught by an institution.

Course has Content (1: N): Each course must have at least 1 content. Each content must belong to a course.

Student takes Course (N: M): Each student may take more than one course or not at all. At least 1 student can be enrolled in each course or not at all. (Note) Note information of students who buy leather is kept.

Instructor teach Course (N:M): Each instructor may teach more than one lesson or no lesson at all. Each course must have at least 1 instructor.

#### e. What are the constraints related to entities, their characteristics and the relationships among them?

* The user cannot access the records of the course that he is not registered for.
* The user cannot send messages to himself.
* The user cannot enroll in Moodle without enrolling in a school.
* After enrolling in a course, it is not possible to enroll again.

## LINKEDIN

### **1. Write a brief explanation using your own words (in English) about these applications in terms of their scope.**

LinkedIn is a social media platform that allows people to follow what they are doing in the workplace and interact with each other. In order to log in to this social media platform, if you have not registered before, you need to fill in the information such as name, surname, phone, e-mail, date of birth, gender, password and whether you are a student looking for a job or working somewhere. You can become a member by filling out the required information. If you are a member before, it will be enough to enter your password and login with your e-mail address. If you are a member for the first time, you can fill in some information about your profile.

You can choose the schools you have studied or are currently studying, whether you have previous work experience or not, and if not, you can choose the business departments and cities you are looking for. Thanks to the features you have chosen, the ones recommended to you will also be in this area. You can also add a profile picture if you wish. Again, in order to make your profile look more attractive, you can write your areas of expertise and the skills you have developed in the Talents section. These abilities you write can also be approved by other users.

In the activity section, you can share all your activities such as articles, posts and documents. You can celebrate a new job of a friend or business partner or announce a new project yourself. You can add the trainings you have received in the Trainings section. Likewise, you can add the certificates you have earned from the trainings you have received in the Licenses and Certificates section. You can also add languages that you can speak in the Languages section. As you add them to your profile, your profile will have a more effective appearance.

One of the main purposes of LinkedIn is to build connections. Here, you can specify your request by pressing the connect button to the people you want to connect with. If this person also wants to contact you, they will accept your request. Thus, as you connect with people, your own network is formed.

The Organizations section consists of two main parts: Schools and Companies. As I mentioned above, you can specify the school you studied and the company you work for here. You can study and work at the same time in schools, but you can only work in companies. Job postings can be shared by organizations if you are looking for a job. In this way, you can see and apply to these job postings or share them as statuses to be seen.

Users can comment and like the statuses you share. In this way, you can communicate with other users.You can turn on notifications for a job you are waiting for. In these notifications, you can see who has viewed your profile, who wants to connect, or if there is a job posting for you.

If you want to communicate privately with another user you have connected with, you can send him a message. If you want to connect with other trusted people who share and support your goals, you can find communities here in the Group section and join them.

### **2. Write an analysis report for each web application:**

#### a. What is the aim of each application?

The area that LinkedIn is targeting is specifically the business community. For this purpose, you will find the work you do, the projects you are involved in, your skills and training. With these fields we fill, companies and employers are drawn to attention. It increases your chances that these fields are qualified to get started or move to a new job. You can also see that there are jobs, because of the flones that companies share, or the people you connect with.

#### b. What are the main entities of them?

L\_USER, L\_LANGUAGE, JOB\_OFFER, MEDIA, ABILITY, PATENT, L\_ORGANIZATION, PUBLICATION, PROJECT, COURSE, TEST\_SCORE, HONOR\_AND\_AWARD, ORGANIZATION\_ROLE, CERTIFICATE, COUNTRY, CITY, LOCATION, L\_GROUP, L\_STATUS, L\_COMMENT

#### c. What are the characteristics of each entity?

USER: The entities that people represent.

user\_id: Keeps a unique value for each user.

user\_password: Allows a password to be kept for the user.

fname: Keeps the user's first name.

mname: Keeps the middle name of the user.

lname: Keeps the user's last name.

gender: Allows the user's gender to be kept.

phone: Allows the user's phone number to be kept.

e\_mail: Allows keeping the e-mail address of the user.

create\_date: Keeps the date the user record was created.

user\_type: There are 2 user types on LinkedIn, free and premium. It keeps its knowledge.

friend\_count: Allows the user to keep the number of friends.

JOB\_OFFER: Entities where job postings are represented. Users and companies can interact with job postings. Companies can post job postings, people can apply to these job postings.

job\_offer\_id: Provides a unique value for each job posting.

create\_date: The date the job posting was created.

deadline: The date that the ad will expire.

apply\_count: Counts the number of users who applied to the post.

industry: It keeps the information of which industry the job posting is in.

employment\_type: It keeps the information of what type of employment (part time, full time, etc.) the job posting contains.

seniority\_level: Seniority type information is kept. (intermediate, senior, etc.)

job\_functions: Keeps information about job fields. (marketing etc.)

offer\_description: This is the part where the explanations about the ad are kept.

MEDIA: Media are assets created by data such as photos and videos. Users can have media in their shared status, cover photos, profile photos.

media\_id: Keeps a unique value for each media.

media\_url: Keeps the url information of each media.

ABILITY: Entities where talents and abilities are represented.

ability\_name: Ability name information is kept as unique.

LANGUAGE: It is the entity that contains the language names and degrees. Users can own licenses on LinkedIn.

language\_name: Contains the language name.

Profiency: Keeps the level of the language. (like B1, B2)

ORGANIZATION: These are the entities that represent structures such as schools, companies, associations.

organization\_id: Provides a unique value for each organization.

organization\_name: It keeps the name of the organization.

employee\_count: People can work in organizations. Keeps the number of employees.

about: This is the part where organizations introduce themselves.

head\_quarter: Keeps where the headquarters is.

web\_site: Holds the web site address of the organization.

sector: It keeps the information in which sector the organization is in.

foundation\_date: It keeps the information of the date when the organization was founded in real life.

service\_type: Specifies how it serves within the sector.

organization\_type: Specifies the type of organization (school, company, association).

PUBLICATION: It is the entity where the published articles are represented. Users have to broadcast through an organization.

publication\_id: Keeps a unique value for each publication.

Title: Indicates the title of the publication.

publication\_url: Keeps the publication URL address.

publish\_date: Indicates the date on which the publication was published.

publish\_description: Specifies the descriptions about the publication.

PROJECT: Represents projects. Users can create projects.

project\_id: Keeps a unique value for each project.

start\_date: Specifies the start date of the project.

end\_date: Indicates the end date of the project.

project\_url: Keeps the project URL address.

project\_description: Specifies descriptions about the project.

COURSE: Organizations can offer courses and users can take courses. They are the entities for which courses are represented.

course\_id: Keeps a unique value for each course.

course\_name: Specifies the course name.

TEST\_SCORE: It is the scores that the users get from the exams. These exams may be made by organizations.

test\_score\_id: Keeps a unique value for each test

Test\_score\_name: Keeps the name of each exam.

score: Indicates the value taken from the exam.

test\_date: Specifies the time of the exam.

test\_description: Specifies the explanations about the exam.

HONOR\_AND\_AWARD: Entities that represent structures containing Award/Honor. Users can have these awards/honors. These awards/honors can be given by organizations.

honor\_and\_award\_id: Keeps a unique value for each award/honor.

title: Indicates the title information about the award/honor.

h\_a\_issuer: Keeps the award/honor issuer information.

h\_a\_date: Indicates the date the award was given.

h\_a\_description: Indicates a description about the reward.

CERTIFICATE: Users can have certificates and these certificates can be issued by organizations.

certificate\_id: It provides a unique value for each certificate.

cerfiticate\_name: Specifies the certificate name.

issue\_date: Specifies the date the certificate was issued.

expiration\_date: Specifies the expiration date of the certificate.

certificate\_url: Specifies the URL address of the certificate.

PATENT: They are the assets that represent the inventions that people find.

patent\_id: Allows a unique value to be kept for each patent.

patent\_number: Allows the patent number to be kept.

patent\_url: Keeps the patent URL address.

title: Holds the title information for the patent published in the profile.

patent\_status: Indicates the status of the patent.

patent\_date: Specifies the patent date.

patent\_description: Specifies the patent description.

LOCATION: Represents addresses. Organizations and users are linked by addresses.

location\_id: Keeps a unique value for each address.

zipcode: Keeps the zip code information of the address.

neighborhood: It keeps the neighborhood information of the address.

apt\_no: It keeps the number of the address.

description: Contains descriptions about the address.

COUNTRY: Users and other entities can only interact with countries, regardless of addresses. Addresses can be associated with countries.

country\_name: Keeps the country name.

CITY: Weak entity. COUNTRY must exist for this entity to exist.

city\_name: Keeps the city name.

GROUP: Users can interact with each other in groups. There are the group owner, the users who manage the group, and the group members. Only group members can interact in the group.

group\_id: Keeps a unique value for each group.

privacy\_type: : Allows the privacy status of the group to be kept.

group\_name: : It keeps the name information of the group.

about text: Keeps the description information about the group.

group\_rules: Group rules are specified.

STATUS: Users can share statuses that contain media and text to interact with other users. Statuses can be liked by users.

status\_id: Keeps a unique value for each status.

status\_text: Keeps the text in the status.

Status\_date: Keeps the date the status was shared.

counter: Keeps the number of likes of the status.

Counter\_type: Keeps the type of reactions left to the situation.

COMMENT: It is the high level entity where the comments are kept. Users can comment on statuses and users can like these comments.

comment\_id: Keeps a unique value for each media.

comment\_text: It keeps the content of the comment made.

C\_like\_count: It keeps the number of likes of the comment.

#### d. What relationships exists among the entities?

CITY in (N: 1) COUNTRY: Cities are located within countries. There can be many cities in a country. A city can only be in one country. To be a city, it has to be a country.

ORGANIZATION publish\_publication (N: 1) PUBLICATION: Every published publication must be published by an organization. To be able to broadcast, the user must be registered. An organization can publish more than one.

USER has (1: N) HONOR\_AND\_AWARD: A user can have multiple awards/honors. The award belongs to only 1 person.

USER has (1: N) TEST\_SCORE: A user can have many test values. The test value owned belongs to only 1 person. In order for the test value to be valid, it must be a user.

USER has (1: N) ORGANIZATION\_ROLE: A user may have held many organizational roles. The assigned person can only be 1 person. (Even if the mission type is the same, the squad is different.)

USER has\_location (N: 1) LOCATION: A user has an address. There can be many users at one address.

LOCATION has\_city (N: 1) CITY: Addresses have cities. An address can be a city. A city can have many addresses. To be an address, it has to be a city.

ORGANIZATION associated\_with (1: N) PROJECT: A project is associated with only 1 organization. 1 organization can be associated with many projects.

ORGANIZATION publish\_publication (1: N) PUBLICATION: A publication can only be published by 1 organization. An organization can publish many publications.

ORGANIZATION publish\_publication (1:N) TEST\_SCORE: A test is only about 1 organization. An organization may have done many tests.

ORGANIZATION publish\_publication (1: N) HONOR\_AND\_AWARD: One award is related to only 1 organization. An organization may have awarded many awards/honors.

ORGANIZATION publish\_publication (1: N) ORGANIZATION\_ROLE: An organization task relates to only one organization. An organization may have assigned many organizational tasks. (at this part)

ORGANIZATION publish\_publication (1:N) CERTIFICATE: An organization can issue many certificates. The issued certificate is issued by only one organization.

STATUS has(N: 1) MEDIA: Statuses can have media. A state can only have one media. A media can belong to many states.

STATUS share (1: N) STATUS: States can contain states. A state can be hosted in many situations. A state can contain at most 1 state.

USER has\_ability (N: M) ABILITY: A user can have many abilities. The ability can belong to many users.

USER apply (N: M) JOB\_OFFER: A user can apply for many job postings. Many people may have applied for the job posting.

USER author (N: M) PUBLICATION: A publication can be written by many people. A user can write many posts. It must be a user to be able to broadcast.

USER create (N: M) PROJECT: A user can create many projects. The created project can belong to many people. In order to be a project, it must be a user.

USER take (N: M) COURSE: A user can take many courses. The course taken may have been taken by many users.

USER follow (N: M) L\_ORGANIZATION: A user can follow many organizations. The organization that is followed can be followed by many people.

USER works\_on (N: M) L\_ORGANIZATION: A user can work in many organizations. Many people can work in the company.

USER member\_of (N: M) L\_GROUP: A user can be a member of many groups. Many users can be members of the group to which they are members.

USER has\_certificate (N: M) CERTIFICATE: A user can have many certificates. The owned certificate can be owned by many users.

USER message (N: M) L\_USER: A user can send messages to many people. The person to whom the message is sent can receive messages from many people.

USER connect (N: M) L\_USER: One user can connect with many users. The connected user can connect with many people.

USER has (N: M) HAS\_ABILITY : A user can have many abilities. Possession can be confirmed by many users.

USER has\_language (N: M) LANGUAGE: A user can speak multiple languages. The known language may be known to many people.

USER invents (1:N) PATENT: A person can have more than one patent. Every patent must have an owner.

ORGANIZATION invents PATENT(1:N):Organizations can hold patents. Every patent must have an owner.

ORGANIZATION (N: M) LOCATION: An organization can be located at many addresses. There may be many organizations at the current address.

JOB\_OFFER ask (N:M) ABILITY: A job posting can request many skills. The requested talent may have been requested from many job postings.

USER share\_status GROUP (N:M): Users can share status within groups. A user can share many statuses in many groups.

STATUS in GROUP: A group can have many states. A status can be shared in many groups.

ORGANIZATION publish\_offer JOB\_OFFER : A job posting, only one organization can be advertised. An organization can post many job postings.

USER publish\_offer JOB\_OFFER(1:N): Users can post many job postings. A job posting can only be shared by one user.

USER and , ORGANIZATION has\_profile\_photo (1: 1) MEDIA: People and organizations can have a profile photo. A person or organization can have up to 1 profile photo. Owned profile photo can only belong to one user. It must be the profile photo of the user. (It can be a photo by default.)

USER and ORGANIZATION has\_cover\_photo (1: 1) MEDIA: People and organizations can have a cover photo. A person or organization can have up to 1 cover photo. Owned cover photo can only belong to one user. The user must have a cover photo. (It can be a photo by default.)

USER owns (1:N) GROUP and ORGANIZATION: A user can own multiple groups or organizations. For a group or organization to exist, it must be an owner.

USER and ORGANIZATION share (1: N) STATUS: Individuals and organizations can share states. A user can share many statuses. Shared status is for one user only.

USER and ORGANIZATION manages (N: M) GROUP and ORGANIZATION: Users can manage organizations and groups. An organization or group can be managed by many people. A user can manage many organizations or groups.

USER and ORGANIZATION comment (N: M) STATUS: Individuals and organizations can comment on situations. A user can comment on many situations. Many users can comment on a situation.

USER and ORGANIZATION like (N: M) STATUS and COMMENT: Persons and organizations; They can like comments and statuses. Many users may like many comments and statuses. Comments and statuses can be liked by many users.

USER message USER(N:M): Users can send messages to multiple users.

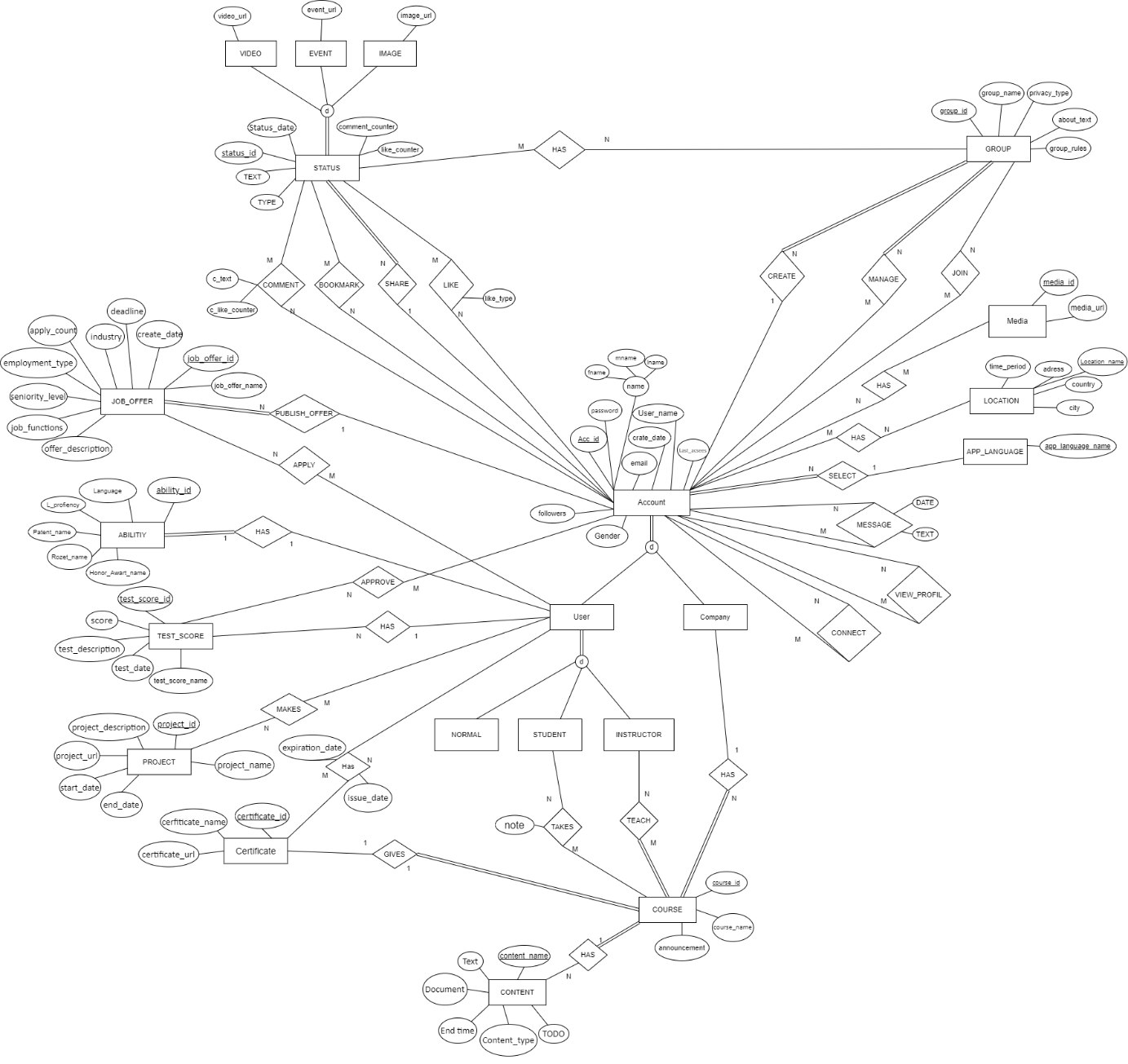
#### e. What are the constraints related to entities, their characteristics and the relationships among them?

* To become a member of LinkedIn, you must be at least 13 years old.

#### A user can connect with up to 1000 users.

* A connection cannot be re-established between established contacts.
* To be an administrator in the group, it is necessary to be a member of the group.
* The user cannot send messages to himself.

# DESIGN-CONCEPTUAL MODEL



# DESIGN-LOGICAL MODEL

## 5. Convert EER diagram into relational model using the methodology that will be introduced in your course.

1.ITERATION

Step1:

App\_Language(App\_language\_name)

Location( Location\_name, City, Country, time\_period, address)

Group( group\_id, group\_name, privacy\_type, about\_text, group\_rules)

Job\_Offer( job\_offer\_id, job\_offer\_name, create\_date, deadline, apply\_count, Industry, employment\_type, seniority\_level, job\_functions, offer\_description)

Test\_score(test\_score\_id, test\_score\_name, Score, test\_date, test\_description)

Certificate(certificate\_id, cerfiticate\_name, cerfiticate\_url, expiration\_date, issue\_date)

Project(project\_id, project\_name, project\_description, project\_url, start\_date, end\_date)

Course(course\_id, course\_name, announcement)

Content(content\_name , text, document, end\_time, content\_type, TODO )

Media(Media\_id,media\_url)

Ability(ability\_id, language, L\_profiency, Patent\_name, Rozet\_name, Honor\_Award\_name)

Step2:

Step3:

Course(…., certificate\_id(FK)) //1:1 ilişki course to Certificate

Step4:

Content(…, course\_id(FK)) // 1:m ilişki course to content

Step5:

Step:6

Step7:

Step8:

Account(Acc\_id, user\_name, pasword, Fanme, Mname, Lname, gender, email,create\_date, premium, last\_acsess, ACCOUNT\_TYPE)

Note: Açılımı ilk başta 8A ya göre açtık. Sonrasında modelimizi 8C ye cevirdik.

User(Acc\_id(FK),User\_type)

Company(Acc\_id(fk))

Status(status\_id, text, Status\_date ,Status\_type) (8A)

Video(status\_id(fk), video\_url)

Event(status\_id(fk), event\_url)

İmage(status\_id(Fk), image\_url)

Step:9

2.ITERATION

Step1:

Step2:

Step3:

Ability(...acc\_id(fk))//asstertion

Step4:

Account(... App\_Language\_name(FK))

Group(….,Crate\_Acc\_id(FK))

Test\_Score(….., acc\_id(FK)) //assterition

Course(…..,acc\_id(FK)) //assterition

Job\_offer(…,acc\_id(fk))

Status(...acc\_id(FK))

Step5:

User\_makes\_Project(makes)(user\_id(FK), proje\_id(FK))

Share\_in\_Group(has)(Status\_id, Group\_id)

Acc\_has\_Location(has)(acc\_id(FK), location\_name(FK))

Acc\_bookmark\_Status(acc\_id(FK), status\_id(FK))

Acc\_comment\_Status(acc\_id(FK),status\_id(FK),c\_text, c\_like\_counter)

Acc\_share\_Status(acc\_id(FK), status\_id(FK))

Acc\_like\_Status(acc\_id(FK), status\_id(FK), like\_type)

Acc\_manage\_Group(acc\_id(FK), group\_id(FK))

Acc\_join\_Group(acc\_id(FK), group\_id(FK))

User\_has\_Ability(acc\_id (FK), ability\_name (FK))//assertion

User\_apply\_Job\_Offer(acc\_id (FK), job\_offer\_id (FK))//assertion

Acc\_approve\_Test\_Score(acc\_id(FK), test\_score\_id(FK))

User\_has\_Certificate(acc\_id (FK), certificate\_id(FK),issue\_date,expiration\_date)//assertion

Message ( sender\_id(FK acc\_id), receiver\_id(FK acc\_id), date, text)

View\_Profile(acc\_id(FK), Viewed\_id(FK) )

Connet( acc\_id(FK), ConnecterId (FK acc\_id))

Acc\_has\_Media(acc\_id(FK), Media\_id(FK))

Step6:

Step7:

Step8:

Student(acc\_id (FK)) //with

Instructer(acc\_id (FK)) //with

Normal(acc\_id (FK)) //with

Step9:

3.ITERATION

Step1:

Step2:

Step3:

Step4:

Step5:

Stu\_takes\_Course(acc\_id(FK), course\_id(FK),note) //assertion

Instructor\_teaches\_Course(acc\_id (FK), course\_id(FK)) //assertion

Step6:

Step7:

Step8:

# IMPLEMENTATION-PHYSICAL MODEL

## Type.txt

Kullandığımız typlere göre temsil ettikleri değerleri not ettiğimiz kısımdır bilgilendirme amaçlıdır.

account\_type

------------

0 USER

1 COMPANY

status\_type

-----------

NULL BLANK

0 VIDEO

1 EVENT

2 IMAGE

user\_type

---------

0 NORMAL

1 STUDENT

2 INSTRUCTOR

ability\_type

------------

NULL NORMAL

1 - LANGUAGE

like\_type

---------

0 NORMAL

1 KALPLİ

2 GÜLÜCÜK

3 ÜZGÜN

4 KIZGIN

## Delection.sql

/\*Tablolarda yeniden oluşturma işlerini tanımladığımız zaman toplu silme için kullandığımız alandır\*/

USE database\_schema;

SET FOREIGN\_KEY\_CHECKS = 0;

DELETE FROM ability WHERE TRUE;

DELETE FROM account\_ WHERE TRUE;

DELETE FROM account\_approve\_test\_score WHERE TRUE;

DELETE FROM account\_bookmark\_status WHERE TRUE;

DELETE FROM account\_comment\_status WHERE TRUE;

DELETE FROM account\_has\_location WHERE TRUE;

DELETE FROM account\_has\_media WHERE TRUE;

DELETE FROM account\_like\_status WHERE TRUE;

DELETE FROM account\_manage\_group WHERE TRUE;

DELETE FROM app\_language WHERE TRUE;

DELETE FROM certificate WHERE TRUE;

DELETE FROM company WHERE TRUE;

DELETE FROM connect WHERE TRUE;

DELETE FROM content WHERE TRUE;

DELETE FROM course WHERE TRUE;

DELETE FROM event\_ WHERE TRUE;

DELETE FROM group\_ WHERE TRUE;

DELETE FROM image WHERE TRUE;

DELETE FROM instructor WHERE TRUE;

DELETE FROM instructor\_teaches\_course WHERE TRUE;

DELETE FROM job\_offer WHERE TRUE;

DELETE FROM location WHERE TRUE;

DELETE FROM media WHERE TRUE;

DELETE FROM message WHERE TRUE;

DELETE FROM normal WHERE TRUE;

DELETE FROM project WHERE TRUE;

DELETE FROM share\_in\_group WHERE TRUE;

DELETE FROM status\_ WHERE TRUE;

DELETE FROM student WHERE TRUE;

DELETE FROM student\_takes\_course WHERE TRUE;

DELETE FROM test\_score WHERE TRUE;

DELETE FROM user\_ WHERE TRUE;

DELETE FROM user\_apply\_job WHERE TRUE;

DELETE FROM user\_has\_certificate WHERE TRUE;

DELETE FROM user\_makes\_project WHERE TRUE;

DELETE FROM video WHERE TRUE;

DELETE FROM view\_profile WHERE TRUE;

DROP TABLE ability;

DROP TABLE account\_;

DROP TABLE account\_approve\_test\_score;

DROP TABLE account\_bookmark\_status;

DROP TABLE account\_comment\_status;

DROP TABLE account\_has\_location;

DROP TABLE account\_has\_media;

DROP TABLE account\_like\_status;

DROP TABLE account\_manage\_group;

DROP TABLE app\_language;

DROP TABLE certificate;

DROP TABLE company;

DROP TABLE connect;

DROP TABLE content;

DROP TABLE course;

DROP TABLE event\_;

DROP TABLE group\_;

DROP TABLE image;

DROP TABLE instructor;

DROP TABLE instructor\_teaches\_course;

DROP TABLE job\_offer;

DROP TABLE location;

DROP TABLE media;

DROP TABLE message;

DROP TABLE normal;

DROP TABLE project;

DROP TABLE share\_in\_group;

DROP TABLE status\_;

DROP TABLE student;

DROP TABLE student\_takes\_course;

DROP TABLE test\_score;

DROP TABLE user\_;

DROP TABLE user\_apply\_job;

DROP TABLE user\_has\_certificate;

DROP TABLE user\_makes\_project;

DROP TABLE video;

DROP TABLE view\_profile;

DROP DATABASE database\_schema;

SET FOREIGN\_KEY\_CHECKS = 1;

## Creation.sql

/\* Tabloları eklediğimiz kısımdır\*/

CREATE DATABASE database\_schema;

USE database\_schema;

CREATE TABLE app\_language (

app\_language\_name varchar(255),

PRIMARY KEY (app\_language\_name)

);

CREATE TABLE account\_ (

account\_id int,

user\_name varchar(255),

user\_password varchar(255),

first\_name varchar(255),

middle\_name varchar(255),

last\_name varchar(255),

gender varchar(255),

email\_address varchar(255),

create\_date date,

premium bool,

last\_access date,

account\_type int,

user\_type int,

PRIMARY KEY (account\_id),

app\_language\_name varchar(255),

FOREIGN KEY (app\_language\_name) REFERENCES app\_language(app\_language\_name),

followers int DEFAULT 0

);

CREATE TABLE location (

location\_name varchar(255),

city varchar(255),

country varchar(255),

time\_period int,

address varchar(255),

PRIMARY KEY (location\_name)

);

CREATE TABLE group\_ (

group\_id int,

group\_name varchar(255),

privacy\_type bool,

about\_text varchar(255),

group\_rules varchar(255),

PRIMARY KEY (group\_id),

create\_account\_id int,

FOREIGN KEY (create\_account\_id) REFERENCES account\_(account\_id)

);

CREATE TABLE job\_offer (

job\_offer\_id int,

job\_offer\_name varchar(255),

create\_date date,

deadline date,

apply\_count int DEFAULT 0,

industry varchar(255),

employment\_type varchar(255),

seniority\_level varchar(255),

job\_functions varchar(255),

description varchar(255),

PRIMARY KEY (job\_offer\_id),

creator\_id int,

FOREIGN KEY (creator\_id) REFERENCES account\_(account\_id)

);

CREATE TABLE test\_score (

test\_score\_id int,

test\_score\_name varchar(255),

score int,

test\_date date,

test\_description varchar(255),

PRIMARY KEY (test\_score\_id),

account\_id int,

FOREIGN KEY (account\_id) REFERENCES account\_(account\_id)

);

CREATE TABLE certificate (

certificate\_id int,

certificate\_name varchar(255),

certificate\_url varchar(255),

PRIMARY KEY (certificate\_id)

);

CREATE TABLE project (

project\_id int,

project\_name varchar(255),

project\_description varchar(255),

project\_url varchar(255),

start\_date date,

end\_date date,

PRIMARY KEY (project\_id)

);

CREATE TABLE course (

course\_id int,

course\_name varchar(255),

announcement varchar(255),

PRIMARY KEY (course\_id),

certificate\_id int,

account\_id int,

FOREIGN KEY (certificate\_id) REFERENCES certificate(certificate\_id),

FOREIGN KEY (account\_id) REFERENCES account\_(account\_id)

);

CREATE TABLE content (

content\_name varchar(255),

content\_text varchar(255),

document varchar(255),

end\_time date,

content\_type varchar(255),

todo varchar(255),

PRIMARY KEY (content\_name),

course\_id int,

FOREIGN KEY (course\_id) REFERENCES course(course\_id)

);

CREATE TABLE media ( /\* Profil fotoğrafı \*/

media\_id int,

media\_url varchar(255),

PRIMARY KEY (media\_id)

);

CREATE TABLE status\_ (

status\_id int,

text\_ varchar(255),

status\_date date,

status\_type int,

like\_count int DEFAULT 0,

comment\_count int DEFAULT 0,

PRIMARY KEY (status\_id),

account\_id int,

FOREIGN KEY (account\_id) REFERENCES account\_(account\_id)

);

CREATE TABLE user\_ ( /\* DEPRECATED! \*/

/\* user\_type int, \*/

account\_id int,

PRIMARY KEY (account\_id),

FOREIGN KEY (account\_id) REFERENCES account\_(account\_id)

);

CREATE TABLE company ( /\* DEPRECATED! \*/

account\_id int,

PRIMARY KEY (account\_id),

FOREIGN KEY (account\_id) REFERENCES account\_(account\_id)

);

CREATE TABLE ability (

ability\_id int,

language\_ varchar(255),

language\_profiency varchar(255),

patent\_name varchar(255),

badge\_name varchar(255),

honor\_award\_name varchar(255),

PRIMARY KEY (ability\_id),

account\_id int,

FOREIGN KEY (account\_id ) REFERENCES account\_(account\_id)

);

CREATE TABLE video (

video\_url varchar(255),

status\_id int,

PRIMARY KEY (status\_id),

FOREIGN KEY (status\_id) REFERENCES status\_(status\_id)

);

CREATE TABLE event\_ (

event\_url varchar(255),

status\_id int,

PRIMARY KEY (status\_id),

FOREIGN KEY (status\_id) REFERENCES status\_(status\_id)

);

CREATE TABLE image (

image\_url varchar(255),

status\_id int,

PRIMARY KEY (status\_id),

FOREIGN KEY (status\_id) REFERENCES status\_(status\_id)

);

CREATE TABLE user\_makes\_project (

account\_id int,

project\_id int,

PRIMARY KEY (account\_id, project\_id),

FOREIGN KEY (account\_id) REFERENCES account\_(account\_id),

FOREIGN KEY (project\_id) REFERENCES project(project\_id)

);

CREATE TABLE share\_in\_group (

status\_id int,

group\_id int,

PRIMARY KEY (status\_id, group\_id),

FOREIGN KEY (status\_id) REFERENCES status\_(status\_id),

FOREIGN KEY (group\_id) REFERENCES group\_(group\_id)

);

CREATE TABLE account\_has\_location (

account\_id int,

location\_name varchar(255),

PRIMARY KEY (account\_id, location\_name),

FOREIGN KEY (account\_id) REFERENCES account\_(account\_id),

FOREIGN KEY (location\_name) REFERENCES location(location\_name)

);

CREATE TABLE account\_bookmark\_status (

account\_id int,

status\_id int,

PRIMARY KEY (account\_id, status\_id),

FOREIGN KEY (account\_id) REFERENCES account\_(account\_id),

FOREIGN KEY (status\_id) REFERENCES status\_(status\_id)

);

CREATE TABLE account\_comment\_status (

account\_id int,

status\_id int,

comment\_text varchar(255),

comment\_like\_counter int,

PRIMARY KEY (account\_id, status\_id),

FOREIGN KEY (account\_id) REFERENCES account\_(account\_id),

FOREIGN KEY (status\_id) REFERENCES status\_(status\_id)

);

CREATE TABLE account\_like\_status (

account\_id int,

status\_id int,

like\_type int,

PRIMARY KEY (account\_id, status\_id),

FOREIGN KEY (account\_id) REFERENCES account\_(account\_id),

FOREIGN KEY (status\_id) REFERENCES status\_(status\_id)

);

CREATE TABLE account\_manage\_group (

account\_id int,

group\_id int,

PRIMARY KEY (account\_id, group\_id),

FOREIGN KEY (account\_id) REFERENCES account\_(account\_id),

FOREIGN KEY (group\_id) REFERENCES group\_(group\_id)

);

CREATE TABLE user\_apply\_job (

account\_id int,

job\_offer\_id int,

PRIMARY KEY (account\_id, job\_offer\_id),

FOREIGN KEY (account\_id) REFERENCES account\_(account\_id),

FOREIGN KEY (job\_offer\_id) REFERENCES job\_offer(job\_offer\_id)

);

CREATE TABLE account\_approve\_test\_score (

account\_id int,

test\_score\_id int,

PRIMARY KEY (account\_id, test\_score\_id),

FOREIGN KEY (account\_id) REFERENCES account\_(account\_id),

FOREIGN KEY (test\_score\_id) REFERENCES test\_score(test\_score\_id)

);

CREATE TABLE user\_has\_certificate (

account\_id int,

certificate\_id int,

issue\_date date,

expiration\_date date,

PRIMARY KEY (account\_id, certificate\_id),

FOREIGN KEY (account\_id) REFERENCES account\_(account\_id),

FOREIGN KEY (certificate\_id) REFERENCES certificate(certificate\_id)

);

CREATE TABLE message (

sender\_id int,

receiver\_id int,

text\_message varchar(255),

message\_date date,

PRIMARY KEY (sender\_id, receiver\_id),

FOREIGN KEY (sender\_id) REFERENCES account\_(account\_id),

FOREIGN KEY (receiver\_id) REFERENCES account\_(account\_id)

);

CREATE TABLE view\_profile (

account\_id int,

viewed\_id int,

PRIMARY KEY (account\_id, viewed\_id),

FOREIGN KEY (account\_id) REFERENCES account\_(account\_id),

FOREIGN KEY (viewed\_id) REFERENCES account\_(account\_id)

);

CREATE TABLE connect (

account\_id int,

connecter\_id int,

PRIMARY KEY (account\_id, connecter\_id),

FOREIGN KEY (account\_id) REFERENCES account\_(account\_id),

FOREIGN KEY (connecter\_id) REFERENCES account\_(account\_id)

);

CREATE TABLE account\_has\_media (

account\_id int,

media\_id int,

PRIMARY KEY (account\_id, media\_id),

FOREIGN KEY (account\_id) REFERENCES account\_(account\_id),

FOREIGN KEY (media\_id) REFERENCES media(media\_id)

);

CREATE TABLE student ( /\* DEPRECATED! \*/

account\_id int,

PRIMARY KEY (account\_id),

FOREIGN KEY (account\_id) REFERENCES account\_(account\_id)

);

CREATE TABLE instructor ( /\* DEPRECATED! \*/

account\_id int,

PRIMARY KEY (account\_id),

FOREIGN KEY (account\_id) REFERENCES account\_(account\_id)

);

CREATE TABLE normal ( /\* DEPRECATED! \*/

account\_id int,

PRIMARY KEY (account\_id),

FOREIGN KEY (account\_id) REFERENCES account\_(account\_id)

);

CREATE TABLE student\_takes\_course (

account\_id int,

course\_id int,

note int,

PRIMARY KEY (account\_id, course\_id),

FOREIGN KEY (account\_id) REFERENCES account\_(account\_id),

FOREIGN KEY (course\_id) REFERENCES course(course\_id)

);

CREATE TABLE instructor\_teaches\_course (

account\_id int,

course\_id int,

PRIMARY KEY (account\_id, course\_id),

FOREIGN KEY (account\_id) REFERENCES account\_(account\_id),

FOREIGN KEY (course\_id) REFERENCES course(course\_id)

);

CREATE TABLE account\_join\_group (

account\_id int,

group\_id int,

PRIMARY KEY (account\_id, group\_id),

FOREIGN KEY (account\_id) REFERENCES account\_(account\_id),

FOREIGN KEY (group\_id) REFERENCES group\_(group\_id)

);

## Trigger and Assertion.sql

/\*Bizden istenen Assertion ve Triggeri oluşturduğumuz kısımdır\*/

/\* TRIGGERS \*/

/\* COUNT LIKES \*/

DELIMITER $$

CREATE TRIGGER count\_status\_likes AFTER INSERT ON account\_like\_status FOR EACH ROW BEGIN

UPDATE status\_

SET like\_count = like\_count + 1 WHERE status\_id = NEW.status\_id;

END

$$ DELIMITER ;

/\* COUNT COMMENTS \*/

DELIMITER $$

CREATE TRIGGER count\_status\_comments AFTER INSERT ON account\_comment\_status FOR EACH ROW

BEGIN

UPDATE status\_

SET comment\_count = comment\_count + 1 WHERE status\_id = NEW.status\_id;

END

$$ DELIMITER ;

/\* COUNT FOLLOWERS \*/

DELIMITER $$

CREATE TRIGGER count\_followers AFTER INSERT ON connect FOR EACH ROW BEGIN

UPDATE account\_

SET followers = followers + 1 WHERE account\_id = NEW.account\_id OR account\_id = NEW.connecter\_id;

END

$$ DELIMITER ;

/\* COUNT JOB OFFER APPLIES \*/

DELIMITER $$

CREATE TRIGGER count\_job\_offer\_applies AFTER INSERT ON user\_apply\_job FOR EACH ROW BEGIN

UPDATE job\_offer

SET apply\_count = apply\_count + 1 WHERE job\_offer\_id = NEW.job\_offer\_id;

END

$$ DELIMITER ;

/\* STUDENT NOTE CONTROL \*/

/\* DELIMITER $$

CREATE TRIGGER student\_note\_control BEFORE UPDATE ON student\_takes\_course FOR EACH ROW BEGIN

IF note > 100 THEN

UPDATE student\_takes\_course

SET note = 100;

END IF;

END

$$ DELIMITER ; \*/

/\* ASSERTIONS WITH USING TRIGGERS \*/

/\* TEST SCORE ACCOUNT TYPE CONTROL \*/

DELIMITER $$

CREATE TRIGGER test\_score\_control BEFORE INSERT ON test\_score FOR EACH ROW

BEGIN

IF EXISTS (

SELECT \*

FROM account\_ AS a

WHERE a.account\_id = NEW.account\_id AND a.account\_type = 1 )

THEN SIGNAL SQLSTATE '45000'; /\* İstenmeyen koşul sağlanmadığında hata kodu gönderme. \*/

END IF;

END

$$ DELIMITER ;

/\* COURSE ACCOUNT TYPE CONTROL \*/

DELIMITER $$

CREATE TRIGGER course\_control BEFORE INSERT ON course FOR EACH ROW

BEGIN

IF EXISTS (

SELECT \*

FROM account\_ AS a

WHERE a.account\_id = NEW.account\_id AND a.account\_type = 0 )

THEN SIGNAL SQLSTATE '45000'; /\* İstenmeyen koşul sağlanmadığında hata kodu gönderme. \*/

END IF;

END

$$ DELIMITER ;

/\* USER APPLY JOB ACCOUNT TYPE CONTROL \*/

DELIMITER $$

CREATE TRIGGER user\_apply\_job\_control BEFORE INSERT ON user\_apply\_job FOR EACH ROW

BEGIN

IF EXISTS (

SELECT \*

FROM account\_ AS a

WHERE a.account\_id = NEW.account\_id AND a.account\_type = 1 )

THEN SIGNAL SQLSTATE '45000'; /\* İstenmeyen koşul sağlanmadığında hata kodu gönderme. \*/

END IF;

END

$$ DELIMITER ;

/\* USER HAS CERTIFICATE ACCOUNT TYPE CONTROL \*/

DELIMITER $$

CREATE TRIGGER user\_has\_certificate\_control BEFORE INSERT ON user\_has\_certificate FOR EACH ROW

BEGIN

IF EXISTS (

SELECT \*

FROM account\_ AS a

WHERE a.account\_id = NEW.account\_id AND a.account\_type = 1 )

THEN SIGNAL SQLSTATE '45000'; /\* İstenmeyen koşul sağlanmadığında hata kodu gönderme. \*/

END IF;

END

$$ DELIMITER ;

/\* STUDENT TAKES COURSE ACCOUNT TYPE CONTROL \*/

DELIMITER $$

CREATE TRIGGER student\_takes\_course\_control BEFORE INSERT ON student\_takes\_course FOR EACH ROW

BEGIN

IF EXISTS (

SELECT \*

FROM account\_ AS a

WHERE a.account\_id = NEW.account\_id AND ( a.account\_type = 1 OR a.user\_type != 1 )) /\* Normal ve Instructor olan User'ları alma. \*/

THEN SIGNAL SQLSTATE '45000'; /\* İstenmeyen koşul sağlanmadığında hata kodu gönderme. \*/

END IF;

END

$$ DELIMITER ;

/\* INSTRUCTOR TEACHES COURSE ACCOUNT TYPE CONTROL \*/

DELIMITER $$

CREATE TRIGGER instructor\_teaches\_course\_control BEFORE INSERT ON instructor\_teaches\_course FOR EACH ROW

BEGIN

IF EXISTS (

SELECT \*

FROM account\_ AS a

WHERE a.account\_id = NEW.account\_id AND ( a.account\_type = 1 OR a.user\_type != 2 )) /\* Normal ve Instructor olan User'ları alma. \*/

THEN SIGNAL SQLSTATE '45000'; /\* İstenmeyen koşul sağlanmadığında hata kodu gönderme. \*/

END IF;

END

$$ DELIMITER ;

/\* ABILITY ACCOUNT TYPE CONTROL \*/

DELIMITER $$

CREATE TRIGGER ability\_account\_control BEFORE INSERT ON ability FOR EACH ROW

BEGIN

IF EXISTS (

SELECT \*

FROM ability AS ab

JOIN account\_ AS ac ON ab.account\_id = ac.account\_id

WHERE ac.account\_type = 1 )

THEN SIGNAL SQLSTATE '45000'; /\* İstenmeyen koşul sağlanmadığında hata kodu gönderme. \*/

END IF;

END

$$ DELIMITER ;

/\* CONNECT CONTROL \*/

DELIMITER $$

CREATE TRIGGER connect\_control BEFORE INSERT ON connect FOR EACH ROW

BEGIN

IF EXISTS (

SELECT \*

FROM connect AS f, connect AS s

WHERE f.account\_id = s.connecter\_id AND f.connecter\_id = s.account\_id )

THEN SIGNAL SQLSTATE '45000'; /\* İstenmeyen koşul sağlanmadığında hata kodu gönderme. \*/

END IF;

END

$$ DELIMITER ;

/\* CONNECT CONTROL \*/

DELIMITER $$

CREATE TRIGGER status\_privacy\_control BEFORE INSERT ON share\_in\_group FOR EACH ROW

BEGIN

IF EXISTS (

SELECT \*

FROM connect AS f, connect AS s

WHERE f.account\_id = s.connecter\_id AND f.connecter\_id = s.account\_id )

THEN SIGNAL SQLSTATE '45000'; /\* İstenmeyen koşul sağlanmadığında hata kodu gönderme. \*/

END IF;

END

$$ DELIMITER ;

## İnsertion.sql

/\*Tablolara veri eklediğimiz kısımdır\*/

/\* APP LANGUAGE | NAME \*/

INSERT INTO app\_language VALUES ("Turkish");

INSERT INTO app\_language VALUES ("English");

INSERT INTO app\_language VALUES ("Russian");

INSERT INTO app\_language VALUES ("German");

INSERT INTO app\_language VALUES ("Spanish");

INSERT INTO app\_language VALUES ("Azerbaycan Türkçesi");

/\* ACCOUNT | ID, USER NAME, PASSWORD, NAME, MIDDLE NAME, SURNAME, GENDER, EMAIL, CREATE DATE, PREMIUM, LAST ACCESS, ACCOUNT TYPE, USER TYPE, LANGUAGE, FOLLOWER \*/

INSERT INTO account\_ VALUES (1, "ekinaslan", "aslanekin97", "Ekin", NULL, "Aslan", "Male", "ekinaslan.js@gmail.com", "2022-01-29", TRUE, NULL, 0, 1, "Turkish", DEFAULT);

INSERT INTO account\_ VALUES (2, "aliilman", "ilmanali123", "Ali", NULL, "İlman", "Male", "aliilman48@gmail.com", "2022-01-29", TRUE, NULL, 0, 1, "Turkish", DEFAULT);

INSERT INTO account\_ VALUES (3, "esraduman", "mcdonalds78", "Esra", NULL, "Duman", "Female", "esradumann11@gmail.com", "2022-01-29", TRUE, NULL, 0, 1, "Turkish", DEFAULT);

INSERT INTO account\_ VALUES (4, "muratosman", "ilovedatabase", "Murat", "Osman", "Ünalır", "Male", "unalir@gmail.com", "2022-01-29", TRUE, NULL, 0, 2, "Turkish", DEFAULT);

INSERT INTO account\_ VALUES (5, "codd", "relationalmodel", "Edgar Frank", "Ted", "Codd", "Male", "edgarfcodd@gmail.com", "2022-01-29", FALSE, NULL, 0, 0, "English", DEFAULT);

INSERT INTO account\_ VALUES (6, "egeuniversitesi", "123456", "Ege Üniversitesi", NULL, NULL, "None", "mail@ege.edu.tr", "2022-01-29", TRUE, NULL, 1, NULL, "Turkish", DEFAULT);

INSERT INTO account\_ VALUES (7, "bilmuh", "while1=1", "Bilgisayar Mühendisliği", NULL, NULL, "None", "bilmuh@ege.edu.tr", "2022-01-29", TRUE, NULL, 1, NULL, "Turkish", DEFAULT);

/\* LOCATION | NAME, CITY, COUNRTY, GMT, ADRESS \*/

INSERT INTO location VALUES ("Ekin Ev", "İzmir", "Turkey", 3, "Mevlana 1751/1");

INSERT INTO location VALUES ("Öğrenci Köyü", "İzmir", "Turkey", 3, "Ege Üniversitesi Kampüsü Öğrenci Köyü");

/\* GROUP | ID, NAME, PRIVATE, DESC, RULES, CREATOR ID \*/

INSERT INTO group\_ VALUES (1, "Database Project", TRUE, "Our Database Project Group", "1. Database Hakkında Konuşmak Yasak 2. Database Hakkında Kesinlikle Konuşmak Yasak", 1);

/\* LOCATION | ID, NAME, CREATE DATE, DEADLINE, APPLY COUNT, INDUSTRY, EMPLOYMENT TYPE, SENIORITY LEVEL, FUNCTIONS, DESC CREATOR ID \*/

INSERT INTO job\_offer VALUES (1, "Proje Yaptırma", "2022-01-29", "2022-02-02", DEFAULT, "Computer Engineering", "Freelance", "Junior", "Database", "Database bilen birini arıyoruz.", 2);

/\* TEST\_SCORE | ID, NAME, SCORE, DATE, DESC, ACCOUNT ID \*/

INSERT INTO test\_score VALUES (1, "Unity", 90, "2021-12-12", "Unity With DOTween", 1);

INSERT INTO test\_score VALUES (2, "Syber Security Analysis", 90, "2021-12-12", "Blue Team Eğitimi", 2);

/\* CERTIFICATE | ID, NAME, URL \*/

INSERT INTO certificate VALUES (1, "Database Management Certificate", "ege.edu.tr");

/\* PROJECT | ID, NAME, DESC, URL, START DATE, END DATE \*/

INSERT INTO project VALUES (1, "Database Management Term Project", "Creating new database with LinkedIn and Moddle.", NULL, "2021-12-01", "2022-02-02");

/\* CERTIFICATE | ID, NAME, ANNOUNCEMENT, CERTIFICATE ID, ACCOUNT ID \*/

INSERT INTO course VALUES (1, "Database Management", NULL, 1, 7);

/\* CONTENT | NAME, DESC, DOCUMENT, END TIME, DOCUMENT TYPE, TODO, COURSE ID \*/

INSERT INTO content VALUES ("EER Model", "EER Model Konu Anlatımı", "eermodel.pdf", NULL, "PDF", TRUE, 1);

INSERT INTO content VALUES ("SQL", "SQL Konu Anlatımı", "sql.mp4", NULL, "MP4", TRUE, 1);

/\* CERTIFICATE | ID, URL \*/

INSERT INTO media VALUES (1, "linkedinmoodle.com/profile/default.png");

INSERT INTO media VALUES (2, "linkedinmoodle.com/profile/ekin.png");

INSERT INTO media VALUES (3, "linkedinmoodle.com/profile/esra.png");

/\* STATUS | ID, TEXT, DATE, TYPE, LIKES, COMMENTS ACCOUNT ID \*/

INSERT INTO status\_ VALUES (1, "Bu benim ilk paylaşımım.", "2022-01-29", NULL, DEFAULT, DEFAULT, 2);

INSERT INTO status\_ VALUES (2, "DATA IS THE NEW OIL!", "2022-01-29", NULL, DEFAULT, DEFAULT, 4);

INSERT INTO status\_ VALUES (3, "Asla pes etme!", "2022-01-29", 0, DEFAULT, DEFAULT, 1);

INSERT INTO status\_ VALUES (4, "Kariyer Toplantısı", "2022-01-29", 1, DEFAULT, DEFAULT, 4);

INSERT INTO status\_ VALUES (5, NULL, "2022-01-29", 2, DEFAULT, DEFAULT, 3);

INSERT INTO status\_ VALUES (6, "İlk grup paylaşımı", "2022-01-29", NULL, DEFAULT, DEFAULT, 3);

INSERT INTO status\_ VALUES (7, "Ege Üniversitesi'nin yapmış olduğu...", "2022-02-02", NULL, DEFAULT, DEFAULT, 7);

/\* ABILITY | ID, LANGUAGE, LANGUAGE LEVEL, PATENT NAME, BADGE NAME, HONOR AWARD NAME, ACCOUNT ID \*/

INSERT INTO ability VALUES (1, "Russian", "A1", NULL, NULL, NULL, 1);

INSERT INTO ability VALUES (2, "English", "B1", NULL, NULL, "Satranç Derece Ödülü", 2);

INSERT INTO ability VALUES (3, "English", "C2", NULL, NULL, NULL, 3);

/\* VIDEO | URL, STATUS ID \*/

INSERT INTO video VALUES ("linkedinmoodle.com/videos/motivasyon.mp4", 3);

/\* EVENT | URL, STATUD ID \*/

INSERT INTO event\_ VALUES ("teams.microsoft.com/joinid123", 4);

/\* IMAGE | URL, STATUS ID \*/

INSERT INTO image VALUES ("linkedinmoodle.com/images/autumn.jpg", 5);

INSERT INTO user\_makes\_project VALUES (1, 1);

INSERT INTO user\_makes\_project VALUES (2, 1);

INSERT INTO user\_makes\_project VALUES (3, 1);

/\* SHARE IN GROUP | STATUS ID, GROUP ID \*/

INSERT INTO share\_in\_group VALUES (6, 1);

/\* ILLEGAL BUT VALID! \*/ /\* INSERT INTO share\_in\_group VALUES (2, 1); \*/

/\* ACCOUNT HAS LOCATION | ACCOUNT ID, LOCATION NAME \*/

INSERT INTO account\_has\_location VALUES (1, "Ekin Ev");

INSERT INTO account\_has\_location VALUES (2, "Öğrenci Köyü");

INSERT INTO account\_has\_location VALUES (3, "Öğrenci Köyü");

/\* ACCOUNT BOOKMARK STATUS | ACCOUNT ID, STATUS ID, \*/

INSERT INTO account\_bookmark\_status VALUES (3, 6);

/\* ACCOUNT COMMENT STATUS | ACCOUNT ID, STATUS ID, COMMENT, LIKE \*/

INSERT INTO account\_comment\_status VALUES (1, 5, "Çok güzel sonbahar fotoğrafı.", 1);

/\* ACCOUNT LIKE STATUS | ACCOUNT ID, STATUS ID, TYPE \*/

INSERT INTO account\_like\_status VALUES (2, 5, 1);

INSERT INTO account\_like\_status VALUES (1, 5, 0);

INSERT INTO account\_like\_status VALUES (1, 2, 2);

/\* ACCOUNT MANAGE GROUP | ACCOUNT ID, GROUP ID \*/

INSERT INTO account\_manage\_group VALUES (2, 1);

/\* USER APPLY JOB | ACCOUNT ID, JOB OFFER ID \*/

INSERT INTO user\_apply\_job VALUES (5, 1);

/\* ACCOUNT APPROVE TEST SCORE | ACCOUNT ID, TEST SCORE ID \*/

INSERT INTO account\_approve\_test\_score VALUES (4, 2);

/\* USER HAS CERTIFICATE | ACCOUNT ID, CERTIFICATE ID, ISSUE, EXPIRATION \*/

INSERT INTO user\_has\_certificate VALUES (1, 1, "2022-02-02", NULL);

INSERT INTO user\_has\_certificate VALUES (2, 1, "2022-02-02", NULL);

INSERT INTO user\_has\_certificate VALUES (3, 1, "2022-02-02", NULL);

/\* MESSAGE | SENDER ID, RECEIVER ID, TEXT, DATE \*/

INSERT INTO message VALUES (1, 2, "Kanka projeyi yapıcak adam bulamadın mı :D", "2022-01-29");

INSERT INTO message VALUES (2, 1, "Çok önemli birini buldum.", "2022-01-30");

/\* VIEW PROFILE | ACCOUNT ID, VIEWED ACCOUNT ID \*/

INSERT INTO view\_profile VALUES (3, 4);

/\* CONNECT | ID, ID \*/

INSERT INTO connect VALUES (1, 4);

INSERT INTO connect VALUES (2, 4);

INSERT INTO connect VALUES (3, 4);

INSERT INTO connect VALUES (1, 2);

INSERT INTO connect VALUES (1, 3);

INSERT INTO connect VALUES (2, 3);

INSERT INTO connect VALUES (2, 7);

INSERT INTO connect VALUES (3, 7);

/\* ILLEGAL! \*/ /\* INSERT INTO connect VALUES (2, 1); \*/

/\* ACCOUNT HAS MEDIA | ACCOUNT ID, PROFILE PHOTO ID \*/

INSERT INTO account\_has\_media VALUES (1, 2);

INSERT INTO account\_has\_media VALUES (2, 1);

INSERT INTO account\_has\_media VALUES (3, 3);

/\* STUDENT TAKES COURSE | ACCOUNT ID, COURSE ID, NOTE \*/

INSERT INTO student\_takes\_course VALUES (1, 1, 90);

INSERT INTO student\_takes\_course VALUES (2, 1, 100);

INSERT INTO student\_takes\_course VALUES (3, 1, 95);

/\* INSTRUCTOR TEACHES COURSE | ACCOUNT ID, COURSE ID \*/

INSERT INTO instructor\_teaches\_course VALUES (4, 1);

/\* INSTRUCTOR TEACHES COURSE | ACCOUNT ID, COURSE ID \*/

INSERT INTO account\_join\_group VALUES (1, 1);

INSERT INTO account\_join\_group VALUES (2, 1);

INSERT INTO account\_join\_group VALUES (3, 1);

## Constraints.sql

/\* Constraints yazdığımız kısımdır\*/

/\* CERTIFICATE DATE CONTROL \*/

ALTER TABLE user\_has\_certificate

ADD CONSTRAINT certificate\_date\_control

CHECK (expiration\_date >= issue\_date);

/\* GENDER CONTROL \*/

ALTER TABLE account\_

ADD CONSTRAINT gender\_control

CHECK (gender IN ('Male', 'Female', 'Other', 'None'));

/\* JOB OFFER DATE CONTROL \*/

ALTER TABLE job\_offer

ADD CONSTRAINT job\_offer\_date\_control

CHECK (deadline >= create\_date);

/\* ALTER TABLE student\_takes\_course

ADD CONSTRAINT student\_note\_control

CHECK (100 >= note); \*/

## Views.Sql

/\* View tanımladığız kısımdır\*/

/\* NORMAL ACCOUNTS \*/

CREATE VIEW normals AS

SELECT \*

FROM account\_

WHERE account\_type = 0 AND user\_type = 0;

/\* STUDENT ACCOUNTS \*/

CREATE VIEW students AS

SELECT \*

FROM account\_

WHERE account\_type = 0 AND user\_type = 1;

/\* INSTRUCTOR ACCOUNTS \*/

CREATE VIEW instructors AS

SELECT \*

FROM account\_

WHERE account\_type = 0 AND user\_type = 2;

/\* COMPANY ACCOUNTS \*/

CREATE VIEW companies AS

SELECT \*

FROM account\_

WHERE account\_type = 1;

/\* ALL USER ACCOUNTS \*/

CREATE VIEW users AS

SELECT \*

FROM account\_

WHERE account\_type = 0;

## Queries.sql

/\*10. Soru için bizden istenen sorguları yazdığımız kısımdır\*/

/\* TEKLİ TABLOLAR \*/

/\* Tüm premium hesaplar. \*/

SELECT \*

FROM account\_

WHERE premium;

/\* Tüm gönderiler. \*/

SELECT \*

FROM status\_;

/\* Tüm gruplar \*/

SELECT g.group\_name AS name\_, g.privacy\_type AS is\_private, g.create\_account\_id AS creator\_id

FROM group\_ AS g;

/\* İKİLİ TABLOLAR \*/

/\* Kimler takipleşiyor. \*/

SELECT f.user\_name, f.first\_name, f.last\_name, s.user\_name, s.first\_name, s.last\_name

FROM account\_ AS f, account\_ AS s, connect AS c

WHERE f.account\_id = c.account\_id AND s.account\_id = c.connecter\_id;

/\* Herhangi bir hesabın takipçileri \*/

SELECT f.user\_name, f.first\_name, f.last\_name, s.user\_name, s.first\_name, s.last\_name

FROM account\_ AS f, account\_ AS s, connect AS c

WHERE f.user\_name = 'aliilman' AND ((f.account\_id = c.account\_id AND s.account\_id = c.connecter\_id) OR (f.account\_id = c.connecter\_id AND s.account\_id = c.account\_id));

/\* Profili görüntüleyen hesaplar. \*/

SELECT f.first\_name AS viewed\_name, f.last\_name AS viewed\_surname, s.first\_name AS viewing\_name, s.last\_name AS viewing\_surname

FROM account\_ AS f, account\_ AS s, view\_profile AS v

WHERE f.user\_name = "muratosman" AND v.viewed\_id = f.account\_id AND v.account\_id = s.account\_id;

/\* Herhangi iki kişi arasındaki mesajlar. \*/

SELECT f.user\_name AS from\_, s.user\_name AS to\_, m.text\_message AS message, m.message\_date AS time

FROM account\_ AS f, account\_ AS s, message AS m

WHERE f.user\_name = "ekinaslan" AND f.user\_name = "aliilman" AND (m.sender\_id = f.account\_id AND m.receiver\_id = s.account\_id) OR (m.sender\_id = s.account\_id AND m.receiver\_id = f.account\_id)

ORDER BY time;

/\* ÜÇLÜ TABLOLAR \*/

/\* Kişilerin beğendiği durumların listesi.\*/

SELECT a.user\_name, s.text\_, s.status\_id

FROM account\_like\_status AS als, account\_ AS a, status\_ AS s

WHERE als.account\_id = a.account\_id AND als.status\_id = s.status\_id;

/\* Öğrencilerin aldığı kursların listesi. \*/

SELECT a.first\_name, a.last\_name, c.course\_name, s.note

FROM instructor\_teaches\_course AS i, student\_takes\_course AS s, account\_ AS a, course AS c

WHERE i.course\_id = s.course\_id AND s.account\_id = a.account\_id AND c.course\_id = s.course\_id;

/\* Herhangi bir hesabın akışı. (Grup paylaşımları dahil değil.) \*/

SELECT f.user\_name AS flow\_account, l.first\_name AS sharer\_name, l.last\_name AS sharer\_surname, s.text\_, s.status\_date AS date\_, s.status\_id

FROM account\_ AS f

JOIN status\_ AS s

JOIN connect AS c ON (c.account\_id = f.account\_id AND c.connecter\_id = s.account\_id) OR (c.account\_id = s.account\_id AND c.connecter\_id = f.account\_id)

JOIN account\_ AS l ON l.account\_id = s.account\_id

JOIN share\_in\_group AS sig ON sig.status\_id != s.status\_id

WHERE f.user\_name = "esraduman"

ORDER BY date\_;

/\* KRİTİK TABLOLAR \*/

/\* Zayıf şifreleri bulma. \*/

SELECT \*

FROM account\_

WHERE user\_password LIKE "%123%";

/\* Belirli tarihlerdeki gönderiler. \*/

SELECT \*

FROM status\_

WHERE status\_date BETWEEN "2022-01-01" AND "2022-02-01";

/\* Belirli grubun mesajları. \*/

SELECT a.user\_name, s.text\_, s.status\_date

FROM status\_ AS s

JOIN share\_in\_group AS sig ON sig.status\_id = s.status\_id

JOIN account\_ AS a ON a.account\_id = s.account\_id

WHERE sig.group\_id = 1

ORDER BY s.status\_date;

/\* Premium hesap sayısı. \*/

SELECT COUNT(\*)

FROM account\_

WHERE premium;

/\* En yüksek, en düşük ve ortalama not. \*/

SELECT AVG(note) AS avarage, MIN(note) AS minimum, MAX(note) AS maximum

FROM student\_takes\_course;

/\* UPDATE, DELETE VE INSERT'LER \*/

/\* Not arttırma :) \*/

UPDATE student\_takes\_course

SET note = LEAST(note + 10, 100);

/\* Sertifika geçerliliğini değiştirme. \*/

UPDATE user\_has\_certificate

SET expiration\_date = "2025-01-01"

WHERE acount\_id = 3;

/\* Şifre değiştirme. \*/

UPDATE account\_

SET user\_password = "#BF0707"

WHERE user\_name = "ekinaslan";

/\* Hesap silme. \*/

SET FOREIGN\_KEY\_CHECKS = 0;

DELETE FROM account\_

WHERE account\_id = 5;

SET FOREIGN\_KEY\_CHECKS = 1;

/\* Sertifika silme. \*/

SET FOREIGN\_KEY\_CHECKS = 0;

DELETE FROM user\_has\_certificate

WHERE account\_id = 2 AND certificate\_id = 1;

SET FOREIGN\_KEY\_CHECKS = 1;

/\* Not silme. \*/

SET FOREIGN\_KEY\_CHECKS = 0;

DELETE FROM student\_takes\_course

WHERE account\_id = 3 AND course\_id = 1;

SET FOREIGN\_KEY\_CHECKS = 1;

/\* Hesap ekleme. \*/

INSERT INTO account\_ VALUES (5, "codd", "relationalmodel", "Edgar Frank", "Ted", "Codd", "Male", "edgarfcodd@gmail.com", "2022-01-29", FALSE, NULL, 0, 0, "English", DEFAULT);

/\* Sertifika ekleme. \*/

INSERT INTO user\_has\_certificate VALUES (2, 1, "2022-02-02", NULL);

/\* Not ekleme. \*/

INSERT INTO student\_takes\_course VALUES (3, 1, 100);