DATABASE MANAGEMENT TERM PROJECT

LinkedInMoodle

5180000879 - MOHAMMAD MAHDI SARHANGI
5180000038 - ECE TEK
5190000841 - VELİ YAŞAR

ANALYSIS

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

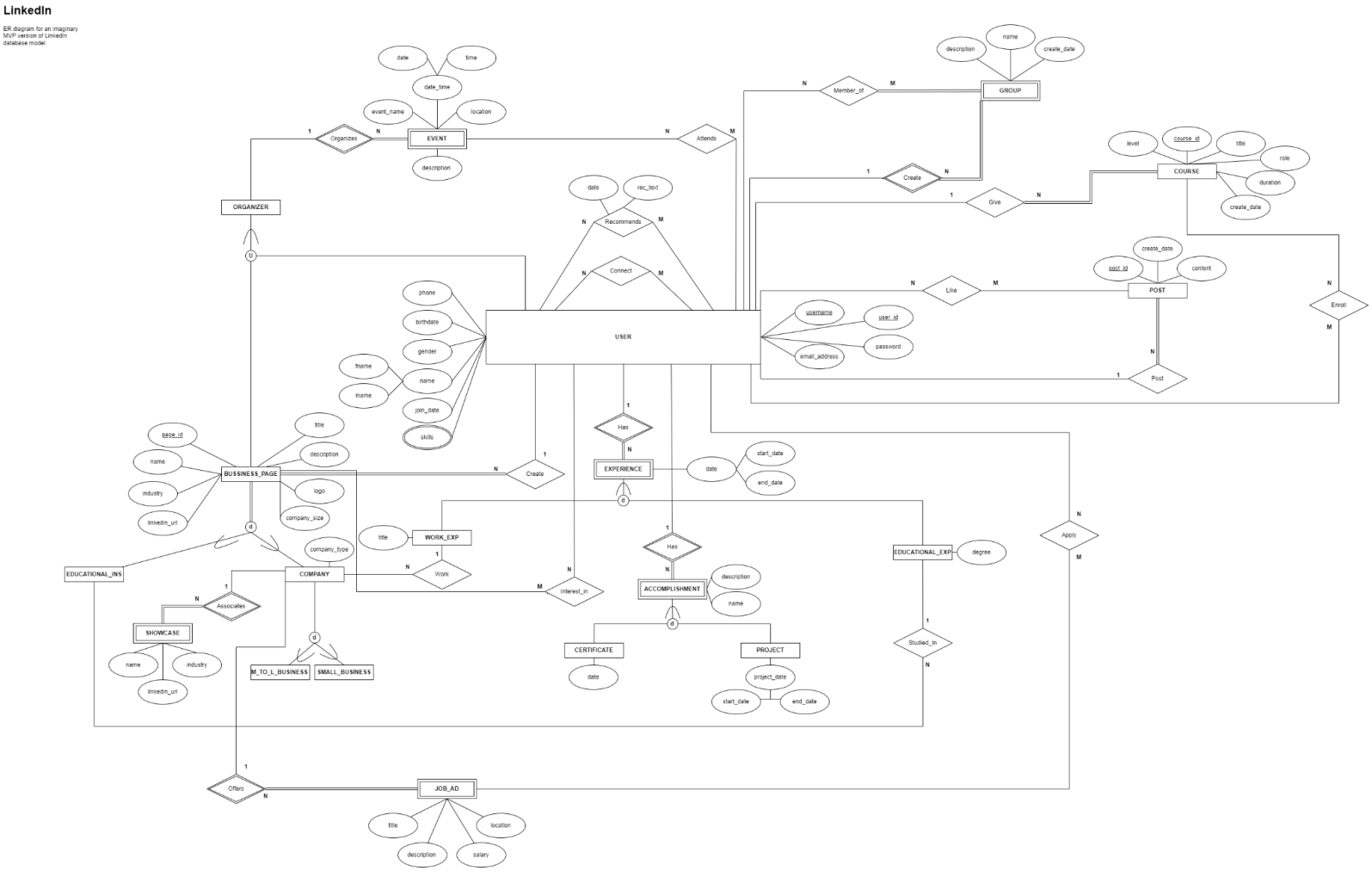
LinkedIn:

- LinkedIn is a social network used for professional networking and career development. It allows users to find jobs, post their CVs, hire employees, and more.
- LinkedIn allows users to connect with each other in order to make real-world professional relationships.
- On LinkedIn people can create their profile page and share their education and work experience with other people.
- LinkedIn provides users information about changes in businesses or job offers that they are interested in.
- LinkedIn provides job search/add feature for people who are searching for jobs or want to recruit professionals.

Moodle:

- Moodle is a learning platform used for e-learning purposes mostly for universities but it is also used in schools, workplaces, and other sectors.
- Moodle is focused on education. In Moodle, each university can have a separate page for each of its departments. Also, each department can have separate pages for their courses.
- Instructors are using Moodle to share course materials, initialize homework submit areas, and more.
- Its open-source structure allows communities to create required environments through plugins.

2. Write an analysis report for each web application:



a. What is the aim of each application? (LinkedIn)

- As LinkedIn says on its own page: “The mission of LinkedIn is simple: connect the world's professionals to make them more productive and successful.”
- LinkedIn aims to connect people with respect to their business and line of work. It has all features of a social network like:
  - Sharing a post
  - Having a profile page
  - Commenting
  - Like
  - Connecting with other users

- LinkedIn is designed for professional networking purposes so people have CV-like profiles that let them build a network with people who are relevant to their career.
- There are also company pages on LinkedIn.
- In general, LinkedIn aims to be a professional networking site with social network-like overtones.

b. What are the main entities of them? (LinkedIn)

ORGANIZER: An entity that is a combination of a user and a company.
EVENT: Represents an entity that is created by an organizer and attended by users.
USER: The most comprehensive entity of the diagram, represents a user of LinkedIn.
BUSINESS_PAGE: An entity that's created by a user in order to be either a company or an educational institute.
EDUCATIONAL_INSTITUTE: Represents educational institutions.
SHOWCASE: Represents the display of companies.
COMPANY: A broad entity that represents businesses of all sizes and types.
M_TO_L_BUSINESS: Represents middle to large businesses/companies.
SMALL_BUSINESS: Represents small businesses/companies.
EXPERIENCE: Represents experiences of users on various grounds.
WORK_EXPERIENCE: Represents occupational experiences.
EDUCATIONAL_EXPERIENCE: Represents educational experiences.
CERTIFICATE: An entity that's had by some users as an accomplishment.
PROJECT: An entity that represents past projects of users as an accomplishment.
ACCOMPLISHMENT: Represents achieved goals which can be a certificate or a project.
POST: An entity created and reacted by users in order to express their ideas.
COURSE: An entity where users enroll or give lessons.
GROUP: Represents a community that is created or joined by users.

c. What are the characteristics of each entity? (LinkedIn)

- USER
  - phone
  - birthdate
  - gender
  - fname
  - lname
  - join_date
  - skills
  - username
  - user_id
  - password
  - email_address
- BUSINESS_PAGE
  - title

        - description
        - logo
        - company_size
        - linkedin_url
        - industry
        - name
        - page_id
- COMPANY
    - company_type
- WORK_EXPERIENCE
    - title
- EDUCATIONAL_EXPERIENCE
    - degree
- CERTIFICATE
    - date
- PROJECT
    - start_date
    - end_date
- COURSE
    - course_id
    - level
    - title
    - role
    - duration
    - create_date
- POST
    - create_date
    - post_id
    - content

d. What relationships exist among the entities? (LinkedIn)

- ORGANIZER > Organizes > EVENT
- WORK_EXPERIENCE > Works > COMPANY
- COMPANY > Associates > SHOWCASE
- COMPANY > Offers > JOB_AD
- EDUCATIONAL_EXP > Studies_In > EDUCATIONAL_INS
- USER > Has > EXPERIENCE
- USER > Has > ACCOMPLISHMENT
- USER > Creates > BUSINESS_PAGE
- USER > Interests_In > BUSINESS_PAGE

- USER > Attends > EVENT
- USER > Recommends > USER
- USER > Connects > USER
- USER > Member_Of > GROUP
- USER > Creates > GROUP
- USER > Gives > COURSE
- USER > Enrolls > COURSE
- USER > Posts > Post
- USER > Likes > Post
- USER > Applies > JOB_AD

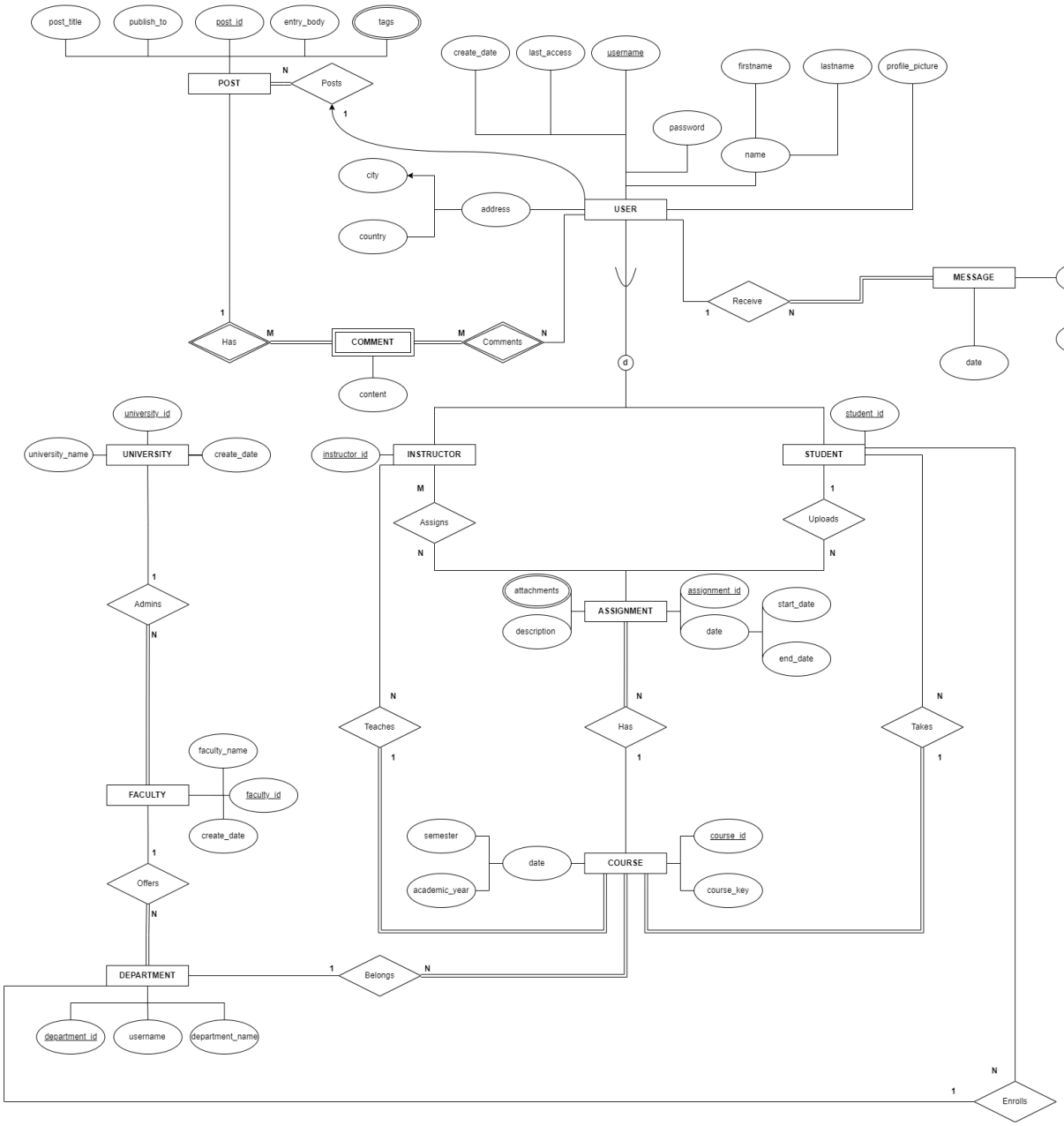
e. What are the constraints related to entities, their characteristics, and the relationships among them? (LinkedIn)

- An Experience must have a User
- A Post requires a User
- A Course must have a User
- A Group must have User
- An Event has to have an Organizer
- A Businesss_Page must be either a Company or an Educational_Institute
- A Showcase must have a Company
- A Business_Page needs a User
- An Accomplishment must have a User
- An Accomplishment can either be a Certificate or a Project .
- An Experience can either be a Work_Experience or an Educational_Experience
- A Company can either be a Middle_to_Large_Business or a Small_Business
- A Job_Ad requires a Company
- A Business_Page and a User unions to an Organizer

2. Write an analysis report for each web application:

Moodle

The graph in the right side is representation of MVP version of Moodle database system.



a. What is the aim of each application? (Moodle)

- As Moodle describes itself: “Moodle is a learning platform designed to provide educators, administrators, and learners with a single robust, secure and integrated system to create personalized learning environments.”

- Moodle is an open-source management system. People should sign up and then log in to the system with the profile they created.
- Users can enroll in the courses provided by universities or organizations with an enroll key. After that, all of the documentation for the enrolled course can be accessible.
- The system lets users upload data to the course like assignments, projects, lecture notes and more.
- Moodle has a forum-like feature that lets users communicate with each other, such as students discussing details of an assignment with the teacher.

b. What are the main entities of them? (Moodle)

POST: An entity created and commented by users in order to express their ideas.
UNIVERSITY: Represents a university.
FACULTY: An entity that admins universities and is offered by departments.
DEPARTMENT: An entity with many courses and students.
COMMENT: Represents an entity that can be had by posts and made by users.
COURSE: An entity which has assignments, also taught by instructors and taken by students.
MESSAGE: An entity that's received by users.
ASSIGNMENT: Represents homeworks.
INSTRUCTOR: Represents a user that teaches courses.
STUDENT: Represents a user enrolled in a department.
USER: The most comprehensive entity that can either be an instructor or a student.

c. What are the characteristics of each entity? (Moodle)

- POST
  - post_title
  - publish_to
  - post_id
  - entry_body
  - tags

- COMMENT
  - content

- UNIVERSITY
  - university_id
  - university_name
  - create_date

- FACULTY
  - faculty_name
  - faculty_id

- create_date

- DEPARTMENT
  - department_id
  - department_name
  - username

- COURSE
  - course_id
  - course_key
  - semester
  - acadamic_year

- ASSIGNMENT
  - assignment_id
  - start_date
  - end_date
  - attachments
  - description

- INSTRUCTOR
  - instructor_id

- STUDENT
  - student_id

- USER
  - create_date
  - last_access
  - username
  - password
  - name
  - firstname
  - lastname
  - profile_picture
  - city
  - country

- MESSAGE
  - message_id
  - content
  - date

d. What relationships exist among the entities? (Moodle)

- USER > Posts > POST
- COMMENT > Has > POST
- COMMENT > Comments > USER
- USER > Receives > Message
- ASSIGNMENT > Has > COURSE
- INSTRUCTOR > Teaches > COURSE
- STUDENT > Takes > COURSE
- STUDENT > Enrolls > DEPARTMENT
- DEPARTMENT > Offers > FACULTY
- FACULTY > Admins > UNIVERSITY
- INSTRUCTOR > Assigns > ASSIGNMENT
- STUDENT > Upload > ASSIGNMENT

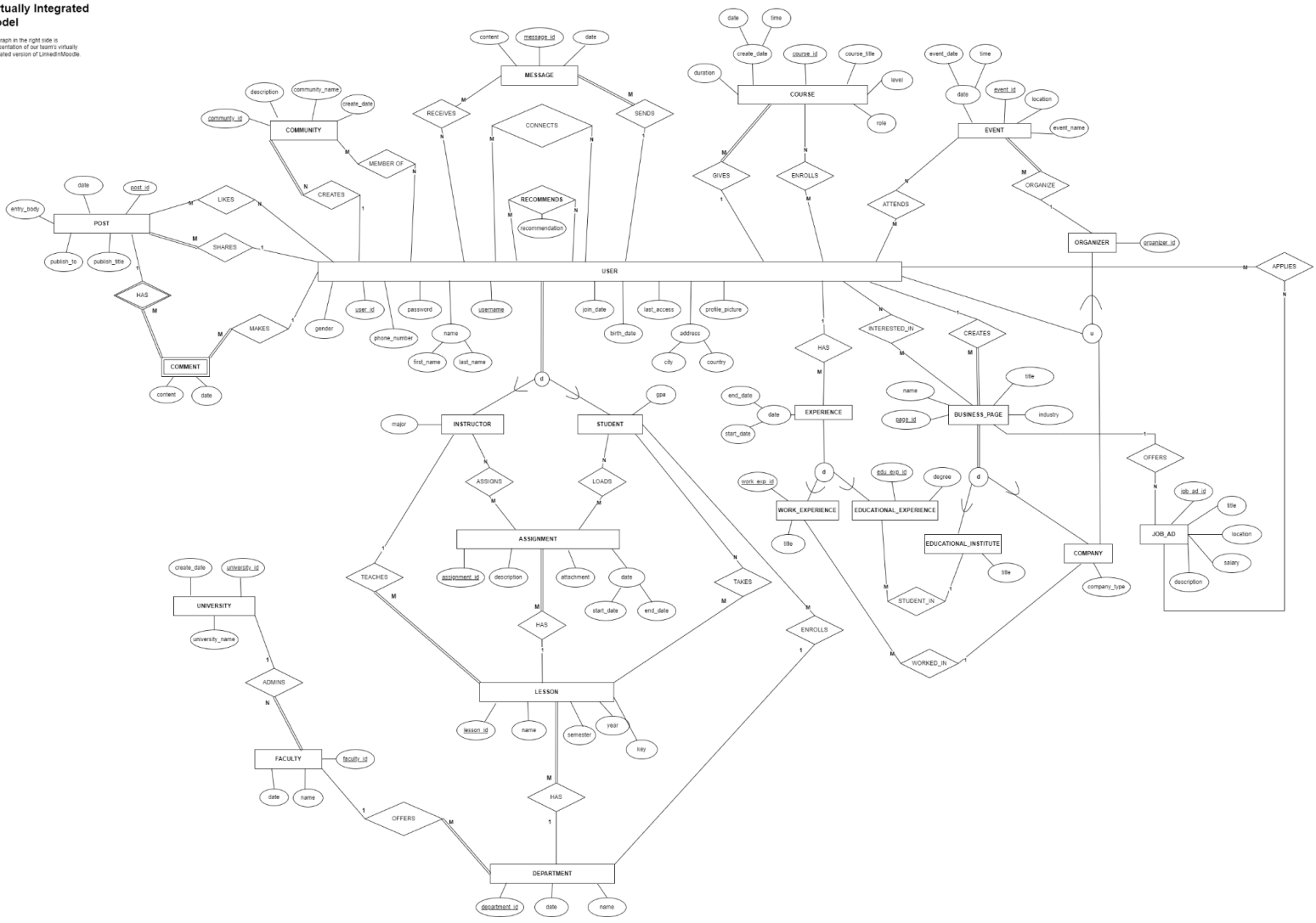
e. What are the constraints related to entities, their characteristics, and the relationships among them? (Moodle)

- A Post needs a User
- A Comment must have a Post and a User
- A Faculty needs a University
- A Department requires a Faculty
- A Course has to have a Department, an Instructor and a Student
- An Assignment must have a Course
- A Message must have a User
- A User can either be an Instructor or a Student

3. Create an EER diagram for the virtually integrated version of the applications, LinkedinMoodle.

Virtually Integrated Model

The graph in the right side is representation of our team's virtually integrated version of LinkedInMoodle.



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

1st Iteration

STEP 1:

POST(post_id, entry_body, publish_to, publish_title)
COMMUNITY(community_id, description, community_name, create_date)
MESSAGE(message_id, content, date)
COURSE(course_id, title, level, role, duration, date, time)
EVENT(event_id, date, time, location, name)
JOB_AD(job_ad_id, user_id, title, location, salary, description)
ASSIGNMENT(assignment_id, description, attachment, start_date, end_date)
LESSON(lesson_id, lesson_name, enrollment_key, semester, year)
UNIVERSITY(university_id, create_date, university_name)
FACULTY(faculty_id, date, name)
DEPARTMENT(department_id, date, name)
EVENT(event_id, create_date, time, organizer_id, location, name)

STEP 2:

COMMENT(post_id, user_id, create_time, content)

STEP 3:

No one-to-one relation

STEP 4:

FACULTY(..., university_id)
DEPARTMENT(..., faculty_id)
LESSON(..., department_id)
ASSIGNMENT(..., lesson_id)

STEP 5:

No many-to-many relation

STEP 6:

No multivalued attribute

STEP 7:

No N-ary relation

STEP 8A:

USER(user_id, username, gender, password, phone_number, first_name, last_name, city, country, birth_date, last_access, profile_photo, join_date)
STUDENT(user_id, gpa)
INSTRUCTOR(user_id, major)

2nd Iteration

STEP 4:

POST(..., user_id)
COMMUNITY(..., user_id)
MESSAGE(..., user_id)
COURSE(..., user_id)
STUDENT(..., department_id)
LESSON(..., user_id)

STEP 5:

POST_LIKE(post_id, user_id)
COMMUNITY_MEMBERS(group_id, user_id)
USER_CONNECTS(user_id, connected_id)
USER_RECOMMENDS(user_id, recommend_id, recommendation)
USER_ENROLLS(user_id, course_id)
USER_ATTENDS(event_id, user_id)
INS_ASSIGNS(instructor_id, assignment_id)
STD_LOADS(user_id, assignment_id)
TAKEN_LESSON(lesson_id, user_id)
USER_APPLY(user_id, job_ad_id)

STEP 6:

No multivalued attribute

STEP 7:

No N-ary relation

STEP 8A:

BUSINESS_PAGE(page_id, name, title, industry)
EDU_INSTITUTE(page_id)
COMPANY(page_id, company_type)

STEP 9:

USER(..., organizer_id)
COMPANY(..., organizer_id)
ORGANIZER(organizer_id)

3rd Iteration

STEP 4:
EDU_EXPERIENCE(edu_exp_id, page_id, degree)
WORK_EXPERIENCE(work_exp_id, page_id, title)
BUSINESS_PAGE(..., user_id)

STEP 5:
USER_APPLIES(user_id, job_ad_id)
USER_INTERESTS(user_id, page_id)

STEP 6:
STEP 7:
STEP 8:

STEP 9:
EDU_EXPERIENCE(edu_exp_id, page_id, degree)
WORK_EXPERIENCE(work_exp_id, page_id, title)
EXPERIENCE(exp_id)

4th Iteration

STEP 4:
EXPERIENCE(experience_id, user_id)

Results

ASSIGNMENT(assignment_id, description, attachment, start_date, end_date, *lesson_id*)
BUSINESS_PAGE(page_id, *user_id*, name, title, industry)
COMMENT(*post_id*, *user_id*, content, create_date)
COMMUNITY(community_id, description, community_name, create_date, *created_by*)
COMPANY(*page_id*, company_type)
COURSE(course_id, create_date, duration_minutes, course_title, course_level, rate, *given_by*)
DEPARTMENT(department_id, create_date, dep_name, *faculty_id*)
EDU_EXPERIENCE(edu_exp_id, degree, *page_id*)
EDU_INSTITUTE(*page_id*)
EVENT(event_id, create_date, time, organizer_id, location, event_name)
EXPERIENCE(experience_id, *user_id*)
FACULTY(faculty_id, create_date, faculty_name, *university_id*)

COMMUNITY_MEMBERS(*community_id, user_id*)
INS_ASSIGNS(*instructor_id, assignment_id*)
INSTRUCTOR(*instructor_id*, major)
JOB_AD(job_ad_id, title, location, salary, description, *page_id*)
LESSON(lesson_id, lesson_name, enrollment_key, lesson_name, semester, year, *department_id*, *instructor_id*)
MESSAGE(message_id, content, send_date, *sent_by, sent_to*)
ORGANIZER(organizer_id)
POST(post_id, entry_body, publish_title, *published_by*)
POST_LIKE(*post_id*, *user_id*)
STD_LOADS(*student_id, assignment_id*, attachment)
STUDENT(*student_id*, gpa, *department_id*)
TAKEN_LESSONS(*student_id, lesson_id*)
UNIVERSITY(university_id, create_date, univesity_name)
USER_APPLIES(*user_id, job_ad_id*)
USER_ATTENDS(*user_id, event_id*)
USER_ENROLLS (*user_id, course_id*)
USER_INTERESTS(*user_id, page_id*)
USER_RECOMMENDS(*recommender_id, recommended_id*, recommandation)
USER(user_id, username, gender, password, phone_number, first_name, last_name, city, country, birth_date, last_access, profile_photo, join_date)
WORK_EXPERIENCE(work_exp_id, title, *page_id*)

ASSIGNMENT

assignment_id	description	attachment	start_date	end_date	*lesson_id*

BUSINESS_PAGE

page_id	*user_id*	name	title	industry

COMMENT

post_id	*user_id*	content	create_date

COMMUNITY

community_id	description	community_name	create_date	*created_by*

COMPANY

page_id	company_type

COURSE

course_id	create_date	duration_minutes	course_title	course_level	rate
given_by	duration				

DEPARTMENT

department_id	create_date	dep_name	*faculty_id*

EDU_EXPERIENCE

edu_exp_id	degree	*page_id*

EDU_INSTITUTE

page_id

EVENT

event_id	create_date	time	organizer_id	location	event_name

EXPERIENCE

experience_id	*user_id*

FACULTY

faculty_id	create_date	faculty_name	*university_id*

COMMUNITY_MEMBERS

community_id	*user_id*

INS_ASSIGNS

instructor_id	*assignment_id*

INSTRUCTOR

instructor_id	major

JOB_AD

job_ad_id	title	location	salary	description	page_id

LESSON

lesson_id	lesson_name	enrollment_key	semester	year	*department_id*	*instructor_id*

MESSAGE

message_id	content	date	*sent_by*	*sent_to*

ORGANIZER

organizer_id

POST

post_id	entry_body	publish_title	*published_by*

POST_LIKE

post_id	user_id

STD_LOADS

student_id	*assignment_id*	attachment

STUDENT

student_id	gpa	*department_id*

TAKEN_LESSONS

student_id	*lesson_id*

UNIVERSITY

university_id	create_date	university_name

USER_APPLIES

user_id	*job_ad_id*

USER_ATTENDS

user_id	*event_id*

USER_ENROLLS

user_id	*course_id*

USER_INTERESTS

user_id	*page_id*

USER_RECOMMENDS

recommender_id	*recommended_id*	recommandation

USER

user_id	username	gender	password	phone_number	first_name	last_name
city	country	birth_date	last_access	profile_photo	join_date	

WORK_EXPERIENCE

work_exp_id	title	*page_id*

6. Write down the appropriate SQL scripts (DDL statements) for creating the database and its relational model.

CREATE TABLE **User**(
user_id int NOT NULL,
username varchar(50) NOT NULL,
gender varchar(50),
password varchar(50) NOT NULL,
phone_number varchar(50) NOT NULL,
first_name varchar(50) NOT NULL,
last_name varchar(50) NOT NULL,
city varchar(50) NOT NULL,
country varchar(50) NOT NULL,
birth_date date,
last_access date NOT NULL,
profile_photo varchar(50),
join_date date NOT NULL,
UNIQUE(username),
PRIMARY KEY(user_id));

CREATE TABLE **Course**(
course_id int NOT NULL,

```
create_date date NOT NULL,
duration_minutes int,
course_title varchar(50) NOT NULL,
course_level varchar(20) NOT NULL,
rate double,
given_by int NOT NULL,
CONSTRAINT user_id_fk_course FOREIGN KEY (given_by) references User(user_id) ON
DELETE CASCADE ON UPDATE CASCADE,
PRIMARY KEY(course_id));
```

```
CREATE TABLE Post(
post_id int NOT NULL,
entry_body varchar(100) NOT NULL,
publish_title varchar(50) NOT NULL,
published_by int NOT NULL,
CONSTRAINT  user_id_fk_post FOREIGN KEY(published_by) references User(user_id) ON
DELETE CASCADE ON UPDATE CASCADE,
PRIMARY KEY(post_id));
```

```
CREATE TABLE Community(
community_id int NOT NULL,
description varchar(50),
community_name varchar(50) NOT NULL,
create_date DATE NOT NULL,
created_by int NOT NULL,
CONSTRAINT user_id_fk_community FOREIGN KEY(created_by) references User(user_id)
ON DELETE CASCADE ON UPDATE CASCADE,
PRIMARY KEY (community_id));
```

```
CREATE TABLE Message(
message_id int NOT NULL,
content varchar(50) NOT NULL,
send_date DATE NOT NULL,
sent_by int NOT NULL,
sent_to int NOT NULL,
CONSTRAINT sender_id_fk_message FOREIGN KEY(sent_by) references User(user_id) ON
DELETE CASCADE ON UPDATE CASCADE,
CONSTRAINT reciever_id_fk_message FOREIGN KEY(sent_to) references User(user_id) ON
DELETE CASCADE ON UPDATE CASCADE,
PRIMARY KEY (message_id));
```

```
CREATE TABLE Event(
event_id int NOT NULL,
create_date DATE NOT NULL,
time DATETIME NOT NULL,
organizer_id int NOT NULL,
location varchar(50) NOT NULL,
event_name varchar(50) NOT NULL,
CONSTRAINT chk_EventDate CHECK (time > create_date),
PRIMARY KEY (event_id));


CREATE TABLE Job_Ad(
job_ad_id int NOT NULL,
title varchar(50) NOT NULL,
location varchar(50) NOT NULL,
salary double,
description varchar(50) NOT NULL,
page_id int NOT NULL,
PRIMARY KEY (job_ad_id));


CREATE TABLE University(
university_id int NOT NULL,
create_date DATE NOT NULL,
university_name varchar(50) NOT NULL,
PRIMARY KEY (university_id));


CREATE TABLE Faculty(
faculty_id int NOT NULL,
create_date DATE NOT NULL,
faculty_name varchar(50) NOT NULL,
university_id int NOT NULL,
CONSTRAINT faculty_uni_fk FOREIGN KEY(university_id) references University(university_id)
ON DELETE CASCADE ON UPDATE CASCADE,
PRIMARY KEY(faculty_id));


CREATE TABLE Department(
department_id int NOT NULL,
create_date DATE NOT NULL,
dep_name varchar(50) NOT NULL,
faculty_id int NOT NULL,
```

```
CONSTRAINT dep_fac_fk FOREIGN KEY(faculty_id) references Faculty(faculty_id) ON
DELETE CASCADE ON UPDATE CASCADE,
PRIMARY KEY(department_id));


CREATE TABLE Student(
student_id int NOT NULL,
gpa double,
department_id int NOT NULL,
CONSTRAINT user_id_fk_student FOREIGN KEY(student_id) references User(user_id) ON
DELETE CASCADE ON UPDATE CASCADE,
CONSTRAINT stud_dep_fk FOREIGN KEY(department_id) references
Department(department_id) ON DELETE CASCADE ON UPDATE CASCADE,
PRIMARY KEY (student_id));


CREATE TABLE Instructor(
instructor_id int NOT NULL,
major varchar(50) NOT NULL,
CONSTRAINT user_id_fk_instructor FOREIGN KEY(instructor_id) references User(user_id) ON
DELETE CASCADE ON UPDATE CASCADE,
PRIMARY KEY (instructor_id));


CREATE TABLE Lesson(
lesson_id int NOT NULL,
enrollment_key int,
lesson_name varchar(50) NOT NULL,
semester varchar(50) NOT NULL,
year int NOT NULL,
department_id int NOT NULL,
instructor_id int NOT NULL,
UNIQUE(enrollment_key),
CONSTRAINT lesson_dept_fk FOREIGN KEY(department_id) references
DEPARTMENT(department_id) ON DELETE CASCADE ON UPDATE CASCADE,
CONSTRAINT lesson_inst_fk FOREIGN KEY(instructor_id) references Instructor(instructor_id)
ON DELETE CASCADE ON UPDATE CASCADE,
PRIMARY KEY (lesson_id));


CREATE TABLE Assignment(
assignment_id int NOT NULL,
description varchar(50),
attachment varchar(50) NOT NULL,
```

```
start_date DATETIME NOT NULL,
end_date DATETIME NOT NULL,
lesson_id int NOT NULL,
CONSTRAINT assignment_lesson_fk FOREIGN KEY(lesson_id) references Lesson(lesson_id)
ON DELETE CASCADE ON UPDATE CASCADE,
CONSTRAINT chk_Date CHECK (end_date > start_date),
PRIMARY KEY (assignment_id));


CREATE TABLE Comment(
post_id int NOT NULL,
user_id int NOT NULL,
content varchar(50) NOT NULL,
create_date DATETIME NOT NULL,
CONSTRAINT comment_post_fk FOREIGN KEY(post_id) references Post(post_id) ON
DELETE CASCADE ON UPDATE CASCADE,
CONSTRAINT user_id_fk_comment FOREIGN KEY(user_id) references User(user_id) ON
DELETE CASCADE ON UPDATE CASCADE,
PRIMARY KEY(user_id, post_id, create_date));


CREATE TABLE Post_Like(
post_id int NOT NULL,
user_id int NOT NULL,
CONSTRAINT postlike_postid_fk FOREIGN KEY(post_id) references Post(post_id) ON
DELETE CASCADE ON UPDATE CASCADE,
CONSTRAINT user_id_fk_postlike FOREIGN KEY(user_id) references User(user_id) ON
DELETE CASCADE ON UPDATE CASCADE,
PRIMARY KEY(user_id, post_id));


CREATE TABLE Community_Members(
community_id int NOT NULL,
user_id int NOT NULL,
CONSTRAINT commember_commid_fk FOREIGN KEY(community_id) references
Community(community_id) ON DELETE CASCADE ON UPDATE CASCADE,
CONSTRAINT user_id_fk_communitymem FOREIGN KEY(user_id) references User(user_id)
ON DELETE CASCADE ON UPDATE CASCADE,
PRIMARY KEY(user_id, community_id));


CREATE TABLE User_Recommends(
recommender_id int NOT NULL,
recommended_id int NOT NULL,
```

recommendation varchar(50),
CONSTRAINT user_id_fk_userrecom_1 FOREIGN KEY(recommended_id) references User(user_id) ON DELETE CASCADE ON UPDATE CASCADE,
CONSTRAINT user_id_fk_userrecom_2 FOREIGN KEY(recommender_id) references User(user_id) ON DELETE CASCADE ON UPDATE CASCADE,
PRIMARY KEY(recommended_id,recommender_id));

CREATE TABLE **User_Enrolls**(
user_id int NOT NULL,
course_id int NOT NULL,
CONSTRAINT user_id_fk_userenroll FOREIGN KEY(user_id) references User(user_id) ON DELETE CASCADE ON UPDATE CASCADE,
CONSTRAINT userenroll_courseid_fk FOREIGN KEY(course_id) references Course(course_id) ON DELETE CASCADE ON UPDATE CASCADE,
PRIMARY KEY (user_id, course_id));

CREATE TABLE **User_Attends**(
user_id int NOT NULL,
event_id int NOT NULL,
CONSTRAINT user_id_fk_userattends FOREIGN KEY(user_id) references User(user_id) ON DELETE CASCADE ON UPDATE CASCADE,
CONSTRAINT userattend_eventid_fk FOREIGN KEY(event_id) references Event(event_id) ON DELETE CASCADE ON UPDATE CASCADE,
PRIMARY KEY (user_id, event_id));

CREATE TABLE **Ins_Assigns**(
instructor_id int NOT NULL,
assignment_id int NOT NULL,
CONSTRAINT insassign_instructor_fk FOREIGN KEY(instructor_id) references Instructor(instructor_id) ON DELETE CASCADE ON UPDATE CASCADE,
CONSTRAINT insass_assid_fk FOREIGN KEY(assignment_id) references Assignment(assignment_id) ON DELETE CASCADE ON UPDATE CASCADE,
PRIMARY KEY (instructor_id, assignment_id));

CREATE TABLE **Std_Loads**(
student_id int NOT NULL,
assignment_id int NOT NULL,
attachment varchar(100) NOT NULL,
CONSTRAINT stdloads_studentid_fk FOREIGN KEY(student_id) references Student(student_id) ON DELETE CASCADE ON UPDATE CASCADE,

```
CONSTRAINT stdloads_assid_fk FOREIGN KEY(assignment_id) references
Assignment(assignment_id) ON DELETE CASCADE ON UPDATE CASCADE,
PRIMARY KEY (student_id, assignment_id));
```

```
CREATE TABLE Taken_Lessons(
student_id int NOT NULL,
lesson_id int NOT NULL,
CONSTRAINT takenlesson_stdid_fk FOREIGN KEY(student_id) references Student(student_id)
ON DELETE CASCADE ON UPDATE CASCADE,
CONSTRAINT takenlesson_lessid_fk FOREIGN KEY(lesson_id) references Lesson(lesson_id)
ON DELETE CASCADE ON UPDATE CASCADE,
PRIMARY KEY (student_id, lesson_id));
```

```
CREATE TABLE User_Applies(
user_id int NOT NULL,
job_ad_id int NOT NULL,
CONSTRAINT user_id_fk_userapplies FOREIGN KEY(user_id) references User(user_id) ON
DELETE CASCADE ON UPDATE CASCADE,
CONSTRAINT userapplies_jobid_fk FOREIGN KEY(job_ad_id) references Job_Ad(job_ad_id)
ON DELETE CASCADE ON UPDATE CASCADE,
PRIMARY KEY (user_id, job_ad_id));
```

```
CREATE TABLE Business_Page(
page_id int NOT NULL,
user_id int NOT NULL,
page_name varchar(50) NOT NULL,
title varchar(50),
industry varchar(50) NOT NULL,
CONSTRAINT user_id_fk_businesspage FOREIGN KEY(user_id) references User(user_id) ON
DELETE CASCADE ON UPDATE CASCADE,
PRIMARY KEY (page_id));
```

```
CREATE TABLE Edu_Institute(
page_id int NOT NULL,
CONSTRAINT eduinst_pageid_fk FOREIGN KEY(page_id) references
Business_Page(page_id) ON DELETE CASCADE ON UPDATE CASCADE,
PRIMARY KEY (page_id));
```

```
CREATE TABLE Company(
```

```
page_id int NOT NULL,
company_type varchar(50) NOT NULL,
CONSTRAINT company_businesspage_fk FOREIGN KEY(page_id) references
Business_Page(page_id) ON DELETE CASCADE ON UPDATE CASCADE,
PRIMARY KEY (page_id));


CREATE TABLE Organizer(
organizer_id int NOT NULL,
PRIMARY KEY (organizer_id));


CREATE TABLE Edu_Experience(
edu_exp_id int NOT NULL,
degree varchar(50) NOT NULL,
start_date DATE NOT NULL,
end_date DATE NOT NULL,
user_id int NOT NULL,
page_id int NOT NULL,
CONSTRAINT user_id_fk_eduexp FOREIGN KEY (user_id) REFERENCES User(user_id) ON
DELETE CASCADE ON UPDATE CASCADE,
CONSTRAINT eduexp_eduinst_fk FOREIGN KEY(page_id) references Edu_Institute(page_id)
ON DELETE CASCADE ON UPDATE CASCADE,
CONSTRAINT chk_EduDate CHECK (end_date> start_date),
PRIMARY KEY (edu_exp_id));


CREATE TABLE Work_Experience(
work_exp_id int NOT NULL,
title varchar(50) NOT NULL,
start_date DATE NOT NULL,
end_date DATE NOT NULL,
user_id int NOT NULL,
page_id int NOT NULL,
CONSTRAINT user_id_fk_workexp FOREIGN KEY (user_id) REFERENCES User(user_id) ON
DELETE CASCADE ON UPDATE CASCADE,
CONSTRAINT workexp_compid_fk FOREIGN KEY(page_id) references Company(page_id) ON
DELETE CASCADE ON UPDATE CASCADE,
CONSTRAINT chk_WorkDate CHECK (end_date> start_date),
PRIMARY KEY (work_exp_id));

-- Business_Page ve Organizer ancak şimdi oluştuğu için ALTER TABLE ile FOREIGN KEY'ler
burada ekleniyor
```

```
ALTER TABLE Job_Ad ADD CONSTRAINT jobadd_bussid_fk FOREIGN KEY (page_id)
references Business_Page (page_id) ON DELETE CASCADE ON UPDATE CASCADE;
ALTER TABLE Event ADD CONSTRAINT event_orgid_fk FOREIGN KEY (organizer_id)
references Organizer (organizer_id) ON DELETE CASCADE ON UPDATE CASCADE;


CREATE TABLE User_Interests(
user_id int NOT NULL,
page_id int NOT NULL,
CONSTRAINT user_id_fk_userinster FOREIGN KEY(user_id) references User(user_id) ON
DELETE CASCADE ON UPDATE CASCADE,
CONSTRAINT userinterests_companyid_fk FOREIGN KEY(page_id) references
Company(page_id) ON DELETE CASCADE ON UPDATE CASCADE,
PRIMARY KEY (user_id,page_id));


CREATE TABLE User_Work_Experiences(
work_exp_id int NOT NULL,
user_id int NOT NULL,
CONSTRAINT user_id_fk_userworkexp FOREIGN KEY (user_id) references User(user_id) ON
DELETE CASCADE ON UPDATE CASCADE,
CONSTRAINT userworkexp_workexpid_fk FOREIGN KEY(work_exp_id) references
Work_Experience(work_exp_id) ON DELETE CASCADE ON UPDATE CASCADE,
PRIMARY KEY (user_id,work_exp_id));


CREATE TABLE User_Edu_Experiences(
edu_exp_id int NOT NULL,
user_id int NOT NULL,
CONSTRAINT user_id_fk_usereduexp FOREIGN KEY (user_id) references User(user_id) ON
DELETE CASCADE ON UPDATE CASCADE,
CONSTRAINT usereduexp_eduexpid_fk FOREIGN KEY(edu_exp_id) REFERENCES
Edu_Experience(edu_exp_id) ON DELETE CASCADE ON UPDATE CASCADE,
PRIMARY KEY (user_id,edu_exp_id));


-- Alınan mesajlar INSERT yerine TRIGGER ile dolduruluyor
CREATE TABLE Recieved_Messages(
sender_id int DEFAULT 0,
reciever_id int DEFAULT 0,
content VARCHAR(50) DEFAULT ' ',
PRIMARY KEY (sender_id ,reciever_id , content));
```

7. Populate the database you just created again using SQL script file loaded with sample tuples.

```
INSERT INTO User values (101, 'burakylmz', 'male', 'password', '065605606',
'Burak','Yılmaz','Eskişehir','Turkey','1978-11-17', '2020-02-01', '/user.jpg', '2022-02-02');
INSERT INTO User values (102, 'eldergarlic', 'male', 'password-1', '05467981379',
'Mahdi','Sarhangi','Tehran','Iran','2000-02-07', '2021-01-22', '/user.jpg', '2022-01-01');
INSERT INTO User values (103, 'cece', 'female', 'password-2', '05350643899',
'Ece','Tek','Mersin','Turkey','1999-10-26', '2022-01-22', '/user.jpg', '2022-01-01');
INSERT INTO User values (104, 'velicious', 'male', 'password-3', '05316672404',
'Veli','Yasar','Manisa','Turkey','1997-05-29', '2022-01-22', '/user.jpg', '2022-01-08');
INSERT INTO User values (105, 'selinpaksoy35', 'female', 'password-4', '0511605606',
'Selin','Paksoy','Izmir','Turkey','2001-03-30', '2022-01-22', '/user.jpg', '2022-01-27');
INSERT INTO User values (106, 'murat.osma.unalir', 'male', 'password-5', '05325843535',
'Murat Osman','Unalir','Aydin','Turkey','1975-01-01', '2022-01-22', '/user.jpg', '2022-01-22');
INSERT INTO User values (107, 'seliamaksoy', 'male', 'password-6', '0511601106',
'Selim','Aksoy','Manisa','Turkey','1987-03-30', '2022-01-22', '/user.jpg', '2022-01-29');
INSERT INTO User values (108, 'emineplt', 'female', 'password-7', '05325843789',
'Emine','Polat','Aydin','Turkey','2000-06-06', '2022-01-22', '/user.jpg', '2022-01-29');

INSERT INTO Course values (1, '2022-01-22', 100, 'Mathematics', 'Easy', 7.2, 101);
INSERT INTO Course values (2, '2020-01-22', 120, 'Data Structors', 'Hard', 9.0, 105);
INSERT INTO Course values (3, '2019-01-22', 150, 'Database Management', 'Hard', 8.6, 106);
INSERT INTO Course values (4, '2022-01-25', 110, 'Statistics', 'Medium', 6.6, 107);
INSERT INTO Course values (5, '2022-01-27', 110, 'Digital Computer Design', 'Medium', 3.2,
108);

INSERT INTO Post values (1, 'Welcome to new semester!', 'Greeting Message!', 102);
INSERT INTO Post values (2, 'It is raining today!', 'Weather Information', 102);
INSERT INTO Post values (3, 'The homework is due tomorrow!', 'Homework Information', 106);
INSERT INTO Post values (4, 'Your exam is postponed!', 'Exam Information', 107);
INSERT INTO Post values (5, 'Looking for internship!', 'Job Info', 103);

INSERT INTO Community values(1, 'Alumni of Ege Univesity gather here!', 'Computer
Engineers', '2022-02-22', 106);
INSERT INTO Community values(2, 'Here we talk about tech!', 'Root Tech Community',
'2021-12-02', 101);
INSERT INTO Community values(3, 'We are always looking for the best talents!', 'Projects
Community', '2022-01-13', 107);
INSERT INTO Community values(4, 'Through engaging learning journeys!', 'Professional
Training', '2021-10-14', 108);

INSERT INTO Message values(1,'Selam Mahdi', '2022-01-22', 101,102);
```

```
INSERT INTO Message values(2,'Burak selam', '2022-01-22', 102,101);
INSERT INTO Message values(3,'how you doin veli?', '2022-01-22', 104,103);
INSERT INTO Message values(4,'im doin fine and you ece?', '2022-01-22', 103,104);
INSERT INTO Message values(5,'im fine too see you later', '2022-01-22', 104,103);
INSERT INTO Message values(6,'raporu da yazmadik aaaağğğğ!', '2022-01-22', 102,105);

INSERT INTO University values(1,'2022-01-29', 'Ege University');
INSERT INTO University values(2,'2020-02-19', 'Bogazici University');
INSERT INTO University values(3,'2021-03-27', 'Istanbul Technical University');

INSERT INTO Faculty values(1,'2022-01-29', 'Faculty of Engineering ',1);
INSERT INTO Faculty values(2,'2021-03-27', 'Faculty of Medicine',1);
INSERT INTO Faculty values(3,'2020-02-19', 'Faculty Nursing ',1);
INSERT INTO Faculty values(4,'2022-01-29', 'Faculty of Engineering ',2);
INSERT INTO Faculty values(5,'2021-03-27', 'Faculty of Medicine',3);
INSERT INTO Faculty values(6,'2020-02-19', 'Faculty Nursing ',3);

INSERT INTO Department values(1,'2022-01-22', 'Computer Engineering', 1);
INSERT INTO Department values(2,'2022-01-22', 'Chemical Engineering', 1);
INSERT INTO Department values(3,'2022-01-22', 'Electrical Engineering', 1);

INSERT INTO Student values(102,3.6,1);
INSERT INTO Student values(103,3.1,2);
INSERT INTO Student values(104,2.6,3);
INSERT INTO Student values(107,2.1,1);
INSERT INTO Student values(108,2.8,3);

INSERT INTO Instructor values(101, "Computer Science");
INSERT INTO Instructor values(106, "Database Management");
INSERT INTO Instructor values(105, "Statistics");

INSERT INTO Lesson values(1, 4545, "Algorithms","Spring",2022,1,101);
INSERT INTO Lesson values(2, 4546, "Database Management","Fall",2022,1,106);
INSERT INTO Lesson values(3, 4547,"Data Structures","Spring",2022,1,101);
INSERT INTO Lesson values(4, 4548,"Data Structures-2","Fall",2022,1,105);

INSERT INTO Assignment values(1, 'Database Homework', './test1.pdf', '2022-01-29 09:00:00',
'2022-01-31 23:30:00', 2);
INSERT INTO Assignment values(2, 'Database Homework', './test2.pdf', '2022-01-20 10:00:00',
'2022-01-28 23:59:00', 2);
INSERT INTO Assignment values(3, 'Algo-1 Term Project', './test3.pdf', '2022-01-20 10:00:00',
'2022-01-30 23:59:00', 2);

-- farklı zamanlarda aynı kullanıcı aynı yourumu yapabilir
```

```
INSERT INTO Comment values(2,102,'Great!','2022-01-29 13:10:01');
INSERT INTO Comment values(2,102,'Great!','2022-01-29 14:41:22');
INSERT INTO Comment values(3,105,'Good news!','2020-01-29 09:11:50');
INSERT INTO Comment values(5,108,'Good Afternoon!','2021-01-29 19:01:12');

INSERT INTO Post_Like values(1,101);
INSERT INTO Post_Like values(1,102);
INSERT INTO Post_Like values(1,103);
INSERT INTO Post_Like values(1,104);
INSERT INTO Post_Like values(2,105);
INSERT INTO Post_Like values(2,102);
INSERT INTO Post_Like values(2,108);
INSERT INTO Post_Like values(3,101);
INSERT INTO Post_Like values(4,102);
INSERT INTO Post_Like values(4,103);
INSERT INTO Post_Like values(4,104);

INSERT INTO Community_Members values(1,102);
INSERT INTO Community_Members values(1,103);
INSERT INTO Community_Members values(1,104);
INSERT INTO Community_Members values(2,102);
INSERT INTO Community_Members values(2,108);
INSERT INTO Community_Members values(3,105);

INSERT INTO User_Recommends values(101,102, 'I think you should check him out!');
INSERT INTO User_Recommends values(102,105, 'I recommend this ambitious student!');
INSERT INTO User_Recommends values(102,103, 'very good programmer!');

INSERT INTO User_Enrolls values(101,1);
INSERT INTO User_Enrolls values(102,1);
INSERT INTO User_Enrolls values(103,2);
INSERT INTO User_Enrolls values(102,2);
INSERT INTO User_Enrolls values(105,2);

INSERT INTO Ins_Assigns values(101,1);
INSERT INTO Ins_Assigns values(106,2);
INSERT INTO Ins_Assigns values(106,1);

INSERT INTO Std_Loads values(102,1,'./file.pdf');
INSERT INTO Std_Loads values(103,1,'./file.pdf');
INSERT INTO Std_Loads values(104,2,'./file.pdf');
INSERT INTO Std_Loads values(103,2,'./file.pdf');

INSERT INTO Taken_Lessons values(102,1);
```

```
INSERT INTO Taken_Lessons values(102,2);
INSERT INTO Taken_Lessons values(103,1);

-- ilk 3 eğitim son 3 şirket
INSERT INTO Business_Page values(4,103,"Ece's LTD.", "Carpe diem", "Data Science");
INSERT INTO Business_Page values(5,102,"Mahdi Sube 2 LTD.", "lorem ipsum",
"Transportation");
INSERT INTO Business_Page values(6,103,"Mahdi Sube 3 LTD.", "lorem ipsum", "Data
Science");
INSERT INTO Business_Page values(1,104,"Mahdi Sube 4 LTD.", "lorem ipsum", "Education");
INSERT INTO Business_Page values(2,104,"University of Veli", "Contemporary solutions!",
"Education");
INSERT INTO Business_Page values(3,102,"Mahdi LTD.", "Lorem ipsum", "Self-Improvement");


INSERT INTO Edu_Institute values(1);
INSERT INTO Edu_Institute values(2);
INSERT INTO Edu_Institute values(3);

INSERT INTO Company values(4, 'Charity');
INSERT INTO Company values(5, 'Non Profit');
INSERT INTO Company values(6, 'Charity');

INSERT INTO Edu_Experience values(1,'Masters','2022-01-29','2023-01-29', 102,3);
INSERT INTO Edu_Experience values(2,'Certificate','2022-01-29','2023-01-29', 102,2);
INSERT INTO Edu_Experience values(3,'Bachelors','2022-01-29','2023-01-29', 102,1);
INSERT INTO Edu_Experience values(4,'Masters','2022-01-29','2023-01-29', 104,2);
INSERT INTO Edu_Experience values(5,'Bachelors','2022-01-29','2023-01-29', 105,3);
INSERT INTO Edu_Experience values(6,'Certificate','2022-01-29','2023-01-29', 106,2);

INSERT INTO Work_Experience values(1,'CEO','2021-04-29','2022-01-29', 102,4);
INSERT INTO Work_Experience values(2,'Junior Frontend
Developer','2019-01-29','2020-01-29', 102,5);
INSERT INTO Work_Experience values(3,'Software Developer','2019-11-29','2022-01-20',
103,6);
INSERT INTO Work_Experience values(4,'Junior Developer','2020-10-29','2021-03-29', 103,5);
INSERT INTO Work_Experience values(5,'Full Stack Developer','2020-02-29','2022-01-29',
104,5);
INSERT INTO Work_Experience values(6,'Junior Software
Developer','2019-05-29','2021-01-29', 105,6);

INSERT INTO Job_Ad values(1,"Front-end developer needed!","İzmir",20000,"Job Descb",6);
INSERT INTO Job_Ad values(2,"Windows developer position is open to
hire!","İzmir",40000,"Job Descb",5);
```

```
INSERT INTO Job_Ad values(3,"Internship programs are open!","İzmir",3400,"Job Descb",4);
INSERT INTO Job_Ad values(4,"Janitor needed!","İzmir",19000,"Job Descb",3);
INSERT INTO Job_Ad values(5,"Data scientist needed!","İzmir",70000,"Job Descb",2);
INSERT INTO Job_Ad values(6,"New graduates are welcome for master's
degree!","İzmir",5000,"Job Descb",1);

INSERT INTO Event values(1,'2021-12-29','2022-01-29 20:30:00',105,"Online","Teatalk");
INSERT INTO Event values(2,'2022-02-01','2022-03-10 11:00:00',4,"İzmir","Tech Week");

INSERT INTO User_Interests values(101 ,4);
INSERT INTO User_Interests values(101 ,6);
INSERT INTO User_Interests values(102 ,4);

INSERT INTO User_Work_Experiences values(4 ,101);
INSERT INTO User_Work_Experiences values(6 ,102);
INSERT INTO User_Work_Experiences values(5 ,102);
INSERT INTO User_Work_Experiences values(5 ,104);

INSERT INTO User_Edu_Experiences values(2 ,101);
INSERT INTO User_Edu_Experiences values(2 ,105);
INSERT INTO User_Edu_Experiences values(1 ,106);
```

8. Write down 3 triggers for 3 different tables. Triggers should be meaningful.

```
-- User'a yeni tuple eklendiğinde Organizer'e User'ın user_id'si eklenir
CREATE TRIGGER insert_organizer_usr
   AFTER INSERT
       ON User
       FOR EACH ROW BEGIN
       INSERT INTO Organizer SET organizer_id = NEW.user_id;
      END;

-- Company'e yeni tuple eklendiğinde Organizer'e Company'nin page_id'si eklenir
CREATE TRIGGER insert_organizer_comp
   AFTER INSERT
       ON Company
       FOR EACH ROW BEGIN
       INSERT INTO Organizer SET organizer_id = NEW.page_id;
      END;
```

```
-- User'ı UPDATE ettiğimizde güncel User'ın user_id'si Organizer'e eklenir
CREATE TRIGGER update_organizer_usr
   AFTER UPDATE
       ON User
       FOR EACH ROW BEGIN
       UPDATE Organizer SET organizer_id = NEW.user_id WHERE organizer_id =
NEW.user_id;
      END;

-- Company'ı UPDATE ettiğimizde güncel Company'nin page_id'si Organizer'e eklenir
CREATE TRIGGER update_organizer_comp
   AFTER UPDATE
       ON Company
       FOR EACH ROW BEGIN
       UPDATE Organizer SET organizer_id = NEW.page_id WHERE organizer_id =
NEW.page_id;
      END;

-- User'dan bir tuple silinirse Organizer'dan da aynı tuple silinir
CREATE TRIGGER delete_organizer_usr
   AFTER DELETE
       ON User
       FOR EACH ROW BEGIN
       DELETE FROM Organizer WHERE organizer_id = user_id;
      END;

-- Company'den bir tuple silinirse Organizer'dan da aynı tuple silinir
CREATE TRIGGER delete_organizer_comp
   AFTER DELETE
       ON Company
       FOR EACH ROW BEGIN
       DELETE FROM Organizer WHERE organizer_id = page_id;
      END;

-- Received_Messages tablosu dolduruluyor
CREATE TRIGGER recieved_messages
   AFTER INSERT
       ON Message
       FOR EACH ROW BEGIN
       INSERT INTO Recieved_Messages SET sender_id = NEW.sent_by, reciever_id =
NEW.sent_to, content = NEW.content;
      END;
```

9. Write down 3 check constraints and 3 assertions. Check constraints and assertions should be meaningful.

- - *Check constraints are included in Create Table section.*

```
…
CONSTRAINT chk_EventDate CHECK (time > create_date),
…
CONSTRAINT chk_Date CHECK (end_date > start_date),
…
CONSTRAINT chk_EduDate CHECK (end_date> start_date),
…
CONSTRAINT chk_WorkDate CHECK (end_date> start_date),
…
```

- - Assertions are not supported in MySQL. Instead, we used Triggers.

```
-- SQLSTATE '45000' unhandled user-defined exception
CREATE TRIGGER assertion_communitymember
        BEFORE INSERT ON Community_Members
   FOR EACH ROW BEGIN
        IF (NEW.user_id NOT IN(
                SELECT user_id
                FROM User)) THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'HATA: Topluluğa eklenmek
istenen kullanıcı veritabanında mevcut değil!';
        END IF;
END;


CREATE TRIGGER assertion_instructorstudent
        BEFORE INSERT ON Instructor
   FOR EACH ROW BEGIN
        IF (NEW.instructor_id IN(
                SELECT student_id
                FROM Student)) THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'HATA: Eklenmek isteyen öğrenci
eğitmen olamaz!';
        END IF;
END;
```

```
CREATE TRIGGER assertion_studentinstructor
        BEFORE INSERT ON Student
   FOR EACH ROW BEGIN
        IF (NEW.student_id IN(
                SELECT instructor_id
                FROM Instructor)) THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'HATA: Eklenmek isteyen eğitmen
öğrenci olamaz!';
        END IF;
END;
```

10. Write down the following SQL statements:

a. Write sample INSERT, DELETE and UPDATE statements for 3 of the tables you have chosen.

```
UPDATE User SET username='emineplt'WHERE username='emine_plt';
DELETE FROM Business_Page where page_id = 3;

UPDATE Work_Experience SET user_id=105 WHERE page_id=5;
DELETE FROM Work_Experience WHERE user_id=105 AND page_id=6;

DELETE FROM Job_Ad WHERE page_id=4;
UPDATE Job_Ad SET page_id=3 WHERE page_id=7;
```

- - Inserts are included in Triggers

```
CREATE TRIGGER recieved_messages
   AFTER INSERT
       ON Message
       FOR EACH ROW BEGIN
       INSERT INTO Recieved_Messages SET sender_id = NEW.sent_by, reciever_id =
NEW.sent_to, content = NEW.content;
      END;


CREATE TRIGGER insert_organizer_comp
   AFTER INSERT
       ON Company
       FOR EACH ROW BEGIN
       INSERT INTO Organizer SET organizer_id = NEW.page_id;
      END;
```

```
CREATE TRIGGER insert_organizer_usr
   AFTER INSERT
       ON User
       FOR EACH ROW BEGIN
       INSERT INTO Organizer SET organizer_id = NEW.user_id;
      END;
```

b. Write 10 SELECT statements for the database you have implemented.

i. 3 of them should use just one table

```
SELECT DISTINCT content
FROM Message
WHERE content LIKE '%selam%';

SELECT *
FROM Message
WHERE sent_by > 2
ORDER BY sent_by;

SELECT post_id, COUNT(*) as Likes
FROM Post_Like
GROUP BY post_id;
```

ii. 4 of them should use minimum 2 tables

```
SELECT user_id, first_name, last_name, entry_body
FROM Post, User
WHERE user_id = published_by;

SELECT first_name, last_name
FROM User, Course
WHERE given_by = user_id;

-- Job_Ad oluşturan Business_Page'lerin bilgileri getirilir
SELECT page_name, Job_Ad.title, description
FROM Business_Page, Job_Ad
WHERE Business_Page.page_id = Job_Ad.page_id AND Job_Ad.page_id = 6;

SELECT first_name, last_name, gpa
FROM User, Student
```

```
WHERE student_id = user_id;
```

iii. 3 of them should use minimum 3 tables.

```
-- İstenen kullanıcın eğitim enstitüsü bilgilerini listeliyor
SELECT first_name, last_name, page_name
FROM User, Edu_Experience AS edu_ex, Business_Page AS bp
Where bp.page_id = edu_ex.page_id AND User.user_id = 102 AND User.user_id =
edu_ex.user_id;

-- farklı açıklamalara sahip ödevlerin ders bilgileri listeleniyor
SELECT DISTINCT Assignment.description ,lesson_name, User.first_name, User.last_name
FROM Assignment, Lesson, User
WHERE Assignment.lesson_id = Lesson.lesson_id AND User.user_id = Lesson.instructor_id;

-- Post'a yapılan yorum ve kullanıcı bilgileri
SELECT Post.post_id, Comment.content, User.username
FROM Post, Comment, User
WHERE Post.post_id = Comment.post_id AND Comment.user_id = User.user_id;
```

c. Write 5 original SELECT statements that you think critical to interaction and integration points for the database.

```
-- bir kullanıcının bütün tecrübeleri (work + educational)
SELECT DISTINCT first_name, last_name, page_name AS Experience_at
FROM User, Edu_Experience, Business_Page
WHEREBusiness_Page.page_id = Edu_Experience.page_id AND User.user_id = 102 AND
User.user_id = Edu_Experience.user_id
UNION
SELECT DISTINCT first_name, last_name, page_name AS Experience_at
FROM User, Work_Experience, Business_Page
Where Business_Page.page_id = Work_Experience.page_id AND User.user_id = 102 AND
User.user_id = Work_Experience.user_id;

-- belirlenen aralıklardaki maaş için iş ve sektör bilgileri
SELECT ja1.title, ja1.salary, bp.industry
FROM Business_Page AS bp, Job_Ad AS ja1
WHERE bp.page_id = ja1.page_id AND ja1.page_id IN (
        SELECT ja2.page_id
        FROM Job_Ad AS ja2
        WHERE salary BETWEEN 30000 AND 90000);
```

```
-- organizer_id iki farklı tablodan (User&Company) toplanıyor
-- trigger kullanılmadığı durumda şart
WITH Mix as (
SELECT * FROM User
JOIN Company
UNION
SELECT * FROM Company
JOIN User)
SELECT DISTINCT user_id AS organizer_id from Mix;

-- İşyerlerinin 4 tablodan detaylarının görüntülenmesi
SELECT Company.page_id as ID, Business_Page.page_name as BusinessName,
Business_Page.user_id as CreatorID, User.first_name as FirstName, User.last_name as
LastName, Business_Page.title, Business_Page.industry
FROM User, Company JOIN Business_Page ON Company.page_id = Business_Page.page_id
WHERE User.user_id = Business_Page.user_id
UNION
SELECT Edu_Institute.page_id as ID, Business_Page.page_name as BusinessName,
Business_Page.user_id as CreatorID, User.first_name as FirstName, User.last_name as
LastName, Business_Page.title, Business_Page.industry
FROM User, Edu_Institute JOIN Business_Page ON Edu_Institute.page_id =
Business_Page.page_id
WHERE User.user_id = Business_Page.user_id;

-- Her eğitmenin verdiği ödev sayısı
SELECT instructor_id, User.first_name, User.last_name, COUNT(assignment_id) AS
NumOfAssignments
FROM Ins_Assigns, User
WHERE instructor_id = user_id
GROUP BY instructor_id;
```