2021-2022 FALL SEMESTER

EGE UNIVERSITY
FACULTY of ENGINEERING
COMPUTER ENGINEERING DEPARTMENT
DATABASE MANAGEMENT
2021-2022

LINKEDINMOODLE

Kaan Caner Kurtcephe 05180000022

Sinan Döşeyici 05180000037

Emrehan Arıkmert 05180000052

İçindekiler

Introduction to LinkedinMoodle Database Analysis

LINKEDIN

Linkedin is a web page that has been called a professional business network environment and has carried the network business to the internet world. With this page, we can speed up communication with the people we want to network with.

MOODLE

Moodle provides the service of viewing and publishing the required documentation of the students and the administrators.

Aim of Each Application

Linkedin

Linkedin is a social network established for business purposes. It has all the content a social network can contain, for example: sharing posts, making comments, viewing profiles. And it also lets you post and apply for jobs . You can connect to people, then you can offer a CV-like profile service to your connections. There are also company pages for users who are interested in company information.

Moodle

Moodle is a course management system. This system can be created by an university or educational institution. You must specify a user name in the system before your user can login to the page. You can then sign up for the courses provided with a specific key and access the documentation for that course. This system lets you upload information, assignments and projects to the course. Then your instructor can collects and evaluates all of these. You can communicate with other users through the forums that have been created.

LinkedinMoodle

Our system allows students to view company pages to apply for both internships and job adverts. In addition, the content of courses registered in the university; It provides a system in which the students can view the projects of the courses and upload documents to the grades of the exams. From a instructor's point of view, our system allows him/her to teach in the courses opened by his department and to share his projects on this course page. For the Company, our system ensures that: it publishes and receives applications; to be able to view the notes, projects, courses taken in the profile information of the students. Also, our system allows employees to connect with students, teachers and to accsess other company pages.

LINKEDIN

Main Entities of Linkedin

MEMBER: It is the part where the members' information is kept.

ORGANIZATIONS: It is where organizational information is kept.

CONNECTIONS: It is where the connection information is kept.

CV: It is the part where a member's cv information is kept.

MEMBER_PROFILE: It is the section where member profile information is kept.

ADRESSES: It is the part where members' address information is kept

GROUP: It is the part where the groups of members are kept.

Characteristics of each entity of Linkedin

MEMBER

- member_id
- address_id
- current_organizatin_id
- date_joined
- date_of_birth
- email_address
- email_password
- first_name
- last_name
- gender

ORGANIZATON

- organization_id
- organization_name

CONNECTION

- connection_id
- connection_member_id
- member_id
- date_connection_made

MEMBER_BEING_FOLLOWED

- member_id
- member_being_followed_id
- date_started_following
- date_stopped_following

RECOMMENDATION

- member_recommending_id
- member_being_recommended_id
- date_of_recommendation

CV

- cv_id
- member_id
- date_created
- date_updated

MEMBER_PROFILE

- profile_id
- member_id
- date_created
- date_last_updated

MEMBER_GROUP

- member_id
- group_id
- date_joined
- date_left

GROUP

- group_id
- created_by_member_id
- group_name
- group_description
- started_date
- ended_date

ADDRESSES

- adresses_id
- line_1
- line_2
- city
- state_country_province
- zip
- country

POST

- created_by_member_id
- created_date
- text
- reaction

MESSAGE

- sender_id
- receiver_id
- message_text
- sending_date

JOB_ADVERT

- employer_id
- advert_text
- advert_date

NOTIFICATION

- member_id
- list_notifications

Relationships exists among the entities of Linkedin

- MEMBER -> CREATE -> ORGANIZATION
- MEMBER -> CAN -> CONNECTION
- MEMBER -> HAS -> CV
- MEMBER -> HAS -> MEMBER_PROFILE
- MEMBER -> CREATE -> GROUP
- MEMBER-> HAS -> ADDRESSES

Constraints related to entities, their characteristics and the relationships of Linkedin

- A MEMBER can create multiple ORGANIZATION.
- An ORGANIZATION belongs to a MEMBER.
- A MEMBER must create at least one CONNECTION.
- A CONNECTION belongs to a MEMBER.
- A MEMBER must follow at least one MEMBER.
- A MEMBER may be followed by more than one MEMBER.
- A MEMBER must leave RECOMMENDATION to at least one MEMBER.
- A MEMBER can receive RECOMMENDATION by more than one MEMBER.
- A MEMBER must have at least one CV.
- A MEMBER can have more then one NOTIFICATION.
- A MEMBER can send more then one MESSAGE to MEMBER.
- A MEMBER can recieve more then one MESSAGE from MEMBER.
- A MEMBER can post more then one text or image.
- A MEMBER can advert more then one job.
- A CV belongs to a MEMBER.
- A MEMBER definitely has MEMBER_PROFIL.
- A MEMBER can have more than one ADDRESS.
- A MEMBER can create more than one GROUP.
- A GROUP can be created by more than one MEMBER.

MOODLE

Main Entities of Moodle

COURSES: It is the entity that holds information about the courses.

SUBJECTS: It is the entity that holds information about the courses.

COURSE_AUTHORS_AND_TUTORS: It is the entity where the subjects of the courses are held.

STUDENT_COURSE_ENROLMENT: It is the entity that establishes the connection with the course to which the student will enroll and contains the password information.

STUDENT_TESTS_TAKEN: It is the entity that holds the information of the tests presented to the students.

STUDENT: It is the entity that holds the characteristics of the student.

Characteristics of each entity of Moodle

COURSES

- course_id
- author_id
- subject_id
- course_name
- course_description
- others

STUDENT

- student_id
- date_of_registration
- date_of_latest_login
- login_name
- passwords
- personal_name
- last_name
- others

COURSE_AUTHORS_AND_TUTORS

- author_id
- login_name
- password
- personal_name
- last_name
- gender
- addres_line
- others

STUDENT_COURSE_ENROLMENT

- registration_id
- course_id
- date_of_enrollment
- date_of_completion

SUBJECTS

- subject_id
- subject_name

PROJECTS

- student_id
- author_id
- project_grade
- deadline
- delivery_date

STUDENT_TESTS_TAKEN

- registration_id
- test_grade
- date_test_taken

Relationships exists among the entities of Moodle

- COURSES -> HAVE -> SUBJECTS
- COURSES -> NEED- > COURSE_AUTHORS_AND_TUTORS
- COURSES -> HAVE -> STUDENT_COURSE_ENROLMENT
- STUDENTS -> ENROLL -> STUDENT_COURSE_ENROLMENT
- STUDENT_TESTS_TAKEN> BELONG -> STUDENT_COURSE_ENROLMENT
- STUDENT -> DO -> PROJECTS

Constraints related to entities, their characteristics and the relationships of moodle

- A SUBJECT can belong to more than one COURSE.
- A COURSES does not have to be a SUBJECT.
- You need to be a COURSE_AUTHOR_AND_TUTORS of a COURSE.
- A COURSE_AUTHOR_AND_TUTORS can be linked to more than one COURSES.
- A COURSES must have a STUDENT_COURSE_ENROLLMENT.
- A STUDENT_COURSE_ENROLMENT can belong to more than one COURSES.
- A STUDENT_COURSE_ENROLMENT has to have a STUDENT.
- A STUDENT can have more than one STUDENT_COURSE_ENROLLMENT.
- A STUDENT_COURSE_ENROLLMENT contain multiple STUDENT_TESTS_TAKEN.
- A STUDENT_TEST_TAKEN belongs to a STUDENT_COURSE_ENROLLMENT.

LINKEDINMOODLE

Main Entities of LinkedinMoodle

USER: It is the entity where user's records are kept.

POST: It is the entity where posts records are kept.

GROUP: It is the entity where groups records are kept.

MEMBER: It is the entity where member records are kept.

MEMBER_SKILL: It is the entity where member with skills records are kept.

SKILL: It is the entity where skills records are kept.

EMPLOYEE: It is the entity where employee records are kept.

COMPANY: It is the entity where company's records are kept.

INSTRUCTOR: It is the entity where instructors records are kept.

STUDENT: It is the entity where student's records are kept.

JOB_ADVERT: It is the entity where job advertisers records are kept.

DEPARTMENT: It is the entity where departments records are kept.

UNIVERSITY: It is the entity where universities records are kept.

COURSE: It is the entity where courses records are kept.

PROJECT: It is the entity where projects records are kept.

Characteristics of each entity of LinkedinMoodle

USER

- Username
- Address
- Email
- Post_num
- Created_date
- User_id
- Context

UNIVERSITY

- Uni_id
- Uni_name

COMPANY

- Comp_name
- Sector
- Comp_id

JOB_ADVERT

- Advert_name
- Advert_id
- Advert_time
- Working_type

STUDENT

- Student_id
- Start_year
- GPA

DEPARTMENT

- Dept_id
- Dept_name

PROJECT

- Project_name
- Project_id

COURSE

- Course_id
- Course_name

INSTRUCTOR

- Inst_id

EMPLOYEE

- Emp_id

SKILL

- Skill_id
- Skill_name

MEMBER

- Ssn
- BDate
- Sex
- FName
- LName

POST

- Post_id
- Post_date
- Text

GROUP

- Group_name
- Group_id
- Created_date

Relationships exists among the entities of LinkedinMoodle

- USER -> LIKE -> POST
- USER -> PUBLISH -> POST
- USER -> COMMENT -> POST
- USER -> CREATE -> GROUP
- USER -> COM_MEMBER -> GROUP
- MEMBER -> MESSAGE -> MEMBER
- MEMBER -> HAS -> SKILL
- MEMBER -> ENDORSE -> MEMBER_SKILL
- EMPLOYEE -> EXP_ON -> COMPANY
- COMPANY -> ADVERTS -> JOB_ADVERT
- STUDENT -> JOB_APP -> JOB_ADVERT
- STUDENT -> STU_DEPT -> DEPARTMENT
- STUDENT -> TAKES -> COURSE
- STUDENT -> DO -> PROJECT
- INSTRUCTOR -> ASSIGN -> PROJECT
- INSTRUCTOR -> WORKS_AT -> DEPARTMENT
- INSTRUCTOR -> GIVES -> COURSE
- INSTRUCTOR -> CHAIR -> DEPARTMENT
- COURSE -> HAS -> PROJECT
- DEPARTMENT -> BELONG - > UNIVERSITY

Constraints related to entities, their characteristics and the relationships of LinkKariyerMood

Each user can publish many posts.

Each post must belongs to only one user.

Each user can like many posts.

Posts could be liked from many users.

A user can comment to many posts.

Posts can include many comments.

Users can create many groups and also groups can publish many posts.

Each group must created by only one user.

Users can make a relationships with each other. (like Followed-Follower relationship)

User must have a type which is Member(Employee, Student, Instructor) and Company.

Company can advert many job adverts.

Job adverts must belong to only one company.

Members can send text to each other.

Member can have many skills.

Skills can belong to many members.

A member could be an Employee, a Student or an Instructor.

An Employee may have many experience on Companys. // Experience on

A student can apply for a job.

A student must enroll only one department.

A department may have many students.

A student can take many courses.

A course can taked by many students.

A department must belong to university.

An university may have many departments.

Departments may have courses.

A course must belong to department.

A course can open moodle course.

A moodle course must belong to a course.

A moodle course has many projects.

A project must opened by moodle course.

Students can do many projects.

Projects can be done from many students.

Instructors can assign many projects.

Projects must assigned by an instructor.

An instructor can advise many students.

A student must advised by an instructor.

Instructors can give many courses.

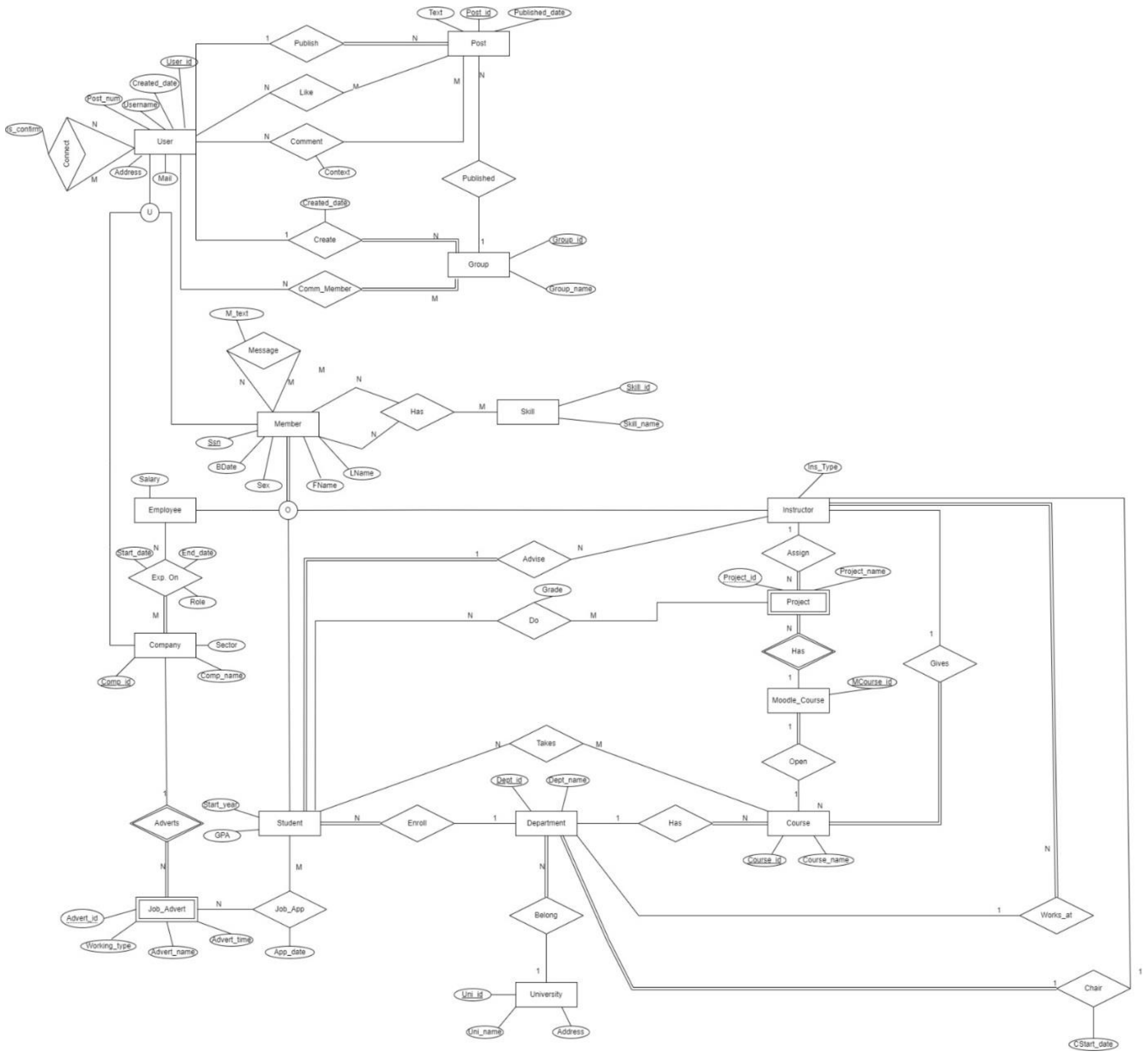
Courses must given by an instructor.

Department must have only one chairman who is an instructor.

An instructor must works at department.

A department can have many instructors.

Converting to EER Diagram



The most important point of our design

Our main purpose of creating Linkedin-Moodle database system is connecting a Linkedin which is most popular social network website and Moodle which is most popular course management system to each other. Thus, Companies can reach to the course projects that being done from students and can give a chance to them for working their companies.

Mapping Phase

1st Iteration:

Step-1

Post(Post_id, Text, Published_date)
Group(Group_id, Group_name)
Skill(Skill_id, Skill_name)
Department(Dept_id, Dept_name)
University(Uni_id, Uni_name, Address)
Course(Course_id, Course_name)
Moodle_Course(MCourse_id)

Step-2

Project(MCourse_id, Project_id, Project_name)

Step-3

Moodle_Course(…, Course_id)

Step-4

Post(…, Group_id)
Department(…, Uni_id)
Course(…, Dept_id)

Step-5

-

Step-6

-

Step-7

-

Step-8

-

Step-9

User(User_id, Username, Created_date, Post_number, Address, Mail)
Member(Ssn, BDate, Sex, FName, LName, User_id)
Company(Comp_id, Comp_name, Sector, User_id)

2nd Iteration:

Step-1

-

Step-2

Job_Advert(Comp_id, Advert_id, Working_type, Advert_name, Advert_time)

Step-3

-

Step-4

Post(…, User_id)

Group(…, User_id, Created_date)

Step-5

Message(Sender_Ssn, Reciever_Ssn,Text)

Like_Post(User_id, Post_id)

Comment_Post(User_id, Post_id, Context)

Comm_Member(User_id, Group_id)

Connection(User_id_from, User_id_to, is_confirm)

Step-6

-

Step-7

Skill_Endorse(Approved_Ssn, Approver_Ssn, Skill_id)

Step-8

Member(…, Member_type)

Employee(Ssn, Salary)

İnstructor(Ssn, Ins_type)

Student(Ssn, Start_year, GPA)

Step-9

-

3rd Iteration:

Step-1

-

Step-2

-

Step-3

Department(…, Instructor_Ssn)

Step-4

Student(…, Department_id, Ins_Ssn)

Project(…, Ins_Ssn)

Course(…, Ins_Ssn)

Instructor(…, Dept_id)

Step-5

Exp_on(Emp_Ssn, Comp_id, Start_date, Role, End_date)

Job_App(Comp_id, Advert_id, Student_Ssn, App_date)

Do_Project(Student_Ssn, MCourse_id, Project_id, Grade)

Take_Course(Student_Ssn, Course_id)

Step-6

-

Step-7

-

Step-8

-

Step-9

-

Creating Database and Creating Tables

```
CREATE TABLE Users(
        User_id INT PRIMARY KEY NOT NULL,
        User_name VARCHAR(30) NOT NULL,
        Mail VARCHAR(30) NOT NULL,
        Address VARCHAR(50),
        Created_date DATE NOT NULL,
        Post_number INT
);
```

```
);


CREATE TABLE Connections(
        User_id_from INT NOT NULL,
        User_id_to INT NOT NULL,
        Is_confirm BOOLEAN NOT NULL,
        PRIMARY KEY (User_id_from, User_id_to),
        FOREIGN KEY (User_id_from) REFERENCES Users(User_id) ON DELETE CASCADE ON UPDATE
CASCADE,
        FOREIGN KEY (User_id_to) REFERENCES Users(User_id) ON DELETE CASCADE ON UPDATE
CASCADE
);


CREATE TABLE Linkedin_Groups(
        Group_id INT PRIMARY KEY NOT NULL,
        Group_name VARCHAR(30) NOT NULL,
        Host_user_id INT,
        Created_date DATE NOT NULL,
        FOREIGN KEY (Host_user_id) REFERENCES Users(User_id) ON DELETE CASCADE ON UPDATE
CASCADE
);


CREATE TABLE Linkedin_Posts(
        Post_id INT PRIMARY KEY NOT NULL,
        Post_text VARCHAR(100),
        Publish_date DATE NOT NULL,
        Group_id INT,
        Host_user_id INT NOT NULL,
        FOREIGN KEY (Host_user_id) REFERENCES Users(User_id) ON DELETE CASCADE ON UPDATE
CASCADE,
        FOREIGN KEY (Group_id) REFERENCES Linkedin_Groups(Group_id) ON DELETE CASCADE ON
UPDATE CASCADE
);
```

```
CREATE TABLE Post_Likes(
        Post_id INT NOT NULL,
        Host_user_id INT NOT NULL,
        FOREIGN KEY (Host_user_id) REFERENCES Users(User_id) ON DELETE CASCADE ON UPDATE
CASCADE,
        FOREIGN KEY (Post_id) REFERENCES Linkedin_Posts(Post_id) ON DELETE CASCADE ON
UPDATE CASCADE,
        PRIMARY KEY (Post_id, Host_user_id)
);

CREATE TABLE Post_Comments(
        Post_id INT NOT NULL,
        Host_user_id INT NOT NULL,
        Context VARCHAR(50),
        FOREIGN KEY (Host_user_id) REFERENCES Users(User_id) ON DELETE CASCADE ON UPDATE
CASCADE,
        FOREIGN KEY (Post_id) REFERENCES Linkedin_Posts(Post_id) ON DELETE CASCADE ON
UPDATE CASCADE,
        PRIMARY KEY (Post_id, Host_user_id)
);

CREATE TABLE Group_Members(
        Group_id INT NOT NULL,
        User_id INT NOT NULL,
        FOREIGN KEY (User_id) REFERENCES Users(User_id) ON DELETE CASCADE ON UPDATE
CASCADE,
        FOREIGN KEY (Group_id) REFERENCES Linkedin_Groups(Group_id) ON DELETE CASCADE ON
UPDATE CASCADE,
        PRIMARY KEY (Group_id, User_id)
);
```

```
CREATE TABLE Members(
	Ssn INT PRIMARY KEY NOT NULL,
	BDate DATE NOT NULL,
	Sex VARCHAR(1),
	FName VARCHAR(20) NOT NULL,
	LName VARCHAR(20) NOT NULL,
	User_id INT,
	Member_type VARCHAR(10) NOT NULL,
	FOREIGN KEY (User_id) REFERENCES Users(User_id) ON DELETE SET NULL ON UPDATE
CASCADE
);

CREATE TABLE Messages(
	Sender_Ssn INT NOT NULL,
	Receiver_Ssn INT NOT NULL,
	M_text VARCHAR(50),
	FOREIGN KEY (Sender_Ssn) REFERENCES Members(Ssn) ON DELETE CASCADE ON UPDATE
CASCADE,
	FOREIGN KEY (Receiver_Ssn) REFERENCES Members(Ssn) ON DELETE CASCADE ON UPDATE
CASCADE
);

CREATE TABLE Skills(
	Skill_id INT PRIMARY KEY NOT NULL,
	Skill_name VARCHAR(50) NOT NULL
);
```

```
);

CREATE TABLE Skills_Endorse(
        Skill_id INT NOT NULL,
        Approved_Ssn INT NOT NULL,
        Approver_Ssn INT NOT NULL,
        FOREIGN KEY (Skill_id) REFERENCES Skills(Skill_id) ON DELETE CASCADE ON UPDATE
CASCADE,
        FOREIGN KEY (Approved_Ssn) REFERENCES Members(Ssn) ON DELETE CASCADE ON UPDATE
CASCADE,
        FOREIGN KEY (Approver_Ssn) REFERENCES Members(Ssn) ON DELETE CASCADE ON UPDATE
CASCADE
);

CREATE TABLE Companys(
        Comp_id INT PRIMARY KEY NOT NULL,
        Comp_name VARCHAR(30) NOT NULL,
        Sector VARCHAR(20),
        User_id INT,
        FOREIGN KEY (User_id) REFERENCES Users(User_id) ON DELETE SET NULL ON UPDATE
CASCADE
);

CREATE TABLE Job_Adverts(
        Comp_id INT NOT NULL,
        Advert_id INT NOT NULL,
        Advert_name VARCHAR(30) NOT NULL,
        Working_Type VARCHAR(20),
        Advert_time DATE NOT NULL,
        PRIMARY KEY (Comp_id, Advert_id),
        FOREIGN KEY (Comp_id) REFERENCES Companys(Comp_id) ON DELETE CASCADE ON UPDATE
CASCADE
);
```

```
CREATE TABLE Employees(
	Ssn INT PRIMARY KEY NOT NULL,
	Salary INT,
	FOREIGN KEY (Ssn) REFERENCES Members(Ssn) ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE Exp_On(
	Ssn INT NOT NULL,
	Comp_id INT NOT NULL,
	Start_date DATE NOT NULL,
	End_date DATE,
	Comp_role VARCHAR(20) NOT NULL,
	PRIMARY KEY (Ssn, Comp_id),
	FOREIGN KEY (Ssn) REFERENCES Members(Ssn) ON DELETE CASCADE ON UPDATE CASCADE,
	FOREIGN KEY (Comp_id) REFERENCES Companys(Comp_id) ON DELETE CASCADE ON UPDATE
CASCADE
);

CREATE TABLE University(
	Uni_id INT PRIMARY KEY NOT NULL,
	Uni_name VARCHAR(20) NOT NULL,
	Address VARCHAR(40)
);

CREATE TABLE Departments(
	Dept_id INT PRIMARY KEY NOT NULL,
	Dept_name VARCHAR(50) NOT NULL,
	Uni_id INT NOT NULL,
	FOREIGN KEY (Uni_id) REFERENCES University(Uni_id) ON DELETE CASCADE ON UPDATE
CASCADE
);
```

```
CREATE TABLE Instructors(

            Ssn INT PRIMARY KEY NOT NULL,

            Ins_type VARCHAR(20),

            Dept_id INT NOT NULL,

            FOREIGN KEY (Dept_id) REFERENCES Departments(Dept_id) ON DELETE CASCADE ON
UPDATE CASCADE,

            FOREIGN KEY (Ssn) REFERENCES Members(Ssn)

);

ALTER TABLE Departments ADD Dean_Ssn INT;

ALTER TABLE Departments ADD FOREIGN KEY (Dean_Ssn) REFERENCES Instructors(Ssn) ON DELETE
CASCADE ON UPDATE CASCADE;


CREATE TABLE Courses(

            Course_id INT PRIMARY KEY NOT NULL,

            Course_name VARCHAR(50),

            Dept_id INT NOT NULL,

            Ins_Ssn INT NOT NULL,

            FOREIGN KEY (Ins_Ssn) REFERENCES Instructors(Ssn),

            FOREIGN KEY (Dept_id) REFERENCES Departments(Dept_id)

);


CREATE TABLE Moodle_Courses(

            MCourse_id INT PRIMARY KEY NOT NULL,

            Course_id INT NOT NULL,

            FOREIGN KEY (Course_id) REFERENCES Courses(Course_id)

);
```

```
CREATE TABLE Projects(
	MCourse_id INT NOT NULL,
	Project_id INT NOT NULL,
	Project_name VARCHAR(20),
	Ins_Ssn INT NOT NULL,
	FOREIGN KEY (MCourse_id) REFERENCES Moodle_Courses(MCourse_id),
	FOREIGN KEY (Ins_Ssn) REFERENCES Instructors(Ssn),
	PRIMARY KEY (MCourse_id, Project_id)
);

CREATE TABLE Students(
	Ssn INT PRIMARY KEY NOT NULL,
	GPA FLOAT,
	Start_date DATE,
	Dept_id INT NOT NULL,
	Ins_Ssn INT NOT NULL,
	FOREIGN KEY (Ssn) REFERENCES Members(Ssn) ON DELETE CASCADE ON UPDATE CASCADE,
	FOREIGN KEY (Dept_id) REFERENCES Departments(Dept_id) ON DELETE CASCADE ON
UPDATE CASCADE,
	FOREIGN KEY (Ins_Ssn) REFERENCES Instructors(Ssn) ON DELETE CASCADE ON UPDATE
CASCADE
);

CREATE TABLE Take_Courses(
	Course_id INT NOT NULL,
	Student_Ssn INT NOT NULL,
	FOREIGN KEY (Student_Ssn) REFERENCES Students(Ssn) ON DELETE CASCADE ON UPDATE
CASCADE,
	FOREIGN KEY (Course_id) REFERENCES Courses(Course_id) ON DELETE CASCADE ON UPDATE
CASCADE,
	PRIMARY KEY (Course_id, Student_Ssn)
);
```

```
CREATE TABLE Do_Projects(
        Project_id INT NOT NULL,
        MCourse_id INT NOT NULL,
        Student_Ssn INT NOT NULL,
        Grade INT,
        FOREIGN KEY (Student_Ssn) REFERENCES Students(Ssn) ON DELETE CASCADE ON UPDATE
CASCADE,
        FOREIGN KEY (MCourse_id, Project_id) REFERENCES Projects(MCourse_id, Project_id) ON
DELETE CASCADE ON UPDATE CASCADE,
        PRIMARY KEY (Project_id, MCourse_id, Student_Ssn)
);

CREATE TABLE Job_Apps(
        Comp_id INT NOT NULL,
        Advert_id INT NOT NULL,
        Student_Ssn INT NOT NULL,
        App_date DATE,
        FOREIGN KEY (Student_Ssn) REFERENCES Students(Ssn) ON DELETE CASCADE ON UPDATE
CASCADE,
        FOREIGN KEY (Comp_id, Advert_id) REFERENCES Job_Adverts(Comp_id, Advert_id) ON
DELETE CASCADE ON UPDATE CASCADE,
        PRIMARY KEY (Comp_id, Advert_id, Student_Ssn)
);
```

Inserting Values

```
INSERT INTO users(
        user_id, user_name, mail, address, created_date, post_number)
        VALUES (999999999, 'Emrehan Arıkmert', 'emrehan.arikmert@gmail.com', 'Bornova/İzmir',
'2021-12-05', 6),
         (999999998, 'Sinan Döşeyici', 'sinan.doseyici@gmail.com', 'Bornova/İzmir', '2022-01-11', 4),
         (999999997, 'Caner Kurtcephe', 'caner.kurtcephe@gmail.com', 'Bornova/İzmir', '2022-01-
15',5),
         (999999996, 'İrem Kaya', 'irem.kaya@gmail.com', 'Bornova/İzmir', '2021-12-05', 2),
         (999999995, 'Melisa Erdem', 'melisa.erdem@gmail.com', 'Bornova/İzmir', '2021-11-08', 3),
         (999999994, 'Ayhan Gümüşay', 'ayhan.gumusay@gmail.com', 'Kadıköy/İstanbul', '2021-02-
05', 6),
         (999999993, 'Beren Kulaç', 'beren.kulac@gmail.com', 'Bornova/İzmir', '2021-08-06', 2),
         (999999992, 'Ela Uçar', 'ela.ucar@gmail.com', 'Kadıköy/İstanbul', '2021-10-11', 4),
         (999999991, 'Serdar Karakum', 'serdar.karakum@gmail.com', 'Beşiktaş/İstanbul', '2022-01-
05', 2),
         (999999990, 'Ayça Yılmaz', 'ayca.yilmaz@gmail.com', 'Urla/İzmir', '2021-05-22', 3),
         (999999989, 'Ayhan Yıldırım', 'ayhan.yildirim@gmail.com', 'Urla/İzmir', '2021-06-18', 1),
         (999999988, 'Buse Öner', 'buse.oner@gmail.com', 'Beşiktaş/İstanbul', '2021-11-09', 1),
         (999999987, 'Berna Çağdaş', 'berna.cagdas@gmail.com', 'Beşiktaş/İstanbul', '2021-01-03', 1),
         (999999986, 'Rasim Sezgi', 'rasim.sezgi@gmail.com', 'Beşiktaş/İstanbul', '2020-10-02', 1),
         (999999985, 'Emre Serbest', 'emre.serbest@gmail.com', 'Bornova/İzmir', '2022-01-11', 4),
         (999999984, 'Fatma Çakmak', 'fatma.cakmak@gmail.com', 'Bornova/İzmir', '2022-01-15',5),
         (999999983, 'Çağatay Kurucu', 'cagatay.kurucu@gmail.com', 'Bornova/İzmir', '2021-12-05',
2),
         (999999982, 'Betül Çadır', 'betul.cadir@gmail.com', 'Bornova/İzmir', '2021-11-08', 3),
         (999999981, 'Emre Bozok', 'emre.bozok@gmail.com', 'Kadıköy/İstanbul', '2021-02-05', 6),
         (999999980, 'Aleyna Çelik', 'aleyna.celik@gmail.com', 'Bornova/İzmir', '2021-08-06', 2),
         (999999979, 'Taha Varol', 'taha.varol@gmail.com', 'Kadıköy/İstanbul', '2021-10-11', 4),
         (999999978, 'Zeynep Bastık', 'zeynep.bastik@gmail.com', 'Beşiktaş/İstanbul', '2022-01-05',
2),
         (999999976, 'Talha Demirci', 'talha.demirci@gmail.com', 'Urla/İzmir', '2021-05-22', 3),
         (999999975, 'Ayyüce Yıldız', 'ayyuce.yildiz@gmail.com', 'Urla/İzmir', '2021-06-18', 1),
         (999999974, 'Eyüp Kurnaz', 'eyup.kurnaz@gmail.com', 'Beşiktaş/İstanbul', '2021-11-09', 1),
         (999999973, 'Ayça Er', 'ayca.er@gmail.com', 'Beşiktaş/İstanbul', '2021-01-03', 1),
         (999999972, 'Mehmet Çan', 'mehmet.can@gmail.com', 'Beşiktaş/İstanbul', '2020-10-02', 1),
         (999999971, 'Gizem Aldatmaz', 'gizem.aldatmaz@gmail.com', 'Kadıköy/İstanbul', '2021-10-
11', 4),
         (999999970, 'Halil Varol', 'halil.varol@gmail.com', 'Beşiktaş/İstanbul', '2022-01-05', 2),
         (999999969, 'Nurcan Parlak', 'nurcan.parlak@gmail.com', 'Urla/İzmir', '2021-05-22', 3),
         (999999968, 'Erdem Zor', 'erdem.zor@gmail.com', 'Urla/İzmir', '2021-06-18', 1),
         (999999967, 'Zehra Bozkurt', 'zehra.bozkurt@gmail.com', 'Beşiktaş/İstanbul', '2021-11-09',
1),
         (999999966, 'İbrahim Şengül', 'ibrahim.sengul@gmail.com', 'Beşiktaş/İstanbul', '2021-01-03',
1),
         (999999965, 'Halide Sapıtmaz', 'halide.sapitmaz@gmail.com', 'Beşiktaş/İstanbul', '2020-10-
02', 1),
         (999999964, 'Adem Elma', 'adem.elma@gmail.com', 'Beşiktaş/İstanbul', '2021-01-03', 1),



         (000000001, 'Murat Osman Ünalır', 'unalir@gmail.com', 'Bornova/İzmir', '2015-03-12', 150),
         (000000002, 'Emine Sezer', 'emine.szr@gmail.com', 'Bornova/İzmir', '2017-01-12', 72),
         (000000003, 'Levent Toker', 'leven.toker@gmail.com', 'Bornova/İzmir', '2013-01-02', 80),
         (000000004, 'Vecdi Aytaç', 'vecdi.aytac@gmail.com', 'Bornova/İzmir', '2015-04-22', 50),
         (000000005, 'Aybars Uğur', 'aybars.ugur@gmail.com', 'Bornova/İzmir', '2017-01-12', 110),
         (000000006, 'Aylin Kantarcı', 'aylin.kantarci@gmail.com', 'Bornova/İzmir', '2016-01-12', 63),
         (000000007, 'Birol Çiloğlugil', 'birol.ciloglugil@gmail.com', 'Bornova/İzmir', '2016-04-25', 41),
         (000000008, 'Beste Kaysı', 'beste.kaysi@gmail.com', 'Bornova/İzmir', '2017-07-22', 26),
         (000000009, 'Sezercan Tanışman', 'sezercan.tanisman@gmail.com', 'Bornova/İzmir', '2016-
08-07', 33),
         (000000010, 'Anıl Güven', 'anil.guven@gmail.com', 'Bornova/İzmir', '2018-07-04', 35),
         (000000011, 'Okan Bursa', 'okan.bursa@gmail.com', 'Bornova/İzmir', '2018-04-11', 34),
         (000000012, 'Şebnem Bora', 'sebnem.bora@gmail.com', 'Bornova/İzmir', '2013-11-12', 66),
         (000000013, 'Murat Şimşek', 'murat.simsek@gmail.com', 'Beşiktaş/İstanbul', '2014-04-17',
24),
         (000000014, 'Selinay Altuğ', 'selinay.altug@gmail.com', 'Beşiktaş/İstanbul', '2014-05-15', 72),
         (000000015, 'Sevinç Aydoğdu', 'sevinc.aydogdu@gmail.com', 'Kadıköy/İstanbul', '2015-07-
08', 28),
         (000000016, 'Mustafa Cihan', 'mustafa.cihan@gmail.com', 'Kadıköy/İstanbul', '2016-06-12',
82),
         (100000001, 'Mark Zuckerberg', 'mark.zuckerberg@gmail.com', 'Queens-New York', '2009-
10-21', 70),
         (200000001, 'Joe Green', 'joe.green@gmail.com', 'Alpine-California', '2014-08-27', 12);


INSERT INTO connections(
	user_id_from, user_id_to, is_confirm)
	VALUES (999999999,999999998,true),
	(999999999,999999997,true),
	(999999998,999999997,true),
	(999999996,999999997,true),
	(999999995,999999997,true),
	(999999994,999999997,true),
	(999999993,999999997,false),
	(999999998,999999996,true),
	(999999998,999999995,false),
	(999999998,999999994,true),
	(999999997,999999999,false),
	(999999996,999999999,true),
	(999999995,999999994,true),
	(999999999,999999993,true),
	(999999999,999999995,false);

INSERT INTO linkedin_groups(
	group_id, group_name, host_user_id, created_date)
	VALUES (0001, 'Ege University Job Network', 000000001, '2022-01-01'),
	(0003, 'Bogazici University', 000000014, '2020-07-17'),
	(0004, 'Marmara University', 000000016, '2021-09-11'),
	(0005, 'Facebook', 100000001, '2015-08-18'),
	(0006, 'Elektrik Severler', 000000002, '2021-11-19'),
	(0007, 'IEEE', 000000015, '2022-02-01');

INSERT INTO group_members(
	group_id, user_id)
	VALUES (0001, 000000001),
		(0003, 000000014),
		(0004, 000000016),
                    (0005, 100000001),
                    (0006, 000000002),
                    (0007, 000000015),
                    (0001, 999999999),
                    (0001, 999999998),
                    (0001, 999999997),
                    (0004, 999999992),
                    (0004, 999999991),
                    (0006, 999999993),
                    (0006, 999999996),
                    (0006, 999999998);

INSERT INTO linkedin_posts(
          post_id, post_text, publish_date, group_id, host_user_id)
          VALUES (00001, 'Trendyol ile buluşuyoruz.' , '2021-12-03', 0001, 000000001),
                    (00002, 'Yarın 15.00 te buluşuyoruz...', '2018-02-05', 0006, 000000002),
                    (00003, 'Elektrik labı 13.00 da', '2022-01-03', 0006, 000000002),
                (00004, 'M.U öğrencileriyle buluşuyor.' , '2022-02-03', 0004, 000000016);

INSERT INTO post_likes(
          post_id, host_user_id)
          VALUES (00001, 999999999),
                    (00001, 999999998),
                    (00001, 999999997),
                    (00001, 999999996),
                    (00001, 999999995),
                    (00001, 999999994),
                    (00001, 999999993),
                    (00001, 999999992),
                    (00001, 999999991),
                    (00001, 999999990),
                    (00001, 999999989),

                    (00002, 999999999),

                    (00002, 999999998),

                    (00002, 999999997),

                    (00003, 999999999),

                    (00003, 999999998),

                    (00003, 999999997),

                    (00003, 999999996),

                    (00003, 999999995),

                    (00003, 999999994),

                    (00003, 999999992);


INSERT INTO post_comments(

          post_id, host_user_id, context)

          VALUES (00001, 999999999, 'Harikaa !'),

                    (00001, 999999998, 'Çok güzell !'),

                    (00001, 999999997, 'Teşekkürler, devamını bekliyoruz.');

INSERT INTO members(

          ssn, bdate, sex, fname, lname, user_id, member_type)

          VALUES (518011, '2000-01-25', 'M', 'Emrehan', 'Arıkmert', 999999999, 'Student'),

                    (518037, '2001-01-01', 'M', 'Sinan', 'Döşeyici', 999999998, 'Student'),

                    (518022, '2000-03-08', 'M', 'Caner', 'Kurtcephe', 999999997, 'Student'),

                    (504001, '1982-02-12', 'M', 'Osman', 'Ünalır', 000000001, 'Instructor'),

                    (201001, '1990-03-15', 'M', 'Joe', 'Green', 200000001, 'Employee'),

                    (518027, '2000-02-17', 'F', 'İrem', 'Kaya', 999999996, 'Student'),

                    (518028, '2000-03-21', 'F', 'Melisa', 'Erdem', 999999995, 'Student'),

                    (518029, '2001-12-11', 'M', 'Ayhan', 'Gümüşay', 999999994, 'Student'),

                    (518030, '2001-11-15', 'F', 'Beren', 'Kulaç', 999999993, 'Student'),

                    (518031, '2000-06-05', 'F', 'Ela', 'Uçar', 999999992, 'Student'),

                    (518023, '2000-07-25', 'M', 'Serdar', 'Karakum', 999999991, 'Student'),

                    (518032, '2000-08-03', 'F', 'Ayça', 'Yılmaz', 999999990, 'Student'),
                    (518033, '2000-08-08', 'M', 'Ayhan', 'Yıldırım', 999999989, 'Student'),
                    (518034, '2001-08-11', 'F', 'Buse', 'Öner', 999999988, 'Student'),
                    (518035, '2000-07-15', 'F', 'Berna', 'Çağdaş', 999999987, 'Student'),
                    (518036, '2001-08-23', 'M', 'Rasim', 'Sezgi', 999999986, 'Student'),

                    (518038, '2001-08-23', 'M', 'Emre', 'Serbest', 999999985, 'Student'),
                    (518039, '2001-08-23', 'F', 'Fatma', 'Çakmak', 999999984, 'Student'),
                    (518040, '2001-08-23', 'M', 'Çağatay', 'Kurucu', 999999983, 'Student'),
                    (518041, '2001-08-23', 'F', 'Betül', 'Çadır', 999999982, 'Student'),
                    (518042, '2001-08-23', 'M', 'Emre', 'Bozok', 999999981, 'Student'),
                    (518043, '2001-08-23', 'F', 'Aleyna', 'Çelik', 999999980, 'Student'),
                    (518044, '2001-08-23', 'M', 'Taha', 'Varol', 999999979, 'Student'),
                    (518045, '2001-08-23', 'F', 'Zeynep', 'Bastık', 999999978, 'Student'),
                    (518046, '2001-08-23', 'M', 'Talha', 'Demirci', 999999976, 'Student'),
                    (518047, '2001-08-23', 'F', 'Ayyüce', 'Yıldız', 999999975, 'Student'),
                    (518048, '2001-08-23', 'M', 'Eyüp', 'Kurnaz', 999999974, 'Student'),
                    (518049, '2001-08-23', 'F', 'Ayça', 'Er', 999999973, 'Student'),
                    (518050, '2001-08-23', 'M', 'Mehmet', 'Çan', 999999972, 'Student'),
                    (518051, '2001-08-23', 'F', 'Gizem', 'Aldatmaz', 999999971, 'Student'),
                    (518052, '2001-08-23', 'M', 'Halil', 'Varol', 999999970, 'Student'),
                    (518053, '2001-08-23', 'F', 'Zeynep', 'Bastık', 999999969, 'Student'),
                    (518054, '2001-08-23', 'M', 'Erdem', 'Zor', 999999968, 'Student'),
                    (518055, '2001-08-23', 'F', 'Zehra', 'Bozkurt', 999999967, 'Student'),
                    (518056, '2001-08-23', 'M', 'İbrahim', 'Şengül', 999999966, 'Student'),
                    (518057, '2001-08-23', 'F', 'Halide', 'Sapıtmaz', 999999965, 'Student'),
                    (518058, '2001-08-23', 'M', 'Adem', 'Elma', 999999964, 'Student'),

                    (504002, '1990-05-25', 'F', 'Emine', 'Sezer', 000000002, 'Instructor'),
                    (504003, '1982-06-11', 'M', 'Levent', 'Toker', 000000003, 'Instructor'),
                    (504004, '1988-01-17', 'M', 'Vecdi', 'Aytaç', 000000004, 'Instructor'),
                    (504005, '1982-02-21', 'M', 'Aybars', 'Uğur', 000000005, 'Instructor'),
                    (504006, '1983-03-24', 'F', 'Aylin', 'Kantarcı', 000000006, 'Instructor'),
                    (504007, '1988-03-29', 'M', 'Birol', 'Çiloğlugil', 000000007, 'Instructor'),
                    (504008, '1993-04-22', 'F', 'Beste', 'Kaysı', 000000008, 'Instructor'),
                    (504009, '1992-05-02', 'M', 'Sezercan', 'Tanışman', 000000009, 'Instructor'),
                    (504010, '1991-06-01', 'M', 'Anıl', 'Güven', 000000010, 'Instructor'),
                    (504011, '1990-11-08', 'M', 'Okan', 'Bursa', 000000011, 'Instructor'),
                    (504012, '1985-12-13', 'F', 'Şebnem', 'Bora', 000000012, 'Instructor'),
                    (504013, '1982-09-12', 'M', 'Murat', 'Şimşek', 000000013, 'Instructor'),
                    (504014, '1990-10-06', 'F', 'Selinay', 'Altuğ', 000000014, 'Instructor'),
                    (504015, '1991-06-11', 'F', 'Sevinç', 'Aydoğdu', 000000015, 'Instructor'),
                    (504016, '1995-03-23', 'M', 'Mustafa', 'Cihan', 000000016, 'Instructor');

    INSERT INTO messages(
            sender_ssn, receiver_ssn, m_text)
            VALUES (518011, 518037, 'Sinan selam, meetinge katılacak mısın?'),
                    (518037, 518011, 'Selam Emrehan, katılacağım.'),
                    (518022, 518037, 'Meeting adresi: kisalink/15a27J'),
                    (518037, 518022, 'Sağol Caner'),
                    (518037, 518011, 'hatta şu da şurda dursun: kisalink/15a27J');

    INSERT INTO skills(
            skill_id, skill_name)
            VALUES (1, 'Foreign language skills'),
                    (2, 'Coding Ability'),
                    (3, 'Photoshop knowledge'),
                    (4, 'Writing proficiency'),
                    (5, 'Empathy'),
                    (6, 'Communication'),
                    (7, 'Leadership'),
                    (8, 'Time management');
```

```
INSERT INTO skills_endorse(
        skill_id, approved_ssn, approver_ssn)
        VALUES (1, 518011, 518022),
                        (2, 518011, 518022),
                        (3, 518011, 518022),
                        (1, 518022, 518037),

                        (6, 518022, 518011),
                        (7, 518022, 518011),
                        (8, 518022, 518011),
                        (8, 518022, 518037),

                        (3, 518037, 518022),
                        (4, 518037, 518011),
                        (3, 518037, 518022),
                        (4, 518037, 518011),
                        (3, 518022, 504001),
                        (1, 518022, 504002),
                        (1, 518022, 504002),
                        (2, 518011, 504001),
                        (3, 518011, 504002),
                        (1, 518011, 504001),
                        (4, 518037, 504001),
                        (3, 518037, 504001),
                        (3, 518037, 504004),
                        (4, 518037, 504005),
                        (3, 518037, 504006);

INSERT INTO companys(
        comp_id, comp_name, sector, user_id)
        VALUES (1, 'Facebook', 'Social Media', 100000001);
```

```
INSERT INTO job_adverts(
        comp_id, advert_id, advert_name, working_type, advert_time)
        VALUES (1, 1, 'Database Manager Hiring', 'Part-time', '2022-01-25'),
                (1, 2, 'Frontend Developer Hiring', 'Full-time', '2022-03-05'),
                (1, 3, 'Backend Developer Hiring', 'Part-time', '2021-12-03'),
                (1, 4, 'Ios Developer Hiring', 'Full-time', '2021-11-21'),
                (1, 5, 'Android Developer Hiring', 'Internship', '2021-08-17');

INSERT INTO public.employees(
        ssn, salary)
        VALUES (201001, 7500);

INSERT INTO exp_on(
        ssn, comp_id, start_date, end_date, comp_role)
        VALUES (201001, 1, '2003-11-07', NULL, 'Ios Developer');

INSERT INTO university(
        uni_id, uni_name, address)
        VALUES (1, 'Ege University', 'Bornova-İzmir'),
                (2, 'Bogazici University', 'Beşiktaş-İstanbul'),
                (3, 'Marmara University', 'Kadıköy-İstanbul');


INSERT INTO departments(
        dept_id, dept_name, uni_id)
        VALUES (1, 'Bilgisayar Mühendisliği', 1),
                (2, 'Makine Mühendisliği', 1),
                (3, 'İşletme', 1),
                (4, 'Olasılık İstatistik', 1),
                (5, 'Güzel Sanatlar', 1),
                (6, 'Bilgisayar Mühendisliği', 2),
                (7, 'Makine Mühendisliği', 2),
                (8, 'İşletme', 2),
                (9, 'Olasılık İstatistik', 2),
                (10, 'Matematik Mühendisliği', 2),
                (11, 'Bilgisayar Mühendisliği', 3),
                (12, 'Makine Mühendisliği', 3),
                (13, 'İşletme', 3),
                (14, 'Olasılık İstatistik', 3);

INSERT INTO instructors(
        ssn, ins_type, dept_id)
        VALUES (504001, 'Professor', 1),
                (504002, 'Professor', 1),
                (504003, 'Professor', 1),
                (504004, 'Professor', 1),
                (504005, 'Professor', 1),
                (504006, 'Professor', 1),
                (504007, 'Associate Professor', 1),
                (504008, 'Assistant', 1),
                (504009, 'Assistant', 1),
                (504010, 'Assistant', 1),
                (504011, 'Associate Professor', 1),
                (504012, 'Professor Assistant', 1),
                (504013, 'Professor', 6),
                (504014, 'Assistant', 6),
                (504015, 'Professor', 11),
                (504016, 'Professor', 11);

INSERT INTO courses(
        course_id, course_name, dept_id, ins_ssn)
        VALUES (1, 'Database Management', 1, 504001),
                (2, 'Electrical Circuits', 1, 504002),
                (3, 'Network', 1, 504003),
                (4, 'Discrete Mathematics', 1, 504004),
                (5, 'Operating Systems', 1, 504006),
                (6, 'Digital Computer Design', 1, 504007),
                (7, 'Microprocessors', 1, 504012),
                (8, 'Database Management', 6, 504014),
                (9, 'Database Management', 11, 504015);

INSERT INTO moodle_courses(
        mcourse_id, course_id)
        VALUES (1,1),
                (2, 2),
                (3, 3),
                (4, 5),
                (5, 6),
                (6, 9),
                (7, 8);

INSERT INTO projects(
        mcourse_id, project_id, project_name, ins_ssn)
        VALUES (1, 1, 'Linkedin-Moodle', 504001),
                (2, 2, 'Super node', 504002),
                (3, 3, 'Star Tophology ', 504003),
                (6, 4, 'Airport', 504015),
                (7, 5, 'Asana', 504015),
                (7, 8, 'Netflix', 504014);

INSERT INTO students(
        ssn, gpa, start_date, dept_id, ins_ssn)
```

```
VALUES (518011, 3.60, '2018-09-23', 1, 504007),
             (518037, 3.59, '2018-09-23', 1, 504007),
             (518022, 3.51, '2017-08-23', 1, 504007),
             (518028, 3.8, '2017-08-23', 1, 504002),
             (518029, 3.76, '2019-09-15', 11, 504015),
             (518030, 3, '2018-09-23', 1, 504001),
             (518031, 2.70, '2019-09-15', 11, 504015),
             (518032, 2.86, '2017-09-23', 1, 504003),
             (518033, 3.75, '2019-09-23', 1, 504003),
             (518034, 3.59, '2018-09-23', 6, 504014),
             (518023, 3.17, '2018-09-23', 6, 504014),
             (518035, 2.88, '2018-09-23', 2, 504005),
             (518036, 3.11, '2018-09-23', 2, 504006),

             (518038, 3.19, '2018-09-23', 3, 504014),
             (518039, 3.56, '2018-09-23', 4, 504014),
             (518040, 2.71, '2018-09-23', 4, 504014),
             (518041, 2.49, '2018-09-23', 5, 504012),
             (518042, 3.53, '2018-09-23', 5, 504013),
             (518043, 2.29, '2018-09-23', 7, 504012),
             (518044, 3.11, '2018-09-23', 7, 504013),
             (518045, 3.22, '2018-09-23', 8, 504013),
             (518046, 2.41, '2018-09-23', 8, 504012),
             (518047, 3.53, '2018-09-23', 9, 504012),
             (518048, 3.61, '2018-09-23', 9, 504013),
             (518049, 2.59, '2018-09-15', 10, 504013),
             (518050, 3.21, '2018-09-15', 10, 504014),
             (518051, 3.11, '2018-09-15', 12, 504015),
             (518052, 2.62, '2018-09-15', 12, 504014),
             (518053, 2.56, '2018-09-15', 13, 504015),
             (518054, 3.27, '2018-09-15', 13, 504014),
                    (518055, 3.21, '2018-09-15', 13, 504014),
                    (518056, 3.66, '2018-09-15', 14, 504014),
                    (518057, 3.27, '2018-09-15', 14, 504015),
                    (518058, 3.21, '2018-09-15', 14, 504015);


INSERT INTO take_courses(
          course_id, student_ssn)
          VALUES(1, 518011),
                    (1, 518037),
                    (1, 518022),
                    (2, 518037),
                    (3, 518011),
                    (3, 518037),
                    (3, 518022),
                    (6, 518011),
                    (8, 518034),
                    (8, 518023),
                    (9, 518029),
                    (9, 518031);



INSERT INTO do_projects(
   project_id, mcourse_id, student_ssn, grade)
   VALUES (1, 1, 518011, 30),
      (1, 1, 518037, 80),
      (1, 1, 518022, 90),
      (2, 2, 518037, 90),
      (3, 3, 518011, 20),
      (3, 3, 518037, 85),
      (3, 3, 518022, 82),
      (4, 6, 518011, 10),
      (5, 7, 518034, 58),
      (5, 7, 518023, 91),
      (8, 7, 518023, 80),
      (8, 7, 518034, 43)
;


INSERT INTO job_apps(
   comp_id, advert_id, student_ssn, app_date)
   VALUES (1, 1, 518011, '2021-01-15'),
      (1, 2, 518011, '2021-04-27'),
      (1, 3, 518022, '2020-02-12'),
      (1, 4, 518022, '2021-12-01'),
      (1, 5, 518022, '2020-09-30'),
      (1, 1, 518037, '2021-05-03'),
      (1, 5, 518037, '2021-04-18');
```

TRIGGERS

-Yeni bir post eklendiğinde ekleyen user'ın post sayısını 1 arttıran trigger.

```
CREATE OR REPLACE FUNCTION post_number_changes()
  RETURNS TRIGGER
  LANGUAGE PLPGSQL
  AS
$$
BEGIN
	UPDATE Users SET post_number = post_number + 1 WHERE user_id =
	(SELECT user_id
	FROM Users
	WHERE user_id = NEW.host_user_id
	);

	RETURN NEW;
END;
$$

CREATE TRIGGER Post_number_changes
AFTER INSERT
ON Linkedin_posts
FOR EACH ROW
EXECUTE PROCEDURE post_number_changes();
```

-Yeni bir proje eklendiği zaman sisteme bunun ile ilgili mesaj gönderen trigger

```
CREATE OR REPLACE FUNCTION project_alert()
  RETURNS TRIGGER
  LANGUAGE PLPGSQL
  AS
$$
BEGIN
	RAISE NOTICE 'New Project is assigned';
	RETURN NEW;
END;
$$

CREATE TRIGGER project_alert
AFTER INSERT
ON Projects
FOR EACH ROW
EXECUTE PROCEDURE project_alert();
```

-User tablosu üzerinde username attr değiştirme işlemi yapılmadan önce değiştirilmek istenen username daha önceden kullanılıyorsa değişimine izin vermeyen trigger.

```
CREATE OR REPLACE FUNCTION user_name_changing()
  RETURNS TRIGGER
  LANGUAGE PLPGSQL
  AS
$$
BEGIN
        IF OLD.user_name <> NEW.user_name THEN
                IF (SELECT U.user_name FROM Users AS U WHERE U.user_name =
NEW.user_name) IS NULL THEN
                        RETURN NEW;
                END IF;
        END IF;
        RETURN OLD;
END;
$$


CREATE TRIGGER user_name_changes
BEFORE UPDATE
ON Users
FOR EACH ROW
EXECUTE PROCEDURE user_name_changes();
```

CONSTRAINTS AND ASSERTIONS

```
-ALTER TABLE Members ADD CONSTRAINT sex_list
CHECK (Sex = 'M' OR Sex = 'F');
-ALTER TABLE Instructors ADD CONSTRAINT ins_types
CHECK (ins_type = 'Professor' OR ins_type = 'Associate Professor'
        OR ins_type='Assistant' OR ins_type='Professor Assistant');
-ALTER TABLE do_projects ADD CONSTRAINT grade_check
CHECK (grade >= 0);
ASSERTION
-CREATE OR REPLACE FUNCTION salary_assert()
  RETURNS TRIGGER
  LANGUAGE PLPGSQL
  AS
  $$
BEGIN
      IF NEW.Salary < 0 THEN
            RAISE EXCEPTION 'Salary can not be a negative value Salary: %', NEW.Salary
USING HINT = 'Please check your salary value';
      END IF;
      RETURN NEW;
END;
      $$

CREATE TRIGGER salary_assert
BEFORE INSERT
ON Employees
FOR EACH ROW
EXECUTE PROCEDURE salary_assert();
-CREATE OR REPLACE FUNCTION member_type_assert()
  RETURNS TRIGGER
```

```
 LANGUAGE PLPGSQL
 AS
 $$
BEGIN
      IF NEW.member_type <> 'Student' OR NEW.member_type <> 'Employee' OR
NEW.member_type <> 'Instructor' THEN
            RAISE EXCEPTION 'Invalid member type entered Member type: %',
NEW.mem_type USING HINT = 'Please check your member type';
      END IF;
      RETURN NEW;
END;
      $$

CREATE TRIGGER member_type_assert
BEFORE INSERT
ON Members
FOR EACH ROW
EXECUTE PROCEDURE member_type_assert();
-CREATE OR REPLACE FUNCTION gpa_assert()
 RETURNS TRIGGER
 LANGUAGE PLPGSQL
 AS
 $$
BEGIN
      IF NEW.gpa < 0 THEN
            RAISE EXCEPTION 'GPA can not be a negative value GPA: %', NEW.GPA USING
HINT = 'Please check your GPA value';
      END IF;
      RETURN NEW;
END;
      $$
```

```
CREATE TRIGGER gpa_assert

BEFORE INSERT

ON Students

FOR EACH ROW

EXECUTE PROCEDURE gpa_assert();
```

INSERT-UPDATE- DELETE STATEMENTS

```
UPDATE Departments SET dean_ssn = 504014 WHERE dept_id = 6
UPDATE Members SET member_type = 'Employee '  WHERE ssn = 518022
UPDATE Employee  SET Salary = 10000 WHERE ssn = 201001
DELETE FROM Departments WHERE dept_id = 7
DELETE FROM Users WHERE user_id = 999999996
DELETE FROM Instructors WHERE ssn = 504011
```

SQL STATEMENTS

1 TABLE

- **2 den fazla user'a sahip olan grupları id göre azalan olarak sıralayan ve id ile beraber kaçar user bulunduğu bilgisini döndüren sorgu.**

```
SELECT Group_id, COUNT(*) AS Eleman_Sayısı
FROM Group_members
GROUP BY Group_id
HAVING COUNT(*) > 2
ORDER BY Group_id
```

- **Bir user'ın onaylanmış arkadaş sayısını döndüren sorgu.**

```
SELECT User_id_to AS User_id
FROM Connections
WHERE User_id_from = 999999993 AND is_confirm = true

UNION

SELECT User_id_from AS User_id
FROM Connections
WHERE User_id_to = 999999993 AND is_confirm = true
```

- **Ege üniversitesininki tüm mühendislik bölümlerini isimlerine göre sıralanmış olarak döndüren sorgu.**

```
SELECT Dept_name
FROM Departments
WHERE Dept_name LIKE '%Mühendisliği' and Uni_id = 2
ORDER BY Dept_name
```

2 TABLE

- **Group'ların host' u dışındaki elemanlarını döndüren sorgu**

```
SELECT Group_name, User_id
FROM (Linkedin_Groups AS LG INNER JOIN Group_members AS GM ON LG.group_id
= GM.group_id)
WHERE User_id IN (SELECT User_id FROM Group_members EXCEPT(
SELECT Host_user_id
FROM Linkedin_Groups))
ORDER BY Group_name
```

- **Memberların endorse'lanmış skilllerinin sadece 2'den fazla kişilerin endorse'ladığı skilleri döndüren ve kaç kişinin endorse'ladığını döndüren sorgu**

```
SELECT skill_id, approved_ssn, COUNT(*) AS Skill_Count
FROM (Skills_endorse INNER JOIN Members ON Skills_endorse.Approved_Ssn =
Members.Ssn)
GROUP BY (skill_id, approved_ssn)
HAVING COUNT(*)>2
ORDER BY approved_ssn
```

- **Öğrencilerin bir dersten almış olduğu toplam projelerin ortalaması 60 üzeri olanları bulma**

```
SELECT Student_Ssn, MCourse_id, AVG(Grade)
FROM (Students AS S INNER JOIN Do_Projects AS D ON S.Ssn = D.Student_Ssn)
WHERE Dept_id = 1
Group BY Student_Ssn, MCourse_id
HAVING AVG(Grade) > 60
ORDER BY Student_Ssn.    (Aynı dersten birden fazla proje alma durumunu popule et)
```

- **504001 ssn'lı Instructor' un dekanlık yaptığı departmanda çalışan toplam instructor sayısını döndüren sorgu**

```
SELECT Dept_id, COUNT(*)
FROM Instructors
WHERE Dept_id IN (SELECT D.Dept_id
FROM (Departments AS D INNER JOIN Instructors AS I ON D.Dean_Ssn = I.Ssn)
WHERE Dean_Ssn = 504001)
GROUP BY Dept_id
```

3 TABLE

- **‘Şu isimli’ firmanın açmış olduğu tüm işlerden birine katılmış olan öğrencilerin ortalaması 3.0 ve üzeri olan öğrencileri geri hangi işe başvurdukları ile döndüren sorgu.**

```
SELECT Comp_name, Advert_name, Student_ssn, GPA
FROM ((Companys AS CMP NATURAL JOIN Job_Adverts AS JA) AS CJ NATURAL JOIN
(Job_Apps AS JAP NATURAL JOIN Students AS S) AS JS) J
WHERE comp_name = '' AND GPA > 3.0 (Popule edilmesi gerekiyor)
```

- **‘şu grubun’ kurucusunun ‘şu tarihten’ sonra yayınlanmış postlarından en çok beğeni alan postunun içeriğini döndüren sorgu**

```
SELECT J.post_id, post_text, COUNT(*)

FROM( Linkedin_groups AS lg INNER JOIN Linkedin_posts AS lp ON lg.host_user_id =
lp.host_user_id) AS J

INNER JOIN post_likes as pl ON J.post_id = pl.post_id

WHERE group_name = 'Elektrik Severler' AND J.publish_date > '2020-01-01'

AND J.post_id = pl.post_id

GROUP BY J.post_id

ORDER BY COUNT(*) desc

LIMIT 1
```

- **University’lerin açmış olduğu department’lardaki en yüksek ortalamalı öğrencinin ortalaması 3.6’dan fazla olanları döndüren sorgu**

```
SELECT Dept_id, MAX(GPA) AS Max_GPA
FROM (University NATURAL JOIN Departments) NATURAL JOIN (Students NATURAL
JOIN Members)
GROUP BY dept_id
HAVING MAX(GPA) > 3.6
ORDER BY dept_id
```

ORIGINAL

- **‘Şu projeyi veren’ instructor’un verdiği derslerden sadece moodle’ u açılan derslerin moodle üzerinden verdiği projeleri döndüren sorgu**

```
SELECT *

FROM Projects

WHERE mcourse_id = (SELECT mcourse_id

FROM Moodle_courses

WHERE course_id = (SELECT course_id FROM (Instructors NATURAL JOIN Courses)
WHERE course_name = 'Network' AND dept_id = 1 LIMIT 1))
```

- **User' a sahip company'lerin connection kurduğu userlardan student olanının 'Database' dersi üzerinde açılan projelerde elde ettiği puanların ortalaması 60 üzeri olanları geri döndüren sorgu**

```
SELECT student_ssn, AVG(grade)

FROM do_projects

WHERE student_ssn IN (SELECT Ssn

FROM Members

WHERE member_type = 'Student' AND user_id IN (SELECT user_id_to

FROM Connections

WHERE user_id_from = (SELECT user_id

FROM Users

WHERE user_id = (SELECT user_id

FROM Companys

WHERE user_id IS NOT NULL AND comp_name = 'Facebook'))))

GROUP BY student_ssn

HAVING AVG(grade) > 60
```

- **Bir firmanın skill_id = 3 i birden fazla endorse olmuş memberlardan sadece student ve employee olanları döndüren sorgu. (Skill Name alabiliriz)**

```
SELECT Ssn, Fname, LName, Member_type, user_id
FROM Members
WHERE ssn IN
(SELECT approved_ssn
FROM Skills_endorse
WHERE skill_id = 3
GROUP BY (skill_id, approved_ssn)
HAVING COUNT(*) > 1
) AND member_type IN ('Student', 'Employee') AND user_id IS NOT NULL (Popule
edilmesi lazım)
```

- **Firmaların açmış olduğu iş ilanlarına başvuran student'lardan sadece gpa'yi 3.0 ve üzeri olan sorguları döndüren sorgu**

```
SELECT DISTINCT ON (student_ssn)(student_ssn), comp_name
FROM ((Job_apps NATURAL JOIN Companys) NATURAL JOIN Students)
WHERE student_ssn IN
(SELECT ssn
FROM Students
WHERE GPA > 3.0)
```

- **Studentlerın almış olduğu derslerden açılan projeleri döndüren sorgu**

```
SELECT Pro.project_name, Pro.course_id, Pro.course_name, (Ins.Fname, Ins.LName)
AS Ins_Name

FROM (Projects NATURAL JOIN Courses) AS Pro, (Instructors NATURAL JOIN
Members) AS Ins

WHERE mcourse_id IN

(SELECT mcourse_id

FROM (((Take_courses NATURAL JOIN Courses) AS J INNER JOIN Students AS S ON
J.student_ssn = S.ssn) NATURAL JOIN Moodle_courses))

AND Pro.ins_ssn = Ins.ssn

ORDER BY  Pro.project_name, Pro.course_id
```