



CS 353: DATABASE SYSTEMS

Group 3: Online Course Platform

Project Design Report

Group Members:

Işık Özsoy 21703160 Section: 2

Defne Betül Çiftci 21802635 Section: 3

Şebnem Uslu 21802068 Section: 2

Melike Fatma Aydoğan 21704043 Section: 2

Instructors: Özgür Ulusoy, Uğur Güdükbay

Teaching Assistant: Mustafa Can Çavdar

Design Report

April 2, 2021

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Database Systems course CS353.

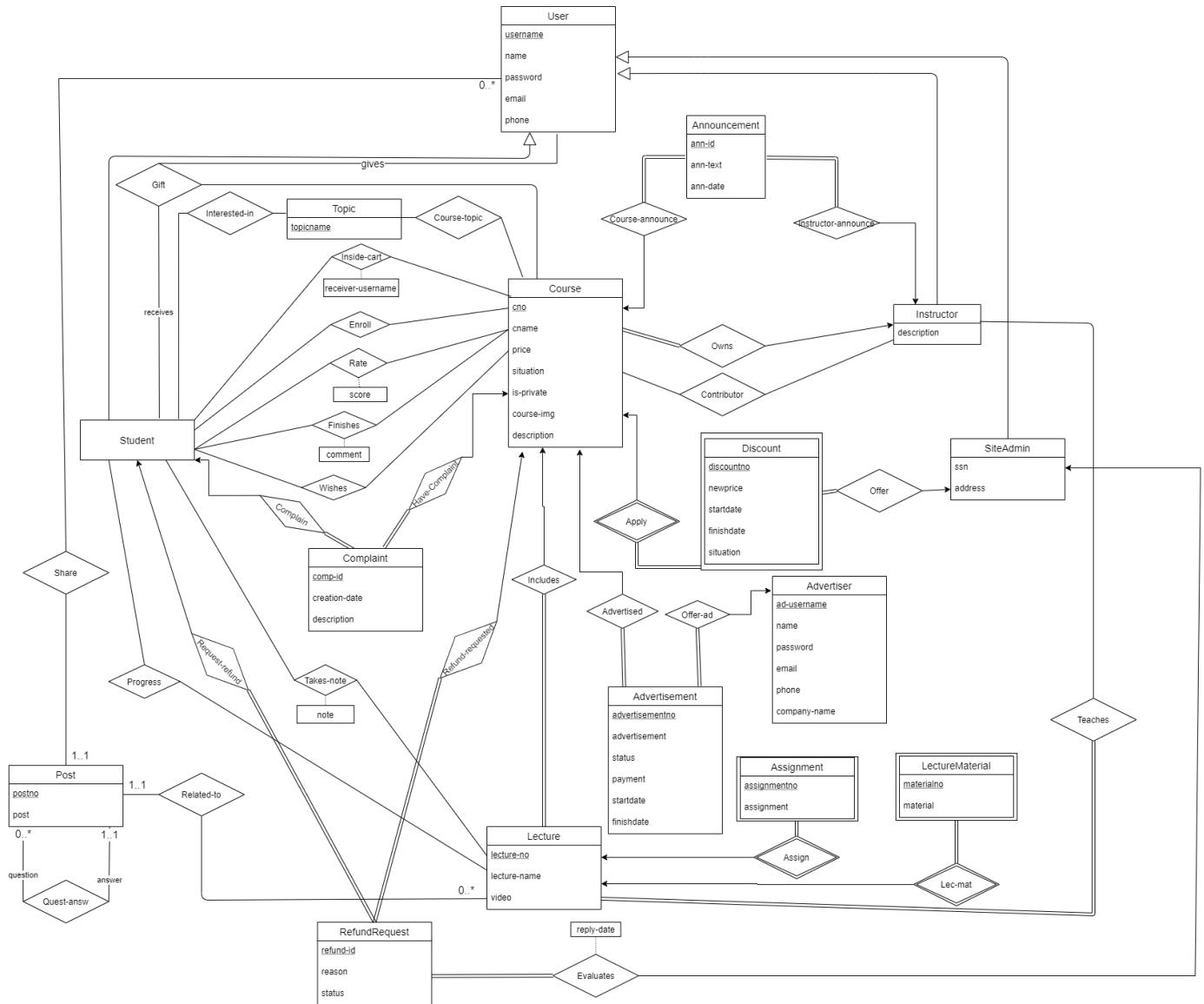
Table of Contents

Revised E/R Model	3
Revised E/R Diagram	3
Changes Made In E/R Diagram	3
Relation Schemas	4
User	4
Student	4
Instructor	5
SiteAdmin	5
Advertiser	5
Course	6
Gift	6
Complaint	7
Takes-note	7
Wishes	7
Finishes	8
Rate	8
Enroll	9
Announcement	9
Contributor	10
Lecture	10
Progress	10
Teaches	11
Topic	11
Course-topic	11
Interested-in	12
Discount	12
Post	13
Quest-answ	13
Advertisement	13
RefundRequest	14
Evaluates	14
Assignment	15
LectureMaterial	15
Inside-cart	16
Interface Designs and Corresponding SQL Statements	16
Main Page Before Login (Scroll Down)	17
SQL Statement:	18
Student Signup Page	18
SQL Statement:	18
Instructor Signup	19
SQL Statement:	19
Advertiser Signup	19

SQL Statement:	19
Login Page for Advertiser	20
SQL Statement:	20
Login Page for Other User Types	20
SQL Statement:	20
Main Page of Student After Login (Scroll Down)	21
SQL Statement:	21
SQL Statement:	22
SQL Statement:	22
Main Page of Instructor After Login	23
SQL Statement:	23
Course Info Page	24
SQL Statement:	24
Shopping Cart Page and with Gift Property Page	25
SQL Statement:	25
Notifications	26
SQL Statement:	27
Student's Courses Page	27
SQL Statement:	27
Wishlist Page	28
SQL Statement:	28
Watching Lecture Page	29
SQL Statement:	29
Course Overview Page	29
SQL Statement:	30
Completing a Course Page	30
SQL Statement:	30
Course Refund Request	31
SQL Statement:	31
Course Ask Question	32
SQL Statement:	32
Note Page	33
SQL Statement:	33
Course Page for Advertisers - Additional Functionality	33
Advertise Page for Advertisers - Additional Functionality	34
SQL Statement:	34
Notification Page for Instructors that Shows the Advertisement Offer - Additional Functionality	35
SQL Statement:	35
Advertisement Requests for Advertisers - Additional Functionality	36
SQL Statement:	36
Extra Pages Made for Instructor Which Are Unnecessary for Design Report	36
Add Lecture Material Page	38
Systems and Technologies	39

I. Revised E/R Model

Revised E/R Diagram



Changes Made In E/R Diagram

- Primary key for RefundRequest (refund-id) was added.
- Participation of RefundRequest in the Request-refund relation became total.
- Advertiser is now not a User, but a separate entity with mostly same attributes except for company-name and ad-username.
- Lecture became a normal entity instead of a weak entity.
- Evaluates relation was added with attribute reply-date which shows the date that Admin evaluates the request. This relation demonstrates which request is evaluated by who and when.
- Complain relation was transformed from a three-way relation into two different binary relations named “Complain” that connects Complaint and Student and “Have-Complain” that connects Course and Complaint.

- Offer-ad relation was transformed from a three-way relation into two different binary relations named “Offer-ad” that connects Advertisement and Advertiser and “Advertised” that connects Advertisement and Course.
- Request-refund relation was transformed from a three-way relation into two different binary relations named “Request-refund” that connects Student and RefundRequest and “Refund-requested” that connects Course and RefundRequest.
- Takes-note is now not connected to Course but connected to Lecture.
- A many-to-many “In-cart” relation is added between Student and Course that has an attribute named “receiver-email”.
- Participation of Announcement to Course and Instructor became total.

II. Relation Schemas

User

Relational Model: User (username, name, password, email, phone)

Candidate Key:

username
email

Primary Key: username

Table Definition:

```
create table User (
    username char (50),
    name char (50),
    password char(50),
    email char (50),
    phone char (50),
    primary key (username)
)
engine=InnoDB;
```

Student

Relational Model: Student (username)

Candidate Key:

username

Primary Key: username

Foreign Key:

username referencing User

Table Definition:

```
create table Student (
    username char (50),
    primary key (username),
    foreign key (username) references User
)
engine=InnoDB;
```

Instructor

Relational Model: Instructor (username, description)

Candidate Key:

username

Primary Key: username

Foreign Key:

username referencing User

Table Definition:

```
create table Instructor (
    username char(50),
    description char(1000),
    primary key (username),
    foreign key (username) references User
)
engine=InnoDB;
```

SiteAdmin

Relational Model: SiteAdmin (username, ssn, address)

Candidate Key:

username

ssn

Primary Key: username

Foreign Key:

username referencing User

Table Definition:

```
create table SiteAdmin (
    username char (50),
    ssn char (20),
    address char (100),
    primary key (username),
    foreign key (username) references User
)
engine=InnoDB;
```

Advertiser

Relational Model: Advertiser (ad-username, name, password, email, phone, company-name)

Candidate Key:

ad-username

email

Primary Key: username

Table Definition:

```
create table Advertiser (
    ad-username char (50),
    name char (50),
    password char (50),
    email char (50),
```

```
    phone char (50),  
    company-name char (100),  
    primary key (ad-username)  
)  
engine=InnoDB;
```

Course

Relational Model: Course (cno, owner-username, cname, price, situation, is-private, course-img, description)

Candidate Key:

cno

Primary Key: cno

Foreign Key:

owner-username referencing Instructor (username)

Table Definition:

```
create table Course (  
    cno int,  
    owner-username char (50),  
    cname char (50),  
    price numeric (6,2),  
    situation smallint,  
    is-private smallint,  
    course-img varchar(512),  
    description varchar (1000),  
    primary key (cno),  
    foreign key (owner-username) references Instructor (username)  
)  
engine=InnoDB;
```

Gift

Relational Model: Gift (sender-username, receiver-username, cno)

Candidate Key:

sender-username, receiver-username, cno

Primary Key: sender-username, receiver-username, cno

Foreign Key:

sender-username referencing User (username)

receiver-username referencing Student (username)

cno referencing Course

Table Definition:

```
create table Gift (  
    sender-username int,  
    receiver-username int,  
    cno int,  
    primary key (sender-username, receiver-username, cno),  
    foreign key (sender-username) references User (username),  
    foreign key (receiver-username) references Student (username),
```

```
    foreign key (cno) references Course  
)  
engine=InnoDB;
```

Complaint

Relational Model: Complaint (comp-id, s-username, cno, creation-date, description)

Candidate Key:

comp-id

Primary Key: comp-id

Foreign Key:

s-username referencing Student (username)

cno referencing Course

Table Definition:

```
create table Complaint-made (  
    comp-id int,  
    s-username char(50),  
    cno int,  
    creation-date date,  
    description char(1000),  
    primary key (comp-id),  
    foreign key (s-username) references Student (username),  
    foreign key (cno) references Course  
)
```

```
engine=InnoDB;
```

Takes-note

Relational Model: Takes-note (s-username, lecture-no, note)

Candidate Key:

s-username, lecture-no

Primary Key: s-username, lecture-no

Foreign Key:

s-username referencing Student (username)

cno referencing Course

Table Definition:

```
create table Takes-Note (  
    s-username char (50),  
    lecture-no int,  
    note varchar (4000),  
    primary key (s-username, lecture-no),  
    foreign key (s-username) references Student (username)  
)
```

```
engine=InnoDB;
```

Wishes

Relational Model: Wishes (s-username, cno)

Candidate Key:

s-username, cno

Primary Key: s-username, cno

Foreign Key:

s-username referencing Student (username)

cno referencing Course

Table Definition:

create table Wishes (

 s-username char(50),

 cno int,

 primary key (s-username, cno),

 foreign key (s-username) references Student (username),

 foreign key (cno) references Course

)

engine=InnoDB;

Finishes

Relational Model: Finishes (s-username, cno, comment)

Candidate Key:

s-username, cno

Primary Key: s-username, cno

Foreign Key:

s-username referencing Student (username)

cno referencing Course

Table Definition:

create table Finishes (

 s-username char(50),

 cno int,

 comment varchar (150),

 primary key (s-username, cno),

 foreign key (s-username) references Student (username),

 foreign key (cno) references Course

)

engine=InnoDB;

Rate

Relational Model: Rate (s-username, cno, score)

Candidate Key:

s-username, cno

Primary Key: s-username, cno

Foreign Key:

s-username referencing Student (username)

cno referencing Course

Table Definition:

create table Rate (

 s-username char (50),

 cno int,

```

        score int,
        primary key (s-username, cno),
        foreign key (s-username) references Student (username),
        foreign key (cno) references Course
)
engine=InnoDB;

```

Enroll

Relational Model: Enroll (s-username, cno)

Candidate Key:

s-username, cno

Primary Key: s-username, cno

Foreign Key:

s-username referencing Student (username)

cno referencing Course

Table Definition:

```

create table Enroll (
        s-username char (50),
        cno int,
        primary key (s-username, cno),
        foreign key (s-username) references Student (username),
        foreign key (cno) references Course
)
engine=InnoDB;

```

Announcement

Relational Model: Announce (ann-id, cno, i-username, ann-tex, ann-date)

Candidate Key:

ann-id

Primary Key: ann-id

Foreign Key:

cno referencing Course

username referencing Instructor

Table Definition: create table Announcement (

```

        ann-id char(20),
        s-username char(50),
        cno int,
        ann-text varchar(1000),
        ann-date date,
        primary key (ann-id),
        foreign key (i-username) references Instructor (username),
        foreign key (cno) references Course
)
engine=InnoDB;

```

Contributor

Relational Model: Contributor (cno, i-username)

Candidate Key:

cno, i-username

Primary Key: cno, i-username

Foreign Key:

cno referencing Course

i-username referencing Instructor (username)

Table Definition:

```
create Contributor (
    cno int,
    i-username char(50),
    primary key (cno, i-username),
    foreign key (cno) references Course,
    foreign key (i-username) references Instructor (username)
)
engine=InnoDB;
```

Lecture

Relational Model: Lecture (lecture-no, lecture-name, video, cno)

Candidate Key:

lecture-no

Primary Key: lecture-no

Foreign Key:

cno referencing Course

Table Definition:

```
create table Lecture (
    lecture-no int,
    lecture-name char (200),
    video char (100),
    cno int,
    primary key (lecture-no),
    foreign key (cno) references Course (cno)
)
engine=InnoDB;
```

Progress

Relational Model: Progress (s-username, lecture-no)

Candidate Key:

s-username, lecture-no

Primary Key: s-username, lecture-no

Foreign Key:

s-username referencing Student (username)

lecture-no referencing Lecture

Table Definition:

```
create table Progress (
```

```

    s-username char(50),
    lecture-no int,
    primary key (s-username, lecture-no),
    foreign key (s-username) references Student (username),
    foreign key (lecture-no) references Lecture (lecture-no)
)
engine=InnoDB;

```

Teaches

Relational Model: Teaches (i-username, lecture-no)

Candidate Key:

i-username, lecture-no

Primary Key: i-username, lecture-no

Foreign Key:

lecture-no referencing Lecture

i-username referencing Instructor (username)

Table Definition:

```

create table Teaches (
    i-username char(50),
    lecture-no int,
    primary key (i-username, lecture-no),
    foreign key (i-username) references Instructor (username),
    foreign key (lecture-no) references Lecture (lecture-no)
)
engine=InnoDB;

```

Topic

Relational Model: Topic (topicname)

Candidate Key:

topicname

Primary Key: topicname

Table Definition:

```

create table Topic(
    topicname char(100),
    primary key ( topicname )
)
engine=InnoDB;

```

Course-topic

Relational Model: Course-topic (cno, topicname)

Candidate Key:

cno, topicname

Primary Key: cno, topicname

Foreign Key:

cno referencing Course

topicname referencing Topic

Table Definition:

```
create table Course-topic(
    cno int,
    topicname char(100),
    primary key( cno, topicname ),
    foreign key (cno) references Course (cno),
    foreign key ( topicname ) references Topic (topicname)
)
engine=InnoDB;
```

Interested-in

Relational Model: Interested-in (s-username, topicname)

Candidate Key:

s-username, topicname

Primary Key: s-username, topicname

Foreign Key:

s-username referencing Student (username)

topicname referencing Topic

Table Definition:

```
create table Interested-in(
    s-username char (50),
    topicname char (100),
    primary key ( s-username, topicname ),
    foreign key ( s-username ) references Student (username),
    foreign key ( topicname ) references Topic (topicname)
)
engine=InnoDB;
```

Discount

Relational Model: Discount (discountno, newprice, startdate, finishdate, situation, cno, admin-username)

Candidate Key:

discountno

Primary Key: discountno

Foreign Key:

cno referencing Course

admin-username referencing SiteAdmin (username)

Table Definition:

```
create table Discount(
    discountno int,
    newprice numeric(6,2),
    startdate Date,
    finishdate Date,
    situation smallint,
    cno int,
    admin-username char(50),
```

```

primary key ( discountno ),
foreign key ( cno ) references Course ( cno ),
foreign key ( admin-username ) references SiteAdmin ( username )
)
engine=InnoDB;

```

Post

Relational Model: Post (postno, lecture-no, post, username)

Candidate Key:

postno

Primary Key: postno

Table Definition:

```

create table Post (
    postno int,
    lecture-no int,
    post char(200),
    username char(50),
    primary key (postno),
    foreign key (username) references User (username),
    foreign key (lecture-no) references Lecture (lecture-no)
)
engine=InnoDB;

```

Quest-answ

Relational Model: Quest-answ (answer-no, question-no)

Candidate Key:

answer-no

Primary Key: answer-no

Foreign Key:

answer-no referencing Post (postno)

question-no referencing Post (postno)

Table Definition:

```

create table Quest-answ (
    answer-no int,
    question-no int,
    primary key ( answer-no ),
    foreign key (answer-no) references Post (postno),
    foreign key (question-no) references Post (postno)
)
engine=InnoDB;

```

Advertisement

Relational Model: Advertisement (advertisementno, ad-username, cno, advertisement, status, payment, startdate, finishdate)

Candidate Key:

advertisementno

Primary Key: advertisementno

Foreign Key:

ad-username referencing Advertiser (username)
cno referencing Course

Table Definition:

```
create table Advertisement (
    advertisementno int,
    ad-username char (50),
    cno int,
    advertisement varchar(512),
    status smallint,
    payment numeric(20,2),
    startdate Date,
    finishdate Date,
    primary key (advertisementno),
    foreign key (ad-username) references Advertiser (username),
    foreign key (cno) references Course (cno)
)
engine=InnoDB;
```

RefundRequest

Relational Model: RefundRequest (refund-id, s-username, cno, reason, status)

Candidate Key:

refund-id

Primary Key: refund-id, s-username, cno

Foreign Key:

s-username referencing Student (username)
cno referencing Course

Table Definition:

```
create table RefundRequest (
    refund-id int,
    s-username char (50),
    cno int,
    reason char (500),
    status smallint default 0,
    primary key ( refund-id),
    foreign key (s-username) references Student (username),
    foreign key (cno) references Course (cno)
)
engine=InnoDB;
```

Evaluates

Relational Model: Evaluates (refund-id, admin-username, reply-date)

Candidate Key:

refund-id

Primary Key: refund-id

Foreign Key:

admin-username referencing SiteAdmin
refund-id referencing RefundRequest

Table Definition:

```
create table Evaluates (
    refund-id int,
    admin-username char(50),
    reply-date Date,
    primary key (refund-id),
    foreign key ( admin-username ) references SiteAdmin ( username ),
    foreign key ( refund-id ) references RefundRequest ( refund-id )
)
engine=InnoDB;
```

Assignment

Relational Model: Assignment (assignmentno, assignment, lecture-no)

Candidate Key:

assignmentno

Primary Key: assignmentno

Foreign Key:

lecture-no referencing Lecture

Table Definition:

```
create table Assignment(
    assignmentno int,
    assignment longblob,
    lecture-no int,
    primary key (assignmentno),
    foreign key (lecture-no) references Lecture (lecture-no)
)
engine=InnoDB;
```

LectureMaterial

Relational Model: LectureMaterial (materialno, material, lecture-no)

Candidate Key:

materialno

Primary Key: materialno

Foreign Key: lecture-no referencing Lecture

Table Definition:

```
create table LectureMaterial (
    materialno int,
    material longblob,
    lecture-no int,
    primary key ( materialno ),
    foreign key ( lecture-no ) references Lecture (lecture-no)
)
engine=InnoDB;
```

Inside-cart

Relational Model: Inside-cart (cno, username, receiver-username)

Candidate Key:

cno, username

Primary Key: cno, username

Foreign Key:

cno referencing Course

cart-id referencing Cart

receiver-username referencing Student (username)

Table Definition:

```
create table Inside-cart (
    cno int,
    username char (50),
    receiver-username char (50),
    primary key (cno, username),
    foreign key (cno) references Course (cno),
    foreign key (username) references User (username),
    foreign key (receiver-username) references Student (username)
)
engine=InnoDB;
```

III. Interface Designs and Corresponding SQL Statements

As some of the mockups were not obligatory and, in our report, is there mostly for reference purposes, we did not write the queries for those. All the other necessitated designs are placed alongside their queries.

Main Page Before Login (Scroll Down)

Window Title

MAYACAT Teach at Mayacat [LOG IN](#) [SIGN UP](#)

Start learning today! ➤ Broaden your worldview ➤ With certified experts ➤ Categories

Most popular



Build an education app Learn to Code in Python... The Tempest by Shakes... Calculus for Beginners

Highest Ranking



Window Title

MAYACAT Teach at Mayacat [LOG IN](#) [SIGN UP](#)

Start learning today! ➤ Broaden your worldview ➤ With certified experts ➤ Categories

Highest ranking



Learn to Code in Python... Database Systems Calculus for Beginners How to Code in MySQL

[Get started](#)

Window Title

MAYACAT Teach at Mayacat [LOG IN](#) [SIGN UP](#)

Start learning today! ➤ Broaden your worldview ➤ With certified experts ➤ Categories

[Get started](#)

Transform your life through education

[Business](#)

SQL Statement:

- Used to demonstrate the highest ranked courses.

```
select c.cno, c cname  
from Course as c  
natural join (  
    select r.cno  
    from Rate as r  
    group by r.cno  
    order by avg(r.score)) as t  
limit 4;
```

- Used to demonstrate the most popular courses.

```
select c.cno, c cname  
from Course as c  
natural join (  
    select e.cno  
    from Enroll as e  
    group by e.cno  
    order by count(e.s-username)) as t  
limit 4;
```

Student Signup Page



SQL Statement:

```
insert into User values ("mayazsy", "Maya", "123456", "mayaozsoy@gmail.com", "05555555555");  
insert into Student ("mayazsy");
```

Instructor Signup

Window Title

LOG IN

MAYACAT

Student Instructor Advertiser

✉ defne@gmail.com

👤 dbetulcift

★ Defne Betul

📞 0539 662 42 99

🔒 *****

💬 Hi, I am Defne and my passion is web develop...

SIGN UP

SQL Statement:

```
insert into User values ("dbetulcift", "Defne Betul", "Db1234", "defne@gmail.com", "05396624299");
insert into Instructor values ("dbetulcift", "Hi, I am Defne and my passion is web development. I have been teaching it for 15 years.");
```

Advertiser Signup

Window Title

LOG IN

MAYACAT

Student Instructor Advertiser

✉ isikozsoy@bilkent.edu.tr

👤 isikozsoy

★ İşik

📞 0555 555 55 55

🔒 *****

🌐 Bilkent Holding

SIGN UP

SQL Statement:

```
insert into Advertiser values ("isikozsoy", "İşik", "mm19kk", "isikozsoy@bilkent.com", "05459554545",
" Bilkent Holding");
```

Login Page for Advertiser

The screenshot shows a web browser window titled "Window Title". At the top right are standard window controls (minimize, maximize, close). On the right side of the page is a "SIGN UP" button. In the center, there is a logo of a black cat sitting and the text "MAYACAT". Below the logo is a radio button group for "Student / Instructor / Admin" and "Advertiser", with "Advertiser" selected. There are two input fields: one for "Email" containing "isikozsoy" and another for "Password" containing "*****". Below these fields is a "Forgot password?" link. At the bottom is a "LOG IN" button.

SQL Statement:

```
select password  
from Advertiser  
where ad-username ="isikozsoy";
```

Login Page for Other User Types

The screenshot shows a web browser window titled "Window Title". On the right side is a "SIGN UP" button. In the center is the Mayacat logo and text. Below it is a radio button group for "Student / Instructor / Admin" and "Advertiser", with "Advertiser" selected. There are two input fields: one for "Email" containing "mayazsy" and another for "Password" containing "*****". Below these fields is a "Forgot password?" link. At the bottom is a "LOG IN" button.

SQL Statement:

```
select password  
from User  
where username ="mayazsy";
```

Main Page of Student After Login (Scroll Down)

The screenshot shows the Mayacat student dashboard. At the top, there's a navigation bar with links for Programming, Database, and Python. Below the navigation, there's a section titled "Continue your lectures" featuring two cards: "Build an education app" and "Learn to Code in Python...". Underneath this, there's a section titled "My courses" showing four course thumbnails. A sidebar on the right lists categories: Programming, Mathematics, Literature, Music, Design, Finance, and Art.

SQL Statement:

- Used to demonstrate the courses that are not finished yet.

```
select cno, cname, course-img  
from Enroll  
where s-username="mayazsy" and cno not in (  
            select cno  
            from Finishes  
            where s-username="mayazsy");
```
- Used to demonstrate all the courses of the student.

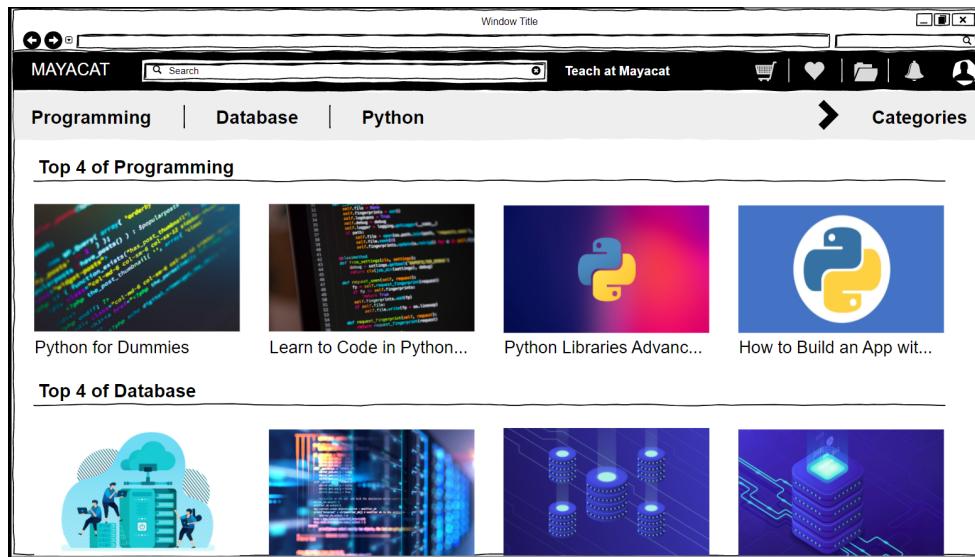
```
select cno, cname, course-img  
from Enroll  
where s-username="mayazsy";
```

This screenshot is similar to the one above, showing the Mayacat student dashboard. However, a dropdown menu has been triggered from the "Categories" link in the top right. The visible items in the dropdown are: Programming, Mathematics, Literature, Music, Design, Finance, and Art.

- Categories show all the topics in the platform. At the top of the page, the topics “Programming”, “Database” and “Python” are selected.

SQL Statement:

```
select distinct topicname
from Topic;
```



- When the student scrolls down the page, s/he sees the courses, which belong to the selected topics, separately.

SQL Statement:

```
with specified-courses (cno) as
```

```
(select cno
  from Course-topic
  where topicname = "Programming")
select cno, cname, course-img
from (select cno, sum(score) as tot-rate
      from specified-courses natural join Rate
      group by cno) natural join Course;
```

```
with specified-courses (cno) as
```

```
(select cno
  from Course-topic
  where topicname = "Database")
select cno, cname, course-img
from (select cno, sum(score) as tot-rate
      from specified-courses natural join Rate
      group by cno) natural join Course;
```

```
with specified-courses (cno) as
```

```
(select cno
```

```
from Course-topic
where topicname = "Python")
select cno, cname, course-img
from (select cno, sum(score) as tot-rate
      from specified-courses natural join Rate
      group by cno) natural join Course;
```

Main Page of Instructor After Login

The screenshot shows the MAYACAT application interface for an instructor. At the top, there's a header bar with 'MAYACAT' and a search bar. Below the header, the title 'Window Title' is visible. The main content area is divided into sections:

- My courses:** Displays four course cards:
 - Python Introductory (with Python logo icon)
 - Learn to Code in Python... (with code editor icon)
 - Build an education app (with code editor icon)
 - Database Systems (with server icon)
- My Contributions:** Displays two contribution cards:
 - How to Code in MySQL (with MySQL database icon)
 - Open Source Database ... (with server icon)

SQL Statement:

```
select cno, cname, course-img
from Course
where owner-username = "dbetulcift";

select cno, cname
from Course natural join Contributor
where i-username = "dbetulcift";
```

Course Info Page

The screenshot shows a course page titled "Build an education app" by Uğur Güdükbay & Özgür Ulusoy. The page includes a "DESCRIPTION" section with a detailed course summary, an "INCLUDING" section listing course components like video lectures and assignments, and a sidebar featuring a "Bubble Tea" advertisement and a "Code" snippet. The price is listed as ₺30.00 with an "ADD TO CART" button.

Build an education app
by Uğur Güdükbay & Özgür Ulusoy

DESCRIPTION
Relational data model. Entity/Relationship model. Relational Algebra. Structured Query Language, SQL. Relational database design. Tree-structured and hash-based indexing. Query processing and optimization. Transaction management, concurrency control and recovery issues in database systems. Development of a relational database application as a term project.

INCLUDING

- 37.5 hours on-demand video lectures
- 6 assignments
- 4 lecture materials

₺30.00

ADD TO CART

This screenshot shows the same course page but with a red border around the main content area. It includes a "COMMENTS" section with three user reviews: "mayazsy: This course helped me a lot! Definitely recommended", "melikeee: Liked the course but it is too fast", and "sebnemsl: It was fine". The rest of the page structure is identical to the first screenshot.

Build an education app
by Uğur Güdükbay & Özgür Ulusoy

INCLUDING

- 37.5 hours on-demand video lectures
- 6 assignments
- 4 lecture materials
- Full lifetime access
- Certificate of completion

COMMENTS

mayazsy: This course helped me a lot! Definitely recommended

melikeee: Liked the course but it is too fast

sebnemsl: It was fine

₺30.00

ADD TO CART

WISHLIST

SQL Statement:

- Assume that the cno of the course named Build an education app is 1.
select cname, owner-username, price, description, name
from Course, User
when cno = 1;

```
select i-username, name  
from Contributor, User  
where cno = 1 and i-username = 'mayazsy';
```

--To demonstrate the comments made by the student who has finished the course.
select comment
from Finishes
where cno=15;

```
select cno, avg(score) as avg-rate  
from Rate  
where cno = 1;
```

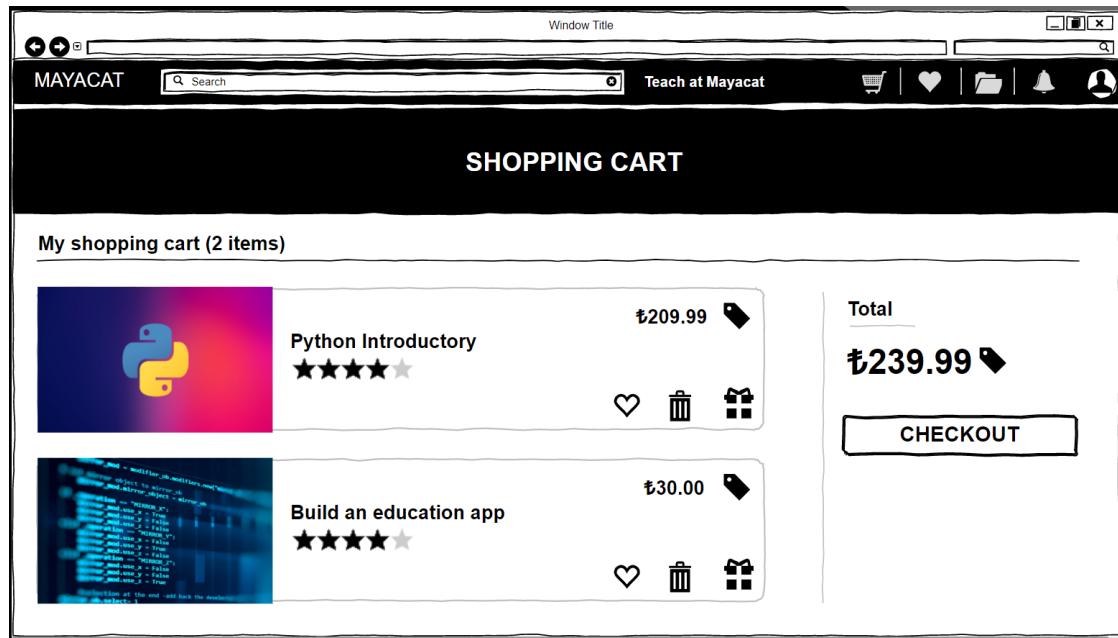
- If the student adds the item to the cart, the following query will be used (assume that the course no is 9). Null represents the username of the student that will receive the course as a gift.

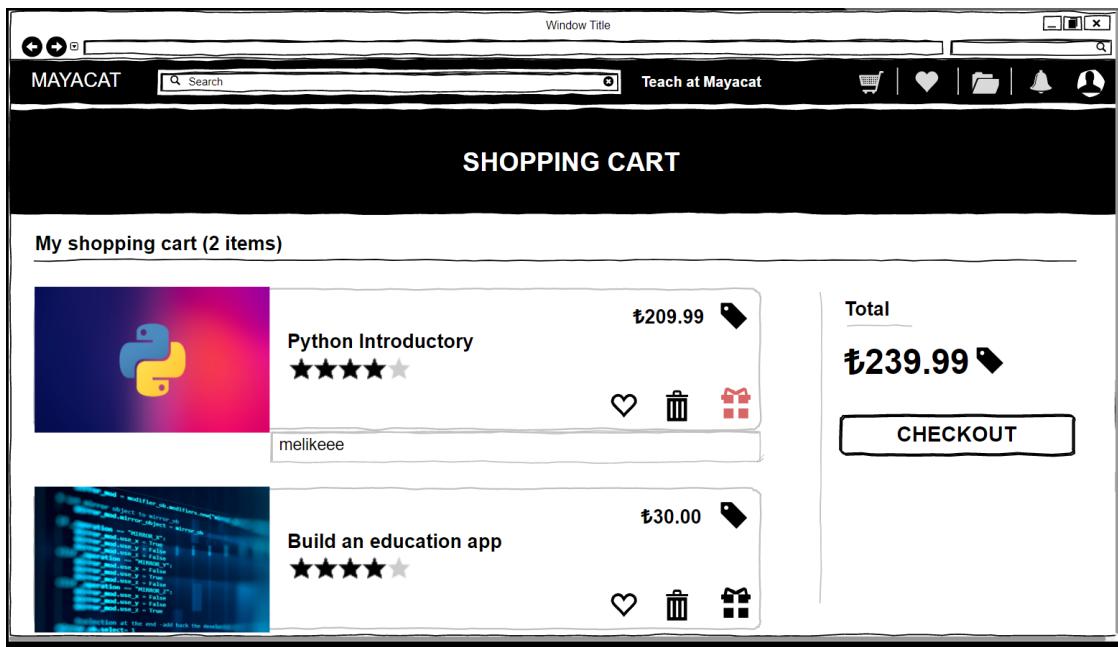
```
insert into Inside-cart values (9, 'mayaozsy', null);
```

```
select advertisementno, advertisement  
from Advertisement  
where cno = 1 and curdate() between startdate and finishdate and status = 1;
```

- If the student adds the course into wishlist by clicking on the heart icon;
insert into Wishes values ('mayazsy', 9);

Shopping Cart Page and with Gift Property Page





SQL Statement:

- When the student enters the shopping cart page,
select cno, cname, price, receiver-email
from Course as C, Inside-cart as I
where C.cno = I.cno and I.username = "mayazsy";

```
select cno, avg(score)
from Rate
where cno in ( select cno
    from In-cart as I
    where I.username = "mayazsy" )
group by cno;
```

- In the mockups above, the user selected the gift option for the first course in the cart, then provided the username of the receiver. Assume that the cno of the first course is 5 and the second course is 15.
update Inside-cart
set receiver-username = 'melikeee'
where cno = 9 and username = 'mayazsy';

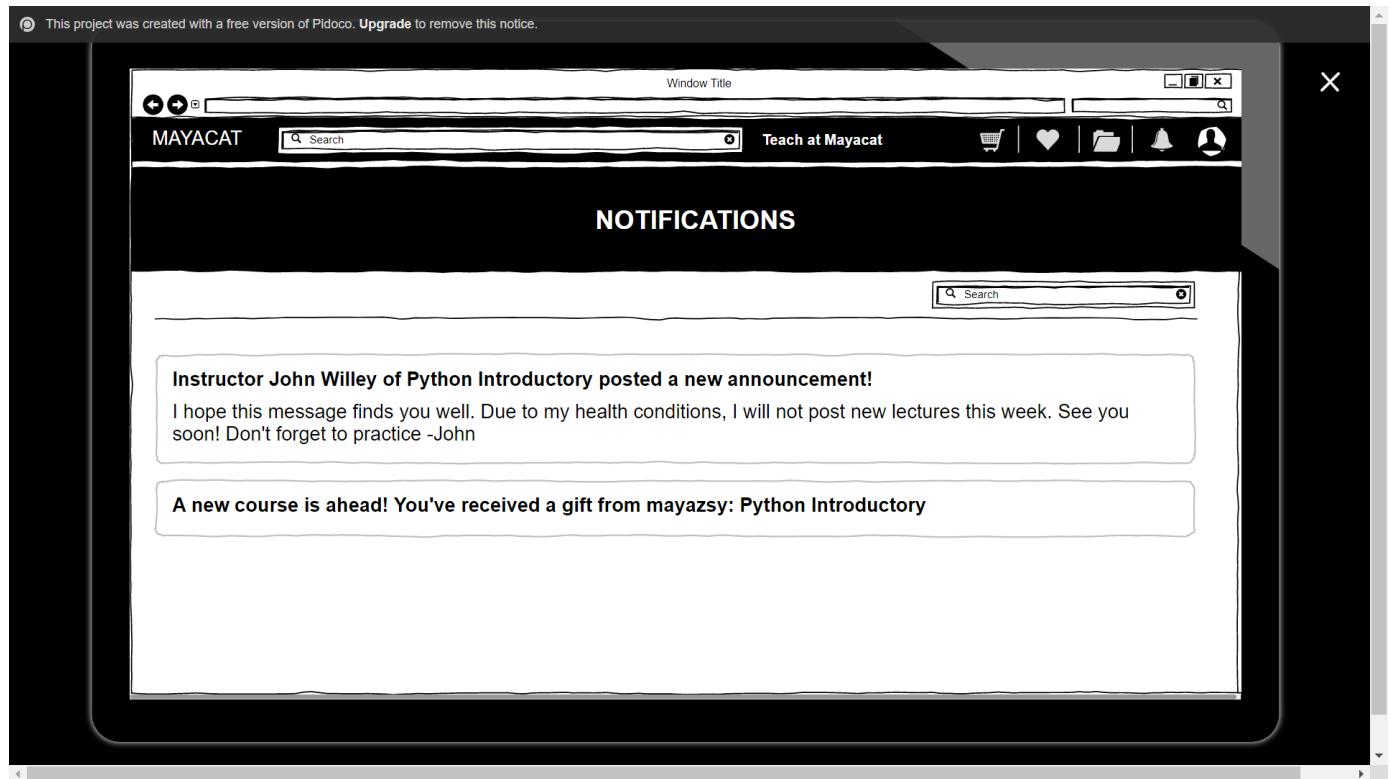
```
• Considering the second page, if the student buys the courses;  
--for the course that is a gift  
insert into Gift ('mayazsy', 'melikeee', 9);  
insert into Enroll ('melikeee', 9);
```

```
--for the course that is not a gift  
insert into Enroll values("mayazsy", 15 );
```

- If the student selects remove option for the second course;

```
delete from Inside-cart  
where cno = 15 and username = 'mayazsy';
```

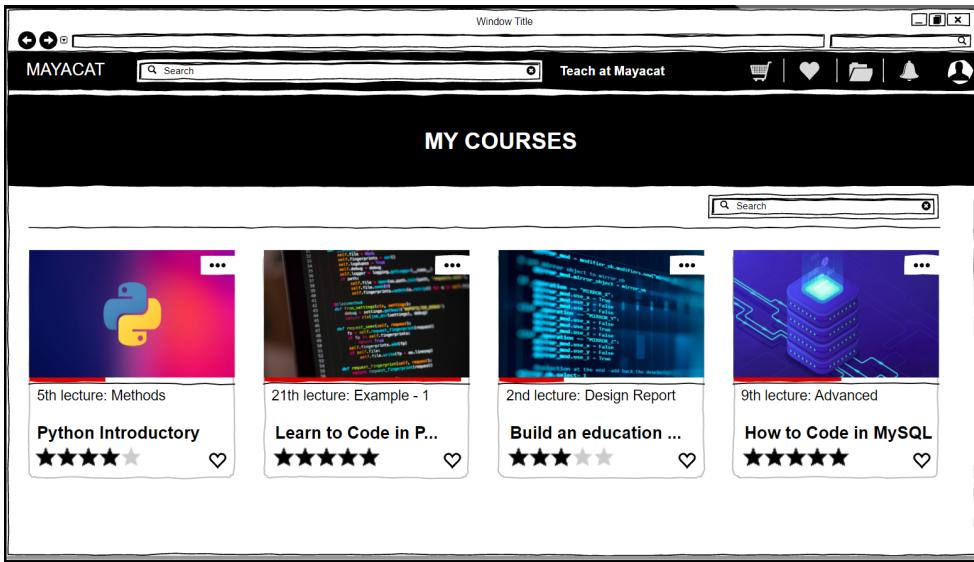
Notifications



SQL Statement:

```
-- for listing the announcements of enrolled courses  
select ann-tex, cname, ann-id, ann-date, owner-username  
from Course, Announce, Enroll  
where Enroll.s-username = "mayazsy" and Enroll.cno = Course.cno and Announce.cno =  
Course.cno;  
-- for the list of received gifts  
select g.sender-username, c cname  
from Gift as g, Course as c  
where g.cno = c.cno;
```

Student's Courses Page



SQL Statement:

```
select cno, cname, course-img  
from Course, Enroll  
where Course.cno = Enroll.cno and Enroll.s-username = "mayazsy";
```

```
select cno,avg( score )  
from Rate  
where cno in ( select cno  
                from Enroll as E  
                where E.s-username = "mayazsy" )  
group by cno;
```

- The red lines behind the course images demonstrate the general progress for the course. In order to find the progress rate, the number of the completed lectures will be divided into the total number of the lectures of the given course. Assume that the cno of Python Introductory course is 5.

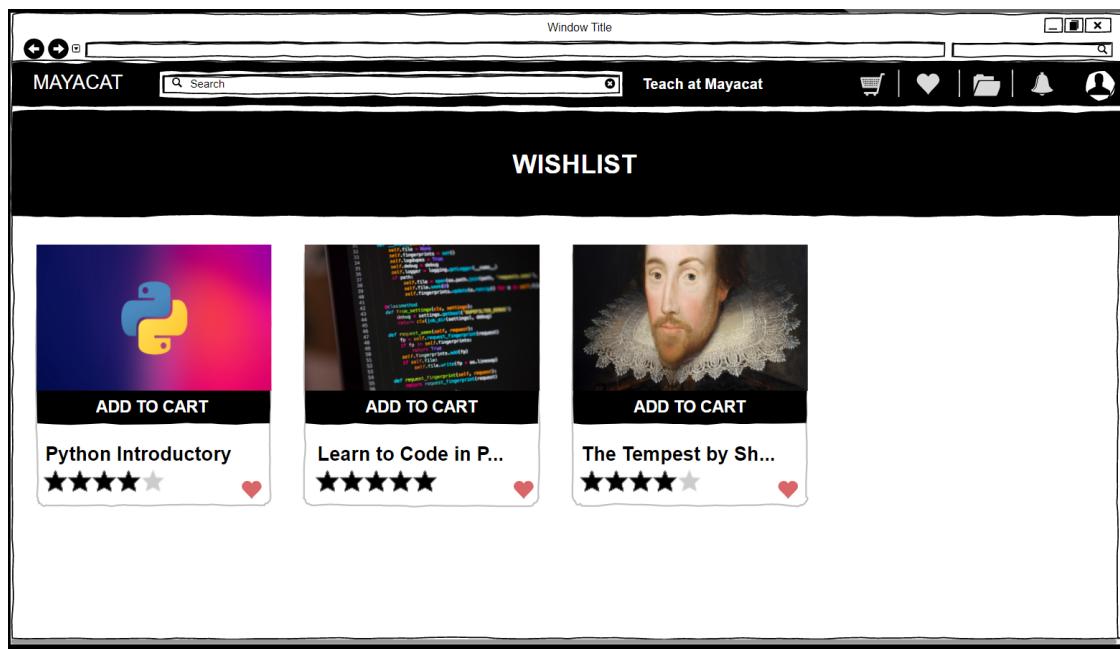
--The number of lectures of the Python Introductory course is found.

```
select count (lecture-no) as tot-lec-count  
from Lecture  
where cno = 5;
```

-- The number of completed lectures of the Python Introductory course is found.

```
select count (lecture-no) as comp-lec-count  
from Lecture as L, Progress as P  
where L.cno = 5 and L.cno = P.cno and username='mayazsy';
```

Wishlist Page



SQL Statement:

```
-- listing the courses inside the wishlist
select cno, cname, course-img
from Course, Wishes
where Wishes.cno = Course.cno and Wishes.s-username = "mayazsy";
```

- If the student adds the course called The Tempest by Shekspare to cart (Assume that the cno is 22);
insert into Inside-cart values (22, 'mayazsy', null);

Watching Lecture Page

The screenshot shows a web-based lecture platform. At the top, there's a navigation bar with 'MAYACAT', a search bar, and links for 'Teach at Mayacat', 'Request Refund', and 'Lectures'. Below the navigation is a video player window titled 'Build an education app' showing a video of a person writing SQL queries on a notepad. The video player has a progress bar at 3:30 / 6:33. To the right of the video is a sidebar titled 'Lectures' containing a list of checked boxes next to lecture titles. The sidebar includes three dots next to each entry.

Lecture	Status
Introduction	<input checked="" type="checkbox"/>
Relational Algebra	<input checked="" type="checkbox"/>
E/R Diagrams	<input checked="" type="checkbox"/>
Proposal Report	<input checked="" type="checkbox"/>
MySQL Beginner	<input checked="" type="checkbox"/>
MySQL Intermediate	<input checked="" type="checkbox"/>
MySQL Advanced	<input checked="" type="checkbox"/>
Design Report	<input checked="" type="checkbox"/>
File Organization	<input type="checkbox"/>

SQL Statement:

- It was assumed that the cno of Build an education app is 15 and the lecture no of the current lecture (Design Report) is 1432.

```
select lecture-no, lecture-name, video  
from Lecture as L  
where L.cno = 15;
```

- When the lecture is opened, it is added into the Progress table and considered as completed.
insert into Progress ("mayazsy", 1432);

Course Overview Page

The screenshot shows a course overview page. At the top, there's a navigation bar with 'MAYACAT', a search bar, and links for 'Teach at Mayacat', 'Request Refund', and 'Lectures'. Below the navigation is a video player window titled 'Query Optimization' showing a video of a person writing SQL queries on a notepad. The video player has a progress bar at 6:33 / 6:33. To the right of the video is a sidebar titled 'Lectures' containing a list of checked boxes next to lecture titles. The sidebar includes three dots next to each entry.

DESCRIPTION
Relational data model. Entity/Relationship model. Relational Algebra. Structured Query Language, SQL. Relational database design. Tree-structured and hash-based indexing. Query processing and optimization. Transaction management, concurrency control and recovery issues in database systems. Development of a relational database application as a term project.

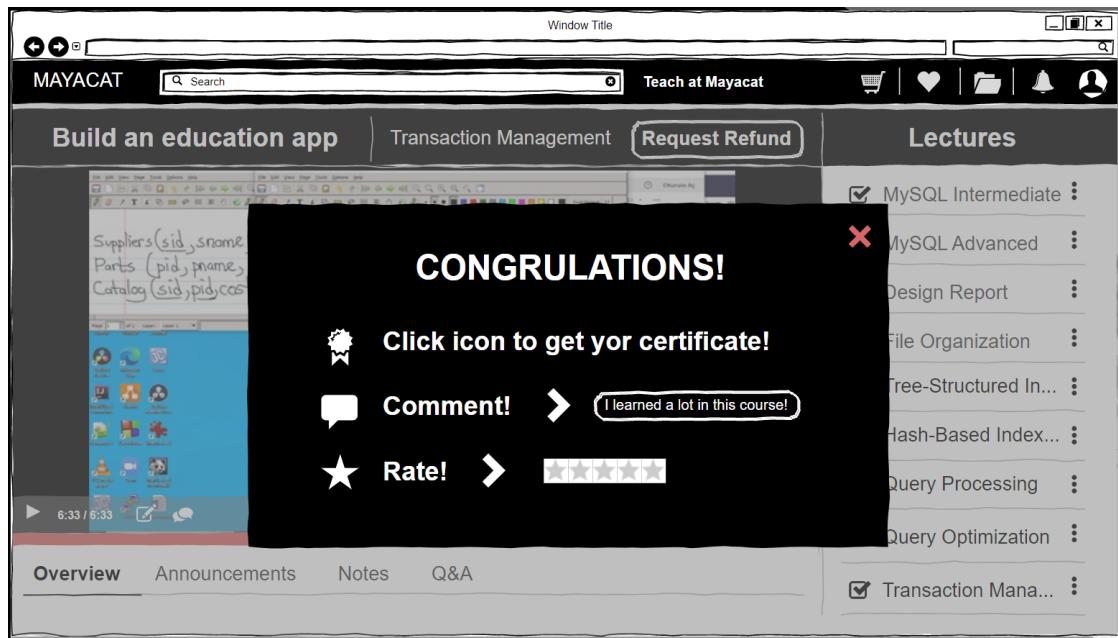
INCLUDING
• 37.5 hours on-demand video lectures
• 6 assignments

Lecture	Status
MySQL Intermediate	<input checked="" type="checkbox"/>
MySQL Advanced	<input checked="" type="checkbox"/>
Design Report	<input checked="" type="checkbox"/>
File Organization	<input checked="" type="checkbox"/>
Tree-Structured In...	<input checked="" type="checkbox"/>
Hash-Based Index...	<input checked="" type="checkbox"/>
Query Processing	<input checked="" type="checkbox"/>
Query Optimization	<input checked="" type="checkbox"/>
Transaction Mana...	<input type="checkbox"/>

SQL Statement:

```
-- list course contents  
select cno, description, including, owner-username  
from Course  
where cno = 15;
```

Completing a Course Page



SQL Statement:

- Assume that the cno of the Build an education app is 15
- ```
select count(lecture-no) as finished-lec-cnt
from Progress natural join Lecture
where s-username = "mayazsy" and cno = 15;
```

```
select count(lecture-no) as lecture-cnt
from Lecture
where cno = 15;
```

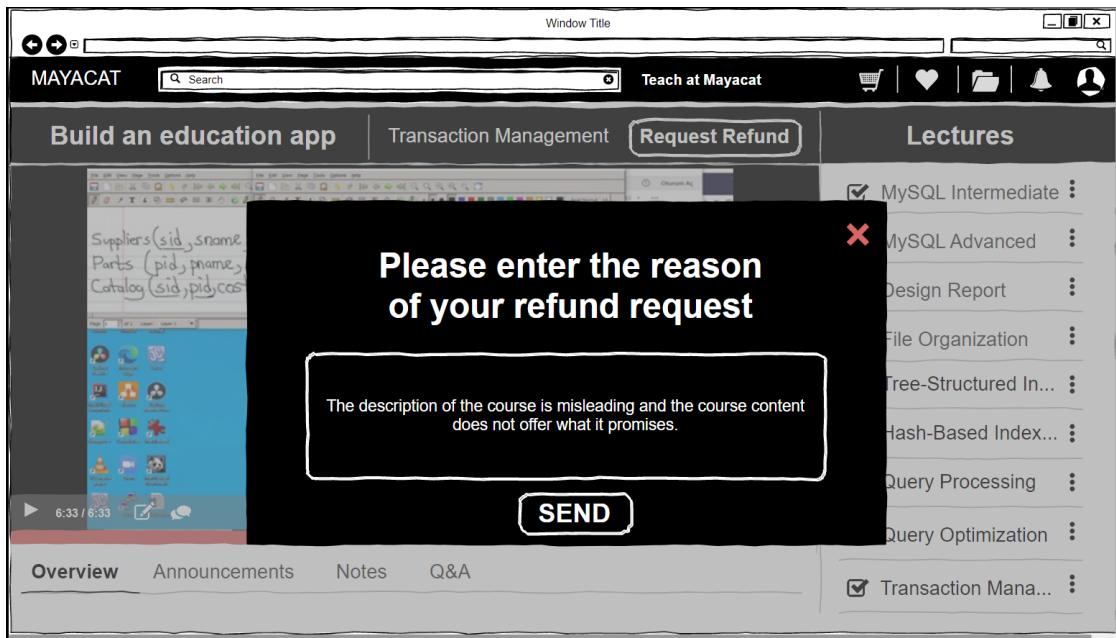
- If finished-lec-cnt == lecture-cnt;  
insert into Finishes values ( "mayazsy", lecture-no, "I learned a lot in this course!" );

```
select cno, cname, s-username, comment
from Finishes natural join Course
where s-username = "mayazsy" and cno = 15;
```



- A mockup of the certificate pdf template can be seen above.

## Course Refund Request



### SQL Statement:

- The student requested a refund for the course called Build and education app (cno: 15)

insert into RefundRequest values (512, "mayazsy", 15, 'The description of the course is misleading and the course content does not offer what it promises', 0);  
--it automatically enters to the database as 0, meaning not evaluated yet

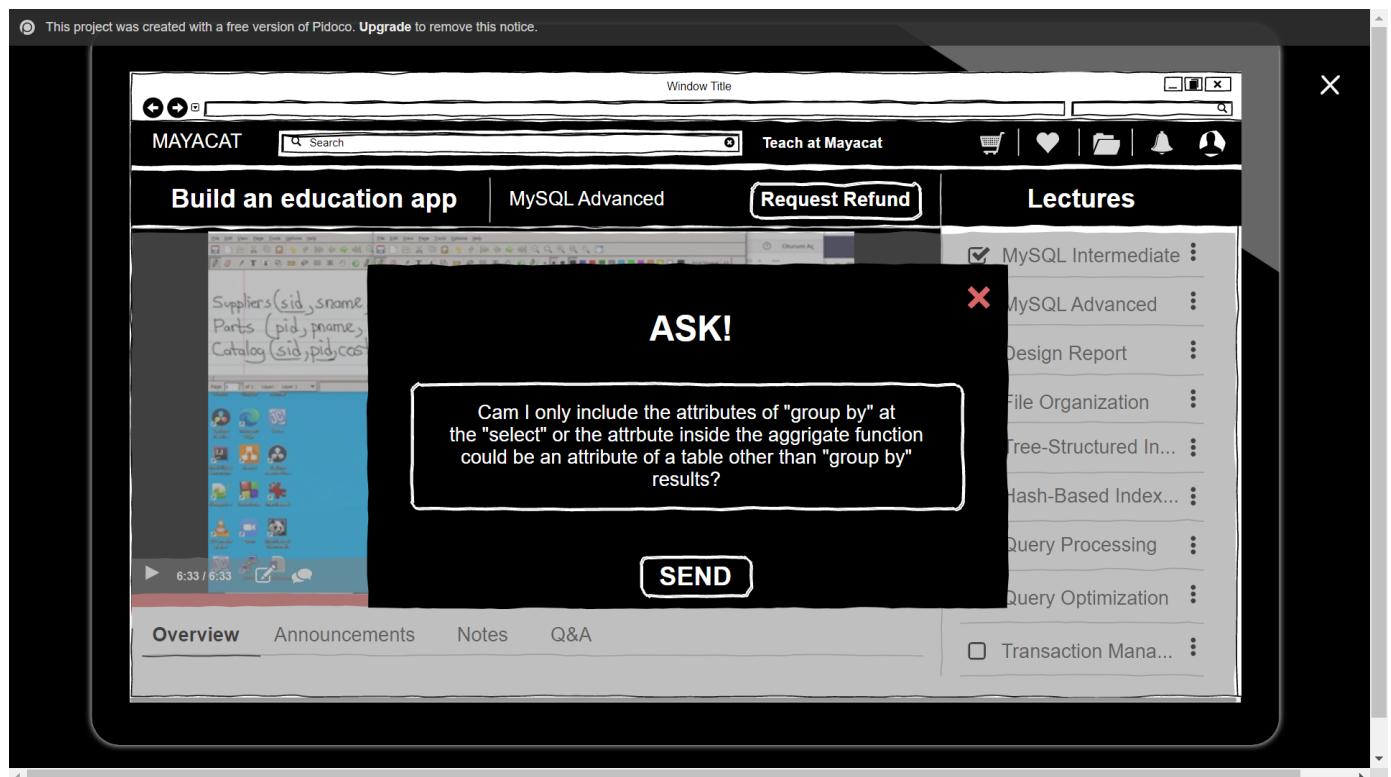
- If an admin whose username is marythead selects and evaluates the request as approved; insert into Evaluates values (512, 'marythead', '02/04/2020');

```
update RefundRequest
set status = 1
where refund-id = 512;
```

- If an admin whose username is marythead selects and evaluates the request as rejected;

```
select 'marythead', refund_id, curdate()
from RefundRequest
where status = -1;
```

## Course Ask Question



### SQL Statement:

```
-- 1028 is the id of the post, while 15 is the id of the course the post is inside of
-- inserts the comment seen above as a discussion post to the forum of the course
```

insert into Post values (1028, 15, 'Can I only include the attributes of "group by" at the "select" or the attribute inside the aggregate function could be an attribute of a table other than "group by" results?', "mayazsy");

## Note Page

The screenshot shows a web-based application window titled 'MAYACAT'. At the top, there's a navigation bar with icons for search, user profile, and notifications. Below the bar, a header includes 'Build an education app', 'Query Optimization', and a 'Request Refund' button. A sidebar on the right lists various lecture topics with checkboxes, some of which are checked. The main content area has tabs for 'Overview', 'Announcements', 'Notes' (which is selected), and 'Q&A'. A large text box labeled 'MY NOTES' contains two entries: 'Lecture MySQL Advanced: Do not forget to revise this lecture' and 'Lecture Query Processing: After 4:17, the topics are not included in th'. At the bottom of the notes box are 'APPLY' and 'CANCEL' buttons.

## SQL Statement:

- A student can take a note for the current lecture.

insert into Takes-note ("mayazsy", 14321, "Do not forget to revise this lecture");

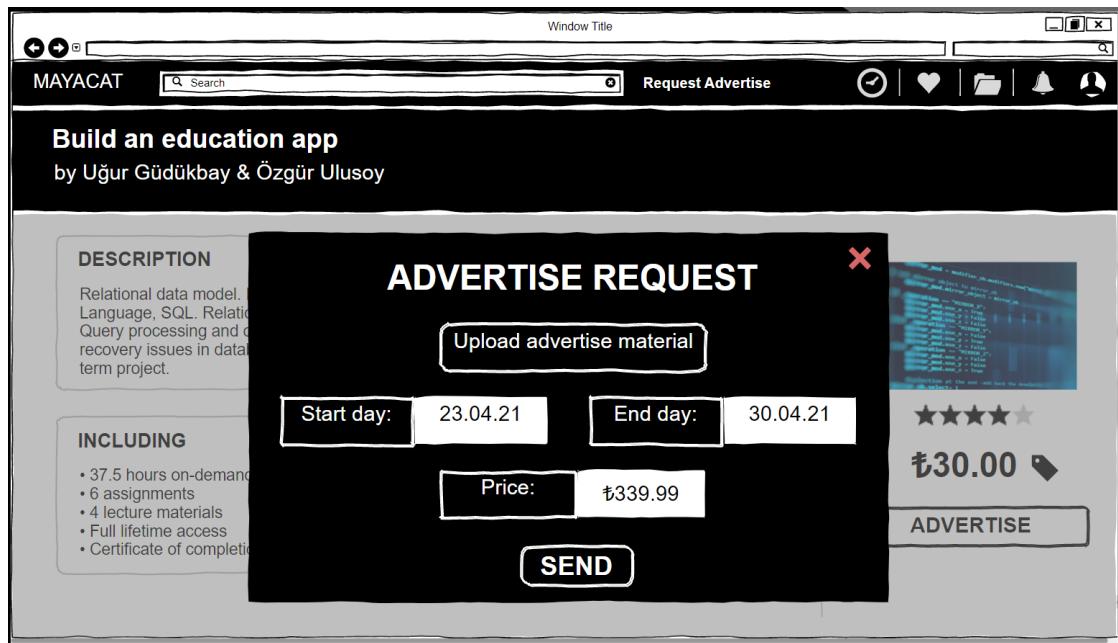
insert into Takes-note ("mayazsy", 10283, "After 4:17, the topics are not included in the recommended textbooks.");

## Course Page for Advertisers - Additional Functionality

The screenshot shows a web-based application window titled 'MAYACAT'. At the top, there's a navigation bar with icons for search, user profile, and notifications. Below the bar, a header includes 'Build an education app' and 'Request Advertise'. The main content area features a 'DESCRIPTION' box containing a detailed text about relational database concepts and a term project. To the right of the description is a thumbnail image of a computer screen displaying code. Below the description is an 'INCLUDING' box listing course components: 37.5 hours on-demand video lectures, 6 assignments, 4 lecture materials, Full lifetime access, and Certificate of completion. To the right of the including box is a price section showing '₺30.00' with a small discount icon. At the bottom right is a large 'ADVERTISE' button.

```
-- list course contents (assume that cno is 15)
select cno, description, including, owner-username
from Course
where cno = 15;
```

## Advertise Page for Advertisers - Additional Functionality

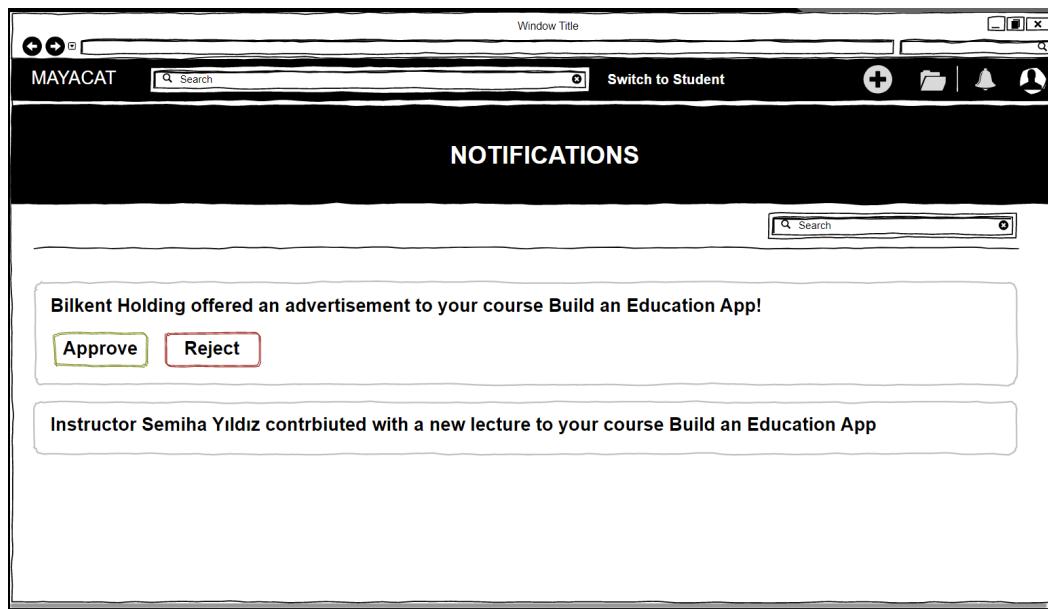


SQL Statement:

*--status = 0 means it is not reviewed yet*

```
insert into Advertisement (836, "isikozsoy", 1, "ad-836-img.php", 0, 339.99, '2021-04-21',
'2021-04-30');
```

## Notification Page for Instructors that Shows the Advertisement Offer - Additional Functionality



SQL Statement:

```
select advertisementno, advertisement, company-name, cno
from Advertisement natural joins Advertiser, Course
where Advertisement.cno = Course.cno and Course.owner-username = "mayazsy";
```

```
select cno, cname, i-username, name
from Contributor natural join Course, User
where Course.owner-username = "mayazsy" and User.username = name;
```

## Advertisement Requests for Advertisers - Additional Functionality

The screenshot shows a web application interface titled "ADVERTISE REQUESTS". At the top, there is a navigation bar with icons for search, request, and user profile. Below the title, there is a table with columns: Course, Start date, End date, Price, and Status. The table contains three rows of data:

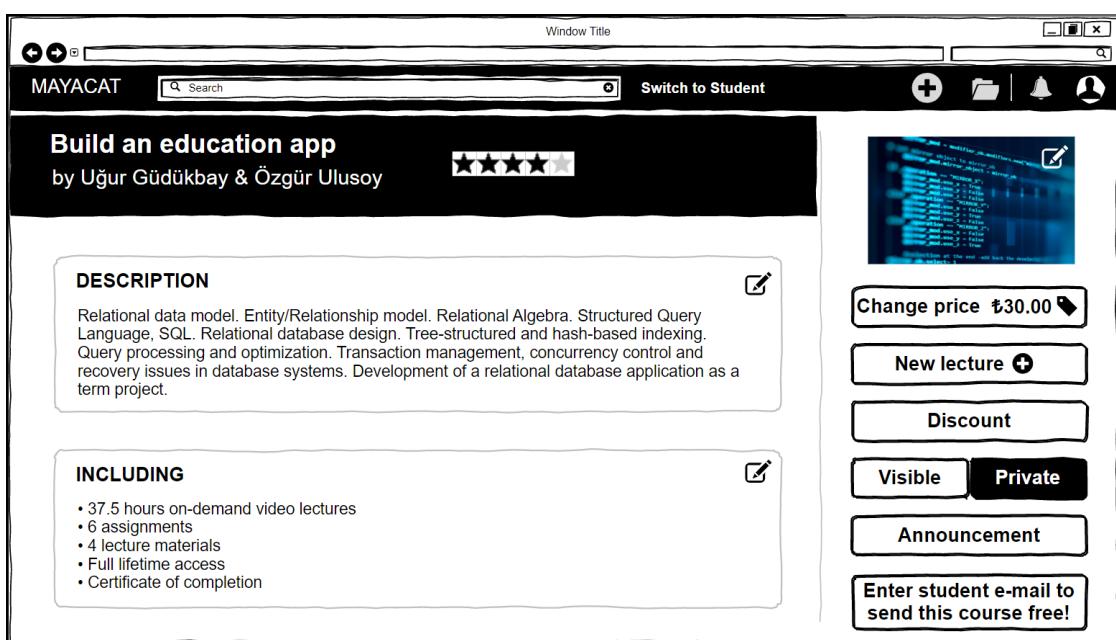
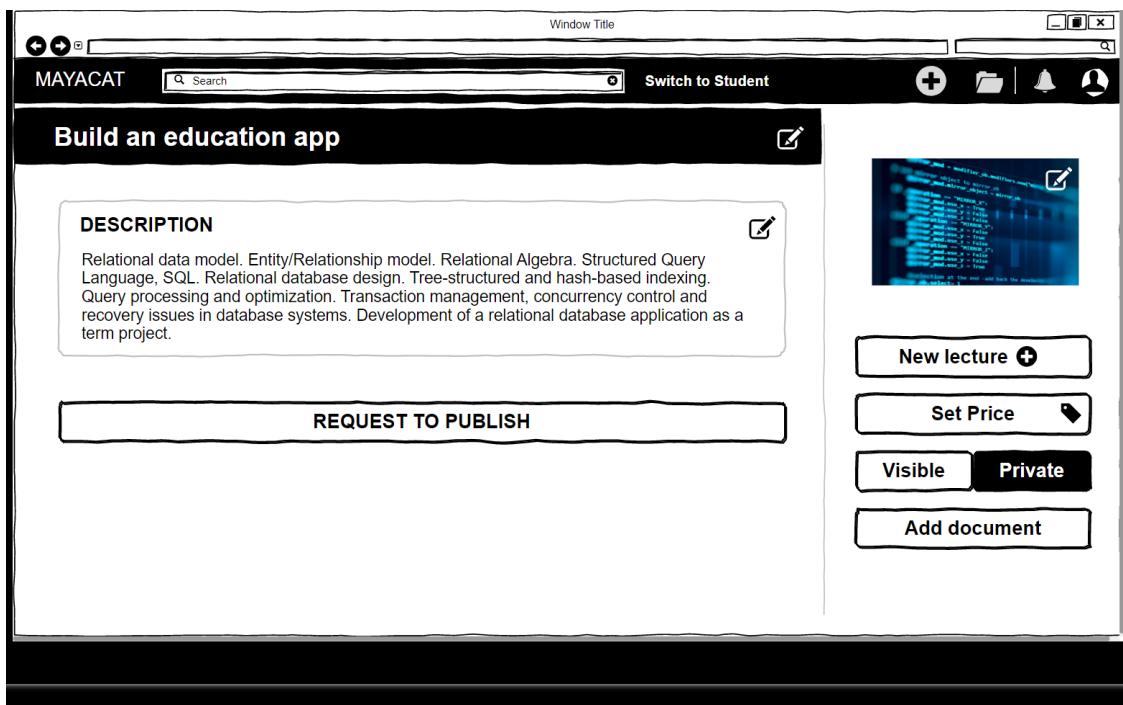
| Course                 | Start date | End date | Price   | Status   |
|------------------------|------------|----------|---------|----------|
| Python Introduction    | 23.04.21   | 25.04.21 | ₺250.00 | Pending  |
| Build an Education App | 23.04.21   | 30.04.21 | ₺399.99 | Accepted |
| Calculus for Beginners | 30.04.21   | 04.05.21 | ₺350.99 | Rejected |

SQL Statement:

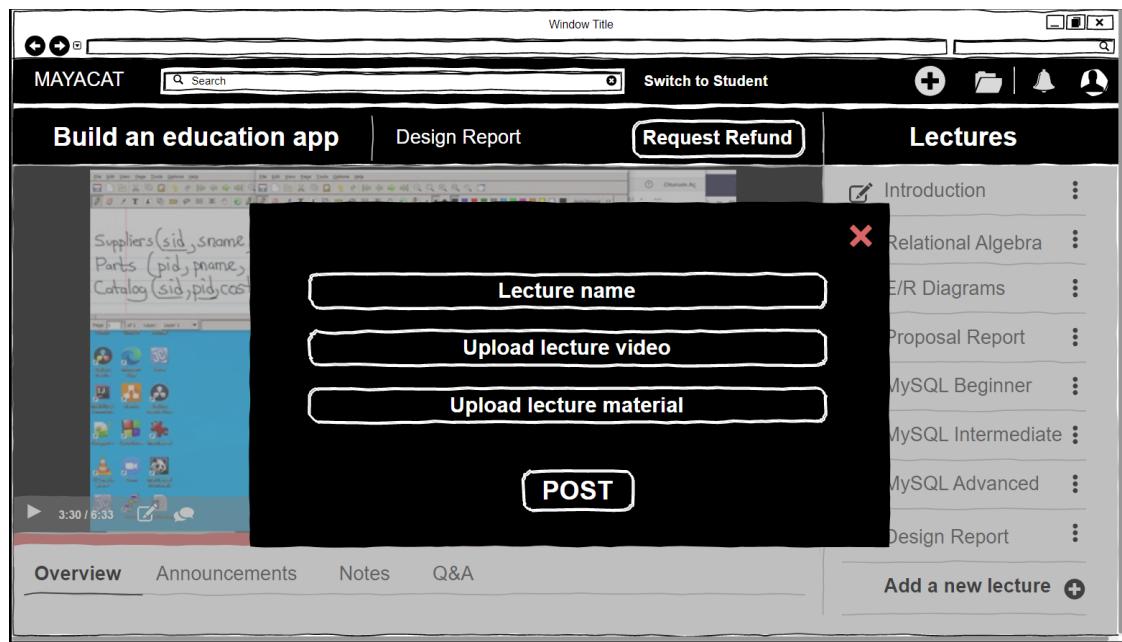
```
select advertisementname, startdate, finishdate, price, status
from Advertisement
where ad-username = 'isikozsoy'; --isikozsoy is an Advertiser
```

## Extra Pages Made for Instructor Which Are Unnecessary for Design Report

The screenshot shows a web application interface for an instructor. At the top, there is a navigation bar with icons for search, switch to student, and user profile. Below the navigation bar, there are tabs for "Build an education app", "Design Report", and "Request Refund". The "Build an education app" tab is active, showing a video player with a SQL query and some handwritten notes. The "Design Report" tab is also visible. To the right, there is a sidebar titled "Lectures" with a list of checked boxes for various MySQL topics. A context menu is open over the "E/R" item, showing options like "Change video", "Upload assignment", and "Upload resource". At the bottom, there are links for "Overview", "Announcements", "Notes", and "Q&A".



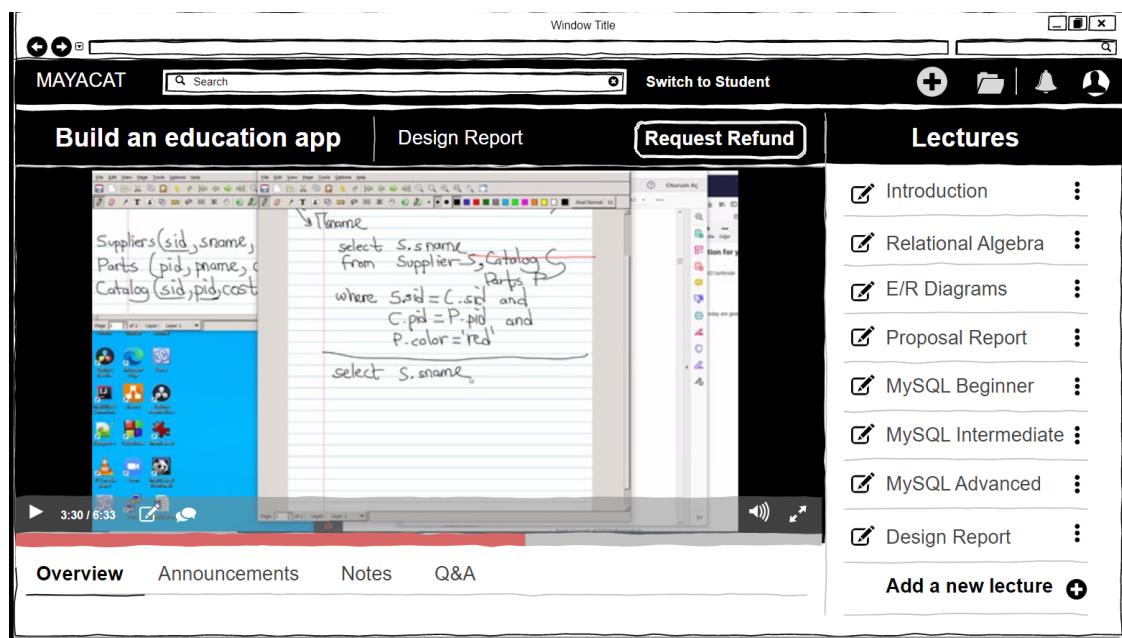
## Add Lecture Material Page



SQL Statement:

--1432 is the number of the lecture Indexing in the Build an education app and an additional material is added to that lecture.

insert into LectureMaterial values (100321, "100321.php", 1432);



## **IV. Systems and Technologies**

We have decided to use JavaScript, HTML, and PHP for website design and functionalities and MySQL for the database management system.