Professor Nathaniel Derbinsky

Database Management Systems Fall 2017 Section 2

Submission Date – 10 December 2017

Due Date – 11 December 2017

TEAM MEMBER 1

NAME: Bingqi Liu
MS Electrical and Computer Engineering Spring 2016
liu.b@husky.neu.edu

TEAM MEMBER 2

NAME: Mitresh Pandya
MS Computer Science Fall 2017
Pandya.mi@husky.neu.edu

TEAM MEMBER 3

NAME: Syed Aman Alam
MS Computer Science Spring 2017
Alam.sy@husky.neu.edu

TEAM MEMBER 4

NAME: Yue Cheng
MS Electrical and Computer Engineering Fall 2017
cheng.yue@husky.neu.edu

Abstract

From a bird eye view, the problem being solved is to store data in a permanent storage for a website which offers a service and thus the problem is to store all the data the revolves around the transactions done in providing such a service and the child services for that website. Thus, each and every action taken under the umbrella of services provided by the website should be tracked, stored, analyzed, distributed and ultimately used in an efficient manner to fulfill the goals of both the client and the website provider. This information should be available all the time to all the users associated and the website as well to keep such a service running efficiently.

The solution provided was a database - MySQL chosen by us but there were alternatives. The solution started with hearing what the website owner wants to track and how s/he wants to store (tables involved), what criteria is important (normalization) and how any particular set of data should be identified (indexing). The idea/ requirements were taken from the narrative provided by the website owners to us and we converted that idea into a real working database that can be started in under 2 minutes and can be kept running all the time (How to start and setup this database is shown in video as well).

The service involved was related to providing training/education and the user involved was by default considered a student. A student can obtain multiple roles and the service offers courses which are prepared by faculty members. These faculty members can also extend their education and take courses and also get involved in administrational area of this service. Thus, the roles of faculty/administration/student are interchangeable and by default everyone is a student. All the information of these roles being changed are tracked by the database and which user falls under which category, and when they make the transition. Since the main service provided is providing courses, every enrollment and completion of course, date and time of enrollment/completion, interests and certificate delivery are tracked by the database. The courses involved are broken into course materials just like chapters which have to be completed in a set sequence but since this is a modern website – a chapter can be quiz, post, video, link etc. All these types are also tracked and which course material is completed by which student and created by which faculty is also tracked and is part of the solution. Every course has a primary agenda and some secondary topics that it deals with – again tracked. It is assumed that when a student enrolls and studies course, s/he will be curious and will come up with questions which is highly encouraged and each and every question is tracked and forwarded to the creator of course. All such transactions are monitored for future purposes with the aim to improve the quality of course. If a faculty finds a question interesting, they can make the question public so that everyone can benefit from it. At the end of courses, feedback is highly encouraged so that the quality of content is revised and enhanced using the constructive feedback and the ratings provided.

A special sub problem that was tackled was how to make this website (trainly.io) special and how to differentiate it from others. The solution provided by us was to develop a special feature can playlists – every student can make their playlists of course materials – add course materials, remove course materials and customize it to his/her wish. The student can make multiple playlist on different areas say some course materials student finds difficult or some course materials s/he finds very interesting etc.

Ultimately the solution was provided in the form of a database and we worked as Lead Engineers for trainly.io and provided the services and developed solutions for the above problem.

A detailed textual description of the problem

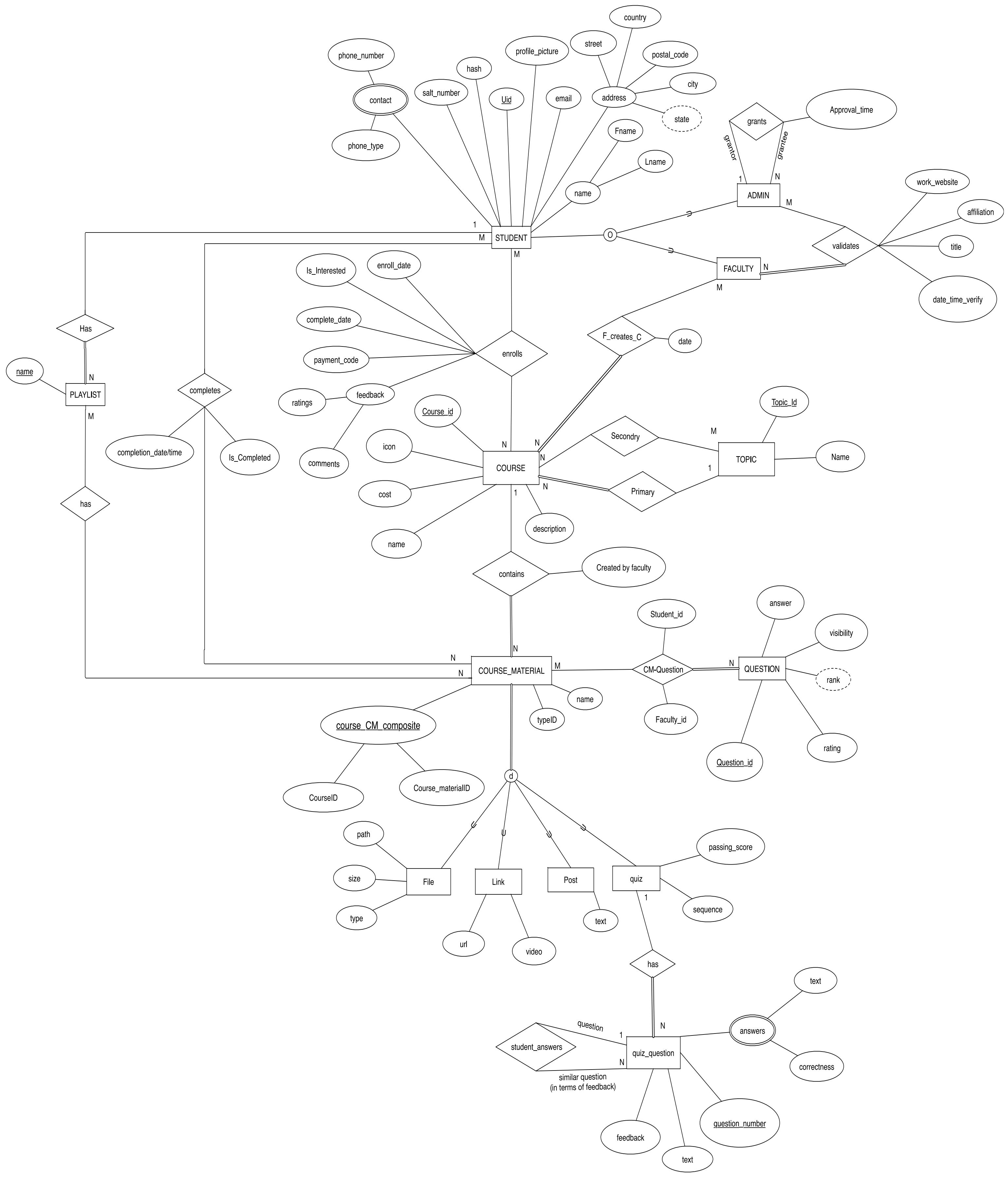
1. A Student has a unique Student ID, email, first name, last name, password, profile picture, contact details (phone number and type), and an address. The address contains street, city, postal code, state and country. Due to security issues, we should not directly save user's password in the database: standard security procedures(salting/hashing) to protect user's password should be observed.
2. A user is by default a student and may fall in either/both/none category for administrator and faculty.
3. Faculty are the creators of courses. When a faculty creates a course, we save the date when faculty create a course.
4. Faculty are validated by administrators. When a faculty is validated by an administrator- their work website, the time of verification, affiliation and title must be recorded.
5. For any enhanced positions, an admin takes: Administrators must be granted by another administrator who is grantor– for purposes of security auditing, the approval time must also be recorded.
6. Students can enroll in different courses. Each course has a unique Course id, name, description, icon, cost and a faculty creator. When a student enrolls, or completes a course, the enroll date/time and completion date/time should be recorded. This completion date/time serves as a marker to track the issuance of certificates. When student pays for a course, the payment code is also recorded. After student completes the course, student can give their feedback about the course which includes a rating on a scale of 1-5 and some comments for improvement.
7. Each course has a primary topic, as well as any number of secondary topics. Every topic has a unique Topic ID and name.
8. Every course contains its own materials, which is also created by faculty. Every material has a name, type ID, unique course ID and course material ID. This unique course material id is just to track of order of sequence. The student's completion status of each and every course material should be tracked- if one student completes all the course material, s/he also completes that course, and hence the completion status and date of each and every course material and course is tracked.
9. Course material can fall in one of four types: a downloadable file (Type 1), a link (Type 2), a post (Type 3), or a quiz (Type 4). We have given types to all course material types so that we can easily recognize simply from the number that what kind of content that course material is and we do not have to check with any other table. Downloadable files have a path, size, and type (used to provide MIME information). Links have a URL and are tagged as video or not. A post just has a large block of text. A quiz comprises a minimum

passing score, and a sequence of multiple-choice questions. The quiz has a sequence of questions.

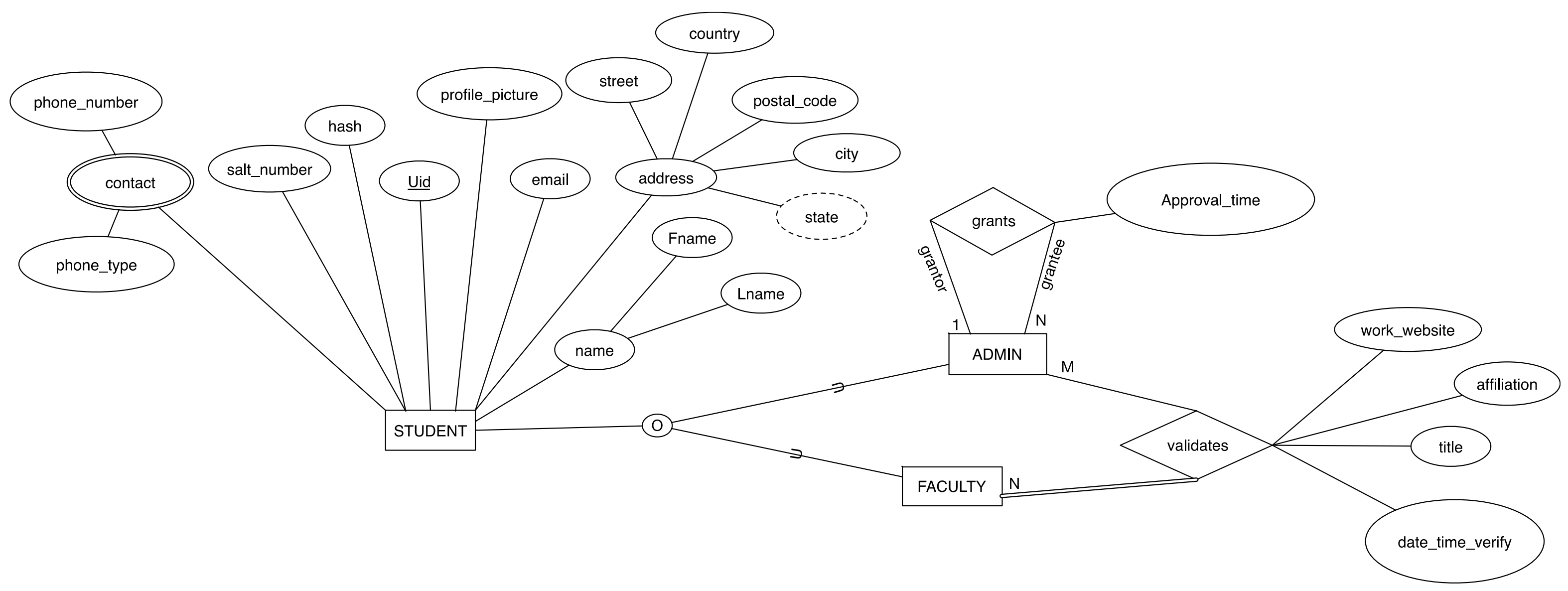
10. Each question in quiz has a unique number (not necessarily the number shown to user), a text, and a set of answers (each with a text, and indication of correct or not). All questions are taken from a single question bank and thus each question can be identified alone. After a student provides an answer, they will receive the feedback and it will also be indicated whether the answer is correct or not. Thus, every question will have a set of answers which have an indication of correctness and a text associated with the answer.
11. As students progresses in a course, they will have many questions when they read different course materials. Every question has a unique question ID. If any of the creators believes the question will be of use to other students, s/he makes the question visible and supplies an answer. If the question shows up, other students can rate questions and give high ratings to questions they find useful, we can calculate the total ratings to make a rank on different question.
12. Each playlist belongs to a student. Also, one student can have many playlists. Every playlist has a unique name which identifies it. Each playlist can have lots of course material which a student is interested in or maybe finds difficult or anything. The student can customize his/her playlist – add playlists/course materials, update, delete course materials, delete playlists etc.
13. Students will have many questions when they read different course materials, every question have a unique question ID. If any of the creators believes the question will be of use to other students, s/he makes the question visible and supplies an answer. If the question shows up, other students can rate questions they find useful with higher rating; we can calculate the rating to make a rank on different question.
14. The main tracking of course and its progress/completion is done at two points – when s/he enrolls and when s/he progresses/completes.

ER diagrams

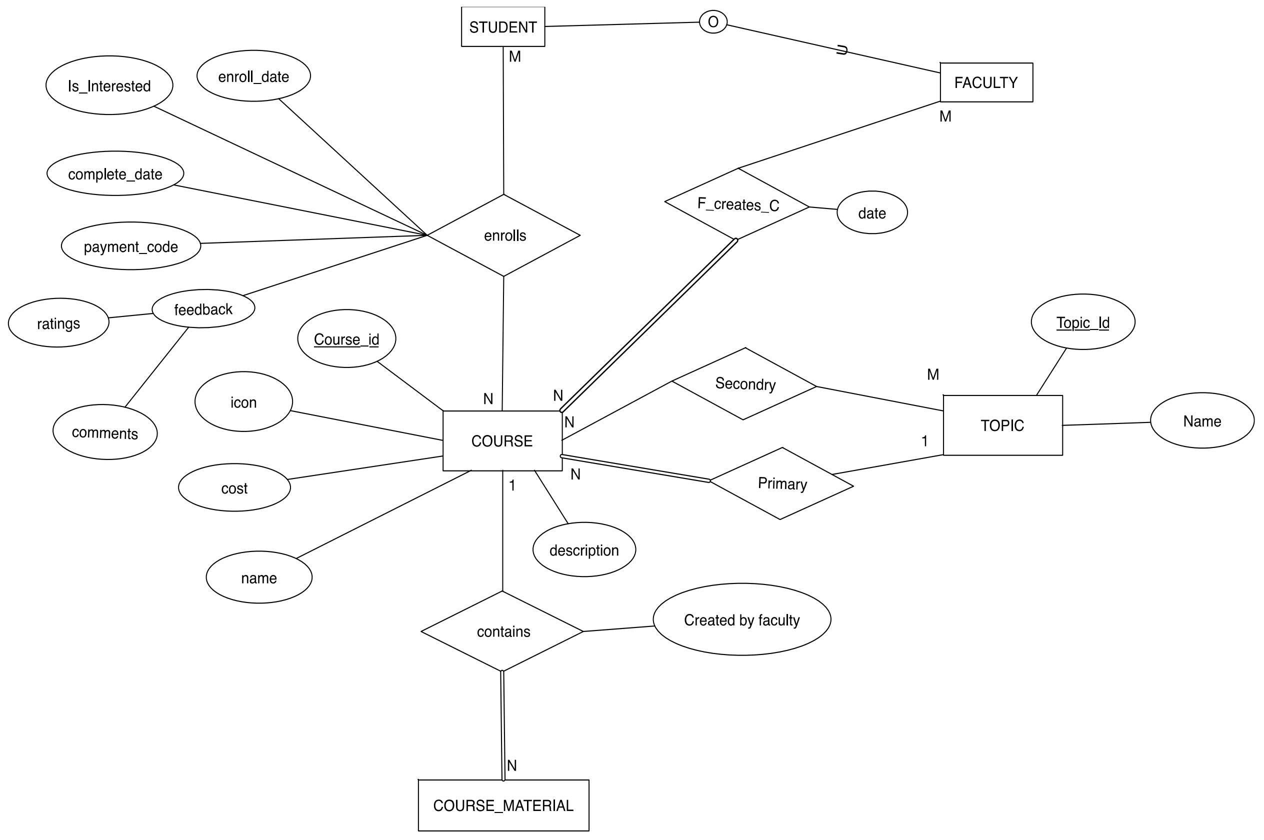
Below are the ER Diagrams added in the document. But Word has its limits and if an ER Diagram is not visible clearly, we have also uploaded a higher resolution version to a file sharing website - https://ibb.co/cqErvG . Please refer to it, if something is not clear enough.


phone_number
contact
phone_type
salt_number
hash
Uid
profile_picture
email
street
country
postal_code
city
address
state
Fname
Lname
name
STUDENT
grants
Approval_time
grantor
grantee
1
N
ADMIN
M
work_website
affiliation
validates
title
date_time_verify
O
FACULTY
N
M
1
M
M
Has
N
name
PLAYLIST
M
has
completes
completion_date/time
Is_Completed
Is_Interested
enroll_date
complete_date
payment_code
enrolls
ratings
feedback
comments
F_creates_C
date
Course_id
icon
cost
name
COURSE
N
N
N
N
1
description
Secondry
M
TOPIC
Topic_Id
Name
1
Primary
contains
Created by faculty
Student_id
answer
visibility
N
N
N
COURSE_MATERIAL
M
CM-Question
N
QUESTION
rank
name
typeID
Faculty_id
rating
Question_id
course_CM_composite
CourseID
Course_materialID
d
path
size
File
type
Link
url
video
Post
text
quiz
passing_score
sequence
1
has
text
N
question
1
student_answers
N
quiz_question
answers
correctness
similar question
(in terms of feedback)
question_number
feedback
text

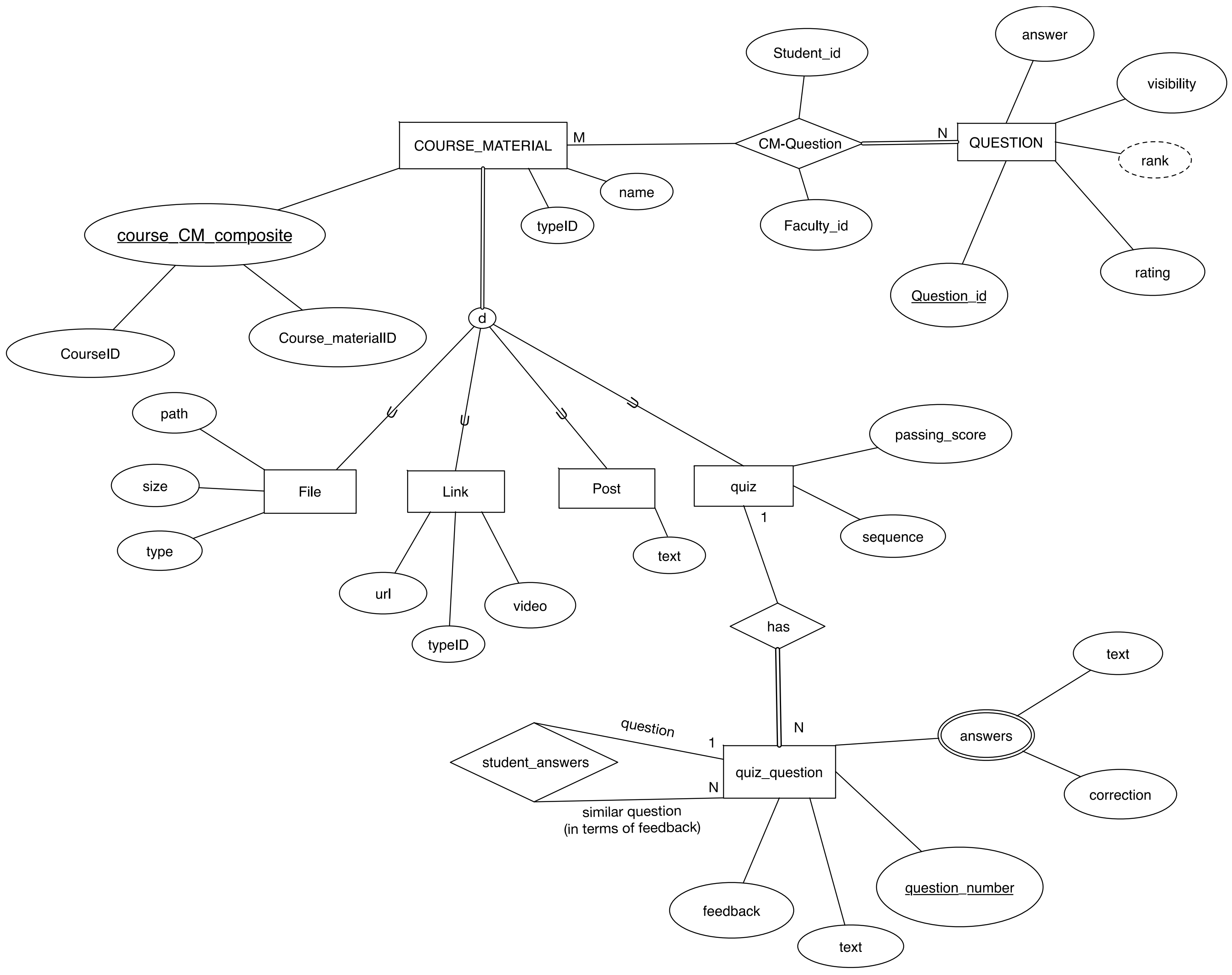
Student sub-view:



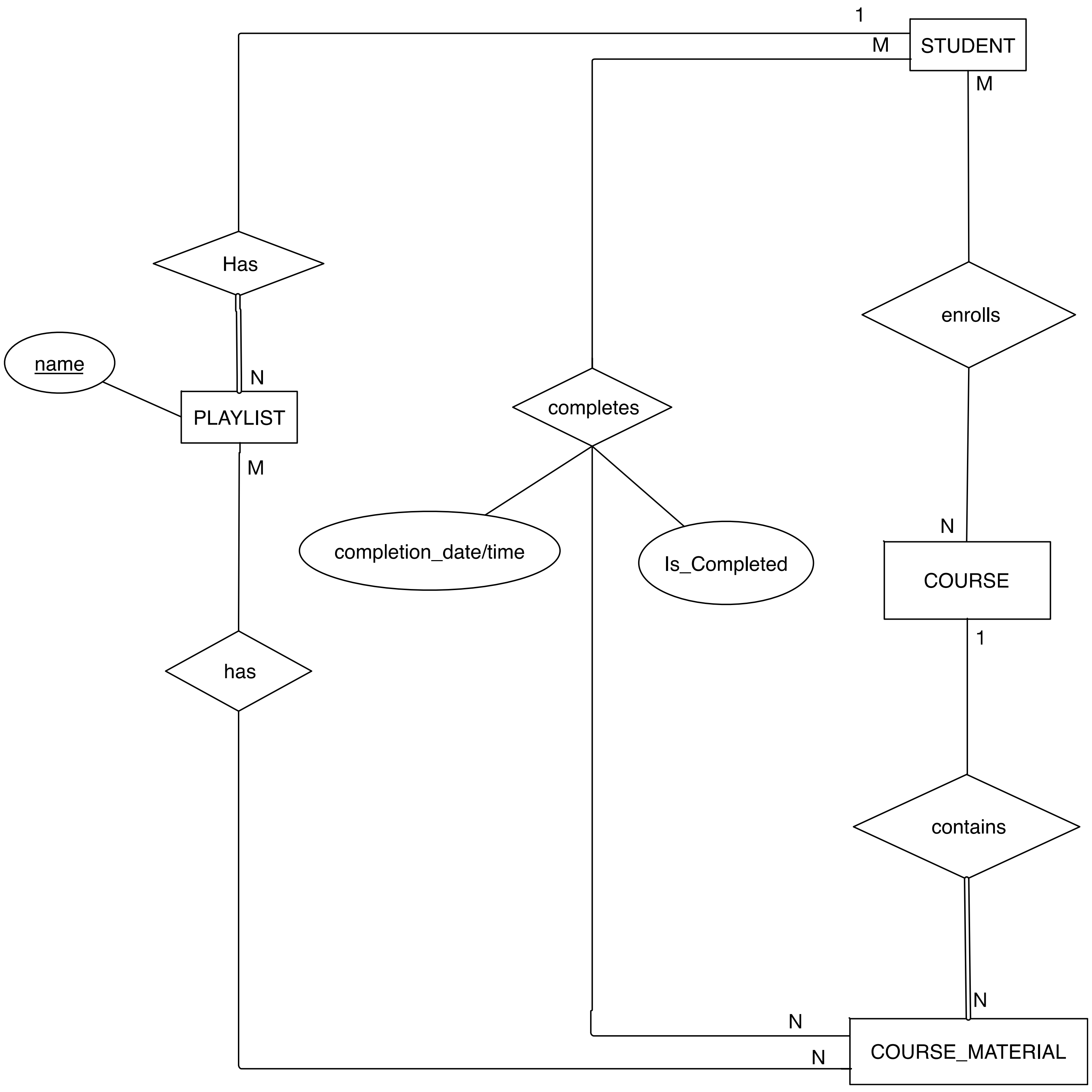
course sub-view:



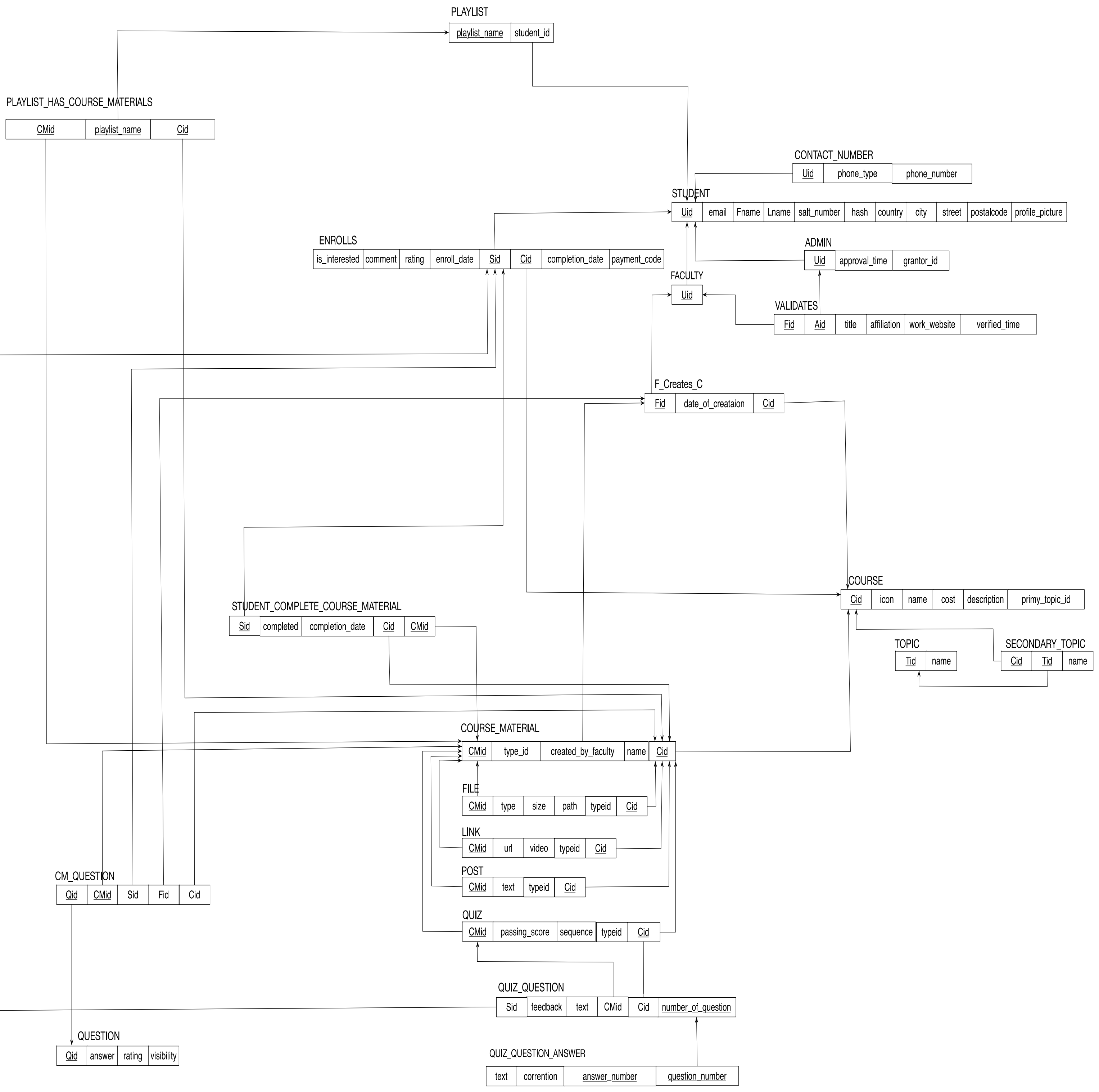
Course material sub-view:



Playlist Subview:



– Normalized relations: There is no deviation from 3NF and all tables are normalized. All primary keys are underlined and foreign keys shown with arrows. If this is not clear enough, we have also created a higher quality version of it (which does not fit in this document – refer to this image sharing link- https://ibb.co/kLeyqG Just download the image and zoom and it will be a lot easier to check).



Physical design

In general, if an ERD is converted to relational diagram properly, there won't be a need for denormalization (unless we have special constraints/needs). So, we did not had to do any denormalization. Below is each of the tables we made and all the attributes/columns involved. If in a table, two keys are listed as primary keys, then they both combine to be a single Primary Key and are not individually primary keys.

Table	Column	Datatype	Notes
Student	U_ID	INT (10)	Primary key (Obvious Key as it gives all other columns)
	First name	VARCHAR (255)	
	Last name	VARCHAR (255)	
	Email	VARCHAR (255)	
	Salt number		
	Hash	VARCHAR (40)	
	Street	VARCHAR (255)	
	City	VARCHAR (255)	
	Postal code	VARCHAR (10)	
	Country	VARCHAR (255)	
	Profile picture	LONGBLOB	
Faculty	U_ID	INT (10)	Primary key Foreign key to Student (Faculty is also a student)
Admin	U_ID	INT (10)	Primary key Foreign key to Student (Admin is also a student)
	Grantor-ID	INT (10)	
	Approval-time	TIMESTAMP	
Contact-number	U_ID	INT(10)	Primary key Foreign key to Student being multivalued attribute with student
	Phone-type	VARCHAR(10)	
	Phone-number	INT(10)	
Validates	Faculty ID	INT (10)	Primary key Foreign key to Faculty (M-N Relation between Admin and Faculty and hence the PK is the combination of faculty id and admin id)
	Admin ID	INT (10)	Primary key

			Foreign key to Admin (M-N Relation between Admin and Faculty and hence the PK is the combination of faculty id and admin id)
	Title	VARCHAR (100)	
	Affiliation	VARCHAR (100)	
	Work website	VARCHAR (100)	
	Data time verify	TIMESTAMP	
Course	Course ID	INT (10)	Primary key (Obvious choice and identifies the course)
	Icon	LONGBLOB	
	Name	VARCHAR (100)	
	Cost	FLOAT	
	Description	VARCHAR (255)	
	Primary topic ID	INT (10)	
Topic	Topic ID	INT (10)	Primary Key
	Name of topic	VARCHAR (255)	
Secondary topic	Course ID	INT (10)	Primary key Foreign key to course
	Topic ID	INT (10)	Primary key Foreign key to topic
	Name of topic	VARCHAR (255)	
Enrolls	Student ID	INT (10)	Primary key Foreign key to student (Student enrolls in course and M-N enrolls is the relation and hence the compound key comprising of StudentId and CourseId)
	Course ID	INT (10)	Primary key Foreign key to course
	Is Interested	TINYINT (1)	
	Enroll date	DATE	
	Completion date	DATE	
	Payment cod	INT (10)	
	Ratings	DECIMAL (2,1)	$1 \leqslant \text{rating} \leqslant 5$
	comment	VARCHAR (100)	
F_creates_c	Faculty ID	INT (10)	Primary key Foreign key to faculty
	Course ID	INT (10)	Primary key Foreign key to course (PK is the compound key comprising of

			FacultyId anc Course ID as it is the M-N relation between Faculty and Course)
	Date of creation	DATE	
Course material	Course ID	INT (10)	Primary key Foreign key to course
	Course material ID	INT (10)	Primary key
	Name	VARCHAR (100)	
	Created by faculty	INT (10)	Foreign key to F_creates_c
	Type ID	INT (4)	
File	File course ID	INT (10)	Primary key Foreign key to course material
	File course material ID	INT (10)	Primary key Foreign key to course material
	Type	VARCHAR (20)	
	Size	VARCHAR (20)	
	Path	VARCHAR (255)	
	Type ID	INT (1)	
Link	Link course ID	INT (10)	Primary key Foreign key to course material
	Link course material ID	INT (10)	Primary key Foreign key to course material
	url	VARCHAR (255)	
	Video	TINYINT (1)	
	Type ID	INT (2)	
Post	Post course ID	INT (10)	Primary key Foreign key to course material
	Post course material ID	INT (10)	Primary key Foreign key to course material
	text	VARCHAR (255e)	
	Type ID	INT (3)	
Quiz	Quiz course ID	INT (10)	Primary key Foreign key to course material
	Quiz course material ID	INT (10)	Primary key Foreign key to course material
	Passing score	INT (5)	
	Sequence	INT (5)	
	Type ID	INT (4)	
Question	Visibility	TINYINT (1)	DEFAULT 0
	Rating	DECIAL (2,1)	DEFAULT 1, 1≤rating≤5
	Answer to question	VARCHAR (255)	
	Question ID	INT (10)	Primary key

Student completes course material	Student ID	INT (10)	Primary key Foreign key to enrolls (Compound key comprising of Student Id, CourseId and Course Material ID as it is the M-N relation between Student and Course Material)
	Course ID	INT (10)	Primary key Foreign key to course material
	Course material ID	INT (10)	Primary key Foreign key to course material
	Completed	TINYINT (1)	
	Completion data time	DATE	
Quiz questions	Question number	INT (10)	Primary key
	Feed back	VARCHAR (255)	
	Text	VARCHAR (255)	
	Course ID	INT (10)	Foreign key to quiz
	Course material ID	INT (10)	Foreign key to quiz
	sid	INT (10)	Foreign key to enrolls
Quiz question answer	Text	VARCHAR (255)	
	Correct	TINYINT (1)	
	Question number	INT (10)	Primary key Foreign key to quiz question
	Answer number	VARCHAR (255)	Primary key
Course material question relation	Course ID	INT(10)	Foreign key to course material
	Material ID relation	INT(10)	Primary key Foreign key to course material
	Student ID	INT(10)	Foreign key to enrolls
	Faculty ID	INT(10)	Foreign key to F_creates_c
	Question ID relation	INT(10)	Primary key Foreign key to question
Playlist	Name playlist	VARCHAR (255)	Primary key
	Created by student	INT (10)	Foreign key to student
Playlist has course materials	Playlist name	VARCHAR (255)	Primary key Foreign key to playlist
	Course ID	INT (10)	Primary key Foreign key to course material
	Course material ID	INT (10)	Primary key Foreign key to course material

Above table includes all the tables we created by using DDL in our database trainly. Almost all primary keys type is INT, a fraction of key is VARCHAR.

The fact that they are of type INT here is almost purely arbitrary—or rather, almost arbitrary, because it is actually faster to search on numeric fields in many database engines; hence, numeric fields make good primary keys. That's the reason why we use every ID as primary key.

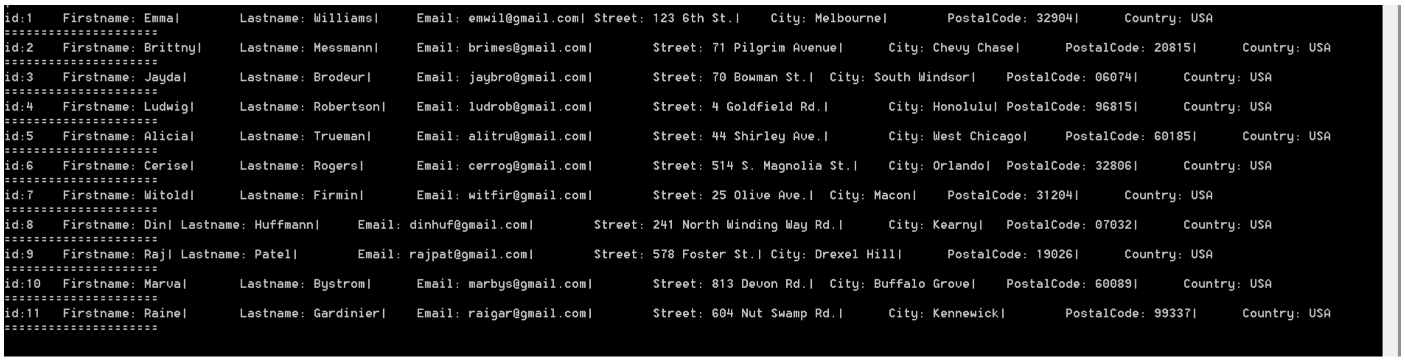
VARCHAR is a set of character data of indeterminate length. Its length can vary according with input. So, In reality, We should do some analysis of sample data to determine the length of text fields, to make sure we have enough space to save the date we input. If we set VARCHAR too short, it may end up with a database that cannot capture all the data we need to store. Thus, almost all the time, we have set VARCHAR (255), except in some case where we think that there can't be an input of that size. So, we then set VARCHAR to smaller size. For example, we set VARCHAR (255) in TEXT or DESCREPTION, because people may input lots of things in these areas. And, we set VARCHAR(20) in TYPE or SIZE, because there don't need an input of bigger size- just few word or numeric can express the meaning.

Between different table, we have lots of 1-to-1, 1-to-N and M-to-N relationships—the M-to-N relationships were resolved via separate tables. We created relationships between different table by using foreign keys. We also set the constraints on ratings and used bits to set/unset those fields such that ratings are between 1-5.

Screenshots of your running system

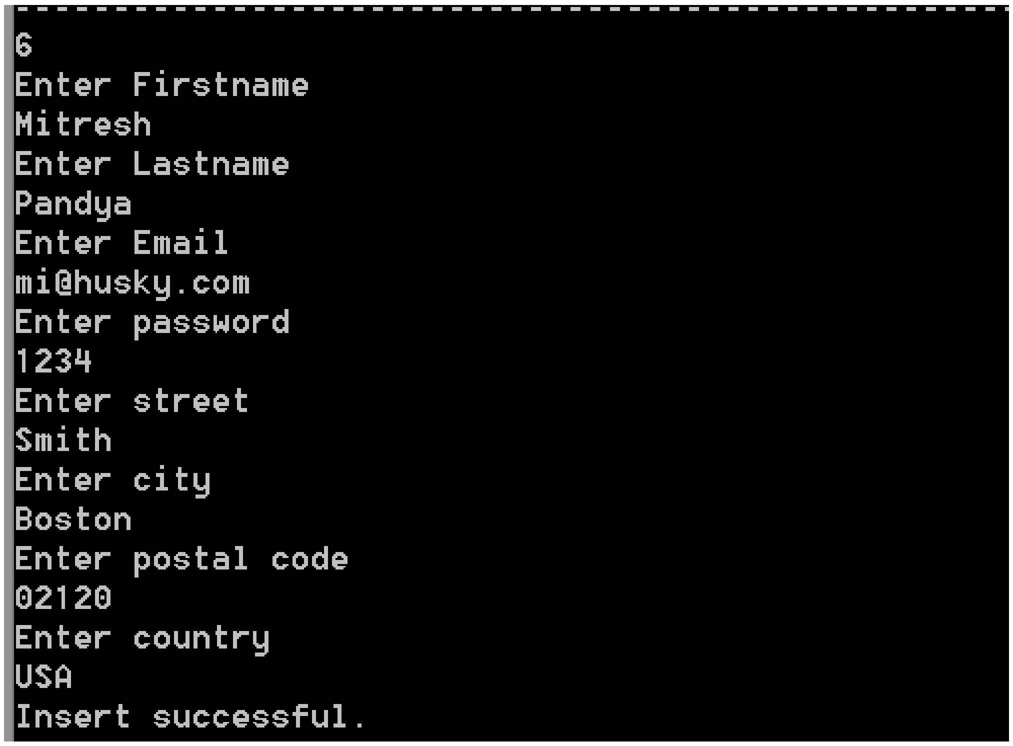
1) TASK 1: Register a New User

Initial State



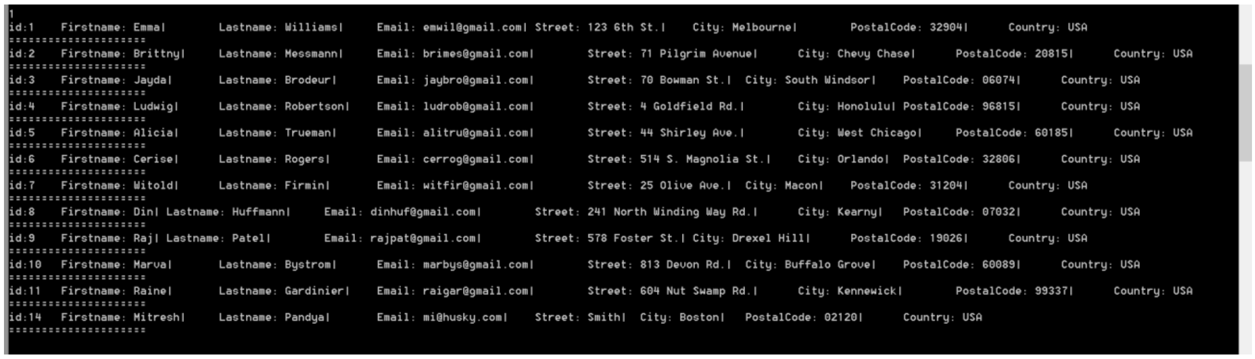
```
id:1    Firstname: Emma|        Lastname: Williams|     Email: emwil@gmail.com| Street: 123 6th St.|    City: Melbourne|        PostalCode: 32904|      Country: USA
=====================
id:2    Firstname: Brittny|     Lastname: Messmann|     Email: brimes@gmail.com|        Street: 71 Pilgrim Avenue|      City: Chevy Chase|      PostalCode: 20815|      Country: USA
=====================
id:3    Firstname: Jayda|       Lastname: Brodeur|      Email: jaybro@gmail.com|        Street: 70 Bowman St.|  City: South Windsor|    PostalCode: 06074|      Country: USA
=====================
id:4    Firstname: Ludwig|      Lastname: Robertson|    Email: ludrob@gmail.com|        Street: 4 Goldfield Rd.|        City: Honolulu| PostalCode: 96815|      Country: USA
=====================
id:5    Firstname: Alicia|      Lastname: Trueman|      Email: alitru@gmail.com|        Street: 44 Shirley Ave.|        City: West Chicago|     PostalCode: 60185|      Country: USA
=====================
id:6    Firstname: Cerise|      Lastname: Rogers|       Email: cerrog@gmail.com|        Street: 514 S. Magnolia St.|    City: Orlando|  PostalCode: 32806|      Country: USA
=====================
id:7    Firstname: Witold|      Lastname: Firmin|       Email: witfir@gmail.com|        Street: 25 Olive Ave.|  City: Macon|    PostalCode: 31204|      Country: USA
=====================
id:8    Firstname: Din| Lastname: Huffmann|     Email: dinhuf@gmail.com|        Street: 241 North Winding Way Rd.|      City: Kearny|   PostalCode: 07032|      Country: USA
=====================
id:9    Firstname: Raj| Lastname: Patel|        Email: rajpat@gmail.com|        Street: 578 Foster St.| City: Drexel Hill|      PostalCode: 19026|      Country: USA
=====================
id:10   Firstname: Marva|       Lastname: Bystrom|      Email: marbys@gmail.com|        Street: 813 Devon Rd.|  City: Buffalo Grove|    PostalCode: 60089|      Country: USA
=====================
id:11   Firstname: Raine|       Lastname: Gardinier|    Email: raigar@gmail.com|        Street: 604 Nut Swamp Rd.|      City: Kennewick|        PostalCode: 99337|      Country: USA
=====================
```

Then we ran task 1 and inputs given fed to command line interface-



```
------------------------------------------------
6
Enter Firstname
Mitresh
Enter Lastname
Pandya
Enter Email
mi@husky.com
Enter password
1234
Enter street
Smith
Enter city
Boston
Enter postal code
02120
Enter country
USA
Insert successful.
```

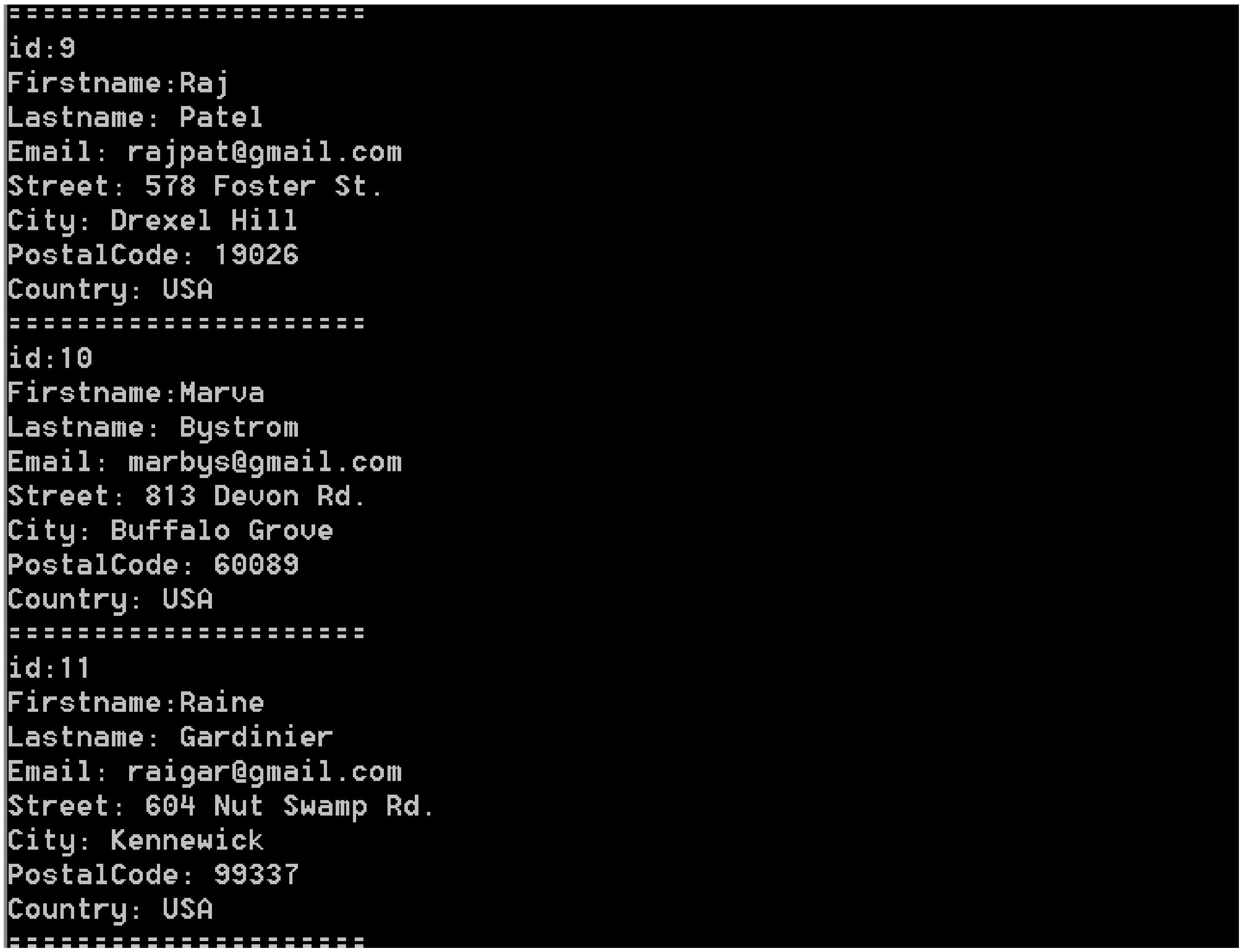
After running task 1, we have the state as:



```
1
id:1    Firstname: Emma|        Lastname: Williams|     Email: emwil@gmail.com| Street: 123 6th St.|    City: Melbourne|        PostalCode: 32904|      Country: USA
=====================
id:2    Firstname: Brittny|     Lastname: Messmann|     Email: brimes@gmail.com|        Street: 71 Pilgrim Avenue|      City: Chevy Chase|      PostalCode: 20815|      Country: USA
=====================
id:3    Firstname: Jayda|       Lastname: Brodeur|      Email: jaybro@gmail.com|        Street: 70 Bowman St.|  City: South Windsor|    PostalCode: 06074|      Country: USA
=====================
id:4    Firstname: Ludwig|      Lastname: Robertson|    Email: ludrob@gmail.com|        Street: 4 Goldfield Rd.|        City: Honolulu| PostalCode: 96815|      Country: USA
=====================
id:5    Firstname: Alicia|      Lastname: Trueman|      Email: alitru@gmail.com|        Street: 44 Shirley Ave.|        City: West Chicago|     PostalCode: 60185|      Country: USA
=====================
id:6    Firstname: Cerise|      Lastname: Rogers|       Email: cerrog@gmail.com|        Street: 514 S. Magnolia St.|    City: Orlando|  PostalCode: 32806|      Country: USA
=====================
id:7    Firstname: Witold|      Lastname: Firmin|       Email: witfir@gmail.com|        Street: 25 Olive Ave.|  City: Macon|    PostalCode: 31204|      Country: USA
=====================
id:8    Firstname: Din| Lastname: Huffmann|     Email: dinhuf@gmail.com|        Street: 241 North Winding Way Rd.|      City: Kearny|   PostalCode: 07032|      Country: USA
=====================
id:9    Firstname: Raj| Lastname: Patel|        Email: rajpat@gmail.com|        Street: 578 Foster St.| City: Drexel Hill|      PostalCode: 19026|      Country: USA
=====================
id:10   Firstname: Marva|       Lastname: Bystrom|      Email: marbys@gmail.com|        Street: 813 Devon Rd.|  City: Buffalo Grove|    PostalCode: 60089|      Country: USA
=====================
id:11   Firstname: Raine|       Lastname: Gardinier|    Email: raigar@gmail.com|        Street: 604 Nut Swamp Rd.|      City: Kennewick|        PostalCode: 99337|      Country: USA
=====================
id:14   Firstname: Mitresh|     Lastname: Pandya|       Email: mi@husky.com|    Street: Smith|  City: Boston|   PostalCode: 02120|      Country: USA
=====================
```

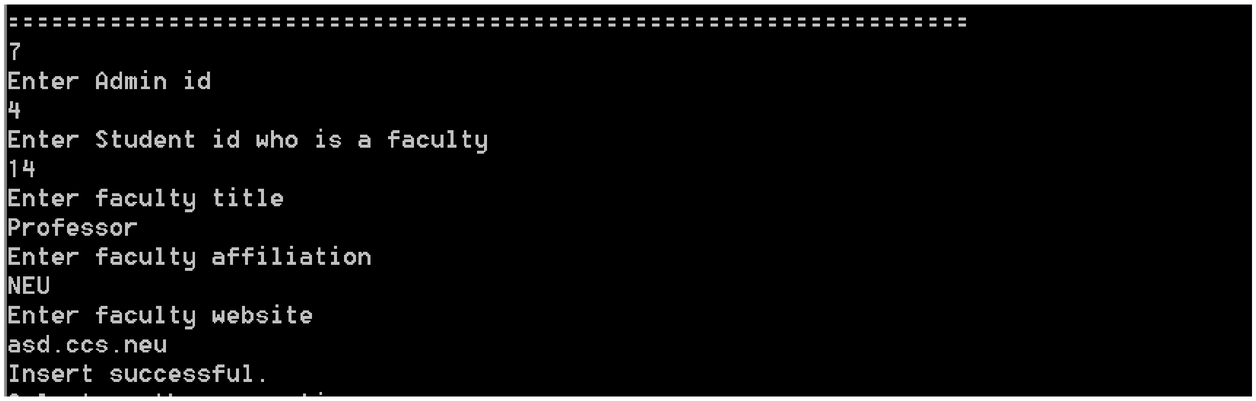
2) TASK 2: As an administrator, authenticate a faculty user (based upon title/affiliation/website/email) or fellow administrator

Initial State



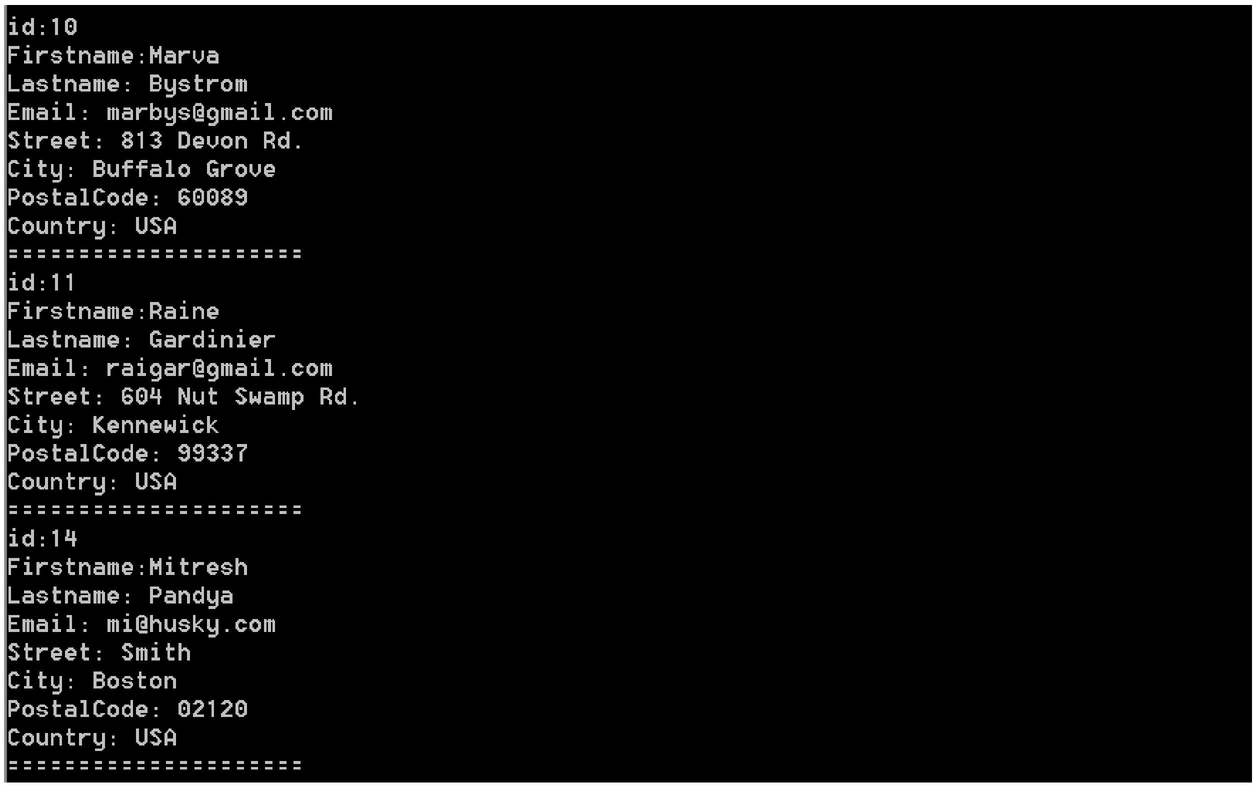
```
=====================
id:9
Firstname:Raj
Lastname: Patel
Email: rajpat@gmail.com
Street: 578 Foster St.
City: Drexel Hill
PostalCode: 19026
Country: USA
=====================
id:10
Firstname:Marva
Lastname: Bystrom
Email: marbys@gmail.com
Street: 813 Devon Rd.
City: Buffalo Grove
PostalCode: 60089
Country: USA
=====================
id:11
Firstname:Raine
Lastname: Gardinier
Email: raigar@gmail.com
Street: 604 Nut Swamp Rd.
City: Kennewick
PostalCode: 99337
Country: USA
=====================
```

Entered details for authentication-



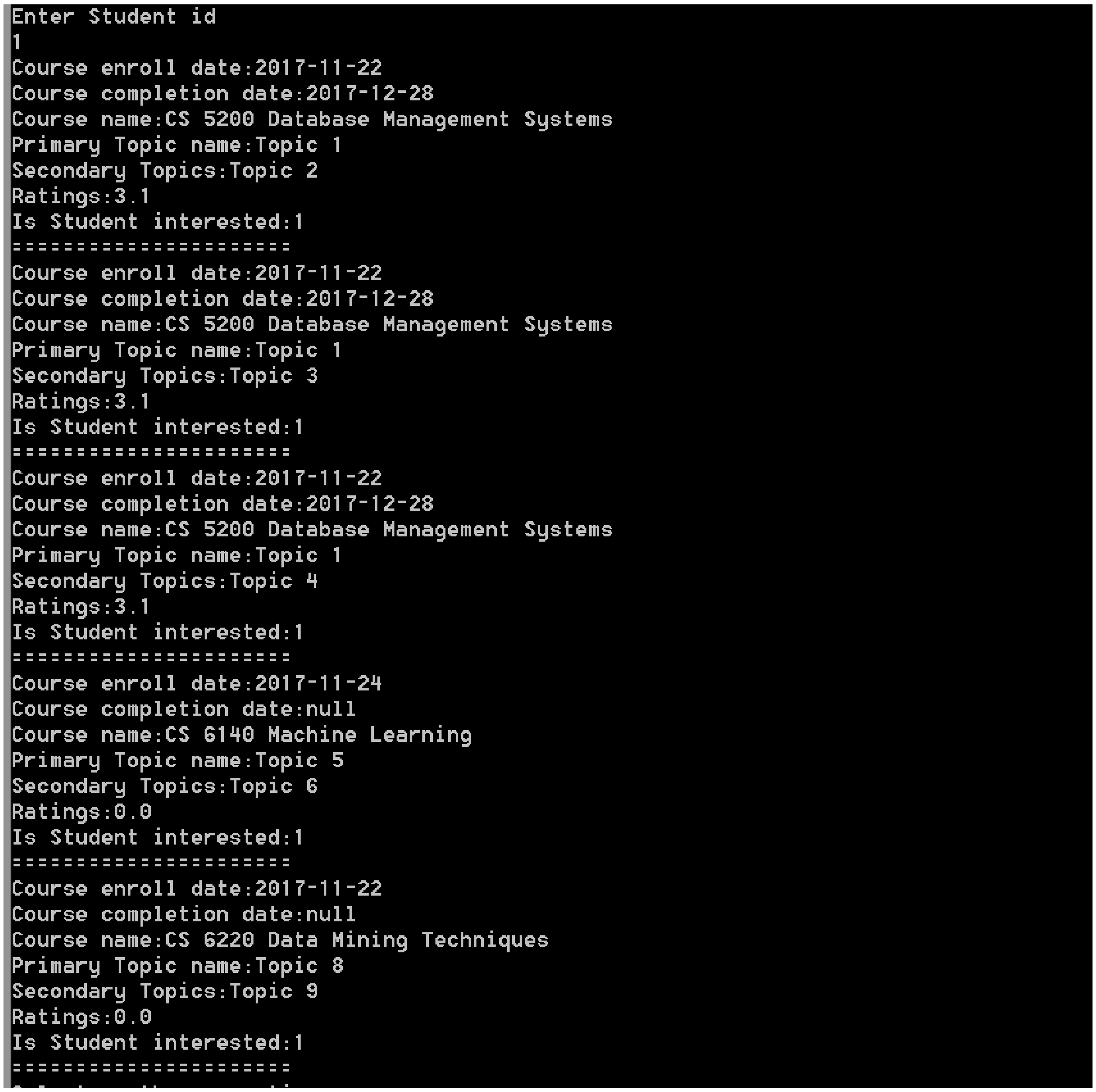
```
=================================================================
7
Enter Admin id
4
Enter Student id who is a faculty
14
Enter faculty title
Professor
Enter faculty affiliation
NEU
Enter faculty website
asd.ccs.neu
Insert successful.
```

After Insertion-



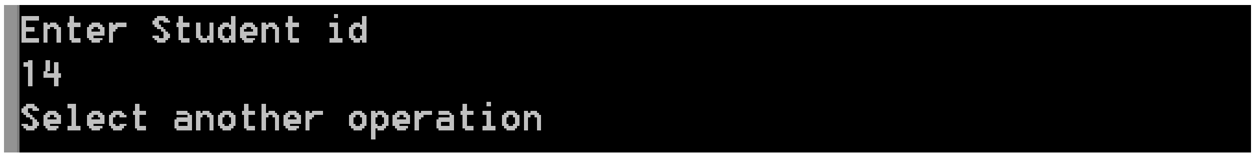
```
id:10
Firstname:Marva
Lastname: Bystrom
Email: marbys@gmail.com
Street: 813 Devon Rd.
City: Buffalo Grove
PostalCode: 60089
Country: USA
=====================
id:11
Firstname:Raine
Lastname: Gardinier
Email: raigar@gmail.com
Street: 604 Nut Swamp Rd.
City: Kennewick
PostalCode: 99337
Country: USA
=====================
id:14
Firstname:Mitresh
Lastname: Pandya
Email: mi@husky.com
Street: Smith
City: Boston
PostalCode: 02120
Country: USA
=====================
```

3) Task 3: Provide a categorized list of a student’s courses (each with primary/secondary topics, ranked by average evaluation score): currently enrolled, completed, of interest.

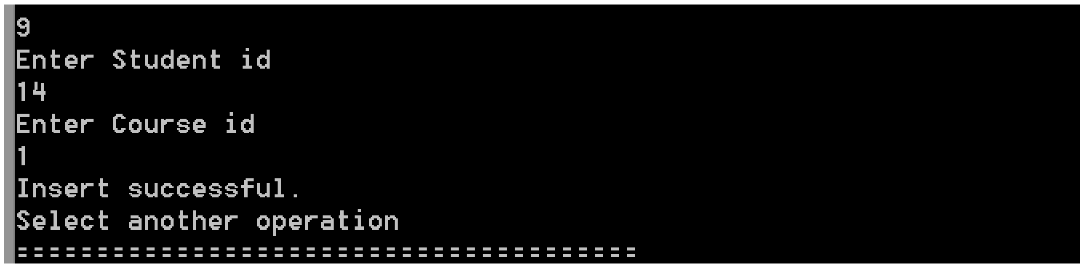


```
Enter Student id
1
Course enroll date:2017-11-22
Course completion date:2017-12-28
Course name:CS 5200 Database Management Systems
Primary Topic name:Topic 1
Secondary Topics:Topic 2
Ratings:3.1
Is Student interested:1
======================
Course enroll date:2017-11-22
Course completion date:2017-12-28
Course name:CS 5200 Database Management Systems
Primary Topic name:Topic 1
Secondary Topics:Topic 3
Ratings:3.1
Is Student interested:1
======================
Course enroll date:2017-11-22
Course completion date:2017-12-28
Course name:CS 5200 Database Management Systems
Primary Topic name:Topic 1
Secondary Topics:Topic 4
Ratings:3.1
Is Student interested:1
======================
Course enroll date:2017-11-24
Course completion date:null
Course name:CS 6140 Machine Learning
Primary Topic name:Topic 5
Secondary Topics:Topic 6
Ratings:0.0
Is Student interested:1
======================
Course enroll date:2017-11-22
Course completion date:null
Course name:CS 6220 Data Mining Techniques
Primary Topic name:Topic 8
Secondary Topics:Topic 9
Ratings:0.0
Is Student interested:1
======================
```

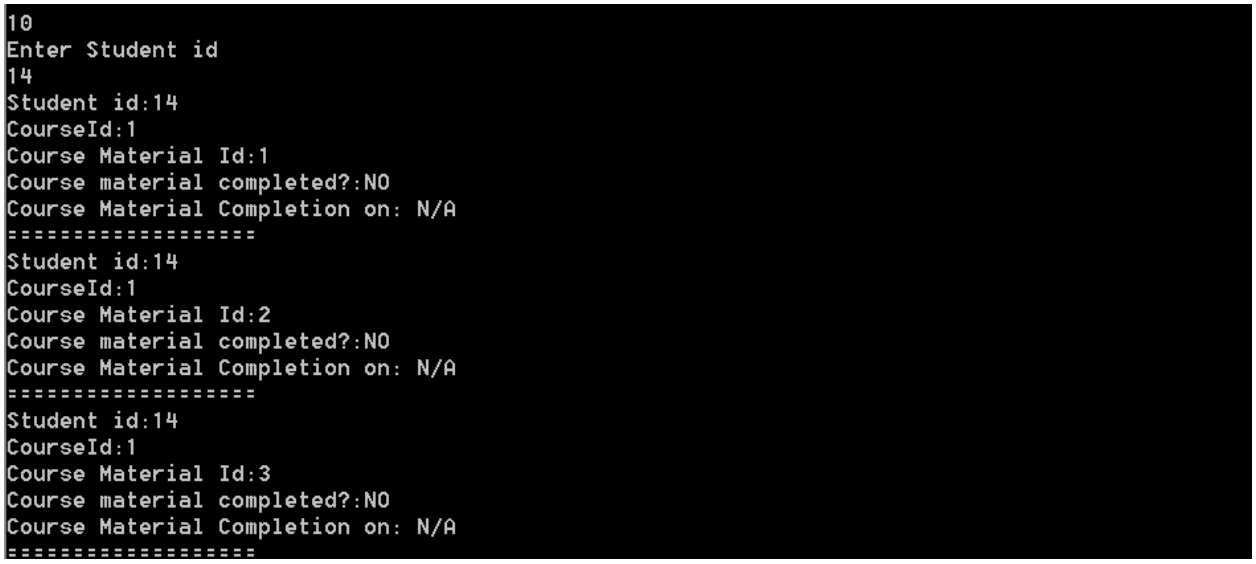
4) Task 4: Enroll a student in a course



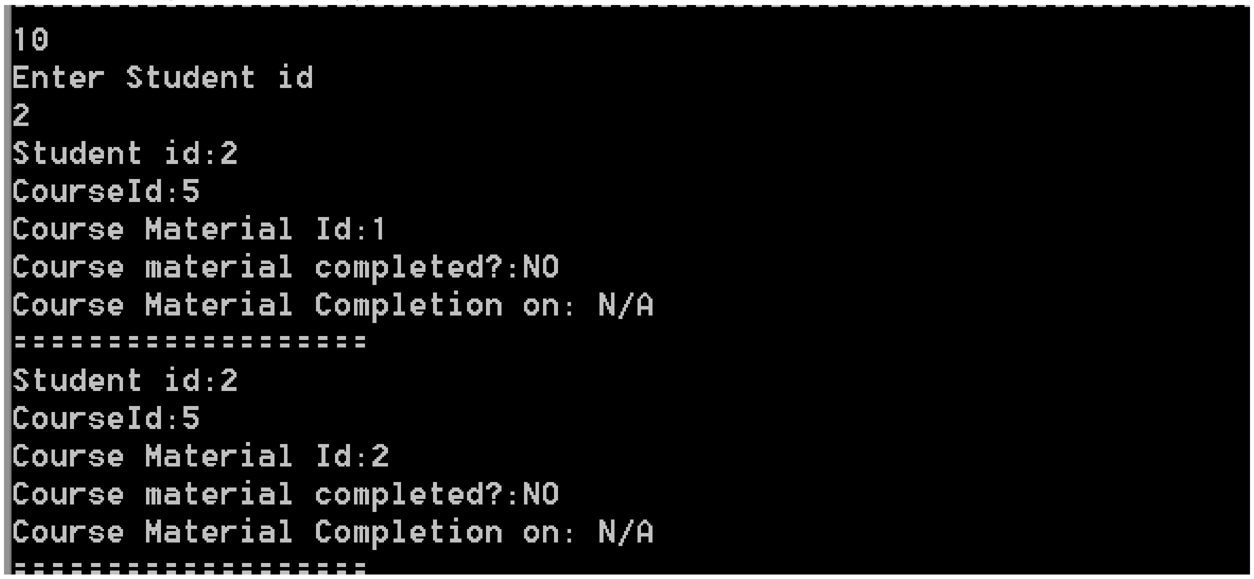
```
Enter Student id
14
Select another operation
```


```
9
Enter Student id
14
Enter Course id
1
Insert successful.
Select another operation
=======================================
```

Enrollment Successful. As you can see in the output below that when the student enrolls in a course, s/he is enrolled automatically in all course materials for that course.

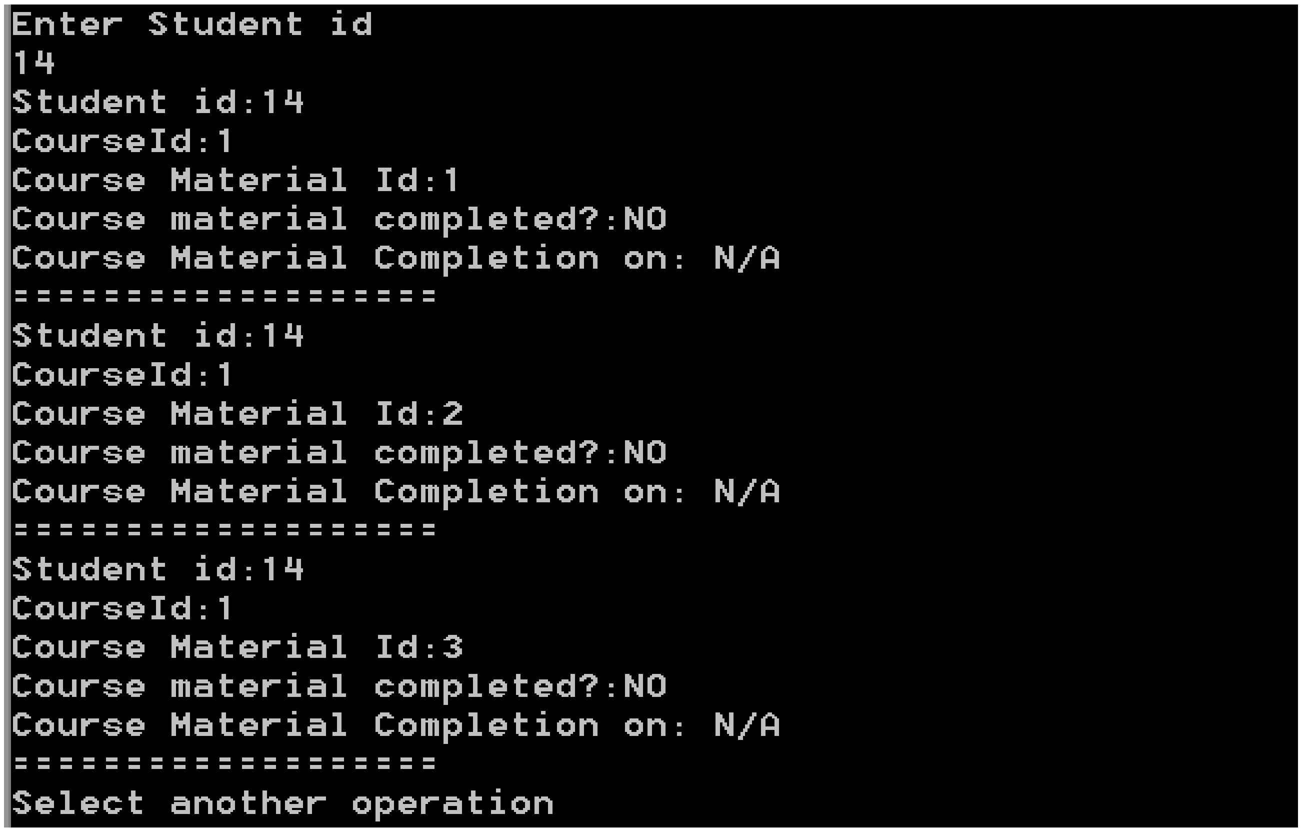

```
10
Enter Student id
14
Student id:14
CourseId:1
Course Material Id:1
Course material completed?:NO
Course Material Completion on: N/A
==================
Student id:14
CourseId:1
Course Material Id:2
Course material completed?:NO
Course Material Completion on: N/A
==================
Student id:14
CourseId:1
Course Material Id:3
Course material completed?:NO
Course Material Completion on: N/A
==================
```

5) Task 5: For a student enrolled in a course, list materials, in order, indicating the line of demarcation between completed/not completed


```
10
Enter Student id
2
Student id:2
CourseId:5
Course Material Id:1
Course material completed?:NO
Course Material Completion on: N/A
===================
Student id:2
CourseId:5
Course Material Id:2
Course material completed?:NO
Course Material Completion on: N/A
===================
```

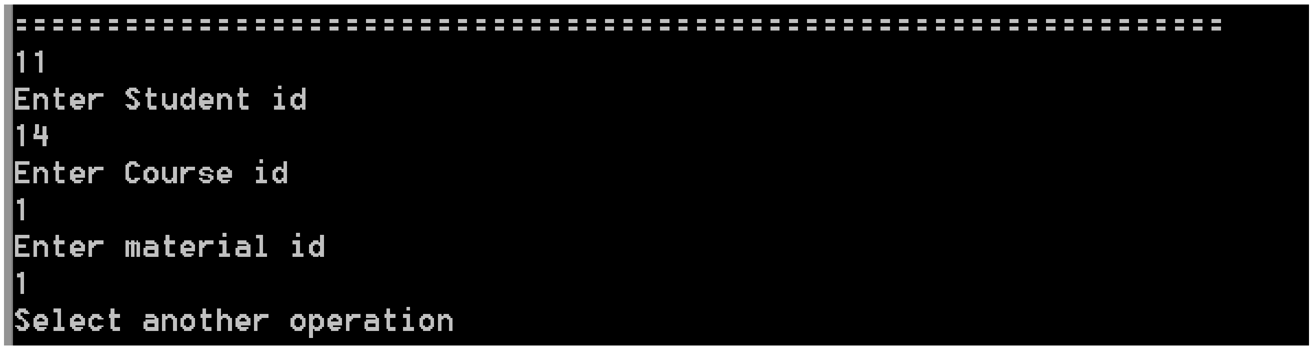
6) Task 6: Mark course material as having been completed by a student (possibly resulting in course completion)

Now a student (Student Id 14 starts with course 1 and all course materials are incomplete).



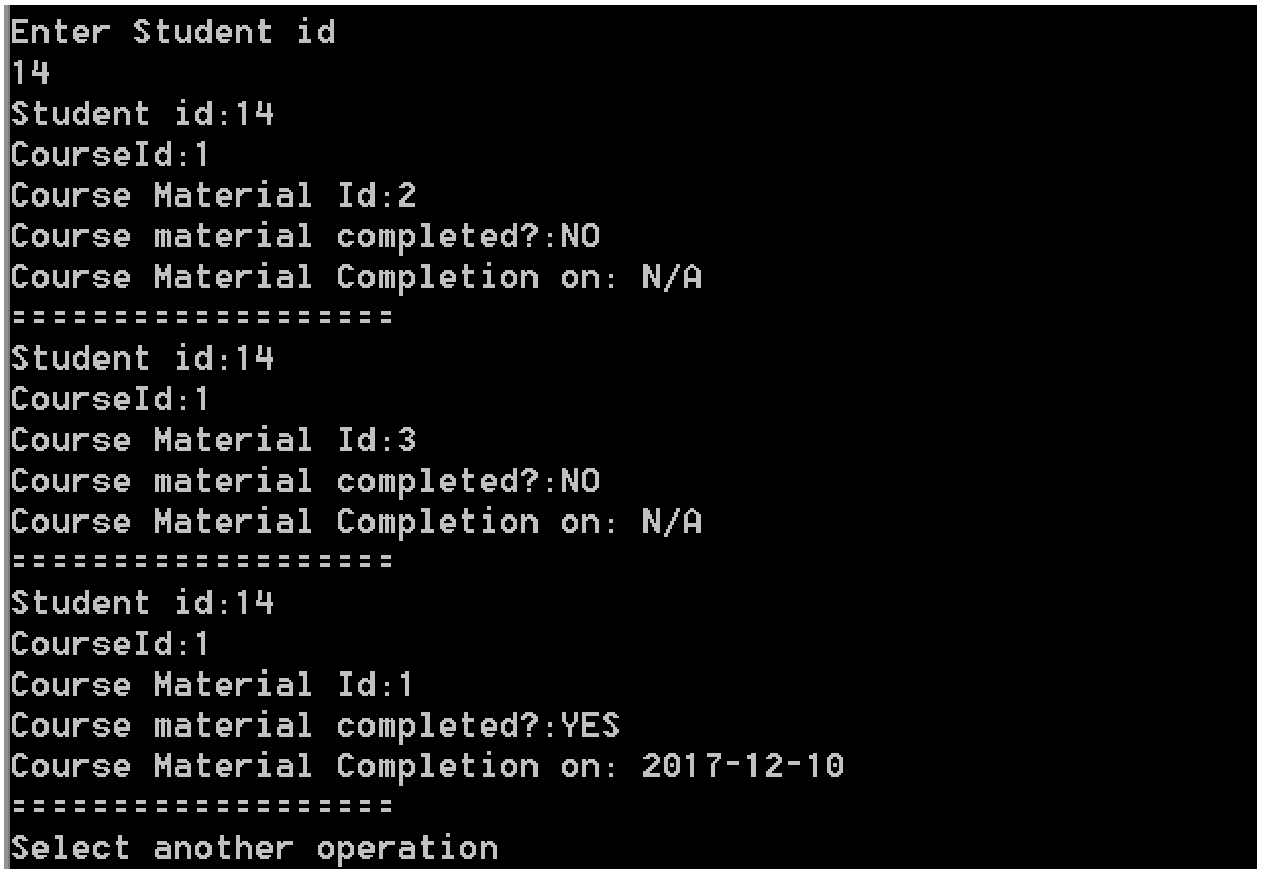
```
Enter Student id
14
Student id:14
CourseId:1
Course Material Id:1
Course material completed?:NO
Course Material Completion on: N/A
===================
Student id:14
CourseId:1
Course Material Id:2
Course material completed?:NO
Course Material Completion on: N/A
===================
Student id:14
CourseId:1
Course Material Id:3
Course material completed?:NO
Course Material Completion on: N/A
===================
Select another operation
```

Student then completes course material 1



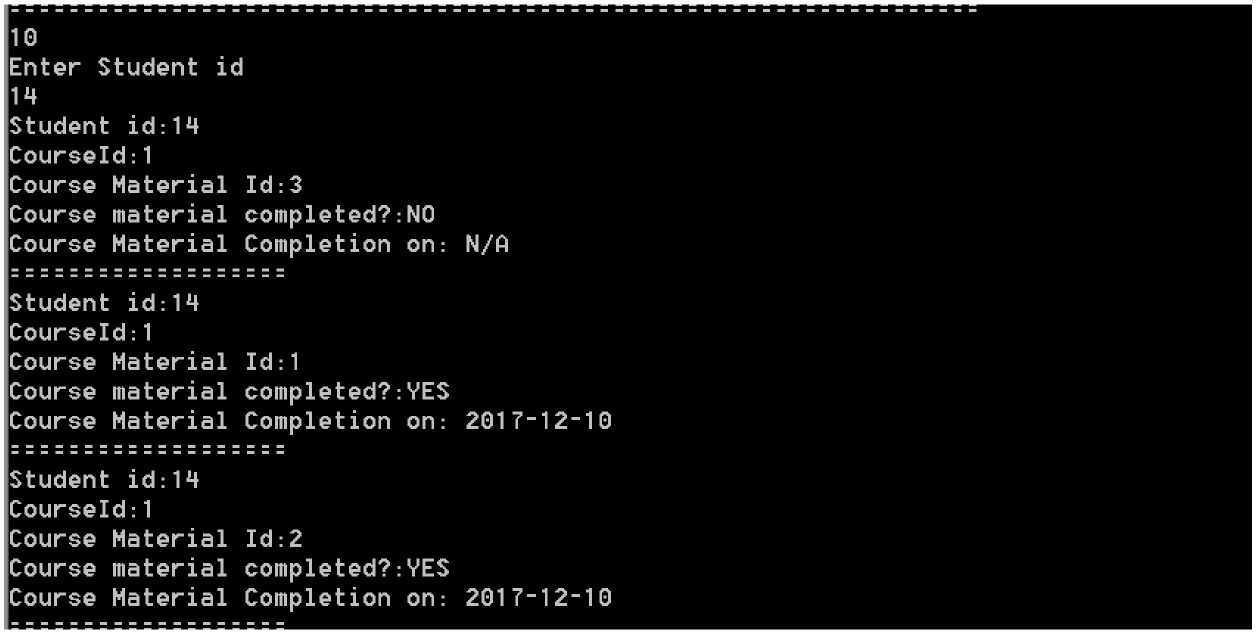
```
==================================================================
11
Enter Student id
14
Enter Course id
1
Enter material id
1
Select another operation
```

After student completes course material 1, this is the output-



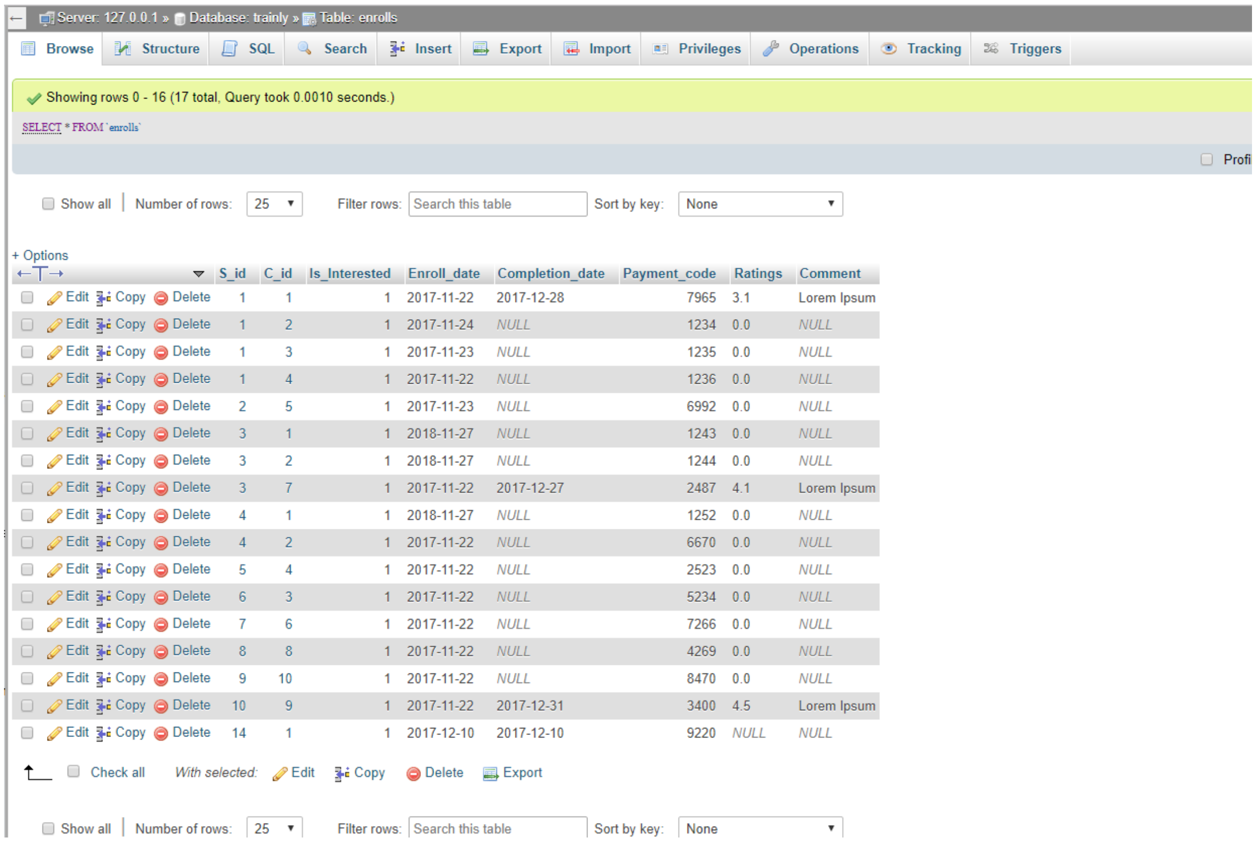
```
Enter Student id
14
Student id:14
CourseId:1
Course Material Id:2
Course material completed?:NO
Course Material Completion on: N/A
===================
Student id:14
CourseId:1
Course Material Id:3
Course material completed?:NO
Course Material Completion on: N/A
===================
Student id:14
CourseId:1
Course Material Id:1
Course material completed?:YES
Course Material Completion on: 2017-12-10
===================
Select another operation
```

Similarly, the student completes Course Material 2 and this is the output after it:



```
================================================================
10
Enter Student id
14
Student id:14
CourseId:1
Course Material Id:3
Course material completed?:NO
Course Material Completion on: N/A
==================
Student id:14
CourseId:1
Course Material Id:1
Course material completed?:YES
Course Material Completion on: 2017-12-10
==================
Student id:14
CourseId:1
Course Material Id:2
Course material completed?:YES
Course Material Completion on: 2017-12-10
==================
```

After student completes all the course material in a similar fashion, the enrolls table gets updated as well. Here is the database snapshot-



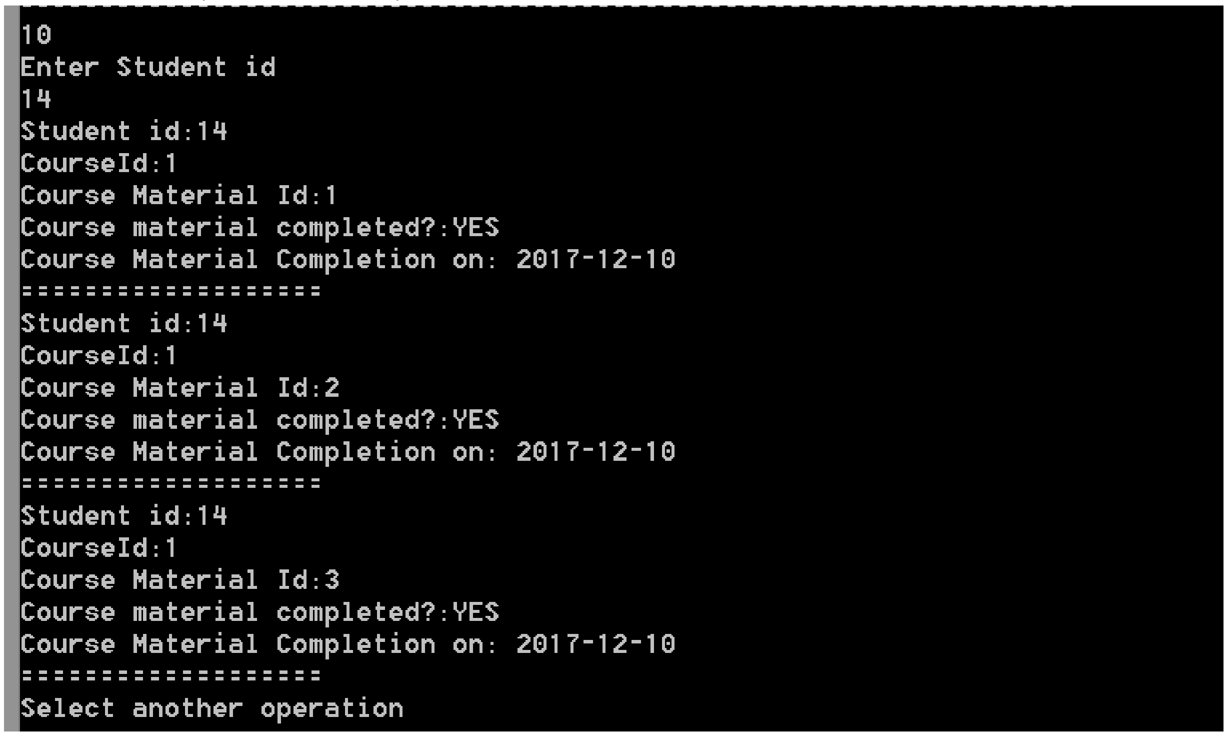
Server: 127.0.0.1 » Database: trainly » Table: enrolls

Showing rows 0 - 16 (17 total, Query took 0.0010 seconds.)

```
SELECT * FROM `enrolls`
```

S_id	C_id	Is_Interested	Enroll_date	Completion_date	Payment_code	Ratings	Comment
1	1	1	2017-11-22	2017-12-28	7965	3.1	Lorem Ipsum
1	2	1	2017-11-24	NULL	1234	0.0	NULL
1	3	1	2017-11-23	NULL	1235	0.0	NULL
1	4	1	2017-11-22	NULL	1236	0.0	NULL
2	5	1	2017-11-23	NULL	6992	0.0	NULL
3	1	1	2018-11-27	NULL	1243	0.0	NULL
3	2	1	2018-11-27	NULL	1244	0.0	NULL
3	7	1	2017-11-22	2017-12-27	2487	4.1	Lorem Ipsum
4	1	1	2018-11-27	NULL	1252	0.0	NULL
4	2	1	2017-11-22	NULL	6670	0.0	NULL
5	4	1	2017-11-22	NULL	2523	0.0	NULL
6	3	1	2017-11-22	NULL	5234	0.0	NULL
7	6	1	2017-11-22	NULL	7266	0.0	NULL
8	8	1	2017-11-22	NULL	4269	0.0	NULL
9	10	1	2017-11-22	NULL	8470	0.0	NULL
10	9	1	2017-11-22	2017-12-31	3400	4.5	Lorem Ipsum
14	1	1	2017-12-10	2017-12-10	9220	NULL	NULL

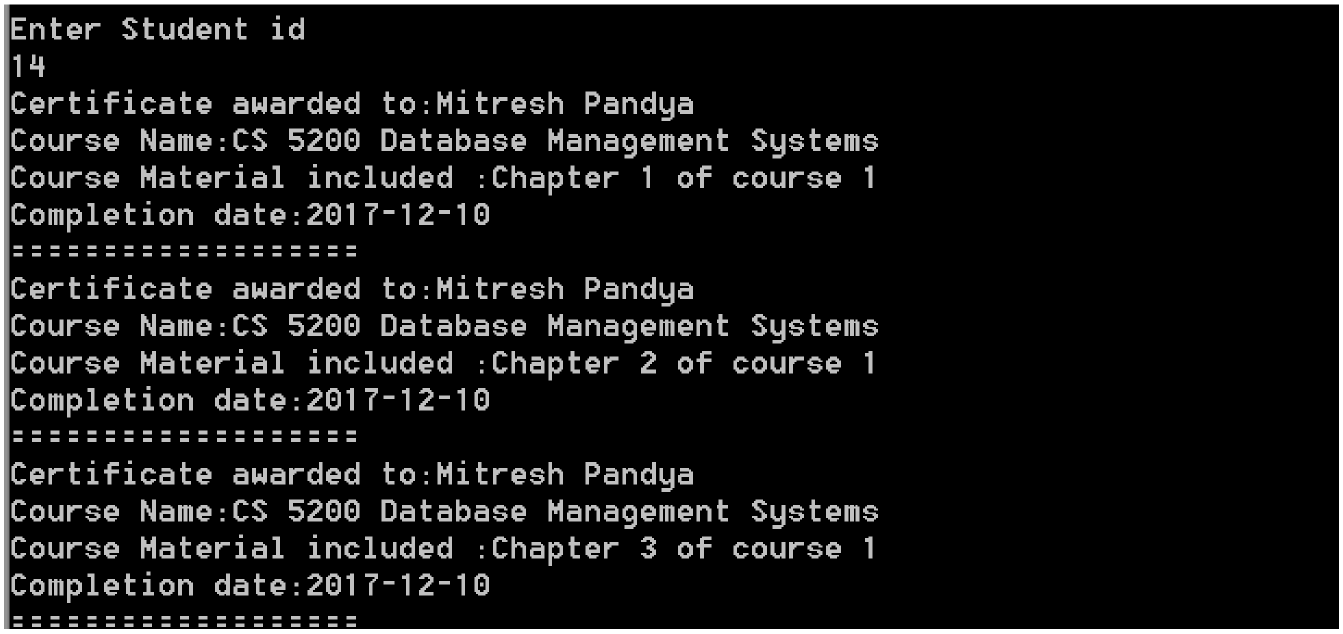
And also the snapshot of our output-



```
10
Enter Student id
14
Student id:14
CourseId:1
Course Material Id:1
Course material completed?:YES
Course Material Completion on: 2017-12-10
===================
Student id:14
CourseId:1
Course Material Id:2
Course material completed?:YES
Course Material Completion on: 2017-12-10
===================
Student id:14
CourseId:1
Course Material Id:3
Course material completed?:YES
Course Material Completion on: 2017-12-10
===================
Select another operation
```

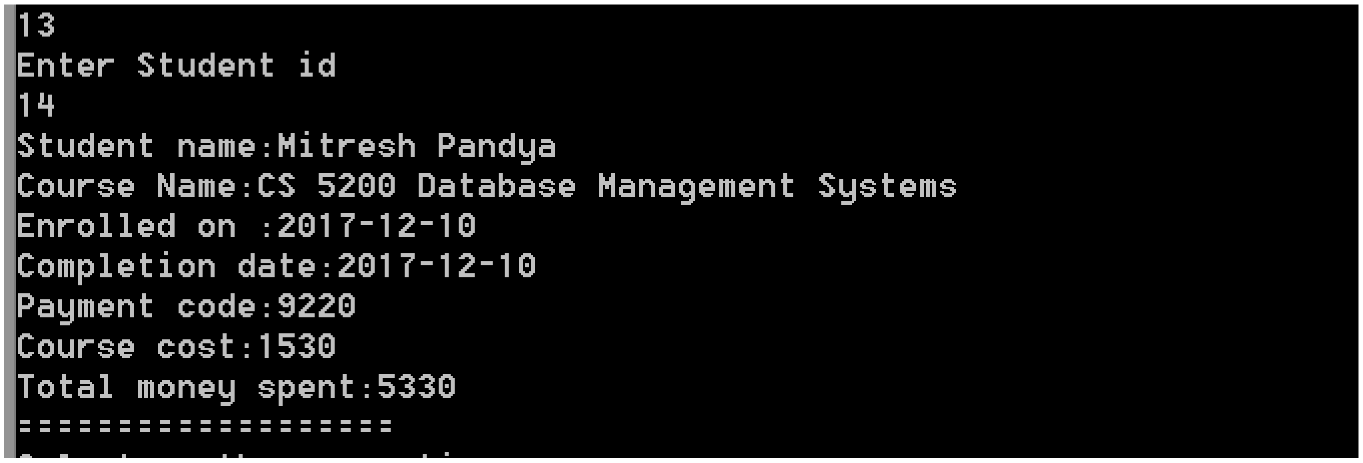
Hence the student completes the course.

7) Task 7: Provide a certificate of completion for a student (assuming s/he has successfully completed all materials)



```
Enter Student id
14
Certificate awarded to:Mitresh Pandya
Course Name:CS 5200 Database Management Systems
Course Material included :Chapter 1 of course 1
Completion date:2017-12-10
==================
Certificate awarded to:Mitresh Pandya
Course Name:CS 5200 Database Management Systems
Course Material included :Chapter 2 of course 1
Completion date:2017-12-10
==================
Certificate awarded to:Mitresh Pandya
Course Name:CS 5200 Database Management Systems
Course Material included :Chapter 3 of course 1
Completion date:2017-12-10
==================
```

8) Task 8: Provide an account history for a user: dates of enrollment/completion for each course, amount paid (with confirmation code), and total spent.



```
13
Enter Student id
14
Student name:Mitresh Pandya
Course Name:CS 5200 Database Management Systems
Enrolled on :2017-12-10
Completion date:2017-12-10
Payment code:9220
Course cost:1530
Total money spent:5330
===================
```

Project retrospective

The project exposed us to all basic required steps of database design and management in a practical setting. At the beginning of the project, we started with the Entity Relationship Diagram and transformed the narrative and specifications into diagrammatic view. Although this is one of the most important step, we did not like it that much. But we later on realized its importance and begin to pay more attention to each and every detail. This is because a properly created ERD, leads to a properly created Relational Diagram and hence we do not need to perform additional steps while writing actual DDL. Hence, we read it word for word, and considered every possible case during the design once we realized its true power. After countless revisions, we finalized the ERD diagram.

We think that the easiest part is ERD diagram but the hardest is also the ERD diagram. Because when we draw the ERD, it is very easy- just make some blocks and connect them. It's also hardest because we need to make sure it contains all the meaning expressed in the description and it is where the actual transformation of Idea to actual DBMS starts. Thus, if foundation is weak, the whole system will be weak. We also realized that we face a different kind of problem when we are actually filling the sample data using DML. We realized that, many of the tables we actually made did not work well and hence we had to modify the DDL and also the whole chain in backward fashion – DML -> DDL -> Tables -> ERD. So, filling the tables and hence DML part was most enlightening and fun. Hence that was a very good experience. We all think that we learnt a lot from this project. First is, we learnt how to create a complete database management system from scratch and got exposed to all major design steps involved. We also learnt what were the loop holes in our system when filling actual data and hence when we will be designing the database later on, we can use that experience and build a better DDL design. We also learnt teamwork and how to divide the work and yet build a single working system. All of us worked on different parts – One of us worked on ERD, other on relational diagrams, the third one on DDL and DML and fourth one on creating the Java CLI. Although we all took parts in everything, but this was how we divided the majority of tasks.

Conclusion statement:

Hence a working Command Line Interface was produced for a database `trainly` for **trainly.io**. This database can be later on integrated with web or mobile interface (Changing to sqlite preferred here for mobile). Apart from this, the logging part is left and the database encryption part and advanced practices were also left. What we have is a fully functional basic database system which can be integrated into almost all systems (maybe with slight modifications) in different sectors of industry (where database is required). We also realized the true power of database and how each and every data and transaction has to be modified according to use cases – what to index, where to index, when to denormalize, how to deal with foreign key constraints when adding data in individuals etc. Overall it was a great learning curve and very enlightening one which equipped us to integrate Databases in all our future projects within limited time frame.
