Project Report

Database Record of Coursera

Group Members :

Dilip Puri(201351014)
Vivek Kumar Singh(201352015)
Chirag Panpalia(201351001)

Instructor : Professor P M Jat

Indian Institute of Information Technology, Vadodara

April 21, 2015

Contents

Indian Institute of Information Technology,Vadodara

Database Management System

(4th semester)

DBMS Project

Coursera

Course : **Database Management System(CSE205)**

Instructor : **P M Jat**

TAs :

Brijesh Patel
Milan Kathrotia

Project Members :

Dilip Puri(201351014)
Vivek Kumar Singh(201352015)
Chirag Panpalia(201351001)

Acknowledgement

We are grateful to professor **P M Jat**, professor at **Indian Institute of Information Technology, Vadodara** and Teaching Assistants **Brijesh Patel** & **Milan Kathrotia** of course **DBMS** at IIIT, Vadodara.

For permission to reproduce copyright material in this document we would like to thank the following :

dia software for making ERDiagram

LibreOffice for making Schema

postgreSQL(pg Admin III) for using this for our keep record of database

Coursera for coursera logo and some data information

LaTeXfor using LaTeXplatform to produce this document

and also thankful to our classmates. The efforts of the classmates we are also highly appreciated.

Project members

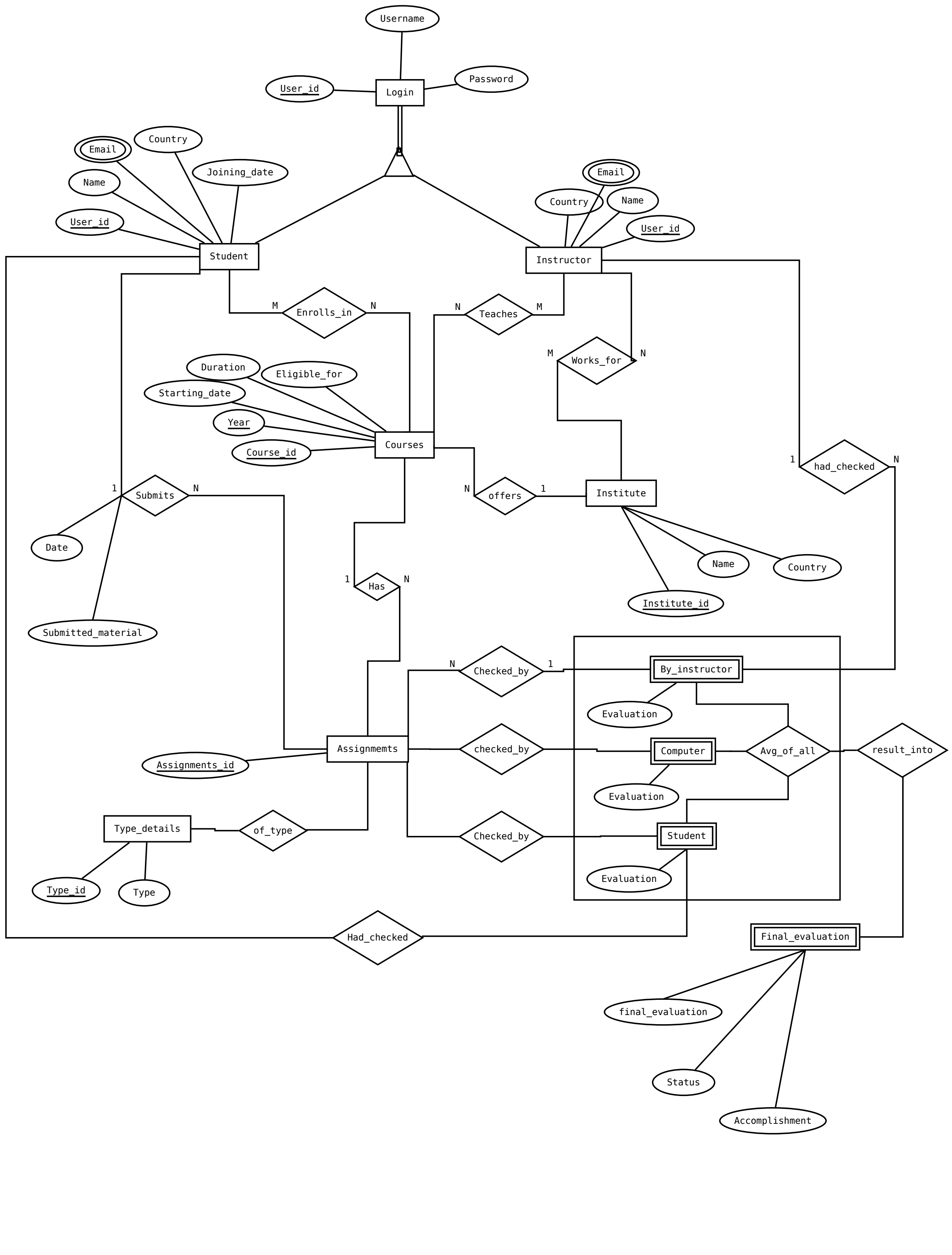
0.1 Description

This description defines our **CS205** project named as **Online course system (coursera)**. Our main aim is to build an Online course system in perspective of **Database Management System**. How will our database system store required information in order to complete participation.
Online course system fundamentally co-ordinates the courses of students of specific university. The system will keep record of Students, courses, Instructors, Institutions and student′s participation etc.

0.2 Scope of our project

Since we knew the importance of e-education and how it is helpful in accessing free education to everyone so this project will help us to understand how e-education works. How they maintain such a huge database. How they inter-relate or distinguish data of all the users. How they retrieve important data from all the raw data and how to make them more efficient. This project will lead us to understand the important aspects of Database and SQL quries.

0.3 ER Diagram and Schema(Coursera)


Username
User_id
Login
Password
Email
Country
Joining_date
Name
User_id
Student
Email
Country
Name
User_id
Instructor
M
Enrolls_in
N
N
Teaches
M
M
Works_for
N
Duration
Eligible_for
Starting_date
Year
Course_id
Courses
1
had_checked
N
1
Submits
N
N
offers
1
Institute
Date
Name
Country
Institute_id
1
Has
N
Submitted_material
N
Checked_by
1
By_instructor
Evaluation
Assignmemts
checked_by
Computer
Avg_of_all
result_into
Assignments_id
Evaluation
Type_details
of_type
Checked_by
Student
Evaluation
Type_id
Type
Had_checked
Final_evaluation
final_evaluation
Status
Accomplishment

login		
user_id	username	password

institution		
institute_id	institute_name	country

instructor				
instructor_id	instructor_name	email	country	institute_id

student				
student_id	student_name	email	country	joining_date

course						
course_id	coursename	startdate	duration	language	yrs	eligible_for

course_instructor		
course_id	yrs	instructor_id

enroll			
student_id	course_id	yrs	joindate

discussion				
question_id	course_id	user_id	yrs	question

answer			
answer_id	question_id	user_id	answer

content		
course_id	yrs	material

assignmenttype	
type_id	type

assignment						
assignment_id	course_id	yrs	assignment_type	topic	due_date	instructor_id

submission			
assignment_id	student_id	submission_date	submit_material

peer			
assignment_id	user_id	student_id	evaluation

computer		
assignment_id	student_id	evaluation

instruct_evaluation			
assignment_id	instructor_id	student_id	evaluation

final_evaluation			
user_id	course_id	yrs	final_evaluation

0.4 DDL file

```
--creating schema----------------------
create schema coursera;

--set search path----------------------
set search_path to coursera;

------------------------------------------
--Now we are going to create tables --
------------------------------------------



------------------------------------------------------------------
--Login table /* In this table, we allow all the user  --
-- who sign in for online course at coursera */   --
------------------------------------------------------------------
create table login(
  user_id varchar(15) primary key,
  name varchar(20),
  passwd varchar(15)

);

----------------------------------------------------------------------------------
--Institution table /* In this table, we list all the institute   --
--we are going to take part(means offers course(s) at our online program) --
--in this MOOC system */                --
----------------------------------------------------------------------------------

create table institution(
  institute_id varchar(15) primary key,
  institute_name varchar(40),
  country varchar(20)

);

--------------------------------------------------------------------------
--Instructor table /* In this table, we list all the Instructors  --
--who teaches online course in our program */      --
--------------------------------------------------------------------------
create table instructor(
  instructor_id varchar(15),
  instructor_name varchar(15),
  email varchar(20),
  country varchar(20),
  institute_id varchar(15),
  FOREIGN KEY(instructor_id) references login(user_id)
  ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY(institute_id) references institution(institute_id)
  ON DELETE CASCADE ON UPDATE CASCADE,
  primary key(instructor_id)

);

------------------------------------------------------------------
--Student table /* In this table, we list all the students  --
--who are participate(enroll) in this program */  --
------------------------------------------------------------------
create table student(
  student_id varchar(15),
  student_name varchar(20),
  email varchar(30),
  country varchar(20),
  joining_date date,
  FOREIGN KEY(student_id) references login(user_id)
  ON DELETE CASCADE ON UPDATE CASCADE,
  primary key(student_id)

);

----------------------------------------------------------
--Course table /* In this table, we list all the course --
--which availabel in this our MOOC project */  --
----------------------------------------------------------
create table course(
  course_id varchar(15),
  coursename text,
  startdate date,
  duration varchar(4),
  language varchar(15),
  yrs int,
  eligible_for varchar(15),
  primary key(course_id, yrs)

);

----------------------------------------------------------------------------------
--Course_instructor table /* In this table, we list all the instructors  --
--and their courses */                  --
----------------------------------------------------------------------------------
create table course_instructor(
  course_id varchar(15),
  instructor_id varchar(15),
  yrs int,
  FOREIGN KEY(course_id, yrs) references course(course_id, yrs)
  ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY(instructor_id) references instructor(instructor_id)
  ON DELETE CASCADE ON UPDATE CASCADE,
  primary key(course_id, yrs, instructor_id)
);
```

```
------------------------------------------------------------------------------------
--Enroll table /* In this table, we list which student enroll for which course
   */--
------------------------------------------------------------------------------------
create table enroll(
  student_id varchar(15),
  course_id varchar(15),
  yrs int,
  joindate date,
  FOREIGN KEY(student_id) references student(student_id)
  ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY(course_id, yrs) references course(course_id, yrs)
  ON DELETE CASCADE ON UPDATE CASCADE,
  primary key(student_id, course_id, yrs)

);

-----------------------------------------------------------------------------------
--Discussion Forum table /* In this table, we list monitor board of students  --
--that how many s/he has current, past, or upcoming */      --
-----------------------------------------------------------------------------------
create table discussion(
  question_id varchar(15),
  course_id varchar(15),
  yrs int,
  user_id varchar(15),
  question text,
  FOREIGN KEY(course_id, yrs) references course(course_id, yrs)
  ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY(user_id) references login(user_id)
  ON DELETE CASCADE ON UPDATE CASCADE,
  primary key(question_id)

);

-----------------------------------------------------------------------------------
--Answer table /* In this table, all the users may give their answers about  --
--the posted question(student and instructor) */      --
-----------------------------------------------------------------------------------
create table answer(
  answer_id varchar(15),
  question_id varchar(15),
  user_id varchar(15),
  ans text,
  FOREIGN KEY(question_id) references discussion(question_id)
  ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY(user_id) references login(user_id)
  ON DELETE CASCADE ON UPDATE CASCADE,
  primary key(answer_id)
);

------------------------------------------------------------------------------------------
--Course Content table /* In this table, all the content like lectures, video
   lectures, --
-- and assignments etc. */                 --
------------------------------------------------------------------------------------------
create table content(
  course_id varchar(15),
  yrs int,
  material text,
  FOREIGN KEY(course_id, yrs) references course(course_id, yrs)
  ON DELETE CASCADE ON UPDATE CASCADE,
  primary key(course_id, yrs)

);

----------------------------------------------------------------------------
--Assignments Type table /* In this table, list of all the assignments */--
----------------------------------------------------------------------------
create table assignmenttype(
  type_id varchar(15) primary key,
  type varchar(30)

);

------------------------------------------------------------------------------------------
--Assignments table /* In this table, all the assignments to be listed like
   quizzes,  --
peer assesments, and final exam etc. */              --
------------------------------------------------------------------------------------------
create table assignment(
  assignment_id varchar(15),
  course_id varchar(15),
  yrs int,
  instructor_id varchar(15),
  assignment_type varchar(15),
  topic text,
  due_date date,
  FOREIGN KEY(course_id, yrs) references course(course_id, yrs)
  ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY(instructor_id) references instructor(instructor_id)
  ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY(assignment_type) references assignmenttype(type_id)
  ON DELETE CASCADE ON UPDATE CASCADE,
  primary key(assignment_id)

);

------------------------------------------------------------------------------------------
```

```
--Submission table /* In this table, all the submission to be listed here like
    quizzes, --
--peer assesements, and final exam(submission) */         --
-------------------------------------------------------------------------------------------
create table submission(
   assignment_id varchar(15),
   submitted_material text,
   submission_date date,
   student_id varchar(15),
   FOREIGN KEY(assignment_id) references assignment(assignment_id)
   ON DELETE CASCADE ON UPDATE CASCADE,
   FOREIGN KEY(student_id) references student(student_id)
   ON DELETE CASCADE ON UPDATE CASCADE,
   primary key(assignment_id, student_id)

);

-------------------------------------------------------------------------------------------
--Peers Evaluation table /* In this table, all the peer will evaluate by others
    peers */--
-------------------------------------------------------------------------------------------
create table peer(
   assignment_id varchar(15),
   student_id1 varchar(15),
   student_id2 varchar(15),
   evaluation int,
   FOREIGN KEY(assignment_id) references assignment(assignment_id)
   ON DELETE CASCADE ON UPDATE CASCADE,
   FOREIGN KEY(student_id1) references student(student_id)
   ON DELETE CASCADE ON UPDATE CASCADE,
   FOREIGN KEY(student_id2) references student(student_id)
   ON DELETE CASCADE ON UPDATE CASCADE,
   primary key(student_id1, student_id2, assignment_id)

);

------------------------------------------------------------------------------------------------
--Computerized Evaluation table /* In this table, all the student will evaluate
    by computer(like quizzes) */--
------------------------------------------------------------------------------------------------
create table computer(
   assignment_id varchar(15),
   user_id varchar(15),
   evaluation int,
   FOREIGN KEY(assignment_id) references assignment(assignment_id)
   ON DELETE CASCADE ON UPDATE CASCADE,
   FOREIGN KEY(user_id) references login(user_id)
   ON DELETE CASCADE ON UPDATE CASCADE,
   primary key(user_id, assignment_id)

);
```

```
-------------------------------------------------------------------------------------------------
--Instructor Evaluation table /* In this table, all the student will evaluate by
   instructor(like final exam) */--
-------------------------------------------------------------------------------------------------
create table instruct_evaluation(
  assignment_id varchar(15),
  student_id varchar(15),
  instructor_id varchar(15),
  evaluation int,
  FOREIGN KEY(assignment_id) references assignment(assignment_id)
  ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY(instructor_id) references instructor(instructor_id)
  ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY(student_id) references student(student_id)
  ON DELETE CASCADE ON UPDATE CASCADE,
  primary key(assignment_id, student_id)

);
-------------------------------------------------------------------------------------------------
```

0.4.1 final evaluation view

```
------------------------------------------
 set  search_path to coursera;
------------------------------------------
create view final_evaluation as(
SELECT user_id,course_id,yrs,sum(points)as sum2
From (
(SELECT
computer.user_id as user_id,
  course.course_id,
  course.yrs,

  sum(computer.evaluation)as points
FROM
  coursera.course,
  coursera.assignment,
  coursera.computer
WHERE
  course.course_id = assignment.course_id AND
  assignment.assignment_id = computer.assignment_id
GROUP BY computer.user_id, course.course_id,course.yrs
order by course.course_id,course.yrs
)
UNION

(SELECT
instruct_evaluation.student_id as user_id,
  course.course_id,
  course.yrs,

  sum(instruct_evaluation.evaluation)as points
FROM
  coursera.course,
  coursera.assignment,
  coursera.instruct_evaluation
WHERE
  course.course_id = assignment.course_id AND
  assignment.assignment_id = instruct_evaluation.assignment_id
GROUP BY instruct_evaluation.student_id, course.course_id,course.yrs
order by course.course_id,course.yrs
)
UNION
(SELECT
  peer.student_id1 as user_id,
  course.course_id,
  course.yrs,
  sum(peer.evaluation)as points
FROM
  coursera.peer,
  coursera.course,
  coursera.assignment
WHERE
  course.course_id = assignment.course_id AND
  assignment.assignment_id = peer.assignment_id
GROUP BY peer.student_id1, course.course_id,course.yrs
order by course.course_id,course.yrs
)

) as x
GROUP BY x.user_id,x.course_id,x.yrs
order by x.course_id,x.yrs);
-------------------------------------------------------------------
```

You can download all the files from this given link
https://www.dropbox.com/sh/c8etydmaf9k0fh9/AADLZ4giOTLjbjkV7nbNOeDGa?dl=0

0.5 Relations, Functional Dependancies(FDs), and Boyce-Codd Normal Form(BCNF)

login

Key: user_id
FDs: user_id $\mapsto$ {user_name, password}

Comment: Relation is in BCNF.

institution

Key: institute_id
FDs: institute_id $\mapsto$ {institute_name, country}

Comment: Relation is in BCNF.

instructor

Key: instructor_id
Foreign Key(s): instructor_id references **login**(user_id)
FDs: instructor_id $\mapsto$ {instructor_name, email, country, institute_id}

Comment: Relation is in BCNF.

student

Key: student_id
Foreign Key(s): student_id references **login**(user_id)
FDs: student_id $\mapsto$ {student_name, email, country, joining_date}

Comment: Relation is in BCNF.

course

Key: (course_id, yrs)
FDs: {course_id, yrs} $\mapsto$ {coursename, startdate, duration, language, eligible_for}

Comment: Relation is in BCNF.

course_instructor

Key: (course_id, yrs, instructor_id)
Foreign Key(s): instructor_id references **instructor**(instructor_id)
(course_id,yrs) references **course**(course_id, yrs)

Comment: Relation is in BCNF.

enroll

Key: (student_id, course_id, yrs)
Foreign Key(s): student_id references **student**(student_id)
(course_id,yrs) references **course**(course_id, yrs)
FDs: $\{$student_id, course_id, yrs$\} \mapsto \{$joindate$\}$

Comment: Relation is in BCNF.

discussion

Key: (question_id, course_id, user_id, yrs)
Foreign Key(s): user_id references **login**(user_id)
(course_id,yrs) references **course**(course_id, yrs)
FDs: $\{$question_id, course_id, user_id, yrs$\} \mapsto \{$question$\}$

Comment: Relation is in BCNF.

answer

Key: (answer_id, question_id, user_id)
Foreign Key(s): question_id references **discussion**(question_id)
user_id references **login**(user_id)
FDs: $\{$answer_id, question_id, user_id$\} \mapsto \{$answer$\}$

Comment: Relation is in BCNF.

assignmenttype

Key: type_id
FDs: type_id $\mapsto \{$type$\}$

Comment: Relation is in BCNF.

assignment

Key: (assignment_id, course_id, yrs)
Foreign Key(s): assignment_type references **assignmenttype**(type_id)
(course_id,yrs) references **course**(course_id, yrs)
instructor_id references **instructor**(instructor_id)
FDs: $\{$assignment_id, course_id, yrs$\} \mapsto \{$assignment_id, topic, due_date, instructor_id$\}$

Comment: Relation is in BCNF.

submission

Key: (assignment_id, student_id)
Foreign Key(s): assignment_id references **assignment**(assignment_id)
student_id references **student**(student_id)

FDs: {assignment_id, student_id} $\mapsto$ {submission_date, submit_material}

Comment: Relation is in BCNF.

peer

Key: (assignment_id, user_id)
Foreign Key(s): assignment_id references **assignment**(assignment_id)
user_id references **login**(user_id)
FDs: {assignment_id, user_id} $\mapsto$ {student_id, evaluation}

Comment: Relation is in BCNF.

computer

Key: (assignment_id, student_id)
Foreign Key(s): assignment_id references **assignment**(assignment_id)
student_id references **student**(student_id)
FDs: {assignment_id, student_id} $\mapsto$ {evaluation}

Comment: Relation is in BCNF.

instruct_evaluation

Key: (assignment_id, course_id, yrs)
Foreign Key(s): assignment_id references **assignment**(assignment_id)
instructor_id references **instructor**(instructor_id)
FDs: {assignment_id, instructor_id, student_id} $\mapsto$ {evaluation}

Comment: Relation is in BCNF.

final_evaluation

Key: (student_id, course_id, yrs)
Foreign Key(s): (student_id, course_id, yrs) references **enroll**(student_id, course_id, yrs)
FDs: {student_id, course_id, yrs} $\mapsto$ {final_evaluation, status, accomplishment}

Comment: Relation is in BCNF.

0.6 Queries, SQL, and Outputs

What kind of queries can be ask to our database?

There are many queries that can be fired on our database.
There are some common queries related to **student, course, institution, instructor, discussion forum, assignments, submission, evalaution(peers, computerized, instructor)**.

0.6.1 Assumptions

- If student has scored less than 3 out of 5 in particular assignment then his/her answer consider wrong.
- If student has scored less than 13 points out of 30 then s/he consider failed in course.

0.6.2 Quries and Output

1. List of students who have taken course that is offered by an institute same country as student have.

   SQL

   ```
    SELECT  distinct
     student.student_name,
     student.country
   FROM
     coursera.enroll,
     coursera.course_instructor,
     coursera.instructor,
     coursera.institution,
     coursera.student
   WHERE
     enroll.course_id = course_instructor.course_id AND
     enroll.yrs = course_instructor.yrs AND
     enroll.student_id = student.student_id AND
     course_instructor.instructor_id = instructor.instructor_id AND
     instructor.institute_id = institution.institute_id AND
     student.country = institution.country;
   ```

   Output

   ```
   'student_name';'country'
   'shewata tiwari';'canada'
   'pratik paliwal';'canada'
   'vishal dilip nagrale';'usa'
   'mukesh sahu';'usa'
   'vishal vaishanv';'india'
   'charu chimpa';'india'
   'sharad patel';'india'
   'shresthi priya';'canada'
   ```

```
'dheeraj reddy';'india'
'vikash singh';'usa'
'ashish kumar unni';'canada'
'ayush lamba';'usa'
'krishann unni';'india'
'sourabh jain';'canada'
'lalit kumar';'india'
'anand rahul';'usa'
'avi aryan';'usa'
'prem chand saini';'usa'
'aashish yadvally';'india'
'lalit singh';'india'
```

2. List of courses that have available in at least 3 different languages and provided by different universities.

SQL

```
 SELECT  distinct
  course.course_id,
  count(course.course_id)
FROM
  coursera.course,
  coursera.institution,
  coursera.course_instructor,
  coursera.instructor
WHERE
  course.course_id = course_instructor.course_id AND
  course.yrs = course_instructor.yrs AND
  course_instructor.instructor_id = instructor.instructor_id AND
  instructor.institute_id = institution.institute_id
GROUP BY
  course.course_id
HAVING
  count(course.course_id) > 1;
```

Output

```
'course_id';'count'
'ml01';'4'
'hci01';'3'
'itf02';'3'
'cpt01';'3'
'aiipp01';'2'
'smsd01';'2'
'cs01';'4'
'mn01';'2'
'se01';'2'
'itom01';'2'
```

3. List of those students who have not completed a course and they enroll again in that

course.

SQL

```
 SELECT  distinct
  student.student_id,
  student.student_name
FROM
   (SELECT
    r1.student_id
   FROM
      (SELECT
         enroll.student_id,
         enroll.course_id,
         count(enroll.yrs)
      FROM
         coursera.enroll
      GROUP BY
         enroll.student_id,
         enroll.course_id
      HAVING count(enroll.yrs) >= 2) as r1
   JOIN
     coursera.enroll
   ON
   (r1.student_id = enroll.student_id and r1.course_id = enroll.course_id))
      as r2
JOIN
 coursera.student
ON
r2.student_id = student.student_id;
```

Output

```
'student_id';'student_name'
'201412';'shresthi priya'
'201502';'lalit kumar'
'201542';'lalit singh'
```

4. Find out those computer science courses which are taught by professor of 'MIT'.

SQL

```
 SELECT  distinct
  course.coursename
FROM
  coursera.institution,
  coursera.course_instructor,
  coursera.instructor,
  coursera.course
WHERE
  institution.institute_id = instructor.institute_id AND
  course_instructor.instructor_id = instructor.instructor_id AND
  course.course_id = course_instructor.course_id AND
  course.yrs = course_instructor.yrs AND
  institution.institute_name = 'mit';
```

Output

```
'coursename'
'Introduction to Finance'
'Model Thinking'
```

5. Rank the computer sciece courses on the basis of no. of registration of the course in last 3 years.

   SQL

```
 SELECT
  enroll.course_id,
  count(enroll.course_id)
FROM
   coursera.enroll
GROUP BY
  enroll.course_id
ORDER BY
  enroll.course_id;
```

Output

```
'course_id';'count'
'aiipp01';'27'
'cpt01';'22'
'fppc01';'12'
'itd01';'20'
'itf02';'27'
'mt01';'11'
```

6. List of those students who discussed at least 1 question which are posted in discussion forum.

   SQL

```
 SELECT  distinct
  student.student_name,
  student.email
FROM
  coursera.answer,
  coursera.student
WHERE
  answer.user_id = student.student_id;
```

Output

```
'student_name';'email'
'ayush lamba';'ayushlamba12@gmail.com'
```

```
'venkata sairahul';'sairahul12@gmail.com'
'lalit kumar';'lalitkumar12@gmail.com'
'chahat jain';'chahatjain12@gmail.com'
'anugu bharath reddy';'anugubharath12@gmail.com'
'aashish yadvally';'aashish12@gmail.com'
'sharad patel';'sharadpatel12@gmail.com'
'vineela chandra';'vineela12@gmail.com'
'vikash singh';'vikashsingh12@gmail.com'
'shewata tiwari';'shweta12@gmail.com'
'anand rahul';'rahulanand12@gmail.com'
'ankit kumar';'ankitkumar12@gmail.com'
'shresthi priya';'spriya12@gmail.com'
'pratik paliwal';'palwal12@gmail.com'
```

7. Find out thse students who submit peer assessments and whose peer assessnment checked by peers have same country.

SQL

```
 SELECT  distinct
  s1.student_name,
  s1.country,
  s2.student_name,
  s2.country
FROM
  coursera.peer,
  coursera.student s1,
  coursera.student s2
WHERE
  peer.student_id2 = s1.student_id AND
  peer.student_id1 = s2.student_id AND
  s1.country = s2.country;
```

Output

```
'student_name';'country';'student_name';'country'
'shresthi priya';'canada';'shewata tiwari';'canada'
'dheeraj reddy';'india';'lalit kumar';'india'
'shewata tiwari';'canada';'shresthi priya';'canada'
'lalit kumar';'india';'dheeraj reddy';'india'
'vineela chandra';'uk';'harshit purohit';'uk'
'harshit purohit';'uk';'vineela chandra';'uk'
```

1. List of those students whose questions are answered by this(alka parikh) professor.

SQL

```
 SELECT
  answer.ans,
  discussion.question,
  student.student_name
FROM
```

```
  coursera.discussion,
  coursera.answer,
  coursera.student
WHERE
  discussion.question_id = answer.question_id AND
  student.student_id = discussion.user_id AND
  answer.user_id = 'mit0101';
```

Output

```
"ans";"question";"student_name"
"Is this course provide cretificate?-no";"Is this course provide
    cretificate?";"vishal vaishanv"
"Sir I have problem with submission? I dont find link. - go to help desk and
    request for link.";"Sir I have problem with submission? I dont find
    link.";"mukesh kumar"
"Can I enroll in this course again?-yes you can.";"Can I enroll in this
    course again?";"prakash vajekar"
```

2. List all the students who have answered all the correct answers in all the courses.

SQL

```
 SELECT
student.student_name
FROM
coursera.student,(SELECT
  instruct_evaluation.student_id
FROM
  coursera.instruct_evaluation
except
SELECT
  instruct_evaluation.student_id
FROM
  coursera.instruct_evaluation
WHERE
  instruct_evaluation.evaluation < 3)as x

WHERE
student.student_id=x.student_id
  ;
```

Output

```
"student_name"
"vishal vaishanv"
"venkata sairahul"
"lalit kumar"
"mukesh kumar"
"dheeraj reddy"
"pratik paliwal"
```

3. Top 2 successful(criteria : most of student enrolled and 60 percent passed) courses of this year(2014).

   SQL

```
select x.course_id,count(x.student_id) as no_of_passing_students
from
((SELECT
 assignment.course_id,
 instruct_evaluation.assignment_id,
  instruct_evaluation.student_id,

  instruct_evaluation.evaluation

FROM

  coursera.instruct_evaluation,
  coursera.assignment
  where
  assignment.assignment_id=instruct_evaluation.assignment_id and
      instruct_evaluation.evaluation>2
  )
UNION
(SELECT
assignment.course_id,

peer.assignment_id,
peer.student_id1,
  peer.evaluation


FROM

  coursera.peer,
  coursera.assignment
  where
  assignment.assignment_id=peer.assignment_id and peer.evaluation>2
  )
  UNION
  (SELECT
assignment.course_id,
computer.assignment_id,
  computer.user_id,

  computer.evaluation


FROM

  coursera.computer,
  coursera.assignment
  where
  assignment.assignment_id=computer.assignment_id and computer.evaluation>2

)
)as x
where x.evaluation>3
group by x.course_id
order by no_of_passing_students desc limit(2)
;
```

Output

```
"course_id";"no_of_passing_students"
"mt01";21
"itf02";19
```

4. Give the email of student(s) who have enrolled in maximum number of courses.

SQL

```
 SELECT
  student.student_name,
  student.email,
  count(student.student_id) as no_of_courses
FROM
  coursera.enroll,
  coursera.course,
  coursera.student
WHERE
  enroll.course_id = course.course_id AND
  enroll.yrs = course.yrs AND
  student.student_id = enroll.student_id
group by student.student_id
order by no_of_courses desc limit(10);
```

Output

```
"student_name";"email";"no"
"pedapalii akhil";"akhilp12@gmail.com";1
"jaya kishan kumar";"kishankumar12@gmail.com";1
"anjali kumari";"kumarianjali12@gmail.com";1
"anand rahul";"rahulanand12@gmail.com";1
"manoj kumar";"kumarm12@gmail.com";1
"aniket raj";"aniket12@gmail.com";1
"monika maheshwari";"monika12@gmail.com";1
"sandeep kumar";"sandykumar12@gmail.com";2
"parul bindal";"parulbindal12@gmail.com";2
"kamal avashthi";"avashthi12@gmail.com";2
"ashish kumar unni";"ashish12@gmail.com";2
"charu chimpa";"charu12@gmail.com";2
"chahat jain";"chahatjain12@gmail.com";2
"avi aryan";"aviaryan12@gmail.com";2
```

```
"prem chand saini";"premchand12@gmail.com";2
"ankit kumar";"ankitkumar12@gmail.com";2
"rakesh bakolia";"rakeshb12@gmail.com";2
"sahil luthra";"sahilluthra12@gmail.com";2
"vishal dilip nagrale";"dilipn12@gmail.com";2
"pratik paliwal";"palwal12@gmail.com";2
"krishann unni";"kunni12@gmail.com";2
"karan kumar";"kk12@gmail.com";2
"vaibhav anand";"anandv12@gmail.com";2
"vikash singh";"vikashsingh12@gmail.com";2
"mukesh sahu";"mukesh12@gmail.com";2
"vineela chandra";"vineela12@gmail.com";3
"anugu bharath reddy";"anugubharath12@gmail.com";3
"harshit purohit";"harshit12@gmail.com";3
"chirag garg";"chiraggarg12@gmail.com";3
"prakash vajekar";"prakashv12@gmail.com";3
"sharad patel";"sharadpatel12@gmail.com";3
"goutam kumar";"goutamk12@gmail.com";3
"sourabh jain";"jainsourabh12@gmail.com";3
"mukesh kumar";"mukesh12@gmail.com";3
"venkata sairahul";"sairahul12@gmail.com";3
"dheeraj reddy";"dheerajreddy12@gmail.com";4
"manish singhla";"manishsinghla12@gmail.com";4
"ajmeera bhavik naik";"ajmeera12@gmail.com";4
"ayush lamba";"ayushlamba12@gmail.com";4
"aashish yadvally";"aashish12@gmail.com";4
"vishal vaishanv";"vvaisnav12@gmail.com";4
"shewata tiwari";"shweta12@gmail.com";5
"lalit kumar";"lalitkumar12@gmail.com";5
"lalit singh";"lalit12@gmail.com";5
"shresthi priya";"spriya12@gmail.com";7
```

5. List of those instructor who have failed maximum no of sutdent.

   SQL

```
 SELECT
  instructor.instructor_name,count(instructor.instructor_name)as failed
FROM
  coursera.instruct_evaluation,
  coursera.instructor
WHERE
  instructor.instructor_id = instruct_evaluation.instructor_id
  and instruct_evaluation.evaluation<3
  group by instructor.instructor_name
  order by failed desc
```

   Output

```
"instructor_name";"failed"
"james bond";4
"alka parikh";2
```

```
"sajay srivastva";2
"p j deitel";2
"m baron";1
```

6. Rank all the courses according to maximum no of passing student group by years.
   SQL

```
 SELECT
  course.coursename,
  course.yrs,
  count(final_eval) as num_of_registration
FROM
  coursera.final_eval,
  coursera.course
WHERE
  course.course_id = final_eval.course_id AND
  course.yrs = final_eval.yrs AND
  final_eval.final_evaluation < 13
group by course.yrs,course.coursename
order by course.yrs;
```

   Output

```
"coursename";"yrs";"num_of_registration"
"An Intro to Interactive Programming in Python";2013;4
"Cryptography I";2013;2
"Functional Programming Principles in C ";2013;1
"Introduction to Finance";2013;1
"Introduction to Databases";2014;2
"Introduction to Finance";2014;1
"An Intro to Interactive Programming in Python";2015;4
"Introduction to Finance";2015;1
"Model Thinking";2015;3
```

7. List all the assignments, instructor name for the assignment in which more than 50 percent student submitted wrong answers.
   SQL

```
SELECT user_id,course_id,yrs,sum(points)as sum2
From (
(SELECT
computer.user_id as user_id,
  course.course_id,
  course.yrs,

  sum(computer.evaluation)as points
FROM
  coursera.course,
  coursera.assignment,
  coursera.computer
WHERE
  course.course_id = assignment.course_id AND
  assignment.assignment_id = computer.assignment_id
GROUP BY computer.user_id, course.course_id,course.yrs
order by course.course_id,course.yrs
)
UNION

(SELECT
instruct_evaluation.student_id as user_id,
  course.course_id,
  course.yrs,

  sum(instruct_evaluation.evaluation)as points
FROM
  coursera.course,
  coursera.assignment,
  coursera.instruct_evaluation
WHERE
  course.course_id = assignment.course_id AND
  assignment.assignment_id = instruct_evaluation.assignment_id
GROUP BY instruct_evaluation.student_id, course.course_id,course.yrs
order by course.course_id,course.yrs
)
UNION
(SELECT
  peer.student_id1 as user_id,
  course.course_id,
  course.yrs,
  sum(peer.evaluation)as points
FROM
  coursera.peer,
  coursera.course,
  coursera.assignment
WHERE
  course.course_id = assignment.course_id AND
  assignment.assignment_id = peer.assignment_id
GROUP BY peer.student_id1, course.course_id,course.yrs
order by course.course_id,course.yrs
)

) as x
GROUP BY x.user_id,x.course_id,x.yrs
order by x.course_id,x.yrs;
```

Output

```
"topic";"instructor_name"
"Is synergetics multifaceted";"james bond"
"What is cyclic codes";"authur benjamin"
```

```
"what is model thinking";"james bond"
"What is investment";"salinee singh"
```

0.7 References

- www.google.com
- Fundamentals of Database Systems, 6th Edition by Ramez Elmasri and Shamkant B. Navathe

0.8 Project members

	Name - Dilip Puri ID - 201351014 Branch - Computer Science Institute - Indian Institute of Information Technology, Vadodara

Name - Vivek Kumar Singh ID - 201352015 Branch - Information Technology Institute - Indian Institute of Information Technology, Vadodara	

	Name - Chirag Panpalia ID - 201351001 Branch - Computer Science Institute - Indian Institute of Information Technology, Vadodara