QUESTION PAPER GENERATOR

A Database Project Report

Master of Science

In

Computer Science

Submitted

By

DEEPTI SHARMA(09)
SANDHYA(32)
SHIVANI TIWARY(36)

Under the Supervision of
ASHOK KUMAR YADAV

DEPARTMENT OF COMPUTER SCIENCE

DELHI UNIVERSITY

NEW DELHI-110007

Nov-2017

DEPARTMENT OF COMPUTER SCIENCE

DELHI UNIVERSITY

NEW DELHI-110007

Nov-2017

DECLARATION

This is to certify that the Project Report entitled

“QUESTION PAPER GENERATOR”

is being submitted to the Department of Computer Science, Delhi University, New Delhi in partial fulfillment of the requirement for the course “Database System” in **Master of Science** is a record of bonafide work carried out by us under the supervision of **Prof. ASHOK KUMAR YADAV.**

The matter embodied in the declaration has not been submitted in part or full to any university or institution for the award of any degree or diploma.

DEEPTI SHARMA(09)

SANDHYA(32)

SHIVANI TIWARY(36)

DEPARTMENT OF COMPUTER SCIENCE

DELHI UNIVERSITY

NEW DELHI-110007

CERTIFICATE

This is to certify that this Project Report entitled “Question Paper Generator” submitted by **Deepti Sharma , Sandhya , Shivani Tiwary** to the **Department of Computer Science, Delhi University, New Delhi** in partial fulfillment of the requirement for the course “Database System” in **Master of Science** is a project report carried out by them under the supervision of

Prof. ASHOK KUMAR YADAV.

To the best of my knowledge this work has not been submitted in part or full to any other University or Institution for the award of any degree or diploma.

Delhi University **Prof. Ashok Kumar Yadav**

New Delhi, India

ACKNOWLEDGMENT

It is our great pleasure to express our gratitude to all the people who have supported us greatly during this project.

First of all, We would like to give special and sincere thanks to our professor ,**Prof. Ashok Kumar Yadav,** who has provided us with the best of his knowledge and experience and has encouraged us to delve into such a project and learning activity. His untiring and diligent effort, methodical approach made it possible for us to complete this work on time.

We would like to extend my appreciation to Department of Computer Sciences, Delhi University, New Delhi for motivating us towards project development and providing us with all necessary resources to accomplish the project on time.

As a team , we all are thankful to each other , and our Seniors for guiding us and understanding our needs.

This acknowledgement is incomplete without mentioning our parents and god who have enabled us to reach this place, we are deeply obliged to all the people who helped us and took the patience and time to encouragement to face challenges.

Deepti Sharma(09)
Sandhya(32)
Shivani Tiwary(36)

ABSTRACT

Automatic Question Paper Generator Application allows instructor to generate question paper for testing/quizzing students and allows student to practice questions for preparation purpose. Everyday new concepts and ideas are released in the market; the students not only need to learn them but practice them as well. Our software provides a huge number of questions, curated for clarity, and solutions, curated for quality and correctness. On the whole, it is a very good platform for practice and preparation and especially has a lot to offer beginners, who need not just practice questions but require a system which keeps on increasing the level as well, if they perform well. Realising the fact that question paper creation is a tedious and time consuming task; our system has the functionality for the instructor to generate and print the question paper automatically on a click with required preference

LIST OF ABBREVIATIONS

DESG	Designation
UID	User Identification
Fname	First name
Sname	Second name
Lname	Last name
Cno	Contact number
C_sid	Course identification
Cname	Course name
L_uid	Learners uid
Corr_rep	Correction reported
Diff	Difficulty

SNO	Serial number
Corr_subm	Correction Submission
Gradeptr	Grade pointer
Spec	Specification
Qual	Qualifications
Revptr	Review pointer
Contr_to_ppr	Contribution to paper
Frwd_to_expert	Forward to expert

Table of Contents

PROBLEM STATEMENT

A QUES_PAPER_DESIGN Software Database keeps track of all the types of Users, Courses and Questions Available on the topic.

I. Users have a unique ID associated with them. The Database keeps track of their Login Name, First Name, Second Name, Password, Birth date, Sex, Address, and Contact Number.

II. The system has three possible types of user: Instructors, Learners (from any age group) and Admin.

III. Instructor has certain special attributes given as: Qualification, Specialization, Contribution to Question Paper, Review Pointer.

IV. While Learners have special properties like Degree, Grade Pointer, Attempts, Corrections Reported, and Level (defined in the system: beginner, expert).

V. Admin is a special type of privileged who can add questions to the questions table.

VI. Each and Every Instructor teaches many courses/subjects and each course can be taught by many instructors.

VII. Each Course has a special Id and Name. Each Course added in the system's database has at least one question of its own.

VIII. Questions are stored with the following properties saved: Unique serial number,, Question Statement, Answer, Four Options, Correct Submissions, Difficulty Level(Easy , Difficult, Medium).

IX. Each Question belongs to a particular course only.

X. A learner can learn as many courses as (s) he wants.

XI. Each "Beginner" learner is assigned an "expert" for help.

XII. A learner can mark questions and forward doubts to expert. Learners doubt certain questions. A question can be doubted by many Learners

FUNCTIONAL DEPENDENCIES

- USERNAME/UID OF A USER SHOULD UNIQUELY IDENTIFY OTHER PROPERTIES :

    ○ **UID -> FirstName, SecondName, LastName, LoginName, Password, BirthDate, Sex, Address, ContactNo.**

Explanation : *Each user has a special Unique Identification number just like SSN in US , to identify them distinctly. Hence all the attributes of a user can be determined by using the UID attribute of the Users' Entity .*

- LEARNERS' UID UNIQUELY IDENTIFY IT'S ATTRIBUTES

    ○ **UID -> Degree, Attempts , CorrectionReported**

Explanation : *Again Learners is a subclass of Users Entity and all the specific attributes can be determined uniquely by the UID of the Users(Learners).*

- ATTEMPTS PERFORMED DETERMINE THE LEVEL OF THE USER AND THEIR GRADE POINTER

    ○ **Attempts -> Level, GradePointer**

Explanation : *We have assumed that the number of attempts made by a learner can determine at what level he or she is (Beginner / Expert) and we can give them a Grade based on the number of Attempts they have made so far.*

- INSTRUCTORS' UID UNIQUELY IDENTIFY ITS ATTRIBUTES:

    ○ **UID -> Specialization, Qualification, ContributionToQP, ReviewPointer**

Explanation : *Instructors is a subclass of Users Entity and all the specific attributes can be determined uniquely by the UID of the Users(Instructors).*

● ADMINS’ UID UNIQUELY IDENTIFY ITS ATTRIBUTES

○ **UID -> Designation, Permissions**

Explanation : *Admin is a subclass of Users Entity and all the specific attributes can be determined uniquely by the UID of the Users(Admin).*

● SERIAL NUMBER UNIQUELY IDENTIFIES QUESTION , ITS ANS AND RELATED ATTRIBUTES

○ **Sno -> QuestionStatement, Answer, CorrectSubmission, Difficulty**

Explanation : *Each Question has a special Serial Number to identify it distinctly. Hence all the attributes of a Question can be determined by using the SNO attribute of the Questions’ Entity .*

● SERIAL NUMBER UNIQUELY IDENTIFIES OPTIONS OF EACH QUESTION

○ **Sno,Options -> Sno,Options**

Explanation : *This is a Trivial Dependency as a Serial Number of a particular question and Options can determine the Sno and Options .But it’s important to realise that we list them to realize that the pair of both together uniquely make a pair.*

● QUESTIONS AND NUMBER OF CORRECT SUBMISSION OF THAT QUESTION DETERMINE THE DIFFICULTY LEVEL FOR THAT QUESTION

○ **Sno, CorrectSubmission -> Difficulty**

Explanation : *The Serial Number of a question basically identifies uniquely the question in the database and The number of times a question has been correctly submitted determines the difficulty level. Correct Submission is a range indicator.*

● SUBJECTID UNIQUELY DETERMINES COURSE NAME

○ **C_SID -> Cname**

Explanation : *Each Course has a special Subject Id to identify it distinctly. Hence all the attributes of a Course can be determined by using the C_SID attribute of the Courses’ Entity .*

DETERMINING CANDIDATE KEY

From the Given Functional Dependency we derive the Candidate key for the various entities we have :

1. **USERS:**

$\{UID\}^+$ = { UID,FirstName, SecondName, LastName, Password, BirthDate, Sex, Address, ContactNo. }

$\{FIRSTNAME\}^+$= {FirstName}
$\{SECONDNAME\}^+$ = {SecondName}
$\{LASTNAME\}^+$ = {LastName}
$\{PASSWORD\}^+$ = {Password}
$\{BIRTHDATE\}^+$ = {Birthtdate}
$\{SEX\}^+$ = {Sex}
$\{ADDRESS\}^+$= {Address}
$\{CONTACTNO.\}^+$ ={ContactNo.}

$\{FIRSTNAME,SECONDNAME\}^+$ = {FirstName, SecondName}
$\{FIRSTNAME,LASTNAME\}^+$ = {FirstName,LastName}
$\{FIRSTNAME,PASSWORD\}^+$ = {FirstName,Password}
$\{FIRSTNAME,BIRTHDATE\}^+$ = {FirstName,BirthDate}
$\{FIRSTNAME,SEX\}^+$ = {FirstName,Sex}
$\{FIRSTNAME,ADDRESS\}^+$ = {FirstName,Address}
$\{FIRSTNAME,CONTACTNO\}^+$ = {FirstName,ContactNo.}
$\{SECONDNAME,LASTNAME\}^+$ = {SecondName,LastName}
$\{SECONDNAME,PASSWORD\}^+$ = {SecondName,Password}
:
:
:
:

Upon making all possible combinations that there are only two keys possible for the Users' entity

1) Super Key : {UID,FIRSTNAME, SECONDNAME, LASTNAME, LOGINNAME, PASSWORD, BIRTHDATE, SEX, ADDRESS, CONTACTNO.}
2) Candidate Key : {UID}

2. INSTRUCTORS:

$\{UID\}^{+}$={UID,SPECIALIZATION, QUALIFICATION, CONTRIBUTIONTOQP, REVIEWPOINTER }

Again ,Upon making all possible combinations that there are only two keys possible for the Instructors' entity

1) Super Key : {UID, SPECIALIZATION, QUALIFICATION, CONTRIBUTIONTOQP, REVIEWPOINTER}
2) Candidate Key : {UID}

3. ADMIN :

$\{UID\}^{+}$={UID, DESIGNATION, PERMISSIONS}

Upon making all possible combinations that there are only two keys possible for the Admin's Entity :

1) Super Key : {UID, DESIGNATION, PERMISSIONS}
2) Candidate Key : {UID}

4. LEARNERS :

$\{UID\}^{+}$={UID,DEGREE, ATTEMPTS , CORRECTIONREPORTED, LEVEL, GRADEPOINTER}

Upon making all possible combinations that there are only two keys possible for the Admin's Entity :

1) Super Key : {UID,DEGREE, ATTEMPTS , CORRECTIONREPORTED, LEVEL, GRADEPOINTER}
2) Candidate Key : {UID}

5. COURSE :

$\{C\_SID\}^{+}$ = {C_SID, CNAME}

Upon making all possible combinations that there are only two keys possible for the Admin's Entity :

1) Super Key : {C_SID, CNAME}
2) Candidate Key : {C_SID}

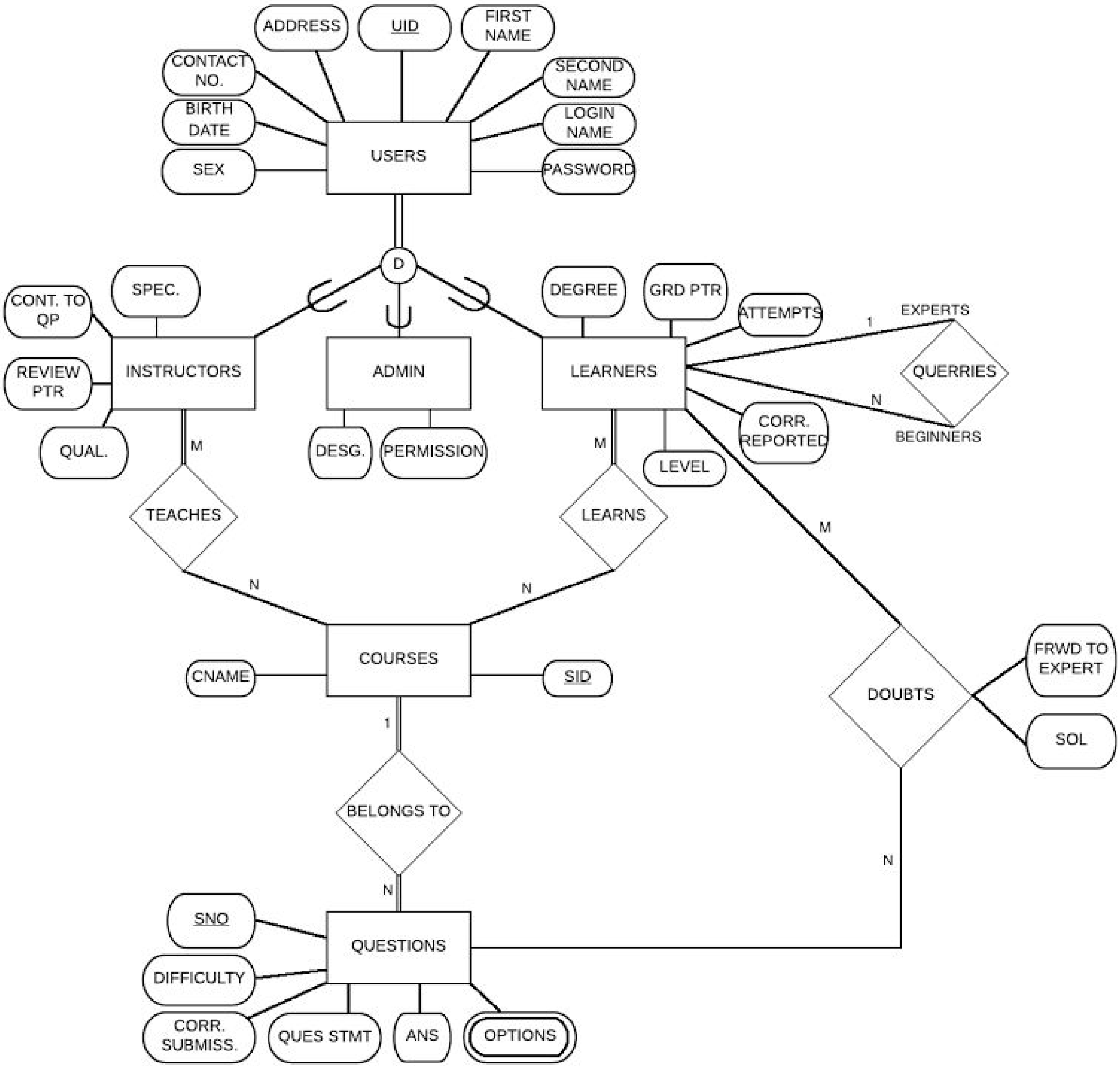
6. QUESTIONS :

$\{SNO\}^+$= { SNO, QUESTIONSTATEMENT, ANSWER, CORRECTSUBMISSION, DIFFICULTY }

Upon making all possible combinations that there are only two keys possible for the Admin's Entity :

1) Super Key : {SNO, QUESTIONSTATEMENT, ANSWER, CORRECTSUBMISSION, DIFFICULTY }
2) Candidate Key : {SNO}

ER DIAGRAM



ADDRESS
UID
FIRST NAME
CONTACT NO.
SECOND NAME
BIRTH DATE
LOGIN NAME
SEX
USERS
PASSWORD
D
SPEC.
CONT. TO QP
DEGREE
GRD PTR
ATTEMPTS
1
EXPERTS
INSTRUCTORS
ADMIN
LEARNERS
QUERRIES
REVIEW PTR
N
CORR. REPORTED
BEGINNERS
QUAL.
M
DESG.
PERMISSION
M
LEVEL
TEACHES
LEARNS
M
N
N
COURSES
CNAME
SID
FRWD TO EXPERT
DOUBTS
1
SOL
BELONGS TO
N
N
SNO
QUESTIONS
DIFFICULTY
CORR. SUBMISS.
QUES STMT
ANS
OPTIONS

ER DIAGRAM EXPLANATION :

This database stores information about USERS, COURSES and QUESTIONS.
Basic and Required Information of Users of the system are captured in Entity USERS.
There are three types of users :

1) Instructor
2) Admin
3) Learner

Questions Entity is used to capture all the information required for a question and Courses entity is used to capture the data of courses learnt and taught by the various users. There are attributes according to the problem statement we have specified earlier.
Also there are relations to reflect how various entities interact :

1. Instructor teaches Courses.
2. Learner learns Courses
3. Learner queries Expert Learner
4. Leaner raise doubts on Questions
5. Questions belong to Courses.

NOTATIONS:

The Figure displays the QUESTION_PAPER_GENERATOR database schema as an ER diagram.
ER diagram notation :

1. Entity types such as USER, INSTRUCTOR, ADMIN, LEARNER, COURSES and QUESTIONS are shown in rectangular boxes.
2. Relationship types such as TEACHES, LEARNS, DOUBTS, BELONGS_TO and QUERIES are shown in diamond-shaped boxes attached to the participating entity types with straight lines.
3. Key attributes have their names underlined.
4. The cardinality ratio of each binary relationship type is specified by attaching a 1, M, or N on each participating edge.

ASSUMPTIONS:

1. The cardinality ratio :
    a. In relation TEACHES of INSTRUCTORS:COURSES is M:N . We have assumed that an instructor can teach many courses and a course can be taught by many Instructor.
    b. In relation LEARNS of LEARNERS:COURSE is again M:N, Assuming that a student can learn as many courses as much he/she want to and a course can be opted / practiced by many students.
    c. In relation BELONGS_TO of COURSES:QUESTION is 1:N , under the assumption that a course can have many questions but a question can belong to only one specific course.
    d. A relation DOUBTS is present between LEARNERS:QUESTION is a special one with ratio
    e. The recurrence relation QUERIES between LEARNERS:LEARNERS is 1:N with Role Names , EXPERT:BEGINNER.We have assumed that each Beginner has been associated with an Expert to query upon having a doubt, while an expert can be queried by many learners associated with him/her.
2. The Participation Type :
    a. TEACHES has a TOTAL participation on Instructor side and PARTIAL participation on Courses side , that is ALL Instructor in the system must teach some or the other course , but a course needn't be taught by any of the Instructor , because the course is introduced newly in the system.
    b. LEARNS relation has TOTAL participation on Learner side and PARTIAL participation on Courses side , that is ALL learners must be learning some of the courses in the system while it is not necessary that a course is practiced by any of the learners ,again under the assumption that the course is new.
    c. BELONGS_TO has TOTAL participation on both side; COURSES:QUESTIONS ; implying that a All courses have question associated with them and All question must belong to some course.
    d. DOUBTS has PARTIAL participation on both side LEARNERS:QUESTIONS; because a learner can doubt many question but it's not necessary that all questions are doubted by some learner; also it is not necessary that all Learners are doubting some question.
    e. QUERIES has PARTIAL participation , which is obvious because a learner can be either an expert or beginner. We want beginners to be able to send their doubts to their respective expert while expert can directly mark doubt to send them to instructor of teaching the course (to which the question belongs).
3. Further we assume that the superclass USERS have three distinct /non overlapping classes : INSTRUCTORS, ADMINS, LEARNERS, and we have also assumed a TOTAL PARTICIPATION of all the entities in USERS.

MAPPING

USER TABLE:

We are using the mapping suitable for disjoint and total participation. We are mapping the user entity's attribute into USER table and we will create table for each of the other sub classes and add the key of Super Class into all the subclasses.

Primary key : UID

USER TABLE								
UID	FNAME	SNAME	LNAME	PASS WORD	BIRTH DATE	SEX	CNO.	ADDRESS

INSTRUCTOR TABLE :

Primary key : UID

INSTRUCTOR TABLE				
UID	SPECIALIZATION	QUALIFICATION	CONTRIBUTION TO QP	REVIEW POINTER

ADMIN TABLE :

Primary key : UID

ADMIN TABLE		
UID	DESIGNATION	PERMISSION

LEARNERS TABLE :

Since the Learners' entity is involved in the 1:N recursive relation : Queries , we have included the Learner's UID (CK) as Expert UID(FK).

Primary key : UID

Foreign Key : EXPERT_UID (REFERENCES LEARNER'S UID)

LEARNERS TABLE

UID	GRADE POINTER	DEGREE	ATTEMPTS	CORRECTION REPORTED	LEVEL	EXPERT UID

COURSES TABLE :

The Courses Entity is mapped directly to the course table.

Primary key : C_SID

COURSES TABLE	
C_SID	CNAME

QUESTIONS TABLE :

The Entity Question directly maps to the table Question and since the Options attribute is a multivalue attribute , it appears as a separate table in the database.

Further, Questions table is involved in the 1:N relation : BELONGS _TO , therefore the CK of the 1 side ; i.e. Courses appears as FK on the N side.

Primary key : SNO

Foreign Key : C_SID

QUESTIONS TABLE					
SNO	QUESTION STATEMENT	ANSWER	DIFFICULTY	CORRECTION SUBMISSION	C_SID

QUESTION_OPTIONS TABLE:

Primary key : SNO

QUESTION_OPTION TABLE	
SNO	OPTIONS

MAPPING THE M:N RELATION INTO TABLES :

TEACHES TABLE :

Primary Key : (I_uid,C_sid)

Foreign Key : I_uid (References Instructors' Uid) And C_sid (References Courses' C_sid)

TEACHES TABLE	
I_UID	C_SID

LEARNS TABLE :

Primary Key :(L_uid,C_sid)

Foreign Key : L_uid(References Learners' Uid) And C_sid (References Courses' C_sid)

LEARNS TABLE	
L_UID	C_SID

DOUBTS TABLE:

Primary Key :(L_uid,_sid)

Foreign Key :L_uid(References Learner's Uid) And C_sid (References Courses' C_sid)

DOUBTS TABLE			
L_UID	SNO	FORWARD TO EXPERT	SOLUTION

NORMALISATION

Using the functional dependencies we identified , We can normalize our tables upto BCNF .

TABLE NAME	**1 NF**	**2 NF**	**3 NF**	**BCNF**
USER	YES	YES	YES	YES
INSTRUCTOR	YES	YES	YES	YES
ADMIN	YES	YES	YES	YES
LEARNER	**YES**	**NO**	**-**	**-**
COURSES	YES	YES	YES	YES
QUESTIONS	**YES**	**NO**	**-**	**-**
QUESTION OPTIONS	YES	YES	YES	YES
TEACHES	YES	YES	YES	YES
LEARNS	YES	YES	YES	YES
DOUBTS	YES	YES	YES	YES

1. **LEARNERS TABLE :**

   We were given :

   - *UID -> Attempts , Degree, CorrectionReported*
   - *Attempts -> Level, GradePointer*

   So the decomposition we get are:

   Learners (UID, Degree, Attempts, CorrectionReported)

   Attempts (Attempts, GradePointer, Level)

Checking whether this decomposition is :

1) LOSSLESS
2) ATTRIBUTE PRESERVING
3) DEPENDENCY PRESERVING

1) LOSSLESS:

Since the relation Learners and Attempts have attempts as a commonattribute and it serve as a candidate key for the attempts table , the decomposition is LOSSLESS.

{ATTEMPTS}$^+$ = {ATTEMPTS, GRADEPOINTER,LEVEL}

2) ATTRIBUTE PRESERVING :

Yess , this decomposition is attribute preserving as the union of the two newly formed relation give all the attributes of the original table.

3) DEPENDENCY PRESERVING :

Yess this decomposition is dependency preserving as both the dependency :

- *UID -> Attempts , Degree, CorrectionReported*
- *Attempts -> Level, GradePointer*

Still hold in the new relations.Hence , the two new relations Learners and Attempts are in BCNF.

LEARNERS:

Primary Key : Uid

Foreign Key : Uid(References User's Uid) And Attempts (References Attempts' Attempts)

LEARNERS TABLE:			
UID	DEGREE	ATTEMPTS	CORRECTION REPORTED

ATTEMPTS :

Primary Key : Attempts

ATTEMPTS TABLE		
ATTEMPTS	GRADE POINTER	LEVEL

2. QUESTION TABLE :

We have :

- *Sno, CorrectSubmission -> Difficulty*
- *Sno -> QuestionStatement, Answer, CorrectSubmission*

So the decomposition we get is :

Questions(Sno, Ques_Stmt, Answer)

Ques_2(Sno, Corr_subm, Difficulty)

Checking whether this decomposition is :

4) LOSSLESS
5) ATTRIBUTE PRESERVING
6) DEPENDENCY PRESERVING

1) LOSSLESS:

Since the relation Questions and Ques_2 have Sno as a common attribute and it serve as a candidate key for both the tables , the decomposition is hence LOSSLESS.

$\{SNO\}^{+}$ = {SNO,QUES_STMT,ANSWER}

$\{SNO\}^{+}$={SNO, CORR_SUBM,DIFF}

2) **ATTRIBUTE PRESERVING :**

Yess , this decomposition is attribute preserving as the union of the two newly formed relation give all the attributes of the original table.

3) DEPENDENCY PRESERVING :

Yess this decomposition is dependency preserving as both the dependency , Still hold in the new relations.

- *Sno, CorrectSubmission -> Difficulty*
- *Sno -> QuestionStatement, Answer, CorrectSubmission*

Hence , the two new relations Questions and Ques_2 are in BCNF.

QUESTIONS:

Primary Key : Sno

QUESTIONS TABLE		
SNO	QUESTION STATEMENT	ANSWER

QUES_2 TABLE:

Primary Key : (Sno,Corr_subm)

QUES_2 TABLE		
SNO	CORR_SUBM	DIFFICULTY

Hence Now all the tables we get after normalization are in BCNF.

TABLE NAME	**1 NF**	**2 NF**	**3 NF**	**BCNF**
USER	YES	YES	YES	YES
INSTRUCTOR	YES	YES	YES	YES
ADMIN	YES	YES	YES	YES
LEARNER	YES	YES	YES	YES
ATTEMPTS	YES	YES	YES	YES
COURSES	YES	YES	YES	YES
QUESTIONS	YES	YES	YES	YES
QUES_2	YES	YES	YES	YES

QUESTION OPTIONS	YES	YES	YES	YES
TEACHES	YES	YES	YES	YES
LEARNS	YES	YES	YES	YES
DOUBTS	YES	YES	YES	YES

TABLE CREATION

USER TABLE :

```
mysql> create table user
(
uid varchar(50) not null primary key,
fname varchar(25)not null,
sname varchar(25) ,
lname varchar(25) not null,
 password varchar(25) not null,
sex varchar(1) not null,
 cno int(10),
 address varchar(25),
birth_date date
);
```

INSTRUCTORS TABLE:

```
mysql> create table instructors
(uid varchar(50) not null,
 foreign key(uid) references user(uid) on delete cascade on update cascade,
spec varchar(25),
qual varchar(25) not null ,
contr_to_ptr int(5) not null,
revptr  int(5) not null
Primary key (uid)
);
```

ADMIN TABLE :

```
mysql> create table admin
(
uid varchar(50) not null,
foreign key(uid) references user(uid) on delete cascade on update cascade,
desg varchar(25),
permission varchar(10) not null
Primary key (uid)
);
```

LEARNER TABLE:

```
mysql> create table learners
(
uid varchar(50) not null,
 foreign key(uid) references user(uid) on delete cascade on update cascade,
gradeptr varchar(1) ,
corr_rep int(5),
degree varchar(50),
attempts int(5) ,
level varchar(10),
expert_uid varchar(50) ,
Primary key (uid)
);
```

//Updated table to make expert_uid refer the learner’s uid

COURSES TABLE :

```
mysql> create table courses
(
c_sid  varchar(50) not null primary key,
cname varchar(50) not null
);
```

QUESTIONS TABLE :

```
mysql> create table questions
(
sno varchar(50) not null primary key,
c_sid varchar(50) foreign key(c_sid) references courses(c_sid) on delete cascade on update cascade,
ques_stmt varchar(1000) not null,
ans int(5) not null,
difficulty varchar(50) not null,
corr_subm int(5)
);
```

QUES_2 TABLE :

```
mysql> create table ques_2
(
sno varchar(50),
corr_subm int(5),
diff int(2),
primary key(sno,corr_subm),
foreign key(sno) references questions(sno)
);
```

ATTEMPTS TABLE :

```
mysql> create table attempts
(
attempts int(5) not null primary key,
Gradeptr varchar(1) not null,
Level int(2) not null,
);
```

QUESTION_OPTIONS TABLE:

```
mysql> create table question_options
(
sno varchar(50) not null,
 foreign key(sno)references questions(sno) on delete cascade on update cascade,
options varchar(50) not null
);
Query OK, 0 rows affected (0.73 sec)
```

TEACHES TABLE :

```
mysql> create table teaches
(i_uid varchar(50),
c_sid varchar(50) ,
primary key(i_uid,c_sid)
foreign key(i_uid) references instructors(uid),
Foreign key (c_sid) references courses(c_sid)
);
```

LEARNS TABLE :

```
mysql> create table learns
(
l_uid varchar(50),
c_sid varchar(50) ,
primary key(l_uid,c_sid)
foreign key(l_uid) references learners(uid),
Foreign key (c_sid) references courses(c_sid)
);
```

DOUBTS TABLE:

```
create table doubts(
l_uid varchar(50),
sno varchar(50),
frwd_to_expert varchar(3),
solution varchar(1000),
primary key(l_uid,sno),
foreign key(l_uid) references user(uid),
foreign key(sno) references questions(sno)
);
```

VALUE INSERTION

USER TABLE :

```
mysql> insert into user

values('1','Ritika’','Chopra',’Wadhwan’,'ritz','F','978654321','Delhi’, ‘1985-08-17’);

Query OK, 1 row affected (0.05 sec)

mysql> insert into user(uid, fname,lname,password,sex,cno,address)
values('2','Urmil','Bharti','deepti','F','987654321','Bihar');

Query OK, 1 row affected (0.05 sec)


mysql> insert into user(uid, fname,lname,password,sex,cno,address)

 values('3','Deepali','Null','Bajaj','deeps','F','890765786','Delhi','8');

Query OK, 1 row affected (0.06 sec)


mysql> insert into user(uid,fname,lname,password,sex,cno,address,birth_date)

values('4','Tulika','Singh','lika','F',' 986532741  ','UP','1990-6-13');

Query OK, 1 row affected (0.41 sec)


mysql> insert into user(uid,fname,lname,password,sex,cno,address,birth_date)

 values('5','Sumit','Kumar','sum','M','876541286','Noida','1992-8-11');

Query OK, 1 row affected (0.09 sec)


mysql> insert into user(uid,fname,lname,password,sex,cno,address,birth_date)

 values('6','Sandhya','Raghav','sunshine','F','920526051','Haryana','1996-03-10');

Query OK, 1 row affected (0.06 sec)
```

```
mysql> insert into user(uid,fname,lname,password,sex,cno,address,birth_date)
values('7','Shivani','Tiwary','shivi','F','897626051','Bihar','1996-07-29’');

Query OK, 1 row affected (0.06 sec)


mysql> insert into user(uid,fname,lname,password,sex,cno,address,birth_date)

values('8','Deepti','Sharma','dia','F','998765761','Delhi','1996-05-09');

Query OK, 1 row affected (0.08 sec)
```

INSTRUCTOR TABLE :

```
mysql> insert into instructors values(‘3’,’Database’,’PHD’,’7’,’5’);

mysql> insert into instructors values(‘4’,’AI’,’M.tech’,’7’,’5’);

mysql> insert into instructors values(‘5’,’Networks’,’M.tech’,’5’,’3’);
```

ADMIN TABLE :

```
mysql>insert into admin values(1,Head,110);

mysql>insert into admin values(2,Head,111);

mysql>insert into admin values(1,Head,110);

mysql>insert into admin values(2,Head,111);
```

LEARNER TABLE :

```
mysql> insert into learners values('6','3','Bsc',200,'8');

Query OK, 1 row affected (0.42 sec)


mysql> insert into learners values('7','6','Bsc',100,'6');

Query OK, 1 row affected (0.05 sec)


mysql> insert into learners values('8','10','Btech',500,'8');
```

```
Query OK, 1 row affected (0.38 sec)
```

ATTEMPTS :

```
mysql> insert into attempts values(100,"E",0);
Query OK, 1 row affected (0.38 sec)


mysql> insert into attempts values(200,"D",1);
Query OK, 1 row affected (0.36 sec)


mysql> insert into attempts values(300,"C",2);
Query OK, 1 row affected (0.34 sec)


mysql> insert into attempts values(400,"B",3);
Query OK, 1 row affected (0.09 sec)


mysql> insert into attempts values(500,"A",4);
Query OK, 1 row affected (0.08 sec)
```

COURSES :

```
mysql> insert into courses values(“C_101”,’’Algorithms”);
mysql> insert into courses values(“C_104”,”Artificial Intelligence”);
Mysql> insert into courses values(“C_105”,”Computational Intelligence”);
```

QUESTIONS :

```
mysql> insert into questions(sno,c_sid,ques_stmt,ans,difficulty)
  values('C_101_1','C_101','The operation of processing each element in the list is known
as','4','0');
```

```
mysql>  insert into questions(sno,c_sid,ques_stmt,ans,difficulty)
 values('C_104_1','C_104','What is the rule of simple reflex agent?','2','1');
Query OK, 1 row affected (0.06 sec)
```

```
mysql>  insert into questions(sno,c_sid,ques_stmt,ans,difficulty)
values('C_104_2','C_104','In which of the following agent does the poblem generator is
present?','1','4');
Query OK, 1 row affected (0.07 sec)
```

```
mysql>  insert into questions(sno,c_sid,ques_stmt,ans,difficulty)
values('C_105_1','C_105',' Which of the following is not the promise
 of artificial neural network?','1','5');
Query OK, 1 row affected (0.09 sec)
```

```
mysql>  insert into questions(sno,c_sid,ques_stmt,ans,difficulty)
 values('C_105_2','C_105','Which of the following is an application of NN (Neural
Network)?','4','6');
Query OK, 1 row affected (0.05 sec)
```

QUES_2:

```
mysql> insert into ques_2 values("C_101_1",49,9);
Query OK, 1 row affected (0.11 sec)


mysql> insert into ques_2 values( "C_101_2",479,1);
Query OK, 1 row affected (0.06 sec)


mysql> insert into ques_2 values("C_104_1",300,5);
Query OK, 1 row affected (0.17 sec)


mysql> insert into ques_2 values("C_104_2",995,0);
Query OK, 1 row affected (0.05 sec)


mysql> insert into ques_2 values("C_105_1",100,8);
Query OK, 1 row affected (0.05 sec)


mysql> insert into ques_2 values("C_105_2",105,7);
Query OK, 1 row affected (0.08 sec)
```

QUESTION_OPTIONS :

```
mysql> insert into question_options values("C_104_2","Learning agent");
Query OK, 1 row affected (0.08 sec)


mysql> insert into question_options values("C_104_2","Observing agent");
```

```
Query OK, 1 row affected (0.09 sec)

mysql> insert into question_options values("C_104_2","Reflex agent");
Query OK, 1 row affected (0.08 sec)

mysql> insert into question_options values("C_104_2","None of the mentioned");
Query OK, 1 row affected (0.08 sec)

mysql> insert into question_options values("C_101_2","Time only");
Query OK, 1 row affected (0.08 sec)

mysql> insert into question_options values("C_101_2","Space only");
Query OK, 1 row affected (0.09 sec)

mysql> insert into question_options values("C_101_2","Both Time and Space");
Query OK, 1 row affected (0.09 sec)

mysql> insert into question_options values("C_101_2","None of the mentioned");
Query OK, 1 row affected (0.36 sec)

mysql> insert into question_options values("C_104_2","It can explain result");
Query OK, 1 row affected (0.39 sec)

mysql> insert into question_options values("C_104_2","It can survive the failure of some
nodes");
Query OK, 1 row affected (0.05 sec)
```

```
mysql> insert into question_options values("C_104_2","It can inherenr parallelism");
Query OK, 1 row affected (0.08 sec)

mysql> insert into question_options values("C_104_2","It can handle noise");
Query OK, 1 row affected (0.38 sec)
```

TEACHES :

```
mysql> insert into teaches values("3","C_101");
Query OK, 1 row affected (0.08 sec)

mysql> insert into teaches values("3","C_104");
Query OK, 1 row affected (0.06 sec)

mysql> insert into teaches values("3","C_105");
Query OK, 1 row affected (0.05 sec)

mysql> insert into teaches values("3","C_101");
Query OK, 1 row affected (0.08 sec)

mysql> insert into teaches values("3","C_104");
Query OK, 1 row affected (0.06 sec)

mysql> insert into teaches values("3","C_105");
Query OK, 1 row affected (0.05 sec)
```

```
mysql> insert into teaches values("4","C_104");

Query OK, 1 row affected (0.06 sec)


mysql> insert into teaches values("5","C_105");

Query OK, 1 row affected (0.06 sec)


mysql> insert into teaches values("5","C_101");

Query OK, 1 row affected (0.08 sec)
```

LEARNS :

```
mysql> insert into learns values("6","C_104");

Query OK, 1 row affected (0.38 sec)


mysql> insert into learns values("6","C_105");

Query OK, 1 row affected (0.06 sec)


mysql> insert into learns values("7","C_101");

Query OK, 1 row affected (0.06 sec)


mysql> insert into learns values("7","C_105");

Query OK, 1 row affected (0.05 sec)


mysql> insert into learns values("8","C_101");

Query OK, 1 row affected (0.08 sec)


mysql> insert into learns values("8","C_104");
```

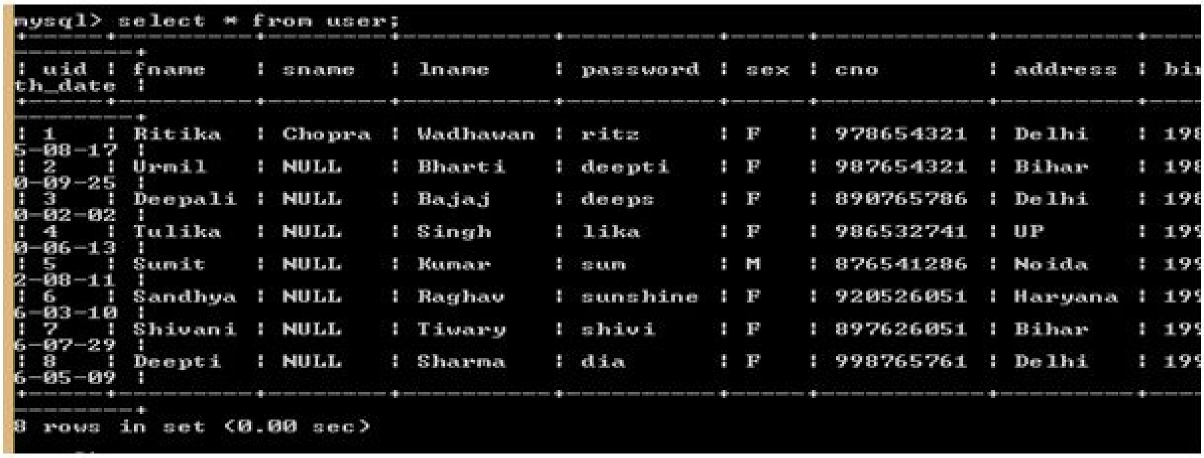
```
Query OK, 1 row affected (0.08 sec)


mysql> insert into learns values("8","C_105");

Query OK, 1 row affected (0.06 sec)
```

DOUBTS :

```
mysql> insert into doubts values("6","C_104_2","yes","no solution");

Query OK, 1 row affected (0.42 sec)


mysql> insert into doubts values("8","C_101_1","no","List is a data structure.");

Query OK, 1 row affected (0.11 sec)
```

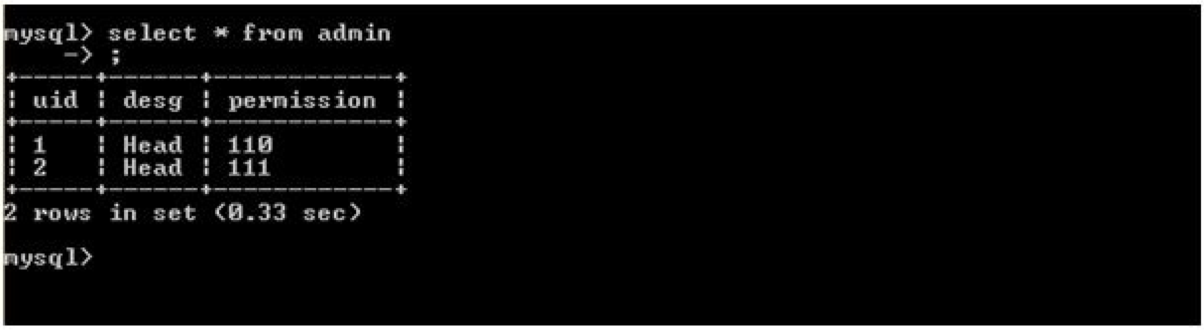
DATABASE SNAPSHOT

1. USER TABLE :



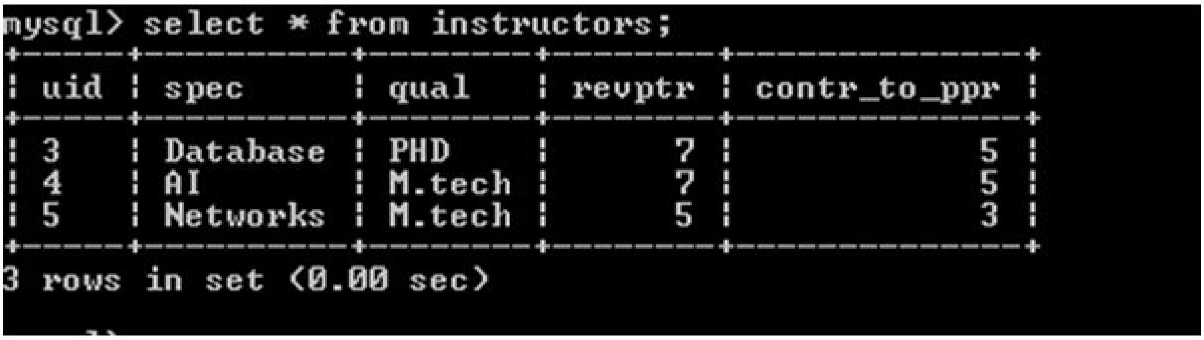
```
mysql> select * from user;
+-----+---------+--------+----------+----------+-----+------------+----------+----
---------+
| uid | fname   | sname  | lname    | password | sex | cno        | address  | bi
th_date |
+-----+---------+--------+----------+----------+-----+------------+----------+----
---------+
| 1   | Ritika  | Chopra | Wadhawan | ritz     | F   | 978654321  | Delhi    | 198
5-08-17 |
| 2   | Urmil   | NULL   | Bharti   | deepti   | F   | 987654321  | Bihar    | 198
0-09-25 |
| 3   | Deepali | NULL   | Bajaj    | deeps    | F   | 890765786  | Delhi    | 198
0-02-02 |
| 4   | Tulika  | NULL   | Singh    | lika     | F   | 986532741  | UP       | 199
0-06-13 |
| 5   | Sumit   | NULL   | Kumar    | sum      | M   | 876541286  | Noida    | 199
2-08-11 |
| 6   | Sandhya | NULL   | Raghav   | sunshine | F   | 920526051  | Haryana  | 199
6-03-10 |
| 7   | Shivani | NULL   | Tiwary   | shivi    | F   | 897626051  | Bihar    | 199
6-07-29 |
| 8   | Deepti  | NULL   | Sharma   | dia      | F   | 998765761  | Delhi    | 199
6-05-09 |
+-----+---------+--------+----------+----------+-----+------------+----------+----
---------+
8 rows in set (0.00 sec)
```

2. ADMIN TABLE:



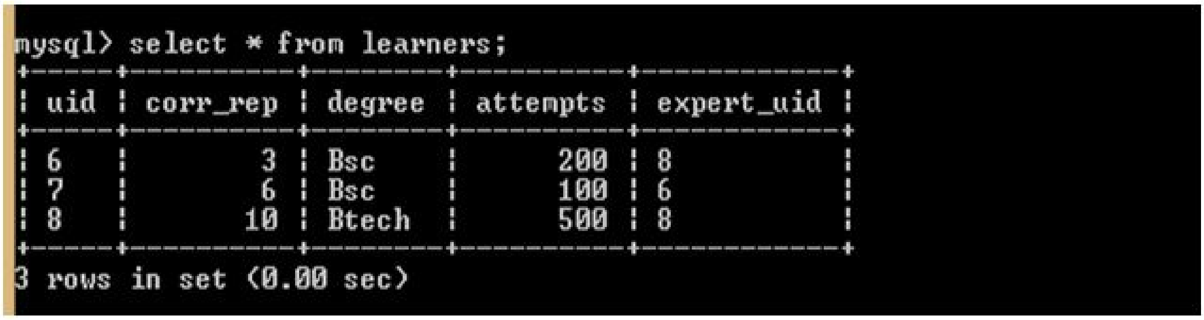
```
mysql> select * from admin
    -> ;
+-----+------+------------+
| uid | desg | permission |
+-----+------+------------+
| 1   | Head | 110        |
| 2   | Head | 111        |
+-----+------+------------+
2 rows in set (0.33 sec)

mysql>
```

3. INSTRUCTORS TABLE :



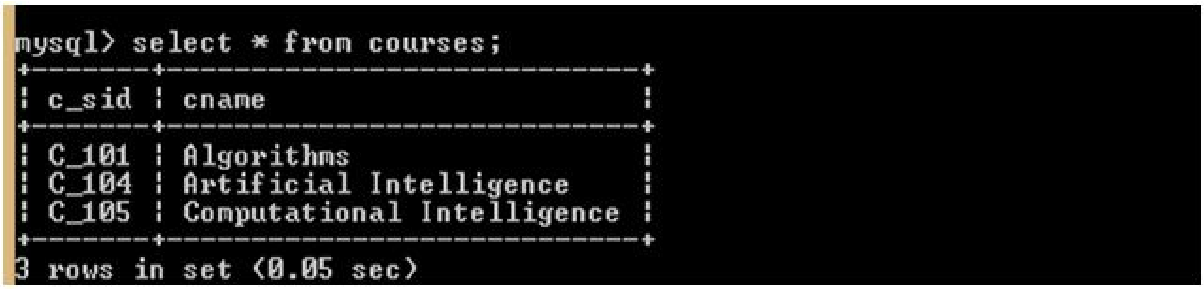
```
mysql> select * from instructors;
+-----+----------+--------+--------+-------------+
| uid | spec     | qual   | revptr | contr_to_ppr |
+-----+----------+--------+--------+-------------+
| 3   | Database | PHD    |      7 |           5 |
| 4   | AI       | M.tech |      7 |           5 |
| 5   | Networks | M.tech |      5 |           3 |
+-----+----------+--------+--------+-------------+
3 rows in set (0.00 sec)
```

4. LEARNER TABLE :



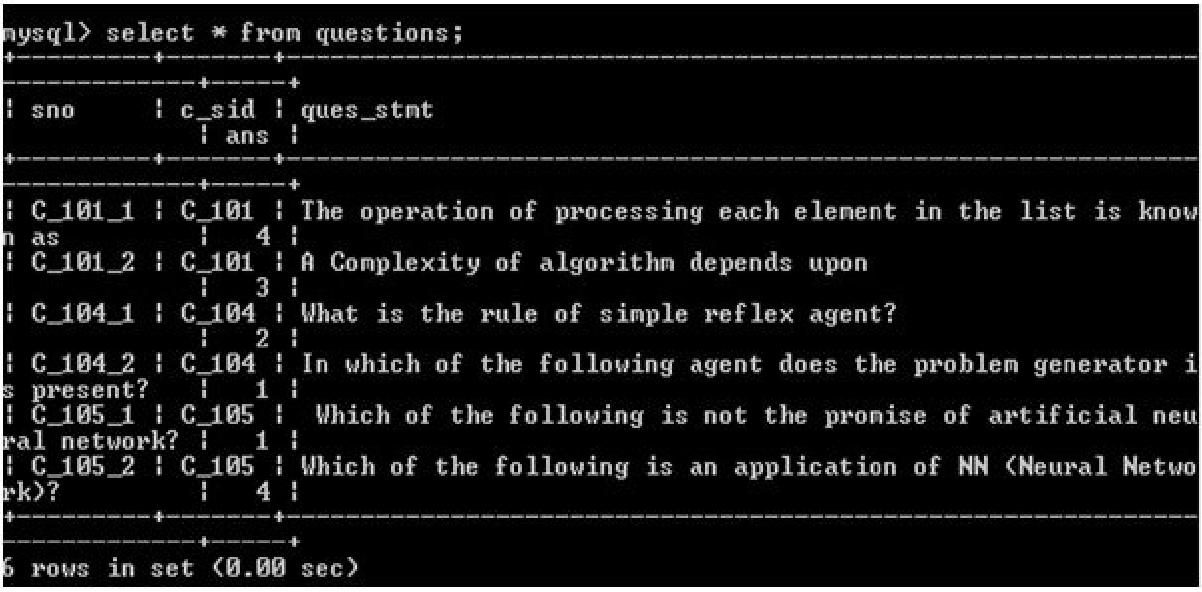
```
mysql> select * from learners;
+-----+----------+--------+----------+------------+
¦ uid ¦ corr_rep ¦ degree ¦ attempts ¦ expert_uid ¦
+-----+----------+--------+----------+------------+
¦ 6   ¦        3 ¦ Bsc    ¦      200 ¦ 8          ¦
¦ 7   ¦        6 ¦ Bsc    ¦      100 ¦ 6          ¦
¦ 8   ¦       10 ¦ Btech  ¦      500 ¦ 8          ¦
+-----+----------+--------+----------+------------+
3 rows in set (0.00 sec)
```

5. COURSES :



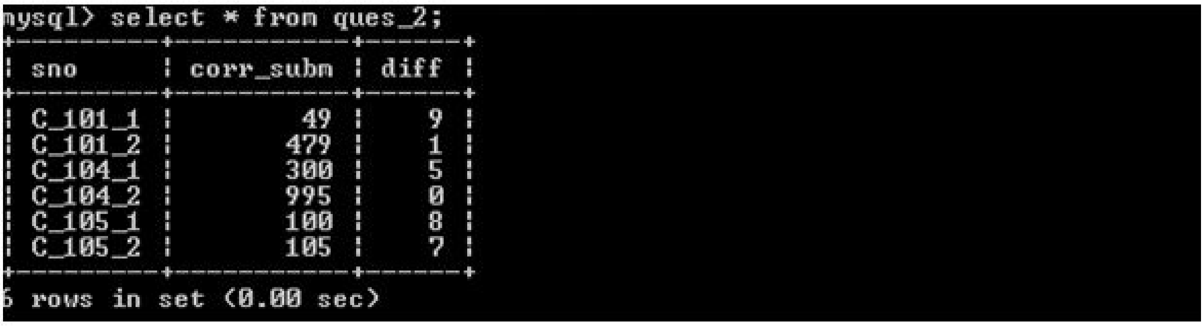
```
mysql> select * from courses;
+-------+-----------------------------+
¦ c_sid ¦ cname                       ¦
+-------+-----------------------------+
¦ C_101 ¦ Algorithms                  ¦
¦ C_104 ¦ Artificial Intelligence     ¦
¦ C_105 ¦ Computational Intelligence ¦
+-------+-----------------------------+
3 rows in set (0.05 sec)
```

6. QUESTIONS



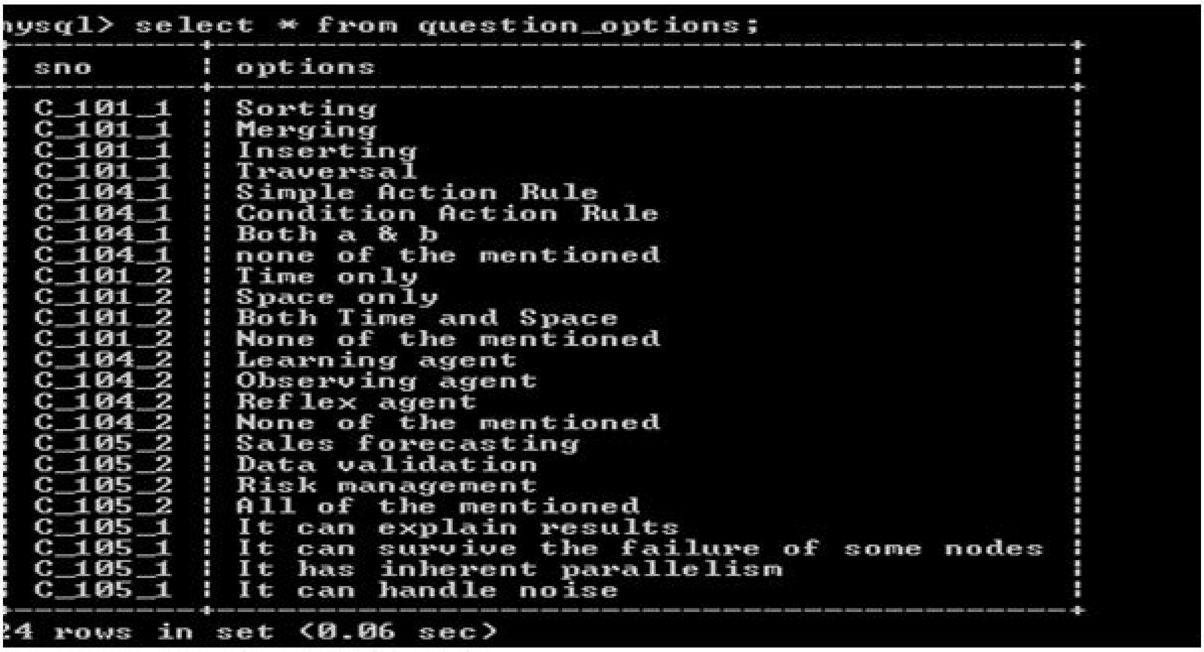
```
mysql> select * from questions;
+---------+-------+------------------------------------------------------------
-------------+-----+
¦ sno     ¦ c_sid ¦ ques_stmt
            ¦ ans ¦
+---------+-------+------------------------------------------------------------
-------------+-----+
¦ C_101_1 ¦ C_101 ¦ The operation of processing each element in the list is know
n as        ¦   4 ¦
¦ C_101_2 ¦ C_101 ¦ A Complexity of algorithm depends upon
            ¦   3 ¦
¦ C_104_1 ¦ C_104 ¦ What is the rule of simple reflex agent?
            ¦   2 ¦
¦ C_104_2 ¦ C_104 ¦ In which of the following agent does the problem generator i
s present?  ¦   1 ¦
¦ C_105_1 ¦ C_105 ¦  Which of the following is not the promise of artificial neu
ral network? ¦   1 ¦
¦ C_105_2 ¦ C_105 ¦ Which of the following is an application of NN (Neural Netwo
rk)?        ¦   4 ¦
+---------+-------+------------------------------------------------------------
-------------+-----+
6 rows in set (0.00 sec)
```

7. QUES_2



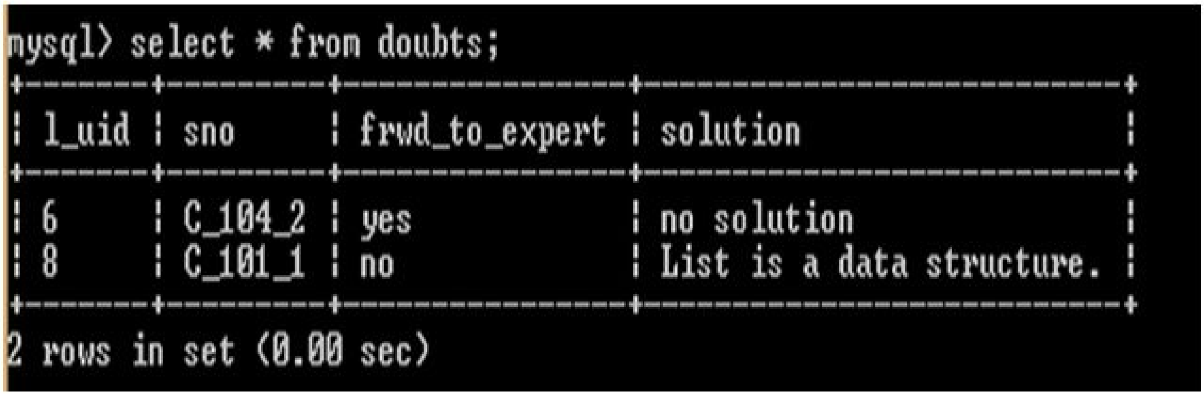
```
mysql> select * from ques_2;
+---------+-----------+------+
| sno     | corr_subm | diff |
+---------+-----------+------+
| C_101_1 |        49 |    9 |
| C_101_2 |       479 |    1 |
| C_104_1 |       300 |    5 |
| C_104_2 |       995 |    0 |
| C_105_1 |       100 |    8 |
| C_105_2 |       105 |    7 |
+---------+-----------+------+
6 rows in set (0.00 sec)
```

8. QUESTION_OPTIONS



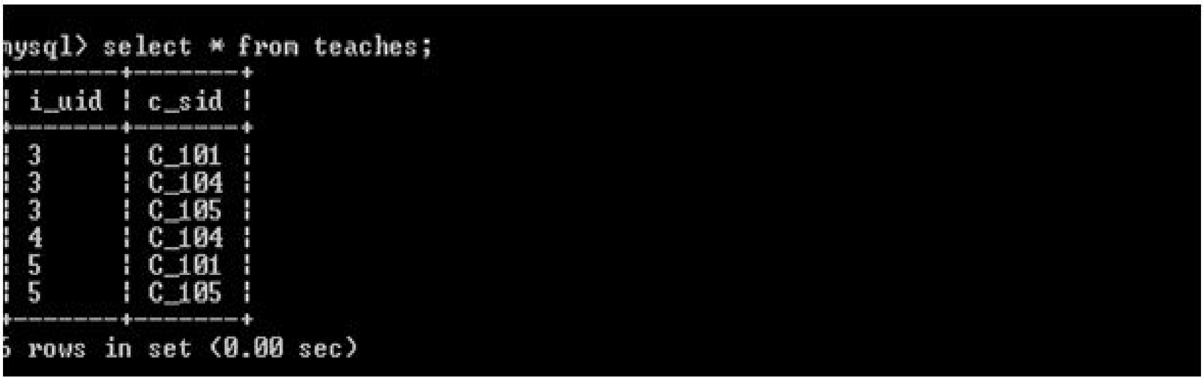
```
mysql> select * from question_options;
+---------+------------------------------------------+
| sno     | options                                  |
+---------+------------------------------------------+
| C_101_1 | Sorting                                  |
| C_101_1 | Merging                                  |
| C_101_1 | Inserting                                |
| C_101_1 | Traversal                                |
| C_104_1 | Simple Action Rule                       |
| C_104_1 | Condition Action Rule                    |
| C_104_1 | Both a & b                               |
| C_104_1 | none of the mentioned                    |
| C_101_2 | Time only                                |
| C_101_2 | Space only                               |
| C_101_2 | Both Time and Space                      |
| C_101_2 | None of the mentioned                    |
| C_104_2 | Learning agent                           |
| C_104_2 | Observing agent                          |
| C_104_2 | Reflex agent                             |
| C_104_2 | None of the mentioned                    |
| C_105_2 | Sales forecasting                        |
| C_105_2 | Data validation                          |
| C_105_2 | Risk management                          |
| C_105_2 | All of the mentioned                     |
| C_105_1 | It can explain results                   |
| C_105_1 | It can survive the failure of some nodes |
| C_105_1 | It has inherent parallelism              |
| C_105_1 | It can handle noise                      |
+---------+------------------------------------------+
24 rows in set (0.06 sec)
```

9. DOUBTS



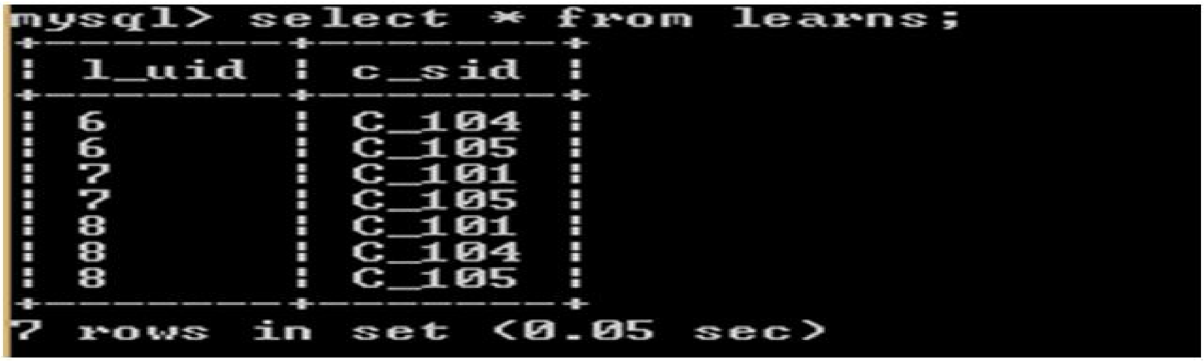
```
mysql> select * from doubts;
+-------+---------+----------------+--------------------------+
| l_uid | sno     | frwd_to_expert | solution                 |
+-------+---------+----------------+--------------------------+
| 6     | C_104_2 | yes            | no solution              |
| 8     | C_101_1 | no             | List is a data structure. |
+-------+---------+----------------+--------------------------+
2 rows in set (0.00 sec)
```

10. TEACHES



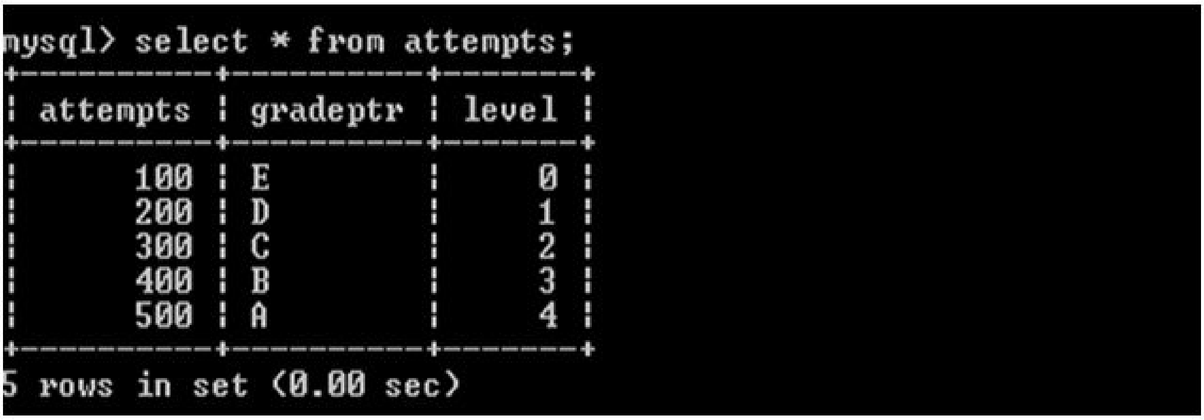
```
nysql> select * from teaches;
+-------+-------+
| i_uid | c_sid |
+-------+-------+
| 3     | C_101 |
| 3     | C_104 |
| 3     | C_105 |
| 4     | C_104 |
| 5     | C_101 |
| 5     | C_105 |
+-------+-------+
5 rows in set (0.00 sec)
```

11. LEARNS



```
mysql> select * from learns;
+--------+--------+
| l_uid  | c_sid  |
+--------+--------+
| 6      | C_104  |
| 6      | C_105  |
| 7      | C_101  |
| 7      | C_105  |
| 8      | C_101  |
| 8      | C_104  |
| 8      | C_105  |
+--------+--------+
7 rows in set (0.05 sec)
```

12. ATTEMPTS

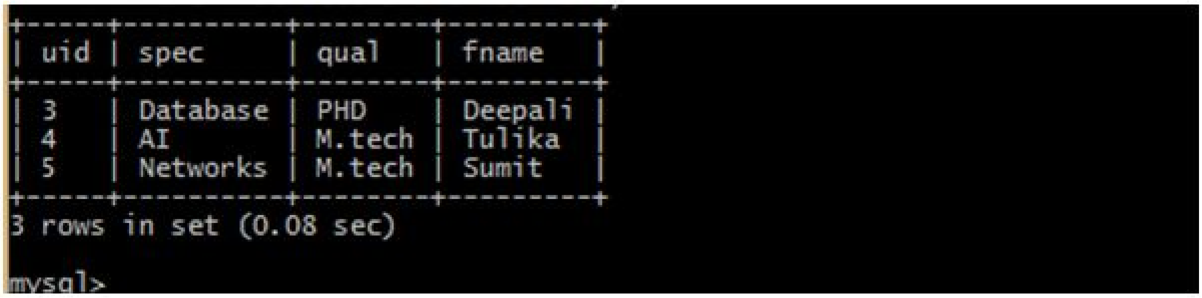


```
nysql> select * from attempts;
+----------+----------+-------+
| attempts | gradeptr | level |
+----------+----------+-------+
|      100 | E        |     0 |
|      200 | D        |     1 |
|      300 | C        |     2 |
|      400 | B        |     3 |
|      500 | A        |     4 |
+----------+----------+-------+
5 rows in set (0.00 sec)
```

QUERYING DATABASE

Query 1: Query to display Instructor Name, UID, Specialisation , Qualification.

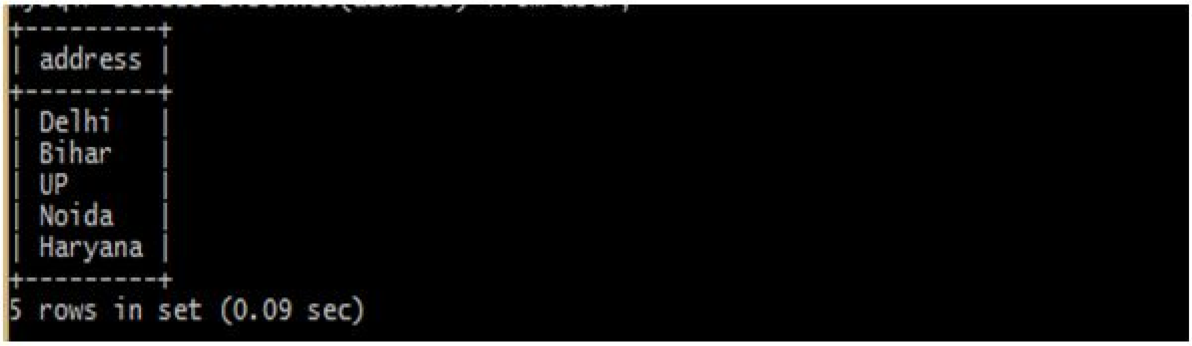
mysql> select instructors.uid,spec,qual,fname from instructors,user where instructors.uid=user.uid;



```
+-----+----------+--------+---------+
| uid | spec     | qual   | fname   |
+-----+----------+--------+---------+
| 3   | Database | PHD    | Deepali |
| 4   | AI       | M.tech | Tulika  |
| 5   | Networks | M.tech | Sumit   |
+-----+----------+--------+---------+
3 rows in set (0.08 sec)

mysql>
```

Query 2: Query to display unique address regions from the user table.

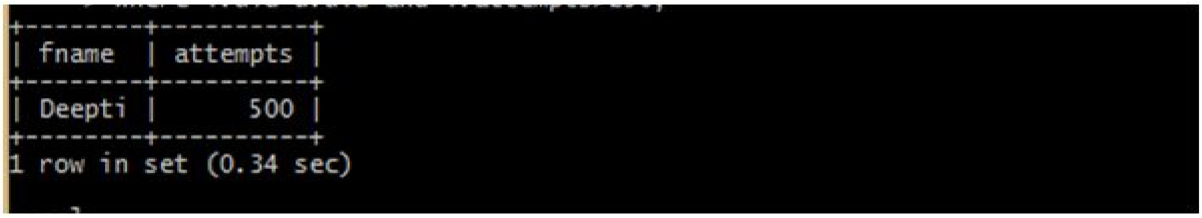
mysql> select distinct(address) from user;



```
+---------+
| address |
+---------+
| Delhi   |
| Bihar   |
| UP      |
| Noida   |
| Haryana |
+---------+
5 rows in set (0.09 sec)
```

Query 3. Query to display the user name and attempts of all the learners with attempts more than 250.

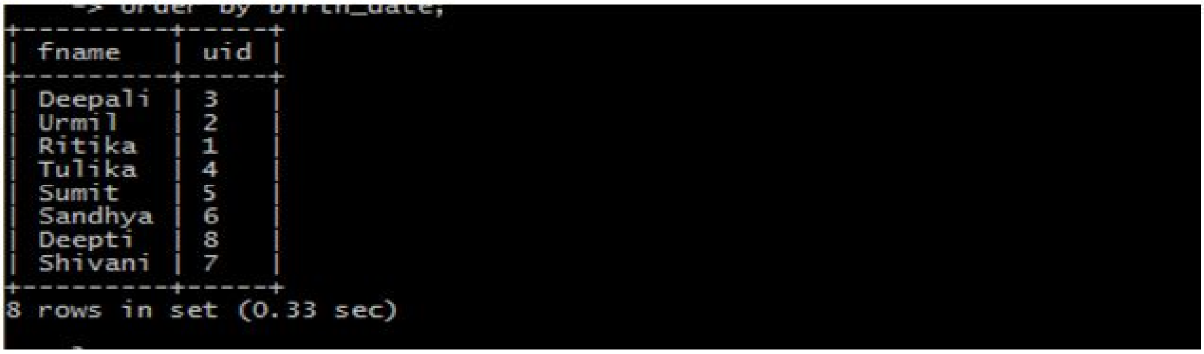
mysql> select u.fname,l.attempts rom user u,learners l where l.uid=u.uid and l.attempts>250;



```
+--------+----------+
| fname  | attempts |
+--------+----------+
| Deepti |      500 |
+--------+----------+
1 row in set (0.34 sec)
```

Query 4: Query to display fname and uid of users . Order the query in ascending order of birthdate.

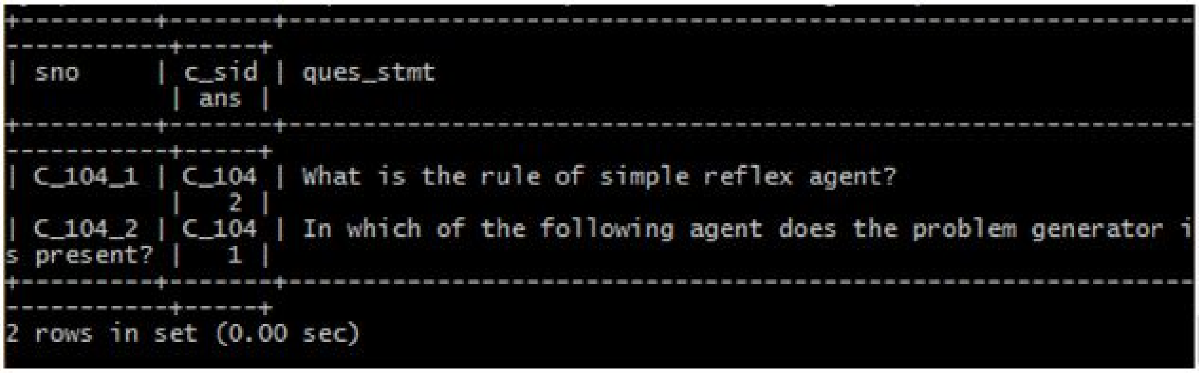
mysql> select fname,uid from user order by birth_date;



```
    -> order by birth_date;
+---------+-----+
| fname   | uid |
+---------+-----+
| Deepali | 3   |
| Urmil   | 2   |
| Ritika  | 1   |
| Tulika  | 4   |
| Sumit   | 5   |
| Sandhya | 6   |
| Deepti  | 8   |
| Shivani | 7   |
+---------+-----+
8 rows in set (0.33 sec)
```

Query 5: Query to display questions where the question contains the word “agent”.

*mysql> select * from questions where ques_stmt like '%agent%';*

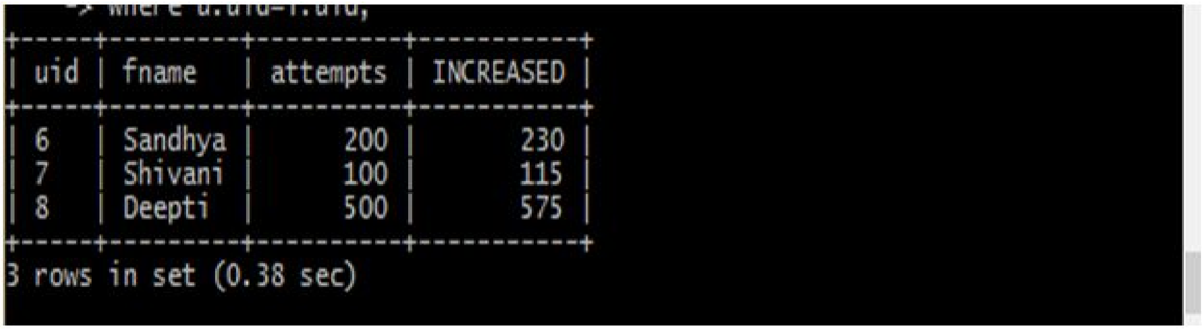


```
+---------+-------+--------------------------------------------------------------
----------+-----+
| sno     | c_sid | ques_stmt
          | ans |
+---------+-------+--------------------------------------------------------------
----------+-----+
| C_104_1 | C_104 | What is the rule of simple reflex agent?
          |   2 |
| C_104_2 | C_104 | In which of the following agent does the problem generator i
s present? |   1 |
+---------+-------+--------------------------------------------------------------
----------+-----+
2 rows in set (0.00 sec)
```

Query 6 : Query to display Learners UID, Name, and the attempts increased by 15 % expressed as a absolute whole number.

*mysql> select u.uid,u.fname,l.attempts,abs(round(l.attempts+l.attempts*0.15,0))*

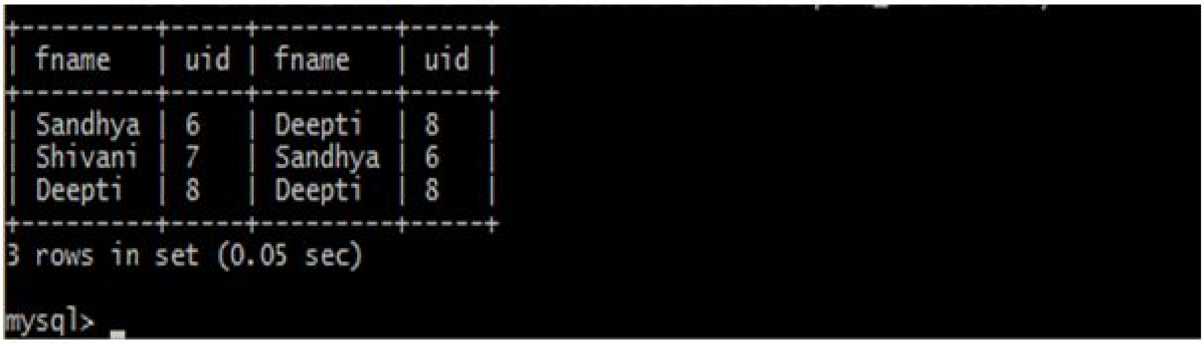
INCREASED from user u,learners l where u.uid=l.uid;



```
    -> where u.uid=i.uid;
+-----+---------+----------+-----------+
| uid | fname   | attempts | INCREASED |
+-----+---------+----------+-----------+
| 6   | Sandhya |      200 |       230 |
| 7   | Shivani |      100 |       115 |
| 8   | Deepti  |      500 |       575 |
+-----+---------+----------+-----------+
3 rows in set (0.38 sec)
```

Query 7 Query to display Name and UID. along with their Expert's Name and Expert's employee no.

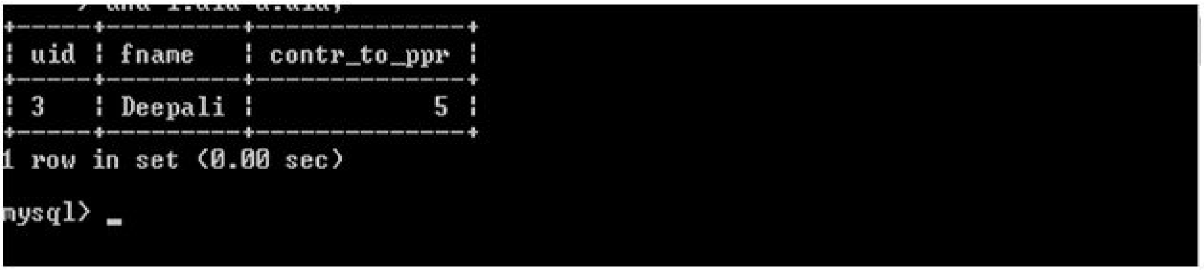
mysql> select u1.fname,u1.uid,u2.fname,u2.uid from user u1,user u2,learners l1,learners l2
where l1.uid=u1.uid and l2.uid=u2.uid and l1.expert_uid=l2.uid;



```
+---------+-----+---------+-----+
| fname   | uid | fname   | uid |
+---------+-----+---------+-----+
| Sandhya | 6   | Deepti  | 8   |
| Shivani | 7   | Sandhya | 6   |
| Deepti  | 8   | Deepti  | 8   |
+---------+-----+---------+-----+
3 rows in set (0.05 sec)

mysql> _
```

Query 8. Query to display the UID, Name and Number of Contribution for all instructors who contributed more than the average contribution rate and who teach Algorithms.

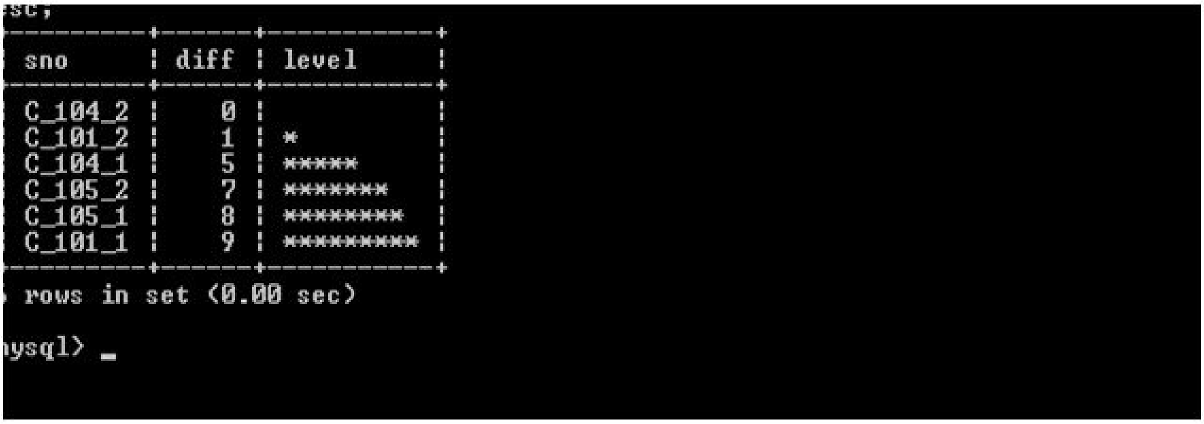
Select U.uid, u.fname , i.contr_to_ppr From instructors i, user u
Where i.contr_to_ppr > (select avg(contr_to_ppr) from instructors) and i.uid in(select i_uid from teaches where c_sid = (select c_sid from courses where cname="Algorithms")) and i.uid=u.uid;



```
    > and l.uid=u.uid;
+-----+---------+---------------+
| uid | fname   | contr_to_ppr  |
+-----+---------+---------------+
| 3   | Deepali |             5 |
+-----+---------+---------------+
1 row in set (0.00 sec)

mysql> _
```

Query 9. Query to display Question Sno, difficulty level represented by Asterisks – "Each asterisks (*) signifying 1"

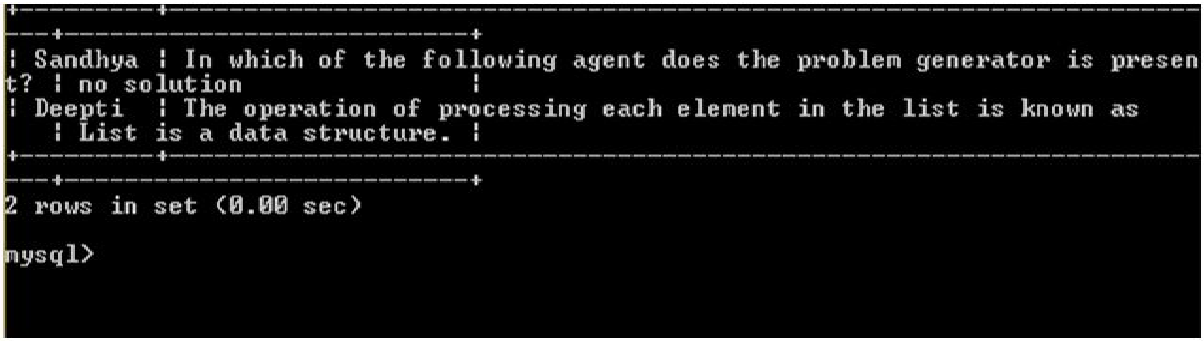
mysql> select sno,diff,rpad(' ',diff,'') level from ques_2 order by corr_subm desc;*



```
sc;
+----------+------+------------+
 sno      | diff | level      |
+----------+------+------------+
 C_104_2 |    0 |            |
 C_101_2 |    1 | *          |
 C_104_1 |    5 | *****      |
 C_105_2 |    7 | *******    |
 C_105_1 |    8 | ********   |
 C_101_1 |    9 | *********  |
+----------+------+------------+
 rows in set (0.00 sec)

ysql> _
```

Query 10. Query to display the name of the learner's name and their doubt solution and question statement.

Select u.uname,d.solution,q.ques_stmt
From user u, learner l, doubts d, questions q
Where l.uid=u.uid and d.l_uid=l.uid and d.sno=q.sno;



```
+---------+-------------------------------------------------------------------
---+---------------------------+
| Sandhya | In which of the following agent does the problem generator is presen
t? | no solution               |
| Deepti  | The operation of processing each element in the list is known as
   | List is a data structure. |
+---------+-------------------------------------------------------------------
---+---------------------------+
2 rows in set (0.00 sec)

mysql>
```

CONCLUSION

After the development of the project we have realised that such a database can be implemented in a good software project .We plan to develop the front end of the project in the near future to realise more usages of the database we developed in the semester.

Besides, this project has motivated us to realize the usefulness of the database in a software. We actually realized the usages and power of the tools provided to us by the database management system and SQL.