DATABASE MANAGEMENT SYSTEMS (10211CS207)

TASK:12 MICRO PROJECT

Team Details:

Team Leader:

P.Venkatadhri naidu (24UECS1071) VTU2613)

Team Members:

NAMES:C.BalaDurgeshkumar (24UECS1153) VTU: (VTU29843)

J.Sanjeev varma (24UECS0128) (VTU27899)

N.jeevan (24UECS0239) (VTU30481)

SK.kallimulla (24UECS0624) (VTU29843)

Registration no:

TITLE

STUDENT DATABASE

1.ER Diagram:

Aim:To draw the conceptual design for student database

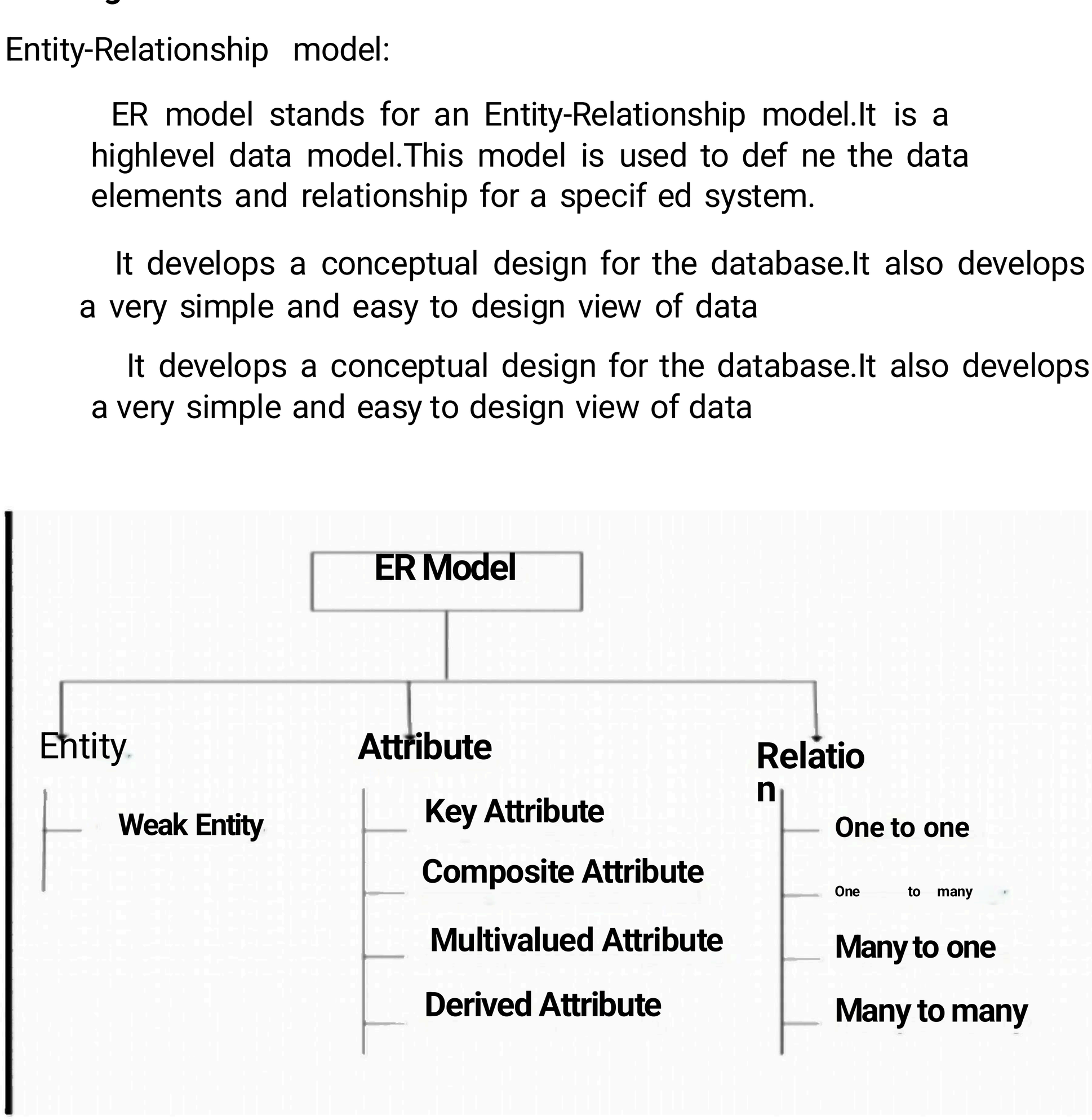
E-R Diagram

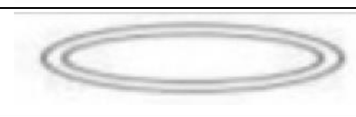
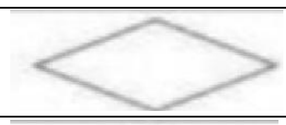
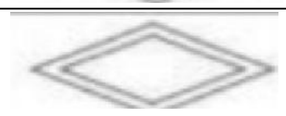
Entity-Relationship model:

ER model stands for an Entity-Relationship model.It is a highlevel data model.This model is used to def ne the data elements and relationship for a specif ed system.

It develops a conceptual design for the database.It also develops a very simple and easy to design view of data

It develops a conceptual design for the database.It also develops a very simple and easy to design view of data

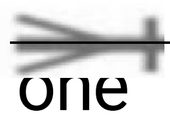


Entity Set	Strong Entity Set	
	Weak Entity Set	
Attributes 	Simple Attribute	
	Composite Attribute	
	Single- valued Attribute	
	Multivalued Attribute	
	Derived Attribute	
	Null Attribute	
Relationship	Strong Relationship	
	Weak Relationship	

one to one

one to many(mandatory)

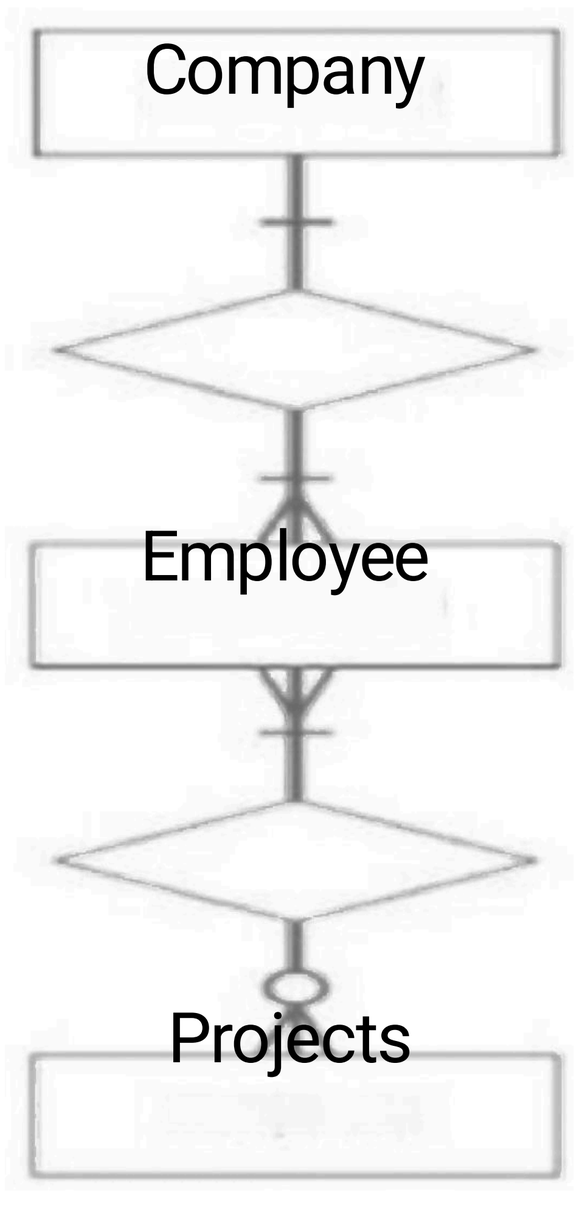

many


one or more(mandatory)


one and only one(mandatory)

zero or one(optional)


zero or many(optional)



WEAK ENTITY: An entity that depends on another entity called a weak entity.The weak entity doesnt contain any key attribute of its own.The weak entity is represented by a double rectangle.

ATTRIBUTE:The attribute is used to describe the property of an entity. Eclipse is used to represent an attribute

KEY ATTRIBUTE:The key attribute is used to represent the main characteristics of an entity.It represents a primary key.The key attribute is represented by an ellipse with the text underlined.

COMPOSITE ATTRIBUTE:An attribute that composed of many other attributes is known as a composite attribute.The composite attribute is represented by an ellipse,and those ellipses are connected with an ellipse.

MULTI VALUED ATTRIBUTE :An attribute can have more than one value. These attributes are known as a multivalued attribute.The double oval is used to represent multivalued attribute.

DERIVED ATTRIBUTE:Attributes which are derived from other attributes

ER-MODEL FOR STUDENT DATABSE

Entities: Student,Course,Professor,Department
Attributes:

- Student:StudentID,FirstName,LastName,DateOfBirth,Address
- Course:CourselD,CourseName,Credits
- Professor:ProfessorlD,FirstName,LastName,Department
- Department:DepartmentID,DepartmentName

Relations:

1.Enrolls In relation (between Student and Course):

- Foreign Key:Student ID in the Course entity
- Foreign Key:CourselD in the Student entity

2.Teaches relation (between Professor and Course):

- Foreign Key:Professor ID in the Course entity
- Foreign Key:CourselD in the Professor entity

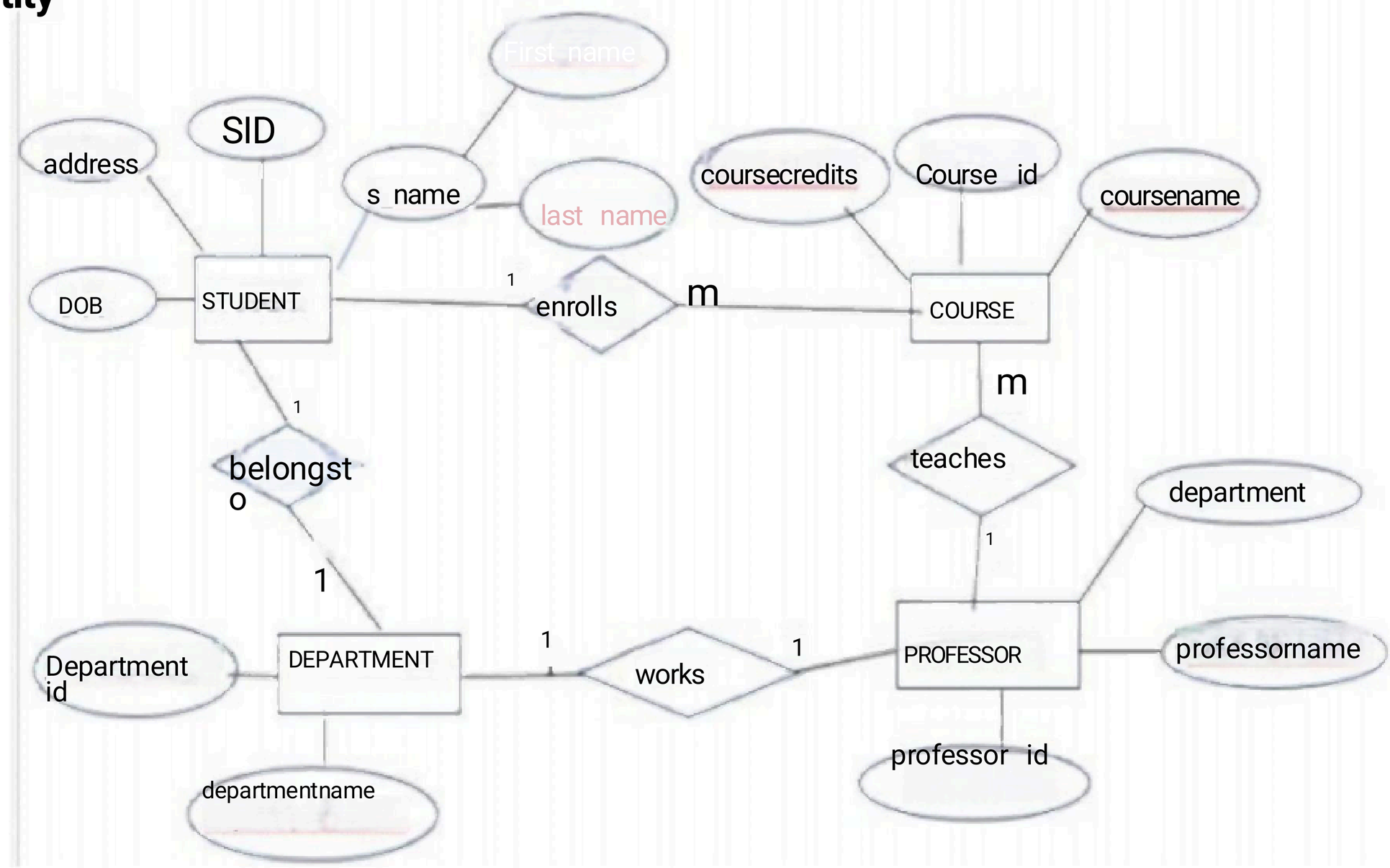
3.Belongs To relation (between Student and Department):

- Foreign Key:StudentID in the Department entity
- Foreign Key:DepartmentID in the Student entity

4.Works In relation(between Professor and Department):

·Foreign Key:ProfessorID in the Department entity

·Foreign Key:DepartmentID in the Professor entity

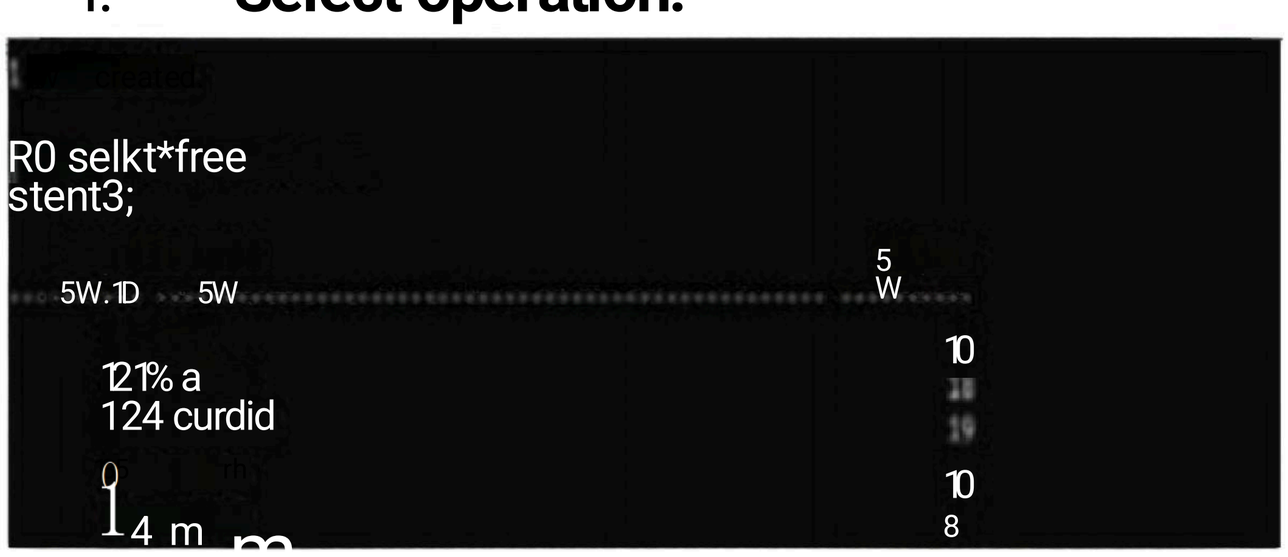


Result:Thus,the creating er diagram is completed successfully.

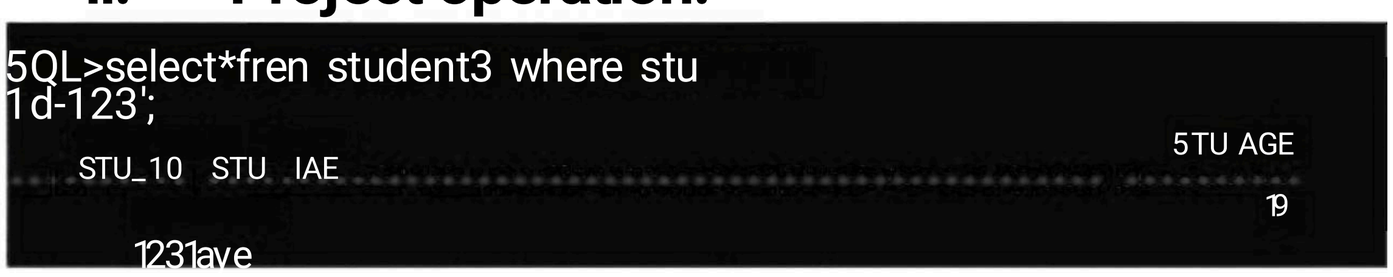
Aim:To execute relational operations,SQL aggregates,Joint queries for library management system.

Relational operations:

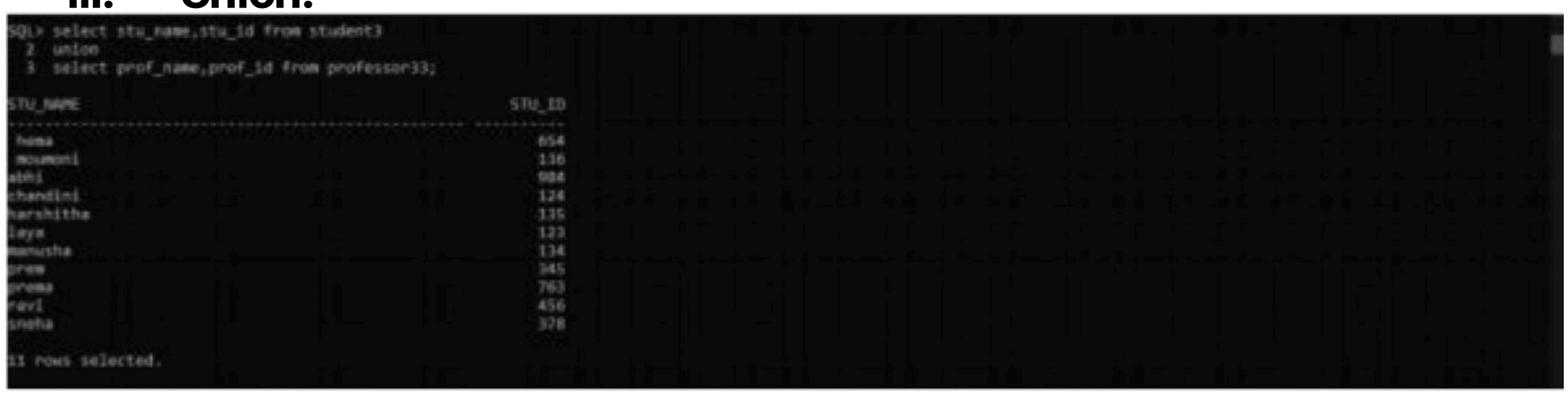
i. **Select operation:**



ii. **Project operation:**



iii. **Union:**



```
SQL> select stu_name,stu_id from student3
  2  union
  3  select prof_name,prof_id from professor33;

STU_NAME                                           STU_ID
-------------------------------------------------- ------
 hema                                                654
 moumoni                                             136
abhi                                                 984
chandini                                             124
harshitha                                            135
laya                                                 123
manusha                                              134
prem                                                 345
prema                                                763
ravi                                                 456
sneha                                                378

11 rows selected.
```

iv. **Union all:**

```
SQL> select stu_id,stu_name  from student3
  2   union all
  3  select prof_id,prof_dept from professor33;

    STU_ID STU_NAME
---------- --------------------------------------------------
       123 laya
       124 chandini
       135 harshitha
       134 manusha
       136  moumoni
       345  cse
       456 mech
       654 ece
       763 eee
       378  arts
       984  music

11 rows selected.
```

v.Minus:

```
Run SQL Command Line
SQL>select stu_name,stu_age from student3
  2     minus
  3  selec  prof_name,prof_id  from  professor33;
         t

STU_NAME                                                    STU_AGE
---------------------------------------------------------- ----------
 moumonj                                                           18
Chandini                                                           18
harshitha                                                          19
laya                                                               19
manusha                                                            18

sQl>
```

SQL Aggregate

I. Count:

```
SQL>select count(*)fren student3;
  count()
----------
         5
```

ji. Sum:

```
5QL>select   sun(stu_age)from   student3;
SUM(STU_AGE)
------------
          92
```

jii. Average:

```
5QL>select   avg(prof_id)from
professor3;

AVG(PROF_ID)---

   596.666667
```

iv. Maximum:

```
5QL>select  max(stu_1d)from  student3;
WAX(STU_ID)

          136
```

v. Minimum:

```
5QL>select   min(prof_id)from   professor3;
MIN(PROF_ID)

          345
5QL>abhi
```

JOIN QUERIES

1. LEFT JOIN

```
SQL>select   pofessor33.prof_id,student3.stu_name   from   profesor33   left   join
  2                on    professor33.stu_id=student3.stu_id;
  student3
                   STU  NAME
      PROF_ID
------------  ------------
        345          Laya
                  Chandhini
        346       Harshitha
        347          Null
                     Null
        34
```

2.I **NNER** JOIN

```
SQL> select  pofessor33.prof_id, student3.stu_name, student3.stu_age  from
          2    Student3  inner  join  professor33  on  student3.stu  id= professor33.stu  id;
     PROF_ID     STU_NAME     STU_AGE


        345          Laya
                                  19
        346    Chandhini          18
        347    Harshitha          19
        348       Manusha         18
        349       Moumoni         18
```

3.RIGHT JOIN

```
SQL>select   pofessor33.prof_id,student3.stu_name   from   profesor33   right   join 2
   student3   on    professor33.stu_id=student3.stu_id;
     PROF_ID       STU_NAME



        345
        Laya
        346      Chandhini

        347      Harshitha
       Null         Manusha
       Null         Moumoni
```

4.FULL
JOIN

```
SQL> select  pofessor33.prof  id,student3.stu_id,student3.stu_name,
  2  student3.stu age from profesor33 full join student3 on
  3    professor33.stu_id=student3.stu_id;
```

PROF_ID	STU_ID	STU NAME	STU AGE
345	123	Laya	19
346	124	Chandhini	18
347	135	Harshitha	19
348	134	Manusha	18
349	136	Moumoni	18

RESULT:Thus,to execute relational operations,SQL aggregates,join queries for library management system is successfully executed.

Normalization

Normalization in the context of databases refers to the process of organizing data in a database ef Eiently.The goal is to reduce data redundancy and dependency by organizing f elds and table of a database.This helps in minimizing the anomalies that can arise when modifying the data.

There are several normalforms(NF)that def ne the levels of normalization, with each normal form addressing different types of issues:

First Normal Form(1NF):

- Eliminate duplicate columns from the same table.
- Create a separatetable for each group of related data and identify each row with a unique column or set of columns.

Second Normal Form (2NF):

- Meet all the requirements of 1NF.
- Remove partial dependencies-ensure that non-prime attributes are fully functionally dependent on the primary key.

Boyce-Codd Normal Form(BCNF):

- A more stringent form of 3NF.
- For a table to be in BCNF,it must satisfy an additional requirement compared to 3NF,dealing specif cally with certain types of functional dependencies.

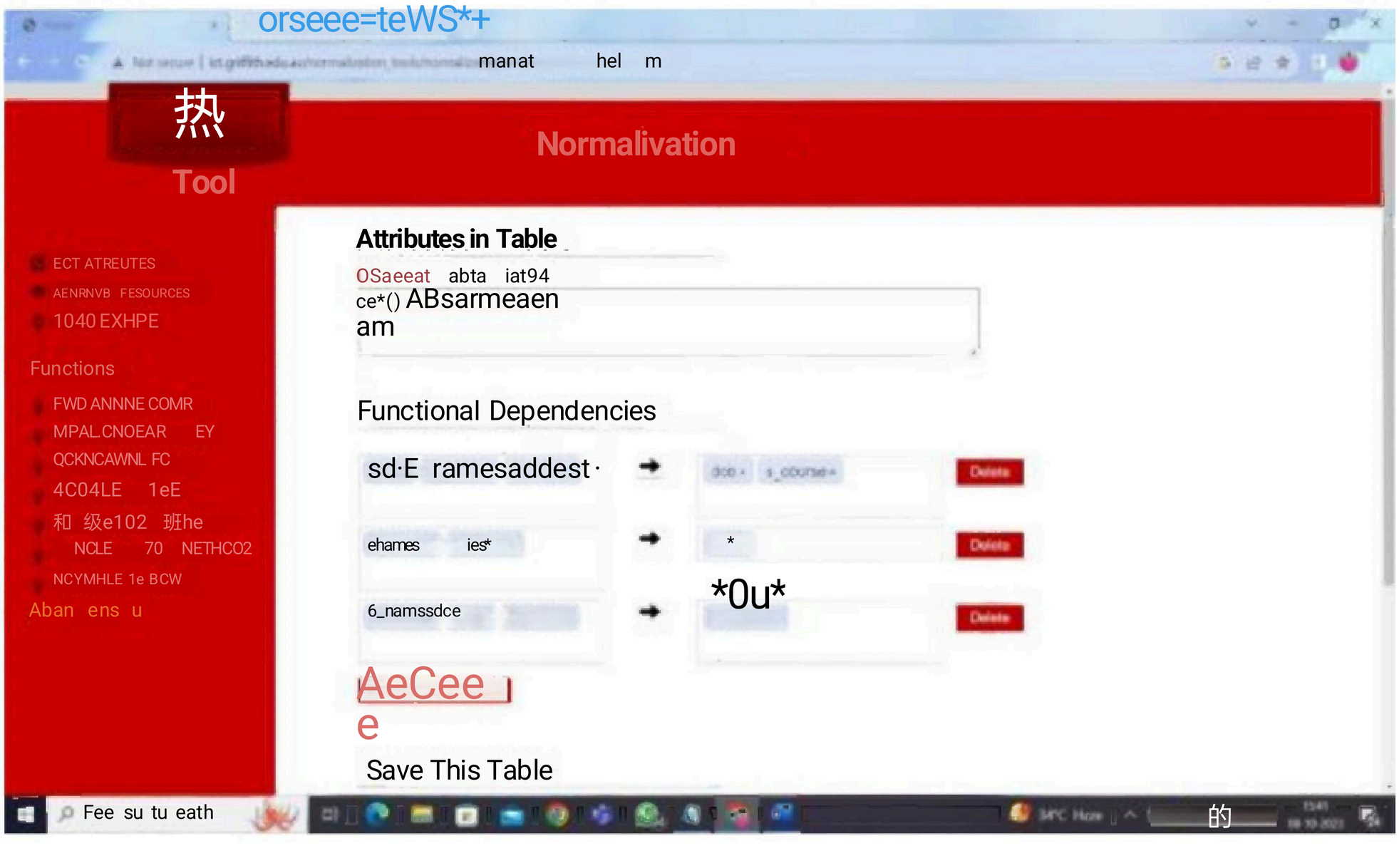
In this database we perform normalisation using Grif fh university normalisation tool

Steps to follow for doing normalisation using Grif fh normalisation process:

Step1: search for Grif th university normalisation tool in web browser

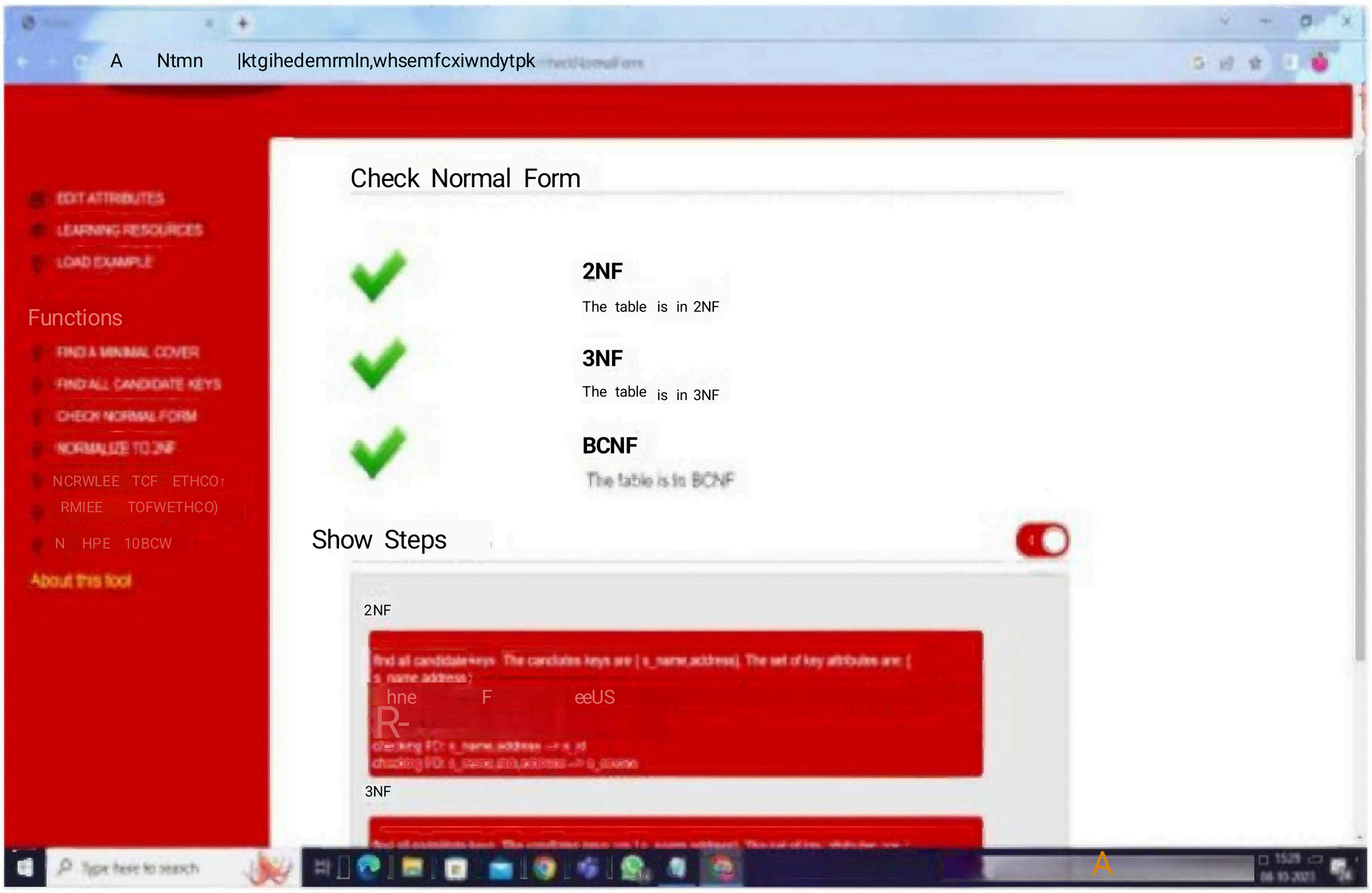
Step2:After opening the tool enter the attributes of the entity S_ID,s-name,address,dob,s-course.Make sure to separate the **attributes using commas in between them.**

Step3:Add the dependencies of the attributes



Do as per your entity and add the dependencies as shown in the f g.

Step4:In the left if the window below functions on check normal form option



We will get the screen shown above the normal form of the given attributes is checked (BCNF)

And the following steps are displayed below:

2NF:

1.f nd all candidate keys.The Candidate keys are{s_name,address},
The set of key attributes are:{s_name,address}
2.for each non-trivial FD,check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes
3.checking FD:s_id,s_address -->dob,s_course
4.checking FD:s_name,address ->s_id
5.checking FD:s_name,dob,address ->s_course

3NF:

1.f nd all candidate keys.The candidate keys are{s_name,address},
The set of key attributes are:{s_name,address}
2.for each FD,check whether the LHS is super key or the RHS are all key attributes

3.checking functional dependency s_id,s_name,address->dob,s_course

4.checking functional dependency s_name,address ->s_id

5.checking functional dependency s_name,dob,address ->s_course

BCNF:

1.A table is in BCNF if and only if for every non-trivial FD,the LHS is a **super key.**

Result:Thus,the normalization to 1nf,2nf,3nf,BCNF is completed successfully.

Aim:To implement the document database and graph database by using Mon gosh.

```
1      .crotstollstion("stadnt")
2(@11)
sa,lab,Imertow(tus"jte",aeeizs,des                taenti['sdesce'],siaeifhra,9s.5,a*
4                                                  wJ)
ackrouledyd:trus,
s
imertedId:chjscttd("as2)f12ad)*1PHbeofadts")
   ,lab,flnd((mei"j*))
■(
     ie:CbjktId("652)f47MBb4IbaTNbuloL*),
      qeel'johe',
     a* B
     ertw
     siie:4 hi 2a,ke 38.5,m
      mt:(wime),
  *   ulededitne,
16*issrtsdlah*:[
1s  jctla  ss sa  s a  hcyt  te  ,)
   ject1d(   2sfs2a4sai?rbe  4eH)·
20
1)
2    @,1ab,find(,(   1,     ))
a{    aat   aifatnuonelr)                    酒
     ia:cj*t1d(*6523f4*4*172te920LDfo=*),
  (iei              cojktid(W)fs*4stte      tof),eae',          }
20        a,mlab.find(),(a:,4y=)).rwttyO
                                          0to
```

Owtput

```
omplle          (
 4
   ind:* j =tid(*6534@                 adfas4s*
                                        ).
   hb

    t23,
   drpartasnt:['scimee'].
   slLrei(    2*,    35,5,
mkoepll
e
symtastrro   wissirg
rt           semicolan.(3:a)

 11(
 21             ceject1*6523f4700401A27bu4B1*),
>3                *jon',
 4|             23,
```

1

```
db.mylab.find({},{name:1,qty:1}).pretty()
{ _id: ObjectId("6523fc194182d56b489f9231"),
"name": 'john',
"age":23
}
{ _id: ObjectId("6523fc91b1f70a831ff9d09a"),
name: 'sure',
"age":25
}
db.mylab.find({name:"john"}).pretty().sort({john:
(
   i4   bject1(*652D#e91222d040d000e2S4*),


   是          ,
```

mtaet
tiawrrhotatk

Output

```
>    1)

 2

tler             (
 (    4=kject1K*024c00xhtxkaldfdo*),
ytsgl)e* o... rc                 :
Spstasfrrer:   ssing semicolan.《150)
                                      >11(j
      jktiesettcrat?Neaanfak"),a
 21(is:ceject1"fbueat7N           ofad*),n:
 )
 tmgller*              ... c:
 ataatrrer      ssirg sesicolon,(1150)
```

m te h&p tsdg

```
  _id: ObjectId("6523fe91225d94e6d3d6e2d4"),
  name: 'john',
  age: 23,
  department: [ 'science' ],
  size: { h: 28, w: 35.5, uom: 'cm' }
 }
]
db.mylab.deleteOne({name:"sure"})
            ... ... ...
db.mylab.find({},{name:1,age:1}).pretty()
insertedId: ObjectId("6523f32ad33d17fbe0f4dc75c"),
  "name":"john",
  "age":23
}
{
  "_id":ObjectId("6523f32ad33d17fbe0f4dc75d"),
    "name":"sure"
    "age":25
}
{"_id":ObjectId("6523f32ad33d17fbe0f4dc75e"),"name":"
{
 "_id":ObjectId("6523f32ad33d17fbe0f4dc75f"),
   "name":"usha",
   "age":34
}
```

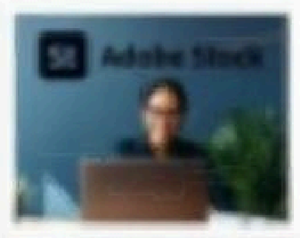
Output

```
satasfrrarsumapeeted tokm,weectad ",*(2=1#)

   114h.mlab,leleteoe(lne:*sre")
>21             ... ... ...
31

myromgtler   ergpeto(
(ld:okjectiK*652*02                *),         :'jem',
myrogtler          eredo... Uheaht:

setatrrore pissing semicolon.(2:10)
1l iasertene:jctidf wafsaataftestaecDk],
2        *a**jcn²,
```



Result:Thus implemented the document database and graph database by using Mon gosh.