VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

DEPARTMENT OF INFORMATION TECHNOLOGY
SCHOOL OF INFORMATION TECHNOLOGY AND ENGINEERING
VELLORE INSTITUE OF TECHNOLOGY

Submitted by:

R. Uma Sai Charan—18BIT0160

Satya AAkash Chowdary.o---18BIT0128

Rohit. M.Somani---18BIT0236

In partial fulfilment of the requirements for Project J component – ITE 1003

INDEX:

Introduction:

This is a complete data base of a university education system. This contains all the information about the university. It has all the information about administration, students, co-curricular activities, faculty details etc. This is a very good way to keep a record on all the accepts of the college.

Software used: Oracle 11g

Conclusions from ER diagram:

Major Entities:

This database consists of major entities like:

(1) College
(2) Student
(3) Hostel
(4) Faculty
(5) Library
(6) Department
(7) Courses
(8) Extra-curricular activities

Weak entities:

This database consists of minor entities like:

(1) Student Clubs

Relationships:

Courses→are offered by→college

Students→studies in→college

Library→is a part of→college

Extra-curricular→are conducted by→college

Student clubs→are formed in→college

Faculty→works for →college

Hostel facilities →are provided by→college

Department →is division of faculty in→college

Identifying Relation:

1) Student clubs→are formed in→college

To make a relational database there should be a like every relation (table) should be related to each other that is a query that can be made upon our database should be able to retrieve the values as such.so here by inserting the new foreign keys into relation we make the relation to be related with each other.

This can also said to be mapping of e-r diagram to a relational schema. The constraints are the ones which impose or enforce the rules over the database to remove complexity and the possibilities of the redundancy. Constraints are of 3 types

1. Key constraints

2. Entity integrity constraints

3. Referential integrity constraints possibly we should avoid the null values as less as possible to make our database In order a design a database some GUIDELINES are provided

1. INFORMAL GUIDELINES

2. FORMAL GUIDELINES

MAPPING OF E-R DIAGRAM TO RELATIONAL SCHEMA :

Table Name: College

ATTRIBUTE NAME	DATA TYPE	CONSTRAINT
College Name	VARCHAR2(30)	Not Null
College Id	NUMBER(10)	Primary Key
College phone No	NUMBER(12)	Multi-valued
College type	VARCHAR2(30)	Not-null
College Address	VARCHAR2(30)	Not Null
E-mail	VARCHAR2(30)	Not Null

Table Name: Courses

ATTRIBUTE NAME	DATA TYPE	CONSTRAINT
Course Id	NUMBER(12)	Primary Key
Course Name	VARCHAR2(10)	Not Null
Modules	NUMBER(2)	Not Null

Table Name: College Faculty

ATTRIBUTE NAME	DATA TYPE	CONSTRAINT
Faculty Id	NUMBER(10)	Primary Key
Faculty Name	VARCHAR2(10)	Not Null
Faculty Address	VARCHAR2(10)	Not Null
Faculty phone number	NUMBER(12)	Not Null
Faculty E-mail	VARCHAR2(10)	Not Null
Department Number	NUMBER(6)	Not Null
Date of birth	DATE	Not Null
Salary	NUMBER(6)	Not Null

Table Name: Student

ATTRIBUTE NAME	DATA TYPE	CONSTRAINT
Name	VARCHAR2(10)	Not Null
Reg No	VARCHAR2(10)	Primary Key
DOB	DATE	Not Null
Phone No.	NUMBER(12)	Not Null
E-mail	VARCHAR2(10)	Not Null
Branch	VARCHAR2(10)	Not Null

Tale Name: Hostel

ATTRIBUTE NAME	DATA TYPE	CONSTRAINT
Block Name	VARCHAR2(10)	Primary Key
Phone No	NUMBER(12)	Not Null
Staff	VARCHAR2(30)	Not Null
Address	VARCHAR2(10)	Not Null
Mess	VARCHAR2(30)	Not Null

Table Name: Library

ATTRIBUTE NAME	DATA TYPE	CONSTRAINT
Name	VARCHAR2(10)	Not Null
Library ID	NUMBER(5)	Primary Key
Staff	VARCHAR2(10)	Not Null
Books	VARCHAR2(10)	Not Null
Books issue ID	NUMBER(5)	Primary Key

Table Name: Department

ATTRIBUTE NAME	DATA TYPE	CONSTRAINT
Name	VARCHAR2(10)	Not Null
Dept. No	NUMBER(5)	Primary Key
Faculty ID	NUMBER(5)	Not Null

Table Name: Extra Curricular

ATTRIBUTE NAME	DATA TYPE	CONSTRAINT
Event Name	VARCHAR2(10)	Not Null
Event ID	NUMBER(5)	Primary Key
Faculty ID	NUMBER(10)	Not Null

Table Name: Student club

ATTRIBUTE NAME	DATA TYPE	CONSTRAINT
Name	VARCHAR2(10)	Not Null
No. of members	NUMBER(5)	Not Null
Email	VARCHAR2(10)	Not Null
Faculty ID	NUMBER(5)	Not Null
College ID	NUMBER(5)	Foreign Key

College:

College Name	College Id	College type	College Address	E-mail

Constraints:

1) Constraint coid_pk PRIMARY KEY(college id);

2) Constraint name_chk CHECK(name!=NULL);

3) Constraint name_chk1 CHECK (college type!=NULL);

4) Constraint name_chk2 CHECK (college address!=NULL);

5) Constraint name_chk3 CHECK (E-mail!=NULL);

Functional Dependencies

College ID→{ College Name, College type, College Address, E-mail};

College……

College ID	College Name	College type	College Address	E-mail
001234	VIT	Deemed	Vellore	vit@vit.ac.in
001235	SRM	Deemed	Chennai	srm@gmail.com
001239	KLU	Deemed	Vijayawada	klu@gmail.com
001325	Anna Universi	University	Chennai	anna@gmail.com

	ty			
032562	Andhra university	University	Vizag	au@gmail.com

Courses

Course Id	Course Name	Modules	College ID

Constraints:

1) Constraint cid_pk PRIMARY KEY(course id);

2) Constraint coid_fk FOREIGN KEY(College id) references College(college id);

3) Constraint course_chk CHECK (name!=NULL);

4) Constraint modules_chk CHECK(modules!=NULL);

Functional Dependencies

Course Id→{course name, modules, college id};

College id→{ Course Id};

Course Id	Course name	modules	college id
1001	DBMS	7	001234
1002	CAO	8	001235
2001	OS	6	001239

2002	DSA	7	001325
2003	ALA	9	032562
2004	TOC	7	001234

College Faculty

Faculty Id	Faculty Name	Faculty Address	Faculty phone number	Faculty E-mail	Department Number	Date of birth	Salary	College id

Constraints

1) Constraint fid_pk PRIMARY KEY(faculty id);

2) Constraint cid_fk1 FOREIGN KEY(college id) references college(college id);

3) Constraint fn_chk CHECK(Faculty Name!=NULL);

4) Constraint fn_chk1 CHECK(Faculty Address !=NULL);

5) Constraint fn_chk2 CHECK(Faculty phone number!=NULL);

6) Constraint fn_chk3 CHECK(Faculty E-mail !=NULL);

7) Constraint fn_chk4 CHECK(Department Number!=NULL);

8) Constraint fn_chk5 CHECK(Date of birth!=NULL);

9) Constraint fn_chk6 CHECK(Salary!=NULL);

Functional Dependencies

Faculty Id→{ Faculty Name, Faculty Address, Faculty phone number, Faculty E-mail, Department Number, Date of birth, Salary};

College id→{ Faculty Id};

Faculty……..

Faculty Id	Faculty Name	Faculty Address	Faculty phone number	Faculty E-mail	Dept No:	DoB	Salary	College id
1011	Kiran	Vellore	9123456789	Kiran@gmail.com	230	1-2-86	150000	001234
1012	praveen	Chennai	8987654332	praveen@gmail.com	120	9-5-83	50000	001235
1023	Kumar	Vizag	9894573133	kumar@gmail.com	542	9-4-81	75000	001239
1024	Raju	Chennai	8897456212	raju@gmail.com	325	9-3-86	100000	001325
1234	Lakshmi	Vijayawada	7894256422	lakshmi@gmail.com	132	8-6-88	80000	032562

Student

Name	Reg No	DoB	Phone No	E-mail	Branch	College id

Constraints

1) Constraint reg_pk PRIMARY KEY(Reg no !=NULL);

2) Constraint coid_fk2 FOREIGN KEY(college id) references College (college id);

3) Constraint sn_chk CHECK(name!=NULL);

4) Constraint sn_chk1 CHECK(DoB!=NULL);

5) Constraint sn_chk2 CHECK(phone no!=NULL);

6) Constraint sn_chk3 CHECK(e-mail!=NULL);

7) Constraint sn_chk4 CHECK(Branch!=NULL);

Functional Dependencies

Reg no→{ Name, DoB, Phone No, E-mail, Branch,College id};

College id→{reg no}

Student……

Name	Reg no	DoB	Phone No	e-mail	Branch	College id
Gowtham	18BCE0128	1-12-2001	8974565689	gowtham@gmail.com	CSE	001234
Vijaya	18BME5644	2-5-01	7892351555	vijay@gmail.com	Mech	001235
Aditya	18BEC5695	2-6-00	7816489995	a@gmail.com	ECE	001239
Sucheta n	18BIT02556	1-12-00	9859455666	SU@gmail.com	IT	001325
vamsi	18BEE5478	2-4-99	8457991253	vamsi@gmail.com	EEE	032562

Hostel

Block Name	Phone no	Staff	Address	Mess	College Id

Constraints

1) Constraint hn_pk PRIMARY KEY(Block Name);

2) Constraint coid_fk3 FOREIGN KEY(College id) references college(college id);

3) Constraint hn_chk CHECK(phone no !=NULL);

4) Constraint hn_chk1 CHECK(staff !=NULL);

5) Constraint hn_chk2 CHECK(address!=NULL);

6) Constraint hn_chk 3CHECK(mess !=NULL);

Functional Dependencies

Block name→{Phone no, staff, address, mess, college id};

College id→{Block name};

Hostel……

Block Name	Phone No	Staff	address	Mess	College id
f-block	9596626553	Veera	Vellore	NV-veg	001234
g-block	8594856862	Narayan	Chennai	Aac4-non veg	001235
d-block	9845236232	Venkat	Vijayawada	R1c1-spl	001239
p-block	9641235631	Raju	Chennai	J22-veg	001325
m-block	7532352322	kumar	vizag	G88-non veg	032562

Library:

Name	Library ID	Staff	Books	Books issue ID	College id

Constraints:

1) Constraint ln_pk PRIMARY KEY(Library id);

2) Constraint ln_pk1 PRIMARY KEY(Books issue id);

3) Constraint coid_fk4 FOREIGN KEY(College id) references college(college id);

4) Constraint ln_chk CHECK(Name!=NULL);

5) Constraint ln_chk1 CHECK(staff!=NULL);

6) Constraint ln_chk2 CHECK(books!=NULL);

Functional Dependencies

Library Id→{name, staff, books, books issue id, college id} ;

Books issue id→{books,college id};

College id→{library id, books issue id };

Library……

Name	Library id	staff	books	Books issue id	College id
periyar	11011	kumar	Ece 6.1	2201	001234
central	12345	raju	Commerce 5.2	1201	001235
mahavir	23567	praveen	Dbms 7	1301	001239

ramanjuan	87945	roy	Os 2.1	1401	001325
Radha krishna	58746	ravi	Cao 3.2	1501	032562

Department

Dept.name	Dept.no	Faculty id	College id

Constraints

1)Constraint dn_pk PRIMARY KEY(Dept.no);

2)Constraint coid_fk5 FOREIGN KEY(college id) references college(college id);

3)Constraint dn_chk CHECK(Dept.name!=NULL);

4) Constraint dn_chk CHECK(Faculty id!=NULL);

Functional Dependencies

Dept.no→{dept.name,faculty id,college id};

College id→{dept.no};

Department……

Dept.name	Dept.no	Faculty id	College id
SCOPE	1102	87545	001234
SITE	1234	85214	001235

SENSE	2345	89652	001239
SELECT	3456	23568	001325
SMEC	4567	78945	032562

EXTRA Curricular

Event name	Event id	Faculty id	College id

Constraints

1) constraint ex_pk PRIMARY KEY(Event id);

2) constraint coid_fk6 FOREIGN KEY(College id) references College(college id);

3) constraint ex_chk CHECK(Eventname!=NULL);

4) constraint ex_chk1 CHECK(Faculty id!=NULL);

Functional Dependencies

Event id→{event name,faculty id,college id};

College id→{event id};

EXTRA Curricular……

Event name	Event id	Faculty id	College id

Ncc	1234	78945	001234
Nss	2354	56321	001235
sports	2547	25478	001239
Dance	8965	14569	001325
Music	7854	87452	032562

Student club

Name	No.of members	Email	Faculty id	College id

Constraints

1) Constraint coid_fk6 FOREIGN KEY(College id) references college(college id);

2) Constraint sn_chk CHECK(name!=NULL);

3) Constraint sn_chk1 CHECK(no.of members!=NULL);

4) Constraint sn_chk2 CHECK(email!=NULL);

5) Constraint sn_chk3 CHECK(faculty id!=NULL);

Functional Dependencies

College id→{name,no.ofmembers,email,faculty id, college id};

Student club……

Name	No.of members	Email	Faculty id	College id
codechef	80	codechef@gmail.com	1123	001234
IEEE	75	ieee@gmail.com	1234	001235
Fepsi	84	fepsi@gmail.com	1334	001239
Spartans	90	Spartan@gmail.com	1434	001325
GDG	114	gdg@gail.com	1534	032562

Phone_number

College id	Ph1	Ph2	Ph3	Ph4	Ph5

Constraints

Constraint ph_fk FOREIGN KEY(College id) references college(college id);

Functional Dependencies

College id→{ph1,ph2,ph3,ph4,ph5};

Phone_number……..

College id	Ph1	Ph2	Ph3	Ph4	Ph5
001234	9445677111	9445677112	9445677113	9445677114	9445677115
001235	8974588111	8974588112	8974588113	8974588114	8974588115
001239	7458967111	7458967112	7458967113	7458967114	7458967115
001325	9925578111	9925578112	9925578113	9925578114	9925578115
032562	8874598111	8874598112	8874598113	8874598114	8874598115

Normalization:

WE can say that one given relation is in 1NF by eliminating the

1.) multi-valued and creating new table for it

2) composite attribute and creating new table for it

3.) nested relations

College Name	College Id	College type	College Address	E-mail

[college]

Course Id	Course Name	Modules	College ID

[courses]

Faculty Id	Faculty Name	Faculty Address	Faculty phone number	Faculty E-mail	Department Number	Date of birth	Salary	College id

[college faculty]

Name	Reg No	DoB	Phone No	E-mail	Branch	College id

[student]

Block Name	Phone no	Staff	Address	Mess	College Id

[hostel]

Name	Library ID	Staff	Books	Books issue ID	College id

[library]

Dept.name	Dept.no	Faculty id	College id

[department]

Event name	Event id	Faculty id	College id

[extra curricular]

Name	No.of members	Email	Faculty id	College id

[student club]

College id	Ph1	Ph2	Ph3	Ph4	Ph5

[phone_number]

Above all the relations has no multivalued attribute except the college table has the phone_number So we have created a separate table for the phone_number with college id as key. And in the remaining tables there are no multivalued attributed or composite attributes and no nested relations. There by here we can conclude that the above formed relations are in '1NF'.

2NF:

X->A BELONGS TO "F" {THE FUNCTIONAL DEPENDENCIES SET}
X MUST BE A PRIME ATTTRIBUTE. AND A IS A NON-PRIME ATTRIBUTE

College Name	College Id	College type	College Address	E-mail

[college]

Course Id	Course Name	Modules	College ID

[courses]

Faculty Id	Faculty Name	Faculty Address	Faculty phone number	Faculty E-mail	Department Number	Date of birth	Salary	College id

[college faculty]

Name	Reg No	DoB	Phone No	E-mail	Branch	College id

[student]

Block Name	Phone no	Staff	Address	Mess	College Id

[hostel]

Name	Library ID	Staff	Books	Books issue ID	College id

[library]

Dept.name	Dept.no	Faculty id	College id

[department]

Event name	Event id	Faculty id	College id

[extra curricular]

Name	No.of members	Email	Faculty id	College id

[student club]

College id	Ph1	Ph2	Ph3	Ph4	Ph5

[phone_number]

IN THE ABOVE TABLES THE IMPLIED FUNCTIONAL DEPENDECIES ON EACH RELATION OF TYPE ‘X->A’ THE ‘X’ IS A PRIME ATTRIBUTE IN EACH FUNCTIONAL DEPENDENCIES WHERE AS THE ‘A’ IS THE NON-PRIME ATTRIBUTE.

College ID→{ College Name, College type, College Address, E-mail};

THE ABOVE ARE THE FUNCTIONAL DEPENDENCIES FROM THE TABLE ‘CUSTOMER’.HERE ABOVE WE OBSERVE THAT

FOR EACH FUNCTIONAL DEPENDENCY THE X->A X IS A PRIME ATTRIBUTE.

3NF:

1.) MUST BE IN 2NF.

2.) FOR EACH RELATION OF TYPE ‘X->A’ ‘X’ MUST BE A SUPER KEY OR ‘A’ MUST BE A ALTERNATE KEY

College Name	College Id	College type	College Address	E-mail

[college]

Course Id	Course Name	Modules	College ID

[courses]

Faculty Id	Faculty Name	Faculty Address	Faculty phone number	Faculty E-mail	Department Number	Date of birth	Salary	College id

[college faculty]

Name	Reg No	DoB	Phone No	E-mail	Branch	College id

[student]

Block Name	Phone no	Staff	Address	Mess	College Id

[hostel]

Name	Library ID	Staff	Books	Books issue ID	College id

[library]

Dept.name	Dept.no	Faculty id	College id

[department]

Event name	Event id	Faculty id	College id

[extra curricular]

Name	No.of members	Email	Faculty id	College id

[student club]

College id	Ph1	Ph2	Ph3	Ph4	Ph5

[phone_number]

.IN THE ABOVE TABLES THE IMPLIED FUNCTIONAL DEPENDECIES ON EACH RELATION OF TYPE ‘X->A’ X IS A SUPER KEY AS X IS A PRIME ATTRIBUTE. SO THE ABOVE DECOMPOSITION OF THE RELATIONS ARE IN 1NF,2NF.3NF. THE ABOVE EXPLAINS HOW THE TABLES ARE IN 1NF ,2NF,3NF.

Functions:

Explanation:

This function helps to find out the department name when we know the department name.

```
create or replace function myfunc1(dep_no number)

return varchar2

is

depart varchar2(20);

begin

select name into depart from dept where deptno=dep_no;

return(depart);

end myfunc1;

execute:par_dep:=myfunc1(1102);
```

Procedures:

A Procedure is a subprogram unit that consists of a group of PL/SQL statements. Each procedure in Oracle has its own unique name by which it can be referred. This subprogram unit is stored as a database object.

→ Procedures are standalone blocks of a program that can be stored in the database.

→ Call to these procedures can be made by referring to their name, to execute the PL/SQl

Statements.

-→It is mainly used to execute a process in PL/SQL.

→ It can have nested blocks, or it can be defined and nested inside the other blocks or packages.

→ It contains declaration part (optional), execution part, exception handling part (optional).

→ The values can be passed into the procedure or fetched from the procedure through parameters.

→ These parameters should be included in the calling statement.

→ Procedure can have a RETURN statement to return the control to the calling block, but it cannot return any values through the RETURN statement.

→ Procedures cannot be called directly from SELECT statements. They can be called from another block or through EXEC keyword.

Stored procedures:

→This stored procedure is used to award scholarships to students according to their rank in the entrance exam.

```
set serveroutput on

create or replace procedure sproc

(temp_rankstudent number)

is

temp_rank number(10);

begin

temp_rank := temp_rankstudent;
```

```
if temp_rank between 1 and 50 then
dbms_output.put_line('you have secured 75% scholarship');
else if temp_rank between 50 and 100 then
dbms_output.put_line('you have secured 50% scholarship');
else if temp_rank between 100 and 1000 then
dbms_output.put_line('you have secured 25% scholarship');
else
dbms_output.put_line('secured no scholarship');
end if;
end if;
end if;
end
```

→This stored procedure is used to find the number of members in a club.

```
create or replace procedure myproc2(temp_club_name in
varchar2,temp_no_students out number)
is
temp_students number(4);
begin
select noofmembers into temp_students from studentclub where
name = temp_club_name;
if temp_students>80 then
dbms_output.put_line('students>80');
```

```
else
dbms_output.put_line('students<80');
end if;
temp_no_students:=temp_students;
exception
when no_data_found then
dbms_output.put_line('students not found');
end myproc2;
execute myproc2('fepsi',:par_students);
```

Triggers:

Objective:

1) Creation of Triggers

2) Implementing Triggers: Create a transparent audit system for a table Customer. The system must keep track of the records that are being deleted. The functionality being when a record is deleted the original record details and the date of operation is stored in the audit table then delete is allowed to go through.

→Row Type Triggers

Explanation:

This trigger is used to sort out the students whose attendance is less than 75%.

This trigger activates when the attendance updates and goes below 75%,

```
set serveroutput on
create or replace trigger atrig1
```

```
 after update on student_3
 for each row
declare
cursor min_attendance is  select * from  student_3;
temp_attend   min_attendance%rowtype;
begin
open min_attendance;
loop
fetch min_attendance into temp_attend;
exit when min_attendance%notfound;
if(temp_attend.attendance <'75%') then
dbms_output.put_line(temp_attend.regno);
dbms_output.put_line(temp_attend.sname);
dbms_output.put_line(temp_attend.attendance);
end if;
end loop;
close min_attendance;
end;
```

(2)

Explanation:

When a new student joins in the college or a student goes out from the college trigger activates and keeps a record of students who are leaves the college under xstudent table.

delete trigger

```
create or replace trigger mytrig1

after delete on student

for each row

begin

if deleting then

insert into
xstudent(name,regno,dob,phoneno,email,branch,collegeid)

values(:old.name,:old.regno,:old.dob,:old.phoneno,:old.email,:old.br
anch,:old.collegeid);

end if;

end;

create table xstudent(name VARCHAR2(10),regno
VARCHAR2(10),dob DATE,phoneno NUMBER(12), email
VARCHAR2(10),branch VARCHAR2(10),collegeid number(10));
```

→Statement Trigger

Explanation: This trigger restricts the user to alter the table in restricted times i.e. before 8:30 in the morning and after 6:30 in the evening. Whenever an update is happening to the library table this trigger activates.

```
create or replace trigger mytrig2

before delete or insert or update on library

BEGIN
```

```
IF (TO_CHAR(SYSDATE, 'day') IN ('sat', 'sun',’thu’)) OR
(TO_CHAR(SYSDATE,'hh:mi') NOT BETWEEN '08:30' AND '18:30')
THEN RAISE_APPLICATION_ERROR(-20500, 'library is closed');

END IF;

 END;
```