PROJECT REPORT

ON

<u>COLLEGE DATABASE</u>

Presented by

Alekhya Vanga

DATABASE MANAGEMENT SYSTEMS

(CS 470)

SPRING 2017

ORACLE ACCOUNTS:

User Name: **av145**

Password: **qu43pKNg**

NOTE: All the Tables and the data has been created and stored in the account which has User Name: av145

(Alekhya Vanga)

1) INTRODUCTION:

This is a database allows us to access the information about the college, it’s departments, staff, students, courses that it offers and track record of the fees that the student has paid or not and some more details etc. Here we will get the latest information which will be updated regularly about the students, departments, courses and staffs. This database is basically designed for assisting the administrator of an institute regarding information on the departments, courses, faculty, students and their fee. Here administrator will manage the accounts of the student and faculties, update and delete the details if needed.

ENTIETIES:

1. **College(college_ code,** college_ name).

Description: This entity gives the information about a particular college which maintains the whole database. This entity has attributes college_ code and college_ name. The college_ code is the unique attribute here. College_ name specifies the name of the college.

2. **Department**(**department_ id**, location, department_ name).

Description: This entity gives the information about different departments of which what work they do is specified. This entity has attributes department_ id which is unique and specifies a department by a ID to be remembered by the records. It also has attributes location which specifies where the department is in the college and department_ name which specifies the name of the department.

3. **Course(course_ id,** course_ name, duration).

Description: This entity gives the information about different list of courses being offered in that college. This entity has attributes Course_Id which is unique and specifies a course with a particular ID. It also has attributes Course_ Name which gives the data about the name of the course and duration specifies the amount of time the course will be finished.

4. **Teacher(teacher_ id,** teacher_name, salary, phone).

Description: This entity gives the information about the teachers and what courses they teach. This entity has attributes teacher_ id which is unique and gives information of the teacher's Id and the other attributes are teacher_ name which specifies the name of the teacher and the salary specifies the salary of a particular teacher and finally phone which specifies the phone number of a particular teacher.

5. **Student(student_ id,** student_ name, address, student_ phone).

Description: This entity gives the information about a particular student. It has attributes student_ id which is unique which every student is assigned a particular ID. It also has other attributes which are student_ name which specifies the name of the student. Address attribute specifies the house address of a student and student_ phone specifies the contact number of a particular student.

6. **Fees(receipt_ no,** amount, payment_ date).

Description: This entity gives the information about Fee that a student has paid. receipt_ no specifies the proof that the student has paid the fee or not and it is considered as unique. amount specifies the total amount the student has paid and payment_ date specifies when the student has made the payment.

RELATIONSHIPS:

1. **College to Department:** consists departments, One to Many

A single college has many number of departments. So we can say that the relationship between the college and department is One to Many.

2. **Department to Teacher:** has teachers , One to Many

A single department will have many teachers. So we can say that the relationship between department and teacher is One to Many.

3. **Teacher to Course:** teaches many, One to Many

A single teacher can teach many number of courses. So we can say that the relationship between teacher and course is One to Many

4. **Course to Student:** enrolled by student , Many to Many

There are many courses that are available for a student to choose from. We can say that different courses are being selected by different students. So the relationship between course and student is Many to Many.

5. **Student to Fees:** pays fees, One to Many

A single student pays fees. But the fees is not limited to a single because it may be like different types of fees for example tution fees and infrastructure fees. So that is the reason Fees is taken as Many. So the relationship between student and fees is considered as One to Many.

E–R DIAGRAM:

CONCEPTUAL LEVEL:

1. College(college code , college name);

Collegecode	varchar(10)
collegename	char(10)

Here the primary key is college and there is no foreign key for this table

Column Domains:

collegecode: varchar limited to 10

collegename: char limited to 10

Domain Integrity checks:

collegecode: Unique, Not Null

collegename: characters, Not Null

Functional Dependencies:

collegecode collegename

2. Department (deptid , location, deptname, collegecode);

Deptid	number(9)
Location	char(15)
Deptname	char(10)
Collegecode	varchar(10)

In this table the primary key is deptid and foreign key is collegecode.

Column Domains:

deptid: number limited to 9

location: character limited to 15

deptname: character limited to 10

collegecode: varchar limited to10

Domain Integrity checks:

deptid: Unique, Not Null

location: character, Not Null

deptname: character, Not Null

collegecode: number, Not Null

Functional Dependencies:

deptid location, deptname

3. Teacher(teacherid, teachername, salary, phone, deptid);

teacherid	number(9)
teachername	char(15)
Salary	number(10)
Phone	number(10)
deptid	number(10)

Here the primary key is teacherid and deptid is the foreign key for this table

Column Domains:

teacherid: number limited to 9

teachername: characters limited to 15

salary: number limited to 10

phone: number limited to 10

deptid: number limited to 10

Domain Integrity checks:

teacherid: Unique, Not Null

teachername: characters, Not Null

salary: number, Not Null, value must be greater than 0
phone: number, Not Null

deptid: number, Not Null, value must be greater than 0

Functional Dependencies:

teacherid teachername, salary, phone

4. Course(courseid, coursename, duration, teacherid);

courseid	varchar(20)
coursename	char(10)
duration	number(10)
teacherid	number(9)

Here the primary key is courseid and the foreign key is teacherid.

column domains:-

courseid: varchar limited to 20

coursename: char limited to 10

duration: number limited to 10
teacherid: number limited to 9

domain integrity checks:

courseid: Unique, Not Null

coursename: characters, Not Null

duration: number, Not Null, value must be greater than 0 teacherid: number, Not Null

Functional Dependencies:

courseid coursename, duration

5. **Student**(studentid, studentname, address, studentphone);

Studentid	number(9)
Studentname	char(9)
Address	varchar(15)
studentphone	number(10)

Here studentid is the primary key and there are no foreign keys,

column domains:

studentid: number limited to 9

studentname: character limited to 9

address: varchar limited to 15

studentphone: number limited to 10

domain integrity checks:

studentid: Unique, Not Null

studentname: characters, Not Null

address: varchar, Not Null

studentphone: number, Not Null

Functional dependencies:-
studentid studentname, address, studentphone

Fees(receiptno, amount, paymentdate, studentid)

Receiptno	varchar(19)
Amount	number(10)
Paymentdate	date
studentid	number(10)

Here receiptno is the primary key and studentid is the foreign key,

column domains:

receiptno: varchar limited to 19

amount: number limited to10

paymentdate: date

studentid: number limited to 10

domain integrity checks:

receiptno: Unique, Not Null

amount: number, Not Null, value must be greater than 0

payment date : date, Not Null

studentid: number, Not Null

Functional dependencies:-

receiptno amount, paymentdate

6. **Enrolledby**(courseid, studentid).

Coursed	varchar(20)
Studentid	number(9)

Here courseid and studentid are both primary keys and even foreign keys

column domains:-

courseid : varchar limited to 20

studentid : number limited to 9

domain integrity checks:-

courseid : Unique, Not Null

studentid : Unique, Not Null

Functional Dependencies:

courseid studentid

EXTERNAL VIEW:

We are going to represent the users in the column and the tables in the rows as follows

TABLE NAME	**ADMINISTRATOR**	**HEAD OF THE DEPARTMENT**	**STUDENT**
college	Select, Insert, Update	N/A	N/A
department	Select, insert, delete	N/A	N/A
teacher	N/A	Select, insert, delete, update	N/A
course	N/A	N/A	N/A
student	N/A	Select, insert, delete	Select
fees	Select, Insert, Update	N/A	N/A
enrolledby	N/A	N/A	N/A

INTERNAL VIEW:

1. **Simple query:**

Retrieve the location deptid is '3285'

Query:

```
SELECT location
FROM   dept_hash_tbl
WHERE  deptid = 3285;
```

File structure used: Hashing is used for the simple query

2. **Built in query:**

Retrieve the teacher name having maximum salary

Query:

```
SELECT teachername
 FROM   teacher
 WHERE  salary = (SELECT Max(salary)
                  FROM   teacher);
```

3. **Range query:**

Retrieve receipt no who paid the fee amount between 1000-1400

Query:

```
SELECT receiptno
FROM   fees
WHERE  amount BETWEEN 1000 AND 1400;
```

4. **Join query:**

Retrieve the fee paid by a particular student (your answer should contain studentname and receiptno)

Query:

```
SELECT studentname,
       receiptno
FROM   student,
        Fees
WHERE  student.studentid = fees.studentid;
```

File structure used:

Clustered files is used in join query.

5. Order by:

Arrange teacher table in increasing order of their salary

Query:

```
SELECT teacherid,
       teachername
FROM   teacher
ORDER  BY salary ASC;
```

File structure used:

Clustered-B Tree is used in Order by function.

DATA DICTIONARY:

1. college:

Attribute Name	**Type**	**Not Null**
Collegecode	varchar(10)	Not null
Collegename	char(10)	Not null

Primary Key: collegecode

Foreign Key: N/A

Privileges: select,insert,update to Administrator

2. department:

Attribute Name	**Type**	**Not Null**
departmentid	number(9)	Not null
Location	char(15)	Not null
Deptname	char(10)	Not null
Collegecode	varchar(10)	Not null

Primary Key: ID
Foreign Key: collegecode

Privileges: select,insert,delete to Administrator

3. teacher:

Attribute Name	**Type**	**Not Null**
teacherid	varchar(30)	Not null
teachername	varchar(10)	Not null
salary	varchar(10)	Not null
phone	number(10)	Not null
deptid	number(9)	Not null

Primary Key: teacherid

Foreign Key: deptid

Privileges: select, insert, delete, update to Head of the Department

4. course

Attribute Name	**Type**	**Not Null**
courseid	varchar(20)	Not null
coursename	char(10)	Not null
duration	number(10)	Not null
teacherid	number(9)	Not null

Primary Key: courseid

Foreign Key: teacherid

Privileges: N/A

5. student:

Attribute Name	**Type**	**Not Null**
studentid	number(9)	Not null
studentname	char(9)	Not null
address	varchar(15)	Not null
studentphone	number(10)	Not null

Primary Key: studentid

Foreign key: N/A

Privileges: select,insert,delete to Head of the department

select to Student

6. fees

Attribute Name	**Type**	**Not Null**
receiptno	varchar(19)	Not null
Amount	number(10)	Not null
Paymentdate	date	Not null
Studentid	number(9)	Not null

Primary Key: receiptno

Foreign key: studentid

Privileges: select,insert,update,delete to Administrator

7. enrolledby

Attribute Name	**Type**	**Not Null**
courseid	varchar(19)	Not null
studentid	number(9)	Not null

Primary Key: courseid, studentid

Foreign key: courseid, studentid

Privileges: N/A