Database Management Design Project

MARIST HOUSING MOVE OUT

12/01/2014

Submitted By:
Siting Wang

Table of Contents

Table of Contents

Executive Summary 3
Entity Relationship Diagram 4
Objective 4
Entities 5
STUDENT TABLE 5
BUILDING TABLE 6
VISIT TABLE 7
VISITOR TABLE 9
STAFF TABLE 10
MOVEOUT TABLE 11
Views 12
Reports and Queries: 13
Stored procedures 14
Triggers 14
Security 16
Implementation Notes and Known Problems: 16
Future Enhancements 16

SITING WANG

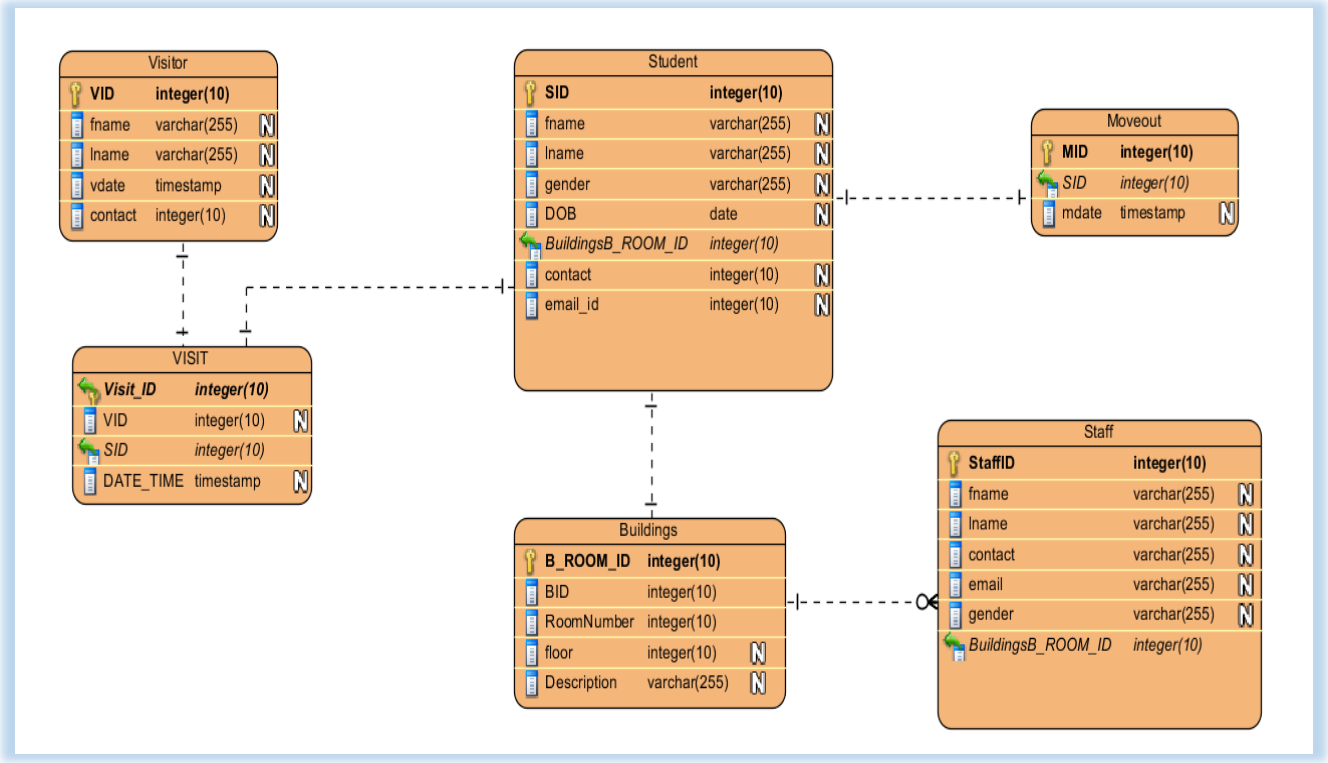
Executive Summary

Housing selection happens every academic year. At the end of every school year, Marist resident students are require to move out of the building with their personal belongings, and will replace with new resident areas in the following year. When the moving out day approached, it brings a lots of work for housing staff and security officers. Housing staff need to keep track on the date and time that students are leaving; security officers need to get the information of the non-Marist students and faculty who are entering the building. With all the considering, creating a database with all the information will be easy for them, and it might reduce their quantities of work.

This database is designed to keep track of the people who are enter the building when they swipe their ID card, and for those who need to sign in with security people in order to get in the building. This database is also designed to keep track the date that student move out, so the housing staff can check with students when they leave.

SITING WANG

Entity Relationship Diagram



Objective

This database is needed to keep track the information of students who are living in the building and planning to move out the building at the end of school year. Table student, building, room, staff are benefit for housing people, and table visitor is benefit for security officers to track on people who are enter the building.

Entities

STUDENT TABLE

Description:

This table is needed to keep track the information of students who are living in the campus housing.

Create Statement:

```
--student
                Drop table if exists Student;
                create table Student
                (
                 sid int primary key,
                 fname char(20),
                 lname text,
                 gender text,
                 dob date,
                 contact varchar(20),
                 email varchar(20),
                 BuildingsB_ROOM_ID int not null
                 );
```

Insert Statements:

```
--insert into student
insert into student(sid,fname,lname,gender,dob,contact,email,BuildingsB_ROOM_ID)
        values('20048890','kevel','Kim','male','02-04-1995','888-888-8888','123@marist.edu','1');
insert into Student(Sid,fname,lname,gender,dob,contact,email,BuildingsB_ROOM_ID)
        values('20024482','kan','Smith','male','12-04-1995','111-111-1111','234@marist.edu','1');
insert into student(sid,fname,lname,gender,dob,contact,email,BuildingsB_ROOM_ID)
        values('20045650','Ada','molon','female','07-08-1994','222-222-2222','456@marist.edu','2');
insert into Student(Sid,fname,lname,gender,dob,contact,email,BuildingsB_ROOM_ID)
        values('20048824','Kyle','Bend','male','06-04-1993','333-333-3333','678@marist.edu','1');
insert into student(sid,fname,lname,gender,dob,contact,email,BuildingsB_ROOM_ID)
        values('20048678','Susan','Lee','female','11-25-1994','333-333-3313','672@marist.edu','3');
insert into Student(Sid,fname,lname,gender,dob,contact,email,BuildingsB_ROOM_ID)
        values('20040067','Caroline','Sean','female','04-15-1994','333-333-3331','674@marist.edu','4');
insert into student(sid,fname,lname,gender,dob,contact,email,BuildingsB_ROOM_ID)
        values('20048867','Kaitlyn','Fold','female','10-10-1992','333-333-3332','675@marist.edu','2');
insert into Student(Sid,fname,lname,gender,dob,contact,email,BuildingsB_ROOM_ID)
        values('20024483','Kayla','Bean','female','03-19-1995','333-333-3334','679@marist.edu','3');
```

Sample Data:

Data Output | Explain | Messages | History

	sid integer	fname character(20)	lname text	gender text	dob date	contact character varying(20)	email character varying(20)	buildingsb_room_id integer
1	20048890	kevel	Kim	male	1995-02-04	888-888-8888	123@marist.edu	1
2	20024482	kan	Smith	male	1995-12-04	111-111-1111	234@marist.edu	1
3	20045650	Ada	molon	female	1994-07-08	222-222-2222	456@marist.edu	2
4	20048824	Kyle	Bend	male	1993-06-04	333-333-3333	678@marist.edu	1
5	20048678	Susan	Lee	female	1994-11-25	333-333-3313	672@marist.edu	3
6	20040067	Caroline	Sean	female	1994-04-15	333-333-3331	674@marist.edu	4
7	20048867	Kaitlyn	Fold	female	1992-10-10	333-333-3332	675@marist.edu	2
8	20024483	Kayla	Bean	female	1995-03-19	333-333-3334	679@marist.edu	3
9	20048892	Unknown	Kim	male	1995-02-04	888-888-8888	123@marist.edu	1

Functional Dependencies:
Sid->fname, lname, gender, birth, contact, email.

BUILDING TABLE

Description:
This building table consists of the freshmen buildings, and lists of buildings, floor, and room number.

Create a sequence for Building_ROOM_ID column:

```
CREATE SEQUENCE SEQ_BuildingsB_ROOM_ID;
```

Create Statement:

```
--building
Drop table if exists Building;
create table Building
(
BuildingsB_ROOM_ID int DEFAULT NEXTVAL('SEQ_BuildingsB_ROOM_ID') primary key ,
bid char(3),
Description text not null,
roomNumber int,
floor char(20)
);
```

Insert Statements:

```
--insert into building
insert into Building(bid,Description,floor,roomNumber)
        values('b01','Leo','3','305');
insert into Building(bid,Description,floor,roomNumber)
        values('b01','Leo','2','215');
insert into Building(bid,Description,floor,roomNumber)
        values('b03','Marian','2','215');
insert into Building(bid,Description,floor,roomNumber)
        values('b02','Champagnat','2','204');
insert into Building(bid,Description,floor,roomNumber)
        values('b04','Sheahan','2','204');
insert into Building(bid,Description,floor,roomNumber)
        values('b04','Sheahan','3','311');
insert into Building(bid,Description,floor,roomNumber)
        values('b04','Sheahan','3','311');
```

Sample Data:

	buildingsb_room_id integer	bid character(3)	description text	roomnumber integer	floor character(20)
1	19	b01	Leo	305	3
2	20	b01	Leo	215	2
3	21	b03	Marian	215	2
4	22	b02	Champagnat	204	2
5	23	b04	Sheahan	204	2
6	24	b04	Sheahan	311	3
7	25	b04	Sheahan	311	3

Functional Dependencies:

BuildingsB_ROOM_ID-> BID, building, Description, floor, roomNumber

VISIT TABLE

Description:

This table contain the information of the visits has been paid to the students staying in the housing.

Create Sequence:

```
CREATE SEQUENCE SEQ_VISIT_ID;
```

SITING WANG

Create Default current date and time function:

```
create function public.getdate() returns timestamptz
       stable language sql as 'select now()';

       select * from public.getdate();
```

Create Statement:

```
--Visit
drop table if exists Visit;
create table Visit
(
visit_ID int DEFAULT NEXTVAL('SEQ_VISIT_ID') primary key,
VID int not null,
SID int not null,
DATE_TIME TIMESTAMP default public.getdate()
);
```

Insert Statements:

```
insert into Visit(VID,SID)
         values(1,'20048890');
insert into Visit(VID,SID)
         values(2,'20045650');
insert into Visit(VID,SID)
         values(3,'20048890');
insert into Visit(VID,SID)
         values(2,'20045650');
insert into Visit(VID,SID)
         values(1,'20048890');
```

Sample Data:

	visit_id integer	vid integer	sid integer	date_time timestamp without time zone
1	1	1	048890	2014-12-17 20:14:55.693
2	2	2	045650	2014-12-17 20:14:55.693
3	3	3	048890	2014-12-17 20:14:55.693
4	4	2	045650	2014-12-17 20:14:55.693
5	5	1	048890	2014-12-17 20:14:55.693

Functional Dependency:

Visit_id->Vid, Sid, date_time

SITING WANG

VISITOR TABLE

Description:

This visitor table lists visitor's first name, last name, visit date, visit time. With more information of visitors, it will be easy for security officers to keep track on.

Create Statement:

```
--visitor
drop table if exists Visitor cascade;
create table Visitor
(
vid int primary key,
fname text,
lname text,
contact varchar (15)
);
```

Insert Statements:

```
--insert into visitor
insert into Visitor(vid,fname,lname,contact)
        values('20047387','Dan','Mathew','111-222-3333');
insert into Visitor(vid,fname,lname,contact)
        values('20037893','Kenny','Ocean','111-222-3332');
insert into Visitor(vid,fname,lname,contact)
        values('20043453','John','Miller','111-222-3331');
insert into Visitor(vid,fname,lname,contact)
        values('20046535','Dannel','Mcbrain','111-222-3334');
insert into Visitor(vid,fname,lname,contact)
        values('20045643','Bryn','kushi','111-222-3335');
insert into Visitor(vid,fname,lname,contact)
        values('20042465','Sally','Ross','111-222-3336');
insert into Visitor(vid,fname,lname,contact)
        values('20043465','Roza','Mura','111-222-3337');
insert into Visitor(vid,fname,lname,contact)
        values('20049323','William','Curry','111-222-3338');
```

Sample Data:

	vid integer	fname text	lname text	contact character varying(15)
1	20047387	Dan	Mathew	111-222-3333
2	20037893	Kenny	Ocean	111-222-3332
3	20043453	John	Miller	111-222-3331
4	20046535	Dannel	Mcbrain	111-222-3334
5	20045643	Bryn	kushi	111-222-3335
6	20042465	Sally	Ross	111-222-3336
7	20043465	Roza	Mura	111-222-3337
8	20049323	William	Curry	111-222-3338

Functional Dependency:
vid->vid,fname,lname,contact

STAFF TABLE

Description:
This table lists the staff's first name, last name, gender, building that they in charge, and all related information.

Create Statement:

```
--Resident staff
drop table if exists Staff;
create table Staff
(
 staffid char(20) primary key,
 fname char(20),
 lname text,
 contact varchar(15),
 email varchar(20),
 gender text,
 Building char(3)
 );
```

Insert Statements:

```
--inset into staff
insert into Staff(staffid,fname,lname,contact,email,gender,Building)
        Values('20048890','kelly','McDough','111-222-3332','123@gmail.com','female','b01');
insert into Staff(staffid,fname,lname,contact,email,gender,Building)
        Values('20034532','Paggy','Smith','111-222-3332','123@gmail.com','female','b02');
insert into Staff(staffid,fname,lname,contact,email,gender,Building)
        Values('20040053','Wendy','Sung','111-222-3332','123@gmail.com','female','b03');
insert into Staff(staffid,fname,lname,contact,email,gender,Building)
        Values('20022244','Warren','Park','111-222-3332','123@gmail.com','male','b01');
insert into Staff(staffid,fname,lname,contact,email,gender,Building)
        Values('20048888','kimi','Yuk','111-222-3332','123@gmail.com','male','b04');
insert into Staff(staffid,fname,lname,contact,email,gender,Building)
        Values('20024234','Lucas','Zain','111-222-3332','123@gmail.com','male','b02');
select*
from Staff;
```

Sample Date:

	staffid character(20)	fname character(20)	lname text	contact character varying(15)	email character varying(20)	gender text	building character(3)
1	20048890	kelly	McDough	111-222-3332	123@gmail.com	female	b01
2	20034532	Paggy	Smith	111-222-3332	123@gmail.com	female	b02
3	20040053	Wendy	Sung	111-222-3332	123@gmail.com	female	b03
4	20022244	Warren	Park	111-222-3332	123@gmail.com	male	b01
5	20048888	kimi	Yuk	111-222-3332	123@gmail.com	male	b04
6	20024234	Lucas	Zain	111-222-3332	123@gmail.com	male	b02

Functional Dependency:
Staid->fname, lname, contact, email, gender, building

MOVEOUT TABLE

Description:
This table lists the movement, moveout date, and move time of students, so staff can check with students.

Create Statement:

```
drop table if exists Moveout;
create table Moveout
(
mid char(20) not null primary key,
SID int  REFERENCES student(sid),
mdate timestamp default public.getdate()
);
```

Insert Statements:

```
--insert into moveout
insert into Moveout(mid,sid)
        values('m01','20048890');
insert into Moveout(mid,sid)
        values('m02','20024483');
insert into Moveout(mid,sid)
        values('m03','20040067');
insert into Moveout(mid,sid)
        values('m04','20040067');
insert into Moveout(mid,sid)
        values('m05','20048890');
insert into Moveout(mid,sid)
        values('m06','20045650');
insert into Moveout(mid,sid)
        values('m10','20045650');
```

Sample Date:

	mid character(20)	sid integer	mdate timestamp without time zone
1	m01	20048890	2014-12-20 15:27:12.094
2	m02	20024483	2014-12-20 15:27:12.094
3	m03	20040067	2014-12-20 15:27:12.094
4	m04	20040067	2014-12-20 15:27:12.094
5	m05	20048890	2014-12-20 15:27:12.094
6	m06	20045650	2014-12-20 15:27:12.094
7	m10	20045650	2014-12-20 15:27:12.094

Functional Dependency:
mid->fname,lname,mdate,mtime

Views

Description:

This view displays anyone who visit the students.

Create Statement:

```
--view
        drop view if exists visit_info;
        create view visit_info
        AS
                select distinct a.fname||' '||a.lname as "VISITOR",
                a.contact,s.fname||' '||s.lname as "STUDENT",
                d.Description as "VISIT BUILdING",d.roomNumber,d.floor,b.DATE_TIME
                from Visitor a,visit b,student s,Building d
                where a.VID = b.VID
                and b.SID = s.SID
                and s.BuildingsB_ROOM_ID = d.BuildingsB_ROOM_ID;


                 select *
                 from visit_info;
```

Sample data:

Data Output | Explain | Messages | History

	VISITOR text	contact character varying(15)	STUDENT text	VISIT BUILdING text	roomnumber integer	floor character(20)	date_time timestamp without time zone
1	John Miller	111-222-3331	Ada molon	Leo	215	2	2014-12-20 11:06:09.2
2	Roza Mura	111-222-3337	kevel Kim	Leo	305	3	2014-12-20 11:06:09.2
3	Bryn kushi	111-222-3335	Ada molon	Leo	215	2	2014-12-20 11:06:09.2
4	Dan Mathew	111-222-3333	kevel Kim	Leo	305	3	2014-12-20 11:06:09.2

Reports and Queries:

This query gets the information of students who moved out from the housing.

Create Statement:

```
-- to find who is moving out and from which building and room number
 select s.sid,s.fname as "FIRST NAME",lname AS "LAST NAME",b.description,
 b.roomNumber as "ROOM VACANT",b.floor, m.mdate as "MOVING ON"
 from Student s,moveout m,building b
 where s.BuildingsB_ROOM_ID = b.BuildingsB_ROOM_ID
 and s.sid = m.sid
 and b.Description in ('Sheahan','Leo','Marian')
 and s.gender='female';
```

Sample Data:

Data Output | Explain | Messages | History

	sid integer	FIRST NAME character(20)	LAST NAME text	description text	ROOM VACANT integer	floor character(20)	MOVING ON timestamp without time zone
1	20024483	Kayla	Bean	Marian	215	2	2014-12-20 15:27:12.094
2	20045650	Ada	molon	Leo	215	2	2014-12-20 15:27:12.094
3	20045650	Ada	molon	Leo	215	2	2014-12-20 15:27:12.094

Create Statement:

```
-- to find the employees mamaging the building or working staff
select distinct b.description,s.staffid,s.fname,s.lname,s.contact,
s.email,s.gender,b.bid
from building b, staff s
where b.bid = s.Building
and b.bid = 'b01';
```

SITING WANG

Stored procedures

Count the total number of students.

Create Statement:

```
--store procedures,Count the total number of students.
CREATE OR REPLACE FUNCTION studentcount() RETURNS INT AS $$
DECLARE
        studentcount INT;
BEGIN
        SELECT COUNT(*)INTO studentcount
        FROM student;
        RETURN studentcount;
END;
$$
language plpgsql;
SELECT studentcount();
```

Sample Data:

Output pane

Data Output | Explain | Messages | History

	studentcount integer
1	8

Triggers

```
--trigger-

        /*function to perform a task during trigger call, if the name is null or
        blank, it will be replaced by 'unknown'*/
        CREATE OR REPLACE FUNCTION unknown_first_name() RETURNS trigger AS $my_trigger$ BEGIN
        IF NEW.fname IS NULL OR NEW.fname = '' THEN NEW.fname := 'Unknown';
        END IF;
        RETURN NEW;
        END;
        $my_trigger$
        LANGUAGE plpgsql;

        -- trigger for student table
        CREATE TRIGGER my_trigger
        BEFORE INSERT ON student
        FOR EACH ROW
        EXECUTE PROCEDURE unknown_first_name();
```

Here, we are not giving any value to fname of the student table, the trigger will automatically set the null value to “Unknown”

```
insert into student(sid,lname,gender,dob,contact,email,BuildingsB_ROOM_ID)
              values('20048892','Kim','male','02-04-1995','888-888-8888','123@marist.edu','1');

select * from student;
```

Sample output:

Output pane

Data Output | Explain | Messages | History

	sid integer	fname character(20)	lname text	gender text	dob date	contact character varying(20)	email character varying(20)	buildingsb_room_id integer
1	20048890	kevel	Kim	male	1995-02-04	888-888-8888	123@marist.edu	1
2	20024482	kan	Smith	male	1995-12-04	111-111-1111	234@marist.edu	1
3	20045650	Ada	molon	female	1994-07-08	222-222-2222	456@marist.edu	2
4	20048824	Kyle	Bend	male	1993-06-04	333-333-3333	678@marist.edu	1
5	20048678	Susan	Lee	female	1994-11-25	333-333-3313	672@marist.edu	3
6	20040067	Caroline	Sean	female	1994-04-15	333-333-3331	674@marist.edu	4
7	20048867	Kaitlyn	Fold	female	1992-10-10	333-333-3332	675@marist.edu	2
8	20024483	Kayla	Bean	female	1995-03-19	333-333-3334	679@marist.edu	3
9	20048892	Unknown	Kim	male	1995-02-04	888-888-8888	123@marist.edu	1

Date trigger:

```
--although we have already used default getdate() function, but we can create trigger for the same task
CREATE OR REPLACE FUNCTION current_date_set() RETURNS trigger AS $date_trigger$ BEGIN
IF NEW.mdate IS NULL OR NEW.mdate = '' THEN NEW.mdate := current_date;
END IF;
RETURN NEW;
END;
$date_trigger$
LANGUAGE plpgsql;

CREATE TRIGGER date_trigger
BEFORE INSERT ON Moveout
FOR EACH ROW
EXECUTE PROCEDURE current_date_set();

drop trigger date_trigger on Moveout;

SELECT CURRENT_DATE;
```

SITING WANG

Security

There are two types of users identified for the database.

1. Database admins are able to change, update, and maintain all the tables on the database.

   “*CREATE ROLE_admin*
   GRANT SELECT, INSERT, UPDATE, DELETE
   ON ALL TABLES
   TO db_admin;”

2. Users are only allow to see the database, but not able to modify anything.

   “*CREATE ROLE user*
   GRANT SELECT
   ON ALL TABLE IN SCHMA USER
   TO user;”

Implementation Notes and Known Problems:

Implementation seems pretty good so far. With all the databases, admin will able to know more about the move out date of students. This database could be more interesting if admin add more items into each table, and add more tables. Since there are all sample database, it would be more useful if having the real database, because it will avoid the redundancies. Also, in the database, it separate the student and staff which means in this case, student do not have the role of being staff. However, in many real situation, student can also play a role as staff. So if there is student who is also staff, there might be more problems.

Future Enhancements

Each table could have include more specific items. For the staff table, it would be better to point out that they are more referring to resident assistant or resident director. Plus, it will be better to add a table to indicate the connection between staff and student, because student can also be a staff in some case.