Group : 06

Submitted to:

SajjadurRahman
Lecturer, Dept. of CSE

Johra Muhammad Moosa
Lecturer, Dept. of CSE

Submitted by:

Shishir Kumar Sarkar	(1005040)
Yahya Ahmed Sharif	(1005048)
Ehsan-Ul-Haque	(1005052)
Mahmood Ahmed	(1005055)
Nayeem Islam	(1005060)

Introduction:

Our present hall management system is way too backdated. This is not suitable for both the students and the hall management authority. To keep up with this current digital world we need an upgraded well design information system.

Subsystems:

1. Computerized Hall Seat Allocation
2. Hall Mess management System
3. Virtual Notice Board & Complain box
4. Room Maintenance System

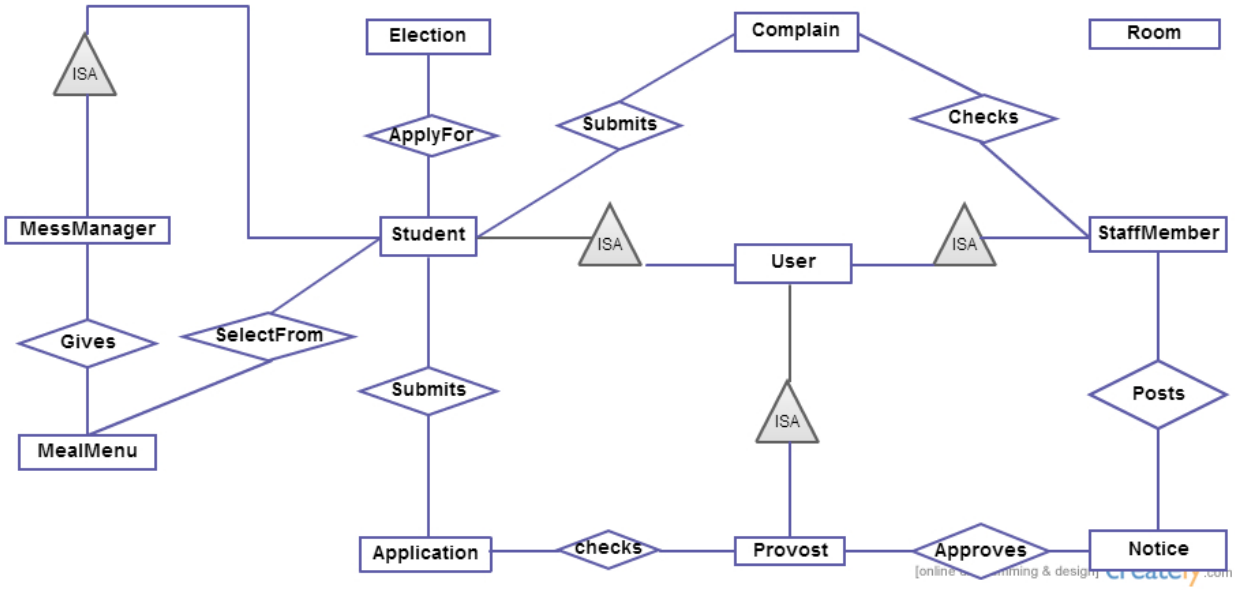

Figure : Entity Relationship Diagram

Here is the complete entity relationship diagram for hall management system which shows that students, provost, staff members are users and mess manager is a student. Which shows a student can submit application for seat allocation and provost checks it and after approving it gives a notice. A student can also apply for mess manager and all students have to participate in the election to get their desired mess manager. A mess manager gives a meal menu which is updated every day and the students can select meals from the meal menu. Also a student can submit a complain using the virtual complain box after logging into account then staff member checks the complain.

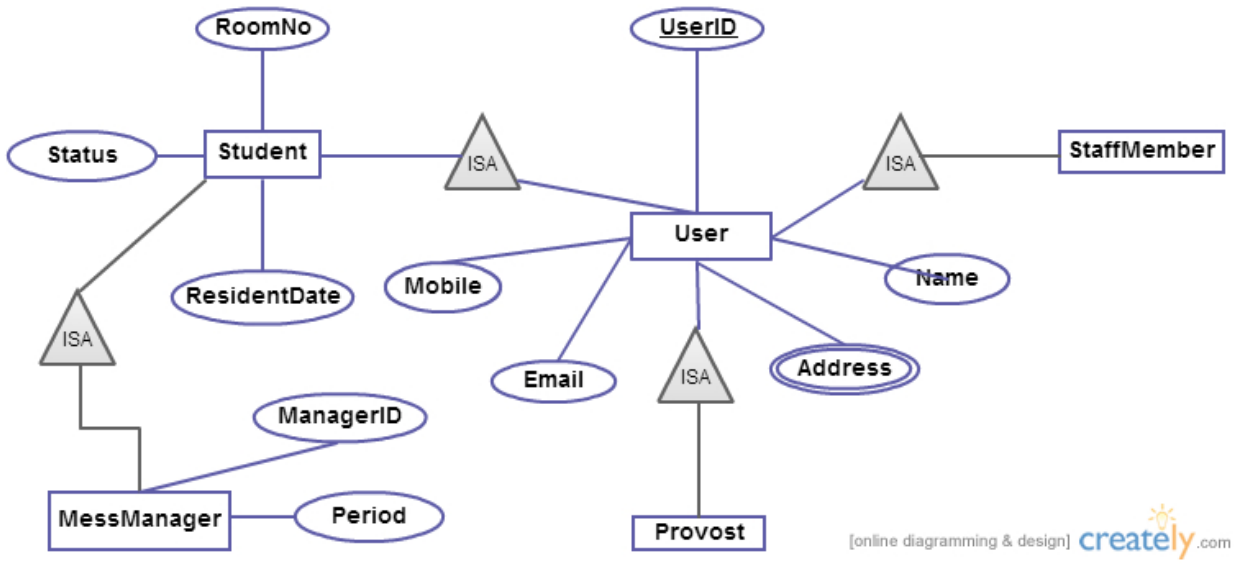

Figure : Entity Relationship Diagram

Here is the in depth analysis. A user has attributes userID, name, Mobile, Email, Address. By the entity student one can know his RoomNo, Status, ResidentDate (the date when he joined first). A mess manager is a student but he must have a ManagerID so that one can identify a mess manager and the time duration when he started working as a mess manager and stopped working.

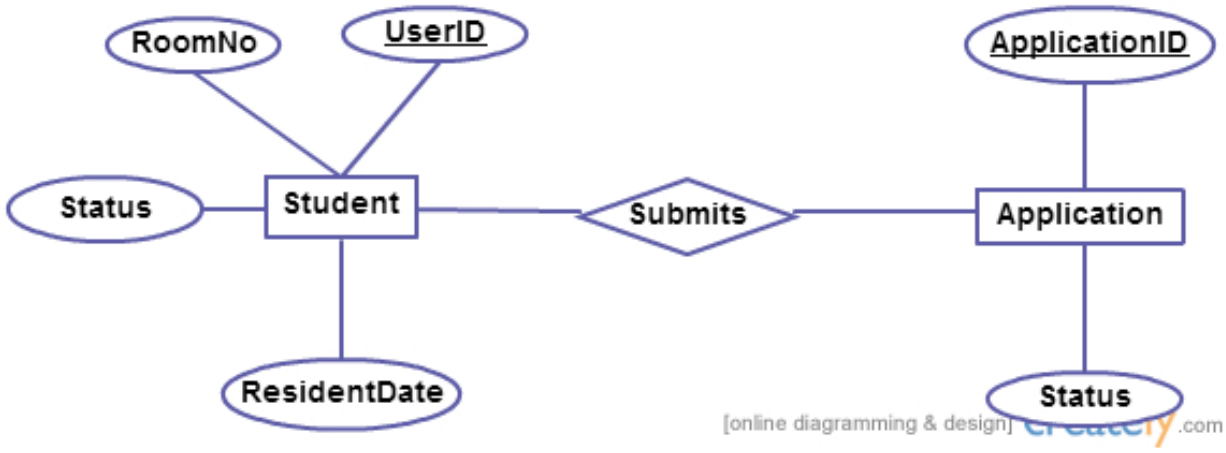

Figure : Entity Relationship Diagram

This ER Diagram shows that firstly a student must login to his account using his UserID and password. After logging into his account he will get an application form which he has to fill up for allocation. An application must have an ApplicationID which is a primary key for the Application entity and an attribute status which will show is it received yet or not.

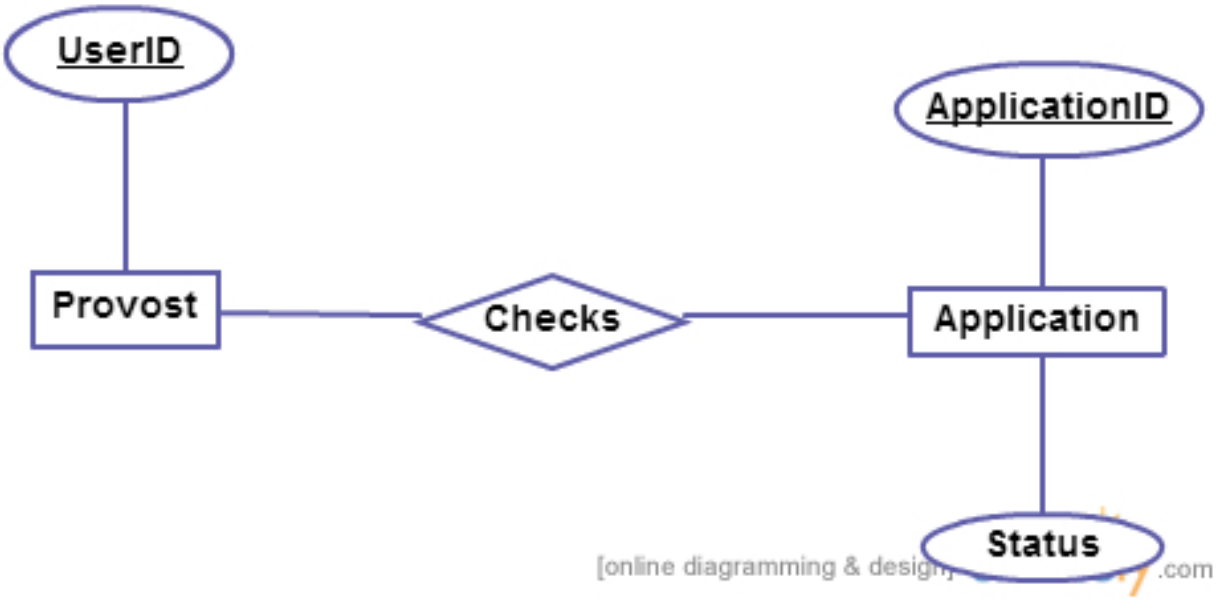

Figure : Entity Relationship Diagram

In this ER Diagram provost will login to his account using his UserID and password. After logging into his account he will search for application. If there is any application available then he will check the application and status will be set to checked. He will check for further applications until none is left.

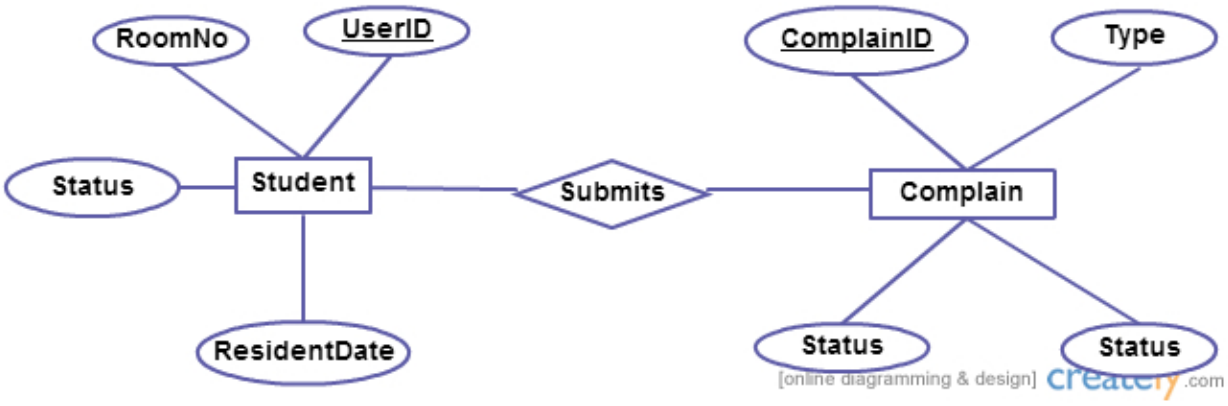

Figure : Entity Relationship Diagram

This diagram is about submitting complain in complain box. A student must login to his account to his account for submitting complain and he must be a resident student. He can submit any kind of problem related to his room or problem with the meal menu etc. Complain must have a ComplainIDby which complain can be identified and the student's UserID, RoomNo can also be identified using this relation.

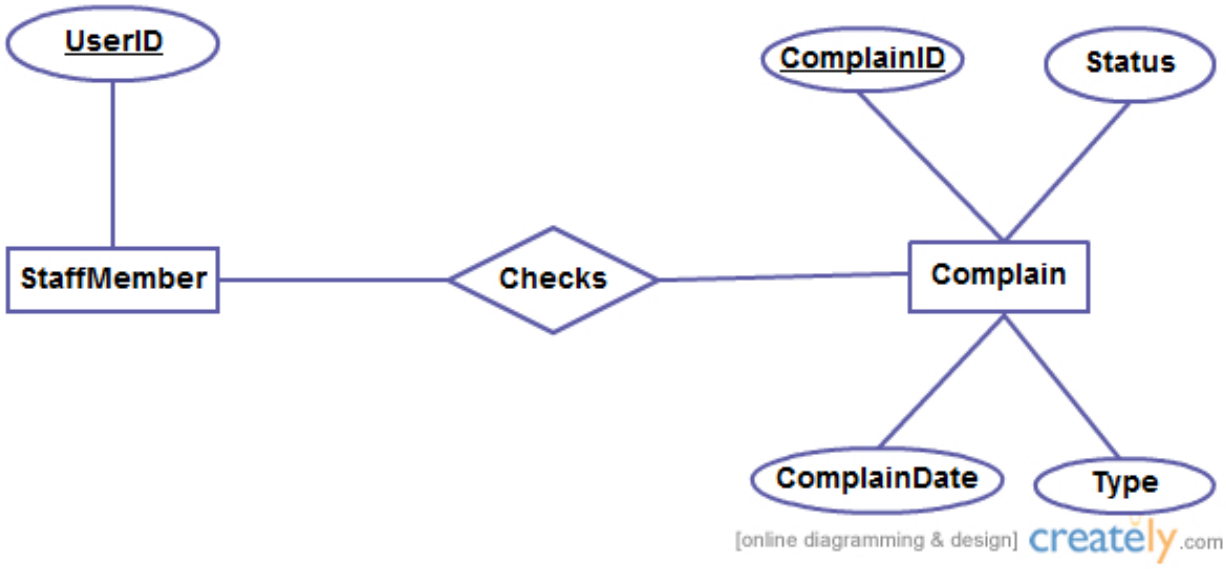

Figure : Entity Relationship Diagram

The previous ER Diagram was about student's submitting complain. When complain will arrive into complain box staff member can check complain by simply logging into his account. He can check when complain has been submitted by a student and the type of complain. If it's very old and type is related to something serious he will take necessary steps to solve the problem. After solving it he will set the status to solved.

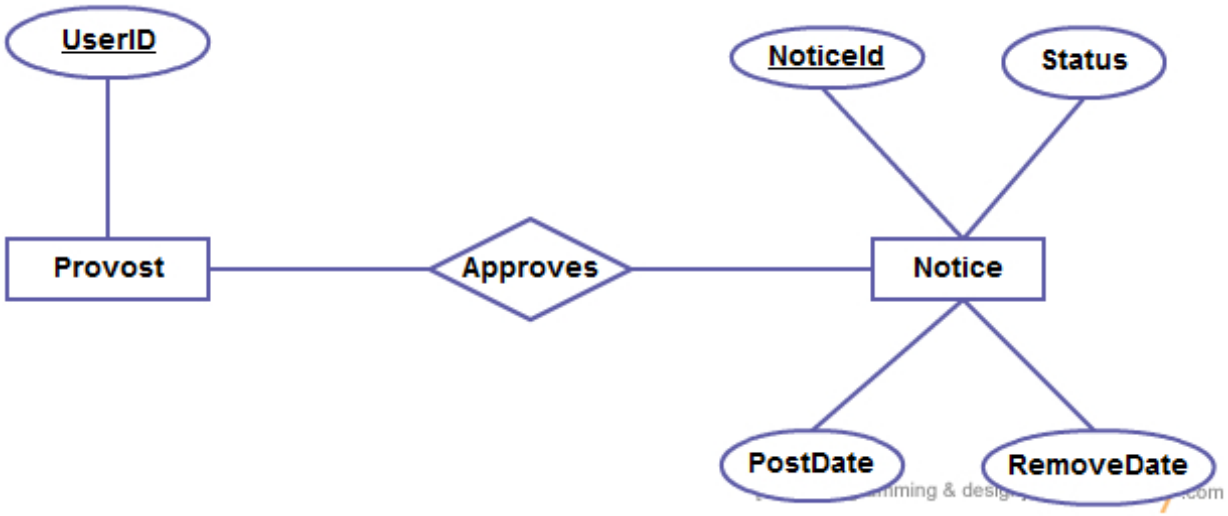

Figure : Entity Relationship Diagram

A provost can publish a notice or remove it. Firstly he needs to login to his account then he needs check if a notice is post able or not if he thinks it has ability to be posted then he approves the notice. A notice must have a NoticeID which is the primary key for the Notice entity.

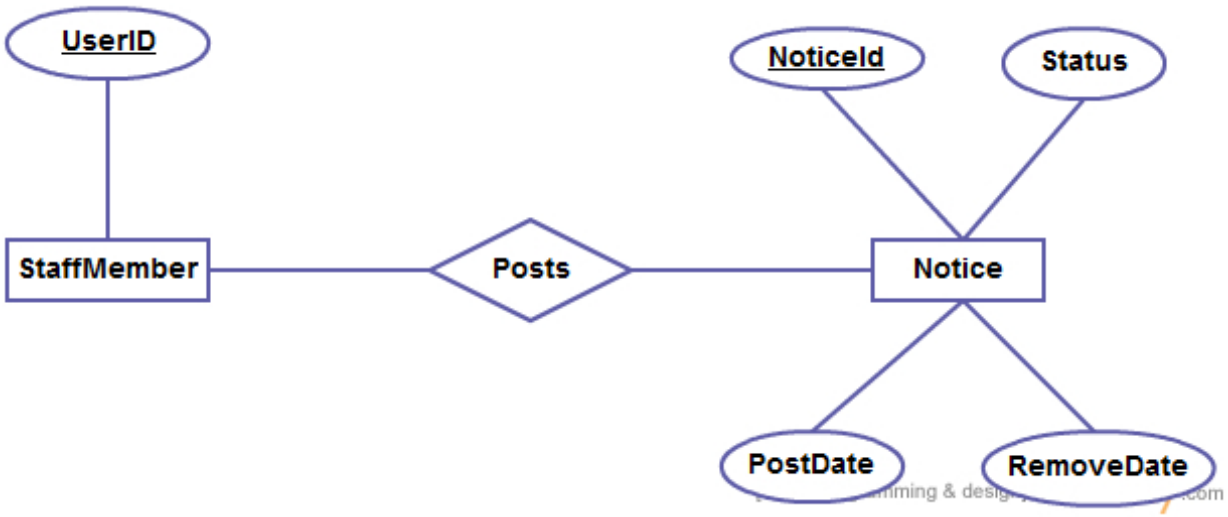

Figure : Entity Relationship Diagram

After getting approval from the provost the notice will be posted in the virtual notice board. The time will be also updated when the notice has been posted and staff member can also delete previous notices which is too old. A remove date will be also updated when the post will be removed. Sometimes it is not necessary for staff members to delete a notice. There will be a maximum time duration for a notice after that the notice will be removed automatically. In this case a remove date also will be updated. A notice can have two kind of status available or unavailable.

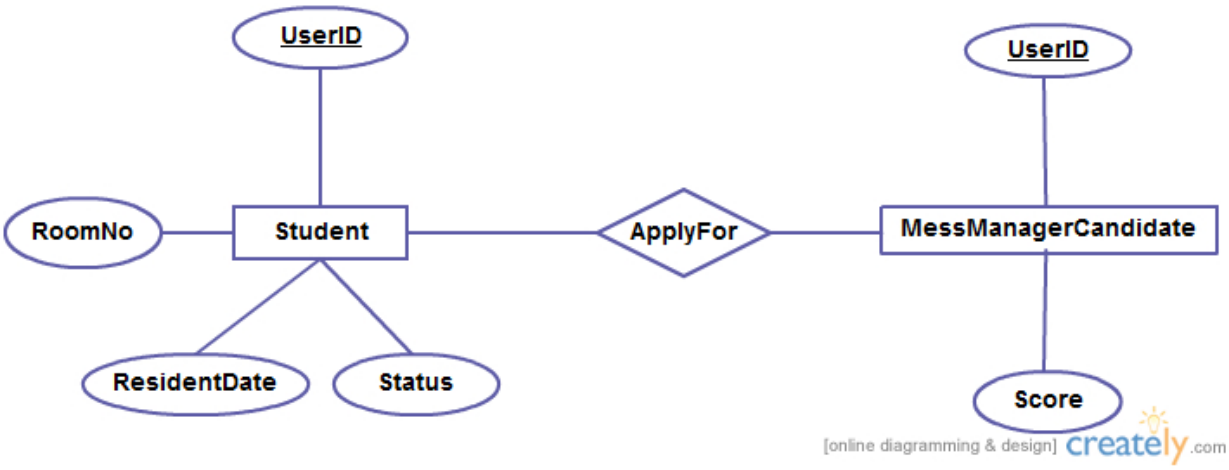

Figure : Entity Relationship Diagram

A student can apply for becoming mess manager. Firstly he needs to be approved by provost. After being approved by provost mess manager candidates will stand for an election where all the students can submit his vote to his desired mess manager. Then the score will be kept in the system. The provost will finally announce who has won the contest and the winner will be chosen as the mess manager.

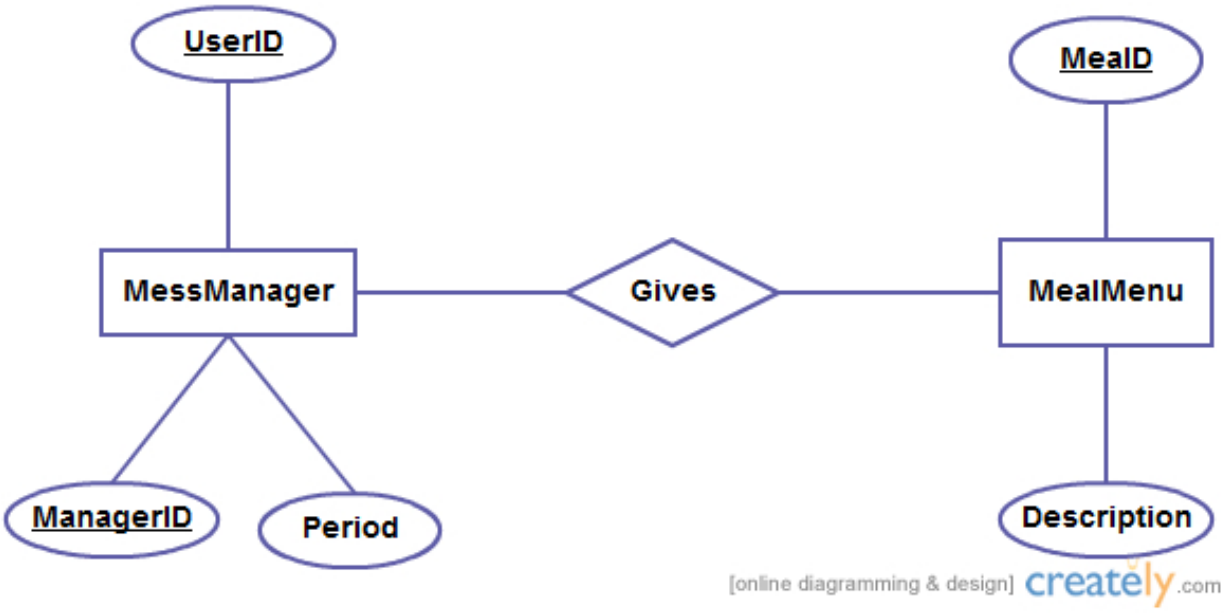

Figure : Entity Relationship Diagram

Mess manager has to update the meal menu everyday so that student can see what is available for his dinner or lunch. A meal must have a MealID by which the meal can be identified. It also must have some description by which a student can know how it tastes or the quality of this meal. If a student doesn't choose a meal menu a default meal will be given to the student.

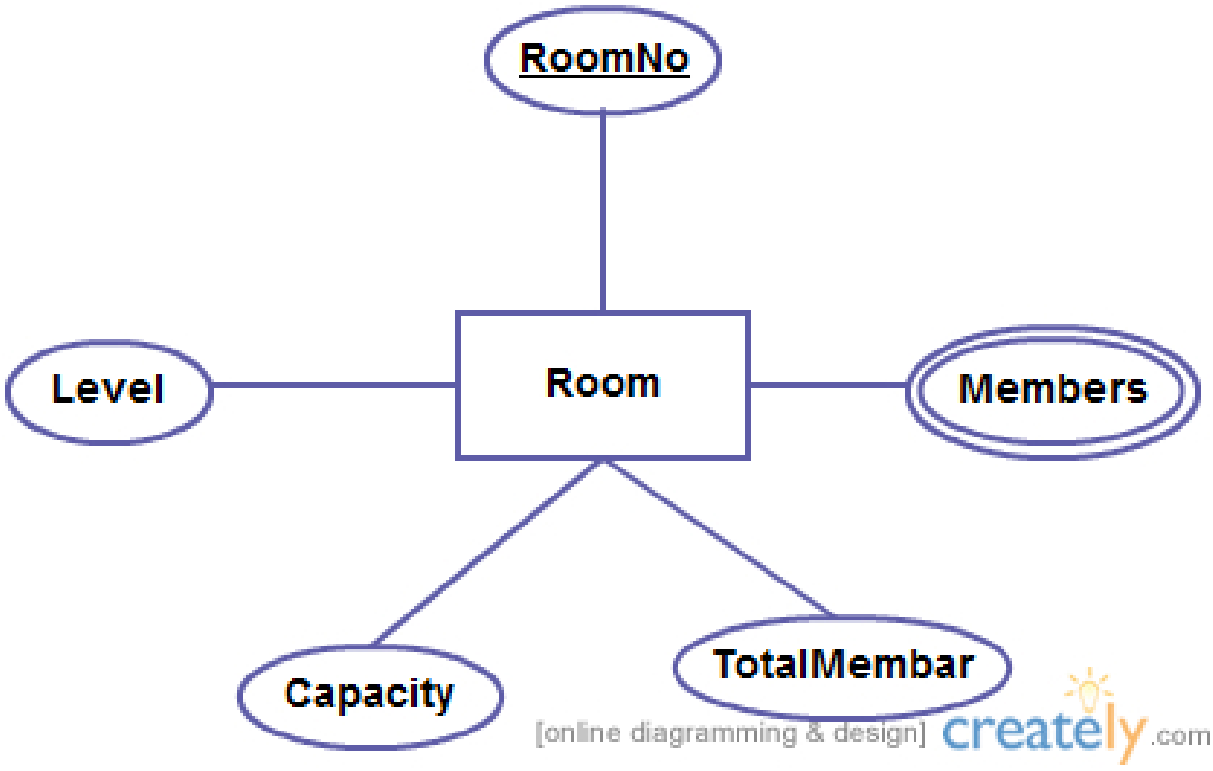

Figure : Entity Relationship Diagram

Here is the Entity Room showing the attributes. The primary key is the RoomNo. It also shows the capacity that how many students a room can hold. Also the total member of the room and the level number of the room.

Procedure (1)

Procedure Name	getResidentStudentList
Time of Execution	Provost has to submit monthly a resident list to the administration office
Input	Nil
Output	List of names of resident students and the date of their being resident
Apply on	Student

Action	Remarks
Select name and date of being resident from Student	Select the name and resident date from table Student where status = 'Resident'

In this procedure Provost will search the database to find the resident students and then he will submit a resident list to the administration office in a monthly basis. He can also obtain the date when a student joined into his room.

Procedure (2)

Procedure Name	getMealList
Time of Execution	When the provost wants to see if any mess manager is managing meals okay
Input	Date1, Date2
Output	List of name of the meals that are served within Date1 and Date2
Apply on	Meal

Action	Remarks
Select meal from Meal where date of meal >= Date1 and =< Date2	Select all the food meals from the table Meal and showed in a ascending order by date

A Mess Manager has got a lot of duty but managing meal is the most important thing for him/her. So the hall provost can keep an eye on the mess manager. He can see what type of meal he is providing to the student from the system within a time duration cause system will keep record of the meal menu which has been updated in the system by the student.

Procedure (3)

Procedure Name	getRoomMaintenanceComplain
Time of Execution	Any time a student posts a complain that is under Room Maintenance it is immediately acknowledged
Input	Nil
Output	List of name room and the problem description
Apply on	Complain

Action	Remarks
Select complain from Complain where type = 'Room Maintenance'	This type of problems are served immediately by the staff members

This procedure diagram is about room maintenance. There is a virtual complain box for each student which he/she will be able to see after logging into account. In there the student will be able to post what kind of problem he/she is facing in the room. If the fan or light needs to be repaired or replaced or any other problems will be acknowledged immediately. Then the provost will send the staff members to solve all the problems related to the room. After taking actions complain will be deleted from the system.

Procedure (4)

Procedure Name	getExpiredNotice
Time of Execution	Any time a student wants to see those notices that are old and expired
Input	Date1, Date2
Output	List of notice along with their publishing date
Apply on	Notice

Action	Remarks
Select notice from Notice where publishing date is >= Date1 and =< Date2	Select all notice from Notice within Date1 and Date2 and showed in a ascending order by date

Our system's job is to delete notice which has been posted a long time ago. But still a student can get those expired notice by simply logging into their account and just putting a date limit in the notice option. The main thing is that system will stop publishing all those notices which are too old but it will be kept into the systems memory and students will be able to fetch that notices from the systems memory.

Procedure (5)

Procedure Name	getRecentResidentList
Time of Execution	When a provost wants to see the resident students that joined recently
Input	Date1
Output	List of recently resident students that joined after Date1
Apply on	Notice

Action	Remarks
Select name and student_ID from Student where date of being resident >= Date1	Show the name and student_ID from Student in an ascending order of date

This is a procedure where provost will be able to see all the students joined recently. When a student will join into his desired hall and desired room. His information will be updated in the system such as joining date, student id, room number etc. So provost can search the database of the students joined recently. Firstly he needs to insert a date into the system then the system will show all the students list joined from that date.

VIEW 1

Election view:

Here the students can see the selected candidate list for voting in the mess manager election.

VIEW 2

Notice view :

Here students can see the published notice by the hall staff members.

VIEW 3

Room View :

Here the list of the rooms which are empty is supplied to the hall supervisor. After having the supplied list he can assign the eligible student an empty room.

ROLE 1

Create role provost_role;

Grant all privileges on

Application
to provost_role

ROLE 2

Create role supervisor_role;

Grant all privileges on
empty room
to supervisor_role;

Trigger 1

Trigger Name: defaultMealSelection

Time of Execution: After Meal selection timeLimit ends and isMealSelected sets to NO.

Actions:

Commands: SelectDefault

Attributes: MealID

Tables: Meal

Remarks: To prevent issues when no meal is selected by a student.

Trigger 2

Trigger Name: startVoteCount

Time of Execution: After isVotingFinished sets to YES

Actions:

Commands: Count

Attributes: studentID

Tables: Mess Manager

Remarks: To process result count when voting is done.

Trigger 3

Trigger Name: deleteExpiredNotice

Time of Execution: After noticeStatus sets to EXPIRED

Actions:

Commands: delete

Attributes: NoticeID

Tables: Notice

Remarks: helps maintenance easy and up to date with most important notices.

Trigger 4

Trigger Name: makeStudentResident

Time of Execution: After isApproved sets to YES

Actions:

Commands: makeResident

Attributes: StudentID

Tables: Student

Remarks: Autoupdate resident status after being approved by provost.

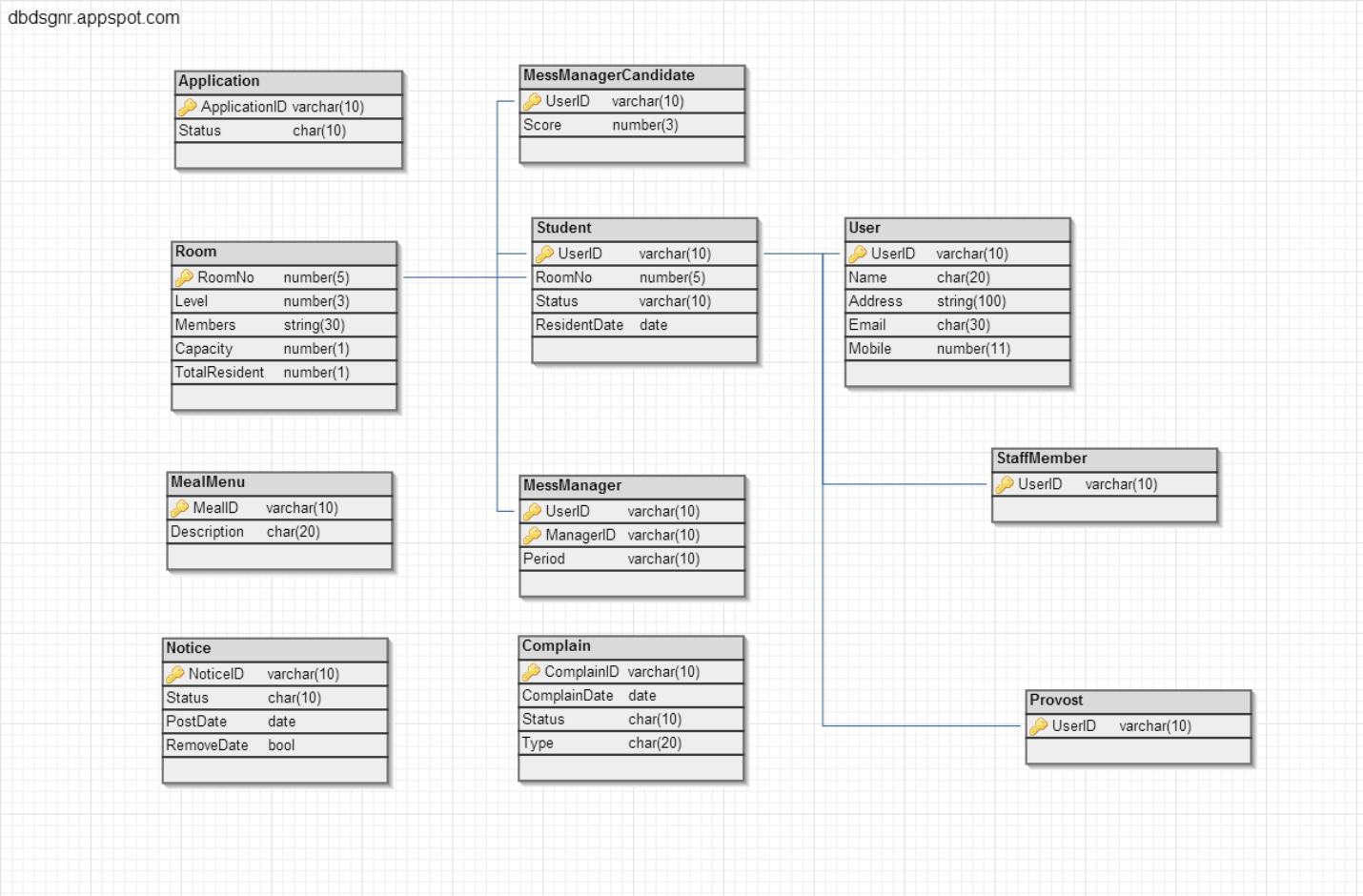

Figure : Database Schema for Hall Management System