DBMS - Mini Project

HOSTEL MANAGEMENT SYSTEM

Submitted By:

MEDA VARUN KUMAR

PES1UG20CS250

SECTION E

Short Description and Scope of the Project

Hostel Management System is a web application which aims at computerization of current procedure of allocating hostel rooms. Currently the process involves students filling up the forms and submitting them in respective hostel offices which involve a lot of paperwork, hence less efficient.

Scope of the Project

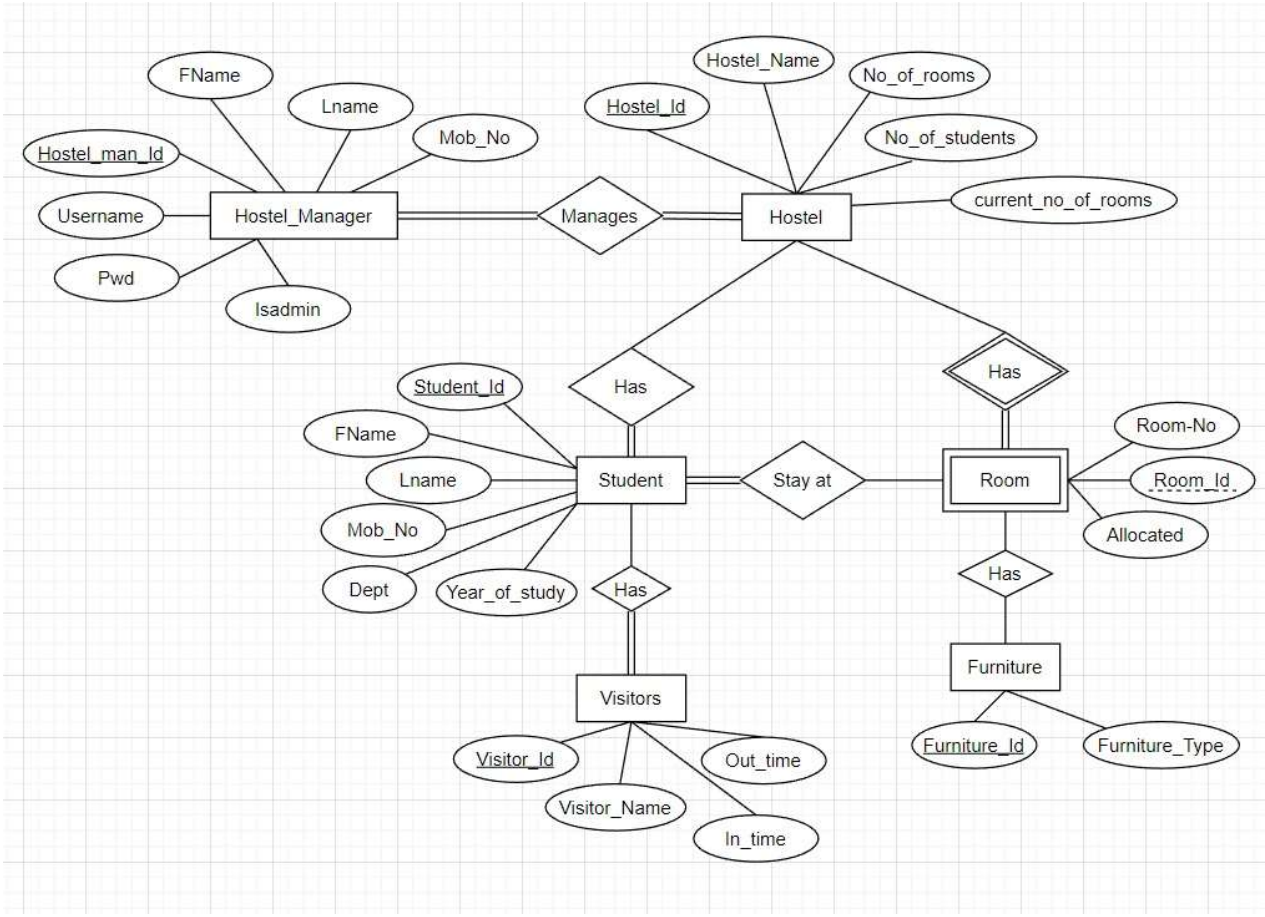
● Hostel Management System is designed for Hostel (like schools, Universities)

●There will be predefined criteria for the Reservation to the hostels.

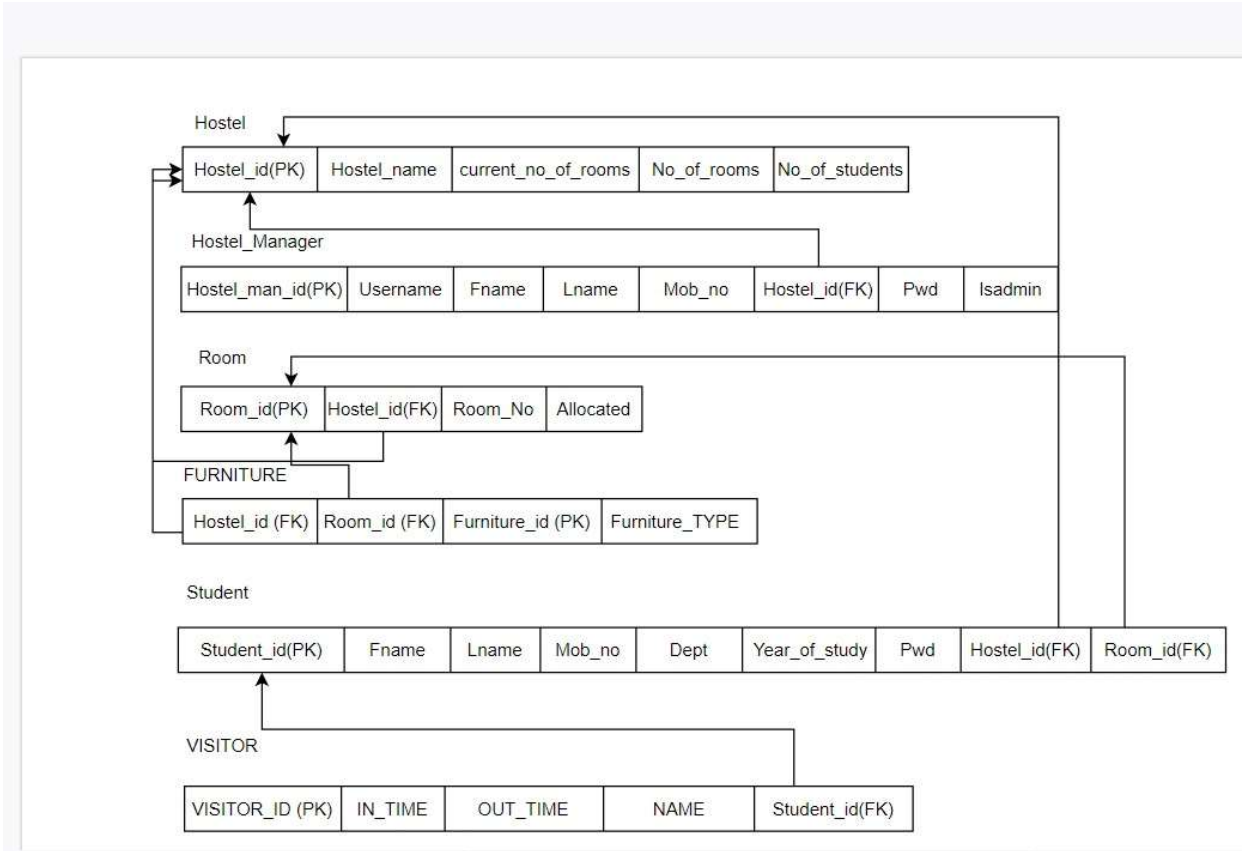
● Manager checks the attested application forms of the students obtained from the internet and verify it with the student database.

● If the students are found eligible then they are allotted to the hostel Room.

ER Diagram


FName
Lname
Mob_No
Hostel_man_Id
Username
Hostel_Manager
Pwd
Isadmin
Manages
Hostel_Name
No_of_rooms
Hostel_Id
No_of_students
Hostel
current_no_of_rooms
Has
Student_Id
FName
Lname
Mob_No
Dept
Year_of_study
Student
Has
Visitors
Visitor_Id
Visitor_Name
In_time
Out_time
Stay at
Has
Room
Room-No
Room_Id
Allocated
Has
Furniture
Furniture_Id
Furniture_Type

Relational Schema



DDL statements - Building the database

```
DROP TABLE IF EXISTS `Hostel`;
CREATE TABLE `Hostel` (
  `Hostel_id` int(10) NOT NULL AUTO_INCREMENT,
  `Hostel_name` varchar(255) NOT NULL,
  `current_no_of_rooms` varchar(255) DEFAULT NULL,
  `No_of_rooms` int DEFAULT NULL,
  `No_of_students` int DEFAULT NULL,
  PRIMARY KEY (`Hostel_id`)
) ENGINE=InnoDB AUTO_INCREMENT=7 DEFAULT CHARSET=latin1;


--
-- Table structure for table `Hostel_Manager`
--

DROP TABLE IF EXISTS `Hostel_Manager`;

CREATE TABLE `Hostel_Manager` (
  `Hostel_man_id` int(10) NOT NULL AUTO_INCREMENT,
  `Username` varchar(255) NOT NULL,
  `Fname` varchar(255) NOT NULL,
  `Lname` varchar(255) NOT NULL,
  `Mob_no` varchar(255) NOT NULL,
  `Hostel_id` int(10) NOT NULL,
  `Pwd` LONGTEXT NOT NULL,
  `Isadmin` tinyint(1) DEFAULT '0',
  PRIMARY KEY (`Hostel_man_id`),
  UNIQUE (`Username`),
  KEY `Hostel_id` (`Hostel_id`),
  FOREIGN KEY (`Hostel_id`) REFERENCES `Hostel` (`Hostel_id`)
);
-- Table structure for table `Room`
DROP TABLE IF EXISTS `Room`;

CREATE TABLE `Room` (
  `Room_id` int(10) NOT NULL AUTO_INCREMENT,
  `Hostel_id` int(10) NOT NULL,
  `Room_No` int(10) NOT NULL,
  `Allocated` tinyint(1) DEFAULT '0',
  PRIMARY KEY (`Room_id`),
  KEY `Hostel_id` (`Hostel_id`),
  FOREIGN KEY (`Hostel_id`) REFERENCES `Hostel` (`Hostel_id`)
) ;
-- tABLE STRUCTURE FOR FURNITURE
```

```sql
CREATE TABLE FURNITURE (
    Hostel_id int,
    Room_id int(10) NOT NULL,
    Furniture_id int NOT NULL,
    Furniture_TYPE varchar(30),
    PRIMARY KEY (Furniture_id),
    FOREIGN KEY (Hostel_id) REFERENCES Hostel(Hostel_id),
    FOREIGN KEY (Room_id) REFERENCES Room(Room_id)
);
-- Table structure for table `Student`
DROP TABLE IF EXISTS `Student`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `Student` (
  `Student_id` varchar(255) NOT NULL,
  `Fname` varchar(255) NOT NULL,
  `Lname` varchar(255) NOT NULL,
  `Mob_no` varchar(255) NOT NULL,
  `Dept` varchar(255) NOT NULL,
  `Year_of_study` varchar(255) NOT NULL,
  `Hostel_id` int(10) DEFAULT NULL,
  `Room_id` int(10) DEFAULT NULL,
  PRIMARY KEY (`Student_id`),
  KEY `Hostel_id` (`Hostel_id`),
  KEY `Room_id` (`Room_id`),
  FOREIGN KEY (`Hostel_id`) REFERENCES `Hostel` (`Hostel_id`)
on  delete  cascade on  update cascade,
  FOREIGN KEY (`Room_id`) REFERENCES `Room` (`Room_id`) on  delete  cascade on
update  cascade
) ;

-- tABLE STRUCTURE FOR VISITOR TABLE

CREATE TABLE VISITOR(
    VISITOR_ID INT NOT NULL,
    IN_TIME DATETIME,
    OUT_TIME DATETIME,
    NAME VARCHAR(20),
    Student_id VARCHAR(20),
    PRIMARY KEY  (VISITOR_ID),
    FOREIGN KEY (Student_id) REFERENCES STUDENT(Student_id)
on  delete  cascade on  update cascade
);
```

Populating the Database

```
INSERT INTO `Hostel` VALUES (1,'MM BLOCK',NULL,50,NULL),
(2,'IH BLOCK',NULL,52,NULL),
(3,'IT BLOCK',NULL,45,NULL),
(4,'MESS BLOCK',NULL,40,NULL),
(5,'NEW BLOCK',NULL,25,NULL),
(6,'NBX',NULL,20,NULL);

INSERT INTO Hostel_Manager VALUES
(100,'M_REDDY','MAHESH','REDDY','9871234560',1,'helloworld',DEFAULT),
(101,'B_REDDY','BABU','REDDY','9813234560',2,'helloworld1',DEFAULT),
(102,'K_REDDY','KIRAN','REDDY','9845634560',3,'helloworld2',DEFAULT),
(103,'R_REDDY','RAKESH','REDDY','9789234560',4,'helloworld3',DEFAULT),
(104,'K_KIRAN','KIRAN','KUMAR','9852634560',5,'helloworld4',DEFAULT),
(105,'S_GUPTHA','SAFAL','GUPTHA','9963234560',5,'helloworld5',DEFAULT);

INSERT INTO Room VALUES (5600,1,01,DEFAULT),
(5645,2,01,1),
(5663,3,03,DEFAULT),
(5644,4,02,1),
(5633,5,03,DEFAULT),
(5689,6,02,1);

INSERT INTO FURNITURE VALUES (1,5600,10001,'AC'),
(2,5645,10002,'AC WITH BED'),
(3,5663,10003,'NON AC'),
(4,5644,10004,'NON AC WITH BED'),
(5,5633,10005,'ATTACHED BT'),
(6,5689,10006,'AC WITH ATTACHED BT');

INSERT INTO STUDENT VALUES(250,'SURYA','TEJA','9873214560','CSE',2024,1,5600),
(251,'VARUN','KUMAR','7337265192','CSE',2024,2,5645),
(252,'AJAY','DEVAGAN','6937265192','ECE',2024,3,5663),
(253,'PRADEEP','KUMAR','9637265192','CSE',2024,4,5645),
(254,'NILESH','KUMAR','6397265192','ECE',2024,5,5633),
(255,'SUNDEEP','A','9667265192','CSE',2024,6,5689);

INSERT INTO VISITOR VALUES(661,'2022-06-18 10:34:09','2022-06-18
11:34:09','VIJAY KUMAR',251);
```

Join Queries

```
-- list all the hostel manager details with the number of students in that particular block they are administrating for
```

```
SELECT *
FROM Hostel
LEFT JOIN Hostel_Manager
ON Hostel.Hostel_id = Hostel_Manager.Hostel_id;
```

Hostel_id	Hostel_name	current_no_of_rooms	No_of_rooms	No_of_students	Hostel_man_id	Username	Fname	Lname	Mob_no	Hostel_id	Pwd
1	MM BLOCK	20	50	100	100	M_REDDY	MAHESH	REDDY	9871234560	1	helloworld
2	IH BLOCK	30	52	104	101	B_REDDY	BABU	REDDY	9813234560	2	helloworld1
3	IT BLOCK	44	45	90	102	K_REDDY	KIRAN	REDDY	9845634560	3	helloworld2
4	MESS BLOCK	15	40	80	103	R_REDDY	RAKESH	REDDY	9789234560	4	helloworld3
5	NEW BLOCK	25	25	75	104	K_KIRAN	KIRAN	KUMAR	9852634560	5	helloworld4
6	NBX	20	20	60	105	S_GUPTHA	SAFAL	GUPTHA	9963234560	6	helloworld5

```
List all the hostel blocks with hostel managers
```

```
SELECT *
FROM Hostel
RIGHT JOIN Hostel_Manager
ON Hostel.Hostel_id = Hostel_Manager.Hostel_id;
```

Hostel_id	Hostel_name	current_no_of_rooms	No_of_rooms	No_of_students	Hostel_man_id	Username	Fname	Lname	Mob_no	Hostel_id	Pwd
1	MM BLOCK	20	50	100	100	M_REDDY	MAHESH	REDDY	9871234560	1	helloworld
2	IH BLOCK	30	52	104	101	B_REDDY	BABU	REDDY	9813234560	2	helloworld1
3	IT BLOCK	44	45	90	102	K_REDDY	KIRAN	REDDY	9845634560	3	helloworld2
4	MESS BLOCK	15	40	80	103	R_REDDY	RAKESH	REDDY	9789234560	4	helloworld3
5	NEW BLOCK	25	25	75	104	K_KIRAN	KIRAN	KUMAR	9852634560	5	helloworld4
6	NBX	20	20	60	105	S_GUPTHA	SAFAL	GUPTHA	9963234560	6	helloworld5

```
lists details of the visitor and the student the visitor came to meet
```

```
SELECT *
FROM STUDENT
INNER JOIN VISITOR
ON STUDENT.Student_id = VISITOR.Student_id;
```

Student_id	Fname	Lname	Mob_no	Dept	Year_of_study	Hostel_id	Room_id	VISITOR_ID	IN_TIME	OUT_TIME	NAME	Student_id
201	VARUN	KUMAR	7337265192	CSE	2024	2	5645	661	2022-06-18 10:34:09	2022-06-18 11:34:09	VIJAY KUMAR	201

```sql
-- furniture present in different blocks
SELECT furniture_type,hostel.Hostel_id AS Hostel_id FROM
Hostel INNER JOIN furniture
ON Hostel.Hostel_id = furniture.Hostel_id;
```

Showing rows 0 - 5 (6 total, Query took 0.0056 seconds.)

```sql
SELECT furniture_type,hostel.Hostel_id AS Hostel_id FROM Hostel INNER JOIN furniture ON Hostel.Hostel_id = furniture.Hostel_id;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

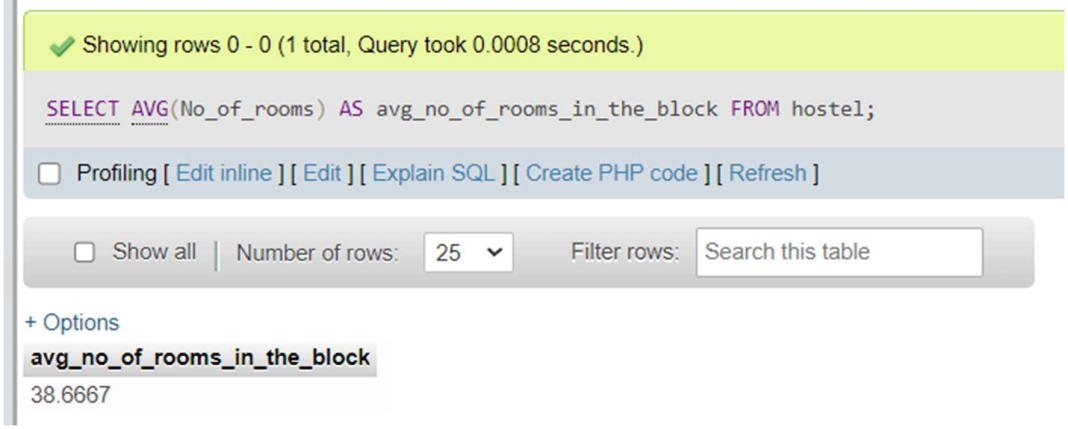
Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

+ Options

furniture_type	Hostel_id
AC	1
AC WITH BED	2
NON AC	3
NON AC WITH BED	4
ATTACHED BT	5
AC WITH ATTACHED BT	6

Aggregate Functions

```
-- average number of rooms in the block
```




```
-- number of visitors on a particular date
select DATE(IN_TIME),count(VISITOR_ID)
FROM VISITOR
GROUP BY date(IN_TIME);
```

DATE(IN_TIME)	count(VISITOR_ID)
2022-06-18	1

```
-- NO OF STUDENTS STAYING IN THE HOSTEL
```

```
SELECT COUNT(Student_id) AS NUMBER_OF_STUDENTS
FROM STUDENT;
```

NUMBER_OF_STUDENTS
6

```
-- SUM OF STUDENTS THE HOSTEL CAN ACCOMIDATE
```

```
SELECT SUM(No_of_students)
FROM HOSTEL;
```

SUM(No_of_students)
509

Set Operations

```
--details about mmblock and nbx block
SELECT * from hostel WHERE Hostel_name = 'MM Block'
UNION
SELECT * from hostel WHERE Hostel_name = 'NBX';
```

Hostel_id	Hostel_name	current_no_of_rooms	No_of_rooms	No_of_students
1	MM BLOCK	20	50	100
6	NBX	20	20	60

```
--details of hostel_manager whose lname is reddy except
manager kiran reddy
SELECT * from hostel_manager WHERE Lname = 'REDDY'
EXCEPT
SELECT * FROM hostel_manager WHERE Fname = 'KIRAN'
```

Hostel_man_id	Username	Fname	Lname	Mob_no	Hostel_id	Pwd	Isadmin
100	M_REDDY	MAHESH	REDDY	9871234560	1	helloworld	0
101	B_REDDY	BABU	REDDY	9813234560	2	helloworld1	0
103	R_REDDY	RAKESH	REDDY	9789234560	4	helloworld3	0

```
--fname and lname of hostel managers whose lname is reddy
with hostel_man_id > 100
SELECT Fname,Lname FROM hostel_manager WHERE Lname = 'REDDY'
INTERSECT
SELECT Fname,Lname FROM hostel_manager WHERE Hostel_man_id >
101
```

Fname	Lname
BABU	REDDY
KIRAN	REDDY
RAKESH	REDDY

```sql
--To find rooms that have ac
SELECT Hostel_id,Room_ID,Furniture_id FROM FURNITURE WHERE
Furniture_TYPE = 'AC'
UNION
SELECT Hostel_id,Room_ID,Furniture_id FROM FURNITURE WHERE
Furniture_TYPE = 'AC WITH BED'
UNION
SELECT Hostel_id,Room_ID,Furniture_id FROM FURNITURE WHERE
Furniture_TYPE = 'AC WITH ATTACHED BT'
```

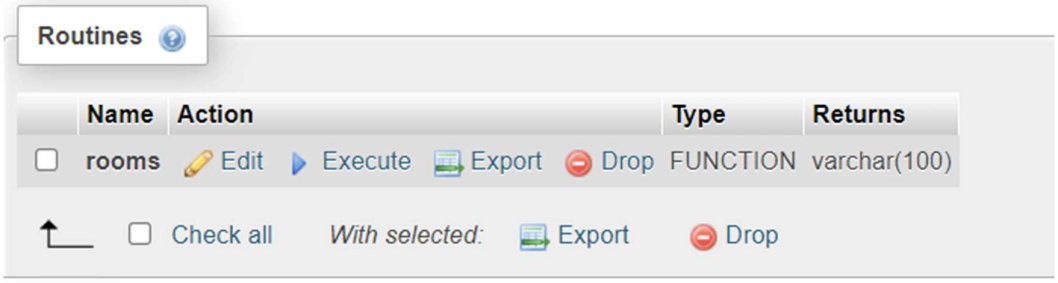
Hostel_id	Room_ID	Furniture_id
1	5600	10001
2	5645	10002
6	5689	10006

Functions and Procedures

```
DELIMITER $$
CREATE FUNCTION rooms(No_of_rooms int)
  RETURNS varchar(100)
  DETERMINISTIC
  BEGIN
    if No_of_rooms>42 THEN
      RETURN ('Rooms are filling fast');
    ELSE
      RETURN ('Rooms are yet to be filled');
    END if;
    END $$
    DELIMITER ;
-- CALLING THE FUNCTION
--SELECT Hostel_name,rooms(No_of_rooms) AS COMMENTS FROM
hostel;


-- PROCEDURE
DELIMITER $$
CREATE PROCEDURE STUDENTS_IN_HOSTEL(IN X INT)
  BEGIN
    SELECT * FROM STUDENT WHERE STUDENT_ID = X;
    END $$
DELIMITER ;
CALL STUDENTS_IN_HOSTEL(250);
```

When the hostel managers inputs the number of rooms booked this function helps him by showing that if more than 42 rooms are filled then the message displayed is “Rooms are filling fast” or if number of rooms filled is less than 42 then we display the message “Rooms are yet to be filled”.

We have a procedure named STUDENTS_IN_HOSTEL which gives the details of the student present in the hostel based on his student_id




```
SET @p0='20'; SELECT `rooms`(@p0) AS `rooms`;
```

Execution results of routine `rooms`

rooms
Rooms are yet to be filled

```
SET @p0='45'; SELECT `rooms`(@p0) AS `rooms`;
```

Execution results of routine `rooms`

rooms
Rooms are filling fast

Student_id	Fname	Lname	Mob_no	Dept	Year_of_study	Hostel_id	Room_id
206	SIDDHARTH	M	9591677265	CSE	2024	8	5695

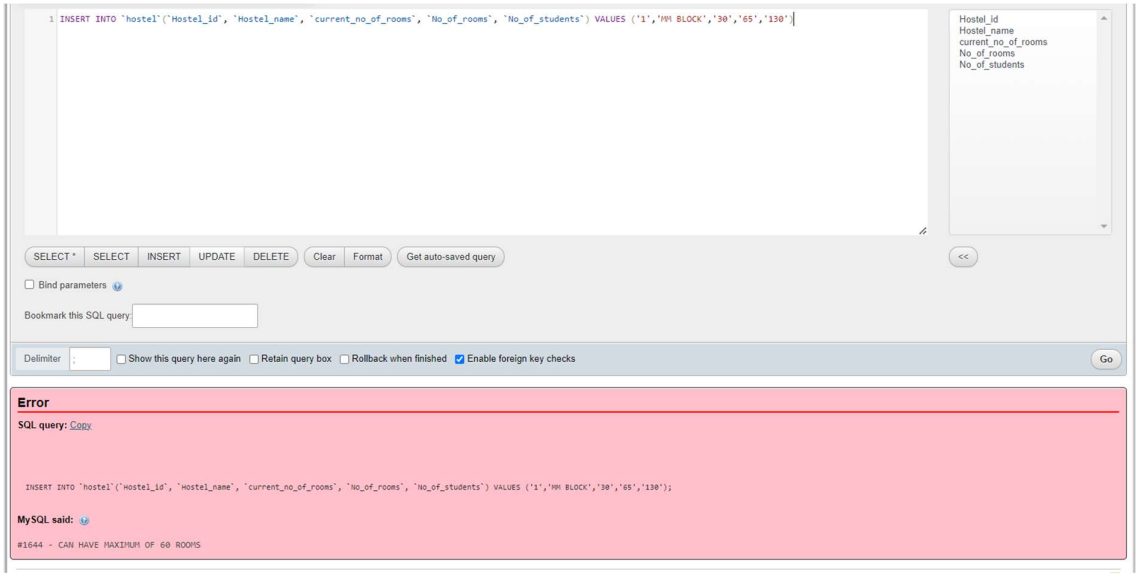
Triggers and Cursors

```
-- TRIGGER
DELIMITER //
CREATE TRIGGER hostel_insertion
BEFORE INSERT
ON HOSTEL FOR EACH ROW
BEGIN
DECLARE error_msg VARCHAR(255);
SET error_msg = ('CAN HAVE MAXIMUM OF 60 ROOMS');

IF (New.No_of_rooms ) > 60
THEN
SIGNAL SQLSTATE '45000'
SET MESSAGE_TEXT = error_msg;
END IF;
END //
DELIMITER ;

-- CURSOR
create table student_backup ( STUD_ID varchar(6),
f_name varchar(30),
L_NAME varchar(16),
MOBILE_NUMBER Varchar(10) ,
DEPARTMENT varchar(6),
GRADUATION_YEAR varchar(6),
H_ID varchar(6),
ROOM_ID varchar(6));


DELIMITER //
CREATE PROCEDURE cursor_1()
   BEGIN
      DECLARE done INT DEFAULT 0;
      DECLARE A,B,C,D,E,F,G,H VARCHAR(30);
      DECLARE cur CURSOR FOR SELECT * FROM STUDENT;
      DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;
      OPEN cur;
      label: LOOP
```

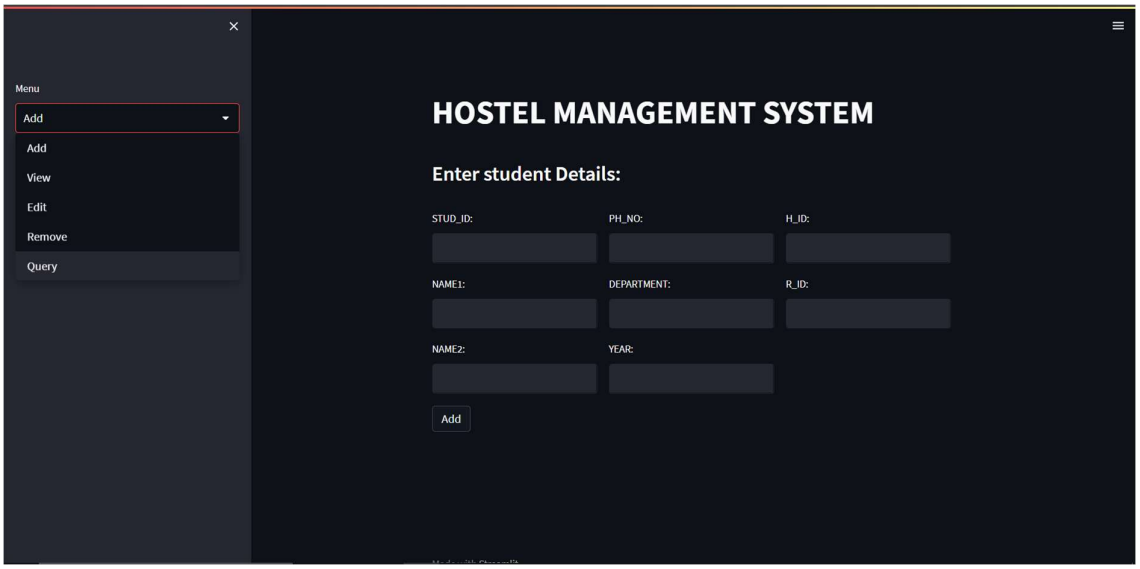
```
      FETCH cur INTO A,B,C,D,E,F,G,H;
      INSERT INTO student_backup VALUES(A,B,C,D,E,F,G,H);
      IF done = 1 THEN LEAVE label;
      END IF;
      END LOOP;
      CLOSE cur;
   END//
DELIMITER ;

call cursor_1();
```

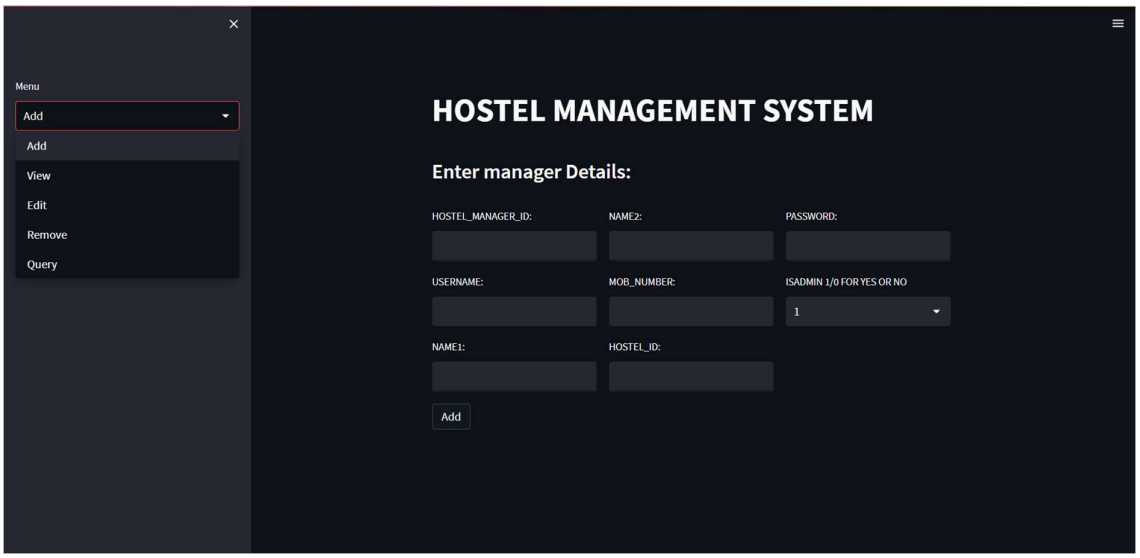
Cursor is a temporary work station. It is allocated by Database Server at the time of performing DML operations on Table by User. Cursors are used to store Database Tables. In our project we are using cursor to take the backup of student table as it's very important to save the data of the students. Having a backup will always be helpful in case the first table gets corrupted

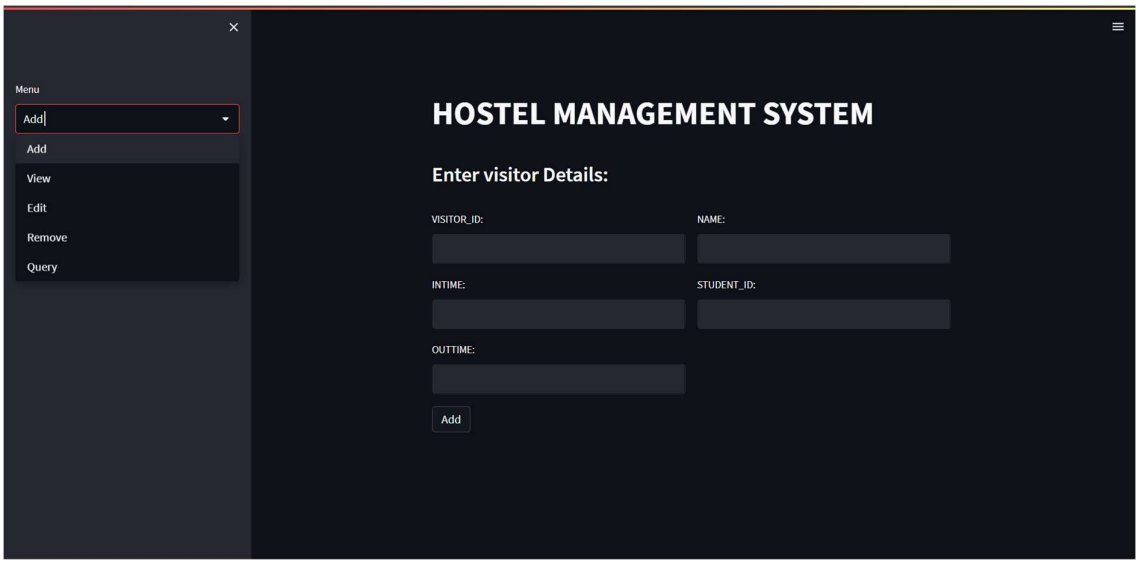
A trigger is a stored procedure in database which automatically invokes whenever a special event in the database occurs. In our project an error message is triggered if it has reached the limit, we can have maximum 60 rooms. The message is displayed if the number of rooms exceeds 60.



Developing a Frontend






Menu
Add
Add
View
Edit
Remove
Query
HOSTEL MANAGEMENT SYSTEM
Enter visitor Details:
VISITOR_ID:
NAME:
INTIME:
STUDENT_ID:
OUTTIME:
Add

THE - END