Student Housing and Maintenance Management System

1.Description:

This database aims to manage student dormitories, room assignments, housing contracts, and maintenance requests in a centralized system.
Universities often keep this information in separate spreadsheets or emails, which causes data inconsistency, double booking, and lost maintenance requests.
The system integrates all housing and maintenance operations, enforcing clear key and participation constraints to ensure data accuracy.

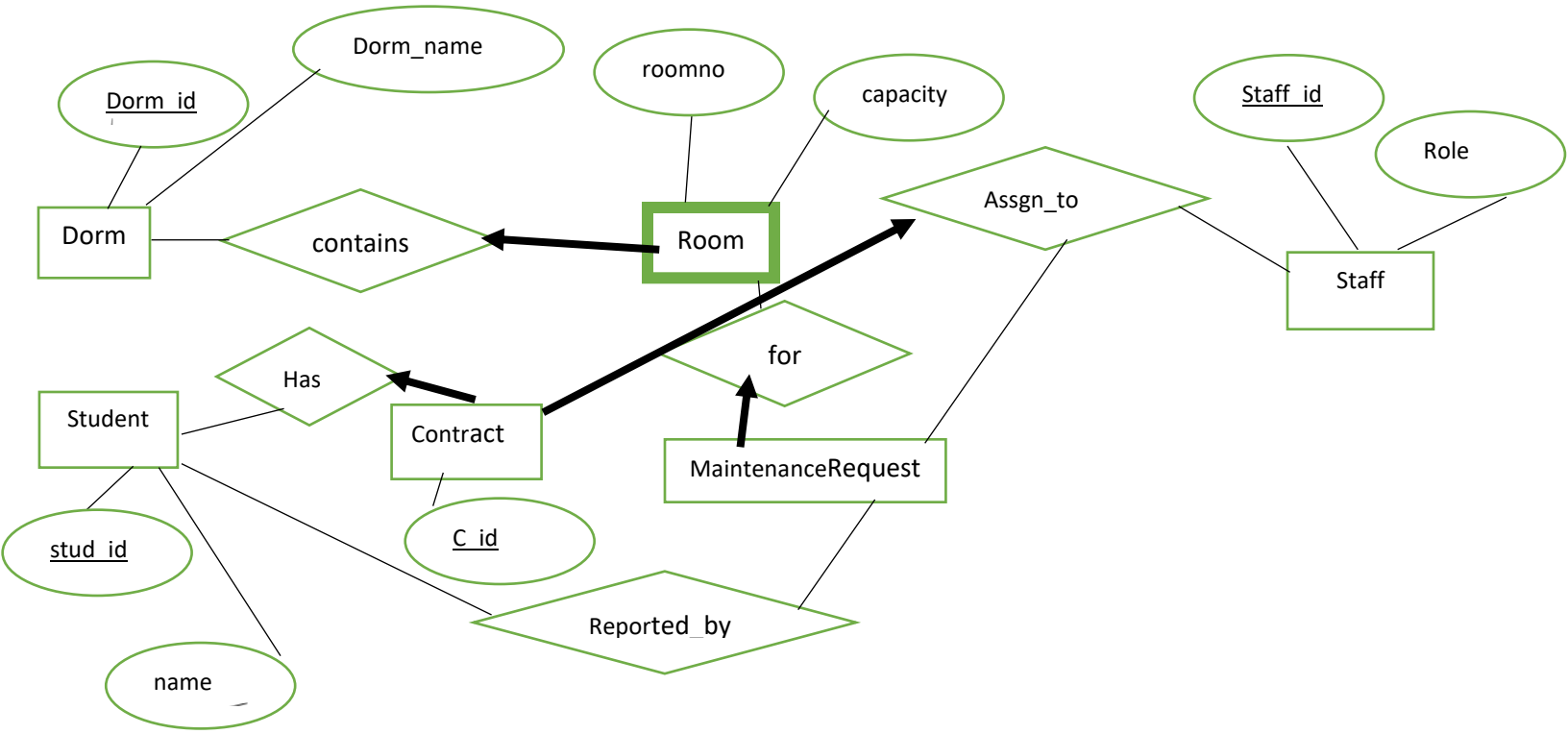
Entities

- **Dorm**: Represents each residence hall.
  Attributes: **Dorm_id (PK)**, Dorm_name, Address, Gender_policy.
  Each dorm may contain many rooms, but can also be temporarily empty.
- **Room (weak entity)**: Represents a room located within a dorm.
  Attributes: **Roomno (partial key)**, Capacity, Floor, Type.
  Identified by Dorm_id and Roomno through the **Contains** relationship (total participation on Room).
- **Student**: Represents a university student.
  Attributes: **Stud_id (PK)**, Name, Email, Gender.
  Students may or may not currently have a housing contract or maintenance request.
- **Staff**: Represents maintenance employees or supervisors.
  Attributes: **Staff_id (PK)**, Role, Name.
  Staff members may be assigned to maintenance requests.
- **Contract**: Represents a student's housing agreement.
  Attributes: **C_id (PK)**, Start_date, End_date, Rent, Status, (FKs: Stud_id, Dorm_id, Roomno).
  Each contract must be linked to exactly one student and one room.
- **MaintenanceRequest**: Represents a reported problem in a room.
  Attributes: **Request_id (PK)**, Category, Priority, Status, Date_reported, (optional FKs: Stud_id, Staff_id).
  Each request must refer to a room but may or may not have a reporter or assignee.

Key Relationships & Constraints

- **Contains (Dorm–Room):** one Dorm → many Rooms; total participation on Room (Room is a weak entity).
- **Has (Student–Contract):** one Student → many Contracts; total on Contract.
- **Assign_to (Room–Contract):** one Room → many Contracts; total on Contract.
- **For (Room–MaintenanceRequest):** one Room → many Requests; total on Request.
- **Reported_by (Student–MaintenanceRequest):** optional on both sides.
- **Assign_to (Staff–MaintenanceRequest):** optional on both sid
-

2.ER Model:


Dorm_name
roomno
capacity
Staff_id
Role
Dorm_id
Assgn_to
Dorm
contains
Room
Staff
for
Has
Student
Contract
MaintenanceRequest
stud_id
C_id
Reported_by
name

3.Relational Model:

1) DORM

```
CREATE TABLE Dorm (
  dorm_id       INT PRIMARY KEY,
  dorm_name     VARCHAR(80) NOT NULL
  )
INSERT INTO Dorm (dorm_id, dorm_name) VALUES
(1, 'Ada Dorm'),
(2, 'Babbage Dorm'),
(3, 'Curie Dorm'),
(4, 'Dijkstra Dorm'),
(5, 'Edsger Dorm'),
(6, 'Feynman Dorm'),
(7, 'Grace Dorm'),
(8, 'Hopper Dorm'),
(9, 'Iverson Dorm'),
(10, 'Joule Dorm');
```

2) ROOM

```
CREATE TABLE Room (
  dorm_id    INT NOT NULL,
  roomno     INT NOT NULL,
  capacity   INT NOT NULL,
  PRIMARY KEY (dorm_id, roomno),
      FOREIGN KEY (dorm_id) REFERENCES Dorm(dorm_id)
)
```

INSERT INTO Room (dorm_id, roomno, capacity) VALUES

(1, 101, 2),

(1, 102, 1),

(2, 201, 3),

(2, 202, 2),

(3, 305, 1),

(4, 410, 2),

(5, 115, 3),

(8, 220, 2),

(9, 130, 1),

(10, 510, 2);

3) STUDENT

CREATE TABLE Student (

  stud_id INT PRIMARY KEY,

  name   VARCHAR(80) NOT NULL,

)

INSERT INTO Student (stud_id, name) VALUES

(1001, 'Ayşe Yılmaz'),

(1002, 'Mehmet Demir'),

(1003, 'Elif Kaya'),

(1004, 'Can Arslan'),

(1005, 'Zeynep Çelik'),

(1006, 'Efe Kaan'),

(1007, 'Deniz Acar'),

(1008, 'Selin Koç'),

(1009, 'Burak Er'),

(1010, 'İdil Sunar');

4) STAFF

```
CREATE TABLE Staff (
  staff_id INT PRIMARY KEY,
  name     VARCHAR(80) NOT NULL,
  role     VARCHAR(40) NOT NULL
)
INSERT INTO Staff (staff_id, name, role) VALUES
(501, 'Ahmet Usta', 'Electrician'),
(502, 'Fatma Kılıç', 'Plumber'),
(503, 'Oguzhan Tek', 'HVAC Technician'),
(504, 'Seda Bilgin', 'Supervisor'),
(505, 'Yunus Kaya', 'Carpenter'),
(506, 'Gizem Top', 'Cleaner'),
(507, 'Serkan Öz', 'IT Support'),
(508, 'Gül Şen', 'Painter'),
(509, 'Bora Soy', 'Technician'),
(510, 'Merve Tunç', 'Supervisor');
```

5) CONTRACT

```
CREATE TABLE Contract (
  c_id       INT PRIMARY KEY,
  stud_id    INT NOT NULL,
  dorm_id    INT NOT NULL,
```

```
  roomno      INT NOT NULL,
    FOREIGN KEY (stud_id) REFERENCES Student(stud_id)
    FOREIGN KEY (dorm_id, roomno) REFERENCES Room(dorm_id, roomno)
)

INSERT INTO Contract (c_id, stud_id, dorm_id, roomno) VALUES
(9001, 1001, 1, 101),
(9002, 1002, 1, 102),
(9003, 1003, 2, 201),
(9004, 1004, 2, 202),
(9005, 1005, 3, 305),
(9006, 1006, 4, 410),
(9007, 1007, 5, 115),
(9008, 1008, 8, 220),
(9009, 1009, 9, 130),
(9010, 1010, 10, 510);
```

6) MAINTENANCEREQUEST

```
CREATE TABLE MaintenanceRequest (
  request_id        INT PRIMARY KEY,
  dorm_id           INT NOT NULL,
  roomno            INT NOT NULL,
 FOREIGN KEY (dorm_id, roomno) REFERENCES Room(dorm_id, roomno)
  FOREIGN KEY (reporter_stud_id) REFERENCES Student(stud_id)
  FOREIGN KEY (assignee_staff_id) REFERENCES Staff(staff_id)
   )
```

```
INSERT INTO MaintenanceRequest
(request_id, dorm_id, roomno, reporter_stud_id, assignee_staff_id) VALUES
(3001, 1, 101, 1001, 502),
(3002, 1, 102, 1002, 501),
(3003, 2, 201, 1003, 503),
(3004, 2, 202, 1004, 505),
(3005, 3, 305, 1005, 507),
(3006, 4, 410, 1006, 506),
(3007, 5, 115, 1007, 508),
(3008, 8, 220, 1008, 501),
(3009, 9, 130, 1009, 509),
(3010, 10, 510, 1010, 503);
```