

**ATILIM UNIVERSITY**

**2024-2025 FALL SEMESTER**

**CMPE341-Database Design and Management**

**HOSPITAL MANAGEMENT**

**SYSTEM**

**Group Members:**

Oğuzhan Kaya – 21243510059

Kadir Özbek – 21243510098

Atakan Bektaş – 21243510150

Yusuf Emre Konuk – 21243610047

**Supervisor: Mahmut Furkan Bakal**

**TABLE OF CONTENTS**

1. Introduction………………………………………………………………………1
2. Requirements……………………………………………………………………..1
3. ER/ERR Diagram of the System…………………………………………………5
4. Logical Design……………………………………………………………………7
5. Physical Design…………………………………………………………………...9
6. Interface………………………………………………………………………….14

**1. Introduction**

* **Form Groups**

We did this project with a group of 4 people. We did not make a special distribution of work among group members. Everyone worked on every subject and we tried to put the best of our work into the project.

Our group members are as follows:

1. Oğuzhan Kaya 21243510059
2. Kadir Özbek 21243510098
3. Atakan Bektaş 21243510150
4. Yusuf Emre Konuk 21243610047

* **Brief Explanation of the Project Topic**

Our project focuses on designing and implementing a Hospital Management System.A system that a secretary in a hospital would use. With this database and project, a secretary will be able to record the patients in the hospital. Also, if there is a new doctor, secreter will be able to register this doctor in the system and enter the doctor's room and the doctor's branch. After doing all these, you will be able to arrange an appointment for a registered patient with a doctor in the branch of your choice.

**2. Requirements**

* **Requirement Collection & Analysis Phase**

**Identifying User Expectations**

To develop a functional and efficient Hospital Management System Database, we communicated with potential users, including doctors and patients. Through questionnaires, and interviews, we identified the following user expectations:

- A user-friendly interface for patients to book, reschedule, and cancel appointments.

- Tracking of doctors and doctors' room assignments.

- Automatic conflict detection for overlapping appointments or unavailable doctors/rooms.

* **Distinguishing Data and Operations**

**Data to be Stored:**

**Patients Record:** Patient Id, First Name, Last Name

**Doctors Record:** Doctor Id, First Name ,Last Name , Branch, Room

**Room Details:** Room Id and Room Number

**Branch Details:** Branch Id ,Branch Name

**Appointments Record:** Appointment ID, Price, Appointment Time, Doctor ,Branch,

Patient

**Operations Needed on Data:**

**Patients Information**

**Operation:** Creating new Patients

**Description:** This operation enables the create new patients with their Name and Surname.

**Operation:** Deleting Patient

**Description:** This operation allows the patient to be deleted when patient deregisters from the hospital.

**Operation:** Updating Patient information

**Description:** This operation allows the information to be changed if a patient's information has been entered incorrectly or has been changed.

**Room Information**

**Operation:** Creating new Rooms

**Description:** This allows us to open a new room with the number of the room..

**Operation:** Deleting Room

**Description:** This operation allows us to delete a room if it is no longer in use.

**Operation:** Updating Room Information

**Description:** This operation allows us to change the number of a room if we want to change it to another number.

**Branch Information**

**Operation:** Creating new Branch

**Description:** This operation allows us to create a new branch with the name of the branch. If a new branch is wanted to opened in the hospital.

**Operation:** Updating Branch Information

**Description:** This operation allows us to correct the name of branch and save it again. If a branch name is entered incorrectly.

**Operation:** Deleting Branch

**Description:** This operation allows us to delete the branch totally. If it is no longer use in that hospital.

**Doctor Information**

**Operation:** Adding new Doctor

**Description:** If there is a new doctor in the hospital, this operation makes us to add that doctor into our database. With doctor’s Name, Surname, Branch and Room.

**Operation:** Updating Doctor Information

**Description:** This operation allows us to update Doctor’s Name, Surname, Branch or Room. If a Doctor's information has been entered incorrectly or has been changed.

**Operation:** Delete Doctor

**Description:** If there is a doctor who leaves the hospital, we can delete the doctor with this delete operation.

**Appointments Information**

**Operation:** Create a new Appointment

**Description:** We can create new appointment with create operation. With details such as Patient, Doctor, Branch, Appointment Time and Price.

**Operation:** Updating Appointment Information

**Description:** This operation allows us to update Appointment details such as Patient, Doctor, Branch, Appointment Time and Price. If a Appointment’s information has been entered incorrectly or has been changed.

**Operation:** Delete Appointment

**Description:** This operation allows the appointments to be deleted. If a wrong appointment is set into database.

**3. ER Diagram of the System**

* **Conceptual Schema Design Phase:**

**Forming Entities and Their Attributes**

**Doctors Entity:**It has DoctorID, first name, last name, BranchID(FK) and RoomID(FK). DoctorID is the Primary Key of doctor entity.

**Patients Entity:**It has PatientID and first name and last name. PatientID is a Primary Key.

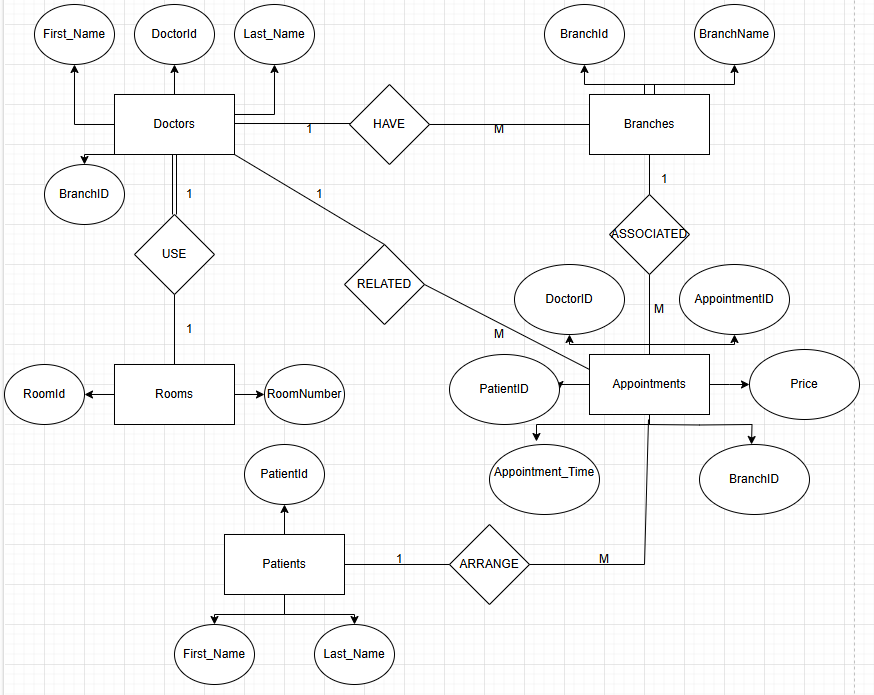
**Branches Entity:**It has BranchID, and name of the branch. BranchID is the Primary Key for Branch entity.

**Rooms Entity:**It has RoomID and number of room. RoomID is the Primary Key for Room Entity.

**Appointments Entity:** It has AppointmentID, DoctorID(FK) ,PatientID(FK), BranchID(FK) Appointment time and Price. AppointmentID is the Primary Key of this entity.

**Relationships between Entities**

* A patient can make many Appointments but one appointment is associated by only one Patient. The relationship 1:N.
* A doctor can have many Appointments but an appointment is related with only one Doctor. The relationship 1:N.
* A Branch can have many doctors within it but a doctor is related with only one branch. The relationship is 1:N.
* A Branch can contain multiple Appointments but an appointment is associated with only one Branch. The relationship is 1:N.
* Each room can used by only one doctor and each doctor have only one room. The relationship is 1:1.
* **ER/EER Data Model is Designed**

****

**4. Logical Design**

**DOCTORS**

**Columns:** DoctorID (Primary Key), First\_name, Last\_name ,BranchID (Foreign Key referencing Branch), RoomID (Foreign Key referencing Room),

**BRANCHES**

**Columns:** BranchID (Primary Key), BranchName

**ROOMS**

**Columns:** RoomID (Primary Key), Room\_Number

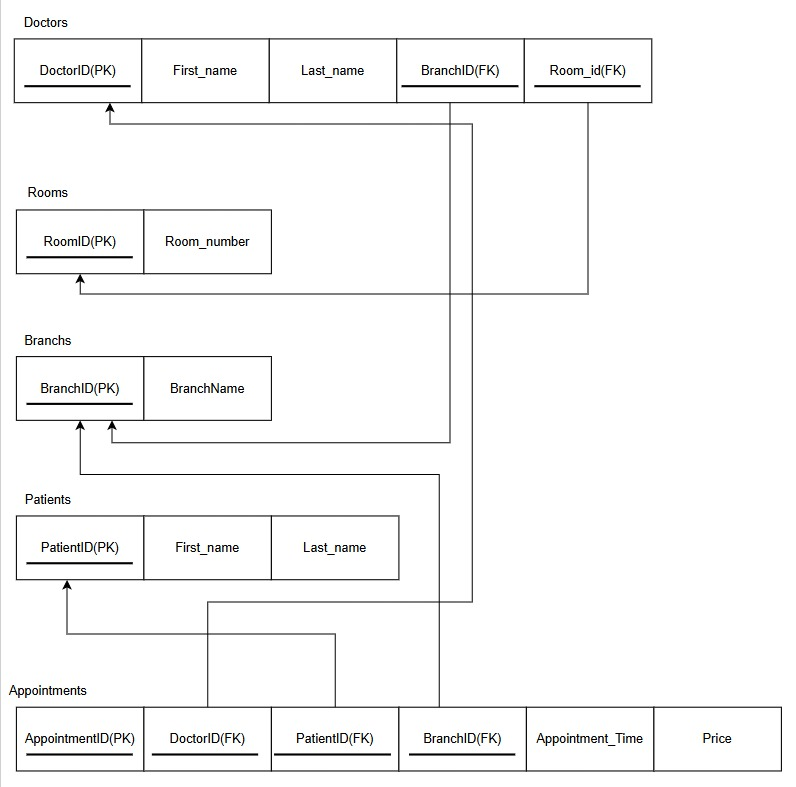
**PATIENTS**

**Columns:** PatientID (Primary Key), First\_name, Last\_name

**APPOINTMENTS**

**Columns:** AppointmentID (Primary Key), DoctorID(Foreign Key referencing Doctor), PatientID(Foreign Key referencing Patient), BranchID(Foreign Key referencing Branch),Appointment\_Time ,Price

* **Logical Desing Model (Mapping)**



**5. Physical Design in Oracle**

* **Database choice and implementation**

**Creating Final Tables:**

**Creating Doctors Table**

CREATE TABLE Doctors (

DOCTOR\_ID number(4),

FIRST\_NAME varchar(20),

LAST\_NAME varchar(20),

BRANCH\_ID number(4) not null,

ROOM\_ID number(3) not null,

primary key(DOCTOR\_ID),

foreign key(BRANCH\_ID) references BRANCH(BRANCH\_ID)

foreign key(ROOM\_ID) references ROOM(ROOM\_ID));

**Creating Patients Table**

CREATE TABLE Patients (

PATIENT\_ID number(4),

FIRST\_NAME varchar(20),

LAST\_NAME varchar(20),

primary key(PATIENT\_ID)

);

**Creating Branches Table**

CREATE TABLE Branches (

BRANCH\_ID number(4),

BRANCH\_NAME varchar(20),

primary key(BRANCH\_ID)

);

**Creating Rooms Table**

CREATE TABLE Rooms (

ROOM\_ID number(3),

ROOM\_NUMBER number(3),

primary key(ROOM\_ID)

);

**Creating Appointments Table**

CREATE TABLE Appointments (

APPOINTMENT\_ID number(3),

DOCTOR\_ID number(4) not null,

PATIENT\_ID number(4) not null,

BRANCH\_ID number(4) not null,

APPOINTMENT\_TIME DATE not null,

PRICE number(4),

Unique(APPOINTMENT\_TIME),

primary key(APPOINTMENT\_ID),

foreign key(DOCTOR\_ID) references DOCTOR(DOCTOR\_ID),

foreign key(BRANCH\_ID) references BRANCH(BRANCH\_ID),

foreign key(PATIENT\_ID) references PATIENT(PATIENT\_ID)

);

**Data Loading:**

**Inserting data into Doctors Table**

INSERT INTO Doctors VALUES (0001,’Mahmut Furkan’, ‘Bakal’,0010,1);

INSERT INTO Doctors VALUES (0002,’Aysu İrem’,’Adem’,0011,2);

INSERT INTO Doctors VALUES (0003,’Güzin’,’Türkmen’,0012,3);

INSERT INTO Doctors VALUES (0004, ‘Umay Hilde’, ’Yayla’,0012,4);

**Inserting data into Patients Table**

INSERT INTO Patients VALUES (1000, ‘Kadir’, ’Ozbek’);

INSERT INTO Patients VALUES (2000, ‘Oguzhan’, ’Kaya’);

INSERT INTO Patients VALUES (3000, ‘Atakan, ’Bektas’);

INSERT INTO Patient VALUES (4000, ‘Yusuf Emre, ’Konuk’);

**Inserting data into Branches Table**

INSERT INTO Branches VALUES (0010,’Cardiology’);

INSERT INTO Branches VALUES (0011,’Psychology’);

INSERT INTO Branches VALUES (0012,’Dermatology’);

**Inserting data into Rooms Table**

INSERT INTO Rooms VALUES (1, 111)

INSERT INTO Rooms VALUES (2, 222)

INSERT INTO Rooms VALUES (3, 333)

INSERT INTO Rooms VALUES (4, 444)

**Inserting data into Appointments Table**

INSERT INTO Appointments VALUES (101,0002,2000,0011,’28-FEB-25’,999);

INSERT INTO Appointments VALUES (202,0003,1000,0012,’02-JAN-25’,500);

INSERT INTO Appointments VALUES (303,0002,3000,0011,’31-MAR-25’,2000);

INSERT INTO Appointments VALUES (404,0001,4000,0010,’01-JAN-25’,3000);

**Physical Design-Queries:**

**A) Display Data From Multiple Tables (JOIN)**

- This query shows us the names of the branches to which our doctors belong. For this purpose, it combines doctor and branch tables.

**Query:**

select b.branch\_name,d.first\_name

from branches b

join doctors d on b.branch\_id= d.branch\_id

**B) Query including group-by statement, order-by statement**

-This query shows how many patients the doctors have in our appointment table. For this purpose, we group the each of doctor’s first name with their total patient number and provide relationships between the tables via foreign keys. We also order the result with first name.

**Query:**

select count(p.patient\_id) as "PATIENT NUMBER", d.first\_name

from appointmentss a, doctor d, patients p

where a.doctor\_id=d.doctor\_id and a.patient\_id=p.patient\_id

group by d.first\_name

order by d.first\_name

**C) SubQuery**

-We showed the branch name and the name of the doctor who stays in room id 4. To do this, we reached the branch name by using a subquery in which we found the doctor whose room id is 4.

select b.branch\_name, d.first\_name , d.last\_name

from branches b , doctors d

where b.branch\_id= d.branch\_id

and d.doctor\_id = (select doctor\_id from doctor where room\_id=4)

**6. Interface**

I developed an MVC project with C# via visual studio for interface operations. Through this website, they can add, delete and update this system in various tables.