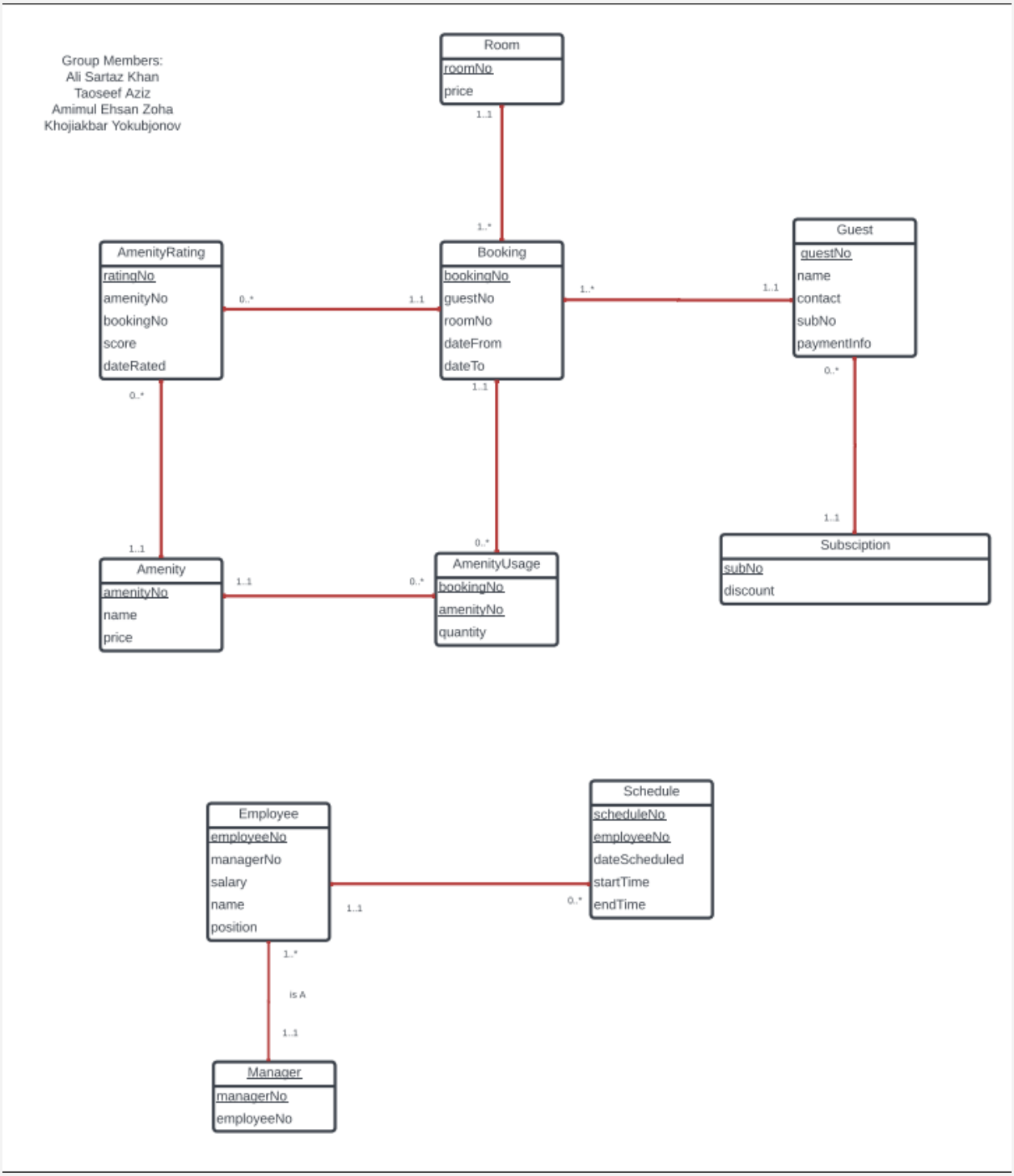
I.Conceptual database design

We have PK-FK relationships in the relational schema to create one to many relationships.



II.Logical database design

AmenityRating

ratingNo	amenityNo	bookingNo	score	date
1	1	1	10	4/6/23
2	2	2	8	4/7/23
3	3	3	5	4/8/23

Amenity

amenityNo	name	price
1	pool	100
2	spa	150

AmenityUsage

bookingNo	amenityNo	quantity
1	2	2
2	3	5

Room

roomNo	Price
1	300
2	350
3	400

Booking

We only allow one room per booking, we do allow multiple guests per booking.

bookingNo	guestNo	roomNo	dateFrom	dateTo
1	1	1	4/20/23	4/22/23
2	2	2	4/21/23	4/23/23

Employee

employeeNo	managerNo	salary	name	position
1	1	50000	Jimmi	cashier
2	2	55000	Tom	cashier

Guest

guestNo	name	contact	subNo	paymentInfo
1	Khojiakbar	5200000000	1	Chase
2	Sartaz	5200000001	2	CreditOne

Subscription

subNo	subName	discount
1	gold	20
2	club460	25

Schedule

scheduleNo	employeeNo	date	startTime	endTime
1	1	4/20/23	0:00:00	2:00:00
2	2	4/20/23	1:00:00	3:00:00

Manager

managerNo	employeeNo	salary	name	position
1	17	220000	Sajeeb	supervisor
2	18	230000	Roy	supervisor

III. Normalization analysis

AmenityRating

{ratingNo} → {amenityNo, bookingNo,score, date}

- 1NF: no set valued attributes.
- 2NF: since non prime attributes (amenityNo, bookingNo,score, date) are F.F.D on the CK (ratingNo).
- 3NF: 1) it is in 2NF.
  2) Each non-primary key attribute (amenityNo, bookingNo,score, date) is FD on the primary key attr. (ratingNo). All the FDs are through the ratingNo.

Amenity

{amenityNo} → {name, price}

-follows 1NF since no attributes are set values.

-follows 2NF because both name and price are fully dependent on the amenityNo

-follows 3NF: 1) it is in 2NF.
2) each non-primary key attribute (name, price) is only FD on the primary key attribute (amenityNo) and not each other.

AmenityUsage
{bookingNo, amenityNo} → {quantity}
-follows 1NF since no attributes are set values.
-follows 2NF because quantity is fully dependent on the primary key (which is a composite key) {amenityNo and bookingNo}
-follows 3NF: 1) it is in 2NF.
2) quantity, the only non-primary key attribute is only FD on the primary key.

Room
{roomNo} → {price}
-follows 1NF since no attributes are set values.
-follows 2NF because price is fully dependent on the primary roomNo.
-follows 3NF: 1) it is in 2NF.
2) price, the only non-primary key attribute is only FD on the primary key.

Booking
{bookingNo} → {guestNo, roomNo, dateFrom, dateTo}
-follows 1NF since no attributes are set values.
-follows 2NF because roomNo, dateFrom, dateTo are fully dependent on the CK {bookingNo}.
-follows 3NF: 1) it is in 2NF.
2) guestNo, roomNo, dateFrom and dateTo are only FD on the primary key (bookingNo) and not on one another.

Employee
{employeeNo} → {managerNo, salary, name, position}
-follows 1NF since no attributes are set values.
-follows 2NF because managerNo, salary, name, position are fully dependent on the CK employeeNo.
-follows 3NF: 1) it is in 2NF.
2) managerNo, salary, name and position are only FD on the primary key (employeeNo) and not on one another. managerNo cannot FD other attributes since it can be null (not all employees have managers in my HMS).

Guest
{guestNo} → { name,contact,subNo, paymentInfo}
-follows 1NF since no attributes are set values.
-follows 2NF because name,contact,subNo, paymentInfo are fully dependent on the CK guestNo.
-follows 3NF: 1) it is in 2NF.
2) all the non-primary key attributes are only FD on the primary key (guestNo) and not on one another.

Subscription
{subNo} → { subName,discount}
{subName} → {subNo,discount}
-follows 1NF since no attributes are set values.
-does not follow 2NF because discount is not fully dependent on both the CKs subNo and subName.
- doesn't follow 3NF because it is not in 2NF.

Schedule
{scheduleNo} → {employeeNo,date,startTime,endTime}
- follows 1NF since no attributes are set values.
-follows 2NF because employeeNo,date,startTime,endTime are fully dependent on the CK scheduleNo.
-follows 3NF: 1) it is in 2NF.
2) No transitive dependencies. All non-prime attributes can only be FD by the primary key (scheduleNo).

Manager
{managerNo} → {employeeNo, salary, name, position}
{employeeNo} → {employeeNo, salary, name, position}
-1NF: no set valued attributes.
-Not 2NF: since non prime attributes (salary, name, position) are not F.F.D on both the CKs managerNo and employeeNo.
-Not in 3NF since not in 2NF. – We need to delete the attr. Salary, name and position since they are redundant.

IV. Query description

For the fifth query, we decided to do the following:

Given the employee name, provide the schedule of the employee's manager.

Essentially, this query allows the employees to see the availability of their managers. Having access to the manager's schedule enables the employees to easily schedule meetings, maintain good communication, and prioritize their tasks.