CPCS241 – Database I – Spring 2022 – Group Project

[Smash Up GYM]

DB Design

Group No: 7

Student Name	Student Number
Asma Saleh	

Contents

PART I: Analysis

1 Problem Definition and Data Requirements

1.1 Problem Description

Any successful business owner knows that to run a successful business, one must have a database management system that stores all needed data when analysing, improving, and maintaining the profits and contracts. While managing records concerning branches, employees, and departments. Making sure that clients are satisfied with the given services that the business provides them with.

That is why when the owner of Smash Up GYM contacted us to design a database to help him organize the GYM's management system, increase, or maintain its profits, we made sure to design a database that satisfies his requirements in keeping records about the employees, members, health records, contracts, lockers, and equipment. Organizing stored data about the branches, departments, classes, membership plans, and events.

The system will also help in linking the branch's information together. It will help in making accurate and precise estimations about the total net profits. It will also help in making decisions about the branches and membership plans. Our goal for Smash Up GYM is to design a comprehensive database, as well as make it easy to deal with in a highly efficient manner, to meet the owner's requirements.

1.2 Data Requirements

1. **Employees:**

   An employee would have the following data:

   - Employee ID
   - Name (First name, Middle name, Last name)
   - Phone number
   - Date of birth (Day, Month, Year)
   - Age
   - Address (City, Street, Building number)
   - Gender (F/M)
   - Bank account
   - Salary

2. **Members:**

   Members of the GYM must have the following information stored in the system:

   - Member ID
   - Name (First name, Middle name, Last name)
   - Date of birth (Day, Month, Year)
   - Age
   - Gender (F/M)
   - Social state (Single/Married)
   - Phone number
   - Email

3. **Branches:**

   All branches must have the following data stored in the system:

   - Branch No
   - Address (City, Street, Building number)
   - Phone number
   - Working hours (Opening hour, Closing hour)
   - Branch's capacity
   - Number of members
   - Number of employees

4. **Departments:**

   Each department will have:

   - Department number
   - Name (Maintenance, Customer service, Management, Medical care, Coaching staff)
   - Number of employees

5. **Classes:**

   To organize the classes, each class must have the following data stored:

   - Class reference number
   - Place (Days, Hours, Room No)
   - Activity

6. **Lockers:**

   All lockers will have the following information:

   - Locker No
   - Password
   - State (Occupied/Available)

7. **Membership plans:**

   All membership plans that the GYM offers will have:

   - Membership plan ID
   - Plan's name (Diamond/Golden/Platinum)
   - Services {Personalized meal plan, Massage sessions, Personal trainer, Nursery}
   - Duration (3 months/6 months/12 months)
   - Price

8. **Sports Equipment:**

   An equipment in the GYM will have the following data:

   - Machine No
   - Name
   - State (Out of service / In service)

9. Events:

Any event that is hosted by the GYM must have:

- Event's name
- Start date (Day, Month, Year)
- Duration (Number of Days, Number of Hours per day)
- Location (City, Hosting place name)

10. Health records:

The health records of every member will be stored in the database as the following:

- Date of record
- Height
- Weight
- Blood type
- Blood pressure
- Body fat
- BMI
- Body water
- Body mass
- Muscle mass

11. Contracts:

To maintain all contracts, the following data must be stored:

- Contract ID
- Start date (Day, Month, Year)
- End date (Day, Month, Year)
- Date of payment (Day, Month, Year)
- Payment method (Cash/Credit card)
- Applied discounts
- Applied taxes

1.3 Business Rules

To manage the GYM's business efficiently without any conflicts, there are some rules, specified and stated by the owner, that we must take into consideration whilst designing the database.

1. **Employees**

   Each employee's contact information and bank account must be stored.

2. **Members**

   A member must be at least 16 years old to apply to the GYM.

3. **Branches**

   Depending on the branch's location, each branch will have different working hours. Overall working hours of a branch should be at least 12 hours.

4. **Departments**

   The department's number and name should be stored.

5. **Classes**

   To organize classes information and distinguish them, each class must have a reference number. To prevent overlapping, no two classes can occur at the same time, at the same room.

6. **Lockers**

   To determine the locker's availability, each locker's state must be recorded.
   The locker's number, password, and owner's ID must be saved.

7. **Membership plans**

   All plans must have distinct IDs. The services, duration, and price of plans must be stored.

   For example, the Golden plan with the duration of 3 months will have a distinct ID than the Golden plan with 6 months duration. Same goes for Diamond, and Platinum plan.

8. **Sports Equipment**

   Each device will have a unique number to distinguish it. The device's state and location must be recorded, in case of an equipment that is out of service, the maintenance administrator will manually request to fix it.

9. Events

An event's location, start date, duration, and name must be saved in the system, the supervisor's ID must be stored.

10. Health records

The health records for every member must be measured and saved. Also, the date of when the measurements were taken must be stored.

11. Contracts

In any contract, the details must be stated, start and end date, the payment method (cash or credit card), and any discounts or taxes that were applied should be recorded.

- Each employee must belong to a department, and each department must have at least one employee belonging to it. Number of employees in each department must be counted.
- Each employee must work in at most one branch, and each branch must have at least one employee. The number of current working employees in a branch must be counted.
- A branch must have several registered members, each member must go to one branch. The number of members applied in the branch should not exceed the branch's capacity.
- A branch must be managed by only one employee.
- A department must be managed by only one employee.
- A class must be instructed by a single instructor, and an instructor may instruct several classes.
- A locker can be owned by a single member. And a member can own only one locker.
- Each branch includes several lockers. Each locker must be found in a specific branch.
- Each branch must have several sports equipment, and an equipment must be located at a specific branch.
- An event must be supervised by at least one supervisor, and a supervisor may supervise several events.
- Each member must have at least one or more health records, each record with different record dates. If a month has passed since the last measurements were taken, the health records should be measured again.
- Each member must be assigned to one membership plan that is specified by the contract. The total cost for the membership plan including the applied taxes and discounts that the member has paid for will be stored.

1.4 Intended Output of the system

Queries:

- Display information of an employee, Member, and Membership plan by using its ID.
- Display information of a locker, branch, department, class, and equipment by using its number.
- To display a member health records, use the member's ID.
- Check the state of availability of any locker by using its number.
- Check the state of a specific machine (Out of service / In service) by using its number.
- Display any event information by using its name.
- If a branch cannot take any more members, the responsible employee can look for other available branches based on the request of the client.
- Display sports equipment's state of a specific branch.
- Display the classes schedule either by days or by specific hours or both.
- Display the classes held by a specific instructor using the instructor ID.

Transactions:

- An employee can update their address, phone number, and bank account.
- Modifying salary, department, and branch of an employee by the DBA.
- A member can update their contact information such as phone number, email, and social state.
- If a contract has expired, the contract data and the member information will be removed.
- Modifying a branch's number of working employees and members, the manager and working hours.
- Updating a department number of employees and replacing its manager.
- Modify any class's days, hours, room No, and instructor.
- Insert a new class to the schedule or remove one.
- Modify a locker's owner, password, and state.
- Adding a new plan to the system or deleting a stored plan.
- Modifying an event's start date, location, and its duration in case of need.
- Insert a new health record measurement of a member.

Invoices:

When a contract deal is done, the member will receive an invoice that contains the following information:

- Contract ID
- Member ID
- Plan's name
- Branch No
- Start date (Day, Month, Year)
- End date (Day, Month, Year)
- Date of payment (Day, Month, Year)
- Payment method (Cash/Credit card)
- Membership plan price
- Applied discounts
- Applied taxes
- Total cost

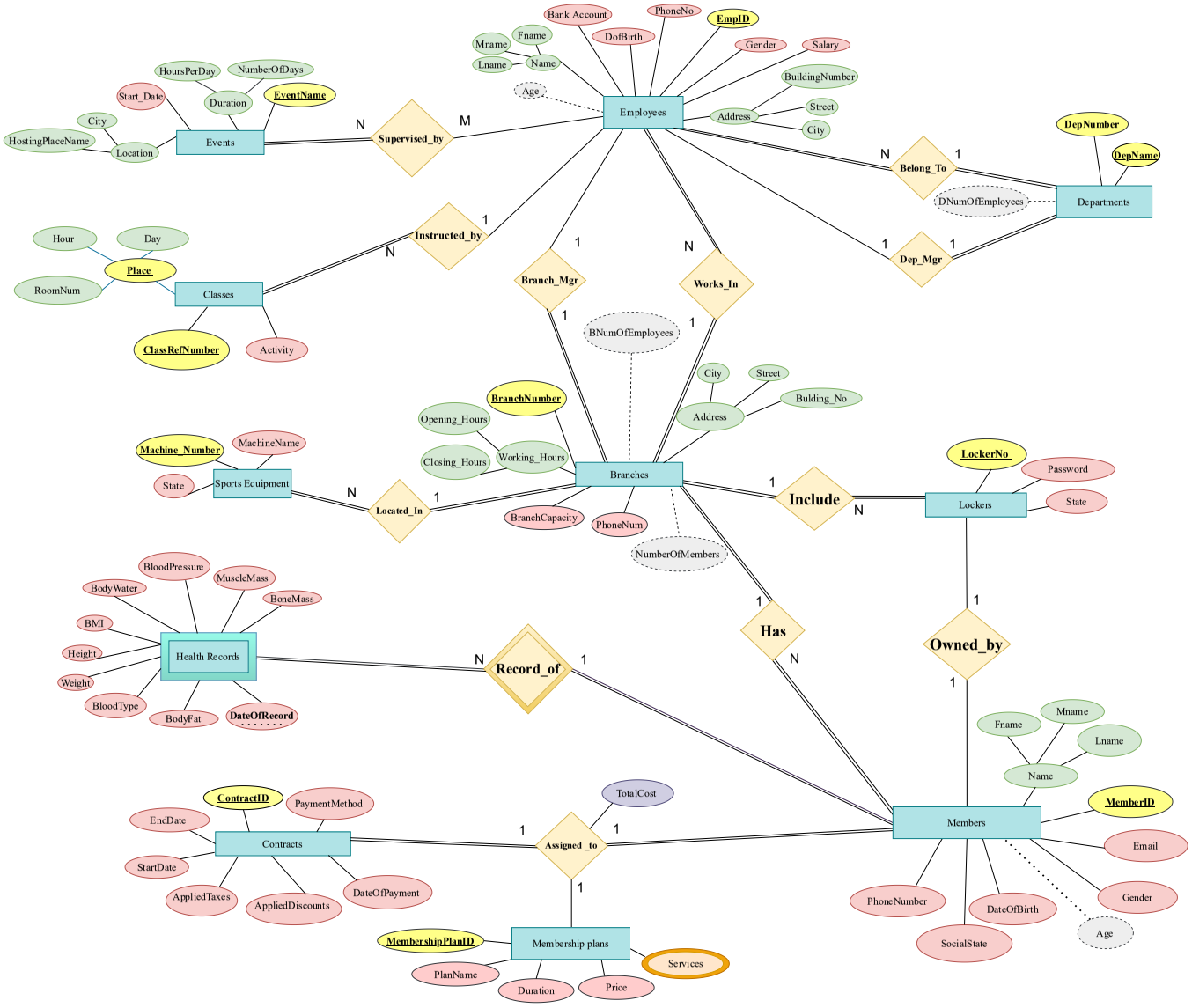
Calculations:

Using contracts information, we can calculate the monthly, and yearly profits of a certain branch. Based on that, we can decide whether a branch needs an adjustment in the number of employees. By using the date of payment of all contracts, we can determine the month with the highest enrolment rate and use that to offer exclusive deals and discounts to attract more customers to register in the GYM during these seasonal months. By observing the number of registered members in each branch, we can check if a branch has made the estimated profits. Based on that, a decision whether a branch should be shut down or not will be made.

PART II: DB DEISGN

2 ER Diagram Design

2.1 ER Diagram


Employees
Bank Account
PhoneNo
EmpID
DofBirth
Fname
Mname
Lname
Name
Age
Gender
Salary
BuildingNumber
Address
Street
City
Events
HoursPerDay
NumberOfDays
Duration
EventName
Start_Date
City
HostingPlaceName
Location
Supervised_by
N
M
Belong_To
1
N
Departments
DepNumber
DepName
DNumOfEmployees
Dep_Mgr
Instructed_by
Hour
Day
Place
RoomNum
Classes
ClassRefNumber
Activity
Branch_Mgr
Works_In
BNumOfEmployees
Branches
BranchNumber
Opening_Hours
Closing_Hours
Working_Hours
Address
City
Street
Bulding_No
BranchCapacity
PhoneNum
NumberOfMembers
Machine_Number
MachineName
State
Sports Equipment
Located_In
Include
LockerNo
Password
State
Lockers
Owned_by
Has
Record_of
Health Records
BloodPressure
MuscleMass
BodyWater
BoneMass
BMI
Height
Weight
BloodType
BodyFat
DateOfRecord
Members
Fname
Mname
Lname
Name
MemberID
Email
Gender
Age
DateOfBirth
SocialState
PhoneNumber
Assigned_to
TotalCost
Contracts
ContractID
PaymentMethod
EndDate
StartDate
AppliedTaxes
AppliedDiscounts
DateOfPayment
Membership plans
MembershipPlanID
PlanName
Duration
Price
Services
weak entity
entity
composite attribute
key attribute
simple attribute
multivalue attribute
relationship attribute
derived attribute
weak relationship
relationship

2.2 Design of Business Rules

Business Rule	Design Decisions	Justification (if any)
A member must be at least 16 years old to apply to the GYM.	Derived attribute called **Age** that is computed from the subtraction of date of birth attribute and the current date.	If the computed age value was under 16 then this member instance validates the age constraint which we will show in upcoming phases (Implementation phase) how the database will deal with it.
Overall working hours of a branch should be at least 12 hours.	This done by having a composite attribute called **Working_Hours** which is composed of **Opening_Hours** and **Closing_Hours** attributes.	To make sure that the overall working hours is at least 12 in total, we can subtract the closing hour from the opening hour and check if the result was equal to or greater than 12.
To prevent overlapping, no two classes can occur at the same time, at the same room.	This is accomplished by making the classes entity have two key attributes, first one is the **ClassRefNumber**, and the other one is the **Place** attribute which is a composite attribute made up of: room number, hour, and day simple attributes.	From the perspective of the relation schema, which prevents a relation from having more than one primary key, we chose the place (which is made up of day, hour, and room number) as the primary key, because the unique combination of this composite key will prevent overlapping, and the class reference number will be the unique attribute.
The device's state and location must be recorded, in case of an equipment that is out of service, the maintenance administrator will manually request to fix it.	The **State** of a device is an attribute that can have either one of these two values: in service or out of service.	If the maintenance staff wishes to check which devices in a specific branch needs to be fixed, they can simply check the recorded states of the devices in a specific branch.

Each branch must have several sports equipment, and an equipment must be located at a specific branch.	This is implemented by creating a one-to-many relationship between the branches and sports equipment entities that is called **Located_In**.	This relationship shows total participation from branches entity because a branch cannot be classified as a gym branch without having sports equipment. And it shows total participation from the sports equipment side because an equipment needs to be located in a branch in order to be used by members.
Each employee must Belong to a department, and each department must have at least one employee belonging to it. Number of employees in each department must be counted.	This is implemented by creating a many-to-one relationship between the employees and departments entities that is called **Belong_To**. **Number of employees** will be a derived attribute in the departments entity.	The derived attribute number of employees is computed from the relationship **Belong_To** which counts the total number of employees in each specific department. Also, this relationship shows total participation from the departments because a department cannot exist without having employees. At the same time, it shows total participation from the employees because each employee must belong to one department.
Each employee must work in at most one branch, and each branch must have at least one employee. The number of current working employees in a branch must be counted.	This is implemented by creating a many-to-one relationship between the employees and branches entities that is called **Works_In**. **Number of employees** will be a derived attribute in the branches entity.	The derived attribute number of employees is computed from the relationship **Works_In** which counts the total number of employees currently working in each specific branch. Also, this relationship shows total participation from both participating entities. From the branches side, a branch does not exist without having employees. At the same time, an employee must be working in a single branch.

A branch must have several registered members, each member must go to one branch. The number of members applied in the branch should not exceed the branch's capacity.	This is implemented by creating a one-to-many relationship between the branches and members entities that is called **Has**. **Number of members** will be a derived attribute in the branches entity.	The derived attribute number of members is computed from the relationship **Has** which counts the total number of members in each specific branch, this computed value should not exceed the specific capacity of the branch. also, this relationship shows total participation from both participating entities. A branch cannot exist without having registered members, and each member must be registered in a single branch.
A branch must be managed by only one employee.	This is implemented by creating a one-to-one relationship between the employees and branches entities that is called **Branch_Mgr**.	This relationship shows total participation from the branches entity, because all branches must have a manager, and shows partial participation from the employees entity, because not all employees have to manage a branch.
A department must be managed by only one employee.	This is implemented by creating a one-to-one relationship between the employees and departments entities that is called **Dep_Mgr**.	This relationship shows total participation from the departments entity, because all departments must have a manager, and shows partial participation from the employees entity, because not all employees have to manage a department.
A class must be instructed by a single instructor, and an instructor may instruct several classes.	This is implemented by creating a one-to-many relationship between the employees and classes entities that is called **Instructed_by**.	This relationship shows total participation from the classes entity, since a class must be instructed by an employee, and partial participation from the employees entity, since not all employees are instructors.
A locker can be owned by a single member. And a member can own only one locker.	This is implemented by creating a one-to-one relationship between the lockers and members entities that is called **Owned_by**.	This relationship shows partial participation from both participating entities, because not all lockers are occupied by members, and not all members own lockers.

Each branch includes several lockers. Each locker must be found in a specific branch.	This is implemented by creating a one-to-many relationship between the branches and lockers entities that is called **Include**.	This relationship shows total participation from both participating entities. Because each branch needs to have lockers for its members and each locker needs to be located in a specific branch.
An event must be supervised by at least one supervisor, and a supervisor may supervise several events.	This is implemented by creating a many-to-many relationship between the events and employees entities that is called **Supervised_by**.	This relationship shows total participation from the events entity, because each event must be supervised by one or more supervisors to manage and organize the event, and it shows partial participation from the employees entity, because not all employees are qualified to supervise an event.
Each member must have at least one or more health records, each record with different record dates.	This is implemented by creating a many-to-one relationship between the members and health records entities that is called **Record_Of**. This relationship is considered as an identifying relationship for the weak entity that is health records, and the owner entity is the members entity. The **date of record** attribute for the health records entity is its partial key, and the key that distinct each health record instance is a unique combination of the **member id** and the **date of record**.	This relationship is one-to-many because several health records will belong to one member, only difference between each record is the date of when it was taken, and each member may have many health records. This relationship must show total participation from the weak entity (health records) side because it is existence dependent and shows total participation from the members entity, because all members of the gym will have their measurements taken each month.
If a month has passed since the last measurements were taken, the health records should be measured again.	The **Date of record** for the health records entity is an attribute that takes the role of the date of when the records of a member were measured.	The medical care administrator will check the last month's recorded health measurements of some members to ensure that a month has passed since the records were last taken, and to also check if they need to retake them again for these members.

Each member must be assigned to one membership plan that is specified by the contract. The total cost for the membership plan including the applied taxes and discounts that the member has paid for will be stored.	This is implemented by creating a ternary relationship that is called **Assigned_To,** which is one-to-one among the members, membership plans and contracts entities. The **total cost** is an attribute of the ternary relationship.	This relationship shows total participation from the contracts side because a contract instance will not exist without being assigned to a specific member, and it shows total participation from the members side because when a member is assigned to a membership plan in the gym, they must have a contract that shows all the registration details. The relationship also shows partial participation from the membership plans entity because a membership plan can exist in the system without having to be assigned to any members.

3 ER-to-logical Schema Mapping

3.1 Mapping of Regular Entity Types

To map regular strong entity types from ER to relation schema, we must create a relation that includes all the simple attributes that form the entity, unlike the ER diagram, relational schema disallows an entity to have more than one key attribute as the primary key for the relation , that is why when mapping an ER entity to relation schema we must choose one of its key attributes as the primary key for the relation schema. Also, if an entity had an attribute that is composed of more than one simple attribute, we map the simple attributes that form this composite attribute to the relation schema.

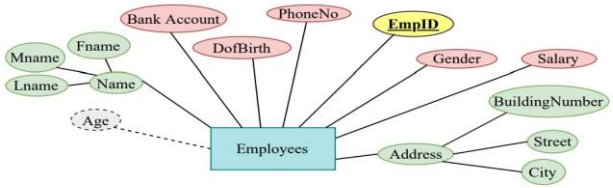
- Mapping of Employees entity

  We mapped all the simple attributes in the employees entity, and we also mapped the simple attributes that form the composite attribute (Name and Address) in the entity. The primary key of the employees relational schema is the key attribute of the employees entity which is EmpID.

  The age attribute is a derived attribute that will not be mapped when mapping the entity to relational schema, the derived attributes will be handled at the implementational phase.

Employees

EmpID	Fname	Mname	Lname	DofBirth	PhoneNo	Gender	Salary	Bank Account	City	Street	BuildingNumber



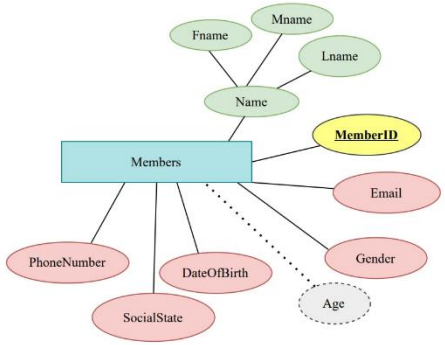
- Mapping of Members entity

We mapped all the simple attributes in the members entity, and we also mapped the simple attributes that form the composite attribute (Name) in the entity. The primary key of the members relational schema is the key attribute of the members entity which is MemberID.

The age attribute is a derived attribute that will not be mapped when mapping the entity to relational schema, the derived attributes will be handled at the implementational phase.

Members

MemberID	Fname	Mname	Lname	Email	Gender	DateOfBirth	SocialState	PhoneNumber



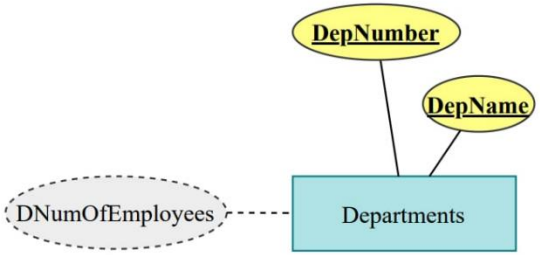
- Mapping of departments entity

Since the departments entity contain two key attributes which are the DepNumber and DepName and contains no simple attributes, we chose DepNumber as the primary key of the departments relational schema and the other key attribute (DepName) will be a candidate key (unique).

The DNumOfEmployees attribute is a derived attribute that will not be mapped when mapping the entity to relational schema, the derived attributes will be handled at the implementational phase.

Departements

DepNumber	DepName

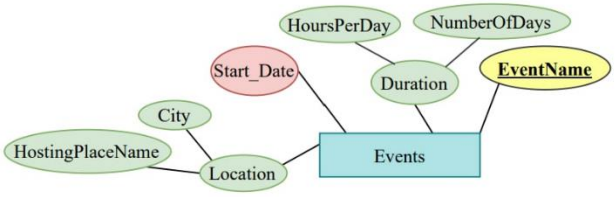


- Mapping of events entity

We mapped the simple attribute in the events entity, and we also mapped the simple attributes that form the composite attribute (Location and Duration) in the entity. The primary key of the events relational schema is the key attribute of the events entity which is EventName.

Events

EventName	Start_Date	NumberOfDays	HoursPerDay	City	HostingPlaceName



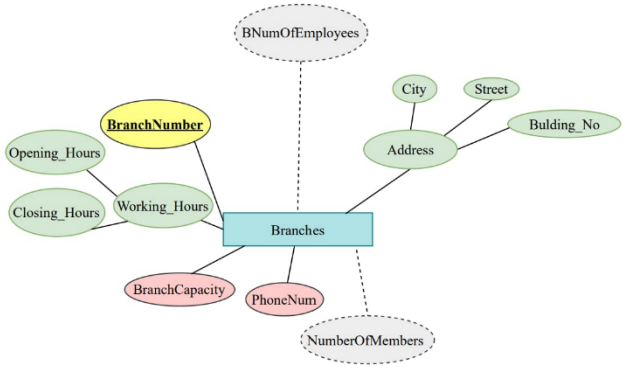
- Mapping of branches entity

We mapped all the simple attributes in the branches entity, and we also mapped the simple attributes that form the composite attribute (Working_Hours and Address) in the entity. The primary key of the branches relational schema is the key attribute of the branches entity which is BranchNumber.

The BNumOfEmployees attribute and NumberOfMembers are derived attributes that will not be mapped when mapping the entity to relational schema, the derived attributes will be handled at the implementational phase.

Branches

BranchNumber	City	Street	Building_No	PhoneNum	BranchCapacity	Opening_Hours	Closing_Hours

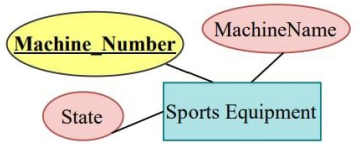


- Mapping of sports equipment entity

We mapped all the simple attributes in the sports equipment entity. The primary key of the sports equipment relational schema is the key attribute of the sports equipment entity which is Machine_Number.

Sports Equipment

Machine_Number	MachineName	State

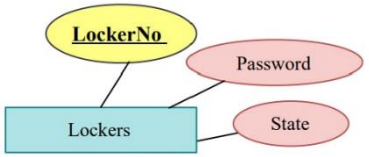


- Mapping of lockers entity

We mapped all the simple attributes in the lockers entity. The primary key of the lockers relational schema is the key attribute of the lockers entity which is LockerNo.

Lockers

LockerNo	Password	State

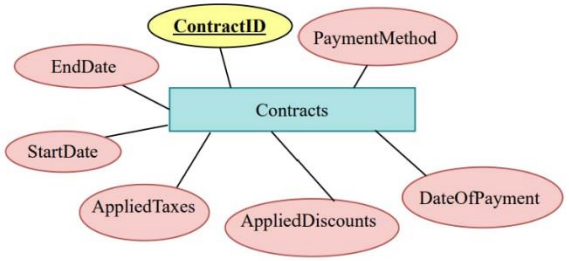


- Mapping of contracts entity

We mapped all the simple attributes in the contracts entity.The primary key of the contracts relational schema is the key attribute of the contracts entity which is ContractID.

Contracts

ContractID	StartDate	EndDate	PaymentMethod	AppliedDiscounts	AppliedTaxes	DateOfPayment



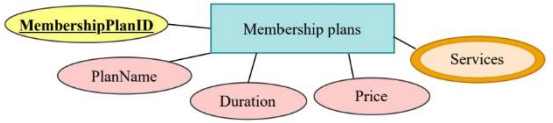
- Mapping of membership plans entity

We mapped all the simple attributes in the membership plans entity. The primary key of the membership plans relational schema is the key attribute of the membership plans entity which is MembershipPlanID.

The services attribute is a multivalued attribute that will not be mapped in the same relational schema when mapping the regular entity type, but we will show how we will map it at section 3.6.

Membership Plans

MembershipPlanID	PlanName	Duration	Price

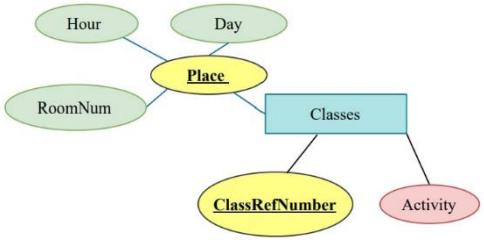


- Mapping of classes entity

We mapped the simple attribute in the classes entity, and we also mapped the simple attributes that form the composite attribute (Place) in the entity. Since the classes entity contain two key attributes which are the Place and ClassRefNumber, we chose Place as the primary key of the classes relational schema and the other key attribute (ClassRefNumber) will be a candidate key (unique). The primary key Place of classes relational schema is composite, so the set of the simple attributes that form it (Hour,Day,RoomNum) will together form the composite primary key of the relational schema.

Classes

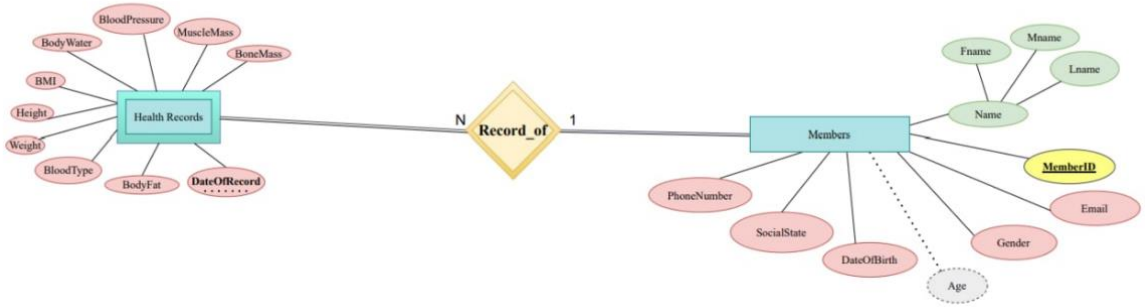
Day	**Hour**	**RoomNum**	ClassRefNumber	Activity



3.2 Mapping of Weak Entity Types

- Mapping of health records weak entity

  The weak entity health records is owned by the members entity. When mapping the health records entity, we will create a relation schema which will contain all of the simple attributes (as well as the simple attributes of any composite attribute if the weak entity had one) of the entity. To show the relationship between the owning entity (members) as well as the weak entity (health records), we will include in the relation schema of health records a foreign key which is MemId that references the primary key of the owner entity (members). The combination of the foreign key (MemId) and the partial key (DateOfRecord) will together form the primary key of the health records relation schema.



Members

MemberID	Fname	Mname	Lname	Email	Gender	DateOfBirth	SocialState	PhoneNumber

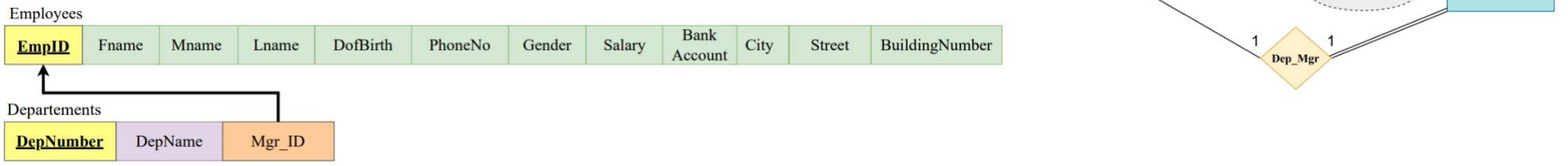
Health Records

MemId	DateOfRecord	BodyFat	BloodType	Weight	Height	BMI	BodyWater	BloodPressure	MuscleMass	BodyMass

3.3 Mapping of Binary 1-1 Relationship Types

- Mapping of Dep_Mgr relationship

  To map the binary 1:1 Dep_Mgr relation between the Departments and Employees entities, we will consider the approach of adding a foreign key to one of the participating entities in the relation, it is better to include a foreign key in the entity that shows total participation to reduce Null values in the tuples of the relation schema. That is why we chose to add a foreign key (Mgr_ID) in the departments relation that references the primary key (EmpId) of the employees relation.



- Mapping of Branch_Mgr relationship

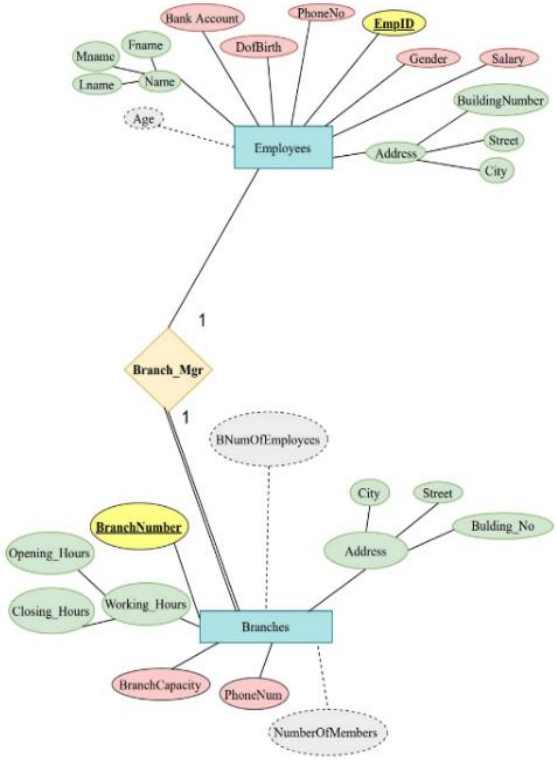
To map the binary 1:1 Branch_Mgr relation between the Branchess and Employees entities, we will consider the approach of adding a foreign key to one of the participating entities in the relation, it is better to include a foreign key in the entity that shows total participation to reduce Null values in the tuples of the relation schema. That is why we chose to add a foreign key (Manager_ID) in the branches relation that references the primary key (EmpId) of the employees relation.

Employees

EmpID	Fname	Mname	Lname	DofBirth	PhoneNo	Gender	Salary	Bank Account	City	Street	BuildingNumber

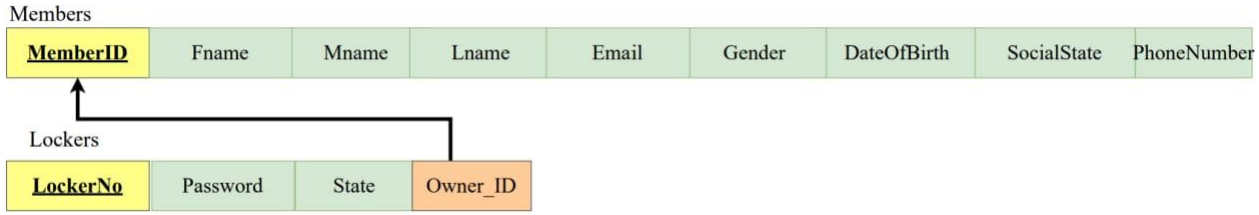
Branches

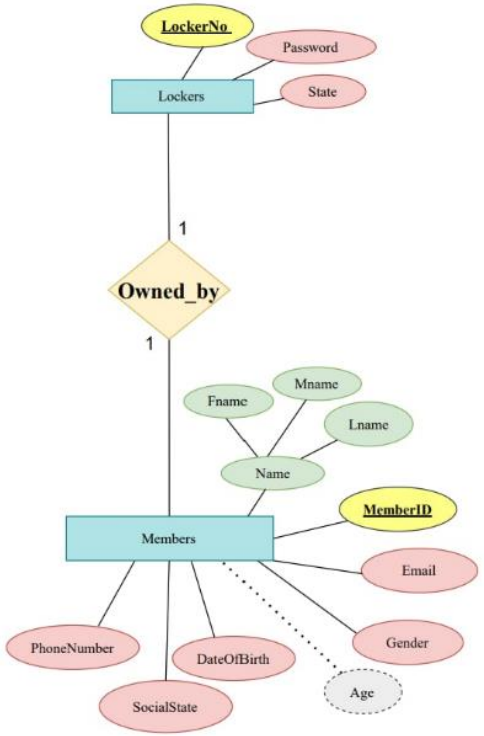
BranchNumber	City	Street	Building_No	PhoneNum	BranchCapacity	Opening_Hours	Closing_Hours	Manager_ID



- Mapping of Owned_By relationship

To map the binary 1:1 Owned_by relation between the Lockers and Members entities, we will consider the approach of adding a foreign key to one of the participating entities in the relation , it is better to include a foreign key in the entity that shows total participation to reduce Null values in the tuples of the relation schema, if both of the participating entities show partial participation in the relationship , it will make no difference to choose which relation schema to add the foreign key in. We chose to add a foreign key (Owner_ID) in the Lockers relation that references the primary key (MemberID) of the members relation.





3.4 Mapping of Binary 1-N Relationship Types

To map a 1:N relationship, we will include a foreign key that references the primary key of the relation that represent the participating entity at the 1-side of the relationship in the relation that represent the participating entity at the N-side of the relationship.

- Mapping of Has relationship

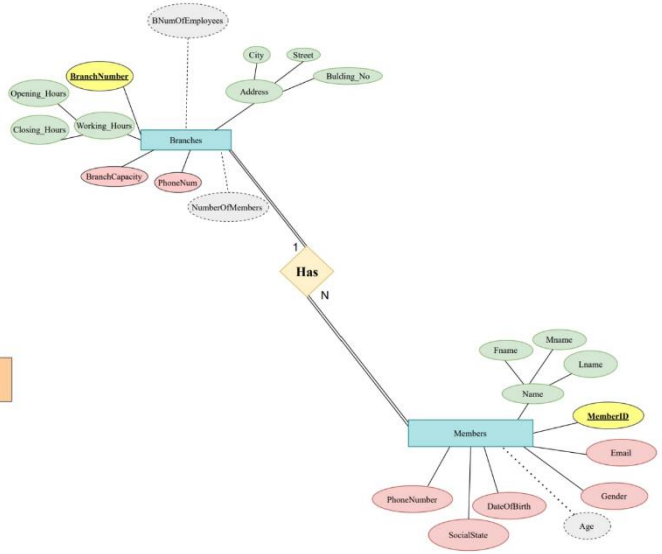
To map the Has relationship between the Branches and members entities, we will include a foreign key in the relation that represents the entity at the N-side of the relationship which is members. The foreign key BranchNo in the members relation will reference the primary key of the branches relation that represents the branches entity at the 1-side of the relationship.

Branches

BranchNumber	City	Street	Building_No	PhoneNum	BranchCapacity	Opening_Hours	Closing_Hours	Manager_ID

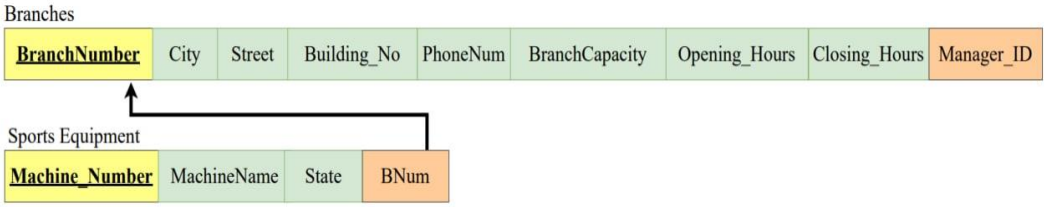
Members

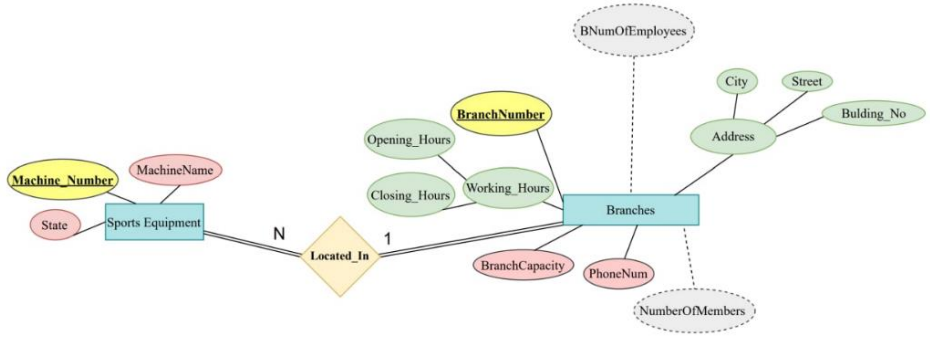
MemberID	Fname	Mname	Lname	Email	Gender	DateOfBirth	SocialState	PhoneNumber	BranchNo



- Mapping of Located_In relationship

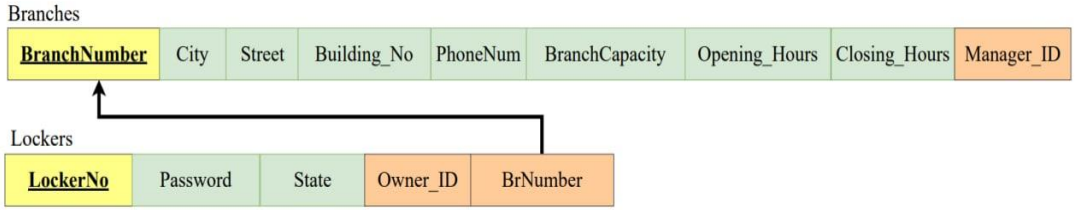
To map the Located_In relationship between the Branches and sports equipment entities, we will include a foreign key in the relation that represents the entity at the N-side of the relationship which is sports equipment. The foreign key BNum in the sports equipment relation will reference the primary key of the Branches relation that represents the branches entity at the 1-side of the relationship.

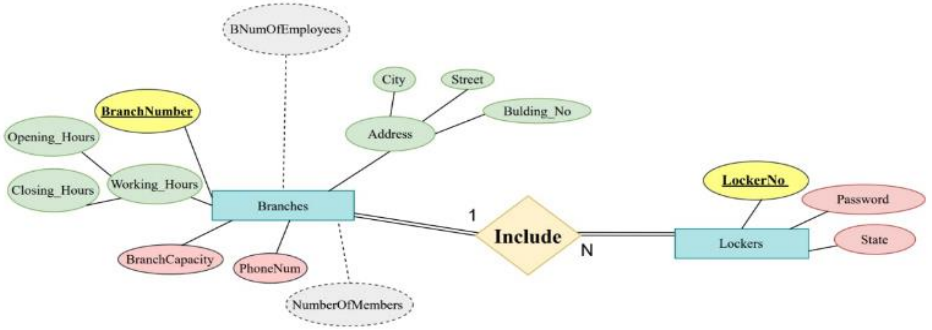




- Mapping of Include relationship

To map the Include relationship between the Branches and Lockers entities, we will include a foreign key in the relation that represents the entity at the N-side of the relationship which is Lockers. The foreign key BrNumber in the Lockers relation will reference the primary key of the branches relation that represents the branches entity at the 1-side of the relationship.





- Mapping of Instructed_By relationship

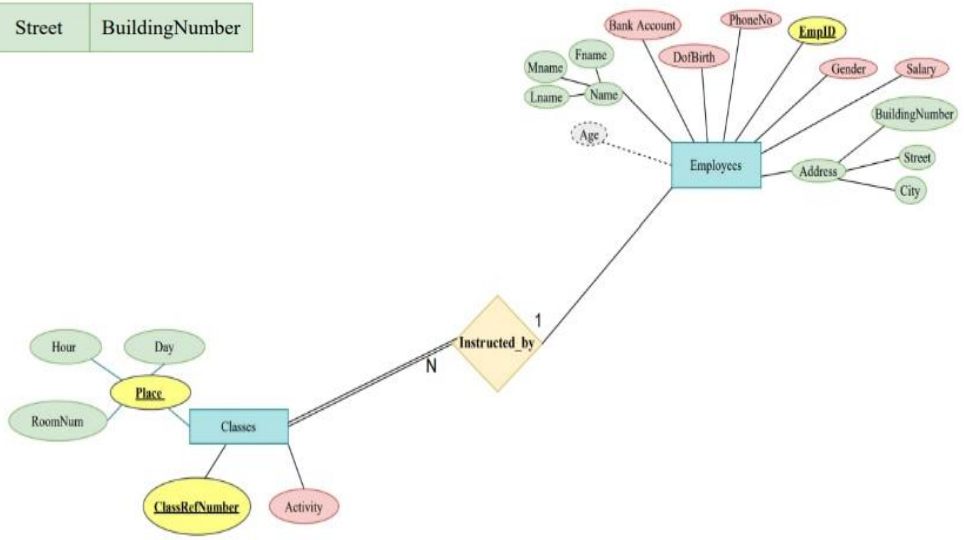
  To map the Instructed_By relationship between the Classes and Employees entities, we will include a foreign key in the relation that represents the entity at the N-side of the relationship which is Classes. The foreign key InstructerID in the Classes relation will reference the primary key of the employees relation that represents the employees entity at the 1-side of the relationship.

Employees

EmpID	Fname	Mname	Lname	DofBirth	PhoneNo	Gender	Salary	Bank Account	City	Street	BuildingNumber

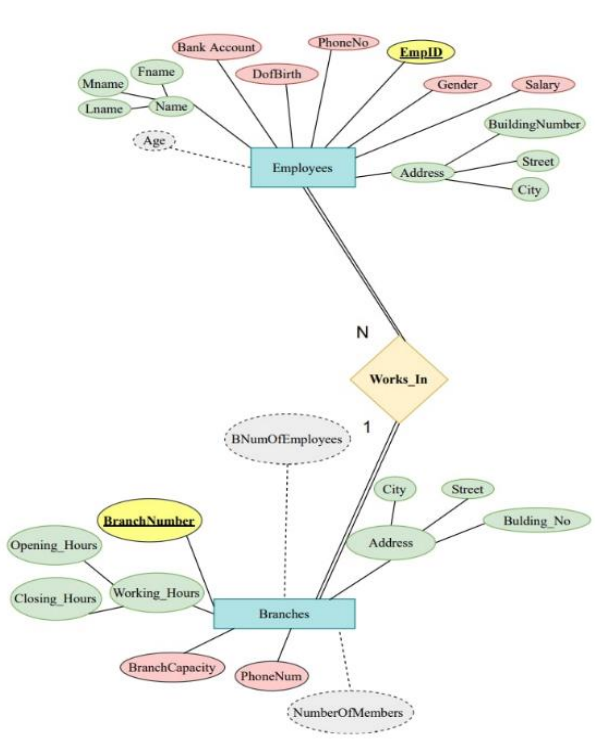
Classes

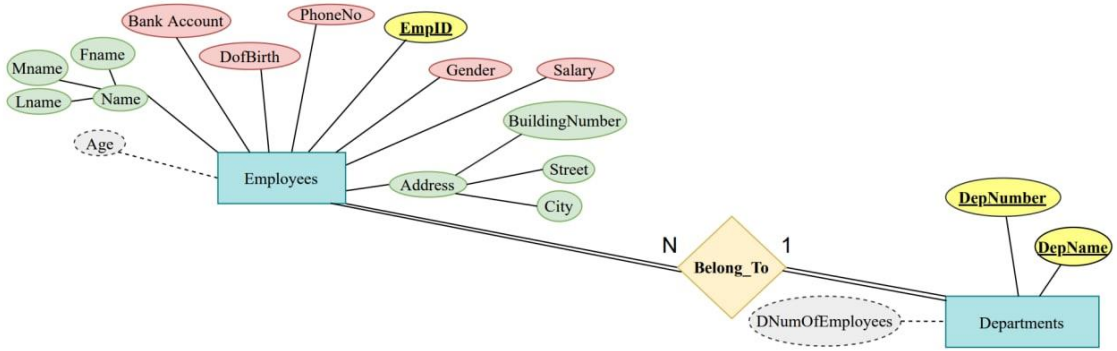
Day	**Hour**	**RoomNum**	ClassRefNumber	Activity	InstructerID

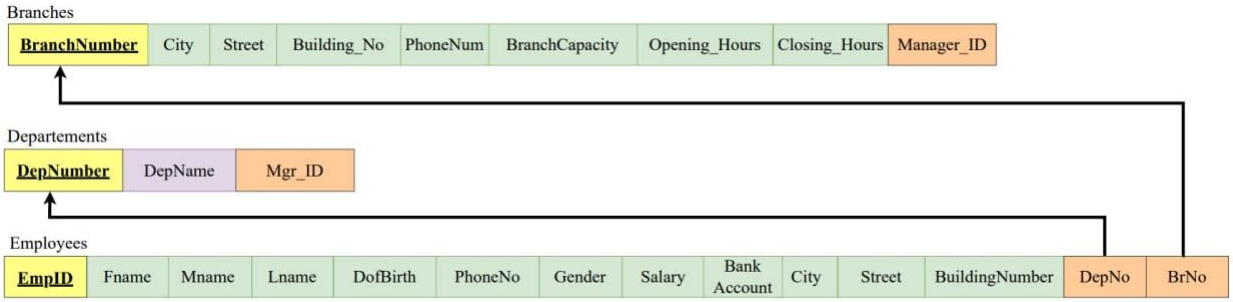


- Mapping Belong_To and Works_In relationships

When mapping the Belong_To relation between employees and departments as well as the Works_In relation between the employees and branches relation, we included two foreign keys in the employees relation because the entity employees is at the N-side of the both of the relationships. The first one is DepNo that references the primary key of the departments relation, and the second foreign key is BrNo that references the primary key of the branches relation, because both of these participating entities were at the 1-side of both of the relationships.



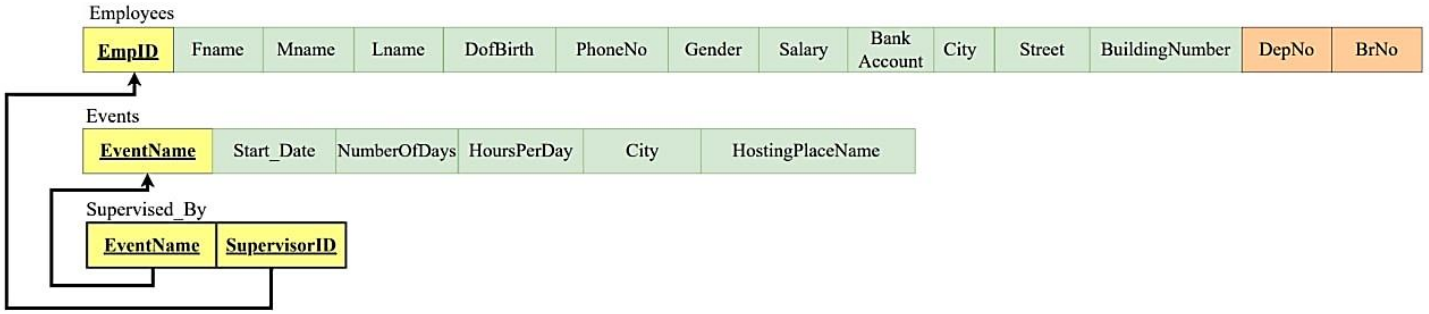


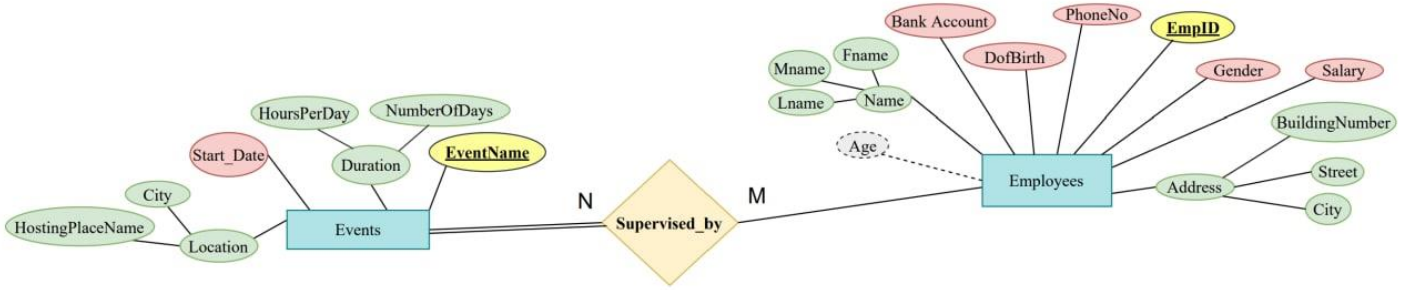


3.5 Mapping of Binary M-N Relationship Types

- Mapping of Supervised_By relationship

  To map the M:N relationship that is Supervised_By between the Events and Employees entities, we will create a new relation for the relationship. This relation will contain two foreign keys (EventName and SupervisorID) that references both participating entities (employees and events) primary keys. Together, these two foreign keys will form the primary key of this relation.

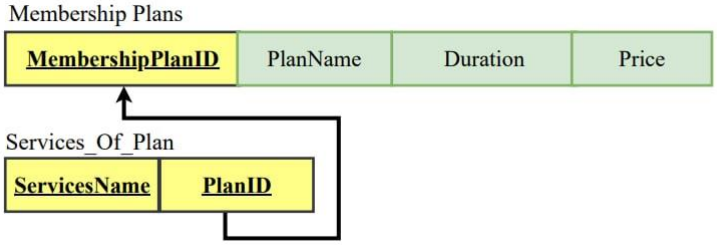


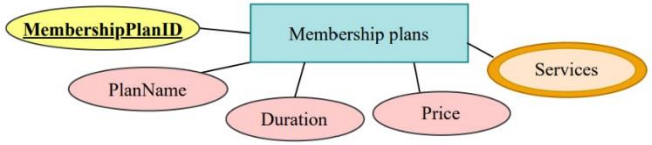


3.6 Mapping of Multivalued Attributes

- Mapping Services multivalued attribute of the membership plans entity

  To map the services multivalued attribute of the membership plans entity, we will create a sperate relation for this attribute. This relation will be called Services_Of_Plan and it will include a foreign key (PlanID) that references the primary key of the relation that represents the entity that has this multivalued attribute (membership plans), and it will also include an attribute (ServicesName) that represents each value of this multivalued attribute. These two values (PlanID and SevicesName) will be the primary key of the new relation.

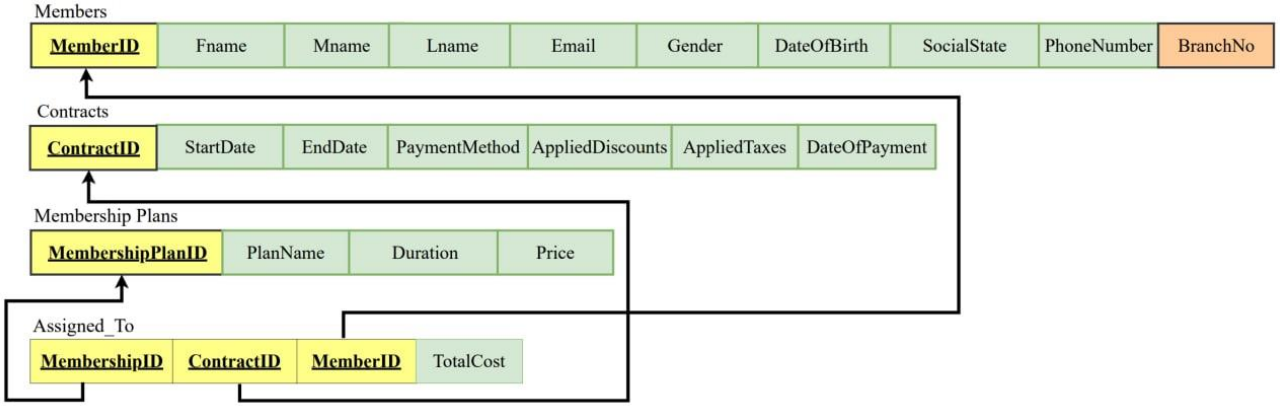


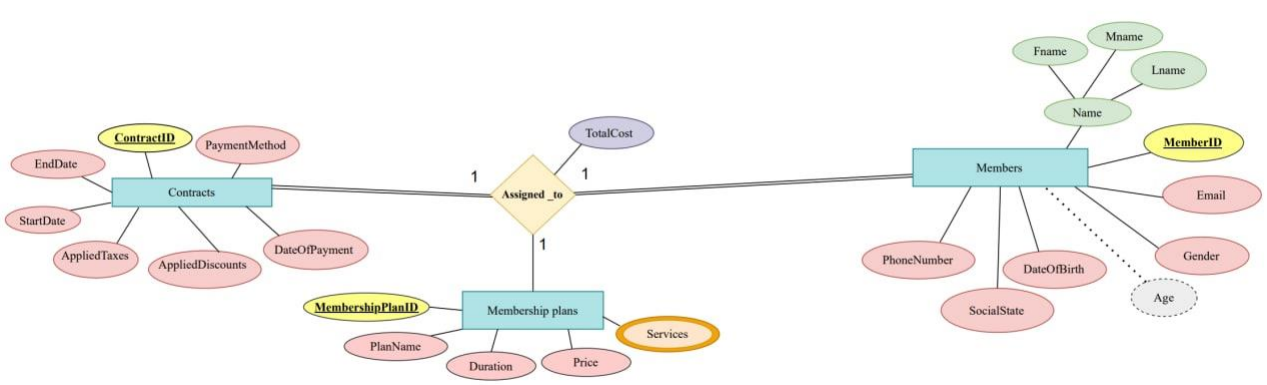


3.7 Mapping of N-ary Relationship Types

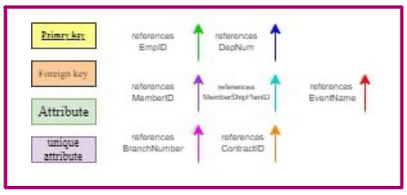
- Mapping of Assigned_To ternary relationship

To map the ternary relationship Assigned_To between contracts, members, and membership plans entities, we will create a new relation called Assigned_To to show this relationship. This relation will contain foreign keys (MembershipID, ContractID, and MemberID) that will reference all of the primary keys of the participating entities that are represented by these referenced relations. Also, the relation must include any attribute of the relationship itself which is TotalCost in this relationship. The primary of this relation will be composed of all the foreign keys in the relation.





3.9 Schema Diagram



Employees

EmpID	Fname	Mname	Lname	DofBirth	PhoneNo	Gender	Salary	Bank Account	City	Street	BuildingNumber	DepNo	BrNo

Members

MemberID	Fname	Mname	Lname	Email	Gender	DateOfBirth	SocialState	PhoneNumber	BranchNo

Lockers

LockerNo	Password	State	Owner_ID	BrNumber

Branches

BranchNumber	City	Street	Building_No	PhoneNum	BranchCapacity	Opening_Hours	Closing_Hours	Manager_ID

Departements

DepNumber	DepName	Mgr_ID

Health Records

MemId	**DateOfRecord**	BodyFat	BloodType	Weight	Height	BMI	BodyWater	BloodPressure	MuscleMass	BodyMass

Sports Equipment

Machine_Number	MachineName	State	BNum

Contracts

ContractID	StartDate	EndDate	PaymentMethod	AppliedDiscounts	AppliedTaxes	DateOfPayment

Events

EventName	Start_Date	NumberOfDays	HoursPerDay	City	HostingPlaceName

Membership Plans

MembershipPlanID	PlanName	Duration	Price

Supervised_By

EventName	**SupervisorID**

Services_Of_Plan

ServicesName	**PlanID**

Assigned_To

MembershipID	**ContractID**	**MemberID**	TotalCost

Classes

Day	**Hour**	**RoomNum**	ClassRefNumber	Activity	InstructerID

4 Normalization

4.1 First Normal Form

For a relational schema to be in first normal form, it should not contain composite attributes as well as multivalued attributes and nested relations.

Composite attribute is an attribute that is composed of more than one simple attribute, like the Address, Name, Place attributes in members, employees, and classes entities that are in the ER diagram in section 2.1.

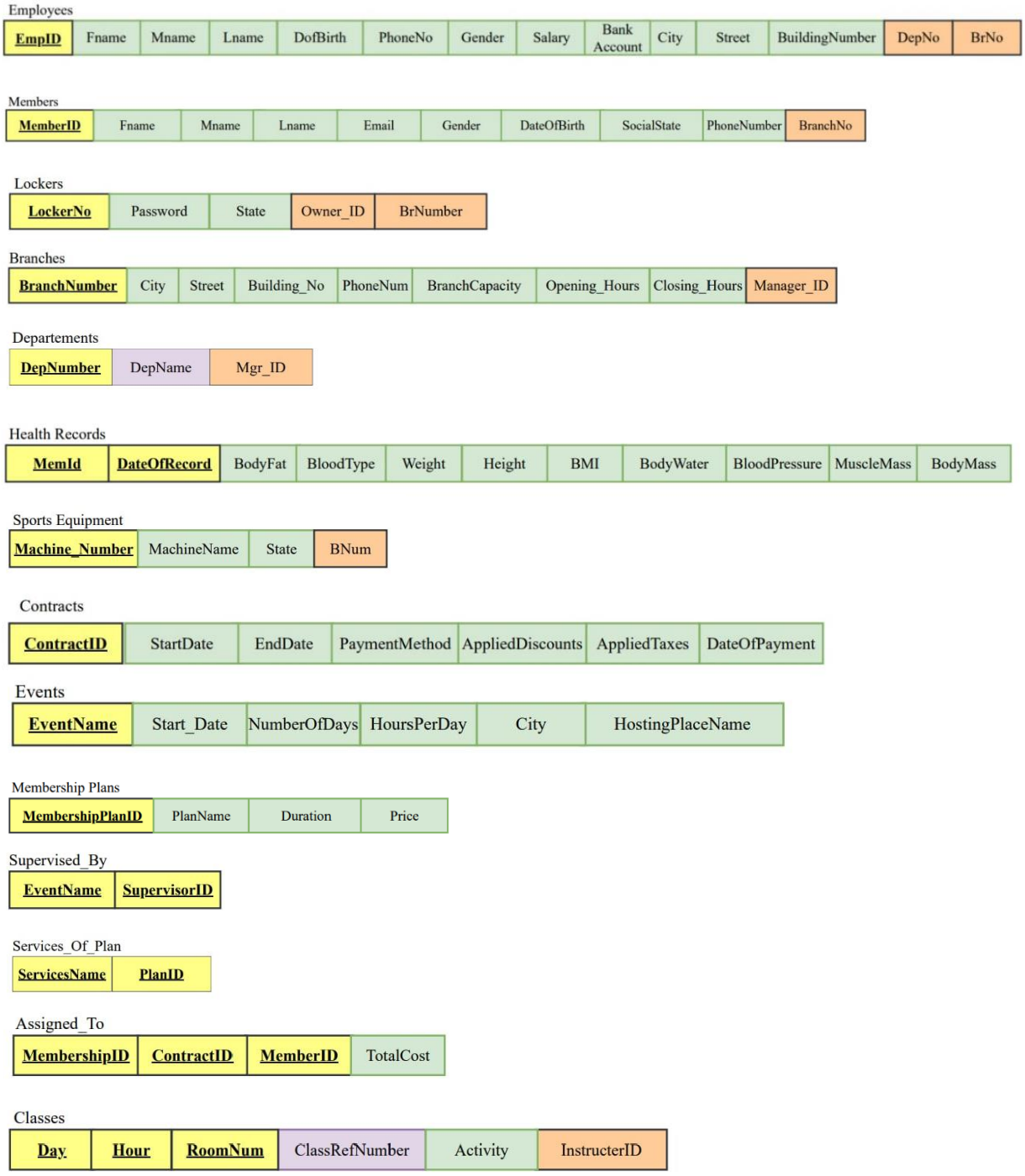
To map composite attributes to the first normal form of relational schema, we mapped the simple attributes that made the composite attribute into the relational schema. We have already done this when mapping the regular entities of our ER diagram into relational schemas shown in section 3.1.

Multivalued attribute is an attribute that can take several values for the same instance, in our ER diagram we had only one multivalued attribute in the membership plans entity which is services, shown in section 2.1.

In order to map multivalued attributes to the first normal form of relational schema, we mapped the multivalued attribute by separating it in a new relational schema along with the primary key of its regular entity that is represented in a relational schema. We have done this while mapping the services attribute of membership plans entity into a separate relation with the primary key of membership plans relation which is membership plan ID in section 3.6.

Nested relations are attributes that can represent relations within themselves. Our ER diagram does not include any nested relations.

The final relational schema diagram after normalizing it to the first normal form is as follows:



4.2 Second Normal Form

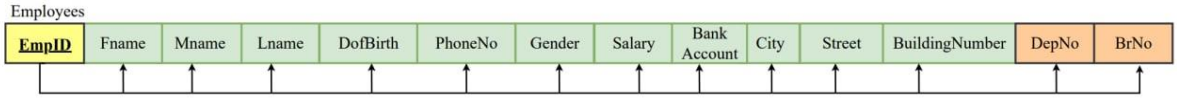
To normalize a relational schema into the second normal form, all non-prime attributes should be fully functionally dependent on the prime attribute(s) of the relation.

A prime attribute is an attribute that make up the relation's primary key, if a relation had a composite primary key, then the set of attributes that make up the primary key are called prime attributes.

From the above definition of prime attributes, we can conclude that any attribute in the relation that is not a part of the primary key will be called a non-prime attribute.

- Employees:

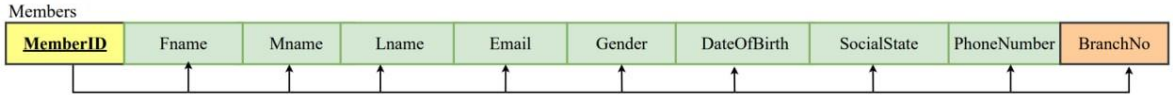
  The employees relation has only one prime attribute that is EmpID. All the non-prime attributes in the relation are fully functionally dependent on the prime attribute EmpID. Therefore, the employees relation is in second normal form.



{EmpID} -> Fname, Mname, Lname, DofBrith, PhoneNo, Gender, Salary, BankAccount, City, Street, BuildingNumber,DepNo,BrNo.

- Members:

  The Members relation has only one prime attribute that is MemberID. All the non-prime attributes in the relation are fully functionally dependent on the prime attribute MemberID. Therefore, the Members relation is in second normal form.

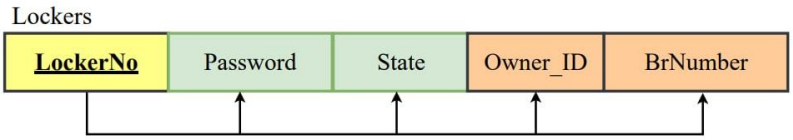


{MemberID}-> Fname,Mname,Lname,Email,Gender,DateOfBirth,SocialState,

PhoneNumber,Street, BranchNO.

- Lockers:

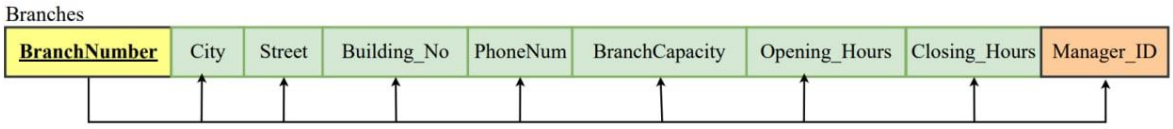
  The Lockers relation has only one prime attribute that is LockerNo. All the non-prime attributes in the relation are fully functionally dependent on the prime attribute LockerNo. Therefore, the Lockers relation is in second normal form.



  {LockerNo}-> Passwor,State,OwnerID,BrNumber.

- Branches:

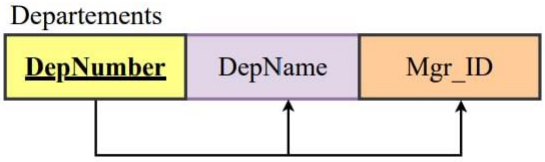
  The Branches relation has only one prime attribute that is BranchNumber. All the non-prime attributes in the relation are fully functionally dependent on the prime attribute BranchNumber. Therefore, the branches relation is in second normal form.



  {BranchNumber}-> City, Street,Building_No, BranchCapacity, Opening_Hours, Closing_Hours, Manger_ID.

- Departments:

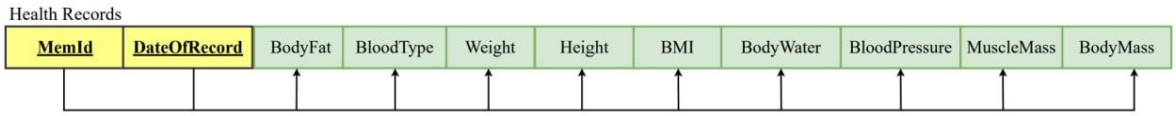
  The Departments relation has only one prime attribute that is DepNumber. All the non-prime attributes in the relation are fully functionally dependent on the prime attribute DepNumber. Therefore, the Departments relation is in second normal form.



  {DepNumber}-> DepName, Mgr_ID.

- Health Records:

  The Health Records relation has two prime attributes that are MemID and DateOfRecord. All the non-prime attributes in the relation are fully functionally dependent on both of the prime attributes MemID and DateOfRecord. Therefore, the Health Records relation is in second normal form.



  {MemID,DateOfRecord}-> BodyFat, BloodType, Weight, Height, BMI, BodyWater, BloodPressure, MuscleMass, BodyMass.

  The removal of any one of the two prime attributes means that the functional dependency will not hold anymore. Proving that all the non-prime attributes are fully functionally dependent on both prime attributes.

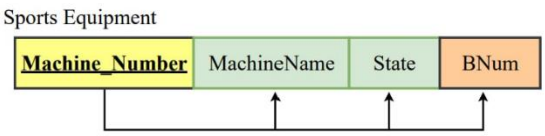
  Example:

  Suppose a member instance had a member ID that is 1234, and its health records were taken at 4th of March 2022, the same member had its health records remeasured at 4th of April 2022. That is why each non-prime attribute will be fully functional dependent on this combination of prime attributes. To show that each instance of the health records for this member is different, we chose to make both date of record as well as the member ID prime attributes (composite key) to enforce the functional dependency on the combination of both attributes.

  {1234,04/03/2022}-> body mass is deferent than {1234,04/04/2022} -> body mass. That is why each non-prime attribute will be fully functional dependent on this combination of prime attributes

- Sports Equipment:

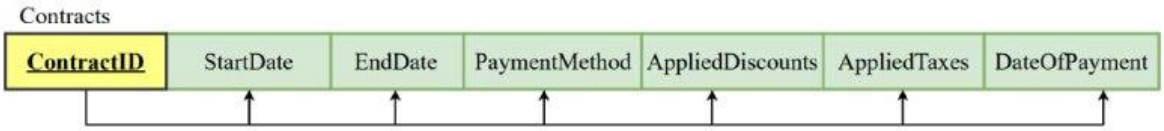
  The Sports equipment relation has only one prime attribute that is Machine_Number. All the non-prime attributes in the relation are fully functionally dependent on the prime attribute Machine_Number. Therefore, the Sports equipment relation is in second normal form.



  {Machine_Number}-> MachineName,State,BNum.

- Contracts:

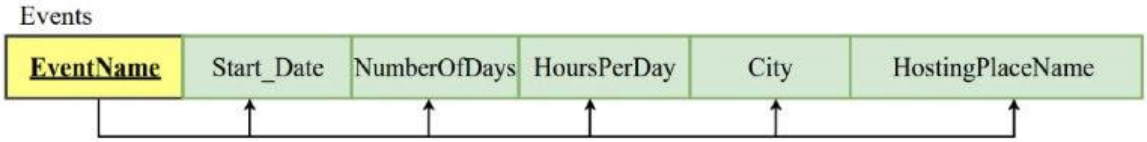
  The contracts relation has only one prime attribute that is ContractID. All the non-prime attributes in the relation are fully functionally dependent on the prime attribute ContractID. Therefore, the contracts relation is in second normal form.



  {ContractID}-> StartDate, EndDate,PaymentMethod, AppliedDiscounts, AppliedTaxes, DateOfPayment.

- Events:

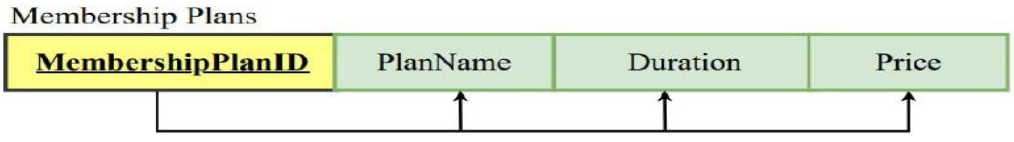
  The Events relation has only one prime attribute that is EventName. All the non-prime attributes in the relation are fully functionally dependent on the prime attribute EventName. Therefore, the Events relation is in second normal form.



  {EventName}-> Start_Date, NumberOfDays, HoursPerDay, City, HostingPlaceName.

- Membership plans:

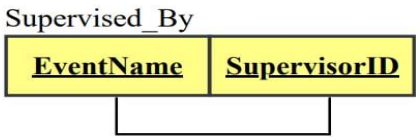
  The Membership Plans relation has only one prime attribute that is MembershipPlanID. All the non-prime attributes in the relation are fully functionally dependent on the prime attribute MembershipPlanID. Therefore, the Membership Plans relation is in second normal form.



{MembershipPlanID}-> PlanName, Duration,Price

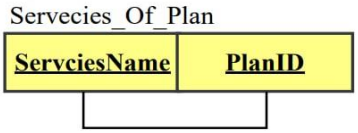
- Supervised_By:

  The Supervised_By relation has only two prime attributes that are EventName and SupervisorID and it contains no non-prime attributes. Therefore, the Supervied_By relation is in second normal form. If the Supervised_By relation had any non-prime attributes, then we will need to prove that the non-prime attributes are fully functional dependent on the prime attributes and if not, we must normalize them to the second normal form.



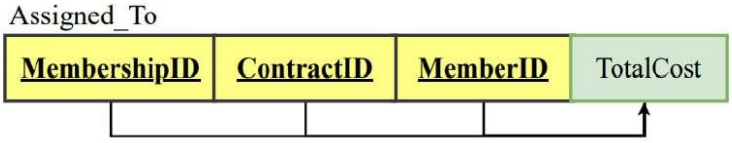
- Services_Of_Plan:

  The Services of plan relation has only two prime attributes that are ServicesName and PlanID and it contains no non-prime attributes. Therefore, the Services of plan relation is in second normal form. If the services of plan relation had any non-prime attributes, then we will need to prove that the non-prime attributes are fully functional dependent on the prime attributes and if not, we must normalize them to the second normal form.



- Assigned_To:

  The Assigned relation has three prime attributes that are MembershipID, ConractID, and MemberID, the only non-prime attribute in the relation is fully functionally dependent on all of the prime attributes MembershipID, ConractID, and MemberID. Therefore, the Assigned relation is in second normal form.

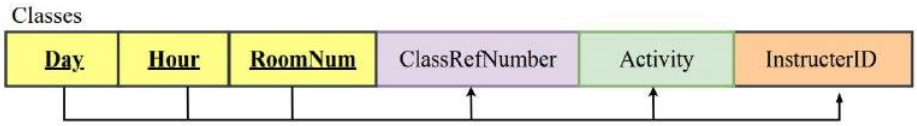


  The total cost of a contract for any member will be different depending on the membership plan that the member has been assigned to after applying to the gym, as well as the applied taxes and applied discounts to this contract. Showing that the total cost is fully functional dependent on all the three prime attributes.

  {MembershipID, ConractID, MemberID}-> TotalCost.

- Classes:

  The classes relation has three prime attributes that are Day,Hour and RoomNum. All the non-prime attributes in the relation are fully functionally dependent on the three prime attributes Day, Hour and RoomNum. Therefore, the classes relation is in second normal form.



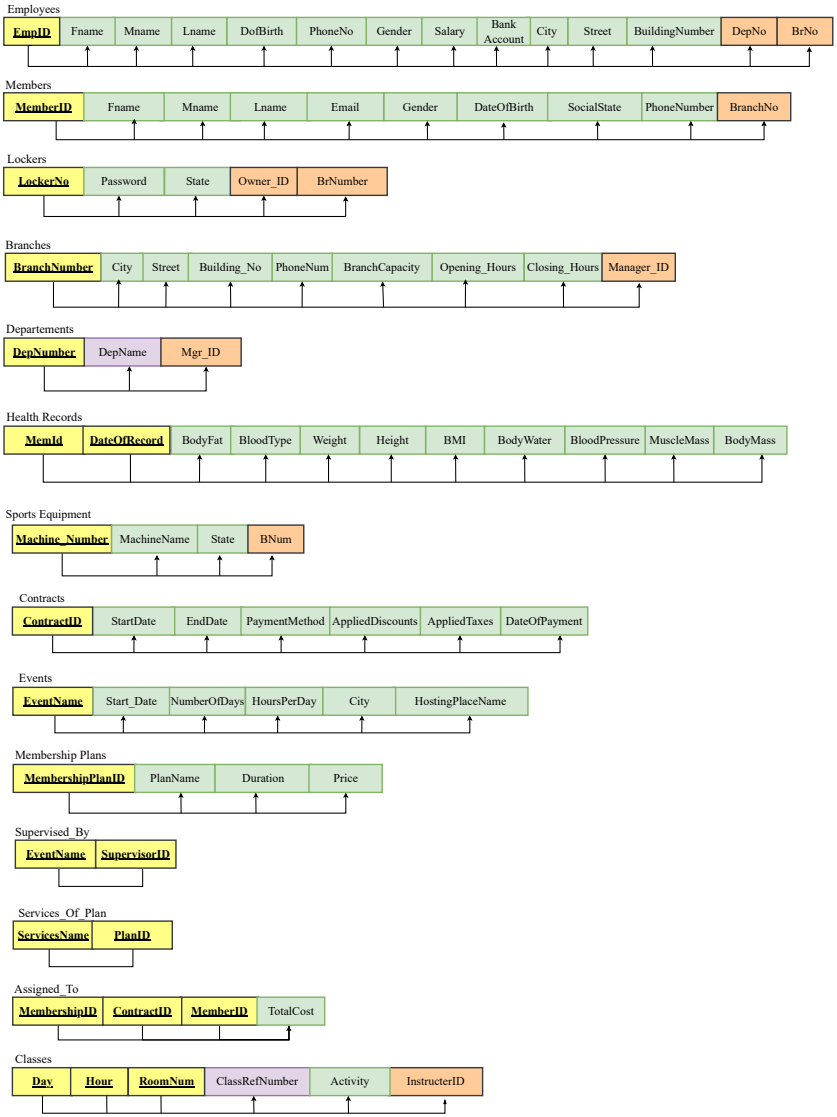
  To show that each instance of classes is different as well as to prevent overlapping, we chose to make the Day, Hour, RoomNum prime attributes for the classes relation.

  For example, the class that is held on Friday at 5 PM room number 6 has 1234 as its reference number (ClassRefNumber). This class is different than the class that is held on Friday at 6 PM room number 6 that has 2468 as its reference number (ClassRefNumber). This shows that the classes instances differ depending on the combination of these three attributes.

  {Day, Hour, RoomNum}-> ClassRefNumber, Activity, InstructerID.

  That is why each non-prime attribute will be fully functional dependent on this combination of prime attributes.

Final relation schema in 2NF:

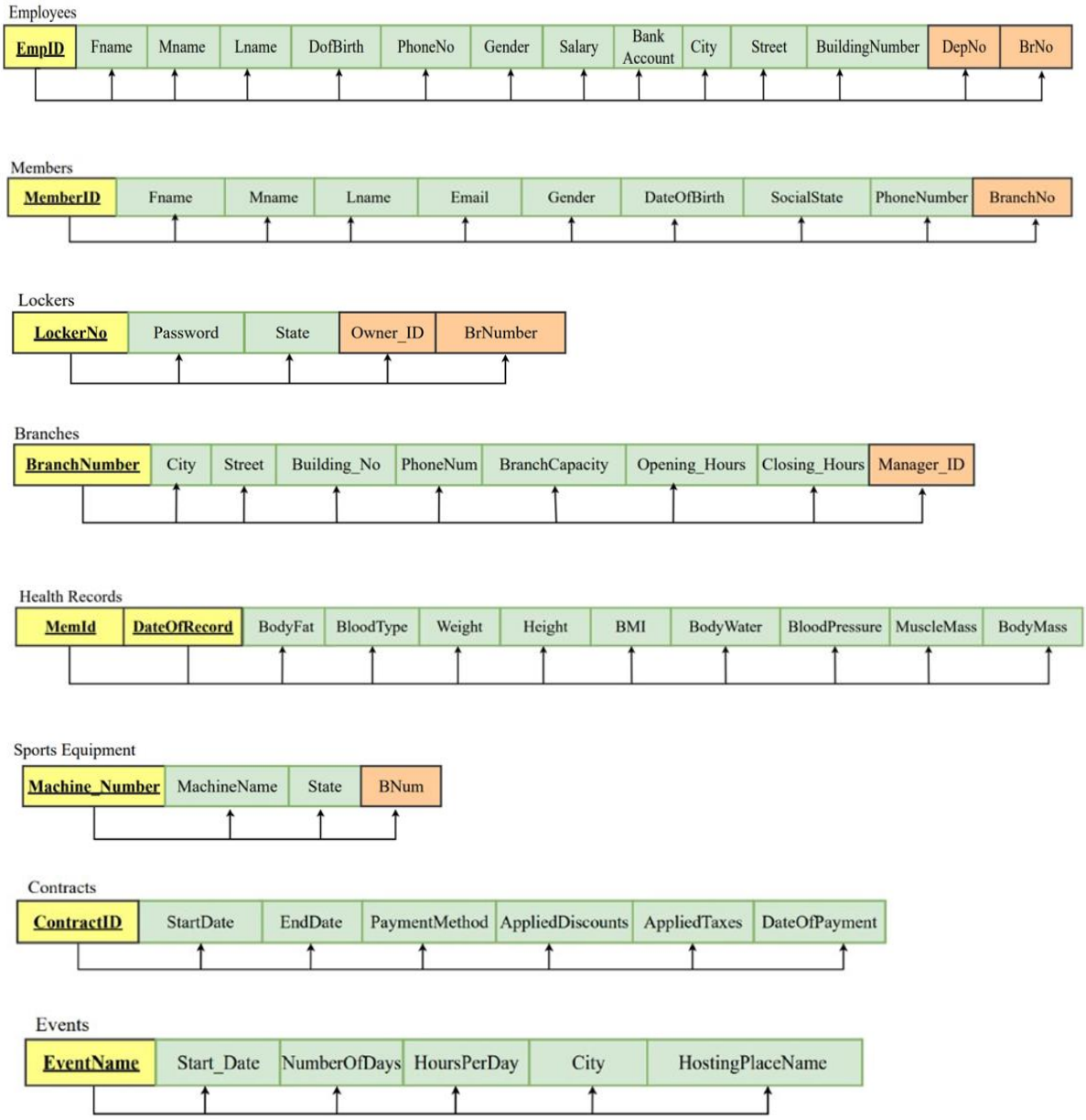

Employees
EmpID
Fname
Mname
Lname
DofBirth
PhoneNo
Gender
Salary
Bank Account
City
Street
BuildingNumber
DepNo
BrNo
Members
MemberID
Fname
Mname
Lname
Email
Gender
DateOfBirth
SocialState
PhoneNumber
BranchNo
Lockers
LockerNo
Password
State
Owner_ID
BrNumber
Branches
BranchNumber
City
Street
Building_No
PhoneNum
BranchCapacity
Opening_Hours
Closing_Hours
Manager_ID
Departements
DepNumber
DepName
Mgr_ID
Health Records
MemId
DateOfRecord
BodyFat
BloodType
Weight
Height
BMI
BodyWater
BloodPressure
MuscleMass
BodyMass
Sports Equipment
Machine_Number
MachineName
State
BNum
Contracts
ContractID
StartDate
EndDate
PaymentMethod
AppliedDiscounts
AppliedTaxes
DateOfPayment
Events
EventName
Start_Date
NumberOfDays
HoursPerDay
City
HostingPlaceName
Membership Plans
MembershipPlanID
PlanName
Duration
Price
Supervised_By
EventName
SupervisorID
Services_Of_Plan
ServicesName
PlanID
Assigned_To
MembershipID
ContractID
MemberID
TotalCost
Classes
Day
Hour
RoomNum
ClassRefNumber
Activity
InstructerID

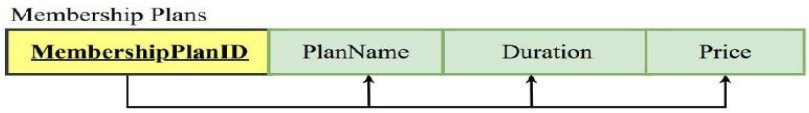
4.3 Third Normal Form

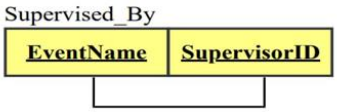
To normalize a relation in third normal form, we must remove all transitive dependencies on the primary key.

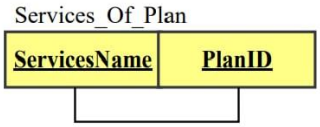
Transitive dependency is when a non-prime attribute is functionally dependent on another non-prime attribute. Third normal form allows transitive dependency only when the non-prime attribute the other non-prime attributes are functionally dependent on is a candidate key (unique).

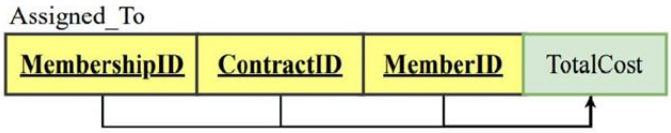
In section 4.2, we have normalized all relational schemas into the second normal form. Now we will list all of the relational schemas that are already in third normal form since they do not include any transitive dependency on the primary key.





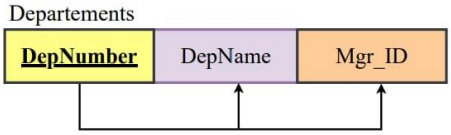






- Departments:

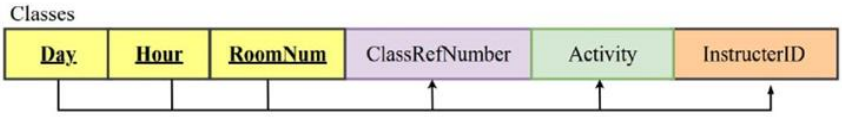
  In departments relation schema, there is a functional dependency on a non-prime attribute which is DepName to another non-prime attribute which is Mgr_ID. Even though it is considered to be a transitive depency which is disallowed in third normal form, it is however acceptable in this case because the non-prime attribute that the other non-prime attribute is functionally dependent on is a candidate key (unique attribute).



{DepNumber} -> DepName -> Mgr_ID

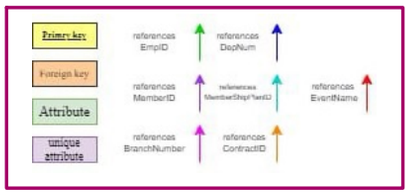
- Classes:

  In classes relation schema, there is a functional dependency on a non-prime attribute which is ClassRefNumber to other non-prime attributes which are Activity and InstructerID. Even though it is considered to be a transitive depency which is disallowed in third normal form, it is acceptable in this case because the non-prime attribute that the other non-prime attributes are functionally dependent on is a candidate key (unique attribute).



{Day,Hour,RoomNum] -> ClassRefNumber -> {Activity, InstructerID}

5 Final DB Schema Diagram



Employees

EmpID	Fname	Mname	Lname	DofBirth	PhoneNo	Gender	Salary	Bank Account	City	Street	BuildingNumber	DepNo	BrNo

Members

MemberID	Fname	Mname	Lname	Email	Gender	DateOfBirth	SocialState	PhoneNumber	BranchNo

Lockers

LockerNo	Password	State	Owner_ID	BrNumber

Branches

BranchNumber	City	Street	Building_No	PhoneNum	BranchCapacity	Opening_Hours	Closing_Hours	Manager_ID

Departements

DepNumber	DepName	Mgr_ID

Health Records

MemId	**DateOfRecord**	BodyFat	BloodType	Weight	Height	BMI	BodyWater	BloodPressure	MuscleMass	BodyMass

Sports Equipment

Machine_Number	MachineName	State	BNum

Contracts

ContractID	StartDate	EndDate	PaymentMethod	AppliedDiscounts	AppliedTaxes	DateOfPayment

Events

EventName	Start_Date	NumberOfDays	HoursPerDay	City	HostingPlaceName

Membership Plans

MembershipPlanID	PlanName	Duration	Price

Supervised_By

EventName	**SupervisorID**

Services_Of_Plan

ServicesName	**PlanID**

Assigned_To

MembershipID	**ContractID**	**MemberID**	TotalCost

Classes

Day	**Hour**	**RoomNum**	ClassRefNumber	Activity	InstructerID

PART III: IMPLEMENTATION

6 Table Creation Script

In the following section, we will start creating all the tables for the smash up Gym data base, there are 14 tables, containing all the information of the gym's database.

6.1 *Employees* TABLE

In the create table command of the employees we added all the attributes that form the table and the primary key. Since it is the first table to be inserted in the database, after adding and creating all the other tables of the system, we will then add the foreign key constraints using the alter table command. We will do the same for the rest of the tables.

```
create table EMPLOYEES (
    EmpID numeric(10) primary key ,
    Fname varchar(31) ,
    Mname varchar(31) ,
    Lname varchar(31) ,
    DofBirth date not null,
    PhoneNo numeric(10) ,
    Gender char(1) ,
    Salary float(5) ,
    BankAccount numeric(15),
    City varchar(31),
    Street varchar(31) ,
    BuildingNumber varchar(31) ,
    DepNo numeric(2),
    BrNo numeric(3),
    check (Gender in ('F','M')));

Alter table EMPLOYEES
add constraint FK_DepNo foreign key (DepNo) references DEPARTMENTS(DepNumber) on
delete CASCADE  ;

Alter table EMPLOYEES
add constraint FK_BrNo foreign key (BrNo) references BRANCHES(BranchNumber) on delete
CASCADE ;
```

```
create table EMPLOYEES (
    EmpID numeric(10) primary key ,
    Fname varchar(31) ,
    Mname varchar(31) ,
    Lname varchar(31) ,
    DofBirth date not null,
    PhoneNo numeric(10) ,
    Gender char(1) ,
    Salary float(5) ,
    BankAccount numeric(15),
    City varchar(31),
    Street varchar(31) ,
    BuildingNumber varchar(31) ,
    DepNo numeric(2),
    BrNo numeric(3),
    check (Gender in ('F','M')));
```

```
Alter table EMPLOYEES
add constraint FK_DepNo foreign key (DepNo) references DEPARTMENTS(DepNumber) on delete CASCADE  ;

Alter table EMPLOYEES
add constraint FK_BrNo foreign key (BrNo) references BRANCHES(BranchNumber) on delete CASCADE ;
```

6.2 *Members* TABLE

```
create table MEMBERS (
    MemberID numeric(10) primary key ,
    Fname varchar(31) ,
    Mname varchar(31) ,
    Lname varchar(31) ,
    Email varchar(50) ,
    Gender char(1) ,
    DateOfBirth date not null,
    SocialState varchar(20),
    PhoneNumber numeric(10),
    BranchNo numeric(3),
   check (Gender in ('F','M')),
   check (SocialState in ('Single','Married')));

Alter table MEMBERS
add constraint FK_BranchNo foreign key (BranchNo) references BRANCHES(BranchNumber)on
delete CASCADE;

Alter table Members
add constraint CheckAge check ( DATEDIFF(Year , DateOfBirth,CURRENT_TIMESTAMP) >= 16)
```

```
create table MEMBERS (
    MemberID numeric(10) primary key ,
    Fname varchar(31) ,
    Mname varchar(31) ,
    Lname varchar(31) ,
    Email varchar(50) ,
    Gender char(1) ,
    DateOfBirth date not null,
    SocialState varchar(20),
    PhoneNumber numeric(10),
    BranchNo numeric(3),
   check (Gender in ('F','M')),
   check (SocialState in ('Single','Married')));
```

```
Alter table MEMBERS
add constraint FK_BranchNo foreign key (BranchNo) references BRANCHES(BranchNumber)on delete CASCADE;
```

```
Alter table Members
add constraint CheckAge check ( DATEDIFF(Year , DateOfBirth,CURRENT_TIMESTAMP) >= 16)
```

6.3 *Lockers* TABLE

```
create table LOCKERS (
LockerNo numeric(10) primary key,
Passwordd numeric(6) default 124567 ,
Statee varchar(20) not null default 'Available',
Owner_ID numeric(10) default null ,
BrNumber numeric(3) ,
check (Statee in('Occupied','Available'))
);

Alter table LOCKERS
add constraint FK_OwnerID foreign key (Owner_ID) references MEMBERS(MemberID) on
delete set default ;

Alter table LOCKERS
add constraint FK_BrNumber foreign key (BrNumber) references BRANCHES(BranchNumber);
```

```
create table LOCKERS (
LockerNo numeric(10) primary key,
Passwordd numeric(6) default 124567 ,
Statee varchar(20) not null default 'Available',
Owner_ID numeric(10) default null ,
BrNumber numeric(3) ,
check (Statee in('Occupied','Available'))
);
```

```
Alter table LOCKERS
add constraint FK_OwnerID foreign key (Owner_ID) references MEMBERS(MemberID) on delete set default ;

Alter table LOCKERS
add constraint FK_BrNumber foreign key (BrNumber) references BRANCHES(BranchNumber);
```

6.4 *Branches* TABLE

```
create table BRANCHES (
BranchNumber numeric(3) primary key,
City varchar(31),
Street varchar(20),
Building_No varchar(4),
PhoneNum numeric(10),
BranchCpacity numeric(3) not null,
Opening_Hours time(0)  not null,
Closing_Hours time(0) not null,
Manager_ID numeric(10));

Alter table BRANCHES
add constraint FK_ManagerID foreign key (Manager_ID) references EMPLOYEES(EmpID);
```

```
create table BRANCHES (
BranchNumber numeric(3) primary key,
City varchar(31),
Street varchar(20),
Building_No varchar(4),
PhoneNum numeric(10),
BranchCpacity numeric(3) not null,
Opening_Hours time(0)  not null,
Closing_Hours time(0) not null,
Manager_ID numeric(10));
```

```
Alter table BRANCHES
add constraint FK_ManagerID foreign key (Manager_ID) references EMPLOYEES(EmpID);
```

6.5 *Departments* TABLE

```
create table DEPARTMENTS (
DepNumber numeric(2) primary key,
DepName varchar(20),
Mgr_ID numeric(10),
Unique(DepName));

Alter table DEPARTMENTS
add constraint FK_MgrID foreign key (Mgr_ID) references EMPLOYEES(EmpID);
```

```
create table DEPARTMENTS (
DepNumber numeric(2) primary key,
DepName varchar(20),
Mgr_ID numeric(10),
Unique(DepName));
```

```
Alter table DEPARTMENTS
add constraint FK_MgrID foreign key (Mgr_ID) references EMPLOYEES(EmpID);
```

6.6 *Health records* TABLE

```
create table HealthRecords(
MemID numeric(10),
DateOfRecord date Not null,
BodyFat numeric(5,2),
BloodType char(3),
Weightt numeric(5,2),
Height numeric(5,2),
BMI numeric(5,2),
BodyWater numeric(5,2),
BloodPressure numeric(5,2),
MuscleMass numeric(5,2),
BodyMass numeric(5,2),
primary key(MemID,DateOfRecord), );


Alter table HEALTHRECORDS
add constraint FK_MemID foreign key (MemId) references MEMBERS(MemberID) on delete
CASCADE;
```

```
create table HealthRecords(
MemID numeric(10),
DateOfRecord date Not null,
BodyFat numeric(5,2),
BloodType char(3),
Weightt numeric(5,2),
Height numeric(5,2),
BMI numeric(5,2),
BodyWater numeric(5,2),
BloodPressure numeric(5,2),
MuscleMass numeric(5,2),
BodyMass numeric(5,2),
primary key(MemID,DateOfRecord), );
```

```
Alter table HEALTHRECORDS
add constraint FK_MemID foreign key (MemId) references MEMBERS(MemberID) on delete cascade ;
```

6.7 *Sports equipment* TABLE

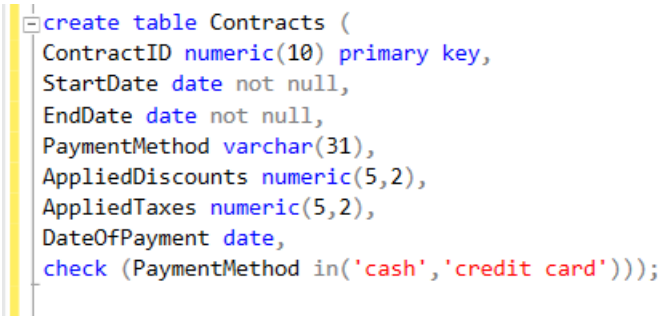
```
create table SportsEquipment (
Machine_Number numeric(10) primary key,
MachineName varchar(31) ,
State varchar(31),
Bnum numeric(3) default 0,
check (State in('In Service','Out of service')));


Alter table SportsEquipment
add constraint FK_BNum foreign key (BNum) references BRANCHES(BranchNumber) on delete
CASCADE;
```

```
create table SportsEquipment (
Machine_Number numeric(10) primary key,
MachineName varchar(31) ,
State varchar(31),
Bnum numeric(3) default 0,
check (State in('In Service','Out of service')));
```

```
Alter table SportsEquipment
add constraint FK_BNum foreign key (BNum) references BRANCHES(BranchNumber) on delete CASCADE;
```

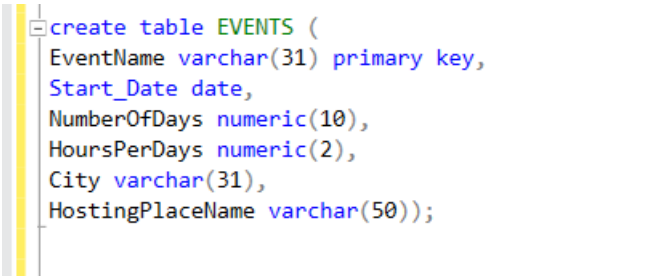
6.8 *Contracts* TABLE

```
create table Contracts (
ContractID numeric(10) primary key,
StartDate date not null,
EndDate date not null,
PaymentMethod varchar(31),
AppliedDiscounts numeric(5,2),
AppliedTaxes numeric(5,2),
DateOfPayment date,
check (PaymentMethod in('cash','credit card')));
```



6.9 *Events* TABLE

```
create table EVENTS (
EventName varchar(31) primary key,
Start_Date date,
NumberOfDays numeric(10),
HoursPerDays numeric(2),
City varchar(31),
HostingPlaceName varchar(50));
```



6.10 *Membership plans* TABLE

```
create table MembershipPlans (
MembershipPlanID numeric(10) primary key,
PlanName varchar(31),
Durationn varchar(31),
Price numeric(5),
check (PlanName in('Diamond','Golden','Platinum')),
check (Durationn in('3 months','6 months','12 months')));
```

```
create table MembershipPlans (
MembershipPlanID numeric(10) primary key,
PlanName varchar(31),
Durationn varchar(31),
Price numeric(5),
check (PlanName in('Diamond','Golden','Platinum')),
check (Durationn in('3 months','6 months','12 months')));
```

6.11 *Supervised_By* TABLE

```
create table Supervised_By(
EventName varchar(31) not null,
supervisorID numeric(10) ,
primary key(EventName , supervisorID));


Alter table Supervised_By
add constraint FK_EventName foreign key (EventName) references Events(EventName)on
delete CASCADE;

Alter table Supervised_By
add constraint FK_supervisorID foreign key (supervisorID) references
Employees(EmpID)on delete CASCADE;
```

```
create table Supervised_By(
EventName varchar(31) not null,
supervisorID numeric(10) ,
primary key(EventName , supervisorID));
```

```
Alter table Supervised_By
add constraint FK_EventName foreign key (EventName) references Events(EventName)on delete CASCADE;

Alter table Supervised_By
add constraint FK_supervisorID foreign key (supervisorID) references Employees(EmpID)on delete CASCADE;
```

6.12 *Services_Of_Plan* TABLE

```
create table Services_Of_Plan(
ServicesName varchar(31),
PlanID numeric(10),
primary key(ServicesName , PlanID));


Alter table Services_Of_Plan
add constraint FK_PlanID foreign key (PlanID) references
MembershipPlans(MembershipPlanID) on delete CASCADE;
```

```
create table Services_Of_Plan(
ServicesName varchar(31),
PlanID numeric(10),
primary key(ServicesName , PlanID));
```

```
Alter table Services_Of_Plan
add constraint FK_PlanID foreign key (PlanID) references MembershipPlans(MembershipPlanID) on delete CASCADE;
```

6.13 *Assigned_To* TABLE

```
create table Assigned_To(
MembershipID numeric(10),
ContractID numeric(10),
MemberID numeric(10),
TotalCost float(5),
primary key(MembershipID , ContractID , MemberID ));


Alter table Assigned_To
add constraint FK_MembershipID foreign key (MembershipID) references
MembershipPlans(MembershipPlanID) on delete CASCADE;

Alter table Assigned_To
add constraint FK_ContractID foreign key (ContractID) references Contracts(ContractID)
on delete CASCADE;

Alter table Assigned_To
add constraint FK_MemberID foreign key (MemberID) references Members(MemberID)on
delete CASCADE;
```

```
create table Assigned_To(
MembershipID numeric(10),
ContractID numeric(10),
MemberID numeric(10),
TotalCost float(5),
primary key(MembershipID , ContractID , MemberID ));
```

```
Alter table Assigned_To
add constraint FK_MembershipID foreign key (MembershipID) references MembershipPlans(MembershipPlanID) on delete CASCADE;

Alter table Assigned_To
add constraint FK_ContractID foreign key (ContractID) references Contracts(ContractID) on delete CASCADE;

Alter table Assigned_To
add constraint FK_MemberID foreign key (MemberID) references Members(MemberID)on delete CASCADE;
```

6.14 *Classes* TABLE

```
create table Classes(
Dayy varchar(31),
Hourr  time(0),
RoomNum  numeric(5),
ClassRefNumber numeric(10) unique,
Activity varchar(31),
InstructerID numeric(10),
primary key(Dayy , Hourr , RoomNum ));

Alter table Classes
add constraint FK_InstructerID foreign key (InstructerID) references EMPLOYEES(EmpID)
on delete CASCADE;
```

```
create table Classes(
Dayy varchar(31),
Hourr  time(0),
RoomNum  numeric(5),
ClassRefNumber numeric(10) unique,
Activity varchar(31),
InstructerID numeric(10),
primary key(Dayy , Hourr , RoomNum ));
```

```
Alter table Classes
add constraint FK_InstructerID foreign key (InstructerID) references EMPLOYEES(EmpID) on delete CASCADE;
```

7 Constraints Script

<table>
<tr><th>Business Rule</th><th>SQL Script</th><th>Table</th></tr>
<tr><td>A member must be at least 16 years old to apply to the GYM.</td><td>

```
Alter table Members
add constraint CheckAge check ( DATEDIFF(Year ,
DateOfBirth,CURRENT_TIMESTAMP) >= 16)
```

```
insert into Members(MemberID ,Fname , Mname , Lname , Email,
Gender, DateOfBirth , SocialState, PhoneNumber , BranchNo )
values(1000000033 , 'Amal' , 'Ahmed' , 'Saeed' ,
'Ameal09@gmail.com' ,'F' , '2008-05-17' ,
'Single' , 0554144781 , 01);
```

.00 %

Messages

```
Msg 547, Level 16, State 0, Line 21
The INSERT statement conflicted with the CHECK constraint "CheckAge".
The statement has been terminated.
```

</td><td>Members</td></tr>
<tr><td>Overall working hours of a branch should be at least 12 hours.</td><td>

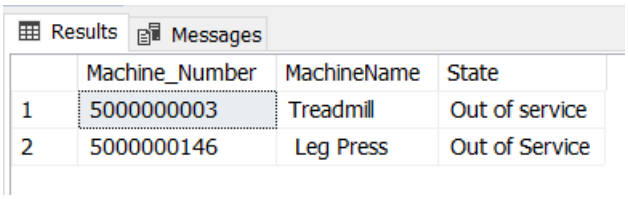
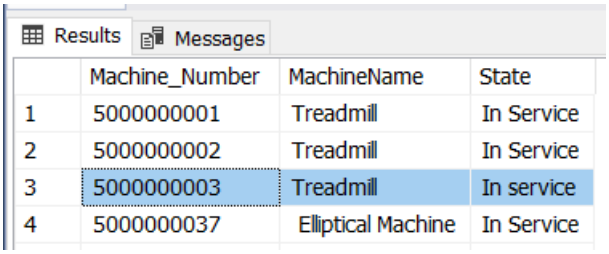
```
select DATEDIFF(HOUR, Opening_Hours
,Closing_Hours) as WorkingHour, BranchNumber from
BRANCHES ;
```

Results Messages

	WorkingHour	BranchNumber
1	12	1
2	12	2
3	15	3
4	12	4
5	12	5
6	12	6
7	16	7
8	14	8
9	14	9
10	15	10

</td><td>Branches</td></tr>
</table>

To prevent overlapping, no two classes can occur at the same time, at the same room. So, we made the primary key a composite key made up of the combination of day, hour, and room number as heighted in the SQL script.	`create table Classes (` `Dayy varchar(31),` `Hourr  time(0),` `RoomNum  numeric(5),` `ClassRefNumber numeric(10) unique,` `Activity varchar(31),` `InstructerID numeric(10),` `primary key(Dayy , Hourr , RoomNum ));`	Classes Table

The device's state and location must be recorded, in case of an equipment that is out of service, the maintenance administrator responsible for a specific Branch will manually request to fix the devices.	``` create table SportsEquipment (…… …… …… State varchar(31), Bnum numeric(3) default 0, check (State in('In Service','Out of service'))); Select Machine_Number , MachineName ,State From SportsEquipment , BRANCHES Where BranchNumber=Bnum and Bnum =1 and state='Out of service' ```  Results Messages Machine_Number MachineName State 1 5000000003 Treadmill Out of service 2 5000000146 Leg Press Out of Service In case the device was fixed we must update its state to (In service). ``` update SportsEquipment set State='In service' where Bnum=1 and Machine_Number= 5000000003 ```  Results Messages Machine_Number MachineName State 1 5000000001 Treadmill In Service 2 5000000002 Treadmill In Service 3 5000000003 Treadmill In service 4 5000000037 Elliptical Machine In Service	Sports Equipment

Each branch must have several sports equipment, and an equipment must be located at a specific branch. To show this relation between branches and equipments, we added a foreign key that references the branches' number in the sports equipment table.	`Alter table SportsEquipment` `add constraint FK_BNum foreign key (BNum)` `references BRANCHES(BranchNumber);`	Sports Equipment

<table>
<tr>
<td>Each employee must
Belong to a department, and each department must have at least one employee belonging to it. Number of employees in each department must be counted.</td>
<td>

```
Alter table EMPLOYEES
add constraint FK_DepNo foreign key (DepNo)
references DEPARTMENTS(DepNumber);

select  DepNumber ,DepName ,count(*) as
NumberOfEmployee
From EMPLOYEES , DEPARTMENTS
where DepNumber = DepNo
group by DepName , DepNumber
order by DepNumber;
```

DepNumber	DepName	NumberOfEmployee
1	Maintenance	10
2	Customer service	11
3	Management	10
4	Medical care	12
5	Coaching staff	31

</td>
<td>Employees</td>
</tr>
<tr>
<td>Each employee must work in at most one branch, and each branch must have at least one employee. The number of current working employees in a branch must be counted.</td>
<td>

```
Alter table EMPLOYEES
add constraint FK_BrNo foreign key (BrNo)
references BRANCHES(BranchNumber);

select  BranchNumber  ,count(*) as
NumberOfEmployee
From EMPLOYEES , BRANCHES
where BranchNumber = BrNo
group by BranchNumber
order by BranchNumber;
```

BranchNumber	NumberOfEmployee
1	7
2	9
3	8
4	7
5	7
6	7
7	8
8	7
9	7
10	7

</td>
<td>Employees</td>
</tr>
</table>

<table>
<tr>
<td>A branch must have several registered members, each member must go to one branch. The number of members applied in the branch should not exceed the branch's capacity.</td>
<td>

```
Alter table MEMBERS
add constraint FK_BranchNo foreign key (BranchNo)
references BRANCHES(BranchNumber);
```

```
select
BranchNumber,branchCpacity,count(MemberID) as
NumOfMembers, (branchCpacity-count(MemberID))
as RemaindorOfCapacity
from Branches , members
where BranchNumber = BranchNo
group by BranchNumber,branchCpacity
```

BranchNumber	branchCpacity	NumOfMembers	RemaindorOfCapacity
1	150	3	147
2	300	3	297
3	100	3	97
4	250	3	247
5	150	3	147
6	200	3	197
7	125	3	122
8	150	3	147
9	120	3	117
10	150	3	147

</td>
<td>Members</td>
</tr>
<tr>
<td>A branch must be managed by only one employee. To show this relation between branches and employees, we added a foreign key that references the employees' ID in the branches table.</td>
<td>

```
Alter table BRANCHES
add constraint FK_ManagerID foreign key
(Manager_ID) references EMPLOYEES(EmpID);
```

</td>
<td>Branches</td>
</tr>
</table>

A department must be managed by only one employee. To show this relation between departments and employees, we added a foreign key that references the employees' ID in the departments table.	`Alter table DEPARTMENTS` `add constraint FK_MgrID foreign key (Mgr_ID)` `references EMPLOYEES(EmpID);`	Departments
A class must be instructed by a single instructor, and an instructor may instruct several classes. To show this relation between classes and employees, we added a foreign key that references the employees' ID in the classes table.	`Alter table Classes` `add constraint FK_InstructerID foreign key` `(InstructerID) references EMPLOYEES(EmpID);`	Classes

A locker can be owned by a single member. And a member can own only one locker. To show this relation between lockers and members, we added a foreign key that references the member' ID in the lockers table.	`Alter table LOCKERS` `add constraint FK_OwnerID foreign key (Owner_ID)` `references MEMBERS(MemberID);`	Lockers
Each branch includes several lockers. Each locker must be found in a specific branch. To show this relation between lockers and branches, we added a foreign key that references the branch's number in the lockers table.	`Alter table LOCKERS` `add constraint FK_BrNumber foreign key (BrNumber)` `references BRANCHES(BranchNumber);` `select BranchNumber , count(*) as NumberOfLockers` `From LOCKERS , BRANCHES` `where BranchNumber = BrNumber` `group by BranchNumber` `order by BranchNumber` BranchNumber / NumberOfLockers 1 / 10 2 / 15 3 / 9 4 / 15 5 / 10 6 / 15 7 / 11 8 / 11 9 / 10 10 / 10	Lockers

An event must be supervised by at least one supervisor, and a supervisor may supervise several events.	create table Supervised_By(EventName varchar(31) not null, supervisorID numeric(10) , primary key(EventName , supervisorID)); Alter table Supervised_By add constraint FK_EventName foreign key (EventName) references Events(EventName); Alter table Supervised_By add constraint FK_supervisorID foreign key (supervisorID) references Employees(EmpID);	Supervised_By
Each member must have at least one or more health records, each record with different record dates. To show this relation between health records and members, we added a foreign key that references the member' ID in the health record's table.	create table HealthRecords(MemID numeric(10), DateOfRecord date Not null, …… …… …… primary key(MemID,DateOfRecord),); Alter table HEALTHRECORDS add constraint FK_MemID foreign key (MemId) references MEMBERS(MemberID);	Health Records

<table>
<tr>
<td>If a month has passed since the last measurements were taken, the health records should be measured again.</td>
<td>

```
Select MemberID , DATEDIFF(MONTH ,
max(DateOfRecord), CURRENT_TIMESTAMP)as
LastRecord
from MEMBERS , HealthRecords
where MemberID = MemID
group by MemberID
order by LastRecord  ;
```

MemberID	LastRecord
1000000012	0
1000000014	0
1000000015	0
1000000017	0
1000000019	0
1000000018	1
1000000016	1
1000000013	1
1000000008	1
1000000009	1
1000000010	1
1000000011	1

</td>
<td>Health Records</td>
</tr>
<tr>
<td>Each member must be assigned to one membership plan that is specified by the contract. The total cost for the membership plan including the applied taxes and discounts that the member has paid for will be stored.</td>
<td>

```
create table Assigned_To(
MembershipID numeric(10),
ContractID numeric(10),
MemberID numeric(10),
TotalCost float(5),
primary key(MembershipID , ContractID , MemberID
));

Alter table Assigned_To
add constraint FK_MembershipID foreign key
(MembershipID) references
MembershipPlans(MembershipPlanID);

Alter table Assigned_To
add constraint FK_ContractID foreign key
(ContractID) references Contracts(ContractID);

Alter table Assigned_To
add constraint FK_MemberID foreign key (MemberID)
references Members(MemberID);
```

</td>
<td>Assighned_To</td>
</tr>
</table>

8 Queries and Transactions

8.1 *<Finding the minimum salary of employees>*

Query in Natural Language:

To evaluate and manage the gym's financial situation, we will use this query to help find the employees that get the least amount of salary out of all the employees of the gym, this query will display their IDs, full names, the department and the branch that they belong to and their salary amount.

SQL Script

```
select EmpID , Fname, Mname ,Lname , DepName,BrNo , Salary
from EMPLOYEES , DEPARTMENTS, Branches
where DepNo=DepNumber and BrNo=BranchNumber
and Salary =(select min(Salary)
             from EMPLOYEES)
```

Caption of the output:

Results Messages

	EmpID	Fname	Mname	Lname	DepName	BrNo	Salary
1	2022000006	Arwa	Maher	Abukhammas	Management	1	2000
2	2022000044	Ebtihag	Atiah	Hakami	Coaching staff	5	2000

8.2 <*Invoices* >

Query in Natural Language:

In section 1.4 we have said that one of our intended outputs of this system is generating an invoice once a member joins the gym and signs a contract deal for which membership plan they want to register, this generated invoice will display:

- Contract ID
- Member ID
- Plan's name
- Branch No
- Start date (Day, Month, Year)
- End date (Day, Month, Year)
- Date of payment (Day, Month, Year)
- Payment method (Cash/Credit card)
- Membership plan price
- Applied discounts
- Applied taxes
- Total cost

SQL Script:

```
select BranchNo, Fname ,Lname ,Contracts.ContractID ,
               Assigned_To.MemberID , PlanName,  StartDate , EndDate ,
               DateOfPayment ,PaymentMethod , Price , AppliedDiscounts ,
               AppliedTaxes , TotalCost
from   Contracts  , MEMBERS  , Assigned_To  , MembershipPlans
where  Contracts.ContractID = Assigned_To.ContractID
               and Assigned_To.MemberID= MEMBERS.MemberID
               and MembershipPlanID = MembershipID
               and DateOfPayment between '2022-05-15' and '2023-01-01'
order by BranchNo
```

Caption of the Output:

Results | Messages

	BranchNo	Fname	Lname	ContractID	MemberID	PlanName	StartDate	EndDate	DateOfPayment	PaymentMethod	Price	AppliedDiscounts	AppliedTaxes	TotalCost
1	1	Seham	Nahlawi	4000000021	1000000021	Diamond	2022-07-05	2022-10-05	2022-07-09	cash	2000	0.05	0.15	2185
2	2	Shahira	Nahlawi	4000000022	1000000022	Diamond	2022-06-22	2022-12-22	2022-06-22	credit card	2200	0.10	0.15	2277
3	3	Najah	Nahlawi	4000000023	1000000023	Golden	2022-09-28	2023-03-28	2022-09-29	cash	1600	0.02	0.15	1803.2
4	4	Sawsan	Nahlawi	4000000024	1000000024	Platinum	2022-12-01	2023-06-01	2022-12-02	credit card	1000	0.10	0.15	1035
5	5	Huda	Nahlawi	4000000025	1000000025	Platinum	2022-11-20	2023-11-20	2022-11-21	cash	1200	0.10	0.15	1242
6	6	Razan	Nahlawi	4000000026	1000000026	Golden	2022-07-08	2023-07-08	2022-07-09	credit card	1800	0.20	0.15	1656
7	7	Rawan	Nahlawi	4000000027	1000000027	Golden	2022-09-01	2023-09-01	2022-09-03	credit card	1800	0.15	0.15	1759.5
8	8	Remma	Nahlawi	4000000028	1000000028	Golden	2022-09-01	2022-12-01	2022-09-04	credit card	1400	NULL	0.15	1380
9	9	Hala	Nahlawi	4000000029	1000000029	Platinum	2022-07-01	2023-01-01	2022-07-01	credit card	1000	0.10	0.15	1035
10	10	Lena	Nahlawi	4000000030	1000000030	Diamond	2022-07-12	2023-07-12	2022-07-10	cash	2400	0.45	0.15	1518
11	10	Hawaa	Qari	4000000020	1000000020	Platinum	2022-06-23	2022-09-23	2022-06-23	credit card	800	0.10	0.15	828

Query executed successfully. DESKTOP-K1LBA0F\SQLEXPRESS ... DESKTOP-

8.3 <*Calculating the total profits of membership plans*>

Query in Natural Language:

To calculate the profits of every membership plan offered by our gym, we will use a query to help us count the total profits of each plan and how many members in our gym are registered in that plan. We will also benefit from this query to show which plan is most popular and profitable.

SQL Script:

```
select  PlanName,Durationn ,sum(TotalCost)as TotalProfits, count(*) as NumOfMembers
from Assigned_To,MembershipPlans
where MembershipID=MembershipPlanID
group by PlanName,Durationn
```

Caption of the First Five Rows of the Output:

Results | Messages

	PlanName	Durationn	TotalProfits	NumOfMembers
1	Diamond	12 months	6348	3
2	Golden	12 months	3415.5	2
3	Platinum	12 months	3864	3
4	Diamond	3 months	6210	3
5	Golden	3 months	4209	3
6	Platinum	3 months	3404	4
7	Diamond	6 months	7084	3
8	Golden	6 months	4931.19995117188	3
9	Platinum	6 months	4140	4

8.4 *<Events schedule in a specific city>*

Query in Natural Language:

In order to see our plan of events in Jeddah for the year 2022 , we used a query to help and filter out only these events who were going to be hosted in Jeddah for 2022 , this query will not only display these events, but it will also display all of its information , like the place where it will be hosted and the supervisors IDs which will supervise these events, the duration of each event and so on. The schedule will be ordered in terms of the start dates of each event.

SQL Script:

```
select events.*,supervisorID
from events , Supervised_By
where events.EventName=Supervised_By.EventName and city IN('jeddah') and Start_Date
LIKE'2022-%%-%%'
order by Start_Date
```

Caption of the Output:

Results Messages

	EventName	Start_Date	NumberOfDays	HoursPerDays	City	HostingPlaceName	supervisorID
1	Horse Riding	2022-05-18	3	5	Jeddah	Alreem.Stabel	2022000003
2	Horse Riding	2022-05-18	3	5	Jeddah	Alreem.Stabel	2022000032
3	Golf Match	2022-07-12	1	3	Jeddah	Groovy Golf	2022000028
4	Roller Skating	2022-08-15	3	2	Jeddah	Ice Land	2022000027
5	Running Marathon	2022-08-20	3	4	Jeddah	Jeddah Waterfront	2022000009
6	Running Marathon	2022-08-20	3	4	Jeddah	Jeddah Waterfront	2022000025
7	Tennis Match	2022-10-03	1	3	Jeddah	Spin&Smach	2022000066
8	Tennis Match	2022-10-03	1	3	Jeddah	Spin&Smach	2022000070
9	Biking Marathon	2022-11-15	3	4	Jeddah	Jeddah Waterfront	2022000034
10	Biking Marathon	2022-11-15	3	4	Jeddah	Jeddah Waterfront	2022000047
11	Football Match	2022-12-12	1	3	Jeddah	Jawharat Alajaweed Stadium	2022000035
12	Football Match	2022-12-12	1	3	Jeddah	Jawharat Alajaweed Stadium	2022000050

Query executed successfully.

8.5 <*Potential Qualified Blood Volunteers in each city*>

Query in Natural Language:

The National Blood Bank Center released an announcement for an open opportunity for donating with a few sets of conditions that the donators must fulfil, first the donator should be more than 20 years old with a weight range no less than 50 Kg. Since we want to urge our members to donate blood as it is very important, we used a query to help us calculate how many members belong to each specific blood type in each specific city. This query will filter out all the non-qualified members first and then calculate all the qualified members for each blood type.

SQL Script:

The query used for all the branches in Jeddah:

```
select BloodType ,  count(  Distinct MemberID) as NumOfmember
from HealthRecords , MEMBERS
where MemberID= MemID  and  memID = any (
                                        Select  MemberID
                                        from MEMBERS , BRANCHES
                                        where BranchNo=BranchNumber and DATEDIFF(Year
, DateOfBirth,CURRENT_TIMESTAMP) >20  and Weightt >=50 and  City='Jeddah'  )
group by  BloodType
```

The query used for all of the branches in Riyadh:

```
select BloodType ,  count(  Distinct MemberID) as NumOfmember
from HealthRecords , MEMBERS
where MemberID= MemID  and  memID = any (
                                        Select  MemberID
                                        from MEMBERS , BRANCHES
                                        where BranchNo=BranchNumber and DATEDIFF(Year
, DateOfBirth,CURRENT_TIMESTAMP) >20  and Weightt >=50 and  City='Riyadh'  )
group by  BloodType
```

Caption of the Output:

Jeddah's qualified members statistics:

Results | Messages

	BloodType	NumOfmember
1	A-	2
2	A+	2
3	AB-	1
4	AB+	1
5	B-	1
6	B+	2
7	O+	3

Riyadh's qualified members statistics:

Results | Messages

	BloodType	NumOfmember
1	A+	1
2	AB-	1
3	AB+	1
4	O-	1
5	O+	1

8.6 <*Weekly Schedule for a specific activity*>

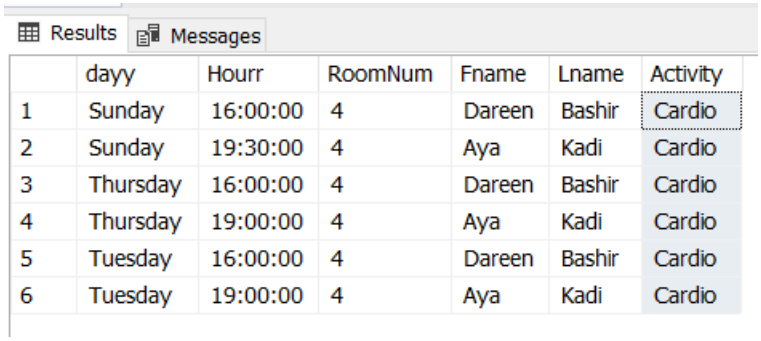
Query in Natural Language:

A new member has joined branch number 1, so they asked for the weekly schedule of the Cardio classes hosted in the branch. We used a query to display the weekly schedule of the cardio classes, the query will display the time, day, and room number of where the cardio class will be held, and it will also display the name of the coach that will be instructing this class.

SQL Script:

```
select dayy, Hourr, RoomNum, Fname, Lname,  Activity
from ((Classes join EMPLOYEES on InstructerID =EmpID )  join BRANCHES on brNo=
branchNumber)
where BrNo=1  and  Activity= 'Cardio'
```

Caption of the Output:



Results | Messages

	dayy	Hourr	RoomNum	Fname	Lname	Activity
1	Sunday	16:00:00	4	Dareen	Bashir	Cardio
2	Sunday	19:30:00	4	Aya	Kadi	Cardio
3	Thursday	16:00:00	4	Dareen	Bashir	Cardio
4	Thursday	19:00:00	4	Aya	Kadi	Cardio
5	Tuesday	16:00:00	4	Dareen	Bashir	Cardio
6	Tuesday	19:00:00	4	Aya	Kadi	Cardio

8.7 <Raising the salary for some employees>

Update in Natural Language:

Giving a raise to the coaching staff department whose employees' salaries are somewhere between 2000 and 3000 SR.

SQL Script:

```
update employees

set salary *=1.3
where depNo=5 and salary =  any (select salary
                                 from employees where salary between 2000 and 3000)
```

Caption of the Output:

Before update:

Results | Messages

	EmpID	Fname	Lname	DepName	salary
1	2022000009	Alhanuf	Algulsai	Coaching staff	3000
2	2022000020	Aya	Kadi	Coaching staff	4400
3	2022000021	Dana	Alsuqaie	Coaching staff	4600
4	2022000022	Dareen	Bashir	Coaching staff	4000
5	2022000026	Layana	Kattouha	Coaching staff	4000
6	2022000027	Laila	Alsaqri	Coaching staff	4200
7	2022000028	Fatima	Najjar	Coaching staff	3800
8	2022000032	Seham	Nahdi	Coaching staff	6000
9	2022000033	Shaden	Nahdi	Coaching staff	8500
10	2022000034	Najwah	Baward	Coaching staff	2500
11	2022000038	Eman	Alwael	Coaching staff	3000
12	2022000039	Amal	Alamoudi	Coaching staff	4500
13	2022000040	Razan	Salleh	Coaching staff	2500
14	2022000043	Asrar	Alali	Coaching staff	2500
15	2022000044	Ebtihag	Hakami	Coaching staff	2000
16	2022000045	Abrar	Faqiha	Coaching staff	4000
17	2022000048	Basm...	Kamoni	Coaching staff	4000
18	2022000049	Walaa	Alnahdi	Coaching staff	3000
19	2022000050	Toqa	Banabilah	Coaching staff	3500
20	2022000054	Asia	Alshehri	Coaching staff	4500
21	2022000055	Malk	Alharbi	Coaching staff	4500
22	2022000056	Afnan	Alahmadi	Coaching staff	4500
23	2022000060	Jawa...	Alnahdi	Coaching staff	4500
24	2022000061	Duaa	Turkus...	Coaching staff	4500
25	2022000062	Arwaa	Alenzi	Coaching staff	4500
26	2022000066	Najwa	MAhm...	Coaching staff	3200
27	2022000067	Tala	Emad	Coaching staff	3000
28	2022000068	Malak	Sami	Coaching staff	3000
29	2022000072	Salma	Mazen	Coaching staff	3200
30	2022000073	Hala	Hamad	Coaching staff	3000
31	2022000074	Haifa	Foad	Coaching staff	3000

Query executed successfully.

After update:

Results Messages

	EmpID	Fname	Lname	DepName	salary
1	2022000009	Alhanuf	Algulsai	Coaching staff	3900
2	2022000020	Aya	Kadi	Coaching staff	4400
3	2022000021	Dana	Alsuqaie	Coaching staff	4600
4	2022000022	Dareen	Bashir	Coaching staff	4000
5	2022000026	Layana	Kattouha	Coaching staff	4000
6	2022000027	Laila	Alsaqri	Coaching staff	4200
7	2022000028	Fatima	Najjar	Coaching staff	3800
8	2022000032	Seham	Nahdi	Coaching staff	6000
9	2022000033	Shaden	Nahdi	Coaching staff	8500
10	2022000034	Najwah	Baward	Coaching staff	3250
11	2022000038	Eman	Alwael	Coaching staff	3900
12	2022000039	Amal	Alamoudi	Coaching staff	4500
13	2022000040	Razan	Salleh	Coaching staff	3250
14	2022000043	Asrar	Alali	Coaching staff	3250
15	2022000044	Ebtihag	Hakami	Coaching staff	2600
16	2022000045	Abrar	Faqiha	Coaching staff	4000
17	2022000048	Basm...	Kamoni	Coaching staff	4000
18	2022000049	Walaa	Alnahdi	Coaching staff	3900
19	2022000050	Toqa	Banabilah	Coaching staff	3500
20	2022000054	Asia	Alshehri	Coaching staff	4500
21	2022000055	Malk	Alharbi	Coaching staff	4500
22	2022000056	Afnan	Alahmadi	Coaching staff	4500
23	2022000060	Jawa...	Alnahdi	Coaching staff	4500
24	2022000061	Duaa	Turkus...	Coaching staff	4500
25	2022000062	Arwaa	Alenzi	Coaching staff	4500
26	2022000066	Najwa	MAhm...	Coaching staff	3200
27	2022000067	Tala	Emad	Coaching staff	3900
28	2022000068	Malak	Sami	Coaching staff	3900
29	2022000072	Salma	Mazen	Coaching staff	3200
30	2022000073	Hala	Hamad	Coaching staff	3900
31	2022000074	Haifa	Foad	Coaching staff	3900

Query executed successfully.

8.8 <Deleting previous events>

Delete in Natural Language:

To organize the upcoming events schedule, we must check if an event has already been done so that we can delete it from the system's schedule. Therefore, we deleted the events that have already happened from the database by checking if their starting date has already passed. The date that we have checked the events was 2022-05-15. So all events in the past will be deleted.

SQL Script:

```
delete from EVENTS where Start_Date < CURRENT_TIMESTAMP
```

Caption of the Output:

Before deleting:

Results Messages

	EventName	Start_Date	NumberOfDays	HoursPerDays	City	HostingPlaceName
1	Boxing Competition	2022-03-04	2	3	Riyadh	Flagboxing
2	Horse Riding	2022-05-18	3	5	Jeddah	Alreem.Stabel
3	Mount Hiking	2022-06-01	1	5	Ola	Muntajae Habitas Alola
4	Golf Match	2022-07-12	1	3	Jeddah	Groovy Golf
5	Roller Skating	2022-08-15	3	2	Jeddah	Ice Land
6	Running Marathon	2022-08-20	3	4	Jeddah	Jeddah Waterfront
7	Diving	2022-09-07	2	2	Yanbu	Saudi Diving Center
8	Tennis Match	2022-10-03	1	3	Jeddah	Spin&Smach
9	Biking Marathon	2022-11-15	3	4	Jeddah	Jeddah Waterfront
10	Football Match	2022-12-12	1	3	Jeddah	Jawharat Alajaweed Stadium

After deleting:

Results Messages

	EventName	Start_Date	NumberOfDays	HoursPerDays	City	HostingPlaceName
1	Horse Riding	2022-05-18	3	5	Jeddah	Alreem.Stabel
2	Mount Hiking	2022-06-01	1	5	Ola	Muntajae Habitas Alola
3	Golf Match	2022-07-12	1	3	Jeddah	Groovy Golf
4	Roller Skating	2022-08-15	3	2	Jeddah	Ice Land
5	Running Marathon	2022-08-20	3	4	Jeddah	Jeddah Waterfront
6	Diving	2022-09-07	2	2	Yanbu	Saudi Diving Center
7	Tennis Match	2022-10-03	1	3	Jeddah	Spin&Smach
8	Biking Marathon	2022-11-15	3	4	Jeddah	Jeddah Waterfront
9	Football Match	2022-12-12	1	3	Jeddah	Jawharat Alajaweed Stadium

Since the events table is referenced by the Supervised_By table (which is responsible for stroing information about each event's supervisors) using the event's name ,this means that the events that have been deleted from the events table will cascade the deletion of all the rows that references the deleted event. Here is the Supervised_By table before and after deleting the events.

Results Messages

	EventName	supervisorID
1	Biking Marathon	2022000034
2	Biking Marathon	2022000047
3	Boxing Competition	2022000019
4	Boxing Competition	2022000020
5	Diving	2022000021
6	Diving	2022000064
7	Diving	2022000072
8	Football Match	2022000035
9	Football Match	2022000050
10	Golf Match	2022000028
11	Horse Riding	2022000003
12	Horse Riding	2022000032
13	Mount Hiking	2022000004
14	Mount Hiking	2022000055
15	Mount Hiking	2022000062
16	Roller Skating	2022000027
17	Running Marathon	2022000009
18	Running Marathon	2022000025
19	Tennis Match	2022000066
20	Tennis Match	2022000070

Results Messages

	EventName	supervisorID
1	Biking Marathon	2022000034
2	Biking Marathon	2022000047
3	Diving	2022000021
4	Diving	2022000064
5	Diving	2022000072
6	Football Match	2022000035
7	Football Match	2022000050
8	Golf Match	2022000028
9	Horse Riding	2022000003
10	Horse Riding	2022000032
11	Mount Hiking	2022000004
12	Mount Hiking	2022000055
13	Mount Hiking	2022000062
14	Roller Skating	2022000027
15	Running Marathon	2022000009
16	Running Marathon	2022000025
17	Tennis Match	2022000066
18	Tennis Match	2022000070

8.9 <Removing a specific member>

Delete in Natural Language:

Deleting a specific member by using its ID will cascade the deletion of all its health records in the system as well as it will return its locker's owner ID to be null.

SQL Script:

```
delete from MEMBERS where MemberID= 1000000001
```

Caption of the Output:

Members table before deleting the member with ID = 1000000001:

Results Messages

	MemberID	Fname	Mname	Lname	Email	Gender	DateOfBirth	SocialState	PhoneNumber	BranchNo
1	1000000001	Fatima	Omar	Saeed	Fatima909@gmail.com	F	1975-05-17	Married	531287446	1
2	1000000002	Sumaia	Khaled	alqarni	soso100@gmail.com	F	1999-11-18	Single	531233346	4
3	1000000003	Layan	Ahmed	Alrasheed	Layan83@gmail.com	F	2000-01-19	Single	588987446	2
4	1000000004	Hawazen	Naser	Alsalman	Hwazen_90@gmail.com	F	1992-07-22	Married	531278556	3
5	1000000005	Amira	Mohammed	Alfaresi	Memo879@gmail.com	F	1995-05-07	Married	556687446	5

Members table after deleting the member with ID = 1000000001:

Results Messages

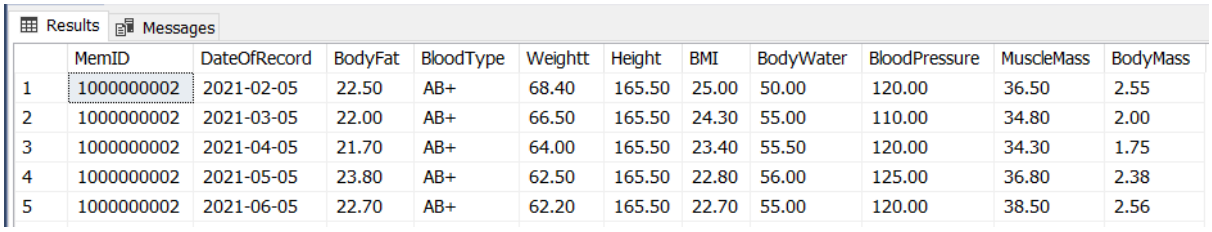
	MemberID	Fname	Mname	Lname	Email	Gender	DateOfBirth	SocialState	PhoneNumber	BranchNo
1	1000000002	Sumaia	Khaled	alqarni	soso100@gmail.com	F	1999-11-18	Single	531233346	4
2	1000000003	Layan	Ahmed	Alrasheed	Layan83@gmail.com	F	2000-01-19	Single	588987446	2
3	1000000004	Hawazen	Naser	Alsalman	Hwazen_90@gmail.com	F	1992-07-22	Married	531278556	3
4	1000000005	Amira	Mohammed	Alfaresi	Memo879@gmail.com	F	1995-05-07	Married	556687446	5
5	1000000006	Suha	Mohammed	Alwadayi	Suha1989@gmail.com	F	1989-06-17	Married	531223356	4

HealthRecords table before deleting the member with ID = 1000000001:

Results Messages

	MemID	DateOfRecord	BodyFat	BloodType	Weightt	Height	BMI	BodyWater	BloodPressure	MuscleMass	BodyMass
1	1000000001	2021-02-01	24.30	O+	80.00	167.00	28.70	48.00	98.00	38.00	2.50
2	1000000001	2021-03-01	25.80	O+	75.00	167.00	26.90	47.80	105.00	39.60	2.04
3	1000000001	2021-04-01	26.30	O+	72.00	167.00	25.80	46.60	113.00	39.90	2.02
4	1000000001	2021-05-01	25.00	O+	68.00	167.00	24.40	46.90	97.00	41.05	2.00
5	1000000002	2021-02-05	22.50	AB+	68.40	165.50	25.00	50.00	120.00	36.50	2.55
6	1000000002	2021-03-05	22.00	AB+	66.50	165.50	24.30	55.00	110.00	34.80	2.00
7	1000000002	2021-04-05	21.70	AB+	64.00	165.50	23.40	55.50	120.00	34.30	1.75
8	1000000002	2021-05-05	23.80	AB+	62.50	165.50	22.80	56.00	125.00	36.80	2.38
9	1000000002	2021-06-05	22.70	AB+	62.20	165.50	22.70	55.00	120.00	38.50	2.56
10	1000000002	2021-07-05	22.50	AB+	61.40	165.50	22.40	56.50	119.00	40.50	2.60
11	1000000002	2021-08-05	22.00	AB+	60.50	165.50	22.10	55.50	120.00	40.80	2.62
12	1000000003	2021-02-11	30.60	O+	75.50	178.50	23.70	55.50	128.00	35.70	2.88

HealthRecords table after deleting the member with ID = 1000000001:



Results | Messages

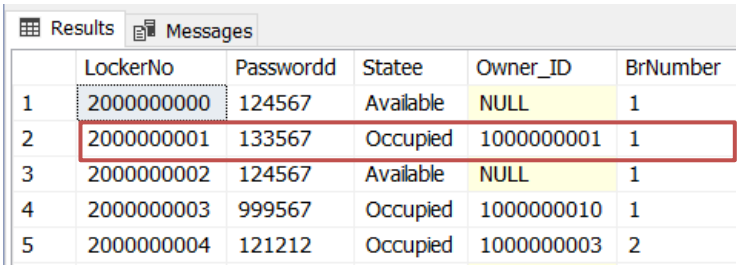
	MemID	DateOfRecord	BodyFat	BloodType	Weightt	Height	BMI	BodyWater	BloodPressure	MuscleMass	BodyMass
1	1000000002	2021-02-05	22.50	AB+	68.40	165.50	25.00	50.00	120.00	36.50	2.55
2	1000000002	2021-03-05	22.00	AB+	66.50	165.50	24.30	55.00	110.00	34.80	2.00
3	1000000002	2021-04-05	21.70	AB+	64.00	165.50	23.40	55.50	120.00	34.30	1.75
4	1000000002	2021-05-05	23.80	AB+	62.50	165.50	22.80	56.00	125.00	36.80	2.38
5	1000000002	2021-06-05	22.70	AB+	62.20	165.50	22.70	55.00	120.00	38.50	2.56

Since the owner' ID references the member ID, when deleting the ID that is equal to 1000000001of the members, the locker that had the owner ID equal to 1000000001 will also change its owner ID to become null as we wrote in the alter table statement. Since the owner ID has become null the state and password of the locker will remain the same. So we must update the lockers state and password to the default values set in the create table statements.

SQL Script:

```
update lockers
set Statee = default  , Passwordd= default
where LockerNo= 2000000001
```

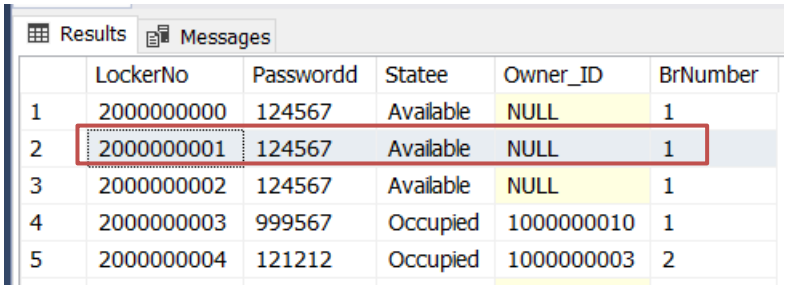
Lockers table before deleting the member with ID = 1000000001:



Results | Messages

	LockerNo	Passwordd	Statee	Owner_ID	BrNumber
1	2000000000	124567	Available	NULL	1
2	2000000001	133567	Occupied	1000000001	1
3	2000000002	124567	Available	NULL	1
4	2000000003	999567	Occupied	1000000010	1
5	2000000004	121212	Occupied	1000000003	2

Lockers table after deleting the member with ID = 1000000001:



Results | Messages

	LockerNo	Passwordd	Statee	Owner_ID	BrNumber
1	2000000000	124567	Available	NULL	1
2	2000000001	124567	Available	NULL	1
3	2000000002	124567	Available	NULL	1
4	2000000003	999567	Occupied	1000000010	1
5	2000000004	121212	Occupied	1000000003	2

9. APPENDIX:

1. The Employees table:

Results Messages

	EmpID	Fname	Mname	Lname	DofBirth	PhoneNo	Gender	Salary	BankAccount	City	Street	BuildingNumber	DepNo	BrNo
1	2022000001	Sarah	Maher	Abukhammas	2000-04-28	569084148	F	3000	12566789234254	jeddah	alrayan	67	1	5
2	2022000002	Shahad	omar	alnahdi	1996-06-03	505678923	F	4500	12234986578234	jeddah	heraa	44	2	6
3	2022000003	Asma	Saleh	Alwael	1998-06-18	596000666	F	5000	12367889954321	jeddah	sari	03	4	4
4	2022000004	seham	khaldon	nahlawi	2000-09-06	534868411	F	3500	12333332254546	jeddah	Almuzi	12	4	3
5	2022000005	raghad	musa	Alghamdi	2000-09-02	567892222	F	3000	12888886734567	jeddah	quraish	06	3	2
6	2022000006	Arwa	Maher	Abukhammas	1997-12-06	566666232	F	2000	12236788809096	jeddah	Sari	55	3	1
7	2022000007	Maha	Saif	Binladen	1993-09-24	556565644	F	2400	12000054367894	jeddah	princeMajed	23	3	4
8	2022000008	Mayson	Omar	Alqurashi	1995-09-23	556145644	F	3500	12495867675456	Riyadh	Tahlia	21	2	7
9	2022000009	Alhanuf	Faisal	Algulsai	1998-02-02	5557895467	F	3000	12600070003456	Riyadh	Tahlia	65	5	8
10	2022000010	Esraa	Hussain	Alahmadi	1996-05-14	588866445	F	2500	12987654333567	Riyadh	Olia	66	3	7
11	2022000011	Haifaa	Saad	Alahmadi	1999-11-09	523232345	F	3000	12345678999554	Riyadh	PrinceKhalid	08	3	9
12	2022000012	Alyaa	Adel	Sait	1995-12-30	505054444	F	4000	12676767634345	Riyadh	Olia	98	3	8
13	2022000013	Salma	Yahia	Albaity	1990-08-29	567667678	F	3500	12393939397777	Jeddah	Tahlia	78	3	5
14	2022000014	Bayan	Muha...	Fahad	1997-11-29	529292911	F	2500	12345543666789	Jeddah	Heraa	88	3	10
15	2022000015	Ebtes...	Saleh	Amer	1995-03-27	505054784	F	4000	12789145788234	Jeddah	Alhamdanih	54	3	6
16	2022000016	Samia	Ahmed	Alwael	1989-08-20	505051184	F	4500	12543882217646	Jeddah	Alhamraa	85	3	3
17	2022000017	Asalah	Hazem	Almansour	1990-03-25	568411177	F	3000	12338764539289	Jeddah	Heraa	45	1	1
18	2022000018	Elaf	Muha...	ALjareed	2000-05-16	539367111	F	3500	12338746575983	Jeddah	Shatie	33	2	1
19	2022000019	Areeb	Khalid	Almutairi	1980-07-13	565552425	F	5000	12287553905753	Jeddah	Muhamm...	24	4	1
20	2022000020	Aya	Khalil	Kadi	1997-05-06	554636872	F	4400	12673874592635	Jeddah	Marwah	65	5	1
21	2022000021	Dana	Mustafa	Alsuqaie	1995-08-17	533265888	F	4600	12453787459386	Jeddah	Safa	40	5	1
22	2022000022	Dareen	Mahm...	Bashir	1977-01-24	545866552	F	5200	12346986432973	Jeddah	Zahra	22	5	1
23	2022000023	Deema	Abdul...	Nass	1988-02-02	530459994	F	5500	12347654987228	Jeddah	Salamah	12	1	2
24	2022000024	Esraa	Maan	Abualdouh	1999-03-09	529487697	F	3000	12438763981287	Jeddah	Nahda	36	2	2
25	2022000025	Sama	Fahad	Alyafi	1982-09-01	598866640	F	4700	12653987645392	Jeddah	Aziziyah	75	4	2
26	2022000026	Layana	Saif	Kattouha	1991-04-10	548559088	F	4000	12438915273407	Jeddah	Alsamer	89	5	2
27	2022000027	Laila	Ahmed	Alsaqri	1985-08-28	507526770	F	4200	12097245639467	Jeddah	Albwadi	39	5	2
28	2022000028	Fatima	Nasir	Najjar	1997-09-01	549872083	F	3800	12092739469501	Jeddah	Alnaseem	27	5	2
29	2022000029	Seham	Khaldon	Nahlawi	2000-09-04	534868411	F	3500	12333332904546	jeddah	alAziza	12	4	3
30	2022000030	Sedra	Khaldon	Nahlawi	2003-03-02	534868221	F	3500	12338332254546	jeddah	almuzi	30	1	3
31	2022000031	Samar	Mahm...	Nahlawi	2005-01-06	534868411	F	5500	12333332254546	jeddah	almuzi	13	2	3

Results Messages

	EmpID	Fname	Mname	Lname	DofBirth	PhoneNo	Gender	Salary	BankAccount	City	Street	BuildingNumber	DepNo	BrNo
32	2022000032	Seham	Khaled	Nahdi	2000-05-05	534468411	F	6000	12389332254546	jeddah	dar alNajah	34	5	3
33	2022000033	Shaden	Khaled	Nahdi	2002-08-03	534863411	F	8500	12395332254546	jeddah	alFallah	18	5	3
34	2022000034	Najwah	Khaleel	Baward	2000-09-06	534862411	F	2500	12332132254546	jeddah	alRagamah	08	5	3
35	2022000035	Amjad	Saeed	Ali	1997-12-10	551102550	F	5000	12114552844025	jeddah	sari	03	4	2
36	2022000036	Aliaa	Saleh	Alwael	1999-08-12	596068666	F	5500	12367679954321	jeddah	sari	12	1	4
37	2022000037	Mai	Omar	Algamdi	1995-12-09	596055666	F	4000	12367339954321	jeddah	quraysh	56	2	4
38	2022000038	Eman	Qassem	Alwael	1994-01-21	596051666	F	3000	12367219954321	jeddah	hira	33	5	4
39	2022000039	Amal	Saleh	Alamoudi	1998-03-29	596052666	F	4500	12367909954321	jeddah	haeel	09	5	4
40	2022000040	Razan	Mahm...	Salleh	1998-06-18	596038666	F	2500	12367119954321	jeddah	almawadah	03	5	4
41	2022000041	Dalal	Ahmad	Sait	1996-12-30	505058974	F	4500	12434512123564	Jeddah	Olia	09	2	5
42	2022000042	Maram	Adel	Banoun	1995-03-05	578787844	F	3000	12176541888995	Jeddah	Heraa	45	4	5
43	2022000043	Asrar	Tariq	Alali	2000-12-23	534343499	F	2500	12989898654321	Jeddah	Anas Bin ...	08	5	5
44	2022000044	Ebtihag	Atiah	Hakami	1997-09-22	532229696	F	2000	12131313454545	Jeddah	Makaronah	83	5	5
45	2022000045	Abrar	Raid	Faqiha	1998-12-20	523232344	F	4000	12234234234567	Jeddah	Sultan	88	5	5
46	2022000046	Sarah	Talal	Sait	1999-12-22	505053422	F	4000	12676787999999	Jeddah	Olia	97	1	6
47	2022000047	Farah	Adel	Mandili	1995-09-20	512121233	F	3000	12675555343435	Jeddah	Kurnaish	09	4	6
48	2022000048	Basmah	Yasser	Kamoni	1995-11-03	598989898	F	4000	12121234345345	Jeddah	Tahlia	88	5	6
49	2022000049	Walaa	Albaraa	Alnahdi	1995-01-06	509090988	F	3000	12232323444445	Jeddah	Rayyan	33	5	6
50	2022000050	Toqa	Bakur	Banabilah	1999-09-09	508888884	F	3500	12009999932321	Jeddah	Tahlia	77	5	6
51	2022000051	Mayson	Omar	Alqurashi	1995-09-23	556145644	F	3500	12495867675456	Riyadh	Tahlia	21	2	7
52	2022000052	Mays...	Hisham	Sindi	1999-05-25	552543289	F	3000	12888867678859	Riyadh	Al Urubah	30	1	7
53	2022000053	Amal	Ahmad	Alseari	1988-07-01	503677676	F	5000	12442000895332	Riyadh	Al Sarhan	78	4	7
54	2022000054	Asia	Mubark	Alshehri	1989-12-02	542298787	F	4500	12220876636654	Riyadh	Kaab Ibn ...	55	5	7
55	2022000055	Malk	Turki	Alharbi	1993-10-18	502346626	F	4500	12115612161811	Riyadh	Al Rif	44	5	7
56	2022000056	Afnan	Khaled	Alahmadi	1992-08-20	553363006	F	4500	12446779006223	Riyadh	Olaya	35	5	7
57	2022000057	Alhanuf	Faisal	Algulsai	1998-02-02	5557895467	F	3500	12600070003456	Riyadh	Tahlia	65	2	8
58	2022000058	Anfal	Fares	Almalki	1991-02-12	501023320	F	3000	126000865532...	Riyadh	Alshahir	60	1	8
59	2022000059	Afrah	Moha...	Alzahrani	1987-11-03	555886433	F	5000	126000992121...	Riyadh	Altwaf	80	4	8
60	2022000060	Jawa...	Faisal	Alnahdi	1998-06-22	540002020	F	4500	12600055500023	Riyadh	Tahlia	72	5	8
61	2022000061	Duaa	Muhsin	Turkustani	1998-03-10	538774343	F	4500	12600040000287	Riyadh	Aldiwaniah	68	5	2
62	2022000062	Arwaa	Ali	Alenzi	1986-03-15	555828368	F	4500	12600098770052	Riyadh	Aljanayin	42	5	8

63	2022000063	Hnaan	Ameen	Alahmadi	1991-06-11	557784426	F	3500	120055514789...	Riyadh	PrinceKhalid	08	1	9
64	2022000064	Jwaher	Suliman	Hameed	1988-05-13	522888105	F	2900	123335698000...	Riyadh	Thlia	09	4	9
65	2022000065	Zenah	Majed	Yaser	1996-11-17	551036662	F	3000	123355800111...	Riyadh	Kaab Ibn ...	17	2	9
66	2022000066	Najwa	Emad	MAhmoud	1985-03-19	558852366	F	3200	12003663128585	Riyadh	Kaab Ibn ...	25	5	9
67	2022000067	Tala	Loai	Emad	1988-06-19	554414231	F	3000	12335211104844	Riyadh	PrinceKhalid	08	5	9
68	2022000068	Malak	Ahmad	Sami	1987-10-13	552063441	F	3000	12003995611052	Riyadh	Tahlia	78	5	9
69	2022000069	Suad	Ahmed	Khaled	1986-11-25	532166678	F	3500	12258886415222	Jeddah	Heraa	88	1	10
70	2022000070	Khoul...	Muha...	Fahad	1996-11-02	556441458	F	2900	12770000258861	Jeddah	Sari	55	4	10
71	2022000071	Ryhana	Ali	Abdullah	1984-08-09	501118546	F	3000	12668522215500	Jeddah	princeMajed	60	2	10
72	2022000072	Salma	Hassan	Mazen	1997-10-07	522356788	F	3200	12321444788445	Jeddah	princeMajed	80	5	10
73	2022000073	Hala	Rayan	Hamad	1986-04-21	544511199	F	3000	12555699330021	Jeddah	Alhamdanih	65	5	10
74	2022000074	Haifa	Fadi	Foad	1985-07-14	552693330	F	3000	12999515002111	Jeddah	Shatie	33	5	10

2. The Member table:

Results Messages

	MemberID	Fname	Mname	Lname	Email	Gender	DateOfBirth	SocialState	PhoneNumber	BranchNo
1	1000000001	Fatima	Omar	Saeed	Fatima909@gmail.com	F	1975-05-17	Married	531287446	1
2	1000000002	Sumaia	Khaled	alqarni	soso100@gmail.com	F	1999-11-18	Single	531233346	4
3	1000000003	Layan	Ahmed	Alrasheed	Layan83@gmail.com	F	2000-01-19	Single	588987446	2
4	1000000004	Hawazen	Naser	Alsalman	Hwazen_90@gmail.com	F	1992-07-22	Married	531278556	3
5	1000000005	Amira	Mohammed	Alfaresi	Memo879@gmail.com	F	1995-05-07	Married	556687446	5
6	1000000006	Suha	Mohammed	Alwadayi	Suha1989@gmail.com	F	1989-06-17	Married	531223356	4
7	1000000007	Samar	Yaser	Abed	Samar_145@gmail.com	F	1998-02-10	Married	56647446	2
8	1000000008	Dareen	Mohammed	barkah	Dareen_2@gmail.com	F	1999-03-05	Married	539985146	3
9	1000000009	Albandari	Sulaman	Alasiri	AlBandari11@gmail.com	F	1998-10-10	Married	558857446	5
10	1000000010	Wed	Omar	Alfaresi	WedOmar@gmail.com	F	1988-09-19	Married	552587446	1
11	1000000011	Raghad	Yasser	Abukha...	raghad2002@hotmail...	F	2002-12-12	Single	505051313	6
12	1000000012	Rahaf	Muhammad	Faisal	rahofaa@gmail.com	F	2000-09-28	Single	568686833	6
13	1000000013	Reem	Anas	Ahmad	reem2000@outluck.c...	F	1998-07-23	Married	599933322	7
14	1000000014	Aeshah	Husssain	Naitah	Aeshaute@hotmail.com	F	1996-11-14	Married	505050544	7
15	1000000015	Amjad	Ali	Alghamdi	Amjad1999@gmail.com	F	1999-09-17	Single	567899876	8
16	1000000016	Khadija	Abdulrazaq	Alharbi	khokha@hotmail.com	F	1980-06-26	Married	512312344	8
17	1000000017	Amnah	Abdulrazaq	Alharbi	moona231@yahoo.c...	F	1977-11-09	Married	543432213	9
18	1000000018	Reema	Ali	Alqurashi	roomaa45@gmail.com	F	1988-02-01	Single	566663232	9
19	1000000019	Sahar	Essam	Alnahdi	SaharEssam@hotmail....	F	1998-07-17	Single	543434399	10
20	1000000020	Hawaa	Muhammad	Qari	hwaa1995@gmail.com	F	1995-12-25	Married	598989822	10
21	1000000021	Seham	Khaldoun	Nahlawi	Seha88@gmail.com	F	2000-05-08	Single	553251588	1
22	1000000022	Shahira	Khaldoun	Nahlawi	Shahira08@gmail.com	F	2002-05-06	Single	559251588	2
23	1000000023	Najah	Khaldoun	Nahlawi	nano66@gmail.com	F	1999-12-05	Single	559951589	3
24	1000000024	Sawsan	Khaldoun	Nahlawi	Sawsan98@gmail.com	F	1997-05-07	Single	550987456	4
25	1000000025	Huda	Khaldoun	Nahlawi	dody76@gmail.com	F	1995-08-03	Single	548896752	5
26	1000000026	Razan	Anas	Nahlawi	rosy837@gmail.com	F	1993-12-25	Single	554351276	6
27	1000000027	Rawan	Anas	Nahlawi	rawaan28@gmail.com	F	1991-08-22	Single	553251568	7
28	1000000028	Remma	Anas	Nahlawi	Remaa7@gmail.com	F	2003-11-11	Single	552389588	8
29	1000000029	Hala	Saad	Nahlawi	halllol38@gmail.com	F	2005-09-13	Single	553268588	9
30	1000000030	Lena	Saad	Nahlawi	lena23@gmail.com	F	1998-04-21	Single	553097588	10

3. The Lockers table:

Results | Messages

	LockerNo	Passwordd	Statee	Owner_ID	BrNumber
1	2000000000	124567	Available	NULL	1
2	2000000001	133567	Occupied	1000000001	1
3	2000000002	124567	Available	NULL	1
4	2000000003	999567	Occupied	1000000010	1
5	2000000004	121212	Occupied	1000000003	2
6	2000000005	124567	Available	NULL	2
7	2000000006	345678	Occupied	1000000007	2
8	2000000007	124567	Available	NULL	2
9	2000000008	154123	Occupied	1000000008	3
10	2000000009	159856	Occupied	1000000004	3
11	2000000010	124567	Available	NULL	3
12	2000000011	999567	Occupied	1000000006	4
13	2000000012	155504	Occupied	1000000002	4
14	2000000013	124567	Available	NULL	4
15	2000000014	155478	Occupied	1000000016	8
16	2000000015	124567	Available	NULL	8
17	2000000016	124567	Available	1000000015	5
18	2000000017	177767	Occupied	1000000009	5
19	2000000018	177067	Occupied	1000000005	5
20	2000000019	124567	Available	NULL	6
21	2000000020	121212	Occupied	1000000011	6
22	2000000021	124567	Available	NULL	6
23	2000000022	124567	Available	NULL	7
24	2000000023	188812	Occupied	1000000014	7
25	2000000024	188812	Occupied	1000000013	7
26	2000000025	124567	Available	NULL	8
27	2000000026	124567	Available	NULL	9
28	2000000027	122236	Occupied	1000000018	9
29	2000000028	144699	Occupied	1000000019	10
30	2000000029	124567	Available	NULL	10
31	2000000030	124567	Available	NULL	10

Results Messages

	LockerNo	Passwordd	Statee	Owner_ID	BrNumber
32	2000000031	124567	Available	NULL	9
33	2000000032	124567	Available	NULL	1
34	2000000033	124567	Available	NULL	1
35	2000000034	124567	Available	NULL	1
36	2000000035	124567	Available	NULL	1
37	2000000036	124567	Available	NULL	1
38	2000000037	124567	Available	NULL	1
39	2000000038	124567	Available	NULL	2
40	2000000039	124567	Available	NULL	2
41	2000000040	124567	Available	NULL	2
42	2000000041	124567	Available	NULL	2
43	2000000042	124567	Available	NULL	2
44	2000000043	124567	Available	NULL	2
45	2000000044	124567	Available	NULL	2
46	2000000045	124567	Available	NULL	2
47	2000000046	124567	Available	NULL	2
48	2000000047	124567	Available	NULL	2
49	2000000048	124567	Available	NULL	2
50	2000000049	124567	Available	NULL	3
51	2000000050	124567	Available	NULL	3
52	2000000051	124567	Available	NULL	3
53	2000000052	124567	Available	NULL	3
54	2000000053	124567	Available	NULL	3
55	2000000054	124567	Available	NULL	3
56	2000000055	124567	Available	NULL	4
57	2000000056	124567	Available	NULL	4
58	2000000057	124567	Available	NULL	4
59	2000000058	124567	Available	NULL	4
60	2000000059	124567	Available	NULL	4
61	2000000060	124567	Available	NULL	4
62	2000000061	124567	Available	NULL	4

Results | Messages

	LockerNo	Passwordd	Statee	Owner_ID	BrNumber
63	2000000062	124567	Available	NULL	4
64	2000000063	124567	Available	NULL	4
65	2000000064	124567	Available	NULL	4
66	2000000065	124567	Available	NULL	4
67	2000000066	124567	Available	NULL	4
68	2000000067	124567	Available	NULL	5
69	2000000068	124567	Available	NULL	5
70	2000000069	124567	Available	NULL	5
71	2000000070	124567	Available	NULL	5
72	2000000071	124567	Available	NULL	5
73	2000000072	124567	Available	NULL	5
74	2000000073	124567	Available	NULL	5
75	2000000074	124567	Available	NULL	6
76	2000000075	124567	Available	NULL	6
77	2000000076	124567	Available	NULL	6
78	2000000077	124567	Available	NULL	6
79	2000000078	124567	Available	NULL	6
80	2000000079	124567	Available	NULL	6
81	2000000080	124567	Available	NULL	6
82	2000000081	124567	Available	NULL	6
83	2000000082	124567	Available	NULL	6
84	2000000083	124567	Available	NULL	6
85	2000000084	124567	Available	NULL	6
86	2000000085	124567	Available	NULL	6
87	2000000086	124567	Available	NULL	7
88	2000000087	124567	Available	NULL	7
89	2000000088	124567	Available	NULL	7
90	2000000089	124567	Available	NULL	7
91	2000000090	124567	Available	NULL	7
92	2000000091	124567	Available	NULL	7
93	2000000092	124567	Available	NULL	7

94	2000000093	124567	Available	NULL	7
95	2000000094	124567	Available	NULL	8
96	2000000095	124567	Available	NULL	8
97	2000000096	124567	Available	NULL	8
98	2000000097	124567	Available	NULL	8
99	2000000098	124567	Available	NULL	8
100	2000000099	124567	Available	NULL	8
101	2000000100	124567	Available	NULL	8
102	2000000101	124567	Available	NULL	8
103	2000000102	124567	Available	NULL	9
104	2000000103	124567	Available	NULL	9
105	2000000104	124567	Available	NULL	9
106	2000000105	124567	Available	NULL	9
107	2000000106	124567	Available	NULL	9
108	2000000107	124567	Available	NULL	9
109	2000000108	124567	Available	NULL	9
110	2000000109	124567	Available	NULL	10
111	2000000110	124567	Available	NULL	10
112	2000000111	124567	Available	NULL	10
113	2000000112	124567	Available	NULL	10
114	2000000113	124567	Available	NULL	10
115	2000000114	124567	Available	NULL	10
116	2000000115	124567	Available	NULL	10

4. The Branches table:

Results | Messages

	BranchNumber	City	Street	Building_No	PhoneNum	BranchCpacity	Opening_Hours	Closing_Hours	Manager_ID
1	1	Jeddah	Omer Bin Al Kattab	145	567998321	150	08:30:00	20:30:00	2022000006
2	2	Jeddah	Bani Malik	109	553152465	300	09:30:00	21:30:00	2022000005
3	3	Jeddah	Al Balad	389	566769051	100	08:30:00	23:00:00	2022000016
4	4	Jeddah	Almakruna	231	549946752	250	10:30:00	22:30:00	2022000007
5	5	Jeddah	Sari	127	559678892	150	11:30:00	23:30:00	2022000013
6	6	Jeddah	Heraa	201	567880321	200	08:00:00	20:00:00	2022000015
7	7	Riyadh	Tahlia	155	589223478	125	07:30:00	23:00:00	2022000010
8	8	Riyadh	king Khalid street	110	533908321	150	09:00:00	23:00:00	2022000012
9	9	Riyadh	King Faisal	156	508998321	120	08:00:00	22:00:00	2022000011
10	10	Jeddah	Filastin	240	567799832	150	08:30:00	23:00:00	2022000014

5. The Departments table:

	DepNumber	DepName	Mgr_ID
1	1	Maintenance	2022000001
2	2	Customer service	2022000002
3	3	Management	2022000003
4	4	Medical care	2022000004
5	5	Coaching staff	2022000009

6. The Health records table:

Results | Messages

	MemID	DateOfRecord	BodyFat	BloodType	Weightt	Height	BMI	BodyWater	BloodPressure	MuscleMass	BodyMass
1	1000000001	2021-02-01	24.30	O+	80.00	167.00	28.70	48.00	98.00	38.00	2.50
2	1000000001	2021-03-01	25.80	O+	75.00	167.00	26.90	47.80	105.00	39.60	2.04
3	1000000001	2021-04-01	26.30	O+	72.00	167.00	25.80	46.60	113.00	39.90	2.02
4	1000000001	2021-05-01	25.00	O+	68.00	167.00	24.40	46.90	97.00	41.05	2.00
5	1000000002	2021-02-05	22.50	AB+	68.40	165.50	25.00	50.00	120.00	36.50	2.55
6	1000000002	2021-03-05	22.00	AB+	66.50	165.50	24.30	55.00	110.00	34.80	2.00
7	1000000002	2021-04-05	21.70	AB+	64.00	165.50	23.40	55.50	120.00	34.30	1.75
8	1000000002	2021-05-05	23.80	AB+	62.50	165.50	22.80	56.00	125.00	36.80	2.38
9	1000000002	2021-06-05	22.70	AB+	62.20	165.50	22.70	55.00	120.00	38.50	2.56
10	1000000002	2021-07-05	22.50	AB+	61.40	165.50	22.40	56.50	119.00	40.50	2.60
11	1000000002	2021-08-05	22.00	AB+	60.50	165.50	22.10	55.50	120.00	40.80	2.62
12	1000000003	2021-02-11	30.60	O+	75.50	178.50	23.70	55.50	128.00	35.70	2.88
13	1000000003	2021-03-11	28.00	O+	70.50	178.50	22.10	54.80	125.00	35.80	2.70
14	1000000003	2021-04-11	27.50	O+	68.50	178.50	21.50	55.80	120.00	35.70	2.67
15	1000000003	2021-05-11	25.70	O+	67.70	178.50	21.20	55.50	118.00	37.40	2.66
16	1000000003	2021-06-11	24.00	O+	67.00	178.50	21.00	56.00	120.00	37.80	2.66
17	1000000003	2021-07-11	23.80	O+	66.50	178.50	20.90	55.00	120.00	38.00	2.63
18	1000000003	2021-08-11	22.50	O+	65.00	178.50	20.40	55.50	120.00	39.70	2.66
19	1000000004	2021-03-15	22.33	A-	98.00	160.00	38.33	47.30	100.00	42.05	1.88
20	1000000004	2021-04-15	23.40	A-	90.00	160.00	35.20	47.80	110.00	42.60	1.90
21	1000000004	2021-05-15	23.76	A-	84.00	160.00	32.80	48.40	96.00	42.03	2.00
22	1000000004	2021-06-15	26.80	A-	72.00	160.00	28.10	48.80	112.00	44.03	2.23
23	1000000005	2021-03-15	15.22	B-	79.00	154.00	33.30	43.06	110.00	45.00	2.40
24	1000000005	2021-04-15	18.98	B-	73.00	154.00	30.80	45.77	117.00	41.98	2.32
25	1000000005	2021-05-15	20.33	B-	68.00	154.00	28.70	48.80	117.00	43.30	2.98
26	1000000005	2021-06-15	22.87	B-	65.00	154.00	27.40	47.90	106.00	48.93	2.32
27	1000000006	2021-04-07	21.50	A+	50.30	156.00	20.70	45.80	110.00	36.20	1.80
28	1000000006	2021-05-07	21.80	A+	51.00	156.00	21.00	46.00	100.00	36.70	2.00
29	1000000006	2021-06-07	21.70	A+	51.50	156.00	21.20	46.30	111.00	38.10	2.20
30	1000000006	2021-07-07	22.00	A+	52.60	156.00	21.60	46.50	110.00	38.40	2.40
31	1000000007	2021-04-15	15.80	B+	45.50	158.50	18.10	44.00	108.00	30.00	1.67

Results Messages

	MemID	DateOfRecord	BodyFat	BloodType	Weightt	Height	BMI	BodyWater	BloodPressure	MuscleMass	BodyMass
32	1000000007	2021-05-15	17.00	B+	47.00	158.50	18.70	46.50	110.00	32.00	1.69
33	1000000007	2021-06-15	19.10	B+	47.50	158.50	18.90	47.00	118.00	33.50	1.87
34	1000000007	2021-07-15	19.80	B+	48.20	158.50	19.20	47.70	120.00	34.00	1.89
35	1000000007	2021-08-15	19.80	B+	48.50	158.50	19.30	50.00	120.00	34.30	2.22
36	1000000007	2021-09-15	20.10	B+	49.50	158.50	19.70	50.00	123.00	34.80	2.25
37	1000000007	2021-10-15	20.20	B+	50.10	158.50	20.30	51.00	120.00	35.00	2.35
38	1000000008	2021-05-15	27.00	A-	90.00	180.00	27.80	49.00	220.00	42.70	2.50
39	1000000008	2021-06-15	26.80	A-	89.00	180.00	27.50	49.00	190.00	41.40	2.60
40	1000000008	2021-07-15	26.00	A-	88.00	180.00	27.20	49.70	200.00	40.90	2.66
41	1000000008	2021-08-15	25.90	A-	87.00	180.00	26.80	50.00	180.00	40.70	2.00
42	1000000008	2021-09-15	25.30	A-	86.00	180.00	26.50	49.50	170.00	39.80	1.90
43	1000000008	2021-10-15	25.00	A-	85.00	180.00	26.20	48.00	160.00	39.50	1.89
44	1000000008	2021-11-15	24.80	A-	85.00	180.00	26.20	48.10	159.00	36.70	1.50
45	1000000008	2021-12-15	24.10	A-	84.00	180.00	25.90	47.00	140.00	36.47	1.50
46	1000000008	2022-01-15	24.00	A-	83.00	180.00	25.60	47.00	150.00	37.70	1.70
47	1000000008	2022-02-15	23.60	A-	82.00	180.00	25.30	44.00	140.00	37.50	1.40
48	1000000008	2022-03-15	23.80	A-	80.00	180.00	24.60	43.00	130.00	36.70	1.30
49	1000000008	2022-04-15	23.20	A-	79.00	180.00	24.40	43.20	120.00	36.90	1.53
50	1000000009	2021-06-15	22.06	AB-	68.00	154.00	28.07	45.00	108.00	43.00	1.78
51	1000000009	2021-07-15	22.87	AB-	64.00	154.00	27.00	42.90	106.00	43.00	1.78
52	1000000009	2021-08-15	22.87	AB-	64.00	154.00	27.00	41.80	117.00	43.00	1.90
53	1000000009	2021-09-15	25.37	AB-	65.00	154.00	27.40	44.80	110.00	45.98	1.76
54	1000000009	2021-10-15	21.07	AB-	65.00	154.00	27.40	46.70	120.00	42.87	1.92
55	1000000009	2021-11-15	26.77	AB-	63.00	154.00	26.60	43.80	111.00	42.78	1.82
56	1000000009	2021-12-15	23.97	AB-	64.00	154.00	27.40	45.00	119.00	43.76	1.82
57	1000000009	2022-01-15	22.00	AB-	63.00	154.00	26.60	47.80	120.00	45.76	1.82
58	1000000009	2022-02-15	21.55	AB-	62.00	154.00	26.10	43.90	115.00	43.63	1.92
59	1000000009	2022-03-15	25.00	AB-	56.00	154.00	23.06	42.90	116.00	41.93	1.92
60	1000000009	2022-04-15	25.08	AB-	56.00	154.00	23.06	42.90	119.00	40.73	1.92
61	1000000010	2022-01-01	23.60	O+	60.20	165.00	22.10	45.70	120.00	37.70	2.70
62	1000000010	2022-02-01	24.00	O+	62.30	165.00	22.90	46.00	115.00	38.10	2.80
63	1000000010	2022-03-01	24.10	O+	61.40	165.00	22.60	46.20	111.00	37.80	2.00
64	1000000010	2022-04-01	23.40	O+	59.80	165.00	22.00	45.80	112.00	37.60	2.40
65	1000000011	2022-01-15	20.10	B+	48.90	154.00	20.60	42.30	110.00	34.80	1.50
66	1000000011	2022-02-15	20.80	B+	49.50	154.00	20.90	43.40	114.00	35.00	1.70
67	1000000011	2022-03-15	21.20	B+	50.20	154.00	21.20	44.20	116.00	35.40	1.90
68	1000000011	2022-04-15	22.00	B+	51.30	154.00	21.60	45.00	118.00	36.00	2.20
69	1000000012	2022-02-01	21.40	A+	65.00	160.00	25.40	45.00	120.00	34.50	1.67
70	1000000012	2022-03-01	23.10	A+	67.00	160.00	26.20	48.00	117.00	36.50	1.66
71	1000000012	2022-04-01	22.40	A+	66.00	160.00	25.80	46.00	129.00	35.50	1.97
72	1000000012	2022-05-01	20.80	A+	63.00	160.00	24.60	43.00	100.00	37.50	2.60
73	1000000013	2022-02-20	15.80	B+	40.00	150.00	17.80	42.10	99.00	27.50	1.60
74	1000000013	2022-03-20	16.40	B+	42.00	150.00	18.70	45.30	80.00	28.90	1.80
75	1000000013	2022-04-20	18.20	B+	45.00	150.00	20.00	46.70	112.00	30.70	2.00
76	1000000014	2022-03-01	36.00	AB-	60.00	150.00	25.50	46.70	140.00	30.70	2.66
77	1000000014	2022-04-01	34.70	AB-	61.00	150.00	27.10	46.70	90.00	30.40	2.90
78	1000000014	2022-05-01	33.60	AB-	59.00	150.00	26.20	46.70	120.00	30.90	1.00
79	1000000015	2022-04-04	25.74	A+	75.23	168.20	23.50	39.50	129.90	36.50	1.32
80	1000000015	2022-05-04	22.80	A+	68.41	168.20	20.50	42.36	120.10	42.11	2.15
81	1000000016	2022-03-21	28.30	O+	80.30	175.50	26.30	45.30	130.40	20.30	1.64
82	1000000016	2022-04-21	25.10	O+	71.20	175.50	24.60	42.10	125.30	29.30	1.85
83	1000000017	2022-02-01	32.20	AB+	120.30	166.20	38.90	41.20	140.30	25.30	1.45
84	1000000017	2022-03-01	29.40	AB+	110.40	166.20	35.30	45.30	134.50	27.30	1.55
85	1000000017	2022-04-01	26.50	AB+	92.50	166.20	29.10	46.70	128.20	35.30	1.66
86	1000000017	2022-05-01	24.30	AB+	85.30	166.20	25.70	48.90	125.30	41.20	1.76
87	1000000018	2022-04-20	23.00	O-	80.00	177.00	25.50	59.00	110.00	40.70	2.50
88	1000000019	2022-05-01	24.50	B+	89.20	161.40	27.40	38.20	125.30	33.40	1.78

7. The Sports equipment table:

Results Messages

	Machine_Number	MachineName	State	Bnum
1	5000000001	Treadmill	In Service	1
2	5000000002	Treadmill	In Service	1
3	5000000003	Treadmill	Out of s...	1
4	5000000004	Treadmill	In Service	2
5	5000000005	Treadmill	In Service	2
6	5000000006	Treadmill	In Service	2
7	5000000007	Treadmill	In Service	3
8	5000000008	Treadmill	In Service	3
9	5000000009	Treadmill	In Service	3
10	5000000010	Treadmill	In Service	3
11	5000000011	Treadmill	In Service	3
12	5000000012	Treadmill	In Service	4
13	5000000013	Treadmill	In Service	4
14	5000000014	Treadmill	Out of s...	4
15	5000000015	Treadmill	In Service	4
16	5000000016	Treadmill	In Service	5
17	5000000017	Treadmill	In Service	5
18	5000000018	Treadmill	In Service	5
19	5000000019	Treadmill	In Service	6
20	5000000020	Treadmill	In Service	6
21	5000000021	Treadmill	In Service	6
22	5000000022	Treadmill	In Service	6
23	5000000023	Treadmill	In Service	7
24	5000000024	Treadmill	In Service	7
25	5000000025	Treadmill	In Service	7
26	5000000026	Treadmill	In Service	8
27	5000000027	Treadmill	In Service	8
28	5000000028	Treadmill	In Service	8
29	5000000029	Treadmill	In Service	9
30	5000000030	Treadmill	In Service	9
31	5000000031	Treadmill	In Service	9

Results Messages

	Machine_Number	MachineName	State	Bnum
32	5000000032	Treadmill	In Service	10
33	5000000033	Treadmill	In Service	10
34	5000000034	Treadmill	In Service	10
35	5000000035	Treadmill	In Service	10
36	5000000036	Treadmill	In Service	10
37	5000000037	Elliptical Ma...	In Service	1
38	5000000038	Elliptical Ma...	In Service	1
39	5000000039	Elliptical Ma...	In Service	1
40	5000000040	Elliptical Ma...	Out of service	2
41	5000000041	Elliptical Ma...	In Service	2
42	5000000042	Elliptical Ma...	In Service	2
43	5000000043	Elliptical Ma...	In Service	3
44	5000000044	Elliptical Ma...	In Service	3
45	5000000045	Elliptical Ma...	In Service	3
46	5000000046	Elliptical Ma...	In Service	3
47	5000000047	Elliptical Ma...	In Service	3
48	5000000048	Elliptical Ma...	In Service	4
49	5000000049	Elliptical Ma...	In Service	4
50	5000000050	Elliptical Ma...	In Service	4
51	5000000051	Elliptical Ma...	In Service	4
52	5000000052	Elliptical Ma...	In Service	5
53	5000000053	Elliptical Ma...	In Service	5
54	5000000054	Elliptical Ma...	Out of service	5
55	5000000055	Elliptical Ma...	In Service	6
56	5000000056	Elliptical Ma...	In Service	6
57	5000000057	Elliptical Ma...	In Service	6
58	5000000058	Elliptical Ma...	In Service	6
59	5000000059	Elliptical Ma...	In Service	7
60	5000000060	Elliptical Ma...	In Service	7
61	5000000061	Elliptical Ma...	In Service	7

Results Messages

	Machine_Number	MachineName	State	Bnum
62	5000000062	Elliptical Ma...	In Service	8
63	5000000063	Elliptical Ma...	In Service	8
64	5000000064	Elliptical Ma...	In Service	8
65	5000000065	Elliptical Ma...	In Service	9
66	5000000066	Elliptical Ma...	In Service	9
67	5000000067	Elliptical Ma...	In Service	9
68	5000000068	Elliptical Ma...	In Service	10
69	5000000069	Elliptical Ma...	In Service	10
70	5000000070	Elliptical Ma...	In Service	10
71	5000000071	Elliptical Ma...	In Service	10
72	5000000072	Elliptical Ma...	In Service	10
73	5000000073	Stationary ...	In Service	1
74	5000000074	Stationary ...	In Service	1
75	5000000075	Stationary ...	In Service	1
76	5000000076	Stationary ...	In Service	2
77	5000000077	Stationary ...	In Service	2
78	5000000078	Stationary ...	In Service	2
79	5000000079	Stationary ...	In Service	3
80	5000000080	Stationary ...	In Service	3
81	5000000081	Stationary ...	In Service	3
82	5000000082	Stationary ...	In Service	3
83	5000000083	Stationary ...	In Service	3
84	5000000084	Stationary ...	In Service	4
85	5000000085	Stationary ...	In Service	4
86	5000000086	Stationary ...	In Service	4
87	5000000087	Stationary ...	In Service	4
88	5000000088	Stationary ...	In Service	5
89	5000000089	Stationary ...	In Service	5
90	5000000090	Stationary ...	In Service	5
91	5000000091	Stationary ...	In Service	6
92	5000000092	Stationary ...	In Service	6

Results Messages

	Machine_Number	MachineName	State	Bnum
93	5000000093	Stationary ...	Out of service	6
94	5000000094	Stationary ...	In Service	6
95	5000000095	Stationary ...	In Service	7
96	5000000096	Stationary ...	In Service	7
97	5000000097	Stationary ...	Out of service	7
98	5000000098	Stationary ...	In Service	8
99	5000000099	Stationary ...	In Service	8
100	5000000100	Stationary ...	In Service	8
101	5000000101	Stationary ...	In Service	9
102	5000000102	Stationary ...	In Service	9
103	5000000103	Stationary ...	In Service	9
104	5000000104	Stationary ...	In Service	10
105	5000000105	Stationary ...	In Service	10
106	5000000106	Stationary ...	In Service	10
107	5000000107	Stationary ...	In Service	10
108	5000000108	Stationary ...	In Service	10
109	5000000109	Stair Climber	In Service	1
110	5000000110	Stair Climber	In Service	1
111	5000000111	Stair Climber	In Service	1
112	5000000112	Stair Climber	In Service	2
113	5000000113	Stair Climber	In Service	2
114	5000000114	Stair Climber	In Service	2
115	5000000115	Stair Climber	In Service	3
116	5000000116	Stair Climber	Out of service	3
117	5000000117	Stair Climber	In Service	3
118	5000000118	Stair Climber	In Service	3
119	5000000119	Stair Climber	In Service	3
120	5000000120	Stair Climber	In Service	4
121	5000000121	Stair Climber	In Service	4
122	5000000122	Stair Climber	In Service	4
123	5000000123	Stair Climber	In Service	4

Results Messages

	Machine_Number	MachineName	State	Bnum
124	5000000124	Stair Climber	In Service	5
125	5000000125	Stair Climber	In Service	5
126	5000000126	Stair Climber	In Service	5
127	5000000127	Stair Climber	In Service	6
128	5000000128	Stair Climber	In Service	6
129	5000000129	Stair Climber	In Service	6
130	5000000130	Stair Climber	In Service	6
131	5000000131	Stair Climber	In Service	7
132	5000000132	Stair Climber	In Service	7
133	5000000133	Stair Climber	In Service	7
134	5000000134	Stair Climber	Out of service	8
135	5000000135	Stair Climber	In Service	8
136	5000000136	Stair Climber	In Service	8
137	5000000137	Stair Climber	In Service	9
138	5000000138	Stair Climber	In Service	9
139	5000000139	Stair Climber	In Service	9
140	5000000140	Stair Climber	Out of service	10
141	5000000141	Stair Climber	In Service	10
142	5000000142	Stair Climber	In Service	10
143	5000000143	Stair Climber	In Service	10
144	5000000144	Stair Climber	In Service	10
145	5000000145	Leg Press	In Service	1
146	5000000146	Leg Press	Out of Servi...	1
147	5000000147	Leg Press	In Service	1
148	5000000148	Leg Press	In Service	2
149	5000000149	Leg Press	In Service	2
150	5000000150	Leg Press	In Service	2
151	5000000151	Leg Press	In Service	3
152	5000000152	Leg Press	In Service	3
153	5000000153	Leg Press	Out of service	3
154	5000000154	Leg Press	In Service	3

Results Messages

	Machine_Number	MachineName	State	Bnum
155	5000000155	Leg Press	In Service	3
156	5000000156	Leg Press	In Service	4
157	5000000157	Leg Press	In Service	4
158	5000000158	Leg Press	In Service	4
159	5000000159	Leg Press	In Service	4
160	5000000160	Leg Press	In Service	5
161	5000000161	Leg Press	In Service	5
162	5000000162	Leg Press	In Service	5
163	5000000163	Leg Press	In Service	6
164	5000000164	Leg Press	In Service	6
165	5000000165	Leg Press	In Service	6
166	5000000166	Leg Press	In Service	6
167	5000000167	Leg Press	In Service	7
168	5000000168	Leg Press	In Service	7
169	5000000169	Leg Press	In Service	7
170	5000000170	Leg Press	In Service	8
171	5000000171	Leg Press	In Service	8
172	5000000172	Leg Press	In Service	8
173	5000000173	Leg Press	In Service	9
174	5000000174	Leg Press	Out of service	9
175	5000000175	Leg Press	In Service	9
176	5000000176	Leg Press	In Service	10
177	5000000177	Leg Press	In Service	10
178	5000000178	Leg Press	Out of service	10
179	5000000179	Leg Press	In Service	10
180	5000000180	Leg Press	In Service	10
181	5000000181	Smith Machi...	In Service	1
182	5000000182	Smith Machi...	In Service	1
183	5000000183	Smith Machi...	In Service	2
184	5000000184	Smith Machi...	In Service	2
185	5000000185	Smith Machi...	In Service	2

Results Messages

	Machine_Number	MachineName	State	Bnum
186	5000000186	Smith Machi...	Out of service	3
187	5000000187	Smith Machi...	In Service	3
188	5000000188	Smith Machi...	In Service	3
189	5000000189	Smith Machi...	In Service	3
190	5000000190	Smith Machi...	In Service	3
191	5000000191	Smith Machi...	In Service	4
192	5000000192	Smith Machi...	Out of service	4
193	5000000193	Smith Machi...	In Service	4
194	5000000194	Smith Machi...	In Service	4
195	5000000195	Smith Machi...	In Service	5
196	5000000196	Smith Machi...	In Service	5
197	5000000197	Smith Machi...	In Service	5
198	5000000198	Smith Machi...	In Service	6
199	5000000199	Smith Machi...	In Service	6
200	5000000200	Smith Machi...	In Service	6
201	5000000201	Smith Machi...	In Service	6
202	5000000202	Smith Machi...	In Service	7
203	5000000203	Smith Machi...	In Service	7
204	5000000204	Smith Machi...	In Service	7
205	5000000205	Smith Machi...	In Service	8
206	5000000206	Smith Machi...	In Service	8
207	5000000207	Smith Machi...	In Service	8
208	5000000208	Smith Machi...	In Service	9
209	5000000209	Smith Machi...	In Service	9
210	5000000210	Smith Machi...	In Service	9
211	5000000211	Smith Machi...	In Service	10
212	5000000212	Smith Machi...	In Service	10
213	5000000213	Smith Machi...	Out of service	10
214	5000000214	Smith Machi...	In Service	10
215	5000000215	Smith Machi...	In Service	10
216	5000000216	Smith Machi...	In Service	1

8. The Contracts table:

Results Messages

	ContractID	StartDate	EndDate	PaymentMethod	AppliedDiscounts	AppliedTaxes	DateOfPayment
1	4000000001	2021-02-01	2021-05-01	cash	0.10	0.15	2021-01-31
2	4000000002	2021-02-05	2021-08-05	cash	NULL	0.15	2021-02-01
3	4000000003	2021-02-11	2021-08-11	credit card	NULL	0.15	2021-02-04
4	4000000004	2021-03-15	2021-06-15	cash	0.20	0.15	2021-03-12
5	4000000005	2021-03-15	2021-06-15	credit card	NULL	0.15	2021-03-12
6	4000000006	2021-04-07	2021-07-07	credit card	NULL	0.15	2021-04-04
7	4000000007	2021-04-15	2021-10-15	credit card	0.10	0.15	2021-04-15
8	4000000008	2021-05-15	2022-05-15	cash	0.10	0.15	2021-05-07
9	4000000009	2021-06-15	2022-06-15	credit card	NULL	0.15	2021-06-13
10	4000000010	2022-01-01	2022-04-01	cash	0.10	0.15	2021-12-30
11	4000000011	2022-01-15	2022-04-15	cash	0.15	0.15	2022-01-14
12	4000000012	2022-02-01	2022-05-01	credit card	NULL	0.15	2022-01-31
13	4000000013	2022-02-20	2022-08-20	credit card	0.10	0.15	2022-01-31
14	4000000014	2022-03-01	2022-09-01	credit card	0.30	0.15	2022-02-27
15	4000000015	2022-04-04	2022-10-04	cash	0.10	0.15	2022-04-02
16	4000000016	2022-03-21	2023-03-21	cash	0.25	0.15	2022-03-19
17	4000000017	2022-02-01	2023-02-01	credit card	NULL	0.15	2022-01-29
18	4000000018	2022-04-20	2023-04-20	credit card	0.50	0.15	2022-04-18
19	4000000019	2022-05-01	2022-08-01	credit card	0.08	0.15	2022-05-03
20	4000000020	2022-06-23	2022-09-23	credit card	0.10	0.15	2022-06-23
21	4000000021	2022-07-05	2022-10-05	cash	0.05	0.15	2022-07-09
22	4000000022	2022-06-22	2022-12-22	credit card	0.10	0.15	2022-06-22
23	4000000023	2022-09-28	2023-03-28	cash	0.02	0.15	2022-09-29
24	4000000024	2022-12-01	2023-06-01	credit card	0.10	0.15	2022-12-02
25	4000000025	2022-11-20	2023-11-20	cash	0.10	0.15	2022-11-21
26	4000000026	2022-07-08	2023-07-08	credit card	0.20	0.15	2022-07-09
27	4000000027	2022-09-01	2023-09-01	credit card	0.15	0.15	2022-09-03
28	4000000028	2022-09-01	2022-12-01	credit card	NULL	0.15	2022-09-04
29	4000000029	2022-07-01	2023-01-01	credit card	0.10	0.15	2022-07-01
30	4000000030	2022-07-12	2023-07-12	cash	0.45	0.15	2022-07-10

9. The Events table:

Results Messages

	EventName	Start_Date	NumberOfDays	HoursPerDays	City	HostingPlaceName
1	Biking Marathon	2022-11-15	3	4	Jeddah	Jeddah Waterfront
2	Boxing Competition	2022-03-04	2	3	Riyadh	Flagboxing
3	Diving	2022-09-07	2	2	Yanbu	Saudi Diving Center
4	Football Match	2022-12-12	1	3	Jeddah	Jawharat Alajaweed Stadium
5	Golf Match	2022-07-12	1	3	Jeddah	Groovy Golf
6	Horse Riding	2022-05-18	3	5	Jeddah	Alreem.Stabel
7	Mount Hiking	2022-06-01	1	5	Ola	Muntajae Habitas Alola
8	Roller Skating	2022-08-15	3	2	Jeddah	Ice Land
9	Running Marathon	2022-08-20	3	4	Jeddah	Jeddah Waterfront
10	Tennis Match	2022-10-03	1	3	Jeddah	Spin&Smach

10. The Membership plans table:

Results | Messages

	MembershipPlanID	PlanName	Durationn	Price
1	1000100001	Diamond	3 months	2000
2	1000100002	Diamond	6 months	2200
3	1000100003	Diamond	12 months	2400
4	2000200001	Golden	3 months	1400
5	2000200002	Golden	6 months	1600
6	2000200003	Golden	12 months	1800
7	3000300001	Platinum	3 months	800
8	3000300002	Platinum	6 months	1000
9	3000300003	Platinum	12 months	1200

11. The Supervised_By table:

Results | Messages

	EventName	supervisorID
1	Biking Marathon	2022000034
2	Biking Marathon	2022000047
3	Boxing Competition	2022000019
4	Boxing Competition	2022000020
5	Diving	2022000021
6	Diving	2022000064
7	Diving	2022000072
8	Football Match	2022000035
9	Football Match	2022000050
10	Golf Match	2022000028
11	Horse Riding	2022000003
12	Horse Riding	2022000032
13	Mount Hiking	2022000004
14	Mount Hiking	2022000055
15	Mount Hiking	2022000062
16	Roller Skating	2022000027
17	Running Marathon	2022000009
18	Running Marathon	2022000025
19	Tennis Match	2022000066
20	Tennis Match	2022000070

12. The Services_Of_Plan table:

Results | Messages

	ServicesName	PlanID
1	24/7 Access	1000100001
2	24/7 Access	1000100002
3	24/7 Access	1000100003
4	Aquatic Pool	1000100001
5	Aquatic Pool	1000100002
6	Aquatic Pool	1000100003
7	Aquatic Pool	2000200001
8	Aquatic Pool	2000200002
9	Aquatic Pool	2000200003
10	Nursery	1000100001
11	Nursery	1000100002
12	Nursery	1000100003
13	Personal trai...	1000100001
14	Personal trai...	1000100002
15	Personal trai...	1000100003
16	Personal trai...	2000200001
17	Personal trai...	2000200002
18	Personal trai...	2000200003
19	Personal trai...	3000300001
20	Personal trai...	3000300002
21	Personal trai...	3000300003
22	Personalized...	1000100001
23	Personalized...	1000100002
24	Personalized...	1000100003
25	Personalized...	2000200001
26	Personalized...	2000200002
27	Personalized...	2000200003
28	Personalized...	3000300001
29	Personalized...	3000300002
30	Personalized...	3000300003
31	Spa sessions	1000100001
32	Spa sessions	1000100002
33	Spa sessions	1000100003
34	Spa sessions	2000200001
35	Spa sessions	2000200002
36	Spa sessions	2000200003

13.The Assigned_To Table:

Results Messages

	MembershipID	ContractID	MemberID	TotalCost
1	1000100001	4000000001	1000000001	2070
2	1000100001	4000000011	1000000011	1955
3	1000100001	4000000021	1000000021	2185
4	1000100002	4000000002	1000000002	2530
5	1000100002	4000000015	1000000015	2277
6	1000100002	4000000022	1000000022	2277
7	1000100003	4000000009	1000000009	2760
8	1000100003	4000000016	1000000016	2070
9	1000100003	4000000030	1000000030	1518
10	2000200001	4000000005	1000000005	1380
11	2000200001	4000000010	1000000010	1449
12	2000200001	4000000019	1000000019	1481.2
13	2000200001	4000000028	1000000028	1380
14	2000200002	4000000003	1000000003	1840
15	2000200002	4000000014	1000000014	1288
16	2000200002	4000000023	1000000023	1803.2
17	2000200003	4000000018	1000000018	1035
18	2000200003	4000000026	1000000026	1656
19	2000200003	4000000027	1000000027	1759.5
20	3000300001	4000000004	1000000004	736
21	3000300001	4000000006	1000000006	920
22	3000300001	4000000012	1000000012	920
23	3000300001	4000000020	1000000020	828
24	3000300002	4000000007	1000000007	1035
25	3000300002	4000000013	1000000013	1035
26	3000300002	4000000024	1000000024	1035
27	3000300002	4000000029	1000000029	1035
28	3000300003	4000000008	1000000008	1242
29	3000300003	4000000017	1000000017	1380
30	3000300003	4000000025	1000000025	1242

14. The Classes table:

Results | Messages

	Dayy	Hourr	RoomNum	ClassRefNumber	Activity	InstructerID
1	Monday	08:00:00	31	6000000187	Yoga Class	2022000054
2	Monday	08:00:00	34	6000000190	Boxing Class	2022000055
3	Monday	09:00:00	41	6000000247	Yoga Class	2022000066
4	Monday	09:00:00	44	6000000250	Boxing Class	2022000067
5	Monday	10:00:00	1	6000000007	Yoga Class	2022000020
6	Monday	10:00:00	4	6000000010	Boxing Class	2022000021
7	Monday	10:00:00	6	6000000037	Yoga Class	2022000026
8	Monday	10:00:00	9	6000000040	Boxing Class	2022000028
9	Monday	10:00:00	11	6000000067	Yoga Class	2022000032
10	Monday	10:00:00	14	6000000070	Boxing Class	2022000033
11	Monday	10:00:00	36	6000000217	Yoga Class	2022000009
12	Monday	10:00:00	39	6000000220	Boxing Class	2022000060
13	Monday	11:00:00	26	6000000157	Yoga Class	2022000048
14	Monday	11:00:00	29	6000000160	Boxing Class	2022000049
15	Monday	11:00:00	46	6000000277	Yoga Class	2022000072
16	Monday	11:00:00	49	6000000280	Boxing Class	2022000073
17	Monday	12:00:00	16	6000000097	Yoga Class	2022000038
18	Monday	12:00:00	19	6000000100	Boxing Class	2022000039
19	Monday	12:00:00	21	6000000127	Yoga Class	2022000043
20	Monday	12:00:00	24	6000000130	Boxing Class	2022000044
21	Monday	14:00:00	2	6000000008	Zumba Class	2022000022
22	Monday	14:00:00	7	6000000038	Zumba Class	2022000061
23	Monday	14:00:00	12	6000000068	Zumba Class	2022000034
24	Monday	14:00:00	17	6000000098	Zumba Class	2022000040
25	Monday	14:00:00	22	6000000128	Zumba Class	2022000045
26	Monday	14:00:00	27	6000000158	Zumba Class	2022000050
27	Monday	14:00:00	32	6000000188	Zumba Class	2022000056
28	Monday	14:00:00	37	6000000218	Zumba Class	2022000062
29	Monday	14:00:00	42	6000000248	Zumba Class	2022000068
30	Monday	14:00:00	47	6000000278	Zumba Class	2022000074

Results | Messages

	Dayy	Hourr	RoomNum	ClassRefNumber	Activity	InstructerID
31	Monday	18:00:00	5	6000000009	Zumba Class	2022000022
32	Monday	18:00:00	10	6000000039	Zumba Class	2022000061
33	Monday	18:00:00	15	6000000069	Zumba Class	2022000034
34	Monday	18:00:00	20	6000000099	Zumba Class	2022000040
35	Monday	18:00:00	25	6000000129	Zumba Class	2022000045
36	Monday	18:00:00	30	6000000159	Zumba Class	2022000050
37	Monday	18:00:00	35	6000000189	Zumba Class	2022000056
38	Monday	18:00:00	40	6000000219	Zumba Class	2022000062
39	Monday	18:00:00	45	6000000249	Zumba Class	2022000068
40	Monday	18:00:00	50	6000000279	Zumba Class	2022000074
41	Satur...	08:00:00	31	6000000207	Yoga Class	2022000054
42	Satur...	08:00:00	34	6000000210	Boxing Class	2022000055
43	Satur...	09:00:00	41	6000000267	Yoga Class	2022000066
44	Satur...	09:00:00	44	6000000270	Boxing Class	2022000067
45	Satur...	10:00:00	1	6000000027	Yoga Class	2022000020
46	Satur...	10:00:00	4	6000000030	Boxing Class	2022000021
47	Satur...	10:00:00	6	6000000057	Yoga Class	2022000026
48	Satur...	10:00:00	9	6000000060	Boxing Class	2022000028
49	Satur...	10:00:00	11	6000000087	Yoga Class	2022000032
50	Satur...	10:00:00	14	6000000090	Boxing Class	2022000033
51	Satur...	10:00:00	36	6000000237	Yoga Class	2022000009
52	Satur...	10:00:00	39	6000000240	Boxing Class	2022000060
53	Satur...	11:00:00	26	6000000177	Yoga Class	2022000048
54	Satur...	11:00:00	29	6000000180	Boxing Class	2022000049
55	Satur...	11:00:00	46	6000000297	Yoga Class	2022000072
56	Satur...	11:00:00	49	6000000300	Boxing Class	2022000073
57	Satur...	12:00:00	16	6000000117	Yoga Class	2022000038
58	Satur...	12:00:00	19	6000000120	Boxing Class	2022000039
59	Satur...	12:00:00	21	6000000147	Yoga Class	2022000043
60	Satur...	12:00:00	24	6000000150	Boxing Class	2022000044

Results | Messages

	Dayy	Hourr	RoomNum	ClassRefNumber	Activity	InstructerID
61	Satur...	14:00:00	2	6000000028	Zumba Class	2022000022
62	Satur...	14:00:00	7	6000000058	Zumba Class	2022000061
63	Satur...	14:00:00	12	6000000088	Zumba Class	2022000034
64	Satur...	14:00:00	17	6000000118	Zumba Class	2022000040
65	Satur...	14:00:00	22	6000000148	Zumba Class	2022000045
66	Satur...	14:00:00	27	6000000178	Zumba Class	2022000050
67	Satur...	14:00:00	32	6000000208	Zumba Class	2022000056
68	Satur...	14:00:00	37	6000000238	Zumba Class	2022000062
69	Satur...	14:00:00	42	6000000268	Zumba Class	2022000068
70	Satur...	14:00:00	47	6000000298	Zumba Class	2022000074
71	Satur...	18:00:00	5	6000000029	Zumba Class	2022000022
72	Satur...	18:00:00	10	6000000059	Zumba Class	2022000061
73	Satur...	18:00:00	15	6000000089	Zumba Class	2022000034
74	Satur...	18:00:00	20	6000000119	Zumba Class	2022000040
75	Satur...	18:00:00	25	6000000149	Zumba Class	2022000045
76	Satur...	18:00:00	30	6000000179	Zumba Class	2022000050
77	Satur...	18:00:00	35	6000000209	Zumba Class	2022000056
78	Satur...	18:00:00	40	6000000239	Zumba Class	2022000062
79	Satur...	18:00:00	45	6000000269	Zumba Class	2022000068
80	Satur...	18:00:00	50	6000000299	Zumba Class	2022000074
81	Sunday	09:00:00	1	6000000001	Yoga Class	2022000020
82	Sunday	09:00:00	11	6000000061	Yoga Class	2022000032
83	Sunday	09:00:00	46	6000000271	Yoga Class	2022000072
84	Sunday	09:00:00	47	6000000272	Spinning S...	2022000073
85	Sunday	10:00:00	2	6000000004	Spinning S...	2022000022
86	Sunday	10:00:00	6	6000000031	Yoga Class	2022000026
87	Sunday	10:00:00	7	6000000034	Spinning S...	2022000061
88	Sunday	10:00:00	12	6000000064	Spinning S...	2022000034
89	Sunday	10:00:00	26	6000000151	Yoga Class	2022000048
90	Sunday	10:00:00	27	6000000152	Spinning S...	2022000049

Results | Messages

	Dayy	Hourr	RoomNum	ClassRefNumber	Activity	InstructerID
91	Sunday	11:00:00	2	6000000002	Spinning S...	2022000021
92	Sunday	11:00:00	7	6000000032	Spinning S...	2022000027
93	Sunday	11:00:00	12	6000000062	Spinning S...	2022000033
94	Sunday	11:00:00	16	6000000091	Yoga Class	2022000038
95	Sunday	11:00:00	17	6000000092	Spinning S...	2022000039
96	Sunday	11:00:00	31	6000000181	Yoga Class	2022000054
97	Sunday	11:00:00	32	6000000182	Spinning S...	2022000055
98	Sunday	11:00:00	36	6000000211	Yoga Class	2022000009
99	Sunday	11:00:00	37	6000000212	Spinning S...	2022000060
100	Sunday	11:00:00	41	6000000241	Yoga Class	2022000066
101	Sunday	11:00:00	42	6000000242	Spinning S...	2022000067
102	Sunday	13:00:00	21	6000000121	Yoga Class	2022000043
103	Sunday	13:00:00	22	6000000122	Spinning S...	2022000044
104	Sunday	14:00:00	17	6000000094	Spinning S...	2022000040
105	Sunday	14:00:00	22	6000000124	Spinning S...	2022000045
106	Sunday	14:00:00	27	6000000154	Spinning S...	2022000050
107	Sunday	14:00:00	32	6000000184	Spinning S...	2022000056
108	Sunday	14:00:00	37	6000000214	Spinning S...	2022000062
109	Sunday	14:00:00	42	6000000244	Spinning S...	2022000068
110	Sunday	14:00:00	47	6000000274	Spinning S...	2022000074
111	Sunday	15:00:00	3	6000000005	Aerobics Cl...	2022000021
112	Sunday	15:00:00	8	6000000035	Aerobics Cl...	2022000027
113	Sunday	15:00:00	13	6000000065	Aerobics Cl...	2022000033
114	Sunday	15:00:00	18	6000000095	Aerobics Cl...	2022000039
115	Sunday	15:00:00	23	6000000125	Aerobics Cl...	2022000044
116	Sunday	15:00:00	28	6000000155	Aerobics Cl...	2022000049
117	Sunday	15:00:00	33	6000000185	Aerobics Cl...	2022000055
118	Sunday	15:00:00	38	6000000215	Aerobics Cl...	2022000060
119	Sunday	15:00:00	43	6000000245	Aerobics Cl...	2022000067
120	Sunday	15:00:00	48	6000000275	Aerobics Cl...	2022000073

Results Messages

	Dayy	Hourr	RoomNum	ClassRefNumber	Activity	InstructerID
121	Sunday	16:00:00	4	6000000003	Cardio	2022000022
122	Sunday	16:00:00	9	6000000033	Cardio	2022000028
123	Sunday	16:00:00	14	6000000063	Cardio	2022000034
124	Sunday	16:00:00	19	6000000093	Cardio	2022000040
125	Sunday	16:00:00	24	6000000123	Cardio	2022000045
126	Sunday	16:00:00	29	6000000153	Cardio	2022000050
127	Sunday	16:00:00	34	6000000183	Cardio	2022000056
128	Sunday	16:00:00	39	6000000213	Cardio	2022000062
129	Sunday	16:00:00	44	6000000243	Cardio	2022000068
130	Sunday	16:00:00	49	6000000273	Cardio	2022000074
131	Sunday	19:00:00	29	6000000156	Cardio	2022000048
132	Sunday	19:30:00	4	6000000006	Cardio	2022000020
133	Sunday	19:30:00	9	6000000036	Cardio	2022000028
134	Sunday	20:00:00	44	6000000246	Cardio	2022000066
135	Sunday	22:00:00	14	6000000066	Cardio	2022000032
136	Sunday	22:00:00	19	6000000096	Cardio	2022000038
137	Sunday	22:00:00	24	6000000126	Cardio	2022000043
138	Sunday	22:00:00	34	6000000186	Cardio	2022000054
139	Sunday	22:00:00	49	6000000276	Cardio	2022000072
140	Sunday	22:30:00	39	6000000216	Cardio	2022000009
141	Thurs...	09:00:00	1	6000000021	Yoga Class	2022000020
142	Thurs...	09:00:00	11	6000000081	Yoga Class	2022000032
143	Thurs...	09:00:00	46	6000000291	Yoga Class	2022000072
144	Thurs...	09:00:00	47	6000000292	Spinning S...	2022000073
145	Thurs...	10:00:00	2	6000000024	Spinning S...	2022000022
146	Thurs...	10:00:00	6	6000000051	Yoga Class	2022000026
147	Thurs...	10:00:00	7	6000000054	Spinning S...	2022000061
148	Thurs...	10:00:00	12	6000000084	Spinning S...	2022000034
149	Thurs...	10:00:00	26	6000000171	Yoga Class	2022000048
150	Thurs...	10:00:00	27	6000000172	Spinning S...	2022000049

Results Messages

	Dayy	Hourr	RoomNum	ClassRefNumber	Activity	InstructerID
151	Thurs...	11:00:00	2	6000000022	Spinning S...	2022000021
152	Thurs...	11:00:00	7	6000000052	Spinning S...	2022000027
153	Thurs...	11:00:00	12	6000000082	Spinning S...	2022000033
154	Thurs...	11:00:00	16	6000000111	Yoga Class	2022000038
155	Thurs...	11:00:00	17	6000000112	Spinning S...	2022000039
156	Thurs...	11:00:00	31	6000000201	Yoga Class	2022000054
157	Thurs...	11:00:00	32	6000000202	Spinning S...	2022000055
158	Thurs...	11:00:00	36	6000000231	Yoga Class	2022000009
159	Thurs...	11:00:00	37	6000000232	Spinning S...	2022000060
160	Thurs...	11:00:00	41	6000000261	Yoga Class	2022000066
161	Thurs...	11:00:00	42	6000000262	Spinning S...	2022000067
162	Thurs...	13:00:00	21	6000000141	Yoga Class	2022000043
163	Thurs...	13:00:00	22	6000000142	Spinning S...	2022000044
164	Thurs...	14:00:00	17	6000000114	Spinning S...	2022000040
165	Thurs...	14:00:00	22	6000000144	Spinning S...	2022000045
166	Thurs...	14:00:00	27	6000000174	Spinning S...	2022000050
167	Thurs...	14:00:00	32	6000000204	Spinning S...	2022000056
168	Thurs...	14:00:00	37	6000000234	Spinning S...	2022000062
169	Thurs...	14:00:00	42	6000000264	Spinning S...	2022000068
170	Thurs...	14:00:00	47	6000000294	Spinning S...	2022000074
171	Thurs...	15:00:00	3	6000000025	Aerobics Cl...	2022000021
172	Thurs...	15:00:00	8	6000000055	Aerobics Cl...	2022000027
173	Thurs...	15:00:00	13	6000000085	Aerobics Cl...	2022000033
174	Thurs...	15:00:00	18	6000000115	Aerobics Cl...	2022000039
175	Thurs...	15:00:00	23	6000000145	Aerobics Cl...	2022000044
176	Thurs...	15:00:00	28	6000000175	Aerobics Cl...	2022000049
177	Thurs...	15:00:00	33	6000000205	Aerobics Cl...	2022000055
178	Thurs...	15:00:00	38	6000000235	Aerobics Cl...	2022000060
179	Thurs...	15:00:00	43	6000000265	Aerobics Cl...	2022000067
180	Thurs...	15:00:00	48	6000000295	Aerobics Cl...	2022000073

Results Messages

	Dayy	Hourr	RoomNum	ClassRefNumber	Activity	InstructerID
181	Thurs...	16:00:00	4	6000000023	Cardio	2022000022
182	Thurs...	16:00:00	9	6000000053	Cardio	2022000028
183	Thurs...	16:00:00	14	6000000083	Cardio	2022000034
184	Thurs...	16:00:00	19	6000000113	Cardio	2022000040
185	Thurs...	16:00:00	24	6000000143	Cardio	2022000045
186	Thurs...	16:00:00	29	6000000173	Cardio	2022000050
187	Thurs...	16:00:00	34	6000000203	Cardio	2022000056
188	Thurs...	16:00:00	39	6000000233	Cardio	2022000062
189	Thurs...	16:00:00	44	6000000263	Cardio	2022000068
190	Thurs...	16:00:00	49	6000000293	Cardio	2022000074
191	Thurs...	19:00:00	4	6000000026	Cardio	2022000020
192	Thurs...	19:00:00	9	6000000056	Cardio	2022000028
193	Thurs...	19:00:00	29	6000000176	Cardio	2022000048
194	Thurs...	20:00:00	44	6000000266	Cardio	2022000066
195	Thurs...	22:00:00	14	6000000086	Cardio	2022000032
196	Thurs...	22:00:00	19	6000000116	Cardio	2022000038
197	Thurs...	22:00:00	24	6000000146	Cardio	2022000043
198	Thurs...	22:00:00	34	6000000206	Cardio	2022000054
199	Thurs...	22:00:00	49	6000000296	Cardio	2022000072
200	Thurs...	22:30:00	39	6000000236	Cardio	2022000009
201	Tuesd...	09:00:00	1	6000000011	Yoga Class	2022000020
202	Tuesd...	09:00:00	11	6000000071	Yoga Class	2022000032
203	Tuesd...	09:00:00	46	6000000281	Yoga Class	2022000072
204	Tuesd...	09:00:00	47	6000000282	Spinning S...	2022000073
205	Tuesd...	10:00:00	2	6000000014	Spinning S...	2022000022
206	Tuesd...	10:00:00	6	6000000041	Yoga Class	2022000026
207	Tuesd...	10:00:00	7	6000000044	Spinning S...	2022000061
208	Tuesd...	10:00:00	12	6000000074	Spinning S...	2022000034
209	Tuesd...	10:00:00	26	6000000161	Yoga Class	2022000048
210	Tuesd...	10:00:00	27	6000000162	Spinning S...	2022000049

Results | Messages

	Dayy	Hourr	RoomNum	ClassRefNumber	Activity	InstructerID
211	Tuesd...	11:00:00	2	6000000012	Spinning S...	2022000021
212	Tuesd...	11:00:00	7	6000000042	Spinning S...	2022000027
213	Tuesd...	11:00:00	12	6000000072	Spinning S...	2022000033
214	Tuesd...	11:00:00	16	6000000101	Yoga Class	2022000038
215	Tuesd...	11:00:00	17	6000000102	Spinning S...	2022000039
216	Tuesd...	11:00:00	31	6000000191	Yoga Class	2022000054
217	Tuesd...	11:00:00	32	6000000192	Spinning S...	2022000055
218	Tuesd...	11:00:00	36	6000000221	Yoga Class	2022000009
219	Tuesd...	11:00:00	37	6000000222	Spinning S...	2022000060
220	Tuesd...	11:00:00	41	6000000251	Yoga Class	2022000066
221	Tuesd...	11:00:00	42	6000000252	Spinning S...	2022000067
222	Tuesd...	13:00:00	21	6000000131	Yoga Class	2022000043
223	Tuesd...	13:00:00	22	6000000132	Spinning S...	2022000044
224	Tuesd...	14:00:00	17	6000000104	Spinning S...	2022000040
225	Tuesd...	14:00:00	22	6000000134	Spinning S...	2022000045
226	Tuesd...	14:00:00	27	6000000164	Spinning S...	2022000050
227	Tuesd...	14:00:00	32	6000000194	Spinning S...	2022000056
228	Tuesd...	14:00:00	37	6000000224	Spinning S...	2022000062
229	Tuesd...	14:00:00	42	6000000254	Spinning S...	2022000068
230	Tuesd...	14:00:00	47	6000000284	Spinning S...	2022000074
231	Tuesd...	15:00:00	3	6000000015	Aerobics Cl...	2022000021
232	Tuesd...	15:00:00	8	6000000045	Aerobics Cl...	2022000027
233	Tuesd...	15:00:00	13	6000000075	Aerobics Cl...	2022000033
234	Tuesd...	15:00:00	18	6000000105	Aerobics Cl...	2022000039
235	Tuesd...	15:00:00	23	6000000135	Aerobics Cl...	2022000044
236	Tuesd...	15:00:00	28	6000000165	Aerobics Cl...	2022000049
237	Tuesd...	15:00:00	33	6000000195	Aerobics Cl...	2022000055
238	Tuesd...	15:00:00	38	6000000225	Aerobics Cl...	2022000060
239	Tuesd...	15:00:00	43	6000000255	Aerobics Cl...	2022000067
240	Tuesd...	15:00:00	48	6000000285	Aerobics Cl...	2022000073

Results | Messages

	Dayy	Hourr	RoomNum	ClassRefNumber	Activity	InstructerID
241	Tuesd...	16:00:00	4	6000000013	Cardio	2022000022
242	Tuesd...	16:00:00	9	6000000043	Cardio	2022000028
243	Tuesd...	16:00:00	14	6000000073	Cardio	2022000034
244	Tuesd...	16:00:00	19	6000000103	Cardio	2022000040
245	Tuesd...	16:00:00	24	6000000133	Cardio	2022000045
246	Tuesd...	16:00:00	29	6000000163	Cardio	2022000050
247	Tuesd...	16:00:00	34	6000000193	Cardio	2022000056
248	Tuesd...	16:00:00	39	6000000223	Cardio	2022000062
249	Tuesd...	16:00:00	44	6000000253	Cardio	2022000068
250	Tuesd...	16:00:00	49	6000000283	Cardio	2022000074
251	Tuesd...	19:00:00	4	6000000016	Cardio	2022000020
252	Tuesd...	19:00:00	9	6000000046	Cardio	2022000028
253	Tuesd...	19:00:00	29	6000000166	Cardio	2022000048
254	Tuesd...	20:00:00	44	6000000256	Cardio	2022000066
255	Tuesd...	22:00:00	14	6000000076	Cardio	2022000032
256	Tuesd...	22:00:00	19	6000000106	Cardio	2022000038
257	Tuesd...	22:00:00	24	6000000136	Cardio	2022000043
258	Tuesd...	22:00:00	34	6000000196	Cardio	2022000054
259	Tuesd...	22:00:00	49	6000000286	Cardio	2022000072
260	Tuesd...	22:30:00	39	6000000226	Cardio	2022000009
261	Wedn...	08:00:00	31	6000000197	Yoga Class	2022000054
262	Wedn...	08:00:00	34	6000000200	Boxing Class	2022000054
263	Wedn...	09:00:00	41	6000000257	Yoga Class	2022000066
264	Wedn...	09:00:00	44	6000000260	Boxing Class	2022000067
265	Wedn...	10:00:00	1	6000000017	Yoga Class	2022000020
266	Wedn...	10:00:00	4	6000000020	Boxing Class	2022000021
267	Wedn...	10:00:00	6	6000000047	Yoga Class	2022000026
268	Wedn...	10:00:00	9	6000000050	Boxing Class	2022000028
269	Wedn...	10:00:00	11	6000000077	Yoga Class	2022000032
270	Wedn...	10:00:00	14	6000000080	Boxing Class	2022000033

271	Wedn...	10:00:00	36	6000000227	Yoga Class	2022000009
272	Wedn...	10:00:00	39	6000000230	Boxing Class	2022000009
273	Wedn...	11:00:00	26	6000000167	Yoga Class	2022000048
274	Wedn...	11:00:00	29	6000000170	Boxing Class	2022000049
275	Wedn...	11:00:00	46	6000000287	Yoga Class	2022000072
276	Wedn...	11:00:00	49	6000000290	Boxing Class	2022000073
277	Wedn...	12:00:00	16	6000000107	Yoga Class	2022000038
278	Wedn...	12:00:00	19	6000000110	Boxing Class	2022000039
279	Wedn...	12:00:00	21	6000000137	Yoga Class	2022000043
280	Wedn...	12:00:00	24	6000000140	Boxing Class	2022000044
281	Wedn...	14:00:00	2	6000000018	Zumba Class	2022000022
282	Wedn...	14:00:00	7	6000000048	Zumba Class	2022000061
283	Wedn...	14:00:00	12	6000000078	Zumba Class	2022000034
284	Wedn...	14:00:00	17	6000000108	Zumba Class	2022000040
285	Wedn...	14:00:00	22	6000000138	Zumba Class	2022000045
286	Wedn...	14:00:00	27	6000000168	Zumba Class	2022000050
287	Wedn...	14:00:00	32	6000000198	Zumba Class	2022000056
288	Wedn...	14:00:00	37	6000000228	Zumba Class	2022000060
289	Wedn...	14:00:00	42	6000000258	Zumba Class	2022000068
290	Wedn...	14:00:00	47	6000000288	Zumba Class	2022000074
291	Wedn...	18:00:00	5	6000000019	Zumba Class	2022000022
292	Wedn...	18:00:00	10	6000000049	Zumba Class	2022000061
293	Wedn...	18:00:00	15	6000000079	Zumba Class	2022000034
294	Wedn...	18:00:00	20	6000000109	Zumba Class	2022000040
295	Wedn...	18:00:00	25	6000000139	Zumba Class	2022000045
296	Wedn...	18:00:00	30	6000000169	Zumba Class	2022000050
297	Wedn...	18:00:00	35	6000000199	Zumba Class	2022000056
298	Wedn...	18:00:00	40	6000000229	Zumba Class	2022000060
299	Wedn...	18:00:00	45	6000000259	Zumba Class	2022000068
300	Wedn...	18:00:00	50	6000000289	Zumba Class	2022000074