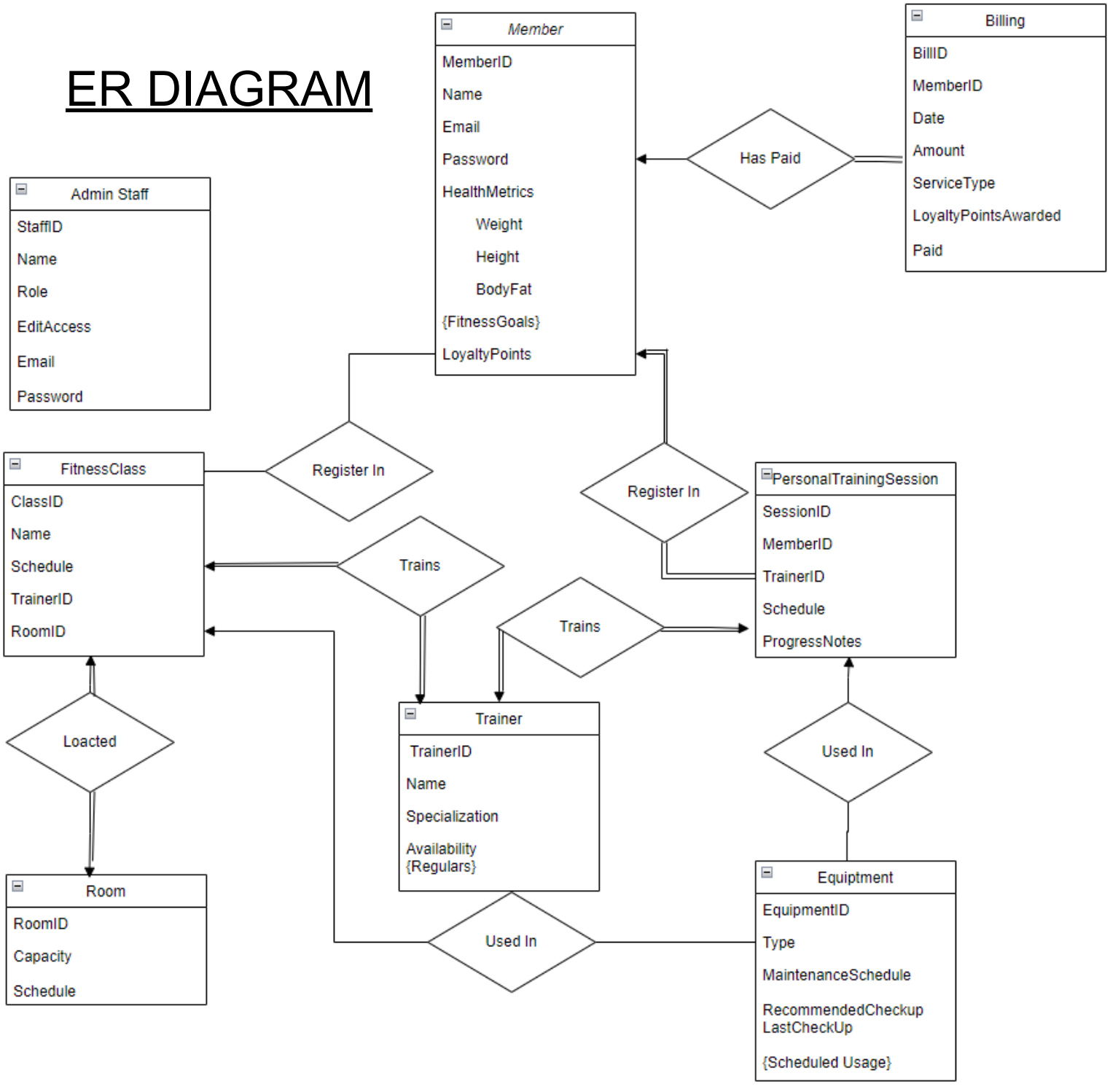
GROUP PROJECT COMP 3005

Sameer Ahmadzai
Luka Grahek
Mc·hammad Steitieh

We are a group of highly motivated students looking forward to careers in computer science. We would like to thank the TAs as well as Prof Abdelghny in person. You have provided us with a wide and solid base of understanding regarding databases. The teaching methods provided throughout the semester will allow us to retain this information in our coops and develop our careers for the better. Thank you for the positive impact made on us, we wish all the best for you in the future.

ER DIAGRAM



Entities and Attributes (in reference to the ER diagram):

Member:
Attributes: MemberID, Name, Email, Password, HealthMetrics (Weight, Height, BodyFat), FitnessGoals, LoyaltyPoints.
Assumption: Each member has a unique ID and can have multiple fitness goals. Health metrics are assumed to be self-reported by the member.

Admin Staff:
Attributes: StaffID, Name, Role, EditAccess, Email, Password.
Assumption: Admin staff have different roles and permissions within the system, indicated by the 'EditAccess' attribute.

Fitness Class:
Attributes: ClassID, Name, Schedule, TrainerID, RoomID.
Assumption: Classes are scheduled in specific rooms and are assigned to trainers.

Trainer:
Attributes: TrainerID, Name, Specialization, Availability.
Assumption: Trainers have specialized areas of training and have a set availability schedule for training/class sessions.

Personal Training Session:
Attributes: SessionID, MemberID, TrainerID, Schedule, ProgressNotes.
Assumption: Each session is unique and associated with one member and one trainer. Progress notes are recorded after each session.

Room:
Attributes: RoomID, Capacity, Schedule.
Assumption: Rooms are scheduled for fitness classes and personal training sessions and have a capacity limit.

Equipment:
Attributes: EquipmentID, Type, MaintenanceSchedule, RecommendedCheckup, LastCheckUp.
Assumption: Equipment is regularly maintained, and there is a schedule for recommended and last checkups.

Billing:
Attributes: BillID, MemberID, Date, Amount, ServiceType, LoyaltyPointsAwarded, Paid.
Assumption: Bills are generated for various services, including membership fees and personal training sessions. Loyalty points are awarded upon billing.

Relationships:

Register In:

Members can register in fitness classes and personal training sessions.
Assumption: Registration is required to attend a class or session, and this relationship captures the many-to-many associations.

Trains:
A trainer can train in many fitness classes and personal training sessions.
Assumption: Trainers may be associated with more than one class or session.

Located:
Fitness classes are located in rooms.
Assumption: A room can have multiple classes scheduled, but not at the same time. One room per event.

Used In:
Equipment is used in personal training sessions.
Assumption: The model assumes a direct relationship between equipment and sessions, to track the equipment usage.

Has Paid:
Members have a payment history with the club.
Assumption: The 'Has Paid' relationship tracks all financial transactions a member has with the club.

Normalization format:

PK: Primary Key
1NF: tables created to satisfy the requirement. There should not be multivalued or nested fields in a relation
2NF: tables created to satisfy the requirement. There should not be fields that are fully functionally dependent on fields that are a subset of the primary keys
3NF: tables created to satisfy the requirement. There should not be non-primary key fields fully functionally dependent on non-primary keys.
Other: tables created to satisfy logical or functional requirements. Often will be used for composite fields.
NO CONFLICT: No tables were created to satisfy this requirement based on the basic reading of the ER model, i.e. fields in the ER model would not conflict with the requirement.
*: an assumption made that relates to the choices behind how to satisfy the requirement. Assumptions will typically be used to justify decomposing a table further.
←: the name for the table that has been created

Each heading will relate to tables created for an entity. Relationships between entities will be "absorbed" by an entity so there is no specific heading for them, although they will be labelled.

Member:

Fields: MemberID, Name, Email, Password, FitnessGoals, Password, Weight, Height, BodyFatPercentage, LoyaltyPoints, Password, LoyaltyPoints

Dependencies:
PK: MemberID
MemberID → Name
MemberID → Email
MemberID → FitnessGoals
MemberID → Password
MemberID → Weight
MemberID → Height
MemberID → BodyFatPercentage
MemberID → LoyaltyPoints
Email → Password
Email → LoyaltyPoints

1NF: MemberID, FitnessGoal ← FitnessGoals
MemberID, Email, Password, Weight, Height, BodyFatPercentage, LoyaltyPoints← Member
FitnessGoals is multivalued as a member can have more than one fitness goal.

2NF: NO CONFLICT

3NF: Email, Password, LoyaltyPoints ← MemberEmailInformation

MemberID, Email, Name, Weight, Height, BodyFatPercentage ← MemberHealthMetrics
MemberID, FitnessGoal ← MemberFitnessGoals

Since (MemberID → Email and Email → Password) AND (MemberID → Email and Email → LoyaltyPoints) we need to decompose to a new table with email as the PK and Password and LoyaltyPoints as fields.

*: multiple accounts can share an email and loyalty points.
LoyaltyPoints can be shared between members who use the same email.

Other: MemberID, Weight, Height, BodyFatPercentage ←MemberHealthMetrics
MemberID, Email, Name ←Member
Email, Password, LoyaltyPoints ←MemberAccountInformation
MemberID, FitnessGoal ←MemberFitnessGoals

This is not strictly multivalued. But this separation is made to make the code more compartmentalized (If a GUI was to be made)

FitnessClass:

Fields: Class ID, ClassName, ScheduledTime, TrainerID, RoomID, MembersAttending

Dependencies :
PK: ClassID
ClassID → ClassName
ClassID → Schedule
ClassID → TrainerID
ClassID → RoomID
ClassID → MembersAttending
{ScheduledTime, TrainerID} → ClassID
{ScheduledTime, TrainerID} → ClassName
{ScheduledTime, TrainerID} → MembersAttending
{ScheduledTime, TrainerID} → RoomID

1NF: ClassID, MemberID ←FitnessClassSchedule
ClassID, ClassName, Schedule, TrainerID, RoomID
MembersAttending is a multivalued field so it has been decomposed into a new table

2NF: NO CONFLICT

3NF: ClassID, ScheduledTime ←FitnessClassSchedule
ClassID, MemberID ←FitnessClassAttendingMembers
ClassID, ClassName, TrainerID, RoomID ← FitnessClass
ClassID, ClassName and Members attending are fully functionally dependent on the combination of ScheduledTime and TrainerID. Decomposing by separating ScheduledTime from TrainerID makes the tables meet the criteria

*: A Trainer can teach only one class at the same time.

Billing:
Fields: BillingID, Paid, MemberID, Date, Price, ServiceType, PointsAwarded
Dependencies:
PK: BillingID
BillingID → Paid
BillingID → MemberID
BillingID → Date
BillingID → Price
BillingID → ServiceType
BillingID → PointsAwarded
ServiceType → PointsAwarded
ServiceType → Price

1NF: NO CONFLICT
*: Assuming that there is a bill for each service provided, no periodized bills with all services in a period
.
2NF: NO CONFLICT

3NF: ServiceType, Price, PointsAwarded ← ServiceBillingInfo
BillingID, MemberID, Date, ServiceType ←BillingInfo
(BillingID → ServiceType and ServiceType → Price) AND (BillingID → ServiceType and ServiceType → PointsAwarded) hold. Decomposing Servicetype into a new table with Price and Points Awarded makes the table meet the 3NF requirement.
*: Assuming that the price and points awarded are the same for every service
Assuming that the member can have more

Room:
Fields: RoomID, Capacity

Dependencies:
RoomID → Capacity

1NF: NO CONFLICT

2NF: NO CONFLICT

3NF: NO CONFLICT

AdminStaff:
Fields: StaffID, Name, Role, EditAccess, Email, Password

Dependencies:
StaffID → Name
StaffID → Role
StaffID → EditAccess
StaffID → Email
StaffID → Password
Email → Password
Email → Role
Email → EditAccess
Role → EditAccess

1NF: NO CONFLICT

2NF: NO CONFLICT

3NF: Email, Password, Role ← AdminLoginInfo
Role, EditAccess ← Admin Role
StaffID, Name, Email ← AdminInfo
Since (StaffID → Email and Email → Password) AND (StaffID → Email and Email → Role) AND (StaffID → Email and Email EditAccess) we need to decompose the table into a table with email as the PK. Since Email → Role and Role → EditAccess, this can be resolved by decomposing further into a table with role and edit access.
*: We assume that multiple admin users can share an email

Trainer:
Fields: TrainerID, Name, Specialization, Availability, Regulars

Dependencies:
TrainerID → Regulars
TrainerID → Name
TrainerID → Specialization
TrainerID → Availability
TrainerID → Regulars

1NF: TrainerID, MemberID ← TrainerRegularClients
TrainerID, Name, Specialization, Availability ← Trainer
Regulars is multivalued so we decompose the table adding a new table with a row for each regular per trainer.

2NF: NO CONFLICT

3NF: NO CONFLICT

PersonalTrainingSession:
Fields: SessionID, MemberID, TrainerID, ScheduledTime, ProgressNotes

Dependencies:
SessionID → MemberID
SessionID → TrainerID
SessionID → ScheduledTime
SessionID → ProgressNotes
{TrainerID, ScheduledTime} → SessionID
{TrainerID, ScheduledTime} → MemberID
{TrainerID, ScheduledTime} → ProgressNotes
{MemberID, ScheduledTime} → SessionID
{MemberID, ScheduledTime} → TrainerID
{MemberID, ScheduledTime} → ProgressNotes

1NF: NO CONFLICT
2NF: NO CONFLICT
3NF: SessionID, MemberID, TrainerID ←PersonalTrainingSession
SessionID, ProgressNotes, BillingID, ScheduledTime ←PersonalTrainingInfo
Since the combination of either TrainerID or MemberID and ScheduledTime can be used as a unique key, we decompose the tables further into two tables

Equipment:
Fields: EquipmentID, Type, MaintenanceSchedule, RecommendedCheckup, ScheduledUsage

Dependencies:
EquipmentID → Type
EquipmentID → MaintenanceSchedule
EpuipmentID → ReccomendedCheckup
EquipmentID → ScheduledUsage
Type → ReccomendedCheckup

1NF: EquipmentID, UsageTime ← EquipmentUsageSchedule
EquipmentID, Type, MaintenanceSchedule, ReccomendedCheckup, ScheduledUsage

2NF: NO CONFLICT

3NF: EquipmentID, Type, Maintenance Schedule, LastCheckUp ← EquipmentInformation
Type, ReccomendedCheckup ← EquipmentTypeInformation

EquipmentID, UsageTime← EquipmentUsageSchedule

Since EquipmentID → Type and Type → RecommendedCheckup holds. We decompose the table into a table with Type and ReccomendedCheckUp with Type as the PK

*: We assume that the same type of equipment requires the same amount of periodic checkups. We assume that a check-up is scheduled on the day of the current check-up.

DATABASE SCHEMA