**GYMLIFE**

1. **Requirements:**

* Owner of gym club wants to have a management system to keep track of everything.
* List of staffs with their necessary information. Their work schedule (date and time), role, wage ($/hour)
* List of equipment available in the gym, their serials, brand, manufactures, cost, purchased date.
* List of members of the gym with their necessary information. Their join date, membership types, status, expiration date, contract. Last visit date and time. Transaction history (payment to their membership). Remaining owed balance, checking/saving bank account, routine number, bank name.
* List of trainers with their necessary information. Sign up date, contract, salary
* Member access log every day, monthly, year
* Membership types with price
* New members sign up, date, type.
* ROLE: OWNER, MANAGER, EMPLOYEE, TRAINER, GYMMER

1. **Design:**

**2.1 Conceptual design**

**Initial Views**

* Member View: Name, address, phone number, date of birth, gender, role, first day sign up, membership type, bank account info (bank name, account number, routine number), remaining balance, expiration date, last visit date, last visit time, transaction history, active status, trainer
* Employee View: Name, address, phone number, date of birth, gender, role, first day of work, wage ($/hour), active status, bank account info, transaction
* Trainer View: Name, address, phone number, date of birth, gender, role, first day of work, wage ($/hour), active status, bank account info, transaction, students
* Work schedule View: Work day, staff name, time
* Membership View: type name, price
* Equipment View: machine name, serial numbers, brand, target type, manufacture name, cost
* Manufacture View: manufacture name, address, phone number, email, machines imported
* Access log View: date, member name, time access in, time log out

**Business rules**

* Each member (including owner, manager, employee, trainer, gymmer) has an account stored in Member table with all their basis information
* Each member has a bank account that is stored in Bank Account table. Owner will use his business bank account for receiving money from gymmer, paying wages to staffs, and equipment costs to manufactures.
* Any transaction that happens between 2 parties will be kept tracked in the Transaction table.
* Each member will have a corresponding role with their wages set to a certain amount if the member is a staff. All roles that are tracked in this table including: OWNER, MANAGER, TRAINER, EMPLOYEE, GYMMER

**Application flows**

**…**

* 1. **Logical Design**

**Initial entities & attributes:**

* Member: Name, address, phone number, date of birth, gender, email, role, membership type, membership price, first day sign up, expiration date, active status, last visit date, last visit time, trainer, bank name, account number, routine number, remaining balance, transaction history,
* Employee: Name, address, phone number, date of birth, gender, email, role, first day of work, active status, wage ($/hour), bank name, account number, routine number, transaction history,
* Trainer: Name, address, phone number, date of birth, gender, email, role, first day of work, active status, wage ($/hour), bank account number, routine number, transaction history, students
* Work schedule: Work day, staff name, time
* Equipment: machine name, serial numbers, brand, target type, manufacture name, cost
* Manufacture: manufacture name, address, phone number, email, equipment imported from, bank name, account number, routine number, remaining balance owed, transaction history

Since member, employee, and trainer have most of attributes that are similar, we can combine them all together as one entity called Member and distinct them by role including gymmer, employee, trainer. In addition, we will add more role including owner and manager. We also added attribute named password to Member relation, so it would be used for the application security.

The initial entities and attributes are shortened to the following version below. In addition, all repeating groups/attributes are also identified and marked inside parentheses.

* Member: Name, address, phone number, date of birth, username, password, email, gender, (role, wage), (membership type, membership price), first day sign up, expiration date, active status, (last visited date, last visit time), (bank name, account number, routine number, remaining balance owed, transaction history), (trainer, student)
* Work schedule: Work day, staff name, time
* Equipment: machine name, serial numbers, target type, manufacture name, cost
* Manufacture: manufacture name, address, phone number, email, (equipment imported from), (bank name, account number, routine number, remaining balance owed, transaction history)

**Normalization:**

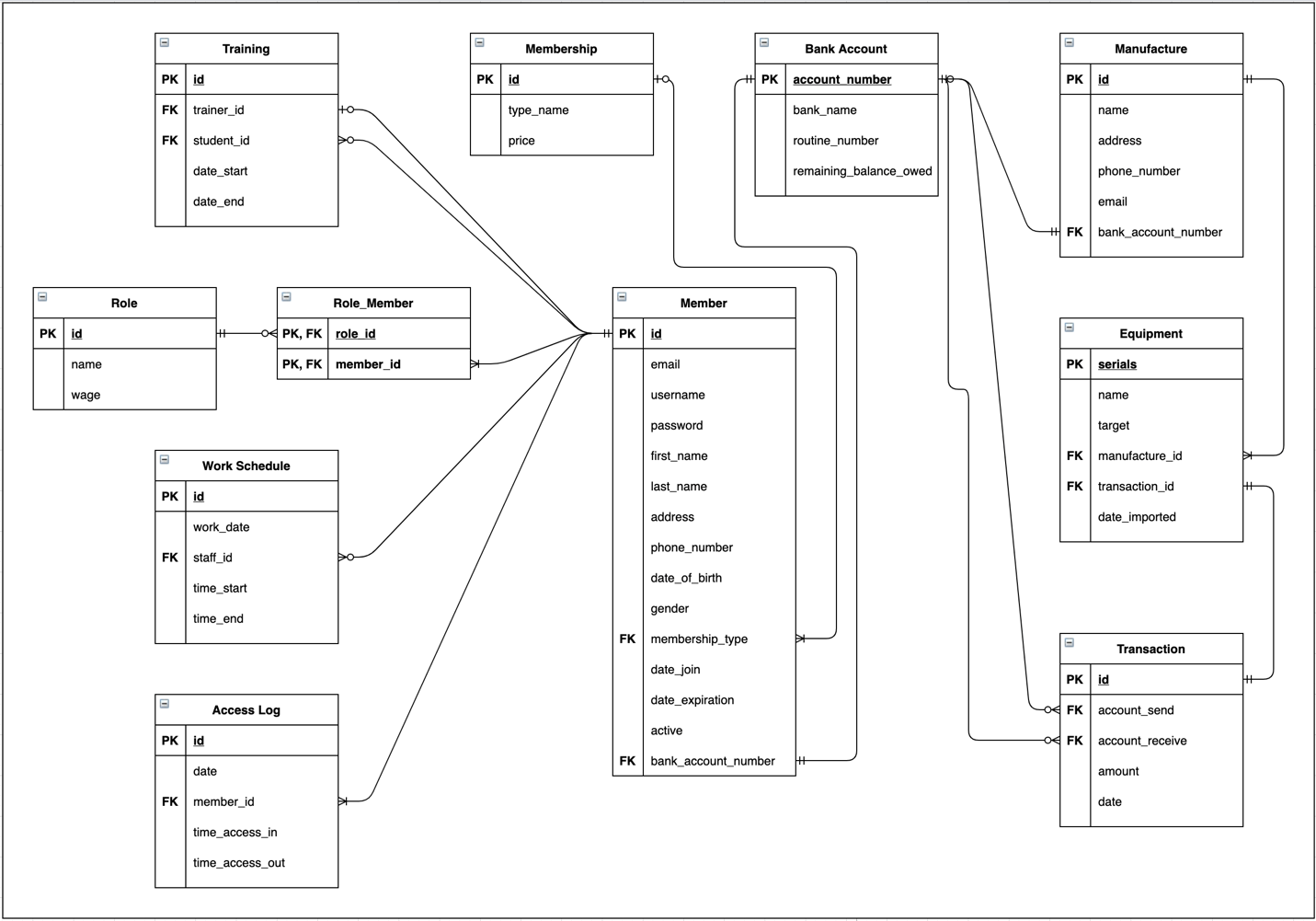
**1NF: Assigning primary key and move repeating groups to new relations.**

**2NF: Move all non-key attributes that are not fully functional dependent on primary key to new relations**

**3NF: Move all non-key attributes that are transitively dependent to new relations**

* Member: id (PK), email, username, password, first\_name, last\_name, address, phone\_number, date\_of\_birth, gender, role\_id (FK), membership\_id (FK), date\_join, date\_expiration, active, bank\_account\_number (FK)
* Bank Account: account\_number (PK), bank\_name, routine\_number, remaining\_balance\_owed
* Transaction: id (PK), account\_send (FK), account\_receive (FK), amount, date
* Role: id (PK), name, wage
* Training: id (PK), trainer\_id (FK), student\_id (FK), date\_start, date\_end
* Work schedule: id (PK), work\_date, staff\_id (FK), time\_start, time\_end
* Membership: id (PK), type\_name, price
* Equipment: serials (PK), name, target, manufacture\_id (FK), transaction\_id (FK), date\_imported
* Manufacture: id (PK), name, address, phone\_number, email, bank\_account\_number (FK)
* Access log: id (PK), date, member\_id (FK), time\_access\_in, time\_access\_out

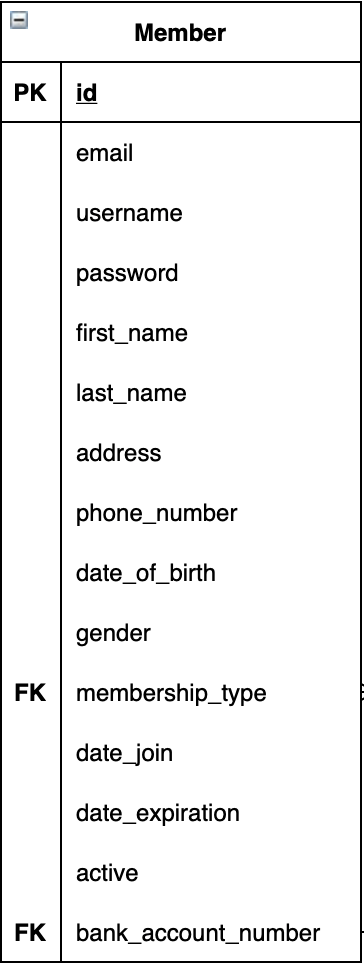
**Entity relationship diagram (ERD)**

****

* 1. **Physical Design**

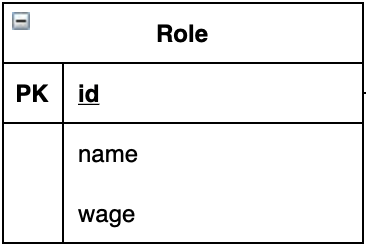
Mapping relations to tables, and attributes to columns.

Note: Integer display width in MySQL is deprecated and will be removed in a future release. Therefore, all attributes that are INT type with a certain width such as INT(12) will be replaced as INT in SQL code. The size constraint of INT type will be enforced in the View instead of SQL code.

****

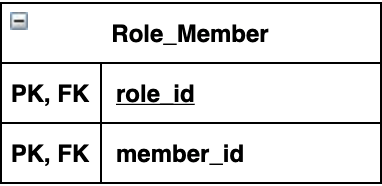
**Member** table stores all private data of all members in the gym.

* id: a unique 7-digit identity number assigned to each member.
* Data type: INT, NOT NULL, AUTO INCREMENT=1000000
* Primary Key (PK)
* Unique Identifier
* email: a unique email that each member registers.
* Data type: VARCHAR(100), NOT NULL
* username: a unique username that each member uses to access the application
* Data type: VARCHAR(50), NOT NULL, UNIQUE
* password: a password to access the application with corresponding username.
* Data type: VARCHAR(72) NOT NULL
* first\_name: first name of member
* Data type: VARCHAR(25), NOT NULL
* last\_name: last name of member.
* Data type: VARCHAR(25), NOT NULL
* address: address that member lives.
* Data type: VARCHAR(150), NOT NULL
* phone\_number: phone number for contact.
* Data type: VARCHAR(15), NOT NULL
* date\_of\_birth: birthday.
* Data type: DATE, NOT NULL
* gender
* Data type: VARCHAR(11), NOT NULL
* membership\_type: id of the membership type that member (gymmer) signs up
* Data type: INT
* Foreign Key (FK): references to Membership(id)
* date\_join: first date member signs up/joins the gym
* Data type: DATE, NOT NULL
* date\_expiration: last day that membership expires.
* Data type: DATE
* active: active status of this member in the gym.
* Data type: BOOLEAN, DEFAULT TRUE
* bank\_account\_number: a bank account number that member acquires.
* Data type: VARCHAR(50), NOT NULL
* Foreign Key (FK): references to Bank\_Account(account\_number)



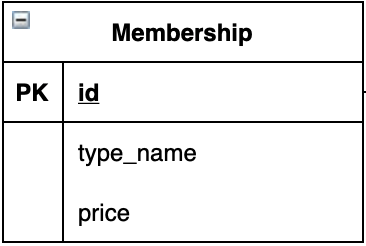
**Role** table stores all role types in the gym

* id: a unique identity number assigned to each role type.
* Data type: INT, NOT NULL, AUTO INCREMENT
* Primary Key (PK)
* Unique Identifier
* name: name of that role.
* Data type: VARCHAR(25), NOT NULL
* wage: wage per hour for the corresponding role
* Data type: DOUBLE



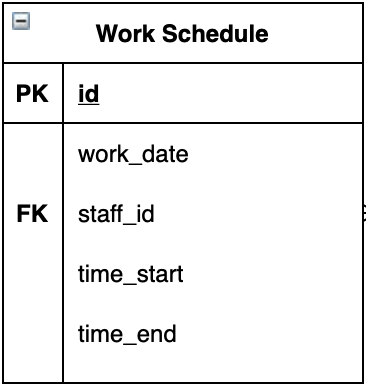
**Role\_Member** stores all members with their roles. This is necessary for Many-To-Many relationship between Member and Role tables.

* role\_id: a unique identity number of the corresponding role.
* Data type: INT, NOT NULL
* Primary Key (PK)
* Foreign Key (FK): references to Role(id)
* member\_id: a unique identity number of the corresponding member.
* Data type: INT, NOT NULL
* Primary Key (PK)
* Foreign Key (FK): references to Member(id)



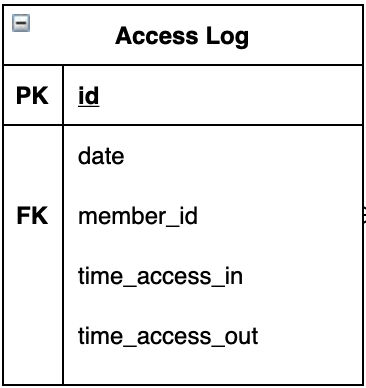
**Membership** table stores all membership types in the gym

* id: a unique identity number assigned to each membership type.
* Data type: INT, NOT NULL, AUTO INCREMENT
* Primary Key (PK)
* Unique Identifier
* type\_name: name of the role.
* Data type: VARCHAR(25), NOT NULL
* price: price for each membership type
* Data type: DOUBLE, NOT NULL



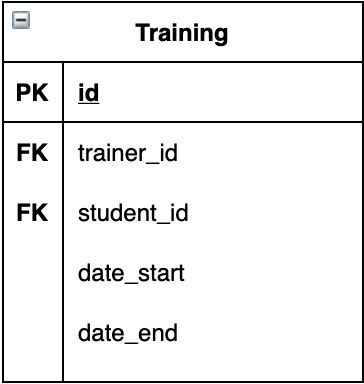
**Work\_Schedule** table stores all work schedule of all staffs in a month

* id: a unique identity number assigned to each membership type.
* Data type: INT, NOT NULL, AUTO INCREMENT
* Primary Key (PK)
* Unique Identifier
* work\_date: a date in the month.
* Data type: DATE, NOT NULL
* staff\_id: id of the staff from Member table
* Data type: INT, NOT NULL
* Foreign Key (FK): references to Member(id)
* time\_start: starting time of the work day.
* Data type: TIME, NOT NULL
* time\_end: ending time of the work day.
* Data type: TIME, NOT NULL



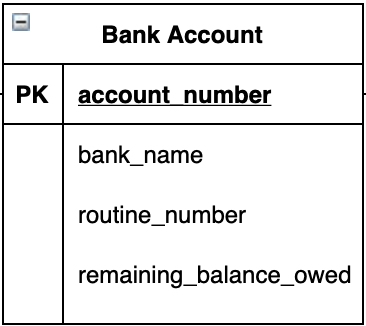
**Access\_Log** table stores all access info from all members in the gym

* id: a unique identity number assigned to each access.
* Data type: INT, NOT NULL, AUTO INCREMENT
* Primary Key (PK)
* Unique Identifier
* date: the date that member access to the gym.
* Data type: DATE, NOT NULL
* member\_id: identify number of the member
* Data type: INT, NOT NULL
* Foreign Key (FK): references to Member(id)
* time\_access\_in: the time that member accesses the gym.
* Data type: TIME, NOT NULL
* time\_access\_out: the time that member leaves the gym.
* Data type: TIME



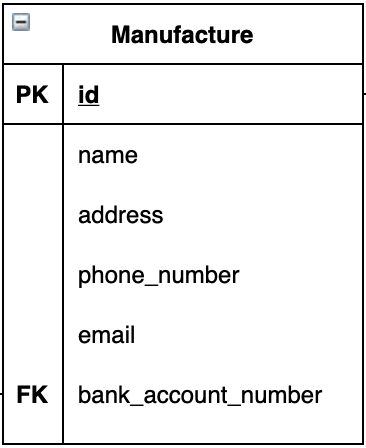
**Training** table stores all training pairs in the gym between trainer and student (gymmer)

* id: a unique identity number assigned to each training.
* Data type: INT, NOT NULL, AUTO INCREMENT
* Primary Key (PK)
* Unique Identifier
* trainer\_id: the identify number of the trainer.
* Data type: INT, NOT NULL
* Foreign Key (FK): references to Member(id)
* student\_id: the identify number of the student (gymmer)
* Data type: INT, NOT NULL
* Foreign Key (FK): references to Member(id)
* date\_start: the first day of this training.
* Data type: DATE, NOT NULL
* date\_end: the last day of this training.
* Data type: DATE



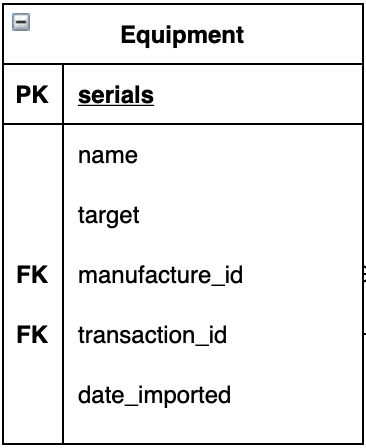
**Bank\_Account** table stores bank account information of members and manufacture (optional)

* account\_number: a unique bank account number
* Data type: VARCHAR(50), NOT NULL, UNIQUE
* Primary Key (PK)
* Unique Identifier
* Note: Most bank accounts in U.S. have from 8-12 digits. However, some bank accounts could be as many as 17 digits.
* bank\_name: name of the bank.
* Data type: VARCHAR(100), NOT NULL
* routine\_number: the routine number of this bank account
* Data type: INT, NOT NULL
* Note: Most of the time, routine number is a 9-digit number.
* remaining\_balance\_owed: the owed balance of members to the gym
* Data type: DOUBLE, DEFAULT 0.0



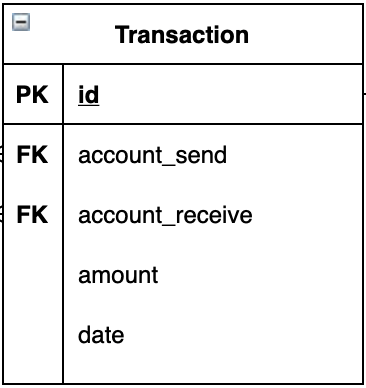
**Manufacture** table stores all manufacture information

* id: a unique identify number for the manufacture
* Data type: INT, AUTO\_INCREMENT
* Primary Key (PK)
* Unique Identifier
* name: the name of the manufacture.
* Data type: VARCHAR(100), NOT NULL
* address: the address of the manufacture
* Data type: VARCHAR(150), NOT NULL
* phone\_number: the phone number of the manufacture
* Data type: VARCHAR(15)
* email: the email of the manufacture
* Data type: VARCHAR(100)
* bank\_account\_number: the bank account number that the manufacture holds, so the gym can pay when buying equipments
* Data type: VARCHAR(50)
* Foreign Key (FK): references to Bank\_Account(bank\_account\_number)



**Equipment** table stores all equipment in the gym

* serials: a unique serial number for each equipment
* Data type: VARCHAR(30), NOT NULL
* Primary Key (PK)
* Unique Identifier
* name: the name of the equipment.
* Data type: VARCHAR(100), NOT NULL
* target: the target type from the equipment
* Data type: VARCHAR(50)
* manufacture\_id: the id of the manufacture that sells the equipment to the gym
* Data type: INT
* Foreign Key (FK): references to Manufacture(id)
* transaction\_id: the transaction that the gym buys the equipment
* Data type: INT
* Foreign Key (FK): references to Transaction(id)
* date\_imported: the date that this equipment imported
* Data type: DATE

****

**Transaction** table stores all transactions that the gym involves

* id: a unique identity number of each transaction
* Data type: INT, NOT NULL, AUTO\_INCREMENT
* Primary Key (PK)
* Unique Identifier
* account\_send: the bank account number sending money
* Data type: VARCHAR(50), NOT NULL
* Foreign Key (FK): references to Bank\_Account(account\_number)
* account\_receive: the bank account number receiving money
* Data type: VARCHAR(50), NOT NULL
* Foreign Key (FK): references to Bank\_Account(account\_number)
* amount: a specific amount of money
* Data type: DOUBLE, NOT NULL
* date: the date that transaction happens
* Data type: DATE, NOT NULL