Lebanese American University

School of Arts and Sciences

Database Management Systems

CSC375 | Section 11



Instructor: Ramzi Haraty | Project

Database for a Fitness Club

Members:

Lama Al Sheikh: 202100077

Jalal El Zein: 202105966

Hussein Fadel: 202105220

Rayane Yassine: 202103810

Monday, December 5th , 2022

Table of Contents:

Entity – Relation Diagram …………………………………………………….....1

Introduction to the Project ……………………………………………………….2

System Description ………………………………………………………. 3

Entity Types ………………………………………………………………...…....4

Body Composition ……………………………………………………....4

Branch …………………………………………………………………...5

Class ……………………………………………………………………..5

Employee ………………………………………………………………..6

Equipment ……………………………………………………………….7

Fee ……………………………………………………………………….7

Schedule ………………………………………………………………....8

Sellable …………………………………………………………………..8

Shop …………………………………………………………..………….9

Member …………………………………………………………………10

Membership ………………………………………………………….…10

Relationships ……………………………………………………………….…...11

Subscribes With A………………………………………………………11

Has A……………………………………………………………………12

Uses …………………………………………………………………….12

Follows ……………………………………………………………….…13

Manages ………………………………………………………………...13

Instructs …………………………………………………………………14

Works In …………………………………………………………………14

Takes Place In …………………………………………………………..14

Keeps ……………………………………………………………………15

Available In ……………………………………………………………..15

Pays ……………………………………………………………………..16

ER to Relational model………………………………………………………….16

Step One: Mapping of Regular Entity Types……………………………16

Branch ………………………………………………………….16

Class …………………………………………………………….16

Employee ………………………………………………………..17

Equipment ……………………………………………………….17

Fee ……………………………………………………………….17

Schedule ………………………………………………………....17

Sellable …………………………………………………………..18

Shop …………………………………………………………….18

Member …………………………………………………………18

Membership ………………………………………………….…18

Step Two: Mapping of Weak Entity Type……………………………….19

BODY\_COMPOSITION………………………………………..19

Step 3: Mapping of Binary 1:1 Relation Types…………………………..19

USES…………………………………………………………….19

PAYS……………………………………………………………20

HAS\_A………………………………………………………….20

TAKES\_PLACE\_IN……………………………………………20

Step 4: Mapping of Binary 1:N Relationship Types…………………….21

SUBSCRIBES\_WITH\_A……………………………………….21

KEEPS…………………………………………………………...21

WORKS\_IN……………………………………………….…….21

MANAGES……………………………………………………..21

INSTRUCTS……………………………………………………22

Step 5: Mapping of Binary M:N Relationship Types……………………22

AVAILABLE\_IN………………………………………………22

FOLLOWS……………………………………………………..22

Step 6: Mapping of Multivalued attributes………………………………23

Branch\_Phone………………………..…………………………23

Employee\_Shift…………………………………………………23

Employee\_Phone……………………………………………….24

Result of Mapping the ER Schema into a Relational Database Schema….……..24

Creating the Tables on Oracle Express………………………………………….25

Populating the Tables in Oracle Express……………………………………….29

Branch………………………………………………………………….29

Class……………………………………………………………………30

Equipment……………………………………………………………..31

Fee…………………………………………………………………….33

Member………………………………………………………………..35

Membership……………………………………………………………36

Schedule……………………………………………………………….37

Sellable…………………………………………………………………38

Shop……………………………………………………………………39

Available\_In……………………………………………………………40

Body\_Composition…………………………………………………….41

Branch\_Phone…………………………………………………………42

Employee\_Shift………………………………………………………..43

Employee\_Phone………………………………………………………44

Follows………………………………………………………………..45

Pays……………………………………………………………………46

Takes\_Place\_In………………………………………………………..47

Uses……………………………………………………………………48

Some Transactions on the Database……………………………………………49

Changing Time Zones!...........................................................................49

You get a raise, and you get a raise, everyone gets a raise!...................50

Researching Muscle Hypertrophy in Women…………………………51

Finding the Thieves……………………………………………………52

Who are our managers?..........................................................................53

Keeping the Promise!.............................................................................54

Premium Fitness Campaign……………………………………………55

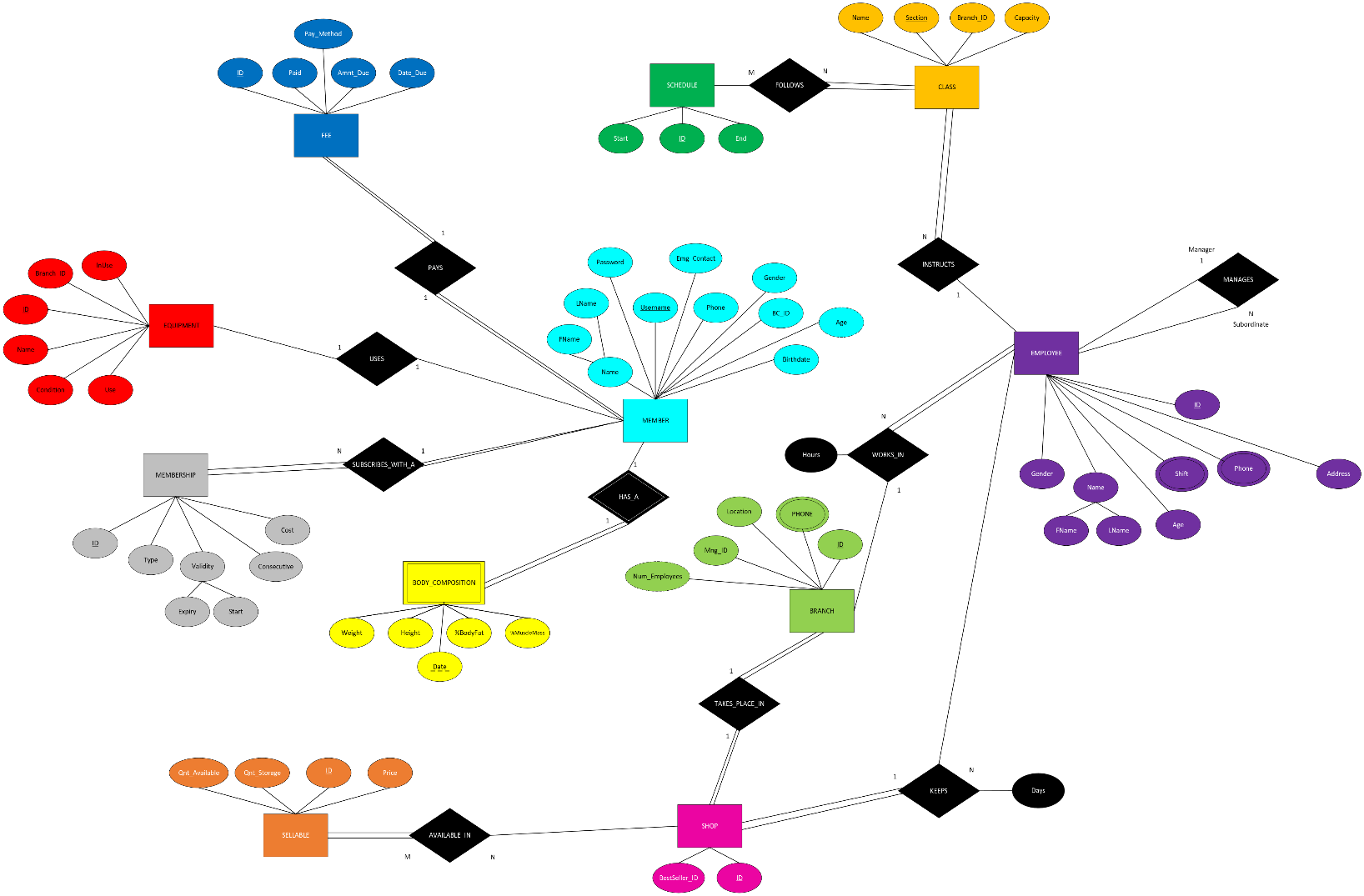
Jalal has a crush!.....................................................................................56

Sweaty Rowing………………………………………………………...57

Quantity over Quality…………………………………………………..58

Normalization……………………………………………………………………..60

Relations with non-prime attributes……………………………………………….70

1. ER-Diagram
2. ***Introduction to the Project:***

Our project is based around creating a database solution for a Fitness Club called “MUSCLE ASSEMBLY”. MUSCLE ASSEMBLY is a gym center founded in 2026 and located originally in Lebanon. Since then, it has spread to the ends of the globe for its innovative approach to the fitness journey. This innovative approach was developed by the four founding members, Hussein, Jalal, Lama, and Rayane who had the brilliant idea of integrating Augmented Reality (AR) technology and Artificial Intelligence (AI) to maximize the effectiveness of Muscle Hypertrophy. Through its membership system, MUSCLE ASSEMBLY allows its users to have the most efficient path to a perfect body with AI-assisted recommendations for both exercises and diet! The decision to integrate AR into the fitness atmosphere made it possible to apply social features like machine-specific leaderboards for members without resorting to distractive mobile apps. Other social features include noticeably increasing the size of every member’s glutes in AR to avoid some issues with perversion and sexual harassment in the fitness space, especially toward female members.

“When everyone has a HUGE ass, no one wants to look at yours!”

* Jalal El Zein, 2022



***III – System Description***

MUSCLE ASSEMBLY is a fitness club with a holistic view toward physical improvement and wellbeing. We aim to deliver a complete fitness journey at our facilities, encompassing diet, resistance training, endurance training, and equipment! In order to provide the best experience to our customers and maintain an ideal functioning status, we keep track of important information about our staff, facilities, and members in a secure database.

Our great success locally has allowed to us to expand with various branches to many parts of the world. And with this much diversity, comes the need to have some branches work a little differently than others. All branches fall into one of two categories, either they are traditional facilities, meaning they accept and employ human employees to operate the branch, while few are fully automated with AI-powered humanoid robots that take on the functions of an employee.

Another important side of the journey our members take with MUSCLE ASSEMBLY is the physical education they can receive through our provided classes that cover Weight Lifting, Calisthenics, Body Building, Cardio, CrossFit, etc. These classes vary from branch to branch, and from instructor to instructor, so it’s important to keep track of which class is taught where and by who. These classes also follow specific, pre-set templates for schedules that indicate the starting and ending times for said classes in the days that they are given.

The backbone of our institution, is our rigid body of employees that work hard to provide the customers with the best possible experience. We have many types of employees, each specialized in their roles, which include:

* Instructors
* Shop Keepers
* Managers
* Janitors
* Receptionists

For our members, we store important information to ensure optimal service. This information can include unique account information such as usernames and passwords, contact information, and things like birthdate to account for different age groups in relation to physical needs, and tracking body composition for our specialized AI tools that facilitate analysis of performance and progress.

Our members subscribe to our services with different types of memberships. These different tiers of membership have different costs depending on the membership type and validity. Each member chooses the payment method of their choice to pay the fee.

One of the services provided to our members is the access to different equipment such as machines, dumbbells, yoga mats, and other apparatus which are unique to each branch and target all body parts.

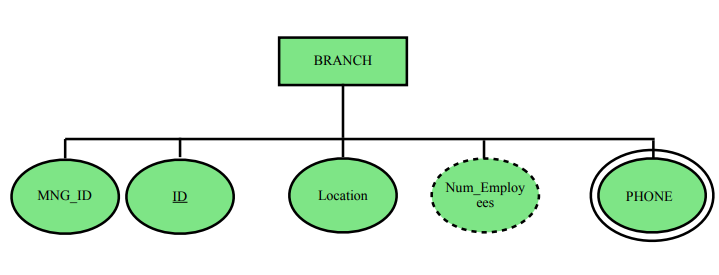
***IV – Entity Types***

1 – Body Composition

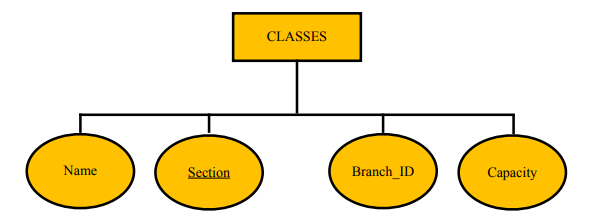
Body Composition

The *Body Composition* is a “Weak Entity Type” concerned with a member. Silver and above Members can opt in to track this information in MUSCLE ASSEMBLY’s database. Utilizing the Artificially Intelligent Companion (AIC), MUSCLE ASSEMBLY analyzes information from the database regarding a member’s workouts, including meals consumed, machine used, time spent on each machine, and many other parameters. This AIC provides members with a complete plan depending on their goals, and even recommends purchases from the shop to suite the daily nutritional intake of members in relation with energy spent during a workout session!

The Body Composition entity has four attributes in the database, the *Height* <int> in centimeters, *Weight* <int> in kilograms,  *Percent Body Fat* (%BodyFat) <int>, and *Percent Muscle Mass* (%MuscleMass) <int>.

2 – Branch

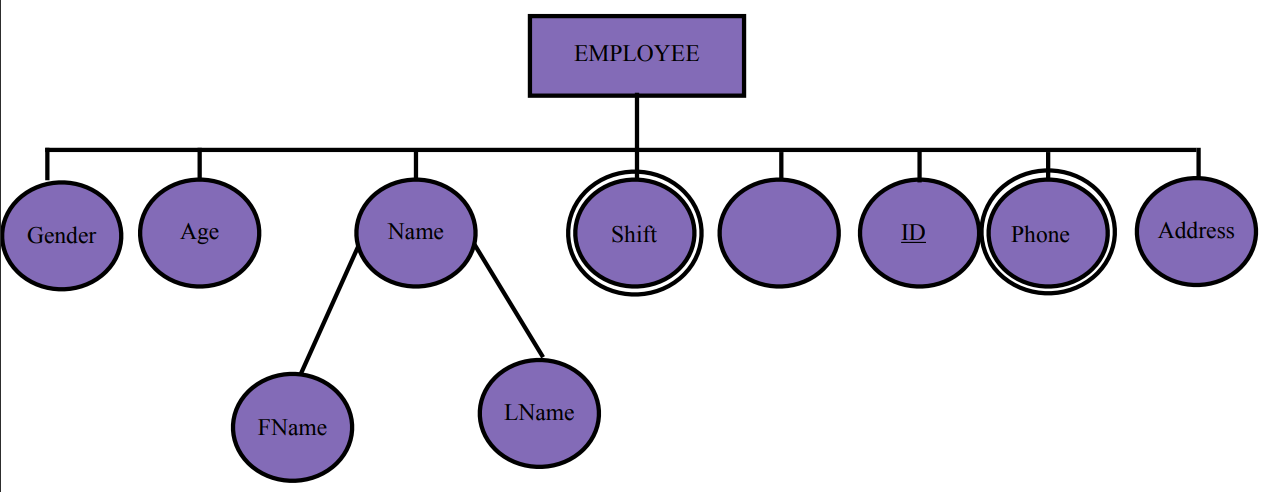
The branch location of the gym. MUSCLE ASSEMBLE has club locations all around the world to provide as many people as possible with the best fitness journey created, so it’s important that every region keeps sound records of data about its branches. Each branch has different employees, equipment, payments and schedules. Every branch is identified by a specific *ID* of type integer which is the attribute. The attribute *Mng\_ID*, also of type integer, represents the ID of the manager who is responsible for the branch. To represent the gym location, the attribute *Location* of type string was used. Each branch can have multiple phone numbers, thus the use of the multivalued attribute *Phone* of type string. Finally, the *Num\_Employees* is a derived attribute of type integer that represents the number of employees working in the branch. This number is not present but can be calculated.

3 – Class

CLASS

CLASS are educational sessions provided to members by world-class trainers, with subjects varying from beginner workouts, to Olympic level body building and fitness. CLASS are given by a trainer for a specific workout. Every class has a *Section* attribute of type integer, a *Name* attribute of type string, a *Branch\_ID* of type integer to specify to which branch the class belongs to, and a *Capacity* attribute of type integer that represents the number of members a class can contain.

4 – Employee



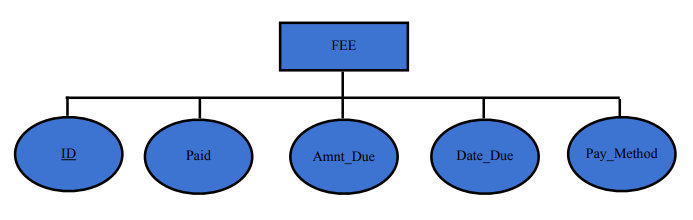
Salary \_ID

An employee is an individual who works under control of an employer. The *Employee* entity has seven attributes. Employee can divide into 3 main employees which are instructor, shop keeper, or manager. The instructor will help the members and instruct them in different CLASS. The shop keeper works at the shop. Also, the manager manages the employees. Each employee will have unique ID, and a specific name. *Name* is a composite attribute indicating the employee’s name (first and last name). Each employee also has other attributes that assist in determining them, such as their *Gender*, *Address*, *Date* *of birth*, and *Age*. Also, each employee will work at specific *Shift,* either in the day or the night, which is specified by its respective attribute of type character. The *Salary* for each employee differs from one another, such that employees will have their own *salary\_ID* which is given at a specific time. Every employee can have one or more *Phone* number (cellular number, home, office, etc.), so that people can reach the employee when its needed.

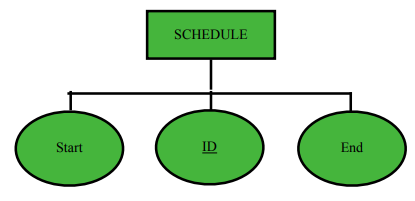
5 – Equipment

EQUIPMENT

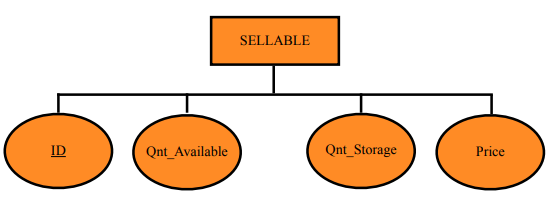
Machines and other gym supplies (e.g., dumbbells, yoga mats, etc.). These machines are the heart and soul of any fitness club and gym, MUSCLE ASSEMBLY considers them especially an important asset due to the AR augmentations on most machines. Each and every equipment is identified by a unique *ID* of type integer. The attribute *Branch \_ID*, also of type integer, represents the branch ID to which the equipment belongs to. Moreover, the attribute *InUse* of type Boolean indicates the availability of the equipment, whether it's being used by a member or not. The attribute *Condition* of type string specifies the state of the equipment, whether it's broken, under repair, or in good condition. The attribute *Name* is of type String that represents the name of each equipment existing in the MUSCLE ASSEMBLY gym .

6 – Fee

The fee is the amount the individual has to pay in order to purchase a subscription and be provided with all the fitness center’s services. The member is provided with high flexibility with the due date and the various payment methods available. The unique primary key attribute is the *ID* <integer> used to identify every payment done or still required. The second attribute is *Amnt\_Due* <decimal> which holds the amount the member has to pay to renew the subscription. The attribute *Date\_Due* <date> indicates the deadline before which the payment has to be made. At last, there is the *Paid* <Boolean> attribute which indicates whether the payment has been done or not.

7 – Schedule

A schedule is a plan that indicates the duration of each class. Schedule entity has three attributes. The first two attributes are the *Start* and the *End* time <time> of each class. The third attribute represents the schedule’s *ID* <integer>.

8 – Sellable

A sellable is what the member can buy at the fitness center. This can be edible products to provide the member with energy and sustain such as protein shakes, energy bars, and other not necessarily fitness related products. As well as accessories such as lifting belts, gloves, different kinds of water bottles, and even sportswear. The *Sellable* entity has four attributes, the first of which is the *ID* <integer> attribute it must be unique for every type of product. The second attribute is *Qnt\_Available* <integer> which is the quantity of the product directly available for purchase. For the products not directly available in the shops there is the *Qnt\_ Storage* <integer> attribute which tracks the quantity of the sellable available in storage that needs to be transported to the shop in order to be available for display. Finally, there is the *Price* <decimal> attribute which as the name suggests indicates the price of the specific product.

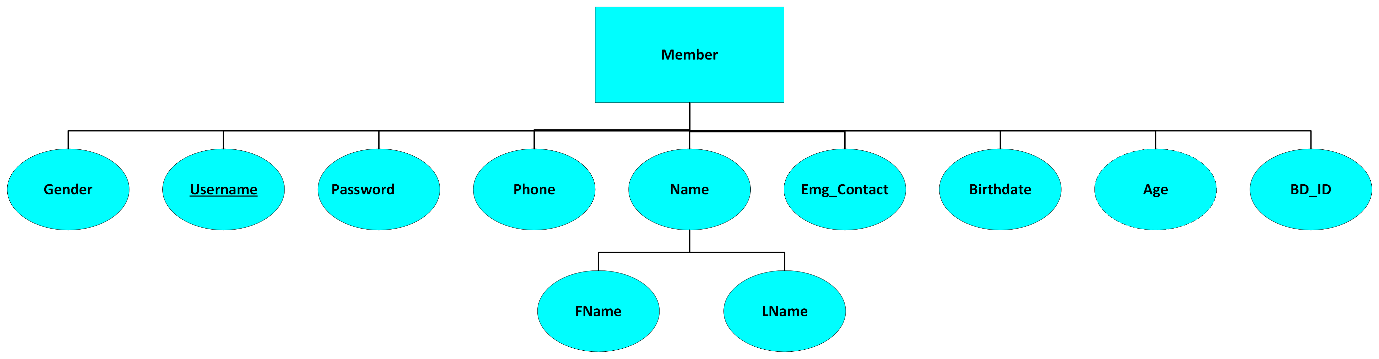
9 – Shop

SHOP

**ID**

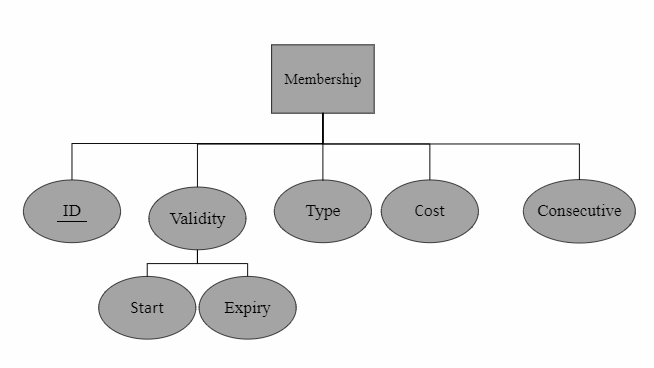
**BestSeller\_ID**

A place where a member can buy whatever they want to help them with their fitness journey. Based on their routine and type of workout, a member can be guided through recommendations on what and when to buy from the shop for the best possible results. The shop also displays the products for easy access based on their popularity to make the shopping process easy and comfortable. The *Shop* entity has two attributes the ID entity and the BestSeller\_ ID. The BestSeller\_ ID is of type integer. The ID which is the primary key of type integer to identify the shop.

10 – Member

The *Member* is a user of the gym, its online portal, and its shop and other facilities. This entity holds various attributes because it’s concerned directly with the most important resource to MUSCLE ASSEMBLY; it’s loyal customers! We keep track of the basic information about members such as their names in a composite attribute, holding their *First Name* (FName) <String> and *Last Name* (LName) <String>, their *Username* <String>, *Password* <String>, *Age* <Integer>, *Gender* <char>, *Phone Number* <String>, *Emergency Phone Number Contact* (Emg \_ Contact) <String>. On top of that, as mentioned before Silver Members can opt in to track their *Body Composition*!

11– Membership

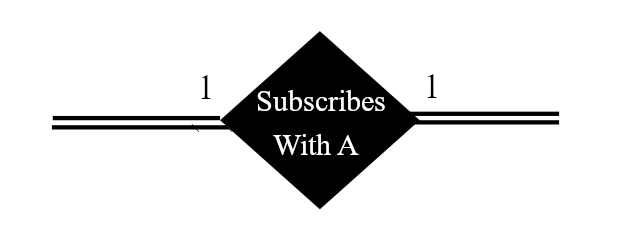


The *Membership* entity represents the subscription a *member* buys to access MUSCLE ASSEMBLY. Memberships are purchased on a monthly basis, and naturally, there are multiple tiers and variations of subscriptions to choose from, with the most expensive providing greater benefits. A “Bronze Tier” subscription is the cheapest option and provides no special perks besides access to the club’s facilities, including of course, all the AR functionality MUSCLE ASSEMBLY is famous for. Upgrading to “Silver Tier” will grant a MEMBER access to the Body Composition tracking equipment, and the AIC, in addition to 2 professional CLASS. Taking a subscription one step further to “Gold Tier” is not easy, as it required 3 consecutive membership renewals in addition to the increased fee. Even with this difficulty of becoming a GOLD member, all members are incentivized to reach that point because GOLD members are granted access to all CLASS, as well as free unlimited access to shops, in addition to cross-branch data sharing and unlocking social features like machine-specific leaderboards.

The *Membership* entity has five attributes, *ID* <int>, the *Type* <String> which refers to the tier of the membership, the *Validity* is another composite attribute that consists of two attributes, *Start* <Date> indicating the date the membership becomes valid, and the *Expiry* <Date> which indicates when this Membership is no longer valid. In addition, we have a *Consecutive* <int> attribute that tracks how many consecutive memberships a member has bought, and the *Cost* <int> attribute specifies the price of a specific membership.

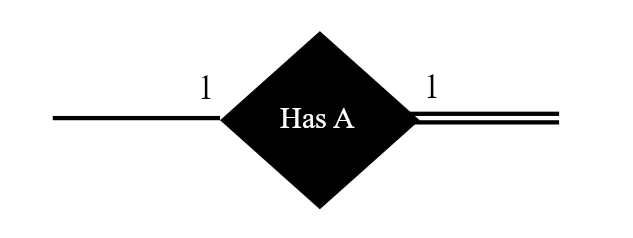
***V - Relationships:***

1 – *Subscribes With A*:

 The “*Subscribes With A”* relationship occurs between a MEMBER and a MEMBERSHIP; a MEMBER *Subscribes with A* MEMBERSHIP. This relation is a One-To-One relation, since every Member can have only one valid membership or subscription at any point in time. In this relationship, both members and memberships are totally participating, since all memberships need to be associated with a member, and a member cannot use the club without a membership!

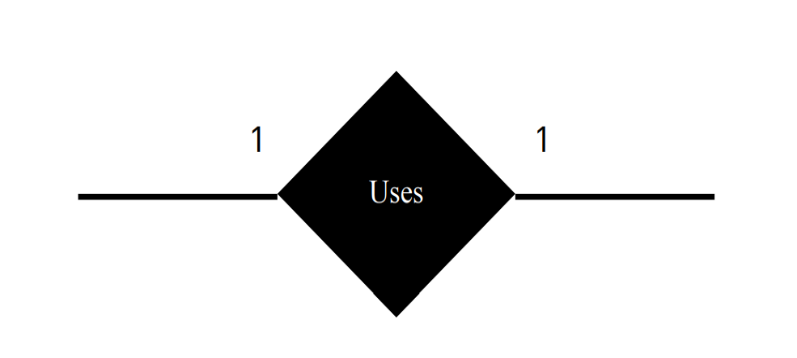
2 - *Has A*:

This “*Has A*” is One-To-One since it occurs between a MEMBER entity and a BODY COMPOSITION. Since Body Composition is a weak entity type, this relationship is also an Identifying Relationship because this relationship “Identifies” a Body Composition by linking it to a MEMBER. The relation is defined to be such that Body Composition totally participates in the relationship, while a member partially participates, this is because not all members can access this feature, and even those who do can choose not opt out of it. Thus, all Body Compositions need to be associated with a MEMBER, but not all Members need to have a Body Composition associated with them.



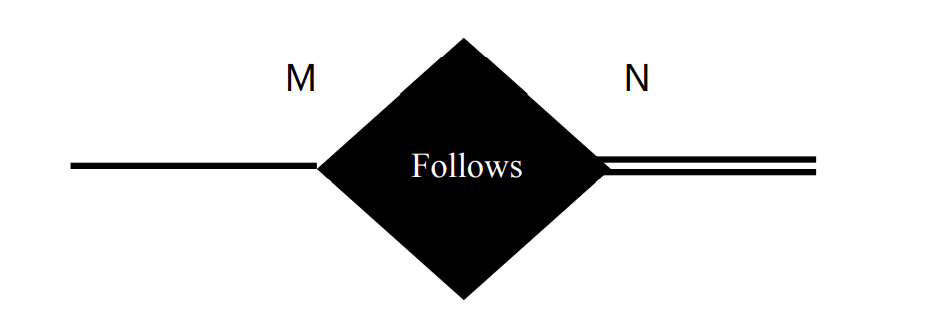
3 - *Uses*:

The “uses” relationship is created between the MEMBER entity and the EQUIPMENT entity to show that each member uses one equipment at a time. This relationship has partial participation from both sides, since not all MEMBERs are using EQUIPMENTs all the time, and EQUIPMENTs do not need to be used by a MEMBER at all times. Additionally, one MEMBER can use a single machine at a time, so this relation is also One-To-One.

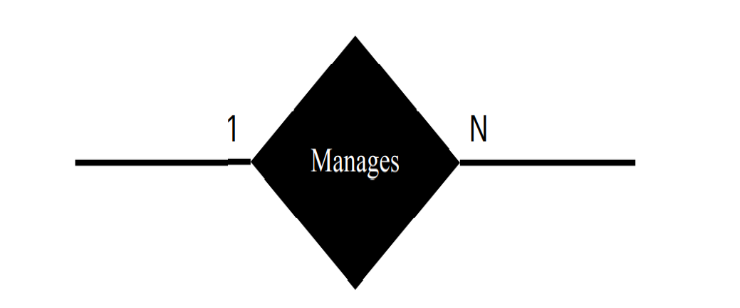


4 – *Follows*:

The “*follows*” relationship is created between the SCHEDULE entity and the CLASS entity with a total participation from CLASSES to show that all CLASS follow a schedule but not all schedules are associated with a class. Many *CLASSes* can manage many SCHEDULES, so this entity is Many-To-Many Relationship. In other words, there might unused schedules.

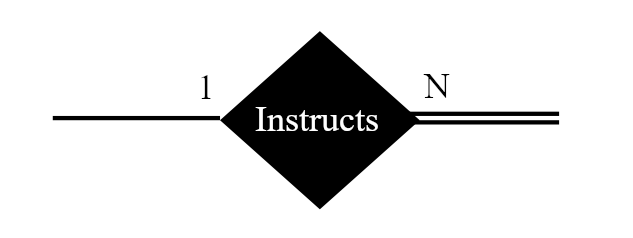


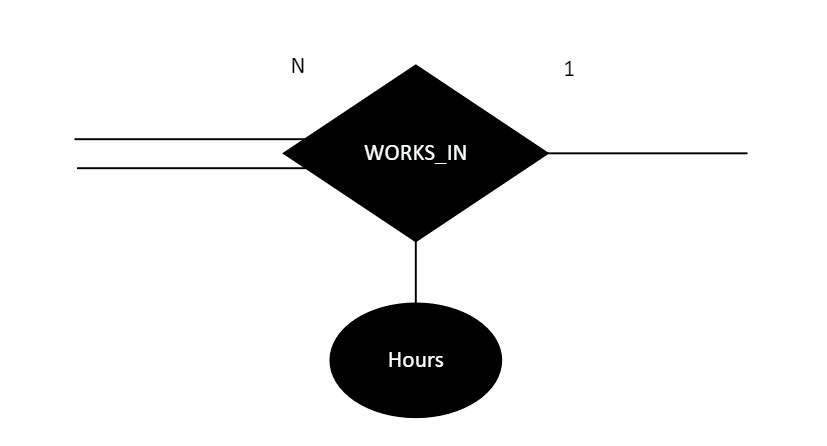
5 – *Manages*:

The “*Manages*” relationship is created between the EMPLOYEE entity (as a manager) and also the EMPLOYEE entity (as a subordinate). One manager can manage many employees, so this entity is One-To-Many. This relationship has partial participation on both sides, since not all employees are managing others, and not all employees are managed by others.

6 - *Instructs*:

The “*Instructs*” relationship is created between the CLASS entity and the EMPLOYEE entity. The participation is total on the CLASS side since all CLASS are instructed by an employee, and employees can instruct several CLASSes. But one class can only be instructed by one instructor.

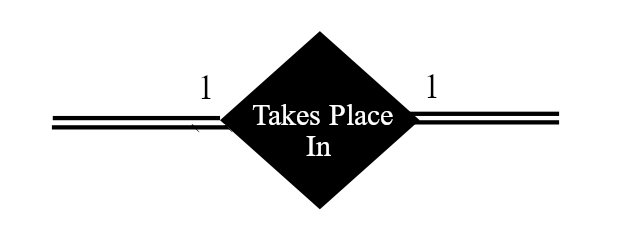


7 – *Works In*:

The “*Works In*” relationship is created between the EMPLOYEE entity and BRANCH entity. The relationship has an attribute, *Hours* indicating the number of hours the EMPLOYEE has to work at that particular branch per week. This relation is One-To-Many since more than one employee works at a branch, but an employee can work in one branch only. Also, this relationship has total participation from the EMPLOYEE side, since all employees need to be associated with a branch, but not all branches need to have employees (Some newer branches are fully automated!).

8 – *Takes Place In*

The “*Takes Place In*” relationship is created between the entities SHOP and BRANCH to indicate that a shop is located at a certain branch. This is a One-To-One relationship which means that a branch can have only one shop. Moreover, there it requires full participation which means that every branch has to have a shop and every shop must be in a branch.

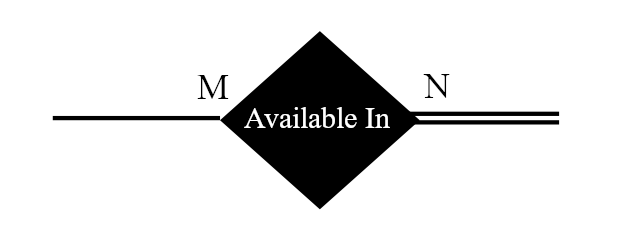


9 - *Keeps*

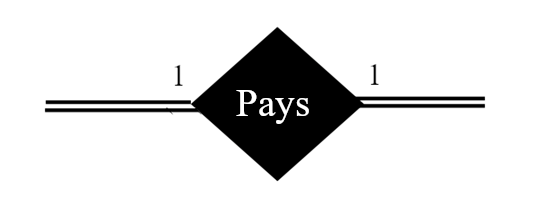
The “*Keeps*” relationship is created between the EMPLOYEE and the SHOP entities which means that an employee keeps a shop. The relationship has an attribute, *Date* indicating the date that the shopkeeper started to work in the shops. It is a One-To-Many relationship because multiple employees can keep one shop. And every shop must have an employee as a keeper. This relation has total participation from SHOP entity but not all Employees are associated with a shop.

10 - *Available In*:

The “*Available In*” relationship created between the SELLABLE and SHOP entities is a Many-To-Many relationship that indicates what products are available in what shop. It also indicates that every sellable must be available in a shop (full participation for SELLABLE, but partial participation for SHOP).



11 – *Pays*:

 The “*Pays*” relationship created between the MEMBER and FEE entities is a One-To-One relation that indicates what payments do the members pay in the gym, which is a sum of their membership cost, as well as additional spending in the club shop. It has total participation on both sides of the relationship because all fees need to be paid by a member, and no member can have a membership without the fee.

***VI – ER-to-Relational Mapping***

This section is concerned with translating our high-level design of entity types, attributes, and relationships into a relational database design. We achieved this by following the Seven-Step ER-to-Relational Mapping Algorithm.

**Step One: Mapping of Regular Entity Types:**

Step one consists of the regular entity types that are mapped into relations. The eleven entity types include different of simple, derived, multivalued attributes. However, only the simple attributes will be included in the relation. Also, it consists of the primary key, and it is known by underlining it in the relation.

1. **Branch**

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Mng\_ ID | Phone | Location |

The Branch is an entity type that includes simple and derived attributes. However, the derived attribute representing the number of employees, “*Num\_Employees*” is not represented in this relation. The primary key for this relation, “*ID*”, is underlined. The remaining attributes, “Mng\_ID” and “Phone” are also simple attributes.

1. **Class**

|  |  |  |  |
| --- | --- | --- | --- |
| Section | Name | Branch\_ ID | Capacity |

The Class is an entity type that contains only simple attributes, like the “*Name*”, “*Branch\_ID*”, and “*Capacity*”, and are represented normally. Additionally, we have another simple attribute, “*Section”* which serves as the primary key for this relation.

1. **Employee**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Gender | Age | Salary | FName | LName | Address |

The Employee is an entity type that has simple, multivalued and composite attributes. The Employee entity has the *NAME* as composite attribute of two simple attributes FName and LName. In addition, this relation consists of one simple primary key attribute “ID”. On the other hand, the multivalued attributes “Shift” and “Phone” are not represented in this relation. The remaining attributes “Gender”, “Age”, “Salary”, and “Address”, are the simple attributes.

1. **Equipment**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Branch\_ ID | In Use | Condition | Name |

The Equipment is an entity type that consists of simple attributes only. The “ID” is the primary key attribute for this relation. Furthermore, the remaining attributes “Branch\_ ID”, “In Use”, “Condition” and “Name” are the simple attributes.

1. **Fee**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Paid | Amnt\_ Due | Date\_ Due | Pay\_ Method |

The Fee entity type includes only simple attributes. The “ID” attribute is the primary key in this relation. Moreover, the remaining attributes are the simple attributes such as “Paid”, “Amnt\_ Due”, “Date\_ Due”, and “Pay\_ Method”.

1. **Schedule**

|  |  |  |
| --- | --- | --- |
| ID | Start | End |

Schedule is an entity that insists of simple attributes only. The simple attributes are the “Start” and the “End” attributes. Besides, the “ID” simple attribute is the primary key attribute in this relation.

1. **Sellable**

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Qnt\_ Storage | Qnt\_ Available | Price |

The Sellable is an entity type that includes only simple attributes that are represented normally, like the “Qnt\_ Storage”, “Qnt\_ Available”, and “Price”. The primary key for this relation, “ID”, is underlined and also a simple attribute.

1. **Shop**

|  |  |
| --- | --- |
| ID | BestSeller\_ID |

The Shop is an entity type that contains only two simple attributes, the “BestSeller\_ID”, and “ID” which serves as the primary key for this relation.

1. **Member**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Username | Gender | Birthdate | FName | LName | Address | Password | BC\_ID | Emg\_Contact | Phone |

The Member is an entity type that has simple, derived, and composite attributes. The Member entity has the “Name” as composite attribute of two simple attributes FName and LName. However, the derived attribute representing the age of the member, “Age” is not represented in this relation. In addition, this relation consists of one simple primary key attribute “Username”. The remaining attributes “Gender”, “Birthdate”, “Address”, “Password”, “BC\_ID”, “Emg\_Contact”, and “Phone” are the simple attributes.

1. **Membership**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | Type | Expiry | Start | Consecutive | Cost |

The Membership is an entity type that has simple and composite attributes. The Membership entity has the “Validity” as composite attribute of two simple attributes Expiry and Start. In addition, this relation consists of one simple primary key attribute “ID”. The remaining attributes “Type”, “Consecutive”, and “Cost” are the simple attributes.

Step Two: Mapping of Weak Entity Types:

In this step, there is only one weak entity that will be mapped into relation. This relation consists of simple attributes only. Also, weak entity relation includes a foreign key. Whereas, the partial key that is represented in the weak relation with dotted underline it, represent the primary key.

**BODY\_COMPOSITION**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date | Weight | Height | %BodyFat | %MuscleMass |

The BODY\_COMPOSITION is a weak entity type that has only simple attributes, the “Weight”, “Height”, “%BodyFat”, and “%MuscleMass”. The partial key for this relation is the “Date” which is also a simple attribute.

**Step Three: Mapping of Binary 1:1 Relationship Types:**

1. **USES**

|  |  |
| --- | --- |
| Username | E\_id |

USES is a one-to-one relationship that relates the two entities “Member” and “Equipment”, which means that one Member can use only one Equipment. No total participation is present in the relationship since a Member can exist without using any equipment and an Equipment can be without being used by a Member. It has two fields, the first one holds the Username of the Member related to the equipment while the second field holds the E\_id which is the ID of the equipment and is a foreign key in the Member entity.

1. **PAYS**

|  |  |
| --- | --- |
| Username | F\_id |

PAYS is a one-to-one relationship between the “Member” and “Fee” entities which states that one Member can have only one Fee and vice versa, there is also a requirement for full participation which means that every member must be related to a fee and the opposite as well. The relationship has only two fields each holding the primary key of the two entities. Username is the primary key of the Member entity while F\_id is the primary key “ID” of the Fee which acts as a foreign key in the Member entity.

1. **HAS\_A**

|  |  |
| --- | --- |
| Username | BC\_Date |

The relationship HAS A relates the Member entity with the weak entity Body Composition. It is one-to-one and has partial participation from the Member side since the member can choose whether to track or not their body composition, and a full participation from the Body Composition side since every body composition must belong to a member. The relationship has only one field which holds the Username of the Member that acts as a primary key for the Member entity.

1. **TAKES PLACE IN**

|  |  |
| --- | --- |
| B\_id | S\_id |

TAKES PLACE IN is a relationship between Branch and Shop which is one-to-one and requires full participation from both entities it relates. The two fields in this relationship hold the primary key of Branch “ID” and the primary key of Shop “ID” where “S\_id” (Shop ID) is a foreign key in the Branch entity which links it to the Shop entity.

**Step 4: Mapping of Binary 1:N Relationship Types**

**1 - SUBSCRIBES\_WITH\_A**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Type | Expiry | Start | Consecutive | Cost | Username |

This relationship connects the MEMBERs and their MEMBERSHIPs. Since it’s a One-To-Many relationship, we are going to use the foreign key approach, by adding the primary of the 1-Side entity to the relation of the N-Side entity as a foreign key. In this case, we add the “Username” attribute to the MEMBERSHIP relation.

**2 – KEEPS**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Gender | Age | Salary | FName | LName | Address | Shop\_ID | Keeping\_Days |

The “KEEPS” relationship connects EMPLOYEEs with a certain SHOP that they manage and work in at certain days of the week. This relation has to be One-To-Many since multiple people can keep the shop at different days, and even in the same day. Moreover, it has full participation on the side of the SHOP since all ships need to be managed by an EMPLOYEE. We use the foreign key approach here as well by including the “Shop\_ID” attribute to the EMPLOYEE relation. But we also need to add the “Keeping\_Days” attribute since it is an attribute on the relationship and needs to be mapped here as well.

**3 - WORKS\_IN**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Gender | Age | Salary | FName | LName | Address | Shop\_ID | Keeping\_Days | B\_ID |

To map the “WORKS\_IN” relationship, we also resort to the foreign key approach, where we get the primary key of the 1-Side entity to the relation of the N-Side entity. In this case, “WORKS\_IN” relates EMPLOYEEs and BRANCHes, so we get the Branch’s ID as an attribute, “B\_ID”, into the EMPLOYEE relation.

**4 – MANAGES**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Gender | Age | Salary | FName | LName | Address | Shop\_ID | Keeping\_Days | B\_ID | Super\_ID |

This self-referencing relationship type describes how an EMPLOYEE can be the manager of another EMPLOYEE. It is One-To-Many since one EMPLOYEE can manage many EMPLOYEEs, also it has only partial participation on both sides since not all Employees are managed and not all EMPLOYEEs are managers. To translate this relation using the foreign key approach, we add the Manager’s ID, “Super\_ID”, to the EMPLOYEE relation.

**5 – INSTRUCTS**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Gender | Age | Salary | FName | LName | Address | Shop\_ID | Keeping\_Days | B\_ID | Super\_ID | C\_ID |

This is yet another One-To-Many relation, since an EMPLOYEE can teach many classes, also, it has full participation on the side of the Class since every Class needs to be taught by an EMPLOYEE, but not all EMPLOYEEs teach. For this relationship, we add the CLASS’s ID, “C\_ID”, as a foreign key in the EMPLOYEE relation as well.

**Step 5: Mapping of Binary M:N Relationship Types**

1. **AVAILABE\_IN**

|  |  |
| --- | --- |
| S\_id | Sell\_id |

AVAILABLE\_IN is a relationship between the two entities Shop and Sellable, it is many-to-many which means that many multiple shops can be related to the same sellable and multiple sellable items can be related to the same shop. There is partial participation from the shop’s side which means that a shop can be not related to any item, however the sellable side there is full participation which requires every item to be related to a shop. The relationship has two fields, one with “S\_id” which is the Shop “ID” (primary key) and the other with “Sell\_id” which is the primary key “ID” of the Sellable and the foreign key in the Shop entity.

1. **FOLLOWS**

|  |  |
| --- | --- |
| C\_ id | Sched\_ id |

The relationship FOLLOWS between the entities Class and Schedule is a many-to-many relationship which can relate multiple classes to a schedule and vice versa. It requires full participation from the Class side which means that every class must have a schedule, but partial participation from the Schedule side which means that a schedule can exist without being related to a class. Like the relationship before, it has only two fields “C\_id” and “Sched\_id” each holding the primary keys of the Class and Schedule respectively. Moreover, “Sched\_id” acts as a foreign key in the Class entity.

**Step 6: Mapping of Multivalued attributes**

In this step, the multivalued attributes will be mapped. Each multivalued attribute will have its own relation, which will have also simple and primary key. We have three multivalued attributes which are:

Branch\_ Phone, Employee\_ Shift, Employee\_ Phone.

1. **Branch\_ Phone**

|  |  |
| --- | --- |
| Branch \_ ID | Phone |

Phone is a multivalued attribute which belongs to the Branch entity. The primary key of this relation is the “Branch\_ Id” which is the primary key of Branch entity type. The “Phone” indicates that each branch may have several phone numbers.

1. **Employee \_ Shift**

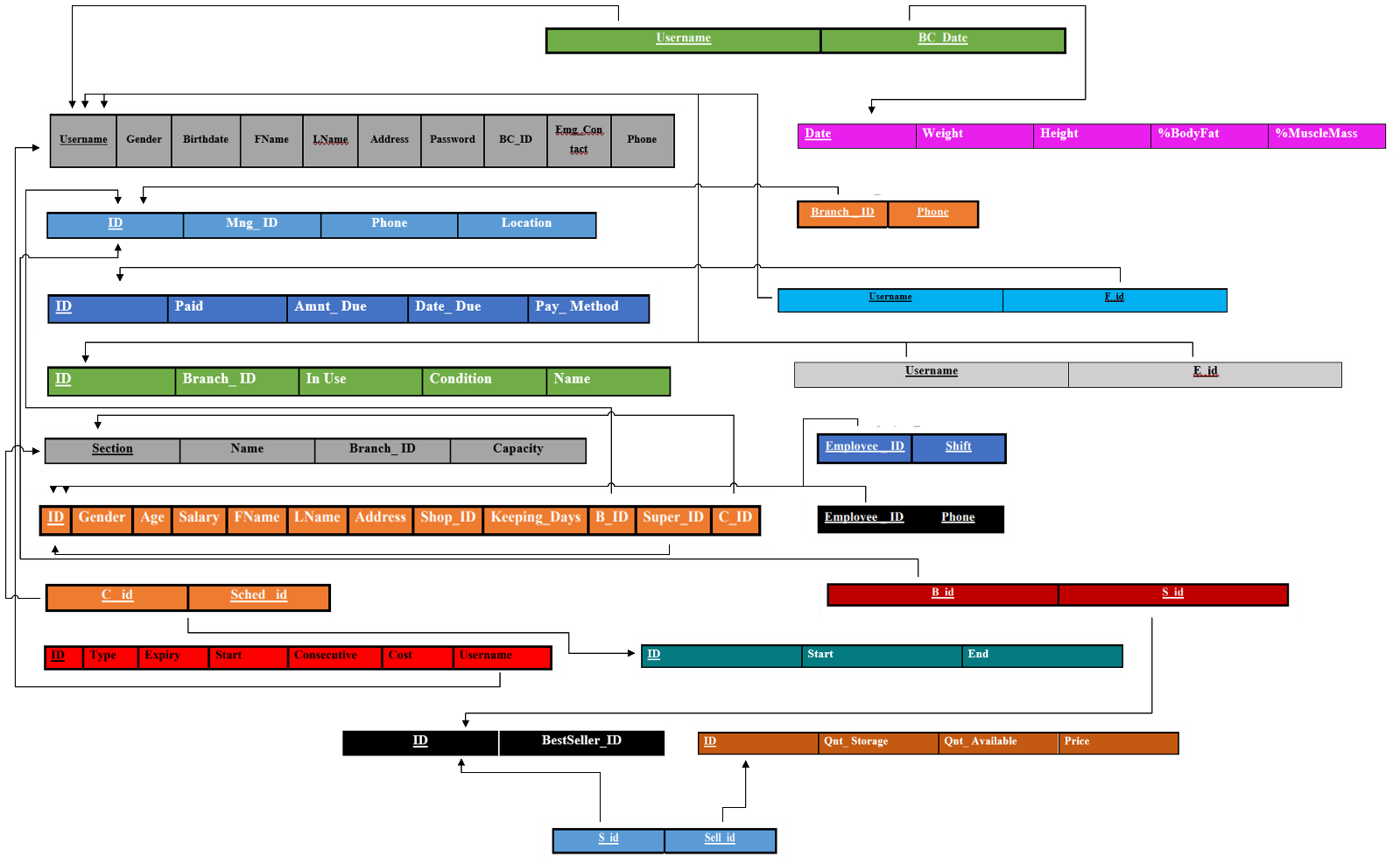
|  |  |
| --- | --- |
| Employee \_ ID | Shift |

Shift is a multivalued attribute which belongs to the Employee entity. The “Shift” indicates that an employee can have several shifts per day (Day shift or night shift). Moreover, the primary key in this relation is Employee \_ ID which belongs to the Employee and serves as its primary key.

1. **Employee \_ Phone:**

|  |  |
| --- | --- |
| Employee \_ ID | Phone |

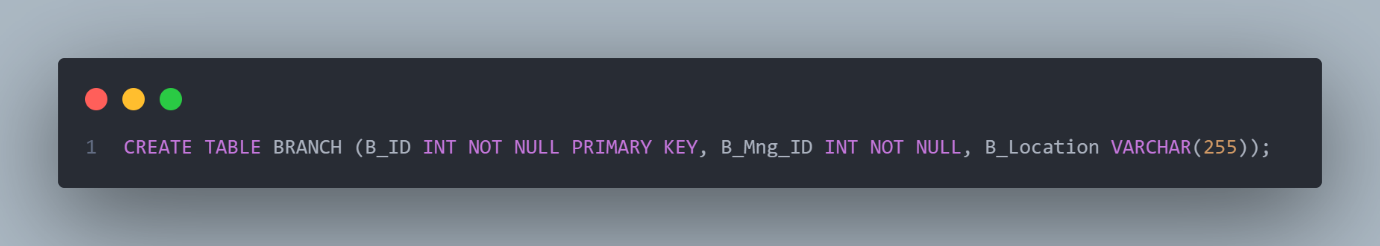
Phone is a multivalued attribute which belongs to the Employee entity. The primary key of this relation is the “Employee\_ Id” which is the primary key related to Employee entity type. Additionally, the “Phone” indicates that each Employee can have several phone numbers.

***Result of Mapping the MUSCLE ASSEMBLY ER Schema into a Relational Database Schema***

***VII – Creating the Tables on Oracle Express***

To implement our database design, we are using the Oracle Express 11g software as a Database Management System. Accordingly, we used PL/SQL as a querying language.

Here, we will use the CREATE TABLE command to create tables in our database.

The example below shows the full command to create the *Branch* table.

CREATE TABLE is the SQL reserved keywords to create a new table

BRANCH is the name of the table

B\_ID is the name of the first attribute representing the branch’s ID

* INT (Integer) is the type of this field
* NOT NULL specifies that this field does not accept NULL values
* PRIMARY KEY indicates that this field is the primary key for this table

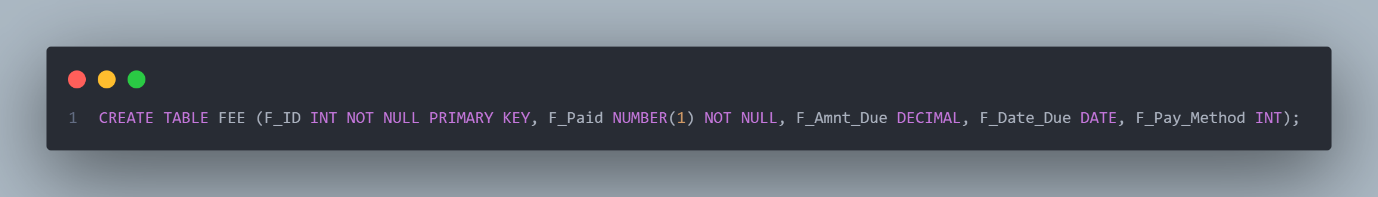
B\_Mng\_ID is the name of the second attribute representing the branch’s manager’s ID

* INT (Integer) is the type of this field
* NOT NULL indicates that we do not accept NULL values for this field

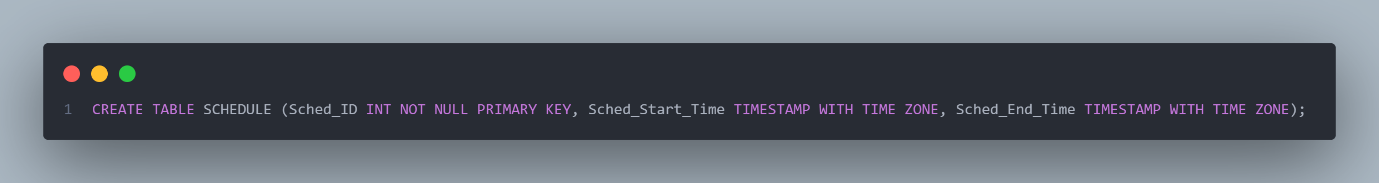
B\_Location is the name of the third attribute representing the branch’s location address

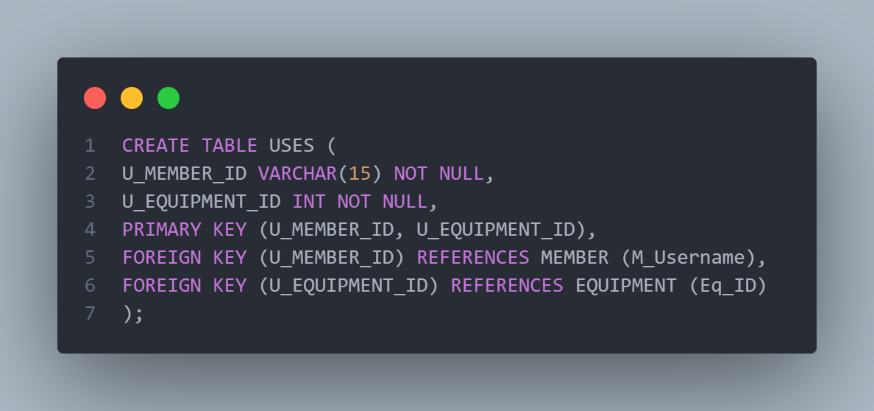
* VARCHAR(255) is the type of this field, a string of characters with variable length, with a default length of 255.

FEE:



In the above example, we see F\_PAID which is a field that represents whether or not the fee has been paid or not and acts as a Boolean value. Since Boolean data types do not exist in PL/SQL, we use a number in range 0 to 1 instead, where 0 represents a *false* value and 1 represents a *true* value.



Here, we have Sched\_Start\_Time, which is a field representing at what time of day a certain class starts, and similarly we have Sched\_End\_Time to represent the time of day that same class ends. To implement these values, we used TIMESTAMP WITH TIME ZONE which is a datatype that gives us the option to format time with hours, minutes, and seconds!

In this command, we see an example of a table representing a relationship USES, which uses the relation approach to represent itself in a table. We use the PRIMARY KEY (U\_MEMBER\_ID, U\_EQUIPMENT\_ID) in this syntax to signify that the combination of the U\_MEMBER\_ID and U\_EQUIPMENT\_ID attributes is the primary key of this table.

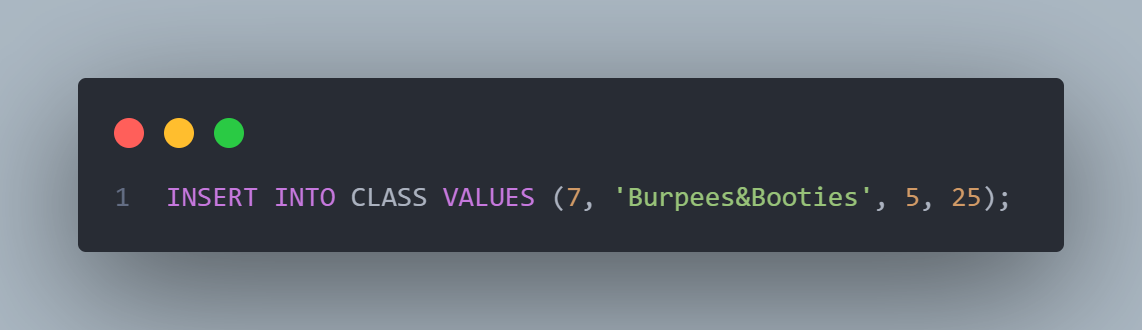
Additionally, we use FOREIGN KEY (U\_EQUIPMENT\_ID) REFERENCES EQUIPMENT(Eq\_ID), this tells SQL that the field U\_EQUIPMENT\_ID is a foreign key for the Eq\_ID field in the table for EQUIPMENT.

Below are all the commands to create the tables in our Database

CREATE TABLE BRANCH (B\_ID INT NOT NULL PRIMARY KEY, Mng\_ID INT NOT NULL, Phone INT, Location VARCHAR(255));  
  
CREATE TABLE CLASS( C\_Section INT NOT NULL PRIMARY KEY, C\_Name VARCHAR(255) NOT NULL, C\_Branch\_ID INT NOT NULL, Capacity INT );  
  
CREATE TABLE EQUIPMENT ( Eq\_ID INT NOT NULL PRIMARY KEY, Branchh\_ID INT NOT NULL , In\_Use NUMBER(1) NOT NULL, Condition VARCHAR(255), E\_Name VARCHAR(255) );  
  
CREATE TABLE FEE (FEE\_ID INT NOT NULL PRIMARY KEY, F\_Paid NUMBER(1) NOT NULL, Amnt\_Due DECIMAL, Date\_Due DATE, Pay\_Method INT);  
  
CREATE TABLE MEMBER (M\_Username VARCHAR(50) PRIMARY KEY, Gender VARCHAR(6), Birthday DATE, M\_FName VARCHAR(20) NOT NULL, M\_LName VARCHAR(20) NOT NULL, M\_Address VARCHAR(255), M\_Password VARCHAR(20) NOT NULL, BNC\_ID INT NOT NULL, Emg\_Contact INT, M\_Phone INT);  
  
CREATE TABLE MEMBERSHIP (MS\_ID INT NOT NULL PRIMARY KEY, M\_Type VARCHAR(255), Expiry\_Date DATE, Start\_Date DATE, Consecutive INT, Cost INT, MS\_Username VARCHAR(100));  
  
  
CREATE TABLE SCHEDULE (Sched\_ID INT NOT NULL PRIMARY KEY, Start\_Time TIMESTAMP WITH TIME ZONE, End\_Time TIMESTAMP WITH TIME ZONE);  
  
CREATE TABLE SELLABLE (Sellable\_ID INT NOT NULL PRIMARY KEY, Qnt\_Storage INT, Qnt\_Available INT NOT NULL, Price FLOAT NOT NULL);  
  
CREATE TABLE SHOP ( SHOP\_ID INT NOT NULL PRIMARY KEY, BestSeller\_ID INT NOT NULL );  
  
CREATE TABLE AVAILABLE\_IN (Shop\_ID INT NOT NULL, Sellable\_ID INT NOT NULL ,  PRIMARY KEY (Shop\_ID, Sellable\_ID), FOREIGN KEY (Shop\_ID) REFERENCES SHOP (SHOP\_ID), FOREIGN KEY (Sellable\_ID) REFERENCES SELLABLE (Sellable\_ID));  
  
CREATE TABLE BODY\_COMPOSITION (BCDate DATE NOT NULL, Weight INT, Height INT, PBodyFat INT, PMuscleMass INT, M\_ID VARCHAR(20) NOT NULL, PRIMARY KEY (BCDate, M\_ID), FOREIGN KEY (M\_ID) REFERENCES MEMBER (M\_Username));  
  
CREATE TABLE BRANCH\_PHONE (Branch\_ID INT NOT NULL, Phone CHAR(8) NOT NULL, FOREIGN KEY (Branch\_ID) REFERENCES BRANCH(B\_ID));  
  
CREATE TABLE EMPLOYEE( E\_ID int NOT NULL PRIMARY KEY, E\_Gender VARCHAR(255) , E\_Age INT, E\_Salary DECIMAL , E\_FName VARCHAR(255) NOT NULL, E\_LName varchar(255) NOT NULL, E\_Address varchar(255), E\_Shop\_ID INT NOT NULL, Keeping\_Days VARCHAR(250), BB\_ID INT NOT NULL, Super\_ID INT, C\_ID INT NOT NULL, FOREIGN KEY (C\_ID) REFERENCES CLASS(C\_Section), FOREIGN KEY (BB\_ID) REFERENCES BRANCH (B\_ID), FOREIGN KEY (Super\_ID) REFERENCES EMPLOYEE (E\_ID) );  
  
CREATE TABLE EMPLOYEE\_PHONE (Employee\_ID INT NOT NULL, Phone CHAR(8) NOT NULL, PRIMARY KEY (Employee\_ID, Phone), FOREIGN KEY (Employee\_ID) REFERENCES EMPLOYEE (E\_ID));  
  
CREATE TABLE EMPLOYEE\_SHIFT (Employee\_ID INT NOT NULL, Shift VARCHAR(5) NOT NULL, PRIMARY KEY (Employee\_ID, Shift), FOREIGN KEY (Employee\_ID) REFERENCES EMPLOYEE(E\_ID));  
  
CREATE TABLE FOLLOWS (Cc\_ID NOT NULL PRIMARY KEY, Scched\_ID INT NOT NULL, FOREIGN KEY (Scched\_ID) REFERENCES SCHEDULE (Sched\_ID), FOREIGN KEY (Cc\_ID) REFERENCES CLASS (C\_Section));  
  
CREATE TABLE PAYS (Username VARCHAR(15) NOT NULL, Fee\_ID INT NOT NULL, PRIMARY KEY (Username, Fee\_ID), FOREIGN KEY (Username) REFERENCES MEMBER (M\_Username), FOREIGN KEY (Fee\_ID) REFERENCES FEE (FEE\_ID));  
  
CREATE TABLE TAKES\_PLACE\_IN (Branch\_ID INT NOT NULL, Shop\_ID INT NOT NULL, PRIMARY KEY (Branch\_ID, Shop\_ID), FOREIGN KEY (Branch\_ID) REFERENCES BRANCH (B\_ID), FOREIGN KEY (Shop\_ID) REFERENCES SHOP (SHOP\_ID));  
  
CREATE TABLE USES (Username VARCHAR(15) NOT NULL, Equipment\_ID INT NOT NULL, PRIMARY KEY (Username, Equipment\_ID), FOREIGN KEY (Username) REFERENCES MEMBER (M\_Username), FOREIGN KEY (Equipment\_ID) REFERENCES EQUIPMENT (Eq\_ID));​

***VIII – Populating the Tables in Oracle Express***

In general, populating the tables was done using the INSERT command. Below is an example:



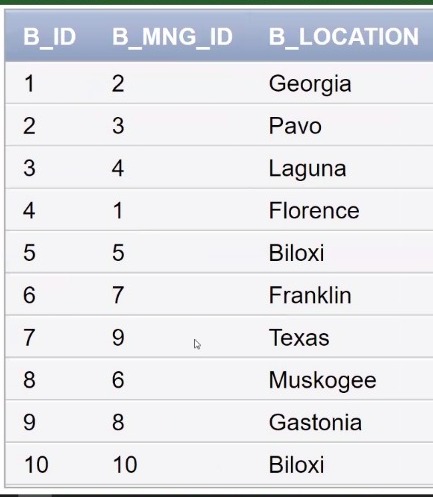
INSERT INTO CLASS specifies that we’re inserting a tuple into the CLASS table

VALUES () marks what are the values for each of the tuple will be

7, 'Burpees&Booties', 5, 25 are the 2 values for this table. In this case, representing the Class Section ID, Name of the Class, Class’s Branch ID, and Class Capacity.

Below are all the commands we used to insert the values into our Database!

BRANCH:

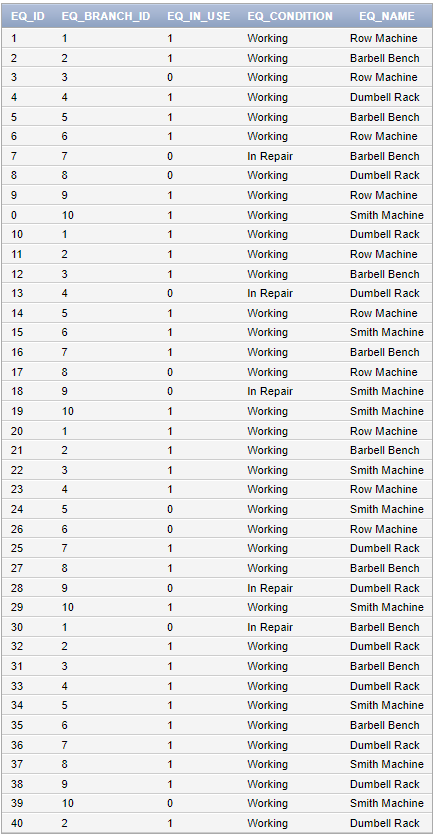
INSERT INTO BRANCH VALUES (1,2 , 96816330, 'Georgia');  
INSERT INTO BRANCH VALUES (2, 3, 6816331, 'Pavo');  
INSERT INTO BRANCH VALUES (3, 4, 96816332, 'Laguna');  
INSERT INTO BRANCH VALUES (4, 1, 96816333, 'Florence');  
INSERT INTO BRANCH VALUES (5, 5,96816334, 'Biloxi');  
INSERT INTO BRANCH VALUES (6, 7, 96816335,'Franklin');  
INSERT INTO BRANCH VALUES (7, 9, 96816336, 'Texas');  
INSERT INTO BRANCH VALUES (8, 6, 96816337, 'Muskogee');  
INSERT INTO BRANCH VALUES (9, 8, 96816338, 'Gastonia');  
INSERT INTO BRANCH VALUES (10, 10, 96816339, 'Biloxi');

CLASS:

INSERT INTO CLASS VALUES (1,'Zumba', 1,15);  
INSERT INTO CLASS VALUES (2,'Abs\_Fab',2,20);  
INSERT INTO CLASS VALUES (3, 'Move\_it\_Shake\_it\_Lift\_it', 3,25);  
INSERT INTO CLASS VALUES (4, 'Werk\_it', 4,15);  
INSERT INTO CLASS VALUES (5, 'Power\_up', 1,15);  
INSERT INTO CLASS VALUES (6, 'Stop\_Drop\_Flow ',7 ,20);  
INSERT INTO CLASS VALUES (7, 'Burpees&Booties',5,25);  
INSERT INTO CLASS VALUES (8, 'Just\_peachy',6,10);  
INSERT INTO CLASS VALUES (9, 'Cardio', 9,15);  
INSERT INTO CLASS VALUES (10, 'Sparkle+Sweat' , 2,20);



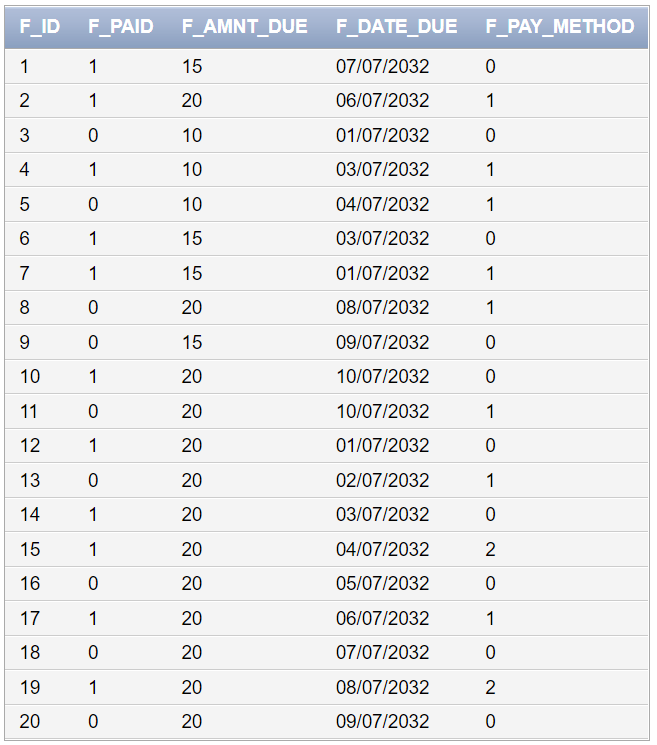
EQIPMENT:

INSERT INTO EQUIPMENT VALUES (1, 1, 1, 'Working', 'Row Machine');  
INSERT INTO EQUIPMENT VALUES (2, 2, 1, 'Working', 'Barbell Bench');  
INSERT INTO EQUIPMENT VALUES (3, 3, 0, 'Working', 'Row Machine');  
INSERT INTO EQUIPMENT VALUES (4, 4, 1, 'Working', 'Dumbell Rack');  
INSERT INTO EQUIPMENT VALUES (5, 5, 1, 'Working', 'Barbell Bench');  
INSERT INTO EQUIPMENT VALUES (6, 6, 1, 'Working', 'Row Machine');  
INSERT INTO EQUIPMENT VALUES (7, 7, 0, 'In Repair', 'Barbell Bench');  
INSERT INTO EQUIPMENT VALUES (8, 8, 0, 'Working', 'Dumbell Rack');  
INSERT INTO EQUIPMENT VALUES (9, 9, 1, 'Working', 'Row Machine');  
INSERT INTO EQUIPMENT VALUES (0, 10, 1, 'Working', 'Smith Machine');  
INSERT INTO EQUIPMENT VALUES (10, 1, 1, 'Working', 'Dumbell Rack');  
INSERT INTO EQUIPMENT VALUES (11, 2, 1, 'Working', 'Row Machine');  
INSERT INTO EQUIPMENT VALUES (12, 3, 1, 'Working', 'Barbell Bench');  
INSERT INTO EQUIPMENT VALUES (13, 4, 0, 'In Repair', 'Dumbell Rack');  
INSERT INTO EQUIPMENT VALUES (14, 5, 1, 'Working', 'Row Machine');  
INSERT INTO EQUIPMENT VALUES (15, 6, 1, 'Working', 'Smith Machine');  
INSERT INTO EQUIPMENT VALUES (16, 7, 1, 'Working', 'Barbell Bench');  
INSERT INTO EQUIPMENT VALUES (17, 8, 0, 'Working', 'Row Machine');  
INSERT INTO EQUIPMENT VALUES (18, 9, 0, 'In Repair', 'Smith Machine');  
INSERT INTO EQUIPMENT VALUES (19, 10, 1, 'Working', 'Smith Machine');  
INSERT INTO EQUIPMENT VALUES (20, 1, 1, 'Working', 'Row Machine');  
INSERT INTO EQUIPMENT VALUES (21, 2, 1, 'Working', 'Barbell Bench');  
INSERT INTO EQUIPMENT VALUES (22, 3, 1, 'Working', 'Smith Machine');  
INSERT INTO EQUIPMENT VALUES (23, 4, 1, 'Working', 'Row Machine');  
INSERT INTO EQUIPMENT VALUES (24, 5, 0, 'Working', 'Smith Machine');  
INSERT INTO EQUIPMENT VALUES (26, 6, 0, 'Working', 'Row Machine');  
INSERT INTO EQUIPMENT VALUES (25, 7, 1, 'Working', 'Dumbell Rack');  
INSERT INTO EQUIPMENT VALUES (27, 8, 1, 'Working', 'Barbell Bench');  
INSERT INTO EQUIPMENT VALUES (28, 9, 0, 'In Repair', 'Dumbell Rack');  
INSERT INTO EQUIPMENT VALUES (29, 10, 1, 'Working', 'Smith Machine');  
INSERT INTO EQUIPMENT VALUES (30, 1, 0, 'In Repair', 'Barbell Bench');  
INSERT INTO EQUIPMENT VALUES (32, 2, 1, 'Working', 'Dumbell Rack');  
INSERT INTO EQUIPMENT VALUES (31, 3, 1, 'Working', 'Barbell Bench');  
INSERT INTO EQUIPMENT VALUES (33, 4, 1, 'Working', 'Dumbell Rack');  
INSERT INTO EQUIPMENT VALUES (34, 5, 1, 'Working', 'Smith Machine');  
INSERT INTO EQUIPMENT VALUES (35, 6, 1, 'Working', 'Barbell Bench');  
INSERT INTO EQUIPMENT VALUES (36, 7, 1, 'Working', 'Dumbell Rack');  
INSERT INTO EQUIPMENT VALUES (37, 8, 1, 'Working', 'Smith Machine');  
INSERT INTO EQUIPMENT VALUES (38, 9, 1, 'Working', 'Dumbell Rack');  
INSERT INTO EQUIPMENT VALUES (39, 10, 0, 'Working', 'Smith Machine');  
INSERT INTO EQUIPMENT VALUES (40, 2, 1, 'Working', 'Dumbell Rack');  


FEE:

INSERT INTO FEE VALUES (2, 1, 20, TO\_DATE('07-JUN-2032', 'DD-MON-YYYY'), 1);  
INSERT INTO FEE VALUES (3, 0, 10, TO\_DATE('07-JAN-2032', 'DD-MON-YYYY'), 0);  
INSERT INTO FEE VALUES (4, 1, 10, TO\_DATE('07-MAR-2032', 'DD-MON-YYYY'), 1);  
INSERT INTO FEE VALUES (5, 0, 10, TO\_DATE('07-APR-2032', 'DD-MON-YYYY'), 1);  
INSERT INTO FEE VALUES (6, 1, 15, TO\_DATE('07-MAR-2032', 'DD-MON-YYYY'), 0);  
INSERT INTO FEE VALUES (7, 1, 15, TO\_DATE('07-JAN-2032', 'DD-MON-YYYY'), 1);  
INSERT INTO FEE VALUES (8, 0, 20, TO\_DATE('07-AUG-2032', 'DD-MON-YYYY'), 1);  
INSERT INTO FEE VALUES (9, 0, 15, TO\_DATE('07-SEP-2032', 'DD-MON-YYYY'), 0);  
INSERT INTO FEE VALUES (10, 1, 20, TO\_DATE('07-OCT-2032', 'DD-MON-YYYY'), 0);  
INSERT INTO FEE VALUES (11, 0, 20, TO\_DATE('07-OCT-2032', 'DD-MON-YYYY'), 1);  
INSERT INTO FEE VALUES (12, 1, 20, TO\_DATE('07-JAN-2032', 'DD-MON-YYYY'), 0);  
INSERT INTO FEE VALUES (13, 0, 20, TO\_DATE('07-FEB-2032', 'DD-MON-YYYY'), 1);  
INSERT INTO FEE VALUES (14, 1, 20, TO\_DATE('07-MAR-2032', 'DD-MON-YYYY'), 0);  
INSERT INTO FEE VALUES (15, 1, 20, TO\_DATE('07-APR-2032', 'DD-MON-YYYY'), 2);  
INSERT INTO FEE VALUES (16, 0, 20, TO\_DATE('07-MAY-2032', 'DD-MON-YYYY'), 0);  
INSERT INTO FEE VALUES (17, 1, 20, TO\_DATE('07-JUN-2032', 'DD-MON-YYYY'), 1);  
INSERT INTO FEE VALUES (18, 0, 20, TO\_DATE('07-JUL-2032', 'DD-MON-YYYY'), 0);  
INSERT INTO FEE VALUES (19, 1, 20, TO\_DATE('07-AUG-2032', 'DD-MON-YYYY'), 2);  
INSERT INTO FEE VALUES (20, 0, 20, TO\_DATE('07-SEP-2032', 'DD-MON-YYYY'), 0);

FEE (CONT.D)

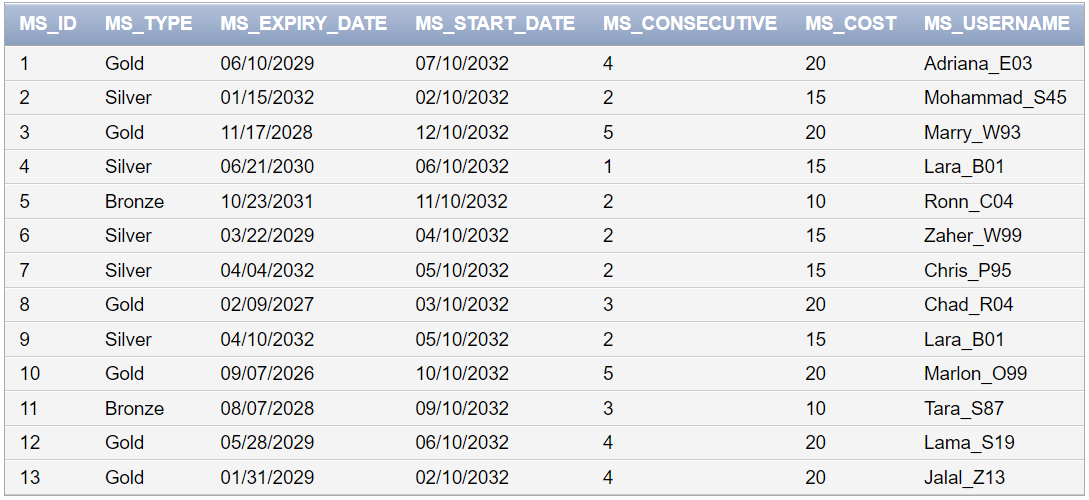


MEMBER:

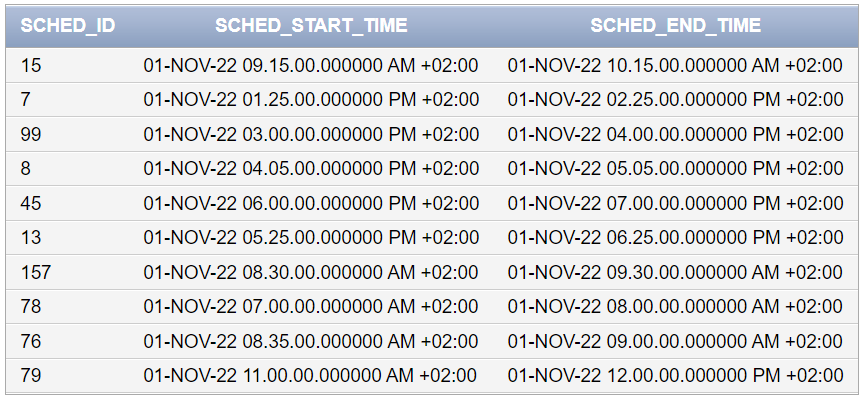
INSERT INTO MEMBER VALUES ('Marry\_W93', 'F', TO\_DATE('05-JUL-1995', 'DD-MON-YYYY'), 'Marry', 'Walt', 'Jbeil', 'DVbgUK64', 1, 79414211, 78242891);  
INSERT INTO MEMBER VALUES ('Adriana\_E03', 'F', TO\_DATE('04-APR-2005', 'DD-MON-YYYY'), 'Adriana', 'Edison', 'Achrafieh', 'J3bw3nq5', 2, 27268761,53191091);  
INSERT INTO MEMBER VALUES ('Marlon\_O99', 'M', TO\_DATE('25-JUN-1998', 'DD-MON-YYYY'), 'Marlon', 'Osborn', 'Beqaa', '7YzzMDWq', 3, 31149362, 45242661);  
INSERT INTO MEMBER VALUES ('Tara\_S87', 'F', TO\_DATE('15-MAY-2010', 'DD-MON-YYYY'), 'Tara', 'Shukriya', 'Baalbak', 'XmH2AgR8', 4, 61897856, 11755762);  
INSERT INTO MEMBER VALUES ('Chad\_R04', 'M',TO\_DATE('15-MAR-2003', 'DD-MON-YYYY'), 'Chad', 'Rod', 'Hamra', 'DUCvQvNZ', 5, 14788146, 40104356);  
INSERT INTO MEMBER VALUES ('Zaher\_W99', 'M', TO\_DATE('15-JUL-2009', 'DD-MON-YYYY'), 'Zaher', 'Wafi', 'Verdun', 'h3vfVmqA', 6, 53001744, 91602623);  
INSERT INTO MEMBER VALUES ('Patrice\_L00', 'F', TO\_DATE('25-AUG-2000', 'DD-MON-YYYY'), 'Patrice', 'Lecia', 'Quraitim', 'NM7ZAvpG', 7, 84326053, 10309786);  
INSERT INTO MEMBER VALUES ('Lara\_B01', 'F',TO\_DATE('01-NOV-2001', 'DD-MON-YYYY'), 'Lara' ,'Bell', 'Barbir', 'LbS2rAa2', 8, 32948707, 12627473);  
INSERT INTO MEMBER VALUES ('Ronn\_C04', 'M', TO\_DATE('05-DEC-2008', 'DD-MON-YYYY'), 'Ronn', 'Carey', 'Dahye', 'WknCYe8h', 9, 16938588, 80640854);  
INSERT INTO MEMBER VALUES ('Chris\_P95', 'M',TO\_DATE('05-OCT-2015', 'DD-MON-YYYY'), 'Chris' , 'Payton', 'Jnah', 'PR3GkzIj', 10, 70948979, 96816332);  
INSERT INTO MEMBER VALUES ('Jalal\_Z13', 'M',TO\_DATE('13-JUL-2003', 'DD-MON-YYYY'), 'Jalal' ,'El-Zein', 'Chiyah', 'KsPSpqE', 7, 03654371, 79164777);  
INSERT INTO MEMBER VALUES ('Lama\_S19', 'F',TO\_DATE('01-APR-2003', 'DD-MON-YYYY'), 'Lama' ,'AL-Sheikh', 'Dahye', 'sDfgAoiGh', 8, 78872569, 71531928);  
INSERT INTO MEMBER VALUES ('Mohammad\_S45', 'M',TO\_DATE('10-JUN-2020', 'DD-MON-YYYY'), 'Chris' ,'Payton', 'Jnah', 'PR3GkzIj', 4, 76923204, 01825406);

MEMBERSHIP:

INSERT INTO MEMBERSHIP VALUES (1, 'Gold', TO\_DATE('10-JUN-2029', 'DD-MON-YYYY'), TO\_DATE('10-JUL-2032', 'DD-MON-YYYY'), 4, 20, 'Adriana\_E03');  
INSERT INTO MEMBERSHIP VALUES (2, 'Silver', TO\_DATE('15-JAN-2032', 'DD-MON-YYYY'), TO\_DATE('10-FEB-2032', 'DD-MON-YYYY'), 2, 15, 'Mohammad\_S45');  
INSERT INTO MEMBERSHIP VALUES (3, 'Gold', TO\_DATE('17-NOV-2028', 'DD-MON-YYYY'), TO\_DATE('10-DEC-2032', 'DD-MON-YYYY'), 5, 20, 'Marry\_W93');  
INSERT INTO MEMBERSHIP VALUES (4, 'Silver', TO\_DATE('21-JUN-2030', 'DD-MON-YYYY'), TO\_DATE('10-JUN-2032', 'DD-MON-YYYY'), 1, 15, 'Lara\_B01');  
INSERT INTO MEMBERSHIP VALUES (5, 'Bronze', TO\_DATE('23-OCT-2031', 'DD-MON-YYYY'), TO\_DATE('10-NOV-2032', 'DD-MON-YYYY'), 2, 10, 'Ronn\_C04');  
INSERT INTO MEMBERSHIP VALUES (6, 'Silver', TO\_DATE('22-MAR-2029', 'DD-MON-YYYY'), TO\_DATE('10-APR-2032', 'DD-MON-YYYY'), 2, 15, 'Zaher\_W99');  
INSERT INTO MEMBERSHIP VALUES (7, 'Silver', TO\_DATE('04-APR-2032', 'DD-MON-YYYY'), TO\_DATE('10-MAY-2032', 'DD-MON-YYYY'), 2, 15, 'Chris\_P95');  
INSERT INTO MEMBERSHIP VALUES (8, 'Gold', TO\_DATE('09-FEB-2027', 'DD-MON-YYYY'), TO\_DATE('10-MAR-2032', 'DD-MON-YYYY'), 3, 20, 'Chad\_R04');  
INSERT INTO MEMBERSHIP VALUES (9, 'Silver', TO\_DATE('10-APR-2032', 'DD-MON-YYYY'), TO\_DATE('10-MAY-2032', 'DD-MON-YYYY'), 2, 15, 'Lara\_B01');  
INSERT INTO MEMBERSHIP VALUES (10, 'Gold', TO\_DATE('07-SEP-2026', 'DD-MON-YYYY'), TO\_DATE('10-OCT-2032', 'DD-MON-YYYY'), 5, 20, 'Marlon\_O99');  
INSERT INTO MEMBERSHIP VALUES (11, 'Bronze', TO\_DATE('07-AUG-2028', 'DD-MON-YYYY'), TO\_DATE('10-SEP-2032', 'DD-MON-YYYY'), 3, 10, 'Tara\_S87');  
INSERT INTO MEMBERSHIP VALUES (12, 'Gold', TO\_DATE('28-MAY-2029', 'DD-MON-YYYY'), TO\_DATE('10-JUN-2032', 'DD-MON-YYYY'), 4, 20, 'Lama\_S19');  
INSERT INTO MEMBERSHIP VALUES (13, 'Gold', TO\_DATE('31-JAN-2029', 'DD-MON-YYYY'), TO\_DATE('10-FEB-2032', 'DD-MON-YYYY'), 4, 20, 'Jalal\_Z13');

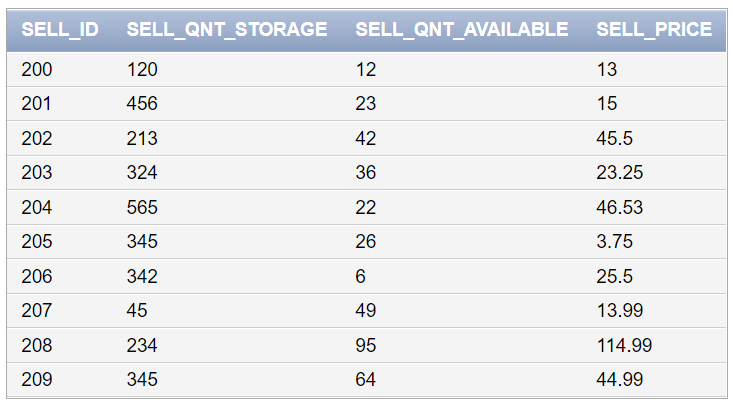


SCHEDULE:  
INSERT INTO SCHEDULE VALUES (15,(to\_date('9:15:00','HH24:MI:SS')), (to\_date('10:15:00','HH24:MI:SS')));  
INSERT INTO SCHEDULE VALUES (7,(to\_date('13:25:00','HH24:MI:SS')), (to\_date('14:25:00','HH24:MI:SS')));  
INSERT INTO SCHEDULE VALUES (99,(to\_date('15:00:00','HH24:MI:SS')), (to\_date('16:00:00','HH24:MI:SS')));  
INSERT INTO SCHEDULE VALUES (8,(to\_date('16:05:00:00','HH24:MI:SS')), (to\_date('17:05:00','HH24:MI:SS')));  
INSERT INTO SCHEDULE VALUES (45,(to\_date('18:00:00','HH24:MI:SS')), (to\_date('19:00:00','HH24:MI:SS')));  
INSERT INTO SCHEDULE VALUES (13,(to\_date('17:25:00','HH24:MI:SS')), (to\_date('18:25:00','HH24:MI:SS')));  
INSERT INTO SCHEDULE VALUES (157,(to\_date('8:30:00','HH24:MI:SS')), (to\_date('9:30:00','HH24:MI:SS')));  
INSERT INTO SCHEDULE VALUES (78,(to\_date('7:00:00','HH24:MI:SS')), (to\_date('8:00:00','HH24:MI:SS')));  
INSERT INTO SCHEDULE VALUES (76,(to\_date('8:35:00','HH24:MI:SS')), (to\_date('9:00:00','HH24:MI:SS')));  
INSERT INTO SCHEDULE VALUES (79,(to\_date('11:00:00','HH24:MI:SS')), (to\_date('12:00:00','HH24:MI:SS')));



SELLABLE:

INSERT INTO SELLABLE VALUES (200, 120, 12, 13.00);  
INSERT INTO SELLABLE VALUES (201, 456, 23, 15.00);  
INSERT INTO SELLABLE VALUES (202, 213, 42, 45.50);  
INSERT INTO SELLABLE VALUES (203, 324, 36, 23.25);  
INSERT INTO SELLABLE VALUES (204, 565, 22, 46.53);  
INSERT INTO SELLABLE VALUES (205, 345, 26, 3.75);  
INSERT INTO SELLABLE VALUES (206, 342, 6, 25.50);  
INSERT INTO SELLABLE VALUES (207, 45, 49, 13.99);  
INSERT INTO SELLABLE VALUES (208, 234, 95, 114.99);  
INSERT INTO SELLABLE VALUES (209, 345, 64, 44.99);



SHOP:

INSERT INTO SHOP VALUES (1, 203);  
INSERT INTO SHOP VALUES (2, 204);  
INSERT INTO SHOP VALUES (3, 205);  
INSERT INTO SHOP VALUES (4, 206);  
INSERT INTO SHOP VALUES (5, 203);  
INSERT INTO SHOP VALUES (7, 207);  
INSERT INTO SHOP VALUES (6, 206);  
INSERT INTO SHOP VALUES (8, 204);  
INSERT INTO SHOP VALUES (9, 206);  
INSERT INTO SHOP VALUES (0, 202);



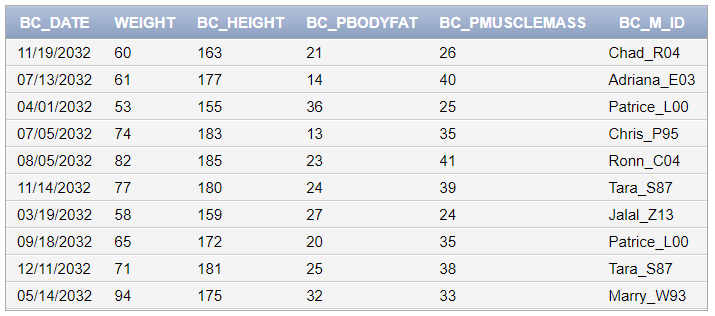
AVAILABLE\_IN:



INSERT INTO AVAILABLE\_IN VALUES (0, 202);  
INSERT INTO AVAILABLE\_IN VALUES (0, 205);  
INSERT INTO AVAILABLE\_IN VALUES (0, 206);  
INSERT INTO AVAILABLE\_IN VALUES (1, 203);  
INSERT INTO AVAILABLE\_IN VALUES (1, 201);  
INSERT INTO AVAILABLE\_IN VALUES (1, 202);  
INSERT INTO AVAILABLE\_IN VALUES (2, 204);  
INSERT INTO AVAILABLE\_IN VALUES (2, 205);  
INSERT INTO AVAILABLE\_IN VALUES (2, 200);  
INSERT INTO AVAILABLE\_IN VALUES (3, 205);  
INSERT INTO AVAILABLE\_IN VALUES (3, 203);  
INSERT INTO AVAILABLE\_IN VALUES (3, 209);  
INSERT INTO AVAILABLE\_IN VALUES (3, 207);  
INSERT INTO AVAILABLE\_IN VALUES (4, 205);  
INSERT INTO AVAILABLE\_IN VALUES (4, 200);  
INSERT INTO AVAILABLE\_IN VALUES (4, 206);  
INSERT INTO AVAILABLE\_IN VALUES (5, 203);  
INSERT INTO AVAILABLE\_IN VALUES (5, 207);  
INSERT INTO AVAILABLE\_IN VALUES (5, 201);  
INSERT INTO AVAILABLE\_IN VALUES (6, 206);  
INSERT INTO AVAILABLE\_IN VALUES (6, 208);  
INSERT INTO AVAILABLE\_IN VALUES (6, 209);  
INSERT INTO AVAILABLE\_IN VALUES (7, 205);  
INSERT INTO AVAILABLE\_IN VALUES (7, 204);  
INSERT INTO AVAILABLE\_IN VALUES (7, 207);  
INSERT INTO AVAILABLE\_IN VALUES (8, 204);  
INSERT INTO AVAILABLE\_IN VALUES (8, 203);  
INSERT INTO AVAILABLE\_IN VALUES (8, 202);  
INSERT INTO AVAILABLE\_IN VALUES (9, 206);  
INSERT INTO AVAILABLE\_IN VALUES (9, 208);  
INSERT INTO AVAILABLE\_IN VALUES (9, 200);

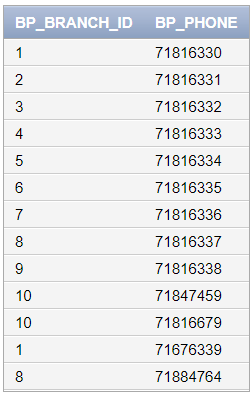
BODY\_COMPOSITION:

INSERT INTO BODY\_COMPOSITION VALUES (TO\_DATE('19-NOV-2032', 'DD-MON-YYYY'), 60, 163, 21, 26, 'Chad\_R04');  
INSERT INTO BODY\_COMPOSITION VALUES (TO\_DATE('13-JUL-2032', 'DD-MON-YYYY'), 61, 177, 14, 40, 'Adriana\_E03');  
INSERT INTO BODY\_COMPOSITION VALUES    (TO\_DATE('01-APR-2032', 'DD-MON-YYYY'), 53, 155, 36, 25, 'Patrice\_L00');  
INSERT INTO BODY\_COMPOSITION VALUES (TO\_DATE('05-JUL-2032', 'DD-MON-YYYY'), 74, 183, 13, 35, 'Chris\_P95');  
INSERT INTO BODY\_COMPOSITION VALUES (TO\_DATE('05-AUG-2032', 'DD-MON-YYYY'), 82, 185, 23, 41, 'Ronn\_C04');  
INSERT INTO BODY\_COMPOSITION VALUES (TO\_DATE('14-NOV-2032', 'DD-MON-YYYY'), 77, 180, 24, 39, 'Tara\_S87');  
INSERT INTO BODY\_COMPOSITION VALUES (TO\_DATE('19-NOV-2032', 'DD-MON-YYYY'), 58, 159, 27, 24, 'Jalal\_Z13');  
INSERT INTO BODY\_COMPOSITION VALUES (TO\_DATE('18-NOV-2032', 'DD-MON-YYYY'), 65, 172, 20, 35, 'Patrice\_L00');  
INSERT INTO BODY\_COMPOSITION VALUES    (TO\_DATE('11-NOV-2032', 'DD-MON-YYYY'), 71, 181, 25, 38, 'Tara\_S87');  
INSERT INTO BODY\_COMPOSITION VALUES (TO\_DATE('14-NOV-2032', 'DD-MON-YYYY'), 94, 175, 32, 33, 'Marry\_W93');



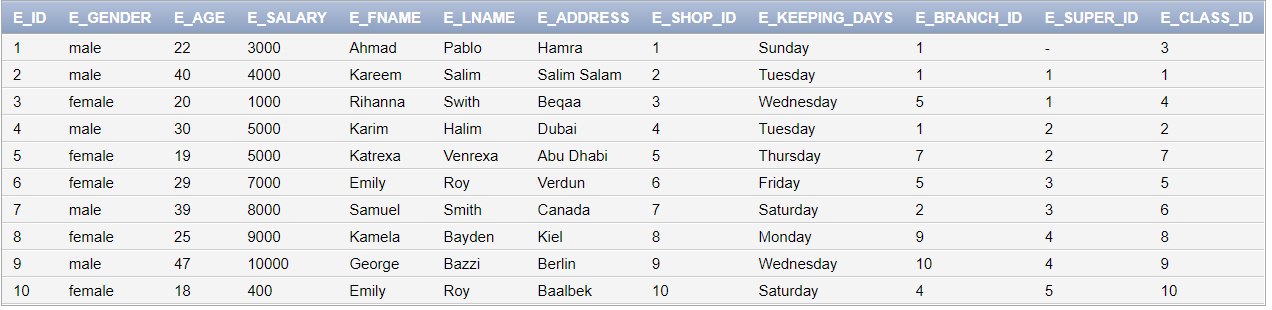
BRANCH\_PHONE:

INSERT INTO BRANCH\_PHONE VALUES (1, 96816330);  
INSERT INTO BRANCH\_PHONE VALUES (2, 96816331);  
INSERT INTO BRANCH\_PHONE VALUES (3, 96816332);  
INSERT INTO BRANCH\_PHONE VALUES (4, 96816333);  
INSERT INTO BRANCH\_PHONE VALUES (5, 96816334);  
INSERT INTO BRANCH\_PHONE VALUES (6, 96816335);  
INSERT INTO BRANCH\_PHONE VALUES (7, 96816336);  
INSERT INTO BRANCH\_PHONE VALUES (8, 96816337);  
INSERT INTO BRANCH\_PHONE VALUES (9, 96816338);  
INSERT INTO BRANCH\_PHONE VALUES (10, 96847459);  
INSERT INTO BRANCH\_PHONE VALUES (10, 96816679);  
INSERT INTO BRANCH\_PHONE VALUES (1, 96676339);  
INSERT INTO BRANCH\_PHONE VALUES (8, 96884764);



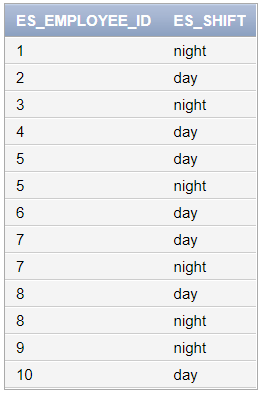
EMPLOYEE:

INSERT INTO EMPLOYEE VALUES (1, 'male', 22, 3000, 'Ahmad', 'Pablo', 'Hamra', 1, 'Sunday', 1, NULL, 3);  
INSERT INTO EMPLOYEE VALUES (2, 'male', 40, 4000, 'Kareem', 'Salim', 'Salim Salam', 2, 'Tuesday', 1, 1, 1);  
INSERT INTO EMPLOYEE VALUES (3, 'female', 20, 1000, 'Rihanna', 'Swith', 'Beqaa', 3, 'Wednesday', 5, 1, 4);  
INSERT INTO EMPLOYEE VALUES (4, 'male', 30, 5000, 'Karim', 'Halim', 'Dubai', 4, 'Tuesday', 1, 2, 2);  
INSERT INTO EMPLOYEE VALUES (5, 'female', 19, 5000, 'Katrexa', 'Venrexa', 'Abu Dhabi', 5, 'Thursday', 7, 2, 7);  
INSERT INTO EMPLOYEE VALUES (6, 'female', 29, 7000, 'Emily', 'Roy', 'Verdun', 6, 'Friday', 5, 3, 5);  
INSERT INTO EMPLOYEE VALUES (7, 'male', 39, 8000, 'Samuel', 'Smith', 'Canada', 7, 'Saturday', 2, 3, 6);  
INSERT INTO EMPLOYEE VALUES (8, 'female', 25, 9000, 'Kamela', 'Bayden', 'Kiel', 8, 'Monday', 9, 4, 8);  
INSERT INTO EMPLOYEE VALUES (9, 'male', 47, 10000, 'George', 'Bazzi', 'Berlin', 9, 'Wednesday', 10, 4, 9);  
INSERT INTO EMPLOYEE VALUES (10, 'female', 18, 400, 'Emily', 'Roy', 'Baalbek', 10, 'Saturday', 4, 5, 10);

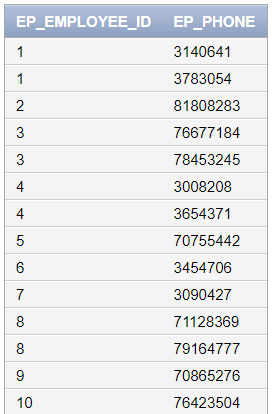


EMPLOYEE\_SHIFT:

INSERT INTO EMPLOYEE\_SHIFT VALUES (2,'day' );  
INSERT INTO EMPLOYEE\_SHIFT VALUES (3, 'night');  
INSERT INTO EMPLOYEE\_SHIFT VALUES (10, 'day');  
INSERT INTO EMPLOYEE\_SHIFT VALUES (1, 'night');  
INSERT INTO EMPLOYEE\_SHIFT VALUES (9, 'night');  
INSERT INTO EMPLOYEE\_SHIFT VALUES (7, 'day');  
INSERT INTO EMPLOYEE\_SHIFT VALUES (8, 'night');  
INSERT INTO EMPLOYEE\_SHIFT VALUES (5, 'night');  
INSERT INTO EMPLOYEE\_SHIFT VALUES (4, 'day');  
INSERT INTO EMPLOYEE\_SHIFT VALUES (8, 'night');  
INSERT INTO EMPLOYEE\_SHIFT VALUES (5, 'day');  
INSERT INTO EMPLOYEE\_SHIFT VALUES (3, 'night');  
INSERT INTO EMPLOYEE\_SHIFT VALUES (6, 'day');

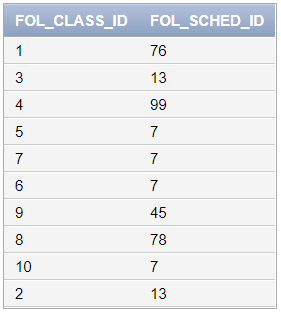


EMPLOYEE\_PHONE:

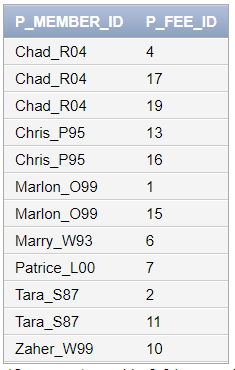
INSERT INTO EMPLOYEE\_PHONE VALUES (2, 81808283);  
INSERT INTO EMPLOYEE\_PHONE VALUES (4, 03008208);  
INSERT INTO EMPLOYEE\_PHONE VALUES (6, 03454706);  
INSERT INTO EMPLOYEE\_PHONE VALUES (8, 71128369);  
INSERT INTO EMPLOYEE\_PHONE VALUES (10, 76423504);  
INSERT INTO EMPLOYEE\_PHONE VALUES (9, 70865276);  
INSERT INTO EMPLOYEE\_PHONE VALUES (7, 03090427);  
INSERT INTO EMPLOYEE\_PHONE VALUES (5, 70755442);  
INSERT INTO EMPLOYEE\_PHONE VALUES (1, 03140641);  
INSERT INTO EMPLOYEE\_PHONE VALUES (3, 76677184);  
INSERT INTO EMPLOYEE\_PHONE VALUES (3, 78453245);  
INSERT INTO EMPLOYEE\_PHONE VALUES (8, 79164777);  
INSERT INTO EMPLOYEE\_PHONE VALUES (4, 03654371);  
INSERT INTO EMPLOYEE\_PHONE VALUES (1, 03783054);

FOLLOWS:

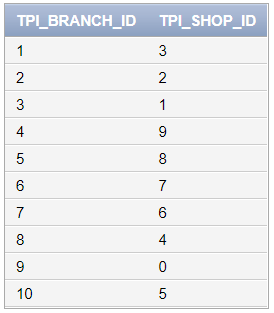
INSERT INTO FOLLOWS VALUES (1, 12);  
INSERT INTO FOLLOWS VALUES (3, 13);  
INSERT INTO FOLLOWS VALUES (3, 12);  
INSERT INTO FOLLOWS VALUES (4, 99);  
INSERT INTO FOLLOWS VALUES (5, 7);  
INSERT INTO FOLLOWS VALUES (7, 7);  
INSERT INTO FOLLOWS VALUES (6, 7);  
INSERT INTO FOLLOWS VALUES (9, 45);  
INSERT INTO FOLLOWS VALUES (8, 78);  
INSERT INTO FOLLOWS VALUES (10, 7);



PAYS:

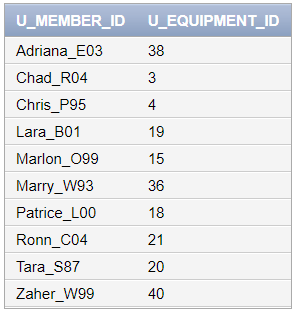
INSERT INTO PAYS VALUES ('Marlon\_O99', 1);  
INSERT INTO PAYS VALUES ('Tara\_S87', 2);  
INSERT INTO PAYS VALUES ('Chad\_R04', 4);  
INSERT INTO PAYS VALUES ('Marry\_W93', 6);  
INSERT INTO PAYS VALUES ('Patrice\_L00', 7);  
INSERT INTO PAYS VALUES ('Zaher\_W99', 10);  
INSERT INTO PAYS VALUES ('Chris\_P95', 13);  
INSERT INTO PAYS VALUES ('Tara\_S87', 11);  
INSERT INTO PAYS VALUES ('Marlon\_O99', 15);  
INSERT INTO PAYS VALUES ('Chris\_p95', 16);  
INSERT INTO PAYS VALUES ('Chad\_R04', 17);  
INSERT INTO PAYS VALUES ('Chad\_R04', 19);

TAKES\_PLACE\_IN:

INSERT INTO TAKES\_PLACE\_IN VALUES (1, 3);  
INSERT INTO TAKES\_PLACE\_IN VALUES (2, 2);  
INSERT INTO TAKES\_PLACE\_IN VALUES (3, 1);  
INSERT INTO TAKES\_PLACE\_IN VALUES (4, 9);  
INSERT INTO TAKES\_PLACE\_IN VALUES (5, 8);  
INSERT INTO TAKES\_PLACE\_IN VALUES (6, 7);  
INSERT INTO TAKES\_PLACE\_IN VALUES (7, 6);  
INSERT INTO TAKES\_PLACE\_IN VALUES (8, 4);  
INSERT INTO TAKES\_PLACE\_IN VALUES (9, 0);  
INSERT INTO TAKES\_PLACE\_IN VALUES (10, 5);

USES:

INSERT INTO USES VALUES ('Chad\_R04', 3);  
INSERT INTO USES VALUES ('Chris\_P95', 4);  
INSERT INTO USES VALUES ('Marlon\_O99', 15);  
INSERT INTO USES VALUES ('Patrice\_L00', 18);  
INSERT INTO USES VALUES ('Lara\_B01', 19);  
INSERT INTO USES VALUES ('Ronn\_C04', 21);  
INSERT INTO USES VALUES ('Tara\_S87', 20);  
INSERT INTO USES VALUES ('Zaher\_W99', 40);  
INSERT INTO USES VALUES ('Adriana\_E03', 38);  
INSERT INTO USES VALUES ('Marry\_W93', 36);​



***IX – Some Transactions on the Database***

Changing Time Zones!

Old habits die hard, and in 2032, the human race still has not figured out how to bypass daylight saving times completely, and instead adopted a new system to change time zones to UTC. So, our facilities need to accommodate for the time zone changes that happen near the end of every year to make sure our data stays relevant and intact!

For this reason, we wrote the following SQL script to change the start and end times of the schedules in our classes to UTC!

UPDATE SCHEDULE

SET Sched\_Start\_Time = CASE

WHEN SCHED\_ID < 20 THEN Sched\_Start\_Time AT TIME ZONE 'UTC'

WHEN SCHED\_ID > 20 THEN Sched\_Start\_Time

END;

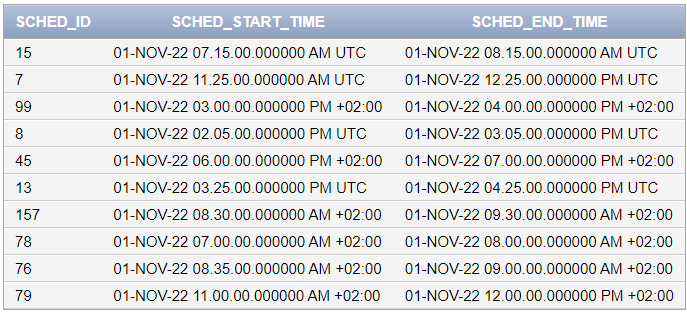
UPDATE SCHEDULE

SET Sched\_End\_Time = CASE

WHEN SCHED\_ID < 20 THEN Sched\_End\_Time AT TIME ZONE 'UTC'

WHEN SCHED\_ID > 20 THEN Sched\_End\_Time

END;



YOU GET A RAISE, AND YOU GET A RAISE, EVERYONE GETS A RAISE!



On its 6-year anniversary, MUSCLE ASSEMBLY decided to reward its hard working employees by giving them a raise! However, Shukri is the current Finances Officer and he is very greedy, so he wants to give less money for those who already make a lot of money from the company. Shukri decided to ask us for help, so we wrote the following code to make this transaction possible:

UPDATE EMPLOYEE

SET E\_Salary = CASE

WHEN E\_Salary < 5000 THEN E\_Salary + 1000

WHEN E\_Salary >= 5000 THEN E\_Salary + 550

END;



Researching Muscle Hypertrophy in Women

Being the most popular and successful fitness facility in history, researchers often approach MUSCLE ASSEMBLY with collaboration offers. This time, a certain researcher was studying muscle hypertrophy in women and needed the data for these subjects for his research. As MUSCLE ASSEMBLY highly values its customers privacy, and as all female members already gave consent to participate in this study, we decided to make a special view for the researcher to access the database, without exposing the non-consenting male customers’ data! The following SQL script makes this transaction possible:

CREATE VIEW FEMALE AS

SELECT M\_Username, M\_FName, M\_LName

FROM MEMBER

WHERE M\_Gender = 'F';



Finding the Thieves

The police approach MUSCLE ASSEMBLY with a call for help! It seems like 2 criminals that are being hunted down actually attend our gyms! The police has asked to help identify these 2 criminals, with only 1 hint, that these thieves only pay in cash because they don’t want to be detected. The problem is that these criminals are using fake names, so getting those names is useless, instead, the smart security officer at MUSCLE ASSEMBLY suggests we only track the IDs of members who have already paid in cash and report them whenever they log into the gym to be monitored! That’s why we created the following transaction:

SELECT DISTINCT F\_ID

FROM PAYS

WHERE F\_ID IN (

SELECT F\_ID

FROM FEE

WHERE F\_Paid = 1

) AND F\_ID IN (

SELECT F\_ID

FROM FEE

WHERE F\_Pay\_Method = 0

);



Who are our managers?

For the yearly global MUSCLE ASSEMBLY summit, we need to invite all the managers in our branches across the globe to a certain hotel. And that of course entails custom invitation and greeting cards that must contain the names of these managers. Unfortunately, Rayane is a very irresponsible employee and she often forgets the names of these managers, so she asked us for help to remember them. This code helps Rayane get the names of the managers in all branches:

SELECT E\_FNAME, E\_LName

FROM EMPLOYEE

WHERE EXISTS (

SELECT \*

FROM BRANCH

WHERE E\_Super\_ID = B\_Mng\_ID

) AND EXISTS (

SELECT \*

FROM BRANCH

WHERE B\_ID = E\_BRANCH\_ID

);



Keeping the Promise!

While in a meeting, the founders of MUSCLE ASSEMBLY, who were all LAU students in the past, remembered Dr. Faisal Abu Khzam told them the labs were getting expanded to accommodate a larger number of students. That did not happen during their time with Dr. Faisal, so they decided to get some closure with this open-ended ended promise and increase the capacity of their own classes that accommodate for less than 20 students on Dr. Faisal’s behalf.

SELECT C\_Name, C\_Capacity

FROM CLASS

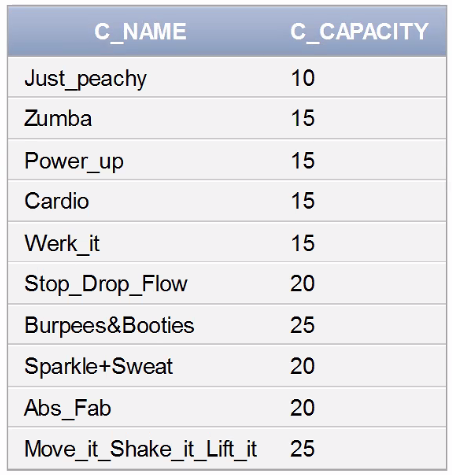
ORDER BY (

CASE WHEN C\_Capacity < 20

THEN C\_Capacity + 5

END

);



Premium Fitness Campaign

In collaboration with the World Health Organization, MUSCLE ASSEMBLY is holding an offer in hopes of motivating people who need fitness to get the most out of it! For this reason, the offer entails raising the membership types of all users who are Bronze tier and have a Body Fat Percentage that exceeds 20 to Silver Tier for free! This SQL scripts applies the membership changes:

UPDATE MEMBERSHIP

SET MS\_Type = CASE WHEN MS\_Username IN (

SELECT BC\_M\_ID

FROM BODY\_COMPOSITION

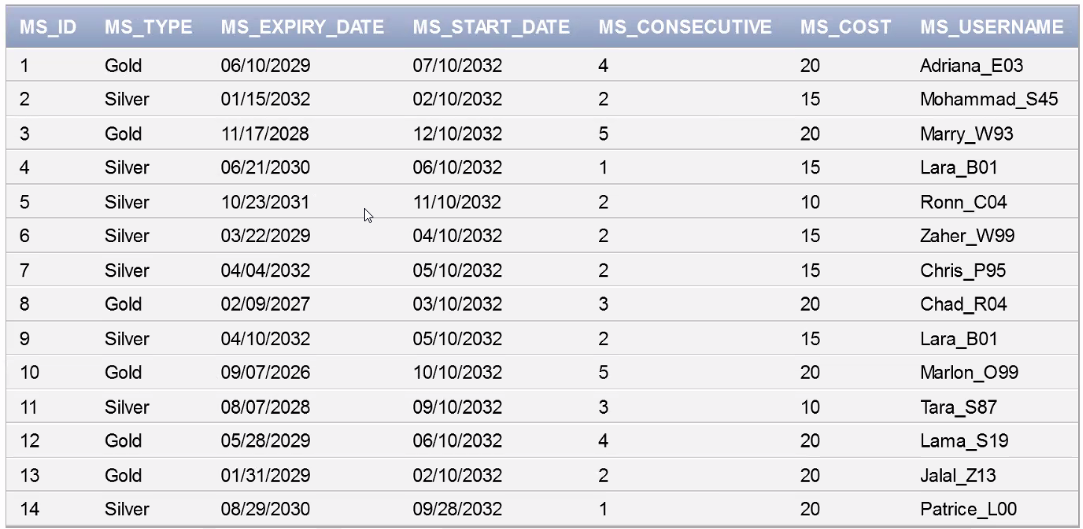
WHERE BC\_PBodyFat > 20

) AND MS\_Type = 'Bronze'

THEN 'Silver'

ELSE MS\_Type

END;



Jalal has a crush!

Jalal spotted a very beautiful woman in the gym one day, and he seems to like her and wants to talk to her. However, Jalal wants to know her name first so he can check out her Instagram before he talks to her in real life. The problem is that Jalal attends the gym irregularly and he doesn’t know her name because he doesn’t see her often enough to hear it or ask! He only knows that her name ends in “ra” and that she attends the branch of ID 8 (that’s where they met). The database engineers created this script to help him identify his gym wife:

SELECT M\_LName, M\_FName

FROM MEMBER JOIN BRANCH ON M\_Branch\_ID = B\_ID

WHERE M\_FName LIKE '%ra' AND B\_ID = 8;



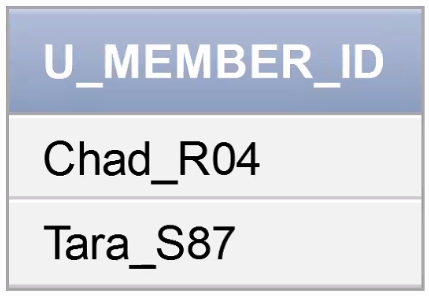
Sweaty Rowing

The staff has been getting annoyed finding the same piece of equipment, the Row Machine, drenched in sweat for several days, but they can’t find the perpetrator! They decided to search for the member through the database by checking who was the last person to use the equipment:

SELECT U\_MEMBER\_ID

FROM USES, EQUIPMENT

WHERE U\_EQUIPMENT\_ID = Eq\_ID AND Eq\_Name = 'Row Machine';



Quantity over Quality

It seems like the storage space for food is almost full, and the managers of the departments decided it was best to motivate people to buy the items that are taking up the most storage to free up space for new items. The sales team suggested that the best way to get customers buying products more often is by running sales! So the managers asked us to run a sale of 60% off on all items that exceed 300 units in storage

UPDATE SELLABLE

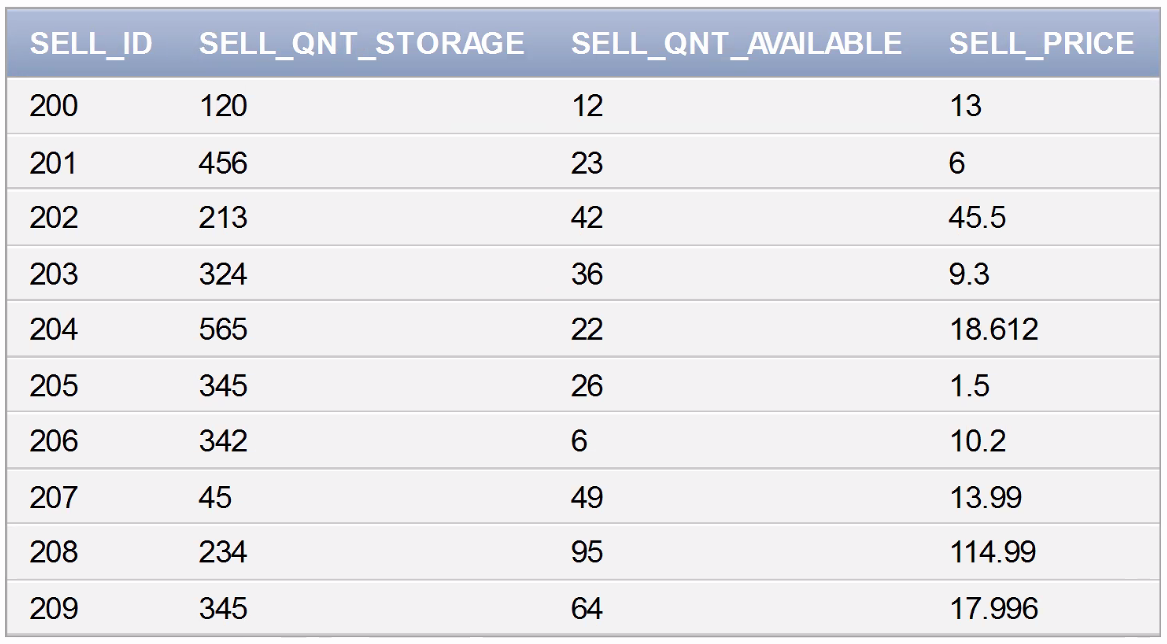
SET SELL\_Price = CASE

WHEN SELL\_Qnt\_Storage >= 300

THEN SELL\_Price \* 0.4

ELSE SELL\_Price

END;

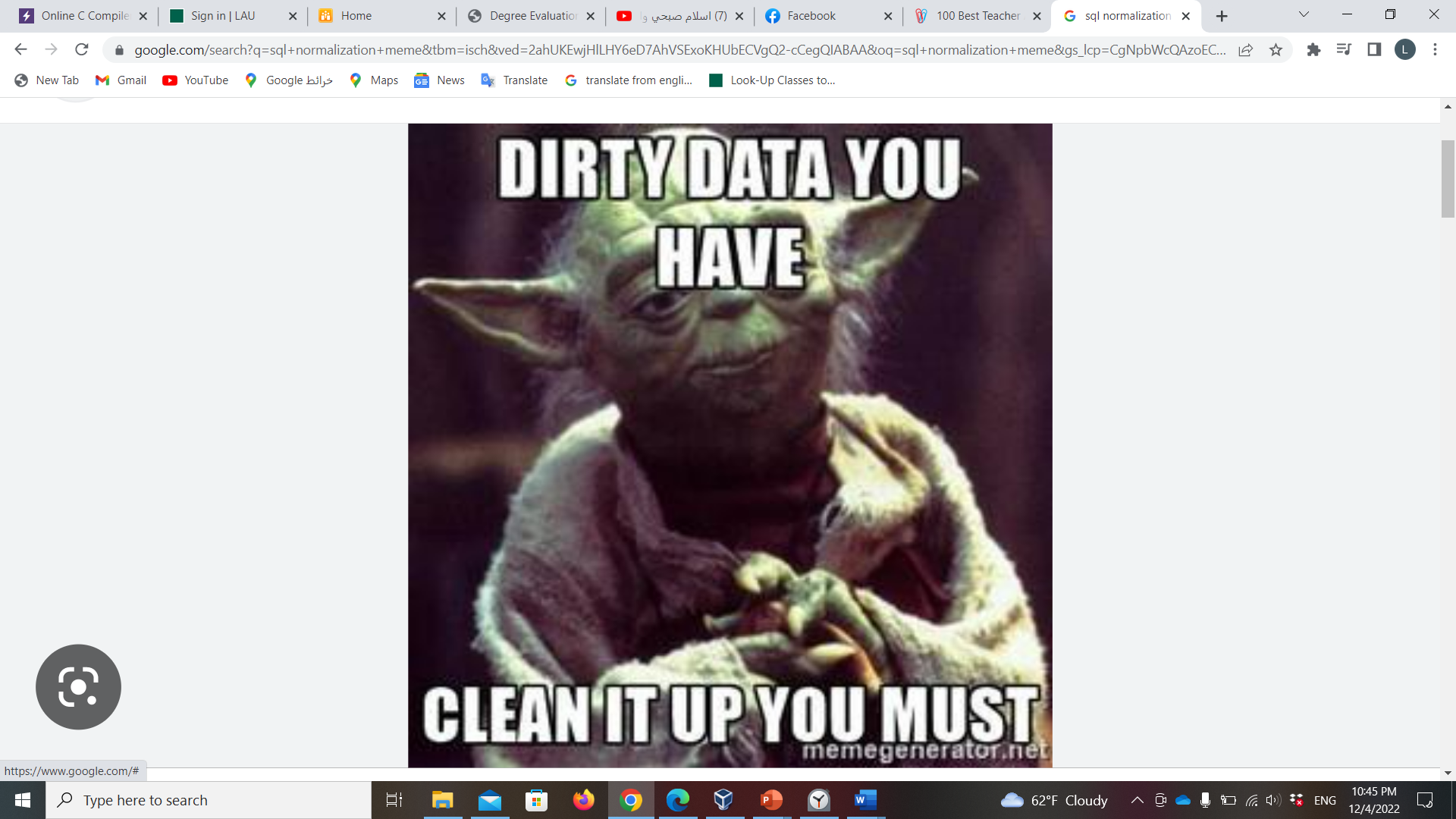


***X- Normalization***

Normalization is **the process to eliminate data redundancy and enhance data integrity in the table**. Normalization also helps to organize the data in the database. It is a multi-step process that sets the data into tabular form and removes the duplicated data from the relational tables.



*In other words, Normalization is:*



*Branch*

|  |  |  |  |
| --- | --- | --- | --- |
| ID | MNG\_ID | PHONE | LOCATION |

1. The BRANCH relation schema satisfies all the conditions of 1NF since there is no multivalued attributes or composite attributes, where all attributes are single.
2. The BRANCH relation schema satisfies all the conditions of 2NF since all attributes depends on the primary key (ID), where if ID is dropped from this relation, the functional dependency does not hold any more.
3. The BRANCH relation schema satisfies all the conditions of 3NF since it is in 2NF and there is no non-prime attribute in BRANCH relation is transitively dependent on the primary key (ID).
4. The BRANCH relation schema satisfies all the conditions of BCNF since there exists no FD X -> A, such that X is not super key.

CLASS

|  |  |  |  |
| --- | --- | --- | --- |
| Section | Name | Branch\_ ID | Capacity |

1. The CLASS relation schema satisfies all the conditions of 1NF since there is no multivalued attributes or composite attributes, where all attributes are single.
2. The CLASS relation schema satisfies all the conditions of 2NF since all attributes depends on the primary key (Section), where if SECTION is dropped from this relation, the functional dependency does not hold any more.
3. The CLASS relation schema satisfies all the conditions of 3NF since it is in 2NF and there is no non-prime attribute in BRANCH relation is transitively dependent on the primary key (Section).
4. The CLASS relation schema satisfies all the conditions of BCNF since there exists no FD X -> A, such that X is not super key.

*EMPLOYEE*

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Gender | Age | Salary | FName | LName | Address | Shop\_ID | Keeping\_Days | B\_ID | Super\_ID | C\_ID |

1. The EMPLOYEE relation schema satisfies all the conditions of 1NF since there is no multivalued attributes or composite attributes, where all attributes are single.
2. The EMPLOYEE relation schema satisfies all the conditions of 2NF since all attributes depends on the primary key (ID), where if ID is dropped from this relation, the functional dependency does not hold any more.
3. The EMPLOYEE relation schema satisfies all the conditions of 3NF since it is in 2NF and there is no non-prime attribute in BRANCH relation is transitively dependent on the primary key (ID).
4. The EMPLOYEE relation schema satisfies all the conditions of BCNF since there exists no FD X -> A, such that X is not super key.

SELLABLE

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Qnt\_Storage | Qnt\_Available | Price |

*SELLABLE A*

|  |  |  |
| --- | --- | --- |
| ID | QNT\_AVAILABLE | PRICE |

**SELLABLE *B***

|  |  |
| --- | --- |
| QNT\_STORAGE | QNT\_AVAILABLE |

1. The SELLABLE relation schema satisfies all the conditions of 1NF since there is no multivalued attributes or composite attributes, where all attributes are single.
2. The SELLABLE relation schema satisfies all the conditions of 2NF since all attributes depends on the primary key (ID), where if ID is dropped from this relation, the functional dependency does not hold any more.
3. The SELLABLE relation schema does not satisfy all the conditions of 3NF since the FD represented by **QNT\_AVAILABLE--->QNT\_STORAGE** is a functional dependency were neither QNT\_AVAILABLE is a super key nor QNT\_STORAGE is prime attribute. Thus, the above decomposition should be done.
4. The SELLABLE relation schema satisfies all the conditions of BCNF after the separation since there is no FD X -> A, such that X is not super key.

***Schedule***

|  |  |  |
| --- | --- | --- |
| ID | Start | End |

1. The SCHEDULE relation schema satisfies all the conditions of 1NF since there is no multivalued attributes or composite attributes, where all attributes are single.
2. The SCHEDULE relation schema satisfies all the conditions of 2NF since all attributes depends on the primary key (ID), where if ID is dropped from this relation, the functional dependency does not hold any more.
3. The SCHEDULE relation schema satisfies all the conditions of 3NF since it is in 2NF and there is no non-prime attribute in BRANCH relation is transitively dependent on the primary key (ID).
4. The SCHEDULE relation schema satisfies all the conditions of BCNF since there exists no FD X -> A, such that X is not super key.

**Fee**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Paid | Amnt\_ Due | Date\_ Due | Pay\_ Method |

1. The FEE relation schema satisfies all the conditions of 1NF since there is no multivalued attributes or composite attributes, where all attributes are single.
2. The FEE relation schema satisfies all the conditions of 2NF since all attributes depends on the primary key (ID), where if ID is dropped from this relation, the functional dependency does not hold any more.
3. The FEE relation schema satisfies all the conditions of 3NF since it is in 2NF and there is no non-prime attribute in BRANCH relation is transitively dependent on the primary key (ID).
4. The FEE relation schema satisfies all the conditions of BCNF since there exists no FD X -> A, such that X is not super key.

***Equipment***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Branch\_ ID | In Use | Condition | Name |

1. The EQUIPMENT relation schema satisfies all the conditions of 1NF since there is no multivalued attributes or composite attributes, where all attributes are single.
2. The EQUIPMENT relation schema satisfies all the conditions of 2NF since all attributes depends on the primary key (ID), where if ID is dropped from this relation, the functional dependency does not hold any more.
3. The EQUIPMENT relation schema satisfies all the conditions of 3NF since it is in 2NF and there is no non-prime attribute in BRANCH relation is transitively dependent on the primary key (ID).
4. The EQUIPMENT relation schema satisfies all the conditions of BCNF since there exists no FD X -> A, such that X is not super key.

**Shop**

|  |  |
| --- | --- |
| ID | BestSeller\_ID |

1. The SHOP relation schema satisfies all the conditions of 1NF since there is no multivalued attributes or composite attributes, where all attributes are single.
2. The SHOP relation schema satisfies all the conditions of 2NF since all attributes depends on the primary key (ID), where if ID is dropped from this relation, the functional dependency does not hold any more.
3. The SHOP relation schema satisfies all the conditions of 3NF since it is in 2NF and there is no non-prime attribute in BRANCH relation is transitively dependent on the primary key (ID).
4. The SHOP relation schema satisfies all the conditions of BCNF since there exists no FD X -> A, such that X is not super key.

***Member***

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Username | Gender | Birthdate | FName | LName | Address | Password | BC\_ID | Emg\_Contact | Phone |

1. The MEMBER relation schema satisfies all the conditions of 1NF since there is no multivalued attributes or composite attributes, where all attributes are single.
2. The MEMBER relation schema satisfies all the conditions of 2NF since all attributes depends on the primary key (Username), where if Username is dropped from this relation, the functional dependency does not hold any more.
3. The MEMBER relation schema satisfies all the conditions of 3NF since it is in 2NF and there is no non-prime attribute in BRANCH relation is transitively dependent on the primary key (Username).
4. The MEMBER relation schema satisfies all the conditions of BCNF since there exists no FD X -> A, such that X is not super key.

***Membership***

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Type | Expiry | Start | Consecutive | Cost | Username |

***Membership A:***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Type | Expiry | Start | Consecutive |

***Membership B:***

|  |  |
| --- | --- |
| Type | Cost |

1. The MEMBERSHIP relation schema satisfies all the conditions of 1NF since there is no multivalued attributes or composite attributes, where all attributes are single.
2. The MEMBERSHIP relation schema satisfies all the conditions of 2NF since all attributes depends on the primary key (ID), where if ID is dropped from this relation, the functional dependency does not hold any more.
3. The MEMBERSHIP relation schema does not satisfy all the conditions of 3NF since the FD represented by **TYPE--->COST** is a functional dependency were neither TYPE is a super key nor COST is prime attribute. Thus, the above decomposition should be done.
4. The MEMBERSHIP relation schema satisfies all the conditions of BCNF after the separation since there is no FD X -> A, such that X is not super key.

**Body\_ Composition**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date | Weight | Height | %BodyFat | %MuscleMass |

1. The BODY\_COMPOSITION relation schema satisfies all the conditions of 1NF since there is no multivalued attributes or composite attributes, where all attributes are single.
2. The BODY\_COMPOSITION relation schema satisfies all the conditions of 2NF since all attributes depends on the primary key (DATE), where if DATE is dropped from this relation, the functional dependency does not hold any more.
3. The BODY\_COMPOSITION relation schema satisfies all the conditions of 3NF since it is in 2NF and there is no non-prime attribute in BRANCH relation is transitively dependent on the primary key (DATE).
4. The BODY\_COMPOSITION relation schema satisfies all the conditions of BCNF since there exists no FD X -> A, such that X is not super key.

***Relations with no non-prime attributes:***

**1-USES:**

|  |  |
| --- | --- |
| Username | E\_ID |

**2- PAYS**

|  |  |
| --- | --- |
| Username | F\_ID |

**3- HAS\_A**

|  |  |
| --- | --- |
| Username | BC\_DATE |

**4-TAKES\_PLACE\_IN**

|  |  |
| --- | --- |
| B\_ID | S\_ID |

**5- AVAILABLE\_IN**

|  |  |
| --- | --- |
| S\_ID | SELL\_ID |

**6-FOLLOWS**

|  |  |
| --- | --- |
| C\_ID | SCHED\_ID |

**7- BRANCH\_PHONE**

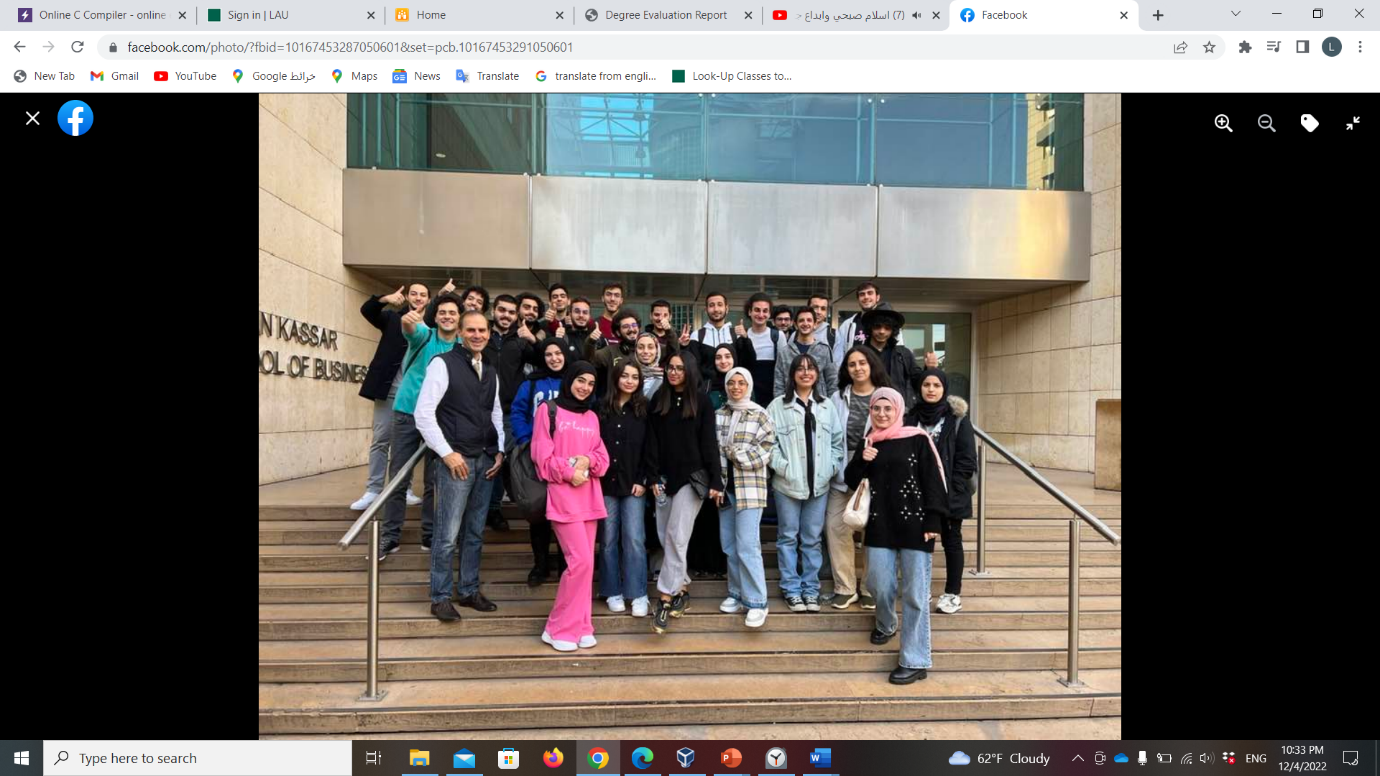
|  |  |
| --- | --- |
| BRANCH\_ID | PHONE |

**8- EMPLOYEE\_SHIFT**

|  |  |
| --- | --- |
| EMPLOYEE\_ID | SHIFT |

**9- EMPLOYEE\_PHONE**

|  |  |
| --- | --- |
| EMPLOYEE\_ID | PHONE |



Thank you for being the best instructor this semester. You inspired in us a love for learning and made us feel like we could ask you anything. We feel so truly lucky to have a teacher who shows all of the care, understanding, and patience that you do. Our fond memories of the time in your classroom will last a lifetime.