University of British Columbia, Vancouver
Department of Computer Science

CPSC 304 Project Cover Page

Milestone #: 2

Date: 1st March 2023

Group Number: 76

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Kushagra Sethi	45293503	p3c9u	kushagras.collegeapps@gmail.com
Duc Anh Dang	52272614	q7x2b	kevindang243@gmail.com
Carter Yam	32068884	d9e5c	cey1432@gmail.com

By typing our names and student numbers in the above table, we certify that the work in the attached assignment was performed solely by those whose names and student IDs are included above. (In the case of Project Milestone 0, the main purpose of this page is for you to let us know your e-mail address, and then let us assign you to a TA for your project supervisor.)

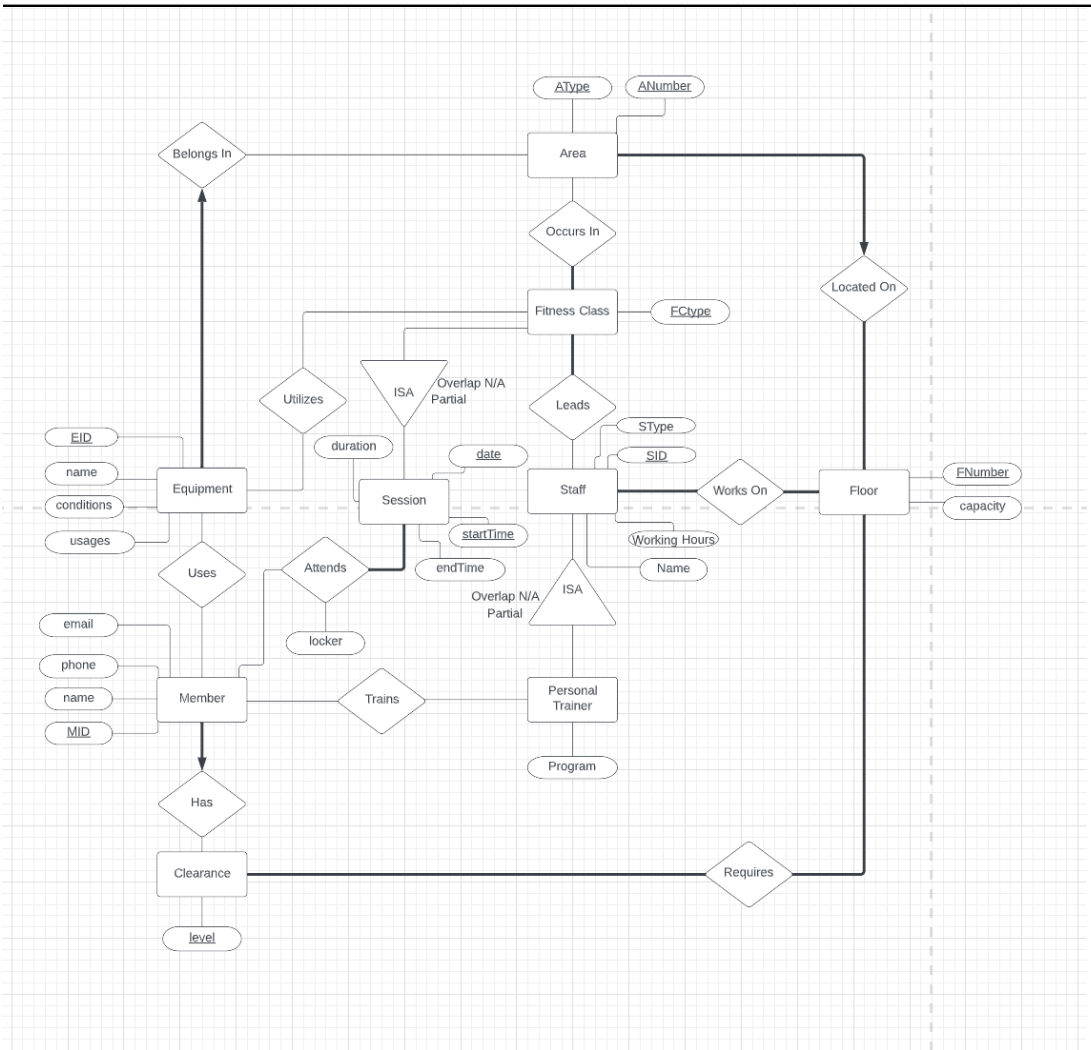
In addition, we indicate that we are fully aware of the rules and consequences of plagiarism, as set forth by the Department of Computer Science and the University of British Columbia

2. A brief (~2-3 sentences) summary of your project. Many of your TAs are managing multiple projects so this will help them remember details about your project.

This project is a gym management software that models various aspects of gym operations, such as security, staff coordination, member sessions, equipment management, and session management. The database allows gym staff to view and manage memberships, organize staff shifts, schedule personal training sessions, and book gym areas for fitness classes. The application technology stack is full stack with JDBC with Oracle as the backend, and a JavaScript/React.js framework for the web frontend to display queried information from the database.

3. The ER diagram you are basing your item #3 (below) on. This ER diagram may be the same as your milestone 1 submission or it might be different. If you have made changes from the version submitted in milestone 1, attach a note indicating what changes have been made and why.

   If you have decided not to implement the suggestions given by your project mentor, please be sure to leave a note stating why. This is **not** to say that you must do everything that your project mentor says. In many instances, there are trade-offs between design choices and your decision may be influenced by different factors. That being said, your TAs will often leave suggestions that are meant to help massage your project into a form that will fit with the requirements in future project milestones. If you choose not to take their advice, it would be helpful for them to know why in order to better assist the group moving forward.



Change notes:

- Added “Overlap N/A” to both ISA triangles present as the overlapping/disjoint constraints do not really apply to our ISAs as they each only have a single subclass.
- Added attributes “duration” for Session, “ANumber” for Area to create new FDs aside from PKs and CKs
- Renamed some similarly named attributes such as ID to names that are associated with their entities (i.e. “MID”, “SID”, instead of “ID” for Member and Staff) to make the schemas more readable.

4. The schema derived from your ER diagram (above). For the translation of the ER diagram to the relational model, follow the same instructions as in your lectures. The process should be reasonably straightforward. For each table:
    a. List the table definition (e.g., Table1(attr1: domain1, attr2: domain2, ...)). Make sure to include the domains for each attribute.
    b. Specify the primary key (PK), candidate key, (CK) foreign keys (FK), and other constraints (e.g., not null, unique, etc.) that the table must maintain.

---

a.

Member(MID: INTEGER, name: VARCHAR(20), phone: VARCHAR(13), email:VARCHAR(30), **clearanceLevel**: INTEGER)

Clearance(level: INTEGER)

Floor(FNumber: INTEGER, capacity: INTEGER)

Requires(**clearanceLevel**: INTEGER, **FNumber**: INTEGER)

Staff(SID: INTEGER, workingHours: INTEGER, name: VARCHAR(20), SType: VARCHAR(20))

WorksOn(**SID**: INTEGER, **FNumber**: INTEGER)

Equipment(EID: INTEGER, name: VARCHAR(20), conditions: VARCHAR(20), usages:VARCHAR(20), **AType**:VARCHAR(20))

Area(AType: VARCHAR(20), ANumber: INTEGER **FNumber**: INTEGER)

Uses(**MID**: INTEGER, **EID**: INTEGER)

FitnessClass(FCtype:VARCHAR(15), **startTime**: VARCHAR(8), **date**: VARCHAR(10))

Leads(**SID**: INTEGER, **FCDate**: VARCHAR(10), **FCStartTime**: VARCHAR(8), **FCType**: VARCHAR(15))

OccursIn(**AType**: VARCHAR(20), **FCType**: VARCHAR(15), **FCDate**: VARCHAR(10), **FCStartTime**: VARCHAR(8))

Session(date: VARCHAR(10), startTime: VARCHAR(8), endTime: VARCHAR(8), duration: INTEGER)

Attends(**MID**: INTEGER, **sessionDate**: VARCHAR(10), **sessionStartTime**: VARCHAR(8), locker: INTEGER)

Utilizes(**EID**: INTEGER, **FCType**: VARCHAR(15), **FCDate**: VARCHAR(10), **FCStartTime**: VARCHAR(8))

PersonalTrainer(**SID**: INTEGER, program: VARCHAR(15))

Trains(**SID**: INTEGER, **MID**: INTEGER)

b.

Member:

PK: MID
CK: {[name, phone],[email, phone]}
FK: ClearanceLevel
Non-null: Clearance Level

Clearance:
PK: level

Floor:
PK: FNumber
Note: total participation requires assertions for Floor-LocatedOn

Requires:
PK: clearanceLevel, FNumber
FK: clearanceLevel, FNumber
Note: total participation requires assertions for Floor-Requires and Clearance-Requires

Staff:
PK: SID

WorksOn:
PK: SID, FNumber
FK: SID, FNumber
Note: total participation requires assertions for Staff-WorksOn and Floor-WorksOn

Equipment:
PK: EID
FK: AType
Non-null: AType

Area:
PK: AType
FK: FNumber
Non-null: FNumber

Uses:
PK: MID, EID
FK: MID, EID

FitnessClass:
PK: FCtype, startTime, date
FK: startTime, date

Leads:
PK: SID, FCDate, FCStartTime, FCType
FK: SID, FCDate, FCStartTime, FCType
Note: total participation requires assertions for FitnessClass-Leads

OccursIn:
PK: AType, FCType, FCDate, FCStartTime
FK: AType, FCType, FCDate, FCStartTime
Note: total participation requires assertions for FitnessClass-OccursIn

Session:
PK: date, startTime

Attends:
PK: MID, sessionDate, sessionStartTime
FK: MID, sessionDate, sessionStartTime

Utilizes:
PK: EID, FCType, FCDate, FCStartTime
FK: EID, FCType, FCDate, FCStartTime

PersonalTrainer:
PK: SID
FK: SID

Trains:
PK: SID, MID
FK: SID, MID

5. Functional Dependencies (FDs)
   a. Identify the functional dependencies in your relations, including the ones involving all candidate keys (including the primary key).

      PKs and CKs are considered functional dependencies and should be included in the list of FDs. You do not need to include trivial FDs such as A → A.

      Note: In your list of FDs, there must be some kind of valid FD other those identified by a PK or CK. If you observe that no relations have FDs other than the PK and CK(s), then you will have to intentionally add some (meaningful) attributes to show valid FDs. We want you to get a good normalization exercise. Your design must go through a normalization process.

Member:
FDs:
- MID -> name, phone, email, clearanceLevel
- Name, phone -> MID, email, clearanceLevel

- Email, phone -> MID, name, clearanceLevel

Equipment:

FDs:

- EID → name, conditions, usages, AType
- name → usages

Session:

FDs:

- date, startTime → endTime, duration
- startTime, endTime → duration

FitnessClass

FD:

- FCType, startTime, date -> endTime, duration
- startTime, endTime → duration

Staff:

FDs:

- SID → workingHours, name, SType
- SType → workingHours

PersonalTrainer:

FDs:

- SID → workingHours, name, Program, SType

Floor:

FDs:

- FNumber -> capacity

Attends:

FDs:

- MID, sessionDate, sessionStartTime -> locker

Area:

FDs:

- ANumber, AType → FNumber

---

6. Normalization
    a. Normalize each of your tables to be in 3NF or BCNF. Give the list of tables, their primary keys, their candidate keys, and their foreign keys after normalization.

       You should show the steps taken for the decomposition. Should there be errors, and no work is shown, no partial credit can be awarded without steps shown. The format should be the same as Step 3, with tables listed similar to Table1(attr1:domain1, attr2:domain2, …). ALL Tables must be listed, not only the ones post normalization.

Following BCNF for all:

Member

FDs:
MID -> name, phone, email
Name, phone -> MID, email
email, phone -> MID, name

Closures of FDs:
MID+ = {MID, name, phone, email}
{Name, phone}+ = {name, phone, MID, email}
{email, phone}+ = {email, phone, MID, name}

All left sides of FDs are superkeys, therefore, Member is in BCNF.

Session1(startTime, endTime, duration), Session2(date, **startTime**, **endTime**)

Session
FDs:
date, startTime → endTime, duration
startTime, endTime → duration

Finding Closures:
{date, startTime}+ = {date, startTime, endTime, duration}
{startTime, endTime}+ = {startTime, endTime, duration}

Session is not in BCNF as LHS of FD(startTime, endTime → duration) is not a superkey for Session.

Decomposing on FD(startTime, endTime → duration):
Session1(startTime, endTime, duration), Session2(date, **startTime**, **endTime**)

Session1 is in BCNF because it has 2 attributes.
Session2 is in BCNF because LHS of FD(date, startTime → endTime, duration) is a superkey for the relation.

FitnessClass

- FCType, startTime, date -> endTime, duration
- startTime, endTime → duration

FDs:
FCType, date, startTime → endTime, duration
startTime, endTime → duration

Finding Closures:
{FCType, date, startTime}+ = {FCType, date, startTime, endTime, duration}
{startTime, endTime}+ = {startTime, endTime, duration}

FitnessClass is not in BCNF as LHS of FD(startTime, endTime → duration) is not a superkey for Session.

Decomposing on FD(startTime, endTime → duration):

FitnessClass1(startTime, endTime, duration), FitnessClass2(FCType, date, **startTime**, **endTime**)

FitnessClass1 is in BCNF because it has 2 attributes.
FitnessClass2 is in BCNF because LHS of FD(FCType, date, startTime → endTime, duration) is a superkey

Equipment(EID, name, conditions, usages, AType)
FDs:
EID → name, conditions, usages, AType
name → usages

Closures of LHS of FDs:
{EID}+ = {EID, name, conditions, usages, AType}
{name}+ = {name, usages}

Equipment is not in BCNF, as the (name → usages) FD does not satisfy the conditions of the LHS (name) being a superkey for Equipment.

Decompose on (name → usages):

Equipment1(name, usages), Equipment2(EID, name, conditions, AType)

Equipment1 is in BCNF because it only has 2 attributes.
Equipment2 is in BCNF because for the remaining FD (EID → name, conditions, usages, AType), EID is a superkey for the relation.

Staff(SID, name, workingHours, SType)
FDs:
SID → workingHours, name, SType

SType → workingHours

Closures of LHS of FDs:
{SID}+ = {SID, name, workingHours, SType}
{SType}+ = {SType, workingHours}

Staff is not in BCNF, as the (SType → workingHours) FD does not satisfy the conditions of the LHS (SType) being a superkey for Staff.

Decompose on (SType → workingHours):

Staff1(SType, workingHours), Staff2(SID, name, **SType**)

Staff1 is in BCNF as it only has 2 attributes.
Staff2 is in BCNF because for the remaining FD (SID → workingHours, name, SType), SID is a superkey for the relation.

PersonalTrainer(SID, name, workingHours, SType, Program):
FDs:
SID → workingHours, name, Program, SType
SType → workingHours

{SID}+ = {SID, name, workingHours, SType, Program}
{SType}+ = {SType, workingHours}

PersonalTrainer is not in BCNF, as the (SType → workingHours) FD does not satisfy the conditions of the LHS (SType) being a superkey for PersonalTrainer.

Decompose on (SType → workingHours):

PersonalTrainer1(SType, workingHours), PersonalTrainer2(SID, name, **SType**, Program)

PersonalTrainer1 is in BCNF as it only has 2 attributes.
PersonalTrainer2 is in BCNF because for the remaining FD (SID → workingHours, name, Program, SType), SID is a superkey for the relation.

Floor(FNumber, capacity)
FDs:
FNumber -> capacity

Closure of LHS of FD:
{FNumber}+ = {FNumber, capacity}

The LHS of the FD is a superkey for the relation, therefore Floor is in BCNF.

Attends

FDs:
MID, sessionDate, sessionStartTime -> locker

Finding Closures:

{MID, sessionDate, sessionStartTime}+ = {MID, sessionDate, sessionStartTime, locker}

LHS of FD(MID, sessionDate, sessionStartTime -> locker) is a superkey for Attends. Therefore, Attends is in BCNF.

Area(ANumber, AType, FNumber)
FDs:
ANumber, AType → FNumber

Closure of LHS of FD:
{ANumber, AType}+ = {ANumber, AType, FNumber}

The LHS of the FD is a superkey for the relation, therefore Area is in BCNF.

Therefore our final tables in step 6 are:

- Member(MID: INTEGER, name: VARCHAR(20), phone: VARCHAR(13), email: VARCHAR(30), **clearanceLevel**: INTEGER)
- Session1(startTime: VARCHAR(8), endTime: VARCHAR(8), duration: INTEGER)
- Session2(date: INTEGER, **startTime**: : VARCHAR(8), **endTime**: VARCHAR(8))
- FitnessClass1 **startTime:**VARCHAR(8), **endTime**: VARCHAR(8), **duration:** INTEGER)
- FitnessClass2(FCType: VARCHAR(15), **date**: INTEGER, **startTime**: VARCHAR(8), **endTime**: VARCHAR(8))
- Equipment1(name: VARCHAR(20), usages: VARCHAR(20))
- Equipment2(EID: INTEGER, **name**: VARCHAR(20), conditions: VARCHAR(20), **AType**: VARCHAR(20))
- Staff1(SType: VARCHAR(20), workingHours: INTEGER)
- Staff2(SID: INTEGER, name: VARCHAR(20), **SType**: VARCHAR(20))
- PersonalTrainer1(SType: VARCHAR(20), workingHours: INTEGER)
- PersonalTrainer2(SID: INTEGER, name: VARCHAR(20), **SType**: VARCHAR(20), Program: VARCHAR(15))
- Attends(**MID**: INTEGER, **sessionDate**: VARCHAR(10), **sessionStartTime**: VARCHAR(8), locker: INTEGER)
- Floor(FNumber: INTEGER, capacity: INTEGER)
- Area(AType: VARCHAR(20), ANumber: INTEGER **FNumber**: INTEGER)

7. The SQL DDL statements required to create all the tables from item #6. The statements should use the appropriate foreign keys, primary keys, UNIQUE constraints, etc.

Note: We would use the ON UPDATE CASCADE option for our tables as it makes the most sense, but we plan to implement our project using Oracle, which does not support the ON UPDATE CASCADE option.

```
CREATE TABLE Member (
        MID INTEGER,
        name VARCHAR(20),
        phone VARCHAR(13),
        email VARCHAR(30),
        clearanceLevel INTEGER NOT NULL,
        PRIMARY KEY (MID),
        FOREIGN KEY (clearanceLevel) REFERENCES Clearance(level)
                ON DELETE SET DEFAULT
)

CREATE TABLE Equipment1 (
        name VARCHAR(20),
        usages VARCHAR(20),
        PRIMARY KEY (name)
)

CREATE TABLE Equipment2 (
        EID INTEGER,
        name VARCHAR(20),
        conditions VARCHAR(20),
        AType VARCHAR(20),
        PRIMARY KEY (EID),
        FOREIGN KEY (AType) REFERENCES Area(AType)
                ON DELETE CASCADE
                ON UPDATE SET NULL,
        FOREIGN KEY (name) REFERENCES Equipment1(name)
                ON DELETE CASCADE
 )

CREATE TABLE Session1 (
        startTime VARCHAR(8),
        endtime VARCHAR(8),
        duration INTEGER,
        PRIMARY KEY (startTime, endTime)
 )

CREATE TABLE Session2 (
        date INTEGER,
        startTime VARCHAR(8),
        endtime VARCHAR(8),
        PRIMARY KEY (date, startTime)
        FOREIGN KEY (startTime, endTime) REFERENCES Session1(startTime, endTime)
                ON DELETE CASCADE
 )

CREATE TABLE Staff1 (
        SType  VARCHAR(20),
        workingHours INTEGER,
```

```
        PRIMARY KEY (SType)
 )

CREATE TABLE Staff2 (
        SID INTEGER,
        name VARCHAR(20),
        SType VARCHAR(20),
        PRIMARY KEY (SID),
        FOREIGN KEY (SType) REFERENCES Staff1(SType)
                ON DELETE CASCADE
                ON UPDATE SET NULL
 )

CREATE TABLE FitnessClass1 (
        startTime VARCHAR(8),
        endTime VARCHAR(8),
        duration INTEGER,
        PRIMARY KEY (startTime, endTime)
 )

CREATE TABLE FitnessClass2 (
        FCType VARCHAR(15),
        date VARCHAR(10),
        startTime VARCHAR(8),
        endTime VARCHAR(8),
        PRIMARY KEY (FCType, date, startTime)
        FOREIGN KEY (startTime, endTime) REFERENCES FitnessClass1(startTime,
        endTime)
                ON DELETE CASCADE
 )

CREATE TABLE PersonalTrainer1 (
        SType VARCHAR(20),
        workingHours INTEGER,
        PRIMARY KEY (SType),
 )

CREATE TABLE PersonalTrainer2 (
        SID INTEGER,
        name VARCHAR(20),
        SType VARCHAR(20),
        program VARCHAR(15),
        PRIMARY KEY (SID),
        FOREIGN KEY (SType) REFERENCES PersonalTrainer1(SType)
                ON DELETE CASCADE
                ON UPDATE SET NULL
 )

CREATE TABLE Area (
```

```
        ANumber INTEGER,
        AType VARCHAR(20),
        FNumber INTEGER,
        PRIMARY KEY (ANumber, AType),
        FOREIGN KEY (FNumber) REFERENCES Floor(Number)
                ON DELETE CASCADE
                ON UPDATE SET NULL
 )


CREATE TABLE Floor (
        FNumber INTEGER,
        capacity INTEGER,
        PRIMARY KEY (FNumber)
)

CREATE TABLE Attends (
        MID INTEGER,
        sessionDate VARCHAR(10),
        sessionStartTime VARCHAR(8),
        locker INTEGER,
        PRIMARY KEY (MID, sessionDate, sessionStartTime),
        FOREIGN KEY (MID) REFERENCES Member(MID)
                ON DELETE CASCADE,
        FOREIGN KEY (sessionsDate, sessionStartTime) REFERENCES Session(date,
        startTime)
                ON DELETE CASCADE
)
```

8. INSERT statements to populate each table with at least 5 tuples. You will likely want to have more than 5 tuples so that you can have meaningful queries later on.

MEMBER

```
INSERT
INTO        Member(MID, name, phone, email, clearanceLevel)
VALUES      ('1','Kevin Dang', '(778) 123 1234', 'kevindang@gmail.com', '1')

INSERT
INTO        Member(MID, name, phone, email, clearanceLevel)
VALUES      ('2', 'Carter Yam', '(778) 234 2345', 'carteryam@gmail.com', 2)

INSERT
INTO        Member(MID, name, phone, email, clearanceLevel)
VALUES      ('3', 'Kushagra Sethi', '(778) 345 3456', 'kushagrasethi@gmail.com', 3)

INSERT
INTO        Member(MID, name, phone, email, clearanceLevel)
VALUES      ('4, 'John Smith', '(778) 456 4567', 'johnsmith@gmail.com', '3')
```

```
INSERT
INTO        Member(MID, name, phone, email, clearanceLevel)
VALUES      ('5', 'Jane Doe', '(778) 567 5678', 'janedoe@gmail.com', '1')

INSERT
INTO        Member(MID, name, phone, email, clearanceLevel)
VALUES      (6, 'Pascal Siakam', '(778) 678 6789', 'pascalsiakam@gmail.com', 2)
```

EQUIPMENT1

```
INSERT
INTO        EQUIPMENT1(name, usages)
VALUES      ("Lat pulldown", "Back muscle")

INSERT
INTO        EQUIPMENT1(name, usages)
VALUES      ("Squat rack", "Legs")

INSERT
INTO        EQUIPMENT1(name, usages)
VALUES      ("Bench Press Rack", "Chest")

INSERT
INTO        EQUIPMENT1(name, usages)
VALUES      ("Treadmill", "Cardio")

INSERT
INTO        EQUIPMENT1(name, usages)
VALUES      ("Yoga matts", "Yoga exercises")

INSERT
INTO        EQUIPMENT1(name, usages)
VALUES      ("Toes raise machine", "Calves")
```

EQUIPMENT2

```
INSERT
INTO        EQUIPMENT2(EID, name, conditions, AType)
VALUES      (1, "Lat pulldown", "Brand spankin new", "Body Building")

INSERT
INTO        EQUIPMENT2(EID, name, conditions, AType)
VALUES      (2, "Squat rack", "Brand spankin new", "Body Building")

INSERT
INTO        EQUIPMENT2(EID, name, conditions, AType)
VALUES      (3, "Bench Press Rack", "Used", "Body Building")

INSERT
```

INTO EQUIPMENT2(EID, name, conditions, AType)
VALUES (4, “Treadmill”, “Rusty”, “Cardio”)

INSERT
INTO EQUIPMENT2(EID, name, conditions, AType)
VALUES (5, “Yoga matts”, “Brand spankin new”, “Calisthenics”)

INSERT
INTO EQUIPMENT2(EID, name, conditions, AType)
VALUES (6, “Toes raise machine”, “Used”, “Body Building”)

SESSION1

INSERT
INTO Session1(startTime, endTime, duration)
VALUES (‘14:23:05’, ‘15:56:22’, 5641)

INSERT
INTO Session1(startTime, endTime, duration)
VALUES (‘02:01:52’, ‘04:11:57’, 8025)

INSERT
INTO Session1(startTime, endTime, duration)
VALUES (‘20:13:48’, ‘21:00:43’, 3045)

INSERT
INTO Session1(startTime, endTime, duration)
VALUES (‘10:08:00’, ‘11:46:21’, 4701)

INSERT
INTO Session1(startTime, endTime, duration)
VALUES (‘15:33:56’, ‘16:48:06’, 4580)

SESSION2

INSERT
INTO Session2(date, startTime, endTime)
VALUES (‘2022-09-22’, ‘14:23:05’, ‘15:56:22’)

INSERT
INTO Session2(date, startTime, endTime)
VALUES (‘2022-10-21’, ‘02:01:52’, ‘04:11:57’)

INSERT
INTO Session2(date, startTime, endTime)
VALUES (‘2022-11-20’, ‘20:13:48’, ‘21:00:43’)

INSERT
INTO Session2(date, startTime, endTime)

VALUES ('2022-12-19', '10:08:00', '11:46:21')

INSERT
INTO Session2(date, startTime, endTime)
VALUES ('2023-01-18', '15:33:56', '16:48:06')

FITNESSCLASS1

INSERT
INTO FITNESSCLASS1(startTime, endTime, duration)
VALUES ("15:23:05", "17:23:05",2401)

INSERT
INTO FITNESSCLASS1(startTime, endTime, duration)
VALUES ("16:23:05", "18:23:05",2402)

INSERT
INTO FITNESSCLASS1(startTime, endTime, duration)
VALUES ("17:23:05", "19:23:05",2403)

INSERT
INTO FITNESSCLASS1(startTime, endTime, duration)
VALUES ("18:23:05", "20:23:05",2404)

INSERT
INTO FITNESSCLASS1(startTime, endTime, duration)
VALUES ("19:23:05", "21:23:05",2405)

FITNESSCLASS2

INSERT
INTO FITNESSCLASS2(FCType, date, startTime, endTime)
VALUES ("Cardio", "2022-08-14", "16:23:05", "17:23:05")

INSERT
INTO FITNESSCLASS2(FCType, date, startTime, endTime)
VALUES ("Zumba", "2022-08-15", "17:23:05", "18:23:05")

INSERT
INTO FITNESSCLASS2(FCType, date, startTime, endTime)
VALUES ("Yoga", "2022-08-16", "18:23:05", "19:23:05")

INSERT
INTO FITNESSCLASS2(FCType, date, startTime, endTime)
VALUES ("HIIT", "2022-08-17", "19:23:05", "20:23:05")

INSERT

```
INTO        FITNESSCLASS2(FCType, date, startTime, endTime)
VALUES      (“Flow“, “2022-08-18”, “20:23:05”, “21:23:05”)
```

STAFF1

```
INSERT
INTO        STAFF1(SType, workingHours)
VALUES      (“Trainer”, “9am-5pm”)

INSERT
INTO        STAFF1(SType, workingHours)
VALUES      (“Trainer”, “1pm-5pm”)

INSERT
INTO        STAFF1(SType, workingHours)
VALUES      (“Trainer”, “1pm-5pm”)

INSERT
INTO        STAFF1(SType, workingHours)
VALUES      (“Trainer”, “7pm-9pm”)

INSERT
INTO        STAFF1(SType, workingHours)
VALUES      (“Trainer”, “9am-5pm”)

INSERT
INTO        STAFF1(SType, workingHours)
VALUES      (“Trainer”, “7pm-9pm”)

INSERT
INTO        STAFF1(SType, workingHours)
VALUES      (“Custodian”, “7pm-9pm”)
```

STAFF2

```
INSERT
INTO        STAFF2(SID, name, SType)
VALUES      (‘1’, “Ice Cube”, “Trainer”)

INSERT
INTO        STAFF2(SID, name, SType)
VALUES      (‘2’, “Kevin Hart”, “Trainer”)

INSERT
INTO        STAFF2(SID, name, SType)
VALUES      (‘3’, “Rihanna”, “Trainer”)
```

```
INSERT
INTO        STAFF2(SID, name, SType)
VALUES      ('4', "The Rock", "Trainer")

INSERT
INTO        STAFF2(SID, name, SType)
VALUES      ('5', "Ryan Gosling", "Trainer")

INSERT
INTO        STAFF2(SID, name, SType)
VALUES      ('6', "Ice Spice", "Trainer")

INSERT
INTO        STAFF2(SID, name, SType)
VALUES      ('7', "Rebecca Black", "Custodian")
```

PERSONALTRAINER1

```
INSERT
INTO        PERSONALTRAINER1(SType, workingHours)
VALUES      ("Trainer", "9am-5pm")

INSERT
INTO        PERSONALTRAINER1(SType, workingHours)
VALUES      ("Trainer", "1pm-5pm")

INSERT
INTO        PERSONALTRAINER1(SType, workingHours)
VALUES      ("Trainer", "1pm-5pm")

INSERT
INTO        PERSONALTRAINER1(SType, workingHours)
VALUES      ("Trainer", "7pm-9pm")

INSERT
INTO        PERSONALTRAINER1(SType, workingHours)
VALUES      ("Trainer", "9am-5pm")

INSERT
INTO        PERSONALTRAINER1(SType, workingHours)
VALUES      ("Trainer", "7pm-9pm")
```

PERSONALTRAINER2

```
INSERT
INTO        PERSONALTRAINER2(SID, name, SType, program)
VALUES      ("1", "Ice Cube", "Trainer", "Cardio")
```

```
INSERT
INTO        PERSONALTRAINER2(SID, name, SType, program)
VALUES      (“2”, “Kevin Hart”, “Trainer”, “Free Weights”)

INSERT
INTO        PERSONALTRAINER2(SID, name, SType, program)
VALUES      (“3”, “Rihanna”, “Trainer”, “Free Weights”)

INSERT
INTO        PERSONALTRAINER2(SID, name, SType, program)
VALUES      (“4”, “The Rock”, “Trainer”, “Yoga”)

INSERT
INTO        PERSONALTRAINER2(SID, name, SType, program)
VALUES      (“5”, “Ryan Gosling”, “Trainer”, “Spin”)

INSERT
INTO        PERSONALTRAINER2(SID, name, SType, program)
VALUES      (“6”, “Ice Spice”, “Trainer”, “Free Weights”)
```

ATTENDS
Attends(**MID**: INTEGER, **sessionDate**: VARCHAR(10), **sessionStartTime**: VARCHAR(8), locker: INTEGER)

```
INSERT
INTO        Attends(MID, sessionDate, sessionStartTime, locker)
VALUES      (121,“2022-07-14”, “20:23:05”, 23)

INSERT
INTO        Attends(MID, sessionDate, sessionStartTime, locker)
VALUES      (122,“2022-07-15”, “20:23:05”, 24)

INSERT
INTO        Attends(MID, sessionDate, sessionStartTime, locker)
VALUES      (123,“2022-07-16”, “20:23:05”, 25)

INSERT
INTO        Attends(MID, sessionDate, sessionStartTime, locker)
VALUES      (124,“2022-07-17”, “20:23:05”, 26)

INSERT
INTO        Attends(MID, sessionDate, sessionStartTime, locker)
VALUES      (125,“2022-07-18”, “20:23:05”, 27)
```

AREA

Area(AType: VARCHAR(20), ANumber: INTEGER, **FNumber**: INTEGER)

```
INSERT
INTO        Area(AType, ANumber, FNumber)
VALUES      ('Cardio', 1, 2)

INSERT
INTO        Area(AType, ANumber, FNumber)
VALUES      ('Free Weights', 2, 1)

INSERT
INTO        Area(AType, ANumber, FNumber)
VALUES      ('Turf', 3, 2)

INSERT
INTO        Area(AType, ANumber, FNumber)
VALUES      ('Squat Racks', 4, 1)

INSERT
INTO        Area(AType, ANumber, FNumber)
VALUES      ('Machines', 5, 1)
```

FLOOR

```
INSERT
INTO        Floor(FNumber, capacity)
VALUES      (1, 75)

INSERT
INTO        Floor(FNumber, capacity)
VALUES      (2, 70)

INSERT
INTO        Floor(FNumber, capacity)
VALUES      (3, 65)

INSERT
INTO        Floor(FNumber, capacity)
VALUES      (4, 60)

INSERT
INTO        Floor(FNumber, capacity)
VALUES      (5, 55)
```