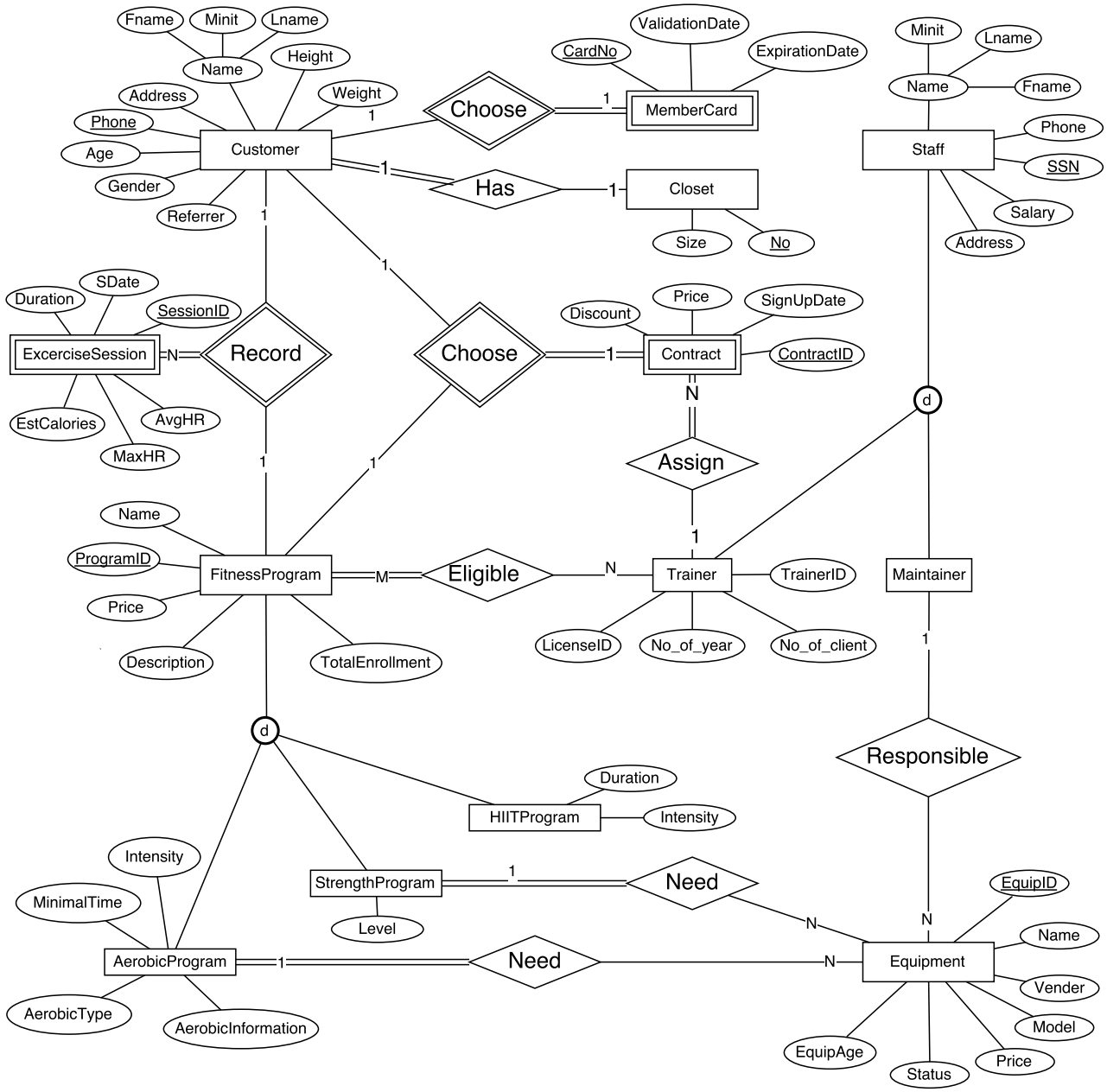
# Final Project Presentation

Hanlin He  Kai Kang

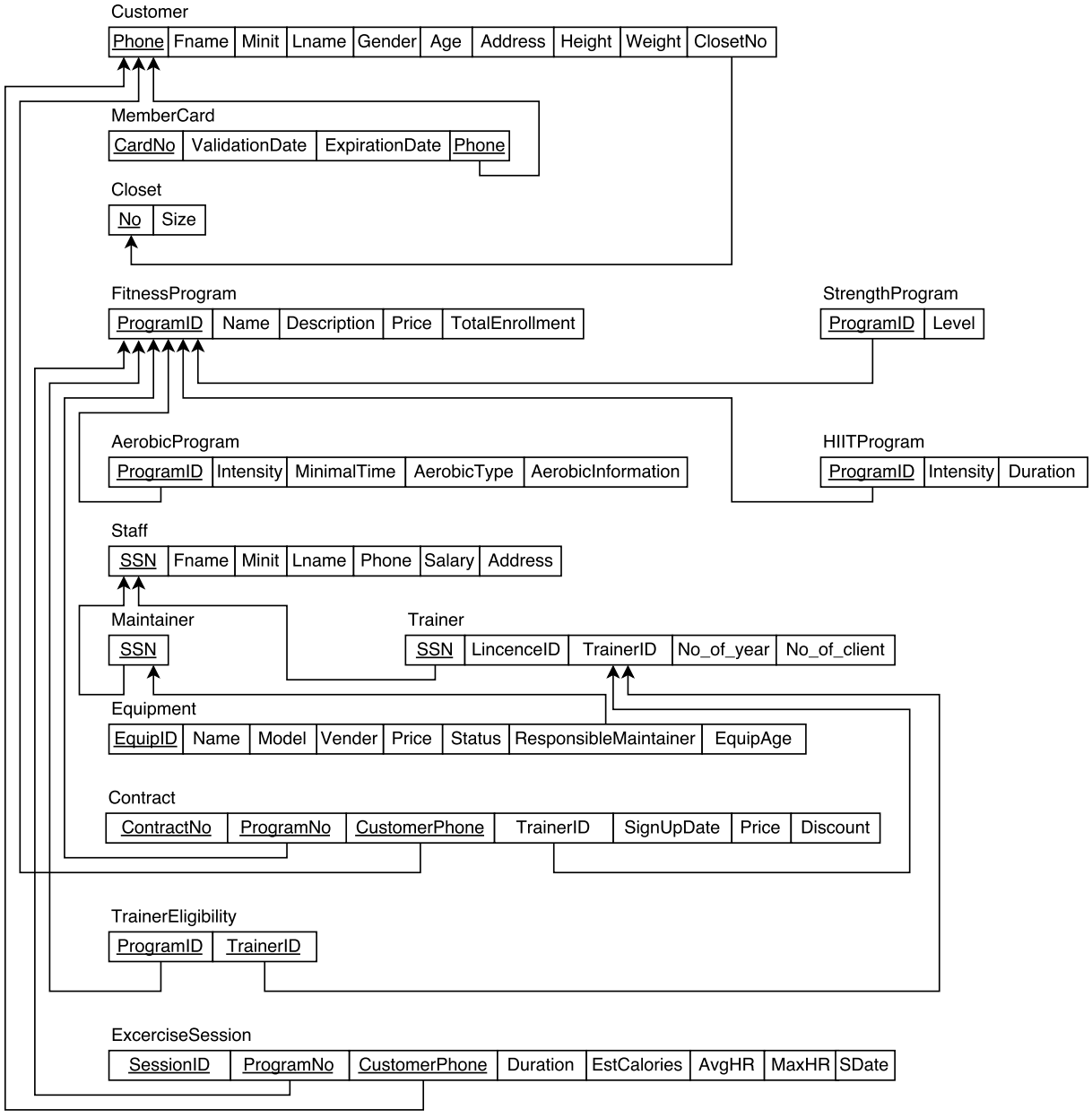
University of Texas at Dallas

August 12, 2017

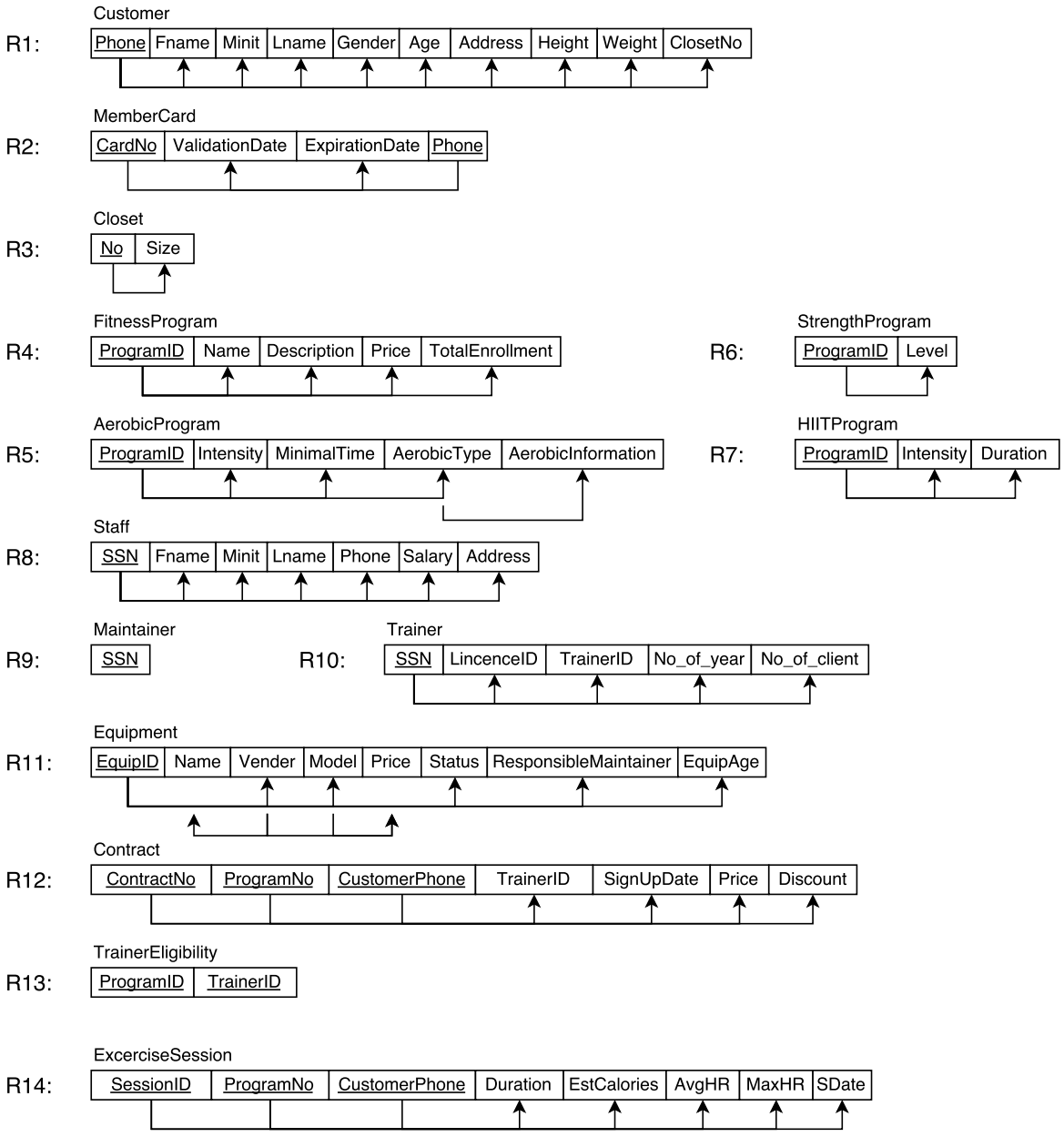
# EER Diagram


Fname
Minit
Lname
Name
Height
Address
Weight
Phone
Age
Gender
Referrer
Customer
Choose
CardNo
ValidationDate
ExpirationDate
MemberCard
Has
Closet
Size
No
Minit
Lname
Name
Fname
Staff
Phone
SSN
Salary
Address
Duration
SDate
SessionID
ExcerciseSession
Record
EstCalories
AvgHR
MaxHR
Choose
Discount
Price
SignUpDate
Contract
ContractID
Assign
Name
ProgramID
Price
Description
FitnessProgram
TotalEnrollment
Eligible
Trainer
TrainerID
LicenseID
No_of_year
No_of_client
Maintainer
Responsible
Duration
HIITProgram
Intensity
Intensity
MinimalTime
AerobicProgram
AerobicType
AerobicInformation
StrengthProgram
Level
Need
Need
EquipID
Name
Vender
Model
Price
Status
EquipAge
Equipment

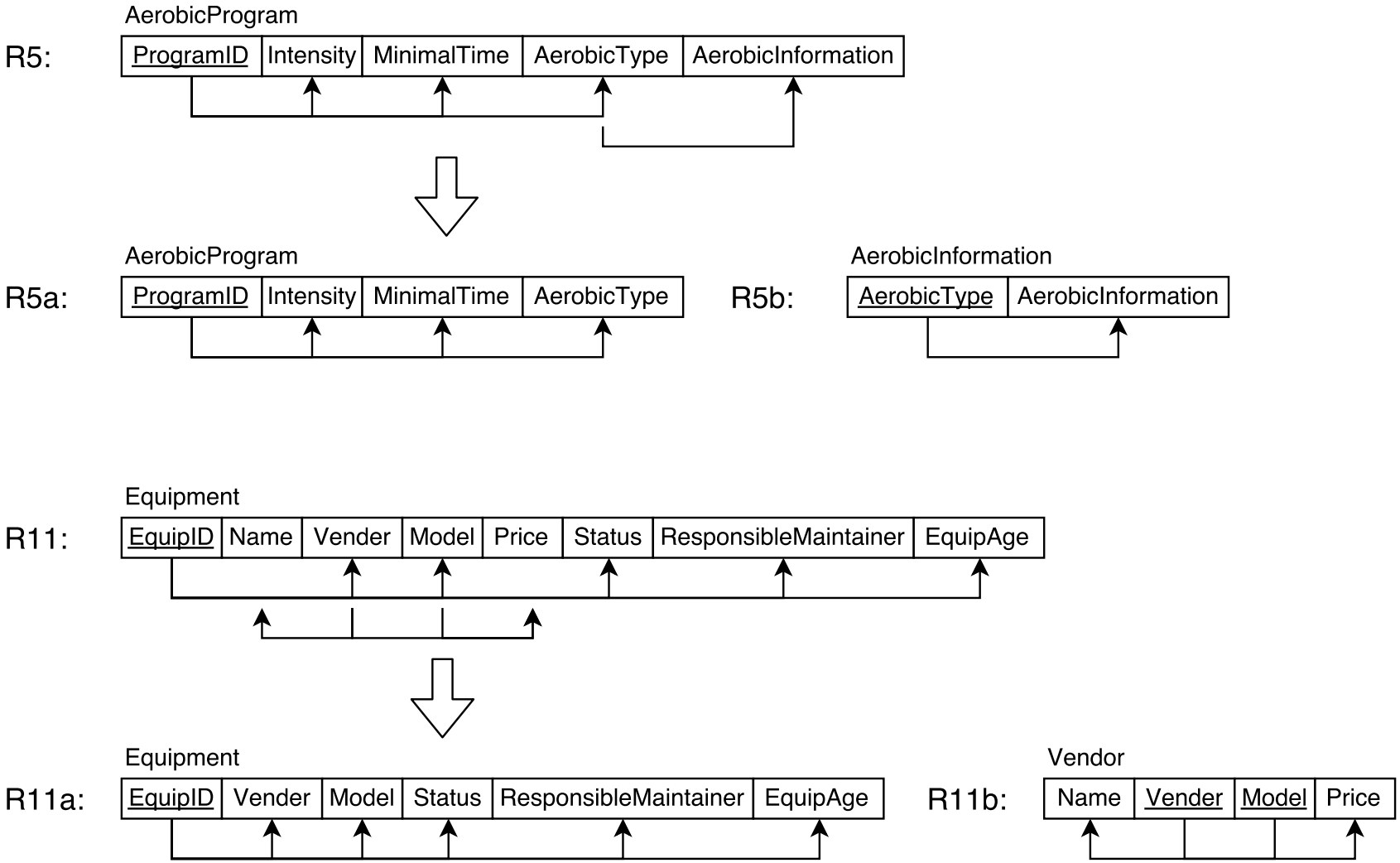
# Relational Schema after Mapping



# Functional Dependencies



# Normalization



# Update Total Enrollment of Program

Each time a record was added to the CONTRACT table, the *TotalEnrollment* in table FITNESSPROGRAM of the particular program should add 1.
The trigger is defined as follow:

```
1  CREATE OR REPLACE TRIGGER addTotalEnrollmentTrigger
2    AFTER INSERT ON CONTRACT
3         FOR EACH ROW
4  BEGIN
5    UPDATE FITNESSPROGRAM
6      SET TOTALENROLLMENT = TOTALENROLLMENT + 1
7      WHERE PROGRAMID = :new.PROGRAMNO;
8    UPDATE TRAINER
9      SET NO_OF_CLIENTS = NO_OF_CLIENTS + 1
10     WHERE TRAINERID = :new.TRAINERID;
11 END;/
```

# Trigger: Log All the Changes about Customer

Each time a record was added/updated/deleted in CUSTOMER table, record the modification in CUSTOMER_LOG table.
The trigger is defined as follow:

```
1  CREATE TABLE Customer_Log ( log_date DATE, action VARCHAR2(50),
2    old_Phone VARCHAR2(10), old_Gender CHAR, old_Age NUMBER,
3    old_Height FLOAT, old_Weight FLOAT, old_ClosetNo VARCHAR2(10),
4    new_Phone VARCHAR2(10), new_Gender CHAR, new_Age NUMBER,
5    new_Height FLOAT, new_Weight FLOAT, new_ClosetNo VARCHAR2(10)
6  );
7
8  CREATE OR REPLACE TRIGGER customerLogTrigger
9    AFTER INSERT OR UPDATE OR DELETE ON CUSTOMER
10       FOR EACH ROW
11 DECLARE
12   log_action  Customer_Log.action%TYPE;
13 BEGIN
14   IF INSERTING THEN
15     log_action := 'Insert';
16     INSERT INTO Customer_Log
17     VALUES (SYSDATE, log_action, null, null, null, null, null, null,
18             :new.PHONE, :new.GENDER, :new.AGE,
19             :new.HEIGHT, :new.WEIGHT, :new.CLOSETNO );
20   ELSIF UPDATING THEN
21     log_action := 'Update';
22     INSERT INTO Customer_Log
23     VALUES (SYSDATE, log_action,
24             :old.PHONE, :old.GENDER, :old.AGE,
25             :old.HEIGHT, :old.WEIGHT, :old.CLOSETNO,
26             :new.PHONE, :new.GENDER, :new.AGE,
27             :new.HEIGHT, :new.WEIGHT, :new.CLOSETNO );
28   ELSIF DELETING THEN
29     log_action := 'Delete';
30     INSERT INTO Customer_Log
31     VALUES (SYSDATE, log_action,
32             :old.PHONE, :old.GENDER, :old.AGE,
33             :old.HEIGHT, :old.WEIGHT, :old.CLOSETNO,
34             null, null, null, null, null, null );
35   ELSE
36     DBMS_OUTPUT.PUT_LINE('This code is not reachable.');
37   END IF;
38 END;/
```

# To-do