HW 2: 3.2, 3.4, 3.5, 3.6, 3.7, 3.12, 3.15, 3.16

3.2) There are 22 distinct tuples in a relation with cardinality 22 because that is what the cardinality of a relationship denotes.

3.4) A candidate key is a minimal set of the fields of the relation that uniquely identifies a tuple of the relation. A primary key is the key chosen by the DB designer to be used as the key that uniquely identifies a tuple. A superkey is any set of fields that contains a candidate key.

3.5) 1. {name}, {age} are not candidate keys. {gpa} is not a candidate key either, as we know from common sense that two students can have the same gpa, but given this instance, we cannot deduce that.

2. A candidate key cannot be determined by looking at one instance of a relation. At any other time, the table can have a completely different set of tuples and we cannot make predictions about those instances based on what is given.

3.6) A foreign key constraint requires that the values on a set X of attributes of a relation R1 must appear as values for the primary key of the referenced relation R2. Foreign key constraints are important because they provide safeguards for insuring the referential integrity of data.

3.7) 1. Foreign key constraints (FKC) aren’t needed for Students, Fculty, Courses, or Rooms relations because they are unique in themselves. In the Enrolled relation, sid and cid should both have FKCs placed on them because students and classes have to correspond. Similarly, the fid and cid fields in the Teachs relation should have FKCs and cid and rno fields in the Meets\_In relation should have FKCs.

2. Some other constraints involve domain constraints. For example, we would limit the characters of the sid to the number of digits a student id number is at the school. Another example would be to have an enumerated type on the grade field to only allow certain letters to be assigned and not for example, a K.

3.12) 1.

CREATE TABLE Professor ( ssn CHAR(9),

PRIMARY KEY (ssn))

CREATE TABLE Courses ( cid CHAR(10),

PRIMARY KEY (cid))

CREATE TABLE Semester ( sid CHAR(10),

PRIMARY KEY (sid))

CREATE TABLE Teaches ( ssn CHAR(9),

cid CHAR(10),

sid CHAR(10),

PRIMARY KEY (ssn, cid, sid),

FOREIGN KEY(ssn) REFERENCES Professor(ssn),

FOREIGN KEY(cid) REFERENCES Courses(cid),

FOREIGN KEY(sid) REFERENCES Semester(sid))

2. CREATE TABLE Professor ( ssn CHAR(9),

PRIMARY KEY (ssn))

CREATE TABLE Courses ( cid CHAR(10),

PRIMARY KEY (cid))

CREATE TABLE Teaches ( ssn CHAR(9),

cid CHAR(10),

sid CHAR(10),

PRIMARY KEY (ssn, cid),

FOREIGN KEY(ssn) REFERENCES Professor(ssn),

FOREIGN KEY(cid) REFERENCES Courses(cid))

3. Same as #2. We cannot enforce any participation constraint with just primary or foreign keys.

4. CREATE TABLE Courses ( cid CHAR(10),

PRIMARY KEY (cid))

CREATE TABLE Professor\_Teaches(ssn CHAR(9),

cid CHAR(10),

sid CHAR(10),

PRIMARY KEY (ssn),

FOREIGN KEY(ssn) REFERENCES Professor(ssn),

FOREIGN KEY(cid) REFERENCES Courses(cid))

5. Same as #4. The only difference now is that the Course table is not necessary because a professor teaches every course, so all of the courses will be present in the Professor\_Teaches table.

6.

CREATE TABLE Professor ( ssn CHAR(9),

PRIMARY KEY (ssn))

CREATE TABLE Courses ( cid CHAR(10),

PRIMARY KEY (cid))

CREATE TABLE Group ( gid CHAR(10),

PRIMARY KEY (gid))

CREATE TABLE Member\_Of ( ssn CHAR(9),

gid CHAR(10),

PRIMARY KEY (ssn, gid),

FOREIGN KEY(ssn) REFERENCES Professor(ssn),

FOREIGN KEY(gid) REFERENCES Group(gid))

CREATE TABLE Teaches ( gid CHAR(10),

cid CHAR(10),

sid CHAR(10),

PRIMARY KEY (ssn, cid),

FOREIGN KEY(ssn) REFERENCES Professor(ssn),

FOREIGN KEY(cid) REFERENCES Courses(cid))

3.15)

CREATE TABLE Musicians ( ssn CHAR(10),

name CHAR(30),

PRIMARY KEY (ssn))

CREATE TABLE Instruments ( instrId CHAR(10),

dname CHAR(30),

key CHAR(5),

PRIMARY KEY (instrId))

CREATE TABLE Plays ( ssn CHAR(10),

instrId INTEGER,

PRIMARY KEY (ssn, instrId),

FOREIGN KEY (ssn) REFERENCES Musicians(ssn),

FOREIGN KEY (instrId) REFERENCES Instruments(instrId))

CREATE TABLE Songs Appears ( songId INTEGER,

author CHAR(30),

title CHAR(30),

albumId INTEGER NOT NULL,

PRIMARY KEY (songId),

FOREIGN KEY (albumId) REFERENCES Album Producer(albumId))

CREATE TABLE Telephone\_Home ( phone CHAR(11),

address CHAR(30),

PRIMARY KEY (phone),

FOREIGN KEY (address) REFERENCES Place(address))

CREATE TABLE Lives ( ssn CHAR(10),

phone CHAR(11),

address CHAR(30),

PRIMARY KEY (ssn, address),

FOREIGN KEY (phone, address) REFERENCES Telephone\_Home(phone, address),

FOREIGN KEY (ssn) REFERENCES Musicians(ssn))

CREATE TABLE Place (address CHAR(30))

CREATE TABLE Perform ( songId  INTEGER,

ssn  CHAR(10),

PRIMARY KEY (ssn, songId),

FOREIGN KEY (songId) REFERENCES Songs(songId),

FOREIGN KEY (ssn) REFERENCES Musicians(ssn))

CREATE TABLE Album Producer( albumIdentifier INTEGER,

ssn CHAR(10),

copyrightDate DATE,

speed INTEGER,

title CHAR(30),

PRIMARY KEY (albumIdentifier),

FOREIGN KEY (ssn) REFERENCES Musicians(ssn))

3.16)

CREATE TABLE Expert ( ssn CHAR(11),

model\_no INTEGER,

PRIMARY KEY (ssn, model\_no),

FOREIGN KEY (ssn) REFERENCES Technician,

FOREIGN KEY (model\_no) REFERENCES Models(model\_no))

The participation constraint cannot be captured in the Expert table.

CREATE TABLE Models ( Model\_no INTEGER,

capacity INTEGER,

weight INTEGER,

PRIMARY KEY (model\_no))

CREATE TABLE Employees ( ssn CHAR(11),

union\_mem\_no INTEGER,

PRIMARY KEY (ssn))

CREATE TABLE Technician\_emp ( ssn CHAR(11),

name CHAR(20),

address CHAR(20),

phone\_no CHAR(14),

PRIMARY KEY (ssn),

FOREIGN KEY (ssn) REFERENCES Employees(ssn)

ON DELETE CASCADE)

CREATE TABLE Traffic\_control\_emp ( ssn CHAR(11),

exam\_date DATE,

PRIMARY KEY (ssn),

FOREIGN KEY (ssn) REFERENCES Employees(ssn)

ON DELETE CASCADE)

CREATE TABLE Plane\_Type ( reg\_no INTEGER,

Model\_no INTEGER,

PRIMARY KEY (reg\_no),

FOREIGN KEY(model\_no) REFERENCES Models(model\_no))

CREATE TABLE Test\_info ( FAA\_no INTEGER,

ssn CHAR(11),

reg no INTEGER,

hours INTEGER,

date DATE,

score INTEGER,

PRIMARY KEY (ssn, reg no, FAA no),

FOREIGN KEY(reg\_no) REFERENCES Plane\_Type(reg\_no),

FOREIGN KEY(FAA\_no) REFERENCES Test(FAA\_no),

FOREIGN KEY(ssn) REFERENCES Technician\_emp(ssn)) CONSTRAINT MODEL CHECK (

SELECT \* FROM Expert, Type

WHERE Expert.ssn = ssn AND

Expert.model\_no = Type.model\_no AND

Type.reg\_no = reg\_no)

The last constraint makes sure that tests on a plane must be conducted by a technician who is an expert on that model.