Ace Cassidy-HW2-CPTS451-Spring ’20

Q1

```
-- post
CREATE TABLE posts(
  -- popularity COMPUTED
  ID int PRIMARY KEY,
  kind char,
  ts timestamp,
  content varchar,
  userID int NOT NULL, -- publishes
  FOREIGN KEY(userID) REFERENCES users(ID)
);


-- about
CREATE TABLE about (
  PRIMARY KEY(postID, courseworkID),
  postID int,
  courseworkID int,
  FOREIGN KEY(postID) REFERENCES posts(ID),
  FOREIGN KEY(courseworkID) REFERENCES courseworks(ID)
);


-- responds
CREATE TABLE responds (
  rToPost int,
  rForPost int,
  PRIMARY KEY(rToPost, rForPost),
  FOREIGN KEY(rToPost) REFERENCES posts(ID),
  FOREIGN KEY(rForPost) REFERENCES posts(ID)
);


-- likes
CREATE TABLE likes (
  postID int,
  userID int,
  PRIMARY KEY(postID, userID),
  FOREIGN KEY(postID) REFERENCES posts(ID),
  FOREIGN KEY(userID) REFERENCES users(ID)
);
```

```sql
-- users
CREATE TABLE users (
  ID int PRIMARY KEY,
  firstName varchar(50),
  lastName varchar(50),
  email varchar(50)
);


-- phones
CREATE TABLE phones (
  PRIMARY KEY(ID, num),
  num varchar(11),
  type char,
  userID int,
  FOREIGN KEY(userID) REFERENCES users(ID)
);


-- instructors
CREATE TABLE instructors (
  userID int PRIMARY KEY,
  title varchar,
  FOREIGN KEY(userID) REFERENCES users(ID)
);


-- students
CREATE TABLE students (
  userID int PRIMARY KEY,
  major varchar(4),
  FOREIGN KEY(userID) REFERENCES users(ID)
);


-- submits
CREATE TABLE submits (
  assignmentID int,
  studentID int,
  PRIMARY KEY(assignmentID, studentID),
  FOREIGN KEY(assignmentID) REFERENCES assignments(courseworkID),
  FOREIGN KEY(studentID) REFERENCES students(userID)
);
```

```sql
-- takes
CREATE TABLE takes (
  examID int,
  studentID int,
  score int,
  PRIMARY KEY(examID, studentID),
  FOREIGN KEY(examID) REFERENCES exams(courseworkID),
  FOREIGN KEY(studentID) REFERENCES students(userID)
);


-- courseworks
CREATE TABLE courseworks(
  ID int PRIMARY KEY,
  title varchar,
  courseNum int,
  sectionNum int,
  term int,
  year int,
  major varchar(4),
  FOREIGN KEY(courseNum, sectionNum, term, year, major) REFERENCES classes(courseNum, se
);


-- assignments
CREATE TABLE assignments (
  courseworkID int PRIMARY KEY,
  deadline datetime,
  FOREIGN KEY(courseworkID) REFERENCES courseworks(ID)
);


-- exams
CREATE TABLE exams (
  courseworkID int PRIMARY KEY,
  examTime time,
  examLocation varchar,
  FOREIGN KEY(courseworkID) REFERENCES courseworks(ID)
);
```

```sql
-- classes
CREATE TABLE classes(
  PRIMARY KEY(courseNum, sectionNum, term, year, major),
  enrollmentLimit int,
  endDate date,
  startDate date,
  courseNum int,
  sectionNum int,
  term int,
  year int,
  major varchar(4),
  FOREIGN KEY(courseNum) REFERENCES courses(num),
  FOREIGN KEY(major) REFERENCES courses(major)
);


-- enrolls_in
CREATE TABLE enrolls_in (
  grade char,
  PRIMARY KEY(
    courseNum,
    sectionNum,
    term,
    year,
    major,
    studentID
  ),
  courseNum int,
  sectionNum int,
  term int,
  year int,
  major varchar(4),
  FOREIGN KEY(courseNum, sectionNum, term, year, major) REFERENCES classes(
    courseNum,
    sectionNum,
    term,
    year,
    major
  ),
  studentID int,
  FOREIGN KEY(studentID) REFERENCES students(userID)
);
```

```sql
-- teaches
CREATE TABLE teaches(
  PRIMARY KEY(
    courseNum,
    sectionNum,
    term,
    year,
    major,
    instructorID
  ),
  courseNum int,
  sectionNum int,
  term int,
  year int,
  major varchar(4),
  FOREIGN KEY(courseNum, sectionNum, term, year, major) REFERENCES classes(
    courseNum,
    sectionNum,
    term,
    year,
    major
  ),
  instructorID int,
  FOREIGN KEY(instructorID) REFERENCES instructors(userID)
);


-- courses
CREATE TABLE courses (
  major varchar(4),
  courseNum int,
  title varchar,
  level int
);


-- prerequisites
CREATE TABLE prerequisites (
  pOfMajor varchar(4),
  pOfCourseNum int,
  pForMajor varchar(4),
  pForCourseNum int,
  PRIMARY KEY(pOfCourseNum, pOfMajor, pForCourseNum, pForMajor),
  FOREIGN KEY(pOfMajor, pOfCourseNum) REFERENCES courses(major, courseNum),
  FOREIGN KEY(pForMajor, pForCourseNum) REFERENCES courses(major, courseNum),
);

-- Could not enforce total participation for enrolls_in and teaches
```

Q2

a

i.

A	B	C
a1	b10	s100
a2	b10	s300
a2	b20	s200
a3	b10	s500
a4	b20	s100
a4	b10	s100

ii.

J	K	L
s200	d20	22
s500	d50	55
s200	d30	11

iii.

M	N	O	P
a1	b10	d10	25
a1	b10	d20	5
a2	b10	d20	20
a2	b20	d20	15
a3	b10	d40	15
a4	b20	d40	5

M	N	O	P
a4	b20	d50	10
a4	b10	d50	5

iv.

cannot be inserted, duplicate primary keys not allowed

b

i.

S	T	U
s100	20	555
s200	20	333
s300	30	111
s500	40	444

ii.

A	B	C
a1	b10	s100
a2	b10	s300
a2	b20	s200
a4	b20	s100

M	N	O	P
a1	b10	d10	25
a1	b10	d20	5
a2	b10	d20	20
a2	b20	d20	15

M	N	O	P
a4	b20	d40	5
a4	b20	d50	10
a4	b10	d50	5

iii.

D	E	F
d10	50	100
d30	150	300
d40	75	400
d50	100	200

M	N	O	P
a1	b10	d10	25
a3	b10	d40	15
a4	b20	d40	5
a4	b20	d50	10

iv.

A	B	C
a1	b20	s100
a2	b10	s300
a2	b20	s200
a3	b10	s500
a4	b20	s100

M	N	O	P

M	N	O	P
NULL	NULL	NULL	NULL
NULL	NULL	NULL	NULL
a2	b10	d20	20
a2	b20	d20	15
a3	b10	d40	15
a4	b20	d40	5
a4	b20	d50	10

v.

S	T	U
s600	60	666
s200	20	333
s300	30	111
s400	30	555
s500	40	444

A	B	C
a1	b10	NULL
a2	b10	s300
a2	b20	s200
a3	b10	s500
a4	b20	NULL