S1 The "Cheat Sheet" Part (20 points)

Give a simple definition or an example for the following terms/concepts.

Note: This must be written in your own words. Anything copied from the slides/text will be considered as incorrect.

1. Referential Integrity

If a value appears in a foreign key column of one table, it must also appear as a value in the Primary Key of the table it has a relationship with.

2. Primary Key

The unique identifier for a row. Must be unique and no subset is unique.

3. Foreign Key

A column in one table that represents a relationship with the Primary Key of another

4. Null

Absence of data. Not 0 nor spaces nor False.

5. View

A dynamic table-like structure for seeing data through a query.

6. Strong and Weak Entity Type

Strong – does not depend on another entity for its existence.

Weak – depends on another entity for its existence

7. Fan Trap

A situation where one entity has a many relationship with two other entities, but a relationship between these other entities is necessary. Both "end" entity rows can find their "middle" entity counterpart, but cannot find the counterpart for the other "end" entity.

8. Chasm Trap

Where participation with a entity is not required, but if a row has no counterpart, it is still necessary to find its counterpart with a third entity.

9. Base Relation

A table in a database

10. Difference between Functional Dependency and Full Functional Dependency

Functional Dependency – One set of columns (determinant) determines the values in another set of columns

Full – a set of columns is not dependent on any subset of columns of the determinant

S2 Abby someone. Abby who?... Abby... Normal. (20 points)

Beginning with the following un-normalized dataset in a relation called DepartmentCourse, produce a 3NF version of the database with the same data. You must show the following:

1. Identify a Candidate Key for the DepartmentCourse relation

D, E, G combination (no single column or double column options exist)

2. Functional dependencies (For simplicity, you may use the lettered columns above the column names for this part)

A -> B; D -> C; E -> F, J; I -> H; E, G -> H, I (others exist too)

3. Identify any partial dependencies that exist.

Anything where D, E, or G alone (or two of them) is a determinant

4. Identify any transitive dependencies that exist

E, G -> H, I and I -> H; E -> A, B and A -> B

5. Identify Primary Keys and any Foreign Keys in your final 3NF relations

See Homework assignment 4 for a solution to the tables and PKs and FKs.

DepartmentCourse

A	B	C	D	E	F	G	H	I	J
Dept	**Dept Name**	**Faculty Name**	**Faculty Email**	**Course ID**	**Course Name**	**Section**	**Instructor**	**Instructor Email**	**Course Capacity**
CSE	Computer Engineering	Melih GUNAY	melihgunay	CSE204	Database	A	Joseph LEDET	josephledet	50
CSE	Computer Engineering	Murat AK	muratak	CSE204	Database	A	Joseph LEDET	josephledet	50
CSE	Computer Engineering	Joseph LEDET	josephledet	CSE204	Database	A	Joseph LEDET	josephledet	50
CSE	Computer Engineering	Melih GUNAY	melihgunay	CSE102	Intro to Java II	A	Murat AK	muratak	60
CSE	Computer Engineering	Murat AK	muratak	CSE102	Intro to Java II	A	Murat AK	muratak	60
CSE	Computer Engineering	Joseph LEDET	josephledet	CSE102	Intro to Java II	A	Murat AK	muratak	60
CSE	Computer Engineering	Melih GUNAY	melihgunay	CSE102	Intro to Java II	B	Joseph LEDET	josephledet	60
CSE	Computer Engineering	Murat AK	muratak	CSE102	Intro to Java II	B	Joseph LEDET	josephledet	60
CSE	Computer Engineering	Joseph LEDET	josephledet	CSE102	Intro to Java II	B	Joseph LEDET	josephledet	60
MAT	Mathematics	Al CIBRA	alcibra	MAT214	Differential Equations	A	Al CIBRA	alcibra	100

S3 Make Tables Not War (20 points)

Write the necessary SQL to perform the following:

1. Make a table called **Assignment** with the following columns and requirements:
   a. **AssignmentID -** PK, Integer
   b. **CourseID -** FK to the table you created that has Course ID as a PK, character string of 7 characters, Department abbreviation and course number
   c. **Title -** variable character string of 20 characters
   d. The combination of **CourseID** and **Title** must be distinct for different rows.

```
CREATE TABLE Assignment (
     AssignmentID INT NOT NULL,
     CourseID VARCHAR(7),
     Title VARCHAR(20),
     PRIMARY KEY (AssignmentID),
     UNIQUE (CourseID, Title),
     FOREIGN KEY (CourseID) REFERENCES Course(CourseID)
          ON DELETE CASCADE ON UPDATE CASCADE);
```

2. Make a table called **Question** with the following columns and requirements:
   a. **AssignmentID -** PK, FK to Assignment table
   b. **Num -** PK, Integer
   c. **Text -** variable character string of 255 characters
   d. **Points -** Integer

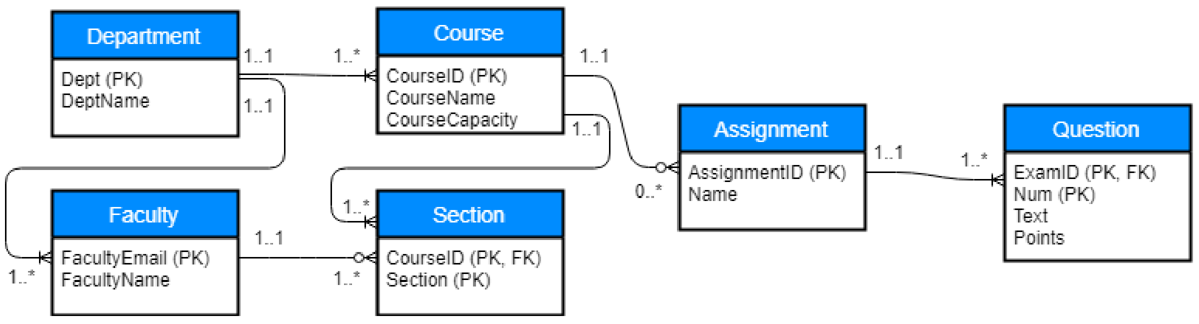
```
CREATE TABLE Question (
     AssignmentID INT NOT NULL,
     Num INT NOT NULL,
     Text VARCHAR(255),
     Points INT,
     PRIMARY KEY (AssignmentID, Num),
     FOREIGN KEY (AssignmentID) REFERENCES Assignment
          ON DELETE CASCADE ON UPDATE CASCADE);
```

3. Add a column to the **Assignment** table to have the total number of points for the assignment; the default value for this field should be 0 (do not populate this field, yet).

```
ALTER TABLE Assignment
     ADD TotalPoints INT DEFAULT 0;
```

S4 E-R, but without Clooney (20 points)

Using the tables **<u>you produced</u>** in Q2 and the tables defined in Q3, draw the E-R diagram. Include in your diagram each of the relations (entities) and the relationship along with participation/cardinality (i.e. 1..*, 0..1, etc.).



S5 SQL With Delight (20 points)

Using the tables **you produced** in Q2 and the tables defined in Q3, write SQL queries to perform the following **(choose 5 of these to answer)**:

1. Identify the unique course IDs, course names, sections, and instructors that have assignments in our database.

```
SELECT DISTINCT c.CourseID, c.CourseName,
                s.Section, f.FacultyName
 FROM Course c, Section s, Faculty f, Assignment a
WHERE c.CourseID = s.CourseID
  AND f.FacultyEmail = s.FK_FacultyEmail
  AND c.CourseID = a.CourseID;
```

2. Give a list of course IDs, assignments, and number of questions for any assignments with at least 5 questions.

```
SELECT a.CourseID, a.Title, COUNT(*) AS QuestionCount
FROM Assignment a, Question q
WHERE a.AssignmentID = q.AssignmentID
GROUP BY a.CourseID, a.Title
HAVING COUNT(*) > 4;
```

3. Remove all assignments for courses in the "HIST" department. (NOTE: if you did not define the action for Referential Integrity in Q3, you will need to also do something with the Question table).

```
DELETE FROM Assignment
WHERE CourseID IN (SELECT CourseID
FROM Course
WHERE FK_Dept = 'HIST');
```

4. Add a new instructor to the "PHYS" department with the name "Ken ETTIK". Also assign him as the instructor to an existing course - "PHYS203" section A.

```
INSERT INTO Faculty (FacultyName, FacultyEmail, FK_Dept)
VALUES ('Ken ETTIK', 'kenettik', 'PHYS');
UPDATE Section
SET FK_FacultyEmail = 'kenettik'
WHERE FK_CourseID = 'PHYS203' AND Section = 'A';
```

5. Change the data in the Assignment table to set the total number of points for each assignment in the column you created in Q3.3.

```
UPDATE Assignment a SET a.TotalPoints = (SELECT
SUM(Points)
FROM Question q
WHERE q.AssignmentID = a.AssignmentID);
```

6. Create a new assignment for "CSE204" called "Assignment01"

```
INSERT INTO Assignment (AssignmentID, CourseID, Name)
VALUES (##NewIntValue##, 'CSE204', 'Assignment01');
```

7. Create new questions for the "CSE204" – "Assignment01" and make them the same as the questions for "CSE102" – "Project01" (i.e. copy the same num, text, and points).

```
INSERT INTO Question (AssignmentID, Num, Text, Points)
SELECT ##NewIntValueFromQ6##, Num, Text, Points
FROM Question
WHERE AssignmentID = (SELECT AssignmentID
FROM Assignment
WHERE CourseID = 'CSE102' AND Title = 'Project01');
```

8. Make a view called AllAssignments that will have the Department Name, Course ID, Assignment Title, and total number of questions for all assignments.

```
CREATE VIEW AllAssignments AS
SELECT d.DeptName, c.CourseID,
a.Title, COUNT(*) AS QuestionCount
FROM Department d, Course c, Assignment a, Question q
WHERE d.Dept = c.Dept AND c.CourseID = a.CourseID
AND a.AssignmentID = q.AssignmentID
GROUP BY d.DeptName, c.CourseID, a.Title
```

9. Make a view called CSE204AssignmentQuestions that will have the Assignment Title, Question Text, and Question Points for all assignments in course “CSE204”.

```
CREATE VIEW CSE204AssignmentQuestions AS
SELECT a.Title, q.Text, q.Points
FROM Assignment a, Question q
WHERE a.AssignmentID = q.AssignmentID AND a.CourseID =
‘CSE204’
```

10. Remove the view AllAssignments from the database.

```
DROP VIEW AllAssignments;
```