CSCI 5707 Midterm

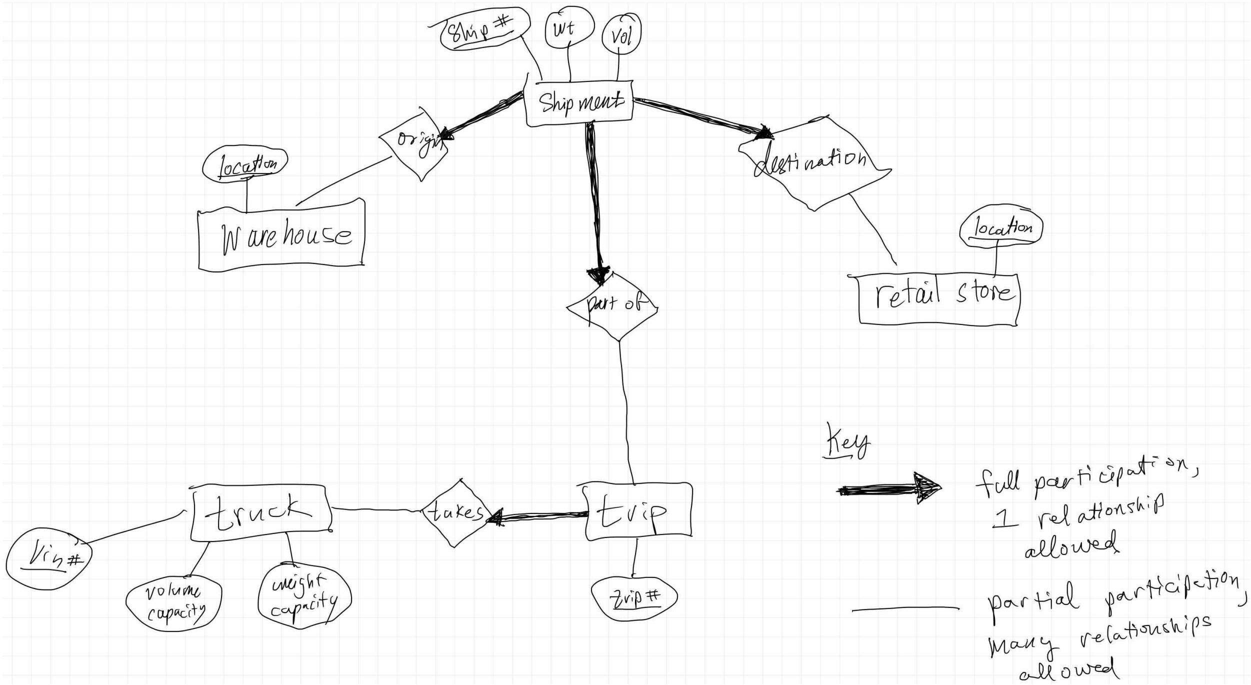
Email	neis@umn.edu
Name:	Kelsey Neis

Question 1

A **trucking company** called TRUCKERS is responsible for picking up shipments for warehouses of a **retail chain** called MAZE BROTHERS and delivering the shipments to individual retail locations of MAZE BROTHERS.

- Currently there are 6 **warehouse** locations and
- 45 **retail stores** of MAZE BROTHERS.
- A truck may carry several shipments during a single **trip**, which is identified by a Trip#, and delivers those shipments to multiple stores.
- Each **shipment** is identified by a Shipment# and includes data on shipment volume, weight, destination, etc.
- **Trucks** have different capacities for both the volumes they hold and the weights
  they can carry. The TRUCKERS company currently has 150 trucks, and a truck makes 3 to 4 trips per week. A database – to be used by both TRUCKERS and MAZE BROTHERS – is being designed to keep track of truck usage and deliveries, and to help in scheduling trucks to provide timely deliveries to stores.

Design an E-R schema showing the above relationships. Indicate what kind of relationships they are. Clearly state any assumptions you make.

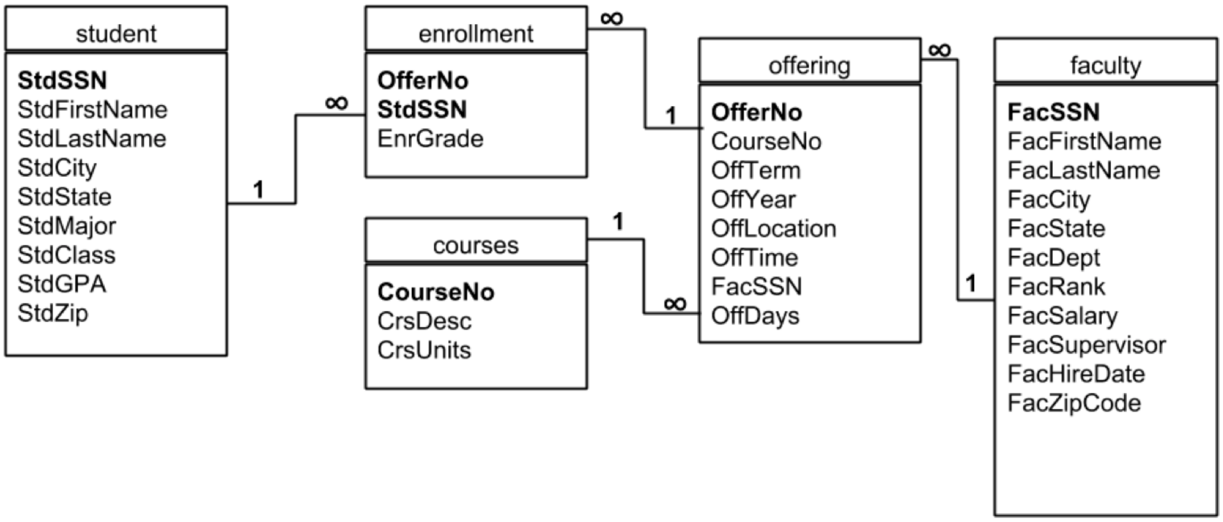


My assumptions:

- Underlined attributes are the unique identifiers
- I added a unique identifier, vin # to truck, since I assume multiple trucks can have the same volume capacity/weight capacity combination
- Since only two companies are involved and their roles are well-defined, it is not necessary to model them in the DB.
- A single shipment has one origin and one destination, whereas a trip can have many shipments, thus many stops at different warehouses and retail stores
- There needs to be a constraint to limit the total wt and vol of all shipments in a single trip, such that those numbers do not exceed the volume capacity and weight capacity of the truck taking the trip. This cannot be represented in the ER diagram, but would be included in the creation of the database.

Question 2

The following questions are based on a database schema described below. The keys are in boldface. Also, the notation 1–oo means the relationship is 1–Many.



(a) Suppose faculty hired after January 1st, 1991 are given a 10% salary raise. Write an SQL query to list the Name, City, current salary and the new salary for faculty hired after January 1st, 1991. (6 pts)

```
SELECT f.FacFirstName AS FirstName, f.FacLastName As LastName,
f.FacCity AS City, f.FacSalary AS CurrentSalary,
f.FacSalary * 1.1 AS NewSalary
FROM faculty f
WHERE f.FacHireDate > '1991-01-01';
```

(b) Write an SQL query to list the offer number, the course number, the first name and the last name of instructors teaching the courses scheduled in Fall 1999, taught by assistant professors. (6 pts)

```
SELECT o.OfferNo, o.CourseNo, f.FacFirstName, f.FacLastName
FROM offering o, faculty f
WHERE o.OffTerm = 'Fall'
AND o.OffYear = 1999
AND f.FacRank = 'Assistant Professor';
```

(**c) Write an SQL query to list number of Computer Science courses offered, grouped by course description. (6 pts)**

```
SELECT c.CrsDesc, COUNT(c.CourseNo)
FROM offering o, courses c, faculty f
```

```
WHERE o.CourseNo = c.CourseNo
AND o.FacSSN = f.FacSSN
AND f.FacDept = 'Computer Science'
GROUP BY c.CrsDesc;
```

(d) Suppose we create a B-tree index on the Offering relation. Which attribute would be a good choice for it? What kind of queries would benefit from it – give an example in English of such a query. (3+3 pts)

A B-tree index on CourseNo could be beneficial. Consider an example where a student wants to find all 5000-level courses. This would be a range search which would benefit greatly from a B-tree index, because it would only require traversing the tree once, then scanning the leaves until the range is found.

Question 3

(a) Explain the exact meaning of the following ER model notation: (solid one-directional arrow, bold)

The fact that the line is bold means that the entity must have full participation in the relationship it's pointing to. The arrow indicates that an entity instance can only partake in one such relationship.

(b) What is an "identifying relationship"? What kind of constraints does it have?

An identifying relationship is relevant for weak entities. Weak entities are only valid in relation to another entity. The Identifying relationship is the one that the weak entity depends on in order to meet the uniqueness constraint. Therefore, all weak entity instances must participate in the identifying relationship.

(c) Distinguish between "aggregation" and "generalization".

With aggregation, an entity participates in a relationship with a group of entities, which is treated as one entity. Generalization, on the other hand, describes the relationship between a subclass and superclass in the direction of subclass to superclass (i.e. ISA relationship). Aggregation gives the possibility of relating one entity to a complex system and storing data about that distinct relationship.

(d) What is a "cascading delete"? What is it useful for?

When table A has a foreign key to another table B, the foreign key should have an action for what to do when a tuple is deleted from B to prevent null pointers. If cascading delete is chosen, then when a tuple is deleted from B, then all tuples in A referencing the deleted tuple will also be deleted.

(e) Give examples to illustrate where cascading delete is a (i) good idea, (ii) bad idea.

i) Cascading
delete is a good idea if the referring tables do not have additional
data that needs to be preserved even if a tuple is deleted. For example,
if a student is deleted from the students table, it may be ok to
cascade to the enrolled table.

ii) It is a bad idea if the above is not the case, or if the database
system has deep relationships where a cascade delete may lead to many
layers of deletions. In the student example, if there is billing data,
then we don't want to delete that without maybe sending the student an
invoice and closing the account.

(f) Distinguish between a "key" and a "super key". Show an example.

A key is a set of one or more fields that together make a tuple unique in a relation such that no subset of the fields are unique. If there is a subset that make the tuple unique, then the key is called a superkey. In other words, a superkey has a subset of fields which form a key.

Question 4

The following questions are based on a database schema described below. The keys are underlined.

Student(Name, StudentNumber, Class, Major)

Course(CourseName, CourseNumber, CreditHours, Department)

Prerequisite(CourseNumber, PrerequisiteNumber)

Section(SectionIdentifier, CourseNumber, Semester, Year, Instructor)

Grade_Report(StudentNumber, SectionIdentifier, Grade)

Please write SQL queries for each of the following queries.

(a) For each section taught by Prof. Krishna, retrieve the course number, semester, year, and number of students who were in the section. (6pts)

```
SELECT s.CourseNumber, s.Semester, s.Year, COUNT(g.StudentNumber) AS scount
FROM Section s, Grade_Report g
WHERE s.Instructor = 'Prof. Krishna'
AND s.SectionIdentifier = g.SectionIdentifier
GROUP BY s.SectionIdentifier;
```

(b) Retrieve the name and major of all students who do not have a grade of A in any of their courses. (6pts)

```
SELECT s.Name, s.Major
FROM Student s
--Make sure the student has taken at least one class
WHERE EXISTS(SELECT *
  FROM Grade_Report g
  WHERE g.StudentNumber = s.StudentNumber)
AND NOT EXISTS (SELECT g.StudentNumber
  FROM Grade_Report g
  WHERE g.StudentNumber = s.StudentNumber
  AND g.Grade = 'A');
```

Assumption: Students who have not taken any classes are not relevant

(c) Retrieve the name and major of all straight-A students (students who have a grade of A in all their courses). (6pts)

```
SELECT s.Name, s.Major
FROM Student s
--Make sure the student has taken at least one class
WHERE EXISTS(SELECT *
  FROM Grade_Report g
  WHERE g.StudentNumber = s.StudentNumber)
WHERE NOT EXISTS (SELECT g.StudentNumber
  FROM Grade_Report g
  WHERE g.StudentNumber = s.StudentNumber
  AND g.Grade <> 'A');
```

Assumption: Students who have not taken any classes are not relevant

(d) Write the SQL statement to create a unique clustering index on the StudentNumber attribute of the Student relation. What type of query will become more efficient because of this index? (6pts)

```
CREATE UNIQUE CLUSTERED INDEX student_no
ON Student (StudentNumber)
```

Queries searching for a single StudentNumber (equality search) will be made much faster. Range queries on StudentNumber involving a small portion of the total number of Students will also be much faster.