Database Systems

CSCI-GA.2433-001 – Fall 2022

Dr. Jean-Claude Franchitti

New York University

Computer Science Department

Courant Institute of Mathematical Sciences

Sample Midterm Examination

This midterm exam covers introductory database systems topics, the relational data model and relational database constraints, enterprise data modeling using the entity-relationship model, practical relational database design, relational algebra and relational/domain calculus topics, and some SQL DDL/DML/DQL topics.

Problems:

1. In a relational database management system, a rule that ensures that every record in a table is unique is called a …

   (a) candidate key constraint
   (b) referential integrity constraint
   (c) key constraint
   (d) participation constraint

2. Which of the following is allowed for relations?

   (e) There can be a field with a non-atomic value.
   (f) The columns can be in any order.
   (g) Some of the values of a field can come from a domain that is not the same as the field's domain.
   (h) Two rows may be identical.

3. Briefly describe the difference between:

DML and DDL

DML (data manipulation language): operations to add, modify, retrieve data
DDL (data definition language): operations to define or modify the schema and other administrative operations

a. key and partial key (with respect to E/R modeling)

key: uniquely identifies entities
partial key: applies only to weak entities. A partial key must be combined with the "owning" entity's key to get a unique key for the weak entity.

b. atomic and non-atomic value

atomic: a non-divisible piece of data (e.g. an elementary data type)
non-atomic: has some visible internal structure (multiple values, nested attributes, etc.)

c. schema and instance

schema: describes the structure only
instance: actual data

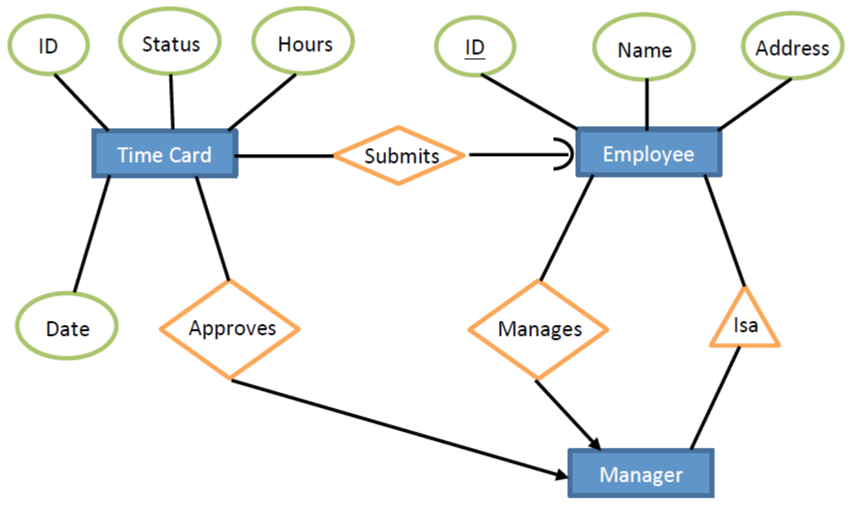
d. natural join vs joins in general?

natural join: joined columns have the same name (don't have to in general join); it is not necessary to specify the name (is necessary in a general join); and the superfluous columns are automatically removed (not so in general).

4. Bert the Payroll Guy is about to retire after 40 years and it's time to replace his manual time card system with some sort of computerized database. You have been asked to come up with the database design. As best we can tell, the time card system has the following properties:

- A *timecard* contains hours worked and date submitted
- Each *timecard* is associated with exactly one *employee*
- Each *timecard* has a unique id
- Each *timecard* has a status: approved, not approved, or pending (not examined yet)
- Each *employee* has a name, address, and a unique id
- Each *employee* submits a time card every pay period. i.e. In 1 year, they will submit multiple time cards
- Each *employee* is associated with exactly one *manager*
- Each *manager* is also an *employee*
- Each *manager* is in charge of one or more employees
- Each *manager* approves time cards for one or more employees

Draw an ER diagram that captures this information



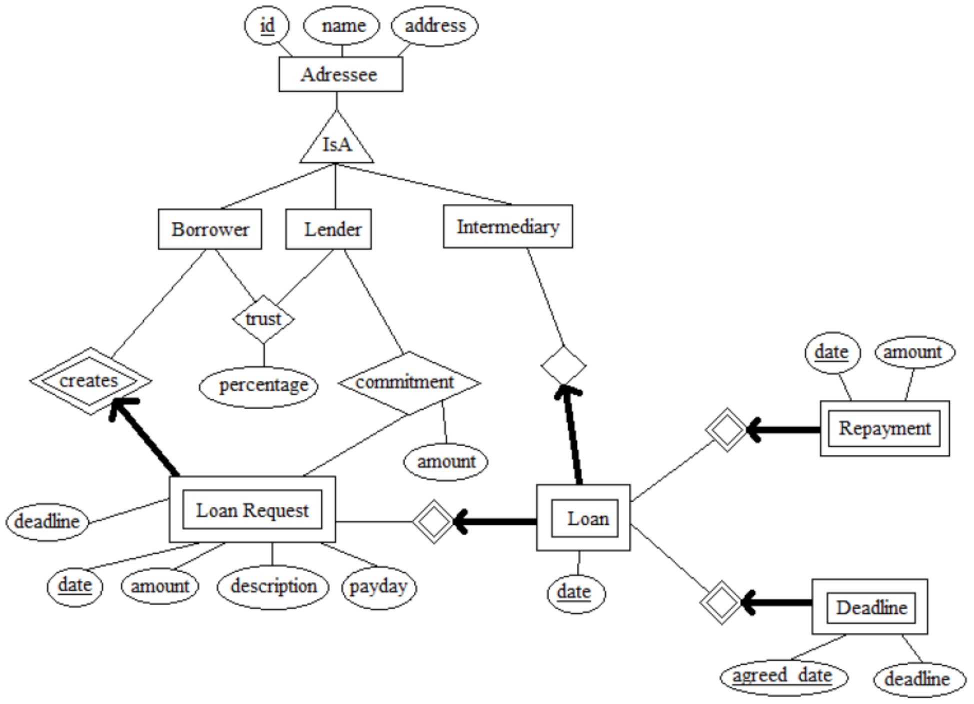
5. Micro loans are small loans, which is beginning to gain popularity especially among borrowers in developing countries. The idea is to bring venture lenders together using information technology. Typically, the loans will be used to finance startup or development of the borrower's company, so that there is a realistic chance for repayment. The money ina loan can, unlike traditional loans, come from many lenders. In this problem, you must create an E-R model that describes the information necessary to manage micro loans. The following information form the basis for creating the model:

   - Each borrower and lender must be registered with information about name and address.
   - A loan starts with a loan request, which contains information about when the loan should at latest be granted, The total amount being discussed (US-dollars), and how long the payback period is. Also, a description is included of how the money will be used. The rent on the payment is calculated in the loan amount, which is to say, the full amount is not paid.
   - Lenders can commit to an optional portion of the total amount of a loan request.
   - When the commitments for the loan request covers the requested amount, the requestis converted to a loan. If not enough commitments can be reached, the loan requestis cancelled. A borrower can have more than one request, and more than one loan ata time, but can at most make one request per day.
   - The loan is paid through an "intermediary", typically a local department of a charity,who has a name and an address.
   - The borrower chooses when he or she will make a payment. Every payment must be registered in the database with an amount and a date (at most one payment per loan per day). The lenders share the repayment based on how large a part of the loan they are responsible for.
   - If the loan is not repaid before the agreed upon deadline, a new date is agreed. The database

must not delete the old deadline, but save the history (the deadline can be overridden multiple times).

- Each lender can for each burrower save a “trust”, which is a number between 0 and 100 that determines the lender’s evaluation of the risk of lending money to that person. The number must only be saved for the borrowers, for whom there has been made such an evaluation.

Make an E-R model for the data described above. If you make any assumptions about data that does not show from the problem, they must be described. Put an emphasis on having the model express as many properties about the data as possible, for instance participation constraints.



6. Convert your E/R diagram from question 5 above into relations, and write SQL statements to create the relations. You may make any reasonable choice of data types. Remember to include any constraints that follow from the description of the data set or your E/R diagram, including primary key and foreign key constraints.

   Note: Describe at least two of the relations using SQL DDL (make reasonable assumptions about data types and state the relation schemas for the other relations. Include primary keys and foreign key constraints being for all relations. It is not necessary to state CHECK constraints and the like.

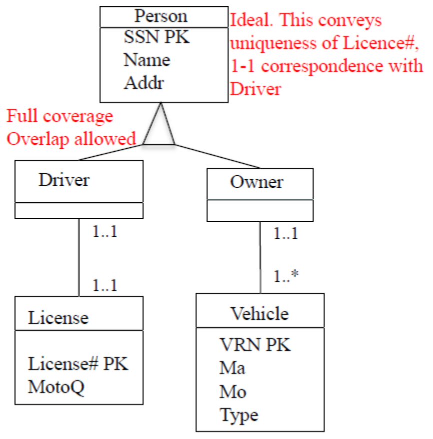
It is assumed below that borrowers, lenders, and intermediaries are disjoint entities.

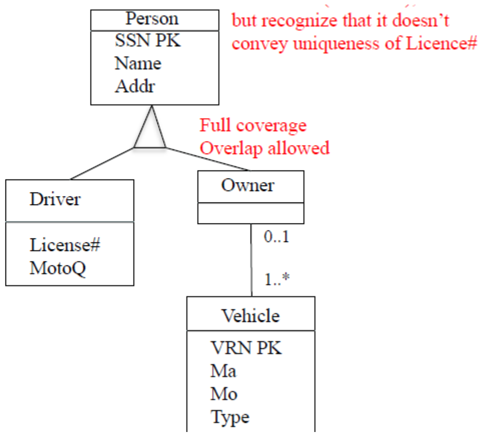
```
CREATE TABLE Adressee (
     id INT PRIMARY KEY,
     type ENUM('borrower', 'lender', 'intermediary'), name
     VARCHAR(50),
     address VARCHAR(50)
);


CREATE TABLE Trust (
     borrower INT REFERENCES Adressee(id), lender INT REFERENCES
     Adressee(id), percentage INT,
     PRIMARY KEY (borrower,lender)
);


CREATE TABLE LoanRequest (
     id  INT  REFERENCES  Adressee(id), date,
     amount INT,
     description VARCHAR(1000), payday DATE,
     deadline DATE, PRIMARY KEY (id,date)
);


CREATE TABLE Commitment (
     lender INT REFERENCES Adressee(id), borrower INT,
     loanrequestDate DATE,
     FOREIGN KEY (borrower,loanrequestDate) REFERENCES
     LoanRequest(id,dato), amount INT,
     PRIMARY KEY (lender, borrower, loanrequestDate,amount)
);


CREATE TABLE Loan (
     id  INT, RequestDate DATE, date,
     intermediary REFERENCES Adressee(id),
     FOREIGN KEY (id,RequestDate) REFERENCES
     LoanRequest(id,date), PRIMARY KEY(date,id,RequestDate)
);

CREATE TABLE Repayment (
     id INT,
     date,
     RequestDate DATE, amount INT,
     FOREIGN KEY (id,RequestDate) REFERENCES
     LoanRequest(id,date), PRIMARY KEY (date,id,RequestDate)
);


CREATE TABLE Deadline ( id INT,
     agreedDate DATE, RequestDate DATE, deadline DATE,
     FOREIGN KEY (id,RequestDate) REFERENCES
```

```
        LoanRequest(id,date), PRIMARY KEY
        (agreedDate,id,RequestDate)
    );
```

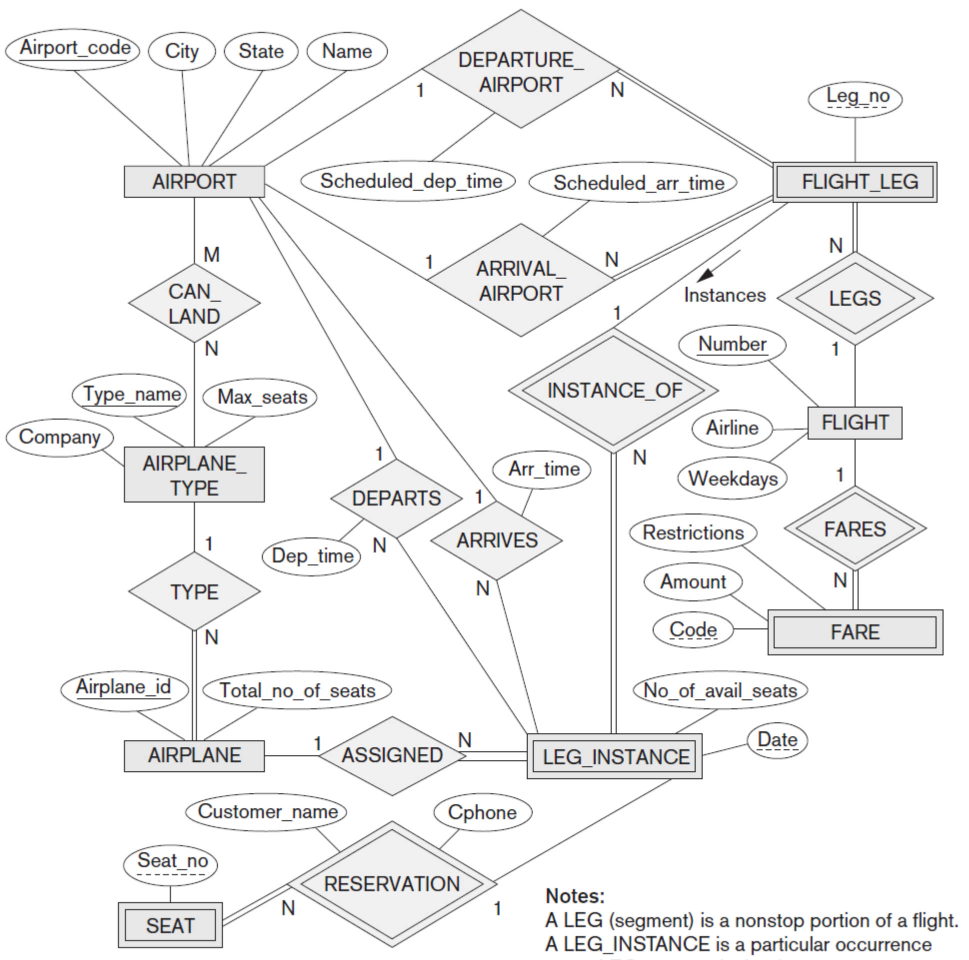
7. Give a UML diagram that captures the following information for a simplified DMV/TDOT-like database. Your UML should include a hierarchy of subclasses that are complete coverage and overlap allowed.

   - A Person can be a licensed vehicle driver and/or a vehicle owner. A person is uniquely identified by SSN, and is also described by name and address. Only persons that are licensed drivers and/or owners are to be recorded in the DB (a driver need not own a vehicle, and an owner need not be a licensed driver).
   - Each vehicle is identified uniquely by vehicle registration number (VRN), with other attributes: make, model, and type (e.g., motorcycle, car, truck).
   - Each vehicle is owned by AT MOST one person (i.e., owner).
   - A driver has exactly one driver's license
   - Each license is associated with exactly one driver, and has an attribute license number that uniquely identifies the license and an (Boolean) attribute MotorcycleQualified (MotoQ)





8. Answer the following questions with respect to the AIRLINE relational database schema shown below:



Notes:
A LEG (segment) is a nonstop portion of a flight.
A LEG_INSTANCE is a particular occurrence of a LEG on a particular date.

Figure 3.21
An ER diagram for an AIRLINE database schema.

Give an example (of relations, relation instances, tuples, or whatever is appropriate) to illustrate each of the following concepts: (Lots of possible examples. Only one is needed for each +question.)

- a "foreign key"

  FLIGHT_NUMBER within FLIGHT_LEG (or within LEG_INSTANCE, FARES, SEAT_RESERVATION)

  AIRPLANE_TYPE_NAME within CAN_LAND (or within AIRPLANE)

  AIRPLANE_ID within LEG_INSTANCE

  AIRPORT_CODE within FLIGHT_LEG, etc.

  There are others involving compound keys.

- a violation of "referential integrity."

  Example of a violation: suppose FLIGHT_LEG contains a tuple with a FLIGHT_NUMBER of #477, but that FLIGHT has no tuple with that value. Since FLIGHT_NUMBER is a foreign key, this violates referential integrity.

- a violation of a "key constraint."

  Suppose AIRPORT had two different tuples with an AIRPORT_CODE of "LAX". Since AIRPORT_CODE is a key, this violates the key constraint (which is that keys are unique).

  or you could give any key a null value; that is another key constraint violation.

9. Considering the AIRLINE relational database schema outlined in question 5 above, write the following queries in each of the languages as requested:

   List all flights which originate in (1st leg leaves from) Spokane. [RA]

   P FLIGHT_NUMBER (S LEG_NUMBER=1 (FLIGHT_LEG) JN DEPARTURE_AIRPORT_CODE=AIRPORT_CODE (S CITY="SPOKANE" (AIRPORT)))

   List types of airplanes which cannot land in Spokane. [RA]

   The idea in both versions is to find all the planes that can land in Spokane, and subtract that from all the types of planes.

   P TYPE_NAME (AIRPLANE_TYPE) - P AIRPLANE_TYPE_NAME (CAN_LAND * (S CITY="SPOKANE" (AIRPORT)))

   List types of airplanes which cannot land in Spokane. [SQL]

   The SQL version could be written using the EXCEPT operator, or without it as follows:

```
SELECT TYPE_NAME

FROM AIRPLANE_TYPE

WHERE TYPE_NAME NOT IN

          (SELECT AIRPLANE_TYPE_NAME

          FROM CAN_LAND, AIRPORT
```

```
                    WHERE CAN_LAND.AIRPORT_CODE =
                    AIRPORT.AIRPORT_CODE AND

                    AND CITY="SPOKANE")
```

List types of airplanes which CAN land in Spokane. [Tuple RC]

{t.AIRPLANE_TYPE_NAME | CAN_LAND(t) AND (∃ u) (AIRPORT(u) and t.AIRPORT_CODE=u.AIRPORT_CODE and u.CITY="SPOKANE")}

How many American Airlines planes took off exactly on time on 10/31/97? [SQL]

```
SELECT COUNT (*)

FROM FLIGHT, FLIGHT_LEG L, LEG_INSTANCE I

WHERE AIRLINE="AMERICAN AIRLINES"

                    AND FLIGHT.NUMBER = F.FLIGHT_NUMBER

                    AND F.FLIGHT_NUMBER=I.FLIGHT_NUMBER

                    AND F.LEG_NUMBER=I.LEG_NUMBER

                    AND SCHEDULED_DEPARTURE_TIME=DEPARTURE_TIME
```

List the companies which might compete well with Boeing. That is, for each type of plane made by Boeing, these companies make a plane with exactly the same seating capacity as the Boeing plane. [RA]

This is a typical division query. It is not required that the airplane types be the same (in fact, one might expect them to be different for different companies), so type needs to be removed before the division is done.

P MAX_SEATS,COMPANY (AIRPLANE_TYPE) DIVIDED-BY

P MAX_SEATS (S COMPANY="BOEING" (AIRPLANE_TYPE))

Prepare a list showing how many airports there are in each state contained in the database. [SQL]

```
SELECT STATE, COUNT (*)

FROM AIRPORT

GROUP BY STATE
```

10. The following database contains information about actors, plays, and rolesperformed.

Actor(<u>actor_id</u>, name, year_born)

Play(<u>play_id</u>, title, author, year_written)

Role(<u>actor_id</u>, <u>character_name</u>, <u>play_id</u>)

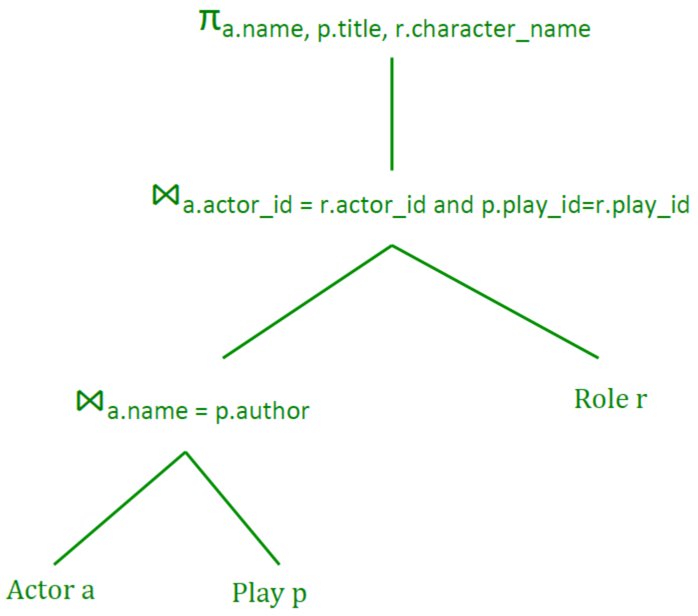
Where:

- Actor is a table of actors, their names, and the year they were born. Each actor has a uniqueactor_id, which is a key.
- Play is a table of plays, giving the title, author, and year written for each play. Each play has aunique play_id, which is a key.
- Role records which actors have performed which roles (characters) in which plays. Attributes actor_id and play_id are foreign keys to Actor and Play respectively. All three attributes make up the key since it is possible for a single actor to play more than one character in the same play.

The following query returns information about all persons who haveacted in a play that they have written:

```
SELECT a.name, p.title, r.character_name
FROM Actor a, Play p, Role r
Where a.name = p.author AND a.actor_id = r.actor_id AND p.play_id = r.play_id
```

Give a relational algebra query plan drawn as a tree that correctly computes this query.


$\pi_{\text{a.name, p.title, r.character\_name}}$
$\bowtie_{\text{a.actor\_id = r.actor\_id and p.play\_id=r.play\_id}}$
$\bowtie_{\text{a.name = p.author}}$
Role r
Actor a
Play p