Exercise 19.5 Consider the following collection of relations and dependencies. Assume that each relation is obtained through decomposition from a relation with attributes *ABCDEFGHI* and that all the known dependencies over relation *ABCDEFGHI* are listed for each question. (The questions are independent of each other, obviously, since the given dependencies over *ABCDEFGHI* are different.) For each (sub)relation: (a) State the strongest nonnal fonn that the relation is in. (b) If it is not in BCNF, decompose it into a collection of BCNF relations.

1. *R1(A. C,B.D,E), A → 13, C → D*
2. *R2(A,B,F), AC → B, B → F*
3. *R3(A,D), D → G, G → H*
4. *R4(D, C,H, G), A → I, I → A*
5. *R5(A.I,C.B)*

```
NORMALIZATION: An example
-------------------------

I. The algorithm for translating into 3NF
-----------------------------------------

Here is the algorithm:

INPUT: a relation R and a set of FDs F
OUTPUT: a collection of relations in normal form

Step 1. Transform F into a minimal cover G using the algorithm on p. 626.
Step 2. Use the algorithm of Section 19.6.1. to transform relation R into
        a collection of relatios R1, ..., Rn that are in 3NF, using the FDs
in G.
Step 3. For each FD X-->A that is not preserved by the decomposition obtained
        in Step 3, add relation XA to the collection R1,...,Rn.
Step 4. Output the final collection obtained in Step 3.

Note: this algorithm ouputs a collection of relations in 3NF that are
      lossless-join and dependency preserving.
```

II. Example

A typical task that arises in databases is the following: given a database instance, tell whether there are problems related to updating, inserting or deleting with respect to that instance. If there are problems (usually related to update, insertion and deletion anomalies), you may be asked to solve them by normalization, i.e. decomposition of the schema of the original relation instance into a set of smaller schemas.

So consider the following instance of the EMPLOYEES relation:

eid	name	seniority	bonus	rating	sal	deptid	deptname	deptaddress
1	Joe	12	8K	2	100K	5	justice	Rideau
2	Mills	4	3K	4	60K	5	justice	Rideau
2	Mills	4	3K	4	60K	5	justice	Rideau
3	jim	10	7K	2	100K	5	justice	Rideau
4	Moll	4	3K	4	60K	4	defense	Laurier
5	Ann	12	8K	2	100K	5	justice	Rideau
6	Mae	4	3K	4	60K	4	defense	Laurier

By observing this instance, we see several repeating groups of values.

Functional dependencies (FDs) that we can discover from this instance are:

1) eid ---> name seniority bonus rating sal deptid deptname deptaddress
2) seniority ---> bonus
3) rating ---> sal
4) deptid ---> deptid deptname deptaddress

We may use these dicovered FDs to tranform the schema of the relation instance above into the third normal form as follows:

First Step: We get the following minimal cover (Using the algorithm on p. 626)

1) eid ---> name seniority bonus rating sal deptid deptname deptaddress
2) seniority ---> bonus
3) rating ---> sal
4) deptid ---> deptname
5) deptid ---> deptaddress

Second step:

The FD Seniority ---> Bonus introduces a transitive dependency on the primary key;
We deal with it by creating the following 2 tables:

(a) (eid, name, seniority, rating, sal, deptid, deptname, deptaddress)
(b) (seniority, bonus)

Since the relation (a) in the previous step is not yet in 3NF (because

of the FDs (2)--(5) above), we continue decomposing (a) by using the FD Rating --> Sal:

(a1) (eid, name, seniority, rating, deptid, deptname, deptaddress)
(a2) (rating, sal)
(b) (seniority, bonus)

We now use the FD (4)-(5) above and perform the optimization described on p.627 to obtain:

(a11) (eid, name, seniority, rating, deptid)
(a12) (deptid, deptname, deptaddress)
(a2) (rating, sal)
(b) (senirity, bonus)

Third Step: None of the the FDs (1)--(5) is violated.

Fourth Step:

The relations (a11) -- (b) obtained in Step 3 above form our set of relation schemas that will replace the original, unique schema that we got from the relation instance above.