1. **(20 points)** Use the mapping algorithms to convert the EER database schema shown in Figure 8.9 to Relational form. Please use a design or drawing program, or draw neatly and legibly. Attach your solution separately.

   See attached drawing.

2. Consider the relation \( R = \{A, B, C, D, E, F, G, H, I, J\} \) and the set of functional dependencies \( F = \{\{A, B\} \rightarrow \{C\}, \{A\} \rightarrow \{D, E\}, \{B\} \rightarrow \{F\}, \{F\} \rightarrow \{G, H\}, \{D\} \rightarrow \{I, J\}\}. \) In the prequiz you might have found that the key of this relation is \( \{A, B\} \).

   (a) **(5 points)** Decompose \( R \) into 2NF.

   Splitting out attributes based on relations only partially dependent on the key gives:

   \[
   \begin{align*}
   R_1 &= \{A, D, E, I, J\} \text{ preserves the functional dependencies} \\
   &= \{A\} \rightarrow \{D, E\}, \{D\} \rightarrow \{I, J\} \\
   R_2 &= \{B, F, G, H\} \text{ preserves } \{\{B\} \rightarrow \{F\}, \{F\} \rightarrow \{G, H\}\} \\
   R_3 &= \{A, B, C\} \text{ preserves } \{\{A, B\} \rightarrow \{C\}\}
   \end{align*}
   \]

   The primary keys of these subrelations are underlined.

   (b) **(5 points)** Decompose that further into 3NF.

   Further splitting attributes with transitive dependencies on their keys gives:

   \[
   \begin{align*}
   R_{1a} &= \{A, D, E\} \\
   R_{1b} &= \{D, I, J\} \\
   R_{2a} &= \{B, F\} \\
   R_{2b} &= \{F, G, H\} \\
   R_3 &= \{A, B, C\}
   \end{align*}
   \]
Relational schema mapped from the EER schema in Figure 8.9
3. Consider the relation \( R \), which has attributes that hold schedules of courses and sections at a university; \( R = \{ \text{Course no}, \text{Sec no}, \text{Offering dept}, \text{Credit hours}, \text{Course level}, \text{Instructor ssn}, \text{Semester}, \text{Year}, \text{Days hours}, \text{Room no}, \text{No of students} \} \). Suppose that the following functional dependencies hold on \( R \):

\[
\begin{align*}
\{ \text{Course no} \} & \rightarrow \{ \text{Offering dept}, \text{Credit hours}, \text{Course level} \} \\
\{ \text{Course no}, \text{Sec no} \}, \{ \text{Semester}, \text{Year} \} & \rightarrow \{ \text{Days hours}, \text{Room no}, \text{No of students}, \text{Instructor ssn} \} \\
\{ \text{Room no}, \text{Days hours} \}, \{ \text{Semester}, \text{Year} \} & \rightarrow \{ \text{Instructor ssn}, \text{Course no}, \text{Sec no} \}
\end{align*}
\]

(a) (5 points) Which sets of attributes form candidate keys of \( R \)?

The closure of \( K_1 = \{ \text{Course no}, \text{Sec no}, \text{Semester}, \text{Year} \} \) under the functional dependencies is the entire \( R \). This is also the case for \( K_2 = \{ \text{Room no}, \text{Days hours}, \text{Semester}, \text{Year} \} \), and no other set that doesn’t already contain \( K_1 \) or \( K_2 \) as subsets, so these are the two candidate keys.

(b) (10 points) Decompose this relation to 3NF.

Start by decomposing it to 2NF. The only FD that is partial on the left and nonkey on the right is \( \{ \text{Course no} \} \rightarrow \{ \text{Offering dept}, \text{Credit hours}, \text{Course level} \} \), so decompose on that to:

\[
\begin{align*}
R_1 & = \{ \text{Course no}, \text{Offering dept}, \text{Credit hours}, \text{Course level} \} \\
R_2 & = \{ \text{Course no}, \text{Sec no}, \text{Instructor ssn}, \text{Semester}, \text{Year}, \text{Days hours}, \text{Room no}, \text{No of students} \}
\end{align*}
\]

Neither of these relations have a nonkey attribute transitively dependent on a key, so this is also in 3NF.

4. (20 points) Write a program for accessing a LIBRARY database with the schema shown in Figure 4.6, that allows a user to search for books having a particular title, author, or both. Attach the source code.

```python
#!/usr/bin/env python
# -*- coding: utf-8 -*-

import MySQLdb

# There are many ways to do this. Here's one.
host = 'example.com'
user = 'brothgar'
pwd = 'ih8grendL'
db = 'library'

try:
    conn = MySQLdb.connect(host, user, pwd, db)
    cursor = conn.cursor()

    title = raw_input('Title to search for (leave blank if "any"): ')
    author = raw_input('Author to search for (leave blank if "any"): ')
    both = len(title) > 0 and len(author) > 0

    query = 'select Book_id, Title, Publisher_name from Book natural join Book_Authors
    where '}
```
if len(title) > 0:
    query += "Title = '%s'" % title
if len(author) > 0:
    if both:
        query += ' and '
    query += "Author_name = '%s'" % author

cursor.execute(query)
rows = cursor.fetchall()
print 'Found %d books matching that search:' % len(rows)
for row in rows:
    print '[%5d] %s, published by %s' % row

except MySQLdb.Error, e:
    print "database error %d: %s" % (e.args[0], e.args[1])
Figure 8.9
An EER conceptual schema for a UNIVERSITY database.
Figure 4.6
A relational database schema for a LIBRARY database.