1) [25 Points] By looking at the PHLogger table:
   a) List all non-trivial functional dependencies.
      \[ \text{phlid} \rightarrow \text{name}, \text{address\_street}, \text{address\_city}, \text{address\_state}, \text{address\_pcode} \]
   b) What is the highest normal form the PHLogger table is in currently?
      \[ \text{BCNF} \]
   c) The external consulting experts at DBInstructor, Inc., have noticed that city and state of an
      address can be inferred by its postal code (zip code). What new functional dependencies would
      be introduced by codifying this rule?
      \[ \text{address\_pcode} \rightarrow \text{address\_city}, \text{address\_state} \]
   d) What is the highest normal form the PHLogger table is in after adding the new functional
      dependencies?
      \[ \text{2NF. Transitive dependency is now introduced. Example: phlid} \rightarrow \text{address\_pcode and}
      \text{adress\_pcode} \rightarrow \text{address\_state, address\_city} \]
   e) Decompose the PHLogger table into multiple tables to the highest normal form possible.
      \[ \text{PHLogger(phlid, name, address\_pcode, address\_street) key: phlid}
      \text{PHLogger\_pcode(address\_pcode, address\_city, address\_state) key: (address\_pcode)} \]
   f) After decomposition, what is the highest normal form design that you could produce which is
      lossless and dependency preserving[3NF/BCNF]? Explain.
      \[ \text{BCNF. All dependencies are key dependencies.} \]

2) [25 points] Consider the following relation:

<table>
<thead>
<tr>
<th>G</th>
<th>H</th>
<th>M</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>h1</td>
<td>m1</td>
</tr>
<tr>
<td>10</td>
<td>h2</td>
<td>m2</td>
</tr>
<tr>
<td>11</td>
<td>h4</td>
<td>m1</td>
</tr>
<tr>
<td>12</td>
<td>h3</td>
<td>m4</td>
</tr>
<tr>
<td>13</td>
<td>h1</td>
<td>m1</td>
</tr>
<tr>
<td>14</td>
<td>h3</td>
<td>m4</td>
</tr>
</tbody>
</table>

a) Given the current state of the database, for each one of the following functional dependencies
   answer a) Does this functional dependency hold in the above relation instance [Yes/No]? b) If
   your answer to previous question was no, explain why by listing a tuple that causes a violation.
i) \[ G \rightarrow H \]
   - No. 10 → h1, 10 → h2

ii) \[ H \rightarrow M \]
    - Yes

iii) \[ M \rightarrow H \]
    - No. m1 → h1, m1 → h4

iv) \[ H \rightarrow G \]
   - No. h1→ 10, h1 → 13

v) \[ M \rightarrow G \]
   - No. m1 → 10, m1 → 11

b) List all potential candidate keys (if there are any) for the above relation.
   (GH), (GM)

3) [25 points] Considering the relation \( R(A,B,C,D,E) \) and the following functional dependencies, answer the questions.

   FD1: \( AB \rightarrow C \)
   FD2: \( CD \rightarrow E \)
   FD3: \( DE \rightarrow B \)

A. List all the candidate keys.
   ABD, ADE, ACD

B. What is the highest normal form that \( R \) satisfies and why?
   3NF. There is not partial or transitive dependency to non-prime attributes so it is in 3nf. It is not in bcnf as not all left hand sides of functional dependencies are super keys. Example: AB -> C while AB is not a super key.

C. If \( R \) is not already at least in 3NF, then normalize \( R \) into 3NF and show the resulting relation(s) and their candidate keys. Your decomposition should be both join-lossless and dependency-preserving. If \( R \) is already in 3NF, just list the candidate keys of \( R \).
   It is already in 3NF. CK:ABD, ADE, ACD

D. Is your decomposition in BCNF as well?[Yes/No]. Explain.
   No. not all left hand sides of functional dependencies are super keys. Example: AB -> C while AB is not a super key.
   A bcnf decomposition(not necessary to provide it in this question):
   R1:(A,B,C) key:A,B
   R2:(D,E,B) key:D,E
   R3:(A,D,E) key: ADE

4) [25 points] Considering the relation \( R(A,B,C,D,E) \) and the following functional dependencies, answer the questions.

   FD1: \( A \rightarrow BC \)
   FD2: \( BC \rightarrow AD \)
   FD3: \( D \rightarrow E \)

A. List all the candidate keys.
(A) (BC)

B. What is the highest normal form that R satisfies and why?
   2NF. All attributes are atomic and there is no partial dependency hence it is 2NF. It violates 3NF because of transitive dependency: BC → AD, D → E (therefore BC → E)

C. If R is not already at least in 3NF, then normalize R into 3NF and show the resulting relation(s) and their candidate keys. Your decomposition should be both join-lossless and dependency-preserving. If R is already in 3NF, just list the candidate keys of R.
   Answer:
   R1:(A,B,C,D) key: BC, A
   R2:(D,E) key:D

D. Is your decomposition in BCNF as well?[Yes/No]. Explain.
   Yes. All FDs in each relation are now key constraints.