Q1: Given the relational schema: \( R (A, B, C, D, E) \) with the following functional dependencies:

\[ B \rightarrow C, \quad D \rightarrow B, \quad \text{and} \quad E \rightarrow A \]

and the following instance:

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>7</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>9</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>0</td>
</tr>
</tbody>
</table>

Is \( R \) consistent with the dependencies specified above? Is it a valid instant? YES NO

If the answer is NO, circle the dependencies that are violated by \( R \).

\[ B \rightarrow C, \quad D \rightarrow B, \quad E \rightarrow A \]

Q2: What is the key for the universal relation \( R = \{A, B, C, D, E, F, G, H, I, J\} \)?

Given the following set of functional dependencies:

\[ F = \{ AB \rightarrow C, \quad BD \rightarrow EF, \quad AD \rightarrow GH, \quad A \rightarrow I, \quad H \rightarrow J\} \]

A minimal set of attributes whose closure includes all the attributes in \( R \) is a key.

Since the closure of \( \{A, B\} \), \( \{A, B\}^+ = R \), one key of \( R \) is \( \{A, B\} \) (in this case, it is the only key).
Q3: Consider a database system with the following relations: 

Student(StudNo, StudName)

StudMajor(StudNo, Major, Advisor)

StudCourse(StudNo, Major, CourseNo, Ctitle, InstrucName, InstructLocn, Grade)

Functional dependencies:

StudNo → StudName

CourseNo → Ctitle, InstrucName

InstrucName → InstructLocn

StudNo, CourseNo, Major → Grade

StudNo, Major → Advisor

Advisor → Major

a. Does the above DB system satisfy 1NF? Circle: YES NO

If NO, normalize it to 1NF.

b. Does the DB system (generated from part a) satisfy 2NF? YES NO

If NO, normalize it to 2NF.

2NF Remove partial key dependencies

Student(StudNo, StudName)

StudMajor(StudNo, Major, Advisor)

StudCourse(StudNo, Major, CourseNo, Grade)

Course(CourseNo, Ctitle, InstrucName, InstructLocn)
c. Does the DB system (generated from part b) satisfy 3NF? **YES**

If NO, normalize it to 3NF.

3NF: Remove transitive dependencies

- Student(StudNo, StudName)
- StudMajor(StudNo, Major, Advisor)
- StudCourse(StudNo, Major, CourseNo, Grade)
- Course(CourseNo, Ctitle, InstrucName)
- Instructor(InstructName, InstructLocn)

**d.** Does the DB system (generated from part c) satisfy BCNF? **YES**

If NO, normalize it to BCNF.

BCNF: Every determinant is a candidate key

- Student: only determinant is StudNo
- StudCourse: only determinant is StudNo, Major
- Course: only determinant is CourseNo
- Instructor: only determinant is InstrucName
- StudMajor: the determinants are StudNo, Major, or Adviser

Only StudNo, Major is a candidate key

BCNF

- Student(StudNo, StudName)
- StudCourse(StudNo, Major, CourseNo, Grade)
- Course(CourseNo, Ctitle, InstrucName)
- Instructor(InstructName, InstructLocn)
- StudMajor(StudNo, Advisor)
- Adviser(Adviser, Major)
Q4: Consider the relation for an appliance dealer (2 Points)

REFRIG (Model#, Year, Price, Manuf_plant, Color)

Assume that

Model# → Manuf_plant

Model#, Year → Price

Manuf_plant → Color

Consider the decomposition:

REFRIG1 (Model#, Year, Manuf_plant)

REFRIG2 (Model#, Price, Color).

Does this decomposition have the lossless join property? Circle: YES NO

Justify:

R1 ∩ R2 = Model#

R1 – R2 = (Year, Manuf_Plant)

R2 – R1 = (Price, Color)

Because Model# doesn’t functionally define either (R1 – R2) or (R2 – R1), this decomposition is not lossless join.
Q5: Consider the following relation with its functional dependencies:  

It refers to options installed on cars (e.g., cruise control) that were sold at a dealership and the list and discounted price for the option.

\[ R = CAR\_SALE\ (Car\#, \ Option\_type, \ Option\_Listprice, \ Sale\_date, \ Discounted\_price) \]

\[ F = \{\ Car\# \rightarrow Sale\_date, \ Option\_type \rightarrow Option\_Listprice, \ Car\#, \ Option\_type \rightarrow Discounted\_price\}\]

Also, consider a decomposed version of the above relation:

\[ R1 = CAR\_OPTION\_SALE\(Car\#, \ Option\_type, \ Sale\_date)\]

\[ R2 = CAR\(Car\#, \ Sale\_date)\]

\[ R3 = OPTION\(Option\_type, \ Option\_Listprice, \ Discounted\_price)\]

Using the algorithm for lossless join decomposition testing (Algorithm 11.1), determine if this decomposition is indeed lossless. Describe how you came to your conclusion.

(2pts – tableau setup and change & 2pts- explanation & correct conclusion)

Answer: Denote R1 as relation CAR\_OPTION\_SALE, R2 as relation SALE, R3 as relation OPTION, then

<table>
<thead>
<tr>
<th>Car#</th>
<th>Option_type</th>
<th>Option_ListPrice</th>
<th>Sale_date</th>
<th>Discounted_price</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>a1</td>
<td>a2</td>
<td>b13</td>
<td>a4</td>
</tr>
<tr>
<td>R2</td>
<td>a1</td>
<td>b22</td>
<td>b23</td>
<td>a4</td>
</tr>
<tr>
<td>R3</td>
<td>b31</td>
<td>a2</td>
<td>a3</td>
<td>b34</td>
</tr>
</tbody>
</table>

Considers FDs \{ Car\# \rightarrow Sale\_date, Option\_type \rightarrow Option\_Listprice, Car\#, Option\_type \rightarrow Discounted\_price \}

<table>
<thead>
<tr>
<th>Car#</th>
<th>Option_type</th>
<th>Option_ListPrice</th>
<th>Sale_date</th>
<th>Discounted_price</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>a1</td>
<td>a2</td>
<td>a3</td>
<td>a4</td>
</tr>
<tr>
<td>R2</td>
<td>a1</td>
<td>b22</td>
<td>b23</td>
<td>a4</td>
</tr>
<tr>
<td>R3</td>
<td>b31</td>
<td>a2</td>
<td>a3</td>
<td>b34</td>
</tr>
</tbody>
</table>

Since there is no row of entirely “a” symbols, this decomposition is not lossless.

EXPLANATION: In the first row, we can set b13 to a3 because a2 \rightarrow a3 and a2 and a3 are present in row3. In row1 b15 cannot be set to a5 because a1, a2 \rightarrow a5 needs to have another row that has all of a1, a2 and a5 present in the same row. Since no such row is presented, we cannot change b15 to a5 and conclude that no row can be set to all a’s.
Q6: Consider the relation R(A, B, C) with a MVD A \(\rightarrow\rightarrow\) B. R currently has the following tuples:

\[(a, b1, c1), (a, b2, c2), \text{ and } (a, b3, c3).\]

What other tuples must also be in R? 

\textbf{Answer:}

Each of the tuples \((a,b1,c2), (a,b1,c3), (a,b2,c1), (a,b2,c3), (a,b3,c1),\) and \((a,b3,c2)\) are also in R.

Q7: Consider the relation R(empName, childName, autoSerialNo). The relation keeps records of employee name (empName), all his/her children names (childName), and for each auto the employee owns, its serial number. Does the relation R has any nontrivial MVD? If so, what is it/are they?

\[\text{SSN} \rightarrow\rightarrow \text{childSSN}\]

\[\text{SSN} \rightarrow\rightarrow \text{autoSerialNo}\]

Q8: Consider the relation R(A, B, C, D) with MVD’s A \(\rightarrow\rightarrow\) B and A \(\rightarrow\rightarrow\) C. Decompose R to 4NF.

\textbf{The final set of relations are AB, AC, and AD}
Q9: Consider a table \textbf{emp} (for employees) with \( r = 200,000 \) records of \( R = 100 \) bytes each, created with the statement:

\[
\text{create table emp (eid integer primary key, ...)} \textbf{pctfree 25};
\]

where \textbf{pctfree n} clause refers to the amount of free space (\( \%n \)) that must be left on each data page in placing records on the page.

How many data pages needed to store the records of the table \textbf{emp}. Assuming only \textbf{2-byte inter-record gap}, page size is \( B = 2048 \) bytes, and an unspanned organization. Ignore the file header size. Hint: first, compute the file blocking factor \( bfr \).

\[
bfr = \frac{2048 \times 0.75}{102} = 15 = \text{floor}(B \times 0.75/R)
\]

\[
b = \text{Celing}(200000/15) = 13,334 = r/bfr
\]
Q10: Consider a disk with block size $B = 512$ bytes. Suppose that a file has $r = 30,000$ records of fixed length. Each record occupies 128 bytes. Assume records are un-spanned, that one of the fields F1 is 16 bytes and that a pointer is 7 bytes long. Suppose we want to construct a secondary index on field F1. What will be the index blocking factor? (2 Points)

Hint: compute first the size of each index entry $R_i$.

$R_i = V + P = 16 + 7 = 23$

$bf_{RI} = \frac{B}{R_i} = \frac{512}{23} = 22$

Q11: Consider a disk with the following characteristics: (1 Point)

8,192 cylinders,

a block size of 4096 bytes,

an average rotational latency of 5ms,

an average seek time of 7ms,

a block transfer time of 0.5ms.

How much time (in ms) would it take to read 100 blocks that are randomly stored on disk?

1250
Q12: Consider the following $B^+$-tree. Which nodes (disk pages) would be accessed to find all items in the range 5 to 12, inclusive? (1.5 Points)

1, 2, 5, 6, and 7

Q13: Give the $B^+$-tree below, draw the final tree that results after we insert 18. (1.5 Points)