Problem #1

a) There is no functional dependency since it's many to many mapping, which means a can have multiple b values and vice versa, which does not satisfy the definition of functional dependency.

b) $b \rightarrow a$
Each b value is associated at most one a value, but a can have multiple b values

c) $a \rightarrow b$
By the same logic as above.

d) $a \rightarrow b$ & $b \rightarrow a$
Both are one-to one, so each a and b value is associated at most one b and a value, respectively (both unique).

Problem #2

1) Union's rule: If $\alpha \rightarrow \beta$ holds, and $\alpha \rightarrow \gamma$ holds, then $\alpha \rightarrow \beta\gamma$ holds.
   a) Augmentation rule: If $\alpha \rightarrow \gamma$ holds, and α is a set of attributes, then $\alpha\alpha \rightarrow \alpha\gamma$ holds. Thus, $\alpha \rightarrow \alpha\gamma$ holds (since $\alpha\alpha$ is just two identical attribute sets)
   b) Augmentation rule: If $\alpha \rightarrow \beta$ holds, and γ is a set of attributes, then $\alpha\gamma \rightarrow \beta\gamma$ holds
   c) Transitivity rule: If $\alpha \rightarrow \alpha\gamma$ holds, and $\alpha\gamma \rightarrow \beta\gamma$ holds, then $\alpha \rightarrow \beta\gamma$ holds.

   And we're done.

2) Decomposition Rule: If $\alpha \rightarrow \beta\gamma$ holds, then $\alpha \rightarrow \beta$ holds and $\alpha \rightarrow \gamma$ holds.
   a) Reflexivity rule: If $\beta\gamma$ is a set of attributes and $\beta \subseteq \beta\gamma$, then $\beta\gamma \rightarrow \beta$ holds.
   b) Reflexivity rule: If $\beta\gamma$ is a set of attributes and $\gamma \subseteq \beta\gamma$, then $\beta\gamma \rightarrow \gamma$ holds.
   c) Transitivity rule: If $\alpha \rightarrow \beta\gamma$ holds, and $\beta\gamma \rightarrow \beta$ holds, then $\alpha \rightarrow \beta$ holds.
   d) Transitivity rule: If $\alpha \rightarrow \beta\gamma$ holds, and $\beta\gamma \rightarrow \gamma$ holds, then $\alpha \rightarrow \gamma$ holds.

   And we're done.

3) Pseudotransitivity rule: If $\alpha \rightarrow \beta$ holds, and $\gamma\beta \rightarrow \delta$ holds, then $\alpha\gamma \rightarrow \delta$ holds.
   a) Augmentation rule: If $\alpha \rightarrow \beta$ holds, and γ is a set of attributes, then $\gamma\alpha \rightarrow \gamma\beta$ holds.
   b) Transitivity rule: If $\gamma\alpha \rightarrow \gamma\beta$ holds, and $\gamma\beta \rightarrow \delta$ holds, then $\gamma\alpha \rightarrow \delta$ holds.

   And we're done.

Problem #3

Part A

Given:
F= {A→BC, CD→E, B→D, E→A}
R= (A,B,C,D,E),

If If α → R then α is a superkey

a) A:
   i) Given: A→BC. This means that A → B & A → C
   ii) Reflexivity rule: If A is a set of attributes and A ⊆ A, then A → A holds.
   iii) Transitivity: If A → B holds, and B → D holds, then A → D holds.
   iv) Union: If A → C holds, and A → D holds, then A → CD holds.
   v) Transitivity: If A → CD holds, and CD → E holds, then A → E holds.
   vi) Union: From i) to v), which would give the following: A → ABCDE.

   And we're done.

b) E:
   i) Transitivity: If E → A holds, and A → ABCDE holds, then E → ABCDE holds.

   And we're done.

c) CD:
   i) Transitivity: If CD → E holds, and E → ABCDE holds, then CD → ABCDE holds.

   And we're done.

d) BC:
   i) Augmentation: If B → D holds, and C is a set of attributes, then BC → CD holds. Transitivity: If BC →CD holds, and CD → ABCDE holds, then BC → ABCDE holds.

   And we're done.

Part B

1) $A \rightarrow ABCDE$, and all dependencies generated from this by applying the Decomposition Rule.
2) $E \rightarrow ABCDE$, and all dependencies generated from this by applying the Decomposition Rule.
3) $CD \rightarrow ABCDE$, and all dependencies generated from this by applying the Decomposition Rule.
4) $BC \rightarrow ABCDE$, and all dependencies generated from this by applying the Decomposition Rule.
5) $B \rightarrow D$ is given
6) $B \rightarrow BD$ (Union Rule)
7) $B \rightarrow B$ (Reflexivity Rule)
8) $C \rightarrow C$ (Reflexivity Rule)
9) $D \rightarrow D$ (Reflexivity Rule)
10) $BD \rightarrow BD$ (Reflexivity Rule)
11) $BD \rightarrow B$ (Decomposition Rule)
12) $BD \rightarrow D$ (Decomposition Rule)
13) $ABCDE \rightarrow ABCDE$
14) Trivial dependencies $\alpha \rightarrow \beta$, where $\alpha = ABCDE$ and $\beta \subseteq \alpha$ by applying the Decomposition Rule.

Problem #4

The answer is no, and we will prove this using a counter example. Consider the table below:

A	B	C	D
0	1	2	5
0	1	3	4
0	1	2	4
0	1	3	5

The following proves that $A \rightarrow\rightarrow BC$:

1) $t1[A] = t2[A] = t3[A] = t4[A] = 0$
2) $t1[BC] = t3[BC] = 12$ and $t2[BC] = t4[BC] = 12$

3) $t1[AD] = t4[AD] = 05$ and $t2[AD] = t3[AD] = 04$ $(R - CD = AD)$

Here, consider $A \rightarrow\rightarrow C$:

1) $t1[A] = t2[A] = t3[A] = t4[A] = 0$
2) $t1[C] = t3[C] = 2$ and $t2[C] = t4[C] = 3$
3) $t1[ABD] = t4[ABD] = 015$ and $t2[ABD] = t3[ABD] = 014$ $(R - C = ABD)$

Thus, the above condition is satisfied.

However, consider $A \rightarrow\rightarrow B$:

1) $t1[A] = t2[A] = t3[A] = t4[A] = 0$
2) $t1[B] = t3[B] =$ and $t2[B] = t4[B] = 1$
3) $t1[ACD] = 025 \neq t4[ACD] = 035$ and $t2[ACD] = 034 \neq t3[ACD] =024$ $(R - B = ACD)$

Since it doesn't satisfy the 3rd condition, it doesn’t satisfy the following: $A \rightarrow\rightarrow B$

Thus, by proof by condition, we logically proved that $A\rightarrow\rightarrow BC$ doesn’t imply $A\rightarrow\rightarrow B$ and $A\rightarrow\rightarrow C$.

Problem #5

Part a

Given: $F= \{A\rightarrow E, BC\rightarrow D, C\rightarrow A, AB\rightarrow D, D\rightarrow G, BC\rightarrow E, D\rightarrow E, BC\rightarrow A\}$

a) $BC \rightarrow E$ is extraneous
   i) Transitivity rule: If $BC\rightarrow A$ holds, and $A\rightarrow E$,holds, then $BC\rightarrow E$ holds.

      $F= \{A\rightarrow E, BC\rightarrow D, C\rightarrow A, AB\rightarrow D, D\rightarrow G, D\rightarrow E, BC\rightarrow A\}$

b) Union Rule: If $D \rightarrow E$ holds, and $E \rightarrow G$ holds, then $D \rightarrow EG$ holds.

   $F= \{A\rightarrow E, BC\rightarrow D, C\rightarrow A, AB\rightarrow D, D\rightarrow EG, BC\rightarrow A\}$

c) $BC \rightarrow D$ is extraneous
   i) Augmentation rule: If $C \rightarrow A$ holds, and B is a set of attributes, then $BC \rightarrow AB$

ii) Transitivity rule: If BC→ AB holds, and AB → D, holds then BC→ D holds.

F= {A→E, C→A, AB→D, D→EG, BC→A}

d) B is extraneous in BC → A since we have that C → A

F= {A→E, C→A, AB→D, D→EG}

Part b

Can easily test if α is a superkey
●Compute α^+
●If $R \subseteq \alpha^+$ then α is a superkey of R

1) $BC^+ = BC$
   a) BC → D (extraneous)
2) $BC^+ = BCD$
   a) BC → E (extraneous)
3) $BC^+ = BCDE$
   a) BC → A (given)
4) $BC^+ = ABCDE$
   a) Transitivity rule: If BC→ D holds, and D → G, holds then BC→ G holds.
5) $BC^+ = ABCDEG$

Thus, BC is a superkey of R.

●Compute B^+ & C^+

1) $B^+ = B$
2) $C^+ = C$
   a) C → A
3) $C^+ = AC$
   a) Transitivity rule: If C→ A holds, and A → E, holds then C→ E holds.
4) $C^+ = ACE$

Thus, it is a candidate key.

Problem 5c

Based on the previous problem, we know that BC is a candidate (super) key, so we know that BC —> D, BC —> E, BC —> A is in BCNF.

#1: $A \rightarrow E$

Decompose into $R_1(A,E)$ and $R_2(A,B,C,D,G)$

$D \rightarrow G$, so D is a primary key.

Decompose into $R_2(A,B,C,D)$

$C \rightarrow A$, so C is a primary key.

Decompose into $R_4(B,C,D)$

$BC \rightarrow A$, so BC is a primary key.

Thus, we have the following:

$R_1(\underline{A},E)$ and $R_2(\underline{D},G)$ $R_3(\underline{C},A)$ and $R_4(\underline{B},\underline{C},D)$

Here, AB —> D and D —> E are not preserved in the decomposition.

Problem 5d

#1: $D \rightarrow E$

Decompose into $R_1(D,E)$ and $R_2(A,B,C,D,G)$

$AB \rightarrow D$, so AB is a primary key
Decompose into $R_2(A,B,C,G)$

$BC \rightarrow A$, so BC is a primary key.

Decompose into $R_2(B,C,G)$

$BC \rightarrow A$, so BC is a primary key.

Thus, we have the following:

$R_1(\underline{D},E)$ and $R_2(\underline{A},\underline{B},D)$ $R_3(\underline{B},\underline{C},A)$ and $R_4(\underline{B},\underline{C},G)$
Here, A —> E and D —> G are not preserved in the decomposition.

Problem 5e

Here, we know that $F_c = \{A\rightarrow E, C\rightarrow A, AB\rightarrow D, D\rightarrow EG\}$.
We use 3NF synthesis algorithm.

For each F_c:
1) R_1 ($\underline{A}$,E), where A is a primary key
2) R_2 ($\underline{C}$,A), where C is a primary key
3) R_3 ($\underline{A,B}$,D), where AB is a primary key
4) R_4 ($\underline{D}$,E,G), where D is a primary key
5) R_5 ($\underline{B,C}$), where BC is a candidate key

Problem 6a

For implication:
R(course_id, section_id, dept, units, course_level, instructor_id, term, year,

meet_time, room, num_students) = R(A,B,C,D,E,F,G,H,I,J,K), respectively.

Functional dependency:

{A→CDE , ABGH → IJKF, JIGH→FAB}

1) Consider $\{JIGH\}^+ = JIGH$:

   JIGH → FAB

   $\{JIGH\}^+ = ABFGHIJ$

   A is in ABFGHIJ, so A —> CDE

   $\{JIGH\}^+ = ABCDEFGHIJ$

   ABGH in ABCDEFGHIJ, so ABGH → IJKF

   $\{JIGH\}^+ = ABCDEFGHIJK$

   Thus, JIGH is a candidate key.

2) Consider $\{ABGH\}^+ = ABGH$:

   ABGH → IJKF

   $\{ABIJ\}^+ = ABFGHIJK$

   A is in ABFGHIJK, so A —> CDE

   $\{ABIJ\}^+ = ABCDEFGHIJK$

   Thus, ABGH is a candidate key.

3) Consider $\{A\}^+ = A$:

   A→ CDE

   $\{A\}^+ = ACDE$

   Thus, A is NOT a candidate key

As a whole, {course_id, section_id, term, year} and {room, meet_time, term, year} are candidate keys.

Problem 6b

Note that instructor_id is extraneous, so we can create two different versions of canonical covers:

#1:

{course_id} ➔ {dept, units, course_level}

{course_id, section_id, term, year} ➔ {meet_time, room, num_students, instructor_id}

{room, meet_time, term, year} ➔ {course_id, section_id}

#2:

{course_id} ➔ {dept, units, course_level}

{course_id, section_id, term, year} ➔ {meet_time, room, num_students}

{room, meet_time, term, year} ➔ {instructor_id, course_id, section_id}

I believe the first canonical cover would be more appropriate for what is being modeled by the schema and dependence above. To illustrate, in a user perspective, people are more likely to look up specific class information and get information about the instructor, not when the user looks up information with regards to classes taken in a specific room and time of the year.

Problem 6c

Based on 6b, we figured out a better-suited canonical cover:

$F_c = \{A \rightarrow CDE , ABGH \rightarrow IJKF, JIGH \rightarrow AB\}$

Normal Form (3NF):

For each F_c:

1) R_1 (A,C, D, E), where A is a primary key
2) R_2 (A,B,G,H,I,J,K,F), where ABGH and JIGH are candidate keys (JIGH also included in this attribute set, so we have two relations)

Schema Decomposition (BCNF):

#1: A → CDE where A is a primary key

Decompose into R_1(A,C,D,E) and R_2(A,B,F,G,H,I,J,K)

ABGH → IJKF and JIGH→AB where ABGH and JIGH are candidate keys

Thus, we have two relations: R_1(A,C,D,E) and R_2(A,B,F,G,H,I,J,K)

FINAL:

R_1(<u>course_id</u>, dept, units, course_level)

R_2(<u>course_id, section_id, term, year</u>, meet_time, room, num_students, instructor_id)

R_3(<u>room, meet_time, term, year,</u> course_id, section_id)

Where R_1 is courses, R_2 is classes, and R_3 is classrooms

Where foreign key R_2(course_id) and R_3(course_id) references R_1(course_id)
And where foreign key R_3(course_id, section_id, term, year) references R_2(course_id, section_id, term, year)

As shown above, both BCNF and 3NF produce same schemas. Due to relatively small scale of this database, we don't have to worry much about the runtime for 3NF algorithms. Since BCNF eliminates more redundant information than 3NF, I would say that 3NF is a better choice for use in a course management system.

Problem 7

For simplification, we will let emails
(email_id, send_date, from_addr , to_addr, subject, email_body, attachment_name, attachment_body) = R(A,B,C,D,E,F,G,H)

Given: $A \rightarrow B,C\ E,F$ and $A \rightarrow \rightarrow D$ and. $A, G \rightarrow H$
4NF algorithm:
R(A,B,C,D,E,F,G,H)

#1: $A \rightarrow B, C, E, F$ where A is primary key
Decompose into R_1(A,B,C,E,F) and R_2(A,D,G,H)

$A \rightarrow \rightarrow D$, where AB is a primary key
Decompose into and R_2(A,G,H)

Thus, we have the following:

R_1(A,B,C,E,F) and R_2(A,D) and R_3(A,G,H)
which is:

R_1(<u>email_id,</u> send_date, from_addr, subject, email_body)

R_2(email_id, to_addr)

R_3(email_id, attachment_name, attachment_body)

Where Where R_1 is emails, R_2 is recipients, and R_3 is email_contents

Where foreign key R_2(email_id) and R_3(email_id) references R_1(email_id)

R_2 is in 4NF since it has a trivial multivalve dependency between email_id and to_addr (shown in slides).
R_1 and R_3 are both in 4NF since they have super keys - email_id as a super key for both.