Problem 1

(a) There are no non-trivial functional dependencies.

(a) $F = \{B \to A\}$

(a) $F = \{A \to B\}$

(a) $F = \{A \to B, B \to A\}$.

Problem 2

(a) Here we will prove that the Union rule is sound. So, we want to prove that

$$A \to B \text{ and } A \to C \implies A \to BC$$

So, we have that $A \to B$ and $A \to C$. Then we have that by augmentation on the first depedency, $AC \to BC$ and that by augmentation on the second depedency, $AA \to AC \implies A \to AC$. Then, using transitivity with these two augmentations, we have that $A \to BC$, as desired.

(b) Here we will prove that the Decomposition rule is sound. So, we want to prove that

$$A \to BC \text{ and } A \to B \implies A \to C$$

So, we have that $A \to BC$ and $A \to B$. Then, using the reflexivity rule, we have that $BC \to B$ and $BC \to C$. Using transitivity with these two dependencies and the original two dependencies, we get that $A \to B$ and $A \to C$, as desired.

(c) Here we will prove that the Pseudotransitivity rule is sound. So, we want to prove that

$$A \to B \text{ and } CB \to D \implies AC \to D$$

So, we have that $A \to B$ and $CB \to D$. Then, using augmentation on the first dependency, we have that $AC \to BC$. Then, using transitivity with $CB \to D$, we have that $AC \to D$, as desired.

Problem 3

(a) We have that:

$$A^+ = ABCDE$$

$$B^+ = BD$$

$$C^+ = C$$

$$D^+ = D$$

$$E^+ = EABCD$$

Thus, we have that our candidate keys are A, E, CD, BC. A and E are candidate keys for obvious reasons. CD is a candidate key because $CD \to E$ and E is a superkey and neither C or D is a superkey. BC is a candidate key because with B and C you can get E (because $B \to D$) and E is a superkey and neither B or D is a superkey.

To properly show each candidate key is a superkey for R, we can compute each keys attribute-set closure. So:

$$A^+ = ABCDE$$

We can see this because $A \to BC$ causes $A^+ = ABC$, then $B \to D$ causes $A^+ = ABCD$, then $CD \to E$ causes $A^+ = ABCDE$.

$$E^+ = ABCDE$$

We can see this because $E \to A$ and A is a superkey.

$$(CD)^+ = ABCDE$$

We can see this because $CD \to E$ and E is a superkey.

$$(BC)^+ = ABCDE$$

We can see this because $B \to D$ causes $(BC)^+ = BCD$, then $CD \to E$ causes $(BC)^+ = ABCDE$ because E is a superkey.

(b) We want to describe all functional dependencies that will appear in the closure F^+ of F. So, we can summarize this as the following. First, we have all dependencies $\alpha \to \beta$ where α is an element from the set of all superkeys and β is $ABCDE$, and all dependencies generated from this by applying the Decomposition Rule. Then we also have all the dependencies in F and all trivial dependencies $\alpha \to \beta$ where $\alpha \subseteq \{ABCDE\}$ and $\beta \subseteq A$.

Problem 4

Here is a counterexample. Here is a table where $A \twoheadrightarrow BC$ holds.

	A	B	C	$R - (A \cup BC)$
t_1	1	2	7	9
t_2	1	7	5	2
t_3	1	2	7	2
t_4	1	7	5	9

Now assume to the contrary that $A \twoheadrightarrow B$. Then we have that $t_1[R-B] = t_4[R-B]$. But we can see here that $t_1[R-B] = 179$ and that $t_4[R-B] = 159$. So we get a contradiction. So clearly the original statement is false.

Problem 5

(a) Here we wish to compute a canonical cover of

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

So first, we can get rid of $BC \to E$. This dependency is extraneous because $BC \to A$ and $A \to E$. So clearly these two dependencies and transitivity logically imply the extraneous one. Now we have

$$F_c = \{A \to E, BC \to D, C \to A, AB \to D, D \to G, D \to E, BC \to A\}$$

Secondly, we can get rid of $BC \to A$. This is extraneous because $BC \to C$ and $C \to A$. Clearly, these two dependencies with transitivity imply the extraneous one. Now we have

$$F_c = \{A \to E, BC \to D, C \to A, AB \to D, D \to G, D \to E\}$$

Thirdly, we can add $D \to GE$ in replacement of $D \to G$ and $D \to E$. This is because we have $D \to E$ and $D \to G$, so by the Union rule, we can add $D \to GE$ (and by the Decomposition rule, putting this in is logically equivalent to having both $D \to G$ and $D \to E$. Now we have

$$F_c = \{A \to E, BC \to D, C \to A, AB \to D, D \to GE\}$$

Finally, we can get rid of $BC \to D$. This is because we have that $C \to A$. Applying augmentation to this gives us $CB \to AB$. Then, by transitivity, (and using the dependency $AB \to D$) we have that $CB \to D$. So that dependency is extraneous because other dependencies in F_c logically imply it. Thus, finally, we are left with

$$F_c = \{A \to E, C \to A, AB \to D, D \to GE\}$$

and by looking at it, we can see that it is a valid canonical cover.

(b) We have that a candidate key for R is BC. Now we will compute its attribute-set closure, $(BC)^+$. We can see that $C \to A$ causes $(BC)^+ = BCA$, then $AB \to D$ causes $(BC)^+ = BCAD$, then $D \to GE$ causes $(BC)^+ = ABCDEG$. So we have computed the attribute-set closure for our candidate key. Now we must demonstrate that it is a candidate key by computing the attribute-set closures of its proper subsets. So, let us begin. First we will compute B^+. We have that $B^+ = B$; this is clear given F. Next, we will compute C^+. We have that $C \to A$ causes $C^+ = CA$, then $A \to E$ causes $C^+ = CAE$. Neither B^+ nor C^+ is a superkey, so we have that BC is a valid candidate key for R.

(c) We want to decompose R into a BCNF schema. We will do this in the following way. We have that $A \to E$ is not a trivial dependency, and that A isn't a superkey for R. So we can decompose R into $R_1 = (\underline{A}, E)$, $R_2 = (\underline{B}, \underline{C}, A, D, G)$. Then we have that $D \to GE$ is not a trivial dependency, and that D isn't a superkey for R_2. So we can decompose further into $R_1 = (\underline{A}, E)$, $R_2 = (\underline{D}, G, E)$, $R_3 = (\underline{B}, \underline{C}, A, D)$. Then we have that $AB \to D$ isn't a trival dependency, and that AB isn't a superkey for R_3. So we can decompose further into $R_1 = (\underline{A}, E)$, $R_2 = (\underline{D}, G, E)$, $R_3 = (\underline{A}, \underline{B}, D)$, $R_4 = \{\underline{B}, \underline{C}, A)$. Then we have that $C \to A$ isn't a trivial dependency, and that C isn't a superkey for R_4. So finally, we end up with $R_1 = (\underline{A}, E)$, $R_2 = (\underline{D}, G, E)$, $R_3 = (\underline{A}, \underline{B}, D)$, $R_4 = (\underline{C}, A)$, $R_5 = (\underline{B}, \underline{C})$. We can now see that there are no dependencies that break BCNF. So each of our final relation-schemas is in BCNF. More specifically, for R_1, the only non-trivial dependency $\alpha \to \beta$ that holds has α as the superkey of R_1 ($A \to E$). For R_2, the only non-trivial dependencies $\alpha \to \beta$ that holds has α as the superkey of R_2 ($D \to E$, $D \to G$). For R_3, the only non-trivial dependency $\alpha \to \beta$ that holds has α as the superkey of R_3 ($AB \to D$). For R_4, the only non-trivial dependency $\alpha \to \beta$ that holds has α as the superkey of R_4 ($C \to A$). And for R_5, we clearly don't have any dependencies that break BCNF.

As far as which dependencies in F_c are not preserved by this decomposition, we have that it preserves all of them (because each dependency appears in one of our decomposed schemas). So basically, if $F' = F_1 \cup F_2 \cup \cdots \cup F_5$, we have that $F'^+ = F^+$.

(d) We want to decompose R into a BCNF schema. We will do this in the following way. We have that $A \to E$ is not a trivial dependency, and that A isn't a superkey for R. So we can decompose R into $R_1 = (\underline{A}, E)$, $R_2 = (\underline{B}, \underline{C}, A, D, G)$. Then we have that $C \to A$ is not a trivial dependency, and that C is not a superkey for R_2. So we can decompose further into $R_1 = (\underline{A}, E)$, $R_2 = (\underline{C}, A)$, $R_3 = (\underline{B}, \underline{C}, D, G)$. Then we have that $D \to G$ is not a trivial dependency, and that D is not a superkey for R_3. So we can decompose further into $R_1 = (\underline{A}, E)$, $R_2 = (\underline{C}, A)$, $R_3 = (\underline{D}, G)$, $R_4 = (\underline{B}, \underline{C}, D)$. We can now see that this is our final answer. This is because we can now see that there are no dependencies that break BCNF. So each of our final relation-schemas is in BCNF. More specifically, for R_1, the only non-trivial dependency $\alpha \to \beta$ that holds has α as the superkey of R_1 ($A \to E$). For R_2, the only non-trivial dependencies $\alpha \to \beta$ that holds has α as the superkey of R_2 ($C \to A$). For R_3, the only non-trivial dependency $\alpha \to \beta$ that holds has α as the superkey of R_3 ($D \to G$). For R_4, the only non-trivial dependency $\alpha \to \beta$ that holds has α as the superkey of R_4 ($BC \to D$).

Now we will specify which dependencies in F_c are not preserved by this decomposition. We have that this decomposition does not preserve $AB \to D$ and $D \to GE$ because these dependencies are not included F'^+, where $F' = F_1 \cup \cdots \cup F_4$.

(e) We have that our $F_c = \{A \to E, C \to A, AB \to D, D \to GE\}$. Now, using the 3NF Synthesis Algorithm, we can create a 3NF Schema for R. Following slide 19 in lecture 20, we get $R_1 = (\underline{A}, E)$, $R_2 = (\underline{C}, A)$, $R_3 = (\underline{A}, \underline{B}, D)$, $R_4 = (\underline{D}, G, E)$, $R_5 = (\underline{B}, \underline{C})$. We get the first four relation schemas from F_c, because none of the dependencies in F_c contain each other completely. Then we get the last one by adding in the candidate key for R because no schema in R_1 through R_4 contains a candidate key for R.

Problem 6

(a) Our first candidate key is $course\_id, section\_id, term, year$. We can show that this is a superkey by finding the attribute-set closure. We have that the attribute-set closure is $\{course\_id, section\_id, term, year, meet\_time, room, num\_students, instructor\_id\}$ from the second dependency. Then we have that the attribute-set closure is $\{course\_id, section\_id, term, year, meet\_time, room, num\_students, instructor\_id, course\_id\}$ from the third dependency. Then we have that the attribute-set closure is everything ($\{course\_id, section\_id, term, year, meet\_time, room, num\_students, instructor\_id, course\_id, dept, units, course\_level\}$) from the first dependency.

Our second candidate key is $room, meet\_time, term, year$. We can show that this is a superkey by finding the attribute-set closure. We have that the attribute-set closure is $\{room, meet\_time, term, year, instructor\_id, course\_id, section\_id\}$ from the third dependency. Then we have that the attribute-set closure is everything ($\{course\_id, section\_id, term, year, meet\_time, room, num\_students, instructor\_id, course\_id, dept, units, course\_level\}$) from the first and second dependencies.

(b) We have that one canonical cover F_{c1} is as follows:

$$\{course\_id\} \rightarrow \{dept, units, course\_level\}$$

$$\{course\_id, section\_id, term, year\} \rightarrow \{meet\_time, room, num\_students, instructor\_id\}$$

$$\{room, meet\_time, term, year\} \rightarrow \{course\_id, section\_id\}$$

We have that the *instructor_id* in the third dependency is extraneous because it is in the attribute-set closure of $A = \{room, meet\_time, term, year\}$. This is true because with the third dependency in F_{c1} we have that the attribute-set closure of A is $\{room, meet\_time, term, year, course\_id, section\_id\}$. Then, with the second dependency in F_{c1}, we have that the attribute-set closure is $\{room, meet\_time, term, year, course\_id, section\_id, num\_students, instructor\_id\}$ (*instructor_id* is in there!).

We have that one canonical cover F_{c2} is as follows:

$$\{course\_id\} \rightarrow \{dept, units, course\_level\}$$

$$\{course\_id, section\_id, term, year\} \rightarrow \{meet\_time, room, num\_students\}$$

$$\{room, meet\_time, term, year\} \rightarrow \{instructor\_id, course\_id, section\_id\}$$

We have that the *instructor_id* in the second dependency is extraneous because it is in the attribute-set closure of $A = \{course\_id, section\_id, term, year\}$. This is true because with the second dependency in F_{c2} we have that the attribute-set closure of A is $\{course\_id, section\_id, term, year, meet\_time, room, num\_students\}$. Then, with the third dependency in F_{c2}, we have that the attribute-set closure is $\{course\_id, section\_id, term, year, meet\_time, room, num\_students, instructor\_id\}$ (*instructor_id* is in there!).

We have that F_{c1} is most appropriate because it makes more sense to explicitly and visibly associate *instructor_id* with *course_id* and *section_id* than with *room* and *meet_time*. This is because an instructor is more strongly correlated with a class and a section than with a room and a time.

(c) We will use 3NF here. The schema decomposition that results in using this is

$$R_1 = (\underline{course\_id}, dept, units, course\_level)$$

$$R_2 = (\underline{course\_id, section\_id, term, year}, meet\_time, room, num\_students)$$

$$R_3 = (\underline{room, meet\_time, term, year}, course\_id, section\_id)$$

The primary keys are underlined, as are the candidate keys. We have a foreign key $R_2[course\_id]$ references $R_1[course\_id]$. We also have a foreign key $R_3[course\_id]$ references $R_1[course\_id]$. We have a foreign key $R_3[course\_id, section\_id, term, year]$ references $R_2[course\_id, section\_id, term, year]$.

3NF makes the most since here because in this case, we would rather preserve the dependencies than get rid of redudancies (like BCNF does). 3NF also makes checking constraints faster because it eliminates the needs for joins. In this case, since the scale isn't dramatically large (it is just a database for courses after all) we are not too hampered by redundancies.

Problem 7

We have that $email\_id \twoheadrightarrow to\_addr$ is not a trivial multivalued dependency and *email_id* is not a superkey for R. So, we can first decompose this as follows, using the second given dependency:

$$email\_recipients = (\underline{email\_id, to\_addr})$$

$$email\_info = (\underline{email\_id, attachment\_name}, send\_date, from\_addr, subject, email\_body, attachment\_body)$$

Then we have that $email\_id, attachment\_name \rightarrow attachment\_body$. And since if $\alpha \rightarrow \beta$ then $\alpha \twoheadrightarrow \beta$, then we can decompose this as follows, using the third given dependency:

$$email\_recipients = (\underline{email\_id, to\_addr})$$

$$email\_attachments = (\underline{email\_id, attachment\_name}, attachment\_body)$$

$$email\_info = (email\_id, send\_date, from\_addr, subject, email\_body, attachment\_name)$$

Then we have that $email\_id$ isn't a superkey for $email\_info$ because $attachment\_name$ is still in there, so we get non-trivial multivalued dependencies of the form $\alpha \twoheadrightarrow \beta$ where α is not a superkey of $email\_id$. So then we can decompose this as follows, using the first given dependency:

$$email\_recipients = (\underline{email\_id, to\_addr})$$

$$email\_attachments = (\underline{email\_id, attachment\_name}, attachment\_body)$$

$$email\_info = (\underline{email\_id}, send\_date, from\_addr, subject, email\_body)$$

$$email\_attachment\_names = (\underline{email\_id, attachment\_name})$$

We have that for $email\_recipients$, $email\_id \twoheadrightarrow to\_addr$ is a trivial multivalued dependency because $email\_id \cup to\_addr = email\_recipients$. And all other dependencies that apply are trivial .So that schema is in 4NF. Then, for $email\_attachments$, every multivalued dependency $\alpha \twoheadrightarrow \beta$ that applies to it has α as a superkey for $email\_attachments$ $(email\_id, attachment\_name \rightarrow attachment\_body)$.Then, for $email\_info$, every multivalued dependency $\alpha \twoheadrightarrow \beta$ that applies to it has α as a superkey for $email\_info$, since the only dependencies that apply have $\alpha = email\_id$ and $email\_id$ is a superkey. Then, we have that for $email\_attachment\_names$, there are only trival dependencies. So all our schemas are in 4NF. Note that the last table is actually redudant so we will not consider it in the following section.

The primary keys are underlined. We have a foreign key $email\_recipients[email\_id]$ references $email\_info[email\_id]$. We have a foreign key $email\_attachments[email\_id]$ references $email\_info[email\_id]$.