CS121 Assignment 7 Derek Ing, PS 1

1a) $\boxed{A} - \langle R \rangle - \boxed{B}$: No non-trivial functional dependencies because it is a many-to-many mapping

1b) $\boxed{A} - \langle R \rangle \rightarrow \boxed{B}$: $a \rightarrow b$

1c) $\boxed{A} \leftarrow \langle R \rangle - \boxed{B}$: $b \rightarrow a$

1d) $\boxed{A} \leftarrow \langle R \rangle \rightarrow \boxed{B}$: $b \rightarrow a$, $a \rightarrow b$

2.

Union: if $a \rightarrow b$ holds on R and $a \rightarrow g$ holds on $R \Rightarrow a \rightarrow bg$ holds on R.

Given $a \rightarrow b$ holds and $a \rightarrow g$ holds

Augmentation Rule: $a \rightarrow g$ holds $\Rightarrow aa \rightarrow ag$ holds

Augmentation Rule: $a \rightarrow b$ holds $\Rightarrow ab \rightarrow bg$ holds

Transitivity Rule: If $aa \rightarrow ag$ and $ag \rightarrow bg$ holds $\Rightarrow aa \rightarrow bg$ holds

Since $aa \rightarrow bg$ holds $\Rightarrow a \rightarrow bg$ holds since a and aa are equivalent

$\square$

Decomposition: if $a \rightarrow bg$ holds $\Rightarrow a \rightarrow b$ and $a \rightarrow g$ hold

Given $a \rightarrow bg$ holds

Reflexivity rule: $bg \rightarrow b$ holds since $b \subseteq bg$

Reflexivity rule: $bg \rightarrow g$ holds since $g \subseteq bg$

Transitivity rule: $a \rightarrow bg$ holds and $bg \rightarrow b$ holds $\Rightarrow a \rightarrow b$ holds

Transitivity rule: $a \rightarrow bg$ holds and $bg \rightarrow g$ holds $\Rightarrow a \rightarrow g$ holds

$\Rightarrow a \rightarrow b$, $a \rightarrow g$ hold

$\square$

Pseudotransitivity: If $\alpha \rightarrow \beta$ holds, and $\gamma\beta \rightarrow \delta$ holds $\Rightarrow \alpha\gamma \rightarrow \delta$ holds.

Given $\alpha \rightarrow \beta$ holds, given $\gamma\beta \rightarrow \delta$ holds

Augmentation rule: $\alpha \rightarrow \beta$ holds $\Rightarrow \alpha\gamma \rightarrow \beta\gamma$ holds

Transitivity rule: $\alpha\gamma \rightarrow \beta\gamma$ holds and $\gamma\beta \rightarrow \delta$ holds $\Rightarrow \alpha\gamma \rightarrow \delta$ holds because $\beta\gamma$ and $\gamma\beta$ are equivalent

$\square$

3. Given $R=(A,B,C,D,E)$, $F=\{A\to BC, CD\to E, B\to D, E\to A\}$

3a) Specify all candidate keys for R:

→ Candidate key is a minimal superkey ⇒ show each candidate key is a superkey

A:

1. $\alpha^+ = A$
2. $A\to BC \Rightarrow A\to B, A\to C$ (Decomposition)
3. $A\to B, B\to D \Rightarrow A\to D$ (Transitivity)
4. $A\to C, A\to D \Rightarrow A\to CD$ (Union)
5. $A\to CD, CD\to E \Rightarrow A\to E$ (Transitivity)
6. $A\subseteq A \Rightarrow A\to A$ (Reflexivity)
7. $A\to A, A\to BC, A\to D, A\to E \Rightarrow A\to ABCDE$ (Union)

$\Rightarrow \alpha^+ = ABCDE \Rightarrow$ A is a superkey of R ⇒ A is a candidate key

E: 1. $E\to A, A\to ABCDE \Rightarrow E\to ABCDE$ (Union)

⇒ E is a superkey ⇒ E is a candidate key

CD: 1. $CD\to E, E\to ABCDE \Rightarrow CD\to ABCDE$

⇒ CD is a superkey ⇒ CD is a candidate key

BC:

1. $B\to D \Rightarrow BC\to CD$ (Augmentation)
2. $BC\to CD, CD\to ABCDE \Rightarrow BC\to ABCDE$ (Transitivity)

⇒ BC is a superkey ⇒ BC is a candidate key

$\boxed{\text{Candidate keys: } A, BC, CD, E}$

3b) Describe all functional dependencies that appear in closure F^+ of F

1. $A \to ABCDE$ and all dependencies generated by applying the Decomposition Rule
2. $E \to ABCDE$ and all dependencies generated by applying the Decomposition Rule
3. $BC \to ABCDE$ and all dependencies generated by applying the Decomposition Rule
4. $CD \to ABCDE$ and all dependencies generated by applying the Decomposition Rule
5. All trivial dependencies $\alpha \to \beta$ where $\alpha = \beta \Rightarrow \beta \subseteq \alpha$ (Reflexivity)
6. $B \to D$ (given)
7. $B \to BD$ (union)
8. $BD \to BD$ (Augmentation)
9. $BD \to B$ (Decomposition)
10. $BD \to D$ (Decomposition)
11. All combinations $\alpha\beta \to \gamma$:
    - $\alpha \in \{A, BC, CD, E\}$
    - $\beta \in \{A, B, C, D, E\}$ (attributes from R)
    - $\gamma \subseteq R$

4. Given $R(A,B,C,D)$, does $A \to\to BC \Rightarrow A \to\to B$, $A \to\to C$?

<u>No</u>

Consider:

	A	B	C	D
t_1	x	1	2	a
t_2	x	3	2	b
t_3	x	1	2	b
t_4	x	3	2	a

$A \to\to BC$: $t_1[A] = t_2[A] = t_3[A] = t_4[A] = x$
$t_1[BC] = t_3[BC] = (1,2)$, $t_2[BC] = t_4[BC] = (3,2)$ ✓
$t_1[AD] = t_4[AD] = (x,a)$, $t_2[AD] = t_3[AD] = (x,b)$

$A \to\to B$: $t_1[A] = t_2[A] = t_3[A] = t_4[A] = x$ ✓
$t_1[B] = t_3[B] = 1$, $t_2[B] = t_4[B] = 3$
$t_1[ACD] = t_4[ACD] = (x,2,a)$, $t_2[ACD] = t_3[ACD] = (x,2,b)$

$A \to\to C$? $t_1[A] = t_2[A] = t_3[A] = t_4[A] = x$
$t_1[C] = t_3[C] = 2$, $t_2[C] = t_4[C] = 2$
$t_1[ABD] = (x,1,a) \neq t_4[ABD] = (x,3,a) \Rightarrow A \not\to\to C$

$\Rightarrow A \to\to BC \not\Rightarrow A \to\to B, A \to\to C$

5. R(A, B, C, D, E, G), $F = \{A \to E, BC \to D, C \to A, AB \to D, D \to G, BC \to E, D \to E, BC \to A\}$

Sol) Compute canonical cover F_c over F

1. $F_c = F = \{A \to E, BC \to D, C \to A, AB \to D, D \to G, BC \to E, D \to E, BC \to A\}$
2. Union Rule: $\{A \to E, C \to A, BC \to ADE, D \to EG, AB \to D\}$
3. D is extraneous for $BC \to ADE$
   ↳ Consider $\{A \to E, C \to A, AB \to D, D \to EG, BC \to AE\}$
   - $C \to A \Rightarrow BC \to AB$ (Augmentation)
   - $BC \to AB, AB \to D \Rightarrow BC \to D$ (Transitivity)
   $\Rightarrow$ D is extraneous in $BC \to ADE$
4. $F_c = \{A \to E, C \to A, AB \to D, D \to EG, BC \to AE\}$
5. B is extraneous in $BC \to AE$
   ↳ Consider $\{A \to E, C \to A, AB \to D, D \to EG, BC \to AE\}$
   1. $C \to A$ (given)
   2. $C \to A, A \to E \Rightarrow C \to E$ (Transitivity)
   3. $C \to A, C \to E \Rightarrow C \to AE$ (Union)

   $\Rightarrow$ B is extraneous in $BC \to AE$
5. $F_c = \{A \to E, C \to A, AB \to D, D \to EG, C \to AE\}$
6. $C \to AE \Rightarrow C \to A, C \to E$ (Decomposition) $\Rightarrow C \to A$ is extraneous
7. $F_c = \{A \to E, AB \to D, D \to EG, C \to AE\}$
8. E extraneous in $C \to AE$
   $\to$ Consider $\{A \to E, AB \to D, D \to EG, C \to A\}$
   1. $C \to A, A \to E \Rightarrow C \to E$ (Transitivity)

   $\Rightarrow$ E extraneous in $C \to AE$
9. $\boxed{F_c = \{A \to E, AB \to D, D \to EG, C \to A\}}$ No more extraneous dependencies

5b) First check if BC is a superkey:

$\{BC\}^+$:

1. $\{BC\}^+ = BC$
2. $BC \to D \Rightarrow \{BC\}^+ = BCD$ (Union)
3. $BC \to D, D \to G \Rightarrow \{BC\}^+ = BCDG$ (Transitivity)
4. $BC \to A \Rightarrow \{BC\}^+ = ABCDG$ (Union)
5. $BC \to E \Rightarrow \{BC\}^+ = ABCDEG = R$ (Union)

$\Rightarrow$ BC is a superkey

BC is a candidate key:

Compute $\{B\}^+$ and $\{C\}^+$:

$\{B\}^+$: 1. $\{B\}^+ = B$ ← Done

$\{C\}^+$: 1. $\{C\}^+ = C$

2. $C \to A \Rightarrow \{C\}^+ = AC$
3. $A \to E \Rightarrow \{C\}^+ = ACE$ (Transitivity)

Done

$\{B\}^+ \neq R$ and $\{C\}^+ \neq R \Rightarrow$ BC is a minimal superkey

$\Rightarrow$ $\boxed{\text{BC is a candidate key for } R}$

5C) Decompose R into BCNF

1. $A \to E \Rightarrow R_1 = (\underline{A}, E)$
   - BCNF because LHS A is primary key
2. $R_2 = R - E = (A, B, C, D, G)$
   - $C \to A$, choose $R_2 = (\underline{C}, A)$
     - BCNF because LHS C is primary key
3. $R - A = \{B, C, D, G\}$
   - $D \to G$, choose $R_3 = (\underline{D}, G)$
     - BCNF because LHS D is primary key
4. $\{B, C, D, G\} - G = \{B, C, D\}$
   - Given $BC \to D$, choose $R_4 = (\underline{B}, \underline{C}, D)$
     - BCNF because BC is a candidate key and primary key and is on LHS of $BC \to D$

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

$D \to E$, $AB \to D$ not preserved

5d) 3NF schema for R

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

$i=1$: $A \to E \Rightarrow R_1 = A \cup E \Rightarrow R_1 = (\underline{A}, E)$

$i=2$: $C \to A \Rightarrow R_2 = (\underline{C}, A)$ (R_1 does not contain $\{A, C\}$)

$i=3$: $AB \to D \Rightarrow R_3 = (\underline{A}, \underline{B}, D)$ (R_1, R_2 do not contain $\{A, B, D\}$)

$i=4$: $D \to EG \Rightarrow R_4 = (\underline{D}, E, G)$ (R_1, R_2, R_3 do not contain $\{D, E, G\}$)

None of R_1, R_2, R_3, R_4 contain candidate key BC

$i=5$: $R_5 = (\underline{B}, \underline{C})$

$$\boxed{R_1(\underline{A}, E), R_2(\underline{C}, A), R_3(\underline{A}, \underline{B}, D), R_4(\underline{D}, E, G), R_5(\underline{B}, \underline{C})}$$

6. R(course_id, section_id, dept, units, course_level, instructor_id, term, year, meet_time, room, num_students)

A = course_id, B = section_id, C = dept, D = units, E = course_level, G = instructor_id, H = term, I = year, J = meet_time, K = room, L = num_students

$F = \{A \to CDE, ABHI \to GJKL, HIJK \to ABG\}$

6a)

$\{A\}^+$: 1. $\{A\}^+ = A$

2. $A \to CDE \Rightarrow \{A\}^+ = ACDE$

3. No more updates, $ABHI \not\subseteq \{A\}^+$, $HIJK \not\subseteq \{A\}^+$

$\{ABHI\}^+$: 1. $\alpha^+ = ABHI$

2. $ABHI \subseteq \alpha^+ \to GJKL \Rightarrow \alpha^+ = ABGHIJKL$

3. $A \subseteq \alpha^+ \Rightarrow \alpha^+ = ABCDEGHIJKL = R$

$\Rightarrow \{ABHI\} = $ (course_id, section_id, term, year) is a candidate key

$\{HIJK\}^+$: 1. $\alpha^+ = HIJK$

2. $HIJK \subseteq \alpha^+ \to ABG \Rightarrow \alpha^+ = ABGHIJK$

3. $A \subseteq \alpha^+ \to CDE \Rightarrow \alpha^+ = ABCDEGHIJK$

4. $ABHI \subseteq \alpha^+ \to GJKL \Rightarrow \alpha^+ = ABCDEGHIJKL = R$

$\Rightarrow \{HIJK\}$ is a candidate key

$$\boxed{\text{Candidate keys: } \{\text{course\_id, section\_id, term, year}\}, \{\text{room, meet\_time, term, year}\}}$$

6b) {A -> CDE, ABHI -> GJKL, HIJK -> ABG}

G is extraneous for both ABHI -> GJKL and HIJK -> ABG:

F_c^1: {course_id} -> {dept, units, course_level}
{course_id, section_id, term, year} -> {meet_time, room, num_students}
{room, meet_time, term, year} -> {course_id, section_id, instructor_id}

F_c^2: {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}

F_c^2 is more appropriate because we can get information about the instructor from information about the specific course. This makes more sense than getting instructor from information about the room and time of a class. Thus, it is more appropriate to remove instructor_id from the last dependency.

6c) 3NF using F_c^2:
i=1: R_1 = (course_id, dept, units, course_level)
i=2: R_2 = (course_id, section_id, term, year, meet_time, room, instructor_id, num_students)

(room, meet_time, term, year) in R_2

3NF: course (course_id, dept, units, course_level)
section_info (course_id, section_id, term, year, meet_time, room, instructor_id, num_students)
- Foreign key course_id references course.course_id
- Candidate key (term, year, room, meet_time)

6c)

BCNF decomposition:

1. course_id → dept, units, course_level ⇒ (course_id, dept, units, course_level)

2. R − {dept, units, course_level} = (course_id, section_id, instructor_id, term, year, meet_time, room, num_students)

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

BCNF decomposition:

course(course_id, dept, units, course_level)

section_info(course_id, section_id, term, year, meet_time, room, instructor_id, num_students)

- foreign key course_id references course.course_id
- candidate key (term, year, room, meet_time)

3NF and BCNF decomposition give the same schemas. Both of these would be the best for use in a course management system because we eliminate any redundancies while conserving all functional dependencies. We would not need to keep many records in the database relative to other schemas. It would also be relatively easy to enforcing foreign key constraints and candidate key constraints. Each course should have unique sections, term, and year and no two sections meet at the same time in the same room in the same term & year. These schemas make sense and are practical to implement.

7. emails (email_id, send_date, from_addr, to_addr, subject, email_body, attachment_name, attachment_body)

email_id → send_date, from_addr, subject, email body
email_id →→ to_addr
email_id, attachment_name → attachment_body

4NF decomposition on emails:

email_info = (<u>email_id</u>, send_date, from_addr, subject, email_body)
↳ Primary key
– From (email_id → send_date, from_addr, subject, email body), email_id is a superkey
– email_id is primary key ⇒ email_info is in BCNF ⇒ 4NF

$R_2 = R - \beta = R -$ (send_date, from_addr, subject, email_body) = (email_id, to_addr, attachment_name, attachment_body)
Given email_id →→ to_addr ⇒ email_receiver (<u>email_id</u>, <u>to_addr</u>)
↳ trivial multivalued dependency email_id →→ to_addr ⇒ 4NF
↳ Primary key: (email_id, to_addr)
↳ Foreign key: email_id references email_info.email_id

R_2 – to_addr = (email_id, attachment_name, attachment_body)
R_3 = (<u>email_id</u>, <u>attachment_name</u>, attachment body)
↳ from email_id, attachment_name → attachment_body
↳ email_id, attachment_name is primary key ⇒ in BCNF ⇒ 4NF

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

email_receiver (<u>email_id</u>, <u>to_addr</u>)
– Foreign key email_id references email_info.email_id

email_attachment (<u>email_id</u>, <u>attachment_name</u>, attachment body)
– Foreign key email_id references email_info.email_id