1.1

(a) $W \to Y, X \to Z \vdash W X \to Y$

Verdadero, usando Aumento (Para cualquier W, si $X \to Y$ entonces $WX \to W Y$)
si tenemos $W \to Y$, por aumento también $WX \to XY$,
por descomposición $WX \to Y$

(b) $X \to Y$ y $Z \subseteq Y \vdash X \to Z$

Verdadero, usando reflexividad (Si $Y \subseteq X$ entonces $X \to Y$) y transitividad (Si $X \to Y$ e $Y \to Z$ entonces $X \to Z$)

Como $Z \subseteq Y$ entonces $Y \to Z$, y como también tenemos que $X \to Y$, por transitividad $X \to Y$

(c) $X \to Y, X \to W, WY \to Z \vdash X \to Z$

Verdadero
Usando (unión: Si $X \to Y$ y $X \to Z$ entonces $X \to YZ$)
Como $X \to Y$ y $X \to W$, tenemos que $X \to WY$.
Como $WY \to Z$ y $X \to WY$, por transitividad, $X \to Z$

(d) $XY \to Z, Y \to W \vdash XW \to Z$

$Y \to W$, por aumento, $XY \to XW$, y también $XY \to Z$, pero no parece valer la proposición

Contraejemplo:
X = País, Y = provincia, Z = capital de la provincia, W = longitud nombre provincia

Un país y una provincia determinan la capital, y la provincia determina una longitud de nombre, pero si tenemos país y longitud de nombre, no se determina la capital ya que puede haber varias con misma longitud

(e) $X \to Z, Y \to Z \vdash X \to Y$

No parece valer, contraejemplo:

X = LU, Z = edad, Y = fecha de nacimiento

La edad depende funcionalmente de la LU y la fecha de nacimiento, pero la LU no determina una fecha de nacimiento.

(f) $X \rightarrow Y, XY \rightarrow Z \vdash X \rightarrow Z$

Pseudotransitividad (Para cualquier W, Si $X \rightarrow Y$ e $Y W \rightarrow Z$ entonces $XW \rightarrow Z$)

Tenemos $X \rightarrow Y$, $YX \rightarrow Z$ entonces $XX \rightarrow Z$ con W=X, y queda $X \rightarrow Z$

1.2. Dados los siguientes conjuntos de dependencias funcionales, decidir cuales son equivalentes:

(a)
{BC → D, ACD → B, CG → B, CG → D, AB → C, C → B, D → E, BE → C, D → G, CE → A }
(b)
{AB → C, C → A, BC → D, CD → B, D → E, D → G, BE → C, CG → D}
(c)
{AB → C, C → A, BC → D, D → G, BE → C, CG → D, CE → G}
(d)
{C → A, BC → D, D → E, D → G, BE → C, CG → B, CE → G}

Def: Decimos que dos conjuntos de dependencias funcionales F y G sobre R son equivalentes (F ≡ G) si F+ = G+
También si F |= G y G |= F (si F cubre a G y G cubre a F).

a)
C → B
D → E
D → G
BC → D
CG → B
CG → D
AB → C
BE → C
CE → A
ACD → B

(b)
C → A
D → E
D → G
AB → C
BC → D
CD → B
BE → C
CG → D

c)
$C \rightarrow A$
$D \rightarrow G$
$AB \rightarrow C$
$BC \rightarrow D$
$BE \rightarrow C$
$CG \rightarrow D$
$CE \rightarrow G$

d)
$C \rightarrow A$
$D \rightarrow E$
$D \rightarrow G$
$BC \rightarrow D$
$BE \rightarrow C$
$CG \rightarrow B$
$CE \rightarrow G$

viendo lo que queda no me dan equivalentes, pero no sé bien cómo hacerlo. Muy largo calcular clausura. Hay que ir probando con Armstrong?
[según grupo de telegram ninguno es equivalente]

1.3. Sea la relación R(A,B,C,D) y los siguientes conjuntos de dependencias funcionales:

F D1 : $\{B \rightarrow C, D \rightarrow A\}$
F D2 : $\{AB \rightarrow C, C \rightarrow A, C \rightarrow D\}$
F D3 : $\{A \rightarrow B, B \rightarrow C, C \rightarrow D\}$

Decidir cuáles de las siguientes descomposiciones son lossless-join y preservan dependencias funcionales:

Sea $R = (A_1, A_2, \ldots, A_n)$ y F conjunto de dependencias funcionales y ρ una descomposición de R

$$\rho = R_1, R_2, \ldots R_k$$

tal que $\bigcup_{i=1}^{k} R_i = R$

Decimos que ρ es una descomposición sin pérdida de información (lossless join o SPI) si para cada instancia r de R que satisface F, se verifica:

$$r = \bowtie_{i=1}^{k} \pi_{R_i}(r)$$

Entonces es importante determinar, dados R, F y ρ, si ρ es sin perdida de información ((lossless join, SPI)).

Descomposición binaria:

Decimos que una descomposición de R, $\rho = (R_1, R_2)$ es *lossless join* con respecto a un conjunto de dependencias funcionales F, si y sólo si:

La dependencia funcional $(R_1 \cap R_2) \to (R_1 - R_2)$ está en F^+

o

La dependencia funcional $(R_1 \cap R_2) \to (R_2 - R_1)$ está en F^+

O sea que si la intersección de los atributos de R1 y R2 forman una superclave para uno de los dos esquemas, entonces ρ es SPI.

(a) F D1 : (A, D) y (B, C)

La intersección de los atributos (A,D) y (B,C) es vacía, hay pérdida de información.

(b) F D2 : (A, B, D) y (A, B, C)

La intersección de los atributos es {A, B}
Tengo que ver si forman una superclave para uno de los dos esquemas, tenemos que {AB → C, C → A, C → D}, para el esquema (A,B,C), AB es superclave ya que AB → ABC.
Es lossless join

(c) F D2 : (B, C, D) y (A, C)
La intersección de los atributos es {C}.
Tenemos que C → A, por lo que C → AC superclave para el esquema (A,C)
Es lossless join

(d) F D3 : (A, B, D) y (B, C)
La intersección de los atributos es {B} y tenemos que B → C,...
SPI

(e) F D1 : (A, B, D) y (C, D)
La intersección de los atributos es {D}, no es superclave para ninguno. No es lossless join

(f) F D3 : (A, B) y (A, C, D)
La intersección es {A}, y tenemos A → B, A superclave de (A, B). Es lossless join

FALTA TESTEAR PÉRDIDA DE DEPENDENCIA FUNCIONAL

Pérdida de dependencias funcionales

Además de preservar la información (SPI) una descomposición ρdebería preservar las dependencias funcionales, lo que llamamos descomposición sin pérdidas de dependencias funcionales (SPDF).

Proyección de un conjunto de DF

Decimos que la proyección de un conjunto de dependencias funcionales F sobre un conjunto de atributos Z, que se escribe $\pi_Z(F)$, es el conjunto de dependencias funcionaes $X \to Y \in F^+$ tal que $XY \subseteq Z$.

Dados R y $\rho = (R_1, R_2, \ldots, R_k)$ y F entonces ρ preserva F si la unión de todas las dependencias funcionales en $\pi_{R_i}(F)$ para $i = 1 \ldots k$ implica lógicamente F

$F^+ = (\bigcup_{i=1}^{k} \pi_{R_i}(F))^+$

Tomar en cuenta que ρ podría ser SPI y SPDF (o ninguna de las dos) o SPI pero no SPDF (o al revés)

3.2.1. Algoritmo para Testear Perdidas de Dependiencias

Hay un algoritmo para probar si una descomposición ρ preserva las dependencias funcionales sin necesidad de calcular F^+.

La idea es computar $Z \cup ((Z \cap R_i)^+ \cap R_i)$ donde Z inicialmente es igual al lado izquierdo de la dependencia funcional $X \to Y$ que se desea testear.
La clausura de $(Z \cap R_i)$ es con respecto a F.

Dados R, F y ρ, queremos verificar si se preserva $X \to Y$,

Z:= X;
mientras Z *cambie* **hacer**
  para $i \leftarrow 1$ **a** k **hacer**
    $Z \cup ((Z \cap R_i)^+ \cap R_i)$
  fin
fin

Si al finalizar $Y \subseteq Z$ entonces $X \to Y$ está en F^+ Si esto esto se cumple para toda dependencia funcional $X \to Y$ entonces podemos afirmar que ρ es SPDF. Obviamente se podría finalizar el algoritmo si se detecta que $Y \subseteq Z$ en algún paso.

F D1 : {B → C, D → A }
F D2 : {AB → C, C → A, C → D}
F D3 : {A → B, B → C, C → D}

Decidir cuáles de las siguientes descomposiciones son lossless-join y preservan dependencias funcionales:

(a) F D1 : (A, D) y (B, C)

quiero testear $B \rightarrow C$
$Z = B$
$B \cup ((B \cap (A,D))^+ \cap (A,D)) = B$
$B \cup ((B \cap (B,C))^+ \cap (B,C)) = (B,C)$

quiero testear $D \rightarrow A$
$Z = D$
$D \cup ((D \cap (A,D))^+ \cap (A,D)) = (A, D)$
$D \cup ((D \cap (B,C))^+ \cap (B,C)) = D$

es SPDF

(b) F D2 : (A, B, D) y (A, B, C)

quiero testear $AB \rightarrow C$

$Z = AB$
$AB \cup ((AB \cap (A,B,D))^+ \cap (A,B,D)) = AB$
$AB \cup ((AB \cap (A,B,C))^+ \cap (A,B,C)) = ABC$

quiero testear $C \rightarrow A$ y $C \rightarrow D$

$Z = C$
$C \cup ((C \cap (A,B,D))^+ \cap (A,B,D)) = CD$
$C \cup ((C \cap (A,B,C))^+ \cap (A,B,C)) = CA$

es SPDF

(c) F D2 : (B, C, D) y (A, C)

quiero testear $AB \rightarrow C$

$Z = AB$
$AB \cup ((AB \cap (B,C,D))^+ \cap (B,C,D)) = AB$
$AB \cup ((AB \cap (A,C))^+ \cap (A,C)) = AB$

no preserva dependencias funcionales

(d) F D3 : (A, B, D) y (B, C)

quiero testear C → D

Z = C
C u ((C ∩ (A,B,D))+ ∩ (A,B,D)) = C
C u ((C ∩ (B,C))+ ∩ (B,C)) = CD

CD u ((CD ∩ (A,B,D))+ ∩ (A,B,D)) = CD
CD u ((CD ∩ (B,C))+ ∩ (B,C)) = CD

no preserva dependencias funcionales

(e) F D1 : (A, B, D) y (C, D)

quiero testear B → C
Z = B
B u ((B ∩ (A,B,D))+ ∩ (A,B,D)) = B
B u ((B ∩ (C,D))+ ∩ (C,D)) = B

no preserva dependencias funcionales

(f) F D3 : (A, B) y (A, C, D)

quiero testear B → C

Z = B
B u ((B ∩ (A,B))+ ∩ (A,B)) = B
B u ((B ∩ (A,C,D))+ ∩ (A,C,D)) = B

no preserva dependencias funcionales

1.4. Sea R(A,B,C,D,E,F,G,H,I) y DF: {A → B, CD → F, H → AD, I → C, D → H}

Decir si las siguientes descomposiciones son SPI:

3.1.2. Algoritmo de TABLEAU

La regla de la descomposición binaria sólo se aplica cuando se descompone una relación en dos subesquemas. Para el caso más general tenemos un algoritmo que lo cubre. Es el algoritmo del *Tableau*:

Dados R, F y $\rho = (R_1, R_2, \ldots, R_k)$, inicialmente se construye un tableau (matriz) T inicial T_0 donde las columnas son los atributos y las filas los subesquemas de ρ. Luego completamos el tableau con símbolos distinguidos (a_j) si $A_j \in R_i$ y con los que denominamos no distinguidos (b_{ij}) si $A_j \notin R_i$..

Luego vamos modificando el tableau aplicando las dependencias funcionales de las siguiente forma, para cada $X \to A$ si $fila_i[X] = fila_h[X]$ y $fila_i[A] \neq fila_h[A]$ hacemos

(i) si $fila_i[A] = a_j$ hacemos $fila_h[A] = a_j$ o

(ii) si $fila_i[A] = b_{ij}$ y $fila_h[A] = b_{hj}$ hacemos $fila_h[A] = b_{ij}$

Así vamos generando una serie $T_0, T_1, T_2, \ldots, T^*$. terminamos cuando no se puedan hacer más cambios en el tableau final (T^*) aplicando las dependencias funcionales. Si T^* contiene una fila con todos símbolos distinguidos entonces ρ es **SPI**, de lo contrario ρ es con pérdida de información.

(a) R1(A,B,D), R2(D,E,F), R3(F,G,C), R4(C,H,I)

	A	B	C	D	E	F	G	H	I
ABD	a1	a2	b13	a4	b15	b16	b17	b18	b19
DEF	b21	b22	b23	a4	a5	a6	b27	b28	b29
FGC	b31	b32	b33	b34	b35	a6	a7	a8	b39
CHI	b41	b42	a3	b44	b45	b46	b47	a8	a9

uso D → H, igualo b18 con b23, queda:

	A	B	C	D	E	F	G	H	I
ABD	a1	a2	b13	a4	b15	b16	b17	b18	b19
DEF	b21	b22	b23	a4	a5	a6	b27	b18	b29
FGC	b31	b32	b33	b34	b35	a6	a7	a8	b39
CHI	b41	b42	a3	b44	b45	b46	b47	a8	a9

uso H → AD

	A	B	C	D	E	F	G	H	I
ABD	a1	a2	b13	a4	b15	b16	b17	b18	b19
DEF	a1	b22	b23	a4	a5	a6	b27	b18	b29
FGC	b31	b32	b33	b34	b35	a6	a7	a8	b39
CHI	b31	b42	a3	b34	b45	b46	b47	a8	a9

uso A → B

	A	B	C	D	E	F	G	H	I
ABD	a1	a2	b13	a4	b15	b16	b17	b18	b19
DEF	a1	a2	b23	a4	a5	a6	b27	b18	b29
FGC	b31	b32	b33	b34	b35	a6	a7	a8	b39
CHI	b31	b32	a3	b34	b45	b46	b47	a8	a9

Usados en negrita: A → B, CD → F, **H → AD**, I → C, **D → H.**
Parece que no puedo seguir. Como no quedó fila con símbolos distinguidos es con perdida de información.

(b) R1(A,B,C,D), R2(E,F,G), R3(H,I) y DF: {A → B, CD → F, H → AD, I → C, D → H}

	A	B	C	D	E	F	G	H	I
ABCD	a1	a2	a3	a4	b15	b16	b17	b18	b19
EFG	b21	b22	b23	b24	a5	a6	a7	b28	b29
HI	b31	b32	b33	b34	b35	b36	b37	a8	a9

No puedo hacer nada. Es con pérdida de información

2.1. Para el ejercicio 1.4, indicar:
(a) En que Forma Normal se encuentra R.

Seguro está en 1FN, veamos si está en 2FN.
Sea R(A,B,C,D,E,F,G,H,I) y DF: {A → B, CD → F, H → AD, I → C, D → H}

Busquemos la clave:
obs: I no está nunca del lado derecho, debe pertenecer a la clave y vemos que I → C.
obs: D → H → AD (puedo tener D en la clave y no pertenecen A y H).
obs: A → B, pero D → AD y por lo tanto D → A, por transitividad D → B

tenemos DI → {ABCDHI} y a su vez todo ABCDHI → ABCDFHI

Es la clave minimal.

Definición: 2FN

Un esquema de relación R está en **2FN** si todo atributo no primo A de R es totalmente dependiente de todas las claves de R.

Definición: Atributo Primo

Un atributo A de un esquema relacional R se dice primo si A pertenece a alguna de las claves de R. Si A no pertenece a ninguna clave de R, A se dice no primo.

Definición: Totalmente Dependiente

$X \to Y$ es una dependencia funcional es total si no existe $Z \subsetneq X$ tal que $Z \to Y$.

Sabemos que I siempre está en la clave y tenemos que I → C, por lo que C no es primo ya que no pertenece a ninguna clave .Vemos que no depende totalmente de toda la clave DI, solo I.
No está en 2FN.

(b) Descomponer en 3FN SPI y SPDF.

(c) Descomponer en FNBC SPI utilizando el algoritmos de descomposición visto en clase teórica, es decir, NO se debe partir de la descomposición en 3FN

Def FNBC: si R es un esquema de relación descompuesto en los esquemas R1, R2, ..., Rk y F es un conjunto de dependencias, decimos que la descomposición está en FNBC si para cada relación Ri se cumple que para toda dependencia funcional no trivial X → A, X es superclave en Ri

2.2. Dado el siguiente esquema de relación que describe páginas web:
Pagina(URL, autor, titulo, keyword)

Una tupla < u, a, t, k > de la relación dice que la URL u posee título t, autor a y contiene la clave de búsqueda k. Cada página posee exactamente un título, un autor y está unívocamente identificada con una URL. Una página puede tener muchas keywords.

Dar un conjunto de dependencias funcionales para Página y demostrar que no se encuentra en FNBC.

ej; (url, titulo, autor, keyword1), …, (url, titulo, autor, keywordn)
url → título, autor

url no es superclave, pues no define la keyword ya que puede tener varias. Por lo tanto no está en FNBC.

2.3. Se desea modelar la actividad de un broker bursátil, quien maneja las carteras de acciones de varios inversores. Los atributos relevantes son: B (broker), I (inversor), E (domicilio comercial del broker), A (acción de una empresa que cotiza en bolsa), D (dividendo), C (cantidad de acciones).

Además se cumplen las dependencias funcionales F: {A → D, I → B, IA → C, B → E}

(a) Determinar una clave y demostrar que realmente lo es.

Posible clave: AI (A e I nunca aparecen del lado derecho de df)
I → B → E
A → D
IA → C

(b) Si se descompone el esquema en D1= {{I, B}, {I, A, C}, {A, D}, {I, A, E}}. La descomposición, ¿es SPI? ¿es SPDF?

	A	B	C	D	E	I
IB	b11	a2	b13	b14	b15	a6
IAC	a1	b22	a3	b24	b25	a6
AD	a1	b32	b33	a4	b35	b36
IAE	a1	b42	b43	b44	a5	a6

Uso A → D

	A	B	C	D	E	I
IB	b11	a2	b13	b14	b15	a6
IAC	a1	b22	a3	a4	b25	a6
AD	a1	b32	b33	a4	b35	b36
IAE	a1	b42	b43	a4	a5	a6

Uso IA → C

	A	B	C	D	E	I
IB	b11	a2	b13	b14	b15	a6
IAC	a1	b22	a3	a4	b25	a6
AD	a1	b32	b33	a4	b35	b36
IAE	a1	b42	a3	a4	a5	a6

Uso I → B

	A	B	C	D	E	I
IB	b11	a2	b13	b14	b15	a6
IAC	a1	a2	a3	a4	b25	a6
AD	a1	b32	b33	a4	b35	b36
IAE	a1	a2	a3	a4	a5	a6

Última fila solo con símbolos distinguidos, es sin pérdida de información.

(c) Verificar si la descomposición está en 3FN. Si no lo está, dar una descomposición que esté en 3FN.