Assignment 6: Decomposition and Normal Forms

Hien Tu - tun1

November 23, 2021

Part 1: The analysis of a quick-event wizard for a local community

1. Minimal cover of all realistic non-trivial functional dependencies:

   - Since each event is organized by a user, we know that if the event id (*id*) is the same, then the organizer (*user_id*) must be the same. So, we can have id $\longrightarrow$ user_id. The reverse would not hold since an organizer could organize many events.
   - Furthermore, each event happens in a day, so, we know that if the event id (*id*) is the same, the date that the event happens in (*date*) must be the same. So, we can have id $\longrightarrow$ date.
   - We also have id, inv_id $\longrightarrow$ inv_confirmed since each guest would determine what event to go to. We need both the event id (*id*) and the guest (*inv_id*) to determine if the invitation is confirmed (*inv_confirmed*). Only the guest (*inv_id*) would not be able to determine *inv_confirmed* since we also need to know what event the guest is confirming. Similarly, only the event id (*id*) would not be able to determine *inv_confirmed* since we also need to know who confirmed going to the event.
   - Furthermore, we have id, product $\longrightarrow$ p_amount since we would only know how much to bring if we know what to bring and where we need to bring it to. Only the product wouldn't be able to determine the amount since different event could need different amount of the same product. For example, in the given table, event 1 needs 4 chips while event 2 only needs 2 chips.

- We have product $\longrightarrow$ p_price since each product has its own price. The price of the product wouldn't depend on the event that the product is brough into. An example is that, in the given table, the chips cost \$2 no matter if it is brought into event with id 1 or 2 and the cola would cost \$4 no matter if it is brought into event with id 1 or 2.
- So, the cover is {id $\longrightarrow$ user_id, id $\longrightarrow$ date, id, inv_id $\longrightarrow$ inv_confirmed, id, product $\longrightarrow$ p_amount, product $\longrightarrow$ p_price}. Note that this is already minimal since there is no dependency that is trivial or can be derived from the other dependencies in the cover. Therefore, this is the minimal cover of all realistic non-trivial functional dependencies.

2. An example of a non-trivial dependency is id $\twoheadrightarrow$ inv_id, inv_confirmed. This would hold since, for example, in the given table, from the first row and the fourth row, we know that there would exist a row where inv_id and inv_confirmed are the same as the first row and the rest of the attributes (user_id, date, product, p_price, p_amount) are the same as the fourth row. This row is the thrid row. We also know that there would exist a row where inv_id and inv_confirmed are the same as the fourth row and the rest of the attributes (user_id, date, product, p_price, p_amount) are the same as the first row. This row is the second row.

Part 2: Refinement of an order-table for a cinema chain
We will use the short hand notation for each attribute for brevity.

3. The relational schema is not in 3NF since I $\longrightarrow$ St, Si, Ss, Sd would violate the 3NF property since {St, Si, Ss, Sd} $\not\subseteq$ I, I is not a (super)key and each attribute in {St, Si, Ss, Sd} $\setminus$ I = {St, Si, Ss, Sd} is not part of a key.

   To decompose, first, we need to compute the minimal cover of the functional dependencies

- We can use Decomposition on the first three functional dependencies to get
  {I $\longrightarrow$ St; I $\longrightarrow$ Si; I $\longrightarrow$ Ss; I $\longrightarrow$ Sd; I $\longrightarrow$ Fi; I $\longrightarrow$ Fl; I $\longrightarrow$ Fs; I $\longrightarrow$ Ri; I $\longrightarrow$ Rs; Si $\longrightarrow$ Si; Si $\longrightarrow$ Ss; Si $\longrightarrow$ Sd; Ss $\longrightarrow$ Sd; Sd $\longrightarrow$ Ss; St, Ri $\longrightarrow$ Fi; Fi $\longrightarrow$ Fl; Fi, Si $\longrightarrow$ Fs; Ri $\longrightarrow$ Rs}
- From Reflexivity, since Si $\subseteq$ Si, we can get Si $\longrightarrow$ Si. So, we don't need to include Si $\longrightarrow$ Si in the minimal cover.
  By Transitivity on I $\longrightarrow$ Si and Si $\longrightarrow$ Ss, we can get I $\longrightarrow$ Ss. So, we don't need I $\longrightarrow$ Ss.
  By Transitivity on I $\longrightarrow$ Si and Si $\longrightarrow$ Sd, we can get I $\longrightarrow$ Sd. So, we don't need I $\longrightarrow$ Sd.
  By Union on I $\longrightarrow$ St and I $\longrightarrow$ Ri; we can get I $\longrightarrow$ St, Ri. By Transitivity on I $\longrightarrow$ St, Ri and St, Ri $\longrightarrow$ Fi; we can get I $\longrightarrow$ Fi. So we don't need I $\longrightarrow$ Fi.
  By Transitivity on I $\longrightarrow$ Fi (that we just proved) and Fi $\longrightarrow$ Fl, we can get I $\longrightarrow$ Fl. So, we don't need I $\longrightarrow$ Fl.
  By Union on I $\longrightarrow$ Fi and I $\longrightarrow$ Si; we can get I $\longrightarrow$ Fi, Si. By Transitivity on I $\longrightarrow$ Fi, Si and Fi, Si $\longrightarrow$ Fs, we can get I $\longrightarrow$ Fs. So, we don't need I $\longrightarrow$ Fs.
  By Transitivity on Si $\longrightarrow$ Ss and Ss $\longrightarrow$ Sd, we can get Si $\longrightarrow$ Sd. So, we don't need Si $\longrightarrow$ Sd.
  Thus, we would get
  {I $\longrightarrow$ St; I $\longrightarrow$ Si; I $\longrightarrow$ Ri; Si $\longrightarrow$ Ss; Ss $\longrightarrow$ Sd; Sd $\longrightarrow$ Ss; St, Ri $\longrightarrow$ Fi; Fi $\longrightarrow$ Fl; Fi, Si $\longrightarrow$ Fs; Ri $\longrightarrow$ Rs}
- Notice that the set above is already minimal. So it is the minimal cover.

Starting the algorithm, we get know

- *result* = {}
- *cover* = {I $\longrightarrow$ St; I $\longrightarrow$ Si; I $\longrightarrow$ Ri; Si $\longrightarrow$ Ss; Ss $\longrightarrow$ Sd; Sd $\longrightarrow$ Ss; St, Ri $\longrightarrow$ Fi; Fi $\longrightarrow$ Fl; Fi, Si $\longrightarrow$ Fs; Ri $\longrightarrow$ Rs}

First, we have I $\longrightarrow$ St

- Since we have I $\longrightarrow$ St, I $\longrightarrow$ Si, I $\longrightarrow$ Ri $\in$ *cover*, we get $B = \{St, Si, Ri\}$
- So, *result* = {(I, St, Si, Ri)}

Next, we have Si $\longrightarrow$ Ss

- Since only Si $\longrightarrow$ Ss $\in$ *cover* that starts with Si, $B = \{Ss\}$
- So, *result* = {(I, St, Si, Ri), (Si, Ss)}

Then, we have Ss $\longrightarrow$ Sd

- Since only Ss $\longrightarrow$ Sd $\in$ *cover* that starts with Ss, $B = \{Sd\}$
- So, *result* = {(I, St, Si, Ri), (Si, Ss), (Ss, Sd)}

Then, we have Sd $\longrightarrow$ Ss

- Since only Sd $\longrightarrow$ Ss $\in$ *cover* that starts with Sd, $B = \{Ss\}$
- So, *result* = {(I, St, Si, Ri), (Si, Ss), (Ss, Sd), (Sd, Ss)}

Next, we have St, Ri $\longrightarrow$ Fi

- Since only St, Ri $\longrightarrow$ Fi $\in$ *cover* that starts with St, Ri; $B = \{Fi\}$
- So, *result* = {(I, St, Si, Ri), (Si, Ss), (Ss, Sd), (Sd, Ss), (St, Ri, Fi)}

Next, we have Fi $\longrightarrow$ Fl

- Since only Fi $\longrightarrow$ Fl $\in$ *cover* that starts with Fi, $B = \{Fl\}$
- So, *result* = {(I, St, Si, Ri), (Si, Ss), (Ss, Sd), (Sd, Ss), (St, Ri, Fi), (Fi, Fl)}

Next, we have Fi, Si $\longrightarrow$ Fs

- Since only Fi, Si $\longrightarrow$ Fs $\in$ *cover* that starts with Fi, Si; $B = \{Fs\}$
- So, *result* = {(I, St, Si, Ri), (Si, Ss), (Ss, Sd), (Sd, Ss), (St, Ri, Fi), (Fi, Fl), (Fi, Si, Fs)}

Finally, we have Ri $\longrightarrow$ Rs

- Since only Ri $\longrightarrow$ Rs $\in$ *cover* that starts with Ri, $B = \{Fi\}$
- So, *result* = {(I, St, Si, Ri), (Si, Ss), (Ss, Sd), (Sd, Ss), (St, Ri, Fi), (Fi, Fl), (Fi, Si, Fs), (Ri, Rs)}

Since *result* doesn't contain the key, which is "I, P, Rp", we need to add it to *result*. So, *result* = {(I, St, Si, Ri), (Si, Ss), (Ss, Sd), (Sd, Ss), (St, Ri, Fi), (Fi, Fl), (Fi, Si, Fs), (Ri, Rs), (I, P, Rp)}
Since (Sd, Ss) $\subseteq$ (Ss, Sd), we will remove (Sd, Ss) from *result*. Thus, *result* = {(I, St, Si, Ri), (Si, Ss), (Ss, Sd), (St, Ri, Fi), (Fi, Fl), (Fi, Si, Fs), (Ri, Rs), (I, P, Rp)}

Minimal cover for each relational schema of the resulting decomposition

- (I, St, Si, Ri)
  $\Longrightarrow$ {I $\longrightarrow$ St; I $\longrightarrow$ Si, I $\longrightarrow$ Ri}
- (Si, Ss)
  $\Longrightarrow$ {Si $\longrightarrow$ Ss}
- (Ss, Sd)
  $\Longrightarrow$ {Ss $\longrightarrow$ Sd}
- (St, Ri, Fi)
  $\Longrightarrow$ {St, Ri $\longrightarrow$ Fi}
- (Fi, Fl)
  $\Longrightarrow$ {Fi $\longrightarrow$ Fl}
- (Fi, Si, Fs)
  $\Longrightarrow$ {Fi, Si $\longrightarrow$ Fs}
- (Ri, Rs)
  $\Longrightarrow$ {Ri $\longrightarrow$ Rs}

From the minimal covers of the relational schemas of the resulting decomposition, we can see that the decomposition is dependency-preserving. It is also proved in the lecture that Decompose-3NF is dependency-preserving.

Notice that we know that we can join (I, St, Si, Ri) and (Si, Ss) on Si. Then, we can join the result with (Ss, Sd) on Ss. After that, we can join the result with (St, Ri, Fi) on St. Then, we join with (Fi, Fl) and (Fi, Si, Fs) on Fi. Finally, we can join the result with (Ri, Rs) on Ri. Furthermore, it is proved in the lecture that Decompose-3NF is also lossless-join. Thus, this decomposition should be lossless-join as well.

Decompose the example dataset

- (I, St, Si, Ri)

I	St	Si	Ri
1	Nov. 1, 1pm	1	7
2	Nov. 1, 1pm	2	7
3	Nov. 7, 2pm	2	3

- (Si, Ss)

Si	Ss
1	Oct. 1
2	Oct. 3

- (Ss, Sd)

Ss	Sd
Oct. 1	31
Oct. 3	29

- (St, Ri, Fi)

St	Ri	Fi
Nov. 1, 1pm	7	5
Nov. 7, 2pm	3	9

- (Fi, Fl)

Fi	Fl
5	120
9	99

- (Fi, Si, Fs)

Fi	Si	Fs
5	1	great
5	2	awful
9	2	not-scored

- (Ri, Rs)

Ri	Rs
7	medium
3	large

4. The relational schema is not in BCNF since I $\longrightarrow$ St, Si, Ss, Sd $\in \mathfrak{S}^+$ but I is not a (super)key.

   Decompose-BCNF:

   Since I $\longrightarrow$ St, Si, Ss, Sd $\in \mathfrak{S}^+$ violates the BCNF constraint since I is not the (super)key, then

   - $\mathbf{R}_1$ = I$^+$ = {I, St, Si, Ss, Sd, Fi, Fl, Fs, Ri, Rs}
   - $\mathbf{R}_2$ = I $\cup$ Z = {I, P, Rp}
   - Note that {I, P, Rp} satisfies the BCNF constraint, so they would be part of the resulting decomposition.

   For $\mathbf{R}_1$, since Si $\longrightarrow$ Si, Ss, Sd $\in \mathfrak{S}_1^+$ violates the BCNF constraint since Si is not the (super)key, then

   - $\mathbf{R}_{1,1}$ = (Si)$^+$ = {Si, Ss, Sd}
   - $\mathbf{R}_{1,2}$ = Si $\cup$ Z = {Si, I, St, Fi, Fl, Fs, Ri, Rs}

   For $\mathbf{R}_{1,1}$, since Ss $\longrightarrow$ Sd $\in \mathfrak{S}_{1,1}^+$ violates the BCNF constraint, then

   - $\mathbf{R}_{1,1,1}$ = (Ss)$^+$ = {Ss, Sd}
   - $\mathbf{R}_{1,1,2}$ = Ss $\cup$ Z = {Ss, Si}
   - Note that {Ss, Sd} and {Ss, Si} satisfies BCNF, so they would be part of the resulting decomposition.

   For $\mathbf{R}_{1,2}$, since St, Ri $\longrightarrow$ Fi $\in \mathfrak{S}_{1,2}^+$ violates the BCNF constraint, then

   - $\mathbf{R}_{1,2,1}$ = {St, Ri}$^+$ = {St, Ri, Fi, Fl, Rs}
   - $\mathbf{R}_{1,2,2}$ = {St, Ri} $\cup$ Z = {St, Ri, I, Fs, Si}

   For $\mathbf{R}_{1,2,1}$, since Fi $\longrightarrow$ Fl $\in \mathfrak{S}_{1,2,1}^+$ violates the BCNF constraint, then

   - $\mathbf{R}_{1,2,1,1}$ = (Fi)$^+$ = {Fi, Fl}
   - $\mathbf{R}_{1,2,1,2}$ = Fi $\cup$ Z = {Fi, St, Ri, Rs}

- Note that {Fi, Fl} satisfies the BCNF constraint, so they would be part of the resulting decomposition.

For $\mathbf{R}_{1,2,1,2}$, since Ri $\longrightarrow$ Rs $\in \mathfrak{S}^{+}_{1,2,1,2}$ violates the BCNF constraint, then

- $\mathbf{R}_{1,2,1,2,1}$ = (Ri)$^{+}$ = {Ri, Rs}
- $\mathbf{R}_{1,2,1,2,2}$ = Ri $\cup$ Z = {Ri, Fi, St}
- Noe that {Ri, Rs} and {Ri, Fi, St} satisfies the BCNF constraint, so they would be part of the resulting decomposition.

For $\mathbf{R}_{1,2,2}$ = {I, St, Si, Fs, Ri}: We have St, Ri $\longrightarrow$ Fi. By Augmentation on that functional dependency with Si; we get St, Ri, Si $\longrightarrow$ Fi, Si. From the one we just got and Fi, Si $\longrightarrow$ Fs; using Transitivity; we get St, Ri, Si $\longrightarrow$ Fs. However, this dependency violates the BCNF constraint since (St, Ri, Si) is not (super)key. Thus, we will decompose $\mathbf{R}_{1,2,2}$ into

- $\mathbf{R}_{1,2,2,1}$ = {St, Ri, Si}$^{+}$ = {St, Ri, Si, Fs}
- $\mathbf{R}_{1,2,2,2}$ = {St, Ri, Si} $\cup$ Z = {St, Ri, Si, I}
- Note that {St, Ri, Si, Fs} and {St, Ri, Si, I} satisfy the BCNF constraint, so they would be part of the resulting decomposition.

All relational schemas in the resulting decomposition are
{I, P, Rp}, {Ss, Sd}, {Si, Ss}, {Fi, Fl}, {Ri, Rs}, {Ri, Fi, St}, {St, Ri, Si, Fs}, {St, Ri, Si, I}.

Minimal cover for each relational schema

- (I, P, Rp)
  $\Longrightarrow$ {}
- (Ss, Sd)
  $\Longrightarrow$ {Ss $\longrightarrow$ Sd, Sd $\longrightarrow$ Ss}
- (Si, Ss)
  $\Longrightarrow$ {Si $\longrightarrow$ Ss}
- (Fi, Fl)
  $\Longrightarrow$ {Fi $\longrightarrow$ Fl}

- (Ri, Rs)
  $\Longrightarrow$ {Ri $\longrightarrow$ Rs}
- (Ri, Fi, St)
  $\Longrightarrow$ {St, Ri $\longrightarrow$ Fi}
- (St, Ri, Si, Fs)
  $\Longrightarrow$ {St, Ri, Si $\longrightarrow$ Fs}
- (St, Ri, Si, I)
  $\Longrightarrow$ {I $\longrightarrow$ St, I $\longrightarrow$ Ri, I $\longrightarrow$ Si}

This decomposition is not dependency-preserving since we lost the dependencies Fi, Si $\longrightarrow$ Fs.

The decomposition is lossless-join since we can join (I, P, Rp) and (St, Ri, Si, I) on I. Then, we can join this result with (Si, Ss) on Si. After that, we can join the result with (Ss, Sd) on Ss. Then, we can join with (Ri, Rs), (Ri, Fi, St) and (St, Ri, Si, Fs) on Ri. Finally, we can join with (Fi, Fl) on Fi. Furthermore, it is proved in the lecture that DECOMPOSE-BCNF is lossless-join. Thus, the decomposition is lossless-join.

Decompose the example data set

- (I, P, Rp)

I	P	Rp
1	ticket	3D
1	ticket	Dolby
1	3D	3D
1	3D	Dolby
2	ticket	3D
2	ticket	Dolby
2	3D	3D
2	3D	Dolby
3	ticket	IMAX
3	IMAX	IMAX
3	ticket	4D
3	IMAX	4D

- (Ss, Sd)

Ss	Sd
Oct. 1	31
Oct. 3	29

- (Si, Ss)

Si	Ss
1	Oct. 1
2	Oct. 3

- (Fi, Fl)

Fi	Fl
5	120
9	99

- (Ri, Rs)

Ri	Rs
7	medium
3	large

- (St, Fi, Ri)

St	Fi	Ri
Nov. 1, 1pm	5	7
Nov. 7, 2pm	9	3

- (St, Si, Fs, Ri)

St	Si	Fs	Ri
Nov. 1, 1pm	1	great	7
Nov. 1, 1pm	2	awful	7
Nov. 7, 2pm	2	not-scored	3

- (I, St, Si, Ri)

I	St	Si	Ri
1	Nov. 1, 1pm	1	7
2	Nov. 1, 1pm	2	7
3	Nov. 7, 2pm	2	3

5. The relational schema is not in 4NF since, although the multi-valued functional dependencies ID $\longrightarrow$ P and ID $\longrightarrow$ Rp satisfy the 4NF constraint (as ID is the super key), we have other multi-valued dependencies in $\mathfrak{S}^+$ that does not satisfy the 4NF constraint. More specifically, since I $\longrightarrow$ St, Si, Ss, Sd; by Replication, we have I $\twoheadrightarrow$ St, Si, Ss, Sd. Thus, I $\twoheadrightarrow$ St, Si, Ss, Sd $\in \mathfrak{S}^+$. However, I is not a (super)key. Therefore, it violates the 4NF constraint and the relational schema is not in 4NF.

   We will decompose the relational schema using the Decompose-4NF algorithm

   - As mentioned before, by using Decomposition rule on I $\longrightarrow$ St, Si, Ss, Sd and I $\longrightarrow$ Fi, Fl, Fs, Ri, Rs; we get
     I $\longrightarrow$ St, I $\longrightarrow$ Si, I $\longrightarrow$ Ss, I $\longrightarrow$ Sd, I $\longrightarrow$ Fi, I $\longrightarrow$ Fl, I $\longrightarrow$ Fs, I $\longrightarrow$ Ri, I $\longrightarrow$ Rs
   - By Replication on I $\longrightarrow$ St, we get I $\twoheadrightarrow$ St, which violates the 4NF constraint since I is not a (super)key. We will decompose into
     - $\mathbf{R}_1$ = {I, St}
     - $\mathbf{R}_2$ = {I, Si, Ss, Sd, Fi, Fl, Fs, Ri, Rs, P, Rp}
     - Note that {I, St} is in 4NF since I is the key of $\mathbf{R}_1$
   - For $\mathbf{R}_2$, since I $\longrightarrow$ Si, by Replication, we get I $\twoheadrightarrow$ Si, which violates the 4NF constraint since I is not the (super)key. Thus, we will decompose $\mathbf{R}_2$ into
     - $\mathbf{R}_{2,1}$ = {I, Si}
     - $\mathbf{R}_{2,2}$ = {I, Ss, Sd, Fi, Fl, Fs, Ri, Rs, P, Rp}
     - Note that {I, Si} is in 4NF since I is the key of $\mathbf{R}_{2,1}$
   - By repeatedly applying the same argument to I $\longrightarrow$ Ss, I $\longrightarrow$ Sd, I $\longrightarrow$ Fi, I $\longrightarrow$ Fl, I $\longrightarrow$ Fs, I $\longrightarrow$ Ri, I $\longrightarrow$ Rs; we get
     {I, Ss}, {I, Sd}, {I, Fi}, {I, Fl}, {I, Fs}, {I, Ri}, {I, Rs}, {I, P, Rp}
     Note that from the above sets, all except {I, P, Rp} are in 4NF since I is the (super)key.
   - For {I, P, Rp}, by Reflexivity, we know I $\longrightarrow$ I, by Union on I $\longrightarrow$ St, Si, Ss, Sd and I $\longrightarrow$ Fi, Fl, Fs, Ri, Rs; we get I $\longrightarrow$ St, Si,

Ss, Sd, Fi, Fl, Fs, Ri, Rs. Applying this with I $\longrightarrow$ I using Union rule, we get I $\longrightarrow$ I, St, Si, Ss, Sd, Fi, Fl, Fs, Ri, Rs. Note that ID = {I, St, Si, Ss, Sd, Fi, Fl, Fs, Ri, Rs}, so I $\longrightarrow$ ID. Then, using Replication, we get I $\twoheadrightarrow$ ID. From the question, we also have ID $\twoheadrightarrow$ P. By using Transitivity, we get I $\twoheadrightarrow$ P $\setminus$ ID. Since ID does not include P, P $\setminus$ ID = P. Thus, we get I $\twoheadrightarrow$ P. Since I $\twoheadrightarrow$ P violates the 4NF constraint, we will decompose {I, P, Rp} into {I, P} and {I, Rp}, which are in 4NF.

- Therefore, all of the relational schemas in the resulting decomposition are {I, St}, {I, Si}, {I, Ss}, {I, Sd}, {I, Fi}, {I, Fl}, {I, Fs}, {I, Ri}, {I, Rs}, {I, P}, {I, Rp}.

Minimal cover of each relational schema in the resulting decomposition:

- {I, St} $\implies$ {I $\longrightarrow$ St}
- {I, Si} $\implies$ {I $\longrightarrow$ Si}
- {I, Ss} $\implies$ {I $\longrightarrow$ Ss}
- {I, Sd} $\implies$ {I $\longrightarrow$ Sd}
- {I, Fi} $\implies$ {I $\longrightarrow$ Fi}
- {I, Fl} $\implies$ {I $\longrightarrow$ Fl}
- {I, Fs} $\implies$ {I $\longrightarrow$ Fs}
- {I, Ri} $\implies$ {I $\longrightarrow$ Ri}
- {I, Rs} $\implies$ {I $\longrightarrow$ Rs}
- {I, P} $\implies$ {I $\twoheadrightarrow$ P}
- {I, Rp} $\implies$ {I $\twoheadrightarrow$ Rp}

The decomposition is lossless-join since we can do natural join on all the decomposed relational schemas on I (the event id *id*).

The decomposition is not dependency-preserving since we lost Si $\longrightarrow$ Si, Ss, Sd; Ss $\longrightarrow$ Sd; Sd $\longrightarrow$ Ss; St, Ri $\longrightarrow$ Fi and so on.

Decompose the example data set

- (I, St)

I	St
1	Nov. 1, 1pm
2	Nov. 1, 1pm
3	Nov. 7, 2pm

- (I, Si)

I	Si
1	1
2	2
3	2

- (I, Ss)

I	Ss
1	Oct. 1
2	Oct. 3
3	Oct. 3

- (I, Sd)

I	Sd
1	31
2	29
3	29

- (I, Fi)

I	Fi
1	5
2	5
3	9

- (I, Fl)

I	Fl
1	120
2	120
3	99

- (I, Fs)

I	Fs
1	great
2	awful
3	not-scored

- (I, Ri)

I	Ri
1	7
2	7
3	3

- (I, Rs)

I	Rs
1	medium
2	medium
3	large

- (I, P)

I	P
1	ticket
1	3D
2	ticket
2	3D
3	ticket
3	IMAX

- (I, Rp)

I	Rp
1	3D
1	Dolby
2	3D
2	Dolby
3	IMAX
3	4D

6. Since Decompose-BCNF and Decompose-4NF are not dependency-preserving, they do not resolve all design issues. Since Decompose-3NF is both dependency-preserving and lossless-join, it resolves all design issues.