BIT2008A/ITEC2000A: Multimedia Data Management
Fall 2022

Assignment 2

Entity Relationship Model & Normalization

Question 1. A weak entity set is one whose existence depends on another entity, called its identifying entity. Instead of associating a primary key with a weak entity, we use the identifying entity, along with auxiliary attributes called discriminator to uniquely identify a weak entity.

Question 2.

- Create an identifying attribute for E and add any attributes of R to E

- For each relationship (a_i , b_i , c_i) in R, create

1. a new entity e_i in the entity set E
2. add (e_i , a_i) to R_A
3. add (e_i , b_i) to R_B
4. add (e_i , c_i) to R_C

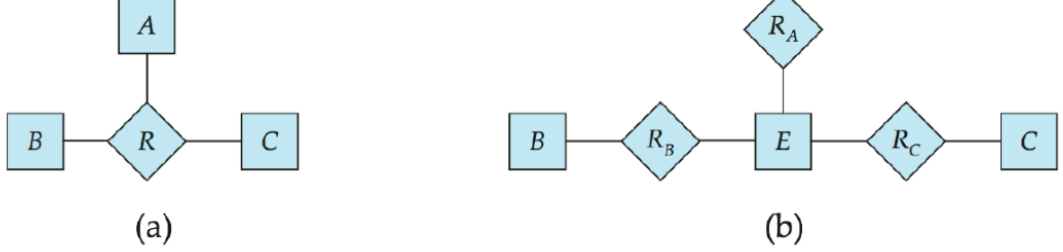

Figure 1

Question 3. There are various solutions to this question including:

- $A \to E$ and $E \to C$ give $A \to C$ by transitivity.
- $A \to B$ and $A \to C$ give $A \to BC$ by union.
- $A \to BC$ and $BC \to DE$ give $A \to DE$ by transitivity.
- $A \to B$, $E \to C$, and $AEF \to G$ give $BCF \to G$ by pseudo-transitivity.

Question 4. The closure of (EFH) is calculated as below:

- $(EFH)^+ = EFH$, initial step.
- $(EFH)^+ = EFGH$ using functional dependency 1.
- $(EFH)^+ = EFGHIJ$ using functional dependency 2.
- $(EFH)^+ = EFGHIJKL$ using functional dependency 3.
- $(EFH)^+ = EFGHIJKLM$ using functional dependency 4.
- $(EFH)^+ = EFGHIJKLMN$ using functional dependency 5.

We conclude that EFH is a superkey because its closure covers all of the attributes. After we check its subsets, we realize that none of them is a superkey. Consequently, EFH is a candidate key.

Question 5. First we decompose right-hand side of functional dependencies $F = \{A \to B, A \to C, B \to A, B \to C, C \to A, C \to B\}$. We find that there are no extraneous attributes. Then, we remove redundant functional dependencies. $B \to C$ and $C \to A$ imply $B \to A$ so we can remove it. Also $A \to B$ and $B \to C$ give $A \to C$ so it is redundant. Finally, $C \to A$ and $A \to B$ imply $C \to B$ so we remove it. As a result, $F_c = \{A \to B, B \to C, C \to A\}$.

Question 6. AB is a candidate key (why?) so the dependency $D \to E$ violates BCNF. As a result, we decompose the relations to $\{D, E\}$ and $\{A, B, C, D\}$. However, for two more functional dependencies $A \to C$, and $B \to D$, the left-hand side is only part of our candidate key, so we decompose $ABCD$ into $\{A, C\}$, $\{B, D\}$, and $\{A, B\}$. Consequently, we end up having 4 relations in our decomposition. Since we can still retrieve $AB \to CD$ the decomposition is dependent-preserving.

Question 7. Although one can often skip ahead to some of the conclusions or combine steps, these solutions are very systematic so that you can see the full pattern.

(A) Compute all keys for R.
- Examining all subsets of attributes would be very time-consuming because there are 2^8 of them. With some careful reasoning we can speed up the process by avoiding computing many closures.
- On inspection, we can see that $A^+ = ACDBEFHG$, which means that A is a key and that no superset of A can be a key.
- Also, $CF^+ = CFAHGDBE$, which means that CF is a key and no superset of CF can be a key. (C alone or F alone could be part of a key, but CF cannot.)
- Therefore, the only keys are A and CF.

(B) Compute F_c. For convenience we sort them alphabetically.
- To find a minimal basis, we'll first eliminate redundant FDs. The order in which we do this will affect the results we get, but we will always get a minimal basis.
- We will simplify to singleton right-hand sides before doing so, since it may be possible to eliminate some but not all of FDs that we get from one of our original FDs. We'll also number the resulting FDs for easy reference, and call this set

  S1:

  1 $A \to C$
  2 $A \to D$
  3 $ACF \to G$
  4 $AD \to B$
  5 $AD \to E$
  6 $AD \to F$
  7 $BCG \to D$
  8 $CF \to A$
  9 $CF \to H$
  10 $CH \to G$
  11 $D \to B$
  12 $H \to D$
  13 $H \to E$

14 $H \to G$

- Now we'll look for redundant FDs to eliminate. Each row in the table below indicates which of the 14 FDs we still have on hand as we consider removing the next one. Of course, as we do the closure test to see whether we can remove $X \to Y$, we can't use $X \to Y$ itself, so an FD is never included in its own row.

FD	Exclude these from S1 when computing closure	Closure	Decision
1	1	There's no way to get C without this FD	keep
2	2	$A^+=AC$	keep
3	3	$ACF^+=ACFDBEHG$	**discard**
4	3, 4	$AD^+=ADCEFB$ …	**discard**
5	3, 4, 5	$AD^+=ADCFGBE$ …	**discard**
6	3, 4, 5, 6	There's no way to get F without this FD	keep
7	3, 4, 5, 7	$BCG^+= BCG$	keep
8	3, 4, 5, 8	There's no way to get A without this FD	keep
9	3, 4, 5, 9	There's no way to get H without this FD	keep
10	3, 4, 5, 10	$CH^+=CHDEG$…	**discard**
11	3, 4, 5, 10, 11	There's no way to get B without this FD	keep
12	3, 4, 5, 10, 12	$H^+=HEG$	keep
13	3, 4, 5, 10, 13	There's no way to get E without this FD	keep
14	3, 4, 5, 10, 14	There's no way to get G without this FD	keep

FIGURE 2

- Let's call the remaining FDs
  S2:
  1 $A \to C$
  2 $A \to D$
  3
  4
  5
  6 $AD \to F$
  7 $BCG \to D$
  8 $CF \to A$
  9 $CF \to H$
  10
  11 $D \to B$
  12 $H \to D$
  13 $H \to E$
  14 $H \to G$
- Now let's try reducing the LHS of any FDs with multiple attributes on the LHS. For these closures, we will close the entire set S2, including even the FD being considered for simplification. For functional dependency 6, $A^+ = ACDF$ that includes D inside it. so we can reduce the LHS to A. Finally, FD2 is also redundant and can be implied using FD1 and FD6 (union), then, FD9, and FD12 (transitivity). We have
  S3:
  1 $A \to C$
  2

3
4
5
6 $A \to F$
7 $BCG \to D$
8 $CF \to A$
9 $CF \to H$
10
11 $D \to B$
12 $H \to D$
13 $H \to E$
14 $H \to G$

- We have $F_c = \{A \to CF, BCG \to D, CF \to AH, D \to B, H \to DEG\}$

(C) Using F_c we employ the 3NF algorithm to obtain a lossless and dependency-preserving decomposition of relation R into a collection of relations that are in 3NF.

- The set of relations that would result would have these attributes:
  $R_1(A, C, F), R_2(B, C, D, G), R_3(A, C, F, H), R_4(B, D), R_5(D, E, G, H)$
- Since the attributes BD occur within R_2, we do not need to keep the relation R_3. Similarly, since the attributes ACF occur in R_3, we do not need to keep the relation R_1.
- A is a key of R, so there is no need to add another relation that includes a key.
- So the final set of relations is:
  $R_2(B, C, D, G), R_3(A, C, F, H), R_5(D, E, G, H)$