Exercise 07: Relational database design and normalisation

Task 1: Normalisation

Given the following relations R and S in the first normal form with functional dependencies $F = F_C$:

$$\begin{aligned} R &= (A, B, C, D, E, F) \; with \\ A, B &\rightarrow C, D, E \\ D &\rightarrow F \; and \\ & \{A, B\} \text{ is Key.} \end{aligned} \qquad \begin{aligned} S &= (V, W, X, Y, Z) \; with \\ V &\rightarrow W, Z \\ W, Z &\rightarrow V, X, Y \\ Y &\rightarrow Z \; and \\ & \{V\}, \{W, Z\}, \{W, Y\} \text{ are keys.} \end{aligned}$$

1. Show as briefly as possible that R does not correspond with the third normal form (3NF).
   To verify this property, we must first know the keys. The only key is A, B - see the last exercise. In order to show that a relation does not correspond with the 3NF, it is sufficient that a non-key attribute is transitively dependent on the key. This is the case through $D \rightarrow F$. F is not prime, D is not a key, and F is transitively dependent on key A, B.

2. Through decomposition, split R into relations that correspond with the third normal form (3NF). Are the resulting relations dependency preserving and lossless join decomposition?
   The functional dependency $D \rightarrow F$ violates the 3NF property. Decomposition:

$$\begin{aligned} R_1 &= \{D\}_F^+ = \{D, F\} \\ R_2 &= R - R_1 \cup \{D\} \\ &= \{A, B, C, D, E, F\} - \{D, F\} \cup \{D\} \\ &= \{A, B, C, D, E\}. \end{aligned}$$

   A decomposition of a schema that does not correspond with the 3NF is always dependency preserving and lossless join decomposition.

3. Split R using the synthesis process. Are the resulting relations dependency preserving and lossless join decomposition?

   (a) Compute the canonical cover. Already given: $F_C = F$
   (b) Create a relation for each left side from the attributes of the left side and the attributes that can be obtained with the left side:

$$\begin{aligned} R_1 &= \{A, B, C, D, E\} \\ R_2 &= \{D, F\} \end{aligned}$$

   (c) Does one of the generated relations contain the key? Yes, R_1 contains $\{A, B\}$.
   (d) Is one of the relations completely contained in another relation? No. Finished.

   The synthesis process is always dependency preserving but sometimes not a lossless join decomposition.,

4. Show as briefly as possible that S does not correspond with the Boyce-Codd normal form (BCNF). In order to show that a relation does not correspond with the BCNF, it is sufficient to show that a left side of a functional dependency is not a superkey. This is the case for $Y \rightarrow Z$ because $\{Y\}_F^+ = \{Y, Z\} \neq S$.

5. Through decomposition, split S into relations that correspond with the Boyce-Codd normal form (BCNF). Are the resulting relations dependency preserving and lossless join decomposition? Is the decomposition useful?

   The functional dependency $Y \rightarrow Z$ violates the BCNF property. Decomposition:

$$\begin{aligned} S_1 &= \{Y\}_F^+ = \{Y,Z\} \\ S_2 &= S - S_1 \cup \{Y\} \\ &= \{V,W,X,Y,Z\} - \{Y,Z\} \cup \{Y\} \\ &= \{V,W,X,Y\} \end{aligned}$$

A decomposition of a schema that does not correspond with the BCNF is always lossless join decomposition. Check dependency preservation:

$$\begin{aligned} F(S_1) &= \{(Y \to Z)\} \\ F(S_2) &= \{(V \to W)\} \\ F &\neq F(S_1) \cup F(S_2),\ denn\ es\ ist \\ F \cap (S_1 \cup S_2) &= \{(V \to Z), (W,Z \to V,X,Y)\} \end{aligned}$$

The decomposition is not dependency preserving and is therefore unusable.

6. Split S using the synthesis process. Are the resulting relations dependency preserving and lossless join decomposition?

   (a) Compute the canonical cover. Already given: $F_C = F$

   (b) Create a relation for each left side from the attributes of the left side and the attributes that can be obtained with the left side:

$$\begin{aligned} S_1 &= \{V,W,Z\} \\ S_2 &= \{V,W,X,Y,Z\} \\ S_3 &= \{Y,Z\} \end{aligned}$$

   (c) Does one of the generated relations contain the key? Yes, S_2 contains $\{V\}$, $\{W,Z\}$ and $\{W,Y\}$.

   (d) (d) Is one of the relations completely contained in another relation? Yes. Remove S_1 and S_3

   The synthesis process is always dependency preserving but sometimes not a lossless join decomposition, but the process has not brought us any improvement.

   Note: In the sample solution of the last exercise, an alternative solution was given for the cover F_C for S. If this alternative solution is used for the synthesis process, the following decomposition results:

$$\begin{aligned} S_1 &= \{V,W,X,Y\} \\ S_2 &= \{W,Z,V\} \\ S_3 &= \{Y,Z\} \end{aligned}$$

   This is also dependency preserving and lossless join decomposition and is better than the other solution for S, since no relation was created with all original attributes.

Task 2: Simplified synthesis process

Given the following relation R and functional dependencies F:

$$\begin{aligned} R &= (A,B,C,D,E,F) \\ A &\to B,C \\ D &\to E,F \end{aligned}$$

1. Obviously R does not correspond with the second normal form. The only key is $\{A,D\}$ and all prime attributes only depend on a real subset of the key. At which point in the synthesis process will a difficulty arise with the runtime?

After initial decomposition into

$$\begin{aligned} R_1 &= \{A, B, C\} \\ R_2 &= \{D, E, F\} \end{aligned}$$

none of the resulting relations contains a key, because $\{R_1\}_F^+ \neq R$ and $\{R_2\}_F^+ \neq R$. Thus, we must look for the key and in the worst case it costs us an exponential amount of runtime. In this case, we are lucky and our heuristics provide us with the appropriate key attributes.

2. If you used the simplified synthesis procedure with the additional rule $A, B, C, D, E, F \to \delta$, what is the improvement and what result is to be expected?

   If $A, B, C, D, E, F \to \delta$ is included, it will shorten to $A, D \to \delta$. This creates a new relation $R3$ with $\{A, B\}_F^+ = R$. We have thus successfully applied the synthesis process without calculating a key.

3. Carry out the simplified synthesis process for R.

   (a) Compute the canonical cover.

      i. $F_C = F$

      ii. $F_C = SPLITTING(F_C) =$

$$\begin{aligned} A &\to B \\ A &\to C \\ D &\to E \\ D &\to F \\ A, B, C, D, E, F &\to \delta \end{aligned}$$

      iii. Minimise left sides

$$\begin{aligned} \delta &\in \{A, D\}_F^+ = \{R\} \\ &\Rightarrow F_C = F_C - \{A, B, C, D, E, F \to \delta\} \cup \{A, D \to \delta\} \end{aligned}$$

      iv. Remove unnecessary FDs: none.

      v. Summarising left sides

$$\begin{aligned} A &\to B, C \\ D &\to E, F \\ A, D &\to \delta \end{aligned}$$

   (b) Create a relation for each left side from the attributes of the left side and the attributes that can be obtained with the left side (ohne δ):

$$\begin{aligned} R_1 &= \{A, B, C\} \\ R_2 &= \{D, E, F\} \\ R_3 &= \{A, D\} \end{aligned}$$

   (c) Does one of the generated relations contain the key? Yes, R_3: $\{A, D\}_F^+ = R$.

   (d) Is one of the relations completely contained in another relation? No. Finished.

Additional task: Comparison of synthesis and decomposition

Given the following relations S_1 and S_2 in the first normal form with functional dependencies:

$$\begin{aligned} S_1 &= (A,P,H,R,O,D,I,T,E)\ with \\ R &\rightarrow O,D \\ O &\rightarrow A,H,P,R\ and \\ & \{I,T,E,R\},\ \{I,T,E,O\} \text{ are keys.} \end{aligned}$$

$$\begin{aligned} S_2 &= (A,P,H,R,O,D,I,T,E)\ with \\ R &\rightarrow O \\ O &\rightarrow A,P,H,R,D\ and \\ & \{I,T,E,R\},\ \{I,T,E,O\} \text{ are keys.} \end{aligned}$$

- Through decomposition, split S_1 and S_2 into relations in the Boyce-Codd normal form (BCNF). Can this be done?

  Obviously, neither S_1 nor S_2 correspond with the BCNF because not all attributes depend on one key. That is not the case for H, for example, since this attribute only depends on a part of the key. Thus, the rules

  $$R \rightarrow O,D \in S_1 \qquad\qquad R \rightarrow O \in S_2$$

  violate the BCNF. After the decomposition process, S_1 and S_2 can be split into

  $$\begin{aligned} S_{1a} &= \{R\}^+_{FD} \\ &= \{R,O,D,A,H,P,R\} \\ S_{1b} &= S_1 - S_{1a} \cup \{R\} \\ &= \{I,T,E,R\} \end{aligned}$$

  $$\begin{aligned} S_{2a} &= \{O\}^+_{FD} \\ &= \{O,A,P,H,R,D\} \\ S_{2b} &= S_2 - S_{2a} \cup \{O\} \\ &= \{I,T,E,O\} \end{aligned}$$

  Decomposition is always lossless join decomposition. To be on the safe side, one more check:

  $$\{I,T,E,R\}^+_{FD} = \{A,P,H,R,O,D,I,T,E\} \qquad\qquad \{I,T,E,O\}^+_{FD} = \{A,P,H,R,O,D,I,T,E\}$$

  The dependency preservation is also ensured, because

  $$\begin{aligned} R \rightarrow O,D &\in FD_{S_1a} \\ O \rightarrow A,H,P,R &\in FD_{S_1a} \end{aligned} \qquad\qquad \begin{aligned} R \rightarrow O &\in FD_{S_2a} \\ O \rightarrow A,H,P,R,D &\in FD_{S_2a} \end{aligned}$$

  The decomposition also creates relations that are BCNF compliant, because in S_{1a} all attributes depend on the $\{R\}$ and $\{O\}$ keys, in S_{1b} all attributes depend on the $\{I,T,E,R\}$ key, in S_{2a} all attributes depend on the $\{R\}$ and $\{O\}$ keys, and in S_{2b} all attributes depend on the $\{I,T,E,O\}$ key.

- Decompose the relational schema using the synthesis process

  $$\begin{aligned} S_{1a} &= R,O,D \\ S_{1b} &= O,A,H,P,R \end{aligned} \qquad\qquad \begin{aligned} S_{2a} &= R,O \\ S_{2b} &= O,A,H,P,R,D \end{aligned}$$

  Is a key of the original S_1 and S_2 relations included in the attributes of the new relational schema? No, so therefore add another relation, e.g.:

  $$S_{1c} = I,T,E,O \qquad\qquad S_{2c} = I,T,E,R$$

- Carry out the simplified synthesis process.

  Add the rule $A, P, H, R, O, D, I, T, E \rightarrow \delta$ to the functional dependencies. This rule can be simplified, for example to

  $$I, T, E, O \rightarrow \delta \qquad\qquad I, T, E, R \rightarrow \delta$$

  This results in an additional relation respectively

  $$S_{1c} = I, T, E, O \qquad\qquad S_{2c} = I, T, E, R$$

- So now, which process is better: decomposition, synthesis or the simplified synthesis process? With the decomposition process, only two relations are created, which also correspond with the BCNF, so this is the better result. With the synthesis process, three relations are created, and for one it is even necessary to know the keys, which again results in a high runtime complexity. With the simplified synthesis procedure, the runtime complexity is lower, but three relations are also created.