Database Design
CS 6360.003: Homework #4

Due on Sunday November 6, 2016 at 11:59pm

Nurcan Yuruk

Hanlin He **(hxh160630)**
hanlin.he@utdallas.edu

Problem 1

Let

$$F_1 = \{A \to C, AC \to D, E \to AD, EC \to DH, DE \to CH\}$$
$$F_2 = \{A \to CD, E \to AH\}$$

For each attributes set on the left side of F_1, calculates its closure with respect to F_2:

$$\{A\}^+ = \{\text{A, C, D}\} \quad \text{including } C \text{ in } F_1$$
$$\{AC\}^+ = \{\text{A, C, D}\} \quad \text{including } D \text{ in } F_1$$
$$\{E\}^+ = \{\text{A, C, D, H}\} \quad \text{including } A, D \text{ in } F_1$$
$$\{EC\}^+ = \{\text{A, C, D, E, H}\} \quad \text{including } D, H \text{ in } F_1$$
$$\{DE\}^+ = \{\text{A, C, D, E, H}\} \quad \text{including } C, H \text{ in } F_1$$

Thus, F_2 covers F_1.
Respectively, consider the attributes sets on the left side of F_2:

$$\{A\}^+ = \{\text{A, C, D}\} \quad \text{including } C, D \text{ in } F_2$$
$$\{E\}^+ = \{\text{A, C, D, E, H}\} \quad \text{including } A, H \text{ in } F_2$$

Thus, F_1 covers F_2.
Hence, F_1 is equivalent to F_2.

Problem 2

Based on Algorithm 16.2(a), it is easy to see that

$$\{AD\}^+ = \{A, B, C, D, E, F, G, H, I, J\}$$
$$\{DG\}^+ = \{A, B, C, D, E, F, G, H, I, J\}$$

So $\{AD\}$ and $\{DG\}$ are candidate keys of relation R, $\{A, D, G\}$ is the prime attribute. Let $\{AD\}$ be the primary key.

$$\{A\}^+ = \{A, I\}$$
$$\{D\}^+ = \{D\}$$

Apparently, only one attribute is partially functional dependent on keys, which is I with functional dependency $A \to I$.

So the following relation would be in 2NF:

$$R_1 : \{\underline{A}, B, C, \underline{D}, E, F, G, H, J\}$$
$$R_2 : \{\underline{A}, I\}$$

R_2 is obviously in 3NF, consider the functional dependencies in R_1.

- $FD_1 : \{AB \to C\}$, AB is not a superkey of R_1, and C is not a prime attribute, thus violates the general definition of 3NF.
- $FD_2 : \{BD \to EF\}$, BD is not a superkey of R_1, and EF is not prime attributes, thus violates the general definition of 3NF.
- $FD_3 : \{AD \to GH\}$, AD is a superkey of R_1, thus does not violate the general definition of 3NF.
- $FD_4 : \{GD \to ABH\}$, GD is a superkey of R_1, thus does not violate the general definition of 3NF.
- $FD_5 : \{H \to J\}$, H is a non-prime attribute, and J is another non-prime attribute, thus violates the general definition of 3NF.

So we could decompose R_1 based on FD_1, FD_2 and FD_5 to achieve 3NF.

The final relations in 3NF are:

$$R_1 : \{\underline{A}, B, \underline{D}, G, H\}$$
$$R_2 : \{\underline{A}, I\}$$
$$R_3 : \{\underline{A}, \underline{B}, C\}$$
$$R_4 : \{\underline{B}, \underline{D}, E, F\}$$
$$R_5 : \{\underline{H}, J\}$$

Problem 3

Step 1 Consider the right side of the functional dependencies. The original functional dependencies turns into:

- $AB \to C$
- $AB \to D$
- $AB \to E$
- $C \to B$
- $C \to D$
- $CD \to E$
- $DE \to B$

Step 2 Consider the left side of the functional dependencies.

- $CD \to E$ can be replaced with $C \to E$ because $C \to D$.

Step 3 Consider the redundancy of the functional dependencies.

- $AB \to C$ and $C \to D$, so $AB \to D$ is redundant.
- $AB \to E$ and $C \to E$, so $AB \to E$ is redundant.
- $C \to D$, $C \to E$ and $DE \to B$, so $C \to B$ is redundant.

Step 4 Hence, the minimal cover would be:

$$\left\{ \begin{array}{rcl} AB & \to & C \\ C & \to & DE \\ DE & \to & B \end{array} \right\}$$

Problem 4

Part (a)

Step 1 Set $K = R = \{A, B, C, D, E, F, G, H, I, J\}$.

Step 2 Remove A from K, $K^+ = \{A, B, C, D, E, F, G, H, I, J\}$, since $F \to A$.
$K = \{B, C, D, E, F, G, H, I, J\}$ is still a superkey.

Step 3 Remove B from K, $K^+ = \{A, B, C, D, E, F, G, H, I, J\}$, since $H \to B$.
$K = \{C, D, E, F, G, H, I, J\}$ is still a superkey.

Step 4 Remove C from K, $K^+ = \{A, B, C, D, E, F, G, H, I, J\}$, since $FI \to C$.
$K = \{D, E, F, G, H, I, J\}$ is still a superkey.

Step 5 Remove D from K, $K^+ = \{A, B, C, D, E, F, G, H, I, J\}$, since $HI \to D$.
$K = \{E, F, G, H, I, J\}$ is still a superkey.

Step 6 Remove E from K, $K^+ = \{A, B, C, D, E, F, G, H, I, J\}$, since $FI \to E$.
$K = \{F, G, H, I, J\}$ is still a superkey.

Step 7 Remove F from K, $K^+ = \{A, B, C, D, E, F, G, H, I, J\}$, since $HI \to F$.
$K = \{G, H, I, J\}$ is still a superkey.

Step 8 Remove G from K, $K^+ = \{A, B, C, D, E, F, G, H, I, J\}$, since $HI \to G$.
$K = \{H, I, J\}$ is still a superkey.

Step 9 Remove H from K, $K^+ = \{I, J\}$.
$K = \{I, J\}$ is no longer a superkey, so H cannot be removed.

Step 10 Remove I from K, $K^+ = \{B, G, H, J\}$.
$K = \{H, J\}$ is no longer a superkey, so J cannot be removed.

Step 11 Remove J from K, $K^+ = \{A, B, C, D, E, F, G, H, I, J\}$, since $HI \to FI \to J$.
$K = \{H, I\}$ is a superkey, and cannot be divided any more.
Thus $\{H, I\}$ is a candidate key.

Step 12 Repeat above operations in different deletion order. It is easy to get that:

$$\{F, I\}^+ = \{A, B, C, D, E, F, G, H, I, J\}$$

$K = \{F, I\}$ is a superkey, and cannot be divided any more.
Thus $\{F, I\}$ is also a candidate key.

In conclusion, $\{F, H, I\}$ are the prime attributes of relation R.

Problem 4 continued on next page. . .

Part (b)

Based on the general definition of 3NF:

- $FD_1 : FI \to EHJC$ does not violate the 3NF general definition since FI is a superkey of relation R.
- $FD_2 : H \to GB$ violates the 3NF general definition since neither is H a superkey nor are GB prime attributes for relation R.
- $FD_3 : F \to EA$ violates the 3NF general definition since neither is F a superkey nor are EA prime attributes for relation R.
- $FD_4 : HI \to FGD$ does not violate the 3NF general definition since HI is a superkey of relation R.
- $FD_5 : A \to C$ violates the 3NF general definition since neither is A a superkey nor is C a prime attribute for relation R.

Thus, we can decompose the relation R directly into the following based on FD_2, FD_3 and FD_5:

$$
\begin{aligned}
&R_1 : \{D, \underline{F}, H, \underline{I}, J\} && \text{Remaining attribute in R} \\
&R_2 : \{B, G, \underline{H}\} && \text{Decomposition based on } FD_2 : H \to GB \\
&R_3 : \{A, E, \underline{F}\} && \text{Decomposition based on } FD_3 : F \to EA \\
&R_4 : \{\underline{A}, C\} && \text{Decomposition based on } FD_5 : A \to C
\end{aligned}
$$

Problem 5

Based on Algorithm 16.6, first calculate the minimal cover.

Step 1 Consider the right side of the functional dependencies. The original functional dependencies is already right side minimal:

- $FG \to E$
- $HI \to E$
- $F \to G$
- $FE \to H$
- $H \to I$

Step 2 Consider the left side of the functional dependencies.

- $FG \to E$ can be replaced with $F \to E$ because $F \to G$.
- $HI \to E$ can be replaced with $H \to E$ because $H \to I$.
- $FE \to H$ can be replaced with $F \to H$ because $F \to E$ (from previous conclusion).

So now the functional dependencies turn into:

- $F \to E$
- $H \to E$
- $F \to G$
- $F \to H$
- $H \to I$

Step 3 Consider the redundancy of the functional dependencies.

- $F \to H$ and $H \to E$, so $F \to E$ is redundant.

So now the functional dependencies turn into minimal cover:

$$\left\{ \begin{array}{ccc} H & \to & E \\ F & \to & G \\ F & \to & H \\ H & \to & I \end{array} \right\}$$

F is a candidate key of the relation R.

Then, for each left-hand-side X, form the relation:

$$R_1 : \{E, \underline{H}, I\}$$
$$R_2 : \{\underline{F}, G, H\}$$

The key F is contained in R_2 and there is no redundant relation. Hence the result is the 3NF relations with dependency preservation and non-additive join property.

Problem 6

```sql
CREATE OR REPLACE PROCEDURE
  UpdateSalary25 AS
CURSOR SalesEmployeeSSN IS
SELECT EMPLOYEE."SSN"
FROM EMPLOYEE, DEPARTMENT
WHERE EMPLOYEE."DNO" = DEPARTMENT."DNO"
    AND DEPARTMENT."DNAME" = 'Sales';
thisSSN EMPLOYEE."SSN"%TYPE;
BEGIN
    OPEN SalesEmployeeSSN;
    LOOP
        FETCH SalesEmployeeSSN INTO thisSSN;
        EXIT WHEN (SalesEmployeeSSN%NOTFOUND);
        UPDATE EMPLOYEE SET Salary = Salary * 1.25
        WHERE SSN = thisSSN;
    END LOOP;
    CLOSE SalesEmployeeSSN;
END;
/
BEGIN
  UPDATESALARY25();
END;
/
```

Problem 7

```sql
CREATE OR REPLACE PROCEDURE
  Check_Loans (BName IN LIBRARY_BRANCH."Branch_name"%TYPE) AS
CURSOR BOOK_INFO IS
SELECT BOOK."Title",
       BORROWER."Name",
       BORROWER."Phone",
       BOOK_LOANS."Date_out",
       BOOK_LOANS."Return_date"
FROM BOOK_LOANS, BOOK, BORROWER, LIBRARY_BRANCH
WHERE BOOK_LOANS."Book_id" = BOOK."Book_id"
  AND BOOK_LOANS."Card_no" = BORROWER."Card_no"
  AND BOOK_LOANS."Branch_id" = LIBRARY_BRANCH."Branch_id";
thisRow BOOK_INFO%ROWTYPE;
BEGIN
    OPEN BOOK_INFO;
    DBMS_OUTPUT.PUT_LINE(
        'List of books borrowed from '
        || BName
        || 'within 30 days which have not been returned:'
    );
    LOOP
        FETCH BOOK_INFO INTO thisRow;
        EXIT WHEN (BOOK_INFO%NOTFOUND);
        IF(to_date(thisRow."Date_out",'DD-MON-YY') > to_date(SYSDATE,'DD-MON-YY') - 30
            AND thisRow."Return_date" IS NULL) THEN
          DBMS_OUTPUT.PUT_LINE('Book Title: ' || thisRow."Title" );
          DBMS_OUTPUT.PUT_LINE('Borrower Name: ' ||thisRow."Name");
          DBMS_OUTPUT.PUT_LINE('Borrower Phone: ' ||thisRow."Phone");
          DBMS_OUTPUT.PUT_LINE('');
        END IF;
    END LOOP;
    CLOSE BOOK_INFO;
END;
/

-- Validation of the previous procedure
DECLARE
  in_string LIBRARY_BRANCH."Branch_name"%TYPE := 'Harrington';

BEGIN
  Check_Loans(in_string);  -- Procedure invocation
END;
/
```