

## ▼ Homework FD

- 이름: 이준용
- 학번: 201904458
- 제출일: 2022-05-13

Algorithm to compute  $X^+$  (the closure of  $X$ ) under  $F$ :

---

```

 $X^+ := X$ 
repeat
   $oldX^+ := X^+$ 
  for each functional dependency  $Y \rightarrow Z$  in  $F$  do
    if  $Y \subseteq X^+$  then  $X^+ := X^+ \cup Z$ 
until ( $oldX^+ = X^+$ )
  
```

---

Algorithm to check if  $F$  covers  $G$ :

```

for each FD :  $X \rightarrow Y$  in  $G$ 
   $Y$  must be in  $X^+$  under  $F$ 
  
```

---

## ▼ Problem 1 (20 pts).

20

- "Algorithm to compute  $X^+$  (the closure of  $X$ ) under  $F$ "을 구현하시오.
- 아래는 Python starter code다.
- 구현된 코드를 실행한다.

```

def xplus(X, F):
    X_plus = X
    while True:
        oldX_plus = X_plus
        for Y, Z in F:
            if Y <= X_plus:
                X_plus = X_plus | Z
        if oldX_plus == X_plus:
            break
    return X_plus
  
```

# RUN THIS CELL

# Schema

S = {'ssn', 'ename', 'pnumber', 'pname', 'plocation', 'hours'}

# 3 Functional dependencies: 1st part functionally determines the 2nd part

F = [{('ssn', 'ename')},  
 {('pnumber', 'plocation', 'pname')},  
 {('pnumber', 'ssn', 'hours')}]

```

print(xplus({'ssn'}, F))
print(xplus({'pnumber'}, F))
print(xplus({'ssn', 'pnumber'}, F))
  
```

```

{'ssn', 'ename'}
{'pnumber', 'pname', 'plocation'}
  
```

```
{'pname', 'plocation', 'hours', 'ssn', 'pnumber', 'ename'}
```

Ssn->ename [({'ssn'}, {'ename'})] 위의 내용으로 아래와 같이 정의하겠다

파이썬 코드로 9줄 맞는지 확인 set 이용

## ▼ Problem 2 (5 pts).

5

- Schema S, FD F에서 K가 Superkey인지를 검사하는 코드를 구현하시오.
- 아래는 Python starter code다.
- 구현된 코드를 실행한다.

```
def is_superkey(K, S, F):
    if xplus(K, F) == S:
        return True
    else:
        return False
# super key 이냐 구하는것 -> 1줄로 표현
```

```
# RUN THIS CELL
```

```
# Schema
```

```
S = {'ssn', 'ename', 'pnumber', 'pname', 'plocation', 'hours'}
```

```
# 3 Functional dependencies: 1st part functionally determines the 2nd part
```

```
F = [({'ssn'}, {'ename'}),
      ({'pnumber'}, {'plocation', 'pname'})],
      ({'pnumber', 'ssn'}, {'hours'})]
```

```
print(is_superkey({'ssn', 'pnumber'}, S, F))
print(is_superkey({'ssn'}, S, F))
print(is_superkey({'ssn', 'pnumber', 'ename'}, S, F))
```

```
True
False
True
```

## ▼ Problem 3 (15 pts).

15

- FD F가 G를 cover하는지를 검사하는 코드를 구현하시오.
- FD F와 G가 동등(equivalent)한지를 검사하는 코드를 구현하시오.
- 아래는 Python starter code다.
- 구현된 코드를 실행한다.

```
def covers(F, G):
    for X, Y in G:
        if Y.issubset(xplus(X, F)) != True:
            return False
    return True
# 4줄짜리 코드
# 위에 적혀있는 covers 알고리즘 보고 만들기 g가 f 커버
def equiv(F, G):
    if covers(F, G) == True and covers(G, F) == True:
        return True;
    else:
        return False
```

```
# YOUR CODE HERE      1줄짜리코드      f가 g 커버
```

```
# RUN THIS CELL
```

```
S = {'a', 'c', 'd', 'e', 'h'}
```

```
F = [({'a'}, {'c'}),
      ({'e'}, {'a','h'})
      ]
```

```
G = [({'a'}, {'c'}),
      ({'a','c'}, {'d'}),
      ({'e'}, {'a','d'}),
      ({'e'}, {'h'})
      ]
```

```
print(covers(F, G))
print(covers(G, F))
print(equiv(F, G))
```

```
False
True
False
```

```
# RUN THIS CELL
```

```
S = {'a', 'c', 'd', 'e', 'h'}
```

```
F = [({'a'}, {'c','d'}),
      ({'e'}, {'a','h'})
      ]
```

```
G = [({'a'}, {'c'}),
      ({'a','c'}, {'d'}),
      ({'e'}, {'a','d'}),
      ({'e'}, {'h'})
      ]
```

```
print(covers(F, G))
print(covers(G, F))
print(equiv(F, G))
```

```
True
True
True
```

## ▼ Problem 4 (5 pts).



Consider a relation  $R(A, B, C, D)$ , with FDs  $AB \rightarrow C$ ,  $BC \rightarrow D$ ,  $CD \rightarrow A$ . Find the closure of  $AB$  by hand.

AB 찾기+

Old\_AB<sup>+</sup> = AB<sup>+</sup> 현재 Old\_AB<sup>+</sup> = AB

AB  $\rightarrow$  C의 경우

AB  $\subseteq$  AB 따라서 AB<sup>+</sup> = AB  $\cup$  C

BC  $\rightarrow$  D

BC  $\subseteq$  ABC 따라서 AB<sup>+</sup> = ABC  $\cup$  D

CD  $\rightarrow$  A

CD  $\subseteq$  ABCD 따라서 AB<sup>+</sup> = ABCD  $\cup$  A

```

이제  $AB^+ = ABCD$ 
 $AB^+ == Old\_AB^+$ 인지 확인
 $ABCD \neq AB$  그것들은 같지 않습니다. 따라서 다음 라운드로 이동

 $Old\_AB^+ =$  현재  $Old\_AB^+ = ABCD$ 인  $AB^+$ 
 $AB \rightarrow C$ 의 경우
 $AB \subseteq ABCD$  따라서  $AB^+ = ABCD \cup C$ 
 $BC \rightarrow D$ 
 $BC \subseteq ABCD$  따라서  $AB^+ = ABCD \cup D$ 
 $CD \rightarrow A$ 
 $CD \subseteq ABCD$  따라서  $AB^+ = ABCD \cup A$ 
이제  $AB^+ = ABCD$ 
 $AB^+ == Old\_AB^+$ 인지 확인
 $AB^+ == Old\_AB^+$ ,  $ABCD == ABCD$ 
따라서  $AB^+ = ABCD$ 

```

(You may paste an image here. CTRL+V)

## ▼ Problem 5 (10 pts). 10

Consider relation  $R(A,B,C,D,E)$  with the following functional dependencies:  $AB \rightarrow C$ ,  $D \rightarrow E$ ,  $DE \rightarrow B$ . Justify your answer!!!

- Is R in 2NF?
- Is R in 3NF?
- Is R in BCNF?

WRITE HERE (To edit, double click this cell)

**\*\*Is R in 2NF?\*\***

$R\{A, B, C, D, E\}$

$AB \rightarrow C, D \rightarrow E, DE \rightarrow B$

속성 조합: AD

속성 폐쇄 계산

$AD^+ = ABCDE$ , AD는 후보 키입니다.

2NF의 정의: 비 프라임 속성은 후보 키에 부분적으로 종속되어서는 안 됩니다.

R에는 5개의 속성이 있으므로 - A, B, C, D E 및 후보 키는 AD이므로 프라임 속성(후보 키의 일부)은 A 및 D이고 비 프라임

a), b), c)가 완전한 함수적 종속성인지 찾아야 함.

A의 closure는  $\{A\}$ , D의 closure는  $\{D, E, B\}$

a) FD:  $AD \rightarrow C$ 는 2NF의 정의를 충족합니다.

후보키 AD의 일부에 부분적으로 종속되지 않습니다.

b) FD:  $AD \rightarrow E$ 는 2NF의 정의를 충족하지 않으며,

후보키 AD의 일부에 부분적으로 종속됩니다.

그러므로 2NF가 아닙니다.

c) FD:  $AD \rightarrow B$ 는 2NF의 정의를 충족하지 않으며, 후보키 AD의 일부에 부분적으로 종속됩니다.

2NF에 있는 분해된 테이블

$AD \rightarrow C$ 에서 D때문에 2NF가 될 수 없으므로 D의 closure를 구해서 쪼갬다.

$R_1(D, E, B)$

$R_2(A, C)$

**\*\*Is R in 3NF?\*\***

-----3가지 조건-----

관계 스키마 R은 FD의 집합 F에 대한 제3정규형(3NF)이다.

F의 각 FD  $X \rightarrow Y$ 에 대해  $X \subseteq R$  및  $Y \subseteq R$ 인 경우에만

다음 조건 중 하나 이상이 유지됩니다.

조건1)  $X \rightarrow Y$  is a trivial FD(즉,  $Y \subseteq X$ )

조건2) X는 R의 슈퍼키입니다.

조건3) Every element of  $Y - X$  is a prime attribute (즉, 일부 후보키에 포함됨) of R

trivial하지 않고 lhs에 슈퍼키가 없는 FD를 S에 저장합니다.

$AB \rightarrow C$ 는 조건 1과 2를 위반

따라서 S에 추가

$D \rightarrow E$ 는 조건 1과 2를 위반

따라서 S에 추가

$DE \rightarrow B$ 는 조건 1과 2를 위반

따라서 S에 추가

$S = \{ 'AB \rightarrow C', 'D \rightarrow E', 'DE \rightarrow B' \}$

S는 비어 있지 않습니다

모든 주요 속성 결정

후보 키는 AD입니다.

prime attribute은 A,D입니다.

3NF의 조건 3(C - AB의 모든 요소가 주요 속성임)과 관련하여 S에서  $AB \rightarrow C$ 를 확인합니다.

조건 3 위반: C는 주요 속성이 아닙니다.

3NF의 Condition 3(E - D의 모든 요소가 주요 속성임)과 관련하여 S에서  $D \rightarrow E$ 를 확인합니다.

조건 3 위반: E는 주요 속성이 아닙니다.

3NF의 조건 3(B의 모든 요소 - DE가 주요 속성임)과 관련하여 S에서  $DE \rightarrow B$ 를 확인합니다.

조건 3 위반: B는 주요 속성이 아닙니다.

R은 3NF에 없습니다.

**\*\*Is R in BCNF?\*\***

BCNF에 대한 두 가지 조건 위반: FD  $\Rightarrow AB \rightarrow C$ 는 trivial이 아니며 왼쪽에 SK가 없습니다.

BCNF에 대한 두 가지 조건 위반: FD  $\Rightarrow D \rightarrow E$ 는 trivial이 아니며 왼쪽에 SK가 없습니다.

BCNF에 대한 두 가지 조건 위반: FD  $\Rightarrow DE \rightarrow B$ 는 trivial하지 않으며 왼쪽에 SK가 없습니다.

그러므로, 주어진 릴레이션은 BCNF가 아닙니다.

(You may paste an image here. CTRL+V)

## ▼ Problem 6 (10 pts).

10

Consider the following set F of functional dependencies for relation schema  $R = \{A, B, C, D, E\}$ .

$\{A \rightarrow BC, CD \rightarrow E, B \rightarrow D, E \rightarrow A\}$

- List all the candidate keys for R.

- You must show all the procedure to list candidate keys.

WRITE HERE (To edit, double click this cell)

attributes:

A,B,C,D,E

속성 합집합 :

속성 폐쇄 계산

FD 양쪽의 속성 : A,B,C,D,E

위의 속성들과 폐쇄를 계산합니다 .

$A^+ = ABCDE = R(\text{후보 키})$

$B^+ = BD \subset R$

$C^+ = C \subset R$

$D^+ = D \subset R$

$E^+ = ABCDE = R(\text{후보 키})$

하나 이상의 속성이 있는 모든 속성 조합

['A', 'E']를 포함하는 항목은 더 이상 고려할 필요가 없다고 생각합니다 .

SK(superkey)로만 연결되기 때문이라고 생각합니다 . .

['A', 'E']를 포함하지 않는 속성 조합이 필요합니다 .

$BC^+ = ABCDE = R(ck)$

$BD^+ = BD \subset R$

$CD^+ = ABCDE = R(ck)$

다른 속성을 추가하면 슈퍼키가 생성됩니다 .

따라서 ['A', 'E', 'BC', 'CD']는 (유일한) 후보 키입니다 .

그러므로 A, E, BC, CD

(You may paste an image here. CTRL+V)

## What to submit

- Run **all cells** after restarting the kernel
- Goto "File -> Print Preview" in Jupyter notebook, print the page as pdf
- (If you use Colab, "File -> Print" in Google colab)
- Pdf file name must be in a form of: homework\_fd\_홍길동\_202200001.pdf
- Submit the pdf file in **google classroom**
- No late homeworks will be accepted

✓ 0초    오후 11:42에 완료됨

● ✕