CSC343 2022W
Assignment 3

Litao (John) Zhou (1006013092):

Shaoheng Wang (1003945181)

Due: 3th Apr 2022

1. Relation *Reservation* is meant to keep track of which skipper reserves which boat model, on which date, from which dock, but its design has some redundancy:

$$Reservation(sID, age, length, sName, dID, day, rating, mID)$$

...where sID identifies the skipper, $sName$ is the skipper's name, whereas $rating$ and age record the skipper's skill (a number between 0 and 5, inclusive) and age in years (a number greater than 0). The reserved model is identified by mID, its $length$ is in feet, day represents the entire day the craft is reserved for, and dID identifies the dock the craft will be anchored at waiting for the skipper. The following dependencies hold:[1]

$$S = \{sID \to sName, rating, age; \quad mID \to length; \quad day, sID \to mID, dID\}$$

(a) Give one example of a redundancy that relation Reservation, combined with FDs S, allow.

sID	age	length	sName	dID	day	rating	mID
1.	18	100	A	10	2	1	5
1	18	120	A	9	3	1	7

(b) Design a schema in DDL called reservation.ddl that represents the same information as Reservation, using exactly the same attribute names, but has the following goals, in descending order of importance:
i. has as few redundancies as possible;
ii. allows as few NULL or DEFAULT values as possible;
iii. enforces as many constraints from the description above as possible, without using triggers or assertions.
Your schema should import into psql without error using the command:
\i reservation.ddl
While you are developing your schema you may want to ensure that your previous version is removed before you read in a new one:
drop schema if exists reservation cascade;
create schema reservation;
set search_path to reservation;
Use comments at the beginning of reservation.ddl to explain which constraints were not enforced (if any) and which redundancies are still allowed (if any). As the designer you have freedom to choose datatypes for the various attributes.

See answer.ddl or as follows

```
-- schema for skippers
-- Our schema satisfies all FDs in S and resulted in no redundancies
-- User have the option to join Skipper and Model to retrieve all attributes in original Reservation relation
DROP schema if exists reservation cascade;
CREATE schema reservation;
SET search_path to reservation;

CREATE TABLE Skipper (
  sID INTEGER PRIMARY KEY,
  sName VARCHAR(50) NOT NULL,
  rating INTEGER NOT NULL
  check(rating in (0,1,2,3,4,5)),
  age INTEGER NOT NULL
  check (age>=0),
  day INTEGER NOT NULL
);

CREATE TABLE Model (
  sID INTEGER NOT NULL,
  day INTEGER NOT NULL,
  mID INTEGER NOT NULL,
  dID INTEGER NOT NULL,
  length INTEGER NOT NULL,
  PRIMARY KEY(sID,day)
);
```

2. Relation F has attributes $KLMNOPQRS$ and functional dependencies G:

$$G = \{P \to LM, KOQ \to PS, L \to N, Q \to RS\}$$

(a) Which FDs in G violate BCNF? List them.

Left Hand Side of FDs	Closure	Result
P	PLMN	Violate BCNF
KOQ	KOQPSLMNR	KOQ is a key
L	LN	Violate BCNF
Q	QRS	Violate BCNF

FDs that violate BCNF: P → LM, L → N, Q → RS.

(b) Use the BCNF decomposition method to derive a redundancy-preventing, lossless, decomposition of F into a new schema consisting of relations that are in BCNF. Be sure to project the FDs from G onto the relations in your final schema. There may be more than one correct answer possible, since there are choices possible at steps in the decomposition. List your final relations alphabetically, and order the attributes within each relation alphabetically (this avoids combinatorial explosion of the number of alternatives, we have to check).

1. Decompose R using FD P → LM. P^+ = LMNP, so this yields two relations: R1 = LMNP, R2 = KOPQRS.
2. Project the FDs onto R1 = LMNP.

L	M	N	P	Closure	FDs
✓				LN	L → N violates BCNF; abort the projection
	✓			M	nothing
		✓		N	nothing
			✓	LMNP	P → LMN

We must decompose R1 further.

3. Decompose R1 using FD L → N. L^+ = LN, so this yields two relations: R3 = LN, R4 = LMP.
4. Project the FDs onto R3 = LN.

L	N	Closure	FDs
✓		LN	L → N; L is a super key of R3
	✓	N	nothing
Superset of L		irrelevant	can only generate weaker FDs than what we already have

This relation satisfies BCNF.

5. Project the FDs onto R4 = LMP.

L	M	P	Closure	FDs
✓			L	nothing
	✓		M	nothing
		✓	LMP	P → LM; P is a superkey of R4

Superset of P			irrelevant	can only generate weaker FDs than what we already have
✓	✓		LM	nothing

This relation satisfies BCNF.

6. Project the FDs onto R2 = KOPQRS.

K	O	P	Q	R	S	Closure	FDs
✓						K	nothing
	✓					O	nothing
		✓				P	nothing
			✓			QRS	Q → RS violates BCNF; abort the projection

We must decompose R2 further.

7. Decompose R2 using FD Q → RS. Q^+ = QRS, so this yields two relations: R5 = QRS, R6 = KOPQ.
8. Project the FDs onto R5 = QRS.

Q	R	S	Closure	FDs
✓			QRS	Q → RS; Q is a superkey of R5
	✓		R	nothing
		✓	S	nothing
Superset of Q			irrelevant	can only generate weaker FDs than what we already have
	✓	✓	RS	nothing

This relation satisfies BCNF.

9. Project the FDs onto R6 = KOPQ.

K	O	P	Q	Closure	FDs
✓				K	nothing

	✓			O	nothing
		✓		P	nothing
			✓	Q	nothing
✓	✓			KO	nothing
✓		✓		KP	nothing
✓			✓	KQ	nothing
	✓	✓		OP	nothing
	✓		✓	OQ	nothing
		✓	✓	PQ	nothing
✓	✓	✓		KOP	nothing
✓		✓	✓	KPQ	nothing
✓	✓		✓	KOPQ	KOQ → P; KOQ is a superkey of R6
	✓	✓	✓	OPQ	nothing
Superset of KOPQ				Irrelevant	can only generate weaker FDs than what we already have

This relation satisfies BCNF.

10. Final Decomposition
    a. R3 = LN with FD L → N
    b. R4 = LMP with FD P → LM
    c. R5 = QRS with FD Q → RS
    d. R6 = KOPQ with FD KOQ → P

(c) Does your final schema preserve dependencies? Explain why you answer yes or no.

Yes, if we simplify the original FDs:

Since $Q \rightarrow RS$ so $KOQ \rightarrow PS$ can be simplified to $KOQ \rightarrow P$. The original FDs will match exactly with the FDs after Decomposition.

1. $L \rightarrow N$
2. $P \rightarrow LM$
3. $Q \rightarrow RS$
4. $KOQ \rightarrow P$

(d) BCNF guarantees a lossless join. However, demonstrate this to a possibly skeptical observer using the Chase Test.

Assume (k,l,m,n,o,p,q,r,s) is a tuple in R3 ⋈ R4 ⋈ R5 ⋈ R6. Note: in the following table, blank space indicates arbitrary values.

K	L	M	N	O	P	Q	R	S
	l		n					
	l	m			p			
						q	r	s
k				o	p	q		

Consider P → LM in original F

K	L	M	N	O	P	Q	R	S
	l		n					
	l	m			p			
						q	r	s
k	l	m		o	p	q		

Consider KOQ → PS in original F

K	L	M	N	O	P	Q	R	S
	l		n					
	l	m			p			
						q	r	s
k	l	m		o	p	q		

Consider L → N in original F

K	L	M	N	O	P	Q	R	S
	l		n					
	l	m			p			

						q	r	s
k	l	m	n	o	p	q		

Consider Q → RS in original F

K	L	M	N	O	P	Q	R	S
	l		n					
	l	m			p			
						q	r	s
k	l	m	n	o	p	q	r	s

So if (k,l,m,n,o,p,q,r,s) is a tuple in R3 ⋈ R4 ⋈ R5 ⋈ R6, then (k,l,m,n,o,p,q,r,s) must be a tuple in original F. Our decomposition passes the chase test

3. Relation R has attributes $ABCDEFGH$ and functional dependencies S:

$$S = \{ADE \to B, B \to CF, CD \to AF, BF \to AD, AB \to H\}$$

(a) Find a minimal basis for S. Your final answer must put the FDs in ascending alphabetical order, and the attributes within the LHS and RHS of each FD into alphabetical order.

1. **Split the RHS of each FD and call it S1**
   (a) ADE→B
   (b) B→C
   (c) B→F
   (d) CD→A
   (e) CD→F
   (f) BF→A
   (g) BF→D
   (h) AB→H
2. **Reduce the LHS of each FD in S1**
   (a) $A^+ = A$
   $D^+ = D$
   $E^+ = E$
   $AD^+ = AD$
   $AE^+ = AE$
   $DE^+ = DE$
   Thus, there is no way to get B without this FD
   (b) Nothing to reduce as singleton
   (c) Nothing to reduce as singleton
   (d) There is no way to get A without this FD
   (e) There is no way to get F without this FD
   (f) $B^+ = ABCDFH$
   Thus, this FD could reduce to B→A
   (g) Same as above(f), this FD could reduce to B→D
   (h) Same as above(f), this FD could reduce to B→H

   The current remaining FDs are as follows as S2:

   (a) ADE→B
   (b) B→C
   (c) B→F
   (d) CD→A
   (e) CD→F
   (f) B→A
   (g) B→D
   (h) B→H

3. Try to reduce the RHS of the FDs

(a) $ADE^{+}_{\{S2\text{-}(a)\}}$ = ADE We could not reduce this FD
(b) $B^{+}_{\{S2\text{-}(b)\}}$ = ABDFH We could not reduce this FD
(c) $B^{+}_{\{S2\text{-}(c)\}}$ = ABCDFH We could discard this FD (c)
(d) $CD^{+}_{\{S2\text{-}(c)(d)\}}$ = CDF We could not reduce this FD
(e) $CD^{+}_{\{S2\text{-}(c)(e)\}}$ = ACD We could not reduce this FD
(f) $B^{+}_{\{S2\text{-}(c)(f)\}}$ = ABCDF We could discard this FD (f)
(g) $B^{+}_{\{S2\text{-}(c)(f)(g)\}}$ = BC We could not reduce this FD
(h) $B^{+}_{\{S2\text{-}(c)(f)(h)\}}$ = ABCDF We could not reduce this FD

4. The final minimal basis for the FD as follows:

(a) ADE→B
(b) B→C
(c) CD→A
(d) CD→F
(e) B→D
(f) B→H

(b) Find all the keys for R using your solution for a minimal basis.

The minimal basis we found in the previous part are as follows:

(a) ADE→B
(b) B→C
(c) CD→A
(d) CD→F
(e) B→D
(f) B→H

It could observe that H, F only exists on the RHS of all FDs and E only exist on the LHS of all FDs,

G does not exist in any FDs

Thus, E, G must be in every key and H, F could not be in any key

This leaves A, B, C, D

AEG^+= AEG **X**

BEG^+= ABCDEFH ✓

CEG^+=CEG **X**

DEG^+=DEG **X**

$ACEG^+$=ACEG **X**

$ADEG^+$= ABCDEFH ✓

$CDEG^+$= ABCDEFH ✓

The keys are as follows: ADEG, BEG, CDEG

(c) Use the 3NF synthesis algorithm to find a lossless, dependency-preserving decomposition of relation R into a new schema consisting of relations that are in 3NF. Your final answer should combine FDs with the same LHS to create a single relation. If your schema has a relation that is a subset of another, keep only the larger relation.

The minimal basis we found in the previous part (a) are as follows:

(a) ADE→B
(b) B→C
(c) CD→A
(d) CD→F
(e) B→D
(f) B→H

We could combine the FD as follows:

(a) ADE→B
(b) B→CDH
(c) CD→AF

We use the 3NF synthesis algorithm and form the following relation:

(a) ADEB
(b) BCDH
(c) CDAF

As G does not exist in any of the FD, non of the key is the super key and we must manually add one super key BEG as we compute in the previous part (b) as a key

Thus, the final relations are:

(a) R1(A, D, E, B)
(b) R2(B, C, D, H)
(c) R3(C, D, A, F)
(d) R5(B, E, G)

(d) Does your solution allow redundancy? Explain how (with an example), or why not.

Yes, our schemas allow the redundancy, as the 3NF does not guarantee the redundancy.

R4(B, E, G) violates BCNF as B^+=ABCDFH and thus, B is not a superkey. This allow the existence of redundancy.