csc343 winter 2021
assignment #3: database (re)design
due April 2nd, 4 p.m.

goals

This assignment aims to help you learn to:

- design a good schema
- understand violations of functional dependencies
- create a minimal basis for a set of functional dependencies
- project a set of functional dependencies onto a set of attributes
- find all the keys for a set of functional dependencies
- re-factor relation(s) into BCNF
- re-factor relations into 3NF

Your assignment must be typed to produce a PDF document **a3.pdf**, and a plain text document **reservation.ddl** (hand-written submissions are not acceptable). You may work on the assignment in groups of 1 or 2, and submit a single assignment for the entire group on MarkUs. You must establish your group well before the due date by submitting an incomplete, or even empty, submission (of course, we only grade the last submission before the due date/time).

exercises

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

   $$Reservation(sID, age, length, sName, day, cName, rating, cID)$$

   ... 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 (a number greater than 0). The reserved craft is identified by cID, its name is $cName$, $length$ is in feet, and the date and time the craft is reserved for by the skipper is day. The following dependencies hold:[1]

   $$S = \{sID \rightarrow sName, rating, age;\ \ cID \rightarrow cName, length\}$$

[1] Since multiple attributes may be separated by commas, we use semicolons to separate FDs.

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

sID	age	sName	length	day	cName	rating	cID
01	30	Thomas	80	'2021-01-01'	Bird	4	1
01	30	Thomas	80	'2021-01-02'	Bird	4	1

Then here we have redundancy of sID, sName, rating, cID, cName , length and age.

(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.

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

$$G = \{KOQ \to PS, L \to KN, KQ \to RS\}$$

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

For FD $KOQ \to PS$, $KOQ^+ = \{K, O, Q, P, S, R\}$
For FD $L \to KN$, $L^+ = \{L, K, N\}$
For FD $KQ \to RS$, $KQ^+ = \{K, Q, R, S\}$
So FDs $\{KOQ \to PS, L \to KN, KQ \to RS\}$ violate BCNF.

(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).

We choose FD $KOQ \to PS$, since it violates BCNF.
We first compute $KOQ^+ = \{K, O, Q, P, S, R\}$
Then we replace relation F with two new schemas $R_1 = KOQ^+ = \{K, O, Q, P, S, R\}$ and $R_2 = R - (KOQ^+ - KOQ) = \{K, L, M, N, O, Q\}$

Then we project the FD's G onto R_1 and R_2
For $R_1 = \{K, O, Q, P, S, R\}$, the FDs that project onto it are: $\{KQ \rightarrow RS, KOQ \rightarrow PSR\}$
For $R_2 = \{K, L, M, N, O, Q\}$, the FDs that project onto it are: $\{L \rightarrow KN\}$

Since FD $KQ \rightarrow RS$ in R_1 violates BCNF, then we recursively decompose R_1
We compute $KQ^+ = \{K, Q, R, S\}$
Then replace relation R_1 with two new schemas $R_{11} = KQ^+ = \{K, Q, R, S\}$ and $R_{12} = R_1 - (KQ^+ - KQ) = \{K, O, Q, P\}$
Then we project the FD onto R_{11} and R_{12}
For $R_{11} = \{K, Q, R, S\}$, the FD that projects onto it is: $\{KQ \rightarrow RS\}$
Since KQ is a superkey, so R_{11} is in DCNF.
For $R_{12} = \{K, O, P, Q\}$, the FD that projects onto it is: $\{KOQ \rightarrow P\}$
Since KOQ is a superkey, so R_{12} is in DCNF.

Since FD $L \rightarrow KN$ in R_2 violates BCNF, then we recursively decompose R_2
We compute $L^+ = \{L, K, N\}$
Then replace relation R_2 with two new schemas $R_{21} = L^+ = \{L, K, N\}$ and $R_{22} = R_2 - (L^+ - L) = \{L, M, O, Q\}$
Then we project the FD onto R_{21} and R_{22}
For $R_{21} = \{K, L, N\}$, the FD that projects onto it is: $\{L \rightarrow KN\}$
Since L is a superkey, so R_{21} is in DCNF.
For $R_{22} = \{L, M, O, Q\}$, the FD that projects onto it is: $\{LMOQ \rightarrow LMOQ\}$
Since $LMOQ$ is a trivial FD, so R_{22} is in DCNF.

So the BCNF decomposition of F is $\{KQRS,\ KOPQ,\ KLN,\ LMOQ\}$
For $KQRS$, its projected FD is $KQ \rightarrow RS$
For $KOPQ$, its projected FD is $KOQ \rightarrow P$
For KLN, its projected FD is $L \rightarrow KN$
For $LMOQ$, its projected FD is $LMOQ \rightarrow LMOQ$, which is a trivial FD.

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

Yes, it preserves dependencies. We still have FDs of $\{KQ \rightarrow RS,\ L \rightarrow KN\}$, but misses FD $\{KOQ \rightarrow PS\}$. However, we can derive $KOQ \rightarrow PS$ by finding the dependency closure of KOQ in decomposed relations, which is $KOQ^+ = \{K, O, Q, P, R, S\}$, then we derive $KOQ \rightarrow PS$ from it.

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

We start by assuming unsubscripted symbols$t = < k, l, m, n, o, p, q, r, s >$
Then we create tuples based on FDs

K	L	M	N	O	P	Q	R	S
k	l1	m1	n1	o1	p1	q	r	s
k	l2	m2	n2	o	p	q	r2	s2
k	l	m3	n	o3	p3	q3	r3	s3
k4	l	m	n4	o	p4	q	r4	s4

Since two rows (third row and fourth row) agree in the left side of $L \rightarrow KN$, then we make their right sides agree too in the tuples

K	L	M	N	O	P	Q	R	S
k	l1	m1	n1	o1	p1	q	r	s
k	l2	m2	n2	o	p	q	r2	s2
k	l	m3	n	o3	p3	q3	r3	s3
k	l	m	n	o	p4	q	r4	s4

Since more than two rows (first row, second row and fourth row) agree in the left side of $KQ \rightarrow RS$, then we make their right sides agree too in the tuples

K	L	M	N	O	P	Q	R	S
k	l1	m1	n1	o1	p1	q	r	s
k	l2	m2	n2	o	p	q	r	s
k	l	m3	n	o3	p3	q3	r3	s3
k	l	m	n	o	p4	q	r	s

Since two rows (second row and fourth row) agree in the left side of $KOQ \rightarrow PS$, then we make their right sides agree too in the tuples

K	L	M	N	O	P	Q	R	S
k	l1	m1	n1	o1	p1	q	r	s
k	l2	m2	n2	o	p	q	r	s
k	l	m3	n	o3	p3	q3	r3	s3
k	l	m	n	o	p	q	r	s

Here, we get the fourth row completely an unsubscripted row. Then we know any tuple in the project-rejoin is in the original. Thus, we prove that BCNF guarantees a lossless join.

Show us the steps in your work. This allows us to assign part marks, if needed, and to be sure that you have not taken any unwarranted short cuts.

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

$$S = \{ACDE \rightarrow B, B \rightarrow CF, CD \rightarrow AF, BCF \rightarrow AD, ABF \rightarrow H\}$$

(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.

Step 1, split the RHS of each FD:

i. $ACDE \rightarrow B$
ii. $B \rightarrow C$
iii. $B \rightarrow F$

iv. $CD \rightarrow A$

v. $CD \rightarrow F$

vi. $BCF \rightarrow A$

vii. $BCF \rightarrow D$

viii. $ABF \rightarrow H$

Step 2, for each FD, (i) through (viii), reduce LHS by finding closure of each LHS:

i. $A^+ = A$
$C^+ = C$
$D^+ = D$
$E^+ = E$
$AC^+ = AC$
$AD^+ = AD$
$AE^+ = AE$
$CD^+ = CDAF$
$CE^+ = CE$
$DE^+ = DE$
$ACD^+ = ACDF$
$ACE^+ = ACE$
$ADE^+ = ADE$
$CDE^+ = CDEAFBH$
So we can get $CDE \rightarrow B$

ii. $B \rightarrow C$

iii. $B \rightarrow F$

iv. $C^+ = C$
$D^+ = D$
Cannot get A for $CD \rightarrow A$ by reducing LHS

v. $C^+ = C$
$D^+ = D$
Cannot get F for $CD \rightarrow F$ by reducing LHS

vi. $B^+ = BCFADH$
So we get $B \rightarrow A$

vii. $B^+ = BCFADH$
So we get $B \rightarrow D$

viii. $A^+ = A$
$B^+ = BCFADH$
So we get $B \rightarrow H$

So now we have:

i. $CDE \rightarrow B$

ii. $B \rightarrow C$

iii. $B \rightarrow F$

iv. $CD \rightarrow A$

v. $CD \rightarrow F$

vi. $B \rightarrow A$

vii. $B \rightarrow D$,

viii. $B \to H$

Step 3, remove redundant FDs from (i) through (viii):

i. CDE^+ without (i) $= CDEAF$, since we cannot get B so keep (i)
ii. B^+ without (ii) $= BFADH$, since we cannot get C so keep (ii)
iii. B^+ without (iii) $= BCADHF$, since we can get F, so remove (iii)
iv. CD^+ without (iv) and (iii) $= CDF$, since we cannot get A, so keep (iv)
v. CD^+ without (v) and (iii) $= CDA$, since we cannot get F, so keep (v)
vi. B^+ without (vi) and (iii) $= BCDHAF$, since we can get A, so remove (vi)
vii. B^+ without (vii), (vi) and (iii) $= BCH$, since we cannot get D, so keep (vii)
viii. B^+ without (viii), (vi) and (iii) $= BCDAF$, since we cannot get H, so keep (viii)

So the minimal basis for S is: $\{B \to C, B \to D, B \to H, CD \to A, CD \to F, CDE \to B\}$

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

Since G doesn't appear on both LHS and RHS, then it must be in every key.
Since E only appears on the LHS, then it must be in every key.
Since F, H only appear on the RHS, then they are not in any key.
Since A, B, C, D appear on both LHS and RHS, then we have to check them.
Then we only have to consider all combinations of A, B, C, D with adding E, G

i. $EGA^+ = EGA$, so it is not a key.
ii. $EGB^+ = ABCDEFGH$, so it is a key.
iii. $EGC^+ = EGC$, so it is not a key.
iv. $EGD^+ = EGD$, so it is not a key.
v. We don't have to consider any possibilities include EGB since it's already been a key.
vi. $EGAC^+ = EGAC$, so it is not a key.
vii. $EGAD^+ = EGAD$, so it is not a key.
viii. $EGCD^+ = ABCDEFGH$, so it is a key.
ix. We don't have to consider any possibilities include $EGCD$ since it's already been a key.

Thus, the keys for R using my solution for a minimal basis are $\{EGB, EGCD\}$

(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.

Since we have construct a minimal basis $\{CDE \to B, B \to C, B \to D, B \to H, CD \to A, CD \to F\}$ for relation R
We first combine FDs with the same LHS.

i. $CDE \to B$
ii. $B \to CDH$
iii. $CD \to AF$

Then for each FD $X \to Y$, we create a single relation with schema $X \cup Y$

i. $CDE \rightarrow B \Rightarrow R_1(BCDE)$
ii. $B \rightarrow CDH \Rightarrow R_2(BCDH)$
iii. $CD \rightarrow AF \Rightarrow R_3(ACDF)$

Since no relation is a superkey for all attributes, then we add a relation with the schema of a key $R_4(EGB)$
So the relations that are in 3NF are $R_1(BCDE)$, $R_2(BCDH)$, $R_3(ACDF)$, $R_4(EGB)$

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

Yes, my solution allows redundancy.
Since the BCNF decomposition guarantees no redundancy, if there are FDs violate BCNF in a relation (for a FD $X \rightarrow Y$, X is not a superkey in a relation), then the redundancy is allowed in that relation.
For example, the FD $B \rightarrow CDA$ will project onto relation R_1. And $B^+ = BCDHAF$, but it doesn't include E in R_1, so B is not a superkey for this relation. Therefore, the relations in 3NF allow redundancy.

Show us the steps in your work. This allows us to assign part marks, if needed, and to be sure that you have not taken any unwarranted short cuts.

submissions

Submit **a3.pdf** and **reservation.ddl** on MarkUs. One submission per group, whether a group is one or two people. You declare a group by submitting an empty, or partial, file, and this should be done **well before** the due date. You may always replace such a file with a better version, until the due date.

Double check that you have submitted the correct version of your file by downloading it from MarkUs.