Βάσεις Δεδομένων
Σειρά Ασκήσεων

Ομάδα Μ

Νικήτας Τσίννας, el18187
Αναστάσιος Παπαζαφειρόπουλος, el18079
Νικόλαος Παγώνας, el18175

Exercise 1

Exercise 2

Q_1 :

$$\Pi_{\text{pid}}(\sigma_{\text{companyname="Google"}}(\text{Person} \bowtie \text{Company})) \bigcap$$

$$\Pi_{\text{pid}}\Big(\sigma_{\text{sharenum}>500}(\sigma_{\text{companyname="Facebook"}}(\text{Shares} \bowtie \text{Company}))\Big)$$

Q_2 :

$$\Pi_{\text{Person.pid}}(\sigma_{\text{Person.cid = Shares.cid} \wedge \text{Person.managerid = Shares.pid} \wedge \text{Sharesnum}>0}(\text{Person} \times \text{Shares}))$$

Q_3 :

$$\Pi_{\text{pid}}\Big({}_{\text{pid}}\mathcal{G}_{(\text{count(cid)}>2)}(\sigma_{\text{sharesnum}>0}(\text{Shares}))\Big)$$

Q_4 :

$$\Pi_{\text{pid,cid}}(\sigma_{\text{sharesnum}>0}(Shares)) \Big/ \Pi_{\text{cid}}(Company)$$

Exercise 3

Exercise 4

A.

Find all attributes that have not appeared on the RHS of any FD: $S_1 = \{B, C\}$
Find all attributes that have not appeared on the LHS of any FD: $S_2 = \{A\}$

Compute the closure set of S_1: $S_1^+ = \{A, B, C, D, E\}$.

$S_1^+ = R$, so $\{B, C\}$ is the only candidate key.

B.

Computing a canonical cover for F:

Initially, $F_c = F$.

Iteration #1:

Can't use union rule.

Checking $B \to EA$:

Removing B:

$\varnothing^+ = \varnothing$, doesn't contain B, so B is needed.

Removing E:
$F'_c = \{B \to A, EBC \to D, BED \to A\}$
$(B)^+ = (AB)$, doesn't contain E, so E is needed.

Removing A:
$F'_c = \{B \to E, EBC \to D, BED \to A\}$
$(B)^+ = (BE)$, doesn't contain A, so A is needed.

Checking $EBC \to D$:

Removing E:
$(BC)^+ = (ABCDE)$, contains E, so E is extraneous in $EBC \to D$.

Now $F_c = \{B \to EA, BC \to D, BED \to A\}$

Iteration #2:

Can't use union rule.

Checking $B \to EA$:

Removing B:
$\varnothing^+ = \varnothing$, doesn't contain B, so B is needed.

Removing E:
$F'_c = \{B \to A, BC \to D, BED \to A\}$
$(B)^+ = (AB)$, doesn't contain E, so E is needed.

Removing A:
$F'_c = \{B \to E, BC \to D, BED \to A\}$
$(B)^+ = (BE)$, doesn't contain A, so A is needed.

Checking $BC \to D$:

Removing B:
$(C)^+ = (C)$, doesn't contain B, so B is needed.

Removing C:
$(B)^+ = (ABE)$, doesn't contain C, so C is needed.

Removing D:
$F'_c = \{B \to EA, BED \to A\}$
$(BC)^+ = (ABCE)$, doesn't contain D, so D is needed.

Checking $BED \to A$:

Removing B:
$(ED)^+ = (ED)$, doesn't contain B, so B is needed.

Removing E:
$(BD)^+ = (ABDE)$, contains E, so E is extraneous in $BED \to A$.

Now $F_c = \{B \to EA, BC \to D, BD \to A\}$

Iteration #3:

Can't use union rule.

Checking $B \to EA$:

Removing B:
$\varnothing^+ = \varnothing$, doesn't contain B, so B is needed.

Removing E:
$F'_c = \{B \to A, BC \to D, BD \to A\}$
$(B)^+ = (AB)$, doesn't contain E, so E is needed.

Removing A:
$F'_c = \{B \to E, BC \to D, BD \to A\}$
$(B)^+ = (BE)$, doesn't contain A, so A is needed.

Checking $BC \to D$:

Removing B:
$(C)^+ = (C)$, doesn't contain B, so B is needed.

Removing C:
$(B)^+ = (ABE)$, doesn't contain C, so C is needed.

Removing D:
$F'_c = \{B \to EA, BD \to A\}$
$(BC)^+ = (ABCE)$, doesn't contain D, so D is needed.

Checking $BD \to A$:

Removing B:
$(D)^+ = (D)$, doesn't contain B, so B is needed.

Removing D:
$(B)^+ = (ABE)$, doesn't contain D, so D is needed.

Removing A:
$F'_c = \{B \to EA, BC \to D\}$
$(BD)^+ = (ABDE)$, contains A, so A is extraneous in $BD \to A$.

Now $F_c = \{B \to EA, BC \to D\}$

Iteration #4:

Can't use union rule.

Checking $B \to EA$:

Removing B:
$\varnothing^+ = \varnothing$, doesn't contain B, so B is needed.

Removing E:
$F'_c = \{B \to A, BC \to D\}$
$(B)^+ = (AB)$, doesn't contain E, so E is needed.

Removing A:
$F'_c = \{B \to E, BC \to D\}$
$(B)^+ = (BE)$, doesn't contain A, so A is needed.

Checking $BC \to D$:

Removing B:
$(C)^+ = (C)$, doesn't contain B, so B is needed.

Removing C:
$(B)^+ = (ABE)$, doesn't contain C, so C is needed.

Removing D:
$F'_c = \{B \to EA\}$
$(BC)^+ = (ABCE)$, doesn't contain D, so D is needed.

F_c didn't change, so the canonical cover is $\{B \to EA, BC \to D\}$
Thus, the minimal cover is $\{B \to A, B \to E, BC \to D\}$

C.

R is in 1NF, assuming all attributes in R are atomic.

But R is not in 2NF, since it has a non-prime attribute (A) that is dependent on a proper subset (B) of a candidate key (BC). This follows from $B \to EA$.

Thus, it is also not in 3NF, BCNF, etc. and the best/strictest normal form is 2NF.

D.

$F_c = \{B \to EA, BC \to D\}$

For each functional dependency, add LHS and RHS attributes in a new schema:

$R_1 = (A, B, E)$
$R_2 = (B, C, D)$

Since R_2 contains a candidate key for R ($\{B, C\}$), there is no need to create a new schema.

Finally, since no schema is contained in another schema, we are done.

Thus, the 3NF decomposition produced by the algorithm is:

$R_1(A, B, E)$, with $FD_1 = \{B \to AE\}$
$R_2(B, C, D)$, with $FD_2 = \{BC \to D\}$

Exercise 5

A.

Find all attributes that have not appeared on the RHS of any FD: $S_1 = B$
Find all attributes that have not appeared on the LHS of any FD: $S_2 = D$

Compute the closure set of S_1: $S_1^+ = \{B, D\}$

$S_1^+ \neq R$, so for each attribute x in $R - B$, test whether $A \cup \{x\}$ is a candidate key:

$(S_1 \cup \{A\})^+ = \{A, B\}^+ = \{A, B, C, D\} = R$, so $\{A, B\}$ is a candidate key.
$(S_1 \cup \{B\})^+ = \{B\}^+ = \{B, D\}$
$(S_1 \cup \{C\})^+ = \{B, C\}^+ = \{A, B, C, D\} = R$, so $\{B, C\}$ is a candidate key.

Thus, the candidate keys are $\{A, B\}$ and $\{B, C\}$.

B.

$result = \{R\} = \{ABCD\}$

Check if $R_i = ABCD$ is in BCNF:

for each subset a of R_i, check if a^+ contains all attributes of R_i, or no attributes of $R_i - a$:

$a = A :$
$a^+ = A$
$R_i - a = BCD$
a^+ contains no attributes of $R_i - a$. OK

$a = B :$
$a^+ = BD$
$R_i - a = ACD$
a^+ doesn't contain all attributes of R_i, and it contains attribute D of $R_i - a$. NOT OK

Thus, $a \to (a^+ - a) \cup R_i \equiv B \to D$ breaks BCNF rules.

$result = (result - R_i) \cup (R_i - D) \cup (B, D) = \{ABC, BD\}$

BD is in BCNF by construction.

Check if $R_i = ABC$ is in BCNF:

for each subset a of R_i, check if a^+ contains all attributes of R_i, or no attributes of $R_i - a$:

$a = A :$
$a^+ = A$
$R_i - a = BC$
a^+ contains no attributes of R˙i - a. OK

$a = B :$
$a^+ = BD$
$R_i - a = AC$
a^+ contains no attributes of $R_i - a$. OK

$a = C :$
$a^+ = AC$
$R_i - a = AB$
a^+ doesn't contain all attributes of R_i, and it contains attribute A of $R_i - a$. NOT OK

Thus, $a \to (a^+ - a) \cup R_i \equiv C \to A$ breaks BCNF rules.

$result = (result - R_i) \cup (R_i - A) \cup (C, A) = \{BD, BC, CA\}$

CA is in BCNF by construction.

Check if $R_i = BC$ is in BCNF:

for each subset a of R_i, check if a^+ contains all attributes of R_i, or no attributes of $R_i - a$:

$a = B :$
$a^+ = BD$
$R_i - a = C$
a^+ contains no attributes of $R_i - a$. OK

$a = C :$

$a^+ = CA$
$R_i - a = B$
a^+ contains no attributes of $R_i - a$. OK

$a = BC$:
$a^+ = ABCD$
a^+ contains all attributes of R_i. OK

Thus, BC is in BCNF.

Note: Normally we need not check if relation schemas with 2 attributes are in BCNF (they always are). Nonetheless, we run the algorithm for completeness' sake.

We are done.

The BCNF decomposition produced by the algorithm is:

$R_1(BD)$, with $FD_1 = \{B \to D\}$
$R_2(BC)$, with $FD_2 = \varnothing$
$R_3(AC)$, with $FD_3 = \{C \to A\}$

We see that the functional dependency $AB \to C$ is not preserved. This is normal, since it is not always possible to get a BCNF decomposition that is dependency preserving.