EXAMINATION GUIDE: FUNCTIONAL DEPENDENCY PROBLEM AND SOLUTION

You are given the below functional dependencies for the relation R = (A, B, C, D, E, F)

FDs = { A→ BC, C →DA, D→E, AD →F}

a) What are the keys for the relation R?
   To find the keys, we need to compute for the closure of the determinants (A, C, D, AD) in the given set of FDs.
   Therefore, closure of A, written as (A)+ is:
   (A)+ =
   1^{st} iteration answer = {A} by reflexivity rule
   2^{nd} iteration answer = {ABC} from A → BC
   3^{rd} iteration answer = {ABCD} from C → DA
   4^{th} iteration answer = {ABCDE} from D → E
   5^{th} iteration answer = {ABCDEF} from AD → F
   6^{th} iteration answer = no change to the 5^{th} iteration answer, therefore
   (A)+ = {ABCDEF} = R ={ABCDEF} i.e., all the attributes in (A)+ are included in R. Hence, we conclude that A is a key (super key)

   Computing C closure (C)+ =
   1^{st} iteration answer = {C} by reflexivity rule
   2^{nd} iteration answer = {CDA} from C → DA
   3^{rd} iteration answer = {CDAE} from D → E
   4^{th} iteration answer = {CDAEF} from AD → F
   5^{th} iteration answer = {CDAEFB} from A → BC
   6th iteration answer = no change to the 5th iteration answer, therefore
   (C)+ = { CDAEFB } OR rewritten as {ABCDEF}, which is same as R ={ABCDEF} i.e., all the attributes in (C)+ are included in R. Hence, we conclude that C is a key (super key)

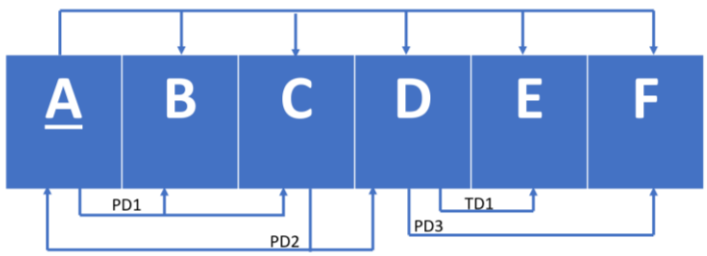
   Computing D closure (D)+ =
   1^{st} iteration answer = {D} by reflexivity rule
   2^{nd} iteration answer = {DE} from D → E
   3^{rd} iteration answer = no change to the 2^{nd} iteration answer, therefore
   (D)+ = { DE } which is not equal to R ={ABCDEF} i.e., (D)+ does not include all the attributes found in R. Hence, we conclude that D is NOT a key (super key)

   Computing AD closure (AD)+ =
   1^{st} iteration answer = {AD} by reflexivity
   2^{nd} iteration answer = {ADBC} from A → BC
   3^{rd} iteration answer = { ADBC } from C → DA
   4^{th} iteration answer = { ADBCE } from D → E
   5^{th} iteration answer = { ADBCEF } from AD → F
   6^{th} iteration answer = no change to the 5th iteration answer, therefore

(AD)+ = { ADBCEF } OR rewritten as {ABCDEF}, which is same as R ={ABCDEF} i.e., all the attributes in (AD)+ are included in R. Hence, we conclude that AD is a key (super key)

Therefore, A, C and AC are super keys as shown above. However, A and C can be candidate keys. AD cannot be a candidate key because candidate keys are subset of super keys, comparing A and AD, A is subset of AD, so we exclude AD from being a candidate. This means ONLY A and C can be Candidate keys. It is from these keys that a designer chooses one key as Primary key.

b) Is the relation R(ABCDEF) in 1NF/2NF/3NF/BCNF? Given FDs = { A→ BC, C →DA, D→E, AD →F}
Draw the dependency diagram and indicate all the FDs



From the computation in (a), we choose between A and C to be a Primary key

In 1st normal form (1NF)

- PK must be identified.
- All dependencies are identified i.e., 3 Partial dependencies (PD) and 1 transitive dependencies (TD)
- This means that partial keys and transitive dependencies exist in the relation R, given in the question.

These conditions confirm a relation in 1NF.

FYI – the relation cannot be in 2NF because of all the partial dependencies. Also, cannot be in 3NF because of the transitive dependency. Further, not in Boyce Codd Normal Form (BCNF) because of C → A, i.e., a non-prime attribute determines a prime/key attribute.

a) Compute the canonical (minimal) cover of FDs = { A→ BC, C →DA, D→E, AD →F}
Step I: decompose FDs to appear as singleton attributes on RHS(dependents), i.e.,
FDs ={
A → B
A → C
C → D
C → A
D →E
AD → F
}

Step II: identify extraneous/redundant attributes on the LHS (determinants).
For AD → F, check if either A or D is extraneous in our set of FDs, excluding AD → F
To check if A is extraneous, compute (D)+
Computing D closure (D)+ =
1st iteration answer = {D} by reflexivity
2nd iteration answer = {DE} from D → E
3rd iteration answer = no change in the answer in 2nd iteration. Since A is not included in (D)+ ={DE} we conclude that A is not extraneous.

To check if D is extraneous, compute (A)+
Computing A closure (A)+ =
1st iteration answer = {A} by reflexivity
2nd iteration answer = {ABC} from A → B and A → C
3rd iteration answer = {ABCD} from C → D
Note that D is included in (A)+ ={ABCD}, thus we conclude that D is extraneous and must be excluded.
Hence AD → F becomes A → F.
New set of FDs is now, Fc ={
A → B
A → C
C → D
C → A
D →E
A → F
}
Step III: identify extraneous/redundant functional dependencies in our new set of FDs, Fc.
To check if a FD in Fc is extraneous, compute the closure of its determinant while excluding the FD in question.

For A → B in question,
Its closure, (A)+ =
1st iteration answer = {A} by reflexivity
2nd iteration answer = {ACF} from A → C and A → F
3rd iteration answer = {ACFD} from C → D and C → A
4th iteration answer = {ACFDE} from D → E
5th iteration answer = no change. Since B is not included in (A)+ ={ACFDE}, we conclude that A → B is not an extraneous FD and must be included in our minimal cover of FDs.

For A → C in question,
Its closure, (A)+ =
1st iteration answer = {A} by reflexivity
2nd iteration answer = {ABF} from A → B and A → F
3rd iteration answer = no change. Since C is not included n (A)+ ={ABF}, so we conclude that A → C is not an extraneous FD and must be included in our minimal cover of FDs.

For C → D in question,
Its closure, (C)+ =
1st iteration answer = {C} by reflexivity
2nd iteration answer = {CA} from C → A
3rd iteration answer = {CABF} from A → B, A → C, and A → F
4th iteration answer = no change. Since D is not included in (C)+ ={ CABF }, so we conclude that C → D is not an extraneous FD and must be included in our minimal cover of FDs.

For C → A in question,
Its closure, (C)+ =
1st iteration answer = {C} by reflexivity
2nd iteration answer = {CD} from C → D
3rd iteration answer = {CDE} from D → E
4th iteration answer = no change. Since A is not included in (C)+ ={ CDE }, so we conclude that C → A is not an extraneous FD and must be included in our minimal cover of FDs.

For D → E in question,
Its closure, (D)+ =
1st iteration answer = {D} by reflexivity
2nd iteration answer = no change. Since E is not included in (D)+ ={ D }, so we conclude that D → E is not an extraneous FD and must be included in our minimal cover of FDs.

For A → F in question,
Its closure, (A)+ =
1st iteration answer = {A} by reflexivity
2nd iteration answer = {ABC} from A → B and A → C
3rd iteration answer = {ABCD} from C → D and C → A
4th iteration answer = {ABCDE} from D → E
5th iteration answer = no change. Since F is not included in (A)+ ={ ABCDE }, so we conclude that A → F is not an extraneous FD and must be included in our minimal cover of FDs.

Our new set of FDs, also minimal cover comprises:

New set of FDs is now, Fc ={
A → B
A → C
C → D
C → A
D → E
A → F
}
Lastly, perform a union rule on RHS to give:
Canonical cover of FDs, Fc ={

$A \rightarrow BCF$

$C \rightarrow DA$

$D \rightarrow E$

}

More examples on computing functional dependencies.

Given the relation R = (A, B, C, D, E) with functional dependencies FDs = {CE → D, D → B, C → A}

a) Find all the candidate keys?

To get the candidate keys, we compute the closures of all determinants, i.e., CE, D and C as given in a set of FDs.

Computing CE closure (CE)+ =

1st iteration answer = {CE} by reflexivity rule

2nd iteration answer = {CED} from CE → D

3rd iteration answer = { CEDA } from C → A

4th iteration answer = { CEDAB} from D → B

5th iteration answer = no change to the 4th iteration answer, therefore

(CE)+ = { CEDAB } OR {ABCDE}, which is equal to R ={ABCDE} i.e., all the attributes in (CE)+ are included in R. Hence, we conclude that CE is a key (super key) of relation R.

Computing D closure (D)+ =

1st iteration answer = {D} by reflexivity rule

2nd iteration answer = {DB} from D → B

3rd iteration answer = no change to the 2nd iteration answer, therefore

(D)+ = { DB } which is not equal to R ={ABCDE} i.e., (D)+ does not include all the attributes found in R. Hence, we conclude that D is NOT a key (super key) of relation R.

Computing C closure (C)+ =

1st iteration answer = {C} by reflexivity rule

2nd iteration answer = {CA} from C → A

3rd iteration answer = no change to the 2nd iteration answer, therefore

(C)+ = { CA } which is not equal to R ={ABCDE} i.e., (C)+ does not include all the attributes found in R. Hence, we conclude that C is NOT a key (super key) of relation R.

Computing CD closure (CD)+ =

1st iteration answer = {CD} by reflexivity rule

2nd iteration answer = {CDB} from D → B

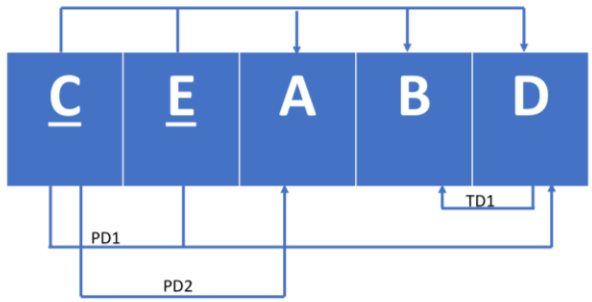
3rd iteration answer = {CDBA} from C → A

4th iteration answer = no change to the 4th iteration answer, therefore

(CD)+ = { CDBA } which is not equal to R ={ABCDE} i.e., (CD)+ does not include all the attributes found in R. Hence, we conclude that CD is NOT a key (super key) of relation R.

Therefore, CE is the only super key, as well as the candidate key of the relation R (A,B,C,D,E) with the given FDs in the question above.

b) Is the relation R = (A,B,C,D,E) with the given FDs = {CE → D, D → B, C → A} in 1NF/2NF/3NF/BCNF? If not in 3NF, show the process to take it to 3NF. Use the dependency diagram to identify the normal forms.



The relation R cannot be in 3NF, because of transitive dependence, TD1 shown in the diagram below. From CE → D ad D → B, then it means CE → B is true transitively.

Similarly, relation R cannot be in 2NF, simply because of the existing partial dependencies (TD1 and TD2) shown in the diagram.

The relation R is in 1NF because of the following:

- Primary key is identified i.e., CE (underlined)
- All dependencies are identified i.e., 2 PDs and 1 TD.

We need to take the relation R to a 3 NF.

First, we need to take relation R to the second normal form (2NF).

2NF

We need to remove all PDs from the relation R, and each PD will form a new table/relation, therefore.

PD1: CE → D

New relation R1 will have CED as attributes with CE as PK i.e., R1 = {CED}

PD2: C → A

New relation R2 will have CA as attributes with C as PK i.e., R2 = {CA}

TD1: D → B

In 2NF, TD does not form its own table, but must exist in R1. This mean R1 becomes R1 = {CEDB}, which shows D→ B exists.

3NF

We need to remove all transitive dependencies(TDs) from our existing relations.

In this case TD1 (D → B), still exists in R1 = {CEDB}

Therefore D → B, forms a new relation R3 with attributes DB with D as the PK, i.e., R3 =(DB)

Whereas R1 now remains as R1 = {CED}, with D as foreign key between R1 and R3

Finally, the tables/relation in 3NF are as follows:

R1 = {CED}, CE as PK

R2 = {CA}, C as PK

R3 = {DB}, D as PK.

c) Compute the canonical (minimal) cover of FDs ={CE →D, D → B, C → A}.
Step I: decompose the RHS to singleton attributes.
FDs ={
CE →D,
D → B,
C → A
}.

Step II: identify extraneous attributes on the LHS.
For CE → D, check if either C or E is extraneous in our set of FDs, excluding CE → D
To check if C is extraneous, compute (E)+
Computing E closure (E)+ =
1st iteration answer = {E} by reflexivity rule
2nd iteration answer = no change in the answer in 1st iteration. Since C is not included in (E)+ ={E} we conclude that C is not extraneous.

To check if E is extraneous, compute (C)+
Computing C closure (C)+ =
1st iteration answer = {C} by reflexivity rule
2nd iteration answer = {CA} from C → A
3rd iteration answer = no change in the answer in 2nd iteration. Since E is not included in (C)+ ={CA} we conclude that E is not extraneous.

Note, neither C nor E is extraneous, so attributes CE remains on the RHS.

Step III: identify functional dependencies that might be extraneous in our current set of FDs, which comprises:

FDs ={
CE →D,
D → B,
C → A
}.

To check if FD is extraneous, compute the closure of the determinant against the set of FDs, without considering the FD in question.

For CE → D in question,
Its closure, (CE)+ =
1st iteration answer = {CE} by reflexivity
2nd iteration answer = {CEA} from C → A
3rd iteration answer = no change. Since D is not included in (CE)+ ={ CEA }, so we conclude that CE → D is not an extraneous FD and must be included in our minimal cover of FDs.

For D → B in question,
Its closure, (D)+ =
1st iteration answer = {D} by reflexivity
2nd iteration answer = no change. Since B is not included in (D)+ ={ D }, so we conclude that D → B is not an extraneous FD and must be included in our minimal cover of FDs.

For C → A in question,
Its closure, (C)+ =
1st iteration answer = {C} by reflexivity
2nd iteration answer = no change. Since A is not included in (C)+ ={ C }, so we conclude that C → A is not an extraneous FD and must be included in our minimal cover of FDs.

New set of FDs is now, Fc ={
CE → D
D → B
C → A
}
Finally, Canonical cover of FDs, Fc ={
CE → D
D → B
C → A
}

<u>More examples on computing functional dependencies.</u>

Suppose relation R = (A, B, C, D, E, F, G,) with functional dependencies FDs = {AD → BF, CD → EGC, BD → F, E → D, F → C, D → F}.

a) Compute the canonical (minimal) cover of FDs.
Step I: decompose the RHS to appear as singleton attributes.
FDs = {
AD → B
AD → F
CD → E
CD → G
CD → C
BD → F
E → D
F → C
D → F
}.

Step II: identify extraneous attributes on the LHS of our FDs.
For AD → B, check if either A or D is extraneous against our set of FDs but excluding AD → B
To check if A is extraneous, compute (D)+
Computing D closure (D)+ =
1st iteration answer = {D} by reflexivity
2nd iteration answer = {DF} from D → F
3rd iteration answer = {DFC} from F → C
4th iteration answer = {DFCEG} from CD → E, CD → G, and CD → C
5th iteration answer = no change in the answer in 4th iteration. Since A is not included in (D)+ ={ DFCEG } we conclude that A is not extraneous.

To check if D is extraneous, compute (A)+
Computing A closure (A)+ =
1st iteration answer = {A} by reflexivity
2nd iteration answer = no change in the answer in 1st iteration. Since D is not included in (A)+ ={ A } we conclude that D is not extraneous.
Therefore, neither A nor D is extraneous, thus must remain on the LHS of AD → B.

For AD → F, check if either A or D is extraneous against our set of FDs but excluding AD → F.
To check if A is extraneous, compute (D)+
Computing D closure (D)+ =
1st iteration answer = {D} by reflexivity
2nd iteration answer = {DF} from D → F
3rd iteration answer = {DFC} from F → C
4th iteration answer = {DFCEG} from CD → E, CD → G, and CD → C

5^{th} iteration answer = no change in the answer in 4^{th} iteration. Since A is not included in (D)+ ={ DFCEG } we conclude that A is not extraneous.

To check if D is extraneous, compute (A)+
Computing A closure (A)+ =
1^{st} iteration answer = {A} by reflexivity
2^{nd} iteration answer = no change in the answer in 1^{st} iteration. Since D is not included in (A)+ ={ A } we conclude that D is not extraneous.
Therefore, neither A nor D is extraneous, thus must remain on the LHS of AD → F.

For CD → E, check if either C or D is extraneous against our set of FDs but excluding CD → E.
To check if C is extraneous, compute (D)+
Computing D closure (D)+ =
1^{st} iteration answer = {D} by reflexivity
2^{nd} iteration answer = {DF} from D → F
3^{rd} iteration answer = {DFC} from F → C
4th iteration answer = {DFCG} from CD → G and CD → C
5^{th} iteration answer = no change in the answer in 4^{th} iteration. Since C is included in (D)+ ={ DFCG}.
We conclude that C is extraneous. So, we remove it from CD → E to become D → E.

Similarly, for CD → G, check if either C or D is extraneous against our set of FDs but excluding CD → G.
To check if C is extraneous, compute (D)+
Computing D closure (D)+ =
1^{st} iteration answer = {D} by reflexivity
2^{nd} iteration answer = {DF} from D → F
3^{rd} iteration answer = {DFC} from F → C
4th iteration answer = {DFCE}
Since C is included in (D)+ ={ DFCE }.
We conclude that C is extraneous. So, we remove it from CD → G to become D → G.

Similarly, for CD → C, check if either C or D is extraneous against our set of FDs but excluding CD → C.
To check if C is extraneous, compute (D)+
Computing D closure (D)+ =
1^{st} iteration answer = {D} by reflexivity
2^{nd} iteration answer = {DF} from D → F
3^{rd} iteration answer = {DFC} from F → C
Since C is included in (D)+ ={ DFCEG }.
We conclude that C is extraneous. So, we remove it from CD → C to become D → C.

For BD → F, check if either B or D is extraneous against our set of FDs but excluding BD → F.
To check if B is extraneous, compute (D)+

Computing D closure (D)+ =
1st iteration answer = {D} by reflexivity
2nd iteration answer = {DF} from D → F
3rd iteration answer = {DFC} from F → C
4th iteration answer = {DFCEG} from D → E, D → G and D → C
5th iteration answer = no change in the answer in 4th iteration. Since B is not included in (D)+ ={ DFCEG } we conclude that B is not extraneous and should remain on the LHS.

To check if D is extraneous in BD → F, compute (B)+
Computing B closure (B)+ =
1st iteration answer = {B} by reflexivity
2nd iteration answer = no change in the answer in 1st iteration. Since D is not included in (B)+ ={ B } we conclude that D is not extraneous in BD → F and should remain on the LHS.
Therefore, neither B nor D is extraneous, so BD remains on the LHS as non-extraneous attributes.

Therefore, our new set of FDs = {
AD → B
AD → F
D → E
D → G
D → C
BD → F
E → D
F → C
D → F
}

Step III: identify extraneous functional dependencies (FD) in our new set of FDs found in step II:
To check if FD is extraneous, compute the closure of the determinant against the set of FDs, without considering the FD in question.

For AD → B in question,
Its closure, (AD)+ =
1st iteration answer = {AD} by reflexivity
2nd iteration answer = {ADF} from AD → F
3rd iteration answer = {ADFC} from F → C
4th iteration answer = {ADFCEF} from A → E, D → G and D → C
5th iteration answer = {ADFCEF} from E → D
6th iteration answer = no change. Since B is not included in (AD)+ ={ ADFCEF }.
So, we conclude that AD → B is not an extraneous FD and must be included in our minimal cover of FDs.

For AD → F in question,
Its closure, (AD)+ =
1st iteration answer = {AD} by reflexivity
2nd iteration answer = {ADB} from AD → B
3rd iteration answer = {ADBEGC} from D → E, D → G, and D → C
4th iteration answer = { ADBEGCF } from D → F
5th iteration answer = { ADBEGCF } from F → C
6th iteration answer = no change. Since F is included in (AD)+ ={ ADBEGCF }.
So, we conclude that AD → F is an extraneous FD and must be excluded from our minimal cover of FDs.

For D → E in question,
Its closure, (D)+ =
1st iteration answer = {D} by reflexivity
2nd iteration answer = {DGCF} from D → G, D → C, and D → F
3rd iteration answer = { DGCF } from F → C
4th iteration answer = no change. Since E is NOT included in (D)+ ={ DGCF }.
So, we conclude that D → E is an NOT extraneous FD and must be included in our minimal cover of FDs.

For D → G in question,
Its closure, (D)+ =
1st iteration answer = {D} by reflexivity
2nd iteration answer = {DECF} from D → E, D → C, and D → F
3rd iteration answer = { DECF } from E → D
4th iteration answer = { DECF } from F → C
5th iteration answer = no change. Since G is NOT included in (D)+ ={ DECF }.
So, we conclude that D → G is an NOT extraneous FD and must be included in our minimal cover of FDs.

For D → C in question,
Its closure, (D)+ =
1st iteration answer = {D} by reflexivity
2nd iteration answer = {DEGF} from D → E, D → G, and D → F
3rd iteration answer = { DEGFC } from F → C
5th iteration answer = no change. Since C is included in (D)+ ={ DEGFC }.
So, we conclude that D → C is an extraneous FD and must be eliminated in our minimal cover of FDs.

For BD → F in question,
Its closure, (BD)+ =
1st iteration answer = {BD} by reflexivity
2nd iteration answer = {BDEGF} from D → E, D → G, and D → F

3rd iteration answer = { BDEGF } from E → D
4th iteration answer = { BDEGFC} from F → C
5th iteration answer = no change. Since F is included in (BD)+ ={ BDEGFC }.
So, we conclude that BD → F is an extraneous FD and must be eliminated in our minimal cover of FDs.

For E → D in question,
Its closure, (E)+ =
1st iteration answer = {E} by reflexivity
2nd iteration answer = no change. Since D is NOT included in (E)+ ={ E }.
So, we conclude that E → D is an NOT extraneous FD and must be included in our minimal cover of FDs.

For F → C in question,
Its closure, (F)+ =
1st iteration answer = {F} by reflexivity
2nd iteration answer = no change. Since C is NOT included in (F)+ ={ F}.
So, we conclude that F → C is an NOT extraneous FD and must be included in our minimal cover of FDs.

For D → F in question,
Its closure, (D)+ =
1st iteration answer = {D} by reflexivity
2nd iteration answer = {DEG} from D → E and D → G
3rd iteration answer = {DEG} from E → D
4th iteration answer = no change. Since F is NOT included in (D)+ ={ DEG }.
So, we conclude that D → F is an NOT extraneous FD and must be included in our minimal cover of FDs.

Therefore, the minimal cover of FDs comprises of :
Fc = {
AD → B
D → E
D → G
E → D
F → C
D → F
}
Lastly, apply a union rule to combine attributes on the RHS to give, minimal cover as:
Fc = {
AD → B
D → EGF
E → D
F → C

}

More examples on computing functional dependencies.

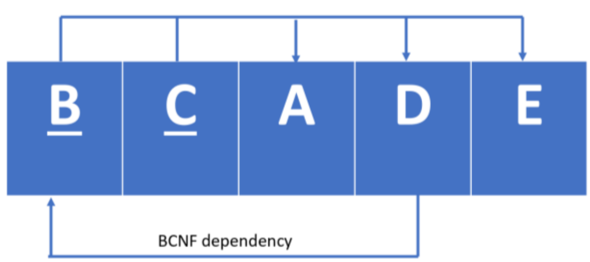
Suppose relation R = (A, B, C, D, E) with functional dependencies FDs = {BC → ADE, D → B}.

a) Find all candidate keys.
To get candidate keys, compute closures for BC and D.
Computing BC closure (BC)+ =
1st iteration answer = {BC} by reflexivity rule
2nd iteration answer = {BCADE} from BC → ADE
3rd iteration answer = { BCADE } from D → B
4th iteration answer = no change to the 3rd iteration answer, therefore
(BC)+ = { BCADE } OR {ABCDE}, which is equal to R ={ABCDE} i.e., all the attributes in (CE)+ are included in R. Hence, we conclude that BC is a key (super key) of relation R.

Computing D closure (D)+ =
1st iteration answer = {D} by reflexivity rule
2nd iteration answer = {DB} from D → B
3rd iteration answer = no change to the 2nd iteration answer, therefore
(D)+ = { DB }, which is NOT equal to R ={ABCDE} i.e., (D)+ does not include all the attributes found in R. Hence, we conclude that D is NOT a key (super key) of relation R.

Therefore, BC is a super key and a candidate key for relation R given in question.

b) Which normal for does relation R = (A, B, C, D, E) satisfy? Given FDs = {BC → ADE, D → B}



Relation R is in 3NF, but not in Boyce Codd normal Codd (BCNF), because of the BCNF dependency. In 3NF we observe that all non-prime/key attributes are dependent on the key attributes (BC). Not in BCNF, because a non-prime attribute determines a key attribute. So we conclude that the relation R given above is in 3NF but not in BCNF.

More examples on computing functional dependencies.

Suppose you are given relation R = (A, B, C, D, E, F, G, H) with functional dependencies FDs = {AC → G, D → EG, BC → D, CG → BD, ACD → B, CE → AG}. Compute the canonical (minimal) cover of FDs.

Step I: decompose the RHS to singleton attributes.

FDs = {
AC → G
D → E
D → G
BC → D
CG → B
CG → D
ACD → B
CE → A
CE → G
}

Step II: identify extraneous attributes on the LHS.

For AC → G, check if either A or C is extraneous against our set of FDs but excluding AC → G
To check if A is extraneous, compute (C)+
Computing C closure (C)+ =
1st iteration answer = {C} by reflexivity
2nd iteration answer = no change in the answer in 1st iteration. Since A is not included in (C)+ ={ C } we conclude that A is not extraneous.

Computing A closure (A)+ =
1st iteration answer = {A} by reflexivity
2nd iteration answer = no change in the answer in 1st iteration. Since C is not included in (A)+ ={ A } we conclude that C is not extraneous.
Therefore, neither A nor C is extraneous, so attributes AC remains on the LHS in AC → G.

For BC → D, check if either B or C is extraneous against our set of FDs but excluding BC → D
To check if B is extraneous, compute (C)+
Computing C closure (C)+ =
1st iteration answer = {C} by reflexivity
2nd iteration answer = no change in the answer in 1st iteration. Since B is not included in (C)+ ={ C } we conclude that B is not extraneous.

Computing B closure (B)+ =
1st iteration answer = {B} by reflexivity
2nd iteration answer = no change in the answer in 1st iteration. Since D is not included in (A)+ ={ B } we conclude that D is not extraneous.
Therefore, neither B nor C is extraneous, so attributes BC remains on the LHS in BC → D.

For CG → B, check if either C or G is extraneous against our set of FDs but excluding CG → B
To check if C is extraneous, compute (G)+
Computing G closure (G)+ =
1st iteration answer = {G} by reflexivity
2nd iteration answer = no change in the answer in 1st iteration. Since C is not included in (G)+ ={ G } we conclude that C is not extraneous.

Computing C closure (C)+ =
1st iteration answer = {C} by reflexivity
2nd iteration answer = no change in the answer in 1st iteration. Since G is not included in (C)+ ={ C } we conclude that G is not extraneous.
Therefore, neither C nor G is extraneous, so attributes CG remains on the LHS in CG → B.

For CG → D, check if either C or G is extraneous against our set of FDs but excluding CG → D
To check if C is extraneous, compute (G)+
Computing G closure (G)+ =
1st iteration answer = {G} by reflexivity
2nd iteration answer = no change in the answer in 1st iteration. Since C is not included in (G)+ ={ G } we conclude that C is not extraneous.

Computing C closure (C)+ =
1st iteration answer = {C} by reflexivity
2nd iteration answer = no change in the answer in 1st iteration. Since G is not included in (C)+ ={ C } we conclude that G is not extraneous.
Therefore, neither C nor G is extraneous, so attributes CG remains on the LHS in CG → D.

For ACD → B, check if either AC or D is extraneous against our set of FDs but excluding ACD → B
To check if D is extraneous, compute (AC)+
Computing AC closure (AC)+ =
1st iteration answer = {AC} by reflexivity
2nd iteration answer = {ACG} from AG → G
3rd iteration answer = {ACGBD} from CG → B and CG → D
4th iteration answer = {ACGBDE} from D → E and D → G
5th iteration answer = no change in the answer in 4th iteration. Since D is included in (AC)+ ={ ACGBDE } we conclude that D is not extraneous, and we must exclude it from ACD → B to become AC → B.

Check if either A or C is extraneous in AC → B.
To check for A, compute C closure (C)+ =
1st iteration answer = {C} by reflexivity
2nd iteration answer = no change in the answer in 1st iteration. Since A is not included in (C)+ ={ C } we conclude that A is not extraneous.

To check for C compute A closure (A)+ =
1st iteration answer = {A} by reflexivity
2nd iteration answer = no change in the answer in 1st iteration. Since C is not included in (A)+ ={ A } we conclude that C is not extraneous.
Therefore, neither A nor C is extraneous, so attributes AC remains on the LHS in AC → B.

For CE → A, check if either C or E is extraneous against our set of FDs but excluding CE → A
To check if C is extraneous, compute (E)+
Computing E closure (E)+ =
1st iteration answer = {E} by reflexivity
2nd iteration answer = no change in the answer in 1st iteration. Since C is not included in (E)+ ={ E } we conclude that C is not extraneous.

Computing C closure (C)+ =
1st iteration answer = {C} by reflexivity
2nd iteration answer = no change in the answer in 1st iteration. Since E is not included in (C)+ ={ C } we conclude that E is not extraneous.
Therefore, neither C nor E is extraneous, so attributes CE remains on the LHS in CE → A.

Similarly, For CE → G, check if either C or E is extraneous against our set of FDs but excluding CE → G
To check if C is extraneous, compute (E)+
Computing E closure (E)+ =
1st iteration answer = {E} by reflexivity
2nd iteration answer = no change in the answer in 1st iteration. Since C is not included in (E)+ ={ E } we conclude that C is not extraneous.

Computing C closure (C)+ =
1st iteration answer = {C} by reflexivity
2nd iteration answer = no change in the answer in 1st iteration. Since E is not included in (C)+ ={ C } we conclude that E is not extraneous.
Therefore, neither C nor E is extraneous, so attributes CE remains on the LHS in CE → G.

Our new set of FDs ={
AC → G
D → E
D → G
BC → D
CG → B
CG → D
AC → B
CE → A
CE → G

}

Step III: identify extraneous FDs found in our new set of FDs.
To check if FD is extraneous, compute the closure of the determinant against the set of FDs, without considering the FD in question.

For AC → G in question,
Its closure, (AC)+ =
1st iteration answer = {AC} by reflexivity
2nd iteration answer = {ACB} from AD → B
3rd iteration answer = {ACBD} from BC → D
4th iteration answer = {ACBDEG} from D → E and D → G
5th iteration answer = { ACBDEG } from AC → B
Since G is included in (AC)+ ={ ACBDEG }.
So, we conclude that AC → G is an extraneous FD and must be excluded in our minimal cover of FDs.

For D → E in question,
Its closure, (D)+ =
1st iteration answer = {D} by reflexivity
2nd iteration answer = {DG} from D → G
3rd iteration answer = no change in the answer in 2nd iteration. Since E is not included in (D)+ ={ DG } we conclude that D → E is not extraneous in our FDs and must be included in the minimal set of FDs.

For D → G in question,
Its closure, (D)+ =
1st iteration answer = {D} by reflexivity
2nd iteration answer = {DE} from D → E
3rd iteration answer = no change in the answer in 2nd iteration. Since G is not included in (D)+ ={ DE } we conclude that D → G is not extraneous in our FDs and must be included in the minimal set of FDs.

For BC → D in question,
Its closure, (BC)+ =
1st iteration answer = {BC} by reflexivity
2nd iteration answer = no change in the answer in 1st iteration. Since E is not included in (BC)+ ={ BC } we conclude that BC → D is not extraneous in our FDs and must be included in the minimal set of FDs.

For CG → B in question,
Its closure, (CG)+ =
1st iteration answer = {CG} by reflexivity
2nd iteration answer = {CGD} from CG → D

3rd iteration answer = {CGDE} from D → E and D → G
4th iteration answer = {CGDEA} from CE → A and CE → G
5th iteration answer = { CGDEAB} from AC → B
Since B is included in (CG)+ ={ CGDEAB } we conclude that CG → B is extraneous and must be excluded from the minimal set of FDs.

For CG → D in question,
Its closure, (CG)+ =
1st iteration answer = {CG} by reflexivity
2nd iteration answer = no change in the answer in 1st iteration. Since D is not included in (CG)+ ={ CG } we conclude that CG → D is not extraneous in our FDs and must be included in the minimal set of FDs.

For AC → B in question,
Its closure, (AC)+ =
1st iteration answer = {AC} by reflexivity
2nd iteration answer = no change in the answer in 1st iteration. Since B is not included in (AC)+ ={ AC } we conclude that AC → B is not extraneous in our FDs and must be included in the minimal set of FDs.

For CE → A in question,
Its closure, (CE)+ =
1st iteration answer = {CE} by reflexivity
2nd iteration answer = {CEG} from CE → G
3rd iteration answer = {CEGD} from CG → D
4th iteration answer = {CEGD} from D → E and D → G
5th iteration answer = no change in the answer in 4th iteration. Since A is not included in (CE)+ ={ CEGD } we conclude that CE → A is not extraneous in our FDs and must be included in the minimal set of FDs.

For CE → G in question,
Its closure, (CE)+ =
1st iteration answer = {CE} by reflexivity
2nd iteration answer = {CEA} from CE → A
3rd iteration answer = {CEAB} from AC → B
4th iteration answer = {CEABD} from BC → D
5th iteration answer = {CEABDG} from D → G and D → E
6th iteration answer = no change in the answer in 5th iteration. Since G is included in (CE)+ ={ CEABDG } we conclude that CE → G is extraneous in our FDs and must be eliminated in the minimal set of FDs.

Therefore,
New set of FDs is now, Fc ={
D → E

D → G
BC → D
CG → D
AC → B
CE → A
}
Lastly, perform a union rule on RHS to give:
Canonical cover of FDs, Fc ={
D → EG
BC → D
CG → D
AC → B
CE → A
}