Relation W = {A, B, C, D, E, F, G, I, J, K, L, M, N, P, Q, R, S, T}

F={ AB->C
A->DEFG
E->G
F->DJ
G->DI
D->KL
D->M
DM->NP
L->D
PQR->ST
PR->S }

1. **Determine a candidate key**
   Step 1: determine a super key
   Selecting all attributes that appear on the left hand side but not on the right hand side, we have {ABQR}

   Calculate {ABQR}+ = {ABQR CDEF GJIK LMNP ST}
   If the closure contains all attributes, it is a super key.

   Step 2: check if the super key is minimal
   Remove one attribute a time and then recalculate the closure
   {BQR}+ doesn't contain A
   {AQR}+ doesn't contain B
   {ABR}+ doesn't contain Q
   {ABQ}+ doesn't contain R

   If no attribute can be removed from the super key, it is minimal and is a candidate key.

2. **Calculating the minimal cover**

   Step 1: Every right hand side has a single attribute
   F={ AB->C
   A->D A->E A->F A->G
   E->G
   F->D F->J
   G->D G->I
   D->K D->L
   D->M
   DM->N DM->P
   L->D
   PQR->S PQR->T
   PR->S }

   Step 2: Every left hand side is irreducible
   Test AB->C:

To test whether B can be removed for the Left-Hand-Side(LHS), calculate {A}+. If B is in {A}+, it is redundant; otherwise it is not.
{A}+ ={ADEFGJKLM}. Since B is not in {A}+, B is not redundant.
To test whether A can be removed for the LHS, calculate {B}+. If A is in {B}+, it is redundant; otherwise it is not.
Calculate {B}+={B}. Since A is not in {B}+, A is not redundant.

In the same way, you can test other FDs where the left hand side contains more than one attribute.

After Step 2, we have

F={ AB->C
A->D A->E A->F A->G
E->G
F->D F->J
G->D G->I
D->K D->L
D->M
D->N D->P
L->D
PQR->S PQR->T
PR->S }

Step 3: remove redundant FDs

Test whether A->D is redundant, calculate {A}+ without A->D, if D is in {A}+, it means A->D is redundant. In the same way you can test the other FDs.

F={ AB->C
~~A->D~~ A->E A->F ~~A->G~~
E->G
F->D F->J
G->D G->I
D->K D->L
D->M
D->N D->P
L->D
~~PQR->S~~ PQR->T
PR->S }

The minimal cover is:

F={ AB->C
A->E A->F
E->G
F->D F->J
G->D G->I

D->K D->L
D->M
D->N D->P
L->D
PQR->T
PR->S }

3. Normalising to 3NF

Step 1: Create a relation based on each FD

R1(ABC) {AB->C}
R2(AE) {A->E}
R3(AF) {A->F}
R4(EG) {E->G}
R5(FD) {F->D}
R6(FJ) {F->J}
R7(GD) {G->D}
R8(GI) {G->I}
R9(DK) {D->K}
R10(DL) {D->L}
R11(DM) {D->M}
R12(DN) {D->N}
R13(DP) {D->P}
R14(LD) {L->D}
R15(PQRT) {PQR->T}
R16(PRS) {PR->S}

Step 2: Merge FDs with the same left hand side

R1(ABC) {AB->C}
R2(AEF) {A->E, A->F}
R3(EG) {E->G}
R4(FDJ) {F->D, F->J}
R5(GDI) {G->D, G->I}
R6(DKLMNP) {D->K, D->L, D->M, D->N, D->P}
R7(LD) {L->D}
R8(PQRT) {PQR->T}
R9(PRS) {PR->S}

Step 3: Merge tables with equivalent keys

find tables that are subsets of other groups and try to merge them
R7(LD) is a subset of R6(DKLMNP). R7 can be merged with R6. We have
R1(ABC) {AB->C} candidate key is AB
R2(AEF) {A->E, A->F}. candidate key is A
R3(EG) {E->G} candidate key is E
R4(FDJ) {F->D, F->J} candidate key is F
R5(GDI) {G->D, G->I} candidate key is G
R6(DKLMNP) {D->K, D->L, D->M, D->N, D->P, L->D}, either D or L is a candidate key

R7(PQRT)	{PQR->T}	candidate key is PQR
R8(PRS)	{PR->S}	candidate key is PR