1.

(a)

A+ = AE, so B is not a superkey and A->E violates BCNF.

BC+ = ABCEF, so BC is not a superkey and BC->AE violates BCNF.

C+ = ACEF, so C is not a superkey and C->ACF violates BCNF.

DE+ = ADE, so DE is not a superkey and C->A violates BCNF.

EFG+ = ABEFG, so EFG is not a superkey and EFG->AB violates BCNF.

I+ = AEIJ, so I is not a superkey and I->J violates BCNF.

J+ = AEIJ, so J is not a superkey and J->AI violates BCNF.

(b)

Decompose R using FD C->ACF. C+ = ACEF, so R1 = ACEF, R2 = BCDGHIJ.

(1)

Project the FDs onto R1 = ACEF:

A+ = AE, A->E, violates BCNF

C+ = ACEF, C->AEF.

F+ = F, nothing.

So we must decompose R1 further.

Decompose R1 using FD A->E. This yields two relations: R3 = AE, R4 = ACF.

(2)

Project the FDs onto R3 = AE:

A+ = AE, A->E, A is a superkey of R3

E+ = E, nothing

Any supersets of A can only generate weaker FDs than what we already have.

This relation satisfies BCNF.

(3)

Project the FDs onto R4 = ACF.

A+ = AE, nothing

C+ = ACEF, C->AF; C is a superkey of R4

F+ = F, nothing

Any supersets of C can only generate weaker FDs than what we already have.

AF+ = AF, nothing.

This also satisfies BCNF.

(4)

Return to R2 = BCDGHIJ and project the FDs onto it.

B+ = B, nothing

C+ = ACEF, nothing

D+ = D, nothing

G+ = G, nothing

H+ = H, nothing

I+ = IJ, I->J, violates BCNF.

J+ = AEIJ, J->I, violates BCNF.

BC+ = ABCEF, nothing.

So we must decompose R2 further.

Decompose R2 using FD I->J, this yields two relations: R5 = IJ, R6 = BCDGHI.

(5)

Project the FDs onto R5 = IJ

I+ = IJ, I->J, I is a superkey

J+ = AEIJ, J->I, J is a superkey

This relation satisfies BCNF.

(6)

Project the FDs onto R6 = BCDGHI

B+ = B, nothing

C+ = ACEF, nothing

D+ = D, nothing

G+ = G, nothing

H+ = H, nothing

I+ = IJ, nothing

BC+ = ABCEF, nothing

This relation satisfies BCNF.

So our Final Decomposition:

(a) R3 = AE with FD A->E
(b) R4 = ACF with FD C->AF
(c) R5 = IJ with FD I->J, J->I
(d) R6 = BCDGHI with no FDs.

(c)

My decomposition is not dependency preserving. For each of the first three original FDs in set S, there is a relation that includes all of the FD's attributes. This ensures they are preserved. However, we can construct valid instances of the relations in the final schema that, when joined, create a table that violates FDs.

We lost BC->AE, DE->A, EFG->AB, J->AI.

For example: DE->A:

A E	C A F	I J	B C D G H I
1 5	3 1 6	9 9	2 3 4 7 8 9
2 5	4 2 6		2 4 4 7 8 9

A	B	C	D	E	F	G	H	I	J
1	2	3	4	5	6	7	8	9	9
2	2	4	4	5	6	7	8	9	9

Which violates DE->A, and other FDs I mentioned are similar.

2.

(a)

Since There are no F and H on the right hand side, so our minimal keys must have F and H.

And FH += FH, this is not a key.

Since there are no D on the left hand side, so our minimal keys shouldn't have D.

CFH += ABCDEFGH, so CFH is a minimal key and no superset of CFH can be a key.

EFH+ = ABCDEFGH, so EFH is a minimal key, no superset of EFH can be a key.

GFH + = ABCDEFGH, so GFH is a minimal key and no superset of GFH can be a key.

Others like AFH, BFH, ABFH aren't keys.

So minimal keys are CFH, EFH, GFH.

(b)

First, we need to simplify to singleton right-hand sides and call this set S1:

1. A->B
2. BC->A
3. BC->C…
4. BC->E
5. C->B
6. EF->C…
7. EF->G
8. EFG->A…
9. EFG->B…
10. EFG->C
11. EFG->D
12. GH->A…
13. GH->B…
14. GH->C
15. GH->D

Then, we need to remove redundant FD.

FD	Exclude this from S1	Closure	Decision
1	1	A+ = A	keep
2	2	BC+ = BCE	keep
3	3	BC+ = ABCE	discard
4	3, 4	BC+ = ABC	keep
5	3, 5	C+ = C	keep
6	3, 6	EF+ = ABCDEFG	discard
7	3, 6, 7	EF+ = EF	keep
8	3,6,8	EFG+ = ABCDEFG	discard
9	3,6,8,9	EFG+ = ABCDEFG	discard
10	3,6,8,9,10	EFG+ = DEFG	keep
11	3,6,8,9,11	EFG+ = ABCEFG	keep
12	3,6,8,9,12	GH+ = ABCDEGH	discard
13	3,6,8,9,12,13	GH+ = ABCDEGH	discard
14	3,6,8,9,12,13,14	GH+ = DGH	keep

15 3,6,8,9,12,13,15 GH+ = ABCEGH keep

Let’s call the remaining FDs S2:

1. A->B
2. BC->A
3.
4. BC->E
5. C->B
6.
7. EF->G
8.
9.
10. EFG->C
11. EFG->D
12.
13.
14. GH->C
15. GH->D

Now let’s try reducing the LHS of FDs with multiple attributes on the LHS

2. BC->A

   B+ = B so we can’t reduce LHS to B.

   C+ = ABCE, so we can reduce LHS to C

4. BC->E
   Same as above, we can reduce LHS to C

7. EF->G

   E+ = E, so we can’t reduce LHS to E

   F+ = F, so we can’t reduce LHS to F

10. EFG->C

    E+ = E, can’t reduce

    F+ = F, can’t reduce

    G+ = G, can’t reduce

    EF+ = CDEFG, we can reduce LHS to EF

    EG+ = EG

FG+ = FG

11. EFG->D

Same as above

14. GH->C

G+ = G, can't reduce

H+ = H, can't recude

15 GH->D

Same as above

Let's call the set of FDs now S3:

1. A->B
2. C->A
3.
4. C->E
5. C->B…
6.
7. EF->G
8.
9.
10. EF->C
11. EF->D
12.
13.
14. GH->C
15. GH->D

Now we need to further simplification

FD	Exclude this from S3	Closure	Decision
1	1	A+ = A	keep
2	2	C+ = BCE	keep
4	4	C+ = ABC	keep
5	5	C+ = ABCE	discard
7	5, 7	EF+ = ABCDEF	keep

10	5,10	EF+ = DEFG	keep
11	5, 11	EF+ = ABCEFG	keep
14	5,14	GH+ = DGH	keep
15	5,15	GH+ = ABCEGH	keep

No further simplifications are possible:

So the following set S4 is a minimal basis:

1. A->B
2. C->A
3. C->E
4. EF->C
5. EF->D
6. EF->G
7. GH->C
8. GH->D

(c)

First, we merge the FDs and called it S5:

A->B

C->AE

EF->CDG

GH->CD

The set of relations are:

R1 = (A, B) R2 = (A, C, E) R3 = (C, D, E, F, G) R4 = (C, D, G, H)

Since there is no super key in our relations, so we need to create a new relation:

R5 = (E,F,H).

So R1,R2,R3,R4,R5 consist our final answer.

(d)Yes，Since we formed each relation from an FD, the LHS of those FDs are superkeys for their relations. However, there may be other FDs that violate BCNF and therefore allow redundancy. For example: C->AE, so if we project the FDs on R3, C->E, C+ = CE but C is not a superkey. So these schema allow redundancy.