Assignment 08

Name: Jingwei Ye

NetID: jy3555

Q1. You are given a relational schema R = ABCDE satisfying the set of FDs □:

1. AC -> E
2. AC->CB
3. E->DE
4. D->B

Goal: Compute a minimal cover for the above set

We first put the FD's in a standard form and get:

1. AC->E
2. AC->C
3. AC->B
4. E->D
5. E->E
6. D->B

We first remove the "defective" parts and get:

1. AC->E
2. AC->B
3. E->D
4. D->B

Then we apply union on 1 and 2 and get:

1. AC->EB
2. E->D
3. D->B

Next we compute the closure and check if we can simplify 1.

we attempt to simplify AC->EB to AC->E,

then in New, we have (AC)+ = ACEDB, since it contains B, we can simplify AC->EB to AC->E

we get:

1. AC->E
2. E->D
3. D->B

No further simplifications can be made

our minimal cover is

New

1. **AC->E**
2. **E->D**
3. **D->B**

Q2.

You are given a relational schema R = ABCDEF satisfying the set of FDs:

```
1. AB->C
2. A->E
3. DE->F
4. E->EF
5. F->B
```

Goal: Compute a minimal cover for the above set

We first remove the "defective" parts and get:

```
1. AB->C
2. A->E
3. DE->F
4. E->F
5. F->B
```

Then we compute the closure and try to minimize the LHS:

(A)+ = AEFBC, hence we can simplify AB->C to A->C

our minimal cover is

New:

1. **A->C**
2. **A->E**
3. **DE->F**
4. **E->F**
5. **F->B**

Q3. You are given a relational schema R = ABCDEFGH satisfying the set of FDs:

```
1. AB->CD
2. C->EF
3. E->C
4. F->G
```

(a) Prove that the given set of FDs is already a minimal cover. To do that show that no simplification is possible. Attempt all simplifications.

We firstly attempt to simplify 1 and propose new as follows:

Old	New
1. AB->CD	1. AB->C
2. C->EF	2. C->EF
3. E->C	3. E->C
4. F->G	4. F->G

We attempt to simplify 1. in **Old** getting 1. in **New**. **New** could only be weaker. So to prove equivalence we need to prove 1. in **Old** from all in **New**

We compute (AB)+ = ABC, (AB)'s CD is not in ABC, we proved non-equivalence. So we continue with **Old** (and do not replace by **New**).

Then we consider the following simplification and check if it is possible:

Old	New
1. AB->CD	1. AB->D
2. C->EF	2. C->EF
3. E->C	3. E->C
4. F->G	4. F->G

We attempt to simplify 1. in **Old** getting 1. in **New**. **New** could only be weaker. So to prove equivalence we need to prove 1. in **Old** from all in **New**

We compute (AB)+ = ABD, (AB)'s CD is not in ABD, we proved non-equivalence. So we continue with **Old** (and do not replace by **New**).

Next, consider:

Old	New
1. AB->CD	1. A->CD
2. C->EF	2. C->EF
3. E->C	3. E->C
4. F->G	4. F->G

We attempt to simplify 1. in **Old** getting 1. in **New**. **New** could only be stronger. So to prove equivalence we need to prove 1. in **New** from all in **Old**

We compute (A)+ = ACD, it contains CD, however, we cannot make a stronger FD without the proper existing FDs. That is to say, we cannot infer A->CD from AB->CD without having A->B in **Old**. So we continue with **Old** (and do not replace by **New**).

Similarly, consider:

Old	New
1. AB->CD	1. B->CD
2. C->EF	2. C->EF
3. E->C	3. E->C
4. F->G	4. F->G

We attempt to simplify 1. in **Old** getting 1. in **New**. **New** could only be stronger. So to prove equivalence we need to prove 1. in **New** from all in **Old**

We compute (B)+ = BCD, it contains CD, however, we cannot make a stronger FD without the proper existing FDs. That is to say, we cannot infer B->CD from AB->CD without having B->A in **Old**. So we continue with **Old** (and do not replace by **New**).

Next, we attempt to simplify 2 in Old.

Old	New
1. AB->CD	1. AB->CD
2. C->EF	2. C->E
3. E->C	3. E->C
4. F->G	4. F->G

We attempt to simplify 2. in **Old** getting 2. in **New**. **New** could only be weaker. So to prove equivalence we need to prove 2. in **Old** from all in **New**

We compute (C)+ = CE, C's EF is not in CE, we proved non-equivalence. So we continue with **Old** (and do not replace by **New**).

Then we consider:

Old	New
1. AB->CD	1. AB->CD
2. C->EF	2. C->F
3. E->C	3. E->C
4. F->G	4. F->G

We attempt to simplify 2. in **Old** getting 2. in **New**. **New** could only be weaker. So to prove equivalence we need to prove 2. in **Old** from all in **New**

We compute (C)+ = CF, C's EF is not in CF, we proved non-equivalence. So we continue with **Old** (and do not replace by **New**).

By exhausting all possible simplifications, we observe that no further simplifications can be made from the original set of FDs, so we conclude that the give set of FDs is already a minimal cover.

(b) Produce a decomposition satisfying the conditions for Output in Section 3.2.

From the above minimal cover, we get the following tables/relations

1. (AB)CD
2. CEF
3. EC
4. FG

We now proceed to get a global key for R. Perhaps one of the 4 tables already includes a global key for R. We compute

1. (ABCD)+ = ABCDEFG
2. (CEF)+ = CEF
3. (EC)+ = ECFG
4. (FG)+ = FG

but we need R = ABCDEFGH

We examine all the attributes of R accounting for all the FDs that are satisfied.

We classify the attributes based on where they appear in FDs:

1. on both sides: C, E, F
2. on left side only: AB
3. on right side only: G, D
4. Nowhere: H

ABH appear in every key, G,D do not appear in any key. C,E,F may appear in a key.

We start with ABHCEF, which must contain a key. We cannot remove ABH but may try to remove C,E, and F.

We attempt to remove C. We compute (ABHEF) + = ABCDEFGH and we continue with ABHEF

We attempt to remove E. We compute (ABHF)+ = ABCDEFGH and we continue with ABHF

We attempt to remove F. We compute (ABH)+ = ABCDEFGH and we continue with ABH

As nothing else can be done, ABH is a global key.

So our final decomposition is

1. (AB)CD
2. CEF
3. EC
4. FG
5. ABH