COMP9311 - Database System

Assignment 2

Question 1:

R (A, B, C, D, E, G, H, I, J)

𝐹 = { 𝐴𝐷 → 𝐵, --- (1)

𝐵𝐷 → 𝐺, --- (2)

𝐵𝐸 → 𝐼, --- (3)

𝐴𝐸 → 𝐷𝐼, --- (4)

𝐴𝐼 → 𝐸, --- (5)

𝐴𝐸𝐼 → 𝐶}. --- (6)

***(1)*** To Prove🡪 𝐴𝐵 → 𝐺

AB+ = AB  
  
Since G is not present in the closure of AB, it is not derivable, or it cannot be proved.

***(2)*** Find all the candidate keys for R.

{A, H, J} will be present in every candidate key as none of them are dependent on another attribute (i.e. they are independent attribute)

{**A, E, H, J}+ = {A, B, C, D, E, G, H, I, J}** It is a candidate key.

**{A, I, H, J}+ = {A, B, C, D, E, G, H, I, J}** It is a candidate key.

{A, E, I, H, J}**+** = {A, B, C, D, E, G, H, I, J}

{A, E, I, H, J} is not a candidate key but a super key since it is the union of both the candidate keys. Also, candidate keys are supposed to be minimal basically, if E or I is taken out, it will result in becoming a candidate key.

(3) Determine the highest normal form of R with respect to F.

Using CK = {{A, E, H, J}, {A, I, H, J}}

Prime attributes = {A, E, I, H, J}

Non-prime attributes = {B, C, D, G}

The highest form for relation R is First Normal Form (1NF) since all the attributes are atomic.

The relation is not in 2NF because {A, E} being a proper subset of candidate key {A, E, H, J} is determines a non-prime attribute D in functional dependency 𝐴𝐸 → 𝐷𝐼.

Meaning D is partially dependent on candidate key {A, E, H, J}. This violates the rule of 2NF where a relation schema R is in 2NF if every nonprime attribute A in R is not partially dependent on any key of R.

(4) Find a minimal cover Fm for F

Step 1: Reduce Right

F’ = {AD 🡪 B, BD 🡪 G, BE 🡪 I, AE 🡪 D, AE 🡪 I, AI 🡪 E, AEI 🡪 C}

Step 2: Reduce Left

For AD 🡪 B: D+ = {D} meaning D 🡪 B is not inferred by F’ thus it can’t replace AD 🡪 B

A+ = {A} meaning A 🡪 B is not inferred by F’ thus it can’t replace AD 🡪 B

For BD 🡪 G: D+ = {D} meaning D 🡪 G is not inferred by F’ thus it can’t replace BD 🡪 G

B+ = {G} meaning B 🡪 G is not inferred by F’ thus it can’t replace BD 🡪 G

For BE 🡪 I: E+ = {E} meaning E 🡪 I is not inferred by F’ thus it can’t replace BE 🡪 I

B+ = {B} meaning B 🡪 I is not inferred by F’ thus it can’t replace BE 🡪 I

For AE 🡪 D: A+ = {A} meaning A 🡪 G is not inferred by F’ thus it can’t replace AE 🡪 D

E+ = {E} meaning E 🡪 G is not inferred by F’ thus it can’t replace AE 🡪 D

For AE 🡪 I: E+ = {E} meaning E 🡪 I is not inferred by F’ thus it can’t replace AE 🡪 I

A+ = {A} meaning A 🡪 I is not inferred by F’ thus it can’t replace AE 🡪 I

For AI 🡪 E: I+ = {I} meaning I 🡪 E is not inferred by F’ thus it can’t replace AI 🡪 E

A+ = {A} meaning A 🡪 E is not inferred by F’ thus it can’t replace AI 🡪 E

For AEI 🡪 C: EI+ = {E, I} meaning EI 🡪 C is not inferred by F’ thus it can’t replace AEI 🡪 C

AE+ = {A, E, I, D, B, G}, AE 🡪 C is not inferred by F’, thus it can’t replace AEI 🡪 C

F’’ = {AD 🡪 B, BD 🡪 G, BE 🡪 I, AE 🡪 D, AE 🡪 I, AI 🡪 E, AEI 🡪 C}

Step 3: Reduce Redundancy

AD+ | F’’ – {AD 🡪 B} = {A, D}; AD 🡪 B is not redundant since it’s not inferred by F’’ – {AD 🡪 B}

BD+ | F’’ – {BD 🡪 G} = {B, D}; BD 🡪 G is not redundant since it’s not inferred by F’’ – {BD 🡪 G}

BE+ | F’’ – {BE 🡪 I} = {B, E}; BE 🡪 I is not redundant since it’s not inferred by F’’ – {BE 🡪 I}

AE+ | F’’ – {AE 🡪 D} = {A, E, I}; AE 🡪 D is not redundant since it’s not inferred by F’’ – {AE 🡪 D}

AE+ | F’’ – {AE 🡪 I} = {A, E, D, B, G, I, C}; AE 🡪 I is redundant, so it can be removed from F’’ to get F’’’

AI+ | F’’ – {AI 🡪 E} = {A, I}; AI 🡪 E is not redundant since it’s not inferred by F’’’ – {AI 🡪 E}

AEI+ | F’’ – {AEI 🡪 C} = {A, E, I, D, B, G}; AEI 🡪 C is not redundant since it’s not inferred by F’’’ – {AEI 🡪 C}

F’’’ = {AD 🡪 B, BD 🡪 G, BE 🡪 I, AE 🡪 D, AI 🡪 E, AEI 🡪 C}

(5) Regarding F, does the decomposition R1 = {ABCDJ}, R2 = {BDGI}, R3 = {BCEH} of R satisfy the lossless join property?

R1 = {ABCDJ},

R2 = {BDGI},

R3 = {BCEH}

For BD 🡪 G

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | ***A*** | ***B*** | ***C*** | ***D*** | ***E*** | ***G*** | ***H*** | ***I*** | ***J*** |
| ***R1*** | a | a | a | a | b | ~~b~~ a | b | b | a |
| ***R2*** | b | a | b | a | b | a | b | a | b |
| ***R3*** | b | a | a | b | a | b | a | b | b |

R1, R2, R3 is not a lossless join since neither row is completely filled with a’s

(6) Provide a step-by-step lossless decomposition of R into BCNF normal form

R (A, B, C, D, E, G, H, I, J)

F = {AD 🡪 B, BD 🡪 G, BE 🡪 I, AE 🡪 DI, AI 🡪 E, AEI 🡪 C}

aeihj

R1 (A, E, D, I) 🡺 with functional dependencies AE 🡪 DI, AI 🡪 E with key {AE}

AI 🡪 E violates the BCNF as AI is not a super key.

R2 (A, B, C, E, F, G, H, J) 🡺 with key {ADEI }

R1(adb)

R2(acefghj)

R21()

Question 2:

Question 3:

P1, P2, P1, P4, P3, P7, P2, P1, P4, P5, P8, P6, P8, P2, P8

1. **LRU**

Initial Empty Buffer Pool 🡪 [empty, empty, empty]

P1: Cache Miss 🡪 [**P1**, empty, empty]

P2: Cache Miss 🡪 [P1, **P2**, empty]

P1: Cache Hit 🡪 [**P1**, P2, empty]

P4: Cache Miss 🡪 [P1, P2, **P4]**

P3: Cache Miss 🡪 [P1, **P3**, P4]

P7: Cache Miss 🡪 [**P7**, P3, P4]

P2: Cache Miss 🡪 [P7, P3, **P2]**

P1: Cache Miss 🡪 [P7, **P1**, P2]

P4: Cache Miss 🡪 [**P4,** P1, P2]

P5: Cache Miss 🡪 [P4, P1, **P5**]

P8: Cache Miss 🡪 [P4, **P8**, P5]

P6: Cache Miss 🡪 [**P6**, P8, P5]

P8: Cache Hit 🡪 [P6, **P8**, P5]

P2: Cache Miss 🡪 [P6, P8, **P2**]

P8: Cache Hit 🡪 [P6, **P8,** P2]

cache hits = 3

cache misses = 12

Hit rate = #cache hits / (#cache hits + #cache misses)

Hit rate = 3/ (3 + 12) = **0.2** or **20%**

1. **MRU**

P1, P2, P1, P4, P3, P7, P2, P1, P4, P5, P8, P6, P8, P2, P8

Initial Empty Buffer Pool 🡪 [empty, empty, empty]

P1: Cache Miss 🡪 [**P1**, empty, empty]

P2: Cache Miss 🡪 [P1, **P2**, empty]

P1: Cache Hit 🡪 [**P1**, P2, empty]

P4: Cache Miss 🡪 [P1, P2, **P4]**

P3: Cache Miss 🡪 [P1, P2, **P3**]

P7: Cache Miss 🡪 [P1, P2, **P7**]

P2: Cache Hit 🡪 [P1, **P2**, P7**]**

P1: Cache Hit 🡪 [**P1**, P2, P7]

P4: Cache Miss 🡪 [**P4,** P2, P7]

P5: Cache Miss 🡪 [**P5**, P2, P7]

P8: Cache Miss 🡪 [**P8**, P2, P7]

P6: Cache Miss 🡪 [**P6**, P2, P7]

P8: Cache Hit 🡪 [**P8**, P2, P7]

P2: Cache Hit 🡪 [P8, **P2**, P7]

P8: Cache Hit 🡪 [**P8**, P2, P7]

cache hits = 6

cache misses = 9

Hit rate = #cache hits / (#cache hits + #cache misses)

Hit rate = 6/ (6 + 9) = **0.4** or **40%**

1. **FIFO**

P1, P2, P1, P4, P3, P7, P2, P1, P4, P5, P8, P6, P8, P2, P8

Initial Empty Buffer Pool 🡪 [empty, empty, empty]

P1: Cache Miss 🡪 [**P1**, empty, empty]

P2: Cache Miss 🡪 [P1, **P2**, empty]

P1: Cache Hit 🡪 [**P1**, P2, empty]

P4: Cache Miss 🡪 [P1, P2, **P4]**

P3: Cache Miss 🡪 [**P3**, P2, P4]

P7: Cache Miss 🡪 [P3, **P7**, P4]

P2: Cache Miss 🡪 [P3, P7, **P2]**

P1: Cache Miss 🡪 [**P1**, P7, P2]

P4: Cache Miss 🡪 [P1, **P4,** P2]

P5: Cache Miss 🡪 [P1, P4, **P5**]

P8: Cache Miss 🡪 [**P8**, P4, P5]

P6: Cache Miss 🡪 [P8, **P6**, P5]

P8: Cache Miss 🡪 [P8, P6, **P8**]

P2: Cache Miss 🡪 [**P2**, P6, P8]

P8: Cache Hit 🡪 [P2, P6, **P8**]

cache hits = 1

cache misses = 14

Hit rate = #cache hits / (#cache hits + #cache misses)

Hit rate = 1/ (1 + 14) = **0.0667** or **6.66667%**