NAME: SUCHETA JHUNJHUNWALA

ROLL No: MDS202151

RDBMS and SQL FINAL EXAM.

1) R(A, B, C, D, E) is split into

R1(A, D, E) and R2(A, B, C, D).

$\{A \rightarrow E, C \rightarrow BD, E \rightarrow AB, AB \rightarrow C\}$

A decomposition is said to be lossless if the natural join of R1 and R2 gives back the original relation R

$A^+ = AEBCD = ABCDE$

$B^+ = B$

$C^+ = CBD$

$D^+ = D$

$E^+ = EABCD = ABCDE$.

∴ A and E are the candidate keys for R.

Note that, $R1 \cup R2 = R$ since all the attributes of R are present in either R1 or in R2.

To show that this decomposition is lossless we need to show either of the following:

- $R1 \cap R2 \rightarrow R1$
- $R1 \cap R2 \rightarrow R2$

$R1 \cap R2 = \{A, D\}$

Since A is a candidate key, $R1 \cap R2 \rightarrow R1$ and $R1 \cap R2 \rightarrow R2$ as well. ∴ This is a lossless decomposition.

2) $R(A, B, C, D, E, F, G, H, I)$ .

$F = \{BC \to GHI, AD \to E, A \to H, E \to BCF, G \to H\}$

Any relation R is in BCNF if for any functional dependency of the form $X \to Y$, at least one of the following holds:

- $X \to Y$ is trivial i.e. $Y \subseteq X$
- X is a super key.

We shall first find the keys of R.

$(BC)^+ = BCGHI$

$(AD)^+ = ADEBCFGHI = ABCDEFGHI$ .

$(AE)^+ = AEBCFGHI$ .

Note that none of the single attributes form a key, Also, since AD are not present on the right hand side of any of the functional dependencies they must be a part of the key. The smallest such possible key is AD itself. Therefore, AD is a candidate key.

Super key is formed by including any combination of B,C, E, F, G, H and I to AD .

Here we can see, $BC \to GHI$ has BC on the left hand side and it is not a super key since $AD \not\subseteq BC$ and also $GHI \not\subseteq BC$ so this is not a trivial functional dependency.

$\therefore$ R is not in BCNF .

The second functional dependencies,

AD → E has AD as the super key.

A → H has neither A to be the super key nor $H \not\subseteq A$ so it violates BCNF.

E → BCF also violates BCNF.

G → H also.

∴ BC → GHI, E → BCF, G → H violate BCNF.

We shall first split R into R1(A, D, E, F) and R2 (B, C, G, H, I) due to the violation by BC → GHI.

Since E → BCF split R1 further into R11 (A, D) and R12(E, F).

Since G → H split R2 (B, C, G, H, I) into R21 (B, C, I) and R22 (G, H)

∴ R is split into

R11(A, D), R12(E, F), R21(B, C, I), R22(G, H).

3) a) $A1 \subset A2 \subset A3$ .

$\pi_{A1}(\pi_{A2}(\pi_{A3}(\sigma_{F1}(\sigma_{F2}(r)))))$ .

(1) $\pi_{A1}(\sigma_{F1 \wedge F2}(r))$

(3) $\pi_{A1}(\pi_{A2}(\sigma_{F1 \wedge F2}(r)))$ .

b) $r1(\underline{P}, Q, R)$ and $r2(\underline{R}, S, T)$ .

r1 has 1500 tuples , r 2 has 1000 tuples .

The maximum size of the join $r1 \bowtie r2$ is $\underline{1500}$

4) key size = 6 bytes

record size = 32 bytes

no. of records = 32768 .

block size = 2048 bytes

pointer size = 10 bytes .

Each block has 2048 bytes and has records of size 32 bytes.

∴ No. of records per block = 2048 ÷ 32 = 64 .

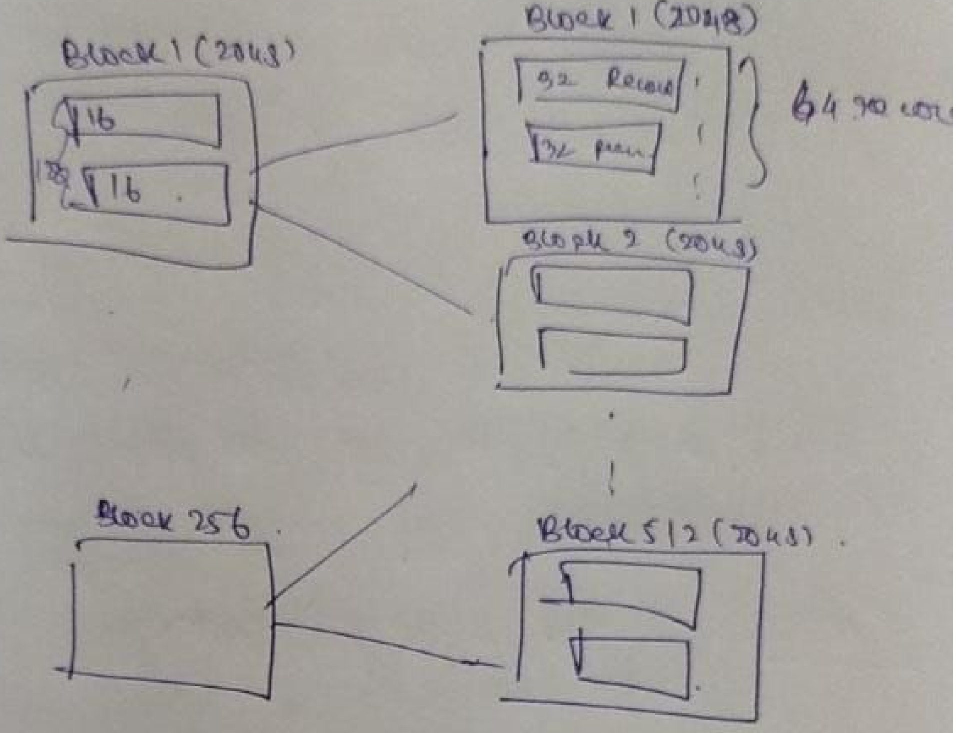
Number of blocks = 32768 ÷ 64 = 512

so we have 512 blocks in the first level .

Now the record for index is of size (6 + 10) bytes = 16 bytes .

∴ Number of index records in each block = 2048/16

= 128.



Since there were 64 records in each block in the first level and now in the second level each block can store 128 records ⇒ each block in the second level can point to two blocks in the first level.

⇒ no. of blocks in the second level = 512/2 = 256

~~[illegible]~~ ~~[illegible]~~

∴ Number of blocks in the first level = ~~512~~ 256.

~~[illegible] the second level = 256.~~

~~do: (512, 256)~~.

Continuing further number of records in second level = 256/128 = 2.

where 128 is the number of records in each block for index level 2 and 256 is the total number of records.

∴ Ans : (256, 2).

5) search key → 12 bytes

pointer → 6 bytes.

block size → 528 bytes.

Let the maximum order of the $B^+$ tree is n.

any $B^+$ tree of order n will have n pointers and (n-1) search keys.

$\therefore$ Total size $= n \times 6 + (n-1) \times 12$.

$= 6n + 12n - 12 = 18n - 12$.

Now, $18n - 12 \leq 528$.

$\Rightarrow 18n \leq 540$

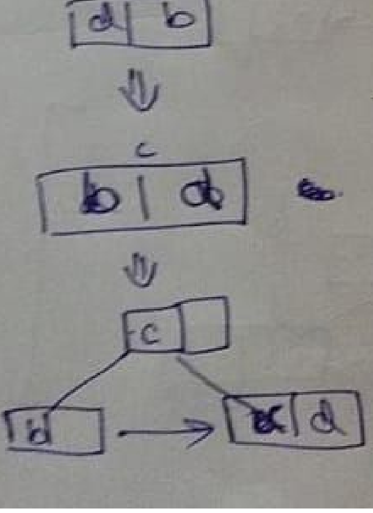
$\Rightarrow n \leq 30$

∴ the maximum order of the $B^+$ tree is 30.

6) Fan-out factor = 3

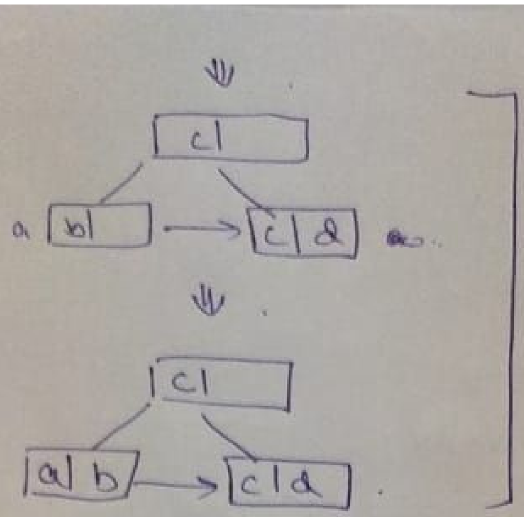
⇒ no. of values stored = 2.

we are inserting, d, b, c, a, e, f, g, g, g, h.

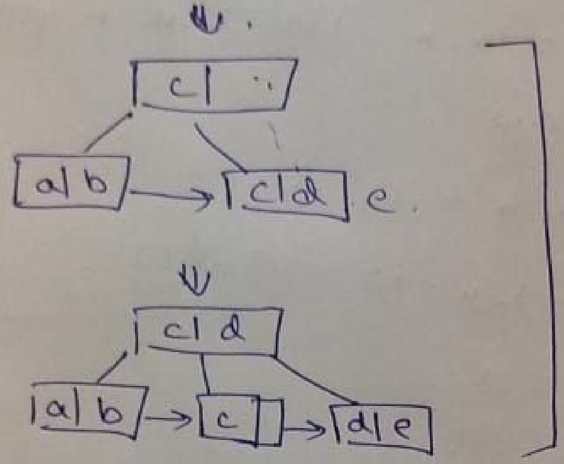


on inserting d and b.

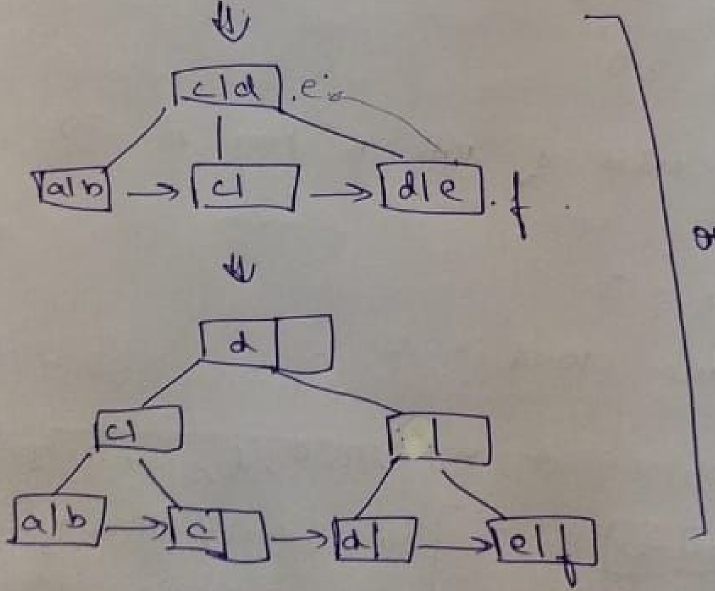
on inserting c.



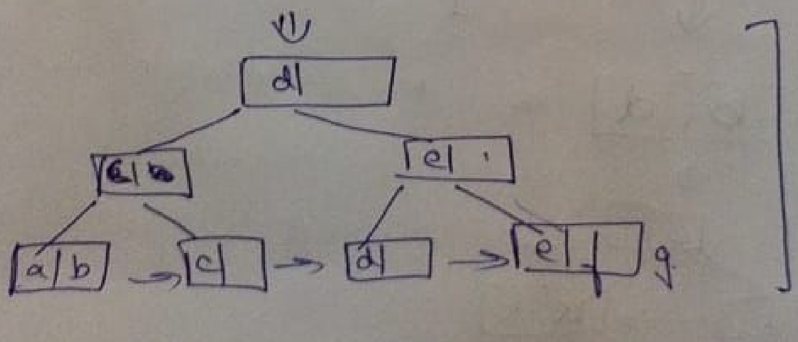
On inserting a .



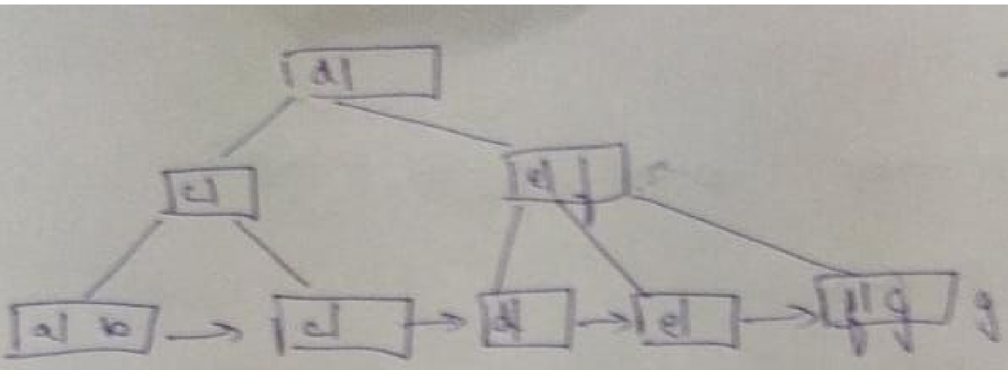
on inserting e .

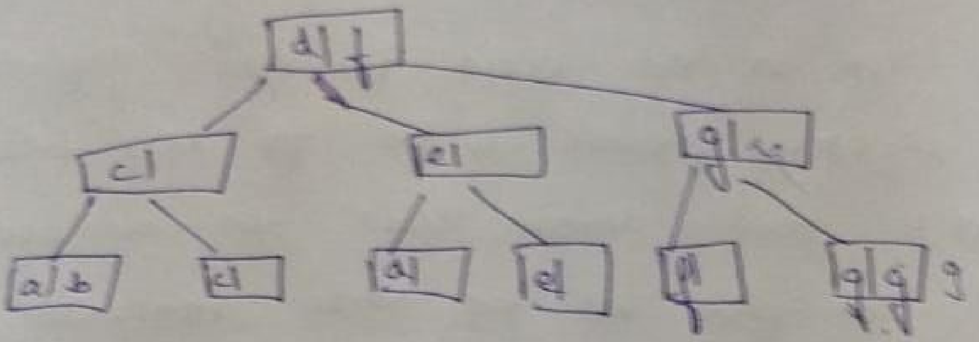


on inserting f

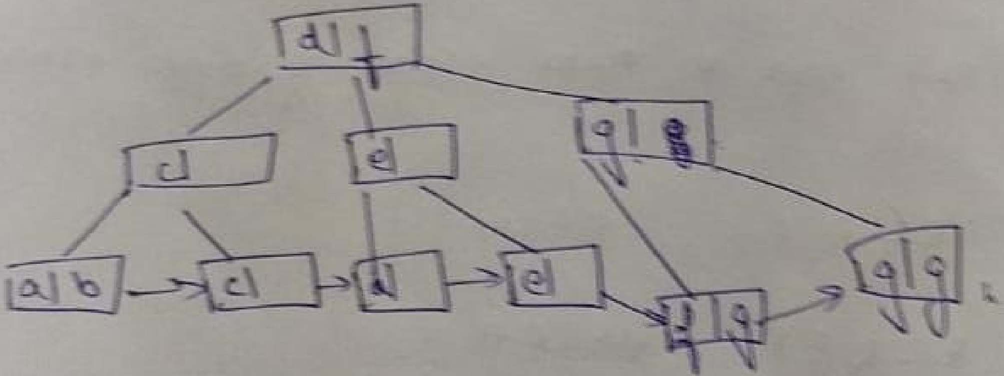


on inserting g

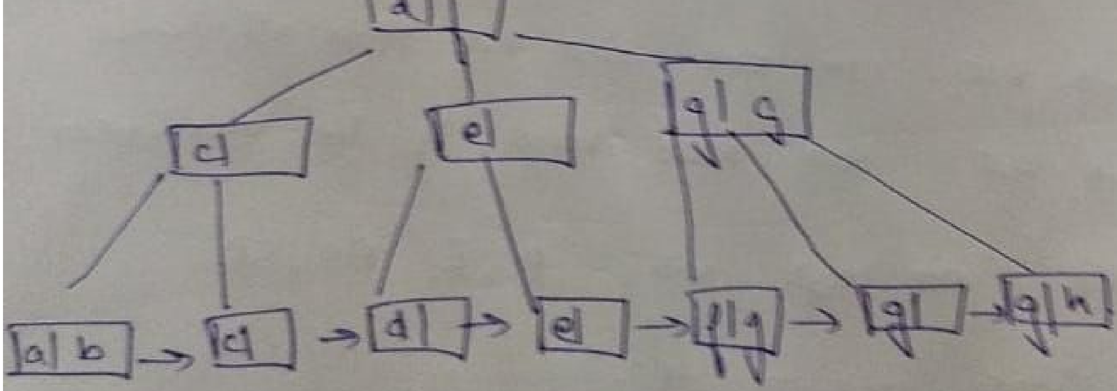




On inserting g.



On inserting g.



On inserting h.

The above is the final $B^+$ tree

7) R(A, B, C, D, E).

$F = \{A \to BC, CD \to E, B \to D, E \to A\}$.

$A^+ = ABCDE$.

$B^+ = BD$

$E^+ = EABCD \therefore = ABCDE$.

Note that for all the above functional dependencies the only sigletons present on the left side are A, B and E and among these we can see A and E can give all the other attributes.

∴ A and E are the candidate keys.

Two candidate keys exist of single attributes

Also,

$(CD)^+ = CDEABC \therefore = ABCDE$

$(BC)^+ = BCDEA = ABCDE$.

$(BD)^+ = BD$

∴ CD and BC can also give the other attributes.

⇒ CD and BC are also keys of two variables.

∴ candidates keys are A, E, BC, CD.

No. of candidate keys = 4.

8) ~~S1 : R1(A) r~~

8) S1 : r1(A) r2(A) w1(A) r2(B)

S2 : r1(A) w1(A) r2(A) w2(B) w2(B) .

| T1 | T2 |
|---|---|
| r1(A) | |
| | r2(A) |
| w1(A) | |
| | r2(B) |

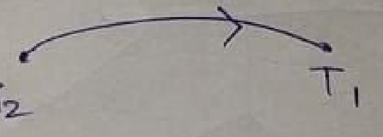
S1

| T1 | T2 |
|---|---|
| r1(A) | |
| w1(A) | |
| | r2(A) |
| | w2(B) |
| | w2(B) |

S2 .

Two instructions conflict if either of them is write in the same variable .

Since in S1 , we have r2(A) and w1(A) conflicting , the precedence graph for S1 is

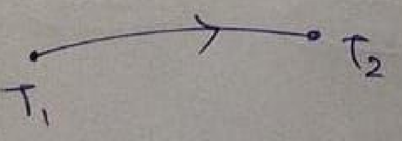


There are no other conflicts. This graph is acyclic

⇒ S1 is conflict serializable .

In S2, the instructions are already in serial order.

∴ S2 is a serial schedule. If we draw the precedence graph for S2 it is ,



Since w1(A) and $r_2$(A) are the conflicts .

9) Let the two entities be movies and users.

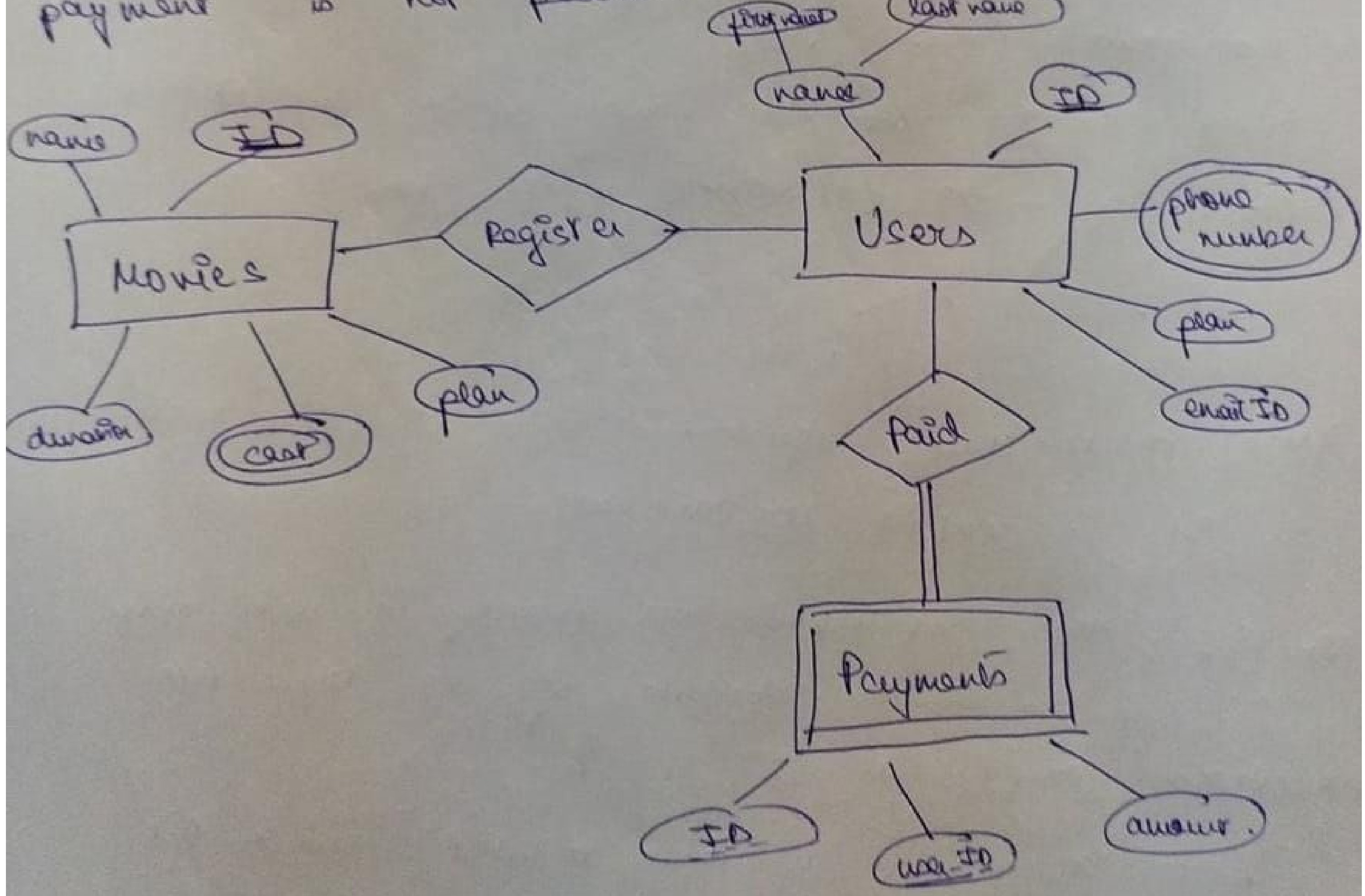
Let a weak entity be payments.

The entity movies has the attributes name, ID, duration, cast and the plan.

Users has the attributes name, ID, phone number, plan, email_id.

Payments has the attributes ID, user-ID, amount.

We assume payments to be a weak entity since if a user does not register for any plan then payment is not possible.



users register for Movies via the relationship 'register' and ~~users~~ pay the plan amount via the relationship 'paid'.

(2) name ID duration cast

(2) The relationship schema is :

Movies ( ID, name, duration, plan )

cast ( ID, cast )

Users ( ID, first name, last name, plan, email(ID)

phone number( user-ID , phone number )

Payments ( ID , user-ID , amount)

Movies

| ID | name | duration | plan |
|---|---|---|---|
| 143 | KPHG | 146 mins | B500 |
| 121 | Harry | 180 mins | C1500 |

cast

| ID | cast |
|---|---|
| 143 | Vicky |
| 143 | Alen |
| 121 | Tom |
| 121 | Martin |

9) Users

| ID | first name | last name | plan | email ID |
|---|---|---|---|---|
| 4131 | Amir | Raj | B1500 | Ar @ gmail.co |
| 2408 | Surabhi | Saha | A500 | Ss @ gmail.co |

Phone numbers

| ID | phone number |
|---|---|
| 4131 | 98315440 |
| 4131 | 123489 |
| 2408 | 876521 |

Payments

| ID | user ID | amount |
|---|---|---|
| 416589 | 2408 | 500 |
| 241382 | 4131 | 1500 |

10) TRC :

$\{ t \mid \exists s \in \text{instruction} \; ( s[ID] = t[ID] \wedge s[name] = t[name]$

$\wedge \neg ( \exists d \in \text{instruction} ( d[salary] > s[salary]$

$\wedge s[department] = \text{'biology'} ) \}$

DRC :

$\{ <n> \mid <i, n, s, d> \in \text{instructor} \wedge d = \text{'Biology'}$

$\wedge \neg$

11) B let schema be,

table ( , student-ID, instructor-ID, course-name ),

| course-ID | student-ID | instructor-ID | course-name |
|---|---|---|---|
| C101 | 4123 | AA | data |
| C101 | 1234 | AA | data |
| C101 | 1567 | AA | data. |
| A121 | 2436 | SD | algebra |
| A121 | 4378 | SD | algebra |
| A121 | 0058 | SD | algebra. |

we can see student-ID → instructor-ID. since. the students have a specific instructor for the cours & these students have also taken a specific course.

There is no transitive dependency so it is in 3NF but not in 4NF since multivalue dependency is there.