DBMS - PAPER - 2019

Q1 (a) In the restaurant table

(i) Dish_Dec → Price
(ii) BILL_NO → Qty.

"Bill_NO → Qty" is violated because BILLNO cannot depend on Qty. Since when Qty = 2 we have different instances of BILLNO.

None of the dependency is violated.

(b) What are the four different types of database users. Specify there roles?

Ans. Two categories of Users: ① Actors on the scene
② Workers behind the scene

Actors on the scene:
① Database administrators ② Database designers
③ End users
3.1 Casual: occasionally access the database.
3.2 Naive or parademic end users: a large section of end-user population. eg. bank-tellers, Reservation clerks or sale staff.
They use previously well defined functions in the form of canned transactions.
3.3 Sophisticated end users: Include engineers, analysts, scientists who thoroughly familiarize themselves with the facilities of DBMS.
3.4 Standalone users: maintain personal databases using ready-to-use packaged applications.

(c) Justify that primary key cannot be NULL.

Primary key must be distinct and it is used to uniquely identify a record in a table.

Null cannot have any value, so it cannot be used as a primary key.

(d) Two transactions T_1 and T_2 ----

If two transactions are T_1 and T_2 are executing concurrently then either of $(x = x + 10)$ or $(x = x + y)$ will get executed first.

~~Case 1:~~ $x = x + 10$ get's executed first

$x = x + 10 = 50 + 10 = 60$

$x = 60$, but value has not yet been written so, when $(x = x + y)$ is executed, x would be equal to $x = 50 + 5 = 55$

~~Case 2~~: $(x = x + y)$ get's executed first

$x = x + y = 50 + 5 = 55$

But now $(x = x + 10)$ will override the values of x if it has not yet been written.

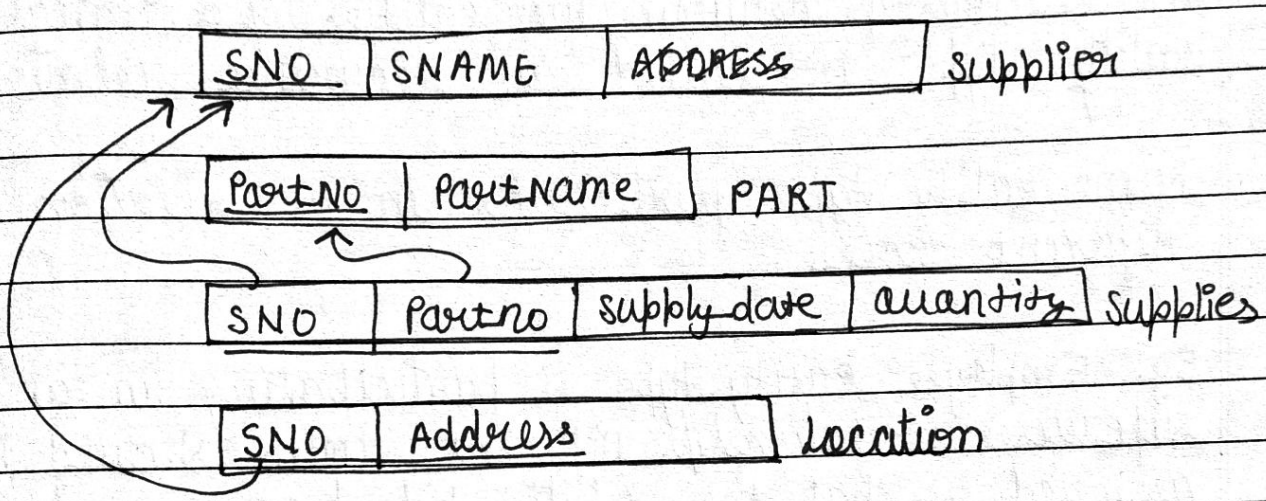
So, $x = x + 10 = 56 + 10 = 60$

~~In both cases~~ Answer is incorrect, correct answer would be $(x = x + 10 = 50 + 10 = 60$; ~~[redacted]~~)

The problem is either ~~[redacted]~~ T_2 is getting older value of x because they are executing concurrently.

This is called lost update problem.

(e) Map the following ER diagram to relation table.



(f) (i) $T_1 \bowtie_{T_1.X = T_2.C} T_2$

A	B	C	X	Y	Z
1	3	1	1	3	1
3	2	1	1	3	1

(ii) $T_1 \cap T_2$

A	B	C
1	3	1

(iii) $\Pi_{(T1.Z, T2.C)} (\sigma_{T1.Y = T2.B} (T_1 \times T_2))$

A	B	C	X	Y	Z
1	3	1	1	3	1

$\Rightarrow$

C	Z
1	1

(g) Why it is necessary to give role names in a recursive relationship.

A relationship between two entities of a similar entity type is called a recursive relationship.

Same entity type participates in a relation with different roles.

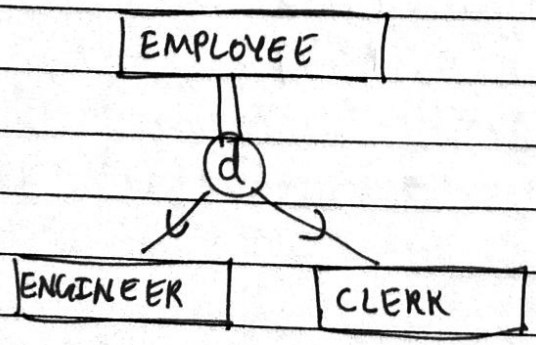
Eg. Employee entity type is participating in a supervises relationship. Different roles should be assigned so that the relationship becomes clear.

(h) Give SQL commands to create table T.

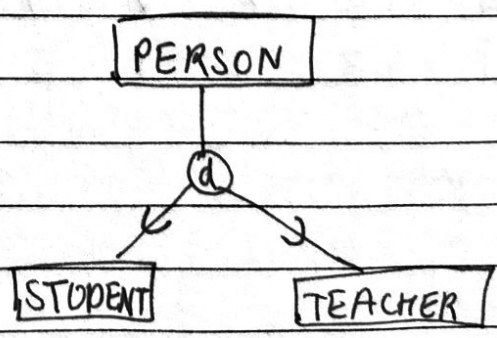
```
CREATE TABLE T (
    A int(10) NOT NULL,
    B varchar(50),
    C int DEFAULT 6,
    D int,
    PRIMARY KEY (A,B),
    FOREIGN KEY(D) REFERENCES S(E)
);
```

(i) Give EER diagram

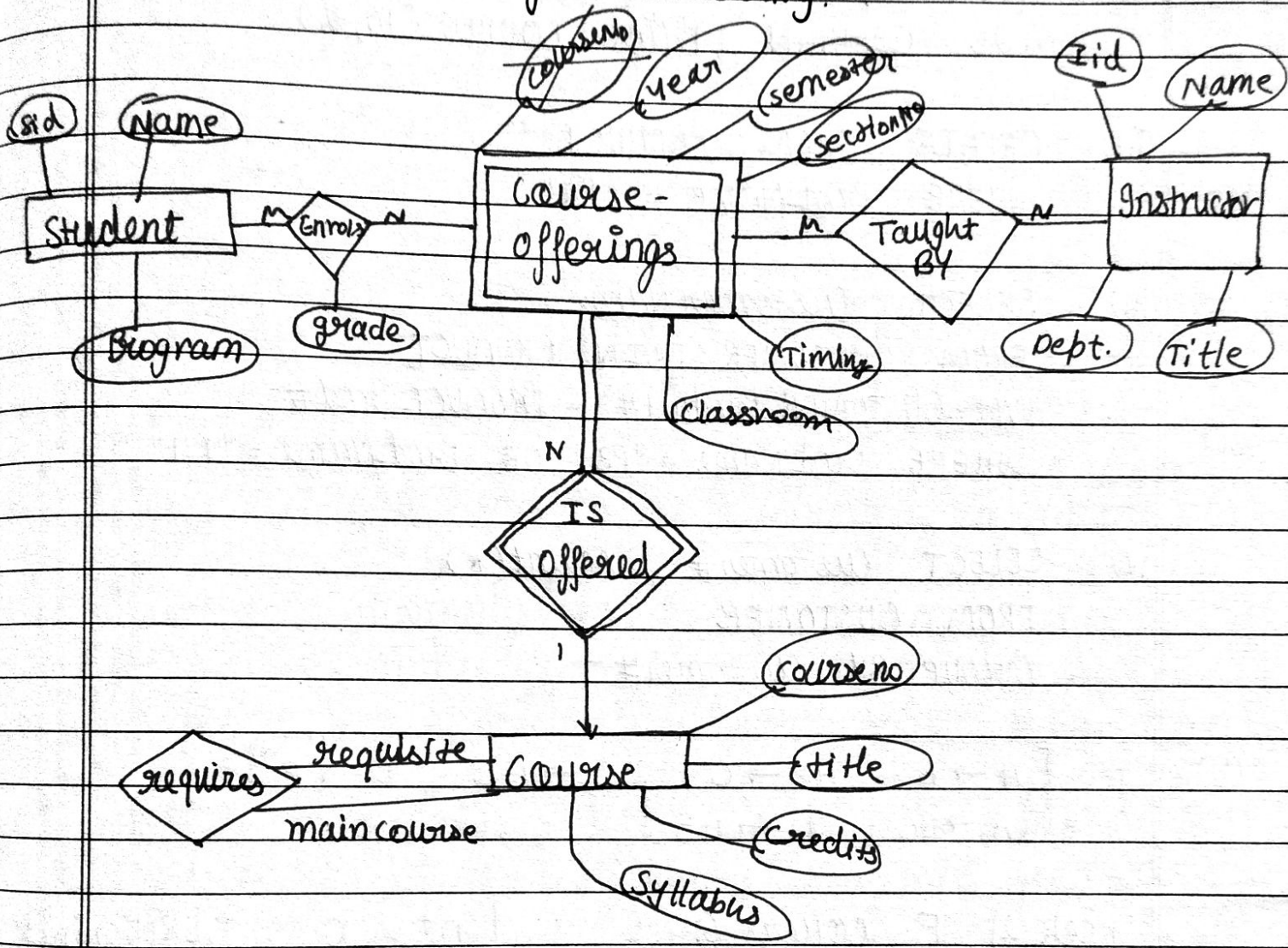
Total-disjoint specialization



Partial-disjoint specilization



Q2. Design ER schema for university.



student (studentid, name, program)
course (courseno, title, syllabus, credits)
course offering (courseno, secno, year, semester, time, room)
instructor (instructorid, name, dept, title)
enrols (studentid, courseno, secno, semester, year, grade)
teaches (courseno, secno, semester, year, instructorid)
requires (maincourse, prerequisite)

Q3 (a) (i) ALTER TABLE PRODUCT
ADD ~~COLUMN~~ Price DOUBLE (10, 2)

(ii) DELETE FROM PRODUCT
WHERE Prod_Name = "P2"

(iii) SELECT CustomerName
FROM CUSTOMER JOIN PRODUCT
ON CUSTOMER.Product# = PRODUCT.Prod#
WHERE Prod_Name = "P3" OR Prod_Name = "P4".

(iv) SELECT Customer#, count(*)
FROM CUSTOMER
GROUP BY Customer#

(b) $F = \{A \to B, AB \to C, D \to AC, D \to E\}$
$G = \{A \to BC, D \to AE\}$

Check if F covers G
$A^+ = A$ [Reflexive]
$A^+ = AB$ [Reflexive][$A \to B$]
$A^+ = ABC$ [$AB \to C$][
Hence $A \to BC$ is derived
$\Rightarrow$ F covers G.

$D^+ = D$ ∵[Reflexive]
$D^+ = DE$ ∵[$D \to E$]
$D^+ = DACE$ ∵[$D \to AC$]
Hence $D \to AE$ is derived.

Check if G covers F
$A \to BC$
$A \to B$ & $A \to C$ [decomposition]
$A \to B$ is derived.

$\Rightarrow$ F & G are equivalent.

$D \to AE$
$D \to A$ & $D \to E$ [Decomposition]
$D \to E$ is derived.
$D^+ = \{A\ E\ B\ C\}$
$D \to AC$ is derived.
$AB^+ = \{A\ B\ C\}$
$AB \to C$ is derived

Red Wings
Page No.__ Date __/__/__20__

Q4(a)

ID	Name	Course	PhoneNo	ProjectNo	ProjectName
01	Abhi	CS	99881	101	Face Mask
02	Aditi	CS	99882	101	Face Mask
03	Divya	CS	99883	NULL	NULL
04	Harsh	CS	99884	102	Sentiment analysis

Up. < Ins. < Del. <

(i) Updation anomaly: In a case when multiple students are enrolled in same project, changing the name of project requires a changes in multiple places. If it was done incorrectly then this may lead to inconsistency in database.

(ii) Insertion anomaly: In a case when inserting a student who is not yet assigned a project, the project field have to be set as NULL.

(iii) Deletion anomaly: Deleting a project detail may lead to the deletion of entire student record in a case when a single student is enrolled in a project.

(b) (i) 1 : 1 (Each college has exactly one principal)
(ii) M : N
(iii) M : N
(iv) M : N

ID, Name, Salary, JoiningDate, Department#

DepID, DName, Location, Mgr#

Q5. (i)

WORK_DEPT ← WORKER ⋈ Department# = DepID Department

WORK_DEPT_MGR ← WORK_DEPT ⋈ Mgr# = ID WORKER

R ← ΠWORKER.Name, DName, Worker.Name (WORK_DEPT_MGR)

OR

(i)

WORK_DEPT ← WORKER ⋈ Department# = DepID DEPARTMENT

R1(WName, DName, Mgr#) ← Π Name, DName, Mgr# (WORKDEPT)

R2 ← R1 ⋈ Mgr# = ID WORKER

R3 ← ΠWName, DName, Name (R2).

(ii) WORK_DEPT ← WORKER ⋈ (Department# = DepID) DEPARTMENT

R1 ← σ (DName = "Finance") (WORK_DEPT)

R2 ← fCOUNT(DName) (R1)

(iii) WORK_DEPT ← WORKER ⋈ (Department# = DepID) DEPARTMENT

R ← DName fCOUNT(DName) (WORK_DEPT)

(iv) WORK_DEPT ← WORKER ⋈ (Department# = DepID) DEPARTMENT

R ← DepID f Avg(Salary) (WORK_DEPT)

(v) R ← σ (JoiningDate)

R ← σ (YEAR(JoiningDate) > 2010) (WORKER)

Q6 (a) BOOKID$^+$ = { BOOKID } [Reflexive]

BOOKID$^+$ = { BOOKID, GenreID, Price } [FD's] [or Transitive]

BOOKID$^+$ = { BOOKID, GenreID, Price, GenreType } [Transitive]

(i) BookID can determine all the attributes, hence it can be used as a primary key.

(ii) There are no multivalue attributes hence the relation is in 1NF.

~~In GenreID → GenreType~~ It is also in 2NF because all Non-prime attributes are fully functionally dependent on primary key.

In GenreID → GenreType, GenreType is a non prime attribute which is being determined by a non prime attribute i.e GenreID Hence transitive dependency exists. The Relation is not in 3NF.

R_1 [BOOKID, GenreID, Price]

R_2 [GenreID, GenreType]

(b) $r = 30,000$
$B = 1024$ bytes
record size = 100 bytes.

(i) Blocking factor = $\frac{B}{RS} = \left\lfloor \frac{1024}{100} \right\rfloor = 10$

(ii) No. of blocks needed = $\frac{\text{Total records}}{\text{Blocking factor}}$

$= \frac{30,000}{10} = 3,000$ blocks

(iii) No. of block access = $\lceil \log_2 3,000 \rceil$ = 12

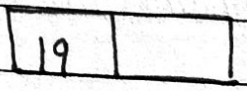
(iv) Index record size = 9 + 6 = 15 bytes
Blocking factor = $\left\lfloor \frac{1024}{15} \right\rfloor = 68$
No. of entries in index file = no. of blocks in data file = 3,000
No. of blocks needed = $\left\lceil \frac{3000}{68} \right\rceil$ = 45
No. of block access = $\lceil \log_2 45 \rceil + 1 = 6 + 1 = 7$ access

* Insertion and deletion is difficult in primary index.

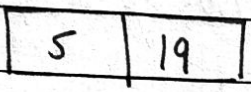
Red Wings
Page No.__ Date __/__/__20__

Q7(a) 19, 5, 12, 7, 40, 3, 15 For B Tree
order = p = 3

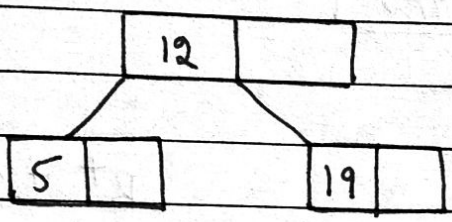
(i) 19



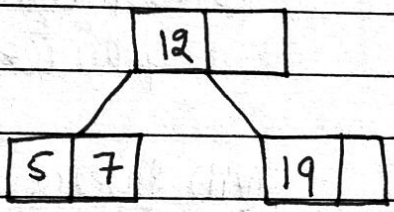
(ii) 5



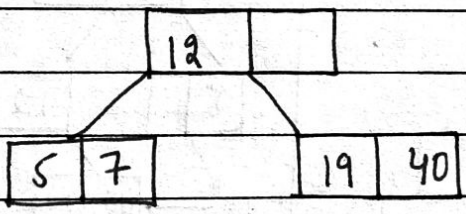
(iii) 12



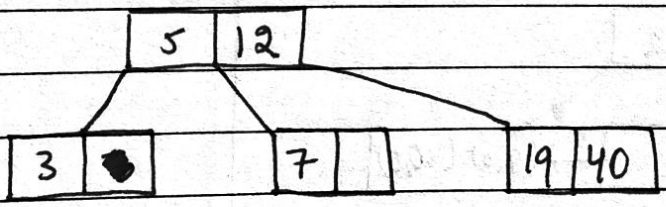
(iv) 7



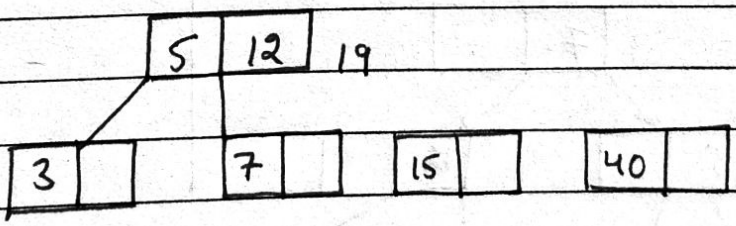
(v) 40

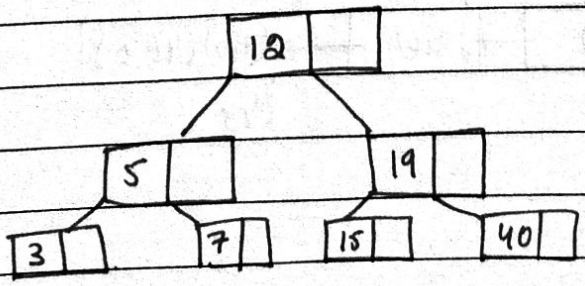


(vi) 3

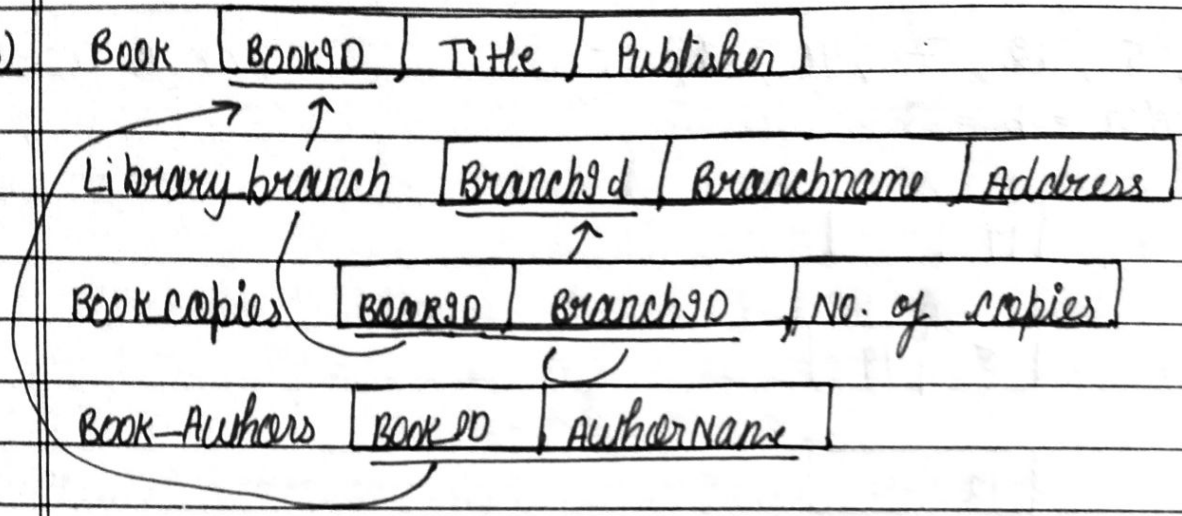


(vii) 15





Q7(b)



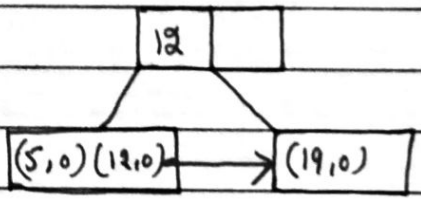
Q7(a)

(5,0) (19,0)

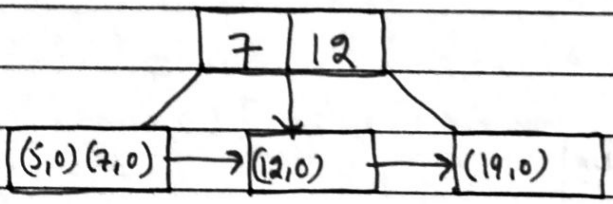
B+ tree

0 → represent data pointer

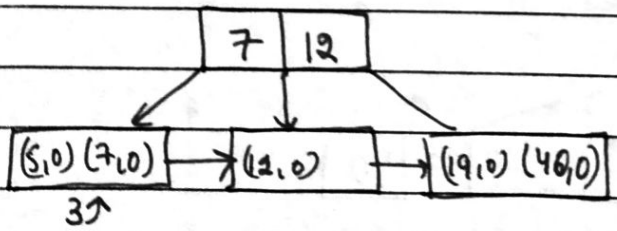
(ii) 12



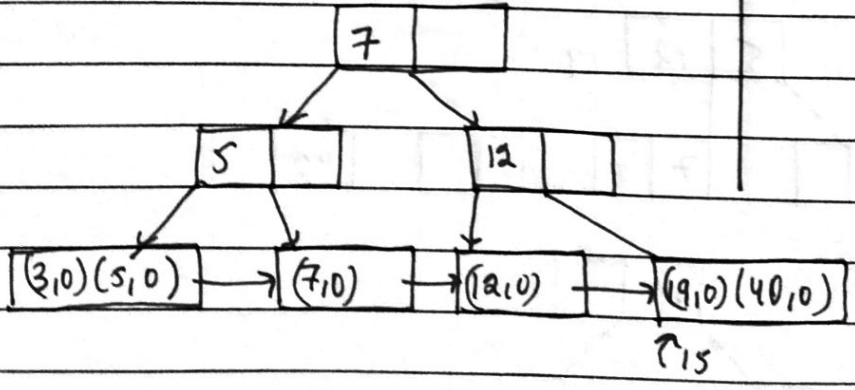
(iii) 7,



(iv) 40

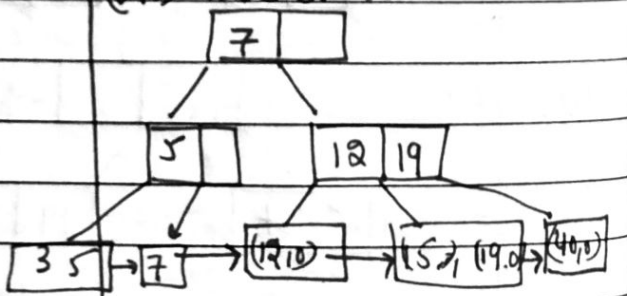


(v) 3



(vi)

(vii) Insert 15



PAPER - 2018

Q1(a) $F = \{ A \to B, AB \to C, D \to AC, D \to E \}$
$G = \{ A \to BC, D \to AE \}$

Check if F covers G:
Find closure of determining attributes of G from FD's of F.

$A^+ = \{ABC\} \because [A \to B, AB \to C]$
Hence $A \to BC$ is covered

$D^+ = \{DACEB\} \because [D \to AC, D \to E]$
Hence $D \to AC$ is covered
$\Rightarrow$ F covers G

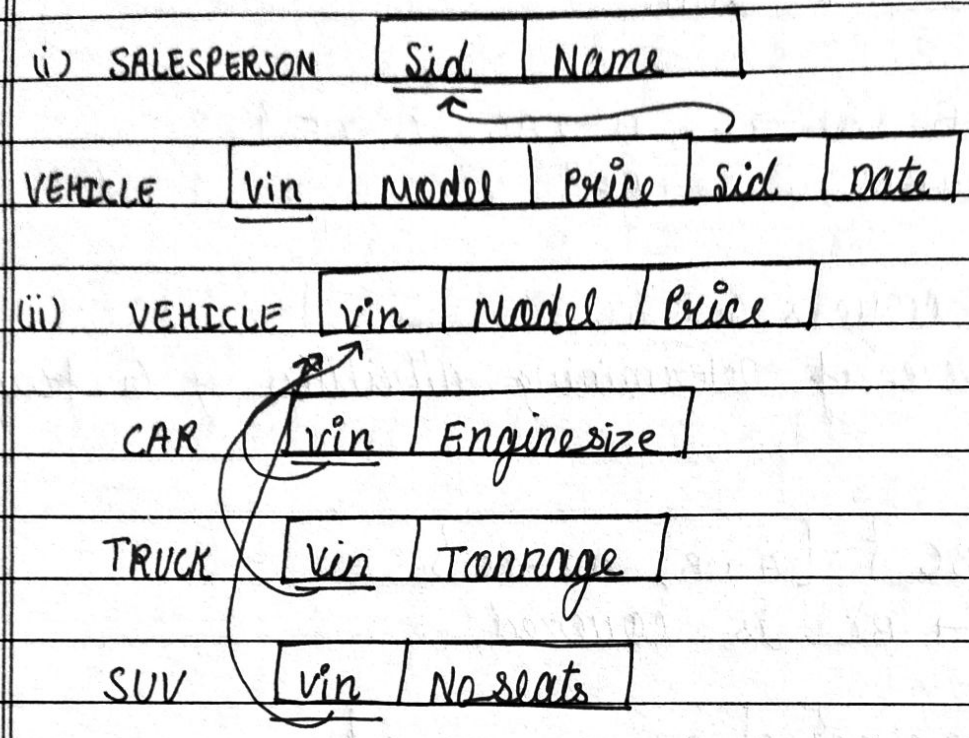
Check if G covers F
Find closure of determining attributes of F from FDs of G
$A^+ = \{ABC\} \because [A \to BC]$
Hence $A \to B$ is covered

$AB^+ = \{ABC\} \because [A \to BC]$
$AB \to C$ is covered

$D^+ = \{DAEBC\} \because [D \to AE][A \to BC]$
$D \to AC$ is covered

$D^+ = \{DAEBC\}$
$D \to E$ is covered
$\Rightarrow$ G covers F
$\Rightarrow$ F and G are equivalent.

(b) (i)



(ii)

(c)

```
create TABLE T(
    T1 INT,
    T2 CHAR(20),
    T3 INT DEFAULT 6,
    T4 CHAR(6) ENUM("Male", "Female")
    T5 INT,
    PRIMARY Key (T1, T2),
    FOREIGN KEY (T5) REFERENCES S(T5)
);
```

(d) (i)

X	Y	Z	(A ∪ B)
15	a	7	
25	b	8	
35	d	6	
25	a	9	
25	c	9	

(ii) $A \bowtie_{A.X = B.U} B$

X	Y	Z	U	V	W
25	b	8	25	a	9
25	b	8	25	c	7
35	d	6	35	d	6

(iii) $\Pi_{A.Z,\ B.W}\left(\sigma_{A.Y = B.V}(A \times B)\right)$

X	Y	Z	U	V	W
15	a	7	25	a	9
35	d	6	35	d	6

$\Rightarrow$

Z	W
7	9
6	6

(e) Data independence is the ability to change schema at one level of database system without having to change the schema at the next higher level.

1. Logical data independence: is the capacity to change the conceptual schema without having to change external schemas or application programs.

eg. Adding a column does not mean that External view or queries need to be changed.

2. Physical data independence is the capacity to change internal schema without having to change the conceptual schema.

eg. Changing internal querying or retrieving procedures does not require changes in conceptual schema, creating additional access structure

(4)

Primary Index

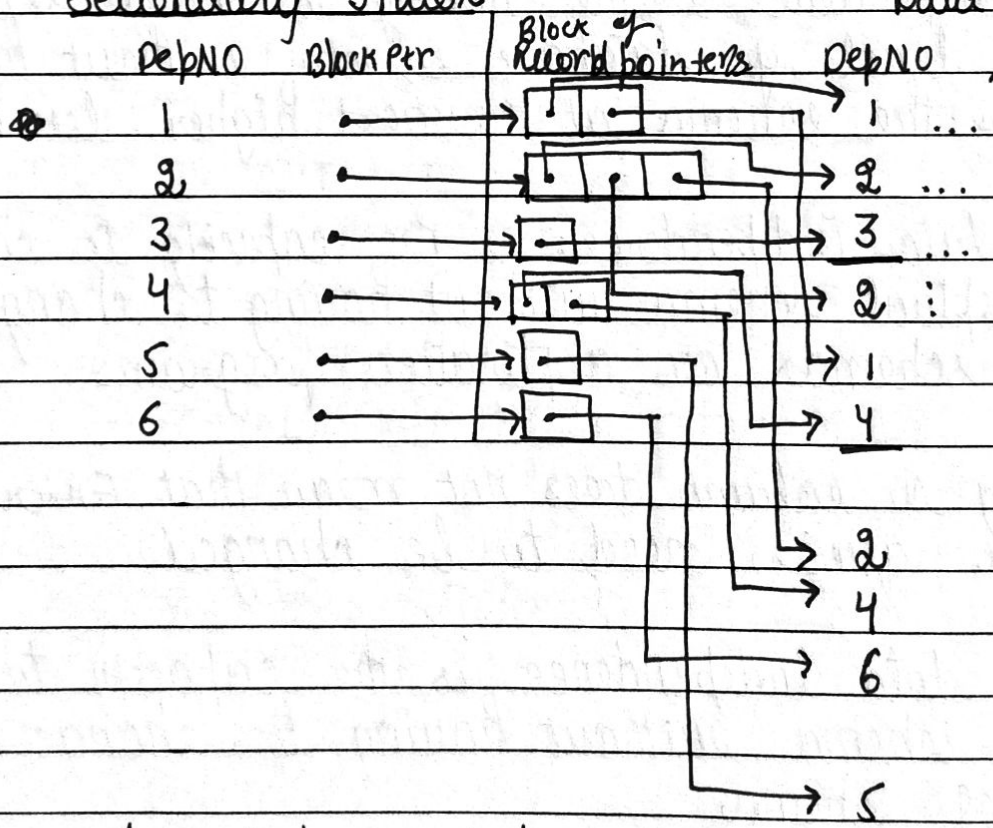
Empid	Block Pointer
01	•
04	•
07	•
⋮	⋮

Data File

Empid, Ename, Sal ....

(01) ...
02 ...
03 ...
(04) ...
05 ...
06 ...
(07) ...
08 ...
09 ...

Secondary Index



Secondary Index will take more space. In primary index only one entry is needed for each block whereas in secondary index a entry is needed for each distinct value and Block of record pointers is needed for no. of times a value repeats.

(g.) Transactions should possess several properties called ACID properties.

Atomicity : A transaction should either be performed in its entirety or not performed at all.
Responsibility of → (transaction recovery subsystem)

Consistency preservation: After the transaction is executed, it should leave database in a consistent state.
(programmers)

Isolation: A transaction should appear as though it is being executed in isolation from other transactions even though many transactions are executing concurrently.
(concurrency control subsystem)

Durability or permanency: The changes applied to a database by a committed transaction must persist in the database.
Responsibility of recovery subsystem.

System log:
To be able to recover from failures that affect transactions, the system maintains a log to keep track of transactions operations.

The log is a sequencial, append only file that is kept on disk, so it is not affected by any type of failure except for disk or catastrophic failure.

Red Wings
Page No.___ Date __/__/20__

Q2. i) MOVIE , ACTOR , DIRECTOR , PRODUCTION

ii) MOVIE (title, year, length , {Genre}, plot outline)

PRODUCTION (name, address)

DIRECTOR (name, dob)

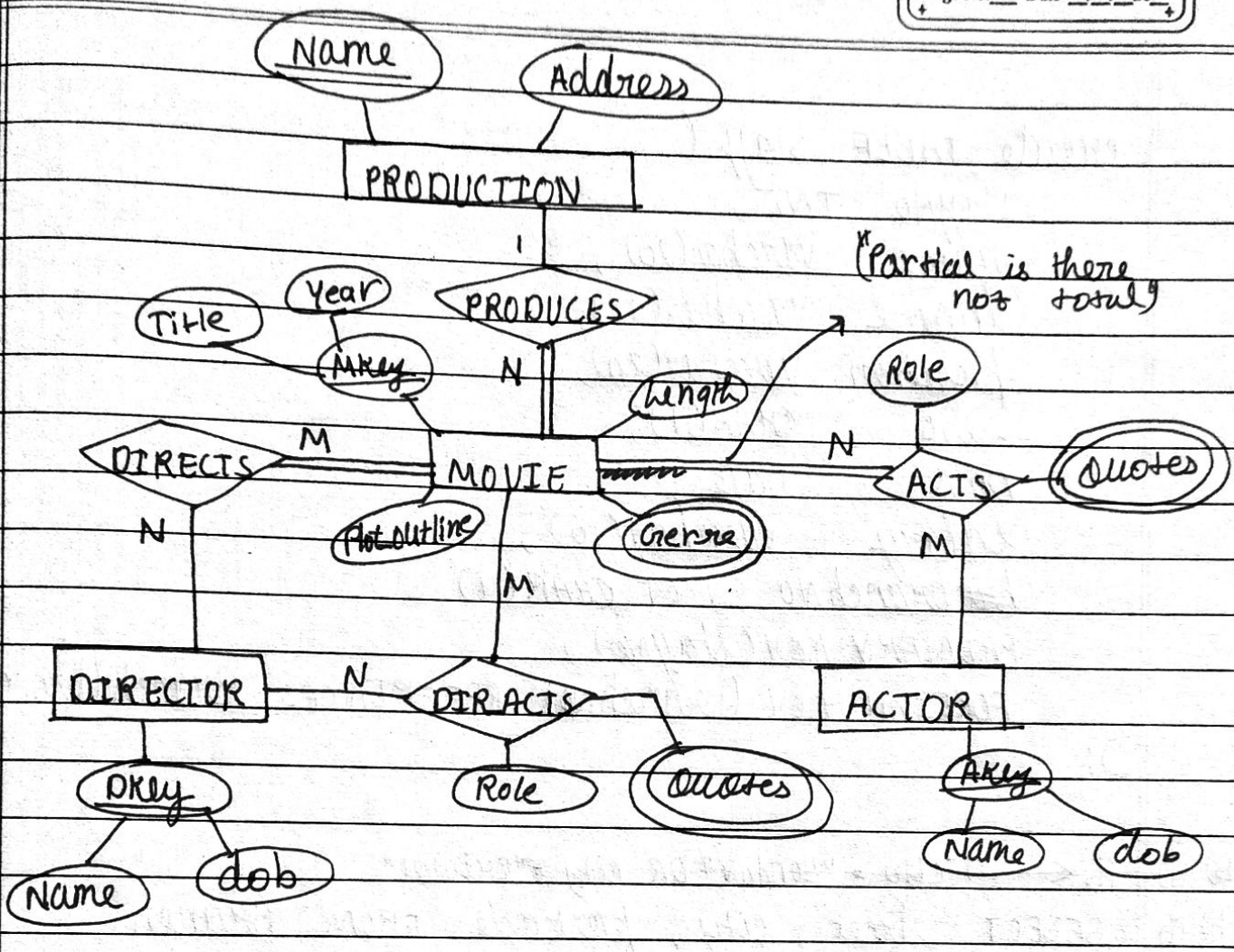
ACTOR (Name, dob)

* Relationship Type:
PRODUCES : Relates PRODUCTION and MOVIE
Cardinality ratio : 1 : N
Total participation on movie
constraint

DIRECTS : Relates DIRECTOR and MOVIE
M : N relationship
Total participation on movie

ACTS : Relates ACTOR and MOVIE | ACTS (Role, {quotes})
M : N relationship

DIR_ACTS : Relates DIRECTOR and MOVIE
DIR_ACTS (Role, {quotes})
M : N relationship



Q3. (a) Branch (branchNO, street, city, postcode)
Staff (staffNO, fname, lname, position, sex, DOB, salary, branchNO)

(b)
```
create TABLE BRANCH (
    branchNO ~~INT~~ CHAR(4),
    ~~TEXT~~ street VARCHAR(20),
    city VARCHAR(20),
    postcode VARCHAR(10),
    PRIMARY KEY (branchNO)
);
```

```
create TABLE Staff (
    staffNO  INT,
    fname    varchar(20),
    lname    varchar(20),
    position varchar(20),
    sex      CHAR(1),
    DOB      Date,
    salary   varchar(10),
    branchNO CHAR(4),
    PRIMARY KEY(staffNO),
    FOREIGN KEY (branchNO) REFERENCES Branch(branchNO)
);
```

b) (i) ~~$R_1 \leftarrow \sigma_{city = "London" \text{ OR } city = "Bristol"}$~~

b) (i)
```
SELECT street, city, postcode FROM Branch
WHERE city = "London" OR city = "Bristol";
```

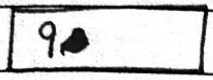
(ii)
```
SELECT MAX(salary), MIN(salary), AVG(salary)
FROM Staff;
```

(iii)
```
SELECT branchNO, COUNT(branchNO) AS count, SUM(salary) AS salsum
FROM Staff
GROUP BY branchNO
HAVING COUNT(branchNO) > 1
```

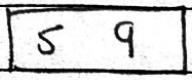
(iv)
```
SELECT * FROM Staff
WHERE SALARY > (SELECT MAX(salary) FROM Staff
                WHERE branchNO = 'B003');
```

Q4. 9, 5, 2, 7, 4, 15 $p = 3$ $p_{leaf} = 2$

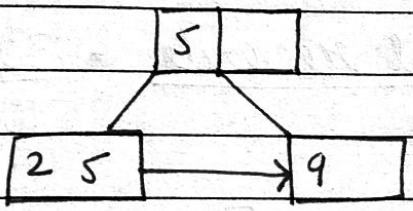
1)



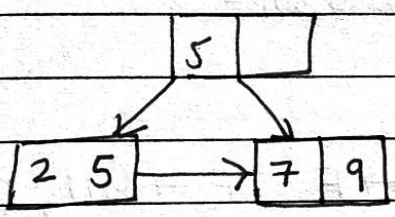
2)



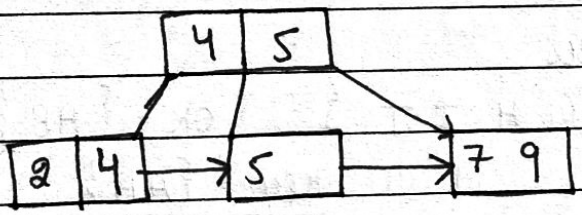
3)



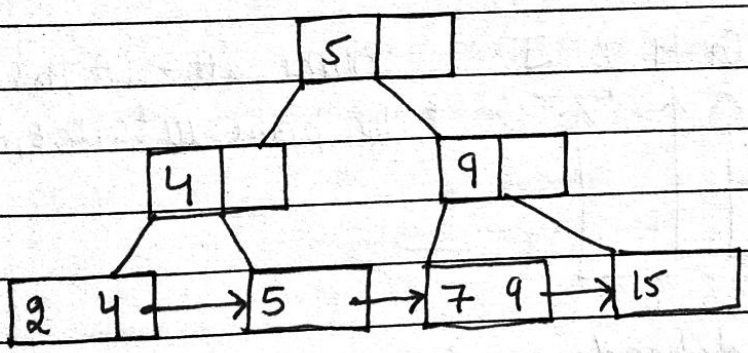
4)



5)



6)



Q4(b) $r = 30,000$ let record size = rs

$B = 1024$

Blocking factor = $\frac{1024}{rs}$

no. of blocks needed = $\frac{\text{Total records}}{Bfi} = \frac{30,000 \times rs}{1024}$

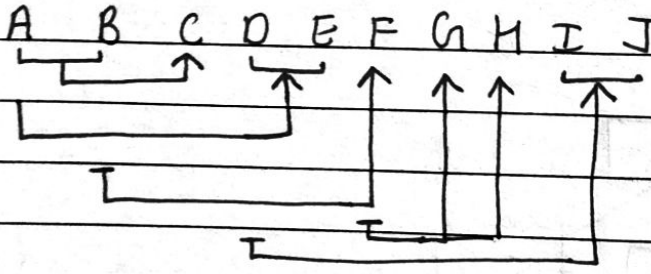
no. of block access = $\lceil \log_2(\text{no. of blocks}) \rceil =$

Q5(a) $F = \{ A, B \rightarrow C \ , A \rightarrow DE \ , B \rightarrow F \ , F \rightarrow GH \ , D \rightarrow IJ \}$

$AB^+ = \{A \ B\}$ Reflexive

$AB^+ = \{A \ B \ C \ D \ E \ F \ G \ H \ I \ J\}$ CK = {AB}

key = {AB}



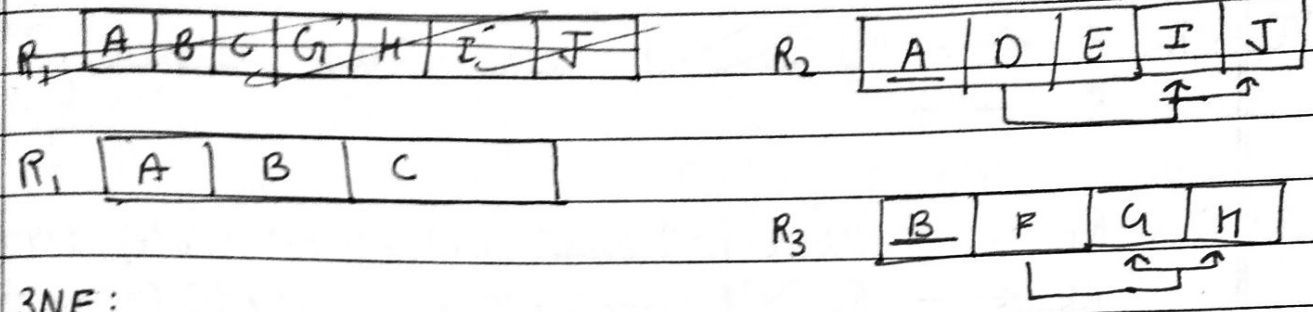
Prime att = {A, B}

Non prime att. {C, D, E, F, G, H, I, J}

2NF : NO partial dependency i.e every non prime attribute must be fully functionally dependent on

$A \rightarrow DE$, $B \rightarrow F$ are violating 2NF

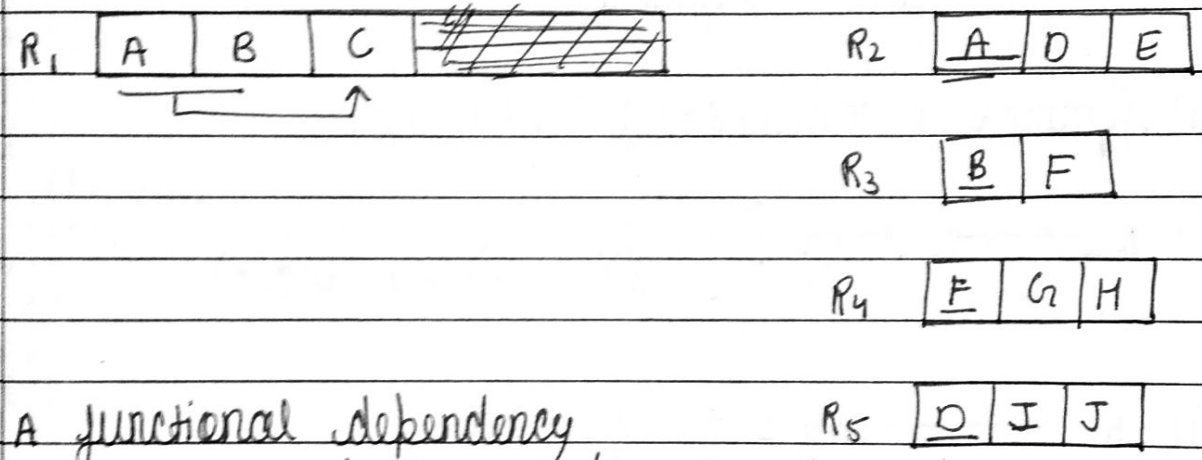
$A^+ = \{A D E I J\}$



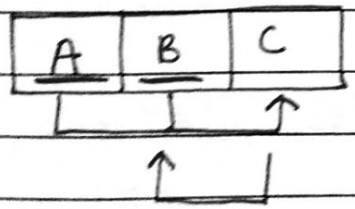
3NF:

No transitive dependency.

$F \rightarrow GH$ & $D \rightarrow IJ$ violates that because non-prime attributes are being determined by non-prime attributes.



(b) A functional dependency $X \rightarrow Y$ is a fully functional dependency if Y is functionally dependent on X and Y is not functionally dependent on any other proper subset of X.



FD $C \rightarrow B$ is in 3NF because Right hand side is a prime attribute. But BCNF disallows that property. According to it left hand side should be super key. Which C is not.

Red Wings
Page No.___ Date __/__/__20__

Q6(a) (ii) $R_1 \leftarrow$ Issue $\bowtie_{(Issue.custcode = customer.custcode)}$ (customer)

$R_2 \leftarrow$

Q6(a) (i) $R_1 \leftarrow$ Issue $\bowtie_{(issue.itemCode = item.itemCode)}$ Item

$R_2 \leftarrow R_1 \bowtie_{(R_1.custcode = customer.custcode)}$ (customer)

$R_3 \leftarrow \sigma_{price > 1000}(R_2)$

$R_4 \leftarrow \pi_{item}$ custname (R_3)

(ii) ~~$R \leftarrow \rho_{R(maxcost)} \leftarrow \mathcal{F}_{max(itemprice)}$ (item)~~

$R \leftarrow \sigma_{itemprice = n}$

(ii) $\rho_{R_1(maxcost)} \mathcal{F}_{max(itemprice)}$ (item)

$R_2 \leftarrow$ item $\bowtie_{(item.itemprice = maxcost)} R_1$

(iii) $R_1 \leftarrow$ Issue * Customer

OR

$R_1 \leftarrow$ Issue $\bowtie_{Issue.custcode = Customer.custcode}$ Customer

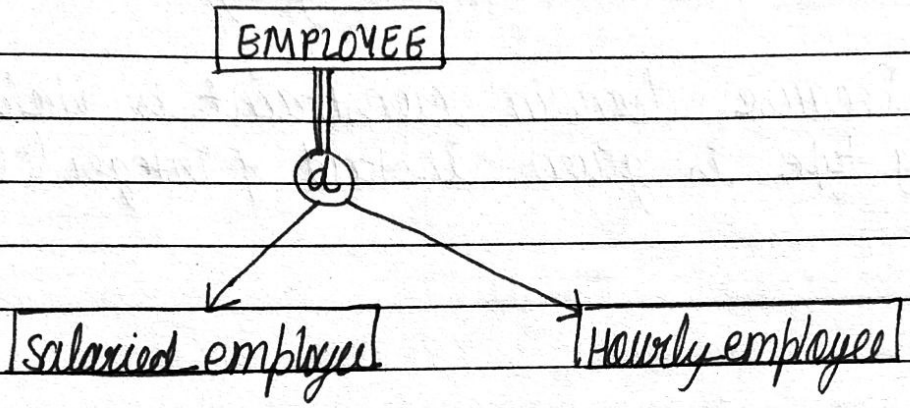
$R_2 \leftarrow \sigma_{itemcode = '1005'}(R_1)$

$R_3 \leftarrow \pi_{custcode, custName, custAddress, custPhone}(R_2)$

(b) i) Self describing nature of database system
2) Insulation between programs and data, and data abstraction.
3) Support of multiple views of the data.
4) Sharing of data and multiuser transaction processing.

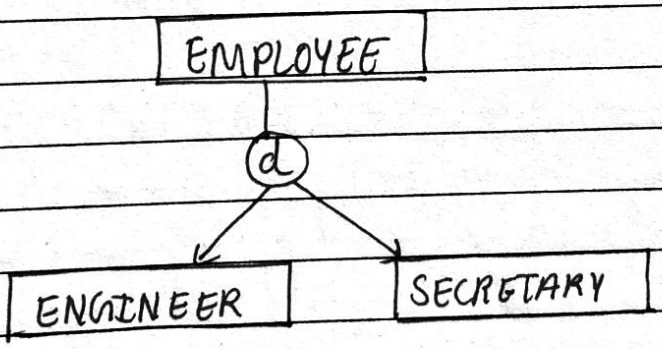
Q7 (a) (i) Disjoint total

When an employee can either be a salaried employee or hourly employee and Employee entity must me a member of one of the subclass.



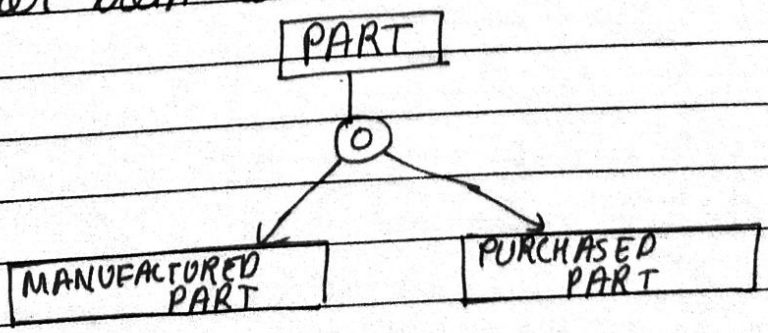
(ii) Disjoint partial

When an employee can either be Engineer or Secretary. OR none of these.



(iii) Overlapping partial

When a PART can be a manufactured part, purchased part or both or none.



OR

A Person can be a student or Instructor, or both, or none.

Q7(b). (i) Yes because CNO = 10 is not there in Course.

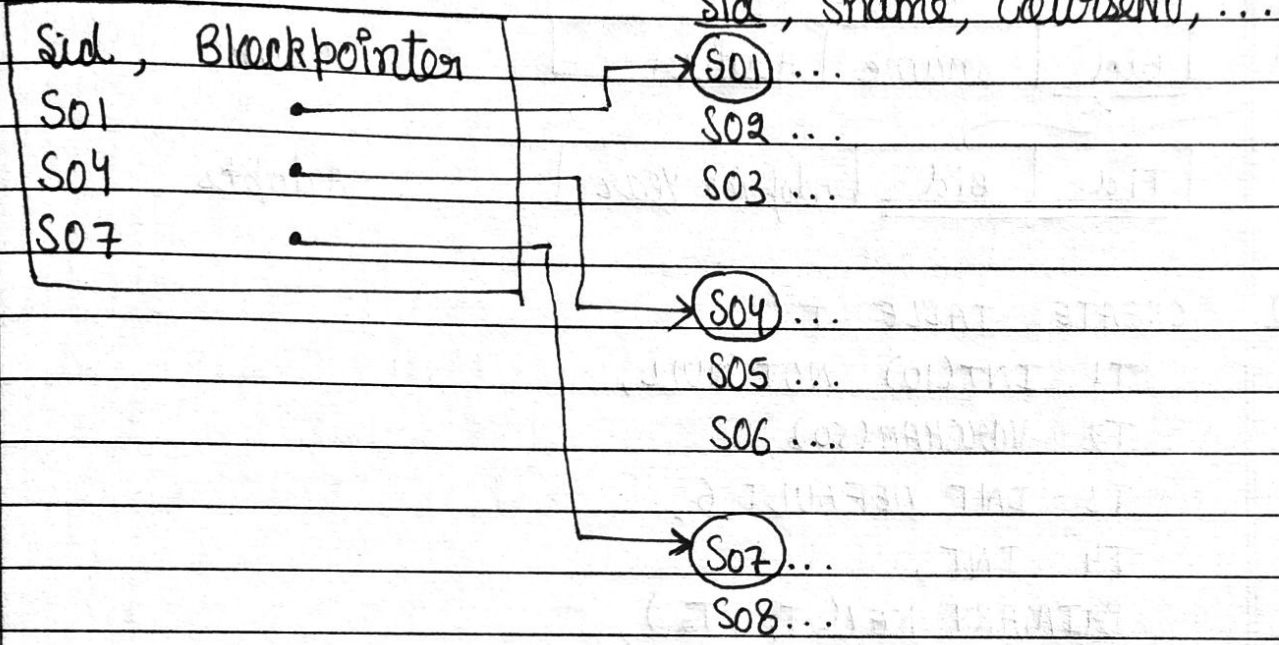
(ii) NO

(iii) ~~Yes~~ NO, because there is not entity type with name DEPT

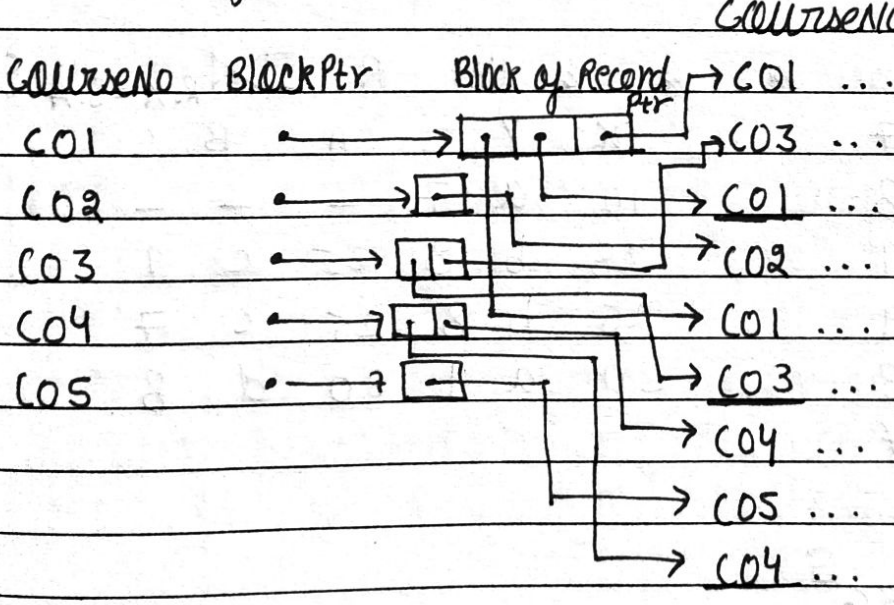
(iv) Yes, because domain constraint is violated.
String type is given instead of integer ("Electronics").

PAPER - 2017

Q1(a) Primary index

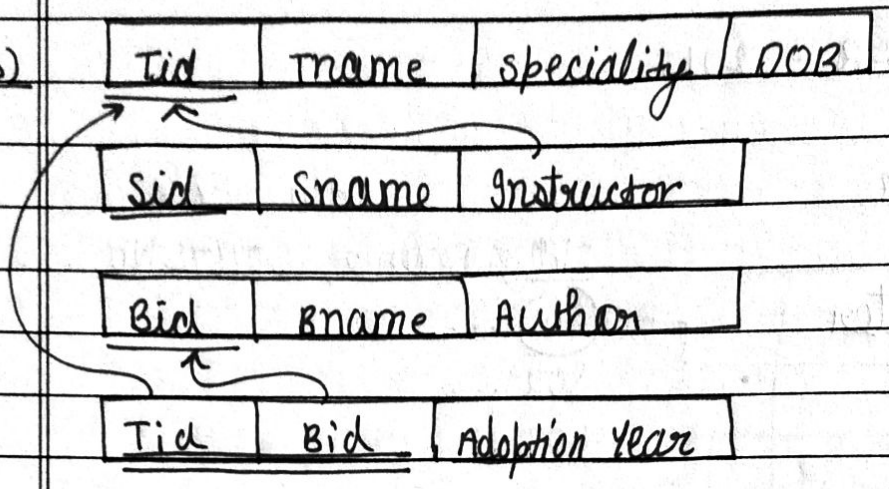


Secondary index



Secondary index will take more space. In primary index only one entry is needed for each block whereas in secondary index an entry is needed for each distinct value and block of record pointer is also needed for no. of times a value repeats.

Q1(b)



Q1(c)

```
CREATE TABLE T(
  T1 INT(10) NOT NULL,
  T2 VARCHAR(50),
  T3 INT DEFAULT 6,
  T4 INT,
  PRIMARY KEY(T1, T2),
  FOREIGN KEY(T4) REFERENCES S(T5)
);
```

Q1(d) RUS

(i)

X	Y	Z
10	a	7
25	b	8
30	a	9
25	c	9
30	d	8
25	c	7

(iii) $R \bowtie_{R.X = S.A} S$

X	Y	Z	A	B	C
10	a	7	–	–	–
25	b	8	25	c	9
25	b	8	25	c	7
30	a	9	30	d	8

(ii) $R \bowtie_{A.Z = B.C} S$

X	Y	Z	A	B	C
10	a	7	25	c	7
25	b	8	30	d	8
30	a	9	25	c	9

Q1(e) Refer to 2018, Q1(e)

To be able to recover from transaction failures, the system maintains a log file to keep track of transaction operations.

Q1(e) $F = \{A \to B,\ A \to C,\ C \to A\}$

$G = \{A \to B,\ B \to C,\ A \to C,\ C \to A\}$

check if F covers G : i.e all FD's of G must be present in F.

* Find closure of L.H.S (Determining) attributes of G FROM FD's of F.

$A^+ = \{A\ B\ C\}$ $\quad C^+ = \{C\ A\ B\}$

Hence $A \to B$ is covered, $A \to C$ & $C \to A$ are also covered.

$B^+ = \{B\}$

We can see that $B \to C$ is not covered by F.

Since $(B \to C)$ FD of G cannot be covered by F. Hence F and G are not equivalent.

Q1(g) ALTER : It is data definition language (DDL). It is used to alter the schema of the table or add or drop various constraints.

eg, ALTER TABLE STUDENT
ADD Email varchar(255);

ALTER TABLE STUDENT
ADD PRIMARY KEY(Sid);

UPDATE : It is Data Manipulation Language (DML). It is used to modify the data in table.

```
UPDATE  STUDENT
SET  Sname = "Yash"
WHERE Sid = "S09";
```

Q2 (i) Team , Player , ~~Game~~ , Injury

(ii) Player (name, position , skill level) ~~injury records~~

Team (name , city, coach , captain)

~~Game (hostteam, guestteam, date , score)~~

Injury (injury date , injury type)

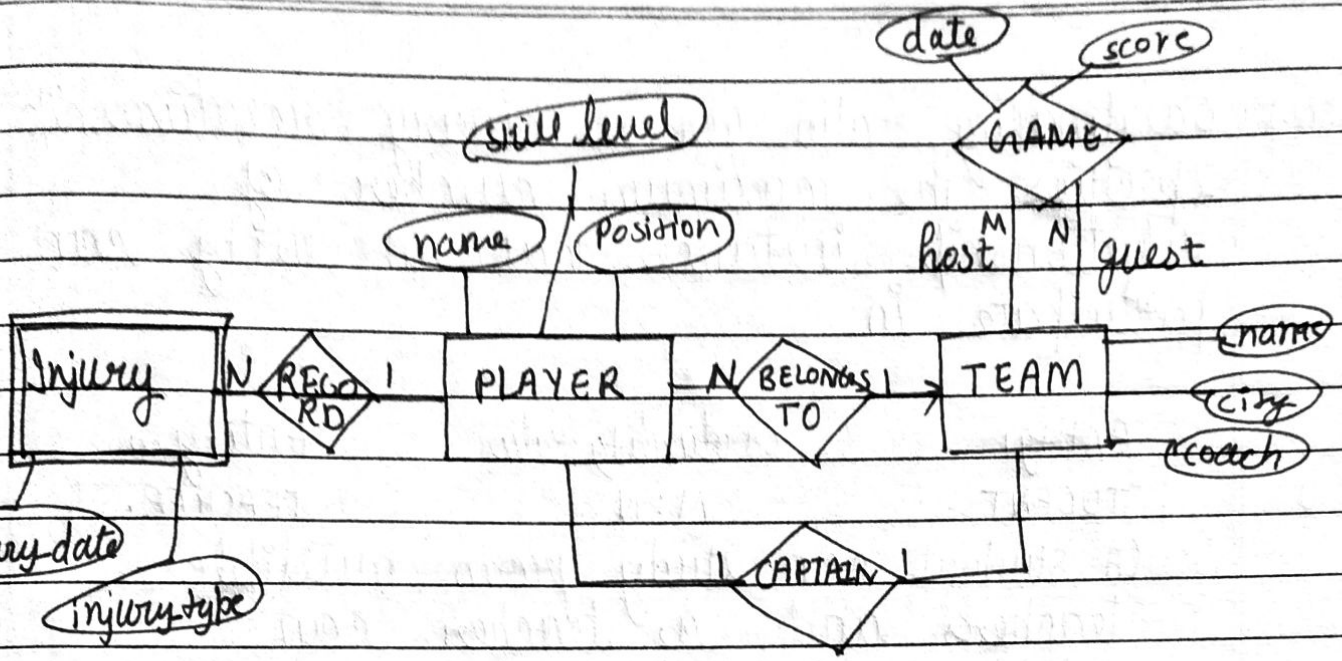
Relationship type :

Belongs to : PLAYER BELONGS TO TEAM
Cardinality ratio : 1 : N

Captain : Between TEAM and PLAYER
Cardinality ratio : 1 : 1 , i.e each team has only one captain.

Record : Between PLAYER and Injury
1 : N because a single player can have multiple injuries

Game : M : N



Q3(a)

(i)
```
SELECT BranchId, title, NoOfCopies
FROM BOOK_COPIES JOIN BOOK
ON BOOK_COPIES.BookId = Book.BookId
```

(ii)
```
SELECT BranchId, No.of Copies
FROM BOOKCOPIES JOIN BOOK
ON BOOK_COPIES.BookId = BOOK.BOOKId
WHERE title = "Fundamentals of Database Systems"
```

(iii)
```
SELECT title, PublisherName
FROM (Book-Loans JOIN BOOK
ON Book-Loans.BookId = Book.BookId) JOIN Library Branch
WHERE DueDate = CURDATE()
ON BookLoans.BranchId = Library Branch.BranchId
WHERE DueDate = CURDATE() AND
BranchName = 'CP'
```

Q3(b) Cardinality ratio for a binary relationship specifies the maximum number of relationship instances that an entity can participate in.

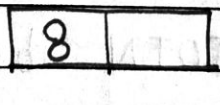
	Entity 1	Cardinality Ratio	Entity 2
1)	STUDENT	M:N	TEACHER

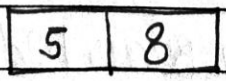
(A student can study from multiple teachers and a teacher can teach many students)

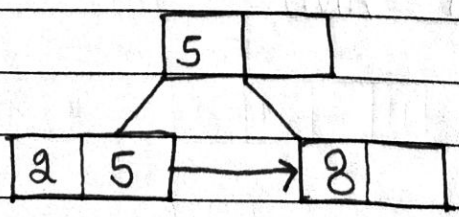
2)	COUNTRY	1:1	CURRENT_PRESIDENT
3)	LIBRARY	M:N	BOOK
4)	ITEM	M:N	ORDER

(A order can have multiple items and same item can be present in multiple orders).

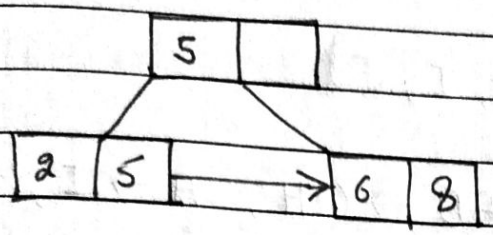
Q4(a) 8, 5, 2, 6, 4, 25 $p = 3$ $\text{leq} = 2$.

① 

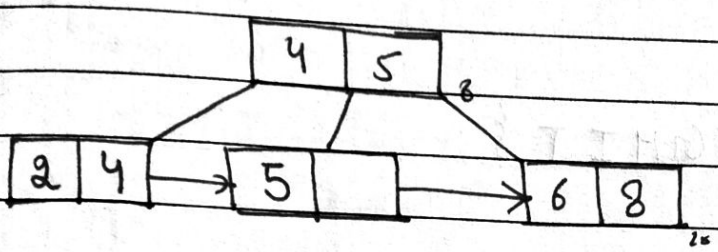
② 

③ 

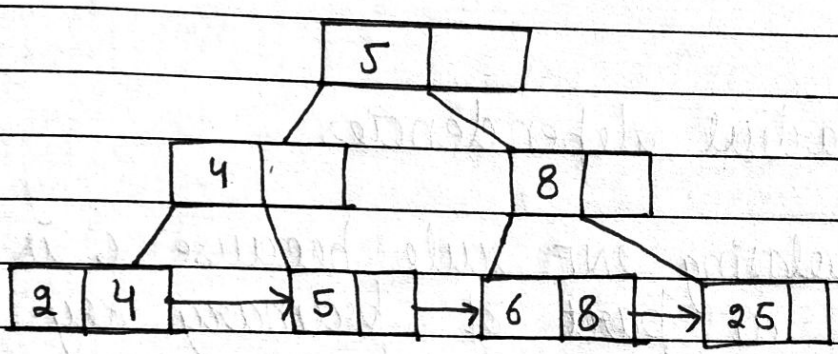
④



⑤



⑥



Q4(b) $r = 30,000$ $B = 1024$ bytes

$r_s = 9 + 6 = 15$ bytes (For primary index)

* Assuming each record in file is of 100 bytes.

Blocking factor = no. of records in each block

$$= \frac{\text{Total size of block}}{\text{Record size}}$$

$$= \frac{1024 \text{ bytes}}{100 \text{ bytes}} = \left\lfloor \frac{1024}{100} \right\rfloor = 10$$

No. of blocks $= \frac{30,000}{10} = 3,000$ block.

Primary Index:

No. of entries in primary index = no. of blocks

$= 3,000$

Blocking factor (Primary index) $= \left\lfloor \frac{1024}{15} \right\rfloor = 68$

No. of blocks needed $= \lceil 3000 / 68 \rceil = 45$

Q5(a) $R = \{A, B, C, D, E, F, G, H, I, J\}$

$F = \{ AB \rightarrow C , BD \rightarrow EF , AD \rightarrow GH , A \rightarrow I , H \rightarrow J \}$

$AB^+ = \{ABCIJ\}$

$ABD^+ = \{ABCDEFGHIJ\}$

Key = $\{ABD\}$

2NF : No partial dependencies

$AB \rightarrow C$ is violating 2NF rule because C is determind by a part of primary key.

R : | A | B | C | D | E | F | G | H | I | J |

R_1 : | A | B | C |

R_2 : | B | D | E | F |

R_3 : | A | D | G | H | J |

R_4 : | A | I |

3NF :

R_3 : | A | D | G | H | J | $H \rightarrow J$

Here in $H \rightarrow J$ an non. prime attribute is being determined by a non. prime attribute.

R_{31} : | A | D | G | H |

R_{32} : | H | J |

Q5(b) $B \rightarrow C$ does not hold because $\{b_1 \rightarrow c_1, b_1 \rightarrow c_2\}$

for same values in B, the value of determined attribute must be same.

$A \rightarrow B$ holds A uniquely identifies tuples in B.

Q6(a)

(i) $(orderCode)\mathcal{F}\ SUM(qtyOrdered)\ (item)$

(ii) $C \leftarrow \sigma_{(custCity = 'Delhi')}\ (customer)$

$O \leftarrow \sigma_{(itemCode = '2020')}\ (order)$

$R \leftarrow C * O$

(iii) $\rho_{R_1}(minCost)\ \mathcal{F}_{MIN(itemPrice)}\ (item)$

$R \leftarrow (item) \bowtie_{(item.itemPrice = minCost)} (R_1)$

Q6(b) (i) Referencial integrity constraint - 10 is not there in Department table.

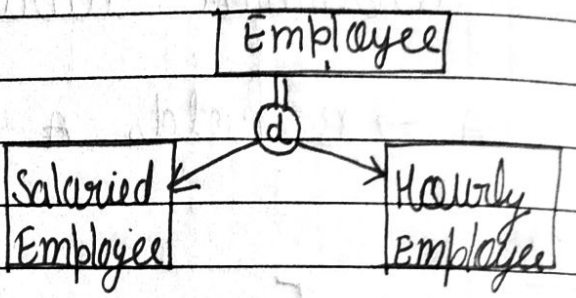
(ii) No violation

(iii) Referencial integrity constraint - 7 is being referenced in Employee table.

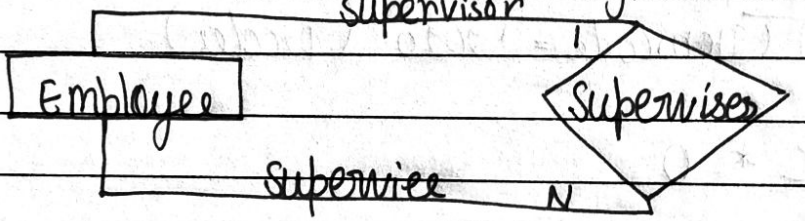
(iv) No violation

Red Wings
Page No.___ Date __/__/__20__

Q7(a) (i) Total and disjoint specializations/generalizations
Total and disjoint means that an entity in parent class must be a member of only one of the subclass.



(ii) A relationship between two entities of similar entity type is called a recursive relationship. Different roles should be assigned to entity types.



(iii) Weak entity:
Entity types that do not have a key attributes of their own are called weak entity types. A weak entity type normally has a partial key, which is the attribute that can uniquely identify weak entities that are related to same owner entity.

Q7(b) $F = \{ A \to B, A \to C, C \to A \}$

(i) $BC^+ = \{BC\}$ (Reflexivity)
$= \{BCA\}$ $\because (C \to A)$ [$BC \to C$ & $C \to A$ (Transitivity)]

(ii) $C^+ = \{C\}$ (Reflexivity)
$= \{CAB\} \because [C \to A$ and $A \to B]$