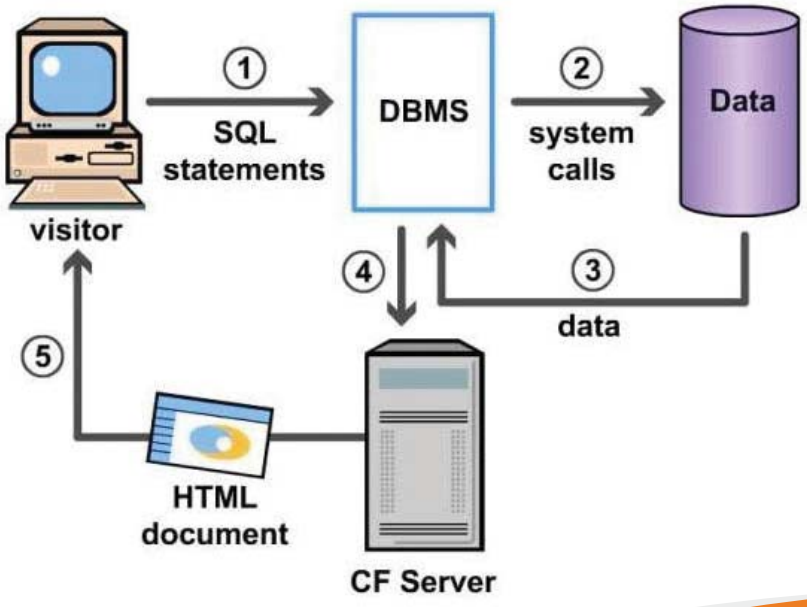
GATE | PSUs

COMPUTER SCIENCE & INFORMATION TECHNOLOGY

DATABASE MANAGEMENT SYSTEM

Volume - I : Study Material with Classroom Practice Questions



ACE Engineering Academy
(Leading institute for ESE/GATE/PSUs)

HYDERABAD | DELHI | BHOPAL | PUNE | BHUBANESWAR | LUCKNOW | PATNA | BENGALURU | CHENNAI | VIJAYAWADA | VIZAG | TIRUPATHI | KUKATPALLY | KOLKATA

Database Management Systems

(Classroom Practice Booklet Solutions)

2. ER and Relational Model

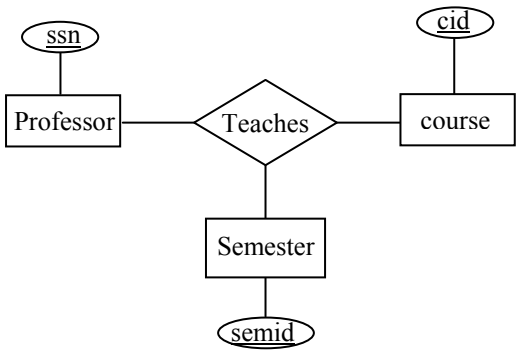
01. Ans: (b)

02. Ans: (c)

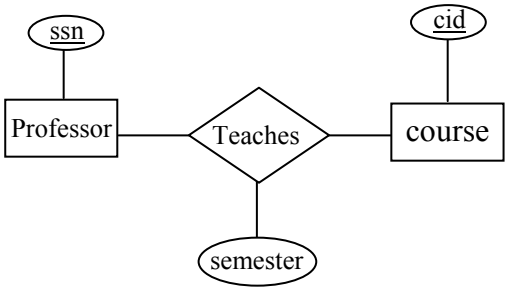
Sol: Because each patient is admitted into one ward, patient is a key to the relationship and of cardinality M : 1.

03. Ans:

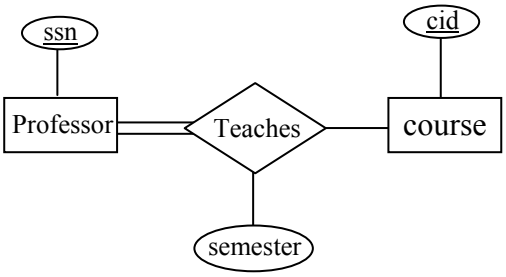
Sol: 1



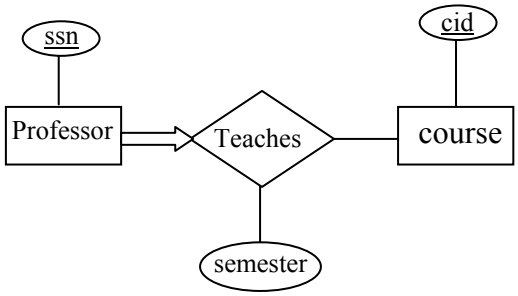
2



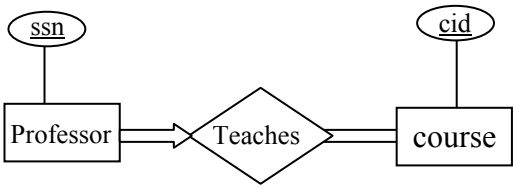
3



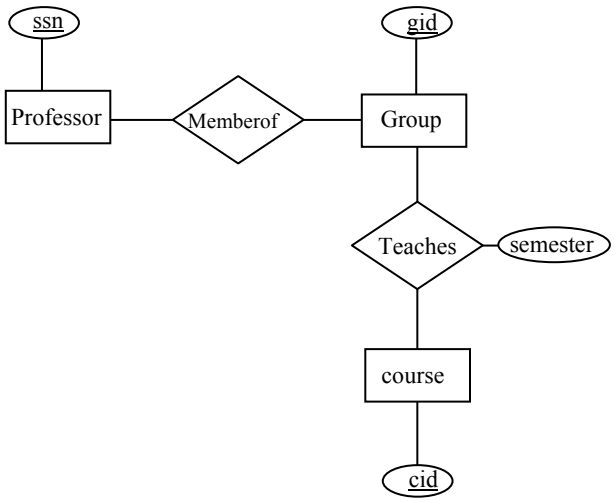
4



5



6



04. Ans: (c)

Sol: Set of attributes which contains a candidate key is said to be a super key.

05. Ans: (b)

Sol: All the values present in Foreign key must present in primary key of the referenced relation.

06. Ans: (c)

Sol: It violates referential integrity constraint.

07. Ans: (c)

Sol: On removal of row (2,4), row (5,2) and (7,2) must also be deleted as they depend on value 2. On removal of row (5,2), row (9,5) must also be deleted as it depends on value 5.

08. Ans: 0

Sol: When <3, 8> is deleted, its related tuples in T_2 is (8, 3) and 3 is to set null. Hence the number of additional tuples to delete is 0

09. Ans: 19.

Sol: For (StudentName, StudentAge) to be a key, all the combinations of these two attributes must unique.

10. Ans: (a)

Sol: As the key constraint from professor, the maximum number of tuples possible in Teaches is number of tuples in professor.

11. Ans: (a)

Sol:

12. Ans: (a)

Sol: As C, A and B in total participation with R_1 and R_3 and there is key constraint, all these are represented with one relation and R_2 is represented separately one relation.

13. Ans: (b)

Sol: There are some tuples of course may not participate with any tuple of professor, then cid is a key for the relation.

14. Ans: (a)

Sol: (AR_1B) will be one table as there is total participation and key constraint.

(CR_2) will be the second table as there is a key constraint.

15. Ans: (b)

Sol: As we get key and participation constraint from course to registration, therefore the number of tuples in registration will be equal to the tuples in the course table.

16. Ans: (b)

Sol: Strong entities E1 and E2 are represented as separate tables, in addition to that many to many relationship (R2) must be converted as separate table by having primary key of E1 and E2 as foreign key. One to many relationship must be transferred to 'many' side table by having primary key of one side

as foreign key. Hence we will have minimum of 3 tables.

17. Ans: (b)

Sol: Strong entities E1 and E2 are converted as separate tables. Since A23 is a multi valued attribute it should also be converted as separate table. Relationship R is transferred to 'm' side (E2).

18. Ans: (b)

Sol: M, P are strong entries hence they must be represented by separate tables. M table is modified to include primary key of P side (i.e P1). N is weak entity, and it is modified to include primary key of P (i.e P1).

19. Ans: (a)

Sol: M and P are strong entities hence they must be represented as separate tables. To include R1, M table is modified to accommodate primary key of P side (i.e P1) as foreign key. N is weak entity, so modify N to accommodate primary key of P (i.e P1) as foreign key.

Therefore tables are (M1, M2, M3, P1), (P1, P2), (N1, N2, P1). So correct answer is (M1, M2, M3, P1).

20. Ans: (b)

3. Functional Dependencies

01. Ans: (d) **02. Ans: (d)**

03. Ans: (a)

04. Ans: (c)

Sol: AF^+ = AFDE not ACDEFG as given.

05. Ans: (c)

Sol: An functional dependency $X \rightarrow Y$ is said to be trivial iff $Y \subseteq X$.

06. Ans: (b)

Sol: CD^+ from functional dependencies (FDs) = **C**DE**A**B, it includes RHS attributes AC so it can be derived from FDs BD^+ from functional dependencies

(FDs) = BD only, RHS attributes CD are not included in the closure hence it can not be derived BC^+ from functional dependencies

(FDs) = BCD EA, it includes RHS attributes CD, so it can be derived from FDs AC^+ from functional dependencies

(FDs) = ACBDE, it includes RHS attributes BC so it can be derived from FDs

07. Ans: (c)

Sol: AC^+ contains I then AC→ I dependency is possible.

08. Ans: (b)

Sol: As 'K' is independent attribute, key is ABDK.

09. Ans: (d)

Sol: ABD^{+} = A, B, C, D, E.

10. Ans: (b)

Sol: $ACEH^{+}$ contains all the attributes of R.

11. Ans: (d)

Sol: Closure of AEH^{+} = BEH^{+}= DEH^{+}= A, B, C, D, E, H. If any closure includes all attributes of a table then it can become candidate key of the table. Closure of AEH, BEH, DEH includes all attributes of table hence they are candidate keys.

12.

Sol: CK: ACD, BCD, ECD.

13. Ans: 6

Sol: ABF, ADF, EBF, EDF, CBF, CDF.

14. Ans: 2

15. Ans: (b)

Sol: The number of super keys are A, B, C, AB, AC, BC, ABC.

16. Ans: (a)

17. Ans: (d)

18. Ans: (c)

19. Ans: (a)

Sol: 'B' can be deleted from $AB \rightarrow C$.

$A \rightarrow BC$ is redundant dependency.

20.

Sol:

$AD \rightarrow CF$
$C \rightarrow B$
$B \rightarrow E$
} Canonical set

$AD \rightarrow C$
$C \rightarrow B$
$AB \rightarrow F$
$B \rightarrow E$
} Minimal set

21.

Sol:

$A \rightarrow BC$
$AE \rightarrow H$
$C \rightarrow D$
$D \rightarrow G$
$E \rightarrow F$
} Minimal set

4. Normalization

01.

Sol: 1. C.K = BD, Lossy, Dependency preserving

2. C.K = AB, CB, Loss-less,
   Not Dependency preserving

3. C.K = A, C, Loss-less,
   Dependency preserving

4. C.K = A, Loss-less,
   Not Dependency preserving

5. C.K = A, Lossy,
   Not Dependency preserving

02.

Sol: (a) FD :{ AB $\rightarrow$C, B $\rightarrow$D}
Where B$\rightarrow$D is partial Functional Dependency

(b) FD: {AB$\rightarrow$C, C$\rightarrow$ D}
Where C$\rightarrow$D is Transitive Functional Dependency

03.

Sol: R is in 1NF $\therefore$ decompose to 2NF

A^+ = {A, D, E, I, J} R_1 = 2NF

B^+ = {B, F, G, H} R_2 = 2NF

{A, B, C} R_3 BCNF

R_1 (A D E I J) $\xrightarrow{D^+}$ R_4 (D I J), R_5 (A E D)

R_2 (B F G H) $\xrightarrow{F^+}$ R_6 (F G H), R_7 (B F)

Then decompose into 2NF

R_1 (ADEIJJ)

R_2 (BFGH)

R_3 (ABC)

3NF also in BCNF

R_3 (ABC)

R_4 (DIJ)

R_5 (AED)

R_6 (FGH)

R_7 (BF)

AB^+ is key.

04.

Sol: Candidate key: AC

A^+ = (ABE) R_1, C^+ = (CD) R_2

AC = R_3

05.

Sol: (1) C $\rightarrow$ D

C $\rightarrow$ A

B $\rightarrow$ C

C.K: B, 2NF but not 3NF

(2) 2NF but not 3NF as no partial dependency CK: BD.

(3) R is in 3NF but not in BCNF

R $\rightarrow$ D^+ = {D, A} R_1, {DBC} R_2

Not D: P ABC $\rightarrow$ D lost

(4) C.K = A

No. P.D ∴ 2NF

R is in 2NF but not in BCNF & 3NF

BC^+ = {BCD} R_1

= {ABC} R_2

(5) AB → C

AB → D

C → A

D → B

Candidate Keys = AB, CD, BC, AD

R is in 3NF but not in BCNF.

C+ {C, A} R_1

D+ {D, B} R_2

{C, D} R_3

Not D.P AB → CD lost.

06. Ans: (b)

Sol: Both the dependencies satisfies the 3NF definition.

07. Ans: (b)

Sol: (Volume, Number) → Year is a partial functional dependency. So, the given relation is in 1 NF but not in 2 NF.

08. Ans: (d)

Sol: If all the determinants are super keys then the relation is in BCNF

09. Ans: (d)

Sol: Relation R_1 satisfies A→B, B→C and C→AB dependencies and all the determinants are super keys. Hence the relation is in BCNF.

10. Ans: (c)

11. Ans: (b)

Sol: 1 represents Partial FD and 2 represents T.D

5. Structured Query Language (SQL)

01. Ans: (b)

02. Ans: (c)

03. Ans: (b)

Sol: Select clause contains either aggregate function or the attributes that appear in group by clause.

04. Ans: (c)

Sol: Union operator eliminates the duplicates.

05. Ans: (c)

06. Ans: (d)

Sol: The inner query returns empty set, and any on empty set returns false always.

07. Ans: (c)

Sol:

R_1	
A	B
1	5
3	7

R_2	
A	C
1	7
4	9

$R_1 \bowtie R_2$		
A	B	C
1	5	7
3	7	null
4	null	9

08. Ans: (b)

09. Ans: (a)

Sol: all(empty) returns true always.

10. Ans: 7

Sol: The output of the query is

ta.player
Klose
Ronaldo
G muller
Fontaine
Pele
Klismann
Kocsis

Exercise - 01

01.

Sol: Display the details of all hotels, located in London

02.

Sol: Display name & address of all guests in ascending order of their name living in London.

03.

Sol: Display the details of all rooms of price below 40 & type of the room is either 'D' or 'F' in ascending order of their prices.

04.

Sol: Display all booking details whose check out time not specified.

05.

Sol: Finds total price of all rooms of type 'D'.

06.

Sol: Find the number of guests who have a booking on 1st august or in the month of august.

07.

Sol: List the type & price of all rooms in Grosvenor hotel.

08.

Sol: List all the guest details who have a booking for the current date in "Grosvenor Hotel'

Select * from guest where guestNo in ( ) &. hotelNo = ( )

= operator – when only 1 value is return.

09.

Sol: List the details of all rooms at the 'Grosvenor hotel' including the name of the guest staying in the room, if the room is occupied.

10.

Sol: Finds total price of all rooms in 'Grosvenor hotel' if the room is currently occupied.

11.

Sol: Find number of rooms in each hotel located in London.

12.

Sol: Find the maximum number of rooms booked in London of same type

A	100(Room Type)
B	200
C	150
D	75

Maximum 200. It displays total no. of bookings for the most popular room in city of London.

13.

Sol: Find total price of all rooms, that are not currently booked in each hotel.

Exercise - 02

01.

Sol: SELECT * FROM Patient ORDER BY patName.

02.

Sol: SELECT Count(patientNo), WardNo FROM Contains GROUP BY WardNo;

03.

Sol: SELECT p.patientNo, p.patName
FROM Patient p, Contains c
WHERE c.patientNo = p.patientNo
AND admissionDate = 'today'.

04.

Sol: SELECT p.patientNo, p.patName FROM Patient p, Ward w, Contains c
WHERE w.wardNo = c.wardNo
AND c.patientNo = p.patientNo
AND wardName = 'Surgical'.

05.

Sol: SELECT p.patName FROM Patient p, Prescribed pr, Drug d
WHERE pr.patientNo = p.patientNo
AND pr.drugNo = d.drugNo
AND drugName = 'Morphine'.

06.

Sol: SELECT SUM(((finishDate – startDate) *unitsPerDay) * costPerUnit) AS totalCost
FROM Patient p, Prescribed pr, Drug d
WHERE pr.patientNo = p.patientNo
AND pr.drugNo = d.drugNo
AND drugName = 'Morphine'
AND patName = 'John Smith'.

07.

Sol: SELECT w.wardNo, wardType, noOfBeds
FROM Patient p, Ward w, Contains c
WHERE w.wardNo = c.wardNo
AND c.patientNo = p.patientNo
AND admissionDate = 'today'

GROUP BY wardNo, wardType, noOfBeds
HAVING COUNT(*) > 10

08.

Sol: SELECT * FROM Patient p Left Join Prescribed pr
ON pr.patientNo = p.patientNo.

6. Relational Algebra & Calculus

01. Ans: (b)

02. Ans: (d)

Sol: Relational Algebra eliminate duplicates always.

03. Ans: (a)

Sol: If A→B holds, group by A results always groups with only one tuple.

04. Ans: (c)

Sol: If (X=Y) $\subseteq$ Z then the expressions to be true.

05. Ans: (a)

Sol: $\Pi_B(r1) - \Pi_C(r2) = \phi$ is always true. Because 'B' is foreign key referencing 'C' , so 'C' must be a primary key, 'B' cannot have a value that is not available in 'C'. Hence operation $\Pi_B(r1)–\Pi_C(r2)$ is always ϕ.

06. Ans: (a)

Sol: Common column between tables 'R' and 'S' is attribute B. In table 'R' B is primary key (B→A, A→C). In table 'S' B is foreign key so join is performed on attribute B. Therefore maximum tuples possible in the output is equal to rows in Table S (as it has less number of rows, provided B values are not repeated in table 'S').

07. Ans: (b)

Sol: bal < 0 filter rows from account ∞ depositor from which we can operate on few rows to filter b city = "Agra".

08. Ans: (d)

Sol: Minus operator indicates rows available in LHS table but not in RHS table. In this expression, LHS table produces all female students, RHS table consists students with less marks hence it produces names of all girl students with more marks than all the boy students.

09. Ans: (c)

Sol: P. duration = 3 months selects all projects of duration 3 months

T. pname = P.name selects project names in the output.

10. Ans: (d)

Sol: The relational algebra expression represents (all empId's) – (empId's of employees whose age is below than some dependent age); results, set of empId's of employee whose age is greater than all of his/her dependent.

11. Ans: (a)

Sol: SQL, Relational algebra, tuple relational calculus and Domain relational calculus all is representing the same. i.e., all these expressions representing to find the distinct names of all students who score more than 90% in the course numbered 107.

12. Ans: (d)

Sol: Minus operator indicates, rows available in LHS table but not in RHS table. In this expression, LHS table produces all female students, RHS table consists students with less marks hence it produces names of all girl students with more marks than all the boy students.

13.

Sol:

(a) $\left\{ \begin{pmatrix} T \mid T \in \text{Employee} \wedge \exists M \in \text{Manger} \\ M.\text{Manager\_name} = \text{'Jones'} \wedge \\ T.\text{Person\_name} = M.\text{Person\_name} \end{pmatrix} \right\}$

(b) $\left\{ \begin{matrix} T \mid \exists E \in \text{Employee} \\ (T.\text{City} = E.\text{City}) \wedge \exists M \in \text{Manager} \begin{pmatrix} M.\text{manager\_name} \\ = \text{'Jones'} \wedge E.\text{person\_name} \\ = M.\text{person\_name} \end{pmatrix} \end{matrix} \right\}$

(c) $\left\{ T \mid \exists\, E \in \text{Employee} \begin{pmatrix} M.\text{person\_name} \\ = \text{'Jones'} \wedge T.\text{Manager\_name} \\ = E.\text{Manager\_name} \end{pmatrix} \right\}$

7. Transaction & Concurrency Control

01. Ans: (d)

Sol: Transaction T_3 perform read on A, which is updated by T_1 and committed before T_1 does.

02. Ans: (a)

Sol: As $R_3(x)$ is dirty operations which read $W_1(x)$ and is committed before T_1. Hence schedule is non-recoverable.

03. Ans: (c)

Sol: A recoverable schedule is one where for each pair of transactions T_i and T_j such that T_j reads a data item previously written by T_i, the commit operation of T_i appear before the read operation of T_j.

04. Ans: (d)

Sol: In option 'd' transaction T_2 performs dirty read.

05. Ans: (c)

06. Ans: (a)

Sol: Construct the wait-for-graph, contains edges from T1→T2, T2→T3, and T4→T2 and there is no circular wait, hence no deadlock.

07. Ans: (a)

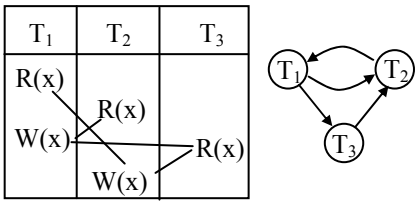
Sol: The algorithm is wound-wait deadlock prevention strategy, hence deadlock free. As the killed transaction restarting with same time stamp, it is starvation free.

08. Ans: (d)

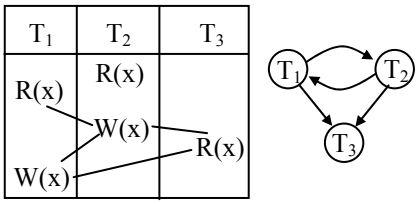
09. Ans: (d)

Sol:

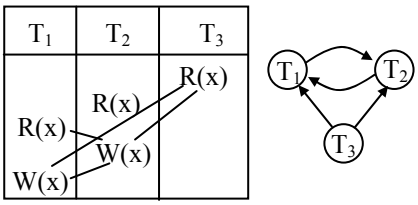
Option (A)



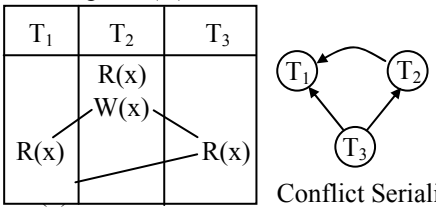
Option (B)



Option (C)



Option (D)



Conflict Serializable

10. Ans: 3

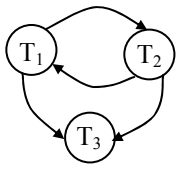
Sol: (3-4-1-2), (3-1-4-2), (4-3-1-2) are serial schedule possible.

11. Ans: (c)

Sol:

T_1	T_2	T_3
R(A)		
	W(A)	
		R(A)
W(A)		
		W(A)

Precedence graph is



12. Ans: (d)

Sol: S1 and S2 are conflict equivalent to serial schedule T2, T3, T1.

S3 is not conflict equivalent as 2RA, 3WA (T2<T3) and 3WA, 2WA (T3<T2) are the conflict operations. There is no serial schedule that satisfies both T2<T3 and T3<T2.

13.

Sol: (a) Not Conflict Serializable,
Not View Serializable,
Recoverable, Avoids Cascading aborts,
Not strict.

(b) Conflict Serializable,
View serializable,
Serializable,
Recoverable,
Cascading aborts, Not strict

(c) Not Conflict Serializable,
Not View Serializable,
Not avoids cascading aborts, not strict, Recoverable, cascading aborts

(d) Conflict Serializable,
View serializable,
Serializable,
Recoverable,
Avoids cascading aborts,
Not strict

(e) Not Conflict Serializable,
Views serializable through Thomas write rule, Serializable,
Recoverable,
Avoids cascading aborts,
Not strict

(f) Conflict Serializable,
View Serializable,
Serializable,
Recoverable,
Avoids cascading aborts,
Not strict

(g) Not Conflict Serializable,
Not View serializable,
Not Serializable,
Not Recoverable,
No need cascading aborts,
Not strict

(h) Conflict Serializable,
View Serializable,
Serializable,
Not Recoverable,
No need cascading aborts,
Not strict

(i) Conflict Serializable,
View Serializable,
Serializable,
Recoverable,
Avoid cascading aborts,
Strict

8. Indexing

01. Ans: 400

Sol: Blocking factor = 512/20=25
Number of data blocks = 10000/25=400
In primary index, the number of index records= 400 which is number of blocks in multi level index.

02. Ans: (c)

Sol: Since block size is 2^{10} (=1024 bytes), $2^m \times 2^n = 2^{10}$. Only option (c) satisfies with m = 8 and n = 2.

03.

Sol: 99 records

04. Ans: (c)

05. Ans: (c)

Sol: $n \times P + (n - 1)\ k <= B$. Where n is order of the node, P is the block pointer, k search key value and B is block size.
$n \times 4 + (n - 1)\ 8 = 512$, $12\ n = 520$,
$n = 43$.

06. Ans: (b)

Sol: $n \times P + (n-1)k <= B$. Where n is order of the tree, P is block pointer, k is key value and B is block size.

Therefore $n \times 6 + (n - 1)9 <= 1024$.

$n = 1033/16 = 64$ (approximately)

07. Ans: 5

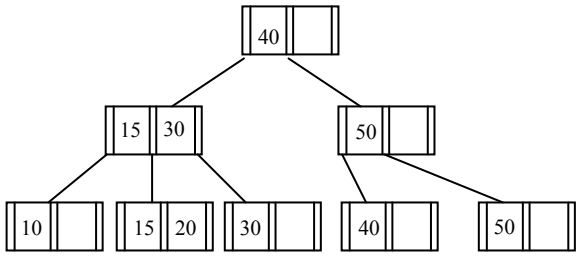
Sol: The nodes to access all records with a "search key greater than or equal to 7 and less than 15" is (9), (5), (5, 7) (9, 11) and (13, 15).

08. Ans: (b)

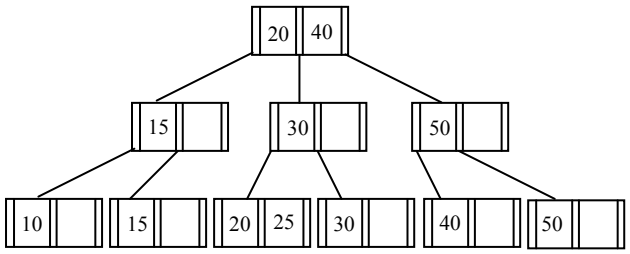
09. Ans: (d)

10. Ans: (a)

Sol: Insert 15



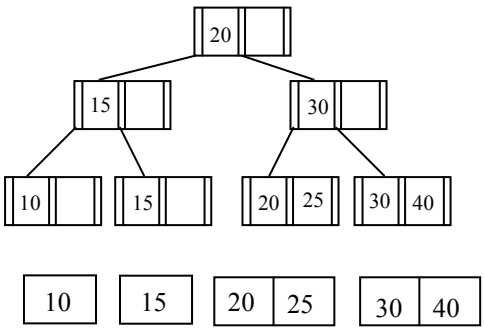
Insert 25



11. Ans: (a)

Sol: If we remove K50 in index node, there is no change in height as still root, index and leaf nodes exist.

Leaf nodes are now, after Delete 50, the B^+ is:



Hence Root now consists: 20

(i) is true, (ii) is true but (iii) is not true