Page No.
Date : / /

Year: 2019, Semester : Fall.

1.a) What do you understand by Data Independence? How is schema different from Instance? Justify with some suitable examples.

Ans. Data Independence:

→ is the ability to modify a schema definition in one level without affecting a schema definition in next high level.

→ Each higher level of data architecture is immune to the changes of next lower level of architecture.

→ We can change the structure of a database without affecting data required by users and programs.

There are two types of data independence:

a) Physical data independence:

- refers to the ability to modify the physical schema without affecting conceptual schema.

b) logical data independence:

- refers to the ability to modify the actual schema without affecting view schema.

Second part:

Database change over time as information is inserted and deleted. The collection of information stored in the database at a particular moment is called an instance of the database. The overall design of the database is called database schema. Schemas are changed infrequently, if at all. The concept of database schema and instances are way apart in their visualization.

Page No.
Date : / /

Let us take a student table having roll, name and address as the columns for Database Table 'student'.

roll	name	address.

Here, the structure itself is the schema of the table. It is initially formulated while modeling for database, and it can only be changed on only required & necessary scenario.

Whereas, If we insert data or do some query on the above schema based table, we will be having different values of data over the time. And, those information stored at a particular moment is called instance.

roll	name	address
1	John	Toronto
2	Hari	Michighan.

Instance 1.

roll	name	address
1	John	Damauli
2	Hari	Michighan

Instance 2.

In this way, we can differentiate bet^n schema and instances, and above example clarifies more of the concept.

Page No.
Date : / /

Q) How does UML diagram assist during data modeling? Draw an E-R diagram for a Gandaki Auto Vechile shop system including primary key, weak entity, composite attribute, derived attribute and multivalued attributes in your ER diagram.

Ans: Unified Modeling language (UML) diagrams are the ways to visually show the behavior and structure of a system or a process. Hence helps to define the detailed structure of the data elements in a system and the relationships between data elements. In this way, it assist during / for data modeling.

Second part:

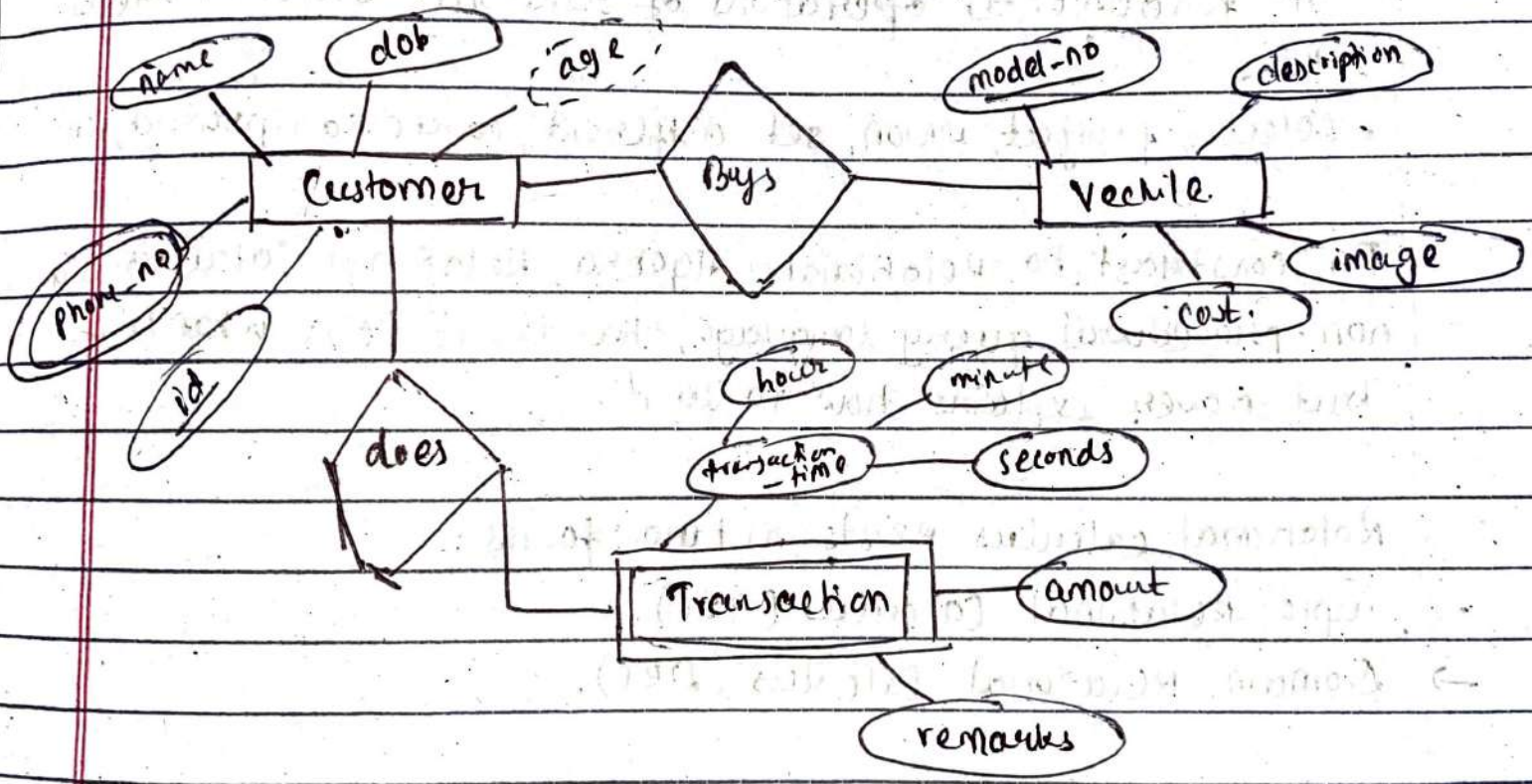


fig: ER diagram for Gandaki Auto-Vechile shop system.

Here, phone_no (Customer) is multivalued attribute
model-no (Vechile) is primary key.
Transaction is Weak Entity.
transaction_time is composite attribute

Page No.
Date : / /

age is derived attribute.

2.a) How Relational Algebra is different from Relational Calculus? Define TRC and DRC.

Ans. Relational algebra:

Relational algebra is a procedural query language, which takes instances of relations as input and yields instances of relations as output. It uses operator to perform queries. An operator can be either unary or binary. They accept relations as their input and yield relations as their output. Relational algebra is performed recursively on a relation and intermediate results are also considered relations. The fundamental operations of relational algebra are as follows:

- select, project, union, set different, cartesian product, rename.

In constrast to Relational Algebra, Relational calculus is a non-procedural query language, that is, it tells what to do but never explains how to do it.

Relational calculus exists in two forms:

→ Tuple Relational Calculus (TRC).
→ Domain Relational Calculus (DRC).

Tuple Relational Calculus (TRC):

It is a non-procedural query language that specifies to select the tuples in relation. It can select the tuples with a range of values or tuples for certain attribute values etc. The resulting relation can have one or

Page No.
Date : / /

more tuples.

Notation:

$\{T \mid P(T)\}$ or $\{T \mid condition(T)\}$.

where T is the resulting tuples and P(T) is a condition used to fetch T.

Example:

$\{T \mid EMPLOYEE(T) \text{ AND } T.DEPT\_ID = 10\}$

This selects all the tuples of employee names who work for Department 10.

Domain Relational Calculus (DRC):

A domain relational calculus uses the list of attributes to be selected from the relation based on the condition. It is same as TRC but differs by selecting the attributes rather than selecting whole tuples.

Notation:

$\{a_1, a_2, a_3, \ldots, a_n \mid P(a_1, a_2, a_3, \ldots, a_n)\}$.

where $a_1, a_2, a_3 \ldots$ are attributes of the relation and P is the condition.

Example:

$\{ \mid < EMPLOYEE > DEPT\_ID = 10\}$

select EMP_ID and EMP_NAME of employee who work for department 10.

Page No.
Date : / /

b> Consider a simple relational database of Hospital Management System. (Underlined attributes represent primary key attributes).
Doctors (DoctorID, DoctorName, Department, Address, Salary).
Patients (PatientID, PatientName, Address, Age, Gender).
Hospitals (PatientID DoctorID, HospitalName, Location).
Write down the SQL statement for the following.

i. Display ID of Patient admitted to hospital at Pokhara and whose name end with 's'.

→
```
SELECT PatientID FROM Patients
LEFT JOIN Hospitals
ON
Patients.PatientID = Hospitals.PatientID
WHERE Hospitals.Location = "Pokhara" AND
Patients.PatientName LIKE '%s';
```

ii. Delete the records of Doctors whose salary is greater than average salary of doctors.

→
```
DELETE FROM Doctors WHERE Salary > (SELECT AVG(Salary)
FROM Doctors);
```

iii. Increase the salary of doctors by 18.5% who works in OPD department.

→
```
UPDATE Doctors SET Salary = (Salary + 18.5/100 * Salary)
WHERE Department = "OPD".
```

iv. Find the average salary of Doctors for each address who have average salary more than 55k.

→

process of organizing data to reduce anomalies

Page No.
Date: / /

3.a) Define Normalization. Explain about 1NF, 2NF and 3NF.

Ans. Normalization of data can be considered a process of analyzing the given relational schemas based on their FDs and primary keys to achieve the desirable properties of:

- Minimizing redundancy and
- Minimizing the insertion, deletion and update anomalies.

It can be considered as a "filtering" or "purification" process to make the design have successively better quality. A series of normal form test can be carried out on individual relation schemas so that the relational database can be normalized to any desired degree.

First Normal form (1NF):

- A relation will be 1NF if it contains an atomic value.
- It states that an attribute of a table cannot hold multiple values. It must hold only single-valued attribute.
- First normal form disallows the multi-valued attribute, composite attribute, and their combinations.

Example: Relation EMPLOYEE is not in 1NF because of multi-valued attribute EMP_PHONE.

Employee table:

EMP_ID	EMP_NAME	EMP_PHONE	EMP_STATE
14	John	9846767896	Gandaki
20	Harry	1234 5678	Bagmati
12	Sam	9012	Province 1

Page No.
Date : / /

The decomposition of the EMPLOYEE table into 1NF has been shown below:

EMP_ID	EMP_NAME	EMP_PHONE	EMP_STATE
14	John	98467	Gandaki
14	John	67896	Gandaki
20	Harry	1234	Bagmati
20	Harry	5678	Bagmati
12	Sam	9012	Province 1

Second Normal form (2NF):

- In 2NF, relational must be in 1NF.
- In 2NF, all non-key attributes are fully functional dependent on the primary key.

Example: Let's assume, a school can store the data of teachers and the subjects they teach. In a school, a teacher can teach more than one subject.

Teacher table

TEACHER_ID	SUBJECT	TEACHER_AGE
25	Chemistry	30
25	Biology	30
47	English	35
83	Math	38
83	Computer	38

In the given table, non-prime attribute TEACHER_AGE is dependent on TEACHER_ID which is a proper subset of a candidate key. That's why it violate the rule of 2NF.

Page No.
Date: / /

To convert the given table into 2NF, we decompose it into two tables:

TEACHER_DETAIL table:

TEACHER_ID	TEACHER_AGE.
25	30
47	35
83	38.

TEACHER_SUBJECT table:

TEACHER_ID	SUBJECT
25	Chemistry
25	Biology
47	English
83	Math
83.	computer.

Third Normal Form (3NF):

- A relation will be in 3NF if it is in 2NF and not contain any transitive partial dependency.
- 3NF is used to reduce the data duplication. It is also used to achieve the data integrity.
- If there is no transitive dependency for non-prime attributes, then the relation must be in third normal form.

A relation is in 3NF if it holds atleast one of the following conditions for every non-trivial function dependency $X \rightarrow Y$.

1. X is super key.
2. Y is prime attribute, i.e. each element of Y is part of some candidate key.

Page No.
Date : / /

Example:

EMPLOYEE_DETAIL table:

EMP_ID	EMP_NAME	EMP_ZIP	EMP_STATE	EMP_CITY
222	Harry	201010	UP	Noida
333	Stephan	02228	US	Boston
444	Lan	60007	US	Chicago
555	Katharine	06389	UK	Norwich
666	John.	462007	MP	Bhopal.

Super key in the table above:
{EMP_ID}, {EMP_ID, EMP_NAME}, {EMP_ID, EMP_NAME, EMP_ZIP}. . . . so on.

Candidate key : {EMP_ID}.

Non-prime attributes: In the given table, all attributes except EMP_ID are non-prime.

Here, EMP_STATE & EMP_CITY dependent on EMP_ZIP and EMP_ZIP dependent on EMP_ID. The nonprime attributes (EMP_STATE, EMP_CITY) transitively dependent on super key (EMP_ID). It violates the rule of third normal form.

That's why we need to move the EMP_CITY and EMP_STATE to the new <EMPLOYEE_ZIP> table, with EMP_ZIP as a primary key.

Page No.
Date : / /

EMPLOYEE table:

EMP_ID	EMP_NAME	EMP_ZIP.
222	Harry	201010
333	stephan	02228
444	Lan	60007
555	Katharine	06389
666	John.	462007.

EMPLOYEE_ZIP table:

EMP_ZIP	EMP_STATE	EMP_CITY.
201010	UP	Noida
02228	US	Boston.
60007	US	Chicago
06389	UK	Norwich
462007	MP	Bhopal.

lossless join, dependency pres.?

→ (lossless, less redundant, attr. preserve)

b). What do you mean by decomposition of relational schema? Suppose we are given schema R = {A, B, C, G, H, I} and set of functional dependencies F = {A → B, A → C, CG → H, CG → I, B → H}. Find the closure of functional dependency F.

ANS. The process of breaking up or dividing relation into two or more sub relations is meant by decomposition of relational schema.

$F^+ = \{ A \rightarrow BC$ ∵ Union rule
$A \rightarrow H$ ∵ transitivity rule
$CG \rightarrow HI$ ∵ " "
$AG \rightarrow H$ ∵ Pseudo " "
$AG \rightarrow I$ ∵ " "
$\}$

$A = \{A\}$
$A^+ = \{ABC\}$

$B = \{B\}$
$B^+ = \{BH\}$

Page No.
Date : / /

4.a> What is Access control Mechanism in database? Explain different types of access control mechanism.

Ans. Access Control Mechanism refers to controlling access of database information. The access mechanism is granted as per the user type and as per the security of the system.

There are commonly two types of access control mechanism one is user based and another is host (machine) based.

- In user based mechanism, we create different users having assigned for access to certain database and tables with certain operations such as CREATE, DELETE, INSERT etc. Once the user is assigned in such way then their access will be limited to the assigned role. Hence our data is protected from unauthorized access.

To provide these kind of access, we can use GRANT command to remove we can make use of REVOKE command.

- Host (machine) based mechanism: This type of mechanism enables to give access to particular host, in which all users on that host can connect to the server. A host-based mechanism is a weaker form of access control. If the host has access to the server, all users on that hosts are allowed to connect to the server.

Other access control mechanism can be applied on the network and in the physical form such as data warehouse, the human beings and soon. Which hence creates the security implementation on database.

Page No.
Date : / /

b) Diagrammatically illustrate and discuss the steps involved in processing a query.

Ans. Query processing is the process of reading & parsing given query inorder to produce desireable output. There are 3 main steps involved in query processing and they are:

1. Parsing and translation.
2. Optimization.
3. Evaluation.

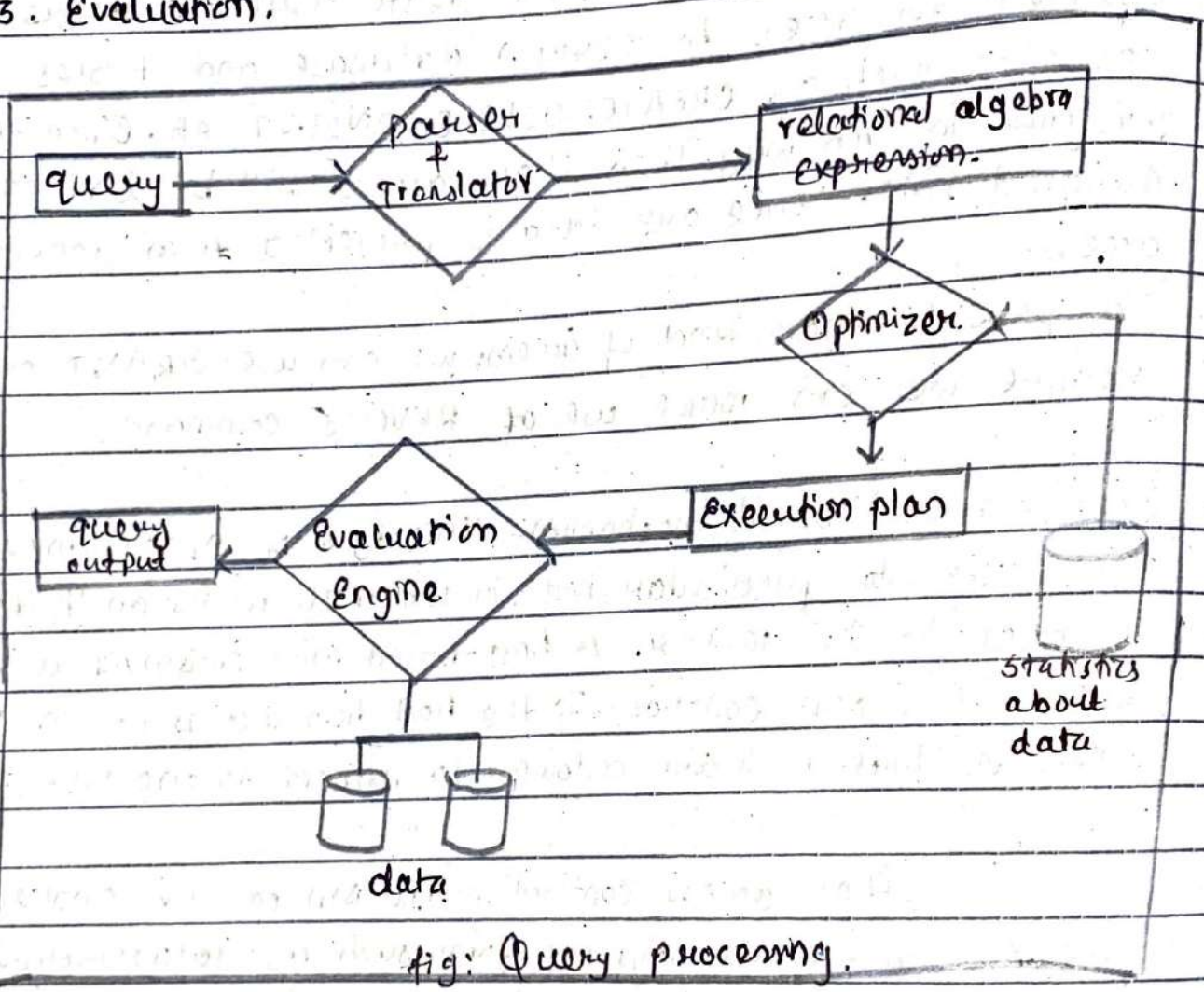


fig: Query processing.

- 1. Parsing and Translation: The first action the system must take in query processing is to translate a given query into its internal form. This translation process is similar to the work performed by the parser of a compiler.

Page No.
Date: / /

In generating the internal form of the query, the parser checks the syntax of user's query, verifies the relation names appearing in the query, names of relation of database and so on. The system constructs a parse-tree representation of the query, which it then translates into a relational-algebra expression.

2. Evaluation: A relational-algebra operation annoted with instructions on how to evaluate it is called an evaluation primitive. A sequence of primitive operations that can be used to evaluate a query is a query-execution plan or query evaluation plan.

As an illustration, consider the query:
SELECT balance FROM account WHERE balance < 2500.
figure below illustrates an evaluation plan for our example query.

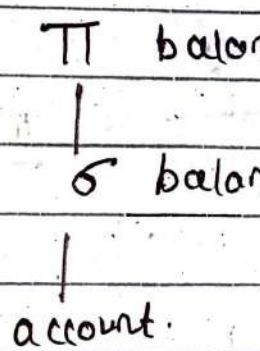


fig: A query evaluation plan.

The query-execution engine takes a query-evaluation plan, executes that plan, and returns the answers to the query

3. Query optimization: Amongst all equivalent evaluation plans choose the one with lowest cost. Cost is estimated using statistical information from the database catalog. E.g: number

Page No.
Date : / /

of tuples in each relation, size of tuples, etc. And, from this cost the query can be optimized by reducing the repeated and unwanted steps.

In this way, queries are processed.

5. a) Construct a B+ tree for the following set of key values: (2, 3, 5, 7, 11, 17, 19, 23, 29, 31). Assume that the tree is initially empty and values are added in ascending order where the pointer number is four.

→ P = 4

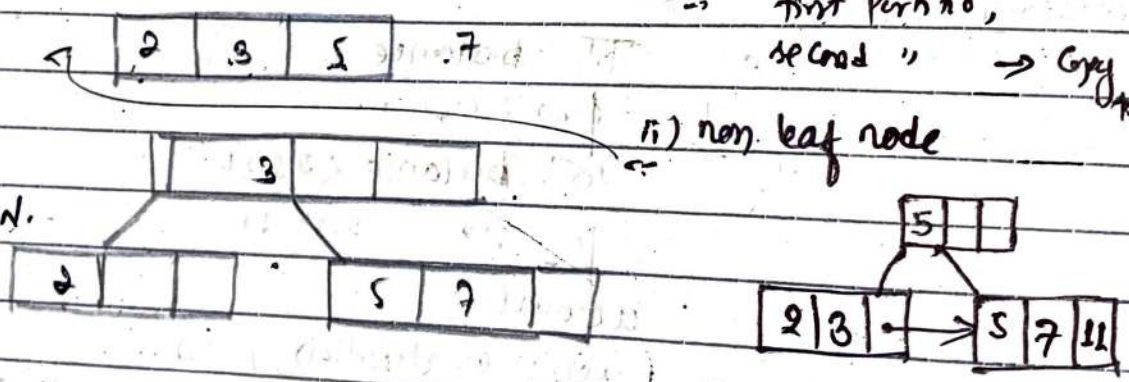
Max keys = P − 1 = 3.

Min = $\left\lceil \frac{P}{2} \right\rceil - 1 = 1$ (ceil)

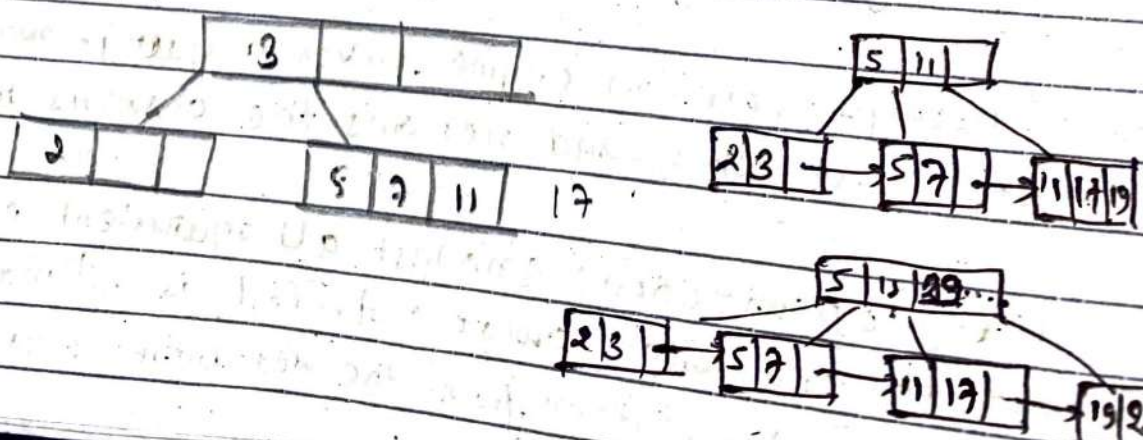
i) leaf node
- two partition
- → $\lceil (N-1)/2 \rceil$ → value
- → first partition, second " → Copy smaller to parent

ii) non leaf node

- two partition
- $\lceil N/2 \rceil - 1$ → f.p.
- remaining → S.p.
- move smallest to p.N.



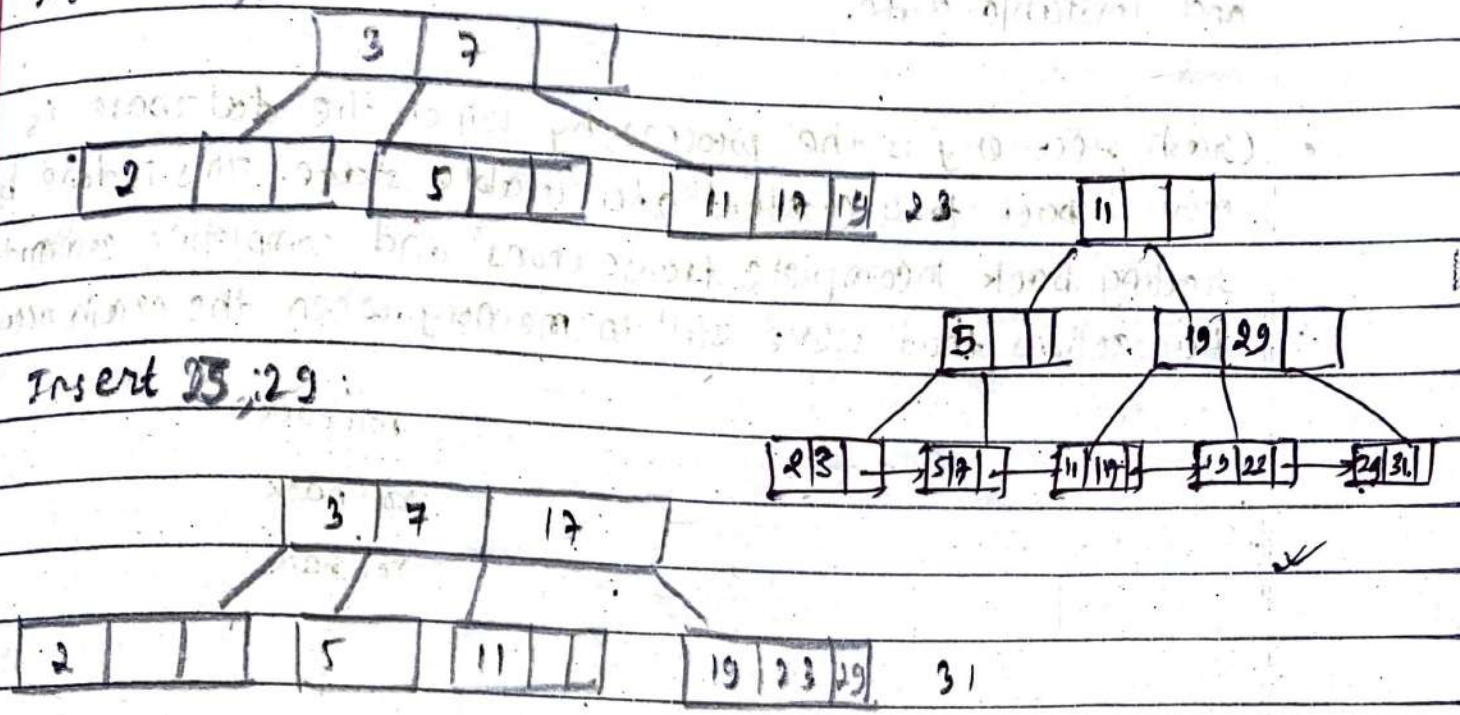
Insert 11:



$\lceil 4/2 \rceil - 1 = 1$

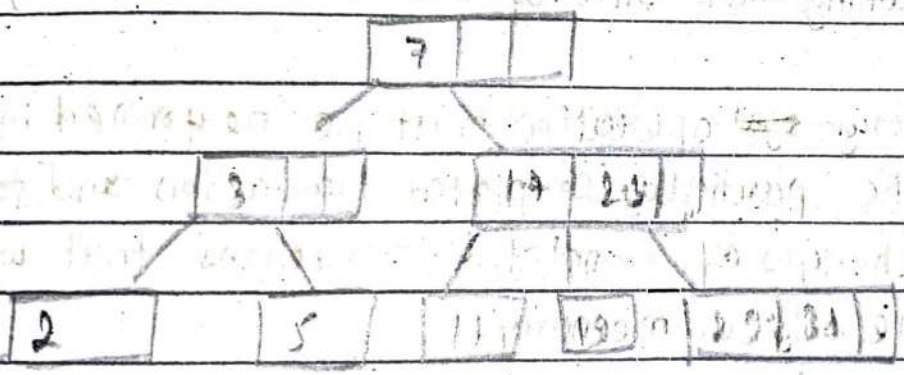
Page No.
Date: / /

Insert 17, 19



Insert 25, 29

Insert 31



b> What is crash Recovery? What are the problems due to crash? How the problems can be avoided, explain any one briefly.

Ans. • Transactions (or units of work) against a database can be interrupted unexpectedly. If a failure occurs before all of the changes that are part of the unit of work are completed, committed, and written to disk, the database is left to an incosistent

Page No.
Date : / /

and unstable state.

- Crash recovery is the process by which the database is moved back to a consistent and usable state. This is done by rolling back incomplete transactions and completing committed transactions that were still in memory when the crash occured.

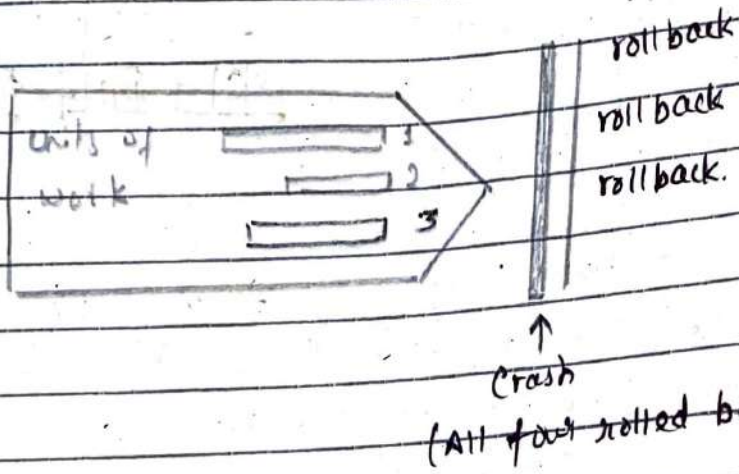


fig: Rolling Back Units of work (Crash Recovery).

A crash recovery ~~sys~~ operation must be performed in order to roll back the partially completed transaction and to write to disk the changes of completed transactions that were previously made only in memory.

Conditions that necessitate a crash recovery include:

→ A power failure on machine, causing the database operation incomplete.

→ A hardware failure such as memory, disk, CPU etc.

→ A serious operating system error, which causes DB instance to end abnormally.

{ based recovery
shadow paging

Page No.
Date : / /

these crash recovery can be done either by:

→ Rolling forward: it applies redoing records of the ongoing data block transaction.

→ Rolling Back: It applies rolling back segments of transaction to their previous stable state.

~~There are~~ :

Although the database crash cannot be mitigated completely, but there are some things that we can have to prent from crash. and they are:

1. Split the database: Not all databases should be in one system
2. Avoid storing large files in Database: large file such as video, photo should always be avoided.
3. Be careful with the power usage, and database technology used for database system.

6.a) When does deadlock occurs? Explain two-phase commit protocol with example.

Ans. A deadlock occures when two or more processes need some resources to complete their execution that is held by the other process.

In the above diagram, process 1 has resource 1 and needs resource 2. Similarly process 2 has resource 2 and needs resour

Page No.
Date: / /

each of these processes need the other's resource to complete but neither of them is willing to relinquish their resources. So, process 1 and process 2 are in deadlock.

Second part:

Two phase commit protocol is a type of distributed commit protocol. There are two different types of databases. In a local database system, every transaction needs to be committed. Therefore, the transaction manager has the role to commit the decision by conveying it to the report manager.

However, when it comes to a distributed system, the transaction manager should convey it from all the servers from various sites included in the distributed system to commit the decision. When each server completes the process at each site, the transaction reaches partially committed state. But it has to wait until all the transaction reaches that state. Once all the transactions from different servers reach the partially committed state, the transaction manager can commit the transaction. However, it is necessary that all the sites must commit the transaction.

Distributed Commit protocols;

There are three different types of commit protocols in the distributed systems. They are as follows:

↳ One-phase commit protocol.
↳ Two phase commit protocol.
↳ Three phase commit protocol.

Page No.
Date: / /

Two phase Commit protocol (Distributed Transaction Management) with example explanation:

Consider we are given with a set of grocery stores where the head of all store wants to query about the available sanitizers inventory at all stores in order to move inventory store to store to make balance over the quantity of sanitizers inventory at all stores. The task is performed by a single transaction T that's component Tn at the nth store and a store so correspond to To where the manager is located. The following sequence of activities are performed by T are below:

(a) Component of transaction (T) To is created at the head-site (head office).

(b) To sends messages to all the stores to order them to create components Ti.

(c) Every Ti executes a query at the store "i" to discover the quantity of available sanitizers inventory and reports this number to To.

(d). Each store receives instruction and update the inventory level and made shipment to other stores where require.

But there are some problems that we may face during the execution of this process:

1> Atomicity property may be violated because any store (Sn) may be instructed twice to send the inventory that may leave the database in an inconsistent state. To ensure atomicity property Transaction T must either commit at all the sites, or it must abort at all sites.

Page No.
Date : / /

2> However the system at store Tn may crash, and the instruction from To are never recieved by Tn because of any network issue and any other reason. So the question arises what will happen to the distributed transaction running whether it about or commit? whether it recover or not?

Two-phase commit protocol:

This protocol is designed with the core intent to resolve the above problems, Consider we have multiple distributed databases which are operated from different servers (sites) let's say $S_1, S_2, S_3, \ldots S_n$. Where every S_i made to maintains a separate log record of all corresponding activities and the transition T has also been divided into the sub Transactions $T_1, T_2, T_3, \ldots T_n$ and each T_i are assigned to S_i. This all maintains by a separate transaction manager at each S_i. We assigned anyone site as a Coordinator.

Some points to be considered regarding this protocol:

(a) In a two-phase commit, we assume that each site logs actions at that site, but there is no global log.

(b) The coordinator (C) plays a vital role in doing conformation whether the distributed transaction would abort or commit.

(c) In this protocol messages are made to send between the coordinator (C) and the other sites. As each messages is sent, it logs are noted at each sending site, to aid in recovery should it be necessary.

Page No.
Date: / /

The two phases of this protocol are as follows:

Phase 1st:

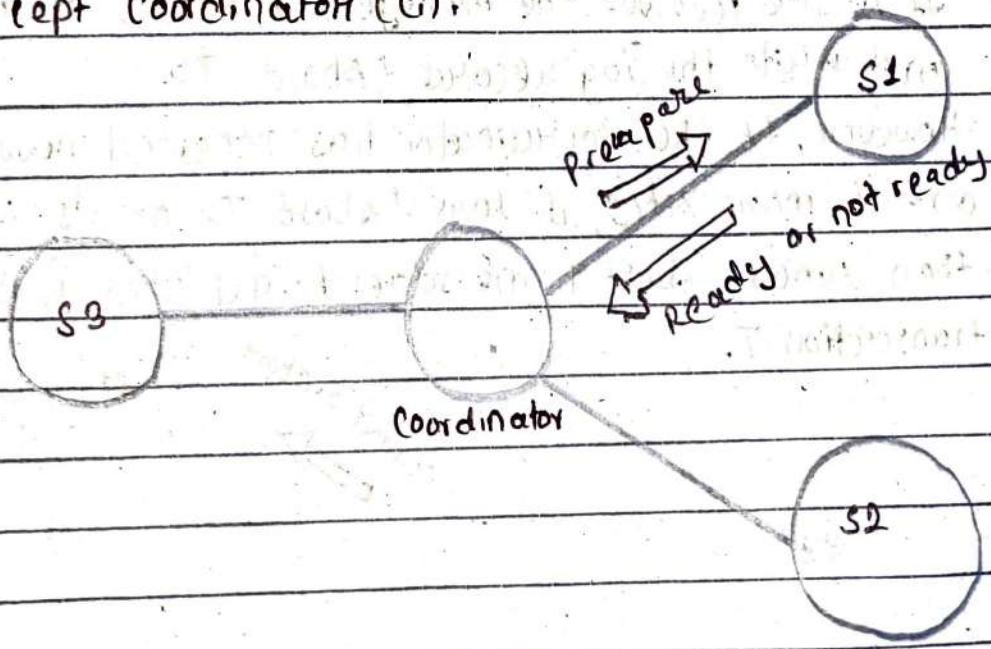
(a) Firstly, the coordinator (Ci) places a log record <Prepare T> on the log record at its site.

(b) Then, the coordinator (Ci) sends a 'prepare T' messages to all the sites where the Transaction (T) executed.

(c) Transaction manager at each site on receiving this message 'Preape T' decides whether to commit or abort its component (portion) of T. The site can delay if the component has not yet completed its activity, but must eventually send a response

(d) If the site doesn't want to commit, so it must write on log record <no T> and local Transaction manager sends a message abort T to Ci.

(e) If the site wants to commit, it must write on log record <ready T>, and local Transaction manager sends a messag ready T to Ci. Once the ready T message at Ci is sent noth can prevent it to commit its portion of transaction T except Coordinator (Ci):



Phase 1st

Page No.
Date : / /

Phase 2nd:

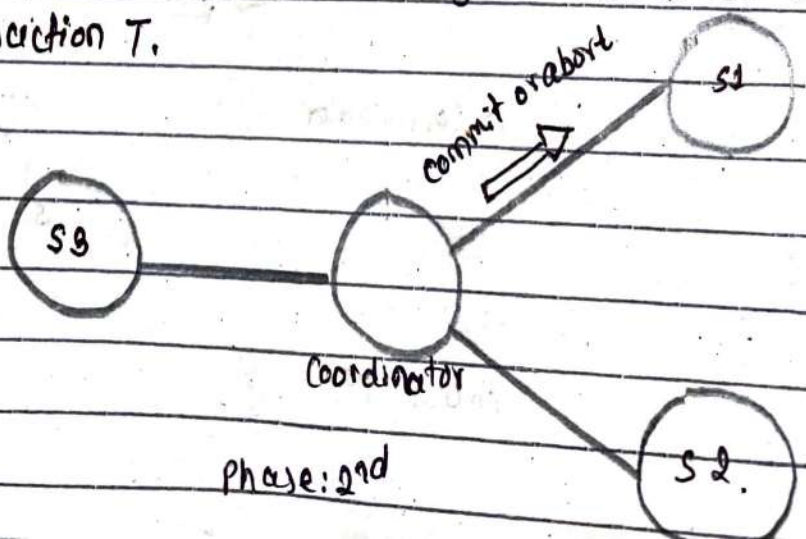
The second phase started as the response abort T or commit T receives by the coordinate (Ci) from all the sites that are collaboratively executing the Transaction T. However, it is possible that some site fails to respond; it may be down, or it has been disconnected by the network. In that case, after a suitable timeout period will be given, after that time it will treat the site as if it had sent abort T. The fate of the transaction depend upon the following points:

a) If the coordinator receives ready T from all the participating sites of T, then it decides to commit T. Then, the coordinator writes on its site log record ⟨Commit T⟩ and sends a message commit T to all sites involved in T.

b) If a site receives a commit T message, it commits the component T at that site, and write it in log records ⟨Commit T⟩.

c) If a site receives the message 'abort T', it aborts T and write the log record ⟨Abort T⟩.

d) However, if the coordinator has received abort T from one or more sites, it logs ⟨Abort T⟩ at its site and then sends abort T messages to all sites involved in transaction T.



Phase: 2nd

Page No.
Date : / /

Disadvantages:

a) The major disadvantage of 2-phase commit protocol is faced when the coordinator site failure may result in blocking, so a decision either to commit or abort Transaction (T) may have to be postponed until coordinate recovers.

b) Blocking problem: Consider a scenario, if a Transaction (T) holds locks on data-items of active sites, but amid the execution, if the coordinator fails and the active sites keep no additional log-record except <readt T> like <abort> or <commit>. So, it becomes impossible to determine what decisions has been made (whether to <commit> / <abort>). So, in that case, the final decision is delayed until the coordinator is restored or fixed. In some cases, this may take a day or long hours to restore and during this time-period, the locked data item remain inaccessible for other Transactions (Ti). This problem is known as Blocking-diagram

6) What are data fragmentations? State the various fragmentations with examples.

Ans. Data fragmentation in Distributed DBMS is a process of dividing the whole or full database into various subtables or sub relations so that data can be stored in different systems. The small pieces of sub relations or subtables are called fragments. These fragments are called logical data units and are stored at various sites. It must be made sure that the fragments are such that they can be used to reconstruct the original relation (i.e., there isn't any loss of data).

Page No.
Date : / /

In the fragmentation process, let's say, if a table T is fragmented and is divided into a number of fragments say $T_1, T_2, T_3 \ldots T_n$. The fragments contain sufficient information to allow the restoration of the original table T. This restoration can be done by the use of UNION or JOIN operation on various fragments. This process is called data fragmentation.

All of these fragments are independent which means these fragments cannot be derived from others. The user doesn't be logically concerned about fragmentation which means they should not concerned that the data is fragmented and this is called fragmentation independence or fragmentation transparency.

Advantages:

- As the data is stored close to the usage site, the efficiency of the database system will increase.
- Local query optimization methods are sufficient for some queries as the data is available locally.
- In order to maintain the security and privacy of the database system, fragmentation is advantageous.

Disadvantages:

- Access speeds may be very high if data from different fragments are needed.
- If we are using recursive fragmentation, it will be very expensive.

Page No.
Date : / /

The various fragmentation methods are :
- Horizontal fragmentation.
- Vertical fragmentation.
- Mixed or Hybrid fragmentation.

Horizontal fragmentation :

Horizontal fragmentation refers to the process of dividing a table horizontally by assigning each row or (a group of rows) of relation to one or more fragments. These fragments are then be assigned to different sides in the distributed system. Some of the rows or tuples of the table are placed in one system and the rest are placed in other systems. The rows that belong to the horizontal fragments are specified by a condition on one or more attributes of the relation. In relational algebra horizontal fragmentation on T, can be represented as follows :

$\sigma_P(T)$
where, σ is relational algebra operator for selection.
P is the condition satisfied by horizontal fragment.

Note that a union operation can be performed on the fragments to construct table T. Such a fragment containing all the rows of table T is called a complete horizontal fragment.

For example, consider an EMPLOYEE table(T).

Page No.
Date : / /

Eno	Ename	Design	Salary	Dep.
101.	A	abc	3000	1
102	B	abc	4000	1
103	C	abc	5500	2
104	D	abc	5000	2
105	E	abc	2000	2.

This EMPLOYEE table can be divided into different fragments like:

EMP1 = $\sigma_{DEP=1}$ EMPLOYEE

EMP2 = $\sigma_{DEP=2}$ EMPLOYEE

These two fragments are: T1 fragment of Dep = 1

Eno	Ename	Design	Salary	Dep.
101	A	abc	3000	1
102	B	abc	4000	1.

Similarly, the T2 fragment on basis of Dep = 2 will be:

Eno	Ename	Design	Salary	Dep.
103	C	abc	5500	2.
104	D	abc	5000	2
105	E	abc	2000	2.

Now, here it is possible to get back T as $T = T_1 \cup T_2 \cup \ldots T_n$.

Page No.
Date: / /

Vertical fragmentation:

Vertical fragmentation refers to the process of decomposing a table vertically by attributes or columns. In this fragmentation, some of the attributes are stored in one system and rest are stored in other systems. This is because each site may not need all columns of a table. In order to take care of restoration, each fragment must contain the primary key field(s) in a table. The fragmentation should be in such a manner that we can rebuild a table from the fragment by taking the natural JOIN operation and to make it possible we need to include a special attribute called Tuple-id to the schema. For this purpose, a user can use any super key. And by this, the tuples or rows can be linked together. The projection is as follows:

$$\Pi_{a1, a2, \ldots an}(T)$$

where, Π is relational algebra operator.

$a_1, a_2, \ldots a_n$ are attributes of T.

T is the table (relation).

For example, for the EMPLOYEE table we have T1 as:

Eno	Ename	Design	Tuple_id
101	A	abc	1
102	B	abc	2
103	C	abc	3
104	D	abc	4
105	E	abc	5

Page No.
Date : / /

For the second sub table of relation after vertical fragmentation is given as follows:

Salary	Dep	Tuple_id
3000	1	1
4000	2	2
5500	3	3
5000	1	4
2000	4	5

This is T2 and to get back to the original T, we join these two fragments T1 and T2 as $T_{EMPLOYEE}(T_1 \bowtie T_2)$

Mixed fragmentation:

The combination of vertical fragmentation of a table followed by further horizontal fragmentation of some fragments is called mixed or hybrid fragmentation. For defining this type of fragmentation we use SELECT and PROJECT operations of related algebra. In some situations, the horizontal and vertical fragmentation isn't enough to distribute data for some applications and in that condition, we need fragmentation called a mixed fragmentation.

↳ Mixed fragmentation can be done in two different ways:

1. The first method is to first create a set or group of horizontal fragments and then create vertical fragments from one or more of the horizontal fragments.
2. The second method is to first create a set or group of vertical fragments and then create horizontal fragments from one or more of the vertical fragments.

Page No.
Date: / /

the original relation can be obtained by the combination of JOIN and UNION operations which is given as follows:

$\sigma_p(\Pi_{a_1,a_2,\ldots,a_n}(T))$

$\Pi_{a_1,a_2,\ldots,a_n}(\sigma_p(T))$.

For example, for our EMPLOYEE table, below is the implementation of mixed fragmentation is $\Pi_{Ename,Design}(\sigma_{Eno<104}(EMPLOYEE))$. The result of this fragmentation is,

Ename	Design
A	abc
B	abc
C	abc.

7. Write short notes on:

b. QBE:

If we talk about normal queries we fire on the database they should be correct and in a well-defined structure which means they should follow a proper syntax if the syntax or query is wrong definitely we will get an error and due to that our application or calculation definitely going to stop. So to overcome this problem QBE was introduced. QBE stands for Query By Example and it was developed in 1970 by Moshe Zloof at IBM.

It is a graphical query language where we get a user interface and then we fill some required fields to get our proper result. In SQL, we will get an error if the query is not correct but in the case of QBE if the query is wrong either we get a wrong answer or the query will not be going to execute but we will never get any error.

Page No.
Date : / /

Example;
Consider the example where a table 'SAC' is present in the database with Name, phone-Number and Branch fields. And we want to get the name of the SAC-Representative name who belongs to the MCA branch. If we write this query in SQL we have to write it like:

```
SELECT NAME
FROM SAC
WHERE BRANCH = 'MCA'"
```

And definitely, we will get our correct result. But in the case of QBE, it may be done as like there is a field present and we just need to fill it with "MCA" and then click on the SEARCH button and we will get our required result.

Points about QBE:
- Supported by most of the database programs.
- It is a Graphical Query language.
- Created in parallel to SQL development.

Object Relational Model:
The object-relational model is designed to provide a relational database management that allows developers to integrate databases with their data types and methods. It is essentially a relational model that allows users to integrate object-oriented features into it. This design is most recently shown in the Norsdic Object/Relational Model. The primary function of this new object-relational model is to power, greater flexibility, greater

Page No.
Date: / /

performance, and greater data integrity then those that came before it.

Some of the benefits that are offered by the Object-Relational Model include:

- Extensibility: Users are able to extend the capability of the database server; this can be done by defining new data types, as well as user-defined patterns. This allows the user to store and manage data.
- Complex types: It allows users to define new data types that combine one or more of the currently existing data types. Complex types aid in better flexibility in organizing the data on a structure made up of columns and tables.
- Inheritance: Users are able to define objects or types and tables that procure the properties of other objects, as well as add new properties that are specific to the object that has been defined.
- A field may contain an object with attributes and operations.
- Complex objects can be stored in relational tables.

The object-relational database management system which are also known as ORDBMS, these systems provide an addition of new and extensive object storage capabilities to the relational models at the center of the more modern information systems of today. These services assimilate the management of conventional fielded data, more complex objects such as a time-series or more detailed geospatial data and varied multimedia such as audio, video, images and applets.

Page No.
Date : / /

This can be done to the model working to summarize methods. With data structures, the ORDBMS server can implement complex analytical data and data management operations to explore and change multimedia and other more complex objects.

Disadvantages of ORDBMSs:

- The ORDBMS approach has the obvious disadvantages of complexity and associated increased costs. Further, there are the proponents of the relational approach that believe the essential simplicity and purity of the relational model are lost with these type of extension.

Page No.
Date : / /

Year : 2019 ; Semester : Spring.

1.a) Explain the concept of DBMS and its applications tracing the evolution.

Ans. A Database Management system is a collection of interrelated data and a set of programs to access those data. The main purpose of DBMS is to provide a way to store and retrieve database information that is both convenient and efficient. Management of data involves both defining structures for storage of information and providing mechanisms for the manipulation of information. In addition, the database system must ensure the safety of the information store, despite system crashes or attempts at unauthorized access.

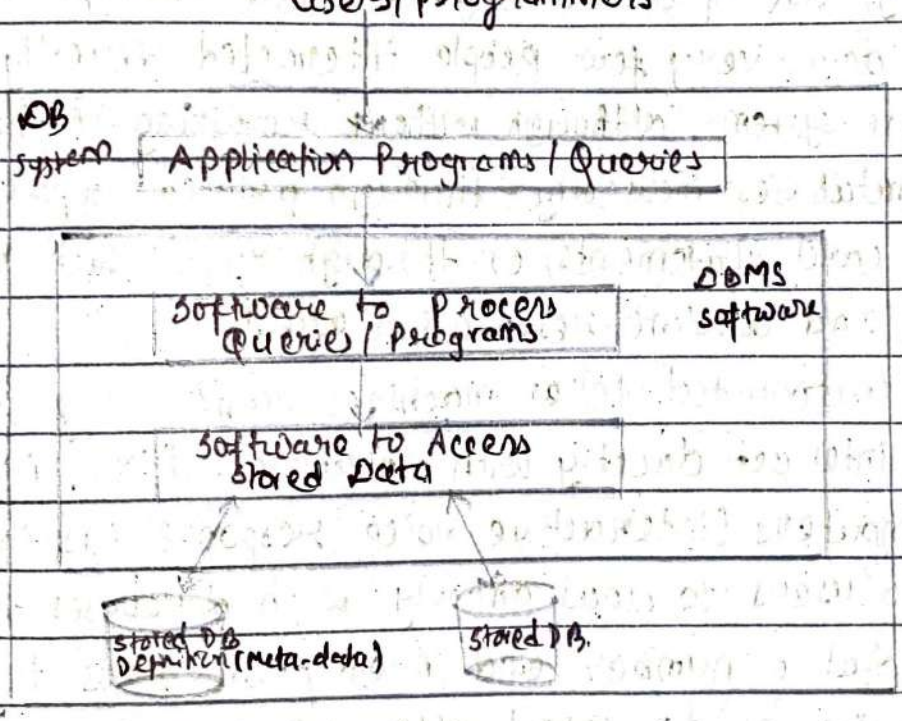


fig : A simplified database system environment.

Applications of Database System:

Databases are widely used. Here are some representative applications:

Page No.
Date : / /

- Banking : For customer information, accounts, and loans, and banking transactions.
- Airlines : For reservations and schedule information.
- Universities : for student information, course registration and grades.
- Credit card transactions : for purchase on credit cards and generation of monthly statements.
- Telecommunication, finance, sales, Manufacturing, Human resources and soon.

Evolution of DBMS :

- Over the course of the last four decades of the twentieth century, use of database grew in all enterprises. In the early days, very few people interacted directly with database systems, although without realizing it they interacted with databases indirectly - through printed reports such as credit card statements, or through agents such as bank tellers and airline reservation agents.
- Then automated teller machines came along and let user interact directly with databases. Phone interfaces to computers (interactive voice response system's) also allowed users to deal directly with databases - a caller could dial a number, and press phone keys to enter information on to select alternative options, to find flight arrival/departure times, for example, or to register for courses in a university.
- The internet revolution of the late 1990s sharply increased direct user access to databases. Organizations converted many of their phone interfaces to databases

Page No.
Date : / /

into Web interfaces, and made a variety of services and information available online. For instance, when you access an online bookstore and browse a book or music collection, you are accessing data stored in a database.

- Moreover, now there is Multimedia databases which can store pictures, video chips and sound messages. Geographic information systems (GIS) to store and analyze maps, ~~where~~ weather data and satellite image. Data warehouse and Online Analytic Processing (OLAP) systems are used in many companies to extract and analyze useful information for decision making.

Page No.
Date : / /

2.a) Using the following schema represent the following queries using Relational algebra:

PROJECT (Project num, projectName, project Type, Project Manager).

EMPLOYEE (Empnum, Empname)

ASSIGNED_TO (Projectnum, Empnum).

i) find Employee details working on a project name starts with 'L'.

Ans. $\prod_{employee.empnum, empname.} \left[\sigma_{projectnum\ like\ 'L\%'} (Project \bowtie (Employee \bowtie Assigned\_to)) \right]$

ii) List all the employee details who are working under project manager "Rohan".

Ans. $\prod_{employee.empnum, empname} \left[\sigma_{ProjectManager = 'Rohan'} (Project \bowtie (Employee \bowtie Assigned\_to)) \right]$

iii) List the employees who are still not assigned with any project.

Ans. $\prod_{employee.empnum, empname} (\sigma_{projectnum\ IS\ NULL} (Employee \bowtie Assigned\_to))$

iv) List the employees who are working in more than one project.

Ans.
```
SELECT * FROM
EMPLOYEE
. INNER JOIN ASSIGNED_TO
ON EMPLOYEE. Empnum = ASSIGNED_TO. Empnum
WHERE count(*) > 1;
```

$\prod_{employee.empnum, empname} (\sigma_{G\ count(empnum) > 1} (employee \bowtie Assigned\_to))$;

Page No.
Date: / /

b) Write the SQL statements for the following queries by referen
of Hotel_details relation:

hotel_id	hotel_name	estb_year	hotel_star	hotel_worth
1	Hyatt	2047	Five	15M
2	Hotel Ktm	2043	Three	5M
3	Fulbari	2058	Five	20M
4	Yak and Yeti	2052	Four	11M
5	Hotel Chitwan	2055	Three	7M.

i. Create a database named hotel & table relation.

Ans.
```
CREATE DATABASE hotel;
USE hotel;
CREATE TABLE Hotel_details (
    hotel_id int PRIMARY KEY,
    hotel_name varchar(50),
    estb_year int(4),
    hotel_star varchar(10),
    hotel_worth varchar(5));
```

ii. Create a view named Price which shows hotel name & its worth.

Ans.
```
CREATE VIEW Price AS
SELECT hotel_name, hotel_worth
FROM hotel_details;

SELECT * FROM Price;
```

Page No.
Date : / /

iii) Modify the data so that Hotel chitwan is now four star level.

Ans.
```
UPDATE hotel_details SET
    hotel_star = "Four"
    WHERE
    hotel_name = "Hotel Chitwan";
```

iv) Delete the records of all hotels having worth more than 9M.

Ans.
```
DELETE FROM hotel_details
    WHERE
    hotel_worth > 9M;
```

3. a) What are store procedures? Explain equi Join, natural join, left and right outer join with example.

Ans. Stored procedure:

A stored procedure in SQL is a type of pre-written code that can be stored for later execution and then used many times hence, saving time. It is a group of SQL statements that performs the task. The stored praedure can be invoked explicitly whenever required. It may accept some inputs in the form of parameters, these may be one parameter or multiple parameters. A procedure won't return values. To create a procedure, we use the CREATE PROCEDURE command.

Page No.
Date: / /

Join in DBMS is a binary operation which allows us to combine join product and selection in one single statement. The goal of creating a join condition is that it helps us to combine the data from two or more DBMS tables.

Types of Join:
There are mainly two types of joins in DBMS:
1. Inner Joins : Theta, Natural, EQUI.
2. Outer Join : Left, Right, Full.

EQUI Join:

EQUI join is done when a Theta join uses only the equivalence condition. EQUI join is the most difficult operation to implement efficiency in an RDBMS, and one reason why RDBMS have essential performance problems.

For example:

A ⋈ A.column2 = B.column2 (B)

column1	column 2
1	1.

Natural Join (⋈):

Natural join doesnot utilize any of the comparison operators. In this type of join, the attributes should have the same name and domain. In Natural join, there should be at least one common attribute between two relation.

It performs selection forming equality on those attributes which appears in both relation and eliminates the duplicate attributes.

Page No.
Date : / /

Example:

Consider the following two tables;

C

Num	Square
2	4
3	9

D

Num	Cube
2	8
3	18

Now, C ⋈ D.

Num	Square	Cube
2	4	8
3	9	18

Outer Join:

An outer join doesn't require each record in the two join tables to have a matching record. In this type of join, the table retains each record even if no other matching record exists.

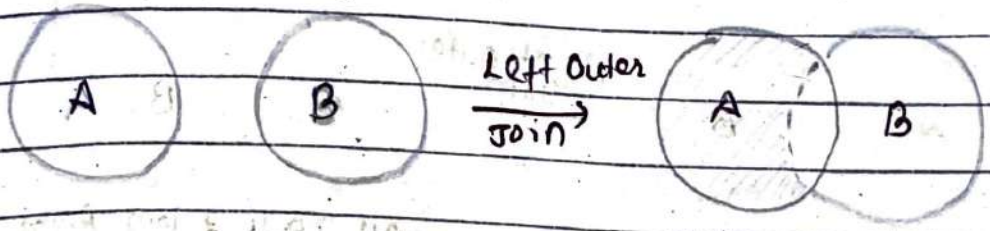
Three types of Outer Join are:

- Left Outer Join.
- Right Outer Join.
- Full Outer Join.

Left Outer Join (A ⟕ B).

Left Outer Join returns all the rows from the table on the left even if no matching rows have been found in the table on the right. When no matching record is found in the table on right, NULL is returned.

Page No.
Date : / /



All rows from Left table

Consider the following 2 Tables;

A

Num	Square
2	4
3	9
4	16

B

Num	Cube
2	8
3	18
5	75

A ⋈ B.

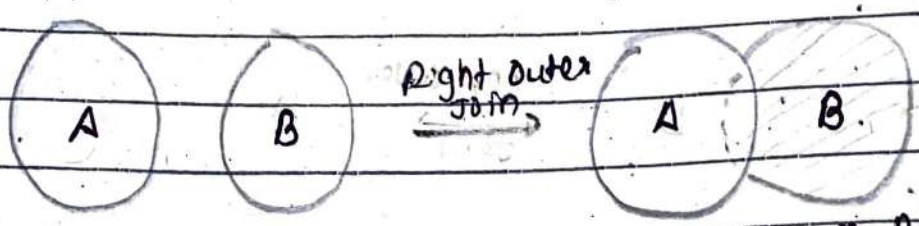
Num	Square	Cube
2	4	8
3	9	18
4	16	-

Right Outer Join (A ⟖ B):

Right Outer Join returns all the columns from the table on the right even if no matching rows have been found in the table on the left. Where no matches have been found in the table on the left, NULL is returned. RIGHT outer JOIN is the opposite of LEFT JOIN.

In our example, let's assume that you need to get the names of members and movies rented by them. Now we have a new member who has not rented any movie yet.

Page No.
Date : / /



All rows from Right table.

A ⟗ B.

Num	Cube	Square
2	8	4
3	18	9
5	125	-

b) Differentiate betⁿ functional dependency and multi-valued dependency? Explain closure set of functional dependencies with example.

Ans

Page No.
Date: / /

Closure set of functional dependencies:

Given a relational schema R, a functional dependencies f on R is logically implied by a set of functional dependencies F on R if every relation instance r(R) that satisfies F also satisfies f.

The closure of F, denoted by F+ is the set of all functional dependencies logically implied by F.

The closure of F can be found by using a collection of rules called Armstrong axioms.

- Reflexivity rule: If A is a set of attribute and B is subset or equal to A, then A → B holds.
- Augmentation rule: If A → B holds and C is a set of attributes, then CA → CB holds.
- Transitivity rule: If A → B holds and B → C holds. then A → C holds.
- Union rule: If A → B holds and A → C then A → BC holds.
- Decomposition rule: If A → BC holds, then A → B holds and A → C holds.

Date : / /

- Pseudo Transitivity rule: If $A \to B$ holds and $BC \to D$ holds, then $AC \to D$ holds.

Suppose we are given a relation schema $R = (A, B, C, G, H, I)$ and the set of function dependencies.

$A \to B$, $A \to C$, $CG \to H$, $CG \to I$, $B \to H$.

- We list several members of F+ here:

$A \to H$; since $A \to B$ and $B \to H$ hold, we apply transitivity rule.

$CG \to HI$; since $CG \to H$ and $CG \to I$; union rule.

$AG \to I$; since $A \to C$, and $CG \to I$; the pseudo transitivity rule implies that $AG \to I$ holds.

$A \to BC$: since $A \to B$ and $A \to C$ hold; union rule.

$AG \to H$, since $A \to C$ and $CG \to H$; the pseudo transitivity rule implies $AG \to H$ holds.

4. a) Define third normal form. Convert the following 2NF relation into 3NF (consider Name as primary key).

Name	Address	Phone	Salary	Post
Gill	KTM	126	600	Engineer
Van	BKT	348	600	Engineer
Robert	KTM	126	600	Engineer
Broon	BKT	348	800	Overseer
Albert	KTM	90	800	Officier.

Ans. According to Codd's original definition, a relation schema R is in 3NF if it satisfies 2NF and no nonprime attribute of R is transitively dependent on primary key.

3NF is based on concept of transitive dependency. A functional dependency $X \to Y$ in a relation schema R is

Page No.
Date: / /

transitive dependency. if there exists a set of attributes Z in R that is neither a candidate key nor a subset of any key of R, and both $X \rightarrow Z$ and $Z \rightarrow Y$ hold.

Name	address	phone	~~salary~~ Post

Phone	Salary	Post
126	600	Eng.
348		
348		

Post	Salary
Eng.	600
or	-
4	-

b) What is security and integrity violations? Explain the need of access control, Authorization and Authentication.

Ans. The process of disobeying the sets of rules defined and going against the security mechanism and data integrity mechanism are called security and integrity violations.

The need of access control, Authorization and Authentication comes in database in order to handle security of the database system.

Page No.
Date : / /

5. a) What is query cost estimation? Explain cost based & heuristic based choice of evaluation plan for query optimization.

Ans. The total elapsed time for answering (processing) the given query is query cost estimation. The cost of query evaluation can be measured in terms of a number of different resources, including disk accesses, CPU time to execute a query, and, in a distributed or parallel database system, the cost of communication.

Second part:

For query optimization, various techniques and methods can be used. From these startegies, we can always optimize our quering performance, with the

Page No.
Date : / /

minimum effort.
For any given query, there may be a number of different ways to execute it. The process of choosing a suitable one for processing a query is known as query optimization. The two forms of query optimization are as follows:

- Heuristic optimization:
  Here the query execution is refined based on heuristic rules for reordering the individual operations.

- Cost based optimization:
  The overall cost of executing the query is systematically reduced by estimating the cost of executing several different execution plans.

Example;
select name from customer, account where customer.name = account.name and account.balance > 2000;
There are two evaluation plans:

(1) $\Pi_{customer.name}(\sigma_{customer.name = account.name \wedge account.balance > 2000}(customer \times account))$.

(2) $\Pi_{customer.name}(\sigma_{customer.name = account.name}((customer \times \sigma_{account.balance > 2000}(Account))$.

Cost evaluator evaluates the cost of different evaluation plans and chooses the evaluation plan with lowest cost. Disk access time, CPU time, number of operations, number of tuples, size of tuples are considered for cost calculations.

Page No.
Date : / /

Heuristic approach is also called rule-based optimization. There are three ways for transforming relational-algebra queries are:

- Perform the SELECTION process foremost in the query. This should be the first action for any SQL table. By doing so, we can decrease the number of records required in the query, rather than using all the tables during the query.
- Perform all the projection as soon as achievable in the query. Somewhat like a selection but this method helps in decreasing the number of columns in the query.
- Perform the most restrictive joins and selection operations. What this means is that select only those set of tables and/or views which will result in a relatively lesser number of records and are extremly necessary in the query. Obviously any query will execute better when tables with few records are joined.

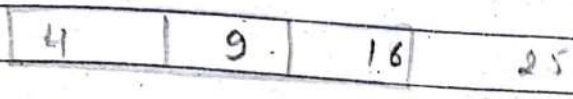
5.b) Create a B+ tree of order 4 with following data; (4, 9, 16, 25, 1, 20, 13, 15, 10, 11, 12) of order 4. Assume that, tree is initially empty and values are added in ascending order. Also, show the formation of tree after the deletion of 16.

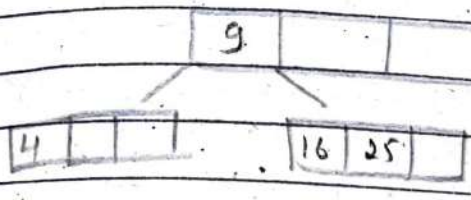
Ans. Order (P) = 4

Max keys = P - 1 = 3.

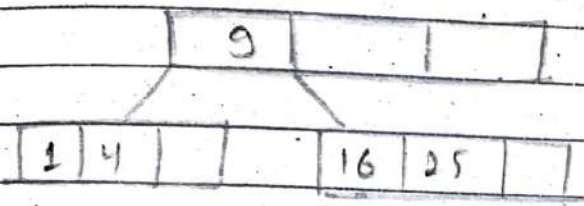
Min = $\left\lceil \frac{P}{2} \right\rceil - 1 = 1$.

Page No.
Date: / /

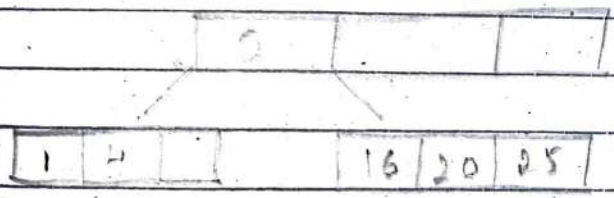




Insert 1.

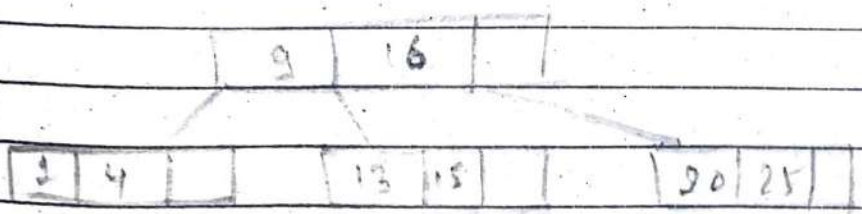


Insert 20.

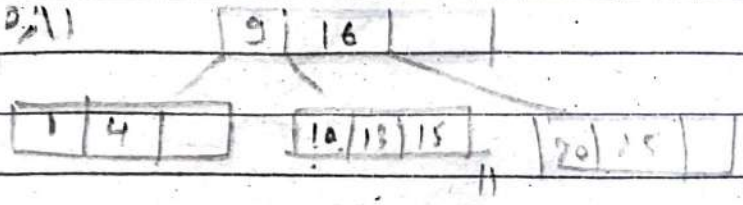


[13, 16, 20] 25

Insert 13, 15

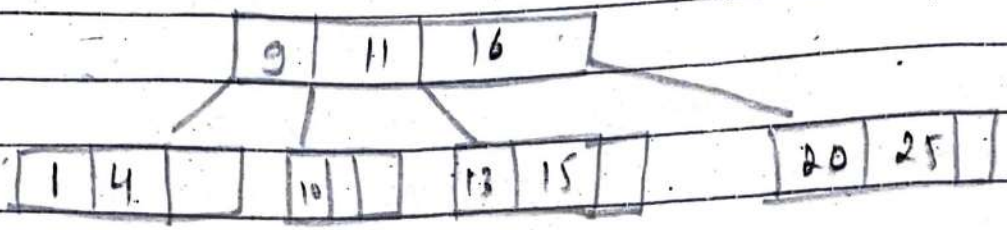


Insert 10, 11

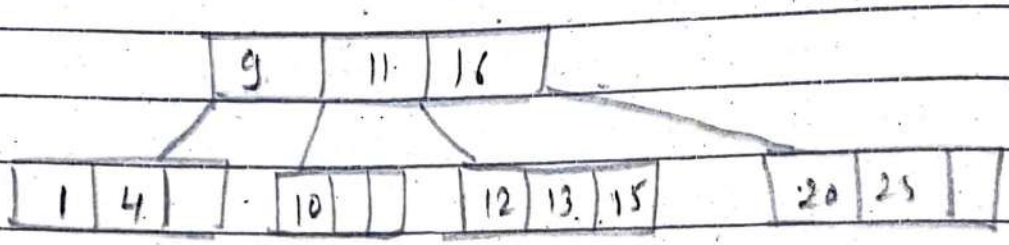


[10, 11, 13] 15.

Page No.
Date : / /



Insert 12 :



(e1) Delete 16.

Page No.
Date: / /

6.a) What is concurrency control? Describe ACID property of transaction.

Ans. Concurrency control concept comes under the Transaction in database management system. It is a procedure in DBMS which help us for the management of two simultaneous processes to execute without conflicts between each other, these conflict occur in multi user system.

Concurrency can simply be said to be executing multiple transactions at a time. It is required to increase time efficiency. If many transactions try to access the same data, then inconsistency arises. Concurrency control required to maintain consistency data.

For example; if we take ATM machine and donot use concurrency, multiple persons cannot draw money at a time in different places. This is where we need concurrency.

Advantages;

- Waiting time will be decreased.
- Response time will decrease.
- Resource utilization will increase.
- System performance and Efficiency is increased.

ACID property of Transaction.

A transaction is a single logical unit of work that accesses and possibly modifies the contents of a database. Transactions access data using read and write operations. In order to maintain consistency in database, before and after transaction, certain properties are followed. These are ACID properties.

Page No.
Date : / /

Atomicity:

By this, we mean that either the entire transaction takes place at once or doesn't happen at all. There is no midway. i.e. transactions do not occur partially. Each transaction is considered as one unit and either runs to completion or is not executed at all. It involves the following two operations.

- Abort: If a transaction aborts, changes made to the database are not visible.
- Commit: If a transaction commits, changes are visible.

Atomicity is also known as 'All or nothing rule'.

Consistency:

This means that integrity constraints must be maintained so that the database is consistent before and after the transaction. It refers to the correctness of a database.

Isolation:

This property ensures that multiple transactions can occur concurrently without leading to the inconsistency of the database state. Transactions occur independently without interference. Changes occurring in a particular transaction will not be visible to any other transaction until that particular change in that transaction is written to memory or has been committed. This property ensures that the execution of transactions concurrently will result in a state that is equivalent to a state

Page No.
Date: / /

achieved these were executed serially in some order.

Durability:

This property ensures that once the transaction has completed execution, the updates and modifications to the databases are stored in and written to disk and they persist even if a system failure occurs. These updates now become permanent and are stored in non-volatile memory. The effects of the transactions, are never lost.

In Gist; Atomicity = Transaction Manager.
Consistency = Application programmer.
Isolation = Concurrency control Manager.
Durability = Recovery Manager.

The ACID properties, in totality, provide a mechanism to ensure the correctness and consistency of a database in a way such that each transaction is a group of operations that acts as a single unit, produces consistent results, acts in isolation from other operations, and updates that it makes are durably stored.

b) Define recovery. When the two transactions are said to be in deadlock state? How these deadlocks can be addressed.

Ans. Database recovery is the process of restoring the database to a correct (consistent) state in the event of a failure. In other words, it is the process of restoring the database to the most recent consistent state that existed shortly before the time of system failure.

Page No.
Date : / /

Two transactions are said to be in deadlock state if each transaction is waiting for a data item that is being locked by some other transaction.

Deadlock Avoidance:

When a database is stuck in a deadlock, it is always better to avoid the deadlock rather than restarting or aborting the database. The deadlock avoidance method is suitable for smaller databases whereas the deadlock prevention method is suitable for larger databases.

One method of avoiding deadlock is using application-consistent logic and another method for avoiding deadlock is to apply both row-level locking mechanism and READ COMMITTED isolation level. However, It does not guarantee to remove deadlocks completely.

Deadlock Detection:

When a transaction waits indefinitely to obtain a lock, the database management system should detect whether the transaction is involved in a deadlock or not.

Wait-for-graph is one of the methods for detecting the deadlock situation. This method is suitable for smaller databases. In this method, a graph is drawn based on the transaction and their lock on the resource. If the graph created has a closed-loop or cycle, then there is a deadlock.

↳ time based
↳ usage Depending.

Page No.
Date : / /

Deadlock prevention:

For a large database, the deadlock prevention method is suitable. A deadlock can be prevented if the resources are allocated in such a way that deadlock never occurs. The DBMS analyzes the operations whether they can create a deadlock situation or not, if they do, that transaction is never allowed to be executed.

Deadlock prevention mechanism proposes two schemes: (time based)

- Wait-die scheme:

In this scheme, if a transaction requests a resource that is locked by another transaction, then the DBMS simply checks the timestamps of both transactions and allows the older transactions to wait until the resource is available for execution. This scheme allows the older transaction to wait but kills the younger one.

- Wound wait scheme:

In this scheme, if an older transaction requests for a resource held by a younger transaction, then an older transaction forces a younger transaction to kill the transaction and release the resource. The younger transaction is restarted with a minute delay but with the same timestamp. If the younger transaction is requesting a resource that is held by an older one, then the younger transaction is asked to wait till the older one releases it.

Page No.
Date : / /

7. Write short notes on :

a) Architecture of Distributed Database:

A distributed database system consists of a collection of sites, each of which maintains a local database system. Each site is able to process local transactions; those transactions that access data in only that single site. In addition, a site may participate in the execution of global transactions; those transactions that access data in several sites. The execution of global transaction requires communication among the sites. The general structure of a distributed system appears in figure below.

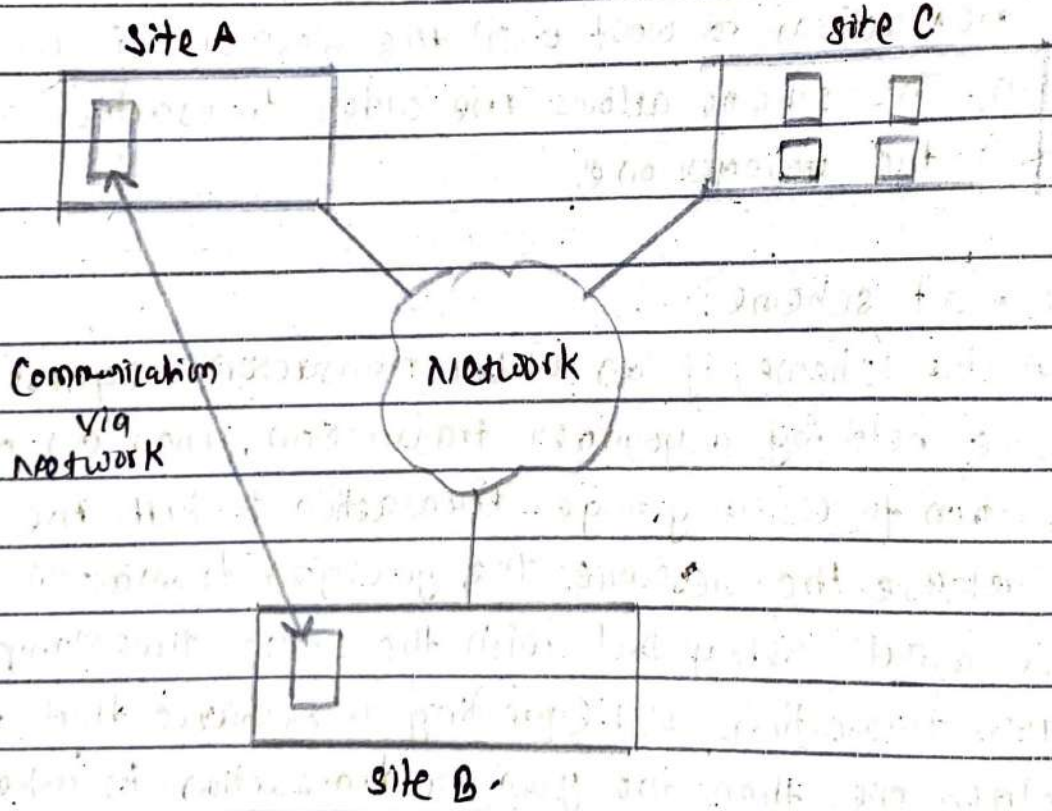


fig: A distributed database system.

There are several reasons for building distributed database systems, including sharing of data, autonomy, & availability.

Page No.
Date: / /

- Sharing data: The major advantage in building a distributed database system is the provision of an environment where users at one site may be able to access the data residing at other sites. For instance, in a distributed banking system, where each branch stores data related to that branch, it is possible for a user in one branch to access data in another branch. Without this capability, a user wishing to transfer funds from one branch to another would have to resort to some external mechanism that would couple existing system.

- Autonomy: The primary advantage of sharing data by means of data distribution is that each site is able to retain a degree of control over data that are stored locally. In a centralized system, the database administrator of the central site controls the database. In the distributed system, there is a global database administrator responsible for entire system. A part of these responsibilities is delegated to the local database administrator for each site. Depending on the design of the distributed database system, each administrator may have a different degree of local autonomy: The possibility of local autonomy is often a major advantage of distributed databases.

- Availability: If one site fails in a distributed system, the remaining sites may be able to continue operating. In particular, if data items are replicated in several sites, a transaction needing a particular data item may find that item in any of several sites. Thus, the failure of a site doesnot necessarily imply the shutdown of the system.

Page No.
Date : / /

c. Dense and Sparse Index:

We know that data is stored in the form of records. Every record has a key field, which help it to be recognized uniquely.

Indexing is a data structure technique to efficiently retrieve records from the database files based on some attributes on which the indexing has been done. Indexing in database system is similar to what we see in books.

There are two basic kinds of indices:
- Ordered indices: search keys are stored in sorted order.
- Hash indices: search keys are distributed uniformly across "buckets" using a "hash function".

There are three types of ordered indices:
- Dense index
- Sparse index
- Multi-level indexing.

Dense indexing:

In a dense index, for every search-key in the file, an index entry is present. In a dense-clustering index, the index record contains the search key value and a pointer to the first data record with that search-key value. The remaining number of records with similar search-key value would be stored in a sequence after the first record.

Page No.
Date: / /

Sparse Indexing:

In a sparse index, an index entry appears for only some of the search-key values. These indicies can be used only if the relation is stored in sorted order of search-key value. To locate a record, one has to find the index entry with the largest search-key value that is less than or equal to the search-key value for which we are looking. We start looking from the record pointed to by that index entry, and then follow the pointers in the file until we find the desired record.

Page No.
Date : / /

Year 2020 ; Semester : Fall.

1.a) Why data independence is important in data modeling? Differentiate between physical and logical data independence.

Ans: Data independence is really important in data modeling. Because from only data independence realization, data schema change at one level won't bother the structure and working of data at another end.

There are two types of data independence and they are:

- Physical data independence
- Logical data independence.

The differences bet^n logical & physical data independence are:

Logical data independence	Physical Data Independence
• It is mainly concerned with the structure or changing the data definition.	• Mainly concerned with the storage of the data.
• It is difficult as retrieving of data is mainly dependent on the logical structure of data.	• It is easy to retrieve
• One need to make changes in Application program if new fields are added or deleted from database.	• A change in physical level usually does not need change at the Application program level.

Page No.
Date : / /

• Modification at the logical level significant whenever the logical structures of the database are changed.	• Modifications made at the interval levels may or may not needed to improve the performance of the structure.
• Concerned with conceptual schema.	• Concerned with internal schema.
• Example : Add / Modify / Delete a new attribute.	• Example : change in compression techniques, hashing algorithms, storage devices, etc.

b) Define and explain benefits of data model. Draw an ER diagram for a Vechile Management System including primary key, weak entity, composite attribute, derived attribute and multi-valued attributes in your ER diagram.

Ans. Data model gives us an idea how the final system will look like after its complete implementation. It defines the data elements and the relationships between the data elements. Data models are used to show how data is stored, connected, accessed and updated in the database management system. Here, we use a set of symbols and text to represent the information so that members of the organization can communicate and understand it. Though there are many data models being used nowadays but the relational model is the most widely used model. Apart from the Relational model, there are many other types of data models about which are listed below: Hierarchical Model, Network Model, Entity - Relationship Model, Relational Model, Object - Oriented Data Model etc.

Page No.
Date: / /

The various benefits of creating data model are:

→ They help to document the entire system.
→ They help to make templates for the construction in the system.
→ They help in visualization of the system.
→ The structural dimension of a system can be specified.
→ Modeling is entirely accepted by engineering technique.

9) Explain Relational Algebra. What are the relational algebra operations that can be performed? Give an example of all.

Page No.
Date : / /

Ans. Relational algebra is a collection of operations on Relations. Relations are operands and the result of an operation is another relation.

Two main collection of relational operators:

- set Theory operations:
  → Union, Intersection, difference and Cartesian product.

- specific relational operations:
  → selection, projection, Join, Division.

Consider the following relations R and S.

First	last	Age
Bill	Smith	22
Sally	Green	28
Mary	Keen	23
Tony	Jones	32.

(R)

First	last	Age
Forrest	Gump	36
Sally	Green	28
DonJuan	DeMarco	27.

(S)

Union (R ∪ S):

First	last	Age
Bill	Smith	22
Sally	Green	28
Mary	Keen	23
Tony	Jones	32
Forrest	Gump	36
DonJuan	DeMarco	27

Difference (R - S):

First	last	Age
Bill	Smith	22
Mary	Keen	23
Tony	Jones	32.

Page No.
Date : / /

Intersection: (R ∩ S).

First	Last	Age
Sally	Green	28

let us convider another example:

First	Last	age
Bill	Smith	22
Mary	Keen	23
Tony	Jones.	23

(R)

Dinner	Desert
Steak	Ice Cream
Lobster	Cheesecake

(S)

So its cartesian product : R x S.

First	last	Age	Dinner	Desert.
Bill	Smith	22	Steak	Ice Cream.
Bill	Smith	22	Lobster	Cheesecake.
Mary	Keen	23	Steak	Ice Cream.
Mary	Keen	23	Lobster	Cheesecake
Tony	Jones	22	Steak	Ice Cream.
Tony	Jones	22	Lobster	Cheesecake.

let us assume EMP table.

Name	Office	Dept
Smith	400	CS
Jones	220	Econ
Green	160	Econ.

Page No.
Date: / /

Select only those employees in 'Econ' department.

$\sigma_{Dept = 'Econ'}(EMP)$.

Name	Office	Dept
Jones	220	Econ
Green	160	Econ.

to.

Project only name and departments of the employee.

$\Pi_{name, dept}(EMP)$.

Name	Dept
Smith	CS
Jones	Econ
Green	Econ

let us assume two relations EMP and DEPART.

Name	Office	Dept	Salary.
Smith	400	CS	45000
Jones	220	Econ	35000
Green	160	Econ	50000
Brown	420	CS	65000
Smith	500	Fin	60000

Dept.	MainOffice	Phone
CS	404	555-1212
Econ	200	555-1234
Fin	501	555-4321
Hist	100	555-9876

for Join, we use $\bowtie$ symbol. So we will be joining EMP and DEPART relation by the name of department.

ie. EMP $\bowtie_{emp.Dept = depart.Dept}$ DEPART.

Page No.
Date : / /

Name	Office	Emp. Dept	Salary	Depart. Dept	Manager	Phone
Smith	400	CS	45000	CS	404	555-1212
Jones	220	Econ	35000	Econ	200	555-1234
Green	160	Econ	50000	Econ	200	555-1034
Brown	420	CS	65000	CS	404	555-1211
Smith	500	Fin	60000	Fin	501	555-4321

b> Write SQL statement for following:

i. Create a table named Automotor with chasis_number as primary key and following attributes:
veh_brand, veh_name, veh_model, veh_year, veh_cost, veh_color, veh_weight.

Ans.
```
CREATE TABLE Automotor (
    chasis_number int PRIMARY KEY,
    veh_brand varchar(20),
    veh_name varchar(30),
    veh_model varchar(10),
    veh_year int(4),
    veh_cost decimal(10,2),
    veh_color varchar(15),
    veh_weight decimal(5,2));
```

ii. Enter a full detailed information of an automotor.

→
```
INSERT INTO Automotor VALUES
(1234, 'Toyota', 'My Toyo', 'A', 2015, 200000, 'red', 120.73)
(4567, 'Bernault', 'The Bes', 'X', 2017, 400000, 'black', 200);
```

Page No.
Date: / /

iii. Change only Automotor's year to 2019.

Ans UPDATE Automotor SET veh_year=2019 WHERE chasis_number=1234;

iv. Remove all Automotor records whose model contains character 'i' in last position.

Ans. DELETE FROM Automotor WHERE veh_model LIKE '%i';

v. Display the total cost of all vechiles of the table Automotor.

Ans SELECT SUM(veh_cost) FROM Automotor;

vi. Create a View from above table having vechiles only red color.

Ans. CREATE VIEW redColorOnly AS
SELECT * FROM
Automotor
WHERE
veh_color = "red";

vii. Change datatype of color so that it only takes one character.

Ans. ALTER TABLE Automotor
MODIFY v veh_color varchar(1);
Column

3.a) Differentiate between join and sub query. Explain different SQL joins with example.

Ans. A SQL Join statement is used to combine data or rows from two or more tables based on a common field between them. A subquery is a query that is nested ~~inside~~ inside a

Page No.
Date : / /

SELECT, INSERT, UPDATE or DELETE statement or inside another subquery. Joins and subqueries are both used to combine data from different tables into a single result. The main difference between them is for join: we have to join relation based on the similar corresponding attributes, whereas for ~~select~~ the sub queries a query is executed inside main query, and its result is used as a value for firing/executing the main query.

SQL sub query or inner query or nested query is a query within another sql query and embedded within WHERE clause. Subquery returns data that will be used in main query as a condition to further restrict data to be retrieved. The SQL Joins clause is used to combine records from two or more tables in a database. A JOIN is a means for combining table fields from two or more table by using values common to each.

SQL Join types:

- INNER JOIN: Returns rows when there is a match in both tables.
- LEFT JOIN: returns all rows from the left table, even if there are no matches in the right table.
- RIGHT JOIN: returns all rows from the right table, even if there are no matches in the left table.
- FULL JOIN: Returns rows when there is a match in one of the tables.

Page No.
Date: / /

~~Inner~~ Sub query example:

```
SELECT * FROM employee WHERE salary > (select max(salary)
FROM employee WHERE did = 1);
```

INNER JOIN example:

```
SELECT employee.ename, department.location
FROM employee
INNER JOIN department
ON
employee.did = department.did;
```

LEFT JOIN example:

```
SELECT employee.ename, department.location
FROM employee
LEFT JOIN department
ON
employee.did = department.did.
```

b) What is functional dependency? Discuss its types. Explain the role of functional dependency in the process of normalization.

Ans. Functional dependency in DBMS, as the name suggest is a relationship between attributes of a table dependent on each other. Introduced by E.F. Codd it helps in preventing data redundancy and gets to know about bad designs.

To understand the concept thoroughly, let us consider P is a relation with attributes A and B. Functional Dependency is represented by → (arrow sign).

Then the following will represent the functional dependency between attributes with an arrow sign. i.e. $A \rightarrow B$.

Page No.
Date : / /

Types :

1. Fully functional dependency :

$X \rightarrow Y$ is a full functional dependency if the removal of any attribute A from X removes the dependency. A full functional dependency occurs when it is already functional dependency and the set of attributes on the left side of FD can't be reduced any further.

E.g:

Order No	Line No	qty	price
A001	001	10	200
A002	001	20	400
A002	002	30	800
A004	001	15	300.

here, {Order No, Line No} → qty.
{Order No, Line No} → price.
are full functional dependencies.

2. Partial functional dependency :

A F.D. $X \rightarrow Y$ is partial functional dependency if some attribute $A \in X$ can be removed from X and the dependency still holds.

E.g: {eid, phone} → name.

Here, if we remove phone then,
{eid} → name still holds, so it is partial functional dependency.

Page No.
Date: / /

4ne role:

Normalization process uses functional dependencies to seek out and eliminate redundancy in a database schema, thereby reducing the possibility of update anamolies. If a database schema is properly normalized, the following should hold true for all tables: all columns should be functionally dependent on the table's primary key. Functional dependencies allow you to verify that this is true and if it is not, determine the steps to take to normalize the tables so that it will be true (without losing any data or losing any connections between data that is related).

4.a) What is multi-valued dependency? Illustrate the advantage of 4NF with suitable example.

Ans.
- Multivalued dependency occurs when two attributes in a table are independent of each other but, both depend on third attribute.
- A multivalued dependency consists of at least two attributes that are dependent on a third attribute that's why it always requires at least three attributes.

For example, let us consider there is a bike manufacturer company which produces two colors (white and black) of each model every year. For this we have a table having BIKE_MODEL, MANUF_YEAR and COLOR as the column names.

Here columns COLOR and MANUF_YEAR are dependent on BIKE_MODEL and independent of each other. In this case, these two columns can be called as multivalued dependent on BIKE_MODEL. The representation of these dependencies is:

Page No.
Date : / /

BIKE_MODEL →→ MANUF_YEAR
BIKE_MODEL →→ COLOR.

This can be read as "BIKE_MODEL multidetermined MANUF_YEAR" and "BIKE_MODEL multidetermined COLOR".

This same concept of multi-valued dependency is used in 4NF.

Fourth Normal form (4NF):

- A relation will be in 4NF if it is in Boyce Codd normal form and has no multi-valued dependency.
- For a dependency A → B, if for a single value of A, multiple values of B exists, then the relation will be a multi-valued dependency.

Example : let us consider a STUDENT table.

Stu_ID	COURSE	HOBBY.
21	Computer	Dancing
21	Math	singing
34	Chemistry	Dancing
74	Biology	Cricket
59	Physics	Hockey.

The given STUDENT table is in 3NF, but the COURSE and HOBBY are two independent entity. Hence there is no relationship between COURSE and HOBBY.

In the STUDENT relation, a student with STU_ID, 21 contains two courses, Computer and Math and two

Page No.
Date : / /

hobbies, Dancing and singing. So, there is a multi-valued dependency on STU_ID, which leads to unnecessary repetition of data.

So to make above table in 4NF, we can decompose it into two tables:

STUDENT_COURSE.

STU_ID	COURSE
21	Computer
21	Math
34	Chemistry
74	Biology
59	Physics.

STUDENT_HOBBY.

STU_ID	HOBBY
21	Dancing
21	singing
34	Dancing
74	Cricket
59	Hockey.

b) Describe the GRANT functions and explain how it relates to security. What types of privileges may be granted? How rights could be revoked?

AN. From the GRANT function, we can grant or provide the specific access of feature to the created user. A database user will only be able to perform the operations which are only provided to them. And, as the person (user) feature depend upon the GRANT function, so directly the concern of security comes in.

From GRANT we can assign all the priveliges to the user or can only provide some specific privliges such as SELECT, UPDATE, DELETE etc.

To Grant, we can make use of following syntax:

GRANT <privilege list>
ON <relation name or view name>
TO <user/role list>

Page No.
Date : / /

Ex :

```
GRANT SELECT
ON employee
TO ram, hari;
```

If we want to remove some features or privileges from the user, we can make use of REVOKE command.

Syntax:

```
REVOKE  <privelege_list>
ON      <relation or view>
FROM    <user or role list>
```

5.a) Define query optimization. What are the basic steps of query processing? Explain.

-) Already done in previous set.

b) In terms of file organization, define Indexing, Elevator Algorithm, log disk. How does a mechanical hard disk work?

Ans.

Page No.
Date: / /

2013. Spring.

2. a).

$$\Pi_{employee.emp\_name,\ city}\left(\sigma_{company\_name = \text{"Nabil Bank"}}(Employee \bowtie Works\ In)\right)$$

b). $$\Pi_{employee.emp\_name,\ street,\ city}\left(\sigma_{company\_name = \text{"Nabil"} \wedge salary > 240000}(Employee \bowtie Works\ In)\right)$$

c). $$employee \leftarrow \Pi_{emp\_name,\ street,\ city = \text{"kathmandu"}}\left(\sigma_{emp\_name = \text{"Kiran"}}(Employee)\right) \cup \sigma_{emp\_name \neq \text{'Kiran'}}(Employee)$$

d) $$Works\ In \leftarrow \Pi_{emp\_name,\ company\_name,\ salary = salary \times 1.1}\left(\sigma_{CompanyName = \text{'Nabil Bank'}}(Works\ In)\right) \cup \sigma_{CompanyName \neq \text{'NabilBank'}}(Works\ In).$$

d) यो question मा select छ र ...

$$Works\ In \leftarrow \Pi_{emp\_name,\ company\_name,\ salary = salary \times 1.1}(Works\ In).$$

2016.

5. a) let N = 4.