Unit-2

DBMS.

Sequential files, index sequential files, direct files, Hashing, B-Trees Index files, Relational models, Relational algebra and various operations, Relational and Tuples calculus.

X X

- A database consists of a huge amount of data. The data is grouped within a table in RDBMS, and each tables has related records. A user can see that the data is stored in form of data table, but in actual this huge amount of data is stored in physical memory in form of files.

File:

A file is a named collection of related information that is recorded on secondary storage such as magnetic disks, optical disks and magnetic tapes.

File organisation:

- File organisation refers to the representation or arrangement of data on external storage.
- or it refers to the logical relationships among various records that constitute the file. In simple terms, Storing the files in certain order is called as file organisation.
- The methods by which records can be retrieved and updated from the files are called as access method.

↓ Types of file organisation

Sequential file organisation:-

• The easiest method for file organisation is sequential file organisation method. In this method, the files are stored one after another in a sequential manner.

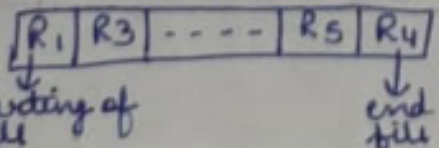
• A sequential file contains records organised by the order in which they were entered. The order of records are fixed. It can only be read or written only sequentially.
(it occupies less space as records are stored in continuous order).

• After you place a record into a sequential file; you cannot shorten, lengthen, or delete the record. However, you can update (REWRITE) a record if the length doesn't change. New records are added at the end of the files.

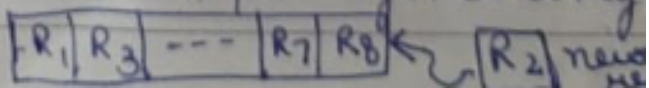
• if the order in which you keep records in a file is not important then sequential file organisation is a good choice whether there are many records or only a few. Sequential output is also useful for printing reports.

→ There are two ways to implement this methods are as follows:-

Pile File Method
t is quite simple and here, we
ore the records in sequence i.e
after other in order in which
y are inserted into the tables.



Sorted file Method
whenever a new records has to be inserted, it is always inserted in a sorted (ascendi or descending) manner. Sor is based on primary or other key



advantages:

- fast and efficient methods for huge amount of data.
- simple design
- files are easily stored in magnetic tapes i.e. cheaper storage mechanism.

disadvantages:

- Time wastage as we cannot jump on a particular record that is required, but we have to move in a sequential manner which takes our time.
- Sorting file method is inefficient as it takes time and space for sorting records.

- Index sequential files:-

- Index sequential file organisation is an advanced sequential file organisation method.
- in this, files can be accessed sequentially as well as randomly with the help of the record key or (the primary key).
- it is possible to store the records in the middle of the file. It occupies more space.
- it also provides slow access (but it is fast as compared to sequential file organisation) as it takes time to search for the index.
- in this, records are written in sequential

order but can be read only in sequential as well as random order.

- one or more keys can be created for storing and accessing the records.

→ The main drawback of sequential file is that searching operation is not efficient because in sequential organisation, primary key of every record is compared with searching key. To optimize, this operation concept of indexed sequential file is introduced.

- in this, a separate file for storing indexes of every record is maintained along with the master file.

eg:

IP	Position
10	5
20	1
30	7
40	3

Position	IP	name	Salary
0			
1	20	Alexa	2000
2			
3	40	Alka	4000
4			
5	10	Pam	1000
6			
7	30	ray	3000
8			

- no need to scan the entire memory block of record.

(ISAM is a file management system developed at IBM that allows records to be accessed either sequentially or randomly).

advantages :-

- desired record can be accessed efficiently by using index which is maintained in separate file.

- Variable length records can also be handled using index sequential file.

disadvantages :-

- atleast 2 files need to be maintained :-
  1) master file 2) Index file

- extra amount of memory is required in order to maintain index file.

- Direct files :-

- direct access file is also known as random access organisation.

- in direct access file, all records are stored in direct access storage device (DASD), such as hard disk.

- The records are randomly placed throughout the file. The records does not need to be in sequence because they are updated directly and rewritten back in the same location.

- The file organisation is useful for immediate access to large amount of information.

→ application → OTP, railway reservation system, airline seat reservations.

• it is used in accessing large database. It is also known as hashing.

advantage :-

• It helps in online Transaction processing system (OLTP) like online railway reservation system.

• sorting of the records is not required in this file organisation.

• It accesses desired records immediately and can update several files quickly.

• it has better control over record allocation and it is more suitable for interactive online applications.

disadvantage :-

• direct access file does not provide backup facility.

• expensive hardware and software are required.

• File updating is more difficult when compared to that of sequential method.

• Less efficient use of storage space.

• Data may be accidently erased or over written unless special precautions are taken.

• used for store and retrieve data at a constant time from database.
(also known as mapping technique as it tries to map large value to smaller value by using hashing)

Hashing :-

• Hashing is an efficient technique to directly search the location of desired data on the disk without using index structure.

• works well for large database.

• Data is stored at the data blocks whose address is generated by using hash function. The memory location where these records are stored is called as data block or data bucket.

↓ Hash file organisation

eg :- K mod 10 (let K = 24)

24 mod 10

4 = Hash value

means 4 par 24 store kar do.

① Data bucket -

data buckets are the memory locations where the records are stored. These buckets are also considered as unit of storage.

② Hash function - (it is mathematical fun^n which store the new record, is used to generate data address to)

Hash function is a mapping function that maps all the set of search keys to actual record address. Generally, hash function uses the primary key to generate the hash index - address of data block.)

③ Hash index -

The prefix of an entire hash value is taken as hash index. Every hash index has a depth value to signify how many bits are used for computing a hash function. The bits can address 2n buckets.

• used for store and retrieving data at a constant time from database.
(also known as mapping technique as we tries to map large values to smaller values by using hashing)

Hashing :-

• Hashing is an efficient technique to directly search the location of desired data on the disk without using index structure.
• works well for large database.
• Data is stored at the data blocks whose address is generated by using hash function. The memory location where these records are stored is called as data block or data bucket.

↓ Hash file organisation

eg :- K mod 10 (K = 24)
24 mod 10
4 = Hash value
means 4 part 24 store karo.

① Data bucket -

data buckets are the memory locations where the records are stored. These buckets are also considered as unit of storage.

② Hash function - (it is mathematical fun^{n} which store the new record, is used to generate data address to)

Hash function is a mapping function that maps all the set of search keys to actual record address. Generally, hash function uses the primary key to generate the hash index - address of data block.)

③ Hash index -

The prefix of an entire hash value is taken as hash index. Every hash index has a depth value to signify (how many bits are used for computing a hash function.) The bits can address 2n buckets.

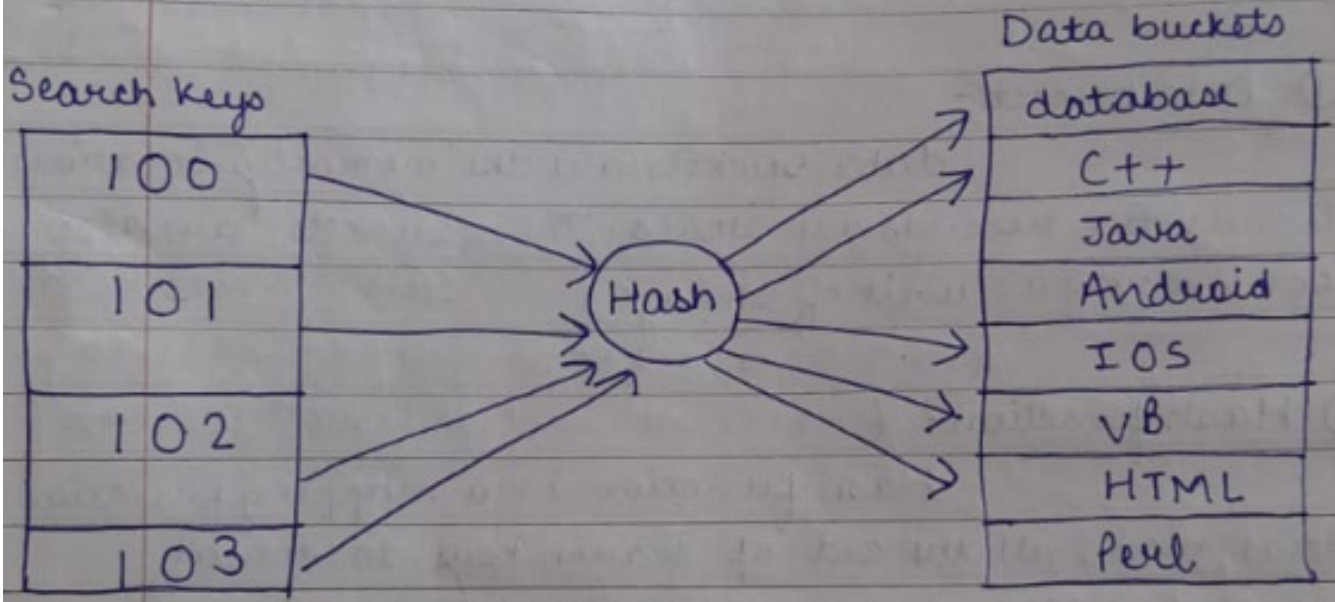
There are 2 types of Hashing :-

1) Static hashing →

• in static hashing, when a search key value is provided, the hash function always computes the same address.

• for eg :-

If mod-4 hash function is used, then it shall generate only 5 value. The o/p addresses shall always be same for that function. The number of buckets provided remains unchanged at all times.

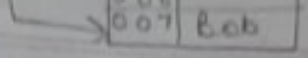


operations

ⓐ insertion: When a record is required to be entered using static hash, the hash function 'h' computes the bucket address for search key 'k', where the record will be stored.

Bucket address = h(k).

ⓑ delete: This is simply a search followed by a deletion operation.



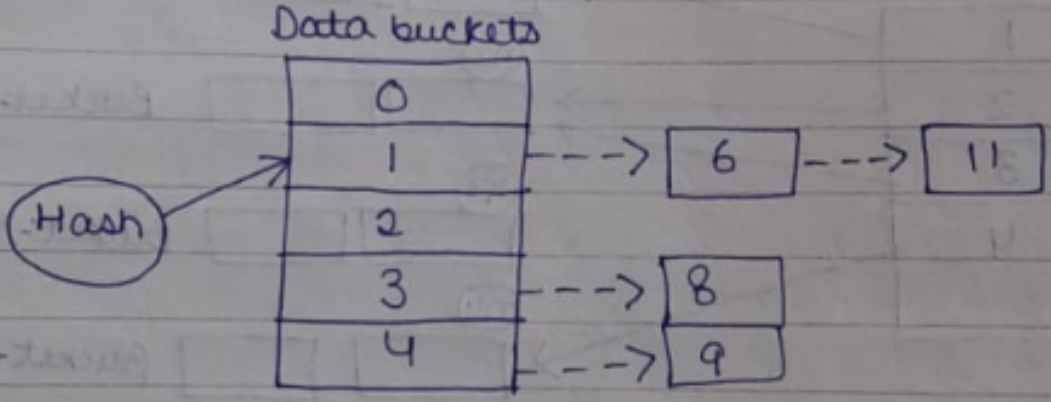
ⓒ **Search :** When a record needs to be retrieved, the same hash function can be used to retrieve the address of the bucket where the data is stored.

↓ Bucket overflow

The condition of bucket overflow is called as collision. This is a fatal state for any static hash function. In this case, overflow chainning can be used.

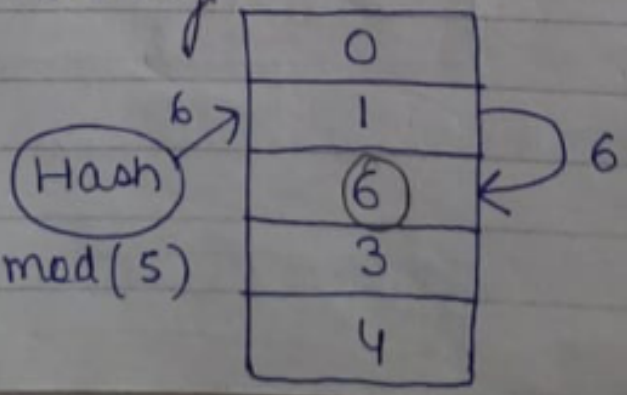
ⓐ **Overflow chaining :**

When buckets are full, a new bucket is allocated for the same hash result and is linked after the previous one. This mechanism is called closed hashing.



ⓑ **Linear Probing :**

When a hash function generates an address at which data is already stored, the next free bucket is allocated to it. This mechanism is called open hashing.

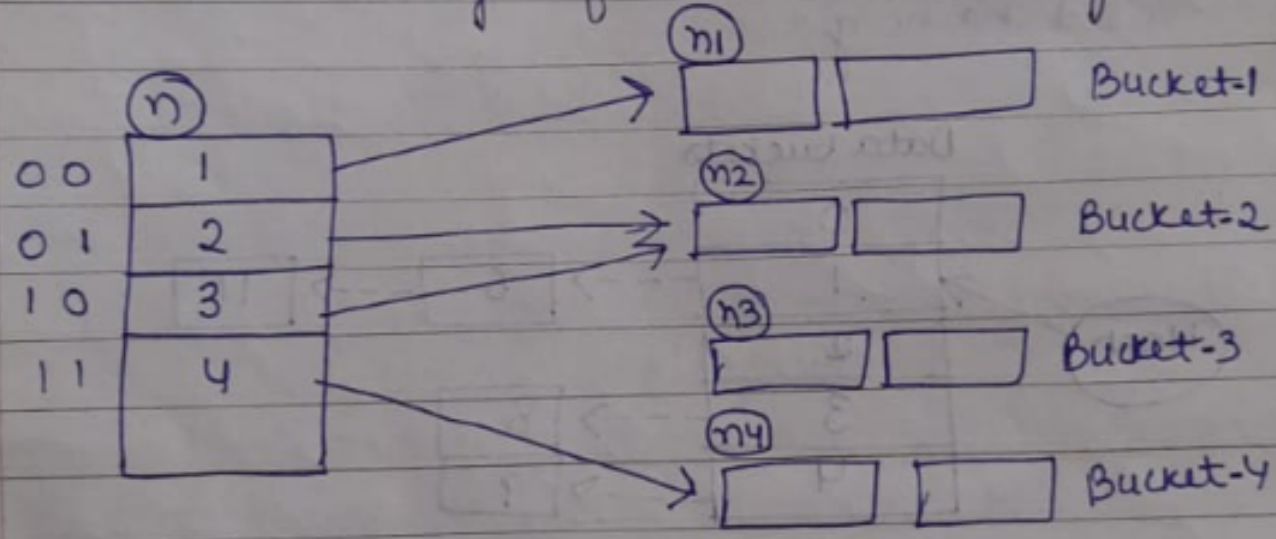


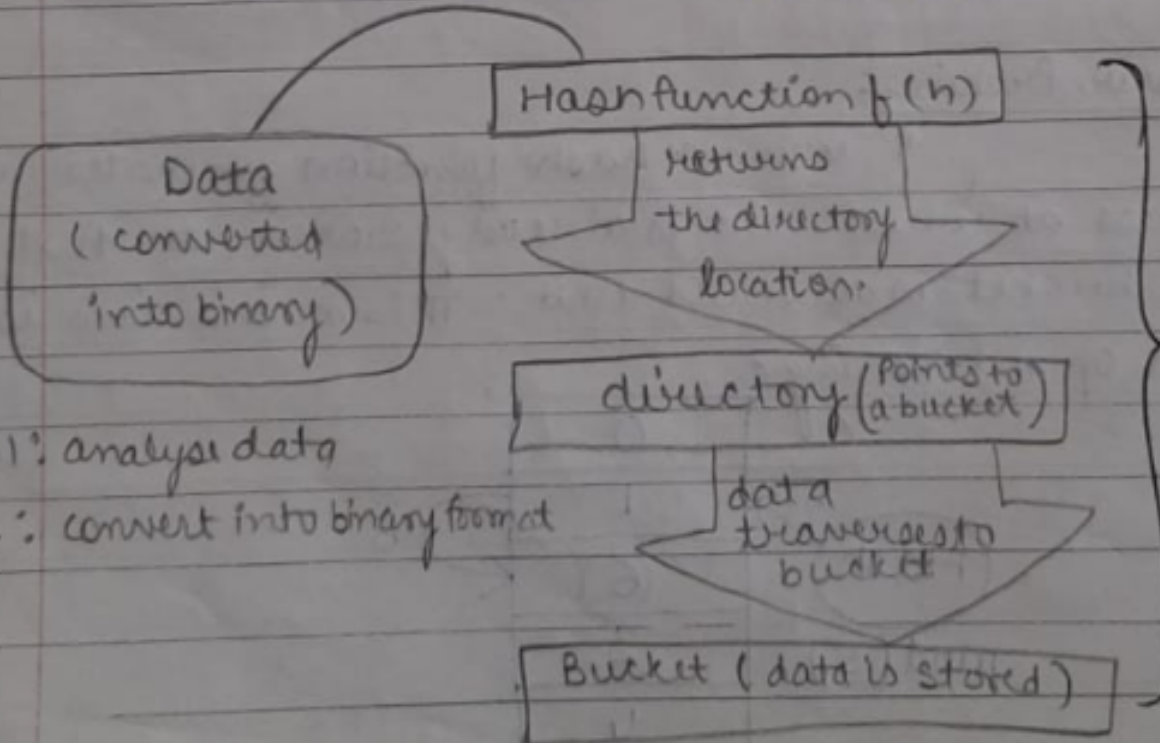
2) Dynamic hashing :→

• The problem with static hashing is that it does not expand or shrink dynamically as the size of the database grows or shrinks.

• Dynamic Hashing provides a mechanism in which data buckets are added or removed dynamically and on-demand. Dynamic hashing is also known as extended hashing.

• Hash function (in dynamic hashing), is made to produce a large number of values and only a few are used initially.





Step 1: analyse data

Step 2: convert into binary format

→ So, extended hashing is a dynamic hashing method where in directories and buckets are used to hash data. It is an aggressively flexible method in which the hash function also experiences dynamic change.

- **Directories:**

The directories store addresses of the bucket in pointer and it is assigned to each directory which may change each time when directory expansion takes place.

- **Bucket:**

The buckets are used to hash the actual data.

for eg → do extendable hashing for given data :-

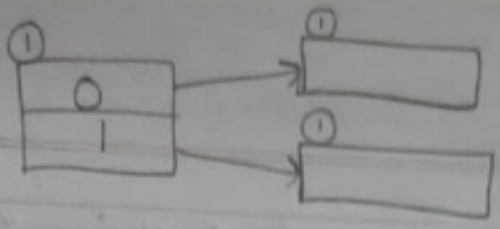
16, 4, 6, 22, 24, 10, 31, 7, 9, 20, 26.

↓ convert into binary

16 - 10000	7 - 00111
4 - 00100	9 - 01001
6 - 00110	20 - 10100
22 - 10110	26 - 11010
24 - 11000	
10 - 01010	
31 - 11111	

now assume the buckets :-

Initially

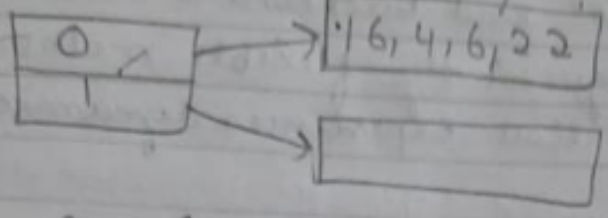


CLASS/YEAR	PAGE NO.
DATE / /	

16 = 10000
4 = 00100
6 = 00110
22 = 10110

insert 16, 4, 6, 22

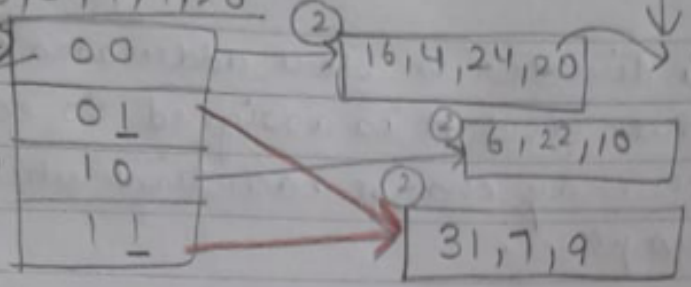
as only 3 so, it overflow!



only split 0 as it is overflow

now.

insert 24, 10, 31, 7, 9, 20



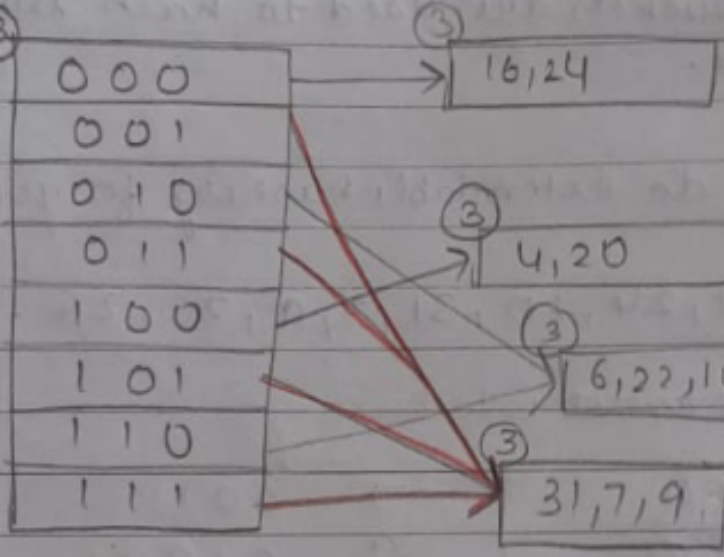
only split 00 as it is overflow

overflow!

24 = 11000
10 = 01010
31 = 11111
7 = 00111
9 = 01001
20 = 10100

now.

insert 26

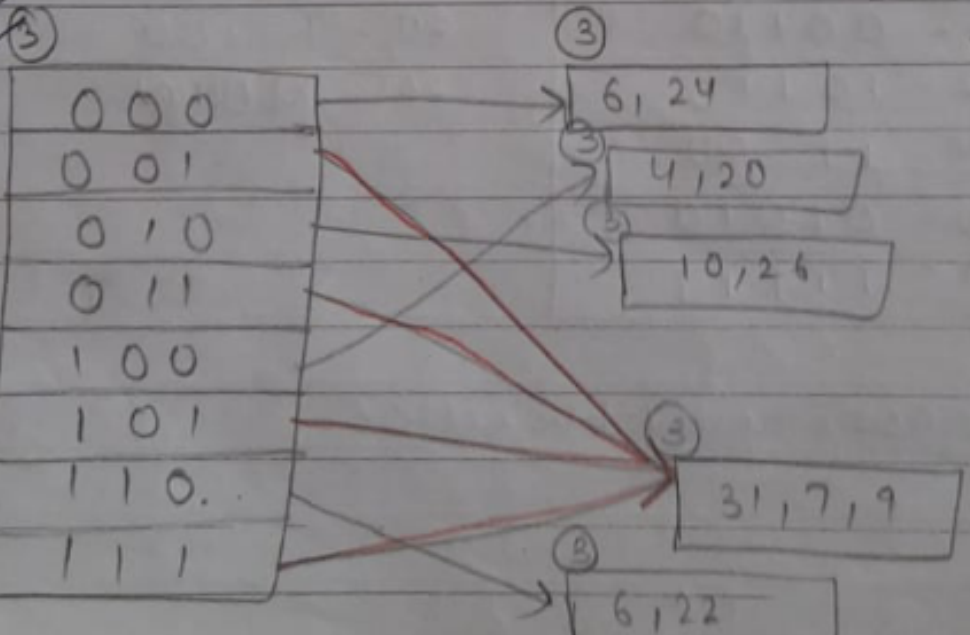


26 = 11010

overflow!

now.

extended hashing!



6 = 00110
22 = 10110
10 = 01010
26 = 11010

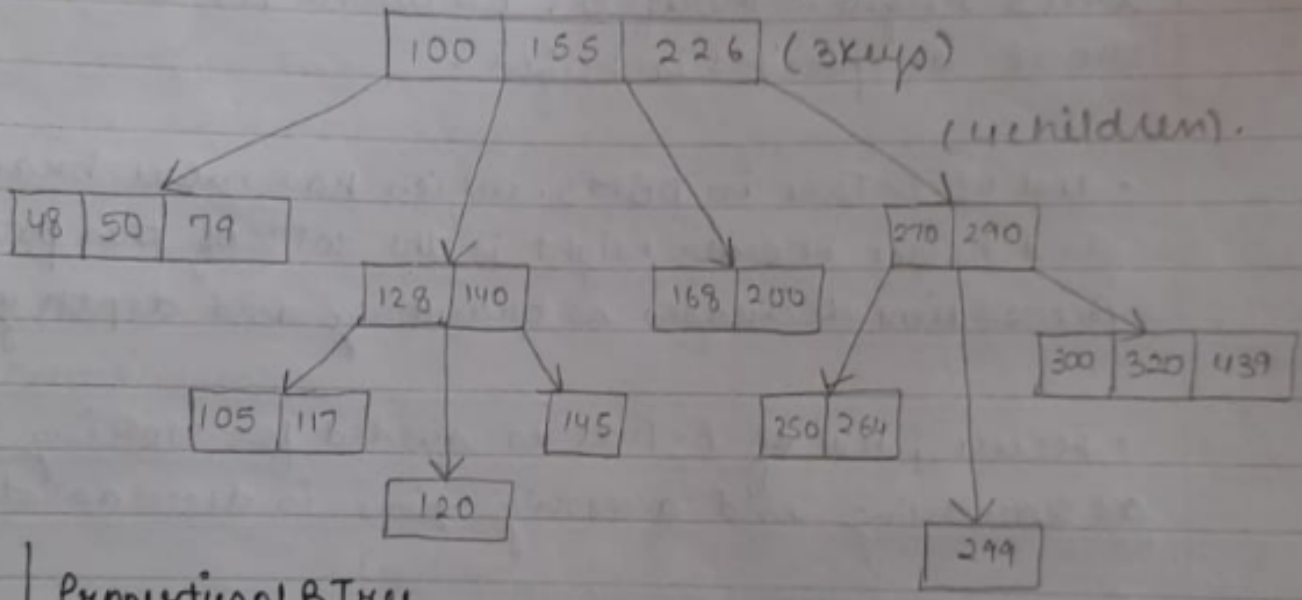
- it has more than one key & more than 2 children.
- maintained sorted data.
- all leaf nodes must be at same level.
- every node has max. (m-1) key.

- **BTree index files:-**

- B Tree in DBMS is a m-way (order) tree which self balances itself. due to their balanced structure, such trees are frequently used to manage and organize enormous database and facilitate searches.

- in B Tree, each node can have a maximum of n child nodes. eg → multilevel indexing.

- Leaf nodes and indexing nodes will have both record references. BTree is called as Balanced stored trees as all the leaf nodes are at same levels.



↓ Properties of BTree

Following are some of the properties of B-Tree are as follows:-

- A non-leaf node's number of keys is one less than the number of its children.

- The number of keys in the root ranges from one to (m-1)

maximum. Therefore, root has a minimum of two and a max. of m children.

- The keys range from min $\left[\frac{m}{2} - 1\right]$ to max. $(m-1)$ for all nodes (non-leaf nodes) besides the root. Thus, they can have between (m) and $\left(\frac{m}{2}\right)$ children.

- The level of each leaf node is the same.

↓ need of BTree

- For having optimized searching we cannot increase a tree's height. Therefore, we want the tree to be as short as possible in height.

- use of B-Tree in DBMS, which has more branches and hence shorter height is the soln of this problem. Access time decreases as branching and depth grow.

- Hence, use of B-Tree is needed for storing data as searching and accessing time is decreased.

- The cost of accessing the disc is high when searching tables. Therefore, minimising disc access is our goal.

- So to decrease time and cost, we use B-Tree for storing data as it makes the index fast.

How database BTree indexing works?

- When BTree is used for database indexing, it becomes a little more complex because it has both a key and a value. The value serves as a reference to a particular data record. A payload is the collective term for the key and a value.

- for index data to particular key and value, the database first constructs a unique random index or a primary key for each of the supplied records. The keys and record byte streams are then all stored on a B+ tree. The random index that is generated is used for indexing of the data.

- So this indexing helps to decrease the searching time of data. In a B-Tree, all the data is stored on the leaf nodes, now for accessing a particular data index, database can make use of binary search on the leaf nodes as the data is stored in the sorted order.

- If indexing is not used, the database reads each and every records to locate the requested record and it increases time and cost for searching the records, So BTree indexing is very efficient.

How searching happens in indexing database?

- The database does a search in the B-Tree for a given key and returns the index in O (log (n)) time. The record is then obtained by running a second B+ tree search in

O(log(n)) time using the discovered index. So, overall approx time taken for searching a record in a BTree in DBMS indexed database is O(log(n)).

→ as in the previous example:-

the data is stored in sorted order according to the values, if we want to search for the node containing the value 48, so the following steps will be applied.

- First, the parent node with key having data 100 is checked, as 48 is less than 100 so the left children node of 100 is checked.

- in left children, there are 3keys, so it will check from the leftmost key as the data is stored in sorted order.

- Leftmost element is having key value as 48 which match the element to be searched, so that ~~how we~~ the element we wanted to search is got.

↓ Conclusion:-

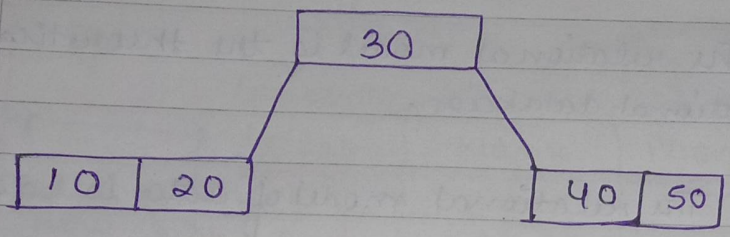
- A m-way tree that self-balances itself is called a "B-Tree".

- Due to their balanced structure, such trees are frequently used to manage and organize enormous

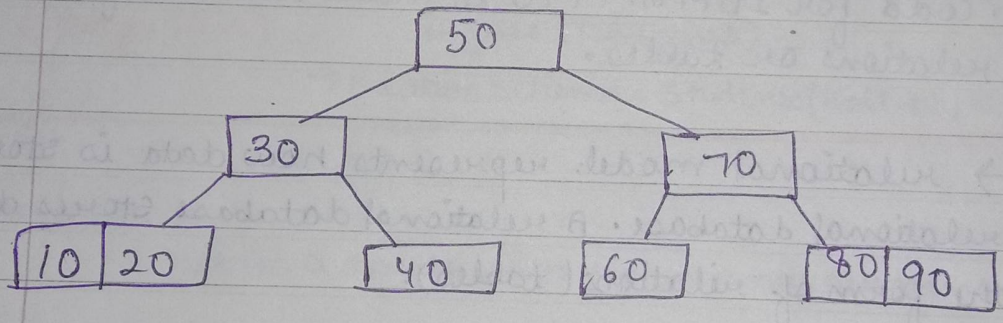
database and facilitate searches.

- B Tree is an example of multilevel indexing.
- Uses of B Tree is needed for storing data as searching and accessing time is decreased.
- B Tree can be used for database indexing to improve searching and storing time of database.

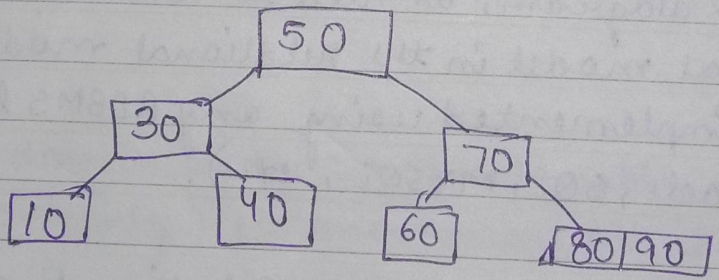
→ for eg: Construct B-Tree for search key values (10,20,30,40, 50) when n=3.



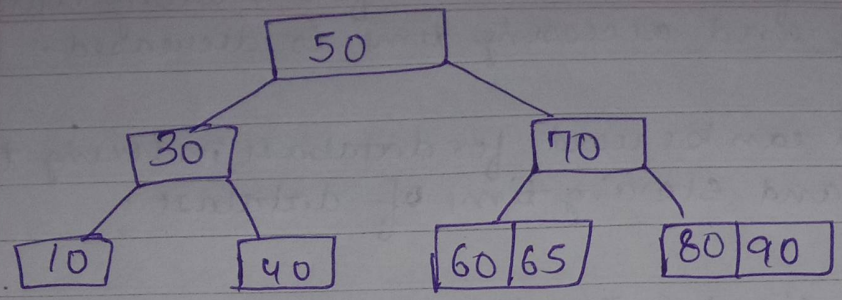
eg: values are 10,20,30,40,50,60,70,80,90.



a) Delete 20 from above tree :-



b) Insert 65 to above tree :-



- **Relational Model :-**

• The relational model is the theoretical basis of relational database.

• The relational model of data is based on the concept of relations.

• "Relational Model" is proposed by E.F. Codd for IBM in 1970 to model data in form of relations or tables.

→ relational model represents how data is stored in relational database. A relational database stores data in the form of relations (tables).

↓

• after designing the conceptual model of database using ER diagram, we need to convert the conceptual model in the relational model which can be implemented using any RDBMS languages (like Oracle, SQL, MySQL, etc.).

• RDBMS stands for Relational database Management system. It is the basis for SQL, and for all model database systems like MS SQL server, IBM DB2, Oracle etc.

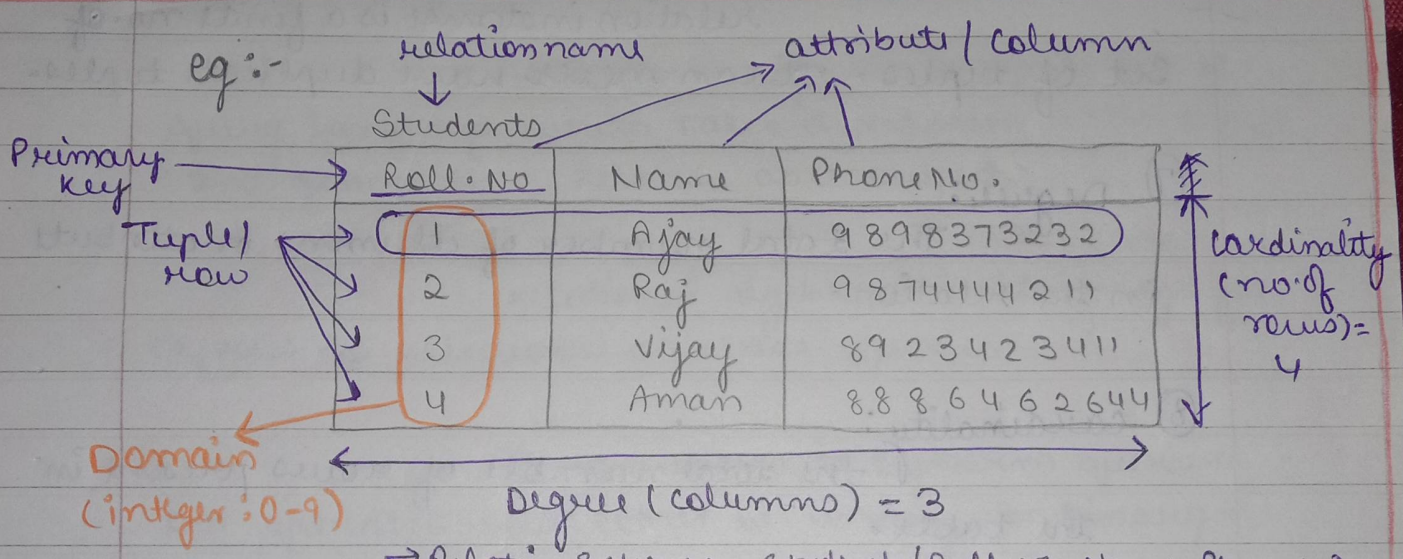
2023.05.22 14:18

CLASSTIME | Page No.
Date / /

• So, RDBMS is a database management system (DBMS) that is based on relational model as introduced by E.F. Codd.

→ Relational model can be represented as table with columns (attribute or name) and rows (tuple).

eg:-



→ Relation Schema = Students(Roll.NO, Name, Phone No.).

Terms

① relation:
it is a table with rows and columns.

② attribute:
an attribute is a named column of a relation.

③ Domain:
A domain is the set of allowable values (eg: roll.NO can be 0-9 only) for one or more attributes.

④ Tuples:
A tuple is a row of a relation.

2023.05.22 14:18

CLASSTIME | Page No.
Date / /

⑤ relation Schema:
A relation schema represents the name of relation with its attribute (for eg: in this table, name of relation is student and attributes are roll.NO., phone.no., name).

⑥ relation instance:
relation instance is a finite no. of set of tuples. It can never have duplicate tuples.

⑦ Degree:
The total number of columns or attribute in the relation.

⑧ cardinality:
the total numble of rows present in the table.

⑨ Relation Key:
Every ~~att~~ row has one or multiple attributes, that can uniquely identify the row in relation (like roll.NO.) which is called as relation key (primary key).

⑩ Tuple Variable:
It is the data stored in a record of the table.

- each relation has unique name; each row is unique; entries in column has the same domain; order of column or rows is irrelevant; each cell of relation contains exactly one values.

2023.05.22 14:18

CLASSTIME | Page No.
Date / /

Formal term	alternative 1	alternative 2
Relation	Table	File
Tuple	row	Record
attribute	column	Field

- **Relational algebra and various operations:-**

• relational algebra is a procedural query language which takes a relation as an input and generates a relation as a output.

• relational algebra is a language for expressing relational database queries.

• It uses operator to perform queries. An operator used either be unary or binary.

→ types of operator are :- ① Basic / fundamental operator, ② Additional / derived operator.

• Relational algebra operations work on one or more relations to define another relation without changing the original relations.

• Relational algebra operations are performed recursively on a relation.

→ for eg :- I/P is a relation (table from which data has to be accessed) and O/P is also a relation (a temporary table holding the data asked for by the user).

2023.05.22 14:18

(2) We can use relational algebra to fetch data from this table (relation).

I/P →

ID	name	Age
1	AKon	17
2	BKon	19
3	CKon	15
4	DKon	13

Select name of students with age less than 17.

O/P

Name
CKon
DKon

(3)

(4) Various operations

① Basic / Fundamental operations
- Selection (σ)
- Projection (π)
- Union (∪)
- Set difference (−)
- Cartesian product (X)
- Rename (ρ)

② additional / Derived operations
- Natural Join (⋈)
- Set intersection (∩)
- Division (÷)
- Assignment (←)
- Left, right, Full order Join (⟕, ⟖, ⟗)

① Selection :- (σ) it selects all tuples (rows) that satisfy the selection condition from relation 'R'.

$\sigma_{<condition>}(R)$

eg :- $\sigma_{(age\ more\ than\ 17)}(R)$

② **Projection (Π)** :- Produce new relation with some of the attributes.

$$\Pi_{<attributes>}(R)$$

③ **union (∪)** :- produce a relation that includes all the tuples from R_1 or R_2

$$(R_1 \cup R_2)$$

④ **Set of difference (−)** :- Produces a relation that includes all tuples in R_1 that are not in R_2.

$$R_1 - R_2$$

⑤ **Cartesian product (X)** :- also known as (cross product) Produce A relation that has attributes of R_1 and R_2, and includes as tuples, all possible combination of tuples from R_1,

$$R_1 \times R_2$$

⑥ **Rename (ρ)** :- used to give a name to a relation obtained after applying any relational algebra operation

$$\rho(R, E)$$

2023.05.22 14:19

⑦ **Natural Join :-**
(⋈) Join two relation having any no. of attributes.

$$R_1 \bowtie R_2$$

⑧ **Set intersection :-**
(∩) produces a relation that includes all the tuples in both R_1 & R_2.

$$R_1 \cap R_2$$

⑨ **Division :-**
(÷) queries that includes 'FOR ALL', 'EVERY'.

$$R_1 \div R_2$$

→ for eg :-

Student

Roll NO.	Name	Age	Address
1	A	20	Bhopal
2	B	17	Mumbai
3	C	16	Mumbai
4	D	19	Delhi
5	E	18	Delhi

Selection

① Select student whose name is D

$\sigma_{name = "D"}$ (Student)

Roll. NO.	Name	Age	Address
4	D	19	Delhi

② Select students whose age is greater than 17

$\sigma_{age > 17}$ (Student)

Roll. NO.	Name	Age	Address
1	A	20	Bhopal
4	D	19	Delhi
5	E	18	Delhi

2023.05.22 14:19

CLASSTIME | Page No.
Date / /

Projection ① Display (or project) name of students in Student table.

Name
A
B
C
D
E

$\Pi_{<name>}(Student)$

② display address of students in student table.

address
Bhopal
Mumbai
Delhi

Note:

→ by default, projection removes duplicate values.

Imp

eg :-

Student

Roll_NO.	Name
1	A
2	B
3	C
4	D

employee

emp._no.	Name
2	B
8	G
9	H

union ① $\Pi_{<name>}(Student) \cup \Pi_{<name>}(employee)$

② $\Pi_{<Student>} \cup \Pi_{<employee>}$

Roll_NO	Name
1	A
2	B
3	C
4	D
8	G
9	H

Name
A
B
C
D
G
H

2023.05.22 14:19

CLASSTIME | Page No.
Date / /

Set of difference

① (Student) - (employee)

Roll-no	name
1	A
3	C
4	D

② $\Pi_{Name}(Student) - \Pi_{Name}(employee)$

Name
A
C
D

Cartesian product

① $R_1 \times R_2$

eg:

R_1

A	B
α	1
β	2

2

R_2

C	D	E
α	10	a
β	10	a
β	20	b
γ	10	b

4

A	B	C	D	E
α	1	α	10	a
α	1	β	10	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
β	2	β	10	a
β	2	β	20	b
β	2	γ	10	b

$2 \times 4 = 8$

② $\sigma_{A=C}(R_1 \times R_2)$

A	B	C	D	E
α	1	α	10	a
β	2	β	10	a
β	2	β	20	b

Rename

① query to rename attributes Name, Age of table Person to N, A.

→ $\rho_{(N,A)}(Person)$

2023.05.22 14:19

② query to rename the table name Project to work and its attributes to P, Q, R.

$\rho_{work(P,Q,R)}$ (Project)

rename table name. — rename attributes

③ query to rename table name Loan to L.

ρ_{L} (Loan)

Set intersection

① (Student) ∩ (employee)

Roll-NO.	Name
2	B

② π_{name} (student) ∩ π_{name} (employee)

Name
B

Natural Join

① Course ⋈ HOD

courses

CID	Course	Dept
CS01	Database	CS
ME01	Mechanics	ME
EE01	electronics	EE

HOD

dept	Head
CS	A
ME	B
EE	C

CID	Course	Dept	Head
CS01	Database	CS	A
ME01	Mechanics	ME	B
EE01	electronics	EE	C

2023.05.22 14:19

CLASSTIME | Page No.
Date / /

division (1)

A

x	y
a	1
b	2
a	2
d	4

B

y
1
2

A ÷ B

x
a

- **Relational and Tuple calculus:-**

→ **Relational calculus** → • it is a non-procedural query language (or declarative language). It uses mathematical predicate calculus (or first order logic) instead of algebra.

• It tells what to do but never explains how to do.

• It provides description about the query to get the result whereas a relational algebra gives the method to get results.

→ when applied in database, it comes in two flavors:-

(1) **Tuples relational Calculus**
- proposed by Codd in 1972
- works on tuples (or rows).
like SQL (structured query language).

(2) **Domain Relational Calculus**
- proposed by Lacroix and Pirotte in 1977
- works on domain of attributes (or columns).
like QBE (query by example)

2023.05.22 14:19

Date : / /
Page No.

- **Tuple relational calculus:-**

• A tuple relational calculus is a non-procedural query language that specifies to select the tuples in a 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 more tuples.

• TRC is variables range over tuples like SQL.

→ This query cannot be expressed using membership cond^n.

• it used for selecting the tuples in a relation that satisfies the given cond^n (or predicate).

notation :- $\{T \mid P(T)\}$ or $\{T \mid \text{condition}(T)\}$

↓

where T is resulting tuples and P(T) is a cond^n used to fetch T.

for eg :- $\{T \mid \text{employee}(T) \text{ and } T.\text{Dept\_ID} = 10\}$

This selects all the tuples of employee names whose work for department 10.

- **Domain Relational calculus :-**

• A domain relational calculus uses the list of attributes to be selected from the relation based on the condition.

2023.05.22 14:20

Date : / /
Page No.

- it is the same as TRC but differs by selecting the attributes rather than selecting whole tuples.

- DRC is a varible range over domain elements like query-by-example (QBE).

notation :-

$$\{a1, a2, \ldots, an \mid P(a_1, a_2, \ldots, a_n)\}$$

↓

where $a_1, a_2, \ldots, a_n$ are attributes of the relation and P is the cond^n.

for eg :-

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

Select EMP_ID and EMP_NAME of employees who work for department 10.

- The query can be expressed using a membership condition.

——— X ——— X ———