Assignment NO.3

Q.1) What is functional dependency and its type.

→ Functional dependency is a constraint that determines the relation of one attribute in a DBMS.

Functional dependency helps to maintain the quality of data in the db.

It plays a imp role is find the difference between good and bad db design.

A functional dependency is denoted by an arrow "→". The functional dependency of X on Y is represented by X → Y.

Let's understand functional dependency in DBMS with example.

Emp-no	Emp-name	salary	city
1	sanjivani	50,000	Pandharpur

In this ex, if we know the value of empno ~~no~~, ~~e~~ we can obtain emp-name, city, salary etc by this we can say that the city, emp-name & salary are functionally depended on emp-no.

Types of FD in DBMS -

i) Multivalued Dependency -

Multivalued dependency occurs in

in the situation where there are multiple independent multivalued attributes in a single table.

consider the following multivalued example -

car-model	maf-year	color
H001	2018	red
H001	2017	Green
H005	2018	Blue
H005	2018	black

In this ex, maf-year & color are independent of each other but dependent on car-model.

In this example, these two columns are said to be multivalue dependent on car-model.

This dependence can be represented like this

car-model → maf-year

car-model ⟶ color

iii) Trivial Function Dependency -

The trivial dependency is a set of attributes which are called a trivial if the set of attributes are included in that attribute.

So, $X \rightarrow Y$ is a trival functional dependency, if Y is a subset of X. Let's

understand with a trivial FD example,

emp-ID	Emp-name
A5555	Harry
A5811	George
A5999	kevin

consider this table of with two columns emp-ID & emp-name.

{Emp-ID, Emp-name} ⟶ Emp-id is a trivial functional dependency as Emp-id is a subset of {emp-id, Emp-name}.

iii) Non-trivial FD:

Functional dependency which also known as non-trivial dependency occurs when A ⟶ B holds true where B is not a subset of A. In a relationship if attribute B is not a subset of attribute then it is considered as a non-trivial dependency.

ex

Company	CEO	Age
Microsoft	Satya Nadella	51
Google	Sundar pichai	46
Apple	Tim cook	57

{company} → {CEO}

But CEO is not a subset of Company & hence it's non-trivial FD.

iv) Transitive Dependency -

A transitive dependency is a type of FD which happens when "t" is indirectly formed by two ~~function~~ FD. Let's understand it by example

Company	CEO	Age
Microsiw	satya	51
Google	sundar	46

{company} → {CEO}

{CEO} → {Age} (if we CEO, we know the age)

∴ According to transitivity rule {company} → {Age} should hold that makes sense because if we know the company name, we can knoco his age.

Q.2) Explain Normalization and its form.

→ Normalization is the process to eliminate data redundancy and enhance data integrity in the table.

Normalization also helps to organize the data in the db, it is a multi-step process that sets the data into tabular form and removes the duplicated data from the relational tables.

1st Normal Form (1NF)

This is the most basic level of normalization. In 1NF, each table cell should contain only a single value, and each column should have a unique name. The first normal form helps to eliminate duplicate data and simplify queries.

2nd Normal Form (2NF)

2nd NF eliminates redundant data by requiring that each non-key attribute the dependent on the primary key. This means that each column should be directly related to the primary key and not to other columns.

3rd Normal form (3NF) -

3NF builds on 2NF by requiring that all non-key attributes are independent of each other. This means that each columns should be directly related

to the primary key & not to any other columns in the same table.

Boyce-codd Normal form (BCNF) -
BCNF is a stricter form of 3NF that ensures that each determiniant in a table is a candidate key. In other words, BCNF ensures that each non-key attribute is dependent only on the candidate key.

4th Normal Form (4NF)
4th NF is a further ~~refinement~~ redefinement of BCNF that ensures that a table does not contain any multi-valued dependencies.

5th Normal form (5NF)
5NF is the highest level of normalization and involves decomposing a table into smaller tables to remove data redundancy and improve data integrity.

Q.3) What is file Organisation and explain Indexing in DBMS.

→ The file is a collection of records. Using the primary key, we can access the records. The type and frequency of access can be determined by the type of file organization which was used for a given set of records.

File organisation is a logical relationship among various records. This method defines how the file records are mapped onto disk blocks.

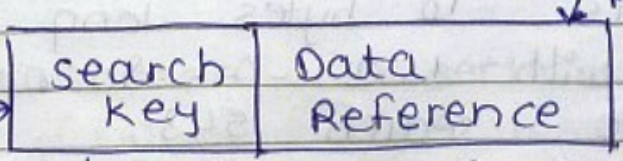
Files of fixed length records are easier to implement that the files of variable length records.

Types of file organization
- sequential FO
- Heap FO
- Hash FO
- ISAM
- B+ tree FO
- Cluste FO

ordered Indices -

Indexing is used to optimize the performance of db by minimizing the number of disk access ~~&~~ required when a query is proposed.

The index is a type of data structure. It is used to locate and access the data in a db quickly.

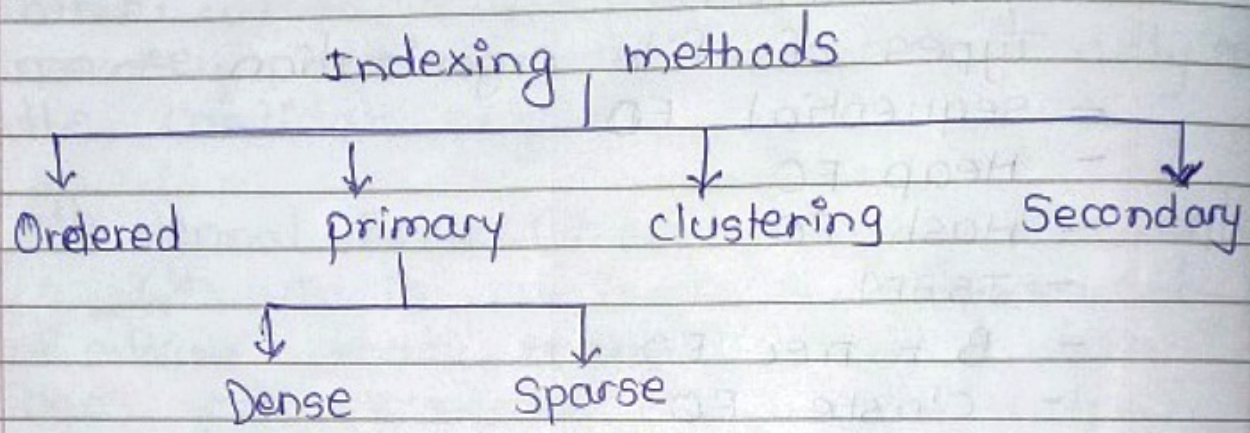


The 1st column of the db is search key that contains copy of primary key or candidate key of table.

The values of primary key are stored in Stored order so that correspo-

nding data can be ~~accor~~ accrossed easily.

The 2nd column of the ~~bt~~ db is the data reference. It contains set of pointers holding the address of the particular key can be found.



Ordered Indices

The indices are usually stored to make searching faster. The ~~d~~ indices which are sorted are known as ordered indices.

Suppose ~~we~~ we have an employee table with thousands of records and each of which is 10 bytes long. If their ID's starts with 1, 2, - - - so on and we have to search 543.

In this case of db with no index, we have to search disk block from starting till it reaches 543.

The DBMS will read the record after reading 543 * 10 = 5430 bytes

In the case of an index, we will search using indexes and dbms will read record after reading 542*2 = 1084 byte which are very less compared to previous case.

Q.4) Differentiate betⁿ B tree index files & B + tree index files.

→

B Tree	B+ Tree
① All the internal & leaf nodes have data pointers.	Only leaf node have data pointers.
② Since, all keys are not available at leaf, search often take more time.	All keys are at leaf nodes, hence search is faster & more accurate.
③ No duplicate of key maintained in tree.	Duplicate of keys are maintained and all nodes are present at leaf.
④ Insertion takes more time & it may not predicateble sometime.	Insertion is easier & the results are always same.
⑤ Deletion of internal node is very complex and tree has to undergo	Deletion of any node is easy and all nodes are found at leaf

⑨ All leaf nodes are not same height.

⑩ B tree occupy less space than B+ tree

⑨ All leaf nodes are same height.

⑩ B+ tree occupy more space than B tree

in any transformation.	
⑥ leaf nodes are not stored as structural linked list.	Leaf node are stored as structural linked list.
⑦ Sequential access to nodes is not possible	Sequential access is possible just like linked list.
⑧ For particular number of nodes height is larger.	Heigh is lesser than B tree for the same number of nodes.

Q.5) Write down Inference Rules (Armstrong Axioms) and prove it.

→ The Armstrong's axioms are the basic inference rule.

Armstrong's axioms are used to conclude functional dependencies on a relational db.

The inference rule is a type of assertion it can apply to a set of functional dependencies to drive other FD.

Using the inference rule, we can drive additional functional dependencies from the initial set.

The six type of ~~Interf~~ Inference Rule.

i) Reflexive Rule (IR1)

In the reflexive rule, if Y is a subset of X, then X determines Y.
if $X \supseteq Y$ then $X \longrightarrow Y$.

ii) Augmentation Rule (IR2)

The augmentation is also called as a partial dependency. In augmentatn if X determines Y; then XZ determines YZ, for any Z.
if $X \longrightarrow Y$ then $XZ \longrightarrow YZ$

iii) Transitive rule (IR3)

In the transitive rule, if X determines Y and Y determine Z, then X must determine Z.

$X \longrightarrow Y$
& $Y \longrightarrow Z$
then $X \longrightarrow Z$

iv) Union Rule (IR4)

Union rule says, if X determines Y and X determines Z, then X must also determine Y & Z.

$X \longrightarrow Y$
$X \longrightarrow Z$
then $X \longrightarrow YZ$

Proof : $X \longrightarrow Y$

$X \longrightarrow Z$

$X.X \longrightarrow XY$

$X \longrightarrow XY$

$XY \longrightarrow YZ$

$X \longrightarrow YZ$

$\therefore X \longrightarrow XY \longrightarrow YZ$

$\therefore X \longrightarrow YZ$

v) Decomposition -

$X \longrightarrow YZ$

then

$X \longrightarrow Y$

$X \longrightarrow Z$

Proof : $X \longrightarrow YZ$

$YZ \longrightarrow Y$ & $YZ \longrightarrow Z$

$X \longrightarrow YZ \longrightarrow Y$

$\therefore X \longrightarrow Y$

$X \longrightarrow YZ \longrightarrow Z$

$\therefore X \longrightarrow Z$

vi) Pseudo transitivity

$X \longrightarrow Y$

$YZ \longrightarrow W$

then

$XZ \longrightarrow W$

Proof : $X \longrightarrow Y$

$XZ \longrightarrow YZ \longrightarrow W$

$\therefore XZ \longrightarrow W$

17/11/23