

6/5/21

DBMS

END-SEMESTER EXAM

BM KALPAJEET
19BCS117
CSE

1) • storing employee tuple in a heap file, with clustered index on empname.

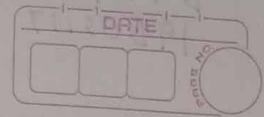
• A clustered index defines the physical order in which the table records are stored in a database.

• If we use empname as a field for clustered indexing, then we must ensure that the names are unique as clustering stores it in alphabetical order.

the operations will be possible if all have unique name.

• with empid as the clustering index if empid is unique and primary key, all the operations are possible.

→



- with empname & empid, it is possible to create custom clustered index and operations are possible.
-

- 2) • DDL is important in defining the database in DBMS, because it is used to describe the schema and constraints.

DDL is used for defining the attributes (columns) of the database, and it is declarative in nature.

It uses CREATE, DROP, ALTER etc for defining the structure of database. It also helps in data independence and managing various integrities.

- DML is used to add, retrieve, delete and update data; it is not important for defining the relations or database or schema.

DML can add or update the rows of table. It affects only one or more rows, and is imperative. It uses commands like INSERT, DELETE, UPDATE.



4) Banking system DBMS that supports transactions and DB operations.

(a) A transaction is a set of instructions, a logical unit of work which interacts with the DB and modifies it. It must be governed by ACID properties.

(a) From a database point of view, the user is a consumer of the banking system and does not need to guarantee anything in particular.

The user, however, must ensure that their user credentials are kept safe, that they don't follow any unethical practices like hacking to modify the database, or cancel their transaction by any other means.

(b) The DBMS must guarantee the ACID properties.

Atomicity: all-or-none.

either the transaction is complete and committed or

the transaction is aborted.

consistency: correctness of values in the database. the DBMS must be consistent before and after transaction.

isolation: multiple transactions can occur simultaneously without inconsistency.

durability: once the transaction is committed, the updates to the database are final and stored to the DBMS.

5) XYZ has created the DBMS with many relations. Let's consider,

Stu ID	Name	Aadhar	DOB	...
18	Virat	9665	05-11-1988	

only one instance is given.

In reality, by analyzing logically, we can say that, ^{Stu ID} you Aadhar can be unique and can be used as primary key.

However, we don't know the composition of the attributes or if they are minimal, and



if it is unique.

Only XYZ will know which is the primary key as XYZ have designed it using constraints.

In easy single valued attributes in DBMS, it can be found. But in general, it cannot be determined.

Student Table

6) qum

S ID	S Name	Email	Age
1000	Jaya	Jaya@xyz.co	20
1005	Krishna	Krishna@pgv.com	22
1030	John	Null	23
1020	John	Jh@xyz.com	22

a) we need to create a clustered index on StudentName and fetch the email column.

for clustered index:

~~CREATE CLUSTERED INDEX Stc.~~

CREATE CLUSTERED INDEX test
ON Student Table (StudentName Asc)

QUERY: SELECT Email
FROM Student Table

b) if WHERE S.Age >= 21 is added.

OUTPUT

Student ID	StudentName	Email	Age
1005	Krishna	Krishna@pgv.com	22
1020	John	Jh@xyz.com	22
1030	John	Null	23

7) Suppliers (sid: int, sname: str, address: str)

Parts (pid: int, pname: str, color: str)

Catalog (sid: int, pid: int, cost: real)

- Relational algebra

we need pid of Parts from 2 diff supplier.

pid is common to catalog and parts.

sid is common to parts, suppliers and catalog.

we need to link suppliers and parts.

$\rho(R_1, \text{catalog})$

$\rho(R_2, \text{catalog})$

→

$$\pi_{R_1 \cdot pid} \sigma_{(R_1 \cdot pid = R_2 \cdot pid) \wedge (R_1 \cdot sid \neq R_2 \cdot sid)}$$

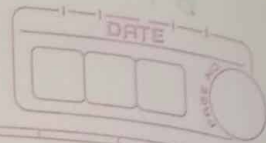
$$\pi_{R_1 \cdot pid} \sigma_{(R_1 \cdot pid = R_2 \cdot pid) \wedge (R_1 \cdot sid \neq R_2 \cdot sid)} (R_1 \times R_2)$$

SQL: ~~SELECT C1.pid~~
~~FROM Catalog C1~~
~~WHERE EXISTS (SELECT C2.sid~~
~~FROM~~

SELECT C1.pid
 FROM Catalog C1
 WHERE EXISTS (SELECT C2.sid
 FROM Catalog C2
 Where
 C2.pid = C1.pid
 AND
 C2.sid \neq C1.sid)

Q8) $\pi_{sname} (\pi_{sid} ((\sigma_{color = 'red'} Parts) \bowtie (\sigma_{cost < 100} Catalog) \bowtie Suppliers))$

→



We need to project the Sname column
of those projections ref sid
when

the parts are red in color
joined with
cost ref catalog is < 100
joined with
suppliers.

On evaluating the relational algebra,
it will give the ~~no~~
supplier names of those suppliers
who supply a red colored
part and it costs less than
100 rupees in price.



9) Emp (eid: int, ename: string, age: int, salary: real)

We need to write a view query on Emp such that it could be automatically updated by updating Emp.

Here, the new view must be updatable.

~~CREATE VIEW~~

```
CREATE OR REPLACE VIEW  
[Emp-test] AS  
  
SELECT *  
FROM Emp.
```

the view can be updated using

```
UPDATE column-name  
SET new-value.
```

by using CREATE OR REPLACE and UPDATE-SET, the views are updated