**Sample questions for Storage and Indexing**

**These are some sample questions. Solving only these problems are not sufficient. You have to go through the book.**

Storage

Q.1: (a) Construct a 2-level (first level (inner level) dense and the second level (outer level) sparse) index structure on customer id (c-id) for the relation given below: (12)

|  |  |  |
| --- | --- | --- |
| **c-id** | **Name** | **City** |
| C-1 | Name 1 | City 1 |
| C-2 | Name 2 | City 1 |
| C-3 | Name 3 | City 1 |
| C-4 | Name 4 | City 1 |
| C-5 | Name 5 | City 1 |
| C-6 | Name 6 | City 1 |
| C-7 | Name 7 | City 1 |
| C-8 | Name 8 | City 1 |
| C-9 | Name 9 | City 1 |
| C-10 | Name 10 | City 1 |
| C-11 | Name 11 | City 1 |
| C-12 | Name 12 | City 1 |

(b) Do you need to create index for city? Why? (3)

(c ) create a sparse index on c-id and delete one record that affect the index structure. Delete another record that does not affect the index structure. With this example, explain the deletion performance in sparse index.

Answer: a

Similar problem has been solved in class. See the class note.

Answer b

No need to create index for city because only one search key and any query for city shall return the entire relation. So no need to create index on city.

Answer c

Discussed in class.

Q. 2: (a)

Given the relational schema as follows:

Student (id, NID, name, f-name, f-NID, m-name, m-NID, DOB, cgpa, tot-cred, uni-id, uni-name, uni-street, uni-city, house-no, street, city, d-no, d-name, building)

Takes (id, course-no, semester, year, grade)

Course (course-no, title, credit, pre-req)

The record size for student, takes and course are 400, 100 and 80 bytes respectively. The block size is 4 KB. Show the slotted page structure after storage of one tuple (record) from each relation as per the above mentioned order. Show the steps.

Answer:

You have to store 400 byte student tuple first, then takes tuple and then course tuple as follows:

Step 1: Storage of one student tuple of 400byte

1 400 400byte

3696 4096

Step 2: Storage of one takes tuple of 100byte

2 400 100 100 400byte

3596 4096 3696

Step 3: Similar as step 2 (Do yourself)

(b) Show the file structure with free list after deletion of C3, C5 and C6 from the customer relation of question 1.

**Similar solution is available in the slide.** (5)

Q. 3: There are six disks D1, D2, D3, D4, D5 and D6. You have to store data blocks B1, B2, B3, B4, B5. B6 and B7 into the disks. Show the storage of the blocks into disks using RAID level 1 and RAID level 5.

See the class note. Similar problem has been solved.

Q.4: Given the relational schema with the size of each attribute as follows:

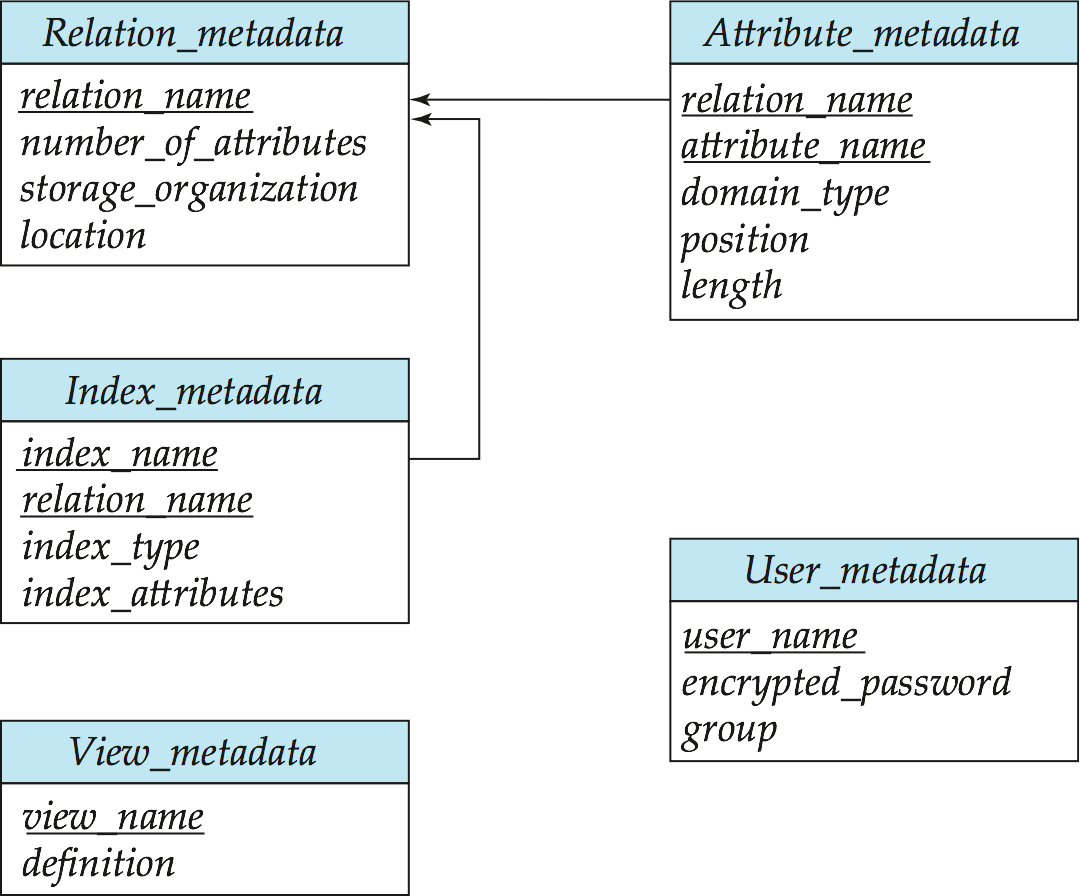
Student (id (10), NID (10), name (30), f-name (30), f-NID (10), m-name (30), m-NID (10), DOB (8), cgpa (8), tot-cred (2), uni-id (10), uni-name (30), uni-street (30), uni-city (30), house-no (10), street (30), city (30), d-no (8), d-name (30), building (30))

Takes (id (10), course-no (10), semester (14), year (8), grade (8), remarks (50))

Course (course-no (10), title (30), credit (1), pre-req (10), remarks (29))

For attributes, you can define appropriate domain types e.g., char, number, float, varchar etc.

Data dictionary schema has been given as follows:



1. After you create the table as per the above relational schema, show the data dictionary entries into the dictionary tables as above. The location is D:/DBMS/DD
2. How is DBMS restrict the users to create multiple tables with same name or in a table, multiple attributes with the same name?

Answer A

Relation metadata

|  |  |  |  |
| --- | --- | --- | --- |
| Relation\_name | Number\_of\_attributes | Storage\_organization | Location |
| student | 19 |  | D:/DBMS/DD |
| ….. |  |  |  |
| …. |  |  |  |
|  |  |  |  |

Attribute metadata

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Relation\_name | Attribute\_name | Domain\_type | Position | length |
| student | id | char | 1 | 10 |
| student | NID | char | 2 | 10 |
| …. |  |  |  |  |
| …. |  |  |  |  |

Q.5: Given the following relation schema for student. 3

Student (id, NID, name, f-name, f-NID, m-name, m-NID, DOB, cgpa, tot-cred, uni-id, uni-name, uni-street, uni-city, house-no, street, city, d-no, d-name, building)

ID is the primary key and the relation has been stored in the disk in sorted order of ID. You have created three indices on three attributes ID, uni-id and city. Identify which one is primary index and which one is secondary index? Which index can be sparse and which index must be sparse? Now fill up the following table.

|  |  |  |
| --- | --- | --- |
| Index Name | (Primary/Secondary) | Possible index (Dense/Sparse) |
| ID |  |  |
| Uni-id |  |  |
| City |  |  |

Q.6: Why is the disk access time is the main concern in developing algorithms for DBMS?