CHAPTER 10

Module IV

Indexing

Syllabus

Indexing : Basic Concepts, Ordered Indices, Index Definition in SQL

10.1 Index - Concept of Indexing

Q. What do mean by index ? Explain with example.

Q. Explain different indexing types in database management system. **MU - Dec.17, 5 Marks**

Q. What is SQL indexes? Explain types of indexes with example. **MU - May 19, 10 Marks**

1. Introduction

- Indexes are used to speed up search process like for example searching a book in library from many available books.
- In many places indexes are used for faster access to user data from set of data.
- Goals of indexing are to improve Speed and efficiency of search process.

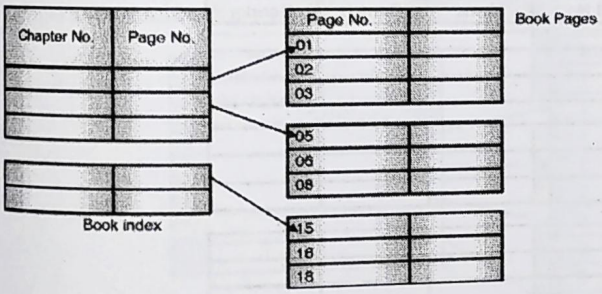

Fig. 10.1.1 : Indexing Concept

Search Key

- A field which is used to create index for database file, such field is called as search key.
- Any file is usually is composed of two parts,
  a. Data
  b. Index

Index File

- The index table is generally stored in a separate file which is called as the *index files*.
- Index file serves as the fast data dictionary index into an encoded file.
- For a given data file F, we create an index file called F_{idx}. Many file systems separate data and dictionary data, for efficiency and reliability.
- Data dictionary is considered more important and so it gets cached, stored, and updated separately.
- The index file is separate from the encoded file data for the same reasons and to allow us to manage each part separately and simply.

Efficiency of index based on,

- **Access type** : Type of data access that are supported efficiently
- **Access time** : Time to access a particular data item.
- **Insertion time** : Time taken to add a new data item
- **Deletion time** - Time taken to remove particular data item
- **Space overhead** : Extra space occupied by an index structure.

10.2 Ordered Indices

Q. What do mean by index ? Explain concept of Single level ordered index.

Q. Explain different indexing types in database management system. **MU - Dec.17, 5 Marks**

Q. Illustrate sparse and dense indexing with suitable example. **MU - Dec.18, 5 Marks**

1. Introduction

- Ordered access structures are similar to table of contents or indexes in books.
- The indexes or tables of contents list the chapter in sequential order, along with page numbers where the detailed information about it can be found.
- We are using indexing on a key field. If indexes are stored and sorted on basis of such search key field then it is called as ordered index.
- **E.g.** Author's Catalog in library
- Search key is a field, which uniquely identify the record in a file. Index files are typically much smaller than the original file. Index record holds search key value and pointers to the records with the value.
- Index-Sequential files are files (which hold information for data) ordered sequentially on a search key.

10.2.1 Single Level ordered Index

- The alternative to index of book is to have a linear search of the textbook, looking through each page one at a time.
- For a file consisting of several fields, and index access structure is usually defined on a single field, which is called an **indexing field**.
- An index usually is composed of two parts,
  a. Index field value
  b. List of pointers to all disk blocks which contain records with that field value.
- The values in the index are ordered, so we can do a binary search on such index.
- The index file is much smaller than the data file, so the binary search is much faster.
- There are different types of indexes, these include:

1. Primary Indexes
2. Clustering index
3. Secondary index

1. Primary index

- In a sequentially ordered file, the index whose search key specifies the sequential order of the data file is called as primary index.
- The search key of a primary index is usually primary key.
- Records of these file are of fixed length with two fields.
  a. Ordering key field primary key of data file
  b. Pointer to a disk clock.
- There is one index entry in the index file for each block in the data file.
- Two field values of index entry i as <K(i), P(i)>.
- Following is the example for primary key index.

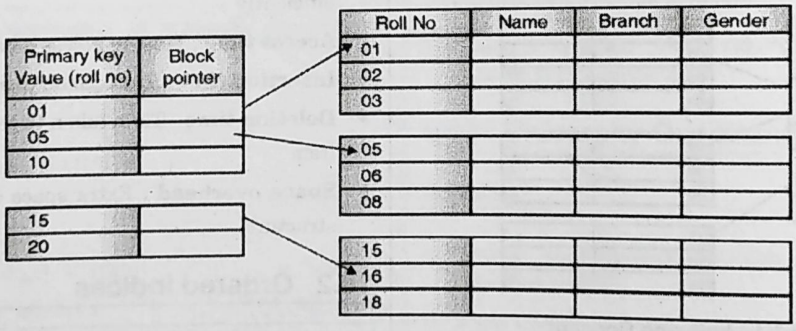

Fig. 10.2.1 : Primary Index

- The first record in each block of the data file is called anchor record or block anchor.
- The indexes can also be categorized into following types.

a. Dense index

- Index record appears for each and every search-key value in the file is called as dense index.
- In a dense primary index, the index record is

  < Search key value, Pointer to first record with search-key>

b. Sparse Index

- Sparse index contains index records for only some search-key values.
- It is Applicable when records are sequentially ordered on search-key.
- To locate a record with search-key value K we:
- Find index record with largest search-key value < K>
- Search file sequentially starting at the record to which the index record points

c. Comparison

- Sparse index occupies less space and less maintenance overhead for insertions and deletions.
- Sparse index is generally slower than dense index for locating records.
- Sparse index with an index entry for every block in file, corresponding to least search-key value in the block.
- Primary index is a non dense index.

2. Clustering indexes

- A clustering index is used to determine how rows in table are physically ordered (clustered) in a table space.
- Clustering indexes provide a good performance in some operations which involve large number of records.
- Examples are grouping operations, ordering operations, and comparisons.
- If records of a file are physically ordered based on any non key fields (a key which may not have a distinct value for each record) such field is called the clustering field. This differs from a primary index (primary key requires that the ordering field of the data file have a distinct value for every record).
- Clustering index is also an ordered file following fields.
  - Data file
  - Block pointer
- There is one clustering index for each distinct value of the clustering field, containing the data value and a pointer to its first block in the data file that has a record with that value for its clustering field.
- Clustering index is a non dense index.
- An index is similar to the directory structure used for extensible hashing. Both are searched to find a pointer to the data block containing the desired record.
- A main difference between index and hashing is that, index search uses the values of the search field itself; whereas it has function directory search uses the hash value that is calculated by applying some hash function to the search field.
- **E.g.** Clustering index can be applied on branch not for Student. Here there can be branch numbers duplicated in student table. Here the clusters are also contains many search key values into one cluster for example in first cluster the records are containing branch no 1 and 2.

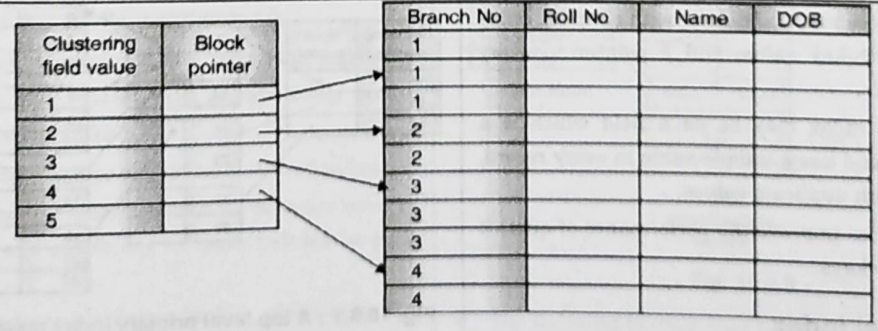

Fig. 10.2.2 : Clustering Index

For example clustering index with a separate block cluster for each group of records that share the same value for the clustering field.

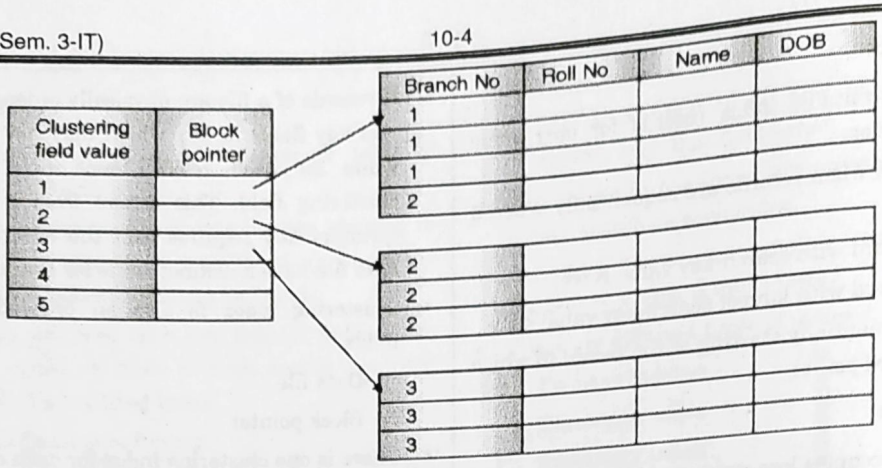

Fig. 10.2.3 : Clustering Index

3. Secondary index

- Secondary index is a type of index which is maintained for a data file, but not used to control the current processing order of the file.
- Example, a secondary index could be maintained for customer name, while the primary index is set up for customer_ID.
- Sometimes we may wants to find all the records whose values in a certain field fulfil some condition. This field may not the search-key of the primary index.
- We can use a secondary index with an index record for each search-key value.
- Secondary index is a non-clustering index.
- A secondary index provides a secondary means of accessing a file for which some primary access already exists.
- Secondary indices must be **dense**, with an index entry for every search-key value, and a pointer to every record in the file.
- The secondary index may be on a field which is a candidate key and has a unique value in every record, or a non key with duplicate values.
- Secondary indices improve the performance of queries on non-primary keys.

10.3 Multilevel Index

Q. Explain different indexing types in database management system **MU - Dec.17, 5 Marks**

- The idea of multilevel indexes is to reduce the part of the index that we continue to search by a larger factor. If primary index does not fit in memory, access becomes expensive.
- The blocking factor of the index, where the bfri is greater than 2. The blocking factor of the index, bfri is called the fan-out, or fo of the multilevel index.
- The index file is called the first level of a multilevel index. It is an ordered file with a distinct value for each key value K(i). Therefore we can create a primary index for the first level. This index to the first level is called the second level of the multilevel index.
- The number of third level entries is $r_3 = [(r_2/f_o)]$.
  - Outer index – a sparse index of primary index.
  - Inner index – the primary index file.
- If even outer index is too large to fit in main memory, yet another level of index can be created, and so on Indices at all levels must be updated on insertion or deletion from the file.

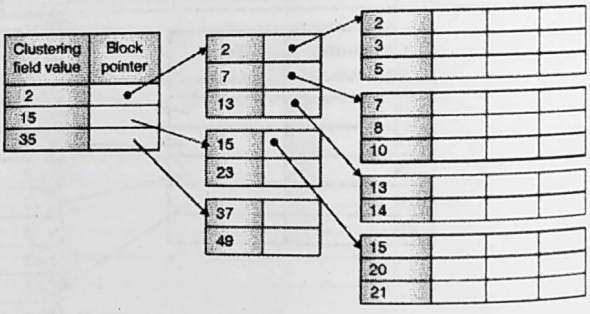

Fig. 10.3.1 : A top level primary index resembling ISAM (Indexed Sequential Access Method) organization

- A common file organization used in business data processing is an ordered file with a multilevel primary index on its ordering key field. Such organizations are called an indexed sequential file (Example is ISAM).

10.4 SQL Indexes

Q. Explain index
Q. Define Index.

1. Introduction

- An index can be created in a table to access data in database more quickly and efficiently.
- MySQL uses indexes to quickly search rows with specific column values as specified in query. Without an index, MySQL engine need to scan the entire table to locate the relevant rows. As table size increases the search speed will be reduced further.
- A database index is a data structure which will improves the speed of operations in a table. Indexes can be created using one or more columns

2. Need of Indexing

- Sometime while fetching data one or more columns in table are more repeating frequently in a WHERE clause of query then indexes on such columns can improve performance and speed of data retrieval.
- The index is generally required on column contains a wide range of values or a large number of null values.

3. Working of Index

- An index will make use of some data structure like B-Tree which will improves the speed of data retrieval on a table, it may include some additional write operations and storage for maintaining it.
- When we create any column of a table with a primary key or unique key, MySQL will automatically create an index named as PRIMARY.
- This type of index is also called the clustered index.
- The index creation must consider all columns used in SQL queries and create one or more indexes on such columns.
- Practically, indexes are type of table, which keep key field and a pointer to each record into the table.
- The INSERT and UPDATE statements will take more time on tables due to updates required in index information which creates extra burden on system, whereas the SELECT statements will become fast on such tables.

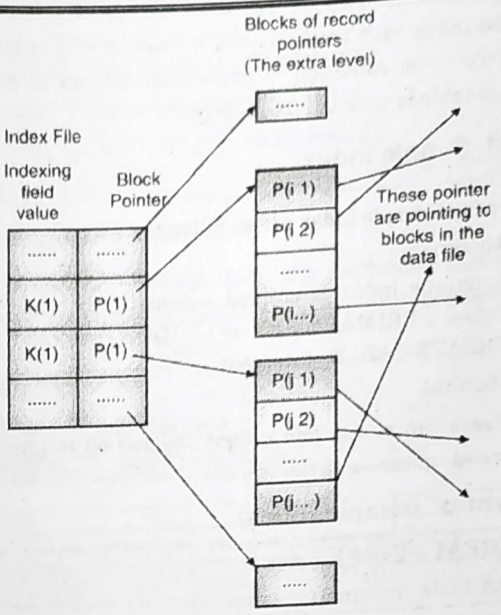

Fig. 10.4.1

- The PRIMARY index is stored together with the data in the same table. The clustered index will maintain the order of rows in the table.
- The indexes other than the PRIMARY index are called secondary indexes or also known as **non-clustered indexes.**

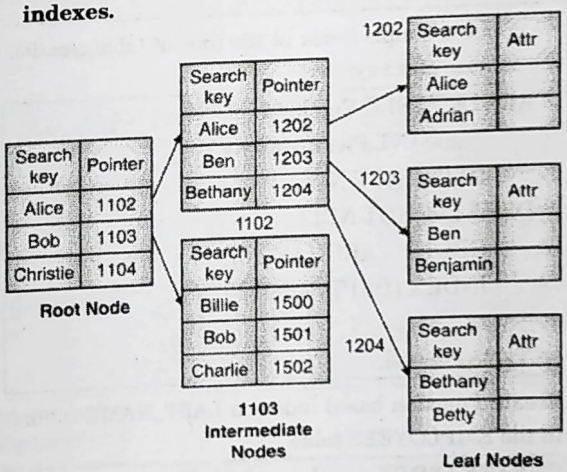

Fig. 10.4.2

- The indexing concept may not be useful if table is very small or there is no condition is applied on column in query also is table is updated very frequently.

4. Performance enhancement by indexing

- The query optimizer may use indexes to locate data at faster speed and without having to scan every row in a table for a given query.

- The index may not be useful if table is very small or there is no condition is applied on column in query also table is updated frequently.

10.4.1 Simple Index

Types of simple index are as follows,

1. **Automatic**

   A unique index is created automatically when you define a PRIMARY KEY or UNIQUE constraint in a CREATE TABLE statement.

2. **Manual**

   Users can create non unique indexes on columns to speed up access to the rows.

Syntax (simple Index)

```
CREATE INDEX index
ON table (column[, column]...)
```

Example of Simple Index

- Create index on LAST_NAME column in the EMPLOYEES table.

```
CREATE INDEX emp_last_name_idx
ON employees(last_name);
Index created.
```

- To create a simple index at the time of table creation, we can use given syntax,

```
CREATE TABLE Employee
(       Name INT PRIMARY KEY,
        ID INT NOT NULL,
    DEPT INT NOT NULL,
        GROUP VARCHAR(10),
        INDEX (DEPT, GROUP)
);
 Table created.
```

- Create Function based index on LAST_NAME column in the EMPLOYEES table.

```
CREATE INDEXemp_last_name_idx
ON     employees (UPPER(last_name))
Index created.
```

10.4.2 Composite Index

- A composite index is an index created on multiple columns of a table.
- MySQL will permit to create a composite index which includes up to 16 columns.
- A composite index is also known as a multiple-column index.
- The order of index columns is very important and it should be arranged in descending order as per their importance.
- Create index on Department name and age column in the EMPLOYEES table.

```
CREATE INDEX emp_Dept_Age_idx
ON employees (dept, age);
Index created.
```

- To create a composite index at the time of table creation, we can use given syntax,

```
CREATE TABLE table_name
(
  c1 data_type PRIMARY KEY,
  c2 data_type,
  c3 data_type,
  c4 data_type,
 INDEX index_name (c2, c3, c4)
);
```

10.4.3 Unique Index

- To enforce the uniqueness value of one or more columns, you often use the PRIMARY KEY constraint. However, each table can have only one primary key.
- Hence, if you want to have a more than one column or a set of columns with unique values, you cannot use the primary key constraint.
- Luckily, MySQL provides another kind of index called UNIQUE index that allows you to enforce the uniqueness of values in one or more columns. Unlike the PRIMARY KEY index, you can have more than one UNIQUE index per table.
- To create a unique index at the time of table creation, we can use given syntax,

```
CREATE UNIQUE INDEX index_name
ON
table_name(index_column_1,index_column_2,...);
```

- To create a unique index at the time of table creation, we can use below given syntax,

```
CREATE TABLE table_name
(
c1 data_type UNIQUE,
c2 data_type,
c3 data_type,
c4 data_type,
UNIQUE KEY U1(c2, c3, c4)
);
```

- Example to create a unique index at on employee table creation, we can use below given syntax,

```
CREATE TABLE IF NOT EXISTS employee (
id INT AUTO_INCREMENT PRIMARY KEY,
first_nameVARCHAR(50) NOT NULL,
last_nameVARCHAR(50) NOT NULL,
mobile VARCHAR(10) NOT NULL,
email VARCHAR(100) NOT NULL,
UNIQUE KEY unique_email (email)
);
```

Viewing Table Indexes

- To view the MySQL internally performed this query, you and the EXPLAIN clause at the beginning of the SELECT statement.

```
SELECT    *
FROM      Employees
WHERE     jobTitle = 'Sales'
```

	id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
▶	1	SIMPLE	employees	NULL	ref	jobTitle	jobTitle	52	const	17	100.00	NULL

- To view the MySQL internally performed this query, you and the EXPLAIN clause at the beginning of the SELECT statement.

```
SHOW INDEXES FROM employees;
```

10.4.4 Removing Index

- Drop above created index.

```
DROP INDEX upper_last_name_idx;
Index dropped.
  UPDATE    emp
  SET       sal = sal + 1000;
            rows_affected := to_char(SQL % rowcount);
  IF SQL % rowcount > 0 THEN
        DBMS_OUTPUT.PUT_LINE(rows_affected);
  ELSE
        DBMS_OUTPUT.PUT_LINE('NO ROWS AFFECTED');
  END IF;
END;
```

Review Questions

1. Explain concept of indexing.
2. Write short note on ordered indices.
3. Explain working of indexing.
4. Write short note on simple index.
5. Write short note on unique index.
6. Explain single level indexes with example.

10.5 University Questions and Answers

Dec. 2017

1. Explain different indexing types in database management system. (10 Marks)

Dec. 2018

2. Illustrate sparse and dense indexing with suitable example. (10 Marks)

May 2019

3. What is SQL indexes? Explain types of indexes with example. (10 Marks)

❑❑❑