Indexing

Problem without indexing

How to find a record or row in a big table?
Search linearly. Time complexity O(n).

What is Indexing?

Indexes are used to find rows with specific column values quickly. Without an index, MySQL must begin with the first row and then read through the entire table to find the relevant rows. The larger the table, the more this costs. If the table has an index for the columns in question, MySQL can quickly determine the position to seek to in the middle of the data file without having to look at all the data. This is much faster than reading every row sequentially.

Most indexes are stored in B-trees. More on the b-tree later in the session if time permits.

Indexing Hands-on

Open terminal and open MySQL cli with this command::

```
mysql -uroot
```

Restart mysql if you need to

```
brew services restart mysql
```

```
create database superx;

use superx;

CREATE TABLE Persons (
 ID int NOT NULL,
 LastName varchar(255) NOT NULL,
 FirstName varchar(255),
 Age int,
 PRIMARY KEY (ID)
 );

insert into Persons values(1, 'Thakur', 'Yuvan',5);
insert into Persons values(2, 'Pandey', 'Priyanka',30);
insert into Persons values(3, 'Stepniak', 'Agnieszka', 29);

CREATE INDEX idx_lastname ON Persons (LastName);
```

If you want to create an index on a combination of columns, you can list the column names within the parentheses, separated by commas:

```
CREATE INDEX idx_fullname ON Persons (LastName, FirstName);
```

Dropping an index:

```
DROP INDEX idx_fullname on Persons;
```

Index Use Cases:

1. Since index entries are stored in sorted order, indexes also help when processing ORDER BY clauses. Without an index the database has to load the records and sort them during execution. An index on age column will allow the database to process the following query by simply scanning the index and fetching rows as they are referenced. To order the records in descending order, the database can simply scan the index in reverse.

```
Select * from Persons order by age;
```

2. Improve query performance in general.

Index Drawbacks

1. Updating a table with indexes takes more time than updating a table without (because the indexes also need an update). So, only create indexes on columns that will be frequently searched against.

2. Indexes are a performance drag when the time comes to modify records.
Any time a query modifies the data in a table the indexes on the data must change also. Achieving the right number of indexes will require testing and monitoring of your database to see where the best balance lies. Static systems, where databases are used heavily for reporting, can afford more indexes to support the read only queries. A database with a heavy number of transactions to modify data will need fewer indexes to allow for higher throughput. Indexes also use disk space. The exact size will depends on the number of records in the table as well as the number and size of the columns in the index. Generally this is not a major concern as disk space is easy to trade for better performance.
3. Creating an index needs tuning and benchmarking, not a disadvantage though.

```
/* sql script*/

create database indexing;

use indexing;

create table emp(id int, name varchar(20));

create index age_idx on emp;

drop index age_idx on emp;


/* Java Program to generate inserts*/

package sql;


import java.util.Random;


public class Main {

    public static void main(String[] args) {

        Random random=new Random();


        for(int i=1;i<30000;i++)

        System.out.println("insert into emp values( "+i+", '"
+givenUsingJava8_whenGeneratingRandomAlphabeticString_thenCorrect() +"');");

    }


    public static String givenUsingJava8_whenGeneratingRandomAlphabeticString_thenCorrect() {

        int leftLimit = 97; // letter 'a'

        int rightLimit = 122; // letter 'z'

        int targetStringLength = 20;

        Random random = new Random();


        String generatedString = random.ints(leftLimit, rightLimit + 1)

                .limit(targetStringLength)

                .collect(StringBuilder::new, StringBuilder::appendCodePoint, StringBuilder::append)

                .toString();


        return generatedString;

    }

}
```

Indexing References

https://dev.mysql.com/doc/refman/8.0/en/mysql-indexes.html
https://odetocode.com/articles/237.aspx
https://dzone.com/articles/database-btree-indexing-in-sqlite