# Lab 9: Indexes

**Tailang Cao u1480633**

## Part 1 - Selecting Indexes

A database contains the following table for former-employee records:

| eID (int) (primary key) | Start Date (Date) | End Date (Date) |
|---|---|---|

There are two common queries performed on this table:

- Find all employees that started after a certain date
- Find all employees that started on a certain date, and worked until at least another certain date

Adding composite indexes on ( ‘Start Date , ‘End Date’) would make the process more efficient. When searching for employees started after the certain date, the start date part of the composite index would be helpful. When doing the second query, both part of the composite would be used. Creating single composite index would be more efficient then creating two indexes on start date and end date separately.

A database contains the following table for tracking student grades in classes

| studentID (int) (primary key) | className (varchar(10)) (primary key) | Grade (char(1)) |
|---|---|---|

This table only tracks whole letter grades, no '-' or '+' modifiers. Assume many students can take a class, and a student can take many classes. The primary key is created in the order (studentID, className).

The common queries performed on this table are:

- Get all students with a grade better than 'B'
- Get all classes where any student earned a grade worse than 'D'

Adding index on Grade.

Using the same grade database, but now the common queries are:

- Get all classes ordered by class name
- Get all students who earned an 'A' in a certain class

Adding index on Grade and className.

**Queries on the chess database**

Assume the only existing indexes are the primary index on each table (despite whatever indexes are on the actual tables. Remember, we're assuming there are NOT indexes created for foreign key constraints). Also assume the actual table sizes in the Chess database (ie big). Common queries are:

- `select Name from Players where Elo >=2050;`
- `select Name, gID from Players join Games where pID=WhitePlayer;`

1. Adding index on Player.Elo
2. Adding index on Games.WhitePlayer

**Queries on the public Library database**

Assume the only existing indexes are the primary index on each table (despite whatever indexes are on the actual tables). Common queries are:

- `select * from Inventory natural join CheckedOut;`

No additional indexes are required.

Assume the only existing indexes are the primary index on each table (despite whatever indexes are on the actual tables). Common queries are:

- `select * from Inventory natural join CheckedOut where CardNum=2;`
- `select * from Patrons natural join CheckedOut;`

1. Adding index on CheckOut.CardNum
2. No additional indexes are required.

Assume the only existing indexes are the primary index on each table (despite whatever indexes are on the actual tables). Also assume that the Library has been auto-scaffolded as discussed in class.

```
var query =
  from t in db.Titles
  select new
  {
    Title = t.Title,
    Serials = from i in t.Inventory select i.Serial
  };
```

Adding Index on Inventory.ISBN

## Part 2 - B+ Tree Index Structures

**Students table:**

Consider the students table from #2 in Part 1 above. Assume that an int occupies 4 bytes, and a varchar(10) occupies 10 bytes.

- How many rows of the table can be placed into the first leaf node of the primary index before it will split?

Student ID is an int which takes 4 bytes, className is a varchar(10) which takes 10 bytes, grade is a char(1) which takes 1 byte.
Therefore each row in the chart takes 15 bytes.
The leaf node of the primary index has 4096 bytes.
4096 / 15 = 273 rows.

- What is the maximum number of keys stored in an internal node of the primary index? (Remember to ignore pointer space. Remember that internal nodes have a different structure than leaf nodes.)

The key is student ID and className (size: 4 bytes + 10 bytes = 14 bytes).
4096 / 14 = 292.

- What is the maximum number of rows in the table if the primary index has a height of 1? (A tree of height 1 has 2 levels and requires exactly one internal node)

The B+ tree has an order of 292 + 1 = 293.
Each leaf node has 273 rows in it.
Therefore maximum number of rows is 293 * 273 = 79,989 rows.

- What is the minimum number of rows in the table if the primary index has a height of 1? (A tree of height 1 has 2 levels). The minimum capacity of a node in a B+ tree is 50%, unless it is the only internal/leaf node. The minimum number of children of a root node is 2.

One node can have up to 273 rows, more than 273 rows will cause a split.
Therefore the minimum number of rows for a height of 1 is 273 + 1 = 274 rows.

- If there is a secondary index on Grade, what is the maximum number of entries a leaf node can hold in the secondary index?

The grade is a char(1) which takes 1 byte. The primary key takes 14 bytes.
4096 / 15 = 273.
There for maximum number of entries is 273 rows.

**Another table**

Assume that for some table, rows occupy 128 bytes.

- What is the maximum number of leaf nodes in the primary index if the table contains 48 rows?
- What is the minimum number of leaf nodes in the primary index if the table contains 48 rows?

1. 4096 / 128 = 32 rows.
   As leaf nodes need to be half full, each node can hold minimum 32 / 2 = 16 rows.
   48 / 16 = 3.
   Therefore maximum number of leaf nodes is 3

2. In this case each node should hold 32 rows.
   Ceil (48 / 32) = 2
   Therefore minimum number of leaf node is 2.