**CS186 Discussion Section Week 2 Solutions**
**File Organization and Indexing**
**Fall 2013**

**What two formats are used for variable length records? What are the pros and cons of each?**

Fields delimited by special symbols (e.g. Hive uses \001 many other control characters that are not printable). Pros: Easy to implement, very simple setup. Cons: Slow sequential scan to find certain field (think 1000 fields). Special symbol might overlap with data in the field.

Array of field offsets. Pros: Quick jumping to new fields. Cons: cpu cost vs space tradeoff

**Why would you use fixed length records? Can you give specific examples? Can you give specific examples of when you would use them?**
Storing hashes, storing student ids, storing social security numbers.

**Why would you use variable length records? Can you give specific examples of when you would use them?**
When you don't know how big your field will be. A discussion post text can be anywhere from 100 - 100000 characters. You don't want to set aside 100000 bytes if the text could possibly just be 100 bytes long.

**Given:**
**CREATE TABLE Students (sid VARCHAR(20) not null, name VARCHAR(20), login VARCHAR(10), age INTEGER, gpa FLOAT not null)**

**What would be the size of a single tuple using fixed length records?**
20 bytes + 20 bytes + 10 bytes + 4 bytes + 4 bytes = 58 bytes

**What would be the maximum size of a single tuple using the two types of variable length records? Assume array offsets are stored as ints and field delimiters are one byte.**
Array of field offsets:
4 * 4 bytes (field offsets)
20 bytes + 20 bytes + 10 bytes + 4 bytes + 4 bytes = 58 bytes
---
74 bytes total

Field Delimiter:
20 bytes + 20 bytes + 10 bytes + 4 bytes + 4 bytes = 58 bytes
4 Delimiters * 1 byte = 4 bytes
---

62 bytes

**What would be the minimum size of a single tuple using the two types of variable length records? Assume array offsets are stored as ints.**
Array of field offsets:
4 * 4 bytes (field offsets) = 16 bytes
1 bytes + 0 bytes + 0 bytes + 0 bytes + 4 bytes = 5 bytes
---
21 bytes

Field Delimiter:
1 bytes + 0 bytes + 0 bytes + 0 bytes + 4 bytes = 5 bytes
4 Deilmiters * 1 byte = 4bytes
---
9 bytes

**Why do we use slotted pages?**
To manage variable length records. It's slotted because variable length records are inserted into slots.

**Let's say you have 52 bytes of free space in your slotted page. What is the biggest record you can add to the page? Let's assume that the entire page is only 256 bytes and you want to minimize the space the slot directory takes up.**
50 bytes because need to add 1 byte for offset and and 1 byte for length

**Fill in the following table for the I/O costs of operations (B = # disk blocks in file) (Please don't look at your lecture slides)**
**Assumptions:**
**Single record insert and delete.**
**For heap files, insert allows appends to end of file, delete leaves free space in page (don't have to worry about compacting data)**
**For sorted files, no gaps allowed and pages are fully packed and files are compacted after deletion. We are assuming worst case here.**

| | Heap File | Sorted File |
|---|---|---|

| | | |
|---|---|---|
| Scan All Records | B | |
| Equality Search (1 match) | | |
| Range Search | | |
| Insert | | |
| Delete | | |

**What does the System Catalog store?**
name, file location, file structure (heap file), for each attribute: attribute name and type, indexes (structure of index, search key fields), integrity constraints (foreign keys), statistics for query optimizer

**What are the important factors in determining whether or not you should add an index to a table?**
Can't use hash indexing with range queries. Should know which field to cluster on (calculate I/Os based on typical queries that you'll need to run). Decide if you even want to cluster (high maintenance cost)

**Consider the table Enrolled(sid, course, grade) with 500 pages, 6,000 tuples and the query SELECT * FROM Enrolled where sid > 4500. Assume SIDs are unique and range from 0 to 6000.**

**How many I/Os would this query take if the table was stored in a heap file?**
B = 500
**How many I/Os would this take if the table was stored in a sorted file sorted by grade?**
B = 500
**How many I/Os would this take if the table was stored in a sorted file sorted by SID?**
log_2(500) + ¼ * 500

**True or False? Given the table Students(sid, gpa, age), a hash index on gpa will**

**significantly increase the performance of the following query: SELECT * FROM Students WHERE age > 20;**
False

**True or False? Given the table Students(sid char(20), gpa float, age integer), a clustered tree based index on gpa will increase the performance of the following query: SELECT * FROM Students where age > 20 AND gpa > 3.5;**
True