CS109A Notes for Lecture 3/15/95

Primary Index

A data structure that makes it efficient to find a tuple, given the value of its domain (often a key) attribute(s).

- Hash table is an excellent choice.
- In practice, a balanced tree structure, called a *B-tree*, a "multiway BST," is more common.

Secondary Index

A binary relation with:

- Domain = values for some particular attribute(s) A.
- Range = pointers to (representations of) tuples.
- (v, p) means that p is a pointer to a tuple whose value in attribute A is v.
- Supports lookups that specify value of A.

Example:

- Classes, with a hash table on key Class.
- Secondary index on Weight.
  - ☐ Has, e.g., four pairs with 32000 as first component. Second components are pointers to cells holding tuples for Colorado, Kongo, Renown, Tennessee.
  - ☐ Possible implementation of secondary index: hash table with hash function based on Weight, with 17 buckets; $h(w) = w \bmod 17$.

Scheme Design Questions

How do we select:

1. The set of attributes (*relation scheme*) for our relations?

2. The set of relation schemes (= *database scheme*)?

3. Key(s) for each relation?

4. What secondary indexes to create?

Relations for "Entity Sets"

Many relations represent some "entity," e.g., the classes of ships or the ships themselves.

- These relations consist of a key attribute(s), e.g., Class, and other attributes that represent data about these entities, e.g., Weight, etc.

- Keep one relation for each kind of entity.

Example: The large table is a second relation, Ships, that complements the relation Classes from the previous lecture.

- Information unique to a ship, its name, year launched, and its class, is kept in Ships.

- Information common to all ships in a class belong in Classes.

  □ Avoids *redundancy*: saying the same thing for each member of a class.

Relations Connecting "Entity Sets"

Another choice of relation scheme: represent an important connection between entities.

- These relation schemes have attributes for the keys of each connected "entity set."

Example: Suppose we wish to represent entity sets students and courses, plus the "taking" relationship between them. Use:

- Student relation with key StudentID, plus other "information" attributes, e.g., Name, Address.

- Courses relation with attributes Number and Dept, which together form the key, plus other information attributes, e.g., Quarter offered.

Name	Launched	Class
Alabama	1942	South Dakota
Alaska	1944	Alaska
Anson	1942	King George V
Arizona	1916	Pennsylvania
Arkansas	1912	Wyoming
Barham	1915	Queen Elizabeth
California	1921	Tennessee
Colorado	1923	Colorado
Duke of York	1941	King George V
Fuso	1915	Fuso
Guam	1944	Alaska
Haruna	1915	Kongo
Hiei	1914	Kongo
Hood	1920	Hood
Howe	1942	King George V
Hyuga	1918	Ise
Idaho	1919	New Mexico
Indiana	1942	South Dakota
Iowa	1943	Iowa
Ise	1917	Ise
King George V	1940	King George V
Kirishima	1915	Kongo
Kongo	1913	Kongo
Malaya	1916	Queen Elizabeth
Maryland	1921	Colorado
Massachusetts	1942	South Dakota
Mississippi	1917	New Mexico
Missouri	1944	Iowa
Musashi	1942	Yamato
Mutsu	1921	Nagato
Nagato	1920	Nagato
Nelson	1927	Nelson
Nevada	1916	Nevada
New Jersey	1943	Iowa
New Mexico	1918	New Mexico
New York	1914	New York
North Carolina	1941	North Carolina
Oklahoma	1916	Nevada
Pennsylvania	1916	Pennsylvania
Prince of Wales	1941	King George V
Queen Elizabeth	1915	Queen Elizabeth
Ramillies	1917	Revenge
Renown	1916	Renown
Repulse	1916	Renown
Resolution	1916	Revenge
Revenge	1916	Revenge
Rodney	1927	Nelson
Royal Oak	1916	Revenge
Royal Sovereign	1916	Revenge
South Dakota	1942	South Dakota
Tennessee	1920	Tennessee
Texas	1914	New York
Valiant	1916	Queen Elizabeth
Warspite	1915	Queen Elizabeth
Washington	1941	North Carolina
West Virginia	1923	Colorado
Wisconsin	1944	Iowa
Yamashiro	1917	Fuso
Yamato	1941	Yamato

* Relation Taking has attributes StudentID, Number, and Dept.

Selecting Keys and Indexes

* What attribute(s) form a key depends on what we imagine the relation might hold as time goes on.
    - ☐ Ask yourself about policy, physics, etc.; e.g., will the administration ever issue the same ID to two students?
* What attribute(s) to use for primary or secondary indexes depends on what operations are likely to be performed.

Example: In Classes/Ships, suppose that the typical operations are:

1. Insert a new class and the ships of that class.
2. Given a ship name, find the year launched, its weight, and number of guns.
3. Given a number of guns, find the ships with that number of guns.

Here are some observations about desirable data structure.

* (1) is facilitated by any primary index, but we should pick one that is expected to yield the most nonempty buckets for each relation.
    - ☐ For Ships: index on Name.
    - ☐ For Classes: index on Class.
    - ☐ Neither is probably a key, but "close."
* (2) is facilitated by the Ships primary index on Name. It also uses the Classes primary index on Class, as we "navigate" from Ships to Classes using the Class value found in the relevant Ships tuple(s).
* (3) desires a secondary index for Classes on Guns. Then, after finding the selected classes, it desires a secondary index for Ships on Class.