Hash Indexes

CS 564- Fall 2015

ACKs: Dan Suciu, Jignesh Patel, AnHai Doan
Hash Indexes

- Good for equality search
- Not so good for range search
- Types of hash indexes:
  - static hashing
  - extendible hashing
  - linear hashing (not covered)
Static Hashing

- A hash index is a collection of buckets
  - bucket = primary page plus overflow pages
  - buckets contain data entries
- Uses a hash function $h$
  - $h(k) \mod M =$ bucket in which (data entry for) record
    with search value $k$ belongs
  - $M =$ # of buckets
**Static Hashing**

- Hash function works on *search key* field and distribute values over range \(0, ..., M-1\).

- What is a good hash function?
  - \(h(k) = (a \times k + b)\)
  - \(a\) and \(b\) are constants, and can help to tune \(h\)

- Issues with static hashing:
  - long overflow chains develop and degrade performance
  - reorganization is expensive and can block queries
EXTENDIBLE HASHING

Reorganize the file by doubling the number of buckets!

- directory of pointers to buckets
- On overflow, double the directory (not # of buckets)
- Why does this help?
  - Directory is much smaller than the entire index file
  - Only one page of data entries is split
  - No overflow page
**Extendible Hashing: Example**

- Directory an array

- **Search for k:**
  - Apply hash function $h(k)$
  - Take last `global-depth` # bits of $h(k)$

- **Insert:**
  - If the bucket has space, insert, DONE!
  - If the bucket is full, *split* it and *redistribute* the entries
  - If necessary, double the directory
MORE ON EXTENDIBLE HASHING

• How many disk accesses for equality search?
  – One if directory fits in memory, else two
• Directory grows in spurts, and, if the distribution of hash values is skewed, the directory can grow large
• We may need overflow pages when multiple entries have the same hash