CONSTRUCTOR UNIVERSITY

Physical Database Design

Ramakrishnan & Gehrke, Chapter 17 & 18

Alternative File Organizations

- File organization = Method of arranging a file of records on external storage
  - Goal: quickly find records needed by query
- Several alternatives
  - Heap files
  - Sorted Files
  - Indexes

Index Selection Guidelines

- Understand workload:
  - Queries vs. update
  - What relations (sizes!), attributes, conditions, joins (selectivity!), …?
- Attributes in WHERE clause are candidates for index keys
  - Exact match condition suggests hash index, range query suggests tree index
  - Consider multi-attribute search keys for several WHERE clause conditions
    - *Order of attributes important for range queries*
- Choose indexes that benefit as many queries as possible
  - impact on updates: Indexes make queries faster, updates slower
  - require disk space
- *understand how DBMS evaluates queries & creates query evaluation plans*

Decisions to Make

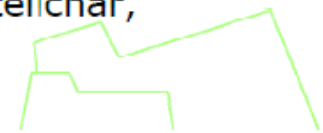
- What indexes?
  - Which relations? What field(s) search key? Several indexes?
  - For each index, what kind of an index should it be?
- Change conceptual schema?
  guided by workload, in addition to redundancy issues
  - Consider alternative normalized schemas? (many choices!)
  - "undo" some decompositions, settle for a lower normal form, such as 3NF? (denormalization)
  - Horizontal partitioning, replication, views ...see manuals
- If made after a database is in use, called schema evolution

Tuning Queries and Views

- query runs slower than expected?
  check if index needs to be re-built or statistics too old
- DBMS may not be executing plan you had in mind.
  Common problems:
  - Selections involving null values
  - Selections involving arithmetic or string expressions
  - Selections involving OR conditions
  - Lack of evaluation features like index-only strategies or certain join methods or poor size estimation
- Check plan used, adjust choice of indexes or rewrite query/view
  - Avoid nested queries, temporary relations, complex conditions, and operations like DISTINCT and GROUP BY

PS: A Moderately Complex Query

```
SELECT stadtbezirk, stadtteil, name, stadtteilchar, 'touche' AS entstehung, the_geom FROM
    (SELECT foo3.stadtbezirk, foo3.stadtteil, foo3.name, foo3.stadtteilchar, foo3.the_geom FROM
        (SELECT foo.gid, max(foo.laengste) AS laengste FROM
            (SELECT a.gid, b.stadtbezirk, b.stadtteil, b.name, b.stadtteilchar,
                (ST_Length(ST_Intersection(a.the_geom, ST_Union(b.the_geom)))) AS laengste
                FROM symdif a, dump b
                GROUP BY a.gid, a.the_geom, b.stadtbezirk, b.stadtteil, b.name, b.stadtteilchar
                HAVING ST_Touches(a.the_geom, ST_Union(b.the_geom))
                ORDER BY a.gid) AS foo
        GROUP BY foo.gid) AS foo2
        ,
        (SELECT a.gid, b.stadtbezirk, b.stadtteil, b.name, b.stadtteilchar, a.the_geom AS the_geom,
            (ST_Length(ST_Intersection(a.the_geom, ST_Union(b.the_geom)))) AS laengste
            FROM symdif a, dump b
            GROUP BY a.gid, a.the_geom, b.stadtbezirk, b.stadtteil, b.name, b.stadtteilchar
            HAVING ST_Touches(a.the_geom, ST_Union(b.the_geom))) AS foo3
    WHERE (foo2.gid = foo3.gid AND foo2.laengste = foo3.laengste)
    GROUP BY foo2.gid, foo3.stadtbezirk, foo3.stadtteil, foo3.name, foo3.stadtteilchar,
        foo3.laengste, foo2.laengste, foo3.the_geom) AS foo4
;
```



Key Performance Factors

Mark Fugate • My experience is that proper, or highest normal form normalization takes care of the first half of the optimization process by reducing the size of the stored data and reducing the numbers of operations required to maintain the data.

Query plans and query behaviours tell us how to properly index. Server tuning includes the proper storage media and knowledge of file systems and media tuning. Understanding your servers and knowing how to tune the OS, file systems, storage and kernel is all part of being a DBA.

Further, keeping SQL out of the client code makes all of the above attainable. I force all client applications in our shop to use stored procedures only. This gives me complete control over indexes, table structures, and all queries ensuring that nothing obnoxious enters the database.

1 day ago • Like

- Ref: discussion "what are the key points to improve the query performance" on the LinkedIn Database list, 2012-07-20

Summary

- Many alternative file organizations, each appropriate in some situation
- If selection queries frequent: sort file or build an index
  - Hash vs tree indexes vs sorted files
- Understand workload & DBMS query plans