Tuning Queries

Frequency of a search (retrieval) on an attribute (zip code) is very high, build an index for that attribute.

<table>
<thead>
<tr>
<th>Acct #</th>
<th>Name</th>
<th>Address</th>
<th>zip</th>
<th>tel</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td>35</td>
<td>Z2</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td>Z3</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td>Z1</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td>Z3</td>
<td></td>
</tr>
</tbody>
</table>

When the index attribute value gets modified (insert/delete/update) operations on the original table, the index entries need to be corrected accordingly.

? Frequency of changes to the attribute (zip) values.

Index Tuning wizard: Creation/Deletion of an index.

Statistics on attributes: number of unique values

no. of retrieval

no. of update
DB Objects
  Tables
  Views
  Index

  \{ one master data file \* mdf
  one transaction log file \* ldf

  one master data file \* mdf
  several secondary data files \* ndf
  one transaction log file \* ldf

  Grouped into filegroups

  By placing different secondary files (or filegroups)
  in different disks,
  disk access on multiple objects from different disks
  will improve query performance.

De Normalization

  \{ Merging (Joining) several base tables
  to provide a join result of frequently used queries

  Denormalized table is temporary.

  When a base table gets modified,
  the denormalized table must be revised or recreated.
Partitioning Tables

Horizontal Partitioning

Virtual Table
Customer

Partition 1
Partition 2
Partition 3

Queries
Site A

Site B

Site C

Vertical Partitioning

Virtual Student Table

PID Name Address major GPA

Partition 1
Partition 2

Queries
Site A
Site B

Both must include
Primary key PID