Overview of Storage and Indexing

Storing Data: Disks and Files

Chapters 8-9
Examples of Clustered Indexes

- B+ tree index on E.age can be used to get qualifying tuples.
  - How selective is the condition?
  - Is the index clustered?
- Consider the GROUP BY query.
  - If many tuples have E.age > 10, using E.age index and sorting the retrieved tuples may be costly.
  - Clustered E.dno index may be better!
- Equality queries and duplicates:
  - Clustering on E.hobby helps!
Indexes with Composite Search Keys

- **Composite Search Keys**: Search on a combination of fields.
  - **Equality query**: Every field value is equal to a constant value. E.g. wrt <sal,age> index:
    - age=20 and sal = 75
  - **Range query**: Some field value is not a constant. E.g.:
    - age = 20; or age=20 and sal > 10

- Data entries in index sorted by search key to support range queries.

Examples of composite key indexes using lexicographic order.
Composite Search Keys

- To retrieve Emp records with \( age=30 \) AND \( sal=4000 \), an index on \(<age,sal>\) would be better than an index on \( age \) or an index on \( sal \).
  - Choice of index key orthogonal to clustering etc.
- If condition is: \( 20<age<30 \) AND \( 3000<sal<5000 \):
  - Clustered tree index on \(<age,sal>\) or \(<sal,age>\) is best.
- If condition is: \( age=30 \) AND \( 3000<sal<5000 \):
  - Clustered \(<age,sal>\) tree index much better than \(<sal,age>\) index!
- Composite indexes are larger, updated more often.
Index-Only Plans

- A number of queries can be answered without retrieving any tuples from one or more of the relations involved if a suitable index is available.

\[
\begin{align*}
\text{SELECT } & E.dno, \text{ COUNT}(*), \\
\text{FROM } & \text{Emp } E, \\
\text{GROUP BY } & E.dno
\end{align*}
\]

\[
\begin{align*}
\text{SELECT } & E.dno, \text{ MIN}(E.sal), \\
\text{FROM } & \text{Emp } E, \\
\text{GROUP BY } & E.dno
\end{align*}
\]

\[
\begin{align*}
\text{SELECT } & \text{ AVG}(E.sal), \\
\text{FROM } & \text{Emp } E, \\
\text{WHERE } & E.age=25 \text{ AND } E.sal \text{ BETWEEN 3000 AND 5000}
\end{align*}
\]
Example

- Available Index:
  - Unclustered B+-tree on Grade

- Assumptions:
  - Number of pages: 4,000
  - Number of Students: 80,000
  - Grade is between 1 and 100
  - Grades are uniformly distributed
  - Non-leaf pages are in memory

```
SELECT S.id, S.grade
FROM Student S
WHERE S.grade > 70
```
**Example**

- **Available Index:**
  - Unclustered B+-tree on Grade

- **Assumptions:**
  - Number of pages: 4,000
  - Number of Students: 80,000
  - Grade is between 1 and 100
  - Grades are uniformly distributed
  - Non-leaf pages are in memory

```
SELECT S.id, S.grade
FROM Student S
WHERE S.grade > 98
```
Example

- Available Index:
  - Unclustered B+-tree on Grade
- Assumptions:
  - Number of pages: 4,000
  - Number of Students: 80,000
  - Grade is between 1 and 100
  - Grades are uniformly distributed
  - Non-leaf pages are in memory

```sql
SELECT S.id, S.grade
FROM Student S
WHERE S.grade > X
```