Physical Database Design (ch. 16 & ch. 3)

Introduction

- The purpose of physical database design is to translate the logical description of data into the technical specifications for storing and retrieving data.
- The goal is to create a design for storing data that will provide adequate performance and insure database integrity, security and recoverability.

Inputs to Physical Design

- Normalized relations.
- Volume estimates.
- Attribute definitions.
- Data usage: entered, retrieved, deleted, updated.
- Response time requirements.
- Requirements for security, backup, recovery, retention, integrity.
- DBMS characteristics.

Physical Design Decisions

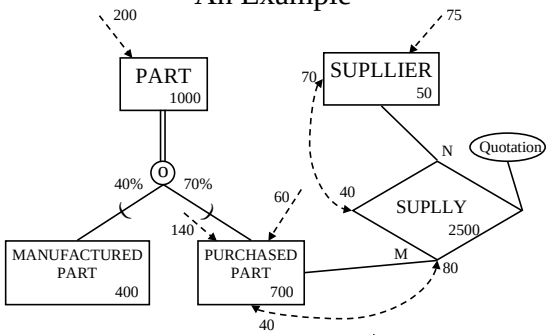
- Specifying attribute data types.
- Modifying the logical design.
- Specifying the file organization.
- Choosing indexes.

Data Volumes and Query Frequencies

- Data volumes: estimation of number of data records in each entity
- Query frequencies: estimation of number of queries per hour towards each entity

These two types of information are useful for determining the storage requirements and performance requirements, which are needed to make physical design decisions.

An Example



Designing Fields

- Choosing data type -- Char(8), Date, etc.
- Coding, compression, encryption.
- Controlling data integrity.
  - Default value.
  - Range control.
  - Null value control.
  - Referential integrity.

An example of code look-up table

PRODUCT File

Product_No	Description	Finish	…
B100	Chair	C	
B120	Desk	A	
M128	Table	C	
T100	Bookcase	B	
…	…	…	

FINISH Look-up Table

Code	Value
A	Birch
B	Maple
C	Oak

Coding is a way to achieve compression

Designing Fields

- Handling missing data.
  - Substitute an estimate of the missing value.
  - Trigger a report listing missing values.
  - In programs, ignore missing data unless the value is significant.

Physical Records

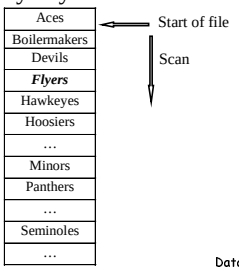
- Physical Record: A group of fields stored in adjacent memory locations and retrieved together as a unit.
- Page: The amount of data read or written in one I/O operation. A page contains usually a number of physical records.
- Blocking Factor: The number of physical records per page.

Designing Physical Files

- Physical File: A file as stored on the disk.
- Constructs to link two pieces of data:
  - Sequential storage.
  - Pointers.
- File Organization: How the files are arranged on the disk.
- Access Method: How the data can be retrieved based on the file organization.

Sequential File Organization

- Records of the file are stored in sequence by the primary key field values.



Indexed File Organizations

- Index: an auxiliary file to improve access efficiency of the main data file.
- B-tree or B^+-tree index.
- Bitmap index
  - Ideal for attributes that have even a few possible values
  - Often requires less storage space
  - Can be used for multiple keys

Bitmap Index on Product Price attribute

	Product Table Row Number									
Price	1	2	3	4	5	6	7	8	9	10
100	0	0	1	0	1	0	0	0	0	0
200	1	0	0	0	0	0	0	0	0	0
300	0	1	0	0	0	0	1	0	0	1
400	0	0	0	1	0	1	0	1	1	0

Product 3 and 5 have Price $100
Product 1 has Price $200
Product 2, 7, and 10 have Price 300
Product 4, 6, 8, and 0 have Price $400

Hashed File Organization

- Hashing Algorithm: Converts a primary key value into a record address.
- Division-remainder method:
  - Given 1000 pages to store employee records
  - Choose the prime number closest to 1000, i.e., 997
  - The bucket number of each record is equal to the remainder of employee ID divided by 997
  - Finding the location of any employee record needs only a computation.

Comparison of File Organizations

- Sequential:
  - No waste space
  - Fast sequential retrieval
  - no random retrieval
  - update requires reorganization and slow

Comparison of File Organizations

- Indexed
  - require additional space for index
  - support random retrieval
  - deletion, addition, and update of records require modification of indexes

Comparison of File Organizations

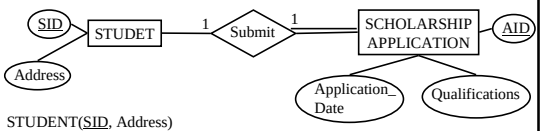
- Hashed
  - May require overflow pages
  - sequential retrieval is impractical
  - random retrieval on primary key is very fast since it does not need to access index
  - deletion, addition, and modification of records are relatively easy

Denormalization

- The reversal of normalization in order to increase query processing efficiency.
- During physical database design, denormalization is done if performance consideration dominates the issue of operational anomalies.

An example

- Two entities with a one-to-one relationship



STUDENT(SID, Address)

APPLICATION(AID, Application_date, Qualification, *SID*)

Denormalized relation: STUDENT(SID, Address, AID, Application_Date, Qualification)

Partitioning

- Horizontal Partitioning: Distributing the rows of a table into several separate files.
- Vertical Partitioning: Distributing the columns of a table into several separate files.
  - The primary key must be repeated in each file.

Partitioning

- Advantages of Partitioning:
  - Records used together are grouped together.
  - Each partition can be optimized for performance.
  - Security, recovery.
  - Partitions stored on different disks.
  - Take advantage of parallel processing capability.
- Disadvantages of Partitioning:
  - Slow retrievals across partitions.
  - Complexity.