Database - 1

3rd Class

Computer science Department

10th Lecture- Handling Missing Data , Designing Physical Files and Denormalization

Sunday 24th of November 2024

LECTURER :

DR. RAYAN YOUSIF ALKHAYAT

Handling Missing Data:

- When field may be null, simply entering no value may be sufficient.
- For example, suppose a customer ZIP code field is null and a report summarizes total sales by month and zip code, how should a sales to customers with unknown zip codes be handled? Two options for handling preventing missing data have already be mentioned: using a default value and not permitting missing (null values) . Missing data are inevitable.
- Other possible methods for handling missing data are the following:

1. Substitute and estimate of the missing value

- For example , for a missing sales value when computing monthly product sales, use a formula involving the mean of an existing monthly sales values for that product indexed by total sales for that month across all products.
- Such estimates must be marked so that users know that theses are not actual values.

2. Track missing data

- So that special reports and other system elements cause people to quickly resolve unknown values.
- This can be done by setting up a trigger in the database definition.
- A trigger is a routine that will automatically execute when some events occur or time period passes.
- One trigger could lock a missing entry to a file when a null or other missing value is stored and another trigger vane periodically to create a report of the contents of this log file.

3. Perform Sensitivity Testing

- Missing data are ignored unless knowing a value might significantly change results; if, for example, total monthly sales for a particular sales person are almost over a threshold that makes a difference in that person compensation

- This is the most complex of the methods mentioned and hence required the sophisticated programming which must be written in application programs since DBMS's do not have the sophistication to handle this method .

DESIGNING PHYSICAL RECORDS AND DENORMALIZATION:

- In a Logical data model, you grouped into a relation those attributes that are determined by the same primary key.
- In contrast, a physical record is a group of fields stored in adjacent memory locations and retrieved together as a unit.
- The design of a physical record involves choosing the sequencing of fields into adjacent storage locations to achieve two goals efficient use of secondary storage and data processing speed.

- The efficient use of secondary storage is influenced by both the size of the physical record and the structure of secondary storage.
- Computer operating systems read data from hard disks in units called pages, not physical records.
- A page is the amount of data read or written in a secondary memory input or output operation .
- The page size is fixed by system programmers and is selected to use RAM most efficiently across all applications.
- Depending on the computer system a physical record may or may not be allowed to span two pages.

- Thus, if page length is not an integer Multiple of the physical record length, wasted space may occur at the end of a page.
- The number of physical records per page is called the blocking factor. If storage space is scarce and physical records cannot span pages, creating multiple physical records from one logical relation will minimize wasted storage space.

Denormalization

- The preceding discussion of physical record design concentrated an efficient use of storage space. In most cases the second goal of physical record design (efficient data processing) dominates the design process.
- Efficient processing of data, just like efficient accessing at books in a library, depends on how close together related data (or book) are.

Denormalization

- **Often all the attributes that appear with a relation are not used together and data from different relations are needed together to answer a query or produce a report.**
- **Thus, although normalized relations solve data maintenance anomalies normalized relations, if implemented one for one as physical record, may not yield efficient data processing**

- Denormalization: is the process transforming normalized physical record specifications.
- In general, denormalization may partition a relation into several physical records, may combine attributes from several relations together Into one physical record or may do a combination of both.

- Denormalization can increase the chance of errors and inconsistencies and can force reprogramming systems if business rules changed. Further, denormalization optimizes certain data processing at the expense of others, so if the frequencies of different processing activities change, the benefits of denormalization may no longer exists.

DESIGNING PHYSICAL FILES

A physical file in a named portion of secondary (hdd or ssd) allocated for the purpose storing physical records.

Some computer operating systems allows a physical file to be split into separated pieces. If this occurs, it typically be transparent to you as the database designer.

In subsequence, we will assume that file is not split and each record in a file has the same structure.

Pointer

- All files ,are organized by using two basic constructs to link one piece of data with another piece of data: sequential storage and pointers.
- With sequential storage, one field or record is stored right after another 'field or record.
- Although simple to implement and use, sometimes sequential storage is not the most efficient way to organize data.

- A **pointer** is a field of data that can be used to locate a related field or record of data.
- In most cases, a pointer contains the address, or location, of the associated data. Pointers are used in a wide variety of data storage structures.

Access Methods

- All input /Output operations are ultimately handled by the data management portion of the computer’s operating system. Each operating systems supports one or more algorithms for storing and retrieving data; these algorithms are called access methods.
- There are two basic types of access methods relative and direct.

1. A relative access - method support accessing data as an offset from the most recently referenced point in secondary memory .

- A sequential access method is a special case of this type since the "next" record begins the distance one record from the beginning of the current record.
- In general, a relative access method supports finding the *n*th record from the current position or from the beginning of the file.

2. A direct access method: uses a calculation to generate the beginning address of a record.

The simplest form of a direct method is to tell the access method to go to a particular disk address.

Another variation is to provide a record's primary- key and the direct access method determines where this record should be located. we will discuss this access method under the more inclusive concept of file organizations in the nest section.

END OF LECTURE 10