Module_4

Physical DB Design and DB Security

CS 3410 DB

KENNESAW STATE UNIVERSITY

Introduction

- Physical database design is the process of transforming logical data models into physical data models.
- Conceptual data modeling is a map of concepts and their relationships used for databases.
- The relational model is one of the most commonly used models in contemporary database applications.
- The principles of logical database design for the relational model apply to many other logical models as well.

Background

- The first database was developed in the 1960s when computers were mostly used for private organizations.
- The two most popular data models during the 1960s were the CODASYL (Conference/Committee on Data Systems Languages) and IMS (Information Management System).
- These database systems began to change during the 1970s when E.F Codd published a paper on his revolutionary ideas about the relational model/database.

Background Continued…

- The ER(Entity-Relationship) Diagram was developed in 1976. By P. Chen. The ER module allows developers to focus less on the logic table structure and more on research data application.
- In 1980, SQL(Structured Query Language) became the standard query language among databases.
- Before 1980, Government organizations were the first ones to invest heavily into security of data.
- Today, research is still being conducted on database security and continues to evolve year to year.

KENNESAW STATE UNIVERSITY

Logical Design vs Physical Design

- The logical design is made up of several characteristics such as: Entity, Relationship, Attribute(s), and a Unique Identifier.

- The physical design consists of a table, foreign key, column(s), and a primary key.

[4.2.1] Physical Database Design Process

- The design of a physical database design is heavily influenced on integrity and performance.
- According to Adrienne Watt, database design starts with a conceptual data model and produces a specification of a logical schema.
- The database design process is initialized from the logical data model that will be used in the database design and can be represented as an E-R(Entity-Relationship) diagram.
- Physical database design is concerned with the design of fields. A field is the smallest unit of application data recognized by system software.

[4.2.2] Data Partitioning

- Partitioning is a concept in databases in which very large tables and data are partitioned into smaller, individual tables, and queries.
- Horizontal partitioning is the classification of the rows based on common characteristics into several, separate tables.
- In range partitioning, each partitioned portion is characterized by a range of values for one or multiple columns such as IDs, or dates.
- Hash partitioning is the spreading of data in even partitions autonomous of the key value.
- List partitioning is a technique where a list of distinct values is defined as the partitioning key in the characterization for each partition

Pros of Data Partitioning

- Partitioning is practical and helps manage the table because partitioning helps identify the area where maintenance is needed and saves storage space.
- Partitioning is also secure as only the relevant and necessary data can be specifically chosen and accessed by the user.
- Backing up and securing files is easier due to their smaller size and if one file is corrupted, the other is still accessible.
- Partitioning also helps with balancing the load. The partitioned files can be designated to different storage locations which reduces conflict and maximizes performance.

KENNESAW STATE UNIVERSITY

Cons of Data Partitioning

- Partitioning is inconsistent with the access speed. Due to all partitions not being identical, the access speeds tend to differ.
- Due to the complex nature of partitions, the code required to program will need to be more complex, and challenging.
- Partition takes up excess storage space and time.

[4.3.1] Describe three types of file organization

File Organization

- According to (Venkataraman, R., Topi, H. 2011) a "*file organization* is a technique for arranging the records of a file on secondary storage devices."
- With modern relational DBMS it is not necessary to design file organizations, but you are to be allowed to select an organization and its parameters for a table or physical file.
- In choosing a file organization for a particular file in a database consider seven important factors: Fast data retrieval, high throughput for processing data input and maintenance transactions, Efficient use of storage space, Protection from failures or data loss, minimizing need for reorganization, accommodating growth and security from unauthorized use

Sequential & Indexed File Organization

- **Sequential File Organization**
- **In a sequential file organization, the records in the file are stored in sequence according to a primary key value.**
- **To locate a particular record, a program must normally scan the file from the beginning until the desired record is located. A common example of a sequential file is the alphabetical list of persons in the white pages of a telephone directory.**
- **A comparison of the capabilities of sequential files with the other two types of files can be seen in figure 1.3. “Because of their inflexibility, sequential files are not used in a database but may be used for files that back up data from a database.” (Venkataraman, R., Topi, H. 2011)**

KENNESAW STATE UNIVERSITY

Indexed File Organization

- **The records are stored either sequentially or not sequential, and an index is created that allows the application software to locate individuals. “A card catalog in a library, an *index* is a table that is used to determine in a file the location of records that satisfy some condition.”(Venkataraman, R., Topi, H. 2011) Each index entry matches a key value with one or more records.**
- **An index can point to unique records or to potentially more than one record. According to (Venkataraman, R., Topi, H. 2011) “an index that allows each entry to point to more than one record is called a *secondary key* index.” Secondary key indexes are important for supporting many reporting requirements and for providing rapid ad hoc data retrieval.**
- **An example would be an index on the ProductFinish column of a Product table. Because indexes are extensively used with relational DBMSs, and the choice of what index and how to store the index entries matters greatly in database processing performance.**

KENNESAW STATE UNIVERSITY

Hash File Organization

- To determine or compute the address of a record within a file is to a hash file organization can be used and implemented as algorithm or function.
- A hash algorithm is an algorithm that takes an input of random size and proceeds to transform the input such that the hash result is an output of fixed length.
- Once the output is determined or computed, the hash result is irreversible, meaning that the algorithm can only process data in one-way.
- The use of hashing algorithms is commonly found in databases for practically any website that requires a password to login to an account and is illustrated in Figure 1.4.

Clustering File Organization:

- Defining a table to be in only one cluster reduces retrieval time for only those tables stored in the same cluster.
- This technique of file organization is known as clustering files and is illustrated in Figure 1.5.

KENNESAW STATE UNIVERSITY

Hash File Organization

```
CREATE CLUSTER Ordering (CustomerID CHAR(25));
```

The term Ordering names the cluster space; the attribute CustomerID specifies the attribute with common values.

Then tables are assigned to the cluster when the tables are created, as in the following example:

```
CREATE TABLE Customer_T(
   CustomerID                    VARCHAR2(25) NOT NULL,
   CustomerAddress               VARCHAR2(15)
   )
   CLUSTER Ordering (CustomerID);
CREATE TABLE Order_T (
   OrderID                  VARCHAR2(20) NOT NULL,
   CustomerID               VARCHAR2(25) NOT NULL,
   OrderDate                DATE
   )
   CLUSTER Ordering (CustomerID);
```

KENNESAW STATE UNIVERSITY

Security

- *Database files are stored in a proprietary format by the database which allows for access controls over the files.*
- *A useful procedures to consider are backups to ensure that stored data may be retrieved in the event that data may be compromised.*
- *Another technique employs the utilization of encryption to encrypt data contained within files and allow for only programs with access to decrypt the encrypted files to read them.*
- *Encryption involves two methods of encryption: symmetric encryption and asymmetric encryption. Symmetric encryption makes use of a single key for all parties communicating and is used for both encrypting data and decrypting data.*
- *Asymmetric encryption makes use of two keys for all parties communicating where the first key is used for encryption and the second key is used for decryption.*

Comparison of File Organization

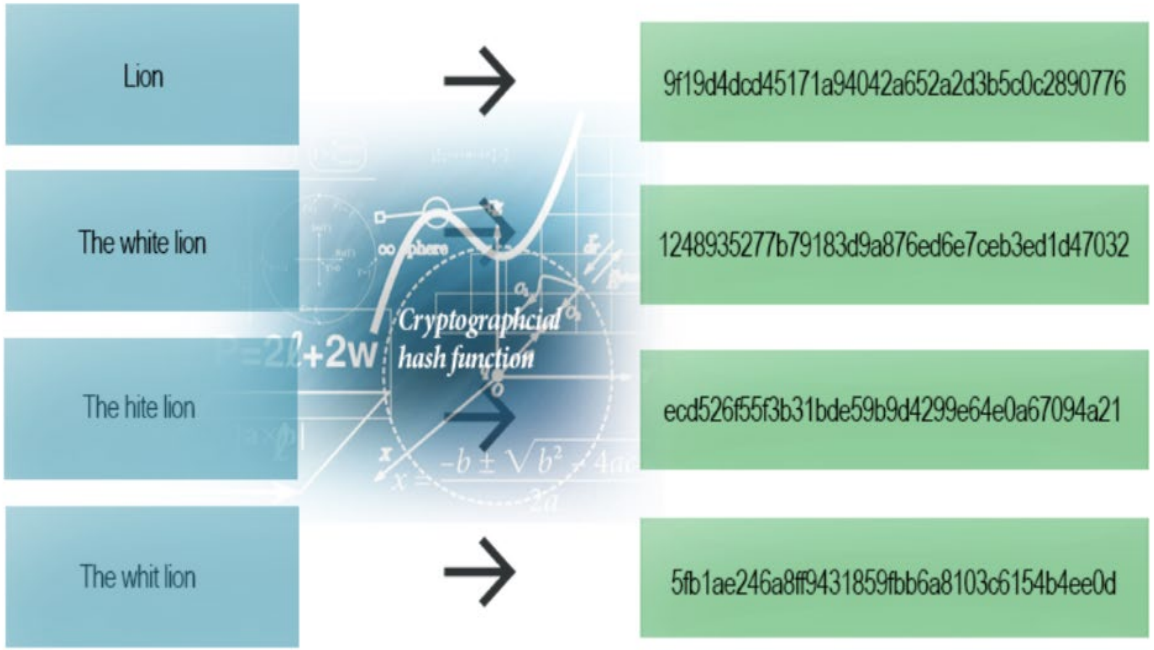


Figure 1.4. Hashing Algorithm, by jscrambler, 2020, https://blog.jscrambler.com/hashing-algorithms/. Copyright 2020 by jscrambler

KENNESAW STATE UNIVERSITY

Comparison of File Organization

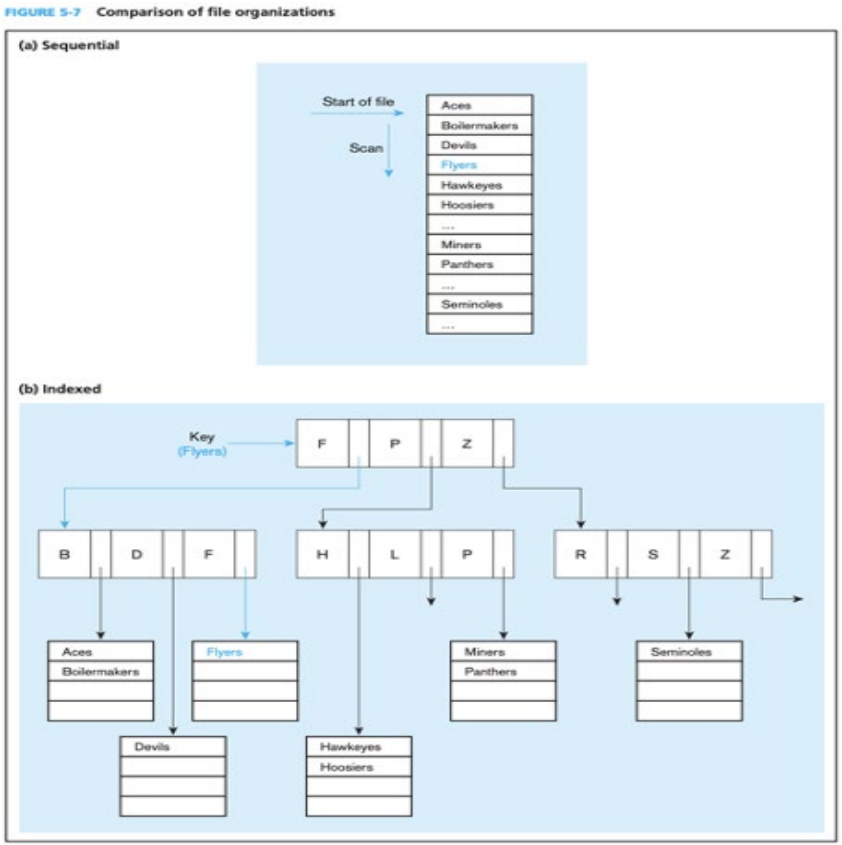


Fig 1.3 Modern Database Management 10th edition.((Venkataraman, R., Topi, H. 2011))

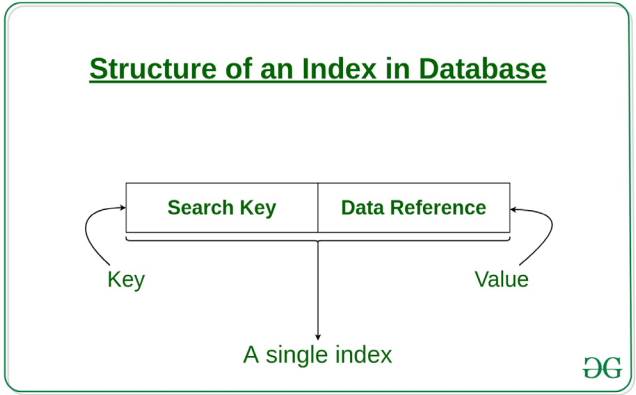
Comparative Features of different File Organization

TABLE 5-3 Comparative Features of Different File Organizations

	File Organization		
Factor	**Sequential**	**Indexed**	**Hashed**
Storage space	No wasted space	No wasted space for data but extra space for index	Extra space may be needed to allow for addition and deletion of records after the initial set of records is loaded
Sequential retrieval on primary key	Very fast	Moderately fast	Impractical, unless using a hash index
Random retrieval on primary key	Impractical	Moderately fast	Very fast
Multiple-key retrieval	Possible but requires scanning whole file	Very fast with multiple indexes	Not possible unless using a hash index
Deleting records	Can create wasted space or require reorganizing	If space can be dynamically allocated, this is easy but requires maintenance of indexes	Very easy
Adding new records	Requires rewriting a file	If space can be dynamically allocated, this is easy but requires maintenance of indexes	Very easy, but multiple keys with the same address require extra work
Updating records	Usually requires rewriting a file	Easy but requires maintenance of indexes	Very easy

[4.4.1] Translate a database model into efficient structures

- Database manipulations demand the location of a row or a collection of rows that satisfies a condition.
- Searching for data can be quite the laborious task, given the magnitude of a database. Hence, using indexes can vastly increase the speed of the process and reduce the time and work.
- The usage and definition of indexes are a crucial spoke on the wheel of physical database design. Indexes are defined as either a primary key, secondary key, or both. It is ordinary to define an index for the primary key of a table.
- The index is formed of two columns: one column for the key and the other column for the address of the record that consists of the key value. In the case of a primary key, the index will only have one entry for each key value.
-



Indexing in Databases, by GeeksforGeeks, 2020, https://www.geeksforgeeks.org/indexing-in-databases-set-1/ Copyright 2020 by GeeksforGeeks.org

KENNESAW STATE UNIVERSITY

Using, Selecting, Creating & When to use Indexes

- **Using and selecting Indexes**
- Given the magnitude of a database, searching for data can be quite the laborious task. Hence, using indexes can vastly increase the speed of the process and reduce the time and work. The usage and definition of indexes are a crucial spoke on the wheel of physical database design. Indexes are defined as either a primary key, secondary key, or both. It is ordinary to define an index for the primary key of a table. The index is formed of two columns: one column for the key and the other column for the address of the record that consists of the key value. In the case of a primary key, the index will only have one entry for each key value.
- **Creating a unique index**
- The syntax to create a unique key index in SQL is "CREATE [UNIQUE] INDEX index_name ON table_name(column1, ... column_n);". The UNIQUE modifier specifies the values in the indexed columns. Creating a non unique key index is equivalent to a secondary key index. The term UNIQUE isn't used to create a secondary key index because values can be repeated.
- **When to use indexes**
- It is important to know when to use an index and which attributes to use when creating an index. Using indexes come at the price of performance. Performance is compromised when using indexes due to the overload for maintenance for insertions, deletions, and updating records. For this reason, indexes should be utilized mainly for data retrieval. According to the book, “Indexes should be used judiciously for databases that support transaction processing and other applications with heavy updating requirements, because the indexes impose additional overhead” (Hoffer, Venkataraman, & Topi, 2011). Here are some rules or conditions that suggest the use of indexes.
-

Rules & Conditions That Suggest The Use of Indexes

- 1. Indexes are a lot more efficient and practical for substantial tables.
- 2. Indexes are useful when there is a need to set out a unique index for the primary key.
- 3. Indexes are frequently used for columns that appear in WHERE modifiers of SQL commands.
- 4. Indexes should be used when for attributes referenced in ORDER BY and GROUP BY statements.
- 5. Indexes are convenient when there is diversity in the values of an attribute. For Oracle's standards, it is unproductive to use an index when an attribute has fewer than 30 values.
- 6. One point to keep in mind is to consider developing a compressed version of the values. Doing this will ensure that the index isn't slower to process.
- 7. If the index is used for finding the location of where the record will be stored, make sure the key of this index is a surrogate key to ensure the records will be fairly spread across the storage space.
- 8. Make sure to check the limit of indexes on the DBMS because some systems do not allow for more than 16 indexes.
- 9. Find a way to index attributes that have null values because rows with a null value won't be referenced.
-

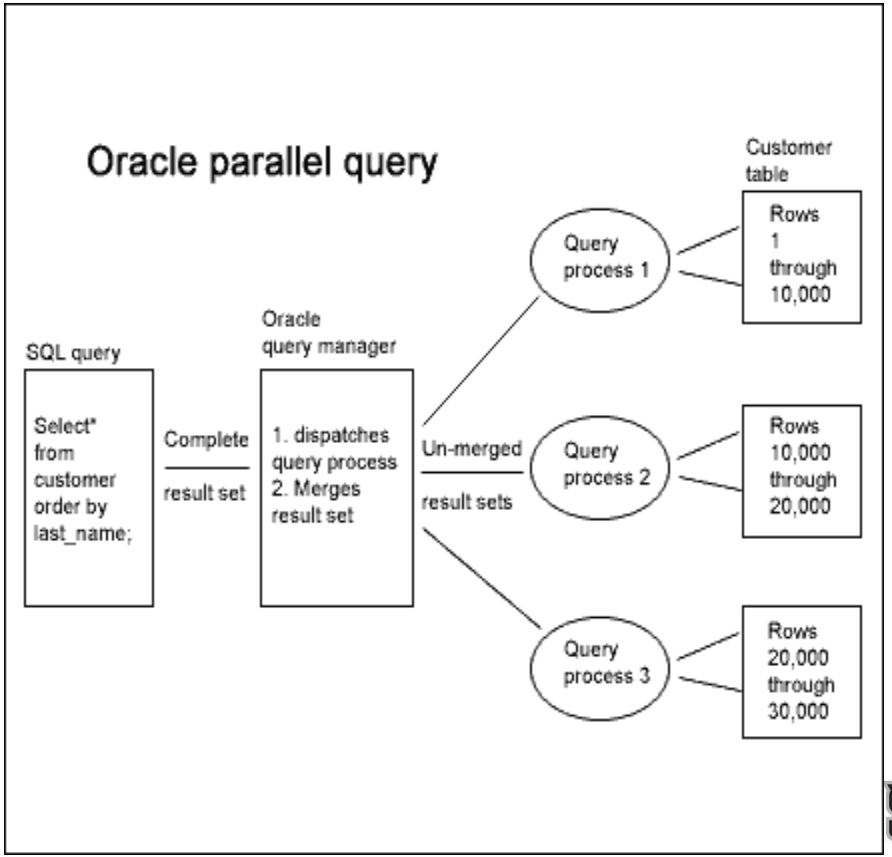
4.4.1 Designing a Database for optimal Query Performance

- Database processing can include adding, deleting and modifying a database along with method of retrieving data. The amount of work required to optimize query for performance heavily relies on DBMS.
- Architecture of modern computers has changed greatly over the years and the use of multiple processors in database servers has become standard. Symmetric multiprocessor (SMP) is commonly used in database servers to allow multiple processing. DBMS that use parallel query processing include planning on breaking up a query that can be processed in parallely by different processors.
- "Suppose you have an Order table with several million rows for which query performance has been slow. To ensure that subsequent scans of this table are performed in parallel, using at least three processors, you would alter the structure of the table with the SQL command:"(Hoffer, Venkataraman, & Topi, 2011)
- ALTER TABLE ORDER_T PARALLEL 3; (*Hoffer, Venkataraman, & Topi, 2011*)

- Schumacher reported, "on a test in which the time to perform a query was cut in half with parallel processing compared to using a normal table scan. Because an index is a table, indexes can also be given the parallel structure, so that scans of an index are also faster." Schumacher also reported an example of parallel processing reducing the time of creating an index from seven minutes to five seconds.(Hoffer, Venkataraman, & Topi, 2011)

Designing a Database for optimal Query Performance (continued...)

- Parallel processing not only improves the time of table scans but also can be used on joining tables, grouping query results,sorting, deleting, updating, and insertion.
- Often the designer creating the query has information to better optimize the query that the query module in the DBMS does not. In most relational DBMs the optimizer's plan for processing the query can be known by the designer before actually running the query. This is done through the command EXPLAIN or EXPLAIN PLAIN which display all the information about the optimizer's plans to process the query.
- The query optimizer makes its decision on how to process the query by looking at data from each table such as average row length or the number or rows. You can submit multiple EXPLAIN commands with a query written in different ways to see if the optimizer predicts different performance.
- That allows you to find the best performance and submit that for actual processing. With some DBMs you can force the optimizer to take different steps or use resources other than what the optimizer thinks is the best performance. The clause (/**/) can be used to overriders what the query determines is the best way to process the query.
-



KENNESAW STATE UNIVERSITY

Extended Resources
Description & Links

- This is a video lecture from the University of Washington by Grey Hay, about the physical database design methodology. This video goes into extreme detail about physical database design from the ground up and how the methodology is implemented. https://www.youtube.com/watch?v=S98_8HalY5Q
- This video talks about the oracle database security in a broad approach, mainly focused in Europe. The video discusses important security topics from current database security laws, benefits, history, risks and many more topics. https://www.youtube.com/watch?v=GXF3T4g2tJg
- This video by Kimberly Tripp where she discusses why physical database design matters. Kimberly discusses the importance of good design and also how poor design and can lead to major performance issues. https://www.youtube.com/watch?v=H-jPsp2QlT0
- Lightstone, S., Nadeau, T., & Teorey, T. J. (2007). Physical Database Design : The Database Professional's Guide to Exploiting Indexes, Views, Storage, and More. Morgan Kaufmann.
- Erickson, J., & Siau, K. (2009). Advanced Principles for Improving Database Design, Systems Modeling and Software Development. IGI Global.
- Carmel-Gilfilen, C. (2013). Bridging security and good design: Understanding perceptions of expert and novice shoplifters. Security Journal, 26(1), 80–105. https://doi.org/10.1057/sj.2011.34
- Burtescu, E. (2009). Database Security - Attacks and Control Methods. Journal of Applied Quantitative Methods, 4(4), 449–454.

KENNESAW STATE UNIVERSITY

References

- CloudGirl, & CloudGirl. (2017, March 26). Data Partitioning: Vertical Partitioning, Horizontal Partitioning, and Hybrid Partitioning Project Update #3. Retrieved from http://cloudgirl.tech/data-partitioning-vertical-horizontal-hybrid-partitioning/
- DigitalOcean. (2019, October 28). Understanding Database Sharding. Retrieved from https://www.digitalocean.com/community/tutorials/understanding-database-sharding
- Database VLDB and Partitioning Guide. (2016, July 20). Retrieved from https://docs.oracle.com/database/121/VLDBG/GUID-BE424ACC-F746-4CA8-973C-F578CF98FF10.htm#VLDBG00225
- Lesov, P. (2010). Database Security: A Historical Perspective. ArXiv, abs/1004.4022.
- Quick Base. (n.d.). A Timeline of Database History. Retrieved from https://www.quickbase.com/articles/timeline-of-database-history
- Watt, Adrienne. (n.d.). Chapter 13 Database Development Process. Retrieved from https://opentextbc.ca/dbdesign01/chapter/chapter-13-database-development-process/
- What is Entity Relationship Diagram? (ERD). (2020). Retrieved from https://www.visual-paradigm.com/guide/data-modeling/what-is-entity-relationship-diagram/
- Hoffer, A. J., Venkataraman, R., Topi, H. (2011). *Modern Database Management 10e*[E-Reader Version]. Retrieved from https://www.amazon.com/Modern-Database-Management-Jeffrey-Hoffer/dp/0136088392
- Burleson Consulting. (2014, April 23). Oracle parallel query tips. Retrieved March 28, 2020, from http://www.dba-oracle.com/art_opq.htm
- Chung, C. (n.d.). Introduction to Hashing and its uses. 2BrightSparks. https://www.2brightsparks.com/resources/articles/introduction-to-hashing-and-its-uses.html

KENNESAW STATE UNIVERSITY