Chapter 3. Storage and Retrieval

- On the most fundamental level, a database needs to do two things: when you give it some data, it should store the data, and when you ask it again later, it should give the data back to you.
- Chapter #2 focuses on data models and query languages used by the user. Chapter #3 discuss the same from the database's point of view: how we can store the data that we're given, and how we can find it again when we're asked for it.
- Understanding from a system point of view allows the user to select a storage engine that is appropriate for their application. Some storage engines are optimized for web services applications and others for computationally intensive application.
- Four types:
    - SQL relational databases
    - NoSQL databases
    - log-structured storage engines
    - page-oriented storage engines such as B-trees

Data Structures That Power Your Database

- A Python dictionary can be thought of as a simple key-value store.
- Because values can be objects the values can be almost anything including a NoSQL-type document and because every value has a unique key.
- This type of storage engine uses hash indexing which is extremely efficient over smaller datasets, but it does not work well for very large databases.
- However, key-value indexing is a useful building block for more complex indexes.
- Some issues with this type of storage engine include:
    - Compaction
    - File format
    - Deleting records
    - Crash recovery
    - Partially written records
    - Concurrency control

Some issues with this type of storage engine include:

SSTables and LSM-Trees

- A Sorted String Table (SSTable) is simply a dictionary-type log-structured storage segment of key-value pairs with the additional requirement that the sequence of key-value pairs is sorted by key.
- Maintaining a sorted structure on disk is possible but maintaining it in memory is much easier and faster.
- Common to use well-known tree data structures such as red-black trees or AVL trees.
- Improvement can be made by making LSM-trees out of SSTables
- B-Trees also another very widely used indexing structure

- Other indexing structures include
    - the addition of a secondary index, storing values with the index
    - Multi-column (concatenated) indexes
    - Multi-dimensional – a different type of multi-column index
    - Full-text search and fuzzy indexes
    - Memory resident indexing

Transaction Processing or Analytics?

- Databases are being increasingly used for data analytics.
- Applications that have interactive access pattern are known as online transaction processing (OLTP).
- Applications that need to perform an analytic query that requires a scan over a huge number to calculate aggregate statistics rather than returning the raw data to the user is knows as online analytic processing (OLAP).

OLTP

- Expected to be highly available and to process transactions with low latency
- Data warehousing can be used to separate the OLTP from the physical storage aspect.
- OLTP utilizes row-based store (i.e. CSV file)
- Examples: MySQL, Oracle

OLAP

- Optimized for data crunching
- OLAP is a column-based store.
    - Store all the values the values from each column together (instead of the rows)
    - column-oriented storage relies on each column file containing the rows in the same order
    - improves memory bandwidth and allows for vectorized processing
    - column compression and column sorting capabilities
- Examples: Hbase, Hive, Spark