ARCTICTIGER

Databases and data repositories an introduction

Database Course

What we’ll cover

- Why do we need them?
- What are the different types of databases?
- What an index is and how they work

Why?

Computers are machines designed for processing data.

10011011 10111111

11001011 00111010

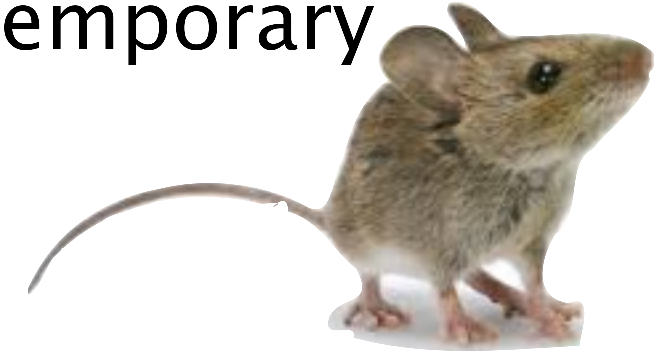
00011011 00011100

01001001 00111110

Data is therefore central to all applications, it is everywhere.

How it is stored is VERY IMPORTANT!

- Some data has a small volume and a short life span and may be suitable to only live in your code as temporary variables.



- Data about where you are in a game for example.
- This data will be lost if the computer runs into problems. Like a powercut.

Other data is large and/or valuable.

It size makes it difficult and expensive to store as in memory variables.

It’s value also makes this a problem.

:-(if the bank lost my pension data because of a power cut.

ARCTIC TIGER

Different types of data

- What you want to do with the data is also important.
- Some applications need very fast read access.
- Others need to be able to write fast.
- Others need to be able to query the data in many different ways.

These different needs have led to a number of different strategies for storing data.

- File system,
- SQL database,
- noSQL database

File system

- Data can be stored as chunks of information in files.
- The data can be retrieved from the file system if you know the path to the file in which data is stored.

File system

Pros

Very fast if you know the file you need, and you always need all the information in a file. I.e. a photo.

Cons

Very limited querying ability.

Always retrieve the entire file, even if you only need a small piece of information within it.

Lots of very small files take up more disk space than they require, leading to waste.

Example of data suitable for file system storage.

Map tiles. Millions of image tiles can be stored on a harddisk.

The file path is all that is needed to locate a specific tile.

Path: \map\1\1.png

Returns the map image for tile in row 1 column 1.

SQL databases

- SQL databases are relational databases.
- They divide your data into one or more tables.

id	firstname	surname	age
1	Tom	Blackmore	37
2	Lotta	Örtstam	33
4	Tina	Freidberg	47

SQL databases

- Each column represents a unique data category.
- Data in one table can have a relationship with data in another table.
  ie. A person described in person table might have a relationship (work for) a company described in a company table, that relationship can be defined and used.

SQL databases

Pros

SQL databases no about relationships between data.

SQL databases are great at adhoc querying.

Give me all people that work at company a, who earn more than x and are older than 30. Is the type of question they are brilliant at answering.

Can retrieve just the data you need.

Relatively good all round performance, a Jack of all trades.

Cons

Queries can be costly on the processor.

Comparatively difficult to scale. Not good on the frontline for popular internet sites.

SQL databases

Example

Info on bank customers, students attending a school etc. etc.

noSQL

- noSQL is an umbrella term that covers all other data storage software.
- It stands for “not only SQL”
- noSQL data storage has become increasingly popular; cloud computing and the internet have pushed it forward.

noSQL

- Since noSQL is such a broad term it is hard to generalise it’s pros and cons.
- In general noSQL databases are good at coping with scale, and working on cloud infrastructure.
- Each target a particular problem area, fast reads “google big table”, easy replication “couchdb”, fast writes “cassandra”, a graph (network) db “neo4j” etc.
- You cherry pick the one that solves your particular data problem best.

noSQL

Pros

Often are very good at scaling.

Are extremely good at a solving a certain data problem.

Cons

Not often generalists,

Harder to define relationships,

Adhoc queries tend to be complex and costly. (Mongodb is the exception)

Physical data storage

RAM Memory

Pros

Very fast access, both reads and writes fly.

Cons

Costly,

All data will be lost when power is removed.

Harddisk

Pros

Cheap.

Safer, disks can be raided together so power loss or physical damage of one disk will not result in data loss.

Cons

Slow, particularly write speeds can be slow.

Indexes

- Indexes are uber important when it comes to data storage and retrieval.
- An index helps your computer find data quickly.

Indexes

An index works by sorting.

Finding a value in a list is much faster if the listed is sorted.

Example:

1, 53, 47, 3, 52, 48, 12, 17, 19.

If I ask if the number 18 is in the list the computer must read the entire list to no that it is not. 9 reads

Indexes

1, 3, 12, 17, 19, 47, 48, 52, 53

If the computer knows the list is sorted. If it starts with the lowest number it knows that if it comes to a number larger than 18 before 18 doesn't exist. 5 reads

Indexes

Binary search (Binary search tree index)

1, 3, 12, 17, 19, 47, 48, 52, 53

This works by always removing half the data set.

The computer looks at 19 first as it’s in the middle of the list.

Since 19 is greater than 18 it knows if it is in the list it must be on the left.

It then selects the number in the middle of items to the left of 19 -> 12.

Since 18 is greater than 12 it knows it must be to the right of 12 but to the left of 19. The number between those two number is 17.

As there are no more numbers between 17 and 19 the program knows 18 doesn’t exist.

(3 Reads)

- If our list contained millions of items, performing a binary search would drastically reduce the number of index reads.

- There are other ways of indexing data such as r–tree and hash maps. But b-tree indexing which is based on the principles of binary search is the most common.

Exercise

- Binary search

Indexes

- Creating and updating indexes are costly.
- How a database handles indexing is key to its properties.
- Updating indexes every time a record is changed makes changes and inserts slow, but queries fast.
- Updating an index on the first query makes queries slow but writes and updates fast.

Indexes

- The number of indexes created on a dataset also effects performance.
- Once the index is created queries fly.
- Maintaining multiple indexes costs processing power when data is changed.

MySQL

- Installation
- http://dev.mysql.com/get/Downloads/MySQL-5.6/mysql-5.6.15-osx10.7-x86_64.dmg
- mysql workbench
- http://dev.mysql.com/downloads/tools/workbench/

- Start server (requires sudo password)