SQL a solid partner

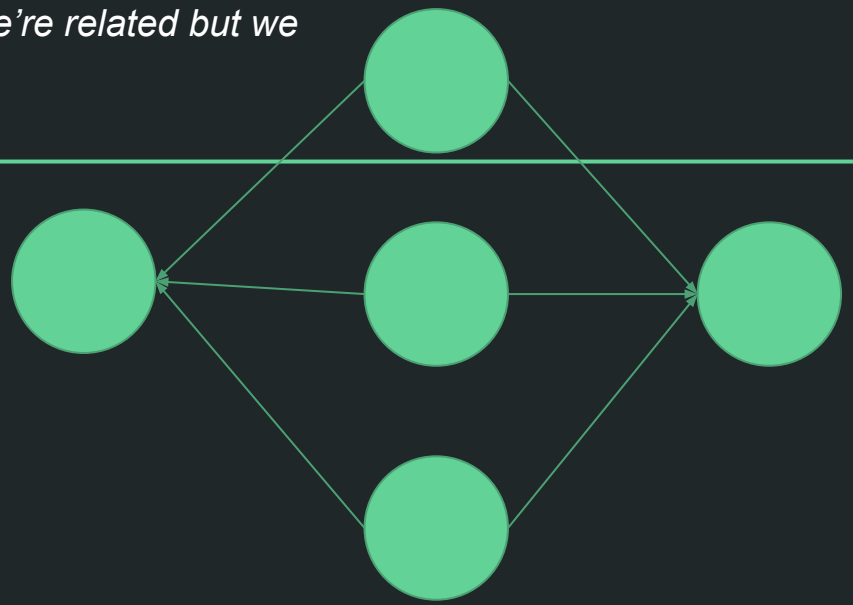
The SQL Table-Based

The usual suspects!

- MySQL
- Postgres
- Oracle
- MSSQL

Age	: 40+
Data-Structure	: Tables
Meaning	: Structured Query Language
Main Focus	: Consistency and Integrity

Well we're like cousins …. we're related but we don't really meet that often



Differences of the Engines

Postgres: (based on ingres)

- pgSQL only has 1 engine
- since 9.3 you can call they support “NoSQL” via JSON
- Since 9.4 there are JSON aggregation methods
- …..
- Best open source SQL database to my knowledge
- A huge amount of data types
- DTL is transactional

MySQL

- MyISAM (No foreign keys, fast reads, fast writes, no transactions, fulltext indexes)
- InnoDB (FK, transactions [DDL])
- TokuDB (Fractal Indexes, for an high amount of inserts)
- Memory-Tables, black hole table
- ……..

Oracle:

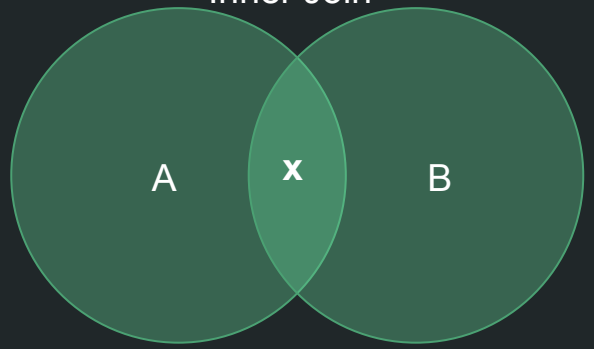
- All the same like MySQL and even more
- CSV tables
- Black Hole tables /dev/null
- Archive tables
- ……..

MSSQL

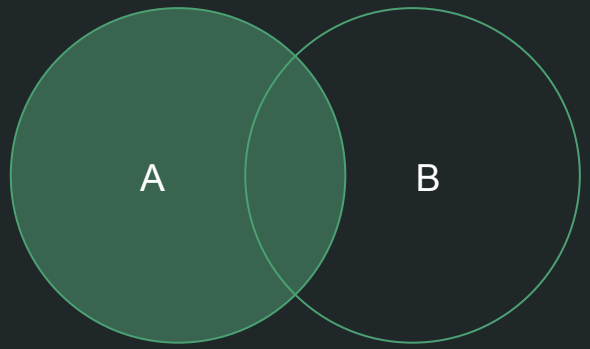
- Sucked in the past
- The new features sounded interesting
- In the end they are the click based solution which obviously can scale for enterprises
- JSON support

Joins the relational power

Inner Join

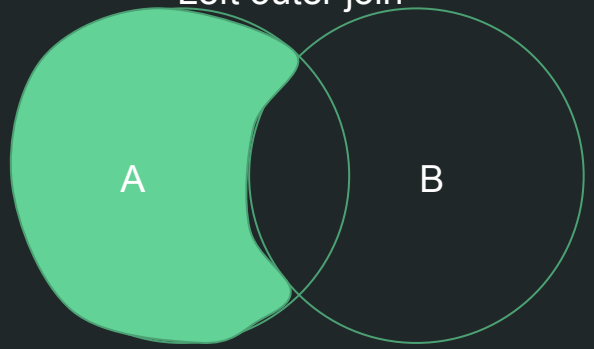


Left inner Join



$A \cap B = \{x: x \in A \text{ and } x \in B\}$ $A \cup (A \cap B) = \{x: x \in A \text{ or } (x \in B \text{ and } x \in A)\}$

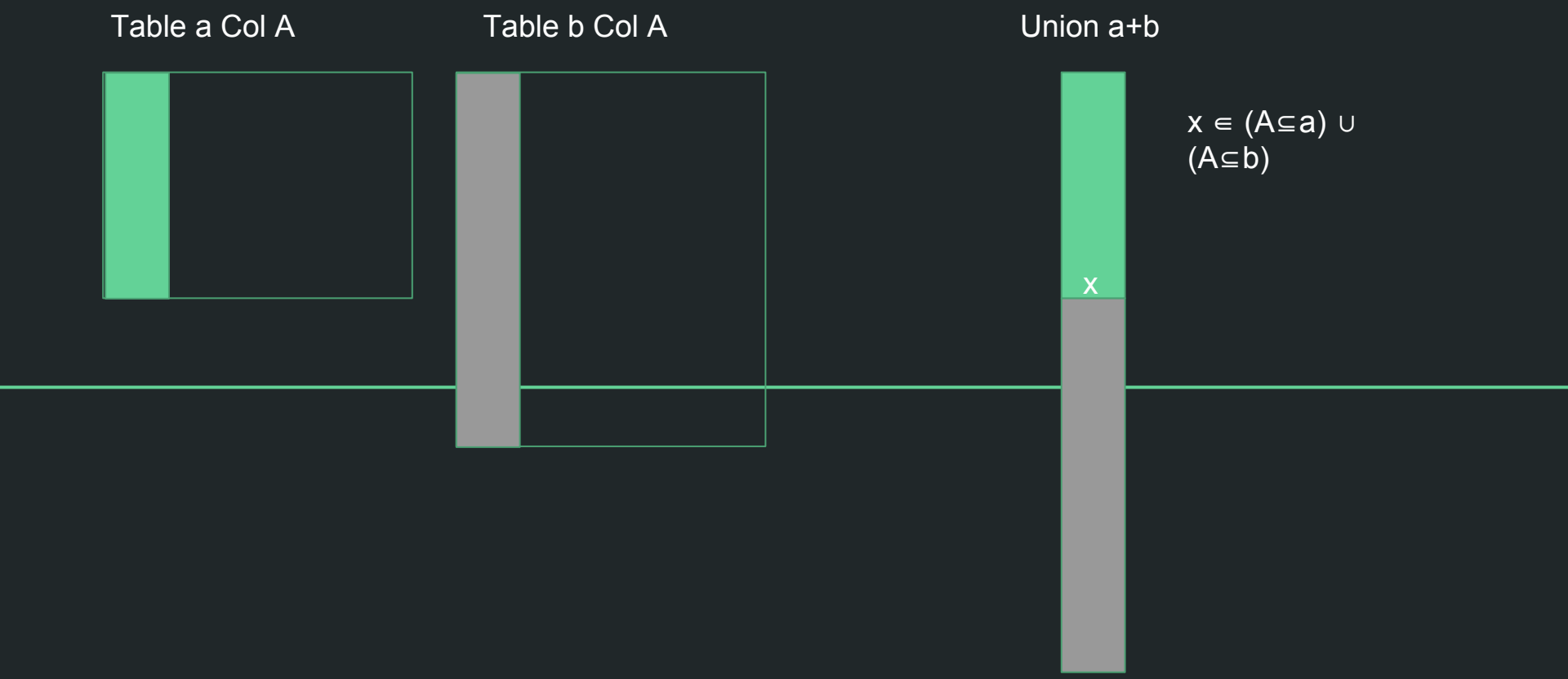
Left outer join



This one i was to lazy to write down ;D

…..

Unions aka let just use columns ...

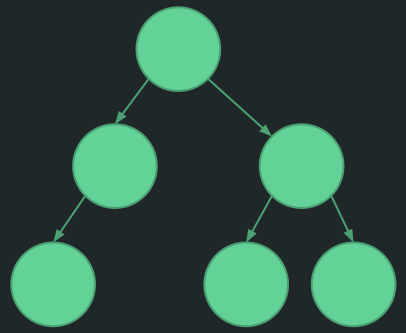


Index Types

Hashmap Index

Constant Time [$\Theta(1)$]

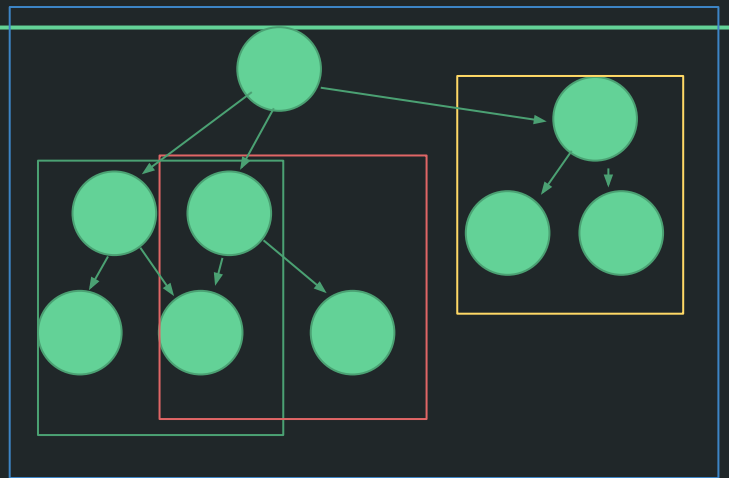
If you know what you're going to access the way to go



Binary Search Tree

Read Average: $\Theta(\log(n))$

If you want to search things based on 2 dimensional vectors this is the way to go >,>=,<=,<



RTree

If you go for multidimensional related data (maps 3d/4d... searches) this is the one

Groupings

SELECT Name,SUM(ALTER) FROM human GROUP BY Name

ID	Name	Alter
1	Karl	46
2	Susi	78
3	Karl	12

Name	Alter
Karl	58
Susi	78

Small examples

Deep VS Flat Tables

Flat table
Pros:
- In a 1:1 comparison of stored data a shorter index, easier filtering and selection

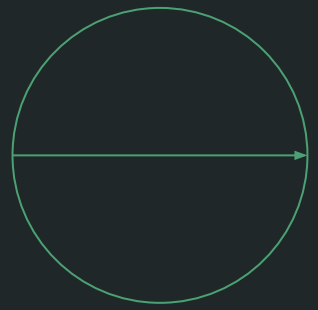
Cons:
- Every new “field” will be a DTL operation
- A maximum amount of columns per table

Deep tables
Pros:
- custom “fields” without a structure change

Cons:
- Harder to be filtered do to possible non existing entries for the same “field”
- The index can really get huge and the time-complexity explodes if you don’t know what you’re doing

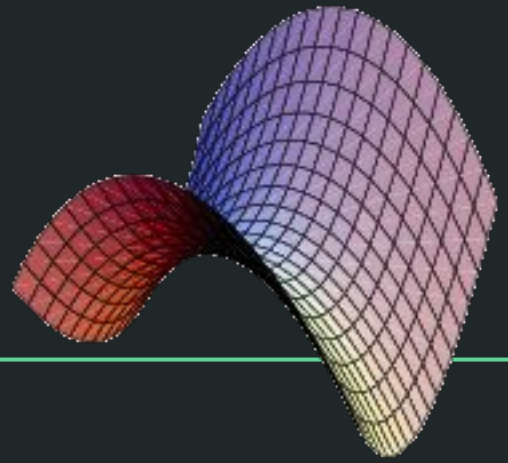
Let’s have some dummy live examples :)

A short example for radial searches (geodb)



Radius Earth: 6371

Hyperbolic to Cartesian System



#	lon	lat
1	16.37	48.209

Degrees to radian measure

lambda = coo.lon * π / 180;
phi = coo.lat * π / 180;

0,285710399 = 16.37 * 3,14.. / 180
0,841388326 = 48,209 * 3,14.. /180

SELECT coo.lon, coo.lat FROM geodb_coordinates AS coo INNER JOIN geodb_textdata AS textdata ON textdata.loc_id = coo.loc_id WHERE textdata.text_val = "1110" AND textdata.text_type = "500300000"