INF1343, Winter 2011

Data Modeling and Database Design

Yuri Takhteyev
University of Toronto

Week 11

Storage, Structure, and Performance

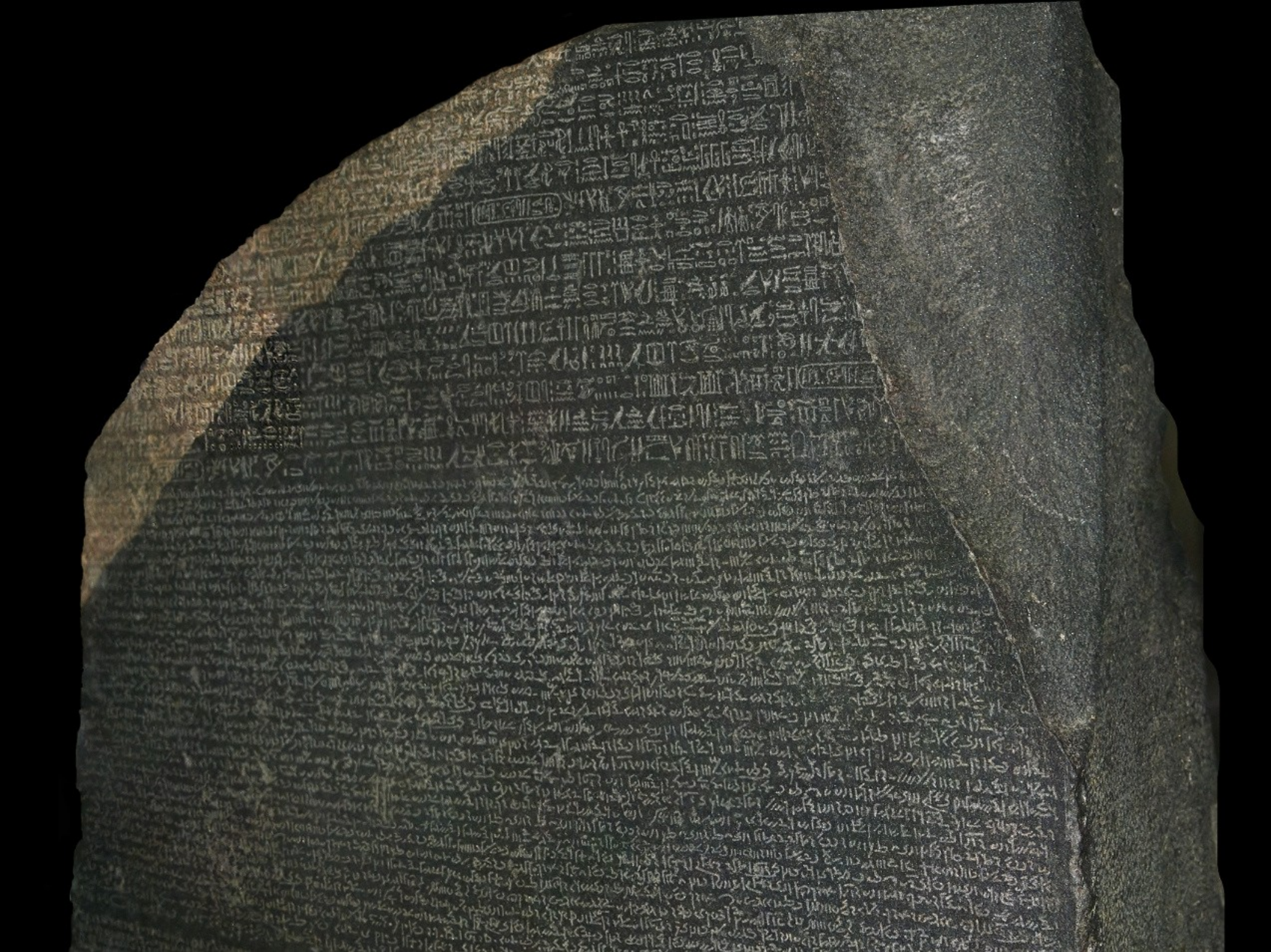
Storage

Persistent

Easy to read

Easy to update

Cost-effective



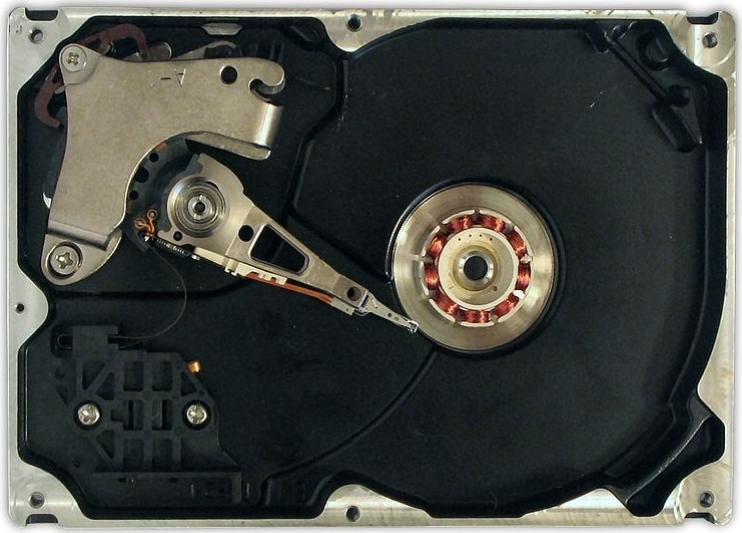
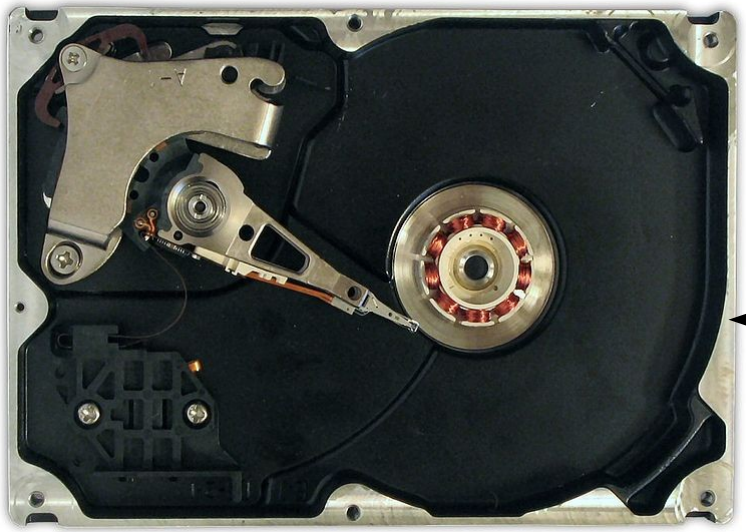
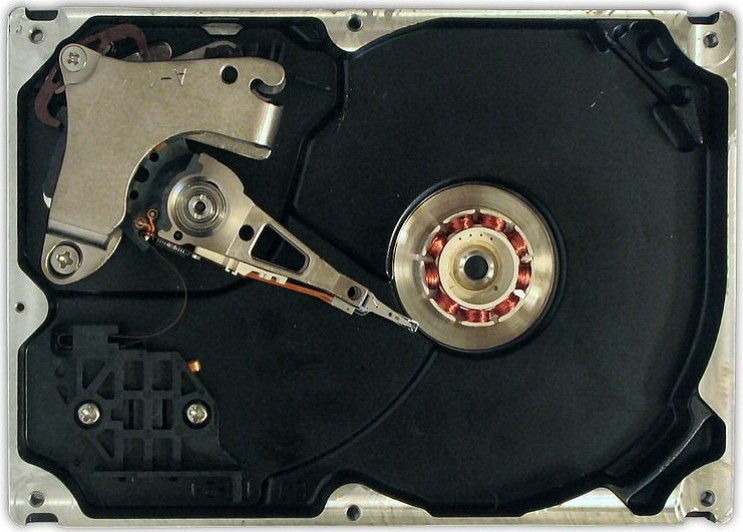


image sources
http://en.wikipedia.org/wiki/File:Hard_disk_dismantled.jpg
http://en.wikipedia.org/wiki/File:Memory_module_DDRAM_20-03-2006.jpg
http://en.wikipedia.org/wiki/File:Largetape.jpg




elixir
M2U51264DS8HC1G-5T
512MB DDR-400MHz/CL3
PC3200U-30331
Warranty Void If Removed
0542.MN05A110TA.5YM0410N08.XX.CN

caching



"hierarchical storage management"

Structure

Finding a “storable” representation

- Not losing information
- Allowing for easy retrieval
- Allowing for easy update
- Minimizing storage size

CSV

simple!
relatively small
but what to do with NULLs?
and how would it perform?

CSV

657807938,559562982,23.47
755280276,889208095,590.32
934625720,459538801,44.66
852067113,660539228,1684.77
+ another billion rows

CSV

657807938,559562982,23.47↵
755280276,889208095,590.32↵
934625720,459538801,44.66↵
852067113,660539228,1684.77↵
+ another billion rows

CSV

657807938,559562982,23.47⏎755280276,889208095,590.32⏎934625720,459538801,44.66⏎852067113,660539228,1684.77⏎.....

Fixed-Length

65780793855956298 2002347
75528027688920809 5059032
93462572045953880 1004466
85206711366053922 8168477
+ another billion rows

start of Nth row = N * length of row

Fixed-Length

65780793855956298200234 7
75528027688920809505903 2
93462572045953880100446 6
85206711366053922816847 7
+ another billion rows

How do we **add** a row?
How do we **delete** a row?
How do we **find** a row?

Sort it?

755280270	029178799	023934
755280276	889208095	059032
755280279	900892340	00020
755290123	939191721	000129

+ another billion rows

Finding is easier!

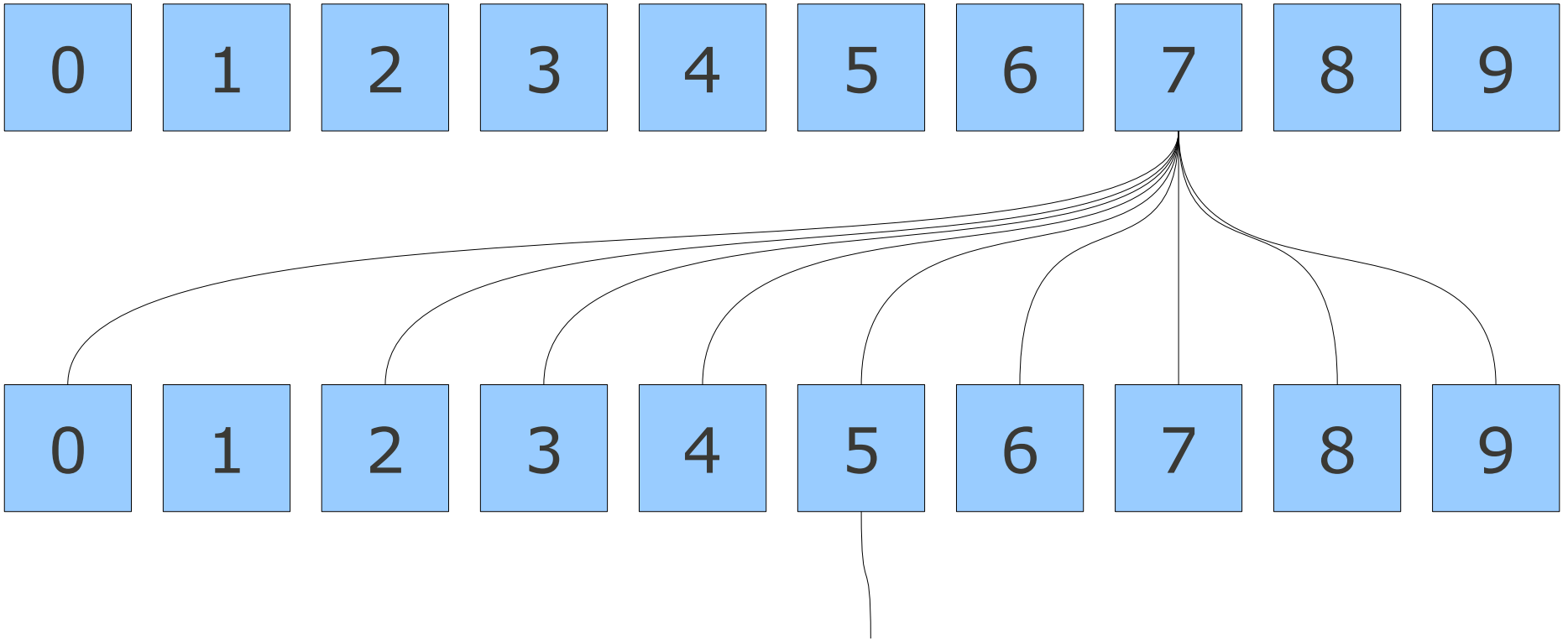
Inserting is **harder**!

And the second field?

75528027002917879902393 4
75528027688920809505903 2
75528027900892340000020
75529012393919172100012 9
+ another billion rows

Sort by the 2nd field?
But avoid duplication!
(Essentially an index.)

Trees



755280276,889208095,590.32

and other items that start with 75

(does not need to be ordered)

Fixed vs Balanced

Fixed:

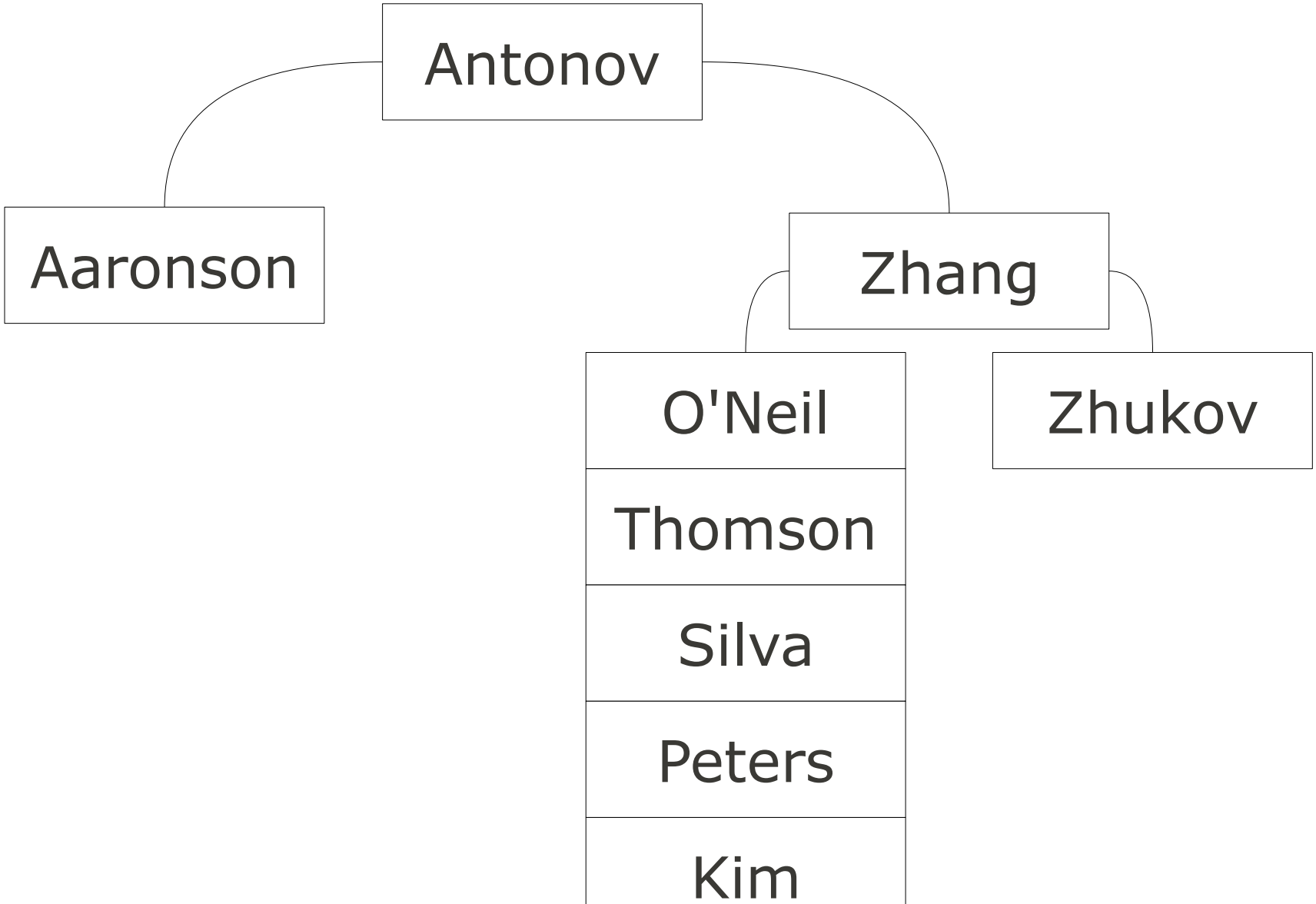
E.g., ISAM

Balanced:

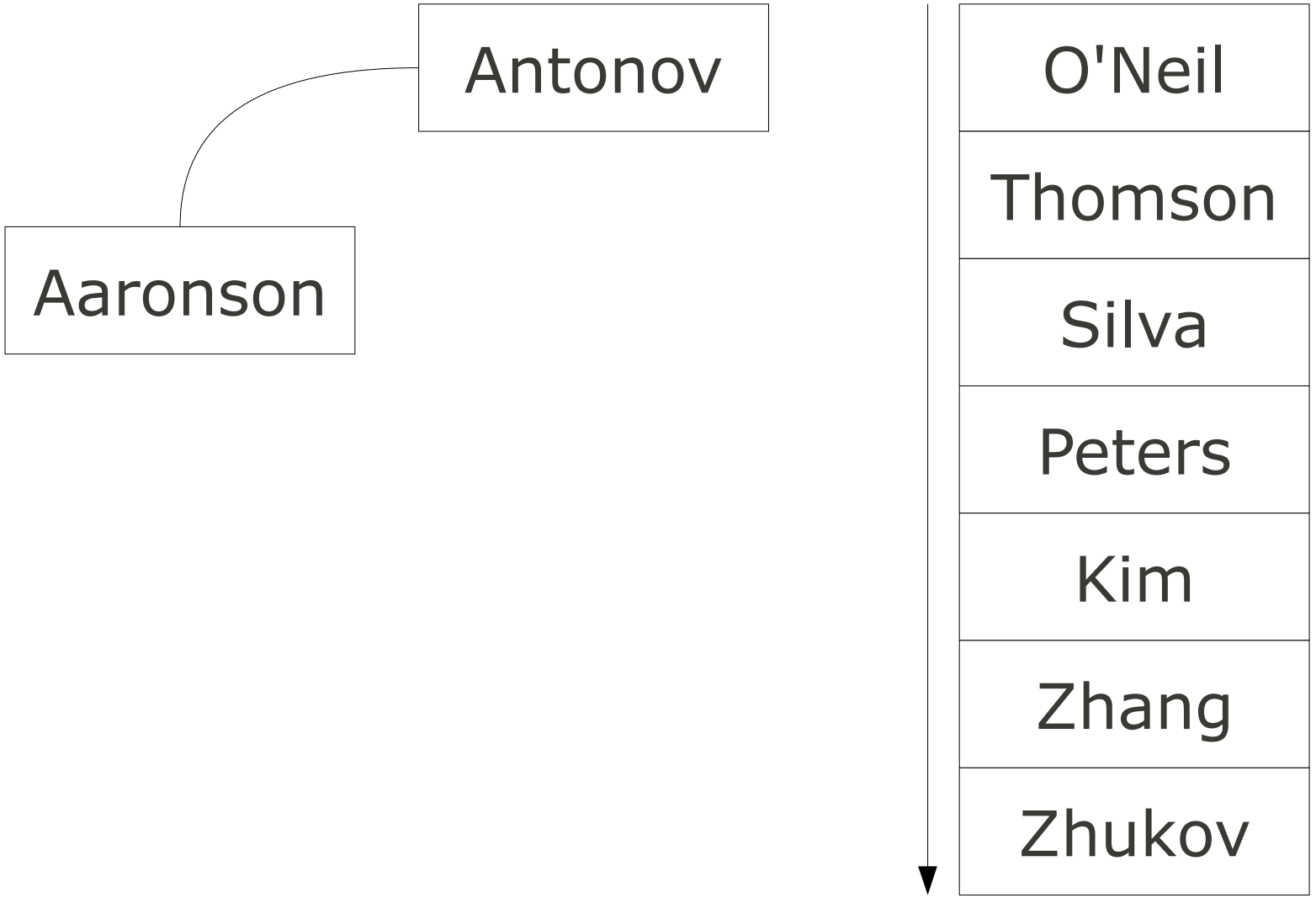
E.g., B+ Trees

A: 4%
M: 9.5%
Q: 0

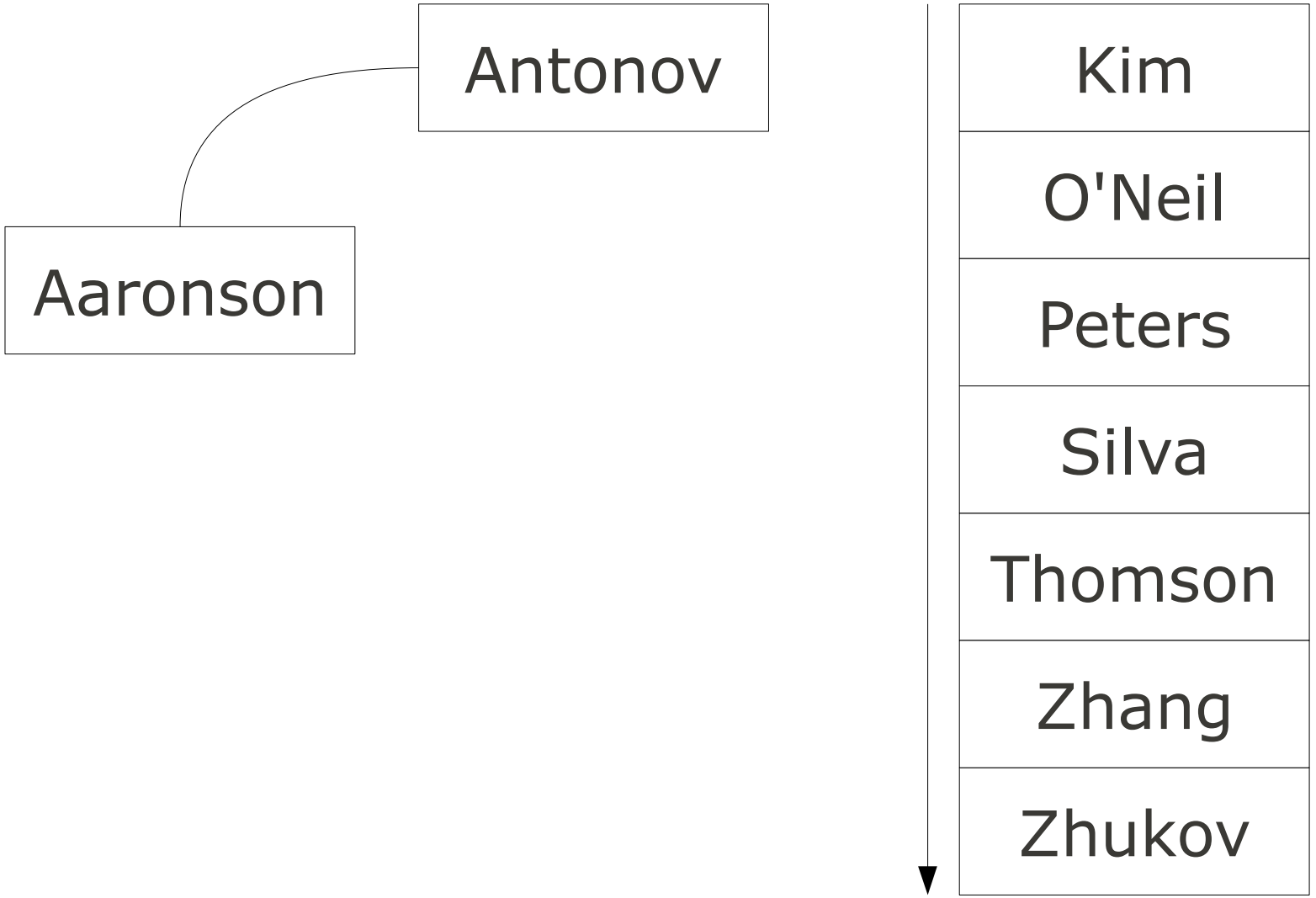
Balanced Tree



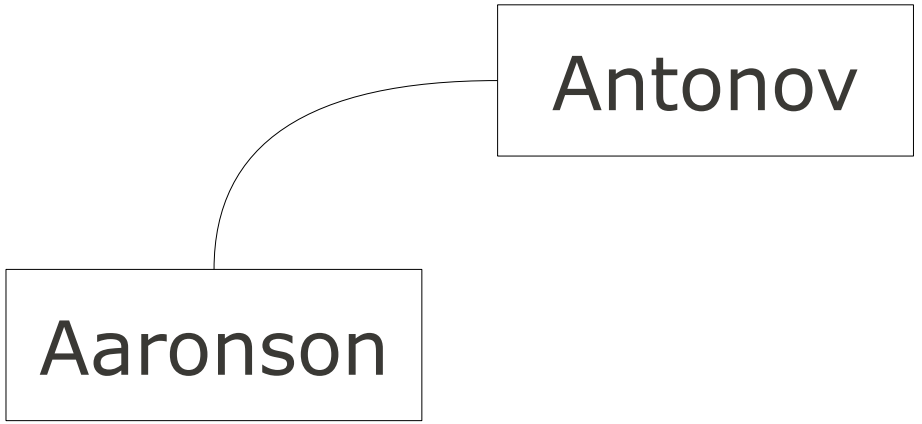
Balanced Tree



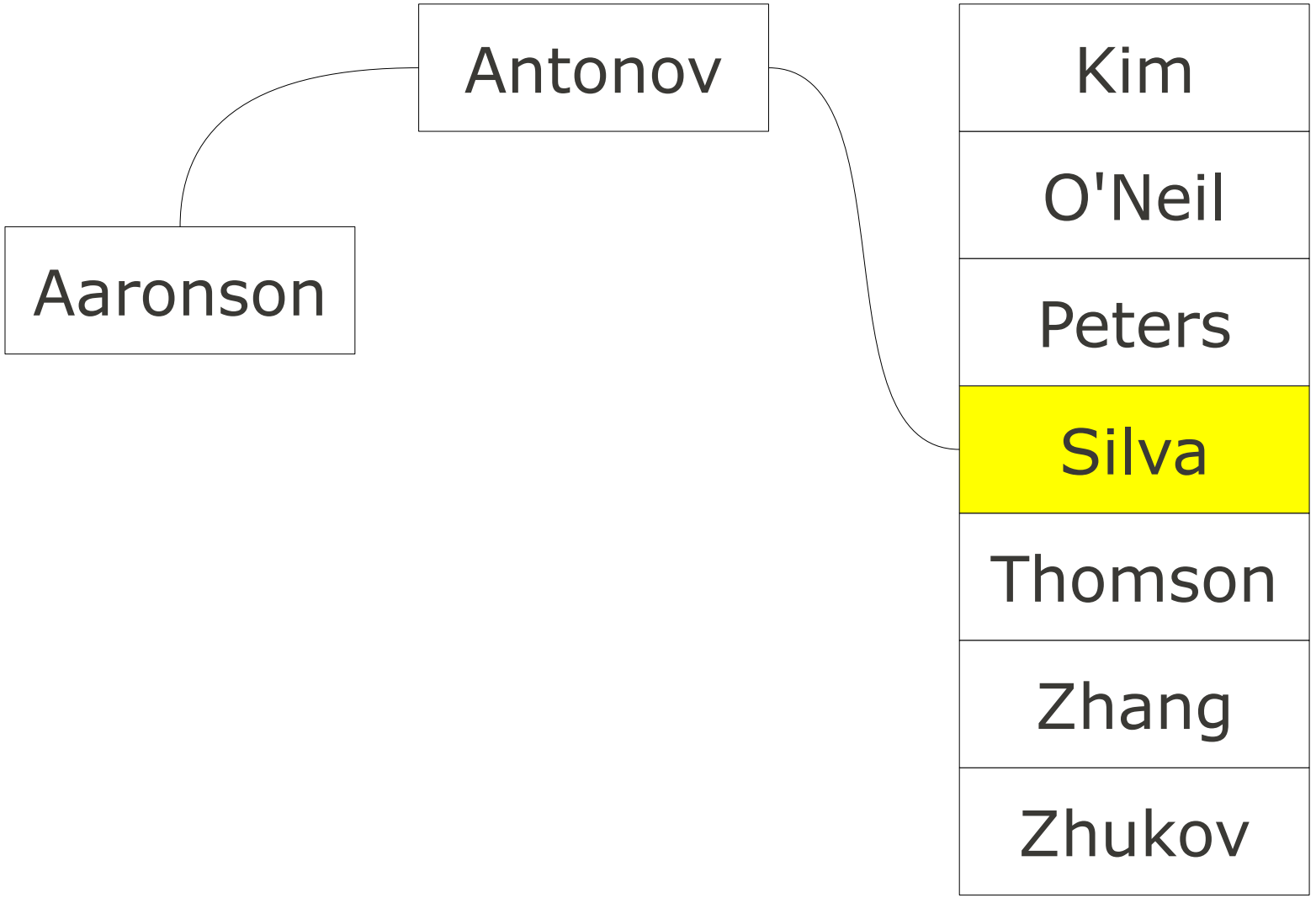
Balanced Tree



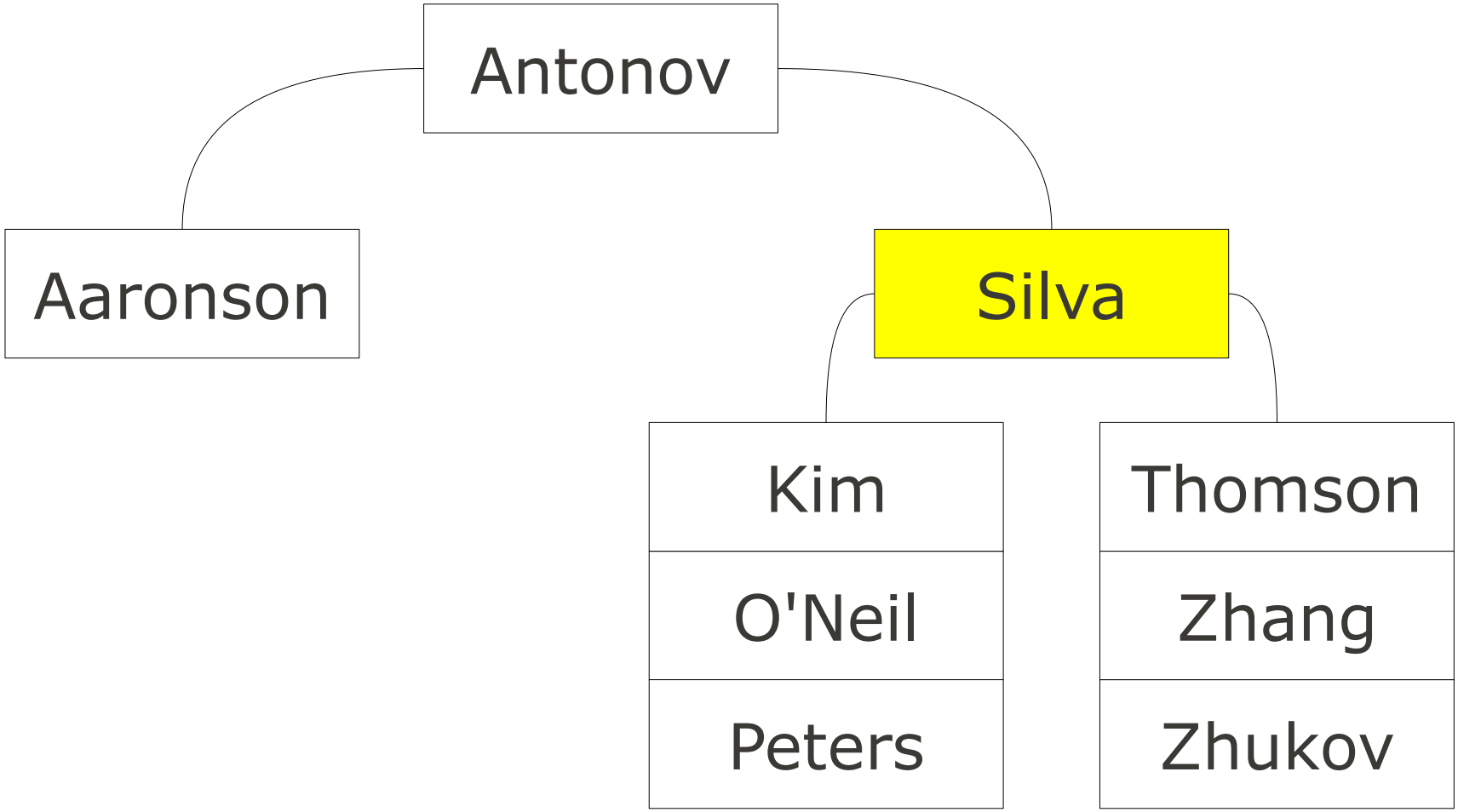
Balanced Tree



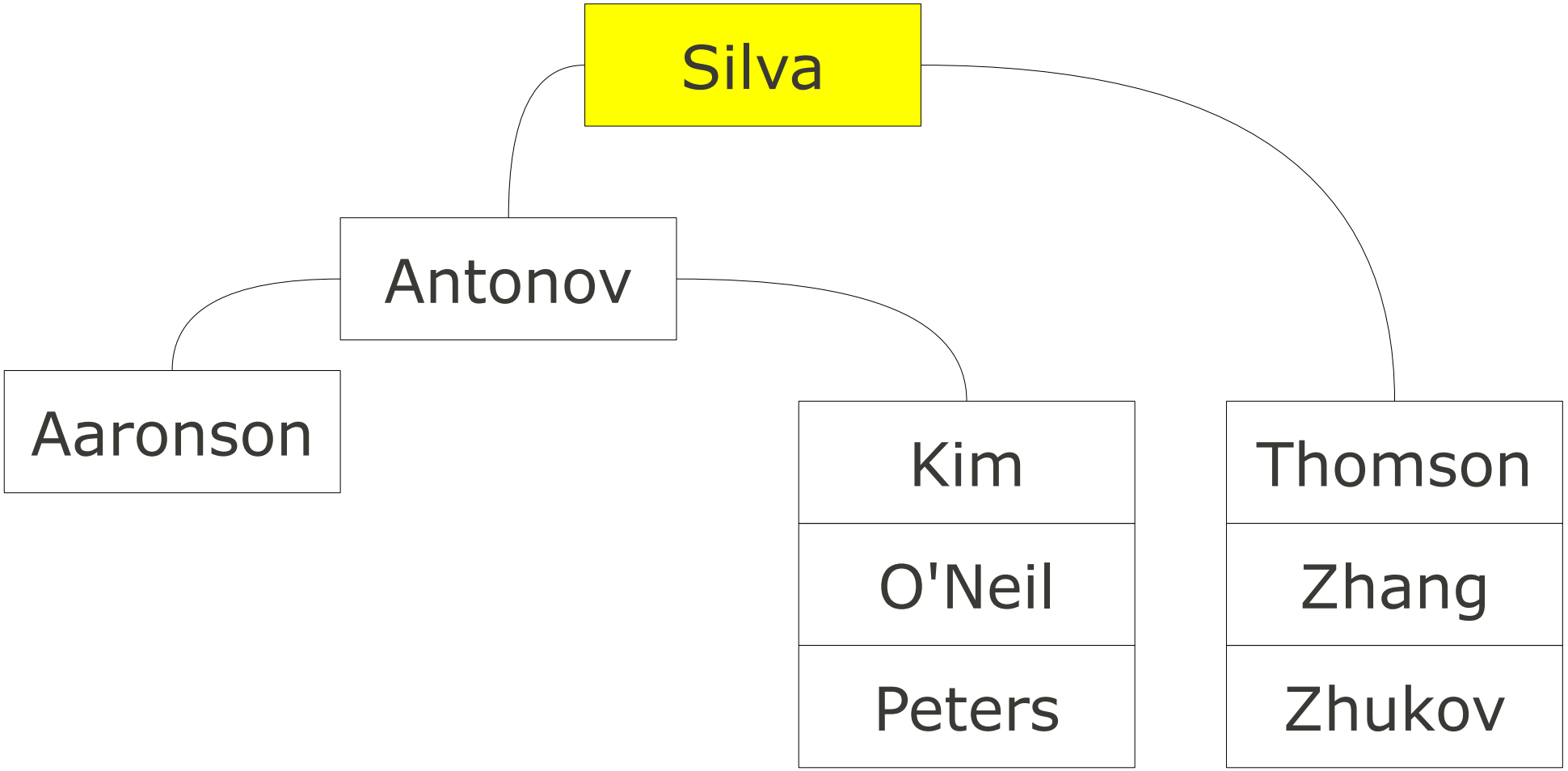
Balanced Tree



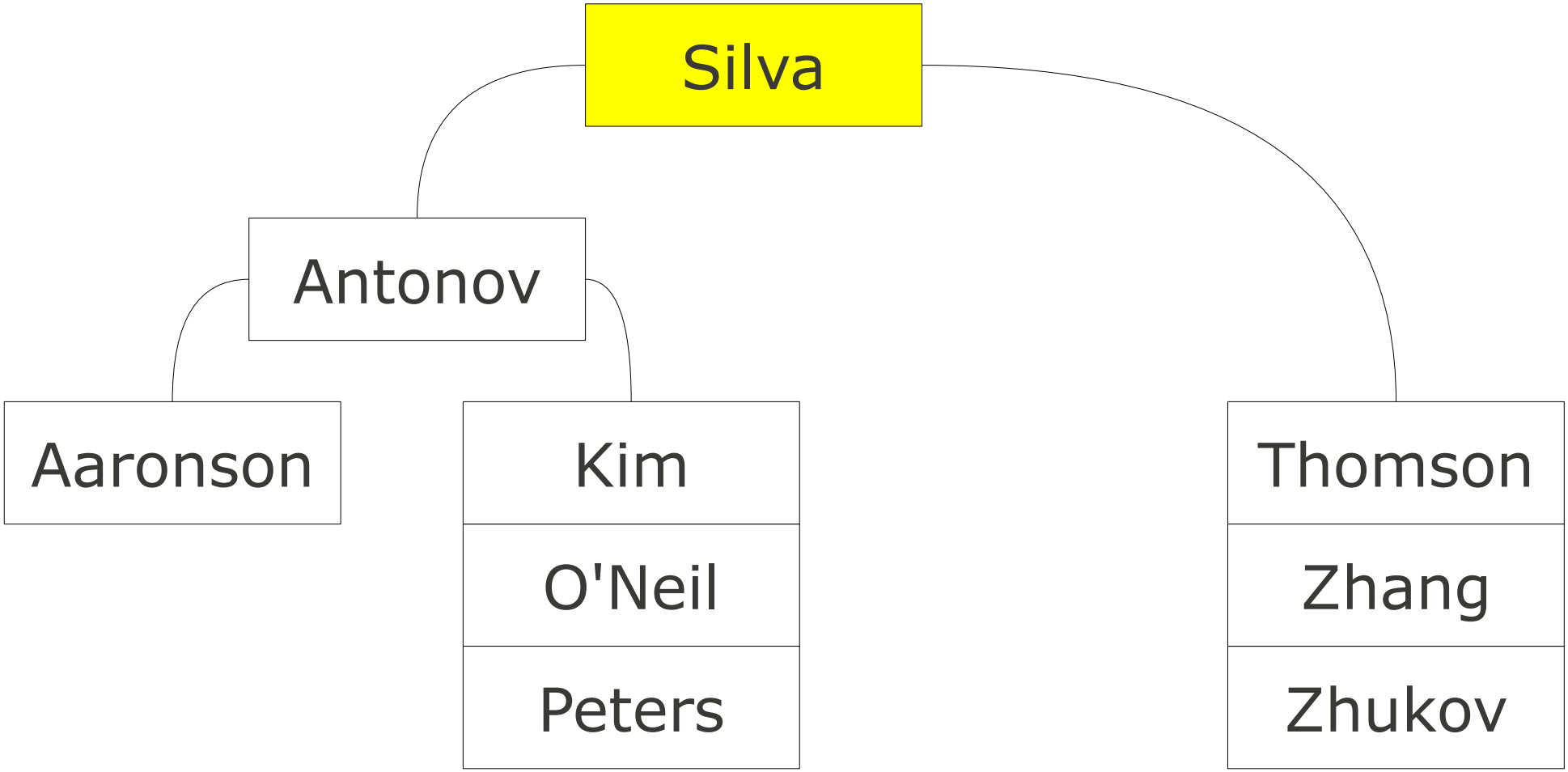
Balanced Tree



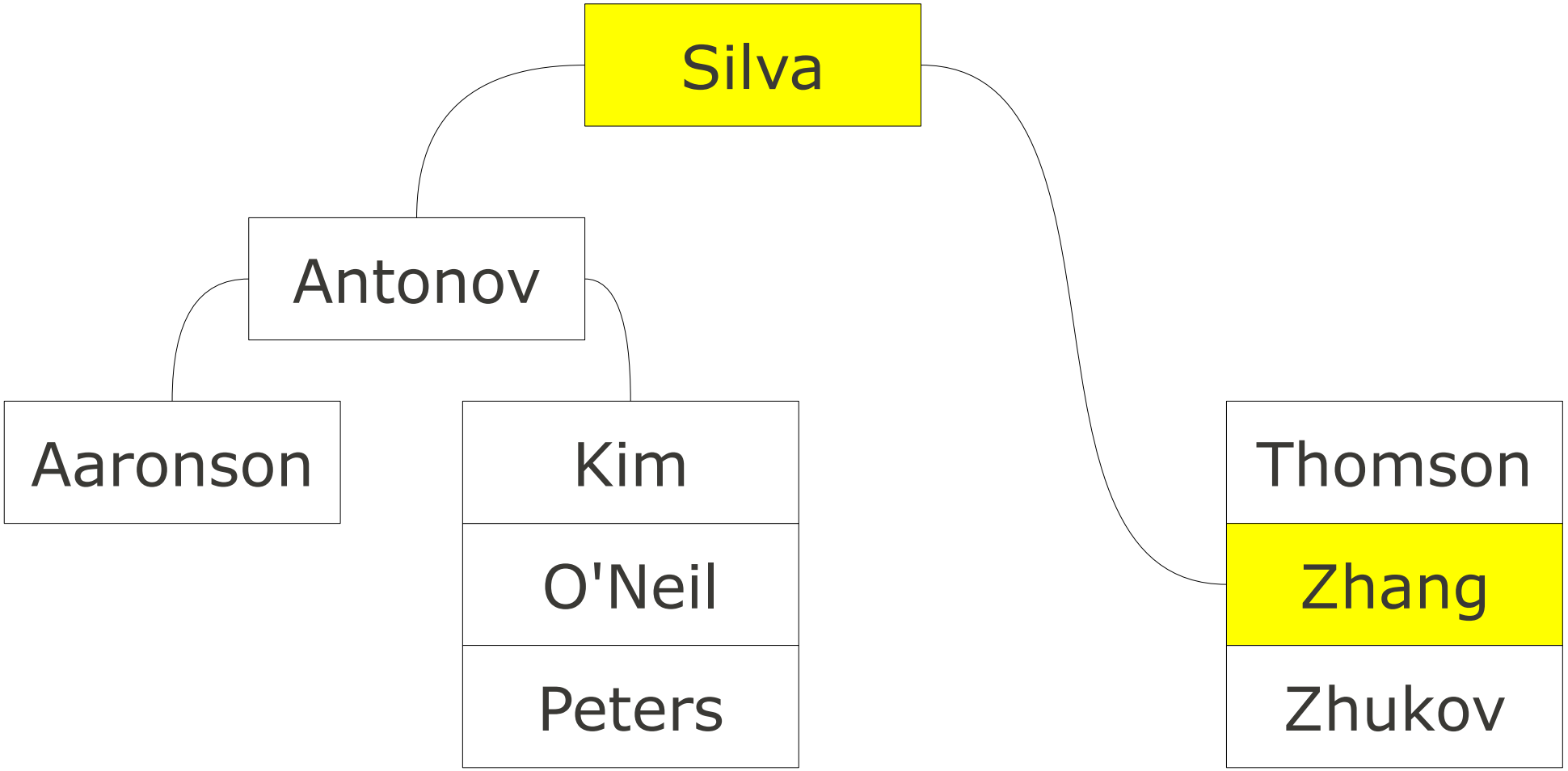
Balanced Tree



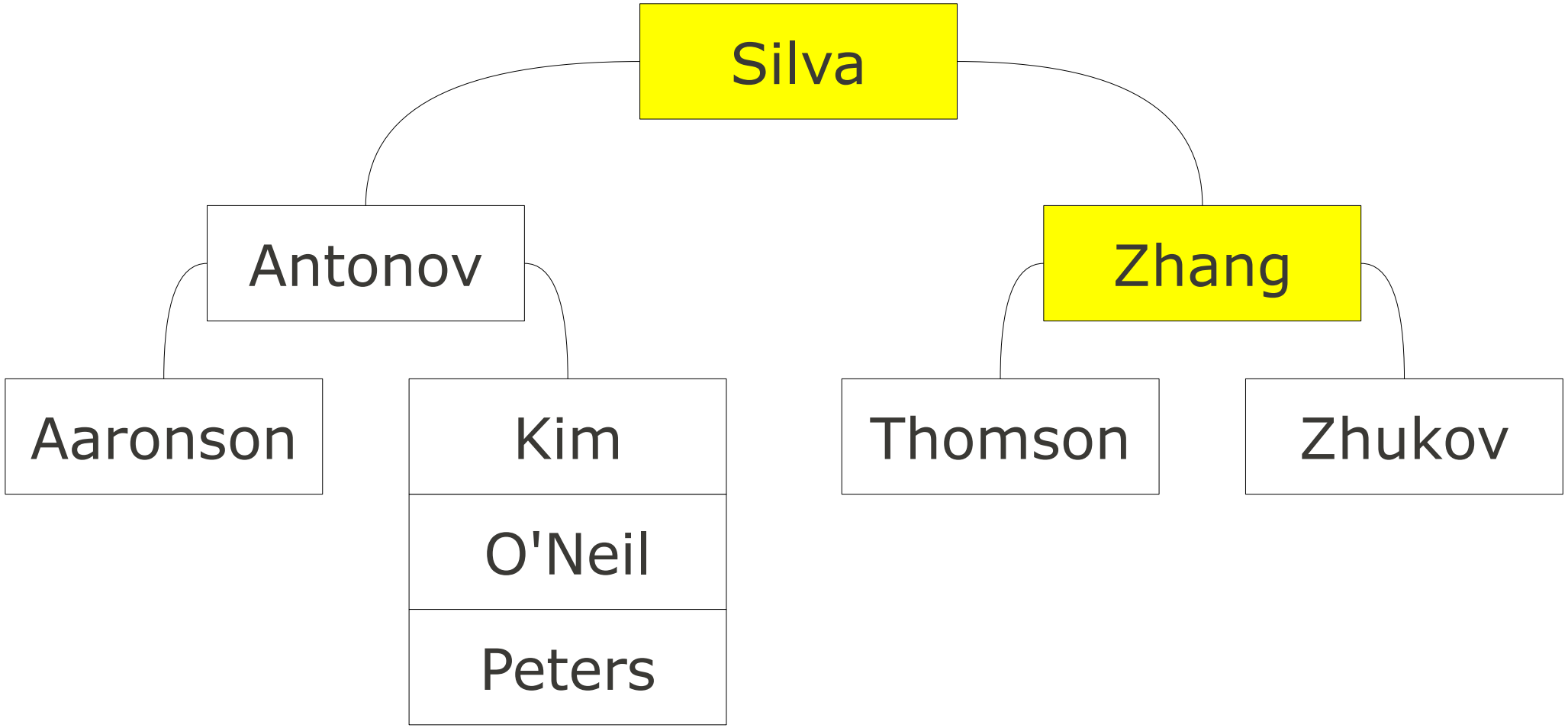
Balanced Tree



Balanced Tree



Balanced Tree



Storage Engines

CVS

yes, it's an option

MyISAM

the default

relatively simple

See: /var/lib/mysql/menagerie/

InnoDB

more features

The Extra Features

Foreign Key Constraints

not in MyISAM!

Transactions

```
start transaction;
update table1 ...;
update table2 ...;
commit;
```

Downside: complexity

More Engines

Memory

no harddrive → faster

Blackhole

doesn't actually store anything

Federated

allows remote tables

Etc.

Picking an Engine

```
create table superhero (
  id integer,
  primary key (id),
  name char(100)
) engine=InnoDB;
```

Creating an Index

Easy retrieval by field

- usually automatic for PK
- optional for other fields
- downsides:
  - additional space
  - harder inserts

```
create index name_index
on superhero(name);
```

Using EXPLAIN

```
explain select * from t
join t2 on t.id2=t2.id2
where id1=100;
```

Backup

mysqldump

```
mysqldump –u kenobio1 –p starwars > starwars.sql
mysql < starwars.sql
```

(Add “create database starwars; use starwars;”)

hot backup

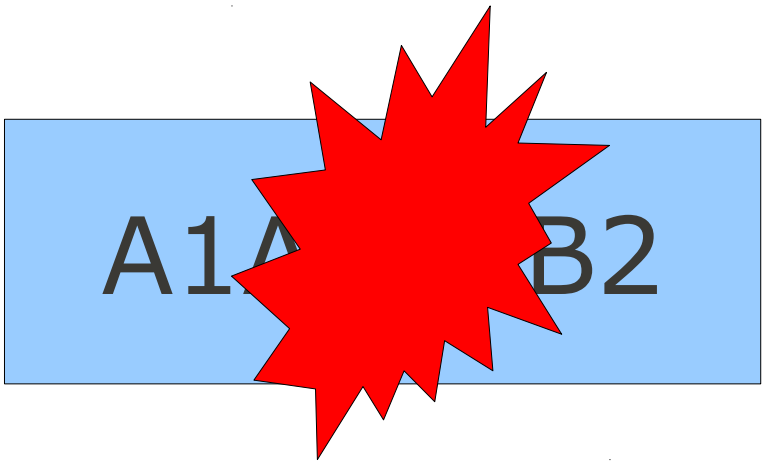
replication

off-site copies

RAID

combining disks for reliability and/or performance

No RAID



RAID-0

can be faster, no gain in reliability

RAID-1

faster reading, slower writing
can withstand losing one disk

RAID >2

RAID >2

reasonable reliability,
not too wasteful

A Web Workshop

cgi3.zip + instructions

- see the course website
- get it working first
- caveat: permissions

Client: “GET form.html”

Server: HTML with <form>

Client: “GET personas.cgi?...”

generated by the browser
based on the form

Server: HTML with the data

generated based on an SQL query
customized by the parameters

personas.cgi?**species**=**Human**&**gender**=**F**

a form object

```
form = cgi.FieldStorage()
if form.has_key("species") :
    print form["species"].value
```

variables

```
species = form["species"].value
species = db.escape_string(species)
```

an sql query

```
query = template % (species, gender)
```

data

```
cursor.execute(query)
if cursor.rowcount() > 0 :
        rows = cursor.fetchall()
```

The Actual Workshop