Index SQL

Born to B-Tree

Benoit Viguier

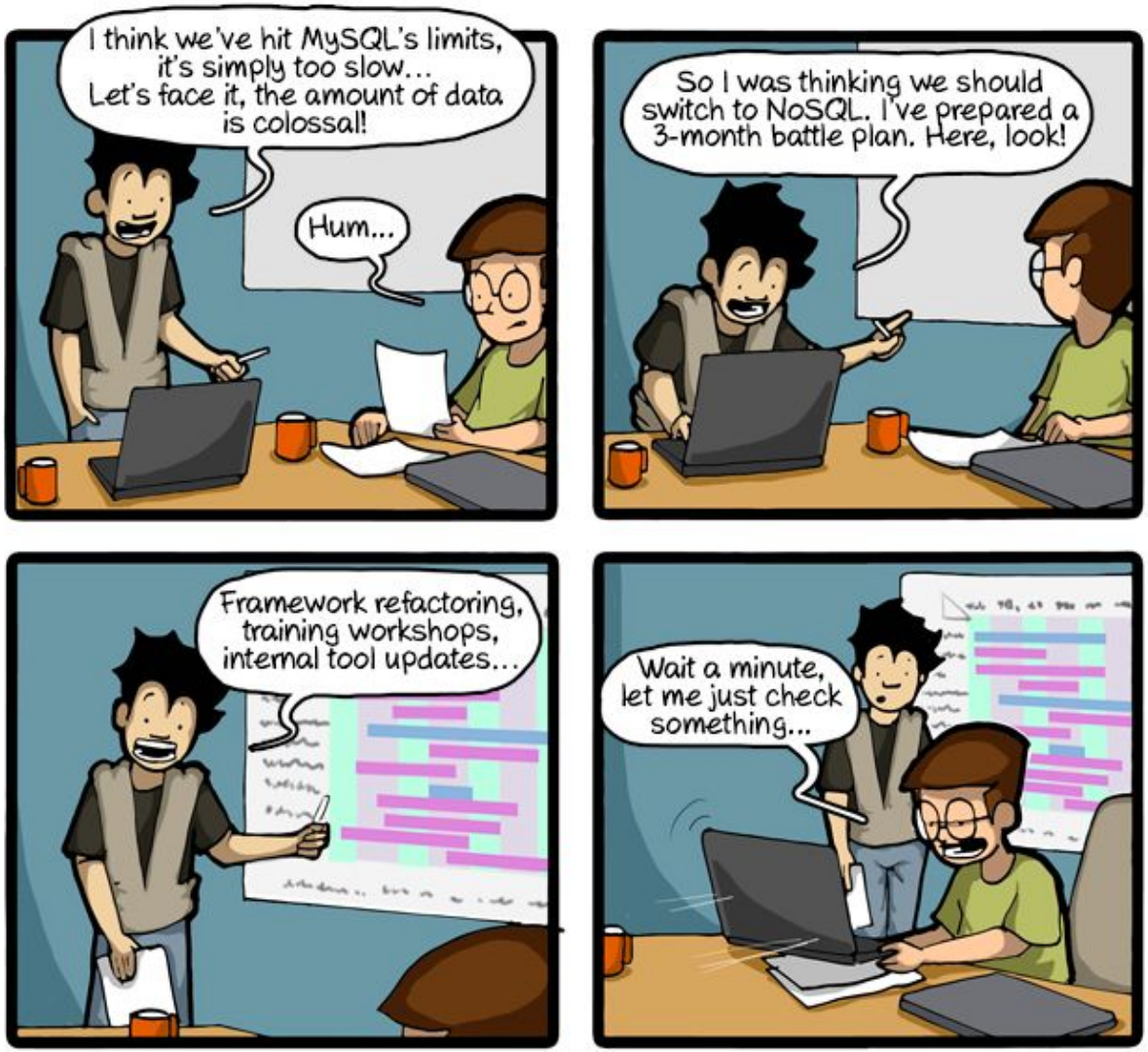
27/11/2015 LFT M6Web

WHY !?

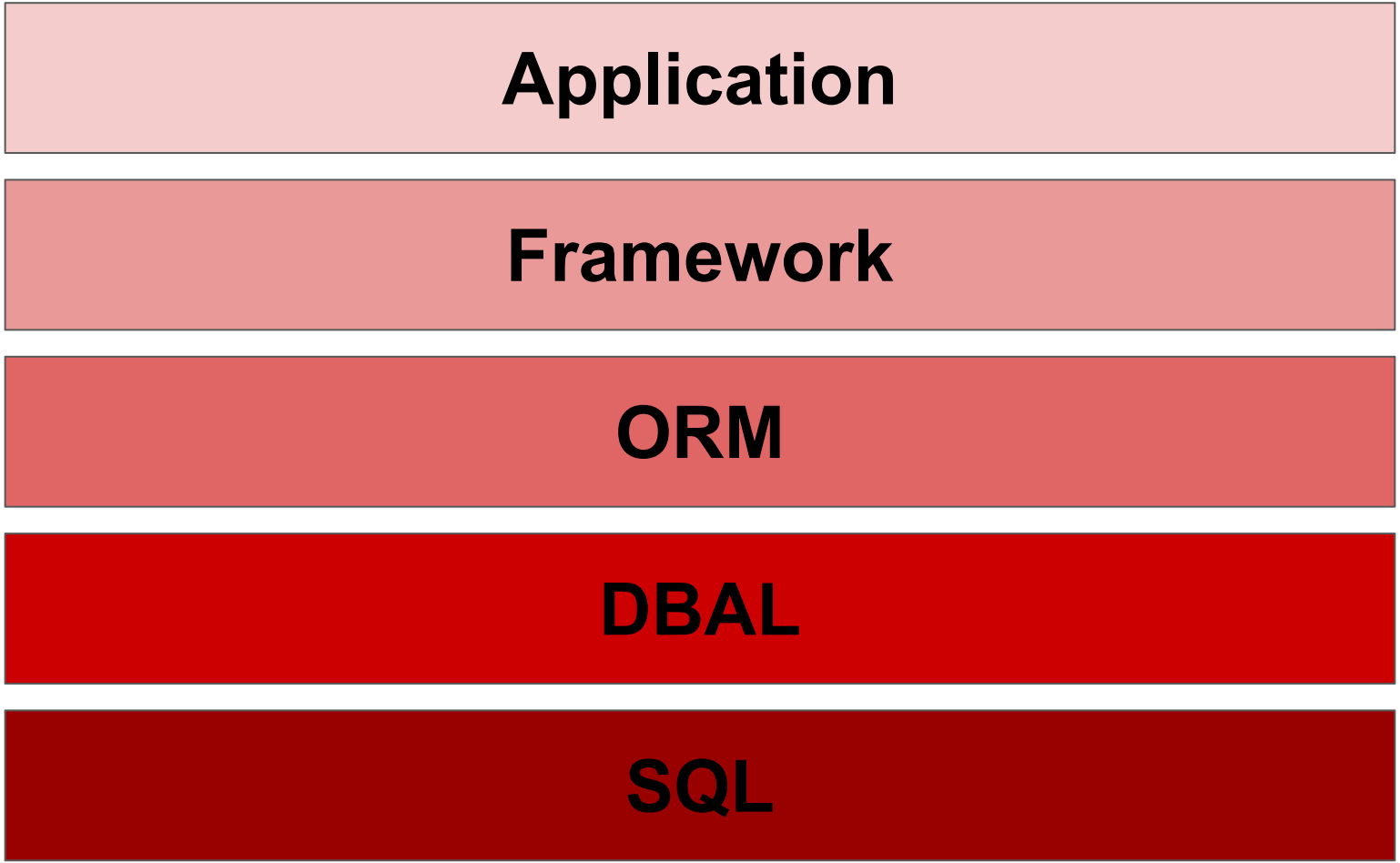

BUT YOU'RE SO
OLD?????????
memegenerator.net

History...

	SQL	MySQL
Creation	1986 (1974)	1995
Last Version	2011 http://modern-sql.com	2015 (5.7)


I think we've hit MySQL's limits, it's simply too slow... Let's face it, the amount of data is colossal!
Hum...
So I was thinking we should switch to NoSQL. I've prepared a 3-month battle plan. Here, look!
Framework refactoring, training workshops, internal tool updates...
Wait a minute, let me just check something...


Just as I thought...
Here you go. It's called an index, and it just made your queries 20 times faster...
Dude, missing indexes? Seriously...
CommitStrip.com

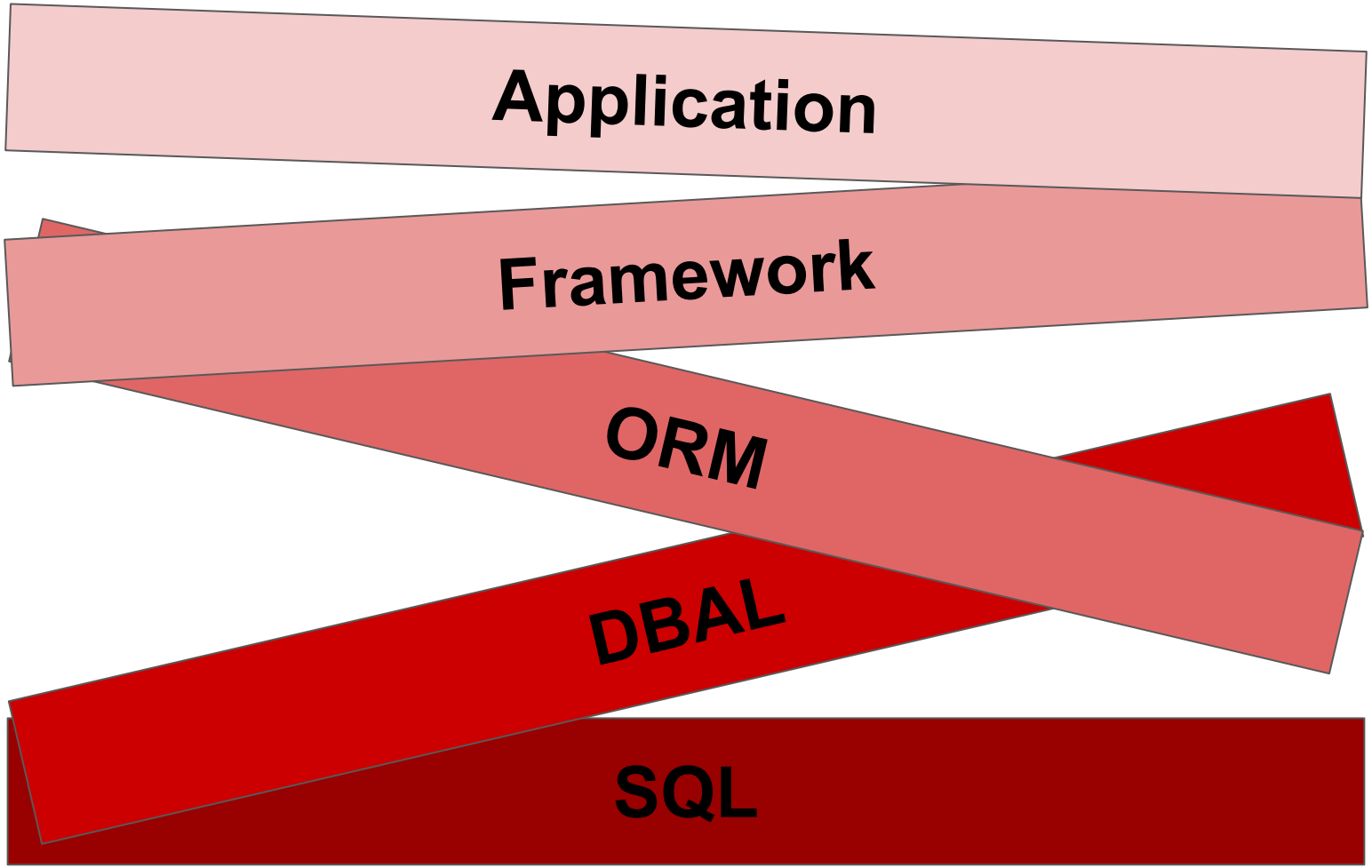

Application
Framework
ORM
DBAL
SQL

Law of Leaky Abstraction

“All non-trivial abstractions, to some degree, are leaky.”

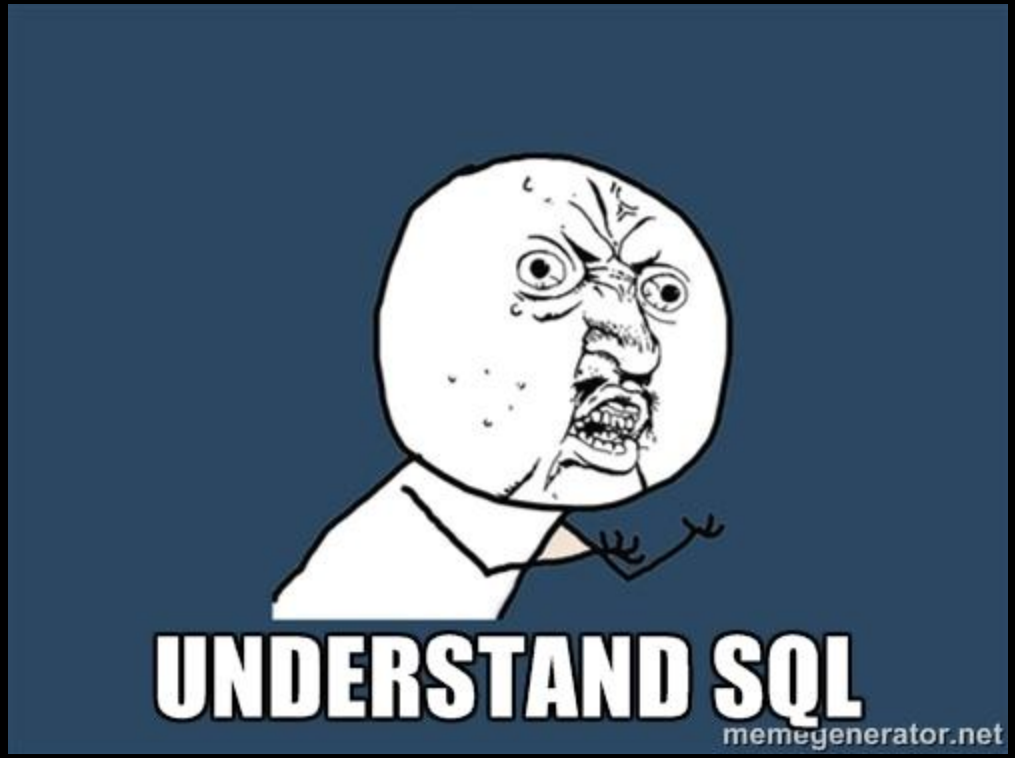
http://www.joelonsoftware.com/articles/LeakyAbstractions.html

J. Spolsky, 11/11/2002


Application
Framework
ORM
DBAL
SQL

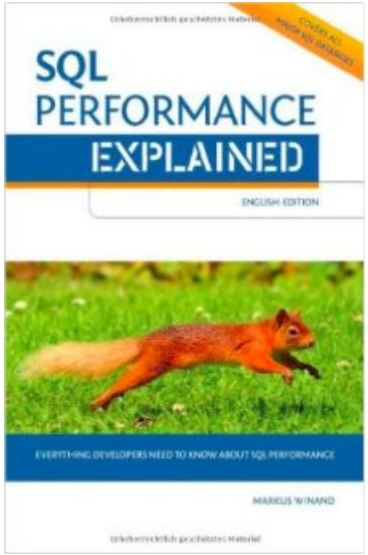
YOU'RE DOING IT WRONG.

No matter how hard you try, it is impossible to fax a cat.


UNDERSTAND SQL
memegenerator.net

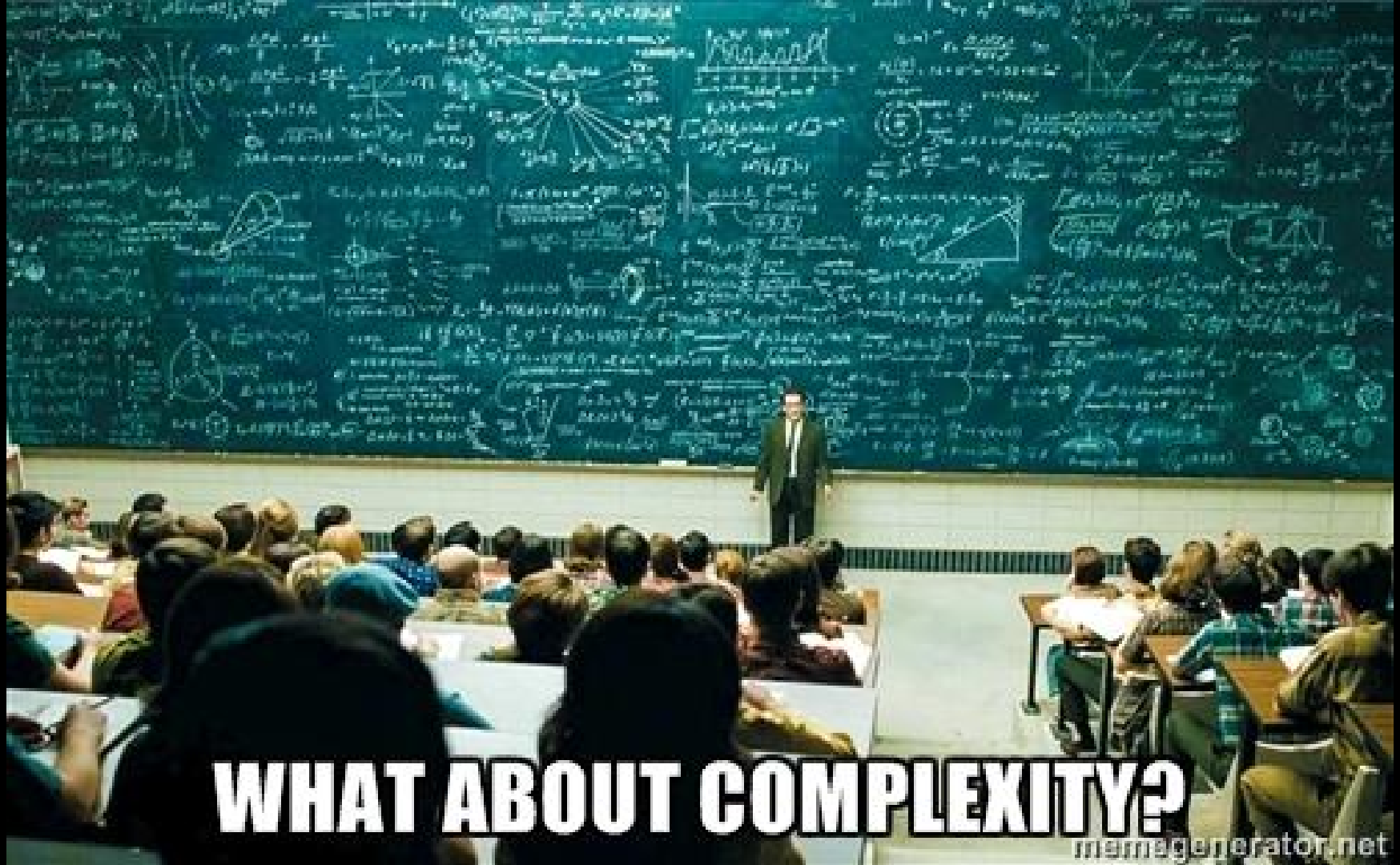
USE THE INDEX, LUKE

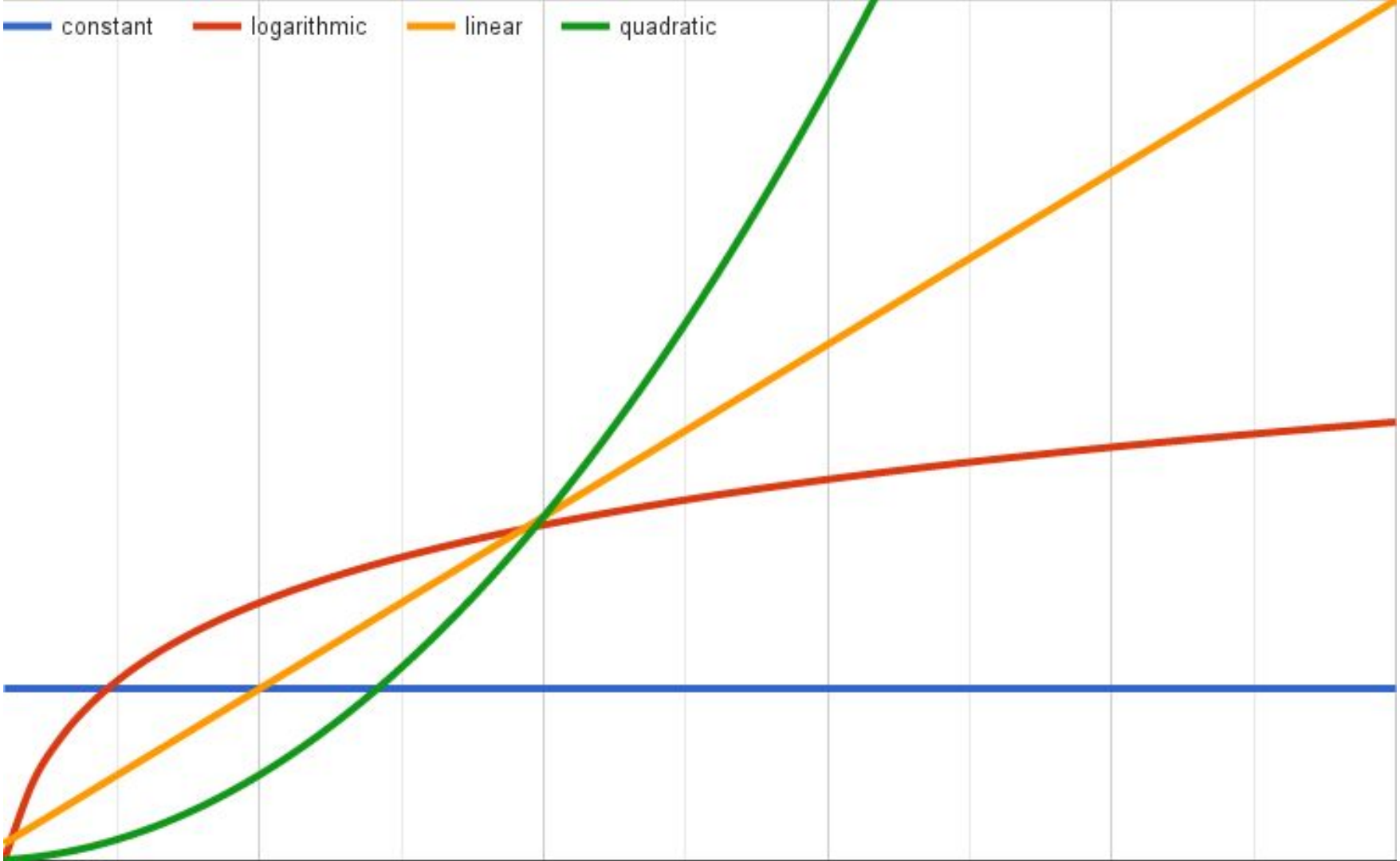
A guide to database performance for developers

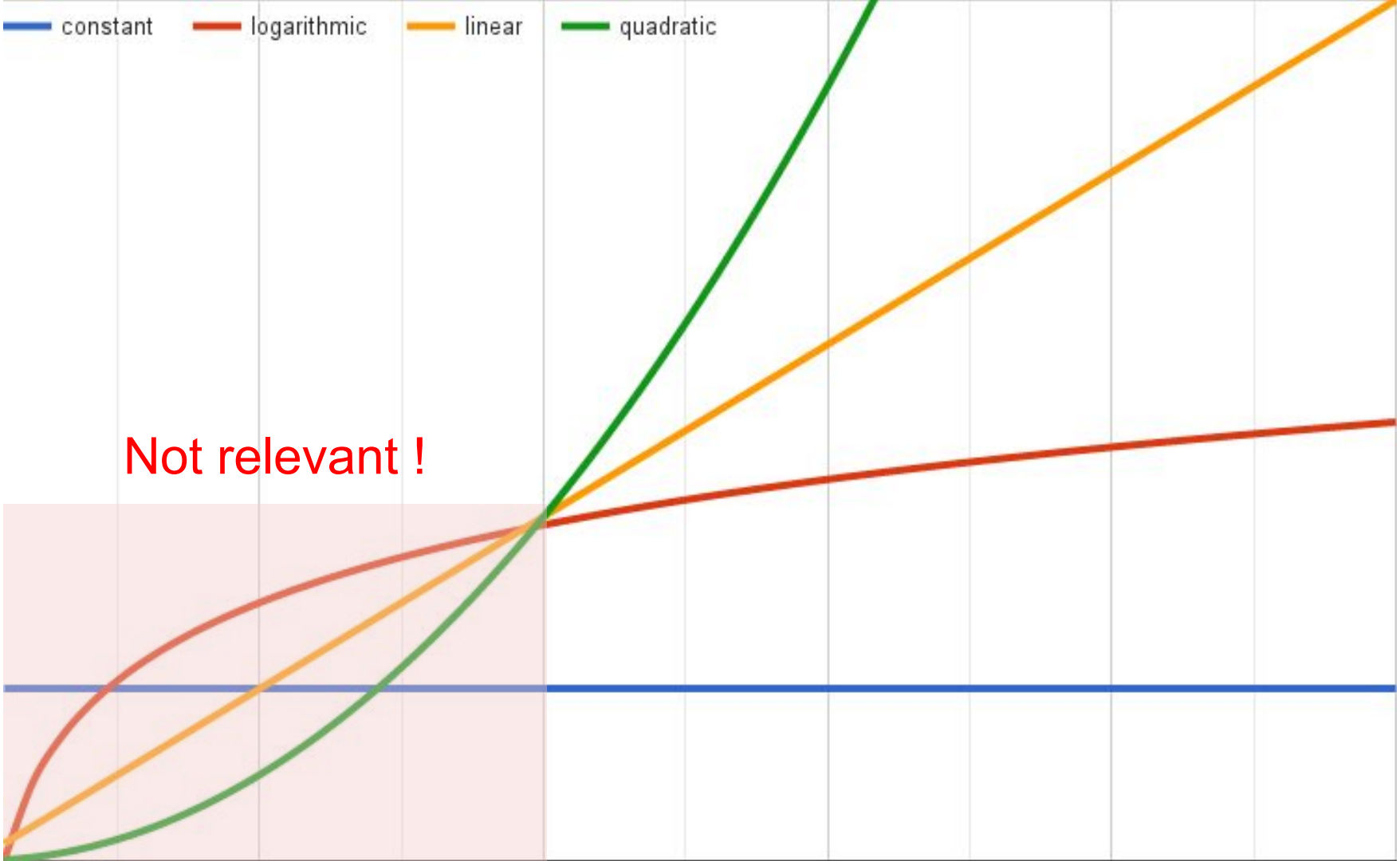


http://use-the-index-luke.com

http://coding-geek.com/how-databases-work/


WHAT ABOUT COMPLEXITY?
memegenerator.net

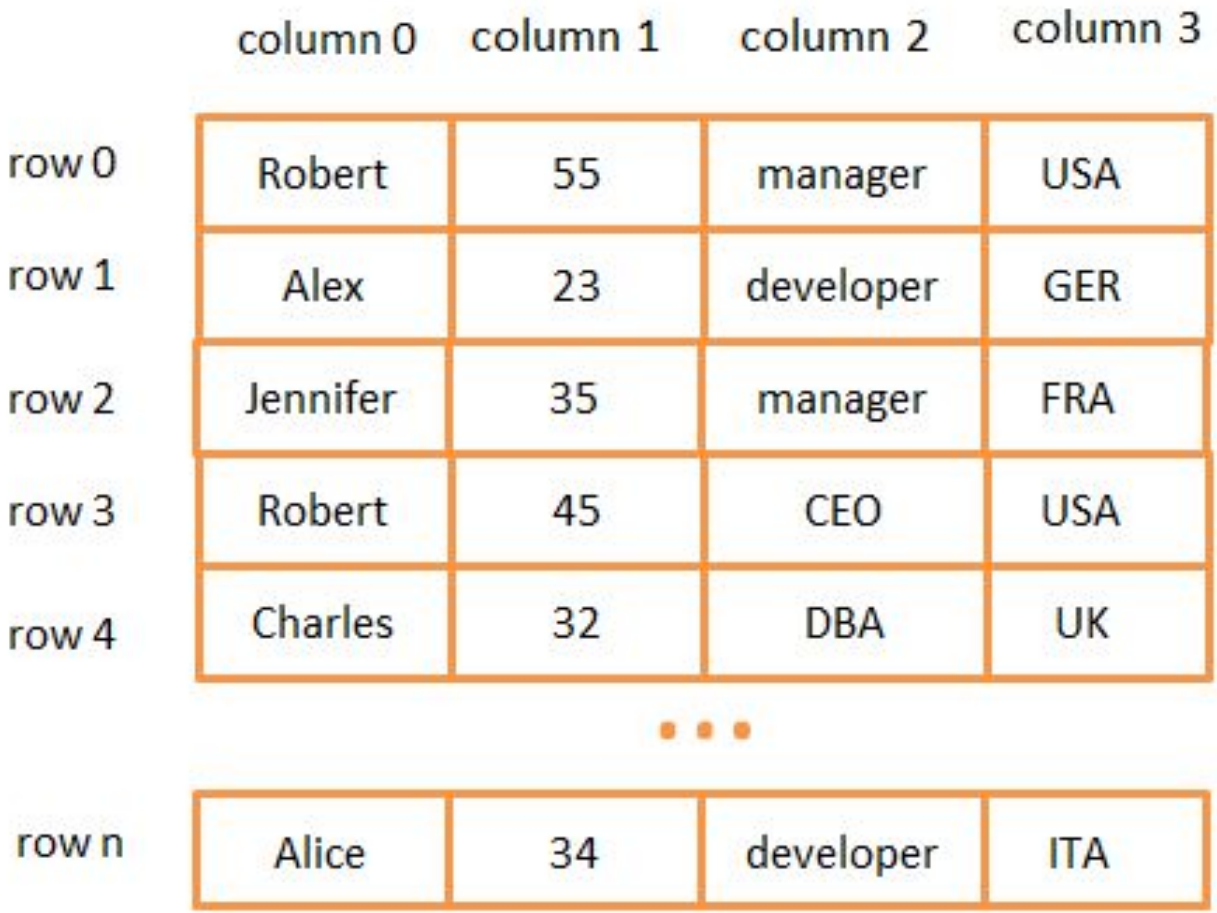

constant
logarithmic
linear
quadratic


constant
logarithmic
linear
quadratic
Not relevant !

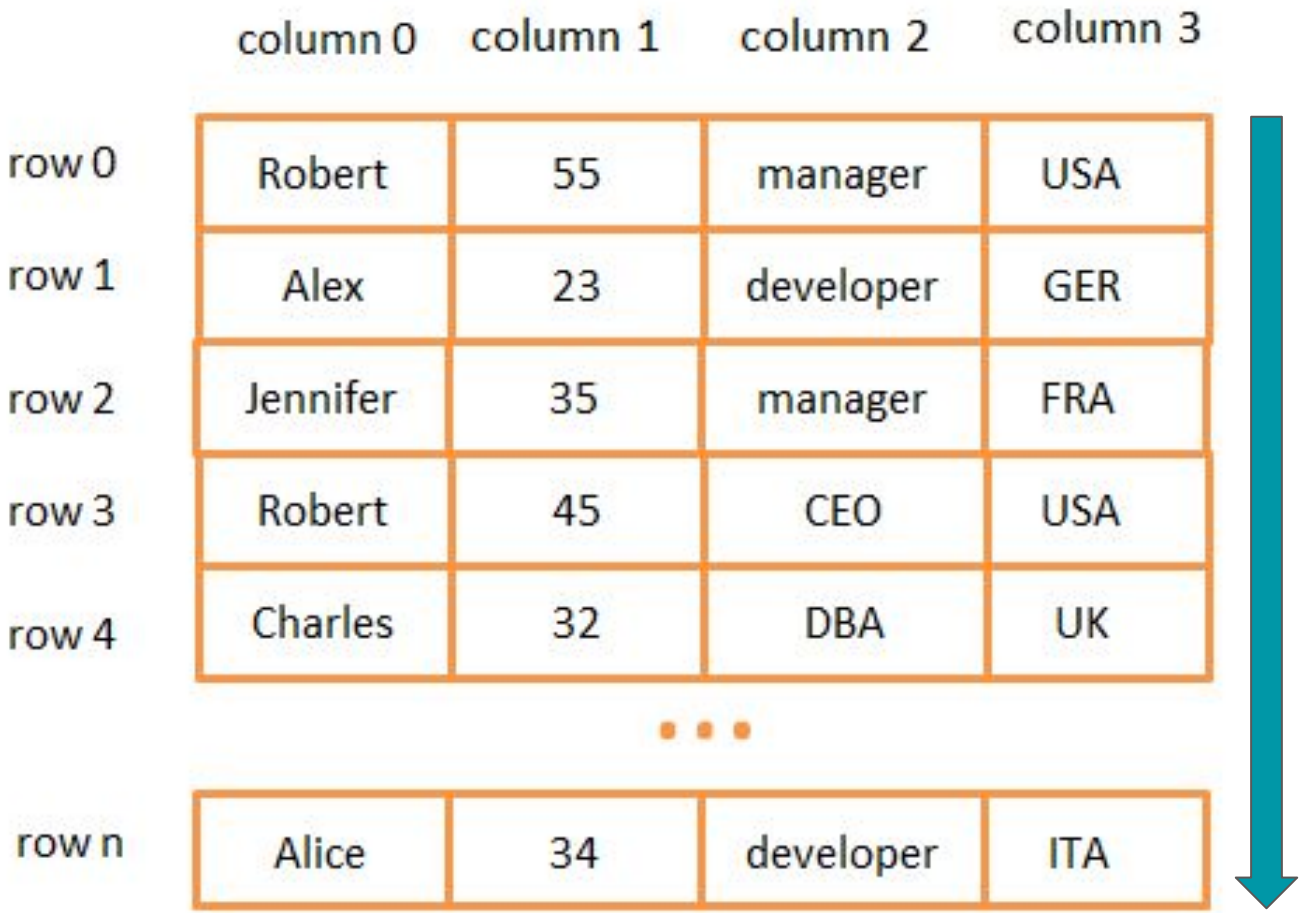
Tables

Back to basics...

Array


column 0
column 1
column 2
column 3
row 0
Robert
55
manager
USA
row 1
Alex
23
developer
GER
row 2
Jennifer
35
manager
FRA
row 3
Robert
45
CEO
USA
row 4
Charles
32
DBA
UK
row n
Alice
34
developer
ITA

Array



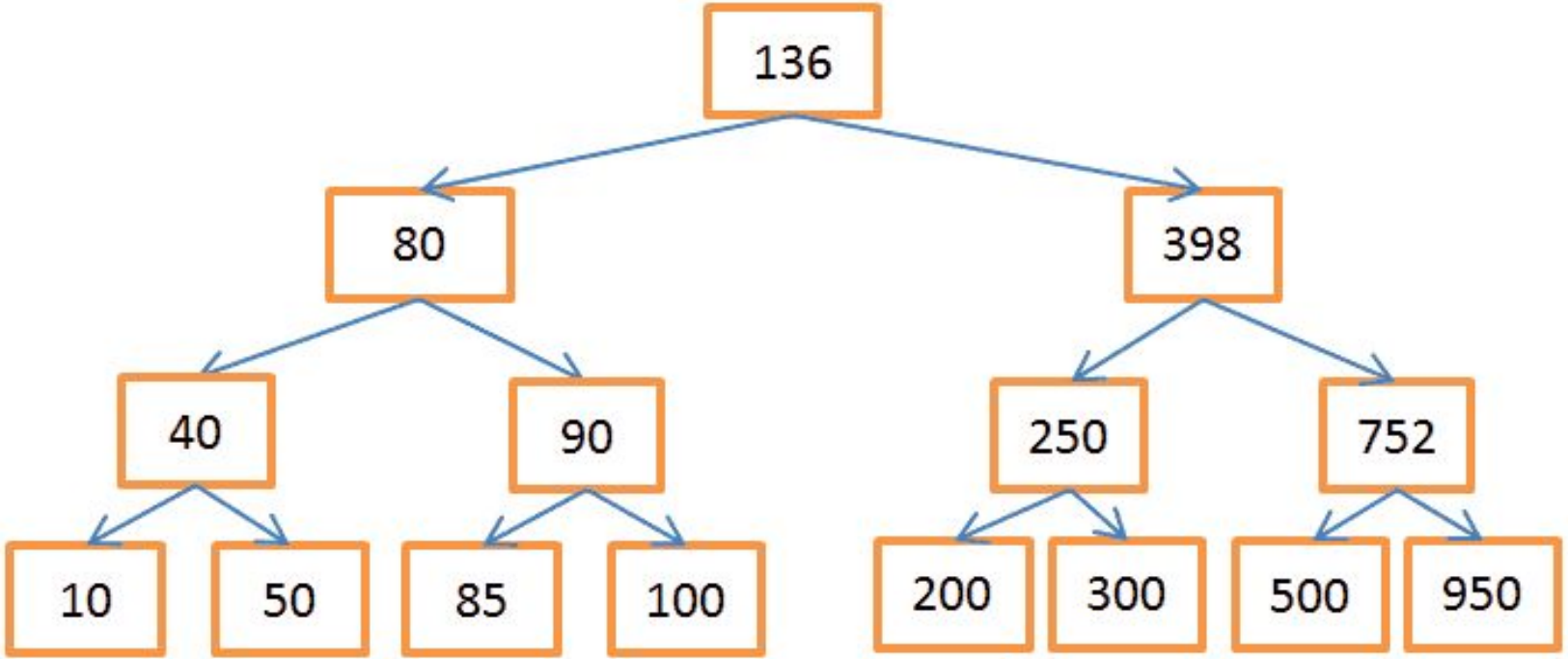
Full Table Scan

O(n)

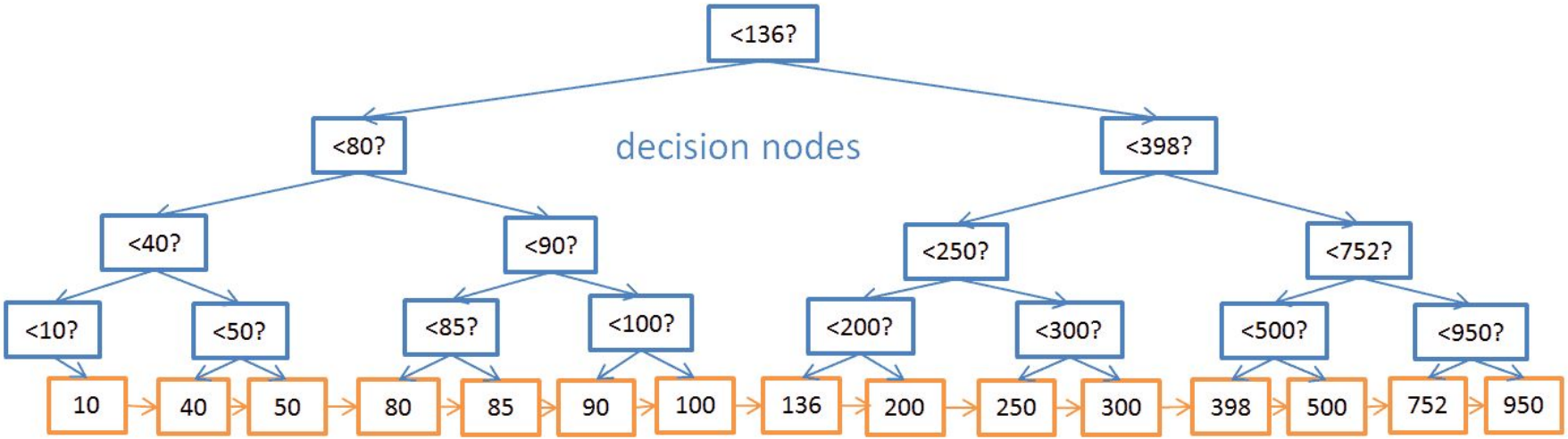
B-Tree

...or not to be...

Binary Search Tree

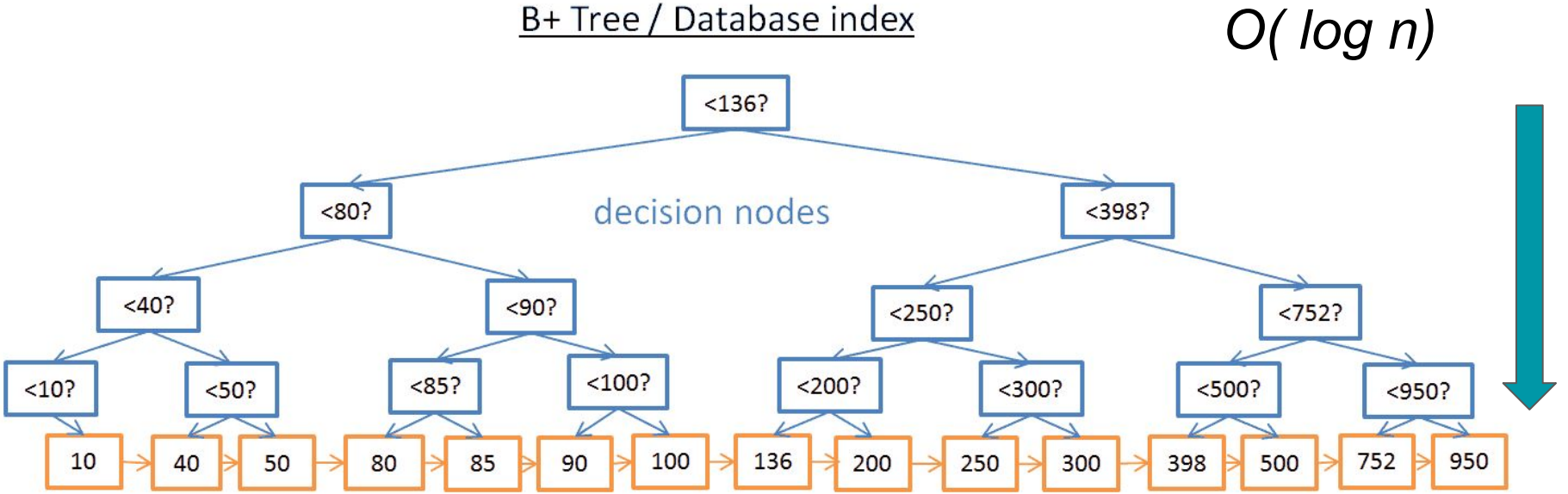


B+ Tree / Database index



Index unique scan

O(log n)

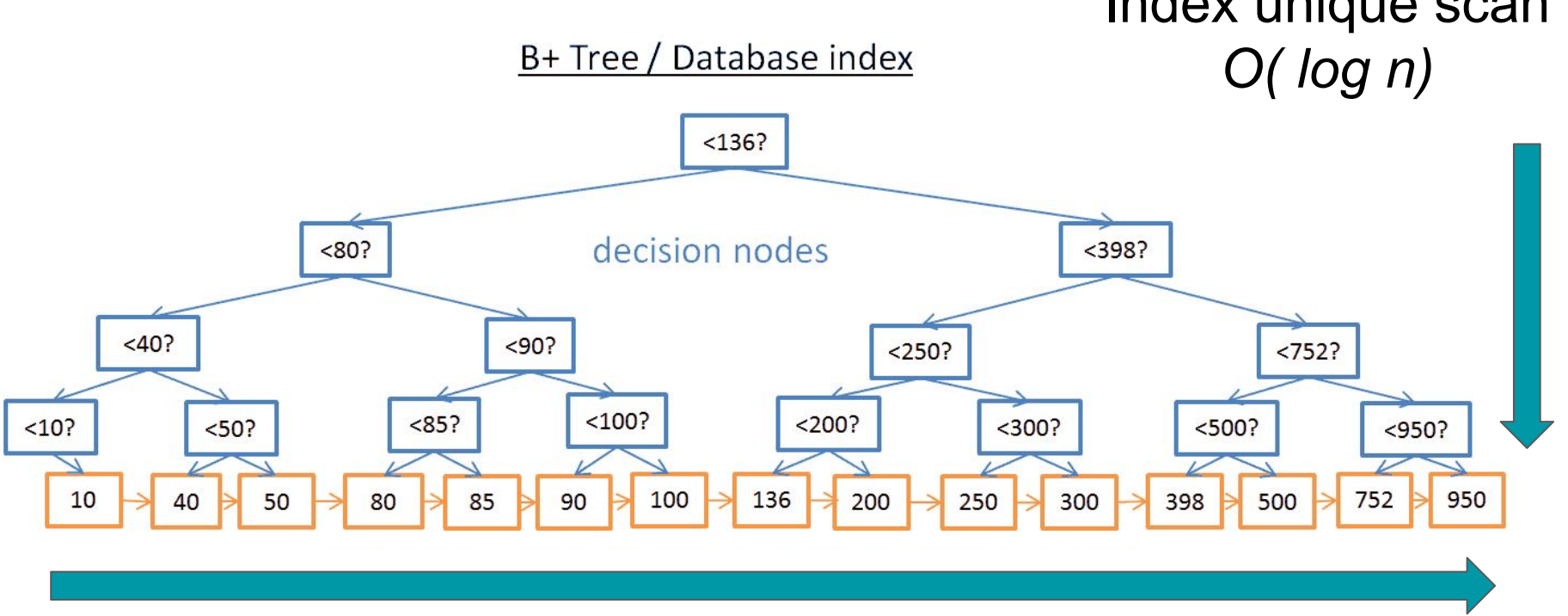


Index Unique Scan *O(log n)*

```
SELECT * FROM table WHERE id = ?
```

```
SELECT * FROM table WHERE id != ?

SELECT * FROM table WHERE id + 10 = ?
```

Index unique scan
O(log n)



Range scan *O(n)*

Range Scan *O(n)*

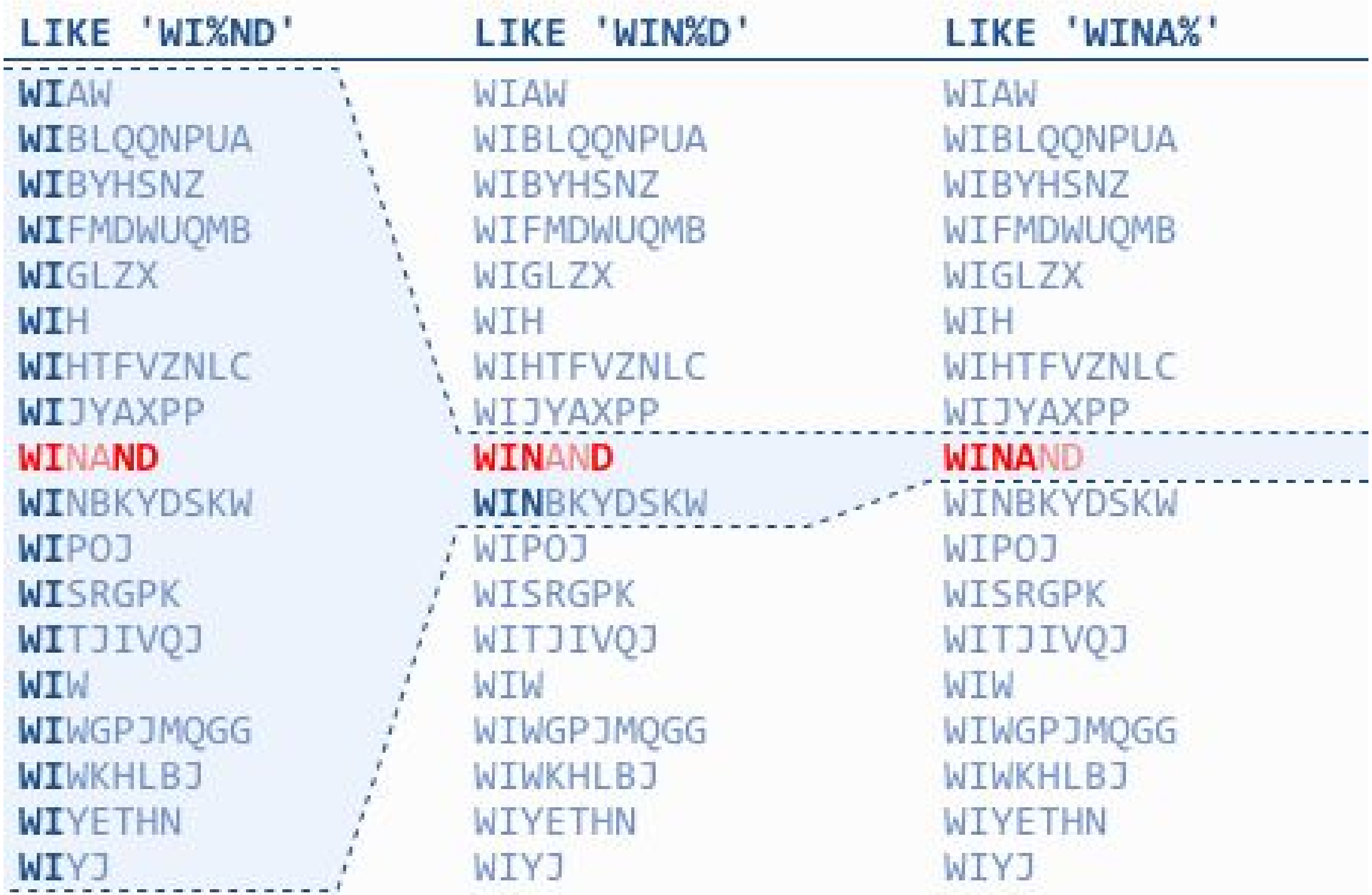
```
SELECT * FROM table WHERE id BETWEEN ? AND ?
```

```
SELECT * FROM table WHERE id > ? OR data = ?
```

```
SELECT * FROM table WHERE id > 0
```

Text Search

LIKE operator


LIKE 'WI%ND'
LIKE 'WIN%D'
LIKE 'WINA%'
WIAW
WIBLQQNPUA
WIBYHSNZ
WIFMDWUQMB
WIGLZX
WIH
WIHTFVZNLC
WIJYAXPP
WINAND
WINBKYDSKW
WIPOJ
WISRGPK
WITJIVQJ
WIW
WIWGPJMQGG
WIWKHLBJ
WIYETHN
WIYJ

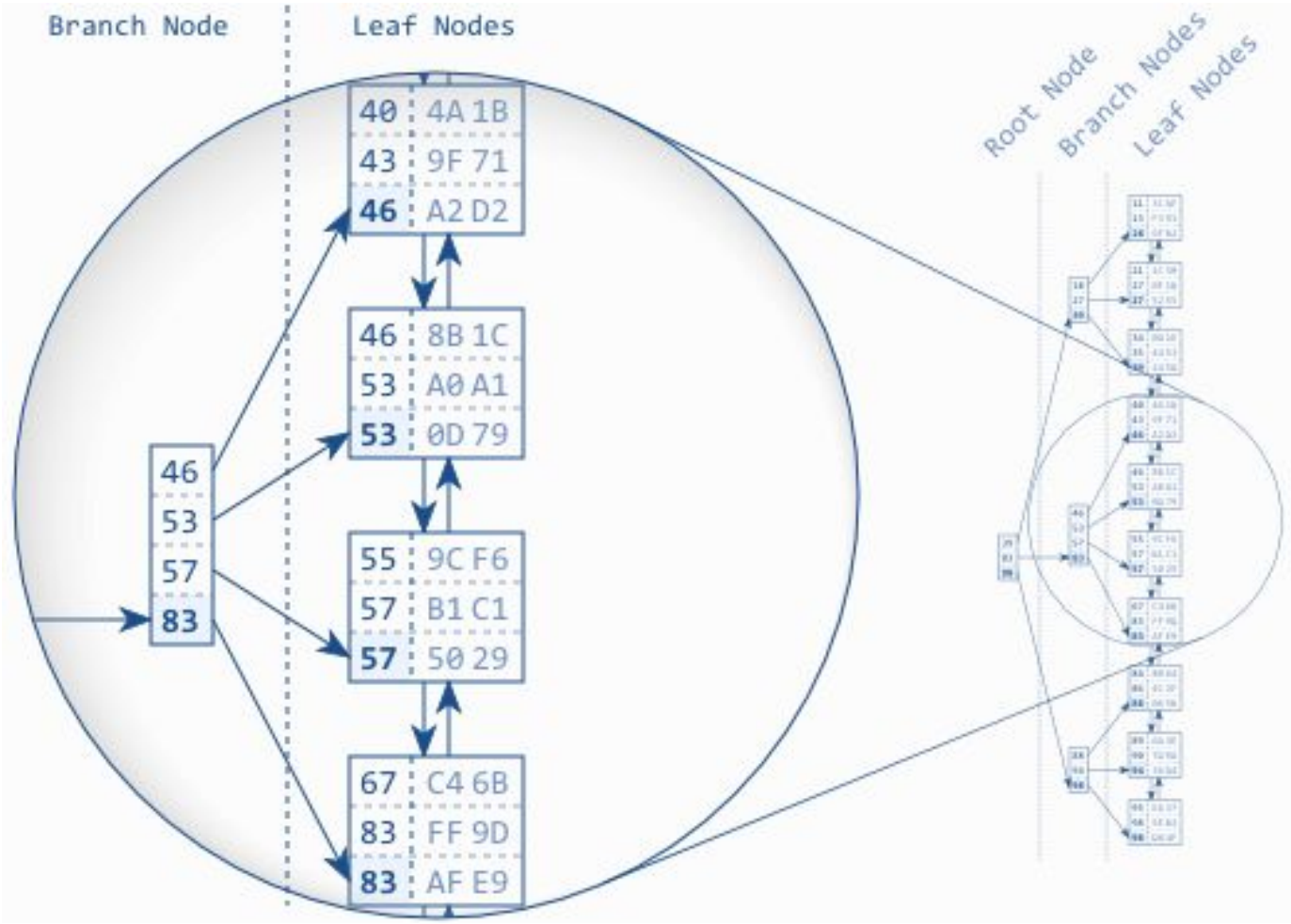
Range Scan *O(n)*

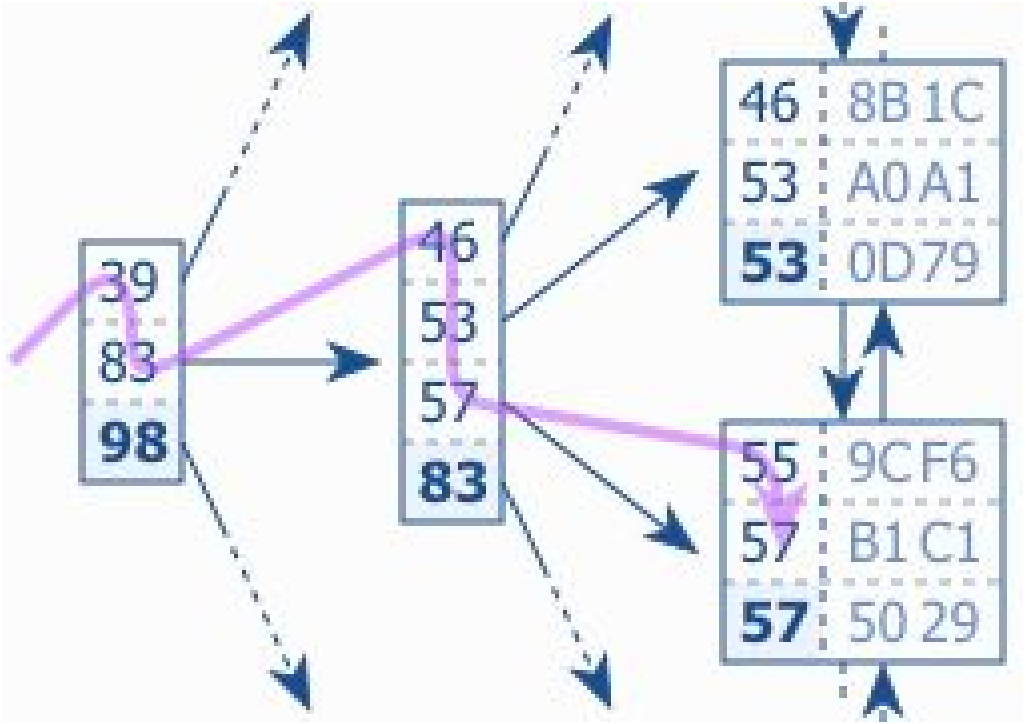
```
SELECT * FROM table WHERE text LIKE 'abc%'
```

```
SELECT * FROM table WHERE text LIKE '%abc%'

SELECT * FROM table WHERE UPPER(text) = 'abc'
```

More nodes!

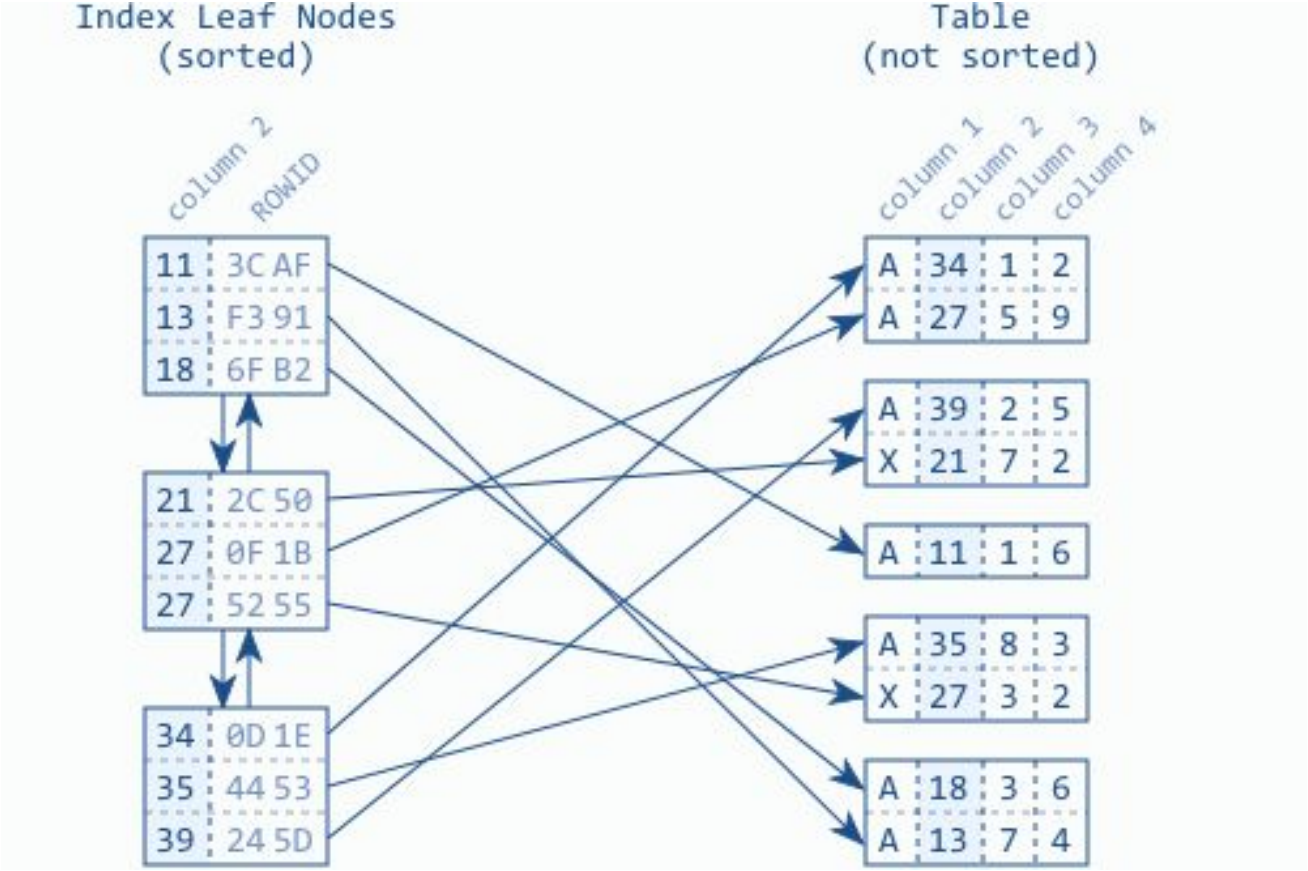
Logarithmic base


Branch Node
Leaf Nodes
40 4A 1B
43 9F 71
46 A2 D2
46 8B 1C
53 A0 A1
53 0D 79
55 9C F6
57 B1 C1
57 50 29
67 C4 6B
83 FF 9D
83 AF E9
46
53
57
83
Root Node
Branch Nodes
Leaf Nodes


39
83
98
46
53
57
83
46 8B 1C
53 A0 A1
53 0D 79
55 9C F6
57 B1 C1
57 50 29

Data access

The *hidden* cost


Index Leaf Nodes
(sorted)
column 2
ROWID
11 3C AF
13 F3 91
18 6F B2
21 2C 50
27 0F 1B
27 52 55
34 0D 1E
35 44 53
39 24 5D
Table
(not sorted)
column 1
column 2
column 3
column 4
A 34 1 2
A 27 5 9
A 39 2 5
X 21 7 2
A 11 1 6
A 35 8 3
X 27 3 2
A 18 3 6
A 13 7 4

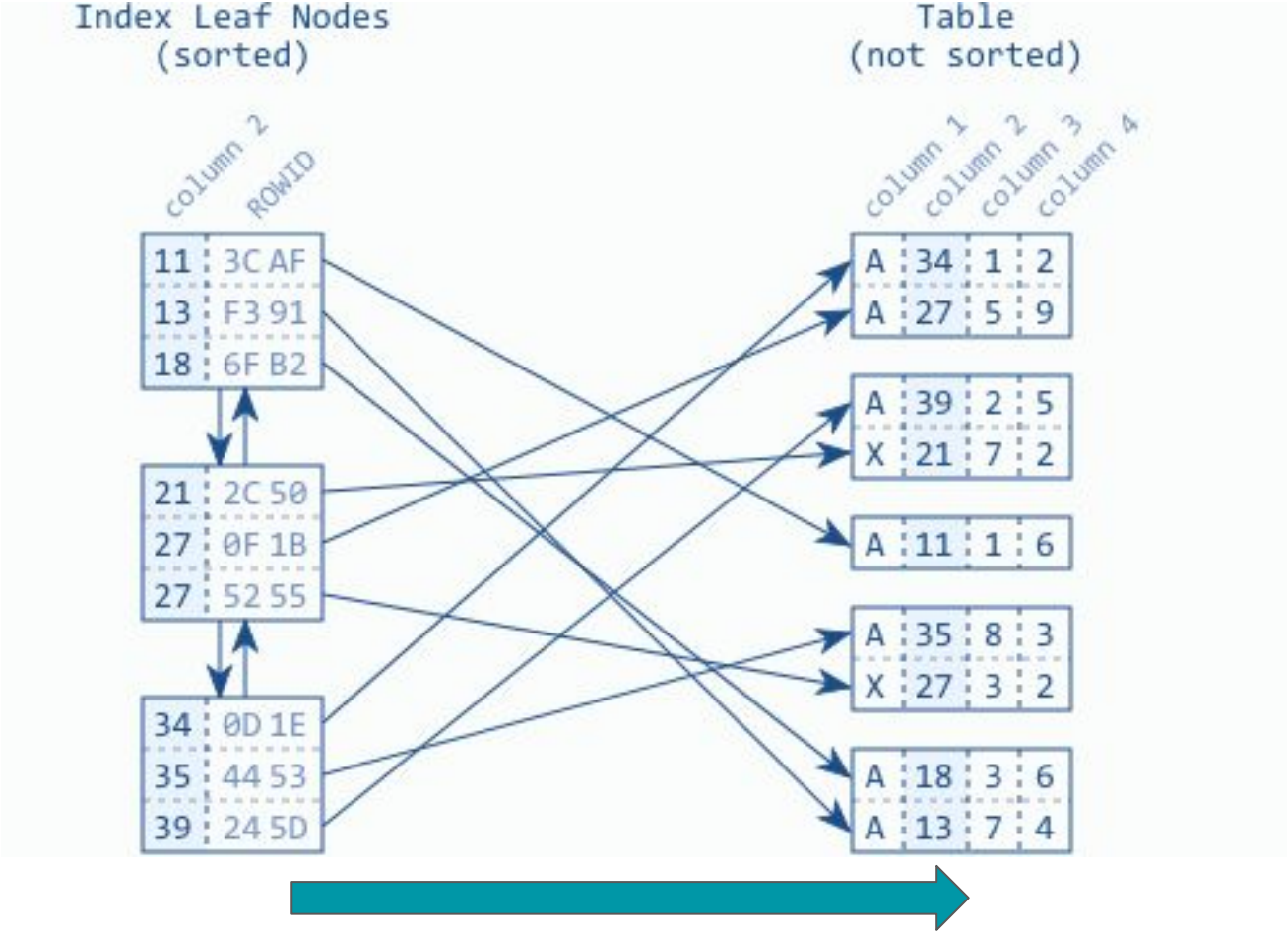


Table access by RowID *O(1)*

Clustering

Index-Only scan

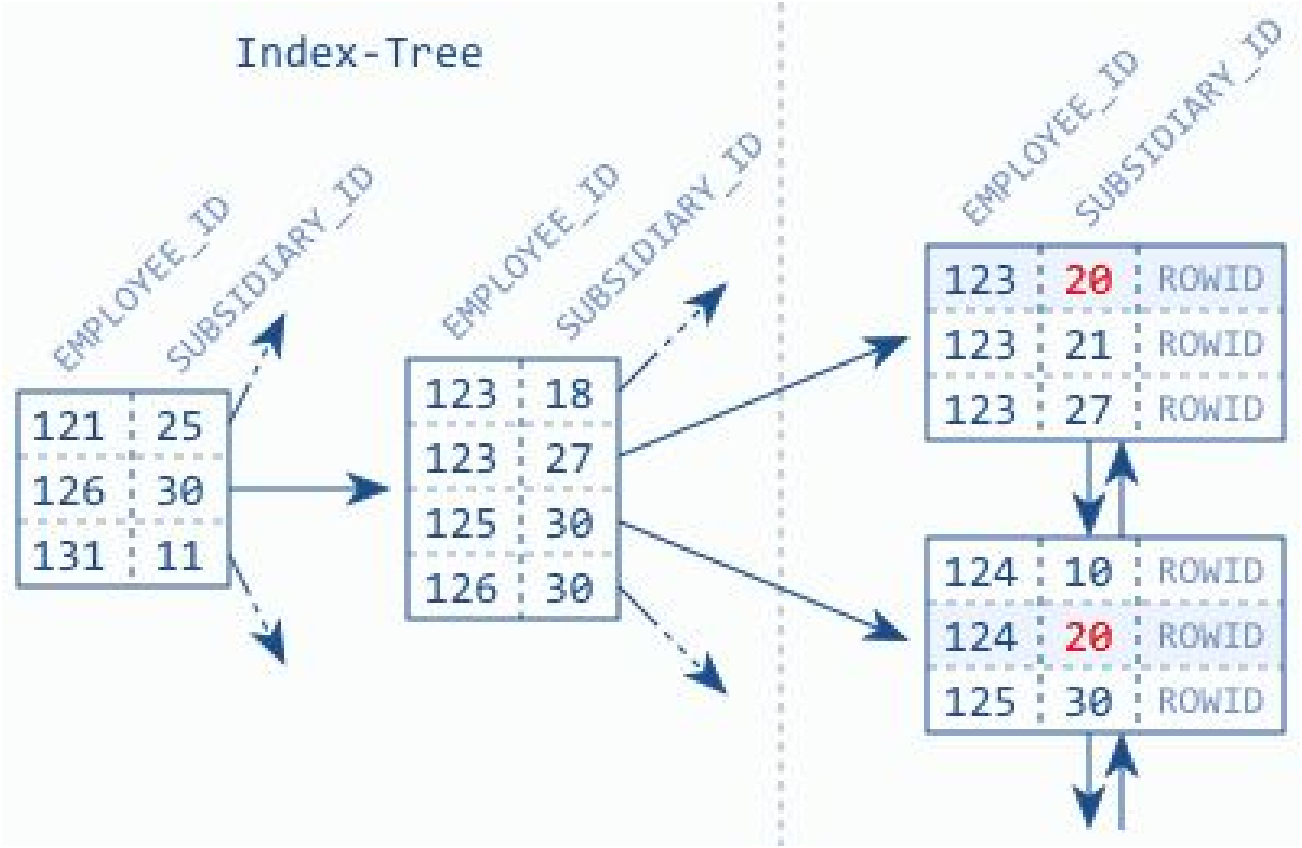
Index Only Scan *O(log n)*

```
SELECT id FROM table WHERE id = ?
```

```
SELECT * FROM table WHERE id = ?
```

Multi-column index

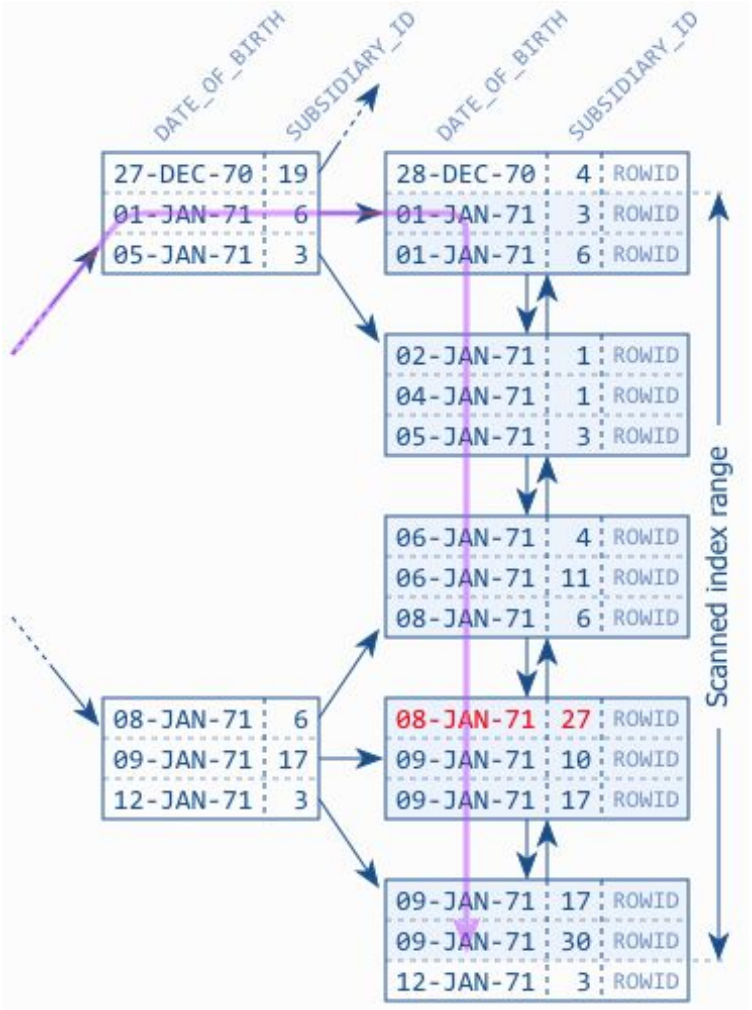
Order matters!



EMPLOYEE_ID, SUBSIDIARY_ID

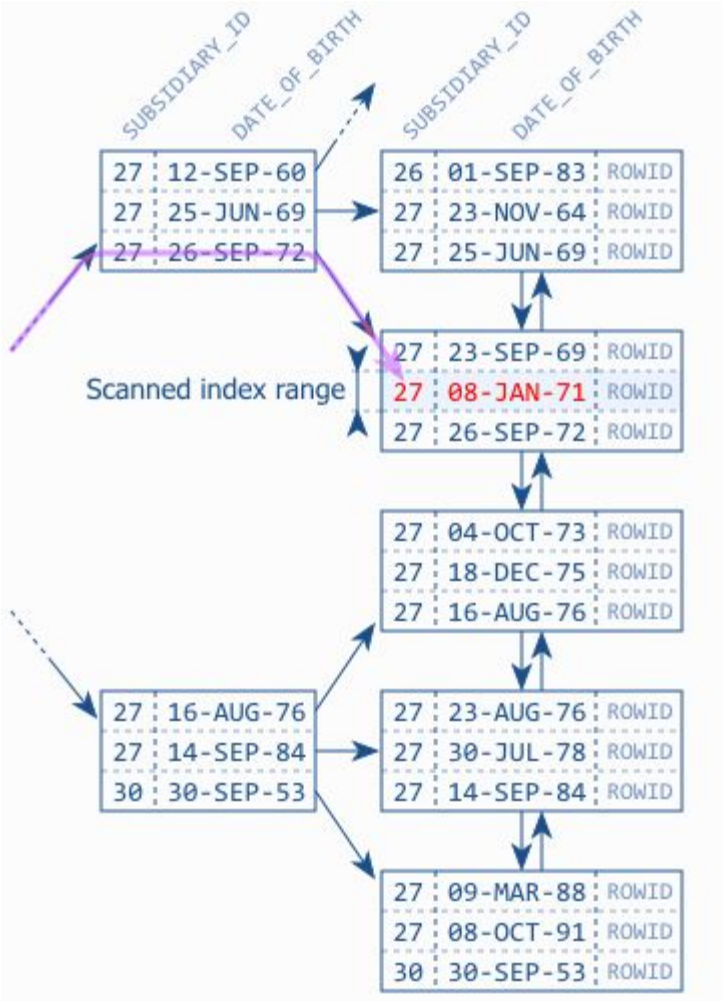
Range first

```
WHERE
subsidiary_id = ?
AND date_of_birth BETWEEN ? AND ?
```



Equality first

```
WHERE
subsidiary_id = ?
AND date_of_birth BETWEEN ? AND ?
```



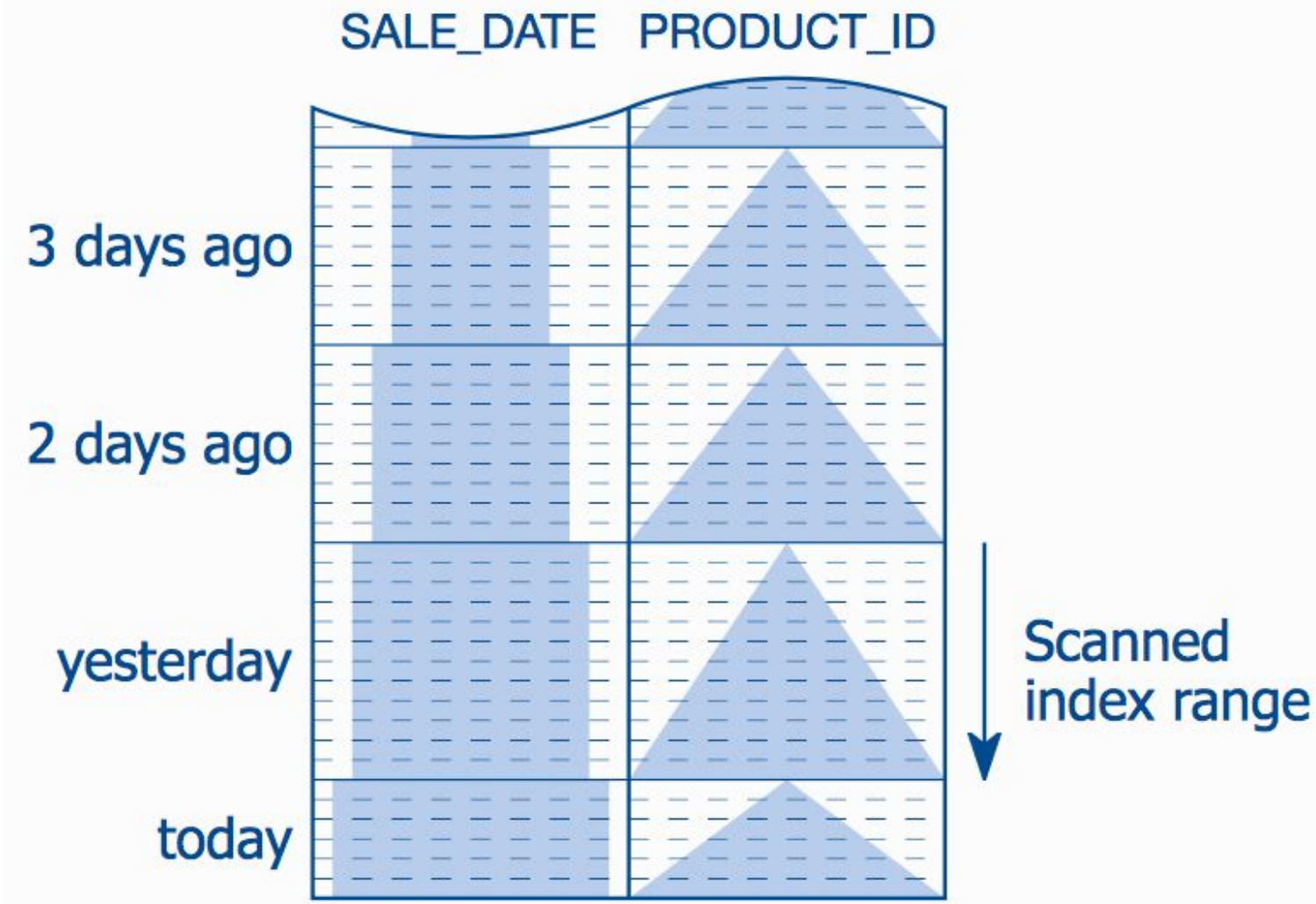
Range Scan *O(n)*

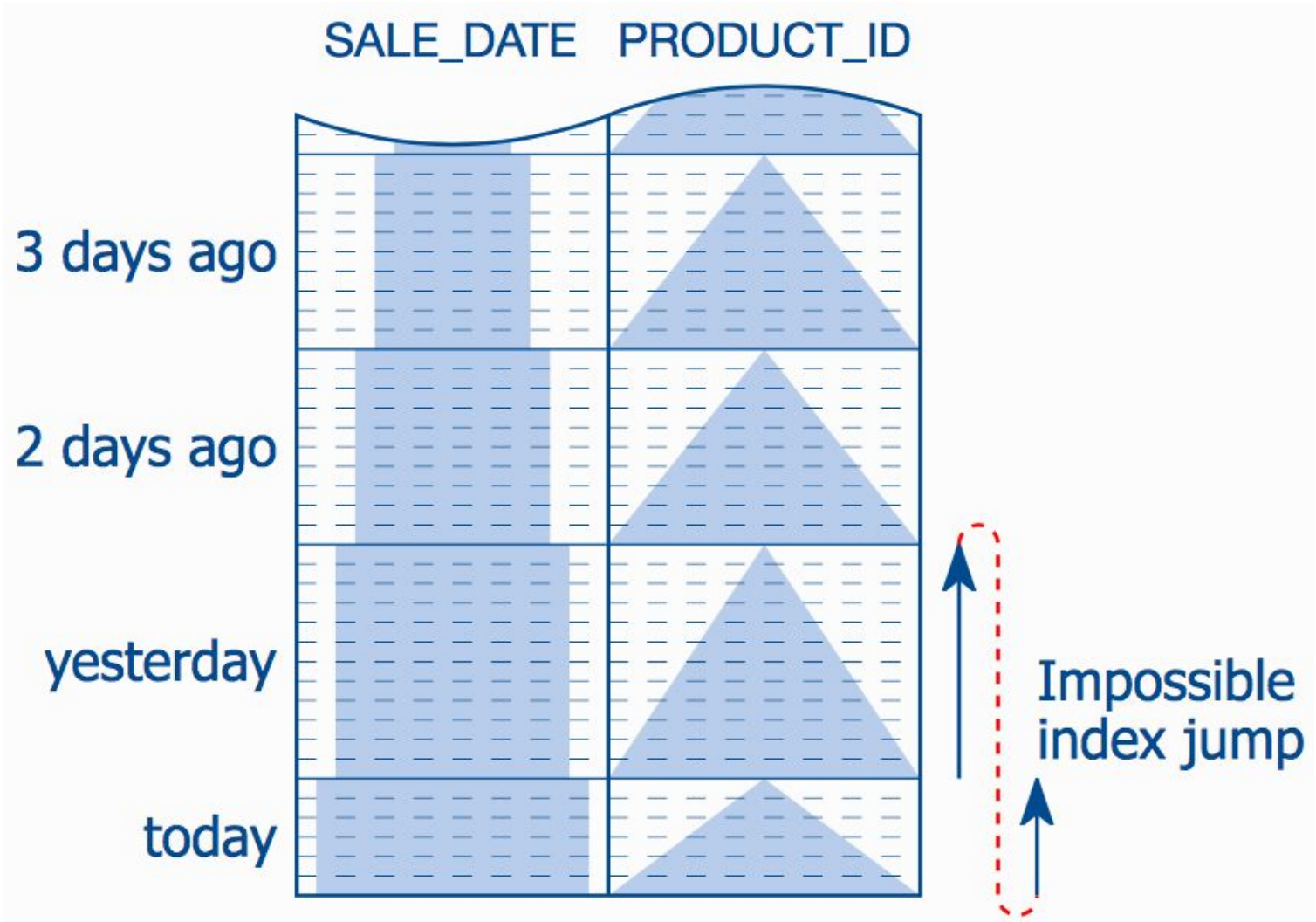
```
SELECT * FROM table WHERE id1 = ?

SELECT * FROM table WHERE id1 = ? AND id2 > ?
```

```
SELECT * FROM table WHERE id2 = ?

SELECT * FROM table WHERE id1 > ? AND id2 = ?
```

Sorting

Use the index, luke!


SALE_DATE
PRODUCT_ID
3 days ago
2 days ago
yesterday
today
Scanned
index range


SALE_DATE
PRODUCT_ID
3 days ago
2 days ago
yesterday
today
Impossible
index jump

Grouping

Use the sorting, luke!

Top-N Queries

Avoid full table scan!

Offset

Pagination helper


IT'S A TRAP !
memegenerator.net


SALE_DATE
3 days ago
2 days ago
yesterday
today
Page 1
Page 2
Page 3
Page 4
Result
Offset

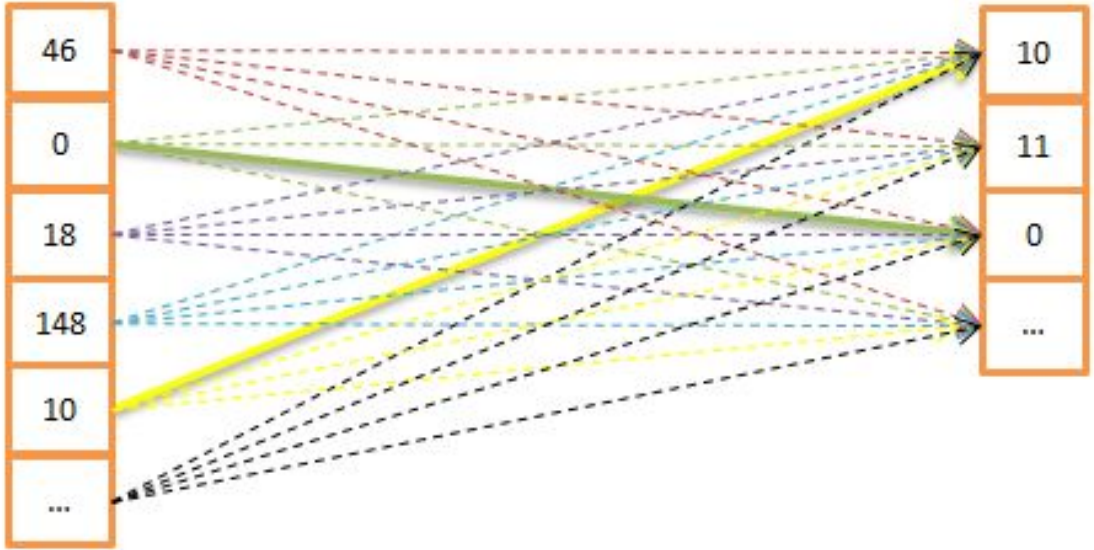

BRACE YOURSELF
JOINS ARE COMING
memegenerator.net

Join

Combinatorial power!


...LA FIN DU MONDE, OUI!...

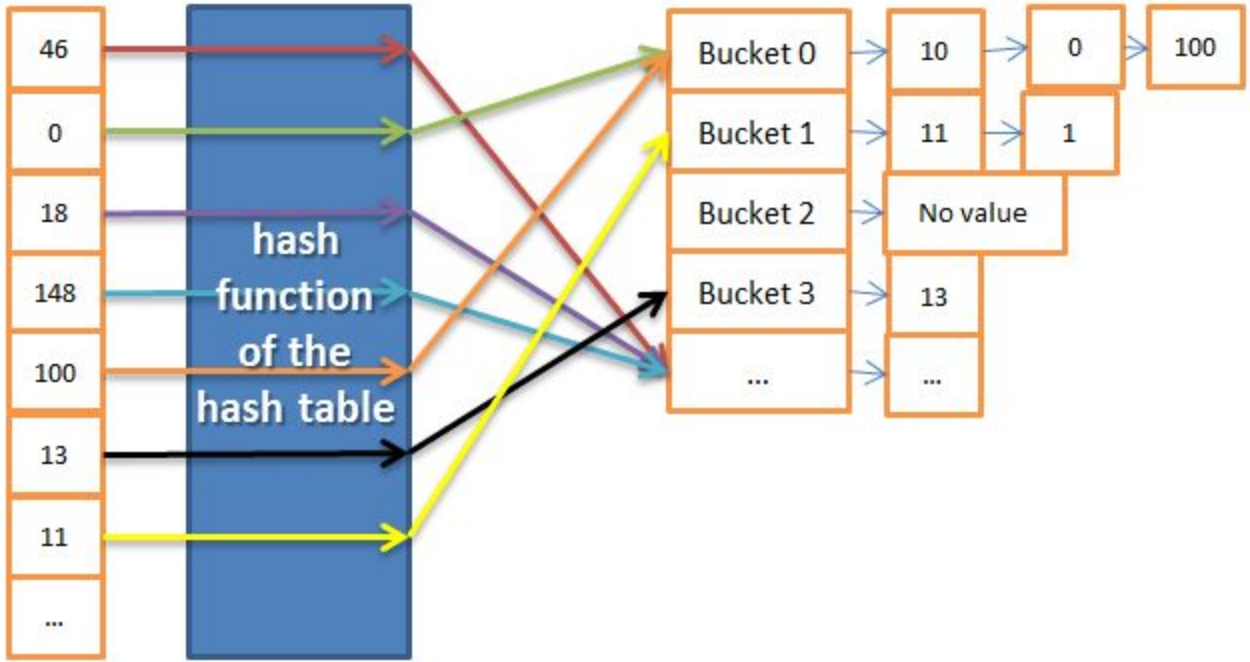
Nested Loop Join



Outer relation

Inner relation

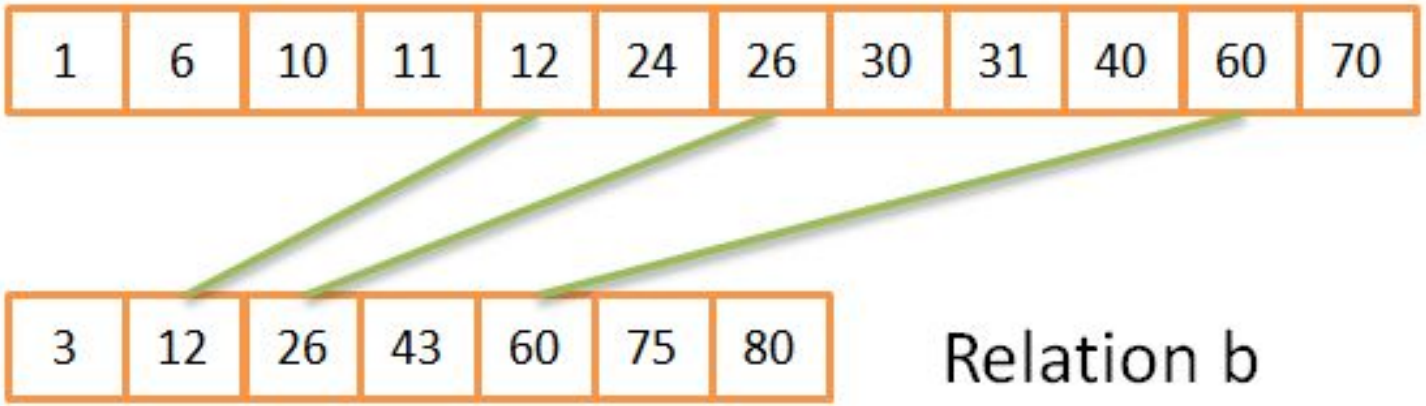
Hash Join



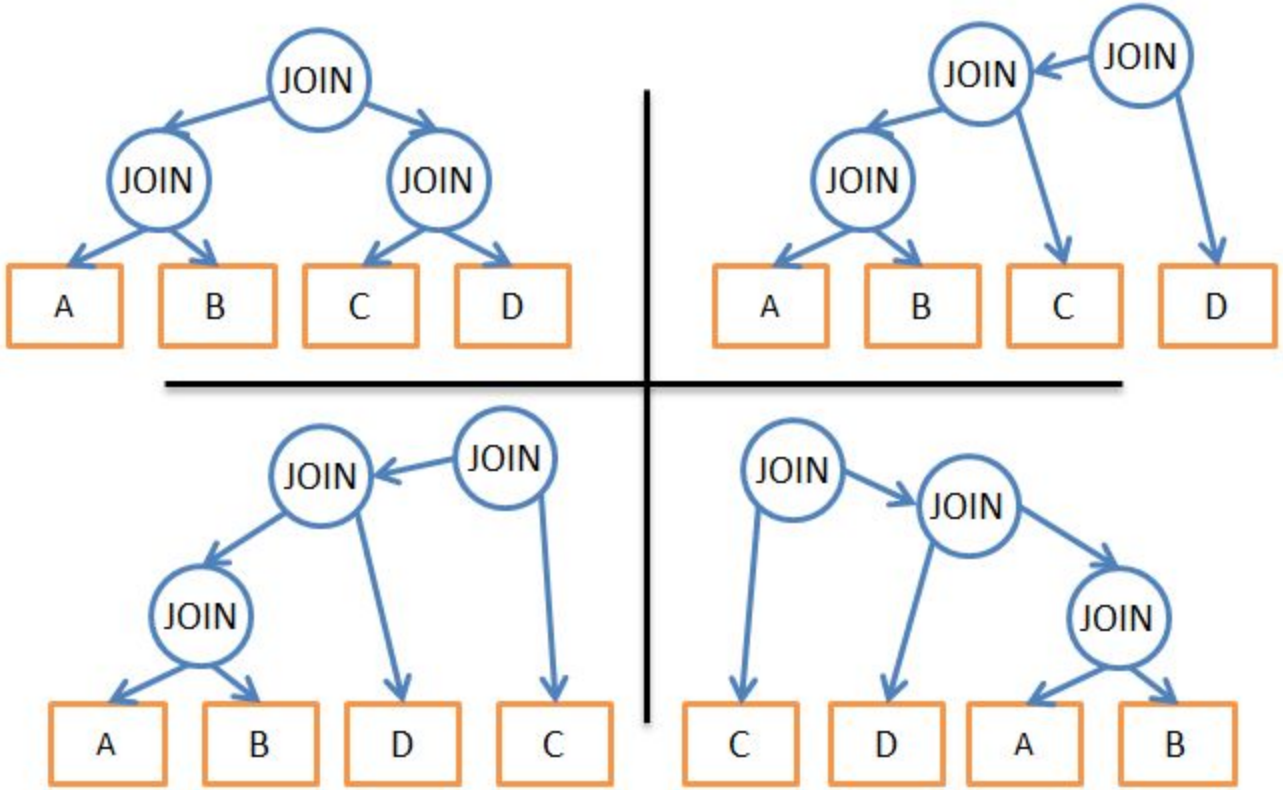
Outer relation

Inner relation (in-memory hash table)

Merge Join

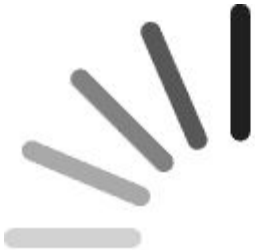

1
6
10
11
12
24
26
30
31
40
60
70
Relation a
3
12
26
43
60
75
80
Relation b

Join ordering problem



Statistics

Best execution plan



1 Query

≈

1 Index

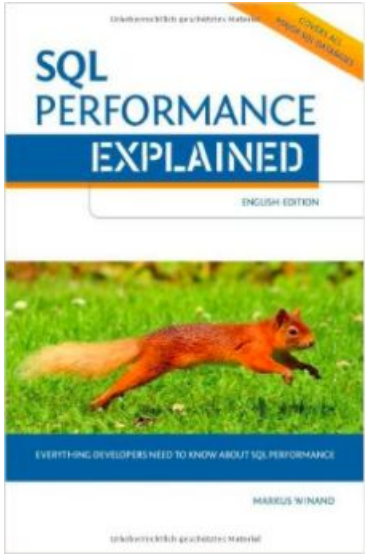
Speed vs (Memory and Speed update)

And now?

- Slow Log
- EXPLAIN

USE THE INDEX, LUKE

A guide to database performance for developers



http://use-the-index-luke.com

http://coding-geek.com/how-databases-work/

Thanks!