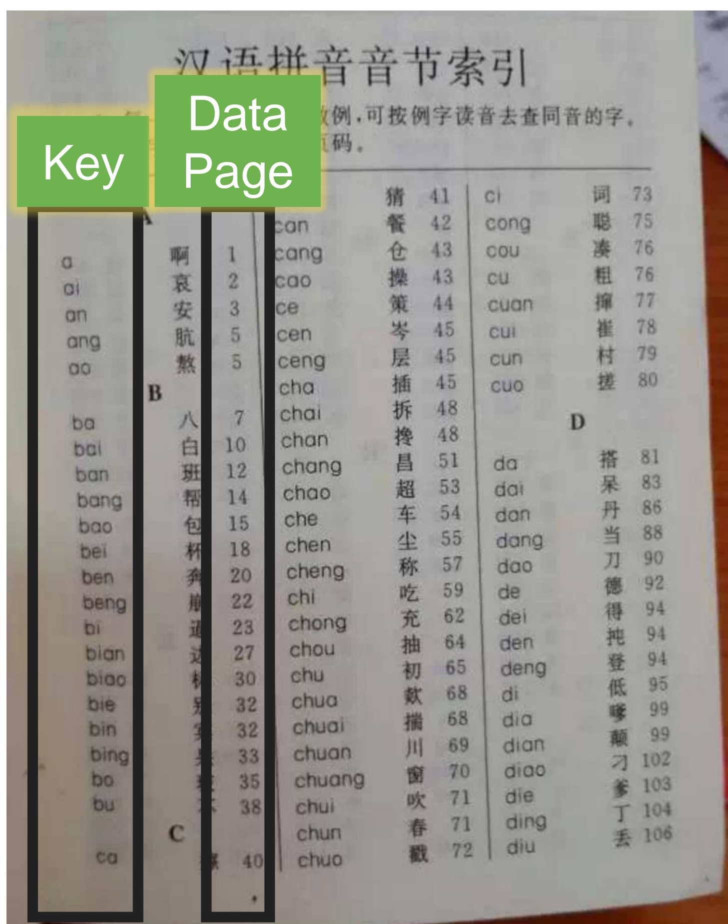
MongoDB索引优化

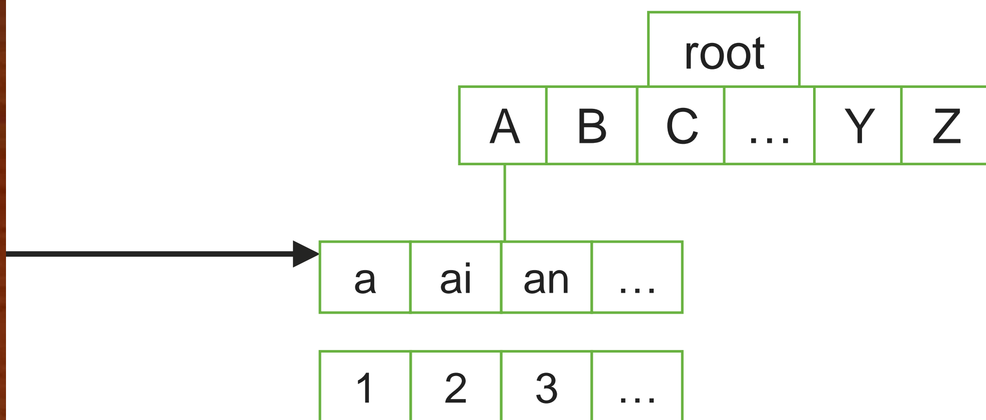
张耀星
Sr. Consulting Engineer
yaoxing.zhang@mongodb.com

mongoDB

Glossaries

Index/Key/DataPage——索引/键/数据页？



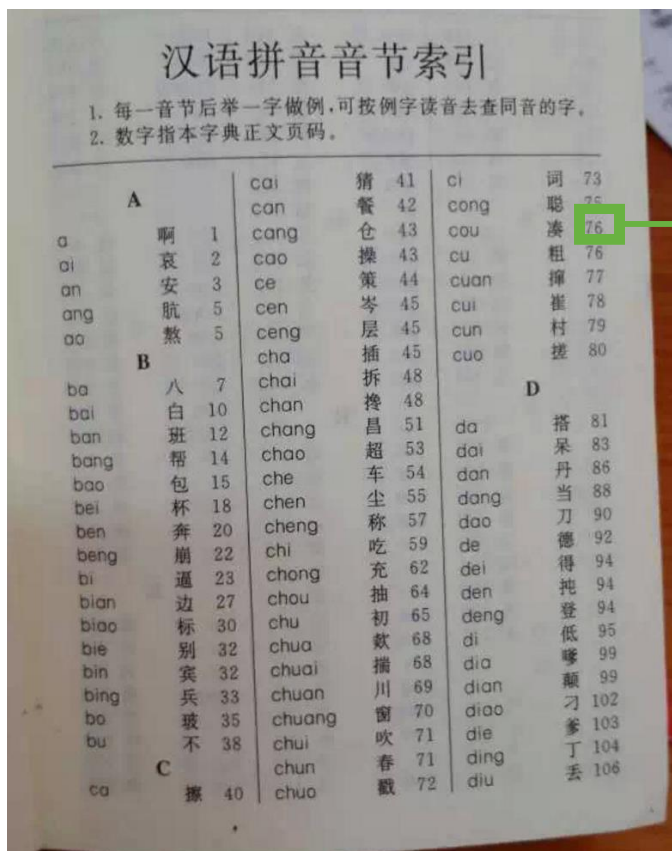


mongoDB

Glossaries

Covered Index/FETCH——索引覆盖/抓取？

```
db.human.createIndex({firstName: 1, lastName: 1, gender: 1, age: 1})
```

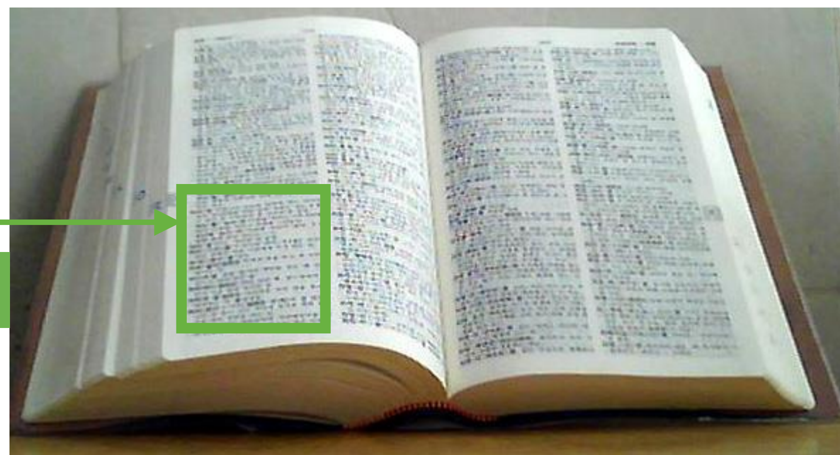


汉语拼音音节索引

1. 每一音节后举一字做例，可按例字读音去查同音的字。
2. 数字指本字典正文页码。

音节	例字	页码
A		
a	啊	1
ai	哀	2
an	安	3
ang	肮	5
ao	熬	5
B		
ba	八	7
bai	白	10
ban	班	12
bang	帮	14
bao	包	15
bei	杯	18
ben	奔	20
beng	崩	22
bi	逼	23
bian	边	27
biao	标	30
bie	别	32
bin	宾	32
bing	兵	33
bo	玻	35
bu	不	38
C		
ca	擦	40
cai	猜	41
can	餐	42
cang	仓	43
cao	操	43
ce	策	44
cen	岑	45
ceng	层	45
cha	插	45
chai	拆	48
chan	搀	48
chang	昌	51
chao	超	53
che	车	54
chen	尘	55
cheng	称	57
chi	吃	59
chong	充	62
chou	抽	64
chu	初	65
chua	欻	68
chuai	揣	68
chuan	川	69
chuang	窗	70
chui	吹	71
chun	春	71
chuo	戳	72
ci	词	73
cong	聪	[illegible]
cou	凑	76
cu	粗	76
cuan	撺	77
cui	崔	78
cun	村	79
cuo	搓	80
D		
da	搭	81
dai	呆	83
dan	丹	86
dang	当	88
dao	刀	90
de	德	92
dei	得	94
den	扽	94
deng	登	94
di	低	95
dia	嗲	99
dian	颠	99
diao	刁	102
die	爹	103
ding	丁	104
diu	丢	106

FETCH



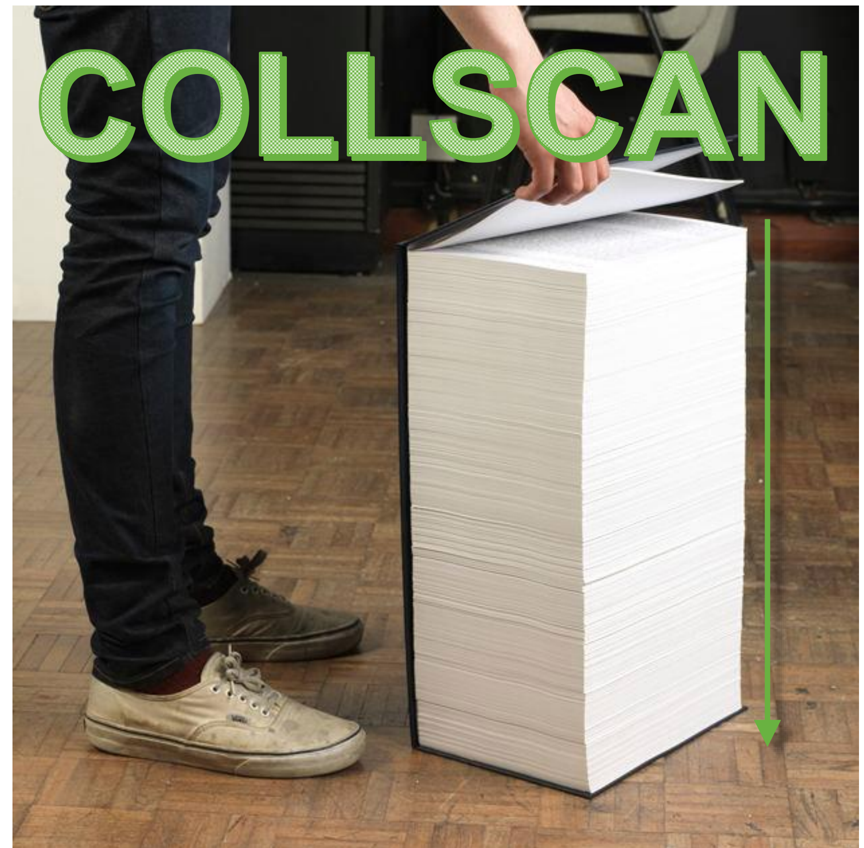
如果所有需要的字段都在索引中，不需要额外的字段，就可以满足索引覆盖的要求，不再需要从数据页加载数据，这就是索引覆盖。

mongoDB

Glossaries

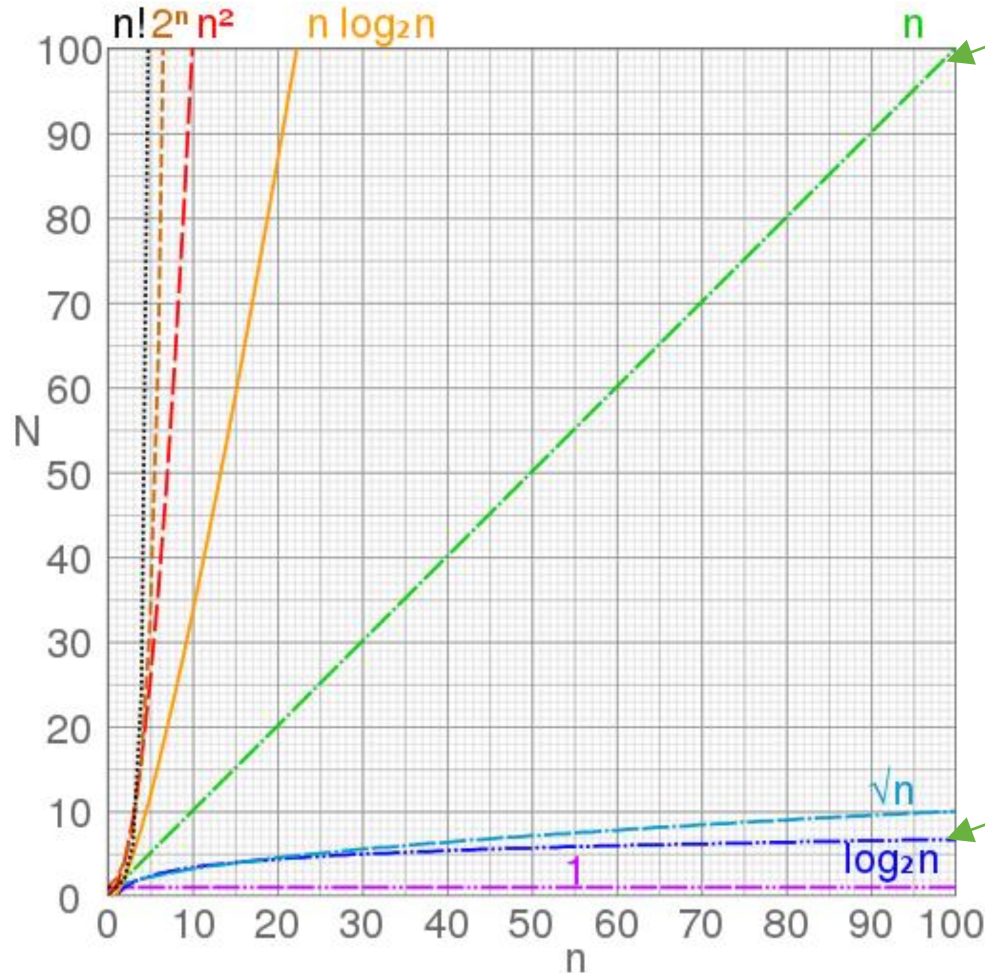
IXSCAN/COLLSCAN——索引扫描/集合扫描

VS



Glossaries

Big O Notation——大O符号



COLLSCAN

IXSCAN

mongoDB

Glossaries

Query Shape——查询的形状？

```
{
    "_id" : ObjectId("579a52d5bf98013eaf82b2ec"),
    "sku" : "sku001",
    "headline" : "LED Tonka Dump Truck Night Light",
    "url" : "B00HFMHPEU",
    "categories" : [1, 12, 39],
    "attributes" :
        "attrId"                                         ea"),
        "name" :
        "value" :
    }, {
        "attrId"                                         eb"),
        "name" : "size",
        "value" : 5
    }]
}
> db.product.find({"attributes.name": "color", "attributes.value": "red"})
> db.product.find({"attributes.name": "size", "attributes.value": "5"})
> db.product.find({"sku": "sku001"})
> db.product.find({"sku": "sku002"})
```

```
query shape = {
  "attributes.name": 1,
  "attributes.value": 1
}
```

mongoDB

Glossaries

Index Prefix——索引前缀？

```
db.human.createIndex({firstName: 1, lastName: 1, gender: 1, age: 1})
```

以上索引的全部前缀包括：

- `{firstName: 1}`
- `{firstName: 1, lastName: 1}`
- `{firstName: 1, lastName: 1, gender: 1}`

所有索引前缀都会被覆盖，没有必要单独建立前缀索引！

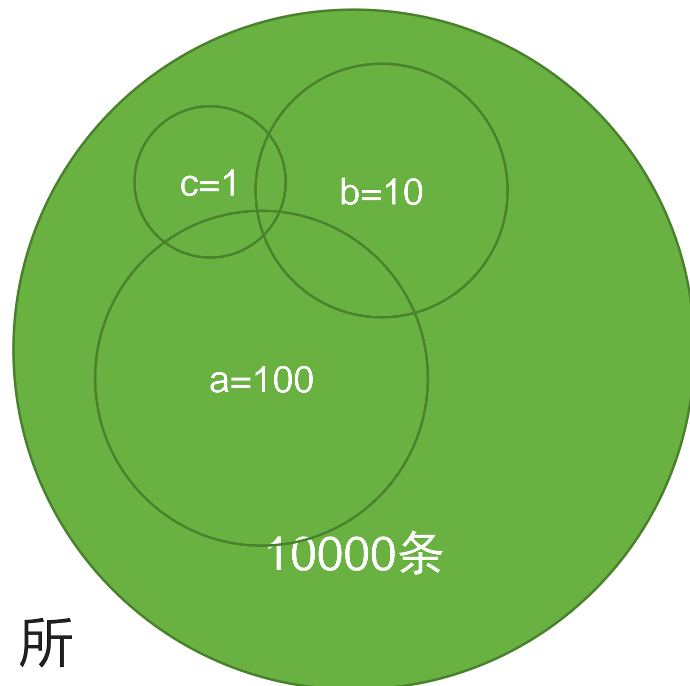
mongoDB

Glossaries

Selectivity——过滤性

在一个有10000条记录的集合中：

- 满足a=100的记录有1000条
- 满足b=10的记录有100条
- 满足c=1的记录有5条



条件c能过滤掉最多的数据，b其次，a最弱。所以c的过滤性(selectivity)大于b大于a。

如果要查询同时满足：

```
a == 100 && b == 10 && c == 1
```

的记录，但只允许为a/b/c中的一个建立索引，应该把索引放在哪里？

mongoDB

Index Fundamentals

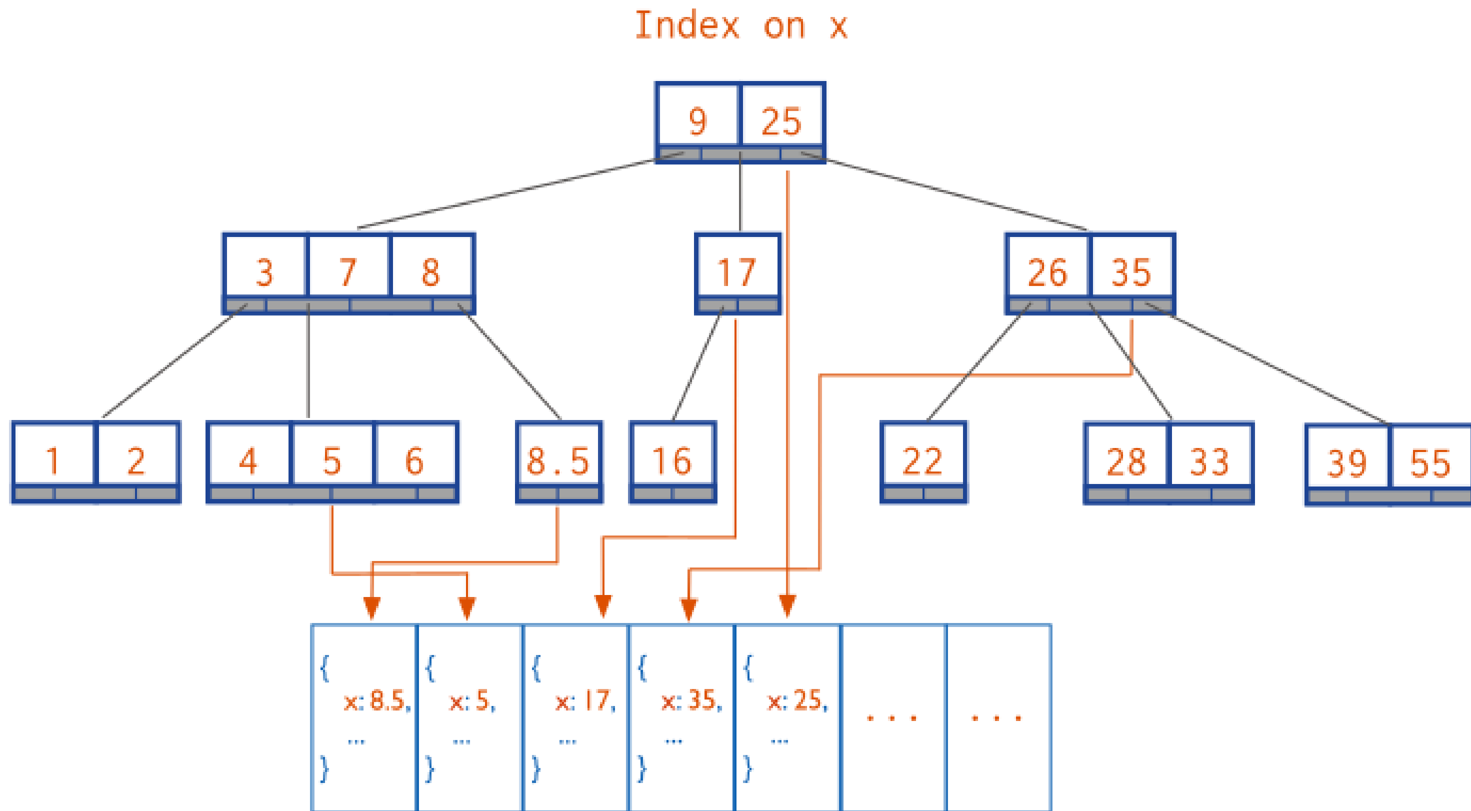
- 单字段索引(Single-field Indexes)
- 复合索引(Compound Indexes)
- 多键索引(Multikey Indexes)
- 地理位置索引(Geospatial Indexes)
- 全文索引(Text Indexes)

单字段索引本质上是特殊的复合索引，所以我们要讨论的其实是一个东西

Index Fundamentals

索引的基本作用——搜索+排序

B-Tree != Binary Tree && B-Tree == Self-Balancing Tree

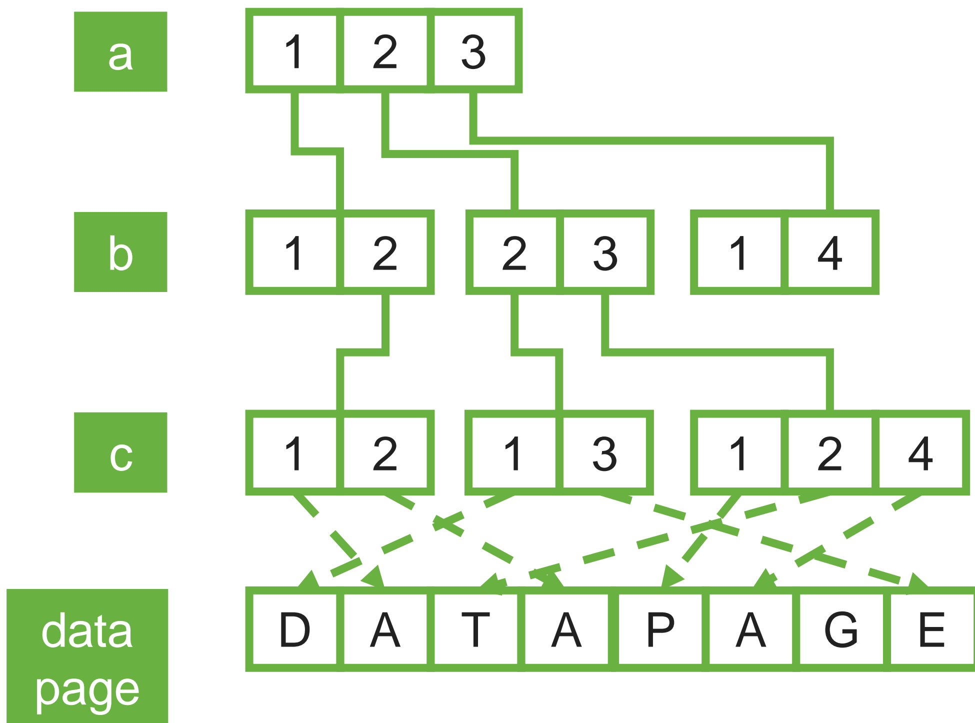


mongoDB

Index Practice

复合索引的工作模式——过滤

```
db.test.createIndex({a: 1, b: 1, c: 1})
```




```
db.test.find({a: 2, b: 2, c: 1})
```

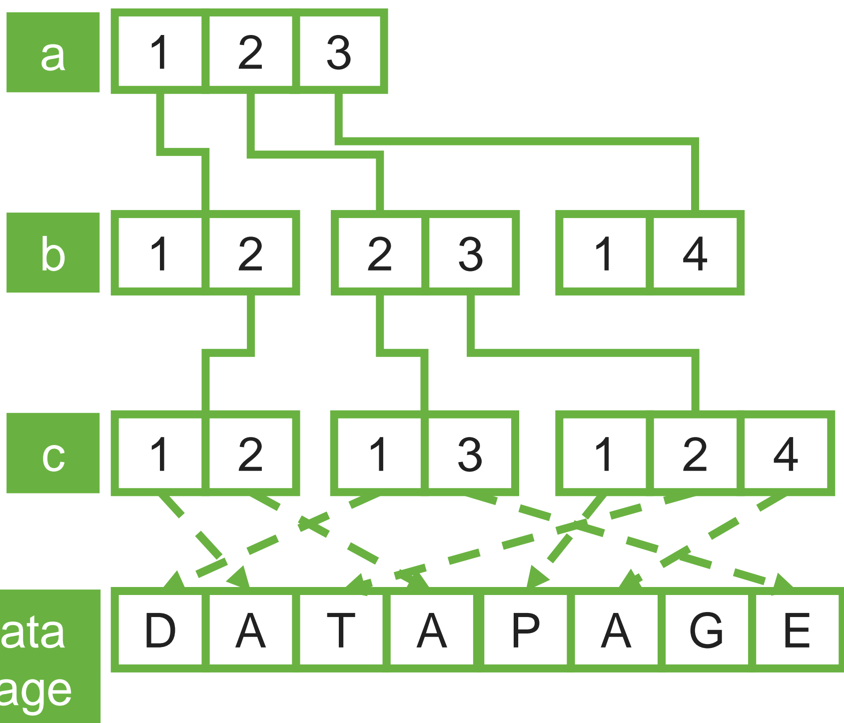
```
db.test.find({a: 2, b: {$gte: 2, $lte: 3}, c: 1})
```

```
db.test.find({a: 2, b: 3, c: {$gte: 2, $lte: 4}})
```

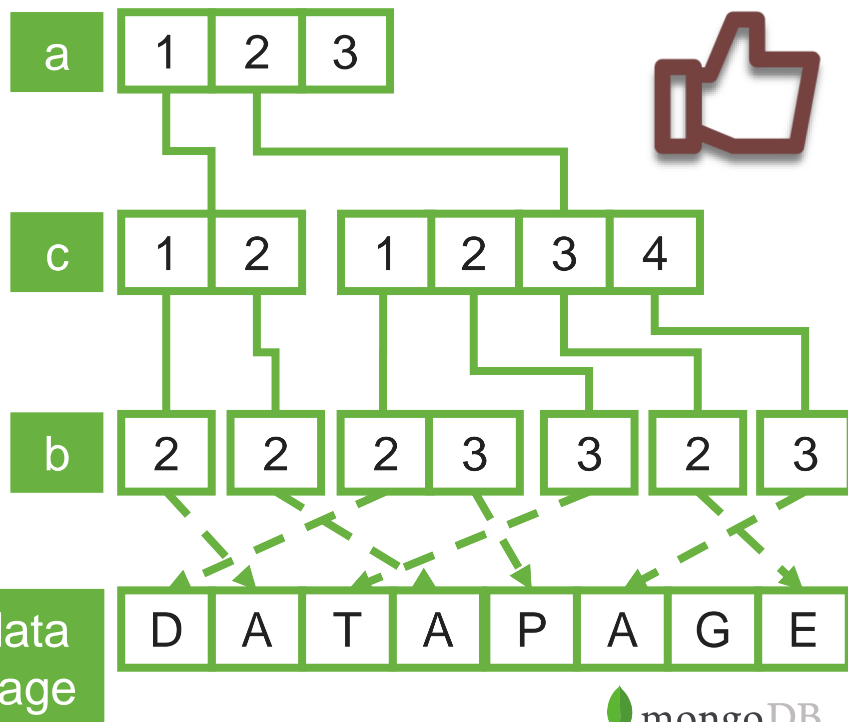
Index Practice

复合索引的工作模式——索引顺序的影响

```
db.test.find({a: 2, b: {$gte: 2, $lte: 3}, c: 1})
```



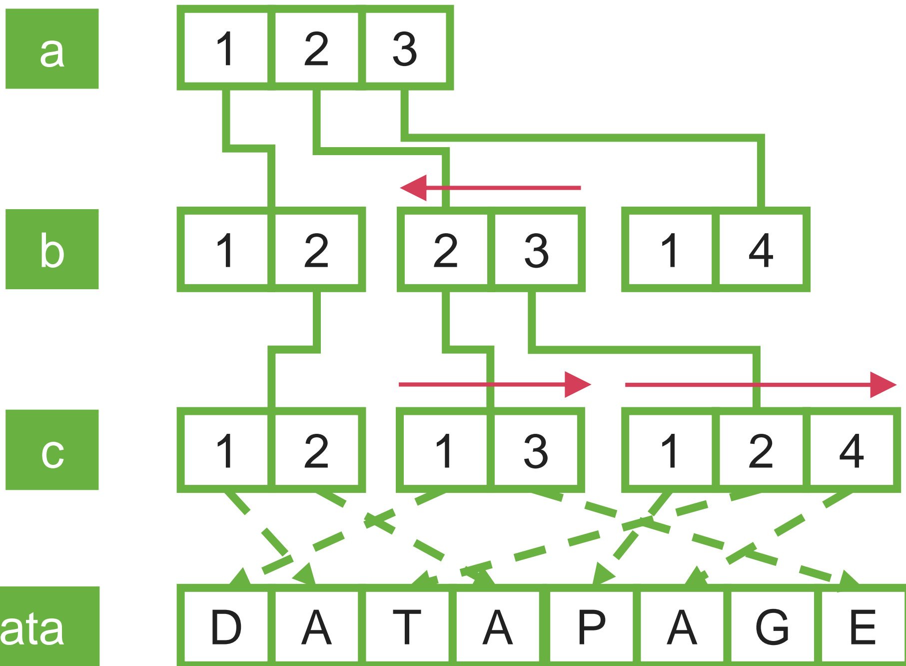
VS



Index Practice

复合索引的工作模式——排序

```
db.test.createIndex({a: 1, b: 1, c: 1})
```




```
db.test.find({a: 2})
    .sort({b: -1, c: 1})
```

```
db.test.find({a: 2})
    .sort({b: -1})
```

mongoDB

Index Practice

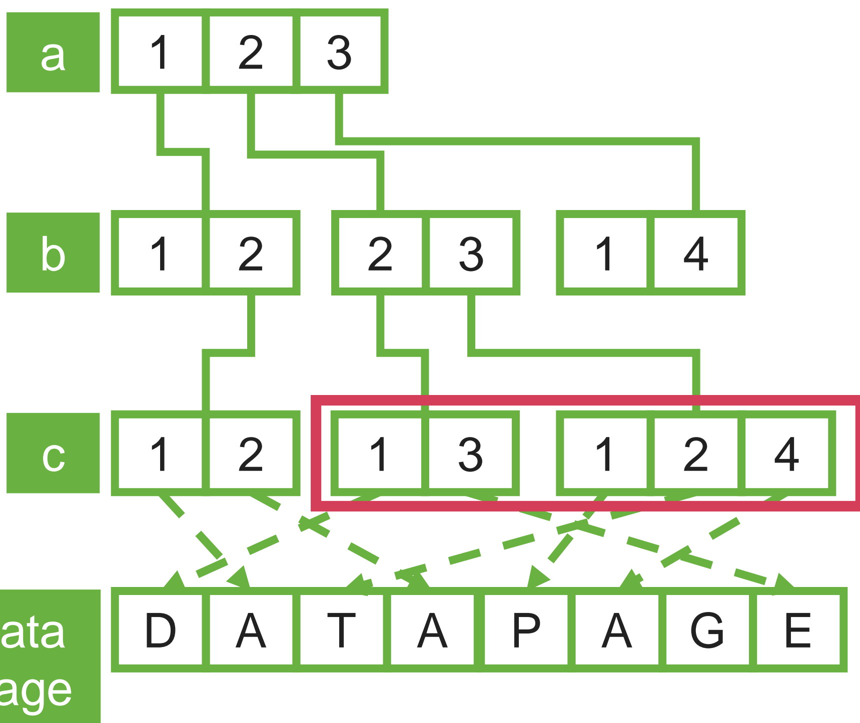
复合索引的工作模式——索引顺序的影响

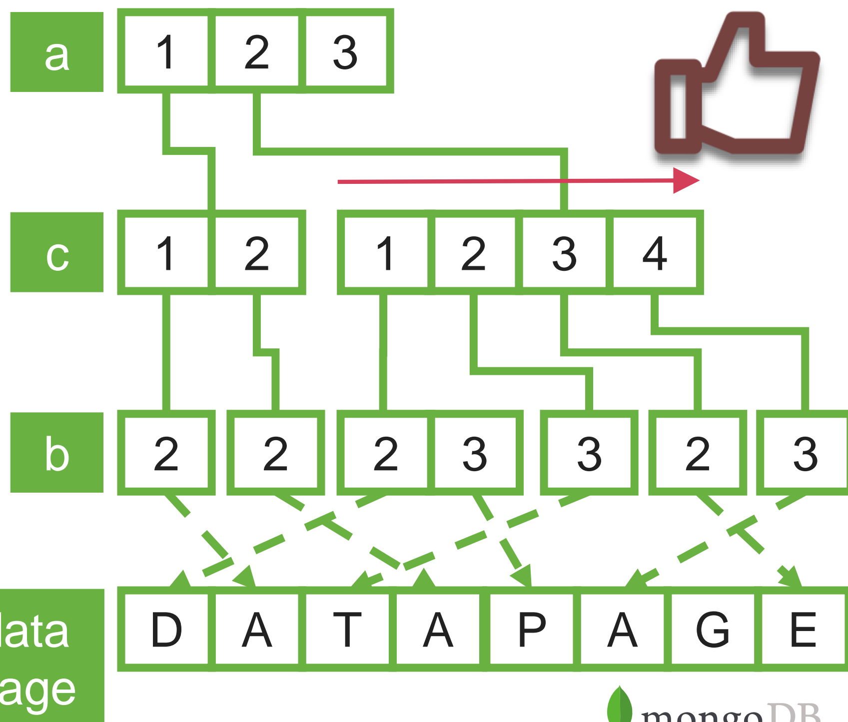
```
db.test.find({a: 2, b: {$gte: 2, $lte: 3}}).sort({c: 1})
```

```
db.test.createIndex({a: 1, b: 1, c: 1})
```

VS

```
db.test.createIndex({a: 1, c: 1, b: 1})
```





FAQ

- 命中了索引是不是一定比较快？
  - 有例外
  - 索引与索引比谁更快？
- 我需要查询按{A: 1, B: 1, C: 1}查询。建立索引{A: 1, B: 1, C: 1}，为什么系统没有使用这个索引？
  - A、B、C谁能过滤更多数据？
  - A、B、C谁是精确匹配，谁是范围匹配？
  - 是否涉及到排序？
- 系统根据什么决定使用哪个索引？
  - 看谁跑得快

Conclusions

- 如果前面的都没听懂，请记住这个结论：

建立联合索引的正确姿势是：
- 精确匹配
- 排序条件
- 范围匹配

同为精确匹配或范围匹配的时候，由过滤性决定优先级

没有了
Q&A

mongoDB

mongoDB

Graphic Element Examples

LOREM IPSUM

LOREM IPSUM

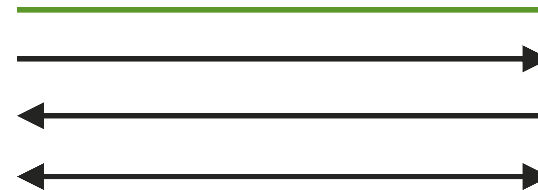
LORE M IPSUM

LORE M IPSUM

Sollicitu din	Venenati s

LOREM IPSUM

LOREM IPSUM

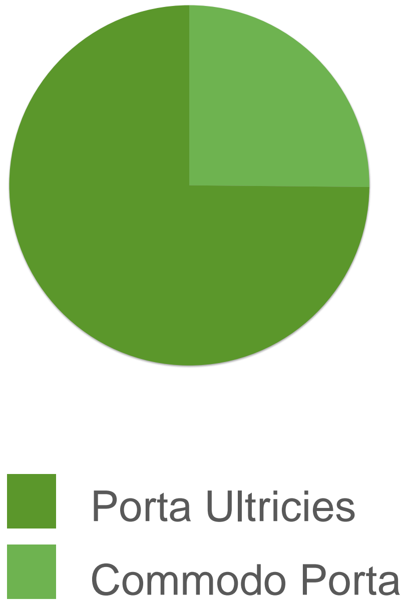


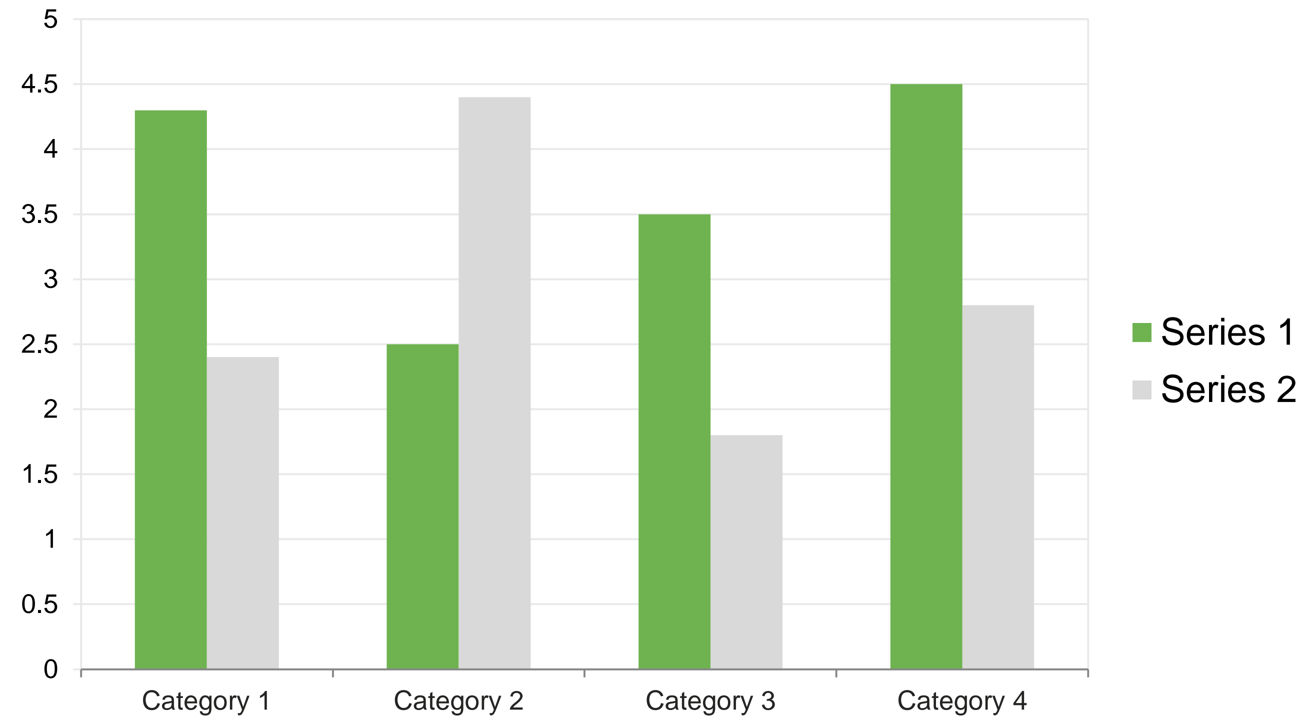
LORE M IPSUM

LORE M IPSUM

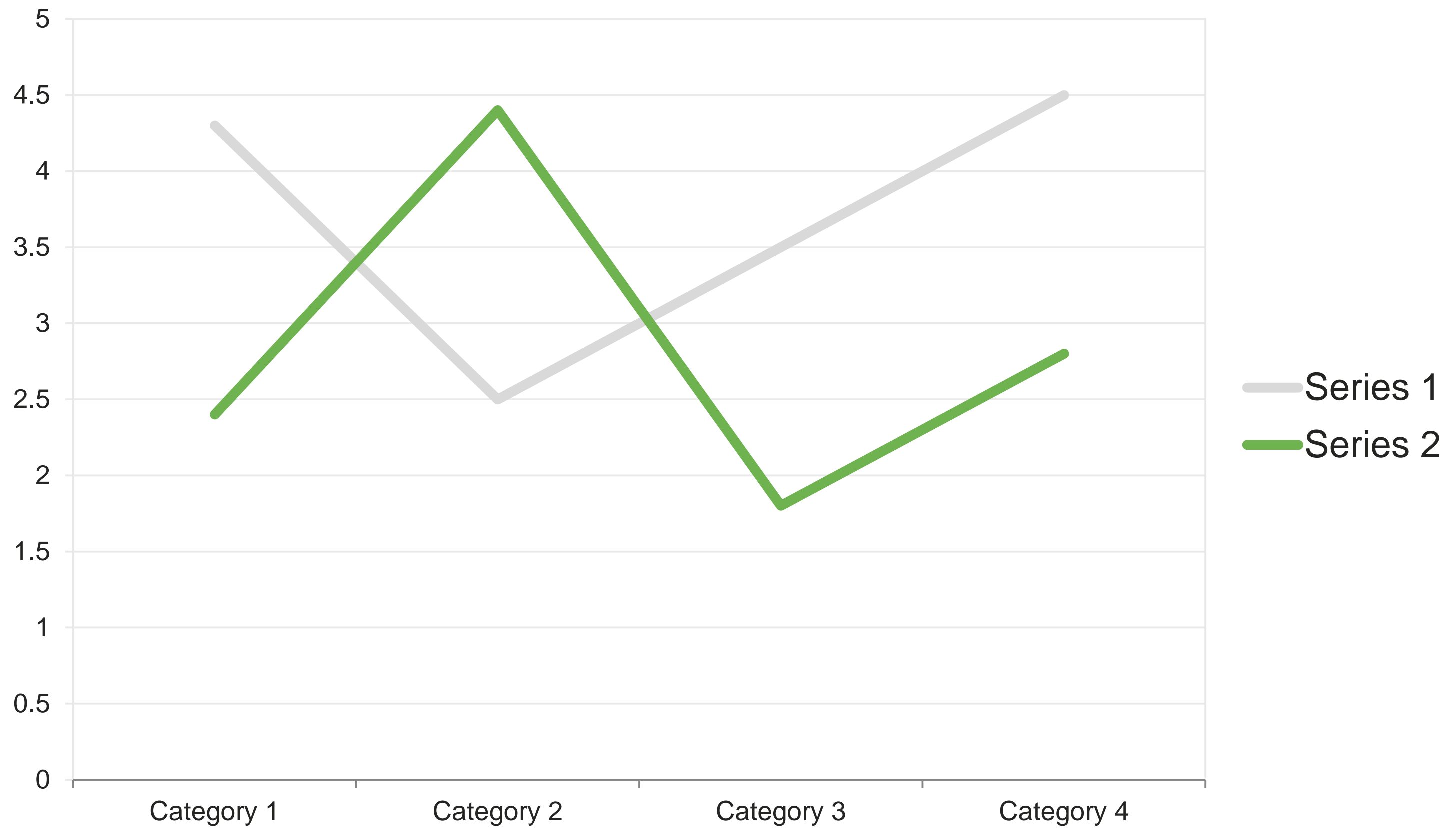
mongoDB

Graph Examples





mongoDB



mongoDB

Code/Highlight Example

```
{
    _id : ObjectId("4c4ba5e5e8aabf3"),
    employee_name: "Dunham, Justin",
    department : "Marketing",
    title : "Product Manager, Web",
    report_up: "Neray, Graham",
    pay_band: “C",
    benefits : [
            {  type :  "Health",
               plan : "PPO Plus" },
            {  type :   "Dental",
               plan : "Standard" }
             ]
}
```

mongoDB