aggregations

Adrien Grand
@jpountz

elasticsearch.

outline

- what aggregations are
- why we built them
- how they work

  what the trade-offs are

aggregations

- analytics
  histograms, distributions, statistics
- over any partition of your data
  anything that can be selected with queries/filters
- in near real time
  computed on the fly, ~1s refresh interval
- that can be composed
  unlike facets

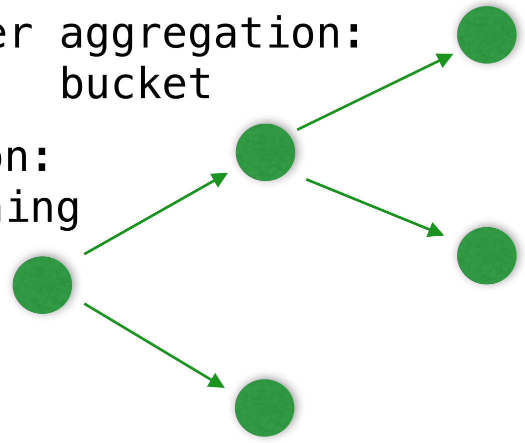
bucket / metrics

- bucket

  terms
  histogram
  range
  filter
  geohash grid

- metrics

  stats
  min / max / avg / sum
  percentiles
  cardinality



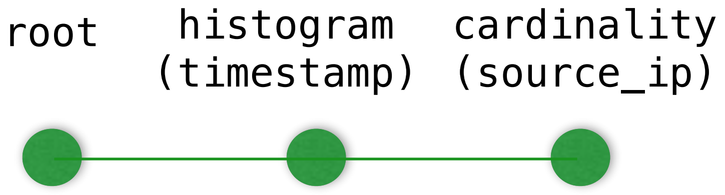
leaf aggregation:
bucket or metric

inner aggregation:
bucket

root aggregation:
collects everything

traffic analysis

```
{
  “source_ip" : “77.104.12.13”,
  “timestamp” : “2014-05-25T23:44:12.779Z”
}
```

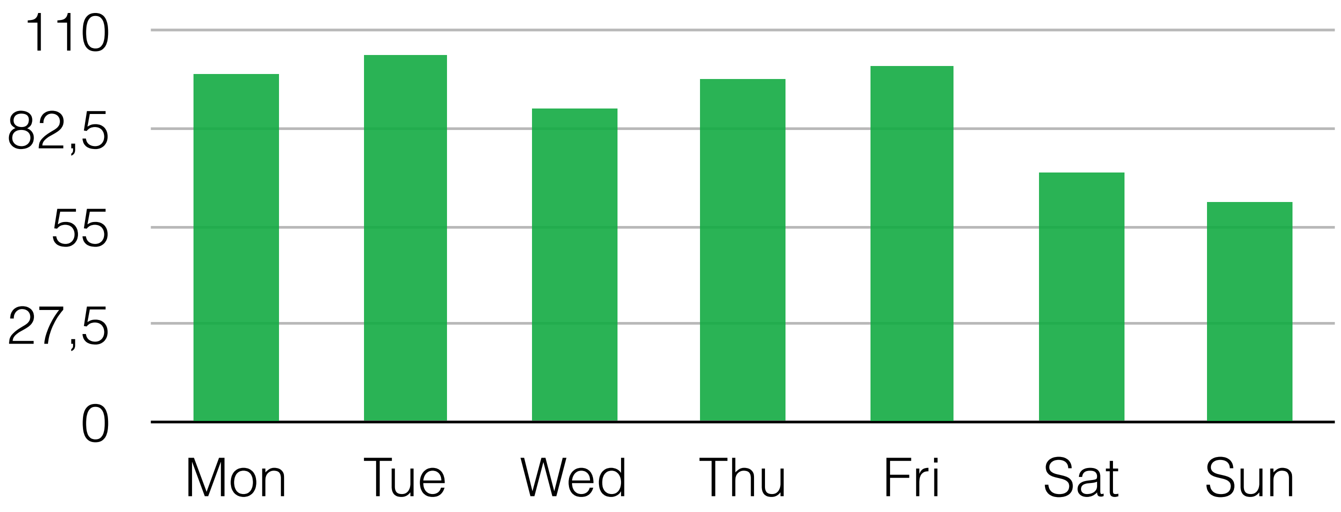


root

histogram
(timestamp)

cardinality
(source_ip)

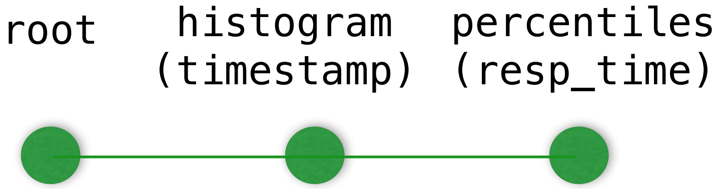
Unique visitors per day



elasticsearch.

performance analysis

```
{
  “resp_time” : 205,
  “timestamp” : “2014-05-25T23:44:12.779Z”
}
```

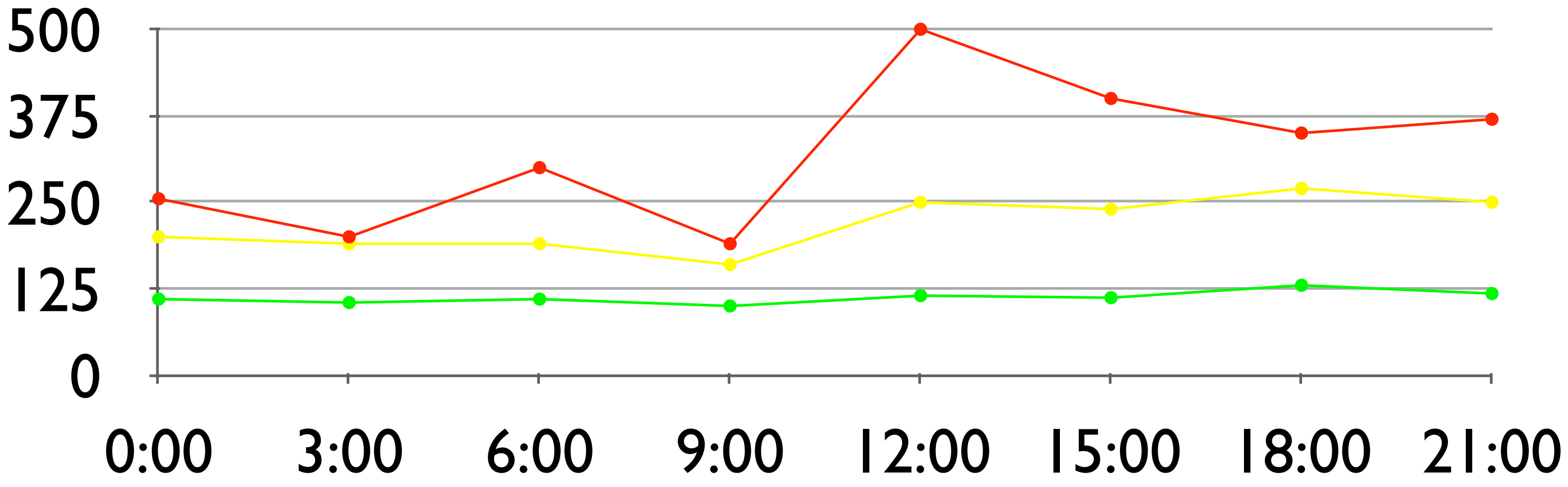


root

histogram
(timestamp)

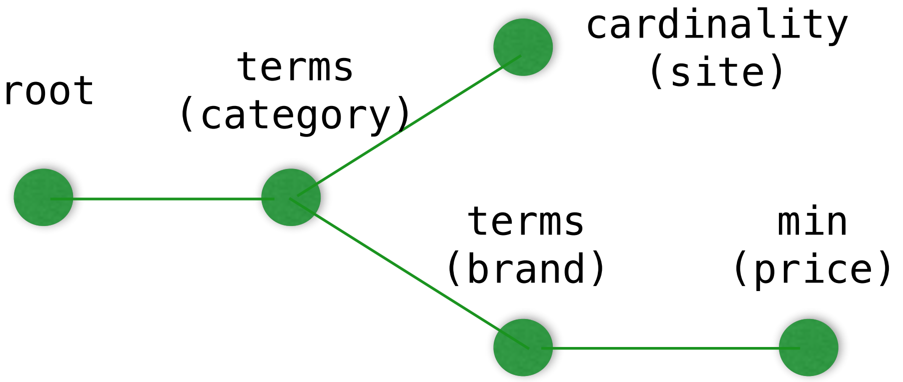
percentiles
(resp_time)

Median, 90th, 99th percentiles over time



e-commerce

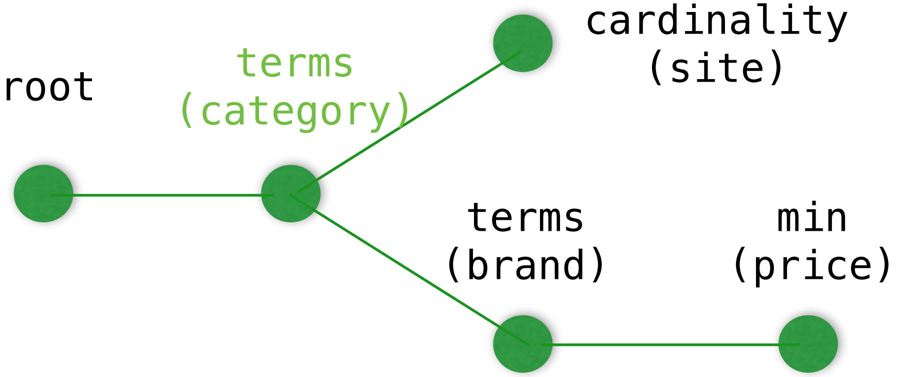
```
{
    “category” : “Dresses”,
   “site” : “Zalando”,
    “brand” : “Desigual”,
    “designation”: “dress”,
    “price”: 85
}
```



- Dresses: 23 offers, 9 sites
    - Urbanist: 12   min_price: 60
    - Desigual: 8   min_price: 85
    - Life: 3   min_price: 52
- Shoes: 19, 3 sites
- Skirts: 8, 5 sites

e-commerce

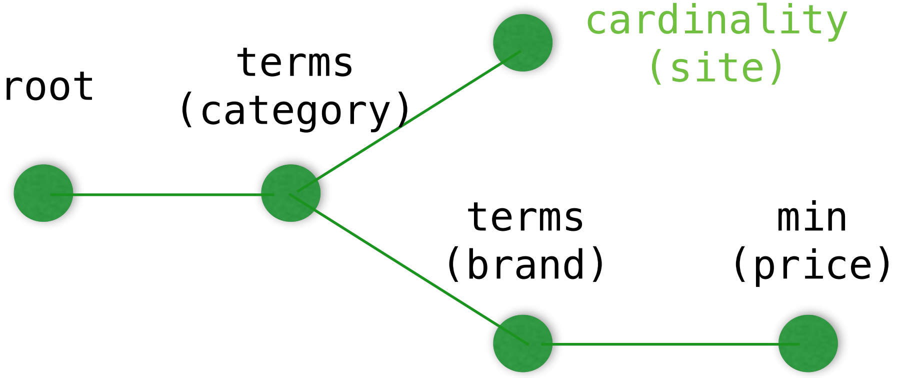
```
{
    “category” : “Dresses”,
   “site” : “Zalando”,
    “brand” : “Desigual”,
    “designation”: “dress”,
    “price”: 85
}
```



- Dresses: 23 offers, 9 sites
    - Urbanist: 12 min_price: 60
    - Desigual: 8 min_price: 85
    - Life: 3 min_price: 52
- Shoes: 19, 3 sites
- Skirts: 8, 5 sites

e-commerce

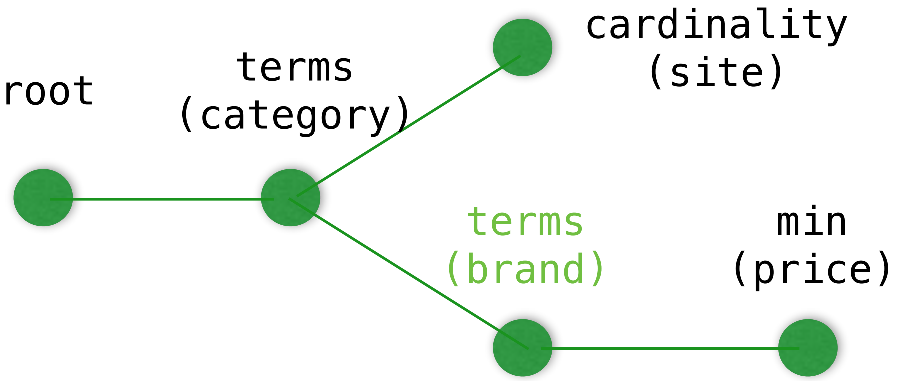
```
{
    “category” : “Dresses”,
    “site” : “Zalando”,
    “brand” : “Desigual”,
    “designation”: “dress”,
    “price”: 85
}
```



- Dresses: 23 offers, 9 sites
    - Urbanist: 12   min_price: 60
    - Desigual: 8   min_price: 85
    - Life: 3   min_price: 52
- Shoes: 19, 3 sites
- Skirts: 8, 5 sites

e-commerce

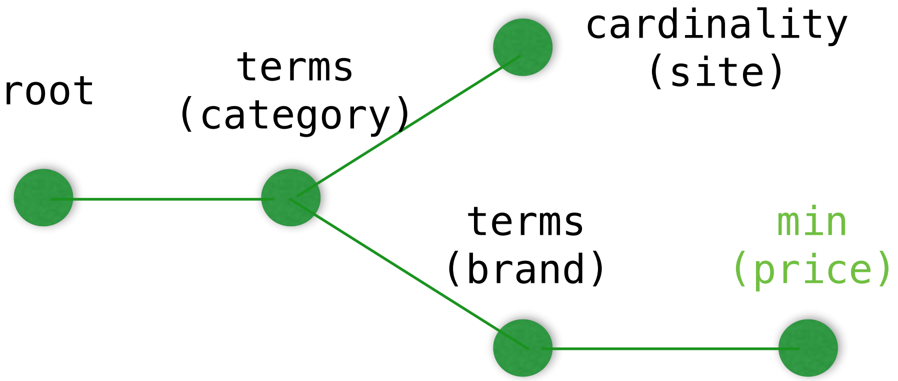
```
{
    “category” : “Dresses”,
   “site” : “Zalando”,
    “brand” : “Desigual”,
    “designation”: “dress”,
    “price”: 85
}
```



```
• Dresses: 23 offers, 9 sites

    • Urbanist:  12    min_price: 60
    • Desigual:   8    min_price: 85
    • Life:       3    min_price: 52


• Shoes: 19, 3 sites


• Skirts: 8, 5 sites
```

e-commerce

```
{
    “category” : “Dresses”,
   “site” : “Zalando”,
    “brand” : “Desigual”,
    “designation”: “dress”,
    “price”: 85
}
```



```
• Dresses: 23 offers, 9 sites

    • Urbanist:  12    min_price: 60
    • Desigual:   8    min_price: 85
    • Life:       3    min_price: 52


• Shoes: 19, 3 sites


• Skirts: 8, 5 sites
```

why on elasticsearch?

- powerful when combined with search
  data exploration

- search engines have had faceted search for a very long time
  storage is optimized for such a workload

- aggregations are a new iteration
  with increased capabilities / flexibility

why is it fast?

- data stored to make information retrieval fast

  yet indexing remains faster than what you expect

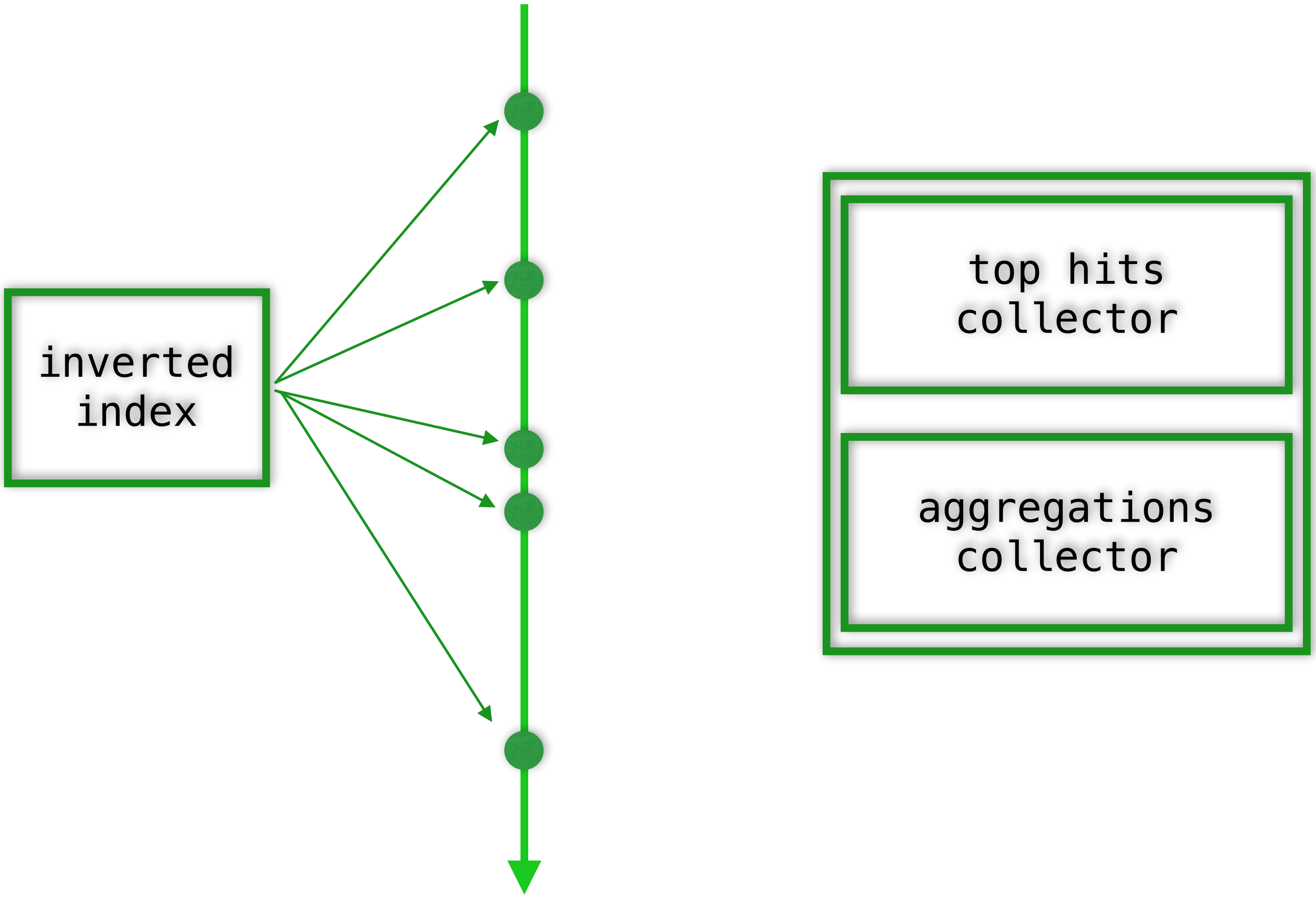
- optimized data structures

  compressed columnar storage (field data / doc values)
  strings are enums (per segment)

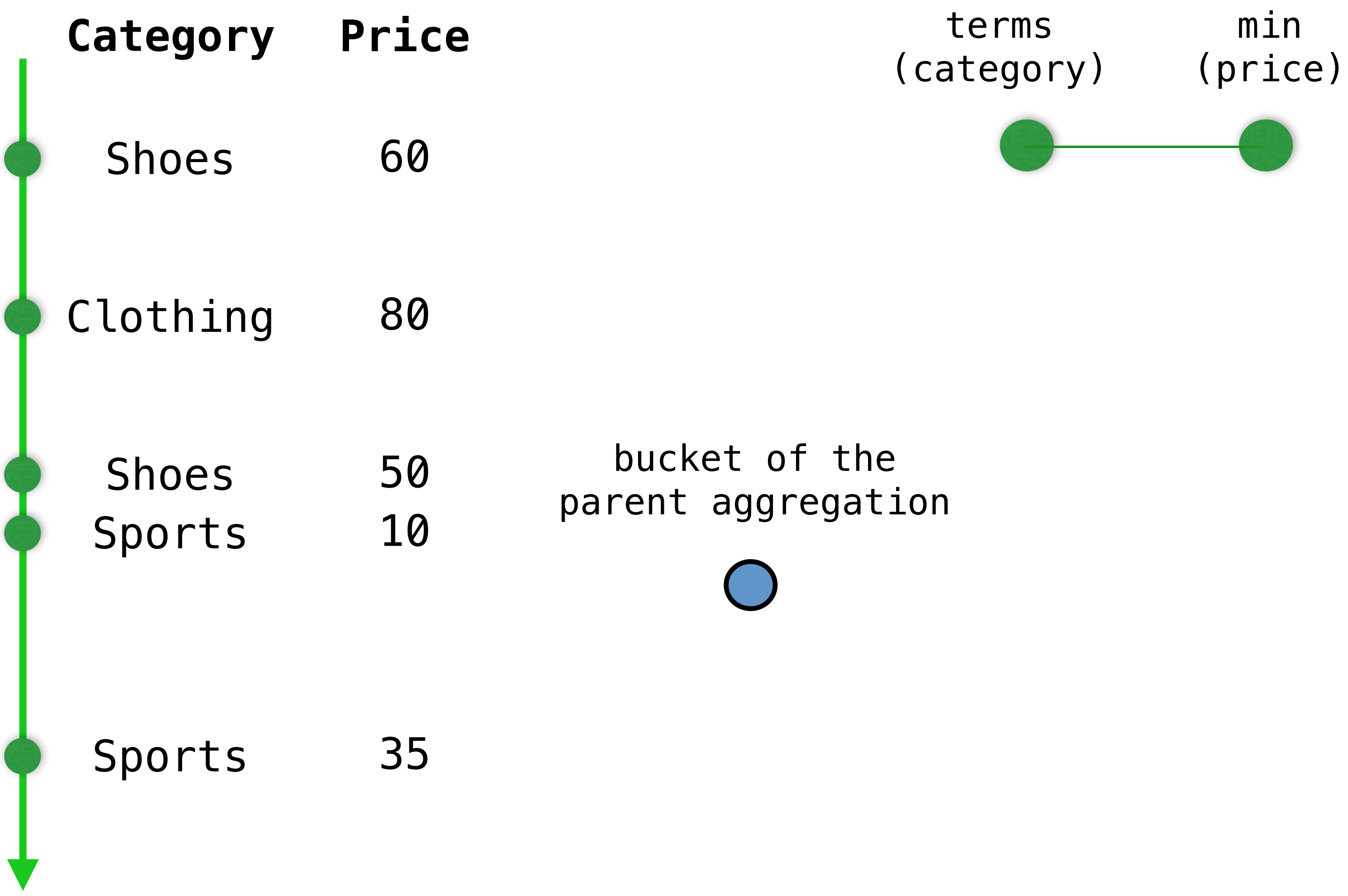
- single pass on your data

  no matter how many levels of aggregations there are

how it works (shard level)



how it works (shard level)



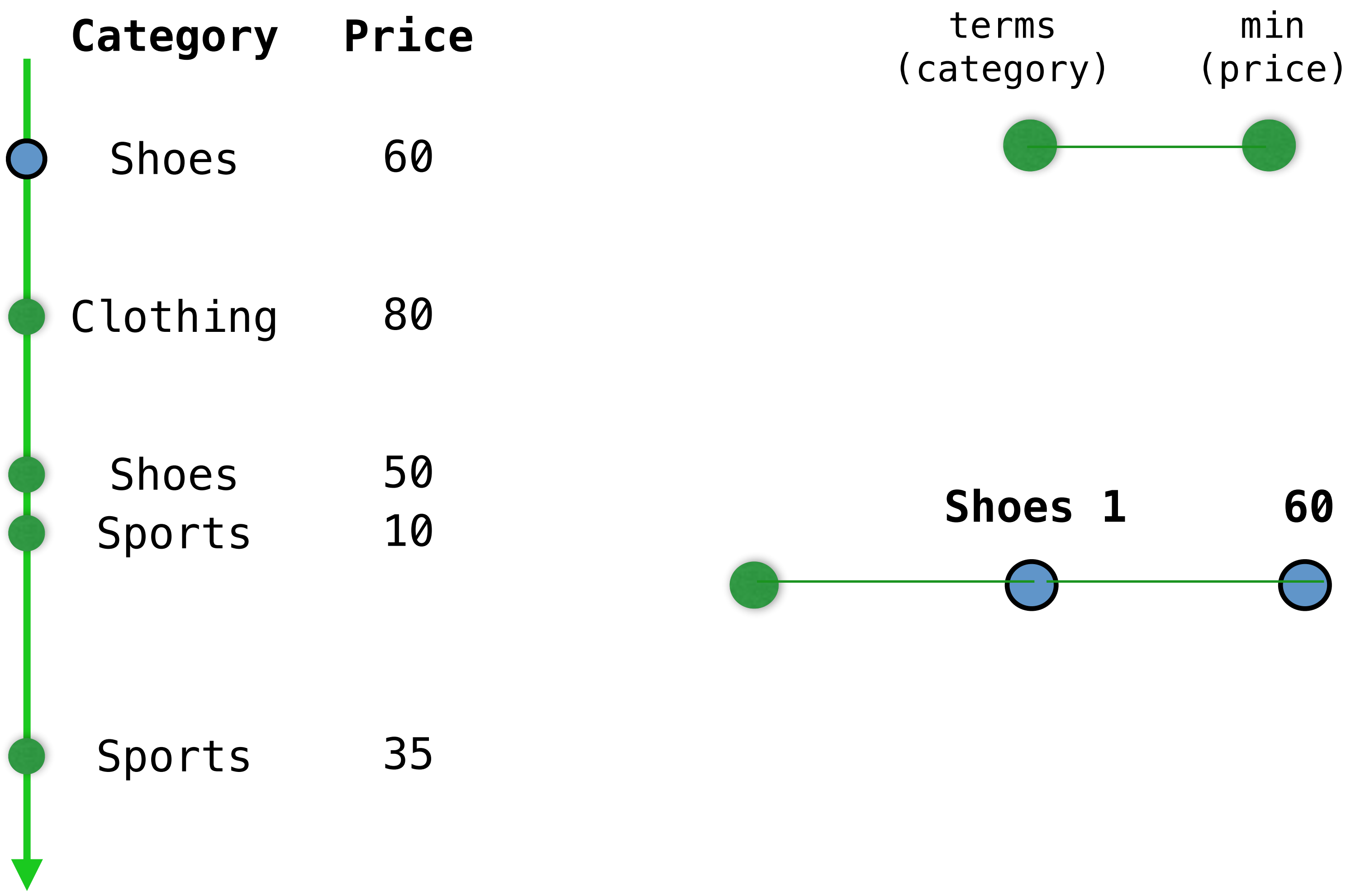
Category	Price
Shoes	60
Clothing	80
Shoes	50
Sports	10
Sports	35

terms (category)

min (price)

bucket of the parent aggregation

how it works (shard level)



Category	Price
Shoes	60
Clothing	80
Shoes	50
Sports	10
Sports	35

terms
(category)

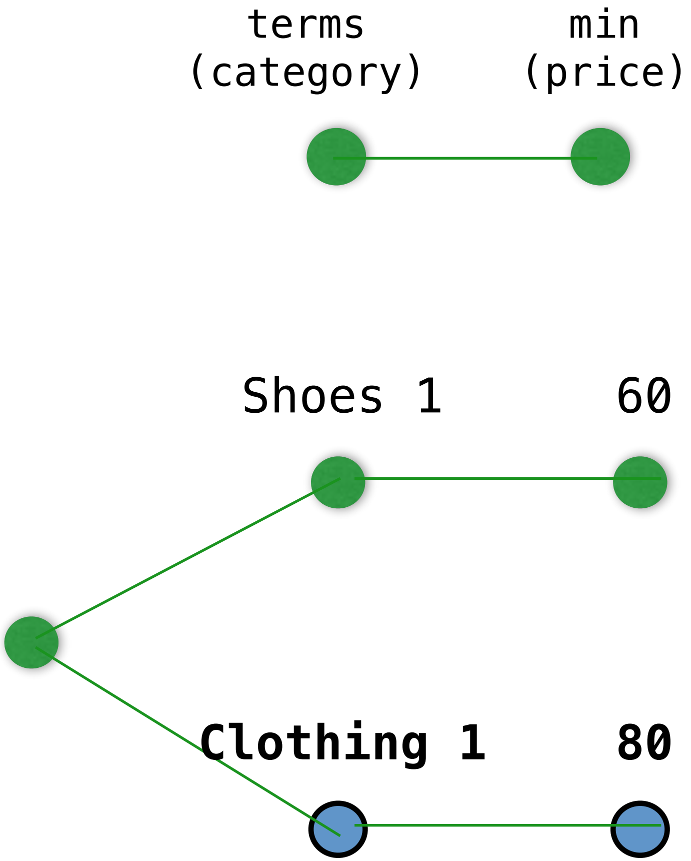
min
(price)

Shoes 1

60

how it works (shard level)

Category	Price
Shoes	60
Clothing	80
Shoes	50
Sports	10
Sports	35



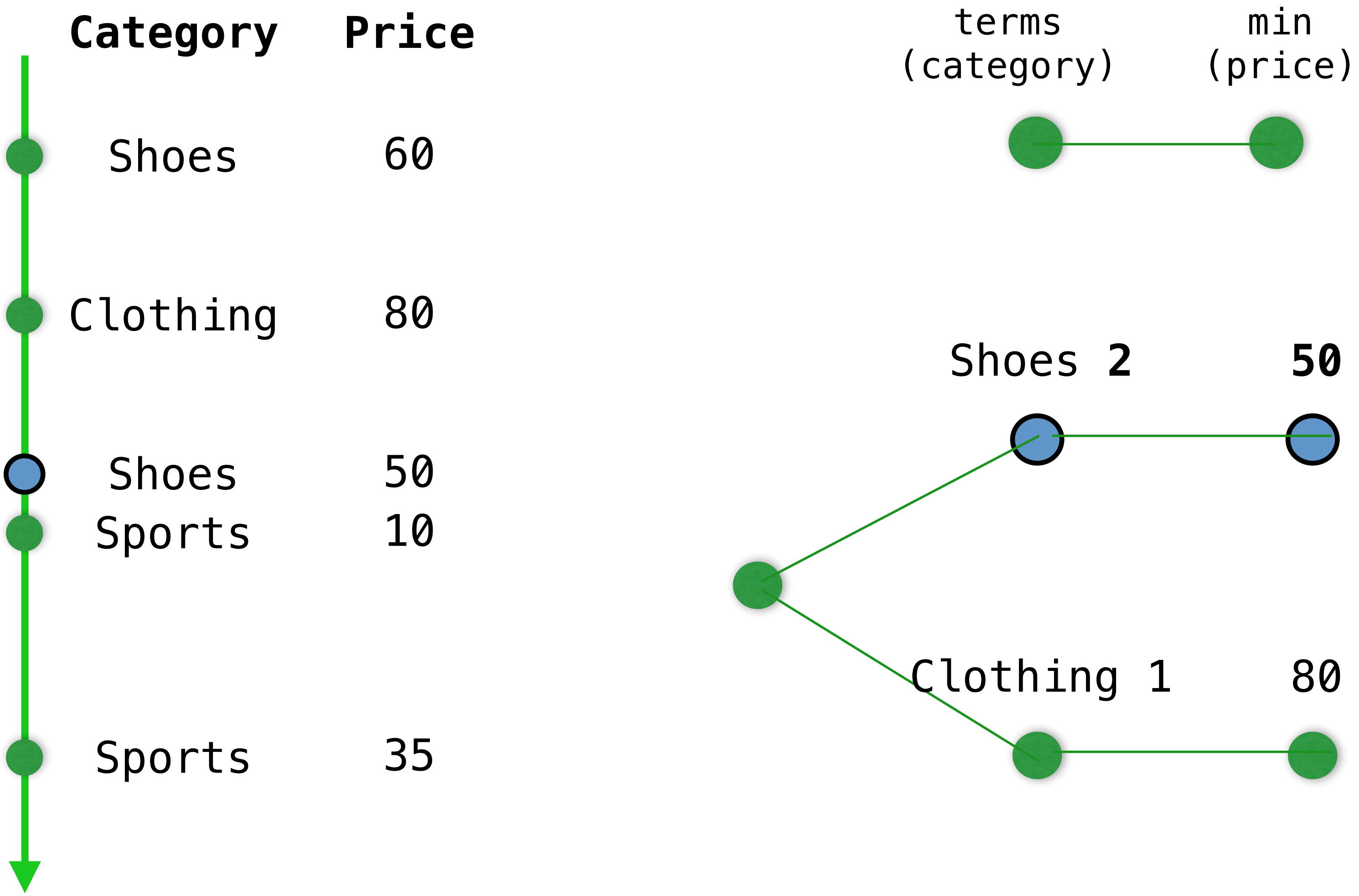
terms
(category)

min
(price)

Shoes 1 60

Clothing 1 80

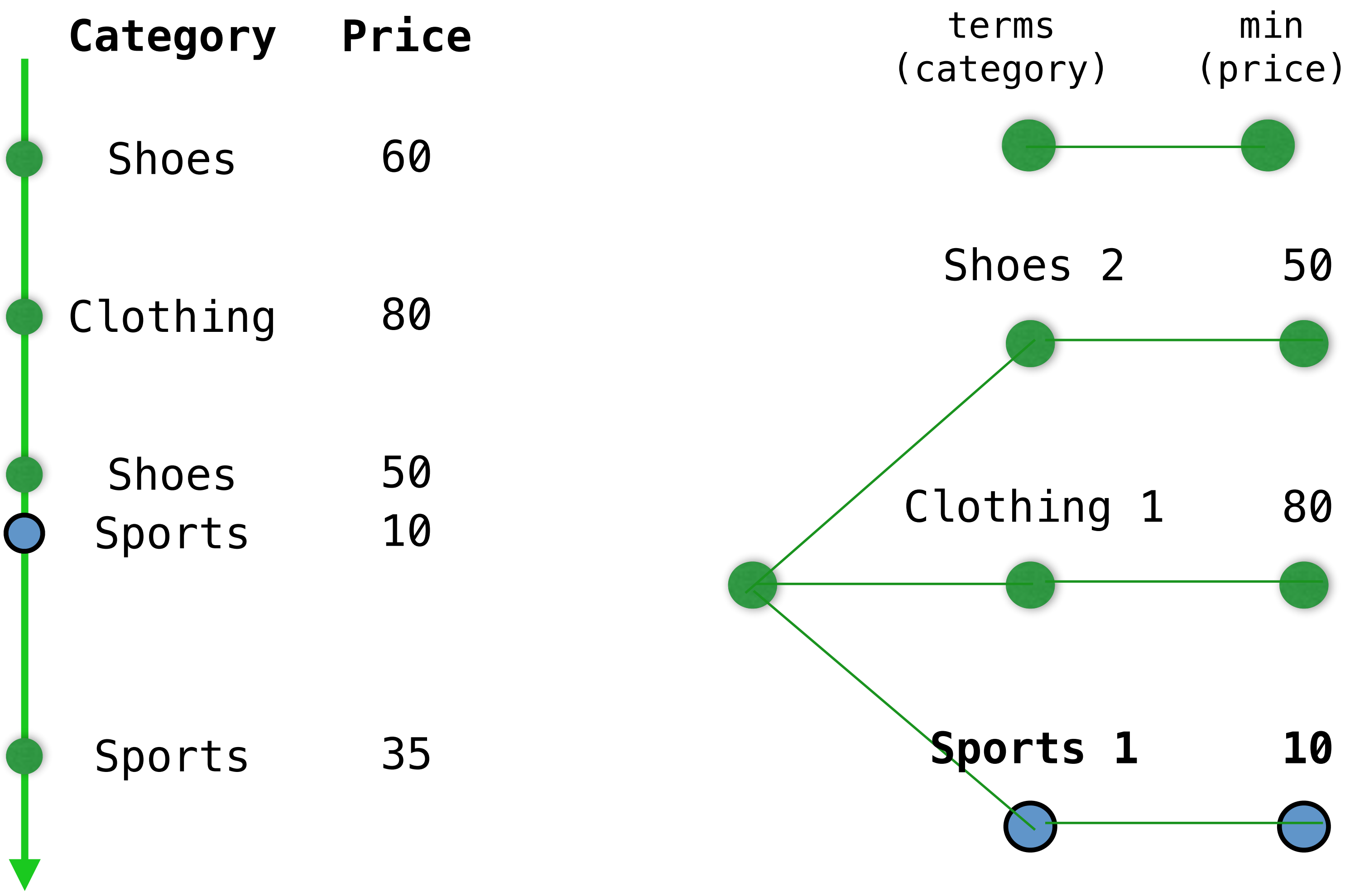
how it works (shard level)



Category	Price
Shoes	60
Clothing	80
Shoes	50
Sports	10
Sports	35

terms (category)	min (price)
Shoes **2**	**50**
Clothing 1	80

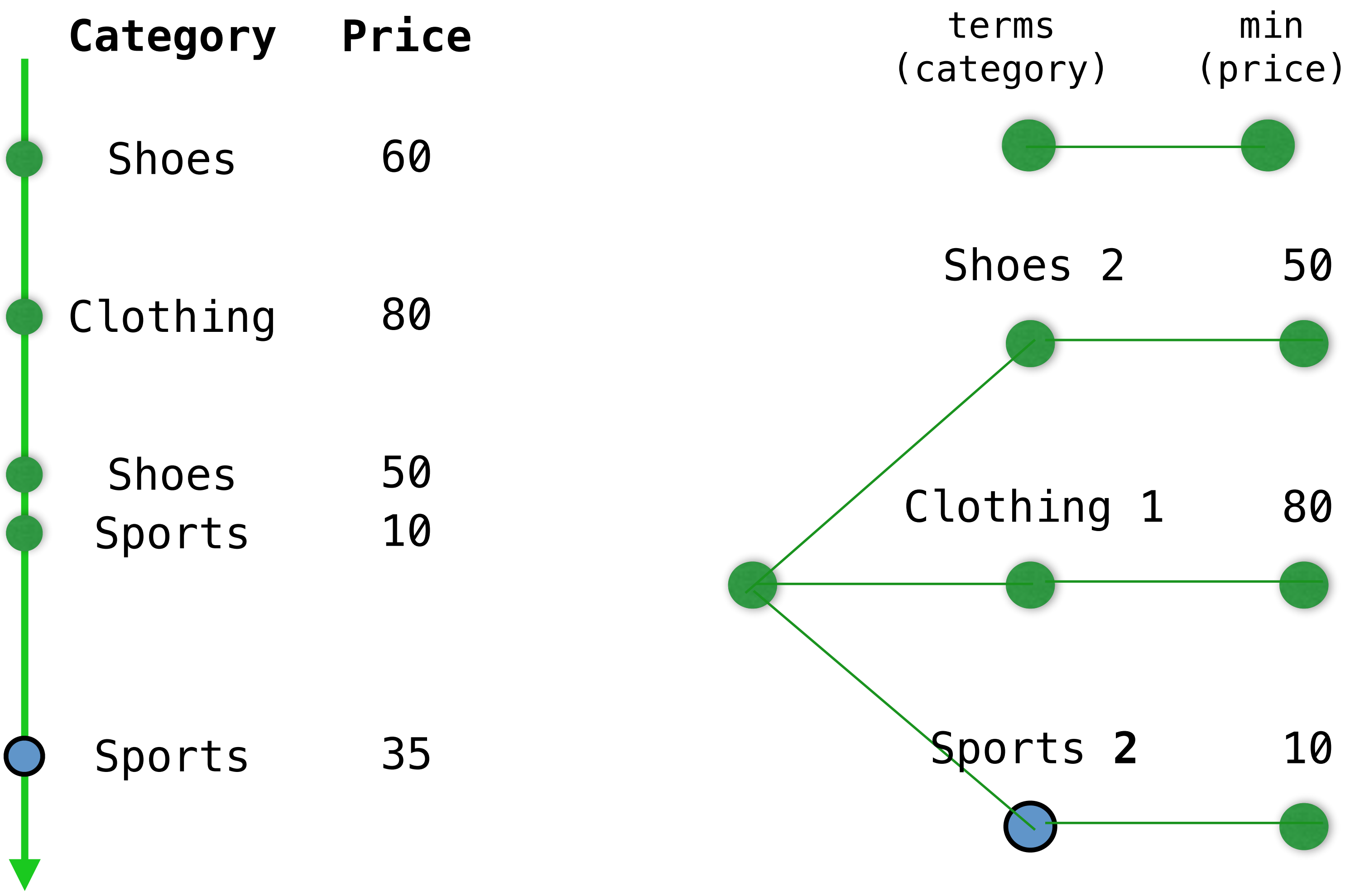
how it works (shard level)



Category	Price
Shoes	60
Clothing	80
Shoes	50
Sports	10
Sports	35

terms (category)	min (price)
Shoes 2	50
Clothing 1	80
Sports 1	**10**

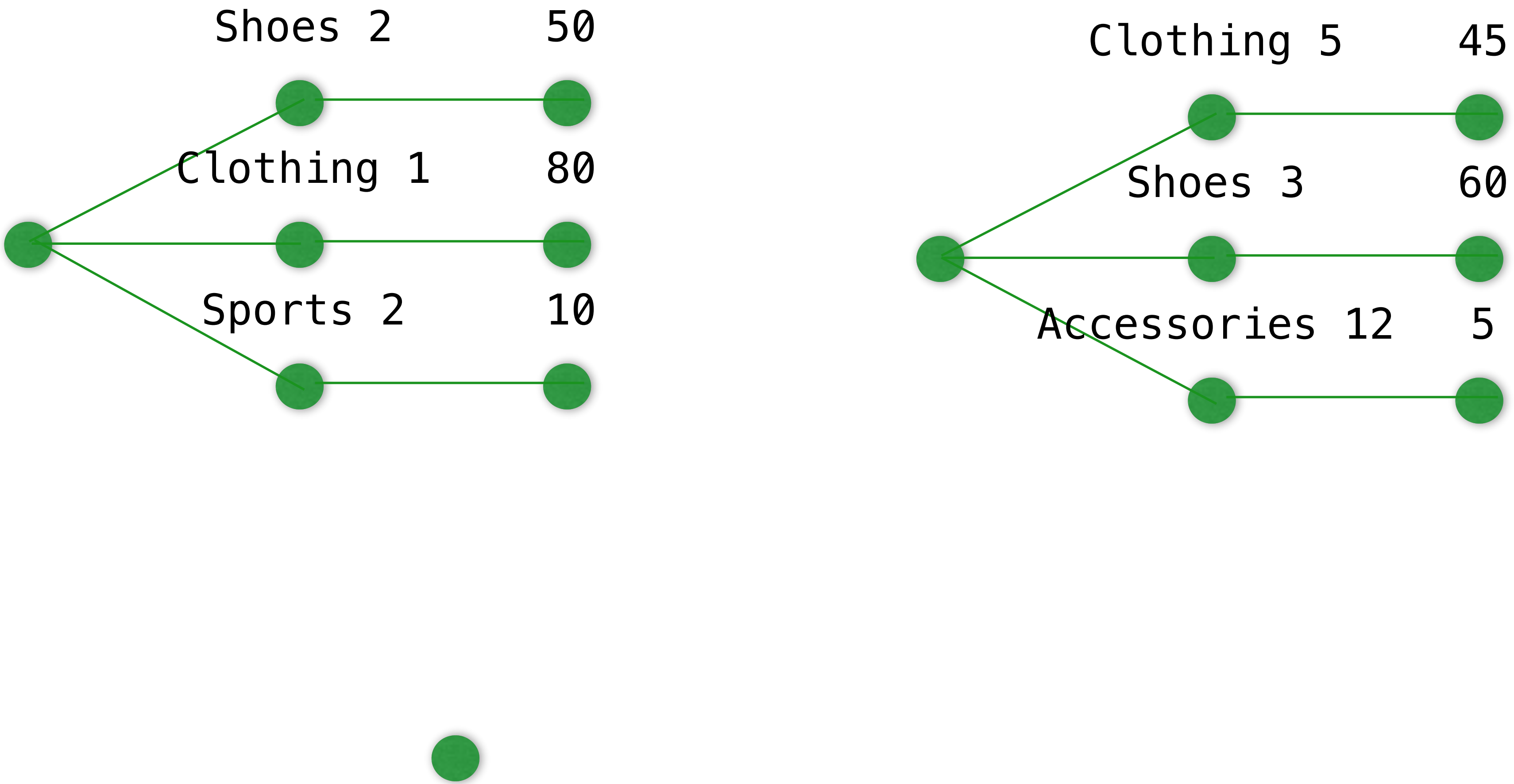
how it works (shard level)



Category	Price
Shoes	60
Clothing	80
Shoes	50
Sports	10
Sports	35

terms (category)	min (price)
Shoes 2	50
Clothing 1	80
Sports **2**	10

how it works (cluster level)



Shoes 2 50

Clothing 1 80

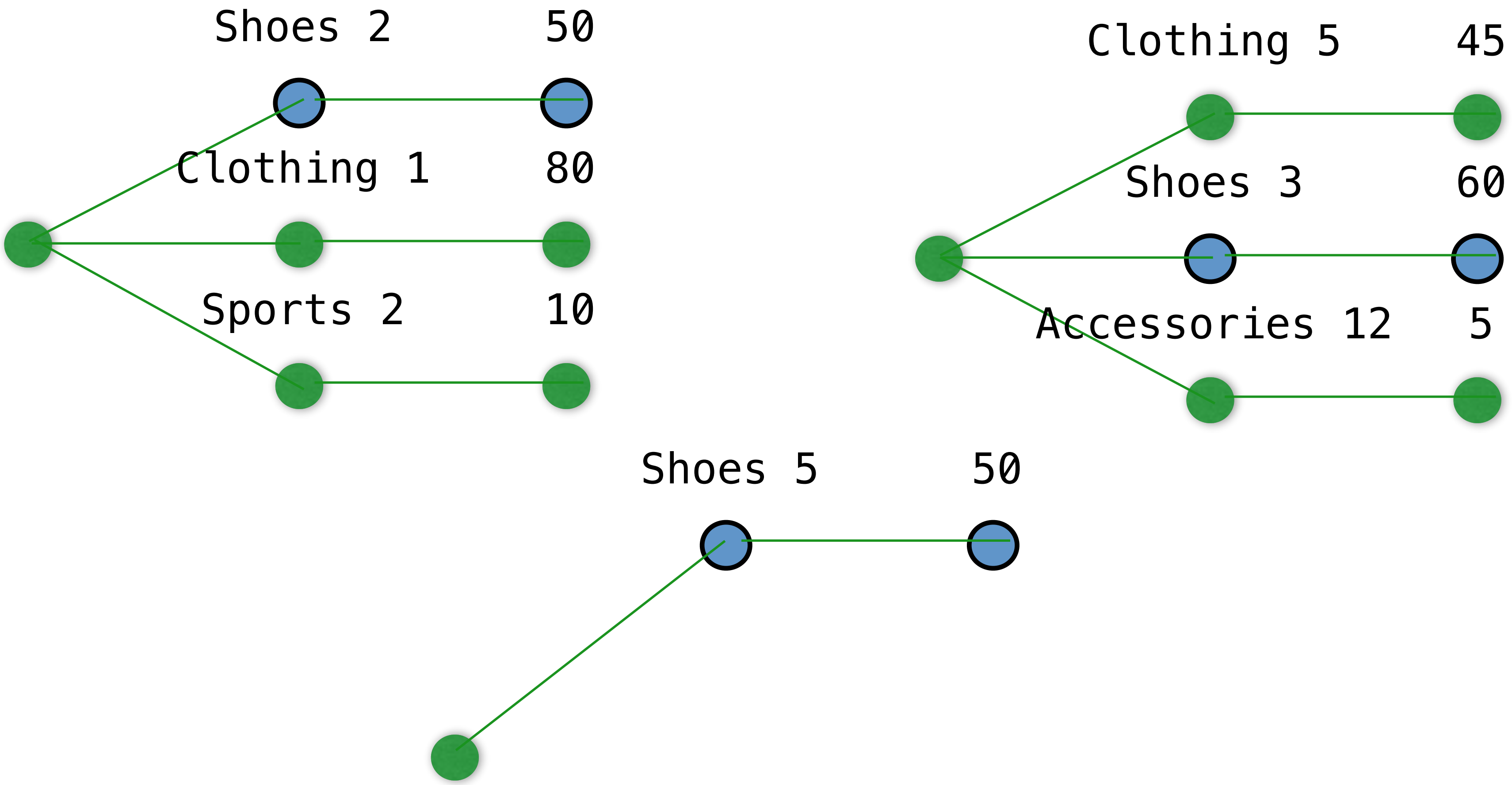
Sports 2 10

Clothing 5 45

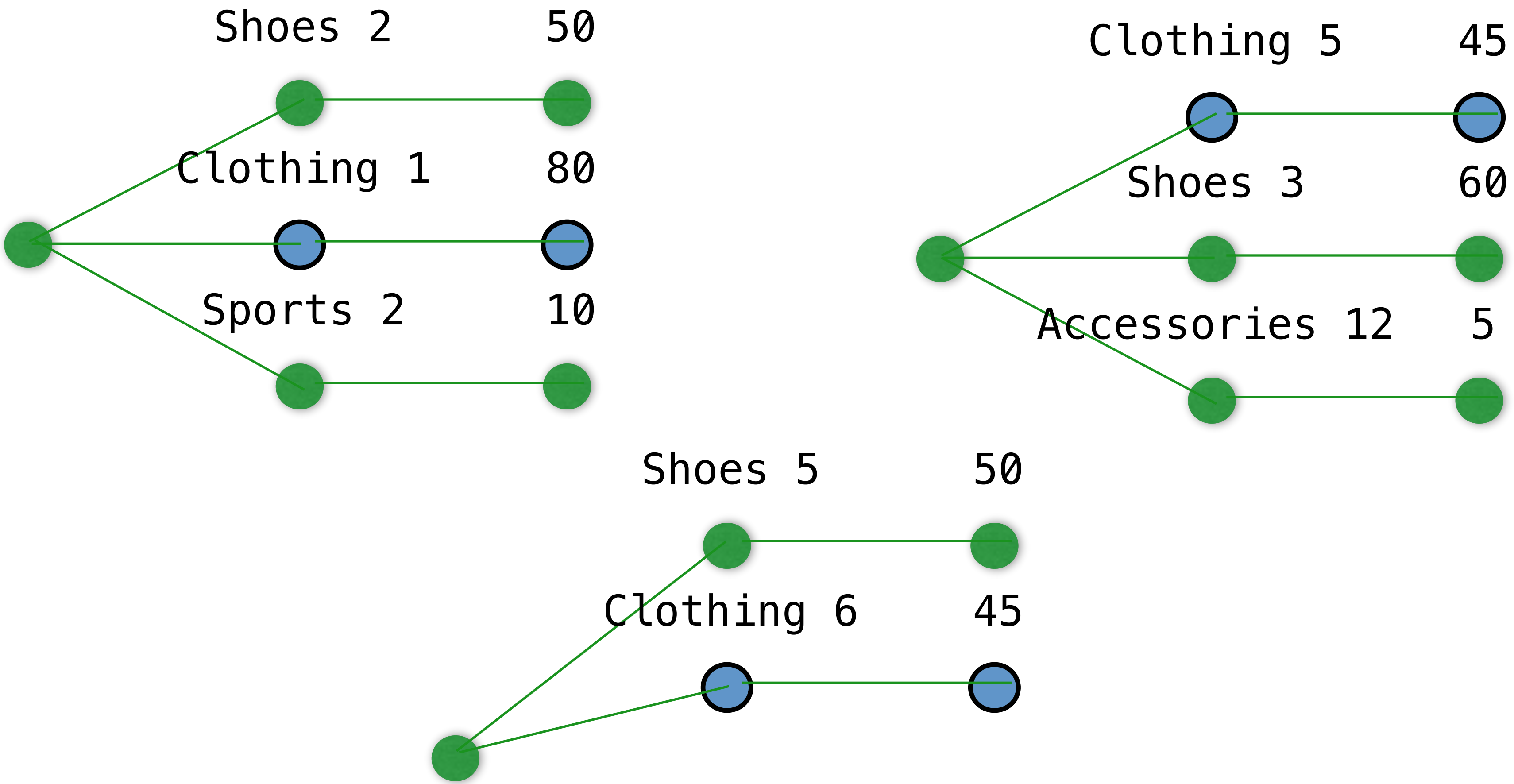
Shoes 3 60

Accessories 12 5

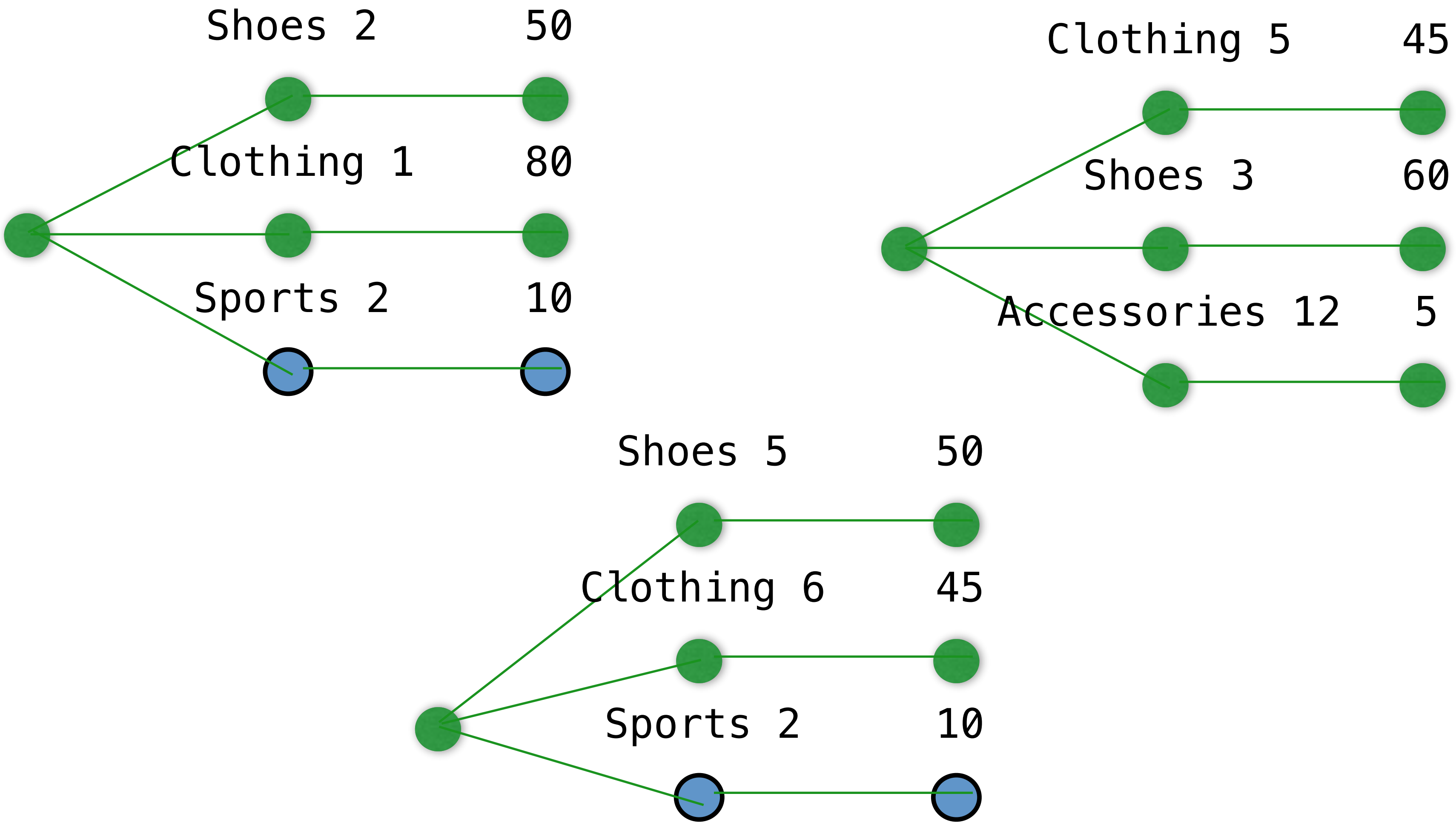
how it works (cluster level)



how it works (cluster level)

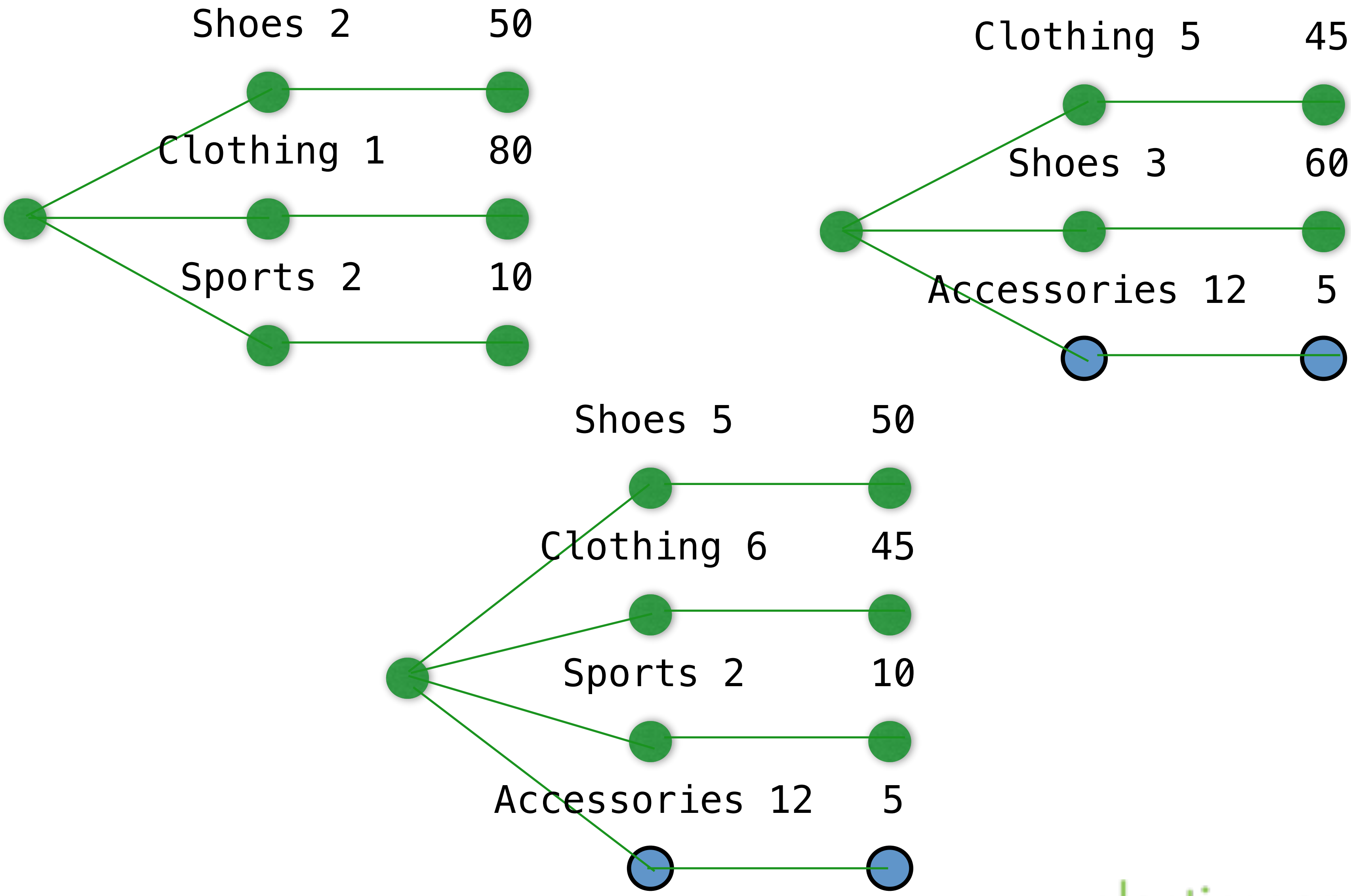


how it works (cluster level)



elasticsearch.

how it works (cluster level)



Shoes 2 50

Clothing 1 80

Sports 2 10

Clothing 5 45

Shoes 3 60

Accessories 12 5

Shoes 5 50

Clothing 6 45

Sports 2 10

Accessories 12 5

goodies

- support for document relations

  via nested documents and the nested/reverse_nested aggs

  no parent/child support (yet?)

- significant_terms

  find the uncommonly common

- upcoming top_hits aggregations in 1.3

  compute top hits on each bucket

- performance / memory usage improved in 1.2

  Upgrade if you rely on aggregations

thank you!