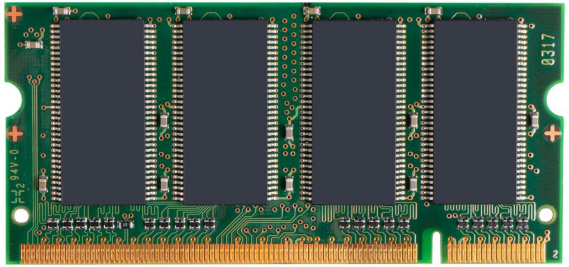
MDBW 2022

Under the Hood

Supercharging Atlas Data Lake

David Golden
Staff Engineer
@xdg


NOW
RENTING
MetLife
MetLife
MetLife




BOX #
CONTENTS
DATE


LAKE ONTARIO
BUFFALO
EMPIRE
PENNSYLVANIA

Welcome!

David Golden, Staff Engineer

david@mongodb.com

twitter: @xdg

Welcome!

David Golden, Staff Engineer

david@mongodb.com twitter: @xdg

My team…

- Data Federation
- Data Lake Storage

Today…

3

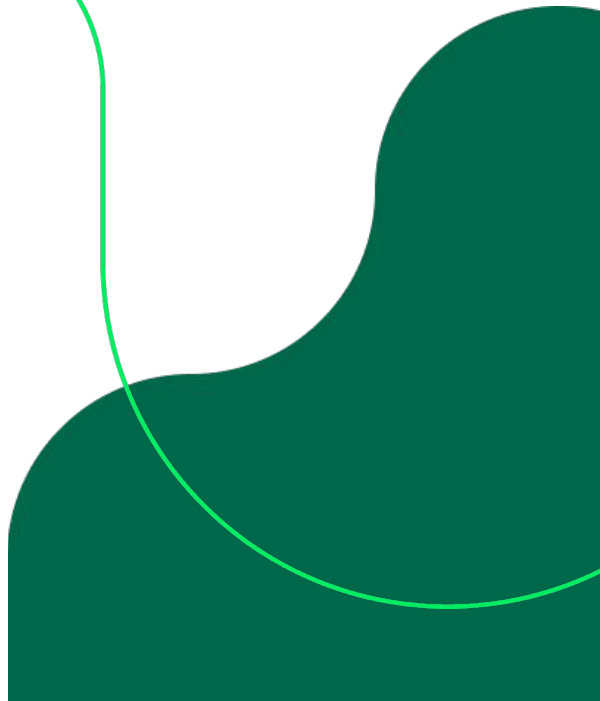
Traps

Principles

Changes

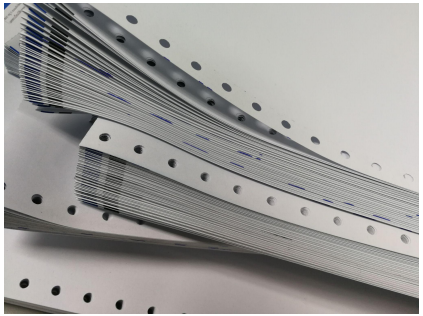
Trap #1:

Tiny files

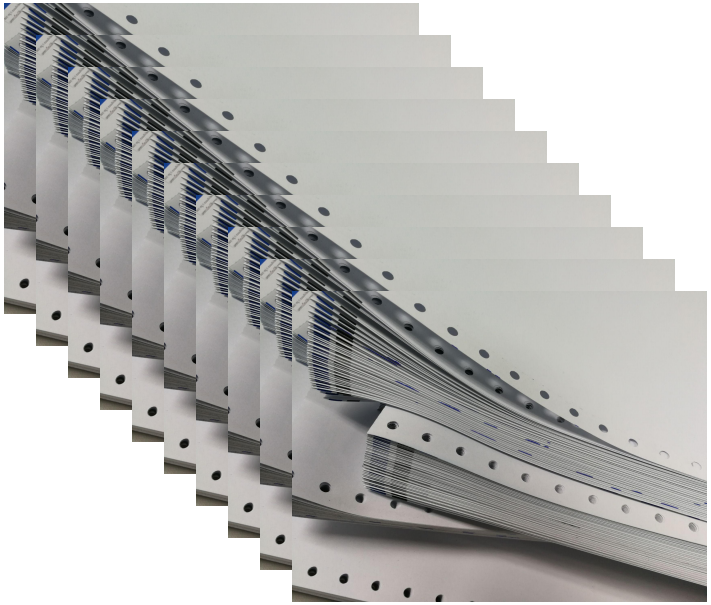


What's in the box?

One more problem…



→ 1,000



Why?

Folder → bad metaphor

`s3://<bucket>/<key>`

Key → anything

7a4f37e9739c4e7a8238a7c67fd9f46e1a48870921144e348c3fdd7fba771

`7a4f37e9739c4e7a8238a7c67fd9f46e1a48870921144e348c3fdd7f.json`

`7a4f37e9739c4e7a8238a7c67fd9f/6e1a48870921144e348c3fdd7f.json`

`7a4f37e9739c4e7a8238a7c67fd9f/6e1a48870921144e348c3fdd7f.json`

Folder? File?

`sales/e9739c4e7a8238a7c67fd9f/6e1a48870921144e348c3fdd7f.json`

`sales/<uploadDate>/8a7c67fd9f/6e1a48870921144e348c3fdd7f.json`

`sales/<uploadDate>/<regionID>/<storeID>/21144e348c3fdd7f.json`

`sales/<uploadDate>/<regionID>/<storeID>/<productID>/dd7f.json`

```
sales/<uploadDate>/<regionID>/<storeID>/<productID>/0001.json
```

Folders → tiny files

`sales/<uploadDate>/<regionID>/<storeID>/<productID>/????.json`

Folders → pay twice!

Worst case

`sales/<uploadDate>/<regionID>/<storeID>/<transactionID>.json`

Unique! → single document file

Principle #1

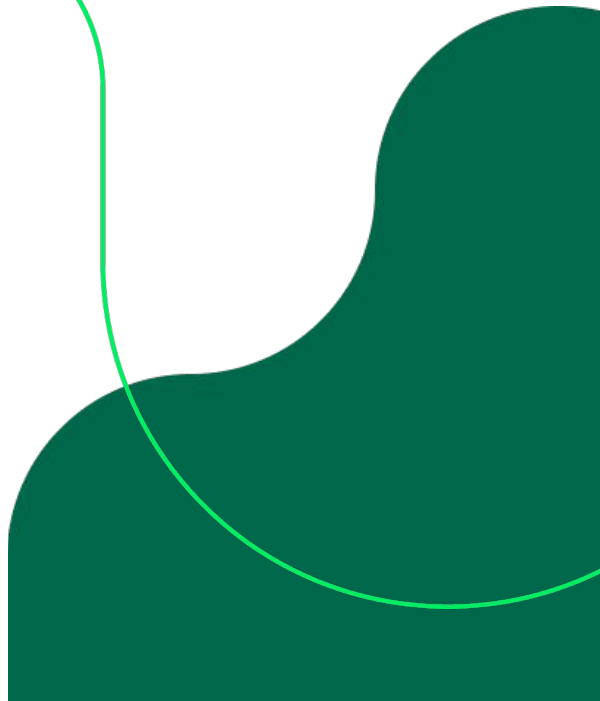
Minimize number of network trips

Big files

(but not too big)

Trap #2:

Wasteful scans



Imagine a tweet data lake

Analyze tweets by @xdg

Assume folders…

Assume daily uploads…

File per screen name?

`tweets/2022-05-12/xdg.json`

How much did I tweet that day?

Roll up by date?

```
tweets/2022-05-12/0001.json
                  ⋮
tweets/2022-05-12/0847.json
```

But which file has "xdg"?

Principle #2

Minimize fetching unneeded documents

Use file bounds?

```
tweets/2022-05-12/abba-cher.json
                  ⋮
tweets/2022-05-12/weezer-zendaya.json
```

Only the last file has "xdg"

Analyze a whole month?

Daily files with bounds

```
tweets/2022-05-01/wyclef-ziggymarley.json
                  ⋮
tweets/2022-05-12/weezer-zendaya.json
tweets/2022-05-13/wutangclan-zztop.json
```

"xdg" is a tiny part of every file

Roll-up

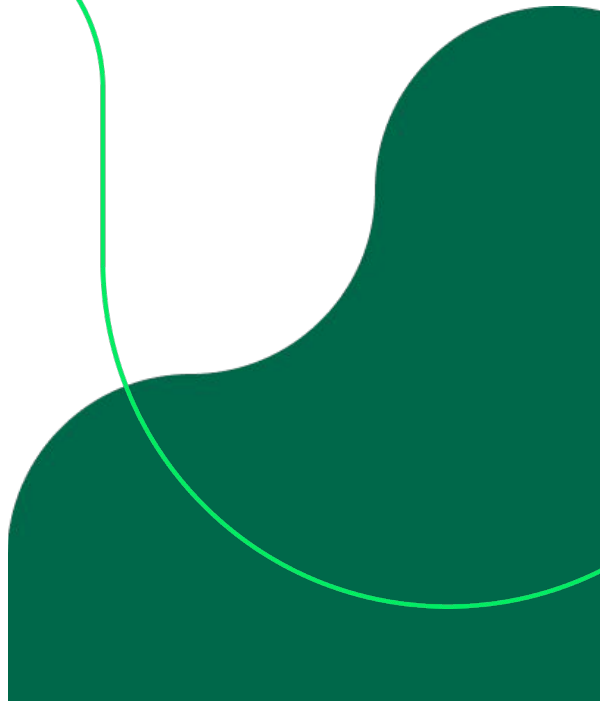
Merging gives a tighter range

`tweets/2022-05/xaarian-xylo.json`

More of the file is "xdg"

Trap #3:

Wide documents



Consider a single tweet…

```
{
      "id": 54691802283900930,
      "text": "RT @PostGradProblem: In preparation for the NFL lockout, I will
be spending twice as much time analyzing my fantasy baseball team during ...",
      "created_at": "Sun Apr 03 23:48:36 +0000 2011",
      "truncated": true,
      "in_reply_to_user_id": null,
      "in_reply_to_status_id": null,
      "favorited": false,

      … [snip] …

      "user": {
            "screen_name": "OldGREG85",
            "notifications": null,
            "profile_use_background_image": true,
            "statuses_count": 351,

      … [snip] …

      }
 }
```

source: https://gist.github.com/hrp/900964

Top tweeters?

Count by screen name?

```
{
      "id": 54691802283900930,
      "text": "RT @PostGradProblem: In preparation for the NFL lockout, I will
be spending twice as much time analyzing my fantasy baseball team during ...",
      "created_at": "Sun Apr 03 23:48:36 +0000 2011",
      "truncated": true,
      "in_reply_to_user_id": null,
      "in_reply_to_status_id": null,
      "favorited": false,

      … [snip] …

      "user": {
            "screen_name": "OldGREG85",
            "notifications": null,
            "profile_use_background_image": true,
            "statuses_count": 351,

      … [snip] …

      }
 }
```

source: https://gist.github.com/hrp/900964

```
{
   "$group": {
      "_id":"$user.screen_name",
      "cnt": {"$sum":1}
   }
}
```

Only need one field!

```
{
      "id": 54691802283900930,
      "text": "RT @PostGradProblem: In preparation for the NFL lockout, I will
be spending twice as much time analyzing my fantasy baseball team during ...",
      "created_at": "Sun Apr 03 23:48:36 +0000 2011",
      "truncated": true,
      "in_reply_to_user_id": null,
      "in_reply_to_status_id": null,
      "favorited": false,

      … [snip] …

      "user": {
            "screen_name": "OldGREG85",
            "notifications": null,
            "profile_use_background_image": true,
            "statuses_count": 351,

      … [snip] …

      }
 }
```

source: https://gist.github.com/hrp/900964

Trap #3: Wide documents

source: https://gist.github.com/hrp/900964

Download → discard

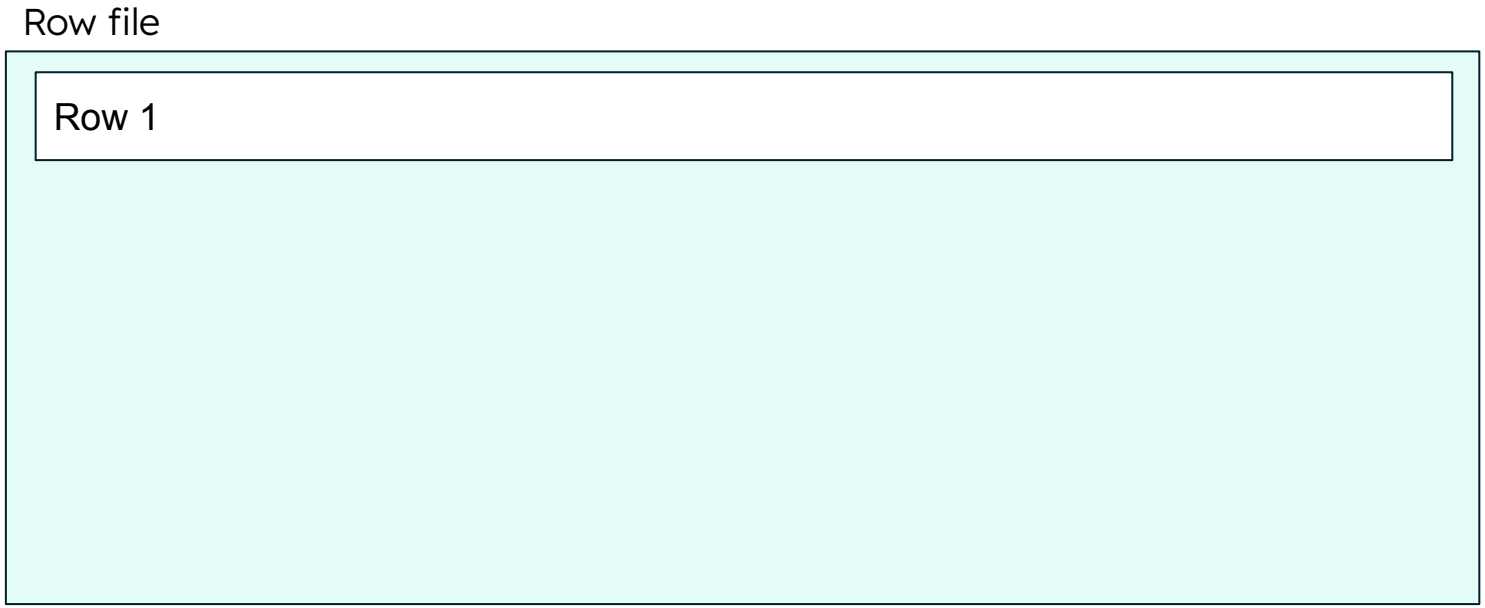
Principle #3

Minimize fetching unneeded fields

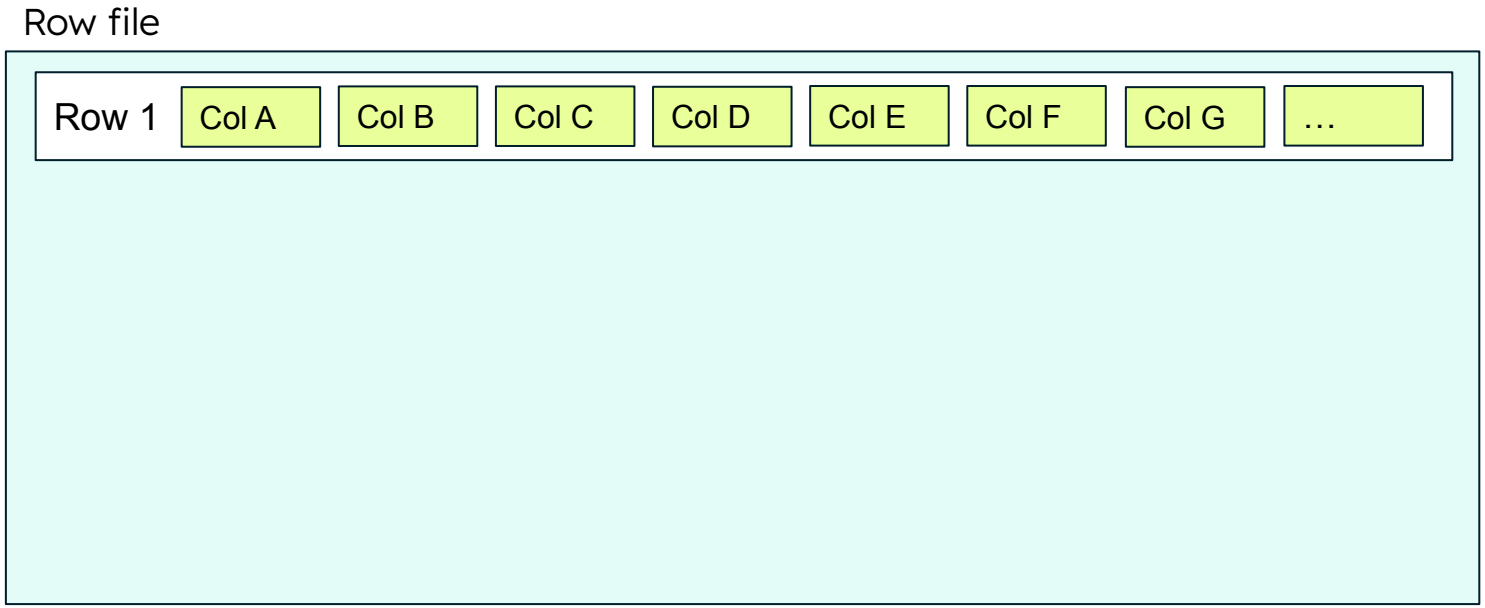
Column storage!

Row	→	Document
Column	→	Field

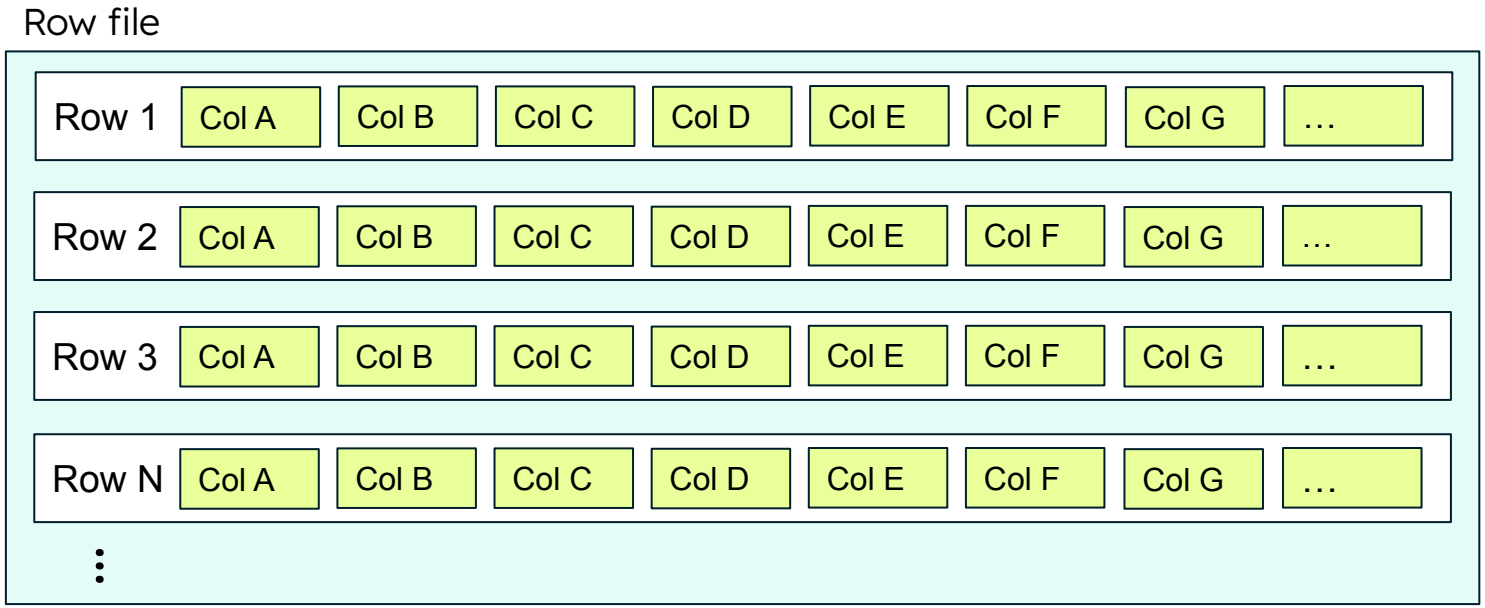
Row storage



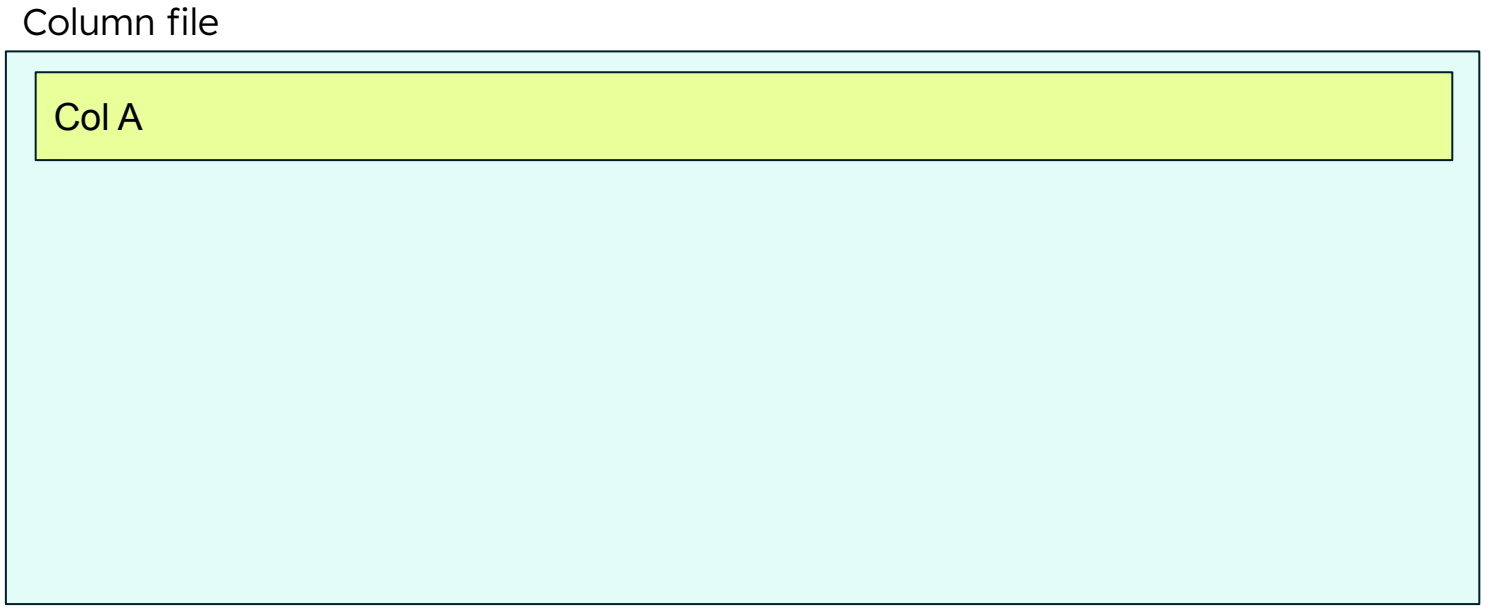
Row storage



Row storage

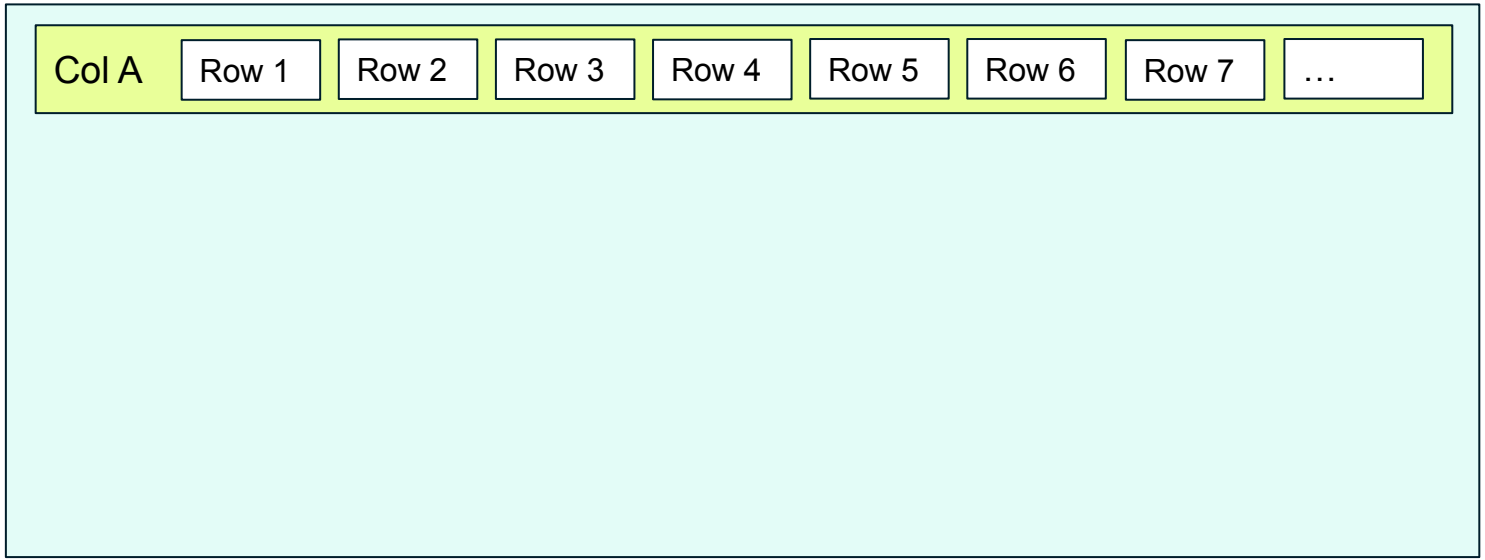

Row file
Row 1
Col A
Col B
Col C
Col D
Col E
Col F
Col G
…
Row 2
Col A
Col B
Col C
Col D
Col E
Col F
Col G
…
Row 3
Col A
Col B
Col C
Col D
Col E
Col F
Col G
…
Row N
Col A
Col B
Col C
Col D
Col E
Col F
Col G
…

Column storage

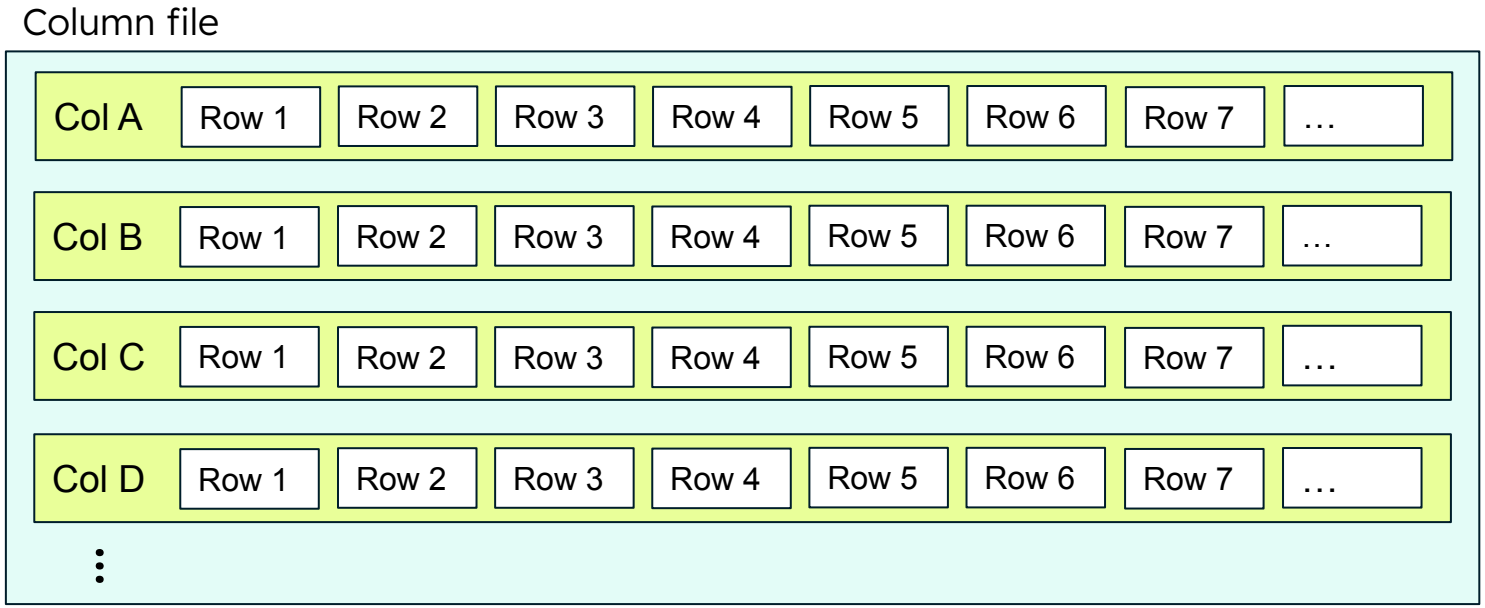


Column storage

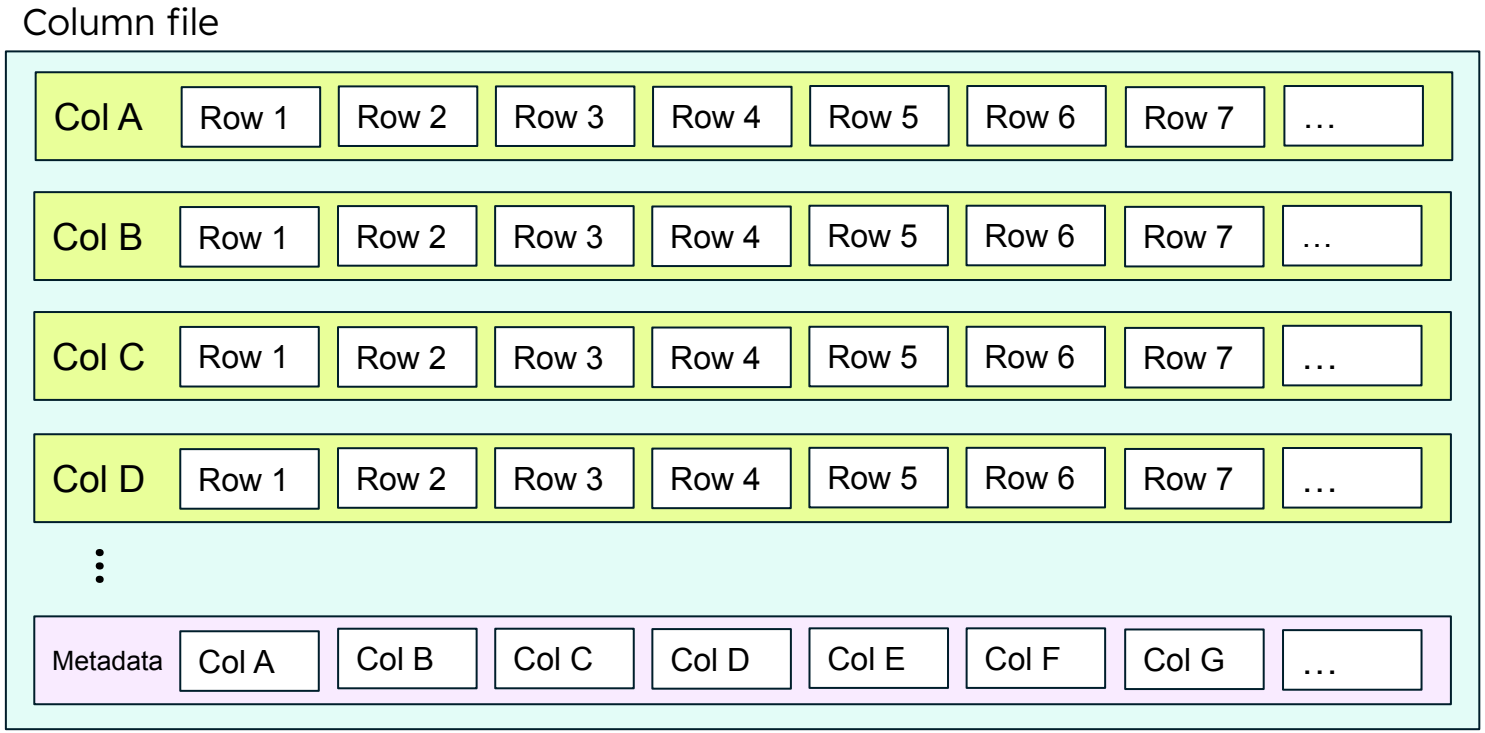
Column file



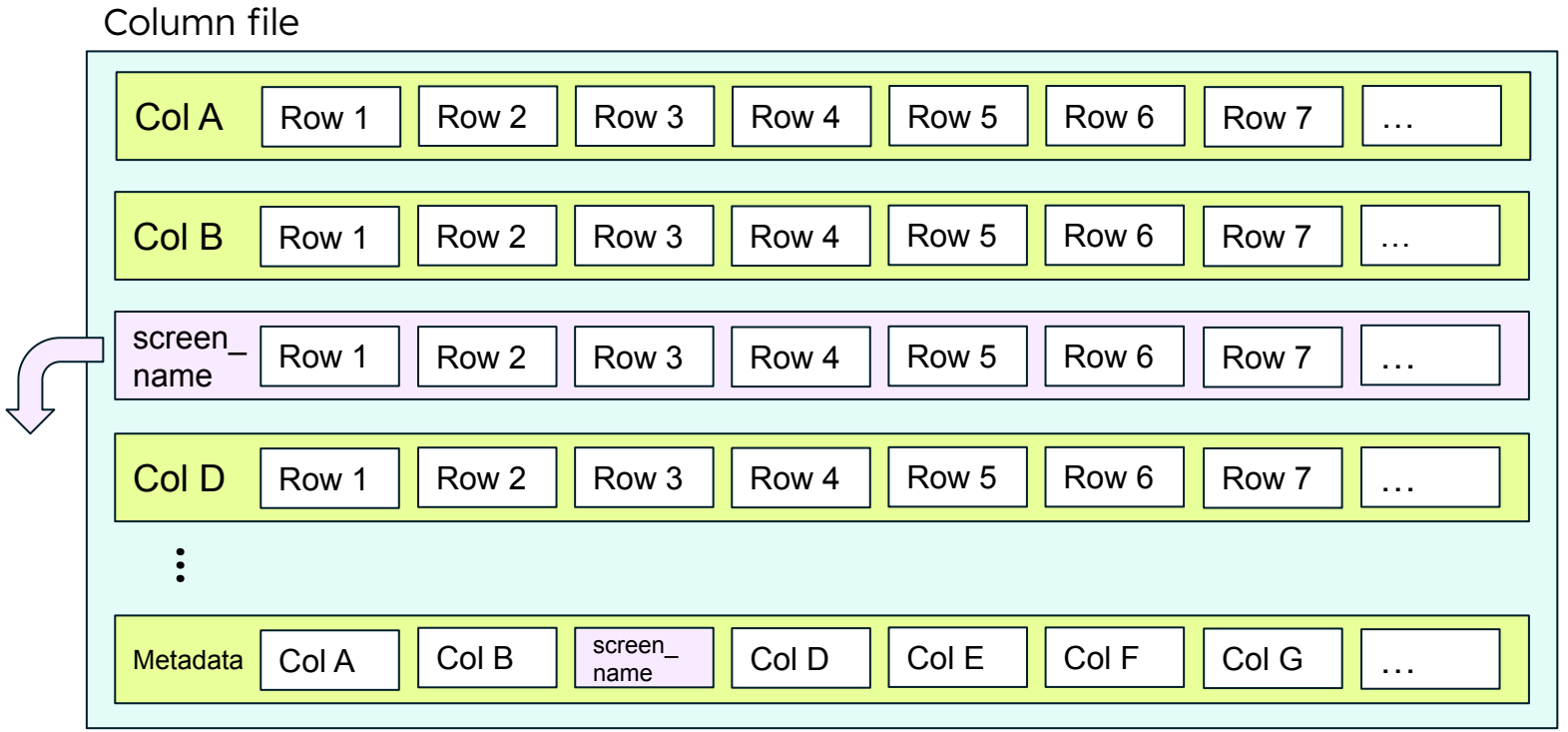
Column storage



Column metadata

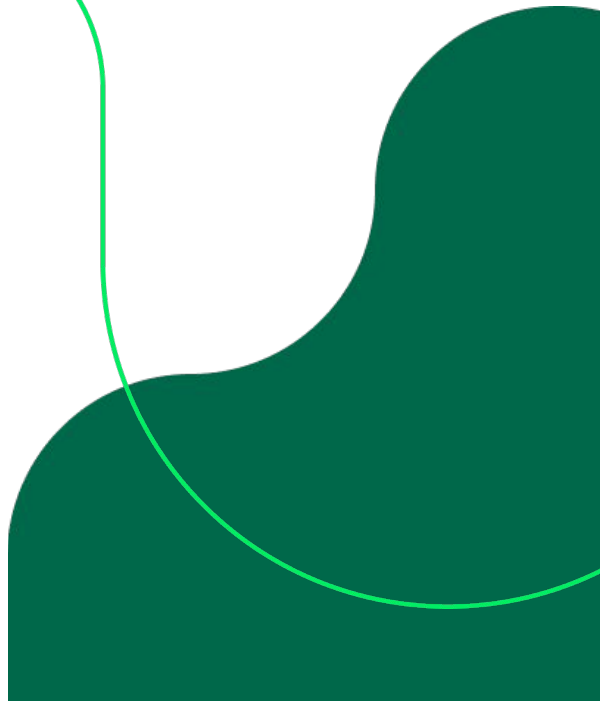


Fetch by column!



Recap:

Three principles



Tiny files → Minimize network trips

Tiny files → Minimize network trips

Wasteful scans → Minimize unneeded documents

Tiny files → Minimize network trips

Wasteful scans → Minimize unneeded documents

Wide documents → Minimize unneeded fields

Minimize network trips

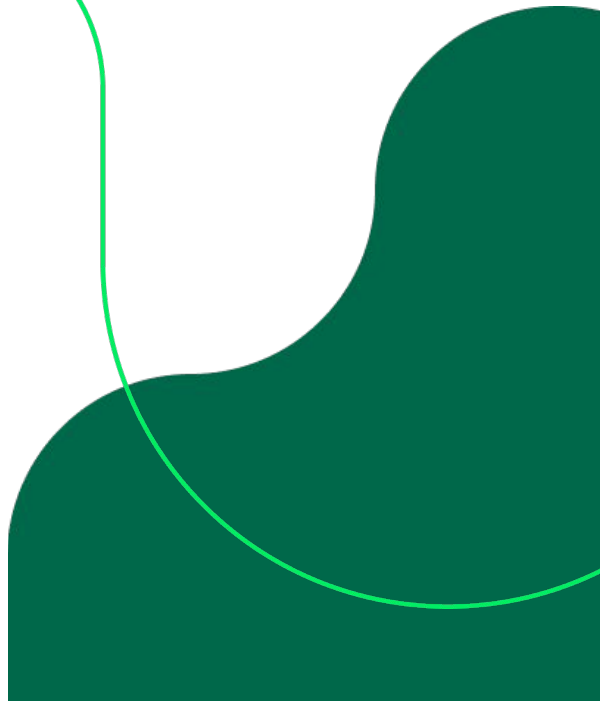
Minimize unneeded documents

Minimize unneeded fields

Optimized storage

Metadata catalog

Auto-rebalancing



Change #1:

Optimized storage

No folders

Shard key

(A secondary index)

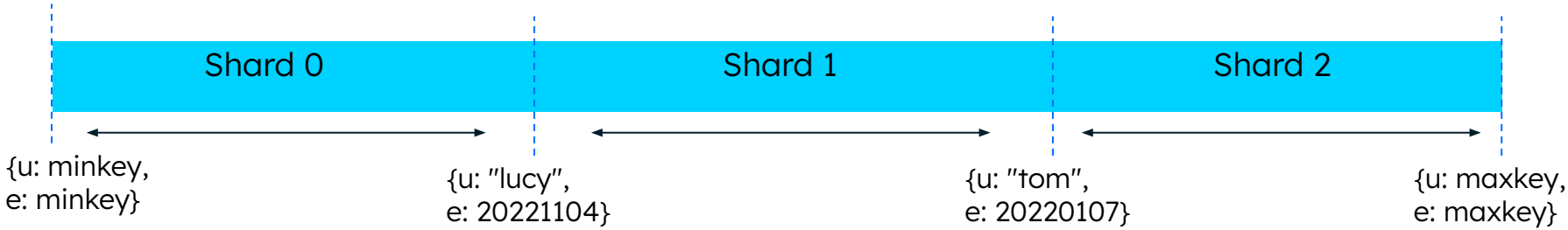
```
{user_id:1, event_time:1}
```

```
{u:1, e:1}
```

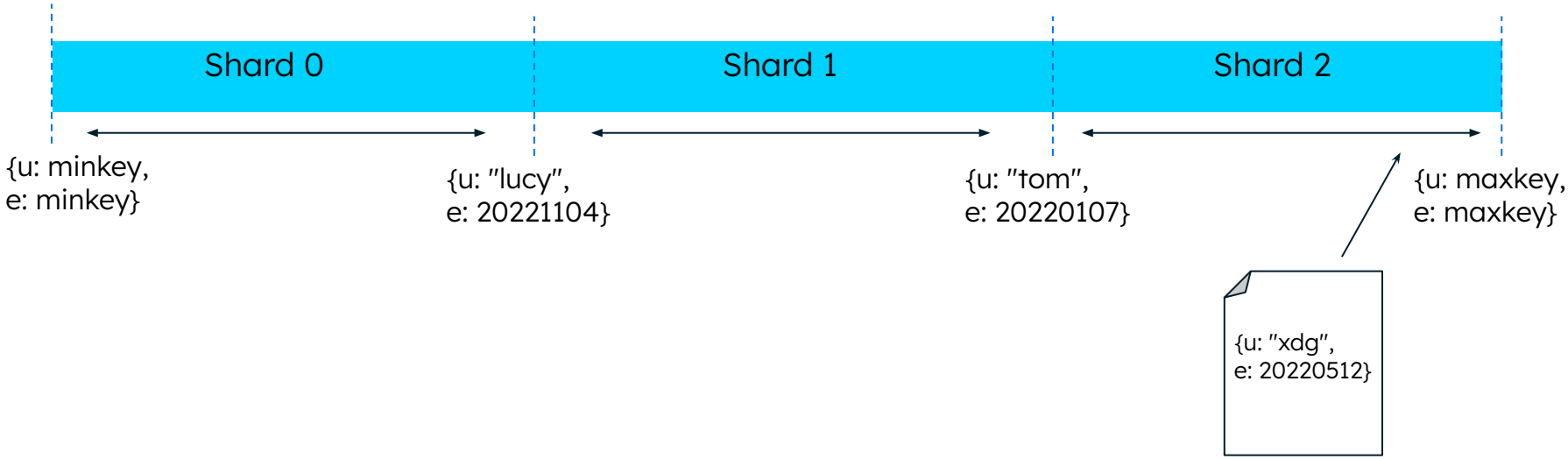
Key space → sorted range of all values

Key space → divide into ranges

Key space → shards

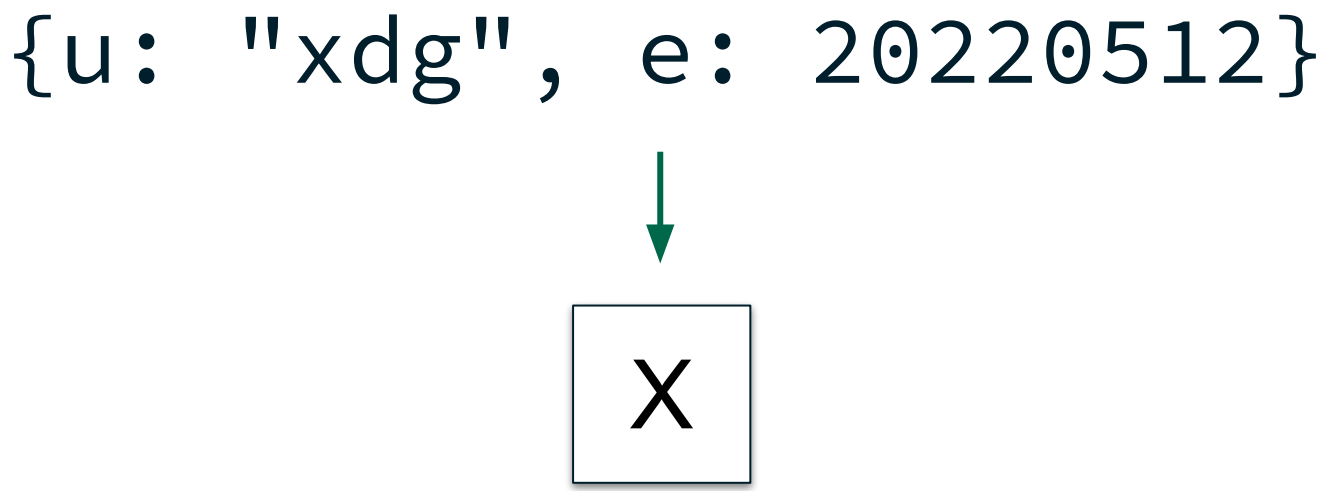


Key space → every doc has a place

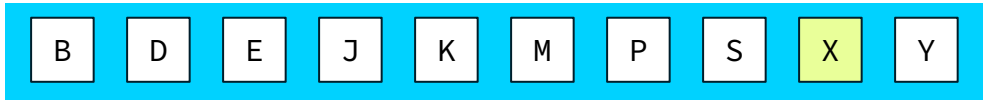


Files?

```
{u: "xdg", e: 20220512}
```

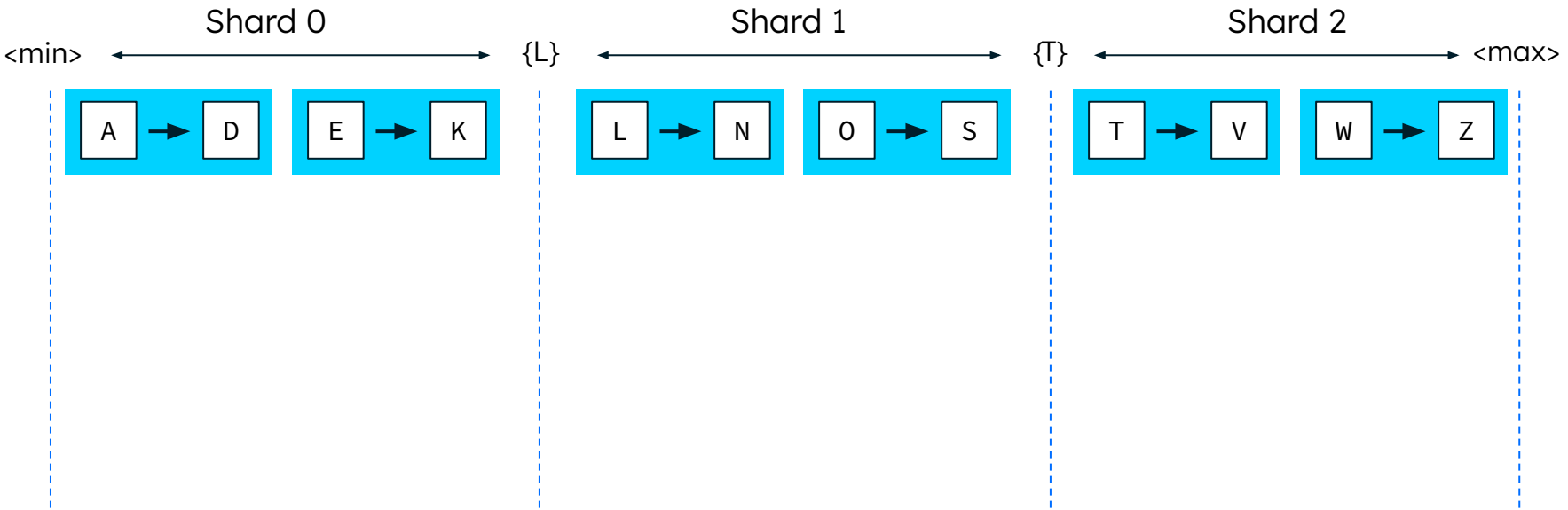

{u: "xdg", e: 20220512}
X

Files → sort docs by the shard key



Files → sort docs by the shard key

Key space → every file has a place



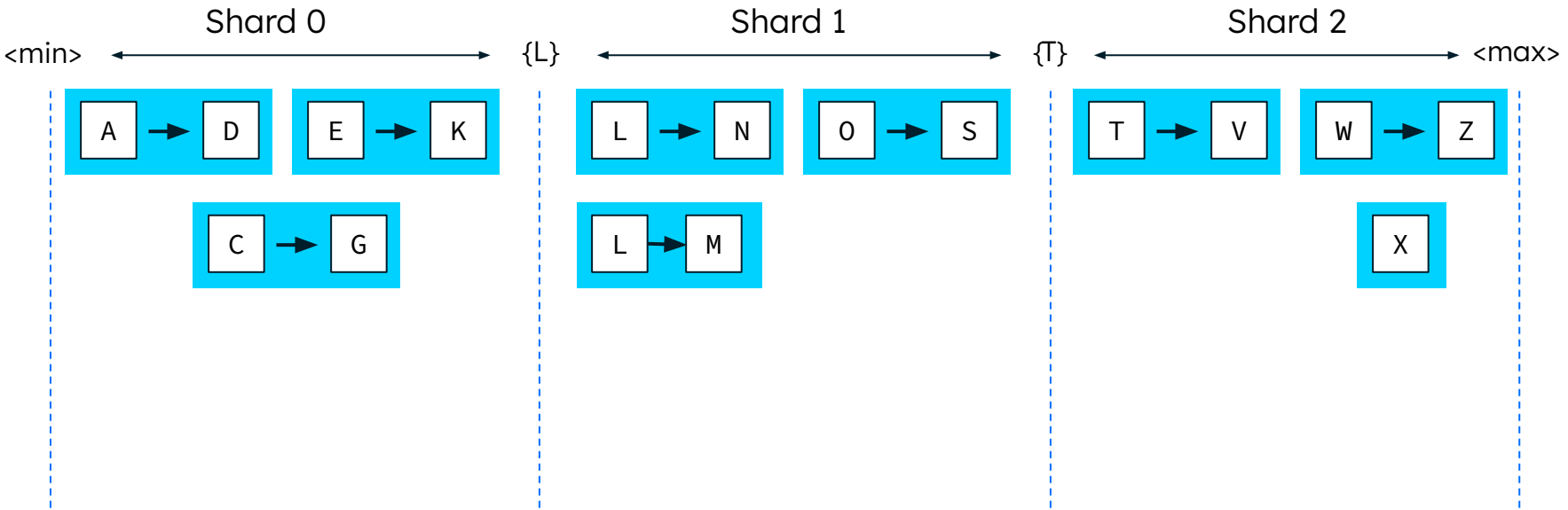
Big files

(but not too big)

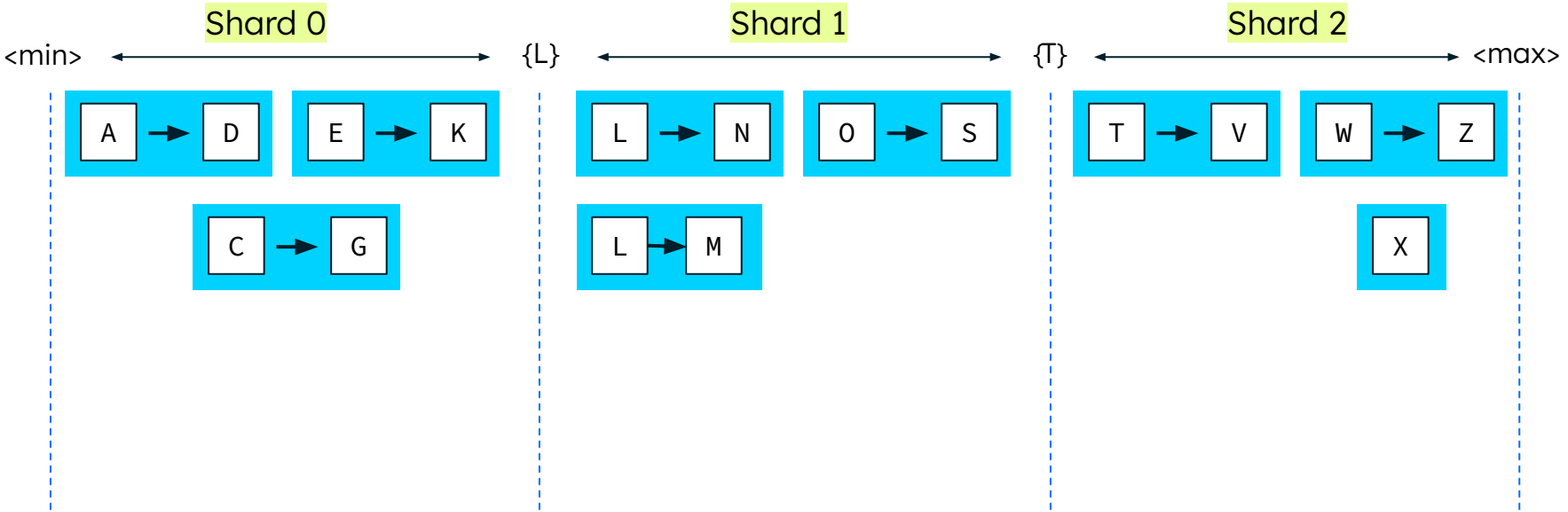
Columns → Parquet format

(usually)

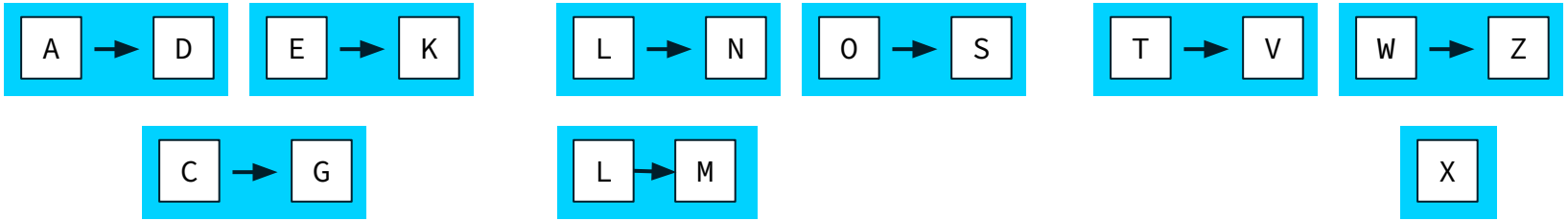
File overlaps are okay!

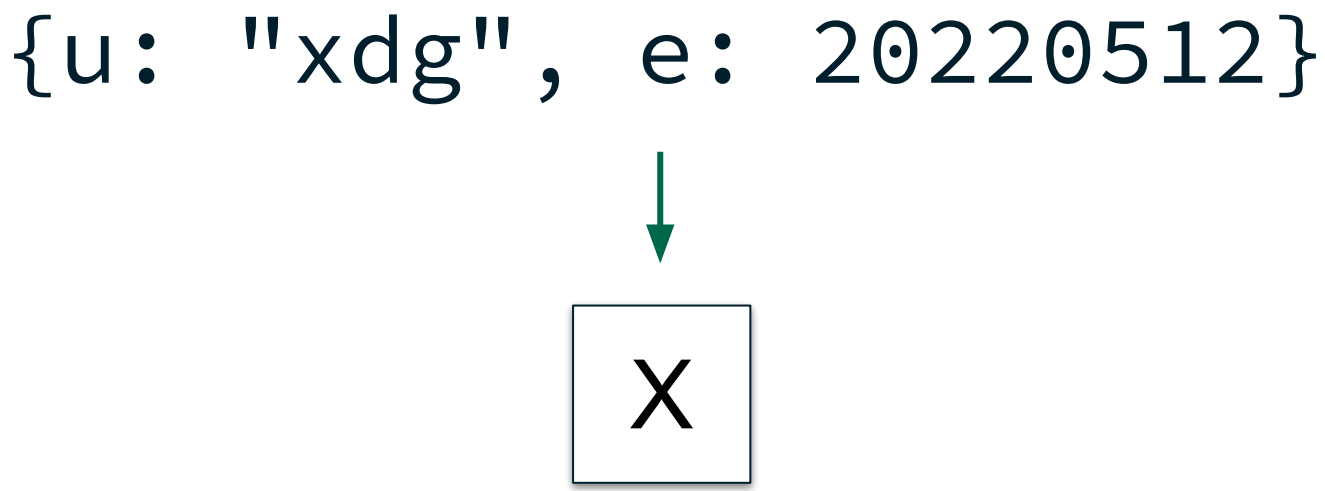


Shard metadata stored outside S3



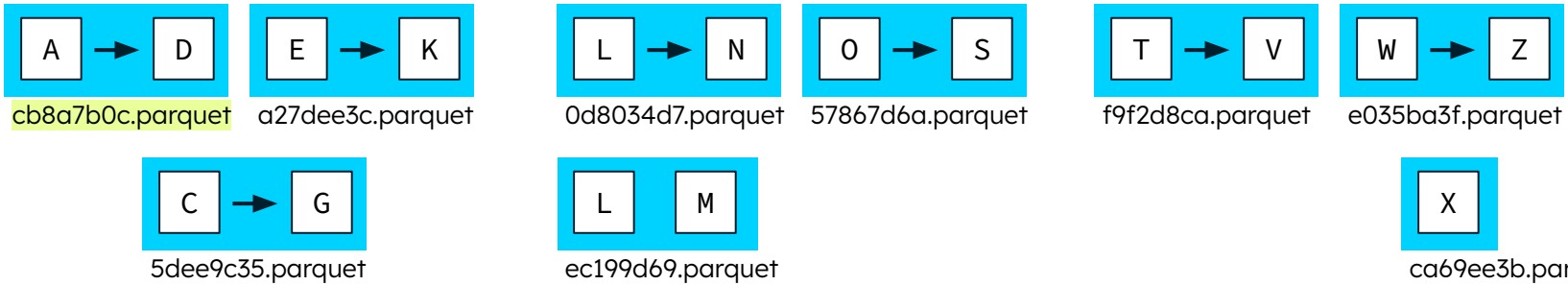
S3 only contains files




{u: "xdg", e: 20220512}
X

File name ≠ values

Unique name from contents



Optimized storage

√ Shard key to group related data

√ Big files, but not too big

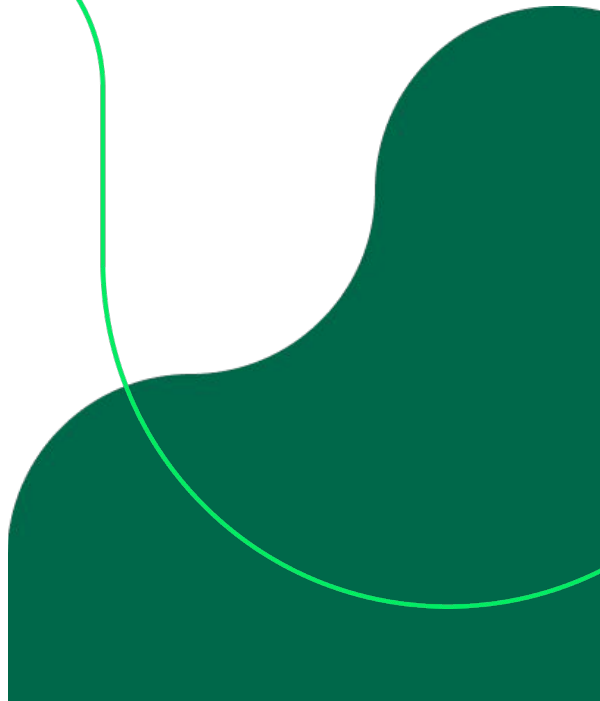
√ Column storage (if appropriate)

‘ca69ee3b.parquet’

??

Change #2:

Metadata catalog



Metadata about your data

File metadata

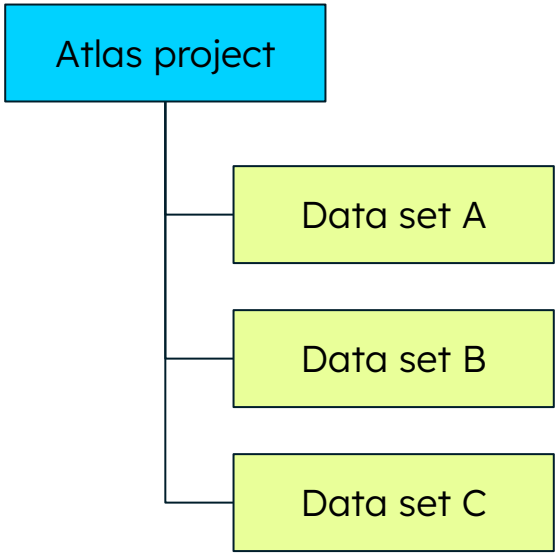
Physical location ("s3://<bucket>/…/cb8a7b0c.parquet")

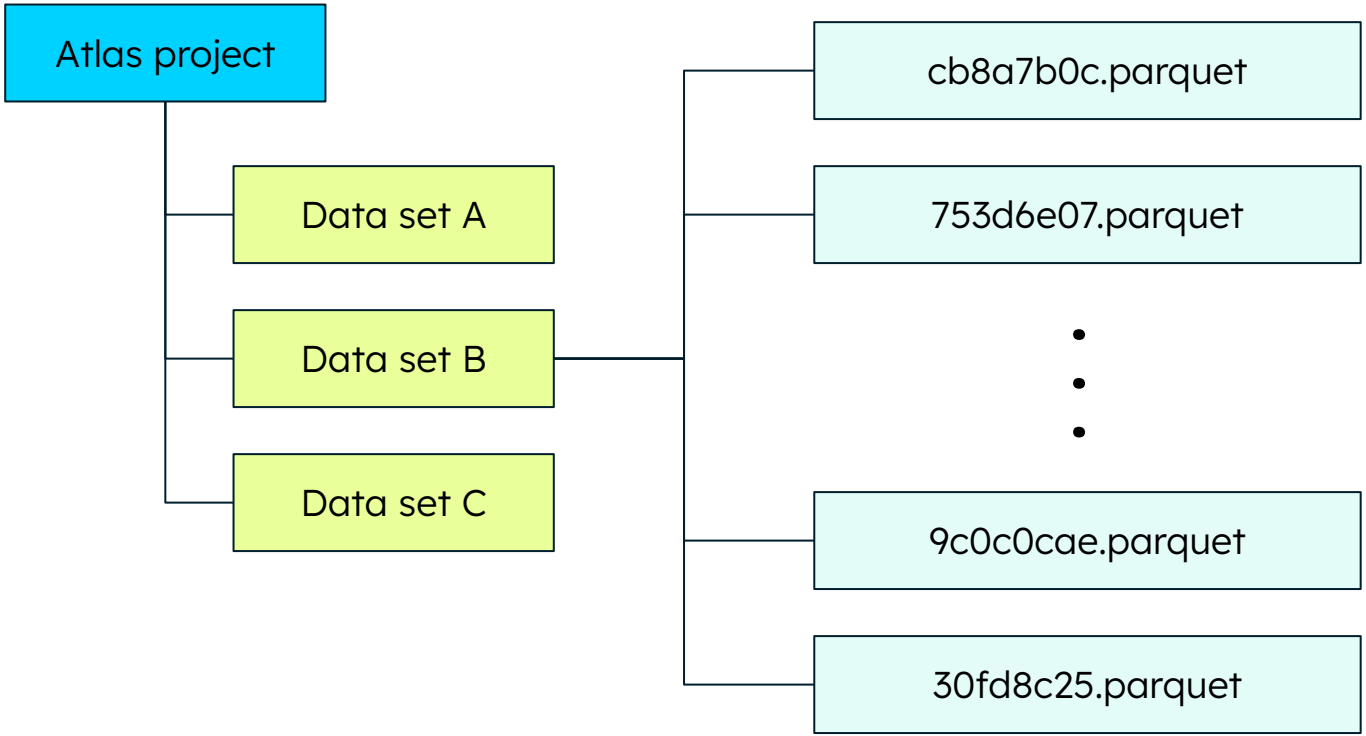
Upper/lower shard key bounds

File size

Document count

Project → holds data sets


Atlas project
Data set A
Data set B
Data set C


Atlas project
Data set A
Data set B
Data set C
cb8a7b0c.parquet
753d6e07.parquet
9c0c0cae.parquet
30fd8c25.parquet

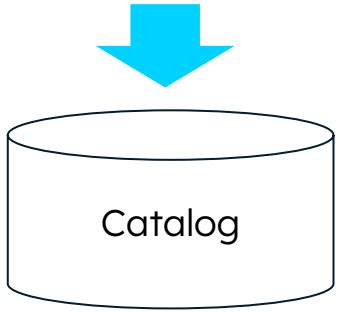
Data sets → Collection

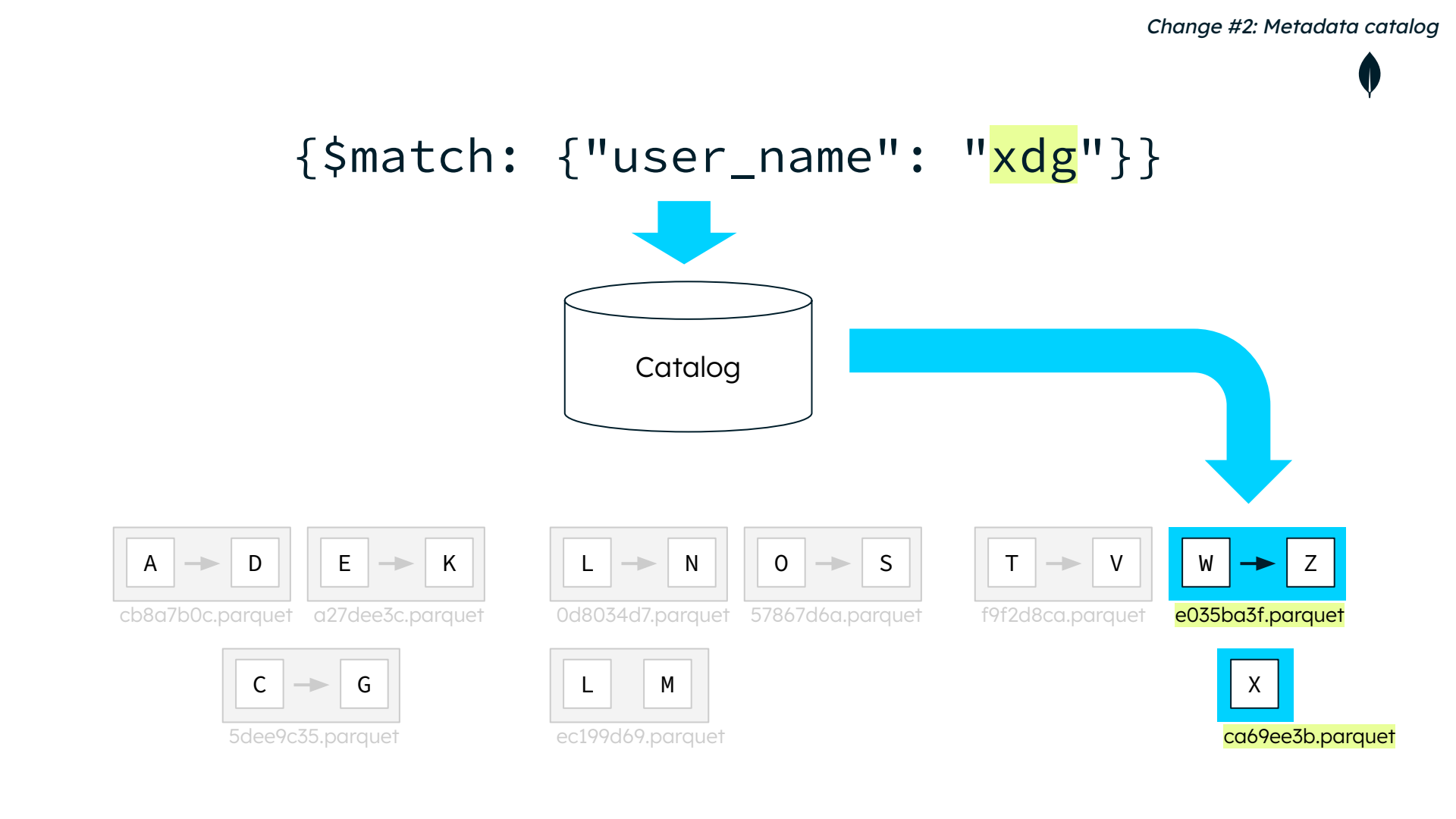
(Atlas Data Federation)

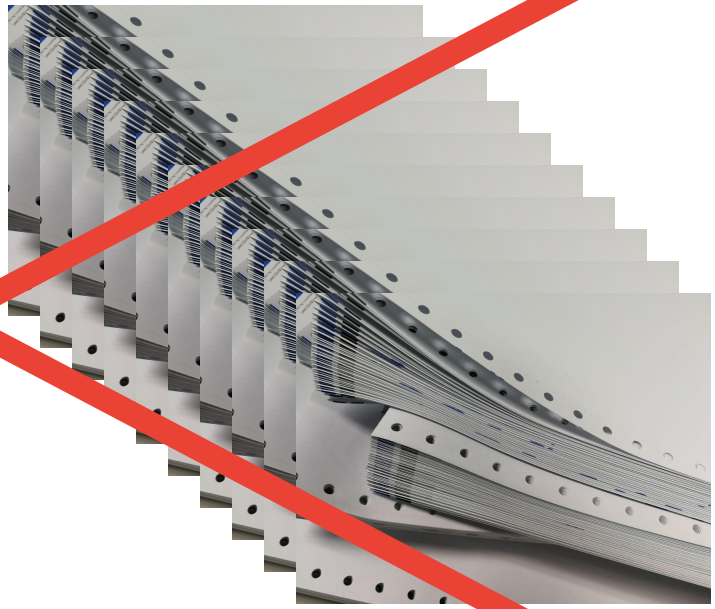
Query plan → list of files

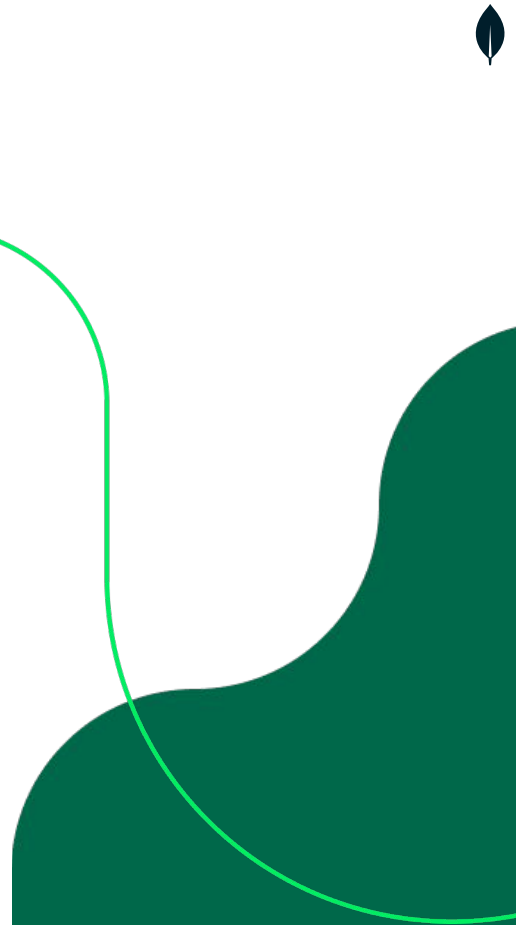
```
{$match: {"user_name": "xdg"}}
```

```
{$match: {"user_name": "xdg"}}
```




{$match: {"user_name": "xdg"}}
Catalog
A
D
cb8a7b0c.parquet
E
K
a27dee3c.parquet
L
N
0d8034d7.parquet
O
S
57867d6a.parquet
T
V
f9f2d8ca.parquet
W
Z
e035ba3f.parquet
C
G
5dee9c35.parquet
L
M
ec199d69.parquet
X
ca69ee3b.parquet

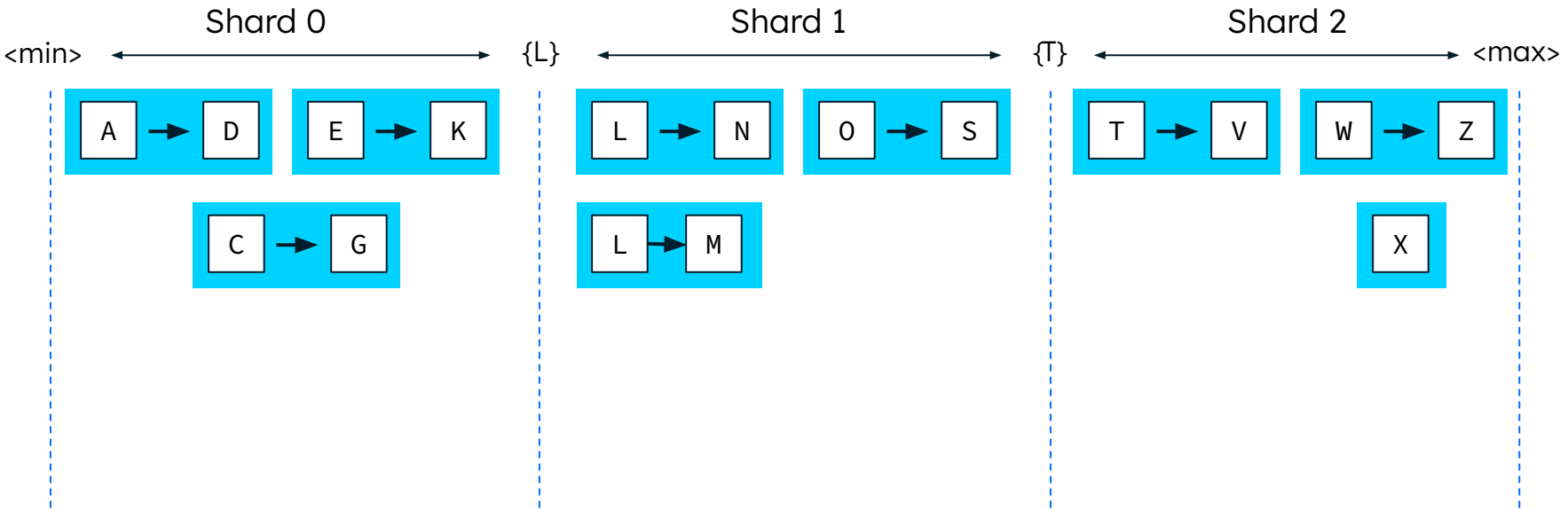




Change #3:

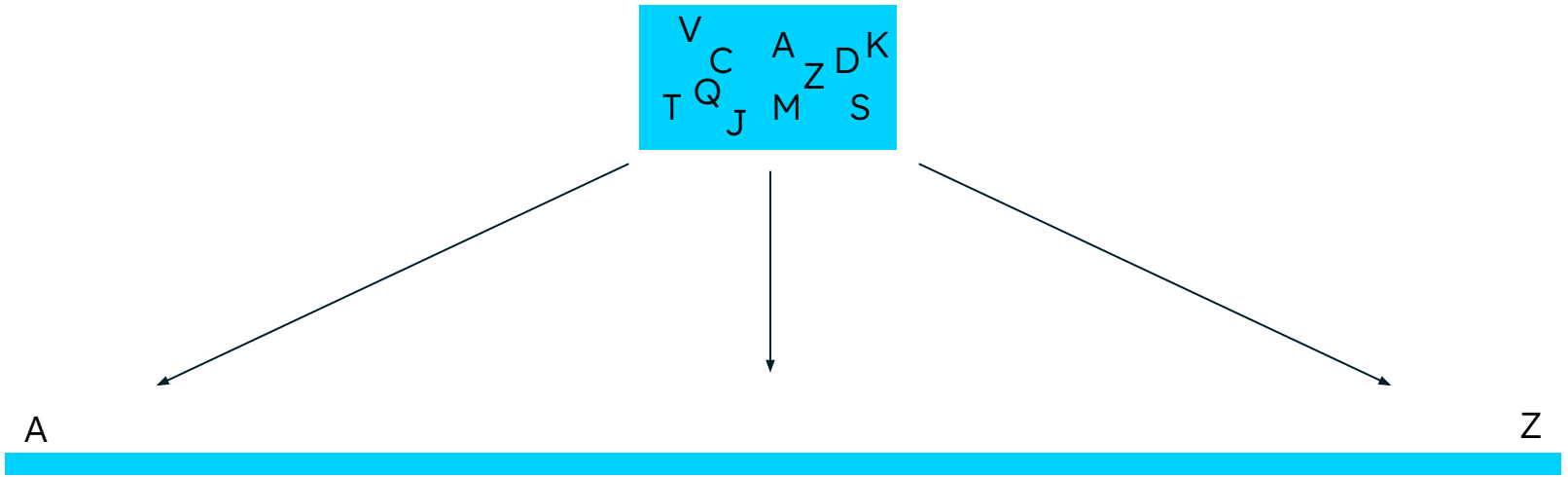
Auto-rebalancing

How did the overlaps happen?

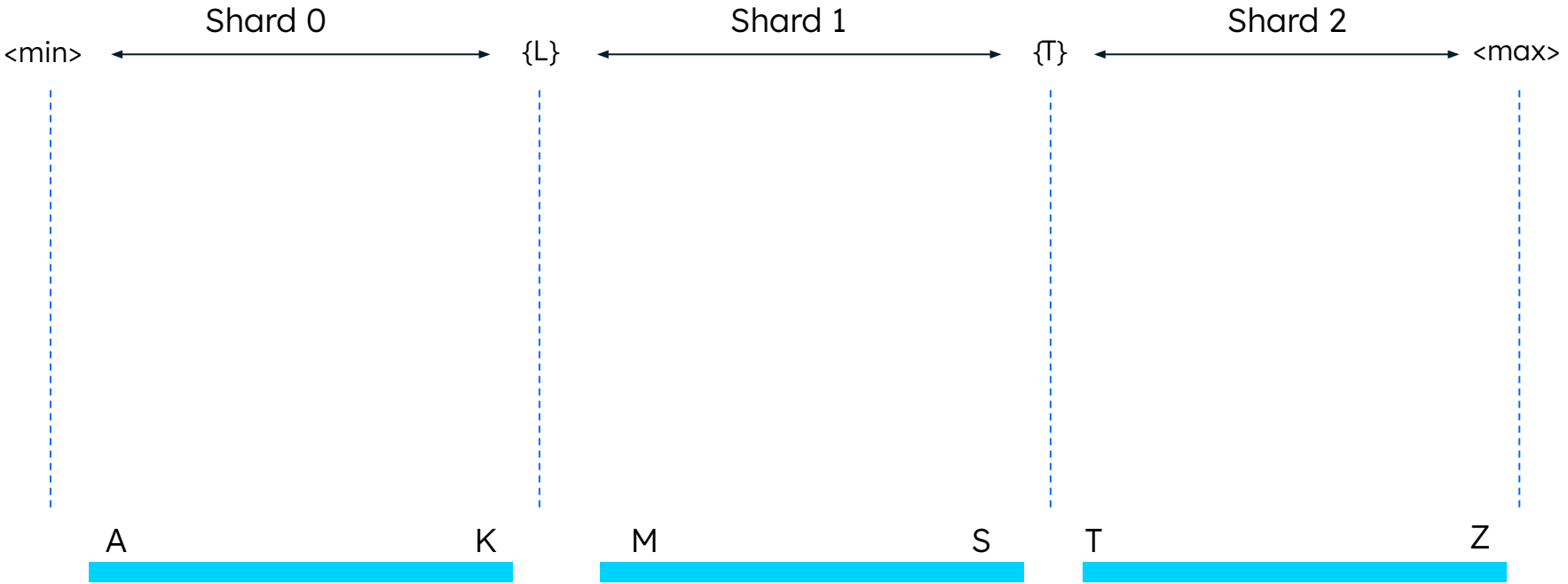


Continuous ingestion

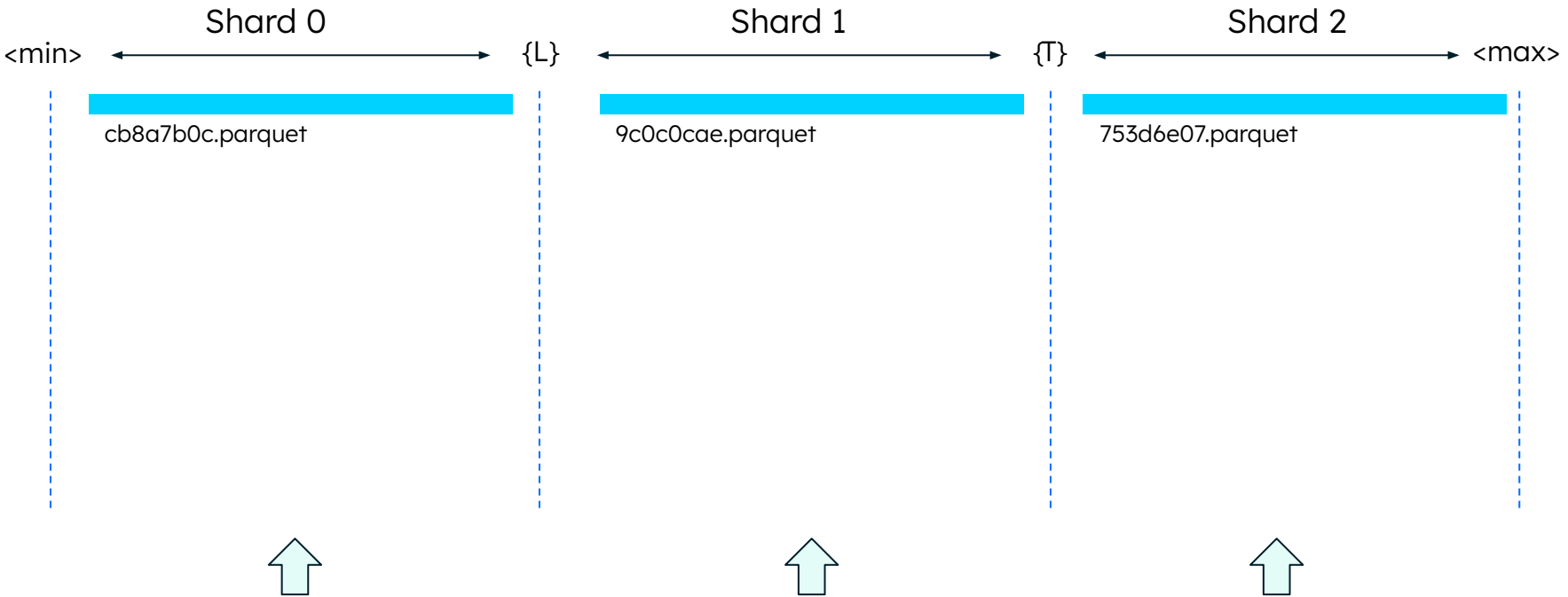
Sort new data along the keyspace



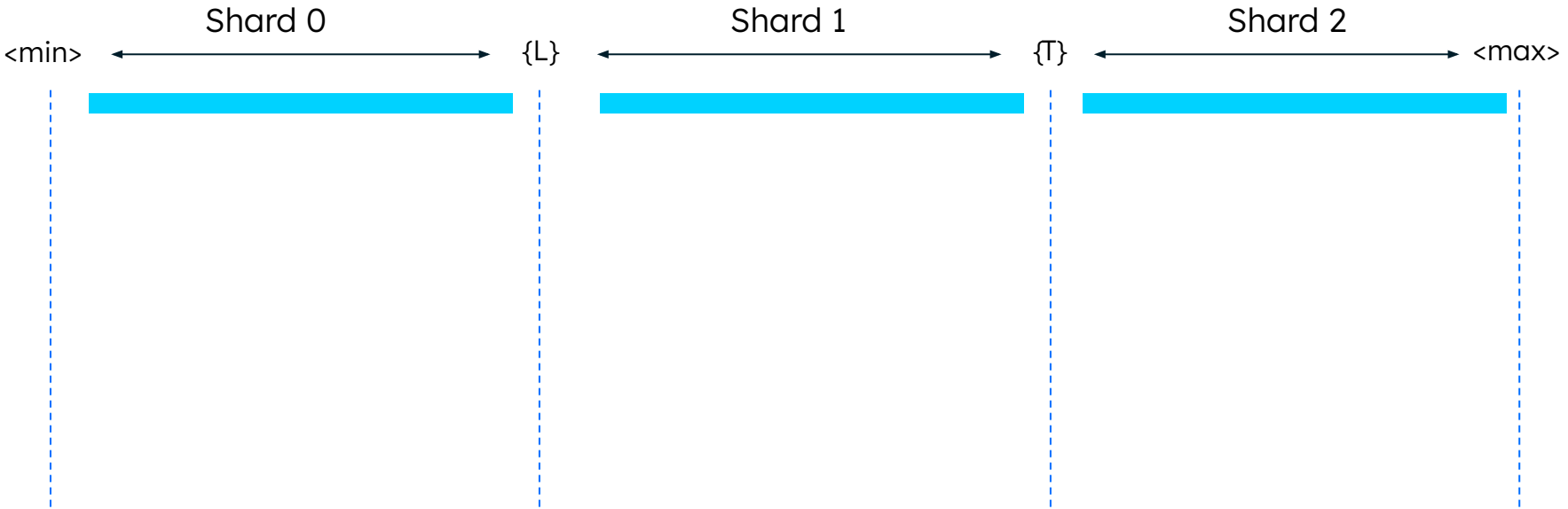
Split along shard boundaries



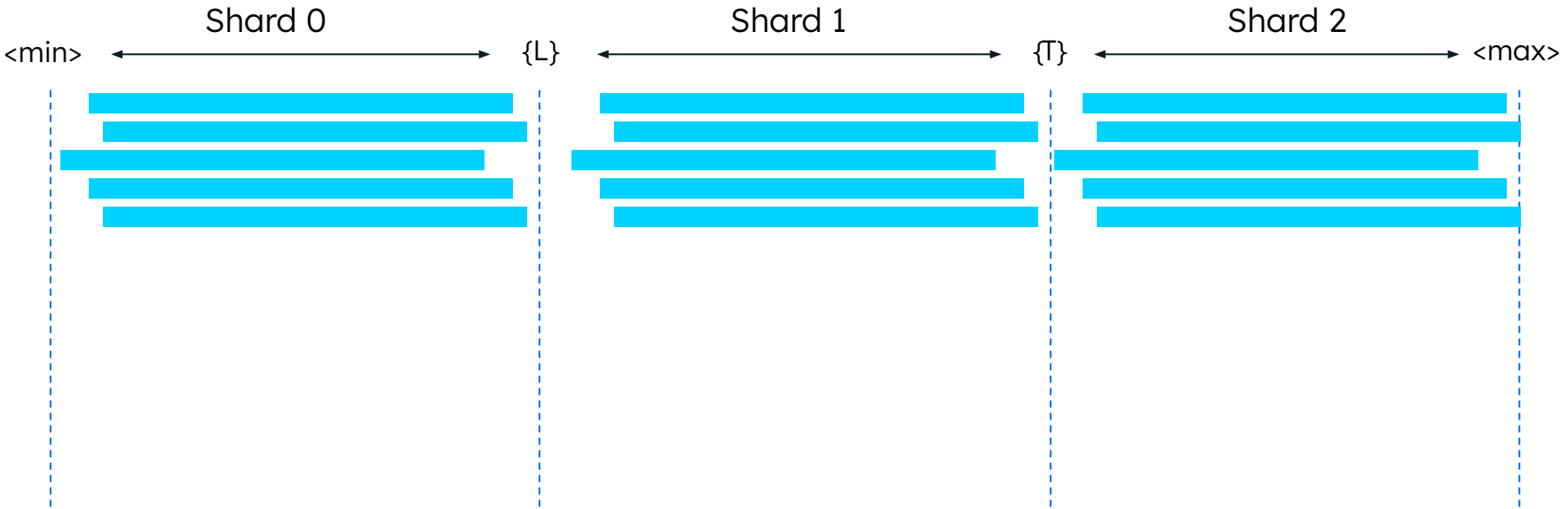
Upload to S3


Shard 0
Shard 1
Shard 2
<min>
{L}
{T}
<max>
cb8a7b0c.parquet
9c0c0cae.parquet
753d6e07.parquet

Files might be small



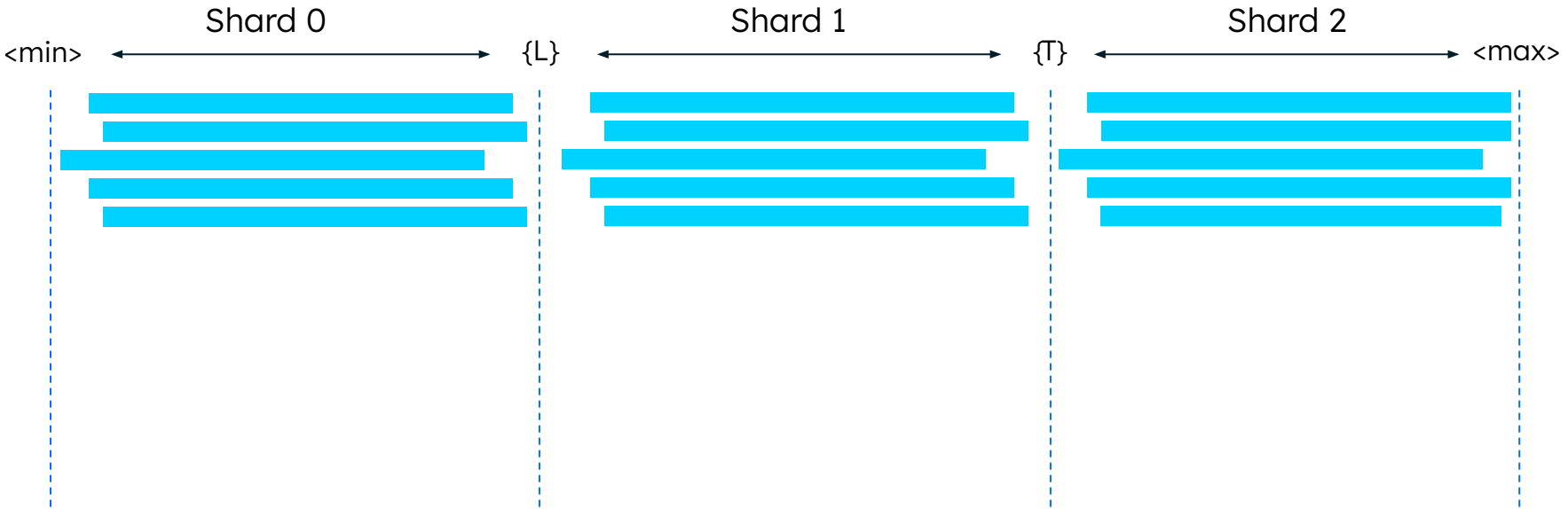
Small files accumulate over time



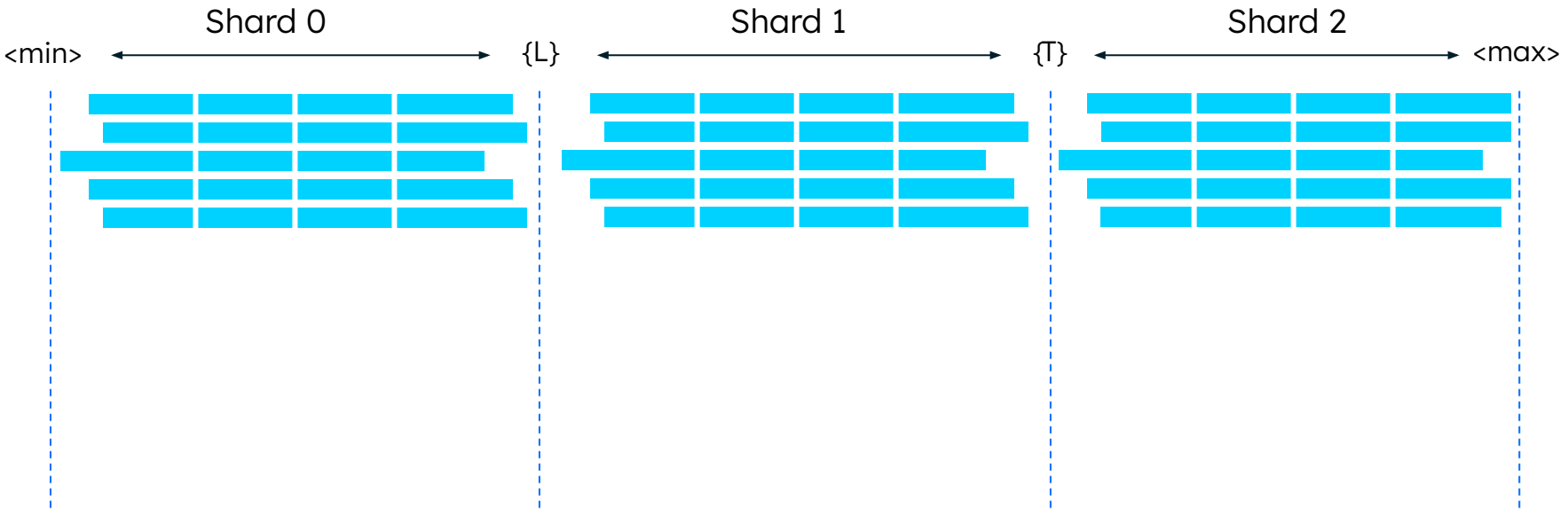
Auto-rebalancing!

Heuristic triggers

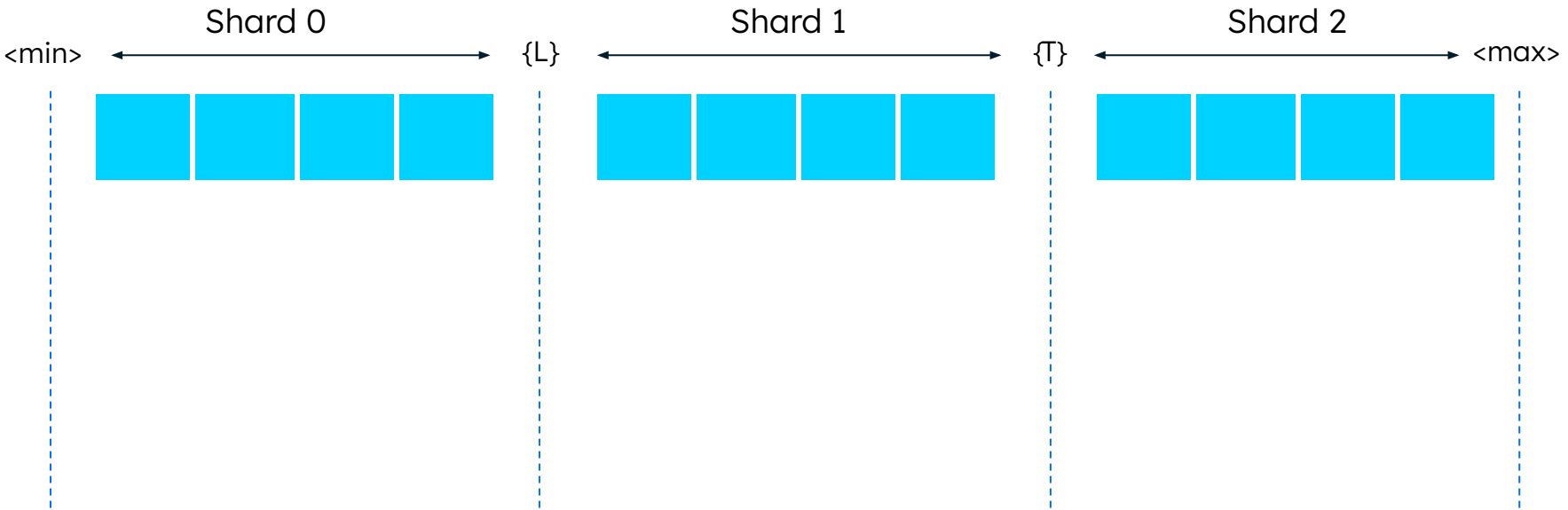
Small files? → Compaction



Split files…

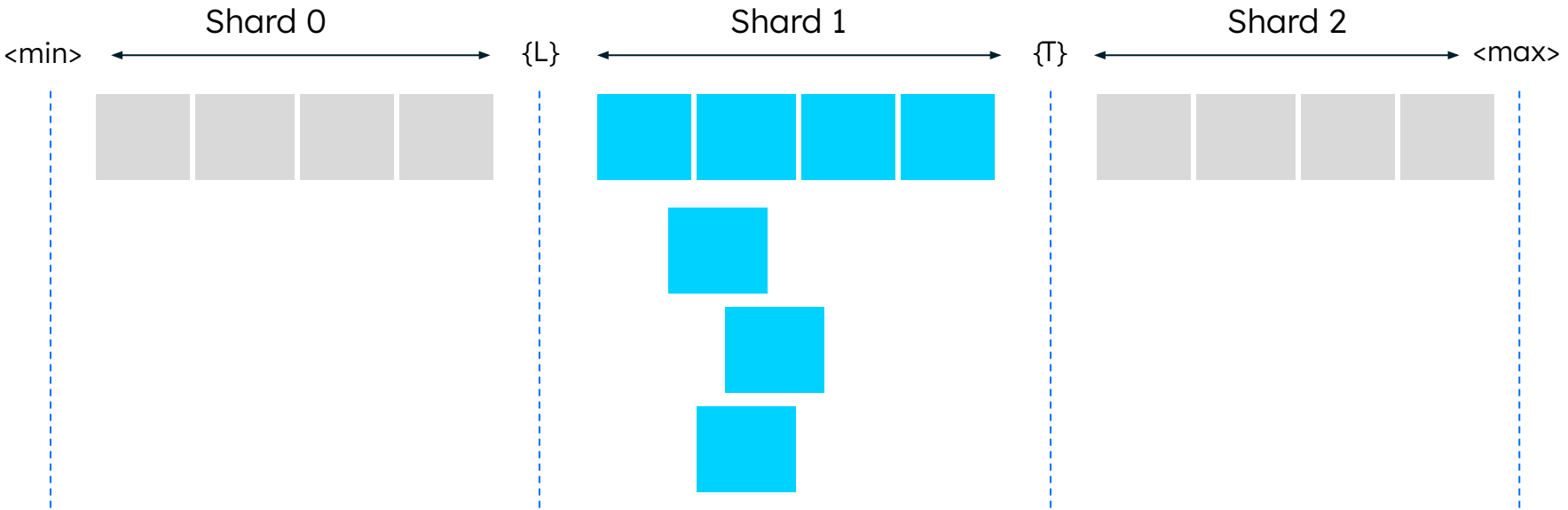


…sort and combine

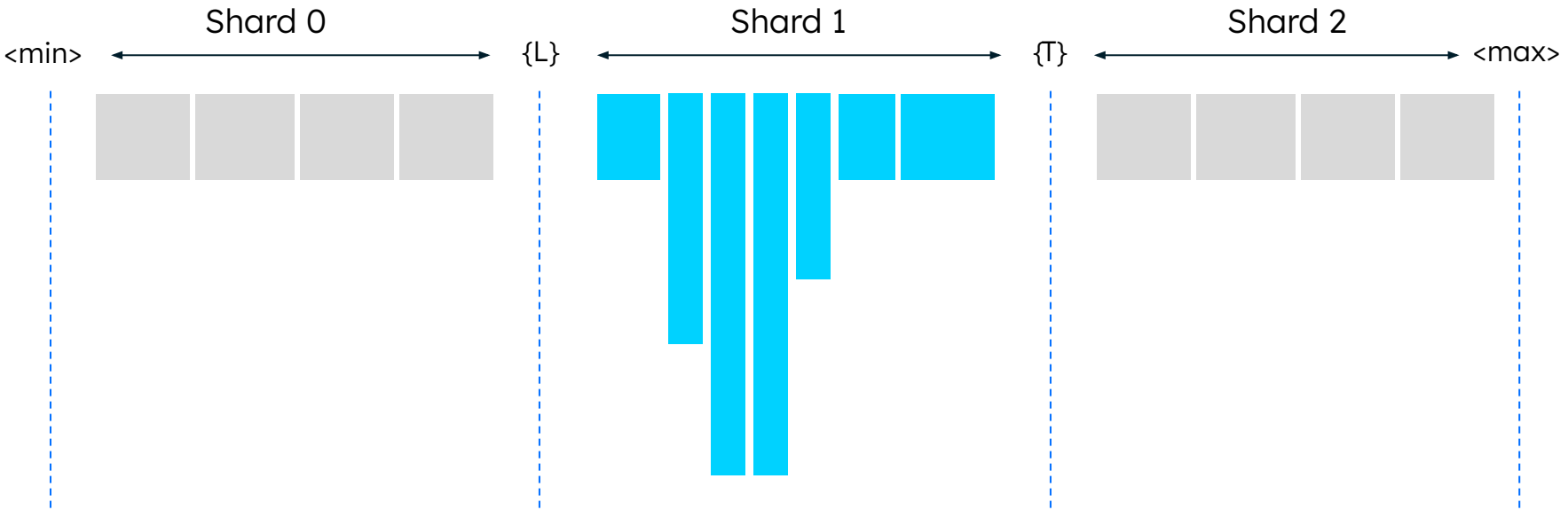


Works shard by shard

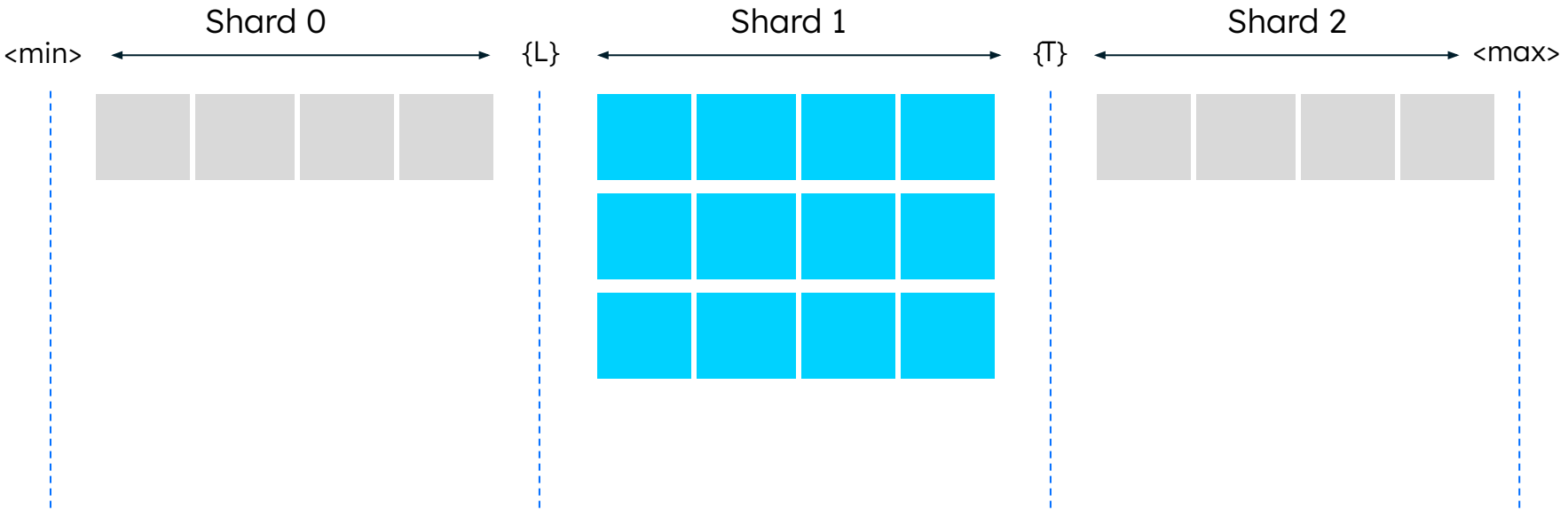
Hot shard? → Rebalance



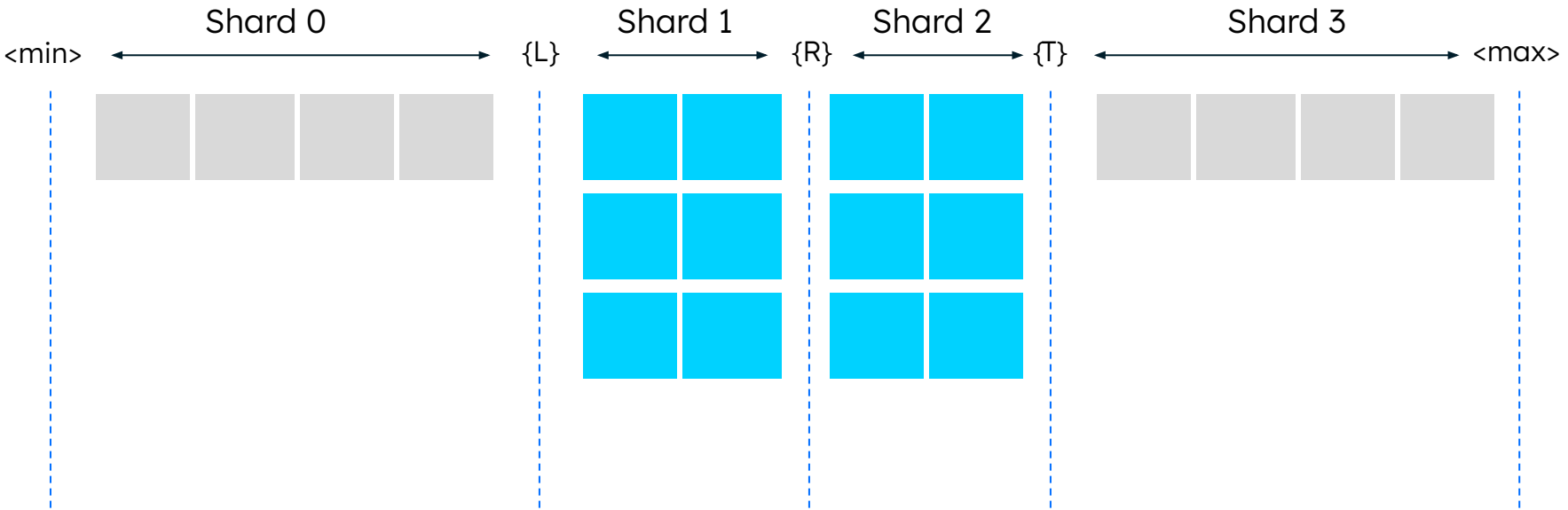
Rewrite with narrower ranges



Big shard?

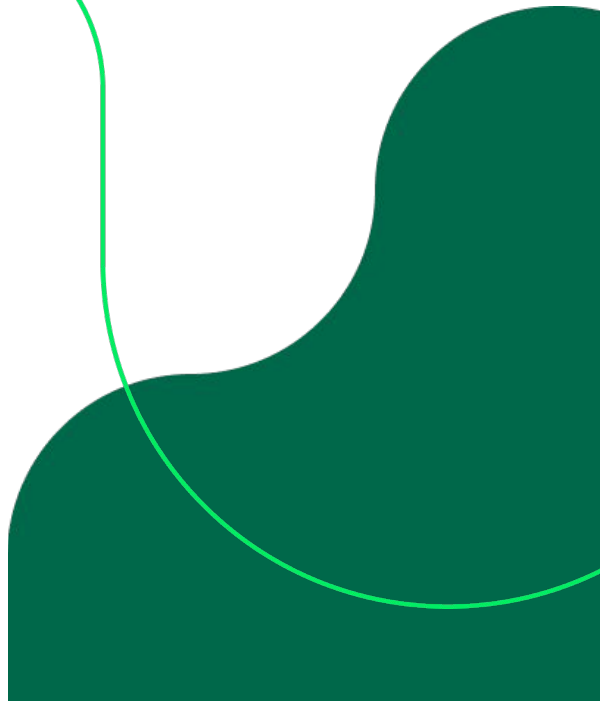


Big shard? → Split it



Continuous optimization

In summary…



Minimize…

Number of network trips

Fetching unneeded documents

Fetching unneeded fields

Better performance

Optimized storage

Metadata catalog

Auto-rebalancing

Roadmap

Available now…

Snapshot ingestion (Preview)

Snapshot ingestion

Ingest Cloud Backup Snapshots

High performance, low cost historical data

Join across history for timeseries analysis

Roadmap

Available now…

Snapshot ingestion (Preview)

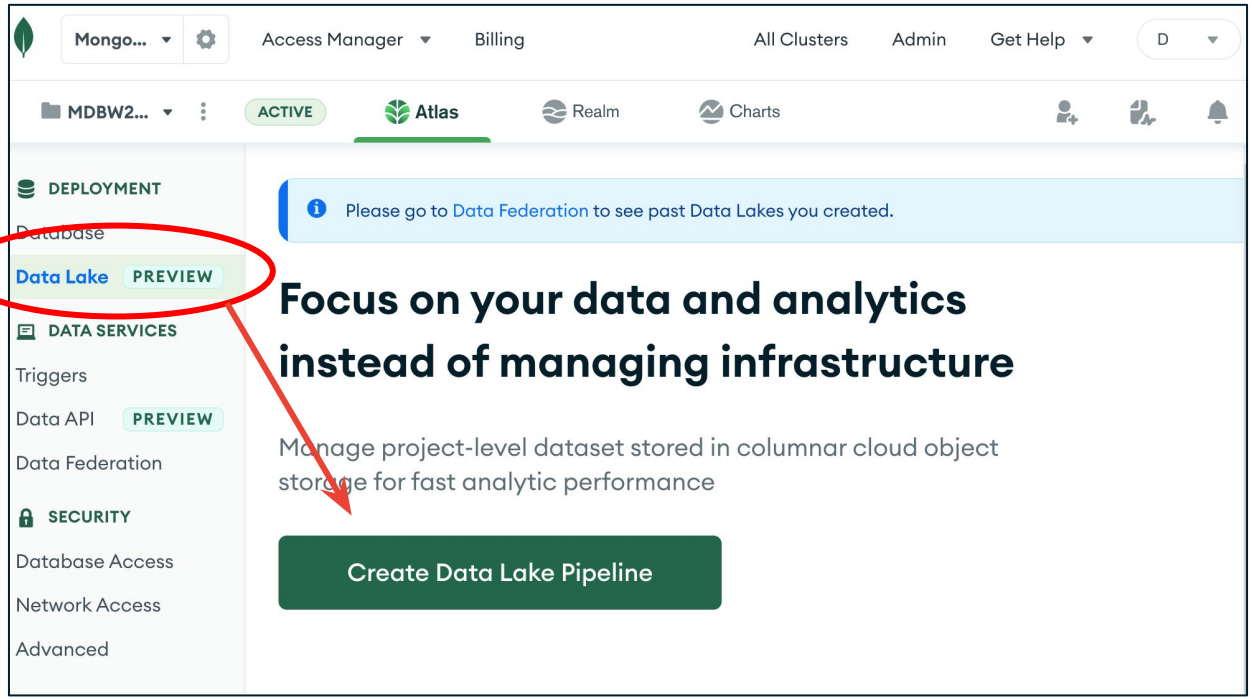
Coming soon…

Migrate online archive to optimized storage

Future…

New ingestion sources

Curious? Try it out!




BOX #
CONTENTS
DATE

Data Lake ≠Boxes

Atlas Data Lake → Answers

Thank you for your time