GETTING STARTED WITH

ClickHouse

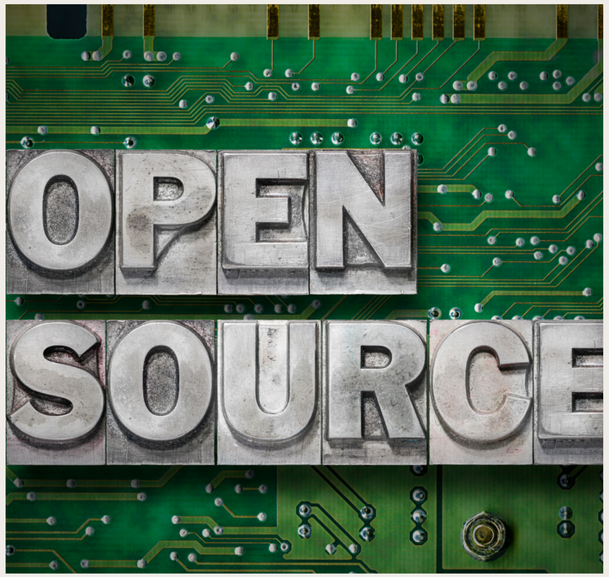
GoldenClover
Education

www.learn-olapdb.com


Data Warehouse

ClickHouse Database

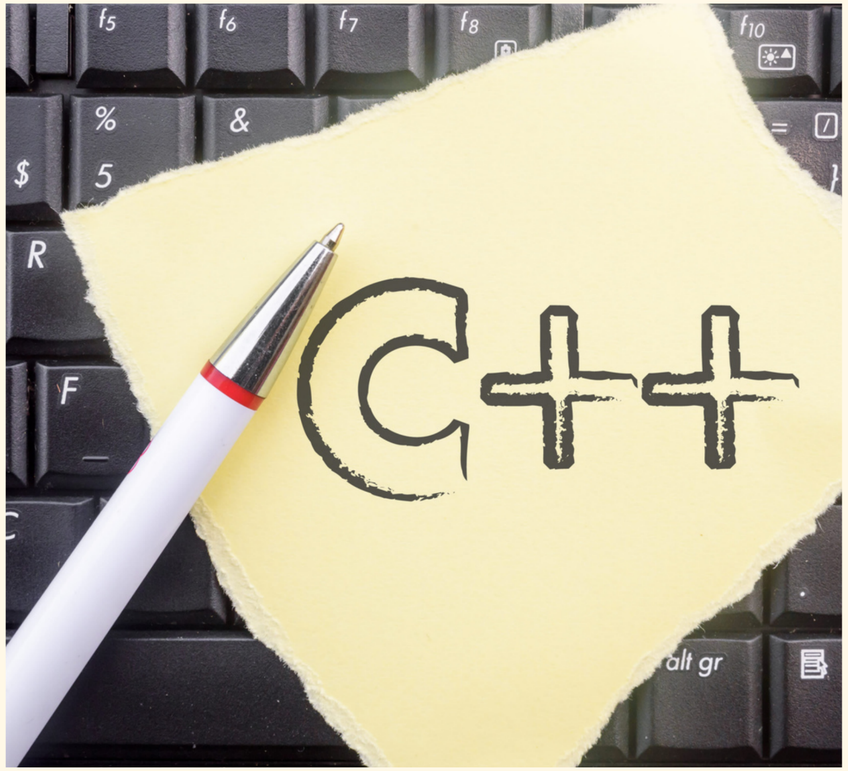
CLICKHOUSE DATABASE



Apache 2.0 Licence

Super Fast

Cost Effective



6K+

Forks

35K +

Pull Requests

1200+

Contributors

30K +

Github Stars

GETTING STARTED WITH

ClickHouse

GoldenClover
Education

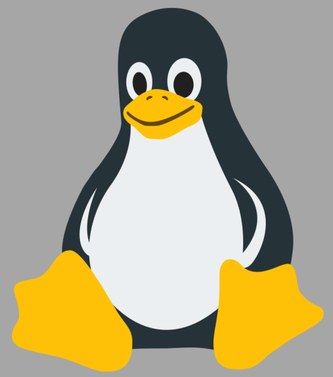
www.learn-olapdb.com

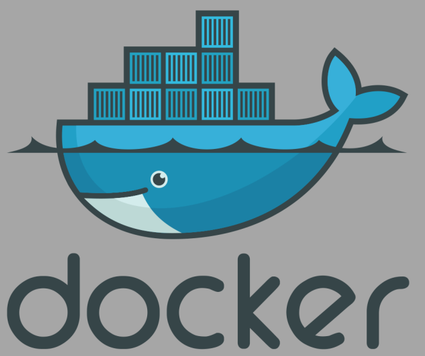
HARDWARE

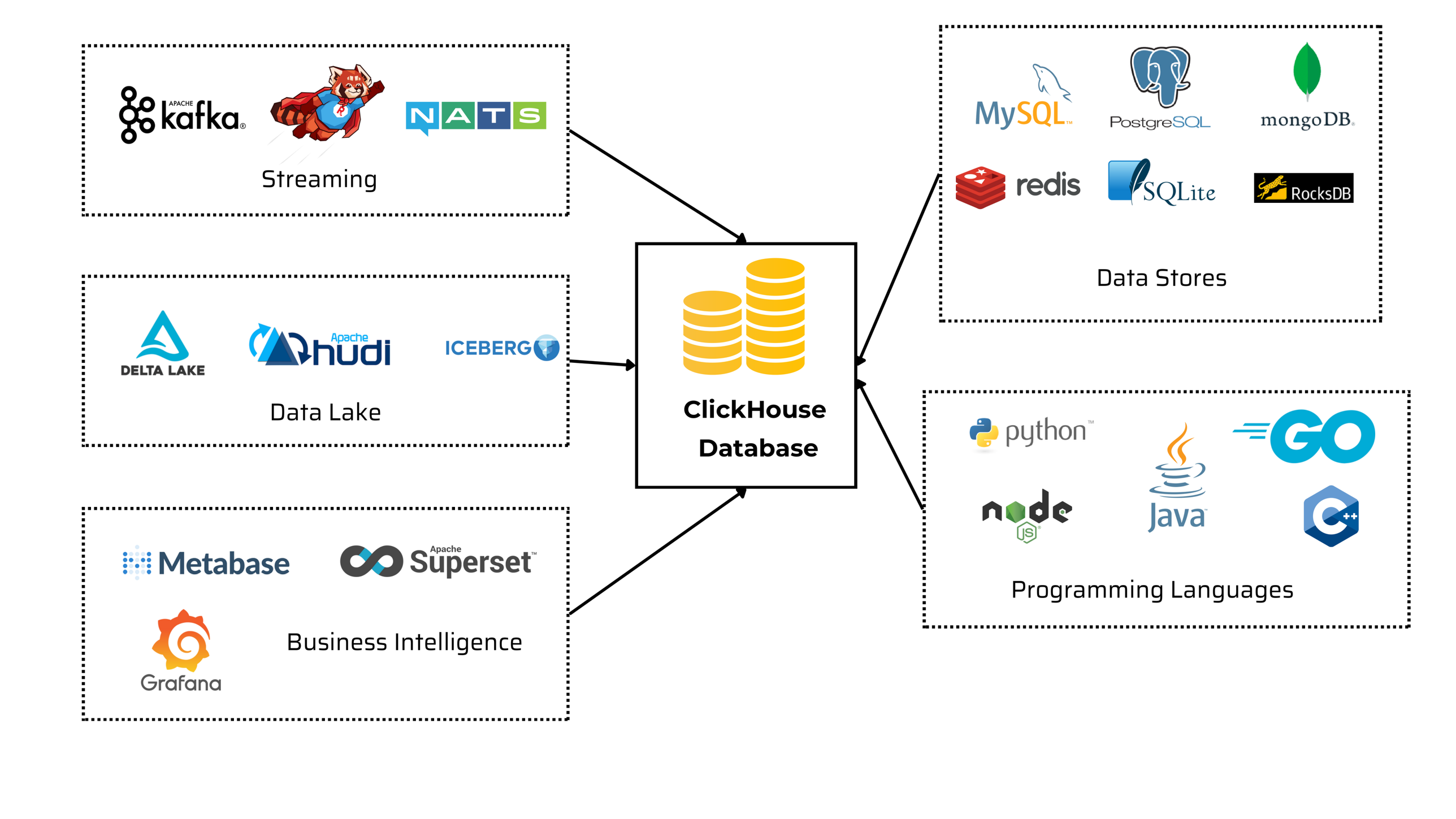
- 8 GB RAM
- 50 GB Hard Disk Space
- Linux (Ubuntu/Debian or Fedora/CentOS) or Mac OSX

SOFTWARE

- Linux commands
- SQL
- Docker and Docker Compose






Streaming
Data Lake
Business Intelligence
ClickHouse Database
Data Stores
Programming Languages

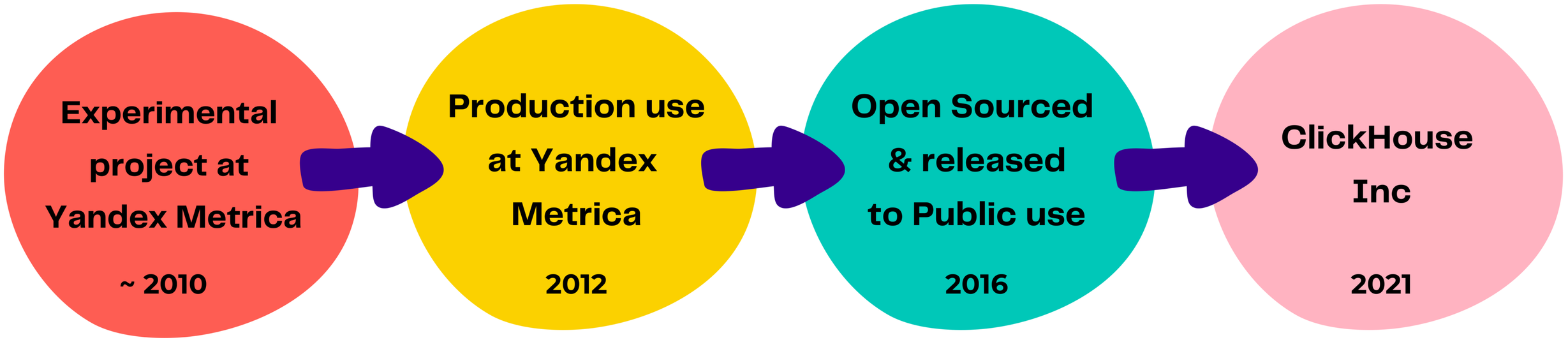
GETTING STARTED WITH

ClickHouse

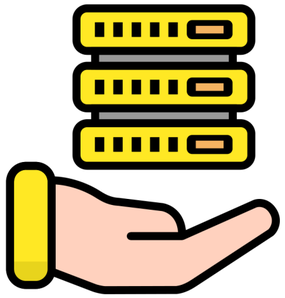
GoldenClover
Education

www.learn-olapdb.com

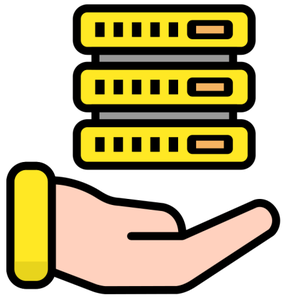
HISTORY


Experimental project at Yandex Metrica
~ 2010
Production use at Yandex Metrica
2012
Open Sourced & released to Public use
2016
ClickHouse Inc
2021

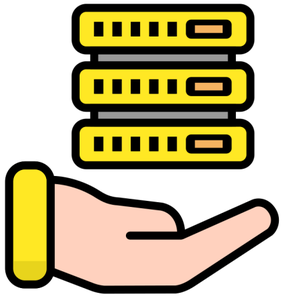
YANDEX METRICA



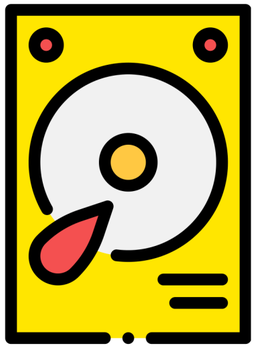
630 Nodes



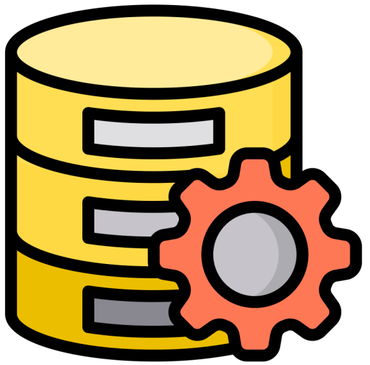
360 Nodes



1860+ Nodes



118 PB



120 Trillion Rows

Row vs Column-Oriented Data Stores

Row	Name	ID	City	Age
1	Name 1	1	City 1	32
2	Name 2	2	City 2	25
3	Name 3	3	City 3	28
x	...	...	...	...

1. Tabular data

File 1	Name	Name 1	Name 2	Name 3	...
File 2	ID	1	2	3	...
File 3	City	City 1	City 2	City 3	...
File 4	Age	32	25	28	...

2. Tabular data stored in column oriented data store

KEY FEATURES

01
SPEED

02
SQL SUPPORT

03
SCALABLE

04
RELIABLE

05
INTEGRATIONS

06
COMMUNITY

WHEN TO USE CLICKHOUSE DB?

- Write << Read & No or very small amounts of updates/Deletes
- Data is inserted in large batches
- Tables have a large number of columns
- Data is filtered or aggregated while querying
- No Transactions

LETS GET STARTED

INSTALLING

ClickHouse

Golden Clover Education

www.learn-olapdb.com

ClickHouse

Officially Supported on Linux and Mac OS

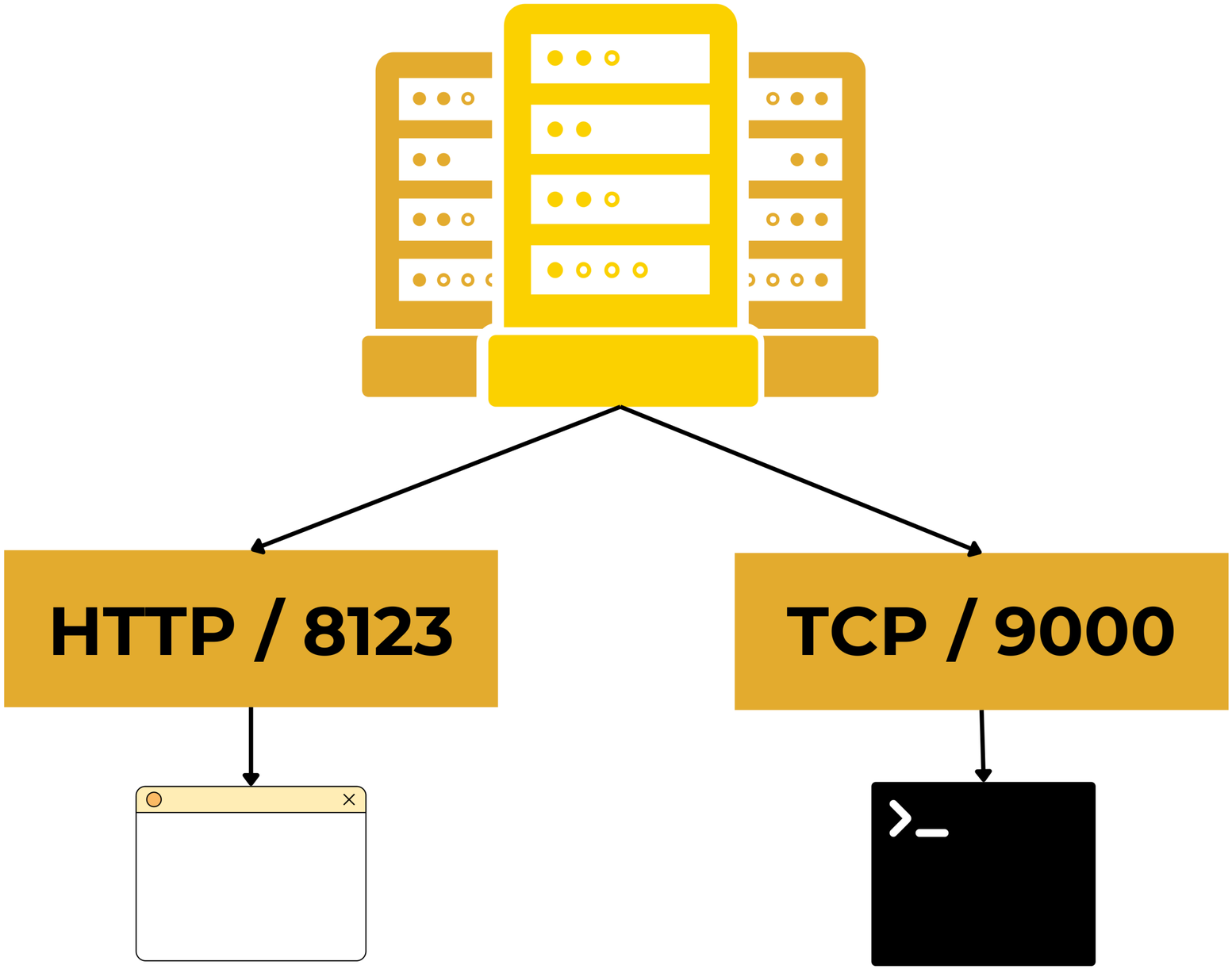
Self managed

- Quick Install
- Install using Precompiled Packages
- Docker
- Kubernetes

Fully Managed

- Altinity Cloud
- ClickHouse Cloud
- ChistaDATA Cloud
- DoubleCloud
- Aiven
- Yandex Cloud

ClickHouse



COMMON PARAMETERS

-- host

-- port

-- user

-- password

-- query

-- multiquery

-- multiline

-- database

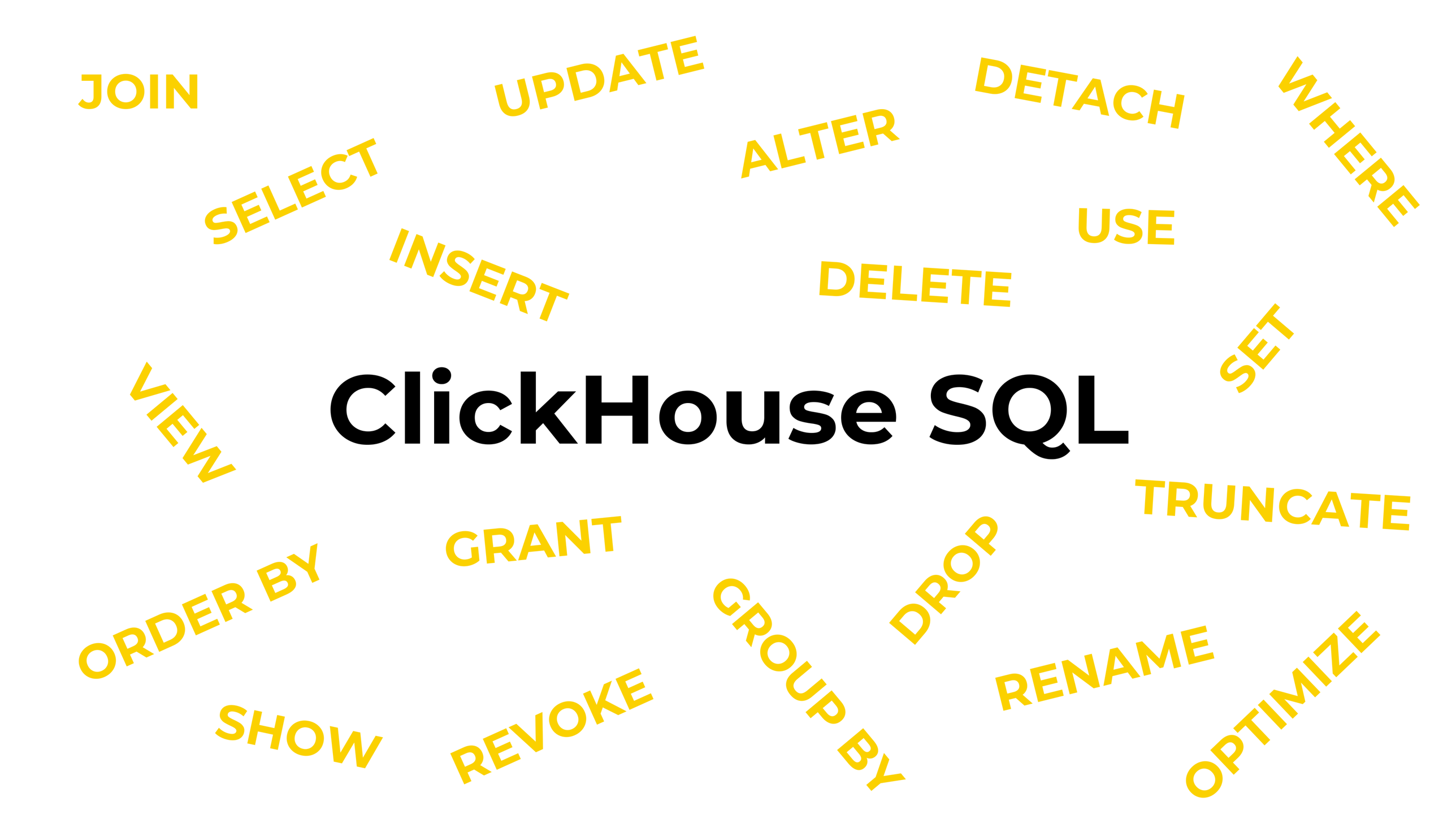
-- format

-- secure

ClickHouse

Golden Clover Education

www.learn-olapdb.com


JOIN
UPDATE
DETACH
WHERE
SELECT
ALTER
USE
INSERT
DELETE
SET
VIEW
ClickHouse SQL
TRUNCATE
ORDER BY
GRANT
DROP
GROUP BY
RENAME
OPTIMIZE
SHOW
REVOKE

MergeTree
Table Engines

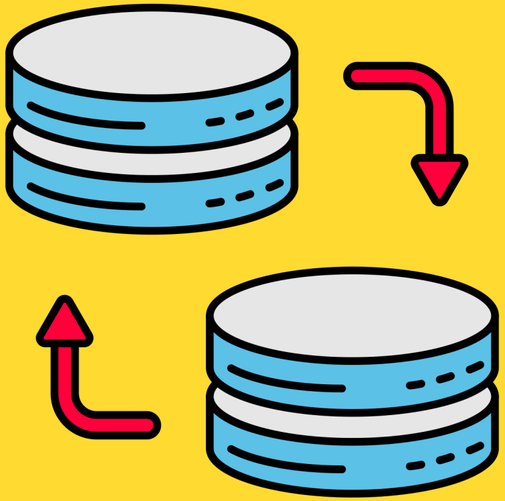
ReplacingMerge Tree

CollapsingMergeTree

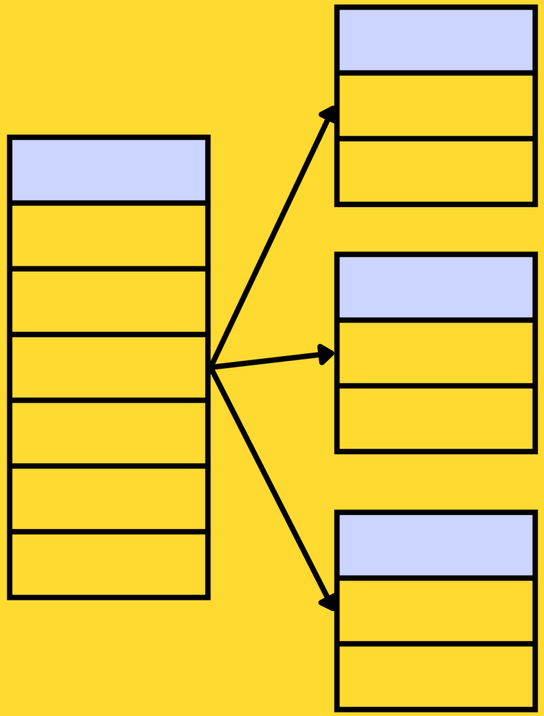
SummingMergeTree

VersionedCollapsing MergeTree

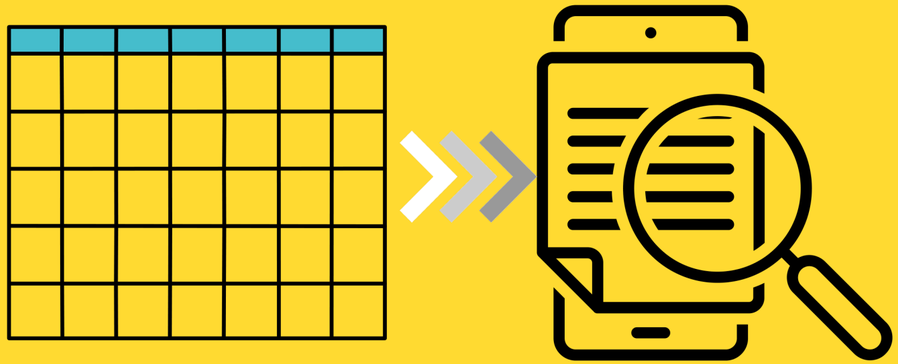
MergeTree



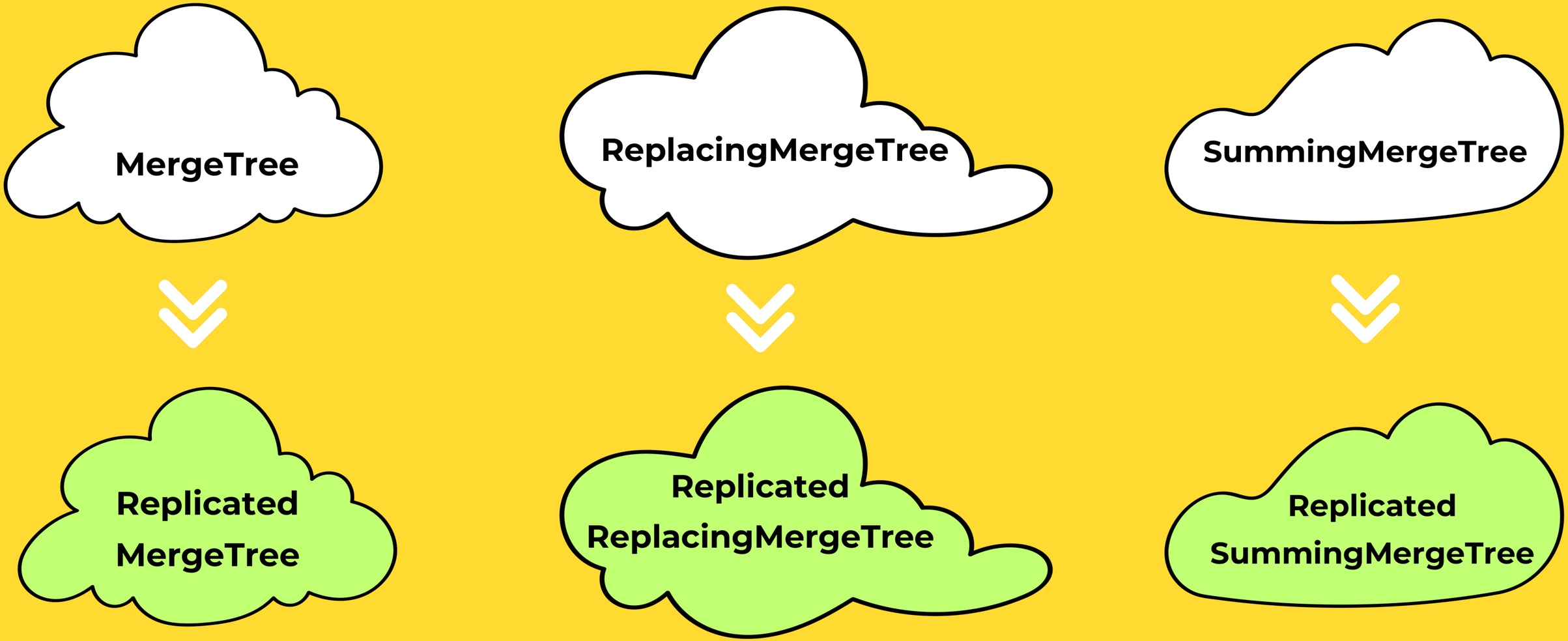
Data Replication



Sharding



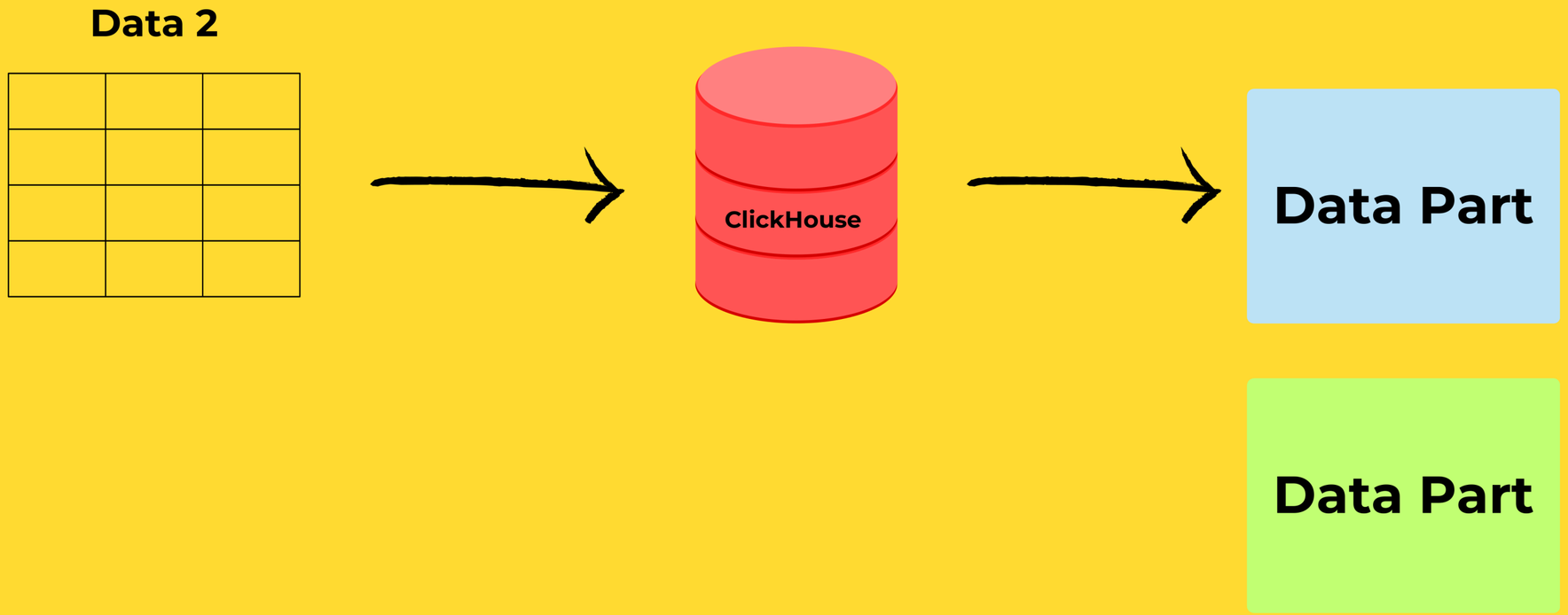
Indexing


MergeTree
ReplacingMergeTree
SummingMergeTree
Replicated
MergeTree
Replicated
ReplacingMergeTree
Replicated
SummingMergeTree

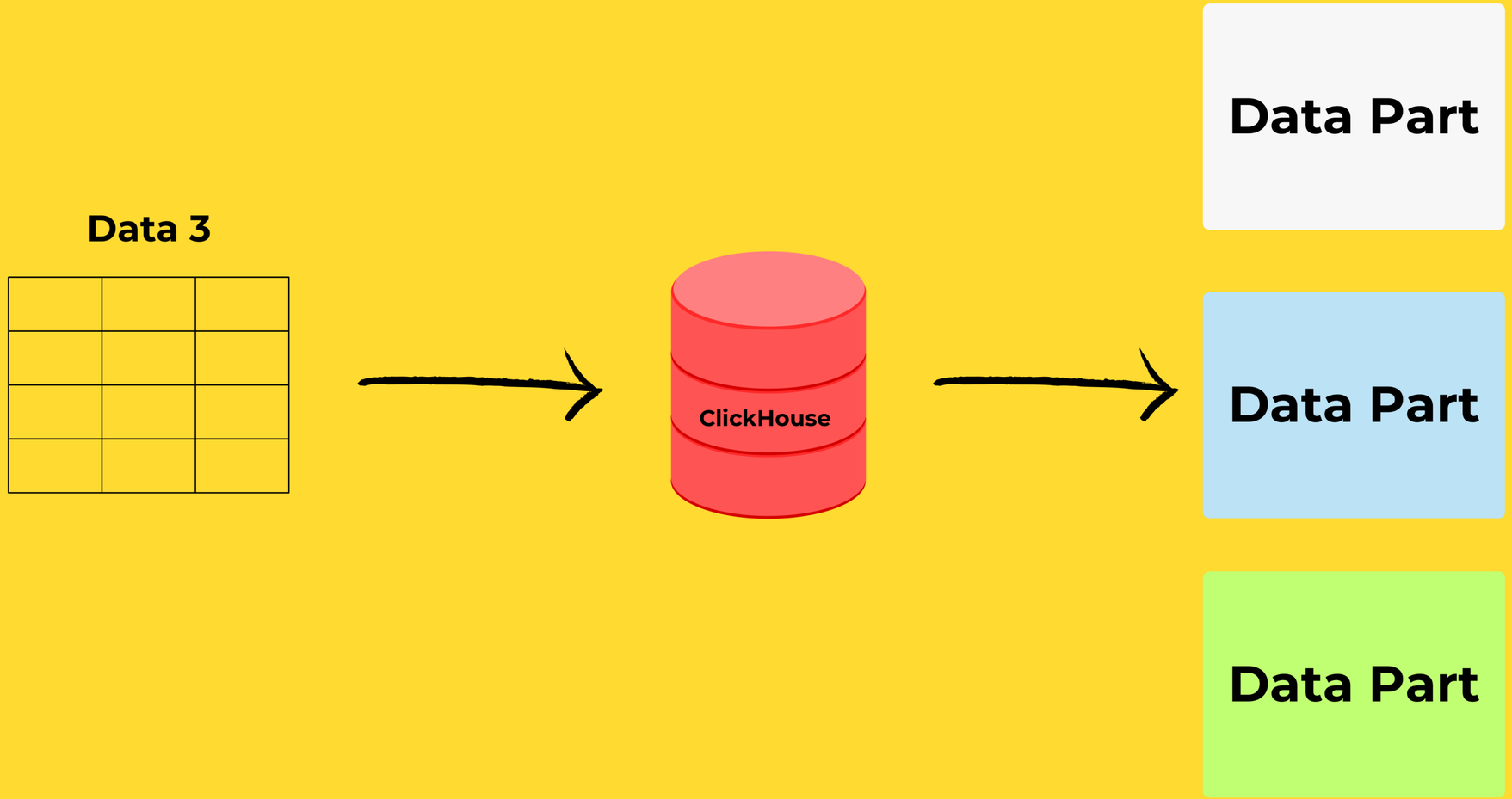

CollapsingMergeTree
Versioned CollapsingMergeTree
AggregatingMergeTree
Replicated CollapsingMergeTree
Replicated Versioned Collapsing MergeTree
Replicated AggregatingMergeTree

Data Part

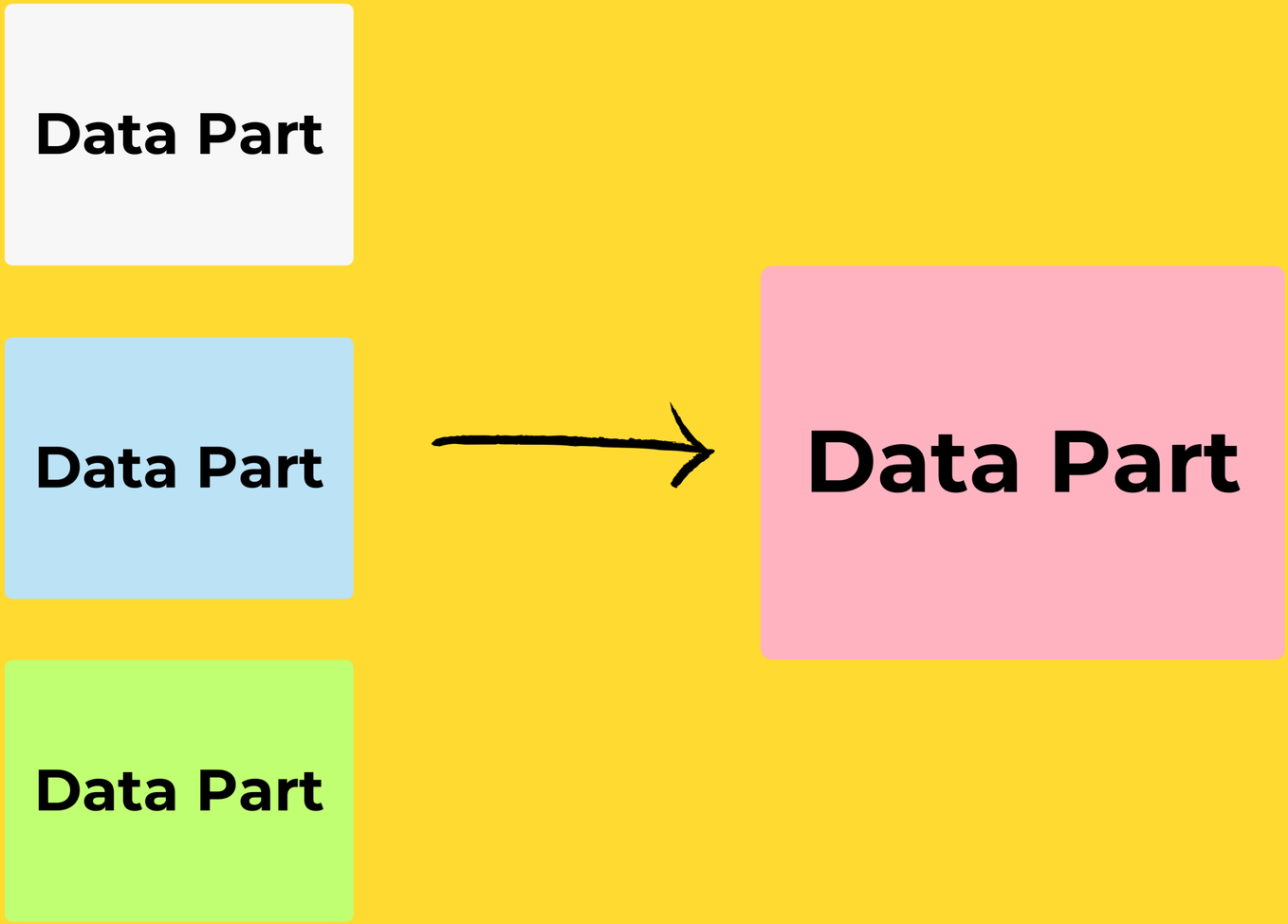
Data parts



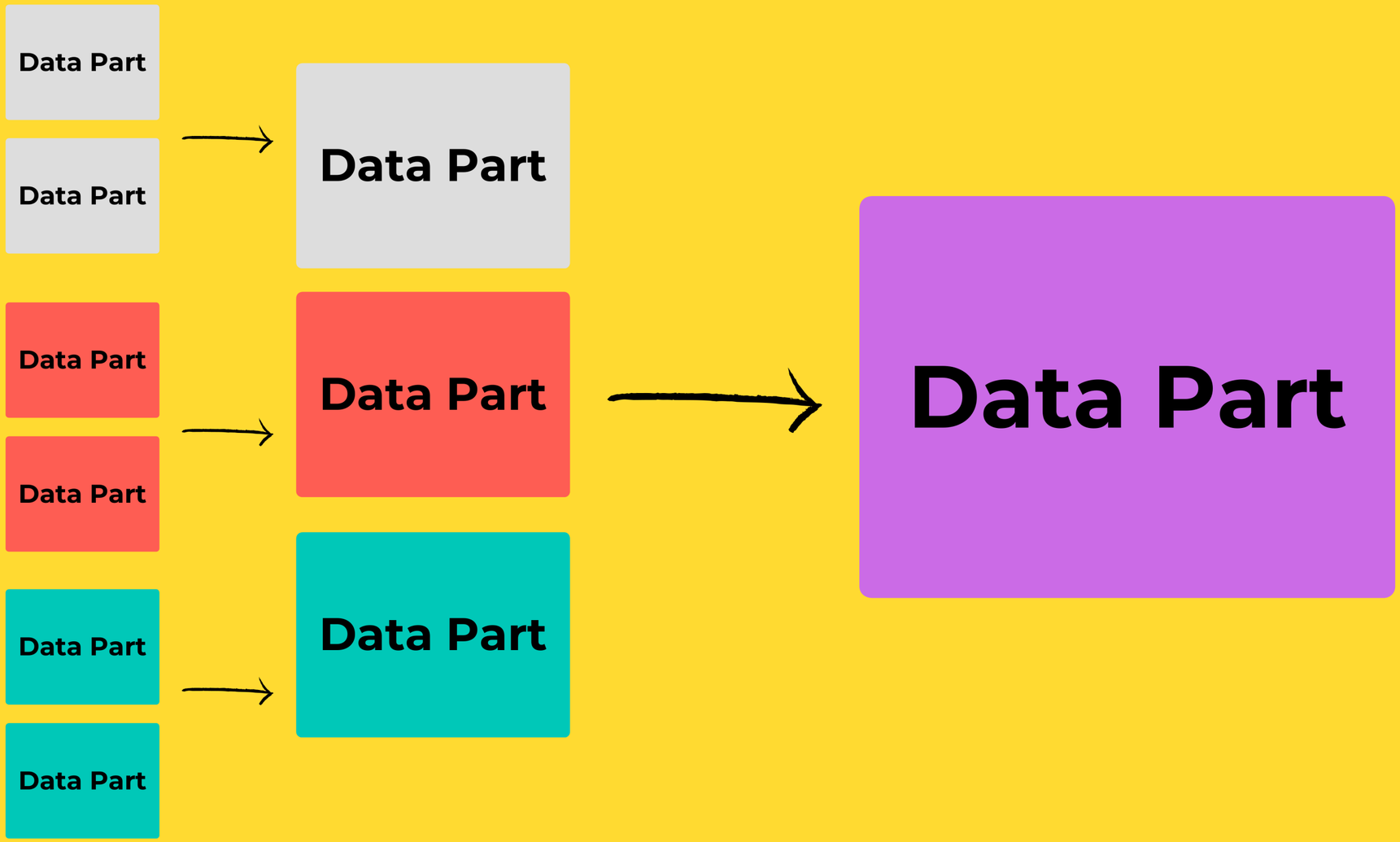
Data parts



Merging the data parts

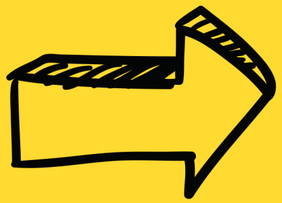


Merging the data parts



MergeTree - Example

ID	Name	Age



```
CREATE TABLE mergetree_example
(
ID Int32,
Name String,
Age UInt8
)
Engine = MergeTree
PRIMARY KEY (ID)
SETTINGS
index_granularity = 4
```

PRIMARY INDEX & Granules

ID	Name	Age
1	k	15
2	t	8
3	a	10
4	l	17
5	c	12
6	g	7
7	q	4
10	e	2

primary.idx
1
5

Inserts & Parts

Insert Data

ID	Name	Age
3	a	10
5	c	12
2	t	8
7	q	4

Sorted Data

ID	Name	Age
2	t	8
3	a	10
5	c	12
7	q	4

Stored in Disk

mergetree_example

Data Part 1

Inserts & Parts

Insert Data

ID	Name	Age
10	e	2
6	g	7
1	k	15
4	l	17

Sorted Data

ID	Name	Age
1	k	15
4	l	17
6	g	7
10	e	2

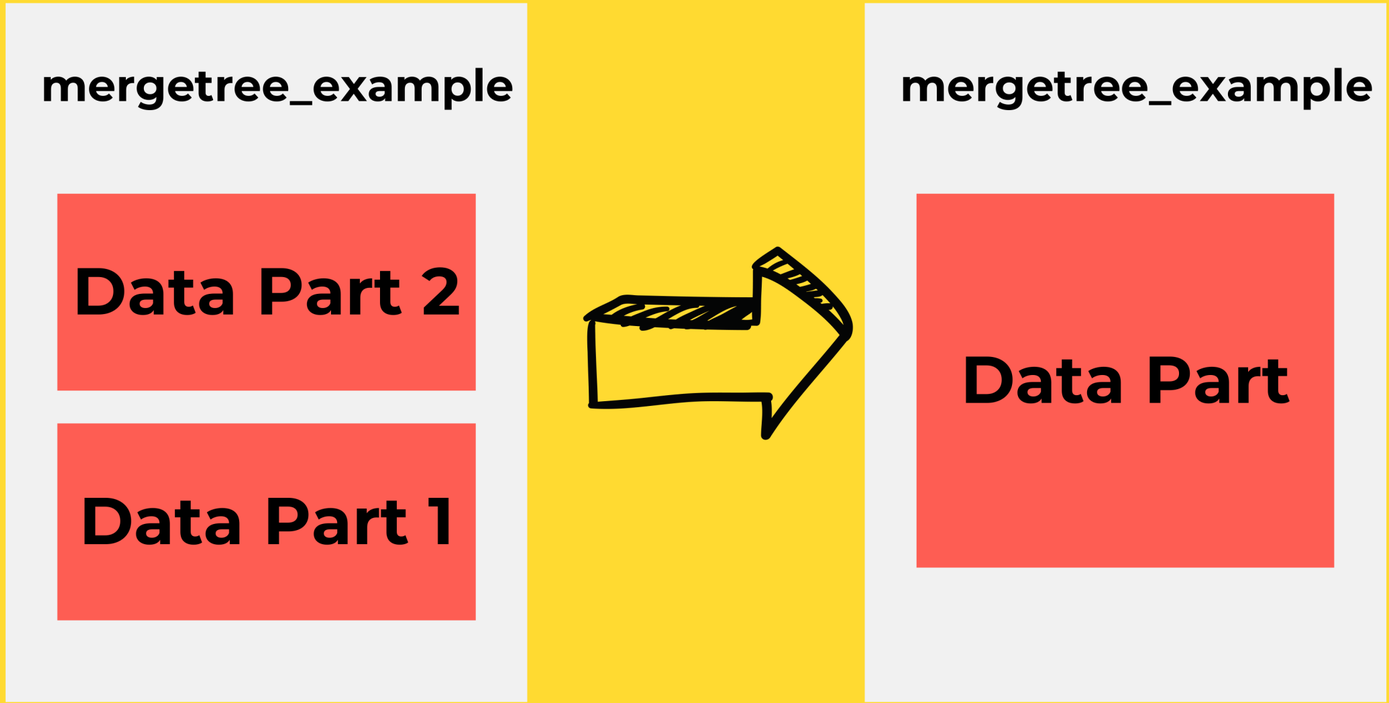
Stored in Disk

mergetree_example

Data Part 2

Data Part 1

Merging of Data Parts



Merging of Data Parts

Data part 1

ID	Name	Age
1	k	15
4	l	17
6	g	7
10	e	2

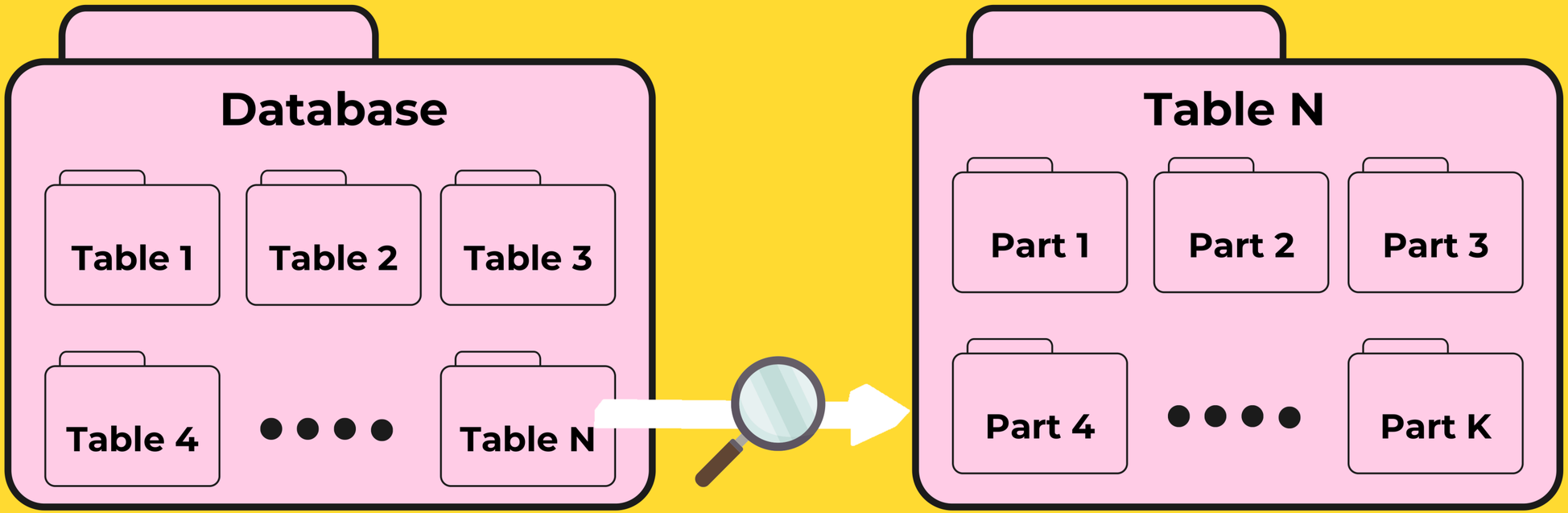
Data part 2

ID	Name	Age
2	t	8
3	a	10
5	c	12
7	q	4

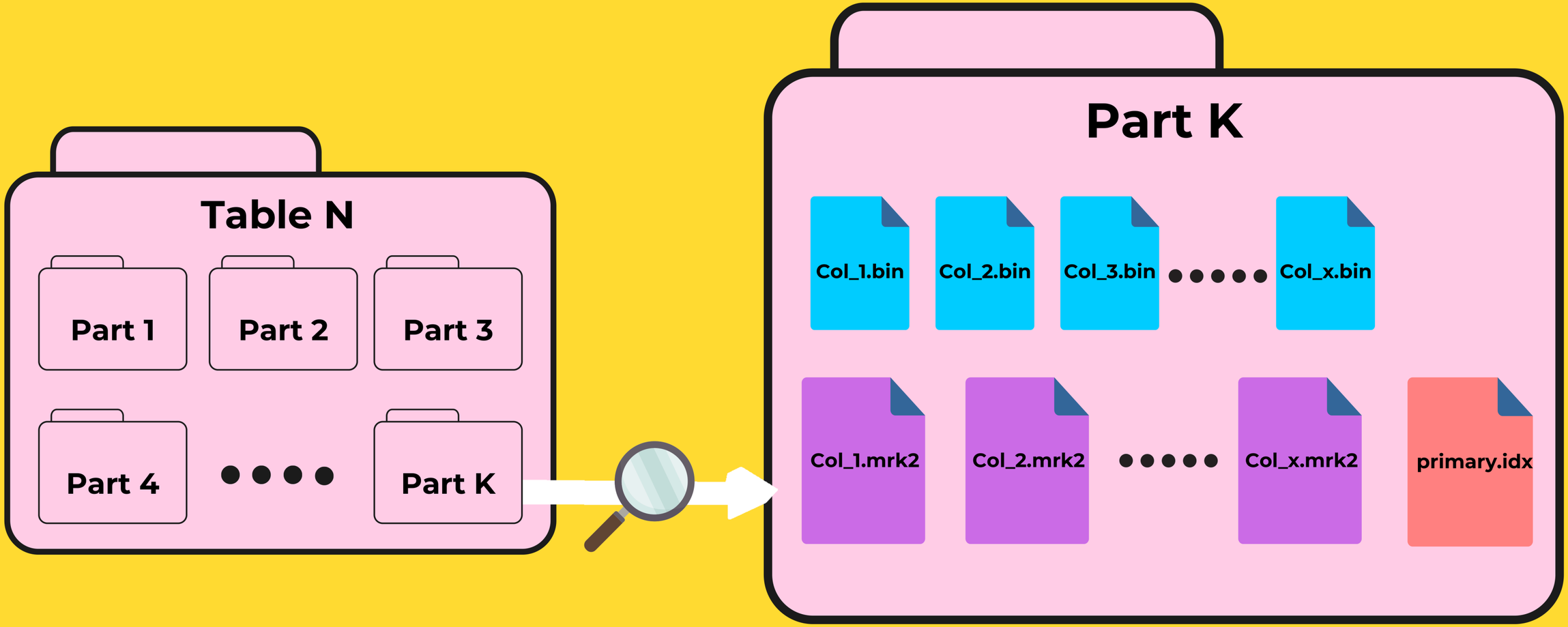
Data Part

ID	Name	Age
1	k	15
2	t	8
3	a	10
4	l	17
5	c	12
6	g	7
7	q	4
10	e	2

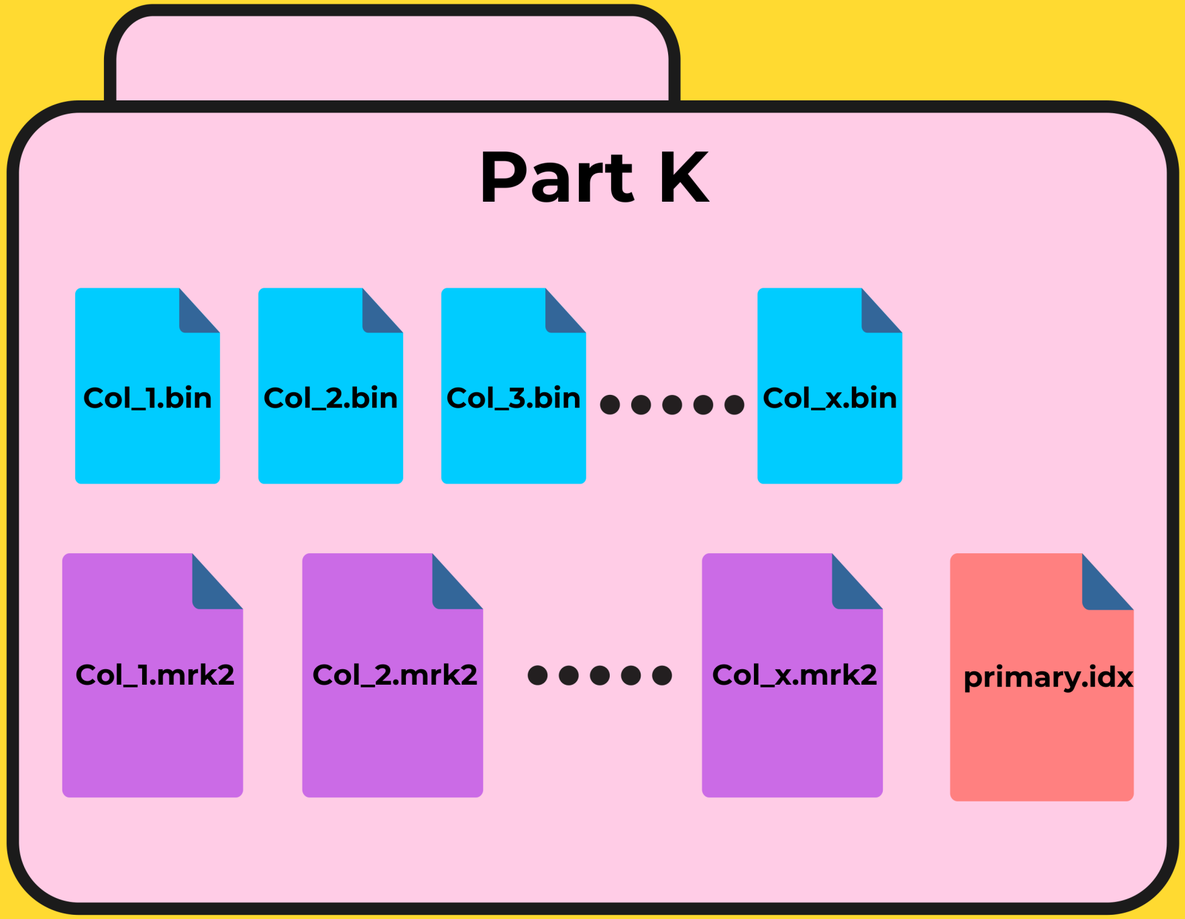
ClickHouse Data Directory



ClickHouse Data Directory



ClickHouse Data Directory



min_bytes_for_wide_part

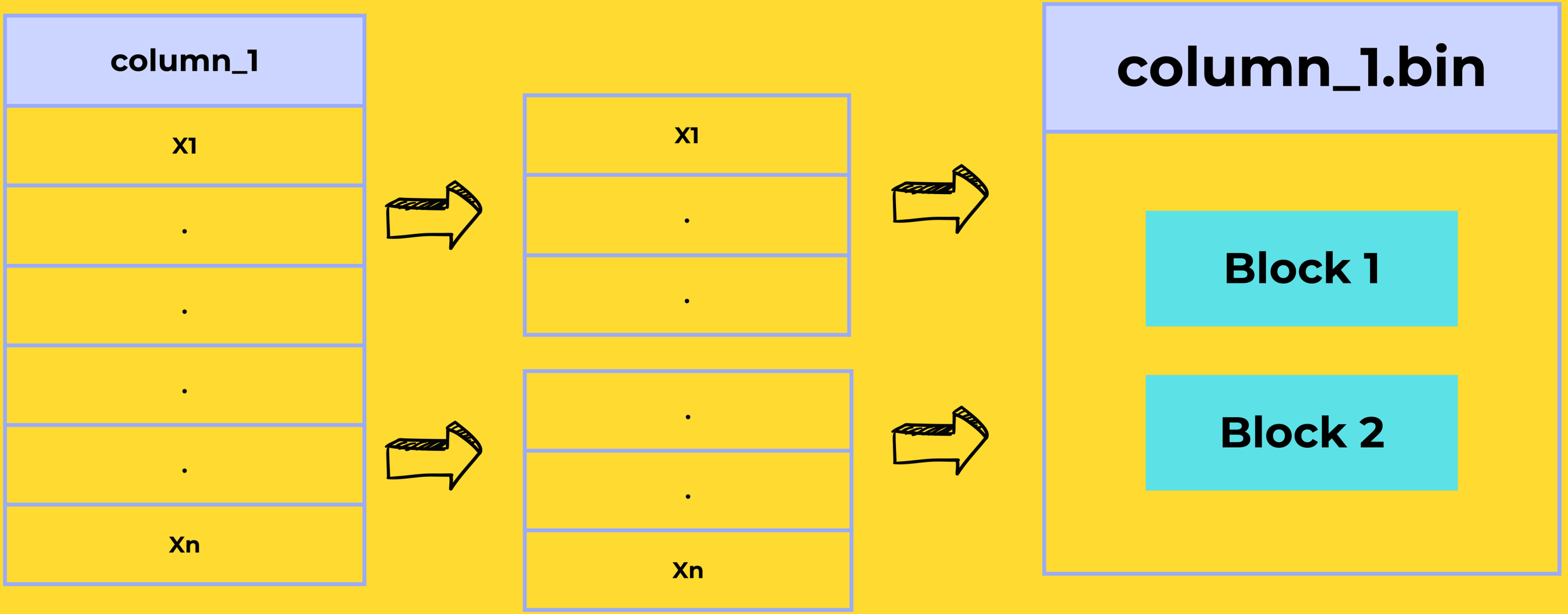
PRIMARY KEY & Granules

ID	Name	Age
1	k	15
2	t	8
3	a	10
4	l	17
5	c	12
6	g	7
7	q	4
10	e	2

primary.idx
1
5

index_granularity_bytes

<column>.bin & Blocks



ClickHouse Compression Codecs

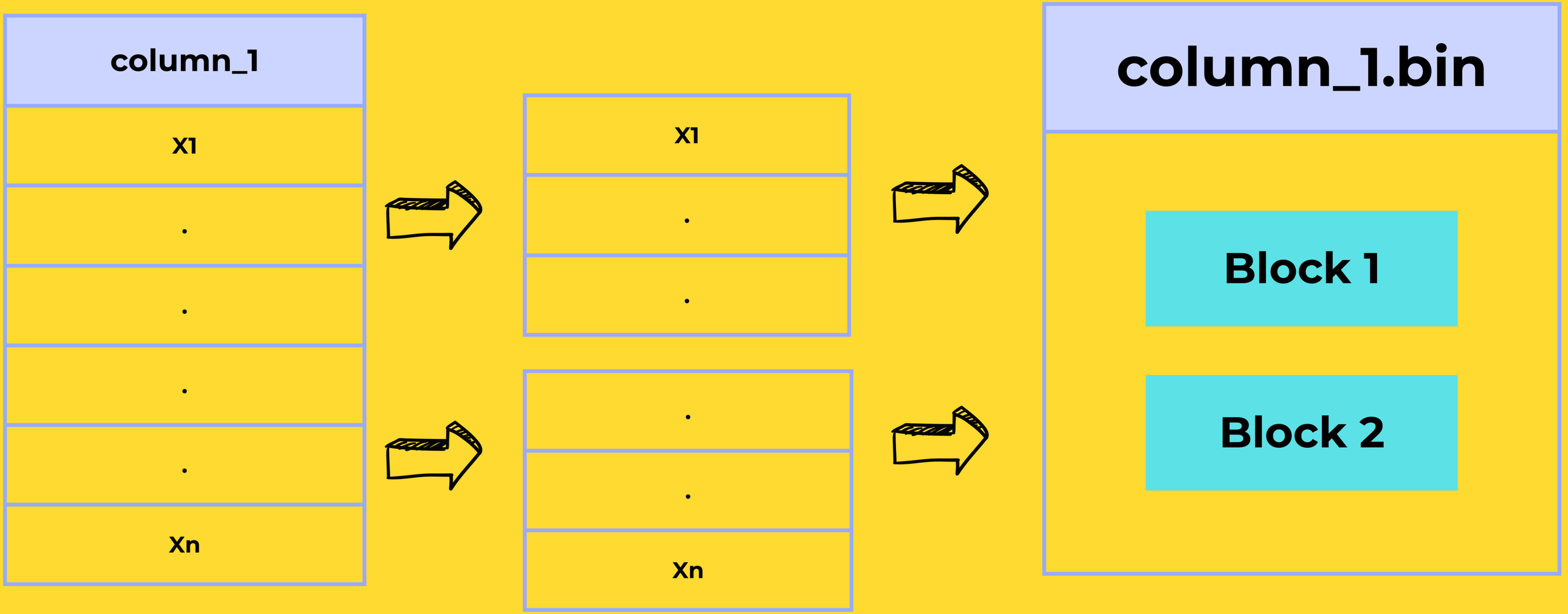
General Purpose Codecs

- LZ4
- ZSTD
- LZ4HC

Special Codecs

- Delta
- Double Delta
- Gorilla
- GCD
- FPC
- T64

<column>.bin & Blocks



Granules	Blocks
Defined by the number of rows	*Defined by the data size*
The first row of the primary key for every granule is stored in primary.idx file	*Every block of uncompressed data is compressed (by default) and stored in columnar files*
index_granularity	*max_compress_block_size, min_compress_block_size*

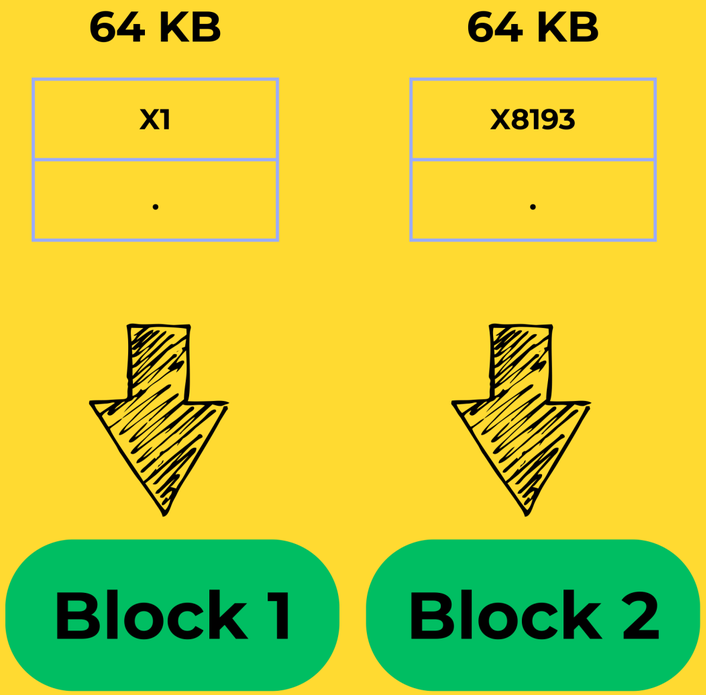
Blocks Example - Int16

16 KB	16 KB	16 KB	16 KB
X1	X8193	X16385	X24577
.	.	.	.

Block 1

- min_compress_block_size - 64 KB
- max_compress_block_size - 1 MB
- index_granularity - 8192

Blocks Example - Int64



- min_compress_block_size - 64 KB
- max_compress_block_size - 1 MB
- index_granularity - 8192

Blocks Example - String

- min_compress_block_size - 64 KB
- max_compress_block_size - 1 MB
- index_granularity - 8192

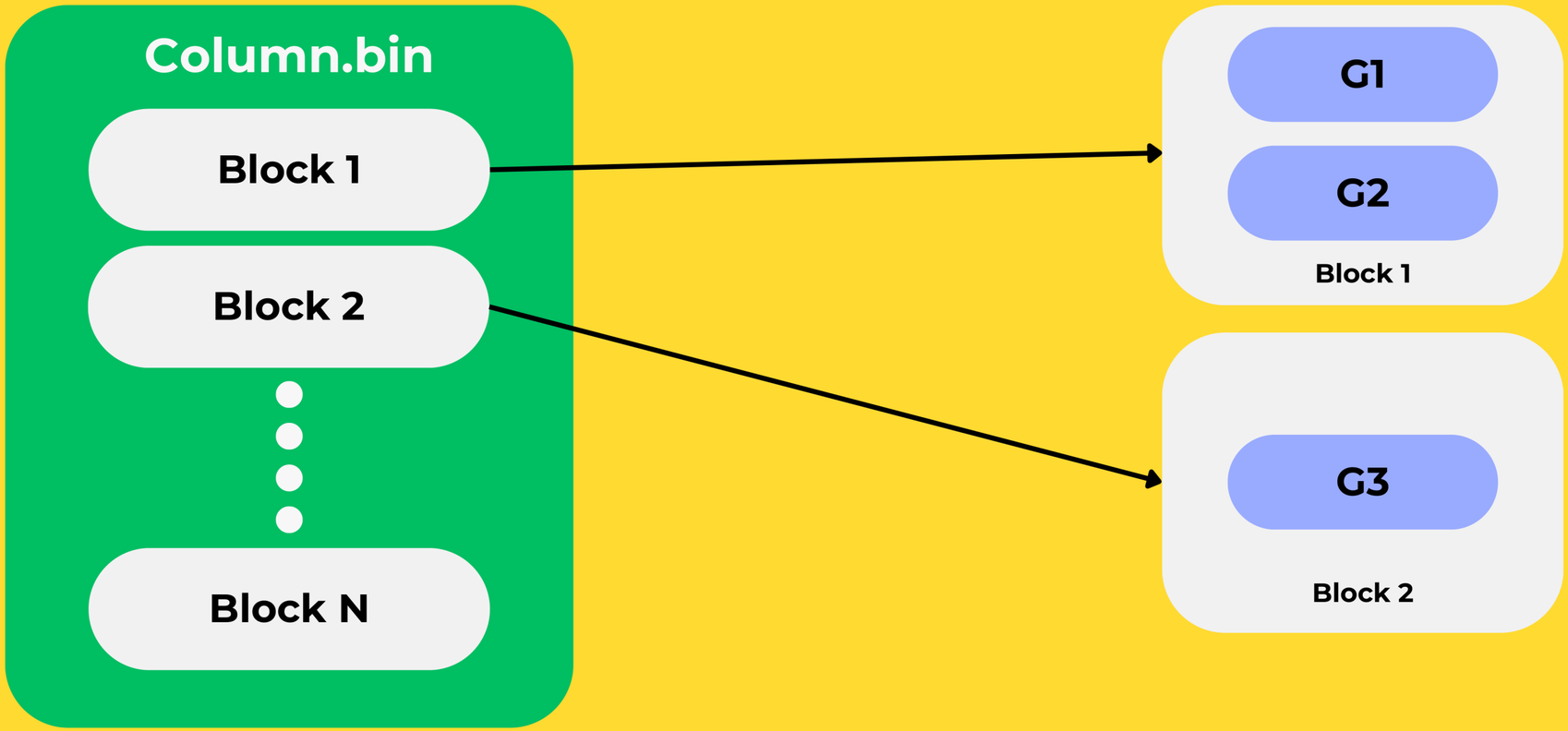
1536 KB

X1
.

Block 1

Block 2

Offsets in Bin File



Mark File & Offsets

primary.idx	GRANULE
x1, y1	1
x2, y2	2
x3, y3	3
x4, y4	4

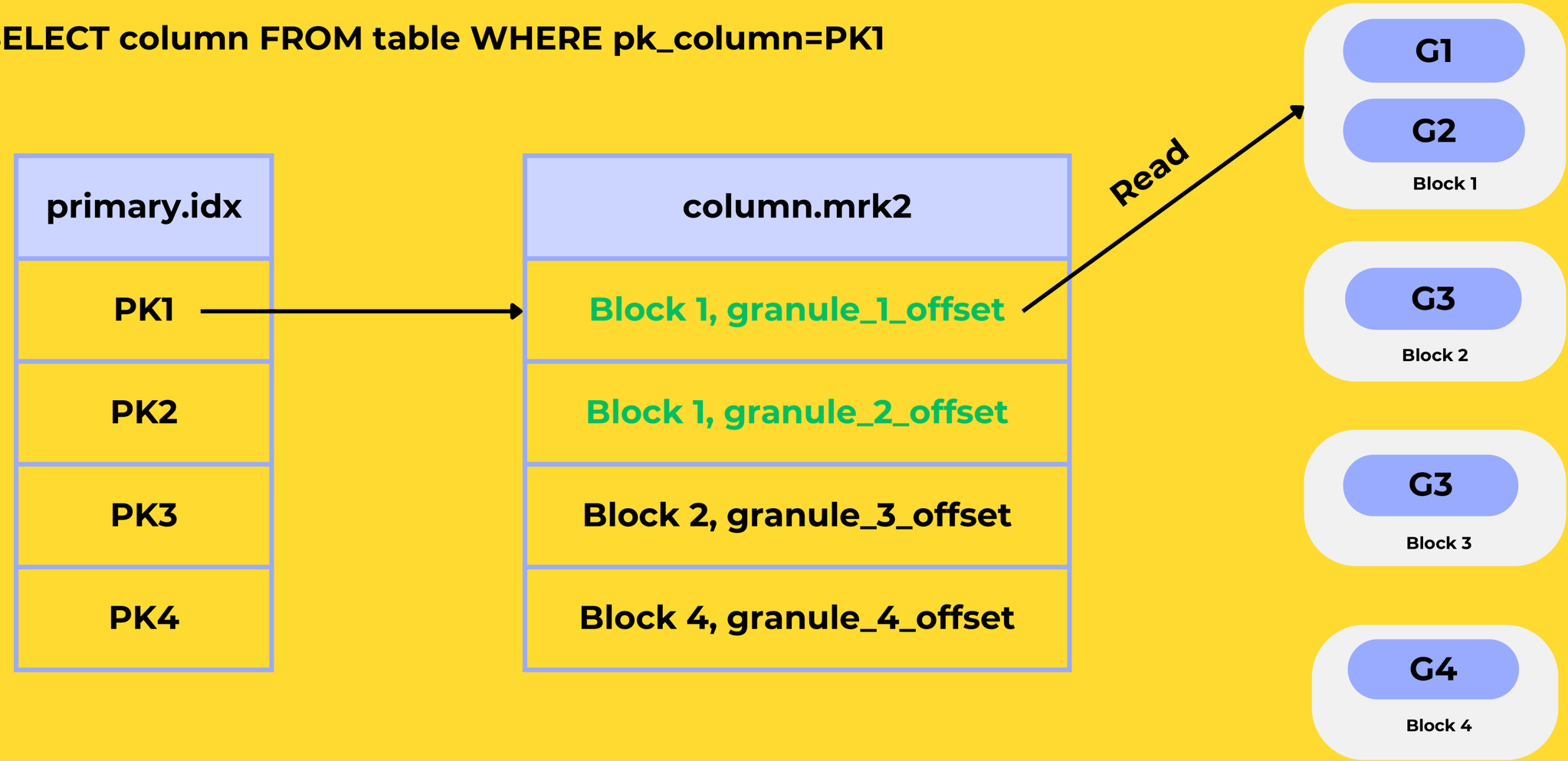
column_x.mrk2
granule_1_block_offset, granule_1_offset
granule_2_block_offset, granule_2_offset
granule_3_block_offset, granule_3_offset
granule_4_block_offset, granule_4_offset

Mark Files

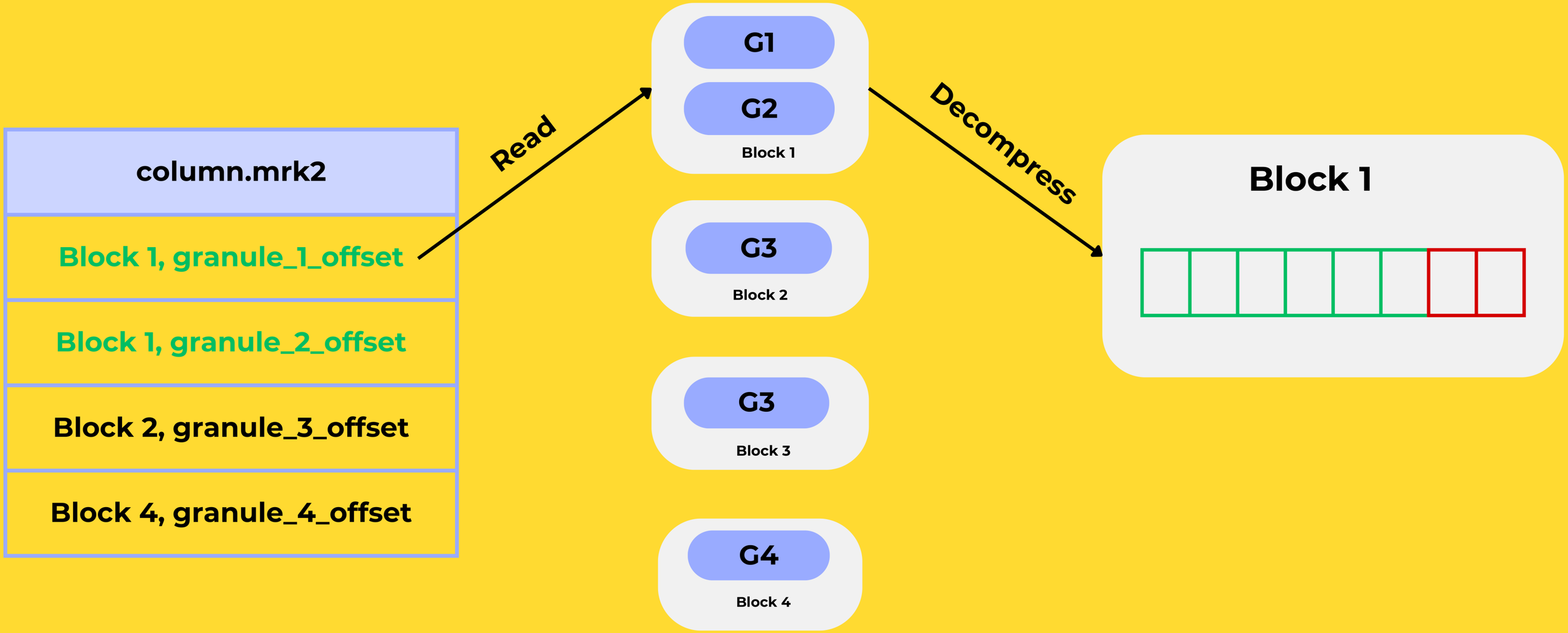
File Type	Data Format	Index Granularity	compress_marks = 1
mrk	Wide	Fixed	cmrk
mrk2	Wide	Adaptive	cmrk2
mrk3	Compact	Fixed or Adaptive	cmrk3

Finding Granule 1

SELECT column FROM table WHERE pk_column=PK1

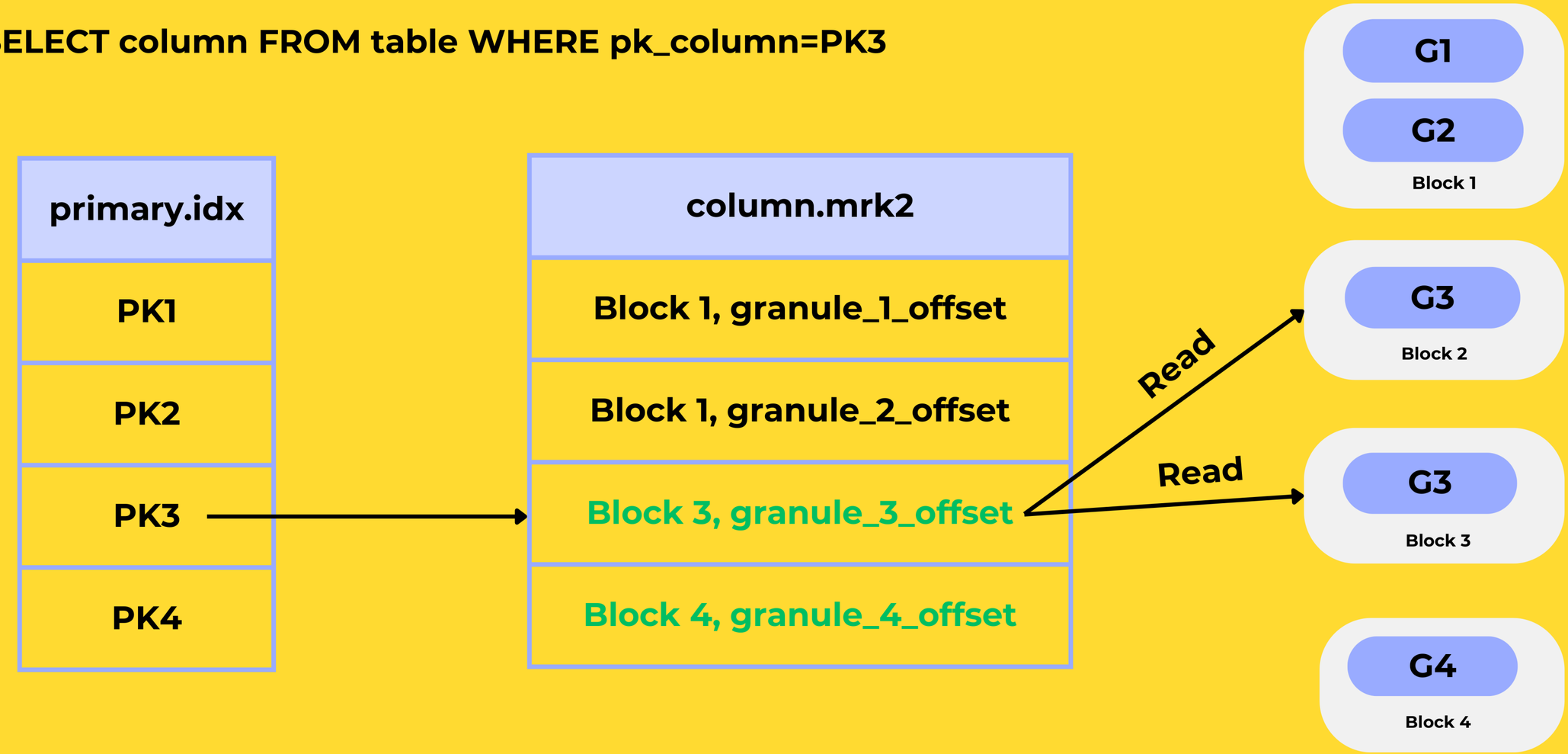


Finding Granule 1

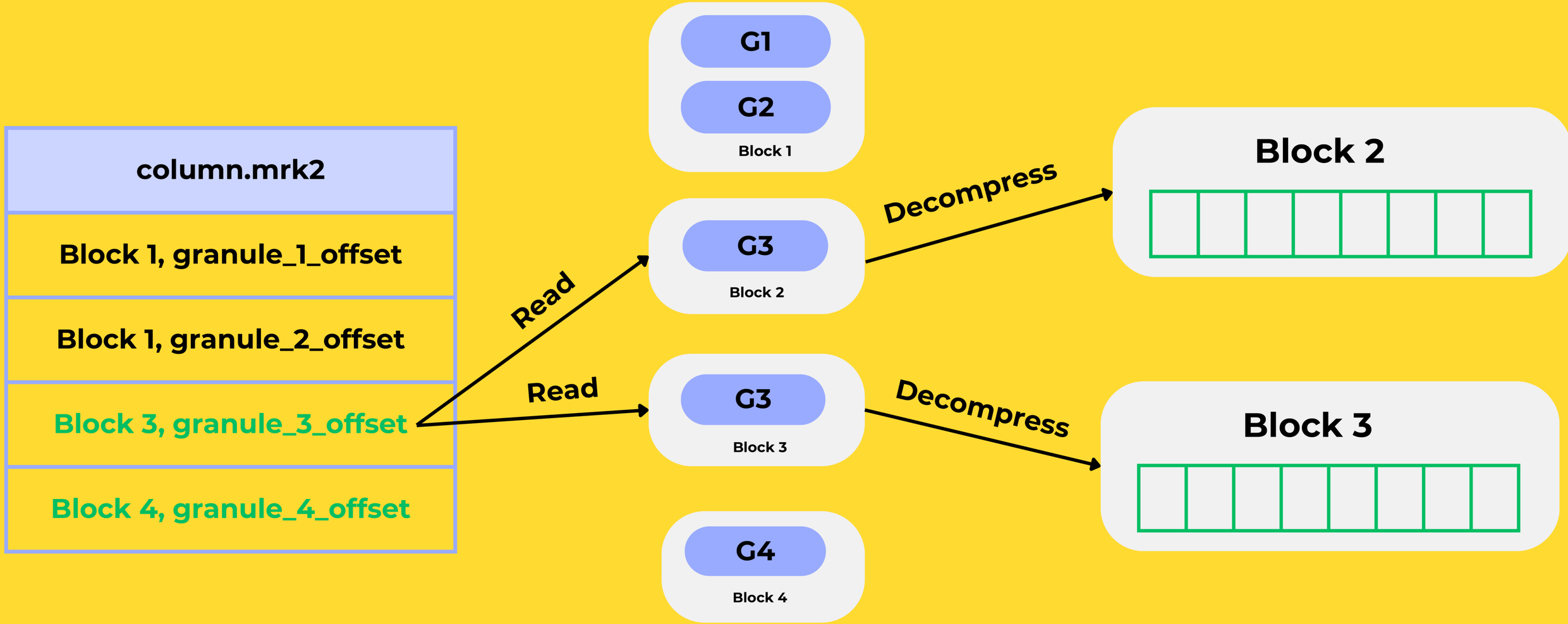


Finding Granule 3

SELECT column FROM table WHERE pk_column=PK3

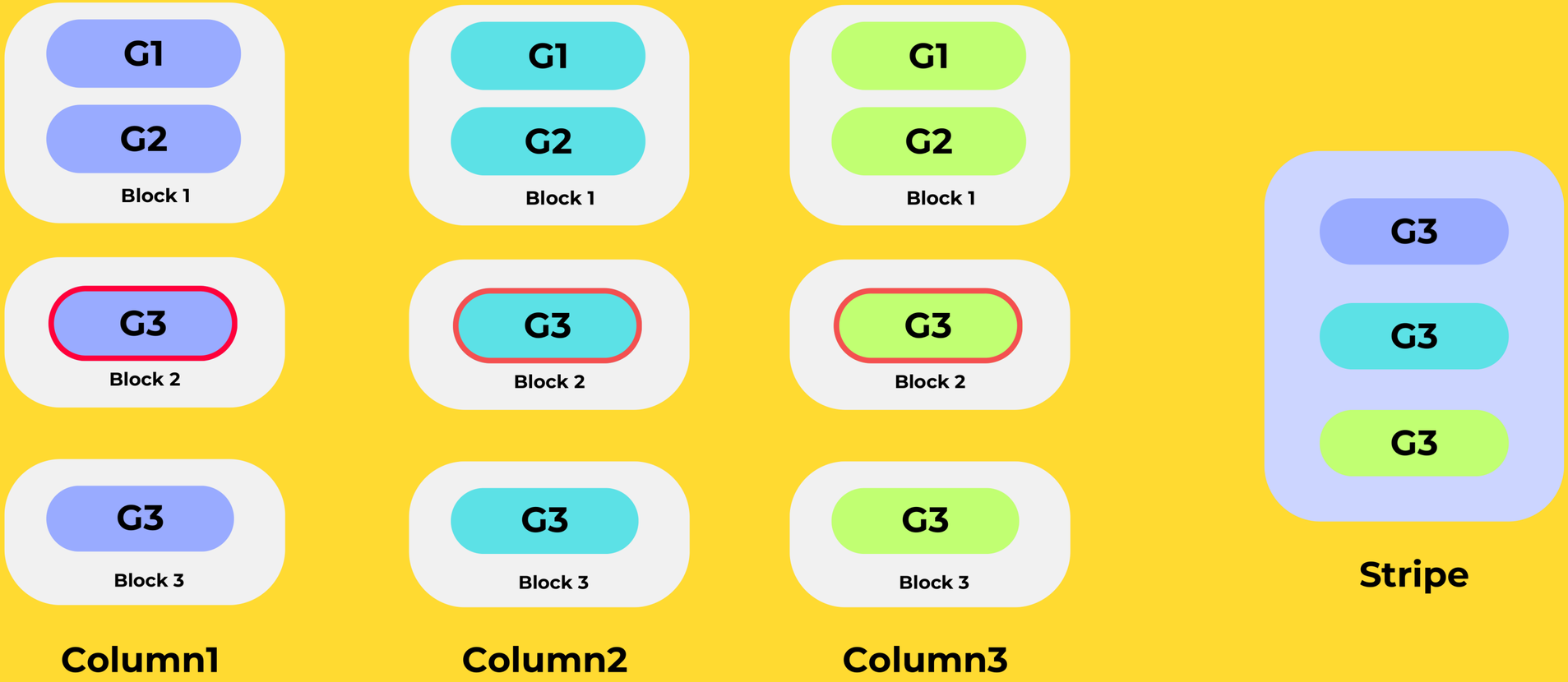


Finding Granule 3



Stripe of granules

SELECT Column1, Column2, Column3 FROM table WHERE pk_column=PK



Primary key vs ORDER BY

PRIMARY KEY	ORDER BY
CREATE TABLE mergetree_example_pk (C1 String, C2 Int32, C3 Float64 C4 DateTime) Engine = MergeTree() PRIMARY KEY (C1, C2, C3)	CREATE TABLE mergetree_example_order_by (C1 String, C2 Int32, C3 Float64 C4 DateTime) Engine = MergeTree() ORDER BY (C1, C2, C3)

Primary key vs ORDER BY

Condition	Data Sorted By	Primary Index
PRIMARY KEY (C1, C2)	C1 and C2	(C1, C2)
ORDER BY (C3, C2)	C3 and C2	(C3, C2)
PRIMARY KEY (C1, C2) ORDER BY (C1, C2, C3)	C1, C2 and C3	(C1, C2)

Mergetree CREATE TABLE statement - Syntax

CREATE TABLE [IF NOT EXISTS] [database_name.]table_name [ON CLUSTER cluster_name]
(
column1 [datatype1] [DEFAULT|MATERIALIZED|ALIAS expr1] [TTL expr1],
column2 [datatype2] [DEFAULT|MATERIALIZED|ALIAS expr2] [TTL expr2],
...
...
)
ENGINE = MergeTree()
[PARTITION BY expr]
[ORDER BY expr]
[PRIMARY KEY expr]
[SAMPLE BY expr]
[TTL expr]
[SETTINGS name=value, ...]

Mergetree settings

Setting	Default	Unit
index_granularity	8192	Rows
index_granularity_bytes	1024,000	Bytes
min_index_granularity_bytes	1024	Bytes
merge_with_ttl_timeout	14400	Seconds
write_final_mark	1	NA
merge_max_block_size	8192	Rows

Mergetree settings

Setting	Default	Unit
min_bytes_for_wide_part	1,024,000	Bytes
max_compress_block_size	1,048,576	Bytes
min_compress_block_size	65,536	Bytes
max_partitions_to_read	-1	Unlimited

Mergetree settings

Setting	Default	Unit
max_suspicious_broken_parts	100	No Unit
parts_to_throw_insert	3000	No Unit
parts_to_delay_insert	1000	No Unit
max_delay_to_insert	1	Millisecond
max_parts_in_total	100000	No Unit

Mergetree settings

Setting	Default	Unit
max_suspicious_broken_parts_bytes	1,073,741,824	Bytes
max_files_to_modify_in_alter_columns	75	No Unit
max_files_to_remove_in_alter_columns	50	No Unit
old_parts_lifetime	480	Seconds
clean_deleted_rows	Never	No Unit
max_part_loading_threads	CPU Cores	No Unit

Operations on Partitions

- Freeze and Unfreeze
- Attach and Detach
- Move to table
- Replace
- Update
- Delete
- Drop

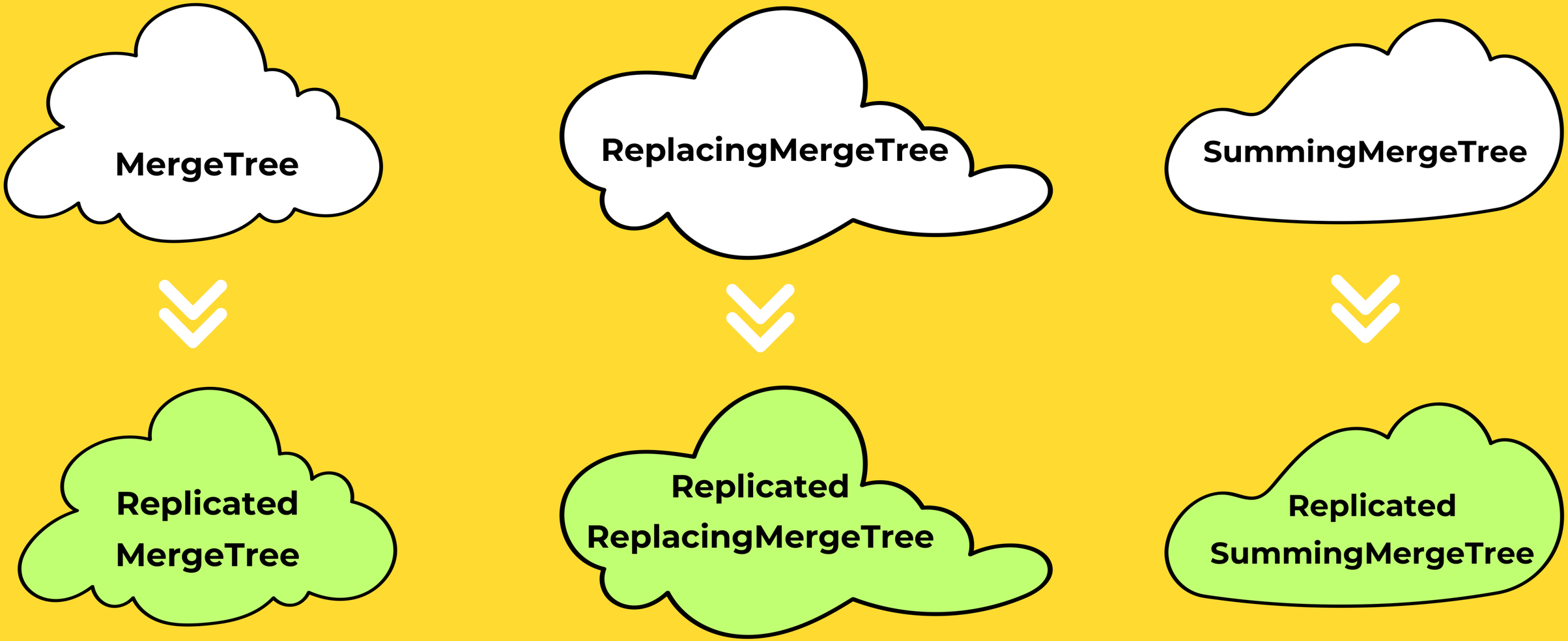
MergeTree
Table Engines

ReplacingMerge Tree

CollapsingMergeTree

SummingMergeTree

VersionedCollapsing MergeTree


MergeTree
ReplacingMergeTree
SummingMergeTree
Replicated
MergeTree
Replicated
ReplacingMergeTree
Replicated
SummingMergeTree


CollapsingMergeTree
Versioned CollapsingMergeTree
AggregatingMergeTree
Replicated CollapsingMergeTree
Replicated Versioned Collapsing MergeTree
Replicated AggregatingMergeTree