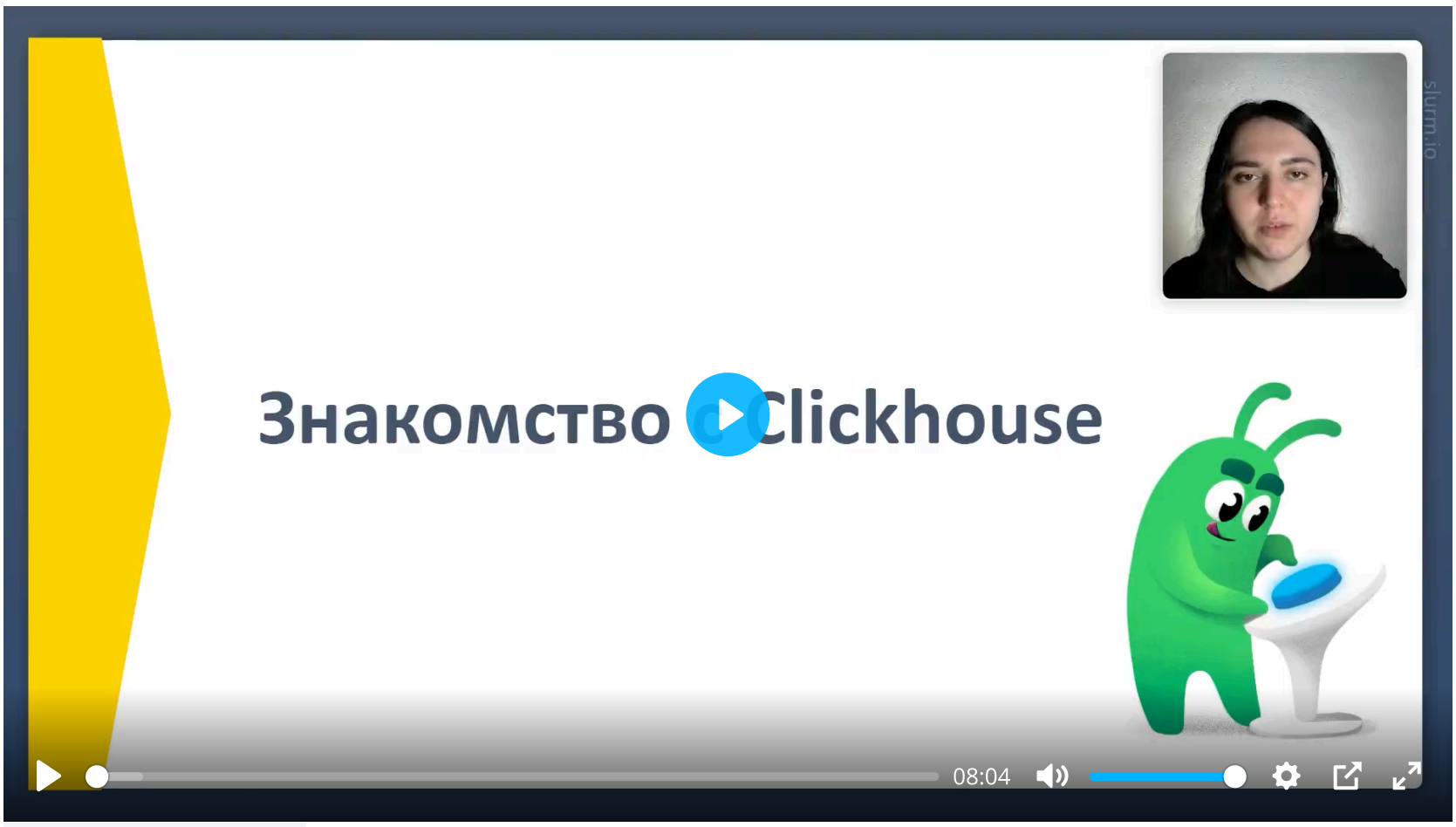
6.3 Практика работы с NoSQL - Clickhouse, Neo4j и Mongo



Презентация к уроку 6.3.1

Код

Прикладная работа с Clickhouse

```
sudo docker run --rm -e CLICKHOUSE_DB=my_database -e CLICKHOUSE_USER=slurm -e CLICKHOUSE_DEFAULT_ACCESS_MANAGEMENT=1 -e CLICKHOUSE_PASSWORD=slurm -p 9000:9000/tcp clickhouse/clickhouse-server:22.9.7
```

Получили контейнер с Clickhouse.

```
sudo docker exec -it <id_контейнера> clickhouse-client
show tables;
```

Зашли в контейнер.

```
1. CREATE DATABASE idplayer;
2. USE idplayer;
3. CREATE TABLE events (
        account_id UInt64,
    event_type String,
    inserted_at DateTime default now()
    )
    ENGINE MergeTree
    ORDER BY inserted_at;
```

Создаем БД idplayer, выбираем ее и затем создаем таблицу events.

```
sudo docker exec -it <id_контейнера> /bin/bash
cd /var/lib/clickhouse/data
```

Заглянули в контейнер еще раз чтобы увидеть структуру данных.

Текстовая расшифровка видео:

ЗНАКОМСТВО С CLICKHOUSE

План:

- Обзор и знакомство с оркестраторами процессинга.

Обзор и знакомство с оркестраторами процессинга

Clickhouse – это столбцовоориентированная СУБД, написанная на C++.

Изначально Clickhouse был написан для Яндекс Метрики, позже получил большое распространение как OpenSource-продукт.

На сегодняшний день Clickhouse используется в:

- Вконтакте;
- Авито;
- Озон;
- Ebay и т.д.

Чаще всего Clickhouse используется для OLAP-задач.

У нас будет прикладная работа с Clickhouse. Для этого вставляем Clickhouse в контейнер, запускаем команду «**rm**», задаем переменные, название БД, юзера, имя и пароль, порт, название контейнера, сервер (на нем тоже есть встроенный клиент для работы с базой):

```
asya@asya-HP-Pavilion-dv6-Notebook-PC:~$ sudo docker run --rm -e CLICKHOUSE_DB=my_database -e CLICKHOUSE_USER=slu
rm -e CLICKHOUSE_DEFAULT_ACCESS_MANAGEMENT=1 -e CLICKHOUSE_PASSWORD=slurm -p 9000:9000/tcp clickhouse/clickhouse-
server:22.9.7
```

По команде «Run» запустился контейнер. Открываем еще одну вкладку терминала. По команде «s**udo docker ps**» смотрим ID и проверяем, что он работает (если статус – «**up**», это значит, что он работает):

```
asya@asya-HP-Pavilion-dv6-Notebook-PC:~$ sudo docker ps
[sudo] password for asya:
CONTAINER ID   IMAGE                                COMMAND            CREATED         STAT
                                                     NAMES
19f44cf61f73   clickhouse/clickhouse-server:22.9.7   "/entrypoint.sh"   4 minutes ago   Up 4
9009/tcp, 0.0.0.0:9000->9000/tcp, :::9000->9000/tcp   festive_jennings
asya@asya-HP-Pavilion-dv6-Notebook-PC:~$
```

Проверим образ сервера на работоспособность. Для этого мы запустим «Клиент» – «**docker exet -it**». Указываем айдишник контейнера и команду, которую нужно запустить внутри контейнера (сразу напишем «clickhouse-client»:

```
asya@asya-HP-Pavilion-dv6-Notebook-PC:~$ sudo docker ps
[sudo] password for asya:
CONTAINER ID   IMAGE                                COMMAND            CREATED         STAT
                                                     NAMES
19f44cf61f73   clickhouse/clickhouse-server:22.9.7   "/entrypoint.sh"   4 minutes ago   Up 4
9009/tcp, 0.0.0.0:9000->9000/tcp, :::9000->9000/tcp   festive_jennings
asya@asya-HP-Pavilion-dv6-Notebook-PC:~$ sudo docker exec -it 19f44cf61f73 /
```

Мы находимся внутри контейнера после запуска clickhouse-client. Посмотрим, какие здесь есть базы и таблицы:

```
asya@asya-HP-Pavilion-dv6-Notebook-PC:~$ sudo docker ps
[sudo] password for asya:
CONTAINER ID   IMAGE                                COMMAND            CREATED         ST
                                                     NAMES
19f44cf61f73   clickhouse/clickhouse-server:22.9.7   "/entrypoint.sh"   4 minutes ago   Up
9009/tcp, 0.0.0.0:9000->9000/tcp, :::9000->9000/tcp   festive_jennings
asya@asya-HP-Pavilion-dv6-Notebook-PC:~$ sudo docker exec -it 19f44cf61f73 clickhouse-clie
ClickHouse client version 22.9.7.34 (official build).
Connecting to localhost:9000 as user slurm.
Connected to ClickHouse server version 22.9.7 revision 54460.
```

```
Warnings:
 * Maximum number of threads is lower than 30000. There could be problems with handling a lot of simultaneous que
ries.

19f44cf61f73 :) show tables;

SHOW TABLES

Query id: 689712f6-0716-473f-adca-bd6f9209e1b7

Ok.

0 rows in set. Elapsed: 0.004 sec.
```

Таблиц нет, поскольку мы их не создавали. Попробуем создать базу данных и таблицу.

Создаем свою БД «**idplayer**» для того, чтобы работать не с дефолтной БД. Делаем «**USE idplayer**», переключаемся на нее. Таблица создается по команде «**CREATE TABLE**», там задаются все параметры:

```
19f44cf61f73 :) CREATE DATABASE idplayer;

CREATE DATABASE idplayer

Query id: 31352a23-cc1d-4fef-a729-2f05ab860fa7

Ok.

0 rows in set. Elapsed: 0.060 sec.

19f44cf61f73 :) USE idplayer;

USE idplayer

Query id: 6f2fbebb-5738-4f80-bf70-9ed1fd259de1

Ok.

0 rows in set. Elapsed: 0.003 sec.

19f44cf61f73 :)     • CREATE TABLE events (
                  account_id UInt64,
                  event_type String,
                  inserted_at DateTime default now())
                ENGINE MergeTree
                ORDER BY inserted_at;
```

- Название таблицы (events);
- Поля;
- Типы полей;
- Engine MergeTree (это движок таблицы, об этом можно почитать в документации);
- Order by inserted_at (это то, как сортируются данные в таблице физически при хранении). Можно сортировать при извлечении данных по другому ключу, но то поле, которое оказывается в Order by является первичным ключом:

```
19f44cf61f73 :) CREATE TABLE events (account_id UInt64, event_type String, inserted_at DateTime default now()) EN
GINE MergeTree ORDER BY inserted_at;
```

- **Account_id** – это ID игрока;
- **Event_type** – это тип ивента/события, которое произошло, которое совершил игрок;
- **Inserted_at** – это время, когда событие было вставлено. Можно его указывать или же оставлять по умолчанию.

Нюанс: если мы Primary Key указываем отдельно, то это будет другое поле, если мы не указываем Primary Key, то Primary Key – это Order by.

Таблица создана:

```
CREATE TABLE events
(
    `account_id` UInt64,
    `event_type` String,
    `inserted_at` DateTime DEFAULT now()
)
ENGINE = MergeTree
ORDER BY inserted_at

Query id: a81d6e9e-46c4-4590-9922-b4f2a50d8d65

Ok.

0 rows in set. Elapsed: 0.073 sec.

19f44cf61f73 :)
```

Также мы можем указать индекс грануларити (по умолчанию – 8192) – это максимальное количество строковых засечек для индекса, то есть, когда ищется какое-то значение, извлекаются 8192 строчки.

Мы не указали поле, по которому можем указать партиционирование (partition by). Это наборы данных, по которым данные будут объединяться, что тоже помогает оптимизировать работу с данными. Если мы создадим слишком мелкий индекс или слишком много партиций, то будет слишком много маленьких кусочков, и файловые дескрипторы будут перегружены.

По умолчанию партиция всего одна – это таблица целиком. Чтобы посмотреть на данные, вставляем в ивент строчку. С ивентом, как пользователь логинится, указываем дату:

```
CREATE TABLE events
(
    `account_id` UInt64,
    `event_type` String,
    `inserted_at` DateTime DEFAULT now()
)
ENGINE = MergeTree
ORDER BY inserted_at

Query id: a81d6e9e-46c4-4590-9922-b4f2a50d8d65

Ok.

0 rows in set. Elapsed: 0.073 sec.

19f44cf61f73 :) insert into events (*) values(1,'Login', 2022-12-22);

INSERT INTO events (*) FORMAT Values

Query id: 54c866f6-f1d5-492e-bd11-5b29ce896a20

Ok.

1 row in set. Elapsed: 0.026 sec.

19f44cf61f73 :)
```

Второе событие о том, как пользователь делает logout без дат. Пробуем указать другой value и запятую, то есть другой айдишник игрока. Мы вставили события logout. Пользователь логинится, а затем делает logout:

```
Exception on client:
Code: 62. DB::Exception: Cannot parse expression of type String here: 'Logout');: While exec
tFormat: data for INSERT was parsed from query. (SYNTAX_ERROR)

19f44cf61f73 :) insert into events (*) values(2, 'Logout');

INSERT INTO events (*) FORMAT Values

Query id: 79cde745-0164-4c24-b492-493ca1b47426
```

```
Ok.
Exception on client:
Code: 62. DB::Exception: Cannot parse expression of type String here: 'Logout');: While executing ValuesBlockInpu
tFormat: data for INSERT was parsed from query. (SYNTAX_ERROR)

19f44cf61f73 :) insert into events (*) values(2, 'Logout',);

INSERT INTO events (*) FORMAT Values

Query id: b2eb4822-9fdf-479a-84f0-c72fd3e28de0

Ok.

1 row in set. Elapsed: 0.004 sec.

19f44cf61f73 :)
```

Время, когда мы не указываем его, указывается как нулевое:

```
Query id: b2eb4822-9fdf-479a-84f0-c72fd3e28de0

Ok.

1 row in set. Elapsed: 0.004 sec.

19f44cf61f73 :) SELECT * FROM events;

SELECT *
FROM events

Query id: f3649587-d47a-4922-a27c-57356fbe39f5

┌─account_id─┬─event_type─┬─────inserted_at─┐
│          1 │ Login      │ 1970-01-01 00:33:08 │
└────────────┴────────────┴─────────────────┘
┌─account_id─┬─event_type─┬─────inserted_at─┐
│          2 │ Logout     │ 1970-01-01 00:00:00 │
└────────────┴────────────┴─────────────────┘

2 rows in set. Elapsed: 0.005 sec.

19f44cf61f73 :) exit
Bye.
```

Посмотрим, как данные хранятся на диске. Выходим из контейнера и заходим в него, как в файловую систему Linux (через bin/bash). Мы находимся в контейнере; перейдем в `/var/lib/clickhouse/data`. Здесь есть ID-плеер (это директория, где хранится информация по базе данных). Остальные тоже хранят информацию в базах данных:

```
asya@asya-HP-Pavilion-dv6-Notebook-PC:~$ sudo docker ps
CONTAINER ID   IMAGE                                COMMAND            CREATED          STATUS          PORTS
                                                    NAMES
19f44cf61f73   clickhouse/clickhouse-server:22.9.7   "/entrypoint.sh"   13 minutes ago   Up 13 minutes   8123/tcp
, 9009/tcp, 0.0.0.0:9000->9000/tcp, :::9000->9000/tcp   festive_jennings
asya@asya-HP-Pavilion-dv6-Notebook-PC:~$ sudo docker exec -it 19f44cf61f73 /bin/bash
root@19f44cf61f73:/# ls
bin   dev                       entrypoint.sh  home  lib32  libx32  mnt  proc  run   srv  tmp  var
boot  docker-entrypoint-initdb.d  etc            lib   lib64  media   opt  root  sbin  sys  usr
root@19f44cf61f73:/# cd /var/lib/clickhouse/data
root@19f44cf61f73:/var/lib/clickhouse/data# ls
default  idplayer  my_database  system
root@19f44cf61f73:/var/lib/clickhouse/data# cd idplayer
root@19f44cf61f73:/var/lib/clickhouse/data/idplayer# ls
events
root@19f44cf61f73:/var/lib/clickhouse/data/idplayer#
```

Посмотрим, какие есть папки. **Events** – это одноименная таблица. Зайдем и посмотрим:

```
CONTAINER ID   IMAGE                                COMMAND            CREATED          STATUS          PORTS
                                                    NAMES
19f44cf61f73   clickhouse/clickhouse-server:22.9.7   "/entrypoint.sh"   13 minutes ago   Up 13 minutes   8123/tcp
, 9009/tcp, 0.0.0.0:9000->9000/tcp, :::9000->9000/tcp   festive_jennings
asya@asya-HP-Pavilion-dv6-Notebook-PC:~$ sudo docker exec -it 19f44cf61f73 /bin/bash
root@19f44cf61f73:/# ls
bin   dev                       entrypoint.sh  home  lib32  libx32  mnt  proc  run   srv  tmp  var
boot  docker-entrypoint-initdb.d  etc            lib   lib64  media   opt  root  sbin  sys  usr
root@19f44cf61f73:/# cd /var/lib/clickhouse/data
root@19f44cf61f73:/var/lib/clickhouse/data# ls
default  idplayer  my_database  system
root@19f44cf61f73:/var/lib/clickhouse/data# cd idplayer
root@19f44cf61f73:/var/lib/clickhouse/data/idplayer# ls
events
root@19f44cf61f73:/var/lib/clickhouse/data/idplayer# cd events
root@19f44cf61f73:/var/lib/clickhouse/data/idplayer/events# ls
all_1_1_0  all_2_2_0  detached  format_version.txt
root@19f44cf61f73:/var/lib/clickhouse/data/idplayer/events#
```

Здесь хранится информация о вставках. Например, all_1_1_0, all_2_2_2 – это директории, соответствующие вставкам, которые мы проделали. Между ними есть разница в одну минуту (как мы и вставляли):

```
asya@asya-HP-Pavilion-dv6-Notebook-PC:~$ sudo docker ps
CONTAINER ID   IMAGE                                COMMAND            CREATED          STA
                                                    NAMES
19f44cf61f73   clickhouse/clickhouse-server:22.9.7   "/entrypoint.sh"   13 minutes ago   Up
, 9009/tcp, 0.0.0.0:9000->9000/tcp, :::9000->9000/tcp   festive_jennings
asya@asya-HP-Pavilion-dv6-Notebook-PC:~$ sudo docker exec -it 19f44cf61f73 /bin/bash
root@19f44cf61f73:/# ls
bin   dev                       entrypoint.sh  home  lib32  libx32  mnt  proc  run   srv  tmp  var
boot  docker-entrypoint-initdb.d  etc            lib   lib64  media   opt  root  sbin  sys  usr
root@19f44cf61f73:/# cd /var/lib/clickhouse/data
root@19f44cf61f73:/var/lib/clickhouse/data# ls
default  idplayer  my_database  system
root@19f44cf61f73:/var/lib/clickhouse/data# cd idplayer
root@19f44cf61f73:/var/lib/clickhouse/data/idplayer# ls
events
root@19f44cf61f73:/var/lib/clickhouse/data/idplayer# cd events
root@19f44cf61f73:/var/lib/clickhouse/data/idplayer/events# ls
all_1_1_0  all_2_2_0  detached  format_version.txt
root@19f44cf61f73:/var/lib/clickhouse/data/idplayer/events# ls -lh
total 16K
drwxr-x--- 2 clickhouse clickhouse 4.0K May 31 11:16 all_1_1_0
drwxr-x--- 2 clickhouse clickhouse 4.0K May 31 11:17 all_2_2_0
drwxr-x--- 2 clickhouse clickhouse 4.0K May 31 11:14 detached
-rw-r----- 1 clickhouse clickhouse    1 May 31 11:14 format_version.txt
root@19f44cf61f73:/var/lib/clickhouse/data/idplayer/events#
```

Заглянем в директорию и посмотрим развернуто, какие данные есть, и как они хранятся:

```
default  idplayer  my_database  system
root@19f44cf61f73:/var/lib/clickhouse/data# cd idplayer
root@19f44cf61f73:/var/lib/clickhouse/data/idplayer# ls
events
root@19f44cf61f73:/var/lib/clickhouse/data/idplayer# cd events
root@19f44cf61f73:/var/lib/clickhouse/data/idplayer/events# ls
all_1_1_0  all_2_2_0  detached  format_version.txt
root@19f44cf61f73:/var/lib/clickhouse/data/idplayer/events# ls -lh
total 16K
drwxr-x--- 2 clickhouse clickhouse 4.0K May 31 11:16 all_1_1_0
drwxr-x--- 2 clickhouse clickhouse 4.0K May 31 11:17 all_2_2_0
drwxr-x--- 2 clickhouse clickhouse 4.0K May 31 11:14 detached
-rw-r----- 1 clickhouse clickhouse    1 May 31 11:14 format_version.txt
root@19f44cf61f73:/var/lib/clickhouse/data/idplayer/events# cd all_1_1_0
root@19f44cf61f73:/var/lib/clickhouse/data/idplayer/events/all_1_1_0# ls
checksums.txt  columns.txt  count.txt  data.bin  data.mrk3  default_compression_codec.txt  primary.idx
root@19f44cf61f73:/var/lib/clickhouse/data/idplayer/events/all_1_1_0# ls -lh
total 28K
-rw-r----- 1 clickhouse clickhouse 185 May 31 11:16 checksums.txt
-rw-r----- 1 clickhouse clickhouse 100 May 31 11:16 columns.txt
-rw-r----- 1 clickhouse clickhouse   1 May 31 11:16 count.txt
-rw-r----- 1 clickhouse clickhouse  96 May 31 11:16 data.bin
-rw-r----- 1 clickhouse clickhouse 112 May 31 11:16 data.mrk3
-rw-r----- 1 clickhouse clickhouse  10 May 31 11:16 default_compression_codec.txt
-rw-r----- 1 clickhouse clickhouse   8 May 31 11:16 primary.idx
root@19f44cf61f73:/var/lib/clickhouse/data/idplayer/events/all_1_1_0#
```

Здесь хранится вся информация о вставке, которую мы произвели:

Файлы с суффиксом «bin» – это сжатые блоки данных, то есть сами данные, которые мы вставили.

Файлы «mrk3» – это файлы засечек. Они нужны для понимания того, как данные лежат в файле «bin».
Если посмотрим эти данные в бинарном формате, то это нужно сделать по сетам, указанным в файле «data.mrk3».

Primary.idx – это разреженный индекс, где хранятся данные индекса с какой-то периодичностью.
Индекс – это поле «Order by». В этих файлах хранится информация о том, как мы сортируем файлы, как и какие данные вставляем. По Order by данные физически укладываются на диске. Если же указываем Primary Key, то он становится первичным ключом (затем эту физическую сортировку поменять нельзя).

Мы можем периодически производить в фоне процедуру «Optimize» для того, чтобы между собой конкатенировать кусочки во вставки. Периодически они подобным образом мёрчатся на фоне:

```
root@19f44cf61f73:/var/lib/clickhouse/data/idplayer/events# ls -lh
total 16K
drwxr-x--- 2 clickhouse clickhouse 4.0K May 31 11:16 all_1_1_0
drwxr-x--- 2 clickhouse clickhouse 4.0K May 31 11:17 all_2_2_0
drwxr-x--- 2 clickhouse clickhouse 4.0K May 31 11:14 detached
-rw-r----- 1 clickhouse clickhouse    1 May 31 11:14 format_version.txt
root@19f44cf61f73:/var/lib/clickhouse/data/idplayer/events#
```

У нас создается новая директория вместо предыдущих двух. Приблизительно через 8 минут после создания нового директория удаляются старые.

Партиции и part – разные понятия, где:

- **Партиция** – это разделение по признаку;
- **Part** – это кусок вставленных данных.

Вопрос: почему данные хранятся таким образом?

Ответ: это следствие движка MergeTree. Если вы прочитаете по LSM алгоритм сортировки и о MergeTree, то увидите множество сходств. Единственное отличие заключается в том, что MergeTree сразу сохраняет все на диск, а не хранит в оперативной памяти, также в MergeTree отсутствует журнал.

Мы можем выйти из контейнера:

```
drwxr-x--- 2 clickhouse clickhouse 4.0K May 31 11:14 detached
-rw-r----- 1 clickhouse clickhouse    1 May 31 11:14 format_version.txt
root@19f44cf61f73:/var/lib/clickhouse/data/idplayer/events# cd all_1_1_0
root@19f44cf61f73:/var/lib/clickhouse/data/idplayer/events/all_1_1_0# ls
checksums.txt  columns.txt  count.txt  data.bin  data.mrk3  default_compression_codec.tx
root@19f44cf61f73:/var/lib/clickhouse/data/idplayer/events/all_1_1_0# ls -lh
total 28K
-rw-r----- 1 clickhouse clickhouse 185 May 31 11:16 checksums.txt
-rw-r----- 1 clickhouse clickhouse 100 May 31 11:16 columns.txt
-rw-r----- 1 clickhouse clickhouse   1 May 31 11:16 count.txt
-rw-r----- 1 clickhouse clickhouse  96 May 31 11:16 data.bin
-rw-r----- 1 clickhouse clickhouse 112 May 31 11:16 data.mrk3
-rw-r----- 1 clickhouse clickhouse  10 May 31 11:16 default_compression_codec.txt
-rw-r----- 1 clickhouse clickhouse   8 May 31 11:16 primary.idx
root@19f44cf61f73:/var/lib/clickhouse/data/idplayer/events/all_1_1_0# cd ..
root@19f44cf61f73:/var/lib/clickhouse/data/idplayer/events# ls -lh
total 16K
drwxr-x--- 2 clickhouse clickhouse 4.0K May 31 11:16 all_1_1_0
drwxr-x--- 2 clickhouse clickhouse 4.0K May 31 11:17 all_2_2_0
drwxr-x--- 2 clickhouse clickhouse 4.0K May 31 11:14 detached
-rw-r----- 1 clickhouse clickhouse    1 May 31 11:14 format_version.txt
root@19f44cf61f73:/var/lib/clickhouse/data/idplayer/events# exit
exit
asya@asya-HP-Pavilion-dv6-Notebook-PC:~$ sudo docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
asya@asya-HP-Pavilion-dv6-Notebook-PC:~$
```

Если контейнер запущен в фоне, можно сделать «Container ID Stop».

Как вам урок?

Изучил, далее