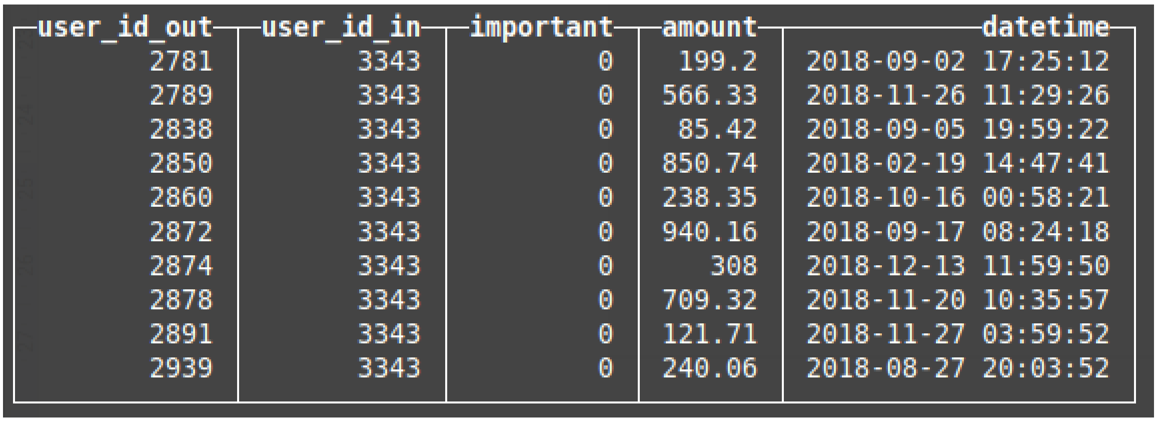
ClickHouse Lab

During this lab you have to implement Data Warehouse (DWH) using ClickHouse (CH) and its techniques such as Materialized View (MV) and Distributed tables.

Dataset

Dataset is presented by a parquet file with users' transactions. Path to this file: `/nfs/shared/clickhouse_data/transactions_12M.parquet`

Dataset sample:



```
┌─user_id_out─┬─user_id_in─┬─important─┬─amount─┬────────────datetime─┐
│        2781 │       3343 │         0 │  199.2 │ 2018-09-02 17:25:12 │
│        2789 │       3343 │         0 │ 566.33 │ 2018-11-26 11:29:26 │
│        2838 │       3343 │         0 │  85.42 │ 2018-09-05 19:59:22 │
│        2850 │       3343 │         0 │ 850.74 │ 2018-02-19 14:47:41 │
│        2860 │       3343 │         0 │ 238.35 │ 2018-10-16 00:58:21 │
│        2872 │       3343 │         0 │ 940.16 │ 2018-09-17 08:24:18 │
│        2874 │       3343 │         0 │    308 │ 2018-12-13 11:59:50 │
│        2878 │       3343 │         0 │ 709.32 │ 2018-11-20 10:35:57 │
│        2891 │       3343 │         0 │ 121.71 │ 2018-11-27 03:59:52 │
│        2939 │       3343 │         0 │ 240.06 │ 2018-08-27 20:03:52 │
│             │            │           │        │                     │
└─────────────┴────────────┴───────────┴────────┴─────────────────────┘
```

Dataset properties:

- ~20% transactions are important (important == 1)
- Total records amount - 12 millions

Task

1. You have to choose **2 or more** MVs. The MVs list is located below.

    a. **Average amount for incoming and outcoming transactions by months and days for each user.**

    b. **The number of important transactions for incoming and outcoming transactions by months and days for each user.**

2. Upload the data into the CH cluster. Table for uploaded data has to be the MergeTree family. To distribute data over the cluster you have to use the Distributed engine and sharding expression.

```
CREATE TABLE prodriges_374222.user_transac
ON CLUSTER kube_clickhouse_cluster
(
    user_id_out Int64,
    user_id_in Int64,
    important Int64,
    amount Float64,
    datetime DateTime
)
ENGINE = MergeTree()
```

```
PARTITION BY toYYYYMM(datetime)
ORDER BY (user_id_out, user_id_in, amount);
```

Justification:

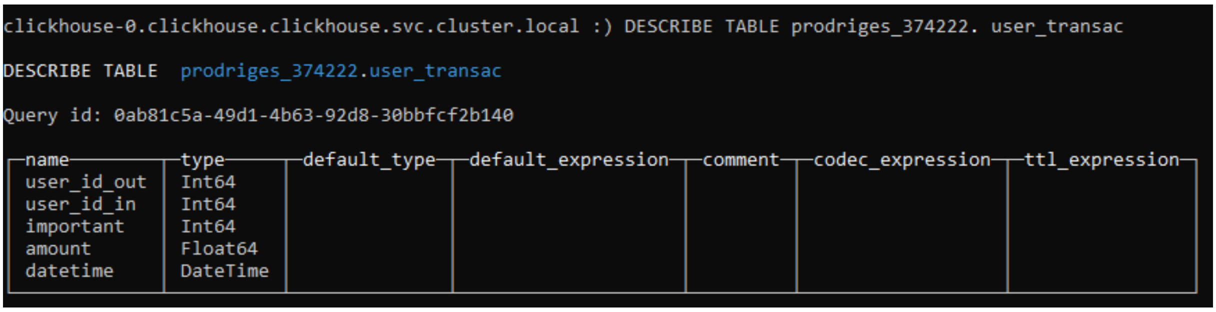
- **Partitioned by toYYYYMM(datetime):**
  - **Reason:** This divides the data into monthly partitions, optimizing time-based queries and making it easier to manage large datasets.
- **Ordered by (user_id_out, user_id_in, amount):**
  - **Reason:** Optimizes queries that filter, or aggregate data based on user IDs and transaction amounts.

```
CREATE TABLE prodriges_374222.distributed_user_transac
ON CLUSTER kube_clickhouse_cluster AS prodriges_374222.user_transac
ENGINE = Distributed(
        kube_clickhouse_cluster,
        prodriges_374222,
        user_transac,
        xxHash64(datetime)
);
```

Justification:

- **Sharding by xxHash64(datetime):**
  - **Explanation:** Sharding by xxHash64(datetime) distributes data evenly across the cluster based on the hash of the datetime value. This ensures a balanced distribution of data, avoiding hot spots and improving query performance.

```
DESCRIBE TABLE prodriges_374222. user_transac
```



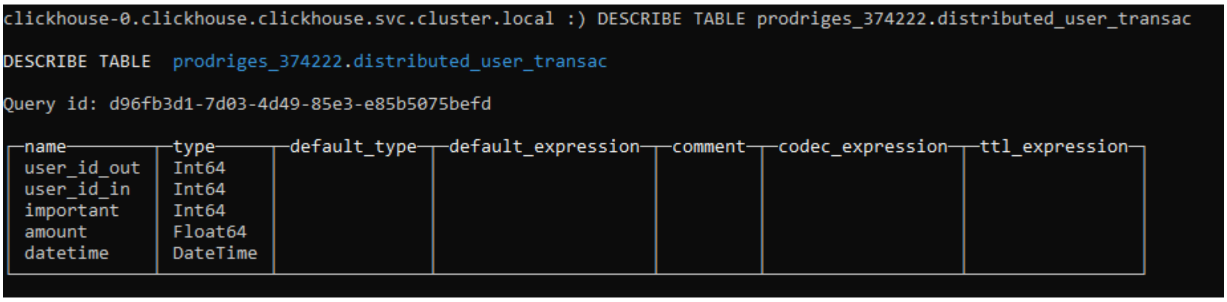
```
clickhouse-0.clickhouse.clickhouse.svc.cluster.local :) DESCRIBE TABLE prodriges_374222. user_transac

DESCRIBE TABLE  prodriges_374222.user_transac

Query id: 0ab81c5a-49d1-4b63-92d8-30bbfcf2b140

┌─name────────┬─type─────┬─default_type─┬─default_expression─┬─comment─┬─codec_expression─┬─ttl_expression─┐
│ user_id_out │ Int64    │              │                    │         │                  │                │
│ user_id_in  │ Int64    │              │                    │         │                  │                │
│ important   │ Int64    │              │                    │         │                  │                │
│ amount      │ Float64  │              │                    │         │                  │                │
│ datetime    │ DateTime │              │                    │         │                  │                │
└─────────────┴──────────┴──────────────┴────────────────────┴─────────┴──────────────────┴────────────────┘
```

```
cat shared-data/clickhouse_data/transactions_12M.parquet | \
clickhouse-client --host=clickhouse-0.clickhouse.clickhouse \
                  --user=prodriges_374222\
                  --password=479Ak98oRi \
                  --query="INSERT INTO
prodriges_374222.distributed_user_transac FORMAT Parquet"
```

Justification:

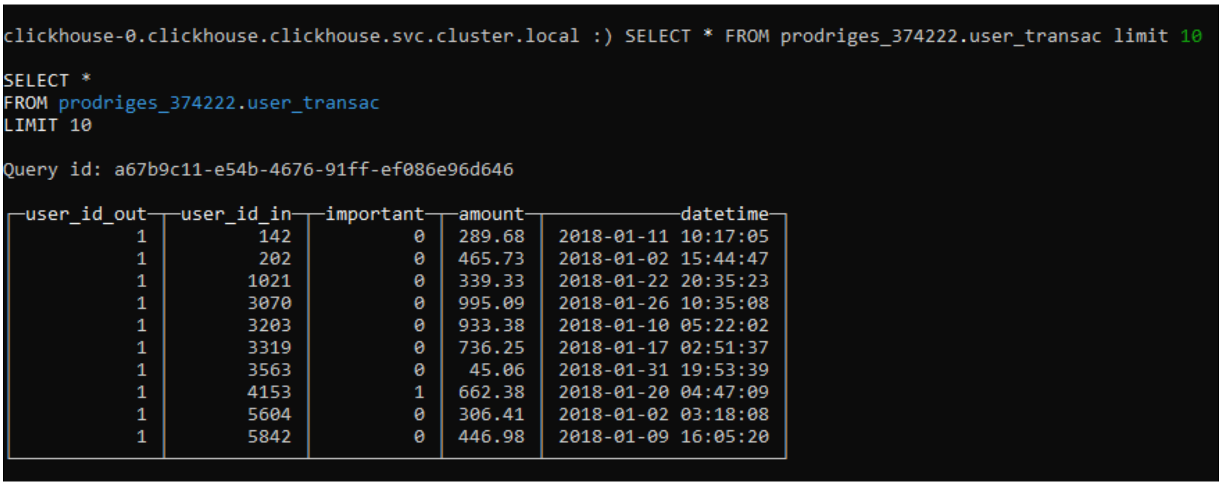
- • **cat shared-data/clickhouse_data/transactions_12M.parquet:**
  - **Reason:** This command reads the file transactions_12M.parquet which contains the data to be inserted into the distributed_user_transac table.
- **clickhouse-client --host=clickhouse-0.clickhouse.clickhouse --user=prodriges_374222 --password=479Ak98oRi --query="INSERT INTO prodriges_374222.distributed_user_transac FORMAT Parquet":**
  - **Reason:** Utilizes the ClickHouse client to insert data into the distributed_user_transac table in the specified cluster.

```
DESCRIBE TABLE prodriges_374222.distributed_user_transac
```



```
clickhouse-0.clickhouse.clickhouse.svc.cluster.local :) DESCRIBE TABLE prodriges_374222.distributed_user_transac

DESCRIBE TABLE  prodriges_374222.distributed_user_transac

Query id: d96fb3d1-7d03-4d49-85e3-e85b5075befd

┌─name────────┬─type─────┬─default_type─┬─default_expression─┬─comment─┬─codec_expression─┬─ttl_expression─┐
│ user_id_out │ Int64    │              │                    │         │                  │                │
│ user_id_in  │ Int64    │              │                    │         │                  │                │
│ important   │ Int64    │              │                    │         │                  │                │
│ amount      │ Float64  │              │                    │         │                  │                │
│ datetime    │ DateTime │              │                    │         │                  │                │
└─────────────┴──────────┴──────────────┴────────────────────┴─────────┴──────────────────┴────────────────┘
```

```
SELECT * FROM prodriges_374222.user_transac limit 10
```



```
clickhouse-0.clickhouse.clickhouse.svc.cluster.local :) SELECT * FROM prodriges_374222.user_transac limit 10

SELECT *
FROM prodriges_374222.user_transac
LIMIT 10

Query id: a67b9c11-e54b-4676-91ff-ef086e96d646

┌─user_id_out─┬─user_id_in─┬─important─┬─amount─┬────────────datetime─┐
│           1 │        142 │         0 │ 289.68 │ 2018-01-11 10:17:05 │
│           1 │        202 │         0 │ 465.73 │ 2018-01-02 15:44:47 │
│           1 │       1021 │         0 │ 339.33 │ 2018-01-22 20:35:23 │
│           1 │       3070 │         0 │ 995.09 │ 2018-01-26 10:35:08 │
│           1 │       3203 │         0 │ 933.38 │ 2018-01-10 05:22:02 │
│           1 │       3319 │         0 │ 736.25 │ 2018-01-17 02:51:37 │
│           1 │       3563 │         0 │  45.06 │ 2018-01-31 19:53:39 │
│           1 │       4153 │         1 │ 662.38 │ 2018-01-20 04:47:09 │
│           1 │       5604 │         0 │ 306.41 │ 2018-01-02 03:18:08 │
│           1 │       5842 │         0 │ 446.98 │ 2018-01-09 16:05:20 │
└─────────────┴────────────┴───────────┴────────┴─────────────────────┘
```

3. Implement the chosen MVs. Also you are able to create extra tables with different engines if you need them. The number of extra tables should be reasonable.
   a. **Average amount for incoming and outcoming transactions by months and days for each user.**

      i. **Average by Month**

```
CREATE MATERIALIZED VIEW prodriges_374222.month_average_amount
ON CLUSTER kube_clickhouse_cluster
ENGINE = AggregatingMergeTree
ORDER BY (user_id, date)
POPULATE AS
WITH
    suma_dentro AS (
        SELECT
            user_id_in AS user_id,
            formatDateTime(datetime, '%Y-%m') AS date,
            ROUND(AVG(amount), 2) AS avg_in
        FROM prodriges_374222.distributed_user_transac
        WHERE user_id_in IS NOT NULL
        GROUP BY user_id, date
    ),
    suma_fuera AS (
        SELECT
            user_id_out AS user_id,
            formatDateTime(datetime, '%Y-%m') AS date,
            ROUND(AVG(amount), 2) AS avg_out
        FROM prodriges_374222.distributed_user_transac
        WHERE user_id_out IS NOT NULL
        GROUP BY user_id, date
    )
SELECT
    suma_dentro.user_id AS user_id,
    suma_dentro.date,
    suma_dentro.avg_in,
    suma_fuera.avg_out
FROM suma_dentro
INNER JOIN suma_fuera
    ON suma_dentro.user_id = suma_fuera.user_id
    AND suma_dentro.date = suma_fuera.date
ORDER BY suma_dentro.user_id,suma_dentro.date;
```

Justification:

- **Materialized View Creation:**
  - **Reason:** The materialized view month_average_amount is created to pre-aggregate the average transaction amounts for each user on a monthly basis, improving query performance by storing precomputed results.
- **AggregatingMergeTree Engine:**
  - **Reason:** The AggregatingMergeTree engine is used to efficiently store and manage aggregate data. This engine is suitable for scenarios where data needs to be aggregated and queried efficiently.

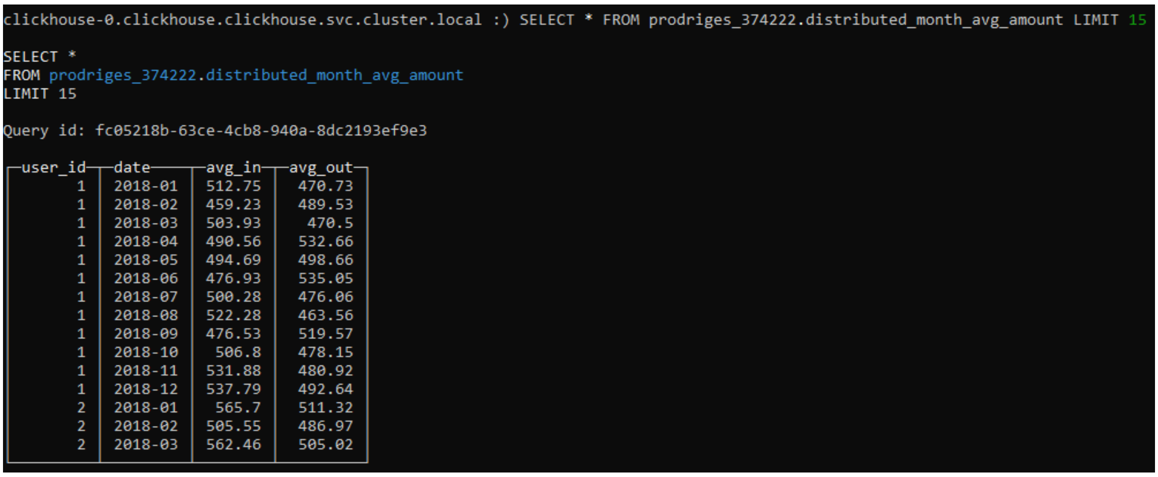
- **ORDER BY (user_id, date):**
  - **Reason:** Ordering by user_id and date ensures that the data is organized in a way that optimizes query performance, particularly for queries that filter or aggregate by user and date.
- **Subqueries suma_dentro and suma_fuera:**
  - **Reason:** These subqueries compute the monthly average amounts for incoming (user_id_in) and outgoing (user_id_out) transactions, respectively. They group the data by user_id and month (date), rounding the average amount to two decimal places for precision.
- **Join Operation:**
  - **Reason:** The join operation combines the results of the suma_dentro and suma_fuera subqueries, matching records by user_id and date. This allows the final selection to include both incoming and outgoing average amounts for each user and month.

```
CREATE TABLE prodriges_374222.distributed_month_avg_amount
ON CLUSTER kube_clickhouse_cluster AS
prodriges_374222.month_average_amount
ENGINE = Distributed(
    kube_clickhouse_cluster,
    prodriges_374222,
    month_average_amount
);
```

Justification:

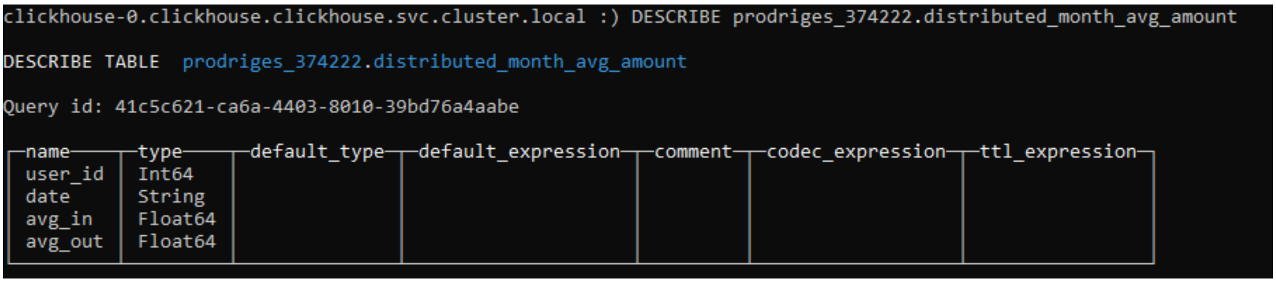
1. **Creation of Distributed Table:**
   - **Reason**: The distributed_month_avg_amount table is created to distribute the data from the materialized view month_average_amount across the cluster, enabling more efficient and scalable queries.
2. **Distributed Engine:**
   - **Reason**: The Distributed engine is used to distribute the data among the nodes of the cluster. This allows queries to be executed in parallel across multiple nodes, improving performance and the ability to handle large volumes of data.
3. **Specification of Cluster and Source Table:**
   - **Reason**: The kube_clickhouse_cluster option specifies the cluster where the data will be distributed. The prodriges_374222 and month_average_amount options specify the database and the source table (materialized view) from which the data will be taken.

```
SELECT * FROM prodriges_374222.distributed_month_avg_amount LIMIT
15
```



```
clickhouse-0.clickhouse.clickhouse.svc.cluster.local :) SELECT * FROM prodriges_374222.distributed_month_avg_amount LIMIT 15

SELECT *
FROM prodriges_374222.distributed_month_avg_amount
LIMIT 15

Query id: fc05218b-63ce-4cb8-940a-8dc2193ef9e3

┌─user_id─┬─date────┬─avg_in─┬─avg_out─┐
│       1 │ 2018-01 │ 512.75 │  470.73 │
│       1 │ 2018-02 │ 459.23 │  489.53 │
│       1 │ 2018-03 │ 503.93 │   470.5 │
│       1 │ 2018-04 │ 490.56 │  532.66 │
│       1 │ 2018-05 │ 494.69 │  498.66 │
│       1 │ 2018-06 │ 476.93 │  535.05 │
│       1 │ 2018-07 │ 500.28 │  476.06 │
│       1 │ 2018-08 │ 522.28 │  463.56 │
│       1 │ 2018-09 │ 476.53 │  519.57 │
│       1 │ 2018-10 │  506.8 │  478.15 │
│       1 │ 2018-11 │ 531.88 │  480.92 │
│       1 │ 2018-12 │ 537.79 │  492.64 │
│       2 │ 2018-01 │  565.7 │  511.32 │
│       2 │ 2018-02 │ 505.55 │  486.97 │
│       2 │ 2018-03 │ 562.46 │  505.02 │
└─────────┴─────────┴────────┴─────────┘
```

```
DESCRIBE prodriges_374222.distributed_month_avg_amount
```



```
clickhouse-0.clickhouse.clickhouse.svc.cluster.local :) DESCRIBE prodriges_374222.distributed_month_avg_amount

DESCRIBE TABLE prodriges_374222.distributed_month_avg_amount

Query id: 41c5c621-ca6a-4403-8010-39bd76a4aabe

┌─name────┬─type────┬─default_type─┬─default_expression─┬─comment─┬─codec_expression─┬─ttl_expression─┐
│ user_id │ Int64   │              │                    │         │                  │                │
│ date    │ String  │              │                    │         │                  │                │
│ avg_in  │ Float64 │              │                    │         │                  │                │
│ avg_out │ Float64 │              │                    │         │                  │                │
└─────────┴─────────┴──────────────┴────────────────────┴─────────┴──────────────────┴────────────────┘
```

i. Average by Day

```
CREATE MATERIALIZED VIEW prodriges_374222.day_average_amount
ON CLUSTER kube_clickhouse_cluster
ENGINE = AggregatingMergeTree
ORDER BY (user_id, date)
POPULATE AS
WITH
    suma_dentro AS (
        SELECT
            user_id_in AS user_id,
            formatDateTime(datetime, '%d-%m-%G') AS date,
            ROUND(AVG(amount), 2) AS avg_in
        FROM prodriges_374222.distributed_user_transac
        WHERE user_id_in IS NOT NULL
        GROUP BY user_id, date
    ),
    suma_fuera AS (
        SELECT
            user_id_out AS user_id,
            formatDateTime(datetime, '%d-%m-%G') AS date,
            ROUND(AVG(amount), 2) AS avg_out
        FROM prodriges_374222.distributed_user_transac
        WHERE user_id_out IS NOT NULL
        GROUP BY user_id, date
    )
SELECT
```

```
    suma_dentro.user_id AS user_id,
    suma_dentro.date,
    suma_dentro.avg_in,
    suma_fuera.avg_out
FROM suma_dentro
INNER JOIN suma_fuera
    ON suma_dentro.user_id = suma_fuera.user_id
    AND suma_dentro.date = suma_fuera.date
ORDER BY suma_dentro.user_id, suma_dentro.date;
```

Justification:

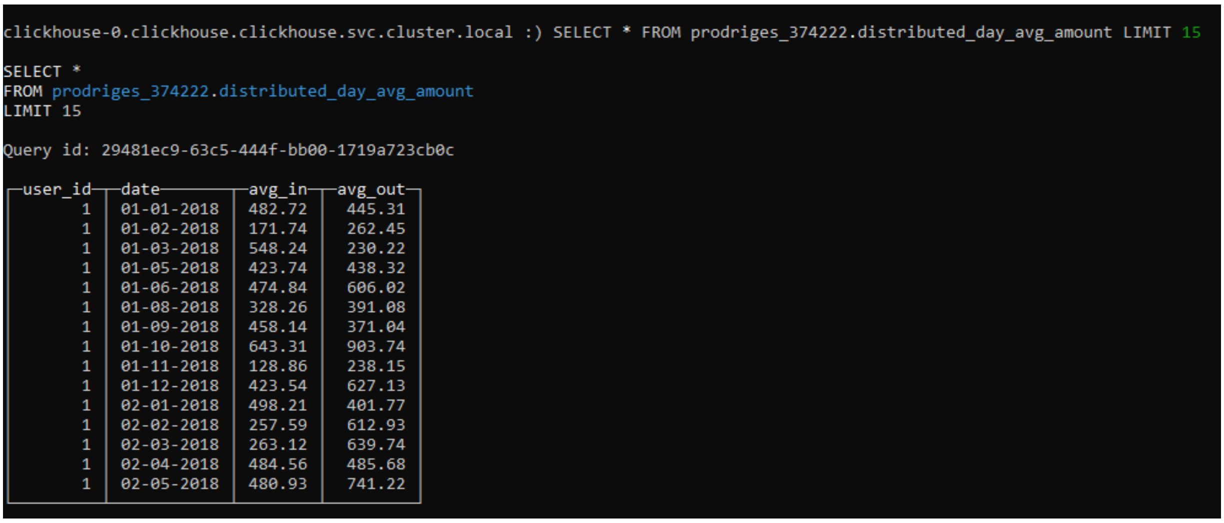
1. **Creation of Materialized View**:
   - **Reason**: The materialized view day_average_amount is created to pre-aggregate the average transaction amounts for each user on a daily basis, improving query performance by storing precomputed results.
2. **AggregatingMergeTree Engine**:
   - **Reason**: The AggregatingMergeTree engine is used to efficiently store and manage aggregate data. This engine is suitable for scenarios where data needs to be aggregated and queried efficiently.
3. **ORDER BY (user_id, date)**:
   - **Reason**: Ordering by user_id and date ensures that the data is organized in a way that optimizes query performance, particularly for queries that filter or aggregate by user and date.
4. **Subqueries suma_dentro and suma_fuera**:
   - **Reason**: These subqueries compute the daily average amounts for incoming (user_id_in) and outgoing (user_id_out) transactions, respectively. They group the data by user_id and day (date), rounding the average amount to two decimal places for precision.
5. **Join Operation**:
   - **Reason**: The join operation combines the results of the suma_dentro and suma_fuera subqueries, matching records by user_id and date. This allows the final selection to include both incoming and outgoing average amounts for each user and day.

```
CREATE TABLE prodriges_374222.distributed_day_avg_amount
ON CLUSTER kube_clickhouse_cluster AS
prodriges_374222.day_average_amount
ENGINE = Distributed (
    kube_clickhouse_cluster,
    prodriges_374222,
    day_average_amount
);
```

Justification:

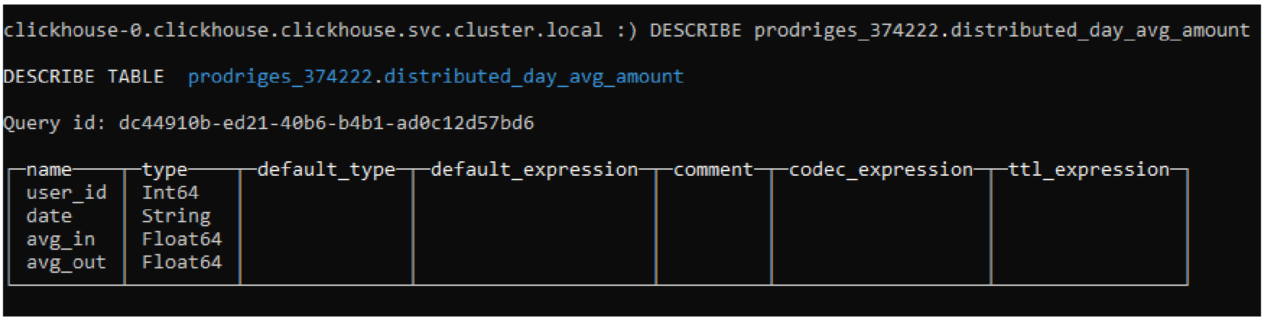
1. **Creation of Distributed Table**:
   - **Reason**: The distributed_day_avg_amount table is created to distribute the data from the materialized view day_average_amount across the cluster, enabling more efficient and scalable queries.
2. **Distributed Engine**:
   - **Reason**: The Distributed engine is used to distribute the data among the nodes of the cluster. This allows queries to be executed in parallel across multiple nodes, improving performance and the ability to handle large volumes of data.
3. **Specification of Cluster and Source Table**:
   - **Reason**: The kube_clickhouse_cluster option specifies the cluster where the data will be distributed. The prodriges_374222 and day_average_amount options specify the database and the source table (materialized view) from which the data will be taken.

```
SELECT * FROM prodriges_374222.distributed_day_avg_amount LIMIT 15
```



```
clickhouse-0.clickhouse.clickhouse.svc.cluster.local :) SELECT * FROM prodriges_374222.distributed_day_avg_amount LIMIT 15

SELECT *
FROM prodriges_374222.distributed_day_avg_amount
LIMIT 15

Query id: 29481ec9-63c5-444f-bb00-1719a723cb0c

┌─user_id─┬─date───────┬─avg_in─┬─avg_out─┐
│       1 │ 01-01-2018 │ 482.72 │  445.31 │
│       1 │ 01-02-2018 │ 171.74 │  262.45 │
│       1 │ 01-03-2018 │ 548.24 │  230.22 │
│       1 │ 01-05-2018 │ 423.74 │  438.32 │
│       1 │ 01-06-2018 │ 474.84 │  606.02 │
│       1 │ 01-08-2018 │ 328.26 │  391.08 │
│       1 │ 01-09-2018 │ 458.14 │  371.04 │
│       1 │ 01-10-2018 │ 643.31 │  903.74 │
│       1 │ 01-11-2018 │ 128.86 │  238.15 │
│       1 │ 01-12-2018 │ 423.54 │  627.13 │
│       1 │ 02-01-2018 │ 498.21 │  401.77 │
│       1 │ 02-02-2018 │ 257.59 │  612.93 │
│       1 │ 02-03-2018 │ 263.12 │  639.74 │
│       1 │ 02-04-2018 │ 484.56 │  485.68 │
│       1 │ 02-05-2018 │ 480.93 │  741.22 │
└─────────┴────────────┴────────┴─────────┘
```

```
DESCRIBE prodriges_374222.distributed_day_avg_amount
```



```
clickhouse-0.clickhouse.clickhouse.svc.cluster.local :) DESCRIBE prodriges_374222.distributed_day_avg_amount

DESCRIBE TABLE prodriges_374222.distributed_day_avg_amount

Query id: dc44910b-ed21-40b6-b4b1-ad0c12d57bd6

┌─name─────┬─type────┬─default_type─┬─default_expression─┬─comment─┬─codec_expression─┬─ttl_expression─┐
│ user_id  │ Int64   │              │                    │         │                  │                │
│ date     │ String  │              │                    │         │                  │                │
│ avg_in   │ Float64 │              │                    │         │                  │                │
│ avg_out  │ Float64 │              │                    │         │                  │                │
└──────────┴─────────┴──────────────┴────────────────────┴─────────┴──────────────────┴────────────────┘
```

b. **The number of important transactions for incoming and outcoming transactions by months and days for each user.**

   i. **Important transactions by months**

```
CREATE MATERIALIZED VIEW prodriges_374222.month_important_number
ON CLUSTER kube_clickhouse_cluster
ENGINE = AggregatingMergeTree
ORDER BY (user_id, date)
POPULATE AS
WITH
    suma_dentro AS (
        SELECT
            user_id_in AS user_id,
            formatDateTime(datetime, '%m-%G') AS date,
            COUNT(amount) AS count_in
        FROM prodriges_374222.distributed_user_transac
        WHERE important = 1 AND user_id_in IS NOT NULL
        GROUP BY user_id, date
    ),
    suma_fuera AS (
        SELECT
            user_id_out AS user_id,
            formatDateTime(datetime, '%m-%G') AS date,
            COUNT(amount) AS count_out
        FROM prodriges_374222.distributed_user_transac
        WHERE important = 1 AND user_id_out IS NOT NULL
        GROUP BY user_id, date
    )
SELECT
    suma_dentro.user_id AS user_id,
    suma_dentro.date AS date,
    suma_dentro.count_in,
    suma_fuera.count_out
FROM suma_dentro
INNER JOIN suma_fuera
    ON suma_dentro.user_id = suma_fuera.user_id
    AND suma_dentro.date = suma_fuera.date
ORDER BY suma_dentro.user_id, suma_dentro.date;
```

Justification:

1. **Creation of Materialized View**:
   - **Reason**: The materialized view month_important_number is created to pre-aggregate the count of important transactions for each user on a monthly basis, improving query performance by storing precomputed results.
2. **AggregatingMergeTree Engine**:
   - **Reason**: The AggregatingMergeTree engine is used to efficiently store and manage aggregate data. This engine is suitable for scenarios where data needs to be aggregated and queried efficiently.
3. **ORDER BY (user_id, date)**:

   - **Reason**: Ordering by user_id and date ensures that the data is organized in a way that optimizes query performance, particularly for queries that filter or aggregate by user and date.

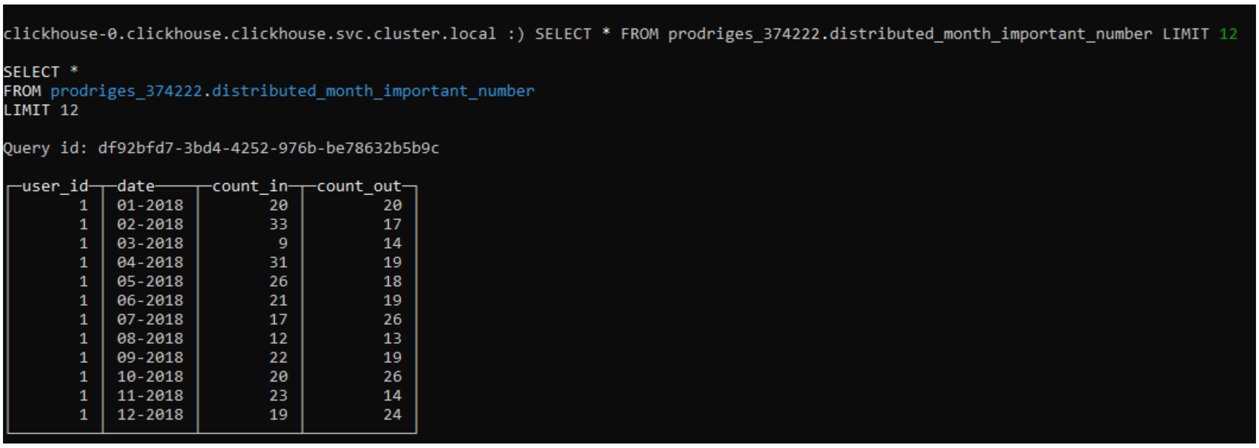
4. **Subqueries suma_dentro and suma_fuera**:
   - **Reason**: These subqueries compute the monthly count of important incoming (user_id_in) and outgoing (user_id_out) transactions, respectively. They group the data by user_id and month (date).
5. **Join Operation**:
   - **Reason**: The join operation combines the results of the suma_dentro and suma_fuera subqueries, matching records by user_id and date. This allows the final selection to include both incoming and outgoing counts of important transactions for each user and month.

```
CREATE TABLE prodriges_374222.distributed_month_important_number
ON CLUSTER kube_clickhouse_cluster AS
prodriges_374222.month_important_number
ENGINE = Distributed(
    kube_clickhouse_cluster,
    prodriges_374222,
    month_important_number
);
```

Justification:

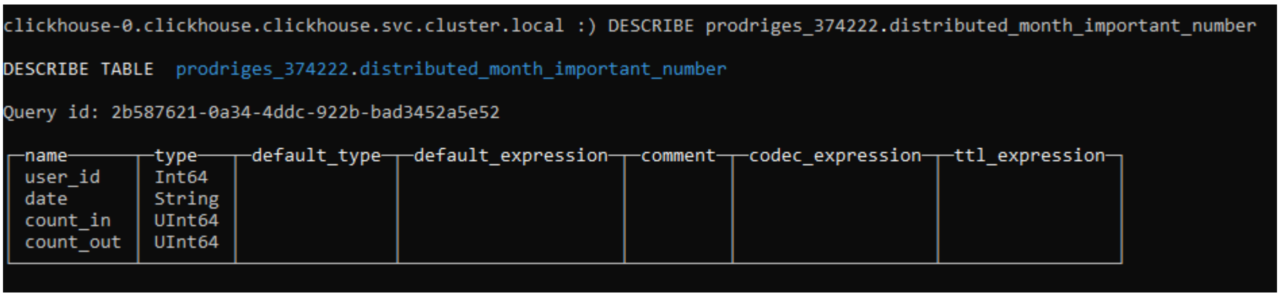
1. **Creation of Distributed Table**:
   - **Reason**: The distributed_month_important_number table is created to distribute the data from the materialized view month_important_number across the cluster, enabling more efficient and scalable queries.
2. **Distributed Engine**:
   - **Reason**: The Distributed engine is used to distribute the data among the nodes of the cluster. This allows queries to be executed in parallel across multiple nodes, improving performance and the ability to handle large volumes of data.
3. **Specification of Cluster and Source Table**:
   - **Reason**:The kube_clickhouse_cluster option specifies the cluster where the data will be distributed. The prodriges_374222 and month_important_number options specify the database and the source table (materialized view) from which the data will be taken.

```
SELECT * FROM prodriges_374222.distributed_month_important_number
LIMIT 12
```



```
clickhouse-0.clickhouse.clickhouse.svc.cluster.local :) SELECT * FROM prodriges_374222.distributed_month_important_number LIMIT 12

SELECT *
FROM prodriges_374222.distributed_month_important_number
LIMIT 12

Query id: df92bfd7-3bd4-4252-976b-be78632b5b9c

┌─user_id─┬─date────┬─count_in─┬─count_out─┐
│       1 │ 01-2018 │       20 │        20 │
│       1 │ 02-2018 │       33 │        17 │
│       1 │ 03-2018 │        9 │        14 │
│       1 │ 04-2018 │       31 │        19 │
│       1 │ 05-2018 │       26 │        18 │
│       1 │ 06-2018 │       21 │        19 │
│       1 │ 07-2018 │       17 │        26 │
│       1 │ 08-2018 │       12 │        13 │
│       1 │ 09-2018 │       22 │        19 │
│       1 │ 10-2018 │       20 │        26 │
│       1 │ 11-2018 │       23 │        14 │
│       1 │ 12-2018 │       19 │        24 │
└─────────┴─────────┴──────────┴───────────┘
```

```
DESCRIBE prodriges_374222.distributed_month_important_number
```



```
clickhouse-0.clickhouse.clickhouse.svc.cluster.local :) DESCRIBE prodriges_374222.distributed_month_important_number

DESCRIBE TABLE  prodriges_374222.distributed_month_important_number

Query id: 2b587621-0a34-4ddc-922b-bad3452a5e52

┌─name──────┬─type───┬─default_type─┬─default_expression─┬─comment─┬─codec_expression─┬─ttl_expression─┐
│ user_id   │ Int64  │              │                    │         │                  │                │
│ date      │ String │              │                    │         │                  │                │
│ count_in  │ UInt64 │              │                    │         │                  │                │
│ count_out │ UInt64 │              │                    │         │                  │                │
└───────────┴────────┴──────────────┴────────────────────┴─────────┴──────────────────┴────────────────┘
```

ii. Important transactions by day

```
CREATE MATERIALIZED VIEW prodriges_374222.day_important_number
ON CLUSTER kube_clickhouse_cluster
ENGINE = AggregatingMergeTree
ORDER BY (user_id, date)
POPULATE AS
WITH
    suma_dentro AS (
        SELECT
            user_id_in AS user_id,
            formatDateTime(datetime, '%d-%m-%G') AS date,
            COUNT(amount) AS count_in
        FROM prodriges_374222.distributed_user_transac
        WHERE important = 1 AND user_id_in IS NOT NULL
        GROUP BY user_id, date
    ),
    suma_fuera AS (
        SELECT
            user_id_out AS user_id,
            formatDateTime(datetime, '%d-%m-%G') AS date,
            COUNT(amount) AS count_out
        FROM prodriges_374222.distributed_user_transac
        WHERE important = 1 AND user_id_out IS NOT NULL
        GROUP BY user_id, date
    )
```

```
SELECT
    suma_dentro.user_id AS user_id,
    suma_dentro.date AS date,
    suma_dentro.count_in,
    suma_fuera.count_out
FROM suma_dentro
INNER JOIN suma_fuera
    ON suma_dentro.user_id = suma_fuera.user_id
    AND suma_dentro.date = suma_fuera.date
ORDER BY suma_dentro.user_id, suma_dentro.date;
```

Justification:

- **Creation of Materialized View:**
  - **Reason:** The materialized view day_important_number is created to pre-aggregate the count of important transactions for each user on a daily basis, improving query performance by storing precomputed results.
- **AggregatingMergeTree Engine:**
  - **Reason:** The AggregatingMergeTree engine is used to efficiently store and manage aggregate data. This engine is suitable for scenarios where data needs to be aggregated and queried efficiently.
- **ORDER BY (user_id, date):**
  - **Reason:** Ordering by user_id and date ensures that the data is organized in a way that optimizes query performance, particularly for queries that filter or aggregate by user and date.
- **Subqueries suma_dentro and suma_fuera:**
  - **Reason:** These subqueries compute the daily count of important incoming (user_id_in) and outgoing (user_id_out) transactions, respectively. They group the data by user_id and day (date).
- **Join Operation:**
  - **Reason:** The join operation combines the results of the suma_dentro and suma_fuera subqueries, matching records by user_id and date. This allows the final selection to include both incoming and outgoing counts of important transactions for each user and day.

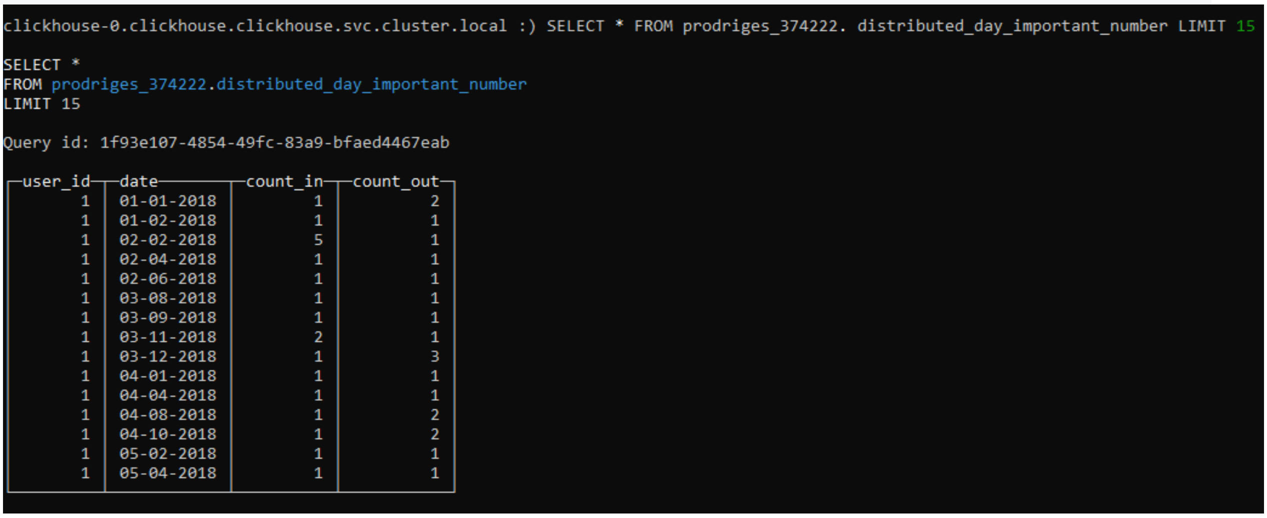
```
CREATE TABLE prodriges_374222.distributed_day_important_number
ON CLUSTER kube_clickhouse_cluster AS prodriges_374222.day_important_number
ENGINE = Distributed(
    kube_clickhouse_cluster,
    prodriges_374222,
    day_important_number
);
```

Justification:

1. **Creation of Distributed Table:**
   - **Reason:** The distributed_day_important_number table is created to distribute the data from the materialized view day_important_number across the cluster, enabling more efficient and scalable queries.

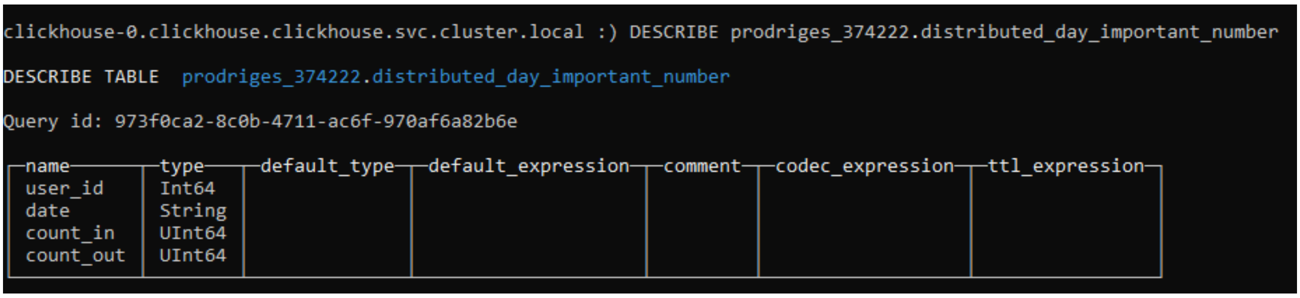
2. **Distributed Engine**:
   - **Reason**: The Distributed engine is used to distribute the data among the nodes of the cluster. This allows queries to be executed in parallel across multiple nodes, improving performance and the ability to handle large volumes of data.
3. **Specification of Cluster and Source Table**:
   - **Reason**: The kube_clickhouse_cluster option specifies the cluster where the data will be distributed.The prodriges_374222 and day_important_number options specify the database and the source table (materialized view) from which the data will be taken.

```
SELECT * FROM prodriges_374222.distributed_day_important_number LIMIT 15
```



```
clickhouse-0.clickhouse.clickhouse.svc.cluster.local :) SELECT * FROM prodriges_374222. distributed_day_important_number LIMIT 15

SELECT *
FROM prodriges_374222.distributed_day_important_number
LIMIT 15

Query id: 1f93e107-4854-49fc-83a9-bfaed4467eab

┌─user_id─┬─date───────┬─count_in─┬─count_out─┐
│       1 │ 01-01-2018 │        1 │         2 │
│       1 │ 01-02-2018 │        1 │         1 │
│       1 │ 02-02-2018 │        5 │         1 │
│       1 │ 02-04-2018 │        1 │         1 │
│       1 │ 02-06-2018 │        1 │         1 │
│       1 │ 03-08-2018 │        1 │         1 │
│       1 │ 03-09-2018 │        1 │         1 │
│       1 │ 03-11-2018 │        2 │         1 │
│       1 │ 03-12-2018 │        1 │         3 │
│       1 │ 04-01-2018 │        1 │         1 │
│       1 │ 04-04-2018 │        1 │         1 │
│       1 │ 04-08-2018 │        1 │         2 │
│       1 │ 04-10-2018 │        1 │         2 │
│       1 │ 05-02-2018 │        1 │         1 │
│       1 │ 05-04-2018 │        1 │         1 │
└─────────┴────────────┴──────────┴───────────┘
```

```
DESCRIBE prodriges_374222.distributed_day_important_number
```



```
clickhouse-0.clickhouse.clickhouse.svc.cluster.local :) DESCRIBE prodriges_374222.distributed_day_important_number

DESCRIBE TABLE  prodriges_374222.distributed_day_important_number

Query id: 973f0ca2-8c0b-4711-ac6f-970af6a82b6e

┌─name──────┬─type───┬─default_type─┬─default_expression─┬─comment─┬─codec_expression─┬─ttl_expression─┐
│ user_id   │ Int64  │              │                    │         │                  │                │
│ date      │ String │              │                    │         │                  │                │
│ count_in  │ UInt64 │              │                    │         │                  │                │
│ count_out │ UInt64 │              │                    │         │                  │                │
└───────────┴────────┴──────────────┴────────────────────┴─────────┴──────────────────┴────────────────┘
```