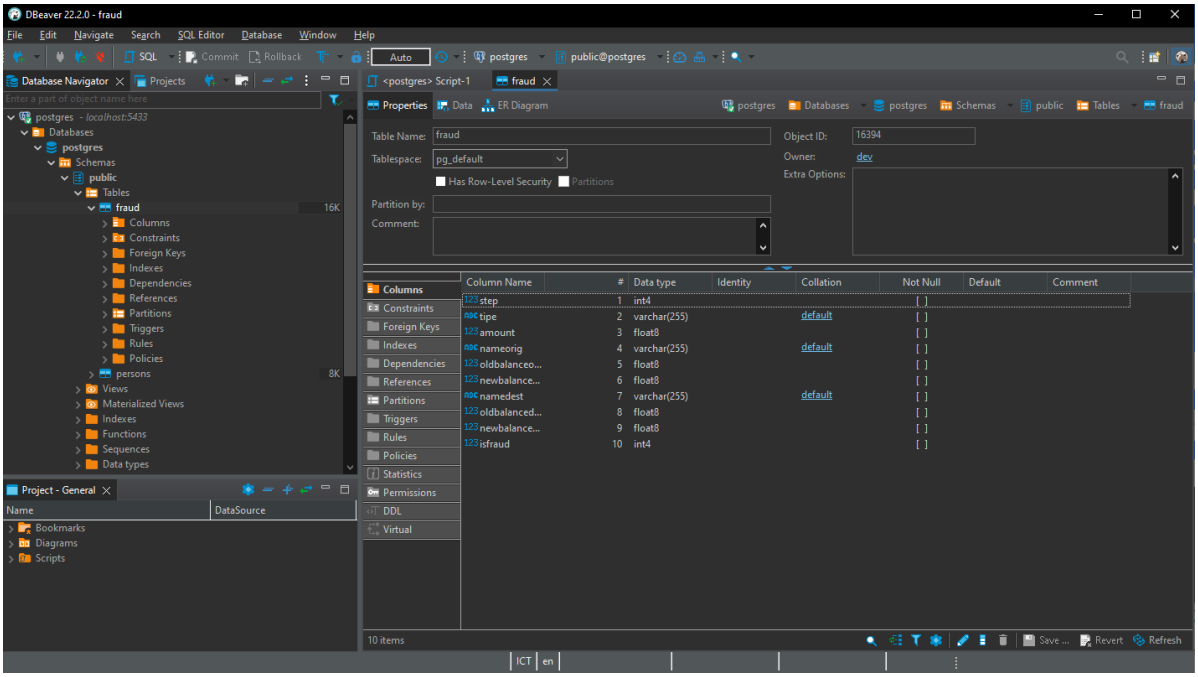
KsqlDB Step by Step

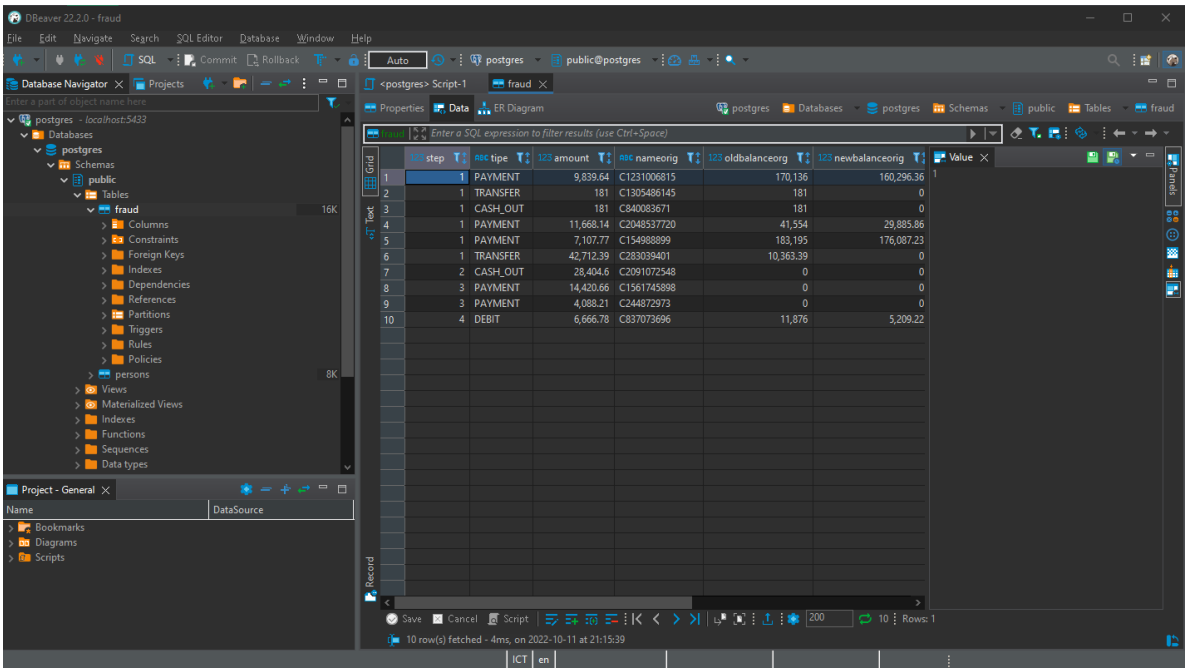
Step 1: Move to docker compose file and run `docker-compose up` command in your terminal. Navigate to ksql CLI using command `docker exec -it ksqldb-cli ksql http://ksqldb-server:8088`.

```
Microsoft Windows [Version 10.0.19043.2006]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Oji\Documents\Docker\ksqldb>docker exec -it ksqldb-cli ksql http://ksqldb-server:8088
OpenJDK 64-Bit Server VM warning: Option UseConcMarkSweepGC was deprecated in version 9.0 and will likely be removed in a future release.

                  ===========================================
                  =       _              _ ____  ____       =
                  =      | | _____  __ _| |  _ \| __ )      =
                  =      | |/ / __|/ _` | | | | |  _ \      =
                  =      |   <\__ \ (_| | | |_| | |_) |     =
                  =      |_|\_\___/\__, |_|____/|____/      =
                  =                   |_|                   =
                  =        The Database purpose-built       =
                  =        for stream processing apps       =
                  ===========================================

Copyright 2017-2022 Confluent Inc.

CLI v7.2.0, Server v7.2.0 located at http://ksqldb-server:8088
Server Status: RUNNING

Having trouble? Type 'help' (case-insensitive) for a rundown of how things work!

ksql>
```

Step 2: Connect your postgres using Dbeaver and set it up based on username, password and ports in your `docker-compose.yml` file. Create a new table based on your preference.



Step 3: Fill in the data in your table with your own decided value. You can see the query file in my repository if you want to use the one that I made.



Step 4: Create stream table in your ksqlDB.

```
ksql> create stream stream_table (step INT, tipe VARCHAR, amount DOUBLE, nameOrig VARCHAR, oldbalanceOrg DOUBLE, newbalanceOrig DOUBLE, nameDest VARCHAR, oldbalanceDest DOUBLE, newbalanceDest DOUBLE, isFraud
INT)
>with (kafka_topic='frauds', value_format='json', partitions=1);

 Message
----------------
 Stream created
----------------
ksql> show streams;

 Stream Name         | Kafka Topic                 | Key Format | Value Format | Windowed
-------------------------------------------------------------------------------------------
 KSQL_PROCESSING_LOG | default_ksql_processing_log | KAFKA      | JSON         | false
 STREAM_TABLE        | frauds                      | KAFKA      | JSON         | false
-------------------------------------------------------------------------------------------
ksql> describe stream_table;

Name                 : STREAM_TABLE
 Field          | Type
----------------------------------
 STEP           | INTEGER
 TIPE           | VARCHAR(STRING)
 AMOUNT         | DOUBLE
 NAMEORIG       | VARCHAR(STRING)
 OLDBALANCEORG  | DOUBLE
 NEWBALANCEORIG | DOUBLE
 NAMEDEST       | VARCHAR(STRING)
 OLDBALANCEDEST | DOUBLE
 NEWBALANCEDEST | DOUBLE
 ISFRAUD        | INTEGER
```

Step 5: Create materialized table of your own choice based on your stream table. You can take a look at my repository query if you want to use the same table.

```
ksql> show tables;

 Table Name       | Kafka Topic      | Key Format | Value Format | Windowed
--------------------------------------------------------------------------------
 FINAL_TABLE      | FINAL_TABLE      | KAFKA      | JSON         | false
 TRANSACTIONUSER  | TRANSACTIONUSER  | KAFKA      | JSON         | false
--------------------------------------------------------------------------------
ksql> describe final_table;

Name                 : FINAL_TABLE
 Field              | Type
-----------------------------------------------------
 NAMEORIG           | VARCHAR(STRING)  (primary key)
 TRANSACTION_TIMES  | BIGINT
 TOTAL_AMOUNT       | DOUBLE
-----------------------------------------------------
```

```
ksql> describe transactionuser;

Name                 : TRANSACTIONUSER
 Field              | Type
-----------------------------------------------------
 NAMEORIG           | VARCHAR(STRING)  (primary key)
 TOTAL_TRANSACTIONS | BIGINT
-----------------------------------------------------
For runtime statistics and query details run: DESCRIBE <Stream,Table> EXTENDED;
ksql>
```

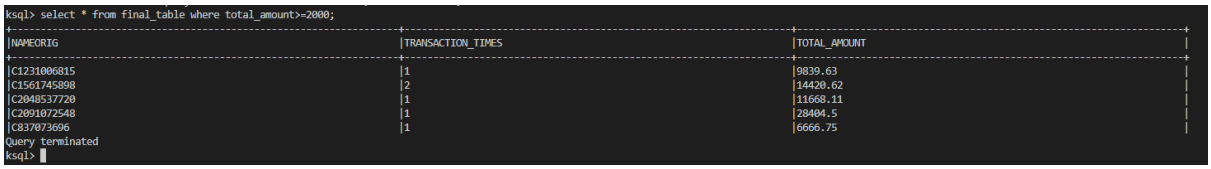
Step 6: Insert data from a new CLI using sql commands to add data and our stream will capture data that is met the conditions given.

```
ksql> insert into stream_table (step, tipe, amount, nameOrig, oldbalanceOrg, newbalanceOrig, nameDest, oldbalanceDest, newbalanceDest, isFraud) values (1, 'PAYMENT', 9839.63, 'C1231006815', 170131, 160296.32,
 'M1979787155', 0, 0, 0);
ksql> insert into stream_table (step, tipe, amount, nameOrig, oldbalanceOrg, newbalanceOrig, nameDest, oldbalanceDest, newbalanceDest, isFraud) values (1, 'TRANSFER', 182, 'C1305486145', 182, 0, 'C553264065',
 0, 0, 1)
>;
ksql> insert into stream_table (step, tipe, amount, nameOrig, oldbalanceOrg, newbalanceOrig, nameDest, oldbalanceDest, newbalanceDest, isFraud) values (3, 'PAYMENT', 14420.62, 'C1561745898', 0, 0, 'M203326892
5', 0, 0, 0);
ksql> insert into stream_table (step, tipe, amount, nameOrig, oldbalanceOrg, newbalanceOrig, nameDest, oldbalanceDest, newbalanceDest, isFraud) values (3, 'PAYMENT', 0, 'C1561745898', 0, 0, 'M203326892
0, 0);
ksql> insert into stream_table (step, tipe, amount, nameOrig, oldbalanceOrg, newbalanceOrig, nameDest, oldbalanceDest, newbalanceDest, isFraud) values (4, 'DEBIT', 6666.75, 'C837073696', 11875, 5209.25, 'C655
381473', 24777, 189534.74, 0);
ksql> insert into stream_table (step, tipe, amount, nameOrig, oldbalanceOrg, newbalanceOrig, nameDest, oldbalanceDest, newbalanceDest, isFraud) values (2, 'CASH_OUT', 28404.5, 'C2091072548', 0, 0, 'C128278802
5', 51740, 0, 0);
ksql>
```

```
ksql> select * from stream_table
>where amount>0 emit changes;
+------------+------------+------------+------------+------------+------------+------------+------------+------------+------------+
|STEP        |TIPE        |AMOUNT      |NAMEORIG    |OLDBALANCEORG|NEWBALANCEORIG|NAMEDEST    |OLDBALANCEDEST|NEWBALANCEDEST|ISFRAUD     |
+------------+------------+------------+------------+------------+------------+------------+------------+------------+------------+
|1           |PAYMENT     |9839.63     |C1231006815 |170131.0    |160296.32   |M1979787155 |0.0         |0.0         |0           |
|1           |TRANSFER    |182.0       |C1305486145 |182.0       |0.0         |C553264065  |0.0         |0.0         |1           |
|3           |PAYMENT     |14420.62    |C1561745898 |0.0         |0.0         |M2033268925 |0.0         |0.0         |0           |
|4           |DEBIT       |6666.75     |C837073696  |11875.0     |5209.25     |C655381473  |24777.0     |189534.74   |0           |
|2           |CASH_OUT    |28404.5     |C2091072548 |0.0         |0.0         |C1282788025 |51740.0     |0.0         |0           |
Press CTRL-C to interrupt
```

```
ksql> select * from stream_table
>where amount>0 emit changes;
+------------+------------+------------+------------+------------+------------+------------+------------+------------+------------+
|STEP        |TIPE        |AMOUNT      |NAMEORIG    |OLDBALANCEORG|NEWBALANCEORIG|NAMEDEST    |OLDBALANCEDEST|NEWBALANCEDEST|ISFRAUD     |
+------------+------------+------------+------------+------------+------------+------------+------------+------------+------------+
|1           |PAYMENT     |9839.63     |C1231006815 |170131.0    |160296.32   |M1979787155 |0.0         |0.0         |0           |
|1           |TRANSFER    |182.0       |C1305486145 |182.0       |0.0         |C553264065  |0.0         |0.0         |1           |
|3           |PAYMENT     |14420.62    |C1561745898 |0.0         |0.0         |M2033268925 |0.0         |0.0         |0           |
|4           |DEBIT       |6666.75     |C837073696  |11875.0     |5209.25     |C655381473  |24777.0     |189534.74   |0           |
|2           |CASH_OUT    |28404.5     |C2091072548 |0.0         |0.0         |C1282788025 |51740.0     |0.0         |0           |
|1           |PAYMENT     |11668.11    |C2048537720 |41551.0     |29885.84    |M1230701703 |0.0         |0.0         |0           |
Press CTRL-C to interrupt
```

Step 7: Use select sql command to get data for your final table with conditions given.



```
ksql> select * from final_table where total_amount>=2000;
+-----------------------------------------------------------------------+-----------------------------------------------------------------------+-----------------------------------------------------------------------+
|NAMEORIG                                                               |TRANSACTION_TIMES                                                      |TOTAL_AMOUNT                                                           |
+-----------------------------------------------------------------------+-----------------------------------------------------------------------+-----------------------------------------------------------------------+
|C1231006815                                                            |1                                                                      |9839.63                                                                |
|C1561745898                                                            |2                                                                      |14420.62                                                               |
|C2048537720                                                            |1                                                                      |11668.11                                                               |
|C2091072548                                                            |1                                                                      |28404.5                                                                |
|C837073696                                                             |1                                                                      |6666.75                                                                |
Query terminated
ksql>
```