

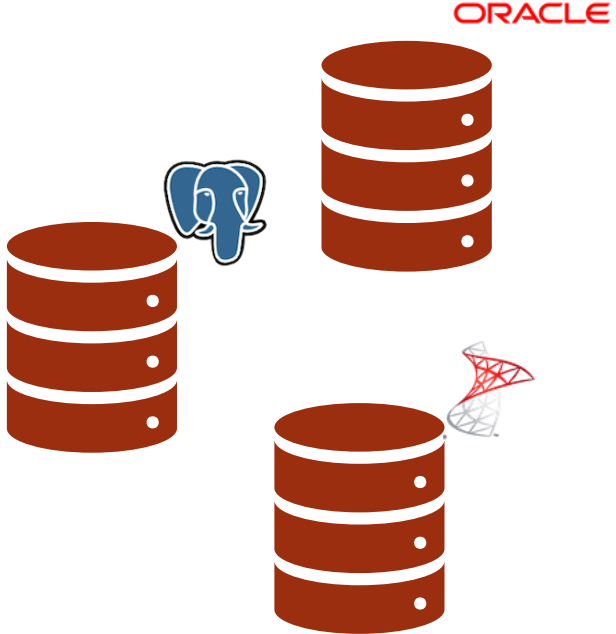
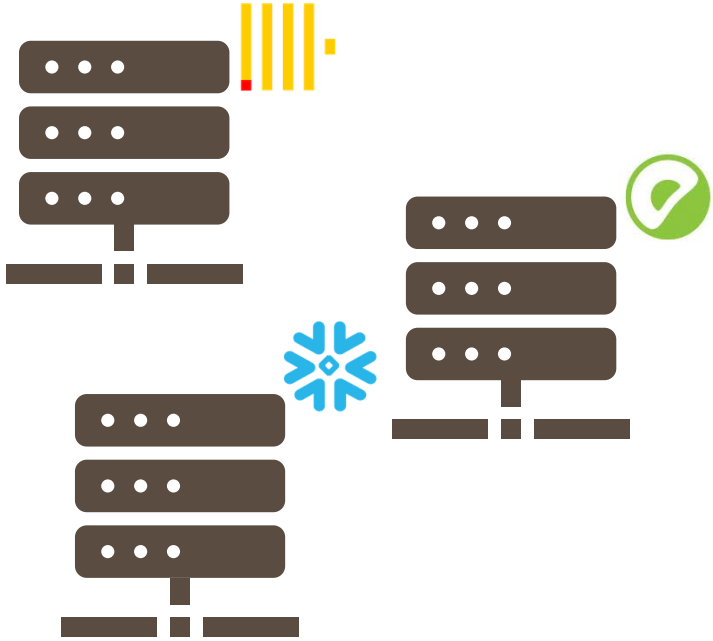


FOREIGN TABLES MODELLING IN UI

TATIANA KRUPENYA AND SERGE RIDER

dbeaver.com

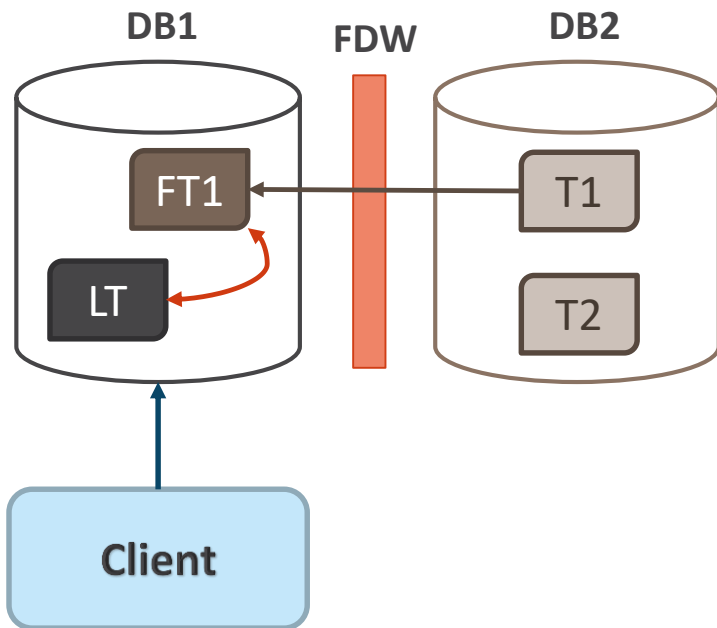
MULTI-DATABASE WORLD



CROSS-DATABASE LINKS

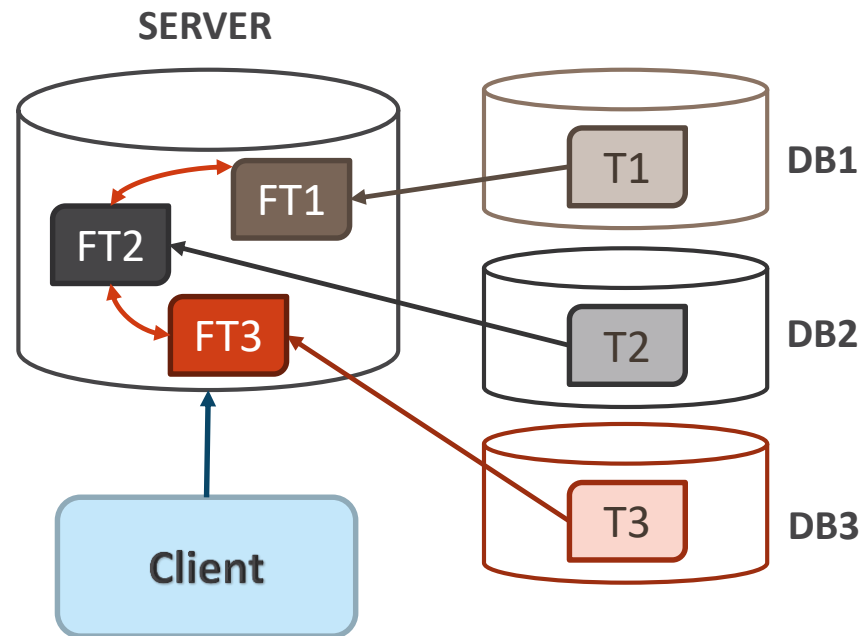
Native

- PostgreSQL FDW
- Oracle Database Links
- DB2 Federated tables



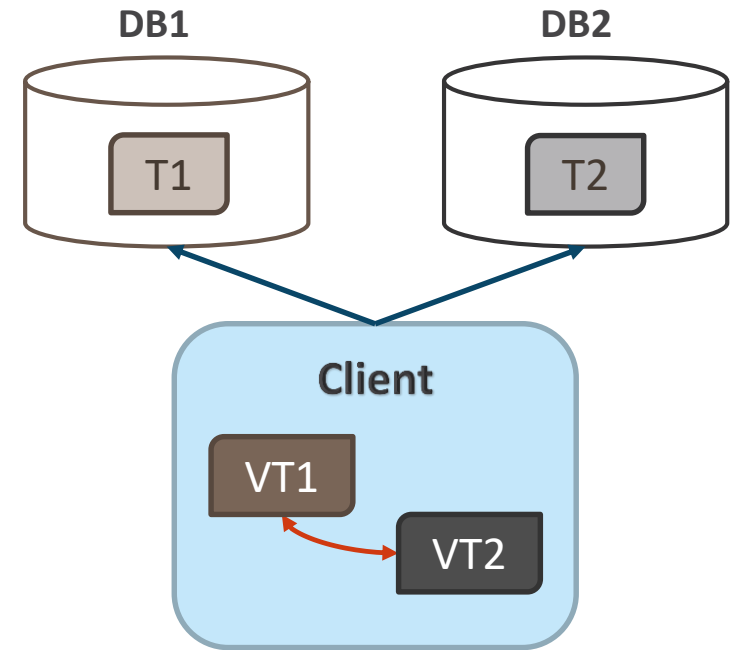
“Composite” Server

- Presto DB



Client side

- Unity JDBC
- DBeaver VFK



DBEAVER FOR POSTGRESQL

- ✧ Metadata viewer and editor
- ✧ Custom data types rendering/editing
- ✧ GIS data viewer
- ✧ Session manager
- ✧ Locks manager
- ✧ SQL query execution plan
- ✧ Administrative tools: vacuum, analyze
- ✧ User/role permissions editor
- ✧ Server health dashboards
- ✧ Foreign data wrappers management
- ✧ Backup/restore wizards

The screenshot displays the DBeaver Enterprise 6.2.2 interface for a PostgreSQL 11 database. The main window shows a table named 'trip_data' with columns 'vendor_id', 'pickup_datetime', and 'dropoff_datetime'. The table contains 13 rows of data for taxi trips on 2013-01-01. A map view is visible on the right, showing the location of the trips in Manhattan, New York City, near the Jacqueline Kennedy Onassis Reservoir. The interface also includes a Database Navigator on the left, a Query Manager at the bottom, and three server health dashboards: 'Server sessions', 'Transactions per second', and 'Block IO'.

	ABC vendor_id	pickup_datetime	dropoff_datetime
1	VTS	2013-01-01 00:00:00	2013-01-01 00:05:00
2	VTS	2013-01-01 00:00:00	2013-01-01 00:09:00
3	VTS	2013-01-01 00:00:00	2013-01-01 00:16:00
4	VTS	2013-01-01 00:00:00	2013-01-01 00:05:00
5	VTS	2013-01-01 00:00:00	2013-01-01 00:08:00
6	VTS	2013-01-01 00:00:00	2013-01-01 00:04:00
7	VTS	2013-01-01 00:00:00	2013-01-01 00:05:00
8	VTS	2013-01-01 00:00:00	2013-01-01 00:08:00
9	VTS	2013-01-01 00:00:00	2013-01-01 00:07:00
10	VTS	2013-01-01 00:00:00	2013-01-01 00:06:00
11	VTS	2013-01-01 00:00:00	2013-01-01 00:04:00
12	VTS	2013-01-01 00:00:00	2013-01-01 00:02:00
13	VTS	2013-01-01 00:00:00	2013-01-01 00:10:00

STORY ABOUT THE LOST WALLET



A person has taken a taxi around 11 am near the Water str. 77 and left a wallet in the car.

Can we find it?

WHAT DO WE HAVE?



Data about the taxi drivers:

- Only information about taxi drivers
- A few gigabytes of data
- Structured data



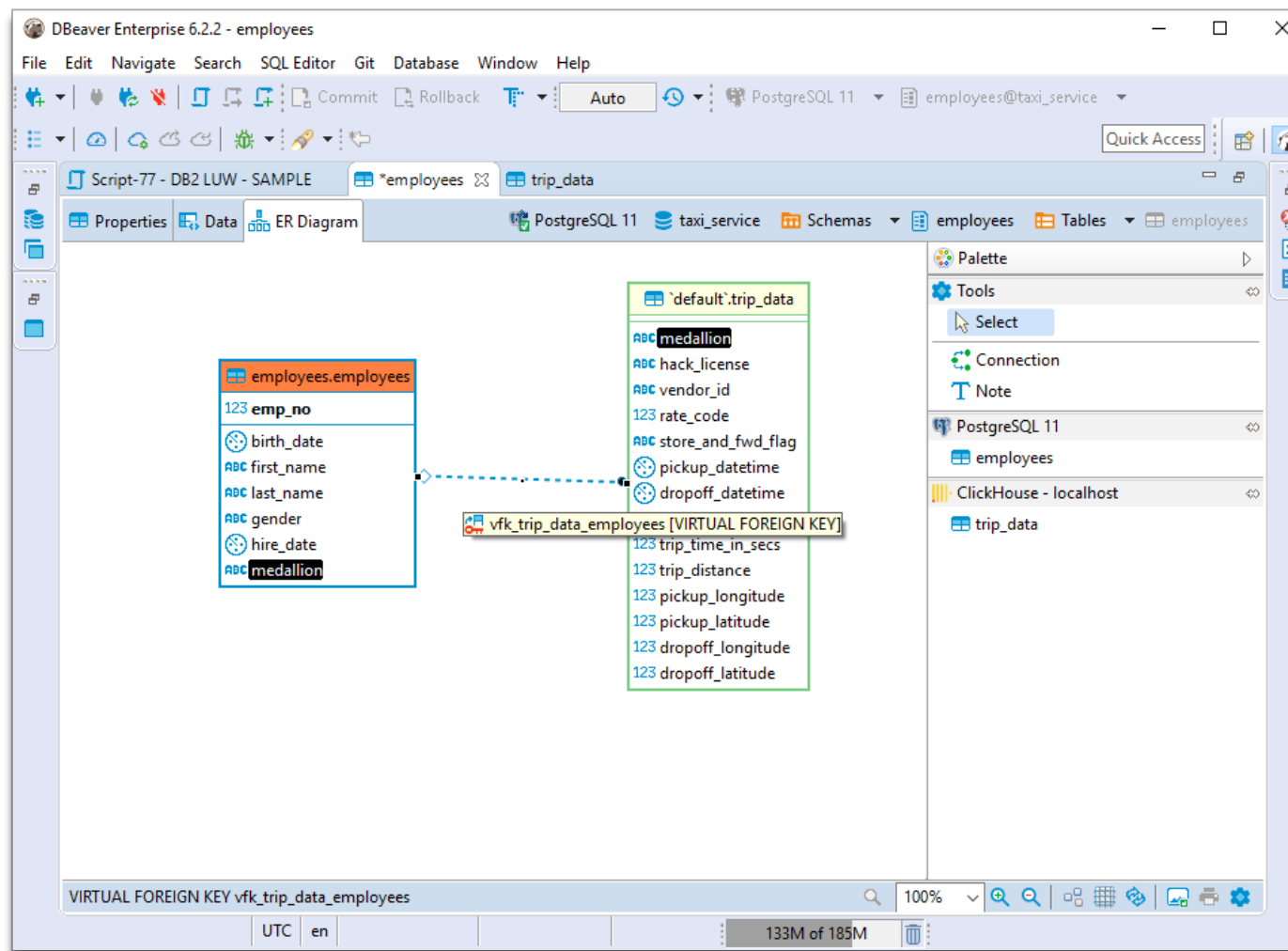
Statistic data about the trips:

- Data about all taxi trips
- A few terabytes of data
- Support analytical queries

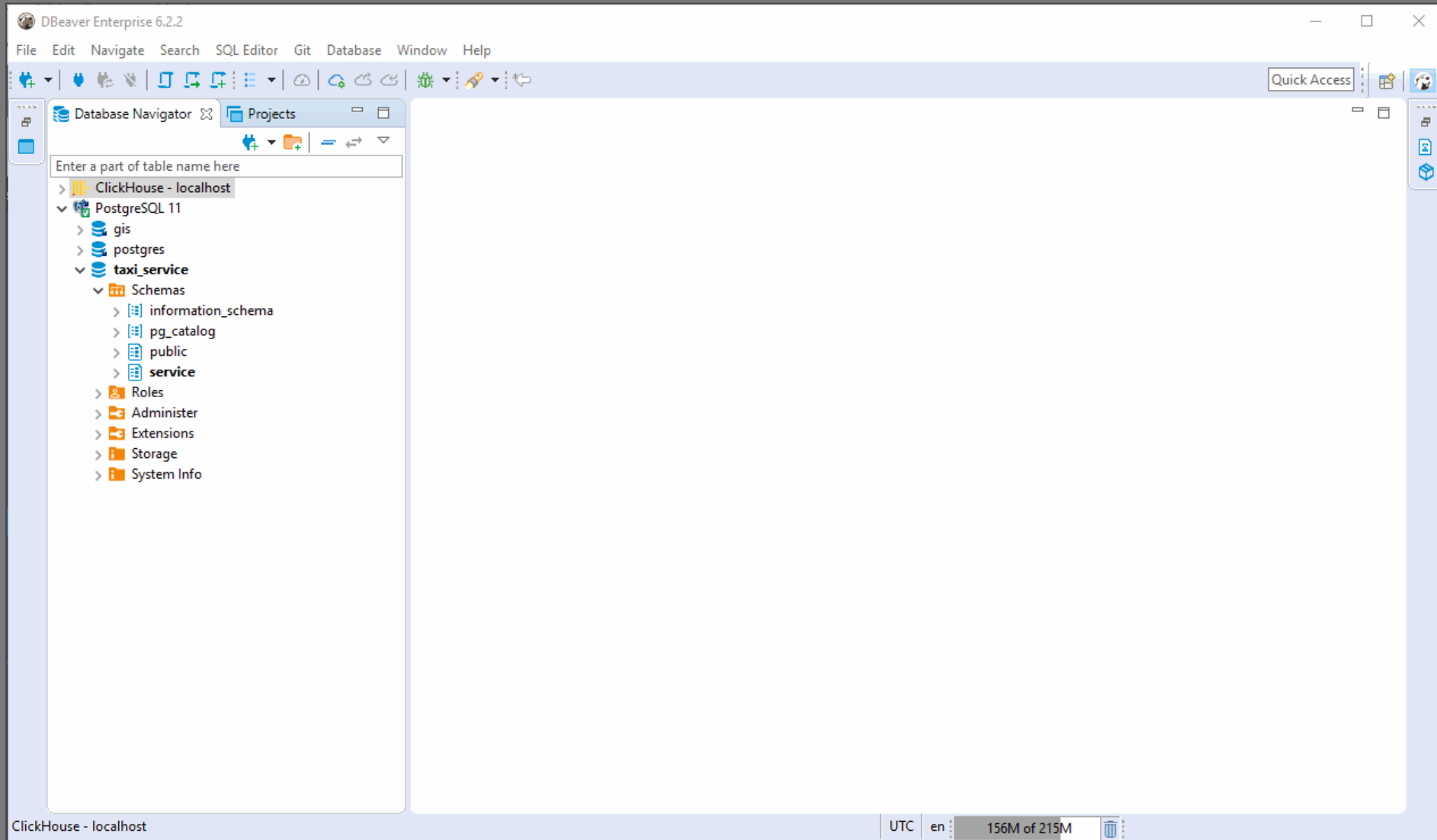
MAKE VIRTUAL CONNECTIONS

Virtual foreign keys

- Allows navigation across tables
- Show referenced data from different databases in one table
- Can connect any databases, including non-relational
- Don't provide referential integrity



HOW TO CREATE VFK



MAKE REAL CONNECTIONS

FDW INSTALL SCRIPT

Foreign Data Wrappers

Benefits:

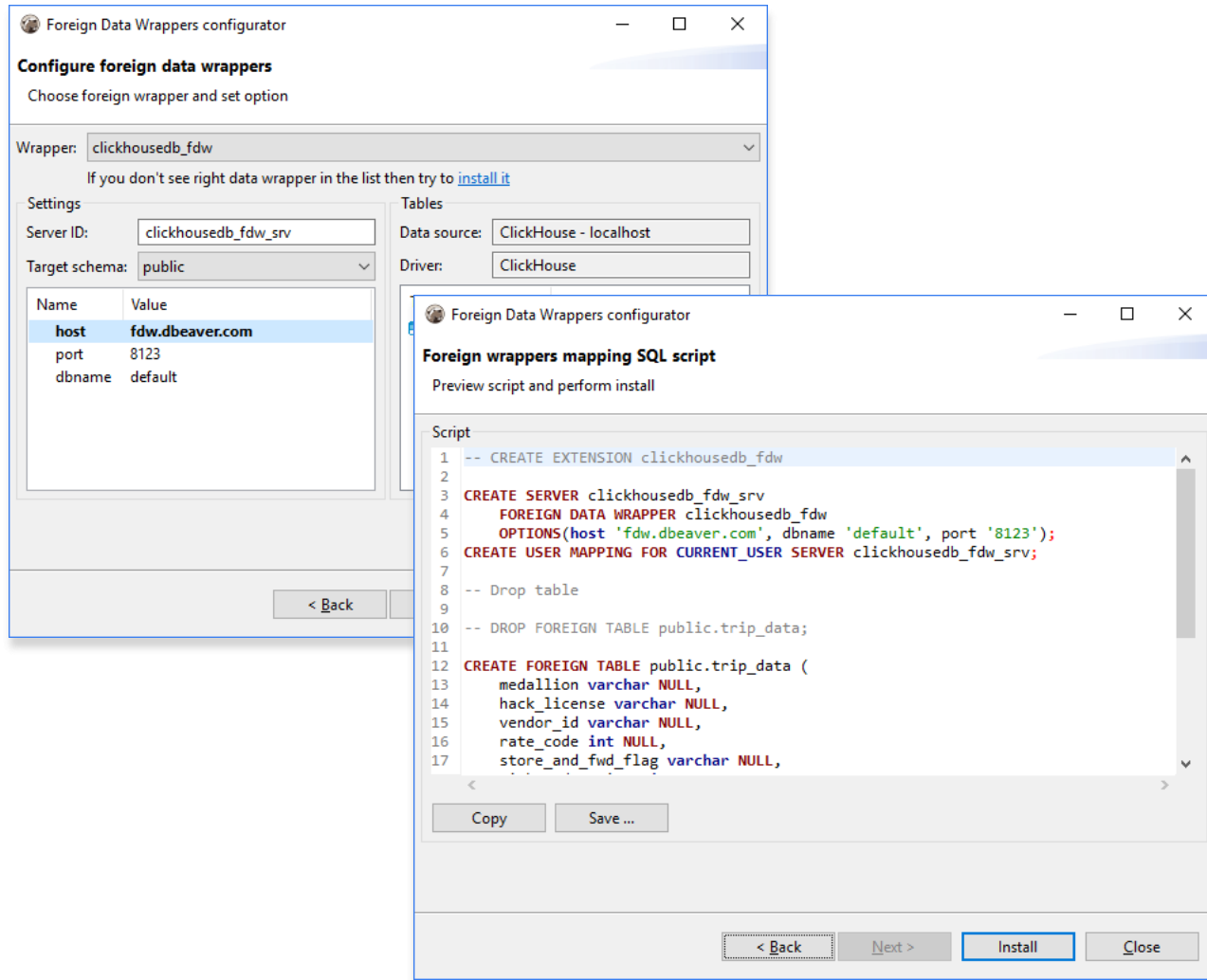
- ✓ Create logical tables and use them just like local ones
- ✓ Join tables from different databases

Problems:

- ✗ FDW extension installation can be tricky
- ✗ Difficult to configure mappings
- ✗ Potential performance problems

```
CREATE SERVER clickhousedb_fdw_srv
FOREIGN DATA WRAPPER clickhousedb_fdw
OPTIONS(host '127.0.0.1', dbname '${database}', port '39523');
CREATE USER MAPPING FOR CURRENT_USER SERVER clickhousedb_fdw_srv;
-- Drop table
-- DROP FOREIGN TABLE public.ontime;
CREATE FOREIGN TABLE public.ontime (
"Year" int NULL,
"Quarter" int NULL,
"Month" int NULL,
"DayofMonth" int NULL,
"DayOfWeek" int NULL,
"FlightDate" timestamp NULL,
"UniqueCarrier" varchar NULL,
"AirlineID" int NULL,
"Carrier" varchar NULL,
"TailNum" varchar NULL,
"FlightNum" varchar NULL,
"OriginAirportID" int NULL,
"OriginAirportSeqID" int NULL,
"OriginCityMarketID" int NULL,
"Origin" varchar NULL,
"OriginCityName" varchar NULL,
"OriginState" varchar NULL,
"OriginStateFips" varchar NULL,
"OriginStateName" varchar NULL,
"OriginWac" int NULL,
"DestAirportID" int NULL,
"DestAirportSeqID" int NULL,
"DestCityMarketID" int NULL,
"Dest" varchar NULL,
"DestCityName" varchar NULL,
"DestState" varchar NULL,
"DestStateFips" varchar NULL,
"DestStateName" varchar NULL,
"DestWac" int NULL,
"CRSDepTime" int NULL,
"DepTime" int NULL,
"DepDelay" int NULL,
"DepDelayMinutes" int NULL,
"DepDel15" int NULL,
"DepartureDelayGroups" varchar NULL,
"DepTimeBlk" varchar NULL,
"TaxiOut" int NULL,
"WheelsOff" int NULL,
"WheelsOn" int NULL,
"TaxiIn" int NULL,
"CRSArrTime" int NULL,
"ArrTime" int NULL,
"ArrDelay" int NULL,
"ArrDelayMinutes" int NULL,
"ArrDel15" int NULL,
"ArrivalDelayGroups" int NULL,
"ArrTimeBlk" varchar NULL,
"Cancelled" int NULL,
"CancellationCode" varchar NULL,
"Diverted" int NULL,
"CRSElapsedTime" int NULL,
"ActualElapsedTime" int NULL,
"AirTime" int NULL,
"Flights" int NULL,
"Distance" int NULL,
"DistanceGroup" int NULL,
"CarrierDelay" int NULL,
"WeatherDelay" int NULL,
"NASDelay" int NULL,
"SecurityDelay" int NULL,
"LateAircraftDelay" int NULL,
"FirstDepTime" varchar NULL,
"TotalAddGTime" varchar NULL,
"LongestAddGTime" varchar NULL,
"DivAirportLandings" varchar NULL,
"DivReachedDest" varchar NULL,
"DivActualElapsedTime" varchar NULL,
"DivArrDelay" varchar NULL,
"DivDistance" varchar NULL,
"Div1Airport" varchar NULL,
"Div1AirportID" int NULL,
"Div1AirportSeqID" int NULL,
"Div1WheelsOn" varchar NULL,
"Div1TotalGTime" varchar NULL,
"Div1LongestGTime" varchar NULL,
"Div1WheelsOff" varchar NULL,
"Div1TailNum" varchar NULL,
"Div2Airport" varchar NULL,
"Div2AirportID" int NULL,
"Div2AirportSeqID" int NULL,
"Div2WheelsOn" varchar NULL,
"Div2TotalGTime" varchar NULL,
"Div2LongestGTime" varchar NULL,
"Div2WheelsOff" varchar NULL,
"Div2TailNum" varchar NULL,
"Div3Airport" varchar NULL,
"Div3AirportID" int NULL,
"Div3AirportSeqID" int NULL,
"Div3WheelsOn" varchar NULL,
"Div3TotalGTime" varchar NULL,
"Div3LongestGTime" varchar NULL,
"Div3WheelsOff" varchar NULL,
"Div3TailNum" varchar NULL,
"Div4Airport" varchar NULL,
"Div4AirportID" int NULL,
"Div4AirportSeqID" int NULL,
"Div4WheelsOn" varchar NULL,
"Div4TotalGTime" varchar NULL,
"Div4LongestGTime" varchar NULL,
"Div4WheelsOff" varchar NULL,
"Div4TailNum" varchar NULL,
"Div5Airport" varchar NULL,
"Div5AirportID" int NULL,
"Div5AirportSeqID" int NULL,
"Div5WheelsOn" varchar NULL,
"Div5TotalGTime" varchar NULL,
"Div5LongestGTime" varchar NULL,
"Div5WheelsOff" varchar NULL,
"Div5TailNum" varchar NULL
)
SERVER clickhousedb_fdw;
```

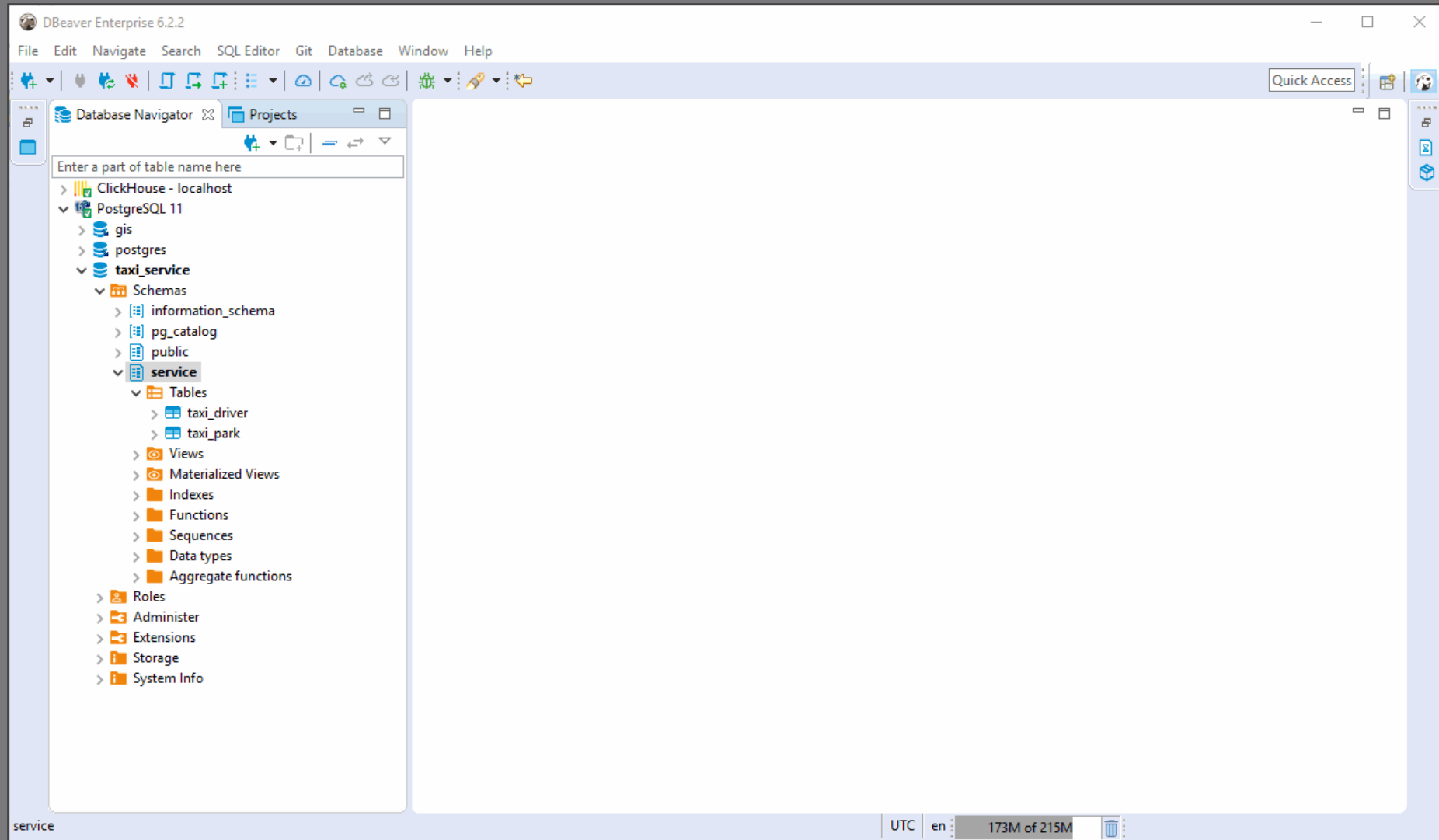
MAKE FDW IN DBEAVER



Why is it easier with DBeaver?

- ✓ Connections (VFK) visualization
- ✓ Foreign data viewer
- ✓ Generate FDW script in a few extra clicks
- ✓ Automatic data types mappings
- ✓ Automatic assignment to proper FDW extension

HOW TO CREATE FDW



BE CAREFUL WITH JOINS

- ⦿ Foreign keys between local and foreign tables exist only in DBeaver logical model.
- ⦿ Performance tuning may be tricky. Do not use complex joins/subqueries in cross-database selects.
- ⦿ Some FDW may work slowly or fail when using range/equals conditions.

RETURN TO OUR PROBLEM

WHAT DO WE KNOW?

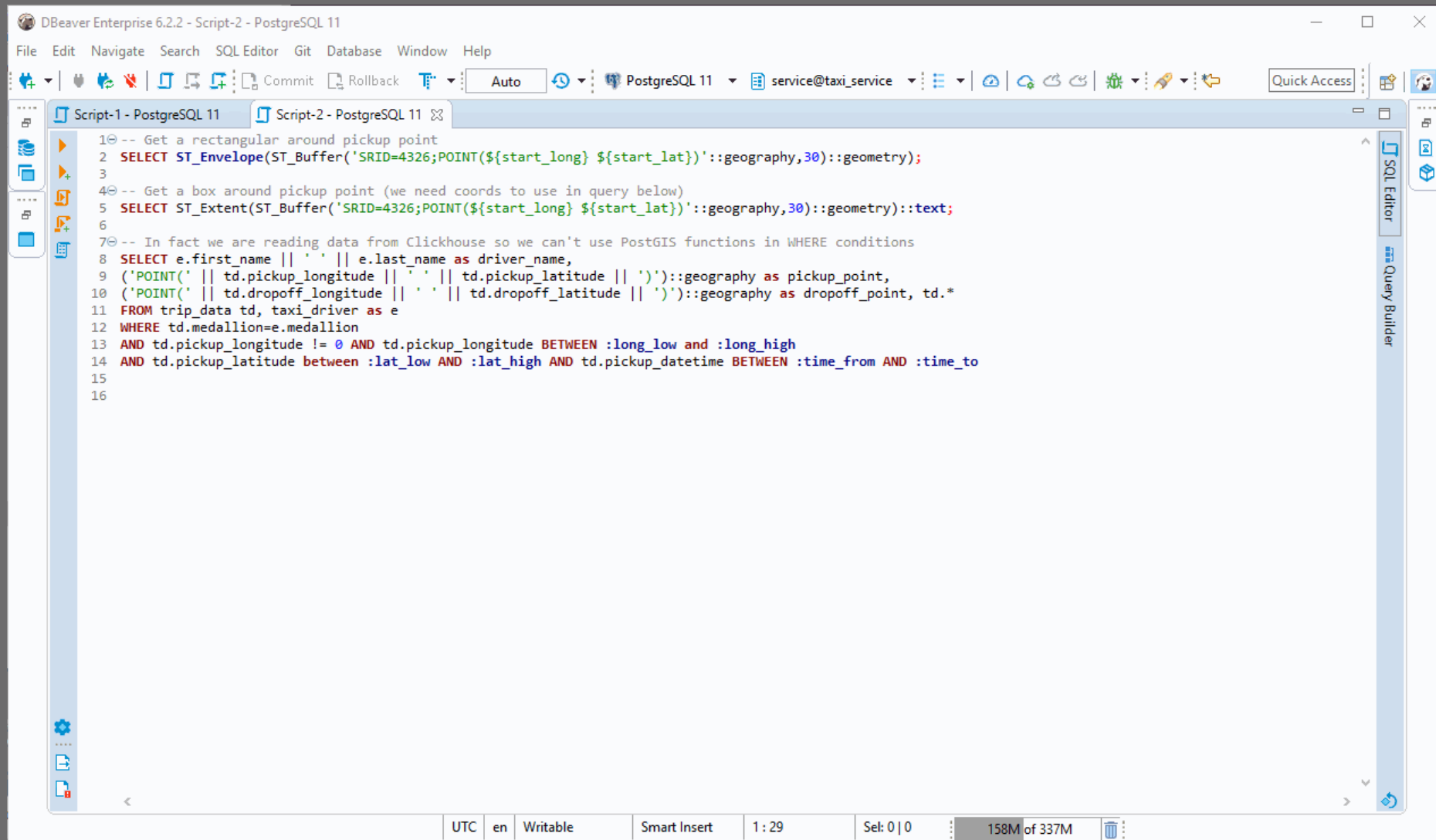
- Pick-up location: Water street 77
- Drop-off location: Columbus Circle
- Pick-up time: around 10:45

SOLUTION

1. Find all trips which started in this area in 30min interval around pick-up time
2. Choose trips which ended in approximate drop-off location
3. Join trip data with drivers data to find driver name/ID.

PROFIT!!!

HOW DOES IT WORK?



```
DBeaver Enterprise 6.2.2 - Script-2 - PostgreSQL 11
File Edit Navigate Search SQL Editor Git Database Window Help
PostgreSQL 11 service@taxi_service
Script-1 - PostgreSQL 11 Script-2 - PostgreSQL 11
1 -- Get a rectangular around pickup point
2 SELECT ST_Envelope(ST_Buffer('SRID=4326;POINT(${start_long} ${start_lat})'::geography,30)::geometry);
3
4 -- Get a box around pickup point (we need coords to use in query below)
5 SELECT ST_Extent(ST_Buffer('SRID=4326;POINT(${start_long} ${start_lat})'::geography,30)::geometry)::text;
6
7 -- In fact we are reading data from Clickhouse so we can't use PostGIS functions in WHERE conditions
8 SELECT e.first_name || ' ' || e.last_name as driver_name,
9 ('POINT(' || td.pickup_longitude || ' ' || td.pickup_latitude || ')')::geography as pickup_point,
10 ('POINT(' || td.dropoff_longitude || ' ' || td.dropoff_latitude || ')')::geography as dropoff_point, td.*
11 FROM trip_data td, taxi_driver as e
12 WHERE td.medallion=e.medallion
13 AND td.pickup_longitude != 0 AND td.pickup_longitude BETWEEN :long_low and :long_high
14 AND td.pickup_latitude between :lat_low AND :lat_high AND td.pickup_datetime BETWEEN :time_from AND :time_to
15
16
UTC en Writable Smart Insert 1:29 Sel: 0 | 0 158M of 337M
```

CONCLUSION

- ❑ FDWs are very handy in some cases
- ❑ Be careful with functions. Queries with foreign tables can't use PG functions if foreign database is not PostgreSQL.
- ❑ Usually FDW don't provide best performance but they are much easier than any custom development.

USEFUL LINKS

USEFUL LINKS

- FDW list: https://wiki.postgresql.org/wiki/Foreign_data_wrappers
- Clickhouse FDW: https://github.com/Percona-Lab/clickhousedb_fdw
- New York TLC database (tip data):
<https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page>

FOLLOW US

- Twitter: https://twitter.com/dbeaver_news
- GitHub: <https://github.com/dbeaver/>
- DBeaver EE: <https://dbeaver.com/>
- DBeaver CE: <https://dbeaver.io/>

CONTACT US

- CEO: tati@dbeaver.com
- CTO: serge@dbeaver.com