КАК УСКОРИТЬ ЗАГРУЗКУ ДАННЫХ В 10 000 РАЗ?

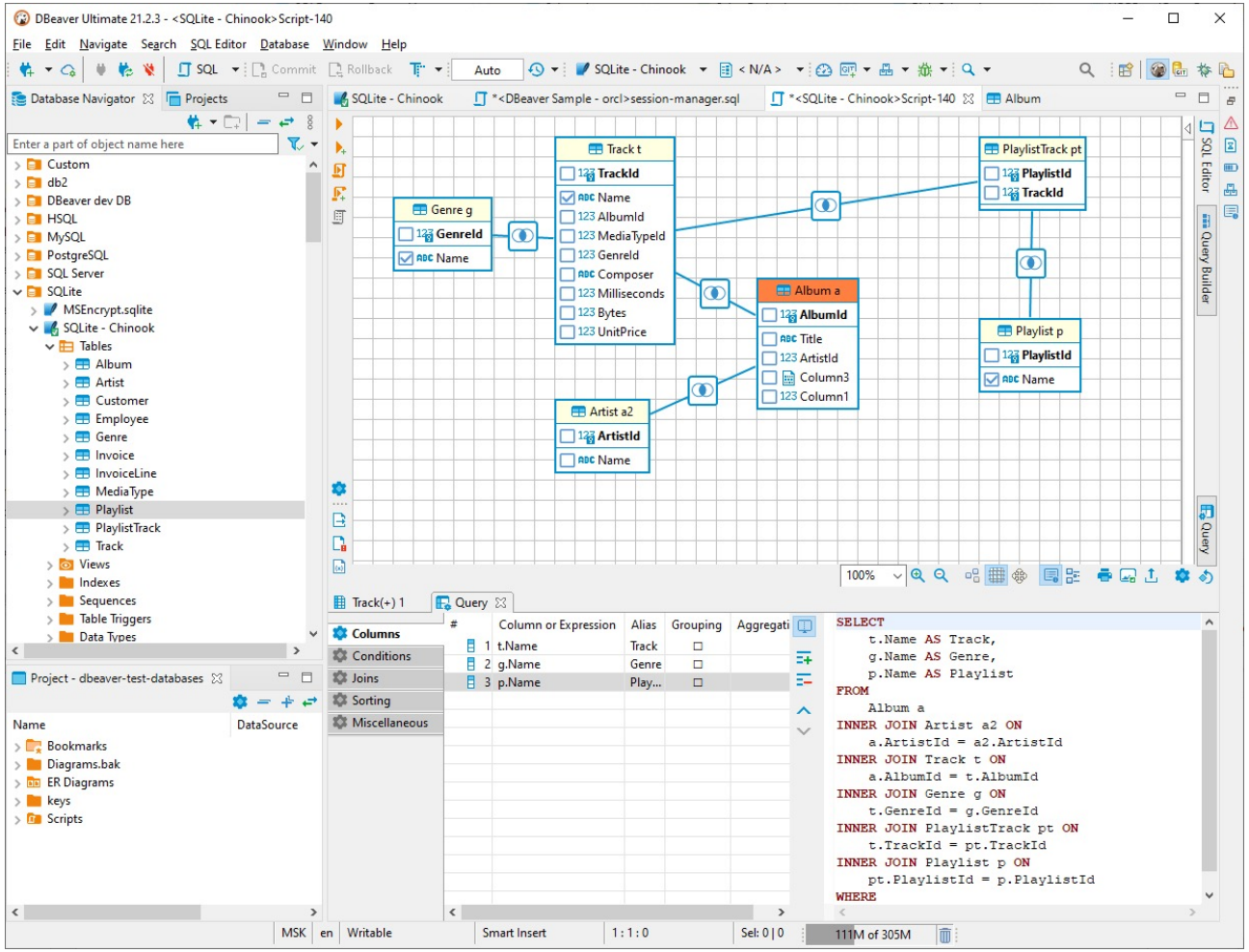
ТАТЬЯНА КРУПЕНЯ И СЕРГЕЙ РИДЕР

dbeaver.com

О НАС

DBeaver

- Универсальный инструмент для работы с базами данных
- DBeaver развивается в двух направлениях: open-source и платные продукты
- Более 6 миллионов пользователей по всему миру



НЕМНОГО О МИГРАЦИИ ДАННЫХ

ЧТО СЛОЖНОГО?

Что у нас есть?

Базы данных:
- PostgreSQL 9-13
- Redshift
- Greenplum

Драйверы:
- PostgreSQL JDBC Driver 42.x
- Redshift JDBC Driver 2.0

Почему просто не взять COPY?

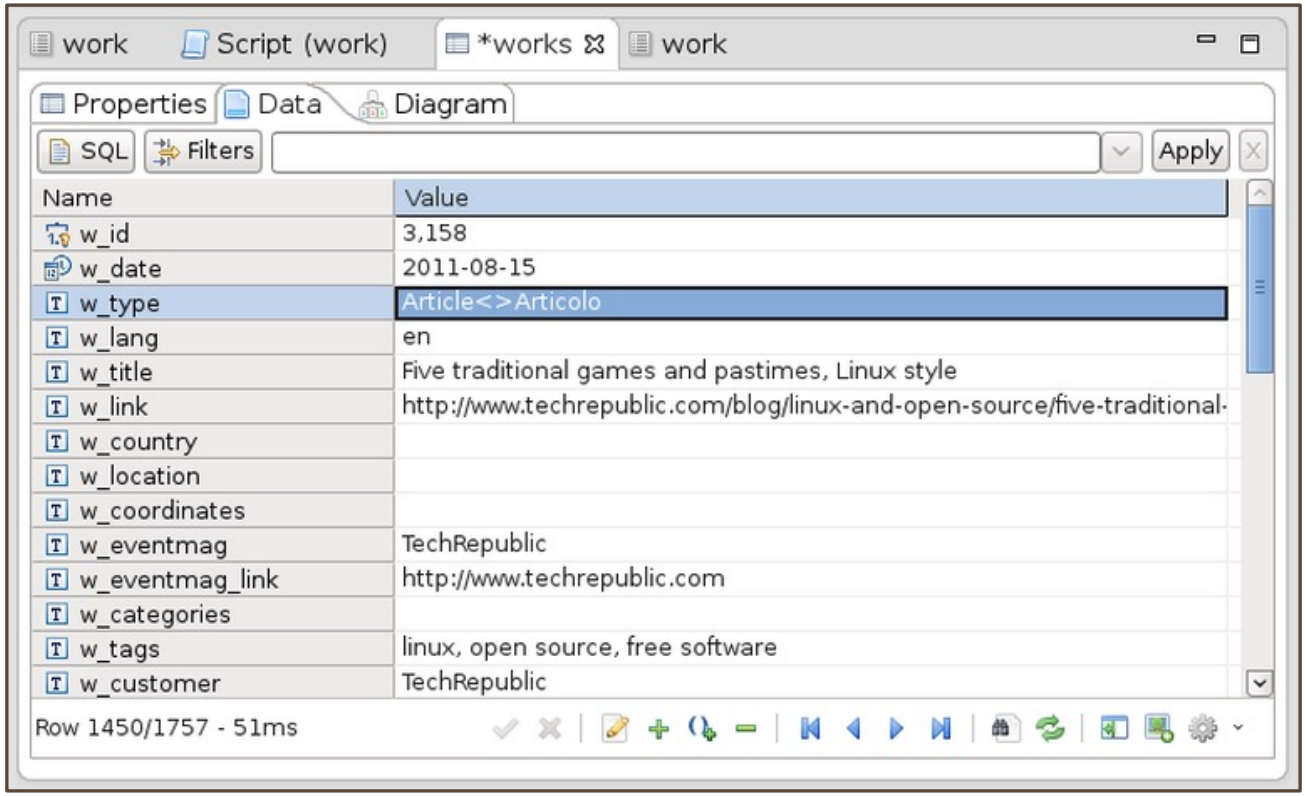
- Нужна трансформация данных
- Не работает, если заливаем не в PostgreSQL

САМЫЙ ПРОСТОЙ СПОСОБ

```
INSERT INTO TEST_TABLE(COL1, COL2, COL3) VALUES (?,?,?);
```

- Вставляем строки по одной
- Используем prepared statements

База данных	Время импорта*
PostgreSQL 12	3 300 мин
Greenplum	11 000 мин
Redshift	11 000 мин
Oracle	3 600 мин
SQL Server	5 200 мин



*** 5 млн записей, auto-commit, prepared statements, AWS Cloud EU-central**

ДОБАВИМ ТРАНЗАКЦИИ

ЛОКАЛЬНАЯ БАЗА

База данных	Время импорта без транзакций*	Время импорта с транзакциями*
PostgreSQL	20 мин	13 мин

VS

УДАЛЕННАЯ БАЗА

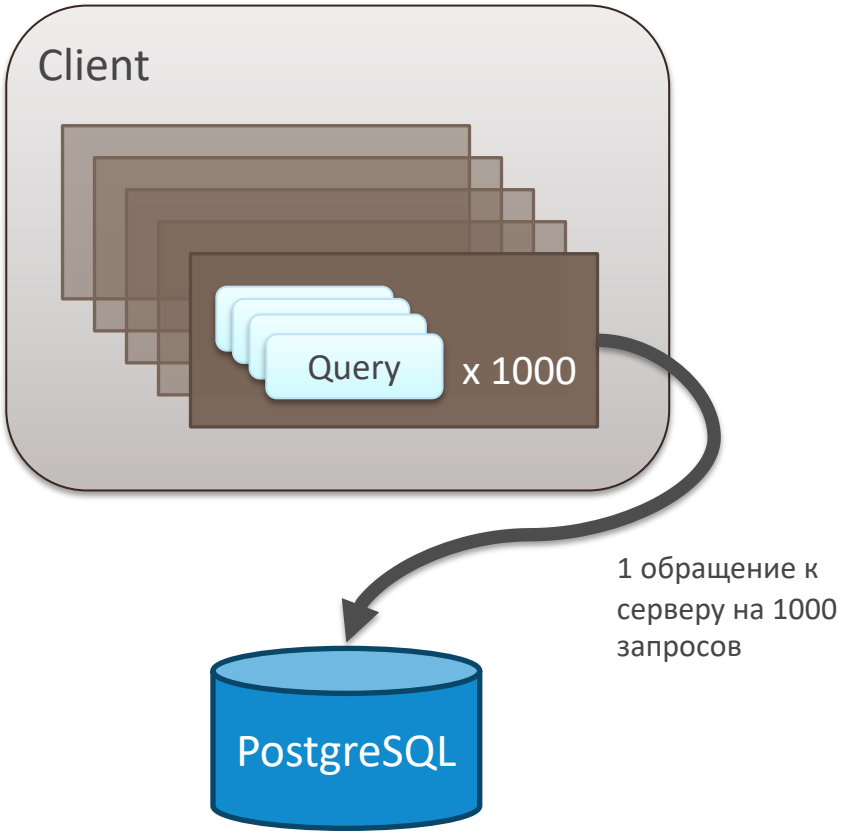
База данных	Время импорта без транзакций**	Время импорта с транзакциями**
PostgreSQL	3 300 мин	3 300 мин
Greenplum	11 000 мин	10 200 мин
Redshift	11 000 мин	8 800 мин
Oracle	3 600 мин	3 350 мин
SQL Server	5 200 мин	4 400 мин

***5 млн записей, commit каждые 10к строк, prepared statements, простая схема без индексов и ключей, локальная база**

****5 млн записей, commit каждые 10к строк, prepared statements, простая схема без индексов и ключей, AWS Cloud EU-central**

УМЕНЬШИМ РАСХОДЫ НА ТРАФИК

JDBC batches



1 обращение к серверу на 1000 запросов

- ✓ Экономия трафика
- ✓ Экономия CPU сервера базы данных
- ✓ Скорость миграции возрастает радикально, если включены транзакции

- ✗ Большое потребление памяти на клиенте
- ✗ Большое потребление CPU

ДОБАВИМ BATCH-ЗАГРУЗКУ

База данных	Время импорта без JDBC batch*	Время импорта с JDBC batch*	Прибавка в скорости
PostgreSQL	3 300 мин	35 мин	в 94 раза
Greenplum	10 200 мин	430 мин	в 24 раза
Redshift	8 800 мин	3 100 мин	в 3 раза
Oracle	3 350 мин	27 мин	в 124 раза
SQL Server	4 400 мин	10 мин	в 440 раз

* 5 млн записей, commit каждые 10к строк, JDBC-batches по 10к запросов, prepared statements, AWS Cloud EU-central

ПОПРОБУЕМ МНОГОСТРОЧНЫЕ ВСТАВКИ

```
INSERT INTO TEST_TABLE(COL1, COL2, COL3) VALUES (?,?,?),(?,?,?),(?,?,?),....,(?,?,?)
```

База данных	Время импорта с однострочными вставками*	Время импорта с многострочными вставками*
PostgreSQL	35 мин	8 мин
Greenplum	430 мин	30 мин
Redshift	3 100 мин	34 мин
Oracle	27 мин	39 мин
SQL Server	10 мин	54 мин

*** 5 млн записей, commit каждые 10к строк, JDBC-batches по 10к запросов, prepared statements, multi-row inserts, AWS Cloud EU-central**

КОГДА REDSHIFT НЕ POSTGRES

ДИНАМИЧЕСКИЕ ПАРАМЕТРЫ

```
INSERT INTO TEST_TABLE(COL1, COL2, COL3)
VALUES (?,?,?), (?,?,?),....,(?,?,?)
```

VS

ЗНАЧЕНИЯ УКАЗАНЫ В ЗАПРОСЕ

```
INSERT INTO TEST_TABLE(COL1, COL2, COL3)
VALUES  (“John”, “Smith”, 1967),(“Vasya”,”Ivanov”, 2001),....,(“Anna”,”Li”,1994)
```

	BINDING	NO BINDING, BATCH 100K
PostgreSQL	8 мин	8 мин
Redshift	34 мин	10 мин

* 5 млн записей, commit каждые 10к строк, JDBC-batches по 10к запросов, multi-row inserts, AWS Cloud EU-central

ВЕРНЕМСЯ К COPY

```
COPY PUBLIC.TARGET_TABLE FROM STDIN (FORMAT CSV)
```

Время импорта в PostgreSQL*	**Удаленная база**	**Локальная база**
С многострочными вставками	10 мин	3 мин
С COPY на стороне клиента	5 мин	1 мин
С COPY на стороне сервера	N/A	<1 мин

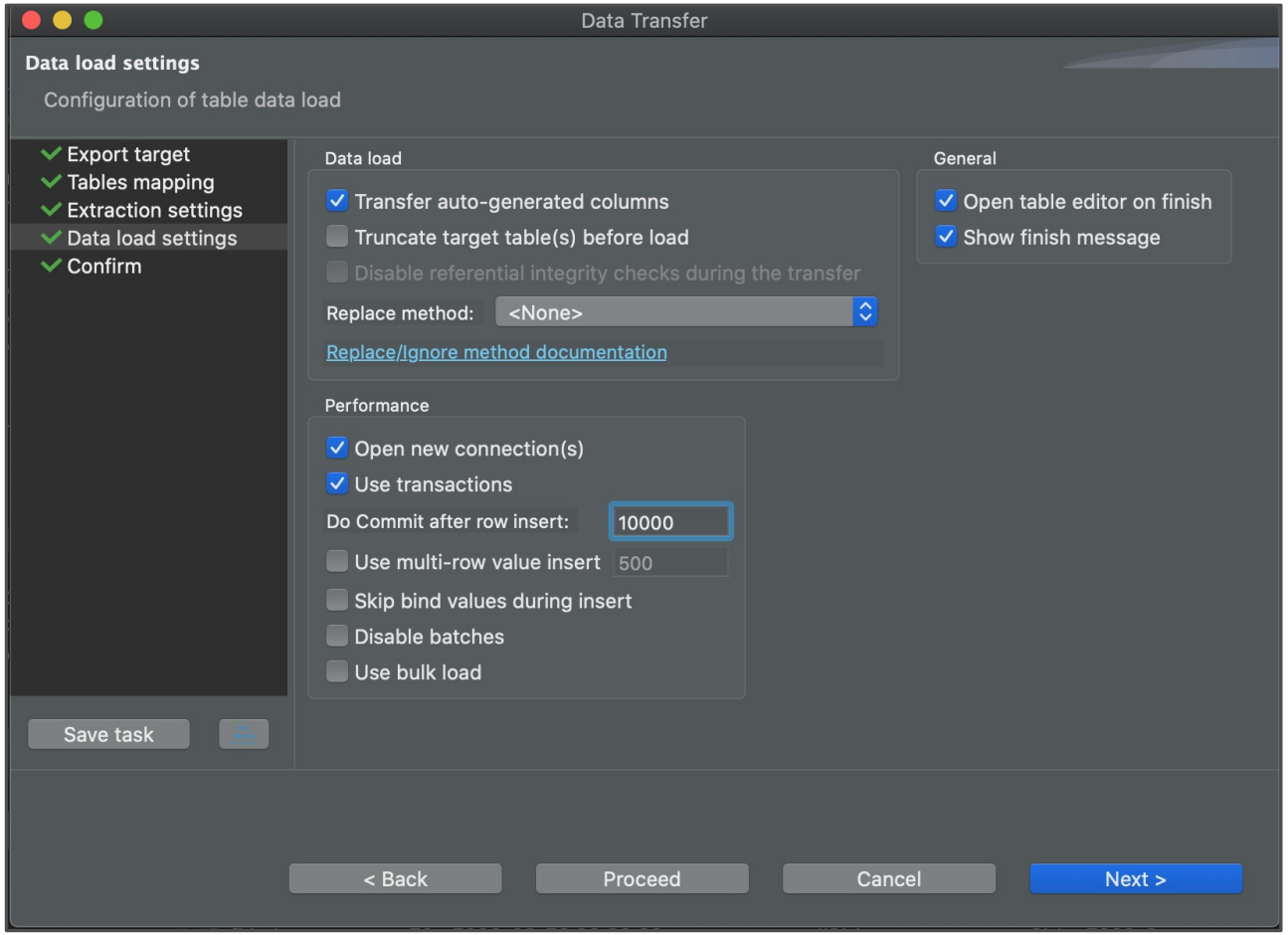
***5 млн записей, commit каждые 10к строк, JDBC-batches по 10к запросов**

- ✓ Самый быстрый способ

- ✗ Невозможно использовать on duplicate update
- ✗ Очень длинные транзакции
- ✗ Не работает в Redshift

ПОИГРАЕМ С ПАРАМЕТРАМИ

- Транзакции помогают в Postgres, но мало влияют на аналитические базы
- Multi-row insert в Postgres (100к + строк в операторе) дает скорость сравнимую с COPY
- Транзакции почти не влияют при больших multi-row операторах
- Отключение привязки параметров помогает в Redshift, а в других базах не влияет или ухудшает
- COPY самый быстрый, но только в ванильном PostgreSQL



ПОЛЕЗНЫЕ ССЫЛКИ

ПОДПИСЫВАЙТЕСЬ

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

ОСТАЛИСЬ ВОПРОСЫ?

- Общие вопросы: tati@dbeaver.com
- Технические вопросы: serge@dbeaver.com