

# Writable FDW

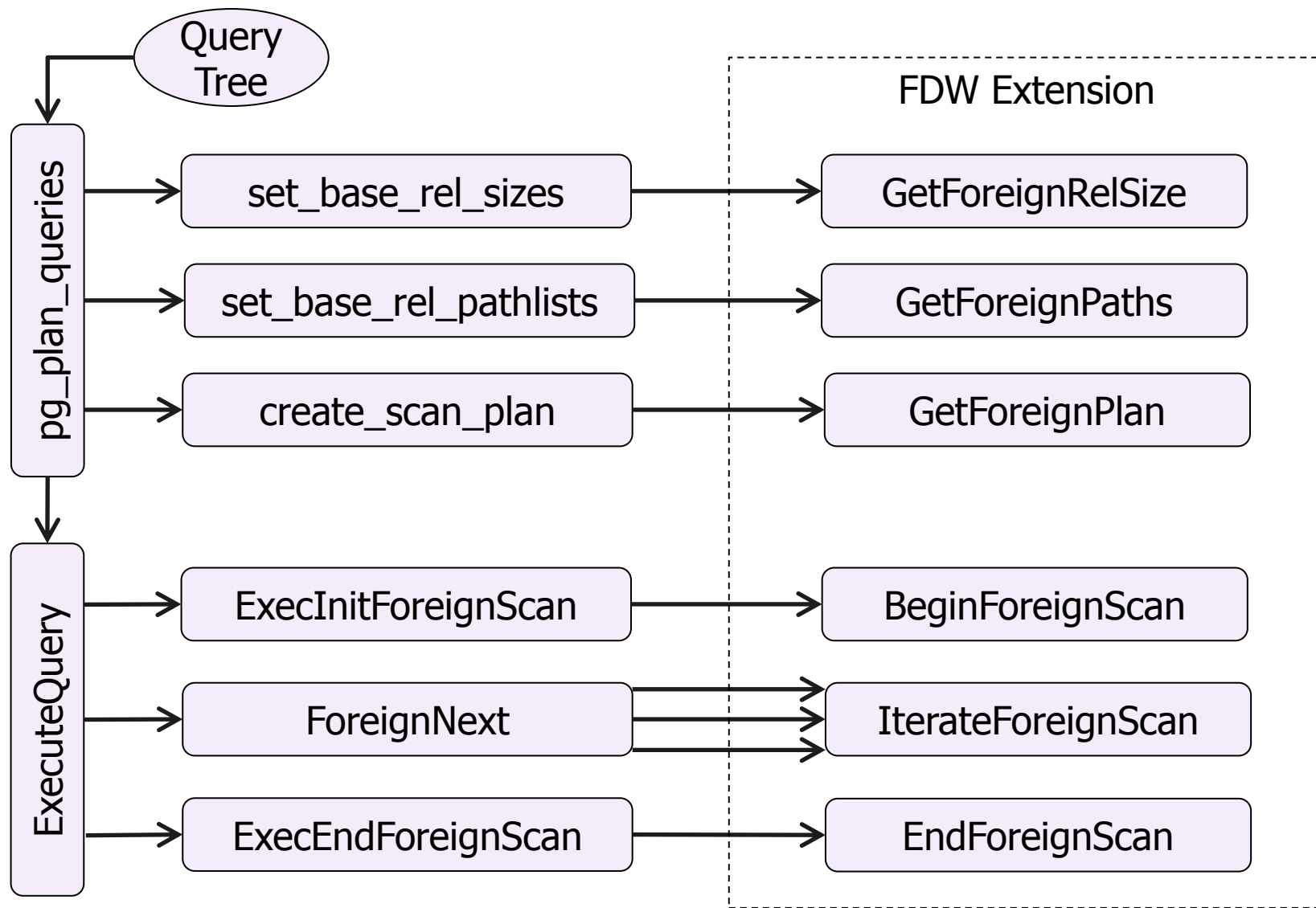
KaiGai Kohei <kaigai@kaigai.gr.jp>

Tw: @kkaigai

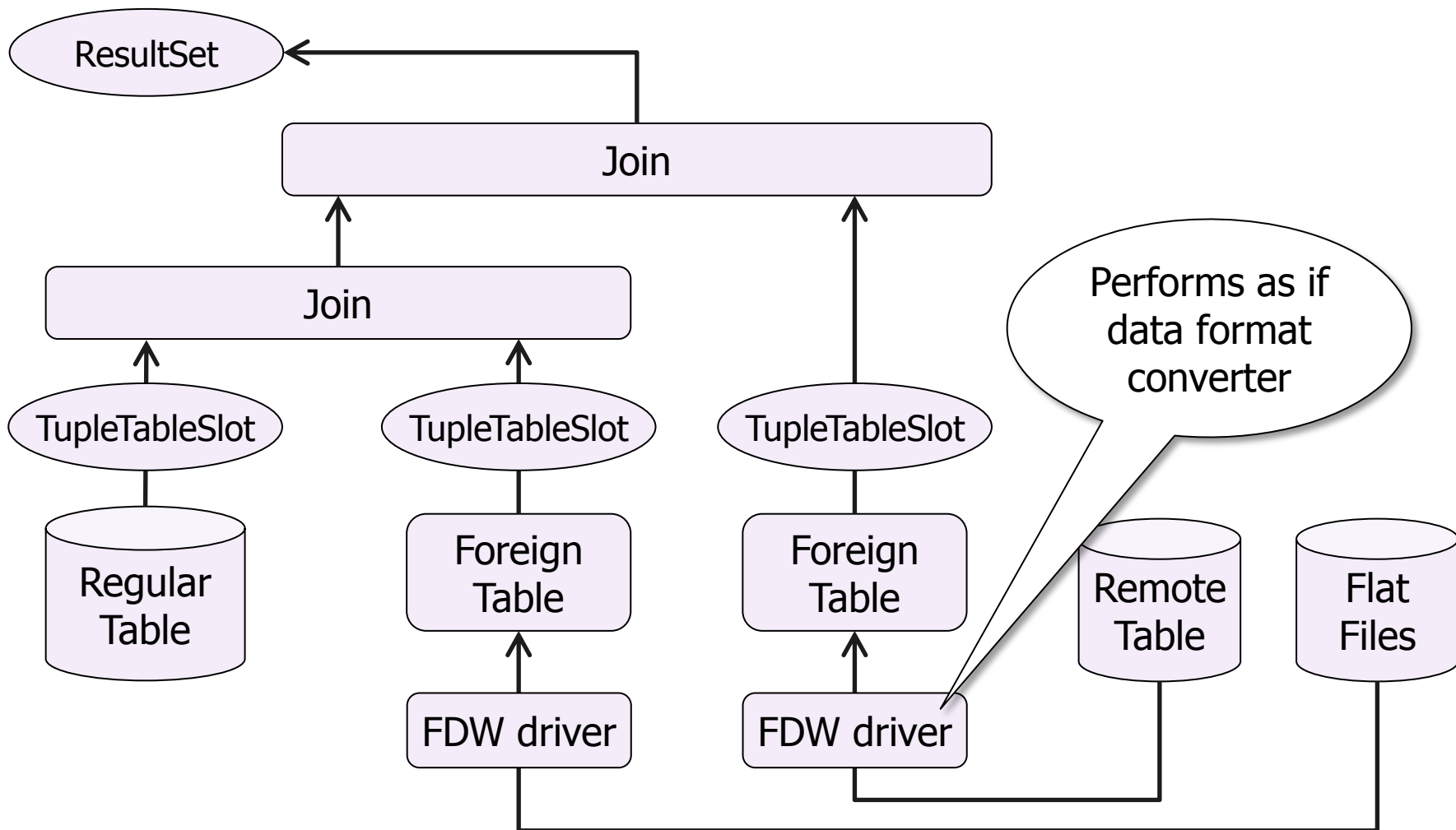
# Overview of FDW

- FDW: Foreign Data Wrapper
- External data source as if regular tables
  - `file_fdw` ... for flat files
  - `postgres_fdw` ... for remote PostgreSQL
- Extension performs as if a part of executor
- **Read-only**, right now,

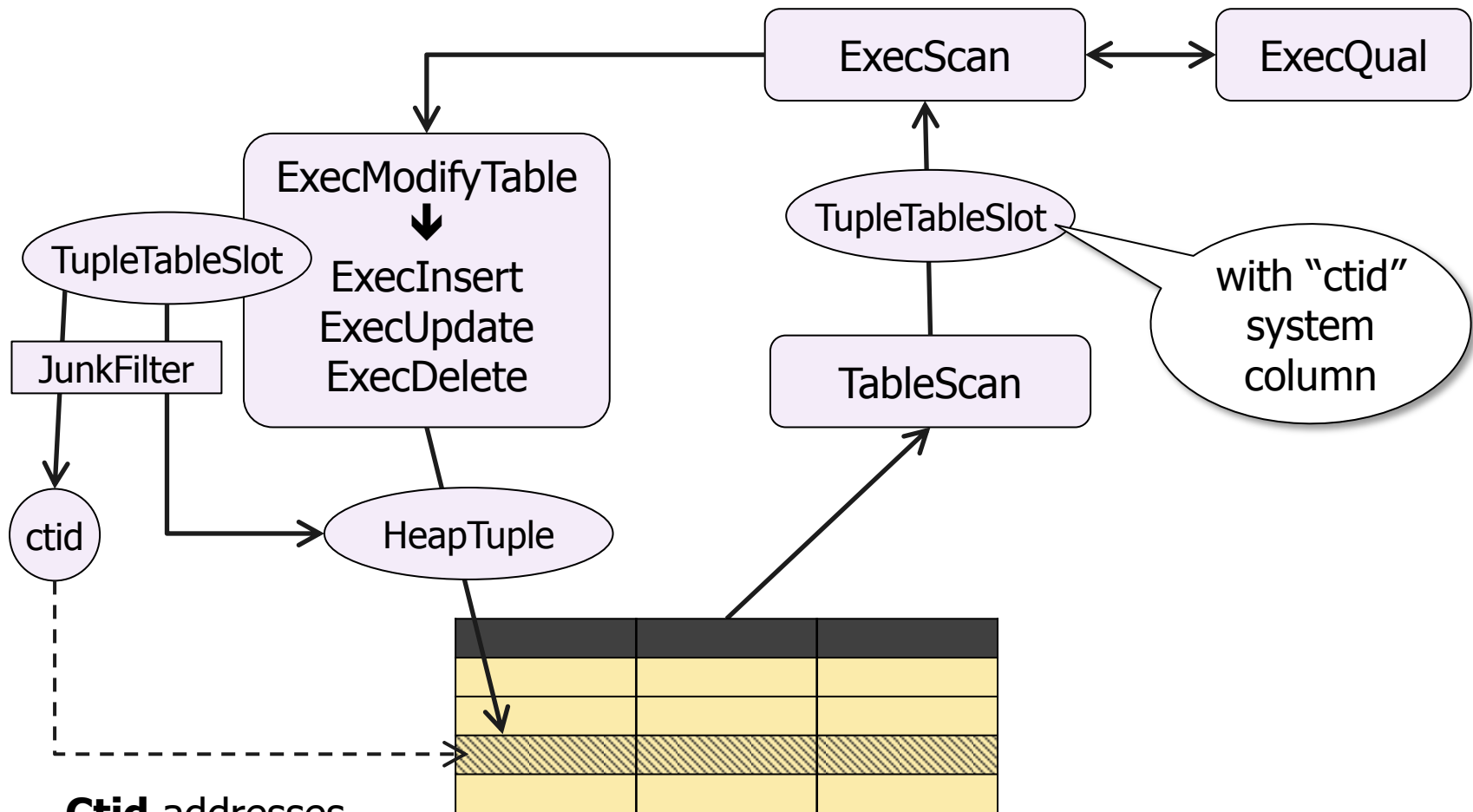
# How does it work? (1/2)



# How does it work? (2/2)



# Learn from existing logic to modify tables

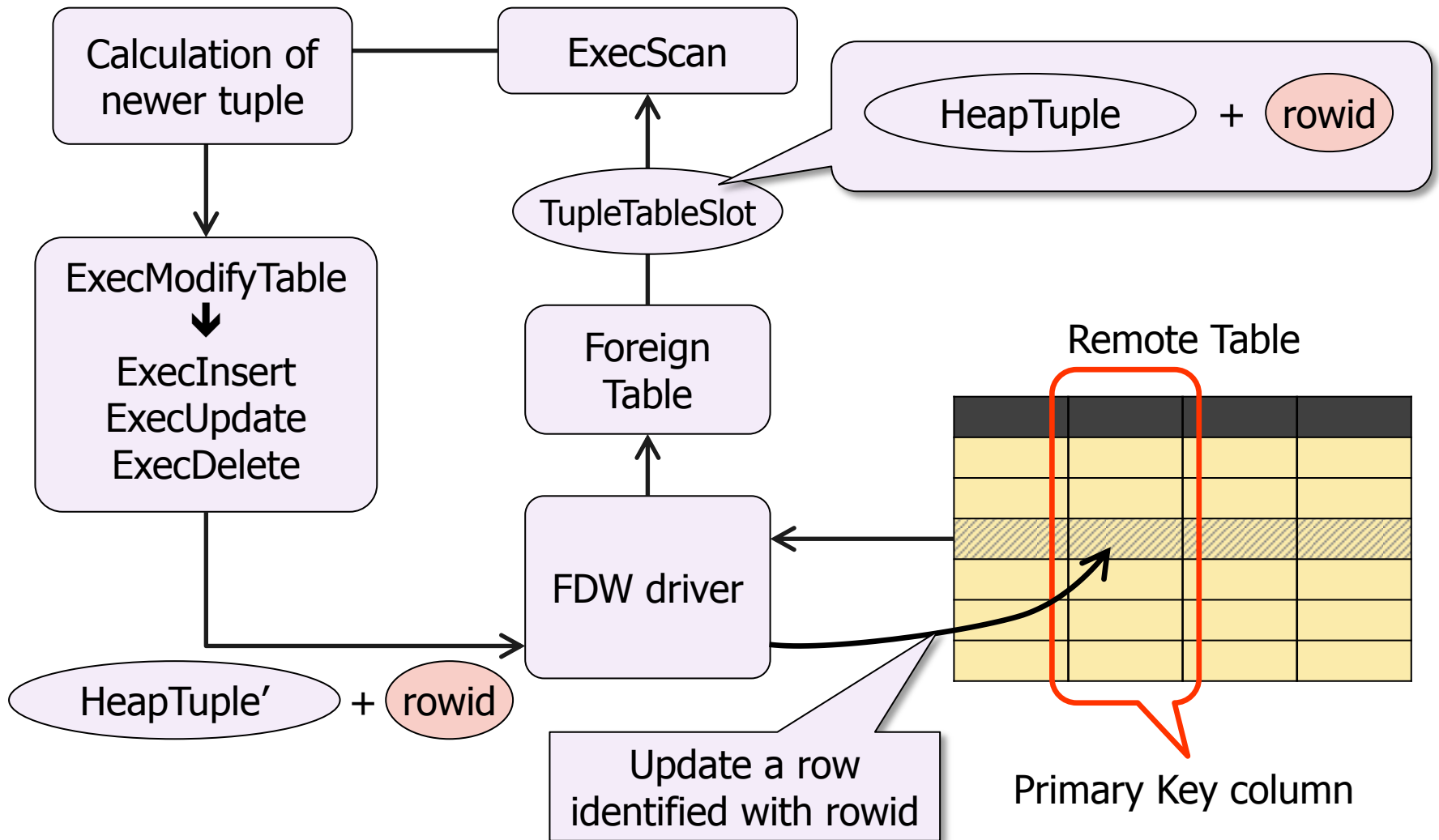


**Ctid** addresses the target row to be modified.

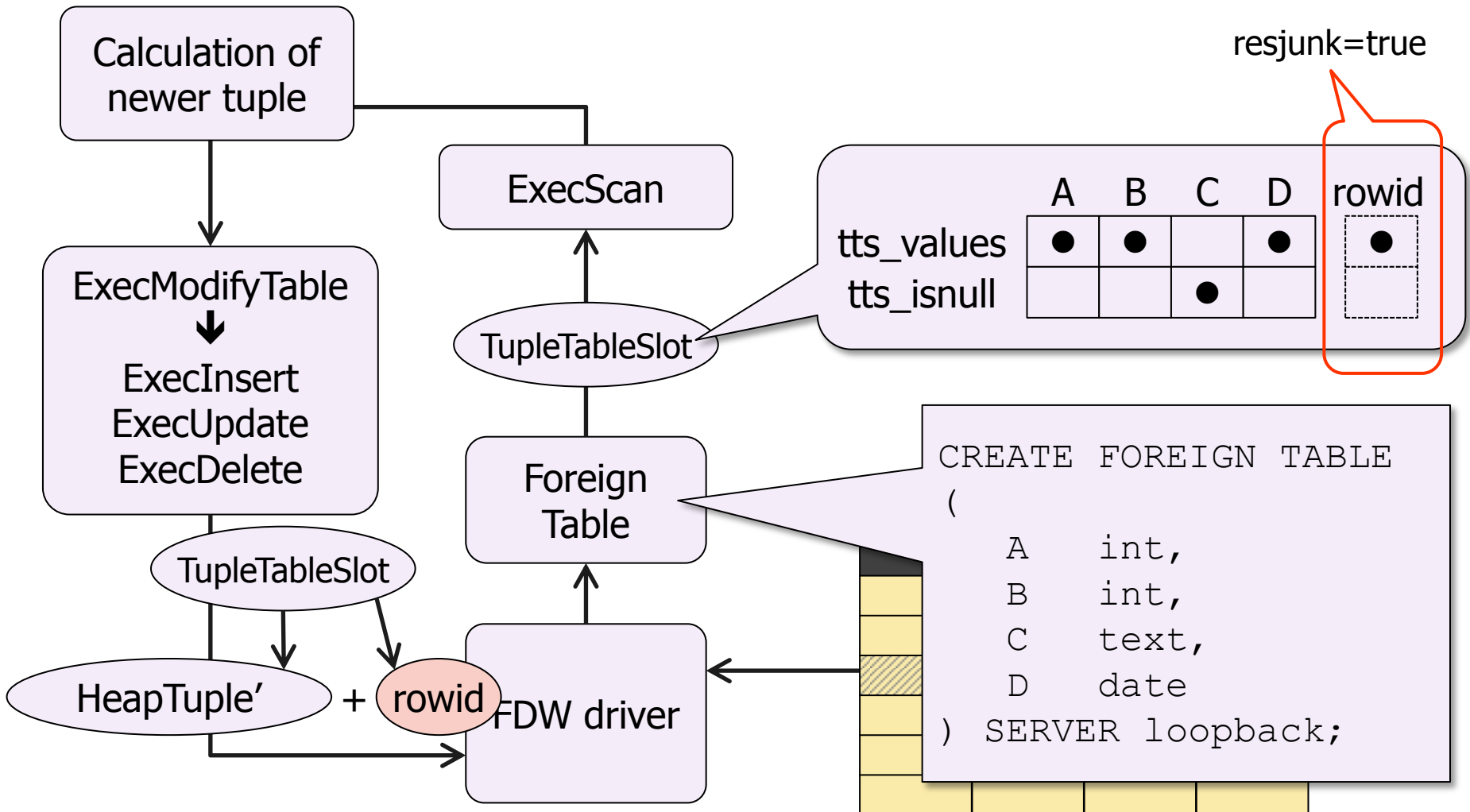
# Issues towards writable foreign table

- Way to identify a remote row to be modified
  - "rowid" to identify a particular remote row
  - "pseudo-column" to move values out of foreign table definition
- New methods relevant to ExecModifyTable
  - Methods relevant to ...
    - ExecInitModifyTable
    - ExecInsert, ExecUpdate, ExecDelete
    - ExecEndModifyTable
- Transaction control with remote data source
  - Responsibility of FDW drivers

# rowid pseudo-column (1/3)



# rowid pseudo-column (2/3)





# rowid pseudo-column (3/3)

AttrNumber

```
GetForeignRelWidth(PlannerInfo *root,  
                   RelOptInfo *baserel,  
                   Relation foreignrel,  
                   bool inhparent,  
                   List *targetList);
```

- A new GetForeignRelWidth() method
  - It returns width of TupleTableSlot, even if it overs number of columns in table definition.
  - if targetList contains "rowid" column, FDW driver has to expand the width of TupleTableSlot.
  - Extra fields are acquired to store values of pseudo-columns

# Why multiple pseudo-column is allowed?

```
SELECT X, Y, (X-Y)^2 from ftable;
```



```
SELECT X, Y, Pcol_1 from ftable;
```

Just reference to  
the calculated result  
in the remote side

```
SELECT X, Y, (X-Y)^2 AS Pcol_1  
from remote_data_source;
```

Remote Query

## • Calculation off-loading

- GetForeignRelWidth() can replace expression node of the given TargetEntry with pseudo-column reference, instead of complex calculation.
- Utilization of external computing resource
  - GPU, remote host, ...

# New APIs

```
List *PlanForeignModify(PlannerInfo *root,  
                        ModifyTable *plan,  
                        Index resultRelation,  
                        Plan *subplan);  
  
void BeginForeignModify(ModifyTableState *mtstate,  
                        ResultRelInfo *resultRelInfo,  
                        List *fdw_private,  
                        Plan *subplan,  
                        int eflags);  
  
TupleTableSlot *ExecForeignInsert(ResultRelInfo *rinfo,  
                                    TupleTableSlot *slot);  
  
TupleTableSlot *ExecForeignUpdate(ResultRelInfo *rinfo,  
                                    const char *rowid,  
                                    TupleTableSlot *slot);  
  
bool ExecForeignDelete(ResultRelInfo *rinfo,  
                        const char *rowid);  
  
void EndForeignModify(ResultRelInfo *rinfo);
```

# Working Example (1/2)

```
postgres=# CREATE FOREIGN TABLE ft1 (a int, b text, c float)
          SERVER loopback OPTIONS (relname 't1');
```

```
CREATE FOREIGN TABLE
```

```
postgres=# SELECT * FROM ft1;
```

a	b	c
1	aaa	1.1
2	bbb	2.2
3	ccc	3.3
4	ddd	4.4
5	eee	5.5

(5 rows)

```
postgres=# EXPLAIN (verbose) SELECT * FROM ft1 WHERE b like '%b%';
```

QUERY PLAN

---

Foreign Scan on public.ft1 (cost=100.00..100.01 rows=1 width=44)

Output: a, b, c

Filter: (ft1.b ~~ '%b%'::text)

Remote SQL: SELECT a, b, c FROM public.t1

(4 rows)

# Working Example (2/2)

```
postgres=# UPDATE ft1 SET b = b || '_updt'
          WHERE a % 2 = 0 RETURNING *;
```

a	b	c
2	bbb_updt	2.2
4	ddd_updt	4.4

(2 rows)

UPDATE 2

```
postgres=# EXPLAIN (verbose, costs off)
          UPDATE ft1 SET b = b || '_updt' WHERE a % 2 = 0 RETURNING *;
          QUERY PLAN
```

-----  
Update on public.ft1

Output: a, b, c

-> Foreign Scan on public.ft1

Output: a, (b || '\_updt'::text), c, ctid, ft1.\*

Remote SQL: SELECT a, b, c, ctid FROM public.t1

WHERE (((a OPERATOR(pg\_catalog.%) 2)

OPERATOR(pg\_catalog.=) 0)) FOR UPDATE

(5 rows)

# Transaction Control

```
postgres=# BEGIN;
BEGIN
postgres=# UPDATE ft1 SET b = b || '_updt' WHERE a % 2 = 1;
UPDATE 3
postgres=# ABORT;
ROLLBACK
postgres=# SELECT * FROM t1;
 a |      b      | c
---+-----+---
 2 | bbb        | 2.2
 4 | ddd        | 4.4
 1 | aaa_updt   | 1.1
 3 | ccc_updt   | 3.3
 5 | eee_updt   | 5.5
(5 rows)
```


- Transaction control is responsibility of FDW drivers
- Register(Sub)XactCallback gives extensions chance to get control on transaction events, but I didn't cover it yet.

# Does someone implement DLM?

```
postgres=# WITH ch AS (  
    UPDATE t1 SET b = 'changed' RETURNING a  
) UPDATE ft1 SET b = 'new val'  
    FROM ch WHERE ch.a = ft1.a;  
^CCancel request sent  
^CCancel request sent
```

- Right now, we have no distributed lock manager :-)
- Be careful not to acquire exclusive lock on same table
- In above example...
  - This session acquires exclusive lock on table t1.
  - table t1 is external data source of ft1 via loopback connection.
  - UPDATE on ft1 also requires exclusive lock on t1 on remote session.

# Current Status

 PostgreSQL

The world's most advanced open source database.

[Home](#) [About](#) [Download](#) [Documentation](#) [Community](#) [Developers](#) [Support](#) [Your account](#)

- » Community
- » Contributors
- » Mailing Lists
  - User lists
  - Developer lists
    - pgsql-cluster-hackers
    - pgsql-committers
    - pgsql-hackers
    - pgsql-rrreviewers
    - pgsql-www
  - Regional lists
  - Associations
  - User groups
  - Project lists
  - Inactive lists
- » IRC
- » Featured Users
- » International Sites
- » Propaganda
- » Resources

## Re: Writable Foreign Tables

**From:** Tom Lane <tgl(at)sss(dot)pgh(dot)pa(dot)us>  
**To:** Craig Ringer <craig(at)2ndQuadrant(dot)com>  
**Cc:** PostgreSQL Hackers <pgsql-hackers(at)postgresql(dot)org>  
**Subject:** Re: Writable Foreign Tables  
**Date:** 2013-01-18 03:10:21  
**Message-ID:** [23984.1358478621@sss.pgh.pa.us](mailto:23984.1358478621@sss.pgh.pa.us) (view [raw](#) or [flat](#))  
**Thread:** 2013-01-18 03:10:21 from Tom Lane <tgl(at)sss(dot)pgh(dot)pa(dot)us>  
**Lists:** [pgsql-hackers](#)

---

Craig Ringer <craig(at)2ndQuadrant(dot)com> writes:  
> The writable foreign tables patch is flagged ready for committer. While  
> its last activity was in late 2012, I haven't noticed much else changing  
> in the area that'd be likely to break it, and FDWs are a somewhat  
> immature feature anyway. It's been revised based on initial reviews.  
  
> Is anyone willing to commit this or do you want further testing?

I think that one is probably my responsibility, along with the  
postgresql FDW which is in the same general turf. My plan is to try to  
commit whatever seems immediately committable over the next few days, and  
then sit down and take a hard look at those patches to see if they're  
really ready to go in or not.

regards, tom lane



FDW, be writable!



# その他のネタ

## PG-Strom勉強会 (2/18)

GPUがPostgreSQLを加速する

Tweet 58

Like 67

+1 0

B! 2



日時: 2013/02/18 18:30 to 21:30

定員: 60 人

会場: Red Hat 株式会社 恵比寿オフィス 5Fセミナールーム (渋谷区恵比寿 4丁目1番18号)

URL: -



管理者:  [KaiGal Kohel](#)

ハッシュタグ: # [pgstrom](#)





このイベントに参加しています

### イベント管理

-  回答したアンケートを確認
-  イベントへの参加をキャンセル

### イベント管理

-  このイベントを編集する
-  このイベントを削除する

### 参加者管理

- [参加者一覧を見る](#)
- [CSVダウンロード](#)
- [UTF-8](#) [Shift JIS](#)

ご無沙汰しております。

久々に東京に戻ってくる事になりましたので、この機会に2年ぶりの勉強会を開催します。

テーマはPG-Strom。GPUを使ってPostgreSQLのDB検索性能を劇的に向上させようという試みです。

PG-Strom 勉強会

