Custom Scan API

KaiGai Kohei <kaigai@kaigai.gr.jp>
(Tw: @kkaigai)

自己紹介

- 海外 浩平 (かいがい こうへい)

➔最近、国内に戻ってきました。

- SE-PostgreSQLやPG-Stromを作っています。

Custom Scan APIとは

- Extensionがエグゼキュータ処理を乗っ取るAPI
- 俺様的方法で Scan を実装できる
- 俺様的方法で Join を実装できる
- （俺様的方法で Xxxx を実装できる）
- PostgreSQL v9.4 の標準機能化に向けて議論の途中

ExecutorXXX_hook ではダメなのか？(1/3)

```
/* Hook for plugins to get control in ExecutorStart() */
typedef void (*ExecutorStart_hook_type) (QueryDesc *queryDesc,
                                         int eflags);
extern PGDLLIMPORT ExecutorStart_hook_type ExecutorStart_hook;

/* Hook for plugins to get control in ExecutorRun() */
typedef void (*ExecutorRun_hook_type) (QueryDesc *queryDesc,
                                       ScanDirection direction,
                                       long count);
extern PGDLLIMPORT ExecutorRun_hook_type ExecutorRun_hook;

/* Hook for plugins to get control in ExecutorFinish() */
typedef void (*ExecutorFinish_hook_type) (QueryDesc *queryDesc);
extern PGDLLIMPORT ExecutorFinish_hook_type ExecutorFinish_hook;

/* Hook for plugins to get control in ExecutorEnd() */
typedef void (*ExecutorEnd_hook_type) (QueryDesc *queryDesc);
extern PGDLLIMPORT ExecutorEnd_hook_type ExecutorEnd_hook;
```

ExecutorXXX_hook ではダメなのか？(2/3)

```
postgres=# explain(costs off) select y from l_tbl join r_tbl
on a = x group by y,b order by b;
                      QUERY PLAN
------------------------------------------
 Group
   Group Key: l_tbl.b, r_tbl.y
   ->  Sort
         Sort Key: l_tbl.b, r_tbl.y
         ->  Merge Join
               Merge Cond: (l_tbl.a = r_tbl.x)
               ->  Sort
                     Sort Key: l_tbl.a
                     ->  Seq Scan on l_tbl
               ->  Materialize
                     ->  Sort
                           Sort Key: r_tbl.x
                           ->  Seq Scan on r_tbl
(13 rows)
```

独自のソート実装をエクステンションとして実装できるか？

ExecutorXXX_hook ではダメなのか？(3/3)

```
TupleTableSlot *ExecProcNode(PlanState *node)
{
        :
    switch (nodeTag(node))
    {
        :
        case T_SeqScanState:
            result = ExecSeqScan((SeqScanState *) node);
            break;
        :
        default:
            elog(ERROR, "unrecognized node type: %d",
                 (int) nodeTag(node));
            result = NULL;
            break;
    }
    return result;
}
```

何か『任意の処理を行う』ノードを PostgreSQL本体が認識できる 必要がある。

Custom Scan API (1/3)

- **オプティマイザへ介入するためのフック**

```
/* Hook for plugins to add custom scan path */
typedef void (*add_scan_path_hook_type)(PlannerInfo *root,
                                        RelOptInfo *baserel,
                                        RangeTblEntry *rte);
extern PGDLLIMPORT add_scan_path_hook_type add_scan_path_hook;

/* Hook for plugins to add custom join path */
typedef void (*add_join_path_hook_type)(PlannerInfo *root,
                                        RelOptInfo *joinrel,
                                        RelOptInfo *outerrel,
                                        RelOptInfo *innerrel,
                                        JoinType jointype,
                                        SpecialJoinInfo *sjinfo,
                                        List *restrictlist,
                                        List *mergeclause_list,
                                        SemiAntiJoinFactors *semafac,
                                        Relids param_source_rels,
                                        Relids extra_lateral_rels);
extern PGDLLIMPORT add_join_path_hook_type add_join_path_hook;
```

Custom Scan API (2/3)

- エグゼキュータに介入するためのコールバック#1

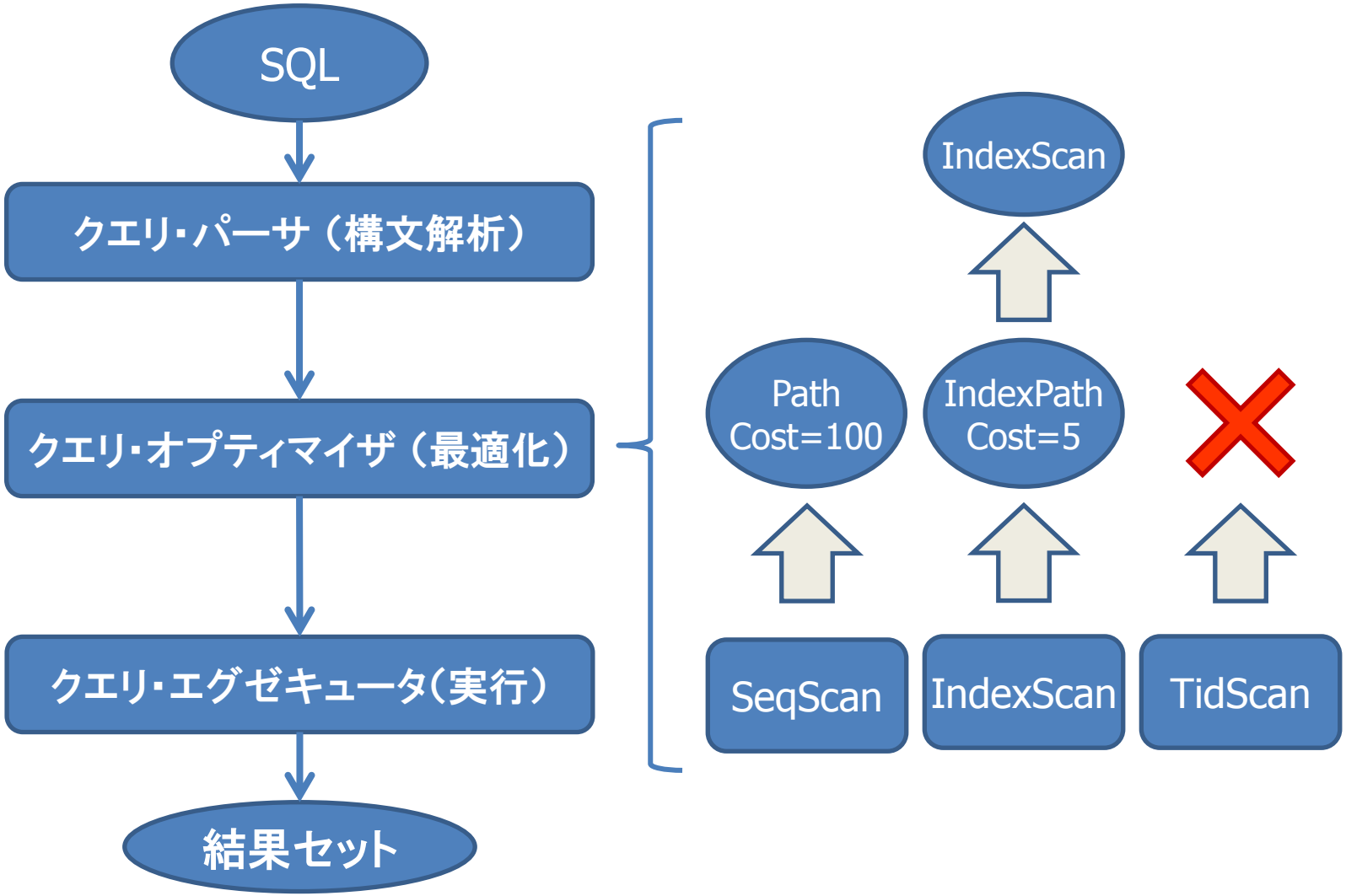
```
typedef void (*InitCustomScanPlan_function)
                    (PlannerInfo *root,
                     CustomScan *cscan_plan,
                     CustomPath *cscan_path,
                     List *tlist,
                     List *scan_clauses);
typedef void (*SetPlanRefCustomScan_function)
                   (PlannerInfo *root,
                    CustomScan *cscan_plan,
                    int rtoffset);
typedef void (*BeginCustomScan_function)
                   (CustomScanState *csstate, int eflags);
typedef TupleTableSlot *
    (*ExecCustomScan_function)(CustomScanState *csstate);
```

Custom Scan API (3/3)

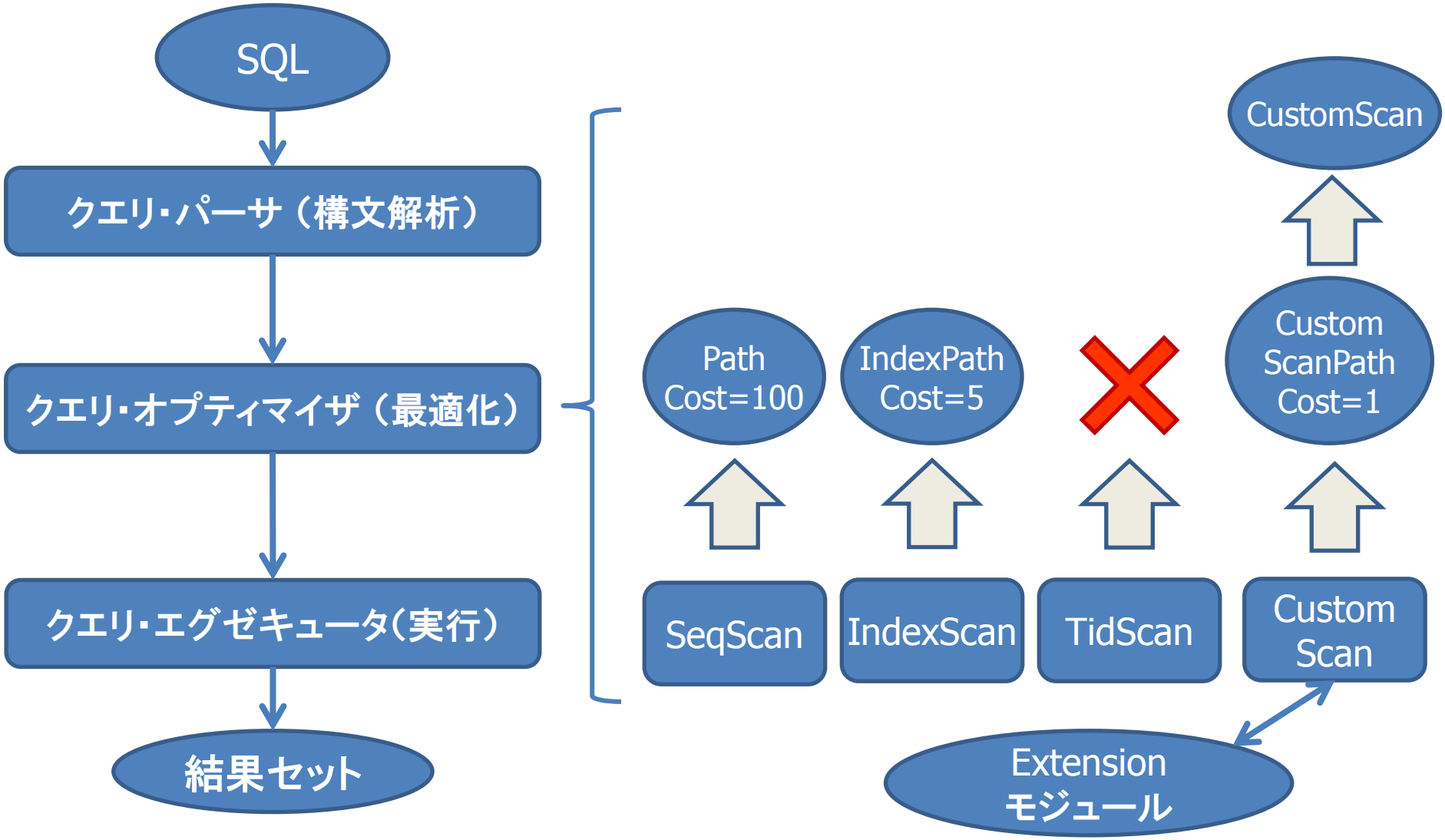
- エグゼキュータに介入するためのコールバック#2

```
typedef Node *(*MultiExecCustomScan_function)
                    (CustomScanState *csstate);
typedef void (*EndCustomScan_function)
                    (CustomScanState *csstate);


typedef void (*ReScanCustomScan_function)
                    (CustomScanState *csstate);
typedef void (*MarkPosCustomScan_function)
                    (CustomScanState *csstate);
typedef void (*RestorePosCustom_function)
                    (CustomScanState *csstate);


typedef void (*ExplainCustomScan_function)
                    (CustomScanState *csstate,
                     ExplainState *es);
```

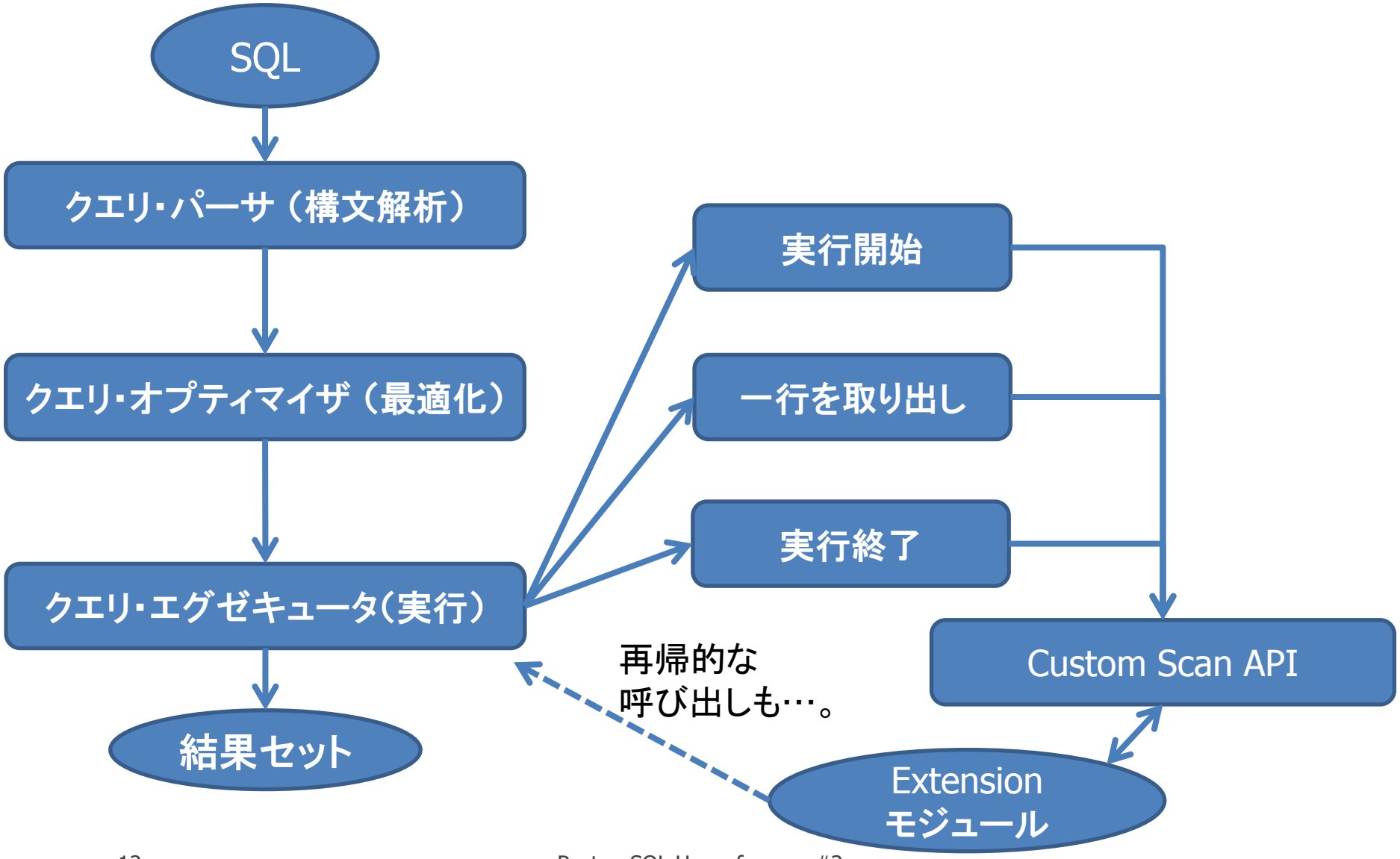
PostgreSQLクエリ処理の流れ (1/2)



PostgreSQLクエリ処理の流れ (2/3)



PostgreSQLクエリ処理の流れ (3/3)

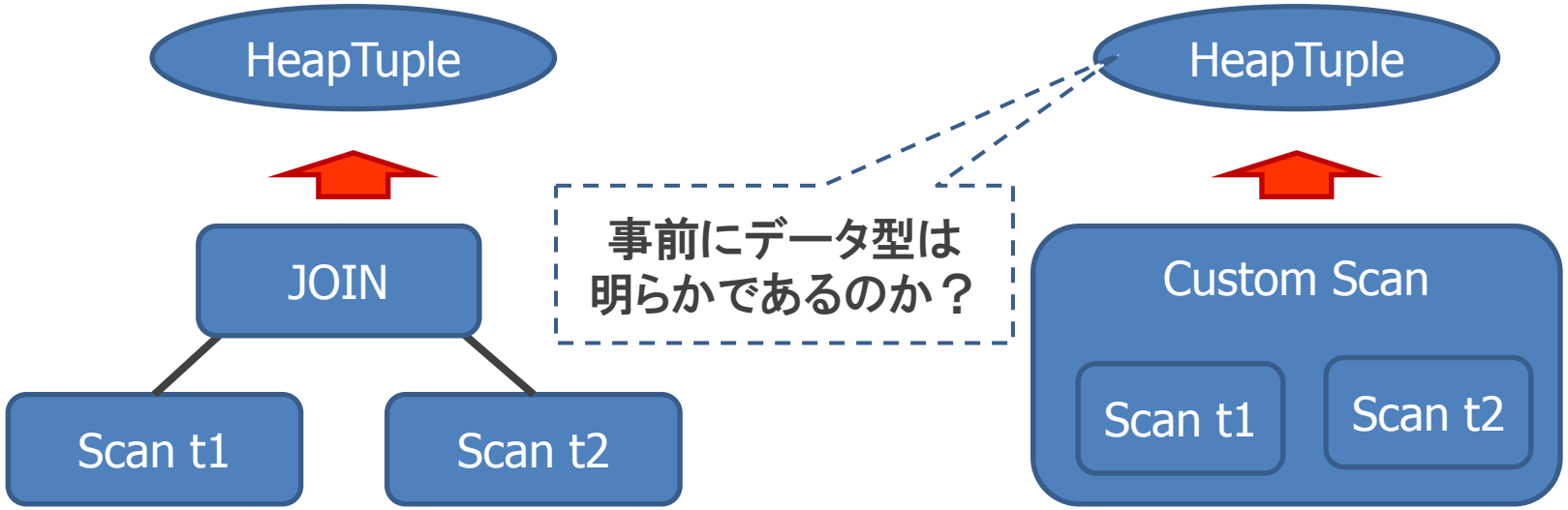


FDWとの違い

- FDW / Foreign Table
  - 参照や更新の対象として使用できる。
  - "一行を返す"時に、データ型は常にテーブル定義と一致していなければならない。
  - オブジェクトを実装する役割
- Custom Scan API
  - 参照や更新の対象として使用できない
  - "一行を返す" 時に、データ型を任意に定める事ができる。（上位ノードの期待通りである事はモジュールの責任）
  - メソッドを実装する役割

データ型が変動するとは？

```
SELECT * FROM t1 JOIN t2 ON t1.a = t2.x;
```



- テーブルスキャンを実装する場合、返却されるタプルの型は事前に明らか
  ➔CREATE TABLEで指定したデータ型
- JOINを実装する場合、返却されるタプルの型は毎回変わる
  ➔JOINされるテーブルの定義による

JoinをCustomScan APIで置き換えた例

- Postgres_fdw への機能拡張
  - Foreign table 同士のJOINで、
  - 同一のForeign serverにホストされており、
  - 結合条件がリモート実行可能なものである場合

```
postgres=# explain (costs off, verbose)
           select * from ft1 where b like '%aaa%';
              QUERY PLAN
-----------------------------------------------------------
Foreign Scan on public.ft1
   Output: a, b
   Remote SQL: SELECT a, b FROM public.t1
               WHERE ((b ~~ '%aaa%'::text))
(3 rows)
```

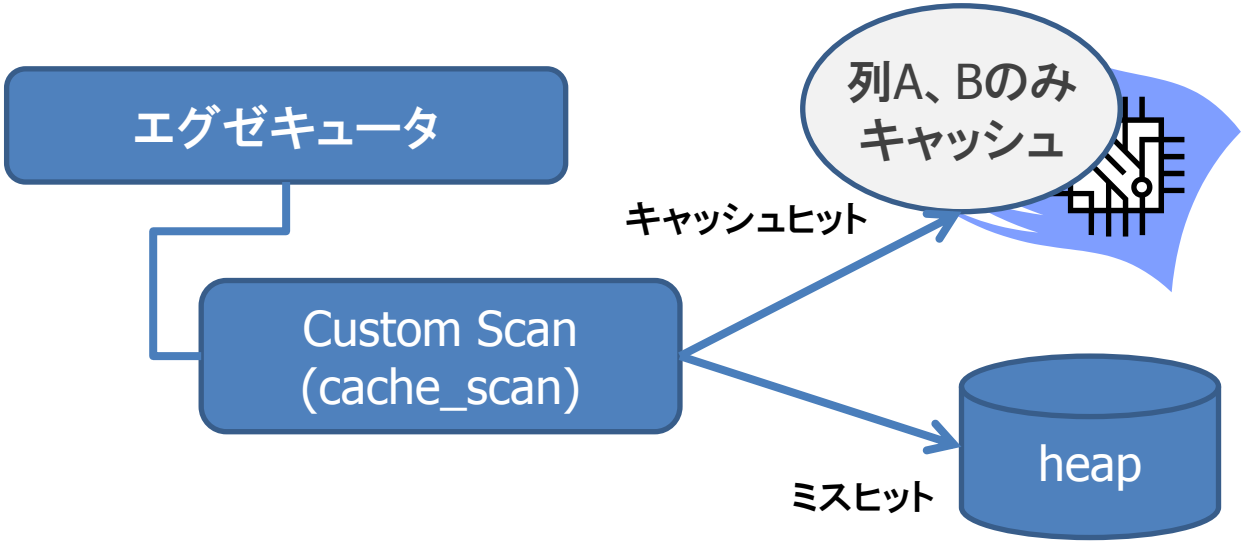
JoinをCustomScan APIで置き換えた例

- Postgres_fdw への機能拡張
  - Foreign table 同士のJOINで、
  - 同一のForeign serverにホストされており、
  - 結合条件がリモート実行可能なものである場合

```
postgres=# explain (costs off, verbose)
           select * from ft1 join ft2 on a = x
           where b like '%aaa%';
                        QUERY PLAN
-----------------------------------------------------------
 Custom Scan (postgres-fdw)
   Output: a, b, x, y
   Remote SQL: SELECT r1.a, r1.b, r2.x, r2.y FROM
    (public.t1 r1 JOIN public.t2 r2 ON ((r1.a = r2.x)))
    WHERE ((r1.b ~~ '%aaa%'::text))
(3 rows)
```

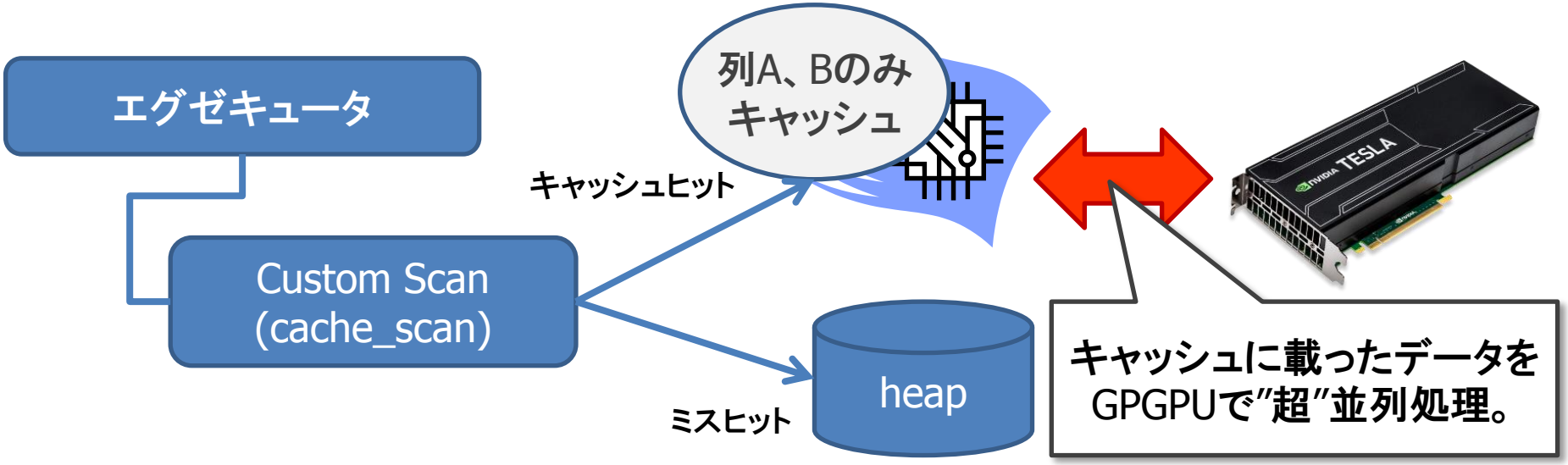
その他の利用例 (1/3) – Cache-only Scan

```
postgres=# EXPLAIN(costs off)
           SELECT a,b FROM t1 WHERE a > b;
              QUERY PLAN
-------------------------------------
 Custom Scan (cache scan) on t1
   Filter: ((a)::double precision > b)
(2 rows)
```



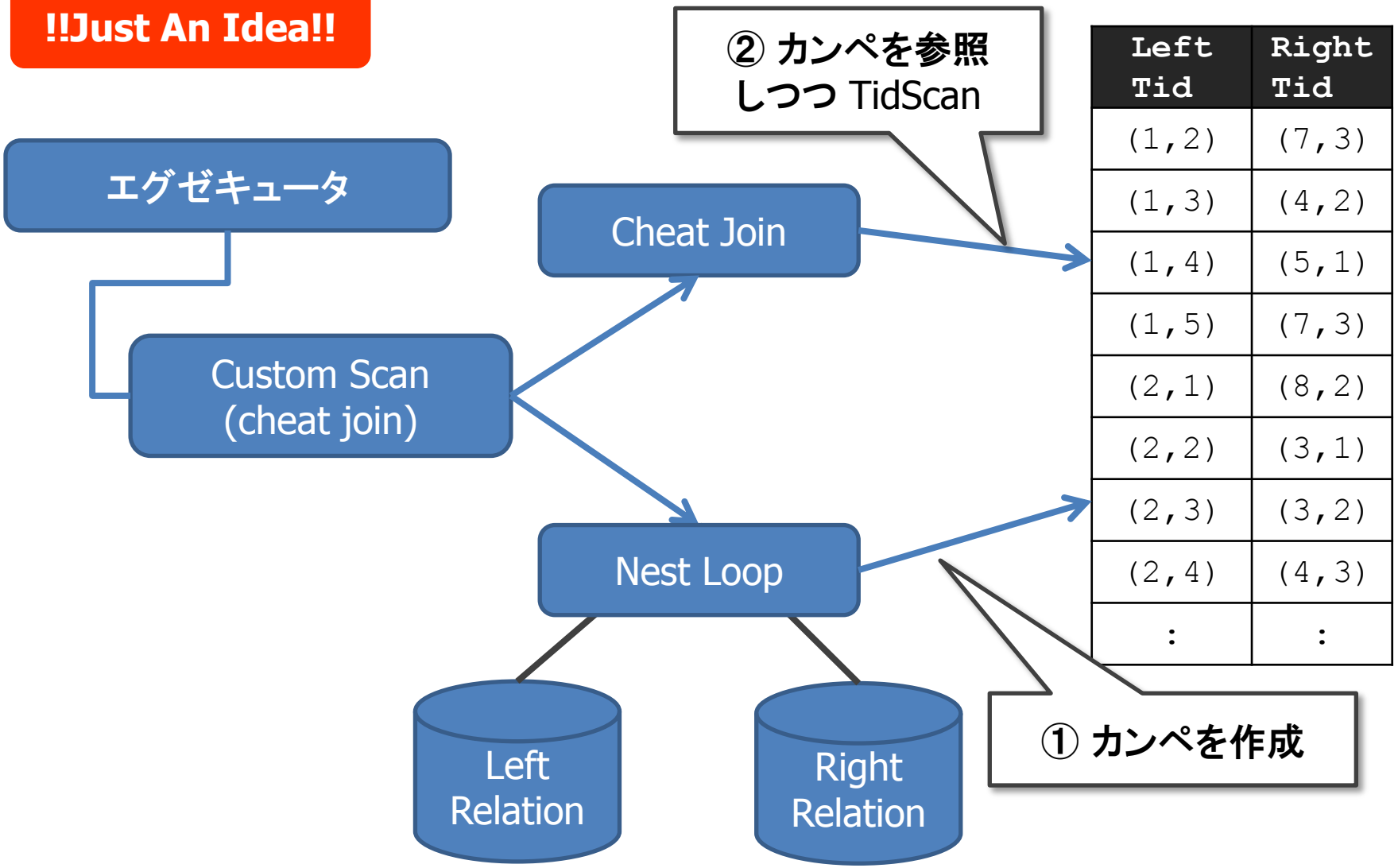
その他の利用例 (2/3) – Cache-only Scan

```
postgres=# EXPLAIN(costs off)
           SELECT a,b FROM t1 WHERE a > b;
              QUERY PLAN
-------------------------------------
 Custom Scan (cache scan) on t1
   Filter: ((a)::double precision > b)
(2 rows)
```



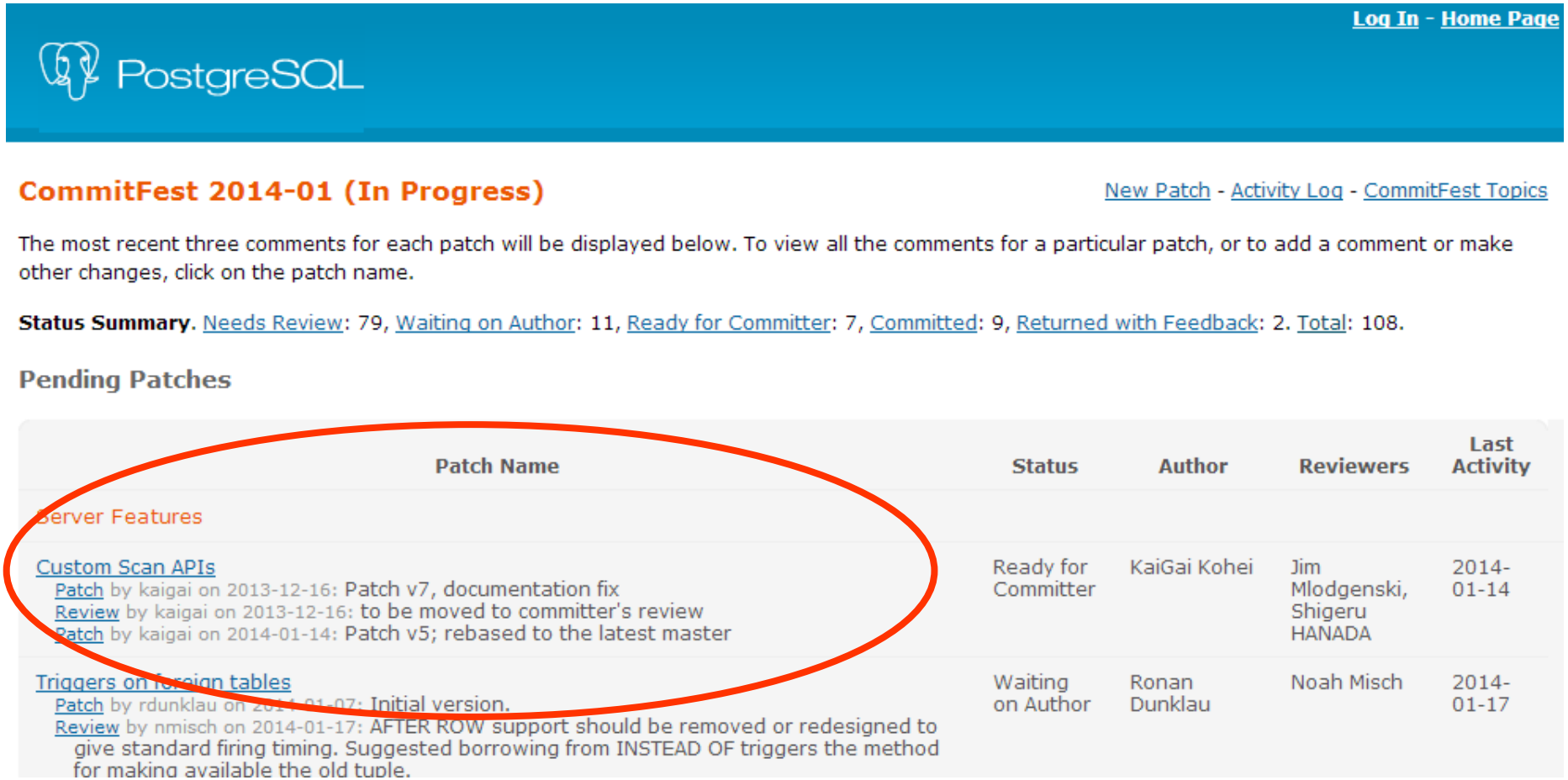
その他の利用例 (3/3) – Cheat Join

!!Just An Idea!!



Left Tid	Right Tid
(1,2)	(7,3)
(1,3)	(4,2)
(1,4)	(5,1)
(1,5)	(7,3)
(2,1)	(8,2)
(2,2)	(3,1)
(2,3)	(3,2)
(2,4)	(4,3)
:	:

Call for your feedback!



Log In - Home Page

PostgreSQL

CommitFest 2014-01 (In Progress)

New Patch - Activity Log - CommitFest Topics

The most recent three comments for each patch will be displayed below. To view all the comments for a particular patch, or to add a comment or make other changes, click on the patch name.

Status Summary. Needs Review: 79, Waiting on Author: 11, Ready for Committer: 7, Committed: 9, Returned with Feedback: 2. Total: 108.

Pending Patches

Patch Name	Status	Author	Reviewers	Last Activity
Server Features				
Custom Scan APIs Patch by kaigai on 2013-12-16: Patch v7, documentation fix Review by kaigai on 2013-12-16: to be moved to committer's review Patch by kaigai on 2014-01-14: Patch v5; rebased to the latest master	Ready for Committer	KaiGai Kohei	Jim Mlodgenski, Shigeru HANADA	2014-01-14
Triggers on foreign tables Patch by rdunklau on 2014-01-07: Initial version. Review by nmisch on 2014-01-17: AFTER ROW support should be removed or redesigned to give standard firing timing. Suggested borrowing from INSTEAD OF triggers the method for making available the old tuple.	Waiting on Author	Ronan Dunklau	Noah Misch	2014-01-17