EnterpriseDB®
The Enterprise Postgres Company

- CREATE FOREIGN DATA WRAPPER
- CREATE SERVER
- CREATE USER MAPPING
- **CREATE FOREIGN TABLE**

CREATE FOREIGN TABLE

EnterpriseDB®
The Enterprise Postgres Company

Syntax:

```
CREATE FOREIGN TABLE <table name>
  [ <left paren> <basic column definition list> <right paren> ]
  SERVER <foreign server name>
[ OPTIONS <left paren> <generic option list> <right paren>
<generic option list> ::=
  <generic option> [ { <comma> <generic option> }... ]
<generic option> ::=
  <option name> [ <option value> ]
<option value> ::=
  <character string literal>
```

Examples:

```
CREATE FOREIGN TABLE remotetable SERVER remoteserver;
CREATE FOREIGN TABLE remotetable (id integer, foo text) SERVER remoteserver;
CREATE FOREIGN TABLE remotetable (id integer) SERVER srv1 OPTIONS ( myoption = 'value' )
```

EnterpriseDB implementation

EnterpriseDB®
The Enterprise Postgres Company

- Foreign Data Wrappers are contrib modules, implementing an API to connect, run queries, and return results
- Three built-in FDWs:
  - Libpq
  - OCI (for Oracle)
  - ODBC (work-in-progress)

```
edb=# CREATE FOREIGN TABLE foreigntable SERVER
foolink OPTIONS  (table=remotetable);
CREATE FOREIGN TABLE
edb=# SELECT * FROM foreigntable;
 id |  data
----+----------
  1 | it works!
(1 row)
```

```
edb=# EXPLAIN SELECT * FROM foreigntable WHERE id = 1;
                                 QUERY PLAN

-------------------------------------------------------------------------------
-----------------------------------------------
 Remote Scan on remotetable foreigntable (SELECT id, data FROM
remotetable WHERE
 (id = 1))  (cost=0.00..110.00 rows=1000 width=0)
(1 row)
```

```
edb=# EXPLAIN SELECT * FROM foreigntable a INNER JOIN foreigntable2 b ON a.id =
b.id;
                                QUERY PLAN


-------------------------------------------------------------------------------

---------------------------------
 Merge Join  (cost=319.66..399.66 rows=5000 width=0)
   Merge Cond: (a.id = b.id)
   ->  Sort  (cost=159.83..162.33 rows=1000 width=0)
         Sort Key: a.id
         ->  Remote Scan on remotetable a (SELECT id, data FROM remotetable)
(cost=0.00..110.00 rows=1000 width=0)
   ->  Sort  (cost=159.83..162.33 rows=1000 width=0)
         Sort Key: b.id
         ->  Remote Scan on remotetable b (SELECT id, data FROM remotetable)
(cost=0.00..110.00 rows=1000 width=0)
(8 rows)
```

Unsolved problems

EnterpriseDB®
The Enterprise Postgres Company

- How to decide what can be executed remotely?
  - Routine mappings
- How to construct the query to be sent to remote server?
  - ruleutils.c only works for same version of PostgreSQL

Proposed FDW planner API

EnterpriseDB®
The Enterprise Postgres Company

Node *plan_remotely(Node *orig);

Where orig can be

- Raw parse tree (Query *)
- RelOptInfo
- Expr tree

- Returns a Plan/Expr tree that executes the given input query/expression
  - Using ForeignScan/Expr structs which are opaque to the planner (except for cost fields)
- Or NULL if it doesn't understand or can't handle the given input tree