Pivotal

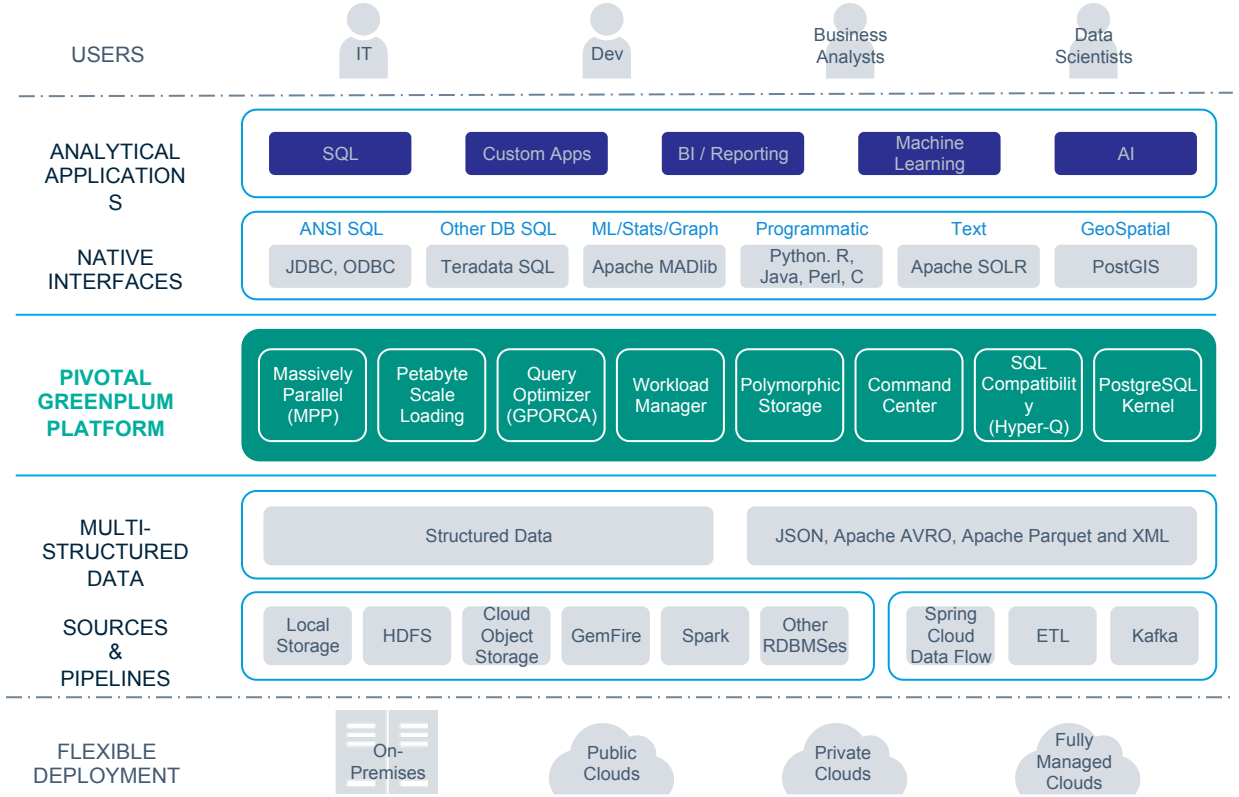
Petabyte Scale Data Warehousing Greenplum

Integrating GP with Other Data Sources

Postgres Conf 2018

Marshall Presser
Craig Sylvester
Andreas Scherbaum
17 April 2018

Data Platform for Analytics

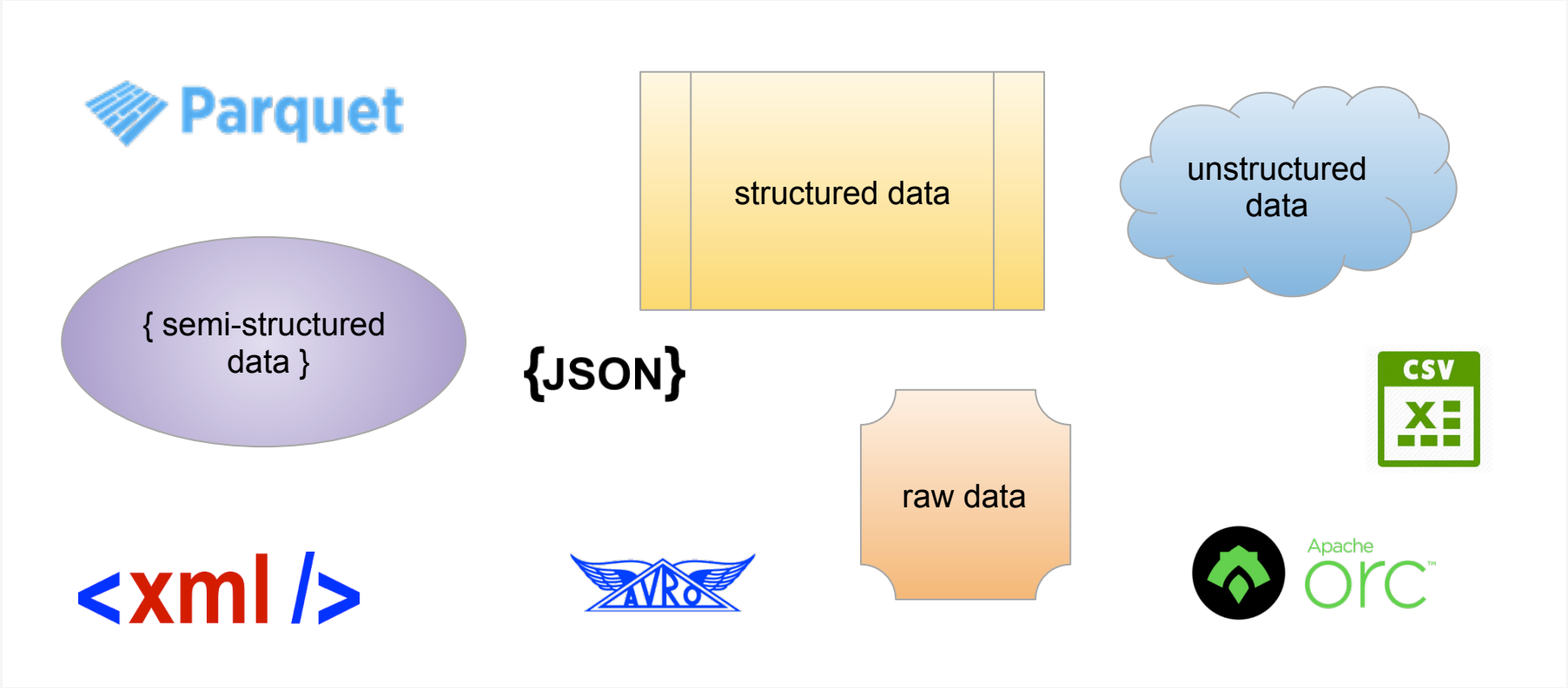


Pivotal Greenplum®

- The world's first open-source massively parallel processing (MPP) data platform for advanced analytics
- Based on PostgreSQL
- Developed since early 2000s
- Open sourced in 2015
- SQL 2003 compliant
- Advanced cost-based optimizer
- ACID transactions guarantees

Pivotal

Modern Enterprise : heterogeneous data formats



Modern Enterprise : wide variety of data engines

RDBMS

Greenplum External Table

Provides the definitions for:

- the ***schema*** of the external data
- the ***protocol*** used to access the data
- the ***location*** of the data in an external system
- the ***format*** of the external data

Participates in query execution and allows plug-in connectors to external data for different protocols.

```
CREATE [READABLE] EXTERNAL TABLE table_name
( col_name data_type [,...] | LIKE other_table )
LOCATION ('<protocol>://<path to data>...)
FORMAT 'TEXT'



CREATE WRITABLE EXTERNAL TABLE table_name
( col_name data_type [,...] | LIKE other_table )
LOCATION ('<protocol>://<path to data>...)
FORMAT 'CUSTOM'
  (Formatter=<formatter_specifications>)
[ ENCODING 'encoding' ]



CREATE [READABLE] EXTERNAL WEB TABLE
table_name ...

CREATE WRITABLE EXTERNAL WEB TABLE table_name
...
```

External Protocol Handler

- Provides ***connectivity*** to an external system
- Implements methods to ***read data*** from the external system and ***write data*** into it
- Defines the ***validation logic*** for external table specifications
- Can be packaged as a ***shared library*** file (`.so`) and loaded dynamically

```
                AVAILABLE PROTOCOLS

file://    -- for files on Greenplum segments

gpfdist:// -- for files on remote hosts

s3://      -- for files in AWS S3 bucket

gphdfs://  -- for files in Hadoop HDFS

http://    -- for WEB tables


pxf://     -- for data sources with JAVA APIs :

              • files in Hadoop HDFS
              • data in Apache Hive  tables
              • data in Apache HBase tables
              • rows in RDBMS tables via JDBC
              • objects in in-memory grids
              • messages in queues
              • ... build your own adapter ...
```

Platform Extension Framework (PXF)

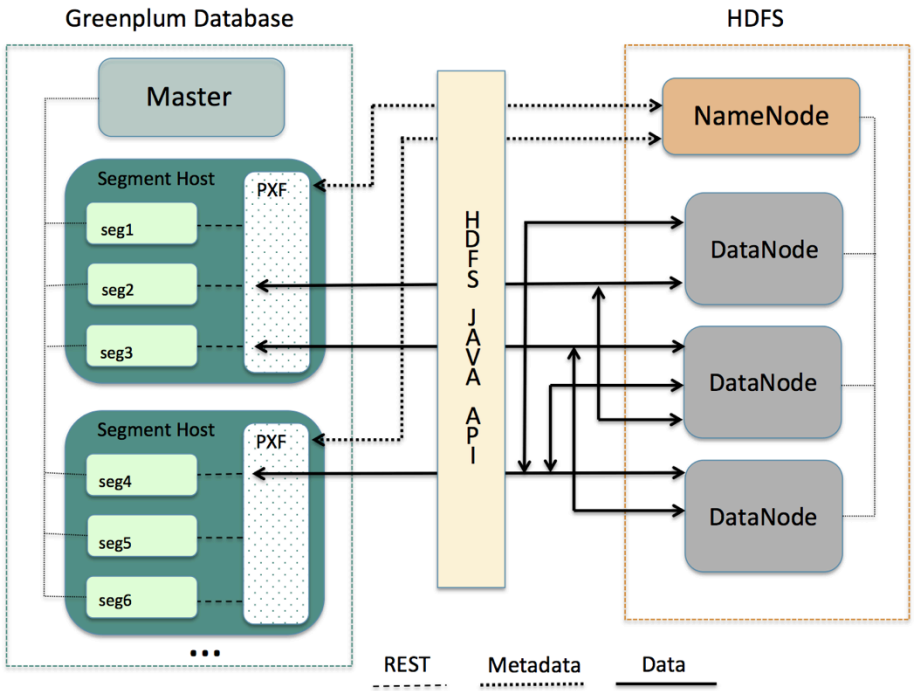
The Platform Extension Framework (PXF) provides:

- ❖ parallel, high throughput data access
- ❖ federated queries across heterogeneous data sources
- ❖ built-in connectors that map a Greenplum Database external table definition to an external data source.

Available in  since 2017 (5.1 release)

- **Apache HAWQ (incubating) launched in 2012 and was open-sourced in 2015**
- **Based on Greenplum and modified to work natively with data stored in HDFS**
- **PXF is used to connect to data in Hadoop ecosystem**
- **PXF is open-sourced under Apache license**

PXF > HDFS Data Import Flow



1. Master submits a query and segments start parallel execution

2. Each segment query execution slice gets a thread in PXF JVM

3. PXF asks HDFS Namenode for the information on file fragments

4. PXF decides on a workload distribution among threads

5. PXF reads data fragments via HDFS APIs from Datanodes and passes it to segments

6. Segments convert data into

PXF Fragmenter

Functional interface which

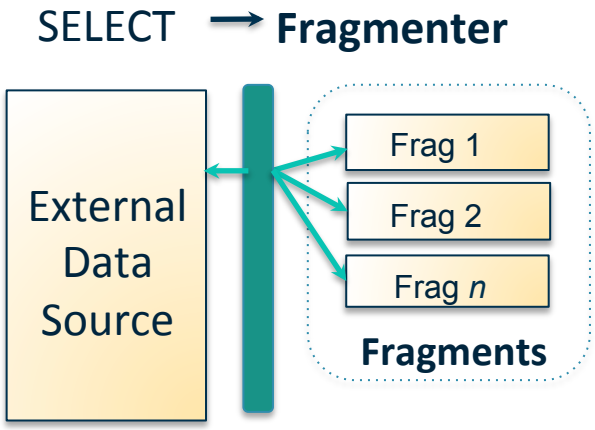
splits data from an external data source

into a **list of independent fragments**

that can be read in parallel.

Examples of fragments:

- FileSplit in HDFS
- Table partition in JDBC



PXF Accessor

Functional interface which

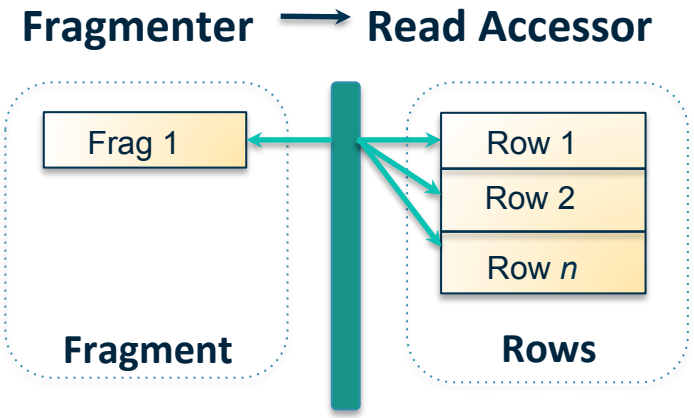
reads a **single fragment**

from an external data source and

produces a **list of records/rows**.

Examples of a record:

- Line in a text file
- Row in a JDBC ResultSet



PXF Resolver

Functional interface which

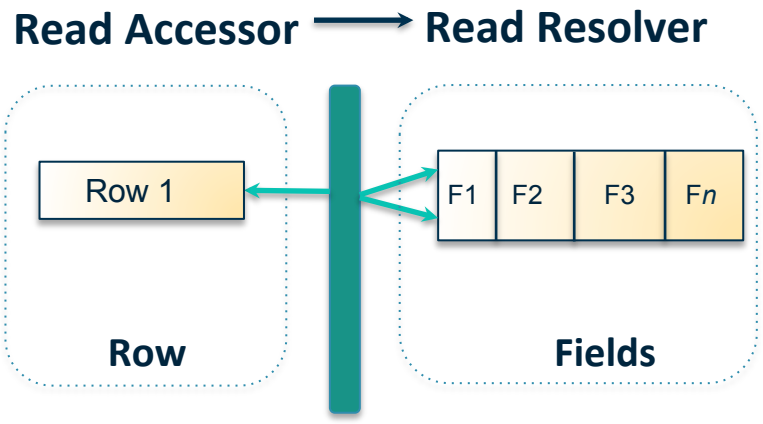
deserializes a record/row into **fields** and

transforms the data types

into those supported by Greenplum

Examples of a field:

- Value between commas in a CSV line
- Column value in a JDBC ResultSet



PXF Profile

A profile is a simple **name mapping** to

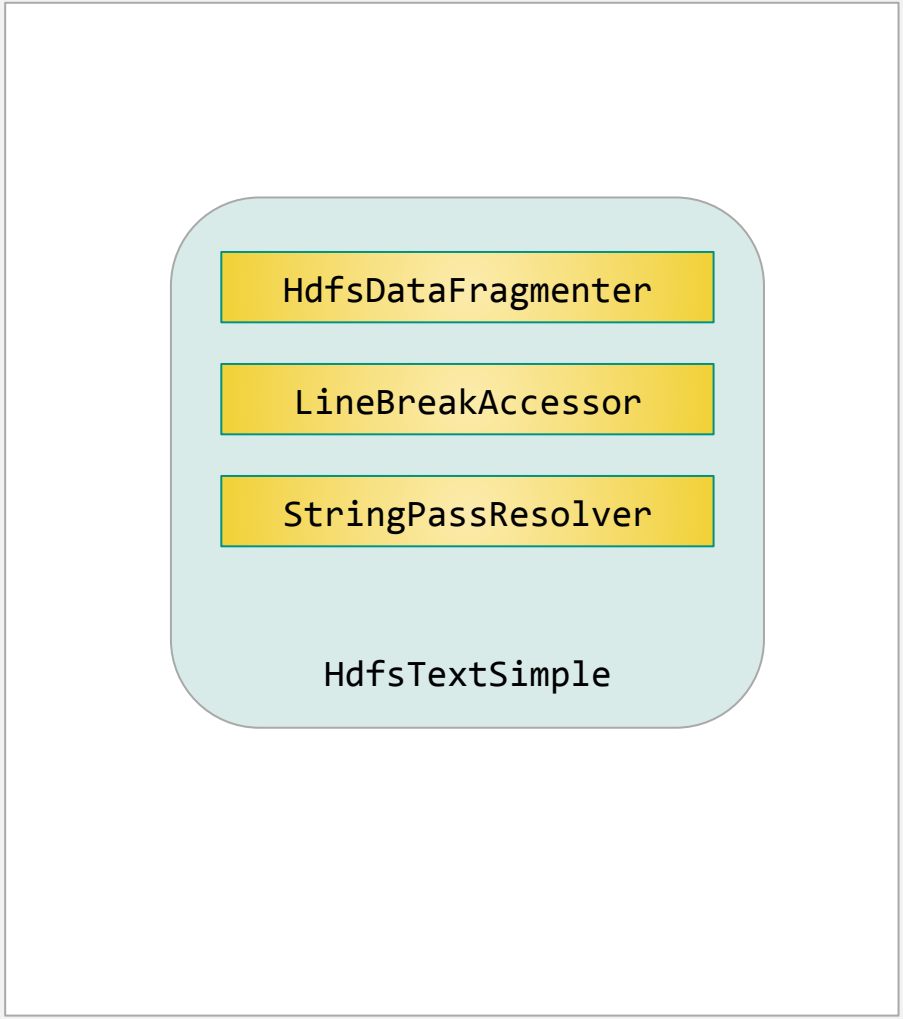
a set of connector plug-in class names

Implementing

Fragmenter, Accessor and Resolver

functional interfaces.

Profiles are useful when defining PXF external tables in Greenplum

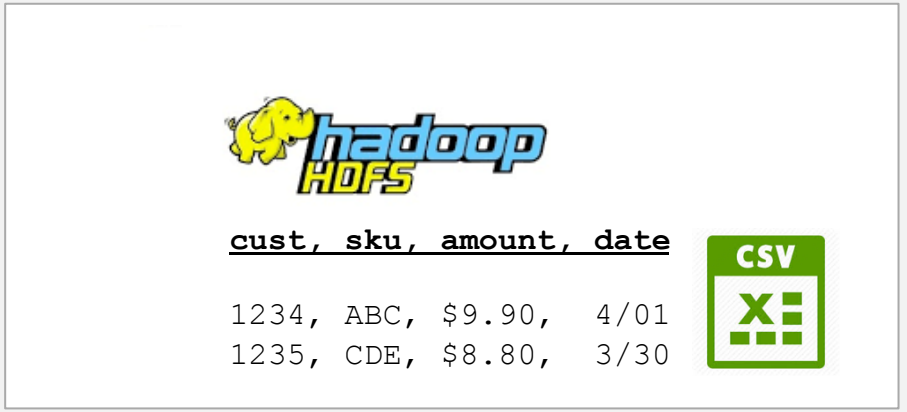


PXF External Table

Register PXF Greenplum extension

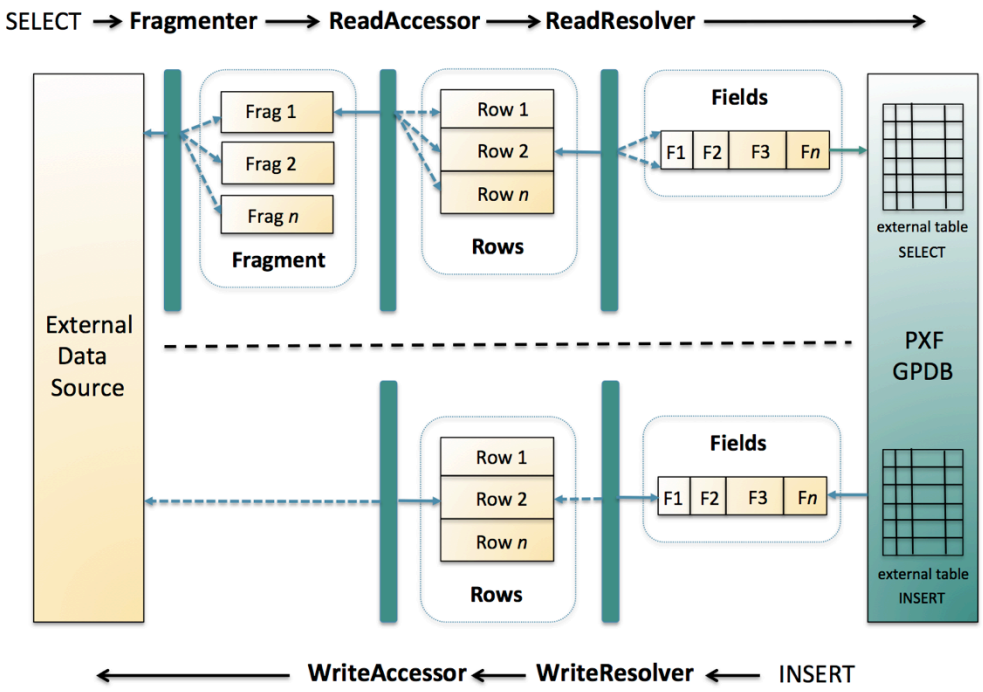
Define an external table with:

- the schema that corresponds to the structure of external data
- the protocol `pxf://` and the location of the data on external system
- the profile to use for accessing the data
- the format of data returned by PXF




```
-- create extension only once per database
CREATE EXTENSION pxf;

-- define external table

CREATE EXTERNAL TABLE sales

(cust int, sku text, amount decimal, date date)

LOCATION

('pxf:///2018/sales.csv?PROFILE=HdfsTextSimple')

FORMAT 'TEXT'
```

PXF > Data Flows Summary



Fragmenter, Accessor and Resolver working in combination

They can be specified as a pre-built profile or independently

Greenplum external table defines data schema, location, format and the profile to use to get the data

PXF can read the data from the external system or write to it

Export (write) Flow does not require a fragmenter since the fragmentation happens on Greenplum side

PXF > HDFS Connector

Data Format	Profile Name	Description
Text	`HdfsTextSimple` `HdfsTextMulti`	Read delimited single or multi-line records from plain text data on HDFS.
Parquet	`Parquet`	Read Parquet format data (`<filename>.parq`).
Avro	`Avro`	Read Avro format binary data (`<filename>.avro`).
JSON	`JSON`	Read JSON format data (`<filename>.json`).

{JSON}

PXF > Hive Connector

File Format	Profile Name	Description
TextFile	`Hive, HiveText`	Flat file with data in comma-, tab-, or space-separated value format or JSON notation.
SequenceFile	`Hive`	Flat file consisting of binary key/value pairs.
RCFile	`Hive, HiveRC`	Record columnar data consisting of binary key/value pairs; high row compression rate.
ORC	`Hive, HiveORC, HiveVectorizedORC`	Optimized row columnar data with stripe, footer, and postscript sections; reduces data size.
Parquet	`Hive`	Compressed columnar data representation.

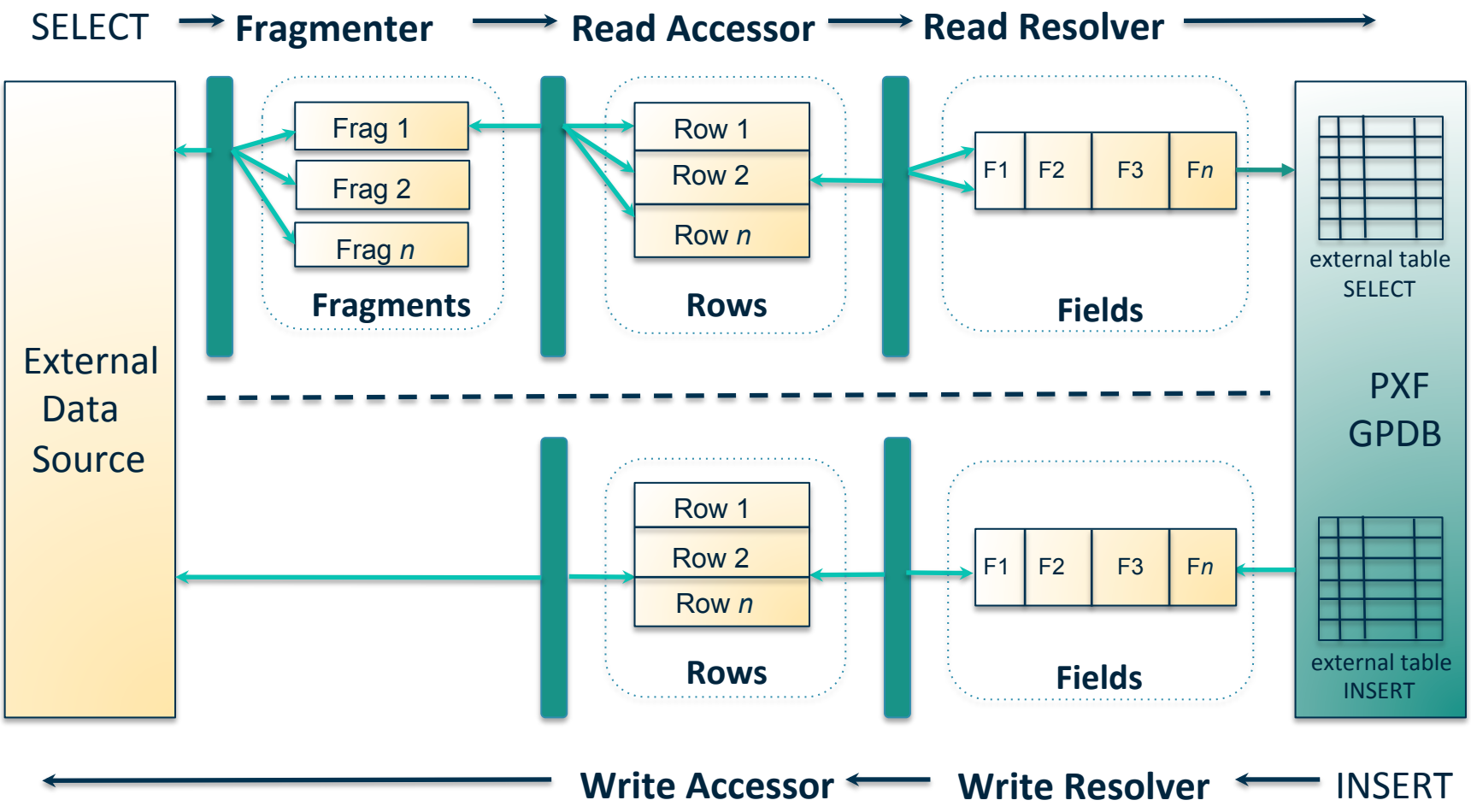
PXF > Other Connectors

- Apache HBase connector

- JDBC connector (community)

- Apache Ignite connector (community)

- Alluxio connector (community)


SELECT
Fragmenter
Read Accessor
Read Resolver
External Data Source
Frag 1
Frag 2
Frag n
Fragments
Row 1
Row 2
Row n
Rows
F1
F2
F3
Fn
Fields
external table SELECT
PXF GPDB
external table INSERT
Row 1
Row 2
Row n
Rows
F1
F2
F3
Fn
Fields
Write Accessor
Write Resolver
INSERT