HPCC JDBC Driver

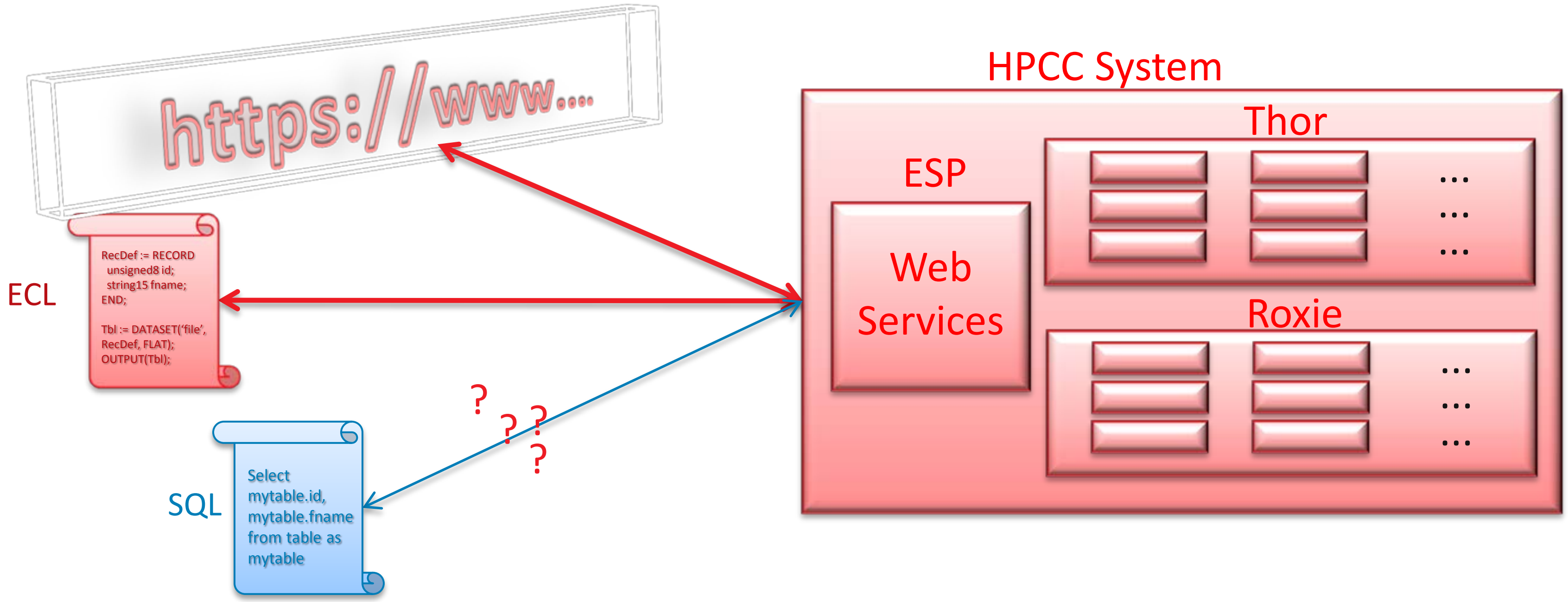
Rodrigo Pastrana

LexisNexis®

Risk Solutions

Leverage data in HPCC via SQL?

- HPCC queries typically written in the powerful ECL language
- Data accessed via raw ECL or published ECL queries
- Currently existing SQL queries would require translation
- Many existing SQL based tools cannot easily access HPCC



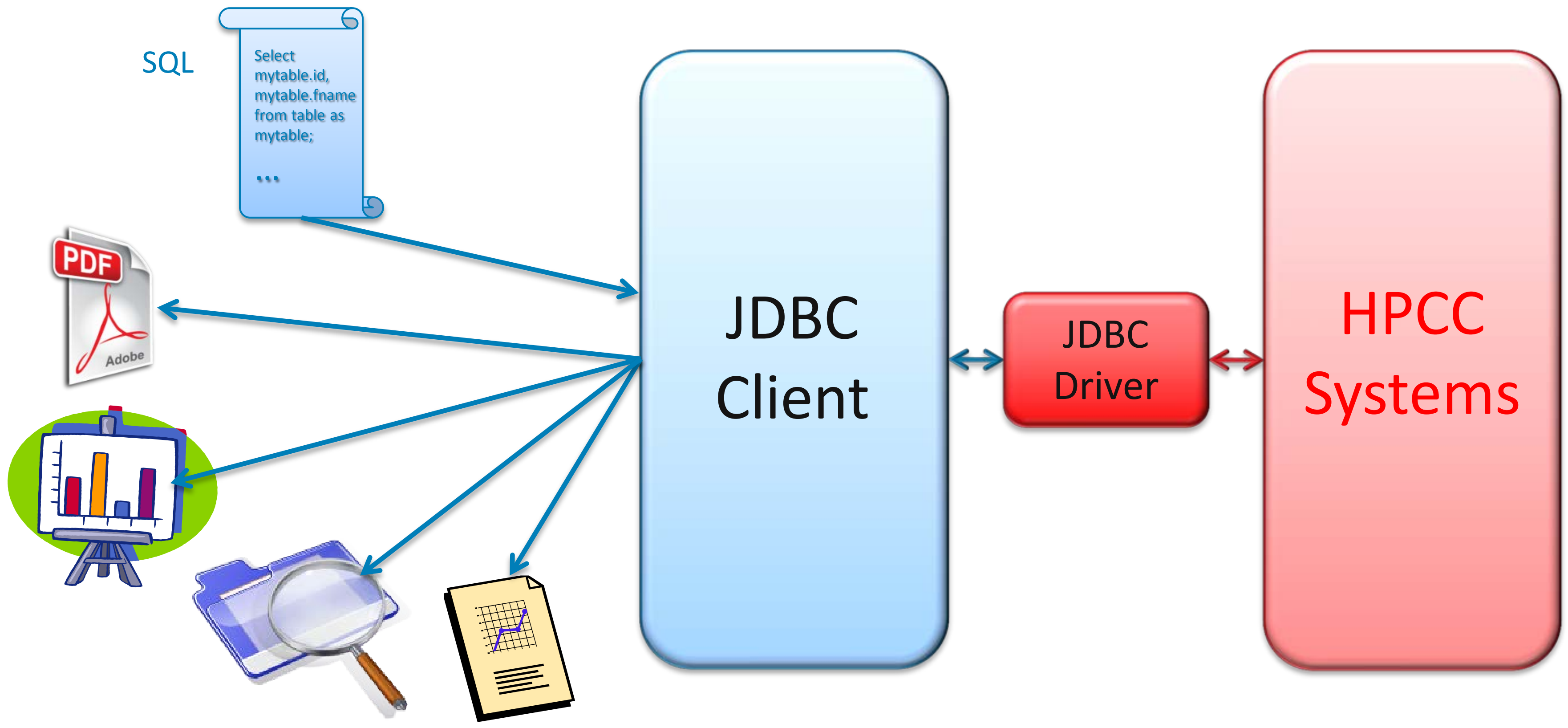
```
RecDef := RECORD
  unsigned8 id;
  string15 fname;
END;

Tbl := DATASET('file',
RecDef, FLAT);
OUTPUT(Tbl);
```

```
Select
mytable.id,
mytable.fname
from table as
mytable
```

HPCC JDBC Driver – the missing piece is now available!

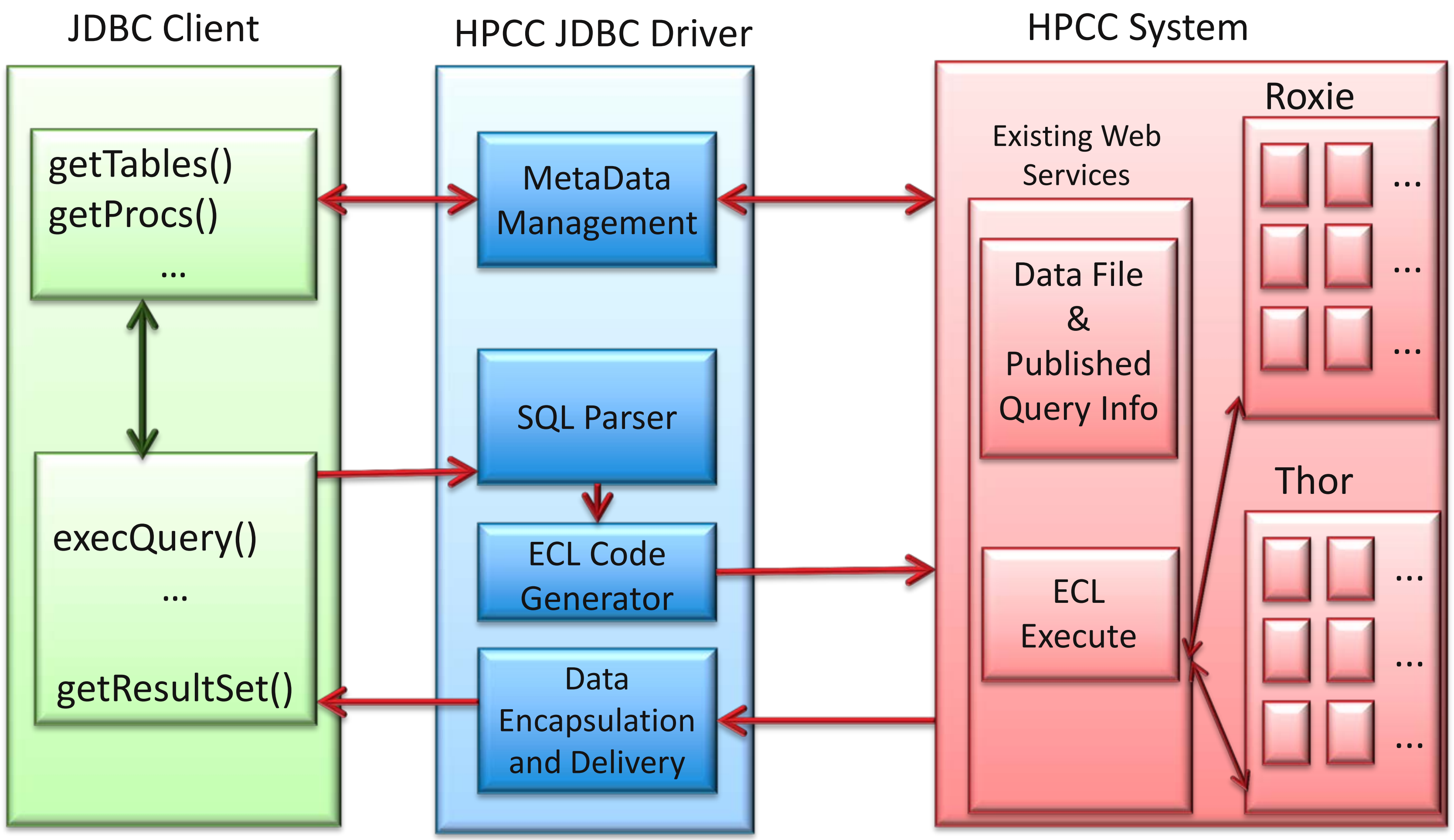
- JDBC Driver for HPCC provides clean SQL-based interface into HPCC Systems
- Many feature-rich JDBC clients are available and can now connect to HPCC
- Many existing SQL-based processes can now target HPCC data



What can be done with the HPCC JDBC Driver?

- Analyze HPCC data via many JDBC clients
- Utilize simple SQL SELECT or CALL syntax
  - Access HPCC data files as DB Tables
  - Access published ECL queries as DB Stored Procedures
- Harnesses the full power of HPCC under the covers
  - Submitted SQL request generates dynamic ECL code
  - Queries are executed on the high performance clusters
  - Automatic Index fetching capabilities for quicker data fetches available
- Provides entry-point for programmatic data access
  - Custom Java programs can easily leverage HPCC data
- Leverage HPCC data and JDBC client functionality without need to learn and write ECL!

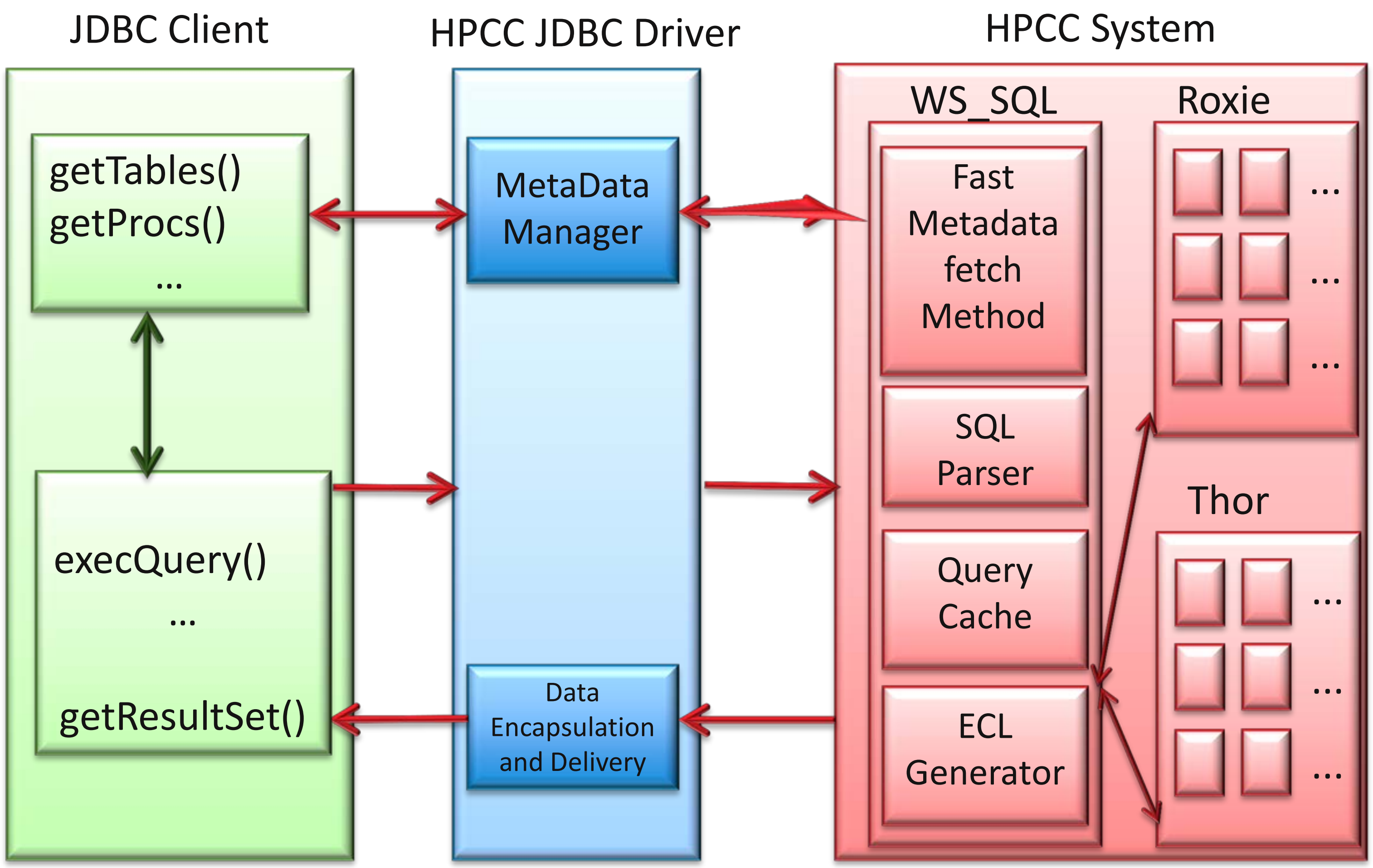
Phase 1 – Thick JDBC Driver



Phase 1 – Thick JDBC Driver

- Provides SQL interface for HPCC!
- Driver is responsible for SQL parsing and ECL code generation
- Existing HPCC Web Services utilized for metadata fetch, and ECL query submission
- Backward compatible with older HPCC versions
- Available now
  - http://www.hpccsystems.com – Downloads, documentation, notes.
  - https://github.com/hpcc-systems/HPCC-Platform/hpcc-jdbc - Source

Phase 2 – Thin JDBC Driver – Dedicated Web service



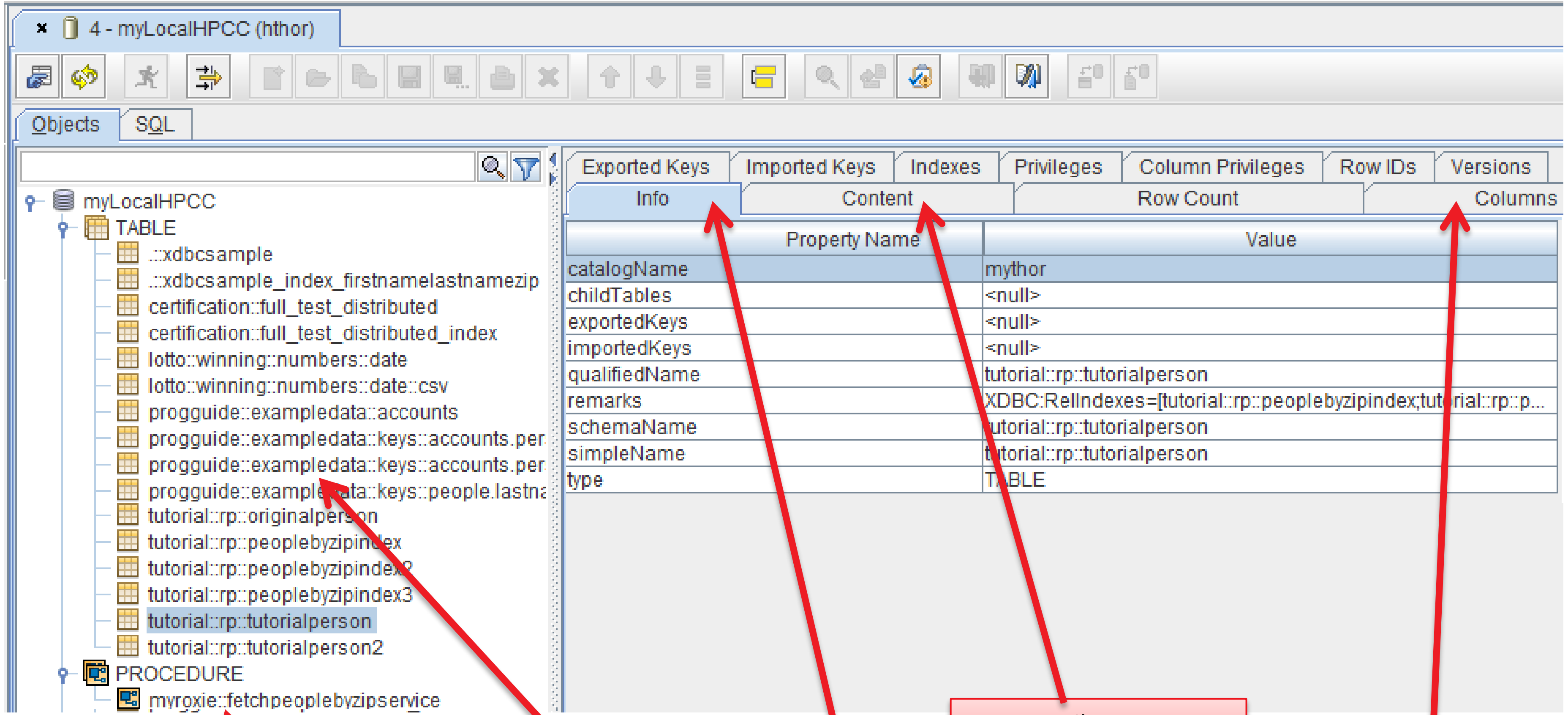
Phase 2 – Thin JDBC Driver – Dedicated Web service

- Major performance improvements
- Delegates SQL processing and ECL generation to back-end server
- Utilizes internal query compilation tracking mechanism
  - Provides means for compiled query re-use
  - Allows result fetch on previously executed queries
  - Introduces paged result set fetch
- Introduces SQL query caching on back-end
  - SQL normalized and mapped to compiled/executed ECL query
- Available – very soon.

Can SQL entirely replace ECL on HPCC?

- No, ECL is very powerful and cannot be replaced by SQL
  - Production level process should be published ECL Query based
- Subset of Read-only SQL operations supported
- Data updates must be performed via ECL or dedicated interfaces
- Powerful published ECL queries accessed via SQL call commands
  - SQL based processes can make use of published ECL queries

HPCC system view via JDBC client


4 - myLocalHPCC (hthor)
Objects
SQL
Exported Keys
Imported Keys
Indexes
Privileges
Column Privileges
Row IDs
Versions
Info
Content
Row Count
Columns
myLocalHPCC
TABLE
.::xdbcsample
.::xdbcsample_index_firstnamelastnamezip
certification::full_test_distributed
certification::full_test_distributed_index
lotto::winning::numbers::date
lotto::winning::numbers::date::csv
progguide::exampledata::accounts
progguide::exampledata::keys::accounts.per
progguide::exampledata::keys::accounts.per
progguide::exampledata::keys::people.lastna
tutorial::rp::originalperson
tutorial::rp::peoplebyzipindex
tutorial::rp::peoplebyzipindex2
tutorial::rp::peoplebyzipindex3
tutorial::rp::tutorialperson
tutorial::rp::tutorialperson2
PROCEDURE
myroxie::fetchpeoplebyzipservice

Property Name	Value
catalogName	mythor
childTables	<null>
exportedKeys	<null>
importedKeys	<null>
qualifiedName	tutorial::rp::tutorialperson
remarks	XDBC:RelIndexes=[tutorial::rp::peoplebyzipindex;tutorial::rp::p...
schemaName	tutorial::rp::tutorialperson
simpleName	tutorial::rp::tutorialperson
type	TABLE

HPCC published queries listed as stored procedures

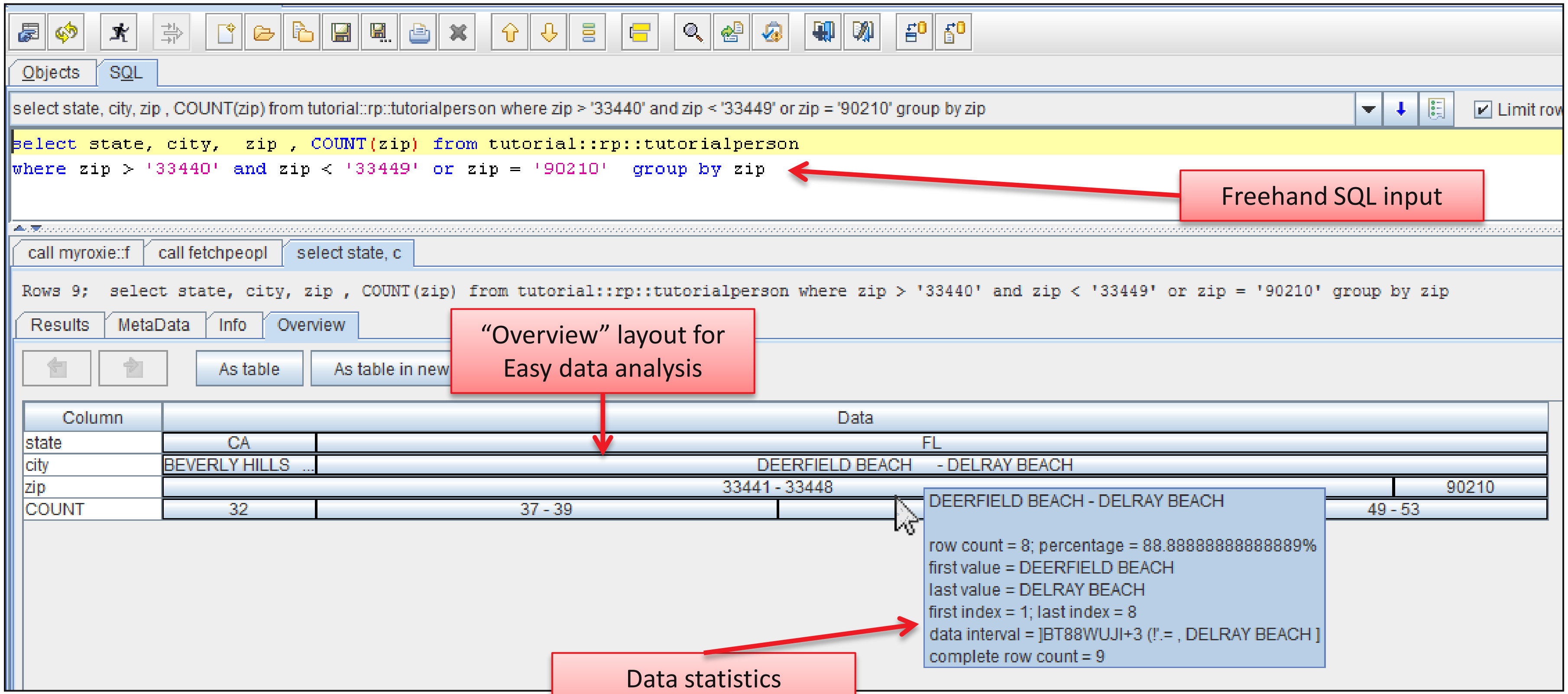
HPCC Data files listed as tables

HPCC File Metadata is available

HPCC Files content easily fetched

HPCC Files column information is available

HPCC data view via JDBC Client



```
select state, city, zip , COUNT(zip) from tutorial::rp::tutorialperson where zip > '33440' and zip < '33449' or zip = '90210' group by zip
```

```
select state, city,  zip , COUNT(zip) from tutorial::rp::tutorialperson
where zip > '33440' and zip < '33449' or zip = '90210'  group by zip
```

Freehand SQL input

```
Rows 9;  select state, city, zip , COUNT(zip) from tutorial::rp::tutorialperson where zip > '33440' and zip < '33449' or zip = '90210' group by zip
```

"Overview" layout for Easy data analysis

Column	Data			
state	CA	FL		
city	BEVERLY HILLS ...	DEERFIELD BEACH - DELRAY BEACH		
zip	33441 - 33448			90210
COUNT	32	37 - 39		49 - 53

```
DEERFIELD BEACH - DELRAY BEACH

row count = 8; percentage = 88.88888888888889%
first value = DEERFIELD BEACH
last value = DELRAY BEACH
first index = 1; last index = 8
data interval = ]BT88WUJI+3 (!'.= , DELRAY BEACH ]
complete row count = 9
```

Data statistics displayed in "fly-over"

SQL Syntax supported

- **SQL CALL: call** *queryname* **([***param list***])**
  - Example: ***call*** *MyHPCCStoredProcedure(33024)*
- **SQL SELECT: select** columnList **from** tableName [as alias]

  [USE INDEX(indexFileName | 0)] [where filterCondition]
  [group by fieldName] [order by columnNames [asc | desc]]

  [LIMIT limitNumber]
  - Example: ***select*** *city, zip , COUNT(zip)* ***from*** *tutorial::rjp:tutorialperson* ***where*** *zip > '33440'* ***group by*** *zip* ***limit*** *100*
- **SQL SELECT Join: select** *columnList* **from** *tableName*

  [<outer | inner > **JOIN** *join TableName* **[as** *alias***] on** *joinCondition***]**
  - Example: ***select*** *t1.person, t2.address from persontable as t1*

    inner join *addresstable as t2* ***on*** *t1.personid = t2.personid*

Sample programmatic query (JAVA)

```
Properties info = new Properties();
try
{
    Driver hpccdriver = DriverManager.getDriver("jdbc:hpcc");
    HPCCConnection hpccconnection =
            (HPCCConnection) hpccdriver.connect("jdbc:hpcc;ServerAddress=192.168.1.1;", info);

    PreparedStatement prepstm = hpccconnection.prepareStatement("select * from  myTable");
    ResultSet qrs = (( HPCCPreparedStatement) prepstm).executeQuery();

    ResultSetMetaData meta = qrs.getMetaData();

    while (qrs.next())
    {
        for (int i = 1; i <= meta.getColumnCount(); i++)
        {
            System.out.print("[ " + qrs.getObject(i) + " ]");
        }
        System.out.println();
    }
}
catch (Exception e) { System.out.println("Error"); }
```

Miscellaneous

- Driver provided as a single JAR file
- Feature set is growing and improving
- Package and documentation:
  - http://hpccsystems.com/products-and-services/products/plugins/JDBC-Driver
- Source code and issue tracker:
  - https://github.com/hpcc-systems/hpcc-jdbc
- Questions and discussions:
  - http://hpccsystems.com/bb/viewforum.php?f=34&sid=8b5b0f7b5a621ece8abac036067d9db0
- Listen to the HPCC JDBC Driver podcast:
  - http://cdn.hpccsystems.com/podcasts/2013_1001_JDBC.mp3
- Feedback is welcomed and encouraged!
  - Join our community at http://hpccsystems.com