Temporal Instrumental Database

TIDB2

ROOT Workshop 2005

29 September

João Simões

António Amorim

Ricardo Neves

CERN

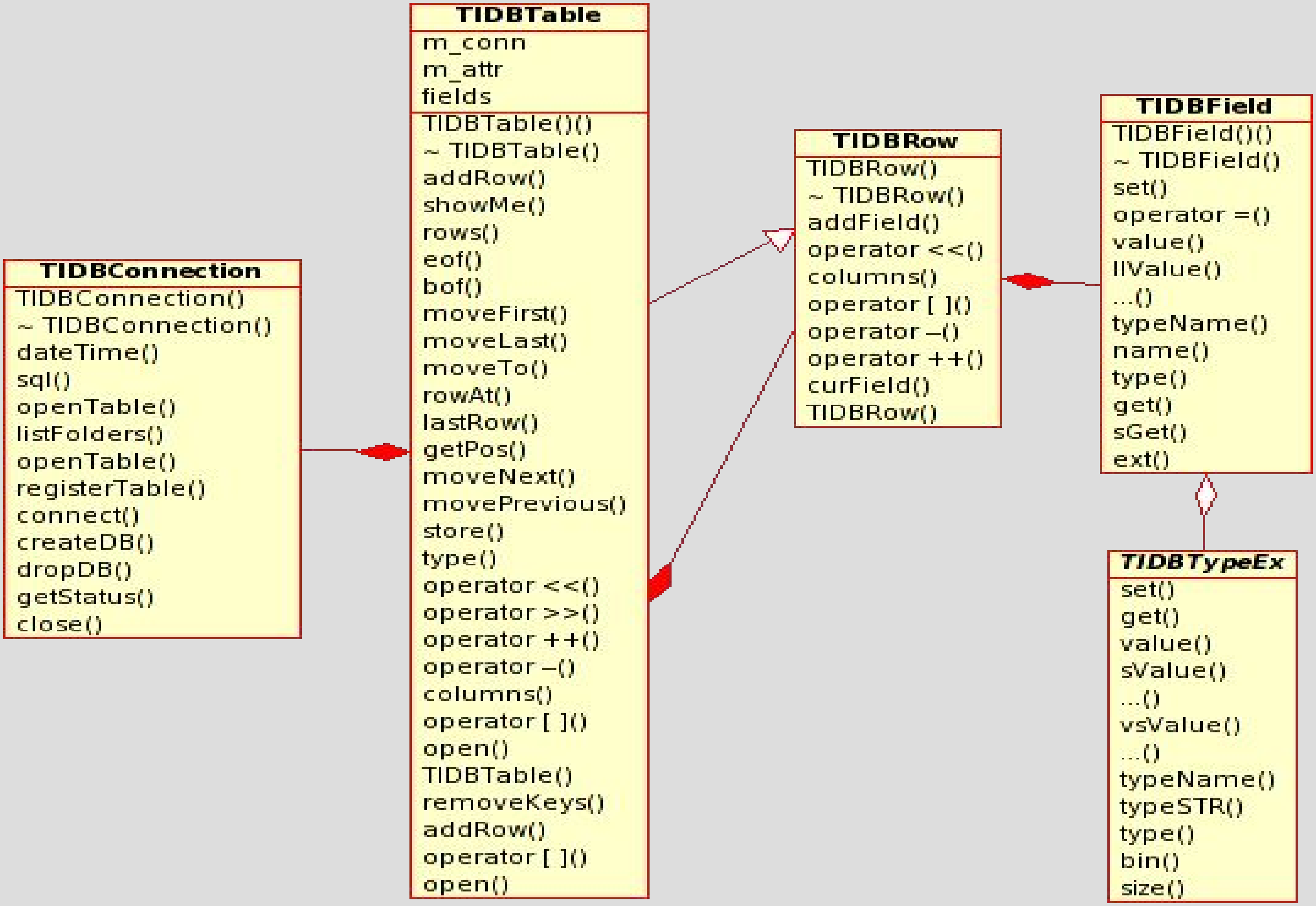
What is TIDB2?

- Fully featured temporal technics database.
- Simple and intuitive C++ interface.
- No need for factories, they are in a layer that users don't use – all classes are standard C++ classes.
- RDBMS independent (via runtime plugin).
- Oriented to store any kind of scientific objects (via runtime plugin).
- Automatic index creation, based on object's schema.

TIDB2 and Conditions

- Using experience with the Lisbon Conditions DB interface.
- While the immediate production needs of HEP databases are being addressed by COOL-LCG.
- We felt it was usefull an R&D effort to address:
  - indexing scientific object data on time or other varibles in a relational database;
  - Wider scope of table and table field object data;
  - Intuitive interface and runtime plugin approach

General Structure



TIDBTable

m_conn
m_attr
fields

TIDBTable()()
~ TIDBTable()
addRow()
showMe()
rows()
eof()
bof()
moveFirst()
moveLast()
moveTo()
rowAt()
lastRow()
getPos()
moveNext()
movePrevious()
store()
type()
operator <<()
operator >>()
operator ++()
operator --()
columns()
operator [ ]()
open()
TIDBTable()
removeKeys()
addRow()
operator [ ]()
open()

TIDBConnection

TIDBConnection()
~ TIDBConnection()
dateTime()
sql()
openTable()
listFolders()
openTable()
registerTable()
connect()
createDB()
dropDB()
getStatus()
close()

TIDBRow

TIDBRow()
~ TIDBRow()
addField()
operator <<()
columns()
operator [ ]()
operator --()
operator ++()
curField()
TIDBRow()

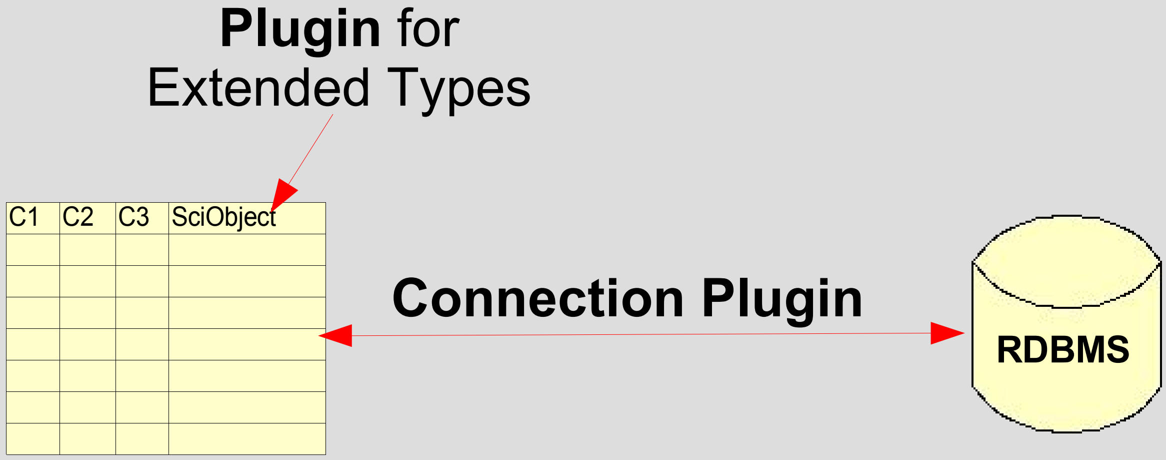
TIDBField

TIDBField()()
~ TIDBField()
set()
operator =()
value()
llValue()
...()
typeName()
name()
type()
get()
sGet()
ext()

TIDBTypeEx

set()
get()
value()
sValue()
...()
vsValue()
...()
typeName()
typeSTR()
type()
bin()
size()

Plugin Architecture



Plugin for
Extended Types

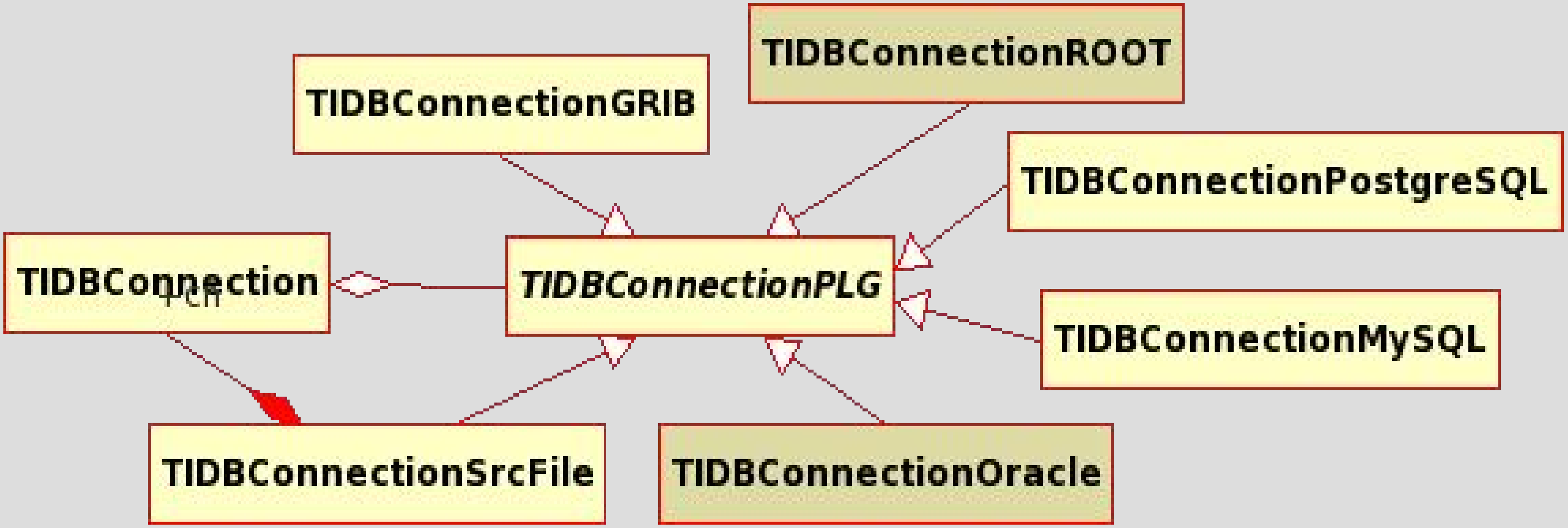
C1	C2	C3	SciObject

Connection Plugin

RDBMS

- **Plugins are shared libraries loaded at runtime.**
- **Connection plugin stores and retrieves data from the DB.**
- **Extended Type plugin manages the columns containing scientific objects.**

The Connection Object



TIDBConnectionROOT

TIDBConnectionGRIB

TIDBConnectionPostgreSQL

TIDBConnection

+ch

TIDBConnectionPLG

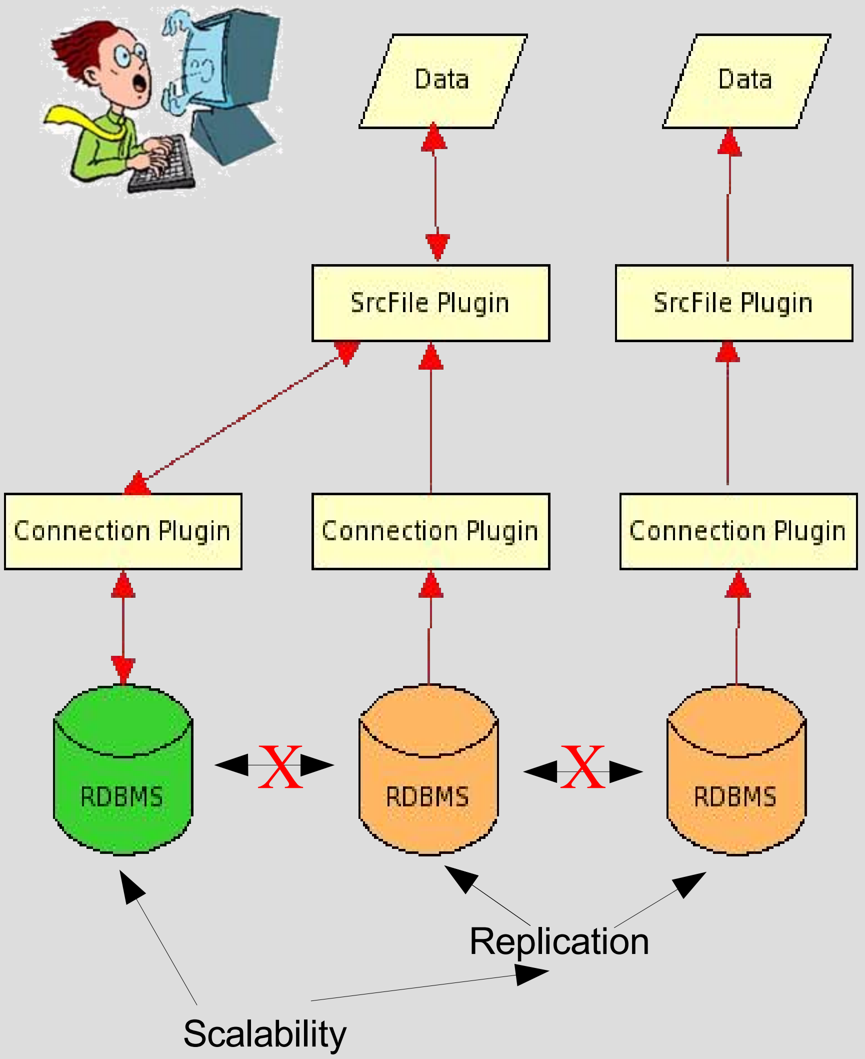
TIDBConnectionMySQL

TIDBConnectionSrcFile

TIDBConnectionOracle

- The TIDBConnection selects the appropriate plugin that will handle the connection (ex. mysql://, oracle://).
- All plugins implement TIDBConnectionPLG (providing all core functions to manipulate the database).

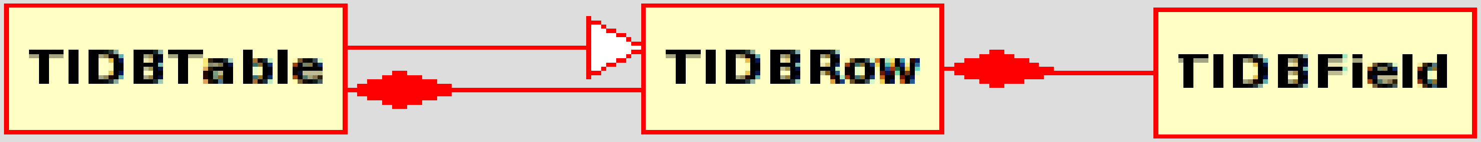
The Source File Plugin



- Uses a Debian “apt-get” like mechanism.
- Servers references are written to a source list file:
  - DB/Connection/Time Period.
- Makes scalability simple.
- Makes replication simple.

The TIDB2 Transient Table

- TIDBTable is avaliable when returning result sets and for storing procedures:



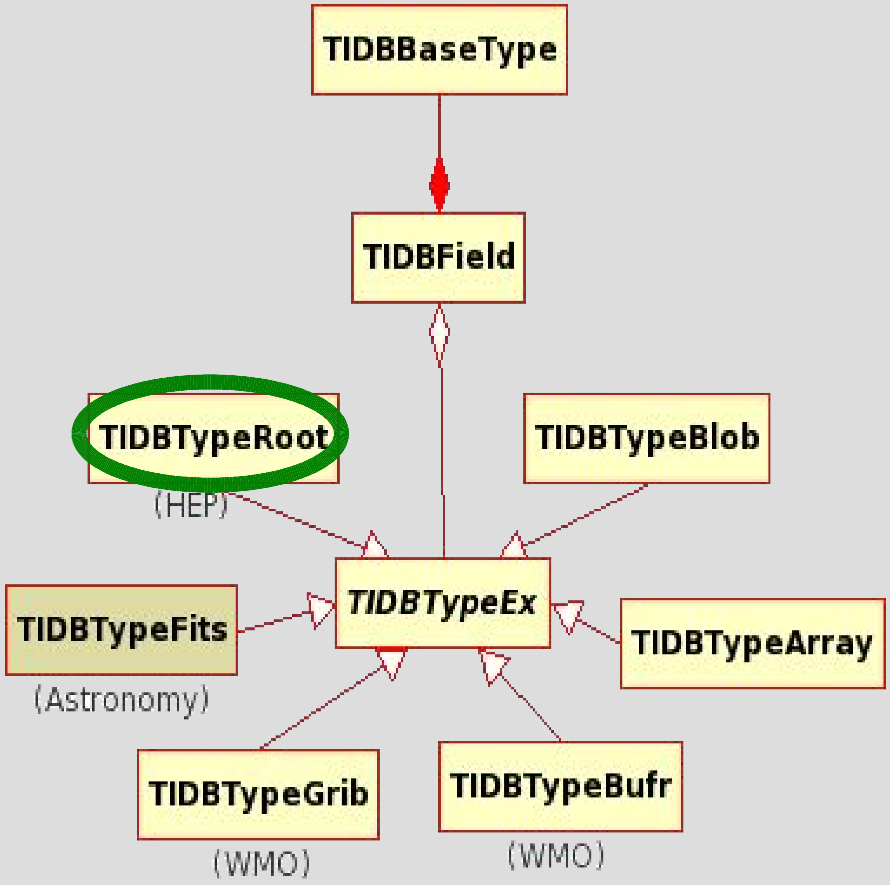
- Tables can be built from scratch using a row as a model, filled with rows and stored.

- Tables can be retrieved from a DB by a TIDBConnection, appended with rows and stored.

- The resulting table could be the result of a SQL query.

- Any external table can be registered in the TIDB database, and opened as a TIDBTable.

The TIDBField and TIDBTypeEX



- TIDBField manages the data types.
- TIDBField provides an interface between the user and the extended types.
- With the appropriate plugin any data type can be supported.
- It's easy to fill a TIDBField with data.

The Special << Operator

```
TIDBRow MyRow(table)  << 1 << “2” << 3.0;
```

- The “clever <<” operator automaticlly casts the data to the respective column type.
- This operator has a special behavior while streaming extended data types.
- TIDBRows can be streamed sequentially into a table.

Complex Data Storage Approach

- Atomized «complex data type» storage:
  -The BLOB is splited into all it's elements.
  -Lots of data redundancy or associations.
  -Occupies a lot of storage space.

- The data is keeped as BLOBs:
  -Unsuitable for seeking objects.
  -Makes it impossible to quickly find the most relevant data properties.

- Mixed mode TIDB2 approach.

Atomized

A	B	C	D	**E**	F	G	H	I	J	K	L
A	B	C	D	**E**	F	G	H	I	J	K	L
A	B	C	D	**E**	F	G	H	I	J	K	L
A	B	C	D	**E**	F	G	H	I	J	K	L
A	B	C	D	**E**	F	G	H	I	J	K	L
A	B	C	D	**E**	F	G	H	I	J	K	L
A	B	C	D	**E**	F	G	H	I	J	K	L
A	B	C	D	**E**	F	G	H	I	J	K	L

BLOB storage

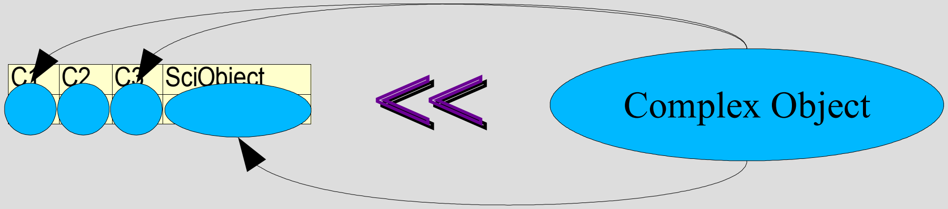
BLOB
BLOB
BLOB
BLOB
BLOB

TIDB2 mixed mode

A	**E**	H	BLOB
A	**E**	H	BLOB
A	**E**	H	BLOB
A	**E**	H	BLOB
A	**E**	H	BLOB
A	**E**	H	BLOB
A	**E**	H	BLOB
A	**E**	H	BLOB

Indexing Scientific Objects

TIDBRow MyRow(table) << SciObject;



- “Streamer” analyses the internal structure of complex objects.
- “Streamer” automatically fills fields matching data inside the SciObject.
- The key fields are tunable (depends on the user defined table structure).

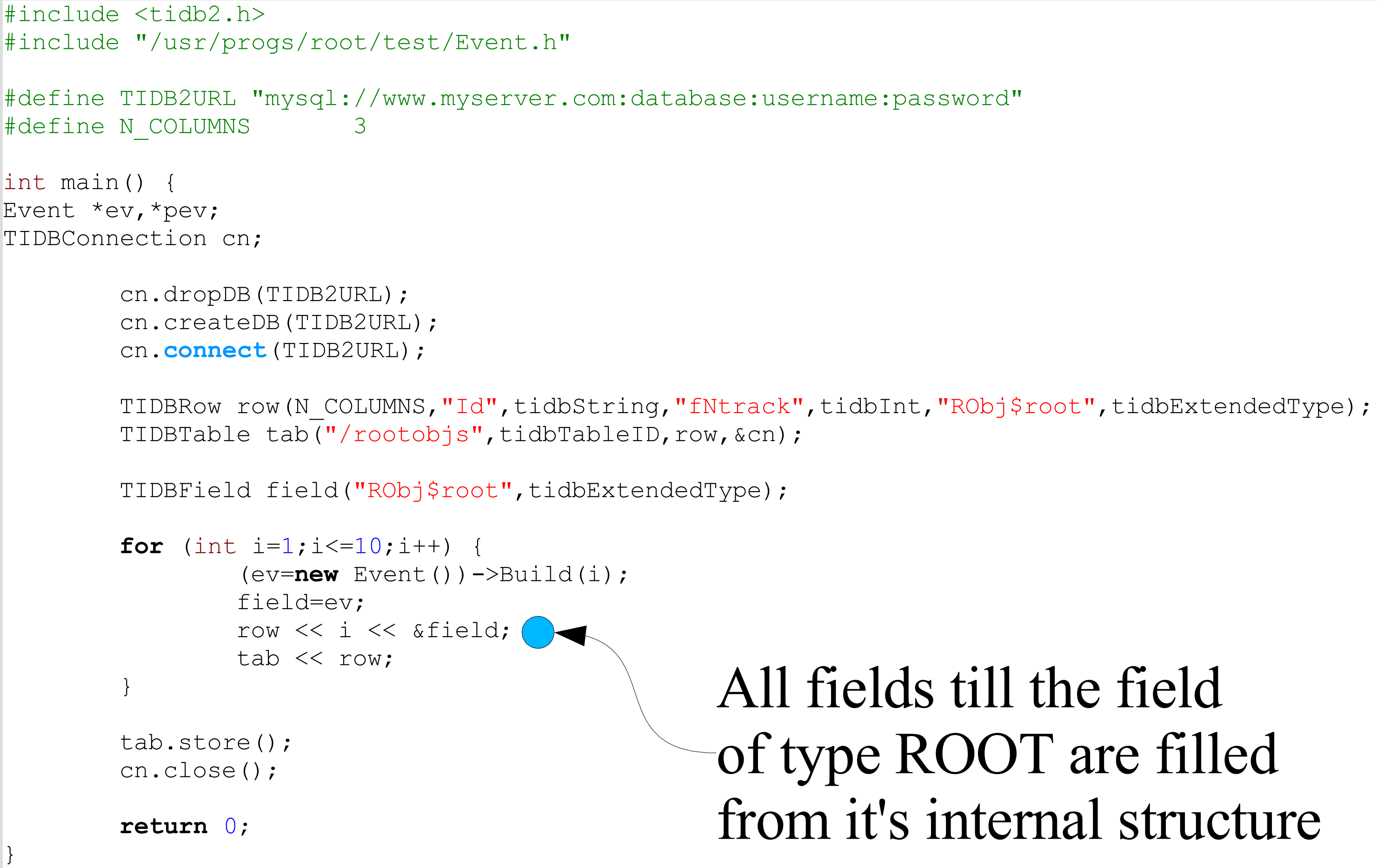
The ROOT Extended Type

- Any TClass object inside a root file could be loaded into a TIDBTable via TIDBField get method.

- Any pointer to TClass object could be used to load the object into a TIDBTable via the “clever streamer” operator.

- Is not crucial to have the object library for loading and “unpacking” the data.

Coding Example (Storing)



```
#include <tidb2.h>
#include "/usr/progs/root/test/Event.h"

#define TIDB2URL "mysql://www.myserver.com:database:username:password"
#define N_COLUMNS       3

int main() {
Event *ev,*pev;
TIDBConnection cn;

        cn.dropDB(TIDB2URL);
        cn.createDB(TIDB2URL);
        cn.connect(TIDB2URL);

        TIDBRow row(N_COLUMNS,"Id",tidbString,"fNtrack",tidbInt,"RObj$root",tidbExtendedType);
        TIDBTable tab("/rootobjs",tidbTableID,row,&cn);

        TIDBField field("RObj$root",tidbExtendedType);

        for (int i=1;i<=10;i++) {
                (ev=new Event())->Build(i);
                field=ev;
                row << i << &field;
                tab << row;
        }

        tab.store();
        cn.close();

        return 0;
}
```

All fields till the field of type ROOT are filled from it's internal structure

- The object could be also streamed directly into the TIDBRow.

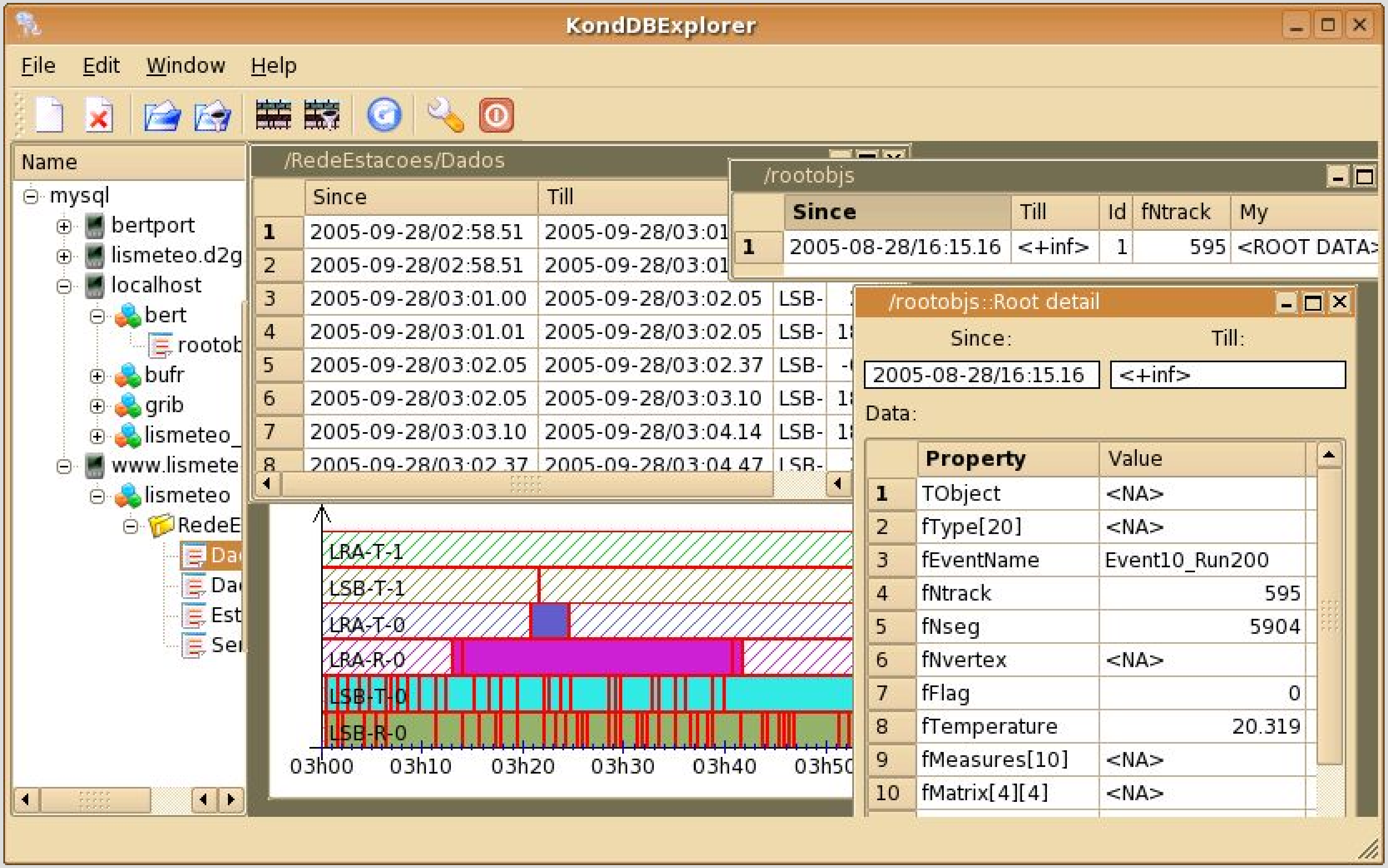
Coding Example (Reading)

```
int main() {
TIDBConnection cn;

        cn.connect(TIDB2URL);
        TIDBTable tab(&cn);

        tab.open("/rootobjs");
        tab.showMe(19);
        cn.close();

        TIDBTypeEx *robj=tab["RObj"].ext();
        cout << "CLASS NAME: [" << robj->typeName()<<"]"<<endl;
        for (int i=0;tab["RObj"].get(i);i++)
                cout <<robj->vsValue()[i]<<" = " << tab["RObj"].sGet(i)<<endl;
        return 0;
}
```

- By using TstreamerInfo, there is no need to link this example with our custom class (ex: libEvent.so).

- If we link with the class library we can cast the object from TIDBTypeEx::value() to our class.

TIDB2 Browser: KTidbExplorer



/RedeEstacoes/Dados

	Since	Till		
1	2005-09-28/02:58.51	2005-09-28/03:01		
2	2005-09-28/02:58.51	2005-09-28/03:01		
3	2005-09-28/03:01.00	2005-09-28/03:02.05	LSB-	
4	2005-09-28/03:01.01	2005-09-28/03:02.05	LSB-	1
5	2005-09-28/03:02.05	2005-09-28/03:02.37	LSB-	-
6	2005-09-28/03:02.05	2005-09-28/03:03.10	LSB-	1
7	2005-09-28/03:03.10	2005-09-28/03:04.14	LSB-	1
8	2005-09-28/03:02.37	2005-09-28/03:04.47	LSB-	

/rootobjs

	Since	Till	Id	fNtrack	My
1	2005-08-28/16:15.16	<+inf>	1	595	<ROOT DATA>

/rootobjs::Root detail

Since: 2005-08-28/16:15.16

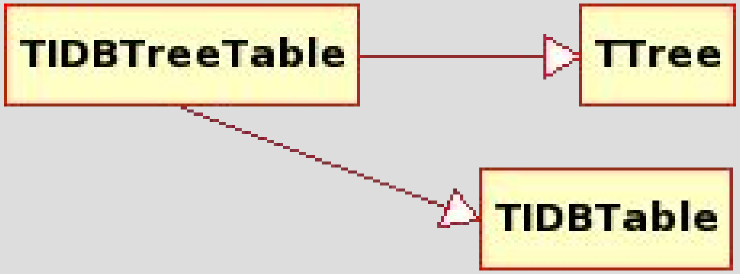
Till: <+inf>

Data:

	Property	Value
1	TObject	<NA>
2	fType[20]	<NA>
3	fEventName	Event10_Run200
4	fNtrack	595
5	fNseg	5904
6	fNvertex	<NA>
7	fFlag	0
8	fTemperature	20.319
9	fMeasures[10]	<NA>
10	fMatrix[4][4]	<NA>

Future Work

- TIDBTreeTable to implement TIDBTable methods for a Ttree.
- All the power of ROOT analysis tools would be integrated with a (semi) relational database.



- All streamer info relative to ROOT objects will be written into a table of schemas inside the DB (transparent to the end user).
  - No need to link with addicional libraries than -ltidb2.
- TIDBConnectionOracle
- TIDBTypeFITS

How to get TIDB2?

- To download tidb2 from CVS:
- cvs -d:pserver:anonymous@cvs.sourceforge.net:/cvsroot/t-i-db login
- cvs -z3 -d:pserver:anonymous@cvs.sourceforge.net:/cvsroot/t-i-db co -P tidb2
- To download ktidbexplorer from CVS:
- cvs -z3 -d:pserver:anonymous@cvs.sourceforge.net:/cvsroot/t-i-db co -P \
  ktidbexplorer
- Tarballs can be found at:
- https://sourceforge.net/project/showfiles.php?group_id=117005
- To contact me for help:
- Email to joao.simoes@fisica.fc.ul.pt