Subject: Re: PROPOSAL: Access to S_* Structure of TABLE crash Application.
Posted by Sender Ghost on Fri, 18 May 2012 20:27:55 GMT
View Forum Message <> Reply to Message

sergeynikitin wrote on Fri, 18 May 2012 21:27A colorful example of how we ourselves create artificial boundaries.
Yes, my conclusions based on current implementation. But I didn't say, that this is not possible in principle.
I showed real example about how to achieve this with current tools.
Quote:Otherwise, we are forced to write for each table, in which there is a field NAME or ID some artificial design.

I think, no. It's normal to write as follows:

```
SQL * Select(ID, NAME, LASTNAME, PLANT_ID).From(WORKER);
LOG(SQL.ToString());
while (SQL.Fetch()) {
 LOG("-----");
 DUMP(SQL[ID]);
 DUMP(SQL[NAME]);
 DUMP(SQL[LASTNAME]);
 DUMP(SQL[PLANT_ID]);
}

SQL * Select(ID, NAME, ADDRESS).From(PLANT);
LOG('\n' << SQL.ToString());
while (SQL.Fetch()) {
 LOG("-----");
 DUMP(SQL[ID]);
 DUMP(SQL[NAME]);
 DUMP(SQL[ADDRESS]);
}
```

With following output:

```
select ID, NAME, LASTNAME, PLANT_ID from WORKER
-----
SQL[ID] = 0
SQL[NAME] = Joe
SQL[LASTNAME] = Smith
SQL[PLANT_ID] = 0
-----
SQL[ID] = 1
SQL[NAME] = Mike
SQL[LASTNAME] = Smith
SQL[PLANT_ID] = 0
-----
SQL[ID] = 2
```

SQL[NAME] = Jon
SQL[LASTNAME] = Goober
SQL[PLANT_ID] = 1

select ID, NAME, ADDRESS from PLANT

SQL[ID] = 0
SQL[NAME] = First Plant
SQL[ADDRESS] = First st.

SQL[ID] = 1
SQL[NAME] = Second Plant
SQL[ADDRESS] = Second st.

For cases with column name clashes there is "AS" clause. And even without this, it still possible to access result set through indexes.
