Subject: Re: Access to S_* Structure of TABLE crash Application.
Posted by Sender Ghost on Thu, 17 May 2012 07:59:52 GMT
View Forum Message <> Reply to Message

sergeynikitin wrote on Thu, 17 May 2012 07:53For example, there are two tables:

```
TABLE_(WORKER)
  INT_    (ID) PRIMARY_KEY
  STRING_  (NAME, 200)
  STRING_  (LASTNAME, 200)
  INT_    (PLANT_ID)
END_TABLE

TABLE_(PLANT)
  INT_    (ID) PRIMARY_KEY
  STRING_  (NAME, 200)
  STRING_  (ADDRESS, 200)
END_TABLE
```

Correct version of tables:

```
TABLE_(WORKER)
  INT_    (ID) PRIMARY_KEY
  STRING_  (NAME, 200)
  STRING_  (LASTNAME, 200)
  INT_    (PLANT_ID)
END_TABLE

TABLE_(PLANT)
  INT    (ID) PRIMARY_KEY
  STRING  (NAME, 200)
  STRING_ (ADDRESS, 200)
END_TABLE
```

The INT_(ID) macro creates new extern SqlId ID;. The same with other *_ macro versions. If you want to use the same SqlId, there are macros without '_' postfix.
Quote:
I think, that S_* structure created to use in this situation, to decide this collision.

```
S_WORKER w;
S_PLANT p;

SQL * Select
(w.NAME,w.LASTNAME,p.NAME,p.ADDRESS).FROM(WORKER).LeftJoin(PLANT).On(w.PLANT
_ID==p.ID);
```

```
while SQL.Fetch() {
   Cout() << SQL[w.NAME] << SQL[w.LASTNAME] << SQL[p.NAME] << SQL[p.ADDRESS];
}
```

Correct version of source code, might look like:

```
LOG("Inserting values:");
SQL * Insert(WORKER)(ID, 0)(NAME, "Joe")(LASTNAME, "Smith")(PLANT_ID, 0);
LOG(SQL.ToString());
SQL * Insert(WORKER)(ID, 1)(NAME, "Mike")(LASTNAME, "Smith")(PLANT_ID, 0);
LOG(SQL.ToString());
SQL * Insert(WORKER)(ID, 2)(NAME, "Jon")(LASTNAME, "Goober")(PLANT_ID, 1);
LOG(SQL.ToString());

SQL * Insert(PLANT)(ID, 0)(NAME, "First Plant")(ADDRESS, "First st.");
LOG(SQL.ToString());
SQL * Insert(PLANT)(ID, 1)(NAME, "Second Plant")(ADDRESS, "Second st.");
LOG(SQL.ToString());

LOG("Selecting values:");
SQL * Select(WORKER(NAME, LASTNAME), PLANT(NAME,
ADDRESS)).From(WORKER).LeftJoin(PLANT).On(WORKER(PLANT_ID) == PLANT(ID));
LOG(SQL.ToString());
while (SQL.Fetch()) {
 LOG("-----");
 DUMP(SQL[NAME]); DUMP(SQL[LASTNAME]);
 DUMP(SQL[2]); DUMP(SQL[3]);
}
```

With following output:

```
Inserting values:
insert into WORKER(ID, NAME, LASTNAME, PLANT_ID) values (0, 'Joe', 'Smith', 0)
insert into WORKER(ID, NAME, LASTNAME, PLANT_ID) values (1, 'Mike', 'Smith', 0)
insert into WORKER(ID, NAME, LASTNAME, PLANT_ID) values (2, 'Jon', 'Goober', 1)
insert into PLANT(ID, NAME, ADDRESS) values (0, 'First Plant', 'First st.')
insert into PLANT(ID, NAME, ADDRESS) values (1, 'Second Plant', 'Second st.')
Selecting values:
select WORKER.NAME, WORKER.LASTNAME, PLANT.NAME, PLANT.ADDRESS from
WORKER left outer join PLANT on WORKER.PLANT_ID = PLANT.ID
-----
SQL[NAME] = Joe
SQL[LASTNAME] = Smith
SQL[2] = First Plant
SQL[3] = First st.
-----
SQL[NAME] = Mike
```

SQL[LASTNAME] = Smith
SQL[2] = First Plant
SQL[3] = First st.

SQL[NAME] = Jon
SQL[LASTNAME] = Goober
SQL[2] = Second Plant
SQL[3] = Second st.