Database - Bug #2942

shared temp-tables can use fields with different extent than the master temp-table

12/21/2015 06:34 AM - Constantin Asofiei

Status:	New	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**case_num:**	
vendor_id:	GCD	**version:**	
Description			

History

#1 - 12/21/2015 06:35 AM - Constantin Asofiei

From #2795 note 22:

> Finally, about shared temp-tables and extent fields: there is a new issue which is (I think) outside of the scope of this task. Consider a master (new shared) temp-table with an extent field f1 of 2. A slave (shared) temp-table is defined with field f1 extent of 3. When going back to the master temp-table, accessing index 3 directly is not possible, but is possible via the :: dereference operator. The issues here:
>
> 1. P2J doesn't support h::f1(3) construct (the Dereferenceable interface doesn't have APIs with the array index).
> 2. in the slave temp-table, the TemporaryBuffer.useShared will return the master Temporary instance - and this will limit the extent to 2... the slave temp-table is losing all its custom extent information (and maybe others, related to indexes, labels, etc).

#2 - 12/21/2015 06:41 AM - Constantin Asofiei

Testcases which show this problem:

1. master temp-table (st1.p):

```
def new shared temp-table tt1 field f1 as int extent 2.

run st2.p.

def var i as int.
def var h as handle.

find first tt1.
h = buffer tt1:handle.

i = h::f1(3) no-error.

if i <> 12345 then message "master tt1.f1(3) must be 12345!".
```

2. slave temp-table (st2.p):

```
def shared temp-table tt1 field f1 as int extent 5.
```

```
create tt1.
tt1.f1 = 12345.

def var i as int.
i = tt1.f1[3].

if extent(tt1.f1) <> 5 then message "shared extent(tt1.f1) must be 5!".
if i <> 12345 then message "shared tt1.f1(3) must be 12345!".
```

Another issue to check is initialization: if tt1.f1 is defined as field f1 as int extent 2 init 5432 in st1.p, then doing a create tt1. in st2.p will set tt1.f1. to extent 2 and 5432. So initialization is still based on master def; explicit assignment (tt1.f1 = 12345. in st2.p) looks like is based on the slave temp-table def.