TYPO3.Flow - Bug # 51972

Status:	New	**Priority:**	Should have
Author:	Adrian Föder	**Category:**	
Created:	2013-09-13	**Assigned To:**	Adrian Föder
Updated:	2013-09-13	**Due date:**	
PHP Version:			
Has patch:	Yes		
Complexity:			
Affected Flow version:	Git 2.0		
Subject:	Joins for every deep property constraint make cartesian selection		

Description

When it comes to property lookups as in

```
$query->logicalAnd(
   $query->contains('categories.posts', $firstPost),
   $query->equals('categories.approved', TRUE)
)
```

there are LEFT JOINs for both constraints each, resulting in something like

```
SELECT e FROM Acme\Post e
LEFT JOIN e.categories categories0
LEFT JOIN e.categories categories1
WHERE ?1 MEMBER OF categories0.posts
AND categories1.approved = ?2
```

This, at the end, likely resulted into unpredicted
matches because it would match those Posts where
a category having approved = false is involved in
the post in question, as long as this category belongs
to an (arbitrary) Post having any category approved = true.

Consider the following testing code:

```
1<?php
2public function complexQueryWithJoinsMakesCorrectConjunctions() {
3    $postEntityRepository = new PostRepository;
4
5    $notApprovedCategory = new Category();
6    $approvedCategory = new Category();
7    $approvedCategory->setApproved(TRUE);
8
9    $firstPost = new Post;
10   $firstPost->setTitle('First Post');
11   $firstPost->addCategory($notApprovedCategory);
12   $postEntityRepository->add($firstPost);
13
```

```
14   $secondPost = new Post;
15   $secondPost->setTitle('Second Post');
16   $secondPost->addCategory($notApprovedCategory);
17   $secondPost->addCategory($approvedCategory);
18   $postEntityRepository->add($secondPost);
19
20   $thirdPost = new Post;
21   $thirdPost->setTitle('Third Post');
22   $thirdPost->addCategory($approvedCategory);
23   $postEntityRepository->add($thirdPost);
24
25   $this->persistenceManager->persistAll();
26
27   $query = new Query('TYPO3\Flow\Tests\Functional\Persistence\Fixtures\Post');
28   $query->matching(
29       $query->logicalAnd(
30           $query->contains('categories.posts', $firstPost),
31           $query->logicalNot($query->equals('Persistence_Object_Identifier', $this
->persistenceManager->getIdentifierByObject($firstPost))),
32           $query->equals('categories.approved', TRUE)
33       )
34   );
35   $this->assertEquals(0, $query->count());
36}
```

The intention is to find all Posts which have the same **approved** category as the *First Post*. In the shown example, the *First Post* shares an **unapproved** category with the *Second Post*, and the *Second Post* shares an **approved** category with the *Third Post*.
The query gives a single result which is the *Second Post*, but this is not correct since the *Second Post* does only share an **unapproved** category with the *First Post* which was asked for.
The *Second* only "coincidentally" also is involved in an **approved** category with the *Third Post*.

A fix is under review at https://review.typo3.org/#/c/23751/

History