Query optimization for the quantifier ONLY

Dirk Van Gucht

In this note we will show how an only quantifier gets translated and optimized. Consider the following query:

'Find the pairs (s, b) where s is the sid of a student and b is the bookno of a book such that student s only bought the books that cite book b.'

$$\{(s.sid, b.bookno) | Student(s) \wedge Book(b) \wedge \\ \{t.bookno | \exists t(Buys(t) \wedge t.sid = s.sid)\} \subseteq \{c.bookno \mid \exists c(Cites(c) \wedge c.citedbook = b.bookno)\}\}$$

We expect the final RA expression to be

$$\pi_{sid}(Student) \times \pi_{bookno}(Book) - \pi_{sid,bookno}(Buys \times \pi_{bookno}(Book) - \pi_{sid,bookno,citedbookno}(Buys \bowtie Cites))$$

1 Formulation and translation in SQL

```
select s.sid, b.bookno
from   student s, book b
where  not exists (select 1
                   from   buys t
                   where  t.sid = s.sid and
                          not exists (select 1
                                      from   cites c
                                      where  t.bookno = c.bookno and
                                             c.citedbookno = b.bookno));
```

Translation in SQL of the not exists set predicates

```
select sid, bookno
from   student, book
except
select sid, bookno
from   (select s.*, b.*, t.sid as tsid, t.bookno as tbno
        from   student s, book b, buys t
        where  s.sid = t.sid
        except
        select s.*, b.*, t.*
        from   student s, book b, buys t, cites c
        where  s.sid = t.sid and t.bookno = c.bookno and c.citedbookno = b.bookno) q;
```

2 Translation to RA as expressed in SQL

```
select sid, bookno
from   student cross join book
except
select sid, bookno
from   (select s.*, b.*, t.sid as tsid, t.bookno as tbno
        from   student s
               join buys t on (s.sid = t.sid)
               cross join book b
        except
        select s.*, b.*, t.*
        from   student s
               join buys t on (s.sid = t.sid)
               join cites c on (t.bookno = c.bookno)
               join book b on (c.citedbookno = b.bookno)) q;
```

3 Optimization

3.1 Using functional constraints

Observe that we have the following functional constraints.

$sid \rightarrow (sname)$	In Student relation
$bookno \rightarrow (title, price)$	In Book relation

These constraints permit us the "project-out" the sname, title and price attributes since they are not part of the output of the query. This gets us to the following query:[1]

```
select sid, bookno
from   (select sid from student) s
        cross join (select bookno from book) b
except
select sid, bookno
from   (select s.sid, b.bookno, t.sid as tsid, t.bookno as tbno
        from   (select sid from student) s
                join buys t on (t.sid = s.sid)
                cross join (select bookno from book) b
        except
        select s.sid, b.bookno, t.sid, t.bookno
        from   (select sid from student) s
               join buys t on (s.sid = t.sid)
               natural join cites c
               join (select b.bookno from book b) b on c.citedbookno = b.bookno) q;
```

[1] We have left out the `distinct` clause to aid with the readability of the query.

3.2 Utilizing foreign key constraints

Observe that we have the following foreign key constraints:

sid	foreign key in *Buys* referencing primary key in *Student*
citedbookno	foreign key in *Cites* referencing primary key in *Book*

Using these constraints, we can optimize the query to become the following.

```
select sid, bookno
from   (select sid from student) s
        cross join (select bookno from book) b
except
select sid, bookno
from   (select t.sid, t.bookno as tbno, b.bookno
        from  buys t
              cross join (select bookno from book) b
        except
        select t.sid, t.bookno, c.citedbookno
        from   buys t
               natural join cites c) q;
```

We get the RA expression:

$$\pi_{sid}(Student) \times \pi_{bookno}(Book) - \pi_{sid,bookno}(Buys \times \pi_{bookno}(Book) - \pi_{sid,bookno,citedbookno}(Buys \bowtie Cites)).$$