NOTE/WARNING: *The present note has been written in 2009 with the scope of publishing a UCD based query interface for XMM source mining. This service has never been made public and the note has never been published although this feature is implemented in SAADA. It has been pushed in the IVOA WIKI as it were at the time. (LM 10/2011)*

Query based on UCDs experienced with Saada

Laurent MICHEL (XMM SSC Strasbourg)
François Xavier PINEAU (XMM SSC Strasbourg)
laurent.michel@astro.unistra.fr (2009)

Introduction

The purpose of the present document is to relate our experience with the implementation of a query engine handling constraints expressed with UCDs. This development has been achieved in the frame of the Xcatdb (http://xcatdb.u-strasbg.fr). The Xcatdb is an interface for the 2nd XMM catalogue developed with Saada and deployed at Strasbourg in summer 2007. This query engine has been integrated to the Saada distribution. Saada using its own native query language (SaadaQL), the syntax presented here is not expected to be used by ADQL. But we believe that the reflection we had, could help the community for a future enhancement of ADQL toward the use of UCDs, UTypes and units.

Our use case: XMM-Newton data

The core of the 2XMM data is a collection of X-ray sources linked with a collection of archival sources extracted from Vizier (200 various catalogues), Ned and Simbad. Links are computed by the ACDS task of the XMM pipeline run by the Survey Science Consortium. They are implemented within the Xcatdb by using the Saada relationship feature.
Our problem was to propose a convenient way to select X-Ray sources by setting cross constraints on archival counterparts:

```
Select X-Ray sources linked with more than 1 archival source
              having a redshift greater than 1.
```

Processing such queries requires at some step to select sources with a redshift greater than one in all archival catalogues. As archival catalogues are quite heterogeneous (column names and units), we need a column name dictionary to select tables with a redshift column. UCDs provide an efficient way to implement such a dictionary.
Correlation links are persistent in the Xcatdb, but the following considérations could also be applied on multiple cross-matches.

NOTE/WARNING: *The present note has been written in 2009 with the scope of publishing a UCD based query interface for XMM source mining. This service has never been made public and the note has never been published although this feature is implemented in SAADA. It has been pushed in the IVOA WIKI as it were at the time. (LM 10/2011)*

UCDs versus UTypes

The first reflection we had was about either using UCDs or Utypes to map table columns.
In term of implementation, using UCDs or Utypes is slightly the same thing; the difference is more about semantic. Using Utypes supposes that data implement a data model. There is no data model enough complete to enclose all quantities contained in all catalogues and there is no certainty that all catalogues can be mapped in a given data model. In other words if constraints set by the use of a DM are very useful for interoperability, they have no interest for our purpose. UCDs are disconnected from any model. They are atomic (one UCD for one column). Using a UCD to build a query does not require extra knowledge. For these reasons, our query engine uses UCDs. The Saada query engine accepts however queries based on Utypes.

Unit Management

Data contained in our collection of catalogues are heterogeneous in term of column names but also in term of units. A same quantity can be expressed with different units. As we want to run queries covering multiple catalogues, we have either to homogenize units at loading time or to do a conversion on the fly. In both cases, we must handle unit conversions. The first solution has been discarded for 2 reasons: 1) We want to keep original data in our database ; 2) want to give users the possibility of using their preferred units. So unit conversion is done by the query engine.

Multiple UCDs Issue

Catalogues can have more than one column tagged with the same UCD. For instance, a lot of XMM data are related to one of the 3 cameras (PN, MOS1 or MOS2) in addition with a 4^{th} value which is the merge of the 3 others. The question is to know how to deal with queries on multiple columns. We can either select one column, but which one, or try to do with all columns matching the UCD.

Using all columns:

This solution is used by the XCatDB service. There are 2 ways to merge columns with the same UCD in a translated query. Column constraints can either be ANDed or ORed. There is no obvious answer to this question.
Let's supposed we want to select data with radial velocity greater than 10. (`src.veloc.hc > 10`)
Let's suppose that columns `V1` and `V2` have `src.veloc.hc` as UCD.
The 2 possible translations are:

1. ANDed `src.veloc.hc > 10 => (V1 > 10 AND V2 > 10)`

NOTE/WARNING: *The present note has been written in 2009 with the scope of publishing a UCD based query interface for XMM source mining. This service has never been made public and the note has never been published although this feature is implemented in SAADA. It has been pushed in the IVOA WIKI as it were at the time. (LM 10/2011)*

2. `ORed src.veloc.hc > 10 => (V1 > 10 OR V2 > 10)`

The most relevant translation depends on the use case. The user should be able to decide which strategy to apply but asking him such consideration could be confusing. When a catalogue has two columns with the same UCD, one of them can contain a lot NULL values. A NULL does not mean that that the value does not match the constraint, but that it has not been set.
The appropriate ANDed translation is

```
(V1 NULL OR V1 > 10) AND (V2 IS NULL OR V2 > 10) AND (V1 NOT
NULL OR V2 NOT NULL)
```

So that we are sure to apply the constraint on the non NULL values and to discard rows with both columns set as NULL. Facing the complexity of this translation (we can have more than 2 columns) we decided to use the ORed translation. That solves the NULL value issue but that raised another difficulty.
Let's consider the following constraint:

```
src.veloc.hc > 100 AND src.veloc.hc < 90
```

The ORed translaton gives

```
(V1 > 100 OR V2 > 100) AND (V1 < 90 OR V2 < 90)
```

This can be developed as:

```
(v1 > 10 AND v1 < 100)
OR
(v2 > 100 AND v2 < 100)
OR
(v1 > 10 AND v2 < 100)
OR
(v2 > 10 AND v1 < 100)
```

Terms 3 and 4 make no sense and alter significantly the query result!! The must be discarder by the query translator.

Even if such cases can be processed by the query engine (with difficulties) they are confusing for users because the result depends on hidden choices of implementation.
We finally opted for syntax restrictions to work around multiple UCDs problem. All constraints look like this:

```
[ucd] operator operand [unit]
```

The translator accepts UCD to be used at most once and the list of possible operators has been delimited as below.

Operator		**Operand**
=	Equals to	atomic
!=	Not equals to	atomic

NOTE/WARNING: *The present note has been written in 2009 with the scope of publishing a UCD based query interface for XMM source mining. This service has never been made public and the note has never been published although this feature is implemented in SAADA. It has been pushed in the IVOA WIKI as it were at the time. (LM 10/2011)*

>	Greater than	atomic
>=	Greater then or equals to	atomic
<	Less than	atomic
<=	Less than or equals to	atomic
`[]`	In range, bounds excluded	(value1, value2)
`[=]`	In range, bounds included	(value1, value2)
`][`	Out of range, bounds excluded	(value1, value2)
`]=[`	Out of range, bounds included	(value1, value2)

Both UCDs and Utypes are built with dot separated words exactly like table-columns association in SQL. To avoid confusion UCD are enclosed in [].

String Issue

Another important issue comes with columns with the same UCDs but different types.
Let's suppose that table T1 has a numerical column OBS_ID ranging from 1234 to 2345 and that table T2 has a string column OBSID ranging from '01234' to '02345'. Both columns have 'meta.id' as UCD.
The query constraint `meta.id > 1000` will be translated like this:

- Table T1: `OBS_ID > 1000`
- Table T2: `OBSID > '1000'`

As the operator '>' (greater than) uses a lexical sort for strings, the second query will return an empty set ('1000' always greater than '0*') whereas the first query will return all OBS_ID greater than 1000. The query translator must manage casting properly to avoid such errors:

UCD like constraint	**Translated in table T1**	**And in table T2**
`meta.id > 1000`	`OBS_ID > 1000`	`OBS-ID::int > 1000`
`meta.id > '1000'`	`OBS_ID::text > '1000'`	`OBSID > '1000'`

Numercical operands can rise SQLcasting error on tables with string values(e.g. what about `OBS-ID::int > 1000` if `OBS-ID` equals to `'QWERTY'`)

NOTE/WARNING: *The present note has been written in 2009 with the scope of publishing a UCD based query interface for XMM source mining. This service has never been made public and the note has never been published although this feature is implemented in SAADA. It has been pushed in the IVOA WIKI as it were at the time. (LM 10/2011)*

Implementation

```
Select ENTRY From CatalogueEntry In CATALOGUE
WhereRelation{
    matchPattern{CatSrcToArchSrc,
        Cardinality > 1,
        AssUCD{[phys.veloc] > 1000 [km/s]}
        }
    }
```