ADQL WD update

- ADQL Grammar is added in Appendix.
  - Construct
  - Identifier (rule for naming a table and a column)
  - Data type (numeric, character, date/time …)
  - Functions, operators, predicates
- Syntax is divided into two categories
  - Core syntax -- Skynode MUST conform to
  - Extension syntax
  - Specification number is assigned, and it is used to check conformity of the service.

Why Core and Extensions ?

- Minimize the effort for building a skynode.
  - Core syntax provides minimum functionalities:
  - Easy transition from a DAL to a SkyNode service.
- Maximize interoperability
  - It is unlikely that all the skynodes have the same level of conformity to the full ADQL syntax.
  - Use Core syntax for complete interoperability among all the skynodes.
- Extend capability of skynode
  - The skynode capability is easily extended by extension syntax.

Core Query Syntax

```
SELECT { [table_alias.]* | count(*) |
    [table_alias].column_name [[AS] alias] [,…] }
    FROM    table_name [AS] table_alias
    WHERE   condition
```

- Restriction to the standard SQL
  - No expression in the selection list.
    - Just return column values or the number of query result.
  - No multiple tables, No join in FROM clause.
    - One query for one table.
    - Table join is assumed to be done on upper level services, such as a skynode portal.
  - No other clauses like DISTINCT, ORDER BY, GROUP BY, HAVING …

Full Query Syntax

```
SELECT [ ALL | DISTINCT ] [ INTO table_name ]
    [ TOP number ] [ OFFSET number ]
    selection_list
    FROM from_item [,…]
    [ WHERE condition ]
    [ GROUP BY expression [,…] ]
    [ HAVING condition [,…] ]
    [ ORDER BY expression
        [ ASC | DESC | USING operator ] [,…] ]
```

SQL-92 + “INTO” + “TOP” + “OFFSET” + “#upload”
+ table name qualified by an archive name
- mandatory table alias

TOP and OFFSET (extensions)

- “TOP n” returns the first n-rows.
- “OFFSET n” skips the first n-rows. [new]
- Caution:
  - It is meaningless unless the order of the query result is not specified.
  - The order of the result may change query by query if it is not explicitly specified by ORDER BY clause.
- Use case:
  - Retrieve only the first 100 records by SELECT TOP 100
  - Retrieve the next 100 records by SELECT TOP 100 OFFSET 100.
  - …

UCD and UTYPE in selection list (extensions)

- Select columns based on a UCD name and UCD pattern matching
  - `SELECT UCD 'pos.eq.RA'`
  - `SELECT UCD 'phot.mag;em.opt.%'`
- It is not allowed to be used in the Where clause.
  - UCD may be mapped to multiple columns.
- XML representation
  -
    ```
    <item xsi:type="ucdSelectionItemType"
                table="t" ucd="pos.eq.RA"/>
    ```

Keyword and Identifier (Core)

- Keyword:
  - SELECT, FROM, WHER …
  - Case insensitive
- Identifier:
  - name of a table, a column and a function.
  - Restriction on ADQL-s
    - Must begins with a letter {a-z}, or an underscore.
    - Subsequent characters in an identifier can be letters, underscores or digits {0-9}.
    - Keyword is not allowed.
    - No restriction on the used character in ADQL-x
  - Case insensitive.

Delimited Identifier (Core)

- Used to allow for the use of keywords or special characters in naming the column and table.
- "[" and "]" are used as delimiters.
- Two adjacent brackets between the delimiters are taken as a single bracket character.
- Case sensitive: "CaseSensitive" attribute in ADQL-x
- Examples
  - 2mass → [2mass], [O/Fe] → [[[O/Fe]]]
  - M, m → [M], [m]

Data types

- Follows VOTable data types
- One of the following data types must be assigned to each column

  - **boolean**
  - **bit**
  - **unsignedByte**
  - **short**
  - **int**
  - **long**
  - **float**
  - **double**
  - **floatComplex**
  - **doubleComplex**
  - **char**
  - **char[n]**
  - **char***
  - **unicodeChar**
  - **Array (int[2], double[2]…)**
  - **timestamp**
  - **date**
  - **time**
  - **time interval**
  - **Space**

Timestamp literal expression

```
[timestamp] ‘2005-10-02 10:00:00+9’


<Literal xsi:type=“timestampType”>
    2005-10-02 10:00:00+9
</Literal>
```

Space data type

[Space] ‘Circle FK5 30.0 20.0 1.0 [*unit*]’

Coordinate Frame :

ICRS, FK5, FK4, J2000, B1950, GALACTIC

```
<Literal xsi:type="reg:searchLocationType">
   <stc:AstroCoordSystem>
      <stc:SpaceFrame><FK5/><GEOCENTER/></stc:SpaceFrame>
   </stc:AstroCoordSystem>
   <stc:AstroCoordArea>
      <stc:Region>
             <reg:Circle unit="deg">
                    <reg:Center>30.0 20.0</reg:Center>
                    <reg:Radius>1.0</reg:Radius>
             </reg:Circle>
      </stc:Region>
   </stc:AstroCoordArea>
</Literal>
```

Example usage of space data type

```
SELECT  access_URL
FROM    imageTable
WHERE   region overlaps
          ‘Box FK5 120 40 0.1 0.1’
```

Xmatch functions (Extension)

- A user is not satisfied with only one specific xmatch function.
- 2 or 3 xmatch functions are defined as standards
  - xmatch_chi2() : XMATCH() function of the previous specification.
  - xmatch_distance() : does xmatch based on the angular distance between two objects.
  - …
- Skynode may implement any specific xmatch function. That information should be provided through a "Functions" interface.

Table name qualified by an archive name (Extension)

- Use SHORTNAME or Identifier.

PhotoPrimary table of a service

ivo://archive.stsci.edu/hdfn/SKYNODE

→ HDF:photoprimary (shortname)

or

archive.stsci.edu:hdfn.skynode.photoprimary (identifier)

Returned VOTable (SkyNode Interface ?)

- The order of the fields must be the same as the order in the selection list.
- Column name must be filled in the NAME attributes of a FIELD element.
  - If an alias is specified in the query, the alias name must be filled.
  - If alias is not specified
    - If the field is a column or a function, the column name or function name qualified by the table alias name must be filled. → tableAlias.columnName
    - If the field is an expression, it may be empty

Date/Time data type

- literal syntax

```
[ timestamp | date | time | time interval ]
                                   ‘SQL expression’


[date] ‘2005-10-24’
[date] ‘20051024’
[time] ‘10:20:08.25’
[time] ‘10:20:08’
[time] ‘10:20’
[timestamp] ‘2005-10-20 04:30:21’
[time interval] ‘1 day 12 hours 59 min 10 sec’
```