JVO Skynode Implementation Experience

Yuji SHIRASAKI

National Astronomical Observatory of Japan, JVO

VOQL session

Contents

- Introduction of JVO SkyNode toolkit
  - used free software
  - architecture of JVO skynode
- Problems in implementation and interoperability
  - XML → Java deserialization problem in AXIS
  - Namespace problem ADQL, VOTable, STC
  - Usage of VOTable → id, name attributes …
  - Complexity of ADQL and STC object
  - …
- Proposal
  - Simplify the ADQL and STC → Define minimum subset of ADQL and STC and freeze them (never update, never change the namespace)
  - VOTable transfer → attachment or URL
  - Standardize the error message (not presented, as a future work)
  - …

Development of the JVO SkyNode Toolkit

- Primary aim:
  - to provide a reference implementation for every kind of data service which uses ADQL & VOTable interface
- Supported DBMS:
  - aimed to be independent on the type of DBMS
  - The only requirement is availability of JDBC driver.
  - but still PostgreSQL native SQL (copy command) is used...
- Restrictions:
  - Not all the ADQL syntax are supported.
  - String representation of ADQL is JVOQL.
- Experimental Release:
  - http://jvo.nao.ac.jp/download/skynode-toolkit/

What can be done with the toolkit ?

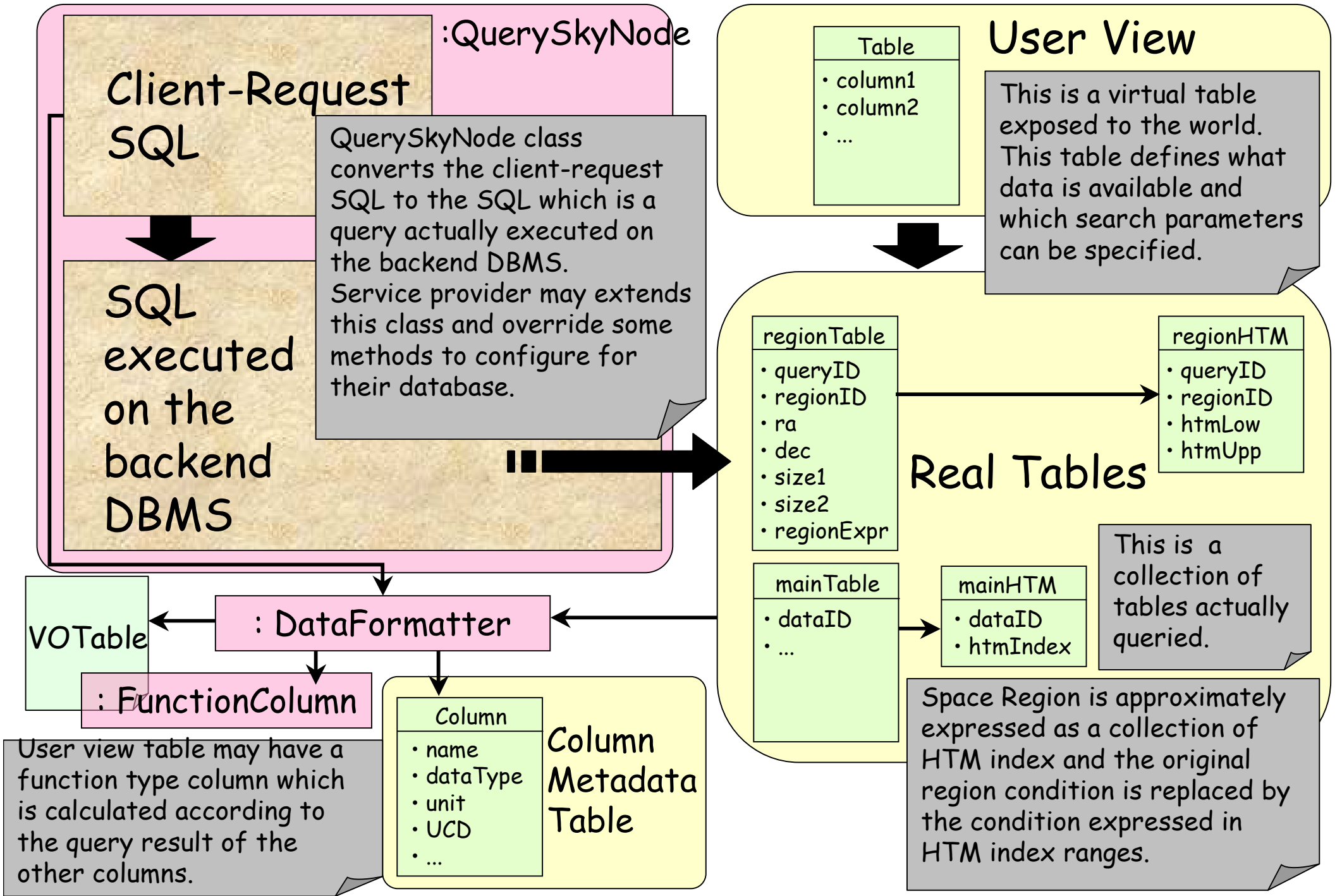
- Catalog data query
- Catalog data cross match query using VOTable
- Image data query
- Image data cross match query using VOTable
- Spectrum data is not supported, but the frame work will be the same as that of Catalog and Image. → next work
- You can build a sample SkyNode service.

Software used

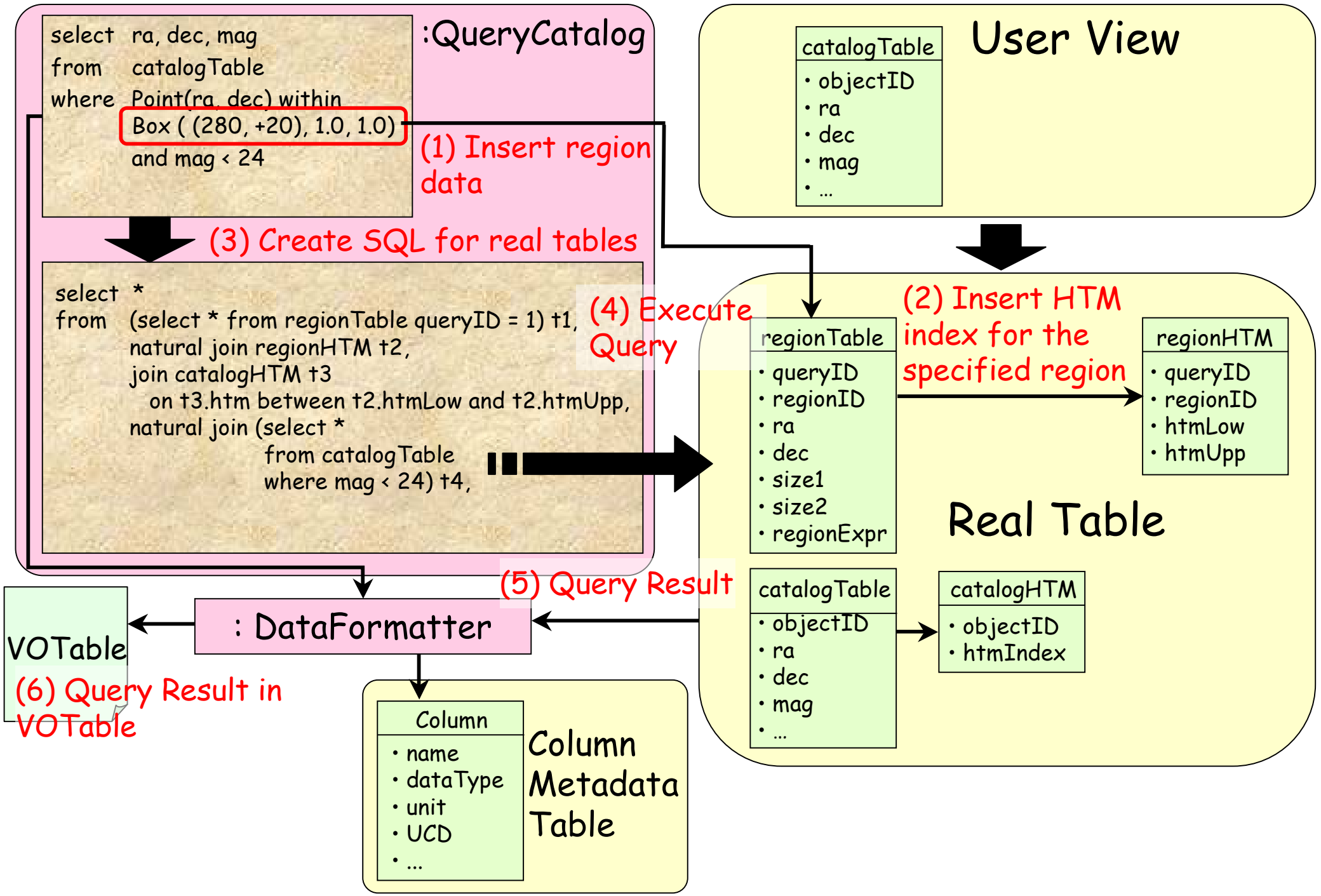
- Tomcat 4.1.31 ---- servlet container
- Axis 1.2RC1 ---- web service engine
- J2SDK 1.4.2 ---- Java compiler & library
- Ant 1.6.1 ---- Java-based build tool
- JavaCC 3.2 ---- parser generator for Java
- JAXB v1.0.3-b18-fcs ---- XML↔Java conversion
- PostgreSQL 7.4.7 ---- DBMS
- Java HTM library (JHU) ---- spherical indexing
- Java FITS library (HEASARC) ---- FITS IO lib
- …

Architecture

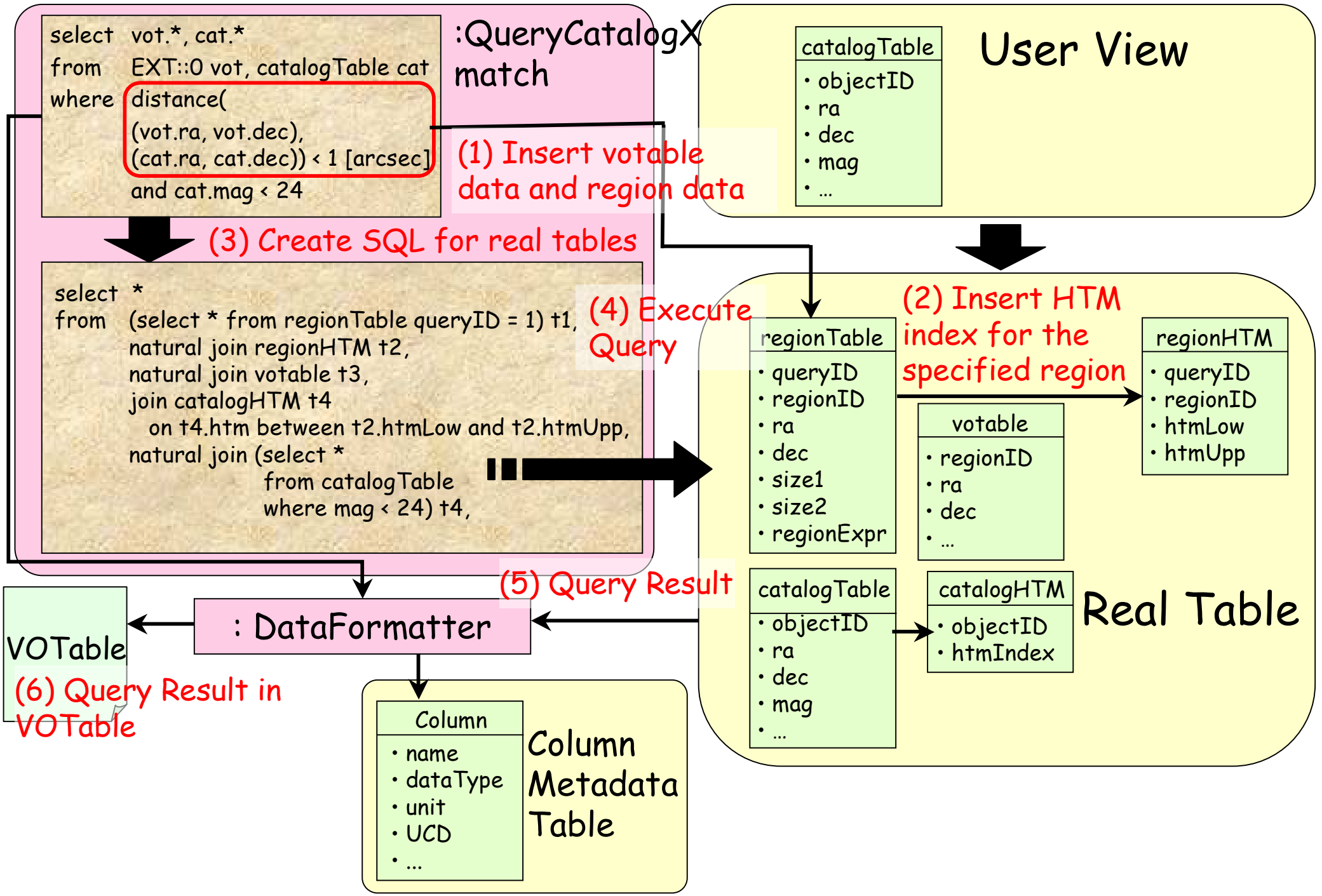
JVO SkyNode Toolkit Architecture



Catalog Data Query by ADQL



Catalog Data Xmatch Query with VOTable



Problem encountered in implementation and problem for interoperability

Problems encountered in implementation (1)

- XML⇔ Java auto-conversion in AXIS
  - With the standard usage of AXIS, creation of a Java object corresponding to a XML document (e.g. VOTable) is, as a default, required.
  - Server memory is easily exhausted.
  - Even several hundreds records of VOTable suffers from out of memory error.
- Possible Solution :
  - Don't use the auto-serialization and deserialization mechanism of AXIS (suggestion from an AstroGrid person).
  - return VOTable as an attachment
  - return a reference URL to retrieve the VOTable

Problem in implementation (2)

- Usage of VOTable is not clearly defined
  - id and name attribute → what should be filled ?
  - Where column alias name should be put. This information might be used for post-search processing on portal side.
  - Location where a table name and an alias table name are put.
  - Information on the origin of the column data should be kept anywhere in VOTable.

Problems for interoperability (1)

- Name space problem (as of 2005 Jan)
  - JVO → ADQL v0.8 + VOTable v1.1 + STC v1.1
  - NVO → ADQL v0.74 + VOTable <v1.0 + NVO-STC
- Temporal workaround
  - External interface → ADQL v0.74, VOTable v1.0
  - Internal interface → ADQL v0.8, VOTable v1.1
  - Namespace exchanger
- Complexity of ADQL and STC object
  - ADQL → 33 elements, 69 types
  - STC → 250 elements, 88 types
- Possible Solutions:
  - Define a core part of ADQL as a minimum subset and assign a permanent namespace. Never update, never change the namespace of the core part.

Minimum subset of ADQL

Element: 33 (full) → 12 (basic)

Simple Type: 13 (f) → 4 (b)

Complex Type: 56 (f) → 12 (b)

Fundamentral Type

xs:unsignedInt
xs:string(*)
xs:double(*)
xs:long(*)

Simple Type

aggregateFunctionNameType
allOrDistinctType
binaryOperatorType
comparisonType(*)
jointTableQualifierType
mathFunctionNameType
orderDirectionType
trigonometricFunctionNameType
unaryOperatorType

Element

Allow
Arg(*)
Column
Condition(*)
EndComment
Expression(*)
From(*)
GroupBy
Having
InTo
Item(*)
Literal(*)
Name
Nature
Order
OrderBy
Params
Pattern
Qualifier
Region(*)
Restrict
Select(*)
SelectionList(*)
Set
Sigma
StartComment
Table(*)
TableName
Tables
Unit(*)
Where(*)
fromTableType
selection

Complex Type

ArrayOfFromTableType
ConstantListSet
aggregateFunctionType
aliasSelectionItemType(*)
allSelectionItemType
archiveTableType
atomType(*)
betweenPredType
binaryExprType
closedExprType
closedSearchType
columnReferenceType(*)
comparisonPredType(*)
dropTableType
exclusiveSearchType
fromTableType
fromType(*)
functionType
groupByType
havingType
includeTableType
inclusionSetType
inclusiveSearchType
integerType(*)
intersectionSearchType(*)
intoType
inverseSearchType
joinTableType
likePredType
literalType(*)
mathFunctionType
notBetweenPredType
notLikePredType
numberType
orderExpressionType
orderOptionType
orderType
realType(*)
regionSearchType
scalarExpressionType
searchType
selectType
selectionItemType
selectionLimitType
selectionListType
selectionOptionType
stringType(*)
subQuerySet
tableType(*)
trigonometricFunctionType
unaryExprType
unionSearchType
userDefinedFunctionType
whereType(*)
xMatchTableAliasType
xMatchTyp

ComparisonPredType as a "must-support" ConditionType

- A basic SkyNode MUST support "comparisonPredType" .
- A basic SkyNode MAY support the other searchType.
- A basic SkyNode MUST recognize an unsupported searchType as a "trueType".
- A basic SkyNode MUST support the following construct:
  - <STC_columnName> <STC_operator> <STC_searchLocationType>
  - STC_operator ::= within, overlaps, outside
    - point within STC('Circle ICRS 200 -20 2.0') (Catalog Query)
    - region overlaps STC('Circle ICRS 200 -20 2.0') (Image Query)
    - spectrum overlaps STC('SpectralInterval A 4000 7000') and observationTime within STC('TimeInterval 2004-05-01 2004-05-31') (Specturm Query)

STC-type column must be compared with an object of STC seachLocationType

utype defines which column represents STC.

ADQL-x version of point within 'Circle ICRS 200 -20 2.0'

It might be convenient to define substitution types for frequently used frames such as <SpaceFrame xsi:tye="ICRSFrameType"/>

```
<Condition comparison="within" xsi:type="comparisonPredType">
  <Arg Table="t" utype="src.position" xsi:type="columnReferenceType"/>
  <Arg xsi:type="searchLocationType">
    <AstroCoordSystem ID="ICRS">
      <SpaceFrame>
        <ICRS/><BARYCENTER/><SPHERICAL coord_naxes="2"/>
      </SpaceFrame>
    </AstroCoordSystem>
    <AstroCoordArea ID="SearchRegion" coord_system_id="ICRS">
      <Region>
        <Circle unit="deg"> <Center>200 -20</Center> <Radius>2.0</Radius>
        </Circle>
      </Region>
    </AstroCoordArea>
  </Arg>
</Condition>
```

Which frame should be defined as a "must be supported" frame

Problem for interoperability (2)

- Column name must be known in advance for writing ADQL.
- We can get column names by "Columns" interface and write ADQL, but it requires human intervention.
- A possible solution:
  - use UCD or Utype for specifying a column
  - Introduce "ucd" and "utype" attributes to the columnReferenceType

<Item xsi:type="columnReferenceType" Name="ra" Table="qso"/>

<Item xsi:type="columnReferenceType" ucd="pos.eq;src" Table="qso"/>

<Item xsi:type="columnReferenceType" utype="Target.pos" Table="qso"/>

  - If the specified utype or ucd is not found in the queried table,
    → ignore the condition for that column
    → return PARAMETER of "NaN" for that column

Summary

- Experimental release of JVO SkyNode toolkit
  - http://jvo.nao.ac.jp/download/skynode-toolkit/
  - Support for Catalog query, Image query
- Some Proposals
  - Need minimum subset of ADQL and STC
  - Minor update on ADQL: ucd and utype attributes to the ColumnReferenceType.
  - Usage of VOTable. Location where column name, column alias name, table name and table alias name are described.
  - Error message (for future work)