P1.2

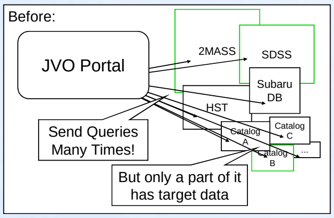
Construction of Multiple-Catalog Database for JVO

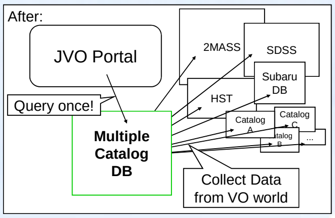
Masahiro Tanaka, Yuji Shirasaki, Masatoshi Ohishi, Yoshihiko Mizumoto (National Astronomical Observatory of Japan) ,
Yasuhide Ishihara, Jumpei Tsutsumi, Yoshihiro Machida (Fujitsu Ltd.),
Hiroyuki Nakamoto, Yuusuke Kobayashi, Michito Sakamoto (SEC Ltd.)

URL: http://jvo.nao.ac.jp/index-e.html

Introduction and Motivation

In a case where a VO user collects all the available information about an astronomical object, it is necessary to send queries to all the VO services. However, this would be inefficient since most services tend not to have data on the target object. We thus developed a new query system for users to efficiently find all the available information from multiple catalogs.





Design of Table Partitioning

The design of this database is one of key issues; an efficient query mechanism for more than billions of objects is required. We employed the Table-Partitioning technique and developed a method to build queries for the partitioned tables.

Table design:
Partitioning with HTM (Hierarchical Triangular Mesh)
Upper level : level 6 : $8 \times 4^6 = 32768$ segments
Tables are partitioned by Upper HTM level : 32768 tables

```
 Schema |    Name     | Type
--------+-------------+-------
 public | psc_32768 | table
 public | psc_32769 | table
 ...
 public | psc_65534 | table
 public | psc_65535 | table
```

A Region Query:

```
select ra, dec, j_m
 from psc
 where Region('Circle 0 0 10');
```

is converted to the following SQL:

```
select ra, dec, j_m
from ( select * from psc_63488 where htm_id between 0 and 4194303
 union select * from psc_63488 where htm_id between 13631488 and 16777215
 union select * from psc_47104 where htm_id between 0 and 4194303
 union select * from psc_47104 where htm_id between 13631488 and 16777215
  ...
) psc;
```

Performance of Table Partitioning

Comparison with PostgreSQL's partitioning function

PostgreSQL is equipped with Table Partitioning function after the version 8.1. We compare the performance of our method with PostgreSQL's partitioning function. For this test, we used 2MASS All Sky Catalog including 4.7 billion objects.

Note: PostgreSQL cannot handle 32768 tables at once since it consume too much shared memory. Therefore, we define the partitioning level as 2048 tables.

Measured elapsed time:

Search radius	Result objects	Elaspted time (sec)			# of HTM conditions	
arcmin	#	Postgre SQL	Our method	ratio	Postgre SQL	Our method
1	2	6.460	0.042	154	32	32
10	165	3.807	0.030	127	16	16
60	6697	6.468	0.107	60	32	32
100	26720	2.016	0.307	6.6	4	16
180	57246	9.044	0.709	12.8	48	72

We found our method is **6-150 times faster**. This result shows that the partitioning with HTM is quite useful for searching a large database for some celestial object.

Design of Multiple Catalog DB

For the multiple catalog DB we employed a simple FLUX-based format shown in the right.

It is another key issue to store catalogs of various data format into a table of single format. In this format, each record has only one FLUX column, which is different from the conventional OBJECT-based format where object records commonly have multiple-band flux data. In our multiple catalog DB, these flux data are decomposed and stored in separated records. Multi-wavelength information of objects such as the SED (Spectral Energy Distribution) is obtained by grouping the flux records in terms of position. Further information about the object can be obtained by accessing the original catalog shown in the "link_ref" column.

Table design of Multiple Catalog DB

column	description
Object information	
id	Object ID
name	Object name
Position information	
ra	Right Ascension
dec	Declination
pos_err	Position error
htm	HTM index
Wavelength information	
band_name	band name
band_unit	band unit
Flux information	
flux	flux
flux_err	flux error
flux_unit	flux unit
flux_srch	flux in Jy
Reference information	
link_ref	Link to reference
org_id	ID in original catalog
cat_id	Catalog ID

New User Interface of JVO Portal

URL: http://jvo.nao.ac.jp/dev/portal/