How I Learned to use the Skyserver

Robert Lupton

Princeton, July 2002

I like to access the skyserver using my emacs-lisp mode; the latest version is `v1_10` and is available from

`http://www.astro.princeton.edu/~rhl/skyserver`

There's even a manual. Warning: older versions don't support the full use of @variables.

You may prefer to use some other interface; in particular Jim Gray and Alex Szalay like the *Microsoft Query Analyser* which supports full Transact-SQL.

A Simple Query

```
select distinct
    run, rerun,
    field0, field0 + nfields - 1,
    photoVersion
from Segment
```

```
run rerun field0  photoVersion

94 7 104 544 v5_2_21
125 7 11 451 v5_2_21
752 8 11 617 v5_2_5
756 8 196 802 v5_2_6
1336 2 11 95 v5_2_12
1339 2 11 95 v5_2_12
1356 2 20 113 v5_2_12
1359 3 19 113 v5_2_12
```

Note the ugly formatting, and that the last field (the one calculated as `field0 + nfields - 1`) has no column heading.

A Formatted Simple Query

```
select distinct
    str(run, 4) as run, str(rerun, 3) as rerun,
    str(field0,3) as field0, str(field0 + nfields - 1,4) as field1,
    photoVersion
from Segment
order by run
```

```
run rerun field0 field1 photoVersion

  94    7 104   544 v5_2_21
 125    7  11   451 v5_2_21
 752    8  11   617 v5_2_5
 756    8 196   802 v5_2_6
1336    2  11    95 v5_2_12
1339    2  11    95 v5_2_12
1356    2  20   113 v5_2_12
1359    3  19   113 v5_2_12
```

Prettier and `field1` is no longer anonymous. We've made the ordering by run explicit.

A Simple Egocentric Query

```
--
-- What's loaded in skyserver?
--
declare @database char  set @database = Robert
--
select distinct
    str(run, 4) as run, str(rerun, 3) as rerun,
    str(field0,3) as field0, str(field0 + nfields - 1,4) as field1,
    photoVersion
from @database..Segment
order by run
```

```
run rerun field0 field1 photoVersion

 745 672 514  514 v5_3_31
 756 672 796  796 v5_3_31
1336  14  93   93 v5_3_31
1339  14  92   92 v5_3_31
```

The same as before, except that this time I wanted to know what was loaded in *my* database.

An Almost-as-Simple Query

```
--
-- What's loaded in skyserver?
--
declare @database char  set @database =
--
select distinct top 2
    str(run, 4) as run, str(rerun, 3) as rerun,
    str(field0,3) as field0, str(field0 + nfields - 1,4) as field1,
    photoVersion
from @database..Segment
where
    run != 745
order by run
```

```
run rerun field0 field1 photoVersion

  94    7 104  544 v5_2_21
 125    7  11  451 v5_2_21
```

Omit run 745, and only show the first two rows found by the query. We're interrogating the main (non-Robert) database this time.

The Galaxy Target Selection Algorithm

```
declare @database char      set @database =
declare @pi float      set @pi = 3.141592654
--
declare @BLENDED int      set @BLENDED = dbo.fPhotoFlags('BLENDED')
declare @BRIGHT int      set @BRIGHT = dbo.fPhotoFlags('BRIGHT')
declare @EDGE int      set @EDGE = dbo.fPhotoFlags('EDGE')
declare @NODEBLEND int      set @NODEBLEND = dbo.fPhotoFlags('NODEBLEND')
declare @SATURATED int      set @SATURATED = dbo.fPhotoFlags('SATURATED')
--
declare @bad_flags int      set @bad_flags = (@SATURATED | @BRIGHT | @EDGE)
--
declare @maglim float      set @maglim = 17.77
declare @SBlim float      set @SBlim = 24.5
declare @delta_psf_model float      set @delta_psf_model = 0.3
--
select top 10
--  Standard fields
run, rerun, camCol, field,
str(rowc,6,1) as rowc, str(colc,6,1) as colc,
str(dbo.fObjFromObjID(ObjId), 4) as id,
':'  as ':',
--
--  Scientific output
--
str(ra,9,4) as ra, str(dec,8,4) as dec
--
from
```

```
from
    @database..PhotoPrimary
where
    --  Our star-galaxy separation and target selection
    psfMag_r - modelMag_r >= @delta_psf_model and
    petroMag_r - reddening_r <= @maglim and
    petroMag_r - 2.5*log10(2*@pi*petroR50_r*petroR50_r) < @SBlim and
    --  Check flags
    (flags & @bad_flags) = 0 and
    (((flags & @BLENDED) = 0) or ((flags & @NODEBLEND) != 0))
```

```
run rerun camCol field rowc colc : ra dec

1336 2 1 11  187.4 1635.3 :  251.2860  64.2985
1336 2 1 11  379.6  716.0 :  251.0971  64.2357
1336 2 1 11  542.9  327.7 :  251.0266  64.2010
1336 2 1 11  661.3  196.4 :  251.0098  64.1830
1336 2 1 11  873.6  811.6 :  251.1732  64.1913
1336 2 1 11  250.6  523.0 :  251.0389  64.2393
1336 2 1 11  261.0 1540.5 :  251.2723  64.2867
1336 2 1 11  324.7  653.6 :  251.0768  64.2382
1336 2 1 11  338.3 1052.6 :  251.1693  64.2558
1336 2 1 11  392.4 1749.8 :  251.3346  64.2836
```

Query both the Spectroscopy and Photometry

```
/*
 * Galaxy target selection with spectroscopic redshifts
 */
declare @database char      set @database =
declare @pi float      set @pi = 3.141592654
--
declare @BLENDED int      set @BLENDED = dbo.fPhotoFlags('BLENDED')
declare @BRIGHT int      set @BRIGHT = dbo.fPhotoFlags('BRIGHT')
declare @EDGE int      set @EDGE = dbo.fPhotoFlags('EDGE')
declare @NODEBLEND int      set @NODEBLEND = dbo.fPhotoFlags('NODEBLEND')
declare @SATURATED int      set @SATURATED = dbo.fPhotoFlags('SATURATED')
--
declare @bad_flags int      set @bad_flags = (@SATURATED | @BRIGHT | @EDGE)
--
declare @maglim float      set @maglim = 17.77
declare @SBlim float      set @SBlim = 24.5
declare @delta_psf_model float      set @delta_psf_model = 0.3
--
select top 10
str(gal.ra,9,4) as ra, str(gal.dec,8,4) as dec,
'|' as '|', cast(spec.type as char (9)) as type,
str(spec.z,7,4) as Z,
dbo.fSpecZStatusN(spec.zStatus) as status,
dbo.fGetUrlSpecImg(spec.specObjID) as Spectra
--
from
```

```
from
    @database..PhotoPrimary as gal,
    @database..specObj as spec
where
    gal.objID = spec.objID and
    --  Our star-galaxy separation and target selection
    psfMag_r - modelMag_r >= @delta_psf_model and
    petroMag_r - reddening_r <= @maglim and
    petroMag_r - 2.5*log10(2*@pi*petroR50_r*petroR50_r) < @SBlim and
    --  Check flags
    (flags & @bad_flags) = 0 and
    (((flags & @BLENDED) = 0) or ((flags & @NODEBLEND) != 0))
```

```
ra dec | type Z status Spectra
 251.3749  64.1473 | GALAXY     0.0689 XCORR_EMLINE http:...
 251.2649  64.0215 | GALAXY     0.0677 XCORR_EMLINE http:...
 251.5386  63.8977 | GALAXY     0.0695 XCORR_EMLINE http:...
 251.5323  63.7413 | STAR_BHB  -0.0001 XCORR_EMLINE http:...
 251.6382  63.6581 | GALAXY     0.1048 XCORR_HIC http:...
 252.0256  63.5788 | GALAXY     0.0333 XCORR_HIC http:...
 251.8941  63.4230 | GALAXY     0.0967 INCONSISTENT http:...
 252.3079  63.1463 | GALAXY     0.1045 INCONSISTENT http:...
 252.6909  62.8880 | GALAXY     0.0681 XCORR_HIC http:...
 252.4744  62.8420 | GALAXY     0.0366 XCORR_EMLINE http:...
```

There's a problem with this query; it doesn't return the objects which passed galaxy target selection but for which we have no spectrum.

There are Two Ways to write that Query

```
from
    @database..PhotoPrimary as gal,
    @database..specObj as spec
where
    gal.objID = spec.objID and
    --  Our star-galaxy separation and target selection
```

and

```
from
        @database..PhotoPrimary as gal
    join
        @database..specObj as spec
    on
        gal.objID = spec.objID
where
    --  Our star-galaxy separation and target selection
```

The two are exactly equivalent; one uses `join ... on` whereas the other prefers to put the `on` condition into the `where` clause. We're about to use a generalisation of this `join` syntax to include non-targetted galaxies in our outputs.

```
/*
 * Galaxy target selection including spectroscopic redshifts where available
 */
declare @database char      set @database =
declare @pi float      set @pi = 3.141592654
--
declare @BLENDED int      set @BLENDED = dbo.fPhotoFlags('BLENDED')
declare @BRIGHT int      set @BRIGHT = dbo.fPhotoFlags('BRIGHT')
declare @EDGE int      set @EDGE = dbo.fPhotoFlags('EDGE')
declare @NODEBLEND int      set @NODEBLEND = dbo.fPhotoFlags('NODEBLEND')
declare @SATURATED int      set @SATURATED = dbo.fPhotoFlags('SATURATED')
--
declare @bad_flags int      set @bad_flags = (@SATURATED | @BRIGHT | @EDGE)
--
declare @maglim float      set @maglim = 17.77
declare @SBlim float      set @SBlim = 24.5
declare @delta_psf_model float      set @delta_psf_model = 0.3
--
select top 10
str(gal.ra,9,4) as ra, str(gal.dec,8,4) as dec,
'|' as '|', cast(ISNULL(spec.type, 'NULL ') as char (9)) as type,
ISNULL(str(spec.z,7,4), 'NULL ') as Z,
ISNULL(dbo.fSpecZStatusN(spec.zStatus), 'NULL ') as status,
ISNULL(dbo.fGetUrlSpecImg(spec.specObjID), 'NULL ') as Spectra
--
from
```

```
from
        @database..PhotoPrimary as gal
    left outer join
        @database..specObj as spec
    on
        gal.objID = spec.objID
where
    --  Our star-galaxy separation and target selection
    psfMag_r - modelMag_r >= @delta_psf_model and
    petroMag_r - reddening_r <= @maglim and
    petroMag_r - 2.5*log10(2*@pi*petroR50_r*petroR50_r) < @SBlim and
    --  Check flags
    (flags & @bad_flags) = 0 and
    (((flags & @BLENDED) = 0) or ((flags & @NODEBLEND) != 0))
```

Here the `left outer join` includes rows for which no spectrum is available, returning `NULL`. The `select` has to handle these `NULL`s.

The phrase

```
ISNULL(str(spec.z,7,4), 'NULL')
```

isn't actually legal ANSI SQL (it's a SQL-server extension); the legal version is the rather wordier

```
(case when spec.z is NULL then 'NULL' else str(spec.z,7,4) end)
```

ra	dec	type	Z	status	Spectra
251.2860	64.2985	NULL	NULL	NULL	NULL
251.0971	64.2357	NULL	NULL	NULL	NULL
251.3141	63.9971	NULL	NULL	NULL	NULL
251.3144	63.9881	NULL	NULL	NULL	NULL
251.3085	63.9892	NULL	NULL	NULL	NULL
251.5633	64.0398	NULL	NULL	NULL	NULL
251.3749	64.1473	GALAXY	0.0689	XCORR_EMLINE	http:...
251.2649	64.0215	GALAXY	0.0677	XCORR_EMLINE	http:...
251.4962	64.0124	NULL	NULL	NULL	NULL
251.3497	63.9756	NULL	NULL	NULL	NULL

A Query that (Implicitly) Creates a Temporary Table

```
--
--  Make a temporary table
--
declare @database char     set @database = Robert
--
select
    *
from
    (select
        run,
        rerun,
        count (*) as nobj
    from
        @database..photoObj
    group by run, rerun
    ) as "tmpTable"
```

```
run rerun nobj

1336 14 581747
1339 14 608460
745 672 887579
756 672 907579
```

Make a temporary table (called `tmpTable`) and select its contents. This isn't actually exactly a temporary table in the usual SQL-server sense, but I find it helpful to think of it that way.

The `count(*) as nobj` says return the total number of rows in the `@database..photoObj` table; the `group by run, rerun` says that the counting should be done separately for each pair of values (`run, rerun`).

Another version of "What's Loaded"

```
--
-- What's loaded in skyserver?
--
declare @database char  set @database = Robert
--
select distinct
    str(seg.run, 4) as run, str(seg.rerun, 3) as rerun,
    str(field0,3) as field0, str(field0 + nfields - 1,4) as field1,
    str(nobj, 7) as nobj,
    photoVersion
from
    @database..Segment as seg,
    (select
        count (*) as nobj, run, rerun
    from
        @database..photoObj
    group by run, rerun
    ) as "fieldSummary"
where
    seg.run = fieldSummary.run and seg.rerun = fieldSummary.rerun
order by run
```

```
run rerun field0 field1 nobj photoVersion

 745 672 514  514  887579 v5_3_31
 756 672 796  796  907579 v5_3_31
1336  14  93   93  581747 v5_3_31
1339  14  92   92  608460 v5_3_31
```

We've included the number of objects in each run in the output.

Fix the field1 Values Using Another Temporary Table

```
--
-- What's loaded in skyserver?
--
declare @database char  set @database = Robert
--
select distinct
    str(seg.run, 4) as run, str(seg.rerun, 3) as rerun,
    -- str(field0,3) as field0, str(field0 + nfields - 1,4) as field1,
    str(field0_rhl,3) as field0,
    str(field1_rhl,3) as field1,
    photoVersion
from
    @database..Segment as seg,
    (select
        min (field) as field0_rhl,
        max (field) as field1_rhl,
        run, rerun
    from
        @database..field
    group by run, rerun
    ) as "runSummary"
where
    seg.run = runSummary.run and seg.rerun = runSummary.rerun
order by run
```

```
run rerun field0 field1 photoVersion

 745 672 395 514 v5_3_31
 756 672 680 796 v5_3_31
1336  14  11  93 v5_3_31
1339  14  11  92 v5_3_31
```

```
--
--  Calculate the žhed point from the skyserver data
--
declare @database char set @database = Robert
--
declare @maglim int     set @maglim = 18
declare @star int     set @star = dbo.fPhotoType('star')
declare @bad_flags int    set @bad_flags = (dbo.fPhotoFlags('SATURATED') | \
            dbo.fPhotoFlags('BRIGHT') | dbo.fPhotoFlags('EDGE'))
--  Possible restriction on runs to be processed
declare @select_run char set @select_run = --  run == 745 and
--
--  Our Query
--
select
field.pspStatus,
blue.run, blue.camCol, blue.field,
N,
gr as zhed_gr,
C0 + gr*C1 as zhed_ri,
C0, C1
from
```

```
@database..field as field,
(select                 --  Fit straight line
    run as run, camCol as camCol, field as field, fieldId, N,
    (-sum_x*sum_xy + sum_xx*sum_y)/(n*sum_xx - sum_x*sum_x) as C0,
    (N*sum_xy - sum_x*sum_y)/(n*sum_xx - sum_x*sum_x) as C1
from
(select                 --  Blue part of locus
    run as run, camCol as camCol, field as field, fieldId,
    count (*) as N,
    sum (g - r) as sum_x,
    sum ((g - r)*(g - r)) as sum_xx,
    sum (r - i) as sum_y,
    sum ((g - r)*(r - i)) as sum_xy
from
    (select
        run, camCol, field, fieldId,
        (psfMag_g - reddening_g) as g,
        (psfMag_r - reddening_r) as r,
        (psfMag_i - reddening_i) as i
    from @database..PhotoPrimary
    where
        @select_run
        (flags & @bad_flags) = 0 and nchild = 0 and
        type = @star and
        --  Not too faint
        psfMag_i < @maglim
    ) as "obj1"
where
    --  Choose stars in nearly horizontal part of g-r-i diagram
    g - r between 0.3 and 1.1 and
    r - i between -0.1 and 0.6
group by run, camCol, field, fieldId
) as "_blue"
) as "blue",
```

```
    (
    select              --  Red part of locus
        fieldId,
        avg(g - r) as gr--  I'd prefer the median
    from
        (select
            fieldId,
            (psfMag_g - reddening_g) as g,
            (psfMag_r - reddening_r) as r,
            (psfMag_i - reddening_i) as i
        from @database..PhotoPrimary
        where
            @select_run
            (flags & @bad_flags) = 0 and nchild = 0 and
            type = @star and
            --  Not too faint
            psfMag_i < @maglim
        ) as "obj2"
    where
        --  Choose stars in vertical part of g-r-i diagram
        g - r between 1.1 and 1.6 and
        r - i between 0.8 and 1.4
    group by fieldId
    ) as "red"
where
    blue.fieldId = red.fieldId and
    blue.fieldId = field.fieldId
order by blue.run, blue.camCol, blue.field
```

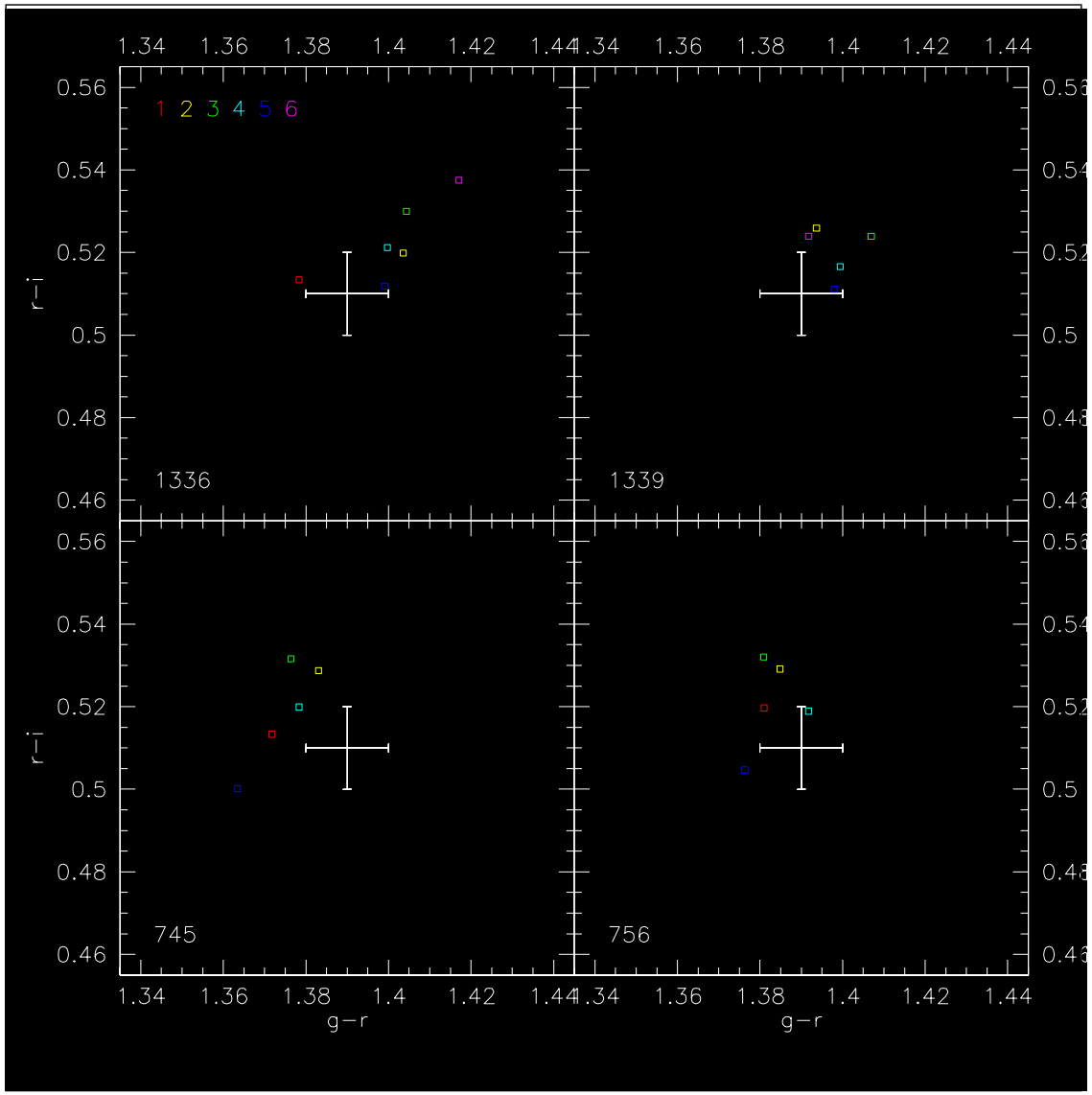
```
pspStatus run camCol field N zhed_gr zhed_ri C0 C1

0 745 1 395 22 1.364067 0.549448 0.003595 0.400166
0 745 1 396 33 1.381796 0.469429 0.001625 0.338548
0 745 1 397 40 1.329576 0.499432 -0.009701 0.382928
0 745 1 398 37 1.434749 0.538586 0.003014 0.373286
0 745 1 399 26 1.380334 0.509576 -0.023451 0.386158
0 745 1 400 36 1.361475 0.513623 -0.011265 0.385528
...
```

But what if I want an average for each run/camCol? Ask and ye shall receive...

```
--
--  Calculate the zhed point from the skyserver data
--
declare @database char      set @database = --  Robert
declare @Nmin int      set @Nmin = 0 --  Minimum number of stars/field
--
declare @maglim int      set @maglim = 18
declare @star int      set @star = dbo.fPhotoType('star')
declare @bad_flags int      set @bad_flags = (dbo.fPhotoFlags('SATURATED') |          db
| dbo.fPhotoFlags('EDGE'))
declare @good char      set @good = (1=1 or field.pspStatus = 0) and blue.N > @Nmin
--  Possible restriction on runs to be processed
declare @select_run char set @select_run = --  run = 1339 and camCol = 1 and
--
--  Our Query
--
select
blue.run, blue.camCol,
sum (case when @good then blue.N    else 0 end ) as N,
avg(case when @good then gr    else NULL end ) as zhed_gr,
avg(case when @good then C0 + gr*C1 else NULL end ) as zhed_ri
from
```

```
@database..field as field,
(select                 --  Fit straight line
    run as run, camCol as camCol, field as field, fieldId, N,
    (-sum _x*sum _xy + sum _xx*sum _y)/(n*sum _xx - sum _x*sum _x) as C0,
    (N*sum _xy - sum _x*sum _y)/(n*sum _xx - sum _x*sum _x) as C1
from
(select                 --  Blue part of locus
    run as run, camCol as camCol, field as field, fieldId,
    count (*) as N,
    sum (g - r) as sum _x,
    sum ((g - r)*(g - r)) as sum _xx,
    sum (r - i) as sum _y,
    sum ((g - r)*(r - i)) as sum _xy
from
    (select
        run, camCol, field, fieldId,
        (psfMag_g - reddening_g) as g,
        (psfMag_r - reddening_r) as r,
        (psfMag_i - reddening_i) as i
    from @database..PhotoPrimary
    where
        @select _run
        (flags & @bad_flags) = 0 and nchild = 0 and
        type = @star and
        --  Not too faint
        psfMag_i < @maglim
    ) as "obj1"
where
    --  Choose stars in nearly horizontal part of g-r-i diagram
    g - r between 0.3 and 1.1 and
    r - i between -0.1 and 0.6
group by run, camCol, field, fieldId
) as "_blue"
) as "blue",
```

```
    (
    select              --  Red part of locus
        fieldId,
        avg(g - r) as gr--  I'd prefer the median
    from
        (select
            fieldId,
            (psfMag_g - reddening_g) as g,
            (psfMag_r - reddening_r) as r,
            (psfMag_i - reddening_i) as i
        from @database..PhotoPrimary
        where
            @select _run
            (flags & @bad_flags) = 0 and nchild = 0 and
            type = @star and
            --  Not too faint
            psfMag_i < @maglim
        ) as "obj2"
    where
        --  Choose stars in vertical part of g-r-i diagram
        g - r between 1.1 and 1.6 and
        r - i between 0.8 and 1.4
    group by fieldId
    ) as "red"
where
    blue.fieldId = red.fieldId and
    blue.fieldId = field.fieldId
group by blue.run, blue.camCol
order by blue.run, blue.camCol
```



The results of the žhed query. The white cross is the canonical cosmic žhed point, with ±1% error bars.