Database exercises (user tasks in red)

To get used to relational databases and SQL, we're going to do some exercises with an online interface to a RDBMS containing astronomical data called CasJobs. This offers access to the terabytes of data that constitute the Sloan Digital Sky Surveys (SDSS).

(1) The first task is to get create an account with the CasJobs system. This can be done by clicking on "**create an account**" at:

http://casjobs.sdss.org/CasJobs/login.aspx

A user account is associated with a space of 500 MB for all your own data, tables, etc. called MyDB.

(2) Now we're going to submit a simple query to select some objects with parameters matching some constraints of interest. Assuming that you've logged in to CasJobs, select the **Query** tab. The **Context** drop down determines which database the query will run against and in this case we are interested in **"DR10"**. Remember that the basic syntax for a SQL query is:

SELECT cols FROM table WHERE …

In this case, we are interested in the following parameters: *objId, rowc_r, colc_r, pertoRad_g, petroRad_r, run, rerun, camcol, field, petroMag_r* from the table *photoObj* for objects which have a value for petroMag_r between 16 and 18. Write this query.

CasJobs offers a syntax checker (the Syntax button) to make sure that your query is valid SQL (not that it makes sense against the holdings of the database). You have two ways of submitting the query in CasJobs – a Quick query will take at most 60 seconds – try this.

You can check whether your query returns sane results by amending the query to make it only return the first 100 results. Try this with the Quick query (note for this version of SQL, you need to use the TOP keyword instead of LIMIT).

The other way is to submit the full query (no TOP restriction) to the CasJobs queue. This will run the query in the background and could take some time. (Optional task: try this when you have some time with varying size limits on the table).

You can see the results of your query by going to the **MyDB** tab. Make sure that the "**MyDB**" context is selected and you can see all your tables, how many rows they have and how big they are. To look at some of the rows, use the **Sample** tab. You can also download the data to your local hard drive with the **Download** tab.

You can see the schema for individual tables in DR10 by switching to the **"DR10"** context under the **MyDB** tab and selecting tables (as well as views, functions and procedures) of interest. Alternatively, click on the **SkyServer** tab and then select **Schema Browser** under the **Help** section. From here, you can see the details of the tables (views, etc.) in the DR10 database. Have a look at the schema for the *PhotoObj* table (using either method) and select some other columns to put into a SQL statement to run against the database. Choose some other constraint values and see what you get.

If you want to drop a table, use the **Drop** tag under **MyDB.**

(3) Now that you are familiar with the basics of SQL, we're going to build a more complicated query. The problem that we are interested in resolving is returning a set of galaxies that belong to a galaxy cluster (all objects close to particular location). However, there is contamination by foreground and background galaxies. Some of the galaxies will also have been observed with a spectrograph and others not so we will also be interested in those galaxies without spectra to target for additional spectroscopic observations.

The SDSS spectroscopic survey is not complete so we will use the spectroscopic data (*SpecObj* table) when it is available. When no spectrum is available, we will use the photometrically determined redshift (*Photoz* table).

The parameters we are interested in are: *specobjid (*from *SpecObj), ra, dec, z (*from *Galaxyj),* the differences between *Petromag_u* and *extinction_u, Petromag_g* and *extinction_g, Petromag_r* and *extinction_r, Petromag_i* and *extinction_i, Petromag_z* and *extinction_z* (all from *Galaxy* and *z* (from *Photoz)*.

To perform the location search, we're going to use a function. Have a look at the example queries to find an appropriate one at:

http://skyserver.sdss3.org/public/en/help/docs/realquery.aspx

The function is applied via an inner join with the *Galaxy* table using the object ids and around the position (208.28, 5.15, 58.31).

This now needs to be matched with the spectroscopic table (*SpecObj*) and the photo-z table (*Photoz)*. In both cases, a left outer join is required.

The constraints on this query are that the *r* magnitude extinction difference must be less than 19.1 and that the *clean* flag in the *Galaxy* table is 1.

The final element is to replace NULLS in the *specobjid, z (*from *SpecObj)* and *z (*from *Photoz)* parameters with a zero using the function ISNULL(column, replacement value).

Execute the query and look at the results – there should be about 1794 rows in the result table.

Answers

SELECT objId, rowc_r, colc_r, petroRad_g, petroRad_r, run, rerun, camcol, field, petroMag_r FROM photoObj WHERE petroMag_r >= 16 and petroMag_r < 18

or

SELECT objId, rowc_r, colc_r, petroRad_g, petroRad_r, run, rerun, camcol, field, petroMag_r FROM photoObj WHERE petroMag_r between 16 and 18

SELECT top 100 objId, rowc_r, colc_r, petroRad_g, petroRad_r, run, rerun, camcol, field, petroMag_r FROM photoObj WHERE petroMag_r between 16 and 18

SELECT ISNULL(s.specobjid, 0) as specobjid, p.ra, p.dec, p.Petromag_u – p.extinction_u, p.Petromag_g – p.extinction_g, p.Petromag_r – p.extinction_r, p.Petromag_i – p.extinction_i, p.Petromag_z – p.extinction_z, ISNULL(s.z, 0) as z, ISNULL(pz.z, 0) as pz
FROM (Galaxy AS p JOIN dbo.fGetNearbyObjEq(208.28, 5.15, 58.31) AS GN on p.objID = GN.objID
LEFT OUTER JOIN SpecObj s on s.bestObjID = p.objD)
LEFT OUTER JOIN Photoz pz on pz.objid = p.objid
WHERE p.Petromag_r – p.extinction_r < 19.1 and p.clean = 1