

Widening Access premium – extraction of data from HESA student records

```
*****
* Algorithm for institutions to use with INSTANCE record merged with STUDENT ON MODULE *
* and MODULE record *
* Example for use with 2008/09 HESA data for 2010/11 funding *
*****
```

```
*****
* Please also refer to guidance given in circular Higher Education Data Requirements *
* 2008/09 (W09/12HE) *
*****
```

```
*****
* 1. Extract eligible students *
* (variables taken from HESA tables are in upper case of the form ENTITY.FIELDNAME) *
*****
```

```
*****
* Notes *
* ne means not equals *
* proc means procedure *
* =: means begins with *
*****
```

```
data popn;
set hesa;
```

```
where INSTANCE.FUNDCODE='1' and
      INSTANCE.FESTUMK ne '3' and
      INSTANCE.MODE in ('01','02','23','24','25','31','52','53') and
      INSTANCE.COURSEAIM in (all C codes, all H codes, all I codes, all J codes, M22) and
      INSTANCE.EXCHANGE not in ('1','2','3','4','6');
```

```
*****
* 2. Include students active between 1 August 2008 and 31 July 2009 and *
* not in the final academic year of a non-standard academic year course *
* avdte is the anniversary of COMDATE in 2008/09 date *
*****
```

```
*****
** standard academic year course *
*****
```

```
if INSTANCE.TYPEYR='1' and
   INSTANCE.COMDATE<='31Jul2009'd and
   (INSTANCE.ENDDATE>='1Aug2008'd or INSTANCE.ENDDATE='') then count='1';
```

```
*****
** non-standard academic year course and student left more than *
** two weeks after the anniversary of commencing the course *
*****
```

```
if INSTANCE.TYPEYR = '2' and
   INSTANCE.ENDDATE <= '31Jul2009'd and
   INSTANCE.ENDDATE ne '' and
   INSTANCE.ENDDATE > (avdte+14) then count='1';
```

```
*****
** non-standard academic year course and student left within *
** two weeks of the anniversary of commencing the course and *
** expected length of course was two weeks or less *
*****
```

```
if INSTANCE.TYPEYR = '2' and
   INSTANCE.ENDDATE <= '31Jul2009'd and
   INSTANCE.ENDDATE ne '' and
   INSTANCE.ENDDATE <= (avdte+14) then do;
   if INSTANCE.UNITLGTH='3' and INSTANCE.SPLENGTH in ('01','02') or
      INSTANCE.UNITLGTH='4' and INSTANCE.SPLENGTH in ('01' to '14') or
      INSTANCE.UNITLGTH='5' and INSTANCE.SPLENGTH in ('01' to '42') then count='1';
```

```
*****
** non-standard academic year course - student not in final year *
*****
```

```
if INSTANCE.TYPEYR='2' and
   INSTANCE.COMDATE <= '31Jul2009'd and
   INSTANCE.DATELEFT='' then count='1';
```

```
*****
** exclude students in final academic year of non-standard academic year course *
*****
```

```
if count ne '1' then delete;
```

```
*****
* 3. Set number of credit points coded as '999' or missing to 0 *
*****
```

```
if MODULE.CRDTPTS in (999,.) then MODULE.CRDTPTS=0;
```

```
*****
* 4. Delete duplicate modules on courses by student *
* nodupkey means delete duplicates with same values of institution HUSID, *
* MTITLE and MODID *
*****
```

```
proc sort nodupkey;
by institution INSTANCE.HUSID MODULE.MODID;
```

```
*****
* 5. Sum credit points by institution and INSTANCE.HUSID (student). Dataset *
* 'outcred' is output and contains the total number of credits *
* per student (totcred) *
*****
```

```
proc summary;
by institution INSTANCE.HUSID;
var MODULE.CRDTPTS;
output out=outcred sum=totcred;
```

```
*****
* 6. Merge population dataset and total credits dataset *
*****
```

```
data popn_cred;
merge popn outcred;
by institution INSTANCE.HUSID;
```

```
*****
* 7. Delete duplicates - a dataset containing one entry per *
* student, with their total number of credit points is output *
*****
```

```
*****
```

Input data will be in the following form:

institution	HUSID	CRDTPTS	totcred
1	1	60	120
1	1	40	120
1	1	20	120
1	2	10	60
1	2	50	60

The next step will produce an output dataset in the following form:

institution	HUSID	totcred
1	1	120
1	2	60

```
*****
```

```
*****
* Merge the student data file selected above with a file containing students' postcodes *
* The postcode file (postmap) contains two fields entitled low_affluence_ind and *
* community_first_ind. If a student's postcode is located within a low affluence area *
* or a community first area the value of the corresponding field will be 1. *
* If students have more than one instance with differing postcodes then *
* preference is given to the postcode which maps to the low affluence area *
*****
```

```
data widacc;
merge popn_cred postmap(keep=institution INSTANCE.HUSID low_affluence_ind community_first_ind);
by institution INSTANCE.HUSID;
```

```
*****
* data is sorted by a field which indicates whether a student has a low affluence postcode, *
* then by mode and level, so that if a student has more than one instance, the instance *
* with a low affluence postcode will be listed before an instance without a low affluence *
* postcode, a full-time instance will be listed before a part-time instance and a first degree *
* before other undergraduate i.e. low affluence postcode, full-time and/or the highest level *
* will be listed first *
*****
```

```
*****
* nodupkey means delete any duplicates having the same key values of *
* institution and HUSID – the record that appears first will be kept *
*****
```

```
proc sort nodupkey;
by institution INSTANCE.HUSID;
```

```
*****
* 8. Only include students with 10 credit points or more *
*****
```

```
if totcred<10 then delete;
```

```
*****
* 9. Identify students eligible for low affluence or community first element *
* using low_affluence_ind and community_first_ind *
*****
```

```
if low_affluence_ind=1 then wa=1;
else wa=0;
```

```
if community_first_ind=1 then cf=1;
else cf=0;
```

```
*****
* 10. Identify students eligible for non-traditional highest qualification on entry element.*
*****
```

```
if ENTRYPROFILE.QUALENT2 in ('29','41','43','44','45','55','56','57','92','93','98') then non_trad=1 ;
else non_trad=0;
```

```
*****
** indicator to count number of students who are eligible for premium *
** i.e. all students who meet initial criteria before low affluence, *
** community first and non-traditional highest qualification on entry *
** elements are applied *
*****
```

```
total=1;
```

```
*****
* 11. Summarise by institution to get total number of eligible students *
* and numbers allocated each element of the premium *
*****
```

```
proc summary;
by institution;
var total wa cf non_trad;
output out=outwa sum= total wa cf non_trad;
```