```
/*
possible geoacy (in metres) values after logic has completed:
        500 - unknown datum, gps not indicated as used
        475 - gps indicated in notes column, but still unknown datum
        280 - datum is gda94, date of record is earlier than 1980
        280 - datum is agd66/agd84, date of record is greater than 1980
        255 - either condition for 280 value, along with gps indicated
         60 - datum is agd66/agd84, date of record is less than 1980
         50 - datum is gda94, date of record is greater than/equal to 1980
         35 - gps in notes, datum is agd66/84, date of record is less than 1980
         25 - gps in notes, datum is gda94, date of record is greater than/equal to 1980

any of the above + 2200 :
        error was introduced because valid grid is provided for conversion, and species is considered at risk
*/



// start with geoacy at 500
// (unknown datum + possible conversion of unknown method, assuming valid correct location measurement)
int geoacy = 500

// get database values
boolean gpsCol = getColumnValue("GPS")
String notesCol = getColumnValue("Other Notes")
String easting = getColumnValue("Eastings")
String northing = getColumnValue("Northings")

// if gps was used, subtract 25 from geoacy
if (gpsCol || notesCol.containsIgnoringCase("GPS"))
{
        geoacy = geoacy - 25
}



boolean unknown = true;

// the use of the earlier aus. national spheroid (aka aus. geodetic datum) is currently not true
boolean agd66 = false, agd84 = false;

// first, we'll try and see if this is the wgs84 spheroid
// if gda-94 or wgs-84 was used, subtract 450
boolean gda94 = notesCol.matches('gda?94') || notesCol.matches('mga?94')
boolean wgs84 = notesCol.matches('wgs?84')



if (gda94 || wgs84)
{
        // a conversion has occurred, and we don't know how it was done...
        // we'll allow for either (a) an unnecessary transform in the accuracy; or (b) an incorrect datum
        If( year-of-record < 1980 ) {
```

```
                geoacy = geoacy - 220
        }
        // it is likely that everything is as it seems - keep 50m for 1/20th of a grid-square or gps error
        else {
                geoacy = geoacy - 450
        }
        unknown = false
}
// we couldn't specifically match to the wgs84/gda94 datums - check in here for other datums,
//      if found, they will still result in a conversion, but the error will be reduced
else {
        agd66 = notesCol.matches('amg?66') || notesCol.matches('agd?66')
        agd84 = notesCol.matches('agd?84') || notesCol.matches('amg?84')

        if( agd66 || agd84 ) {
                // there is a chance that the datum reported is not correct
                If( year-of-record > 1980 ) {
                        geoacy = geoacy - 220
                }
                // it is likely that everything is as it seems
                // we still need to allow for the error in molodensky's conversion
                else {
                        geoacy = geoacy - 440
                }
                unknown = false
        }

        // if unknown is still true - we've not been able to match any datum test so far
        // we will convert anyway if we have valid grid coord's; we will assume data was
        // recorded in the wgs84+ datum; we won't do a shift on the data
}


// valid grid coord's
if (easting.length == 6 && northing.length == 7)
{
        // note, this code must come before the conversion, as it works on grid-coords
        if a command line switch or system parameter does not disable this behaviour {
                if( record is sensitive ) {
                        //obfuscate the location details
                        grid eastings += introduce_error()
                        grid northings += introduce_error()

                        // this is the maximum poss. error distance on the hypot.
                        geoacy += 2200;
                }
        }

        // convert using redfearn's formula, different params depending on whether in gda-94, wgs-84 or agd
        if (gda94) {
                latlong = convertGda94GridToLatLong(easting, northing) //wgs84 constants used
```

```
        }
        else if (wgs84) {
                latlong = convertWgs84GridToLatLong(easting, northing) //wgs84 constants used
        }
        else if (agd66 || agd84) {
                latlong = convertAgdGridToLatLong(easting, northing) //ans constants used
        }
        else {
                // (geoacy will still be 500 or 2700 depending on restricted status)
                latlong = convertUnknownGridToLatLong(easting, northing) //wgs84 constants used
        }




        // if necessary, attempt lat/long datum conversion using molodensky shift

        // (geoacy will cover us for potential unnecessary conversion)
        // (we'll only do this with agd datums at this stage)
        if (agd84)
        {


                doMolodenskyShiftWithAgd84Params (latlong) //this will modify latlong
                gda94 = true    //we've shifted to gda-94 , agd84 will still be true
        }
        elseif (agd66)
        {
                doMolodenskyShiftWithAgd66Params (latlong) //this will modify latlong
                gda94 = true    //we've shifted to gda-94, agd66 will still be true
        }

        // csv will include lat,long,geoacy as well as original sge,sgn
        //write out all data
        // should include the various booleans, done in a way that will not cause errors in the avh feed
        // still thinking about this one... might like to output it as a mask in a suitable 'export status' field ?



}

// grid coord's are invalid or at a different scale than expected
// need the intervention & interpretation of a brain...
else
{
   // csv will include only sge,sgn
   // should we include include geoacy in this case??
        // geoacy is really unknown at this stage - should probably be set to 0,
        //      but if we can have this as a constant, incase we decide to overload geoacy in this situation??
        //      (I dunno if we'd do this, because it might be in contravention of the hispid3 standard)
```

```
        // need some way to indicate an error in sge,sgn ?? (could this be the absence of geographical
coord's?)
        //


        if a command line switch or system parameter does not disable this behaviour {
                if( record is sensitive ) {
                        //no spatial data should be provided (to mitigate the risk that it can be deciphered by a brain)
                }
                else {
                        sge/sgn spatial data provided as per (without conversion)
                }
        }
        else {
                sge/sgn spatial data provided as per (without conversion)
        }

}

introduce_error() {
        rand_mult = 0;

        // we don't want a 0 multiplier because that has no affect on the value(s)
        while( rand_mult >= -1 && <= 1 )
         -> rand_mult = (random in range -4 -> 4)  [possibilities are -4, -3, -2, -1, 0, 1, 2, 3, 4]

        // now introduce our error
        error = rand_mult * (random in range 125 -> 390) [range of 265m]

        /* looking at the logic:
        error = [-4, -3, -2, 2, 3, 4] multipled by [random in range of 265m]
        error = [-1560 -> -250; 250->1560]
        error = add an additional random distance at least ~350 metres away from this point, and at most
        ~2200m [@45,135,205 & 315 degrees bearing from the point]

        why 265 m? because this * 4 gives us the short side of a right-angled triangle (anchored to the
        specimen on one end) that is ~1500m on the hypotenuse
        */

        return error;
}
```