# Report on Particular Contributions to the NGS Esophogeal Cancer Project

Ross Eldridge

## Supplemental Materials (Code Samples)

**Figure 1**

**Schema for Initial Database**

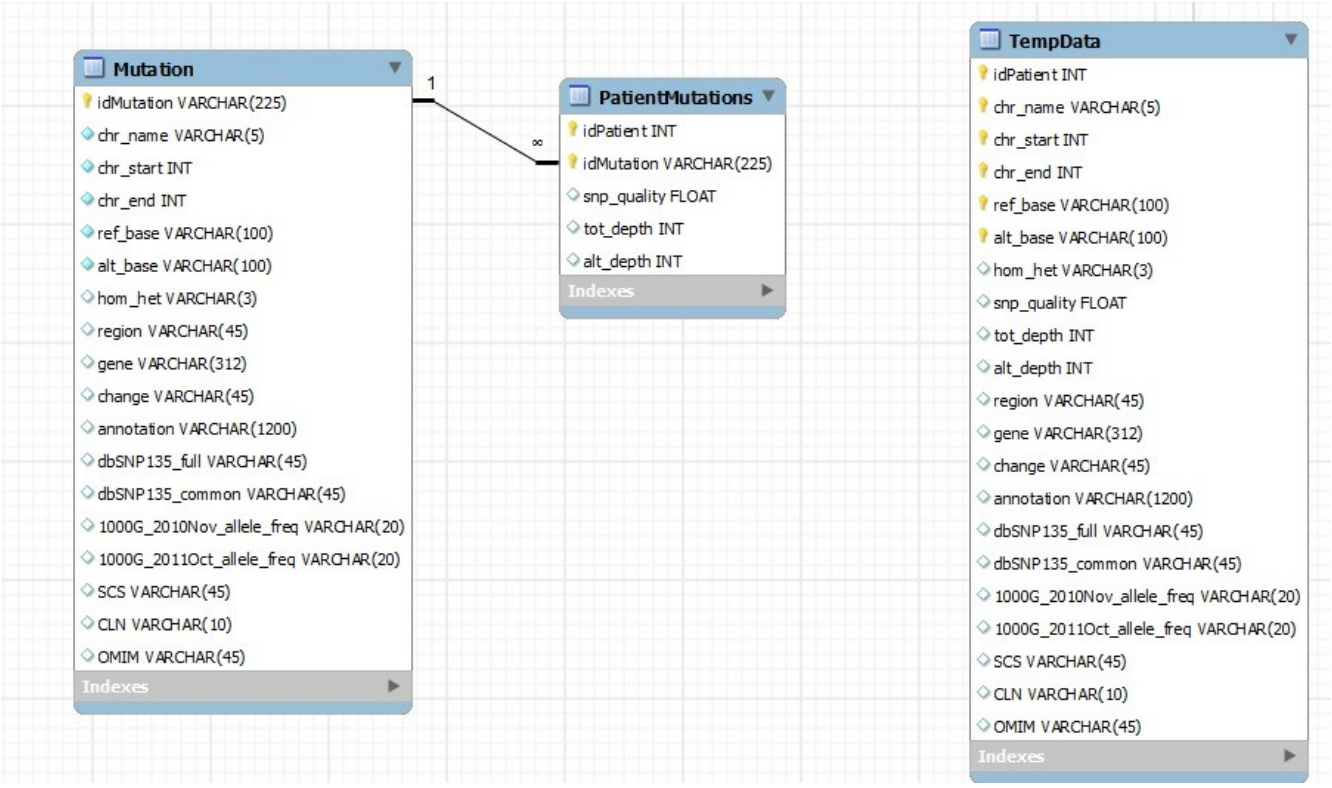
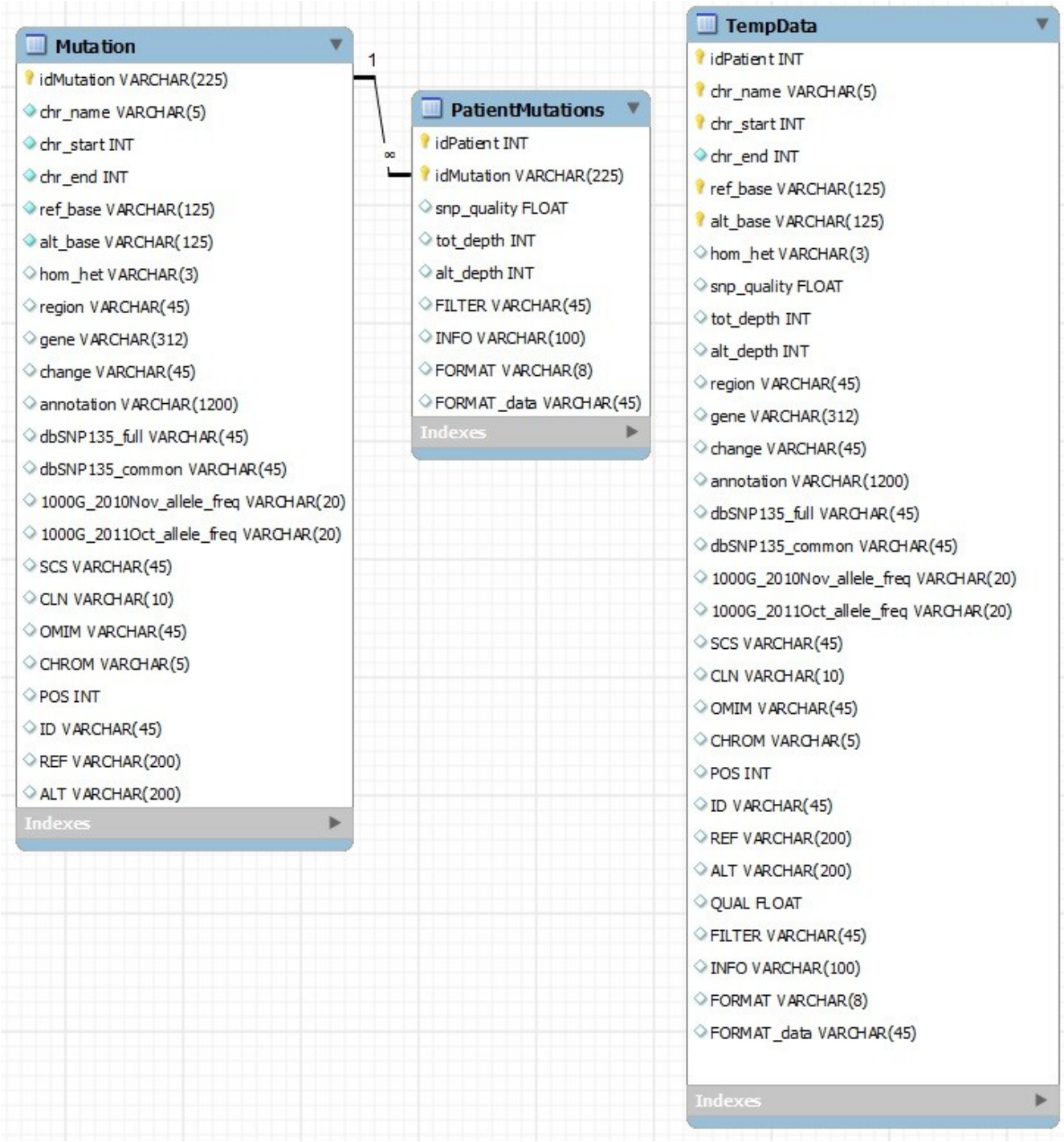


**Figure 2**

**Schema for VCF Modified Database**



**Sample SQL code to insert data from the temporary table into the two main tables (VCF):**

```
REPLACE INTO mutation
SELECT DISTINCT
      CONCAT(T.chr_name,T.chr_start,T.chr_end,T.ref_base,T.alt_base),
      T.chr_name,
      T.chr_start,
      T.chr_end,
      T.ref_base,
      T.alt_base,
      T.hom_het,
      T.region,
      T.gene,
      T.change,
      T.annotation,
      T.dbSNP135_full,
      T.dbSNP135_common,
      T.1000G_2010Nov_allele_freq,
      T.1000G_2011Oct_allele_freq,
      T.SCS,
      T.CLN,
      T.OMIM,
      T.CHROM,
      T.POS,
      T.ID,
      T.REF,
      T.ALT
FROM tempdata T;

INSERT INTO patientmutations
SELECT
      T.idPatient,
      CONCAT(T.chr_name,T.chr_start,T.chr_end,T.ref_base,T.alt_base),
      T.snp_quality,
      T.tot_depth,
      T.alt_depth,
      T.FILTER,
      T.INFO,
```

```
      T.FORMAT,
      T.FORMAT_data
FROM tempdata T;
```

### Sample SQL code to query the database for particular variants:

```
SELECT M.CHROM AS CHROM, M.POS AS POS, "." AS ID, M.REF AS REF, M.ALT AS ALT,
round(snpq,3) AS QUAL, "." AS FILTER, INFO, FORMAT, FORMAT_data
      FROM
      (SELECT count(P.idPatient) AS pcount, M.idMutation AS mut, avg(P.snp_quality)
AS snpq, P.INFO AS INFO, P.FORMAT AS FORMAT, P.FORMAT_data AS FORMAT_data
            FROM patientmutations P, mutation M
            WHERE M.idMutation = P.idMutation AND
                  (M.region = "exonic" OR
                  M.region = "exonic;splicing" OR
                  M.region = "splicing") AND
                  ISNULL(M.dbSNP135_full) AND
                  M.change != "synonymous_SNV"
            GROUP BY M.idMutation) gp, mutation M
      WHERE gp.pcount > 5
            AND M.idMutation = gp.mut
      ORDER BY CHROM ASC;
```

### Sample Python 2.7 code for the concatenation of Excel and VCF files:

```
import sys
import copy
from time import clock

def readXLSCSV(filename):
    with open(filename,'r') as f:
        lines = f.readlines()
        for k in lines:
            lines[lines.index(k)] = k.translate(None,'"')
    return lines

def readVCF(filename):
    with open(filename,'r') as f:
        lines = f.readlines()
        lines_iter = copy.deepcopy(lines)
        for k in lines_iter:
            if k[0] == '#' and k[1] == '#':
                lines.remove(k)
    return lines
```

```
def concatVCF(file1,file2):
    print "Filenames:",file1,"\t",file2
    print "Reading VCF ... ",
    vcf = readVCF(file1)
    print "done.\nReading XLS ... ",
    xls = readXLSCSV(file2)
    print "done."
    mm = 0
    output = xls[0].rstrip('\n')+'\t'+vcf[0]
    print "Concatenation Progress [",
    for i in range(1,len(vcf)-1):
        vcft = vcf[i].split('\t')
        xlst = xls[i].split('\t')
        #Insertion
        if xlst[4] == '-':
            d = len(vcft[3].split(',')[0])-1
        #Deletion
        elif xlst[5] == '-':
            d = len(vcft[4].split(',')[0])
        #SNV
        else:
            d = 0
        #Mismatch conditional statement based on reported "start" location
        if int(vcft[1]) + d != int(xlst[2]):
            print vcft[1]+"   "+xlst[2]+"   "+str(d)
            mm += 1
        else:
            if i / (len(vcf)/10) != (i-1) / (len(vcf)/10):
                print ".",
            output += xls[i].rstrip('\n')+'\t'+vcf[i]
    print "]\nMismatches:",mm
    outfile = "XLS-VCF-"+str(xlst[0])+".csv"
    with open(outfile,'w') as o:
        o.write(output)
    if mm == 0:
        return 1
    return 0

def main(argv):
    clock()
    l = len(argv)/2
    scount = 0
    for i in range(l):
        scount += concatVCF(argv[i*2],argv[i*2+1])
    print "File-pairs successfully concatenated:",str(scount)+"/"+str(l)
    end_scr = clock()
    print "Script completed in",end_scr,"seconds.\n"

if __name__ == '__main__':
    main(sys.argv[1:])
```