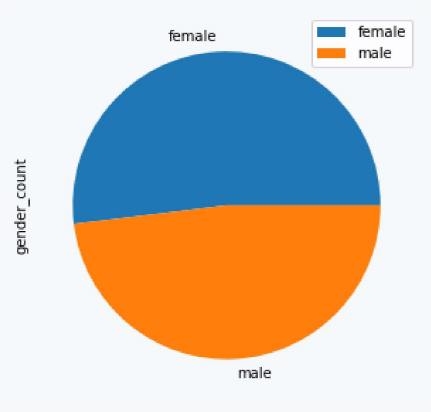
Project 4 Report - Veronika Ebenal, Raghavendra Tikare, Stanislav Klein

```
# Compute sample count and ratio by gender
SELECT gender, gender_count, RATIO_TO_REPORT(gender_count)
OVER ( ORDER BY gender_count) AS gender_ratio
FROM (
  SELECT gender, COUNT(gender) AS gender_count,
  FROM [genomics-public-data:1000_genomes.sample_info]
  WHEREIn_Phase1_Integrated_Variant_Set = TRUE
  GROUP BY gender)
```

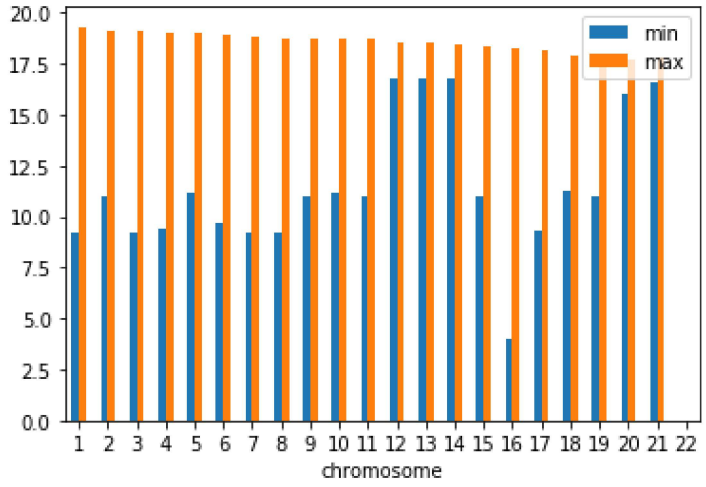
```
Estimated query number of gigabytes: 2.3853033781051636e-05
```



IT: The program searches the database makes a new tables with columns of gender and count of that gender found in the original data. It then uses this new table to get values of gender, gender count, and ratio of gender (which is obtained by dividing the gender count over total sum of gender counts). The results are grouped by gender. We end up getting 567 females and 525 males, with a ratio of 51.9% female and 48.1% male.
Biological: This program is analyzing the samples sizes of genders across populations. The original data has varying amounts of each gender in each population, but here we want to see the total values for each gender, as well as the ratio, to see if the total sample sizes are roughly equal. As can be seen in the above image, they are indeed close to equal.

```
#Min/Max Chromosomal Positions of Variants
SELECT INTEGER(reference_name) AS chromosome, MIN(start) AS min, MAX(start) AS max
FROM [genomics-public-data:1000_genomes.variants]
OMIT RECORD IF reference_name IN ("X", "Y", "MT")
GROUP BY chromosome
```

```
Estimated query number of gigabytes:  0.42134151980280876
```



IT: This will query from the data the reference_name (as an integer) - saving this value as "chromosome" - and the minimum and maximum start positions for each of the chromosomes. The results are grouped by "chromosome".
Biological: We query from the data each chromosome (excluding X, Y, and MT), and the minimum and maximum start positions of variants found on these chromosomes. We can see from this graph that some chromosomes, such as 16, have a wide variety in variant start positions (from position 55 to 81194906), while others are much more constrained, such as chromosomes 12, 13, and 14.

```
# Find variants on chromosome 17 that reside on the same start with the same
reference base

SELECT reference_name, start, reference_bases, COUNT(start) AS num_alternates
FROM [genomics-public-data:1000_genomes.variants]
WHERE reference_name = '17'
GROUP BY reference_name, start, reference_bases
HAVING num_alternates > 1
ORDER BY reference_name, start, reference_bases

Estimated query number of gigabytes:  0.5961888916790485
```

IT: Whenever the reference_name is 17, and the num_alternatives (that is, count of "start") is over 1, we take the reference_name, start, reference_bases, and num_alternatives. This is grouped and ordered by reference_name, then start, then reference_bases.
Biological: What we are really trying to see here, is if reference_name (that is, the chromosome name), (variant) start, and references base can together form a unique key on the data. That is, if each combination occurs at most once. If num_alternatives is never over 1, then this is true. Unfortunately, upon testing this on chromosome 17, we end up with some num_alternatives values of 2, so it does not form a unique key.

```
# Count the number of variants in BRCA1
SELECT count(reference_name) as num_variants,
FROM [genomics-public-data:1000_genomes.variants]
WHERE reference_name = '17' AND start BETWEEN 41196311 AND 41277499

Estimated query number of gigabytes: 0.42134151980280876
```

IT: Find all rows in the data where reference_name is 17 and start is between 41196311 and 41277499 and then collect the total count on this (as num_variants).
Biological: BRCA1 resides on chromosome 17 from position 41196312 to 41277500. We want to see the number of variants with the start position in this gene. Upon running this query, we see there are 879 variants with start positions within the BRCA1 gene.

```
# The following query computes the allelic frequency for BRCA1 variants in the
# 1,000 Genomes dataset further classified by ethnicity from the phenotypic data
# and also includes the pre-computed value from the dataset.
SELECT reference_name, start, super_population, reference_bases, alternate_bases,
  SUM(ref_count)+SUM(alt_count) AS num_sample_alleles,
  SUM(ref_count) AS sample_allele_ref_cnt, SUM(alt_count) AS sample_allele_alt_cnt,
  SUM(ref_count)/(SUM(ref_count)+SUM(alt_count)) AS ref_freq,
  SUM(alt_count)/(SUM(ref_count)+SUM(alt_count)) AS alt_freq, alt_freq_from_1KG
FROM (SELECT
    Reference_name, start, super_population, reference_bases, alternate_bases, alt,
    SUM(INTEGER(0 = call.genotype)) WITHIN RECORD AS ref_count,
    SUM(INTEGER(alt = call.genotype)) WITHIN RECORD AS alt_count,
    CASE
    WHEN super_population = 'EAS' THEN asn_af
    WHEN super_population=  'EUR' THEN eur_af
    WHEN super_population = 'AFR' THEN afr_af
    WHEN super_population = 'AMR' THEN amr_af
    END AS alt_freq_from_1KG
  FROM FLATTEN(FLATTEN((SELECT
          Reference_name, start, reference_bases, alternate_bases,
          POSITION(alternate_bases) AS alt, call.call_set_name, call.genotype,
          Afr_af, amr_af, asn_af, eur_af,
        FROM [genomics-public-data:1000_genomes.variants]
        WHERE
          reference_name = '17' AND start BETWEEN 41196311 AND 41277499
          AND vt='SNP' ), call), alt) AS g
  JOIN [genomics-public-data:1000_genomes.sample_info] p
  ON g.call.call_set_name = p.sample)
GROUP BY reference_name, start, super_population, reference_bases, alternate_bases,
  alt_freq_from_1KG
ORDER BY reference_name, start, Super_population

Estimated query number of gigabytes: 1005.6224315874279
```

IT: Here we first make a flattened table (that is, with no nesting in the structure) called “call”, taking several columns from the original data (where reference_name is 17 and start is in a certain range) and adding some new values such as POSITION(alternate_bases). We join this table on the original data where call_set_name of call matches sample of p. From this we make

another table, taking several columns from the original table and making new ones again like ref_count and alt_count, and alt_freq_from_1KG which will be dependent on super_population. Finally, we make another table from this one, using several column values and making new ones such as num_sample_alleles which is the sum of ref_count added to the sum of alt_count. We then group and order this table on several values.

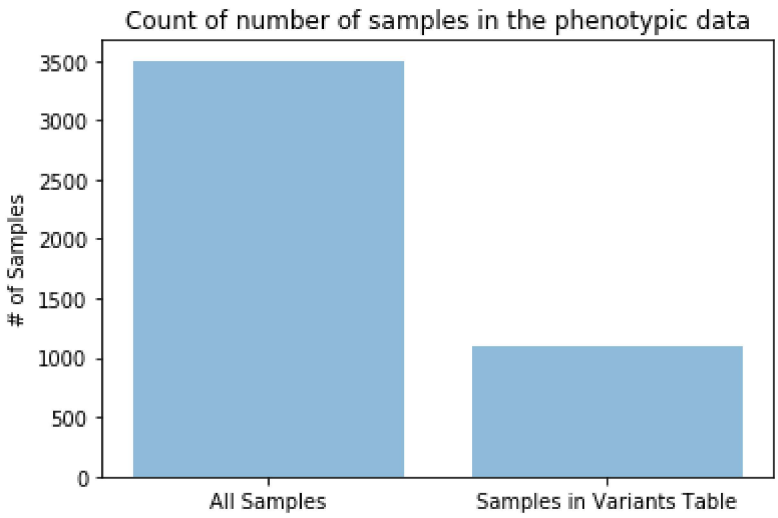
Biological: We want compute some alternate allele frequencies and how they differ by super population groups. This query computes the allelic frequency for BRCA1 variants in the 1,000 Genomes dataset further classified by ethnicity from the phenotypic data and also includes the pre-computed value from the dataset. From the results, we can compare frequencies of alleles and how they vary according to super population (Asia, Europe, etc.)

```
# An example of a pattern one might use for Hardy-Weinberg Equilibrium

# queries upon 1,000 Genomes variants.  It is specifically computing

# the Hardy-Weinberg Equilibrium for the variants found in BRCA1 and

# then computing the chi-squared score for the observed versus

# expected counts for the calls.

SELECT reference_name, start, END, reference_bases, alt, vt,

  ROUND(POW(hom_ref_count - expected_hom_ref_count,

      2)/expected_hom_ref_count +

    POW(hom_alt_count - expected_hom_alt_count,

      2)/expected_hom_alt_count +

    POW(het_count - expected_het_count,

      2)/expected_het_count,

    3) AS chi_squared_score,

  Total_count, hom_ref_count,

  ROUND(expected_hom_ref_count,

    2) AS expected_hom_ref_count,

  het_count,

  ROUND(expected_het_count,

    2) AS expected_het_count,
```

```
  hom_alt_count,

  ROUND(expected_hom_alt_count,

    2) AS expected_hom_alt_count,

  ROUND(alt_freq,

    4) AS alt_freq,

  alt_freq_from_1KG ……

(this query is really long so I won't paste the whole thing)

Estimated query number of gigabytes: 642.0727104060352
```

IT: Create a table with several columns and a new columns alt (from concatenating alternate_bases) also create column from the first row of "genotype" as first_allele, and second row as second_allele. Do this only where reference_name is 17 and the start is within a given range. From this, make a new table, with some of the same columns and new columns made by summing first_allele and second_allele under different conditions (of 1 or 0 for the values). From this, make another new table by selecting many columns again and making new columns by performing mathematic functions on columns hom_ref_freq , hw_ref_freq, and het_freq. Finally, give a table by selecting many columns from that last table and adding new columns, rounding the results of performing mathematic functions of hom_ref_count, expected_hom_ref_count, hom_alt_count, and expected_hom_alt_count. Order this last table by reference_name and then by start.

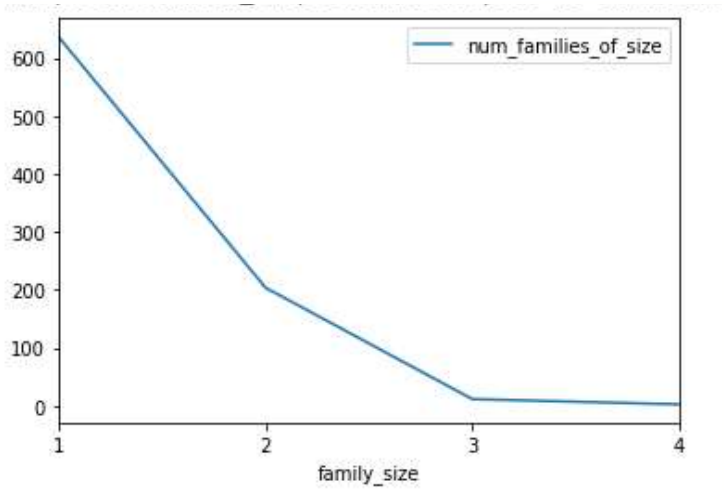
Biological: Here we test for Hardy-Weinberg Equilibrium. That is, whether the relationship between allele frequencies and genotype frequencies stay constant among populations. We are performing this computation for BRCA1 variants and then computing the chi-squared score for the observed versus expected counts for the calls. We know the expected homogenous and heterogenous number of genotypes and frequencies thereof from the HWE. We compare those to the observed values generated from the query. For example, each row in the result gives us the chromosome and position, the allele, the variance type, the chi-squared score (to see how values vary from those expected), and then the counts and frequencies (observed and expected) of the described allele. From the results, the data passes the Hardy-Weinberg Equilibrium Test.



Compute the distribution of family sizes

```
%%bigquery --project eichornchen df
SELECT
num_family_members AS family_size,
COUNT(num_family_members) AS num_families_of_size
FROM (
  SELECT
  family_id,
  COUNT(family_id) AS num_family_members
  FROM
  `genomics-public-data.1000_genomes.sample_info`
  WHERE
  In_Phase1_Integrated_Variant_Set = TRUE
  GROUP BY
  family_id)
GROUP BY
Family_size
```

```
Estimated query number of gigabytes:2.4281442165374756e-05
```



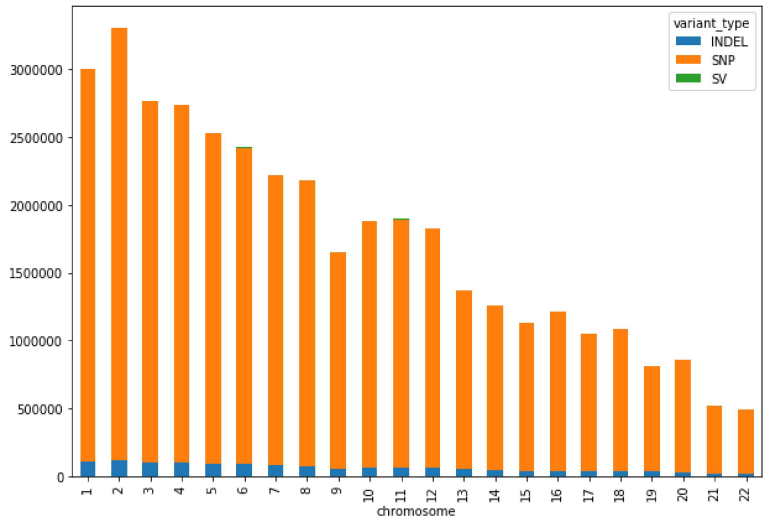
IT: Here we generally compute how the family sizes are distributed. Here we consider the number of family members in the group by selecting the column num_family_members and the count of rows for each value thereof, from a table where rows are selected on by family_id and count thereof. The results are grouped by family_size, giving us a table of four rows, with family size 1 to 4, quickly decreasing in num_family_of_size as family_size increases.

Biological: In order to find the distribution of family sizes within the sample populations, the samples' respective identities are taken (via family_id) and they are grouped accordingly. As we would expect the vast majority of people have family sizes of 1 or 2, with only a few having 3 or 4.

Frequency of variant types per chromosome

```
SELECT
  INTEGER(reference_name) AS chromosome,
  vt AS variant_type,
  COUNT(1) AS cnt
 FROM
   [genomics-public-data:1000_genomes.variants]
 OMIT RECORD IF
   reference_name IN ("X", "Y", "MT")
 GROUP BY
   chromosome,
   Variant_type
```

```
Estimated query number of gigabytes:0.31291691306978464
```



IT: Here we select referecnce_name, variant_type, and count from all rows of the data, except those where the refernce_name is “X”, “Y”, or “MT”. The results are grouped by chromosome and variant type.

Biological : Here we query to see when the frequencies of variants (i.e the number of times the mutation occurs) in each chromosome. We consider all the chromosomes from the datset excluding the sex chromosomes. We can see in the results that though total counts gradually tend to decrease, the ratios tend to stay roughly equal, mostly SNPs (single nucleotide polymorphisms) and very few SVs (structural variants).