SNP Back-end Data Requirements

Author: sc

Created: May 15, 2006

Last Modified: September 6, 2006 09:47

1 Purpose of Document

To describe the scope and constraints on dbSNP data loaded into MGI.

2 Introduction

This document is not meant to replace Richard's Requirements document. It is meant to describe the rules we apply to the data to 1) determine which RefSNPs to load 2) calculate MGI derived values from dbSNP values.

3 Definitions

- Valid Strain Allele - 'A', 'T', 'C', 'G'. Invalid strain alleles are '', and 'N'.

4 Other Related Documents

1. dbSNP Handbook - http://www.ncbi.nih.gov/books/bv.fcgi?rid=handbook.chapter.1143
2. dbSNP FAQ archive - www.ncbi.nlm.nih.gov/books/bv.fcgi?rid=helpsnpfaq.TOC
3. Snp Schema Document
4. dbSNP Build 126 Data Document

5 Scope of dbSNP Data In MGI

dbSNP is currently our only source of SNP data. Analysis is ongoing to determine whether we need to augment with data from MPD or internal Jackson Laboratory sources.

Constraints on dbSNP RefSNPs loaded:

1. Load dbSNP mouse snps
2. Load only those RefSNPs that have Mouse Genome Build 36 coordinates on the C57BL/6J assembly.
3. Load RefSNPs regardless of validation status (validated **and** unvalidated).
4. Load Chromosomes 1-19, X, Y. We **are not** loading Mitochondrial, Unknown
5. **Do not** load RefSNPs with coordinates on **multiple** chromosome.

6. Load RefSNPs with no more than two coordinates on the **same** chromosome.
7. Load only those RefSNPs that have at least one SubSnp with valid strain alleles (see definitions).

6 Data Requirements

Further constraints and rules. See snp_schema_doc.pdf for the data we load and build125data.pdf for the data we parse from dbSNP.

1. SubSnp Constraints and Rules
    1. Load SubSNP observed alleles in order to display these alleles with each SS that have no strain alleles.
    2. 8/2006 - Load only valid strain alleles (see definitions) DO NOT load 'N' or '' strain/alleles.
    3. Translate dbSNP Function Class and Variation Class values to MGI values. e.g. dbSNP Function Class term 'coding-synonymous' is translated to 'Coding-Synonymous' - see *.goodbad files in dbsnpload product data directory.
    4. Translate dbSNP strains to MGI strains. A dbSNP strain can be either a string provided by the submitter, e.g. "B10.D2" which is translated to MGI strain "B10.D2-H2<d>", or a Jax registry ID e.g. 001146 which is associated with MGI strain "SPRET/EiJ"
    5. The data model must support linking to dbSNP via SS ID, Submitter Snp ID, Submitter Handle, and Population Name.
2. RefSnp Constraints and Rules.
    1. Data model must handle multiple assemblies, even though we presently load only coordinates on C57BL/6J. Prior to MGI3.44 the data model supported loading snps on multiple assemblies (we used MAP* tables for snp coordinates, where an assembly was represented by a MAP_Coord_Collection object.). **As of MGI3.44 multiple assemblies no longer supported, but adding an assembly attribute to SNP_Coord_Cache should fix it.**
    2. Compute and store an RS consensus allele for each strain by first flipping all alleles in reverse orientation to the RS flanking sequence (remember, the RS flanking sequence is the flank of the SS exemplar, so there is always an SS with forward orientation). If the simple majority of alleles agree, we have a consensus allele, otherwise the consensus is "?". Store the consensus allele in forward orientation.
    3. Compute and store isConflict bit for each ConsensusSnp strain allele.

       e.g. if the union of alleles across all SS for given strain are:
        - A, C - isConflict = true
        - A - is Conflict = false

4. Compute and store an RS allele summary in forward orientation to the RS flanking sequence.
    1. If there's a deletion "-", always display this first.
    2. For single nucleotide alleles, sort by alphabetical order (i.e. A/C/G/T). Display single nucleotide alleles after a deletion allele (if a deletion allele is present).
    3. For multiple nucleotide alleles, sort first by the number of nucleotides in the allele, and second by alphabetical order. Display multiple nucleotide alleles after single nucleotide alleles (if a single nucleotide allele is present).

    In summary, sort observed alleles by: -/A/C/G/T/multiples (sort multiples first by nucleotide #, second by alphabetical order)
5. Compute and store IUPAC code for the RS allele summary.
6. Compute and store an RS variation class, based on the allele summary. 5/15/2006 new algorithm TR7708 addresses the bug.

    New Rules:
    1. If at least one allele has value "-", then:
        1. If there is only the "-" allele represented, Variation Type="IN-DEL"
        2. If there is only one other allele, Variation Type="IN-DEL"
        3. If at least one allele has multiple nucleotides, then:
            - If all non "-" alleles are of different sizes, and all involve the same nucleotide, Variation Type="IN-DEL" (for example: -/G/GG/GGG)
            - If all non "-" alleles are of different sizes, and involve any different nucleotides, Variation Type="Mixed" (for example: -/A/GG)
            - If any two non "-" alleles are of the same size, Variation Type="Mixed"

    2. If there are no "-" alleles, then:
        1. If every allele represented in the allele summary is a single nucleotide (A, C, G, or T), Variation Type="SNP" (includes cases of single alleles)
        2. If at least one allele has multiple nucleotides, then:
            - - If all have the same size, Variation Type="MNP"
            - - If all alleles are of different sizes, and all involve the same nucleotide, Variation Type="IN-DEL" (for example: G/GG/GGG)

- - If all alleles are of different sizes, and involve any different nucleotides, Variation Type="Mixed" (for example: A/GG)
- - If any two alleles are of the same size and at least one additional allele is of a different size, Variation Type="Mixed"

No changes for "Named" Variation Type.

7. The data model must support querying by aggregate RS consensus data i.e. we should represent the RS consensus in the data model as a union of all contributing SS data.
8. The data model must support querying by RS variation class, and function class and will select a set of RS.
9. The data model must support linking to dbSNP via RS ID

3. General constraints and rules
    1. The data model must support querying by submitter SNP ID, SS ID, and RS ID. All will select an RS.
    2. We **do not** want an EntrezGene ID query to return an RS.
    3. Load dbSNP snp to marker relationships
    4. Compute MGI snp to marker relationships for all markers within one megabase distance from the snp location. Do not create relationships to QTL markers. See TR7829
    5. Snp to marker relationships shall be recomputed weekly after the EntrezGene load to pick up any new markers.
    6. dbSNP data shall be reloaded every time a new dbSNP build occurs. The builds are done when 1) there is a new build of one of the genomes included in dbSNP or 2) when there are a significant number of submissions to be incorporated into dbSNP.
    7. If the dbSNP build is based on a new version of the mouse assembly, then the dbSNP load must be coordinated with NCBI, Ensembl, and Vega coordinate loads, UniSTS load, QTL load, and the microRNA load based on that assembly.