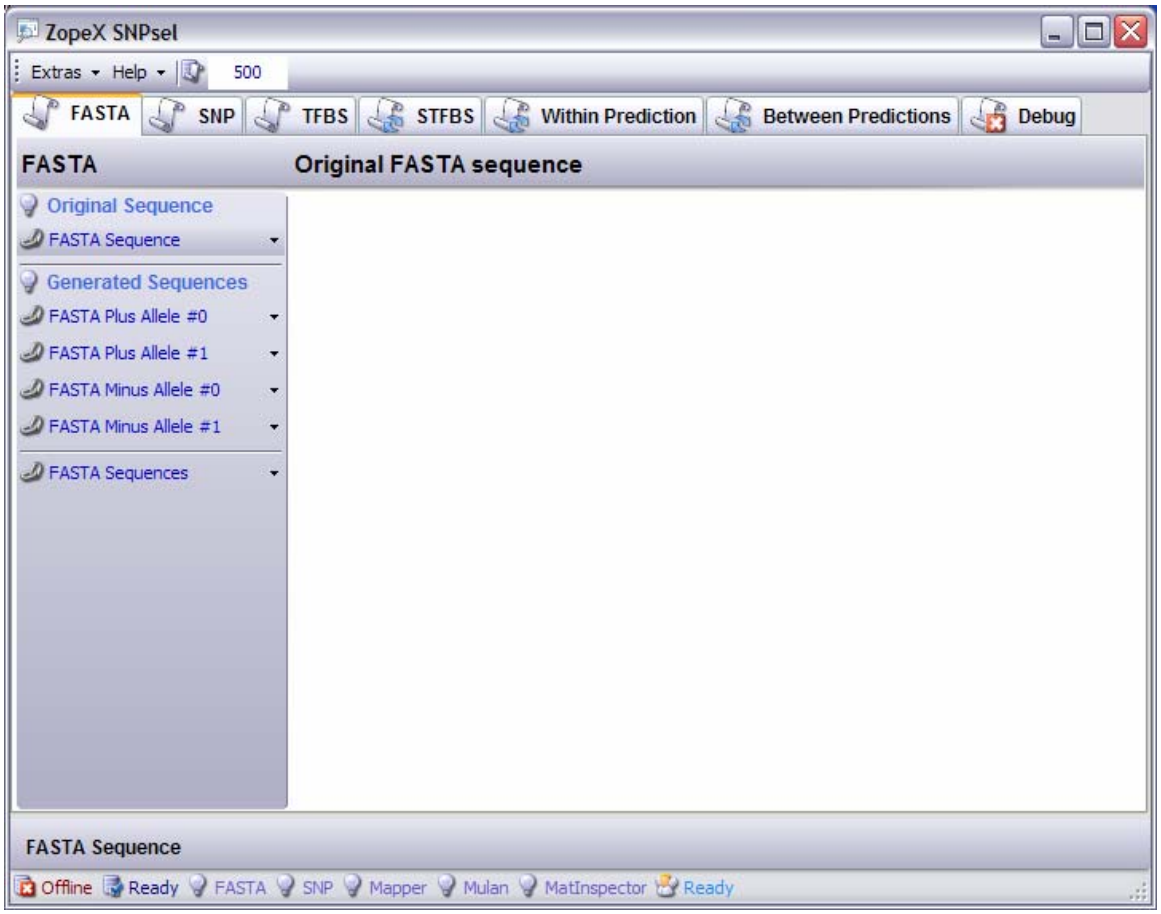
1. Introduction

The tool SNPsel compares TFBS identified by different TFBS prediction programs, in order to find such, which were discovered by several programs at the same sequence position. As base of such an analysis serves a FASTA sequence and a collection of simple nucleotid polymorphisms (SNPs) within the appropriate sequence range. With both informations (FASTA and SNPs) SNPsel generates two sequences, which contain either only the left or only the right alleles of corresponding SNPs. These two sequences can be analyzed afterwards by the external web based applications "Mapper", "Mulan/multiTF" and "MatInspector" to find potential SNP affected TFBS (STFBS).

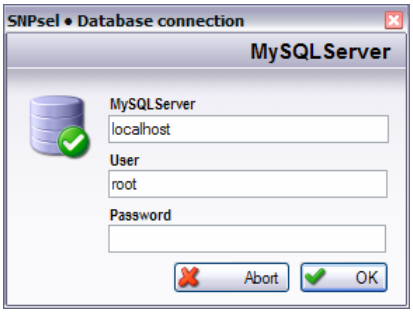
SNPsel uses the results of these analyses, in order to find TFBS, which are influenceable by SNP alleles. Such TFBS can be divided in two classes: Into one, which are resistant concerning their SNPs (unaffected STFBS) and into those, which are influenceable by SNPs (affected STFBS). The use of different TFBS prediction programs can produce different predicted STFBS. To acquire a greater significance of predicted STFBS, the different STFBS collections can be compared among themselves.

1.1. General structure

The Application can be divided into different visual ranges. The (upper) menubar contains different menu options. The tabpage within the middle range of SNPsel contains different sides, which exhibit all together the following structure: The left side holds the menu options which contains hinged menupages. The right side holds a tabular representation of SNPs, TFBS or STFBS. Under the tabpage is the embedded assistance, into whom a short explanation is faded in. The lower statusbar contains different information sources with lamps accordingly placed in front. A lamp shines, as soon as the information source is present and holds some data.



1.2. Database Connection



Since SNPsel is a database application, a connection to a MySQL Community Server must be opened first. If the database server already runs locally, then only the password needs to be indicated. Afterwards informations can be imported from external files. Otherwise you should adapt the information in the input fields to the current conditions.

The address (URL) of the database server should be inserted under "MySQL server", the username under "users" with the appropriate password under "password". Note only that you must have administration rights to use this program, since SNPsel creates a new database on the server, which can be removed by terminating this application again.

2. Import of tab-delimited text files

In order that SNPsel can generate the allelic base sequences, wich are needed by the extenal TFBS prediction programs, the original FASTA sequence and the corresponding SNP collection must be imported. After importing these two files (FASTA and SNPs), SNPsel automatically generates the allelic sequences, which should be exported as FASTA files and provided as input for TFBS predictions.

The TFBS predictions provided by "Mapper", "Mulan/multiTF" and "MatInspector" can be saved as tab-delimited text files on disk, because they are used by SNPsel for finding affected and unaffected STFBS. SNPsel can import the saved TFBS predictions directly without any modifications provided by the user. Once you have imported the TFBS predictions, in which you are interested, you can manually start the comparision analysis performed by SNPsel by clicking ...

You are not forced to use all predictions programs, because SNPsel can perform comparisions with a subset of user provided predictions. To assist you in the understanding of this application, the necessary steps for an SNP/TFBS analysis are exercised by the examples “KIAA1754L” (short sequence) and "KIAA0960" (long sequence), whose text files are published on the SNPsel homepage.

2.1. FASTA base sequence

On the tabpage “FASTA” you can find all informations concerning the FASTA base sequences. The original FASTA base sequence is the basis for the generation of allelic FASTA sequences, which can be exported on this tabpage.

The allelic base sequences can be exported as plus or minus oriented FASTA sequences (conforming to the Ensembl specification), but you should only use equal oriented sequences to get accurate prediction results. Both during the import and the export of informations, the usual dialogue windows appear to select the directory, that holds all necessary files for comparision analysis by SNPsel.

Tutorial: Please import the FASTA text file (coded conforming to the Ensembl standard) "KIAA0960.fasta" available in the directory "Examples/KIAA0960/Import" by clicking on the menu option "FASTA base sequence" → "import" within the FASTA tabpage. A dialogue window appears, in which you navigate to the mentioned directory, select the appropriate text file and click onto the button "open".

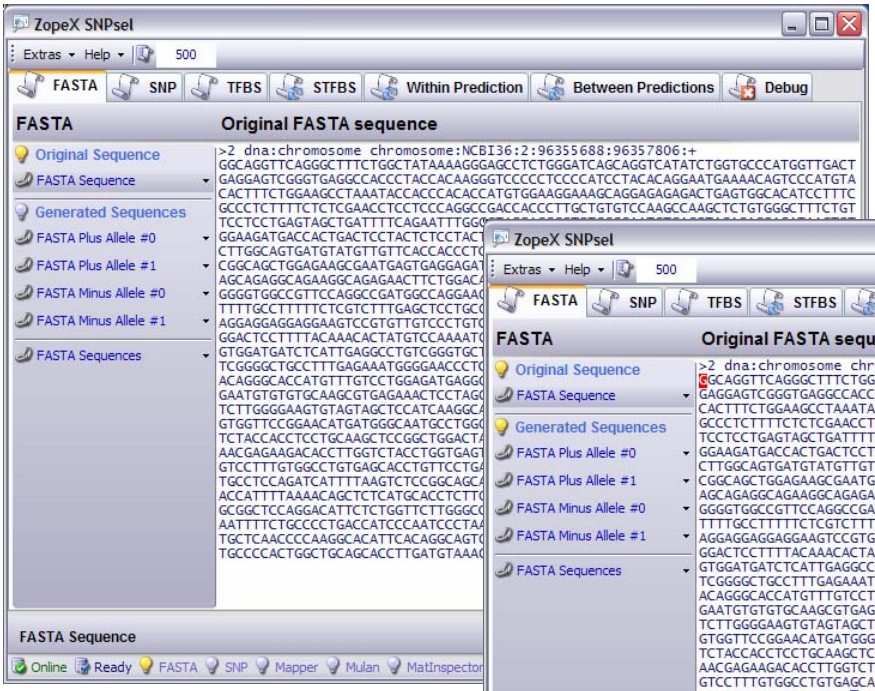


Figure: FASTA without SNPs

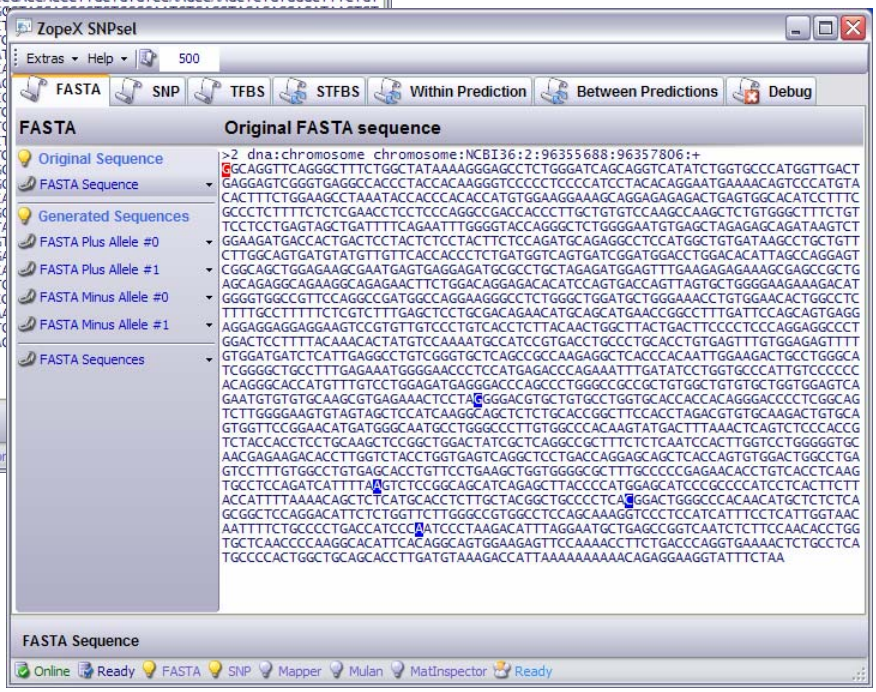


Figure: FASTA with marked SNPs (KIAA1754L)

As soon as the original FASTA sequence and the SNPs collection (see in addition chapter 2.2) were imported, the generated FASTA sequences are available for the export. The bases, which are replaced by appropriate alleles of SNPs, are blue marked. The red-marked base is only an orienting point, signalizing the first base at the beginning of this base sequence. Note, that the ">..." FASTA information line has to be completely indicated within the first line.

Important: If this information line is wrapped over several text lines, the red marking can slip into the information line. As soon as this red marking neither mark the first base of the sequence, the blue colored bases do not mark bases, which are affected by SNP alleles!

2.2. Single Nucleotide Polymorphisms (SNP)

The second tabpage "SNPs" contains the imported SNP collection, which are necessary to generate the allelic sequences and later to identify the TFBS potentially affected by SNPs (STFBS). The SNPs appears as tabular representation with the information columns containing SNP name, chromosome, sequence position, strand orientation, allelic bases [left allele /right allele] and SNP description.

Tutorial: Export the two generated FASTA sequences into the directory "Example/KIAA0960/Export", doing the following steps:

- Go back to the tabpage "FASTA" and click the menu option "FASTA plus allele #0" -> "export".
- A dialogue window appears, with which you export the file as "KIAA0960 plusallel0.fasta" into the directory "examples/KIAA0960/Export".
- Export the second FASTA sequence as "KIAA0960 plusallel1.fasta" named file by clicking the menu option "FASTA plus allele #1" and selecting the directory "Beispiele/KIAA0960/Export" in the appearing dialogue window.

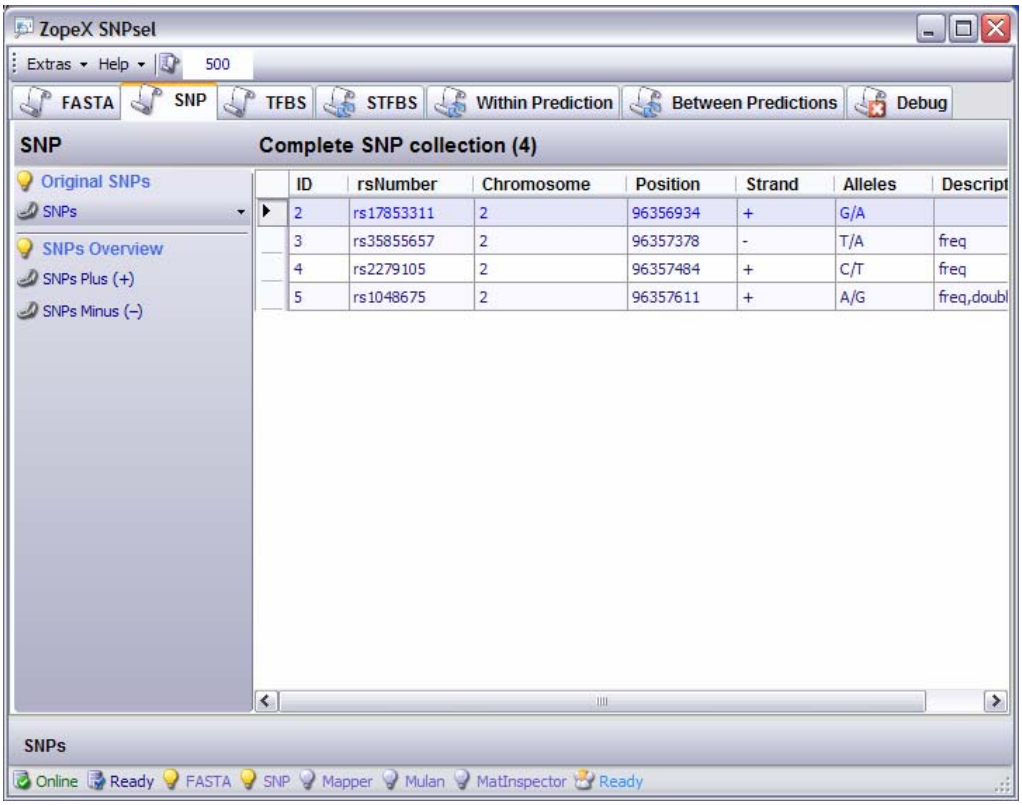


Figure: Imported SNP collection (KIAA1754L)

2.3. Transkriptionsfaktor-Bindestellen (TFBS)

Since the TFBS plays an important role for further analysis, the two generated allelic sequences are used as information basis for external TFBS prediction applications. he SNPsel application is already appropriate for directly importing of TFBS predictions from web based applications "Mapper", "Mulan/multiTF" and "MatInspector". f you are not certain whether the prediction result is in the correct tab-delimited text format, you can open the appropriate (provided) TFBS files from the example "KIAA0960" into a text editor, in order to be able to compare the both formats.

Tutorial: Use the web based programs "Mapper", "Mulan/multiTF" and "MatInspector" to get TFBS predictions for the both allelic base sequences generated by SNPsel:

- Click on menu option "Help" - "Info" to open the information dialoge window, wich contains the links of the TFBS prediction programs "Mapper", "Mulan/multiTF" and "MatInspector" in the paragraph "web applications". By clicking on the link, a new web browser window appears with the right web application.
- Use the both allelic FASTA sequences "KIAA0960 plusallel0.fasta" and "KIAA0960 plusallel0.fasta" to get two different TFBS predictions and save them on disc.
- Store the both TFBS predictions from "Mapper" as tab-delimited text files on disc using the names "KIAA0960 plusallel0.mapper" and "KIAA0960 plusallel1.mapper" with the file extension ".mapper".
- Store the TFBS predictions from "Mulan/multiTF" as tab-delimited text files on disc using the names "KIAA0960 plusallel0.mulan" and "KIAA0960 plusallel1.mulan" with the file extension ".mulan".
- Store the TFBS predictions from "MatInspector" as HTML files, each containing an overlapping range of the FASTA base sequence with the positions [0-20.000], [19.900-40.000] and [39.900-60.000] in the same directory. Name these files „KIAA0960 plusallel0_1-20000.matinspector“, „KIAA0960 plusallel0_19900-40000.matinspector“ and „KIAA0960 plusallel0_39900-60000.matinspector“.
- Save the analogue TFBS prediction made on the alternate allelic version of the base sequence as blocks of same ranges [0-20.000], [19.900-40.000] and [39.900-60.000], and name them „KIAA0960 plusallel1_1-20000.matinspector“, „KIAA0960 plusallel1_19900-40000.matinspector“ and „KIAA0960 plusallel1_39900-60000.matinspector“.

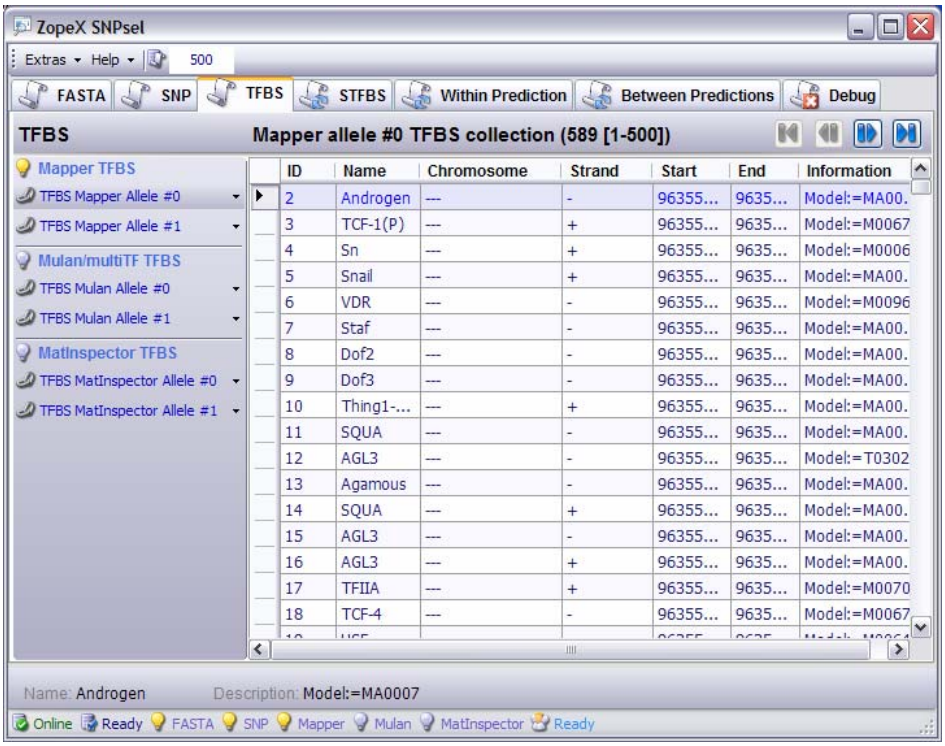
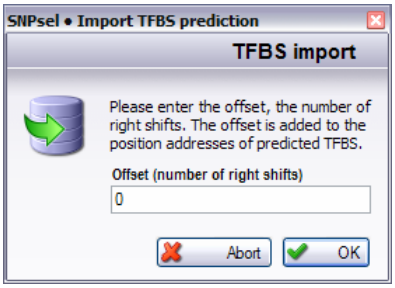


Figure: Imported Mapper TFBS prediction for sequence KIAA1754L considering the left SNP alleles

Since the TFBS comparisons are accomplished by SNPsel with the support of the database server, once imported information can be tabulary displayed again, by clicking directly on the appropriate menu option. You should wait a moment, until the tabular representation was transmitted from the database server.

The TFBS are listed following split-oriented information tabular listed: Name of the TFBS, the chromosom concerned with the orientation of the TFBS, the there absolute starting and final position and a closer description. The content of the tabpages can be exported as tab-delimited text file according to the tabular representation.

The description (column "information") contains additionally informations provided by the TFBS prediction program, like "Model" from "Mapper", the complete name of the "TFBS" and its sequence cutout "Sequence" from "Mulan/multiTF", and the complete description of the TFBS "Name" with the three values for "Opt.", "core sim." and "matrix sim." from "MatInspector".



As soon as both TFBS predictions (supplied by the prediction program) were imported, the appropriate lamp on the left side of the tabpage and the lamp in the lower statusbar shine. Consider, that the TFBS prediction generated by "MatInspector" is limited to the first 5.000 TFBS, when you use the academic edition of this program at no costs. In order to around this limitation, you must split your TFBS analysis into blocks on the way that you obtain predictions with maximally 5.000 TFBS. That's why you can indicate the shift of the TFBS positions (the offset) for repeated import of predicted TFBS. For example, if you import TFBS with an offset of 1 all TFBS positions are incremented (or shifted) by 1.

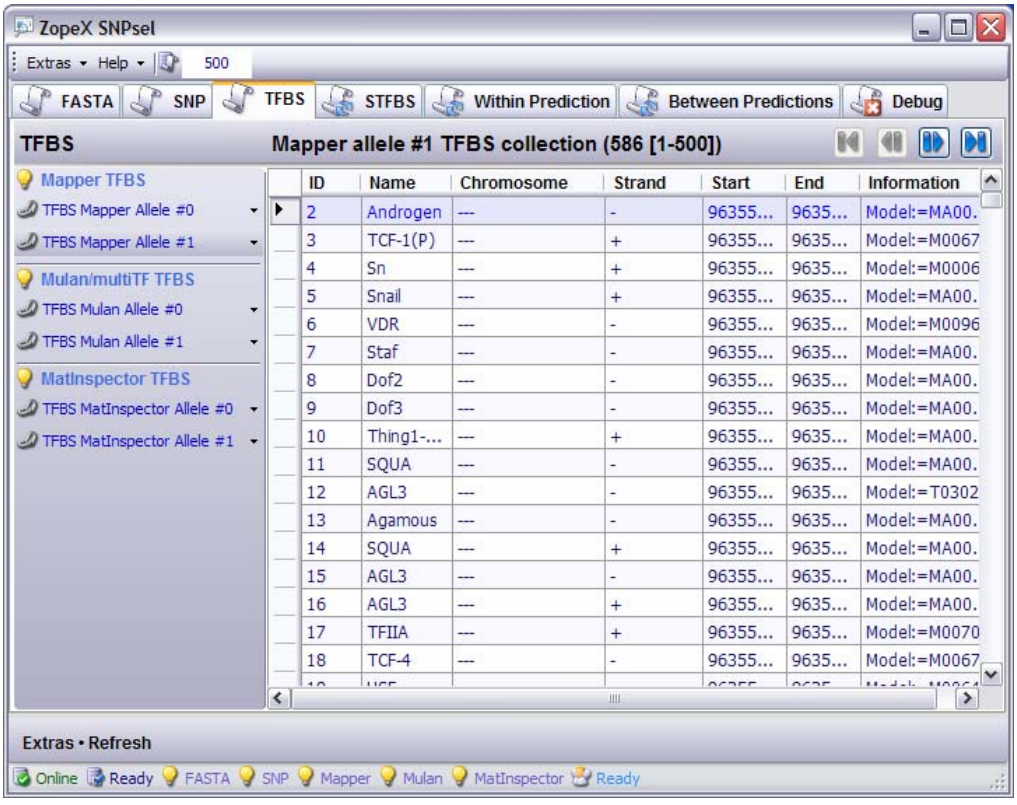


Figure: Imported Mapper TFBS prediction for the sequence KIAA1754L considering the right SNP alleles

3. Execution of comparisons

3.1. Potentially SNP affected TFBS (STFBS)

SNP influenceable TFBS (further simply called STFBS) are TFBS containing one SNP at least. Since a SNP features two different alleles, the interesting question is: Which TFBS is recognized (if such existed at all), if the allele of the SNPs is replaced by the other one. Therefore both available TFBS predictions from the same web based application are compared to identify SNP affected TFBS and not affected STFBS. Uninfluencable STFBS are such TFBS, which are equal on each of both allelic versions of the base sequence containing SNPs. Affected STFBS are such, which are different on each allelic base sequence.

An STFBS comparing can take place only if both TFBS predictions of the same program are imported in SNPsel. The comparing must be manually started by you, because this analysis can take up some time, depending upon the count of the imported TFBS and the available resources of the database server.

- **Tutorial:** After you imported the TFBS predictions of the programs „Mapper“, „Mulan/multiTF “and „MatInspector“, you can start the TFBS analysis manually, by clicking on the menu option “extras” → „refresh“ in the upper menubar. As soon as the analysis was started, the red word „busy“ appears in the lower statusbar of the program. If the analysis is finished, then it switches to the green word „finished“.

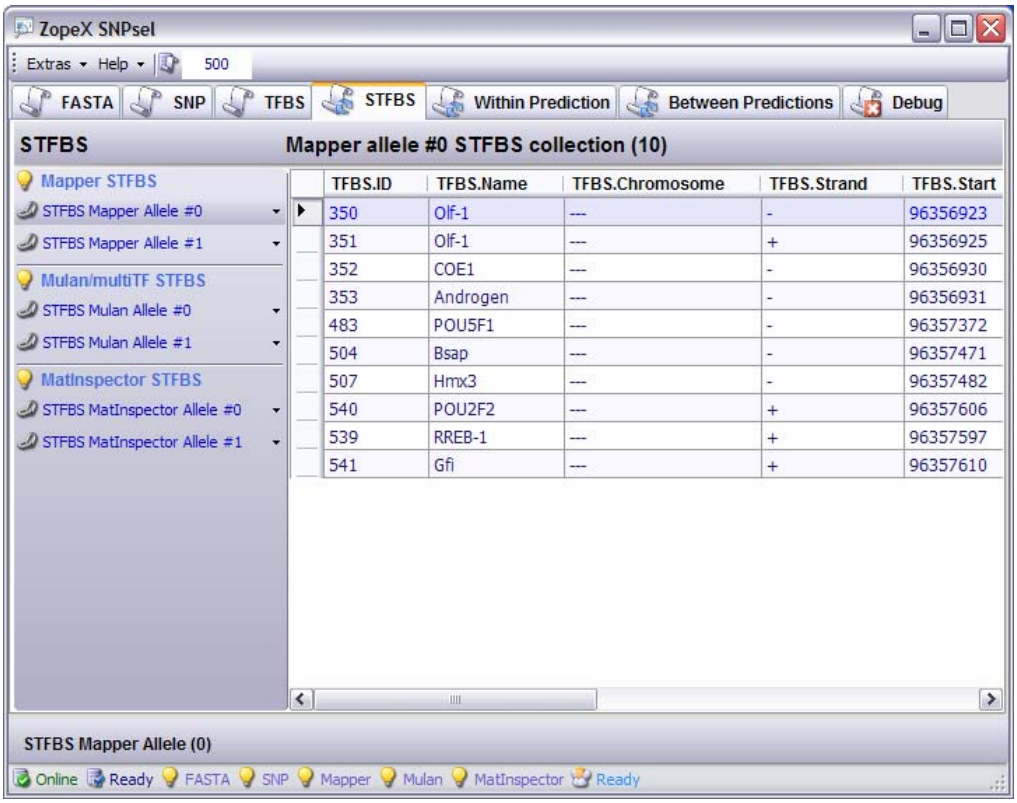


Figure: Identified STFBS in the Mapper TFBS prediction for sequence KIAA1754L considering only the left SNP Alleles

The tabular overview of found STFBS contains the following split-oriented informations about a TFBS (name, chromosome, strand, absolute starting and end position and additional information) followed by the informations about the included SNP (name, strand, absolute position and both SNP alleles).

Tutorial: After you imported the code sequence, the SNP collection and the different TFBS predictions and accomplished the TFBS analysis, you can find in the tabular overview of the tabpage „STFBS“ all predicted TFBS for both allelic base sequences (generated by SNPsel and used for external TFBS prediction), which contains all at least a SNP in their code region.

3.2. STFBS Comparisions

3.2.1. Unaffected STFBS

Identical STFBS designates SNP influenceable TFBS, which has absolutely identical start and end positions, and exists with both alleles of a SNP. Therefore both TFBS predictions from same prediction application are compared with one another, to taken up only the TFBS to the tabular representation, which suit the following characteristic:

The TFBS with the same start and end positions contains both alleles of the same SNP.

If ***T0*** and ***T1*** are both TFBS with different alleles of the same SNP, then they contains at least a SNP in their code regions [***T0.start***…***T0.end***] and [***T1.start***…***T1.end***], whereby ***T0.start = T1.start*** and ***T0.strand = T1 strand*** and ***T0.end = T1 end***.

Tutorial: If you click within the tabpage „comparison“ a menu option of the category „unaffected STFBS “, you will see in the tabular overview all TFBS, which are not affected by their contained SNP.

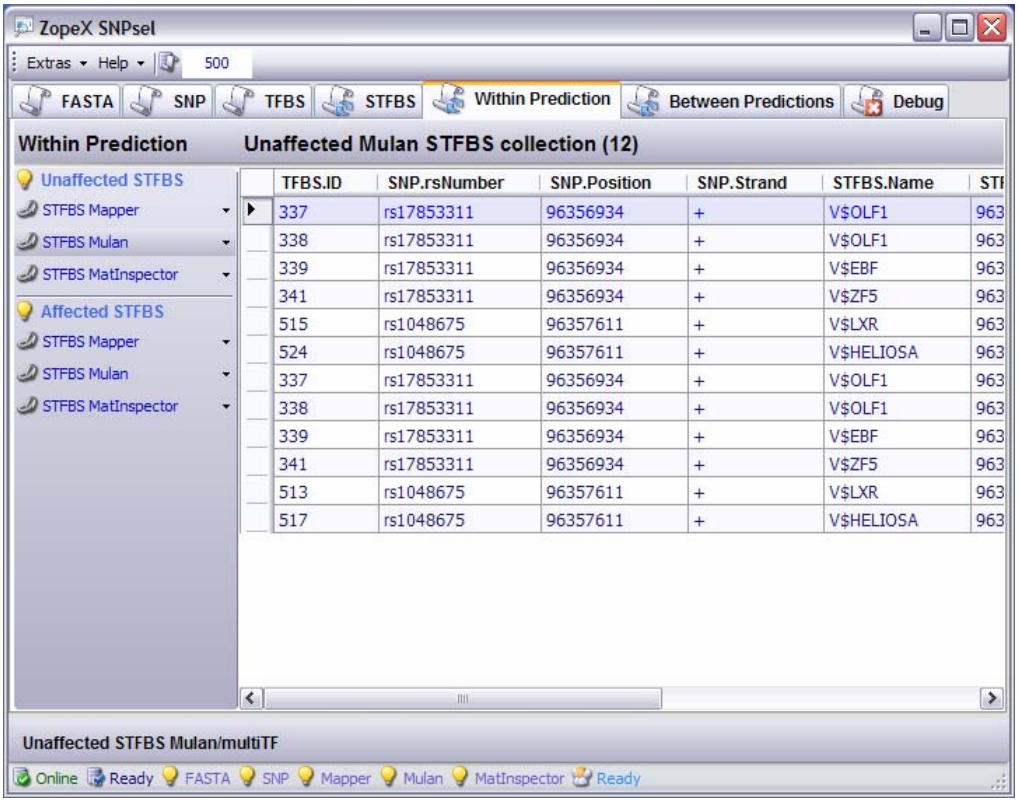


Figure: Overview of identified unaffected STFBS from the Mulan TFBS prediction which contains a SNP at least and exist for both SNP alleles

3.2.2. Affected STFBS

Affected STFBS designates influenceable TFBS, which exists with both alleles of the same SNP and differs at least in the start and/or end positions. Both TFBS predictions of same application are compared, and only these STFBS are taken up to the tabular representation, which obey to the following characteristic:

The TFBS with different start and/or end positions contains different alleles of the same SNP.

If ***T0*** and ***T1*** are both TFBS including different alleles of the same SNP, then their code regions overlap and it is ***T0.start ≠ T1.start*** and/or ***T0.end ≠ T1.end***, whereby additionally applies: ***T0.strand = T1.strand*** and it does not apply for ***T0.end < T1.start*** or ***T1.end < T0.start***.

Additionally there can be situations, in which only one STFBS exists. Such STFBS designates also influenceable TFBS, which exists only with one allele of a SNP. Both TFBS predictions of same application are compared and only those TFBS are taken up to the tabular representation, which have following characteristic:

It exists only a TFBS containing one allele of a SNP and no other TFBS, which contains the other allele of the same SNP. If ***T0*** is the TFBS, which contains an allele of a SNP, then it exists no TFBS ***T1***, which contains the other allele of the same SNP.

Tutorial: If you click within the tabpage „comparison“ a menu option of the category „affected STFBS “, you will see in the tabular overview all TFBS, which are affected by their contained SNP.

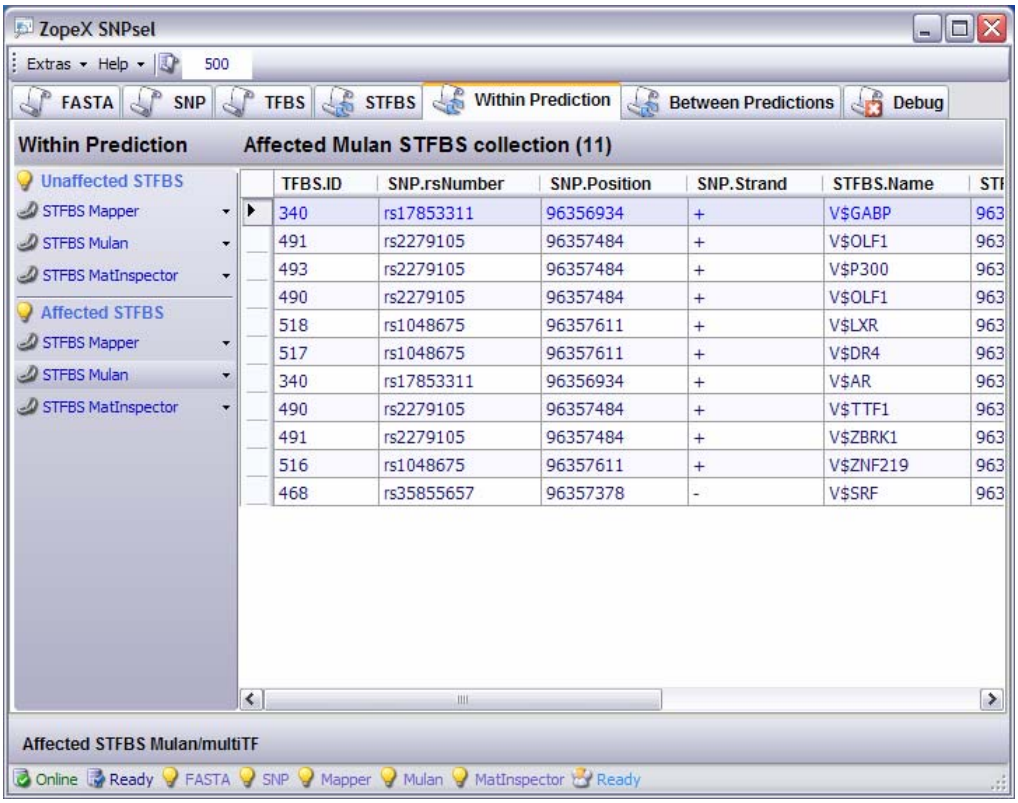


Figure: Overview of identified affected STFBS from the Mapper TFBS prediction which contains a SNP at least and exist only for one of both SNP alleles

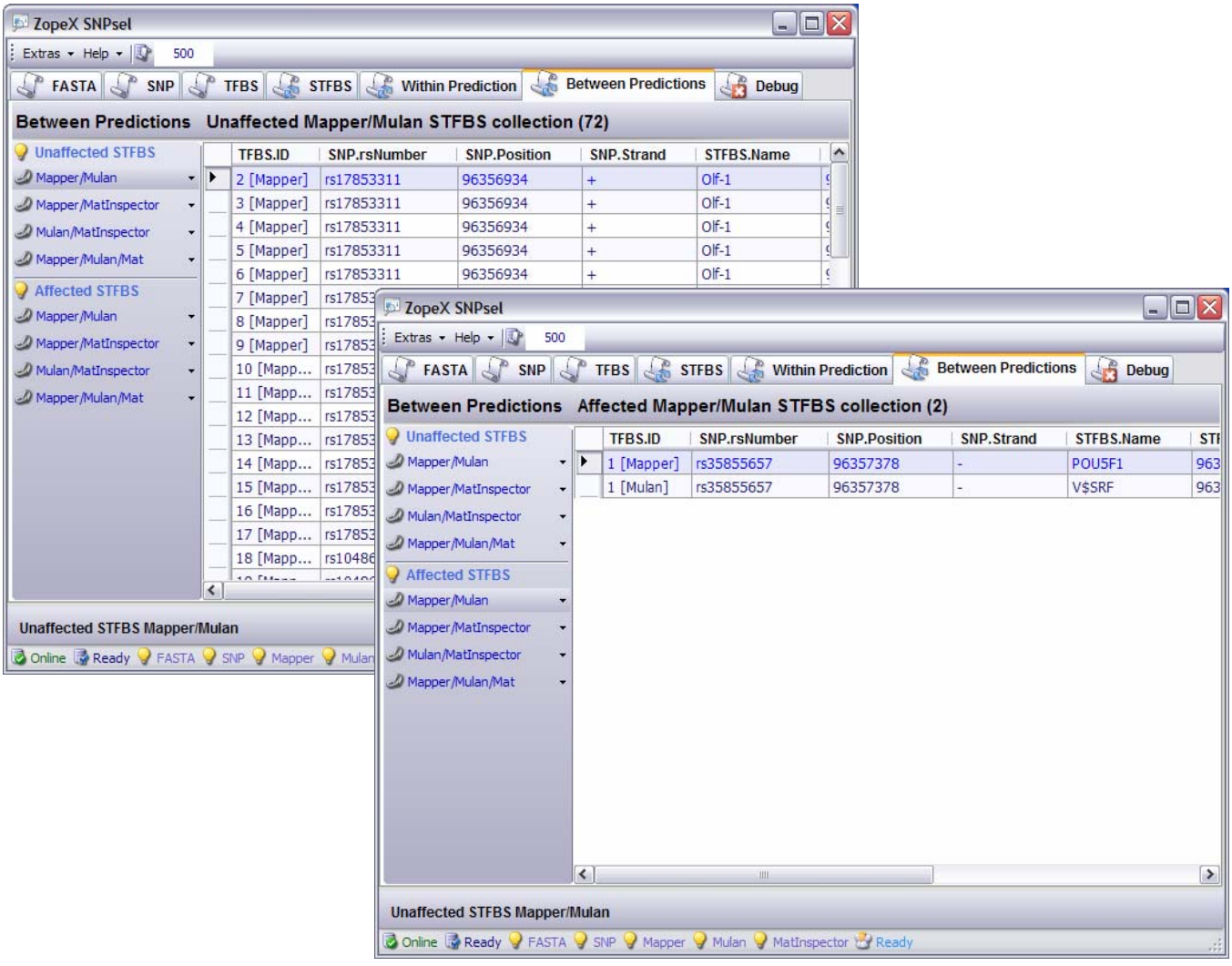
3.3. Same predictions from different applications

3.3.1. Same predictions of unaffected STFBS

With the comparison of the TFBS predictions of different applications, only those TFBS are indicated, which are part of the comparison result of unaffected STFBS and which are discoverable in the predictions - which were generated by different applications (for example by „Mapper“) - on the basis of identical included SNPs.

3.3.2. Same predictions of affected STFBS

By comparing the TFBS predictions of different applications, only those TFBS are indicated, which are part of the comparison result of affected STFBS and which are discoverable in the predictions - which were generated by different applications (for example by „Mapper“) - on the basis of identical included SNPs.



Left: Overview of unaffected STFBS from Mapper and Mulan TFBS predictions, which contains the same SNPs
Right: Collection of affected STFBS from Mapper and Mulan TFBS predictions, which contains the same SNPs

4. SQL commands

SNPsel uses the MySQL Community Server as database management system to perform the identification of SNP affected TFBS and comparisions of these STFBS, which TFBS were originally generated by different prediction tools. Now, you can find all SQL commands used by SNPex in the printed overview.

4.1. Database schema

```
# Create the SNPsel database
create database SNPsel;

# Create table "Times", which contains table name and timestamp, indicating the last write access.
# This table is internally used by SNPsel.
create table SNPsel.Times (Tabelle varchar(50),Ereignis timestamp,index(Tabelle));

# Initialize table "Times" with used table names and actual timestamp.
insert into SNPsel.Times (Tabelle,Ereignis) values ('FASTA',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('IUPAC',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('Plus0',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('Plus1',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('Minus0',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('Minus1',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('SNPS',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MapperTFBS0',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MapperTFBS1',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MulanTFBS0',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MulanTFBS1',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MatInspectorTFBS0',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MatInspectorTFBS1',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MapperSTFBS0',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MapperSTFBS1',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MulanSTFBS0',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MulanSTFBS1',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MatInspectorSTFBS0',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MatInspectorSTFBS1',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MapperSTFBSEqual',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MulanSTFBSEqual',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MatInspectorSTFBSEqual',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MapperSTFBSDifferent',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MulanSTFBSDifferent',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MatInspectorSTFBSDifferent',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MapperSTFBSMissing',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MulanSTFBSMissing',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MatInspectorSTFBSMissing',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MapperMulanSTFBSEqual',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MapperMatSTFBSEqual',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MulanMatSTFBSEqual',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MapperMulanMatSTFBSEqual',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MapperMulanSTFBSDifferent',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MapperMatSTFBSDifferent',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MulanMatSTFBSDifferent',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MapperMulanMatSTFBSDifferent',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MapperMulanSTFBSMissing',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MapperMatSTFBSMissing',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MulanMatSTFBSMissing',Now());
insert into SNPsel.Times (Tabelle,Ereignis) values ('MapperMulanMatSTFBSMissing',Now());
```

Create table "***FASTA***".
```
create table SNPsel.FASTA (ID int unsigned auto_increment unique,Art varchar(30),Information varchar(300),
CodeStream mediumtext);
```

Create table "***SNPS***".
```
create table SNPsel.SNPS (ID int unsigned auto_increment unique,Name varchar(100),Chromosome varchar(10),
Position int,Strand varchar(1),Alleles varchar(10),Description varchar(300),index(Name),index(Position),index(Strand));
```

Create table "MapperTFBS0", containing ***TFBS*** prediction generated by Mapper for sequence #0.
```
create table SNPsel.MapperTFBS0 (ID int unsigned auto_increment unique,Name varchar(100),
Chromosome varchar(10),Strand varchar(1),Anfang int,Ende int,Description varchar(300),Score float,
Eval float,index(Anfang),index(Ende),index(Strand));
```
Create table "MapperTFBS1", containing TFBS prediction generated by Mapper for sequence #1.
```
create table SNPsel.MapperTFBS1 (ID int unsigned auto_increment unique,Name varchar(100),
Chromosome varchar(10),Strand varchar(1),Anfang int,Ende int,Description varchar(300),Score float,
Eval float,index(Anfang),index(Ende),index(Strand));
```

Create other ***TFBS*** database tables.
```
create table 'MulanTFBS0'","create table SNPsel.MulanTFBS0 (ID int unsigned auto_increment unique,Name
varchar(100),Chromosome varchar(10),Strand varchar(1),Anfang int,Ende int,Description varchar(300),Score float,Eval
float,index(Anfang),index(Ende),index(Strand));
create table 'MulanTFBS1'","create table SNPsel.MulanTFBS1 (ID int unsigned auto_increment unique,
Name varchar(100),Chromosome varchar(10),Strand varchar(1),Anfang int,Ende int,Description varchar(300),Score float,
Eval float,index(Anfang),index(Ende),index(Strand));
create table SNPsel.MatInspectorTFBS0 (ID int unsigned auto_increment unique,Name varchar(100),
Chromosome varchar(10),Strand varchar(1),Anfang int,Ende int,Description varchar(300),Score float,
Eval float,index(Anfang),index(Ende),index(Strand));
create table SNPsel.MatInspectorTFBS1 (ID int unsigned auto_increment unique,Name varchar(100),
Chromosome varchar(10),Strand varchar(1),Anfang int,Ende int,Description varchar(300),Score float,
Eval float,index(Anfang),index(Ende),index(Strand));
```

Insert ***default values*** in tables, which indicates a non-existing Item (required to find missing items).
```
insert into SNPsel.SNPS (Name,Chromosome,Position,Strand,Alleles,Description) values ('---','---',0,'?','---','Missing');
insert into SNPsel.MapperTFBS0 (Name,Chromosome,Strand,Anfang,Ende,Description,Score,Eval) values ('---','---','?',0,0,
'Missing',0,0);
insert into SNPsel.MapperTFBS1 (Name,Chromosome,Strand,Anfang,Ende,Description,Score,Eval) values ('---','---','?',0,0,
'Missing',0,0);
insert into SNPsel.MulanTFBS0 (Name,Chromosome,Strand,Anfang,Ende,Description,Score,Eval) values ('---','---','?',0,0,
'Missing',0,0);
insert into SNPsel.MulanTFBS1 (Name,Chromosome,Strand,Anfang,Ende,Description,Score,Eval) values ('---','---','?',0,0,
'Missing',0,0);
insert into SNPsel.MatInspectorTFBS0 (Name,Chromosome,Strand,Anfang,Ende,Description,Score,Eval) values ('---','---',
'?',0,0,'Missing',0,0);
insert into SNPsel.MatInspectorTFBS1 (Name,Chromosome,Strand,Anfang,Ende,Description,Score,Eval) values ('---','---',
'?',0,0,'Missing',0,0);
```

Set the ***ID number*** of these "non-existing" items to zero.
```
update SNPsel.SNPS set ID=0 where Name='---';
update SNPsel.MapperTFBS0 set ID=0 where Name='---';
update SNPsel.MapperTFBS1 set ID=0 where Name='---';
update SNPsel.MulanTFBS0 set ID=0 where Name='---';
update SNPsel.MulanTFBS1 set ID=0 where Name='---';
update SNPsel.MatInspectorTFBS0 set ID=0 where Name='---';
update SNPsel.MatInspectorTFBS1 set ID=0 where Name='---';
```

```
# Create table “MapperSTFBS0”, containing ***STFBS*** (SNP affected TFBS) from Mapper prediction
# for sequence #0.
# The following tables contains ***only references*** (ID numbers) to database items. So, the each item
# from table “MapperSTFBS0” contains respectively a reference to the SNP and to the affected TFBS.
create table SNPsel.MapperSTFBS0 (ID int unsigned auto_increment unique,TFBS_ID int,SNP_ID int,index(TFBS_ID),
index(SNP_ID));

# Create other ***STFBS*** database tables.
create table SNPsel.MapperSTFBS1 (ID int unsigned auto_increment unique,TFBS_ID int,SNP_ID int,index(TFBS_ID),
index(SNP_ID));
create table SNPsel.MulanSTFBS0 (ID int unsigned auto_increment unique,TFBS_ID int,SNP_ID int,index(TFBS_ID),
index(SNP_ID));
create table SNPsel.MulanSTFBS1 (ID int unsigned auto_increment unique,TFBS_ID int,SNP_ID int,index(TFBS_ID),
index(SNP_ID));
create table SNPsel.MatInspectorSTFBS0 (ID int unsigned auto_increment unique,TFBS_ID int,SNP_ID int,index(TFBS_ID),
index(SNP_ID));
create table SNPsel.MatInspectorSTFBS1 (ID int unsigned auto_increment unique,TFBS_ID int,SNP_ID int,index(TFBS_ID),
index(SNP_ID));

# Insert ***default values*** into tables, indicating a default relation between “non-existing” SNP and TFBS.
insert into SNPsel.MapperSTFBS0 (TFBS_ID,SNP_ID) values (0,0);
insert into SNPsel.MapperSTFBS1 (TFBS_ID,SNP_ID) values (0,0)
insert into SNPsel.MulanSTFBS0 (TFBS_ID,SNP_ID) values (0,0)
insert into SNPsel.MulanSTFBS1 (TFBS_ID,SNP_ID) values (0,0)
insert into SNPsel.MatInspectorSTFBS0 (TFBS_ID,SNP_ID) values (0,0)
insert into SNPsel.MatInspectorSTFBS1 (TFBS_ID,SNP_ID) values (0,0)

# Set ***ID numbers*** of “non-existing” STFBS to zero.
update SNPsel.MapperSTFBS0 set ID=0 where SNP_ID=0;
update SNPsel.MapperSTFBS1 set ID=0 where SNP_ID=0;
update SNPsel.MulanSTFBS0 set ID=0 where SNP_ID=0;
update SNPsel.MulanSTFBS1 set ID=0 where SNP_ID=0;
update SNPsel.MatInspectorSTFBS0 set ID=0 where SNP_ID=0;
update SNPsel.MatInspectorSTFBS1 set ID=0 where SNP_ID=0;

# Create tables for ***comparing STFBS*** generated by same prediction tool on basis of one of both transmitted
# sequences (containing only left (sequence #0) or only right (sequence #1) SNP alleles).
# Create tables containing ***STFBS***, that are absolutely ***equal*** (no difference between both STFBS)
create table SNPsel.MapperSTFBSEqual (ID int unsigned auto_increment unique,STFBS0_ID int,STFBS1_ID int,
SNP_ID int,index(STFBS0_ID),index(STFBS1_ID),index(SNP_ID));
create table SNPsel.MulanSTFBSEqual (ID int unsigned auto_increment unique,STFBS0_ID int,STFBS1_ID int,
SNP_ID int,index(STFBS0_ID),index(STFBS1_ID),index(SNP_ID));
create table SNPsel.MatInspectorSTFBSEqual (ID int unsigned auto_increment unique,STFBS0_ID int,STFBS1_ID int,
SNP_ID int,index(STFBS0_ID),index(STFBS1_ID),index(SNP_ID));

# Create tables containing (***different***) ***STFBS***, affected by the same SNP, but overlaps in their coderegion.
create table SNPsel.MapperSTFBSDifferent (ID int unsigned auto_increment unique,STFBS0_ID int,STFBS1_ID int,
SNP_ID int,index(STFBS0_ID),index(STFBS1_ID),index(SNP_ID));
create table SNPsel.MulanSTFBSDifferent (ID int unsigned auto_increment unique,STFBS0_ID int,STFBS1_ID int,
SNP_ID int,index(STFBS0_ID),index(STFBS1_ID),index(SNP_ID));
create table SNPsel.MatInspectorSTFBSDifferent (ID int unsigned auto_increment unique,STFBS0_ID int,
STFBS1_ID int,SNP_ID int,index(STFBS0_ID),index(STFBS1_ID),index(SNP_ID));
```

Create tables containing (***missing***) ***STFBS***, that have no corresponding TFBS on the allelic sequence.
```
create table SNPsel.MapperSTFBSMissing (ID int unsigned auto_increment unique,STFBS0_ID int,STFBS1_ID int,
SNP_ID int,index(STFBS0_ID),index(STFBS1_ID),index(SNP_ID));
create table SNPsel.MulanSTFBSMissing (ID int unsigned auto_increment unique,STFBS0_ID int,STFBS1_ID int,
SNP_ID int,index(STFBS0_ID),index(STFBS1_ID),index(SNP_ID));
create table SNPsel.MatInspectorSTFBSMissing (ID int unsigned auto_increment unique,STFBS0_ID int,
STFBS1_ID int,SNP_ID int,index(STFBS0_ID),index(STFBS1_ID),index(SNP_ID));
```

Create tables containing STFBS from ***two*** different prediction tools, identified by same comparision
method (Equal, Different or Missing).
```
create table SNPsel.MapperMulanSTFBSEqual (ID int unsigned auto_increment unique,MapperID int,
MulanID int,index(MapperID),index(MulanID));
create table SNPsel.MapperMatSTFBSEqual (ID int unsigned auto_increment unique,MapperID int,
MatInspectorID int,index(MapperID),index(MatInspectorID));
create table SNPsel.MulanMatSTFBSEqual (ID int unsigned auto_increment unique,MulanID int,
MatInspectorID int,index(MulanID),index(MatInspectorID));
create table SNPsel.MapperMulanSTFBSDifferent (ID int unsigned auto_increment unique,MapperID int,
MulanID int,index(MapperID),index(MulanID));
create table SNPsel.MapperMatSTFBSDifferent (ID int unsigned auto_increment unique,MapperID int,
MatInspectorID int,index(MapperID),index(MatInspectorID));
create table SNPsel.MulanMatSTFBSDifferent (ID int unsigned auto_increment unique,MulanID int,
MatInspectorID int,index(MulanID),index(MatInspectorID));
create table SNPsel.MapperMulanSTFBSMissing (ID int unsigned auto_increment unique,MapperID int,
MulanID int,index(MapperID),index(MulanID));
create table SNPsel.MapperMatSTFBSMissing (ID int unsigned auto_increment unique,MapperID int,
MatInspectorID int,index(MapperID),index(MatInspectorID));
create table SNPsel.MulanMatSTFBSMissing (ID int unsigned auto_increment unique,MulanID int,MatInspectorID
int,index(MulanID),index(MatInspectorID));
```

Create tables containing STFBS from ***three*** different prediction tools, identi identified by same comparision
method (Equal, Different or Missing).
```
create table SNPsel.MapperMulanMatSTFBSEqual (ID int unsigned auto_increment unique,MapperID int,MulanID
int,MatInspectorID int,index(MapperID),index(MulanID),index(MatInspectorID));
create table SNPsel.MapperMulanMatSTFBSDifferent (ID int unsigned auto_increment unique,MapperID int,MulanID
int,MatInspectorID int,index(MapperID),index(MulanID),index(MatInspectorID));
create table SNPsel.MapperMulanMatSTFBSMissing (ID int unsigned auto_increment unique,MapperID int,MulanID
int,MatInspectorID int,index(MapperID),index(MulanID),index(MatInspectorID));
```

4.2. Read from database tables

Read a ***SNP*** using a where condition (like “ID=1” or “Name like rs%”).
```
select ID,Name,Chromosome,Position,Strand,Alleles,Description from SNPsel.SNPS where [where condition];
```

Read a ***TFBS*** using a where condition.
Replace [TFBS] by name of prediction tool (Mapper, Mulan or MatInspector)
and [Version] by the number of considered sequence (0 or 1).
```
select ID,Name,Chromosome,Strand,Anfang,Ende,Description,Score,Eval
from SNPsel.[TFBS]TFBS[Version] where [where condition];
```

Read a ***STFBS*** using a where condition.
Replace [TFBS] by name of prediction tool (Mapper, Mulan or MatInspector)
and [Version] by the number of considered sequence (0 or 1).
```
select TFBS.ID,TFBS.Name,TFBS.Chromosome,TFBS.Strand,TFBS.Anfang,TFBS.Ende,
    TFBS.Description,TFBS.Score,TFBS.Eval, SNP.Name,SNP.Strand,SNP.Position,SNP.Alleles
from SNPsel.[TFBS]STFBS[Version] as STFBS, SNPsel.[TFBS]TFBS[Version] as TFBS, SNPsel.SNPS as SNP
where TFBS.ID=STFBS.TFBS_ID and SNP.ID=STFBS.SNP_ID and ([where condition]);
```

4.3. Identifying SNP affected TFBS (STFBS)

```
# Insert SNP affected TFBS into STFBS database table.
# Replace the [TFBS] and [Version] by appropriate name of prediction tool (Mapper, Mulan or MatInspector)
# and number of sequence (0 or 1) to get the complete name of referenced database table.
insert into SNPsel.[TFBS]STFBS[Version](TFBS_ID,SNP_ID)
    select TFBS.ID,SNPS.ID from SNPsel.SNPS,SNPsel.[TFBS]TFBS[Version] as TFBS
    where TFBS.Anfang<SNPS.Position and TFBS.Ende>SNPS.Position;

# Read SNP affected TFBS from STFBS database table using references.
select TFBS.ID,TFBS.Name,TFBS.Chromosome,TFBS.Strand,TFBS.Anfang,TFBS.Ende,
    TFBS.Description,TFBS.Score,TFBS.Eval,SNP.Name,SNP.Strand,SNP.Position,SNP.Alleles
from SNPsel.[TFBS]STFBS[Version] as STFBS,SNPsel.[TFBS]TFBS[Version] as TFBS,SNPsel.SNPS as SNP
where TFBS.ID=STFBS.TFBS_ID and SNP.ID=STFBS.SNP_ID;
```

4.4. Comparing allelic STFBS from same prediction tool

```
# Read compared STFBS from database table.
# Replace [TFBS] by the name of used prediction tools (Mapper, Mulan or MatInspector)
# and [Modus] by the comparision method (Equal, Different or Missing).
select Equal.ID,T0.Name,T0.Anfang,T0.Ende,T0.Description,T0.Score,T0.Eval,
    T1.Name,T1.Anfang,T1.Ende,T1.Description,T1.Score,T1.Eval,
    SNP.Name,SNP.Chromosome,SNP.Position,SNP.Strand,SNP.Alleles
from SNPsel.[TFBS]STFBS[Modus] as Equal,SNPsel.[TFBS]TFBS0 as T0,SNPsel.[TFBS]TFBS1 as T1,SNPsel.SNPS as SNP
where T0.ID=(
    select ST0.TFBS_ID from SNPsel.[TFBS]STFBS0 as ST0 where ST0.ID=Equal.STFBS0_ID
) and T1.ID=(
    select ST1.TFBS_ID from SNPsel.[TFBS]STFBS1 as ST1 where ST1.ID=Equal.STFBS1_ID
) and SNP.ID=Equal.SNP_ID;
```

4.5. Equal comparision

```
# Insert compared STFBS into STFBS database table, which are absolutely equal.
# Insert allelic STFBS from same prediction tool, generated and identified using both sequences (#0 and #1),
# which have identical allelic counterpart. (This means TFBS, that have absolutely equal TFBS on the allelic
# sequence affected by the same SNP).
# Replace [TFBS] by the name of used prediction tools (Mapper, Mulan or MatInspector).
insert into SNPsel.[TFBS]STFBSEqual(STFBS0_ID,STFBS1_ID,SNP_ID)
select STFBS0.ID,STFBS1.ID,STFBS0.SNP_ID from SNPsel.[TFBS]STFBS0 as STFBS0,SNPsel.[TFBS]STFBS1 as STFBS1
where STFBS0.SNP_ID=STFBS1.SNP_ID and exists (
    select * from SNPsel.[TFBS]TFBS0 as TFBS0,SNPsel.[TFBS]TFBS1 as TFBS1
    where TFBS0.ID=STFBS0.TFBS_ID and TFBS1.ID=STFBS1.TFBS_ID and TFBS0.Strand=TFBS1.Strand
    and TFBS0.Anfang=TFBS1.Anfang and TFBS0.Ende=TFBS1.Ende
);
```

4.6. Different comparision

```
# Insert compared STFBS into STFBS database table, which are absolutely different.
# Insert allelic STFBS from same prediction tool, generated and identified using both sequences (#0 and #1),
# which have overlapping allelic counterpart. (This means TFBS, that have TFBS with overlapping coderegion
# on the allelic sequence affected by the same SNP).
# Replace [TFBS] by the name of used prediction tools (Mapper, Mulan or MatInspector).
insert into SNPsel.[TFBS]STFBSDifferent(STFBS0_ID,STFBS1_ID,SNP_ID) (
    select STFBS0.ID,0,STFBS0.SNP_ID from SNPsel.[TFBS]STFBS0 as STFBS0,SNPsel.[TFBS]TFBS0 as TFBS0
    where TFBS0.ID=STFBS0.TFBS_ID and not exists (
        select * from SNPsel.[TFBS]TFBS1 as TFBS1,SNPsel.[TFBS]STFBS1 as STFBS1
        where STFBS1.SNP_ID=STFBS0.SNP_ID and TFBS1.ID=STFBS1.TFBS_ID and TFBS0.Strand=TFBS1.Strand
        and TFBS0.Anfang=TFBS1.Anfang and TFBS0.Ende=TFBS1.Ende
    ) and exists (
        select * from SNPsel.[TFBS]TFBS1 as TFBS1,SNPsel.[TFBS]STFBS1 as STFBS1
        where STFBS1.SNP_ID=STFBS0.SNP_ID and TFBS1.ID=STFBS1.TFBS_ID and TFBS0.Strand=TFBS1.Strand
        and (TFBS0.Anfang<>TFBS1.Anfang or TFBS0.Ende<>TFBS1.Ende)
    )
) union (
    select 0,STFBS1.ID,STFBS1.SNP_ID from SNPsel.[TFBS]STFBS1 as STFBS1,SNPsel.[TFBS]TFBS1 as TFBS1
    where TFBS1.ID=STFBS1.TFBS_ID and not exists(
        select * from SNPsel.[TFBS]TFBS0 as TFBS0,SNPsel.[TFBS]STFBS0 as STFBS0
        where STFBS0.SNP_ID=STFBS1.SNP_ID and TFBS0.ID=STFBS0.TFBS_ID and TFBS0.Strand=TFBS1.Strand
        and TFBS0.Anfang=TFBS1.Anfang and TFBS0.Ende=TFBS1.Ende
    ) and exists (
        select * from SNPsel.[TFBS]TFBS0 as TFBS0,SNPsel.[TFBS]STFBS0 as STFBS0
        where STFBS0.SNP_ID=STFBS1.SNP_ID and TFBS0.ID=STFBS0.TFBS_ID and TFBS0.Strand=TFBS1.Strand
        and (TFBS0.Anfang<>TFBS1.Anfang or TFBS0.Ende<>TFBS1.Ende)
    )
);
```

4.7. Missing comparision

```
# Insert compared STFBS into STFBS database table, which are absolutely unique.
# Insert allelic STFBS from same prediction tool, generated and identified using both sequences (#0 and #1),
# which have no allelic counterpart. (This means TFBS, that have no overlapping TFBS on the allelic
# sequence affected by the same SNP).
# Replace [TFBS] by the name of used prediction tools (Mapper, Mulan or MatInspector).
insert into SNPsel.[TFBS]STFBSMissing(STFBS0_ID,STFBS1_ID,SNP_ID)
(select 0,STFBS1.ID,STFBS1.SNP_ID from SNPsel.[TFBS]STFBS1 as STFBS1,SNPsel.[TFBS]TFBS1 as TFBS1
where TFBS1.ID=STFBS1.TFBS_ID and not exists (
    select * from SNPsel.[TFBS]STFBS0 as STFBS0,SNPsel.[TFBS]TFBS0 as TFBS0
    where STFBS0.SNP_ID=STFBS1.SNP_ID and TFBS0.ID=STFBS0.TFBS_ID and TFBS0.Strand=TFBS1.Strand)
) union (
    select STFBS0.ID,0,STFBS0.SNP_ID from SNPsel.[TFBS]STFBS0 as STFBS0,SNPsel.[TFBS]TFBS0 as TFBS0
    where TFBS0.ID=STFBS0.TFBS_ID and not exists (
        select * from SNPsel.[TFBS]STFBS1 as STFBS1,SNPsel.[TFBS]TFBS1 as TFBS1
        where STFBS1.SNP_ID=STFBS0.SNP_ID and TFBS1.ID=STFBS1.TFBS_ID and TFBS0.Strand=TFBS1.Strand)
);
```

4.8. Comparing STFBS from two different prediction tools

```
# Insert STFBS into table, which are generated by two different prediction tools
# and result of same comparision method.
# Replace [TFBS] by the set of used prediction tools (MapperMulan, MapperMat or MulanMat)
# and [Modus] by the comparision method [Equal, Different or Missing]
# and [PT0], [PT1] by the name of used prediction tools (Mapper & Mulan, Mapper & MatInspector or
# Mulan & MatInspector).
insert into SNPsel.[TFBS]STFBS[Modus] ([PT0]ID, [PT1]ID)
select XSTFBS0.ID, XSTFBS1.ID
from SNPsel.[PT0]STFBS[Modus] as XSTFBS0, SNPsel.[PT1]STFBS[Modus] as XSTFBS1
where XSTFBS0.SNP_ID=XSTFBS1.SNP_ID;
```

4.9. Comparing STFBS from three different prediction tools

```
# Insert STFBS into table, which are generated by three different prediction tools
# and result of same comparision method.
# Replace [Modus] by the comparision method [Equal, Different or Missing].
insert into SNPsel.MapperMulanMatSTFBS[Modus](MapperID,MulanID,MatInspectorID)
select XSTFBS0.ID,XSTFBS1.ID,XSTFBS2.ID
from SNPsel.MapperSTFBS[Modus] as XSTFBS0,
     SNPsel.MulanSTFBS[Modus] as XSTFBS1,
     SNPsel.MatInspectorSTFBS[Modus] as XSTFBS2
where XSTFBS0.SNP_ID=XSTFBS1.SNP_ID and XSTFBS1.SNP_ID=XSTFBS2.SNP_ID;
```

5.1. Comparing allelic STFBS from same prediction tool using affected, unaffected

```
# Read compared STFBS from database table and take equal STFBS as unaffected STFBS.
# Replace [TFBS] by the name of used prediction tools (Mapper, Mulan or MatInspector)
# and [Modus] by the comparision method (Equal, Different or Missing).
select distinct T0.ID,SNP.Name as SName,SNP.Position,SNP.Strand,T0.Name,T0.Anfang,T0.Ende,'0' as Related,SNP.Alleles
from SNPsel.[TFBS]STFBS[Modus] as Equal,SNPsel.[TFBS]TFBS0 as T0,SNPsel.SNPS as SNP
where T0.ID>0 and T0.ID=(
    select ST0.TFBS_ID from SNPsel.[TFBS]STFBS0 as ST0 where ST0.ID=Equal.STFBS0_ID
) and SNP.ID=Equal.SNP_ID

union select T1.ID,SNP.Name as SName,SNP.Position,SNP.Strand,T1.Name,T1.Anfang,T1.Ende,'1' as Related,SNP.Alleles
from SNPsel.[TFBS]STFBS[Modus] as Equal,SNPsel.[TFBS]TFBS1 as T1,SNPsel.SNPS as SNP
where T1.ID>0 and T1.ID=(
    select ST1.TFBS_ID from SNPsel.[TFBS]STFBS1 as ST1 where ST1.ID=Equal.STFBS1_ID
) and SNP.ID=Equal.SNP_ID;


# Read compared STFBS from database table and take different or missing STFBS as affected STFBS.
# Replace [TFBS] by the name of used prediction tools (Mapper, Mulan or MatInspector)
# and [Modus] by the comparision method (Equal, Different or Missing).
select distinct T0.ID,SNP.Name as SName,SNP.Position,SNP.Strand,T0.Name,T0.Anfang,T0.Ende,'0' as Related,SNP.Alleles
from SNPsel.[TFBS]STFBSDifferent as Different,SNPsel.[TFBS]TFBS0 as T0,SNPsel.SNPS as SNP
where T0.ID>0 and T0.ID=(
    select ST0.TFBS_ID from SNPsel.[TFBS]STFBS0 as ST0 where ST0.ID=Different.STFBS0_ID
) and SNP.ID=Different.SNP_ID

union select T1.ID,SNP.Name as SName,SNP.Position,SNP.Strand,T1.Name,T1.Anfang,T1.Ende,'1' as Related,SNP.Alleles
from SNPsel.[TFBS]STFBSDifferent as Different,SNPsel.[TFBS]TFBS1 as T1,SNPsel.SNPS as SNP
where T1.ID>0 and T1.ID=(
    select ST1.TFBS_ID from SNPsel.[TFBS]STFBS1 as ST1 where ST1.ID=Different.STFBS1_ID
) and SNP.ID=Different.SNP_ID

union select T0.ID,SNP.Name as SName,SNP.Position,SNP.Strand,T0.Name,T0.Anfang,T0.Ende,'0' as Related,SNP.Alleles
from SNPsel.[TFBS]STFBSMissing as Missing,SNPsel.[TFBS]TFBS0 as T0,SNPsel.SNPS as SNP
where T0.ID>0 and T0.ID=(
    select ST0.TFBS_ID from SNPsel.[TFBS]STFBS0 as ST0 where ST0.ID=Missing.STFBS0_ID
) and SNP.ID=Missing.SNP_ID

union select T1.ID,SNP.Name as SName,SNP.Position,SNP.Strand,T1.Name,T1.Anfang,T1.Ende,'1' as Related,SNP.Alleles
from SNPsel.[TFBS]STFBSMissing,SNPsel.[TFBS]TFBS1 as T1,SNPsel.SNPS as SNP
where T1.ID>0 and T1.ID=(
    select ST1.TFBS_ID from SNPsel.[TFBS]STFBS1 as ST1 where ST1.ID=Missing.STFBS1_ID
) and SNP.ID=Missing.SNP_ID;
```

5.2. Comparing STFBS from two different prediction tools using affected, unaffected

```
# Read compared STFBS from database table and take equal STFBS as unaffected STFBS.
# Replace [TFBS0], [TFBS1] accordingly to [TFBS] by the name of used prediction tools
# (Example: [TFBS0]=Mapper, [TFBS1]=MatInspector and [TFBS]=MapperMat).
select distinct concat('[[TFBS0]]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T0.Name,T0.Anfang,T0.Ende,'0' as Related,SNP.Alleles
from SNPsel.[TFBS]STFBSEqual as Equal,SNPsel.[TFBS0]TFBS0 as T0,SNPsel.SNPS as SNP
where T0.ID>0 and T0.ID=(
    select ST0.TFBS_ID from SNPsel.[TFBS0]STFBS0 as ST0 where ST0.ID=(
        select STX.STFBS0_ID from SNPsel.[TFBS0]STFBSEqual as STX
        where STX.ID=Equal.[TFBS0]ID and SNP.ID=STX.SNP_ID
))
union select concat('[[TFBS0]]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T1.Name,T1.Anfang,T1.Ende,'1' as Related,SNP.Alleles
from SNPsel.[TFBS]STFBSEqual as Equal,SNPsel. [TFBS0]TFBS1 as T1,SNPsel.SNPS as SNP
where T1.ID>0 and T1.ID=(
    select ST1.TFBS_ID from SNPsel.[TFBS0]STFBS1 as ST1 where ST1.ID=(
            select STX.STFBS1_ID from SNPsel.[TFBS0]STFBSEqual as STX
            where STX.ID=Equal.[TFBS0]ID and SNP.ID=STX.SNP_ID
))
union select concat('[[TFBS1]]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T0.Name,T0.Anfang,T0.Ende,'0' as Related,SNP.Alleles
from SNPsel.[TFBS]STFBSEqual as Equal,SNPsel.[TFBS1]TFBS0 as T0,SNPsel.SNPS as SNP
where T0.ID>0 and T0.ID=(
    select ST0.TFBS_ID from SNPsel.[TFBS1]STFBS0 as ST0 where ST0.ID=(
        select STX.STFBS0_ID from SNPsel.[TFBS1]STFBSEquals as STX
        where STX.ID=Equal.[TFBS1]ID and SNP.ID=STX.SNP_ID
))
union select concat('[[TFBS1]]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T1.Name,T1.Anfang,T1.Ende,'1' as Related,SNP.Alleles
from SNPsel.[TFBS]STFBSEqual as Equal,SNPsel.[TFBS1]TFBS1 as T1,SNPsel.SNPS as SNP
where T1.ID>0 and T1.ID=(
    select ST1.TFBS_ID from SNPsel.[TFBS1]STFBS1 as ST1 where ST1.ID=(
        select STX.STFBS1_ID from SNPsel.[TFBS1]STFBSEqual as STX
        where STX.ID=Equal.[TFBS1]ID and SNP.ID=STX.SNP_ID
));


# Read compared STFBS from database table and take different or missing STFBS as affected STFBS.
# Replace [TFBS0], [TFBS1] accordingly to [TFBS] by the name of used prediction tools
# (Example: [TFBS0]=Mapper, [TFBS1]=MatInspector and [TFBS]=MapperMat).
select distinct concat('[[TFBS0]]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T0.Name,T0.Anfang,T0.Ende,'0' as Related,SNP.Alleles
from SNPsel.[TFBS]STFBSDifferent as Equal,SNPsel.[TFBS0]TFBS0 as T0,SNPsel.SNPS as SNP
where T0.ID>0 and T0.ID=(
    select ST0.TFBS_ID from SNPsel.[TFBS0]STFBS0 as ST0 where ST0.ID=(
        select STX.STFBS0_ID from SNPsel.[TFBS0]STFBSDifferent as STX
        where STX.ID=Equal.[TFBS0]ID and SNP.ID=STX.SNP_ID
))
union select concat('[[TFBS0]]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T1.Name,T1.Anfang,T1.Ende,'1' as Related,SNP.Alleles
from SNPsel.[TFBS]STFBSDifferent as Equal,SNPsel.[TFBS0]TFBS1 as T1,SNPsel.SNPS as SNP
where T1.ID>0 and T1.ID=(
    select ST1.TFBS_ID from SNPsel.[TFBS0]STFBS1 as ST1 where ST1.ID=(
        select STX.STFBS1_ID from SNPsel.[TFBS0]STFBSDifferent as STX
        where STX.ID=Equal.[TFBS0]ID and SNP.ID=STX.SNP_ID
))
```

```
union select concat('[[TFBS1]]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T0.Name,T0.Anfang,T0.Ende,'0' as Related,SNP.Alleles
from SNPsel.[TFBS]STFBSDifferent as Equal,SNPsel.[TFBS1]TFBS0 as T0,SNPsel.SNPS as SNP
where T0.ID>0 and T0.ID=(
    select ST0.TFBS_ID from SNPsel.[TFBS1]STFBS0 as ST0 where ST0.ID=(
        select STX.STFBS0_ID from SNPsel.[TFBS1]STFBSDifferent as STX
        where STX.ID=Equal.[TFBS1]ID and SNP.ID=STX.SNP_ID
))
union select concat('[[TFBS1]]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T1.Name,T1.Anfang,T1.Ende,'1' as Related,SNP.Alleles
from SNPsel.[TFBS]STFBSDifferent as Equal,SNPsel.[TFBS1]TFBS1 as T1,SNPsel.SNPS as SNP
where T1.ID>0 and T1.ID=(
    select ST1.TFBS_ID from SNPsel.[TFBS1]STFBS1 as ST1 where ST1.ID=(
        select STX.STFBS1_ID from SNPsel.[TFBS1]STFBSDifferent as STX
        where STX.ID=Equal.[TFBS1]ID and SNP.ID=STX.SNP_ID
))

union select concat('[[TFBS0]]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T0.Name,T0.Anfang,T0.Ende,'0' as Related,SNP.Alleles
from SNPsel.[TFBS]STFBSMissing as Equal,SNPsel.[TFBS0]TFBS0 as T0,SNPsel.SNPS as SNP
where T0.ID>0 and T0.ID=(
    select ST0.TFBS_ID from SNPsel.[TFBS0]STFBS0 as ST0 where ST0.ID=(
        select STX.STFBS0_ID from SNPsel.[TFBS0]STFBSMissing as STX
        where STX.ID=Equal.[TFBS0]ID and SNP.ID=STX.SNP_ID
))
union select concat('[[TFBS0]]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T1.Name,T1.Anfang,T1.Ende,'1' as Related,SNP.Alleles
from SNPsel.[TFBS]STFBSMissing as Equal,SNPsel.[TFBS0]TFBS1 as T1,SNPsel.SNPS as SNP
where T1.ID>0 and T1.ID=(
    select ST1.TFBS_ID from SNPsel.[TFBS0]STFBS1 as ST1 where ST1.ID=(
        select STX.STFBS1_ID from SNPsel. [TFBS0]STFBSMissing as STX
        where STX.ID=Equal.[TFBS0]ID and SNP.ID=STX.SNP_ID
))
union select concat('[[TFBS1]]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T0.Name,T0.Anfang,T0.Ende,'0' as Related,SNP.Alleles
from SNPsel.[TFBS]STFBSMissing as Equal,SNPsel.[TFBS1]TFBS0 as T0,SNPsel.SNPS as SNP
where T0.ID>0 and T0.ID=(
    select ST0.TFBS_ID from SNPsel.[TFBS1]STFBS0 as ST0 where ST0.ID=(
        select STX.STFBS0_ID from SNPsel.[TFBS1]STFBSMissing as STX
        where STX.ID=Equal.[TFBS1]ID and SNP.ID=STX.SNP_ID
))
union select concat('[[TFBS1]]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T1.Name,T1.Anfang,T1.Ende,'1' as Related,SNP.Alleles
from SNPsel.[TFBS]STFBSMissing as Equal,SNPsel.[TFBS1]TFBS1 as T1,SNPsel.SNPS as SNP
where T1.ID>0 and T1.ID=(
    select ST1.TFBS_ID from SNPsel.[TFBS1]STFBS1 as ST1 where ST1.ID=(
        select STX.STFBS1_ID from SNPsel.[TFBS1]STFBSMissing as STX
        where STX.ID=Equal.[TFBS1]ID and SNP.ID=STX.SNP_ID
));
```

5.3. Comparing STFBS from three different prediction tools using affected,unaffected

```
# Read compared STFBS from database table and take equal STFBS as unaffected STFBS.
select distinct concat('[Mapper]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T0.Name,T0.Anfang,T0.Ende,'0' as Related,SNP.Alleles
from SNPsel.MapperMulanMatSTFBSEqual as Equal,SNPsel.MapperTFBS0 as T0,SNPsel.SNPS as SNP
where T0.ID>0 and T0.ID=(
    select ST0.TFBS_ID from SNPsel.MapperSTFBS0 as ST0 where ST0.ID=(
        select STX.STFBS0_ID from SNPsel.MapperSTFBSEqual as STX
        where STX.ID=Equal.MapperID and SNP.ID=STX.SNP_ID
))
union select distinct concat('[Mapper]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T1.Name,T1.Anfang,T1.Ende,'1' as Related,SNP.Alleles
from SNPsel. MapperMulanMatSTFBSEqual as Equal,SNPsel.MapperTFBS1 as T1,SNPsel.SNPS as SNP
where T1.ID>0 and T1.ID=(
    select ST1.TFBS_ID from SNPsel.MapperSTFBS1 as ST1 where ST1.ID=(
        select STX.STFBS1_ID from SNPsel.MapperSTFBSEqual as STX
        where STX.ID=Equal.MapperID and SNP.ID=STX.SNP_ID
))

union select concat('[Mulan]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T0.Name,T0.Anfang,T0.Ende,'0' as Related,SNP.Alleles
from SNPsel.MapperMulanMatSTFBSEqual as Equal,SNPsel.MulanTFBS0 as T0,SNPsel.SNPS as SNP
where T0.ID>0 and T0.ID=(
    select ST0.TFBS_ID from SNPsel.MulanSTFBS0 as ST0 where ST0.ID=(
        select STX.STFBS0_ID from SNPsel.MulanSTFBSEqual as STX
        where STX.ID=Equal.MulanID and SNP.ID=STX.SNP_ID
))
union select concat('[Mulan]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T1.Name,T1.Anfang,T1.Ende,'1' as Related,SNP.Alleles
from SNPsel.MapperMulanMatSTFBSEqual as Equal,SNPsel.MulanTFBS1 as T1,SNPsel.SNPS as SNP
where T1.ID>0 and T1.ID=(
    select ST1.TFBS_ID from SNPsel.MulanSTFBS1 as ST1 where ST1.ID=(
        select STX.STFBS1_ID from SNPsel.MulanSTFBSEqual as STX
        where STX.ID=Equal.MulanID and SNP.ID=STX.SNP_ID
))

union select concat('[MatInspector]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T0.Name,T0.Anfang,T0.Ende,'0' as Related,SNP.Alleles
from SNPsel.MapperMulanMatSTFBSEqual as Equal,SNPsel.MatInspectorTFBS0 as T0,SNPsel.SNPS as SNP
where T0.ID>0 and T0.ID=(
    select ST0.TFBS_ID from SNPsel.MatInspectorSTFBS0 as ST0 where ST0.ID=(
        select STX.STFBS0_ID from SNPsel.MatInspectorSTFBSEqual as STX
        where STX.ID=Equal.MatInspectorID and SNP.ID=STX.SNP_ID
))
union select concat('[MatInspector]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T1.Name,T1.Anfang,T1.Ende,'1' as Related,SNP.Alleles
from SNPsel.MapperMulanMatSTFBSEqual as Equal,SNPsel.MatInspectorTFBS1 as T1,SNPsel.SNPS as SNP
where T1.ID>0 and T1.ID=(
    select ST1.TFBS_ID from SNPsel.MatInspectorSTFBS1 as ST1 where ST1.ID=(
        select STX.STFBS1_ID from SNPsel.MatInspectorSTFBSEqual as STX
        where STX.ID=Equal.MatInspectorID and SNP.ID=STX.SNP_ID
));
```

```
# Read compared STFBS from database table and take different or missing STFBS as affected STFBS.
select distinct concat('[Mapper]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T0.Name,T0.Anfang,T0.Ende,'0' as Related,SNP.Alleles
from SNPsel.MapperMulanMatSTFBSDifferent as Equal,SNPsel.MapperTFBS0 as T0,SNPsel.SNPS as SNP
where T0.ID>0 and T0.ID=(
    select ST0.TFBS_ID from SNPsel.MapperSTFBS0 as ST0 where ST0.ID=(
        select STX.STFBS0_ID from SNPsel.MapperSTFBSDifferent as STX
        where STX.ID=Equal.MapperID and SNP.ID=STX.SNP_ID
))
union select concat('[Mapper]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T1.Name,T1.Anfang,T1.Ende,'1' as Related,SNP.Alleles
from SNPsel.MapperMulanMatSTFBSDifferent as Equal,SNPsel.MapperTFBS1 as T1,SNPsel.SNPS as SNP
where T1.ID>0 and T1.ID=(
    select ST1.TFBS_ID from SNPsel.MapperSTFBS1 as ST1 where ST1.ID=(
        select STX.STFBS1_ID from SNPsel.MapperSTFBSDifferent as STX
        where STX.ID=Equal.MapperID and SNP.ID=STX.SNP_ID
))

union select distinct concat('[Mulan]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T0.Name,T0.Anfang,T0.Ende,'0' as Related,SNP.Alleles
from SNPsel.MapperMulanMatSTFBSDifferent as Equal,SNPsel.MulanTFBS0 as T0,SNPsel.SNPS as SNP
where T0.ID>0 and T0.ID=(
    select ST0.TFBS_ID from SNPsel.MulanSTFBS0 as ST0 where ST0.ID=(
        select STX.STFBS0_ID from SNPsel.MulanSTFBSDifferent as STX
        where STX.ID=Equal.MulanID and SNP.ID=STX.SNP_ID
))
union select concat('[Mulan]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T1.Name,T1.Anfang,T1.Ende,'1' as Related,SNP.Alleles
from SNPsel.MapperMulanMatSTFBSDifferent as Equal,SNPsel.MulanTFBS1 as T1,SNPsel.SNPS as SNP
where T1.ID>0 and T1.ID=(
    select ST1.TFBS_ID from SNPsel.MulanSTFBS1 as ST1 where ST1.ID=(
        select STX.STFBS1_ID from SNPsel.MulanSTFBSDifferent as STX
        where STX.ID=Equal.MulanID and SNP.ID=STX.SNP_ID
))

union select distinct concat('[MatInspector]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T0.Name,T0.Anfang,T0.Ende,'0' as Related,SNP.Alleles
from SNPsel.MapperMulanMatSTFBSDifferent as Equal,SNPsel.MatInspectorTFBS0 as T0,SNPsel.SNPS as SNP
where T0.ID>0 and T0.ID=(
    select ST0.TFBS_ID from SNPsel.MatInspectorSTFBS0 as ST0 where ST0.ID=(
        select STX.STFBS0_ID from SNPsel.MatInspectorSTFBSDifferent as STX
        where STX.ID=Equal.MatInspectorID and SNP.ID=STX.SNP_ID
))
union select concat('[MatInspector]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T1.Name,T1.Anfang,T1.Ende,'1' as Related,SNP.Alleles
from SNPsel.MapperMulanMatSTFBSDifferent as Equal,SNPsel.MatInspectorTFBS1 as T1,SNPsel.SNPS as SNP
where T1.ID>0 and T1.ID=(
    select ST1.TFBS_ID from SNPsel.MatInspectorSTFBS1 as ST1 where ST1.ID=(
        select STX.STFBS1_ID from SNPsel.MatInspectorSTFBSDifferent as STX
        where STX.ID=Equal.MatInspectorID and SNP.ID=STX.SNP_ID
))
```

```
union select concat('[Mapper]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T0.Name,T0.Anfang,T0.Ende,'0' as Related,SNP.Alleles
from SNPsel.MapperMulanMatSTFBSMissing as Equal,SNPsel.MapperTFBS0 as T0,SNPsel.SNPS as SNP
where T0.ID>0 and T0.ID=(
    select ST0.TFBS_ID from SNPsel.MapperSTFBS0 as ST0 where ST0.ID=(
        select STX.STFBS0_ID from SNPsel.MapperSTFBSMissing as STX
        where STX.ID=Equal.MapperID and SNP.ID=STX.SNP_ID
))
union select concat('[Mapper]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T1.Name,T1.Anfang,T1.Ende,'1' as Related,SNP.Alleles
from SNPsel.MapperMulanMatSTFBSMissing as Equal,SNPsel.MapperTFBS1 as T1,SNPsel.SNPS as SNP
where T1.ID>0 and T1.ID=(
    select ST1.TFBS_ID from SNPsel.MapperSTFBS1 as ST1 where ST1.ID=(
        select STX.STFBS1_ID from SNPsel.MapperSTFBSMissing as STX
        where STX.ID=Equal.MapperID and SNP.ID=STX.SNP_ID
))

union select concat('[Mulan]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T0.Name,T0.Anfang,T0.Ende,'0' as Related,SNP.Alleles
from SNPsel.MapperMulanMatSTFBSMissing as Equal,SNPsel.MulanTFBS0 as T0,SNPsel.SNPS as SNP
where T0.ID>0 and T0.ID=(
    select ST0.TFBS_ID from SNPsel.MulanSTFBS0 as ST0 where ST0.ID=(
        select STX.STFBS0_ID from SNPsel.MulanSTFBSMissing as STX
        where STX.ID=Equal.MulanID and SNP.ID=STX.SNP_ID
))
union select concat('[Mulan]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T1.Name,T1.Anfang,T1.Ende,'1' as Related,SNP.Alleles
from SNPsel.MapperMulanMatSTFBSMissing as Equal,SNPsel.MulanTFBS1 as T1,SNPsel.SNPS as SNP
where T1.ID>0 and T1.ID=(
    select ST1.TFBS_ID from SNPsel.MulanSTFBS1 as ST1 where ST1.ID=(
        select STX.STFBS1_ID from SNPsel.MulanSTFBSMissing as STX
        where STX.ID=Equal.MulanID and SNP.ID=STX.SNP_ID
))

union select concat('[MatInspector]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T0.Name,T0.Anfang,T0.Ende,'0' as Related,SNP.Alleles
from SNPsel.MapperMulanMatSTFBSMissing as Equal,SNPsel.MatInspectorTFBS0 as T0,SNPsel.SNPS as SNP
where T0.ID>0 and T0.ID=(
    select ST0.TFBS_ID from SNPsel.MatInspectorSTFBS0 as ST0 where ST0.ID=(
        select STX.STFBS0_ID from SNPsel.MatInspectorSTFBSMissing as STX
        where STX.ID=Equal.MatInspectorID and SNP.ID=STX.SNP_ID
))
union select concat('[MatInspector]') as X,SNP.Name as SName,SNP.Position,SNP.Strand,
    T1.Name,T1.Anfang,T1.Ende,'1' as Related,SNP.Alleles
from SNPsel.MapperMulanMatSTFBSMissing as Equal,SNPsel.MatInspectorTFBS1 as T1,SNPsel.SNPS as SNP
where T1.ID>0 and T1.ID=(
    select ST1.TFBS_ID from SNPsel.MatInspectorSTFBS1 as ST1 where ST1.ID=(
        select STX.STFBS1_ID from SNPsel.MatInspectorSTFBSMissing as STX
        where STX.ID=Equal.MatInspectorID and SNP.ID=STX.SNP_ID
));
```