

Use the VEP online to analyse your variants through a simple point-and-click interface.

The web interface allows you to access the key features of the VEP without using the command line. Interactively filter your results to find the data you want. Download your results in multiple data formats, easily share your results with others, and integrate your variation data with the powerful Ensembl web browser.

If you use the VEP in your work, please cite **McLaren et. al.** ([doi:10.1093/bioinformatics/btq330](https://doi.org/10.1093/bioinformatics/btq330))

Any questions? Send an email to the Ensembl developer's mailing list, dev@ensembl.org or contact the Ensembl Helpdesk at helpdesk@ensembl.org.



Documentation contents

 [Download documentation in PDF format](#)

[Input form](#)

- [Data input](#)
- [Identifiers and co-located variants](#)
- [Extra options](#)
- [Filtering options](#)
- [Plugins](#)
- [Jobs](#)

[Results](#)

- [Results summary](#)
- [Results preview table](#)
- [Navigating results](#)
- [Filtering results](#)
- [Downloading results](#)

[Data formats](#)

- [Input](#)
- [Output](#)

[FAQ](#)

- [General questions](#)
- [Web VEP questions](#)
- [VEP script questions](#)

When you reach the VEP web interface, you will be presented with a form to enter your data and alter various options.

Data input

1. First select the correct species for your data. Ensembl hosts many vertebrate genomes; genomes for plants, protists and fungi can be found at [Ensembl Genomes](#).
2. You can optionally choose a name for the data you upload - this can make it easier for you to identify jobs and files that you have uploaded to the VEP at a later point.
3. You have three options for uploading your data:
 - **File upload** - click the "Choose file" button and locate the file on your system
 - **Paste file** - simply copy and paste the contents of your file into the large text box
 - **File URL** - point the VEP to a file hosted on a publically accessible address. This can be either a **http://** or **ftp://** address.

Once you have uploaded some data, you can select it as the input for future jobs by choosing the data from the drop down menu.



The format of your data is automatically detected; see the examples or the [input format](#) documentation.

4. For pasted data you can get an instant preview of the results of your first variant by clicking the button that appears when you paste your data. This quickly shows you the consequence type, the IDs of any overlapping variants, genes, transcripts and regulatory features, as well as SIFT and PolyPhen predictions. To see the full results set submit your job as normal.
5. For some species you can select which transcript database to use. The default is to use Ensembl transcripts, which offer the most rich annotation through the VEP.

GENCODE Basic is a subset of the GENCODE gene set, and is intended to provide a simplified, high-quality subset of the GENCODE transcript annotations that will be useful to the majority of users. GENCODE Basic includes all genes in the GENCODE gene set, with a representative subset of the transcripts (splice variants).

You can also select to use RefSeq transcripts from the [otherfeatures database](#); note though that these transcripts are simply aligned to the reference genome and the database is missing much of the annotation found when using the main Ensembl database (e.g. protein domains, CCDS identifiers). When using RefSeq transcripts you may choose to include aligned EST and CCDS transcripts also.

Input

Species:  Cow (Bos taurus) 

Assembly: UMD3.1

Name for this data (optional):

Either paste data:

rs480447159
rs462821358
rs465117936

Examples: [Ensembl default](#), [VCF](#), [Variant identifiers](#), [HGVS notations](#), [Pileup](#)

Instant results for first variant >

Instant results for rs480447159

Most severe consequence: missense_variant

Colocated variants: [rs480447159](#)

Gene/Feature/Type	Consequence	Details
RAB4A : ENSBTAT00000049768 Type: protein_coding	missense_variant	Amino acids: F/C SIFT: deleterious

Note: the above is a preview of results using the Bos taurus Ensembl transcript database and does not include all data fields present in the full results set

Or upload file: No file chosen

Or provide file URL:

Identifiers and co-located variants

The VEP can provide additional identifiers for genes, transcripts, proteins and variants. It can also search the Ensembl database for known variants that are co-located with variants from your input data.

- **Gene symbol** - add the gene symbol for the gene to the output. This will typically be, for example, the [HGNC](#) identifier for genes in human. Equivalent to `--symbol` in the VEP script.
- **CCDS** - add the [Consensus CDS](#) transcript identifier where available. Equivalent to `--ccds`
- **Protein** - add the Ensembl protein identifier (ENSP). Equivalent to `--protein`
- **Uniprot** - add identifiers for translated protein products from three [UniProt](#)-related databases (SWISSPROT, TrEMBL and UniParc). Equivalent to `--uniprot`
- **HGVS** - generate [HGVS](#) identifiers for your input variants relative to the transcript coding sequence (HGVS) and the protein sequence (HGVS). Equivalent to `--hgvs`
- **Find co-located known variants** - report known variants from the Ensembl Variation database that overlap with your input. A list of variant sources imported can be viewed [here](#). Note that this feature is only available for species with an Ensembl Variation database. Equivalent to `--check_existing`.

The VEP will also allow you to compare the alleles of your input variant to that of the existing variant by selecting "Yes and compare alleles" from the drop-down menu. By selecting this, the VEP will only report the existing variant ID if none of the alleles in your input variant are novel.

For example, if your input variant has alleles A/G, and the existing variant has alleles A/T, then the existing variant will not be reported. If instead your input variant has alleles A/T, then the existing variant will be reported. This is equivalent to using `--check_alleles` in the VEP script.

For known variants the VEP can also provide PubMed IDs of publications citing the variant (equivalent to `--pubmed`).

- The VEP can also report minor allele frequency (MAF) data for existing variants from two major genotyping projects, the [1000 Genomes Project](#) and the [NHLBI-ESP](#); this only applies when you have selected human as your species.
 - **1000 Genomes global** - the combined phase 1 population (i.e. all individuals from all populations). Equivalent to `--gmaf`
 - **1000 Genomes continental** - the four continent-level populations - AFR (African), AMR (American), ASN (Asian) and EUR (European). Equivalent to `--maf_1kg`
 - **ESP** - AA (African American) and EA (European American) populations. Equivalent to `--maf_esp`

Identifiers and frequency data Additional identifiers for genes, transcripts and variants; frequency data

Identifiers

Gene symbol:	<input checked="" type="checkbox"/>
CCDS:	<input type="checkbox"/>
Protein:	<input type="checkbox"/>
HGVS:	<input type="checkbox"/>
Find co-located known variants:	Yes

Frequency data for co-located variants

1000 Genomes global minor allele frequency:	<input checked="" type="checkbox"/>
1000 Genomes continental minor allele frequencies:	<input type="checkbox"/>
ESP minor allele frequencies:	<input type="checkbox"/>

Extra options

- **Transcript biotype** - add the [transcript biotype](#) to the output. Equivalent to `--biotype` in the VEP script.
- **Protein domains** - report [protein domains](#) from [Pfam](#), [Prosite](#) and [InterPro](#) that overlap input variants. Equivalent to `--domains`

- **Exon and intron numbers** - report the exon or intron number that a variant falls in as NUMBER / TOTAL, i.e. exon 2/5 means the variant falls in the 2nd of 5 exons in the transcript. Equivalent to [--numbers](#)
- **Transcript support level** - report the [transcript support level](#) of the overlapped transcript. Equivalent to [--tsl](#)
- **Identify canonical transcripts** - adds a flag to the output indicating if the reported transcript is the [canonical transcript](#) for the gene. Equivalent to [--canonical](#)
- **SIFT predictions** - [SIFT](#) predicts whether an amino acid substitution affects protein function based on sequence homology and the physical properties of amino acids. Only available in popular species. For both SIFT and PolyPhen the VEP can report either a score between 0 and 1, a prediction in words, or both. Equivalent to [--sift](#)
- **PolyPhen predictions** - [PolyPhen](#) is a tool which predicts possible impact of an amino acid substitution on the structure and function of a human protein using straightforward physical and comparative considerations. Equivalent to [--polyphen](#)
- **Get regulatory region consequences** - in addition to predicting consequences with overlapping transcripts, the VEP can find overlaps with known regulatory regions as determined in the [Ensembl Regulatory build](#).

Using this option, the VEP will also report if a variant falls in a transcription factor binding motif, and give a score that reflects whether the altered motif sequence is more or less similar to the consensus.

Get regulatory consequences is equivalent to [--regulatory](#)

The screenshot shows the 'Extra options' panel in the VEP interface. It includes a title bar with a dropdown arrow and the text 'e.g. SIFT, PolyPhen and regulatory data'. Below the title bar are several rows of options:

- Transcript biotype:** A checkbox that is currently checked.
- Protein domains:** An unchecked checkbox.
- Exon and intron numbers:** An unchecked checkbox.
- Identify canonical transcripts:** An unchecked checkbox.
- SIFT predictions:** A dropdown menu currently set to 'Prediction and score'.
- PolyPhen predictions:** A dropdown menu currently set to 'Prediction and score'.
- Get regulatory region consequences:** A dropdown menu currently set to 'Yes'.

Filtering options

The VEP allows you to pre-filter your results e.g. by MAF or consequence type. Note that it is also possible to perform equivalent operations on the results page for the VEP, so if you aren't sure, don't use any of these options!

- **By frequency** - filter variants by minor allele frequency (MAF). Two options are provided:
 - **Exclude common variants** - filter out variants that are co-located with an existing variant that has a frequency greater than 0.01 (1%) in the 1000 Genomes global population. Equivalent to [--filter common](#) in the VEP script.
 - **Advanced filtering** - enabling this option allows you to specify a population and frequency to compare to, as well whether matching variants should be included or excluded from the results.
- **Return results for variants in coding regions only** - exclude variants that don't fall in a coding region of a transcript. Equivalent to [--coding only](#)
- **Restrict results** - for many variants the VEP will report multiple consequence types - typically this is because the variant overlaps more than one transcript. For each of these options the VEP uses consequence ranks that are subjectively determined by Ensembl. [This table](#) gives all of the consequence types predicted by Ensembl, ordered by rank. Note that enabling one of these options not only loses potentially relevant data, but in some cases may be scientifically misleading. Options:
 - **Show one selected consequence** - pick one consequence type across all those predicted for the variant; the output will include transcript- or feature-specific information. Consequences are chosen by the canonical, biotype status and length of the transcript, along with the ranking of the consequence type according to [this table](#). This is the best method to use if you are interested only in one consequence per variant. Equivalent to [--pick](#)
 - **Show one selected consequence per gene** - pick one consequence type for each gene using the same criteria as above. Note that if a variant overlaps more than one gene, output for each gene will be reported. Equivalent to [--per gene](#)
 - **Show only list of consequences per variant** - give a comma-separated list of all observed consequence types for each variant. No transcript-specific or gene-specific output will be given. Equivalent to [--summary](#)

- **Show most severe per variant** - only the most severe of all observed consequence types is reported for each variant. No transcript-specific or gene-specific output will be given. Equivalent to `--most_severe`

Filtering options Pre-filter results by frequency or consequence type

Filters

By frequency: ☐ No filtering ☐ Exclude common variants ☒ Advanced filtering

Filter:

Return results for variants in coding regions only: ☐

Restrict results:

NB: Restricting results may exclude biologically important data!

Plugins

The VEP can be extended beyond its core functionality using a system of [plugins](#). These are additional chunks of code that get executed during the VEP job, and can be used to retrieve data, run algorithms and modify the VEP's output in other ways.

A number of these plugins are available for use via the VEP's web interface; ^(p) is used to indicate these on the input form. Some plugins may require you to select additional options; these options are revealed on the form and highlighted when you enable the plugin.

Hover the mouse over a plugin's form label to see a summary of its functionality.

Fields added by plugins appear after the VEP's "core" fields in the results page, and can be filtered on in the same way. Note that enabling some plugins can create more than one output field; use the column hover-overs to see column descriptions.

miRNA structure^(p): ☐

Upstream/Downstream distance^(p): ☐ Disabled ☒ Enabled

Distance (bp):

Jobs

Once you have clicked "Run", your input will be checked and submitted to the VEP as a job. All jobs associated with your session or account are shown in the "Recent Tickets" table. You may submit multiple jobs simultaneously.

The "Jobs" column of the table shows the current status of the job.

- **Queued** - your job is waiting to be submitted to the system
- **Running** - your job is currently running
- **Done** - your job is finished - click the [View results] link to be taken to the results page
- **Failed** - there is a problem with your job - click the magnifying glass icon 🔍 to see more details

You may delete a job by clicking the trash can icon 🗑️. If you are logged in to Ensembl, you can save the job by clicking the save icon 💾.

You may also resubmit a job (for example, to re-run with the same data but change some parameters) by clicking the edit icon ✎️.

You can see a summary of the options that you selected for your VEP job by clicking on the magnifying glass icon 🔍.

Show All entries		Show/hide columns (1 hidden)		Filter	
Analysis	Jobs			Submitted at	
Variant Effect Predictor	VEP analysis of pasted data in Bos_taurus	Done	View results 🔍 ✎ 🗑	13/07/2015, 09:44	📄 ✎ 🗑
Variant Effect Predictor	VEP analysis of pasted data in Ovis_aries	Done	View results	08/07/2015, 13:19	📄 ✎ 🗑
Variant Effect Predictor	VEP analysis of pasted data in Ovis_aries	Failed		07/07/2015, 16:51	📄 ✎ 🗑

The VEP presents a [summary](#) and a detailed [results preview](#) on its results page.

Summary

The summary panel on the VEP results page gives a brief overview of the VEP job, along with some basic statistics about the results.

Statistics

Various statistics are listed in a table, including:

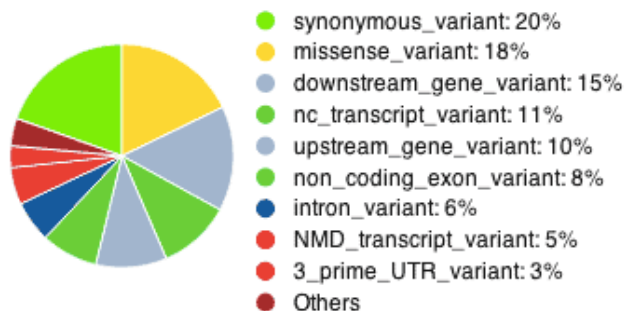
- Variants processed - any variants not parsed by the VEP are not included in this count
- Variants remaining after [filtering](#)
- Novel / known variants - the number and percentage of novel variants vs existing variants in the input (see [input page documentation](#))
- Number of overlapped genes, transcripts and regulatory features

Category	Count
Variants processed	498955
Variants remaining after filtering	498955
Novel / known variants	-
Overlapped genes	825
Overlapped transcripts	2888
Overlapped regulatory features	7309

Pie charts

Pie charts are shown detailing the proportion of consequence types called across all variants in the results. The colour scheme of the pie chart matches the colours used to draw variants on the Ensembl region in detail view.

Consequences (all)



Results preview table

The results table shows one row per transcript and variant. By default all of the columns are shown; to temporarily hide columns, click the blue "Show/hide columns" button and select or deselect the columns you wish to view. The columns you select will be recalled when viewing other jobs.

Hover over a column title to see a description. See the [VEP output format documentation](#) for more details on each of the results columns.

The table can be sorted by any column - click the column header to toggle sorting behaviour.

To download what you see in the table, hover over the spreadsheet icon in the top right corner of the table.

Several columns have special features for the data they contain:

- **Location** - click the link to navigate to the region in detail view for the region surrounding this variant

- **Gene, Feature and Existing Variation** - click the link to bring up a summary view of the gene, transcript, regulatory feature or variation, from which you can navigate to the main Ensembl page for it
- **Consequence** - hover over the consequence name to see the [Sequence Ontology](#) definition. See the [Ensembl Variation documentation](#) for a full list of consequence types used by the VEP and their definitions
- **SIFT and PolyPhen** - predictions and scores are coloured according to the nature of the prediction, with red indicating deleterious or damaging

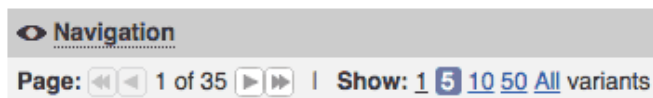
Uploaded variation	Location	Feature	Feature type	Consequence	CDS position	Protein position	Amino acids	Codons	SIFT	GMAF
rs116383664	1:11115461	ENSB000000528923	RegulatoryFeature	regulatory_region_variant	-	-	-	-	-	T:0.0137
rs116383664	1:11115461	ENST00000379317	Transcript	upstream_gene_variant	-	-	-	-	-	T:0.0137
rs116383664	1:11115461	ENST00000486379	Transcript	upstream_gene_variant	-	-	-	-	-	T:0.0137
rs116383664	1:11115461	ENST00000379289	Transcript	missense_variant	247	83	R/W	Cgg/Tgg	tolerated(0.06)	T:0.0137
rs116383664	1:11115461	ENST00000460996	Transcript	upstream_gene_variant	-	-	-	-	-	T:0.0137
rs116383664	1:11115461	ENST00000514695	Transcript	upstream_gene_variant	-	-	-	-	-	T:0.0137
rs116383664	1:11115461	ENST00000379290	Transcript	missense_variant	247	83	R/W	Cgg/Tgg	tolerated(0.06)	T:0.0137
rs116383664	1:11115461	ENST00000379288	Transcript	missense_variant	28	10	R/W	Cgg/Tgg	deleterious(0.03)	T:0.0137

Navigating results

The navigation panel can be used to scroll through pages of results.

By default, the results for five variants are shown. Note that since a variant can overlap multiple transcripts, the table will often show **more than** five rows. To change the number shown, click the appropriate link. Be warned that if your input file is large, it is inadvisable to show all results unless you are sure you have applied sufficient filters - your browser may become unresponsive if it tries to display many thousands of rows in the table.

To navigate between pages of results, use the four arrow icons. Note that when any filters are enabled, it is not possible to navigate to the last page of results as the total number of results cannot be calculated.



Filtering results

You can apply any combination of filters to your results in order to identify interesting data. This is equivalent to using the [VEP filtering script](#) on the command line.

To add a filter, simply select the column you wish to filter on, select an "operator", and input a value for the filter to compare to.

To edit a filter, click the pencil icon . To remove a filter, click the cross icon .

When you have added more than one filter, you are given the option to match any or all of the rules shown; click the "Update" button once you have made your selection.

Certain columns when selected have special features:

- **Location** - for this column you may enter genomic coordinates in the format "chromosome:start-end". It is also possible to enter just a chromosome, e.g. enter "12" to show only variants on chromosome 12.

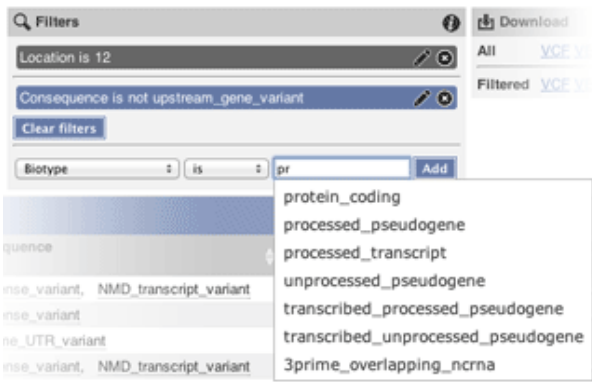
Adding multiple location filters allows you to select multiple regions - location filters are not affected by whether you select "Match all" or "Match any" (see above).

Users should note that enabling at least one location filter will greatly speed up the return of results (this is because [tabix](#) is used behind the scenes).

Location filters are not affected by the operator selected.

- **Allele, Feature type, Consequence, SIFT, PolyPhen and Biotype** - for these columns, autocomplete will help you fill in the value when you start typing
- **SIFT and PolyPhen** - these columns can contain both text (e.g. a SIFT prediction) and a number (e.g. a frequency value). The VEP allows you to filter on either part of this.

For example, you may enter "is" and "deleterious" for SIFT to return deleterious predictions, or "<" and "0.1" to find results with a SIFT score less than 0.1.

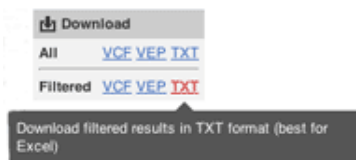


Downloading results

The VEP allows you to download either your full or filtered results set in a choice of data formats.

- **VCF** - [VCF](#) is a portable format for variant data. Consequence data is encoded as a series of delimited strings under the "CSQ" key in the VCF INFO field.
- **VEP** - The default [VEP output format](#) gives one row per variant and transcript overlap.
- **TEXT** - Text format is a tab-delimited format, equivalent to what can be seen in the results table. Note that the columns you select to be visible in the table do not affect the downloaded file - all columns are outputted. This format is best if you intend to import the results into a spreadsheet program such as Microsoft Excel.

You can also send the genes or known variants in your current preview to BioMart. This allows you to easily retrieve any of BioMart's rich data associated with these genes (other database references, GO terms, orthologues/paralogues) and variants (phenotype annotations, synonyms, citations).



Input

Both the web and script version of VEP can use the same input formats. Formats can be auto-detected by the VEP script, but must be manually selected when using the web interface. VEP can use [VCF](#), [variant identifiers](#) and [HGVS notations](#) in addition to the [default](#) format

Default

The default format is a simple **whitespace-separated** format (columns may be separated by space or tab characters), containing five required columns plus an optional identifier column:

1. **chromosome** - just the name or number, with no 'chr' prefix
2. **start**
3. **end**
4. **allele** - pair of alleles separated by a '/', with the reference allele first
5. **strand** - defined as + (forward) or - (reverse).
6. **identifier** - this identifier will be used in VEP's output. If not provided, VEP will construct an identifier from the given coordinates and alleles.

```
1 881907 881906 -/C +
5 140532 140532 T/C +
12 1017956 1017956 T/A +
2 946507 946507 G/C +
14 19584687 19584687 C/T -
19 66520 66520 G/A + var1
8 150029 150029 A/T + var2
```

An insertion (of any size) is indicated by start coordinate = end coordinate + 1. For example, an insertion of 'C' between nucleotides 12600 and 12601 on the forward strand of chromosome 8 is indicated as follows:

```
8 12601 12600 -/C +
```

A deletion is indicated by the exact nucleotide coordinates. For example, a three base pair deletion of nucleotides 12600, 12601, and 12602 of the reverse strand of chromosome 8 will be:

```
8 12600 12602 CGT/- -
```

VCF

VEP also supports using [VCF \(Variant Call Format\) version 4.0](#). This is a common format used by the 1000 genomes project, and can be produced as an output format by many variant calling tools.

Users using VCF should note a peculiarity in the difference between how Ensembl and VCF describe unbalanced variants. For any unbalanced variant (i.e. insertion, deletion or unbalanced substitution), the VCF specification requires that the base immediately before the variant should be included in both the reference and variant alleles. This also affects the reported position i.e. the reported position will be one base before the actual site of the variant.

In order to parse this correctly, VEP needs to convert such variants into Ensembl-type coordinates, and it does this by removing the additional base and adjusting the coordinates accordingly. This means that if an identifier is not supplied for a variant (in the 3rd column of the VCF), then the identifier constructed and the position reported in VEP's output file will differ from the input.

This problem can be overcome by either:

1. ensuring each variant has a unique identifier specified in the 3rd column of the VCF
2. using VCF format as output ([--vcf](#)) - this preserves the formatting of your input coordinates and alleles

The following examples illustrate how VCF describes a variant and how it is handled internally by VEP. Consider the following aligned sequences (for the purposes of discussion on chromosome 20):

```
Ref: a t C g a // C is the reference base
1 : a t G g a // C base is a G in individual 1
```

```
2 : a t - g a // C base is deleted w.r.t. the reference in individual 2
3 : a t CAg a // A base is inserted w.r.t. the reference sequence in individual 3
```

Individual 1

The first individual shows a simple balanced substitution of G for C at base 3. This is described in a compatible manner in VCF and Ensembl styles. Firstly, in VCF:

```
20    3    .    C    G    .    PASS    .
```

And in Ensembl format:

```
20    3    3    C/G    +
```

Individual 2

The second individual has the 3rd base deleted relative to the reference. In VCF, both the reference and variant allele columns must include the preceding base (T) and the reported position is that of the preceding base:

```
20    2    .    TC    T    .    PASS    .
```

In Ensembl format, the preceding base is not included, and the start/end coordinates represent the region of the sequence deleted. A "-" character is used to indicate that the base is deleted in the variant sequence:

```
20    3    3    C/-    +
```

The upshot of this is that while in the VCF input file the position of the variant is reported as 2, in the output file from VEP the position will be reported as 3. If no identifier is provided in the third column of the VCF, then the constructed identifier will be:

```
20_3_C/-
```

Individual 3

The third individual has an "A" inserted between the 3rd and 4th bases of the sequence relative to the reference. In VCF, as for the deletion, the base before the insertion is included in both the reference and variant allele columns, and the reported position is that of the preceding base:

```
20    3    .    C    CA    .    PASS    .
```

In Ensembl format, again the preceding base is not included, and the start/end positions are "swapped" to indicate that this is an insertion. Similarly to a deletion, a "-" is used to indicate no sequence in the reference:

```
20    4    3    -/A    +
```

Again, the output will appear different, and the constructed identifier may not be what is expected:

```
20_3_-/A
```

The solution is to always add a unique identifier for each of your variants to the VCF file, or use VCF as your output format.

Structural variants

VEP can also call consequences on structural variants encoded in tab-delimited or VCF format. To recognise a variant as a structural variant, the allele string (or "SVTYPE" INFO field in VCF) must be set to one of the currently recognised values:

- **INS** - insertion
- **DEL** - deletion
- **DUP** - duplication
- **TDUP** - tandem duplication

Examples of structural variants encoded in tab-delimited format:

1	160283	471362	DUP
1	1385015	1387562	DEL

Examples of structural variants encoded in VCF format:

#CHROM	POS	ID	REF	ALT	QUAL	FILTER	INFO	FORMAT
1	160283	sv1	.	<DUP>	.	.	SVTYPE=DUP;END=471362	.
1	1385015	sv2	.		.	.	SVTYPE=DEL;END=1387562	.

See the [VCF definition document](#) for more detail on how to describe structural variants in VCF format.

HGVS identifiers

See <http://www.hgvs.org/mutnomen/> for details. These must be relative to genomic or Ensembl transcript coordinates. It also is possible to use RefSeq transcripts in both the web interface and the VEP script (see [script documentation](#)). This works for RefSeq transcripts that align to the genome correctly.

Examples:

```
ENST00000207771.3:c.344+626A>T
ENST00000471631.1:c.28_33delTCGCGG
ENST00000285667.3:c.1047_1048insC
5:g.140532T>C
```

Examples using RefSeq identifiers (using [--refseq](#) in the VEP script, or select the other features transcript database on the web interface and input type of HGVS):

```
NM_153681.2:c.7C>T
NM_005239.4:c.190G>A
NM_001025204.1:c.336G>A
```

HGVS protein notations may also be used, provided that they unambiguously map to a single genomic change. Due to redundancy in the amino acid code, it is not always possible to work out the corresponding genomic sequence change for a given protein sequence change. The following example is for a permissible protein notation in dog (*Canis familiaris*):

```
ENSCAFP00000040171.1:p.Thr92Asn
```

HGVS notations may also be given in [LRG](#) coordinates:

```
LRG_1t1:c.841G>T
LRG_1:g.10006G>T
```

Variant identifiers

These should be e.g. dbSNP rsIDs, or any synonym for a variant present in the Ensembl Variation database. See [here](#) for a list of identifier sources in Ensembl.

Output

The default output format ("VEP" format when downloading from the web interface) is a 14 column tab-delimited file. Empty values are denoted by '.'. The output columns are:

1. **Uploaded variation** - as chromosome_start_alleles
2. **Location** - in standard coordinate format (chr:start or chr:start-end)
3. **Allele** - the variant allele used to calculate the consequence
4. **Gene** - Ensembl stable ID of affected gene
5. **Feature** - Ensembl stable ID of feature
6. **Feature type** - type of feature. Currently one of Transcript, RegulatoryFeature, MotifFeature.
7. **Consequence** - [consequence type](#) of this variant

8. **Position in cDNA** - relative position of base pair in cDNA sequence
9. **Position in CDS** - relative position of base pair in coding sequence
10. **Position in protein** - relative position of amino acid in protein
11. **Amino acid change** - only given if the variant affects the protein-coding sequence
12. **Codon change** - the alternative codons with the variant base in upper case
13. **Co-located variation** - known identifier of existing variant
14. **Extra** - this column contains extra information as key=value pairs separated by ";", see below.

Other output fields

- **IMPACT** - the impact modifier for the consequence type
- **VARIANT_CLASS** - Sequence Ontology [variant class](#)
- **SYMBOL** - the gene symbol
- **SYMBOL_SOURCE** - the source of the gene symbol
- **STRAND** - the DNA strand (1 or -1) on which the transcript/feature lies
- **ENSP** - the Ensembl protein identifier of the affected transcript
- **FLAGS** - transcript quality flags:
 - *cds_start_NF*: CDS 5' incomplete
 - *cds_end_NF*: CDS 3' incomplete
- **SWISSPROT** - Best match UniProtKB/Swiss-Prot accession of protein product
- **TREMBL** - Best match UniProtKB/TrEMBL accession of protein product
- **UNIPARC** - Best match UniParc accession of protein product
- **HGVSc** - the HGVS coding sequence name
- **HGVSp** - the HGVS protein sequence name
- **HGVSG** - the HGVS genomic sequence name
- **HGVS_OFFSET** - Indicates by how many bases the HGVS notations for this variant have been [shifted](#)
- **NEAREST** - Identifier(s) of nearest transcription start site
- **SIFT** - the SIFT prediction and/or score, with both given as prediction(score)
- **PolyPhen** - the PolyPhen prediction and/or score
- **MOTIF_NAME** - the source and identifier of a transcription factor binding profile aligned at this position
- **MOTIF_POS** - The relative position of the variation in the aligned TFBP
- **HIGH_INF_POS** - a flag indicating if the variant falls in a high information position of a transcription factor binding profile (TFBP)
- **MOTIF_SCORE_CHANGE** - The difference in motif score of the reference and variant sequences for the TFBP
- **CELL_TYPE** - List of cell types and classifications for regulatory feature
- **CANONICAL** - a flag indicating if the transcript is denoted as the canonical transcript for this gene
- **CCDS** - the CCDS identifier for this transcript, where applicable
- **INTRON** - the intron number (out of total number)
- **EXON** - the exon number (out of total number)
- **DOMAINS** - the source and identifier of any overlapping protein domains
- **DISTANCE** - Shortest distance from variant to transcript
- **IND** - individual name
- **ZYG** - zygosity of individual genotype at this locus
- **SV** - IDs of overlapping structural variants
- **FREQS** - Frequencies of overlapping variants used in filtering
- **AF** - Frequency of existing variant in 1000 Genomes
- **AFR_AF** - Frequency of existing variant in 1000 Genomes combined African population
- **AMR_AF** - Frequency of existing variant in 1000 Genomes combined American population

- **ASN_AF** - Frequency of existing variant in 1000 Genomes combined Asian population
- **EUR_AF** - Frequency of existing variant in 1000 Genomes combined European population
- **EAS_AF** - Frequency of existing variant in 1000 Genomes combined East Asian population
- **SAS_AF** - Frequency of existing variant in 1000 Genomes combined South Asian population
- **AA_AF** - Frequency of existing variant in NHLBI-ESP African American population
- **EA_AF** - Frequency of existing variant in NHLBI-ESP European American population
- **ExAC_AF** - Frequency of existing variant in ExAC combined population
- **ExAC_Adj_AF** - Adjusted frequency of existing variant in ExAC combined population
- **ExAC_AFR_AF** - Frequency of existing variant in ExAC African/American population
- **ExAC_AMR_AF** - Frequency of existing variant in ExAC American population
- **ExAC_EAS_AF** - Frequency of existing variant in ExAC East Asian population
- **ExAC_FIN_AF** - Frequency of existing variant in ExAC Finnish population
- **ExAC_NFE_AF** - Frequency of existing variant in ExAC Non-Finnish European population
- **ExAC_OTH_AF** - Frequency of existing variant in ExAC combined other combined populations
- **ExAC_SAS_AF** - Frequency of existing variant in ExAC South Asian population
- **MAX_AF** - Maximum observed allele frequency in 1000 Genomes, ESP and ExAC
- **MAX_AF_POPS** - Populations in which maximum allele frequency was observed
- **CLIN_SIG** - ClinVar clinical significance of the dbSNP variant
- **BIOTYPE** - Biotype of transcript or regulatory feature
- **APPRIS** - Annotates alternatively spliced transcripts as primary or alternate based on a range of computational methods. NB: not available for GRCh37
- **TSL** - Transcript support level. NB: not available for GRCh37
- **PUBMED** - Pubmed ID(s) of publications that cite existing variant
- **SOMATIC** - Somatic status of existing variant(s)
- **PHENO** - Indicates if existing variant is associated with a phenotype, disease or trait
- **GENE_PHENO** - Indicates if overlapped gene is associated with a phenotype, disease or trait
- **ALLELE_NUM** - Allele number from input; 0 is reference, 1 is first alternate etc
- **MINIMISED** - Alleles in this variant have been converted to minimal representation before consequence calculation
- **PICK** - indicates if this block of consequence data was picked by [--flag_pick](#) or [--flag_pick allele](#)
- **REFSEQ_MATCH** - the RefSeq transcript match status; contains a number of flags indicating whether this RefSeq transcript matches the underlying reference sequence and/or an Ensembl transcript ([more information](#)). NB: not available for GRCh37.
 - *rseq_3p_mismatch*: signifies a mismatch between the RefSeq transcript and the underlying primary genome assembly sequence. Specifically, there is a mismatch in the 3' UTR of the RefSeq model with respect to the primary genome assembly (e.g. GRCh37/GRCh38).
 - *rseq_5p_mismatch*: signifies a mismatch between the RefSeq transcript and the underlying primary genome assembly sequence. Specifically, there is a mismatch in the 5' UTR of the RefSeq model with respect to the primary genome assembly.
 - *rseq_cds_mismatch*: signifies a mismatch between the RefSeq transcript and the underlying primary genome assembly sequence. Specifically, there is a mismatch in the CDS of the RefSeq model with respect to the primary genome assembly.
 - *rseq_ens_match_cds*: signifies that for the RefSeq transcript there is an overlapping Ensembl model that is identical across the CDS region only. A CDS match is defined as follows: the CDS and peptide sequences are identical and the genomic coordinates of every translatable exon match. Useful related attributes are: *rseq_ens_match_wt* and *rseq_ens_no_match*.
 - *rseq_ens_match_wt*: signifies that for the RefSeq transcript there is an overlapping Ensembl model that is identical across the whole transcript. A whole transcript match is defined as follows: 1) In the case that both models are coding, the transcript, CDS and peptide sequences are all identical and the genomic coordinates of every exon match. 2) In the case that both transcripts are non-coding the transcript sequences and the genomic coordinates of every exon are identical. No comparison is made between a coding and a non-coding transcript. Useful related attributes are: *rseq_ens_match_cds* and *rseq_ens_no_match*.
 - *rseq_ens_no_match*: signifies that for the RefSeq transcript there is no overlapping Ensembl model that is identical across either the whole transcript or the CDS. This is caused by differences between the transcript, CDS or peptide sequences or between the exon genomic coordinates. Useful related attributes are: *rseq_ens_match_wt* and *rseq_ens_match_cds*.

- *rseq_mrna_match*: signifies an exact match between the RefSeq transcript and the underlying primary genome assembly sequence (based on a match between the transcript stable id and an accession in the RefSeq mRNA file). An exact match occurs when the underlying genomic sequence of the model can be perfectly aligned to the mRNA sequence post polyA clipping.
- *rseq_mrna_nonmatch*: signifies a non-match between the RefSeq transcript and the underlying primary genome assembly sequence. A non-match is deemed to have occurred if the underlying genomic sequence does not have a perfect alignment to the mRNA sequence post polyA clipping. It can also signify that no comparison was possible as the model stable id may not have had a corresponding entry in the RefSeq mRNA file (sometimes happens when accessions are retired or changed). When a non-match occurs one or several of the following transcript attributes will also be present to provide more detail on the nature of the non-match: *rseq_5p_mismatch*, *rseq_cds_mismatch*, *rseq_3p_mismatch*, *rseq_nctran_mismatch*, *rseq_no_comparison*
- *rseq_nctran_mismatch*: signifies a mismatch between the RefSeq transcript and the underlying primary genome assembly sequence. This is a comparison between the entire underlying genomic sequence of the RefSeq model to the mRNA in the case of RefSeq models that are non-coding.
- *rseq_no_comparison*: signifies that no alignment was carried out between the underlying primary genome assembly sequence and a corresponding RefSeq mRNA. The reason for this is generally that no corresponding, unversioned accession was found in the RefSeq mRNA file for the transcript stable id. This sometimes happens when accessions are retired or replaced. A second possibility is that the sequences were too long and problematic to align (though this is rare).

11_224088_C/A	11:224088	A	ENSG00000142082	ENST00000525319	Transcript	missense_va
11_224088_C/A	11:224088	A	ENSG00000142082	ENST00000534381	Transcript	5_prime_UTR
11_224088_C/A	11:224088	A	ENSG00000142082	ENST00000529055	Transcript	downstream
11_224585_G/A	11:224585	A	ENSG00000142082	ENST00000529937	Transcript	intron_vari
22_16084370_G/A	22:16084370	A	-	ENSR00000615113	RegulatoryFeature	regulatory_

The VEP script will also add a header to the output file. This contains information about the databases connected to, and also a key describing the key/value pairs used in the extra column.

```
## ENSEMBL VARIANT EFFECT PREDICTOR v88.0
## Output produced at 2017-03-21 14:51:27
## Connected to homo_sapiens_core_88_38 on ensembl.ensembl.org
## Using cache in /homes/user/.vep/homo_sapiens/88_GRCh38
## Using API version 88, DB version 88
## polyphen version 2.2.2
## sift version sift5.2.2
## COSMIC version 78
## ESP version 20141103
## gencode version GENCODE 25
## genebuild version 2014-07
## HGMD-PUBLIC version 20162
## regbuild version 16
## assembly version GRCh38.p7
## ClinVar version 201610
## dbSNP version 147
## Column descriptions:
## Uploaded_variation : Identifier of uploaded variant
## Location : Location of variant in standard coordinate format (chr:start or chr:start-end)
## Allele : The variant allele used to calculate the consequence
## Gene : Stable ID of affected gene
## Feature : Stable ID of feature
## Feature_type : Type of feature - Transcript, RegulatoryFeature or MotifFeature
## Consequence : Consequence type
## cDNA_position : Relative position of base pair in cDNA sequence
## CDS_position : Relative position of base pair in coding sequence
## Protein_position : Relative position of amino acid in protein
## Amino_acids : Reference and variant amino acids
## Codons : Reference and variant codon sequence
## Existing_variation : Identifier(s) of co-located known variants
## Extra column keys:
## IMPACT : Subjective impact classification of consequence type
## DISTANCE : Shortest distance from variant to transcript
## STRAND : Strand of the feature (1/-1)
## FLAGS : Transcript quality flags
```

Tab-delimited output

The [--tab](#) flag instructs VEP to write output as a tab-delimited table. This differs from the default output format in that each individual field from the "Extra" field is written to a separate tab-delimited column. This makes the output more suitable for import into spreadsheet programs such as Excel. This is also the format used when selecting the "TXT" option on the VEP web interface.

The choice and order of columns in the output may be configured using [--fields](#).

VCF output

The VEP script can also generate VCF output using the [--vcf](#) flag. Consequences are added in the INFO field of the VCF file, using the key "CSQ" (configure this using [--vcf_info_field](#)). Data fields are encoded separated by "|"; the order of fields is written in the VCF header. Output fields can be configured by using [--fields](#). Unpopulated fields are represented by an empty string.

VCFs produced by VEP can be filtered by [filter_vep.pl](#) in the same way as standard format output files.

If the input format was VCF, the file will remain unchanged save for the addition of the CSQ field and the header (unless using any filtering). If an existing CSQ field is found, it will be replaced by the one added by the VEP (use [--keep_csq](#) to preserve it).

Custom data added with [--custom](#) are added as separate fields, using the key specified for each data file.

Commas in fields are replaced with ampersands (&) to preserve VCF format.

```
##INFO=<ID=CSQ,Number=.,Type=String,Description="Consequence annotations from Ensembl VEP. Format:
#CHROM POS ID REF ALT QUAL FILTER INFO
21 26978790 rs75377686 T C . . CSQ=C|missense_variant|MODERATE|MRPL39|ENS
```

JSON output

VEP can produce output in the form of serialised [JSON](#) objects using the [--json](#) flag. JSON is a serialisation format that can be parsed and processed easily by many packages and programming languages; it is used as the default output format for [Ensembl's REST server](#).

Each input variant is reported as a single JSON object which constitutes one line of the output file. The JSON object is structured somewhat differently to the other VEP output formats, in that per-variant fields (e.g. co-located existing variant details) are reported only once. Consequences are grouped under the feature type that they affect (Transcript, Regulatory Feature, etc). The original input line (e.g. from VCF input) is reported under the "input" key in order to aid aligning input with output.

Here follows an example of JSON output (prettified and redacted for display here):

```
{
  "input": "1 230845794 test1 A G . . .",
  "id": "test1",
  "seq_region_name": "1",
  "start": 230845794,
  "end": 230845794,
  "strand": 1,
  "allele_string": "A/G",
  "most_severe_consequence": "missense_variant",
  "colocated_variants": [
    {
      "id": "rs699",
      "seq_region_name": "1",
      "start": 230845794,
      "end": 230845794,
      "strand": 1,
      "allele_string": "A/G",
      "minor_allele": "A",
      "minor_allele_freq": 0.3384,
      "afr_allele": "A",
      "afr_maf": 0.13,
      "amr_allele": "A",
      "amr_maf": 0.36,
      "asn_allele": "A",
      "asn_maf": 0.16,
      "eur_allele": "A",
```



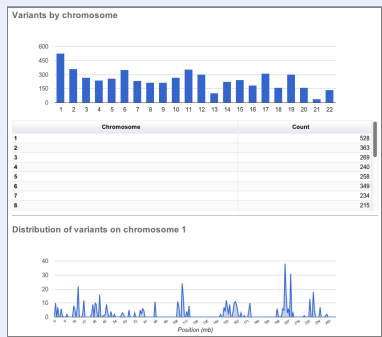
```

    "eur_maf": 0.41,
    "pubmed": [
      18513389,
      23716723
    ]
  },
  {
    "seq_region_name": "1",
    "strand": 1,
    "id": "COSM425562",
    "allele_string": "A/G",
    "start": 230845794,
    "end": 230845794
  }
],
"transcript_consequences": [
  {
    "variant_allele": "G",
    "consequence_terms": [
      "missense_variant"
    ],
    "gene_id": "ENSG00000135744",
    "gene_symbol": "AGT",
    "gene_symbol_source": "HGNC",
    "transcript_id": "ENST00000366667",
    "biotype": "protein_coding",
    "strand": -1,
    "cdna_start": 1018,
    "cdna_end": 1018,
    "cds_start": 803,
    "cds_end": 803,
    "protein_start": 268,
    "protein_end": 268,
    "codons": "aTg/aCg",
    "amino_acids": "M/T",
    "polyphen_prediction": "benign",
    "polyphen_score": 0,
    "sift_prediction": "tolerated",
    "sift_score": 1,
    "hgvc": "ENST00000366667.4:c.803T>C",
    "hgvsp": "ENSP00000355627.4:p.Met268Thr"
  }
],
"regulatory_feature_consequences": [
  {
    "variant_allele": "G",
    "consequence_terms": [
      "regulatory_region_variant"
    ],
    "regulatory_feature_id": "ENSR00001529861"
  }
]
}

```

In accordance with JSON conventions, all keys are lower-case. Some keys also have different names and structures to those found in the other VEP output formats:

Key	JSON equivalent(s)	Notes
Consequence	consequence_terms	
Gene	gene_id	
Feature	transcript_id, regulatory_feature_id, motif_feature_id	Consequences are grouped under the feature type they affect
ALLELE	variant_allele	
SYMBOL	gene_symbol	
SYMBOL_SOURCE	gene_symbol_source	



Distribution of variants across chromosomes

For any questions not covered here, please send an email to the Ensembl [developer's mailing list](#) (public) or contact the [Ensembl Helpdesk](#) (private).

General questions

Q: Why don't I see any co-located variations when using species X?

A: Ensembl only has variation databases for a subset of all Ensembl species - see [this document](#) for details.

Q: Why has my insertion/deletion variant encoded in VCF disappeared from the VEP output?

A: Ensembl treats unbalanced variants differently to VCF - your variant hasn't disappeared, it may have just changed slightly! You can solve this by giving your variants a unique identifier in the third column of the VCF file. See [here](#) for a full discussion.

Q: Why do I see so many lines of output for each variant in my input?

A: While it can be convenient to search for a easy, one word answer to the question "What is the consequence of this variant?", in reality biology does not make it this simple! Many genes have more than one transcript, so VEP provides a prediction for each transcript that a variant overlaps. The VEP script can help here; the [--canonical](#) and [--ccds](#) options indicate which transcripts are canonical and belong to the CCDS set respectively, while [--pick](#), [--per_gene](#), [--summary](#) and [--most_severe](#) allow you to give a more summary level assessment per variant.

Furthermore, several "compound" consequences are also possible - if, for example, a variant falls in the final few bases of an exon, it may be considered to affect a splicing site, in addition to possibly affecting the coding sequence.

Since we cannot possibly predict the exact biology of what will happen, what we provide is the most conservative estimate that covers all reasonable scenarios. It is up to you, the user, to interpret this information!

Web VEP questions

Q: How do I access the web version of the Variant Effect Predictor?

A: You can find the web VEP on the [Tools](#) page.

Q: Why is the output I get for my input file different when I use the web VEP and the VEP script?

A: Ensure that you are passing equivalent arguments to the script that you are using in the web version. If you are sure this is still a problem, please report it on the [ensembl-dev](#) mailing list.

VEP script questions

Q: How can I make VEP run faster?

There are a number of factors that influence how fast VEP runs. Have a look at our [handy guide](#) for tips on improving VEP runtime.

Q: Why do I see "N" as the reference allele in my HGVS strings?

Q: Why do I see the following error (or similar) in my VEP output?

```
substr outside of string at /nfs/users/nfs_w/wm2/Perl/ensembl-variation/modules/Bio/Ensembl/Vari
Use of uninitialized value $ref_allele in string eq at /nfs/users/nfs_w/wm2/Perl/ensembl-variati
Use of uninitialized value in concatenation (.) or string at /nfs/users/nfs_w/wm2/Perl/ensembl-v
```

Both of these error types are usually seen when using a [FASTA file](#) for retrieving sequence. There are a couple of steps you can take to try to remedy them:

1. The index alongside the FASTA can become corrupted. Delete [fastafile].index and re-run VEP to regenerate it. By default this file is located in your \$HOME/.vep/[species]/[version]_[assembly] directory.
2. The FASTA file itself may have been corrupted during download; delete the fasta file and the index and re-download (you can use the [VEP installer](#) to do this).
3. Older versions of BioPerl (1.2.3 in particular is known to have this) cannot properly index large FASTA files. Make sure you are using a later (≥ 1.6) version of BioPerl. The [VEP installer](#) installs 1.6.1 for you.

If you still see problems after taking these steps, or if you were not using a FASTA file in the first place, please [contact us](#).

Q: Why do I see the following warning?

```
WARNING: Chromosome 21 not found in annotation sources or synonyms on line 160
```

This can occur if the chromosome names differ between your input variant and any annotation source that you are using (cache, database, GFF/GTF file, FASTA file, custom annotation file). To circumvent this you may provide VEP with a [synonyms file](#). A synonym file is included in VEP's cache files, so if you have one of these for your species you can use it as follows:

```
./vep -i input.vcf -cache -synonyms ~/.vep/homo_sapiens/88_GRCh38/chr_synonyms.txt
```

The file consists of lines containing pairs of tab-separated synonyms. Order is not important as synonyms can be used in both "directions".

Q: Can I get gnomAD allele frequencies in VEP?

Yes, see [this guide](#).

Q: Why do I see the following error?

```
Could not connect to database homo_sapiens_core_63_37 as user anonymous using [DBI:mysql:database=
Unknown MySQL server host 'ensembldb.ensembl.org' (2) at $HOME/src/ensembl/modules/Bio/EnsEMBL/I

----- EXCEPTION -----
MSG: Could not connect to database homo_sapiens_core_63_37 as user anonymous using [DBI:mysql:da
Unknown MySQL server host 'ensembldb.ensembl.org' (2)
```

A: By default the VEP script is configured to connect to the public MySQL server at ensembldb.ensembl.org. Occasionally the server may break connection with your script, which causes this error. This can happen when the server is busy, or due to various network issues. Consider using a [local copy of the database](#), or the [caching system](#).

Q: Can I use the VEP script on Windows?

Yes - see the [documentation](#) for a few different ways to get the VEP running on Windows.

Q: Can I download all of the SIFT and/or PolyPhen predictions?

A: The Ensembl Variation database and the human VEP cache file contain precalculated SIFT and PolyPhen predictions for every possible amino acid change in every translated protein product in Ensembl. Since these data are huge, we store them in a compressed format. The best approach to extract them is to use our Perl API.

The format in which the data are stored in our database is described [here](#)

The simplest way to access these matrices is to use an API script to fetch a ProteinFunctionPredictionMatrix for your protein of interest and then call its 'get_prediction' method to get the score for a particular position and amino acid, looping over all possible amino acids for your position. There is some detailed documentation on this class in the API documentation [here](#).

You would need to work out which peptide position your codon maps to, but there are methods in the [TranscriptVariationAllele](#) class that should help you (probably translation_start and translation_end).

