Next Generation Sequencing Technologies: Data Analysis and Applications

SAM format Alignment section

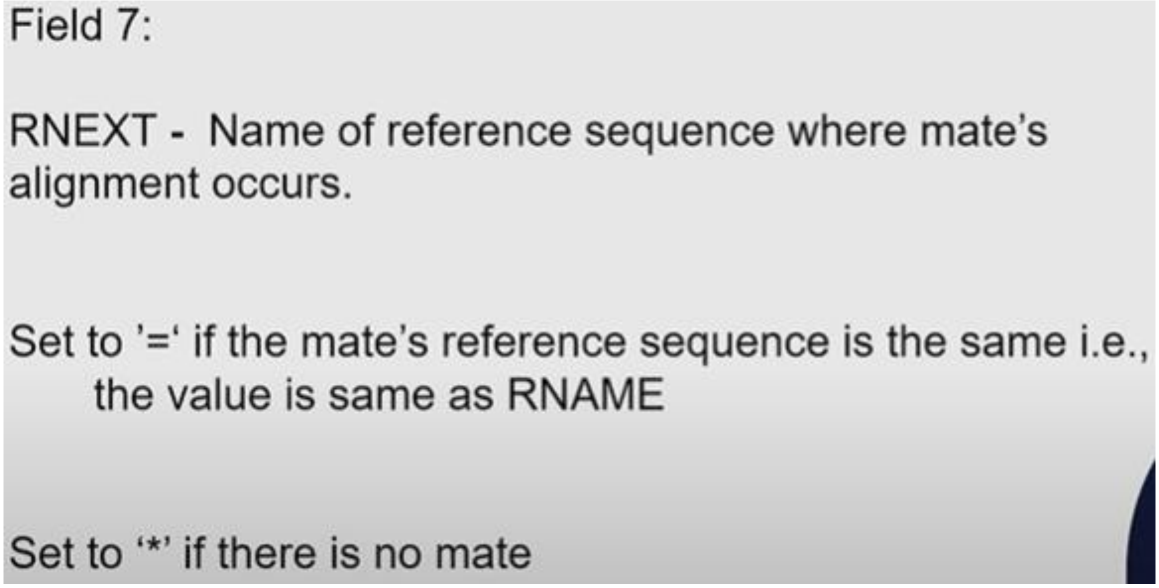
Dr. Riddhiman Dhar, Department of Biotechnology

Indian Institute of Technology Kharagpur

Good day, everyone. Welcome to the course on Generation Sequencing Technologies, Data Analysis, and Applications. In the last class, we discussed the SAM format, the alignment section, and some of the fields that are present. So, we will continue that, and we will complete the SAM file format in this class, right? So, you will be able to completely understand what kind of information you have in that file, ok? And if you are writing code, this would be very useful.

You can process these SAM files, and you can extract some information. For example, if you want to identify the genetic variants from your mapping or other kind of information, you will be able to extract those by writing codes. So, this is the topic that we will be covering today. We will discuss the alignment section and what kind of information you have in it.

These are the keywords for mandatory fields and optional fields. We have already started discussing the mandatory fields, right? So, we have discussed the first six fields, and this is where we ended in the last class, right? So, we talked about CIGAR strings and we have seen what kind of letters we use to represent CIGAR strings, right? So, what we will do is actually look at or interpret the CIGAR string from the real data in the real output file, and we will understand what information you have in this CIGAR string.



Field 7:

RNEXT - Name of reference sequence where mate’s alignment occurs.

Set to ’=‘ if the mate’s reference sequence is the same i.e., the value is same as RNAME

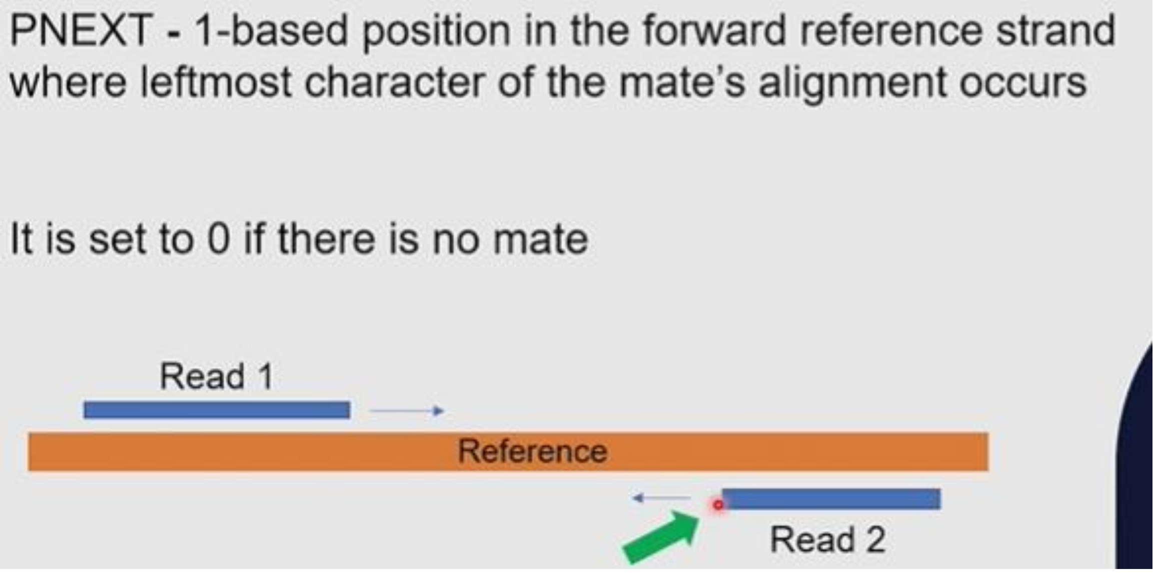
Set to ‘*’ if there is no mate

So, we have these letters, but you also have some numbers, right? So, what kind of reason do you add those numbers? What do they mean? We will explain when we actually go to the real data set. So, moving on to field 7, we have something called RNEXT, ok? So, up to field 6, you are getting alignment information for the read itself, ok? So, the read that we are working with the Q name, right, that information is up to field 6, ok?

From field 7 onwards, if you are dealing with paired-end data, it will give you information about the mate alignment, ok? So, you have read 1, read 2. If you are looking at read 1, up to field 6, you will get alignment information for read 1, but from field 7 onwards, you will have alignment information for read 2. So, the R next means the name of the reference sequence where mate alignment occurs, right? So, similarly to similar to this ref n c some number, right?

So, here you will also get some reference sequence name or ID, right? That will be in field 7. So, this will actually be set to equal the sign, right? So, this will mean that the mate reference sequence is the same as the read reference sequence, right? So, if this reference sequence name is the same as the R name, this is the reference. So, they are mapping to the same chromosome or the same reference genome. That kind of information will be given here.

So, this is set to star, right? If there is no mate if you are dealing with single-end data, this is not present, ok? The next field is field 8. This gives you PNEXT. So, this is again POS equivalent for the mate, ok? So, this gives you the position of the mate's alignment in the reference track.

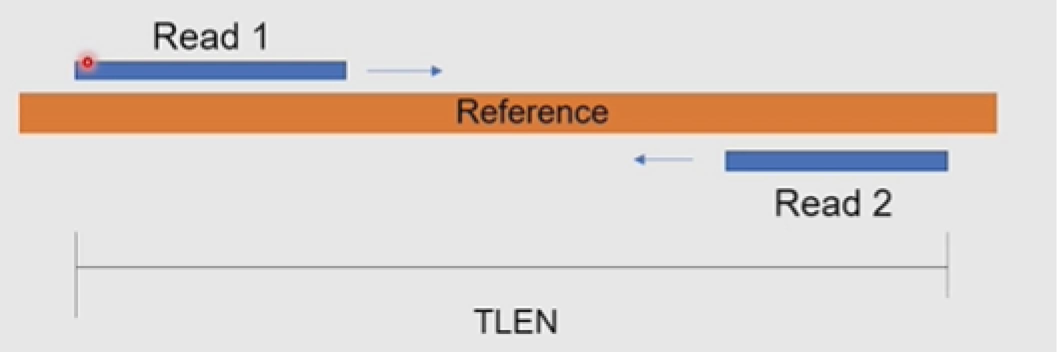

PNEXT - 1-based position in the forward reference strand where leftmost character of the mate's alignment occurs

It is set to 0 if there is no mate

Again, it is based on one base positioning system, which we just described, and this is again describing the leftmost character location in the mate alignment, ok? So, leftmost position. So, as you can see this is kind of consistent for POS as well as for P next, ok, and it is set to 0 if there is no mate in the data, right? So, this is if I just highlight that this is the position that you get, right? So, if you have read 1, read 2, this is the position that will be represented in PNEXT, right the coordinate system, right.

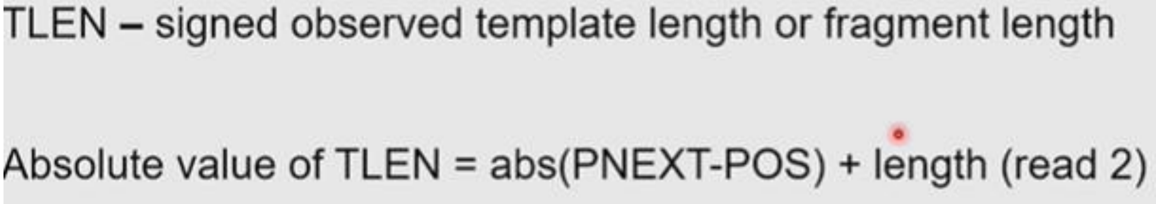
Again, this could be some number we have seen in the real data in the last class, the POS. Right, it has some number. So, here, you will also get some numbers, ok? So, what about field 9, ok? So, field 9 actually gives you something called the signed observed template length or fragment length, ok? So, this is important, right? If you are dealing with paired-end data, you can probably generate what kind of fragment length you got from your experimental procedure.

So, when you fragment the genome, you want to know what kind of fragment length distribution you got, and that information could be useful the next time you want to prepare and sequence it. So, here you get this template length or fragment length. So, you are dealing with paired-end data. You have these fragments, these reads coming from both ends, and we want to calculate the fragment length that actually generated these two reads, read 1 and read 2. So, this is given in TLEN, right? So, this absolute value of TLEN is actually end minus start plus 1, right?



So, that is very simple, right? If you have this end here, the mapping ends here, right on the right. So, you have this end, and then you have the start. So, we can take this example, right? So, that will be easier to understand, right? So, n means this is probably the end, right? This is the end position, and this is the start position here on this side, right?

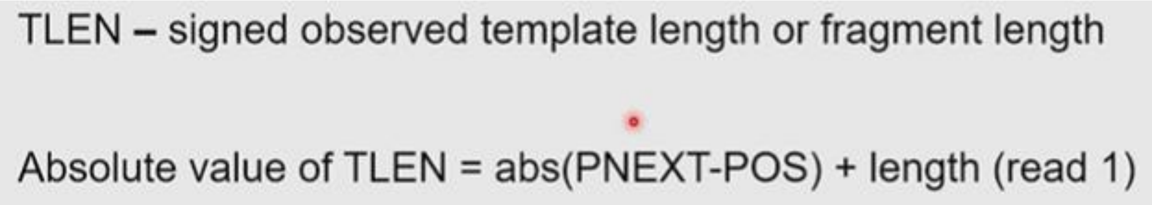
So, this is n minus start plus 1 that would give you TLEN, ok? Now, one of the things you probably have noticed is a signed observed template length. So, we have calculated TLEN, right? This is a positive number, but we will also have to have a sign, ok? So, there is a plus sign and a minus sign, and you will understand why we need a sign. Okay, because we are dealing with paired-end data, one of the reads will get a plus sign and the other read will get a minus sign for the same TLEN, right? It is important to actually figure out which read is occurring upstream of which read, right?



TLEN – signed observed template length or fragment length

Absolute value of TLEN = abs(PNEXT-POS) + length (read 2)

So, whether read 2 is upstream of read 1 or read 1 is upstream of read 2, you can gather that information from this sign of TLEN. So, as I said, right in case you have this situation, right? So, you have this absolute value of TLEN, right? So, this is actually how you calculate from this PNEXT and POS, right? So, if you are looking at, let us say, read 1 alignment, you want to calculate the fragment length.



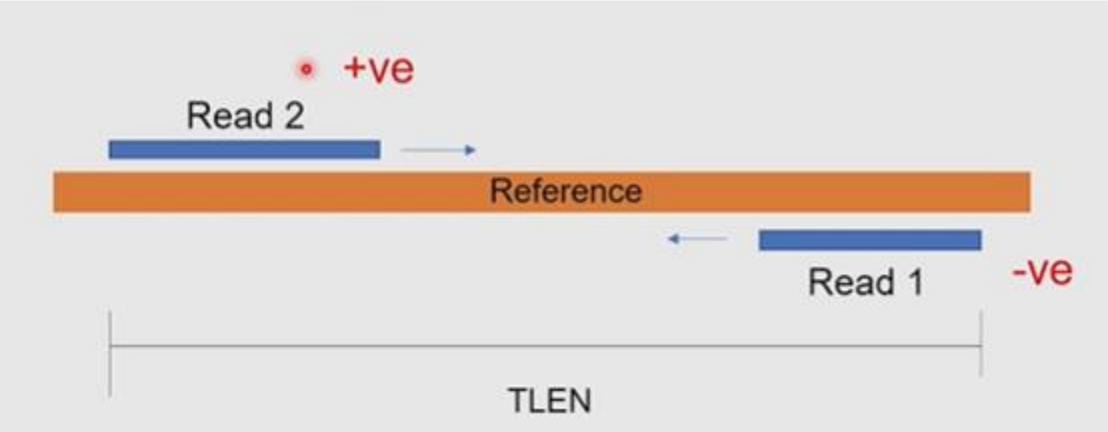
TLEN – signed observed template length or fragment length

Absolute value of TLEN = abs(PNEXT-POS) + length (read 1)

So, how do you go about it? So, remember that this POS gives you the left-most position here, and PNEXT gives you the left-most position here, right? So, you get this absolute value of PNEXT minus POS plus the length of read 2, right? So, here you get the left most positions in the coordinate system, right, and then you have to add the length of the read 2. So, based on some of these fields, you can calculate the TLEN value, ok? So, in case this is an interchange, right, you have this read 2 upstream of read 1, and then it will be replaced by length read 1, ok?

So, this part remains the same absolute PNEXT minus POS plus length of read 1. Now, in the case of Illumina you are working with, let us say read 1 read 2 data, which are of the same length. Then it does not matter; you can use read 1 read 2, but also be careful. If you are working with trim data, the length of read 1 and read 2 may not be the same, right? So, in that case, you have to check again carefully. Now, what about the sign? How do we actually give the sign? So, the sign is very simple.

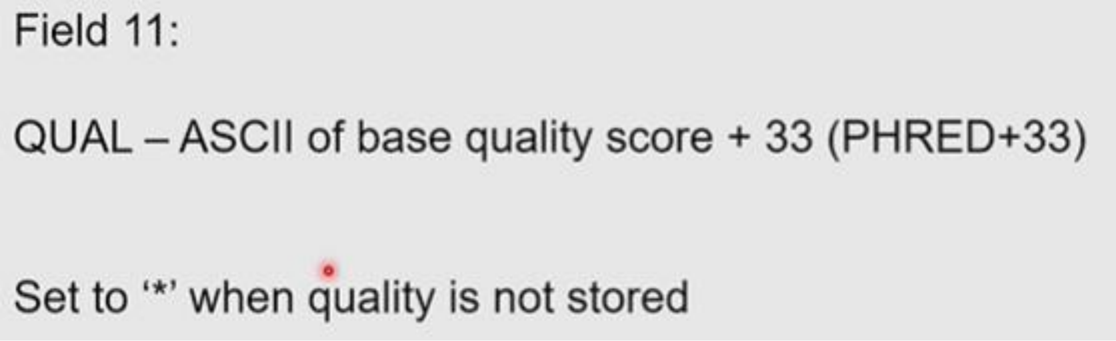
So, I give a positive sign for the left-most segment. So, if read 1 is upstream of lead 2, right here is the leftmost segment. So, if you get a positive sign for the rightmost segment that read 2 is downstream, then read 2 will get a negative sign. The TLEN value and the absolute value remain the same; you will simply change the sign, ok? So, this is what this figure illustrates: how we generate the plus and minus signs.



So, read 2 in this case is upstream of read 1. So, this is the left-most segment. So, this is a positive sign for TLEN, right? So, let us say TLEN is 300. So, for read 2 in the TLEN field, you will have plus 300, right?

So, or simply 300? You do not say plus 300. If you read 1 in the TLEN field, you will get minus 300, right? So, as we go into the real data, you will actually see these signs. So, this value is actually valid only for paired-end data, right? So, if you are dealing with single-end data or an unmapped pair you cannot generate this fragment layer, right? Because you do not know the fragment layer if you are dealing with just sequence data from one end.

So, the value is set to 0, ok? So, the final two fields are actually field 10 and field 11, right? So, field 10 is actually very simple; this is the segment or read sequence, ok? So, the read that you are working with is a read sequence, ok? Again, going back to the first part, as I mentioned, everything is given with respect to the forward reference strand. So, if a read is mapping to the reverse strand, you will get the reverse complement sequence in this field, right?



Field 11:

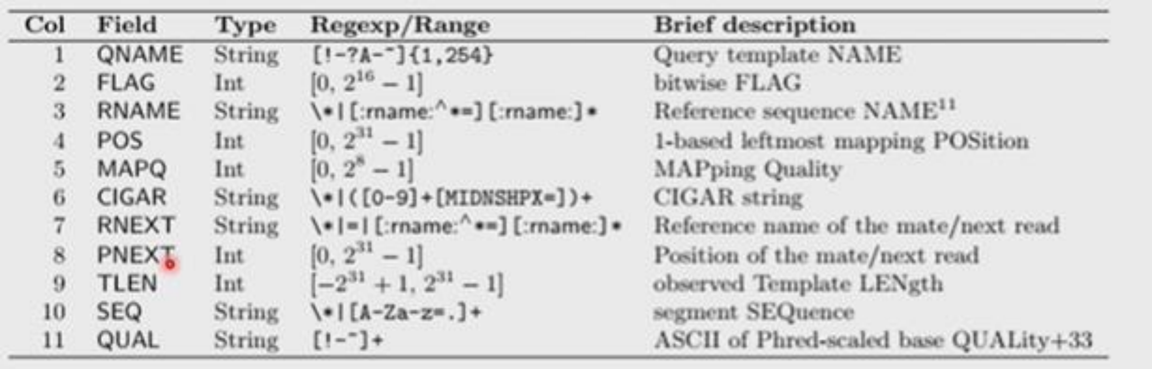
QUAL – ASCII of base quality score + 33 (PHRED+33)

Set to '*' when quality is not stored

So, SEQ will give you the reverse complement sequence, ok? So, this is something you need to keep in mind, ok? Now, if the sequence is not stored, then you get set to this is set to start and the final field, right? So, this is the quality score again; this is the same as the PHRED score that we get from the FASTQ 5, ok? So, this is an ASCII encoded data simply copied from this FASTQ 5, right, and this is in PHRED plus 33 encoding. Right, the new Illumina data for older Illumina data will be PHRED plus 64, ok, and this is set to start if the qualitative score is not stored.

So, sometimes you may not want the sequence or the qualitative score in this same file, right? Because it simply increases the size of the file, and this information is already present in the FASTQ file. So, may not have required these fields, ok? So, just to summarize the compulsory fields that you have discussed so far, ok? This is just a brief summary of what you will get, right? So, you have Q name, which is a string; right, this is a name for the read; you have flag, which is an integer number; right,

So, we can expect some numbers. As we have seen this is a sum of some of the applicable flags. You have something called RNAME, which is again a string. This is the reference name. The reference sequence name could be the genome or chromosome number ID name. Then you have the fourth field, which is the POS; right, this is an integer; right, and again, following this one-based positioning system, this is the leftmost position, which we have discussed again. The fifth field is MAPQ. This is the mapping quality, which gives you the probability of error in the alignment. And then finally, field 6 is the CIGAR string. Again, we have seen at least described how this CIGAR string is derived. Right, what do these letters mean? You have described some letters like M, I, D, H, S, etcetera.



Col	Field	Type	Regexp/Range	Brief description
1	QNAME	String	`[!-?A-~]{1,254}`	Query template NAME
2	FLAG	Int	$[0, 2^{16} - 1]$	bitwise FLAG
3	RNAME	String	`\*|[:rname:^*=][:rname:]*`	Reference sequence NAME[11]
4	POS	Int	$[0, 2^{31} - 1]$	1-based leftmost mapping POSition
5	MAPQ	Int	$[0, 2^{8} - 1]$	MAPping Quality
6	CIGAR	String	`\*|([0-9]+[MIDNSHPX=])+`	CIGAR string
7	RNEXT	String	`\*|=|[:rname:^*=][:rname:]*`	Reference name of the mate/next read
8	PNEXT	Int	$[0, 2^{31} - 1]$	Position of the mate/next read
9	TLEN	Int	$[-2^{31} + 1, 2^{31} - 1]$	observed Template LENgth
10	SEQ	String	`\*|[A-Za-z=.]+`	segment SEQuence
11	QUAL	String	`[!-~]+`	ASCII of Phred-scaled base QUALity+33

Then you have field 7, which is the next, right? This describes these two fields. The next two fields describe the alignment of the mate in the case of paired-end data, right? So, if you have single-end data, you will not get any value, right? So, RNEXT gives the string. So, this is set to equal if it is the same as RNAME, right? So, if RNEXT and RNAME are equal, then you will get this equal sign.

Then PNEXT is the position, right, of the mate in the reference sequence, and we can derive the fragment length, which is given in the ninth field, right. So, this is T length again. We have described how we actually derived this T length value. The final two fields simply give the sequence of the read or reverse complement and the quality score that comes from the first cube file. So, what we will going to do is we are now going to go into the file and we will be going to we are going to check these fields, right, and then how they look like, ok. So, let us do that, and then beyond that, we will discuss the optional fields that are there and what kind of information you get from the optional fields, ok?

So, let us now go to the terminal and enter that file, ok? So, here is the file again, right? So, we have come back to this output file, the output dot sam, and we have already described it up to this point, right? So, this field here is 44. This number that you see describes the mapping quality, right? The next one, as I mentioned, is the sigma string, ok.

So, you can see this 65 m, and then you have 125 m, etcetera, and sometimes you will see this 2 m, 123 m, etcetera. So, as we have described some of these letters, So, m means match. So, match means there is a read, right?

There is no insertion or division. There is a base for every base in that. So, what do we indicate by these numbers, 65 or 125? So, 65 means you have a 65-base match, ok? This could be a match or mismatch; again, we do not count that here.

We cannot get that information here, ok? So, 65 m is good because we have, like, the full alignment for this read that we have, right? So, this information tells us, OK, the full read has been aligned against the reference sequence. 125 m means we have 125 bases in the read sequence that have been aligned against the reference sequence, and there is no insertion or deletion, ok? And these are ideal scenarios, right? So, if you have very good-quality data, most of these will be right.

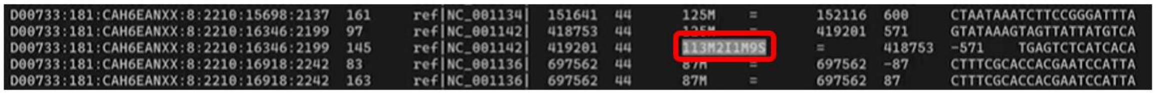
You will not see anything else most of the time. But then here you have these 2 s 123 m. So, what is this? So, s stands for soft clipping, right? So, soft clipping means these bases were clipped from alignment. So, they were not aligned, ok, but they are present in the read sequence in the SAM file, ok.

In the read sequence, they are retained, ok? So, this is what this means now with these numbers: 2 s 123 m. So, 2 s means first 2 bases, right? So, we start, right? The first two bases were soft clips. They are not used for alignment, and then we have 123 base matches, ok?

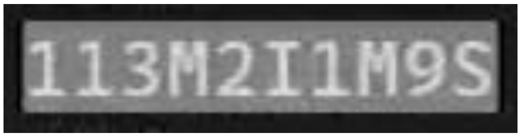
So, this CIGAR string actually signifies that, ok? So, these 2s or s will appear if you are going for local alignment. You will see a lot of these signs, which means there will be a lot of soft clipping because you are going for local alignment. So, you can continue, right, and again, you can interpret this one 1 s 124 m. It means we have the first base soft clip, then the next 124 bases have been aligned, ok?

So, I think this is clear now. So, let us look at a bit more complicated example, right? So, here, for example, you have this 46 m, right? You can see I am highlighting with my pointer. So, we have 46 m 1 i 78 m, ok? What does this mean? It means the first 46 bases match perfectly with the reference.

Then you have one insertion, ok, and then you have 78 bases matching again, ok. So, you have insertion in between, and on one side, you have 46 matches, and on the other side, you have 78 matches, ok? This is what this CIGAR tells us, ok? And similarly, you can actually search a bit more and see more complicated scenarios, for example, here again this slightly more complex one.



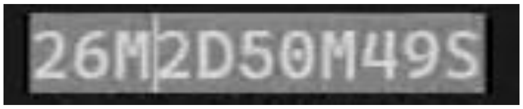
```
D00733:181:CAH6EANXX:8:2210:15698:2137  161   ref|NC_001134|  151641  44   125M        =    152116  600    CTAATAAATCTTCCGGGATTTA
D00733:181:CAH6EANXX:8:2210:16346:2199  97    ref|NC_001142|  418753  44   125M        =    419201  571    GTATAAAGTAGTTATTATGTCA
D00733:181:CAH6EANXX:8:2210:16346:2199  145   ref|NC_001142|  419201  44   113M2I1M9S  =    418753  -571   TGAGTCTCATCACA
D00733:181:CAH6EANXX:8:2210:16918:2242  83    ref|NC_001136|  697562  44   87M         =    697562  -87    CTTTCGCACCACGAATCCATTA
D00733:181:CAH6EANXX:8:2210:16918:2242  163   ref|NC_001136|  697562  44   87M         =    697562  87     CTTTCGCACCACGAATCCATTA
```



113M2I1M9S

Here you see 113 m 2 i 1 m 9 s, ok? So, again, following the same kind of explanation So, 113 m means 113 base matches, then you have two insertions, then again we have one match and nine base soft clips, ok? So, what you see towards the end of this read is that you have this insertion of one match and then nine soft clips, which means this quality of alignment has probably dropped a bit. So, you would have to actually check very carefully the alignment here and probably want to discard this last part, right? So, whether this alignment actually has worked or not, these two insertions may not be real insertions because you have these nine soft clips, which means they are not matching.

So, the program could not find any matches. So, it is removing them, right? So, you would have to be a bit careful when dealing with this fit alignment, ok? So maybe we can search for another example. I will search so that I can show you exactly what this will look like.



26M2D50M49S

Let us search. I think we will see if we can find some, ok? So, here are some nice examples—nice, complicated ones, ok? So, here you see, right, you have 26 m 2 d 50 m 49 s, ok? So, this contains a lot of these different signs, right? So, 26 m means first 26 bases are matching, then you have 2 d, which means 2 bases are deleted, right? So, if these are present in reference but not in the read, then you have 50 m, right?

So, 50 bases matching again, then you have 49 s, 49 bases soft clip, ok? Now, these 49 should ring a bell, right? This is a huge number of bases that have been soft-clipped. So, maybe this read means there is some issue with this mapping here, ok? Similarly, the next one is also quite complex: right 35 base 28 m 2 d 62 m, ok.

So, 35 bases are soft clips, and 28 matches 2 deletions, 62 matches, ok, and so on, right? So you can probably find a lot more examples. So, here is a nice example again: you have 120 m, m, right, 2 d, 1 m, 4 s, right. So, you will see that in most of the cases and situations, you will have these only. So, that is good; you do not have to interpret anything, but the reads that contain insertions or deletions can be identified very quickly using those cigar strings, right?

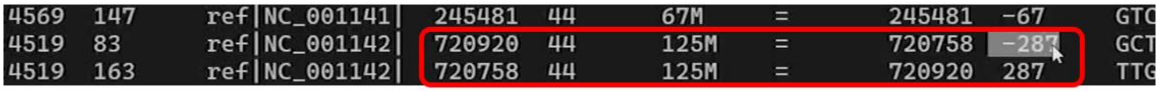
So, if you have cigar strings I or D, you can say, Ok, these are the reads that contain deletion or insertion, right? So, this is quite helpful if you are trying to work with indels only, or maybe you want to filter out indels because you are not interested in indels. In that case, you can use this information. Now, here, I will point you towards some data, right? You can see these two lines, right? These two lines are part of the same fragment, and some of them are given as stars.

So, star means there is no position; there is no sequence; there is no right name. So, it could not map this read to the reference sequence, ok? So, now that this cigar string is clear, the next field is the reference name for the mate. This is the next, right? So, this is said to equal, which means the reference is the same, right?

So, it is mapping to the same chromosome. So, that is why this equal sign is okay. Instead of writing this again, you have the equal sign. Then followed by you have this position PNEXT, right. So, this is comparable to POS, right.

So, you can see these two reads here again; these are paired in data, ok? And you can see these are actually mapping to the same positions, right? So, here is the position of mate 2, ok, of these reads. So, the other mate So, this is the position.

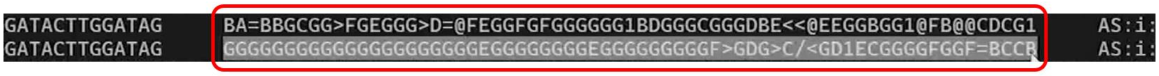
We can take some other examples; for example, we can go down here, right? For example, we can take this one, let us say this is we have 125 m matches, right? So, again, these are equal signs for most of these; right means they are mapping to the same reference sequence, and the mate is mapping to the same reference sequence as the read. Then you have the positions given, right of the mate where this is aligning, and then you have this t lane, ok. So, this is the fragment lane, ok?



```
4569  147      ref|NC_001141|  245481  44      67M      =        245481  -67     GTC
4519  83       ref|NC_001142|  720920  44      125M     =        720758  -287    GCT
4519  163      ref|NC_001142|  720758  44      125M     =        720920  287     TTG
```

Now, for this one, this is negative, which means this is occurring downstream. The mate is occurring upstream of this tree, ok? And for the mate here, right? So, as I said, the data is stored in pairs, right? So, you have read 1, read 2 data points one after another.

So, for the mate, you can see this TLEN is positive, right? So, this one is the one that is occurring upstream, right, or occurring in the leftmost position and as you read here, right, this one is occurring downstream; this is the rightmost position, ok? So, hopefully, these are clear. Now, these two fields are describing the mapping of the mate. The next field is the sequence. You simply get the sequence data here; this is what you see, and then you have the quality score as we go along, right? So, here is the quality score ASCII encoded again, simply taken from the reference, simply taken from the FASTQ file. Okay, alright.



```
GATACTTGGATAG      BA=BBGCGG>FGEGGG>D=@FEGGFGFGGGGGG1BDGGGCGGGDBE<<@EEGGBGG1@FB@@CDCG1      AS:i:
GATACTTGGATAG      GGGGGGGGGGGGGGGGGGGGGEGGGGGGGGGEGGGGGGGGGF>GDG>C/<GD1ECGGGGFGGF=BCCB      AS:i:
```

So, this is actually the end of the compulsory fields, ok? Now, what you have after these are the optional fields, and you can probably see some of them here. If you look carefully, these are actually describing the optional fields: AS, i, etcetera. So, we will now describe what these optional fields actually mean and then we can again look at the data to better understand these fields. So, let us go back to the presentation and discuss these optional fields, ok? So, going to the optional fields, all the optional fields are given in this specific format, ok?

So, we have something called tag; you have to type the value, ok? So, what is a tag? So, this is a two-character string, right, and then type this is a single letter usually, right. This defines the format of the value, right. So, whether the value is an integer or a string, that information will be given in this type of field. So, then there are specific tags that we use to describe the alignment or the mapping, ok?

So, here is the first one. So, notice carefully that this is given as an AS colon. So, that is the tag, right? Then you have the I, which means we can expect an integer in the value, which is given here within the less than or greater than sign N, right? So, here we will get a value. So, A is I, so I means integer. So, we will get an integer value in place of N, ok?

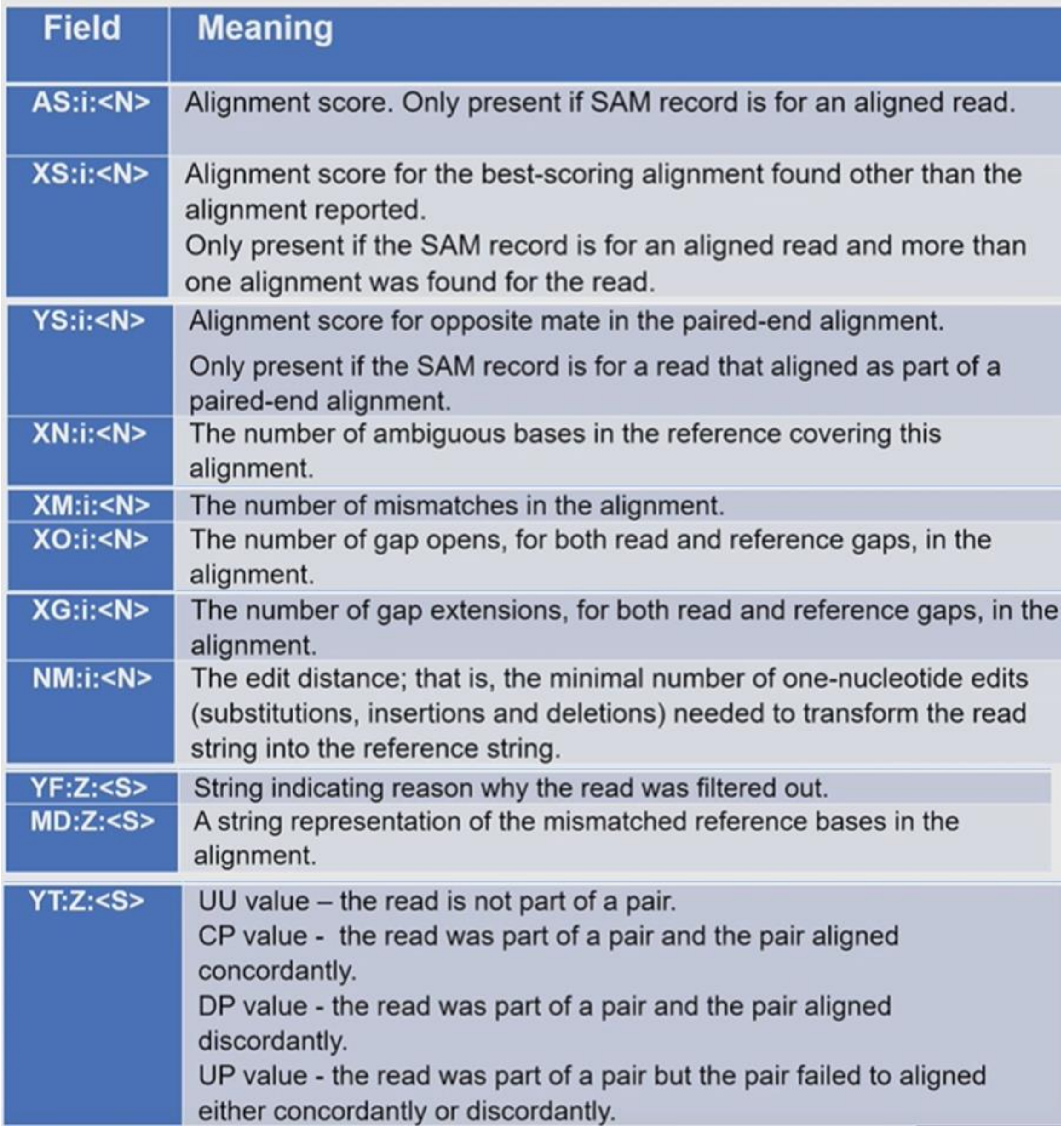
So, this gives us an alignment score, right? So, and this is present if you are working with aligned reads or aligned data, ok? Now, this alignment score is a better score; a higher score means you have better alignment or better matches, right? Then you get something called XS i N, ok?

Again, this is an integer field; right i means integer. So, you get a number. Now, what you get in XS is the alignment score for the base-coding alignment found other than the primary alignment. So, S actually is the reported alignment; right that alignment that is reported is the reference name, the position; this is the primary alignment; right where the alignment has happened, that gives us this alignment score AS. But sometimes reads can map to multiple positions, right? So, that is why you get these alternate alignment scores, or these are the secondary alignments that are reported, ok? And this will be given when you have more than one alignment for the read in the reference sequence, ok?

So, this is something we may have to be careful about. Right when we are processing the SAM file, we may have to carefully check whether there is alternate alignment or whether they are mapping to multiple positions in the genome, because that can also influence some of the downstream analysis. Then you have this YS, right? So, this is again the alignment score for the opposite mate in the paired-end alignment, right? So, if you are working with paired-end data, you will get this information, ok? Then you have something called XN, OK, and the number of ambiguous bases in the reference covering this alignment.

So, again, these are all integer numbers, right? So, if you have an ambiguous basis in the reference that you want to know, right. So, because in those positions we will not be able to call any mutations or mismatches, right? So, this is this just simply gives us the number of ambiguous bases in the reference, ok? Following this, there will be this very important information that actually describes the alignment, whether there are mismatches, etcetera.

Now one thing you should remember is that you may not see all these fields in the data; again, these are optional fields.



Field	Meaning
AS:i:<N>	Alignment score. Only present if SAM record is for an aligned read.
XS:i:<N>	Alignment score for the best-scoring alignment found other than the alignment reported. Only present if the SAM record is for an aligned read and more than one alignment was found for the read.
YS:i:<N>	Alignment score for opposite mate in the paired-end alignment. Only present if the SAM record is for a read that aligned as part of a paired-end alignment.
XN:i:<N>	The number of ambiguous bases in the reference covering this alignment.
XM:i:<N>	The number of mismatches in the alignment.
XO:i:<N>	The number of gap opens, for both read and reference gaps, in the alignment.
XG:i:<N>	The number of gap extensions, for both read and reference gaps, in the alignment.
NM:i:<N>	The edit distance; that is, the minimal number of one-nucleotide edits (substitutions, insertions and deletions) needed to transform the read string into the reference string.
YF:Z:<S>	String indicating reason why the read was filtered out.
MD:Z:<S>	A string representation of the mismatched reference bases in the alignment.
YT:Z:<S>	UU value – the read is not part of a pair. CP value - the read was part of a pair and the pair aligned concordantly. DP value - the read was part of a pair and the pair aligned discordantly. UP value - the read was part of a pair but the pair failed to aligned either concordantly or discordantly.

So, in some cases, you may not see XN. You may not see XN, right? If there is no reported value, any 0 means this may not be reported. These are optional fields for a reason, right? So, you will see maybe some of them and not all, ok? So, the next one is something called XM i N.

Again, it is an integer, and it actually gives us the number of mismatches in the alignment. So, if you compare the read against the reference sequences, how many mismatches do you have in the read? Then you have something called XO and the number of gaps that you get for both read and reference gaps in the alignment, ok? So, this is simply describing the number of indels that you have in the data. And this is the number of gap extensions, XG i N, right?

So, again, this is looking at both indels, right and this is again something I will like. So, by taking XG and XO, you will get information about the indels in the data that you have. So, you can see that cigar will give you an indication of whether you have indels, and this will give you the numbers, right, how many indels you probably have in the data. Then you have something called NMI. So, this is the edit distance. So, that is the minimal number of one nucleotide change, whether it is insertion, deletion, or substitution that you need to do in the read sequence to get to the reference frame, ok?

So, it is kind of like a number of mismatches plus a number of indels that gives you this edit distance, or simply how far the read is from the reference sequence, right? So, in most of these cases, this should be 0 because you do not have any mismatches or indels in the data, ok? Again, we will go back to the same file, and we will carefully look at what these numbers actually look like. Then you have something called YF Z S.

So, if there is some sort of filtering, it might indicate that Y read was filtered out, etcetera. I may not see this right if the read is present. Then you have something called MD Z, ok? So, I should comment on this Z, right? So, you will see Z whenever you see Z in the second type field, right?

So, this means there will be a string in the value, ok? So, this S means string. So, you will see a string, right? So, this will be an alphanumeric, ok? MD Z S will be a string representation of the mismatched reference basis in the alignment. So, what are the changes that you see? Right, that will be given in this MD Z, ok?

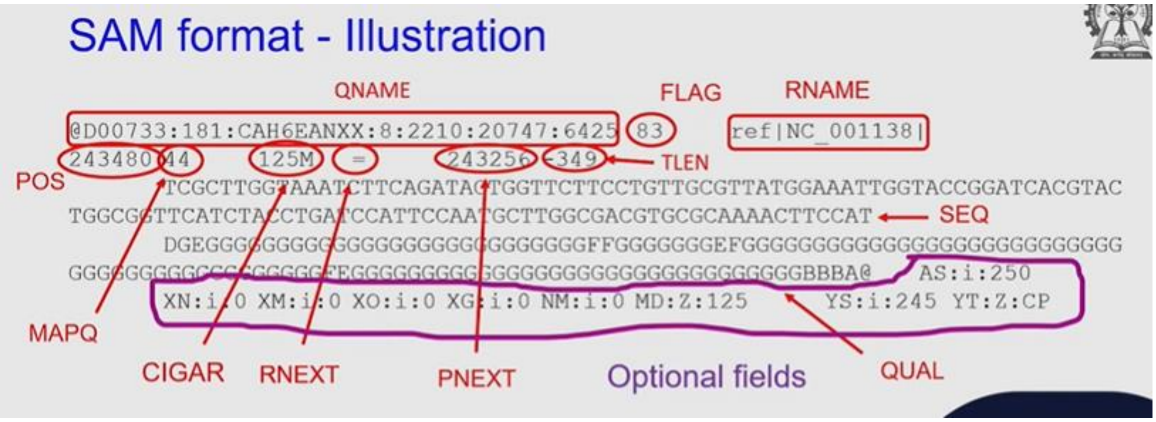
Finally, you have something called the YT Z value, right? So, again, this is a string. You will see some of these options. So, only one of these, right? So, you can have UU that this read is not part of a pair. If you are dealing with single-end data, this will be happening.

If you have a CP value, the read was part of a pair, and then they aligned concordantly. So, this is referring to concordant mapping, right? We describe what concordant mapping is, and most reads will show concordant mapping. If you see YT Z DP, this means this is a discordant mapping, and perhaps you want to look at those data carefully, and if there is an indication of structural variation, gene fusion, etcetera, those will be found with this discordant mapping. Then you have an UP value, right?

So, the read was part of a pair, but the pair failed to align concordantly or discordantly, right? So, this is something that again means you have single mapping, right? So, only one of them aligned. So, they did not map as a pair, alright?

So, just a final illustration, right? So, this is the Q name that you will get. Then you have something called a flag.

This is a flag. This is the RNAME, right? Then you have this POS field. This is the map queue. This is the cigar. Then you have RNEXT. This is P next.



SAM format - Illustration

```
@D00733:181:CAH6EANXX:8:2210:20747:6425 83      ref|NC_001138|
243480 44      125M   =      243256 -349
            TCGCTTGGTAAATCTTCAGATACTGGTTCTTCCTGTTGCGTTATGGAAATTGGTACCGGATCACGTAC
TGGCGGTTCATCTACCTGATCCATTCCAATGCTTGGCGACGTGCGCAAAACTTCCAT
            DGEGGGGGGGGGGGGGGGGGGGGGGGGGGGGGFFGGGGGGGEFGGGGGGGGGGGGGGGGGGGGGGGGGG
GGGGGGGGGGGGGGGGGGFEGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGBBBA@      AS:i:250
        XN:i:0 XM:i:0 XO:i:0 XG:i:0 NM:i:0 MD:Z:125      YS:i:245 YT:Z:CP
```

Then you have the T name, ok? So, these are the compulsory fields. You have sequence and quality, of course. These are also present in the data as part of the compulsory fields. Then, following that, you have these optional fields, ok, and you get these different characteristics, right? So, that describes the alignment, ok? So, what are we going to do? We will go to the file now in the terminal and look at the actual data, the actual output data, and see what these optional fields look like.

So, we have seen up to the compulsory fields. So, we will now check how these optional fields look in the real data, ok? So, let us move here, right?

So, you see this here, okay, and you can see these numbers, right? So, you can see ASI 127. This is the alignment score. There is no excess.

So, there is no alternate alignment form. So, this is a unique mapping. Then you have this XM I, right? So, XN I is 0. There is no ambiguity in the reference. You have XM I 1, which means you have one mismatch. XY 0 XZ XG I 0 means there is no gap, no indel, right in the data.

NMI is the edit distance. So, what will happen is that XM plus XO plus XG—these numbers will give you the NM, ok? Then you have a description of this mismatch that was found, right?

So, if you carefully look at this, So, this is 30T 36, right? So, that means you have this. So, the length is 67 here, right? If you come to the same read data, we just go into this field, which is 67, and that MD Z means 30T 36, ok? So, 30T means you have 30 bases that match exactly.

Then you have a T mutation, right and then you have 36 base matches, ok. Then you have YSI. We have described what YSI is. And then we have this YT Z. This is again quite important. So, this MD Z, this NM I—these are actually the important optional fields that you often look at if you are processing these files ourselves, ok? YT Z CP means you have concordant mapping, ok?

So, we can actually also see some of the reads; maybe you will find this is called a mapping here, for example. So, in here, you can see these two reads, right? So, you have this YT Z DP in this, right? So, they are actually showing this coordinate mapping, ok? So, maybe we can also find out the reason why they are showing this coordinate mapping, whether mapping to two different reference sequences or two different reference chromosomes, right?

So, the chromosomes are reference sequences here, or they are mapping too far away from each other, right? So, here, they are actually mapping to the same chromosome. So, these are the two reads here, right? You can see they are mapping to the same chromosome because of this equal sign, but they are too far from each other.

So, 843. And now the default option is 500. So, the maximum distance that is allowed is 500 in bowtie 2. So, you can, of course, change this using this minus x option when you run Bowtie 2. So, because this distance is too high according to the cutoff or threshold that is set in Bowtie2, we are calling these discordant mappings, ok? So, I think this is clear enough, right? What is the information that we will get in these fields? Some of this information could be very useful. For example, in this read data, you can see you have quite elaborate information and multiple mutations that are present, and you will see this will translate into NM I or XM I equals 6, right?

So, NMI is just a sum of XM I, XO I, and XG I, right? So, XM I 6 means there are 6 match mismatches, ok? And in some cases, you can see these excess fields, right? So, you have this ASI and XSI, and in this case, what you see is that the alignment score and the alternate alignment score are actually the same. And in these cases, you would have to be careful because this means they are mapping to repeat sequences, right?

And the alignment score is exactly identical, right? So, that means they are showing probably exactly the same alignment. It is not like one alignment is better than another; they are probably of similar number of mismatches, etcetera, alright? I think we can now conclude. So, what we have is the reference we have seen. And to conclude, we have actually seen that the sample contains a header section and an alignment section. For each alignment, the sample shows the location of the alignment, the quality of the mapping, as well as the mapping of the mate if you are dealing with paired-end data.

Each alignment line contains 11 mandatory fields, and there are also optional fields. These optional fields are also quite important, right? They give out certain important information about the alignment. And each alignment also reports mismatches in one of the optional fields. So, this is actually important if you are trying to look for genetic variants etcetera. That is it for today. Thank you.