Using the *GenomicFeatures* package

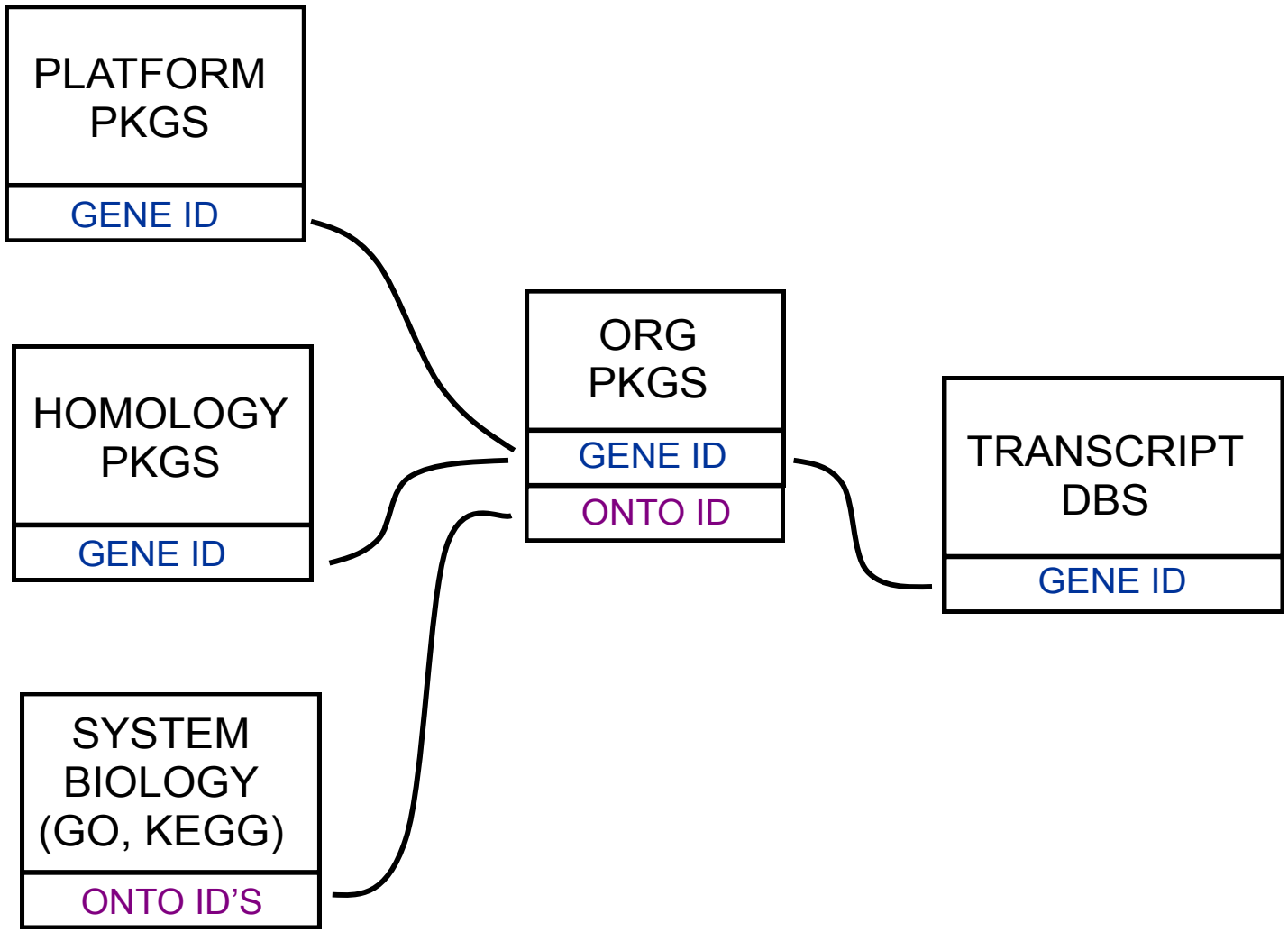
Marc Carlson

Fred Hutchinson Cancer Research Center

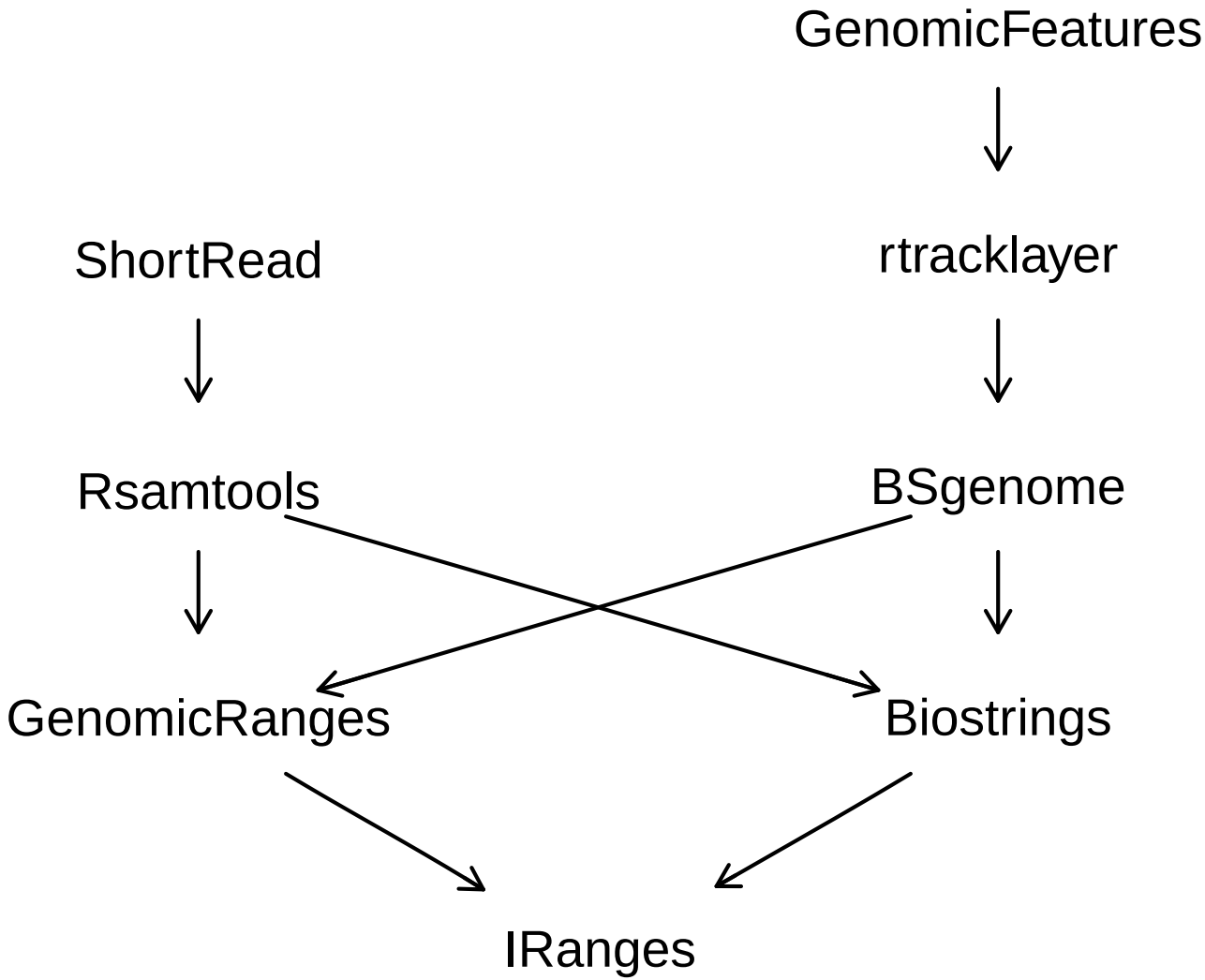
July 30th 2010

Outline

Bioconductor Annotation Packages



Bioconductor Sequence Packages



Bioconductor Sequence Annotations

The GenomicFeatures package

- Transcript information stored in SQLite databases
- These databases attempt to represent the relational nature of splicing data correctly
- Transcript information exposed to user via the *GenomicRanges* infrastructure
- Parsers are available to construct these databases from biomaRt or the UCSC genome browser. Also there is a generic parser for custom jobs.

Outline

GenomicFeatures transcript sources

Constructors

`makeTranscriptDbFromBiomart`, `makeTranscriptDbFromUCSC`

```
> library(GenomicFeatures)
> nrow(supportedUCSCtables())
```

```
[1] 24
```

```
> head(supportedUCSCtables(), 10)
```

	track	subtrack
knownGene	UCSC Genes	<NA>
knownGeneOld3	Old UCSC Genes	<NA>
wgEncodeGencodeManualRel2	Gencode Genes	Genecode Manual
wgEncodeGencodeAutoRel2	Gencode Genes	Genecode Auto
wgEncodeGencodePolyaRel2	Gencode Genes	Genecode PolyA
ccdsGene	Consensus CDS	<NA>
refGene	RefSeq Genes	<NA>
xenoRefGene	Other RefSeq	<NA>
vegaGene	Vega Genes	Vega Protein Genes
vegaPseudoGene	Vega Genes	Vega Pseudogenes

TranscriptDb basics

Making a *TranscriptDb* object

```
> mm9KG <-
+   makeTranscriptDbFromUCSC(genome = "mm9",
+                            tablename = "knownGene")
```

Saving and Loading

```
> saveFeatures(mm9KG, file="mm9KG.sqlite")
```

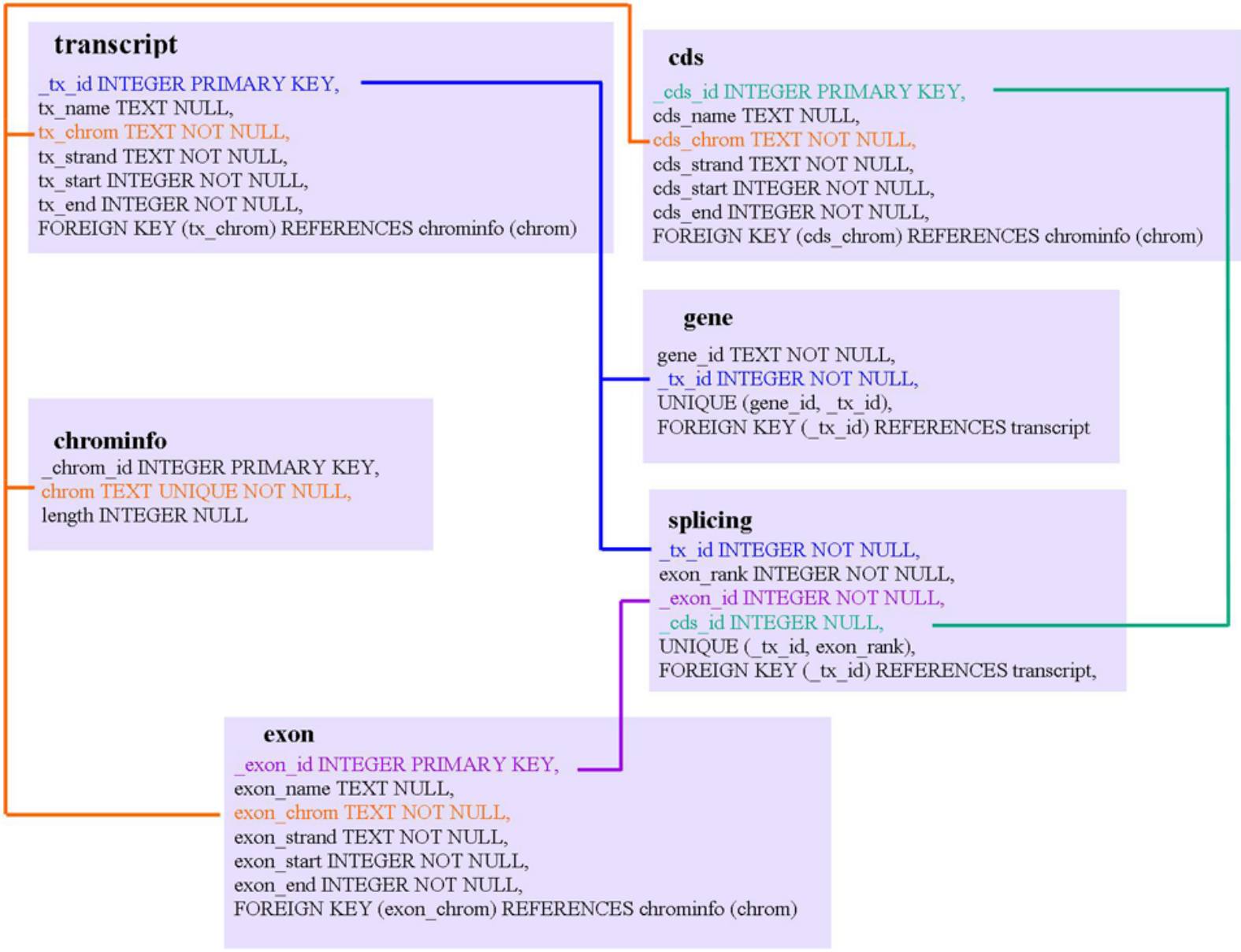
```
> mm9KGChr9 <-
+   loadFeatures(system.file("extdata", "mm9KG.sqlite",
+                            package = "HTSandGeneCentricLabs"))
```

TranscriptDb class

```
> mm9KGChr9

TranscriptDb object:
| Db type: TranscriptDb
| Data source: UCSC
| Genome: mm9
| UCSC Table: knownGene
| Type of Gene ID: Entrez Gene ID
| Full dataset: no
| transcript_nrow: 2910
| exon_nrow: 14110
| cds_nrow: 12270
| Db created by: GenomicFeatures package from Bioconductor
| Creation time: 2010-05-13 17:02:30 -0700 (Thu, 13 May 2010)
| GenomicFeatures version at creation time: 1.1.1
| RSQLite version at creation time: 0.8-3
```

TranscriptDb schema



Ungrouped transcript-related information

Extractors

`transcripts`, `exons`, `cds`

```
> tx <- transcripts(mm9KGChr9)
> length(tx)
```

```
[1] 2910
```

```
> head(tx, 5)
```

```
GRanges with 5 ranges and 2 elementMetadata values
    seqnames              ranges strand |     tx_id     tx_name
       <Rle>           <IRanges>  <Rle> | <integer> <character>
[1]     chr9 [3215314, 3215339]      + |     24312  uc009oas.1
[2]     chr9 [3335231, 3385846]      + |     24315  uc009oat.1
[3]     chr9 [3335473, 3343608]      + |     24313  uc009oau.1
[4]     chr9 [3335473, 3380423]      + |     24314  uc009oav.1
[5]     chr9 [3335478, 3385846]      + |     24316  uc009oaw.1

seqlengths
      chr1       chr2 ...  chrX_random  chrY_random
 197195432  181748087 ...      1785075     58682461
```

Grouped transcript-related information

Extractors

`transcriptsBy`, `exonsBy`, `cdsBy`, `intronsByTranscript`, `fiveUTRsByTranscript`, `threeUTRsByTranscript`

```
> txExons <- exonsBy(mm9KGChr9)
> txIntrons <- intronsByTranscript(mm9KGChr9)
> txExons[6]
```

```
GRangesList of length 1
$24313
GRanges with 3 ranges and 3 elementMetadata values
    seqnames              ranges strand |   exon_id   exon_name
       <Rle>           <IRanges>  <Rle> | <integer> <character>
[1]     chr9 [3335473, 3335594]      + |    117005          NA
[2]     chr9 [3338456, 3338591]      + |    117006          NA
[3]     chr9 [3343015, 3343608]      + |    117007          NA
    exon_rank
    <integer>
[1]         1
[2]         2
[3]         3
```

Standard GRanges accessors can be used

Accessors

`names`, `length`, `elementMetaData`, `width`, `ranges`,`start`,`end`

Examples:

```
> head(start(tx))

[1] 3215314 3335231 3335473 3335473 3335478 3379207

> head(ranges(txExons), n=1)

CompressedIRangesList of length 1
$`24308`
IRanges of length 1
      start     end width
[1] 3186316 3186344    29

> head(elementMetadata(tx), n=2)

DataFrame with 2 rows and 2 columns
      tx_id     tx_name
  <integer> <character>
1     24312  uc009oas.1
```

Overlapping with transcripts

Methods

`findOverlaps`, `countOverlaps`, `match`, `%in%`, `subsetByOverlaps`

Usage and help:

```
> findOverlaps(query, subject, maxgap = 0L, minoverlap = 1L,
+              type = c("any", "start", "end"),
+              select = c("all", "first"))
> help("findOverlaps,GRanges,GRangesList-method")
```

Example

```
> grngs <- GRanges("chr9", gaps(ranges(txExons[[6]])), "+")
> countOverlaps(grngs, tx)
```

```
[1] 4 4
```

```
> rbind(countOverlaps(grngs, txExons), countOverlaps(grngs, txIntrons))
```

```
     [,1] [,2]
[1,]    1    0
```

GenomicFeatures exercise 1

Load the transciptDB object above and then gather the annotations for transcripts as grouped by their genes.

GenomicFeatures exercise 1 solution

```
> library(GenomicFeatures)
> mm9KGChr9 <-
+   loadFeatures(system.file("extdata", "mm9KG.sqlite",
+                            package = "HTSandGeneCentricLabs"))
> myTranscripts <- transcriptsBy(mm9KGChr9, by="gene")
```

Outline

A simple RNA-seq example

1st lets just load some data

`readBamGappedAlignments`

```
> library(Rsamtools)
> testFile <- system.file("bam", "isowt5_13e.bam",
+                         package="leeBamViews")
> aligns <- readBamGappedAlignments(testFile)
> head(aligns)
```

```
GappedAlignments of length 6
      rname strand cigar qwidth  start    end width ngap
[1] Scchr13      -   36M     36 799975 800010    36    0
[2] Scchr13      -   36M     36 799977 800012    36    0
[3] Scchr13      -   36M     36 799979 800014    36    0
[4] Scchr13      -   36M     36 799980 800015    36    0
[5] Scchr13      +   36M     36 799986 800021    36    0
[6] Scchr13      +   36M     36 799986 800021    36    0
```

RNA-seq

Retrieve corresponding Annotations

makeTranscriptDbFromUCSC

```
> library(GenomicFeatures)
> txdb <- makeTranscriptDbFromUCSC(genome="sacCer2",
+                                  tablename="sgdGene")
```

extract exons by transcripts

exonsBy

```
> exonRanges <- exonsBy(txdb, "tx")
> length(exonRanges)
```

```
[1] 6717
```

```
> exonRanges[1]
```

```
GRangesList of length 1
$1
GRanges with 1 range and 3 elementMetadata values
    seqnames            ranges strand |   exon_id   exon_name
       <Rle>         <IRanges>  <Rle> | <integer> <character>
[1]     chrI [130802, 131986]      + |         1          NA
    exon_rank
```

RNA-seq

Correct for chromosome naming

levels

```
> levels(rname(aligns))

 [1] "Scchr01" "Scchr02" "Scchr03" "Scchr04" "Scchr05" "Scchr06"
 [7] "Scchr07" "Scchr08" "Scchr09" "Scchr10" "Scchr11" "Scchr12"
[13] "Scchr13" "Scchr14" "Scchr15" "Scchr16" "Scmito"

> levels(rname(aligns)) <-
+   c(paste("chr", as.roman(1:16), sep=""), "chrM")
```

GenomicFeatures exercise 2

Now that we have the data and annotations stored in the appropriate objects, use the appropriate method to count the overlaps between them.

RNA-seq: GenomicFeatures exercise 2 solution

Use countOverlaps to count Reads

`countOverlaps`

```
> txdb <- makeTranscriptDbFromUCSC(genome="sacCer2",
+                                  tablename="sgdGene")
> exonRanges <- exonsBy(txdb, "tx")
> library("Rsamtools")
> testFile <- system.file("bam", "isowt5_13e.bam",
+                         package="leeBamViews")
> aligns <- readBamGappedAlignments(testFile)
> levels(rname(aligns))
> levels(rname(aligns)) <-
+   c(paste("chr", as.roman(1:16), sep=""), "chrM")
> counts <- countOverlaps(exonRanges, aligns)
```

RNA-seq

Can then calculate RPKM

```
> numBases <- sum(width(exonRanges))
> geneLengthsInKB <- numBases / 1000
> millionsMapped <- sum(counts) / 1000000
> rpm <- counts / millionsMapped
> rpkm <- rpm / geneLengthsInKB
```

RNA-seq

Results can be sorted

```
> sortedRPKM <- sort(rpkm)
> highScoreGenes <- tail(sortedRPKM)
```

GenomicFeatures exercise 3

Now that we have some of the things that were measured the most, we want to know what they are. We don't want to use the transcript ID's though because they are not guaranteed to be meaningful outside of this database. What we want instead are the gene IDs.
So how can we use GenomicFeatures to extract the gene IDs matched to the transcript IDs?

RNA-seq: GenomicFeatures exercise 3 solution

```
> ##All the RPKM stuff since count and then:
> txs <- transcripts(txdb,
+           vals=list(tx_id=names(highScoreGenes)),
+           columns=c("tx_id","gene_id"))
> systNames <- as.vector(
+                 unlist(elementMetadata(txs)["gene_id"]))
```

RNA-seq

use other annot's to learn more

```
> library(org.Sc.sgd.db)
> toTable(org.Sc.sgdGENENAME[systNames])
```

```
  systematic_name gene_name
1         YMR295C       IBI2
2         YMR297W       PRC1
3         YMR305C      SCW10
4         YMR307W       GAS1
```

(combine into sets as needed)

```
> deSeqframe <- data.frame(counts, counts2)
```