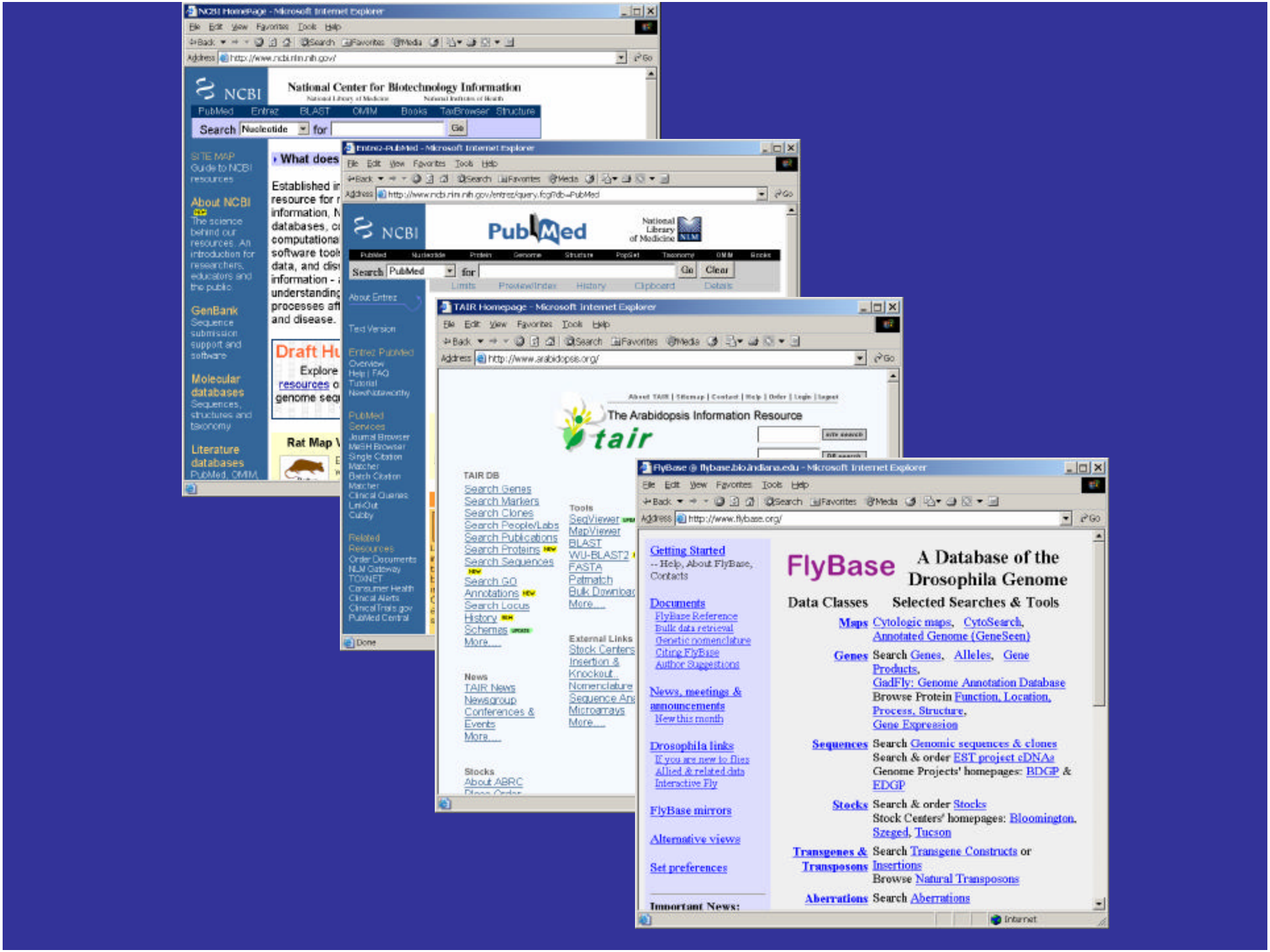
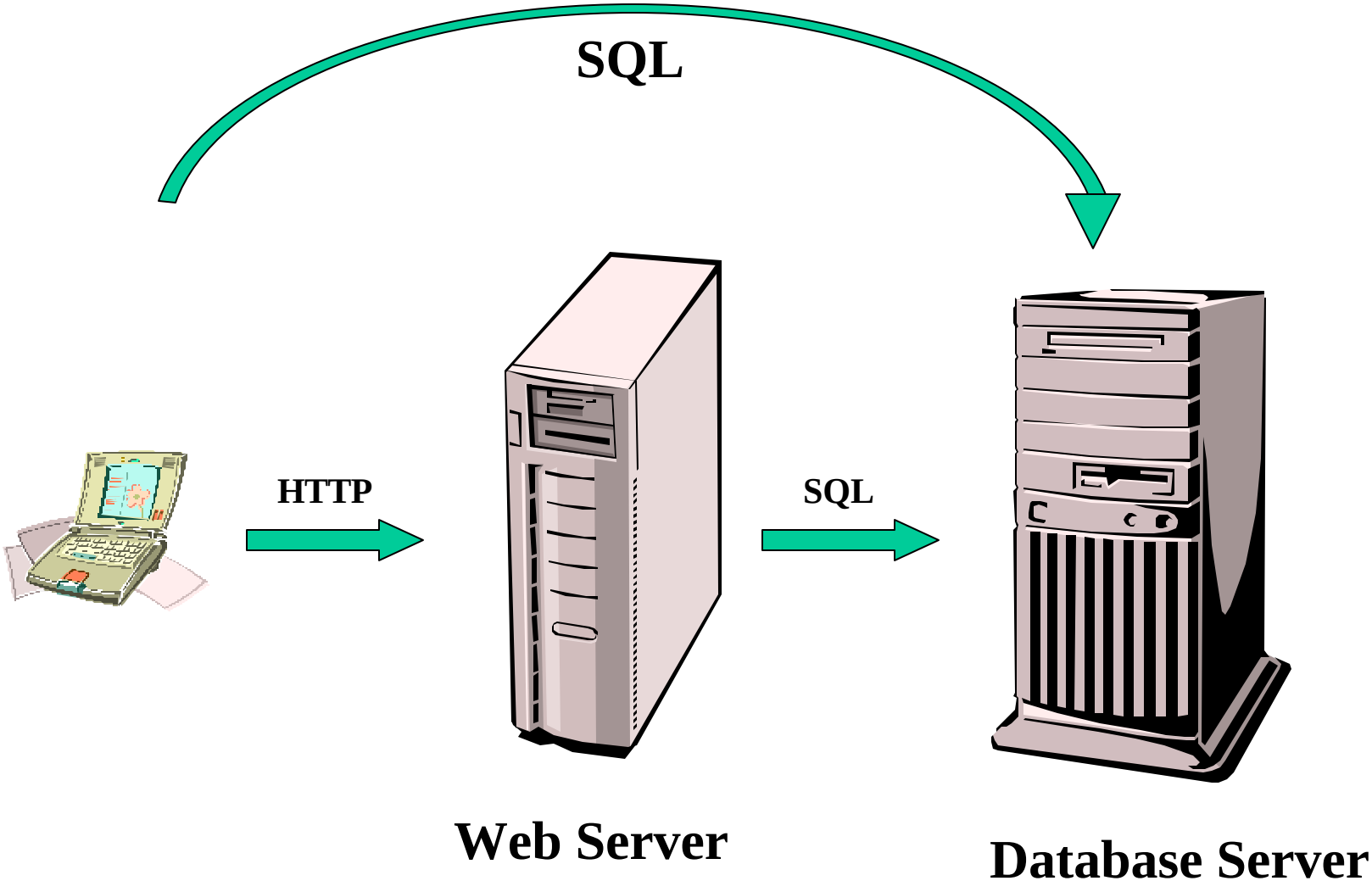
Module 2

Genomic Databases

1. **Data**
   - **Public databases: Genbank, Refseq, ...**
   - **Experimental data**
2. **Data management software**
   - **MS SQL Server**
   - **Designing your own experimental database**
3. **Data processing**
   - **Introduction to PERL and BioPERL**


NCBI HomePage - Microsoft Internet Explorer
National Center for Biotechnology Information
NCBI
PubMed Entrez BLAST OMIM Books TaxBrowser Structure
Search Nucleotide for Go
SITE MAP
Guide to NCBI resources
About NCBI
The science behind our resources. An introduction for researchers, educators and the public
GenBank
Sequence submission support and software
Molecular databases
Sequences, structures and taxonomy
Literature databases
PubMed, OMIM
What does
Draft Hu
Explore
resources o
genome seq
Rat Map
Entrez-PubMed - Microsoft Internet Explorer
http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=PubMed
NCBI
PubMed
National Library of Medicine
Search PubMed for Go Clear
Limits Preview/Index History Clipboard Details
About Entrez
Text Version
Entrez PubMed
Overview
Help | FAQ
Tutorial
New/Noteworthy
PubMed Services
Journal Browser
MeSH Browser
Single Citation Matcher
Batch Citation Matcher
Clinical Queries
LinkOut
Cubby
Related Resources
Order Documents
NLM Gateway
TOXNET
Consumer Health
Clinical Alerts
ClinicalTrials.gov
PubMed Central
Done
TAIR Homepage - Microsoft Internet Explorer
http://www.arabidopsis.org/
The Arabidopsis Information Resource
tair
TAIR DB
Search Genes
Search Markers
Search Clones
Search People/Labs
Search Publications
Search Proteins
Search Sequences
Search GO Annotations
Search Locus History
Schemes
More...
News
TAIR News
Newsgroup
Conferences & Events
More...
Stocks
About ABRC
Tools
SeqViewer
MapViewer
BLAST
WU-BLAST2
FASTA
Patmatch
Bulk Downloa
More...
External Links
Stock Centers
Insertion & Knockout
Nomenclature
Sequence An
Microarrays
More...
FlyBase @ flybase.bio.indiana.edu - Microsoft Internet Explorer
http://www.flybase.org/
Getting Started
-- Help, About FlyBase, Contacts
Documents
FlyBase Reference
Bulk data retrieval
Genetic nomenclature
Citing FlyBase
Author Suggestions
News, meetings & announcements
New this month
Drosophila links
If you are new to flies
Allied & related data
Interactive Fly
FlyBase mirrors
Alternative views
Set preferences
Important News:
FlyBase
A Database of the Drosophila Genome
Data Classes
Selected Searches & Tools
Maps Cytologic maps, CytoSearch, Annotated Genome (GeneSeen)
Genes Search Genes, Alleles, Gene Products, GadFly: Genome Annotation Database
Browse Protein Function, Location, Process, Structure, Gene Expression
Sequences Search Genomic sequences & clones
Search & order EST project cDNAs
Genome Projects' homepages: BDGP & EDGP
Stocks Search & order Stocks
Stock Centers' homepages: Bloomington, Szeged, Tucson
Transgenes & Transposons Search Transgene Constructs or Insertions
Browse Natural Transposons
Aberrations Search Aberrations
Internet


SQL
HTTP
SQL
Web Server
Database Server

1. more powerful data mining

2. to integrate the public database with your experimental data

NCBI Sequence Databases

Archival database (GenBank, GenPept)

vs

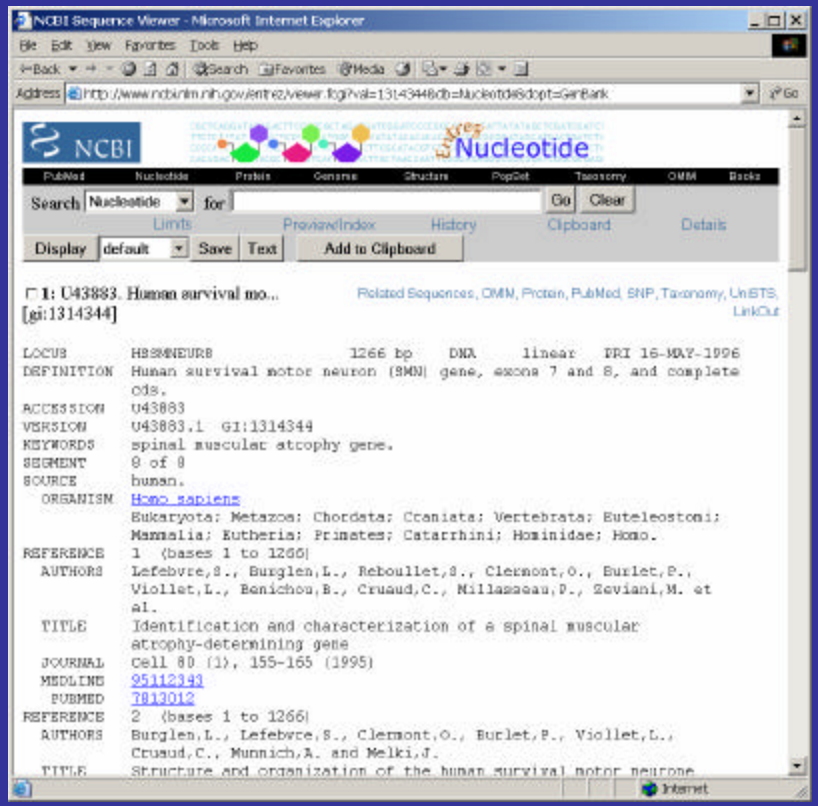
Computer algorithm generated database (Unigene)

vs

Manually curated database (RefSeq, Locuslink ...)

The NCBI Data Model

Genbank- A DNA centered database



Identifier:

1. **LOCUS (obsolete)**
2. **Accession (version)**
3. **GI**

1: U43883. Human survival mo... [gi:1314344]

Related Sequences, OMIM, Protein, PubMed, SNP, Taxonomy, UniSTS, LinkOut

```
LOCUS       HSSMNEUR8                1266 bp    DNA     linear   PRI 16-MAY-1996
DEFINITION  Human survival motor neuron (SMN) gene, exons 7 and 8, and complete
            cds.
ACCESSION   U43883
VERSION     U43883.1  GI:1314344
KEYWORDS    spinal muscular atrophy gene.
SEGMENT     8 of 8
SOURCE      human.
  ORGANISM  Homo sapiens
            Eukaryota; Metazoa; Chordata; Craniata; Vertebrata; Euteleostomi;
            Mammalia; Eutheria; Primates; Catarrhini; Hominidae; Homo.
REFERENCE   1  (bases 1 to 1266)
```

Features

```
mRNA            join(U43876.1:575..688,U43877.1:104..175,
                U43878.1:118..237,U43879.1:84..284,U43880.1:69..221,
                U43881.1:103..198,U43882.1:53..163,209..262,707..>1266)
                /gene="SMN"
                /product="survival motor neuron"
                /note="spinal muscular atrophy gene; cDNA sequence found
                in GenBank Accession Number U18423"
CDS             join(U43876.1:608..688,U43877.1:104..175,
                U43878.1:118..237,U43879.1:84..284,U43880.1:69..221,
                U43881.1:103..198,U43882.1:53..163,209..259)
                /gene="SMN"
                /note="spinal muscular atrophy gene"
                /codon_start=1
                /product="survival motor neuron"
                /protein_id="AAC50473.1"
                /db_xref="GI:1314346"
                /translation="MAMSSGGSGGGVPEQEDSVLFRRGTGQSDDSDIWDDTALIKAYD
                KAVASFKHALKNGDICETSGKPKTTPKRKPAKKNKSQKKNTAASLQQWKVGDKCSAIW
                SEDGCIYPATIASIDFKRETCVVVYTGYGNREEQNLSDLLSPICEVANNIEQNAQENE
                NESQVSTDESENSRSPGNKSDNIKPKSAPWNSFLPPPPPMPGPRLGPGKPGLKFNGPP
                PPPPPPPPHLLSCWLPPFPSGPPIIPPPPPICPDSLDDADALGSMLISWYMSGYHTGY
                YMGFRQNQKEGRCSHSLN"
```

GenPept- A protein centered database

☐ **1:** AAC50473. survival motor ne...[gi:1314346] BLink, Nucleotide, OMIM, Related Sequences, PubMed, SNP, Taxon

```
LOCUS       AAC50473                294 aa            linear   PRI 16-MAY-1996
DEFINITION  survival motor neuron.
ACCESSION   AAC50473
PID         g1314346
VERSION     AAC50473.1  GI:1314346
DBSOURCE    locus HSSMNEUR1 accession U43876.1
            locus HSSMNEUR2 accession U43877.1
            locus HSSMNEUR3 accession U43878.1
            locus HSSMNEUR4 accession U43879.1
            locus HSSMNEUR5 accession U43880.1
            locus HSSMNEUR6 accession U43881.1
            locus HSSMNEUR7 accession U43882.1
            locus HSSMNEUR8 accession U43883.1
KEYWORDS    .
SOURCE      human.
  ORGANISM  Homo sapiens
            Eukaryota; Metazoa; Chordata; Craniata; Vertebrata; Euteleostomi;
            Mammalia; Eutheria; Primates; Catarrhini; Hominidae; Homo.
REFERENCE   1  (residues 1 to 294)
  AUTHORS   Burglen,L., Lefebvre,S., Clermont,O., Burlet,P., Viollet,L.,
            Cruaud,C., Munnich,A. and Melki,J.
  TITLE     Structure and organization of the human survival motor neurone
            (SMN) gene
  JOURNAL   Genomics 32 (3), 479-482 (1996)
  MEDLINE   96435930
   PUBMED   8838816
REFERENCE   2  (residues 1 to 294)
  AUTHORS   Burglen,L.
```

FTP sites:

GenBank: ftp://ftp.ncbi.nih.gov/genbank/

GenPept: ftp://ftp.ncifcrf.gov/pub/genpept/

Problems with Genbank and Genpept

- It does not distinguish the sequence categories.

- Lot of redundancy.
  - Same gene could be deposited into the database many times with different names

  - Different version of the same gene could be submitted many times with different accession number.

- The features of genbank record could be chaotic.

NCBI Sequence Databases

Archival database (GenBank, GenPept)

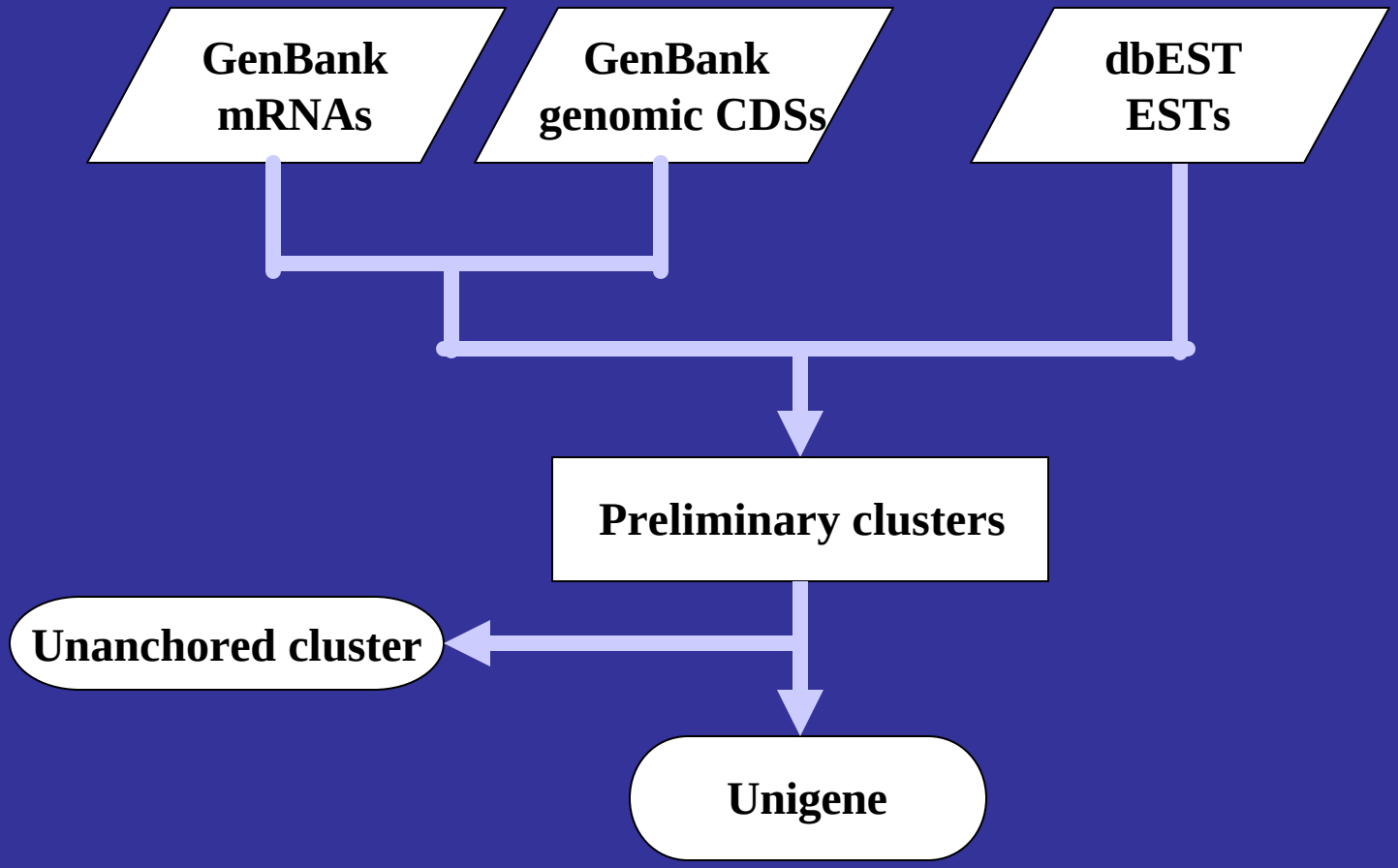
vs

Computer algorithm generated database (Unigene)

vs

Curated database (RefSeq, Locuslink ...)

UniGene
a non-redundant set of gene-oriented clusters



Unigene identifier

Hs for human
Mm for mouse
Rn for rat
Bt for cow
Dr for zebrafish
Dm for fruitfly
Aga for mosquito
Xl for frog
At for cress
Hv for barley
Os for rice
Ta for wheats
Zm for maize

Examples:

Mm.1591

Hs.102456

At.138

NCBI Sequence Databases

Archival database (GenBank, GenPept)

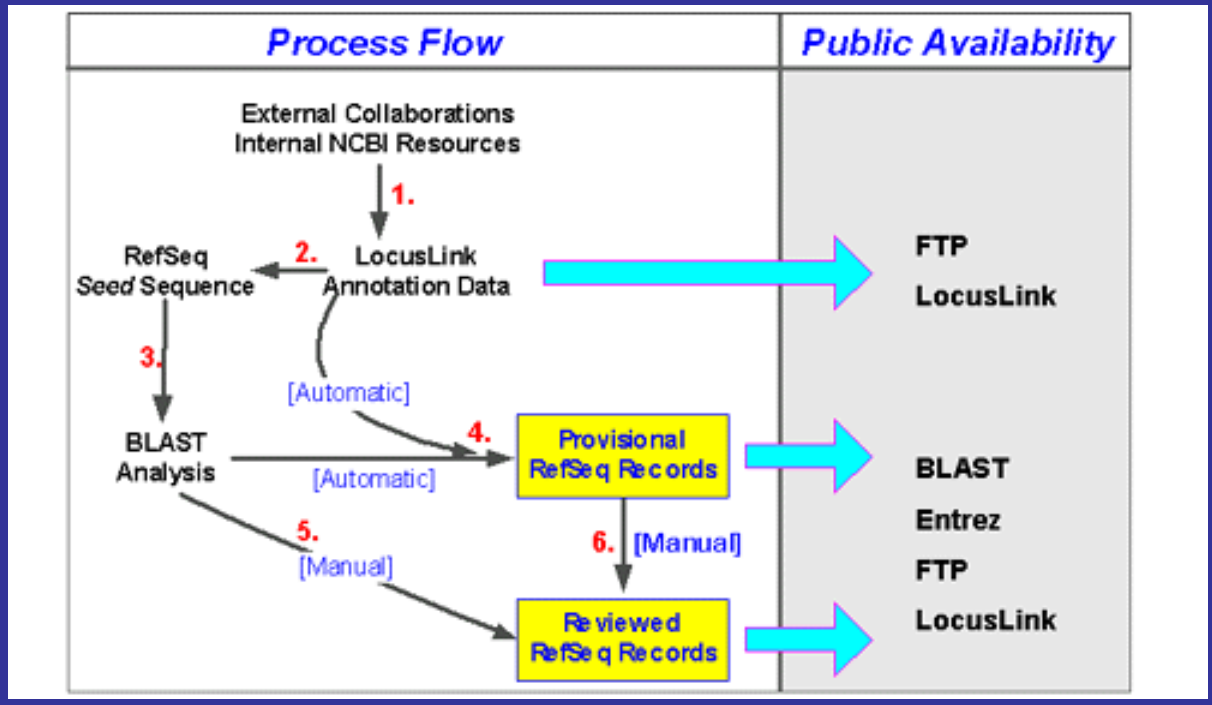
vs

Computer generated database (Unigene)

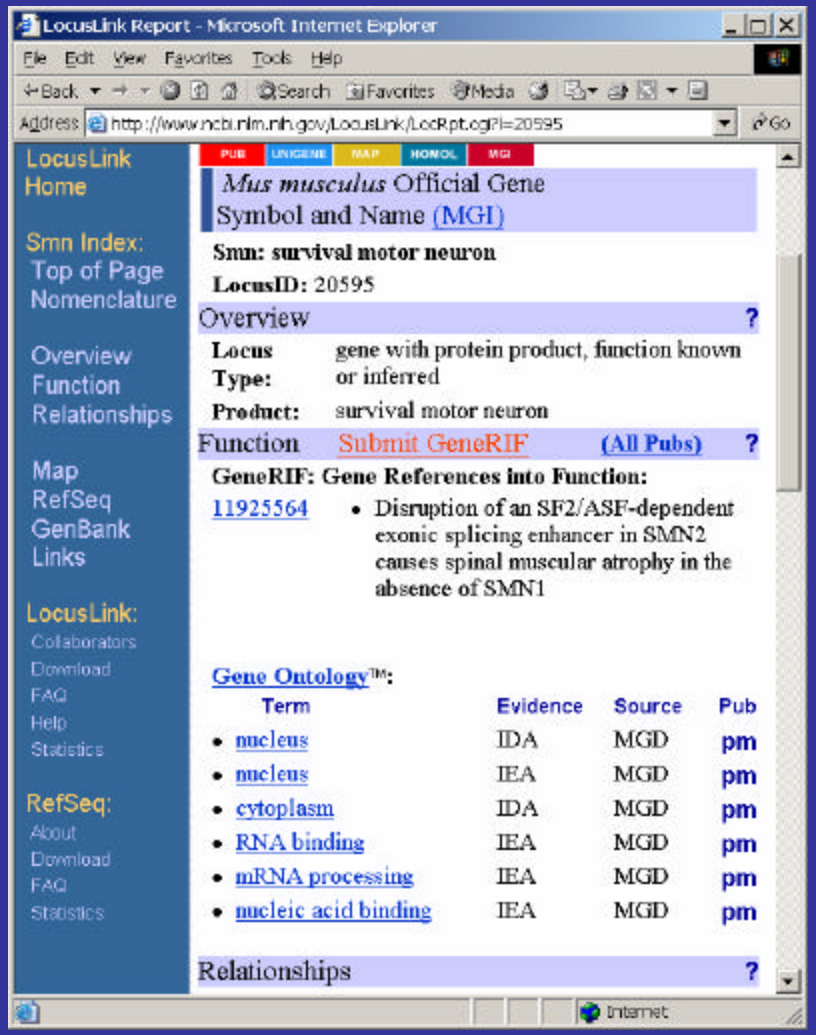
vs

Curated database (RefSeq, Locuslink ...)

RefSeq and LocusLink
NCBI Reference Sequence & Unified Data Access



HTTP



FTP

>>20595
LOCUSID: 20595
LOCUS_CONFIRMED: yes
LOCUS_TYPE: gene with protein product, function known or inferred
ORGANISM: Mus musculus
STATUS: PROVISIONAL
NM: NM_011420|6755579|na
NP: NP_035550|6755580
CDD: Tudor domain|TUDOR|130|na|5.504980e+01
PRODUCT: survival motor neuron
ASSEMBLY: U77714
ACCNUM: AF131205|5932000|129/Sv
TYPE: g
PROT: AAD56757|5932001
ACCNUM: AF240501|8895972|C57BL/6J
TYPE: g
ACCNUM: AF240503|8895974|C57BL/6J
TYPE: g
PROT: AAF81197|8895975
ACCNUM: AF240505|8895978|C57BL/6J
TYPE: g
PROT: AAF81199|8895979
ACCNUM: U92641|4335791|129/SvJ
TYPE: g
PROT: AAD17457|4335792
ACCNUM: U63294|1857113|na
TYPE: m
PROT: AAC53057|1857114
ACCNUM: U77714|1946332|na
TYPE: m
PROT: AAC53144|1946333
ACCNUM: Y12835|3114878|BALB/c
TYPE: m
PROT: CAA73356|3114879
OFFICIAL_SYMBOL: Smn
OFFICIAL_GENE_NAME: survival motor neuron
PREFERRED_PRODUCT: survival motor neuron
CHR: 13

Refseq Accession Numbers:

NT_123456 constructed genomic contigs

NM_123456 mRNAs

NP_123456 proteins

NC_123456 chromosomes

Other Sequence Databases:

Genomic DNA: **Ensembl Genome annotation database**

(http://www.ensembl.org, HTTP, FTP, MySQL interface)

cDNA databases: **RIKEN FANTOM DB**

(http://genome.gsc.riken.go.jp, HTTP FTP interface)

NIH Mammalian Gene Collection (MGC)

(http://mgc.nci.nih.gov, HTTP FTP interface)

GO

Gene Ontology

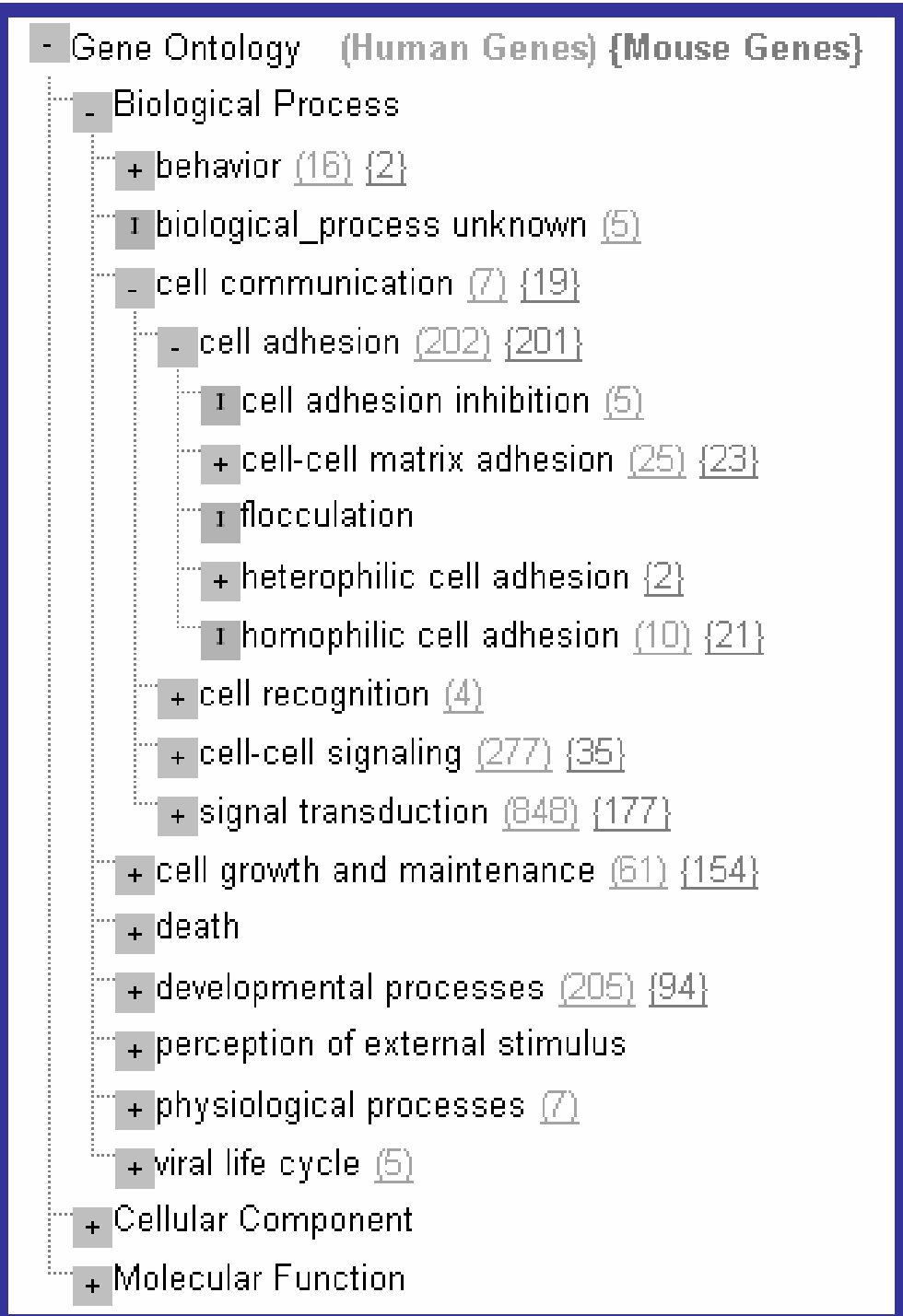
1. **Molecular Function**
2. **Biological Process**
3. **Cellular Component**

brca1

http://www.geneontology.org

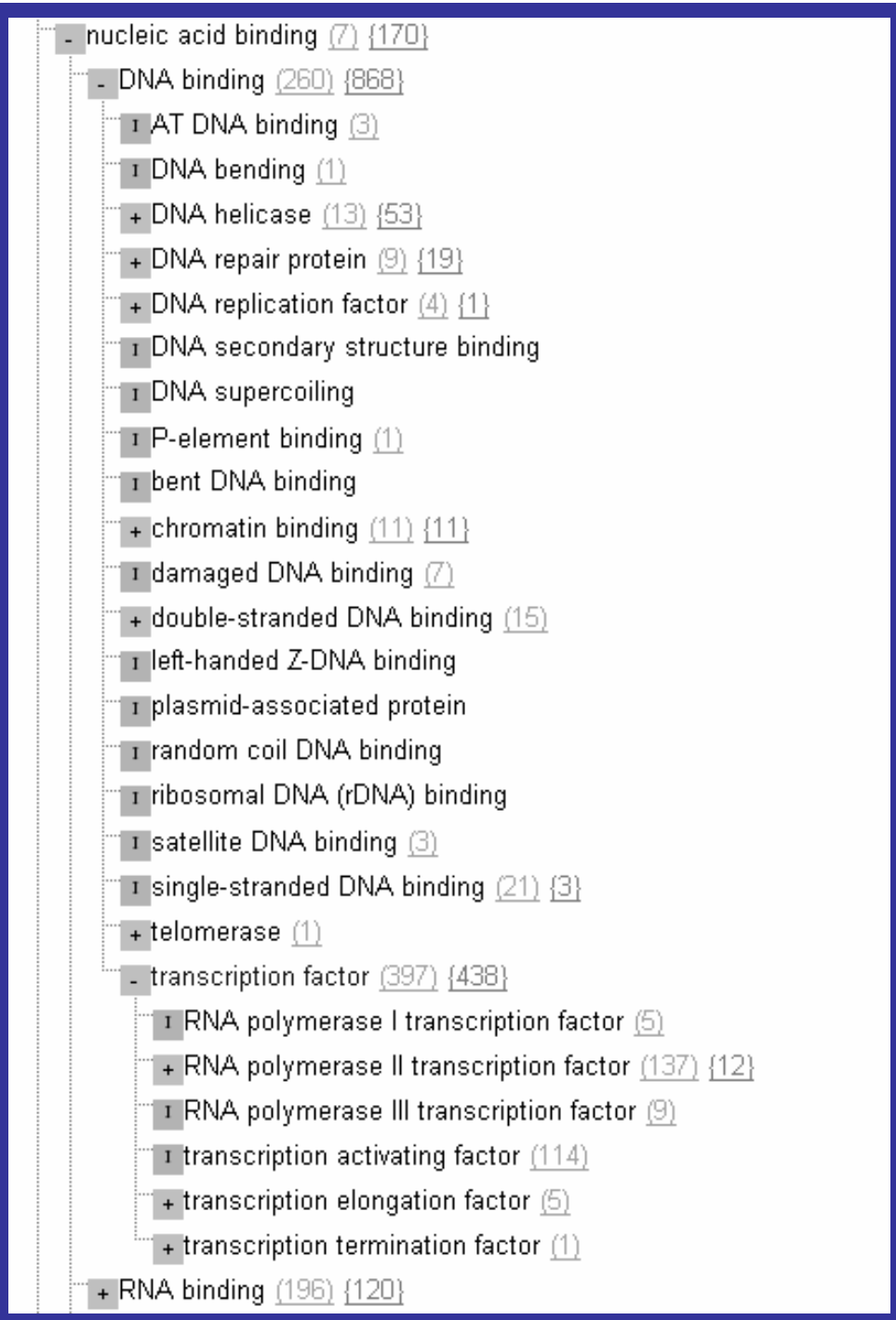
GO Example 1:

Biological Process



GO Example 2:

Molecular Function



Gene Ontology Annotation

Smn: survival motor neuron
LocusID: 20595

Gene Ontology™:

Term	Evidence	Source	Pub
• nucleus	IDA	MGD	pm
• nucleus	IEA	MGD	pm
• cytoplasm	IDA	MGD	pm
• RNA binding	IEA	MGD	pm
• mRNA processing	IEA	MGD	pm
• nucleic acid binding	IEA	MGD	pm

The Pfam and SMART database of protein sequence profile

- PFAM 3849 protein families (v. 7.3)
- SMART 500 extensively annotated domain families

Pfam Examples:

Name	acc number	#seed	#full	av. len	av. % id	structure	Description
GP120	PF00516	24	27468	138 aa	54%	1gc1	Envelope glycoprotein GP120
zf-C2H2	PF00096	197	14973	23 aa	34%	1zaa	Zinc finger, C2H2 type
LRR	PF00560	3732	12110	23 aa	28%	1bnh	Leucine Rich Repeat
rvt	PF00078	178	11477	156 aa	72%	1hmv	Reverse transcriptase (RNA-dependent DNA polymerase)
cytochrome_b_N	PF00033	9	10802	151 aa	68%	3bcc	Cytochrome b(N-terminal)/b6/petB
rvp	PF00077	53	10526	95 aa	87%	1ida	Retroviral aspartyl protease
WD40	PF00400	1930	9117	37 aa	20%	1gp2	WD domain, G-beta repeat
ig	PF00047	113	8581	63 aa	19%	8fab	Immunoglobulin domain
ank	PF00023	1219	6452	31 aa	26%	1awc	Ankyrin repeat
COX1	PF00115	24	6421	228 aa	48%	1occ	Cytochrome C and Quinol oxidase polypeptide I

PDB

3-D biological macromolecular structure data

http://www.rcsb.org

Species Specific Databases

- Arabidopsis – TAIR
- Yeast – SGD
- Fly – FLYBASE
- Worm – WORMBASE
- Mouse – MGD

Using the public database:

1. Data mining through the web

NCBI Entrez

2. Download the database from the FTP server

FANTOM db from RIKEN

Common formats for data download:

Fasta

GenBank

XML

Tab-delimited text format

Relational Database and RDBMS

Some Concepts:

1. **Relational Database**
    - Spreadsheet vs. Relational Database
2. **Relational Database Management System**
    - Oracle
    - Microsoft SQL Server
    - MySQL
3. **SQL - Structured Query Language**
    - Database query language

From Excel spreadsheet to relational database

LocusLink ID	Symbol	Full Name	Species	Accession
1	A1BG	alpha-1-B glycoprotein	Human	AA484435, W25099, AI022193,
2	A2M	alpha-2-macroglobulin	Human	X68728, T80683
3	A2MP	alpha-2-macroglobulin pseudogene	Human	NM_000014
9	NAT1	N-acetyltransferase 1	Human	M11313, AC01110, NM_011100
10	NAT2	N-acetyltransferase 2	Human	AB3432424, AC342308, U34990
12	SERPINA3	serine (or cysteine) proteinase inhibitor, clade	Human	AC93749, BC938749, X430804
13	AADAC	arylacetamide deacetylase (esterase)	Human	U9037459, NM49859
14	AAMP	angio-associated, migratory cell protein	Human	AC043985, X9375904, NM43435
15	AANAT	arylalkylamine N-acetyltransferase	Human	UX34058, PC340854
16	AARS	alanyl-tRNA synthetase	Human	NM_4305003

A1BG a2m

From Excel spreadsheet to relational database

LocusLink ID	Symbol	Full Name	Species	
1	A1BG	alpha-1-B glycoprotein	Human	AA484435
1	A1BG	alpha-1-B glycoprotein	Human	W25099
1	A1BG	alpha-1-B glycoprotein	Human	AI022193
1	A1BG	alpha-1-B glycoprotein	Human	T80683
2	A2M	alpha-2-macroglobulin	Human	X68728
2	A2M	alpha-2-macroglobulin	Human	NM_00001
2	A2M	alpha-2-macroglobulin	Human	M36501
2	A2M	alpha-2-macroglobulin	Human	M11313
3	A2MP	alpha-2-macroglobulin pseudogene	Human	M24415
9	NAT1	N-acetyltransferase 1	Human	NM_00066
9	NAT1	N-acetyltransferase 1	Human	AF071552
9	NAT1	N-acetyltransferase 1	Human	U80835

A1BG a2m

From Excel spreadsheet to relational database

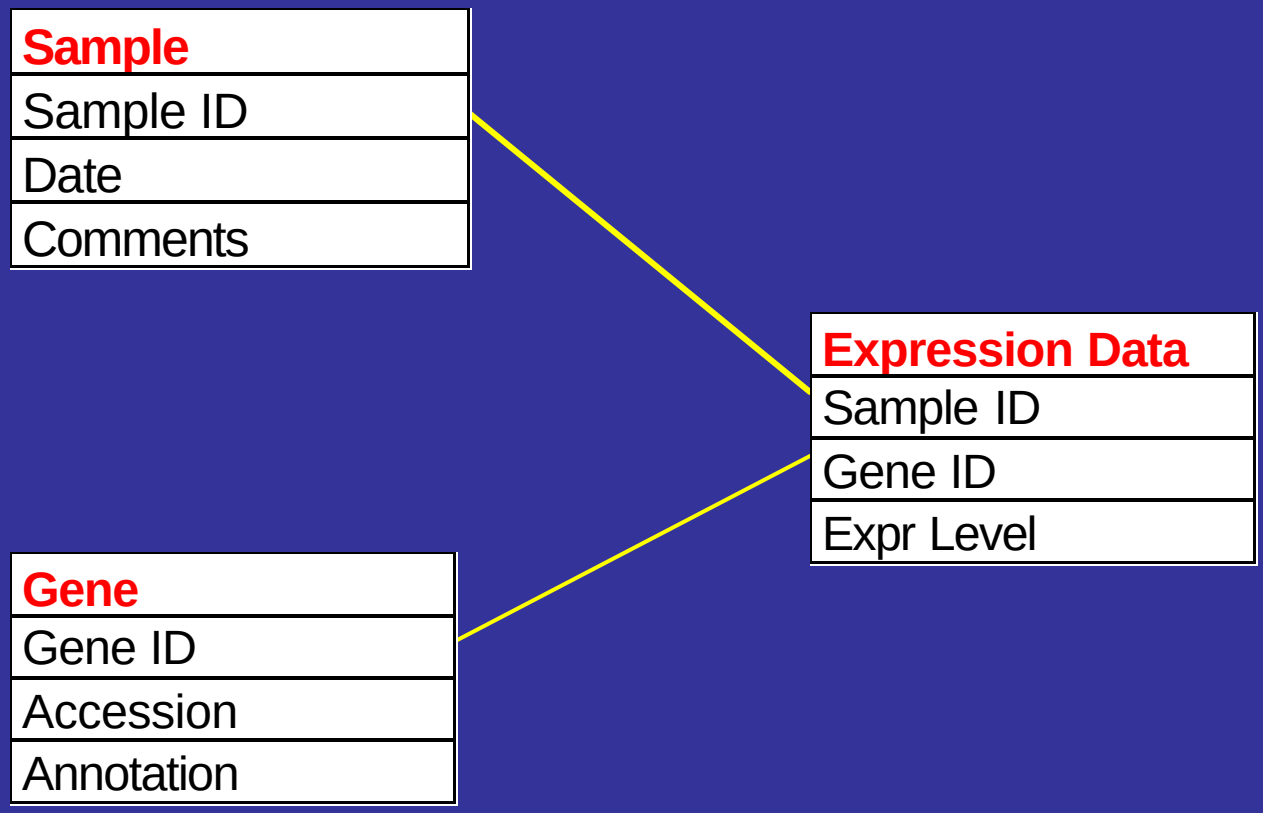
LocusLink ID	Symbol	Full Name	Species
1	A1BG	alpha-1-B glycoprotein	Human
2	A2M	alpha-2-macroglobulin	Human
3	A2MP	alpha-2-macroglobulin pseudogene	Human
9	NAT1	N-acetyltransferase 1	Human
10	NAT2	N-acetyltransferase 2	Human
12	SERPINA3	serine (or cysteine) proteinase inhibitor, clade A (alpha-1 antiproteinase, antitrypsin), member 3	Human
13	AADAC	arylacetamide deacetylase (esterase)	Human
14	AAMP	angio-associated, migratory cell protein	Human
15	AANAT	arylalkylamine N-acetyltransferase	Human
16	AARS	alanyl-tRNA synthetase	Human

LocusLink ID	Accession
1	AA484435
1	W25099
1	AI022193
1	T80683
2	X68728
2	NM_000014
2	M36501
2	M11313
3	M24415
9	NM_000662
9	AF071552
9	U80835

LocusLink ID	GO ID
2	6886
2	4866
2	8320
16	5737
19	4002
20	5524
25	8630
25	6298

A1BG a2m

A simple schema for microarray database

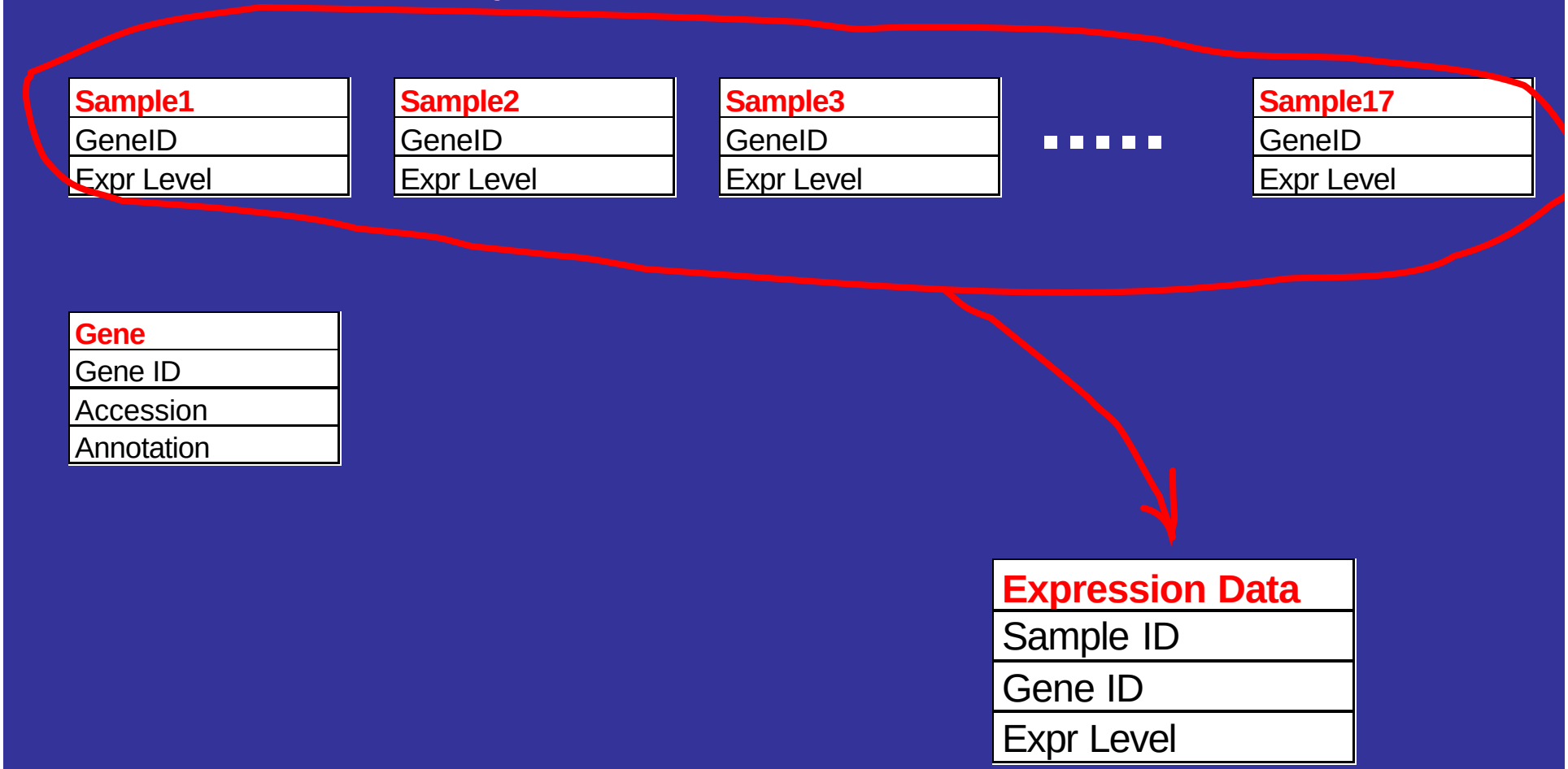


How to design a database schema

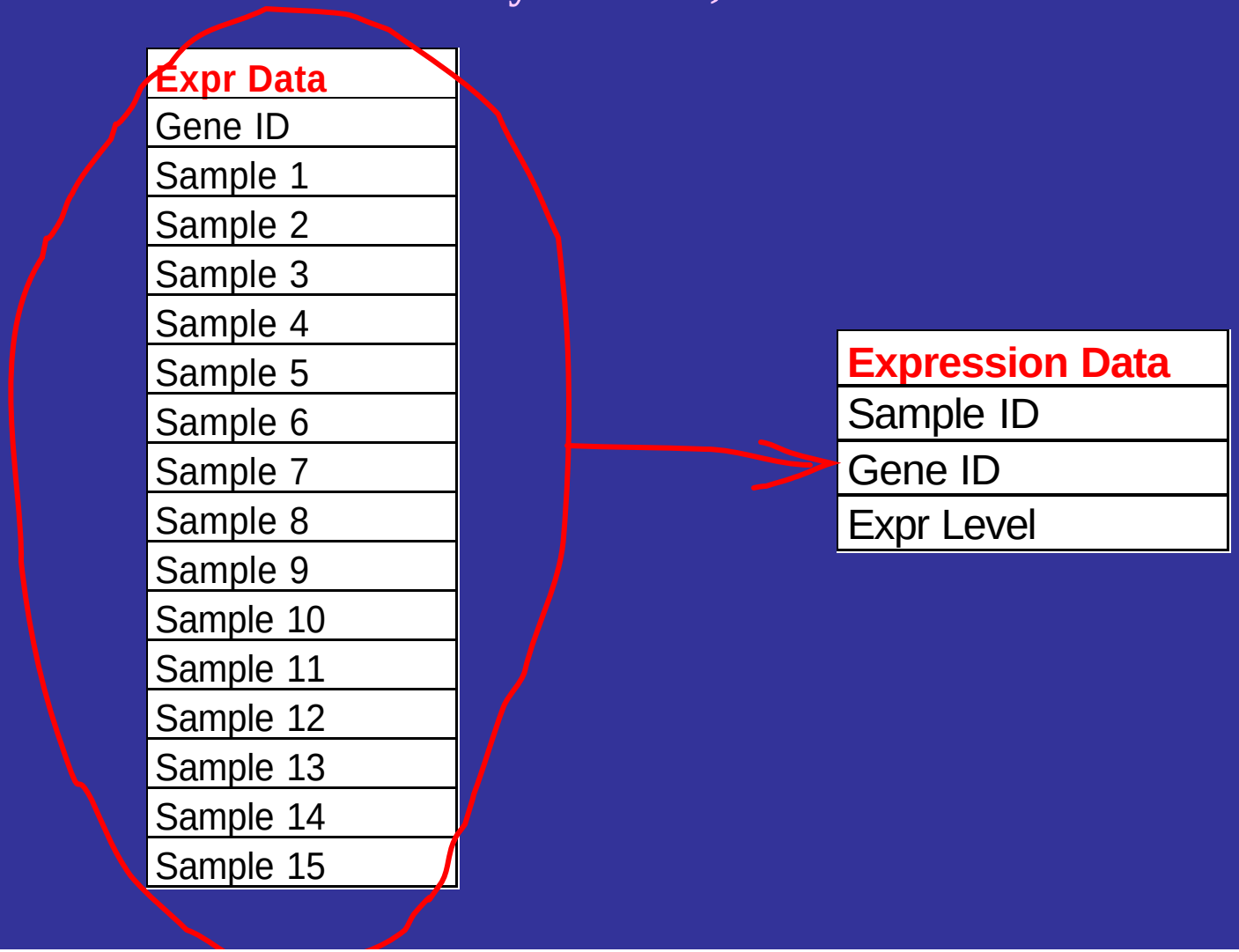
1. How many tables?
2. What is the data relationship between tables?

Common mistakes in database design

Schema 1: Too many tables



Schema 2: Too many columns, and column structure not stable



Data Relationships:

one TO one: **Pubmed ID TO Title (1 table)**

one TO many: **Journal Name TO Pubmed ID (2 tables)**

many TO many: **Pubmed ID TO Authors (3 tables)**

Pubmed ID	Journal ID	Vol	Page	Title
11983181	3	9	911	A caspase-related protease ...
11983180	3	9	903	A Mechanism for Microtubule ...
11983179	3	9	891	Hypermethylation of the Cap ...
11983178	3	9	879	Inhibition of reverse transcription ...
12024052	1	22	4433	A Mammal-Specific Exon ...
12024051	1	22	4419	Chk2 activation and ...
12024050	1	22	4402	The G(1) Cyclin Cln3 Promotes...

Data Relationships:

one TO one: Pubmed ID TO Title (1 table)

one TO many: Journal Name TO Pubmed ID (2 tables)

many TO many: Pubmed ID TO Authors (3 tables)

ID	Journal Name	Abbreviation	ISSN
1	Molecular and Cellular Biology	Mol Cell Biol	0270-7306
2	Molecular Biology of the Cell	Mol Biol Cell	1059-1524
3	Molecular Cell	Mol Cell	1097-2765
4	Nature Review: Molecular Cell Biology	Nat Rev Mol Cell Biol	1471-0072
...	…	…	…

Journal Table

Pubmed ID	Journal ID	Vol	Page
11983181	3	9	911
11983180	3	9	903
11983179	3	9	891
11983178	3	9	879
12024052	1	22	4433
12024051	1	22	4419
12024050	1	22	4402
12006669	2	13	1788
12006670	2	13	1792

Reference Table

Data Relationships:

one TO one: Pubmed ID TO Title (1 table)

one TO many: Journal Name TO Pubmed ID (2 tables)

many TO many: Pubmed ID TO Authors (3 tables)

Pubmed ID	Journal ID	Vol	Page
11983181	3	9	911
11983180	3	9	903
11983179	3	9	891
11983178	3	9	879
12024052	1	22	4433
12024051	1	22	4419
12024050	1	22	4402

Reference Table

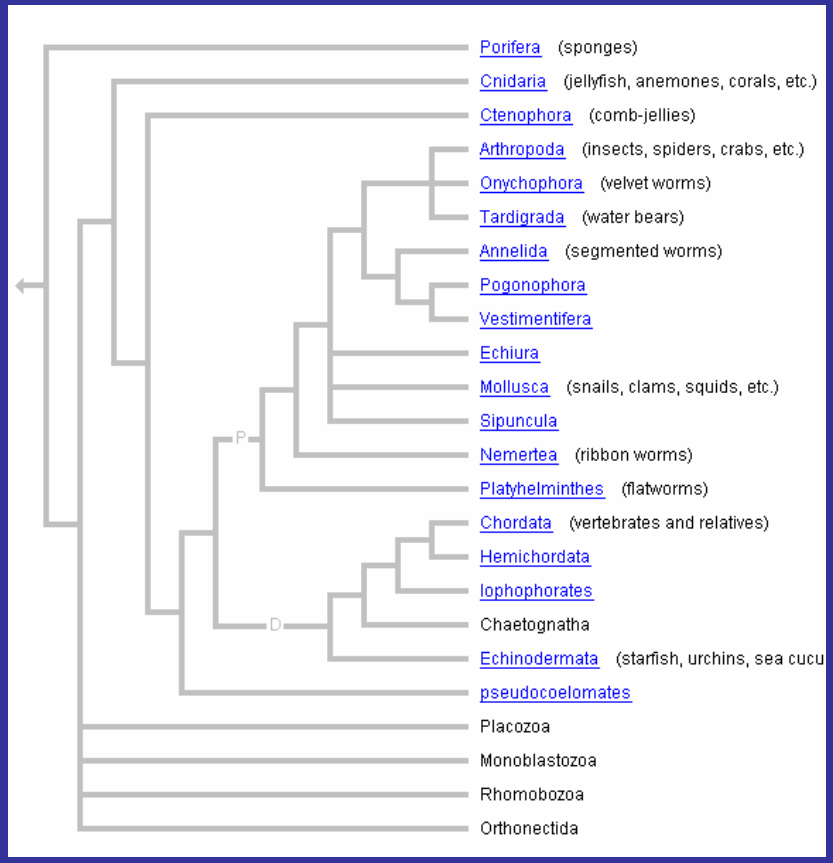
Pubmed ID	Author ID	Rank
11983181	1	6
11983181	2	1
10080892	1	4
9531555	1	3
…	…	…

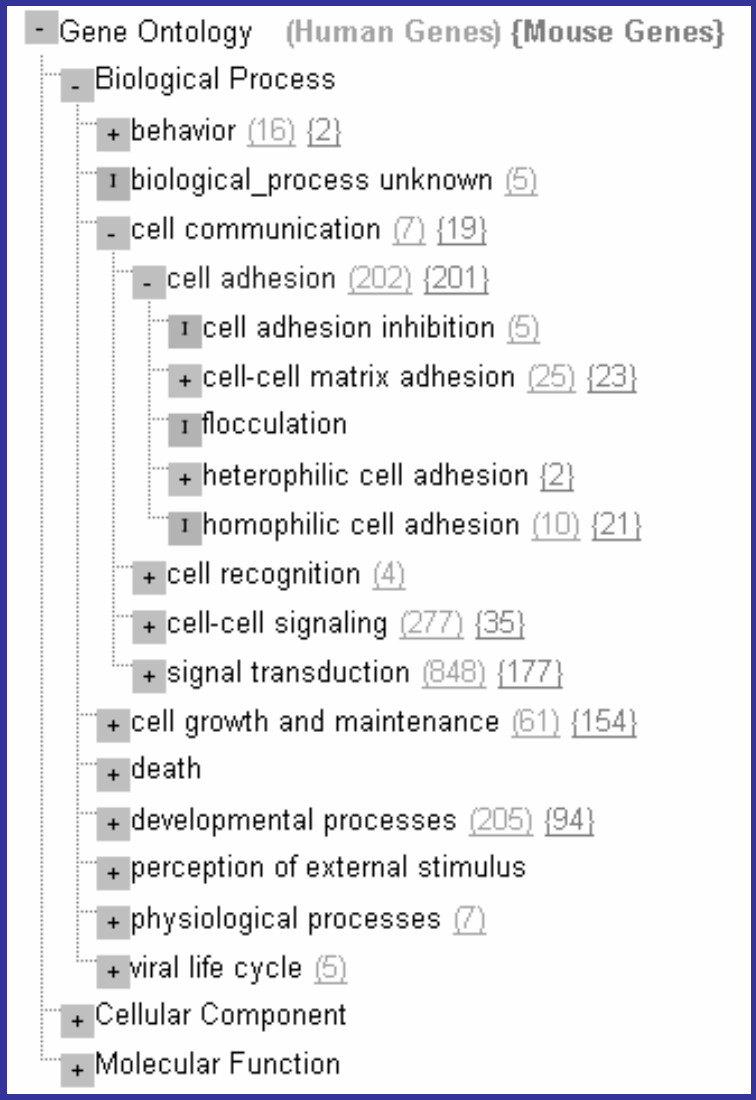
Reference2Author

Author ID	Name	Institution
1	Milligan RA	Scripps
2	Moores CA	Scripps
3	GUO J	Scripps
…	…	…

Author Table

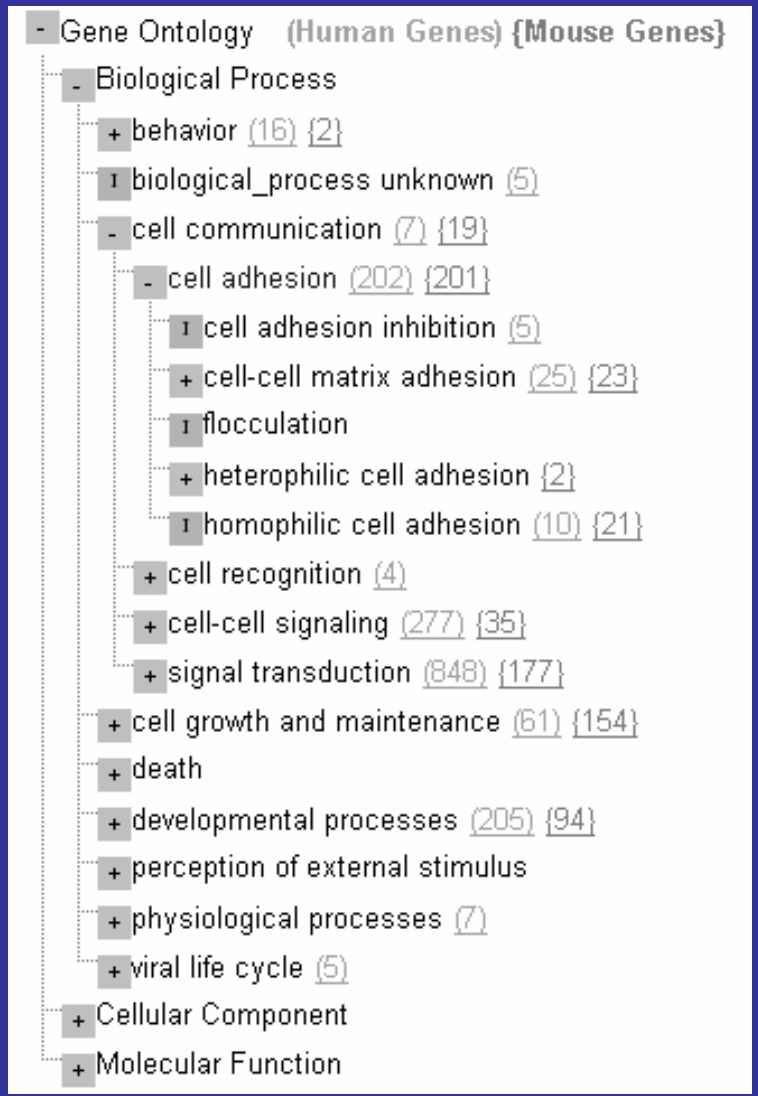
Hierarchical Data Relationship



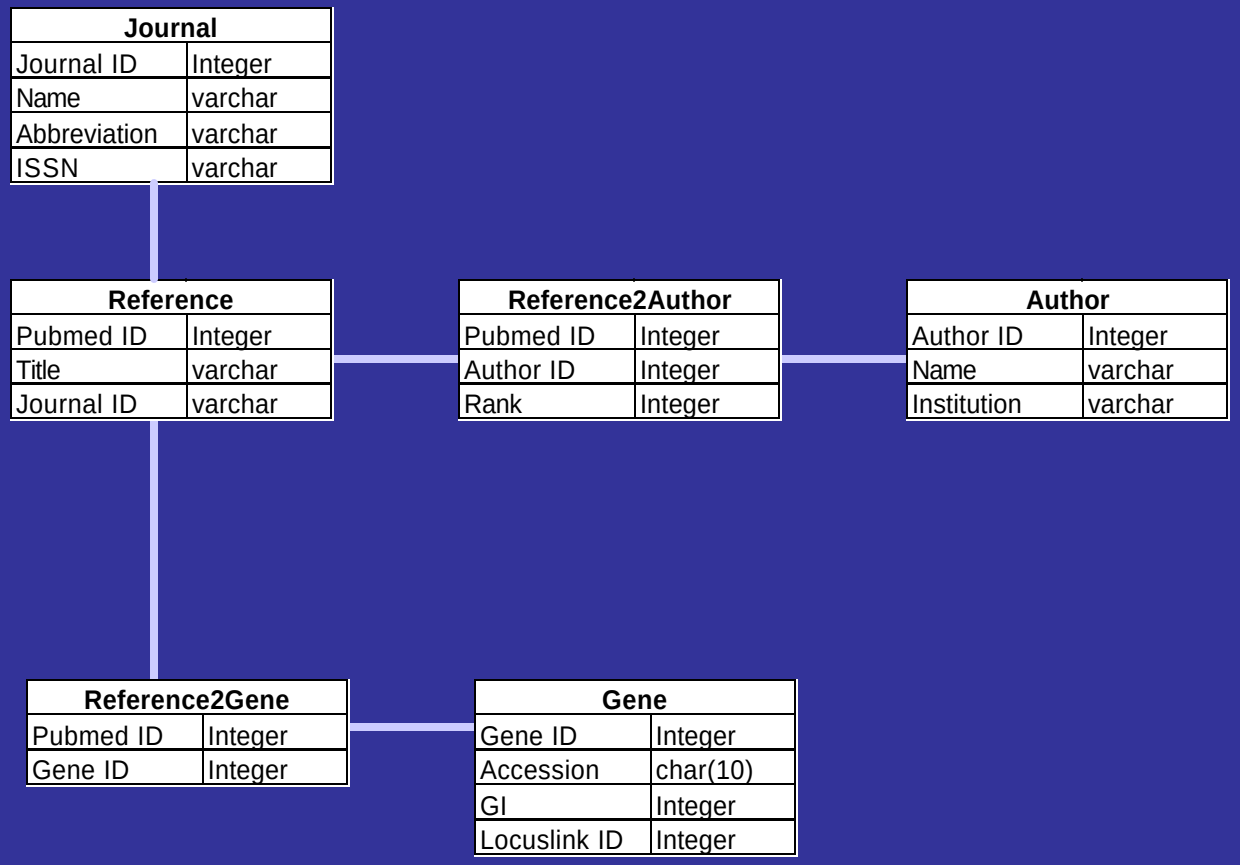


Recursive

Parent	Child
Biological Process	Behavior
Biological Process	Cellular Communication
Biological Process	Death
Biological Process	Developmental Process
Cellular Communication	Cell Adhesion Inhibition
Cellular Communication	Homophilic adhesion

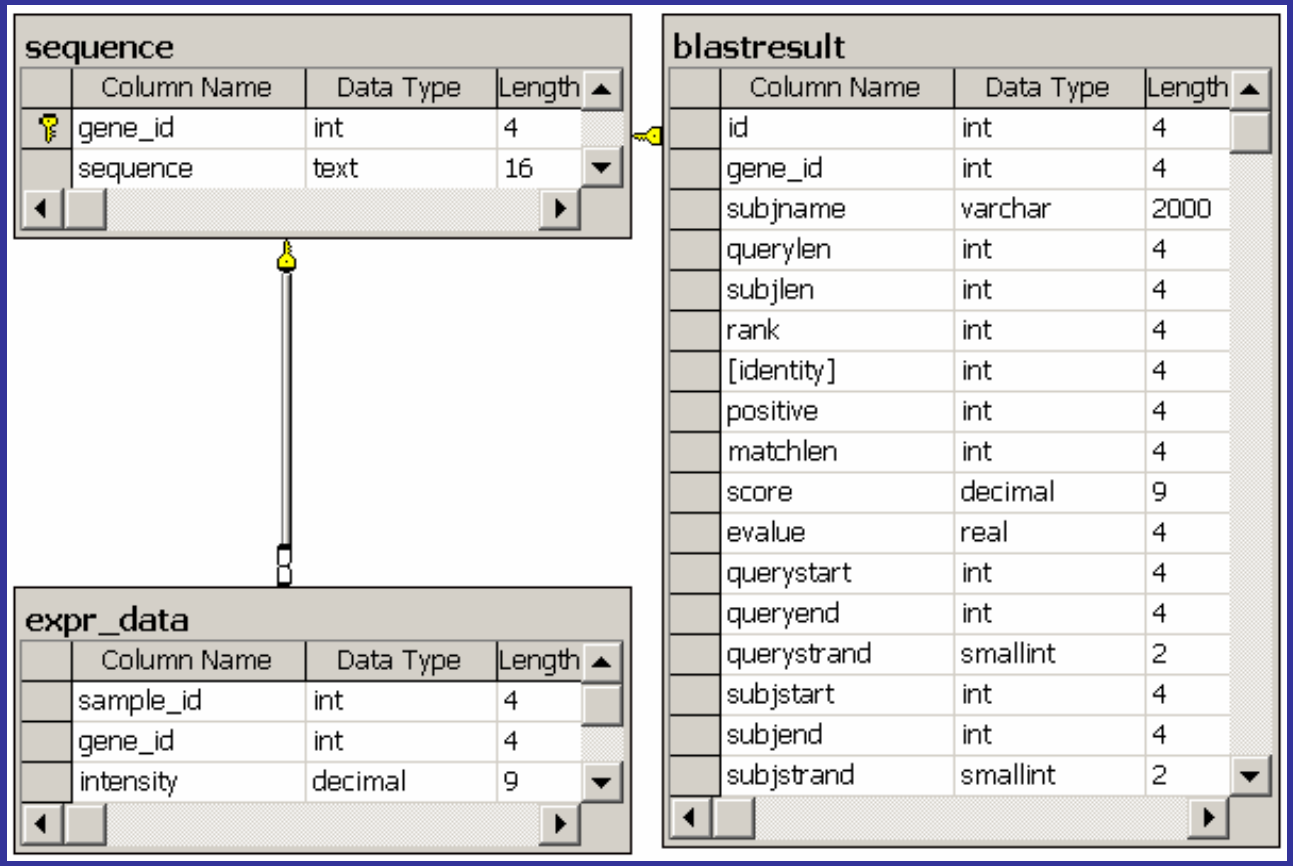


Reference Database Schema



Database project: Create an expression profile database

1. Create tables
2. Import data
3. Query data



Data types:

1. **int** 12, 0, 123, 23943
2. **dec** 12.01, -23.00, 0.343
3. **varchar(100)** "smn gene", "to be tested"
4. **text** "atgcaaatttgcccc ..... atggg"
5. **smallint** 0, 12, -14
6. **datetime** "2002-03-18 11:49:41.980"

Create tables:

Using query analyzer, run scripts:
create_blast.sql, create_expr.sql, create_seq.sql

OR

Using Enterprise manager

Populate data:

Using "bulk insert" to load data from text file into the database

Code:

```
bulk insert blastresult
from '\\ctcfsrv1\tc\cbsu\workshop\database\blastdata'
bulk insert sequence
from '\\ctcfsrv1\tc\workshop\database\sequence.txt'
bulk insert expr_data
from '\\ctcfsrv1\tc\cbsu\workshop\database\sample1.txt'
bulk insert expr_data
from '\\ctcfsrv1\tc\cbsu\workshop\database\sample2.txt'
```

SQL

Structured Query Language

- How to retrieve data?
- How to add data?
- How to modify data?

How to retrieve data from a database

A basic query:

select <column name>

from <table name>

where <condition>

Use bracket for table and column names

eg.
select [intensity]
from [expr_data]
where [gene_id] = 100

The conditions in select statement

Data Comparison

= > < != <= >=

Syntax:

... where [gene_id]<=10

"and", "or", "not", ( )

Syntax:

... where [gene_id]=100 and [sample_id]=1

... where ([gene_id]=100 or [gene_id]=101)
and [sample_id]=1

The conditions in select statement

"between" (Note: between is inclusive)

Syntax:

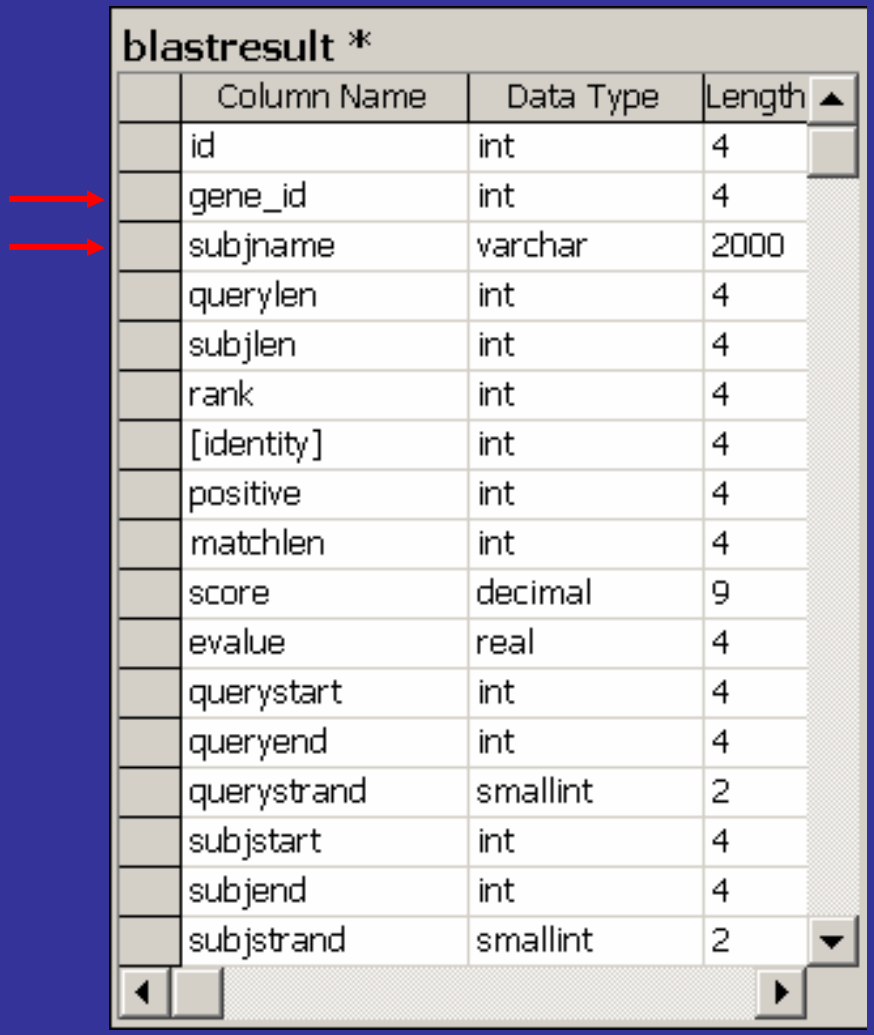
... where gene_id between 10 and 20

"in" "not in"

Syntax:

...where gene_id in (1, 3, 4, 45)

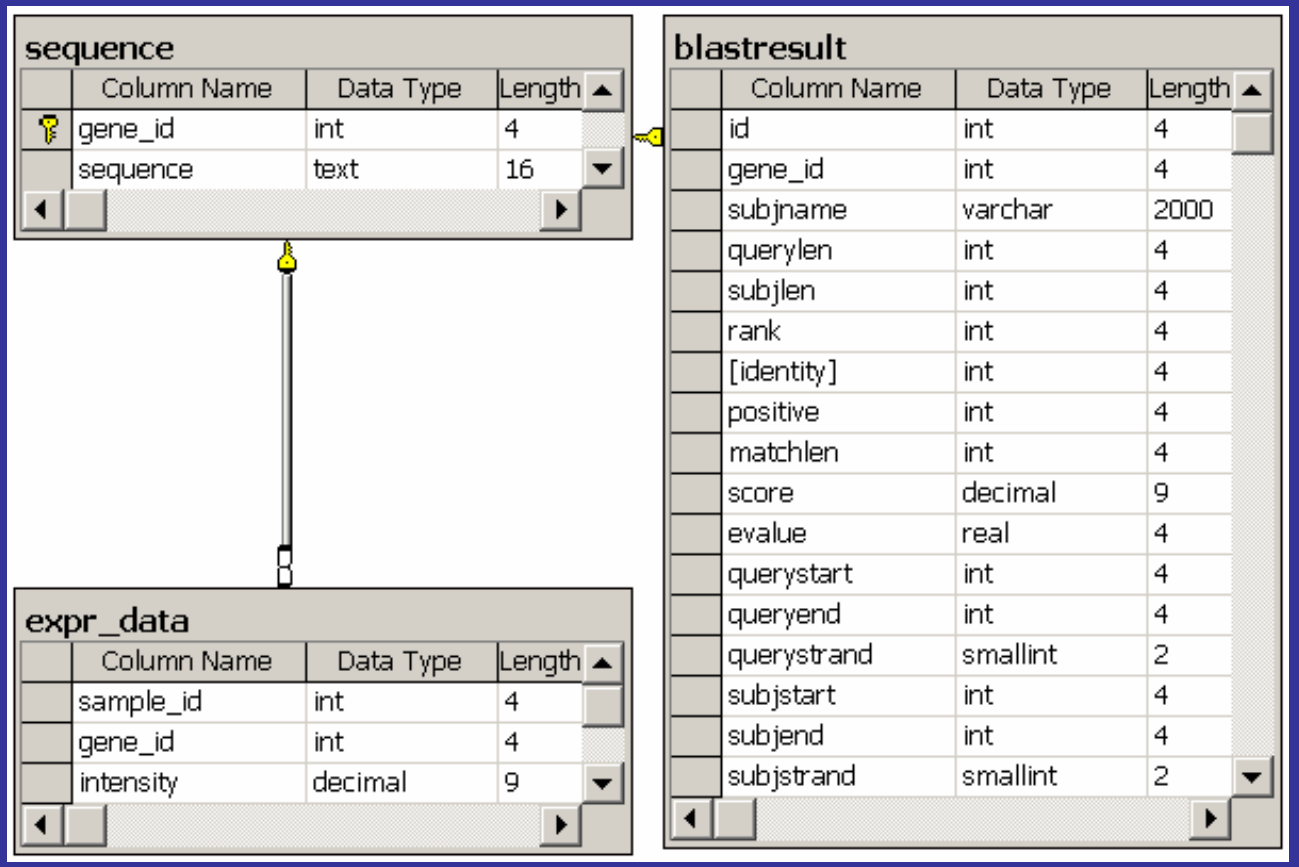
Exercise: Find all the blast targets for gene 100



blastresult *

Column Name	Data Type	Length
id	int	4
gene_id	int	4
subjname	varchar	2000
querylen	int	4
subjlen	int	4
rank	int	4
[identity]	int	4
positive	int	4
matchlen	int	4
score	decimal	9
evalue	real	4
querystart	int	4
queryend	int	4
querystrand	smallint	2
subjstart	int	4
subjend	int	4
subjstrand	smallint	2

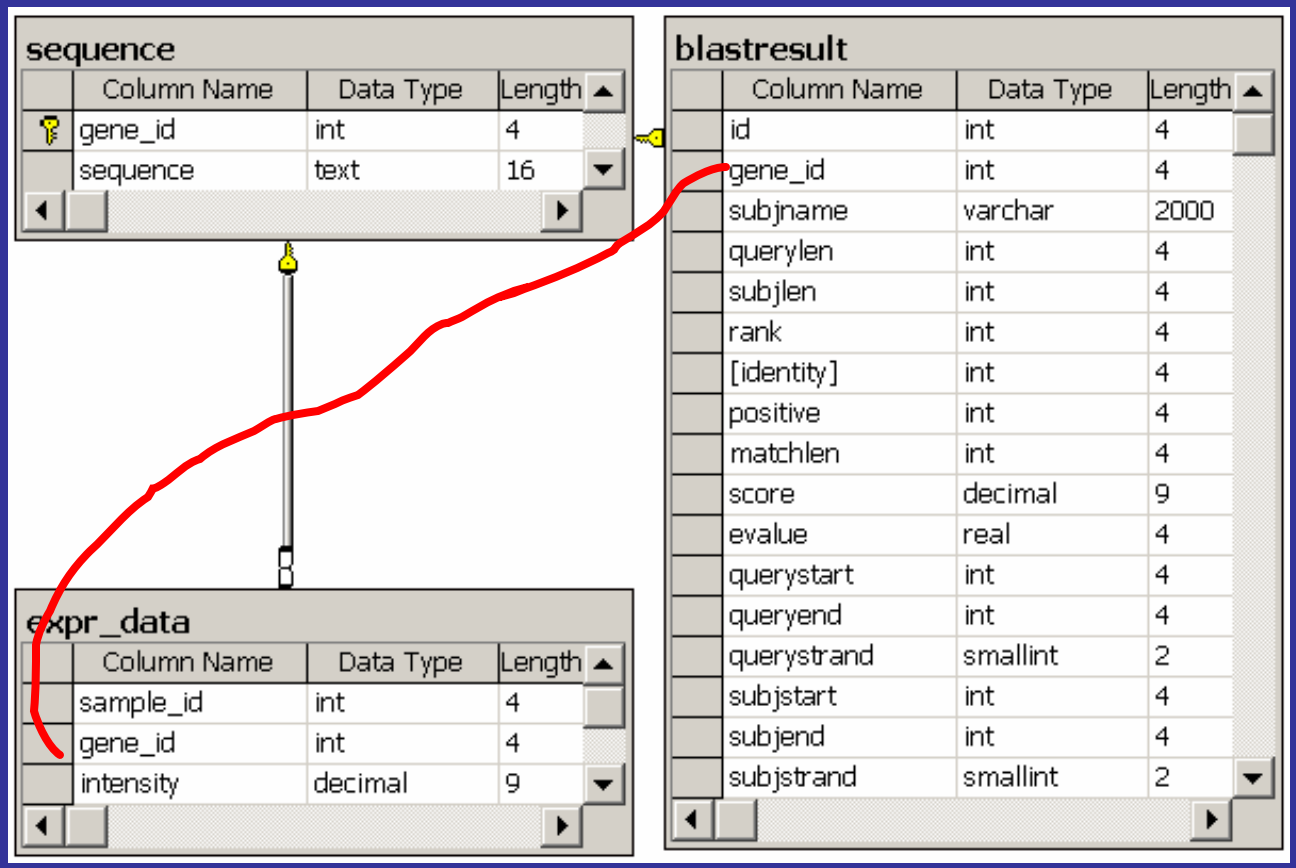
Join: To query data across multiple tables



Question:

List all the genes that have expression level higher than 15000 in experiment 1?

What are the blast targets of these genes?



Answer:

```
select b.[gene_id], b.[subjname]
from [blastresult] b, [expr_data] e
where e.[gene_id]=b.[gene_id]
and e.[intensity]>15000 and
e.[sample_id]=1
```

Alias

Self-join: To compare data on different rows in the same table

expr_data

	Column Name	Data Type	Length
	sample_id	int	4
	gene_id	int	4
	intensity	decimal	9

Question:

Get the list of genes that
with expression level three times higher in sample 2
than in sample 1?

expr_data

	Column Name	Data Type	Length
	sample_id	int	4
	gene_id	int	4
	intensity	decimal	9

a: [sample_id]=1

expr_data

	Column Name	Data Type	Length
	sample_id	int	4
	gene_id	int	4
	intensity	decimal	9

b: [sample_id]=2

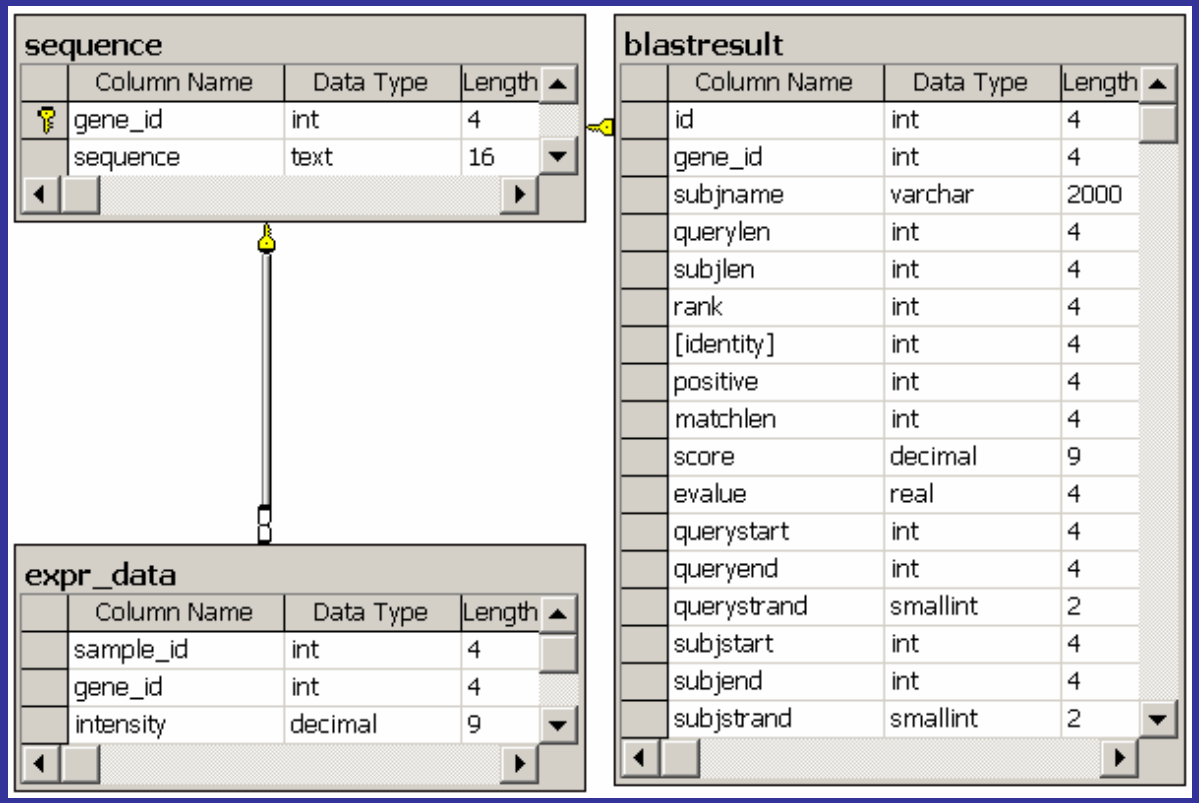
expr_data

	Column Name	Data Type	Length
	sample_id	int	4
	gene_id	int	4
	intensity	decimal	9

Answer:

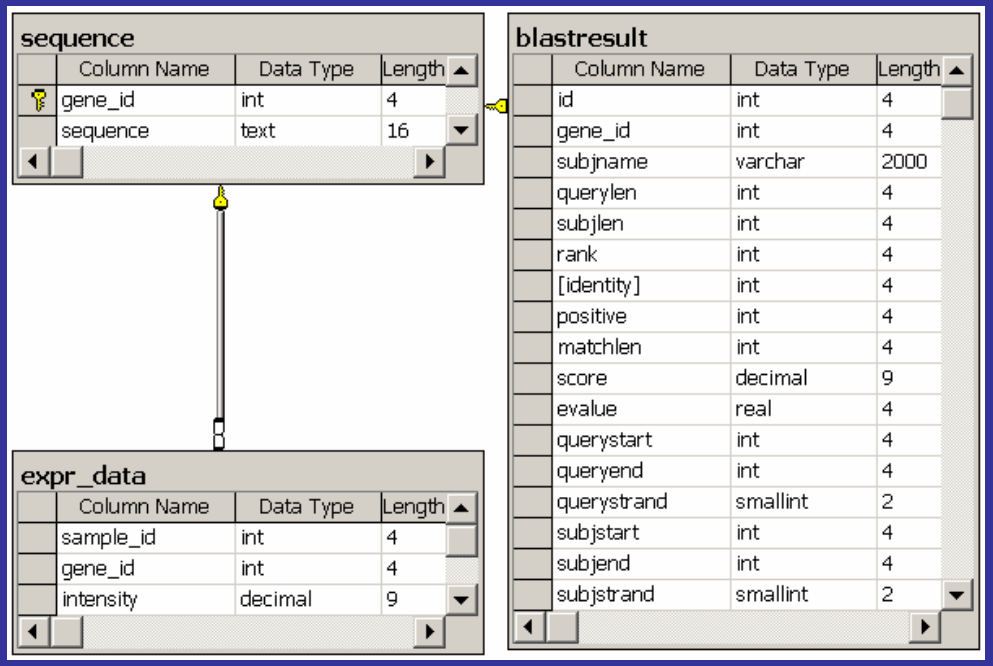
```
select a.[gene_id], a.[intensity], b.[intensity]
from [expr_data] a, [expr_data] b
where a.[gene_id] =b.[gene_id]
and a.[sample_id]=1 and b.[sample_id]=2
and a.[intensity] >100
and b.[intensity] / a.[intensity] >3
```

Sub query: Query a subset of data



Question:

Find the blast targets of the genes in last query?



Answer:

```
select [gene_id], [subjname]
from [blastresult]
where [gene_id] in
        (select a.[gene_id]
        from [expr_data] a, [expr_data] b
        where a.[gene_id]=b.[gene_id]
        and a.[sample_id]=1 and b.[sample_id]=2
        and a.[intensity]>100
        and b.[intensity]/ a.[intensity]>3)
```

Functions for select statement

1. count
2. max
3. min
4. distinct

Examples:

1. **count**

```
select count(*) from expr_data
```

2. **max**

```
select max (intensity) from expr_data
```

3. **distinct**

```
select count(distinct gene_id) from expr_data
```

"order by" in the select statement

```
select gene_id
from expr_data
where sample_id=1
order by intensity
```

"group by" in the select statement

```
select [gene_id],  count ([subjname])
from [blastresult]
group by [gene_id]
```

Modify tables: insert, update and delete command

1. create table [sample]
   ([sample_id] int, [comments] varchar(500))

2. insert into [sample]
   ([sample_id], [comments])
   values (1, '0 hour')

3. update [sample]
   set [comments]='0 hour 30C'
   where [sample_id] =1

4. delete from [sample] where [sample_id] =1;

Database design project:

Design a LocusLink Database that can store the following information:

1. Locus link ID
2. Official Gene Symbol
3. Gene Ontology annotation
4. References (using Pubmed ID)
5. Sequences (using Accession Number, mark whether it is a refseq sequence)
6. Link to other Database: Unigene, GeneCard, OMIM

Read BRCA2 page before you start.

Programming for dummies -- Level 1

```
...
msg = "Hello World"
print msg
exit
```

Programming for dummies -- Level 2

```
For i=1 to 100000
        print "Are we there yet?"
        if arrive() exit
Next
```

Programming for dummies -- Level 3

Reading and Writing

file

Automate the data processing with PERL

Part 1: Basic PERL

Getting started:

1. Install PERL on your computer.

Mac OS X: PERL is installed by default

Windows: Download active perl from www.activeperl.com

2. Write a perl program using any text editor.

Open WordPad, type:

```
print "Hello World";
```

Save as: “C:\cbsu\perl\hello.pl”

3. Run the program.

Open “command prompt” from Start->Programs->Accessories

Type: cd \cbsu\perl

Type: hello.pl

Project: Process a microarray spreadsheet, calculate the mean and median of the expression data

File: c:\cbsu\perl\expr.xls

ID	Exp. Level
26	2198
27	1384
28	1284
29	983
30	4594
31	2761
33	3287
34	[illegible]
[illegible]	2823
496	799
497	1078
498	1203
499	6688
500	1697

mean: (2198 + 1384 + ... + 1697) / 446

median: 146 185 188 189 ... 1648 1650 1652 ... 17409

No. 223

What does a PERL program look like?

```
open IN, "expr.txt";
$sum = 0;
$count = 0;
while ($data= <IN>)
{
        chomp $data;
        ($id, $value) = split "\t", $data;
        $count =$count +1;
        $sum = $sum + $value;
}
close IN;
$avg = $sum / $count;
print $avg;
```

Step 1: Basic elements in a program:

Strings and Numbers: 1, -45, “hello”

Variables: $data1 $data2

Operators: + - * /

Functions: print, length, open

Step 1: Some basic concepts:

Write the program step1.pl:

```
$d1 = 2198;
$d2 = 1384;
$d3 = 1284;
$sum = $d1 + $d2 + $d3;
$avg = $sum / 3;
print $sum;
```

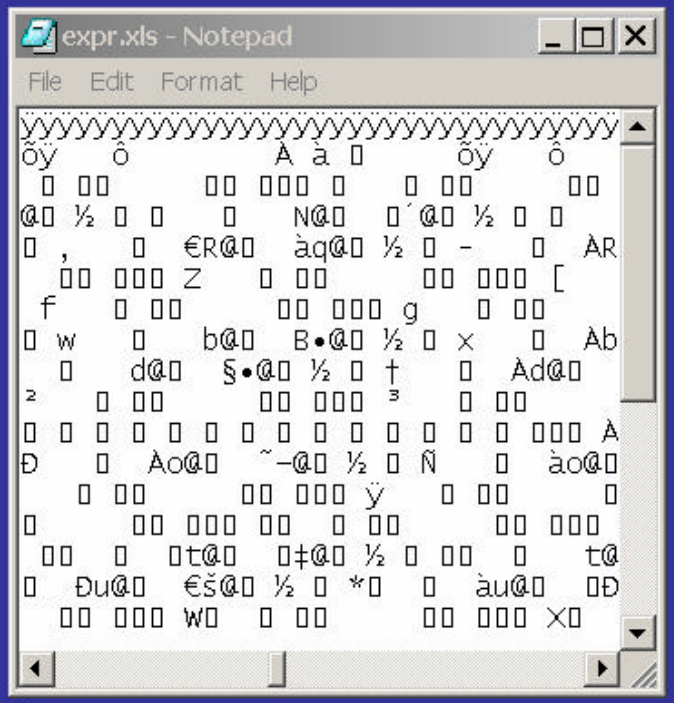
Step 2: About files

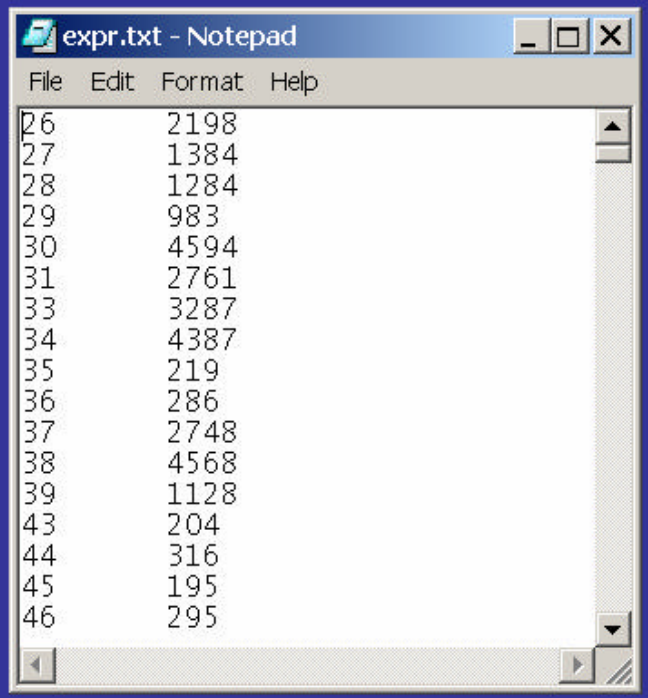
Binary files:

- **Image files**
- **Executable programs**
- **MS Word and Excel documents**

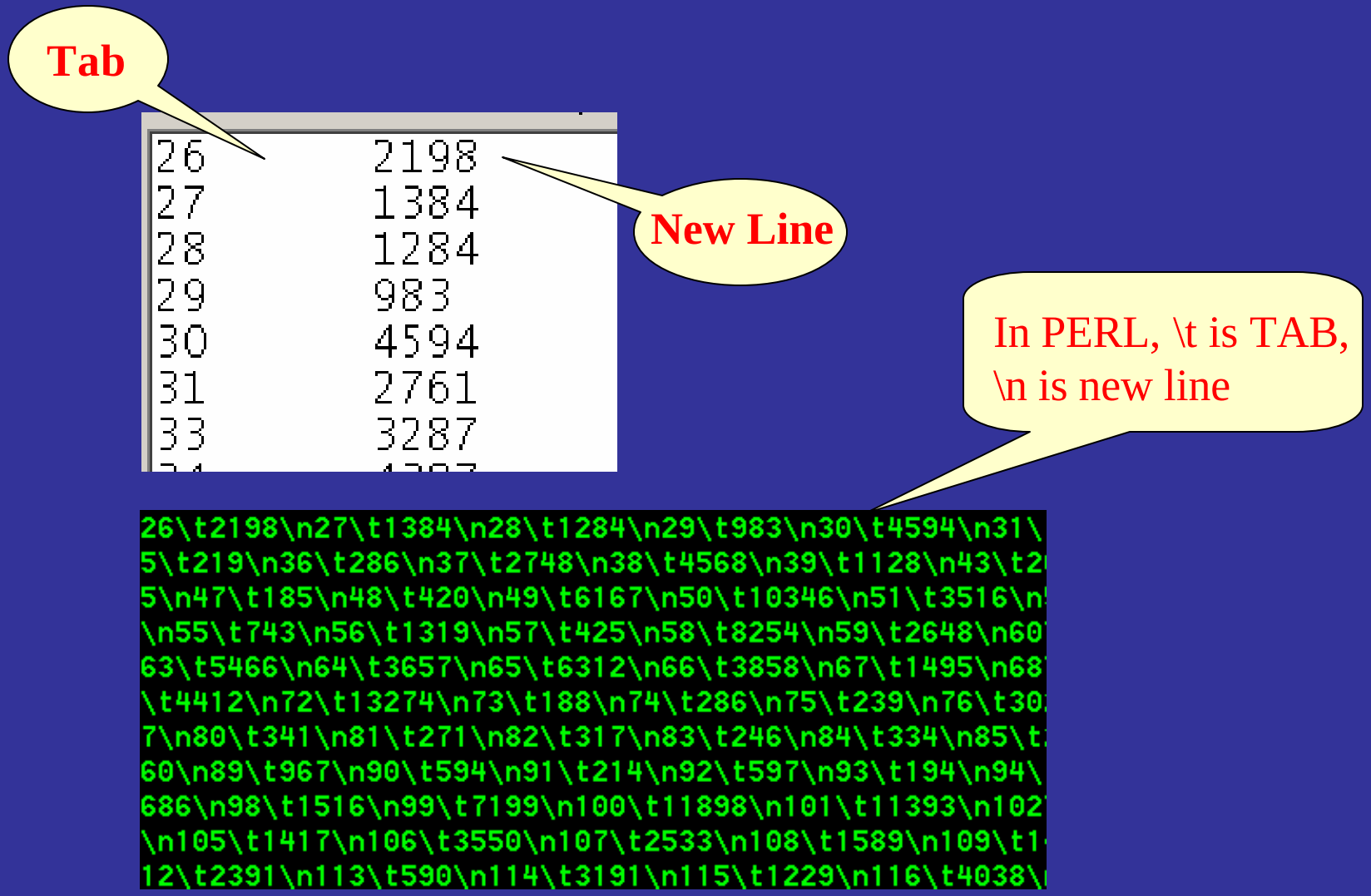
Text file:

- **Notepad (Windows) SimpleText(Mac)**
- **Perl script**





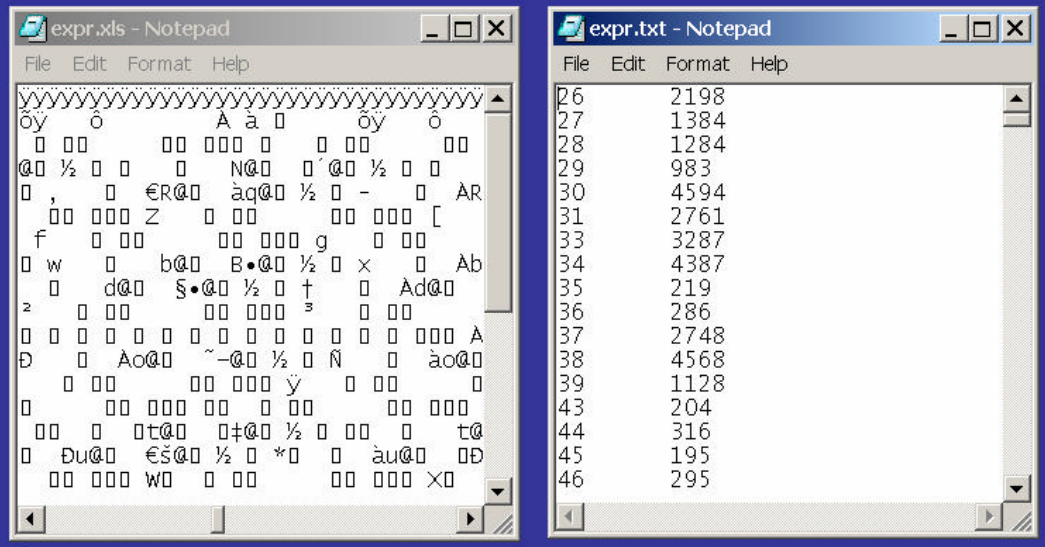
Hidden characters in a tab-delimited text file



Exercise:

Save an Excel document as a tab delimited text file.

C:\cbsu\perl\expr.xls -> C:\cbsu\perl\expr.txt



Read from a File

open a file handle:

```
open IN, "myfile.txt";
```

Read a line from a file:

```
$myline = <IN>;
```

Close a file handle:

```
close <IN>;
```

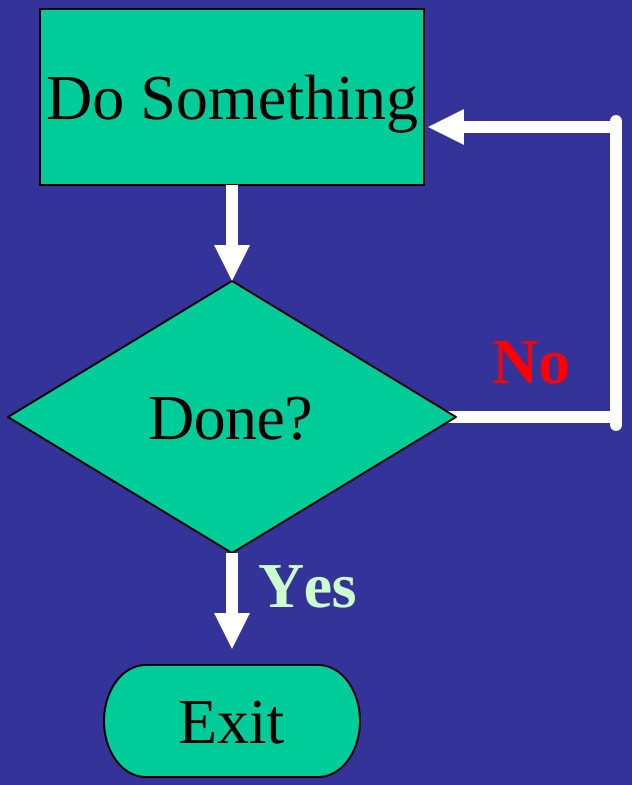
Exercise:

Write program: step2.pl

```
open IN, "expr.txt";     #open a file, get a filehandle
$data = <IN>;            #Read first line in the file
print $data;
$data = <IN>;            #Read next line in the file
print $data;
$data = <IN>;            #Read next line in the file
print $data;
close IN;
```

Step 3: LOOP

```
open IN, "expr.txt";
$data = <IN>;
print $data;
$data = <IN>;
print $data;
$data = <IN>;
print $data;
```




```
open IN, "expr.txt";
while ($data= <IN>)
{
        print $data;
}
```

Step 4: Process String

<IN> actually read in a string: 26\t2198\n

Some string functions:

1. **chomp**
   remove the new line characters
   eg. chomp $data;

2. **split**
   split the string into an array
   eg. ($data1, $data2) = split “\t”, $dataline;

Exercise: Write program:

```
open IN, "expr.txt";
while ($data= <IN>)
{
      chomp $data;                          # remove \n
      ($id, $value) = split "\t", $data;    #split the string
      print $value;
}
close IN;
```

Step 5: Write the program to calculate the mean

Write program:

```
open IN, "expr.txt";
$sum = 0;
$count = 0;
while ($data= <IN>)
{
      chomp $data;
      ($id, $value) = split "\t", $data;
      $count =$count +1;
      $sum = $sum + $value;
}
close IN;
$avg = $sum / $count;
print $avg;
```

Step 6: About array and hash

Array:

```
@datalist = ($data1, $data2, $data3);

$mydata = $datalist[0];
```

Hash:

```
%datahash = ($key1, $value1, $key2, $value2, $key3, $value3);

$mydata = $datahash{$key1};
```

Step 7: Write the program to find the median value

Write program:

```
open IN, "expr.txt";
@datalist = ();;
$count =0;
while ($data= <IN>)
{
      chomp $data;
      ($id, $value) = split "\t", $data;
      push @datalist, $value;
      $count =$count +1;
}
close IN;
@datalist = sort {$a <=> $b} @datalist;    # sort the array
print $datalist[int($count/2)];
```

What is BioPERL

A PERL library for biology data analysis.

Major Functions:

- **Remote connection to major genomic databases**
- **Parsing and converting several sequence format (Fasta, Genbank, EMBL, Swiss-prot, etc.)**
- **Parsing reports from blast, hmmer, genscan, etc.**
- **Sequence manipulation.**
- **Handle large genomic sequence on a computer with limited memory**

.....

BioPERL is written with Object Oriented PERL

What is Object Oriented Programming?

A regular string:

```
$seq = "atgcccgtgctggaatgc";
```

An object:

```
$seq = Bio::PrimarySeq->new (-seq => ' atgcccgtgctggaatgc ');
```

A regular string:

```
$seq = "atgcccgtgctggaatgc";
print $seq;
```

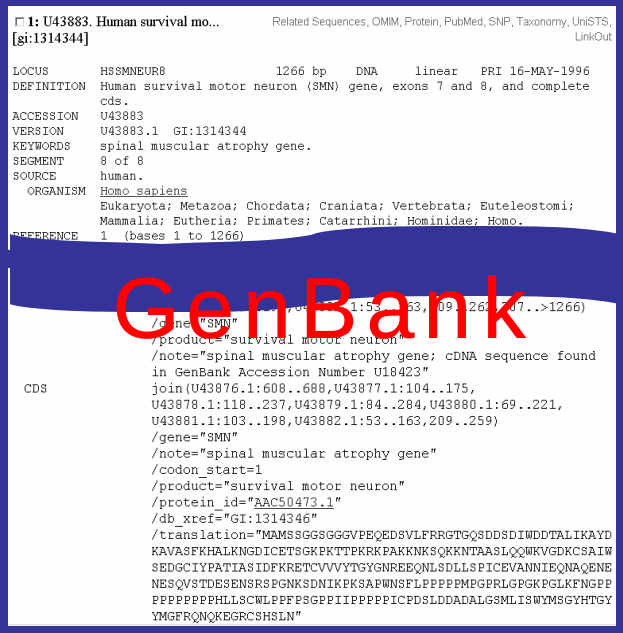
An object:

```
$seq = Bio::PrimarySeq->new (-seq => ' atgcccgtgctggaatgc ');
print $seq ->seq;
print $seq->translate->seq;
print $seq->revcom->seq;
print $seq->subseq(2,4);
```

Commonly used BioPERL objects 1: seqio

Usage:

- Covert to different sequence format



1: U43883. Human survival mo... [gi:1314344]

Related Sequences, OMIM, Protein, PubMed, SNP, Taxonomy, UniSTS, LinkOut

```
LOCUS       HSSMNEUR8               1266 bp    DNA     linear   PRI 16-MAY-1996
DEFINITION  Human survival motor neuron (SMN) gene, exons 7 and 8, and complete
            cds.
ACCESSION   U43883
VERSION     U43883.1  GI:1314344
KEYWORDS    spinal muscular atrophy gene.
SEGMENT     8 of 8
SOURCE      human.
  ORGANISM  Homo sapiens
            Eukaryota; Metazoa; Chordata; Craniata; Vertebrata; Euteleostomi;
            Mammalia; Eutheria; Primates; Catarrhini; Hominidae; Homo.
REFERENCE   1  (bases 1 to 1266)
```

GenBank

```
                     /gene="SMN"
                     /product="survival motor neuron"
                     /note="spinal muscular atrophy gene; cDNA sequence found
                     in GenBank Accession Number U18423"
     CDS             join(U43876.1:608..688,U43877.1:104..175,
                     U43878.1:118..237,U43879.1:84..284,U43880.1:69..221,
                     U43881.1:103..198,U43882.1:53..163,209..259)
                     /gene="SMN"
                     /note="spinal muscular atrophy gene"
                     /codon_start=1
                     /product="survival motor neuron"
                     /protein_id="AAC50473.1"
                     /db_xref="GI:1314346"
                     /translation="MAMSSGGSGGGVPEQEDSVLFRRGTGQSDDSDIWDDTALIKAYD
                     KAVASFKHALKNGDICETSGKPKTTPKRKPAKKNKSQKKNTAASLQQWKVGDKCSAIW
                     SEDGCIYPATIASIDFKRETCVVVYTGYGNREEQNLSDLLSPICEVANNIEQNAQENE
                     NESQVSTDESENSRSPGNKSDNIKPKSAPWNSFLPPPPPMPGPRLGPGKPGLKFNGPP
                     PPPPPPPPHLLSCWLPPFPSGPPIIPPPPPICPDSLDDADALGSMLISWYMSGYHTGY
                     YMGFRQNQKEGRCSHSLN"
```

```
>gi|17530902|ref|NM_078572.1| Drosophila melanogaster Protein tyrosine p
GAAAGAAACTAGCGAAAAGCCCAATTCAAAAATCGCACAATAAATTGCAAAGTCCAAAGCAACAACAACA
TAGGAACAAATAAAACAAGAAGCAGGCAACAAAAGAGAGCCAGGCGAGAGGAAAACAACAACGAAAACAG
CAAAACGTCGGCCCCAAAATGTTGTACCAGCTAAGTAAAGCCACTACTAGAATCCGGCTCAAAAGGCAAA
AAGCCGTTCCACAACATCGCTGGCTCTGGAGTTTAGCATTTCTGGCAGCATTTACACTAAAGGACGTTCG
ATGTGCAGATCTAGCCATAAGTATACCCAATAATCCCGGCCTGGACGATGGGGCCTCCTATCGACTGGAC
TATAGTCCGCCCTTCGGTTATCCGGAGCCCAACACGACGATTGCCTCGCGGGAAATCGGCGATGAGATTC
AATTCTCACGCGCCCTACCGGGCACCAAATACAACTTTTGGCTGTACTACACGAACTTTACGCACCACGA
TTGGCTCACCTGGACGGTGACGATAACAACGGCTCCGATCCCGCCGCTGAATCTGTCCGTTCAGGTGAGA
AGCGGCAAGAATGCCATCATCCTATGGTCACCGCCCACCCAGGGCAGCTATACGGCGTTTAAGATCAAGG
TGCTCGGACTGTCGGAGGCGTCGAGCAGCTACAACCGCACCTTCCAGGTGAACGACAATACATTCCAGCA
CAGCGTCAAGGAGCTGACACCCGGAGCCACCTACCAGGTGCAGGCATATACCATCTACGATGGCAAGGAG
TCGGTGGCCTACACTAGTCGCAACTTCACCACAAAGCCCAAGACTCCGGGGAAATTCATCGTCTGGTTCC
GTAATGAGACGACACTGCTGGTCCTCCGCAGCTGCGATATCGGTGCGCATCTACACGCACTACAAGGT
ATCCATCGAGCCGCCGGATGCCAAGATAGTGTCTTTGTGGAAACGGAGGGCGAACCGCCCGGACCG
GCGCAGGCTGCCTTCAAGGGTCTGGTGCCCGGAAGGGCGTACAACATATCCGTTCAGACGATGTCCGAGG
ATGAGATCTCATTGCCGACGACGGCGCAATATCGAACGGTGCCGTTGCGTCCGCTGAACGTGACCTTTGA
CCGTGACTTTATTACCTCCAATTCGTTCCGTGTCCTGTGGGAGGCGCCCAAAGGGATATCCGAATTTGAC
AAATACCAGGTATCGGTGGCCACAACACGACGCCAATCCACAGTGCCGCGCAGCAATGAACCGGTGGCAT
TCTCTGATTTTCGCGACATCGCCGAACCGGGCAAGACGTTCAATGTGATCGTGAAGACCGTATCCGGCAA
GGTTACCTCGTGGCCAGCCACCGGGGATGTGACACTGCGACCACTGCCCGTTCGCAATTTGCGGAGCATC
AACGATGACAAGACGAATACTATGATCATAACGTGGGAAGCGGATCCGGCCAGCACGCAGGATGAGTATC
GCATTGTATACCACGAACTGGAGACATTTAATGGTGACACCAGTACCCTGACCACGGATCGGACTCGATT
CACACTGGAGAGCCTGCTACCCGGTCGCAACTACTCATTGTCCGTCCAGGCGGTATCCAAGAAGATGGAA
TCGAATGAGACTAGCATCTTTGTGGTCACCCGACCCTCGTCGCCCATCATCGAGGACTTGAAGAGCATAC
GGATGGGTCTGAACATCAGTTGGAAGAGCGATGTCAACTCCAAGCAGGAGCAGTACGAGGTGTTGTACTC
GCGCAACGGAACCAGCGATTTGCGAACCCAAAAGACCAAAGAGTCGCGTCTGGTGATCAAGAATCTGCAG
CCAGGTGCTGCGTATGAACTCAAGGTGTTTGCAGTTAGTCACGATTTGCGCAGCGAACCACATGCCTATT
TCCAAGCAGTTTATCCCAATCCACCACGCAACATGACCATCGAAACGGTGAGGAGTAACTCGGTGCTGGT
ACACTGGTCACCGCCGGAAAGCGGTGAATTTACCGAGTACTCGATACGCTATCGCACGGACAGCGAACAG
```

Fasta

SeqIO class:

Constructor:

```
$in  = Bio::SeqIO->new(-file => "inputfilename" , '-format' => 'Fasta');
$out = Bio::SeqIO->new(-file => ">outputfilename" , '-format' => 'EMBL')
```

Method:

```
$in -> next_seq;
$out-> write_seq($seq);
```

Example of SeqIO:

```
use Bio::SeqIO;
$in  = Bio::SeqIO->new(-file => "inputfilename" , '-format' => 'GenBank');
$out = Bio::SeqIO->new(-file => ">outputfilename" , '-format' => 'Fasta');
while ( my $seq = $in->next_seq() )
{
        out->write_seq($seq);
}
```

Commonly used BioPERL objects 2: Seq

Usage:

For sequence analysis

- **translation**
- **reverse complemention**
- **restriction sites**
- **truncation and mutation**
- **...**

Seq Class

Constructor:

```
$seqobj = $seqio->next_seq();

$seqobj = Bio::PrimarySeq ->new
                    ( -seq => 'ATGGGGTGGGCG',
                    -id  => 'MyGene')
```

Methods:

```
$seqobj ->seq();
$seqobj ->length();
$seqobj ->translate();
$seqobj ->revcom();
$seqobj ->subseq(1, 10);
```

Example of Seq:

```
use Bio::Seq;
use Bio::SeqIO;
$seqin = Bio::SeqIO->new( -format => 'GenBank' , -file => 'myfile.dat');

while(my $seqobj = $seqin->next_seq()) {

        $rev = $seqobj->revcom;
       $pepobj = $seqobj->translate();

       foreach $feat ( $seqobj->top_SeqFeatures() ) {
          if( $feat->primary_tag eq 'exon' ) {
            print "Location ", $feat->start, ":",
            $feat->end," GFF[",$feat->gff_string,"]\n";
          }
       }
}
```

Commonly used BioPERL objects 3: DB

Usage:

For remote access to many sequence database

Bio::DB::GenBank

Bio::DB::GenPept

Bio::DB::SwissProt

Bio::DB::RefSeq

Bio::DB::EMBL

Example of DB:

```
$gb = new Bio::DB::GenBank();

# this returns a Seq object :
$seq2 = $gb->get_Seq_by_acc('AF303112'))

# this returns a SeqIO object :
$seqio = $gb->get_Stream_by_batch([ qw(J00522 AF303112
2981014)]));
```

Commonly used BioPERL objects 4: BPlite

Usage:

For parsing standard blast output

```
BLASTN 2.1.3 [Apr-1-2001]


Reference: Altschul, Stephen F., Thomas L. Madden, Alejandro A. Schaffer,
Jinghui Zhang, Zheng Zhang, Webb Miller, and David J. Lipman (1997),
"Gapped BLAST and PSI-BLAST: a new generation of protein database search
programs",  Nucleic Acids Res. 25:3389-3402.

Query= Contig1
         (870 letters)

Database: genome
           1879 sequences; 832,643 total letters



                                                                  Score     E
Sequences producing significant alignments:                       (bits)  Value

AB010006B20E06                                                      76  1e-014
AC040001000H10                                                      64  4e-011
TT010005A10G01                                                      62  2e-010
DP040002000H03                                                      56  1e-008


>AB010006B20E06
           Length = 464

 Score = 75.8 bits (38), Expect = 1e-014
 Identities = 74/86 (86%)
 Strand = Plus / Minus


Query: 327 agagggagaagcaggttccacgcagggagcccaatgtgggacctaattccaggaccccaa 386
           ||||||||||||||| |||| |||||||||| ||||||||| | || |  ||||||||
Sbjct: 418 agagggagaagcaggctccatgcagggagccctatgtgggacttgatcctgggaccccag 359


Query: 387 gatcactccctgagccaaaggcagac 412
           |||||| ||||||||  |||||||||
Sbjct: 358 gatcacccccctgagctgaaggcagac 333


>AC040001000H10
           Length = 355

 Score = 63.9 bits (32), Expect = 4e-011
 Identities = 71/84 (84%)
```

5126 (183	pir||T0244	183	243	1	40	46	77	68.6
5126 (183	ref|NP_56	183	256	1	40	46	77	68.6
5126 (183	emb|CAA7	183	262	1	49	66	137	63.5
5126 (183	ref|NP_56	183	266	1	47	66	139	58.9
5126 (183	emb|CAC	183	168	1	42	59	117	58.2
5126 (183	pir||T0153	183	746	1	42	59	129	42.4
5126 (183	gb|AAF72	183	1733	1	25	48	104	35
5126 (183	emb|CAB1	183	309	1	31	58	139	32.7
5126 (183	sp|P00567	183	381	1	18	25	59	32.7
5126 (183	pir||I55544	183	381	1	17	25	58	32.7
5126 (183	ref|NP_49	183	1199	1	20	27	58	32.7
5126 (183	gb|AAC96	183	380	1	27	40	99	32
5126 (183	gb|AAC31	183	381	1	17	25	59	32
5126 (183	sp|P05124	183	381	1	17	25	59	32
5126 (183	ref|XP_04	183	381	1	17	25	59	32
5126 (183	pdb|1G0W	183	380	1	17	24	59	31.6
5126 (183	ref|NP_24	183	668	1	23	30	64	31.6
5127 (169	ref|NP_24	169	174	1	16	30	55	33.5
5127 (169	ref|XP_14	169	300	1	19	28	52	32.7
5127 (169	ref|NP_04	169	315	1	27	51	103	32
5129 (204	pir||E7143	204	601	1	75	104	177	133
5129 (204	ref|NP_56	204	517	1	75	104	177	133
5129 (204	emb|CAC3	204	544	1	66	94	183	106
5129 (204	gb|AAK31	204	534	1	58	81	169	92.4
5129 (204	dbj|BAB69	204	447	1	29	45	61	68.9
5129 (204	ref|NP_56	204	527	1	26	34	51	62
5129 (204	ref|NP_18	204	357	1	22	36	43	58.5
5129 (204	ref|NP_17	204	529	1	22	33	52	56.2
5129 (204	dbj|BAB89	204	403	1	26	32	50	53.9
5129 (204	ref|NP_56	204	444	1	26	33	62	52.8
5129 (204	ref|NP_56	204	529	1	24	31	49	52.8
5129 (204	gb|AAF63	204	515	1	26	33	62	52.8
5129 (204	dbj|BAB89	204	398	1	21	28	41	51.2

Example of BPlite:

```
use Bio::Tools::BPlite;
 $report = new Bio::Tools::BPlite(-fh=>\*STDIN);
 $report->query;
 while(my $sbjct = $report->nextSbjct) {
     $sbjct->name;
     while (my $hsp = $sbjct->nextHSP) { $hsp->score; }
 }
```

Project: Finding potential ORFs in the genome

Starting point:

E coli genome sequence

Goal:

A fasta file of all potential ORFs

Step 1: Break the genome sequence into 1 kb fragments

```
# a SeqIO object from the genome file
$in  = Bio::SeqIO->
          new('-format' => 'largefasta' , -file => ecoli_k12');

# a Seq  object of the whole genome
$genomeseq = $in->next_seq();

# get the fragment of the genome
$frag = $genomeseq->subseq(1, 1000);
```

Step 2: 6-frame translation of all the DNA fragments

```
# input is one Seq object (DNA), output are 6 Seq objects (aa)

@seqs = Bio::SeqUtils->translate_6frames($seqobj);
```

Step 3: Finished code C:\cbsu\module2\perl\orf.pl

```
#specify the modules to use
use Bio::SeqIO;
use Bio::SeqUtils;

#make the SeqIO project
$in  = Bio::SeqIO -> new('-format' => 'largefasta' , -file =>
'C:\\cbsu\\module1\\hmmer_projects\\exe2\\ecoli_k12');

#make the Seq project
$genomeseq = $in->next_seq();

#get the length of the sequence
$seqlength = $genomeseq->length;

#specify the output file
$out = Bio::SeqIO->new(-file => ">test.txt" , '-format' => 'Fasta');

#--to be continued in next slide
```

```
for ($start=1; $i<$seqlength; $start=$start+500)  #1kb frag with 0.5 kb overlap
{
        $end = $start + 999;
        if ($end>$seqlength) {
                $end=$seqlength;
        }
        my $tempSeq = $genomeseq->subseq($start, $end);

        my $tempSeqObj = Bio::Seq->new ( -seq => $tempSeq,
                 -id  => "$start-$end",
                 -alphabet => 'dna'
                );
```

#--to be continued in next slide

```
        @seqs = Bio::SeqUtils->translate_6frames($tempSeqObj );
        foreach  (@seqs)
        {
                $out->write_seq($_);
        }
}
```

Some other topics on PERL

System Call

Usage: launch a program

```
$queryfile = "est.fasta";
$outputfile = "result.txt";
system ("blastall -p blastn -d nt -i $queryfile -o $outputfile");
```

ODBC: access SQL Server or other database system

```
use Win32::ODBC;
my $db= new Win32::ODBC (<<EOT);
DRIVER=SQL Server;
SERVER=SQLSRV01;
DATABASE=cbsu_workshop;
TRUSTED_CONNECTION=Yes;
EOT

$db->Sql("select [gene_id] from [sequence]")
while ($db->FetchRow())
{
        ($id)=$db->Data(' gene_id ');
        print $id, "\n";
}
```

CGI: Dynamic Web page

Books and websites:

1. Learning Perl on Win32 Systems *by Randal L. Schwartz, et al*

2. http://www.bioperl.org

3. Object Oriented Perl *by Damian Conway*