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
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**

<table>
<thead>
<tr>
<th>LOCUS</th>
<th>HSSMNEUR8</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEFINITION</td>
<td>Human survival motor neuron (SMN) gene, exons 7 and 8, and complete cds.</td>
</tr>
<tr>
<td>ACCESSION</td>
<td>U43883</td>
</tr>
<tr>
<td>VERSION</td>
<td>U43883.1 GI:1314344</td>
</tr>
<tr>
<td>KEYWORDS</td>
<td>Spinal muscular atrophy gene.</td>
</tr>
<tr>
<td>SEGMENT</td>
<td>8 of 8</td>
</tr>
<tr>
<td>SOURCE</td>
<td>human.</td>
</tr>
<tr>
<td>ORGANISM</td>
<td>Homo sapiens</td>
</tr>
<tr>
<td>REFERENCE</td>
<td>1 (bases 1 to 1266)</td>
</tr>
<tr>
<td>AUTHORS</td>
<td>Jefabourg S., Bigras L., Bachelet S., Clément C., Poulet R.</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>mRNA</th>
<th>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..&gt;1266)</th>
<th>/gene=&quot;SMN&quot;</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>/product=&quot;survival motor neuron&quot; /note=&quot;spinal muscular atrophy gene; cDNA sequence found in GenBank Accession Number U18423&quot;</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CDS</th>
<th>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)</th>
<th>/gene=&quot;SMN&quot;</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>/product=&quot;survival motor neuron&quot; /codon_start=1 /protein_id=&quot;AAC50473.1&quot; /db_xref=&quot;GI:1314346&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>/translation=&quot;MAKSDGS3QGVPBQEDSVLPQRTGSQSDSUWDDTDALIKYDKAVASFKHALKNGDICEQSKKEKTSQRPKPKNKSQKKNTASLQQWKGDKCSAIW SEDGCYEPATIASDIDFRSTCVVYTYGRGNYRSEQNLSDLISFICEVANNIEQEQNAGENE NESQYSDESQRSFQKSNQKPRSAEWSNFLPPPFPMPGRLGFKGKFLKFNGPP PPPPPPPPPLLSSCWPFFPSGPIIEPPPFPICPDSLDDADALSESLISWMSGYHTGY YMFGFRQNQKREGRCSEILN&quot;</td>
<td></td>
</tr>
</tbody>
</table>
### GenPept - A protein centered database

<table>
<thead>
<tr>
<th>LOCUS</th>
<th>AAC50473</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEFINITION</td>
<td>survival motor neuron.</td>
</tr>
<tr>
<td>ACCESSION</td>
<td>AAC50473</td>
</tr>
<tr>
<td>PID</td>
<td>gi1314346</td>
</tr>
<tr>
<td>VERSION</td>
<td>AAC50473.1 GI:1314346</td>
</tr>
<tr>
<td>DBSOURCE</td>
<td>locus HSSMNEUR1 accession U43876.1</td>
</tr>
<tr>
<td></td>
<td>locus HSSMNEUR2 accession U43877.1</td>
</tr>
<tr>
<td></td>
<td>locus HSSMNEUR3 accession U43878.1</td>
</tr>
<tr>
<td></td>
<td>locus HSSMNEUR4 accession U43879.1</td>
</tr>
<tr>
<td></td>
<td>locus HSSMNEUR5 accession U43880.1</td>
</tr>
<tr>
<td></td>
<td>locus HSSMNEUR6 accession U43881.1</td>
</tr>
<tr>
<td></td>
<td>locus HSSMNEUR7 accession U43882.1</td>
</tr>
<tr>
<td></td>
<td>locus HSSMNEUR8 accession U43883.1</td>
</tr>
<tr>
<td>KEYWORDS</td>
<td>.</td>
</tr>
<tr>
<td>SOURCE</td>
<td>human.</td>
</tr>
<tr>
<td>ORGANISM</td>
<td>Homo sapiens</td>
</tr>
<tr>
<td></td>
<td>Eukarya; Metazoa; Chordata; Craniata; Vertebrata; Euteleostomi; Mammalia; Eutheria; Primates; Catarrhini; Hominidae; Homo.</td>
</tr>
<tr>
<td>REFERENCE</td>
<td>1 (residues 1 to 294)</td>
</tr>
<tr>
<td>AUTHORS</td>
<td>Burglen,L., Lefebvre,S., Clermont,O., Burlet,P., Viollet,L., Cruaud,C., Munnich,A. and Melki,J.</td>
</tr>
<tr>
<td>TITLE</td>
<td>Structure and organization of the human survival motor neurone (SMN) gene</td>
</tr>
<tr>
<td>JOURNAL</td>
<td>Genomics 32 (3), 479-482 (1996)</td>
</tr>
<tr>
<td>MEDLINE</td>
<td>96435930</td>
</tr>
<tr>
<td>PUBMED</td>
<td>8838816</td>
</tr>
<tr>
<td>REFERENCE</td>
<td>2 (residues 1 to 294)</td>
</tr>
<tr>
<td>AUTHORS</td>
<td>Burglen L.</td>
</tr>
</tbody>
</table>
FTP sites:


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 ...)

Public Database - 1
UniGene
a non-redundant set of gene-oriented clusters

GenBank mRNAs
GenBank genomic CDSs
dbEST ESTs

Preliminary clusters

Unanchored cluster

Unigene
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
LOCUSID: 20595
LOCUS_CONFIRMED: yes
LOCUS_TYPE: gene with protein product, function known or inferred
ORGANISM: Mus musculus
STATUS: PROVISIONAL
NM: NM_011420|6755579
NP: NP_035550|6755580
CDD: Tudor domain|TUDOR|130|na|5.504980e+01
PRODUCT: survival motor neuron
ASSEMBLY: U77714
ACCNUM: AF131205|5932000
TYPE: g
PROT: AAD56757|5932001
ACCNUM: AF240501|8895972
TYPE: g
PROT: AAF81197|8895975
ACCNUM: AF240503|8895974
TYPE: g
PROT: AAF81199|8895979
ACCNUM: U92641|4335791
TYPE: g
PROT: AAC53057|1857113
ACCNUM: U77714|1946332
TYPE: m
PROT: AAC53144|1946333
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)
Gene Ontology

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

http://www.geneontology.org
GO Example 1:

Biological Process

Gene Ontology  (Human Genes) {Mouse Genes}
- Biological Process
  + behavior (16) {2}
  | biological_process unknown (5)
  - cell communication (7) {19}
    + cell adhesion (202) {201}
      | cell adhesion inhibition (5)
      + cell-cell matrix adhesion (25) {23}
      | flocculation
      + heterophilic cell adhesion (2)
      | homophilic cell adhesion (10) {21}
    + cell recognition (4)
    + cell-cell signaling (277) {35}
    + signal transduction (848) {177}
    + cell growth and maintenance (61) {154}
  + death
  + developmental processes (205) {94}
    + perception of external stimulus
    + physiological processes (7)
      + viral life cycle (5)
  + Cellular Component
  + Molecular Function
GO Example 2:

Molecular Function

nucleic acid binding (7) (170)
- DNA binding (260) (868)
  - AT DNA binding (3)
  - DNA bending (1)
  + DNA helicase (13) (53)
  + DNA repair protein (9) (19)
  + DNA replication factor (4) (1)
  - DNA secondary structure binding
  - DNA supercoiling
  - P-element binding (1)
  - bent DNA binding
  + chromatin binding (11) (11)
  - damaged DNA binding (7)
  + double-stranded DNA binding (15)
  - left-handed Z-DNA binding
  - plasmid-associated protein
  - random coil DNA binding
  - ribosomal DNA (rDNA) binding
  - satellite DNA binding (3)
  - single-stranded DNA binding (21) (3)
  + telomerase (1)
  - transcription factor (397) (438)
    - RNA polymerase I transcription factor (5)
    + RNA polymerase II transcription factor (137) (12)
    - RNA polymerase III transcription factor (9)
    - transcription activating factor (114)
    + transcription elongation factor (5)
    + transcription termination factor (1)
    + RNA binding (196) (120)
**Gene Ontology Annotation**

**Smn:** survival motor neuron  
**LocusID:** 20595

<table>
<thead>
<tr>
<th>Gene Ontology™:</th>
<th>Evidence</th>
<th>Source</th>
<th>Pub</th>
</tr>
</thead>
<tbody>
<tr>
<td>nucleus</td>
<td>IDA</td>
<td>MGD</td>
<td>pm</td>
</tr>
<tr>
<td>nucleus</td>
<td>IEA</td>
<td>MGD</td>
<td>pm</td>
</tr>
<tr>
<td>cytoplasm</td>
<td>IDA</td>
<td>MGD</td>
<td>pm</td>
</tr>
<tr>
<td>RNA binding</td>
<td>IEA</td>
<td>MGD</td>
<td>pm</td>
</tr>
<tr>
<td>mRNA processing</td>
<td>IEA</td>
<td>MGD</td>
<td>pm</td>
</tr>
<tr>
<td>nucleic acid binding</td>
<td>IEA</td>
<td>MGD</td>
<td>pm</td>
</tr>
</tbody>
</table>
The Pfam and SMART database of protein sequence profile

• PFAM 3849 protein families (v. 7.3)
• SMART 500 extensively annotated domain families
## Pfam Examples:

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

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

**Accession**: For A1BG, a2m
From Excel spreadsheet to relational database

<table>
<thead>
<tr>
<th>LocusLink ID</th>
<th>Symbol</th>
<th>Full Name</th>
<th>Species</th>
<th>Accession</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A1BG</td>
<td>alpha-1-B glycoprotein</td>
<td>Human</td>
<td>AA484435</td>
</tr>
<tr>
<td>1</td>
<td>A1BG</td>
<td>alpha-1-B glycoprotein</td>
<td>Human</td>
<td>W25099</td>
</tr>
<tr>
<td>1</td>
<td>A1BG</td>
<td>alpha-1-B glycoprotein</td>
<td>Human</td>
<td>AI022193</td>
</tr>
<tr>
<td>1</td>
<td>A1BG</td>
<td>alpha-1-B glycoprotein</td>
<td>Human</td>
<td>T80683</td>
</tr>
<tr>
<td>2</td>
<td>A2M</td>
<td>alpha-2-macroglobulin</td>
<td>Human</td>
<td>X68728</td>
</tr>
<tr>
<td>2</td>
<td>A2M</td>
<td>alpha-2-macroglobulin</td>
<td>Human</td>
<td>NM_000001</td>
</tr>
<tr>
<td>2</td>
<td>A2M</td>
<td>alpha-2-macroglobulin</td>
<td>Human</td>
<td>M36501</td>
</tr>
<tr>
<td>2</td>
<td>A2M</td>
<td>alpha-2-macroglobulin</td>
<td>Human</td>
<td>M11313</td>
</tr>
<tr>
<td>3</td>
<td>A2MP</td>
<td>alpha-2-macroglobulin pseudogene</td>
<td>Human</td>
<td>M24415</td>
</tr>
<tr>
<td>9</td>
<td>NAT1</td>
<td>N-acetyltransferase 1</td>
<td>Human</td>
<td>NM_000666</td>
</tr>
<tr>
<td>9</td>
<td>NAT1</td>
<td>N-acetyltransferase 1</td>
<td>Human</td>
<td>AF071552</td>
</tr>
<tr>
<td>9</td>
<td>NAT1</td>
<td>N-acetyltransferase 1</td>
<td>Human</td>
<td>U80835</td>
</tr>
</tbody>
</table>

A1BG  a2m
From Excel spreadsheet to relational database

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

<table>
<thead>
<tr>
<th>LocusLink ID</th>
<th>Accession</th>
<th>GO ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AA484435</td>
<td>6886</td>
</tr>
<tr>
<td>1</td>
<td>W25099</td>
<td>4866</td>
</tr>
<tr>
<td>1</td>
<td>A1022193</td>
<td>8320</td>
</tr>
<tr>
<td>1</td>
<td>T80683</td>
<td>5737</td>
</tr>
<tr>
<td>2</td>
<td>X68728</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>NM_00014</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>M36501</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>M11313</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>M24415</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>NM_000662</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>AF071552</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>U80835</td>
<td></td>
</tr>
</tbody>
</table>

A1BG  a2m
A simple schema for microarray database

- **Sample**
  - Sample ID
  - Date
  - Comments

- **Gene**
  - Gene ID
  - Accession
  - Annotation

- **Expression Data**
  - Sample ID
  - Gene ID
  - Expr Level
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

<table>
<thead>
<tr>
<th>Sample1</th>
<th>Sample2</th>
<th>Sample3</th>
<th>Sample17</th>
</tr>
</thead>
<tbody>
<tr>
<td>GeneID</td>
<td>GeneID</td>
<td>GeneID</td>
<td>GeneID</td>
</tr>
<tr>
<td>Expr Level</td>
<td>Expr Level</td>
<td>Expr Level</td>
<td>Expr Level</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Gene</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gene ID</td>
</tr>
<tr>
<td>Accession</td>
</tr>
<tr>
<td>Annotation</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Expression Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sample ID</td>
</tr>
<tr>
<td>Gene ID</td>
</tr>
<tr>
<td>Expr Level</td>
</tr>
</tbody>
</table>
**Schema 2:** Too many columns, and column structure not stable

<table>
<thead>
<tr>
<th>Expr Data</th>
<th>Expression Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gene ID</td>
<td>Sample ID</td>
</tr>
<tr>
<td>Sample 1</td>
<td>Gene ID</td>
</tr>
<tr>
<td>Sample 2</td>
<td>Expr Level</td>
</tr>
<tr>
<td>Sample 3</td>
<td></td>
</tr>
<tr>
<td>Sample 4</td>
<td></td>
</tr>
<tr>
<td>Sample 5</td>
<td></td>
</tr>
<tr>
<td>Sample 6</td>
<td></td>
</tr>
<tr>
<td>Sample 7</td>
<td></td>
</tr>
<tr>
<td>Sample 8</td>
<td></td>
</tr>
<tr>
<td>Sample 9</td>
<td></td>
</tr>
<tr>
<td>Sample 10</td>
<td></td>
</tr>
<tr>
<td>Sample 11</td>
<td></td>
</tr>
<tr>
<td>Sample 12</td>
<td></td>
</tr>
<tr>
<td>Sample 13</td>
<td></td>
</tr>
<tr>
<td>Sample 14</td>
<td></td>
</tr>
<tr>
<td>Sample 15</td>
<td></td>
</tr>
</tbody>
</table>
Data Relationships:

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

<table>
<thead>
<tr>
<th>Pubmed ID</th>
<th>Journal ID</th>
<th>Vol</th>
<th>Page</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>11983181</td>
<td>3</td>
<td>9</td>
<td>911</td>
<td>A caspase-related protease ...</td>
</tr>
<tr>
<td>11983180</td>
<td>3</td>
<td>9</td>
<td>903</td>
<td>A Mechanism for Microtubule ...</td>
</tr>
<tr>
<td>11983179</td>
<td>3</td>
<td>9</td>
<td>891</td>
<td>Hypermethylation of the Cap ...</td>
</tr>
<tr>
<td>11983178</td>
<td>3</td>
<td>9</td>
<td>879</td>
<td>Inhibition of reverse transcription ...</td>
</tr>
</tbody>
</table>

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

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

<table>
<thead>
<tr>
<th>ID</th>
<th>Journal Name</th>
<th>Abbreviation</th>
<th>ISSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Molecular and Cellular Biology</td>
<td>Mol Cell Biol</td>
<td>0270-7306</td>
</tr>
<tr>
<td>2</td>
<td>Molecular Biology of the Cell</td>
<td>Mol Biol Cell</td>
<td>1059-1524</td>
</tr>
<tr>
<td>3</td>
<td>Molecular Cell</td>
<td>Mol Cell</td>
<td>1097-2765</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Pubmed ID</th>
<th>Journal ID</th>
<th>Vol</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>11983181</td>
<td>3</td>
<td>9</td>
<td>911</td>
</tr>
<tr>
<td>11983180</td>
<td>3</td>
<td>9</td>
<td>903</td>
</tr>
<tr>
<td>11983179</td>
<td>3</td>
<td>9</td>
<td>891</td>
</tr>
<tr>
<td>11983178</td>
<td>3</td>
<td>9</td>
<td>879</td>
</tr>
<tr>
<td>12024052</td>
<td>1</td>
<td>22</td>
<td>4433</td>
</tr>
<tr>
<td>12024051</td>
<td>1</td>
<td>22</td>
<td>4419</td>
</tr>
<tr>
<td>12024050</td>
<td>1</td>
<td>22</td>
<td>4402</td>
</tr>
<tr>
<td>12006669</td>
<td>2</td>
<td>13</td>
<td>1788</td>
</tr>
<tr>
<td>12006670</td>
<td>2</td>
<td>13</td>
<td>1792</td>
</tr>
</tbody>
</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)
<table>
<thead>
<tr>
<th>Parent</th>
<th>Child</th>
</tr>
</thead>
<tbody>
<tr>
<td>Biological Process</td>
<td>Behavior</td>
</tr>
<tr>
<td>Biological Process</td>
<td>Cellular Communication</td>
</tr>
<tr>
<td>Biological Process</td>
<td>Death</td>
</tr>
<tr>
<td>Biological Process</td>
<td>Developmental Process</td>
</tr>
<tr>
<td>Cellular Communication</td>
<td>Cell Adhesion Inhibition</td>
</tr>
<tr>
<td>Cellular Communication</td>
<td>Homophilic adhesion</td>
</tr>
</tbody>
</table>

Recursive

Gene Ontology (Human Genes) (Mouse Genes)

- Biological Process
  - behavior (16) [2]
  - biological_process unknown (5)
  - cell communication (7) [19]
    - cell adhesion (202) [201]
      - cell adhesion inhibition (5)
      - cell-cell matrix adhesion (25) [23]
      - flocculation
      - heterophilic cell adhesion (2)
      - homophilic cell adhesion (10) [21]
    - cell recognition (4)
    - cell-cell signaling (277) [35]
    - signal transduction (848) [177]
    - cell growth and maintenance (61) [154]
    - death
      - developmental processes (205) [94]
      - perception of external stimulus
      - physiological processes (7)
      - viral life cycle (5)
    - Cellular Component
    - Molecular Function
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**  
   - "atgcaaatttgccccc ........ 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:

```sql
select <column name>
from <table name>
where <condition>
```

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

Use bracket for table and column names
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

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Data Type</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>int</td>
<td>4</td>
</tr>
<tr>
<td>gene_id</td>
<td>int</td>
<td>4</td>
</tr>
<tr>
<td>subjname</td>
<td>varchar</td>
<td>2000</td>
</tr>
<tr>
<td>querylen</td>
<td>int</td>
<td>4</td>
</tr>
<tr>
<td>subjlen</td>
<td>int</td>
<td>4</td>
</tr>
<tr>
<td>rank</td>
<td>int</td>
<td>4</td>
</tr>
<tr>
<td>[identity]</td>
<td>int</td>
<td>4</td>
</tr>
<tr>
<td>positive</td>
<td>int</td>
<td>4</td>
</tr>
<tr>
<td>matchlen</td>
<td>int</td>
<td>4</td>
</tr>
<tr>
<td>score</td>
<td>decimal</td>
<td>9</td>
</tr>
<tr>
<td>evalue</td>
<td>real</td>
<td>4</td>
</tr>
<tr>
<td>querystart</td>
<td>int</td>
<td>4</td>
</tr>
<tr>
<td>queryend</td>
<td>int</td>
<td>4</td>
</tr>
<tr>
<td>querystrand</td>
<td>smallint</td>
<td>2</td>
</tr>
<tr>
<td>subjstart</td>
<td>int</td>
<td>4</td>
</tr>
<tr>
<td>subjend</td>
<td>int</td>
<td>4</td>
</tr>
<tr>
<td>subjstrand</td>
<td>smallint</td>
<td>2</td>
</tr>
</tbody>
</table>
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
Self-join: To compare data on different rows in the same table

```
<table>
<thead>
<tr>
<th>Column Name</th>
<th>Data Type</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>sample_id</td>
<td>int</td>
<td>4</td>
</tr>
<tr>
<td>gene_id</td>
<td>int</td>
<td>4</td>
</tr>
<tr>
<td>intensity</td>
<td>decimal</td>
<td>9</td>
</tr>
</tbody>
</table>
```

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

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

2. **max**
   
   ```sql
   select max (intensity) from expr_data
   ```

3. **distinct**
   
   ```sql
   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.
msg = "Hello World"
print msg
exit
For $i=1$ to 100000
  print "Are we there yet?"
  if arrive() exit
Next
Programming for dummies -- Level 3

Reading and Writing
ile
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](http://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

<table>
<thead>
<tr>
<th>ID</th>
<th>Exp. Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>26</td>
<td>2198</td>
</tr>
<tr>
<td>27</td>
<td>1384</td>
</tr>
<tr>
<td>28</td>
<td>1284</td>
</tr>
<tr>
<td>29</td>
<td>983</td>
</tr>
<tr>
<td>30</td>
<td>4594</td>
</tr>
<tr>
<td>31</td>
<td>2761</td>
</tr>
<tr>
<td>33</td>
<td>3287</td>
</tr>
<tr>
<td>34</td>
<td>1287</td>
</tr>
<tr>
<td>495</td>
<td>2823</td>
</tr>
<tr>
<td>496</td>
<td>799</td>
</tr>
<tr>
<td>497</td>
<td>1078</td>
</tr>
<tr>
<td>498</td>
<td>1203</td>
</tr>
<tr>
<td>499</td>
<td>6688</td>
</tr>
<tr>
<td>500</td>
<td>1697</td>
</tr>
</tbody>
</table>

**mean:** \( \frac{(2198 + 1384 + \ldots + 1697)}{446} \)

**median:** 146 185 188 189 ... 1648 1650 1652 ... 17409
What does a PERL program look like?

```perl
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

- **Tab**: `\t` in PERL
- **New Line**: `\n` in PERL

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>26</td>
<td>2198</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>1384</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>1284</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>983</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>4594</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>2761</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>3287</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>4787</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In PERL, `\t` is TAB, `\n` is new line.
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

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

Do Something

Done?

Yes

Exit

No

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:

```perl
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:

```perl
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:

```perl
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 = "atgcccgctgctggaatgc";

**An object:**
$seq = Bio::PrimarySeq->new (-seq => 'atgcccgctgctggaatgc');
A regular string:
```perl
$seq = "atgcccgtgctggaatgc";
print $seq;
```

An object:
```perl
$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
**SeqIO class:**

**Constructor:**

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

**Method:**

```perl
$in -> next_seq;
$out-> write_seq($seq);
```
Example of SeqIO:

```perl
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 complementation
• restriction sites
• truncation and mutation
• ...

Seq Class

Constructor:

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

Methods:

```perl
$seqobj ->seq();
$seqobj ->length();
$seqobj ->translate();
$seqobj ->revcom();
$seqobj ->subseq(1, 10);
```
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
$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(['J00522 AF303112 2981014']));
Commonly used BioPERL objects 4: BPlite

Usage:

For parsing standard blast output
Example of BPlite:

```perl
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");
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:


3. Object Oriented Perl by Damian Conway