Session 3 Outline

- SQL query review
- Creating databases
- Creating tables
- Altering table structure
- Inserting data
- Deleting data
- Updating/modifying data
- Automating repetitive tasks

SELECT

```
SELECT *
FROM Data
LIMIT 5;
```

```
# Comments after '#'
SELECT DISTINCT species
FROM LocusDescr;
```

```
+-----------------+----------+-------+
| affyId | exptId  | level |
|---------+---------+-------|
| AFFX-MurIL2_at | hs-cer-1 |    20 |
| AFFX-MurIL10_at | hs-cer-1 |     8 |
| AFFX-MurIL4_at | hs-cer-1 |    77 |
| AFFX-MurFAS_at | hs-cer-1 |    30 |
| AFFX-BioB-5_at | hs-cer-1 |   258 |
```

```
+---------+
<table>
<thead>
<tr>
<th>species</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hs</td>
</tr>
<tr>
<td>Mm</td>
</tr>
</tbody>
</table>
```

```
WHERE And ORDER BY
```

```
SELECT *
FROM RefSeqs
WHERE linkId BETWEEN 50 AND 100
LIMIT 5;
```

```
SELECT *
FROM RefSeqs
WHERE linkId BETWEEN 50 AND 100
ORDER BY ntRefSeq DESC
LIMIT 5;
```

```
+--------+-----------+-----------+
<table>
<thead>
<tr>
<th>linkId</th>
<th>ntRefSeq</th>
<th>aaRefSeq</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>NM_001098</td>
<td>NP_001089</td>
</tr>
<tr>
<td>51</td>
<td>NM_004035</td>
<td>NP_004026</td>
</tr>
<tr>
<td>52</td>
<td>NM_004300</td>
<td>NP_004291</td>
</tr>
<tr>
<td>53</td>
<td>NM_001610</td>
<td>NP_001601</td>
</tr>
<tr>
<td>54</td>
<td>NM_001611</td>
<td>NP_001602</td>
</tr>
</tbody>
</table>
```

```
+--------+-----------+-----------+
<table>
<thead>
<tr>
<th>linkId</th>
<th>ntRefSeq</th>
<th>aaRefSeq</th>
</tr>
</thead>
<tbody>
<tr>
<td>70</td>
<td>NM_005159</td>
<td>NP_005150</td>
</tr>
<tr>
<td>81</td>
<td>NM_004924</td>
<td>NP_004915</td>
</tr>
<tr>
<td>91</td>
<td>NM_004302</td>
<td>NP_004293</td>
</tr>
<tr>
<td>86</td>
<td>NM_004301</td>
<td>NP_004292</td>
</tr>
<tr>
<td>52</td>
<td>NM_004300</td>
<td>NP_004291</td>
</tr>
</tbody>
</table>
```

```
GROUP BY And HAVING
```

```
SELECT affyId, MIN(level) as min,
MAX(level) as max
FROM Data
GROUP BY affyId
HAVING max - min > 5000
LIMIT 5;
```

```
+-------------+------+-------+
<table>
<thead>
<tr>
<th>affyId</th>
<th>min</th>
<th>max</th>
</tr>
</thead>
<tbody>
<tr>
<td>100047_at</td>
<td>20</td>
<td>7784</td>
</tr>
<tr>
<td>100057_at</td>
<td>778</td>
<td>7784</td>
</tr>
<tr>
<td>100058_at</td>
<td>777</td>
<td>7779</td>
</tr>
<tr>
<td>100059_at</td>
<td>778</td>
<td>7784</td>
</tr>
<tr>
<td>100060_at</td>
<td>776</td>
<td>7779</td>
</tr>
</tbody>
</table>
```

```
+----------+-------------+
<table>
<thead>
<tr>
<th>gbId</th>
<th>num_affyIds</th>
</tr>
</thead>
<tbody>
<tr>
<td>J04423</td>
<td>14</td>
</tr>
<tr>
<td>AC002397</td>
<td>9</td>
</tr>
<tr>
<td>AF049905</td>
<td>9</td>
</tr>
<tr>
<td>AL031228</td>
<td>8</td>
</tr>
</tbody>
</table>
```

```
WHERE And ORDER BY
```

```
SELECT gbId, count(affyId) AS num_affyIds
FROM Targets
GROUP BY gbId
HAVING COUNT(gbId) > 4
ORDER BY num_affyIds DESC
LIMIT 5;
```

```
+----------+-------------+
<table>
<thead>
<tr>
<th>gbId</th>
<th>num_affyIds</th>
</tr>
</thead>
<tbody>
<tr>
<td>100047_at</td>
<td>14</td>
</tr>
<tr>
<td>100057_at</td>
<td>14</td>
</tr>
<tr>
<td>100058_at</td>
<td>14</td>
</tr>
<tr>
<td>100059_at</td>
<td>14</td>
</tr>
<tr>
<td>100060_at</td>
<td>14</td>
</tr>
</tbody>
</table>
```

```
GROUP BY And HAVING
```

```
> SELECT affyId, MIN(level) as min,
MAX(level) as max
FROM Data
GROUP BY affyId
HAVING max - min > 5000
LIMIT 5;
```

```
GROUP BY gbId
HAVING count(affyId) > 4
ORDER BY num_affyIds DESC
LIMIT 5;
```

Table Joining

```
> SELECT DISTINCT Unigenes.udl, GO_Descr.description AS GO_description
FROM Unigenes, LocusLinks, Ontologies, GO_Descr
WHERE Unigenes.linkId=LocusLinks.linkId
AND LocusLinks.linkId=Ontologies.linkId
AND Ontologies.goAcc=GO_Descr.goAcc
ORDER BY num_affyIds DESC
LIMIT 5;
```

```
+-----------+-------------------------------+
<table>
<thead>
<tr>
<th>udl</th>
<th>GO_description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hs.373554</td>
<td>calcium ion binding</td>
</tr>
<tr>
<td>Hs.74561</td>
<td>protein carrier</td>
</tr>
<tr>
<td>Hs.155396</td>
<td>arylamine N-acetyltransferase</td>
</tr>
<tr>
<td>Hs.2</td>
<td>arylamine N-acetyltransferase</td>
</tr>
<tr>
<td>Hs.234726</td>
<td>serine protease inhibitor</td>
</tr>
</tbody>
</table>
```

```
> SELECT gbId, count(affyId) AS num_affyIds
FROM Targets
GROUP BY gbId
HAVING COUNT(gbId) > 4
ORDER BY num_affyIds DESC
LIMIT 5;
```

```
**Output Formats**

- Query from MySQL prompt
- Ending query with `\G`
- `mysql < q.sql` to output

```
| gbId  | num_affyIds |
|-------+-------------|
| J04423   |          14 |
| AC002397 |          12 |
| AF109905 |           9 |
| AF100956 |           9 |
| AL031228 |           8 |
```

**Access Privileges**

- Restrict access and prevent accidental alteration of important information
- Can limit what individual users can see and do on particular databases and specific tables
- Access privileges are stored in the "mysql" database

```
> GRANT ALL PRIVILEGES ON db4bio.* TO superuser@"%" IDENTIFIED BY "password";
> GRANT SELECT,INSERT ON db4bio.Data TO admin@"18.157.1.1" IDENTIFIED BY "pass2";
```

**CREATE DATABASE**

- Allows you to create a new database on the database server (if you have permission)

```
> SHOW DATABASES;
> CREATE DATABASE go;
> SHOW DATABASES;
> USE go;
```

**CREATE TABLE**

- Translate an E-R diagram (schema) into a functioning database

```
CREATE TABLE Descriptions (
    gbId VARCHAR(20) NOT NULL,
    description VARCHAR(100),
    PRIMARY KEY (gbId)
);
```

```
<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Null</th>
<th>Key</th>
<th>Default</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td>gbId</td>
<td>varchar(20)</td>
<td></td>
<td>PRI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>description</td>
<td>varchar(100)</td>
<td>YES</td>
<td></td>
<td>NULL</td>
<td></td>
</tr>
</tbody>
</table>
```

```
CREATE TABLE Targets (
    affyId VARCHAR(20) NOT NULL,
    gbId VARCHAR(20) NOT NULL,
    species VARCHAR(20),
    PRIMARY KEY (affyId, gbId)
);
```

```
<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Null</th>
<th>Key</th>
<th>Default</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td>affyId</td>
<td>varchar(20)</td>
<td></td>
<td>PRI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>gbId</td>
<td>varchar(20)</td>
<td></td>
<td>PRI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>species</td>
<td>varchar(20)</td>
<td>YES</td>
<td></td>
<td>NULL</td>
<td></td>
</tr>
</tbody>
</table>
```

**ALTER TABLE**

- Modify a table’s attributes
  - Attribute names, type, null, key, default
  - Add or drop attributes

```
> ALTER TABLE Data
CHANGE level level DOUBLE;
> ALTER TABLE Data
RENAME level expression;
> ALTER TABLE Data
ADD PRIMARY KEY (expId);
> DROP TABLE Data;
```
INSERT INTO

- Finally, add data into tables

> INSERT INTO Data (level, exptId, affyId) VALUES (215, "hs-hrt-1", "100008_at");

DELETE FROM

- Delete data from tables
- Similar syntax as SELECT

> DELETE FROM Data WHERE exptId='hs-hrt-1';

UPDATE

- Modify data already stored in a table
- Again, similar syntax as SELECT

> UPDATE Data MODIFY
SET exptId="hs-hrt-2"
WHERE exptId="hs-hrt-1";

> UPDATE Source FIX
SET exptId="ms-hrt-1", source="Mm"
WHERE exptId="hs-hrt-1";

> UPDATE Data INTERNAL
SET level=level*1.27 "NORMALIZATION"
WHERE exptId="hs-hrt-1";

LOAD DATA And Export

- Read rows from a text file (in the current directory) into a table and vice versa

> LOAD DATA LOCAL INFILE "data.txt"
INTO TABLE db4bio.Data
FIELDS TERMINATED BY '	'
LINES TERMINATED BY '
';

> LOAD DATA LOCAL INFILE "data.txt"
INTO TABLE db4bio.Data;

> SELECT * INTO OUTFILE "data.txt"
FIELDS TERMINATED BY ','
FROM Data;

LOAD DATA warnings

mysql> LOAD DATA LOCAL INFILE "Hs_sources_test.txt"
-> INTO TABLE Sources;
Query OK, 4 rows affected, 3 warnings (0.00 sec)
Records: 4 Deleted: 0 Skipped: 0 Warnings: 3

3 rows in set (0.00 sec)

LOAD DATA LOCAL INFILE "Hs_sources_test.txt"
-> INTO TABLE Sources;
Query OK, 4 rows affected, 3 warnings (0.00 sec)
Records: 4 Deleted: 0 Skipped: 0 Warnings: 3

Automating Repetitive Tasks

- Use .SQL files to perform SQL commands automatically
- Automatically create a series of tables

% mysql -h hebrides.wi.mit.edu -u guest -p -D databasename < create.sql

- Feed a complicated query to the database and receive the results in a text file

% mysql -h hebrides.wi.mit.edu -u web -p -D db4bio < query1.sql > query1.out
Summary

- Design databases with E-R diagrams
- Data mine using combinations of SELECT/FROM with WHERE, GROUP BY, HAVING, ORDER BY, and aggregates
- Create and implement databases
- Input and output data from databases
- Modify existing data within databases

Advanced topics

- Query optimization (adding indexes)
- Dates and times
  - all expected functionality
- Mathematics functions: logs, trig, etc.
- “String” (text) functions
  - substring, concatenate, replace, case change, etc.
- Nested queries
  - SELECT * FROM Ontologies WHERE linkId IN (SELECT linkId FROM LocusLinks WHERE gbId LIKE "A82%")

Where To Go From Here?

- Consult SQL And MySQL Resources
  - http://www.mysql.com
  - Tutorial, Reference Manual
- Graphical interfaces to MySQL
  - DBDesigner (free)
  - MySQL Administrator
  - SQL4XManagerJ (inexpensive)
  - Visio (Microsoft)
  - Visual Case (expensive)
- Ensembl databases with open access
- Sources of data to build your own:
  - UCSC Bioinformatics; Gene Ontology; Entrez Gene

Course Goals

- Conceptualize data in terms of relations (database tables)
- Design relational databases
- Use SQL commands to extract data from (mine) databases
- Use SQL commands to build and modify databases

Exercises

- Create tables
- Input data
- Modify/delete particular data

- Accessing your own database:
  mysql- **username** -p **Username**
  - hebrides.wi.mit.edu