MySQL toolbox

- MySQL in your swiss-army knife
- Designing your database
- Tables, Columns, Data types, Keys & Relations
- Inserting data
- Query data
- Use Perl DBI
- Make Perl API
- Just the basics
- Exercises < Theory & Examples
MySQL in your swiss-army knife

- What is a *relational* database
  - “A relational database is a big spreadsheet several people can update simultaneously” (several sources)
- Why use a relational database
  - Make *queries* in your data
  - Index data for *fast* retrieval
  - Build *programmatic* interface to data
  - Eliminate *duplicate* information
- Database methodology
  - Transactional
  - Datawarehouse
Designing your database

• Understand what *entities* your data contains
• Understand the entities *attributes*
• Understand the entities *relations*
• Build your *model* from this knowledge
• Implement *tables* representing the entities
• Add *columns* to tables representing the entity attributes
• Use *primary* and *foreign keys* in tables representing the relations between entities
• Be careful not to over-generalize
• Use *normalization* where appropriate
• Perhaps use design tools such as mysql workbench
  • http://dev.mysql.com/downloads/workbench/
Entities

- Abstractions from the complexities of some domain (wikipedia)
- The objects, locations, events, concepts about which data is stored.
  - Shop, Manager, Employee
  - Plane, Passenger, Reservation, Flight
  - Doctor, Ward, Patient, Treatment
  - Gene, mRNA, Protein
  - Book, Subject, Author, Publisher
Attributes

• The simple data belonging to the entities in your model.

• Patient
  •
  •
  •

• Protein
  •
  •
  •
Relations

• Relations between entities,
  • An mRNA is a transcription of a Gene
  • Protein is a translation of an mRNA
• One-to-one
  • Only one matching record
  • Uses same primary key for both entities
  • Used for object inheritance
• One-to-many
  • Most common type of relationship
  • Relation between primary and foreign keys
  • One record can have many related records
• Many-to-many
  • Implemented with junction table
  • Implemented with many-to-many
Database model example

- A database of books in a bookstore.
  - Title
  - Author
  - Author Biography
  - ISBN
  - Price
  - Subject
  - Number of Pages
  - Publisher
  - Publisher Address
## As a spreadsheet

<table>
<thead>
<tr>
<th>Title</th>
<th>Author</th>
<th>Bio</th>
<th>ISBN</th>
<th>Price</th>
<th>Subject</th>
<th>Publisher</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Impact of Bayesian Technology on Networking</td>
<td>Dr. John d'Eaux</td>
<td>Dr. John d'Eaux is a leading expert on the use of Bayesian technologies in network analysis.</td>
<td>3639159853</td>
<td>82.00</td>
<td>Computer Networks, Simulation</td>
<td>VDM Verlag</td>
<td><a href="http://www.vdm-publishing.com/">http://www.vdm-publishing.com/</a></td>
</tr>
<tr>
<td>Dragons of summer flame</td>
<td>Margaret Weis, Tracy Hickman</td>
<td>Margaret Weis was born on the 16th March 1948 in Independence, Missouri and is a New York Times bestselling author of fantasy books. Tracy and Laura Hickman have been publishing game designs and stories together for over thirty two years - nearly as long as their marriage - and thus started them both on a life of adventure and imagination.</td>
<td>0786927089</td>
<td>7.99</td>
<td>Fantasy</td>
<td>Wizards of the Coast</td>
<td><a href="http://www..wizards.com/">http://www..wizards.com/</a></td>
</tr>
<tr>
<td>On the Origin of Species</td>
<td>Charles Darwin</td>
<td>English Naturalist and author who established the theory of evolution, published his findings in On the Origin of Species By Means of Natural Selection; or, the Preservation of Favoured Races in the Struggle for Life (1859)</td>
<td>&quot;</td>
<td>5.00</td>
<td>Evolution, Biology</td>
<td>Dover Publications</td>
<td><a href="http://www.doverpublications.com/">http://www.doverpublications.com/</a></td>
</tr>
</tbody>
</table>
Locating the entities
• Tables, Columns, Data types, Keys & Relations

• Tables
• Columns
• Data types
• Keys
  • Primary key
  • Foreign key
  • Index
• Relations

• CREATE DATABASE statement
• CREATE TABLE statement
• SHOW CREATE TABLE statement
• DROP TABLE statement
CREATE DATABASE statement

CREATE {DATABASE | SCHEMA} [IF NOT EXISTS] db_name

- log into your account on the system, and attempt to connect to the mysql server. Simply write 'mysql' on the command prompt, and it will connect you.
- Issue a create database statement, creating the database schema with the name of your user followed by '_private' e.g.

```sql
mysql> CREATE DATABASE stud088_private;
```
CREATE TABLE [IF NOT EXISTS] tbl_name

(create_definition,...)

create_definition:
  col_name column_definition
  | PRIMARY KEY (index_col_name,...)
  | {INDEX|KEY} [index_name] (index_col_name,...)
  | UNIQUE [INDEX|KEY] (index_col_name,...)

column_definition:
  data_type [NOT NULL | NULL] [DEFAULT default_value]
  [AUTO_INCREMENT] [UNIQUE [KEY] | [PRIMARY] KEY]

data_type:
  INTEGER[(length)]
  | DOUBLE[(length,decimals)]
  | DATE
  | TIME
  | DATETIME
  | CHAR[(length)]
  | TEXT
  | ENUM(value1,value2,value3,...)
  | SET(value1,value2,value3,...)

(this is a format subset. For full definition, see http://dev.mysql.com/doc/)
(primary) Datatypes

- INTEGER
- DOUBLE
- DATE
- TIME
- DATETIME
- CHAR(length)
- TEXT
- ENUM('one','two','three' ... )
- SET('one','two','three' ... )
  - 'one,two'
  - FIND_IN_SET('two',column)
CREATE TABLE IF NOT EXISTS Book (  
    ISBN BIGINT NOT NULL,  
    title VARCHAR(128) NULL,  
    pages INT NULL,  
    price DOUBLE NULL,  
    publish DATE NULL,  
    publisher_idPublisher INT NOT NULL,  
    PRIMARY KEY (ISBN),  
    INDEX (Publisher_idPublish)  
);
SHOW CREATE TABLE statement

SHOW CREATE TABLE tbl_name

mysql> SHOW CREATE TABLE \G

*************************** 1. row ***************************
Table: Book
Create Table: CREATE TABLE `Book` (  
`ISBN` bigint(20) NOT NULL,  
`title` varchar(128) default NULL,  
`pages` int(11) default NULL,  
`price` double default NULL,  
`publish` date default NULL,  
`Publisher_idPublisher` int(11) NOT NULL,  
PRIMARY KEY (`ISBN`),  
KEY `Publisher_idPublisher` (`Publisher_idPublisher`)  
) ENGINE=MyISAM DEFAULT CHARSET=latin1
1 row in set (0.00 sec)
• DROP TABLE statement

DROP TABLE [IF EXISTS] tbl_name [, tbl_name] ...

mysql> drop table Book;
Query OK, 0 rows affected (0.00 sec)
Making your DDL statements

CREATE TABLE IF NOT EXISTS Book (  
    ISBN BIGINT NOT NULL ,  
    title VARCHAR(128) NULL ,  
    pages INT NULL ,  
    price DOUBLE NULL ,  
    publish DATE NULL ,  
    Publisher_idPublisher INT NOT NULL ,  
    PRIMARY KEY (ISBN) ,  
    INDEX fk_Book_Publisher1 (Publisher_idPublisher ASC) )  
ENGINE = MyISAM;

CREATE TABLE IF NOT EXISTS Author (  
    idAuthor INT NOT NULL AUTO_INCREMENT ,  
    firstname VARCHAR(128) NOT NULL ,  
    lastname VARCHAR(128) NULL ,  
    biography TEXT NULL ,  
    PRIMARY KEY (idAuthor) ,  
    KEY name (firstname,lastname) )  
ENGINE = MyISAM;
Making your DDL statements

CREATE TABLE IF NOT EXISTS Subject (  idSubject INT NOT NULL AUTO_INCREMENT ,  name VARCHAR(128) NULL ,  PRIMARY KEY (idSubject) ,  UNIQUE KEY name (name) )  ENGINE = MyISAM;

CREATE TABLE IF NOT EXISTS Publisher (  idPublisher INT NOT NULL AUTO_INCREMENT ,  name VARCHAR(128) NULL ,  url VARCHAR(128) NULL ,  PRIMARY KEY (idPublisher) ,  UNIQUE KEY name (name) )  ENGINE = MyISAM;
Making your DDL statements

```sql
CREATE TABLE IF NOT EXISTS Book_has_Author (  
    Book_ISBN BIGINT NOT NULL ,  
    Author_idAuthor INT NOT NULL ,  
    PRIMARY KEY (Book_ISBN, Author_idAuthor) ,  
    INDEX fk_Book_has_Author_Book1 (Book_ISBN ASC) ,  
    INDEX fk_Book_has_Author_Author1 (Author_idAuthor ASC) 
);

CREATE TABLE IF NOT EXISTS Book_has_Subject (  
    Book_ISBN BIGINT NOT NULL ,  
    Subject_idSubject INT NOT NULL ,  
    PRIMARY KEY (Book_ISBN, Subject_idSubject) ,  
    INDEX fk_Book_has_Subject_Book (Book_ISBN ASC) ,  
    INDEX fk_Book_has_Subject_Subject1 (Subject_idSubject ASC) 
);
```
### SHOW TABLES statement

```sql
SHOW TABLES [{FROM | IN} db_name]
```

```
mysql> SHOW TABLES;
+----------------------------------+
| Tables_in_hhs_private            |
+----------------------------------+
| Author                           |
| Book                             |
| Book_has_Author                  |
| Book_has_Subject                 |
| Publisher                        |
| Subject                          |
+----------------------------------+
```

Query OK, 6 rows affected (0.00 sec)
Inserting data into tables

- **INSERT .. VALUES** statement
- **AUTO_INCREMENT / LAST_INSERT_ID()**
- **LOAD DATA LOCAL INFILE** statement
**INSERT statement**

```sql
INSERT [IGNORE]
    [INTO] tbl_name [(col_name,...)]
{VALUES | VALUE} ({expr | DEFAULT},...),(...),...
```

```sql
mysql> INSERT INTO Subject (name) VALUES ('Horror');
Query OK, 1 row affected (0.00 sec)

mysql> SELECT LAST_INSERT_ID();
+-----------------------+
| LAST_INSERT_ID()      |
+-----------------------+
| 1                     |
+-----------------------+
1 row in set (0.00 sec)
```

(this is a format subset. For full definition, see http://dev.mysql.com/doc/)
Variables

• If required, use variables to hold `LAST_INSERT_ID()` e.g.

```sql
mysql> INSERT INTO Subject (name) VALUES ('Fantasy');
Query OK, 1 row affected (0.00 sec)

mysql> SELECT @myvar:=LAST_INSERT_ID();
+----------------------------------+
<table>
<thead>
<tr>
<th>@myvar:=LAST_INSERT_ID()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
+----------------------------------+
1 row in set (0.01 sec)
mysql> INSERT INTO SomeOtherTable VALUES (@myvar,'somedata');
...```
LOAD DATA LOCAL INFILE statement

LOAD DATA LOCAL INFILE 'file_name'
  [REPLACE | IGNORE]
  INTO TABLE tbl_name
  [{FIELDS | COLUMNS}
    [TERMINATED BY 'string']
    [[OPTIONALLY] ENCLOSED BY 'char']
    [ESCAPED BY 'char']
  ]
  [LINES
    [STARTING BY 'string']
    [TERMINATED BY 'string']
  ]
  [IGNORE number LINES]
  [(col_name_or_user_var,...)]

Note: NULL values encoded as \N

(this is a format subset. For full definition, see http://dev.mysql.com/doc/ )
mysql> LOAD DATA LOCAL INFILE 'subjects.tab' INTO TABLE Subject (name);

subjects.tab

, Computer Networks
, Simulation
, Fantasy
, Evolution
, Biology
, Horror
, Science Fiction
, Genetics
, Travel
, Poetry
, Thrillers
, Engineering
, Law
, Humor
, Craft & Hobbies
, Art
, Architecture
Example data from Bookstore

```sql
mysql> select * from Book;
+---------------------+-------------------------------------------------+-------+------|----------------+------------------+
| ISBN                | title                                           | pages | price| publish       | Publisher_idPublisher |
|---------------------+-------------------------------------------------+-------+------|----------------+------------------|
| 3639159853          | The Impact of Bayesian Technology on Networking | 123   | 82   | 2009-00-00    | 1                 |
| 486450066           | On the Origin of Species                        | 89    | 5    | 2006-10-00    | 3                 |
| 786927089           | Dragons of summer flame                         | 389   | 7.99 | 1989-00-00    | 2                 |
| 786916095           | Dragons of Winter Night                         | 400   | 7.99 | 2000-04-00    | 2                 |
+---------------------+-------------------------------------------------+-------+------|----------------+------------------+
4 rows in set (0.00 sec)

mysql> select * from Publisher;
+-----------------+-----------------+------------------+
| idPublisher     | name            | url              |
|-----------------+-----------------+------------------|
| 1               | VDM Verlag      | http://www.vdm-publishing.com/ |
| 2               | Wizards of the Coast | http://www..wizards.com/ |
| 3               | Dover Publications | http://www.doverpublications.com/ |
| 4               | Wildside Press  | http://www.wildsidepress.com/ |
+-----------------+-----------------+------------------+
4 rows in set (0.00 sec)

mysql> select * from Author;
+-----------+---------+---------+----------------------------------+
| idAuthor  | firstname| lastname| biography                        |
|-----------+---------+---------+----------------------------------|
| 1         | John    | D'eaux  | Dr. John d'Eaux is a leading e ... |
| 2         | Margaret| Wies    | Margaret Weis was born on the ... |
| 3         | Tracy   | Hickman | Tracy and Laura Hickman have b ... |
| 4         | Charles | Darwin  | English Naturalist and author ... |
+-----------+---------+---------+----------------------------------+
6 rows in set (0.00 sec)
Example data from Bookstore

```sql
mysql> select * from Author;
+---------+------------+----------+---------------------------------------------+
| idAuthor | firstname   | lastname | biography                                   |
+---------+------------+----------+---------------------------------------------+
|        1 | John       | D'eaux   | Dr. John d'Eaux is a leading e ...           |
|        2 | Margaret   | Wies     | Margaret Weis was born on the ...            |
|        3 | Tracy      | Hickman  | Tracy and Laura Hickman have b ...           |
|        4 | Charles    | Darwin   | English Naturalist and author ...            |
+---------+------------+----------+---------------------------------------------+
6 rows in set (0.00 sec)

mysql> select * from Book_has_Author;
+-----------+------------+
| Book_ISBN | Author_idAuthor |
+-----------+----------------+
| 486450066 |              4 |
| 786916095 |              2 |
| 786916095 |              3 |
| 786927089 |              2 |
| 786927089 |              3 |
| 3639159853|              1 |
+-----------+----------------+
6 rows in set (0.00 sec)
```
### Example data from Bookstore

**Subject Table**

```sql
mysql> select * from Subject;
+--------+-----------------------------+
| idSubject | name                        |
+--------+-----------------------------+
|       1 | Computer Networks          |
|       2 | Simulation                  |
|       3 | Fantasy                    |
|       4 | Evolution                  |
|       5 | Biology                    |
|       6 | Horror                     |
+--------+-----------------------------+
6 rows in set (0.00 sec)
```

**Book_has_Subject Table**

```sql
mysql> select * from Book_has_Subject;
+----------+--------------------------+
| Book_ISBN | Subject_idSubject        |
+----------+--------------------------+
| 486450066 |                          4|
| 486450066 |                          5|
| 786916095 |                          3|
| 786927089 |                          3|
| 3639159853|                          1|
| 3639159853|                          2|
+----------+--------------------------+
6 rows in set (0.00 sec)
```
Query data

• SELECT statement
• WHERE expressions
• SELECT with aggregate functions
• SELECT with GROUP BY
• SELECT with ORDER BY
• SELECT with HAVING
SELECT statement

SELECT
    select_expr [, select_expr ...]
    [FROM table_references]
    [WHERE where_condition]
    [GROUP BY {col_name | expr | position}
        [ASC | DESC], ...]
    [HAVING where_condition]
    [ORDER BY {col_name | expr | position}
        [ASC | DESC], ...]
    [LIMIT {[offset,] row_count | row_count OFFSET offset}]

(this is a format subset. For full definition, see http://dev.mysql.com/doc/ )
WHERE condition

- Normal value expressions using column names
- Boolean operators (AND/OR/NOT)
  - a BETWEEN x AND y
- Numerical functions
  - CIEL(), MOD() ...
- STRING functions
  - a LIKE b, a REGEXP r,
  - RIGHT(), LEFT(), SUBSTR(), CONCAT()
- Subquery functions (later)
  - IN, EXISTS
- DATE functions
  - NOW(), DATE_ADD(x INTERVAL expr unit),
- Own MySQL functions (not today)
SELECT statement

mysql> SELECT * FROM Book WHERE title LIKE 'Dragon%' AND price<10.0;

<table>
<thead>
<tr>
<th>ISBN</th>
<th>title</th>
<th>pages</th>
<th>price</th>
<th>publish</th>
<th>Publisher_idPublisher</th>
</tr>
</thead>
<tbody>
<tr>
<td>786927089</td>
<td>Dragons of summer flame</td>
<td>389</td>
<td>7.99</td>
<td>1989-00-00</td>
<td>2</td>
</tr>
<tr>
<td>786916095</td>
<td>Dragons of Winter Night</td>
<td>400</td>
<td>7.99</td>
<td>2000-04-00</td>
<td>2</td>
</tr>
</tbody>
</table>

2 rows in set (0.00 sec)
SELECT with aggregate functions

mysql> SELECT COUNT(*), AVG(price), GROUP_CONCAT(ISBN) FROM Book WHERE pages > 320;

+-----------------+------------+-------------------+
| count(*) | avg(price) | group_concat(ISBN) |
+----------+-----------+-------------------+
|        2 |    7.99   | 786927089,786916095 |
+----------+-----------+-------------------+
1 row in set (0.00 sec)
SELECT with GROUP BY

mysql> SELECT
AVG(price), GROUP_CONCAT(ISBN), Publisher_idPublisher FROM Book WHERE price<10.0 GROUP BY Publisher_idPublisher;

+-----------------+-----------------------------+-----------------------+
<table>
<thead>
<tr>
<th>avg(price)</th>
<th>group_concat(ISBN)</th>
<th>Publisher_idPublisher</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.99</td>
<td>786927089,786916095</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>486450066</td>
<td>3</td>
</tr>
</tbody>
</table>
+-----------------+-----------------------------+-----------------------+
2 rows in set (0.00 sec)
mysql> SELECT AVG(price), GROUP_CONCAT(ISBN), Publisher_idPublisher FROM Book WHERE price < 10.0 GROUP BY Publisher_idPublisher ORDER BY AVG(price) ASC;

<table>
<thead>
<tr>
<th>avg(price)</th>
<th>group_concat(ISBN)</th>
<th>Publisher_idPublisher</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>486450066</td>
<td>3</td>
</tr>
<tr>
<td>7.99</td>
<td>786927089,786916095</td>
<td>2</td>
</tr>
</tbody>
</table>
SELECT with HAVING

mysql> SELECT AVG(price), GROUP_CONCAT(ISBN), Publisher_idPublisher FROM Book WHERE price<10.0 GROUP BY Publisher_idPublisher HAVING AVG(price)>6.5;

+------------------+-----------------+-------------+
| avg(price)  | group_concat(ISBN) | Publisher_idPublisher |
+------------------+-----------------+-------------+
| 7.99           | 786927089,786916095 | 2           |
+------------------+-----------------+-------------+
1 row in set (0.00 sec)
Updating data in tables

• **UPDATE** statement
  • Change existing rows in table

• **DELETE** statement
  • Remove existing rows in table

• **TRUNCATE** statement
  • Remove all rows from table
UPDATE statement

```
UPDATE [IGNORE] table_reference
    SET col_name1={expr1|DEFAULT} [, col_name2={expr2|DEFAULT}] ...
    [WHERE where_condition]
    [ORDER BY ...]
    [LIMIT row_count]
```

```
mysql> UPDATE Book SET price=10.5 WHERE ISBN=3639159853;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> UPDATE Book SET price=price*1.1 WHERE title LIKE 'Dragon%';
Query OK, 2 rows affected (0.00 sec)
Rows matched: 2  Changed: 2  Warnings: 0

(this is a format subset. For full definition, see http://dev.mysql.com/doc/ )
```
DELETE statement

DELETE FROM tbl_name
[WHERE where_condition]
[ORDER BY ...]
[LIMIT row_count]

mysql> DELETE FROM Book_has_Author WHERE Book_ISBN=786927089 AND Author_idAuthor=2;
Query OK, 1 row affected (0.00 sec)

mysql> DELETE FROM Book WHERE title LIKE 'Dragon%' LIMIT 1;
Query OK, 1 row affected (0.00 sec)

mysql> TRUNCATE TABLE Book;
Query OK, 0 row affected (0.00 sec)

(this is a format subset. For full definition, see http://dev.mysql.com/doc/ )
Coffee Break

We start again 10:15
Questions?
Exercise 1: Designing and creating a relational database

- Problem outline
- Importing / converting bulk data
- Simple queries
- Simple insertion
- Deletion
- Update
Exercise 1: Problem outline

• You are constructing a database to handle spots on several 2D gels. Your experiments generate a lot of gels, each with their own serial as well as information on the gel experiment protocol references, gel experiment date, lab technician. On the gels, spots are assigned molecular weight and pl. Each spot can in turn have assigned a UniProt protein ID, along with the mapping methods that was used ('PMF', 'MS/MS', 'Aa', 'Mi', 'Gm') to identify it.

• (Hint: Were not talking MIAPE here, just a simple exercise. We are aiming at 2 tables!)
Exercise 1: Design

• (A) Model
  • Look at the data files in ~/mysqlcourse/geldata*.txt
  • Build a model of the data entities (draw on paper)

• (B) Make table create statements
  • Make CREATE TABLE statements reflecting your model, attributes and relations.

• (C) Bulk insert of data statements
  • Parse input files in perl into one tab-delimited file for each of your tables.
  • Import given data into created tables with LOAD DATA LOCAL INFILE statement.
Exercise 1: Queries

- **(D) Query**
  - Write a MySQL statement that lists all spots within a mass and electrical range. (pi [5.0:5.2], mw [40000:41000])

- **(E) Query**
  - Write a MySQL statement that returns the average and standard deviation of molecular weight and pi of all spots within a 2D-gel. (serial G53546-F)

- **(F) Query**
  - Write a MySQL statement that returns the number of spots in a given gel that is identified by a given mapping method. (G53546-F, Mi)
Exercise 1: Updates

• (G) Update
  • Write a MySQL statement that updates the accession attribute and mapping method of a spot in the database. (G53546-F, 5.01, 40420 : P0A799, Gm)

• (H) Delete
  • Write a MySQL statements that deletes a given 2D-gel and all of its spots from the database. (Do not execute)
Solutions exercise 1

• (tabula rasa)
Lunch Break

We start again 13:15
Query data from multiple tables

- SELECT with sub-queries
- SELECT with JOIN
- SELECT with JOIN and aliasing
- Importance of indexes
- EXPLAIN command
## SELECT with sub-queries

```sql
mysql> SELECT firstname, lastname FROM Author WHERE firstname IN ('Tracy', 'Charles');
+------------------+------------------+
| firstname  | lastname  |
+------------------+------------------+
| Charles     | Darwin    |
| Tracy       | Hickman   |
+------------------+------------------+
2 rows in set (0.00 sec)

mysql> SELECT name FROM Publisher WHERE idPublisher IN (SELECT Publisher_idPublisher FROM Book WHERE pages<200);
+--------------------------+
| name                      |
+--------------------------+
| VDM Verlag               |
| Dover Publications       |
+--------------------------+
2 rows in set (0.00 sec)
```
SELECT with sub-queries

mysql> SELECT name FROM Publisher WHERE EXISTS ( SELECT * FROM Book WHERE price>5 AND pages<200 AND Publisher_idPublisher=idPublisher );
+-------------------+
| name              |
|-------------------+
| VDM Verlag        |
+-------------------+
1 row in set (0.00 sec)
Select with JOIN

• Query multiple tables, matching elements in one to elements in the other

• JOIN statements & expressions
  • INNER JOIN
  • ON
  • LEFT OUTER JOIN
  • RIGHT OUTER JOIN
  • More than two tables joins
INNER JOIN

mysql> SELECT Book.title, Publisher.name
FROM Book INNER JOIN Publisher ON
Book.Publisher_id=Publisher.id;

+-----------------+---------------+
| title           | name          |
|-----------------+---------------|
| The Impact of Bayesian Technology on Networking | VDM Verlag |
| Dragons of summer flame               | Wizards of the Coast |
| Dragons of Winter Night                   | Wizards of the Coast |
| On the Origin of Species                  | Dover Publications |
+-----------------+---------------+

4 rows in set (0.00 sec)
INNER JOIN

```
mysql> SELECT Book.title,GROUP_CONCAT(Subject.name) AS Subjects FROM Book
INNER JOIN Book_has_Subject ON
INNER JOIN Subject ON
Book_has_Subject.Subject_idSubject=Subject.idSubject
GROUP BY Book.ISBN;
```

<table>
<thead>
<tr>
<th>title</th>
<th>Subjects</th>
</tr>
</thead>
<tbody>
<tr>
<td>On the Origin of Species</td>
<td>Evolution,Biology</td>
</tr>
<tr>
<td>Dragons of Winter Night</td>
<td>Fantasy</td>
</tr>
<tr>
<td>Dragons of summer flame</td>
<td>Fantasy</td>
</tr>
<tr>
<td>The Impact of Bayesian Technology on Networking</td>
<td>Computer Networks,Simulation</td>
</tr>
</tbody>
</table>

4 rows in set (0.00 sec)
**LEFT JOIN**

```sql
mysql> SELECT Book.title, Publisher.name FROM Book LEFT OUTER JOIN Publisher ON Book.Publisher_id=Publisher.id AND Publisher.name LIKE '%%the%%';
```

<table>
<thead>
<tr>
<th>title</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Impact of Bayesian Technology on Networking</td>
<td>NULL</td>
</tr>
<tr>
<td>On the Origin of Species</td>
<td>NULL</td>
</tr>
<tr>
<td>Dragons of summer flame</td>
<td>Wizards of the Coast</td>
</tr>
<tr>
<td>Dragons of Winter Night</td>
<td>Wizards of the Coast</td>
</tr>
</tbody>
</table>

4 rows in set (0.00 sec)
LEFT JOIN

```sql
mysql> SELECT Book.title, Publisher.name FROM Book LEFT OUTER JOIN Publisher ON Book.Publisher_id = Publisher.id WHERE Publisher.name LIKE '%the%';
```

<table>
<thead>
<tr>
<th>title</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dragons of summer flame</td>
<td>Wizards of the Coast</td>
</tr>
<tr>
<td>Dragons of Winter Night</td>
<td>Wizards of the Coast</td>
</tr>
</tbody>
</table>

2 rows in set (0.01 sec)
### RIGHT JOIN

```sql
mysql> SELECT A.title, C.name FROM Book A INNER JOIN Book_has_Subject B ON B.Book_ISBN=A.ISBN RIGHT OUTER JOIN Subject C ON B.Subject_idSubject=C.idSubject;
```

<table>
<thead>
<tr>
<th>title</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Impact of Bayesian Technology on Networking</td>
<td>Computer Networks</td>
</tr>
<tr>
<td>The Impact of Bayesian Technology on Networking</td>
<td>Simulation</td>
</tr>
<tr>
<td>Dragons of summer flame</td>
<td>Fantasy</td>
</tr>
<tr>
<td>Dragons of Winter Night</td>
<td>Fantasy</td>
</tr>
<tr>
<td>On the Origin of Species</td>
<td>Evolution</td>
</tr>
<tr>
<td>On the Origin of Species</td>
<td>Biology</td>
</tr>
<tr>
<td>NULL</td>
<td>Horror</td>
</tr>
</tbody>
</table>

7 rows in set (0.00 sec)

*Hint: You can also use the aliasing to query the same table against itself!*
Importance of indexes

• Attribute lookup
  • BETWEEN
  • Comparison operations (< > <= >= =)
  • VARCHAR( ) as well!

• Key lookup in join
  • Uses indexes to match primary/foreign keys between tables
```sql
mysql> EXPLAIN SELECT A.title, C.name FROM Book A INNER JOIN Book_has_Subject B ON B.Book_ISBN=A.ISBN RIGHT OUTER JOIN Subject C ON B.Subject_idSubject=C.idSubject \G

<table>
<thead>
<tr>
<th>ID</th>
<th>SELECT_TYPE</th>
<th>TABLE</th>
<th>TYPE</th>
<th>POSSIBLE_KEYS</th>
<th>KEY</th>
<th>KEY_LEN</th>
<th>REF</th>
<th>ROWS</th>
<th>EXTRA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SIMPLE</td>
<td>C</td>
<td>ALL</td>
<td>NULL</td>
<td>NULL</td>
<td>NULL</td>
<td>NULL</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>SIMPLE</td>
<td>B</td>
<td>ref</td>
<td>PRIMARY,fk_Book_has_Subject_Book,fk_Book_has_Subject_Subject1</td>
<td>fk_Book_has_Subject_Subject1</td>
<td>4</td>
<td>hhs_private.C.idSubject</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>SIMPLE</td>
<td>A</td>
<td>eq_ref</td>
<td>PRIMARY</td>
<td>PRIMARY</td>
<td>8</td>
<td>hhs_private.B.Book_ISBN</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

3 rows in set (0.00 sec)
Exercise 2: More complex queries

• (I) Complex query
  • Write a MySQL statement that returns the species of all Gels entities that contain a spot assigned to a given accession. (P0A6Y8)

• (J) Complex query
  • Write a MySQL statement that that given a spot (ident!), returns spots that are close to that to spot on other gels.
    − (Pi: +- 0.2 Mw: +-100, spot: G53546-F,4.95,53154 )
    − Hint: use same table twice

• (K) Complex query
Write a MySQL statement that given a gel, returns all close spots that contain different accession assignment on other gels.
  − (Pi: +- 0.2 Mw: +-100, gel: G53546-F)
  − Hint: use same table twice
Solutions exercise 2

• (tabula rasa)
Questions?
Coffee Break

We start again 15:15
Using Perl DBI

• Connecting
• Preparing and executing statements
• Passing arguments
• Error handling
• Making an API module
• Using the module
use DBI;
use strict;

my $dbh = DBI->connect(
    'DBI:mysql:stud088_private:mysql.cbs.dtu.dk',
    'stud088', 'password'
) || die "Could not connect to database: $DBI::errstr";

# (insert query examples here...)

$dbh->do("CREATE TABLE IF NOT EXISTS test (id INT, val TEXT)");

$dbh->disconnect();

Your password:
interaction> cat ~/.my.cnf
Perl DBI - Preparing and executing statements

use DBI;
Use strict;

# (connect $dbh here)

my $sth=$dbh->prepare("SELECT id FROM test WHERE val=?");

$sth->execute("monkey");

while (defined (my $row=$sth->fetchrow_hashref())) {
    print $row->{id}.
}

$sth->finish();

$dbh->disconnect();
use DBI;
Use strict;

# (connect $dbh here)

my $sth=$dbh->prepare("SELECT id FROM test WHERE val=?");

$sth->execute("monkey");

while (defined (my $row=$sth->fetchrow_hashref())) {
    print $row->{id}.
}

$sth->finish();

$dbh->disconnect();
package myapi;
use DBI;
use strict;

sub new {
    my $class=shift;
    my ($dsn,$username,$password)=@_; 
    my $self=bless({},$class);
    $self->{dbh}=DBI->connect($dsn,$username,$password);
    $self->createschema(); 
    $self->{getcount}=
        $self->{dbh}->prepare( 
            "SELECT count(*) AS cnt FROM test WHERE val=?" );
    $self->{getrecord}=
        $self->{dbh}->prepare( 
            "SELECT * FROM test WHERE id=?" );
    $self->{addrecord}=
        $self->{dbh}->prepare( 
            "INSERT INTO test (val) VALUES (?)" );
    $self->{getwords}=
        $self->{dbh}->prepare( 
            "SELECT DISTINCT val FROM test" );
    return $self;
}
sub createschema {
    my $self=shift;
    $self->{dbh}->do("CREATE TABLE IF NOT EXISTS test (id INT NOT NULL AUTO_INCREMENT PRIMARY KEY, val TEXT)");
}

sub getrecord {
    my $self=shift;
    my $id=shift;
    $self->{getrecord}->execute($id);
    my $row=$self->{getrecord}->fetchrow_hashref();
    return $row;
}

sub addrecord {
    my $self=shift;
    my $val=shift;
    $self->{addrecord}->execute($val);
    return $self->{addrecord}->{mysql_insertid}
}
sub getcount {
    my $self=shift;
    my $arg=shift;
    $self->{getcount}->execute($arg);
    my $row=$self->{getcount}->fetchrow_hashref();
    return $row->{cnt};
}

sub getwords {
    my $self=shift;
    my @result;
    $self->{getwords}->execute();
    while (defined (my $row=$self->{getwords}->fetchrow_hashref()))) {
        push @result,$row->{val};
    }
    return @result;
}

1;
Perl DBI – using the module

use myapi;
use strict;

my $h=new myapi(
    "DBI:mysql:hhs_private:mysql.cbs.dtu.dk",
    "hhs",
    "foobaz");

my $id;
$id=$h->addrecord('horse');
$id=$h->addrecord('mouse');
$id=$h->addrecord('mouse');
$id=$h->addrecord('lion');

my @words=$h->getwords();
foreach my $word (@words) {
    my $count=$h->getcount($word);
    print "$word: $count\n";
}
Exercise 3: Implement API module

- Build a Perl module (.pm) file
  - Template in ~/mysqlcourse/exercise3/gelapi.pm
- Create object / Open database handle
  - Implement 'new' method to open database
  - Fill in template.
- Predeclare statements in object
  - Implement predeclaration of statements for exercise statements 'D','E','F' and 'K'
  - Fill in template.
- Expose object functions
  - Expose functions on database statements to perl object in the module for exercise statements 'D','E','F' and 'K'
  - Fill in template.
Exercise 3: Implement API module

- Implement schema creation 'C'
  - Fill in template.

- Implement more statements
  - Implement predecoration and exposiure of statements 'G','H','I' and 'J'
  - Fill in template.
Solutions exercise 3

• (tabula rasa)
Questions?
Thank you