2 Delivery

Question 0

Can you describe the series of steps to open a database for querying?

Answer

First of all, we need to open a MariaDB[1] session.

```
# Start service
sudo systemctl start mariadb.service
# Start MariaDB
mysql -u root -p
```

```
Enter password:
```

After entering the password[2], we get the welcome message:

```
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 3
Server version: 10.1.29-MariaDB MariaDB Server

Copyright (c) 2000, 2017, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input
   statement.

MariaDB [(none)]>
```

Then we need to check if the the database we are interested in (which we already added to MariaDB in class) is in our list of databases:

```
SHOW databases;
```

```
+--------------------+
| Database           |
+--------------------+
| experiments        |
| information_schema |
| mysql              |
| performance_schema |
| test               |
+--------------------+
5 rows in set (0.00 sec)
```

[1] Arch Linux does not include MySQL in the main repositories, and for the purposes of this course, both database servers are essentially interchangable.

[2] A password can be set up using the following command: `mysqladmin -u root password <PASSWORD>`.

Then all we do is tell MariaDB to use the `experiments` database:

```
1 USE experiments;
```

```
Database changed
```

Having changed our database, we can check what tables are contained in it:

```
1 SHOW Tables;
```

```
+-----------------------+
| Tables_in_experiments |
+-----------------------+
| Data                  |
| Descriptions          |
| GO_Descr              |
| LocusDescr            |
| LocusLinks            |
| Ontologies            |
| RefSeqs               |
| Sources               |
| Targets               |
| UniDescr              |
| UniSeqs               |
| Unigenes              |
+-----------------------+
12 rows in set (0.01 sec)
```

Since we will be using all these tables throughout the exercise, we will summarise the fields contained in each of the tables above:

Table	Data fields in the table		
Data	affyId,	exptId,	level
Descriptions	gbId,	description	
GO_Descr	goAcc,	description	
LocusDescr	linkId,	description,	species
LocusLinks	gbId,	linkId	
Ontologies	linkId,	goAcc	
RefSeqs	linkId,	ntRefSeq,	aaRefSeq
Sources	exptId,	source	
Targets	gbId,	affyId,	species
UniDescr	uId,	description	
UniSeqs	uId,	gbId	
Unigenes	linkId,	uId	

Table 2.1: Data fields in the different tables in the `experiments` database

Question 1

What is the purpose of this query?

```
1 SELECT * FROM Sources;
```

Answer

What this command does is show all the features (columns) from table `Sources`:

```
+--------+-------------+
| exptId | source      |
+--------+-------------+
| 1      | Pancreas    |
| 2      | Liver       |
| 4      | Human Liver |
+--------+-------------+
3 rows in set (0.00 sec)
```

Question 2

Get 5 GenBank IDs (`gbId`) and corresponding descriptions.

Answer

We want to select the first 5 records from table `Sources`. We can easily do this using the command `LIMIT`:

```
1 SELECT * FROM Descriptions
2 LIMIT 5;
```

```
+--------+-----------------------------------------+
| gbId   | description                             |
+--------+-----------------------------------------+
| A00142 | granulysin                              |
| A00146 | lypase, gastric                         |
| A03911 | seryne (or cysteine) proteinase inhibitor |
| A06977 | albumin                                 |
| A12027 | S100 calcium binding protein A8         |
+--------+-----------------------------------------+
5 rows in set (0.00 sec)
```

Question 3

What is the purpose of this query?

```
1 SELECT COUNT(*) FROM LocusLinks;
```

Answer

What this command does is count the amount of records (rows) contained in table `LocusLink` and show the resulting number:

```
+----------+
| COUNT(*) |
+----------+
|       22 |
+----------+
1 row in set (0.00 sec)
```

Question 4

How many different Affy IDs (`affyId`) are in the expression data?

Answer

To get the amount of different Affy IDs we need to compose **`COUNT()`** with **`DISTINCT()`**:

```
1 SELECT COUNT(DISTINCT(affyId)) FROM Data;
```

```
+--------------------------+
| COUNT(DISTINCT(affyId)) |
+--------------------------+
|                       23 |
+--------------------------+
1 row in set (0.00 sec)
```

Question 5

What is the expression level of Affy ID `U95-32123_at` in experiment number 1?

Answer

Using the query below, we can see that the expression level is `128`:

```
1 SELECT * FROM Data
2 WHERE affyId='U95-32123_at' AND exptId=1;
```

```
+--------------+--------+-------+
| affyId       | exptId | level |
+--------------+--------+-------+
| U95-32123_at | 1      |   128 |
+--------------+--------+-------+
1 row in set (0.01 sec)
```

Question 6

Find all the gene descriptions, along with their GenBank IDs containing the word "Human"?

Answer

We can do this using **LIKE** when using the **WHERE** clause. Notice that the argument for **LIKE** is case insensitive:

```
SELECT * FROM Descriptions
WHERE description LIKE "%human%";
```

```
+--------+-------------------------------------------------------------+
| gbId   | description                                                 |
+--------+-------------------------------------------------------------+
| A12345 | HSLFBPS7 Human fructose-1, 6-biphosphatase                  |
| A12346 | HSU30872 Human mitosin mRNA                                 |
| A12347 | HSU33052 Human lipid-activated protein kinase               |
| A12348 | HSU33053 Human lipid-activated protein kinase               |
| A12349 | Human clone lambda 5 semaphorin mRNA                        |
| A22124 | Human rearranged immunoglobulin lambda light chain mRNA     |
| A22127 | Human rearranged immunoglobulin lambda light chain mRNA     |
+--------+-------------------------------------------------------------+
7 rows in set (0.00 sec)
```

Question 7

What Gene Ontology descriptions (and corresponding accession) contain the phrase "protein kinase"? *The answer should be provided in ascending order of accessions.*

Answer

```
SELECT * FROM GO_Descr
WHERE description LIKE "%protein␣kinase%"
ORDER BY goAcc ASC;
```

```
+---------+----------------+
| goAcc   | description    |
+---------+----------------+
| 0001236 | protein kinase |
| 0001237 | protein kinase |
| 1112222 | protein kinase |
| 4442222 | protein kinase |
+---------+----------------+
4 rows in set (0.00 sec)
```

Question 8

Which AffyId of table `Data` correspond to sequences in `Targets` table with the phrase "kinase" in their description?

Answer

```
SELECT Data.affyId, Descriptions.description
FROM Data, Targets, Descriptions
WHERE Data.affyId=Targets.affyId AND
      Targets.gbId=Descriptions.gbId AND
      Descriptions.description LIKE "%kinase%";
```

```
Empty set (0.01 sec)
```

Now we use the command

```
LOAD DATA INFILE 'file.tsv' INTO TABLE Descriptions;
```

to add two new entries in the `Descriptions` table with `gbId="M18228", "L02870"` and the string "kinase" as the description for both, where `file.tsv` has the folliwing contents:

```
M18228 kinase
L02870 kinase
```

Now we repeat the query again:

```
SELECT Data.affyId, Descriptions.description
FROM Data, Targets, Descriptions
WHERE Data.affyId=Targets.affyId AND
      Targets.gbId=Descriptions.gbId AND
      Descriptions.description LIKE "%kinase%";
```

```
Empty set (0.00 sec)
```

The idea is that by adding the new records into the `Description` table, we would obtain some results. The problem is that the data is either incomplete or not well-formatted (something we would have to report to the database architect).

For instance, for the first line (`M18228 kinase`) we have done the following test:

```
SELECT * FROM Descriptions WHERE gbId LIKE "M18228";
SELECT * FROM Targets WHERE gbId LIKE "M18228";
SELECT * FROM Data WHERE affyId LIKE "U95_32123_at";
```

```
MariaDB [experiments]> SELECT * FROM Descriptions WHERE gbId LIKE
   "M18228";
+--------+-------------+
| gbId   | description |
+--------+-------------+
```

```
| M18228 | kinase      |
+--------+-------------+
1 row in set (0.00 sec)

MariaDB [experiments]> SELECT * FROM Targets WHERE gbId LIKE
   "M18228";
+--------+---------+---------+
| gbId   | affyId  | species |
+--------+---------+---------+
| M18228 | 1235_at | Mm      |
+--------+---------+---------+
1 row in set (0.00 sec)

MariaDB [experiments]> SELECT * FROM Data WHERE affyId LIKE "%1235%";
Empty set (0.01 sec)
```

Notice how there is no `1235_at affyId` in the table `Data`.

For the second line (`L02870 kinase`), the problem is how the `affyId` is formatted:

```
1 SELECT * FROM Targets WHERE gbId LIKE "L02870";
2 SELECT * FROM Data WHERE affyId LIKE "U95_32123_at";
```

```
MariaDB [experiments]> SELECT * FROM Targets WHERE gbId LIKE
   "L02870";
+--------+--------------+---------+
| gbId   | affyId       | species |
+--------+--------------+---------+
| L02870 | U95_32123_at | Mm      |
+--------+--------------+---------+
1 row in set (0.00 sec)

MariaDB [experiments]> SELECT * FROM Data WHERE affyId LIKE
   "U95_32123_at";
+--------------+--------+-------+
| affyId       | exptId | level |
+--------------+--------+-------+
| U95-32123_at | 1      |   128 |
| U95-32123_at | 2      |   128 |
+--------------+--------+-------+
2 rows in set (0.00 sec)
```

Notice how in `Targets`, the `affyId` is formatted as `U95_32123_at`, whilst in `Data` they are expressed as `U95-32123_at`. This causes the query `WHERE Data.affyId=Targets.affyId` to return no output.

We could use the query below instead, but it is far from being a good practice, and it should be avoided:

```
1 SELECT Data.affyId, Descriptions.description
2 FROM Data, Targets, Descriptions
3 WHERE Data.affyId LIKE Targets.affyId AND
```

```
4       Targets.gbId=Descriptions.gbId AND
5       Descriptions.description LIKE "%kinase%";
```

```
+--------------+-------------+
| affyId       | description |
+--------------+-------------+
| U95-32123_at | kinase      |
| U95-32123_at | kinase      |
+--------------+-------------+
2 rows in set (0.00 sec)
```

Question 9

Get two `affyId`, `uId` and `description` in `LocusDescr` in reverse alphabetical order of descriptions.

Answer

This question can be a little bit tricky, as there is no indication of the specific origin of the `affyId` or `uId`. We will show three different implementations that give us significantly different results:

```
1 SELECT Data.affyId, UniDescr.uId, LocusDescr.description
2 FROM Data, Targets, UniDescr, LocusDescr
3 WHERE Data.affyId=Targets.affyId AND
4       Targets.species=LocusDescr.species AND
5       LocusDescr.description=UniDescr.description
6 ORDER BY LocusDescr.description DESC
7 LIMIT 2;
```

```
+---------+--------+-------------+
| affyId  | uId    | description |
+---------+--------+-------------+
| 5324_at | Hs1691 | Glucan      |
| 5323_at | Hs1691 | Glucan      |
+---------+--------+-------------+
2 rows in set (0.00 sec)
```

```
1 SELECT Targets.affyId, UniDescr.uId, LocusDescr.description
2 FROM Targets, LocusDescr, UniDescr
3 WHERE LocusDescr.species=Targets.species AND
4       LocusDescr.description=UniDescr.description
5 ORDER BY LocusDescr.description DESC
6 LIMIT 2;
```

```
+---------+--------+-------------+
| affyId  | uId    | description |
+---------+--------+-------------+
```

```
| 3156_at | Hs1691 | Glucan      |
| 5323_at | Hs1691 | Glucan      |
+---------+--------+-------------+
2 rows in set (0.00 sec)
```

```
SELECT Targets.affyId, UniSeqs.uId, LocusDescr.description
FROM Targets, UniSeqs, LocusDescr, LocusLinks
WHERE LocusLinks.linkId=LocusDescr.linkId AND
      LocusLinks.gbId=UniSeqs.gbId AND
      Targets.gbId=LocusLinks.gbId
ORDER BY LocusDescr.description DESC
LIMIT 2;
```

```
+--------------+--------+-------------+
| affyId       | uId    | description |
+--------------+--------+-------------+
| U95_40474_at | Hs1691 | Glucan      |
| U95_32123_at | Hs1640 | Collagen    |
+--------------+--------+-------------+
2 rows in set (0.00 sec)
```

The thing to learn from this exercise is that one must be familiar with the database and be aware of the nature of the query.

Question 10

How would you find the average expression level of each experiment in `Data`?

Answer

```
SELECT exptId, AVG(level) FROM Data
GROUP BY exptId;
```

```
+--------+------------+
| exptId | AVG(level) |
+--------+------------+
| 1      |   125.3333 |
| 2      |    95.3333 |
| 3      |   126.3333 |
| 4      |    83.5000 |
| 5      |    92.7500 |
| 6      |    18.3333 |
| 7      |    20.0000 |
| 8      |    40.0000 |
| 9      |    20.0000 |
+--------+------------+
9 rows in set (0.00 sec)
```

Question 11

What is the average expression level of each array probe (`affyId`) across all experiments?

Answer

```
SELECT affyId, AVG(level) FROM Data
GROUP BY affyId;
```

```
+---------------------------+------------+
| affyId                    | AVG(level) |
+---------------------------+------------+
| 31315_at                  |   250.0000 |
| 31324_at                  |    91.0000 |
| 31325_at                  |    89.0000 |
| 31356_at                  |    91.0000 |
| 31362_at                  |   260.0000 |
| 31510_s_at                |   257.0000 |
| 5321_at                   |    90.0000 |
| 5322_at                   |    90.0000 |
| 5323_at                   |    90.0000 |
| 5324_at                   |    73.5000 |
| 5325_at                   |    90.0000 |
| AFFX-BioB-3_at            |    97.0000 |
| AFFX-BioB-5_at            |    20.0000 |
| AFFX-BioB-M_at            |    62.8000 |
| AFFX-HSAC07/X00351_M_at   |    86.0000 |
| AFFX-HUMBAPDH/M33197_3_st |   277.0000 |
| AFFX-HUMTFFR/M11507_at    |    90.0000 |
| AFFX-M27830_3_at          |   271.0000 |
| AFFX-MurIL10_at           |     6.6667 |
| AFFX-MurIL2_at            |    20.0000 |
| AFFX-MurIL4_at            |    49.0000 |
| U95-32123_at              |   128.0000 |
| U98-40474_at              |    57.0000 |
+---------------------------+------------+
23 rows in set (0.00 sec)
```

Question 12

What is the purpose of the following query?

```
SELECT Data.affyId, Data.level, Data.exptId, DataCopy.affyId,
    DataCopy.level, DataCopy.exptId
FROM Data, Data DataCopy
WHERE Data.level > 10 * DataCopy.level AND
      Data.affyId=DataCopy.affyId AND
      Data.affyId LIKE "AFFX%"
LIMIT 10;
```

Answer

```
+----------------+-------+--------+----------------+-------+--------+
| affyId         | level | exptId | affyId         | level | exptId |
+----------------+-------+--------+----------------+-------+--------+
| AFFX-BioB-M_at |   214 | 5      | AFFX-BioB-M_at |    20 | 3      |
| AFFX-BioB-M_at |   214 | 5      | AFFX-BioB-M_at |    20 | 7      |
| AFFX-BioB-M_at |   214 | 5      | AFFX-BioB-M_at |    20 | 9      |
+----------------+-------+--------+----------------+-------+--------+
3 rows in set (0.00 sec)
```

What this query is doing is to make a copy of the `Data` table in order to compare the `affyId`s that start with `AFFX` to show the pairs that fulfil the condition of having an expression `level` of one being 10 times larger than the other one. The query lets us not only compare the expression `level`, but the `exptId` value as well.

The `LIMIT 10` clause is there to show only the 10 first results, although it is unnecessary for this specific query, as there are only 3 results.

Question 13

Write a query to provide three different descriptions for all `gbId` in table `Targets`.

Answer

The tricky part with creating this query is that we have 4 different descriptions (in tables `Descriptions`, `LocusDescr`, `UniDescr`, and `GO_Descr`), so depending on which ones we choose, we will have different results.

Apart from that, one has to be careful on using the right relational path to connect all the tables properly. We will show two queries: (a) one to show `Descriptions.description`, `LocusDescr.description`, and `UniDescr.description`; and (b) another one to show `Descriptions.description`, `GO_Descr.description`, `LocusDescr.description`.

For space reasons, we will only show a sample of the output.

```
1 SELECT Targets.gbId, Descriptions.description AS description,
     LocusDescr.description AS LocusDescr, UniDescr.description AS
     UniDescr
2 FROM Targets, Descriptions, LocusDescr, UniDescr, UniSeqs
3 WHERE Targets.gbId=Descriptions.gbId AND
4       Targets.species=LocusDescr.species AND
5       UniSeqs.uId=UniDescr.uId AND
6       Targets.gbId=UniSeqs.gbId
7 ORDER BY gbId;
```

```
+--------+-------------+------------+----------------+
| gbId   | description | LocusDescr | UniDescr       |
+--------+-------------+------------+----------------+
```

```
| S75295 | Glucan      | Collagen   | Glucan         |
| S75295 | Glucan      | serine     | Glucan         |
| S75295 | Glucan      | Glucan     | Glucan         |
| S75295 | Glucan      | albumin    | Glucan         |
| S75295 | Glucan      | granulysi  | Glucan         |
| ...    | ...         | ...        | ....           |
+--------+-------------+------------+----------------+
42 rows in set (0.00 sec)
```

```
SELECT Targets.gbId, Descriptions.description AS description,
   GO_Descr.description AS GO_Descr, LocusDescr.description AS
   LocusDescr
FROM Targets, Descriptions, GO_Descr, LocusDescr, Ontologies,
   LocusLinks
WHERE Descriptions.gbId=Targets.gbId AND
      GO_Descr.goAcc=Ontologies.goAcc AND
      LocusDescr.linkId=Ontologies.linkId AND
      Targets.gbId=LocusLinks.gbId
ORDER BY gbId;
```

```
+--------+-----------------+--------------+------------+
| gbId   | description     | GoDescr      | LocusDescr |
+--------+-----------------+--------------+------------+
| A00142 | granulysin      | Glucan Enz   | Glucan     |
| A00142 | granulysin      | Glucan Enz   | Glucan     |
| A00142 | granulysin      | Serine Prot. | Collagen   |
| A00142 | granulysin      | Serine Prot. | Collagen   |
| A00146 | lypase, gastric | Glucan Enz   | Glucan     |
| ...    | ...             | ...          | ....       |
+--------+-----------------+--------------+------------+
40 rows in set (0.00 sec)
```

Question 14

Write a query to provide all gene ontology (`GO_descr`) descriptions related with all species in table `Targets` sorted alphabetically and providing the first five results. Export the query to a tab-separated-file with the command:

```
SELECT * FROM TABLE INTO OUTFILE 'data.out';
```

Answer

We just need to do the usual query and add the `INTO OUTFILE` expression at the end of it, as it can be seen below.

```
SELECT Targets.species, GO_Descr.description
FROM Targets, GO_Descr, Ontologies, LocusDescr
WHERE Targets.species=LocusDescr.species AND
```

```
4       LocusDescr.linkId=Ontologies.linkId AND
5       Ontologies.goAcc=GO_Descr.goAcc
6 ORDER BY species
7 INTO OUTFILE 'data.out';
```

```
+---------+--------------+
| species | description  |
+---------+--------------+
| Hs      | Serine Prot. |
| Hs      | Serine Prot. |
| Hs      | Serine Prot. |
| Hs      | Serine Prot. |
| Hs      | Serine Prot. |
+---------+--------------+
5 rows in set (0.00 sec)
```