**MySQL practicals N.B.: before you please the practical please show the demonstrators that you have completed the proposed exercise.**

# Exercise FIVE

**OBJECTIVES:** more SQL queries, stored procedures, parametric queries, data export/import.

**Task 1 -** Using the tables *students*, *courses* and *student\_taking\_course* you created in previous weeks, write (each in a different query tab) and execute the SQL queries that retrieve the following information (verify that the correct results are returned based on the data contained in your database):

1. The number of students;
   1. SELECT COUNT(studentid) FROM practical\_5.students; (Returns 6)
2. The number of different surnames of students;
   1. SELECT COUNT(DISTINCT(lastname)) FROM practical\_5.students;
3. The average cost of courses;
   1. SELECT AVG(coursecost) FROM practical\_5.courses;
4. The number of students that take the course EXL20;
   1. SELECT COUNT(studentid) FROM practical\_5.student\_taking\_course WHERE coursecode = "EXL20";
5. The courses ordered by cost;
   1. SELECT \* FROM practical\_5.courses ORDER BY coursecost ASC;
6. The name of courses with a code that ends with 20;
   1. SELECT (coursecode) FROM practical\_5.courses WHERE coursecode LIKE "%20%";
7. For each studentid, the average cost of courses taken by such a student
   1. SELECT studentid, AVG(coursecost)
   2. FROM practical\_5.courses
   3. INNER JOIN practical\_5.student\_taking\_course
   4. ON practical\_5.courses.coursecode = practical\_5.student\_taking\_course.coursecode
   5. GROUP BY studentid;

**Task 2 –** In your database you can create **stored procedures** which include queries. For example, the following command creates a stored procedure called **null-email** that contains the query that returns the students with null email address:

CREATE PROCEDURE `null-email`()

BEGIN

SELECT \*

FROM students

WHERE email is null;

END

**NOTE**: If the command above does not work on your system, you may need to use the command below instead.

DELIMITER //

CREATE PROCEDURE null\_email()

BEGIN

SELECT \* FROM students

WHERE email IS NULL;

END //

DELIMITER ;

For each of the queries in Task 1, create a stored procedure in your database by modifying the example above. You can create a stored procedure in the active database schema by clicking on the corresponding icon in the main menu or, alternatively, from the Navigator panel by right-clicking on *Stored Procedures* (under the database schema you want to use).

Remember to click on APPLY to **save** all the stored procedures you created. They will all appear in the Navigator panel under the corresponding database schema. **Execute** these stored procedures by clicking on the icon next to the name of each procedure. Alternatively you can execute a stored procedure by opening a query tab and invoking (calling) the procedure. For example if I open a query tab and type:

call mydb.null-email();

and execute this statement via the menu bar, the *null-email* stored procedure in the database called *mydb* is executed.

Exercise 5 – p. 1/2

**Task 3 –** A **parametric query** is a query that takes some parameters in input. For example the stored procedure created by the following statement:

CREATE PROCEDURE `param\_id`(IN p\_id INT)

BEGIN

SELECT \*

FROM students

where studentid=p\_id;

END

contains a parametric query that returns all information related to the student whose *studentid* is equal to the value given in input, i.e., this query requires a *studentid* as input parameter. Note that the datatype of this input parameter has to be compatible with the datatype of studentid. **Create** this stored procedure and **execute** it. You will be prompted with a window; enter a *studentid* in order to obtain the result.

**Task 4 –** Create the stored procedures corresponding to the parametric queries that return the following information:

1. Studentid of students with a given student name;
2. Name of course with a given course code;
3. Name of courses taken by students with a given last name;
4. The number of students with a given last name;
5. The number of courses with a given cost;
6. The number of students taking a given course;

**Task 5 –** Data export/import.

1. To export your database onto a file with extension .sql, from the main menu: Server → Data export. Export onto a single **self-contained file** including all components (stored procedures, etc.). Use a meaningful name and choose an appropriate directory on your laptop where you want to save it.
2. Create a new database, e.g., called *newdb*
3. Data import: from the main menu: Server → Data import. Try to import the .sql file you created into your *newdb* database. All data and functionality you created (eg. stored procedures) should be there.

**Task 6 –** Creating a diagram of a DB.

Select a DB in your application by clicking on it. In the main menu, click on Database → Reverse Engineer: this functionality allows you to create an ER diagram of your database – this will be useful for your assignment. You can create a pdf file containing this diagram by clicking on File → Print to file.

Exercise 5 – p. 2/2