Practical 2

Creating tables with basic constraints and SQL queries -1

Last update: 02 August 2022

Learning objectives

1. Write SQL statements to create a table with a primary key constraint.
2. Write SQL statements to create a table with NOT NULL constraint.
3. Execute .sql file in MySQL to run queries.
4. Write queries to select data from a single table using conditions, and using BETWEEN, IN, LIKE, AS commands, managing NULL values.
5. Write queries to retrieve date and time data in different formats
6. Use CONCATE, TRUNCATE, ROUND functions.

1. Setting-up

- Do this practical in DBS/Prac02 directory. If Prac02 directory is not there, create a one.
- Download and copy *create_tables.sql*, *insdept.sql* and *insemp.sql* files from the Topic 2/ practical -2 link in Blackboard to your Prac02 directory.
- Go to Prac02 directory, open a Terminal and connect to MySQL server using correct username and password (refer to Prac01-Part 1 Task 2 if you wish to look at the command).
- Open another Terminal while you are in the same directory, and open a text file with name *Prac02Commands* in Vim (> `vim Prac02Commands`) or using any text editor you wish to use.
- **Type your commands in this file from this point onward. Use a comment line before starting each task.**
- **Before typing any other command, type the following command in the MySQL prompt.**

```
mysql>tee Prac02Work.out
```

Same command in shorter format.

```
mysql>\t Prac02Work.out
```

This will create a file *Prac02Work.out* in the Prac02 directory. All your commands, error messages, and outputs (all work you do in the MySQL session and displayed on the screen) will be written to this file.

Prac02Work.out file can be read using vim or other text editors. You can refer it after finishing the practical to see your work.

- Make ‘dswork’ database the working database.
- Sample relations:

In this practical we will use two relations which store data bour employees and departments of an organization. Employees information is stored in ‘Emp’ table and department information is stored in ‘Dept’ table.

The descriptions of the ‘Emp’ and ‘Dept’ tables are given below.

Emp (Employee Table)

COL NAME	TYPE	SIZE	NULL	DESCRIPTION
empno	CHAR	6	no	Employee number, unique
firstname	VARCHAR	12	no	First name
midinit	CHAR	1	no	Middle initial
lastname	VARCHAR	15	no	Last name
workdept	CHAR	3		Employee's dept number
phoneno	CHAR	4		Employee's telephone number
hiredate	DATE			Date hired
job	CHAR	8		Job held by employee
edlevel	INT	2		No. of years of formal educ.
gender	CHAR	1		M=male, F=female
birthdate	DATE			Date of birth
salary	DECIMAL	(8,2)		Annual salary
bonus	DECIMAL	(8,2)		Annual bonus
comm	DECIMAL	(8,2)		Annual commission

Dept (Department Table)

COL NAME	TYPE	SIZE	NULL	DESCRIPTION
deptno	CHAR	3	no	Department number, unique
deptname	VARCHAR	36	no	Department name
mgrno	CHAR	6		Dept manager's employee no.
admrdept	CHAR	3	no	ID of administrative dept

2. Creating tables using a .sql file

In the previous lab you have typed your SQL statements in a text file and SOURCE it to run them in MySQL. Alternatively, your SQL statements can be saved as a *.sql* file and then use it to run your SQL statements.

create_tables.sql file contains SQL statements to create ‘Emp’ and Dept’ tables.

1. Open *create_tables.sql* file in vim or another text editor.

You will see the syntax is shown in different colours if used vim. Some other text editors also recognize *.sql* files would show the commands in different colours.

Look at the 'Dept' table description above and see how `NOT NULL` and `PRIMARY KEY` constraints are implemented in the `CREATE TABLE` Dept statement.

2. Edit the *create_tables.sql* file to add a suitable primary key constraint, and not null constraints to 'Emp' table. Look at the 'Emp' table description above to identify the columns for which the constraints should be implemented.

Save the file.

3. Run the *create_tables.sql* file in MySQL prompt using the following command:

```
mysql>\. create_tables.sql
```

Note 1: Running SQL commands from a file

- In general, \. followed by a script filename will execute the script file.
- Alternatively you can use `mysql > SOURCE create_table.sql;`
- SQL statements in a txt file also can be run the same way.

4. See the table structure using DESC command.

5. *insdept.sql* and *insemp.sql* files contain SQL statements to add data to the created two tables. Open *insdept.sql* and *insemp.sql* files and observe the statements.

   Run the *insdept.sql* and *insemp.sql* files to add data to the 'Emp' and 'Dept' tables.

6. Display all the rows and columns of 'Emp' table using a `SELECT *` statement. As 'Emp' table contains lot of rows and columns, the content would not fit the screen.

7. Type the following command to set scrolling the output:

```
mysql>pager less -SFX
```

Now type the SELECT * statement again and observe the output. Use space key to scroll to the next page. You can use 'End' key to go to last page' home' key to go to first page, up arrow, and down arrow keys to scroll one line up or down. Use 'q' to quit the less command.

Note2: Show output with scrolling support

- Unless you make the terminal display more lines, the 'Emp' table won't fit and some of the rows will be scrolled up before you can see them.
- To avoid this, you can set your pager (the part that displays query output pages) to use less, using the above command. Then, when output would scroll off screen immediately, the MySQL pager will call on the Linux `less` utility to pause the scrolling.

3. Writing queries

Write MySQL statements to answer the following queries on 'Emp' and 'Dept' tables.

Refer the table description given above to know the meaning of attributes. Use DES command and look at table structure any time if you want to know about the columns of the table.

1. Display the last name, work department, and the salary of all employees who get a salary of $100,000 or more per year.

2. Display the last name, first name, and the birth date of every employee whose salary is less than $90,000 per year.

3. Show the information of all departments that have the manager's employee number as null.

4. Show the employee number, last name, work department, and the phone number of employees whose work department number is between 'D01' and 'E01' (inclusive).

5. Display information of departments that have names containing the string 'Service'.
   In the query you create, change the string to 'service', re-run and observe the output. Is the output same or different?

6. Show the employee number, last name, and work department of employees of work department ‘D21’ with salary less than (including) $60000.

7. For each employee, get the name and job. Display the names of employees in a column called ‘Full_name’, with only a single space between first name, middle initial and last name.

8. Produce a list of employees who work in the departments with id numbers 'B01', 'C01', 'D11', and 'E21', showing last name, work department and monthly salary. Show only whole dollars (no cents).

9. Produce a list of employees who work in the departments with id numbers 'B01', 'C01' and 'E21', showing last name, work department and weekly salary. Show the weekly salary with two decimal places. (assume 52.1786 weeks per year for weekly salary calculation)

10. Get the names and birth dates of all designers, who are identified by the job value of 'Designer'. Display the birthdate in the form **Wednesday, 6 August 2008.**

11. Produce a list of employees who work in the departments with id numbers 'B01', 'C01', 'D11' and 'E21', showing employee number, first name, birth date and department number. Show only the date and month of the birth date in the form of **6 August**.

12. **Additional task**: Produce a list of employees who work in the departments with id numbers 'B01', 'C01' and 'E21', showing employee number, fist name, last name, work department, and the annual total income. Annual total income is the total of salary, bonus and commission and display it under a column heading ‘Total Income’.

13. **Challenging task**:
    Display the last name of all employees, along with their age.
    You can use several MySQL functions to do this.
    Hint: You may use,
    - `DATEDIFF` and, `NOW` or `CURRDATE` functions. `TRUNCATE` can be used to show only the years, after diving the days by number of days per year (i.e.365.25) without decimal places.

- `TIMESTAMPDIFF` and, `NOW` or `CURRDATE` functions.

Use mysql help function or refer to online reference manual for the details

```
mysql> help DATEDIFF;
```

of the functions (DATEDIFF is explained in Lecture-2 slides)

4. Submitting your work

Zip your Prac02 directory and upload it to Blackboard under Assessments/In class practical Submissions/Practical 2 link. Your directory must contain two files created by you, *Prac03Work.out*, and *Prac02Commands* txt file, with provided *.sql* files.

Note 3: Zip your directory

- Go to your DBS directory and type:

```
> zip -r Prac02_<your student ID> Prac02
```

Check whether you have achieved learning objectives:

I am confident that I can write MySQL statements to,

Implement `PRIMARY KEY, NOT NULL` constraints when creating a table	✓
Retrieve data from a table fulfilling multiple conditions (`AND, OR`)	
Retrieve data from a table with comparisons operations (=, `>, <, <=, >=, <>, !=`)	
Retrieve data from a table using `BETWEEN, IN, NOT IN`	
Retrieve data from a table using arithmetic operators, `ROUND, TRUNCATE.`	
Retrieve data from a table using `LIKE, CONCAT`.	
Retrieve data from a table using suitable data time functions and display date/ time in different formats.	
Create *.sql* files and run them in MySQL.	

If all the objectives were not achieved, please refer lecture slides, reading materials, and online resources and attempt again. Ask your tutor and get help if you need any clarification.

It's always a good practise to try to finish the practical of a particular week, before attempting the next practical worksheet as your work will be building upon the previous week's tasks.