# Practical: SQL – Insert, Update, Delete, DDL

## Objective

In this practical, you will practice following SQL command.

* DDL – CREATE TABLE, ALTER TABLE, CREATE VIEW, and CREATE INDEX
* DML – INSERT, UPDATE, and DELET

## Tasks

1. **Pre-lab Setup**
   1. Download the script *DBMS\_SQL\_DDL\_setup.sql* from NYP LMS.
   2. Launch MySQL Workbench. Connect as **root**.
   3. Open the script *DBMS\_SQL\_DDL\_setup.sql* and Run it. List the tasks the script does.
   4. Create a new database connection with following details.

|  |  |
| --- | --- |
| Connection Name | DDLstudent\_<your AdmNo> |
| Hostname | 127.0.0 (default) |
| Port | 3306 (default) |
| Username | ddlstudent |
| Password (when prompt) | ddlstudPa55 |

**Note: In the following steps, you should keep backup copy of all your scripts so that in case your tables/data are damaged accidentally, you could re-run your scripts to re-build the tables or to populate the data.**

1. **Run Scripts to create tables**
2. Download *CreateHotelBookingTables.sql* from the course web site.
3. Connect MySQL using the *DDLstudent\_<your AdmNo>* connection. Select **File > Open** to open the script. Describe the functions of the script.
4. Run the script by press the **Run Script button**.
5. **SQL INSERT Statement**

In this exercise, you are to practice the INSERT statement. You need to insert the following data into the tables that you created in the previous task.

* 1. Insert the data into the Gx\_HOTEL

|  |  |  |
| --- | --- | --- |
| **HOTEL\_NO** | **HOTEL\_NAME** | **CITY** |
| H111 | Grosvenor Hotel | London |

**insert into gx\_hotel(Hotel\_no , Hotel\_name , City )**

**values('H111' , 'Grosvenor Hotel' , 'London');**

* 1. Insert data into Gx\_ROOM

|  |  |  |  |
| --- | --- | --- | --- |
| **ROOM\_NO** | **HOTEL\_NO** | **ROOM\_TYPE** | **PRICE** |
| 1001 | H111 | S | 172 |
| 2 1002 | H111 | D | 200 |
| 1003 | H111 | F | 300 |

**insert into gx\_room**

**values('1001' , 'H111' , 'S','172');**

**insert into gx\_room**

**values('1002' , 'H111' , 'D','200');**

**insert into gx\_room**

**values('1003' , 'H111' , 'F','300');**

* 1. Insert data into Gx\_GUEST

|  |  |  |
| --- | --- | --- |
| **GUEST\_NO** | **GUEST\_NAME** | **GUEST\_ADDR** |
| G001 | John Smith | London |
| G002 | Michael | London |
| G003 | James | London |

**insert into gx\_guest**

**values('G001' , 'John Smith' , 'London');**

**insert into gx\_guest**

**values('G002' , 'Michael' , 'London');**

**insert into gx\_guest**

**values('G003' , 'James' , 'London');**

* 1. Insert data into Gx\_BOOKING

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **HOTEL\_NO** | **GUEST\_NO** | **DATE\_FROM** | **DATE\_TO** | **ROOM\_NO** |
| H111 | G001 | 01-Jan-2003 | 03-Jan-2003 | 1001 |
| H111 | G002 | 05-Feb-2003 | 07-Feb-2003 | 1001 |
| H111 | G003 | 15-Jan-2003 | 17-Jan-2003 | 1003 |

For MySQL, use the date format ‘YYYY-MM-DD’, e.g. ‘2003-01-17’.

**insert into gx\_booking**

**values('H111' , 'G001' , '2003-01-01' , '2003-01-03' , '1001');**

**insert into gx\_booking**

**values('H111' , 'G002' , '2003-02-05' , '2003-02-07' , '1001');**

**insert into gx\_booking**

**values('H111' , 'G003' , '2003-01-15' , '2003-01-17' , '1003')**

1. **Update / Delete SQL**
   1. Guest ‘John Smith’ has called the hotel to inform the change of his address to ‘Manchester’. Perform the update in the guest database.

**SET SQL\_SAFE\_UPDATES = 0;**

**update gx\_guest**

**set Guest\_Addr = 'Machester'**

**where Guest\_Name = 'John Smith';**

* 1. The ‘Grosvenor Hotel’ has decided to increase all the room rates by 5%. Perform the update in the database table.

**update gx\_room**

**set price = price\*1.05**

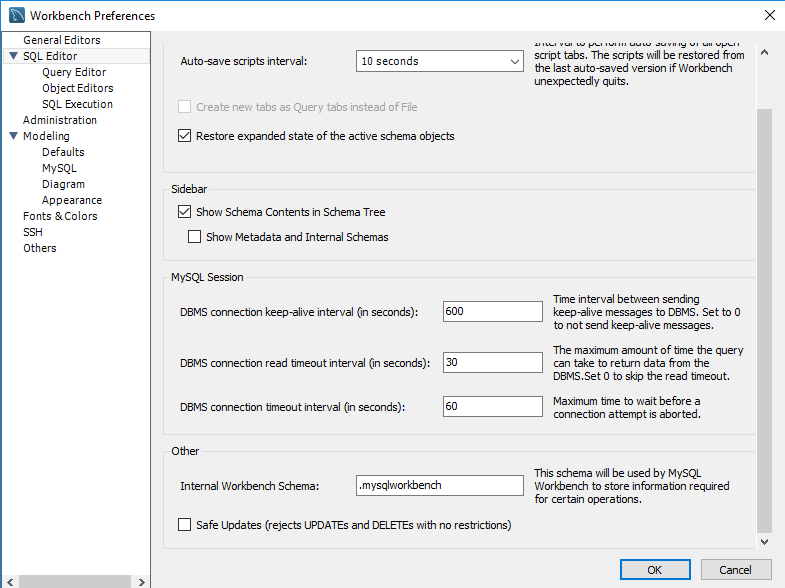
**where Hotel\_No = (Select Hotel\_No FROM Gx\_Hotel where Hotel\_name = ‘**Grosvenor Hotel**’**

Note:

If you got this error “Error Code: 1175. You are using safe update mode …”

You can do the following:

**Way 1**. To disable safe mode, toggle the option in Preferences -> SQL Editor as shown below and **reconnect.**



Uncheck this.

**Way 2.** Execute the following statement in the query window.

SET SQL\_SAFE\_UPDATES = 0;

1. **SQL/DDL: ALTER TABLE Statement**
   1. Add a new column which can contain NULL value

Modify the **Gx\_HOTEL** table to add in the following new column which allows NULL value :

Phone : VARCHAR (12)

**alter table gx\_hotel**

**add (Phone varchar(12) );**

* 1. Add a new column which is NOT NULL

Suppose the business rules have changed and you need to add in a new column, HOTEL\_MANAGER, of type VARCHAR (20) to the table **Gx\_HOTEL,** how can this be done? What happen to the existing records?

**alter table gx\_hotel**

**add (HOTEL\_MANAGER varchar(20) DEFAULT ‘Gary’ Not Null);**

1. **SQL/DDL: CREATE VIEW Statement**
   1. Create a view (**Gx\_HOTEL\_GUEST**) to list all the guests’ bookings at all the hotels. Include the following information in the view:

* Hotel Name
* Guest Name
* Room Type
* Booking From Date
* Booking Duration
  1. Show the data. What is the SQL statement?
  2. What is the advantage of creating view?

1. **Understand Index** 
   1. Insert some more data in Gx\_GUEST table:

use dbms\_ddl;

INSERT INTO gx\_Guest VALUES ('G111', 'John Smith', 'London');

INSERT INTO gx\_Guest VALUES ('G112', 'Mike Tan', 'London');

INSERT INTO gx\_Guest VALUES ('G113', 'Alice', 'London');

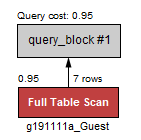
INSERT INTO gx\_Guest VALUES ('G114', 'Ben Wang', 'London');

INSERT INTO gx\_Guest VALUES ('G115', 'Thomas', 'London');

INSERT INTO gx\_Guest VALUES ('G116', 'Shirly', 'London');

INSERT INTO gx\_Guest VALUES ('G117', 'Zoe ', 'London');

* 1. Write a SQL statement that display all detail of guest with name as ‘Shirly’.
  2. With Cursor place at anywhere with in the SELECT statement, Click **Execute EXPLAIN command** button . You will see the visual explain of the SELECT statement as following:



Full table scan indicate in order to find the guest with name ‘Shirly’, database need to scan through the whole table. The cost of the search is 0.95. If the table contains more data, the cost will increase drastically.

* 1. Now create an index on the **guest name** of the **Gx\_GUEST** table. Name the index as **Gx\_GUEST\_NAME\_IX**. Write down the SQL statement
  2. With Cursor place at anywhere with in the SELECT statement, Click **Execute EXPLAIN command** button again. The execution explaination changed to following. Instead of full table scan, it changes to Non-Unique Key Lookup. The query cost is also reduced. In a table with a lot of data, the difference in query cost can be tremendous.

