SCHOOL OF
COMPUTER AND SECURITY SCIENCE

CSG1207/CSI5135: Systems and Database Design
Lab 08 - Solutions

Standard Disclaimer

Many questions you encounter in this and other labs have more than one solution which is valid and correct. There are often numerous ways to achieve the same results in an SQL query.

The solutions provided here may NOT be the only correct answers to the questions. If you have arrived at solution to a lab task that differs substantially from what is provided here and would like feedback on your solution, please contact your tutor.

Lab Tasks

Q1.

```
INSERT INTO job
VALUES ('GN_SEC', 'Secretary', 3500, 10000);
```

Q2.

```
INSERT INTO job (job_id, job_title, min_salary)
VALUES ('GN_JAN', 'Janitor', 1500);
```

or

```
INSERT INTO job (job_id, job_title, min_salary, max_salary)
VALUES ('GN_JAN', 'Janitor', 1500, NULL);
```

Q3.

```
INSERT INTO job
VALUES ('GN_CAF', 'Cafeteria Worker', DEFAULT, 4500);
```

or

```
INSERT INTO job (job_id, job_title, min_salary, max_salary)
VALUES ('GN_CAF', 'Cafeteria Worker', DEFAULT, 4500);
```

Q4.

```
UPDATE job
SET min_salary = 2000
WHERE job_id IN('GN_SEC', 'GN_JAN');
```

Q5. SQL Server will not insert any of the rows in a multiple-insert statement if any of them contain an error – the whole statement is terminated.

Q6.

```
DELETE FROM country
WHERE region_id > 2;
```

Q7.

```
UPDATE job
SET max_salary =   ( SELECT max_salary
                     FROM job
                     WHERE job_id = 'GN_SEC')
WHERE job_id = 'GN_CAF';
```

Q8. You should be able to observe that although you increase the minimum and maximum salaries for all jobs, the transaction is rolled back to a point before that occurs, so the only change that is committed is the addition of the GN_SPY job.

```
BEGIN TRANSACTION;

INSERT INTO job VALUES ('GN_SPY', 'Corporate Spy', 50000, 75000);

SAVE TRANSACTION after_spy;

UPDATE job SET min_salary = 50000, max_salary = 75000;

SELECT * FROM job;

ROLLBACK TRANSACTION after_spy;

COMMIT TRANSACTION;
```

Q9.

```
DELETE FROM job
WHERE job_id LIKE 'GN%';
```

Challenge Query!

Q10.

```
CREATE TABLE emp_summary
(
  emp_id INT NOT NULL CONSTRAINT emp_sum_pk PRIMARY KEY,
  full_name VARCHAR(50) NOT NULL,
  email VARCHAR(75) NULL,
  phone VARCHAR(20) NULL
);
```

Now that the table has been created, the data can be inserted via an INSERT statement that uses a subquery…

```
INSERT INTO emp_summary (emp_id, full_name, email, phone)
SELECT employee_id,
         first_name + ' ' + last_name,
         LOWER(email) + '@company.com',
         phone_number
FROM employee;
```