CpSc 4620/6620 Quiz #8

Name:________________ ID:____________

1. Determine whether the following statements are **TRUE** or **FALSE** (40 points):
   1) Usually, a view is a table stored in the database by running SQL queries on other tables. (F)
   2) You cannot update a view using the SQL statement directly on the view. Instead you have to write SQL statement to the underlying tables to update the view. (F)
   3) The SELECT statement used to create a view cannot contain a subquery in the FROM clause. (T)
   4) The PRIMARY key is a single column that uniquely identifies a row in a relational table. (F)
   5) A unique key is automatically enforced by NOT NULL constraint. (F)
   6) In relational database, a foreign key is used to ensure the referential constraint. (T)
   7) A foreign key in a relational database always reflects a one to many relationship (F)
   8) Index is not necessary for a column on which the records are ordered. (F)
   9) A NO-ACTION referential action does not allow any UPDATE or DELETE to the referenced table. (F)
   10) A trigger will be activated whenever a change to the database happens. (F)
   11) Stored procedures are always good for DBMS applications. (F)
   12) Without using a proper LEAVE statement, the LOOP flow control in a stored procedure will iterate infinitely. (T)
   13) In MySQL, a stored function must have returns. (T)
   14) In MySQL, a stored function does not have OUT parameters. (T)
   15) A stored routine can only be executed by the user who creates it. (F)
   16) In a stored procedure, the scope of a local variable is within the BEGIN ... END block where it is declared. (T)
   17) In MySQL stored procedures, SET is the only statement that can be used to assign values to local variables. (F)
   18) In MySQL stored procedures, a begin_label for LOOP or REPEAT flow control must be accompanied with an end_label and they must be the same. (T)
   19) In stored procedures, REPEAT always enters the loop at least once. (T)
   20) In stored procedures, a CONTINUE handler allows the current routine continues without execution of the handler statement. (F)

2. For a table created by a SQL statement “CREATE TABLE tbl (quantity INT, price INT);” Please write a SQL statement to create a view that contains one extra integer column “total”, which is calculated by “quantity * price”. (20 points):

```
CREATE VIEW v AS SELECT quantity, price, quantity*price AS total FROM tbl;
```

3. After the following sequence of SQL statements, what will be the results of queries (1) to (4)? (20 points)

```
CREATE TABLE test1(a1 INT);
CREATE TABLE test2(a2 INT);
CREATE TABLE test3(a3 INT NOT NULL AUTO_INCREMENT PRIMARY KEY);
CREATE TABLE test4(
  a4 INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
  b4 INT DEFAULT 0
);

DELIMITER |

CREATE TRIGGER testref BEFORE INSERT ON test1
  FOR EACH ROW BEGIN
    INSERT INTO test2 SET a2 = NEW.a1;
    DELETE FROM test3 WHERE a3 = NEW.a1;
    UPDATE test4 SET b4 = b4 + 1 WHERE a4 = NEW.a1;
  END;
|

DELIMITER ;

INSERT INTO test3 (a3) VALUES
  (NULL), (NULL), (NULL), (NULL), (NULL);

INSERT INTO test4 (a4) VALUES
  (0), (0), (0), (0), (0);

INSERT INTO test1 VALUES (1), (3), (1), (1), (4), (4);
```

(1) SELECT * FROM test1;

```
+------+
| a1   |
+------+
|    1 |
|    3 |
|    1 |
|    1 |
|    4 |
|    4 |
+------+
```

(2) SELECT * FROM test2;

```
+------+
| a2   |
+------+
|    1 |
|    3 |
|    1 |
|    1 |
|    4 |
|    4 |
+------+
```

(3) SELECT * FROM test3;

```
+----+
| a3 |
+----+
|  2 |
|  5 |
+----+
```

(4) SELECT * FROM test4;

```
+----+------+
| a4 | b4   |
+----+------+
|  1 |    3 |
|  2 |    0 |
|  3 |    1 |
|  4 |    2 |
|  5 |    0 |
+----+------+
```

4. Given the following stored procedure:

```
CREATE PROCEDURE p (OUT date_param VARCHAR(25), INOUT incr_param INT)
BEGIN
  SELECT CURDATE() INTO date_param;
  SET incr_param = incr_param + 1;
END;
```

If you run the following sequence of MySQL statements today from the command line, what results will be returned by the last statement? (20 points)

```
mysql> SET @x = 10;
mysql> CALL p(@dt, @x);
mysql> SELECT @dt, @x;
```

```
+------------+------------+
| @dt        | @x         |
+------------+------------+
| 2019-04-23 | 11         |
+------------+------------+
```

6. What results will the following MySQL commands return? (10 points)

```
mysql> CREATE FUNCTION hello (s CHAR(20))
   -> RETURNS CHAR(50) DETERMINISTIC
   -> RETURN CONCAT('Hello, ',s,'!');
Query OK, 0 rows affected (0.00 sec)

mysql> SELECT hello('world');
```

```
+----------------+
| hello('world') |
+----------------+
| Hello, world!  |
+----------------+
```

7. Give the results of calling the following stored procedure on MySQL client (15 points).

```
DELIMITER $$
CREATE PROCEDURE my_procedure_Local_Variables()
BEGIN   /* declare local variables */
DECLARE a INT DEFAULT 10;
DECLARE b, c INT;    /* using the local variables */
SET a = a + 100;
SET b = 2;
SET c = a + b;
BEGIN      /* local variable in nested block */
DECLARE c INT;
SET c = 5;
/* local variable c takes precedence over the one of the
same name declared in the enclosing block. */
SELECT a, b, c;
END;
SELECT a, b, c;
END$$
```

```
mysql> CALL my_procedure_Local_Variables();

+------+------+------+
| a    | b    | c    |
+------+------+------+
|  110 |    2 |    5 |
+------+------+------+
1 row in set (0.00 sec)

+------+------+------+
| a    | b    | c    |
+------+------+------+
|  110 |    2 |  112 |
+------+------+------+
1 row in set (0.01 sec)
Query OK, 0 rows affected (0.03 sec)
```