Ilhak Park (501072432)
CIND110 – Data Organization for Data Analysts
Assignment 2 – Mar 16 202

Question 1 — 25 Points

Create a database 'Hollywood' and create the below tables with the constraints listed below:

Movie(mID **int**, title text, **year int**, director text); Reviewer(rID **int**, name text);
Rating(rID **int**, mID **int**, stars **int**, ratingDate **date**);

Enforce the following constraints on the above database:

1. Movie and Reviewer should have primary key constraints on the respective id columns. (5)
2. Place auto increment on the 'mID' and 'rID' columns in the Movie and Reviewer tables. (5)
3. Rating table columns 'rID' and 'mID' should refer to the respective columns in the parent tables i.e. Movie and Reviewer. (5)
4. The default value of the 'ratingDate' column in the Rating table should be the current date. (5)
5. The 'year' column in the Movie table should not be greater than 2016. (5)

#1,2

```
mysql>
mysql> CREATE DATABASE Hollywood;
Query OK, 1 row affected (0.00 sec)

mysql> use Hollywood;
Database changed
mysql> CREATE TABLE Movie(mID, title text, year int, director text);
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that
corresponds to your MySQL server version for the right syntax to use near ' titl
e text, year int, director text)' at line 1
mysql> CREATE TABLE Movie(mID int, title text, year int, director text);
Query OK, 0 rows affected (0.07 sec)

mysql> CREATE TABLE Reviewer(rID int, name text);
Query OK, 0 rows affected (0.08 sec)

mysql> CREATE TABLE Rating(rID int, mID int, stars int, ratingDate timestamp);
Query OK, 0 rows affected (0.02 sec)

mysql> SHOW TABLES;
+---------------------+
| Tables_in_Hollywood |
+---------------------+
| Movie               |
| Rating              |
| Reviewer            |
+---------------------+
3 rows in set (0.01 sec)

mysql> ALTER TABLE Movie MODIFY COLUMN mID int auto_increment;
ERROR 1075 (42000): Incorrect table definition; there can be only one auto colum
n and it must be defined as a key
mysql> ALTER TABLE Movie ADD CONSTRAINT PK_Movie PRIMARY KEY (mID);
Query OK, 0 rows affected (0.09 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> ALTER TABLE Reviewer ADD CONSTRAINT PK_Reviewer PRIMARY KEY (rID);
Query OK, 0 rows affected (0.08 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> ALTER TABLE Movie MODIFY COLUMN mID int auto_increment;
Query OK, 0 rows affected (0.11 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> ALTER TABLE Reviewer MODIFY COLUMN rID int auto_increment;
Query OK, 0 rows affected (0.06 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

#3

```
mysql> describe Movie;
+----------+---------+------+-----+---------+----------------+
| Field    | Type    | Null | Key | Default | Extra          |
+----------+---------+------+-----+---------+----------------+
| mID      | int(11) | NO   | PRI | NULL    | auto_increment |
| title    | text    | YES  |     | NULL    |                |
| year     | int(11) | YES  |     | NULL    |                |
| director | text    | YES  |     | NULL    |                |
+----------+---------+------+-----+---------+----------------+
4 rows in set (0.00 sec)

mysql> describe Reviewer;
+-------+---------+------+-----+---------+----------------+
| Field | Type    | Null | Key | Default | Extra          |
+-------+---------+------+-----+---------+----------------+
| rID   | int(11) | NO   | PRI | NULL    | auto_increment |
| name  | text    | YES  |     | NULL    |                |
+-------+---------+------+-----+---------+----------------+
2 rows in set (0.00 sec)

mysql> ALTER TABLE Rating ADD PRIMARY KEY (mID, rID);
Query OK, 0 rows affected (0.10 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> ALTER TABLE Rating ADD CONSTRAINT FK_mID FOREIGN KEY (mID) REFERENCES Mov
ie (mID);
Query OK, 0 rows affected (0.09 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> ALTER TABLE Rating ADD CONSTRAINT FK_rID FOREIGN KEY (rID) REFERENCES Rev
iewer (rID);
Query OK, 0 rows affected (0.06 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

#4

```
mysql> ALTER TABLE Rating CHANGE COLUMN ratingDate ratingDate timestamp NOT NULL
 DEFAULT CURRENT_TIMESTAMP;
Query OK, 0 rows affected (0.06 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> describe Rating;
+------------+-----------+------+-----+-------------------+-------+
| Field      | Type      | Null | Key | Default           | Extra |
+------------+-----------+------+-----+-------------------+-------+
| rID        | int(11)   | NO   | PRI | NULL              |       |
| mID        | int(11)   | NO   | PRI | NULL              |       |
| stars      | int(11)   | YES  |     | NULL              |       |
| ratingDate | timestamp | NO   |     | CURRENT_TIMESTAMP |       |
+------------+-----------+------+-----+-------------------+-------+
4 rows in set (0.00 sec)
```

#5

```
mysql> CREATE TRIGGER yearCheck BEFORE INSERT ON Movie FOR EACH ROW SET NEW.mID
= IF (NEW.year > 2016, 'text', NEW.mID);
Query OK, 0 rows affected (0.06 sec)

mysql> INSERT INTO Movie VALUES (1, 'title', 2012, 'director');
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO Movie VALUES (2, 'title', 2017, 'director');
ERROR 1366 (HY000): Incorrect integer value: 'text' for column 'mID' at row 1
```

Question 2 — 20 Points

Execute the following script first and then work on questions given below.

```
DROP DATABASE IF EXISTS cind110A2Script1;

CREATE SCHEMA cind110A2Script1;

USE cind110A2Script1;

CREATE TABLE hiking (
 trail CHAR(50),
area  CHAR  (50),
distance  FLOAT,
est_time  FLOAT);

SHOW TABLES;

SHOW COLUMNS FROM hiking;

INSERT INTO hiking VALUES
( 'Cedar Creek Falls', 'Upper San Diego',4.5,2.5);

INSERT INTO hiking(trail, area) VALUES
('East Mesa Loop', 'Cuyamaca Mountains');

SELECT * FROM hiking;

SET SQL_SAFE_UPDATES = 0;

UPDATE hiking
SET distance = 10.5, est_time = 5.5
WHERE trail = 'East Mesa Loop';

USE cind110A2Script1;

DELETE FROM hiking WHERE trail = 'Cedar Creek Falls';

SELECT * FROM hiking;
```

1. Write the SQL statements to insert the following values into the hiking table: (3)

trail	area	distance	est time
East Mesa Loop	Cuyamaca Mountains	10.50	10.50
Oak Canyon	NULL	3.00	NULL

2. Write the SQL statements to update the entry for the 'Oak Canyon' trail. Set the area to 'Mission Trails Regional Park' and the estimated time (est time) to 2 hours. Your table should then look like the following: (5)

trail	area	distance	est time
East Mesa Loop	Cuyamaca Mountains	10.50	10.50
Oak Canyon	Mission Trails Regional Park	3.00	2.00

3. Write the SQL statement to delete trails with a distance greater than 5 miles. (2)
4. Write the SQL statement to create a table called 'rating'. This table rates the difficulty of a hiking trail. It will have two columns: the trail name, 'trail' and the difficulty, 'difficulty'. The tail name is a string of no more than 50 characters and the difficulty is an integer (INT). (3)
5. Write the command to add another column to the hiking table called 'trailID' with Primary key constraint. (2)
6. Add another column called 'trailID' in the 'rating' table, which should be the foreign key with the table referring to the hiking table. (3)
7. What is the command to delete the rating table? (2)

Given SQL Script:

```
mysql> DROP DATABASE IF EXISTS cind110A2Script1;
Query OK, 0 rows affected, 1 warning (0.00 sec)

mysql> CREATE SCHEMA cind110A2Script1;
Query OK, 1 row affected (0.00 sec)

mysql> USE cind110A2Script1;
Database changed
mysql> CREATE Table hiking (
    -> trail CHAR(50)
    -> area CHAR (50),
    -> USE cind110A2Script1;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL se
rver version for the right syntax to use near 'area CHAR (50),
USE cind110A2Script1' at line 3
mysql> CREATE Table hiking (
    -> trail CHAR (50),
    -> area CHAR (50),
    -> distance FLOAT,
    -> est_time FLOAT);
Query OK, 0 rows affected (0.08 sec)

mysql> SHOW TABLES;
+----------------------------+
| Tables_in_cind110A2Script1 |
+----------------------------+
| hiking                     |
+----------------------------+
1 row in set (0.00 sec)

mysql> SHOW COLUMNS FROM hiking;
+----------+----------+------+-----+---------+-------+
| Field    | Type     | Null | Key | Default | Extra |
+----------+----------+------+-----+---------+-------+
| trail    | char(50) | YES  |     | NULL    |       |
| area     | char(50) | YES  |     | NULL    |       |
| distance | float    | YES  |     | NULL    |       |
| est_time | float    | YES  |     | NULL    |       |
+----------+----------+------+-----+---------+-------+
4 rows in set (0.04 sec)

mysql> INSERT INTO hiking VALUES ('Cedar Creek Falls', 'Upper San Diego', 4.5,2.5);
Query OK, 1 row affected (0.03 sec)

mysql> INSERT INTO hiking (trail, area) VALUES ('East Mesa Loop', 'Cuyamaca Mountains');
Query OK, 1 row affected (0.02 sec)

mysql> SELECT * FROM hiking;
+-------------------+--------------------+----------+----------+
| trail             | area               | distance | est_time |
+-------------------+--------------------+----------+----------+
| Cedar Creek Falls | Upper San Diego    |      4.5 |      2.5 |
| East Mesa Loop    | Cuyamaca Mountains |     NULL |     NULL |
+-------------------+--------------------+----------+----------+
2 rows in set (0.00 sec)

mysql> SET SQL_SAFE_UPDATES=0;
Query OK, 0 rows affected (0.00 sec)

mysql> UPDATE hiking
    -> SET distance =10.5, est_time=5.5
    -> WHERE trail = 'East Mesa Loop';
Query OK, 1 row affected (0.08 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> USE cind110A2Script1;
Database changed
mysql> DELETE FROM hiking WHERE trail ='Cedar Creek Falls';
Query OK, 1 row affected (0.07 sec)

mysql> SELECT *FROM hiking;
+----------------+--------------------+----------+----------+
| trail          | area               | distance | est_time |
+----------------+--------------------+----------+----------+
| East Mesa Loop | Cuyamaca Mountains |     10.5 |      5.5 |
+----------------+--------------------+----------+----------+
1 row in set (0.00 sec)
```

#1,2,3

```
mysql> INSERT INTO hiking (trail, distance) VALUES ('Oak Canyon',3.00);
Query OK, 1 row affected (0.06 sec)

mysql> update hiking set est_time =10.5 where trail = 'East Mesa Loop';
Query OK, 0 rows affected (0.06 sec)
Rows matched: 1  Changed: 0  Warnings: 0

mysql> SELECT trail, area, ROUND(distance,2) AS distance, ROUND(est_time, 2) AS est_time FROM hiking;
+---------------+-------------------+----------+----------+
| trail         | area              | distance | est_time |
+---------------+-------------------+----------+----------+
| East Mesa Loop | Cuyamaca Mountains |    10.50 |    10.50 |
| Oak Canyon     | NULL              |     3.00 |     NULL |
+---------------+-------------------+----------+----------+
2 rows in set (0.00 sec)

mysql> UPDATE hiking SET area='Mission Trails Regional Park', est_time=2 WHERE trail='Oak Canyon';
Query OK, 1 row affected (0.06 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SELECT trail, area, ROUND(distance,2) AS distance, ROUND(est_time, 2) AS est_time FROM hiking;
+---------------+---------------------------+----------+----------+
| trail         | area                      | distance | est_time |
+---------------+---------------------------+----------+----------+
| East Mesa Loop | Cuyamaca Mountains        |    10.50 |    10.50 |
| Oak Canyon     | Mission Trails Regional Park |     3.00 |     2.00 |
+---------------+---------------------------+----------+----------+
2 rows in set (0.00 sec)

mysql> DELETE FROM hiking WHERE distance >5;
Query OK, 1 row affected (0.05 sec)

mysql> SELECT trail, area, ROUND(distance,2) AS distance, ROUND(est_time, 2) AS est_time FROM hiking;
+------------+---------------------------+----------+----------+
| trail      | area                      | distance | est_time |
+------------+---------------------------+----------+----------+
| Oak Canyon | Mission Trails Regional Park |     3.00 |     2.00 |
+------------+---------------------------+----------+----------+
1 row in set (0.00 sec)
```

#4,5

```
mysql> CREATE TABLE rating (trail VARCHAR (50), difficulty INT);
Query OK, 0 rows affected (0.03 sec)

mysql> show tables;
+----------------------------+
| Tables_in_cind110A2Script1 |
+----------------------------+
| hiking                     |
| rating                     |
+----------------------------+
2 rows in set (0.00 sec)

mysql> ALTER TABLE hiking ADD COLUMN trailID INT PRIMARY KEY;
Query OK, 0 rows affected (0.13 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> show columns from hiking;
+----------+----------+------+-----+---------+-------+
| Field    | Type     | Null | Key | Default | Extra |
+----------+----------+------+-----+---------+-------+
| trail    | char(50) | YES  |     | NULL    |       |
| area     | char(50) | YES  |     | NULL    |       |
| distance | float    | YES  |     | NULL    |       |
| est_time | float    | YES  |     | NULL    |       |
| trailID  | int(11)  | NO   | PRI | NULL    |       |
+----------+----------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

#6,7

```
mysql> ALTER TABLE rating ADD COLUMN trailID INT;
Query OK, 0 rows affected (0.07 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> ALTER TABLE rating ADD PRIMARY KEY (trailID);
Query OK, 0 rows affected (0.09 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> ALTER TABLE rating ADD CONSTRAINT FK_trailID FOREIGN KEY (trailID) REFERENCES hiking(trailID);
Query OK, 0 rows affected (0.08 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> show columns from rating;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| trail      | varchar(50) | YES  |     | NULL    |       |
| difficulty | int(11)     | YES  |     | NULL    |       |
| trailID    | int(11)     | NO   | PRI | NULL    |       |
+------------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)

mysql> drop table rating;
Query OK, 0 rows affected (0.06 sec)

mysql> show columns from rating;
ERROR 1146 (42S02): Table 'cind110A2Script1.rating' doesn't exist
mysql>
```

Question 3 — 40 Points

Consider the following tables for Customer, Salesman and Order entities.

customer id	cust name	city	grade	salesman id
3002	Nick Rimando	New York	100	5001
3005	Graham Zusi	California	200	5002
3001	Brad Guzan	London		5005
3004	Fabian Johns	Paris	300	5006
3007	Brad Davis	New York	200	5001
3009	Geoff Camero	Berlin	100	5003
3008	Julian Green	London	300	5002
3003	Jozy Altidor	Moscow	200	5007

salesman id	name	city	commission
5001	James Hoog	New York	0.15
5002	Nail Knite	Paris	0.13
5005	Pit Alex	London	0.11
5006	Mc Lyon	Paris	0.14
5003	Lauson Hen		0.12
5007	Paul Adam	Rome	0.13

Order No	Purch Amt	Ord Date	Customer id	salesman id 70001
	150.5	2012-10-05	3005	5002
70009	270.65	2012-09-10	3001	5005
70002	65.26	2012-10-05	3002	5001
70004	110.5	2012-08-17	3009	5003
70007	948.5	2012-09-10	3005	5002
70005	2400.6	2012-07-27	3007	5001
70008	5760	2012-09-10	3002	5001
70010	1983.43	2012-10-10	3004	5006
70003	2480.4	2012-10-10	3009	5003
70012	250.45	2012-06-27	3008	5002
70011	75.29	2012-08-17	3003	5007
70013	3045.6	2012-04-25	3002	5001

Instruction:

You are asked to **provide the correct SQL scripts** for the following questions. You may execute your script using terminal or MySQL workbench to verify your answers. Provide screenshots of only the outputs along with your SQL scripts.

1. Write an SQL statement to prepare a list with salesman name, customer name and their cities for the salesmen and customer who belong to same city. (5)
2. Write an SQL statement to make a list with order no, purchase amount, customer name and their cities for the orders where order amount is between 500 and 2000. (5)
3. Write an SQL statement to find out which salesmen are working for which customer. (5)
4. Write an SQL statement to find the list of customers who appointed a salesman for their jobs whose commission is more than 12%. (6)
5. Write an SQL statement to find the list of customers who appointed a salesman for their jobs who does not live in same city where the customer lives, and gets a commission above 12%. (6)
6. Write an SQL statement to find the details of an order i. e. order number, order date, amount of order, which customer gives the order and which salesman works for that customer and how much commission he gets for an order. (8)
7. Write an SQL statement to make a join within the tables salesman, customer and orders such that the same column of each table will appear once and only the related rows will be returned. (5)

#1
```
SELECT Salesman.name, Customer.cust_name, Customer.city
FROM Customer
INNER JOIN Salesman ON Customer.city = Salesman.city;
```

#	name	cust_name	city
1	Pit Alex	Brad Guzan	London
2	James Hoog	Nick Rimando	New York
3	Nail Knite	Fabian Johns	Paris
4	Mc Lyon	Fabian Johns	Paris
5	James Hoog	Brad Davis	New York
6	Pit Alex	Julian Green	London

#2
```
SELECT `Order`.Order_No, `Order`.Purch_Amt, Customer.cust_name, Customer.city
FROM `Order`
INNER JOIN Customer ON Customer.customer_id = `Order`.Customer_id
WHERE `Order`.Purch_Amt BETWEEN 500 AND 2000;
```

#	Order_No	Purch_Amt	cust_name	city
1	70007	948.5	Graham Zusi	California
2	70010	1983.43	Fabian Johns	Paris

#3
```
SELECT Salesman.name, Customer.cust_name
FROM Customer, Salesman
WHERE Salesman.salesman_id=Customer.salesman_id;
```

#	name	cust_name
1	Pit Alex	Brad Guzan
2	James Hoog	Nick Rimando
3	Paul Adam	Jozy Altidor
4	Mc Lyon	Fabian Johns
5	Nail Knite	Graham Zusi
6	James Hoog	Brad Davis
7	Nail Knite	Julian Green
8	Lauson Hen	Geoff Camero

#4
```
SELECT Customer.cust_name
FROM Customer
INNER JOIN Salesman ON Customer.salesman_id = Salesman.salesman_id
WHERE Salesman.commission >0.12;
```

#	cust_name
1	Nick Rimando
2	Jozy Altidor
3	Fabian Johns
4	Graham Zusi
5	Brad Davis
6	Julian Green

#5
```
SELECT Customer.cust_name
FROM Customer
INNER JOIN Salesman ON Customer.salesman_id = Salesman.salesman_id AND Customer.city !=Salesman.city
WHERE Salesman.commission >0.12;
```

#	cust_name
1	Jozy Altidor
2	Graham Zusi
3	Julian Green

#6
```
SELECT `Order`.Order_No, `Order`.Ord_Date, `Order`.Purch_Amt, Customer.cust_name, Salesman.name,
Salesman.commission
FROM `Order`, Customer, Salesman
WHERE `Order`.Customer_id=Customer.customer_id and Customer.salesman_id = Salesman.salesman_id
ORDER BY `Order`.Order_No ASC;
```

#	Order_No	Ord_Date	Purch_Amt	cust_name	name	commission
1	70001	2012-10-05	150.5	Graham Zusi	Nail Knite	0.13
2	70002	2012-10-05	65.26	Nick Rimando	James Hoog	0.15
3	70003	2012-10-10	2480.4	Geoff Camero	Lauson Hen	0.12
4	70004	2012-08-17	110.5	Geoff Camero	Lauson Hen	0.12
5	70005	2012-07-27	2400.6	Brad Davis	James Hoog	0.15
6	70007	2012-09-10	948.5	Graham Zusi	Nail Knite	0.13
7	70008	2012-09-10	5760	Nick Rimando	James Hoog	0.15
8	70009	2012-09-10	270.65	Brad Guzan	Pit Alex	0.11
9	70010	2012-10-10	1983.43	Fabian Johns	Mc Lyon	0.14
10	70011	2012-08-17	75.29	Jozy Altidor	Paul Adam	0.13
11	70012	2012-06-27	250.45	Julian Green	Nail Knite	0.13
12	70013	2012-04-25	3045.6	Nick Rimando	James Hoog	0.15

#7
```
SELECT *
FROM Customer
NATURAL JOIN Salesman
NATURAL JOIN `Order`;
```

#	salesman_id	customer_id	city	cust_name	grade	name	commission	Order_No	Purch_Amt	Ord_Date
1	5001	3002	New York	Nick Rimando	100	James Hoog	0.15	70002	65.26	2012-10-05
2	5001	3007	New York	Brad Davis	200	James Hoog	0.15	70005	2400.6	2012-07-27
3	5001	3002	New York	Nick Rimando	100	James Hoog	0.15	70008	5760	2012-09-10
4	5005	3001	London	Brad Guzan	NULL	Pit Alex	0.11	70009	270.65	2012-09-10
5	5006	3004	Paris	Fabian Johns	300	Mc Lyon	0.14	70010	1983.43	2012-10-10
6	5001	3002	New York	Nick Rimando	100	James Hoog	0.15	70013	3045.6	2012-04-25

Question 4 — 15 Points

Consider the following Relations for a database that keeps track of student enrollment in courses and books adopted for each course.

STUDENT(Ssn, Name, Major, Bdate)

COURSE(Course#, Cname, Dept)

ENROLL(Ssn, Course#, Quarter, Grade)

BOOK ADOPTION(Course#, Quarter, Book isbn)

TEXT(Book isbn, Book title, Publisher, Author)

Having that a Relation can have zero or more Foreign keys and each Foreign key can refer to different referenced Relations. Specify all possible Foreign keys for this schema.

1. The foreign keys, Course#, in ENROLL table and BOOK_ADOPTION table are referencing the primary key, Course#, in COURSE table.

2. The foreign key, Ssn, in ENROLL table is referencing the primary key, Ssn, in STUDENT table.

3. The foreign key, Book_isbn, in BOOK_ADOPTION table is referencing the primary key, Book_isbn, in TEXT table.