IT252
Database Management System

Assignment VI

NIRAJ NANDISH

191IT234 | IT252 | 04/03/2021

Orders Schema

i. Write a query to create a view that shows for each order the salesman and customer by name.

```
> CREATE VIEW order_scname AS SELECT o.ord_no "order", c.cust_name "customer", s.name "salesman" FROM orders o, customer c, salesman s WHERE o.customer_id = c.customer_id AND o.salesman_id = s.salesman_id;
```

```
mysql> create view order_scname as select o.ord_no "order", c.cust_name "customer", s.name "salesman" from orders o, customer c, salesman s where o.customer
_id = c.customer_id and o.salesman_id = s.salesman_id;
Query OK, 0 rows affected (0.01 sec)

mysql> select * from order_scname;
+-------+----------------+------------+
| order | customer       | salesman   |
+-------+----------------+------------+
| 70002 | Nick Rimando   | James Hoog |
| 70005 | Brad Davis     | James Hoog |
| 70008 | Nick Rimando   | James Hoog |
| 70013 | Nick Rimando   | James Hoog |
| 70001 | Graham Zusi    | Nail Knite |
| 70007 | Graham Zusi    | Nail Knite |
| 70012 | Julian Green   | Nail Knite |
| 70003 | Geoff Cameron  | Lauson Hen |
| 70004 | Geoff Cameron  | Lauson Hen |
| 70009 | Brad Guzan     | Pit Alex   |
| 70010 | Fabian Johnson | Mc Lyon    |
| 70011 | Jozy Altidor   | Paul Adam  |
+-------+----------------+------------+
12 rows in set (0.00 sec)
```

ii. Write a query to create a view that finds the salesman who has the customer with the highest order of a day.

```
> CREATE VIEW schighday AS SELECT o.ord_date, s.salesman_id, s.name FROM orders o, salesman s WHERE o.salesman_id = s.salesman_id AND o.purch_amt = (SELECT MAX(purch_amt) FROM orders o1 WHERE o1.ord_date = o.ord_date);
```

```
mysql> create view schighday as
    -> select o.ord_date, s.salesman_id, s.name from orders o, salesman s where o.salesman_id = s.salesman_id and o.purch_amt = (select max(purch_amt) from
orders o1 where o1.ord_date = o.ord_date);
Query OK, 0 rows affected (0.02 sec)

mysql> select * from schighday;
+------------+-------------+------------+
| ord_date   | salesman_id | name       |
+------------+-------------+------------+
| 2012-07-27 |        5001 | James Hoog |
| 2012-09-10 |        5001 | James Hoog |
| 2012-04-25 |        5001 | James Hoog |
| 2012-10-05 |        5002 | Nail Knite |
| 2012-06-27 |        5002 | Nail Knite |
| 2012-10-10 |        5003 | Lauson Hen |
| 2012-08-17 |        5003 | Lauson Hen |
+------------+-------------+------------+
7 rows in set (0.00 sec)
```

iii. Write a query to create a view to getting a count of how many customers we have at each level of a grade.

```
> CREATE VIEW ccountpgrade AS SELECT grade, COUNT(*) "count" FROM customer GROUP BY grade;
```

```
mysql> create view ccountpgrade as select grade, count(*) "count" from customer group by grade;
Query OK, 0 rows affected (0.01 sec)

mysql> select * from ccountpgrade;
+-------+-------+
| grade | count |
+-------+-------+
|  NULL |     1 |
|   100 |     2 |
|   200 |     3 |
|   300 |     2 |
+-------+-------+
4 rows in set (0.00 sec)
```

iv. Write a query to find the salesmen of the city New York who achieved the commission more than 13%.

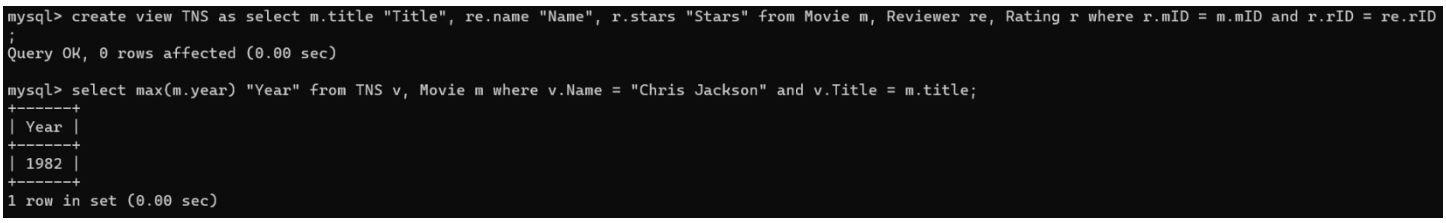
```
> SELECT * FROM salesman WHERE city = 'New York' AND commission > 0.13;
```

```
mysql> select * from salesman where city = 'New York' and commission > 0.13;
+-------------+------------+----------+------------+
| salesman_id | name       | city     | commission |
+-------------+------------+----------+------------+
|        5001 | James Hoog | New York |       0.15 |
+-------------+------------+----------+------------+
1 row in set (0.00 sec)
```

Movie Schema

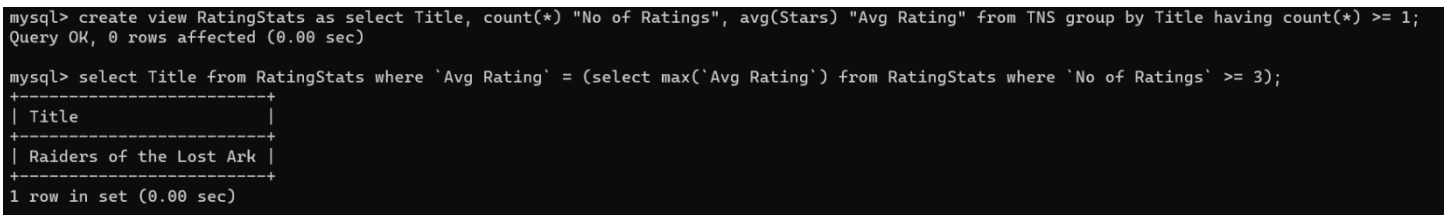
i. Create a view called TNS containing title-name-stars triples, where the movie (title) was reviewed by a reviewer (name) and received the rating (stars). Then referencing only view TNS and table Movie, write a SQL query that returns the latest year of any movie reviewed by Chris Jackson. You may assume movie names are unique.

```
> CREATE VIEW TNS AS SELECT m.title "Title", re.name "Name", r.stars "Stars" FROM Movie m, Reviewer re, Rating r WHERE r.mID = m.mID AND r.rID = re.rID;

> SELECT MAX(m.year) "Year" FROM TNS v, Movie m WHERE v.Name = "Chris Jackson" AND v.Title = m.title;
```



ii. Referencing view TNS from Exercise 1 and no other tables, create a view RatingStats containing each movie title that has at least one rating, the number of ratings it received, and its average rating. Then referencing view RatingStats and no other tables, write a SQL query to find the title of the highest-average-rating movie with at least three ratings.

```
> CREATE VIEW RatingStats AS SELECT Title, COUNT(*) "No of Ratings", AVG(Stars) "Avg Rating" FROM TNS GROUP BY Title HAVING COUNT(*) >= 1;

> SELECT Title FROM RatingStats WHERE `Avg Rating` = (SELECT MAX(`Avg Rating`) FROM RatingStats WHERE `No of Ratings` >= 3);
```



iii. Create a view Favorites containing rID-mID pairs, where the reviewer with rID gave the movie with mID the highest rating he or she gave any movie. Then referencing only view Favorites and tables Movie and Reviewer, write a SQL query to return reviewer-reviewer-movie triples where the two (different) reviewers have the movie as their favorite. Return each pair once, i.e., don't return a pair and its inverse.

```
> CREATE VIEW Favorites AS SELECT rID, mID FROM Rating r WHERE stars
= (SELECT MAX(stars) FROM Rating r1 WHERE r1.rID = r.rID);

> SELECT r1.name, r2.name, m.title FROM Favorites f1, Favorites f2,
Reviewer r1, Reviewer r2, Movie m WHERE f1.mID = f2.mID AND m.mID =
f1.mID AND r1.rID = f1.rID AND r2.rID = f2.rID AND r1.name > r2.name;
```

```
mysql> create view Favorites as select rID, mID from Rating r where stars = (select max(stars) from Rating r1 where r1.rID = r.rID);
Query OK, 0 rows affected (0.00 sec)

mysql> select r1.name, r2.name, m.title from Favorites f1, Favorites f2, Reviewer r1, Reviewer r2, Movie m where f1.mID = f2.mID and m.mID = f1.mID and r1.r
ID = f1.rID and r2.rID = f2.rID and r1.name > r2.name;
+-----------------+----------------+------------------------+
| name            | name           | title                  |
+-----------------+----------------+------------------------+
| Elizabeth Thomas | Daniel Lewis    | Snow White             |
| Chris Jackson    | Brittany Harris | Raiders of the Lost Ark |
| Sarah Martinez   | Mike Anderson   | Gone with the Wind     |
+-----------------+----------------+------------------------+
3 rows in set (0.00 sec)
```