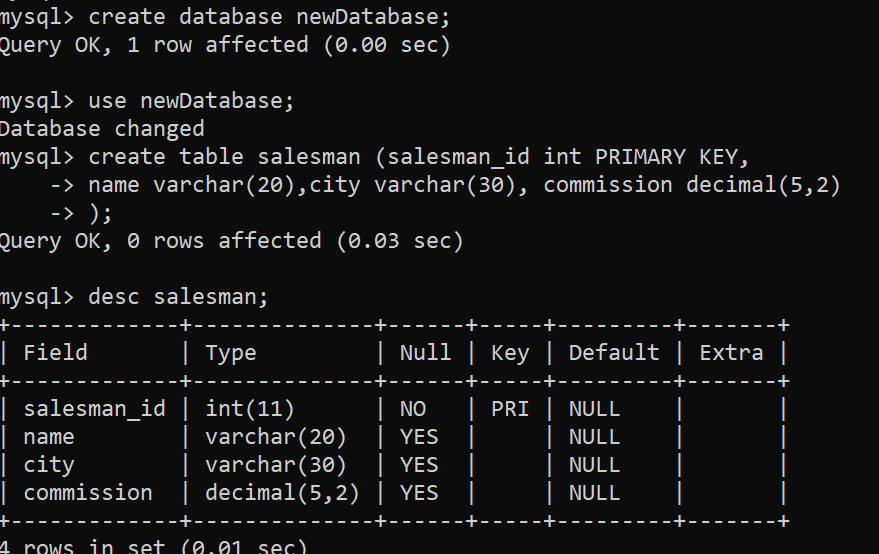
**Practical – 1 :**

create table salesman (salesman\_id int PRIMARY KEY,

name varchar(20),city varchar(30), commission decimal(5,2)

);



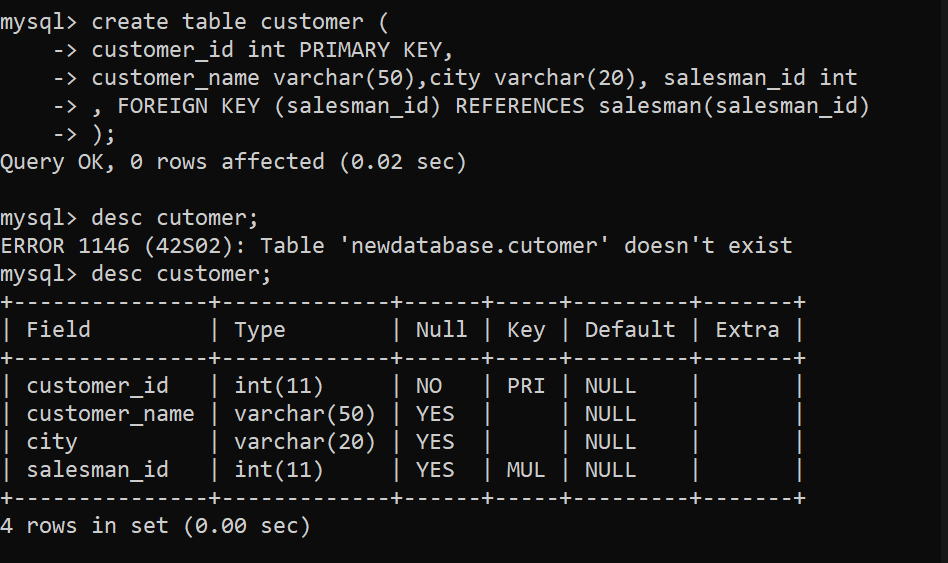
create table customer (

customer\_id int PRIMARY KEY,

customer\_name varchar(50),city varchar(20), salesman\_id int

, FOREIGN KEY (salesman\_id) REFERENCES salesman(salesman\_id)

);



create table orders (

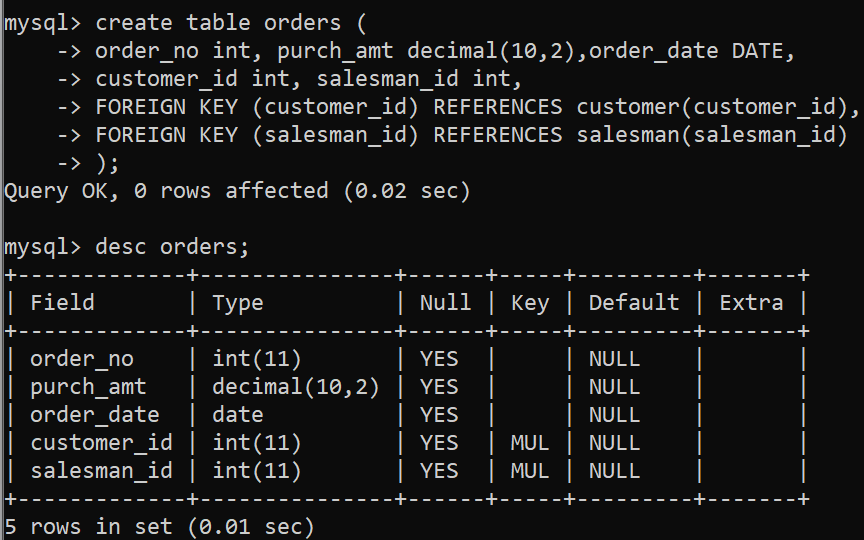
order\_no int, purch\_amt decimal(10,2),order\_date DATE,

customer\_id int, salesman\_id int,

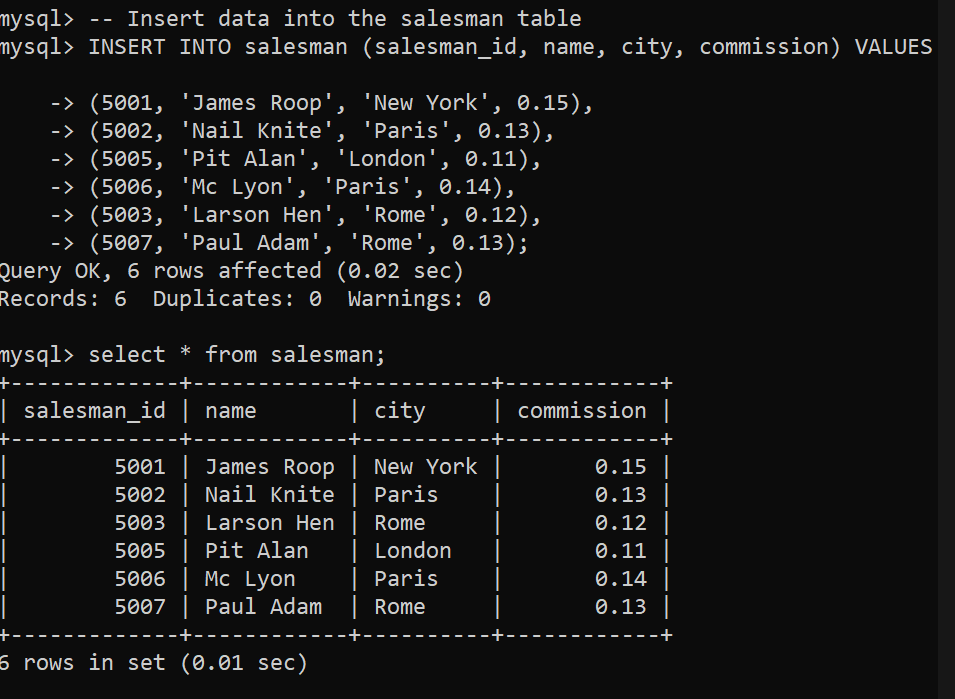
FOREIGN KEY (customer\_id) REFERENCES customer(customer\_id),

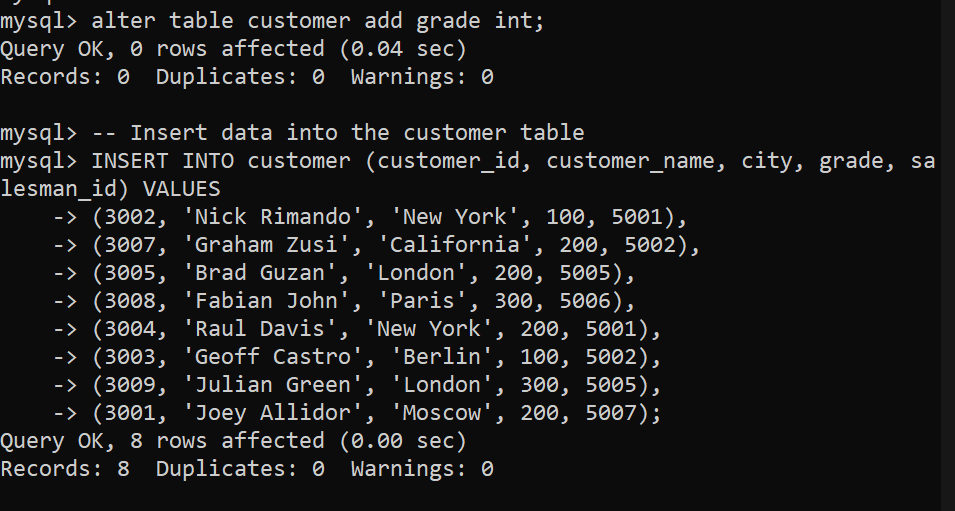
FOREIGN KEY (salesman\_id) REFERENCES salesman(salesman\_id)

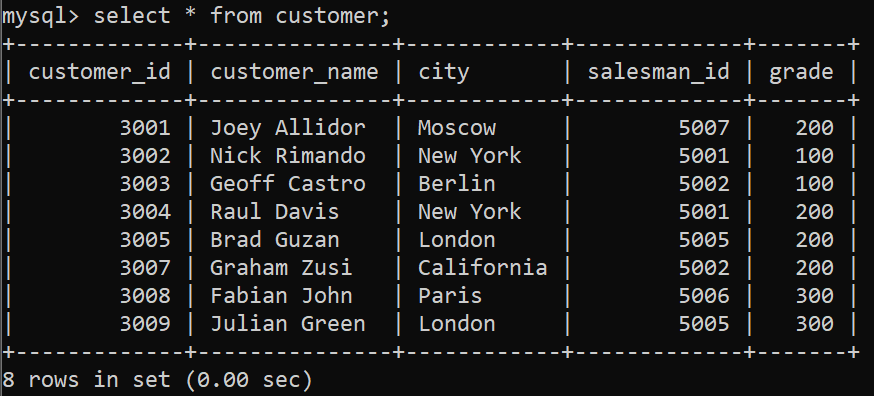
);



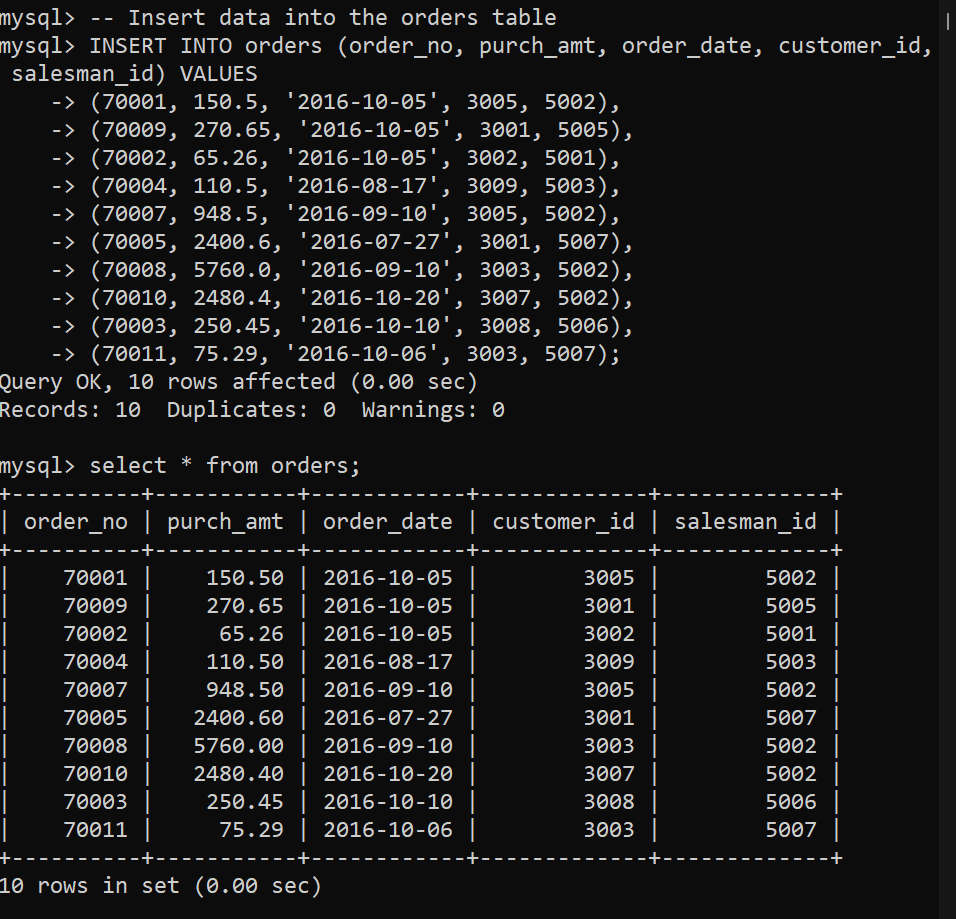
-- Insert data into the salesman table  
INSERT INTO salesman (salesman\_id, name, city, commission) VALUES  
(5001, 'James Roop', 'New York', 0.15),  
(5002, 'Nail Knite', 'Paris', 0.13),  
(5005, 'Pit Alan', 'London', 0.11),  
(5006, 'Mc Lyon', 'Paris', 0.14),  
(5003, 'Larson Hen', 'Rome', 0.12),  
(5007, 'Paul Adam', 'Rome', 0.13);

  
  
-- Insert data into the customer table  
INSERT INTO customer (customer\_id, customer\_name, city, grade, salesman\_id) VALUES  
(3002, 'Nick Rimando', 'New York', 100, 5001),  
(3007, 'Graham Zusi', 'California', 200, 5002),  
(3005, 'Brad Guzan', 'London', 200, 5005),  
(3008, 'Fabian John', 'Paris', 300, 5006),  
(3004, 'Raul Davis', 'New York', 200, 5001),  
(3003, 'Geoff Castro', 'Berlin', 100, 5002),  
(3009, 'Julian Green', 'London', 300, 5005),  
(3001, 'Joey Allidor', 'Moscow', 200, 5007);



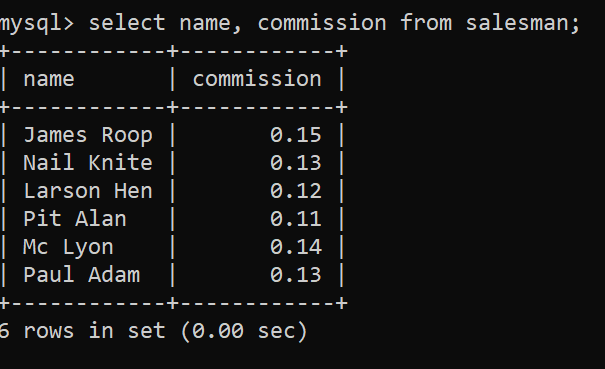


-- Insert data into the orders table  
INSERT INTO orders (order\_no, purch\_amt, order\_date, customer\_id, salesman\_id) VALUES  
(70001, 150.5, '2016-10-05', 3005, 5002),  
(70009, 270.65, '2016-10-05', 3001, 5005),  
(70002, 65.26, '2016-10-05', 3002, 5001),  
(70004, 110.5, '2016-08-17', 3009, 5003),  
(70007, 948.5, '2016-09-10', 3005, 5002),  
(70005, 2400.6, '2016-07-27', 3001, 5007),  
(70008, 5760.0, '2016-09-10', 3003, 5002),  
(70010, 2480.4, '2016-10-20', 3007, 5002),  
(70003, 250.45, '2016-10-10', 3008, 5006),  
(70011, 75.29, '2016-10-06', 3003, 5007);



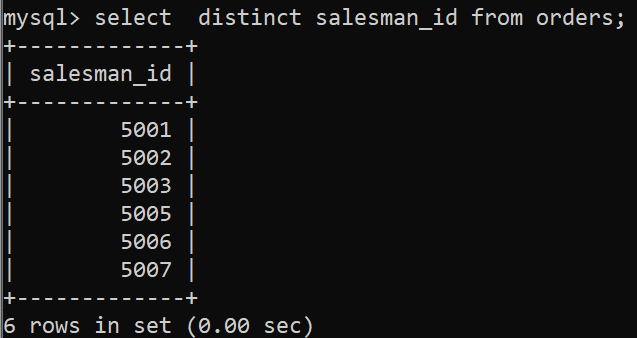
1. Display name and commission for all the salesmen.

select name, commission from salesman;



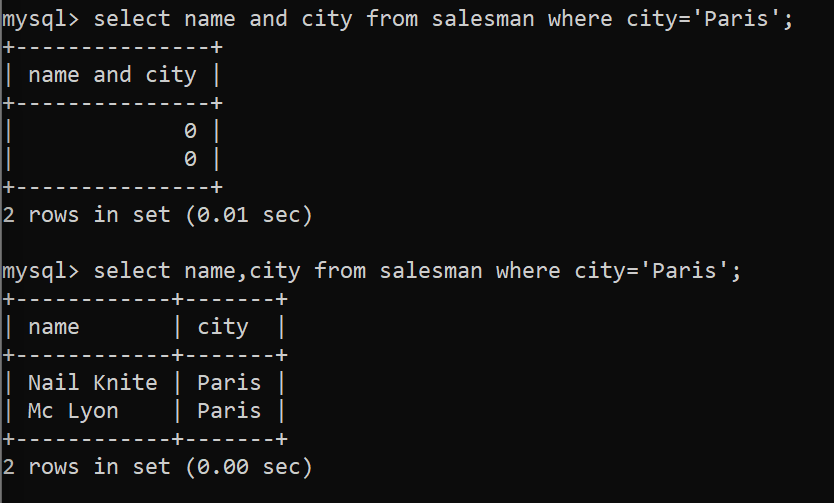
1. Retrieve salesman id of all salesmen from orders table without any repeats.

select distinct salesman\_id from orders;



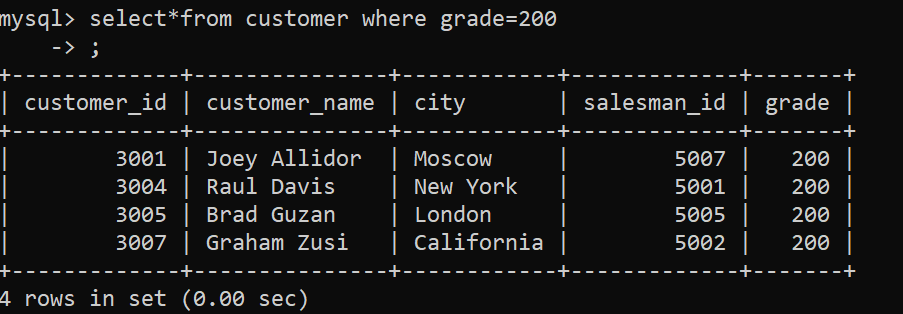
1. Display names and city of salesman, who belongs to the city of Paris.

select name,city from salesman where city='Paris';



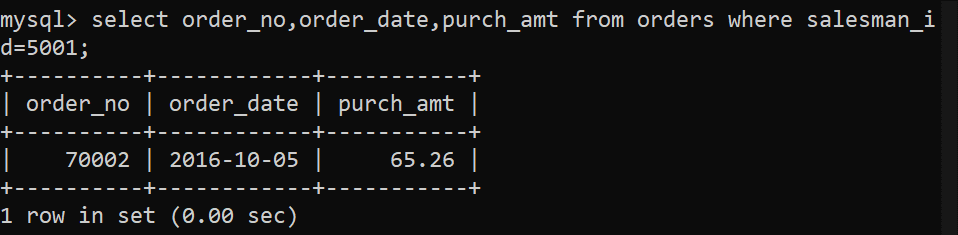
1. Display all the information for those customers with a grade of 200.

select\*from customer where grade=200 ;



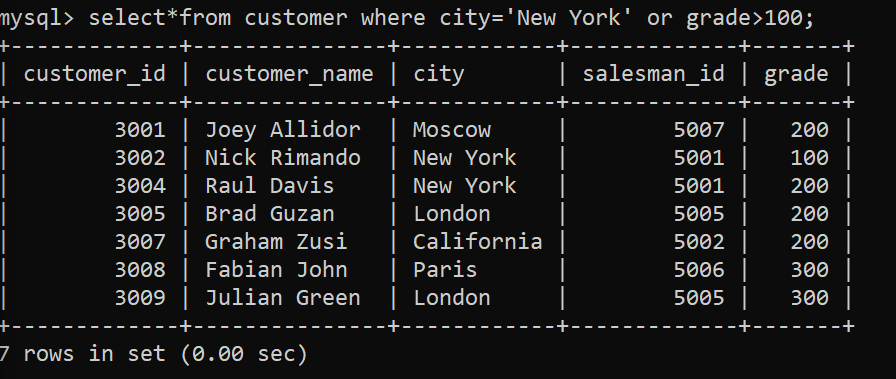
1. Display the order number, order date and the purchase amount for order(s) which will be delivered by the salesman with ID 5001

select order\_no,order\_date,purch\_amt from orders where salesman\_id=5001;



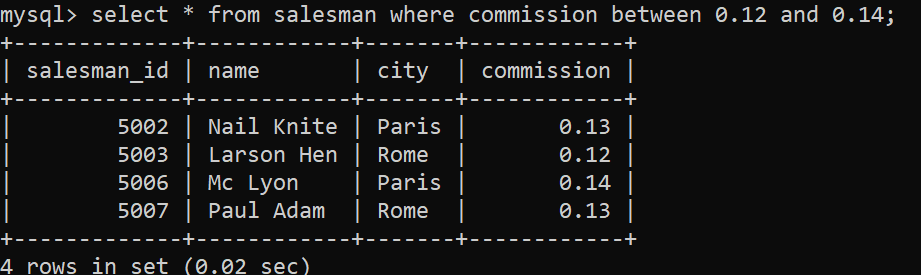
1. Display all the customers, who are either belongs to the city New York or not had a grade above 100.

select\*from customer where city='New York' or grade>100;



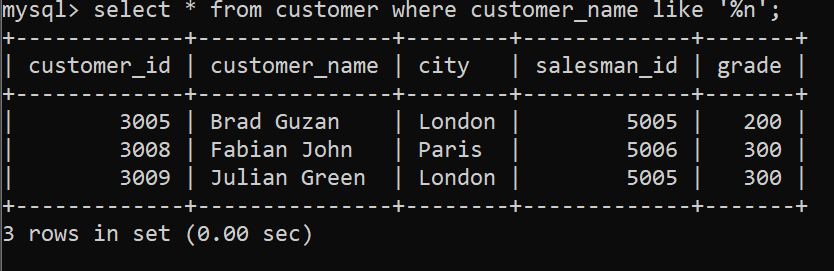
1. Find those salesmen with all information who gets the commission within a range of 0.12 and 0.14.

select \* from salesman where commission between 0.12 and 0.14;



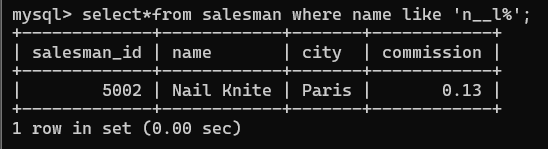
1. Find all those customers with all information whose names are ending with the letter 'n'.

select \* from customer where customer\_name like '%n';



1. Find those salesmen with all information whose name containing the 1st character is 'N' and the 4th character is 'l' and rests may be any character.

select\*from salesman where name like 'n\_\_l%';



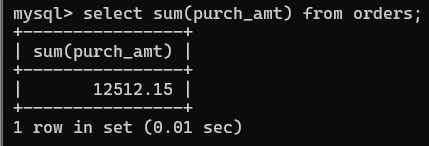
1. Find that customer with all information who does not get any grade except NULL.

select\*from customer where grade is NULL;



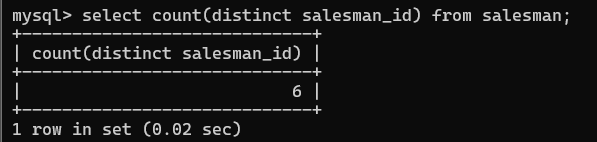
1. Find the total purchase amount of all orders.

select sum(purch\_amt) from orders;



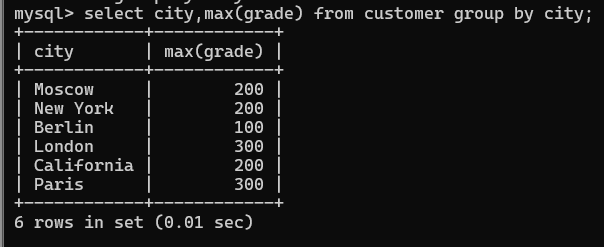
1. Find the number of salesman currently listing for all of their customers.

select count(distinct salesman\_id) from salesman;



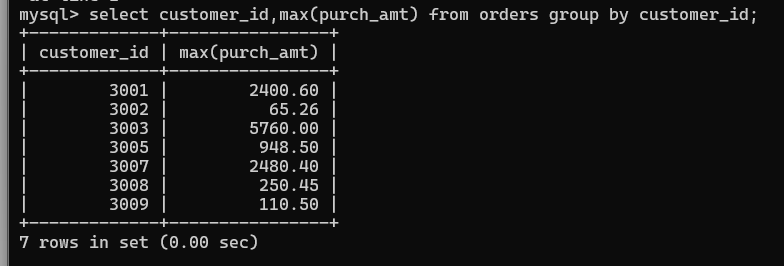
1. Find the highest grade for each of the cities of the customers.

select city,max(grade) from customer group by city;



1. Find the highest purchase amount ordered by each customer with their ID and highest purchase amount.

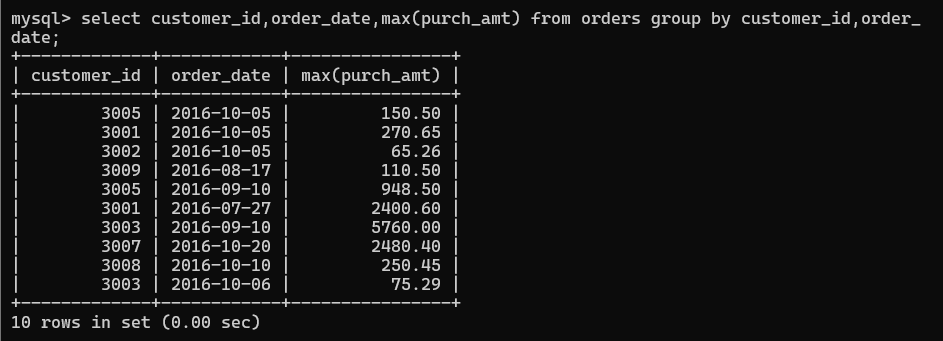
select customer\_id,max(purch\_amt) from orders group by customer\_id;



1. Find the highest purchase amount ordered by each customer on a particular date with their ID, order date and highest purchase amount.

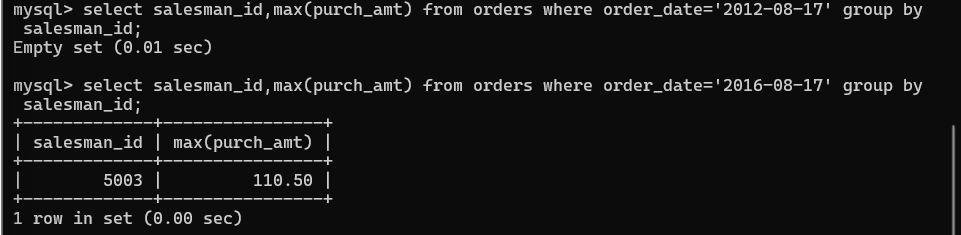
select customer\_id,order\_date,max(purch\_amt) from orders group by customer\_id,order\_

date;



1. Find the highest purchase amount on a date '2012-08-17' for each salesman with their ID.

select salesman\_id,max(purch\_amt) from orders where order\_date='2012-08-17' group by salesman\_id;

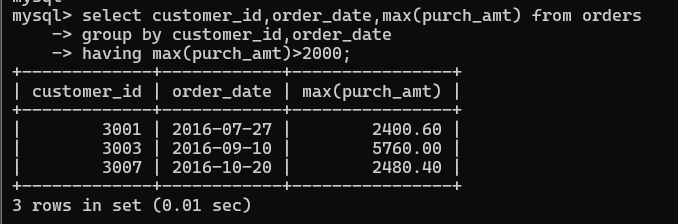


1. Find the highest purchase amount with their customer ID and order date, for only those customers who have the highest purchase amount in a day is more than 2000.

select customer\_id,order\_date,max(purch\_amt) from orders

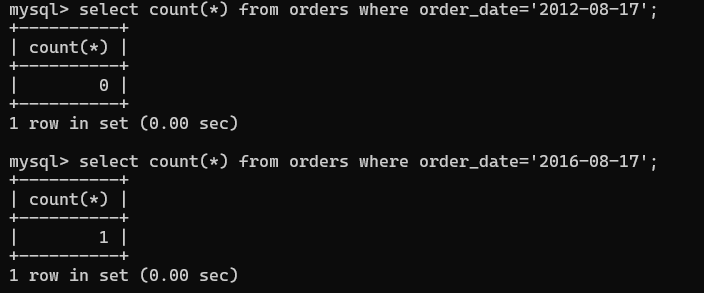
-> group by customer\_id,order\_date

-> having max(purch\_amt)>2000;



1. Write a SQL statement that counts all orders for a date August 17th, 2012.

select count(\*) from orders where order\_date='2012-08-17';



1. Count the customers with grades above Lonodon’s average.

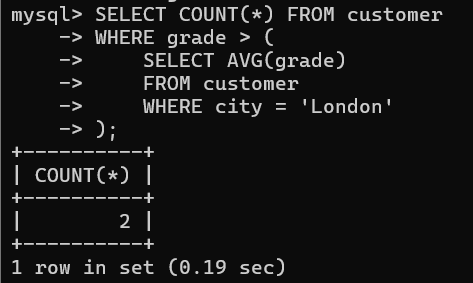
SELECT COUNT(\*) FROM customer

WHERE grade > (

SELECT AVG(grade)

FROM customer

WHERE city = 'London' );



1. Find the name and numbers of all salesmen who had more than one customer.

(USING SUBQUERY)

SELECT name, salesman\_id

FROM salesman

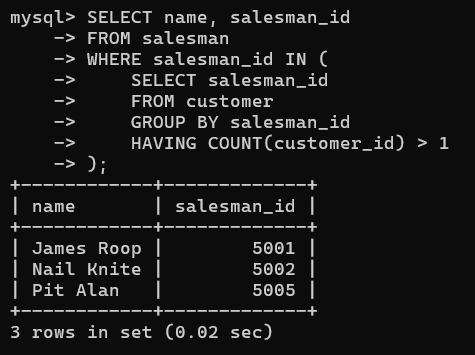
WHERE salesman\_id IN (

SELECT salesman\_id

FROM customer

GROUP BY salesman\_id

HAVING COUNT(customer\_id) > 1 );



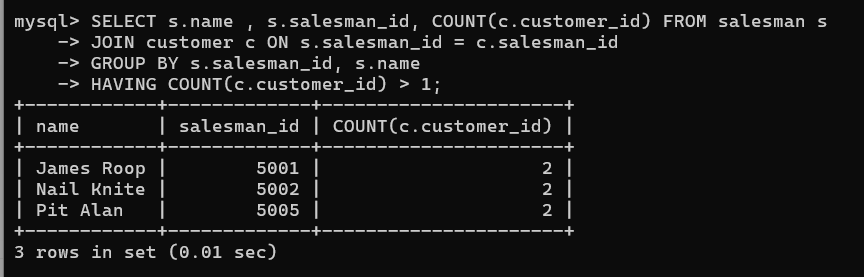
(USING JOINS)

SELECT s.name , s.salesman\_id, COUNT(c.customer\_id) FROM salesman s

JOIN customer c ON s.salesman\_id = c.salesman\_id

GROUP BY s.salesman\_id, s.name

HAVING COUNT(c.customer\_id) > 1;



1. List all salesmen and indicate those who have and don’t have customers in their cities (Use UNION operation.)

-- Salesmen who have customers in their cities

SELECT s.name, s.salesman\_id, s.city, 'Has Customers' AS status

FROM salesman s

JOIN customer c ON s.city = c.city

UNION

-- Salesmen who don't have customers in their cities

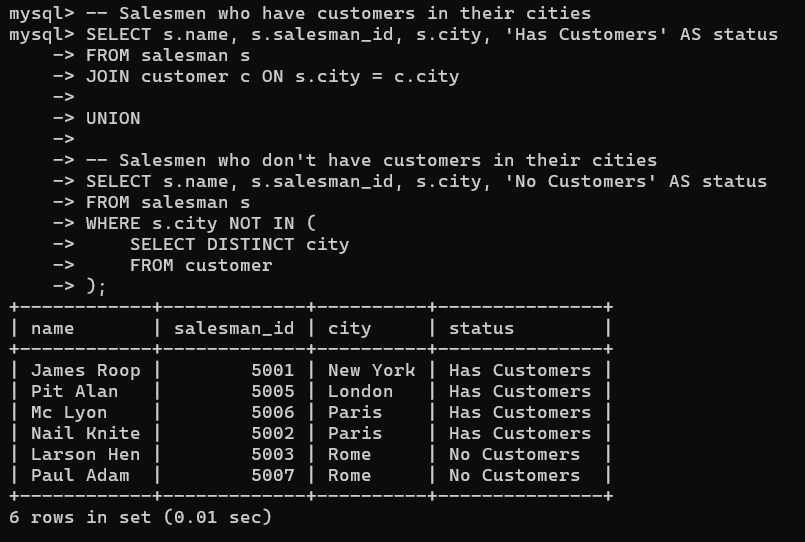
SELECT s.name, s.salesman\_id, s.city, 'No Customers' AS status

FROM salesman s

WHERE s.city NOT IN (

SELECT DISTINCT city

FROM customer );



1. Create a view that finds the salesman who has the customer with the highest order of a

day.

CREATE VIEW highest\_order\_salesman AS

SELECT

o.order\_date,

o.purch\_amt ,

o.customer\_id,

c.customer\_name,

o.salesman\_id,

s.name

FROM orders o

JOIN customer c ON o.customer\_id = c.customer\_id

JOIN salesman s ON o.salesman\_id = s.salesman\_id

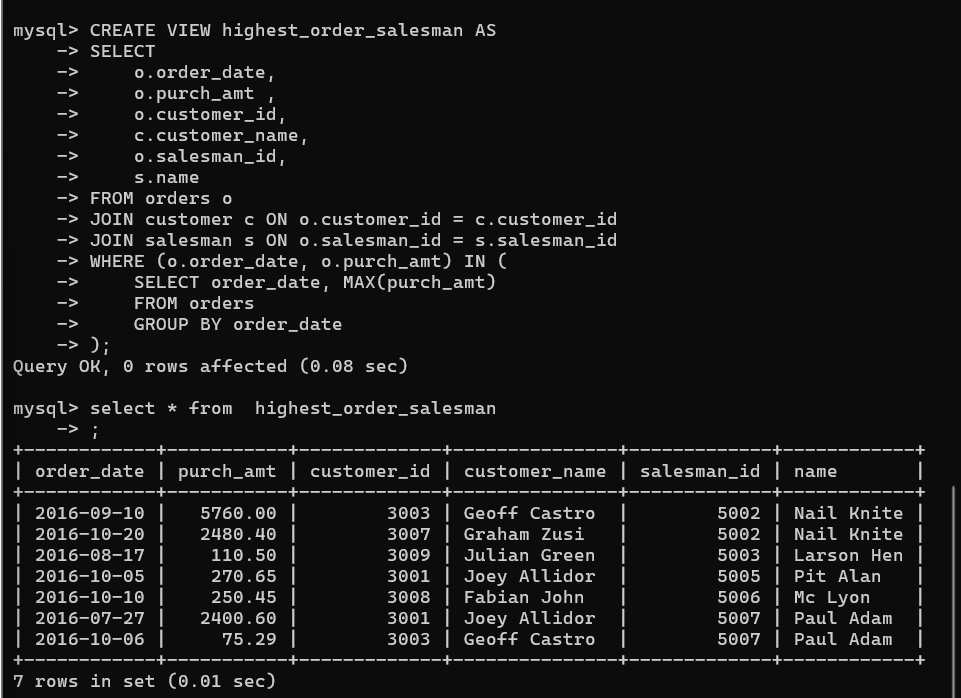
WHERE (o.order\_date, o.purch\_amt) IN (

SELECT order\_date, MAX(purch\_amt)

FROM orders

GROUP BY order\_date );

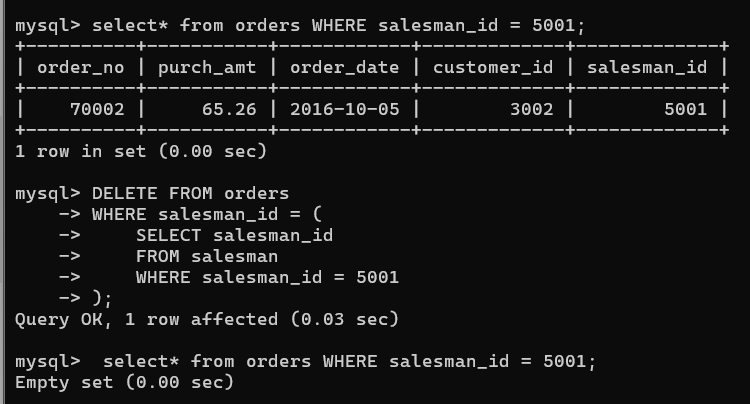
select \* from highest\_order\_salesman ;



1. Demonstrate the DELETE operation by removing salesman with id 5001. All his orders

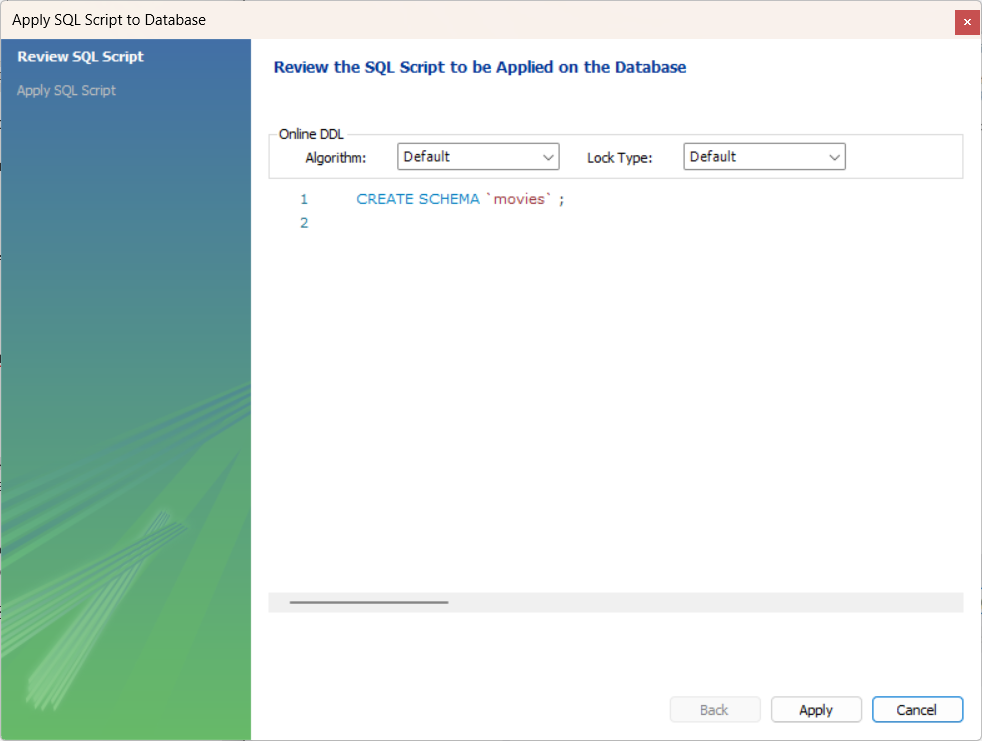
must also be deleted

DELETE FROM orders WHERE salesman\_id = ( SELECT salesman\_id FROM salesman WHERE salesman\_id = 5001 );

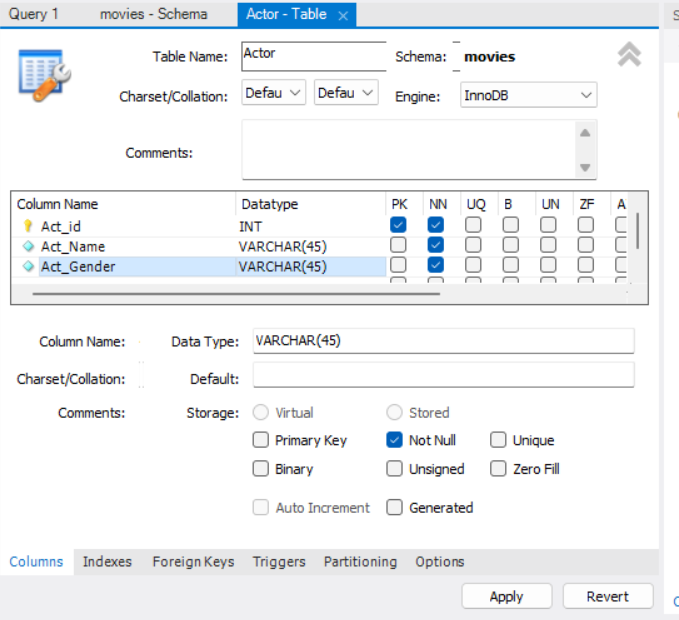


**Practical – 2 :**

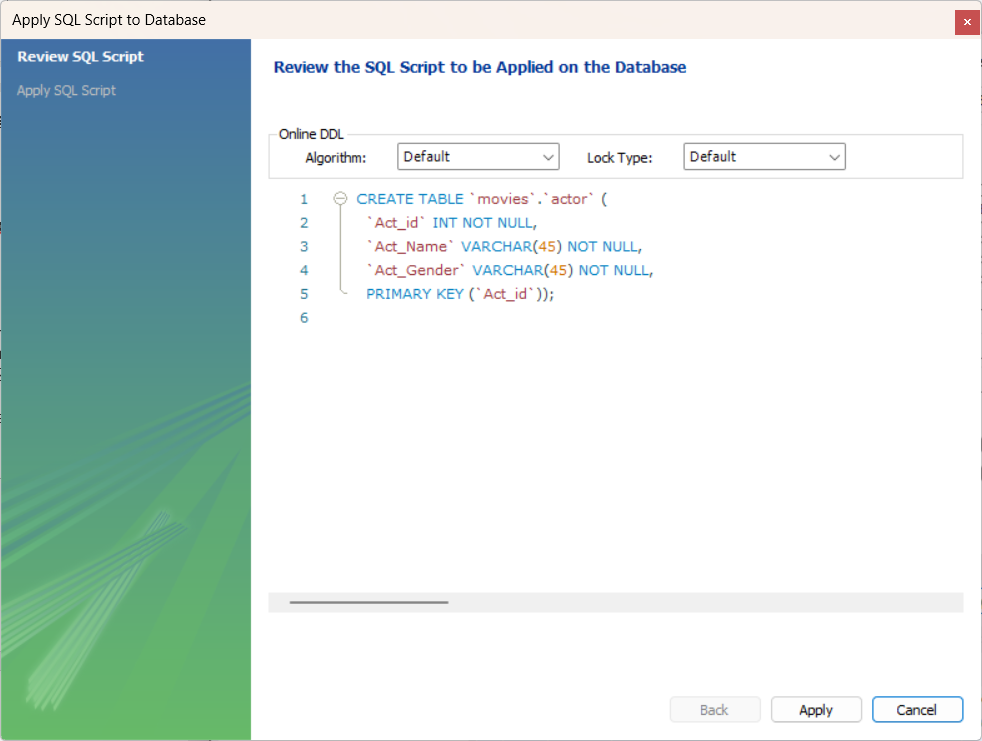
Creating Database “Movies”



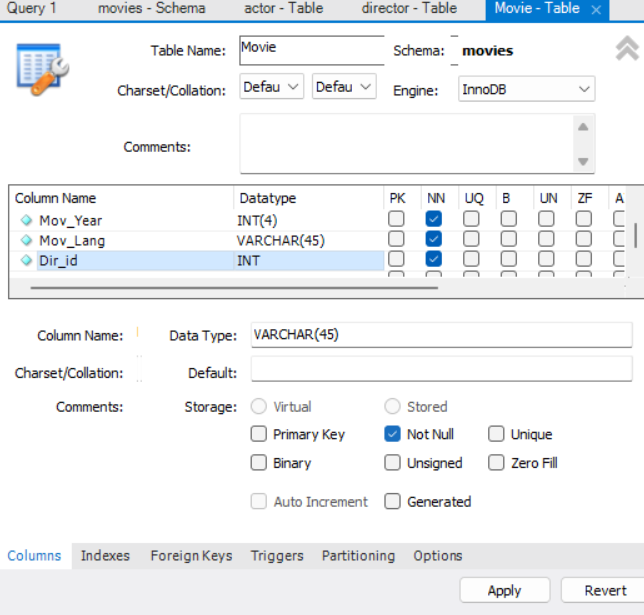
Creating table “Actor”



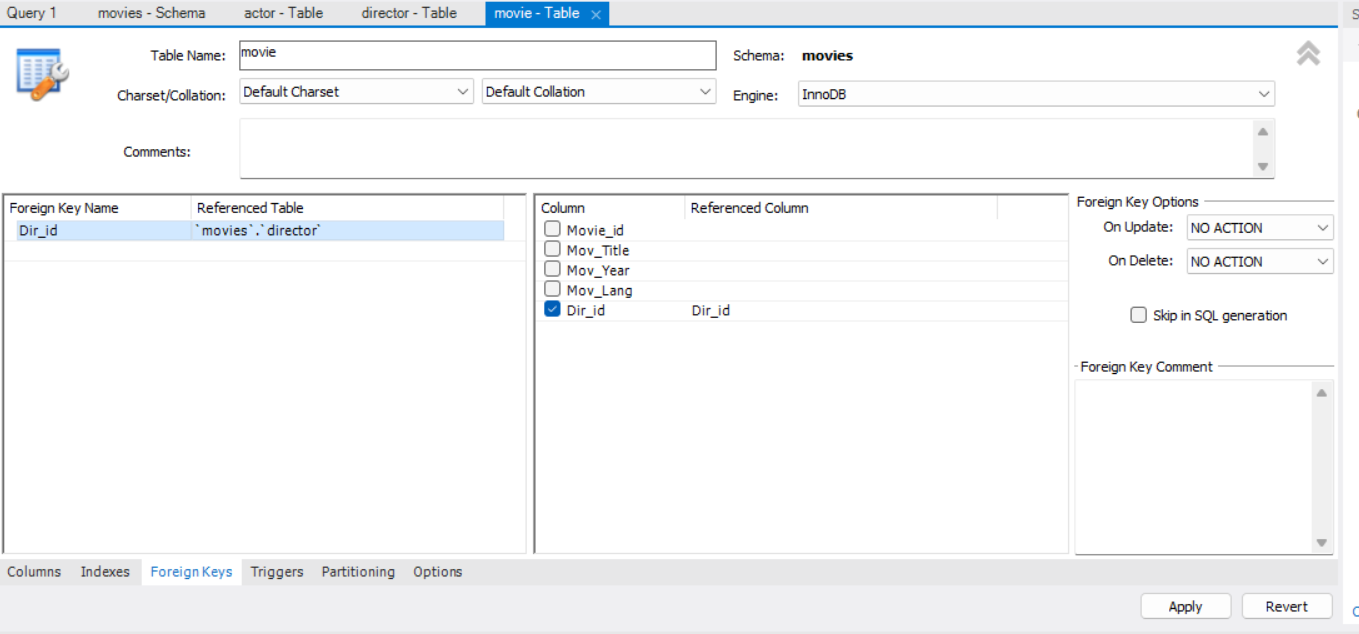
Executing the command



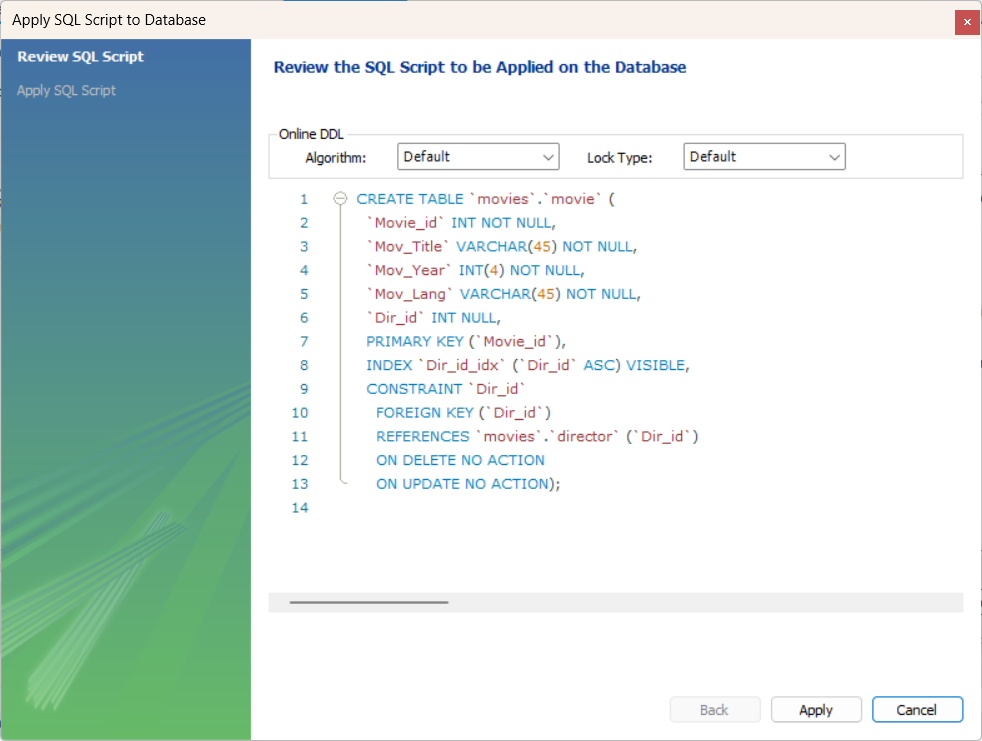
Creating table “Movie”



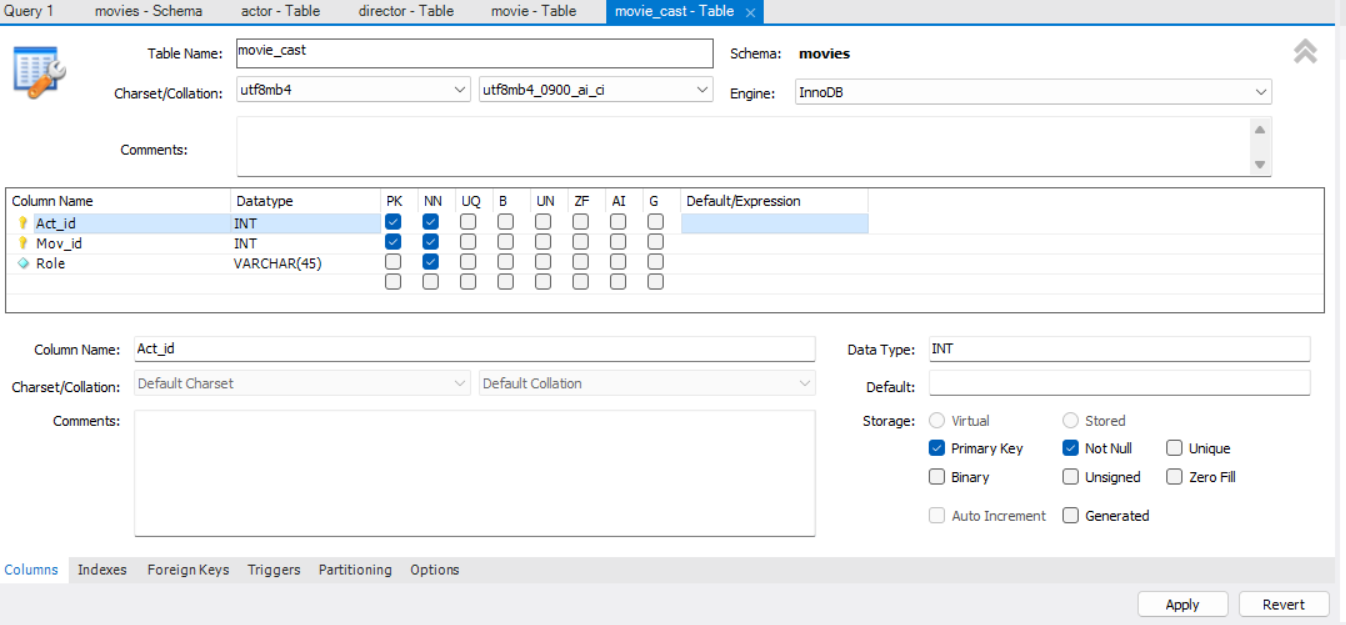
Adding a foreign key



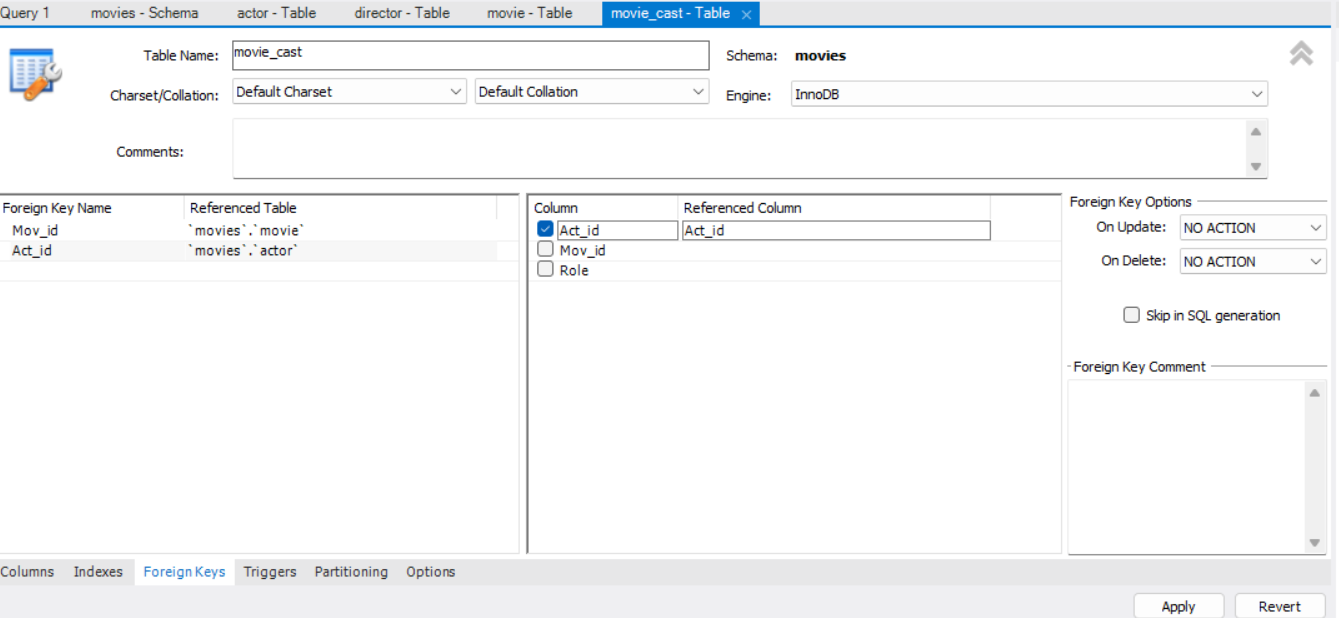
Executing the command



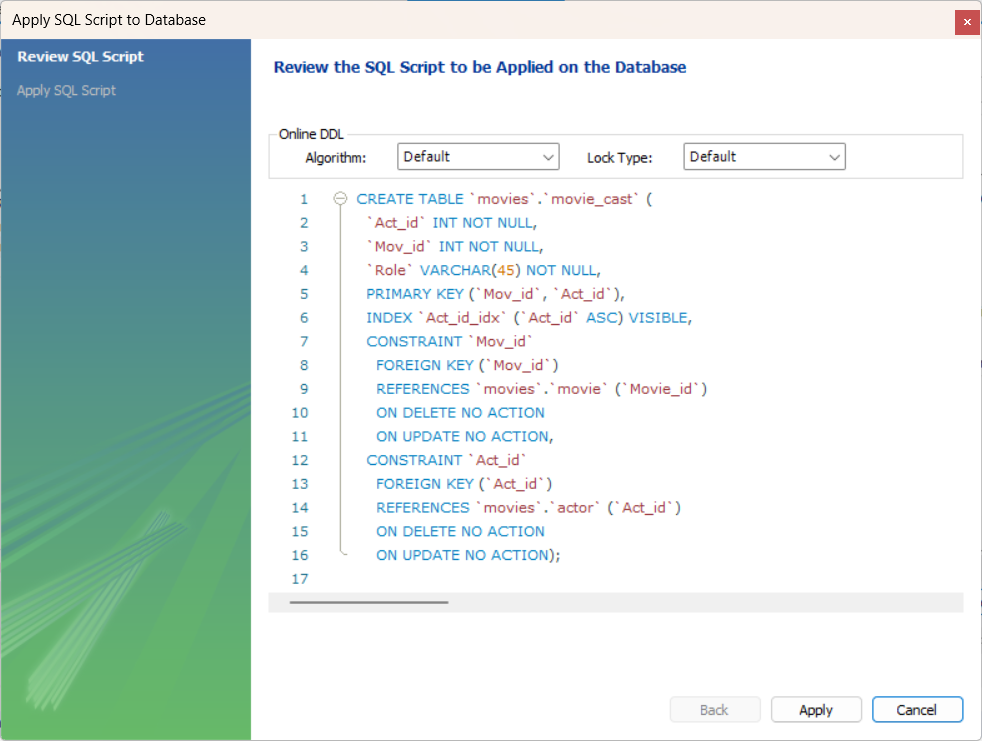
Creating table “Movie\_Cast”



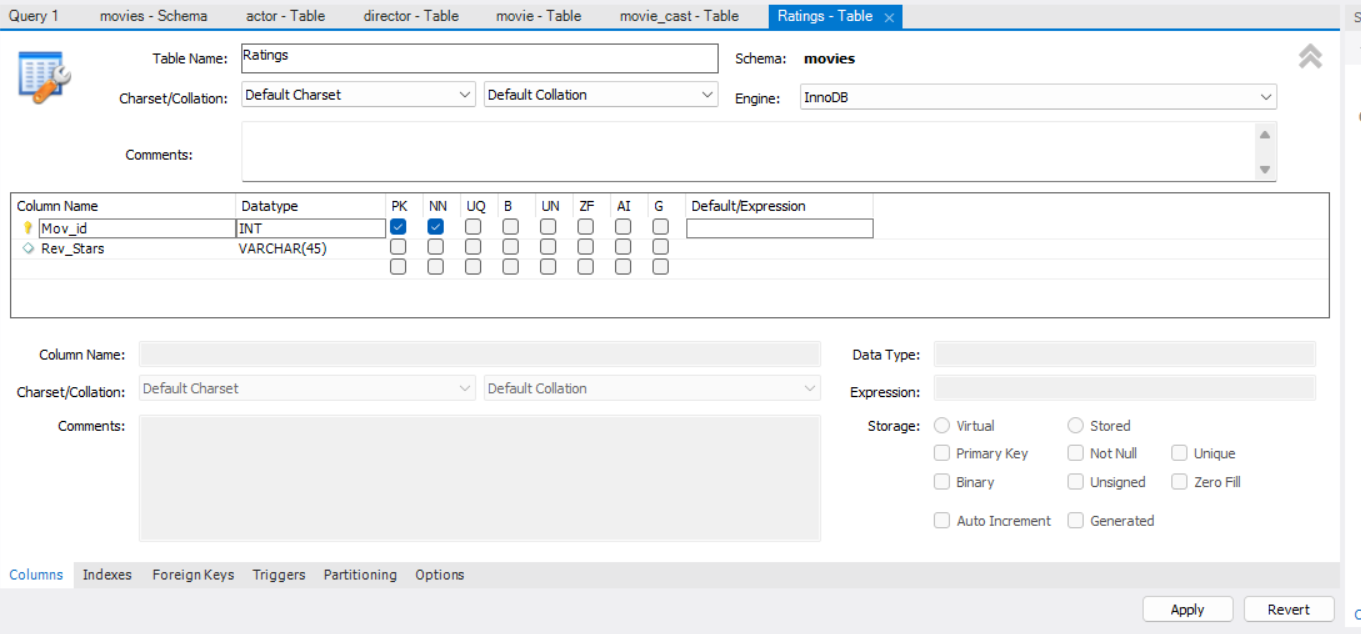
Adding a foreign key



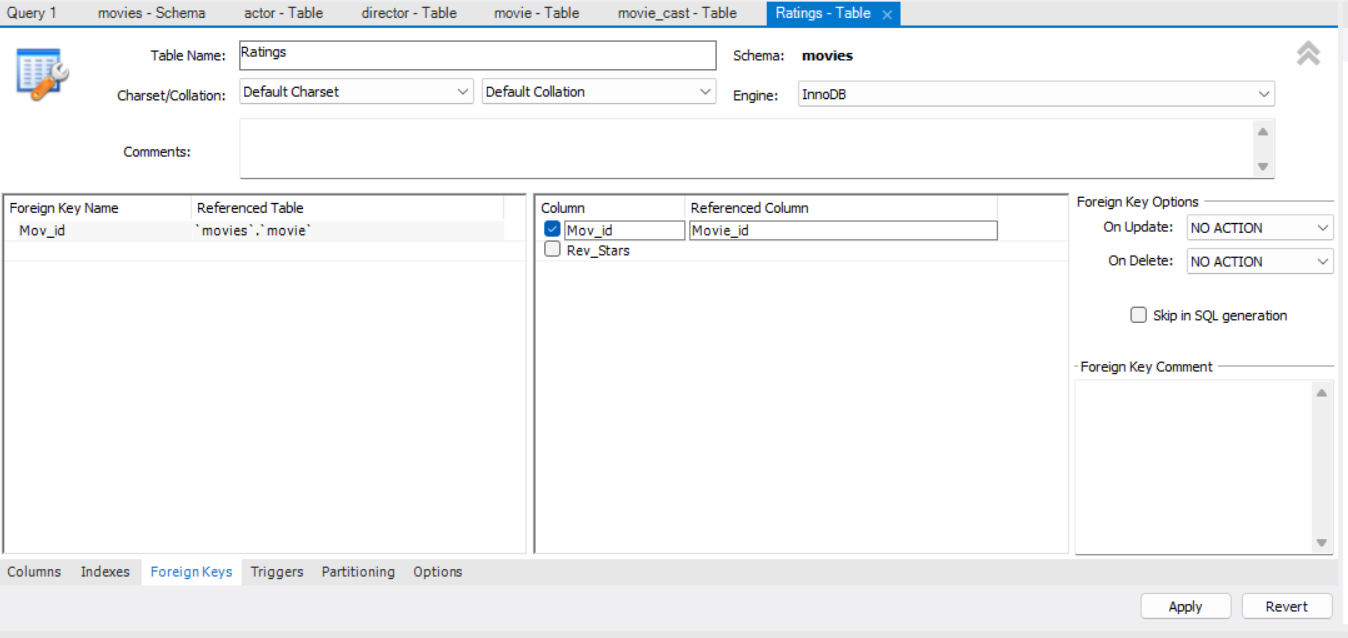
Executing the command



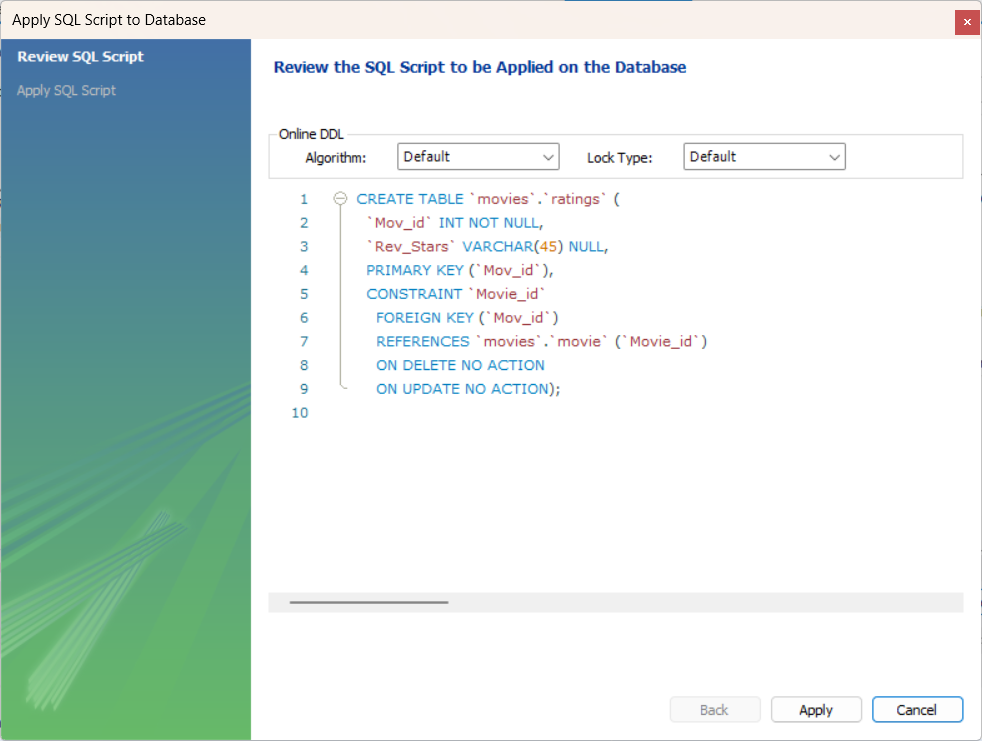
Creating table “Ratings”



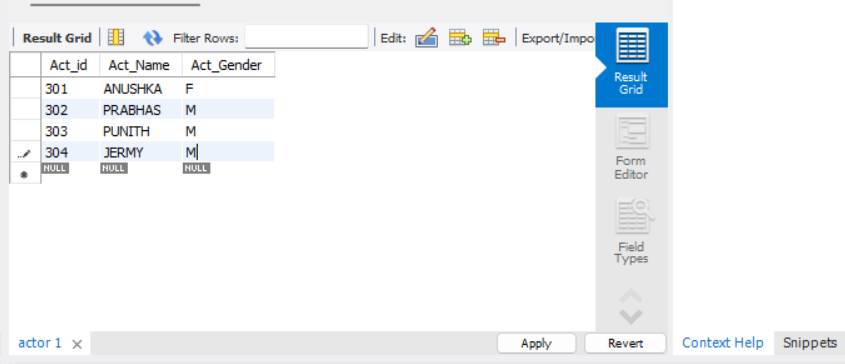
Adding a foreign key



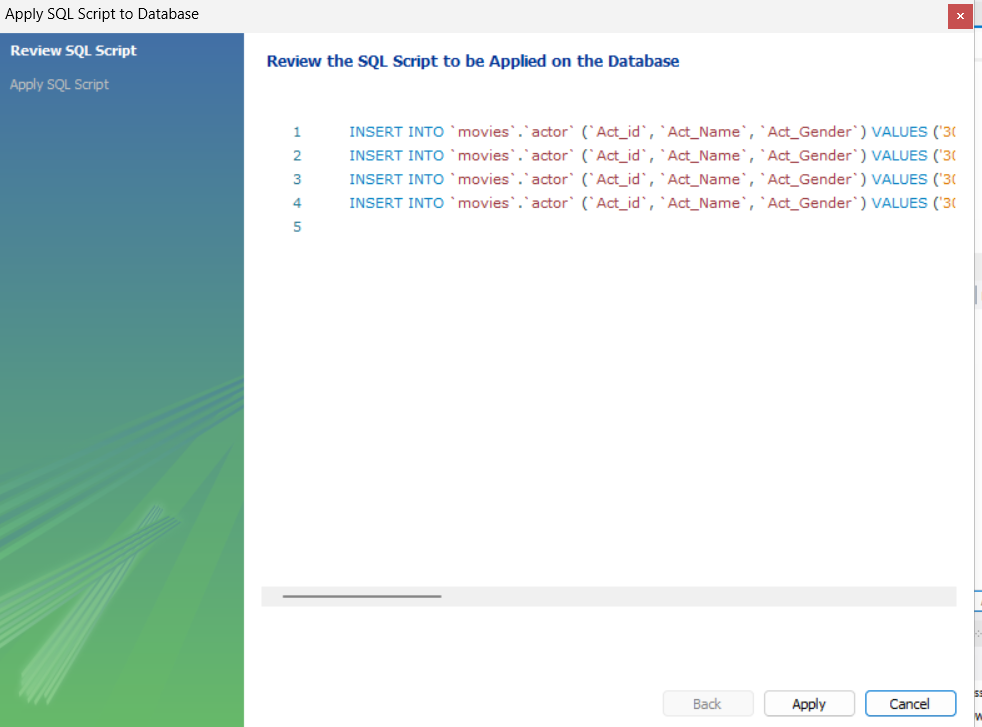
Executing the command



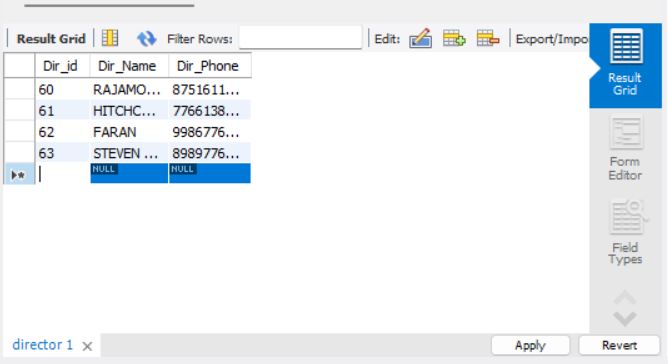
Inserting data into “Actor”



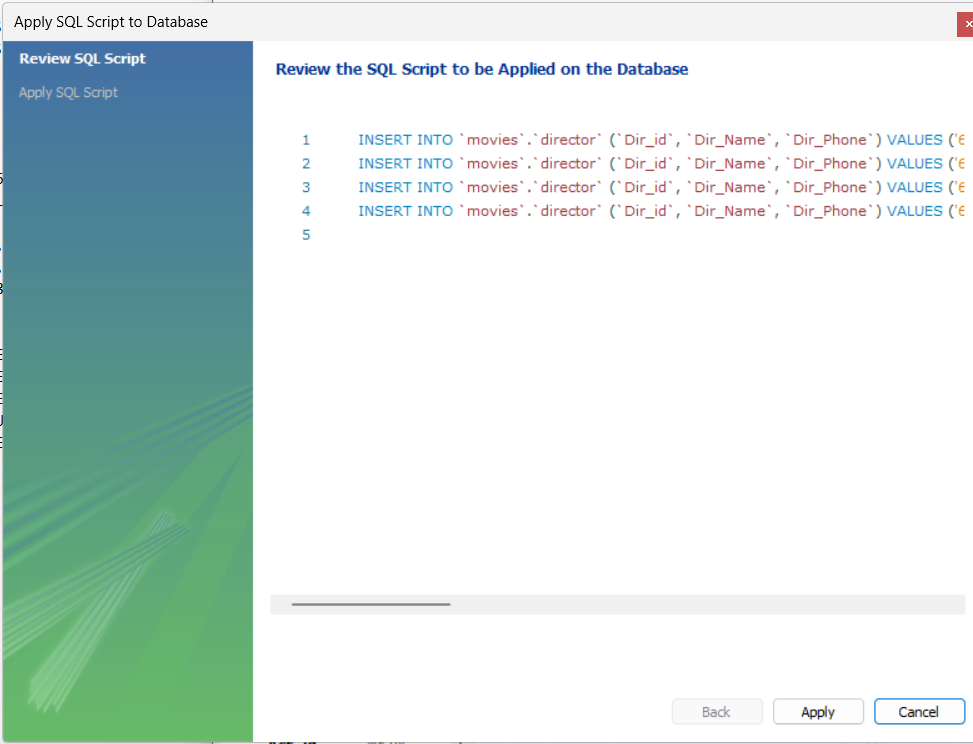
Executing the command



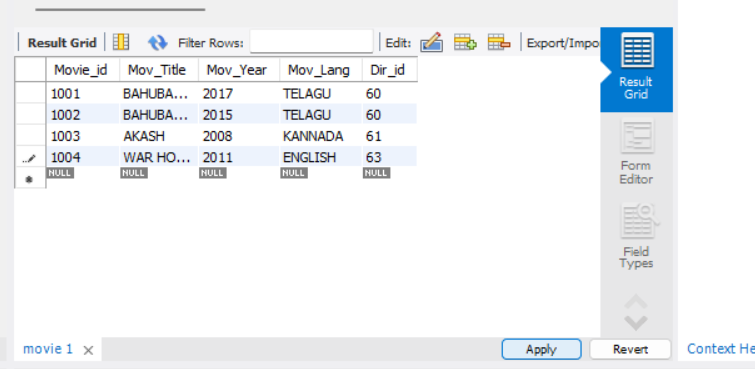
Inserting data into “Director”



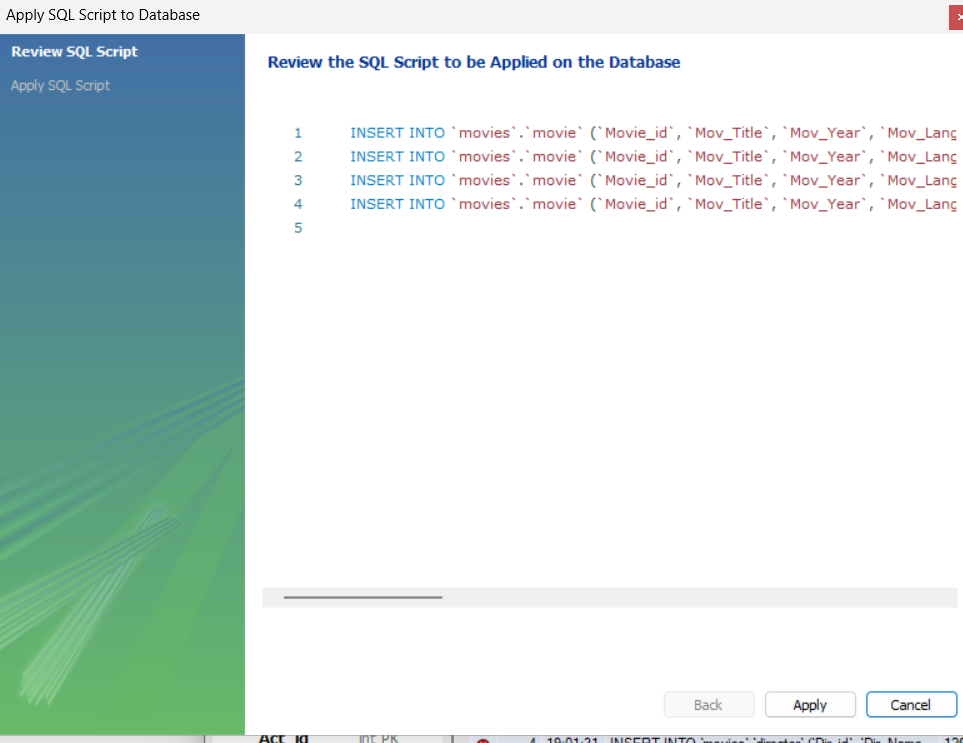
Executing the command



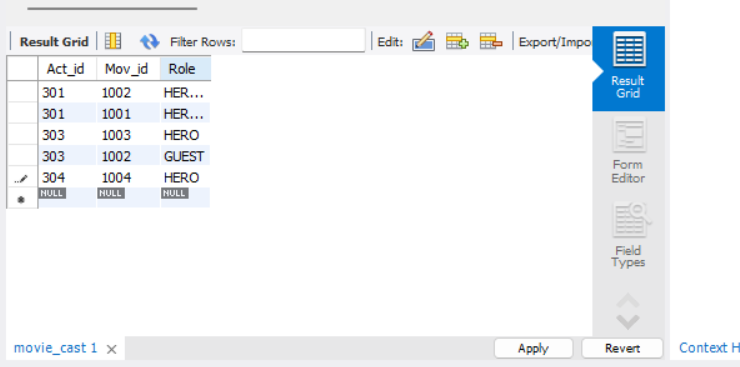
Inserting data into “Movie”



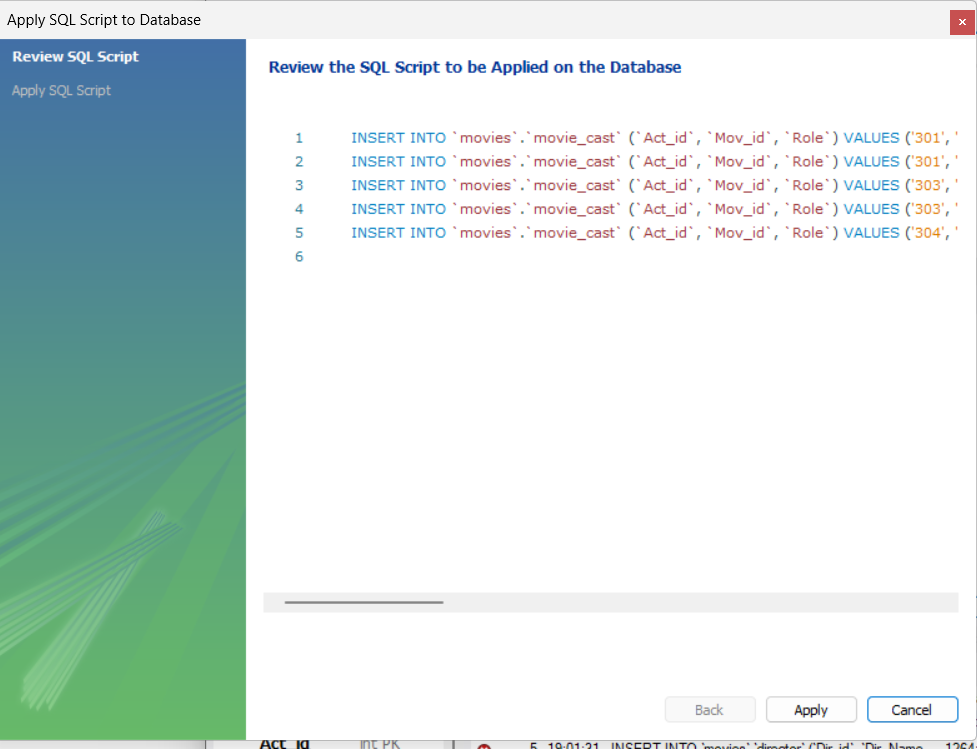
Executing the command



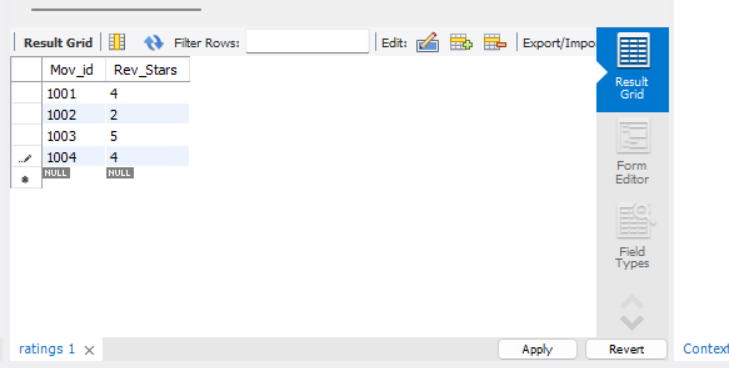
Inserting data into “Movie\_Cast”



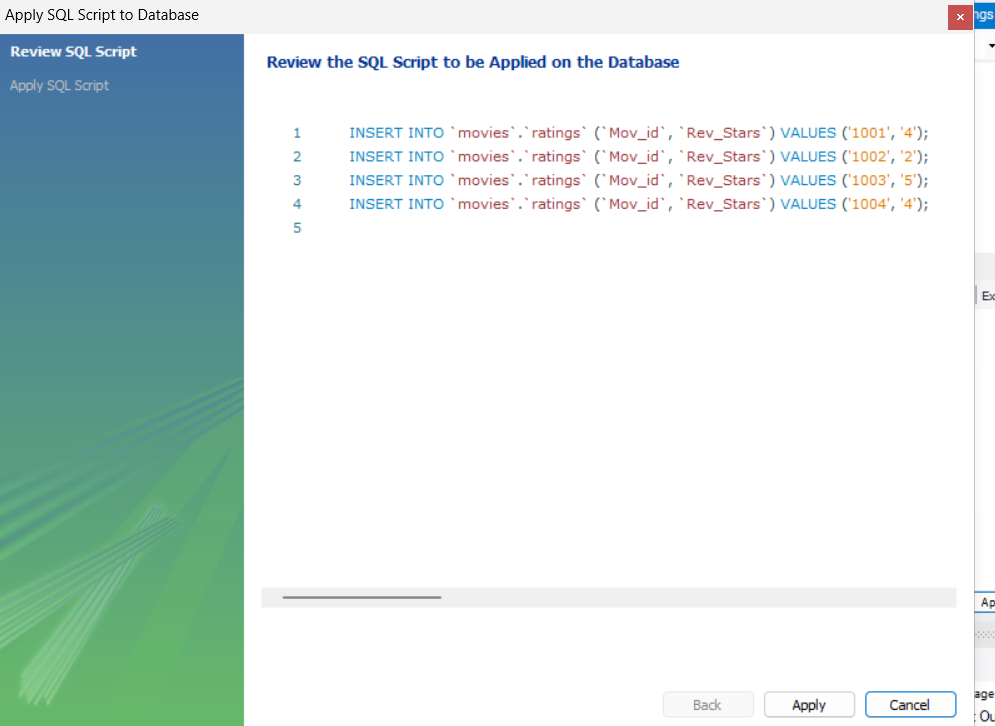
Executing the command



Inserting data into “Ratings”



Executing the command



1. List the titles of all movies directed by ‘Hitchcock’.

SELECT Mov\_Title

FROM movie

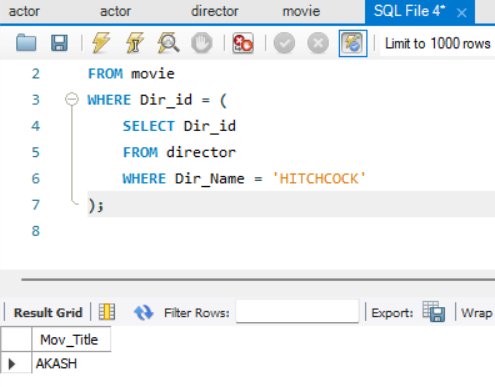
WHERE Dir\_id = (

SELECT Dir\_id

FROM director

WHERE Dir\_Name = 'HITCHCOCK'

);



1. Find the movie names where one or more actors acted in two or more movies.

SELECT DISTINCT Mov\_Title

FROM movie

WHERE Movie\_id IN (

SELECT Mov\_id

FROM movie\_cast

WHERE Act\_id IN (

SELECT Act\_id

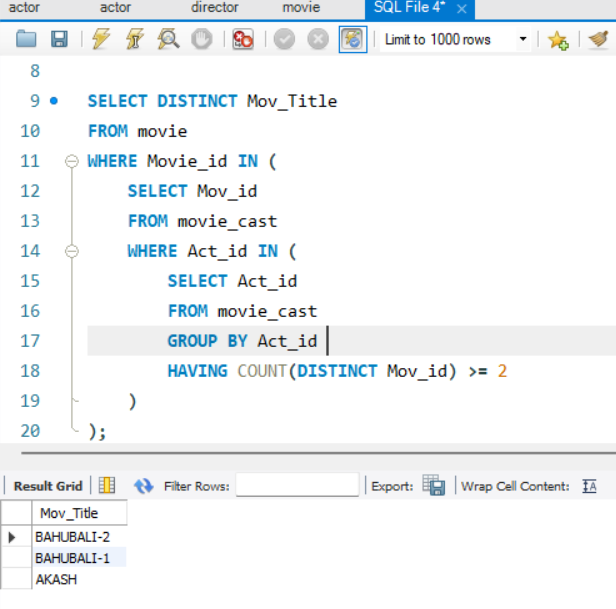
FROM movie\_cast

GROUP BY Act\_id

HAVING COUNT(DISTINCT Mov\_id) >= 2

)

);



1. List all actors who acted in a movie before 2000 and also in a movie after 2015 (use JOIN operation).

SELECT DISTINCT a.Act\_Name

FROM actor a

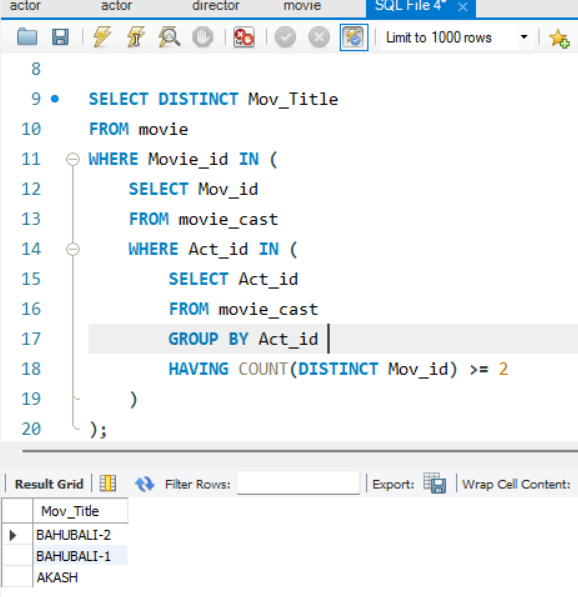
JOIN movie\_cast mc1 ON a.Act\_id = mc1.Act\_id

JOIN movie m1 ON mc1.Mov\_id = m1.Movie\_id

JOIN movie\_cast mc2 ON a.Act\_id = mc2.Act\_id

JOIN movie m2 ON mc2.Mov\_id = m2.Movie\_id

WHERE m1.Mov\_Year < 2000 AND m2.Mov\_Year > 2015;



1. Find the title of movies and number of stars for each movie that has at least one rating and find the highest number of stars that movie received. Sort the result by movie title.

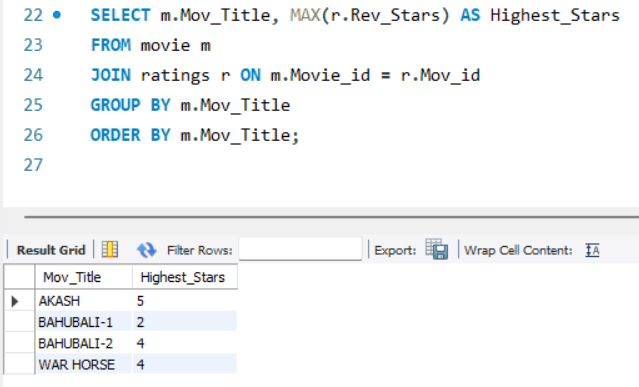
SELECT m.Mov\_Title, MAX(r.Rev\_Stars) AS Highest\_Stars

FROM movie m

JOIN ratings r ON m.Movie\_id = r.Mov\_id

GROUP BY m.Mov\_Title

ORDER BY m.Mov\_Title;



1. Update rating of all movies directed by ‘Steven Spielberg’ to 5.

UPDATE ratings

SET Rev\_Stars = 5

WHERE Mov\_id IN (

SELECT Movie\_id

FROM movie

WHERE Dir\_id = (

SELECT Dir\_id

FROM director

WHERE Dir\_Name = 'STEVEN SPIELBERG'

)

);

select \*from ratings;

