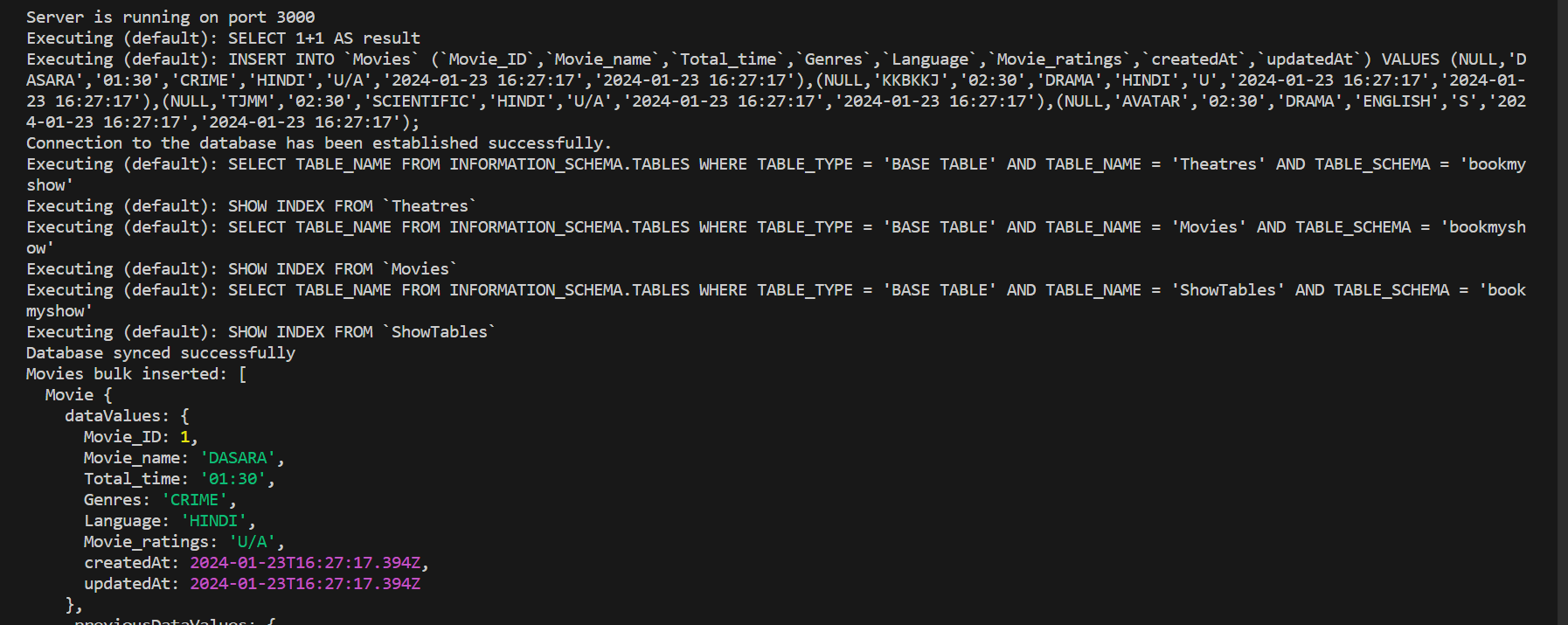
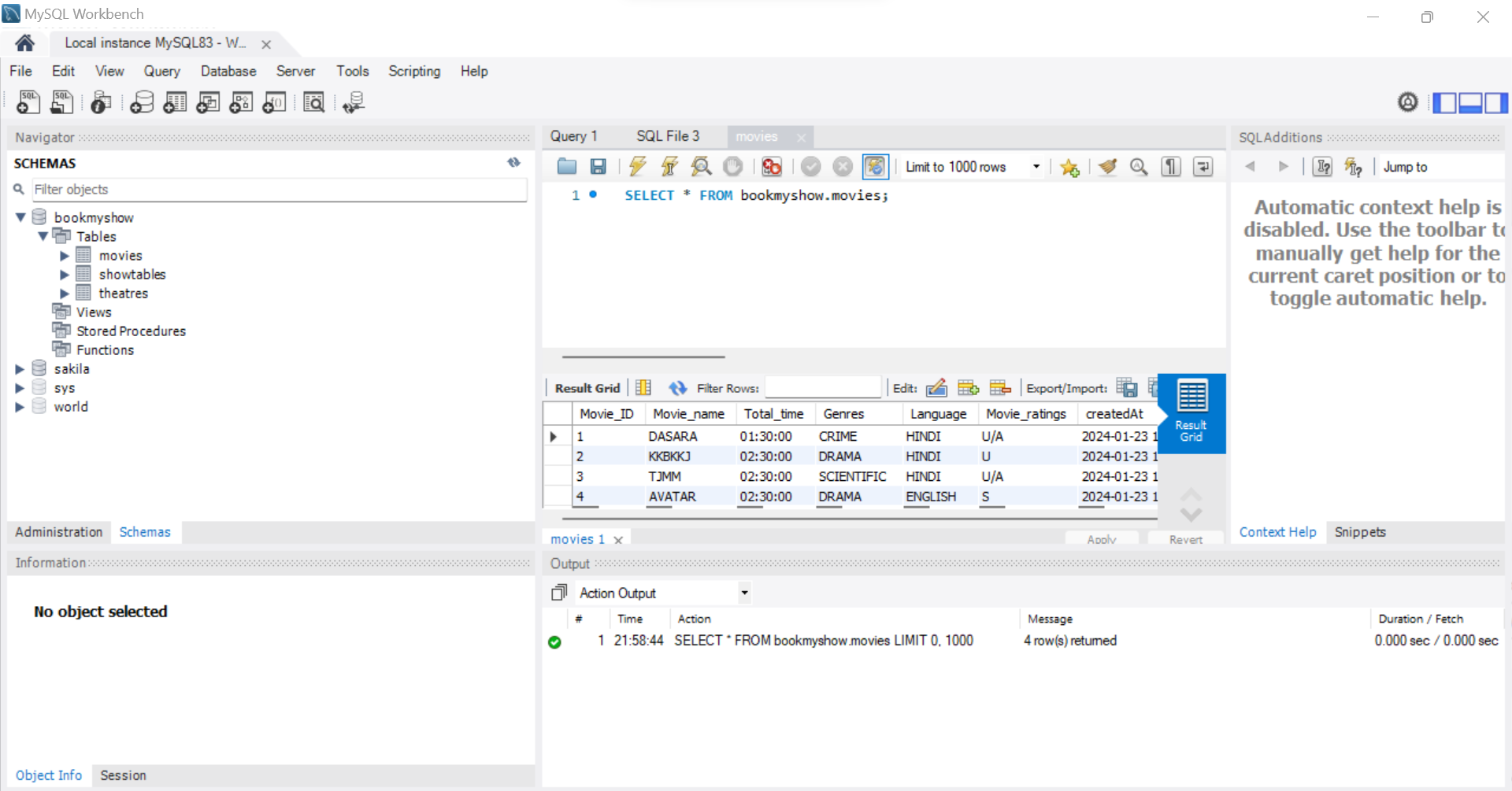
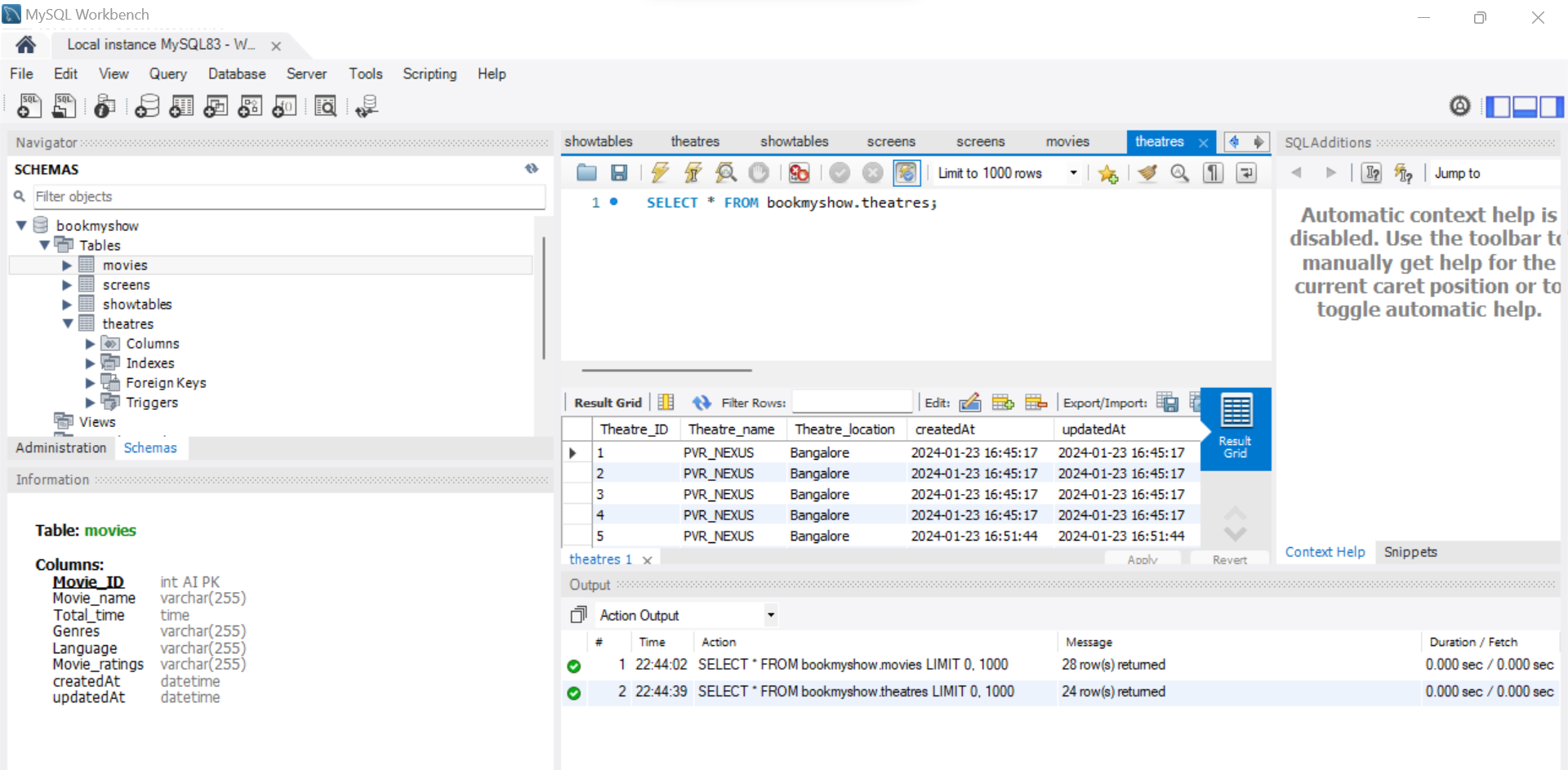
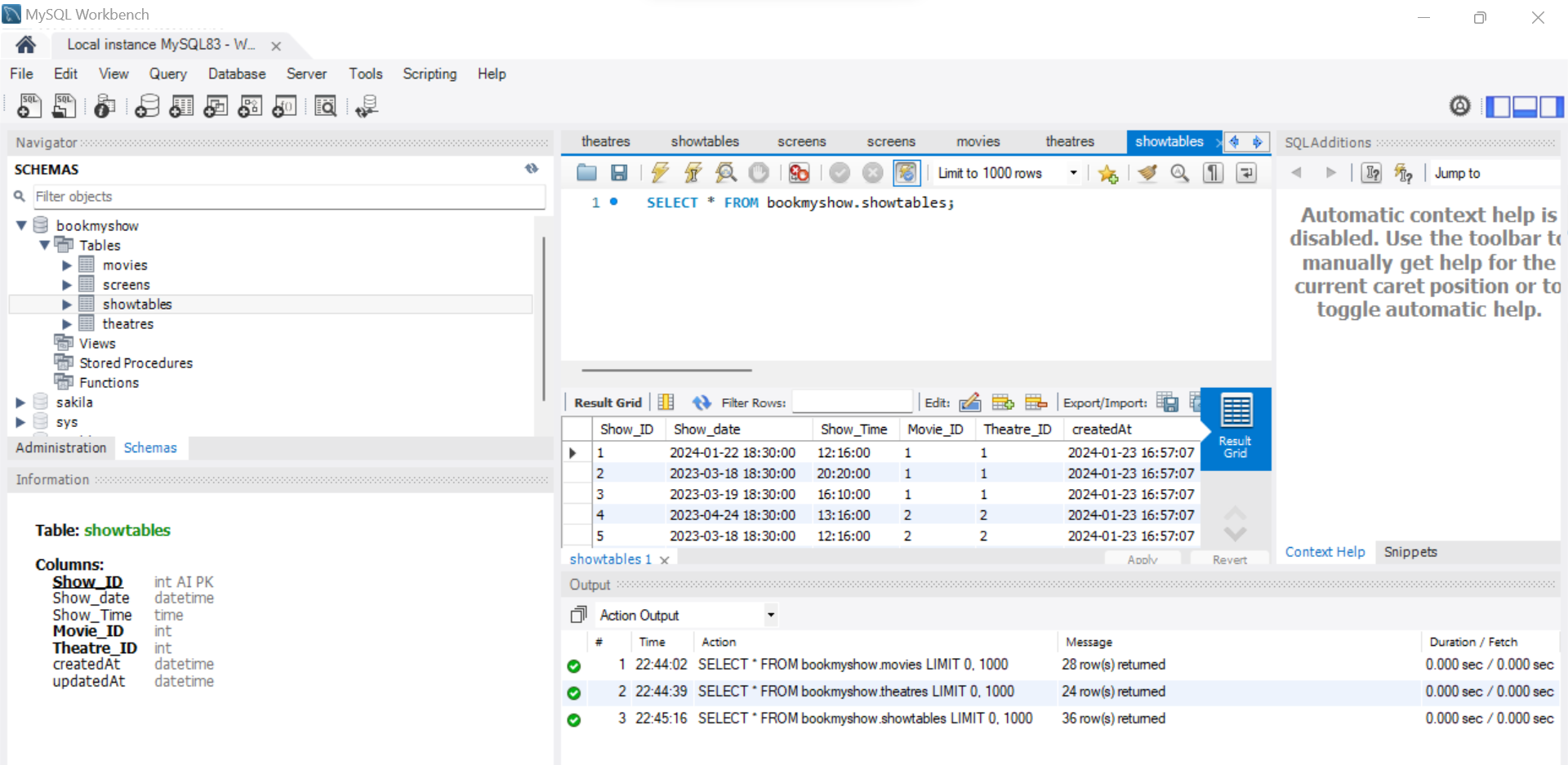
Execution:

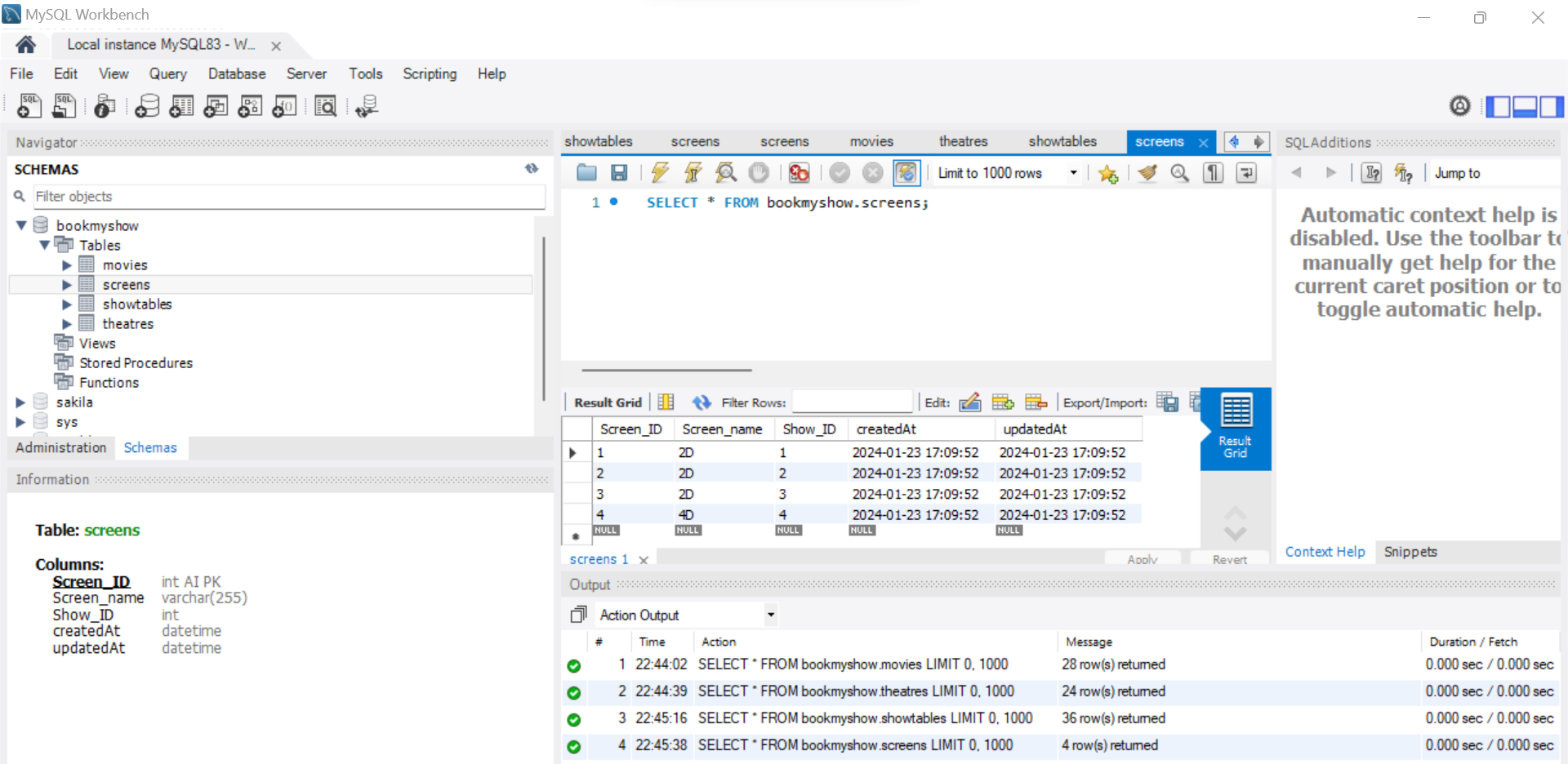
Queries are executed and table is created

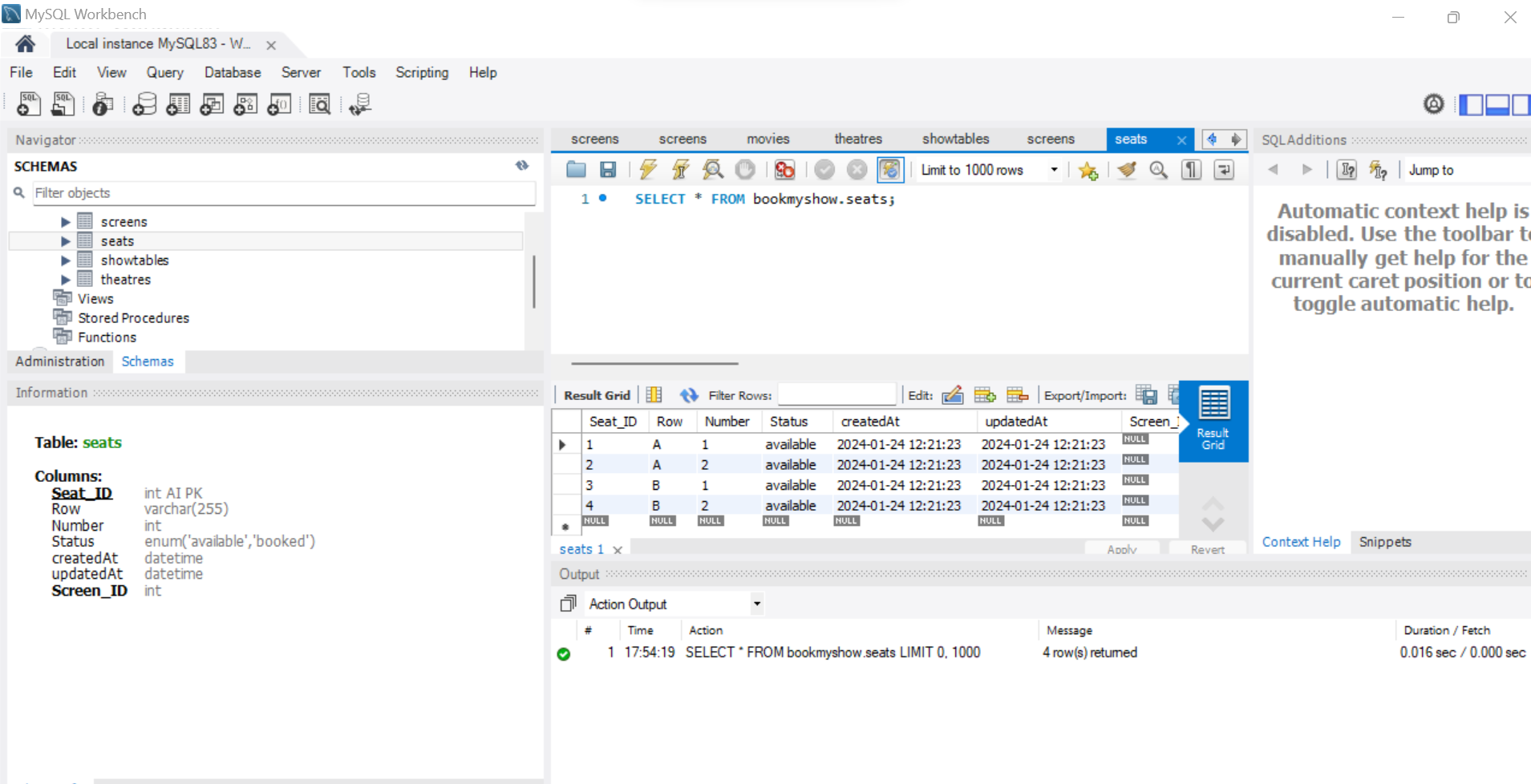


Viewing movies Table in Database:

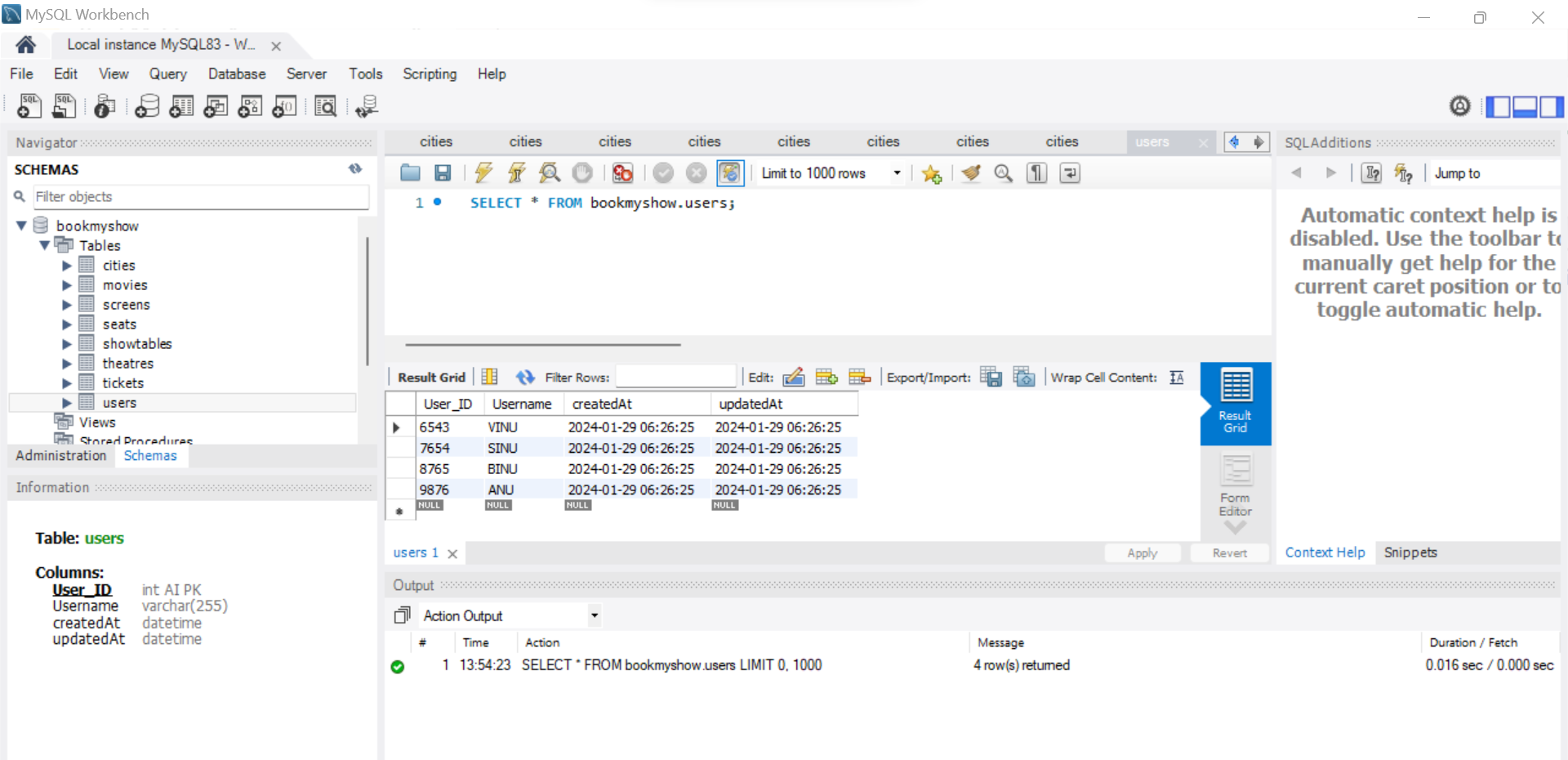


Theatre Table:  
  
  
ShowTable:  
  
  
Screen:

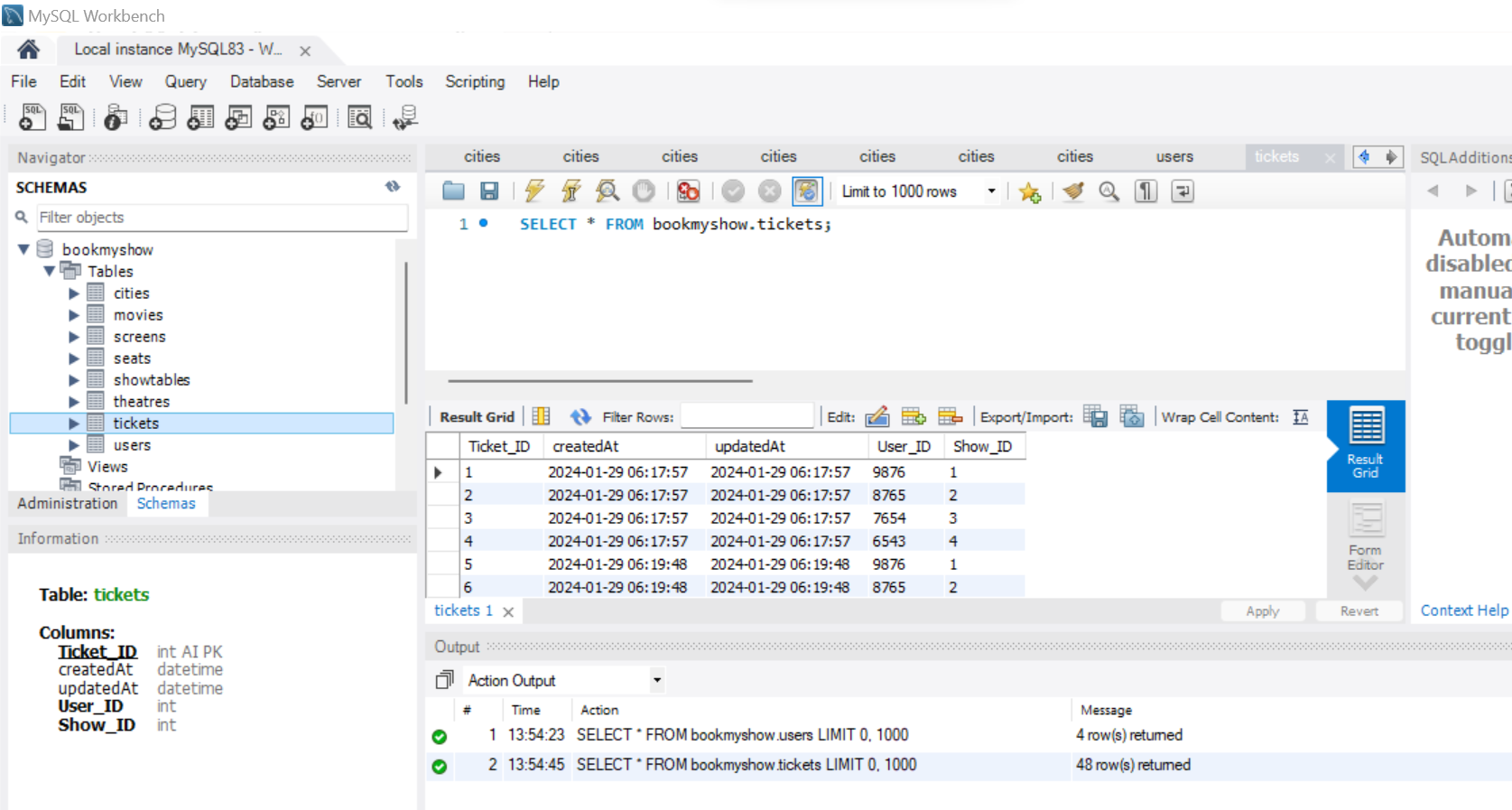


Seat:  
  


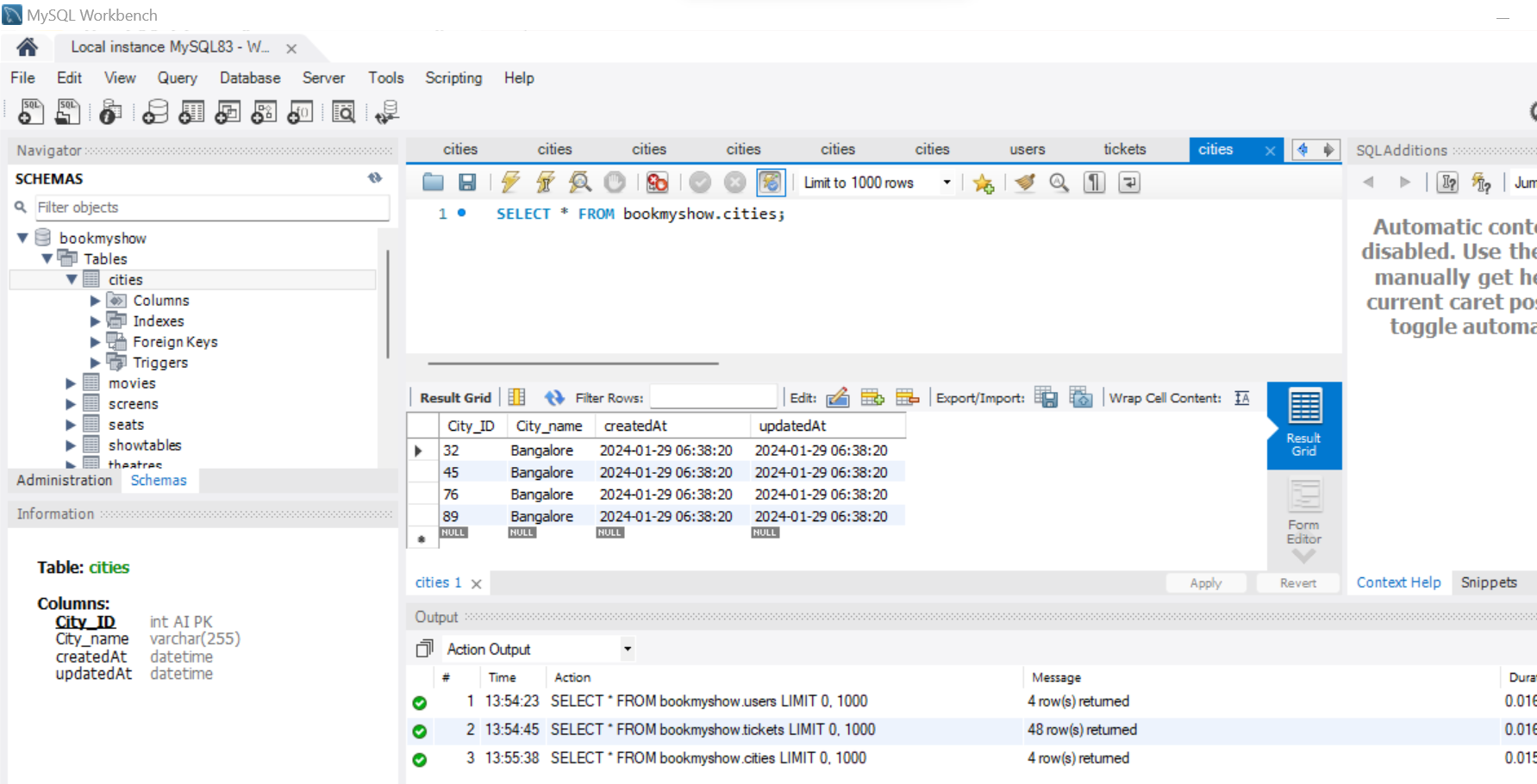
User:



Ticket:



City:



2] Associations between the entities: (1:1, 1:M, M:M)

Theatre, Movie, Screen, Show, Ticket, City, User

City : Theatre – 1:M

Theatre : Show(3Hr) – 1:M / Screen : Shows – 1: M

Theatre : Screen – 1: M

User : Ticket -1:M

Show : Movie – 1:1

Seat(normal, premium, deluxe): Screen (1:M) / Show: Seat (1:M)

……………………………………………………………………………..

**Associations in ORM – takes care of foreign key constraints**

1: 1

HasOne/ BelongsTo

Show.HasOne(Movie);

Movie.BelongsTo(Show);

//////////////////////////

1:M

HasMany & BelongsTo

City.HasMany(Theatres)

Theatre.BelongsTo(City)

//////////////////

M:M

HasMany & BelongsToMany

……………………………………..

**Part 2: Indexing for Performance**

If the API is slow, indexing can significantly improve performance. For the given scenario,

Theatre Name in Theatre table: To quickly locate theatres by name.

Datetime in Showtime table: To efficiently filter showtimes by date

To improve the performance of the API and optimize its speed, we can consider indexing columns that are frequently used in WHERE clauses of your queries. Indexing helps the database engine quickly locate and retrieve the rows that match the search conditions.

**For ShowTable Table:**

Index on Theatre\_ID and Show\_date: This helps speed up queries filtering by theatre and date.

SELECT DISTINCT DATE\_FORMAT(Show\_date, '%Y-%m-%d') AS Show\_date

FROM ShowTable

WHERE Theatre\_ID = :theatreID

AND Show\_date >= CURDATE()

ORDER BY Show\_date ASC

LIMIT 7;

--- CREATE INDEX idx\_theatre\_show\_date ON ShowTable (Theatre\_ID, Show\_date);

--- SHOW INDEX from ShowTable;

--- EXPLAIN SELECT \* FROM ShowTable WHERE Theatre\_ID = :theatreID

AND Show\_date >= CURDATE()

**For Movie Table:**

Index on Movie\_ID: This helps when joining Movie with ShowTable based on Movie\_ID.

SELECT

Movie\_name,

Total\_time,

Genres,

Language,

Movie\_ratings

FROM Movie

INNER JOIN ShowTable ON Movie.Movie\_ID = ShowTable.Movie\_ID

WHERE ShowTable.Theatre\_ID = :theatreID

AND ShowTable.Show\_date = :date;

---- CREATE INDEX idx\_movie\_id ON Movie (Movie\_ID);

----SHOW INDEX from Movie;

----EXPLAIN SELECT \* FROM Movie WHERE ShowTable.Theatre\_ID = :theatreID

AND ShowTable.Show\_date = :date;

**For Theatre Table:**

Index on Theatre\_ID: This helps when joining Theatre with ShowTable based on Theatre\_ID.

SELECT

ShowTable.Show\_date,

ShowTable.Show\_Time,

Movie.Movie\_name,

Theatre.Theatre\_name

FROM ShowTable

INNER JOIN Movie ON ShowTable.Movie\_ID = Movie.Movie\_ID

INNER JOIN Theatre ON ShowTable.Theatre\_ID = Theatre.Theatre\_ID

WHERE ShowTable.Show\_date = :date

AND ShowTable.Theatre\_ID = :theatreID;

--- CREATE INDEX idx\_theatre\_id ON Theatre (Theatre\_ID);

--- SHOW INDEX from Theatre;

--- EXPLAIN SELECT \* FROM ShowTable WHERE ShowTable.Show\_date = :date

AND ShowTable.Theatre\_ID = :theatreID;