Assignment P1:

In order to create the necessary tables and relationships for the BookMyShow scenario while ensuring they follow the 1NF, 2NF, 3NF, and BCNF rules, we can define the following entities, their attributes, and corresponding SQL table structures:

1. Theatre Entity:
- Attributes:
  - Theatre ID (Primary Key)
  - Name
  - Location
- Table Structure (1NF):

Query:
```
  CREATE TABLE Theatre (
    theatre_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(255) NOT NULL,
    location VARCHAR(255) NOT NULL
  );
```

2. **Movie Entity:**
- Attributes:
  - Movie ID (Primary Key)
  - Title
  - Duration (in minutes)
- Table Structure (1NF):

Query:
```
  CREATE TABLE Movie (
    movie_id INT PRIMARY KEY AUTO_INCREMENT,
    title VARCHAR(255) NOT NULL,
    duration INT NOT NULL
  );
```

3. **Showtime Entity:**
- Attributes:
  - Showtime ID (Primary Key)
  - Theatre ID (Foreign Key to Theatre)
  - Movie ID (Foreign Key to Movie)
  - Show Date
  - Show Time
- Table Structure (1NF):

Query:
```
  CREATE TABLE Showtime (
    showtime_id INT PRIMARY KEY AUTO_INCREMENT,
    theatre_id INT NOT NULL,
    movie_id INT NOT NULL,
    show_date DATE NOT NULL,
    show_time TIME NOT NULL,
    FOREIGN KEY (theatre_id) REFERENCES Theatre(theatre_id),
    FOREIGN KEY (movie_id) REFERENCES Movie(movie_id)
  );
```

4. Seat Entity:
- Attributes:
  - Seat ID (Primary Key)
  - Showtime ID (Foreign Key to Showtime)
  - Seat Number
  - Is Booked (Boolean)
- Table Structure (1NF):

Query:
```
CREATE TABLE Seat (
  seat_id INT PRIMARY KEY AUTO_INCREMENT,
  showtime_id INT NOT NULL,
  seat_number INT NOT NULL,
  is_booked BOOLEAN NOT NULL,
  FOREIGN KEY (showtime_id) REFERENCES Showtime(showtime_id)
);
```

5. Movie Detail Entity (For storing static movie details):
- Attributes:
  - Movie Detail ID (Primary Key)
  - Movie ID (Foreign Key to Movie)
  - Cast
  - Crew
  - Plot
  - Runtime
  - Language
  - Genre
- Table Structure (1NF):

Query:
```
CREATE TABLE MovieDetail (
  movie_detail_id INT PRIMARY KEY AUTO_INCREMENT,
  movie_id INT NOT NULL,
  cast VARCHAR(255) NOT NULL,
  crew VARCHAR(255) NOT NULL,
  plot TEXT NOT NULL,
  runtime INT NOT NULL,
  language VARCHAR(50) NOT NULL,
  genre VARCHAR(50) NOT NULL,
  FOREIGN KEY (movie_id) REFERENCES Movie(movie_id)
);
```

6. Comment Entity:
- Attributes:

  - Comment ID (Primary Key)
  - Movie ID (Foreign Key to Movie)
  - Text
- Table Structure (1NF):

Query:

```
CREATE TABLE Comment (
  comment_id INT PRIMARY KEY AUTO_INCREMENT,
  movie_id INT NOT NULL,
  text TEXT NOT NULL,
  FOREIGN KEY (movie_id) REFERENCES Movie(movie_id)
);
```

7. Rating Entity:

- Attributes:
  - Rating ID (Primary Key)
  - Movie ID (Foreign Key to Movie)
  - Rating (e.g., out of 5)
- Table Structure (1NF):

Query:

```
CREATE TABLE Rating (
  rating_id INT PRIMARY KEY AUTO_INCREMENT,
  movie_id INT NOT NULL,
  rating DECIMAL(3, 2) NOT NULL,
  FOREIGN KEY (movie_id) REFERENCES Movie(movie_id)
);
```

These SQL table structures ensure that data is organized into normalized tables that follow the 1NF, 2NF, 3NF, and BCNF rules. You can create these tables in your MySQL database using the provided SQL queries. Here are some sample entries for the Theatre, Movie, and Showtime tables:

Sample insert query for the Theatre table

```
INSERT INTO Theatre (name, location) VALUES
('Theatre 1', 'Location A'),
('Theatre 2', 'Location B');
```

Sample insert query for Movie table

```
INSERT INTO Movie (title, duration) VALUES
('Movie 1', 120),
('Movie 2', 150);
```

Sample insert query for Showtime table

```
INSERT INTO Showtime (theatre_id, movie_id, show_date, show_time) VALUES
(1, 1, '2023-08-10', '18:00:00'),
(2, 2, '2023-08-11', '19:30:00');
```

Assignment P2:

To list down all the shows on a given date at a given theatre along with their respective show timings, you can use a SQL query that joins the `Showtime` and `Theatre` tables while filtering by the desired date and theatre location. Here's a query that accomplishes this:

```
SELECT
  Showtime.show_date,
  Showtime.show_time,
  Movie.title AS movie_title
FROM
  Showtime
JOIN
  Theatre ON Showtime.theatre_id = Theatre.theatre_id
JOIN
  Movie ON Showtime.movie_id = Movie.movie_id
WHERE
  Theatre.location = 'Your Theatre Location' -- Replace with the desired theatre location
AND
  Showtime.show_date = '2023-08-10'; -- Replace with the desired date
```

In this query:
- We select the `show_date`, `show_time`, and `title` (movie title) columns.
- We perform inner joins on the `Showtime`, `Theatre`, and `Movie` tables to link the relevant data.
- We use `WHERE` clauses to filter by the desired theatre location and date. Replace `'Your Theatre

Location'` with the actual location and `'2023-08-10'` with the desired date.
This query will return a list of all shows at the specified theatre on the given date, along with their show timings and corresponding movie titles.