ER Diagram:
https://lucid.app/lucidchart/5a05f911-7f42-4a08-9a96-1088a5210a43/edit?viewport_loc=-2064%2C1302%2C7349%2C3112%2C0_0&invitationId=inv_d0e2117c-eb43-4477-b37e-85fe599911a7

Relational Schema:
https://lucid.app/lucidchart/05f37ece-355a-447d-92f5-35781e4f9d36/edit?viewport_loc=-1445%2C951%2C4904%2C2077%2C0_0&invitationId=inv_98081965-9b23-464e-9de4-0d538929188f

In-Depth discussion of queries:

Query 1:

```
-- available drivers : Quer1
SELECT driver.DriverID, driver.Name , driver.Rating , driver.VehicleID, vehicles.Capacity, vehicles.Status, vehicles.Model
FROM driver
JOIN vehicles ON driver.VehicleID = vehicles.VehicleID
WHERE vehicles.Status =  "Available";
```

Relational Operation: Join
Constraint: Status

Query 2:

```
-- Driver with ratings > 3 : Query 2
SELECT driver.DriverID, driver.Name , driver.Rating , driver.VehicleID, driver.Rating, driver.Email
FROM Driver
WHERE Rating >3;
```

Relational Operation: Projection
Constraint: Rating

Query 3:

```
-- Average earnings for a driver : Query 3
SELECT AVG(driver.TotalEarning)
FROM driver;
```

Relational Operation: AVG
Constraint: -

Query 4:

```sql
-- Query 4: No of Rental Stands in a particular city/place
SELECT COUNT(DetailsofStand)
FROM rentalservice
where DetailsofStand = 'Chandigarh';
```

Relational Operation: COUNT
Constraint: City Name

Query 5:

```sql
-- Query 5: Employees of salary more than 1L
SELECT * FROM employees
WHERE Salary BETWEEN 100000 AND 150000;
```

Relational Operation: Projection
Constraint: Range

Query 6:

```sql
-- Query 6: Top 10 drivers wrt earning
SELECT driver.DriverID, driver.Name , driver.Rating , driver.VehicleID, vehicles.Capacity, vehicles.Status, vehicles.Model, driver.TotalEarning
FROM driver
JOIN vehicles ON driver.VehicleID = vehicles.VehicleID
ORDER BY TotalEarning DESC
LIMIT 10;
```

Relational Operation: Join
Constraint: ORDER, Desc, Limit
Query 7:

```sql
-- Query 7: Trips on a particular date.
SELECT * FROM trip WHERE Date='2022-11-11';
```

Relational Operation: Projection
Constraint: Date

Query 8:

```sql
-- Query 8: Trips with  particular droplocation
create Table IF NOT EXISTS Operational_city(
    name char(50) PRIMARY KEY);
insert into Operational_city(name) value('Kurukshetra');
select * from trip where DropLocation in (
    select name from Operational_city
);
```

Relational Operation: Intersection
Constraint: City name

Query 9:

```sql
-- Query 9: Customers who havennot done any ride
SELECT *
FROM customer
WHERE CustomerID  NOT IN (
    SELECT DISTINCT CustomerID
    FROM trip
);
```

Relational Operation: Diffrence
Constraint: -

Query 10:

```sql
-- Query 10: Trips of 2 particular customer
select * from trip
where CustomerID = 90 or CustomerID = 52;
```

Relational Operation: Projection
Constraint: ID

Query 11:

```
-- Query 11 wildcards
SELECT *
FROM trip
WHERE PickupLocation LIKE '%am%';
```

Relational Operation: Projection
Constraint: Wildcard

Query 12:

```
-- Query 12 update
SET SQL_SAFE_UPDATES = 0;
UPDATE vehicles
SET capacity=1
WHERE Model = 'Bike';
select * from vehicles;
```

Relational Operation: Update
Constraint: Model, Setting update

Query 13:

```
-- Query 13 Union (shows data of non-employee)
SELECT Name, PhoneNumber
FROM driver
UNION
SELECT Name, PhoneNumber
FROM customer;
```

Relational Operation: Union
Constraint:-

Query 14:

```sql
-- Query 14 Trigger for adding new vehicle
create trigger newveh
before INSERT
on
vehicles
for each row
Update rentalservice
SET rentalservice.NOCabAvailable = rentalservice.NOCabAvailable + 1
where ServiceID = 100;

insert into vehicles (Status, RegistrationNo, Model, LicenseNO, Colour, Capacity) values ('Available', 'GUdd8bsZsL', 'SUV', '5N1AR2MMXEC35354', 'Khaki', 3);
select * from rentalservice;
```

Relational Operation: Update, trigger, progression
Constraint: trigger, serviceid