Bay Car Rental Management System

Summary:

Bay Car Rentals Management System

1. Introduction: about the business

2.Business Rules (scope)

3.Operation Procedures

4.ER Diagram(Entity / ER Model/ Table/ERD)

5.Table Data Representation

6.Stored Procedures/Functions/Triggers/Views

7.Conclusion

1.Introduction

- net-based/local based vehicle rental services in Bay Area.
- **5 branches**: Fremont, Union City, Sunnyvale, Milpitas, Foster City, and SF
- **Employee**: Managers and Rental Specialists
- **Vehicle Category** :

i) **small:** 2-4 people (Honda Fit, Toyota Prius, Mini Cooper)

ii) **Medium:** 4-5 people(Ford Escape, Honda Accord, Hyundai Sonata)

iii) **Van:** 7-8 people(Honda Odyssey/Chevrolet Suburban, Ford Expedition)

Continued...

- ❖ **3 basic Functions**:

i) Rent : Customer will inquiry rental car information based on location and a specific date of rent. The system will provide a list of all available cars. There are two options for insurance: Car / liability. Once customer confirms the type of car and date, the system will collect the customer information and confirm reservation.

ii) **Pick up**: When the customer walks in, the system will pull up reservation info about customer. Then the customer provide driver’s license, sign lease agreement, and confirm order.

iii) **Return** : The system will process return service when the customer returns the car. Then it will record date, process, and calculate final rental amount.

2.Business Rules

1) We allow free cancellation or modification of the order before the pick_up time. After the pick_up time however, the fees will be charged.
2) The customer will have the option to drop off the rented car in any convenient branches with no extra fees.
3) Currently, we only have full tank pick up and full tank return option for gas

4) Insurance Policies:

i) Car Insurance:

Small car : $20 per day

Family car: $25 per day

ii) Liability Insurance

Standard Supplemental Protection: $10.95 per day

Premium Supplemental Protection: $14.95 per

2.Business Rules continued...

5) We only accept credits, no cash and debit card. Acceptable credit cards: AMEX, Discover, Visa, Chase, and Capital one.

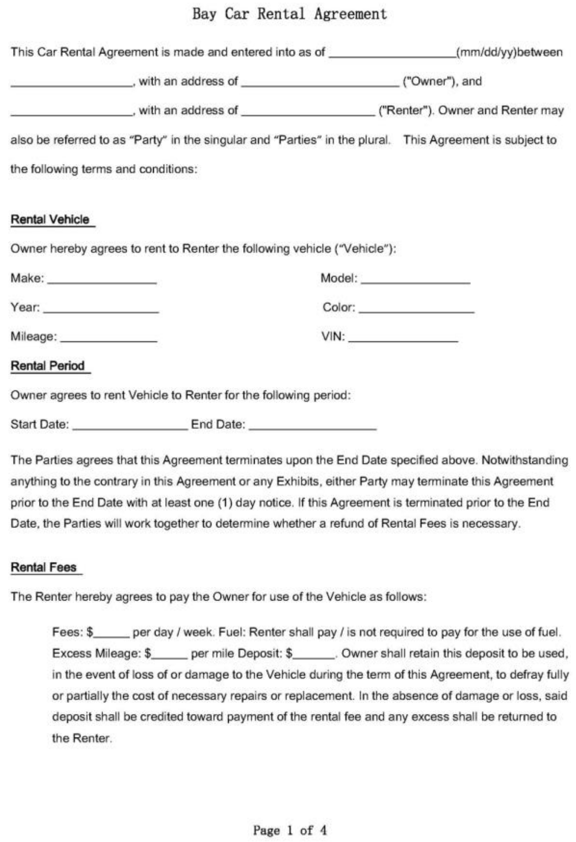
6) The customer must prepay the bill at the pick up time. The additional fees will be charged on the return day depending overdue payment, damage of car, etc

7) An acceptable, valid driver's license issued from your country of residence must be presented at the time of rental

8) we can pay the tickets fees for the customer during the rental period, but with additional 15% of the service fees

2.Business Rules----Lease agreement

The customer should read the lease agreement carefully and sign on our lease agreement.



Bay Car Rental Agreement

This Car Rental Agreement is made and entered into as of ________________(mm/dd/yy)between ________________, with an address of ____________________ ("Owner"), and ________________, with an address of __________________ ("Renter"). Owner and Renter may also be referred to as "Party" in the singular and "Parties" in the plural. This Agreement is subject to the following terms and conditions:

Rental Vehicle

Owner hereby agrees to rent to Renter the following vehicle ("Vehicle"):

Make: ______________	Model: ______________
Year: ______________	Color: ______________
Mileage: ____________	VIN: ______________

Rental Period

Owner agrees to rent Vehicle to Renter for the following period:

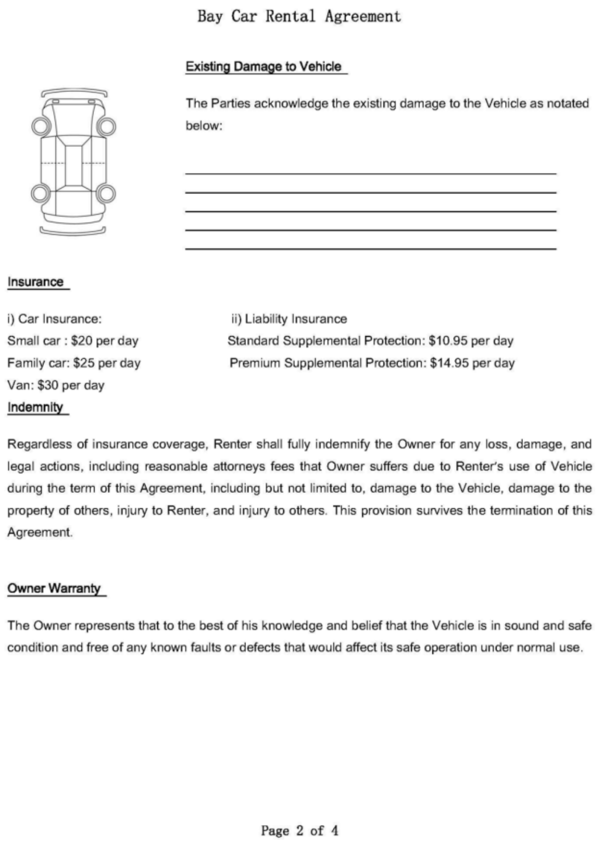
Start Date: ______________ End Date: ________________

The Parties agrees that this Agreement terminates upon the End Date specified above. Notwithstanding anything to the contrary in this Agreement or any Exhibits, either Party may terminate this Agreement prior to the End Date with at least one (1) day notice. If this Agreement is terminated prior to the End Date, the Parties will work together to determine whether a refund of Rental Fees is necessary.

Rental Fees

The Renter hereby agrees to pay the Owner for use of the Vehicle as follows:

Fees: $_____ per day / week. Fuel: Renter shall pay / is not required to pay for the use of fuel. Excess Mileage: $_____ per mile Deposit: $______. Owner shall retain this deposit to be used, in the event of loss of or damage to the Vehicle during the term of this Agreement, to defray fully or partially the cost of necessary repairs or replacement. In the absence of damage or loss, said deposit shall be credited toward payment of the rental fee and any excess shall be returned to the Renter.



Bay Car Rental Agreement

Existing Damage to Vehicle

The Parties acknowledge the existing damage to the Vehicle as notated below:

____

____

____

____

____

Insurance

i) Car Insurance:	ii) Liability Insurance
Small car : $20 per day	Standard Supplemental Protection: $10.95 per day
Family car: $25 per day	Premium Supplemental Protection: $14.95 per day
Van: $30 per day	

Indemnity

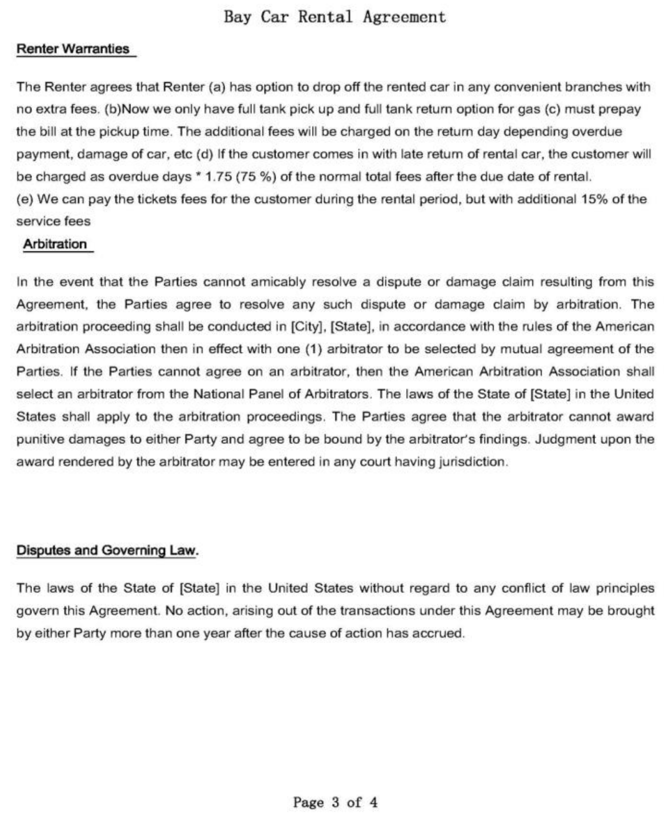
Regardless of insurance coverage, Renter shall fully indemnify the Owner for any loss, damage, and legal actions, including reasonable attorneys fees that Owner suffers due to Renter's use of Vehicle during the term of this Agreement, including but not limited to, damage to the Vehicle, damage to the property of others, injury to Renter, and injury to others. This provision survives the termination of this Agreement.

Owner Warranty

The Owner represents that to the best of his knowledge and belief that the Vehicle is in sound and safe condition and free of any known faults or defects that would affect its safe operation under normal use.

2.Business Rules----Lease agreement

The customer should read the lease agreement carefully and sign on our lease agreement.



Bay Car Rental Agreement

Renter Warranties

The Renter agrees that Renter (a) has option to drop off the rented car in any convenient branches with no extra fees. (b)Now we only have full tank pick up and full tank return option for gas (c) must prepay the bill at the pickup time. The additional fees will be charged on the return day depending overdue payment, damage of car, etc (d) If the customer comes in with late return of rental car, the customer will be charged as overdue days * 1.75 (75 %) of the normal total fees after the due date of rental.

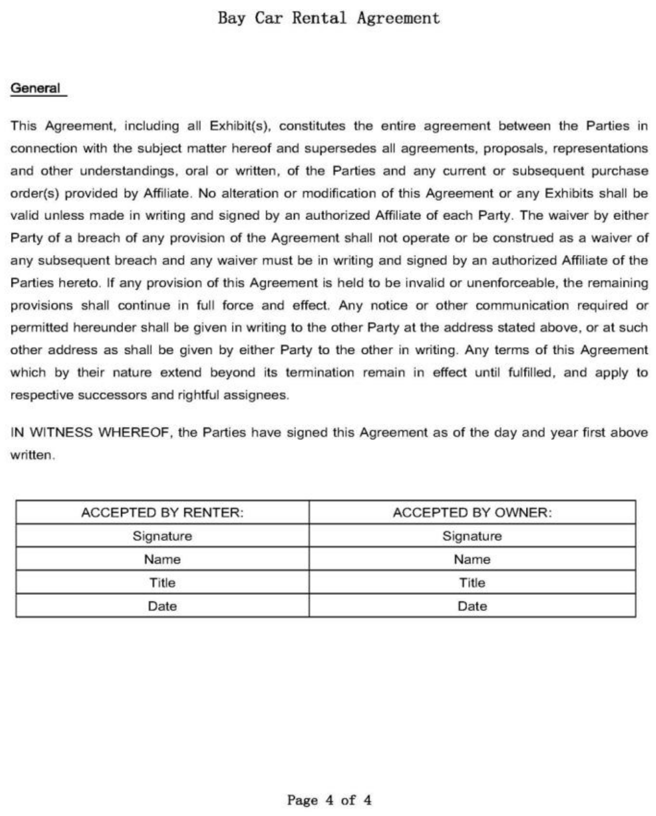
(e) We can pay the tickets fees for the customer during the rental period, but with additional 15% of the service fees

Arbitration

In the event that the Parties cannot amicably resolve a dispute or damage claim resulting from this Agreement, the Parties agree to resolve any such dispute or damage claim by arbitration. The arbitration proceeding shall be conducted in [City], [State], in accordance with the rules of the American Arbitration Association then in effect with one (1) arbitrator to be selected by mutual agreement of the Parties. If the Parties cannot agree on an arbitrator, then the American Arbitration Association shall select an arbitrator from the National Panel of Arbitrators. The laws of the State of [State] in the United States shall apply to the arbitration proceedings. The Parties agree that the arbitrator cannot award punitive damages to either Party and agree to be bound by the arbitrator's findings. Judgment upon the award rendered by the arbitrator may be entered in any court having jurisdiction.

Disputes and Governing Law.

The laws of the State of [State] in the United States without regard to any conflict of law principles govern this Agreement. No action, arising out of the transactions under this Agreement may be brought by either Party more than one year after the cause of action has accrued.



Bay Car Rental Agreement

General

This Agreement, including all Exhibit(s), constitutes the entire agreement between the Parties in connection with the subject matter hereof and supersedes all agreements, proposals, representations and other understandings, oral or written, of the Parties and any current or subsequent purchase order(s) provided by Affiliate. No alteration or modification of this Agreement or any Exhibits shall be valid unless made in writing and signed by an authorized Affiliate of each Party. The waiver by either Party of a breach of any provision of the Agreement shall not operate or be construed as a waiver of any subsequent breach and any waiver must be in writing and signed by an authorized Affiliate of the Parties hereto. If any provision of this Agreement is held to be invalid or unenforceable, the remaining provisions shall continue in full force and effect. Any notice or other communication required or permitted hereunder shall be given in writing to the other Party at the address stated above, or at such other address as shall be given by either Party to the other in writing. Any terms of this Agreement which by their nature extend beyond its termination remain in effect until fulfilled, and apply to respective successors and rightful assignees.

IN WITNESS WHEREOF, the Parties have signed this Agreement as of the day and year first above written.

ACCEPTED BY RENTER:	ACCEPTED BY OWNER:
Signature	Signature
Name	Name
Title	Title
Date	Date

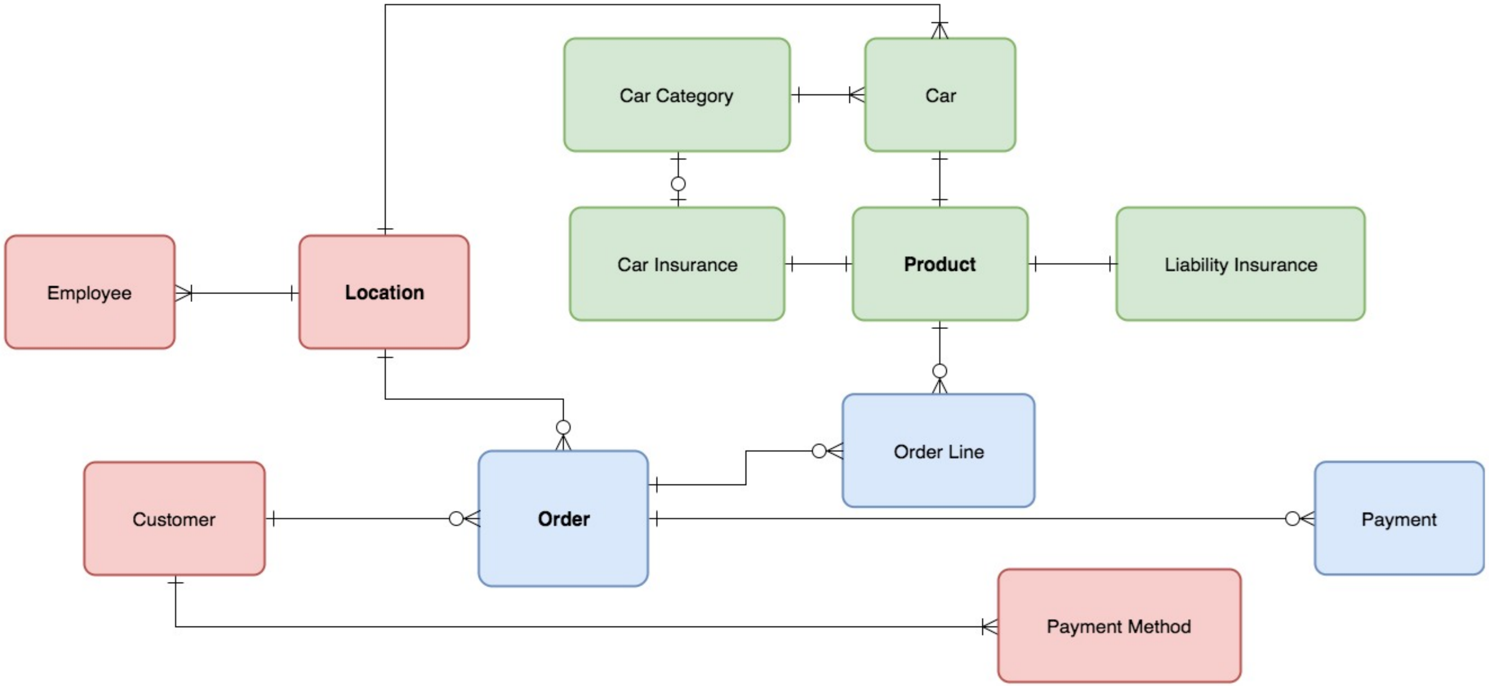
3.Rental Operation Procedures

1.Choose **start_Date**,**End_Date**, **Pick_Location**, **Drop_Location**

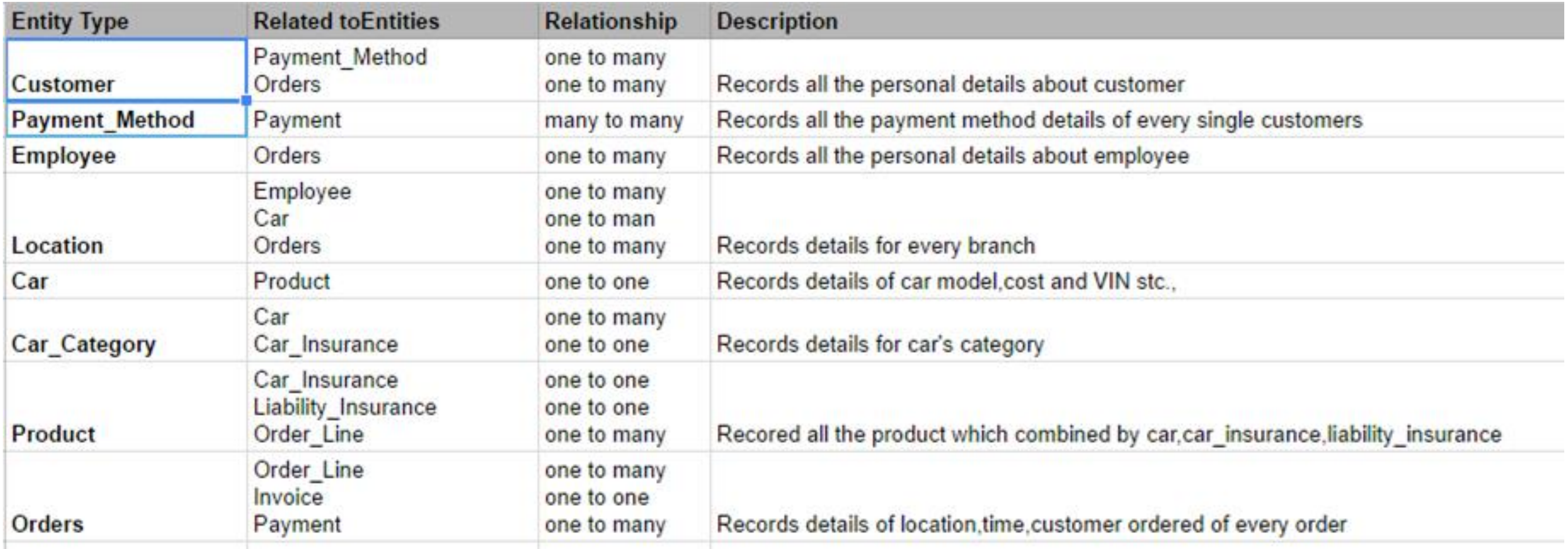
2.**choose car:** Category ---- car_ID (3 category : Small/Medium/VAN)
3.**choose car_insurance or not**: 1 category of car ----- 1 type of car_insurance (1:1)
4.**choose liability_insurance or not**: 2 type of liability for every category of cars(standard/Premium)
5.**Products** within Order: Car + Car_Insurance + Liability_Insurance

6. **Process into orderline**

4.ER Conceptual Model--ER Model

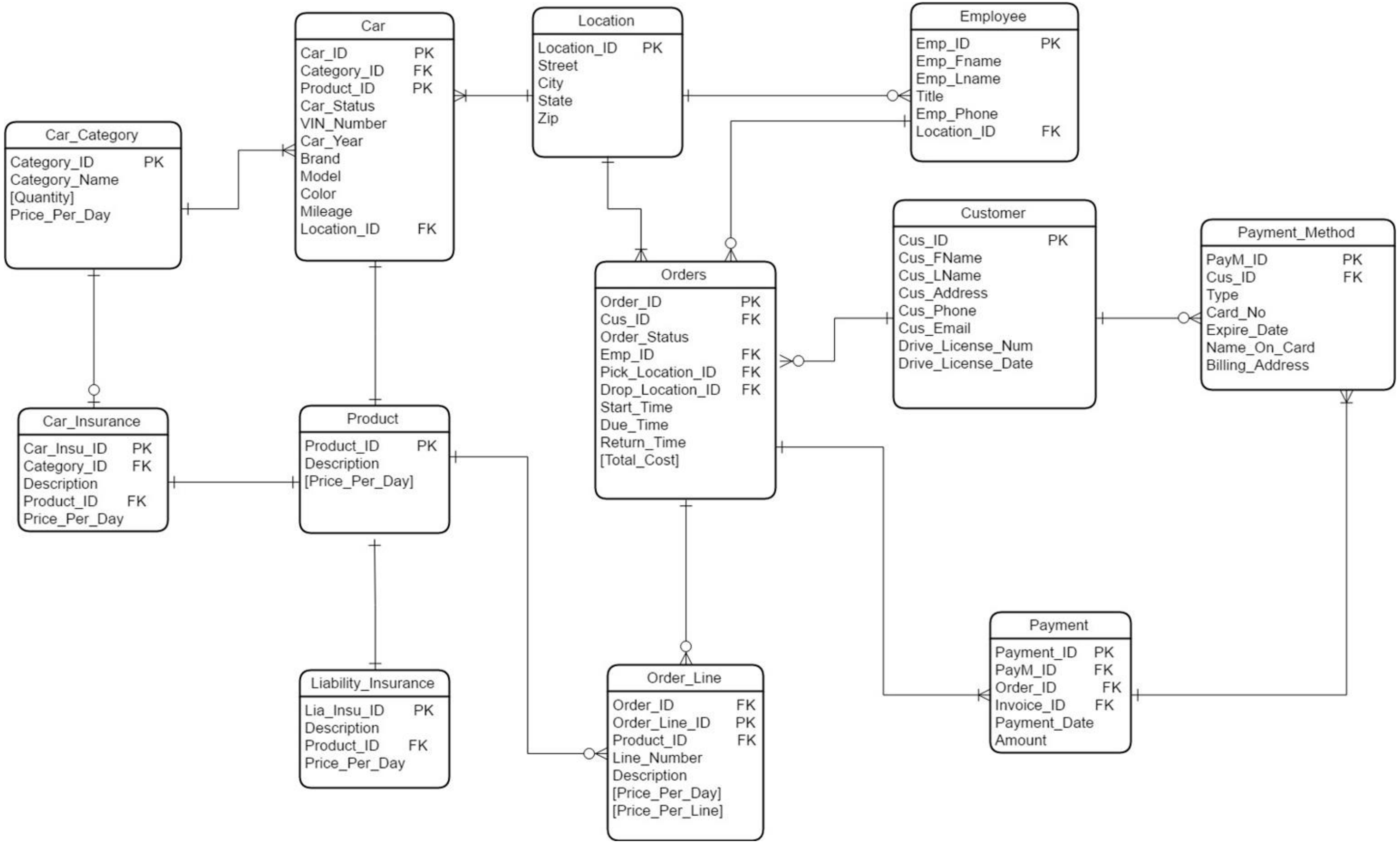


Continue ER Conceptual Model--Entity and relationship



Entity Type	Related toEntities	Relationship	Description
Customer	Payment_Method Orders	one to many one to many	Records all the personal details about customer
Payment_Method	Payment	many to many	Records all the payment method details of every single customers
Employee	Orders	one to many	Records all the personal details about employee
Location	Employee Car Orders	one to many one to man one to many	Records details for every branch
Car	Product	one to one	Records details of car model,cost and VIN stc.,
Car_Category	Car Car_Insurance	one to many one to one	Records details for car's category
Product	Car_Insurance Liability_Insurance Order_Line	one to one one to one one to many	Recored all the product which combined by car,car_insurance,liability_insurance
Orders	Order_Line Invoice Payment	one to many one to one one to many	Records details of location,time,customer ordered of every order

Continued ER Conceptual Model--ERD



Car	
Car_ID	PK
Category_ID	FK
Product_ID	PK
Car_Status	
VIN_Number	
Car_Year	
Brand	
Model	
Color	
Mileage	
Location_ID	FK

Location	
Location_ID	PK
Street	
City	
State	
Zip	

Employee	
Emp_ID	PK
Emp_Fname	
Emp_Lname	
Title	
Emp_Phone	
Location_ID	FK

Car_Category	
Category_ID	PK
Category_Name	
[Quantity]	
Price_Per_Day	

Orders	
Order_ID	PK
Cus_ID	FK
Order_Status	
Emp_ID	FK
Pick_Location_ID	FK
Drop_Location_ID	FK
Start_Time	
Due_Time	
Return_Time	
[Total_Cost]	

Customer	
Cus_ID	PK
Cus_FName	
Cus_LName	
Cus_Address	
Cus_Phone	
Cus_Email	
Drive_License_Num	
Drive_License_Date	

Payment_Method	
PayM_ID	PK
Cus_ID	FK
Type	
Card_No	
Expire_Date	
Name_On_Card	
Billing_Address	

Car_Insurance	
Car_Insu_ID	PK
Category_ID	FK
Description	
Product_ID	FK
Price_Per_Day	

Product	
Product_ID	PK
Description	
[Price_Per_Day]	

Liability_Insurance	
Lia_Insu_ID	PK
Description	
Product_ID	FK
Price_Per_Day	

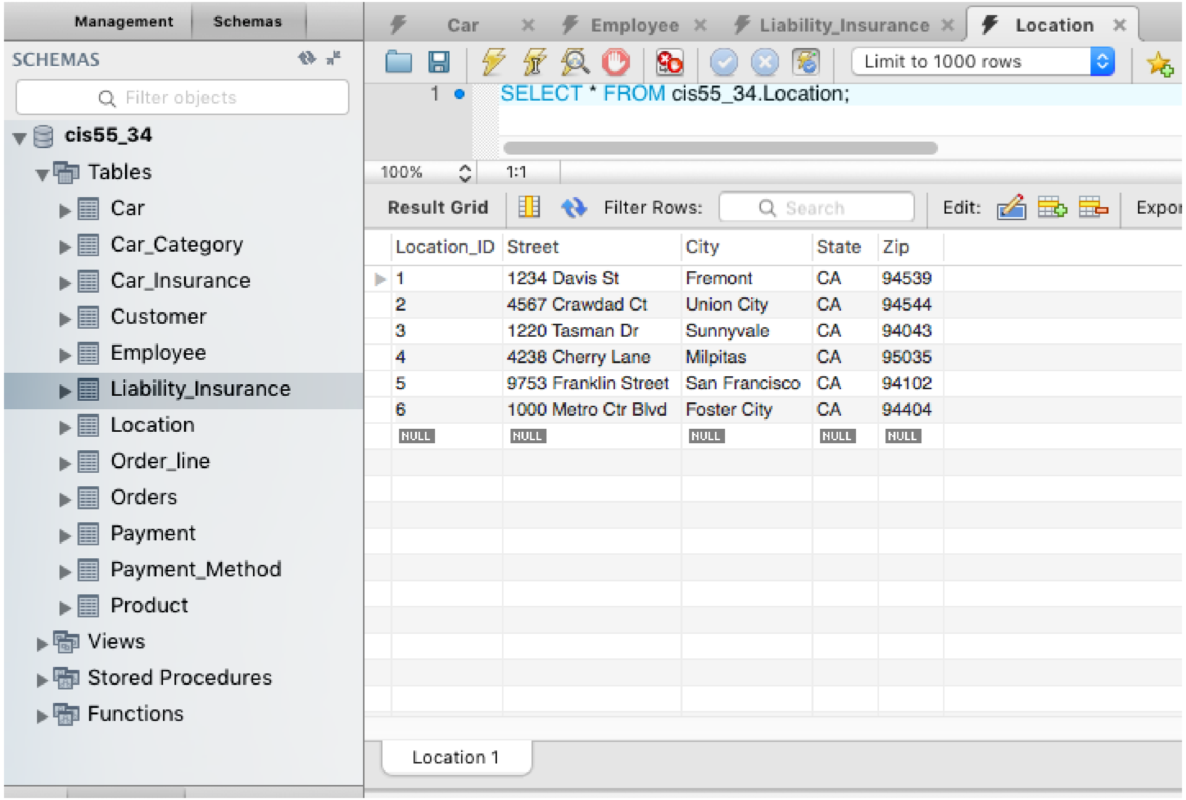
Order_Line	
Order_ID	FK
Order_Line_ID	PK
Product_ID	FK
Line_Number	
Description	
[Price_Per_Day]	
[Price_Per_Line]	

Payment	
Payment_ID	PK
PayM_ID	FK
Order_ID	FK
Invoice_ID	FK
Payment_Date	
Amount	

Continue Tables and Their Implementation

1. Location Table

Create table Location
(
Location_ID int NOT NULL,
Street varchar(50),
City varchar(25),
State varchar(2),
Zip varchar(5),
Primary key (Location_ID)
);



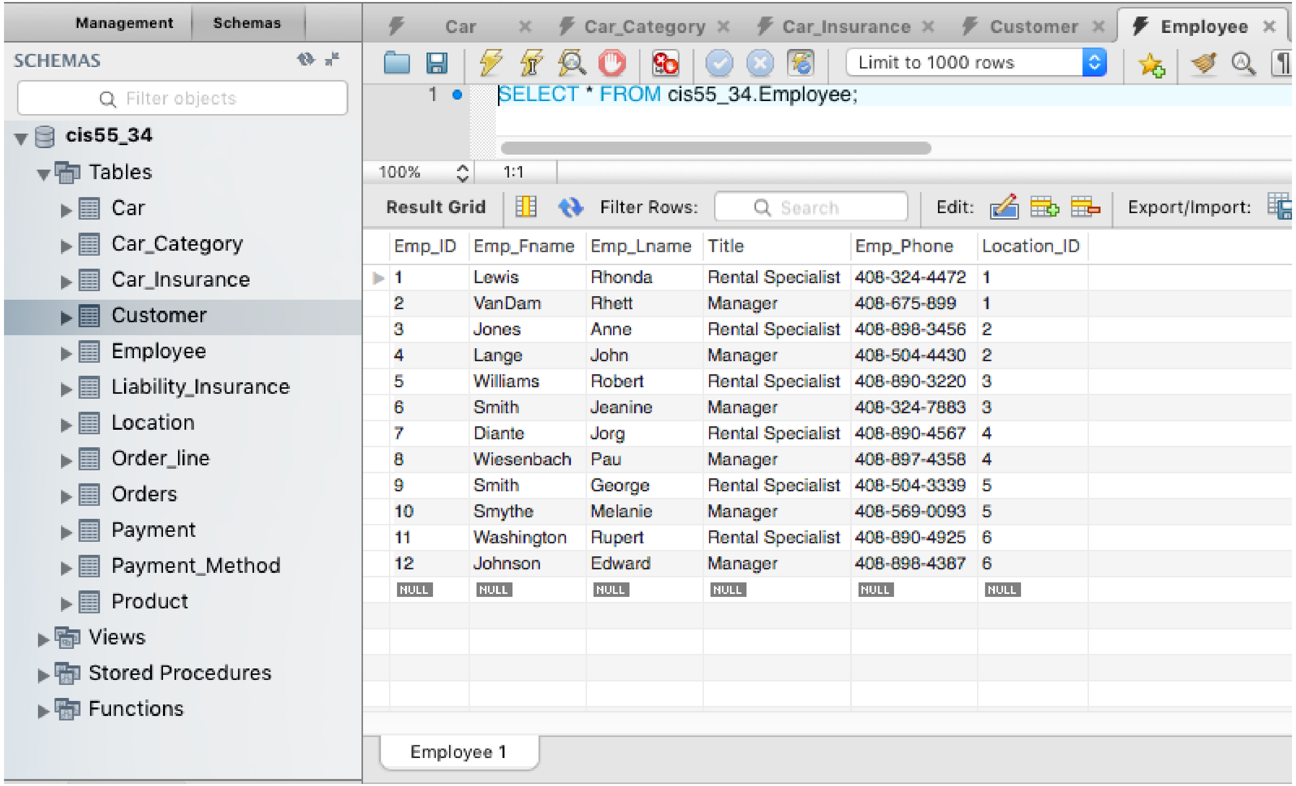
```
SELECT * FROM cis55_34.Location;
```

Location_ID	Street	City	State	Zip
1	1234 Davis St	Fremont	CA	94539
2	4567 Crawdad Ct	Union City	CA	94544
3	1220 Tasman Dr	Sunnyvale	CA	94043
4	4238 Cherry Lane	Milpitas	CA	95035
5	9753 Franklin Street	San Francisco	CA	94102
6	1000 Metro Ctr Blvd	Foster City	CA	94404
NULL	NULL	NULL	NULL	NULL

Continue Tables and Their Implementation

2. Employee Table

```
Create table Employee
(
Emp_ID int NOT NULL,
Emp_Fname varchar(25),
Emp_Lname varchar(25),
Title varchar(30),
Emp_Phone varchar(12),
Location_ID int,
Primary key (Emp_ID),
Foreign key (Location_ID)
references Location(Location_ID)
);
```



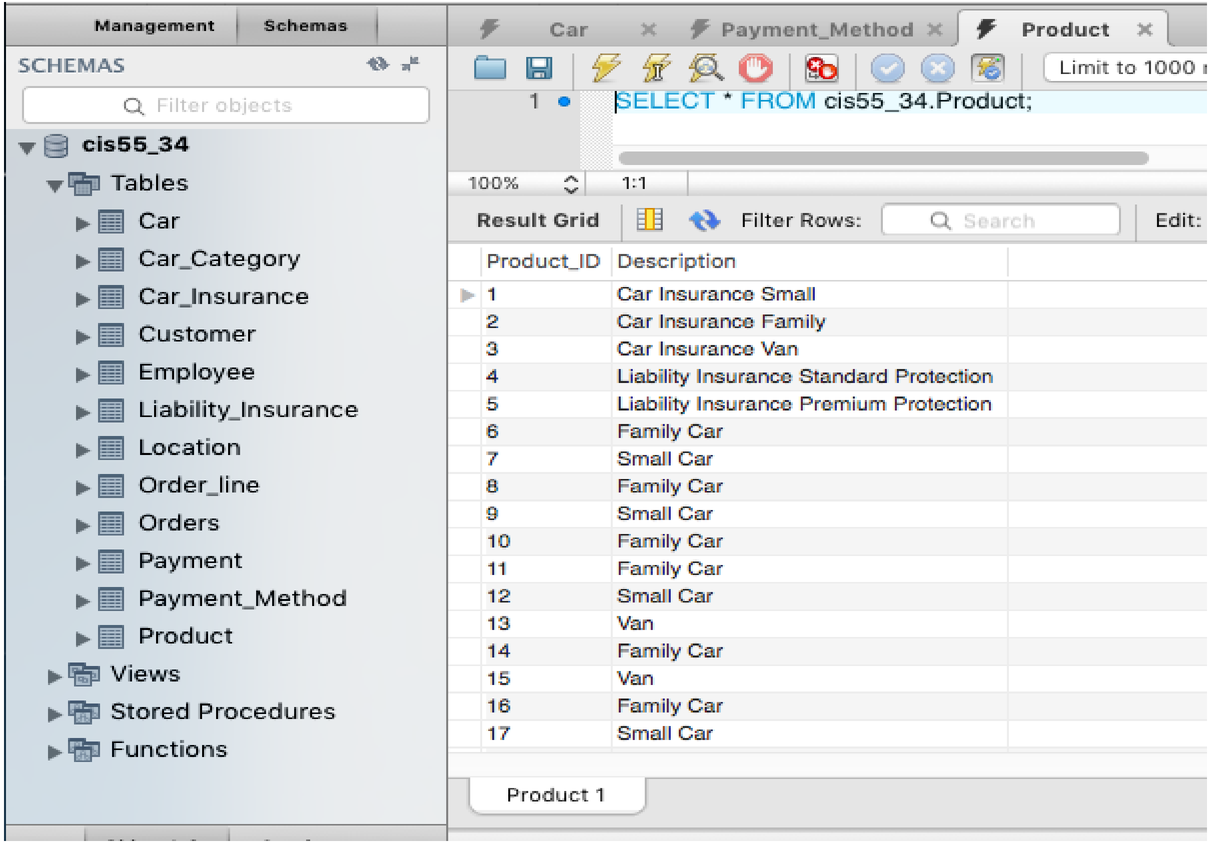
```
SELECT * FROM cis55_34.Employee;
```

Emp_ID	Emp_Fname	Emp_Lname	Title	Emp_Phone	Location_ID
1	Lewis	Rhonda	Rental Specialist	408-324-4472	1
2	VanDam	Rhett	Manager	408-675-899	1
3	Jones	Anne	Rental Specialist	408-898-3456	2
4	Lange	John	Manager	408-504-4430	2
5	Williams	Robert	Rental Specialist	408-890-3220	3
6	Smith	Jeanine	Manager	408-324-7883	3
7	Diante	Jorg	Rental Specialist	408-890-4567	4
8	Wiesenbach	Pau	Manager	408-897-4358	4
9	Smith	George	Rental Specialist	408-504-3339	5
10	Smythe	Melanie	Manager	408-569-0093	5
11	Washington	Rupert	Rental Specialist	408-890-4925	6
12	Johnson	Edward	Manager	408-898-4387	6
NULL	NULL	NULL	NULL	NULL	NULL

Continue Tables and Their Implementation

3. Product Table

Create table Product
(
Product_ID int NOT NULL,
Description varchar(250),
Primary key (Product_ID)
);



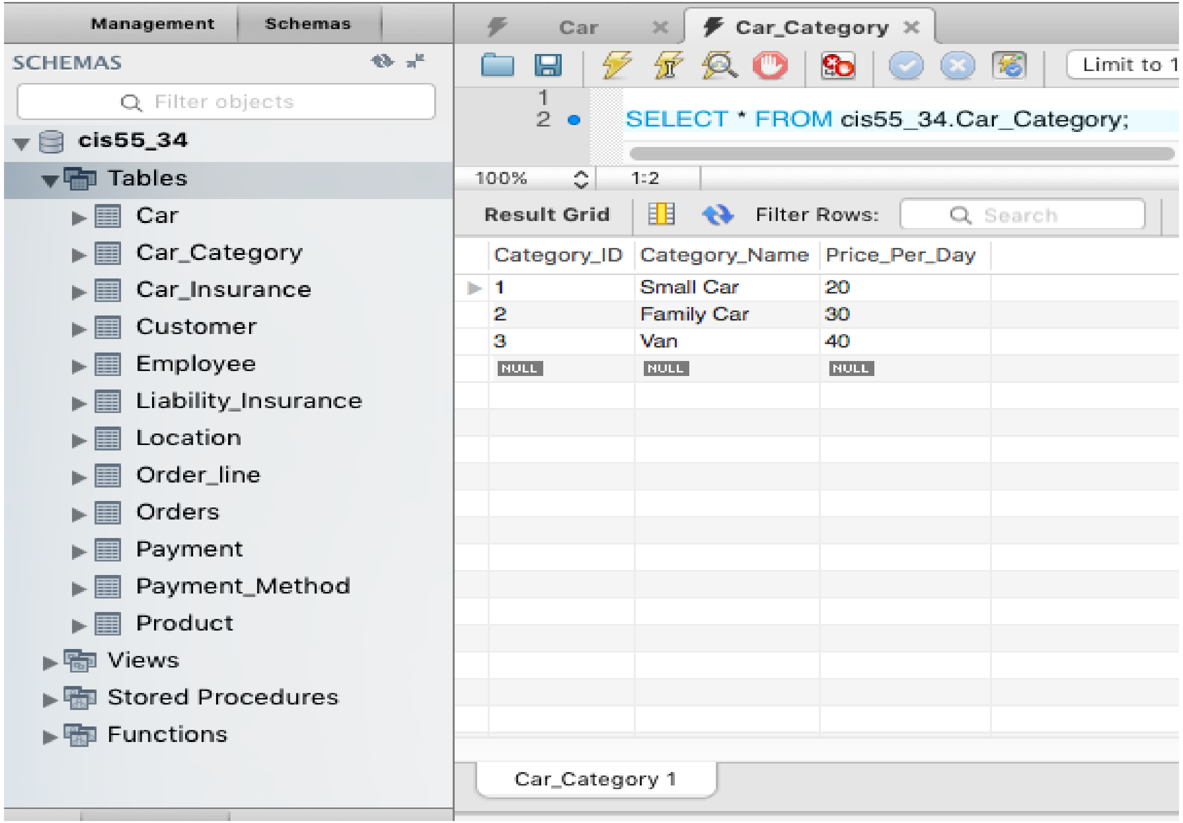
```
1  SELECT * FROM cis55_34.Product;
```

Product_ID	Description
1	Car Insurance Small
2	Car Insurance Family
3	Car Insurance Van
4	Liability Insurance Standard Protection
5	Liability Insurance Premium Protection
6	Family Car
7	Small Car
8	Family Car
9	Small Car
10	Family Car
11	Family Car
12	Small Car
13	Van
14	Family Car
15	Van
16	Family Car
17	Small Car

Continue Tables and Their Implementation

4. Car Category Table

```
Create table Car_Category
(
Category_ID int NOT NULL,
Category_Name varchar(25),
Price_Per_Day float,
Primary key (Category_ID)
);
```



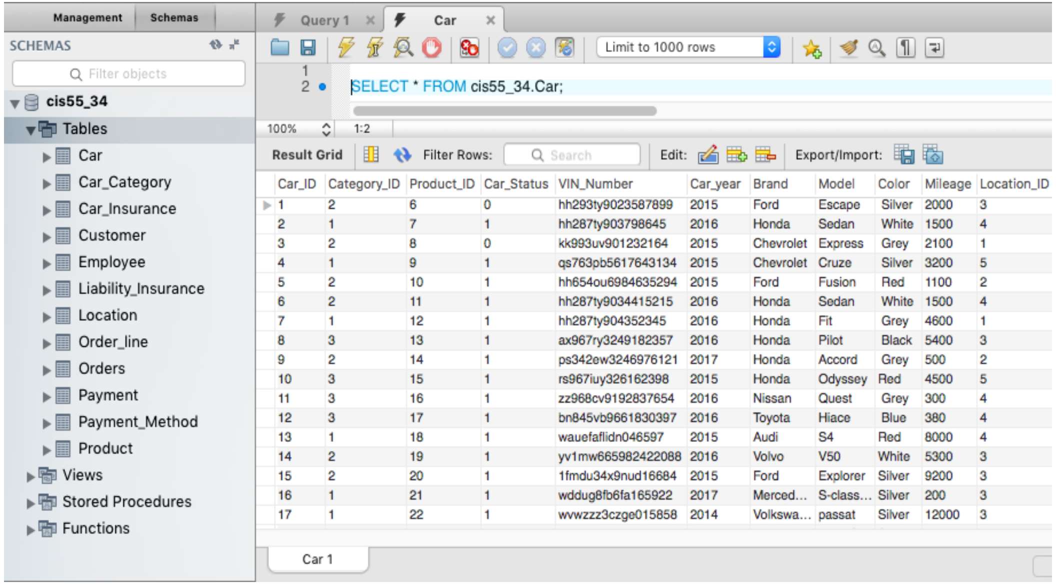
```
SELECT * FROM cis55_34.Car_Category;
```

Category_ID	Category_Name	Price_Per_Day
1	Small Car	20
2	Family Car	30
3	Van	40
NULL	NULL	NULL

Continue Tables and Their Implementation

5. Car Table

```
Create table Car(
Car_ID int NOT NULL,
Category_ID int,
Product_ID int NOT NULL,
Car_Status int,
VIN_Number varchar(17),
Car_year int,
Brand varchar(30),
Model varchar(30),
Color varchar(30),
Mileage float,
Location_ID int NOT NULL,
Primary key (Car_ID),
Foreign key (Product_ID) references Product(Product_ID),
Foreign key (Category_ID) references Car_Category(Category_ID),
Foreign key (Location_ID) references Location(Location_ID)
);
```



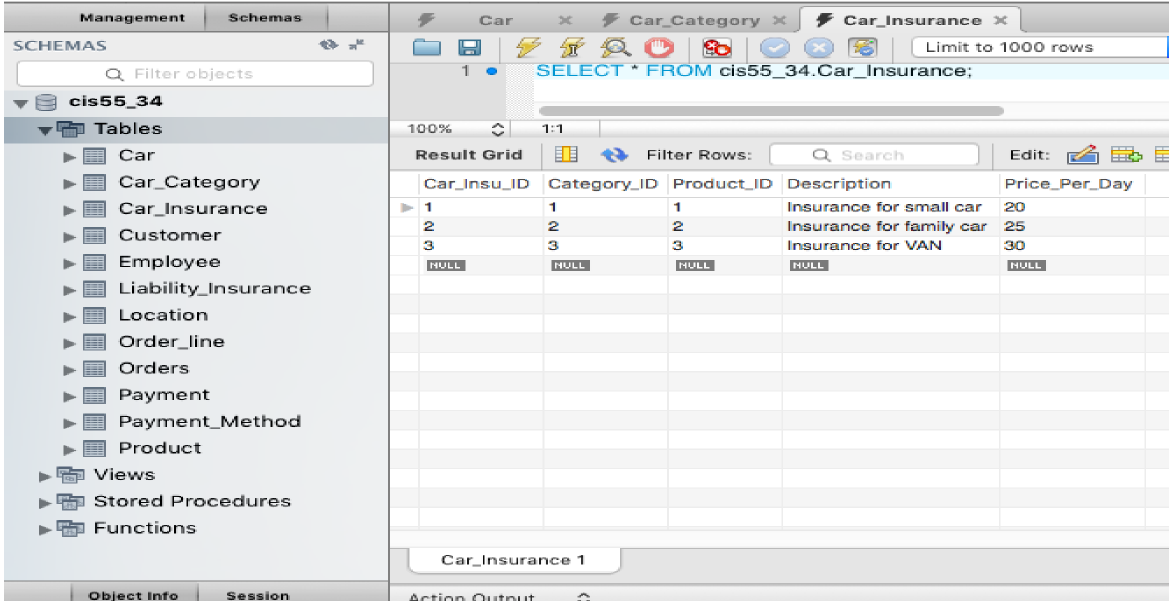
```
SELECT * FROM cis55_34.Car;
```

Car_ID	Category_ID	Product_ID	Car_Status	VIN_Number	Car_year	Brand	Model	Color	Mileage	Location_ID
1	2	6	0	hh293ty9023587899	2015	Ford	Escape	Silver	2000	3
2	1	7	1	hh287ty903798645	2016	Honda	Sedan	White	1500	4
3	2	8	0	kk993uv901232164	2015	Chevrolet	Express	Grey	2100	1
4	1	9	1	qs763pb5617643134	2015	Chevrolet	Cruze	Silver	3200	5
5	2	10	1	hh654ou6984635294	2015	Ford	Fusion	Red	1100	2
6	2	11	1	hh287ty9034415215	2016	Honda	Sedan	White	1500	4
7	1	12	1	hh287ty904352345	2016	Honda	Fit	Grey	4600	1
8	3	13	1	ax967ry3249182357	2016	Honda	Pilot	Black	5400	3
9	2	14	1	ps342ew3246976121	2017	Honda	Accord	Grey	500	2
10	3	15	1	rs967iuy326162398	2015	Honda	Odyssey	Red	4500	5
11	3	16	1	zz968cv9192837654	2016	Nissan	Quest	Grey	300	4
12	3	17	1	bn845vb9661830397	2016	Toyota	Hiace	Blue	380	4
13	1	18	1	wauefaflidn046597	2015	Audi	S4	Red	8000	4
14	2	19	1	yv1mw665982422088	2016	Volvo	V50	White	5300	3
15	2	20	1	1fmdu34x9nud16684	2015	Ford	Explorer	Silver	9200	3
16	1	21	1	wddug8fb6fa165922	2017	Merced...	S-class...	Silver	200	3
17	1	22	1	wvwzzz3czge015858	2014	Volkswa...	passat	Silver	12000	3

Continue Tables and Their Implementation

6. Car_Insurance Table

```
Create table Car_Insurance
(
Car_Insu_ID int NOT NULL,
Category_ID int,
Product_ID int NOT NULL,
Description varchar(250),
Price_Per_Day float,
Primary key (Car_Insu_ID),
Foreign key (Product_ID) references
Product(Product_ID),
Foreign key (Category_ID) references
Car_Category(Category_ID)
);
```



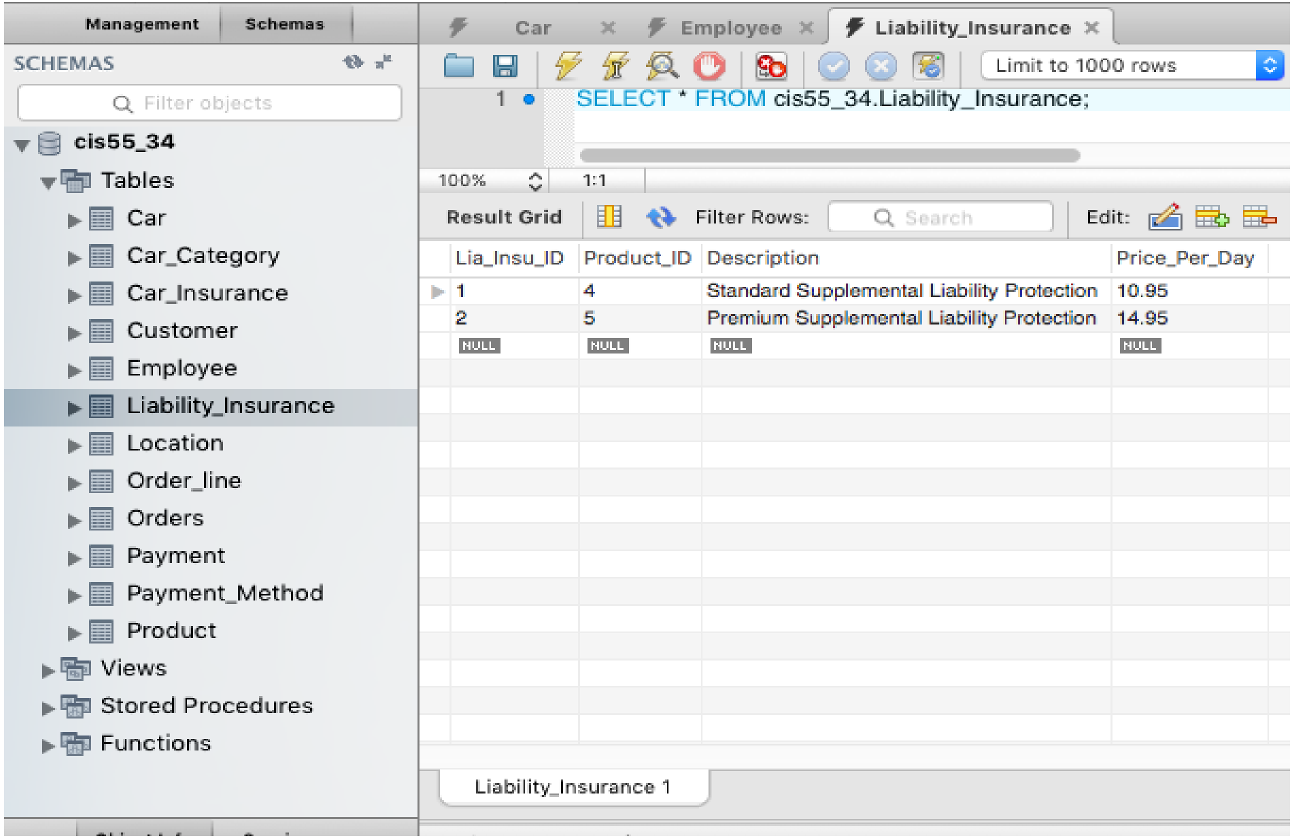
```
SELECT * FROM cis55_34.Car_Insurance;
```

Car_Insu_ID	Category_ID	Product_ID	Description	Price_Per_Day
1	1	1	Insurance for small car	20
2	2	2	Insurance for family car	25
3	3	3	Insurance for VAN	30
NULL	NULL	NULL	NULL	NULL

Continue Tables and Their Implementation

7. Liability Insurance Table

Create table Liability_Insurance
(
Lia_Insu_ID int NOT NULL,
Product_ID int NOT NULL,
Description varchar(250),
Price_Per_Day float,
Primary key (Lia_Insu_ID),
Foreign key (Product_ID) references Product(Product_ID)
);



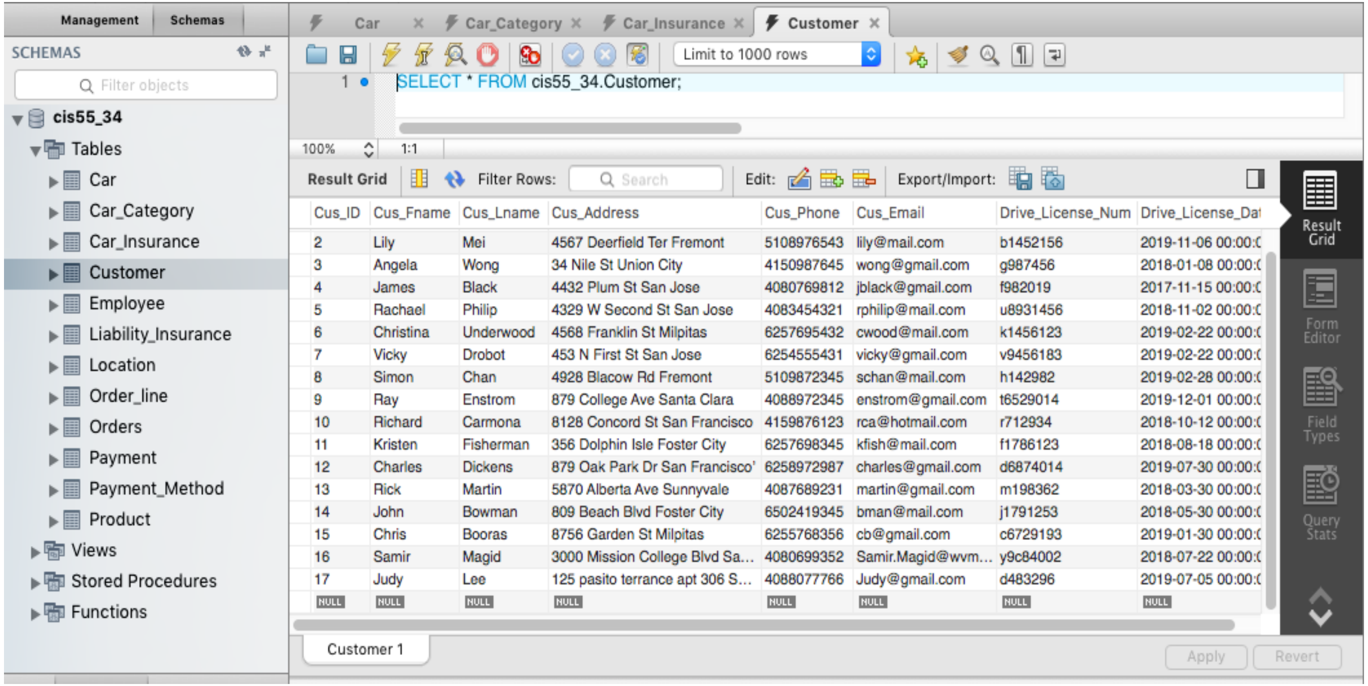
```
SELECT * FROM cis55_34.Liability_Insurance;
```

Lia_Insu_ID	Product_ID	Description	Price_Per_Day
1	4	Standard Supplemental Liability Protection	10.95
2	5	Premium Supplemental Liability Protection	14.95
NULL	NULL	NULL	NULL

Continue Tables and Their Implementation

8. Customer

```
Create table Customer
(
Cus_ID int NOT NULL,
Cus_Fname varchar(25),
Cus_Lname varchar(25),
Cus_Address varchar(50),
Cus_Phone varchar(12),
Cus_Email varchar(25),
Drive_License_Num varchar(15),
Drive_License_Date datetime,
Primary key (Cus_ID)
);
```



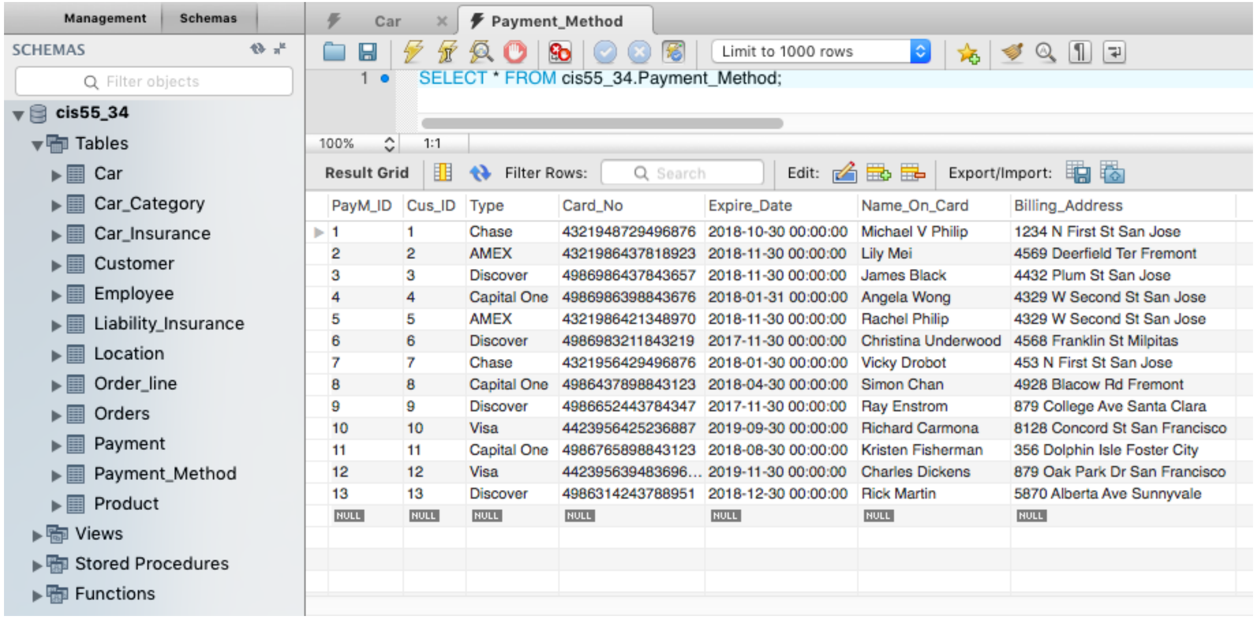
```
SELECT * FROM cis55_34.Customer;
```

Cus_ID	Cus_Fname	Cus_Lname	Cus_Address	Cus_Phone	Cus_Email	Drive_License_Num	Drive_License_Dat
2	Lily	Mei	4567 Deerfield Ter Fremont	5108976543	lily@mail.com	b1452156	2019-11-06 00:00:0
3	Angela	Wong	34 Nile St Union City	4150987645	wong@gmail.com	g987456	2018-01-08 00:00:0
4	James	Black	4432 Plum St San Jose	4080769812	jblack@gmail.com	f982019	2017-11-15 00:00:0
5	Rachael	Philip	4329 W Second St San Jose	4083454321	rphilip@mail.com	u8931456	2018-11-02 00:00:0
6	Christina	Underwood	4568 Franklin St Milpitas	6257695432	cwood@mail.com	k1456123	2019-02-22 00:00:0
7	Vicky	Drobot	453 N First St San Jose	6254555431	vicky@gmail.com	v9456183	2019-02-22 00:00:0
8	Simon	Chan	4928 Blacow Rd Fremont	5109872345	schan@mail.com	h142982	2019-02-28 00:00:0
9	Ray	Enstrom	879 College Ave Santa Clara	4088972345	enstrom@gmail.com	t6529014	2019-12-01 00:00:0
10	Richard	Carmona	8128 Concord St San Francisco	4159876123	rca@hotmail.com	r712934	2018-10-12 00:00:0
11	Kristen	Fisherman	356 Dolphin Isle Foster City	6257698345	kfish@mail.com	f1786123	2018-08-18 00:00:0
12	Charles	Dickens	879 Oak Park Dr San Francisco'	6258972987	charles@gmail.com	d6874014	2019-07-30 00:00:0
13	Rick	Martin	5870 Alberta Ave Sunnyvale	4087689231	martin@gmail.com	m198362	2018-03-30 00:00:0
14	John	Bowman	809 Beach Blvd Foster City	6502419345	bman@mail.com	j1791253	2018-05-30 00:00:0
15	Chris	Booras	8756 Garden St Milpitas	6255768356	cb@gmail.com	c6729193	2019-01-30 00:00:0
16	Samir	Magid	3000 Mission College Blvd Sa…	4080699352	Samir.Magid@wvm…	y9c84002	2018-07-22 00:00:0
17	Judy	Lee	125 pasito terrance apt 306 S…	4088077766	Judy@gmail.com	d483296	2019-07-05 00:00:0
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Tables and Their Implementation

9. Payment Method Table

```
Create table Payment_Method
(
PayM_ID int NOT NULL,
Cus_ID int,
Type varchar(50),
Card_No varchar(25),
Expire_Date datetime,
Name_On_Card varchar(50),
Billing_Address varchar(50),
Primary key (PayM_ID),
Foreign key (Cus_ID) references
Customer(Cus_ID)
);
```



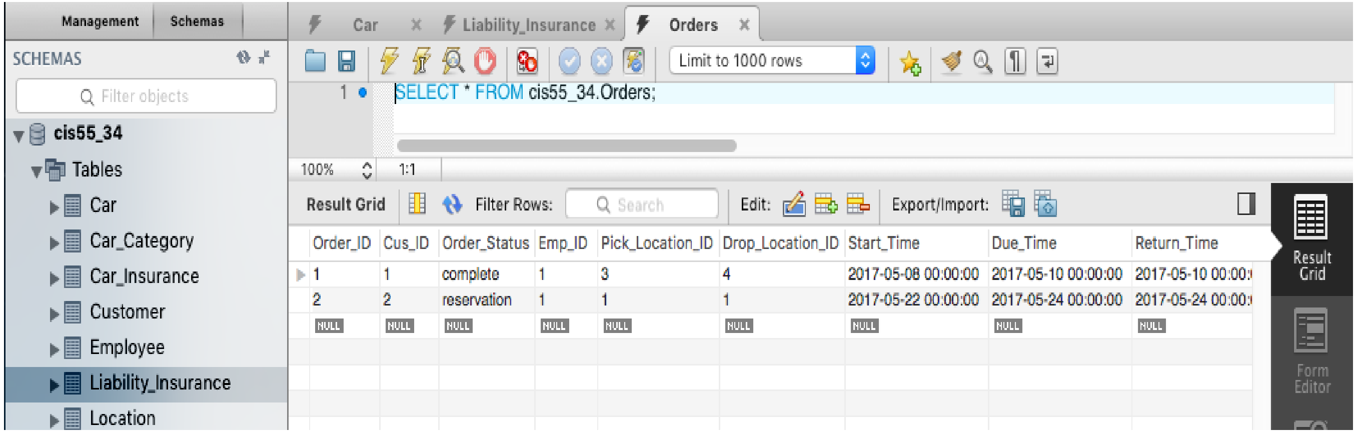
```
SELECT * FROM cis55_34.Payment_Method;
```

PayM_ID	Cus_ID	Type	Card_No	Expire_Date	Name_On_Card	Billing_Address
1	1	Chase	4321948729496876	2018-10-30 00:00:00	Michael V Philip	1234 N First St San Jose
2	2	AMEX	4321986437818923	2018-11-30 00:00:00	Lily Mei	4569 Deerfield Ter Fremont
3	3	Discover	4986986437843657	2018-11-30 00:00:00	James Black	4432 Plum St San Jose
4	4	Capital One	4986986398843676	2018-01-31 00:00:00	Angela Wong	4329 W Second St San Jose
5	5	AMEX	4321986421348970	2018-11-30 00:00:00	Rachel Philip	4329 W Second St San Jose
6	6	Discover	4986983211843219	2017-11-30 00:00:00	Christina Underwood	4568 Franklin St Milpitas
7	7	Chase	4321956429496876	2018-01-30 00:00:00	Vicky Drobot	453 N First St San Jose
8	8	Capital One	4986437898843123	2018-04-30 00:00:00	Simon Chan	4928 Blacow Rd Fremont
9	9	Discover	4986652443784347	2017-11-30 00:00:00	Ray Enstrom	879 College Ave Santa Clara
10	10	Visa	4423956425236887	2019-09-30 00:00:00	Richard Carmona	8128 Concord St San Francisco
11	11	Capital One	4986765898843123	2018-08-30 00:00:00	Kristen Fisherman	356 Dolphin Isle Foster City
12	12	Visa	442395639483696...	2019-11-30 00:00:00	Charles Dickens	879 Oak Park Dr San Francisco
13	13	Discover	4986314243788951	2018-12-30 00:00:00	Rick Martin	5870 Alberta Ave Sunnyvale
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Continued Tables and Their Implementation

10. Orders

```
Create table Orders
(
Order_ID int NOT NULL,
Cus_ID int,
Order_Status varchar(20),
Emp_ID int,
Pick_Location_ID int,
Drop_Location_ID int,
Start_Time datetime,
Due_Time datetime,
Return_Time datetime,
Total_Cost float,
Primary key (Order_ID),
Foreign key (Cus_ID) references Customer(Cus_ID),
Foreign key (Emp_ID) references Employee(Emp_ID),
Foreign key (Pick_Location_ID) references Location(Location_ID),
Foreign key (Drop_Location_ID) references Location(Location_ID)
);
```



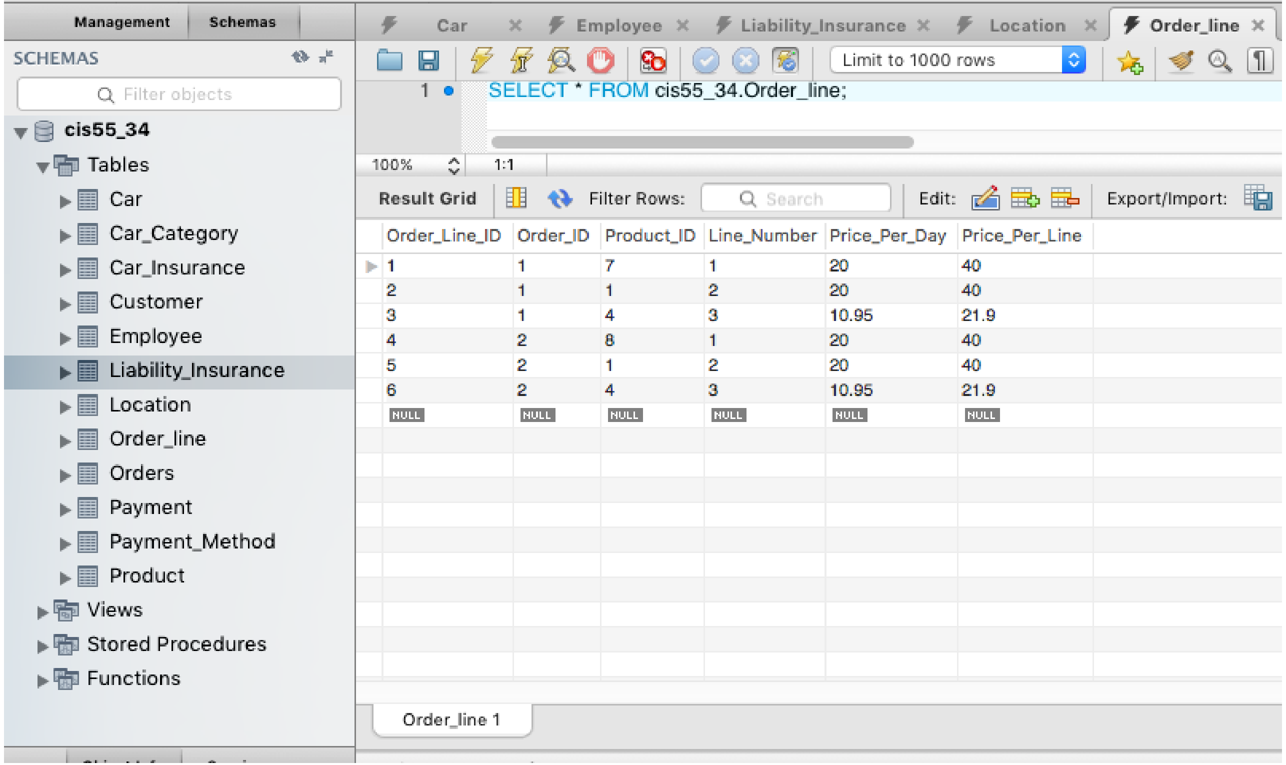
```
SELECT * FROM cis55_34.Orders;
```

Order_ID	Cus_ID	Order_Status	Emp_ID	Pick_Location_ID	Drop_Location_ID	Start_Time	Due_Time	Return_Time
1	1	complete	1	3	4	2017-05-08 00:00:00	2017-05-10 00:00:00	2017-05-10 00:00:
2	2	reservation	1	1	1	2017-05-22 00:00:00	2017-05-24 00:00:00	2017-05-24 00:00:
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Continued Tables and Their Implementation

11. Order_line Table

```
Create table Order_line
(
Order_Line_ID int NOT NULL,
Order_ID int,
Product_ID int,
Line_NUmber int,
Price_Per_Day float,
Price_Per_Line float,
Primary key (Order_Line_ID),
Foreign key (Order_ID)
references Orders(Order_ID),
Foreign key (Product_ID)
references Product(Product_ID)
);
```



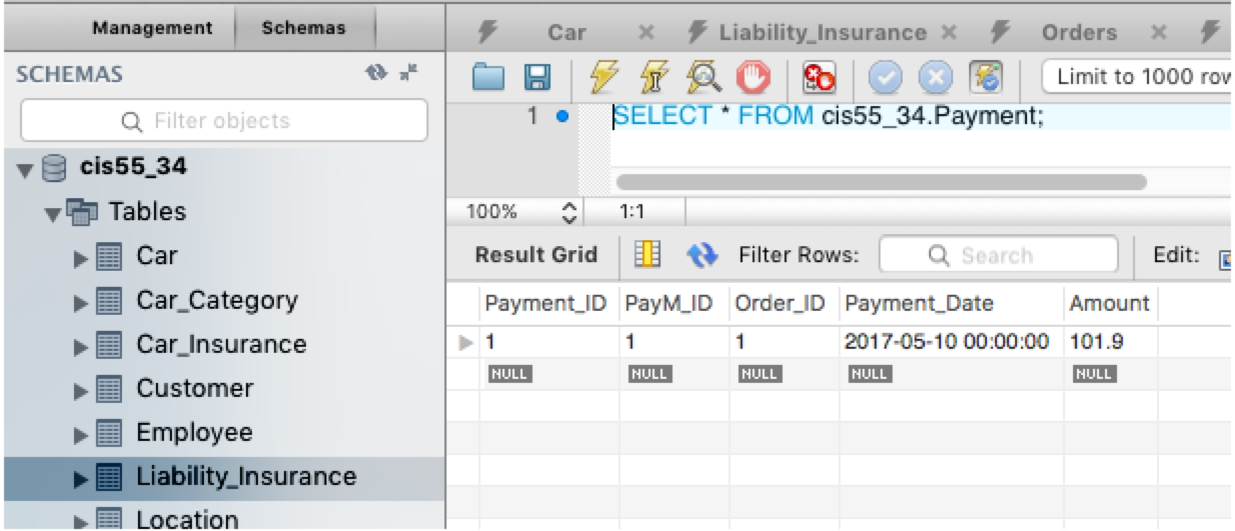
```
SELECT * FROM cis55_34.Order_line;
```

Order_Line_ID	Order_ID	Product_ID	Line_Number	Price_Per_Day	Price_Per_Line
1	1	7	1	20	40
2	1	1	2	20	40
3	1	4	3	10.95	21.9
4	2	8	1	20	40
5	2	1	2	20	40
6	2	4	3	10.95	21.9
NULL	NULL	NULL	NULL	NULL	NULL

Continued Tables and Their Implementation

12. Payment

```
Create table Payment
(
Payment_ID int NOT NULL,
PayM_ID int,
Order_ID int NOT NULL,
Payment_Date datetime,
Amount float,
Primary key (Payment_ID),
Foreign key (Order_ID)
references Orders(Order_ID),
Foreign key (PayM_ID) references
Payment_Method(PayM_ID),
Foreign key (Invoice_ID) references
Invoice(Invoice_ID)
);
```

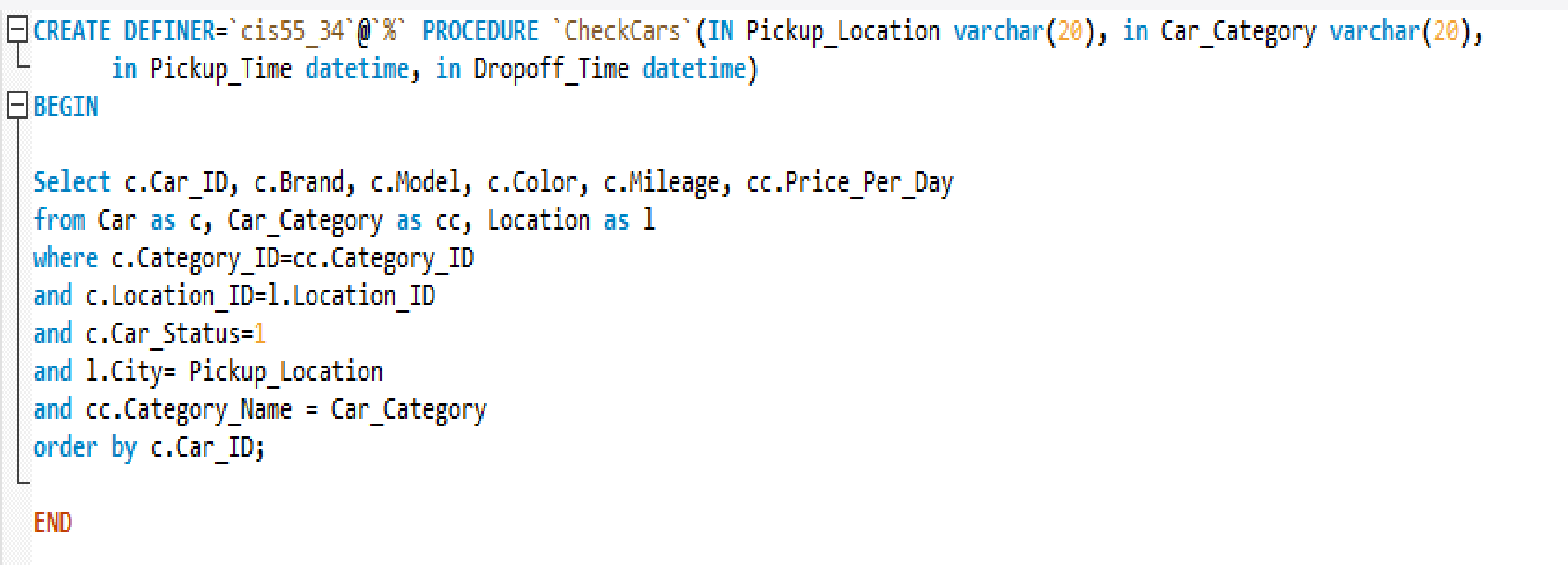


```
1 SELECT * FROM cis55_34.Payment;
```

Payment_ID	PayM_ID	Order_ID	Payment_Date	Amount
1	1	1	2017-05-10 00:00:00	101.9
NULL	NULL	NULL	NULL	NULL

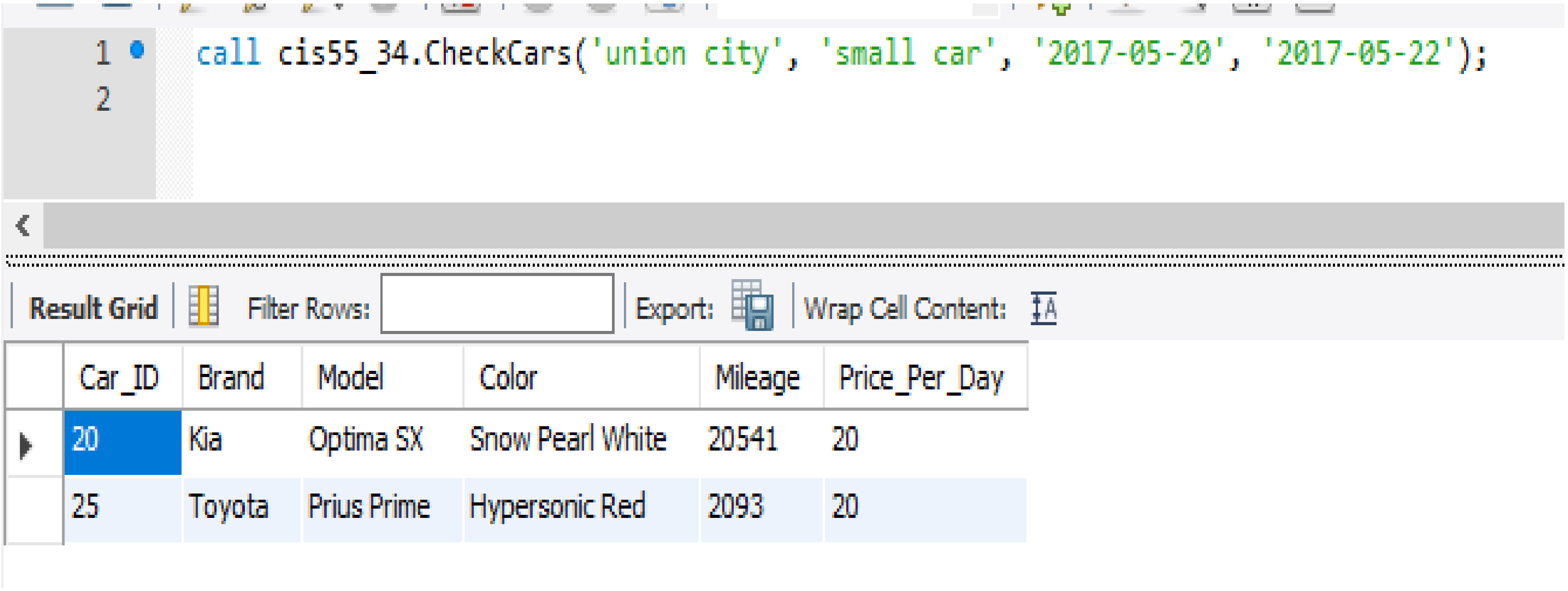
6.Stored Procedures/Functions/Triggers/Views

i) **generate a list of rental cars information available to satisfy the criteria/quote of customers.**



```
CREATE DEFINER=`cis55_34`@`%` PROCEDURE `CheckCars`(IN Pickup_Location varchar(20), in Car_Category varchar(20),
      in Pickup_Time datetime, in Dropoff_Time datetime)
BEGIN

Select c.Car_ID, c.Brand, c.Model, c.Color, c.Mileage, cc.Price_Per_Day
from Car as c, Car_Category as cc, Location as l
where c.Category_ID=cc.Category_ID
and c.Location_ID=l.Location_ID
and c.Car_Status=1
and l.City= Pickup_Location
and cc.Category_Name = Car_Category
order by c.Car_ID;

END
```

Output result : CheckCars stored procedure

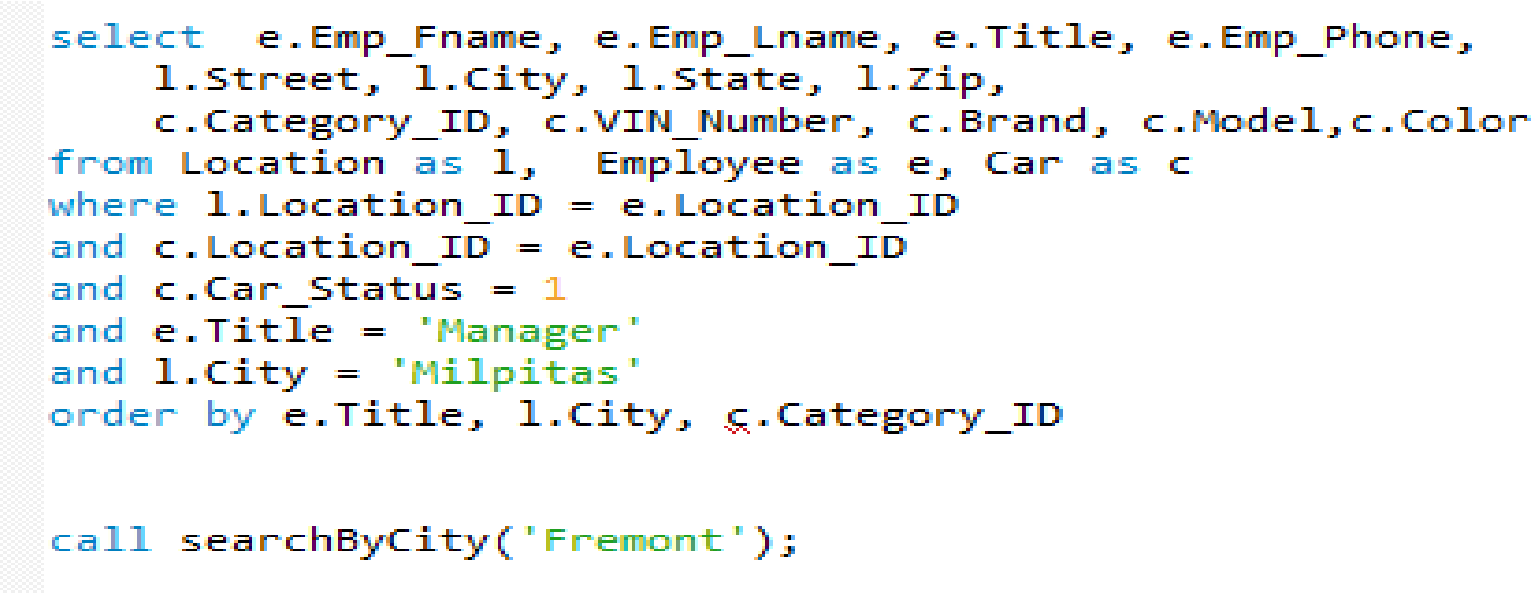


```
call cis55_34.CheckCars('union city', 'small car', '2017-05-20', '2017-05-22');
```

Car_ID	Brand	Model	Color	Mileage	Price_Per_Day
20	Kia	Optima SX	Snow Pearl White	20541	20
25	Toyota	Prius Prime	Hypersonic Red	2093	20

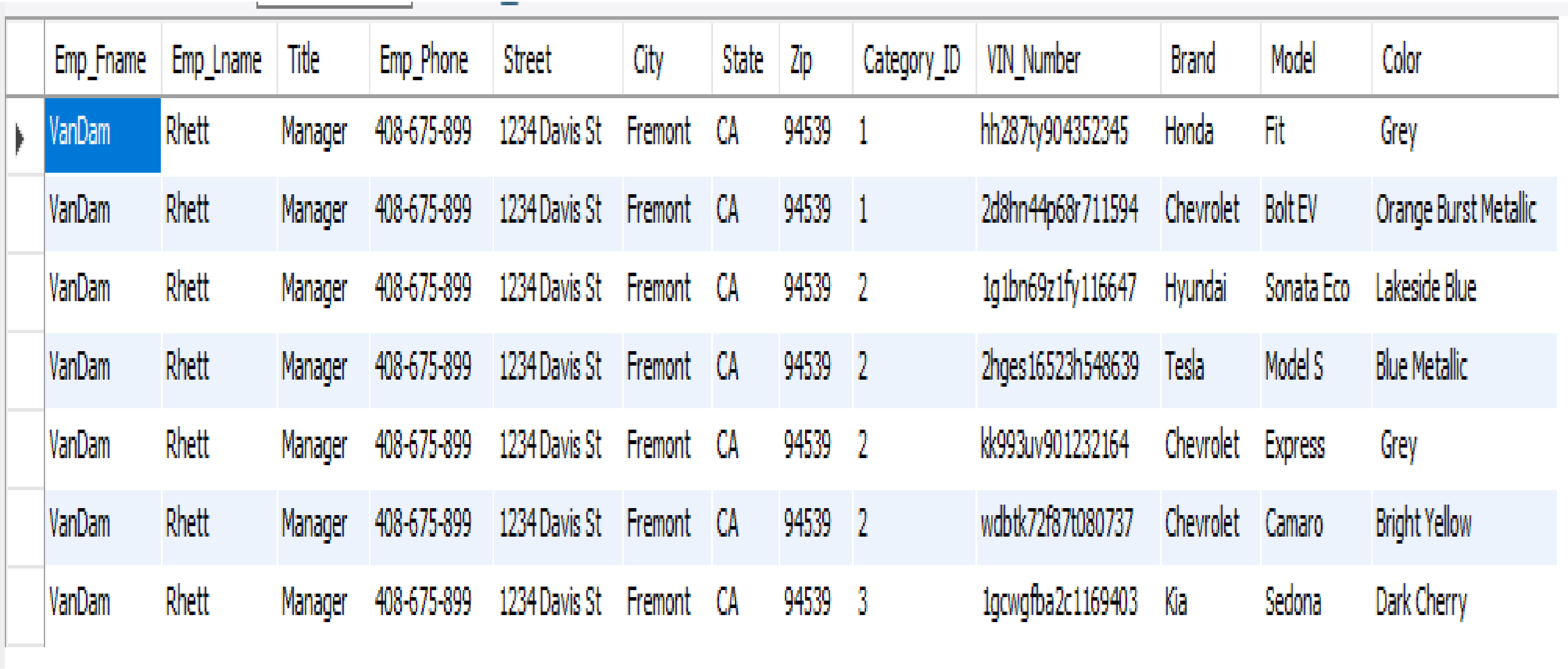
Continued Stored Procedure/ Function/Trigger/View

ii) generate a list of available rental cars for a specific city along with the Manager’s contact



```
select  e.Emp_Fname, e.Emp_Lname, e.Title, e.Emp_Phone,
    l.Street, l.City, l.State, l.Zip,
    c.Category_ID, c.VIN_Number, c.Brand, c.Model,c.Color
from Location as l,  Employee as e, Car as c
where l.Location_ID = e.Location_ID
and c.Location_ID = e.Location_ID
and c.Car_Status = 1
and e.Title = 'Manager'
and l.City = 'Milpitas'
order by e.Title, l.City, c.Category_ID


call searchByCity('Fremont');
```

Output result : searchByCity(‘cityname’)

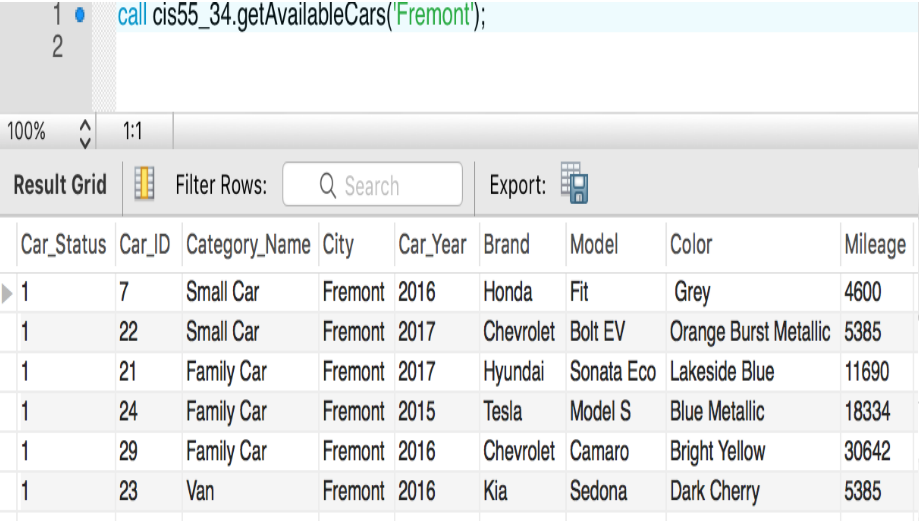


Emp_Fname	Emp_Lname	Title	Emp_Phone	Street	City	State	Zip	Category_ID	VIN_Number	Brand	Model	Color
VanDam	Rhett	Manager	408-675-899	1234 Davis St	Fremont	CA	94539	1	hh287ty904352345	Honda	Fit	Grey
VanDam	Rhett	Manager	408-675-899	1234 Davis St	Fremont	CA	94539	1	2d8hn44p68r711594	Chevrolet	Bolt EV	Orange Burst Metallic
VanDam	Rhett	Manager	408-675-899	1234 Davis St	Fremont	CA	94539	2	1g1bn69z1fy116647	Hyundai	Sonata Eco	Lakeside Blue
VanDam	Rhett	Manager	408-675-899	1234 Davis St	Fremont	CA	94539	2	2hges16523h548639	Tesla	Model S	Blue Metallic
VanDam	Rhett	Manager	408-675-899	1234 Davis St	Fremont	CA	94539	2	kk993uv901232164	Chevrolet	Express	Grey
VanDam	Rhett	Manager	408-675-899	1234 Davis St	Fremont	CA	94539	2	wdbtk72f87t080737	Chevrolet	Camaro	Bright Yellow
VanDam	Rhett	Manager	408-675-899	1234 Davis St	Fremont	CA	94539	3	1gcwgfba2c169403	Kia	Sedona	Dark Cherry

Continued Stored Procedure/ Function/Trigger/View

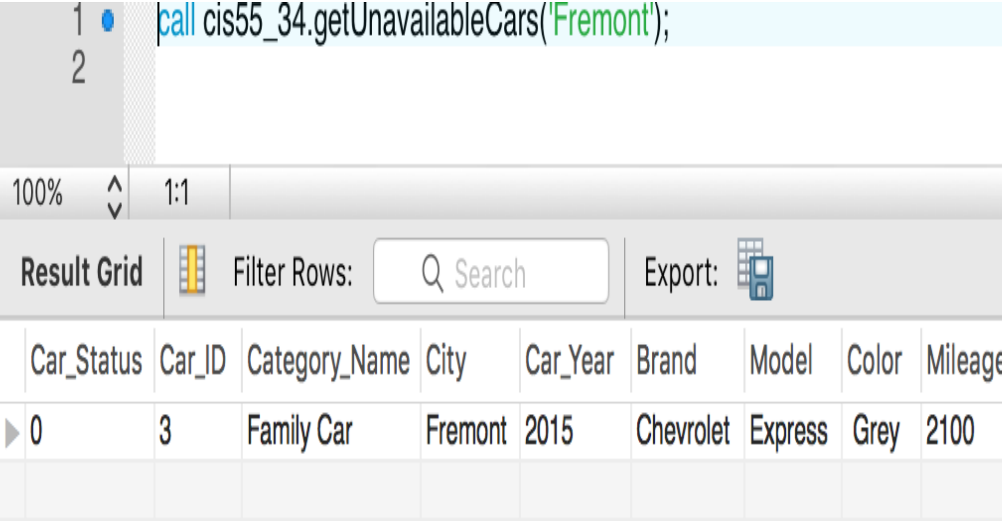
iii) list a detail information of rental cars that are already rent out in a specific city and those available to rent in that city.

We have two stored procedures getAvailableCars and getUnavailableCars to to implement this.



```
call cis55_34.getAvailableCars('Fremont');
```

Car_Status	Car_ID	Category_Name	City	Car_Year	Brand	Model	Color	Mileage
1	7	Small Car	Fremont	2016	Honda	Fit	Grey	4600
1	22	Small Car	Fremont	2017	Chevrolet	Bolt EV	Orange Burst Metallic	5385
1	21	Family Car	Fremont	2017	Hyundai	Sonata Eco	Lakeside Blue	11690
1	24	Family Car	Fremont	2015	Tesla	Model S	Blue Metallic	18334
1	29	Family Car	Fremont	2016	Chevrolet	Camaro	Bright Yellow	30642
1	23	Van	Fremont	2016	Kia	Sedona	Dark Cherry	5385

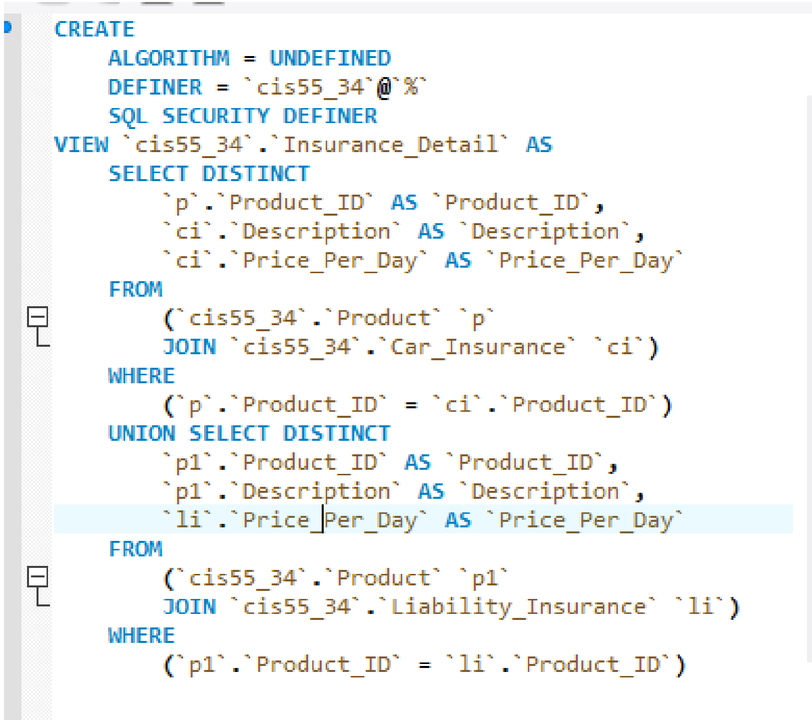


```
call cis55_34.getUnavailableCars('Fremont');
```

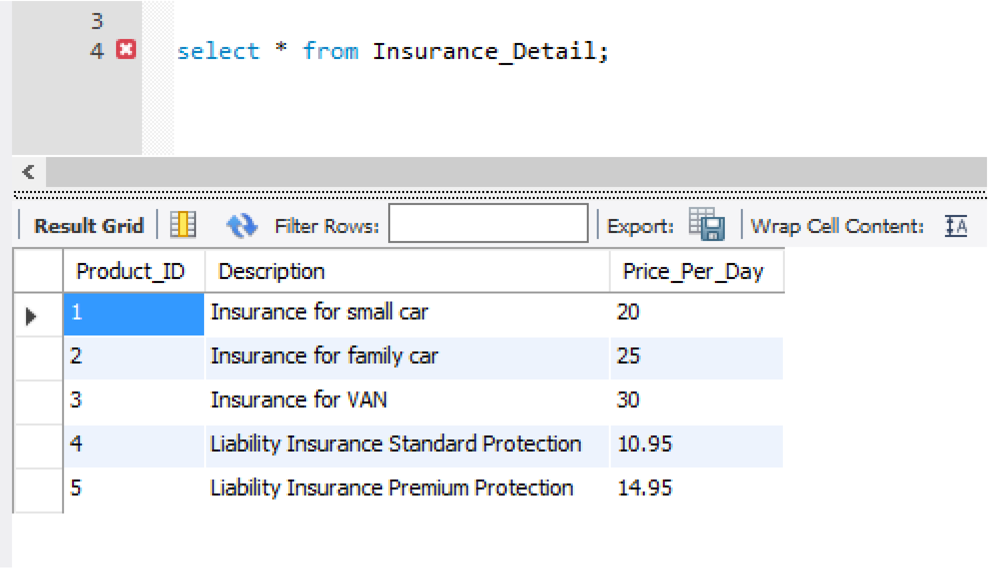
Car_Status	Car_ID	Category_Name	City	Car_Year	Brand	Model	Color	Mileag
0	3	Family Car	Fremont	2015	Chevrolet	Express	Grey	2100

Continued Stored Procedure/ Function/Trigger/View

iv) list a detail information of Car Insurance and Liability Insurance options for the customers.



```sql
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `cis55_34`@`%`
    SQL SECURITY DEFINER
VIEW `cis55_34`.`Insurance_Detail` AS
    SELECT DISTINCT
        `p`.`Product_ID` AS `Product_ID`,
        `ci`.`Description` AS `Description`,
        `ci`.`Price_Per_Day` AS `Price_Per_Day`
    FROM
        (`cis55_34`.`Product` `p`
        JOIN `cis55_34`.`Car_Insurance` `ci`)
    WHERE
        (`p`.`Product_ID` = `ci`.`Product_ID`)
    UNION SELECT DISTINCT
        `p1`.`Product_ID` AS `Product_ID`,
        `p1`.`Description` AS `Description`,
        `li`.`Price_Per_Day` AS `Price_Per_Day`
    FROM
        (`cis55_34`.`Product` `p1`
        JOIN `cis55_34`.`Liability_Insurance` `li`)
    WHERE
        (`p1`.`Product_ID` = `li`.`Product_ID`)
```



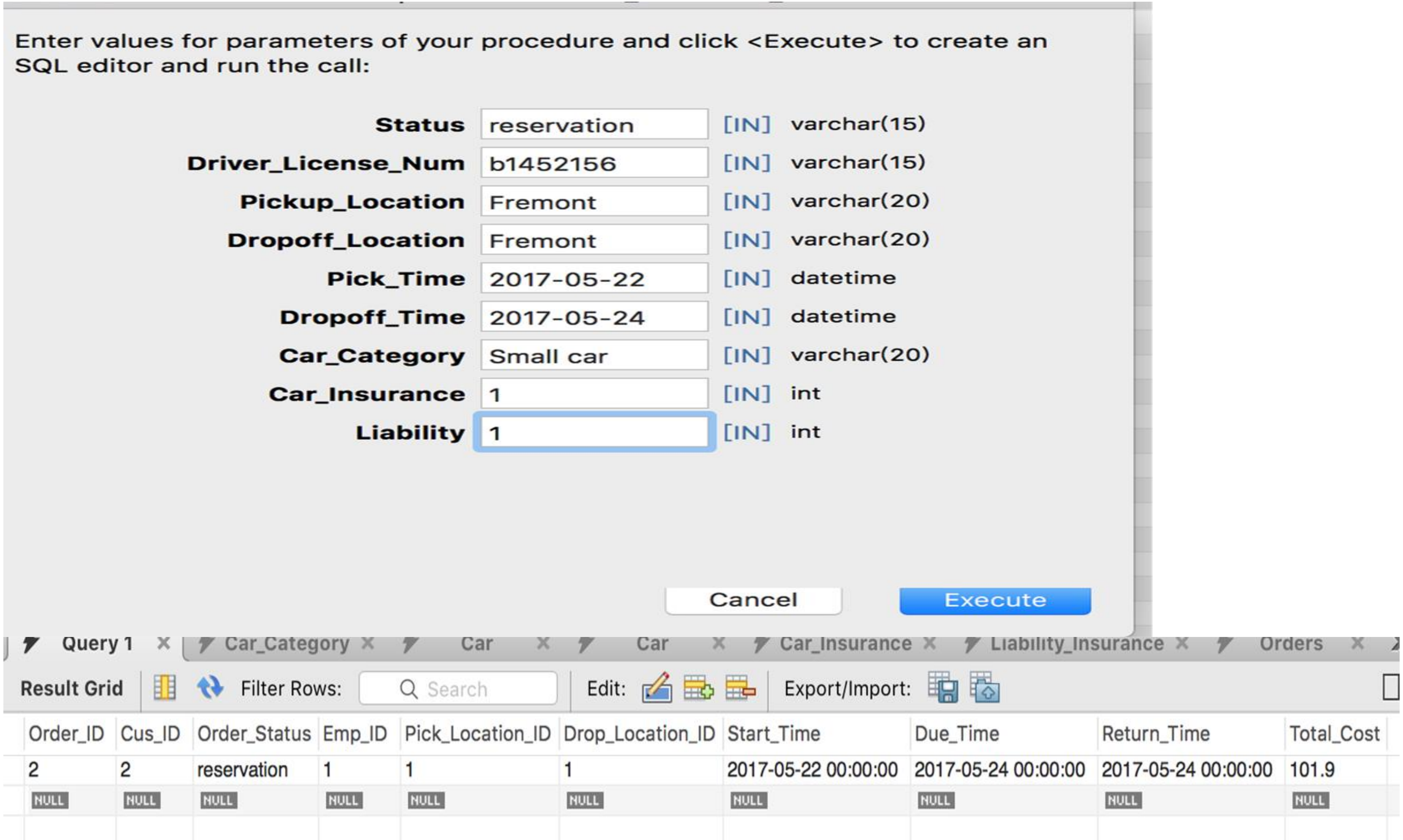
```sql
select * from Insurance_Detail;
```

Product_ID	Description	Price_Per_Day
1	Insurance for small car	20
2	Insurance for family car	25
3	Insurance for VAN	30
4	Liability Insurance Standard Protection	10.95
5	Liability Insurance Premium Protection	14.95

Continued Stored Procedure/ Function/Trigger/View

v) generate an order for customers including detail information of rental car, reservation information, payment details.

```
CREATE DEFINER=`cis55_34`@`%` PROCEDURE `create_orders`(in Status varchar(15), in Driver_License_Num varchar(15), in Pickup_Location varchar(20),  in Dropoff_Location
varchar(20), in Pick_Time datetime, in Dropoff_Time datetime, in Car_Category varchar(20), in Car_Insurance int, in Liability int)
begin
Declare Order_ID int, Cus_ID int, Emp_ID int,  Pick_Location_ID int, Drop_Location_ID int, Liability_Cost float,  Rental_Cost float, CarInsurance_Cost float, Total_Cost float;
SELECT (COUNT(*) + 1) INTO Order_ID FROM cis55_34.Orders LIMIT 1;
SELECT cis55_34.Customer.Cus_ID INTO Cus_ID FROM cis55_34.Customer
WHERE cis55_34.Customer.Drive_License_Num = Driver_License_Num LIMIT 1;
SELECT cis55_34.Employee.Emp_ID INTO Emp_ID FROM cis55_34.Location, cis55_34.Employee WHERE cis55_34.Location.City = Pickup_Location  AND
cis55_34.Location.Location_ID = Employee.Location_ID LIMIT 1;
SELECT cis55_34.Location.Location_ID INTO Pick_Location_ID FROM
cis55_34.Location WHERE cis55_34.Location.City = Pickup_Location LIMIT 1;
SELECT cis55_34.Location.Location_ID INTO Drop_Location_ID FROM
cis55_34.Location WHERE cis55_34.Location.City = Dropoff_Location LIMIT 1;
SELECT Check_Liability_Fees(Pick_Time, Dropoff_Time, Liability) INTO Liability_Cost;
SELECT Check_Rental_Fees(Car_Category, Pick_Time, Dropoff_Time) INTO Rental_Cost;
SELECT Check_CarInsurance_Fees(Car_Category, Pick_Time, Dropoff_Time,
 Car_Insurance) INTO CarInsurance_Cost;
SELECT (Rental_Cost + Liability_Cost + CarInsurance_Cost) INTO Total_Cost;
Insert into Orders values (Order_ID, Cus_ID, Status, Emp_ID, Pick_Location_ID,
Drop_Location_ID, Pick_Time, Dropoff_Time, Dropoff_Time, Total_Cost);
End
```



Enter values for parameters of your procedure and click <Execute> to create an SQL editor and run the call:

Status	reservation	[IN]	varchar(15)
Driver_License_Num	b1452156	[IN]	varchar(15)
Pickup_Location	Fremont	[IN]	varchar(20)
Dropoff_Location	Fremont	[IN]	varchar(20)
Pick_Time	2017-05-22	[IN]	datetime
Dropoff_Time	2017-05-24	[IN]	datetime
Car_Category	Small car	[IN]	varchar(20)
Car_Insurance	1	[IN]	int
Liability	1	[IN]	int

Order_ID	Cus_ID	Order_Status	Emp_ID	Pick_Location_ID	Drop_Location_ID	Start_Time	Due_Time	Return_Time	Total_Cost
2	2	reservation	1	1	1	2017-05-22 00:00:00	2017-05-24 00:00:00	2017-05-24 00:00:00	101.9
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Continued Stored Procedure/ Function/Trigger/View

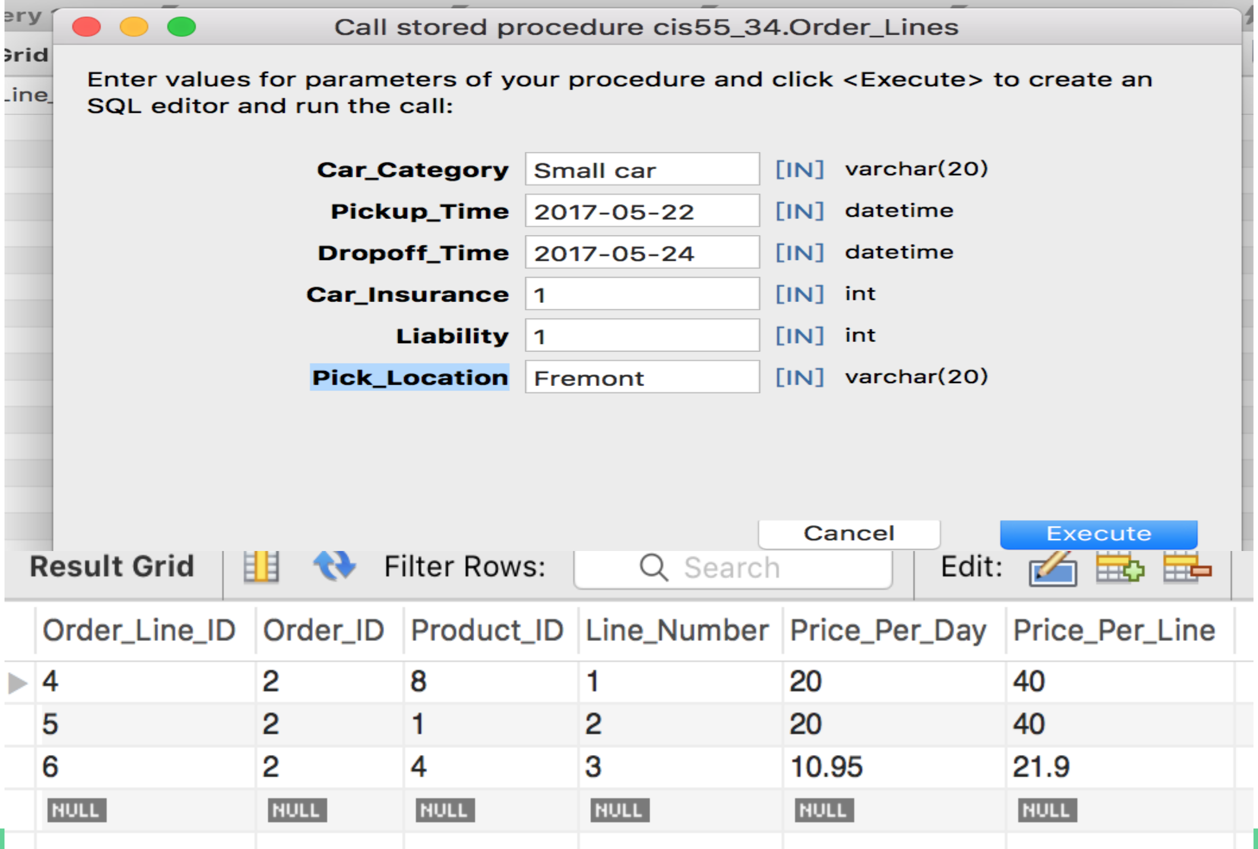
vi) Calculate Cost_Per_Line from Order line table

If a customer make a new order, we meantime make a new order_line, we create a procedure named Order_Lines to create new Order_Lines, and we also create three functions to calculate **Cost_Per_Line** for Order line table as the (Check_Liability_Fees(); Check_Rental_Fees(); Check_CarInsurance_Fees())

```
CREATE DEFINER=`cis55_34`@`%` PROCEDURE `Order_Lines`( in Car_Category varchar(20), in Pickup_Time datetime, in
Dropoff_Time datetime,
in Car_Insurance int, in Liability int, in Pick_Location varchar(20))
```

Continued Stored Procedure/ Function/Trigger/View

vi)Calculate Cost_Per_Line from Order line table



Call stored procedure cis55_34.Order_Lines

Enter values for parameters of your procedure and click <Execute> to create an SQL editor and run the call:

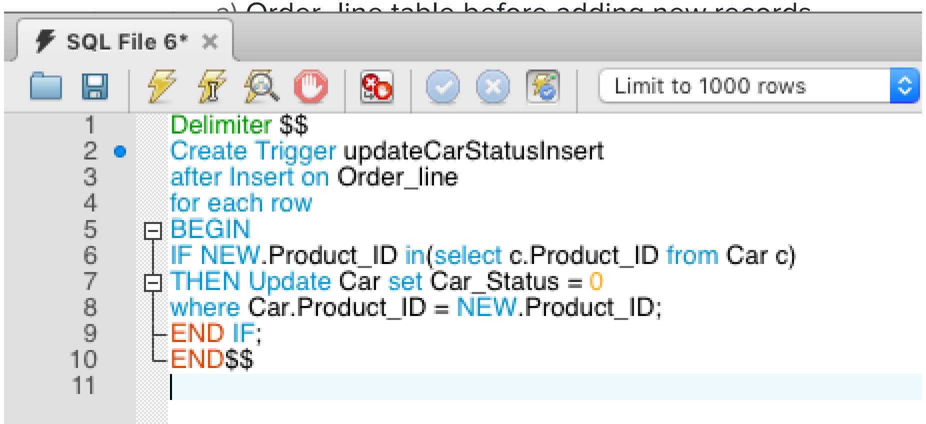
Car_Category	Small car	[IN]	varchar(20)
Pickup_Time	2017-05-22	[IN]	datetime
Dropoff_Time	2017-05-24	[IN]	datetime
Car_Insurance	1	[IN]	int
Liability	1	[IN]	int
Pick_Location	Fremont	[IN]	varchar(20)

Order_Line_ID	Order_ID	Product_ID	Line_Number	Price_Per_Day	Price_Per_Line
4	2	8	1	20	40
5	2	1	2	20	40
6	2	4	3	10.95	21.9
NULL	NULL	NULL	NULL	NULL	NULL

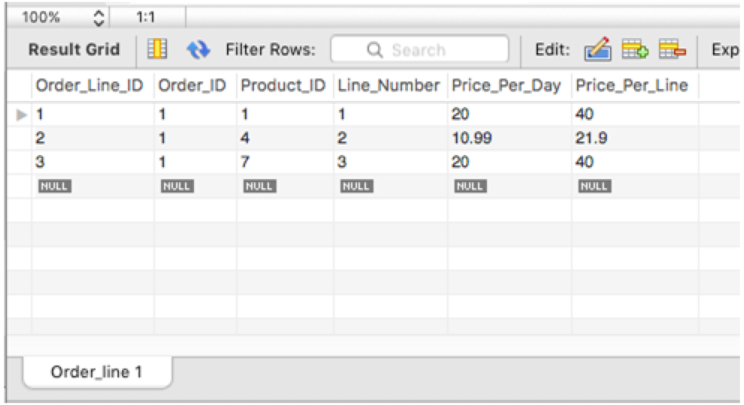
Continued Stored Procedure/ Function/Trigger/View

vii) update the status of car from available to unavailable once a customer makes his/her reservation and chooses a specific car.

a) Order_line table before adding new records

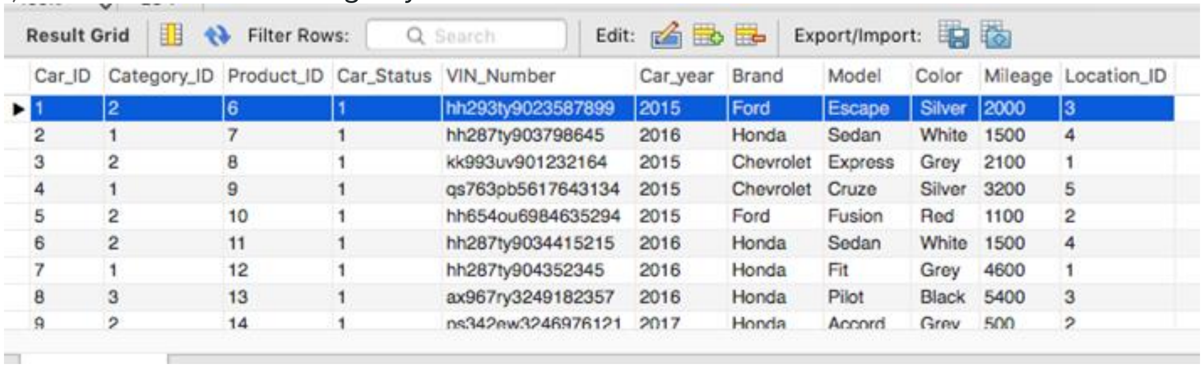


```
Delimiter $$
Create Trigger updateCarStatusInsert
after Insert on Order_line
for each row
BEGIN
IF NEW.Product_ID in(select c.Product_ID from Car c)
THEN Update Car set Car_Status = 0
where Car.Product_ID = NEW.Product_ID;
END IF;
END$$
```



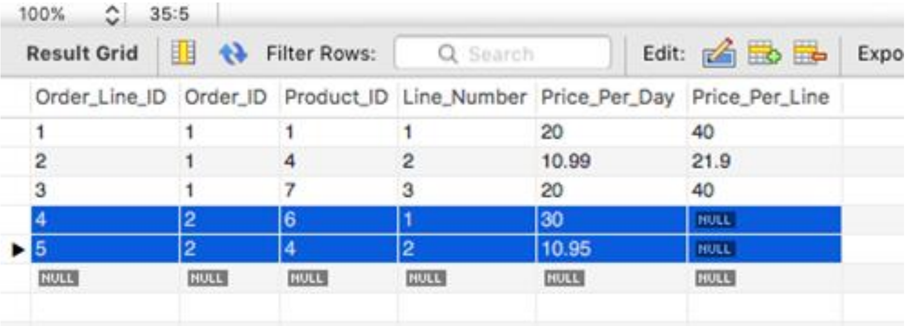
Order_Line_ID	Order_ID	Product_ID	Line_Number	Price_Per_Day	Price_Per_Line
1	1	1	1	20	40
2	1	4	2	10.99	21.9
3	1	7	3	20	40
NULL	NULL	NULL	NULL	NULL	NULL

b) Car table before adding any records into Order_line.



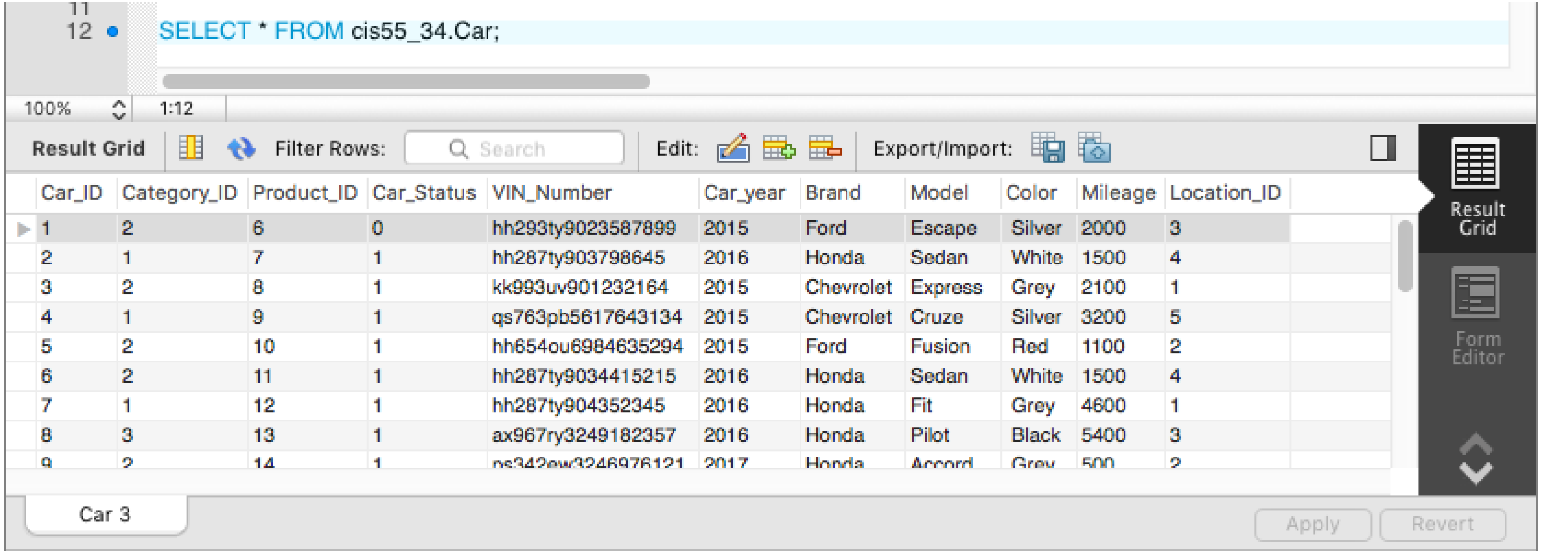
Car_ID	Category_ID	Product_ID	Car_Status	VIN_Number	Car_year	Brand	Model	Color	Mileage	Location_ID
1	2	6	1	hh293ty9023587899	2015	Ford	Escape	Silver	2000	3
2	1	7	1	hh287ty903798645	2016	Honda	Sedan	White	1500	4
3	2	8	1	kk993uv901232164	2015	Chevrolet	Express	Grey	2100	1
4	1	9	1	qs763pb5617643134	2015	Chevrolet	Cruze	Silver	3200	5
5	2	10	1	hh654ou6984635294	2015	Ford	Fusion	Red	1100	2
6	2	11	1	hh287ty9034415215	2016	Honda	Sedan	White	1500	4
7	1	12	1	hh287ty904352345	2016	Honda	Fit	Grey	4600	1
8	3	13	1	ax967ry3249182357	2016	Honda	Pilot	Black	5400	3
9	2	14	1	ps342ew3246976121	2017	Honda	Accord	Grey	500	2

c) Order_line table after adding records with Order_line_IDs 4 and 5.



Order_Line_ID	Order_ID	Product_ID	Line_Number	Price_Per_Day	Price_Per_Line
1	1	1	1	20	40
2	1	4	2	10.99	21.9
3	1	7	3	20	40
4	2	6	1	30	NULL
5	2	4	2	10.95	NULL
NULL	NULL	NULL	NULL	NULL	NULL

d) Car table immediately afterwards. Trigger has been implemented on the car with Product_ID 6.



```
SELECT * FROM cis55_34.Car;
```

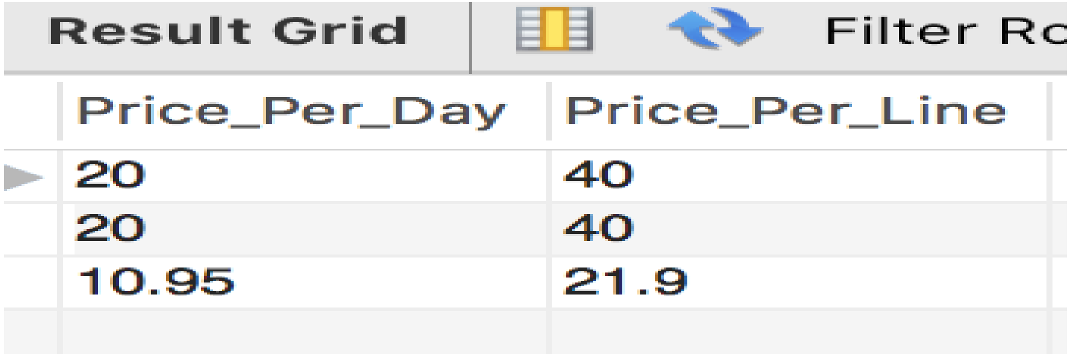
Car_ID	Category_ID	Product_ID	Car_Status	VIN_Number	Car_year	Brand	Model	Color	Mileage	Location_ID
1	2	6	0	hh293ty9023587899	2015	Ford	Escape	Silver	2000	3
2	1	7	1	hh287ty903798645	2016	Honda	Sedan	White	1500	4
3	2	8	1	kk993uv901232164	2015	Chevrolet	Express	Grey	2100	1
4	1	9	1	qs763pb5617643134	2015	Chevrolet	Cruze	Silver	3200	5
5	2	10	1	hh654ou6984635294	2015	Ford	Fusion	Red	1100	2
6	2	11	1	hh287ty9034415215	2016	Honda	Sedan	White	1500	4
7	1	12	1	hh287ty904352345	2016	Honda	Fit	Grey	4600	1
8	3	13	1	ax967ry3249182357	2016	Honda	Pilot	Black	5400	3
9	2	14	1	ps342ew3246976121	2017	Honda	Accord	Grey	500	2

Continued Stored Procedure/ Function/Trigger/View

viii) After customer makes order, aggregate price_per_day and calculate price_per_line from unit prices

When we use procedure to create new Order_line, we can just use select statement to get the price_per_day and price_per_line

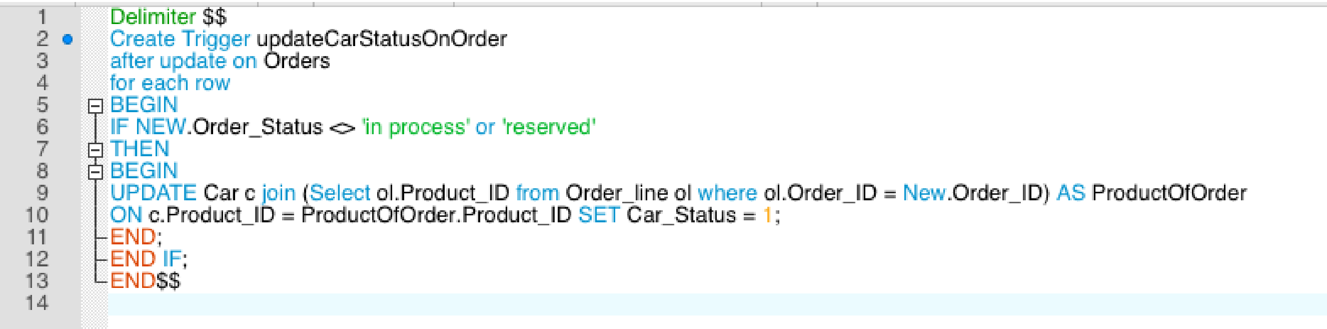
Select Price_Per_Day, Price_Per_Line from Order_line where Order_ID=2;



Price_Per_Day	Price_Per_Line
20	40
20	40
10.95	21.9

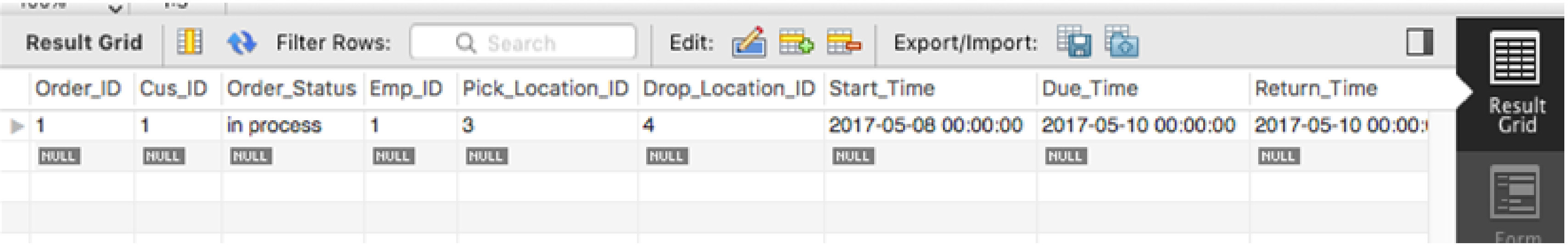
Continued Stored Procedure/ Function/Trigger/View

ix) If order_status changes from in progress or reserved to complete or canceled, car_status must be changed from unavailable to available. (from 0 to 1)



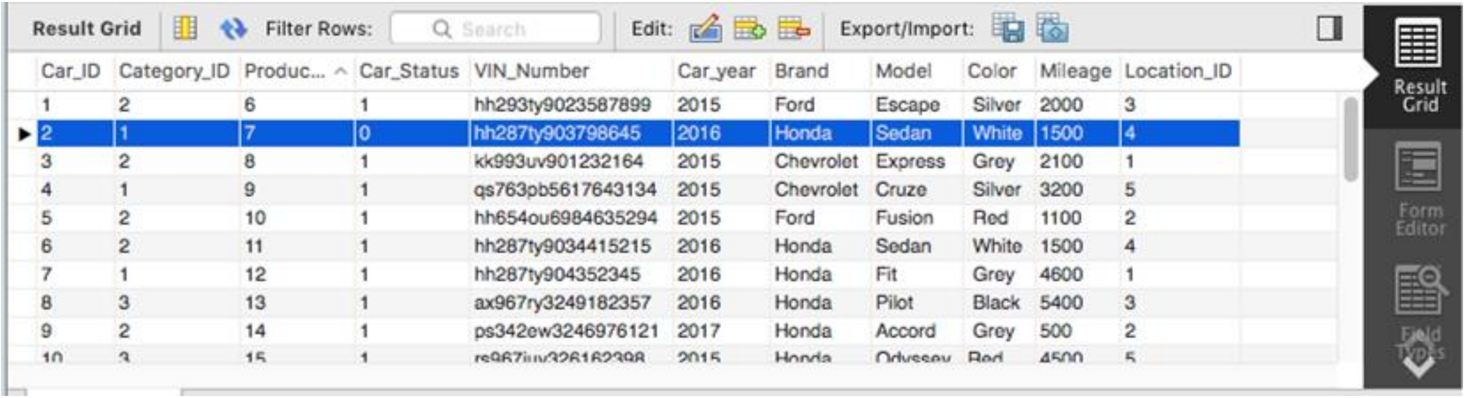
```
Delimiter $$
Create Trigger updateCarStatusOnOrder
after update on Orders
for each row
BEGIN
IF NEW.Order_Status <> 'in process' or 'reserved'
THEN
BEGIN
UPDATE Car c join (Select ol.Product_ID from Order_line ol where ol.Order_ID = New.Order_ID) AS ProductOfOrder
ON c.Product_ID = ProductOfOrder.Product_ID SET Car_Status = 1;
END;
END IF;
END$$
```

a) order 1 starts as “in process”.



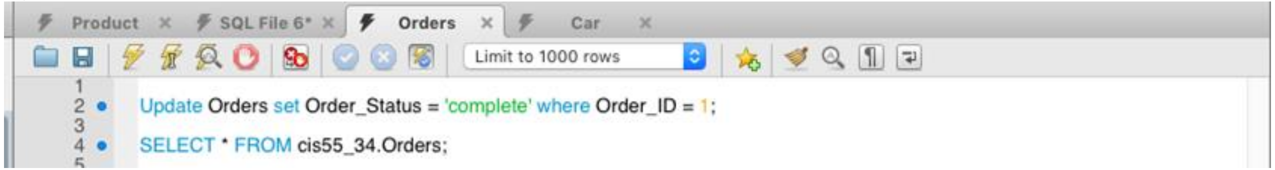
Order_ID	Cus_ID	Order_Status	Emp_ID	Pick_Location_ID	Drop_Location_ID	Start_Time	Due_Time	Return_Time
1	1	in process	1	3	4	2017-05-08 00:00:00	2017-05-10 00:00:00	2017-05-10 00:00:
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

b) corresponding car table: The Car_status of car 7 starts out as 0 or unavailable.

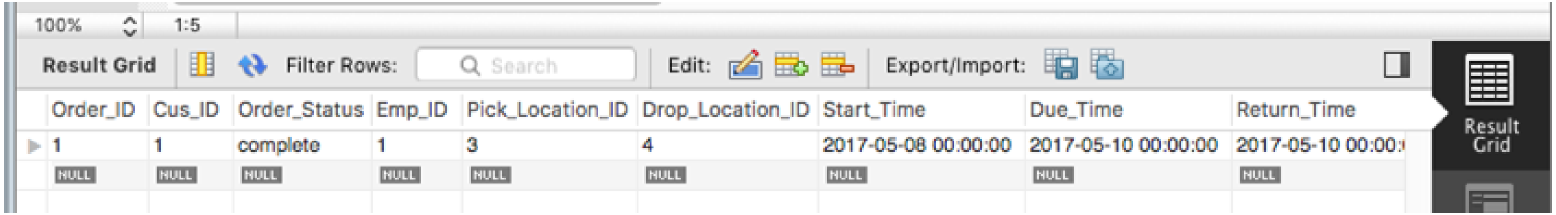


Car_ID	Category_ID	Produc...	Car_Status	VIN_Number	Car_year	Brand	Model	Color	Mileage	Location_ID
1	2	6	1	hh293ty9023587899	2015	Ford	Escape	Silver	2000	3
2	1	7	0	hh287ty903798645	2016	Honda	Sedan	White	1500	4
3	2	8	1	kk993uv901232164	2015	Chevrolet	Express	Grey	2100	1
4	1	9	1	qs763pb5617643134	2015	Chevrolet	Cruze	Silver	3200	5
5	2	10	1	hh654ou6984635294	2015	Ford	Fusion	Red	1100	2
6	2	11	1	hh287ty9034415215	2016	Honda	Sedan	White	1500	4
7	1	12	1	hh287ty904352345	2016	Honda	Fit	Grey	4600	1
8	3	13	1	ax967ry3249182357	2016	Honda	Pilot	Black	5400	3
9	2	14	1	ps342ew3246976121	2017	Honda	Accord	Grey	500	2
10	3	15	1	rs967iuv326162398	2015	Honda	Odyssey	Red	4500	5

c) Order_Status of the order with an Order_ID of 1 is set to complete.

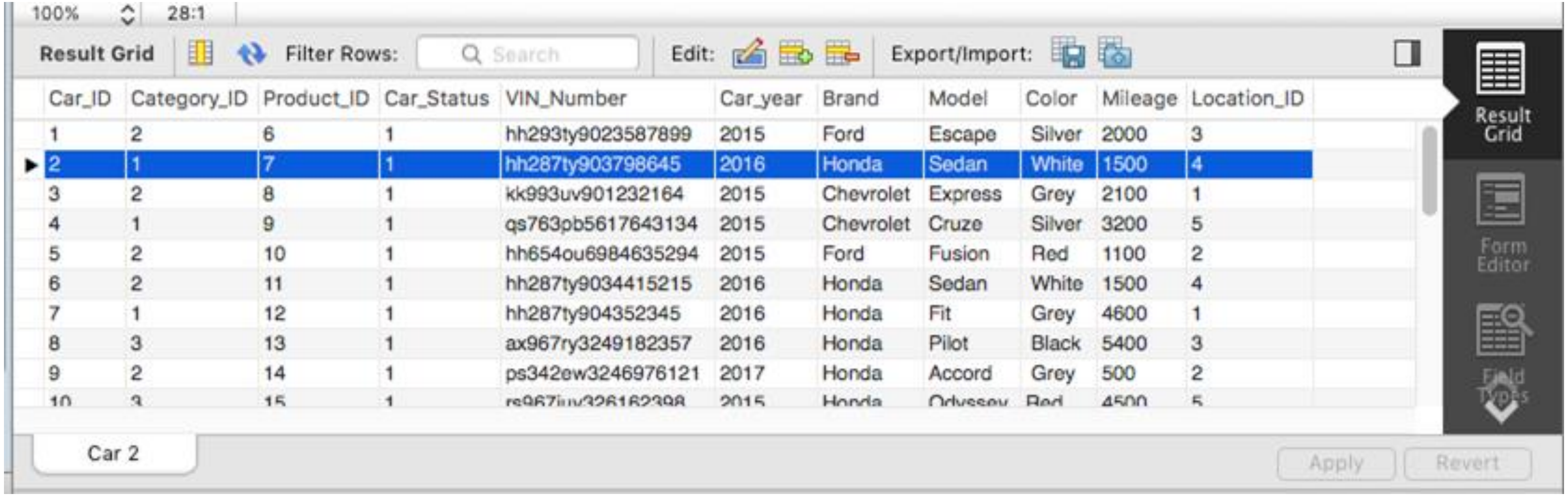


```

Update Orders set Order_Status = 'complete' where Order_ID = 1;

SELECT * FROM cis55_34.Orders;
```



Order_ID	Cus_ID	Order_Status	Emp_ID	Pick_Location_ID	Drop_Location_ID	Start_Time	Due_Time	Return_Time
1	1	complete	1	3	4	2017-05-08 00:00:00	2017-05-10 00:00:00	2017-05-10 00:00:
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

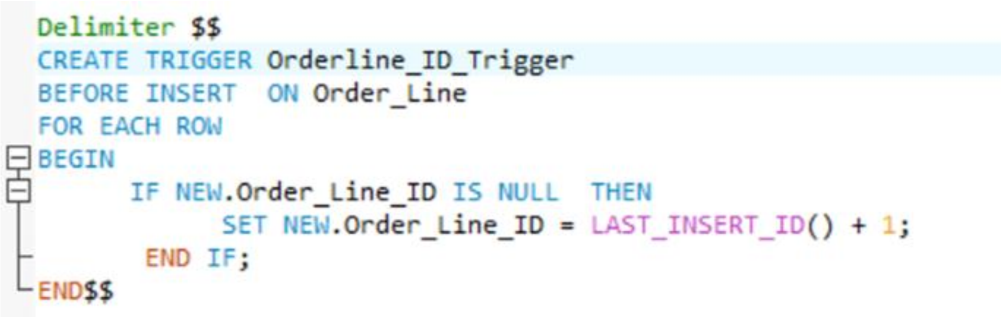
d) As a result of the trigger, in the table below, the Car_Status of the same car, with Product_ID 7, is automatically set to 1 or available.



Car_ID	Category_ID	Product_ID	Car_Status	VIN_Number	Car_year	Brand	Model	Color	Mileage	Location_ID
1	2	6	1	hh293ty9023587899	2015	Ford	Escape	Silver	2000	3
2	1	7	1	hh287ty903798645	2016	Honda	Sedan	White	1500	4
3	2	8	1	kk993uv901232164	2015	Chevrolet	Express	Grey	2100	1
4	1	9	1	qs763pb5617643134	2015	Chevrolet	Cruze	Silver	3200	5
5	2	10	1	hh654ou6984635294	2015	Ford	Fusion	Red	1100	2
6	2	11	1	hh287ty9034415215	2016	Honda	Sedan	White	1500	4
7	1	12	1	hh287ty904352345	2016	Honda	Fit	Grey	4600	1
8	3	13	1	ax967ry3249182357	2016	Honda	Pilot	Black	5400	3
9	2	14	1	ps342ew3246976121	2017	Honda	Accord	Grey	500	2
10	3	15	1	rs967iuv326162398	2015	Honda	Odyssey	Red	4500	5

Continued Stored Procedure/ Function/Trigger/View

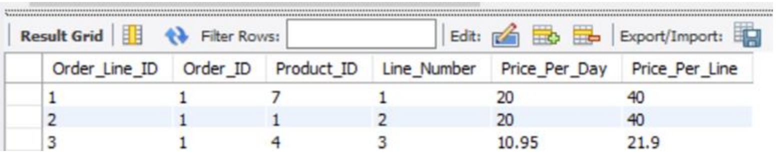
x) Order_line table: automatically generate unique entity id



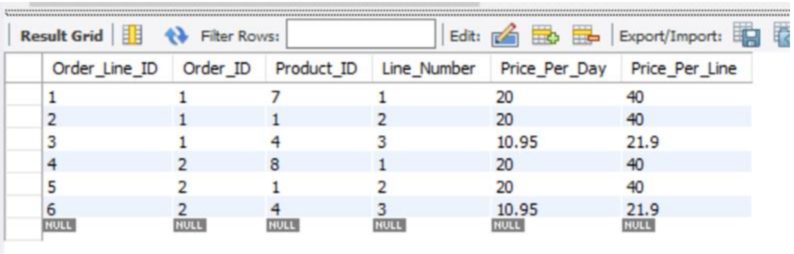
```
Delimiter $$
CREATE TRIGGER Orderline_ID_Trigger
BEFORE INSERT  ON Order_Line
FOR EACH ROW
BEGIN
      IF NEW.Order_Line_ID IS NULL  THEN
            SET NEW.Order_Line_ID = LAST_INSERT_ID() + 1;
       END IF;
END$$
```

b) After Insert

a) Before Insert



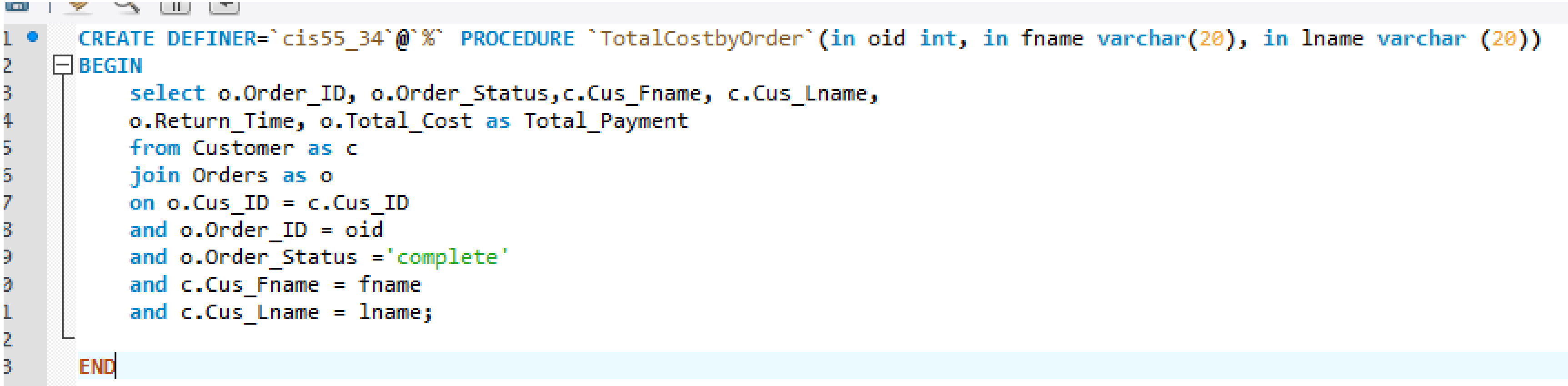
Order_Line_ID	Order_ID	Product_ID	Line_Number	Price_Per_Day	Price_Per_Line
1	1	7	1	20	40
2	1	1	2	20	40
3	1	4	3	10.95	21.9



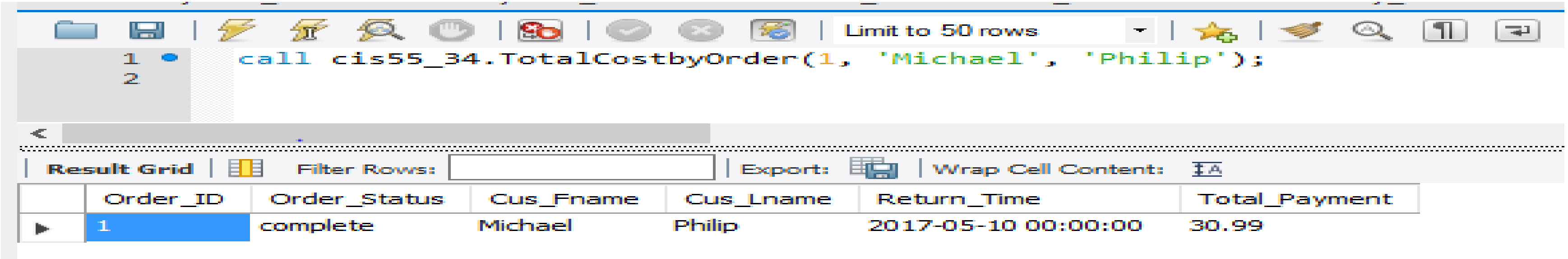
Order_Line_ID	Order_ID	Product_ID	Line_Number	Price_Per_Day	Price_Per_Line
1	1	7	1	20	40
2	1	1	2	20	40
3	1	4	3	10.95	21.9
4	2	8	1	20	40
5	2	1	2	20	40
6	2	4	3	10.95	21.9
NULL	NULL	NULL	NULL	NULL	NULL

Continued Stored Procedure/ Function/Trigger/View

xi) The Total Cost owed by a specific customer when he returns the car.



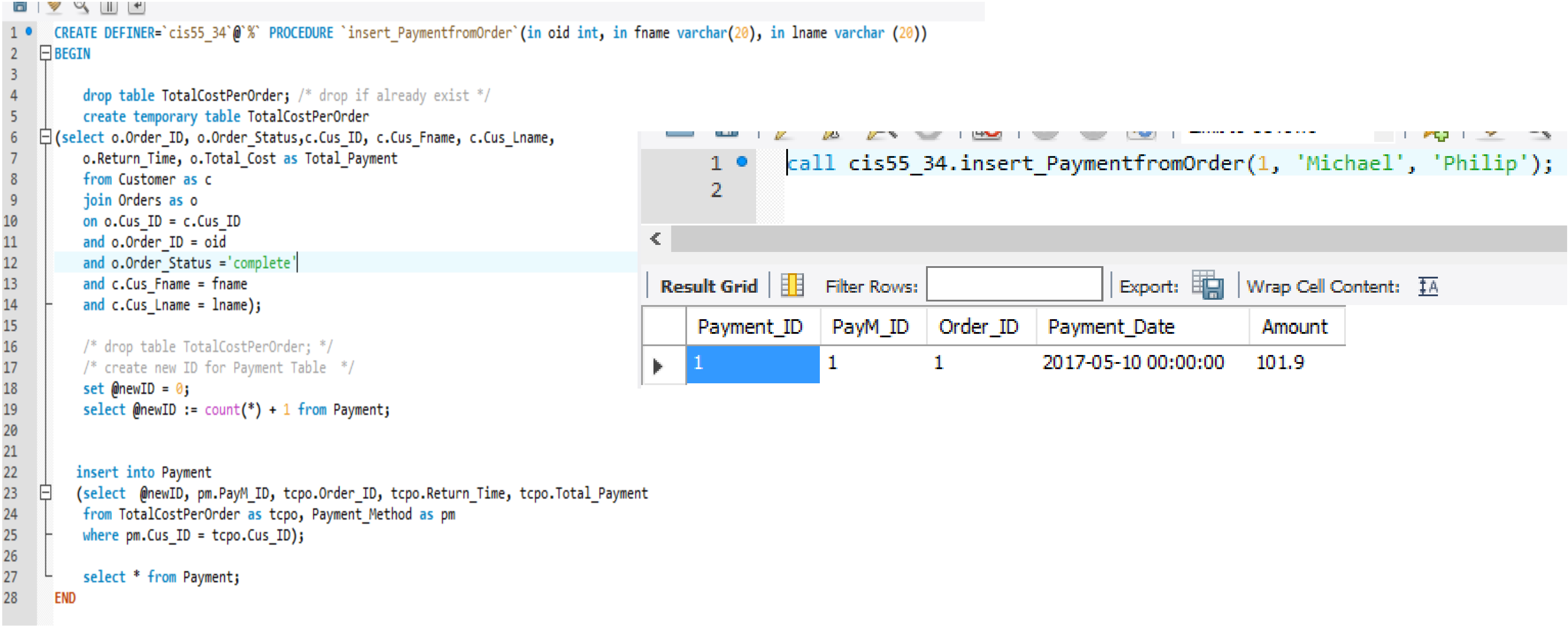
```
CREATE DEFINER=`cis55_34`@`%` PROCEDURE `TotalCostbyOrder`(in oid int, in fname varchar(20), in lname varchar (20))
BEGIN
    select o.Order_ID, o.Order_Status,c.Cus_Fname, c.Cus_Lname,
    o.Return_Time, o.Total_Cost as Total_Payment
    from Customer as c
    join Orders as o
    on o.Cus_ID = c.Cus_ID
    and o.Order_ID = oid
    and o.Order_Status ='complete'
    and c.Cus_Fname = fname
    and c.Cus_Lname = lname;

END
```



```
call cis55_34.TotalCostbyOrder(1, 'Michael', 'Philip');
```

Order_ID	Order_Status	Cus_Fname	Cus_Lname	Return_Time	Total_Payment
1	complete	Michael	Philip	2017-05-10 00:00:00	30.99

Alt: generate total payment by inserting data into Payment table



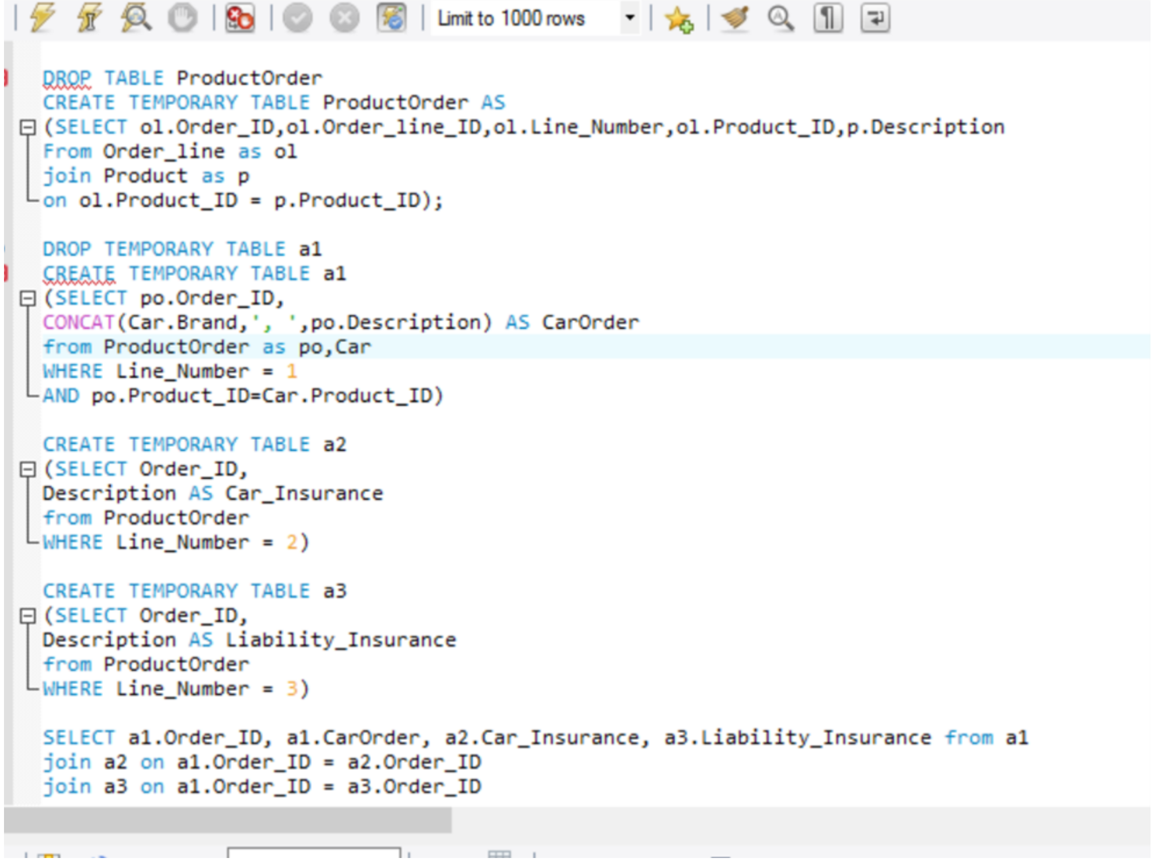
```
CREATE DEFINER=`cis55_34`@`%` PROCEDURE `insert_PaymentfromOrder`(in oid int, in fname varchar(20), in lname varchar (20))
BEGIN

    drop table TotalCostPerOrder; /* drop if already exist */
    create temporary table TotalCostPerOrder
(select o.Order_ID, o.Order_Status,c.Cus_ID, c.Cus_Fname, c.Cus_Lname,
    o.Return_Time, o.Total_Cost as Total_Payment
    from Customer as c
    join Orders as o
    on o.Cus_ID = c.Cus_ID
    and o.Order_ID = oid
    and o.Order_Status ='complete'
    and c.Cus_Fname = fname
    and c.Cus_Lname = lname);

    /* drop table TotalCostPerOrder; */
    /* create new ID for Payment Table  */
    set @newID = 0;
    select @newID := count(*) + 1 from Payment;


   insert into Payment
   (select  @newID, pm.PayM_ID, tcpo.Order_ID, tcpo.Return_Time, tcpo.Total_Payment
    from TotalCostPerOrder as tcpo, Payment_Method as pm
    where pm.Cus_ID = tcpo.Cus_ID);

    select * from Payment;
END
```

```
call cis55_34.insert_PaymentfromOrder(1, 'Michael', 'Philip');
```

Payment_ID	PayM_ID	Order_ID	Payment_Date	Amount
1	1	1	2017-05-10 00:00:00	101.9

Continued Stored Procedure/ Function/Trigger/View

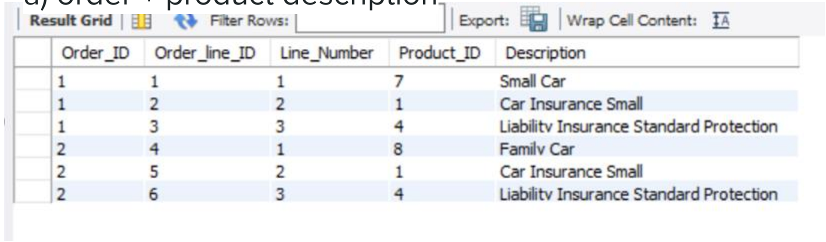
xii) Display all information for one order (orderline, order, car, the 2 insurances



```
DROP TABLE ProductOrder
CREATE TEMPORARY TABLE ProductOrder AS
(SELECT ol.Order_ID,ol.Order_line_ID,ol.Line_Number,ol.Product_ID,p.Description
From Order_line as ol
join Product as p
on ol.Product_ID = p.Product_ID);

DROP TEMPORARY TABLE a1
CREATE TEMPORARY TABLE a1
(SELECT po.Order_ID,
CONCAT(Car.Brand,', ',po.Description) AS CarOrder
from ProductOrder as po,Car
WHERE Line_Number = 1
AND po.Product_ID=Car.Product_ID)

CREATE TEMPORARY TABLE a2
(SELECT Order_ID,
Description AS Car_Insurance
from ProductOrder
WHERE Line_Number = 2)

CREATE TEMPORARY TABLE a3
(SELECT Order_ID,
Description AS Liability_Insurance
from ProductOrder
WHERE Line_Number = 3)

SELECT a1.Order_ID, a1.CarOrder, a2.Car_Insurance, a3.Liability_Insurance from a1
join a2 on a1.Order_ID = a2.Order_ID
join a3 on a1.Order_ID = a3.Order_ID
```

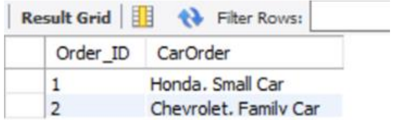
Alt: generate total information of orders

a) order + product description



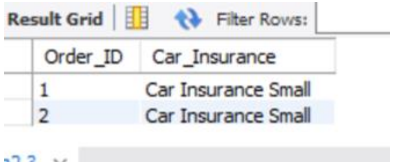
Order_ID	Order_line_ID	Line_Number	Product_ID	Description
1	1	1	7	Small Car
1	2	2	1	Car Insurance Small
1	3	3	4	Liabilitv Insurance Standard Protection
2	4	1	8	Familv Car
2	5	2	1	Car Insurance Small
2	6	3	4	Liabilitv Insurance Standard Protection

b) order + car (According to Line_Number 1)



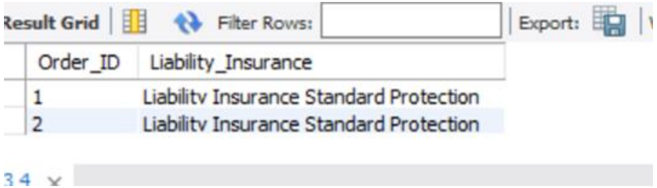
Order_ID	CarOrder
1	Honda. Small Car
2	Chevrolet. Familv Car

c) order + car_Insu (According to Line_Number 2)



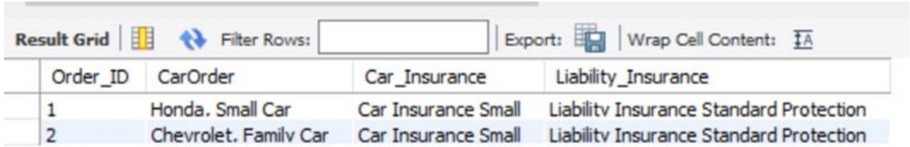
Order_ID	Car_Insurance
1	Car Insurance Small
2	Car Insurance Small

d) order + Lia_Insu (According to Line_Number 3)



Order_ID	Liability_Insurance
1	Liabilitv Insurance Standard Protection
2	Liabilitv Insurance Standard Protection

e) result



Order_ID	CarOrder	Car_Insurance	Liability_Insurance
1	Honda. Small Car	Car Insurance Small	Liabilitv Insurance Standard Protection
2	Chevrolet. Familv Car	Car Insurance Small	Liabilitv Insurance Standard Protection

7.Conclusion

This project has taught us:

- How to implement step by step the database development life cycle.
- How to write a business plan
- Entities
- Entity Relationships
- ER Diagrams
- Sql statements
- Stored procedures
- Triggers