Rent*A*Car
Database Design

Shaily Sahay (ss4596@njit.edu)
New Jersey Institute of Technology
Course Number: CS-631

Table of Contents

Objectives Presented **4**

Business Rules **4**

Additional Design **5**

Assumptions **6**

ER Diagram **6**

Relational Logical Database Design **7**

Normalized Relations **7**

- CUSTOMER 7
  - CUSTOMER 9
  - CREDIT_CARD 10
- MODEL 11
- CLASS 12
- LOCATION 13
- VEHICLE 14
- RESERVATION 15
  - RESERVATION 16
  - RENTAL_RATE 17
- RENTAL_AGREEMENT 18
- BILLING 19

Sample SQL Queries **19**

Objectives Presented

- Business Rules
- Additional Design
- Assumptions
- ER Diagram
- Relational Logical Database Design
- Normalized Relations
- Sample SQL Queries
- Conclusion

Business Rules

A car rental company - 'RentACar' - wishes to implement a database to control all aspects of its operations, including tracking car inventories, rental contracts, and billing. The following statements of business rules and relationships are used to construct a relational model:

Cars and Inventory:

1. The car model includes a make (Ford, Honda, etc.), the year of the model, and the model name.
2. Each car is uniquely identified by a vehicle identification number (VIN)
3. Cars are assigned to locations, and each location has one or more cars
4. The branch to which the vehicle is assigned has an address and a location ID

Car Reservation:

1. The process of renting a car is as follows:
    a. A customer first makes a reservation by telephone for the pickup of a particular class of car at a specific location.
    b. After making the reservation, the customer arrives at the branch location to pick up the car.
    c. The 'RentACar' service representative takes the customer's name and address and the class of a vehicle and the period of rental (date and time in and out) that the customer desires. The customer is informed of the rental rate.
2. Car rental rates are determined by the class of the car. 'RentACar' has two rental rates for each class: daily and weekly.
3. The same customer may make more than one reservation over time.

Rental Agreement:

1. A reservation generally results in a rental agreement, which is established when the customer comes to the location to pick up the car. However, this is not always the case, since a reservation may be canceled or the customer with a reservation may not show up.
2. If and when the customer arrives at the branch location to pick up the car, the service representative first checks for a reservation and, if a reservation exists, they draw up a rental agreement.
3. At that time the service representative obtains other customer information, such as his operator's license number and the state that issued it, and the customer's credit card type and number, including the expiration month and year.
4. A specific vehicle is then assigned to the customer for this rental agreement. At any point in time, a specific vehicle may have participated in zero, one, or more rental agreements.
5. If the customer is a walkin (no reservation), the service representative fills out the reservation information first as part of the process.
6. The rental agreement has a contract number that uniquely identifies it, the VIN number of the vehicle that is being rented, the current date and time for the rental to start, and a current odometer reading.
7. The customer is given a copy of the rental agreement along with the keys to the car. This ends the activities at the time the vehicle is picked up.

Billing and Car return

1. After use, the car is returned to the branch location.
2. Information that will be filled in when the car is returned is the 'date and time' at which the rental ends and the editing 'odometer reading'.
3. When the rental agreement is completed, the actual cost of the rental is computed using the class rental rate, and the cost is charged to the customer's credit card. No other form of payment is accepted.

Additional Design

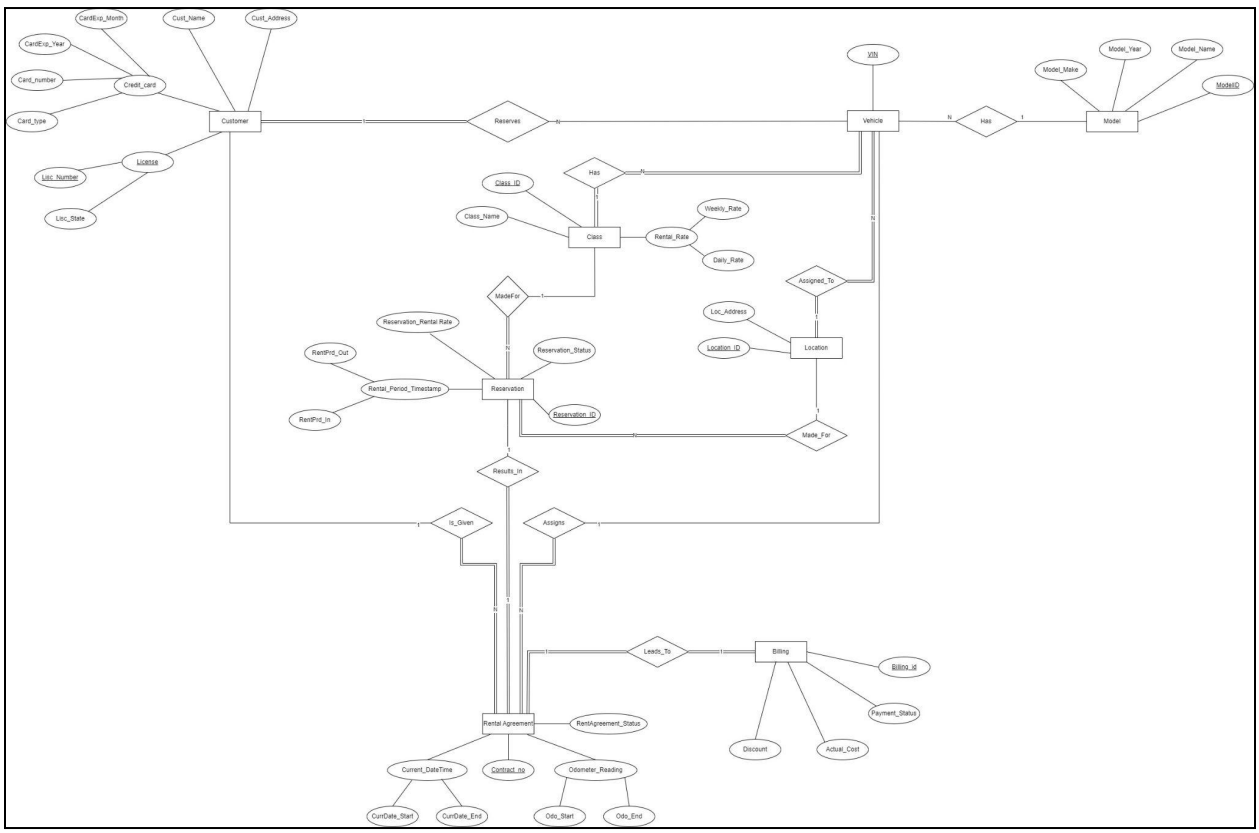
- A new relation is added - 'Billing', which tracks the billing details.
- For every rental contract that is completed, a record is created in the 'Billing' relation
- This relation calculates the actual cost of the rental contract, and applies any discount, if applicable

- It also maintains the payment status of the rental contract through 2 status values - ‘Pending’, ‘Complete’.

Assumptions

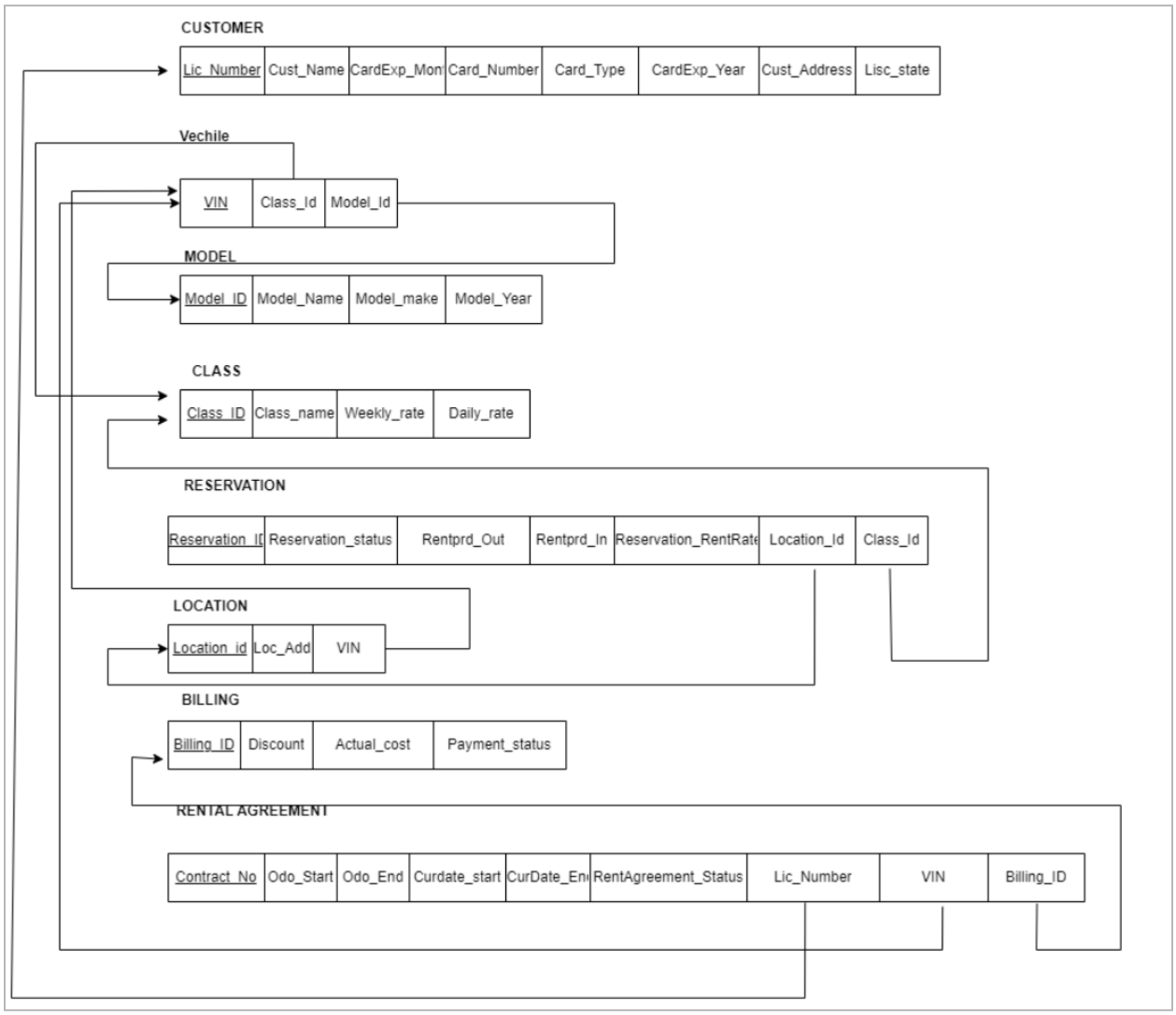
1. Each booking is associated with only one car reservation at a time
2. Car available in the system should be present at some location
3. Billing may or may not have discount code applied
4. Not all Reservations is associated with billing because of the canceled reservations

ER Diagram



(For better readability, a copy of the above ER diagram is included in the project folder)

Relational Logical Database Design



Normalized Relations

CUSTOMER

Lisc_Number	L_State	Cust_Name	Cust_Addr	CardNumber	CardType	CardExp_Month	CardExp_Year

PK

Sample Data

Worksheet | Query Builder

```
SELECT * FROM CUSTOMER
```

Query Result

SQL | All Rows Fetched: 11 in 0.027 seconds

	LISC_NUMBER	L_STATE	CUST_NAME	CUST_ADDR	CARDNUMBER	CARDTYPE	CARDEXP_MONTH	CARDEXP_YEAR
1	Lisc000012345678	Ohio	John Doe	Ohio Address 1	0000111122223333	Visa	9	2024
2	Lisc000112345678	New Jersey	Smith Jones	NJ Address 1	1111222233334444	Visa	11	2024
3	Lisc000212345678	New Jersey	Ravi Kumar	NJ Address 2	2222333344445555	MasterCard	1	2023
4	Lisc000312345678	Texas	Chris Harris	Texas Address 1	3333444455556666	Discover	5	2024
5	Lisc000412345678	Ohio	Clint East	Ohio Address 2	4444555566667777	Discover	2	2025
6	Lisc000512345678	Ohio	Mark Johnson	Ohio Address 3	5555666677778888	MasterCard	3	2025
7	Lisc000612345678	Texas	John Markson	Texas Address 2	6666777788889999	Discover	4	2024
8	Lisc000712345678	New York	Manish Singh	NY Address 1	7777888899990000	Visa	2	2025
9	Lisc000812345678	New Jersey	Neeraj Patel	New Jersey Address 3	4444555566661111	Discover	7	2026
10	Lisc000912345678	New York	Sarah Parker	NY Address 2	7777555566667777	Visa	6	2024
11	Lisc001012345678	Ohio	Ben Friedman	NY Address 3	8888555566667777	MasterCard	10	2025

Functional Dependencies

- {Lisc_Number} → {L_State, Cust_Name, Cust_Addr, CardNumber}
- {CardNumber} → {Cust_Name, CardType, CardExp_Month, CardExp_Year}

Key

{Lisc_Number}

Normalization

The table is not in 3NF due to the following transitive dependency:

- {CardNumber} → {Cust_Name, CardType, CardExp_Month, CardExp_Year}

3NF relations

CUSTOMER (<u>Lisc_Number</u>, L_State, Cust_Name, Cust_Addr, CardNum)

CREDIT_CARD (<u>CardNumber</u>, Cust_Name, CardType, CardExp_Month, CardExp_Year)

CUSTOMER

Lisc_Number	L_State	Cust_Name	Cust_Addr	CardNum

PK

Sample Data

```
SELECT * FROM CUSTOMER
/
```

Script Output × Query Result ×

SQL | All Rows Fetched: 11 in 0.031 seconds

	LISC_NUMBER	L_STATE	CUST_NAME	CUST_ADDR	CARD_NUM
1	Lisc000012345678	Ohio	John Doe	Ohio Address 1	0000111122223333
2	Lisc000112345678	New Jersey	Smith Jones	NJ Address 1	1111222233334444
3	Lisc000212345678	New Jersey	Ravi Kumar	NJ Address 2	2222333344445555
4	Lisc000312345678	Texas	Chris Harris	Texas Address 1	3333444455556666
5	Lisc000412345678	Ohio	Clint East	Ohio Address 2	4444555566667777
6	Lisc000512345678	Ohio	Mark Johnson	Ohio Address 3	5555666677778888
7	Lisc000612345678	Texas	John Markson	Texas Address 2	6666777788889999
8	Lisc000712345678	New York	Manish Singh	NY Address 1	7777888899990000
9	Lisc000812345678	New Jersey	Neeraj Patel	New Jersey Address 3	4444555566661111
10	Lisc000912345678	New York	Sarah Parker	NY Address 2	7777555566667777
11	Lisc001012345678	Ohio	Ben Friedman	NY Address 3	8888555566667777

Functional Dependencies

- {Lisc_Number} → {L_State, Cust_Name, Cust_Addr, CardNumber}

Key

{Lisc_Number}

Normalization

The table is already in 3NF due to lack of partial-key dependency and transitive dependency.

CREDIT_CARD

CardNumber	Cust_Name	CardType	CardExp_Month	CardExp_Year

PK

Sample Data

```
SELECT * FROM CREDIT_CARD
/
```

Script Output x Query Result x

SQL | All Rows Fetched: 11 in 0.025 seconds

	CARDNUMBER	CARDTYPE	CARDEXP_MONTH	CARDEXP_YEAR	CUST_NAME
1	0000111122223333	Visa	9	2024	John Doe
2	1111222233334444	Visa	11	2024	Smith Jones
3	2222333344445555	MasterCard	1	2023	Ravi Kumar
4	3333444455556666	Discover	5	2024	Chris Harris
5	4444555566667777	Discover	2	2025	Clint East
6	5555666677778888	MasterCard	3	2025	Mark Johnson
7	6666777788889999	Discover	4	2024	John Markson
8	7777888899990000	Visa	2	2025	Manish Singh
9	4444555566661111	Discover	7	2026	Neeraj Patel
10	7777555566667777	Visa	6	2024	Sarah Parker
11	8888555566667777	MasterCard	10	2025	Ben Friedman

Functional Dependencies

- {CardNumber} → {Cust_Name, CardType, CardExp_Month, CardExp_Year}

Key

{CardNumber}

Normalization

The table is already in 3NF due to lack of partial-key dependency and transitive dependency.

MODEL

ModelID	Model_Make	Model_Name	Model_Year

PK

Sample Data

Worksheet | Query Builder

```
SELECT * FROM MODEL
```

Query Result

SQL | All Rows Fetched: 10 in 0.025 seconds

	MODELID	MODEL_MAKE	MODEL_NAME	MODEL_YEAR
1	Mod1	Hyundai	Sonata	2013
2	Mod2	Hyundai	Eon	2014
3	Mod4	Chevrolet	Optra	2022
4	Mod5	Mercedes	Benz	2019
5	Mod6	Honda	Accord	2020
6	Mod3	Honda	Civic	2019
7	Mod7	Honda	Mustang	2021
8	Mod8	Hyundai	Accent	2019
9	Mod9	Chevrolet	Malibu	2019
10	Mod10	Chevrolet	Beat	2020

Functional Dependencies

- {ModelID} → {Model_Make, Model_Name, Model_Year}

Key

{ModelID}

Normalization

The table is already in 3NF due to lack of partial-key dependency and transitive dependency.

CLASS

Class_ID	Class_Name	WeeklyRate	DailyRate	Model_ID
PK				FK

Sample Data

Worksheet | Query Builder

```
SELECT * FROM CLASS
```

Query Result

SQL | All Rows Fetched: 10 in 0.026 seconds

	CLASS_ID	CLASS_NAME	WEEKLYRATE	DAILYRATE	MODEL_ID
1	Cls1	ClassA	430	70	Mod1
2	Cls2	ClassB	350	65	Mod2
3	Cls3	ClassC	550	80	Mod3
4	Cls4	ClassD	450	75	Mod5
5	Cls5	ClassE	375	60	Mod4
6	Cls6	ClassF	470	78	Mod6
7	Cls7	ClassG	370	68	Mod7
8	Cls8	ClassH	360	66	Mod8
9	Cls9	ClassI	560	82	Mod9
10	Cls10	ClassJ	440	72	Mod10

Functional Dependencies

- {Class_ID} → Class_Name, WeeklyRate, DailyRate, Model_ID

Key

{Class_ID}

Normalization

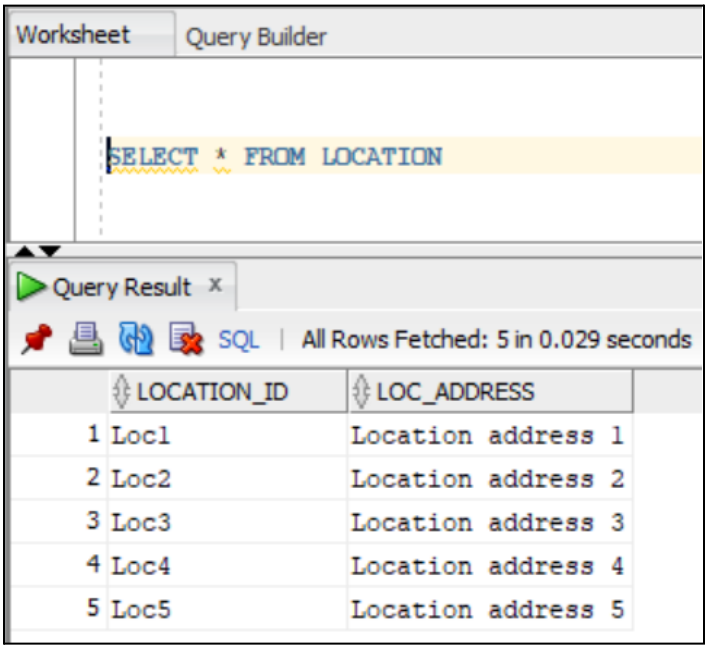
The table is already in 3NF due to lack of partial-key dependency and transitive dependency.

LOCATION

Location_ID	Loc_Address

PK

Sample Data



Functional Dependencies

- {Location_ID} → {Loc_Address}

Key

{Location_ID}

Normalization

The table is already in 3NF due to lack of partial-key dependency and transitive dependency.

VEHICLE

VIN	Cls_ID	Loc_ID	Lisc_Num
PK	FK	FK	FK

Sample Data

Worksheet | Query Builder

```
SELECT * FROM VEHICLE
```

Query Result

SQL | All Rows Fetched: 7 in 0.033 seconds

	VIN	LOC_ID	CLS_ID	LISC_NUM
1	VIN01	Loc1	Cls1	Lisc000012345678
2	VIN02	Loc1	Cls3	Lisc000312345678
3	VIN03	Loc2	Cls1	Lisc000412345678
4	VIN04	Loc2	Cls4	Lisc001012345678
5	VIN05	Loc3	Cls5	Lisc000912345678
6	VIN06	Loc4	Cls6	Lisc000712345678
7	VIN07	Loc5	Cls7	Lisc000612345678

Functional Dependencies

- {VIN} → {Cls_ID, Loc_ID, Lisc_Num}

Key

{VIN}

Normalization

The table is already in 3NF due to lack of partial-key dependency and transitive dependency.

RESERVATION

Reservation_ID	Reservation _Status	Reservation_RentalRate	RentPrd_In	RentPrd_Out	VehClass_Id	Loc_ID
PK					FK	FK

Sample Data

Worksheet | Query Builder

```
SELECT * FROM RESERVATION
```

Query Result

SQL | All Rows Fetched: 9 in 0.048 seconds

	RESERVATION_ID	RESERVATION_STATUS	RESERVATION_RENTALRATE	RENTPRD_IN	RENTPRD_OUT	VEHCLASS_ID	LOC_ID
1	Res1	Active	140	01-NOV-22 10.02.05.000000000 AM	03-NOV-22 10.02.05.000000000 AM	Cls1	Loc1
2	Res2	Active	210	01-DEC-22 10.00.00.000000000 AM	08-DEC-22 10.00.00.000000000 AM	Cls1	Loc1
3	Res3	Active	550	15-NOV-22 10.00.00.000000000 AM	22-NOV-22 10.00.00.000000000 AM	Cls3	Loc2
4	Res4	Cancelled	440	01-JAN-23 04.20.00.000000000 AM	08-JAN-23 12.20.00.000000000 PM	Cls2	Loc3
5	Res5	Active	120	13-NOV-22 08.00.00.000000000 AM	15-NOV-22 08.30.00.000000000 AM	Cls5	Loc4
6	Res6	Complete	440	01-SEP-22 05.30.00.000000000 PM	08-SEP-22 09.00.00.000000000 AM	Cls10	Loc2
7	Res7	Active	150	08-NOV-22 01.30.00.000000000 PM	10-NOV-22 06.00.00.000000000 PM	Cls4	Loc3
8	Res8	Active	150	01-NOV-22 10.00.00.000000000 AM	03-NOV-22 06.00.00.000000000 PM	Cls4	Loc1
9	Res9	Active	370	15-NOV-22 08.00.00.000000000 PM	21-NOV-22 10.00.00.000000000 PM	Cls7	Loc1

Functional Dependencies

- {Reservation_ID} → Reservation_Status, RentPrd_In, RentPrd_Out, VehClass_Id, Loc_ID}
- {RentPrd_In, RentPrd_Out,VehClass_Id} → Reservation_RentalRate

Key

{Reservation_ID}

Normalization

The table is not in 3NF due to the following transitive dependency:

- {RentPrd_In, RentPrd_Out,VehClass_Id} → Reservation_RentalRate

3NF relations

RESERVATION (Reservation_ID, Reservation_Status, RentPrd_In, RentPrd_Out, VehClass_Id, Loc_ID)

RENTAL_RATE (RentPrd_In, RentPrd_Out, VehClass_Id, Reservation_RentalRate)

RESERVATION

Reservation_ID	Reservation_Status	RentPrd_In	RentPrd_Out	VehClass_Id	Loc_ID

PK

Sample Data

```
SELECT * FROM RESERVATION
/
```

Script Output × Query Result ×

SQL | All Rows Fetched: 9 in 0.027 seconds

	RESERVATION_ID	RESERVATION_STATUS	RENTPRD_IN	RENTPRD_OUT	VEHCLASS_ID	LOC_ID
1	Res1	Active	01-NOV-22 10.02.05.000000000 AM	03-NOV-22 10.02.05.000000000 AM	Cls1	Loc1
2	Res2	Active	01-DEC-22 10.00.00.000000000 AM	08-DEC-22 10.00.00.000000000 AM	Cls1	Loc1
3	Res3	Active	15-NOV-22 10.00.00.000000000 AM	22-NOV-22 10.00.00.000000000 AM	Cls3	Loc2
4	Res4	Cancelled	01-JAN-23 04.20.00.000000000 AM	08-JAN-23 12.20.00.000000000 PM	Cls2	Loc3
5	Res5	Active	13-NOV-22 08.00.00.000000000 AM	15-NOV-22 08.30.00.000000000 AM	Cls5	Loc4
6	Res6	Complete	01-SEP-22 05.30.00.000000000 PM	08-SEP-22 09.00.00.000000000 AM	Cls10	Loc2
7	Res7	Active	08-NOV-22 01.30.00.000000000 PM	10-NOV-22 06.00.00.000000000 PM	Cls4	Loc3
8	Res8	Active	01-NOV-22 10.00.00.000000000 AM	03-NOV-22 06.00.00.000000000 PM	Cls4	Loc1
9	Res9	Active	15-NOV-22 08.00.00.000000000 PM	21-NOV-22 10.00.00.000000000 PM	Cls7	Loc1

Functional Dependencies

- {Reservation_ID} → Reservation_Status, RentPrd_In, RentPrd_Out, VehClass_Id, Loc_ID}

Key

{Reservation_ID}

Normalization

The table is already in 3NF due to lack of partial-key dependency and transitive dependency.

RENTAL_RATE

RentPrd_In	RentPrd_Out	VehClass_Id	Reservation_RentalRate
PK	PK	PK	

Sample Data

```
SELECT * FROM RENTAL_RATE
/
```

Script Output × Query Result ×

SQL | All Rows Fetched: 9 in 0.022 seconds

	RESERVATION_RENTALRATE	RENTPRD_IN	RENTPRD_OUT	VEHCLASS_ID
1	140	01-NOV-22 10.02.05.000000000 AM	03-NOV-22 10.02.05.000000000 AM	Cls1
2	210	01-DEC-22 10.00.00.000000000 AM	08-DEC-22 10.00.00.000000000 AM	Cls1
3	550	15-NOV-22 10.00.00.000000000 AM	22-NOV-22 10.00.00.000000000 AM	Cls3
4	440	01-JAN-23 04.20.00.000000000 AM	08-JAN-23 12.20.00.000000000 PM	Cls2
5	120	13-NOV-22 08.00.00.000000000 AM	15-NOV-22 08.30.00.000000000 AM	Cls5
6	440	01-SEP-22 05.30.00.000000000 PM	08-SEP-22 09.00.00.000000000 AM	Cls10
7	150	08-NOV-22 01.30.00.000000000 PM	10-NOV-22 06.00.00.000000000 PM	Cls4
8	150	01-NOV-22 10.00.00.000000000 AM	03-NOV-22 06.00.00.000000000 PM	Cls4
9	370	15-NOV-22 08.00.00.000000000 PM	21-NOV-22 10.00.00.000000000 PM	Cls7

Functional Dependencies

- {RentPrd_In, RentPrd_Out,VehClass_Id} → Reservation_RentalRate

Key

{RentPrd_In, RentPrd_Out,VehClass_Id}

Normalization

The table is already in 3NF due to lack of partial-key dependency and transitive dependency.

RENTAL_AGREEMENT

Contract_No	RentAgree_Status	CurrDate_Start	CurrDate_End	Odometer_Start	Odometer_End	VIN_No	Lisc_Num	Res_ID
PK						FK	FK	FK

Sample Data

Worksheet | Query Builder

```
SELECT * FROM RENTAL_AGREEMENT
```

Query Result

SQL | All Rows Fetched: 7 in 0.024 seconds

	CONTRACT_NO	RENTAGREE_STATUS	CURRDATE_START	CURRDATE_END	ODOMETER_START	ODOMETER_END	VIN...	LISC_NUM	RES_ID
1	Contr1	Active	01-NOV-22 10.00.00.000000000 AM	(null)	2200	(null)	VIN01	Lisc000012345678	Res1
2	Contr2	Active	01-NOV-22 10.00.00.000000000 AM	(null)	4500	(null)	VIN03	Lisc000112345678	Res3
3	Contr3	Complete	13-NOV-22 06.00.00.000000000 PM	15-NOV-22 07.00.00.000000000 PM	10000	10090	VIN02	Lisc000712345678	Res5
4	Contr4	Complete	01-SEP-22 05.30.00.000000000 PM	08-SEP-22 09.00.00.000000000 AM	50000	50500	VIN06	Lisc001012345678	Res6
5	Contr5	Complete	08-NOV-22 01.30.00.000000000 PM	10-NOV-22 06.00.00.000000000 PM	500	750	VIN04	Lisc000812345678	Res7
6	Contr6	Complete	10-NOV-22 06.00.00.000000000 PM	10-NOV-22 06.00.00.000000000 PM	1500	1600	VIN05	Lisc000812345678	Res8
7	Contr7	Complete	15-NOV-22 06.00.00.000000000 PM	21-NOV-22 06.00.00.000000000 PM	8200	8300	VIN07	Lisc000812345678	Res9

Functional Dependencies

- {Contract_No} → {RentAgree_Status, CurrDate_Start, CurrDate_End, Odometer_Start, Odometer_End, VIN_No, Lisc_Num, Res_ID}

Key

{Contract_No}

Normalization

The table is already in 3NF due to lack of partial-key dependency and transitive dependency.

BILLING

Billing_ID	Payment_Status	Discount	ActualCost	Contr_no
PK				FK

Sample Data

Worksheet | Query Builder

```
SELECT * FROM BILLING
```

Query Result

SQL | All Rows Fetched: 5 in 0.028 seconds

	BILLING_ID	PAYMENT_STATUS	DISCOUNT	ACTUALCOST	CONTR_NO
1	Bill_1	Pending	10	430	Contr4
2	Bill_2	Complete	0	150	Contr5
3	Bill_3	Complete	0	120	Contr3
4	Bill_4	Pending	0	150	Contr6
5	Bill_5	Pending	20	350	Contr7

Functional Dependencies

- {Billing_ID} → {Payment_Status, Discount, ActualCost, Contr_No}

Key

{Billing_ID}

Normalization

The table is already in 3NF due to lack of partial-key dependency and transitive dependency.

Sample SQL Queries

Calculate the total cost for the bills in each Billing status type - 'Pending', 'Complete'.

Worksheet | Query Builder

```
SELECT Payment_status, Sum(ActualCost)
FROM BILLING
GROUP BY Payment_status
```

Script Output | Query Result

SQL | All Rows Fetched: 2 in 0.025 seconds

	PAYMENT_STATUS	SUM(ACTUALCOST)
1	Pending	930
2	Complete	270

<u>For each credit card type, list all expiry years which are less than 2026. Sort the list in ascending order of the 'Card Type'.</u>

```
SELECT CardType, CardExp_year
FROM Credit_Card
GROUP BY CardExp_year, CardType
HAVING CardExp_year < 2026
ORDER BY CardType
/
```

Query Result

All Rows Fetched: 6 in 0.035 seconds

	CARDTYPE	CARDEXP_YEAR
1	Discover	2024
2	Discover	2025
3	MasterCard	2023
4	MasterCard	2025
5	Visa	2024
6	Visa	2025

Retrieve all the reservations whose 'rental rate' is greater than the 'actual cost' of all the rental agreement contracts. List the 'Reservation ID', 'Reservation Status' and 'Rental Rate' in the output table.

```
SELECT RESERVATION_ID, RESERVATION_STATUS, RESERVATION_RENTALRATE
FROM Rental_Rate RR
INNER JOIN RESERVATION RS
ON RR.RENTPRD_IN = RS.RENTPRD_IN and RR.RENTPRD_OUT = RS.RENTPRD_OUT and RR.VEHCLASS_ID = RS.VEHCLASS_ID
WHERE RESERVATION_RENTALRATE >= ALL (SELECT     ActualCost
                                     FROM       BILLING
                                     );
/
```

Query Result

All Rows Fetched: 3 in 0.028 seconds

	RESERVATION_ID	RESERVATION_STATUS	RESERVATION_RENTALRATE
1	Res4	Cancelled	440
2	Res6	Complete	440
3	Res3	Active	550

<u>List all the reservations along with their current status, for those Class IDs whose daily rate exceeds $65.</u>

Worksheet | Query Builder

```
SELECT RESERVATION_ID, RESERVATION_STATUS, VEHCLASS_ID
FROM RESERVATION
WHERE VEHCLASS_ID IN (SELECT     Class_Id
                      FROM       Class
                      WHERE      DailyRate > 65);
```

Script Output | Query Result

All Rows Fetched: 6 in 0.024 seconds

	RESERVATION_ID	RESERVATION_STATUS	VEHCLASS_ID
1	Res1	Complete	Cls1
2	Res2	Active	Cls1
3	Res3	Active	Cls3
4	Res7	Active	Cls4
5	Res8	Active	Cls4
6	Res9	Active	Cls7