Systems Analysis

and Design

Instructor : Huang, Chuen-Min

**Teamwork ver.2**

Group 7

|  |  |
| --- | --- |
| ID | Name |
| B10423042 | Jonathan |
| B10423046 | Vicky |
| B10123034 | Celia |
| B10523003 | Hank |
| B10523005 | Aliss |
| B10523021 | Johnny |
| B10523027 | Blaire |
| B10523046 | Reo |
| B10523054 | Michael |
|  |  |
| Date 2018/ 5 / 28 | |

Content

[**1)** 3](#_Toc515292774)

[**2)** 6](#_Toc515292775)

[**3)** 9](#_Toc515292779)

[Coincidental cohesion 9](#_Toc515292780)

[Procedural cohesion 11](#_Toc515292781)

[Functional cohesion 12](#_Toc515292784)

[**4)** 13](#_Toc515292786)

[Name Connascence 13](#_Toc515292787)

[Position Connascence 13](#_Toc515292788)

[Convention Connascence 13](#_Toc515292791)

[**5)** 14](#_Toc515292792)

[**6)** 15](#_Toc515292794)

[Contract 15](#_Toc515292795)

[Method Specification 16](#_Toc515292796)

[**7)** 18](#_Toc515292797)

[**8)** 20](#_Toc515292798)

[**9)** 21](#_Toc515292799)

[Zero Normal Form 21](#_Toc515292800)

[First Normal Form 22](#_Toc515292801)

[Second Normal Form 23](#_Toc515292802)

[Third Normal Form 24](#_Toc515292803)

[**10)** 25](#_Toc515292804)

[**11)** 27](#_Toc515292806)

[**Participate In Assignments** 29](#_Toc515292807)

**Based on your team project of team work 1, please do the following:**

# **Please explain the Law of Demeter (LoD) by using of your project.**

|  |  |
| --- | --- |
| Law of Demeter (LoD) | symbol |
| (1) to itself (O itself) | ① |
| (2) to objects contained in attributes of itself or a superclass (Any objects created/instantiated within M) | ② |
| (3) to an object that is passed as a parameter to the method (M’s parameters) | ③ |
| (4) to an object that is created by the method (O's direct component objects) | ④ |

|  |
| --- |
| Class Register |
| C:\Users\Administrator\Desktop\CODE圖片\register.png  ①  Register will get payment machine as parameter that can use payment machine’s plate (input by customer) to return the info which customer want.  ③  ④  ParkingInfo is created by Register, so register can call ParkingInfo to update Info.  To create info, register will call itself to check temporary plate.  After save the temporary plate, register will call itself to record Enter time.  ① |

|  |
| --- |
| Class PaymentMachine -1 |
| C:\Users\Administrator\Desktop\CODE圖片\PM1.png  ①  After finishing pay bill, payment machine will ask customer to print receipt or nor automatically.  Payment machine has register and parking information as its attribute.  So payment machine can call register to update info, and parking info to check if customer pay the bill or not.  ②  ②  ②  ②  ② |

|  |
| --- |
| Class PaymentMachine -2 |
| Payment machine send itself as parameter to register to get parking info.  ①  ②  Parking info is payment machine’s attribute, so payment machine can call parking info to get some info.  ②  ①  ③  These four methods in payment machine are called by payment machine itself. |

# **There are six (or seven) types of interaction coupling, each falling on different parts of a good-to-bad continuum. Choose three pieces of your project to describe what types of the coupling they belong to.**

|  |
| --- |
| Stamp Coupling |
| Class Register  C:\Users\Administrator\Desktop\CODE圖片\register.png  Payment machine sends all itself to register to get info.  But register will only use car plate that customer input to payment machine.  Class PaymentMachine    Payment machine sends all itself to register to get info. |

|  |
| --- |
| Data coupling |
| Class Controller  C:\Users\Administrator\Desktop\CODE圖片\controller.png  Class ParkingSpace  Only pass the space number (chosen by customer) as parameter to Parking Space.  Not all data structure.  C:\Users\Administrator\Desktop\CODE圖片\space.png |

|  |
| --- |
| Control coupling |
| Class Controller  C:\Users\Administrator\Desktop\CODE圖片\controller.png  Parking space will send return message which will pass space’s status to Controller.  The status will affect what controller will do. |

# **There are seven types of method cohesion, choose three pieces of your project to describe what types of the cohesion they belong to.**

|  |
| --- |
| Coincidental cohesion  Too many things are process in method searchParkInfo. |
| Class PaymentMachine  C:\Users\Administrator\Desktop\CODE圖片\PM1.png |

|  |
| --- |
| Procedural cohesion  There are many things that have weakly related execute in certain sequence.  In method requestEnter, will save temporary plate, open gate, minus one available space, enter, and close gate. |
| Class Controller  C:\Users\Administrator\Desktop\CODE圖片\controller.png |

|  |
| --- |
| Functional cohesion  Each method did one function or one thing. |
| Class ParkingInfo  C:\Users\Administrator\Desktop\CODE圖片\PINFO.png |

# **Connascence generalized the ideas of cohesion and coupling, use three pieces of your project to describe what types of the connascence they belong to.**

**①Name Connascence**

If any name of method in Register changed, then the method in PaymentMachine won’t be able to execute successfully.

(Because PaymentMachine must get ParkingInfo through Register, if method of Register changes its name, PaymentMachine will no longer call the register)

**②Position Connascence**

If setLeaveTime, computeFee or payBill changes the position, PaymentMachine won’t be able to execute successfully.

|  |
| --- |
|  |

**③Convention Connascence**

If feeTable’s value changes, the fee will be different.

|  |
| --- |
|  |

# **Use one class from your project that can create a set of invariants and add them to the CRC card or the class diagram.**

Front

|  |  |  |
| --- | --- | --- |
| **Class name: PaymentMachine** | **ID:3** | **Type:** concrete ,Domain |
| **Description:** | **Association Use Case:** | |
| Get car’s parking time from ParkingInfo.  Compute fee and pay.  Select print receipt or not. | **Pay payment** | |
| **Responsibilities:** | **Collaborators:** | |
| payBill |  | |
| searchParkInfo | **Register** | |
| setLeaveTime |  | |
| computeFee | **ParkingInfo** | |
| printRecipt | **ParkingInfo** | |
| convertTime |  | |

Back

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Attributes: | | | | |
| register | (1..1) | (Register) | |  |
| info | (1..1) | (PrakingInfo) | |  |
| leaveTime | (1..1) | (Date) | |  |
| searchPlate | (1..1) | (String) | |  |
| fee | (1..1) | (int) | |  |
| payType | (1..1) | (int) | |  |
| money | (1..1) | (int) | |  |
| change | (1..1) | (int) | |  |
| printReceipt | (1..1) | (int) | |  |
| diff | (1..1) | (double) | |  |
| week | (1..1) | (String) | |  |
| feeTable | (1..1) | (int[]) | |  |
| Relationships: | | |  | |
| Generalization(a-kind-of): | | |  | |
| Aggregation(has-parts): | | | | |
| Other Associations: Register, ParkingInfo | | | | |

# **Use a method of a class from your project that can create a contract and describe its algorithm specification. Specify the pre- or post- condition and use both Structured English and an activity diagram to specify the algorithm.**

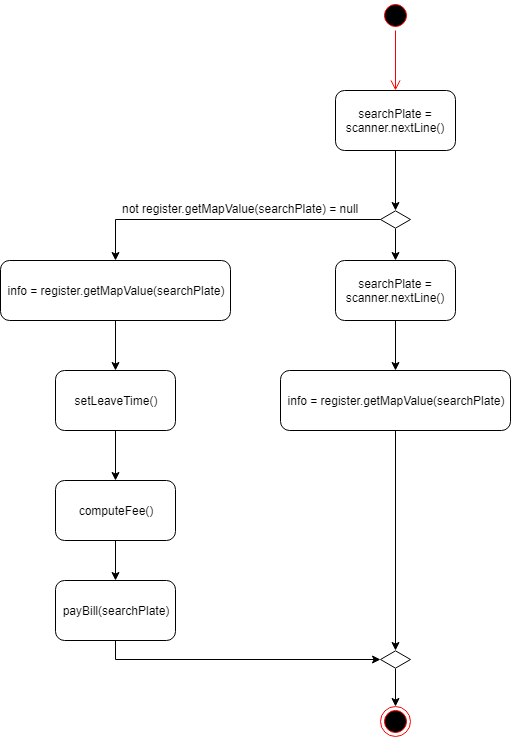
Contract

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Method Name:** | searchParkInfo | | **Class Name:** | PaymentMachine | **ID:** | 1 |
| **Client(consumers):** | | Costumer | | | | |
| **Associated Use Case:** | | | | | | |
| Pay payment | | | | | | |
| **Description of Responsibilities:** | | | | | | |
| Search ParkInfo to get the customer’s enterTime. Show the fee to interface. Customer need to select payType and pay it. To decision money is enough or not. In the end, customer need to select print receipt or not. | | | | | | |
| **Arguments Received:** | | | | | | |
| void | | | | | | |
| **Pre-Conditions:** | | | | | | |
| Not ParkingInfo.include(plate) = NULL. | | | | | | |
| **Post-Conditions:** | | | | | | |
| payBill(Plate) | | | | | | |

Method Specification

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Method Name:** | searchParkInfo | | | **Class Name:** | | PaymentMachine | | | **ID:** | | 2 |
| **Contract ID:** | 12 | | | **Programmer:** | | Michael | | Data Due: | | 5/25/18 | |
| **Programming Language:** | | | | | | | | | | | | |
| Java | | | | | | | | | | | | |
| **Triggers/Events:** | | | | | | | | | | | | |
| Auto Pay Machine wants to know customer’s enter time. | | | | | | | | | | | | |
| **Arguments Received:**  **Arguments Data Type:** | | **Notes:** | | | | | | | | | |
| String | | The car’s plate number. | | | | | | | | | |
|  | |  | | | | | | | | | |
| **Messages Sent & Argument Passed:**  **ClassName.MethodName:** | | | | | **Data Type:** | | **Notes:** | | | | |
| register.getMapValue(searchPlate) | | | | | Register | |  | | | | |
| **Arguments Returned:**  **Data Type:** | | | **Notes:** | | | | | | | | |
| ParkingInfo | | | The customer’s parking information. | | | | | | | | |
| **Algorithm Specification:** | | | | | | | | | | | |
| IF register.getMapValue(Plate) != null  info = register.getMapValue(Plate)  setLeaveTime()  computeFee()  payBill(Plate)  ElSE  Plate = scanner.nextLine()  register.getMapValue(Plate) | | | | | | | | | | | |
| **Misc.Notes:** | | | | | | | | | | | |
| None | | | | | | | | | | | |

**Activity Diagram**



# **Please evaluate any piece of your project in terms of cohesion, coupling, and connascence perspective.**

Coupling (Control, data coupling) (①)

Cohesion (functional cohesion) (②)

Connascence (Position connascence) (③)

|  |
| --- |
| Data coupling  Only pass the space number (chosen by customer) as parameter to Parking Space.  Not all data structure. |
| Class Controller  C:\Users\Administrator\Desktop\CODE圖片\controller.png |

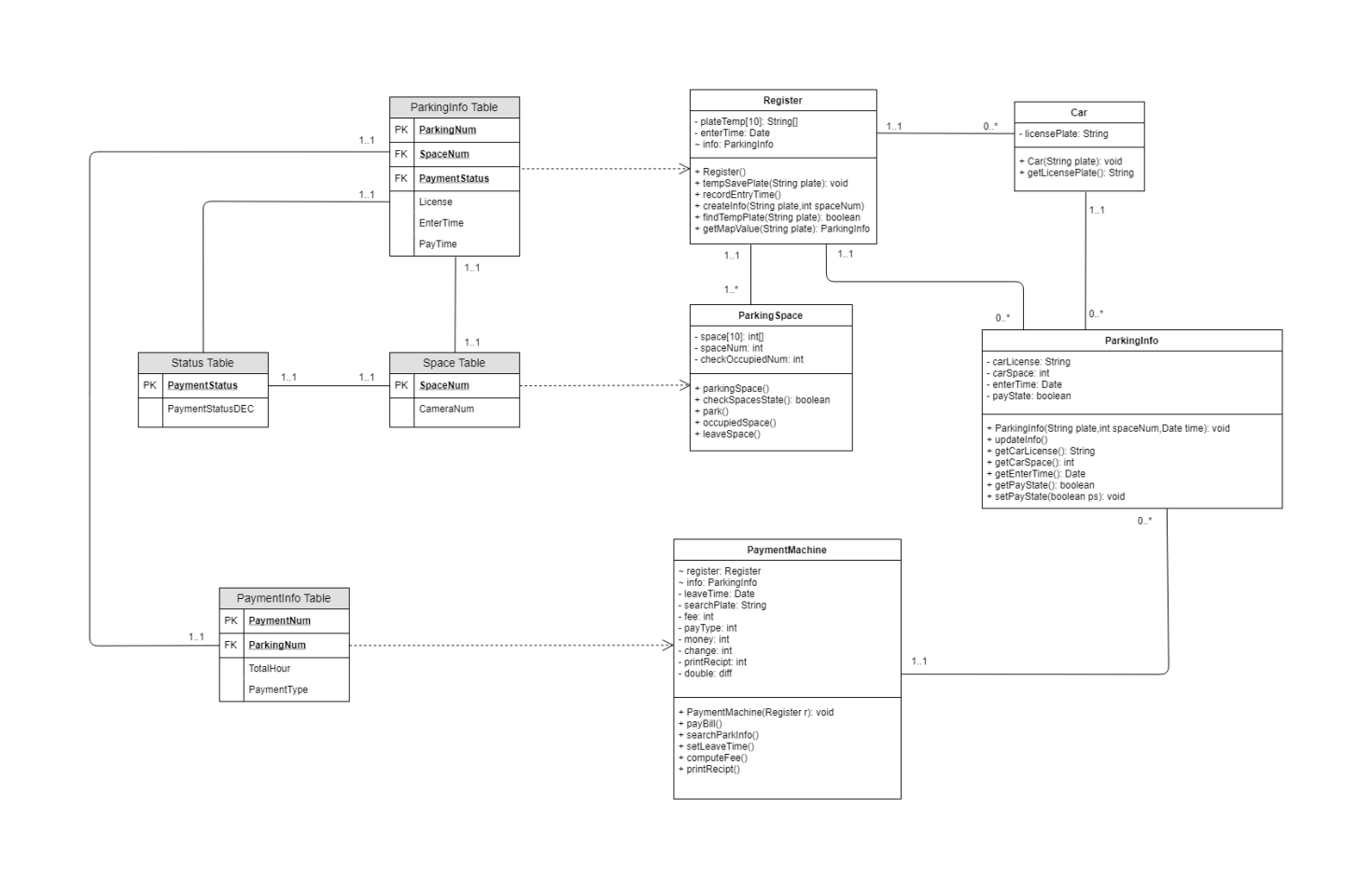
|  |
| --- |
| Control coupling  Parking space will send return message which will pass space’s status to Controller.  The status will affect what controller will do. |
| Class Controller  C:\Users\Administrator\Desktop\CODE圖片\controller.png |

|  |
| --- |
| Functional cohesion  Each method did one function or one thing. |
| Class ParkingInfo  C:\Users\Administrator\Desktop\CODE圖片\PINFO.png |

|  |
| --- |
| Position Connascence  If setLeaveTime, computeFee or payBill changes the position, PaymentMachine won’t be able to execute successfully. |
| Class ParkingInfo |

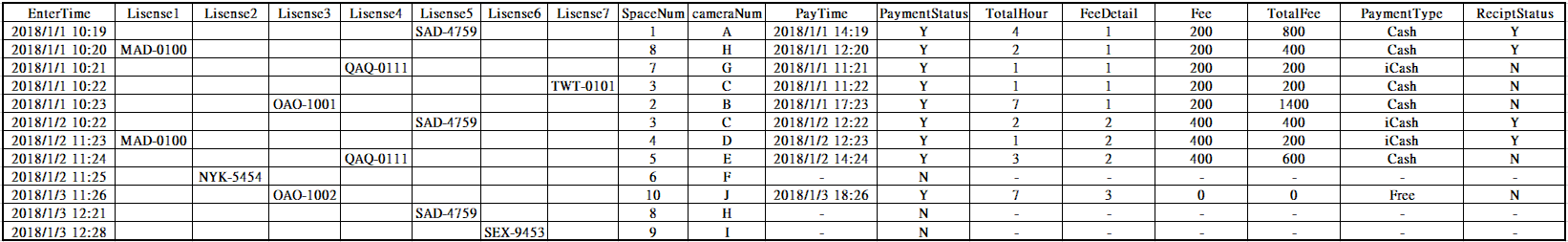
# **Assume that you are going to adopt RDBMs to your project, please describe the referential integrity.**

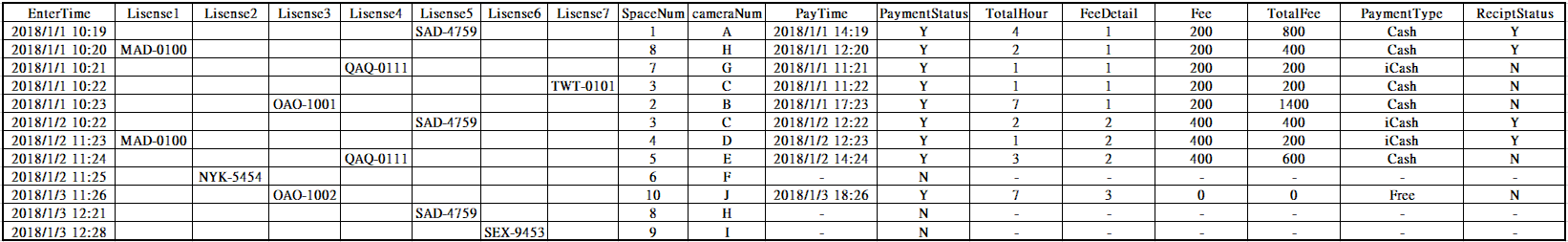
Register class creates ParkingInfo Table. Parking Space class get SpaceNum from Space Table. PaymentMachine get PaymentNum, ParkingNum, TotalHour and PaymentType from Payment Table.



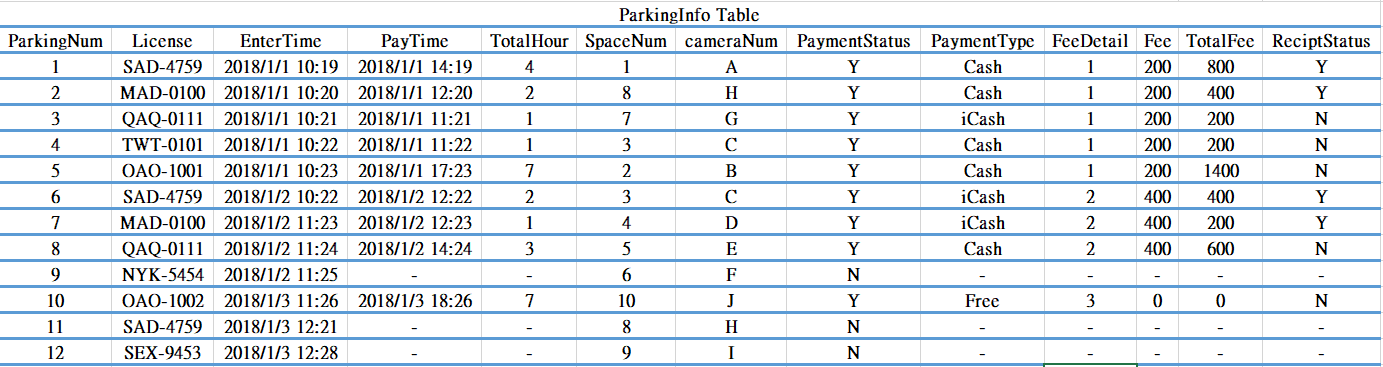
# **Using the steps of normalization, create a model that represents the file of your project in third normal form. Please make necessary assumptions to explain why the tables are related.**

**Zero Normal Form**

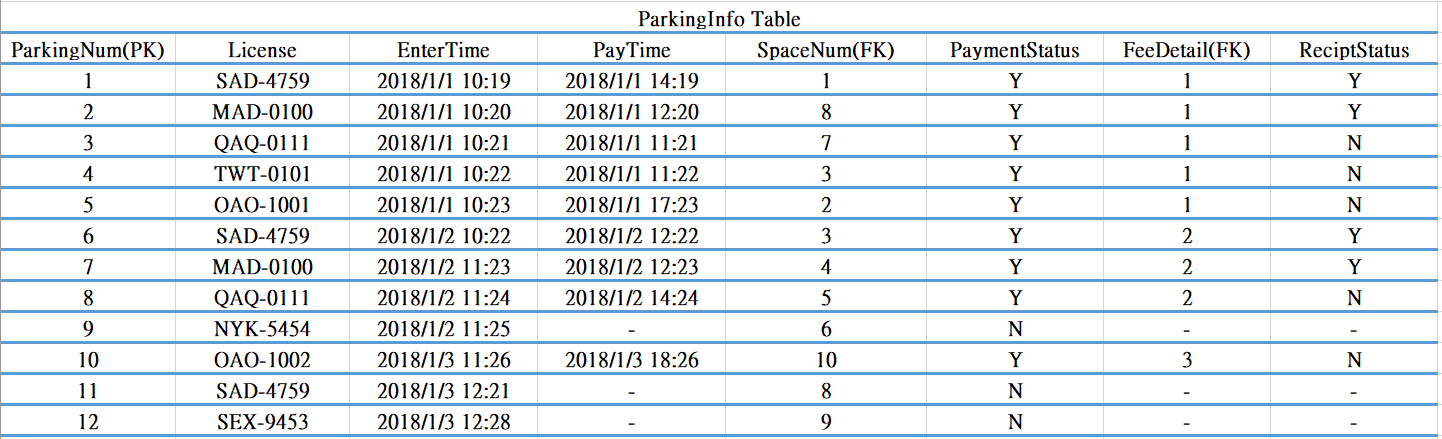
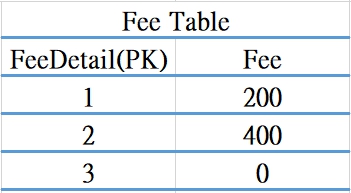


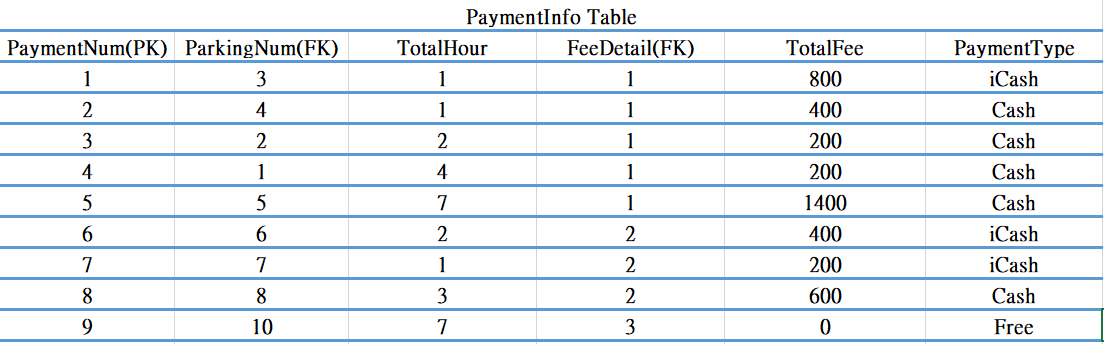
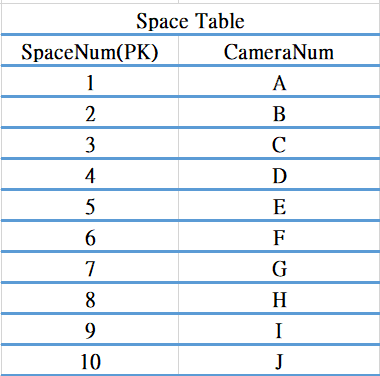


**First Normal Form**



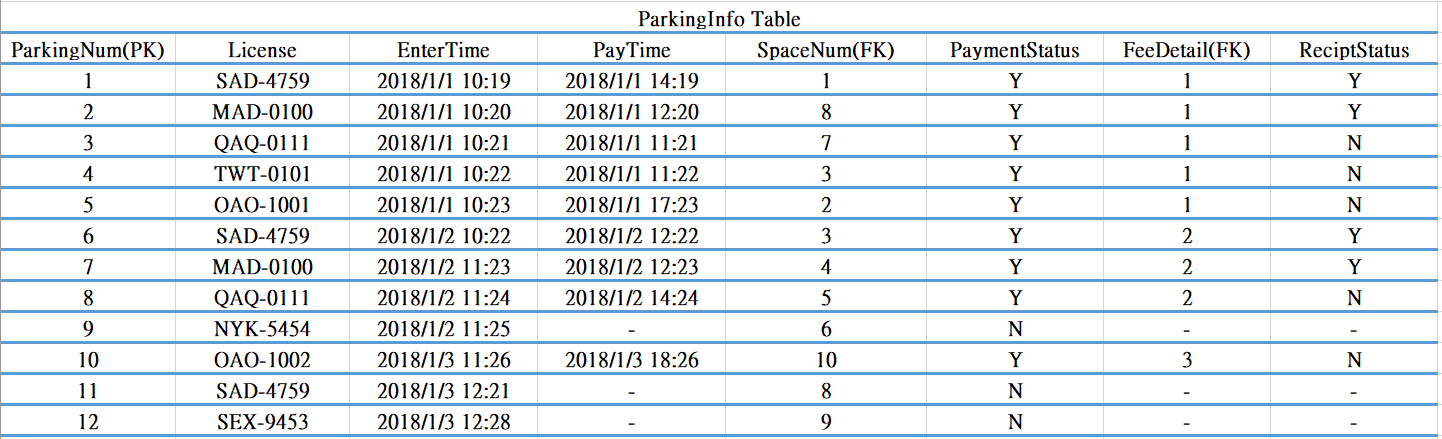
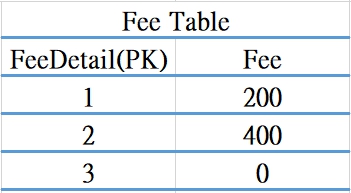
**Second Normal Form**

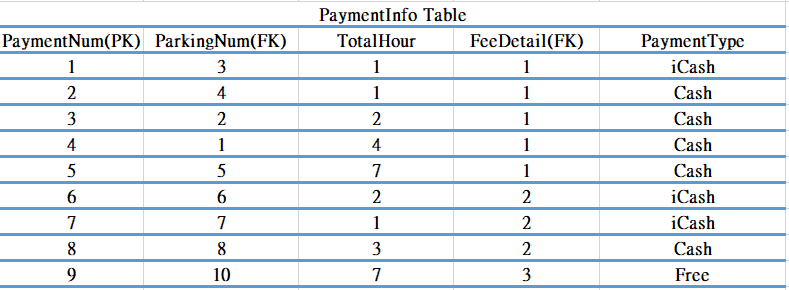
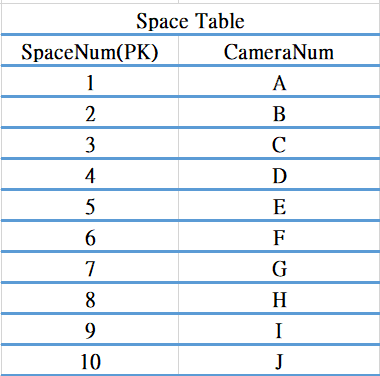


**Third Normal Form**

We turn our parking file into four tables include ParkingInfo, PaymentInfo, Space, and Status. ParkingInfo has SpaceNum and PaymentStatus as FK connect with Payment table and Space table.ParkingNum is PK that is unique data in ParkingInfo table. Space table contains SpaceNum and CameraNum. Other tables that have SpaceNum as FK can get CameraNum through SpaceNum. Payment table has ParkingNum as FK connect ParkingInfo table to get License, EnterTime, PayTime, etc. Status table contains PaymentStatus and PaymentStatusDEC. Other tables that have PaymentStatus as FK can get PaymentStatusDEC through PaymentStatus.



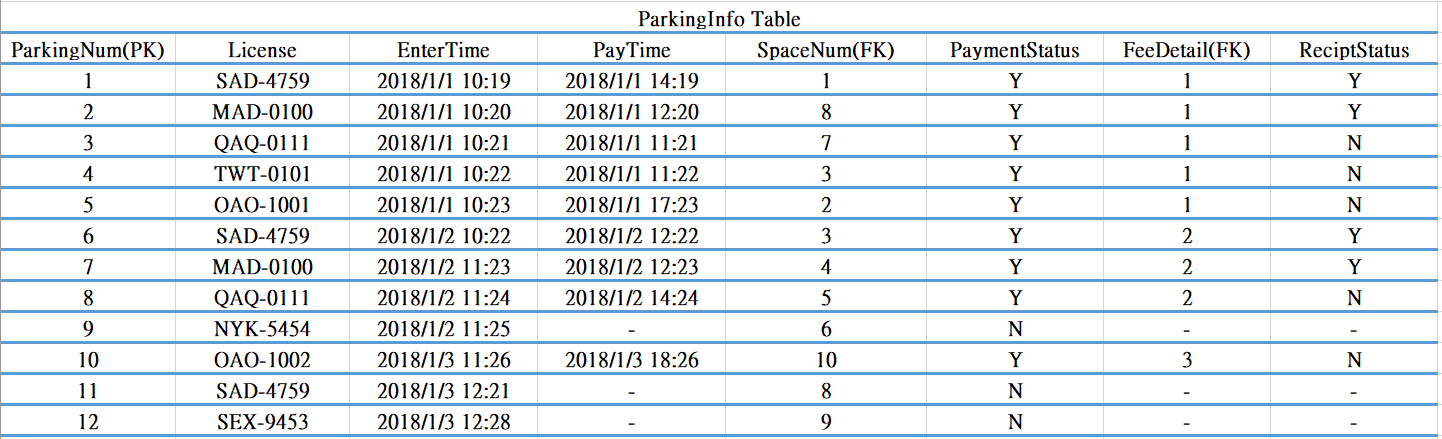
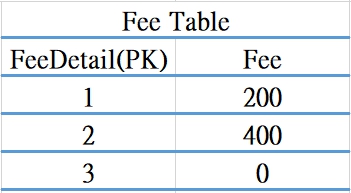
 

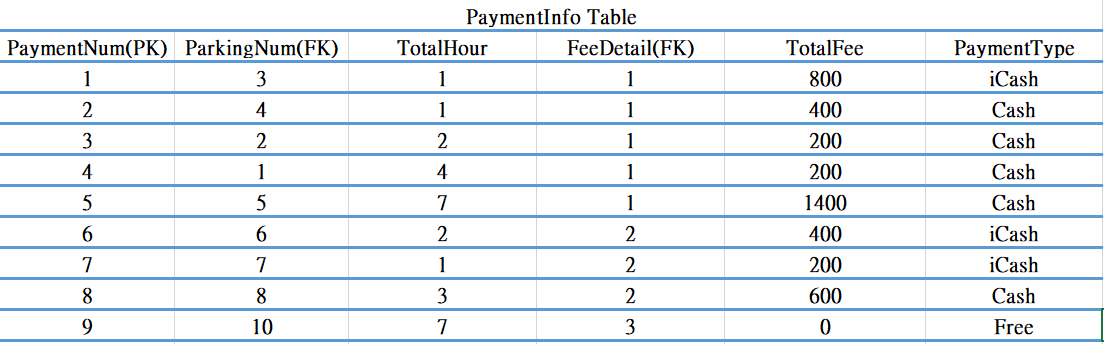
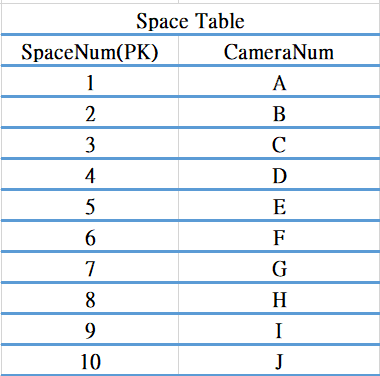
# **Describe how you would denormalize the model that you created in question 9. Draw the new class diagram based on your suggested changes.**

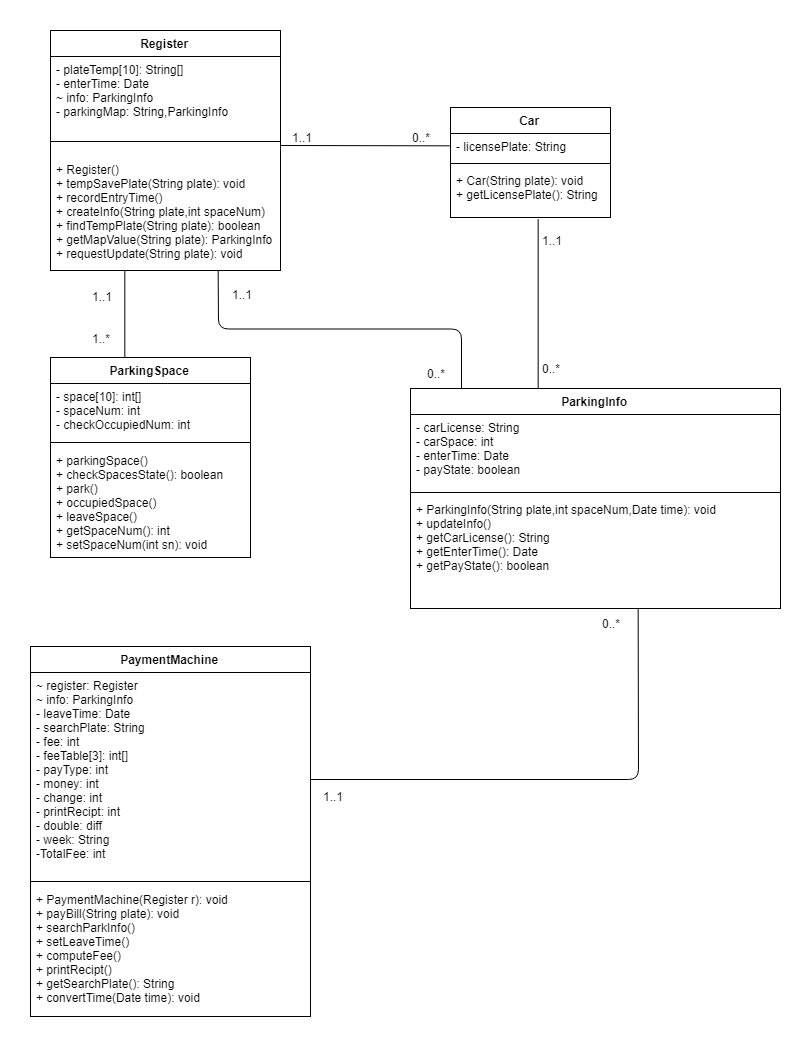
**Denormalize Form**

Whenever we calculate the fee that customer needs to pay, we always need to use FK, FeeDetail, to connect another table.It's not an effective way.

So we decided to change table into the table below. Let system can access data easier and faster.



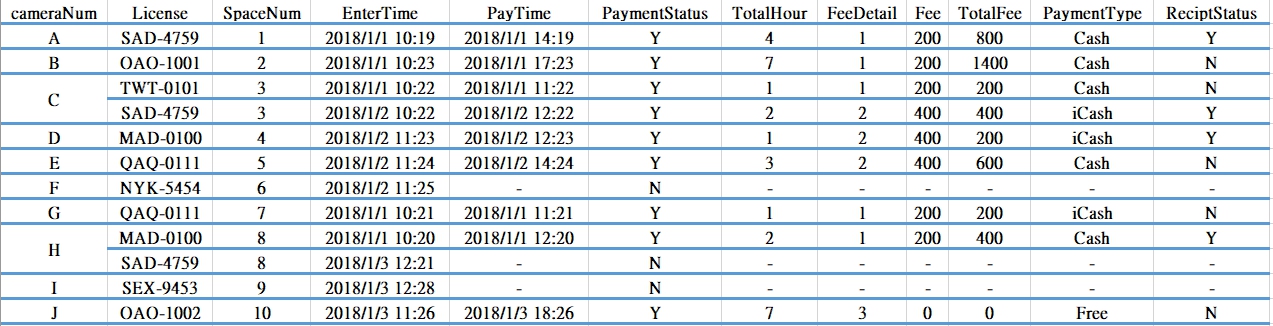
**class diagram**

# **Examine the model that you created in question 10. Develop the inter-file clustering and index strategies. Describe how your clustering strategy will improve the performance of the database. List possible indices you would recommend and describe the reasons.**

**Inter-file clustering**

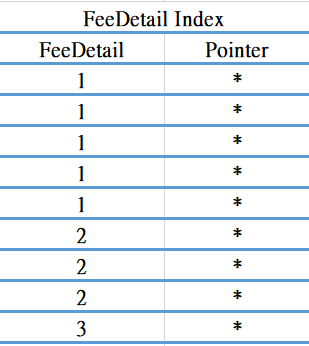
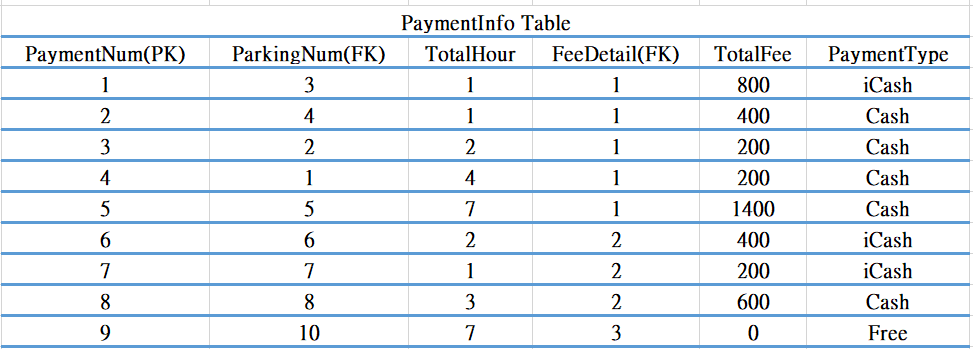
We can clearly know which parking space is parked more often. We also can judge the parking space seldom parked have any problem or not.

Let us can improve the environment of this parking lot.



**Inter-file index**

We can use FeeDetail Index to calculate our revenue that comes from three type of Fee. We can judge is our holiday income more than usual.

# **Participate In Assignments**

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Name | Participate | Responsibility |
| B10423042 | Jonathan | 100% | 6)  8)  9)  10)  11)  check file  Java Code |
| B10523046 | Reo | 100% | 1)  2)  3)  4)  5)  6)  7)  Java Code |
| B10523021 | Johnny | 100% | 6)  8)  9)  10)  11)  Java Code |
| B10523005 | Aliss | 100% | 8)  9)  10)  11)  word  check file |
| B10423046 | Vicky | 70% | 5)  6) |
| B10523054 | Michael | 100% | 1)  2)  3)  4)  Java Code  check file |
| B10123034 | Celia | 75% | 9) |
| B10523003 | Hank | 0% |  |
| B10523027 | Blaire | 0% |  |