國 立 中 央 大 學

資 訊 管 理 學 系

系統分析與設計

系統軟體設計規格書

第二十一組

指導教授:許智誠 教授 、陳以錚 教授

第 2 章 資料庫設計

設計階段之資料庫,根據分析文件之實體關係圖(Entity-Relation Diagram),進行確認並依據其規劃資料庫之資料表,共計包含 5 個實體(Entity)、6 個關係(Relationship)、0 個複合性實體(Compound Entity),下圖(圖 1)為設計階段之ER 圖,亦可使用資料庫綱要圖(Schema Diagram)進行取代:

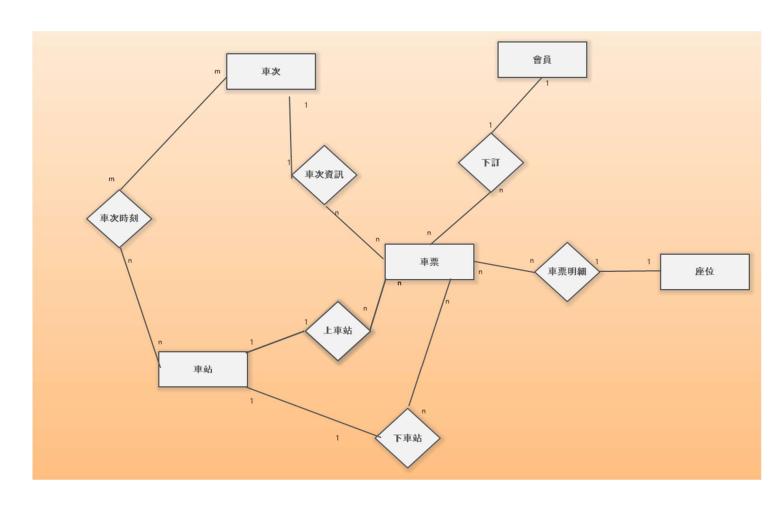


圖 1:實體關係圖

根據上圖進行資料表之設計,以下將逐一說明資料庫每張資料表之欄位,所有 資料表呈現於此:

1. 會員資料表(USER)

表 1: 會員資料表 (USER) 資料結構

Key	名稱	類型	預設值	空值	自動	編碼
					增加	
PK	USER_id	Int	無	否	V	
	USER_email	Varchar(45)	無	否		Utf8
	USER_password	Varchar(45)	無	否		Utf8
	USER_firstname	Varchar(45)	無	否		Utf8
	USER_lastname	Varchar(45)	無	否		Utf8
	USER_dob	date	無	否		
	USER_register_date	date	無	否		
	USER_delete	ENUM('T', 'F')	'F'	否		

註: USER_dob :欄位用於紀錄會員之生日

USER_delete:欄位若為T則代表這個 user 已經被刪除

2. 車站資料表 (STATION)

表 2: 車站資料表 (STATION) 資料結構

Key	名稱	類型	預設	空值	自動	編碼
			值		增加	
PK	STATION_id	Int	無	否	V	
	STATION_name	Varchar(45)	無	否		Utf8
	STATION_location_marker	Int	無	否		
	STATION_time_marker	Int	無	否		

註:

STATION_location_marker:記錄票價基準(票價= 5*(出發站 marker-抵達站

marker)

STATION_time_marker:記錄路程時間基準(路程=(出發站 marker-抵達站

marker)

3. 車次時刻資料表 (TIME)

表 3:分析階段之車次時刻資料表 (TIME) 資料結構

Key	名稱	類型	預	空	自	編
			設	值	動	碼
			值		增	
					加	
PK,FK	BUS_id	Int	無	否	V	
PK,FK	STATION_id_departure_station	Int	無	否		
	TIME_arrival_time	Time	無	是		
	TIME_departure_time	Time	無	是		

註:

TIME_arrival_time :欄位用於紀錄班次抵達該站時間

TIME_departure_time:欄位用於紀錄班次從該站離開時間

4. 車次資料表 (BUS)

表 3:分析階段之車次時刻資料表 (TRAIN) 資料結構

Key	名稱	類型	預設	空值	自動	編碼
			值		增加	
PK	BUS_id	Int	無	否	V	
	BUS _off_date	Date	無	是		
	BUS_on_date	Date	無	否		

註:

BUS_on_date:欄位用於紀錄該班次於幾月幾號開始每日行駛

BUS_off_date:欄位用於紀錄該班次於幾月幾號開始結束行駛

5. 座位資料表 (SEAT)

表 5:分析階段之座位資料表(SEAT)資料結構

Key	名稱	類型	預設	空值	自動	編碼
			值		增加	
PK	SEAT_id	Varchar(3)	無	否		Utf8

註: SEAT_id:欄位用於儲存座位編號(不限車次),如「1A」

6. 車票資料表 (TICKET)

表 6:分析階段之車票資料表(TICKET)資料結構

Key	名稱	類型	預	空	自	編
			設	值	動	碼
			值		增	
					加	
PK	TICKET_id	Int	無	否	V	
FK	USER_id	Int	無	否		
FK	SEAT_id	Int	無	否		
FK	BUS_id	Varchar(3)	無	否		Utf8
FK	STATION_id_arrival_station	Int	無	否		
FK	STATION_id_departure_station	Int	無	否		
	TICKET_bus_date	Int	無	否		
	TICKET_price	Date	無	否		
	TICKET_book_time	Datetime	無	否		
	TICKET_pay_time	Datetime	無	是		

註: TICKET_bus_date:欄位用於紀錄搭乘日期

第3章類別圖

下圖(圖 2、圖 3、圖 4)係依據電子商務線上訂購系統的分析模型和建立的互動圖,以及實體關係圖(Entity-Relation Diagram)所繪製之設 計階段之類別圖(Class Diagram),用於描述系統的類別集合,包含其中之屬性,與類別之間的關係。 本階段之類別圖屬於細部(detail)之設計圖,與上一份文件分析階段之類別圖需要有詳細之變數型態、所擁有之方法,依據這些設計原則,本 類別圖之說明如下所列:類別圖除包含與資料庫相對應之物件外,亦包含相關之控制物件(controller)、DBMgr 與各功能相對應資料庫操作類別 (例如:MemberHelper)和相對應之類別工具(JsonReader)

圖 2: 類別圖

- request_string: String - -request: HttpServletRequest + JsonReader (request : HttpServletRequest) + getParameter (key : String): String + getArray (): JSONArray + getObject (): JSONObject + response (resp_string : String, response : HttpServletResponse) + response (resp : JSONObject, response : HttpServletResponse) + response (status_code : int, resp : JSONObject, response : HttpServletResponse)

DBMgr + JDBC_DRIVER: String + DB_URL: String + USER: String + PASS: String + PASS: String + DBMgr () + getConnection (): Connection + close (stm : Statement, conn : Connection) + close (rs : ResultSet, stm : Statement, conn : Connection)

MemberHelper	
- mh: MemberHelper	
- conn: Connection	
- pres: PreparedStatement	
- MemberHelper ()	
+ getHelper (): MemberHelper	
+ deleteByID (id : int): JSONObject	
+ getAll (): JSONObject	
+ getByID (id : String): JSONObject	
+ getLoginTimesStatus (m : Member): JSONObject	
+ checkDuplicate (m : Member): Boolean	
+ create (m : Member): JSONObject	
+ update (m : Member): JSONObject	
+ updateLoginTimes (m : Member): void	

+ updateStatus (m : member, status : String): void

	BusHelper	
- bh: Bus	All and Marketine	
	onnection	
- pres: Pr	paredStatement	
- BusHelp	er ()	
+ getHel	er (): BusHelper	
	: JSONObject	
+ getByll	List (): JSONObject	
+ getByll) (): BUS	

SeatHelper	
- seath: SeatHelper - conn: Connection - pres: PreparedStatement	
- SeatHelper () + getHelper (): SeatHelper + getAll (): JSONObject + getByIDList (): JSONObject + getByID (): BUS	

TicketHelper
- th: TicketHelper
- conn: Connection
- pres: PreparedStatement
- TicketHelper ()
+ getHelper (): TicketHelper
+ getAll (): JSONObject
+ getByIDList (): JSONObject
+ getByID (): Ticket

StationHelper
- sh: StationHelper - conn: Connection - pres: PreparedStatement
- StationHelper () + getHelper (): StationHelper + getAll (): JSONObject + getBylDList (): JSONObject + getBylD (): BUS

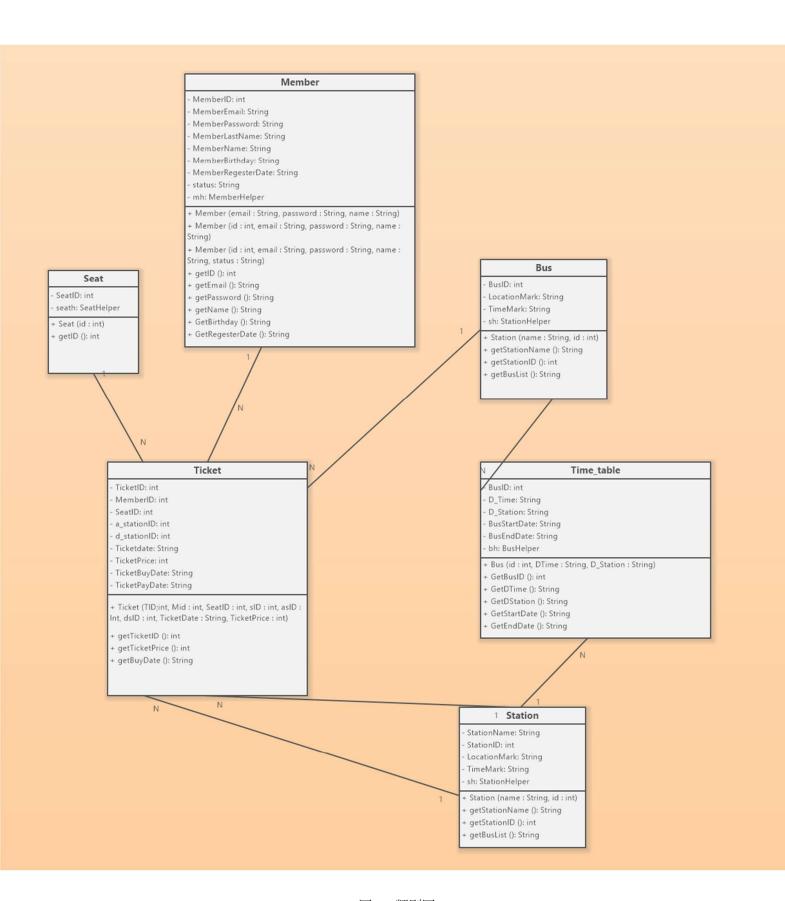


圖 3:類別圖

seatController

- serialVersionID: long
- + SeatController ()
- + doPost (request : HttpServletRequest, response : HttpServletResponse): ()
- + doGet (request : HttpServletRequest, response : HttpServletResponse): ()
- + doDelet (request : HttpServletRequest, response : HttpServletResponse): ()
- + doPut (request : HttpServletRequest, response : HttpServletResponse): () ()

BUSController

- serialVersionID: long
- + bh: BusHelper
- + BusController ()
- + doPost (request : HttpServletRequest, response : HttpServletResponse): ()
- + doGet (request : HttpServletRequest, response : HttpServletResponse): ()
- + doDelet (request : HttpServletRequest, response : HttpServletResponse): () + doPut (request : HttpServletRequest, response : HttpServletResponse): () ()

MemberController

- serialVersionID: long
- mh: MemberHelper
- + MemberController ()
- + doPost (request : HttpServletRequest, response : HttpServletResponse): ()
- + doGet (request : HttpServletRequest, response : HttpServletResponse): ()
- + doDelet (request : HttpServletRequest, response : HttpServletResponse): ()
- + doPut (request : HttpServletRequest, response : HttpServletResponse): () ()

StationController

- serialVersionID: long
- + sh: StationHelper
- + StationController ()
- + doPost (request : HttpServletRequest, response : HttpServletResponse): ()
- + doGet (request : HttpServletRequest, response : HttpServletResponse): ()
- + doDelet (request : HttpServletRequest, response : HttpServletResponse): ()
- $+ \ doPut \ (request: HttpServletRequest, \ response: HttpServletResponse): () \ ()$

TicketController

- serialVersionID: long
- + th: TicketHelper
- + TicketController ()
- + doPost (request : HttpServletRequest, response : HttpServletResponse): ()
- + doGet (request : HttpServletRequest, response : HttpServletResponse): () + doDelet (request : HttpServletRequest, response : HttpServletResponse): ()
- + doPut (request : HttpServletRequest, response : HttpServletResponse): () ()

SearchController

- message: String
- + output: String
- + init ()
- $+ \ doPost \ (request: HttpServletRequest, \ response: HttpServletResponse): ()$
- + doGet (request : HttpServletRequest, response : HttpServletResponse): ()
- $+ \ doDelet \ (request: HttpServletRequest, response: HttpServletResponse): ()$
- + doPut (request : HttpServletRequest, response : HttpServletResponse): () ()

AdminController

- serialVersionID: long
- + ah: AdminHelper
- + AdminController ()
- + doPost (request : HttpServletRequest, response : HttpServletResponse): ()
- + doGet (request : HttpServletRequest, response : HttpServletResponse): ()
- + doDelet (request : HttpServletRequest, response : HttpServletResponse): ()
- + doPut (request : HttpServletRequest, response : HttpServletResponse): () ()

圖 4:類別圖

第 4 章 系統循序圖

本章節主要依照第一份文件需求所產生之使用案例圖(use case)與第二份 文件分析之邏輯階段活動圖與強韌圖為基礎,進行設計階段之循序圖設計,將 每個使用案例進行闡述。於此階段,需要有明確之類別(class)名稱與呼叫之 方法(method)與傳入之變數名稱與型態等細部設計之內容。

4.1 使用案例圖

依據第一份文件針對專案之需求進行確定,本電子商務線上訂購系統預計 共有 4 位動作者與 27 個使用案例,並依照不同之模組區分成不同子系統共計七 個子系統, 其中包含以下圖(圖 5)所示

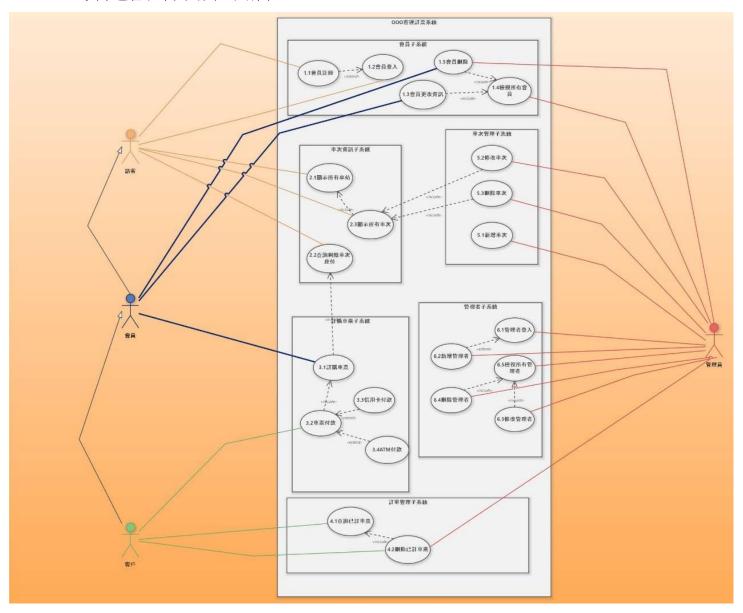


圖 5:使用案例圖

4.2 UseCase 實作之循序圖

4.2.1.1 Sequence Diagram—Use Case 2.3 查詢剩餘車次座位

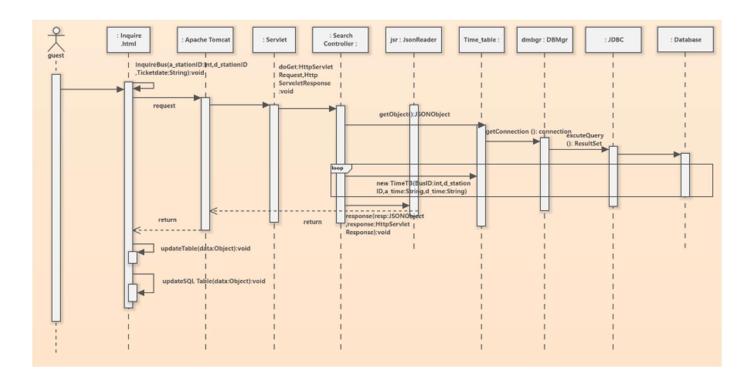


圖 6:商業流程編號 2.3 查詢剩餘車次座位循序圖

- 1. 訪客進入查詢車次頁面(Inquire.html)。
- 2. 進入頁面完成讀取後,透過 JavaScript 之 InquireBus()送出 GET 請求。
- 3. 後端以 SearchController 之 doGet()進行處理,以 JsonReader 判斷參數後,使用 Time_table 物件的 getObject()方法直接取得符合條件的車次時刻資料。
- 4. 透過迴圈將 Result set 內所有車次時刻資料取出,透過資料庫檢索並逐一將每個 車次時刻 new 每個 Time_table 物件,最終將所有車次時刻資料封裝進行回傳。
- 5. 查詢成功則透過 JavaScript 之 updateTable()更新車次時刻列表之表格與 updateSQLTable()更新 SQL 表格之內容。

4.2.1.1 Sequence Diagram—Use Case 3.1 訂購車票

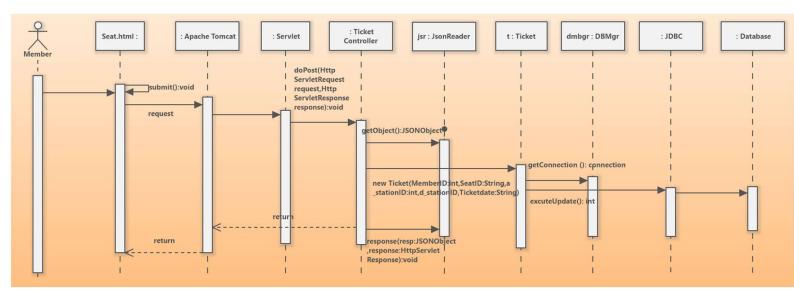


圖 7: 商業流程編號 3.1 訂購車票循序圖

- 1. 會員完成「1.2 會員登入」與「2.3 查詢剩餘車次座位」後,進入選擇座位 頁面(seat.html)。
- 2. 當會員完成座位選擇與確認並通過前端之資料驗證後,透過 JavaScript 之 submit()送出 POST 請求。
- 3. 後端以 TicketController 之 doPost()進行處理
- 4. 以 JsonReader 取回 request 之參數
- 5. 將回傳的資料產生 Ticket 物件
- 6. 使用 Ticket create()新增該車票資料至資料庫當中。