資料庫系統 Final Project II

|  |
| --- |
| **主題: 租車管理系統** |

組別: G06

組員: 50915112廖子科

　 51015107廖玟嫻

　 51015130林敬雅

中華民國 114 年 5 月 3 日

目錄

[壹、 簡介 1](#_Toc197473819)

[貳、 應用情境與使用案例 2](#_Toc197473820)

[參、 系統需求說明 3](#_Toc197473821)

[肆、 概念層模型(完整性限制) 4](#_Toc197473822)

[伍、 ER Diagram及說明 7](#_Toc197473823)

[5.1　ER Diagram 7](#_Toc197473824)

[5.2　ER Diagram – Relationship 8](#_Toc197473825)

[5.3　ER Diagram – Relationship 10](#_Toc197473826)

[5.4　ER Diagram – Entity (and records) 12](#_Toc197473827)

[陸、 SQL與說明 14](#_Toc197473828)

[柒、 參考文獻 19](#_Toc197473829)

1. 簡介

隨著共享經濟的興起，越來越多人選擇租車作為短期出行的解決方案。對於平時無需長期擁有汽車的人來說，購買一輛汽車不僅需要支付數十萬元，還需負擔定期稅金、保險費、保養維修費等長期開銷。因此，當有用車需求時，租車服務成為更為經濟的選擇。

租車管理系統主要為顧客提供線上租車服務。顧客可以透過系統瀏覽可租借車輛的清單，並根據自身需求選擇適合的車輛，提前進行線上預約。當租賃日到來時，顧客只需攜帶有效身分證件與駕照正本，辦理保險與租賃契約，即可順利取車，避免因現場車輛短缺而無法提供租車的情況發生。

此外，租車公司員工可透過系統管理車輛，以及租賃訂單的管理，進一步提升顧客體驗與服務品質。

1. 應用情境與使用案例

難得的連假，許多人都會想和家人或朋友一起出門透透氣，來一趟說走就走的小旅行。但並不是每個人都有自己的車，再加上不同旅伴、目的地與行程安排，常常會需要有更有彈性、能在異地取車還車的交通方式。這時候，透過租車平台預約一台合適的車輛，就成為了最方便又靈活的選擇。

以下為使用案例說明:

|  |  |
| --- | --- |
| 顧客A | 打開本系統，選擇在4/3早上十點在台中分店取車，4/6下午四點在高雄分店還車。 |
| 系統 | 根據所選取的時間查詢可用車輛，回傳車型、租金與可選擇保險。 |
| 顧客A | 透過本系統預約了一台 Toyota Altis，選擇加購全險，獲得訂單編號。 |
| 系統 | 更新該車輛狀態為reserved，透過所選取的車型、時間、保險方案，生成租賃訂單。 |
| 顧客A | 選擇用LINE Pay進行付款。 |
| 系統 | 新增一筆對應的付款紀錄，將付款方式設定為linepay。 |
| 員工 | 在後台查看顧客訂單，確認保險文件與車輛狀態，於4/3完成交車流程。 |
| 顧客A | 在現場出示駕照與身分證後取車。 |
| 系統 | 更新該車輛狀態為rented，更新訂單狀態為active。 |
| 顧客A | 4/6晚上八點在高雄分店還車。 |
| 系統 | 系統判定逾期，計算額外罰金。 |
| 員工 | 檢查車輛確認功能無異常，確認車輛歸還。 |
| 系統 | 更新車輛狀態為available，更新訂單狀態為completed。 |

1. 系統需求說明

針對租車管理系統，須具備以下功能性需求:

1. 會員註冊與登入

會員可以使用Google帳號進行註冊與登入(由後端程式串接Google OAuth 2.0)，系統會檢查資料庫中是否已有該Gmail。如果沒有，顧客將被導向註冊頁面，並需完成電話驗證確保未來聯絡方式，未來也可藉由電話號碼來擴充減少惡意訂單的功能。

1. 車輛瀏覽與選擇

顧客可以在系統中瀏覽所有可租借的車輛，並根據品牌、車型、燃料類型等條件進行篩選。每輛車會顯示詳細資訊，包括每日租金、逾期租金、車輛狀態、可乘坐人數等，讓顧客可以根據需求選擇適合的車輛。

1. 租車預約與個人管理功能

顧客可以選擇租借車輛並進行線上預約，選擇取車和還車的時間與地點，顧客可以查看自己的租賃訂單，包括租賃的車輛、租期、取車地點與還車地點。系統應提供取消訂單的功能，但需遵循時間限制(如:最慢於5天前能到系統修改訂單)。

1. 付款管理

顧客可以在租賃過程中選擇不同的付款方式，如現金、信用卡等。且需提供付款歷史記錄查詢功能，讓顧客可以查看過去的付款紀錄和交易明細。

1. 保險選擇

顧客可以提前了解租車公司提供的各種保險方案，每項保險方案會有詳細的說明，顧客可在租車時選擇附加保險。

1. 車輛與訂單狀態管理

租車公司員工可以查看並更新每輛車的狀態，也可以對租賃訂單進行處理。

1. 安全性功能

每位顧客只能看自己的租賃訂單，不能查看其他顧客的訂單資訊，保護顧客的隱私。

1. 概念層模型(完整性限制)
2. 會員資料表(member Table)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 欄位名稱 | 資料型態 | 是否  可為空 | Domain | 說明 |
| member\_id | INT(11) | 否 | 從1開始遞增的整數 | \*PK |
| google\_id | VARCHAR(50) | 否 |  | Google ID |
| gmail | VARCHAR(255) | 否 | 符合mail格式，且包含'@'  ^[a-zA-Z0-9.\_%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$ | 會員信箱 |
| phone | VARCHAR(10) | 否 | 符合行動電話，  共10碼數字 ^09\d{8}$ | 會員電話 |

1. 保險方案資料表(insurance Table)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 欄位名稱 | 資料型態 | 是否  可為空 | Domain | 說明 |
| insurance\_id | INT(11) | 否 | 從1開始遞增的整數 | \*PK |
| ins\_name | VARCHAR(20) | 否 |  | 保險名稱 |
| coverage | VARCHAR(255) | 否 |  | 保險項目 |
| ins\_fee | INT(4) | 否 | 0 | 每日保險費用 |

1. 地點資料表(location Table)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 欄位名稱 | 資料型態 | 是否  可為空 | Domain | 說明 |
| loc\_id | INT(11) | 否 | 從1開始遞增的整數 | \*PK |
| loc\_name | VARCHAR(15) | 否 |  | 分店名稱 |
| city | VARCHAR(15) | 否 |  | 縣市 |
| district | VARCHAR(15) | 否 |  | 鄉鎮市區 |
| address | VARCHAR(50) | 否 |  | 街道地址 |

1. 車型資料表(Model Table)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 欄位名稱 | 資料型態 | 是否  可為空 | Domain | 說明 |
| model\_id | INT(11) | 否 | 從1開始遞增的整數 | \*PK |
| brand | VARCHAR(30) | 否 |  | 品牌 |
| model\_name | VARCHAR(30) | 否 |  | 車型名稱 |
| car\_type | VARCHAR(10) | 否 | 限定字串  (Compact,Sedan,SUV,MPV) | 車輛類型 |
| fuel\_type | VARCHAR(10) | 否 | 限定字串  (Gasoline,Electric,Hybrid) | 燃油類型 |
| engine\_cc | INT(5) | 否 | 0 | 引擎排氣量(cc) |
| transmission | INT(1) | 否 | 必須0或1 | 變速箱類型  0表示手排 1表示自排 |
| image\_url | VARCHAR(100) | 否 | 圖片路徑位置不得超過100字 | 車型圖片位置 |

1. 汽車資料表(Car Table)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 欄位名稱 | 資料型態 | 是否可為空 | Domain | 說明 |
| car\_id | INT(11) | 否 | 從1開始遞增的整數 | \*PK |
| vin | VARCHAR(50) | 否 |  | 車輛識別碼 |
| plate\_number | VARCHAR(8) | 否 | ^R[A-Z]{2}-\d{4}$ | 車牌號碼 |
| daily\_fee | INT(5) | 否 | 0 | 每日租金 |
| late\_fee | INT(4) | 否 | 0 | 每小時逾期罰金 |
| year\_made | INT(4) | 否 | 1980 (年) | 製造年份 |
| seat\_num | INT(2) | 否 | 0 | 可乘坐人數 |
| color | VARCHAR(20) | 否 |  | 車輛顏色 |
| mileage | INT(6) | 否 | 0 | 里程數 |
| car\_status | ENUM | 否 | 限定字串  ( available, maintenance, disable) | 車輛狀態  預設為available |
| notes | TEXT | 是 |  | 車輛備註 |
| model\_id | INT | 否 | 參照Model Table | FK，車型 |
| loc\_id | INT | 是 | 參照Location Table | FK，地點 |

1. 租賃訂單資料表(Rental Table)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 欄位名稱 | 資料型態 | 是否可為空 | Domain | 說明 |
| rental\_id | INT(11) | 否 | 1以上整數自動增值 | \*PK |
| start\_date | DATETIME | 否 | YYYY/MM/DD  HH:MM:SS | 租賃開始時間 |
| end\_date | DATETIME | 否 | YYYY/MM/DD  HH:MM:SS | 預計結束時間 |
| actual\_return | DATETIME | 是 | 未歸還時可為 NULL | 實際還車時間 |
| created\_at | DATETIME | 否 |  | 訂單建立時間 |
| rental\_status | ENUM | 否 | 限定字串  ( pending , active , completed , cancelled, reject) | 租賃訂單狀態  預設為pending |
| amount | INT(7) | 否 | 0 | 付款金額 |
| payment\_date | DATETIME | 否 | YYYY/MM/DD  HH:MM:SS | 付款日期 |
| method | ENUM | 否 | 限定字串：  ( cash , credit , linepay) | 付款方式 |
| member\_id | INT(11) | 否 | 參照Member Table | FK，會員id |
| car\_id | INT(11) | 否 | 參照Car Table | FK，汽車id |
| pickup\_loc | INT(11) | 否 | 參照Location Table | FK，取車地點 |
| drop\_loc | INT(11) | 否 | 參照Location Table | FK，還車地點 |
| insurance\_id | INT(11) | 否 | 參照Insurance Table | FK，保險代碼 |

1. ER Diagram及說明

5.1　ER Diagram

一張含有 圖表, 方案, 工程製圖, 行 的圖片

AI 產生的內容可能不正確。

圖 1、精簡ERD

一張含有 寫生, 樣式, 白色, 圖畫 的圖片

AI 產生的內容可能不正確。

圖 2、完整ERD

5.2　ER Diagram – Relationship

以下為各實體之間的關聯

1. Member (會員)實體和Rental (租賃訂單)實體之間存在一對多的「擁有」關係：

* 一位會員可以有多筆租賃訂單
* 每筆租賃訂單只會對應到一位會員

Member

Rental

have

1..n

1..1

1. Car (汽車)實體和Model(車型)實體之間存在多對一的「屬於」關係：

* 一個Model可以有多輛Car
* 每輛Car只屬於一個Model

Car

Model

Belongs to

1..1

n..1

1. Car (汽車)實體和Location (地點)實體之間存在多對一的「位於」關係：

* 一輛車只會停一個地點
* 多輛車停放的地點可以是同一個地點

Car

Location

Located at

1..1

n..1

1. Rental (租賃訂單)實體和Insurance (保險方案)實體之間存在多對一的「包含」關係：

* 每一筆租賃只能包含一個保險方案

Rental

Insurance

Include

1..1

n..1

* 一個保險方案可以對應到多筆租賃訂單

1. Rental (租賃訂單)實體和Car(汽車)實體之間存在多對一的「租用」關係：

* 每筆租賃訂單只能租用一輛汽車
* 一輛汽車可以被多筆租賃訂單租用

Rental

Car

Rents

1..1

n..1

1. Rental (租賃訂單)實體和Location(地點)實體之間存在多對一的「取車位置」關係：

* 每筆租賃訂單只能租用指定一個取車的地點
* 一個地點可以是多筆租賃訂單的取車地點

Rental

Location

Pick up

1..1

n..1

1. Rental (租賃訂單)實體和Location(地點)實體之間存在多對一的「還車位置」關係：

* 每筆租賃訂單只能租用指定一個還車的地點
* 一個地點可以是多筆租賃訂單的還車地點

Rental

Location

Drop off

1..1

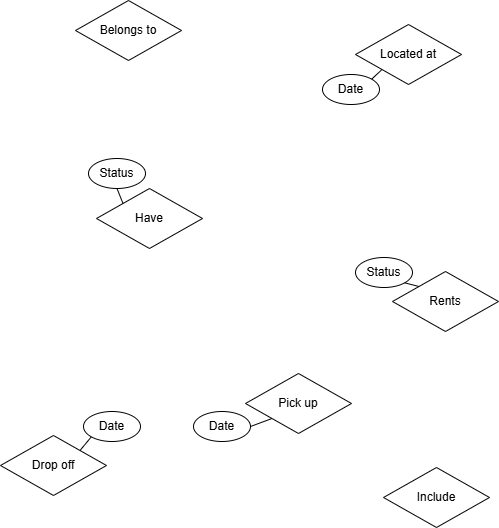
n..1

5.3　ER Diagram – Relationship

1. 一張含有 圖表, 設計 的圖片

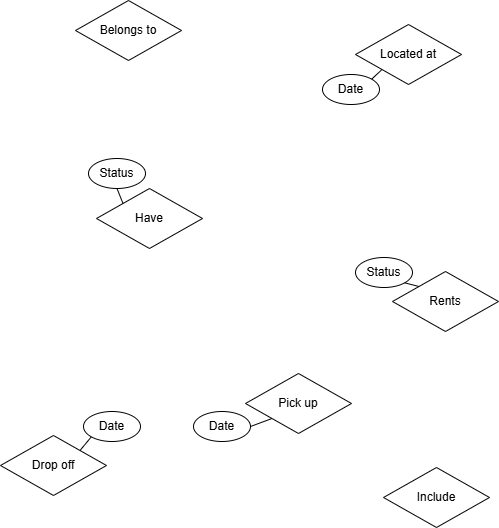
   AI 產生的內容可能不正確。

Car ↔ Model（每台車對應一個車型，一種車型可對應多台車）。

1. 

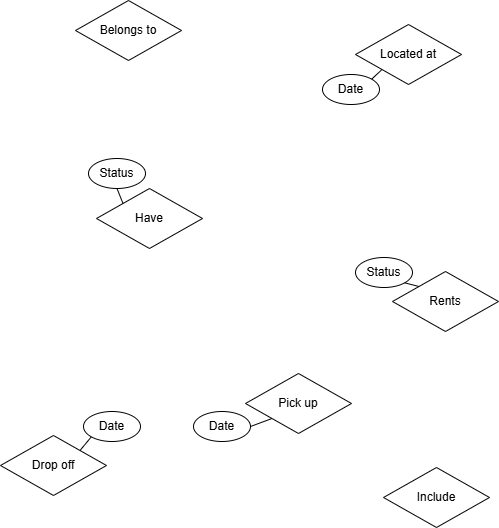
Car ↔ Location（車目前在哪個地點）。

Date：表示該車輛在特定地點的時間點，屬於動態紀錄，可用於追蹤車輛移動歷史。

1. 

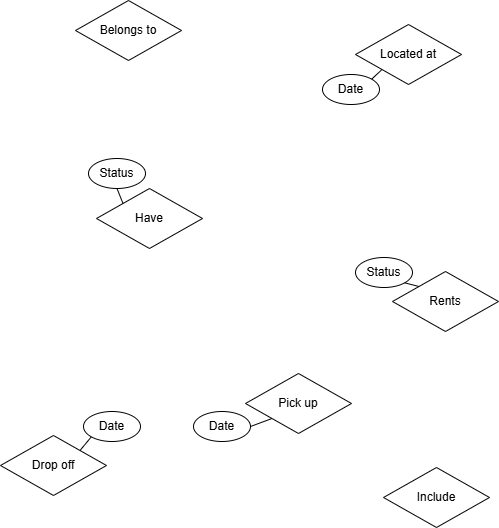
Member ↔ Car，透過 Rental（每次租車是一筆 Rental 記錄）。

Status：表示租車狀態，例如「預約中」「已完成」「已取消」等。

1. 

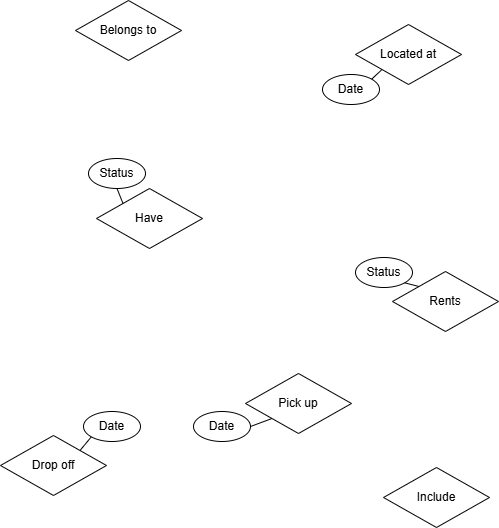
Rental ↔ Location（對應取車地點）。

Date：表示租車開始的日期與時間。

1. 

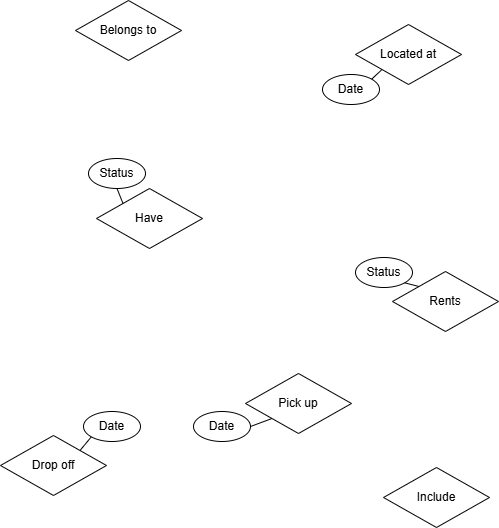
Rental ↔ Location（對應還車地點）。

Date：表示租車還車的日期與時間。

1. 

Member ↔ Rental（每位會員可以有多筆租賃紀錄）。

Status：表示會員對一筆租賃紀錄的狀態，可能是「預約中」「已完成」「已取消」。

1. 

Rental ↔ Insurance（一筆租賃可能包含一或多個保險）。

5.4　ER Diagram – Entity (and records)

1. Member 實體

Member

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| schema | member\_id | google\_id | gmail | phone |
| record1 | 1 | 106455916943881880515 | kulpia@gmail.com | 0978985123 |
| record2 | 2 | 111996782304424104647 | gene\_st@gmail.com | 0914238500 |

1. Model實體

Model

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| schema | model \_id | brand | model \_name | car\_type | fuel\_type | engine  \_cc | trans  mission | image\_url |
| record1 | 1 | TOYOTA | Altis 12 | Compact | Hybrid | 1800 | 1 | img/Altis12.jpg |
| record2 | 2 | HONDA | Odyssey 2019 | MPV | Gasoline | 2500 | 1 | img/ Odyssey12  .jpg |

1. Car實體

Car

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| schema | car \_id | vin | plate\_number | model\_id | daily\_fee |
| record1 | 1 | 1HGCM82633A123456 | RBS-5528 | 1 | 2200 |
| record2 | 2 | 4T1BF1FK5DU123456 | RBX-1128 | 2 | 2600 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| late\_fee | year\_made | loc\_id | seat\_num | color | mileage | notes | car\_status |
| 100 | 2022 | 1 | 5 | 白色 | 38000 |  | available |
| 150 | 2019 | 1 | 7 | 白色 | 25500 |  | available |

1. Rental實體

Rental

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| schema | rental\_id | member\_id | car\_id | start\_date | end\_date | actual\_return |
| record1 | 1 | 1 | 1 | 2025-05-04 08:00:00 | 2025-05-07 21:00:00 | 2025-05-07 20:53:00 |
| record2 | 2 | 1 | 2 | 2025-05-30 18:00:00 | 2025-06-01 21:00:00 | (null) |
| record3 | 3 | 2 | 1 | 2025-05-10 12:00:00 | 2025-05-11 21:00:00 | (null) |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| pickup\_loc | drop\_loc | insurance\_id | rental\_status | amount | method | payment\_date |
| 1 | 1 | 3 | active | 10400 | cash | 2025-05-04 07:50:00 |
| 2 | 2 | 3 | active | 9000 | cash | (null) |
| 1 | 1 | 1 | active | 4400 | credit | (null) |

1. Insurance實體

Insurance

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| schema | insurance\_id | ins\_name | car\_id | ins\_fee |
| record1 | 1 | 基本保險方案 | 第三人責任保險：每人傷害上限200萬、每事故傷害上限400萬、財損上限50萬。駕駛人保險100萬、乘客每人100萬（超載除外）。 | 0 |
| record2 | 2 | 第三人責任升級 | 每人傷害上限提升至500萬、每事故傷害1000萬、財損上限200萬。其餘條件比照基本方案。 | 200 |
| record3 | 3 | 第三人責任與車體損害升級版 | 每人傷害上限提升至500萬、每事故傷害1000萬、財損上限200萬。其餘規定比照基本保險。 | 400 |

1. Location實體

Location

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| schema | loc\_id | loc\_name | city | district | address |
| record1 | 1 | 雲林縣虎尾店 | 雲林縣 | 虎尾鎮 | 文化路64號 |
| record2 | 2 | 雲林縣斗六店 | 雲林縣 | 斗六市 | 大學路三段123號 |

1. SQL與說明
2. customer僅能對公開資料（車輛、車型、保險、地點）進行查詢，只可以對自己的租賃訂單進行查詢、新增與修改操作，確保資料安全性。

權限設定：

* cars表格：允許SELECT操作。
* model表格：允許SELECT操作。
* insurance表格：允許SELECT操作。
* location表格：允許SELECT操作。
* rental表格：允許SELECT, INSERT, UPDATE操作。

1. employee可進行資料管理，包含車輛與保險的增刪修查，且可更新訂單狀態，配合實際業務流程（如繳交罰金、取車與還車等等）。

權限設定：

* model表格：允許SELECT, INSERT, UPDATE, DELETE 操作。
* cars表格：允許SELECT, INSERT, UPDATE, DELETE 操作。
* insurance表格：允許SELECT, INSERT, UPDATE, DELETE 操作。
* location表格：允許SELECT, INSERT, UPDATE, DELETE 操作。
* rental表格：只允許執行UPDATE操作。

|  |
| --- |
| CREATE DATABASE Rental\_Car\_DB;  USE Rental\_Car\_DB;  -- Create member Table  CREATE TABLE member(  member\_id INT(11) AUTO\_INCREMENT PRIMARY KEY,  google\_id VARCHAR(50) NOT NULL,  gmail VARCHAR(255) NOT NULL,  phone VARCHAR(10) NOT NULL,  CONSTRAINT chk\_gmail CHECK (gmail REGEXP '^[a-zA-Z0-9.\_%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$'),  CONSTRAINT chk\_phone CHECK (phone REGEXP '^09\d{8}$')  );  -- Create insurance Table  CREATE TABLE insurance(  insurance\_id INT(11) AUTO\_INCREMENT PRIMARY KEY ,  ins\_name VARCHAR(20) NOT NULL,  coverage VARCHAR(255) NOT NULL,  ins\_fee INT(4) NOT NULL CHECK (ins\_fee >= 0)  );  -- Create location Table  CREATE TABLE location (  loc\_id INT(11) AUTO\_INCREMENT PRIMARY KEY,  loc\_name VARCHAR(15) NOT NULL,  city VARCHAR(15) NOT NULL,  district VARCHAR(15) NOT NULL,  address VARCHAR(50) NOT NULL  );  -- Create model Table  CREATE TABLE model(  model\_id INT(11) AUTO\_INCREMENT PRIMARY KEY,  brand VARCHAR(30) NOT NULL,  model\_name VARCHAR(30) NOT NULL,  car\_type VARCHAR(10) NOT NULL CHECK (car\_type IN ('Compact', 'Sedan', 'SUV', 'MPV')),  fuel\_type VARCHAR(10) NOT NULL CHECK (fuel\_type IN ('Gasoline', 'Electric', 'Hybrid')),  engine\_cc INT(5) NOT NULL CHECK (engine\_cc >= 0),  transmission INT(1) NOT NULL,  image\_url VARCHAR(100) NOT NULL  );  -- Create car Table  CREATE TABLE cars (  car\_id INT(11) AUTO\_INCREMENT PRIMARY KEY,  vin VARCHAR(50) NOT NULL,  plate\_number VARCHAR(8) NOT NULL CHECK (plate\_number REGEXP '^R[A-Z]{2}-[0-9]{4}$'),  daily\_fee INT(5) NOT NULL CHECK (daily\_fee >= 0),  late\_fee INT(4) NOT NULL CHECK (late\_fee >= 0),  year\_made INT(4) NOT NULL CHECK (year\_made >= 1980),  seat\_num INT(2) NOT NULL CHECK (seat\_num > 0),  color VARCHAR(20) NOT NULL,  mileage INT(6) NOT NULL CHECK (mileage > 0),  car\_status ENUM('available', 'maintenance', 'disable') NOT NULL DEFAULT 'available',  notes TEXT,  model\_id INT NOT NULL,  loc\_id INT,  FOREIGN KEY (model\_id) REFERENCES model(model\_id)  ON DELETE NO ACTION,  FOREIGN KEY (loc\_id) REFERENCES location(loc\_id)  ON DELETE SET NULL  );  -- Create rental Table  -- Create rental Table  CREATE TABLE rental (  rental\_id INT(11) AUTO\_INCREMENT PRIMARY KEY,  start\_date DATETIME NOT NULL,  end\_date DATETIME NOT NULL,  actual\_return DATETIME,  created\_at DATETIME NOT NULL,  rental\_status ENUM('pending','active','completed','cancelled','reject') NOT NULL DEFAULT 'pending',  amount INT(7) NOT NULL CHECK (amount >= 0),  payment\_date DATETIME NOT NULL,  method ENUM('cash', 'credit', 'linepay') NOT NULL,  member\_id INT(11) NOT NULL,  car\_id INT(11) NOT NULL,  pickup\_loc INT(11) NOT NULL,  drop\_loc INT(11) NOT NULL,  insurance\_id INT(11) NOT NULL,  CONSTRAINT fk\_member\_id FOREIGN KEY (member\_id) REFERENCES member(member\_id)  ON DELETE CASCADE,  CONSTRAINT fk\_car\_id FOREIGN KEY (car\_id) REFERENCES cars(car\_id)  ON DELETE NO ACTION,  CONSTRAINT fk\_pickup\_loc FOREIGN KEY (pickup\_loc) REFERENCES location(loc\_id)  ON DELETE NO ACTION,  CONSTRAINT fk\_drop\_loc FOREIGN KEY (drop\_loc) REFERENCES location(loc\_id)  ON DELETE NO ACTION,  CONSTRAINT fk\_insurance\_id FOREIGN KEY (insurance\_id) REFERENCES insurance(insurance\_id)  ON DELETE NO ACTION  );  CREATE USER IF NOT EXISTS 'employee'@'localhost' IDENTIFIED BY 'DB@employee';  CREATE USER IF NOT EXISTS 'customer'@'localhost' IDENTIFIED BY 'DB@customer';  -- 客戶端權限  GRANT SELECT ON Rental\_Car\_DB.cars TO 'customer'@'localhost';  GRANT SELECT ON Rental\_Car\_DB.model TO 'customer'@'localhost';  GRANT SELECT ON Rental\_Car\_DB.insurance TO 'customer'@'localhost';  GRANT SELECT ON Rental\_Car\_DB.location TO 'customer'@'localhost';  GRANT SELECT, INSERT, UPDATE ON Rental\_Car\_DB.rental TO 'customer'@'localhost';  -- 授權員工權限  GRANT SELECT ON Rental\_Car\_DB.member TO 'employee'@'localhost';  GRANT SELECT, INSERT, UPDATE, DELETE ON Rental\_Car\_DB.insurance TO 'employee'@'localhost';  GRANT SELECT, INSERT, UPDATE, DELETE ON Rental\_Car\_DB.location TO 'employee'@'localhost';  GRANT SELECT, INSERT, UPDATE, DELETE ON Rental\_Car\_DB.model TO 'employee'@'localhost';  GRANT SELECT, INSERT, UPDATE, DELETE ON Rental\_Car\_DB.cars TO 'employee'@'localhost';  GRANT UPDATE ON Rental\_Car\_DB.rental TO 'employee'@'localhost';  FLUSH PRIVILEGES; |

**Trigger**

確保新增租用資料，1.結束租用時間不能晚於開始租用時間 2.確保車輛狀態可借用 3.確保無其他訂單的該汽車時間衝突

|  |
| --- |
| BEGIN  IF DATE(NEW.start\_date) > DATE(NEW.end\_date) THEN  SIGNAL SQLSTATE '45000'  SET MESSAGE\_TEXT = 'begin date must early than end date!!';  END IF;  IF (SELECT car\_status FROM car WHERE car\_id = NEW.car\_id) != 'available' THEN  SIGNAL SQLSTATE '45000'  SET MESSAGE\_TEXT = 'this car cannot rent';  END IF;  IF EXISTS (  SELECT 1 FROM rental  WHERE car\_id = NEW.car\_id  AND rental\_status IN ('pending', 'active')  AND (  DATE(NEW.start\_date) <= DATE\_ADD(DATE(end\_date), INTERVAL 1 DAY)  AND DATE(NEW.end\_date) >= DATE\_SUB(DATE(start\_date), INTERVAL 1 DAY)  )  ) THEN  SIGNAL SQLSTATE '45000'  SET MESSAGE\_TEXT = 'overlapping lease terms';  END IF;  END |

1. 參考文獻
2. [Chen, K.-J. (2023, November 22). PHP接Google第三方登入API最完整與最簡易範例. Medium.](https://medium.com/@newpage0720/php%E6%8E%A5google%E7%AC%AC%E4%B8%89%E6%96%B9%E7%99%BB%E5%85%A5api%E6%9C%80%E5%AE%8C%E6%95%B4%E8%88%87%E6%9C%80%E7%B0%A1%E6%98%93%E7%AF%84%E4%BE%8B-502c5117b078)
3. [Lin, V. (2020, December 18). 透過Google People API取得更多使用者資料. Vocus.](https://vocus.cc/article/5fdc1bc4fd897800016edda7)
4. [常駐程式. (n.d.). 大家遺忘的enum. iT 邦幫忙.](https://ithelp.ithome.com.tw/articles/10184794)