

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 14 日

目錄

壹、	簡介.....	1
貳、	應用情境與使用案例.....	2
參、	系統需求說明.....	3
肆、	概念層模型(完整性限制).....	4
伍、	ER Diagram 及說明	7
	5.1 ER Diagram	7
	5.2 ER Diagram – Relationship	8
	5.3 ER Diagram – Representation in SQL.....	10
	5.4 ER Diagram – Entity.....	17
	5.5 ER Diagram – 合併 Table.....	23
陸、	SQL 與說明	27
柒、	參考文獻.....	35

壹、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。

貳、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。

參、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。

2. 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。

3. 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)。

4. 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。

5. 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。

6. 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。

7. 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。

肆、概念層模型(完整性限制)

1. 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

2. 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

3. 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

4. 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

5. 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

6. 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	是	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

伍、ER Diagram 及說明

5.1 ER Diagram

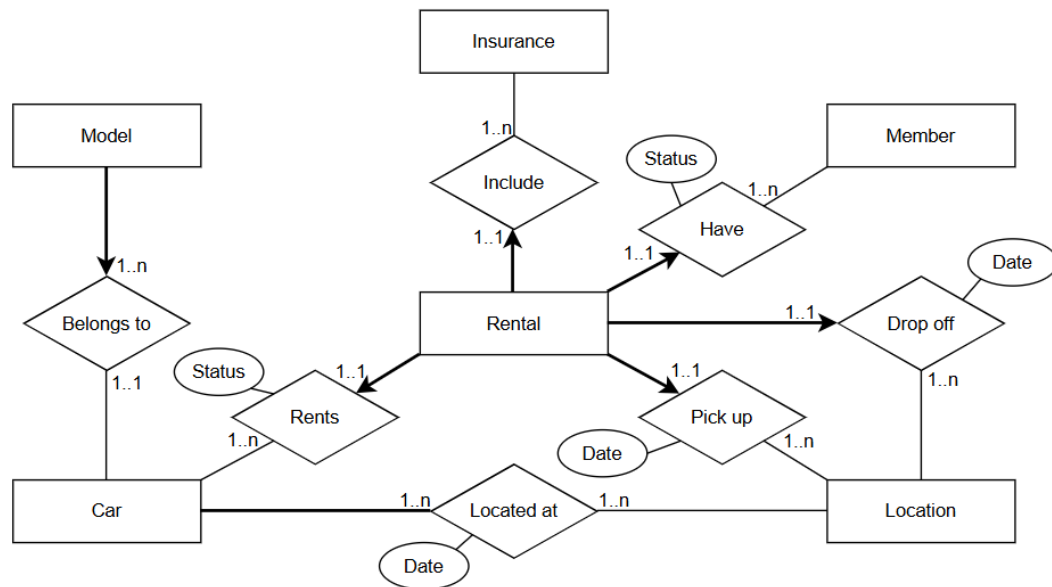


圖 1、詳細 ERD

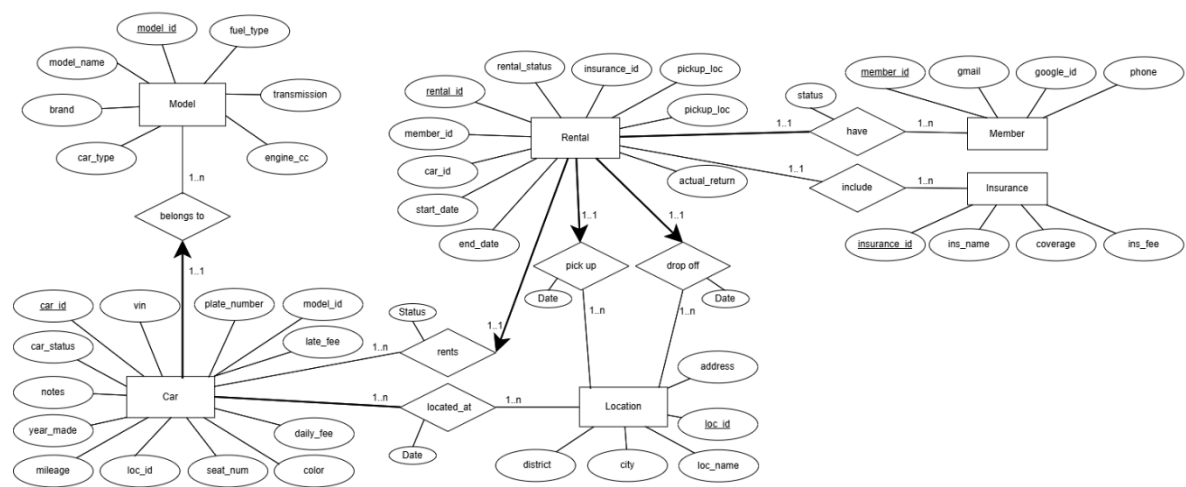


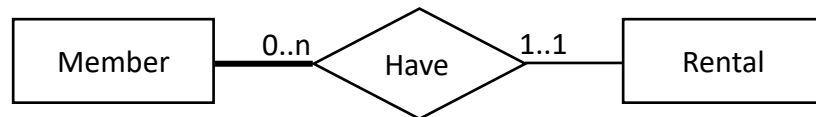
圖 2、完整 ERD

5.2 ER Diagram – Relationship

以下為各實體之間的關聯

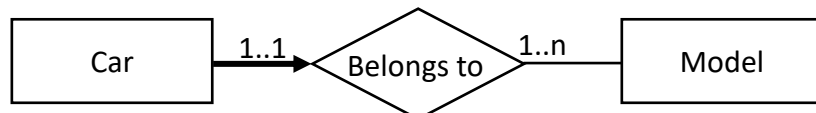
1. Member (會員)實體和 Rental (租賃訂單)實體之間存在一對多的「擁有」關係：

- 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



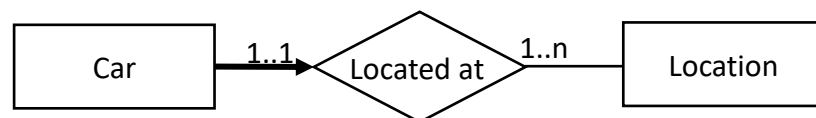
2. Car (汽車)實體和 Model(車型)實體之間存在多對一的「屬於」關係：

- 一個 Model 可以有多輛 Car
- 每輛 Car 只屬於一個 Model



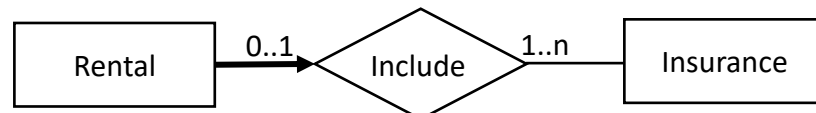
3. Car (汽車)實體和 Location (地點)實體之間存在多對一的「位於」關係：

- 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



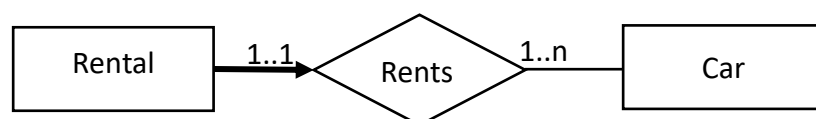
4. Rental (租賃訂單)實體和 Insurance (保險方案)實體之間存在多對一的「包含」關係：

- 每一筆租賃只能包含一個保險方案
- 一個保險方案可以對應到多筆租賃訂單



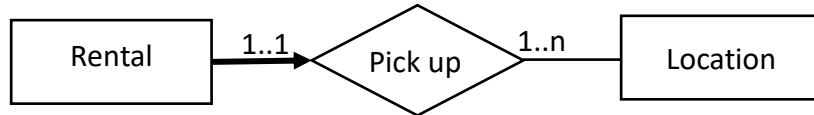
5. Rental (租賃訂單)實體和 Car(汽車)實體之間存在多對一的「租用」關係：

- 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



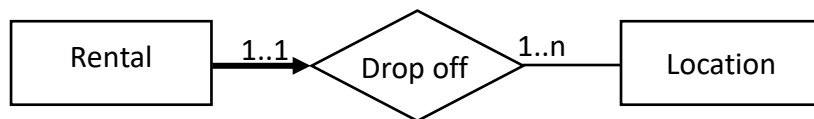
6. Rental (租賃訂單)實體和 Location(地點)實體之間存在多對一的「取車位置」關係：

- 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



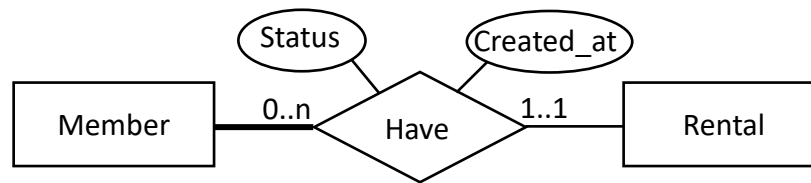
7. Rental (租賃訂單)實體和 Location(地點)實體之間存在多對一的「還車位置」關係：

- 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



5.3 ER Diagram – Representation in SQL

1. Member ↔ Rental



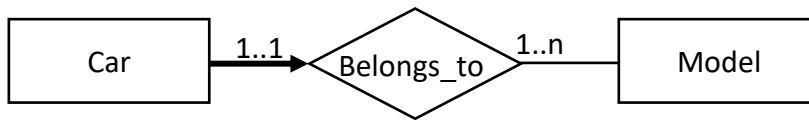
DDL:

```
CREATE TABLE Have (
    Created_at DATETIME,      -- attribute
    Status CHAR (10),        -- attribute
    MemberId INT(11),        -- role
    RentalId INT(11),        -- role
    PRIMARY KEY (RentalId)
    FOREIGN KEY (MemberId) REFERENCES Member(member_id)
    ON DELETE CASCADE,
    FOREIGN KEY (RentalId) REFERENCES Rental(rental_id)
    ON DELETE NO ACTION
)
```

Record:

RentalId	MemberId	Status	Created_at
1	1	active	2025-05-02 18:50:25
2	1	active	2025-04-28 13:20:10
3	2	active	2025-05-01 08:52:14
4	3	active	2025-05-01 18:12:55
5	4	active	2025-05-05 20:12:51

2. Car ↔ Model



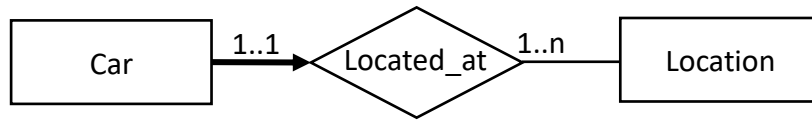
DDL:

```
CREATE TABLE Belongs_to (  
    CarId INT(11),          -- role  
    ModelId INT(11),        -- role  
    PRIMARY KEY (CarId)  
    FOREIGN KEY (CarId) REFERENCES Car(car_id)  
        ON DELETE NO ACTION,  
    FOREIGN KEY (ModelId) REFERENCES Model(model_id)  
        ON DELETE NO ACTION  
)
```

Record:

CarId	ModelId
1	1
2	2
3	5
4	5
5	4

3. Car ↔ Location



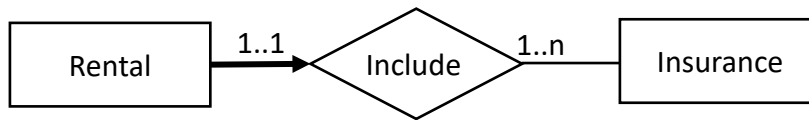
DDL:

```
CREATE TABLE Located_at (  
    CarId INT(11),          -- role  
    LocationId INT(11),     -- role  
    PRIMARY KEY (CarId)  
    FOREIGN KEY (CarId) REFERENCES Car(car_id)  
        ON DELETE NO ACTION,  
    FOREIGN KEY (LocationId) REFERENCES Location(loc_id)  
        ON DELETE SET NULL  
)
```

Record:

CarId	LocationId
1	1
2	1
3	2
4	3
5	4

4. Rental ↔ Insurance



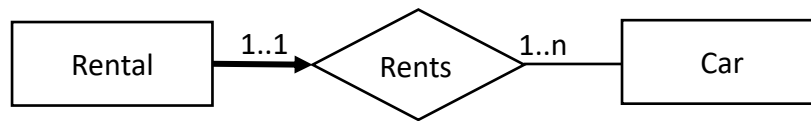
DDL:

```
CREATE TABLE Include (  
    RentalId INT(11),          -- role  
    InsuranceId INT(11),       -- role  
    PRIMARY KEY (RentalId)  
    FOREIGN KEY (RentalId) REFERENCES Rental(rental_id)  
        ON DELETE NO ACTION,  
    FOREIGN KEY (InsuranceId) REFERENCES Insurance(insurance_id)  
        ON DELETE NO ACTION  
)
```

Record:

RentalId	InsuranceId
1	3
2	3
3	1
4	5
5	3

5. Rental ↔ Car



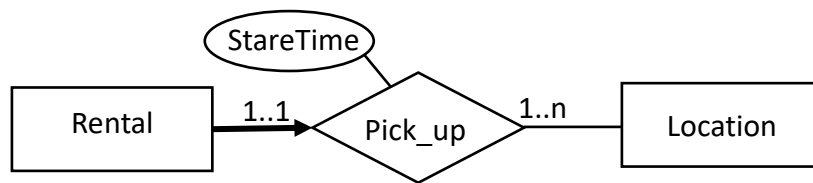
DDL:

```
CREATE TABLE Rents (  
  RentalId INT(11),          -- role  
  CarId INT(11),             -- role  
  PRIMARY KEY (RentalId)  
  FOREIGN KEY (RentalId) REFERENCES Rental(rental_id)  
    ON DELETE NO ACTION,  
  FOREIGN KEY (CarId) REFERENCES Car(car_id)  
    ON DELETE NO ACTION  
)
```

Record:

RentalId	CarId
1	1
2	2
3	1
4	2
5	5

6. Rental ↔ Location



DDL:

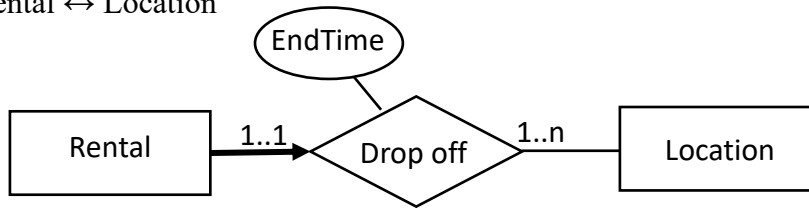
```

CREATE TABLE Pick_up(
    StareTime DATETIME,      -- attribute
    RentalId INT(11),        -- role
    LocationId INT(11),      -- role
    PRIMARY KEY (RentalId)
    FOREIGN KEY (RentalId) REFERENCES Rental(rental_id)
        ON DELETE NO ACTION,
    FOREIGN KEY (LocationId) REFERENCES Location(loc_id)
        ON DELETE NO ACTION
)
    
```

Record:

RentalId	StareTime	LocationId
1	2025-05-04 08:00:00	1
2	2025-05-30 18:00:00	2
3	2025-05-10 12:00:00	1
4	2025-05-15 08:00:00	2
5	2025-05-02 07:30:00	4

7. Rental ↔ Location



DDL:

```

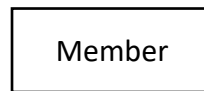
CREATE TABLE Drop off(
    EndTime DATETIME,      -- attribute
    RentalId INT(11),      -- role
    LocationId INT(11),    -- role
    PRIMARY KEY (RentalId)
    FOREIGN KEY (RentalId) REFERENCES Rental(rental_id)
        ON DELETE NO ACTION,
    FOREIGN KEY (LocationId) REFERENCES Location(loc_id)
        ON DELETE NO ACTION
)
    
```

Record:

RentalId	EndTime	LocationId
1	2025-05-07 21:00:00	1
2	2025-06-01 21:00:00	2
3	2025-05-11 21:00:00	4
4	2025-05-17 21:00:00	2
5	2025-05-05 22:00:00	5

5.4 ER Diagram – Entity

1. Member 實體



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

DDL:

```
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,}$'),
    CONSTRAINT chk_phone CHECK (phone REGEXP '^09\d{8}$')
);
```

Record:

member_id	google_id	gmail	phone
1	106455916943881880515	kulpia@gmail.com	0978985123
2	111996782304424104647	gene_st@gmail.com	0914238500
3	192703829478239801723	Wen_Che@gmail.com	0932535148
4	215784230947120938765	asd_tw@gmail.com	0931164655
5	342057390258473869423	kevin_jj@gmail.com	0923969027

2. Model 實體

Model

欄位名稱	資料型態	是否 可為空	Domain
model_id	INT(11)	否	從 1 開始遞增的整數
brand	VARCHAR(30)	否	
model_name	VARCHAR(30)	否	
car_type	VARCHAR(10)	否	限定字串(Compact,Sedan,SUV,MPV)
fuel_type	VARCHAR(10)	否	限定字串(Gasoline,Electric,Hybrid)
engine_cc	INT(5)	否	≥ 0
transmission	INT(1)	否	必須 0 或 1
image_url	VARCHAR(100)	否	圖片路徑位置不得超過 100 字

DDL:

```
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
);
```

Record:

model_id	brand	model_name	car_type	fuel_type	engine_cc	transmission	image_url
1	TOYOTA	Altis 12	Compact	Hybrid	1800	1	img/Altis12.jpg
2	HONDA	Odyssey 2019	MPV	Gasoline	2500	1	img/Odyssey12.jpg
3	TESLA	Model 3	Sedan	Electric	0	1	img/model3.jpg
4	TOYOTA	RAV4 2022	SUV	Hybrid	2500	1	img/rav4_2022.jpg
5	MITSUBISHI	Grand Lancer	Compact	Gasoline	1500	1	img/glancer.jpg

3. Insurance 實體

Insurance

欄位名稱	資料型態	是否可為空	Domain
insurance_id	INT(11)	否	從 1 開始遞增的整數
ins_name	VARCHAR(20)	否	
coverage	VARCHAR(255)	否	
ins_fee	INT(4)	否	≥0

DDL:

```
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)
);
```

Record:

insurance_id	ins_name	car_id	ins_fee
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
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
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
4	全方位保險方案	涵蓋第三人責任升級版保障，並新增車體損失險（含碰撞、自撞、翻車、他人肇逃）、竊盜險。自負額每次事故 5000 元。	800
5	高階全險（免自負額）	等同全方位保險方案，另提供不限事故次數之免自負額保障。亦納入每日車輛停駛補償保險。	1400

4. Location 實體

Location

欄位名稱	資料型態	是否可為空	Domain
loc_id	INT(11)	否	從 1 開始遞增的整數
loc_name	VARCHAR(15)	否	
city	VARCHAR(15)	否	
district	VARCHAR(15)	否	
address	VARCHAR(50)	否	

DDL:

```
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
);
```

Record:

loc_id	loc_name	city	district	address
1	雲林縣虎尾店	雲林縣	虎尾鎮	文化路 64 號
2	雲林縣斗六店	雲林縣	斗六市	大學路三段 123 號
3	嘉義市東區店	東區	嘉義市	學府路 300 號
4	嘉義縣民雄店	民雄鄉	嘉義縣	神農路 145 號
5	雲林縣斗南店	斗南鎮	雲林縣	大業路 106 號

5. Car 實體

Car

欄位名稱	資料型態	是否可為空	Domain
car_id	INT(11)	否	從 1 開始遞增的整數
vin	VARCHAR(50)	否	
plate_number	VARCHAR(8)	否	$^R[A-Z]\{2\}-\backslash d\{4\}\$$
daily_fee	INT(5)	否	≥ 0
late_fee	INT(4)	否	≥ 0

year_made	INT(4)	否	≥1980 (年)
seat_num	INT(2)	否	>0
color	VARCHAR(20)	否	
mileage	INT(6)	否	>0
car_status	ENUM	否	限定字串 (available, maintenance, disable)
notes	TEXT	是	

DDL:

```
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,
);
```

Record:

car_id	vin	plate_number	daily_fee	late_fee		
1	1HGCM82633A123456	RBS-5528	2000	120		
2	4T1BF1FK5DU123456	RBX-1128	2600	150		
3	ML0GLAMN6PF123456	RBP-2236	1500	100		
4	ML0GLAMN4RF654321	RBP-9298	1500	100		
5	2T3BFREV3SW654321	RBZ-2228	2000	120		
year_made	seat_num	color	mileage	notes	car_status	
2022	5	白色	38000		available	
2019	7	白色	25500		available	
2020	5	黑色	30427		available	
2022	5	白色	18512	更換輪胎	maintenance	
2023	5	白色	48512		available	

6. Rental 實體

Rental

欄位名稱	資料型態	是否可為空	Domain
rental_id	INT(11)	否	1 以上整數自動增值
actual_return	DATETIME	是	未歸還時可為 NULL
status	ENUM	否	限定字串 (pending , active ,completed,cancelled, reject)
amount	INT(7)	否	≥0
payment_date	DATETIME	是	YYYY/MM/DD HH:MM:SS
method	ENUM	否	限定字串： (cash , credit , linepay)

DDL:

```
CREATE TABLE rental (
    rental_id INT(11) AUTO_INCREMENT PRIMARY KEY,
    actual_return DATETIME,
    amount INT(7) NOT NULL CHECK (amount >= 0),
    payment_date DATETIME,
    method ENUM('cash', 'credit', 'linepay') NOT NULL,
);
```

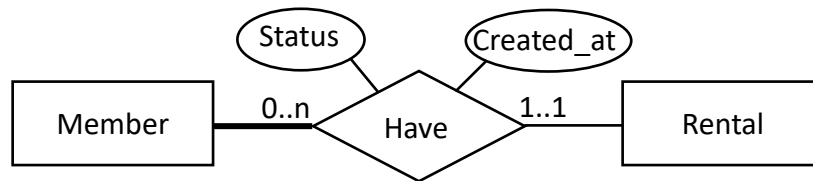
Record:

rental_id	actual_return	amount	method	payment_date
1	2025-05-07 20:53:00	10400	cash	2025-05-04 07:50:00
2	(null)	9000	cash	(null)
3	(null)	4400	credit	2025-05-01 19:20:05
4	(null)	12000	cash	(null)
5	2025-05-05 22:05:00	9600	cash	(null)

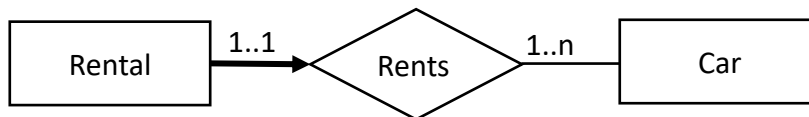
5.5 ER Diagram – 合併 Table

針對於 Rental Table 與多個 relation 有 1..1 的關係，因此合併 Rental Table

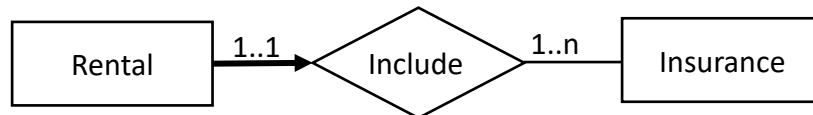
1. Rental & <Have>



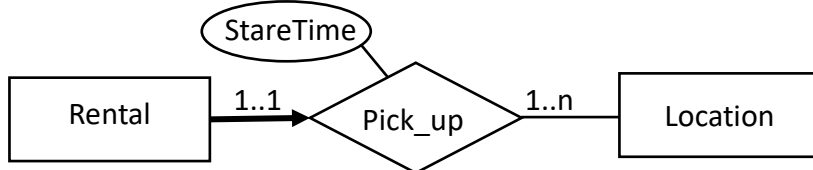
2. Rental & <Rents>



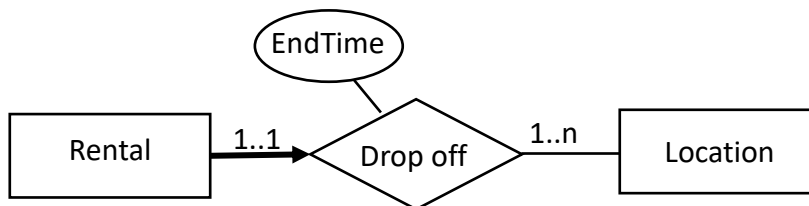
3. Rental & <Include>



4. Rental & <Pick_up>



5. Rental & <Drop_off>



調整後的 DDL:

```
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,
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 SET NULL,
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);

```

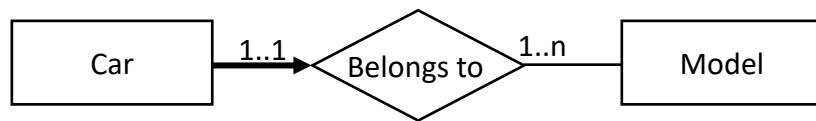
Record:

rental_id	member_id	car_id	start_date	end_date	actual_return	pickup_loc
1	1	1	2025-05-04 08:00:00	2025-05-07 21:00:00	2025-05-07 20:53:00	1
2	1	2	2025-05-30 18:00:00	2025-06-01 21:00:00	(null)	2
3	2	1	2025-05-10 12:00:00	2025-05-11 21:00:00	(null)	1
4	3	2	2025-05-15 08:00:00	2025-05-17 21:00:00	(null)	2
5	4	5	2025-05-02 07:30:00	2025-05-05 22:00:00	2025-05-05 22:05:00	4
drop_loc	insurance_id	rental_status	amount	method	payment_date	created_at
1	3	active	10400	cash	2025-05-04 07:50:00	2025-05-02 18:50:25
2	3	active	9000	cash	(null)	2025-04-28 13:20:10
4	1	active	4400	credit	2025-05-01 19:20:05	2025-05-01 08:52:14

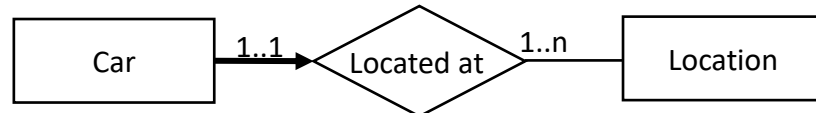
2	5	active	12000	cash	(null)	2025-05-01 18:12:55
5	3	active	9600	cash	(null)	2025-05-05 20:12:51

針對於 Car Table 與<Belongs_to>有 1..1 的關係以及與<Located at>有 1..1 的關係，因此將<Belongs_to>和<Located at>合併到 Car Table

1. Car & <Belongs to>



2. Car & <Located at>



調整後的 DDL:

```

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 );
  
```

Record:

car_id	vin		plate_number		model_id	daily_fee	late_fee
1	1HGCM82633A123456		RBS-5528		1	2000	120
2	4T1BF1FK5DU123456		RBX-1128		2	2600	150
3	ML0GLAMN6PF123456		RBP-2236		5	1500	100
4	ML0GLAMN4RF654321		RBP-9298		5	1500	100
5	2T3BFREV3SW654321		RBZ-2228		4	2000	120
year_made		loc_id	seat_num	color	mileage	notes	car_status
2022		1	5	白色	38000		available
2019		1	7	白色	25500		available
2020		2	5	黑色	30427		available
2022		3	5	白色	18512		maintenance
2023		4	5	白色	48512		available

陸、SQL 與說明

1. 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。

2. 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 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 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 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 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,
    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';

-- INSERT DATA
-- INSERT member DATA
INSERT INTO member (
        member_id, google_id, gmail, phone
) VALUES
(1, '106455916943881880515', 'kulpia@gmail.com', '0978985123'),
(2, '111996782304424104647', 'gene_st@gmail.com', '0914238500'),
(3, '192703829478239801723', 'Wen_Che@gmail.com', '0932535148'),
(4, '215784230947120938765', 'asd_tw@gmail.com', '0931164655'),
(5, '342057390258473869423', 'kevin_jj@gmail.com', '0923969027');

-- INSERT location Data
INSERT INTO location (loc_id, loc_name, city, district, address) VALUES
(1, '雲林縣虎尾店', '雲林縣', '虎尾鎮', '文化路 64 號'),
(2, '雲林縣斗六店', '雲林縣', '斗六市', '大學路三段 123 號'),
(3, '嘉義市東區店', '嘉義市', '東區', '學府路 300 號'),
(4, '嘉義縣民雄店', '嘉義縣', '民雄鄉', '神農路 145 號'),
(5, '雲林縣斗南店', '雲林縣', '斗南鎮', '大業路 106 號');

-- INSERT insurance Data
INSERT INTO insurance (insurance_id, ins_name, coverage, ins_fee) VALUES
(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),

```


(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),
 (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),
 (4, '全方位保險方案', '涵蓋第三人責任升級版保障，並新增車體損失險（含碰撞、自撞、翻車、他人肇逃）、竊盜險。自負額每次事故 5000 元。', 800),
 (5, '高階全險(免自負額)', '等同全方位保險方案，另提供不限事故次數之免自負額保障。亦納入每日車輛停駛補償保險。', 1400);

-- INSERT model Data

```
INSERT INTO model (
    model_id, brand, model_name, car_type, fuel_type, engine_cc, transmission,
    image_url
) VALUES
(1, 'TOYOTA', 'Altis 12', 'Compact', 'Hybrid', 1800, 1, 'img/Altis12.jpg'),
(2, 'HONDA', 'Odyssey 2019', 'MPV', 'Gasoline', 2500, 1, 'img/Odyssey12.jpg'),
(3, 'TESLA', 'Model 3', 'Sedan', 'Electric', 0, 1, 'img/model3.jpg'),
(4, 'TOYOTA', 'RAV4 2022', 'SUV', 'Hybrid', 2500, 1, 'img/rav4_2022.jpg'),
(5, 'MITSUBISHI', 'Grand Lancer', 'Compact', 'Gasoline', 1500, 1,
'img/glancer.jpg');
```

-- Insert car Data

```
INSERT INTO cars (
    car_id, vin, plate_number, model_id, daily_fee, late_fee, year_made, loc_id,
    seat_num, color, mileage, notes, car_status
) VALUES
(1, '1HGCM82633A123456', 'RBS-5528', 1, 2000, 120, 2022, 1, 5, '白色', 38000, "",
'available'),
(2, '4T1BF1FK5DU123456', 'RBX-1128', 2, 2600, 150, 2019, 1, 7, '白色', 25500, "",
'available'),
(3, 'ML0GLAMN6PF123456', 'RBP-2236', 5, 1500, 100, 2020, 2, 5, '黑色', 30427,
'', 'available'),
(4, 'ML0GLAMN4RF654321', 'RBP-9298', 5, 1500, 100, 2022, 3, 5, '白色', 18512,
'更換輪胎', 'maintenance'),
(5, '2T3BFREV3SW654321', 'RBZ-2228', 4, 2000, 120, 2023, 4, 5, '白色', 48512, "",
'available');
```

-- Insert rental Data

```

INSERT INTO rental (
    rental_id, member_id, car_id, start_date, end_date, actual_return,
    pickup_loc, drop_loc, insurance_id, rental_status, amount,
    method, payment_date, created_at
) VALUES
(1, 1, 1, '2025-05-04 08:00:00', '2025-05-07 21:00:00', '2025-05-07 20:53:00',
1, 1, 3, 'active', 10400, 'cash', '2025-05-04 07:50:00', '2025-05-02 18:50:25'),

(2, 1, 2, '2025-05-30 18:00:00', '2025-06-01 21:00:00', NULL,
2, 2, 3, 'active', 9000, 'cash', NULL, '2025-04-28 13:20:10'),

(3, 2, 1, '2025-05-10 12:00:00', '2025-05-11 21:00:00', NULL,
1, 4, 1, 'active', 4400, 'credit', '2025-05-01 19:20:05', '2025-05-01 08:52:14'),

(4, 3, 2, '2025-05-15 08:00:00', '2025-05-17 21:00:00', NULL,
2, 2, 5, 'active', 12000, 'cash', NULL, '2025-05-01 18:12:55'),

(5, 4, 5, '2025-05-02 07:30:00', '2025-05-05 22:00:00', '2025-05-05 22:05:00',
4, 5, 3, 'active', 9600, 'cash', NULL, '2025-05-05 20:12:51');

-- 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
```

柒、參考文獻

- [1]. 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.
- [2]. Lin, V. (2020, December 18). 透過 Google People API 取得更多使用者資料. Vocus.
- [3]. 常駐程式. (n.d.). 大家遺忘的 enum. iT 邦幫忙.