博客來 2024 年度百大暢銷榜

資料庫結構分析與建置

博客來

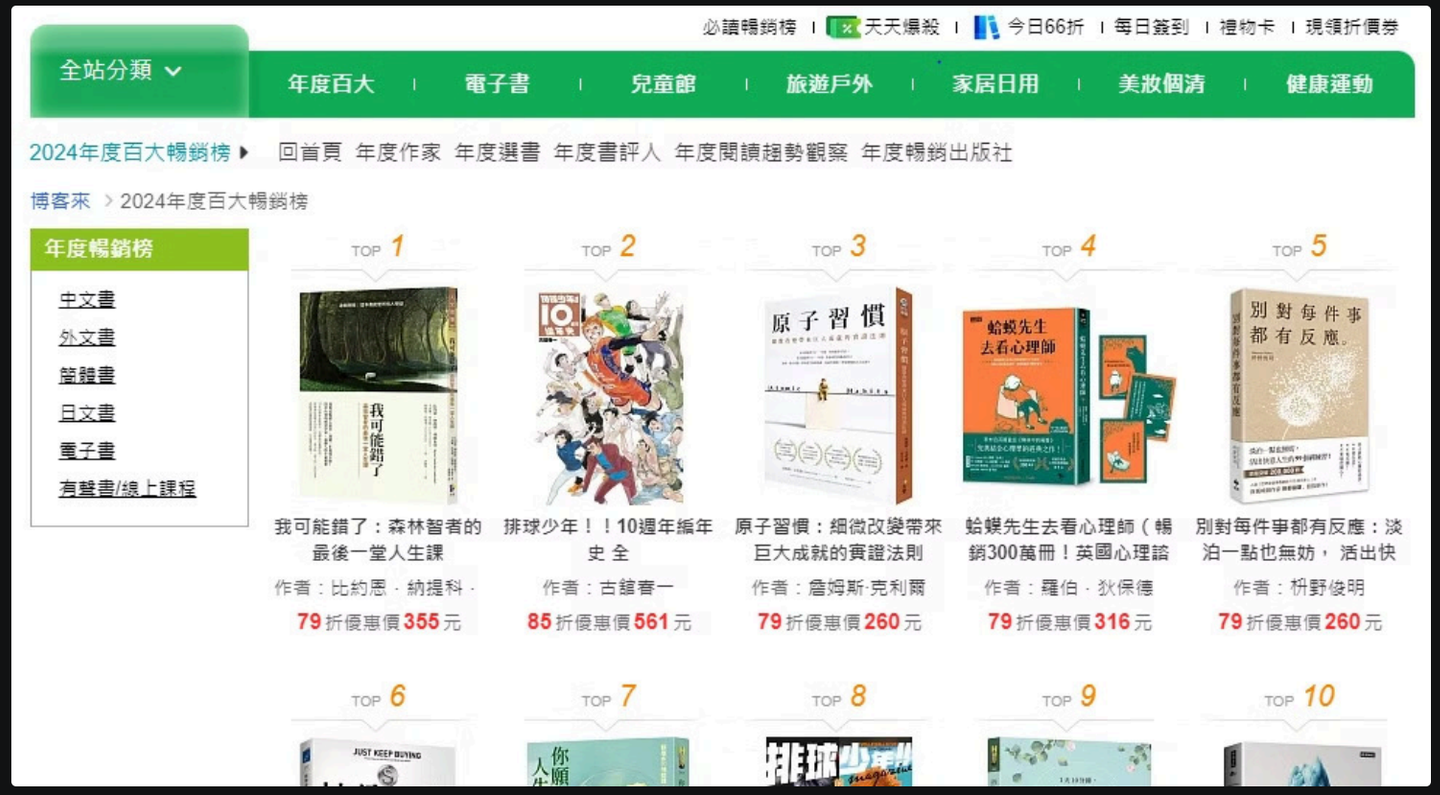
博客來2024年度百大暢銷榜

博客來年度暢銷榜公布！2024年最熱賣的暢銷TOP1是...？從閱讀動態裡觀察大眾的需求

第 3 組

DATE : 09 / 12 / 2024

1. TABLE 結構



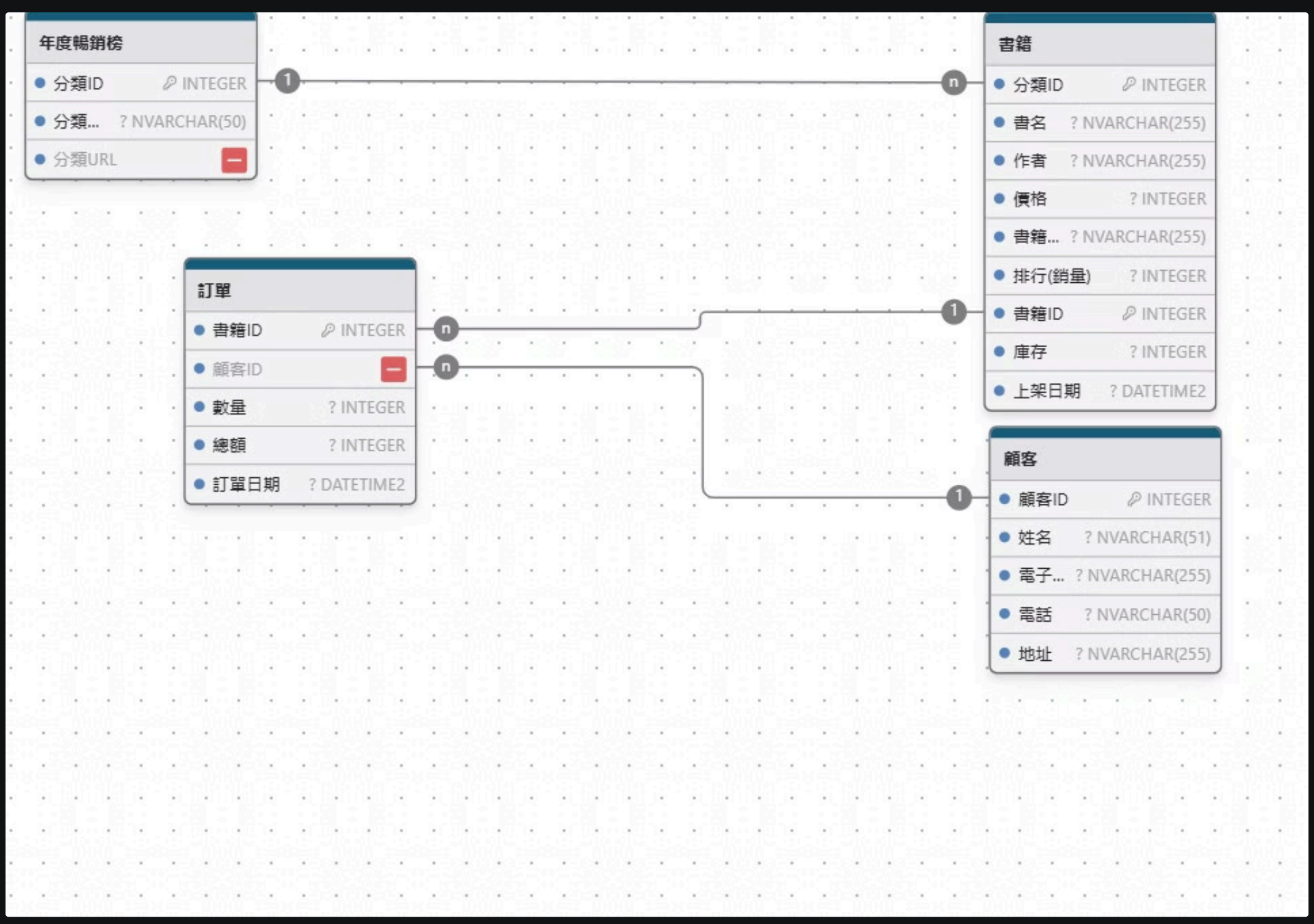
a. 年度暢銷榜

b. 書籍

c. 顧客

d. 訂單

2. ER 圖表示



DB 最終版本

-- 刪除 "訂單" 表格 DROP TABLE IF EXISTS [訂單]; GO

-- 刪除 "顧客" 表格 DROP TABLE IF EXISTS [顧客]; GO

-- 刪除 "書籍" 表格 DROP TABLE IF EXISTS [書籍]; GO

-- 刪除 "年度暢銷榜" 表格 DROP TABLE IF EXISTS [年度暢銷榜]; GO

-- 創建 "年度暢銷榜" 表格 CREATE TABLE [年度暢銷榜] ([分類ID] INTEGER NOT NULL IDENTITY UNIQUE, [分類名稱] NVARCHAR(255), -- 更新長度為 255 [分類URL] NVARCHAR(255), -- 更新長度為 255 PRIMARY KEY([分類ID])); GO

-- 創建 "書籍" 表格 CREATE TABLE [書籍] ([分類ID] INTEGER NOT NULL, [書名] NVARCHAR(255), -- 更新長度為 255 [作者] NVARCHAR(255), -- 更新長度為 255 [價格] INTEGER, [書籍URL] NVARCHAR(255), -- 更新長度為 255 [排行(銷量)] INTEGER, [書籍ID] NVARCHAR(255), -- 更新長度為 255 [庫存] INTEGER, -- 庫存設為整數型別 [上架日期] NVARCHAR(255), -- 更新長度為 255 PRIMARY KEY([書籍ID]), -- 書籍ID 作為主鍵 FOREIGN KEY([分類ID]) REFERENCES **年度暢銷榜** -- 參照年度暢銷榜中的分類ID); GO

-- 創建 "顧客" 表格 CREATE TABLE [顧客] ([顧客ID] INTEGER NOT NULL IDENTITY UNIQUE, [姓名] NVARCHAR(255), -- 更新長度為 255 [電子郵件] NVARCHAR(255), -- 更新長度為 255 [電話] NVARCHAR(255), -- 更新長度為 255 [地址] NVARCHAR(255), -- 更新長度為 255 PRIMARY KEY([顧客ID])); GO

-- 創建 "訂單" 表格 CREATE TABLE [訂單] ([書籍ID] NVARCHAR(255) NOT NULL, -- 書籍ID 作為外鍵 [顧客ID] INTEGER NOT NULL, -- 顧客ID 作為外鍵 [數量] INTEGER, [總額] INTEGER, [訂單日期] DATETIME2, PRIMARY KEY([書籍ID], [顧客ID]), -- 訂單表的複合主鍵 FOREIGN KEY([顧客ID]) REFERENCES **顧客**, -- 參照顧客表格的顧客ID FOREIGN KEY([書籍ID]) REFERENCES **書籍** -- 參照書籍表格的書籍ID); GO

-- 插入資料到 "年度暢銷榜" 表格 INSERT INTO [年度暢銷榜] ([分類名稱], [分類URL]) VALUES ('中文書', '**https://www.books.com.tw/web/annual100_cat/01?loc=P_0004_001'**), ('外文書', '**https://www.books.com.tw/web/annual100_cat/02?loc=P_0004_002'**); GO

-- 插入資料到 "書籍" 表格 INSERT INTO [書籍] ([分類ID], [書名], [作者], [價格], [書籍URL], [排行(銷量)], [書籍ID], [庫存]) VALUES (1, '我可能錯了：森林智者的最後一堂人生課', '比約恩・納提科・林德布勞,卡洛琳・班克勒,納維德・莫迪里', 355, '**https://www.books.com.tw/products/0010947051?loc=P_0019_001'**, 3000, '0010947051', 30), (2, '排球少年！！10週年編年史 全', '古舘春一', 561, '**https://www.books.com.tw/products/0010984893?loc=P_0019_002'**, 2000, '0010984893', 30); GO

-- 插入資料到 "顧客" 表格 INSERT INTO [顧客] ([姓名], [電子郵件], [電話], [地址]) VALUES ('張三', '**zhangsan@example.com**', '0912345678', '台北市中山區'), ('李四', '**lisi@example.com**', '0923456789', '台北市大安區'); GO

-- 插入資料到 "訂單" 表格 INSERT INTO [訂單] ([顧客ID], [書籍ID], [數量], [總額], [訂單日期]) VALUES (1, '0010947051', 2, 710, GETDATE()), (2, '0010984893', 1, 561, GETDATE()); GO

-- 更新所有書籍的庫存數量為 30 UPDATE 書籍 SET 庫存 = 30; GO

3. SQL Schema 定義

a. TABLE: 年度暢銷榜

```
CREATE TABLE [年度暢銷榜] (

    [分類ID] INTEGER NOT NULL IDENTITY UNIQUE,
    [分類名稱] NVARCHAR(255),  -- 更新長度為 255
    [分類URL] NVARCHAR(255),   -- 更新長度為 255
    PRIMARY KEY([分類ID])

);
GO
```

- **分類ID**：唯一標識每個分類的 ID。
- **分類名稱**：分類的名稱。
- **分類URL**：分類的 URL。

3. SQL Schema 定義

b. TABLE: 書籍

```
CREATE TABLE [書籍] (

    [分類ID] INTEGER NOT NULL,
    [書名] NVARCHAR(255),      -- 更新長度為 255
    [作者] NVARCHAR(255),      -- 更新長度為 255
    [價格] INTEGER,
    [書籍URL] NVARCHAR(255),  -- 更新長度為 255
    [排行(銷量)] INTEGER,
    [書籍ID] NVARCHAR(255),   -- 更新長度為 255
    [庫存] INTEGER,           -- 庫存設為整數型別
    [上架日期] NVARCHAR(255), -- 更新長度為 255
    PRIMARY KEY([書籍ID]),    -- 書籍ID 作為主鍵
    FOREIGN KEY([分類ID]) REFERENCES [年度暢銷榜]([分類ID])  -- 參照年度暢銷榜中的分類ID

);
GO
```

- **分類 ID**：書籍所屬的分類 ID。
- **書名**：書籍的名稱。
- **作者**：書籍的作者。
- **價格**：書籍的價格。
- **書籍URL**：書籍的 URL。
- **排行(銷量)**：書籍的銷量排名。
- **書籍ID**：唯一標識每本書的 ID。
- **庫存**：書籍的庫存量。
- **上架日期**：書籍的上架日期。

3. SQL Schema 定義

c. TABLE: 顧客

```
CREATE TABLE [顧客] (

    [顧客ID] INTEGER NOT NULL IDENTITY UNIQUE,
    [姓名] NVARCHAR(255),     -- 更新長度為 255
    [電子郵件] NVARCHAR(255), -- 更新長度為 255
    [電話] NVARCHAR(255),     -- 更新長度為 255
    [地址] NVARCHAR(255),     -- 更新長度為 255
    PRIMARY KEY([顧客ID])

);
GO
```

- **顧客ID**：唯一標識每個顧客的 ID。
- **姓名**：顧客的姓名。
- **電子郵件**：顧客的電子郵件地址。
- **電話**：顧客的電話號碼。
- **地址**：顧客的地址。

3. SQL Schema 定義

d. TABLE: 訂單

```
CREATE TABLE [訂單] (

    [書籍ID] NVARCHAR(255) NOT NULL,  -- 書籍ID 作為外鍵
    [顧客ID] INTEGER NOT NULL,        -- 顧客ID 作為外鍵
    [數量] INTEGER,
    [總額] INTEGER,
    [訂單日期] DATETIME2,
    PRIMARY KEY([書籍ID], [顧客ID]),  -- 訂單表的複合主鍵
    FOREIGN KEY([顧客ID]) REFERENCES [顧客]([顧客ID]),  -- 參照顧客表格的顧客ID
    FOREIGN KEY([書籍ID]) REFERENCES [書籍]([書籍ID])   -- 參照書籍表格的書籍ID

);
GO
```

- **書籍ID**：書籍的唯一標識 ID。
- **顧客ID**：顧客的唯一標識 ID。
- **數量**：訂購的書籍數量。
- **總額**：訂單的總金額。
- **訂單日期**：訂單的日期。

4. SQL Procedure

a. 將訂單總額 由高至低排序（宜貞）

-- 創建儲存過程，來查詢訂單詳細資料

CREATE PROCEDURE 查詢訂單詳細資料

AS

BEGIN

SELECT

顧客.姓名 AS 顧客姓名,

書籍.書名 AS 書名,

訂單.數量 AS 訂單數量,

訂單.總額 AS 訂單總額,

訂單.訂單日期 AS 訂單日期

FROM

訂單

JOIN

顧客 ON 訂單.顧客ID = 顧客.顧客ID

JOIN

書籍 ON 訂單.書籍ID = 書籍.書籍ID

ORDER BY

訂單.總額 DESC;

END;

GO

EXEC 查詢訂單詳細資料;

b. 查詢分類中的暢銷排行前5名 最高排到最低 （廷偉）

create PROCEDURE GetTopBooks2

@CategoryID INT

AS

BEGIN

SELECT TOP 5

書籍.分類ID,

書籍.[排行(銷量)],

書籍.書名,

書籍.作者,

書籍.價格,

年度暢銷榜.分類名稱

FROM 書籍

LEFT JOIN 年度暢銷榜

ON 書籍.分類ID = 年度暢銷榜.分類ID

WHERE 書籍.分類ID = @CategoryID

ORDER BY 書籍.[排行(銷量)] DESC;

END;

EXEC GetTopBooks2 @CategoryID = 1 ;

c. 變更特別書籍的價格折扣（家瑜）

DROP PROCEDURE IF EXISTS update_discount;

GO

CREATE PROCEDURE update_discount

@id INT,

@discount float

AS

BEGIN

BEGIN TRY

UPDATE 書籍

SET

價格 = 價格 * @discount

WHERE 書籍ID = @id;

END TRY

BEGIN CATCH

PRINT 'Error occurred during the update.';

THROW;

END CATCH

END;

GO

/*EXEC update_discount

@id ='0010947051',

@discount = 0.7;

select * from 書籍

*/

d. 查詢一個月內所有一百名外文書的訂單（鉦暘）

CREATE PROCEDURE [dbo].[checkbuyrankontime]

@booktype int

AS

BEGIN

select * from 訂單

left join 書籍 on 書籍.書籍ID = 訂單.書籍ID

left join 年度暢銷榜 on 年度暢銷榜.分類ID = 書籍.分類ID

where 訂單.訂單日期 >= DATEADD(DAY, -30, GETDATE())

and 書籍.分類ID = @booktype

END

exec checkbuyrankontime 2

e. 創建儲存過程：購買書籍並更新銷量和庫存（暐人）

CREATE PROCEDURE 庫存銷量管理 @顧客ID INTEGER, -- 顧客 ID @書籍ID NVARCHAR(255), -- 書籍 ID @數量 INTEGER -- 購買數量 AS BEGIN -- 檢查書籍庫存是否足夠 IF EXISTS (SELECT 1 FROM 書籍 WHERE 書籍ID = @書籍ID AND CAST(庫存 AS INT) >= @數量) BEGIN -- 如果訂單已存在，更新數量和總額 IF EXISTS (SELECT 1 FROM 訂單 WHERE 顧客ID = @顧客ID AND 書籍ID = @書籍ID) BEGIN UPDATE 訂單 SET 數量 = 數量 + @數量, 總額 = 總額 + (@數量 * 書籍.價格) FROM 訂單 INNER JOIN 書籍 ON 訂單.書籍ID = 書籍.書籍ID WHERE 顧客ID = @顧客ID AND 訂單.書籍ID = @書籍ID; END ELSE BEGIN -- 如果訂單不存在，插入新記錄 INSERT INTO 訂單 (顧客ID, 書籍ID, 數量, 總額, 訂單日期) SELECT @顧客ID, @書籍ID, @數量, @數量 * 價格, GETDATE() FROM 書籍 WHERE 書籍ID = @書籍ID; END;

```
    -- 更新書籍的銷量（增加購買數量）
    UPDATE 書籍
    SET [排行(銷量)] = [排行(銷量)] + @數量
    WHERE 書籍ID = @書籍ID;

    -- 更新書籍的庫存（減少購買數量）
    UPDATE 書籍
    SET 庫存 = CAST(庫存 AS INT) - @數量
    WHERE 書籍ID = @書籍ID;
END
ELSE
BEGIN
    -- 如果庫存不足，提示錯誤訊息
    --THROW 50000, '庫存不足，無法完成訂單。', 1;
    SELECT '庫存不足' AS message;
END
```

END; GO

```
EXEC 庫存銷量管理 @顧客ID = 1, @書籍ID = '0010947051', @數量 = 3;
```

f. 查詢暢銷大於 2500 （儀君）

DROP PROCEDURE IF EXISTS GetSales2500;

Go

CREATE PROCEDURE GetSales2500

AS

BEGIN

SELECT *

FROM [書籍]

WHERE [排行(銷量)]> 2500;

END;

-- 使用儲存程序查詢銷售金額超過 2500 的資料

EXEC GetSales2500;

g. 同作者作品查詢（孟旅）

CREATE PROCEDURE [dbo].[此作者作品查詢]

@author NVARCHAR(200) -- 定義輸入參數，用來接收使用者提供的作者名稱

AS

BEGIN

-- 查詢 Book 資料表中屬於該作者的書籍

SELECT

書籍ID, -- 書籍唯一編號

作者, -- 作者名

書名, -- 書名

[排行(銷量)], -- 排行(銷量)

價格 -- 價格

FROM 書籍

WHERE 作者 = @author -- 條件：書籍分類等於輸入的分類名稱

END

exec 此作者作品查詢 @author='古7尋一'

05. Feedback

老師：少做了一個功能導致我們沒有完成簡報主題，且符合排行榜要求。

此查詢會按書籍的銷售數量從高到低進行排序：

```
SELECT
    b.[書籍ID],
    b.[書名],
    SUM(o.[數量]) AS [總銷售數量],
    b.[作者],
    b.[價格],
    b.[書籍URL],
    b.[排行(銷量)]  -- 顯示更新後的排行(銷量)欄位
FROM
    [訂單] o
JOIN
    [書籍] b ON o.[書籍ID] = b.[書籍ID]
GROUP BY
    b.[書籍ID], b.[書名], b.[作者], b.[價格], b.[書籍URL], b.[排行(銷量)]
ORDER BY
    [總銷售數量] DESC;  -- 按銷售數量排序
GO
```

說明

1. **選擇欄位**：
   - `b.[書籍ID]`：書籍的唯一標識 ID。
   - `b.[書名]`：書籍名稱。
   - `SUM(o.[數量]) AS [總銷售數量]`：總銷售數量，匯總每本書的銷售數量。
   - `b.[作者]`：書籍作者。
   - `b.[價格]`：書籍價格。
   - `b.[書籍URL]`：書籍的 URL。
   - `b.[排行(銷量)]`：書籍的銷量排名。
2. **關聯**：
   - `JOIN [訂單] o ON o.[書籍ID] = b.[書籍ID]`：將 `訂單` 表與 `書籍` 表根據書籍ID進行關聯。
3. **分組**：
   - 根據書籍的各個屬性進行分組以進行匯總計算。
4. **排序**：
   - `ORDER BY [總銷售數量] DESC`：按總銷售數量從高到低排序。

這個查詢可以幫助你生成基於銷售數據的排行榜，按書籍的銷售數量進行排序。