



# 建立資料清單



designed by freepik

Estimated time:

50 min.

 資訊工業策進會 Institute for Information Industry

【Key Points】：

# 學習目標

- 13-1: 顯示單頁資料
- 13-2: 取得分頁資料
- 13-3: 完成分頁功能



13-1

 財團法人資訊工業策進會  
INSTITUTE FOR INFORMATION INDUSTRY

本篇將介紹如何將資料庫的資料呈現於HTML頁面之中，並且學習如何使用 `ejs` 來方便程式碼資料渲染，還有實作分頁的功能，具體內容如下：

- 建立專案
- 匯入口罩庫存範例資料
- 以SELECT查詢資料
- 使用EJS處理HTML畫面渲染
- MySQL 的 LIMIT 功能
- SQL LIMIT 語法示範
- 在GET /page 路由傳遞頁數
- 情境範例—分頁資料
- 製作分頁頁碼
- 查詢總筆數與頁數計算
- 製作分頁頁碼
- 情境範例 – 分頁功能完成

## 【Key Points】：

將資料庫的資料呈現於HTML頁面之中  
學習如何使用 `ejs` 來方便程式碼資料渲染  
實作分頁功能

## 13-1: 取得單頁資料

- 建立專案
- 匯入口罩庫存範例資料
- 以SELECT查詢資料
- 使用EJS處理HTML畫面渲染



designed by freepik



designed by freepik

13-2



在這個章節，我們將先建立專案，並且匯入預先準備好的口罩庫存資料，實作呈現資料畫面給 Client。

- 初始化專案
- 安裝套件: `npm install express mysql`
- 啟動 Apache 與 MySQL
- 匯入口罩庫存範例資料
- 寫一個get方法，當Client訪問時，使用 `query("Select * from inventory;")`，搜尋全部的資料
- 查詢結果暫時先顯示於終端機視窗
- 使用EJS處理HTML畫面渲染

### 【Key Points】：

匯入口罩庫存範例資料

使用 `query("Select * from inventory;")`，搜尋全部的資料

使用EJS處理HTML畫面渲染

## 建立專案

1. 在「命令提示字元」，輸入：「**mkdir render\_page**」 建立資料夾，當做我們專案的工作目錄。
2. 繼續輸入：「**cd render\_page**」 切換目錄
3. 輸入：「**npm init -f**」 初始化專案
4. 用 **VS Code** 編輯我們的專案資料夾
5. 在 **VS Code** 按下「**Ctrl + 滑鼠右鍵**」開啟 terminal 終端機視窗。
6. 在終端機視窗，安裝套件: **npm install express mysql**
7. 在 **VS Code**，新增 **index.js** (本專案的主程式)
8. 啟動 **XAMPP Control Pane**  
( 在檔案總管點兩下 **c:\xampp\xampp-control.exe** )
9. 點按 **Apache** 與 **MySQL** 的「**Start**」按鈕，啟動兩套伺服器

13-3



1. 在「命令提示字元」，輸入：「**mkdir render\_page**」 建立資料夾，當做我們專案的工作目錄。
2. 繼續輸入：「**cd render\_page**」 切換到該目錄
3. 輸入：「**npm init -f**」 初始化專案
4. 用 **VS Code** 編輯我們的專案資料夾
5. 在 **VS Code** 按下「**Ctrl + 滑鼠右鍵**」開啟 terminal 終端機視窗。
6. 在終端機視窗，安裝套件: **npm install express mysql**
7. 在 **VS Code**，新增 **index.js** (本專案的主程式)
8. 啟動 **XAMPP Control Pane** ( 在檔案總管點兩下 **c:\xampp\xampp-control.exe** )
9. 點按 **Apache** 與 **MySQL** 的「**Start**」按鈕，啟動兩套伺服器

### 【Key Points】：

初始化專案

安裝套件: **npm install express mysql**

啟動 **Apache** 與 **MySQL**

## 匯入口罩庫存範例資料

1. 在檔案總管點兩下c:\xampp\xampp-control.exe，啟動 XAMPP Control Panel
2. 點按 Apache 與MySQL的「Start」按鈕，啟動兩套伺服器
3. 點按MySQL那列的「Admin」按鈕，啟動phpMyAdmin 管理程式
4. 切換到 SQL 頁籤
5. 複製貼入 mask.sql 內容到SQL 頁籤，然後按下「執行」按鈕。  
(mask.sql位於本模組 example 資料夾)

13-4



1. 在檔案總管點兩下c:\xampp\xampp-control.exe，啟動 XAMPP Control Panel
2. 點按 Apache 與MySQL的「Start」按鈕，啟動兩套伺服器
3. 點按MySQL那列的「Admin」按鈕，啟動phpMyAdmin 管理程式
4. 切換到 SQL 頁籤
5. 複製貼入 mask.sql 內容到SQL 頁籤，然後按下「執行」按鈕。

Note: mask.sql位於本模組 example 資料夾。

### 【Key Points】：

啟動 XAMPP Control Panel

切換到 SQL 頁籤

執行 mask.sql 內的程式

# 以SELECT查詢資料

- 開啟 index.js
- require("mysql")引用 mysql 模組
- 設定連線參數
- 呼叫 connect() 方法連線資料庫伺服器
- connection.query('select...', ...)查詢所有資料並顯示在終端機視窗

```
var express = require('express');
var mysql= require('mysql');
var app = express();

const connection = mysql.createConnection({
  host: '127.0.0.1',
  user: 'root',
  password: 'root', //預設phpmyadmin密碼是空值
  database: "Mask" //指定剛刷新的Mask資料庫
})
//連線
connection.connect(function(error) {
  if (error) {
    console.log('連線失敗');
    return;
  }
  console.log('連線成功');
});
app.get('/', function(req, res){
  //在inventory資料表中，查詢所有欄位的資料
  connection.query('SELECT * FROM inventory;', function(error, rows) {
    if (error) {
      console.log(error);
    }
    var data = rows;
    console.log(data)
  });
  res.end()
});
app.listen(3000);
```

13-5



- 開啟 render\_page 資料夾的 index.js
- require("name") 引用 express、mysql模組
- 設定與 MySQL 連線的參數 ( 帳號、密碼、資料庫 ) // XAMPP 預設的密碼是空字串
- 呼叫 connect() 方法連線資料庫伺服器
- 寫一個get方法，當Client訪問時，使用 query("Select \* from inventory;") ，搜尋全部的資料
- 查詢結果顯示於終端機視窗
- 在 VS Code 按下「Ctrl + 撇號」開啟終端機視窗，輸入node index.js
- 打開瀏覽器訪問<http://localhost:3000>

## 【Key Points】：

設定與 MySQL 連線的參數  
寫一個get方法，當Client訪問時，使用 query("Select \* from inventory;") ，搜尋全部的資料  
查詢結果暫時先顯示於終端機視窗

# 使用EJS處理HTML畫面渲染

- EJS是什麼?

- EJS 是 Express 的樣板引擎，用來渲染HTML頁面
  - 類似 php，`<% if(0.1+0.2==0.3)%>`、`<%= name %>`、`<%- name %>`

- 為什麼要使用 EJS?

- `res.sendFile`只能返回靜態HTML
  - 方便後端動態產生內容給Client端

- 安裝EJS

- `npm install ejs`

```
C:\Users\Administrator\Desktop\13_Ing\render_page>npm install ejs
> ejs@3.0.1 postinstall C:\Users\Administrator\Desktop\13_Ing\render_page\node_modules\ejs
> node ./postinstall.js

Thank you for installing EJS: built with the Jake JavaScript build tool (https://jakejs.com/)
npm WARN render_page@1.0.0 No description
npm WARN render_page@1.0.0 No repository field.

+ ejs@3.0.1
added 1 package from 1 contributor and audited 140 packages in 2.22s
found 0 vulnerabilities
```

13-6



EJS是什麼?

1. 是JavaScript的一個樣板引擎，來在HTML中嵌入JavaScript程式碼，再以程式渲染HTML內容。
2. 類似php用法，EJS在`<% %>`置入JavaScript程式碼
3. Express將會直接執行，並顯示出結果
4. `<%= name%>`即顯示替換過 HTML 特殊字符的內容，例如「小於」轉成「&lt;」。可避免惡意JavaScript程式碼執行於Client端。
5. `<%- name%>`即顯示原始 HTML 的內容

為什麼要使用 EJS?

因為`res.sendFile`將會直接返回固定內容的HTML檔給Client，如果要隨著資料庫內容更改，最好的方法就是內嵌程式邏輯，經過express處理且譯成HTML檔案返回給Client

安裝 EJS:

在 VS Code 按下「Ctrl + 滑鼠右鍵」開啟終端機視窗，輸入 `npm install ejs`

【Key Points】：

EJS是什麼?

為什麼要使用 EJS?

安裝 EJS: `npm install ejs`

# 使用EJS處理HTML畫面渲染

- 使用方式

- 在express的view engine引入ejs
- 以 res.render('檔名', 資料) 渲染
- 新增 views 資料夾
- views下新增 index.ejs

```
var app = express();
//express使用ejs作為模版引擎
app.set('view engine', 'ejs');

app.get('/', function(req, res){
  //在inventory資料表中，查詢所有欄位的資料
  connection.query(' SELECT * FROM inventory', function(error, rows) {
    if (error) {
      console.log(error);
    }
    var data = rows;
    res.render('index',{name: '我是EJS'});
  });
});
```

ejs檔名 要傳遞的參數

13-7



1. 設定 EJS 為 express 使用的樣版引擎: app.set('engine view', 'ejs')
2. res.render()括號中第一個參數是指定ejs檔為視圖輸出，本例是使用index.ejs
3. 第二個參數以{}刮起來，把想傳遞的值輸入進去，比如說定義 name:"我是EJS"
4. 預設的視圖路徑將會在該目錄下的views，所以，請在專案目錄新增 views 資料夾
5. 並在views資料夾新增 index.ejs，後來撰寫，輸入HTML:5，按Tab，更改title標籤為EJS，並在body標籤內打上<%= name%>

## 【Key Points】：

res.render()括號中第一個參數是指定ejs檔為視圖輸出  
第二個參數以{}刮起來，把想傳遞的值輸入進去  
預設的視圖路徑將會在該目錄下的views

# 使用EJS處理HTML畫面渲染

- **查看成果**

- node index.js

- 開啟<http://localhost:3000>

- **將資料顯示在HTML**

- 更改index.js

- 更改index.ejs(迴圈)

```
app.get('/', function(req, res){  
    //在inventory資料表中，查詢所有欄位的資料  
    connection.query('SELECT * FROM inventory;', function(error, rows) {  
        if (error) {  
            console.log(error);  
        }  
        var data = rows;  
  
        res.render('index', {data: data});  
    });  
})
```

```
<html lang="en">  
<head>  
    <meta charset="UTF-8">  
    <meta name="viewport" content="width=device-width, initial-scale=1.0">  
    <title>EJS</title>  
</head>  
<body>  
    <table>  
        <thead>  
            <th>#</th>  
            <th>名稱</th>  
            <th>電話</th>  
            <th>地址</th>  
            <th>成功口罩庫存</th>  
            <th>孩童口罩庫存</th>  
        </thead>  
        <tbody>  
            <% data.forEach((item ,index) => { %>  
                <tr>  
                    <td><%= index+1%></td>  
                    <td><%= item.name%></td>  
                    <td><%= item.phone%></td>  
                    <td><%= item.address%></td>  
                    <td><%= item.adult_mask%></td>  
                    <td><%= item.child_mask%></td>  
                </tr>  
            <% }) %>  
        </tbody>  
    </table>  
</body>  
</html>
```

13-8



1. 在 VS Code 按下「Ctrl + 撇號」開啟 terminal 終端機視窗，輸入:node index.js
2. 以瀏覽器訪問 <http://localhost:3000>
3. 就會看到name直接輸出在頁面上
4. 修改 index.js，將 console.log() 改成 res.render('index', {data: data})
5. 修改 index.ejs，用一個 table 的各個 <tr> 收納 data 陣列。換句話說，在 tbody 標籤內用 data.forEach()，將陣列的每筆資料輸出來。
6. 重新執行 index.js，再以瀏覽器訪問 <http://localhost:3000> 一次。

## 【Key Points】：

將 console.log() 改成 res.render('index', {data: data})

用 data.forEach()，將陣列的每筆資料輸出來

在 VS Code 按下「Ctrl + 撇號」開啟 terminal 終端機視窗，輸入:node index.js

## 13-2：取得分頁資料

- MySQL 的 LIMIT 功能
- SQL LIMIT 語法示範
- 在GET /page 路由傳遞頁數
- 情境範例一分頁資料



designed by freepik



designed by freepik

13-9

 財團法人資訊工業策進會  
INSTITUTE FOR INFORMATION INDUSTRY

在這個章節，將說明使用SQL的Limit來實現分頁功能。

- 為什麼要使用 LIMIT 子句
- 說明 SQL LIMIT 語法並且示範
- 在GET /page 路由傳遞頁數
- 製作分頁頁碼
- 完成情境範例的分頁功能

### 【Key Points】：

說明 SQL LIMIT 語法並且示範  
在GET /page 路由傳遞頁數  
製作分頁頁碼

# MySQL 的 LIMIT 功能

- 為什麼要使用 LIMIT 子句
  - 讓 MySQL 查詢返回特定數量和區段的資料
  - 避免查詢大量資料導致系統過慢
- 用法
  - 第一個數指定返回記錄行的偏移量（跳過幾筆），第二個數指定返回記錄行的最大數目
  - `select * from `inventory` Limit 0, 10`  
(從第 0 筆算), (返回 10 筆資料)
- 注意
  - `LIMIT n` 等同於 `LIMIT 0,n`

13-10



1. 如果一個網頁資料量有到上萬筆的話，沒有經過分頁處理一次性返回，會造成系統負擔
2. 使用 limit 可以返回特定數量的資料，比如說每一頁返回 10 筆
3. 並且利用偏移量可以返回 1~10 筆、11~20 筆，逐段分離出各頁資料
4. Limit a, b，第一個數以 0 為基準，代表從第 a+1 筆開始算，返回 b 筆資料
5. Limit n == Limit 0, n，代表從第一筆開始算，返回 n 筆資料

## 【Key Points】：

為什麼要使用 LIMIT 子句

Limit a, b，第一個數以 0 為基準，代表從第 a+1 筆開始算，返回 b 筆資料

Limit n == Limit 0, n，代表從第一筆開始算，返回 n 筆資料

# SQL LIMIT 語法示範

- 範例 - SQL語句執行

- 打開phpmyadmin的SQL
- `Select * from `inventory``
- `Select * from `inventory` Limit 0, 10`
- `Select * from `inventory` Limit 10`
- `Select * from `inventory` Limit 10, 10`

✓ 顯示第 0 - 9 列 (總計 10 筆, 查詢用了 0.0014 秒。)

```
Select * from `inventory` Limit 0, 10
```

✓ 顯示第 0 - 9 列 (總計 10 筆, 查詢用了 0.0012 秒。)

```
Select * from `inventory` Limit 10
```

✓ 顯示第 0 - 24 列 (總計 28 筆, 查詢用了 0.0019 秒。)

```
SELECT * FROM `inventory`
```

✓ 顯示第 10 - 19 列 (總計 10 筆, 查詢用了 0.0011 秒。)

```
Select * from `inventory` Limit 10, 10
```

13-11



- 在檔案總管點兩下c:\xampp\xampp-control.exe，啟動 XAMPP Control Panel
- 點按 Apache 與MySQL的「Start」按鈕，啟動兩套伺服器
- 點按MySQL那列的「Admin」按鈕，啟動phpMyAdmin 管理程式
- 先點一下Mask(切換到 Mask 資料庫)，再點按SQL頁籤。準備進行SQL語法演練...
- 先從撈出全部資料開始，`Select * from `inventory``，可以看到返回總數量，第0-24列
- `Select * from `inventory` Limit 0, 10` 和 `Select * from `inventory` Limit 10` 兩個語句一樣，但是還是來操作看看，返回的是第0-9列
- `Select * from `inventory` Limit 10, 10`，返回10-19列

## 【Key Points】：

啟動phpMyAdmin 管理程式

先點一下Mask(切換到 Mask 資料庫)，再點按SQL頁籤

`Select * from `inventory` Limit 10, 10`，返回10-19列

# 在GET /page 路由傳遞頁數

- 使用/page路由判斷頁數

- :page([0-9]+) 代表0-9至少出現一次
  - page<=0 自動導到/page/1路由
  - 定義每頁資料量
  - 更改SQL語句

```
//在inventory資料表中，查詢分頁資料
app.get('/page/:page([0-9]+)', function(req, res){
  var page = req.params.page
  //把<=0的1強制改成1
  if(page <= 0 ) {
    res.redirect('/page/1')
  }
  //每頁資料數
  var nums_per_page = 10
  //定義資料偏移量
  var offset = (page - 1) * nums_per_page

  connection.query(`SELECT * FROM inventory LIMIT ${offset}, ${nums_per_page}`, function(error, data) {
    if (error) {
      console.log(error);
    }
    res.render('index',{data: data})
  });
})
```

13-12



1. 開啟 index.js，撰寫新的路由 app.get("/page", ...)
2. 並且在路由中，準備一個 params 以備後來傳遞頁數: app.get("/page/:page", ...)
3. 定義page，並做正則表示式，0-9至少出現一次，如果傳遞<=0的值，全部都轉向/page/1路由  
寫好的程式類似這樣: app.get("/page/:page([0-9]+)", ...)
4. 定義每頁返回數量，var nums\_per\_page = 10;
5. 定義偏移量 ( limit是由0為基準，所以第一頁要先用 -1 )，並且乘上每頁數量
6. 全部代入 connection.query("select ...", ...) 就完成了。

## 【Key Points】：

撰寫新的路由app.get("/page/:page([0-9]+)", ...)

定義每頁返回數量

定義偏移量 ( limit是由0為基準，所以第一頁要先用 -1 )，並且乘上每頁數量

# 情境範例一分頁資料

- 查看成果

- node index.js
- 開啟http://localhost:3000/page/1
- 開啟http://localhost:3000/page/2
- 開啟http://localhost:3000/page/3

#	名稱	電話	地址	成功口罩庫存	孩童口罩庫存
1	賴川藥局	(02)296606	新北市板橋區民權路202巷4弄1號	0	458
2	順儀民權藥局	(02)296095	新北市板橋區民權路234號1樓	0	2065
3	大昌藥局	(02)896225	新北市板橋區北門街28號	0	925
4	瑞明藥局	(02)227284	新北市板橋區文化路1段17號	0	6
5	常仁藥局	(02)296749	新北市板橋區中正路1巷4號	549	696
6	日生大藥局	(02)296752	新北市板橋區中正路1巷16弄16號	0	2096
7	一誠藥局	(02)296739	新北市板橋區府中路62號	0	1096
8	中山大藥局	(02)295750	新北市板橋區中山路1段107號	552	1451
9	華傳綜合大藥局	(02)296700	新北市板橋區大智街40號	546	195
10	順健府中藥局	(02)295411	新北市板橋區中山路1段96號(1樓)	0	1402

#	名稱	電話	地址	成功口罩庫存	孩童口罩庫存
1	賴川藥局	(02)296606	新北市板橋區民權路202巷4弄1號	0	458
2	順儀民權藥局	(02)296095	新北市板橋區民權路234號1樓	0	2065
3	大昌藥局	(02)896225	新北市板橋區北門街28號	0	925
4	瑞明藥局	(02)227284	新北市板橋區文化路1段17號	0	6
5	常仁藥局	(02)296749	新北市板橋區中正路1巷4號	549	696
6	日生大藥局	(02)296752	新北市板橋區中正路1巷16弄16號	0	2096
7	一誠藥局	(02)296739	新北市板橋區府中路62號	0	1096
8	中山大藥局	(02)295750	新北市板橋區中山路1段107號	552	1451
9	華傳綜合大藥局	(02)296700	新北市板橋區大智街40號	546	195
10	順健府中藥局	(02)295411	新北市板橋區中山路1段96號(1樓)	0	1402

#	名稱	電話	地址	成功口罩庫存	孩童口罩庫存
1	國光藥師藥局	(02)296982	新北市板橋區國光路172號	2	1011
2	台安藥局	(02)296636	新北市板橋區中正路204號1樓、2樓、3樓	3	0
3	景好藥師藥局	(02)272747	新北市板橋區國光路182號1樓	0	1417
4	利安藥師藥局	(02)296669	新北市板橋區南雅西路2段117號及119號1樓	507	668
5	醫安藥局	(02)296925	新北市板橋區南雅南路1段11號1樓及2樓	398	1436
6	翰林藥局	(02)296027	新北市板橋區中正路240號	21	36
7	快安藥局	(02)296035	新北市板橋區漢生西路31號1樓	1	2279
8	可安藥局	(02)296501	新北市板橋區南雅南路1段8之6號(1樓)	492	1865

13-13

III 財團法人資訊工業策進會  
INSTITUTE FOR INFORMATION INDUSTRY

- 在 VS Code 按下「Ctrl + 滑鼠」開啟 terminal 終端機視窗
- 輸入node index.js 啟動我們用 Node.js 寫的網站伺服器
- 用瀏覽器連線 http://localhost:3000/page/1
- 用瀏覽器連線 http://localhost:3000/page/2
- 用瀏覽器連線 http://localhost:3000/page/3

<Note>以「http://localhost:3000/page/頁次編號」進行測試

## 【Key Points】：

- 開啟 terminal 終端機視窗
- 啟動我們用 Node.js 寫的網站伺服器
- 以「http://localhost:3000/page/頁次編號」進行測試

## 13-3：完成分頁功能

- 製作分頁頁碼
- 查詢總筆數與頁數計算
- 製作分頁頁碼
- 情境範例 – 分頁功能完成



designed by freepik



designed by freepik

13-14

 財團法人資訊工業策進會  
INSTITUTE FOR INFORMATION INDUSTRY

接下來的這個知識點，將把前一個知識點的分頁功能做得更齊全：

- 精準掌握所有筆數以及總共幾頁，才能夠做得出分頁功能給使用者操作
- 查詢總筆數與頁數計算
- 製作分頁頁碼
- 追蹤目前頁數
- 控管邏輯，讓頁次不會超過最後一頁
- 情境範例 – 分頁功能完成

### 【Key Points】：

精準掌握所有筆數以及總共幾頁

製作分頁頁碼

追蹤目前頁數，控管邏輯，讓頁次不會超過最後一頁。

# 查詢總筆數與頁數計算

- 打開 index.js
- 新增查詢總數SQL語句
  - SELECT COUNT(\*)**
  - AS COUNT FROM inventory**
- 避免page超過last\_page

```
//在inventory資料表中，查詢分頁資料
app.get('/page/:page(10-91+)', function(req, res){
  var page = req.params.page
  //把<=0的id強制改成1
  if(page <= 0 ) {
    res.redirect('/page/1')
  }
  //每頁資料數
  var nums_per_page = 10
  //定義資料庫連接
  connection.query('SELECT * FROM inventory LIMIT ${offset}, ${nums_per_page}', function(error, data) {
    if (error) {
      console.log(error);
    }
    connection.query('SELECT COUNT(*) AS COUNT FROM inventory', function(error, nums) {
      if (error) {
        console.log(error);
      }
      var last_page = Math.ceil(nums[0].COUNT / nums_per_page)

      //避免請求超過最大頁數
      if(page > last_page) {
        res.redirect('/page/' + last_page)
      }

      res.render('page',{
        data: data,
        curr_page: page,
        //本頁資料數量
        total_nums: nums[0].COUNT,
        //總數除以每頁筆數，再無條件取整數
        last_page: last_page
      })
    });
  });
});
```

13-15



精準掌握所有筆數以及總共幾頁，才能夠做得出分頁功能給使用者操作：

- 打開index.js
- 做完分頁功能後，要詳細掌握所有筆數，總共幾頁，才能夠顯示於頁面上讓使用者操作
- 所以直接用COUNT(\*) 來計算筆數，但是COUNT(\*)在返回的results中變數獲取不到
- 要使用AS COUNT把COUNT(\*)換個名稱，並使用object方式獲取
- 查詢完後定義last\_page，判斷page是否大於last\_page，如果是就把它轉向最大頁數的/page/last\_page路由

【Key Points】：

精準掌握所有筆數以及總共幾頁，才能夠做得出分頁功能給使用者操作

用COUNT(\*) 來計算筆數

判斷page是否大於last\_page，如果是就把它轉向最大頁數的/page/last\_page路由

# 製作分頁頁碼

- render內index改成page

- 重要參數

- data
- curr\_page
- total\_nums
- last\_page

```
//在inventory資料表中，查詢分頁資料
app.get('/page/:page([0-9]+)', function(req, res){
  var page = req.params.page
  //把<=0的1強制改成1
  if(page <= 0) {
    res.redirect('/page/1')
  }
  //每頁資料數
  var nums_per_page = 10
  //定義資料偏移量
  var offset = (page - 1) * nums_per_page
  connection.query('SELECT * FROM inventory LIMIT ${offset}, ${nums_per_page}', function(error, data) {
    if (error) {
      console.log(error);
    }
    connection.query('SELECT COUNT(*) AS COUNT FROM inventory', function(error, nums) {
      if (error) {
        console.log(error);
      }
      var last_page = Math.ceil(nums[0].COUNT / nums_per_page)

      //避免請求超過最大頁數
      if(page > last_page) {
        res.redirect('/page/'+last_page)
      }

      res.render('page',{
        data: data,
        curr_page: page,
        //本頁資料數量
        total_nums: nums[0].COUNT,
        //總數除以每頁筆數，再無條件取整數
        last_page: last_page
      })
    })
  });
});
```

13-16



1. render( ) 內的 index 模板改成 page
2. data是原本資料，保持不動
3. curr\_page就是現在的頁數
4. total\_nums則是所有筆數
5. last\_page，即最後一頁，總筆數 / 單頁筆數，並無條件進位

## 【Key Points】：

render( ) 內的 index 模板改成 page

curr\_page就是現在的頁數

last\_page，即最後一頁，總筆數 / 單頁筆數，並無條件進位

# 製作分頁頁碼

- 新增page.ejs
- 複製index.ejs的HTML，在tbody後新增tfoot標籤
- 迴圈輸出所有頁數
- 顯示資料狀態

```
<tfoot>
  <tr>
    <td colspan="4" style="text-align: center;">
      <% for(var i = 1; i <= last_page; i++) { %>
        <span ><a href="/page/<%= i%>"><%= i%></a></span>
      <% } %>
    </td><p>總數 <%= total_nums%> 築，共 <%= curr_page%>/<%= last_page%> 頁</p></td>
  </tr>
</tfoot>
```

13-17



1. 新增一個 page.ejs
2. 複製 index.ejs 的程式碼過來
3. 並且在 tbody下新增 tfoot 標籤
4. 以迴圈把頁數超連結逐個顯示一遍
5. 在最後顯示整個資料總數，總頁數

## 【Key Points】：

新增一個 page.ejs,複製 index.ejs 的程式碼過來  
在 tbody下新增 tfoot 標籤，以迴圈把頁數超連結逐個顯示一遍  
最後顯示整個資料總數，總頁數

# 情境範例 – 分頁功能完成

- 查看成果

- node index.js

- 開啟http://localhost:3000/page/

#	名稱	電話	地址	成功口罩庫存	孩童口罩庫存
1	穎川藥局	(02)296606	新北市板橋區民權路202巷4弄1號	0	458
2	順儻民權藥局	(02)296095	新北市板橋區民權路234號1樓	0	2065
3	大昌藥局	(02)896825	新北市板橋區北門街28號	0	925
4	瑞明藥局	(02)227284	新北市板橋區文化路1段17號	0	6
5	常仁藥局	(02)296749	新北市板橋區中正路1巷4號	549	696
6	日生大藥局	(02)296752	新北市板橋區中正路1巷16弄16號	0	2096
7	一誠藥局	(02)296739	新北市板橋區府中路62號	0	1096
8	中山大藥局	(02)295750	新北市板橋區中山路1段107號	552	1451
9	華傳綜合大藥局	(02)296700	新北市板橋區大智街40號	546	195
10	順儻府中藥局	(02)295411	新北市板橋區中山路1段96號（1樓）	0	1402

1 2 3

總數 28 筆，共 1/3 頁

13-18

 財團法人資訊工業策進會  
INSTITUTE FOR INFORMATION INDUSTRY

1. 在 VS Code 按下「Ctrl + 撇號」開啟 terminal 終端機視窗
2. 輸入 node index.js 啟動伺服器
3. 以瀏覽器連線 http://localhost:3000/page/
4. 檢視與測試分頁功能
5. 不用增加新的路由來更動頁面，直接使用超連結傳參數來進行換頁

【Key Points】：

node index.js 啟動伺服器

以瀏覽器連線 http://localhost:3000/page/

檢視與測試分頁功能

# Summary < 精華回顧 >

- **phpMyAdmin** 匯入口罩庫存範例資料
- 使用 **EJS** 當作 **express** 的樣板引擎
- 介紹 **EJS** 基本用法
- 使用 **LIMIT** 實現資料區段
- 增加頁碼，完成分頁功能



13-19

 財團法人資訊工業策進會  
INSTITUTE FOR INFORMATION INDUSTRY

回顧一下前面的內容

- 一開始，我們建立專案，匯入口罩庫存範例資料。
- 接下來，以SELECT查詢資料，使用EJS處理HTML畫面渲染。
- 透過 MySQL SELECT 敘述的 LIMIT 功能，避免大量加載資料，並更有效率將資料做區段輸出。
- 在GET /page 路由傳遞頁數來顯示各頁資料。
- 使用頁碼優化整個分頁功能，使得Client不用再手動修改URL也能切換頁面

【Key Points】：

使用 EJS 當作 express 的樣板引擎

使用 LIMIT 實現資料區段

增加頁碼，完成分頁功能

# 線上程式題

- **13-1 如何使用 EJS 處理 HTML 畫面渲染**

想要使用 EJS 處理 HTML 畫面渲染，接下來的程式要怎麼寫？

- **13-2 傳遞資料到視圖(View)**

如何將變數藉由EJS傳遞data到我們的index視圖上？

- **13-3 完成分頁功能**

有一段程式的內容如下：

```
app.get('/page/:page([0-9]+)', function(req, res){
```

```
    ...
```

```
})
```

如何以6筆資料為一頁的方式，分頁呈現資料內容。

13-20



題目名稱: 13-1 如何使用 EJS 處理 HTML 畫面渲染

想要使用 EJS 處理 HTML 畫面渲染，接下來的程式要怎麼寫？

題目名稱: 13-2 傳遞資料到視圖(View)

有一段程式的內容如下：

```
app.get('/', function(req, res){
```

```
    //...
```

```
});
```

如何將變數藉由EJS傳遞data到我們的index視圖上？

題目名稱: 13-3 完成分頁功能

有一段程式的內容如下：

```
app.get('/page/:page([0-9]+)', function(req, res){
```

```
    ...
```

```
})
```

如何以6筆資料為一頁的方式，分頁呈現資料內容。

【Key Points】：

13-1 如何使用 EJS 處理 HTML 畫面渲染

13-2 傳遞資料到視圖(View)

13-3 完成分頁功能

# 課後練習題(Lab)

- **情節描述:**  
本練習假設Node.js已經安裝於Windows作業系統並且已安裝好XAMPP，也假設您已學過如何安裝與使用EJS。另外，在開始進行練習之前，請先將建立課程的Mask資料庫，並匯入資料。
- **預設目標:**  
學會如何設計具有資料分頁功能的程式。
- **Lab00: 設定資料庫資料結構與測試資料**
- **Lab01: 分頁功能改善**
- **Lab02: 分頁功能(使用id)**

Estimated time:  
**30 minutes**

13-21



## 【情節描述】

本練習假設Node.js已經安裝於Windows作業系統並且已安裝好XAMPP，也假設您已學過如何安裝與使用EJS。另外，在開始進行練習之前，請先將建立課程的Mask資料庫，並匯入資料。

## 【預設目標】

學會如何設計具有資料分頁功能的程式。

Lab00: 設定資料庫資料結構與測試資料

Lab01: 分頁功能改善

Lab02: 分頁功能(使用id)

完成後的程式與檔案，請參考 Example 資料夾的內容。

## 【Key Points】：

Lab00: 設定資料庫資料結構與測試資料

Lab01: 分頁功能改善

Lab02: 分頁功能(使用id)

# 範例程式使用說明

- 範例程式資料夾: Module\_13\_example
- 使用步驟:
  1. 安裝 Node.js
  2. 安裝 Visual Studio Code
  3. 安裝 XAMPP, 匯入 mask 口罩庫存資料庫
  4. 以 Visual Studio Code 開啟本模組的範例資料夾
  5. 在 Visual Studio Code 按下「Ctrl + 滑鼠」開啟 terminal 終端機視窗。
  6. 在終端機視窗，輸入下列指令，安裝必要的模組套件:  
npm install
  7. 在終端機視窗，輸入 node index.js
  8. 啟動瀏覽器，連接 http://localhost:3000/page/1

13-22



使用步驟:

1. 安裝 Node.js (<https://nodejs.org/en/>)
2. 安裝 Visual Studio Code (<https://code.visualstudio.com/>)
3. 安裝 XAMPP ([https://www.apachefriends.org/zh\\_tw/index.html](https://www.apachefriends.org/zh_tw/index.html))
4. 在檔案總管點兩下c:\xampp\xampp-control.exe，啟動 XAMPP Control Panel
5. 點按 Apache 與MySQL的「Start」按鈕，啟動兩套伺服器
6. 點按MySQL那列的「Admin」按鈕，啟動phpMyAdmin 管理程式，切換到 SQL 頁籤
7. 複製貼入 mask.sql 內容到SQL 頁籤，然後按下「執行」按鈕。
8. 以 Visual Studio Code 開啟範例資料夾  
在檔案總管，滑鼠右鍵點按「本範例資料夾」，從快捷功能表點選「以Code開啟」  
或者，啟動 Visual Studio Code 之後，功能表 File | Open Folder，選擇「本範例資料夾」
9. 在 Visual Studio Code 按下「Ctrl + 滑鼠」開啟 terminal 終端機視窗。
10. 在終端機視窗，輸入下列指令，安裝必要的模組套件:  
npm install
11. 在終端機視窗，輸入 node index.js
12. 啟動瀏覽器，連接 http://localhost:3000/page/1

【Key Points】：

程式執行環境 Node.js

程式開發編輯環境: Visual Studio Code

在 Visual Studio Code 按下「Ctrl + 滑鼠」開啟 terminal 終端機視窗，輸入：「node 主程式.js」