

序

Oracle 資料庫是一套可擴展、可靠性與安全性高，且效能優於同類型產品，具有用戶定義資料類型、繼承和多型等物件導向功能的一種物件關聯式資料庫管理系統（ORDBMS），廣受許多中大企業與機關之青睞。雖然學習 Oracle 資料庫 SQL 會比其他資料庫稍難，但學會 Oracle 資料庫 SQL 之後，可以無痛很快地學會其他資料庫 SQL。此外，就軟體取得的難度而言，只要到 Oracle 官網註冊，便可免費下載安裝做為個人學習、測試與研究等非商業用途。綜此，相較之下，直接選擇學習 Oracle 資料庫 SQL，絕對是一種低成本且高效益之自我投資。

本書編撰目的有二：一、提供大專校院開設實務性資料庫相關，例如資料庫概論、資料庫管理系統等類課程一個學期所需之授課內容，學生完成學習後，可具備完成資料庫設計與應用小專題之能力；二、針對有意參加 Oracle SQL 1Z0-071 認證之讀者，本書參照其認證技術範疇，於第二章至十三章之內容，提供最佳中文學習教材。本書是學習 Oracle 資料庫 SQL 必備的工具書，內容循序漸進，各章節均有實際應用例題暨解析，助益學習成效。

為挹注教學，本書另提供教學投影片、各章課後習題（1Z0-071 認證考試類型相同的選擇題，以及實作題）、習題解答，採用本書授課教師可向基峰業務索取。

本書之編撰雖力求完美，但疏漏或繆誤在所難免，歡迎讀者與先進不吝指正。

辜輝超 謹誌

2021 年 8 月於松山

縮小資料檢索範圍 與排序結果

基於資料庫儲存的資料數量龐大，所以當用 `SELECT` 語句從資料庫檢索資料時，必然想只撈到所要的資料就好了，此時須在 `SELECT` 語句中使用 `WHERE` 子句，加入條件來縮小檢索範圍。另外，從資料庫撈取資料之後，有時會想要把撈出來的資料列依照某個或某些資料欄位加以排序，此時就須在 `SELECT` 語句中加上 `ORDER BY` 子句，排序檢索結果之資料列。

本章首先介紹 `WHERE` 子句，接著介紹當 `WHERE` 子句的條件判斷式遭遇空值 (Null Value) 之處理，以及如何在 `WHERE` 子句中用 `LIKE` 運算符搜尋字串暨如何使用 `ORDER BY` 子句排序檢索資料列之規定。最後並介紹 12c 及往後版本新增的 `FETCH` 子句，用以限制檢索返回資料列數量。

3.1 使用 `WHERE` 子句縮小檢索範圍

一、`WHERE` 子句之格式

在 `SELECT` 命令中使用 `WHERE` 子句，加上條件過濾資料列，以縮小檢索範圍，格式如下：

```
SELECT *|{[DISTINCT] 欄位 | 表達式 [別名],...}  
FROM 資料表  
[WHERE 條件];
```

說明：


1. WHERE 須寫在 FROM 之後。
2. 條件 (Conditions)：可由數個欄位、表達式、常數組成。如果是複合條件，則條件與條件間須再用 AND (及) 或 OR (或) 等邏輯運算符連結。
3. 條件式的運算結果為 TRUE、FALSE 或 UNKNOWN。當條件運算結果為 TRUE 時，對應的資料列就會被檢索。

二、條件運算式之比較運算符

WHERE 子句的條件式之比較運算符包括：等於 (=)、大於 (>)、小於 (<)、大於等於 (>=)、不等於 (<>) ... 等等，彙整如表 3.1。

表 3.1 條件運算之比較運算符彙整表

運算符	意義
=	等於
>	大於
>=	大於等於
<	小於
<=	小於等於
<>	不等
BETWEEN A AND B	>=A，且<= B(介於 A 與 B 之間)
IN(集合)	判斷是否符合集合內之值
LIKE '字符樣式'	判斷是否符合指定的字符樣式
IS NULL	判斷是否為空值

 **例題 3.1.1** 在 HR 帳戶的 EMPLOYEES 資料表中，查詢西元 2003 年 6 月 17 日到職的員工之姓名及工作部門代碼。

解析：

1. SQL 語句：

```
SELECT first_name, last_name, department_id
FROM employees
WHERE hire_date='17-JUN-03';
```

2. 執行結果：

◆ FIRST_NAME	◆ LAST_NAME	◆ DEPARTMENT_ID
1 Steven	King	90

說明：

1. 再次提醒，如果您安裝的資料庫內碼使用 UNICODE，請在進入 SQL Developer 或 SQL*Plus 後執行下述命令，將連結期語言調整成英文，如此才能順利和例題帳戶內含有西元日期之現有資料進行比對。

```
ALTER SESSION SET NLS_LANGUAGE=AMERICAN;
```

2. 員工到職日期存於 EMPLOYEES 資料表之 hire_date 資料欄內。
3. 日期資料之預設儲存格式為 DD-MON-YY，即 2 位數之日期、月份之 3 個英文字縮寫、2 位數之西元年。
4. 一般而言，單引號括著的是字串資料。在本例中，此字串資料恰為儲存日期之預設格式，所以資料庫管理系統會自動把它轉換成日期來和 hire_date 欄位之資料進行比較。

 **例題 3.1.2** 在 HR 帳戶的 EMPLOYEES 資料表中，查詢薪水大於等於 24,000 的員工姓名、工作部門代碼及到職日。

解析：

1. SQL 語句：

```
SELECT first_name, last_name, salary, department_id, hire_date
FROM employees
WHERE salary >= 24000;
```

2. 執行結果：

FIRST_NAME	LAST_NAME	SALARY	DEPARTMENT_ID	HIRE_DATE
Steven	King	24000		90 17-JUN-03

 **例題 3.1.3** 在 HR 帳戶的 EMPLOYEES 資料表中，查詢薪水介於 11,000 與 12,000 之間的員工姓名、工作部門代碼及到職日。

解析：

1. SQL 語句：


```
SELECT first_name, last_name, salary, department_id, hire_date
FROM employees
WHERE salary BETWEEN 11000 AND 12000;
```

2. 執行結果：

	FIRST_NAME	LAST_NAME	SALARY	DEPARTMENT_ID	HIRE_DATE
1	Den	Raphaely	11000		30 07-DEC-02
2	Alberto	Errazuriz	12000		80 10-MAR-05
3	Gerald	Cambrault	11000		80 15-OCT-07
4	Lisa	Ozer	11500		80 11-MAR-05
5	Ellen	Abel	11000		80 11-MAY-04

說明：

1. 要查詢薪水介於 11000 與 12000 之間為複合條例。意即 salary>=11000 AND salary<=12000。
2. 此複合條件可簡化寫成：WHERE salary BETWEEN 11000 AND 12000。
3. BETWEEN 11000 AND 12000 中的 11000 為下界，12000 為上界。

 **例題 3.1.4** 在 HR 帳戶的 EMPLOYEES 資料表中，查詢主管的編號為 101 或 114，且隸屬於 100 號或 110 號部門的員工之姓名、主管編號、工作部門代碼及到職日。

解析：

本例示範多重查詢條件的組合應用情形。

1. SQL 語句：

```
SELECT first_name, last_name, manager_id, department_id, hire_date
FROM employees
WHERE manager_id IN(101,114) AND department_id IN (100,110);
```


2. 執行結果：

	FIRST_NAME	LAST_NAME	MANAGER_ID	DEPARTMENT_ID	HIRE_DATE
1	Nancy	Greenberg	101	100	17-AUG-02
2	Shelley	Higgins	101	110	07-JUN-02

說明：

1. WHERE manager_id IN(101,114)相當於 WHERE manager_id=101 OR manager_id=114。
2. 由於題目中兩個條件之關係為「且」，因此 WHERE 子句的寫法為：

```
WHERE manager_id IN(101,114) AND department_id IN (100,110)
```

 **例題 3.1.5** 如下表 3.2 為 OE 帳戶 production_information 表中的部分資料，請問下述命令的執行結果為何？

```
SELECT product_name, list_price
FROM product_information
WHERE (category_id = 12 AND category_id = 13) AND supplier_id = 102088
```

表 3.2 production_information 表中的部分資料

	PRODUCT_NAME	CATEGORY_ID	SUPPLIER_ID
1	Compact 400/DQ	12	102088
2	Compact 400/LQ	12	102087
3	Industrial 600/DQ	12	102088
4	Industrial 700/HD	12	102086
5	Inkjet B/6	12	102096
6	Inkjet C/4	12	102090
7	Inkjet C/8/HQ	12	102094
8	LaserPro 1200/8/BW	12	102099
9	LaserPro 600/6/BW	12	102087
10	HD 10GB /I	13	102071
11	HD 10GB /R	13	102071
12	HD 10GB /S	13	102051

解析：

整個 SQL 命令語法並沒有錯誤，但是因為資料中不會有類別編號 (CATEGORY_ID) 是 12，且又是 13 者，因此上述命令執行結果一定找不到符合條件之資料，故無輸出。

3.2 使用 LIKE 比較運算符及萬用字符

查詢字串資料時，您可能不總是知道要搜索的字符之精確值，此時就可在 WHERE 子句的條件式中使用 Like 比較運算符來建構可包含任何字符或數字之搜尋字串（以單引號括著）。Like 運算符使用「字符樣式匹配 (Character pattern-matching)」方式搜尋字串資料，被稱為「萬用字符搜尋 (Wildcard search)」模式，它可以使用下列 2 個萬用字符 (Wildcard) 來建構搜尋字串，如表 3.3。

表 3.3 LIKE 運算符的萬用字符

符號	說明
%	代表任何序列的零個或多個字符。 例如：'A%'表示 A 開頭的任何字串。
_	代表任何一個字符。 例如：'_A9%'表示第二字符是 A，第三個字符是 9 的任合字串。


Like 運算符也可當為某些 BETWEEN 比較運算的簡化。例如要查詢 2002 年間(1 月至 12 月)到職的員工，原本用 BETWEEN 運算符的 WHERE 子句須寫成：WHERE hire_date BETWEEN '01-JAN-02' AND '31-DEC-02'，而如果用 Like 運算符，則 WHERE 子句只要寫成：WHERE hire_date LIKE '%02'。

完整 SQL 語句如下所示：

```
SELECT first_name, last_name, department_id, hire_date
FROM employees
WHERE hire_date like '%02';
```

執行結果：

	FIRST_NAME	LAST_NAME	DEPARTMENT_ID	HIRE_DATE
1	Nancy	Greenberg	100	17-AUG-02
2	Daniel	Faviet	100	16-AUG-02
3	Den	Raphaely	300	07-DEC-02
4	Susan	Mavris	400	07-JUN-02
5	Hermann	Baer	700	07-JUN-02
6	Shelley	Higgins	110	07-JUN-02
7	William	Gietz	110	07-JUN-02

 **例題 3.2.1** 寫一 SQL 命令查詢 HR 帳戶 employees 資料表中，職務名稱 (job_id) 中有 CLERK 或 MAN，部門代碼 (department_id) 為 30，薪水 (salary) 大於 3000 的員工之編號、名字及薪資。

解析：

1. 本問題關鍵在於如何撰寫查詢職務名稱中有 CLERK 或 MAN，且部門代碼為 30、薪水大於 3000 等比較條件。

- 欲查詢職務名稱含有 CLERK 或 MAN 之資料，您可用 Like 運算符加萬用字符%為之。例如：`WHERE job_id LIKE '%CLERK' OR job_id LIKE '%MAN'`


SQL 語句：

```
SELECT employee_id, first_name, salary
FROM employees
WHERE (job_id LIKE '%MAN' OR job_id LIKE '%CLERK' )
AND department_id = 30
AND salary > 3000;
```

執行結果：

	EMPLOYEE_ID	FIRST_NAME	SALARY
1	114	Den	11000
2	115	Alexander	3100

在 LIKE 比較運算中，如果想要把 '%' 及 '_' 這兩個萬用字符當成一般字符來搜索時，則須在 '%' 及 '_' 字符之前加上跳脫符號 ESCAPE。

 **例題 3.2.2** 在 HR 帳戶的 employees 資料表中，查詢職務名稱中有 'SA_' 之員工。

解析：

- 本例示範 SQL 語句的跳脫符號之定義與使用情形。
- 本例之尋條件為 'SA_'，因 '_' 為 LIKE 運算符的控制字符，因此需要在 '_' 之前使用跳脫字符。在本例中定義 '\' 當為跳脫字符。

SQL 語句：

```
SELECT employee_id, last_name, job_id
FROM employees
WHERE job_id LIKE 'SA\_%' ESCAPE '\';
```


執行結果：

EMPLOYEE_ID	LAST_NAME	JOB_ID
1	145 Russell	SA MAN
2	146 Partners	SA MAN
3	147 Errazuriz	SA MAN
4	148 Cambrault	SA MAN
5	149 Zlotkey	SA MAN
6	150 Tucker	SA REP
7	151 Bernstein	SA REP
8	152 Hall	SA REP
9	153 Olsen	SA REP
10	154 Cambrault	SA REP
11	155 Tuvault	SA REP
12	156 King	SA REP

3.3 空值 NULL 之判斷

在關聯式資料庫的資料表中，如果欄位出現空值（NULL value），表示該欄位的值可能為下述情況之一：(一)目前不知它的值（Unknown）或，(二)未指派（Unassigned），或(三)沒有值（Unapplicable）。因此您就不能在 WHERE 子句中用等號（=）來判斷某個欄位內容是否為空值。判斷欄位內容是否為空值方法如下：

1. WHERE 欄位 IS NULL; 判斷欄位是否為空值？
2. WHERE 欄位 IS NOT NULL; 判斷欄位是否不為空值？

 **例題 3.3.1** 寫一 SQL 命令查詢 OE 帳戶 production_information 資料表中，哪些產品的定價（list_price）漏填了。

解析：

1. 本例題之關鍵在於如何判斷定價（list_price）欄位是否為空值？
2. 要判斷某欄位是否為空值須用：

```
where 欄位 is null;
```

而不能用

```
where 欄位 = null;
```

SQL 語句：


```
SELECT product_name,list_price
FROM product_information
WHERE list_price IS NULL;
```

執行結果：

	PRODUCT_NAME	LIST_PRICE
1	HD 8GB /SI	(null)
2	8MB Cache /MM	(null)

3.4 在執行期間用連字符替代變數縮小檢索範圍

連字符&（Ampersand）替代變數並非標準 SQL 語句保留字，它是 Oracle SQLplus 與 SQL Developer 的特色之一，目的在增加 SQL 語句執行時與用戶間的互動性，使用時須將連字符&加在要由鍵盤輸入的變數名稱前即可，而此替代變數可用於欄位名稱、表格名稱或條件內容。

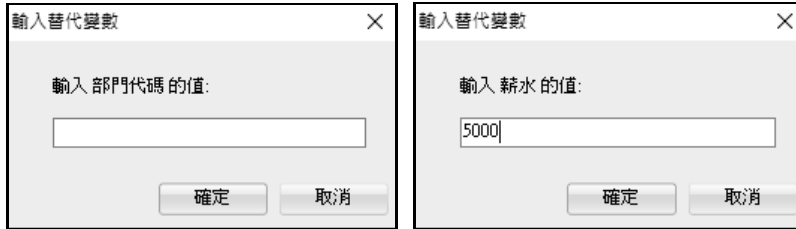
 **例題 3.4.1** 在查詢 HR 帳戶職務名稱中有 MAN 的員工之服務部門代號與薪水值時，可由使用者從鍵盤鍵入查詢值。

SQL 語句：

```
SELECT department_id,employ FROM employees
WHERE job_id like '%MAN'
AND department_id = &部門代碼
AND salary > &薪水;
```

解析：

1. 上述語句檢索條件中共使用二個連字符替代變數，第一個替代變數是「部門名稱」，第二個替代變數是「薪水」。執行時會依序彈出輸入視窗，如下：



2. 此處，依序在部門名稱鍵入 30，在薪水值鍵入 5000，則前述語句就相當於：

```
SELECT department_id,employee_id, last_name,first_name, job_id, salary
FROM employees
WHERE job_id like '%MAN' AND department_id = 30 AND salary > 5000;
```

執行結果：

找出部門 30，職務中有 MAN，薪水高於 5000 元之員工資料。



	DEPARTMENT_ID	EMPLOYEE_ID	LAST_NAME	FIRST_NAME	JOB_ID	SALARY
1	30	114	Raphael y Den	PU MAN	11000	

3.5 排序檢索結果

一、排序語法及規則


在 SELECT 命令中使用 ORDER BY 子句來排序查詢所檢索之資料，格式如下：

```
SELECT *|{[DISTINCT] 欄位|表達式[別名],...}  
FROM 表格  
[WHERE 條件]  
[ORDER BY {欄位, 表達式, 別名, 欄位位置數字}[ASC|DESC]];
```

說明：

1. 排序依據可為欄位、欄位別名、表達式、或代表 SELECT 語句中欄位位置之數字。
2. ASC 表升冪排序，DESC 表降冪排序，預設為升冪。
3. 對多個欄位排序時，指定之升／降冪只派給鄰近的一個欄位，其餘欄位均為升冪。
4. ORDER BY 須寫在 SELECT 命令的最後面。

二、排序檢索結果綜合應用

 **例題 3.5.1** 查詢 HR 帳戶 employees 資料表中，薪資 ≥ 3000 的員工之姓氏、薪資及到職日，並按薪資與年資升冪排序顯示。

解析：

1. 本例之關鍵在於依薪資與年資升冪排序顯示。
2. 由於 order by 預設為升冪排序，因此用 order by salary 即可依薪資升冪排序。
3. 由於到職日 (hire_date) 離目前日期越近者 (即 hire_date 越大者)，表年資越淺，因此如要依年資由小而大排序，則 hire_date 須用降冪排序，即 order by hire_date desc。

SQL 語句：


```
SELECT last_name, salary, hire_date  
FROM employees  
WHERE salary >=3000  
ORDER BY salary, hire_date DESC;
```

或

```
SELECT last_name, salary, hire_date
FROM employees
WHERE salary >=3000
ORDER BY 2, 3 DESC;
```

執行結果（部分）：

	LAST_NAME	SALARY	HIRE_DATE
1	Cabrio	3000	07-FEB-07
2	Feeney	3000	23-MAY-06
3	Walsh	3100	24-APR-06
4	Fleaur	3100	23-FEB-06
5	Davies	3100	29-JAN-05

 **例題 3.5.2** 在 OE 帳戶 product_information 資料表中，要對以下命令的 list_price - min_price 值進行升冪排序時，可能之寫法為何？

```
SELECT product_name, list_price, min_price, list_price - min_price Difference
FROM product_information
```

解析：

可能之寫法如下：

1. 在命令最後加入 ORDER BY 子句


```
ORDER BY list_price - min_price;
```

2. 使用欄位別名

```
ORDER BY Difference;
```

3. 使用欄位位置值

```
ORDER BY 4;
```

 **例題 3.5.3** 在 HR 帳戶的 employees 資表中，執行下列命令，結果為何？

```
SELECT first_name, department_id, salary
FROM employees
ORDER BY department_id, first_name, salary desc;
```


解析：

上述命令可查詢員工的名字、工作部門代碼及薪水。

1. 首先依部門代碼別，升冪排序。
2. 其次對部門相同員工，依名字升冪排序。
3. 最後再對部門相同、名字相同者，依薪水欄降冪排序。

執行結果（部分）：

	FIRST_NAME	DEPARTMENT_ID	SALARY
19	Girard	50	2800
20	Hazel	50	2200
21	Irene	50	2700
22	James	50	2500
23	James	50	2400
24	Jason	50	3300

 **例題 3.5.4** 在 HR 帳戶的 employees 資表中，找出公司最資深的前 6 名員工？

解析：

1. 此為 Top N 資料檢索問題，在 11g 暨以前版本可用每筆資料列的隱藏欄位 ROWNUM 來處理。
2. 年資 = (目前日期-到職日)/365

```
(Sysdate-hire_date)/365
```

3. 在完成檢索後，再用年資將資料排序，此時資料列的 ROWNUM 就已經按所指定欄位的指定順序排序，因此只要以 WHERE 條件式過濾取出所要資料筆數即達成要求。

SQL 語句：

此一部分查詢語句輸出的資料列之 ROWNUM 已經按年資順序排序。

```
SELECT * FROM (
SELECT employee_id,last_name,first_name, department_id , (SYSDATE-hire_date)/365
年資 FROM employees ORDER BY 5 DESC)
WHERE ROWNUM <=6;
```

執行結果：

	EMPLOYEE_ID	LAST_NAME	FIRST_NAME	DEPARTMENT_ID	年資
1	102	De Haan	Lex	90	20.3087199391171993911719939117199391172
2	203	Mavris	Susan	40	18.9114596651445966514459665144596651446
3	206	Gietz	William	110	18.9114596651445966514459665144596651446
4	205	Higgins	Shelley	110	18.9114596651445966514459665144596651446
5	204	Baer	Hermann	70	18.9114596651445966514459665144596651446
6	109	Faviet	Daniel	100	18.71967884322678843226788432267884322679

3.6 限制檢索返回列數子句（本節適用 12c 暨往後版本）


Oracle 資料庫在 12c 暨往後版本的 SQL 中增設查詢輸出列數限制子句（row limiting clause）FETCH，可用來解決前述的 TOP N 問題，其格式如下：

```
[ OFFSET offset ROWS ] FETCH NEXT | FIRST [ row_count | percent PERCENT ] ROWS | ROW
[ ONLY | WITH TIES ]
```

說明：

1. OFFSET 子句：指定在列限制開始前，要跳過的列數。OFFSET 子句是選擇性的，如果不用，則偏移量為 0。偏移量必須是一個數值，或是一個值為數值的運算式，規則如下：
 - 如果偏移量是負值，則視為 0。

- 如果偏移量為 NULL 或大於查詢返回的列數，則不返回任何列。
 - 如果偏移量包含小數，則小數被截斷。
2. FETCH 子句指定要返回的資料列數或百分比。後面接著 NEXT 或是 FIRST 結果相同，之後加上欲擷取的資料列數，或是百分比數目 PERCENT。如下例：


 **例題 3.6.1** 在 HR 帳戶，找出公司內薪水最高的前 3 名員工。

SQL 語句：

```
SELECT employee_id ,last_name,first_name,job_id,hire_date,salary
FROM employees
ORDER BY salary DESC
FETCH FIRST 3 ROWS ONLY;
```

執行結果：

	EMPLOYEE_ID	LAST_NAME	FIRST_NAME	JOB_ID	HIRE_DATE	SALARY
1	100	King	Steven	AD PRES	17-JUN-03	24000
2	101	Kochhar	Neena	AD VP	21-SEP-05	17000
3	102	De Haan	Lex	AD VP	13-JAN-01	17000

 **例題 3.6.2** 在 HR 帳戶，找出公司內薪水最高的前 1%員工。

解析：

1. HR 的 employees 內共有 107 筆資料，1%是 1.07 筆資料。
2. 以百分比擷取，小數無條件進位為整數，因此取出 2 筆資料。

SQL 語句：


```
SELECT employee_id ,last_name,first_name,job_id,hire_date,salary
FROM employees
ORDER BY salary DESC
FETCH NEXT 1 PERCENT ROWS ONLY;
```

執行結果：

EMPLOYEE_ID	LAST_NAME	FIRST_NAME	JOB_ID	HIRE_DATE	SALARY
1	King	Steven	AD PRES	17-JUN-03	24000
2	Kochhar	Neena	AD VP	21-SEP-05	17000

說明：

如果以 **WITH TIES** 代替 **ONLY**，則子句前必須要有 **ORDER BY** 排序子句，且如果值相同者也會被取出。如下例：

 **例題 3.6.3** 在 OE 帳戶，取出在手庫存量最多的前 10 筆商品。

解析：

- 首先我們用 **ONLY** 子句，確實由檢索結果中擷取 10 在手庫存量最多的前 10 筆商品，SQL 語句及結果如下：

```
SELECT product_name, quantity_on_hand
FROM inventories
INNER JOIN product_information USING(product_id)
ORDER BY quantity_on_hand DESC
FETCH NEXT 10 ROWS ONLY;
```

	PRODUCT_NAME	QUANTITY_ON_HAND
1	C for SPNIX4.0 - Sys	353
2	C for SPNIX4.0 - Sys	320
3	Screws <S.32.P>	304
4	Screws <B.28.P>	304
5	C for SPNIX4.0 - Sys	294
6	HD 8GB /SE	282
7	SDRAM - 16 MB	276
8	HD 12GB /I	275
9	HD 8GB /SI	275
10	Screws <S.16.S>	273

- 接著改用 **WITH TIES** 選項，此時因第 11、12 筆資料之庫存量均為 273，此時也會一併被取出，其 SQL 語句及執行結果如下：

```
/* 使用 WITH TIES 選項子句 */
SELECT product_name,quantity_on_hand
```

```
FROM inventories
```

```
INNER JOIN product_information USING(product_id)
```

```
ORDER BY quantity_on_hand DESC
```

```
FETCH NEXT 10 ROWS WITH TIES;
```

	PRODUCT_NAME	QUANTITY_ON_HAND
1	C for SPNIX4.0 - Sys	353
2	C for SPNIX4.0 - Sys	320
3	Screws <S.32.P>	304
4	Screws <B.28.P>	304
5	C for SPNIX4.0 - Sys	294
6	HD 8GB /SE	282
7	SDRAM - 16 MB	276
8	HD 12GB /I	275
9	HD 8GB /SI	275
10	Screws <S.16.S>	273
11	Screws <B.28.S>	273
12	Screws <Z.16.S>	273