第四章　SQL 介紹

黃仁鵬

4-1 資料庫共通的語言－SQL

- 像大多數資料庫相同，ORACLE 只會說一種共通的語言，而這種語言就是 SQL (Structural Query Language，結構化查詢語言)。

4-2 SQL 的歷史

- 在 70 年代，E.F.Codd 首先提出關聯式資料模型(Relational Data Model)。而後 IBM 公司在 System R 關聯式資料庫管理系統中，研發出最早的 SQL 語言叫做 SEQUEL (Structured English Query Language)。

4.3 SQL 的基本語言

- 資料定義語言(Data Definition Language，DDL)
- 資料操縱語言(Data Manipulation Language，DML)
- 資料控制語言(Data Control Language，DCL)

資料定義語言(Data Definition Language，DDL)

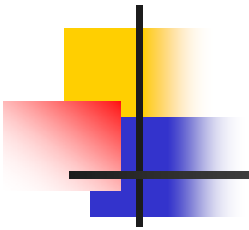
- 用來定義資料庫的綱要(Schema)，如關聯表名稱，關聯表內部的欄位、屬性、資料型態、定義或更改整合限制條件(Integrity Constraint)，給予(Grant) 或取消(Revoke) 權限(Privilege) 或角色(Role)等。

資料操縱語言(Data Manipulation Language，DML)

- 主要提供使用者資料庫中資料操作的指令，如新增(Insert)、刪除(Delete)、修改(Update) 與查詢(Select) 等運算。

資料控制語言(Data Control Language，DCL)

- 用來控制資料庫系統內部的異動交易(Transaction) 處理與系統效能等指令。



- 然而 ORACLE 除了以上三種 SQL 基本語言外，ORACLE 又把資料控制語言(Data Control Language) 更細分成：
  - 異動交易處理指令(Transaction Control Command)
  - 連線會談期間控制指令(Session Control Command)
  - 系統控制指令(System Control Command)

異動交易處理指令(Transaction Control Command)：

- 用來控制對資料作出的改變，也就是處理由 DML 指令在異動交易過程(Transaction) 所做的修改，如委付確認(Commit)、回復(Rollback) 以及設定儲存點(Savepoint) 等。

連線會談期間控制指令 (Session Control Command)：

- 用來動態地控制使用者連線期間的特性(Property)，如改變連線會談期間的內定語言、關閉資料庫鏈結(Database Link) 等。

系統控制指令(System Control Command)

- 用來動態的管理一個ORACLE Instance (資料庫實例) 的特性。

4-3-1 資料定義語言 (Data Definition Language，DDL)

- 資料定義語言(Data Definition Language，DDL) 的主要功用爲：
  1. 新增、修改、刪除物件(Object)。
  2. 給予(Grant)、撤回(Revoke)、刪除(Drop)、權限(Privilege) 與角色(Role)。
  3. 建立系統稽核(Audit) 功能。
  4. 在資料字典(Data Dictionary) 中加入註解。

4-3-2 資料操縱語言 (Data Manipulation Language)

- 用來處理與查詢綱要物件(Schema Object) 的資料。這些 DML 指令並不會自動地做內隱式委付確認(Implicitly Commit)。

4-3-3 異動交易處理指令 (Transaction Control Command)

- 異動交易處理指令(Transaction Control Command) 的主要功用爲：處理 DML 指令在異動交易過程(Transaction) 中所做的修改。這些指令並不會自動地做內隱式委付確認(Implicitly Commit)。

4-3-4 連線會談期控制指令 (Session Control Command)

- 連線會談期間控制指令(Session Control Command) 的主要功用爲：它動態地控制使用者連線會談期間的特性。這些指令並不會自動地做內隱式委付確認(Implicitly Commit)。

4-3-5 系統控制指令 (System Control Command)

- 系統控制指令(System Control Command) 的主要功用爲：它動態地管理 ORACLE 實例(ORACLE Instance) 的特性，這些指令不會自動地做內隱式委付確認(Implicitly Commit)。

4-3-6 內嵌式 SQL 指令 (Embedded SQL Command)

- 內嵌式 SQL 指令的主要功用爲：它可將資料定義語言(Data Definition Language，DDL)、資料操縱語言(Data Manipulation Language，DML)、異動交易處理指令(Transaction Control Command) 等指令放入程序化語言的程式中。

4-4 資料型態

- ANSI-SQL 主要提供三種主要的資料型態：
  - 字元
  - 數字
  - 浮點數

4-5 資料型態轉換

- 在 ORACLE 中能夠將某種資料型態轉換成另一種資料型態。資料型態的轉換可以透過 ORACLE 系統自動轉換，或是透過資料型態轉換函數來完成。

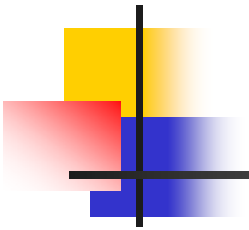
4-6 基本表格(BASED TABLE)

- 一個基本表格(Based Table) 實質上由兩部分組成：
  - 表格名和一組欄位名構成的分類資訊
  - 實際存放的資料。

4-6-1 表格名、欄位名的命名規則

- 在使用 CREATE TABLE 命令建立表格時，定義表格名和欄位名的規則如下：
  1. 表格名、欄位名必須以字母開頭，大小寫均可，其後的字元可以由字母、數字組成，也可以包括“$”、“#”和底線“_”。

2. 同一個使用者的基本表格名必須唯一，而且不能與 ORACLE 的保留字相同；同一基本表格中的欄位名也須唯一，不允許在同一基本表格中兩個欄位名相同。



3. 表格名與欄位名是不區分大小寫的。例如：FIRSTNAME，FirstName 和 Firstname 均認爲是相同的。

4. 如果表格名、欄位名用雙引號括起來，則可以不滿足上述規則。如果使用了雙引號，則要區分字母的大小寫，如“FIRSTNAME”與“Firstname”就不再認爲是相同的。

4-6-2 建立基本表與輸入資料

- 建立基本表命令 － CREATE TABLE

```
CREATE TABLE <表格名>
<欄位名1><型態> [NULL | NOT NULL][,
<欄位名2><型態> [NULL | NOT NULL]]
```

插入資料－INSERT INTO……

INSERT INTO <表格名>(<欄位名1>,<欄位名2>,……)
VALUES(<值1>,<值2>,……);

- 關聯表格內部的整合限制(Integrity Constraints) 可分爲三種：
  - 個体整合限制(Entity Integrity Constraints)
  - 值域整合限制(Domain Integrity Constraints)
  - 參考整合限制(Referential Integrity Constraints

個体整合限制(Entity Integrity Constraints)

- 它規範關聯表內部的整合限制條件，例如：主鍵(Primary Key) 的值不可爲虛值(Null) 、欄位的值是否唯一(Unique)、欄位的值是否可爲虛值(Null)。
  - PRIMARY KEY (<*主鍵欄位名*>)
  - [NOT] NULL
  - UNIQUE

值域整合限制(Domain Integrity Constraints)

- 它規範關聯表內部屬性值必須在某一限制範圍之內，例如：欄位的型態(Type)與大小(Size) 、輸入值的檢查(Check) 。
  - CHECK
  - DEFAULT

參考整合限制(Referential Integrity Constraints)

- 它規範關聯表格與關聯表格之間的整合限制條件，例如：表格中的外來鍵 (Foreign Key) 設定。
  - FOREIGN KEY

4-6-3 更新資料

更新資料－UPDATE……SET

```
UPDATE <表格名> SET <欄位名1>=<新值1>,
                    <欄位名2>=<新值2>,
                    ……
                    <欄位名n>=<新值n>
WHERE <被更新記錄的條件>
```

4-6-4 刪除資料

- 刪除資料－DELETE FROM

```
DELETE FROM <表格名>
WHERE <條件>
```

刪除指令不能只刪除部分欄位中的資料，因此不需要指明欄位名。

4-6-5 修改基本資料表格 (Based Table) 結構

- 修改基本資料表格結構命令，主要可分爲：
  1. 新增欄位定義。
  2. 修改欄位定義。
  3. 廢除欄位定義。
  4. 新增整合限制。
  5. 修改整合限制。
  6. 廢除整合限制。

新增欄位定義－ALTER TABLE……ADD

ALTER TABLE <表格名>

ADD (<欄位名1><資料型態>

[,<欄位名2><資料型態>…]);

修改欄位的定義－ALTER TABLE……MODIFY

ALTER TABLE<表格名>

MODIFY (<欄位名1> [<資料型態>]

[NULL|NOT NULL]

[,<欄位名2>[<資料型態>[NULL|NOT NULL]]…]);

廢除欄位定義--ALTER TABLE ... DROP COLUMN

ALTER TABLE <表格名>

DROP COLUMN <欄位名>

新增整合限制(Integrity Constraint)－ALTER TABLE...ADD CONSTRAINT

ALTER TABLE <表格名>

ADD (CONSTRAINT <整合限制名1><整合限制條件1>

[, CONSTRAINT <整合限制名2><整合限制條件2>...]);

修改整合限制

修改整合限制又可分成:

- 實施整合限制 (ENABLE)

  ALTER TABLE <表格> ENABLE CONSTRAINT。

- 暫時取消整合限制(DISABLE)

  ALTER TABLE <表格> DISABLE CONSTRAINT 。

- 廢除整合限制

  ALTER TABLE <表格> DROP CONSTRAINT。

4-6-6 廢除基本表格

- 廢除基本表格命令－DROP TABLE

  當要廢除的基本表格中含有參考整合限制時，也就是被別的表格參考到表格中的候選鍵時，必須加入 CASCADE CONSTRAINTS 才能廢除表格。

4-6-7 建立表格的建議步驟

1. 廢除表格，以免原表格已存在而產生錯誤。
2. 建立表格。
3. 建立個体整合限制與值域整合限制。
4. 建立參考整合限制。
5. 插入資料。

4-6-8 建立表格時加入整合限制

- 此時建立表格的順序就很重要。被關聯的表格要先建立，例如：parent 表格要比 student表格先建立； parent 表格的資料要比 student 表格資料先輸入。

4-6-9 簡易查詢

- 簡易查詢－SELECT

```
SELECT <查詢內容>
FROM <表格名>
WHERE <條件>
ORDER BY <排序內容>
```

4-7-1 虛擬欄位

- 虛擬欄位(Pseudo Column) 不是任何表格的一個真實欄位，然而，當其出現於一個 SELECT 指令的選擇列表中時，它將提供一個具有特殊或特別意義的值。

4-7-2 虛擬表格

- 虛擬表格(DUAL) 是 ORACLE 提供的最小的工作表，它僅包含一行一列。

4-8 資料型態轉換函數

- HEXTORAW(CHAR)
- RAWTOCHAR(RAW)
- TO_CHAR (DATE, 特定日期格式)
- TO_CHAR (NUMBER, 特定數值格式)
- TO_DATE(CHAR,特定日期格式)
- TO_NUMBER(CHAR ,特定數值格式)

- 特定的日期格式參數舉例說明：

  TO_DATE(‘31-3月-1964’,‘DD-MONTH-YYYY’)

  TO_CHAR(HIREDATE , ‘DD/MM/YYYY’)

  TO_CHAR(HIREDATE , ‘AD DD-MON-YYYY’);

4-9 條件運算

1. 用來比較的：

=	等於	<>	不等於	<	小於
>	大於	<=	小於等於	>=	大於等於

2. 在（或不在）某連續範圍之內

   (NOT) BETWEEN … AND …

3. 在 (或不在) 某值域範圍

   (NOT) IN (值域範圍)

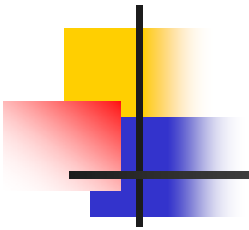
4. 與某一綱要相匹配 (或不匹配)

(NOT) LIKE (綱要)

在 ORACLE 系統中匹配綱要是用“%”(百分比符號) 表示任何字元；用“ _ ”(底線符號) 表示任一字元。

5. 列表或子查詢結果中的任一(ANY) 或全部(ALL) 值

ANY 任一　　ALL 全部



6. 辨別子查詢結果有無傳回值

   (NOT) EXISTS

7. 虛值(NULL) 判斷條件

   IS（NOT）NULL

4-10 SQL 中常用的函數

- SQL 函數是用來處理一個資料項，並傳回結果的運算。函數可以接受零個或多個參數，利用不同參數的函數完成不同的運算，並產生一個或多個結果。

函數種類

1. 數值函數:數值函數接受數值輸入，傳回數值。
2. 字元函數：字元函數接受字元輸入，可傳回字元或數值。
3. 日期函數
   日期函數操作 DATE 資料型態的值，除 MONTHS_BETWEEN 外，其他所有的日期函數會傳回一個 DATE 型態的值。
4. 虛值函數 NVL
5. 群組函數 (Aggregation functions)

數值函數

函　　數	說　　明
ABS(n)	傳回 n 的絕對值。
CEIL(n)	傳回大於或等於 n 的最小整數。
FLOOR(n)	傳回等於或小於 n 的最大整數。
MOD(m,n)	傳回 m 除以 n 的餘數，如果 n=0，則傳回 m。
POWER(m,n)	傳回 m 的 n 次方，如果 m 爲負數，則 n 必須是整數。
ROUND(n[,m])	傳回將 n 四捨五入到小數點右邊 m 位的值。當 m 忽略時，四捨五入到個位。當 m 爲負數，四捨五入到小數點左邊第 m 位。
SIGN(n)	當 n<0，傳回 -1；當 n=0 傳回 0；當 n>0，傳回 1。
SQRT(n)	傳回 n 的平方根，n 不能爲負數。
TRUNC(n[,m])	傳回在 m 位無條件捨去的 n 值，當 m 忽略，在 0 位無條件捨去；m 爲負數，將小數點左邊第 m 位無條件捨去。

字元函數

函　　數	說　　明
CHR(n)	傳回字元，其 ASCII 碼等於 n。
CONCAT(s1, s2)	傳回 s1 字串連接 s2 字串的新字串。
INITCAP(s)	將字串 s 的第一個字母爲大寫，其它的爲小寫。

字元函數(cont.)

LENGTH(s)	傳回字串的長度。
LOWER(s)	將所有字母爲小寫。
LPAD(s1, n[, s2])	用 s2 字元填入 s1字串中的左邊，使其總長度爲 n。s2 的預設值爲單個空格 (‘ ’)。如果 s1 字串長度大於 n，函數則傳回 s1 字串的部分。n 爲傳回值的總長度。
LTRIM(s [,set])	從 s 字串的左邊開始移去 set 字串中的字元，直至第一個不是 set 中的字元爲止。set 的預設值爲單個空格 (‘ ’)。

字元函數(cont.)

REPLACE(s, 搜尋字串 [,取代字串])	將用取代字串代替每一個搜尋字串的值，如果取代字串不寫，則所有搜尋字串的值被刪去。
RPAD(s1, n[,s2])	用 s2 填入字串的右邊，使其總長度爲 n。s2 字串的預設值爲單個空格(‘ ’)。如果 s1 比 n 長，則傳回 s1 的部份。n 爲傳回值的總長度。
RTRIM(s [,set])	從右邊開始移去 set 字串中的字元，直至最後一個不 set 中的字元爲止。set 的預設值爲單個空格 (‘ ’)。

字元函數(cont.)

SUBSTR (s, m [,n])	求 s 字串的子字串，從 m 處的字元開始，取 n 個字元長。
TRANSLATE (s, fromS, toS)	將 s 字串中的 fromS 字串中每一個字元值利用 toS 字串中相應字元替換。不在 fromS 字串中的 s 的字元則不進行轉換。
UPPER(s)	傳回 s，其中所有字母爲大寫，傳回值與引數 s 有相同資料型態。
DECODE(ex, s1, r1 [, s2, r2,…], d)	依照 ex 的值傳回不同的值，若 ex 的值爲 s1 則傳回值 r1，[若 ex 的值爲 s2 則傳回值 r2 …，]否則傳回值 d。

日期函數

函　　數	說　　明
ADD_MONTHS(d,n)	傳回加 n 個月的 d 日期，引數 n 爲整數。如果 d 爲該月中最後一天 (例如：31日)，而且結果月份的總天數較少 (例如：二月的 28 日)，則結果的日期爲結果月份的最後一日 (例如：28 日)，否則結果日期的天數與 d 中的原日期一致。
LAST_DAY(d)	傳回包含 d 日期的月份最後一天之日期

日期函數(Cont.)

MONTHS_BETWEEN(d1,d2)	傳回日期 d1和 d2 之間的月數。如果 d1 比 d2 晚，結果爲正；如果 d1 比 d2 早，則爲負；如果 d1 與 d2 爲同一日期，則結果爲 0
NEXT_DAY(d, 星期幾)	傳回比日期 d 晚的第一個星期幾。
ROUND(d[,fmt])	傳回依照指定的格式(fmt) 的單位將 d 四捨五入到最近日期。
TRUNC(d[,fmt])	傳回依照指定的格式(fmt) 的單位將 d 無條件捨去到最近的日期。
SYSDATE	傳回當前的日期和時間。注意：CHECK 約束條件中不能使用該函數。

虛值函數 NVL

- 虛值不是零，也非空白，用來記錄目前資料值未知的情況。虛值本身不能做任何比較與運算，因爲結果都是虛值。

以虛值的特性與意義

可大致分爲以下三種：

1. 可適用的虛值(Applicable Null Values)
2. 不可適用的虛值(Inapplicable Null Values)
3. 完全不知道的虛值(Totally Unknown)

可適用的虛值(Applicable Null Values)

- 若是某特定欄位的值確實存在，但是我們並不知道其值爲何，所以暫時存放一個虛值，等到日後得知確實數值後再填入。

EMPNO	ENAME	ADDRESS
1234	JOHN	台南
1468	JONES	NULL

不可適用的虛值(Inapplicable Null Values)

- 若是某特定欄位的值根本就不存在，則我們就可放入一個虛值，表示此欄位的值並不存在。

EMPNO	ENAME	JOB	COMM
3838	SCOTT	SALESMAN	200
3849	FORD	ACCOUNTANT	NULL

完全不知道的虛值(Totally Unknown)

- 若是某特定欄位的值，我們完全不知道這個值是否存在。

EMPNO	ENAME	SPOUSE
8088	FORD	LINDA
8092	MARY	NULL

群組函數 (Aggregation functions)

- SQL 可利用 GROUP BY 子句將查詢的結果分組，並對每個群組利用前面介紹的群組函數傳回單列群組資訊。查詢的資料依照 GROUP BY 子句所指定的運算式的值分組，將相同值的資料組成一組。

4-11 SELECT 指令小結

SELECT <欄位名1>，<欄位名2>，.....，<欄位名n>
FROM <表格名1>，<表格名2>，.....，<表格名n>
[WHERE 條件**]**
GROUP BY <欄位名1>，<欄位名2>，.....，<欄位名n>
[HAVING 分組篩選條件**]**
[ORDER BY <欄位名1>，<欄位名2>，.....，<欄位名n>]

4-12-1 視觀表格(View) 的概念

- 可以由一個基本表格中選取的某些資料列和欄位組成，也可以由幾個表格中滿足一定條件的資料組成。
- 視觀表格如同是基本表格(Base Table)窗口，它反映了一個或多個基本表格的局部資料，但視觀表格本身並無實際存放資料，資料都來自於基本表格，視觀表格僅是對應基本表格的部分資料，因此視觀表格只是一個虛擬表格。

使用視觀表格的主要原因

- 視觀表格可以提供安全：視觀表格是建立在一個或數個表格上，使用者可以操作視觀表格而不會直接存取表格中較機密的資料欄位。
- 視觀表格可以隱藏複雜的資料查詢：跨多表格的關聯查詢或是較複雜的查詢都可以利用視觀表格來代替，如此可以簡化查詢操作。

4-12-2 建立視觀表格的指令

- 建立視觀表格(View)－CREATE

VIEWCREATE VIEW <視觀表格名> [<視觀表格欄位名串列>]
AS SELECT 指令
[WITH CHECK OPTION];

4-12-3 使用視觀表格(View) 的指令

- 視觀表格的使用如同基本表格一樣，使用者可以查詢其中的資料，在某些情況下可以對視觀表格進行修改、刪除或插入資料操作，其操作方式與一般基本表格相同。

4-12-4 廢除視觀表格

- 廢除視觀表格－DROP VIEW

  DROP VIEW <視觀表名>;

  雖然視觀表格(View) 的廢除不會影響基本表格的使用，但會導致該視觀表格相關的所有授權被自動取消，該視觀表格相關的所有視觀表格和同義字變成無效。

4-13 基本表格、視觀表格的拷貝

- 在使用資料庫時，經常需要對原始的資料資訊進行備份，備份資料可以用來做刪除前的歸檔資料，或用來做臨時性修改，或是提供給其他人使用。我們可以在資料庫內拷貝基本表格或視觀表格。拷貝一個基本表格，一般包括該表格的欄位定義和所有的資料。

拷貝表格

- CREATE TABLE……AS SELECT

CREATE TABLE <表格名>[(<欄位名> [NOT NULL]…)]

AS SELECT 命令;

其中：<表格名> 爲新表格的名字；[(<欄位名> [NOT NULL]…)] 爲可選項，如果不選這項，則備份表格的各欄位名及資料型態是由SELECT敘述的查詢結果決定。

4-14 建立與使用序列(SEQUENCE)

在 ORACLE 系統中並沒有提供任何自動編號的欄位，所以 ORACLE 就利用序列(SEQUENCES)，來達成相同的功能。它的功能類似一個計數器(Counter)。序列是產生連續數字最有效率的方法。

建立序列方法

CREATE SEQUENCE <序列名>

當你建立了序列之後，一般預設的情況下 ORACLE 系統會在記憶體之中產生 20 組號碼，以供你使用。每一個號碼只能產生一次就換到下一個號碼，你可以使用 sequence_name.CURRVAL 來得到目前值與利用 sequence_name.NEXTVAL 得到下一個可用之值。

更改序列-- ALTER SEQUENCE

ALTER SEQUENCE <序列名>
<New_paramenters>;

其中<New_parameters> 爲新的選項參數。

廢除序列-- DROP SEQUENCE

DROP SEQUENCE <序列名>

此外我們可以利用查詢 ORACLE 系統中資料字典的視觀表格 user_sequences 得知使用者目前建立了那些序列。

4-15-1　索引(Index) 的概念

索引(Index) 就好比書籍中的目錄，當你要找尋某章節時，可迅速地找到，節省找尋的時間。然而在一般查詢中，ORACLE 要逐列查詢該表格，檢查是否爲所需的資料，直到找到所有符合 WHERE 子句中的條件的資料列。

4-15-2 建立索引

建立索引－CREATE INDEX……ON

CREATE [UNIQUE] INDEX <索引名>
ON <表格名>　　(<欄位名> [ASC/DESC]
[, <欄位名>[ASC/DESC]……]);

建立索引的原則

(1) 較大的基本表格應該建立索引

表格越大，索引越能更好地改善回應時間，索引也越有效，ORACLE 系統越有機會找到最短路徑。

建立索引的原則(cont.)

(2) 對一個基本表格可以建立多個索引，但建議最好不要超過三個。

一個表格可以有多個索引，索引越多，查詢速度越快。但是建立太多的索引是不合適的。

建立索引的原則(cont.)

(3) 一般在主鍵欄位上建立索引

當一個主鍵爲複合欄位主鍵時，那麼最好是把資料種類最多的欄位放在 CREATE INDEX 命令列的首位。如果各欄位資料種類相近，則將最經常用到的欄位放在前面。

建立索引的原則(cont.)

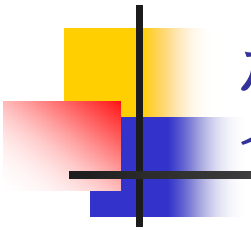
(4) 建立索引要注意先後順序

通常在表格中輸入資料後才建立索引。如果先建索引後輸入資料，則每插入一列都要對索引進行一次更新，這很浪費時間。但如果希望保證輸入資料的唯一性，那麼只能以犧牲性能爲代價，在輸入資料前先建立好唯一索引。

建立索引的原則(cont.)

(5) 建立索引要注意 95/5 法則

在某些情況下使用索引會拖慢速度。你可利用 95/5 法則評估是否使用索引。當查詢回覆結果小於表格全部資料的 5% 時，使用索引是最佳的查詢方式。如果查詢回覆結果超過表格全部資料的 95% 時，不使用索引查詢的方式通常會有比較好的效能。



索引的內部結構

ORACLE 系統採用的索引內部結構是 B + 樹，從而保證對表格中任何一列的存取所需要的時間都差不多。

4-15-3 索引的類型

- 依建立方式分類：
  - 單鍵索引
  - 多鍵索引
- 依索引欄位值是否可重覆分類：
  - 唯一索引
  - 非唯一索引
- 依儲存方式分類：
  - 壓縮索引
  - 非壓縮索引。

單鍵索引和多鍵索引

- 單鍵索引是指在表格中利用某一欄位所建立的索引。
- 多鍵索引是指在表格中利用多於一個欄位所建立的索引。

唯一索引和非唯一索引

- 唯一索引和非唯一索引是根據在建立索引時是否指明 UNIQUE 來確定的
- 唯一索引要求被索引的欄位值不能重覆
- 非唯一索引則允許索引欄位值可以重覆

壓縮索引和非壓縮索引

- 建立索引時，若索引按壓縮方式儲存，則爲壓縮索引，否則爲非壓縮索引。
- 壓縮索引的優點是減少了儲存需要，因而減少了 I/O 的次數，但是爲了完成壓縮以及在檢索時，對壓縮檔做解碼需要耗費一定的時間。
- 使用非壓縮索引，由於在尋找時無需進行解碼，因而加快了執行速度。

4-15-4 廢除索引

- 廢除索引－DROP INDEX

DROP INDEX [<使用者名稱> .]<索引名>;

當使用者不再需要使用索引時，可以使用 DROP INDEX 指令廢除索引。

其中：<索引名> 指出要廢除的索引名字。[<使用者名稱>] 的選項，表示該索引的擁有者。

4-16 同義字(Synonym)

ORACLE 系統中提供了一個很方便的物件－Synonym (同義字)。同義字可分爲 Public (公用) 與 Private (專用) 兩種。

同義字的主要目的

- 隱藏原資料庫物件的原擁有者的資訊。
- 爲某物件提供一個較簡單或是方便的名字。
- 隱藏某資料庫物件的實際位置。

4-16-1 定義同義字

定義同義字－CREATE SYNONYM……FOR

建立同義字的命令如下：

CREATE [PUBLIC] SYNONYM <同義字>
FOR [<使用者名稱>.] <資料庫物件>;

其中：[PUBLIC] 爲選項，如果選擇該項，則表示建立的這個同義字可以被所有使用者使用，也就是一個公用的 (public) 同義字，但只有 DBA (資料庫管理員) 能夠選擇 [PUBLIC] 項。

4-16-2 廢除同義字

- 廢除同義字－DROP SYNONYM

  DROP [PUBLIC] SYNONYM <同義字>;

  只有 DBA 才能選擇 [PUBLIC] 項來刪除公用同義字。同義字的廢除不會影響其原本表格、視觀表格(View)使用權限。改變一個同義字的方法是先廢除該同義字，再重建該同義字。

4-17 資料庫鏈結(Database Link)

資料庫鏈結允許使用者處理遠端資料庫的資料，可使用已經設定好的資料庫鏈結而不需要知道遠端資料庫的實際位置。使用資料庫鏈結時，系統會透過網路啓動一個連線(session)，以便執行你的 SQL 指令。

4-17-1 建立資料庫鏈結

- 建立資料庫鏈結－CREATE DATABASE LINK

CREATE [PUBLIC] DATABASE LINK <資料庫鏈結名>
CONNECT TO <使用者> IDENTIFIED BY <密碼>
USING <主機字串>

其中：<使用者> 爲連線遠端主機中的合法使用者；<主機字串> 爲利用 NET8 Easy Config 設定好的主機字串。

4-17-2　廢除資料庫鏈結

- 廢除資料庫鏈結－DROP DATABASE LINK

  DROP　DATABASE　LINK <資料庫鏈結名>