CHAPTER 5: SORTING & INDEXING

Need for Ordered Information

- Data do not come in any order. It is not unusual to have the records in a table arranged in arbitrary order. Keeping data in this way has a number of drawbacks.
- One obvious drawback is related to record(s) searching, a task often required in data processing.
- For records in arbitrary order, there cannot be any shorter way to search for the record(s) except examine every record of the table sequentially, until the record is found or the end of the file is reached.

Methods to Order Information

- Visual FoxPro provides two different ways to arrange the records in a table, sorting and indexing.
- Sorting changes the physical record position in the table. So it leads to the creation of another table.
- Indexing creates an index file which consists of only enough information to determine the logical record position. The actual record position does not change.

Sorting

- Sorting is used to change the physical order of record in database tables. In Visual FoxPro, sorting creates a totally new table in the order you specified.
- Visual FoxPro can sort the table in ascending order or descending order based on the field or a combination of fields.
- Numeric fields are sorted in the natural order. For character fields, Visual FoxPro sorts according to the ASCII codes.

Sorting

Syntax

- ***SORT TO TableName ON FieldName1 [/A | /D][/C], ;***
  FieldName2 [/A|/D][/C][ASCENDING | ;
  DESCENDING] FIELDS FieldNameList
- **A new table (*TableName*) is created.**
- **/A- ascending, /D- descending, /C- case insensitve**
- **USE student.dbf**
  - **SORT TO nt1 ON name /D, dob /A**
  - **SORT TO nt2 ON name /D/C, dob /A FIELDS addr**
  - **SORT TO nt3 ON name, telno /A, dob**
    ASCENDING
  - **SORT TO nt4 ON name, dob**

Sorting

Disadvantages

- Data redundancy occurs since new tables are created.
- It consumes spaces in the hard disk.
- SORT is not automatic, manual process involves.
- Process is slow because disk access is usually slow.
- It requires extra empty spaces for temporary storage.
- Only fields can be used in sorting. Expressions (e.g. wages*days) are not allowed.

Indexing

Introduction

- Visual FoxPro's index is similar to the index at the back of a book. The purpose of a book index is to help readers to find information on a certain topic quickly. The index file contains the only information (pointers) enough to determine the logical order of the records in the file.
- There are two types of index file in Visual FoxPro. They are single index file (independent index file) and compound index file (dependent index file or independent index file), with extensions IDX and CDX respectively.

Single Index

Creation of Single Index

- Independent index files (IDX) would not be updated automatically, if they are not activated.
- *INDEX ON eExpression TO IDXFileName*
- INDEX ON name TO name.idx
- INDEX ON {01/01/2001}-dob TO ages.idx
- INDEX ON days*wages TO income.idx
- INDEX ON price*0.8 TO discount.idx
- INDEX ON class+classno TO order.idx
- In these examples, files name.idx, ages.idx, income.idx, discount.idx, order.idx would be created.

Types of Compound Index

- A compound index file may consist of more than one index expression, called tags. Each index expression has a tag name.
- There are two types of compound index file. They are structural and non-structural.
- Non-Structural compound index file does NOT has the same name as the table and is dissociated with it. When table is open, the CDX file will NOT also be open automatically.
- Structural compound index file has the same name as the table and is associated with it. When table is open, the CDX file will also be open automatically.

Non-Structural Compound Index

Creation of Non-Structural Compound Index

- *INDEX ON eExpression TAG TagName ;*
  OF CDXFileName [ASCENDING | DESCENDING]
- USE drivers.dbf
  - ➔ INDEX ON name TAG name OF abc.cdx
  - ➔ INDEX ON driverid TAG driverid OF abc.cdx
  - ➔ INDEX ON dob TAG dob OF xyz.cdx DESC
  - ➔ INDEX ON address TAG address OF xyz.cdx
- It is necessary to specify the CDXFileName since it is a non-structural (independent) compound index.
- In these examples, files abc.cdx, xyz.cdx would be created.

Structural Compound Index

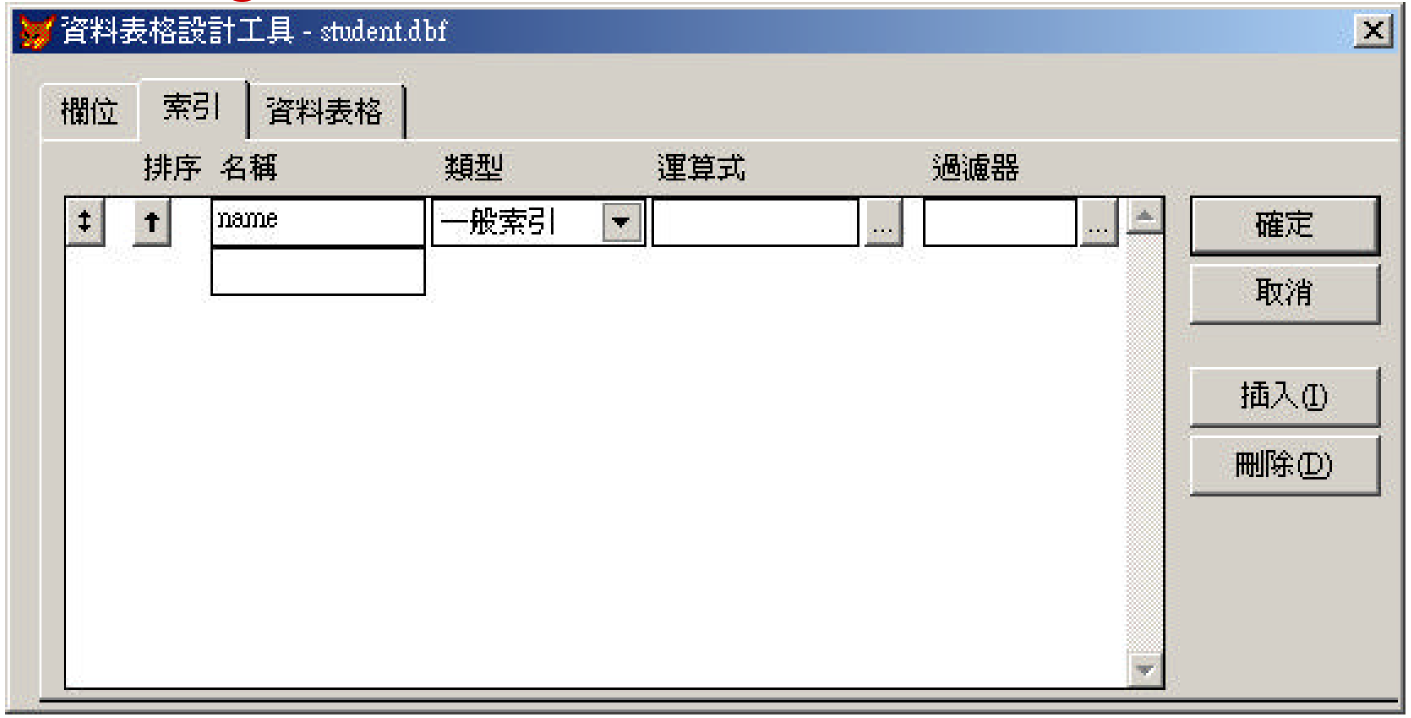
Creation of Structural Compound Index

- *INDEX ON eExpression TAG TagName ;*
  [ASCENDING | DESCENDING]
- USE drivers.dbf
  - INDEX ON name TAG name
  - INDEX ON driverid TAG driverid
  - INDEX ON YEAR(dob) TAG year DESC
  - INDEX ON MONTH(dob) TAG month
- It is not necessary to specify the CDXFileName since it is a structural (dependent) compound index.
- In these examples, only file drivers.cdx would be created.

Structural Compound Index

Creation of Structural Compound Index (cont'd)

- Compound index can also be created by another method. Specify the eExpression and TagName in the Table Designer.

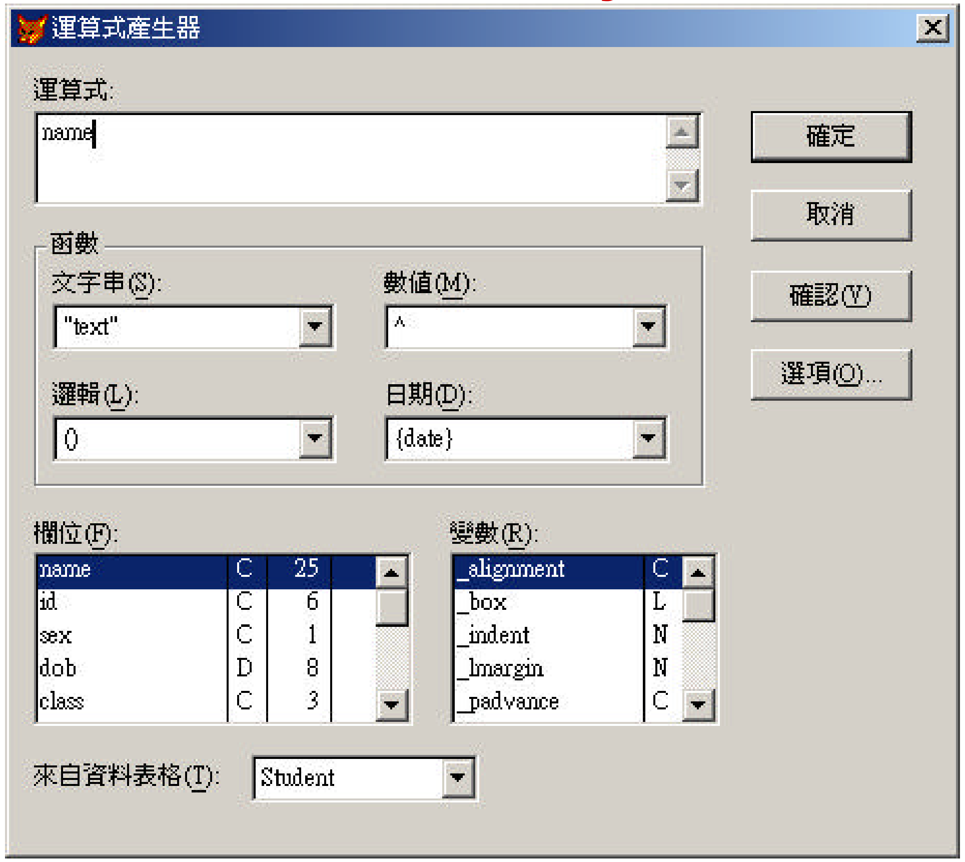


排序	名稱	類型	運算式	過濾器
↑	name	一般索引		

Structural Compound Index

Creation of Structural Compound Index (cont'd)

- Expression Builder provides choices for you to build expression.



運算式產生器

運算式:

name

確定

取消

確認(V)

選項(O)...

函數

文字串(S): "text"

數值(M): ^

邏輯(L): ()

日期(D): {date}

欄位(F):

name	C	25
id	C	6
sex	C	1
dob	D	8
class	C	3

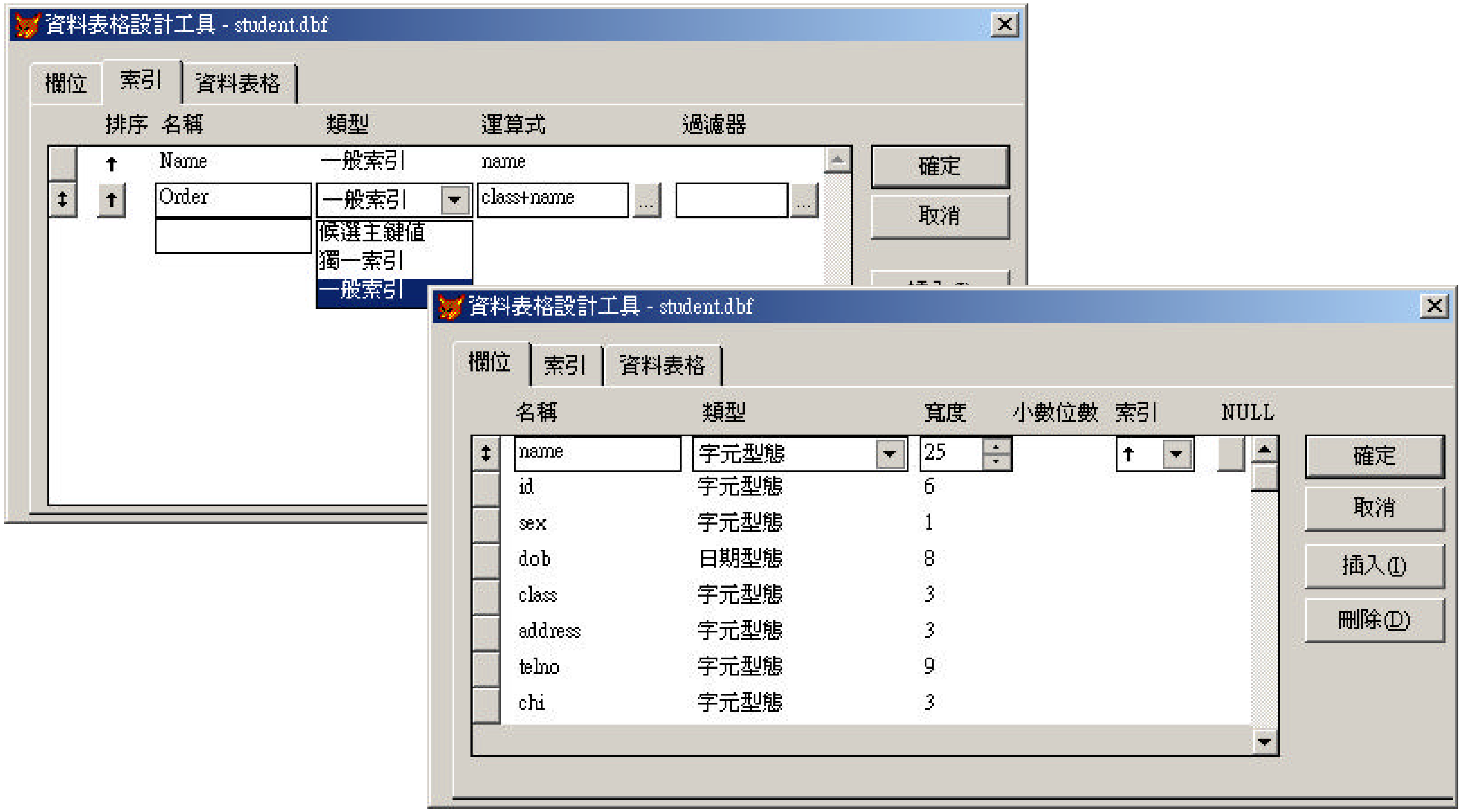
變數(R):

_alignment	C
_box	L
_indent	N
_lmargin	N
_padvance	C

來自資料表格(T): Student

Structural Compound Index

Creation of Structural Compound Index (cont’d)



資料表格設計工具 - student.dbf

欄位 | 索引 | 資料表格

排序	名稱	類型	運算式	過濾器
↑	Name	一般索引	name	
↑	Order	一般索引	class+name	

候選主鍵值
獨一索引
一般索引

資料表格設計工具 - student.dbf

欄位 | 索引 | 資料表格

名稱	類型	寬度	小數位數	索引	NULL
name	字元型態	25		↑	
id	字元型態	6			
sex	字元型態	1			
dob	日期型態	8			
class	字元型態	3			
address	字元型態	3			
telno	字元型態	9			
chi	字元型態	3			

Structural Compound Index

Types of Structural Compound Index

- Primary, candidate, regular and unique are four kinds of structural compound indexes.

Primary Index

- Primary index is used for the key field of a table.
- Only unique values can be used. Thus, it is possible to guard against users to enter duplicate values in a field where a primary index is defined.
- Only one primary index can be defined for a table.
- Primary index type is only available for database tables and NOT available for free tables.

Structural Compound Index

Candidate Index

- Candidate index is similar to Primary index for it also enforces unique values.
- However, there can be more than one Candidate index for a table.

Regular Index

- Regular index does not enforce unique values.
- It only determines the ordering of records.

Unique Index

- Unique index orders only a subset of records based on the first occurrence of a value in the specified field. It is seldom used in database application.

Single Index

Access of Single Index

- Since single index file would not be activated unless you request to do so. Hence you need to open it manually.
- *USE TableName INDEX IDXFileName*
  OR
  USE TableName
  SET INDEX TO IDXFileName
- USE drivers INDEX name.idx
- USE drivers INDEX name.idx, driverid.idx, idno.idx
- USE drivers
  SET INDEX TO name.idx, driverid.idx, idno.idx

Non-Structural Compound Index

Access of Non-Structural Compound Index

- **The non-structural compound index would NOT be activated when a table is open. Hence you need to specify where is the index file.**
- ***USE TableName INDEX CDXFileName***
  SET ORDER TO TagName OF CDXFileName ;
  [ASCENDING | DESCENDING]

-
  ```
  USE drivers INDEX abc.cdx
  SET ORDER TO name OF abc.cdx DESCENDING
  OR
  USE drivers
  SET INDEX TO abc.cdx
  SET ORDER TO name OF abc.cdx
  ```

Structural Compound Index

Access of Structural Compound Index

- **The structural compound index would be activated when a table is open. Hence you NO need to specify where to index file.**
- ***USE TableName***
  SET ORDER TO TagName ;
  [ASCENDING | DESCENDING]
  OR
  USE TableName ORDER TagName
- **USE drivers**
  SET ORDER TO name
  OR
  USE drivers ORDER name

Comparison

Type	Sorting	Indexing
Storage Requirement	Larger.	Smaller.
Speed	Slower as it involves the duplication of the whole file.	Faster as it involves duplicating the minimum amount of data to identify the records.
Convenience	Low as switching to another order needs to sort the table again.	High as switching to any order can be done very fast.
Usefulness	Low as a sorted table can only speed up searching for the expression used for sorting.	High as an index file can contain many tags which can speed up searching based on those expression used to build the tags.

Comparison

Single Index vs Non-Structural Compound Index

- Both indexes are independent and would NOT be activated automatically.
- Non-structural compound index allows more than one index tags.
- Reindexing is necessary for single index.

Structural Compound Index vs Non-Structural Compound Index

- Both indexes allow more than one index tags.
- Structural compound index is dependent (associated) with the table and would be activated automatically while non-structural compound index does not.

Questions to Think About

- Why SORT or INDEX?
- Problems of SORT
- Types of INDEX, properties of each INDEX
- Methods to create and access each INDEX

END