Value Buffers

The search and value buffers are used together to define:

- the search criteria to select a set of records using a FIND command (S1, S2, S4); and
- the range of values to be traversed by logical sequential read commands (L3/6, L9).

If a value buffer is provided, a search buffer is also expected. If it is not provided, Adabas will create a dummy search buffer to pair with the value buffer. For complete information about the relationships between the different types of ABD or buffer specifications, read *Understanding the Different Buffer Types*.

Only one search and value buffer pair should be specified in a single Adabas direct call.

The user provides the search expressions in the search buffer and the values which correspond to the search expressions in the value buffer.

In the value buffer, the user specifies the values for each descriptor specified in the search buffer.

If the search expression is a command ID, no corresponding entry is made in the value buffer.

- Value Buffer Syntax
- SQL Null Values and Indicators
- Sign Handling

Value Buffer Syntax

The values provided must be in the same sequence as the corresponding search expressions specified in the search buffer. All values provided must correspond to the standard length and format of the corresponding descriptor unless the user has explicitly overridden the standard length or format in the search buffer.

No intervening blanks or other characters such as a comma can be inserted between values in the value buffer. A period is not required to end the value buffer entry.

SQL Null Values and Indicators

When searching for fields defined with the NC (SQL null not counted) option, the search buffer field definition must contain a null significance (S) indicator and the corresponding value buffer argument value must display a two-byte binary null value indicator. See the section *S (Significance) and Null Indicators* for more information and examples of the null value indicator in the value buffer.

Sign Handling

Binary values are treated as unsigned numbers. Fixed-point, unpacked, and packed values are treated as signed numbers. Valid signs which may be provided are described in thissection:

- Fixed Value Signs
- Unpacked Value Signs
- Packed Value Signs

Fixed Value Signs

For fixed values, the sign is contained in bit 0 (high-order bit):

- 0 = positive
- 1 = negative (two’s complement)

Here are two fixed value sign examples showing the hexadecimal notation and the decimal equivalent:

```
00000005 = +5
FFFFFFFB = -5
```

Unpacked Value Signs

For unpacked values, the sign is contained in the four high-order bits of the low-order byte:

- C or A or F or E = positive (CAFE)
- B or D = negative (BD)

Here are two unpacked value sign examples showing the hexadecimal notation and the decimal equivalent:

```
F1F2F3  = +123
F1F2D3  = -123
```

Packed Value Signs

For packed values, the sign is contained in the four low-order bits of the low-order byte:

- A or C or E or F = positive
- B or D = negative

If a search value is being provided for a superdescriptor which is derived from a packed field, an F positive sign or a D negative sign must be provided.

Here are two packed value sign examples showing the hexadecimal notation and the decimal equivalent:

```
X'123F' = +123
X'123C' = +123
X'123D' = -123
```