- ncolumn ： 列数
  ncolumn 特性は CSVTable オブジェクトが表現する表の列数を示す。

7.5.2 BinaryTable オブジェクト

BinaryTable オブジェクトは、バイナリ表現された２次元の表形式データを受信し、その中から必要な部分表を取得するために用意されるオブジェクトである。

7.5.2.1 BinaryTable オブジェクトが扱うデータ

BinaryTable オブジェクトが扱う表は、1 個以上の固定長または可変長のレコード(行)から成り、レコードは 1 個以上のフィールド(要素)から成り、これらフィールドやレコードが連接しているバイナリ形式のファイルである。

可変長のレコードは、その先頭にそのレコード長を示す領域と、少なくとも 1 個の可変長のフィールドを持つ。可変長のフィールドの先頭には 1 バイト以上の長さ領域を設定する。

表中の文字の符号化方式は、EUC-JP および JIS X 0221 またはシフト JIS のいずれかとする。

7.5.2.2 BinaryTable オブジェクトのコンストラクタ

- コンストラクタ： BinaryTable オブジェクトを生成する。

文法：

```
BinaryTable new BinaryTable(
          input String table_ref,
          input String structure
)
```

引数：

`table_ref`　表ファイル指定
`structure`　表のフォーマット指定

戻り値：

生成された BinaryTable オブジェクト:　成功
null:　生成不可能

説明：

BinaryTable オブジェクトを生成し、 `table_ref` で指定したファイルを `structure` で指定したフォーマットで表として扱う。

新しく生成されたオブジェクトの [[Prototype]] 特性は、組み込みの BinaryTable プロトタイプオブジェクトに設定され、BinaryTable.prototype の初期値となる。

新しく生成されたオブジェクトの [[Class]] 特性は "BinaryTable" となる。

`table_ref` の記述は、第 9 章の名前空間の規定に従う。このオブジェクトは close() するまで有効である。

structure のフォーマット定義：

```
structure ::= lengthByte "," field[ "," field]*
lengthByte ::= decint | "0"
field ::= type ":" size
type ::=  "B" | "U" | "I" | "S" | "Z" | "P"
size ::= length unit
length ::= decint
decint::= digit [ digit0 ]*
digit0 ::= digit | "0"
digit ::= "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
unit ::= "B" | "b" | "V"
```

lengthByte　レコード長をバイト数で表した値。0 の場合は全て固定長フィールド。

size　unit が"B"または"b"のときフィールドの長さを表す。
unit が"V"の場合は可変長であり、フィールド長の入ったバイト数をあらわし、表にはフィールド長につづいてフィールドデータが入る。ただし可変長フィールドは type が"S"か"Z"の場合のみ。レコード長、フィールド長を表中に入れる場合には符号なし整数、ネットワークバイトオーダー（ビッグエンディアン）で符号化。長さの後ろにつづくデータの長さをバイト単位で入れる。

length　１０進文字で表す。１以上。

unit　フィールドの長さを length で表すときの単位
"B" ... Byte　単位はバイト
"b" ... bit　単位はビット
"V" ... variable　可変長フィールド

type　フィールドの型を表す
"B" ... Boolean
Boolean 型のデータ。長さは 1bit。0 が false、1 が true。unit 指定はビット指定("b")。手続き記述言語中では、Boolean オブジェクトとして扱う。
"U" ... Unsigned Integer
符号なし整数を表す。長さは最大 4Byte(32bit)。ネットワークバイトオーダー（ビッグエンディアン）。4Byte(32bit)の時、最上位ビットは常に 0(つまり、最大 2147483647 (0x7FFFFFFF))。長さが 8 ビット以上の場合、データの後ろ側がバイトアライメントに合っていなければならない。unit 指定はバイト指定 ("B") またはビット指定("b")。手続き記述言語中では、Number オブジェクトとして扱う。7 ビット以下の場合、バイトアライメントは合わせなくて

良い。

"I" ... Integer

符号付整数を表す。長さは最大 4Byte(32bit)。ネットワークバイトオーダー（ビッグエンディアン）。2 の補数表現。バイトアライメントにあっていなければならない。unit 指定はバイト 指定("B")のみ。長さは１バイト、2 バイトまたは４バイト。手続き記述言語中では、Number オブジェクトとして扱う。

"S" ... String

String 型のデータ。バイトアライメントにあっていなければならない。unit 指定はバイト指定 ("B")または可変長フィールド指定("V")。

手続き記述言語中では、String オブジェクトとして扱う。

"Z" ... ZipCode

郵便番号符号化のデータ（付録規定 B に規定。structure の length は付録規定 B.1 の length に、これ自身の長さを加えたものになる。）バイトアライメントにあっていなければならない。unit 指定は可変長フィールド 指定("V")のみ。手続き記述言語中では、Boolean オブジェクトとして扱う。

"P" ... Pad

データの隙間を表す。フィールドとして扱われない。フィールドをバイトアライメントにあわせるために使用。unit 指定はバイト指定(”B‘)またはビット指定("b")。

【例】フォーマット定義の例：

"1,B:1b,B:1b,U:3b,U:3b,S:1V,Z:1V"

1 バイトのレコード長を各レコード先頭に入れる。
以後、先頭から
ブール型 1 ビット
ブール型 1 ビット
符号なし整数 3 ビット
符号なし整数 3 ビット
文字列 可変長　長さデータ 1 バイト
郵便番号 可変長　長さデータ 1 バイト

の順で、レコード中にフィールドが並ぶ。

7.5.2.3 BinaryTable コンストラクタの特性

- BinaryTable.prototype ： プロトタイプ

BinaryTable.prototype の初期値は組み込みの BinaryTable プロトタイプオブジェクト(7.5.2.4)である。

この特性は、DontEnum 属性、DontDelete 属性、ReadOnly 属性を持つ。

7.5.2.4 BinaryTable プロトタイプオブジェクトの特性

BinaryTable プロトタイプオブジェクト自身は、BinaryTable オブジェクトであり、その値は NaN である。

BinaryTable プロトタイプオブジェクトの [[Prototype]] 特性の値は Object プロトタイプオブジェクトである。

- BinaryTable.prototype.constructor ： コンストラクタ

  初期値は、組み込みの BinaryTable コンストラクタ(7.5.2.2)である。

- BinaryTable.prototype.close()： BinaryTable オブジェクトの扱いの終了を宣言する。

  文法：

  ```
  Number BinaryTable.prototype.close()
  ```

  引数：

  なし

  戻り値：

  1:　　成功

  NaN:　失敗

  説明：

  表のための記憶領域などを解放する。

- BinaryTable.prototype.toString()： 表の 1 フィールドを文字列として出力する。

  文法：

  ```
  String BinaryTable.prototype.toString(
            input Number row,
            input Number column
  )
  ```

  引数：

  row　　行

  column　列

  戻り値：

  表の 1 フィールドに対応する文字列:　成功

  null:　　対応するフィールドが存在しない

  説明：

BinaryTable オブジェクトが表現する表の 1 フィールドを文字列として戻り値に返す。row, column はフィールドの位置を示す 0 以上の値である。
フィールドが文字列でない場合には、そのオブジェクトに toString() を適用した結果を戻り値に返すが、フィールドが ZipCode の場合は常に null を返す。

- BinaryTable.prototype.toNumber()： 表の 1 フィールドを数値として出力する。

文法：

```
Number BinaryTable.prototype.toNumber(
          input Number row,
          input Number column
)
```

引数：

row	行
column	列

戻り値：

表の 1 フィールドの数値:	成功
NaN:	対応するフィールドが存在しない

説明：

BinaryTable オブジェクトが表現する表の 1 フィールドを数値として戻り値に返す。row, column はフィールドの位置を示す 0 以上の値である。row, column で指定したフィールドが表に存在しない場合は NaN を戻り値として返すものとする。
フィールドが整数でない場合には、そのオブジェクトに toNumber() を適用した結果を戻り値に返すが、フィールドが ZipCode 型フィールドの場合は常に NaN を返す。

- BinaryTable.prototype.toArray()： 表中のレコードを Array として出力する。

文法：

```
Array BinaryTable.prototype.toArray(
          input Number startRow,
          input Number numRow
)
```

引数：

startRow	抜き出すレコードの開始位置
numRow	抜き出すレコード数

戻り値：

連続するレコードを格納した Array:	成功

null: 失敗

説明:

BinaryTable オブジェクトが表現する表の連続するレコードを Array オブジェクトとして取り出し、それらを要素とする Array オブジェクトを戻り値に返す。
レコードに対応する Array の各要素は、各フィールドの型に対応するオブジェクトであるが、ZipCode 型フィールドに関しては常に null を返す。
0 以上の値である startRow 番目のレコードから numRow に指定された数だけレコードを抜き出す。
対応する全てのレコードあるいは 1 部のレコードが表中に存在しない場合であっても、このメソッドは戻り値として Array オブジェクト（要素は numRow 個の Array オブジェクト）を返す。ただし、表に存在しないレコードに対応する Array オブジェクトは null である。

- BinaryTable.prototype.search()：条件を満たす表中のレコードを出力する。

文法:

```
Number BinaryTable.prototype.search(
          input Number startRow,
          [input Number searchedColumn,
          input Object compared,
          input Number operator]+,
          input Boolean logic,
          input Number limitCount,
          output Array resultArray
)
```

引数:

startRow　検索を開始するレコードの位置
searchedColumn
　　検索の対象となる列の位置
compared　比較の対象(String 型、Number 型、 Boolean 型のいずれか)
operator　比較条件
logic　　複数の検索条件の関係(true : OR　false : AND)
limitCount
　　検索で抜き出されるレコード数の制限値
resultArray
　　出力のレコードを格納する Array

戻り値:

検索を終了したレコードの位置: 成功

-1: limitCount に達せずに、すべてのレコードを検索した場合

NaN: 失敗

説明:

BinaryTable オブジェクトが表現する表から検索条件に合致するレコードを出力する。検索条件は searchedColumn, compared, operator の 1 つ以上の組で示される比較条件とそれらの関係を表す logic によって指定する。指定可能な最大の比較条件数は運用にて規定する。

結果は resultArray に返される。resultArray の各要素は、検索条件に合致するレコードに対応する Array オブジェクトである。これらのレコードに対応する Array オブジェクトの各要素は、各フィールドの型と値に対応するオブジェクトである。ただし、ZipCode 型フィールドに対応する Array オブジェクトの要素については、当該フィールドに対する比較条件が指定されている場合には指定された比較条件に基づいて判定された結果が Boolean オブジェクトとして出力され、比較条件が指定されていない場合には常に null を設定する。

startRow に指定された開始位置から limitCount で指定された数のレコードを抜き出した時点で検索を終了し、最後に抜き出したレコードのインデックスを戻り値として返す。limitCount に達せずに、すべてのレコードを検索した場合は戻り値に -1 を返す。

searchedColumn および compared および operator は可変引数リストであるが、これら 3 つの引数は必ず同時に指定しなければならない。また operator は以下のように定義され、operator の値によって compared の型を知ることができる。ただし、ZipCode と比較する compared は Number オブジェクトとし、郵便番号を 10 進 7 桁の整数とみなして扱う。また compared が String オブジェクトの場合、検索は文字単位で行い、Number オブジェクトの場合、signed で比較する。

searchedColumn で示されるフィールドの型が Unsigned Integer もしくは Integer である場合の operator の値と意味:

0: ＝ <compared>
1: ≠ <compared>
2: ＜ <compared>
3: ≦ <compared>
4: ＞ <compared>
5: ≧ <compared>
6: & <compared>（ビット毎の AND 論理演算）
7: | <compared>（ビット毎の OR 論理演算）
8: ^ <compared>（ビット毎の排他的 OR 論理演算）

9: ~& <compared>（1 の補数とのビット毎の AND 論理演算）
10: ~| <compared>（1 の補数とのビット毎の OR 論理演算）
11: ~^ <compared>（1 の補数とのビット毎の排他的論理演算）

searchedColumn で示されるフィールドの型が String である場合の operator の値と意味：

32: <compared> と完全に一致する
33: <compared> を含む
34: <compared> で始まる
35: <compared> で終わる
36: <compared> と完全に一致しない
37: <compared> を含まない

searchedColumn で示されるフィールドの型が Boolean である場合の operator の値と意味：

64: <compared> と等しい
65: <compared> と等しくない

searchedColumn で示されるフィールドの型が ZipCode である場合の operator の値と意味：

96: <compared> で表現される郵便番号を含む
97: <compared> で表現される郵便番号を含まない

注) FieldType が Unsigned Integer もしくは Integer の場合、&、|、^、～&、～|、～^ の時、結果が 0 か 0 以外かで判別する。0 なら false を返し、0 以外なら true を返す。

7.5.2.5 BinaryTable インスタンスの特性

BinaryTable のインスタンスは、BinaryTable プロトタイプオブジェクトの特性を継承し、[[Value]] 特性と nrow 特性、ncolumn 特性を保持する。

- nrow ： 行数
  nrow 特性は BinaryTable オブジェクトが表現する表の行数を示す。
- ncolumn ： 列数
  ncolumn 特性は BinaryTable オブジェクトが表現する表の列数を示す。

7.5.3 XML 文書オブジェクト

XML 文書オブジェクトは BML 文書の中で、任意の XML 文書を読み込み、ECMAScript を用いて DOM を用いたアクセスを可能とするとともに、DOM ツリーを XML 文書として外部ファイルや外部機器等に書き込むために用意されるオブジェクトである。

以下では、第 6 章に規定された方法での XML 文書の利用と区別するために、本節の規定によって利用される XML 文書を外部 XML 文書とよぶ。