Visual COBOL チュートリアル

ファイルハンドラーを使ったデータファイルアクセス

1. 目的

メインフレームやオフコンからオープンプラットフォームに COBOL 資産を移行した時に必要になる処理は、やはりデータファイルアクセスではない でしょうか。Visual COBOL を使用しないでこれらのデータファイルにアクセスする場合、OS が提供するローレベルな API を使ってカーネル ディスクドライバーを経由しファイルにアクセスします。ただし、この API が提供するのは低水準のバイトストリームファイルアクセスであり、 COBOL アプリケーションから発行されるレコード単位による I/O 処理には対応できません。Visual COBOL は、 COBOL アプリケーショ ンとOSの API の間にファイルハンドラーというインターフェースを介して COBOL アプリからの各種 I/O のハンドリングを行います。これにより、 オープンプラットフォームにおいても、これまで同様、レコード単位によるデータファイルへのアクセスが可能になります。データファイルの種類には 順編成ファイル、案引編成ファイルなどの種類が存在します。

順編成ファイルは、バッチ処理で扱うトランザクションファイルの操作に優れています。多くのバッチ処理は、夜間などのコンピュータ資源を占有で きる環境で実行されるため、排他制御などのオーバーヘッドを受けることなく効率的に実行できる点にメリットがあります。一方で、夜間のバッチ 処理は翌朝の業務開始時刻までには確実に処理が完了していることが必須であり、オンライン処理のレスポンスタイムと同様に、あるいは、そ れ以上に厳しい性能要求にさらされる処理となっています。たとえ 0.01 秒しかかからない処理でも 100 万回繰り返すことで 3 時間以上 になりますので、大量バッチ処理を定時までに完了させるためには慎重なプログラミングが必要となります。トランザクションファイル 1 件の読み 込みでも最も高速な手段が要求されるため、これを COBOL の順編成ファイルとして扱うことは最適な選択となります。

索引編成ファイルは、キー値で指定されたファイル内の固有のレコードにランダムにアクセスできる機能を持つファイル編成です。索引編成ファイルはマスターファイルとして使用されます。たとえば従業員マスターファイルは全従業員に関する様々な情報を保持していますが、従業員番号や従業員名などでランダムアクセスできなければなりません。

索引編成ファイルの概念は SQL を使い慣れた方には理解しやすいものです。実際、歴史的には索引編成ファイルは、データベースの概念 に先立って登場しており、その後複数の索引編成ファイルの有機的な統合を実現するためにデータベース管理システムが考案された経緯があ ります。

このチュートリアルでは、ファイルハンドラーを使用したファイルのアクセス方法、ソートユーティリティの概要、rebuild ユーティリティの使用方法を 学びます。

2. 前提条件

本チュートリアルは、下記の環境を前提に作成されています。

• 開発クライアント ソフトウェア

OS

Windows Server 2022 Standard Edition (64bit)

COBOL 製品

Visual COBOL 9.0 for Eclipse

チュートリアル用プログラム

下記のリンクから事前にチュートリアル用のプログラムやデータファイルをダウンロードして、任意のフォルダに解凍しておいてください。

プログラムおよびデータファイルのダウンロード

内容

- 1. 目的
- 2. 前提条件
- 3. チュートリアル手順の概要
 - 3.1. データファイルの準備
 - 3.2. Visual COBOL for Eclipse の起動とプロジェクト作成の準備
 - 3.3. ネイティブ COBOL プログラムの作成と動作確認
 - 3.4. JVM COBOL プログラムの作成と動作確認
 - 3.5. クラシックデータファイルツール
 - 3.6. ソートユーティリティ MFSOFT の確認
 - 3.7. ファイル管理ユーティリティ REBUILD の確認

3. チュートリアル手順の概要

3.1. データファイルの準備

- 1) チュートリアル用データファイルの用意
 - ダウンロードしたファイルを任意の場所に解凍します。work フォルダ、MFSORT フォルダ、REBUILD フォルダが入っていることを確認します。

3.2. Visual COBOL for Eclipse の起動とプロジェクト作成の準備

- 1) Visual COBOL for Eclipse の起動とプロジェクトの作成
 - ① スタートメニューより [Micro Focus Visual COBOL] > [Visual COBOL for Eclipse] を選択します。



② ワークスペースの選択画面は任意のワークスペースを指定して、[起動(L)] ボタンをクリックします。

3.3. ネイティブ COBOL プログラムの作成と動作確認

- 1) ネイティブ COBOL プロジェクトの作成とプログラムのインポート
 - Eclipse メニューより、[ファイル(F)] > [新規(N)] > [COBOL プロジェクト] を選択し、プロジェクト名に "FileHandlerTutorial"を指定して、[終了(F)] ボタンをクリックします。

COBOL プロジェクト ワークスペースまたは外部の場所にCOBOL ブ	ロジェクトを作成します。
プロジェクト名(P): FileHandlerTutorial	
プロジェクト テンプレートを選択	
132 ビッ 24 Micro Focus テンプレート [32 ビッ 24 Micro Focus テンプレート [64 ビッ	H H

- ② COBOL エクスプローラーにて作成した「FileHandlerTutorial」プロジェクトを右クリックし、コンテクストメニューから [インポート(I)] > [インポート(I)] を選択します。
- ③ [一般] > [ファイル・システム] を選択し、[次へ(N)] ボタンをクリックします。

インポート・ウィザードの選択(S):
フィルタ入力

④ [参照(R)...] をクリックし、エクスプローラーから解凍したフォルダ配下の work フォルダを指定します。
 「FileDemo1.cbl」にチェックを入れて [終了(F)] ボタンをクリックします。

- 2) Eclipse 全体の文字コード設定
 - Visual COBOL 9.0 より Shift-JIS を指定して日本語を表示する場合、文字コードの指定を明確に行う必要があります。最初に、[Window]メニュー > [設定]より[一般] > [ワークスペース]とナビゲートし、テキストファイルエンコードを「MS932」に変更し、[適用して閉じる]ボタンをクリックします。

7ィルタ入力	ワークスペース 🗘
✓ 一般 へ Capabilities	ワークスペースの開始およびシャットダウン設定については、 <u>開始およびシャットダウン</u> を参照してください。
Schema Associations UI フリーズ・モニタリング	□ ネイティブのフックまたはポーリングを使用して更新(R)
> User Storage Service	✓ アクセス時に更新(S)
Web ブラウザ	□ 無関係なプロジェクトを常にプロンプトなしで閉じる(C)
> エディタ	ワークスペース保管間隔(分)(W): 5
+-	
クイック検索	ウィンドウのタイトル
20-70/20 72-200-07	VID-77パー7名を表示(F): eclipse-workspace
+	
> ヤキュリティー	
トレース	 リークスペースのフルバスを表示(F): C¥Users¥Administrator¥eclipse-workspace
> ネットワーク接続	≥ プロダクト名を表示
ハンドラーをリンク パースペクティブ プロジェクト・ネーチャー > ワークスペース > 間始およびシャットグウン > 外親	ブロジェクトを職く際に、参照するブロジェクトを職く、ブロンプト 〜 不明なブロジェクトの性質を以下のように報告(A): 警告 〜
検索 比較/パッチ	システム・エクスプローラーを起動するコマンド(X): explorer /E,/select=\${selected_resource_loc}
> Ant	
AspectJ Compiler	テキスト・ファイル・エンコード(T) 新規テキスト・ファイルの行区切り文字(F)
> Gradle	 デフォルト(U) (MS932) デフォルト(E) (Windows)
> J2EE	○その他(O): MS932 ∨ ○その他(H): Windows ∨
> Java	
> Java 永統15	
< >	デフォルトの復元(T) 適用(L)
? 🗅 🗹 🖲 🛇	通用して閉じる キャンセル

※Preference Recorder ダイアログが表示されたら、[キャンセル] を選択してください。

- 3) プロジェクトプロパティの指定
 - プロジェクトの構成を変更します。COBOL エクスプローラーにて作成した「FileHandlerTutorial」プロジェクトを右 クリックし、コンテクストメニューから[プロパティ(R)]を選択します。
 - プロパティ設定ダイアログが表示されます。[Micro Focus] > [ビルド構成] > [COBOL] をクリックし、[構成の固有な 設定を可能にする(C)] にチェックを入れたうえで、[ソースエンコーディング] を「ANSI」、[.GNT にコンパイル] を「は い」、[追加指令]に、「ASSIGN(EXTERNAL)」を入力します。

 Micro Focus ビルダー ビルドロー 	☑構成の固有な設定を可能にする(C)	
 ビルド構成 COBOL イベント ディブロイ 	 」上位レベルの設定を上書きする (マージしない) 2イルタテキストを入力 	
ビルド環境 > リンク > プロジェクト設定 #5000000	設定 一般 文字セット	値 ASCII
2 実行時構成 サーパー タスク・タグ ビルダー プロジェクト・ネーチャー	ソース エンコーディング COBOL 方言 ソース フォーマット デバッグ用にコンパイル EXIT PROGRAM を GOBACK として処理 詳細	ANSI インドログログ Micro Focus 固定 はい ANSI いいえ
フロジェクト・ファセット プロジェクト参照 > 検証 実行/デバッグ設定	- GNT にコンパイル ▼ 出力 指令ファイルを生成する リストファイルを生成	はい いいえ いいえ
	コート バルレッジを有効にする プロファイラを有効にする ▼ エラー/警告 警告レベル 最大エラー数	Taise false 回復可能なエラーを含める(レベル E) 100
	 ✓ 追加指令 追加指令 	ASSIGN(EXTERNAL)

③ 次に [ビルド構成] > [COBOL] > [リンク] をクリックし、 [ターゲットの種類] から「すべて INT/GNT ファイル」を選択し、[適用して閉じる] をクリックします。

✔ ビルド構成		
> COBOL	設定	値
イベント	✓ Linkage	
	出力の名前	FileHandlerTutorial
ビルド環境	出力パス	New Configuration.bin
> リンク	エントリポイント	
> プロジェクト設定 地合の2000	ターゲットの種類	すべて INT/GNT ファイル

④ 自動的にビルドが行われます。実行ファイルが作成されていることを確認します。

<mark>සු c</mark>	🛿 🗞 t 🍢 A 🔜 t 🖓 🗆
	✓ 📄 🚖 🔩 ♦
~ 🕑	FileHandlerTutorial
~	🧧 COBOL プログラム
	> FileDemo1.cbl
~	New Configuration.bin
	FileDemo1.gnt
	FileDemo1.gnt.1.tlog
	FileDemo1.idy

- 4) プログラムの実行
 - ① 「FileDemo1.gnt」上で右クリックし、コンテクストメニューから [実行(R)] > [実行の構成(N)...] を選択します。

Q.	Coverage A	s >	
0	実行(R)	> (3)	I COBOL アプリケーション
脊	デバッグ(D)	>	実行の構成(N)
	フロファイル())	

② [COBOL アプリケーション] を選択し、マウスの右クリックにてコンテクストメニューを開き、[新規構成] をクリックします。

搣 COBOL JVM リモート アプリ	リケーション 📃 🛛 🍸 - フィルタ・オプ
🚾 COBOL アプリケーシ	÷⊊-坦 堆 成().40
🔄 🔄 COBOL ユニット テス	
😨 Debug Adapter La 🖻	新規プロトタイプ(P)
😑 Eclipse アプリケーショ 🐞	エクスポート(X)

③ [名前] に "FILEDEMO" を入力し、[環境] タブを選択し、 [追加] ボタンをクリックします。[変数]
 に "DEMOFILE"、値には "C:¥FileHandling¥Native¥DemoIDX.dat"を指定して、[OK] ボタンをクリックします。※任意のフォルダを指定できます。変更した場合は以降の手順でも同様の値に変更してください。

変数:	DEMOFILE	I
值:	C:¥FileHandling¥Native¥DemolDX.dat	

- ④ エクスプローラーより「C:¥FileHandling¥Native」ディレクトリを作成します。
- ⑤ [実行] ボタンをクリックすると、DOS プロンプトが表示されプログラムが実行され、ファイル作成が行われます。
 「C:¥FileHandling¥Native¥DemoIDX.dat」が作成されているか確認します。
- 5) 実行結果の確認
 - Windows のスタートメニューから [Micro Focus Visual COBOL] > [クラシックデータファイルツール] を選択します。
 - ② [ファイル(F)] メニュー から [開く(O)] を選択し、「C:¥FileHandling¥Native¥DemoIDX.dat」をオープンします。
 - ③ 以下のようにファイルの中身が表示されます。



3.4. JVM COBOL プログラムの作成と動作確認

- 6) JVM COBOL プロジェクトの作成とプログラムのインポート
 - Eclipse メニューより、[ファイル(F)] > [新規(N)] > [COBOL JVM プロジェクト] を選択し、プロジェクト名に "FileHandlerTutorialJava"を指定して、[終了(F)] ボタンをクリックします。

プロジェクト名(<u>P</u>): FileHandlerTutorialJava	
☑ デフォルト・ロケーションの使用(型)	
ロケーション(<u>L</u>): C:¥WorkSpace¥FileHandlerTutorialJava	参照(<u>R</u>)

- ② COBOL エクスプローラーにて作成した「FileHandlerTutorialJava」プロジェクト配下の src フォルダを右クリック し、コンテクストメニューから [インポート(I)] > [インポート(I)] を選択します。
- ③ [一般] > [ファイル・システム] を選択し、[次へ(N)] ボタンをクリックします。



- ④ [参照(R)...]をクリックし、エクスプローラーから解凍したフォルダ配下の work フォルダを指定します。
 「FileDemo1.cbl」にチェックを入れて [終了(F)] ボタンをクリックします。
- 7) プロジェクトプロパティの指定
 - プロジェクトの構成を変更します。COBOL エクスプローラーにて作成した「FileHandlerTutorialJava」プロジェクトを右クリックし、コンテクストメニューから [プロパティ(R)]を選択します。
 - ② プロパティ設定ダイアログが表示されるので、[Micro Focus] > [ビルド構成] をクリックしたうえで以下の変更を行います。

[パッケージ名にディレクトリ階層を反映する] に「いいえ」

[追加指令] に、「ASSIGN(EXTERNAL) ILNAMESPACE(jp.microfocus.demo)」

/ IVM	
動的呼び出しを使用	10103
パッケージ名にディレクトリ階層を反映する	いいえ
インクリメンタル ビルドの使用	いいえ
パッケージ ターゲットを作成する	いいえ
ビルド後にJAR ファイルを作成する	いいえ
JAR 出力フォルダ	dist
JAR ファイル名	FileHandlerTutorialJava.jar
出力ディレクトリ名	bin
✓ エラー/警告	
警告レベル	重大なエラーだけ(レベル S)
最大エラー数	100
、注册149条	
追加指令 加指令	ASSIGN(EXTERNAL) ILNAMESPACE(jp.microfocus.dem
追加指令 追加指令 ンパイラに渡す追加のCOBOLコンパイラ指令です DBOL コンパイル設定: HARSET"ASCII" SOURCE-ENCODING"UTF8" DIAL (ITPROGRAM"ANSI" jarFolder=dist FileHandlerT RROR"100" ASSIGN(EXTERNAL) ILNAMESPACE(jj	ASSIGN(EXTERNAL) ILNAMESPACE(jp.microfocus.den
追加指令 追加指令 ジパイラに渡す追加のCOBOLコンパイラ指令です DBOL コンパイル設定: HARSET"ASCII" SOURCE-ENCODING"UTF8" DIAL KITPROGRAM"ANSI" jarFolder=dist FileHandlerT RROR"100" ASSIGN(EXTERNAL) ILNAME\$PACE(jj	ASSIGN(EXTERNAL) ILNAMESPACE(jp.microfocus.den く ECT"MF" SOURCEFORMAT"variable" NOLIST anim futorialJava.jar outputDirectory=bin NOWARNING MAX- p.microfocus.demo) デフォルトの復元(I) 適用(L)

③ [適用して閉じる] ボタンをクリックすると、自動的にビルドが行われます。

8) プログラムの実行

 「FileHandlerTutorialJava」プロジェクト上で右クリックし、コンテクストメニューから[実行] > [実行の構成(N)...] を選択します。

	coreagero	•		
0	実行(R)	>	JVM	1 COBOL JVM アプリケーション
*	デバッグ(D)	>	J	2 Java アプリケーション
	プロファイル(P)	>		実行 の構成(N)

- ② 左側の [COBOL JVM アプリケーション] 上で右クリックし、コンテクストメニューから [新規構成(W)] を選択します。
 名前は、"FileDemoJava" と入力します。
- ③ [メイン] タブの [メイン・クラス(M):] 横にある [検索(S)] ボタンをクリックし、「FILEDEMO1 jp.microfocus.demo」を選択し、[OK] ボタンをクリックします。
- ④ [環境] タブを選択し、 [新規(E)] ボタンをクリックします。[変数(N)] に "DEMOFILE"、[値(V)]に
 は "C:¥FileHandling¥JVMCOBOL¥DemoIDX.dat" を入力し、[OK] ボタンをクリックします。

6 C. +1 IICITalian	15+0 (III CODOL+Dellio1DI. aat
🖸 メイン (🛛= 引数 (🛋 JRE (🔩 クラスパス 💱 ソース 🚾 環境 📃 共通(C)
設定する環境変数(S)	
変数	值
DEMOFILE	C:¥FileHandling¥JVMCOBOL¥DemolDX.dat
	-

- ⑤ エクスプローラーより「C:¥FileHandling¥JVMCOBOL」ディレクトリを作成します。
- ⑤ [実行(R)] ボタンをクリックすると、プログラムが実行され、ファイル作成が行われます。
 「C:¥FileHandling¥JVMCOBOL¥DemoIDX.dat」が作成されているか確認します。
- 9) 実行結果の確認
 - ① Windows メニューから [Micro Focus Visual COBOL] > [クラシックデータファイルツール]を選択します。
 - ② [ファイル(F)] メニュー から [開く(O)] を選択し、「C:¥FileHandling¥JVMCOBOL¥DemoIDX.dat」をオープ

ンします。

③ 以下のようにファイルの中身が表示されます。



3.5. クラシックデータファイルツール

- データファイルツールは COBOL ファイル編成の変換、ファイルのブラウズ、レコードの編集等の機能を持った GUI ツールです。
- 1) クラシックデータファイルツールの起動とファイルの読み込み
 - ① 前の章で使用したファイルをオープンしていない場合は再度オープンします。
 - ② Windows のスタートメニューから[Micro Focus Visual COBOL] > [クラシックデータファイルツール]を選択します。
 - ③ [ファイル(F)] メニュー から [開く(O)] を選択し、「C:¥FileHandling¥JVMCOBOL¥DemoIDX.dat」をオープ ンします。



- 2) デバッグ情報ファイルからレコードレイアウトを作成
 - Windows のスタートメニューから [Micro Focus Visual COBOL] > [Visual COBOL コマンドプロンプト(32bit)] を選択します。



② 「Data File Structure Command Line ユーティリティ」を使用してデバッグ情報ファイルからレコードレイアウトを作成します。下記の通りタイプしてレコードレイアウトを生成します。レイアウトファイルは、指定した idy ファイルと同じフォルダに出力されます。

dfstrcl "<FileHandlerTutorial ワークスペースフォルダへのパス>¥New_Configuration.bin¥FileDemo1.idy" /d F-REC

<FileHandlerTutorial ワークスペースフォルダへのパス> この部分は、各自の環境に合わせて変更してください。

```
ファイルハンドラーを使ったデータファイルアクセス Eclipse 編
```

例えば、FileHandlerTutorial ワークスペースフォルダが "C:¥WorkSpace¥FileHandlerTutorial" の場合は、 以下のようになります。

dfstrcl "C:\WorkSpace\FileHandlerTutorial\New_Configuration.bin\FileDemo1.idy" /d F-REC

補足)

Eclipse IDE 上で FileHandlerTutorial ワークスペースへのフォルダパスを、以下の手順で確認できます。

- ① FileHandlerTutorial プロジェクトを選択し、マウスの右クリックにて、[プロパティ(R)]をクリックします。
- ② 画面左側より [リソース] を選択すると、右側の [ロケーション(L)] にワークスペースへのフォルダパスが表示されます。
- 3) レコードレイアウトを適用し可視化できないデータを確認
 - ① [ファイル(F)] メニュー > [データファイルエディタ(D)] > [レコードレイアウトのロード(L)...]を選択します。

データファイル エディタ(<u>D)</u> レコードレイアウトのロード(<u>L</u>).	
レコードレイアウトの関連付け	t(<u>A</u>)

② 前のステップで作成した「FileDemo1.str」ファイルを指定します。

注意)

このファイルは、前手順にて dfstrcl コマンドで指定した idy ファイルと同じフォルダにあります。

③ ファイルの中身がレコード単位で確認できるようになります。



④ [表示(V)] メニュー > [データファイルエディタ(D)] > [16 進表示(H)] を選択します。

表示(V)	ファイル(F)	検索(S)	7	プショ	ョン(0)	ツール(T)	ウィンドウ(W)	VIL
データ	7ファイル エディ	(夕(<u>D</u>)	×	0	ファイル	情報(<u>l</u>)		
<u>I</u> MS	DB エディタ		۲		表示の フォーマ	同期(<u>S</u>) ット済ペイン	Ctrl+ の更新(R)Ctrl+	γ ·F
ドッキ	シグできるウィ	ンドウ(<u>K</u>)			16進表	(<u>H</u>)	Alt+F	2

⑤ データの内容が16進でも表示されます。



- 4) 他のファイル編成に変換
 - ① データファイルを開いているウィンドウを一旦閉じます。



[ツール(T)] メニュー > [変換(C)] を選択します。

)	ツール(T)	ウィンドウ(W)	ヘルプ
	変換	(<u>C</u>)	

③ 「データファイルの変換」ウィンドウが表示されます。「入力ファイル」セクションの「参照(B)…」ボタンをクリックし、エクスプローラーより「C:¥FileHandling¥Native¥DemoIDX.dat」をオープンします。

-入力ファイル ファイル名(៲):					参照(B)
形式:	IDXFORMAT(8)	文字集合(S):	ANSI	•	ファイルの言羊糸田(E)…
編成:	不明				キーの表示(Y)

④ [ファイルの新規作成] セクションの [参照(R)...] ボタンをクリックし、エクスプローラーより

「C:¥FileHandling¥Native¥DemoIDX.txt」を指定します。次に[編成(O)]を「行順」に変更します。

┌ ファイルの新規作成		
ファイル名(N):	C:¥FileHandling¥Native¥DemoIDX.txt	参照(R)
形式(F):	Micro Focus ▼ 文字集合: ANSI	
編成(O):	行順	キーの定義(K)

⑤ [変換(C)] ボタンをクリックします。

<u>変換(C)</u> キャンセル

⑥ 「データファイルの変換」ウィンドウが表示され、変換処理が完了しますので [閉じる] ボタンをクリックします。

データファイルの変	换			?	Х
- ファイル 読込み中: 書込み中: ステータス	C:¥FileHandlir C:¥FileHandlir 変換完了	ng¥Native¥DemoIDXda ng¥Native¥DemoIDX.tx	t		
	だしコード: ドの拒否:	10 0			
	: 1-443	停止	ОК	閉じる	,

⑦ メモ帳で作成したファイルを開きます。各レコードが改行で区切られた行順ファイルになっていることを確認します。



3.6. ソートユーティリティ MFSOFT の確認

ソートユーティリティ MFSORT は、IBM メインフレームの DFSORT とコマンド互換のあるソート・マージユーティリティです。

- 1) ソート用ファイルの用意
 - ① 解凍したフォルダに MFSORT フォルダが含まれていることを確認します。
 - ② MFSOFT フォルダを C ドライブ配下に配置します。例: C:¥MFSORT
- 2) クラシックデータファイルツールの起動と読み込み
 - ① クラシックデータファイルツールを起動して MFSORT フォルダ配下の「airports.dat」を開きます。
 - ② [ファイル編成(O)] に "レコード順 固定長" が選択されていることを確認し、[最大の長さ(X)] に "96" を入力した うえで、[OK] をクリックします。

- ファイル定義 ファイル編成(O):	レコードノ順 - 固定長 🛛 💌
最小の長さ(M)	0 🕂 最大の長さ(X) 96 🕂
	ОК

以下のようなダイアログが表示された場合は、[OK]をクリックします。

Data File Tools		×
後でこのファイルをもう一度開くとき ブロファイルへ保存しますか? ブロファ 名前で保存されます。	に詳細情報を再入力しな ァイルは 'C:¥MFSORT¥air	べてすむように ports.PRO' の
	(はい(Y)	いいえ(N)

- ③ [ファイル(F)] > [データツールエディタ(D)] > [レコードレイアウトのロード(L)...]を選択し、MFSORT フォルダ配下の 「AIRPORTSEQ.STR」ファイルを指定して、[開く(O)]をクリックします。
- ④ ファイルの中身がレコード単位で確認できるようになります。

ANSI ▼ +-なし ▼ IEEE	✓ 10			
Forestville	Forestvillin	F-REC-DEFAULT	ν ί ζόι ΟΚ	
Kuu Magas	Nazran	フィールド名	形式	値
06A Matan Eigld Municipa	Truckee	A01 E-BEC	10204	
049 Elizabethten Municipa	FLizabotht.	wall2 Ecode	PIC X(A)	10H
NP2 Shoestring Aviation	Stewartsto	well2 F-name	PIC X(30)	Fortman Airport
0S9 Jefferson County Int	Port Towns	ven2 F-city	PIČ X(30)	St Marvs
10C Galt Field Airport	Greenwood	✓ 02 E-countiry	PIC X(20)	United States
19A Jackson County Airpo	Jefferson	Ø02 F-seo		
1B9 Mansfield Municipal	Mansfield	Ø03 F-latitude		
ICS Clow International A	Bolingbrool	√04 F-lat-sign	PIC X	+
IUH Fortman Airport	St. Marys	✓04 F-lat-degs	PIC 9(3) COMP-3	040
241 Sumppoo County Airpar	Live Oak		PIC 9(6) COMP-5	555325
243 Suwannee County Arrp 2.19 Duincy Municipal Air	Quincy	Ø03 F-longitude		
369 Atmautluak Airport	Atmaut Luak	↓ 🗸 🗸 🗸 🗸 🗸 🗸 🗸 🗸 🗸	PIC X	-
38W Lynden Airport	Lynden	v∕04 F-long-degs	PIC 9(3) COMP-3	084
3W2 Put-in-Bay Airport	Put-in-Bay	∲04 F-long-mins	PIC 9(6) COMP-5	386618
40J Perry-Foley Airport	Perry			
49A Gilmer County Airpor	Ellijay			
444 Polk County Airport	Cedartown			
4A9 Isbell Field Airport	Fort Payne			
417 Futnam County Airpor	Dell			
541 DeFuniak Springe Air	DeFunial Sr			
<	>			
🔏 airports.dat (固定 長さ レコード珈)	×	*		

- 3) ソート順の変更
 - ① このファイルは「F-code」と呼ばれる空港コードの昇順でソートされています。これを「F-name」という空港名でソートしてみます。

 ソート用のコマンドファイルを用意します。1 行目はインプットするデータファイルの指定、2 行目はアウトプットするデータ ファイルの指定、3 行目がソートするフィールド名の指定です。ここでは「F-name」を昇順で指定するので 5 バイト 目から 30 バイト、昇順(ascending)に並べ替えを指定しています。

use airports.dat record (f, 96) give sorted.dat ORG SQ sort fields (5, 30 CH, a)

補足)

上記ファイルは、MFSORT フォルダ配下内の airport.srt ファイルとして保存されています。

- ⑧ Windowsのスタートメニューから [Micro Focus Visual COBOL] > [Visual COBOL コマンドプロンプト(32-bit)]
 を選択します。
- ⑨ CD C:¥MFSORT」を実行し、C:¥MFSORT フォルダまで移動します。
- 1 mfsort コマンドを実行します。コマンドラインに「mfsort take airport.srt」とタイプしてリターンキーを押します。
 注意)

クラシックデータファイルツールなどで airports.dat を開いている場合、上記コマンドを実行する前に閉じてください。

- 4) 実行結果の確認
 - ① 「SYSOUT」というファイルに実行結果ログが入っているのでメモ帳を起動して内容を確認します。

② クラシックデータファイルツールを起動して「sorted.dat」を開きます。

さきほど同様、[ファイル編成(O)] に "レコード順 – 固定長"を選択、[最大の長さ(X)] に "96"を指定して [OK] をクリックします。

- ファイル定義 ファイル編成(O):	רבע	*順 - 固定長	-
最小の長さ(M)		最大の長さ(X)	96
		ОК	キャンセル

以下のダイアログが表示された場合は、[はい(Y)] をクリックします。 Data File Tools

?	後でこのファイルをもう一度開くときに詳細情報を再入力しなくてすむように プロファイルへ保存しますか? ブロファイルは 'C;¥MFSORT¥sorted.PRO' の名 前で保存されます。'

(はい(Y)	いいえ(N)

 ③ [ファイル(F)]メニュー > [データファイルエディタ(D)] > [レコードレイアウトのロード(L)…] を選択し、MFSORT フォルダ 配下の「AIRPORTSEQ.STR」ファイルを指定します。

I.	ファイル(F)	検索(S)	オプショ	ン(O)	ツール(T)	ウィンドウ(W)	へルプ(H)	
	データファイル エディタ(<u>D</u>)				レコードレイアウトのロード(<u>L</u>)				
_			•		レコードレイア	'ウトの関連付け(<u>A</u>)		
	* ILLL	-	•••						

④ 空港名でソートされていることを確認します。

Ą	-86.//50556	629 ·	^	F-REC-DEFAULT		レイアウト OK	
TRI	7 Novembre	Tabarka	1	フィールド名	形式		値
ĹĊĞ	A Coruna	La Coruna		♦01 F-REC			
AAL	Aalborg	Aalborg		v≉02 F-code	PIC X(4)		ABF
AAR	Aarhus	Aarhus		✓02 F-name	PIC X(30))	Abaiang Atoll Airpor
JEG	Aasiaat	Aasiaat		v¢02 E-city	PIC X(30))	Abaiang Atoll
ARD	Abadan	Abadan		vive vert vert vert vert vert vert vert ve	PIC X(20))	Kiribati
ABE	Abaiang Atoli Airpor	Abaiang Atoli		Ø02 F-geo			
	Abatafard	Abakan		●03 F-latitude	DIA V		
W G	Abdul Rachman Saleh	Malang		v9U4 F-lat-sign	PIC X	OOND O	+
AFH	abeche	8heche		vØU4 F-lat-degs	PIC 9(3)	COMP-3	001
SNII	Abel Santamaria	Santa Clara		ØU4 F-lat-mins	PIC 9(6)	COMP-5	000008
AFA	Abemama Atoll Airpor	Abemama		ØU3 F-Tongitude	DIA N		
ABR	Aberdeen Regional Ai	Aberdeen		v∕9U4 E-long-sign	PIC X	OOND O	+
AHB	Abha	Abha		v∕9U4 E-long-degs	PIC 9(3)	COMP-3	173
4BJ	Abidjan Felix Houpho	Abidjan		ØU4 F-long-mins	PIC 9(6)	COMP-5	000004
ABI	Abilene Rønl	Abilene					
CUS	Abraham Gonzalez Int	Ciudad Juarez					
SPI	Abraham Lincoln Uapi	Springtield					

3.7. ファイル管理ユーティリティ REBUILD の確認

ファイル管理ユーティリティ REBUILD は、COBOL のファイル編成の変換、索引ファイルの索引再編成、破損した索引ファイルのリビルド機能を提供するコマンドラインのユーティリティです。本チュートリアルでは索引ファイルの再編成を行います。

- 1) ファイルの用意
 - ① 解凍したフォルダに REBUILD が含まれていることを確認します。
- 2) rebuild コマンドによるインデックスの再編
 - Windowsのスタートメニューから [Micro Focus Visual COBOL] > [Visual COBOL コマンドプロンプト(32-bit)] を選択します。
 - ② コマンドプロンプト上で、先ほど確認した REBUILD フォルダ配下まで移動します。
 - ③ rebuild コマンドを実行します。コマンドラインに「rebuild airportsIdxOld.dat,airportsIdxNew.dat」とタイプし てリターンキーを押します。
- 実行結果の確認
 - 実行が終わるとメッセージが表示されます。
 再構成が完了しました レコード読み込み = 5383

再編成前のファイル airportsIdxOld.dat はレコードの追加、更新、削除を繰り返していたため、ファイルサイズが大き くなってしまっていました。rebuild コマンドにて、再編成が行われ、不要領域が解放されファイルサイズが縮小されたこと が確認できます。

irportsldxNew.dat	2020/04/16 16:39	COBOL データファイル	604 KB
airportsldxOld.dat	2016/01/14 16:47	COBOL データファイル	607 KB

WHAT'S NEXT

● 本チュートリアルで学習した技術の詳細については製品マニュアルをご参照ください。

免責事項

ここで紹介したソースコードは、機能説明のためのサンプルであり、製品の一部ではございません。ソースコードが実際に動作するか、御社業務に適合するかなどに関しまして、一切の保証はございません。 ソースコード、説明、その他すべてについて、無謬性は保障されません。 ここで紹介するソースコードの一部、もしくは全部について、弊社に断りなく、御社の内部に組み込み、そのままご利用頂いても構いません。 本ソースコードの一部もしくは全部を二次的著作物に対して引用する場合、著作権法の精神に基づき、適切な扱いを行ってください。