WHITE PAPER

SQL Anywhere® Studioとユニコード

ユニコードとは ？

ユニコードは、Unicode Consortiumが策定している文字集合 (UCS) の 1 つで、ユニコードを使用すると書かれた文字やテキストを全てシングル・バイト文字セットで表現できます。ユニコード 3.0 仕様では、100 万種類以上の文字を表現できます。
「ユニコード」という用語は、特定の 1 つのコード化方式を示すものではなく、文字を表現するための複数のコード化方式があります。

UCS-2

多くの人が「ユニコード」と呼ぶものが、このUCS-2で、UCS-2とは、固定幅、つまり 1 つ 1 つの文字が 2 バイト (16 ビット) でコード化されているコード化方式ということです。この方式では、潜在的には 65,636 文字を表現できますが、予約済みの領域や未使用の領域もあります。
UCS-2 は、現在Windows XP/2000/NT オペレーティング・システムおよび Windows CE オペレーティング・システム、またJava プログラミング言語でも使用されています。

UCS-4

UCS-4 も固定幅コード化方式で、4 バイトを使用して 1 つの文字を表現します。そのため、莫大な数の文字を表現できますが、逆に記憶領域にかかる費用も大きくなるため、一般的には使用されていません。
UNIX の一部のバージョンでは、UCS-4 の何らかのサポートをしていますが、SQL Anywhere Studioでは UCS-4 はサポートしていません。

UTF8

UTF とは、ユニコード変形フォーマットのことです。UTF ユニコードのコード化は、UCS コード化とは異なる方式を採用しており、マルチバイト・コード化方式と呼ばれています。UTF のコード化の文字に必要な記憶領域は、一定ではありません。
UTF8 では、1 つの文字に必要な記憶領域は 1 ～ 4 バイトです。文字を構成する最初のバイトとそれに続く「後続のバイト」には、異なる文字範囲が使用されます。
UTF8 のコード化方式は、次の理由により、「通常の」C/C++ プログラムでの処理が比較的簡単です。

- 7 ビット U.S. ASCII 文字セットがサブセットである。
- 「後続のバイト」には 7 ビット U.S. ASCII 文字がない。
- null (0) バイトが埋め込まれていない。

文字列を詳細に操作しないプログラムでは、ほとんど、またはまったく修正せずに UTF8 のデータを使用できる場合があります。文字列のサブセットの処理、文字列の分割、7 ビット U.S. ASCII 以外の文字の検索を行うプログラムでは、もっと多くの修正が必要な場合があります。

UTF-16

UTF-16 のコード化方式は、特殊な「代理ペア」を使用し、100 万種類以上の新しい文字を表現できる UCS-2 のコード化方式です。1 つの文字の記憶領域に 16 ビットのワードが 1 つまたは 2 つ必要な場合があります。
UTF-16 の文字セットは、一般的には使用されていません。SQL Anywhere は UTF-16 をサポートしていません。
UCS-2、UCS-4、UTF-16 の各コード化方式は、「通常の」C/C++ プログラムでは処理が比較的困難です。通常は、文字や文字列のすべての処理を設計し直す必要があります。標準的な char タイプの処理は機能しません。コード化した文字に 0 バイトが含まれ、それが null ターミネータ文字に見える場合があるためです。通常は、wide char タイプが使用されます。

目次

ユニコードのサポートについて

サポート・ユニコード：

Adaptive Server Anywhere	ユニコードUTF8をサポート
	データ・ソート機能強化のため、SORTKEY関数とCOMPARE関数を提供
Adaptive Server AnywhereのODBCドライバ OLE DBドライバ	ユニコードに対応し、ユニコードUCS-2文字セットをサポート
Mobile Link	UCS-2文字セットをサポート

例外：

- Adaptive Server Anywhereでは、識別子（テーブル名やカラム名）、データ、内部記憶領域に関して、ユニコードUSC-2文字セットのサポートはしていません。
- Embedded SQLのAPIは、識別子やデータに関するUCS-2をサポートしません。
- UTF8照合のネイティブのソート順は、最初のバイトにのみ設定できます。

このホワイトペーパーでは、SQL Anywhere Studioにおけるユニコードのサポートについて詳細に説明します。Adaptive Server Anywhereに焦点をあてていますが、Mobile LinkやUltra Lightでのユニコードのサポートについても説明します。特に明記しないかぎり、このホワイトペーパーの情報はSQL Anywhere Studio 8.0.2に該当します。

Adaptive Server Anywhereにおけるユニコードのサポート

Adaptive Server Anywhere におけるのユニコードのサポートには、次の機能が含まれています。

- **UTF8 照合** UTF8 照合を使用して、Adaptive Server Anywhere データベースを作成できます。この照合での文字ソートには制限があります。
- **SORTKEY 関数** SORTKEY SQL 関数を使用して、文字データの組み込みソートを増強できます。
- **COMPARE 関数** COMPARE SQL 関数を使用して、文字データの組み込みソートを増強できます。
- 文字セット変換 データベース・サーバは、クライアントとデータを交換するときに、互換性のある文字セット間の変換を行います。

UTF8 照合

Adaptive Server Anywhere の UTF8 照合を使用すると、4 バイト以内の UTF8 文字であればどれでも格納できます。7 ビット U.S. ASCII 文字は 1 バイトとして格納されるため、英語の文字だけを格納する場合に UTF8 を使用してもオーバーヘッドはありません。アジア言語の文字はほとんどが 3 バイトとして格納され、アクセント記号が付いたラテン語ベースの文字など、その他の文字はほとんどが 2 バイトとして格納されます。

Adaptive Server Anywhere では、UTF8 照合のソートは制限があり、ソート順はマルチバイト文字の最初のバイトにのみ設定できますが、その順序は人には意味がないものです。例えば、ヨーロッパ言語のアクセント記号付きの文字は通常、2 バイトとして格納されますが、アクセント記号付き文字には、最初のバイトが同じものが多くあります。しかし、Adaptive Server Anywhere のソートでは最初のバイトだけを使用するため、同じ基本アルファベットでアクセント記号が付いているものと付いていないものが適切にグループ分けできません。典型的な結果では、アクセント記号のないさまざまな文字が、基本アルファベットとは無関係に、かけ離れた順序でグループ分けします。

SORTKEY 関数

Adaptive Server Anywhere では、SybaseのAdaptive Server Enterprise でも提供されているSORTKEY 関数が提供されています。SORTKEY 関数は、Sybase Unilib ライブラリに基づいて実装されていて、 SQL Anywhere Studioでは8.0 より導入されましたが、SQL Anywhere Studio 7 および 6 で、ストアド・プロシージャ、外部関数、DLL のコレクションとして使用することもできます。

SORTKEY 関数は、基本アルファベット、大文字小文字を区別する値、アクセントの値の順序で構成される、文字列値のマルチパート・バイナリ・ソート・キーを生成できます。この順序は ISO 14651 に基づいていて、比較は次のように行われます。

- アルファベットは、基本アルファベット、大文字小文字の区別、アクセントの値の 3 つの要素に分けられます。合字 (ドイツ語の硬い s を示す "β" など) は、2 文字 (この場合は "ss") に変換され、以下に示す手順では、基本アルファベットを基本文字として使用します。

- アルファベット以外の文字の場合は、その文字自体を基本文字として使用します。

- それぞれの値は、まず基本文字のみを使用して比較され、基本文字が等しくない場合、最後まで比較します。

- 基本アルファベットとその他すべての文字が等しい場合、次に大文字小文字の区別を調べます。大文字小文字の違いがある場合、最後まで比較します。小文字は大文字よりも順序が「下」になります。

- 基本アルファベットとその他すべての文字が等しく、かつ大文字小文字も全く同じである場合、文字のアクセントの値を調べにかかります。文字のアクセントの値が異なれば、元の値も異なることになります。アクセントの値が同じであれば、元の値は等しいことになります。また、アクセント記号のない文字はアクセント記号付きの文字よりも順序が「下」になります。アクセント記号付きの文字同士の比較については、このホワイトペーパーでは説明していません。

ソート・キーの使用

ソート・キーには 2 つの標準的な使用方法があります。動的に作成するか、データベースに格納するかです。

多くの場合、ソート・キーをデータベースに格納することをお勧めします。シャドー・カラムを作成し、COMPUTE 句を使用してシャドー・カラムにソートするカラムのソート・キー値を含める簡単な方法です。例を以下に示します。

```
CREATE TABLE T (
     Name       CHAR(100),
     Sh_Name    VARBINARY(500) COMPUTE(SORTKEY(Name))
);

INSERT INTO T (Name) VALUES (‘François Allaire’);
```

INSERT 文は、Name カラムのソート・キーを自動的に計算します。

```
SELECT Name
FROM T
ORDER BY Sh_Name;

SELECT Name
FROM T
ORDER BY SORTKEY(Name);
```

この 2 つの SELECT 文は同じ結果になります。どちらの場合も、ORDER BY 句は Sh_Name シャドー・カラムを使用してソートします。

データベースのサイズを小さくする必要がある場合や、ORDER BY 操作の回数が少なくサイズが小さい場合には、ソート・キーを動的に作成するという別の方法が便利です。

次のクエリは、Name カラム用に動的に作成したソート・キーを使用して結果をソートします。

```
SELECT Name
FROM T
ORDER BY SORTKEY(Name)
```

クエリを実行すると、ソート・キーの生成、値のソート、ソート・キーの廃棄が行われます。データベース・サイズには、このソート・キーの使用方法による影響はありませんが、文を実行するたびにソート・キーを作成したり廃棄する必要があるため、パフォーマンスに悪影響が出る場合があります。

SQL Anywhere 8.0.1 では、SORT_COLLATION データベース・オプションを使用して、任意の「文字」タイプのカラムや式で実行するすべての ORDER BY 操作に対して、SORTKEY 関数を有効にすることができます。また、データベース・オプション SORT_COLLATION は、すべてのユーザ、特定のユーザ、現在の接続のいずれかに適用できます。例えば、次の文を実行すると、この接続が続いている間、文字タイプに関するすべての ORDER BY 操作でデフォルトの SORTKEY 順序が適用されます。

```
SET TEMPORARY OPTION SORT_COLLATION='default';
```

例えば、

```
SELECT Name
FROM T
ORDER BY Name;
```

は、以下と等しくなります。

```
SELECT Name
FROM T
ORDER BY SORTKEY(Name);
```

または

```
SELECT Name
FROM T
ORDER BY SORTKEY(Name, 'default');
```

SORTKEY のデフォルトの順序はマルチリンガルな順序なため、ほとんどのユーザにはこれで十分です。SORTKEY 順序の完全リストは、SQL Anywhere Studioのマニュアルを参照してください。

COMPARE 関数

COMPARE 関数は、SORTKEY 関数と密接に関連しています。COMPARE 関数は 2 つの文字パラメータをとり、それぞれのパラメータ用のソート・キーを生成して、ソート・キーを比較します。値は、ORDER BY と同じルールを使用して、SORTKEY 関数に基づいて比較します。値が「等しい」場合は 0、「小さい」場合は -1、「大きい」場合

は 1 を返します。

例

```
SELECT Name
FROM T
WHERE COMPARE( Name, 'Straße', 'noaccent' ) = 0;
```

で、Name が 'Strasse' である場合、COMPARE 関数は 0 (等しい) を返します。

これに対して、

```
SELECT Name
FROM T
WHERE Name = 'Straße';
```

は 1 (大きい場合) を返します。これは、等号演算子で使用されているデフォルトの比較機能では合字を処理できないためです。

データベース・サーバの文字セットの変換

アプリケーションでは、多種多様な文字セットの中の 1 つを使用できますが、通常、使用する文字セットはマシンのネイティブの文字セットです。多くの場合、*式コード・ページ*と*文字セット*は互いに交換して使用できます。英語版の Windows アプリケーションで通常使用する文字セットは cp1252 であり、日本語版の Windows アプリケーションで通常使用する文字セットは cp932 です。

Adaptive Server Anywhere クライアントのインタフェースによっては、UCS-2 を使用するアプリケーションもあります (次項を参照)。この項では、シングルバイトまたはマルチバイトの文字セットを使用するアプリケーションについて説明します。

データベースの文字セットは、データベース照合によって特定されます。ほとんどの照合名は、コード・ページ番号と順序付けの方法から成ります。例えば、1252LATIN1 は、ラテン語-1 の順序付けを使用してソートされたコード・ページ 1252 であり、これは英語を含む西ヨーロッパ言語に適しています。932JPN は、日本語の順序付けを使用してソートされたコード・ページ 932 (日本語文字セット) です。

データベース・サーバで、クライアントとは異なる文字セットを使用するデータベースが実行される場合があります。例えば、データベースの文字セットが UTF8 である場合などです。この場合、クライアントとサーバの間で*文字セットの変換*を行う必要があります。この変換は、クライアントのアプリケーション内、クライアントのライブラリ内、データベース・サーバ内のいずれかで実行できます。

Adaptive Server Anywhere データベース・サーバは、アプリケーションの文字セットとデータベースの文字セットを自動的に変換します。文字セットの変換は -ct コマンド・ライン・オプションで制御しますが、SQL Anywhere Studio 8 では、変換はデフォルトで on になっています。SQL Anywhere 7 および 6 では、変換はデフォルトでは *off* になっています。

文字セットの変換は、必要な場合にのみ実行されます。例えば、アプリケーションの文字セットとデータベースの

文字セットが同じ場合、データベース・サーバは、そのアプリケーションからの接続に関して文字セットの変換を行いません。

アプリケーションでは、"charset=" 接続パラメータを使用してアプリケーションの文字セットを指定できます。指定がない場合、アプリケーションの文字セットは Windows の文字セットとみなされます。

文字セットの変換が有効な場合、変換はデータベースの文字セットおよびアプリケーションの文字セットの影響を受けます。文字セットの変換が無効な場合、サーバはアプリケーションまたはクライアント・ライブラリに依存してデータベースの文字セットで通信します。

クライアント・ライブラリにおけるユニコードのサポート

ODBC クライアント

Windows では、ODBC クライアント・ライブラリはユニコードが有効になっていて、「ワイド文字」(UCS-2) を SQL_WCHAR タイプでサポートしています。

そのため、ODBC ドライバ・マネージャが ODBC ドライバのユニコード API を使用することを理解しておく必要があります。これによりクライアント側で文字セットの変換が行われることになります。

アプリケーションは、UNICODE マクロを定義した状態でコンパイルできます。この場合、ODBC 関数に対して指定される文字列はすべて UCS-2 文字列になります。ODBC ドライバ・マネージャのユニコード API を使用して、UCS-2 文字列を ODBC ドライバとの間で修正なしでやり取りします。

アプリケーションを UNICODE マクロが定義された状態でコンパイルしない場合、ODBC ドライバ・マネージャの ANSI API を使用します。ドライバ・マネージャが、ODBC ドライバのユニコード API を呼び出すときに、すべての文字列をアプリケーションの文字セットから UCS-2 に変換し、返された文字列をUCS-2 からアプリケーションの文字セットに変換します。この場合、ODBC ドライバ・マネージャは、アプリケーションの文字セットを Windows の文字セットと*みなします*。

バウンド変数は、ドライバ・マネージャの文字セット変換の影響を*受けません*。SQLタイプが SQL_CHAR のバウンド変数は、Windows の文字セットに含まれる (ODBC ドライバ・マネージャの動作に一致する) と*みなされます*。

ODBC ドライバは、データベース・サーバとネゴシエートして、文字セット変換の処理方法を特定します。ODBC ドライバでは、ユニコード API が使用されているため、すべての UCS-2 文字列をデータベースで必要な文字セットに変換 (またはその逆に変換) し（変換される場所に関しての詳細については、このホワイトペーパーでは説明していません）、データベース・サーバとネゴシエートして、必要な文字セット変換の数を最小限にします。文字セットの変換は、ネゴシエーションの結果によって、ODBC ドライバまたはデータベース・ドライバのどちらかで行われます。

OLE DB クライアント

OLE DB ドライバは UCS-2 の文字をサポートします。ODBC ドライバと同様に、OLE DB ドライバとデータベース・サーバは文字セット変換についてネゴシエートします。

Embedded SQL クライアント

Embedded SQL プリプロセッサおよびクライアント・ライブラリは、UCS-2 の文字セットをサポートしません。アプリケーションで UCS-2 の文字を使用する場合は、必要なデータベースの文字セット間の変換は、アプリケーション側で行います。

Windows コード・ページと UCS-2 の間の変換は、オペレーティング・システムから、WideCharToMultiByte 関数と MultiByteToWideChar 関数を使用して実行できます。

UCS-2 と UTF8 の間の変換は単純にビットをシフトするだけなので効率的で、テーブルは必要ありません。また、Windows デスクトップ・システムでは、WideCharToMultiByte 関数と MultiByteToWideChar 関数も使用できます。

Java クライアント

Java では UCS-2 の文字セットを使用します。Java アプリケーションは Adaptive Server Anywhere データベースと通信する際JDBC を使用しますが、その通信方法には、次の 2 つがあります。

- ｊConnect は、SQL Anywhere Studioに同梱されているピュア Java のJDBC の実装で、TDS 通信プロトコルを使用します、また必要な文字セット変換を行います。
- JDBC-ODBC ブリッジが使用できます。このブリッジは、JDBC とネイティブの ODBC データベース・ドライバの間のインタフェースを提供する Java コードのレイヤです。バージョン 8.0.2 以降、iAnywhere Solutions では独自の JDBC-ODBC ブリッジを SQL Anywhere Studio に実装しています。このブリッジを使用すると、文字セット変換は ODBC に関する項で説明したとおりに処理されます。

Mobile Link におけるユニコードのサポート

Mobile Link サーバはユニコード (UCS-2) アプリケーションです。

Windows CE データベースの場合、アップロードとダウンロードには UCS-2 の文字セットを使用しますが、その他のリモート・プラットフォームの場合は、アップロードとダウンロードはリモート・データベースの文字セットで行います。

Mobile Link では、統合データベースとの通信はODBC のユニコード API を使用して行います。Mobile Link リモート・クライアントから受信したデータは、UCS-2 に変換してから統合データベースに送信され、クライアントに送信するデータは、UCS-2 からリモート・クライアントの文字セットに変換してからクライアントに送信されます。Windows CE クライアントの場合、変換は不要です。

Ultra Light におけるユニコードのサポート

Java では UCS-2 の文字セットを使用します。文字セットの変換は不要です。

C/C++ アプリケーションを、UCS-2 の文字セットを使用して作成できます。この場合、Windows でコンパイルするときに UNICODE マクロを定義してください。UNICODE マクロを定義すると、UL_CHAR タイプは 16 ビット文字に変更されます。また、Ultra Light におけるデータ格納は、UTF8 の文字セットをサポートしています。

結論

ユニコードに対するサポートは、「イエス」か「ノー」かというものではありません。このホワイトペーパーでは、SQL Anywhere Studio でのユニコードのサポートについて、特に Adaptive Server Anywhereに重点を置いてまとめました。

法的注意

iAnywhere、iAnywhere Solutions、Adaptive Server、SQL Anywhereは、iAnywhere Solutions, Inc.またはSybase, Inc.とその系列会社の商標です。その他の商標はすべて各社に帰属します。

Mobile Linkの技術には、Certicom,Inc.より供給を受けたコンポーネントが含まれています。これらのコンポーネントは特許によって保護されています。

本書に記載された情報、助言、推奨、ソフトウェア、文書、データ、サービス、ロゴ、商標、図版、テキスト、写真、およびその他の資料(これらすべてを"資料"と総称する)は、iAnywhere Solutions,Inc.とその供給元に帰属し、著作権や商標の法律および国際条約によって保護されています。また、これらの資料はいずれも、iAnywhere Solutions,Inc.とその供給元の知的所有権の対象となるものであり、iAnywhere Solutions,Inc.とその供給元がこれらの権利のすべてを保有するものとします。

資料のいかなる部分も、iAnywhere Solutionsの知的所有権のライセンスを付与したり、既存のライセンス契約に修正を加えることを認めるものではないものとします。

資料は無保証で提供されるものであり、いかなる保証も行われません。iAnywhere Solutions,Inc.は、資料に関するすべての陳述と保証を明示的に拒否します。これには、商業性、特定の目的への整合性、非侵害性の黙示的な保証を無制限に含みます。

iAnywhere Solutionsは、資料自体の、または資料が依拠していると思われる内容、結果、正確性、適時性、完全性に関して、いかなる理由であろうと保証や陳述を行いません。iAnywhere Solutionsは、資料が途切れていないこと、誤りがないこと、いかなる欠陥も修正されていることに関して保証や陳述を行いません。ここでは、「iAnywhere Solutions」とは、iAnywhere Solutions, Inc.またはSybase,Inc.とその部門、子会社、継承者、および親会社と、その従業員、パートナー、社長、代理人、および代表者と、さらに資料を提供した第三者の情報元や提供者を表します。

＊本書は、米国iAnywhere Solutions, Inc.が作成・テストしたものを日本語に翻訳したものです。

アイエニウェア・ソリューションズ株式会社
http://www.ianywhere.jp

1021787