

IBM Informix Dynamic Server 11 prep 用システム管理認証 Exam 918

パート 1 : IDS のインストールおよび構成

技能レベル : 中程度

Manjula Panthagani ( manjulap@us.ibm.com )

Advanced Support Engineer

IBM

Rashmi Chawak ( chawak@us.ibm.com )

Advanced Support Engineer

IBM

Siva K. Ane ( sane@us.ibm.com )

Advanced Support Engineer

IBM

2007 年 5 月 10 日

IBM® Informix® Dynamic Server Version 11 ( IDS 11 ) のインストールと構成を行い、領域とログを管理し、IDS で使用できる各種のセキュリティ・オプションを構成します。このチュートリアルは、8 つのチュートリアルで構成されているシリーズの最初の部分に該当し、IDS 11 Exam 918 の準備を進めるためのものとなります。

第 1 章 作業を開始する前に

このチュートリアルでは、IBM Informix Dynamic Server 11 のインストールと構成について説明します。

本シリーズについて

IBM IDS V11 ( Exam 918 ) のシステム管理に関する認証の獲得について考えたことがあ

りますか？もしあるなら、ユーザは正しい考えを持っています。8 つから構成される IDS 認証準備チュートリアルのうち、このチュートリアルでは、あらゆる基本事項 Exam の最初の質問を読む前に理解しておく必要のあるトピック について説明しています。たとえ今すぐ認証の獲得を計画していなくとも、この一連のチュートリアルは、IDS 11 のどこが新しいのかを学ぶための有効なスタート・ポイントとなります。

このチュートリアルについて

このチュートリアルでは、IBM Informix Dynamic Server 11 のインストールと構成の方法について説明します。

目的

このチュートリアルの学習を終了すると、ユーザは：

- ・ IDS 11 のインストール方法を理解できます。
- ・ 各種構成パラメータ、環境変数、ネットワーク・プロトコル、および sqlhosts ファイルを理解できます。
- ・ IDS 11 に追加されたいくつかの ONCONFIG パラメータと旧バージョンとを比べる方法を理解できます。
- ・ サイレント、GUI などの各種モードでサーバーをインストールし、サーバーをダウンさせる方法を習得できます。
- ・ 各 ONCONFIG パラメータを理解し、UNIX®オペレーティング・システム・プラットフォームをベースとして onconfig ファイル内の関連パラメータを編集できます。

前提条件

このチュートリアルでは、データベースに関する基本的概念を理解しているデータベース管理者 (DBA) を対象としています。

システム要件

このチュートリアルを学習するにあたって、ユーザが IDS のコピーを入手する必要はありません。ただし、このチュートリアルと併せて作業を行うために IBM IDS 11 の無料トライアル版をダウンロードすれば、より効果的に学ぶことができます。

## 第 2 章 IDS のインストール

いくつか用意されている次のインストール方法から好きな方法を選択することができます。

・コンソール・モード（これはデフォルト・モードです。インストール・アプリケーションをこのモードで開始する場合には、コマンド・ラインへ特定のモード・パラメータを追加する必要はありません。）

- ・ GUI モード
- ・ サイレント・モード
- ・ コマンド・ライン・スクリプトを使ってインストールを抽出
- ・ JAR ファイルを直接実行

デフォルトのインストール・アプリケーションを使って Dynamic Server と他の IBM Informix 製品をインストールする

インストールしたい製品に固有の以下のインストール・コマンドを使用します。

- ・ `ids_install` コマンドは、IDS のすべてのコンポーネントをインストールします。
- ・ `installserver` コマンドは、データベース・サーバーのみインストールします。
- ・ `installconn` コマンドは、Informix Connect のみインストールします。
- ・ `installclientsdk` コマンドは、Client SDK のみインストールします。

表 1 インストール・オプション

オプション	説明
<code>-gui</code>	インストール・プログラムを GUI モードで開始します。
<code>-console</code>	コンソール（デフォルト）モードでのインストール
<code>-silent</code>	サイレント・モードでのインストール
<code>-record</code>	他のインスタンスにおいてサイレント・インストール用としてカスタム <code>.ini</code> ファイルを持つために、応答ファイル、ユーザの GUI を記録するファイル、またはコンソール・インストールの設定値を作成します。
<code>-acceptlicense = yes</code>	サイレント・モードにおけるライセンス契約を受け入れます。

-javahome dir	インストール・プログラムを特定の JRE を使用して強制的に実行します。バンドル JRE を使用する場合には、-javahome none コマンドを使用します。
-tempdir	指定されたテンポラリ・ディレクトリを使用します。
-log file name	インストール・プログラムの進捗状況を記録します。
-is:freediskblocks	インストールの前に、適切な製品インストール・ファイル用の領域が存在するかどうかを確認します。
-is:nospacecheck	適切な製品インストール・ファイル用の領域が存在するかどうかのチェックをインストール・プログラムに行わせないようにします。テンポラリ・ファイルの抽出に十分な領域が存在しない場合には、インストール・プログラムは停止します。

新しいディプロイメント機能によって、ユーザは、インストールの応答を簡単な ASCII ファイル（これは、将来のすべてのサイレント・インストールに対して使用可能です）へ記録することができます。後に同じ構成をサイレント・インストールまたは他のインスタンスに適用できるよう、GUI モードまたはコンソール・モードでの Dynamic Server と応答ファイル内のバンドル Informix 製品のインストールで行った設定を獲得することができます。他のディレクトリでインストールのセットアップを繰り返したい場合に、このプロセスを使用することで時間の節約ができます。このタスクの終了は、まずコマンド・ライン・プロンプトを開き、次に GUI インストール・アプリケーションまたはコンソール・インストール・アプリケーションを開いて行います。選択したインストール・ウィザードが正常に終了するまで、コマンド・ライン・セッションは開いたままにしておきます。

応答ファイルを作成し、サイレント・インストールに使用する

1. Dynamic Server のみをインストールする場合には、引数 `./installserver [-gui] -record responsefile.ini` を入力します。他の Informix 製品がバンドルされた Dynamic Server をインストールする場合には、`./ids_install [-gui] -record responsefile.ini` を入力します。応答ファイルの名前に `server.ini` および `bundle.ini` は使用できません。
2. GUI モードまたはコンソール・モードでインストールを実行します。

3. 応答ファイルが正常に作成されたことを示すメッセージをユーザのシステムが生成していることを確認します。ルート・ユーザとして、次の例のようにサイレント・インストールを実行します。ここで、responsefile.ini はファイル名またはファイル名とパスを表します。

```
./ids_install -silent -acceptlicense=yes -options responsefile.ini
```

### 第3章 IDSの構成

データベース・サーバーの構成には、以下のステップが必要となります。

- ・ データ格納用の領域を用意する。
- ・ 対応の環境変数の設定を行う。
- ・ sqlhosts ファイルの設定を行う。
- ・ \$INFORMIXDIR/etc ディレクトリの中にある構成ファイルを使って、システムを構成する。

#### データ格納用の領域を用意する

IDS サーバーは、入出力に以下の2種類の方法を使用します。

カーネル AIO：カーネル非対称入出力は、オペレーティング・システムを通してディスクの非ブロッキング読み込みと非ブロッキング書き込みを実行する方法です。これは、ディスクへのデータの書き込み中またはディスクからのデータの読み込み中はそれが終了するまで待たされるというプロセスが必要となる従来の読み込み / 書き込み操作に代わる方法です。ここでは、サーバーは読み込み / 書き込み要求を発行し、その後の処理を続けることができます。入出力が終了すると、サーバーに通知が出されます。カーネル AIO は、特定のオペレーティング・システムとハードウェア・プラットフォームでのみ使用することができます。ロウデバイス（定義は下記）にチャックが存在している場合にも、カーネル AIO は実行されます。カーネル AIO スレッドは、CPU VP 上で実行されます。

サーバーがカーネル AIO をサポートしているかどうかを調べるには、  
\$INFORMIXDIR/release ディレクト内のリリース・ノートをチェックします。

AIO VP を介した入出力：サーバーは、AIO VP を通して入出力を実行することができます。AIO VP は、カーネル AIO が実行されなかった場合に読み込み / 書き込み操作の実行を担います。AIO VP は、すべてのクックド・ファイル上でも入出力を実行します。

ロウデバイスとは、デバイスのパス名とデバイス・ドライバとを関連付ける UNIX ユーティリティを使ってユーザが作成するキャラクタ・スペシャル・デバイスです。このドライバはオペレーティング・システムの一部で、入出力要求をディスク・ハードウェアに必要な命令に翻訳します。このドライバは、UNIX ファイル・システムの影響を受けません。

クックド・ファイルとは、オペレーティング・システムによって管理される通常のファイルです。クックド・ファイルは、データベース・サーバーがファイルの内容を制御していますが、オペレーティング・システムに対して入出力要求を行わなければなりません。

#### ロウデバイスの作成

ディスク上に新しい（またはフリーであると識別できる）パーティションを作成し、以下のコマンドを発行します。

```
chmod 660 device_name  
chgrp informix device_name  
chown informix device_name
```

特に、ユーザのシステムがカーネル AIO を利用している場合には、ユーザのチャックとして UNIX ファイルを使用することはお勧めできません。ただし、クックド・ファイルは設定が簡単で、ディスク・デバイスの可用性を調べる必要がありません。チャックに UNIX（クックド）ファイルを使いたい場合には、そのチャックに使用するファイルの設定を行わなければなりません。

## クックド・ファイルの作成

以下のコマンドを発行します。

```
touch filename
chmod 660 filename
chgrp informix filename
chown informix filename
```

## 環境の設定

サーバーを初期化する前に、リストアップされた変数がユーザの環境に入っていることを確認します。現在の設定環境変数をチェックするには、UNIX 内で env コマンドを使用します。

サーバーの初期化の前に、以下の環境変数を設定します。

表 2 環境変数

変数	説明
INFORMIXDIR	IBM Informix 製品がインストールされているディレクトリに設定します (例: /usr/informix)。
PATH	\$INFORMIXDIR/bin を包含していなければなりません。
INFORMIXSERVER	DBSERVERDBNAME または DBSERVERALIASES 構成パラメータのいずれかの値へ設定します。

リスト 1 に、これらの環境変数を設定する設定スクリプトの例を紹介します。

## リスト 1 環境変数を有するサンプル・ファイル (C SHELL)

```
source ~/.env.11.10
setenv INFORMIXDIR /usr3/11.10/sqlldist
setenv TERMCAP ${INFORMIXDIR}/etc/termcap
setenv SHELL /bin/sh
setenv TERM vt100
```

```
setenv INFORMIXSQLHOSTS /$INFORMIXDIR/etc/sqlhosts
setenv PATH .:$INFORMIXDIR/bin
setenv INFORMIXSERVER menlo
setenv ONCONFIG onconfig.11.10
```

## 第 4 章 ディスク領域の管理

### 物理単位

データベース・サーバーは、以下の物理単位を使ってディスク領域を管理します。

- ・チャンク
- ・ページ
- ・エクステント
- ・BLOB ページ
- ・SB ページ

### チャンク

チャンクとは、サーバーに割り当てられたディスク領域または物理領域の単位です。チャンクは、ロウデバイス（キャラクタ・スペシャル・デバイス）または UNIX クックド・ファイルとすることができます。チャンクをサーバーに割り当てる場合には、以下の 3 つの値を指定する必要があります。

1. パス名：チャンクに対して使用するファイルのパスまたはロウデバイス名です。
2. オフセット：デバイスの始めからデバイスの読み込みと書き込みを開始する場所までの物理距離を KB で表したものです。クックド・ファイルを使用したチャンクを作成する場合には、オフセットはゼロを使用してください。値がゼロ以外のオフセットは、ロウデバイスに対してのみ使用します。
3. サイズ：チャンクに対して使用する UNIX ファイルのオフセットまたはサイズからの領域で、KB を使って指定し、ロウデバイスの中で使うために割り当てられた値です。

IDS サーバーには、チャック数が 32,767 という論理リミットがあります。ただし、UNIX カーネルでは、ファイル数にリミットが課せられる場合があり、この場合プロセスは、IDS サーバーのリミットに代わり、UNIX カーネルによるファイル数のリミットが課

せられます。

### ページ

サーバーが使用する入出力の基本単位となるのがページです。IDS の Version 10 以降、ページ・サイズは、製品プラットフォームをベースとした固定値ではなくなっています。ルート DB 領域が作成された後で追加される DB 領域やそのチャンクには、デフォルトのページサイズ(2K または 4K)から 16K まで、かつデフォルトのページ・サイズの倍数のページ・サイズ使うことができます。指定サイズのページを保持するためのバッファプールが存在していない場合には、新しくバッファプールが作られます。

### エクステント

ディスク上において物理的に連続したページの集合体。表に使用する領域は、エクステントを単位として割り当てられます。表のエクステント・サイズは、表の作成時に指定します。

### BLOB ページ

BLOB ページとは、BLOB 領域に格納されている BLOB データの基本格納単位です。BLOB ページのサイズは、システム・ページ・サイズの倍数で構成することができます。

### SB ページ

SB ページは、SB 領域内にスマート・ラージ・オブジェクトを格納するためにデータベース・サーバーが使用するページのタイプです。BLOB ページとは異なり、SB ページは構成することができません。SB ページのサイズは、データベース・サーバー・ページと同じです。

### 論理単位

データベース・サーバーは、以下の論理単位でデータを格納します。

- ・ DB 領域
- ・ BLOB 領域
- ・ SB 領域

- ・ 表領域

### DB 領域

DB 領域は、データベースと表の格納に使われる 1 つまたは複数のチャンクの論理集合体です。各 DB 領域は、そこに割り当てられたチャンクを少なくとも 1 つ有していなければなりません。

### BLOB 領域

BLOB 領域は、テキスト・データのみまたはバイト・データのみを格納する 1 つまたは複数のチャンクから構成される論理格納単位です。データベース・サーバーは、BLOB 領域に格納されているデータをディスクへ直接書き込みます。このデータは、常駐共有メモリを通りません。

### SB 領域

SB 領域は、スマート・ラージ・オブジェクトを格納する 1 つまたは複数のチャンクから構成される論理格納単位です。スマート・ラージ・オブジェクトには、BLOB、CLOB、およびユーザ定義のデータ・タイプ (UDT) があります。

### 表領域

表領域は、特定の表またはインデックスに関する情報を単一 DB 領域へ格納するために割り当てられたすべてのエクステントの集合体です。表領域を使って表された領域は必ずしも連続しているわけではありませんが、エクステントの 1 つを使って表された領域は必ず連続となります。

### リスト 2 onspaces を使って DB 領域を作成する

```
onspaces
-c
-d <dbspace>
-k <pagesize>
-m <mpathname moffset>
-o <offset>
-p <pathname>
-s <size>
-t <tempSPACE>
```

```

where
spacename is the name of the dbspace to be created.
pagesize is non-default page size for the new dbspace.
mpathname is mirror pathname
moffset is mirror offset
offset is offset into the device in KB
pathame is Path to the initial chunk
size is Size of initial chunk in KB
-t indicates if the dbspace created is temporary.
Example : To create a 1-million KB mirrored dbspace dbspacel
with an offset
of 200,000KB for initial(Primary) chunk and an offset of
450,000 for the
mirrored chunk.
onspaces -c -d dbspacel -p /dev/rdisk/device1 -o 200000 -s
1000000 -m
/dev/rdisk/device2 450000

```

### リスト 3 BLOB 領域を作成する

```

onspaces
-c
-b <spacename>
-g <blobpagesize>
-m <mpathname moffset>
-o <offset>
-p <pathname>
-s <size>
where spacename is the name of the blobspace to be created.
blobpagesize is Blobpage size in number of disk pages.
mpathname is mirror pathname.
moffset is mirror offset.
offset is Offset into the device in KB
pathname is Path to the initial chunk
size is Size of initial chunk in KB
Example : To create a 1-million KB mirrored blobspace blobsp1
with an offset of 200,000KB for both initial(Primary) and
mirrored chunk and a blobpage size of 100KB ( 2K system page
size ) .
onspaces -c -b blobsp1 -g 50 -p /dev/rdisk/device8 -o 200000 -s
1000000 -m
/dev/rdisk/device9 200000

```

### リスト 4 SB 領域を作成する

```

onspaces
-c
-S <spacename>

```

```

-m <mpathname moffset>
-o <offset>
-p <pathname>
-s <size>
-t
-Ms <metasize>
-Mo <metaoffset> :
-Df <options>
where spacename is the name of the blobspace to be created.
blobpagesize is Blobpage size in number of disk pages.
mpathname is mirror pathname.
moffset is mirror offset.
offset is Offset into the device in KB
pathname is Path to the initial chunk
size is Size of initial chunk in KB
-t indicates if the dbspace created is temporary.
metasize is Size of the sbpace metadata area in KB.
metaoffset is Offset of the metadata area into the
sbpace in KB
options Lists default specifications for smart large
objects stored
in the sbpace.
Example : To create a 1-million KB mirrored sbpace sbpace1
with an
offset of 200,000KB for both initial(primary) and mirror
chunks, a metadata
size of 7500 KB with 10000 KB offset and expected average smart
blobsize of 32KB.
onspaces -c -S sbpace1 -p /dev/rdisk/device5 -o 200000 -s
1000000 -m
/dev/rdisk/device6 200000 -Ms 7500
-Mo 10000 -Df "AVG_LO_SIZE=32"

```

## 領域の破棄

onspaces コーティリティを使って、DB 領域、BLOB 領域、または SB 領域をシステム・コマンド・ラインから破棄することができます。DB 領域を破棄する場合には、その前に、当該 DB 領域内で作成されたデータベースと表をすべて破棄しておかなければなりません。

BLOB 領域を破棄する場合には、その前に、当該 BLOB 領域を参照しているテキスト / バイト欄を有する表をすべて破棄しておかなければなりません。

## リスト 5 onspaces を使って領域を破棄する

```

Example : To drop the dbspace dbpace1.
onspaces -d dbpace1

```

### DB 領域または BLOB 領域へチャンクを追加する

onspaces コーティリティを使って、DB 領域または BLOB 領域へチャンクを追加することができます。リスト 6 に、onspaces コーティリティを使って DB 領域へチャンクを追加する際に使用する引数を示します。

#### リスト 6 DB 領域または BLOB 領域へチャンクを追加する

```
onspaces
-a <spacename>
-m <mpathname moffset>
-o <offset>
-p <pathname>
-s <size>
where
spacename is the name of dbspace or a blobspace to which
chunk needs
to be added.
mpathname is mirror pathname
moffset is mirror offset
offset is offset into the device in KB
Path to the device or filename of the chunk
Size of the chunk in KB
Example : To add a 500,000 KB mirrored chunk to dbspace2 with a
100,000 KB offset.
onspaces -a dbspace2 -p /dev/rdsk/device4 -o 100000 -s 500000
```

### チャンクを SB 領域へ追加する

onspaces コーティリティを使って、SB 領域へチャンクを追加することができます。リスト 7 に、onspaces コーティリティを使って SB 領域へチャンクを追加する際に使用する引数を示します。

#### リスト 7 SB 領域へチャンクを追加する

```
onspaces
-a <spacename>
-m <mpathname moffset>
-o <offset>
-p <pathname>
```

```

-s <size>
-Ms <metasize>
-Mo <metaoffset>
-U
where
spacename is the name of sbspace to which chunk needs to
be added.
mpathname is mirror pathname
moffset is mirror offset
offset is offset into the device in KB
Path to the device or filename of the chunk
Size of the chunk in KB
metasize is Size of the sbspace metadata area to be
allocated in the
new chunk in KB .
metaoffset is the Offset of the metadata area into the
new chunk in KB
-U Indicates that the new chunk is to contain only user
data
Example : To add a 100,000KB mirrored chunk to the sbspace
named sbpace1 with an
offset of 20,000 KB for both initial and mirrored chunks and
metadata size of 750KB and 1000 KB offset .
onspaces -a sbpace1 -p /dev/rdisk/chunk6 -o 20000 -s 100000 -m
/dev/rdisk/chunk7 -Ms 750 -Mo 1000

```

## SQL 管理 API

Dynamic Server の各種の管理コマンド・ライン・ユーティリティをエミュレートする SQL の EXECUTE FUNCTION 文を通して遠隔操作で管理タスクを実行するために、2 つの新しいビルトイン SQL 管理 API 関数を追加しています。

ADMIN と TASK の両関数は、Dynamic Server の管理コマンド・ライン・ユーティリティに対して SQL インタフェースを提供します。これらのビルトイン関数は、各 Dynamic Server インスタンスの sysadmin データベースの中でのみ定義され、ユーザ Informix のみが呼び出せます。

ユーザ Informix として各 Dynamic Server インスタンスの sysadmin データベースへ接続してある場合には、これらの関数を遠隔操作で実行することができます。

ADMIN 関数または TASK 関数の呼び出しを行うたびに、以下の 2 つの結果が作成されます。

1. 指定されたコマンド・タスクを実行して、通常は、ある管理ユーティリティと同等の作業を行う。
2. sysadmin データベースの表 command\_history へ新しい列を挿入する。

TASK 関数は、ステータスを記述した文字列を返します。ADMIN 関数は、表 command\_history へのリンクとなる整数としてステータスを返します。

#### リスト 8. task()関数を使って DB 領域を作成する

```
EXECUTE FUNCTION task('create dbspace','dbs1',
'/dev/rdisk/device1 ', '200000',
'1000000');
Output : (expression) Space 'dbs1' added.
```

#### リスト 9. task()関数を使って DB 領域を破棄する

```
EXECUTE FUNCTION task ('drop dbspace', 'dbs1');
Output : (expression) Space 'dbs1' dropped.
** WARNING ** A level 0 archive will need to be done before
any chunks from
DBspace dbs1 can be reused (see Dynamic Server Administrator's
manual).
```

#### リスト 10. task()関数を使って上記の DB 領域へミラーリングを追加する

```
EXECUTE FUNCTION task ( 'add mirror', 'dbs1',
'/dev/rdisk/device1','200000',
'/dev/rdisk/device2', '450000');
Output: (expression) Mirror chunk '/dev/rdisk/device2' added to
space 'dbs1'
```

#### リスト 11. task()関数を使って BLOB 領域を作成する

```
EXECUTE FUNCTION task('create blobspace ','blobsp1',
'/dev/rdsk/device8 ', '50',
'200000', '1000000');
Output: (expression) Space 'blobsp1' added.
```

## リスト 12. task()関数を使って SB 領域を作成する

```
EXECUTE FUNCTION task('create blobspace ','blobsp1',
'/dev/rdsk/device8 ', '50',
'200000', '1000000');
Output: (expression) Space 'blobsp1' added.
```

## 第 5 章 コネクティビティの構成

### 接続タイプおよび通信プロトコル

オンライン・システムに対してクライアント・サーバー・コネクティビティを構成するには、以下の 3 つの方法があります。

1. 共有メモリ接続を構成する。クライアント・アプリケーションとデータベース・サーバーが同じホスト・コンピュータ上に存在している場合には、これが望ましい通信方法となります。クライアント・アプリケーションとサーバーは、共有メモリの同じセグメントに連結します。
2. ソケットまたは TLI プログラミング・インタフェースを使用し、TCP/IP を通して構成する。TCP/IP は、ローカル通信およびリモート通信の両方に使用することができます。
3. ストリーム・パイプ接続を通して構成する。これは、UNIX ストリームを使用するローカルのプロセス間通信法です。

### sqlhosts ファイル

アプリケーションがデータベース・サーバーとの接続を試みる場合、接続を行うためには何らかの基本的な情報が必要となります。この情報は、\$INFORMIXDIR/etc ディレクトリ内に常駐していなければならないファイルである \$INFORMIXDIR/etc/sqlhosts に記述

しておく必要があります。sqlhosts ファイルの場所の変更には、INFORMIXSQLHOSTS 環境変数を使用します。データベース・サーバーまたはクライアントのホストとなる各コンピュータは、sqlhosts ファイルを有していなければなりません。

sqlhosts ファイルの各エントリ（各行）には、1 台のデータベース・サーバーに対応した sqlhosts 情報を記述します。フィールドを区切る場合は、ホワイト・スペース（スペース、タブ、またはその両方）を使用します。フィールドには、スペースまたはタブを一切入れてはなりません。sqlhosts ファイルの中へコメントを入れたい場合には、コメント・キャラクター（#）を頭に付けて行を開始させてください。読みやすくするために、行を完全にブランクのまま残しておくこともできます。各フィールドに対応する追加の構文規則は、sqlhosts ファイル内のエントリについて説明している次のセクションに掲載されています。sqlhosts ファイルへ情報を入力する場合には、標準のテキスト・エディタを使用します。

リスト 13 に、sqlhosts ファイルのサンプルを示します。

### リスト 13. sqlhosts ファイル（サンプル）

```
dbservername nettype hostname servicename
menlo onipcshm india menlo
Note: This is not mandatory since it is SHM connection.
lenexa ontlitcp california cupertino
asia.1 onsoctcp node6 svc8
```

dbservername は、INFORMIXSERVER 環境変数ならびに ONCONFIG ファイル内の DBSERVERNAME または DBSERVERALIASES に対応します。

nettype 欄には、データベース・サーバーのタイプと接続方法に関する重要情報が入っています。この nettype 欄は、3 つのカテゴリに分類された 8 つの文字から構成されます。

最初の 2 文字は、データベース・サーバー製品を表しています。2 番目の 3 文字は、接続に使われるプログラミング・インタフェースを表しています。最後の 3 文字は、特定のプロトコルまたは IPC メカニズムを表しています。

## リスト 14. nettype 欄

```
d d i i i p p
on - Dynamic Server
se - Standard Engine
ipc - IPC connection
tli - TLI connection
soc - socket connection
shm - Shared memory
str - Stream pipes
tcp - TCP/IP protocol
spx - IPX/SPX protocol
```

hostname は、ローカル・ホスト・マシンの名前です。

固有のサービス番号を持つ必要のあるサービス名はすべて、/etc/services へ入力します。

## リスト 15. /etc/services ファイル (サンプル)

```
menlo 1543/tcp
cupertino 8262/tcp
svc8 8244/tcp
```

## 第 6 章 構成ファイル

構成ファイル：Linux/UNIX

サーバーの構成パラメータは、\$INFORMIXDIR/etc ディレクトリ内のファイルに格納されています。

このファイルの名前の指定は、ONCONFIG 環境変数を設定して行います。フルパスを指定することはできません。ファイル名だけ指定してください。

ONCONFIG 環境変数が定義されていない場合には、デフォルトのファイル名である onconfig が使われます。

例：

```
export ONCONFIG=onconfig.server1
```

構成ファイルには、ユーザのサーバーを特定ニーズに合わせて構成できる各種パラメータが入っています。一部のパラメータは、サーバーを初めて設定したときにセットされ、サーバーの最初の初期化を終えた後では変更することができません。一方、大部分のパラメータは、サーバーの初期化後に修正することができます。

構成ファイルの中にある以下のパラメータ・セクションは、サーバーの初期化の前に構成しなければなりません。これは、ルート DB 領域に予備ページ、サーバー上の全データベースに関する情報、ならびにサーバーの動作を追跡するデータベースが入っていることによるものです。

- ・ルート DB 領域
- ・メッセージ
- ・サーバー情報

#### ルート DB 領域

各サーバーは、ルート DB 領域を有していなければなりません。当初、このルート DB 領域には、物理ログと論理ログも入っています。ただし、これらのログは後で他の DB 領域へ移すことができます。

#### リスト 16. ルート DB 領域を構成する

```
ROOTNAME rootdbs # Root dbspace name
ROOTPATH /dev/online_root # Path for device containing
root dbspace
ROOTOFFSET 0 # Offset of root dbspace into device
(kilobytes)
ROOTSIZE 20000 # Size of root dbspace (kilobytes)
```

これらのパラメータは、サーバーの最初の初期化を行っていない場合のみ修正することができます。サーバーの初期化中にルート DB 領域用として領域の割り当てが終了している場合には、これらのパラメータを変更することはできません。

## メッセージ

### リスト 17. メッセージ・パスを構成する

```
MSGPATH /usr/informix/online.log # System message log file
path
CONSOLE /dev/console # System console message path
```

IDS サーバーは、以下の 2 つのサーバー・メッセージの宛先を提供します。

- ・ MSGPAPH : このパラメータは、すべてのサーバー・メッセージが書き込まれているファイルのパスと名前を示します。それが既に存在している場合を除き、このファイルは、サーバーの最初の初期化が行われたときに作成されます。

- ・ CONSOLE : これは、サーバーがコンソール・メッセージをどこへ書き込むかを指定するパスです。コンソール・メッセージは、サーバーが常駐するコンピュータの管理者にとって重要なメッセージです。たとえば、テープを変更するためのバックアップ要求と復元要求は CONSOLE へ送られます。デフォルトでは、このパラメータはコンピュータのコンソール・デバイスに設定されていますが、ファイルへ設定することもできます。

## サーバー情報

### リスト 18. サーバー固有情報を構成する

```
SERVERNUM Unique id corresponding to an IDS
server
DBSERVERNAME Name of default database server name
DBSERVERALIASES Names of additional database server
names
```

これらのパラメータは、ユーザのサーバーをホスト・コンピュータ上で固有に識別できるように設定しなければなりません。

## ログ情報

### リスト 19. ログ固有情報を構成する

```
LOGBUFF Size in kilobytes for the three logical-log
buffers in shared memory
LOGFILES Number of logical-log files
LOGSIZE Size of logical-log files
PHYSBUFF Amount of shared memory reserved for the buffers
PHYSFILE Size of the initial physical log
PHYSDBS Name of the dbspace in which the physical log
resides
```

## 論理ログ

論理ログファイルは、サーバーに対するトランザクション・レコードを格納するために使われる連続ページの集合体で、ディスク上に存在します。これらのトランザクション・レコードは、ロギングを使って作成されたデータベースに対して行われたすべての変更を追跡するのに使われます。単一 IDS インスタンス上のすべてのデータベースは、同一セットの論理ログファイルを共有しています。各サーバーは、論理ログファイルを少なくとも 3 つ有していなければなりません。

### 論理ログファイルを手動操作で追加する

ユーザは、以下の理由により論理ログファイルを手動操作で追加する場合があります。

- ・ 論理ログに割り当てられたディスク・スペースを増やす。
- ・ 論理ログファイルのサイズを変更する。
- ・ オープン・トランザクションを完了できるようにする。
- ・ 論理ログファイルを別の DB 領域へ移す。

論理ログファイルの追加には、以下の2つの方法があります。

1. `onparams -a` コマンドを使って、ファイル・リストの最後に追加する。
2. `onparams -a -i` コマンドを使って、現在使用中の論理ログファイルの直後に追加する。

以下のコマンドは、LOGSIZE 構成パラメータで指定したログファイル・サイズを使って、ログ領域の DB 領域の中にあるログファイル・リストの最後に論理ログファイルを追加します。

```
onparams -a -d logspace
```

以下のコマンドは、ログ領域の DB 領域の中にある現在のログファイルの後ろに 1000KB の論理ログファイルを追加します。

```
onparams -a -d logspace -s 1000 -i
```

新しいサイズを有する論理ログファイル（このケースでは 250KB）を追加するには、以下のコマンドを実行します。

```
onparams -a -d logspace -s 250
```

以下のコマンドを使えば、論理ログファイルを破棄することができます。

```
onparams -d -l lognum -y
```

以下の操作を行うことで、論理ログファイルを移動させることができます。

- ・現在の DB 領域から論理ログファイルを破棄する。
- ・論理ログファイルを新しい DB 領域へ追加する。

#### 論理ログのサイズと数の推定方法

一般的に、小さなログファイルを多数管理するよりも、大きなログファイルを少数管理する方が容易です。ログ領域が大きすぎることでパフォーマンスには影響ありませんが、ログファイルとログ領域が不足すると、データベース・サーバーが頻繁にチェックポイントのトリガをかけることになるため、パフォーマンスに影響が出る可能性があります。BLOB 領域内のラージ・オブジェクトはログされませんが、オブジェクトが作成されたログ・バックアップに格納されます。つまり、オブジェクトが作成されたログをサーバーがバックアップするまでは、これらのオブジェクトは解放されません。そのため、BLOB 領域内のラージ・オブジェクトが頻繁に更新される場合は、ログ・バックアップを頻繁に行って BLOB 領域内に空き領域を確保する必要があります。生成するログデータ量が少ないアプリケーションでは、各々が 10MB のログファイル 10 個からスタートします。生成するログデータ量が多いアプリケーションでは、各々が 100MB のログファイル 10 個からスタートします。

データ・サーバーの喪失などの壊滅的イベントにおけるデータ損失に対する耐性を決めるリカバリ時間目標 (RTO) ポリシーを維持する方法は 2 つあります。

RTO ポリシーを維持する方法の 1 つは、ログファイルが一杯になるたびにログ・バックアップのトリガをかける自動ログ・バックアップを使用する方法です。この方法では、バックアップ中にログファイルに含まれるトランザクションとバックアップ中に生成する追加トランザクションに対するデータ損失を制限します。

RTO ポリシーを維持するもう 1 つの方法は、スケジューラを使用する方法です。ユーザーは、最後のログ・バックアップの実行後、決められた時間間隔で新しいログ・データを自動的にバックアップするタスクを作ることができます。この方法では、2 つのバックアップ動作の間でバックアップされなかったトランザクションに対するデータ損失を制限しま

す。スケジューラの使い方については、このチュートリアル・シリーズのパート 2「システム動作のモニタリング」で詳しく説明します。

RTO ポリシーが必要ななら、スケジューラを使って、このポリシーを維持するために望ましい頻度で実行されるタスクを挿入することができます。これにより、1 日のサイクル内の特定の時間においてログファイルを自動的にバックアップします。ログのバックアップとリサイクルの前にログ領域が一杯となった場合には、ログのバックアップを行い、新しいファイルを追加して、トランザクションの処理を続けるか、あるいはスケジューラを使って新しいタスクを追加し、この状況を検出していずれかの操作を自動的に行うことができます。

ログファイルはいつでも追加することができます。また、トランザクションの一貫性を維持する上で必要な場合（たとえば、大きなログ領域を消費する可能性のある長いトランザクション）に、データベース・サーバーは自動的にログファイルを追加します。

論理ログ用の領域を増やすもっとも簡単な方法は、他の論理ログファイルを追加することです。

以下の文に、キロバイトで表した総ログ領域の構成例を示します。

```
LOGSIZE = (((connections * maxrows) * rowsize) / 1024) / LOGFILES
```

### 物理ログ

サーバーは、自動復旧用として使用する特殊なログを持っています。このログを物理ログと呼びます。物理ログは、ディスク上に存在する連続ページの集合体です。

ページを共有メモリ・バッファへ読み出し、ユーザが修正を行った場合、このページのオリジナルの状態がコピーとして物理ログへ書き込まれます。ページのこのコピーは、前イメージ（変更される前のページのコピー）として記憶されます。バッファ内にあるページ

に対して最初の変更を行うと、前イメージが物理ログへ書き込まれます。この後に同じページに対して変更を行っても、前イメージがさらに物理ログへ書き込まれることはありません。これらの前イメージは、自動復旧メカニズムによって使用されます。

物理ログの位置とサイズは、onparams を使って移動することができます。

以下のコマンドで、物理ログを DB 領域 dbspace1 へ移し、サイズを 3000KB へ変更します。

```
onparams -p -d dbspace1 -s 3000
```

#### 物理ログのサイズの推定方法

PHYSFILE 構成パラメータの中に指定されている物理ログのサイズは、次の 2 つの要素によって決まります。

1. トランザクションが物理ログ・アクティビティを生成する速度 - ユーザが RTO\_SERVER\_RESTART 構成パラメータを設定しているかどうか
2. 高速復旧のための目標時間を指定するために、ユーザが RTO\_SERVER\_RESTART 構成パラメータを使用しているかどうか

トランザクションが物理ログ・アクティビティを生成する速度は、チェックポイント・パフォーマンスに影響を与えます。トランザクションが物理ログ・データの作成を進めてゆくのに従って、チェックポイント処理中に物理ログが一杯になり始めた場合には、データベース・サーバーは、トランザクションをブロックしてチェックポイントを終了させ、物理ログのオーバフローを回避します。

トランザクション・ブロッキングを回避するためには、チェックポイント処理中に生成するトランザクション・アクティビティのすべてを包含できる十分な物理ログ領域がデータベース・サーバーに存在しなければなりません。物理ログが満杯の 75% に達すると、チェックポイントにトリガをかけます。チェックポイント処理は、物理ログの残りの 25%

を使い尽くすまでに終了しなければなりません。アクティブ・トランザクションの各々に物理ログ・アクティビティを生成する可能性があるため、システムが物理ログ・オーバーフローの可能性を検出すると、ただちにトランザクション・ブロッキングが行われます。

たとえば、ユーザが 1GB の物理ログと 1000 個のアクティブ・トランザクションを有していると仮定します。各トランザクションが同時にクリティカル・セクションへ入った場合に、1000 個のアクティブ・トランザクションは、約 80MB の物理ログ・アクティビティを生成する可能性があります。物理ログの 750MB が一杯になると、データベース・サーバーはチェックポイントにトリガをかけます。920MB の物理ログが使われるまでにチェックポイントが終了していない場合には、チェックポイントが終了するまでトランザクション・ブロッキングが行われます。トランザクション・ブロッキングが行われると、サーバーは、トランザクション・ブロッキングを回避するためにより高い頻度でチェックポイントのトリガを自動的にかけます。ユーザは、自動チェックポイントの生成を無効にすることができます。

物理ログのサイズの推定に使われる 2 番目の要素は、高速復旧のための目標時間を指定するために、ユーザが `RTO_SERVER_RESTART` 構成パラメータを使用しているかどうかによって決まります。高速復旧時間を考えなくてもよい場合には、`RTO_SERVER_RESTART` 構成パラメータを有効にする必要はありません。`RTO_SERVER_RESTART` 構成パラメータに値を指定すると、トランザクション・アクティビティは、追加の物理ログ・アクティビティを生成します。

一般的に、この追加物理ログ・アクティビティはトランザクションのパフォーマンスにほとんどあるいはまったく影響を与えません。ログ・リプレイを最適に実施できるように、高速復旧中に追加のロギングが使われて、バッファプールのアシストを行います。物理ログがすべてのバッファプールの合計サイズよりはるかに大きい場合には、高速復旧中にページ・フラッシングとページ・フォールティングが発生します。ページ・フラッシングとページ・フォールティングは高速復旧パフォーマンスを大幅に低下させ、データベース・サーバーは、`RTO_SERVER_RESTART` ポリシーを維持することができなくなります。

バッファプール領域が 4GB 未満のシステムでは、物理ログのサイズをすべてのバッファプールの合計サイズの 110%にすることができます。これよりも大きなバッファプールに対しては、4GB の物理ログ領域からスタートして、チェックポイント・アクティビティのモニタを行ってください。チェックポイントの発生頻度が高すぎて、パフォーマンスに影響を与えているように思える場合には、物理ログのサイズを増やしてください。

データベース・サーバーが小さな物理ログで構成され、多数のユーザを有している場合には、まれですが物理ログのオーバーフローと呼ばれる状態の起こることがあります。サイズに関する上記のガイドラインに従うことで、物理ログのオーバーフローの回避を容易に行えます。データベース・サーバーが次善の構成を検出すると、必ず、メッセージ・ログに対してパフォーマンスに関する警告を出します。

次善の構成が検出された場合には、`onstat -g ckp` コマンドを使って、推奨構成を表示させることができます。

#### IDS 11 用の新しい ONCONFIG パラメータ

表 3 ONCONFIG パラメータ

構成パラメータ	内容 / コメント
RTO_SERVER_RESTART	ユーザが Dynamic Server を再起動し、サーバーをオンラインまたは静止モードにしてからサーバーが問題から復旧するまでに必要な時間数（秒）を、リカバリ時間目標（RTO）標準を使って設定することができます。
RAS_PLOG_SPEED	高速復旧中に物理ログが復旧できる速度。
RAS_LLOG_SPEED	高速復旧中に物理ログが復旧できる速度。これは、構成のできないパラメータです。実際の復旧速度を反映させるために、IDS はこれらの値を更新します。（単位は、1 秒あたりのページ数です。）
AUTO_CKPTS	自動チェックポイントを有効または無効にします。
AUTO_LRU_TUNING	自動 LRU チューニングを有効または無効にします。

AUTO_AIOVPS	AIO VP が入出力作業負荷に追従できないことをサーバーが検出した場合に、AIO VP とフラッシュ・スレッドの数を自動的に増やすデータベース・サーバーの機能を有効または無効にします。
SQLTRACE	追跡する SQL 文の数などのデフォルト挙動とクエリ・ドリルダウン機能の追跡モードを制御します。
EXPALIN_STAT	SQL の SET EXPLAIN 文または onmode -Y session_id コマンドを表示できる explain.out ファイルへクエリ統計データ域を含めることを有効または無効にします。
USELASTCOMMITTED	ロック発生時に最後にコミットしたデータのバージョンをデータベース・サーバーが使用するか否かを指定します。
SHMVIRT_ALLOCSEG	Dynamic Server がサーバー・メモリを割り当てるときのしきい値、ならびにサーバーが新しいメモリ・セグメントを割り当てられない場合に出されるアラーム・レベルを割り当てるときのしきい値を指定します。
ENCRYPT_HDR	HDR ペア内のサーバー間の暗号を有効または無効にします。
LOG_INDEX_BUILDS	1 に設定して、インデックス文の実行中におけるインデックス・ページのロギングを有効にします。遠隔スタンドアロン副 (RSS) ノードを使用する場合には、プライマリ・ノード上にこのパラメータが必要です。
ENCRYPT_SMX	0 : SMX 接続時に暗号をしません。 1 : SMX 接続時に暗号の交渉を行います。 2 : SMX 接続時に暗号を使わなければなりません。

## 第7章 セキュリティの構成

### ラベルをベースとしたアクセス制御 (LBAC) セキュリティ機能

LBAC は、セキュリティ・ラベルをテーブル・オブジェクトへ付加することによってこれらのオブジェクトへのアクセスを制御します。オブジェクトへのアクセスを試みているユーザは、これらのオブジェクトに与えられたセキュリティ・ラベルを有していなければなりません。セキュリティ・ラベルが一致すると、アクセスが許可され、セキュリティ・ラベルが一致しない場合には、アクセスは拒絶されます。

セキュリティ・ラベルには、以下の3種類があります。

1. 行セキュリティ・ラベル：データベース・テーブル内のデータ行またはレコードに関するセキュリティ・ラベルです。
2. 列セキュリティ・ラベル：データベース・テーブル内の列に関するセキュリティ・ラベルです。
3. ユーザ・セキュリティ・ラベル：データベース・ユーザに与えられたセキュリティ・ラベルです。

セキュリティ・ラベルは、1 つまたは複数のセキュリティ・ラベル・コンポーネントから構成されます。セキュリティ・ラベルの構築に使用することのできるセキュリティ・ラベル・コンポーネントには、次の3種類があります。

1. セット：セットとは、それらの要素が現れる順番が重要でない要素の集合体を言います。すべての要素は同等と見なされます。
2. アレイ：アレイとは、簡単な階層を表すのに使用できる順番付きのセットを言います。アレイの中では、要素が現れる順番が重要です。たとえば、最初の要素はランクがもっとも高く、2番目の要素は3番目の要素よりも高いランクを有しています。
3. ツリー：ツリーは、複数のノードとブランチを持つことができる複雑度の高い階層を表します。たとえば、ツリーは、組織図を表すのに使うことができます。

ユーザはセキュリティ・ポリシーを使って、特定のセキュリティ・ラベルを構成するセキュリティ・ラベル・コンポーネントの定義を行うことができます。LBAC オブジェクトを

操作するには、DBSECADW が必要です。

UNIX 上または Linux 上で作動するシステムに使用するプラグブル認証モジュール

プラグブル認証モジュール (PAM) は、最初に Sun Microsystems が開発した各種の認証モジュールをサポートする高性能フレームワークです。

システム管理者は、PAM を使って異なるアプリケーションに対して異なる認証メカニズムを実行することができます。たとえば、UNIX ログイン・プログラムのようなシステム・ニーズは、データベースの機密情報へアクセスするアプリケーションとは違っているかもしれません。アプリケーション・レベルで認証サービスが与えられるので、PAM は、1 台のマシンの中でそのような多くのシナリオに対応することができます。

PAM は、必要に応じてアプリケーションに認証を選択させることができるうえ、モジュールの重ね合わせを行うこともできます。多数のモジュールを順次重ね合わせることができるので、アクセスを許可する前に複数の方法でアプリケーションの認証を行うことができます。PAM は一連の API を提供して、認証、アカウントの管理、セッションの管理、およびパスワードの管理をサポートします。

システム管理者は、PAM の使用を有効または無効にすることができます。デフォルトでは、データベース・サーバーは、ユーザにおける大変更の強制実行を回避するために、従来の Informix 認証メカニズム (BSD rhosts メカニズムをベースとしたもの) を使用します。

Dynamic Server で PAM を使用する場合には、以下を守ってください。

- ・ユーザの Informix Dynamic Server が PAM をサポートするオペレーティング・システム・プラットフォーム上になければなりません。
- ・ユーザのクライアント・アプリケーションが適度に新しいバージョンのクライアント SDK を使って記述されていなければなりません。
- ・ユーザは、オペレーティング・システムの中に構成された適切な PAM サービスを持つ

ていなければなりません。

- ・ユーザは、PAM サービスが与えられたパスワードを単純に受け入れるだけのものなのか、あるいはチャレンジ応答プロトコル（たとえば、RADIUS 認証サーバー）を使用するものなのかを知っていなければなりません。

- ・ユーザの PAM サービスがチャレンジ応答プロトコルを使用している場合には、ユーザは、チャレンジと応答を扱うためにそのアプリケーションを修正しなければなりません。アプリケーションは、PAM モジュールが複数のチャレンジを作成できることを認識しなければなりません。

- ・ユーザは、エンタープライズ・レプリケーションと高可用性データ・レプリケーションが PAM 認証の影響を受けないようにしなければなりません。

- ・クライアント・アプリケーションとデータベース・サーバーの両方に対して sqlhosts ファイル内のサーバー・エントリを修正しなければなりません（これらが別のマシン上にあるか、あるいは同じマシンの別の場所にある場合）。

プラグブル認証モジュールは、Solaris、Linux、HP-UX、および AIX®の 32 ビット・モードならびに 64 ビット・モードでサポートされています。

#### Windows 上での LDAP 認証のサポート

Windows 上での LDAP 認証の設定と構成は、UNIX 上または Linux 上で使用するプラグブル認証モジュール（PAM）と同じように行います。システム・ユーザの認証のために LDAP サーバーを使用したい場合には、LDAP 認証サポート・モジュールを使用します。このモジュールには、ユーザに固有の LDAP 認証サポート・モジュールに合わせてユーザによる修正が可能なソース・コードが含まれています。

認証モジュールは、通常は%INFORMIXDIR%\dbssodir\lib\security ディレクトリに常駐している DLL です。この認証モジュールのパラメータは、%INFORMIXDIR%\dbssodir\pam.conf ファイルにリストアップされています。%INFORMIXDIR%\demo\authentication ディレクトリには、全機能 LDAP 認証モジュールのソース・コードと必要な構成ファイルのサンプルが含まれています。

LDAP 認証モジュールは、シングル・モジュール認証のみ提供します。このモジュールは、モジュール・スタッキングのような機能はサポートしていません。システム・アドミニストレータは、認証を有効または無効にすることができます。

#### セッション・プロパティの構成

ユーザは、セッションが実行されているアプリケーションを変えずに、接続時またはアクセス時にデータベース・サーバー・セッションのプロパティを変更することができます。これは、環境オプションまたは環境変数を設定したいのに、あるいは SQL 文にベンダが獲得したコードが入っているなどの理由でセッションに関連した SQL 文を入れないのに、アプリケーションのソース・コードを修正できない場合に有効な方法です。

セッションのプロパティを変更するには、特定ユーザまたは PUBLIC グループのアプリケーションのサポートができるように、各種データベースの sysdbopen( )手順と sysdbclose( )手順のカスタム デザインを行います。sysdbopen( )手順と sysdbclose( )手順は、データベースを開いたとき、または閉じたときにデータベース・サーバーがユーザまたは PUBLIC グループに対して実行する一連の SET 文、SET ENVIRONMENT 文、SQL 文、または SPL 文を持つことができます。

user1 を例にとると、user1 が DATABASE 文または CONNECT TO 文を使ってデータベースを開いたときに必ず実行される SET PDQPRIORITY 文、SET ISOLATION LEVEL 文、SET LOCK MODE 文、SET ROLE 文、または SET EXPLAIN ON 文が入った手順を定義することができます。

sysdbopen( )手順の中の SET ENVIRONMENT 文を使って指定したセッション環境変数の PDQPRIORITY と OPTCOMPIND の設定は、セッションの持続時間に対して存在します。通常の手順では存在しない SET PDQPRIORITY 文と SET ENVIRONMENT OPTCOMPIND 文は、sysdbopen( )手順にこれらが入っている場合に存在します。

手順のオーナーであるユーザがデータベースからの切り離しを行うと、user.sysdbclose( )手順が実行されます (あるいは、PUBLIC.sysdbclose( )手順が存在し、現在のユーザが

sysdbclose( )手順を所有していない場合には、PUBLIC.sysdbclose( )手順が実行されます)。

## 第 8 章 まとめ

このチュートリアルで、ユーザは、IDS のインストールと構成、領域とログの管理、ならびに IDS で使用できる各種のセキュリティ・オプションの構成について学習しました。

インストール中に使用できる各種オプション、ならびにインストールの応答を簡単な ASCII ファイル (将来のすべてのサイレント・インストールに対して使用可能です) へ記録することを可能にする新しいデプロイメント機能について学習しました。

このチュートリアルでは、データ格納用の領域を作成する方法、sqlhosts ファイルへ対応のサーバー入力を行う方法、データベース・サーバーと接続するためにクライアント・アプリケーションで必要な環境変数を設定する方法、構成ファイルを使ってシステムを構成する方法などの簡単なデータベース サーバーを構成する際に実行するステップについて説明しました。このチュートリアルでは、IDS V11 の新しい ONCONFIG パラメータもリストアップしてあります。

このチュートリアルでは、格納の論理単位と物理単位、ならびに onspaces コマンドと sql Admin API を使って DB 領域、SD 領域、および BLOB 領域の追加と破棄を行う方法について説明しました。また、onparams コマンドを使ってログを管理する方法、ならびに論理ログと物理ログの数とサイズを決定する方法についても説明しました。

加えて、このチュートリアルでは、IDS V11 の新しい機能、LBAC セキュリティ、および使用可能な他のセキュリティ・オプションについても紹介しました。

このチュートリアルのパート 2 では、IDS V11 で使用できるモニタリング・ユーティリティを紹介します。

リソース

## 学習

- ・ developerWorks Informix zone : 各種記事とチュートリアルをお読みにになり、他の参考資料と併せて Informix スキルを広げてください。
- ・ IBM Informix Dynamic Server Information Center : Informix について詳しく学んでください。
- ・ IBM Informix Dynamic Server 11 BETA Information Center : IDS 11 について詳しく学んでください。
- ・ developerWorks IDS Experts blog : 開発エンジニアと技術サポート エンジニアの国際的チームが作成した Informix Dynamic Server に関する技術ノートをご覧ください。
- ・ IBM Information Management certification page : IDS 認証のためのリソースについて詳しく学んでください。
- ・ developerWorks Information Management zone : 情報管理について詳しく学んでください。技術資料やハウツー記事、教育資料、ダウンロード、製品情報など、豊富な資料が用意されています。
- ・ developerWorks technical events and webcasts で最新情報を入手してください。
- ・ Technology bookstore : この記事や他の技術的な話題に関する本が豊富に取り揃えられています。

## 製品の入手と技術の獲得

- ・ Informix Dynamic Server Enterprise Edition V10.0 : 無料トライアル版をダウンロードできます。
- ・ Informix Dynamic Server 11 : このチュートリアルと併せて作業するための無料トライアル版をダウンロードできます。
- ・ IBM product evaluation versions : 情報管理、Lotus®、Rational®、Tivoli®および WebSphere®からアプリケーション開発ツールとミドルウェア製品をダウンロードし、入手することができます。

## ディスカッション

- ・ この内容についてのディスカッション・フォーラムに参加してください。
- ・ developerWorks のブログをチェックして、developerWorks コミュニティへ参加でき

ます。

#### 著者について

Manjula Panthagani

Manjula Panthagani は、IBM の Informix Dynamic Server のアドバンスト・サポート・エンジニアです。彼女は、IDS のサポートを行うこの職務に 7 年以上携わっていて、IDS 認証 Exam の開発に加わっています。

Rashmi Chawak

Rashmi Chawak は、IBM の IDS のアドバンスト・サポート・エンジニアです。彼女は、IDS のサポートを行うこの職務に 7 年以上携わっていて、各種プラットフォームへの IDS 製品のポーティングの業務を担当しています。

Siva K. Ane

Siva K. Ane は、IBM の IDS のアドバンスト・サポート・エンジニアです。彼は、この職務を担当する前は Platform Engineering に 9 年在籍していました。

#### 商標

IBM、Informix

UNIX