Sybase® IQ クイック・スタート

ドキュメント ID：DC01146-01-1530-01
改訂：2011 年 5 月

概要

このマニュアルでは、IQ デモ・データベースの作成とクエリの方法について説明します。データベースは、選択したディレクトリ内にコピーを作成し、いつでも再作成できます。

また、デモ・データベースを使用して、IQ マルチプレックスを作成したり使用したりすることもできます。IQ マルチプレックスとは、共有ディスク・クラスタ経由で同じ IQ ストアにアクセスする複数の IQ サーバを指します。デモ用に、複数の IQ サーバで共有されるファイル・システム・ファイルを使用して IQ マルチプレックス環境をシミュレートできます。

データベースをクエリまたは更新するには、Interactive SQL (dbisql) または Sybase Central™ のどちらでも使用できます。IQ サーバのインストールにはこの両方が付属しています。

このマニュアルにはフォーマット用に文字が追加されているため、コマンドをこのマニュアルから直接カット・アンド・ペーストしないでください。コマンドを手動で入力するか、メモ帳または vi にカット・アンド・ペーストし、コマンドを編集してフォーマット文字列を削除してから、コマンド・ラインにコピー・アンド・ペーストします。

デモ・データベース

このデモ・データは、想像上のスポーツ用品会社に対応しています。デフォルトで iqdemo というデータベースが Sybase® IQ 15.3 用に更新されています。

テーブル名

旧エディションからアップグレードする場合、次の表のリストで iqdemo データベースの最新のテーブル名を確認してください。スクリプトまたはアプリケーションを更新するには、このテーブルを使用します。

表 1：デモ・データベースのテーブル名

12.7 での名前	15.3 での名前	15.3 でのテーブル所有者
contact	Contacts	GROUPO
customers	Customers	GROUPO
department	Departments	GROUPO
emp1	emp1	DBA
employee	Employees	GROUPO
fin_code	FinancialCodes	GROUPO
fin_data	FinancialData	GROUPO
product	Products	GROUPO
sale	sale	DBA
sales_order_items	SalesOrderItems	GROUPO
sales_order	SalesOrders	GROUPO

IQ デモ・データベースの作成と使用

1 Sybase IQ サーバをインストールします。『Sybase IQ インストールおよび設定ガイド』を参照してください。

2 demo データベースを作成します。

注意 重要：demo データベースをマルチプレックスに変換する場合は、絶対パス (-absolute) を使用してデータベースを作成してください。

- **UNIX の場合**

  1 IQ インストール・ディレクトリに移動します。

  - `cd <IQ install directory>/IQ-15_3`

  2 IQ 環境を参照します。

  - bash または Korn (ksh) シェルの場合は、`.IQ-15_3.sh` と入力します。
  - tcsh または C (csh) シェルの場合は、`source IQ-15_3.csh` と入力します。

  3 まだ作成されていない場合は、新しいデータベースを格納するディレクトリを作成します。このデモの規則に従い、ディレクトリは */myiqdemo* です。

4 データベースの作成先のディレクトリに移動します。

- `cd /myiqdemo`

5 次のコマンドを入力します。

- `$IQDIR15/demo/mkiqdemo.sh`

- **Windows の場合**

1 コマンド・ウィンドウを開きます。

- **[スタート] - [プログラム] - [MS-DOS プロンプト]** をクリックするか、**[スタート] - [ファイル名を指定して実行]** をクリックし、`cmd` と入力します。

2 まだ作成されていない場合は、新しいデータベースを格納するディレクトリを作成します。このデモの規則に従い、ディレクトリは *C:¥myiqdemo* です。

3 次のいずれかの方法を使用して、データベースの作成先のディレクトリに移動します。

- `c:`
- `cd C:¥myiqdemo`

4 次のコマンドを実行します。

```
"%ALLUSERSPROFILE%¥SybaseIQ¥demo¥mkiqdemo"
```

5 事前に定義された場所で demo データベースを作成するには、次のように選択します。

[スタート] - [プログラム] - [Sybase] - [Sybase IQ デモデータベースの起動]

- データベースの作成をカスタマイズするためのオプションをすべてリストするには、次を実行します。
  - **UNIX の場合**
    `$IQDIR15/demo/mkiqdemo.sh -help`
  - **Windows の場合**
    `"%ALLUSERSPROFILE%¥SybaseIQ¥demo¥mkiqdemo" -help`

IQ データベースは複数のストアから構成されています。IQ ストアへのパスには相対パスか絶対パスのいずれかを指定できます。

相対パスまたは絶対パスを使用して demo データベースを作成できます。demo データベースをコピーまたは移動する場合は、相対パスを使用した方が簡単です。これは、*mkiqdemo* ではデフォルトのオプションです。ただし、demo データベースをマルチプレックスに変換する場合は、絶対パスを使用してください。

- **UNIX の場合**

  1 データベースの作成先のディレクトリに移動します。

  ```
  cd /myiqdemo
  ```

  2 -absolute スイッチを使用します。

  ```
  $IQDIR15/demo/mkiqdemo.sh -absolute
  ```

- **Windows の場合**

  1 コマンド・ウィンドウを開きます。

  [スタート] - [プログラム] - [MS_DOS プロンプト] を選択するか、**[スタート] - [ファイル名を指定して実行]** を選択し、cmd と入力します。

  2 データベースの作成先のディレクトリに移動します。

  ```
  c:
  cd ¥myiqdemo
  ```

  3 次のコマンドを実行します。

  ```
  "%ALLUSERSPROFILE%¥SybaseIQ¥demo¥mkiqdemo" -
  absolute
  ```

3 SQL Anywhere® または以前のバージョンの Sybase IQ が Sybase IQ 15.3 と同じマシンまたはサブネットに存在する場合は、*iqdemo.cfg* 設定ファイルを編集します。デフォルトでは、どちらの製品もデフォルトのポート 2638 を使用するため、IQ サーバのポートを変更してください。

さらに、共有システムに関して混乱を避けるために、ユーザ名をサーバ名に追加してユニークに設定します。たとえば、ユーザ名が *jsmith* である場合は、*jsmith_iqdemo* をサーバ名として demo データベースのサーバ名に指定できます。

iqdemo.cfg ファイルは、demo データベースと同じディレクトリ内に作成されます。このファイルを編集し、起動パラメータを変更できます。このファイルをコピーし、起動パラメータを変更することで、データベースの設定ファイルを作成することもできます。

4 IQ サーバを起動します。設定ファイルとデータベース・ファイルがあるディレクトリに移動し、次の形式でコマンドを発行します。

start_iq @*configuration_file.cfg* dbname.db

たとえば、demo データベースを起動するには、次のコマンドを入力します。

start_iq @iqdemo.cfg iqdemo.db

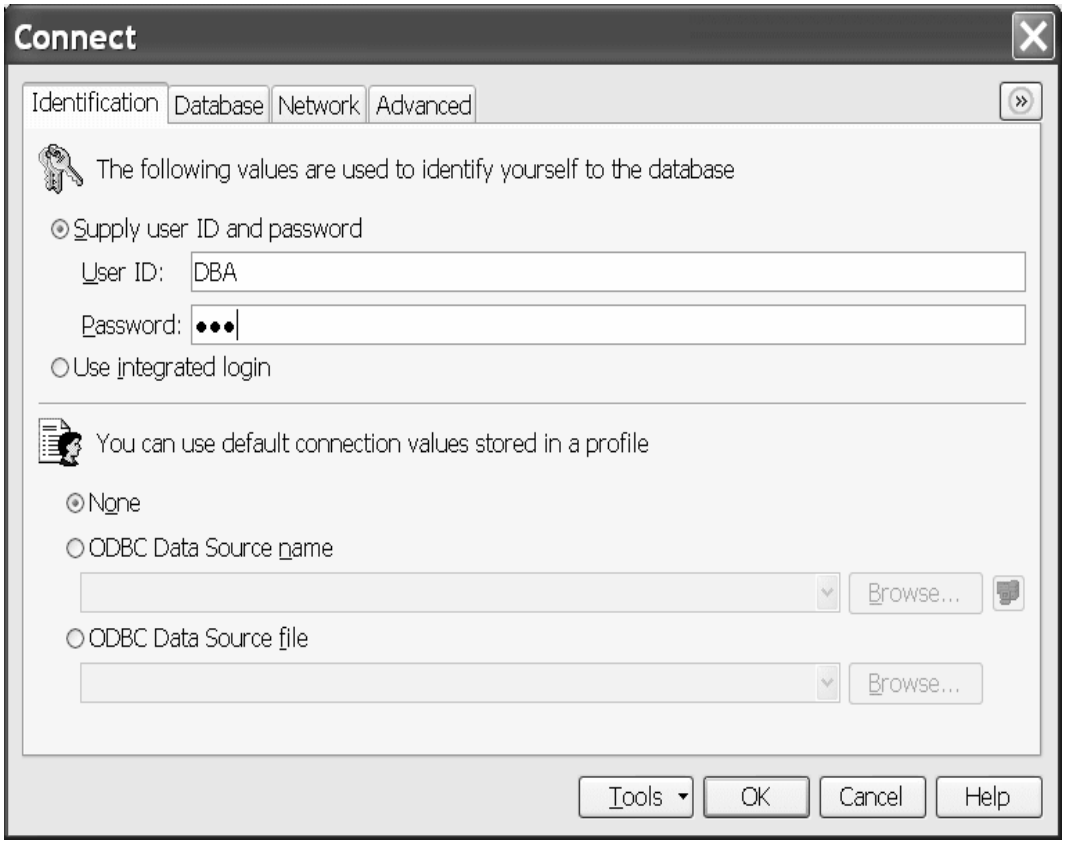
5 Interactive SQL (クエリ・ツール) を起動します。

- UNIX の場合は、システム・プロンプトで dbisql と入力します。
- Windows の場合は、**[スタート] - [プログラム] - [Sybase] - [**Sybase IQ 15.3**] - [Interactive SQL]** をクリックし、コマンド・シェルに dbisql と入力するか、**[スタート] - [ファイル名を指定して実行]** を選択し、dbisql と入力します。

6 demo データベースに接続します。

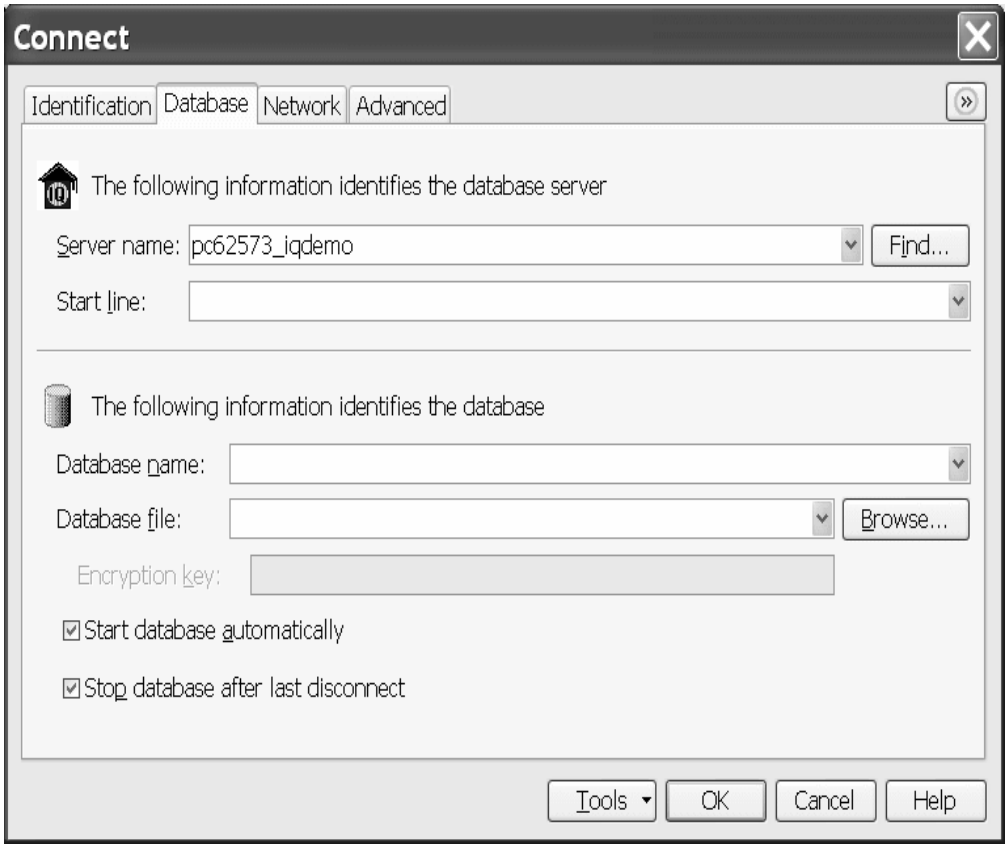
- [接続] ダイアログで、ユーザ ID として **DBA**、パスワードとして **sql** をそれぞれ入力します (この例は Windows の出力を示しています)。

図 1 : [接続] ダイアログ


Connect
Identification
Database
Network
Advanced
The following values are used to identify yourself to the database
Supply user ID and password
User ID:
DBA
Password:
Use integrated login
You can use default connection values stored in a profile
None
ODBC Data Source name
Browse...
ODBC Data Source file
Browse...
Tools
OK
Cancel
Help

- [データベース] タブで、**[検索]** をクリックし、サーバ名を選択します。サーバ名がリストされていない場合は、サーバ名を入力してから **[OK]** をクリックします。

図 2 : サーバの検索

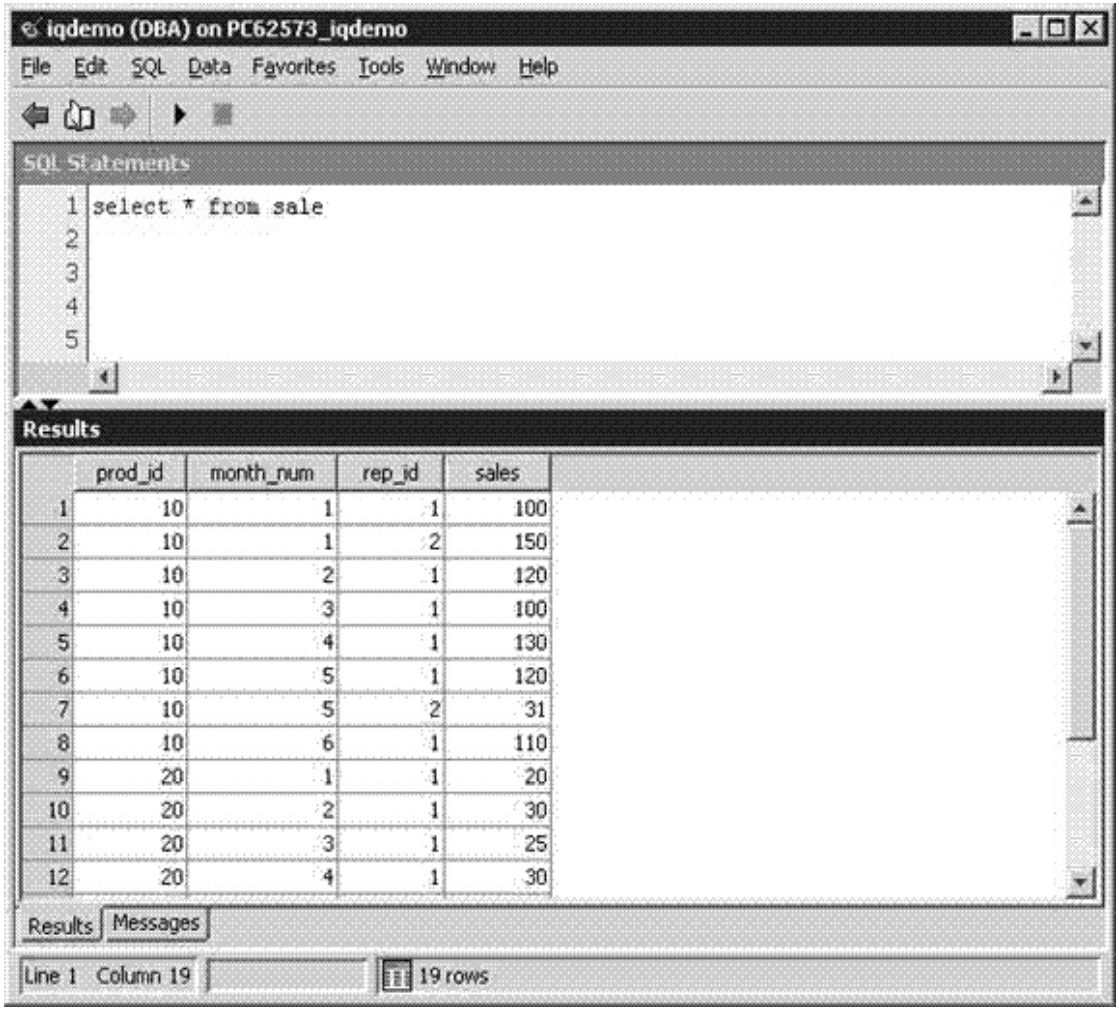


7 テスト・クエリを実行し、sale テーブル内のカラムを表示します。一番上のウィンドウ ([SQL 文]) に、次のコマンドを入力します。

```
SELECT * FROM sale
```

[実行] (ボタン・バーの右向きの三角形) をクリックします。

図 3：テスト・クエリの実行



8 Interactive SQL を終了するには、**[ファイル] - [終了]** を選択します。

IQ デモ・データベースのマルチプレックスへの変換

注意 重要 : demo データベースをマルチプレックスに変換する場合は、絶対パスを使用する必要があります。

demo データベースが絶対パス (-absolute オプション) を使用して作成されたことを確認します。demo データベースに相対パスが使用されている場合は、マルチプレックスを作成できません。

シングル・サーバ (IQ demo データベース) をマルチプレックス・サーバに変換するには、セカンダリ・サーバを追加します。Sybase Central を使用して、サーバを変換し、作成されたマルチプレックスを管理できます。マルチプレックス内のすべてのサーバが、1 つの [マルチプレックス] フォルダに表示されます。

IQ Agent の起動

Sybase Central を起動する前に、IQ サーバを実行するマシン上で IQ Agent を起動します。

UNIX の場合

1 *S99SybaseIQAgent15* スクリプトを使用して Agent を起動します。

- コマンド S99SybaseIQAgent15 は、デフォルトのポート 1099 で Agent を起動します。
- コマンド S99SybaseIQAgent15 ?port 3871 は、ポート 3871 で Agent を起動します。

2 Agent が実行されていることを確認するには、次のコマンドを実行します。

```
stop_iq -agent
```

たとえば、次の Agent は 'smith' というユーザが所有しています。サーバを停止しないでください。

```
##owner        PID   Started    CPU Time       Additional Information
-------      -----    ------     -------        ----------------------
-
1:smith     15549    Feb. 18    10:38           PORT:2008
java -Diq.agent=/sun625742/users/smith/sybase/IQ-
15_3/java/IQAgent1530.jar -D
```

3 Agent ログが存在することを確認します。

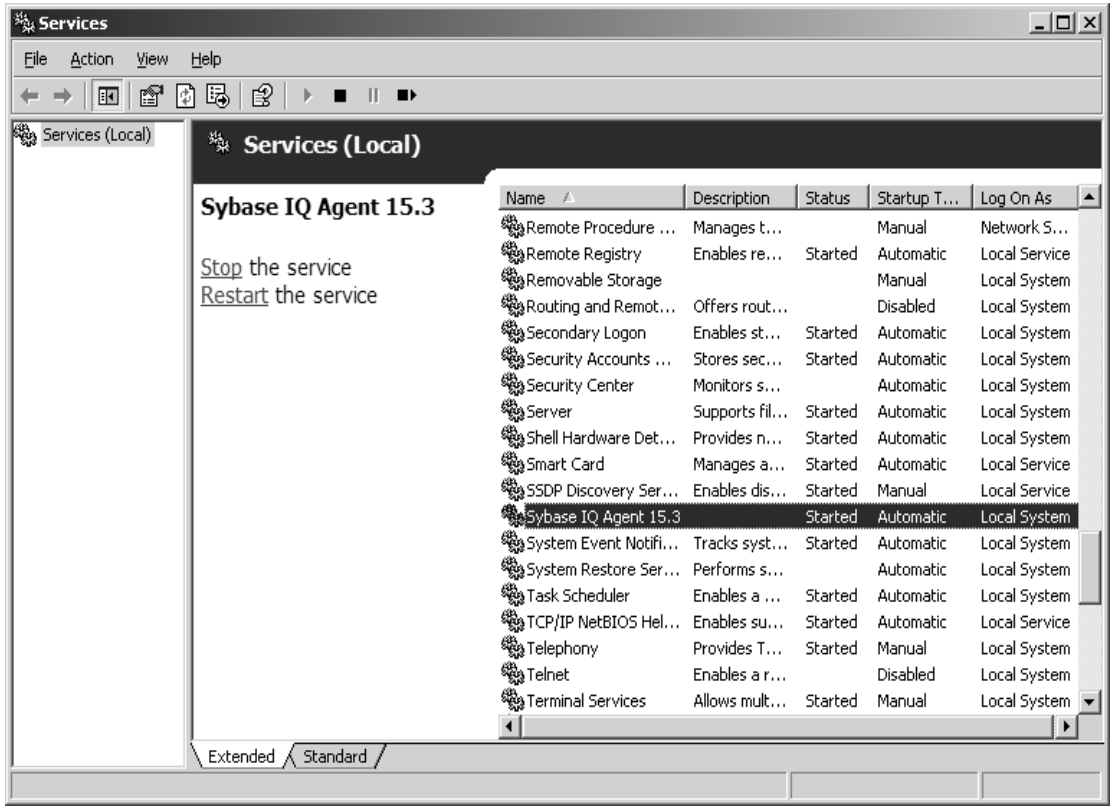
Agent ログ・ファイルは *$IQDIR15/logfiles/SybaseIQAgentNNNN.MMM.log* です (ここで、*NNNN* はポート番号で、*MMM* はシーケンス番号です)。環境変数 *IQLOGDIR15* が設定されている場合、Agent ログ・ファイルは *$IQLOGDIR15* ディレクトリにあります。

Windows の場合

Agent はサービスとして実行するように設定されます。

1 [管理ツール] の [サービス] を起動し、Agent が実行していることを確認します。

図 4 : [管理ツール] の [サービス]



2 Agent ログが存在することを確認します。

- Sybase IQ のインストール後に再起動した場合、Agent ログ・ファイルは *%ALLUSERSPROFILE%¥SybaseIQ¥logfiles¥SybaseIQAgent.NNN.log* です (ここで、*NNN* は連番です)。
- 再起動しなかった場合、Agent ログ・ファイルは *%SYBASE%¥IQ-15_3¥logfiles* にあります。
- 環境変数 *IQLOGDIR15* が設定されている場合、Agent ログ・ファイルは *%IQLOGDIR15%* ディレクトリにあります。

マルチプレックスへの変換

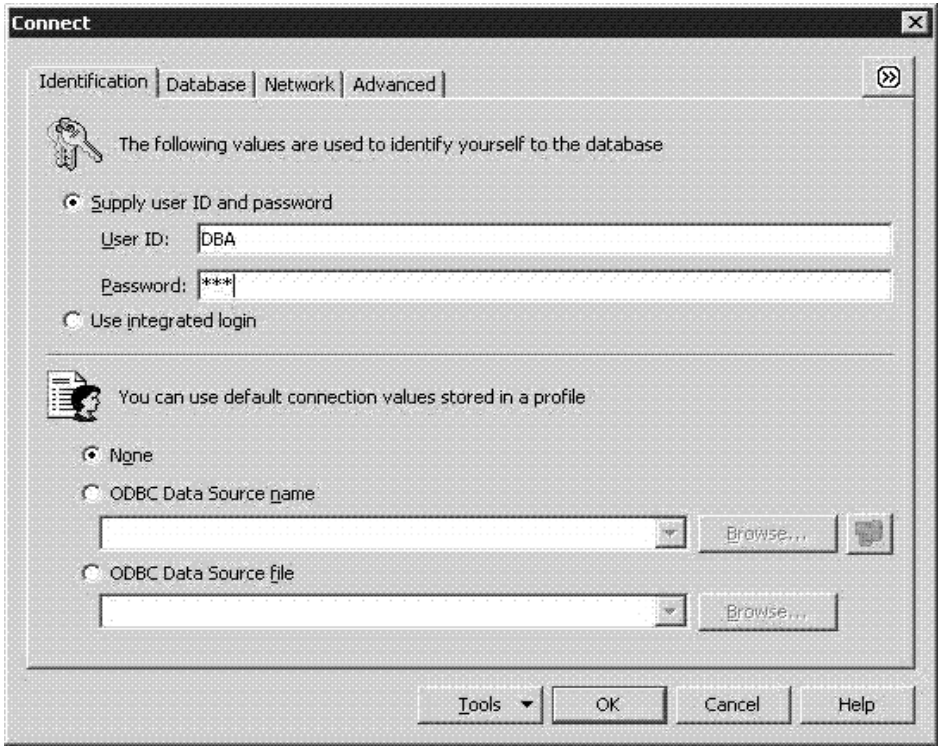
❖ **IQ デモ・データベースのマルチプレックスへの変換**

1 Sybase Central を起動します。

- UNIX の場合は、システム・プロンプトで **scjview** と入力します。
- Windows の場合は、**[スタート] - [プログラム] - [Sybase] - [Sybase IQ 15.3] - [Sybase Central]** を選択します。

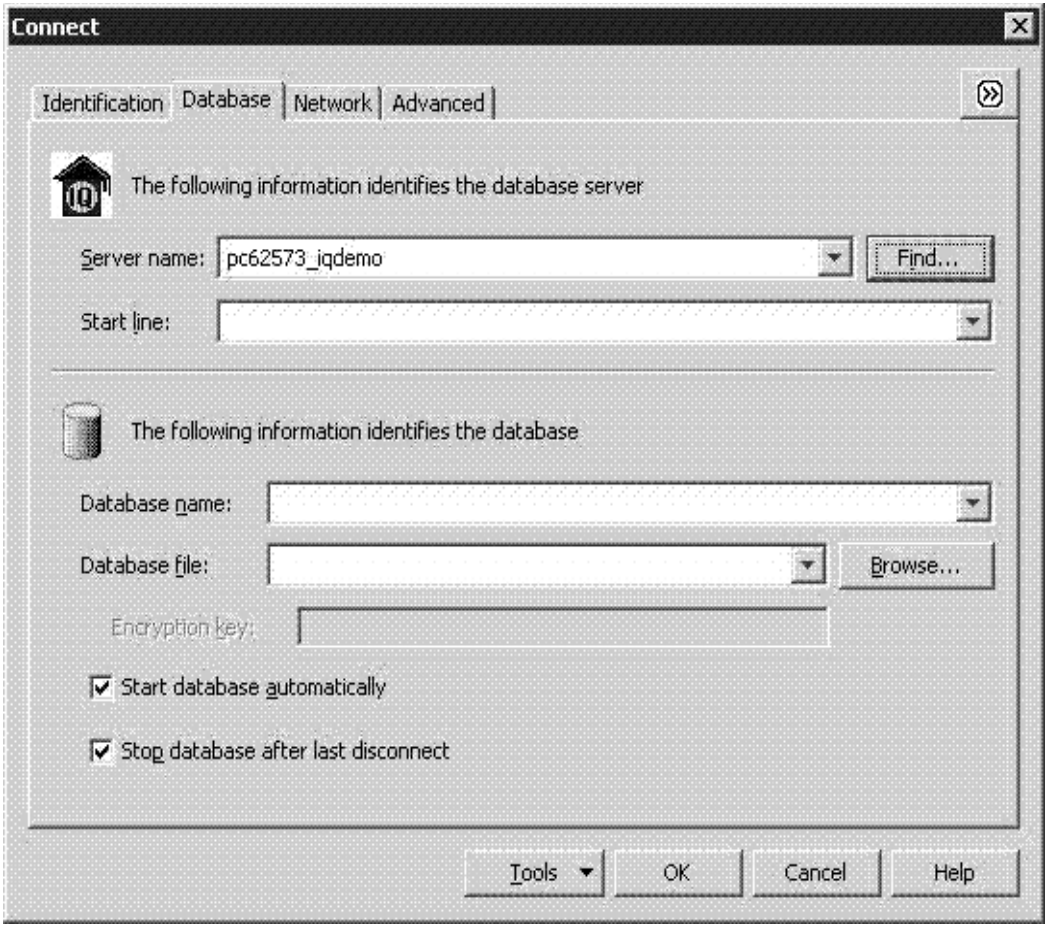
2 **[接続] - [Sybase IQ 15 に接続]** を選択します。[接続] ダイアログで、ユーザ ID として **DBA**、パスワードとして **sql** をそれぞれ入力します。

図 5 : [接続] ダイアログの [ID] タブ



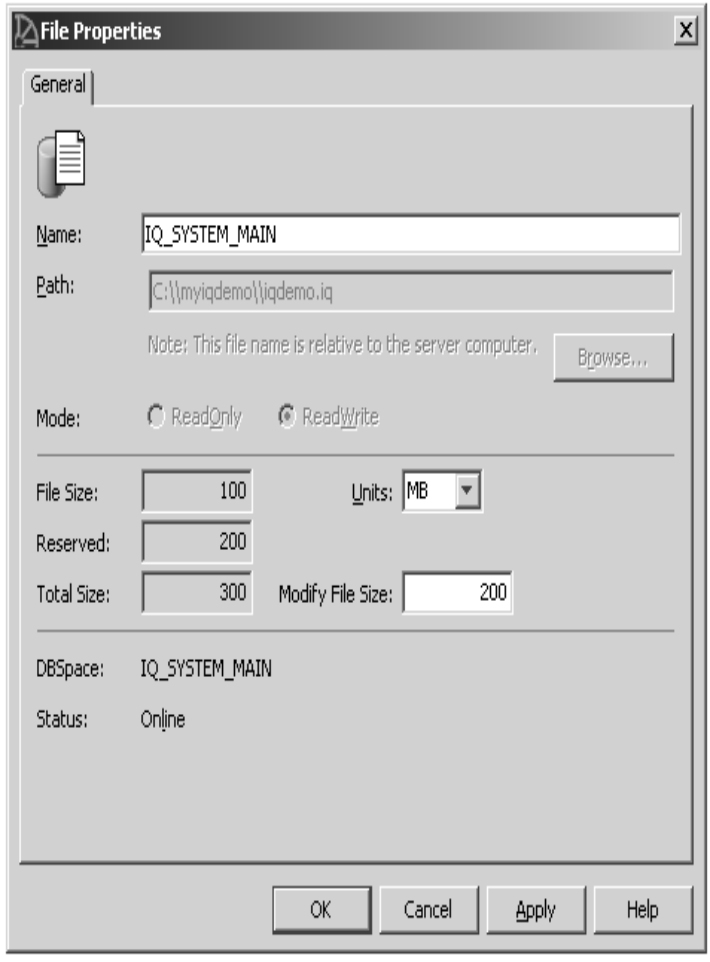
[データベース] タブで、**[検索]** をクリックし、サーバ名を選択します。サーバ名がリストされていない場合は、サーバ名を入力してから **[OK]** をクリックします。

図 6 : [接続] ダイアログの [データベース] タブ



3 IQ_SYSTEM_MAIN でファイルのサイズを増やします。
フォルダ・ビューで、[DB 領域] フォルダをダブルクリックします。次に、[IQ_SYSTEM_MAIN] フォルダを選択します。
[ファイル] タブで、[IQ_SYSTEM_MAIN] ファイルを右クリックし、**[プロパティ]** を選択します。[プロパティ] ダイアログの **[ファイル・サイズの変更]** に 200 と入力します。これで、IQ_SYSTEM_MAIN の領域が 200MB に増加します。

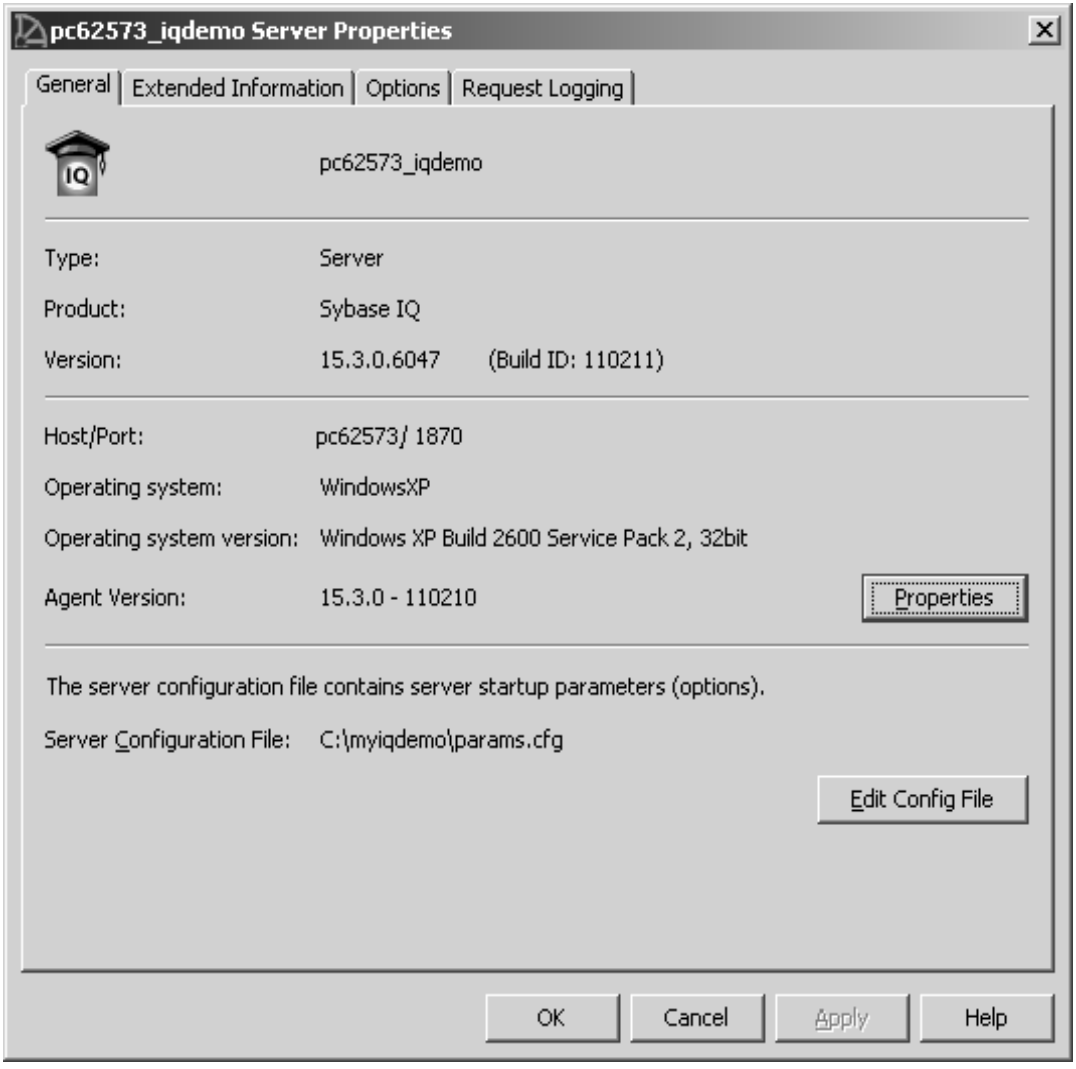
図 7：ファイルのプロパティ



4 **[適用]** をクリックし、次に **[OK]** をクリックします。

5 フォルダ・ビューで Agent のポートを確認するには、[サーバ] フォルダで **iqdemo** サーバのアイコンを右クリックし、**[プロパティ]** をクリックします。

図 8 : iqdemo サーバのプロパティ



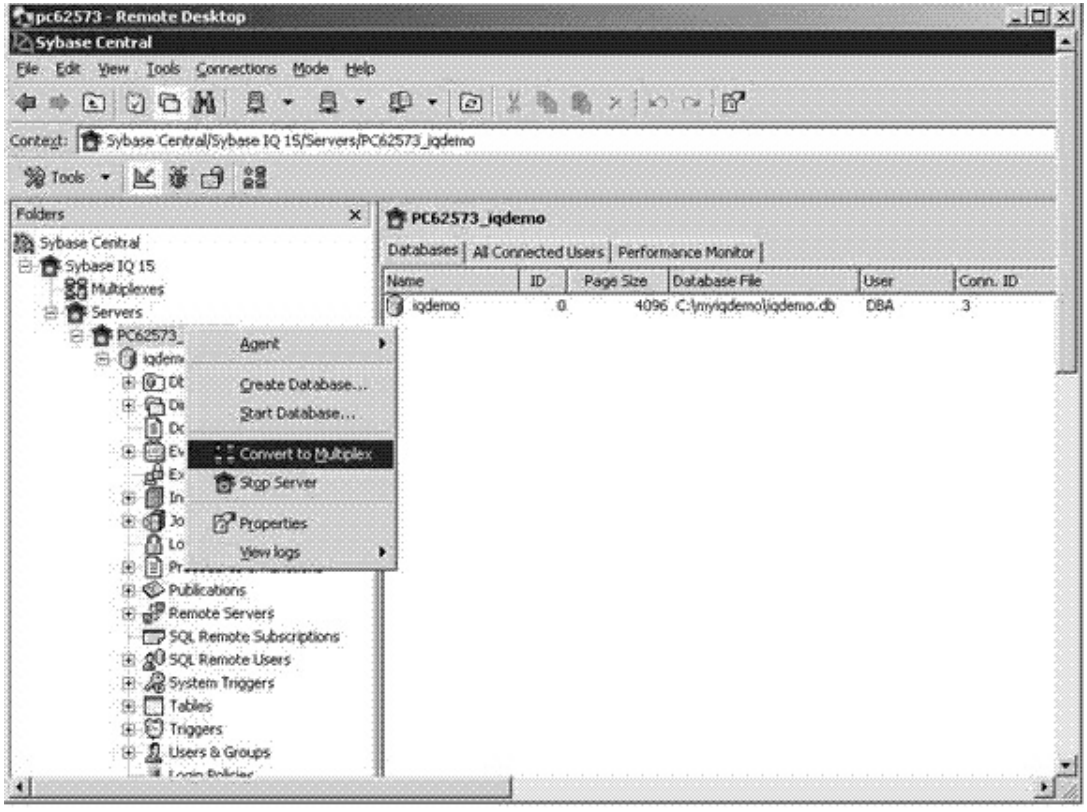
[エージェントのバージョン] の横の **[プロパティ]** をクリックします。

図 9 : Agent のプロパティ

ポート番号が Agent を起動したときのものであることを確認します。**[キャンセル]** をクリックすると、[サーバのプロパティ] ダイアログに戻ります。**[OK]** をクリックすると、フォルダ・ビューに戻ります。

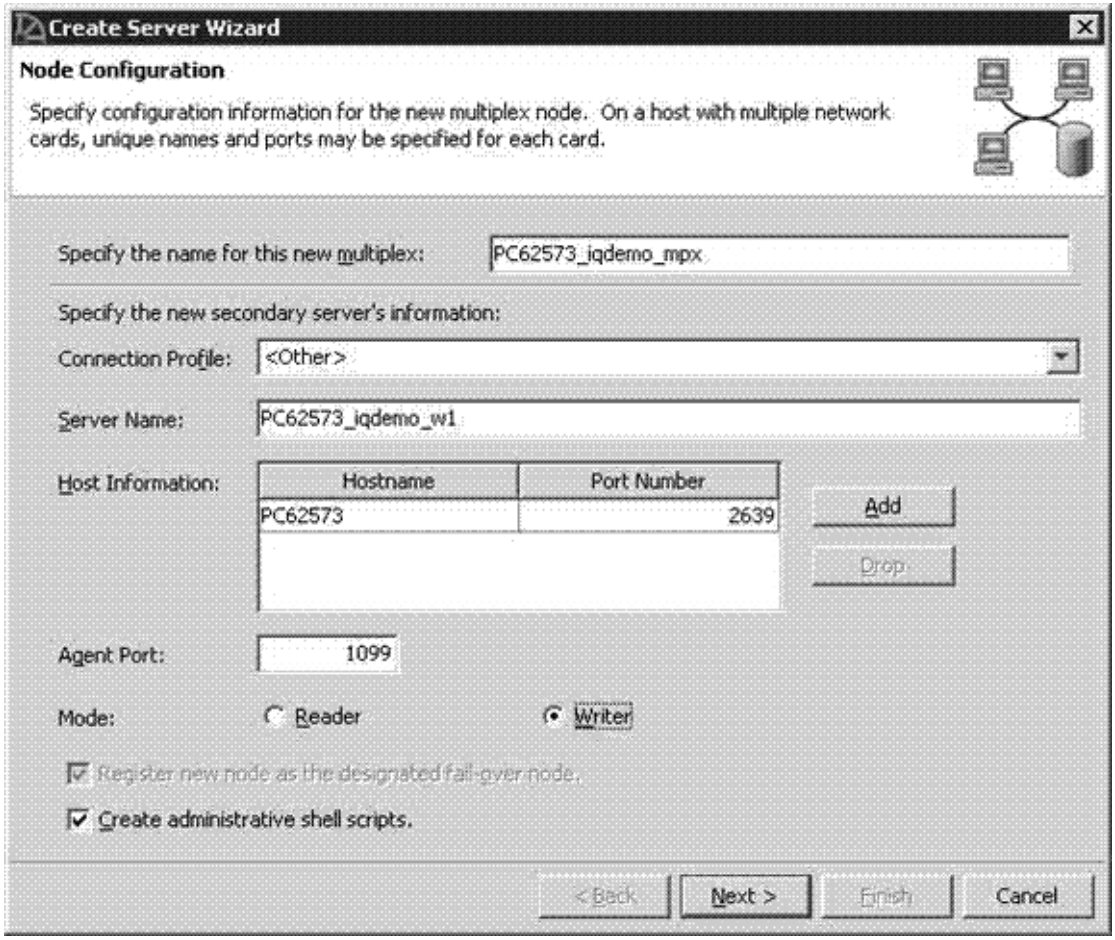
6 フォルダ・ビューの [サーバ] フォルダで **iqdemo** サーバのアイコンを右クリックし、**[Multiplex に変換]** を選択します。

図 10：フォルダ・ビュー



7　これにより、サーバの作成ウィザードが起動します。

図 11：サーバの作成ウィザード



次のすべてのフィールドに入力します。

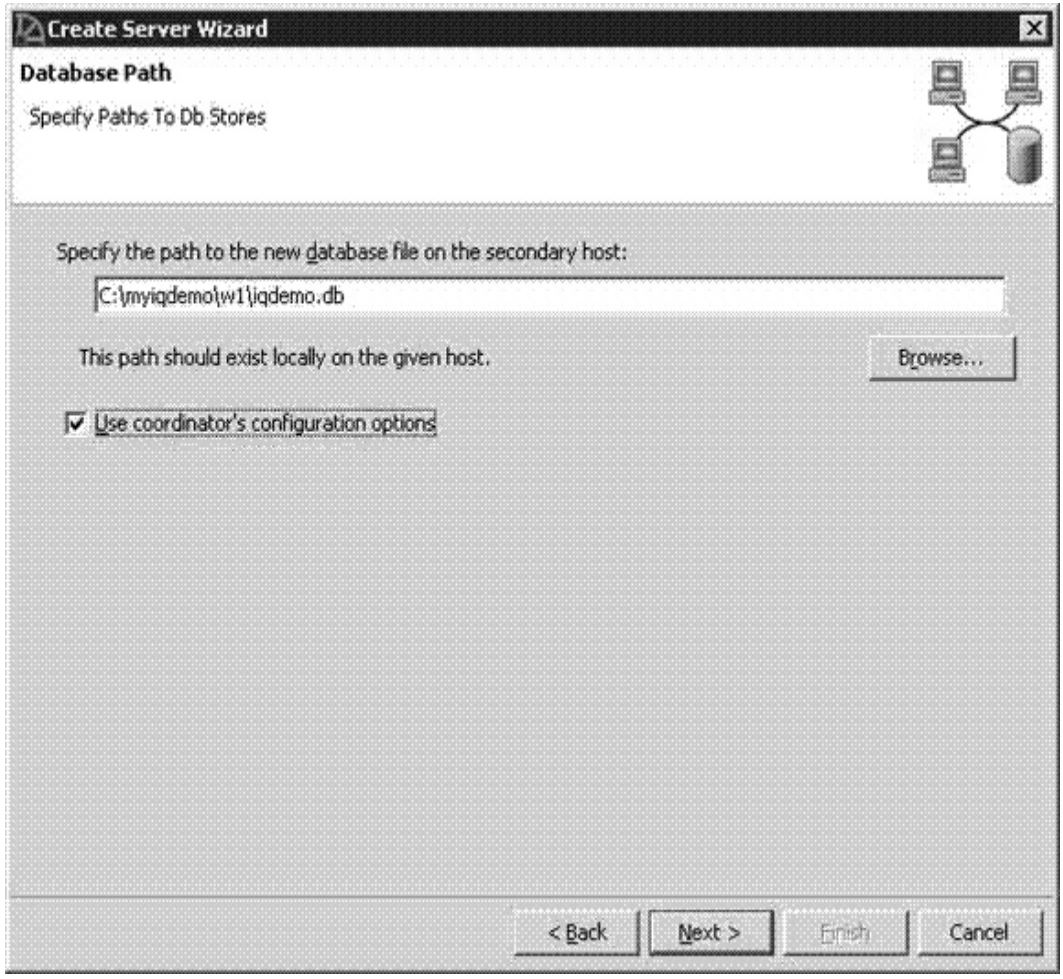
- マルチプレックス名
  - デフォルトでは元のシングル・サーバ名に設定されますが、このサーバ名は変更できます (たとえば、画面のように指定します)。
  - これは、Sybase Central 内でマルチプレックスを一意に識別するのに使用するマルチプレックスの論理名です。

- サーバ名
  - 最初のセカンダリ・サーバの名前。
  - これは、マルチプレックスのコーディネータ・サーバになる元のサーバとは異なる必要があります。
- ホスト情報
  - ホスト名 ― デモ用に、元のサーバと同じホストを使用します。これにより、複数のサーバで構成されるマルチプレックスが同じホスト上に設定されます。運用の設定では、マルチプレックス・サーバごとに異なるホストを使用します。
  - ポート番号 ― セカンダリ・サーバにユニークなポート番号を指定します。デフォルトでは、このポート番号は元のサーバのポート番号に 1 を足した番号になります。
  - ここでホストとポートのペアを複数追加することで、複数のネットワーク・アドレスを持つシングル・サーバを設定できます。ほとんどの場合、マルチプレックス・サーバのネットワーク・アドレスは 1 つのみです。
- プライベート・ホスト情報

  分散クエリ処理を使用するには、プライベートの高速相互接続用のホスト・アドレスとポート・アドレスを指定します。プライベート・ネットワークは TCP/IP プロトコルをサポートする必要があります。
- Agent ポート ― エージェントを起動するポートを入力します。デフォルトのポートは 1099 です。
- モード ― **[ライタ]** を選択します (デフォルトは [リーダ] です)。後で、このノードからデータをロードします。
- **[管理シェル・スクリプトを作成する]** を選択します。

8 **[次へ]** をクリックします。カタログ・データベースのパスを選択します。このパスは、IQ demo データベースを作成した場所とは異なるディレクトリに指定します。demo データベースを作成したディレクトリの下にサブフォルダを作成できます。

図 12 : データベース・パス



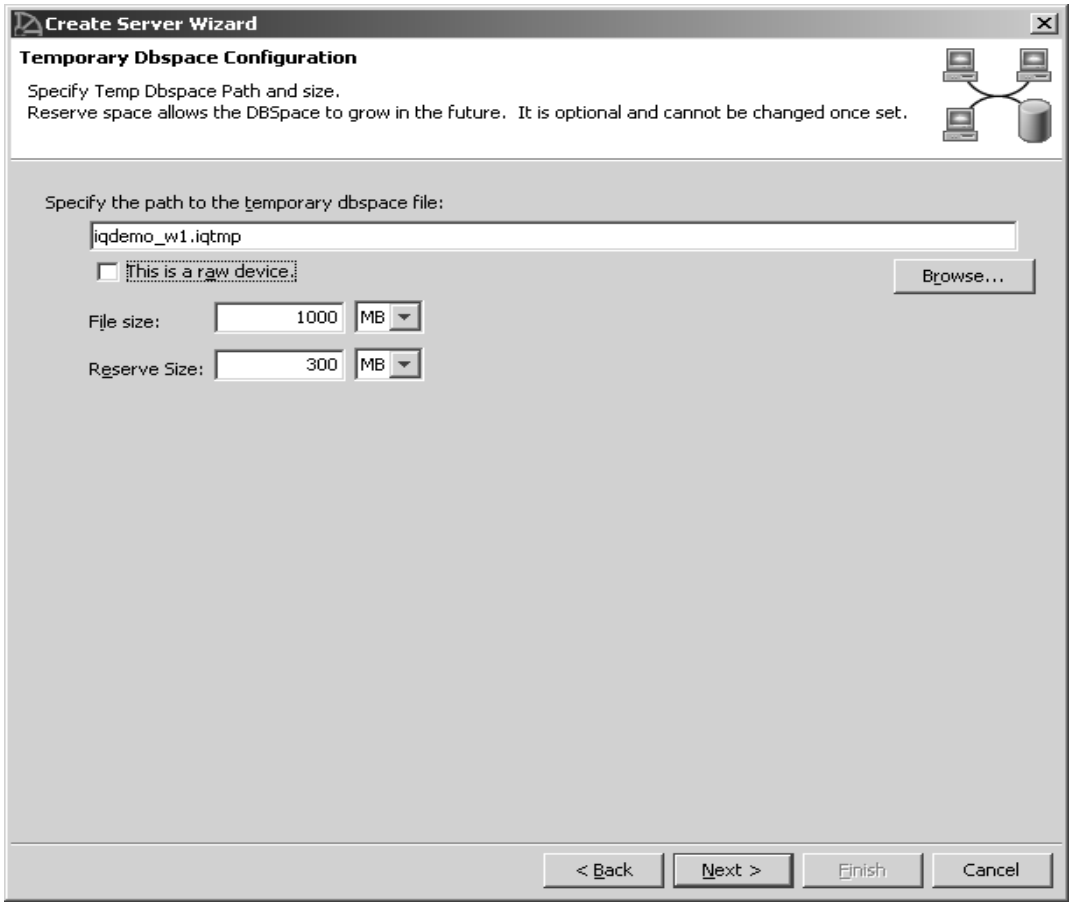
[データベース・パス] ページに情報を入力します。

- データベース・パス
  - データベース・パスは、新しいマルチプレックス・セカンダリ・サーバの *db* ファイルへの絶対パスです。
  - このパスは、既存のデータベースの *db* ファイル・パスとは異なる必要があります。Sybase では、*iqdemo.db* をファイル名として使用することをおすすめします。一般には、任意の名前を使用できます。

- **[コーディネータの設定オプションを使用]** を選択します。このオプションは、既存のデータベースに使用する params.cfg オプションを新しいマルチプレックス・サーバの *db* ディレクトリにコピーします。

9 [次へ] をクリックします。ディレクトリが存在しない場合は、作成を求めるメッセージが表示されたら **[はい]** をクリックします。[テンポラリ DB 領域設定] ページが表示されます。

図 13：テンポラリ DB 領域設定

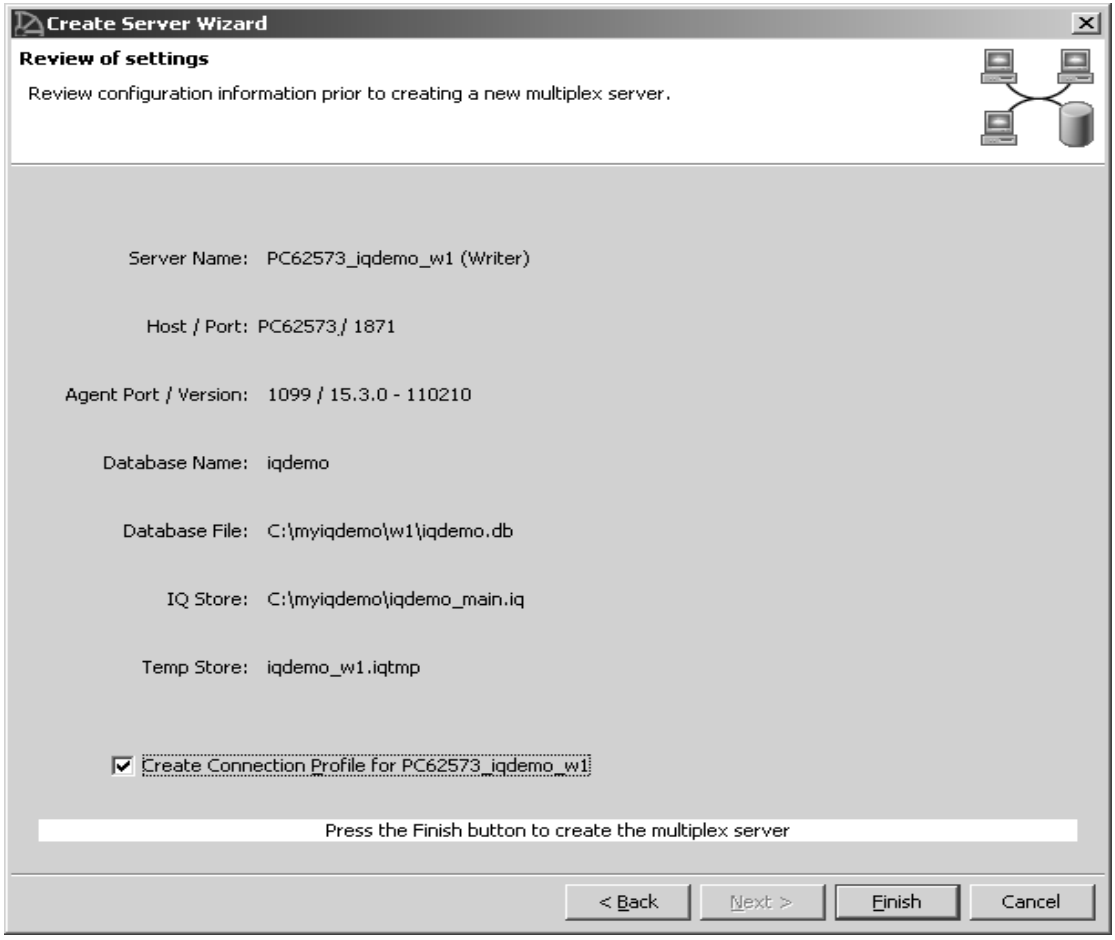


[テンポラリ DB 領域設定] ページに情報を入力します。

- テンポラリ DB 領域のファイル・パス：*iqdemo_w1.iqtmp*。テンポラリ DB 領域のファイル・パスには、絶対パス、またはデータベース *.db* ファイルを基準とする相対パスを指定できます。一意のファイル名を選択します。
- [これはロー・デバイス] ー 選択しないままにします。
- ファイルのサイズは 1000MB です。
- 予約サイズは 300 です。

10 **[次へ]** をクリックします。設定を確認します。

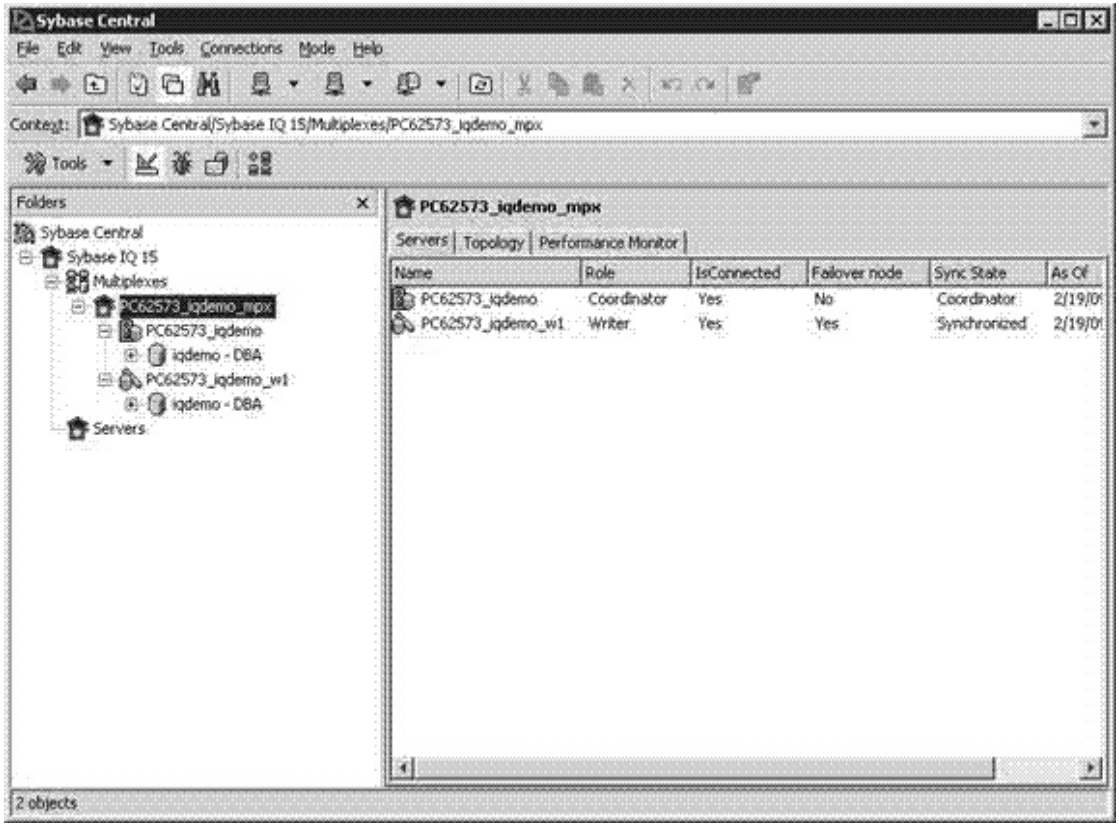
図 14：設定の確認



11 **[完了]** をクリックします。これにより、新しいマルチプレックス・セカンダリ・サーバ・ファイルが作成され、マルチプレックス用のデータベースが設定されます。この操作には数分かかる場合があります。

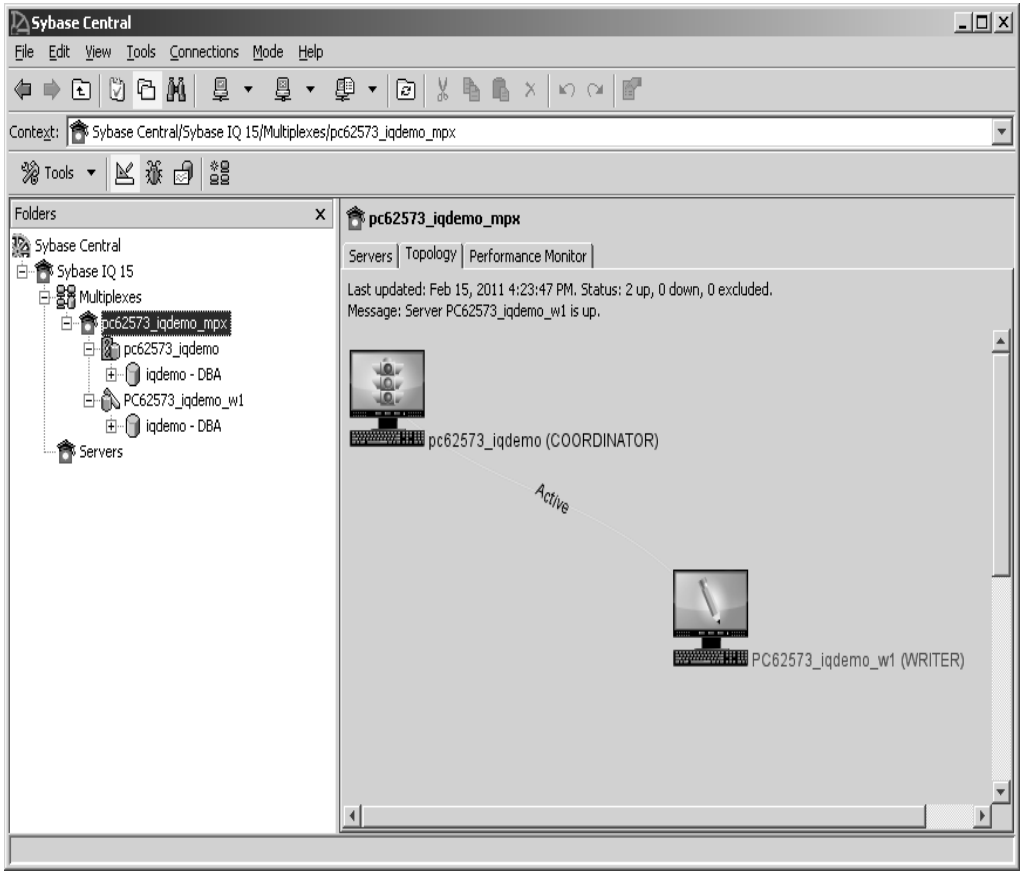
Sybase Central のフォルダ・ビューで、マルチプレックス名が [マルチプレックス] フォルダに挿入されています。マルチプレックス名を展開すると、マルチプレックス内のサーバを表示できます。

図 15：Sybase Central マルチプレックス・サーバ



12 Sybase Central のメイン・ウィンドウで、**[トポロジ]** タブをクリックすると、マルチプレックス・トポロジ・ビューが表示されます。これは、マルチプレックス設定をグラフィカルに表示したものであり、各サーバのステータスと、サーバ間の通信リンクが表示されます。ほとんどのマルチプレックス操作はこのビューまたはサーバ・ビューから実行できます。

図 16 : [トポロジ] タブ



マルチプレックス・サーバの追加 (手動による方法)

現在のマルチプレックス設定には、コーディネータ・サーバと 1 台のマルチプレックス・ライタ・サーバという 2 台のサーバが含まれています。この項の手順に従うと、リーダの役割で 2 番目のサーバを手動で追加できます。新しいリーダ・サーバの名前は PC62573_iqdemo_r2 です。

注意 重要 : demo データベースをマルチプレックスに変換する場合は、絶対パスを使用してください。

Interactive SQL を開きます。**[ファイル] - [サーバの追加]** を選択して、Sybase Central でサーバの作成ウィザードを起動することもできます。

次の手順は、マルチプレックス・サーバを手動で作成する場合に必要です。サーバの作成ウィザードを使用する場合、これらの手順は Sybase Central によって自動的に行われます。

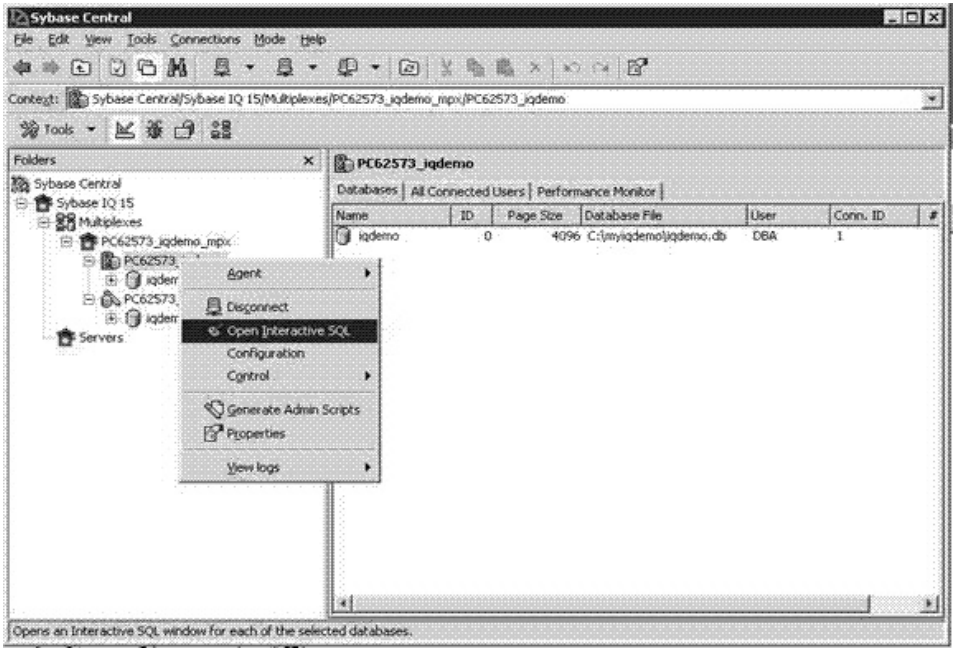
マルチプレックス・サーバ定義の作成

データベース・システム・テーブル SYSIQMPXSERVER でマルチプレックス・サーバを定義します。

❖ **マルチプレックス・サーバの手動による定義**

1 Sybase Central のフォルダ・ビューから、Interactive SQL を起動します。

図 17 : Interactive SQL の起動



2 Interactive SQL で、以下の情報を含んだ次のコマンドを実行します。

- DATABASE
  - PC62573_iqdemo_r2 サーバのファイルを含むローカル・ディレクトリ内のフル *db* ファイル・パス。
  - 正しいパスを指定してください。新しいマルチプレックス・サーバを同期するまで、間違ったパスは検出されません。
- HOST – ローカル・ホストの名前。
- PORT – ローカル・ホスト上のユニークなポート番号。このデモの規則に従って、ポート 2640 を使用します。

  UNIX の場合

```
CREATE MULTIPLEX SERVER SUN62574_iqdemo_r2
DATABASE '/myiqdemo/r2/iqdemo.db' ROLE READER
HOST 'SUN62574' PORT 2640
```

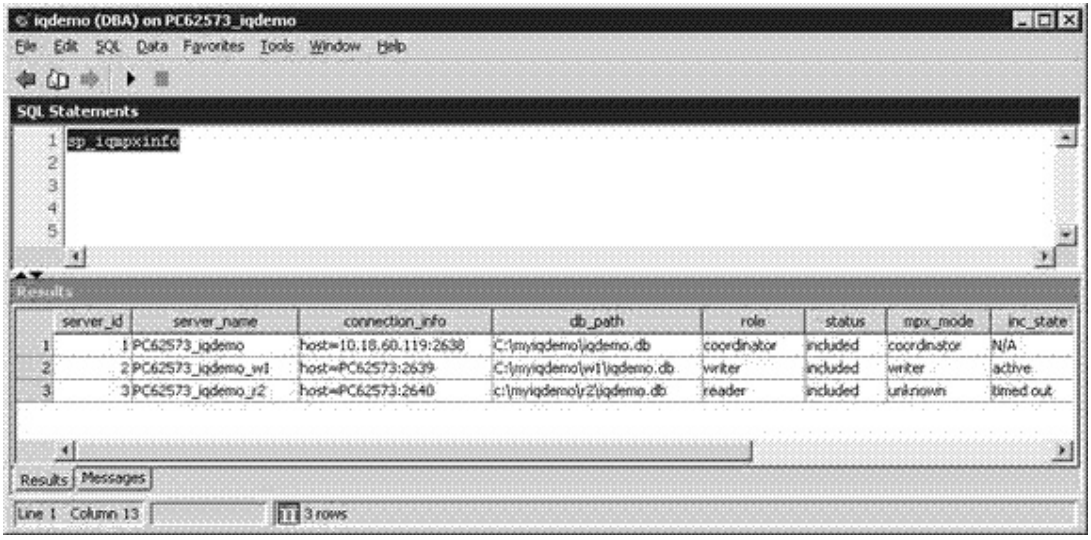
Windows の場合

```
CREATE MULTIPLEX SERVER PC62573_iqdemo_r2
DATABASE 'C:¥myiqdemo¥r2¥iqdemo.db' ROLE READER
HOST 'PC62573' PORT 2640
```

3 [F5] キーを押してコマンドを実行します。

4 すべてのサーバの新しいマルチプレックス設定を表示するには、sp_iqmpxinfo ストアド・プロシージャを実行します。次の出力が生成されます。

図 18 : iqdemo (DBA)



マルチプレックス・サーバの同期と起動 (手動による方法)

コーディネータのカタログ・ストアから *db* ファイルを生成し、新しく作成したリーダのディレクトリにコピーしてから、新しいサーバを起動します。

または、マルチプレックス・サーバ・アイコンを右クリックし、**[コントロール] - [同期]** を選択して、Sybase Central でこの操作を行うこともできます。

❖ **マルチプレックス・サーバを手動で起動するには、次の手順に従います。**

1 まだ作成されていない場合は、マルチプレックス・サーバ・ディレクトリを作成します。このデモの規則に従い、マルチプレックス・ディレクトリは次の場所にあります。

- **UNIX の場合** – */myiqdemo/r2*
- **Windows の場合** – *C:¥myiqdemo¥r2*

2 新しく作成したマルチプレックス・サーバ・ディレクトリに変更します。

- **UNIX の場合** – *cd /myiqdemo/r2*
- **Windows の場合** – *cd C:¥myiqdemo¥r2*

3 存在しない場合は、コーディネータのディレクトリからリーダのディレクトリに *params.cfg* をコピーします。

4 以前のトランザクション・ログ・ファイル (*mpx.log* など) がディレクトリに存在する場合は、削除します。同期後にマルチプレックス・サーバ・ディレクトリに古いトランザクション・ログを残しておくと、サーバが起動しない場合があります。

5 コマンド・ラインから、次のコマンドを実行し、新しいマルチプレックス・サーバに適したパス名、ホスト名、およびサーバ名に置き換えます。以下の各コマンドを 1 行に入力します。

- **UNIX の場合**

```
dbbackup -y -d -c
'uid=DBA;pwd=sql;eng=SUN62574_iqdemo;links="tcp
ip{host=SUN62574;port=2638}"' .

rm ?rf "iqdemo.log"

dblog -r -t "iqdemo.log" "iqdemo.db"

start_iq @params.cfg
-n SUN62574_iqdemo_r2
-x "tcpip{host=SUN62574;port=2640}" iqdemo.db
```

- **Windows の場合**

```
dbbackup -y -d -c
"uid=DBA;pwd=sql;eng=PC62573_iqdemo;links=tcpip
{host=PC62573;port=2638}" .

erase /F /Q "iqdemo.log"
```

```
dblog -r -t "iqdemo.log" "iqdemo.db"


start_iq @params.cfg
-n PC62573_iqdemo_r2
-x "tcpip{host=PC62573;port=2640}" iqdemo.db
```

通常、dbbackup に ODBC DSN を使用し、設定ファイルを編集し、リンク情報とサーバ名を設定ファイルに追加して、サーバを起動します。

注意 手順 3 と手順 5 の間の遅延が 60 分を超えると、サーバの起動時に問題が発生します。これは、MPX_AUTOEXCLUDE_TIMEOUT のデフォルト値が 60 分であるためです。

問題を修正するには、コーディネータに接続し、`ALTER MULTIPLEX SERVER <new-server> STATUS INCLUDED` を使用して新しいサーバを含めます。その後、再同期し (dbbackup および dblog を実行し)、サーバを再起動します。

6 サーバが実行されていることを確認するには、Interactive SQL を使用してサーバに接続します。

テンポラリ・ストア・ファイルの追加 (手動による方法)

セカンダリ・ノードのテンポラリ・ストア・ファイルを追加するまで、IQ データのクエリは行えません。テンポラリ・ストレージがさらに必要な場合は、必要に応じてこの手順を繰り返し、テンポラリ・ストア・ファイルを追加します。

❖ **テンポラリ・ストア・ファイルの追加**

この手順を実行するには、コーディネータ・ノードとセカンダリ・ノードの両方が実行されている必要があります。

1 Interactive SQL を使用して、リーダに接続します。

2 次のコマンドを実行し、適切なファイル・パスに置き換えます。テンポラリ・ストア・ファイル・パスは、各サーバまたはホストに一意なパスである必要があります。パスは、各マルチプレックス・サーバ用に個別の物理ファイルを解決する限りは、絶対パスでも相対パスでもかまいません。

- **UNIX の場合**

```
ALTER DBSPACE IQ_SYSTEM_TEMP ADD FILE
iqdemo_r2_temp 'iqdemo_r2_temp.iqtmp'
SIZE 300 RESERVE 300
```

- **Windows の場合**

```
ALTER DBSPACE IQ_SYSTEM_TEMP ADD FILE
iqdemo_r2_temp 'iqdemo_r2_temp.iqtmp'
SIZE 300 RESERVE 300
```

sp_iqdbspace‘IQ_SYSTEM_TEMP’を実行して、リーダに空きテンポラリ領域が確保されていることを確認します。

データのロード

この項では、コーディネータとライタを使用して、データをマルチプレックス・データベースにロードします。ワークロードが複数のサーバに分配されるため、これはマルチプレックス・リソースの最も効果的な使用方法です。このデモでは、異なるサーバを連続で使用してデータをロードします。実際の運用では、複数のサーバから個別のテーブル上のロードを並行して実行できます。

このデモでは、IQ 15.3 に同梱のロード・ファイルを用いて、Contacts および SalesOrderItems テーブルを使用します。

UNIX では、これらのファイルは *$IQDIR15/demo/adata* にあります。

Windows では、これらのファイルは *%ALLUSERSPROFILE%¥SybaseIQ¥demo¥adata* にあります。

以下の LOAD 文では、これらの入力ファイルへの適切なパスに置き換えます。

❖ コーディネータからのデータのロード

1 Interactive SQL を使用して、コーディネータ・サーバ PC62573_iqdemo に接続します。

2 Contacts テーブルでは、demo データベースの作成中に同じデータがロードされるため、このテーブルをトランケートします。

3 Contacts テーブルをロードします (ファイル・パスを次に示す方法でなく、1 行に入力します)。

UNIX の場合

```
TRUNCATE TABLE Contacts;

LOAD TABLE Contacts (ID, Surname,GivenName,
Title, Street, City, State, Country, PostalCode,
Phone, Fax, CustomerID)
USING FILE '/sun62574/users/userid/
/syb_install_dir/IQ-15_3/demo/adata/contact.dat'
ROW DELIMITED BY '|'
ESCAPES OFF;

COMMIT;
```

Windows の場合

```
TRUNCATE TABLE Contacts;

LOAD TABLE Contacts (ID, Surname,GivenName,
Title, Street, City, State, Country,
PostalCode, Phone, Fax, CustomerID)
USING FILE
'C:¥Documents and Settings¥All Users¥
SybaseIQ¥demo¥adata¥contact.dat'
ROW DELIMITED BY '|'
ESCAPES OFF;

COMMIT;
```

❖ **ライタからのデータのロード**

1 Interactive SQL を使用して、ライタ・サーバ PC62573_iqdemo_w1 に接続します。

2 SalesOrderItems テーブルでは、demo データベースの作成中に同じデータがロードされるため、このテーブルをトランケートします。

3 SalesOrderItems テーブルをロードします (ファイル・パスを次に示す方法でなく、1 行に入力します)。

UNIX の場合

```
TRUNCATE TABLE SalesOrderItems;

LOAD TABLE SalesOrderItems (ID, LineID,
ProductID, Quantity, ShipDate)
USING FILE '/sun62574/users/userid/
/syb_install_dir/IQ-15_3/demo
```

```
/adata/sales_oi.dat'
ROW DELIMITED BY '|'
ESCAPES OFF;

COMMIT;
```

Windows の場合

```
TRUNCATE TABLE SalesOrderItems;

LOAD TABLE SalesOrderItems (ID, LineID,
ProductID, Quantity, ShipDate)
USING FILE 'C:¥Documents and Settings¥All Users¥
SybaseIQ¥demo¥adata¥sales_oi.dat'
ROW DELIMITED BY '|'
ESCAPES OFF;

COMMIT;
```

データがデータベースにロードされたら、マルチプレックス内の任意のサーバからデータに対してクエリを実行できます。このベース・マルチプレックスを使用して、他にもさまざまな試みを行うことができます。また、iqdemo スクリプトを調整することで、このデータに対して別の操作を実行する方法を調べることもできます。

注意 ソフトウェアのインストールまたは実行に関する質問については、Sybase テクニカル・サポートまたは最寄りの Sybase のサポート・センタまでご連絡ください。

分散クエリ処理

分散クエリ処理 (DQP) とは、1 つのクエリを複数の独立した作業に分割し、その作業をマルチプレックス内の他のノードに分散し、中間結果を収集して編成することによって、クエリの最終的な結果セットを生成することです。Sybase IQ マルチプレックス内のサーバは、共有ディスク・アレイなどの永続的な共有データの場合は中央のストアに接続します。マルチプレックスでは、永続的な IQ データには共有記憶領域を使用し、カタログ・メタデータ、プライベートなテンポラリ・データ、トランザクション・ログには独立したノード記憶領域を使用する、ハイブリッド・クラスタ・アーキテクチャが採用されています。

単純なクエリは、単一のマシンでの実行が最も高速です。ただし、大規模なクエリや複雑なクエリが単一のマシンの CPU 処理能力を超えると、Sybase IQ クエリ・オプティマイザがそのクエリを並列の「フラグメント」に分割し、その分散クエリがマルチプレックス内の他のサーバ上で同時に実行されます。

注意 分散には、作業を割り当て、中間結果を格納して転送するためのネットワークと記憶領域のオーバヘッドが必要です。クエリが単一のマシンの CPU リソースを完全には使用していない場合は、単一のマシンを使用する方が最適な結果が得られます。たとえば、オプティマイザが 8 コアのマシンで 1 つのクエリを 7 分割して並列処理しようとした場合 (つまり、同時に 7 つのスレッドがビジー状態になる場合)、Sybase IQ はそのクエリを分散しません。

DQP を使用する前に、次の操作が必要です。

- マルチプレックスを設定します (「IQ デモ・データベースのマルチプレックスへの変換」(10 ページ) を参照してください)。
- マルチプレックス・サーバ間で共有するテンポラリ・ストレージを設定します (「共有テンポラリ・ストレージの追加」(34 ページ) または 「共有テンポラリ・ストレージの追加 (手動による方法)」(37 ページ) を参照してください)。

「分散されるクエリの実行」(39 ページ) では、クエリをデモ・マルチプレックス内の 2 つのサーバに分散して実行する方法について説明しています。

共有テンポラリ・ストレージの追加

分散クエリ処理 (DQP) では、参加する複数のノードが使用するテンポラリ領域が必要です。この方法では、Sybase Central GUI を使用してテンポラリ・ストレージを追加します。

❖ **共有テンポラリ・ストレージの追加**

新規のデータベースと新たに移行されたデータベースの DB 領域 IQ_SHARED_TEMP は空です。DQP を有効にするには、この DB 領域にファイルを追加し、そのファイルをすべてのマルチプレックス・ノードから読み書きアクセス可能にします。

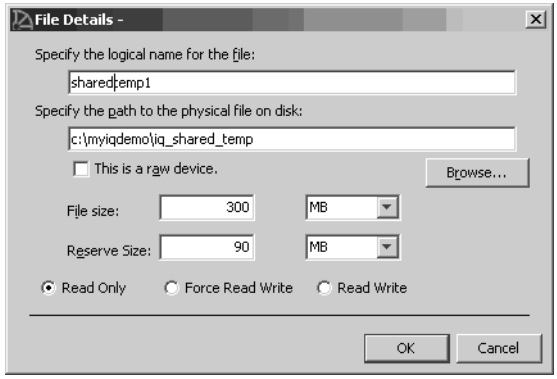
すべてのマルチプレックス・サーバが実行されている必要があります。

1 Sybase Central からコーディネータに接続します。

2 フォルダ・ビューで、[DB 領域] フォルダをダブルクリックします。[DB 領域] フォルダで、[IQ_SHARED_TEMP] をダブルクリックします。[IQ_SHARED_TEMP] を右クリックし、[新規] - [ファイル] を選択します。

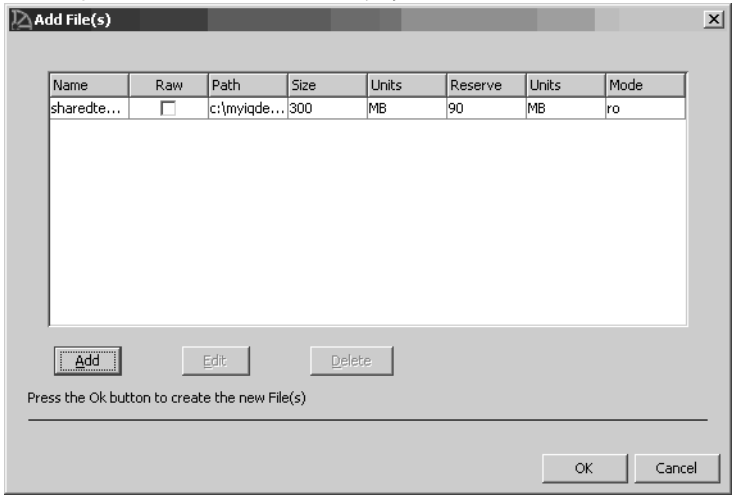
3 [追加] をクリックします。

図 19：共有テンポラリ・ファイルの詳細



4 新しいファイルの論理名、絶対パス、ファイルのサイズ、予約サイズを入力し、[OK] をクリックします。

図 20：共有テンポラリ・ファイルの要約

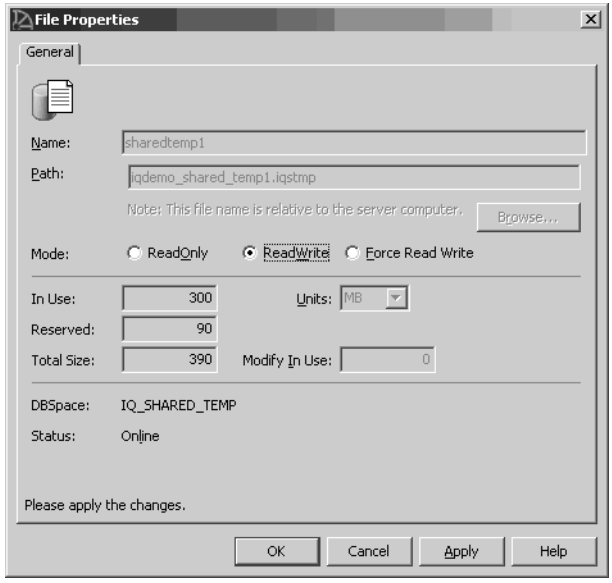


5 選択内容を確認し、[OK] をクリックします。

6 新しいファイル情報がすべてのマルチプレックス・ノードに送信されるまで待機します。これには数分かかる場合があります。

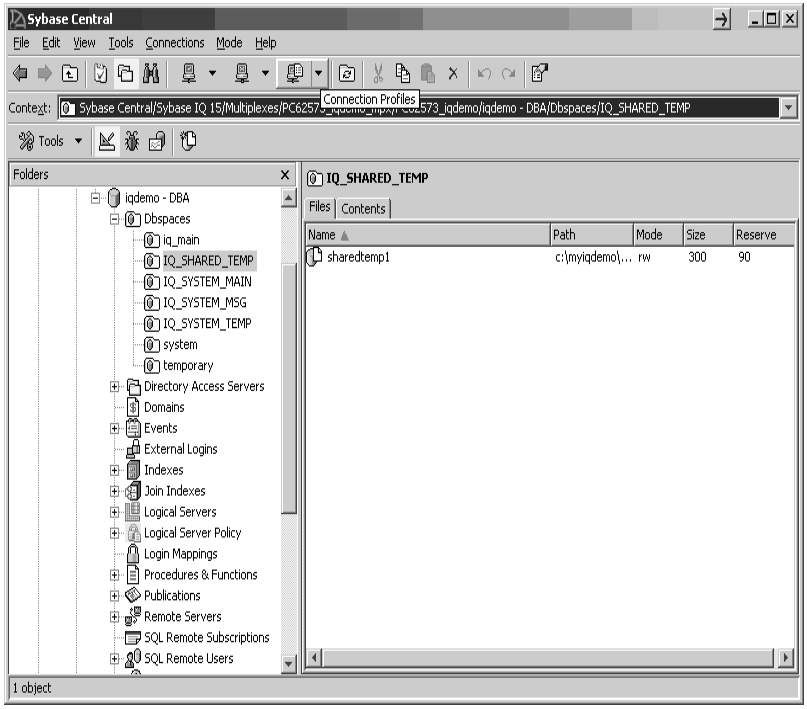
7 [IQ_SHARED_TEMP] を右クリックして、[プロパティ] を選択します。

図 21：共有テンポラリ・ファイルのプロパティ



8 [読み書き] モードを選択して、[OK] をクリックします。

図 22：共有テンポラリ・ファイルのモード



図のように、[ファイル] タブの [モード] (3 列目) に [rw] が表示されていることを確認します。

共有テンポラリ・ストレージの追加 (手動による方法)

分散クエリ処理 (DQP) では、参加する複数のノードが使用するテンポラリ領域が必要です。この方法では、Interactive SQL を使用してテンポラリ領域を追加します。

❖ **共有テンポラリ・ストレージの追加 (手動による方法)**

新規のデータベースまたは新たに移行されたデータベースでは、空の DB 領域 IQ_SHARED_TEMP が作成されます。分散クエリ処理を有効にするには、この DB 領域にファイルを追加し、そのファイルをマルチプレックス内のすべてのノードから読み書きアクセス可能にします。

1 Interactive SQL を起動して、コーディネータに接続します。

2 ALTER DBSPACE ADD FILE コマンドを実行します。

UNIX の場合

```
ALTER DBSPACE IQ_SHARED_TEMP ADD FILE
iqsharedtemp2 '/myiqdemo/iq_shared_temp2.iqstmp'
```

Windows の場合

```
ALTER DBSPACE IQ_SHARED_TEMP ADD FILE iqsharedtemp2
'c:¥myiqdemo¥iq_shared_temp2.iqstmp'
```

3 新しいファイル情報がすべてのマルチプレックス・ノードに送信されるまで待機します。これには数分かかる場合があります。

4 新規ファイルを読み書きモードに変更します。

```
ALTER DBSPACE IQ_SHARED_TEMP ALTER FILE
iqsharedtemp2 READWRITE
```

IQ_SHARED_TEMP が設定され、マルチプレックスで分散クエリを実行できるようになります。

iq_main でのファイル・サイズの増加

分散クエリを実行する前に、ユーザ DB 領域 iq_main でファイル・サイズを増やします。

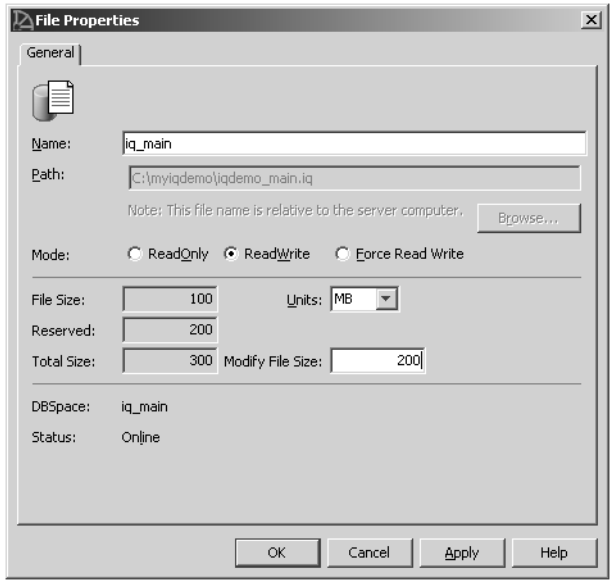
❖ **iq_main でのファイル・サイズの増加**

1 フォルダ・ビューで、[DB 領域] フォルダをダブルクリックします。次に、[iq_main] フォルダを選択します。

2 [ファイル] タブで、ファイル *[iq_main]* を右クリックし、[プロパティ] を選択します。

3 [プロパティ] ダイアログの [ファイル・サイズの変更] に 200 と入力し、[OK] をクリックします。これで、iq_main 領域が 200MB に増加します。

図 23：ユーザ DB 領域 iq_main のファイル・プロパティ



分散されるクエリの実行

DQP では、リーダ・ノードとワーカ・ノードを導入します。リーダ・ノードとは、クエリの発生元のノードです。ワーカ・ノードは、分散クエリ処理の作業を受け入れることができるマルチプレックス内の任意のノードです。

マルチプレックス・ノードのすべてのタイプ (リーダ、ライタ、またはコーディネータ) が、リーダ・ノードまたはワーカ・ノードになることができます。クライアントがサーバにクエリを送信すると、クエリ・オプティマイザがコスト分析を使用して、並列または分散クエリを実行するかどうかを判断します。並列クエリは、述部とデータ・フロー・サブツリーのクエリ・フラグメントに分割されます。クエリ・フラグメントを分散できるのは、Sybase IQ エンジンがフラグメントに含まれているクエリ演算の並列実行と分散実行をサポートしている場合のみです。

クエリが分散されると、リーダ・ノードがクエリ・フラグメントをワーカ・ノードに割り当てます。ワーカ・ノードは、クエリ分散についての決定を行いません。ワーカ・ノードは、割り当てられた作業を実行し、ワーカ・サーバから中間結果を返すだけです。クエリで使用されている行数が多い場合、そのクエリは分散される可能性があります。たとえば、スター・スキーマ上のクエリでは、分散クエリ処理が非常に役立つ可能性があります。

クエリ・オプティマイザが、分散クエリを実行してもサイズが適切に調整されない可能性がある、またはパフォーマンスが低下する可能性さえあると判断した場合は、クエリは分散されずに、マルチプレックス内の単一のノード上で実行されます。

❖ 分散クエリの実行

分散クエリを実行する前に、ユーザ DB 領域 iq_main でファイル・サイズを増やします。

1 Interactive SQL を開いて、次のコマンドを実行します。

UNIX の場合 *–$SYBASE/IQ-15_3/demo/dqpdata.sql*

Windows の場合 –
%ALLUSERSPROFILE%¥SybaseIQ¥demo¥dqpdata.sql

これで、Dimension テーブルと Fact テーブル (1000 万行) が作成されます。

2 Interactive SQL を開いて、次のコマンドを実行します。

UNIX の場合 *–$SYBASE/IQ-15_3/demo/dqpquery.sql*

Windows の場合 –
%ALLUSERSPROFILE%¥SybaseIQ¥demo¥dqpquery.sql

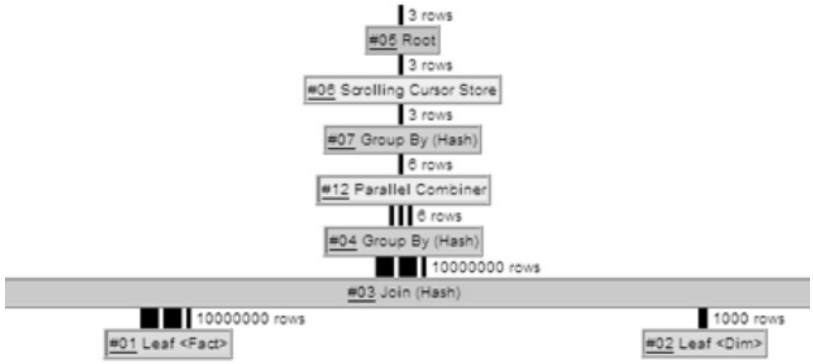
このクエリは、3 行をフェッチし、クエリ・プランを HTML ファイルとして生成して、デモ・ディレクトリとライト・サーバのディレクトリに格納します。また、このクエリは、*.iqmsg* ファイルにパスも書き込みます。

注意 3 行が戻ったら、Interactive SQL から ROLLBACK コマンドを実行して、Sybase IQ を使用してクエリ・プランをフラッシュします。

---

3 マルチプレックス・ノードのディレクトリでクエリ・プランの HTML ファイルを確認します。リーダ・ノードのディレクトリには、包括的なクエリ・プランがあります。また、分散された作業単位ごとに、その作業単位を受け取ったノードによって生成された HTML ファイルがあります。リーダ・ノードのクエリ・プランをダブルクリックし、そのクエリ・プランを Web ブラウザに開きます。

図 24：分散クエリのクエリ・ツリー



グラフィック内でクエリ演算を接続する 3 本の縦線は分散を示しています。たとえば、[Group By] ノードと [Join] との間の 3 本の線は、この演算が並列スレッドで実行され、複数のサーバに分散されることを示しています。[Join] と [#01 Leaf] の間の演算も分散されます。

Windows での ODBC データ・ソースの設定

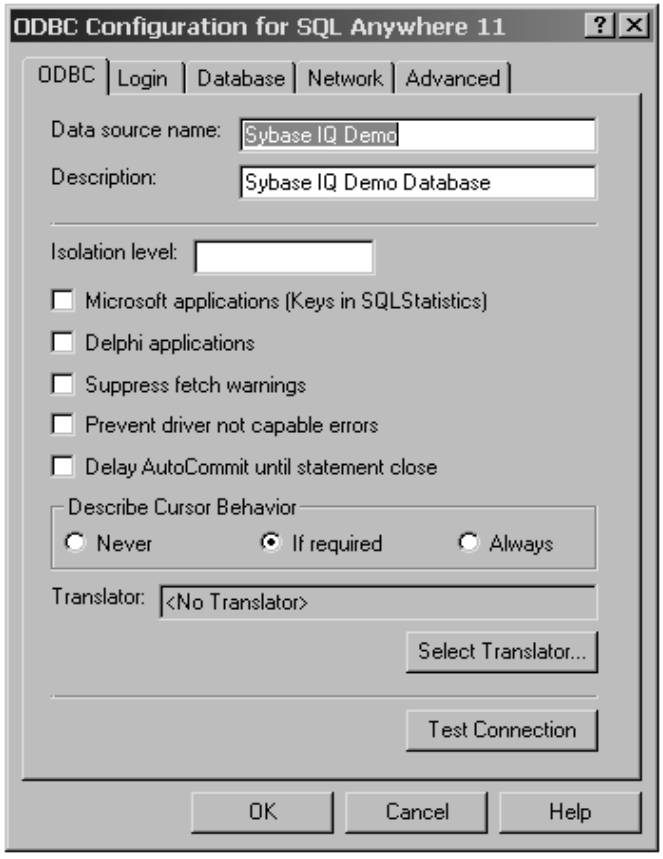
ODBC を使用することで、demo データベースでクライアントを使用できます。

❖ Windows での ODBC データ・ソースの設定

次の例は、ODBC データ・ソース名 (DNS) を設定する方法を示します。

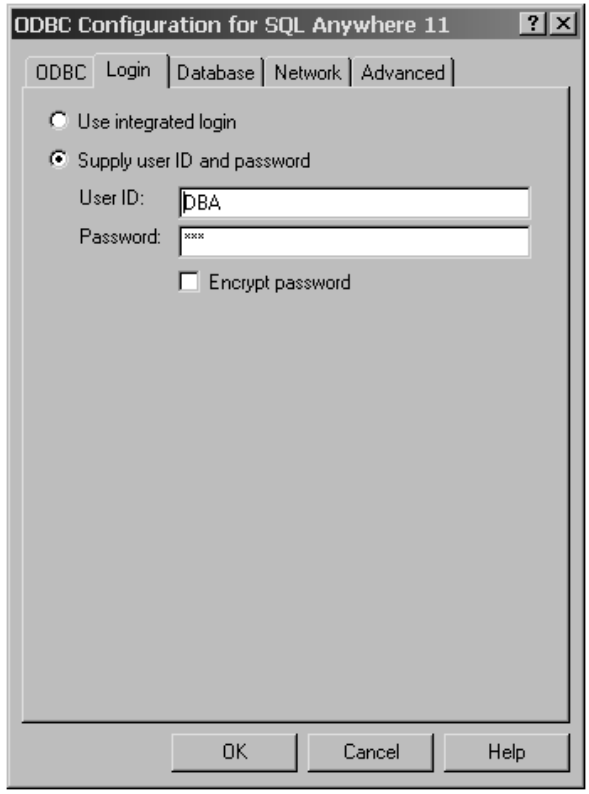
1 [ODBC] タブで、ODBC 管理者がアクセスできる ODBC データ・ソースの名前を指定します。

図 25 : [ODBC] タブ



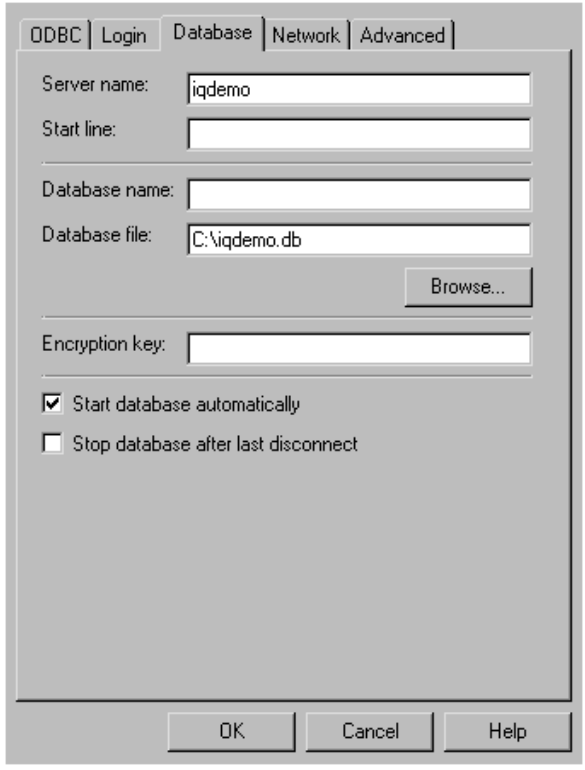
2 [ログイン] タブで、IQ へのこの接続に関連付けるユーザ ID とパスワードを指定します。IQ demo データベースを使用している場合は、ユーザ ID として **DBA**、パスワードとして **sql** をそれぞれ指定します。

図 26 : [ログイン] タブ



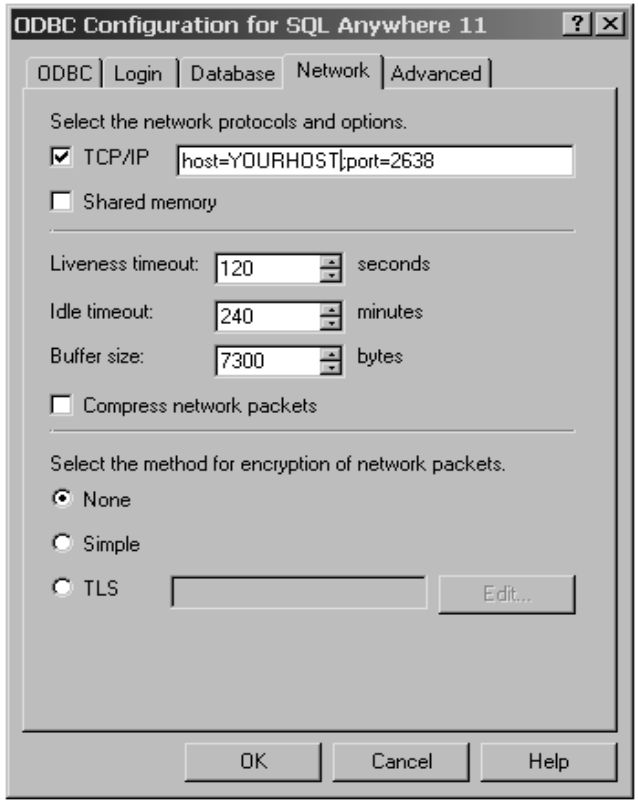
3 [データベース] タブで、サーバの名前とデータベース・ファイルの名前を指定します。-n パラメータ付きで IQ を起動すると、このサーバ名が割り当てられます。データベース・ファイルは、IQ へのこの接続が関連付けられているデータベース・ファイル名の完全修飾パスです。

図 27 : [データベース] タブ



4 [ネットワーク] タブで、ネットワーク・プロトコルとネットワーク・オプションを指定します。必要な TCP/IP オプションは、ホスト (<*yourhost*>) とポート (2638) です。IQ サーバは起動時にポート・オプションを割り当てます。ホストは、IQ サーバが実行されている物理ホストです。

図 28 : [ネットワーク] タブ



ODBC C プログラミングの例

この例は、クライアントを Sybase IQ データベースに接続する ODBC の C 実装を示します。この例は IQ インストールの samples ディレクトリに収められています。

```
//
*************************************************


// Copyright (c) 2001-2009 iAnywhere Solutions, Inc.

// Portions copyright (c) 2001-2009 Sybase, Inc.


// All rights reserved.All unpublished rights reserved.



//
*************************************************


/*
*************************************************


This sample code is provided AS IS, without warranty or
liability of any kind.You may use, reproduce, modify
and distribute this sample code without limitation, on
the condition that you retain the foregoing copyright
notice and disclaimer as to the original iAnywhere code.



********************************************** */



/* odbcconnect.cpp :Defines the entry point for the
console application.*/



    retcode = SQLAllocHandle( SQL_HANDLE_ENV,
SQL_NULL_HANDLE, &env );


    if (retcode == SQL_SUCCESS || retcode ==
SQL_SUCCESS_WITH_INFO) {
```

```
    printf( "env allocated\n" );

    /* Set the ODBC version environment attribute */

    retcode = SQLSetEnvAttr( env,
SQL_ATTR_ODBC_VERSION, (void*)SQL_OV_ODBC3, 0);

    retcode = SQLAllocHandle( SQL_HANDLE_DBC, env, &dbc
);

    if (retcode == SQL_SUCCESS || retcode ==
SQL_SUCCESS_WITH_INFO) {

        printf( "dbc allocated\n" );

        retcode = SQLConnect( dbc,

            (SQLCHAR*) "Sybase IQ Demo", SQL_NTS,

            (SQLCHAR* ) "DBA", SQL_NTS,

            (SQLCHAR*) "sql", SQL_NTS );

        if (retcode == SQL_SUCCESS || retcode ==
SQL_SUCCESS_WITH_INFO) {

       printf( "Connection successful\n" );

        }
        SQLDisconnect( dbc );
    }
    SQLFreeHandle( SQL_HANDLE_DBC, dbc );

     }

     SQLFreeHandle( SQL_HANDLE_ENV, env );
```

```
    return 0;

}
```