SYBASE®
An SAP® Company

Job Scheduler ユーザーズ・ガイド

# Adaptive Server® Enterprise

15.7

ドキュメント ID：DC20135-01-1570-01
改訂：2011 年 8 月



Sybase, Inc., One Sybase Drive, Dublin, CA 94568.

# 目次

第 1 章

# 概要

この章では、Job Scheduler のコンポーネントとアーキテクチャの概要について説明します。



## 概要

Job Scheduler では、データベース管理タスクの定義とスケジューリングができ、ASE の管理が容易になります。Job Scheduler を使用すれば、通常ならデータベース管理者による対話型の操作が必要となるジョブをスケジューリングして、適切な時間に自動的に実行させることができます。これより、データベース管理者は他の業務に時間を使うことができます。

Job Scheduler では、ジョブを作成してスケジューリングしたり、ジョブやスケジュールを共有したりできます。あるデータベース管理者が作成したジョブを、他のデータベース管理者が別のサーバでスケジューリングして実行できます。以下の方法でジョブを作成できます。

- コマンド・ラインまたは GUI を使用してゼロから作成する
- SQL バッチ・ファイルから作成する
- テンプレートから作成する

Job Scheduler はジョブの結果と出力を取得して、その情報をログ・テーブルに記録します。このデータは後から参照できます。さらに、Job Scheduler はスケジュール・ジョブの履歴を保持します。ただし、履歴テーブルのサイズ制限により、Job Scheduler は古くなった不要な履歴レコードを自ら監視して削除します。

---

**注意** Job Scheduler を効果的に利用するには、データベース管理に関する知識が必要です。Job Scheduler は、通常はデータベース管理者が行う操作を実行します。

---

# 用語と概念

この項では、Job Scheduler の使用に関連する基本概念と用語について説明します。

**用語**

- ジョブとは、*1* 回の操作でデータベースに対して実行される一連のアクション (バックアップ、統計情報の更新、データベースのダンプなど) です。
- スケジュールとは、ジョブを実行および再実行する方法と時刻を定義したものです。
- スケジュール・ジョブとは、スケジュールにバインドされているジョブです。スケジュール・ジョブのみが実行されます。
- *Job Scheduler Task (JS Task)* とは、スケジュールを管理して、特定のジョブを実行するよう適時に Job Scheduler Agent に通知する機能コンポーネントです。
- *Job Scheduler Agent (JS Agent)* とは、JS Task からの通知を受けてジョブを実行する機能コンポーネントです。
- 繰り返しスケジュールとは、複数回アクティブになるスケジュールです。すべての繰り返しスケジュールには、開始時刻と終了時刻が必要です。
- ターゲット・サーバとは、ジョブの実行がスケジューリングされている Adaptive Server® です。
- テンプレートとは、Job Scheduler 内でジョブの作成に使用できるパラメータ付き Transact-SQL (T-SQL) 文のセットです。

**概念**

JS Task は、実行がスケジューリングされたジョブによってウェイクアップするか、新規ジョブの作成などのウェイクアップ・イベントが発生するとウェイクアップします。JS Task は一定の間隔 ( データベース管理者が設定可能 ) で *sybmgmtdb* データベース・テーブルをスキャンし、該当するスケジュール情報を収集して専用のサーバ・テーブル js_callouts および js_history を管理します。JS Task は、Job Scheduler が内部タスクを実行できるように関数を実行します。ASE 内で Job Scheduler ランタイムをプログラム可能にするのも JS Task です。

# Job Scheduler のコンポーネントと機能

Job Scheduler は、次のコンポーネントで構成されています。

- JS Task と呼ばれる内部 ASE タスク
- JS Agent と呼ばれる外部プロセス
- sybmgmtdb データベースとストアド・プロシージャ

- Sybase® Central の ASE プラグインを使用したグラフィカル・ユーザ・インタフェース (GUI)
- 時間を節約する有効なジョブをデータベース管理者が作成するときの基礎になる、定義済みテンプレート

**基本タスク**

Job Scheduler の基本機能を構成するタスクは次のとおりです。これらのタスクは、コマンド・ラインまたは Sybase Central グラフィカル管理ツールで実行できます。

- ジョブ、スケジュール、スケジュール・ジョブの作成
- ジョブ、スケジュール、スケジュール・ジョブの修正
- ジョブ、スケジュール、スケジュール・ジョブの削除
- ジョブ履歴の確認

## Job Scheduler のアーキテクチャ

JS Task は、スケジュール・ジョブを実行するタイミングを判断し、実行したジョブの履歴レコードを作成します。また、JS Agent プロセスを起動し、ジョブ情報を取得して指定の ASE でジョブを実行するために必要な情報を JS Agent に提供します。

JS Agent は、sybmgmtdb という Job Scheduler 専用のデータベースからジョブ情報を取得すると、ターゲット ASE にログインしてジョブ・コマンドを発行します。ジョブが完了すると、JS Agent はすべての結果または出力を sybmgmtdb データベースのログ・テーブルに記録します。

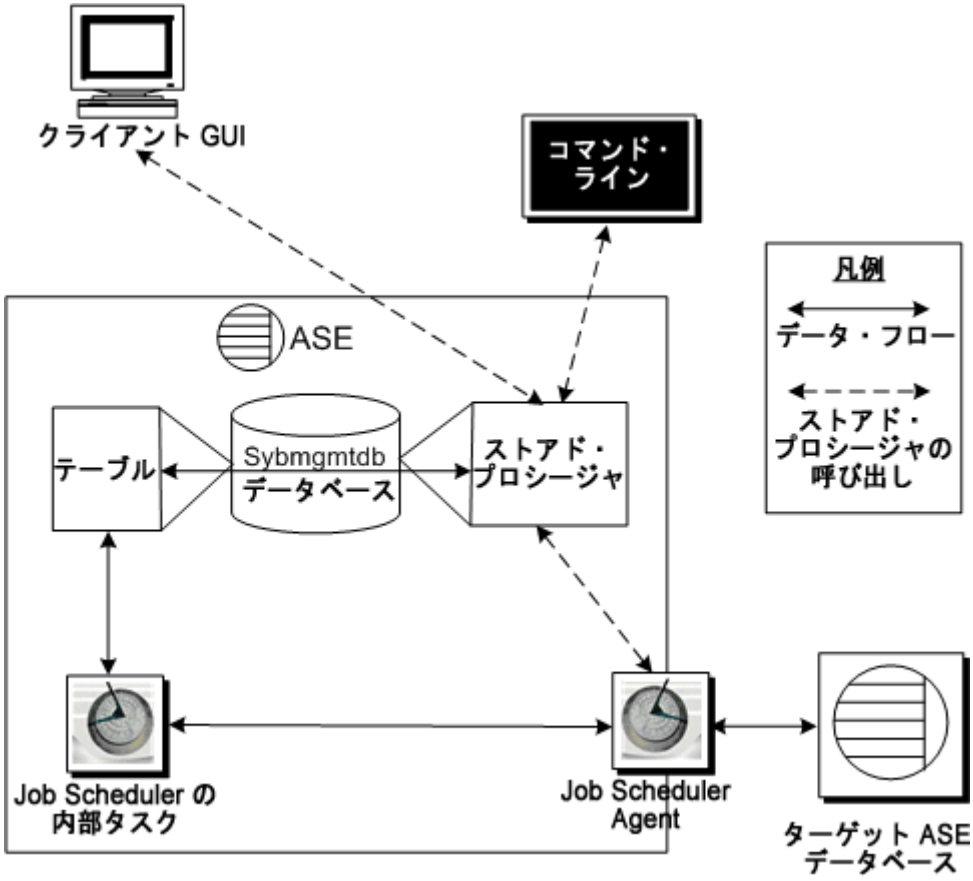
ジョブ、スケジュール、スケジュール・ジョブに関する情報や、JS Task が内部処理のために必要とするデータはすべて、sybmgmtdb データベースに格納されます。通常、sybmgmtdb データベースのデータには、ストアド・プロシージャを通じてアクセスします。ストアド・プロシージャを使用することで、GUI、JS Agent、コマンド・ライン・インタフェースからデータを利用できます。sybmgmtdb データベースのデータに直接アクセスするのは JS Task だけです。

JS Task が取得したデータを使用するときに、GUI は、ジョブの作成とスケジューリング、ジョブ・ステータスとジョブ履歴の参照、ジョブの制御を支援します。ASE 内部タスクのオン／オフを切り替える管理機能も備えているため、Job Scheduler によるスケジュール・ジョブの処理と実行が可能になります。

テンプレートは、データベースのバックアップ、再編成、再構築、設定パラメータの変更、統計情報の更新とモニタリングなど、データベースの自己管理のためのパラメータ化されたタスクを定義する重要なツールです。テンプレートは、パラメータ値を指定できるバッチ T-SQL コマンドとして実装されています。データベース管理者はテンプレートを使用してジョブを作成し、そのジョブを特定の時刻に実行するようスケジューリングできます。

図 1-1 に Job Scheduler のアーキテクチャを構成するコンポーネントを示します。

**図 1-1: Job Scheduler のアーキテクチャ**




**プロセス・フロー**

図に示すとおり、クライアント GUI とコマンド・ライン・インタフェースは、ストアド・プロシージャを通じて Job Scheduler データベース・テーブルと対話します。ストアド・プロシージャは、ユーザから要求されたアクションを実行し、データベース・テーブルでジョブ・コマンド、スケジュール、その他のジョブ情報の定義を管理します。

JS Task は、これらのテーブルからジョブとスケジュールの情報を読み込み、スケジュール・ジョブを実行するタイミングを判断します。JS Task は、実行する必要のあるジョブを適切なタイミングで JS Agent に通知します。通知を受けた JS Agent はジョブ情報を取得し、それをターゲット ASE データベースに対して実行します。ターゲット ASE データベースは、ローカル (Job Scheduler と同じサーバ上) でもリモート (Job Scheduler がインストールされているサーバ上も含め、JS Agent 外部の任意の場所 ) でもどちらでもかまいません。JS Agent は出力と結果セットを収集して、Job Scheduler sybmgmtdb データベースの適切なテーブルに格納します。

# セキュリティ

Job Scheduler では、定義済みの役割である js_user_role と js_admin_role を使用してセキュリティを管理します。

## js_user_role

js_user_role を持つユーザは、Job Scheduler の GUI またはコマンド・ライン・プロシージャを使用して、ジョブ、スケジュール、スケジュール・ジョブを作成、修正、削除、実行するパーミッションを持ちます。ジョブ、スケジュール、またはスケジュール・ジョブの所有者は、オブジェクトを表示したり使用したりできるユーザを制限できますが、別のユーザが所有しているオブジェクトを制御することはできません。

## js_admin_role

js_admin_role を持つユーザには、js_user_role のすべてのパーミッションが割り当てられます。また、ユーザは他のユーザのジョブ、スケジュール、スケジュール・ジョブを表示したり、操作したりできます。さらに、js_admin_role ユーザは以下を変更できます。

- ジョブ、スケジュール、またはスケジュール・ジョブの所有者
- Job Scheduler が使用するデバイス領域
- Job Scheduler によるジョブ・チェックの頻度
- ジョブで書き込むことができるログの最大サイズ

ASE での役割の詳細については、『ASE システム管理ガイド 第 1 巻』の第 1 章を参照してください。

## js_sa_role

Job Scheduler が sa アカウントを使用するように設定されている場合、sa パスワードが変更されるたびにこのコマンドを追加する必要があります。

```
sp_addexternlogin loopback, sa, sa, new_password
go
```

## 共有オブジェクト

デフォルトでは、ジョブの実行はスケジュール・ジョブ・テーブル内のエントリの所有者に制限されています。ただし、スケジュール・ジョブの shared_run プロパティを設定することによって、他の Job Scheduler ユーザもそのジョブを利用できるようになります。また、他のユーザに対してジョブやスケジュールを作成するようにこのプロパティを設定することもできます。このため、他のユーザもそれらのジョブやスケジュールを利用して独自のスケジュール・ジョブを作成できます。ジョブ、スケジュール、スケジュール・ジョブを共有しても、他の Job Scheduler ユーザがそれらを修正したり削除したりすることはできません。共有は読み込み専用です。

第 2 章

# Job Scheduler の設定と実行

この章では、Job Scheduler のインストール、設定、実行について説明します。この項は、次の項目で構成されています。



## Job Scheduler の手動インストール

Adaptive Server のインストール時に Job Scheduler のオプションを選択すると、オプションがインストールされ、設定されます。Adaptive Server のインストール時に Job Scheduler のオプションを選択していないと、オプションはインストールされません。

Job Scheduler を手動でインストールする場合は、Adaptive Server のインストール後に次の手順を完了してください。

### ❖ Job Scheduler のインストール

1 sybmgmtdev という名前のデバイスを 75MB 以上のサイズで作成します。

2 次の *installjsdb* スクリプトを実行します。

```
isql -Usa -Psa_password -Sservername -i
$SYBASE/$SYBASE_ASE/scripts/installjsdb
```

**注意** isql 実行プログラムがあるディレクトリ (*$SYBASE/$SYBASE_OCS/bin*) にパスが通るようにします。

*installjsdb* スクリプトは sybmgmtdb データベースを探します。データベースが存在する場合は、Job Scheduler のテーブルとストアド・プロシージャを作成します。存在しない場合は sybmgmtdev デバイスを探して、そこに sybmgmtdb データベース、テーブル、ストアド・プロシージャを作成します。

**注意** sybmgmtdev デバイスも sybmgmtdb データベースも見つからない場合、*installjsdb* スクリプトは master デバイスに sybmgmtdb データベースを作成します。Sybase では、master デバイスから sybmgmtdb データベースを削除することを強くおすすめします。これにより、混乱を避けられるだけでなく、ディスク障害時のリカバリが容易になります。

3 ディレクトリ・サービスの設定やオペレーティング・システムに応じて、dscp、dsedit、またはテキスト・エディタを使用して、JS Agent 用のディレクトリ・サービスのエントリを作成します。Sybase では、"*server name*_JSAGENT" という名前をおすすめします。

```
servername_JSAGENT
    master tcp ether server_machine port_number
    query  tcp ether server_machine port_number
```

各要素の意味は次のとおりです。

- *server_machine* は、Job Scheduler Adaptive Server がインストールされているマシンの名前です。
- *port_number* は、JS Task が JS Agent と通信するポートです。

**注意** 必ず現在使用されていないポートを指定してください。

ディレクトリ・サービスの詳細については、『システム管理ガイド 第 1 巻』を参照してください。

4 sp_addserver を使用して、sysservers テーブルにエントリを作成します。

```
sp_addserver SYB_JSAGENT, null, <servername_JSAGENT>
```

sp_addserver の詳細については、『ASE リファレンス・マニュアル第 2 巻：コマンド』を参照してください。

5 次のように Job Scheduler を有効にします。

```
sp_configure "enable job scheduler", 1
```

6 number of user connections を適切に設定します。Sybase では、number of user connectionsを job scheduler tasks を増やした量の少なくとも 3 倍は増やすことをおすすめします。たとえば、job scheduler tasks の値をデフォルト値 (4) から 8 に増やした場合は、number of user connections の値を 12 は増やしてください ((8 - 4) x 3 = 12)。

**注意** number of user connections を変更すると、Adaptive Server で max memory の値を増やす必要が生じる場合があります。

7　Job Scheduler を起動するには、サーバを再起動するか、次のコマンドを実行します。

```
use sybmgmtdb
go
sp_sjobcontrol @name=NULL, @option="start_js"
go
```

詳細については、「sp_sjobcontrol」(41 ページ) を参照してください。

# Job Scheduler ユーザの設定

ユーザを設定して、sybmgmtdb データベースでジョブやスケジュールの作成、管理、実行をできるようにします。

**注意**　Job Scheduler のユーザを設定する場合、各ユーザにあらかじめ ASE ログインが必要です。

### ❖ Job Scheduler のユーザ設定

1　sybmgmtdb データベースにユーザを追加します。

```
use sybmgmtdb
go
sp_adduser local_login_name
go
```

*local_login_name* は、Job Scheduler サーバのユーザのログイン名です。

2　ユーザに適切な役割を付与します。

Job Scheduler 管理者の場合

```
sp_role 'grant', js_admin_role, local_login_name
go
sp_modifylogin local_login_name, 'add default role',
js_admin_role
go
```

Job Scheduler ユーザの場合

```
sp_role 'grant', js_user_role, local_login_name
go
sp_modifylogin local_login_name, 'add default role',
js_user_role
go
```

# ターゲット・サーバへのアクセスの設定

Job Scheduler のエージェントは Adaptive Server の外部で実行されるため、ターゲット・サーバはエージェントをリモート・ユーザとして扱います。したがって、(Job Scheduler がインストールされているサーバがターゲット・サーバの場合でも) 各ターゲット・サーバを定義し、Job Scheduler がインストールされている「ローカル・サーバ」にログインします。

### ❖ ターゲット・サーバへのアクセスの設定

1 ディレクトリ・サービスの設定に応じて、dscp、dsedit、またはテキスト・エディタを使用して、各ターゲット・サーバ用のディレクトリ・サービスのエントリを作成します。

```
target_servername
   master tcp ether targetserver_machine port_number
   query tcp  ether targetserver_machine port_number
```

各要素の意味は次のとおりです。

- *target_servername* は、ジョブをスケジューリングして実行するサーバの名前です。
- *targetserver_machine* は、ジョブをスケジューリングして実行するマシンの名前です。
- *port_number* は、JS Agent がターゲット・サーバに接続するポートです。

ディレクトリ・サービスの詳細については、『システム管理ガイド 第 1 巻』を参照してください。

2 Job Scheduler が実行されている Adaptive Server で、ジョブを実行する各ターゲット・サーバのエントリを sysservers テーブルに作成します。ターゲット・サーバは、リモートでも Job Scheduler サーバ自体でもかまいません。いずれの場合も、sysservers テーブルにターゲット・サーバのリモート・エントリを追加してください。

- **リモート・ターゲット・サーバの場合** リモート・ターゲット・サーバを追加するには、sp_addserver を使用します。追加するサーバ・クラスは、ASEnterprise にしてください。

```
sp_addserver target_servername, ASEnterprise,
directoryservices_name
go
```

各要素の意味は次のとおりです。

- *target_servername* は、ローカル・サーバのエイリアスです。Adaptive Server で Job Scheduler をインストールすると、Job Scheduler を実行しているサーバのエイリアスとして “loopback” が使用されます。

- *directoryservices_name* は、ディレクトリ・サービス・ファイル内で使用するターゲット・サーバの名前です。

**注意** リモート・ターゲット・サーバは、Job Scheduler がインストールされているサーバでもかまいません。

- **ローカル・ターゲット・サーバ (Job Scheduler を実行しているサーバ) の場合：** sysservers テーブルには、Job Scheduler がインストールされているサーバへのローカル参照がすでに含まれています。ただし、このサーバをターゲット・サーバにするには、サーバのリモート・エントリを sysservers テーブルに作成します。

```
sp_addserver target_servername, ASEnterprise,
directoryservices_name
go
```

各要素の意味は次のとおりです。

- *target_servername* は、ジョブをスケジューリングして実行するサーバです。この場合は、ローカル (Job Scheduler) サーバです。
- *directoryservices_name* は、ディレクトリ・サービス・ファイル内で使用するこのターゲット・サーバの名前です。

3 sp_addexternlogin を使用して、Job Scheduler ユーザのログインを追加します。

```
sp_addexternlogin target_servername, localname,
remotename, remotepwd
go
```

各要素の意味は次のとおりです。

- *target_servername* は、ジョブをスケジューリングして実行するサーバです。
- *localname* は、Job Scheduler サーバのユーザのログイン名です。
- *remotename* は、ターゲット・サーバのユーザのログイン名です。
- *remotepwd* は、ターゲット・サーバのユーザのパスワードです。

**注意** Job Scheduler がインストールされているサーバがターゲット・サーバである場合も含め、すべてのターゲット・サーバのすべてのユーザの外部ログインを追加します。

これで、コマンド・ラインからストアド・プロシージャを使用するか、Sybase Central の ASE プラグインを使用して、ジョブ、スケジュール、スケジュール・ジョブを作成できるようになりました。Job Scheduler のストアド・プロシージャの詳細については、「第 5 章 コマンド・リファレンス」を参照してください。Sybase Central での Job Scheduler の使用については、「第 6 章 Sybase Central の ASE プラグインでの Job Scheduler の管理」を参照してください。

第 3 章

# テンプレートを使用したジョブの作成

この章では、Job Scheduler でのテンプレートのインストールと使用方法について説明します。



## 概要

テンプレートは、Adaptive Server データベースの管理を一段と容易にします。テンプレートを使用すると、データベース管理者は特定のデータベース・チューニング・タスクや管理タスクを実行する T-SQL スクリプトを効率的に作成できます。作成した T-SQL は、ジョブ自体の T-SQL テキストになります。これらのジョブは、後から特定のサーバ向けにカスタマイズしたり、指定した時刻に任意のネットワーク・サーバ上で実行するようスケジューリングしたりできます。このように、テンプレートを利用することで、データベース管理者は独自の T-SQL や cron スクリプトを作成する作業が軽減され、他の業務に集中できます。

Sybase Central の ASE プラグインを使用すると、ウィザードの指示に従いながら、データベース管理用のプロシージャを呼び出す T-SQL を作成する各手順を実行できます。

---

**注意** ストアド・プロシージャを使用するテンプレートからジョブを作成する場合は、ジョブを実行するサーバにそのストアド・プロシージャが必要です。詳細については、「ターゲット・サーバへのストアド・プロシージャのインストール」(14 ページ) を参照してください。

---

# ターゲット・サーバへのストアド・プロシージャのインストール

ターゲット・サーバでジョブを実行するには、ジョブを実行するターゲット・サーバに、テンプレートで使用されているストアド・プロシージャをあらかじめインストールしておきます。

### ❖ ターゲット・サーバへのストアド・プロシージャのインストール

1 Sybase インストール・ディレクトリで、インストール・ユーティリティが保存されているディレクトリ *$SYBASE/$SYBASE_ASE/jobscheduler/Templates/sprocs* に移動します。

2 次のパラメータを指定して、installTemplateProcs を実行します。

- UNIX の場合

  ```
  installTemplateProcs target_servername username
  password
  ```

- Windows の場合

  ```
  installTemplateProcs.bat target_servername username
  password
  ```

  各要素の意味は次のとおりです。

  - *target_servername* は、ターゲット・サーバの名前です。
  - *username* は、テンプレートをインストールするユーザのターゲット・サーバのログイン名です。
  - *password* は、テンプレートをインストールするユーザのターゲット・サーバのパスワードです。

これで、ターゲット・サーバの sybsystemprocs データベースにストアド・プロシージャがインストールされます。ストアド・プロシージャの詳細については、「第 5 章 コマンド・リファレンス」を参照してください。

# Job Scheduler へのテンプレートのインストール

テンプレートを使用するには、ジョブの管理用に選択した Job Scheduler にテンプレートをあらかじめインストールしておきます。

### ❖ Job Scheduler テンプレートのインストール

1 Sybase インストール・ディレクトリで、インストール・ユーティリティが保存されているディレクトリ *$SYBASE/$SYBASE_ASE/jobscheduler/Templates/xml* に移動します。

2　次のパラメータを指定して、installTemplateXML を実行します。

- UNIX の場合

  ```
  installTemplateXML servername machinename serverport
  username password language
  ```

- Windows の場合

  ```
  installTemplateXML.bat servername machinename
  serverport username password language
  ```

  各要素の意味は次のとおりです。

  - *servername* は、Job Scheduler がインストールされているサーバの名前です。
  - *machinename* は、Job Scheduler サーバをホストするマシンの名前です。
  - *serverport* は、Job Scheduler サーバへの接続に使用するポート番号です。
  - *username* は、Job Scheduler サーバのユーザのログイン名です。
  - *password* は、Job Scheduler サーバのユーザのパスワードです。
  - *language* は、インストールするテンプレートの言語バージョンを示すコードです。

    Job Scheduler は次の言語をサポートしています。次の言語コードを使用して言語を指定してください。

    - 英語 － en
    - フランス語 － fr
    - 日本語 － ja
    - 韓国語 － ko
    - 簡体字中国語 － zh

    言語コードの指定を省略すると、デフォルトの英語版テンプレートが使用されます。

これで、Job Scheduler テンプレートが Job Scheduler サーバにインストールされます。

# Job Scheduler テンプレートの使用

ジョブの作成を支援する次のテンプレートを利用できます。

## バックアップ

バックアップ・テンプレートを使用すると、データベースのバックアップ・ジョブを容易に作成できるため、データの保護に役立ちます。

- **Sybase backup database to disk template** このテンプレートは dump database コマンドを使用して、1 つまたは複数のデータベースをディスクにバックアップします。このテンプレートからジョブを作成するには、データベース名とダンプ・ディレクトリを指定します。データベースを圧縮するかストライプするか、ダンプ・ファイル名にサーバ名と日付を含めるかどうかも指定できます。
- **Sybase backup transaction log to disk template** このテンプレートは dump transaction コマンドを使用して、1 つまたは複数のデータベースのトランザクション・ログをバックアップします。このテンプレートからジョブを作成するには、データベース名とダンプ・ディレクトリを指定します。データベースを圧縮するかストライプするか、ダンプ・ファイル名にサーバ名と日付を含めるかどうかも指定できます。さらに時間スレッショルドを指定でき、前回のダンプからその指定した時間が経過するとログがダンプされます。同様にロー・スレッショルドも指定でき、指定したロー数に達するとログがダンプされます。いずれのパラメータにも値を指定しない場合、ログ・ダンプは実行されません。

『システム管理ガイド 第 2 巻』の「第 12 章 ユーザ・データベースのバックアップとリストア」を参照してください。

## 統計管理

統計管理テンプレートは、統計情報を常に最新の状態に保つのに役立ちます。これにより、効果的なクエリ・プランを作成できます。

- **Sybase server update statistics template** このテンプレートは update statistics コマンドを使用して、サーバのすべてのデータベースのすべてのテーブルで統計情報を更新します。このテンプレートでは、update statistics コマンドの各オプションの値を指定できます。テーブル、カラム、インデックス値はシステム・テーブルから取得されます。

  データ変更、ページ数、ロー数のスレッショルド値も指定できます。これらのオプションの値を指定した場合は、その値によって update statistics コマンドを実行するかどうかが決まります。これらのスレッショルドのいずれかの現在の値が、指定されたテーブルのスレッショルド以上の場合、そのテーブルに対して update statistics が実行されます。update statistics の実行後、テーブルに対して sp_recompile プロシージャが実行されます。

- **Sybase update statistics template**　このテンプレートは update statistics コマンドを使用して、テーブル、パーティション、カラム、インデックスの各レベルで統計情報を更新します。データ変更、ページ数、ロー数のスレッショルド値も指定できます。データ変更を測定基準として、テーブル、カラム、またはパーティションに対する変更を追跡できます。データ変更をインジケータとして使用すると、update statistics コマンドのパフォーマンスを向上させるのに役立ちます。

  スレッショルド値を指定した場合は、その値によって update statistics を実行するかどうかが決まります。update statistics の実行後、テンプレート・ウィザードで指定したテーブルに対して sp_recompile プロシージャが実行されます。

- **Sybase delete statistics template**　このテンプレートは delete statistics コマンドを使用して、指定したテーブルのすべてのカラム、テーブルの特定のカラム、またはパーティションとパーティションのローカル・インデックスの統計情報を削除します。データベースとテーブル名を指定してください。統計情報を削除するカラムのリストも指定できます。カラム・リストの指定を省略すると、テーブルの全カラムの統計情報が削除されます。

## 再編成

再編成テンプレートは、無駄なテーブル領域や不連続テーブル・データの発生を防ぎ、データベースを常に整理された状態に保つのに役立ちます。

- **Sybase rebuild indexes template**　このテンプレートは、テーブルの 1 つまたは複数のインデックスに対して reorg rebuild コマンドを実行します。reorg rebuild コマンドに必要なデータベース、テーブル、インデックスを指定してください。インデックス・パーティション名や、reorg rebuild コマンド実行後のデータベースのバックアップを指定することもできます。

- **Sybase rebuild tables template**　このテンプレートは、テーブル全体に対して reorg rebuild コマンドを実行します。reorg rebuild コマンドに必要なデータベースとテーブルを指定してください。reorg rebuild コマンドの実行後にデータベースをバックアップするよう指定することもできます。

- **Sybase reclaim index space template**　このテンプレートは、ローの削除や短縮などの更新を行った結果としてページ上に残った未使用領域を再利用できるようにします。reorg reclaim space コマンドを呼び出して、「インデックス」のデータ・ページを再編成します。複数のインデックスを指定すると、各インデックスの空き領域が再利用可能になります。指定したインデックスを格納するインデックス・パーティションの名前を指定することもできます。

- **Sybase reclaim table space template**　このテンプレートは、ローの削除や短縮などの更新を行った結果としてページ上に残った未使用領域を再利用できるようにします。reorg reclaim space コマンドを呼び出して、「テーブル」のデータ・ページを再編成します。複数のテーブルを指定すると、各テーブルの空き領域が再利用可能になります。指定したパーティションのテーブルのパーティション名を指定すると、reorg reclaim space コマンドをそのパーティションに限定できます。パーティション名を指定して、再編成の範囲をテーブルの分割された部分に限定することもできます。

## 再設定

再設定テンプレートは、ユーザ接続、メタデータ・キャッシュ、ロックの各設定を、現在の要求に合った適切な状態に保つのに役立ちます。

- **Sybase reconfigure locks template**　このテンプレートを使用すると、サーバ・データとユーザ入力に基づいて、サーバが一度に許容するロック数の自動再設定を設定できます。追加するロック数を指定できます。
- **Sybase reconfigure metadata cache template**　メタデータ・キャッシュとは、データベース、インデックス、オブジェクトに使用される予約済みのメモリ領域です。このテンプレートを使用して、サーバの各メタデータ・オブジェクト・タイプの適切な数を調べるジョブを作成できます。ジョブを作成するときに、実行時に使用される、現在のサーバ条件に適したオブジェクト数を調べるための値を指定します。このジョブは現在アクティブなオブジェクトの数を調べ、指定されたタイプのオブジェクトの許容数を調整します。

  別の方法として、1 つまたはすべてのメタデータ・オブジェクト・タイプに対して正確な数を指定することもできます。この方法では、このテンプレートのセルフチューニング機能をスキップします。
- **Sybase reconfigure user connections template**　このテンプレートは、sp_configure を呼び出してユーザ接続数を設定します。新しいユーザ接続数は、直接指定することも、ウィザードで指定した情報に基づいてテンプレートで計算させることもできます。

## テンプレートの更新

Job Scheduler テンプレートの更新や新しいテンプレートを、インターネット上の Sybase CodeXchange からダウンロードできます。

### ❖ 更新と新規テンプレートの入手

1　http://www.sybase.com で MySybase にログインします。

2　CodeXchange にログインします。

3　[Projects] タブをクリックします。

4　[Projects] ボックスで ASE プロジェクトをクリックします。

5　下へスクロールしてサブプロジェクトを表示します。Job Scheduler を選択します。

---

**注意**　Adaptive Server の各リリースには、利用できるすべての Job Scheduler テンプレートが含まれています。

---

第 4 章

# コマンド・ラインでの Job Scheduler の使用

この章では、ジョブの作成やスケジューリングをコマンド・ラインから実行する例を紹介します。これらの操作は、Sybase Central 管理ツールの ASE プラグインからも実行できます。詳細については、「第 6 章 Sybase Central の ASE プラグインでの Job Scheduler の管理」を参照してください。



## ジョブの作成

ここでは、Hello World を出力するジョブの作成手順について説明します。

### ❖ Hello World ジョブとその履歴リストの作成

1 「第 2 章 Job Scheduler の設定と実行」に記載されているサーバとユーザのセットアップを実行します。sp_addexternlogin ストアド・プロシージャに指定されたユーザ名とパスワードを使用してターゲット・サーバにログインするパーミッションがあることを確認してください。

2 次のコード例では、sp_sjobcreate プロシージャを使用してジョブを作成します。

```
use sybmgmtdb
go
declare @jobcmd varchar(255), @jobid int
select  @jobcmd='jcmd=print "Hello World.",server=
YOUR_SERVER'
+ ',starttime=' + convert(varchar(32),getdate())
exec @jobid=sp_sjobcreate 'sjname=hello', @jobcmd
go
```

各パラメータの意味は次のとおりです。

*YOUR_SERVER* はターゲット・サーバの名前です。

3 sp_sjobhelp を使用して、作成したスケジュール・ジョブを表示します。

```
> exec sp_sjobhelp 'sjname=hello'
> go
```

4 次のようなスケジュール・ジョブの概要が返されます。

```
sjob_id: 127  name: 'hello'
owner          : jsadmin1
created        : Jul 14 2005 4:42AM
state          : enabled
job name       : 114 - 'job_114'
schedule name  : 115 - 'sched_115'
server         : pgibson_js
-- job --------:
description    :
owner          : jsadmin1
created        : Jul 14 2005 4:42AM
-- schedule ---:
description    :
owner          : jsadmin1
created        : Jul 14 2005 4:42AM
starttime      : 04:42
startdate      : 14 Jul 2005
```

5 sp_sjobhistory を使用して、スケジュール・ジョブの実行履歴を示す短いリストを表示します。

```
sp_sjobhistory 'sjname=hello', @option='list_short'
go
```

6 次のような短いリストが返されます。

```
sjob_id  sjob_jobname  sjob_schedname  sjob_server    sjob_state
-------  ------------  --------------  -----------    ----------
127      job_114       sched_115       pgibson_js     C2

sjob_start             sjob_user_run   sjob_user_req  sjob_size
----------             -------------   -------------  ---------
Jul 14 2005 4:42AM    jsadmin1        jsadmin1       51
```

7 sp_sjobhistory を使用して、スケジュール・ジョブ実行時の出力内容を表示します。

```
sp_sjobhistory 'sjname=hello', 'list_output'
go
```

8　次のような出力が返されます。

```
 jsout_run_id   jsout_seqno   jsout_size    jsout_text
------------   -----------   -----------   ----------
144            0             51            Changed database context to 'master'.

hello world
```

## ジョブ・ステータス・コード

次の表は、履歴テーブルで返されるジョブ・ステータス・コードをまとめたものです。

**表 4-1: ジョブ・ステータス・コードの説明**

| ステータス名 | ステータス・コード | 説明 |
|---|---|---|
| 待機 | W | 作成されたジョブ、またはスケジュールされるのを待っているジョブの初期状態。 |
| キューイング | Q | Job Scheduler はジョブを実行可能だが、ジョブを実行するための空きスレッドを待っている状態。 |
| ビジー | B | このジョブはすでに実行されているため、Job Scheduler がジョブを開始しなかった。 |
| 実行可能 | R1 | Job Scheduler がジョブを開始した。 |
| 実行 | R2 | ジョブが現在実行されている。 |
| 完了する | C1 | ジョブが完了し、Job Scheduler がログや履歴などをクリーンアップ中である。 |
| 完了 | C2 | ジョブが完了し、後処理 SQL も完了した。 |
| 終了 | T1 | Job Scheduler がスレッドを強制終了してジョブを終了させた。 |
| 終了 | T2 | Job Scheduler がジョブを終了させ、後処理 SQL も完了した。 |
| timing-out | X1 | ジョブがタイムアウトし、Job Scheduler がジョブを終了させた。 |
| timed-out | X2 | ジョブがタイムアウトし、終了した。 |
| 失敗 | M | Job Scheduler がアクティブでなかったため、スケジュール・ジョブの実行に失敗した。 |

# スケジュールの作成

次のコード例では、Job Scheduler でコマンド・ラインからスケジュールを作成する方法を示します。Sybase Central 管理ツールの ASE プラグインを使用しても同じタスクを実行できます。

繰り返しスケジュールを作成する場合、開始時刻と終了時刻が必要です。

---

**注意** スケジュールの名前を作成するときは、必ず英字で始めてください。数字で始まる名前を作成すると、エラーが発生します。

---

次の例は、毎日 08:00 ～ 18:00 の間、5 分ごとに動作するスケジュールの作成方法を示します。

```
sp_sjobcreate @name='sname=every5m_8to6',@option='repeats=
5minutes, starttime=08:00am, endtime=18:00'
```

次の例は、土曜日と日曜日の 08:00 ～ 18:00 の間、1 時間ごとに動作するスケジュールの作成方法を示します。

```
sp_sjobcreate @name='sname=hourly_8to6_weekends',
@option='repeats=1hour,starttime=08:00am,endtime=18:00,days=
Saturday:Sunday
```

次の例は、毎月 1 日と末日の 04:00am に動作するスケジュールの作成方法を示します。32 は月の末日を表します。

```
sp_sjobcreate @name='sname=run_4am_1st_and_last',
@option='starttime=04:00,endtime=04:00,dates=1:32'
```

次の例は、1 月 1 日～ 2 月 1 日の間、09:00 に動作するスケジュールの作成方法を示します。

```
sp_sjobcreate @name='sname=run_daily_9am_Jan',
@option='starttime=09:00,endtime=09:00,repeats=
1day,startdate=1 January 2005,enddate=1 February 2005'
```

---

**注意** スケジュールの開始時刻にはその時刻自体も含まれます。つまり、ジョブは指定された開始時刻に開始されます。スケジュールの終了時刻ではその時刻は除外されます。つまり、ジョブの実行時刻に終了時刻は含まれません。たとえば、スケジュールの開始時刻が 13:00、終了時刻が 16:00 に指定されており、1 時間ごとに繰り返される場合、ジョブは 13:00、14:00、15:00 に実行されますが、16:00 には実行されません。

---

# スケジュール・ジョブの作成

次のコード例は、Job Scheduler でコマンド・ラインからスケジュール・ジョブを作成する方法を示します。Sybase Central 管理ツールの ASE プラグインを使用しても同じタスクを実行できます。

**注意** スケジュール・ジョブの名前を作成するときは、必ず英字で始めてください。数字で始まる名前を作成すると、エラーが発生します。

### ❖ 試験的に 1 か月間の統計を収集するスケジュール・ジョブの作成

1 スケジュールを作成します。

```
sp_sjobcreate @name='sname=Aug_stats_trial',
@option='starttime=23:00,endtime=23:00,repeats=
1day,startdate=1 August
2005,enddate=31 August 2005'
```

2 ジョブを作成します。

```
sp_sjobcreate @name='jname=new_stats_proc',
@option='jcmd=''exec statsdb..
new_stats_proc'',jdesc=New statistics
proc.,jproperties=shared'
```

3 “devtest1” というサーバにスケジュール・ジョブを作成します。

```
sp_sjobcreate @name='sjname=new_stats_devtest1',
@option='sname=Aug_stats_trial,jname=
new_stats_proc,server=devtest1'
```

**注意** スケジュール・ジョブに割り当てるサーバ名は interfaces ファイル内のネットワーク名です (sp_addserver の *pname* プロパティ )。sysservers テーブルに定義されている論理名ではありません。

# スケジュール・ジョブの削除

スケジュール・ジョブを削除する例を次に示します。

### ❖ スケジュール・ジョブの削除

- sp_sjobdrop を使用して、“new_stats_devtest1” というスケジュール・ジョブを削除します。

```
sp_sjobdrop 'sjname=new_stats_devtest1'
go
```

# スケジュール・ジョブの修正

sp_sjobmodify を使用すると、スケジュール・ジョブを一定の間隔で実行したり、特定の曜日や日付に実行したりするように設定できます。次の例では、“hello world” ジョブを修正します。

次のコードを実行すると、毎日 09:00 にジョブを実行するように設定します。

```
sp_sjobmodify 'sjname=hello', 'starttime=09:00,repeats=1day'
go
```

次のコードでは、ジョブのタイムアウト・プロパティを 120 分に設定します。

```
sp_sjobmodify @name='sjname=hello',
@option='default_timeout=120'
go
```

次のコードでは、月曜日と金曜日の 09:00 にジョブを実行するように設定します。

```
sp_sjobmodify 'sjname=hello', 'starttime=09:00,endtime=
09:00,days=Monday:Friday'
go
```

次のコードでは、スケジュール・ジョブのターゲット・サーバを変更します。

```
sp_sjobmodify 'sjname=hello', 'server=prodASE'
go
```

# ターゲット・サーバでのストアド・プロシージャの呼び出し

必要な機能を実行するために、スケジュール・ジョブがターゲット・サーバでストアド・プロシージャを呼び出すことがよくあります。次は、sp_who を呼び出すスケジュール・ジョブの例です。

```
use sybmgmtdb
go
declare @jobcmd varchar(255), @jobid int
select  @jobcmd='jcmd=exec sp_who,server=YOUR_SERVER'
+ ',starttime=' + convert(varchar(32),getdate())
exec @jobid=sp_sjobcreate 'sjname=hello', @jobcmd
go
```

*YOUR_SERVER* はターゲット・サーバの名前です。

---

**注意** Job Scheduler はジョブを実行する場合、SQL をジョブ・テキストの前に追加して、ジョブ runid とスケジュール・ジョブ ID を設定します。このため、sp_who を呼び出すには、exec を使用してストアド・プロシージャを呼び出す必要があります。

---

## ジョブの管理

sp_sjobcontrol を使用すると、Job Scheduler でアドホックなアクションを実行できます。実行できるアクションは次のとおりです。

- 実行中のジョブの終了
- スケジュール・ジョブの即時実行
- スケジュール・ジョブの有効化と無効化

詳細については、sp_sjobcontrol (41 ページ) を参照してください。

## ジョブの履歴とログの管理

ルールやスレッショルドなどの設定を調整して、ジョブの履歴や出力ログの自動管理を制御できます。

Sybase Central の ASE プラグインを使用してこれらの設定パラメータを調整するには、「Job Scheduler の管理」(50 ページ) を参照してください。

第 5 章

# コマンド・リファレンス

この章では、コマンド・ライン・ストアド・プロシージャのプロパティとパラメータについて説明します。



## コマンド・ライン・ストアド・プロシージャ

**追加、修正、削除**

スケジュール・ジョブの作成方法は 2 つあります。1 回の操作でジョブとスケジュールの両方の情報を指定する方法と、ジョブとスケジュールを別々に定義してから、それらを組み合わせてスケジュール・ジョブを作成する方法です。

**レポートとリスト**

Job Scheduler データベースに設定されているジョブとスケジュールの情報を取得するためのストアド・プロシージャが、コマンド・ライン・インタフェースに用意されています。その情報は、フォーマットされたレポートまたは単純なリストの形式で取得できます。

**ジョブの実行**

ジョブを実行するには、ジョブをスケジューリングするかアドホック・コマンドを使用します。

**ジョブ履歴とジョブ出力**

ジョブ履歴とジョブ出力は別々に管理できます。ジョブ履歴を削除すると対応するジョブ出力もすべて削除されますが、ジョブ出力レコードを削除してもジョブ履歴は削除されません。この柔軟性により、保持するジョブ履歴とジョブ出力の量をさまざまなルールで制御できます。

js_admin_role 以外の Job Scheduler ユーザの動作は、履歴エントリに自分のユーザ名が記録されているジョブの、ジョブ履歴とジョブ出力の表示および操作に限定されます。これに対し、js_admin_role ユーザは、all オプションを使用することによってこの制限を無効にし、履歴と出力のスコープをすべての Job Scheduler ユーザに拡大できます。

**表 5-1: Job Scheduler ストアド・プロシージャの説明**

| プロシージャ名 | プロシージャの目的 |
|---|---|
| 作成、修正、削除 | |
| sp_sjobcreate | スケジュール・ジョブ、ジョブ、スケジュールの作成 |
| sp_sjobcmd | ジョブの SQL コマンド・テキストの管理 |
| sp_sjobmodify | スケジュール・ジョブ、ジョブ、スケジュールの修正 |
| sp_sjobdrop | スケジュール・ジョブ、ジョブ、スケジュールの削除 |
| レポート、リスト | |
| sp_sjobhelp | スケジュール・ジョブと実行中のジョブのレポートおよびリストの作成 |
| ジョブの実行 | |
| sp_sjobcontrol | Job Scheduler の管理とスケジュール・ジョブまたは実行中のジョブの制御 |
| ジョブ履歴、ジョブ出力 | |
| sp_sjobhistory | ジョブ履歴とジョブ出力の表示および管理 |

# コマンドの構文

Job Scheduler には、スケジュール・ジョブを操作するための一連のストアド・プロシージャが用意されています。各ストアド・プロシージャの名前には、プレフィックスとして sp_*sjob* が付けられます。*sjob* はスケジュール・ジョブの省略形です。

ジョブ、スケジュール、スケジュール・ジョブを作成するストアド・プロシージャは、サーバ・ユーザ名を使用して所有者を記録します。このユーザ名の取得には、システム関数 suser_name が使用されます。ストアド・プロシージャは、基本となるジョブ・オブジェクトやスケジュール・オブジェクトの操作もサポートしており、実行中のスケジュール・ジョブを制御したり、それらのジョブが生成する履歴や出力を管理したりするためのインタフェースも備えています。

ストアド・プロシージャは、操作対象となるオブジェクトを指定する name 引数または ID 引数を取ります。ジョブ、スケジュール、スケジュール・ジョブなど複数の異なるオブジェクトを操作対象とするストアド・プロシージャは、name 引数または ID 引数の前に、その名前または ID が指すオブジェクトの種類を指定するプレフィクスを取ります。スケジュールの場合は @sname、ジョブの場合は @jname です。

次は、@sname の使用例です。

```
@name= 'sname=daily_schedule'
```

次は、@jname の使用例です。

```
@name= 'jname=run_update_stats'
```

デフォルトでは、名前または ID はスケジュール・ジョブを指します。

**一般的な使用法**

- コマンド名、コマンド・キーワード、単語 “null”、データ項目、統計タイプは、大文字と小文字が区別されません。ファイル名、ビュー名、ユーザが指定するその他の名前は、大文字と小文字が区別されます。
- パラメータ値に埋め込みスペース (データ項目、統計タイプ、日時指定における埋め込みスペースなど) が含まれる場合は、その値を引用符で囲みます。正しく閉じられた 1 組の一重引用符または二重引用符は、有効なデリミタになります。
- パラメータ値に、値全体を区切るために使用されている文字と同じ埋め込み引用符が含まれる場合、そのパラメータ値の中に 1 組の引用符を使用します。この 1 組の引用符は、Job Scheduler によって 1 文字に圧縮されます。
- 引用符の中の “null” という単語は、キーワードと見なされません。
- Job Scheduler のコマンドは複数行にわたって入力することもできます。

# sp_sjobcreate

**説明**

このプロシージャには複数の用途があります。

- ジョブ情報のみが指定された場合は、新しいジョブを作成します。
- スケジュール情報のみが指定された場合は、新しいスケジュールを作成します。
- ジョブとスケジュールの情報が指定された場合は、既存のジョブとスケジュールを組み合わせて新しいスケジュール・ジョブを作成するか、または新しいジョブとスケジュールを作成し、それらを組み合わせて新しいスケジュール・ジョブを作成します。

デフォルトでは、@name 引数はスケジュール・ジョブの名前です。ジョブまたはスケジュールの名前を指定するには、@name 引数にプレフィクスとして jname または sname を付けます。

**構文**

sp_sjobcreate @name='*jsname*', @options='server, jname, jdesc, jcmd, sname, sdesc, repeats, properties, starttime, enddate, endtime, days, dates'

パラメータ

*name (jname, sname, sjname)*

新しいジョブ、スケジュール、スケジュール・ジョブの名前。

---

**注意** ジョブ、スケジュール、スケジュール・ジョブの名前を作成するときは、名前を必ず英字で始めてください。数字で始まる名前を作成すると、エラーが発生します。

---

*option*

ジョブ、スケジュール、スケジュール・ジョブの作成に使用するフィールド名とその値をカンマで区切ったリスト。値は次のとおり。

- server － ジョブを実行するサーバの名前。デフォルト値はローカル・サーバです。
- jname － ジョブの名前。ユニークな名前を指定します。
- jdesc － ジョブを説明するコメント。
- jcmd － SQL テキスト。SQL テキストを直接渡しやすい単純なジョブで使用します。
- sname － スケジュールの名前。ユニークな名前を指定します。
- default_timeout － ジョブを実行できる最大時間。スケジュール・ジョブのタイムアウト・プロパティが設定されていない場合に、この値が使用されます。
- sdesc － スケジュールを説明するコメント。
- timeout － スケジュール・ジョブを実行できる最大時間。この値は、ジョブの default_timeout 値より優先されます。
- repeats － スケジュールを繰り返す間隔。0 または NULL の場合、スケジュールを繰り返しません。値には、数値の後ろに次のいずれかの単位を指定します。
  - day または d
  - days または dd
  - hour または h
  - hours または hh
  - minute または m
  - minutes または mm

- properties − ジョブ、スケジュール、またはスケジュール・ジョブのプロパティ。それぞれ jproperties、sproperties、sjproperties。次のプロパティがあります。
  - jproperties (ジョブ・プロパティ )
    - multi_task
    - run_as_owner
    - no_job_header
    - no_sql_batching
    - no_conn_redirection
    - shared
  - sjproperties (スケジュール・ジョブのプロパティ )
    - shared_run
    - only_at_starttime
    - disable_on_faillure
    - delete_on_completion
    - no_output_log
    - shared − (スケジュール・プロパティのみ)
- startdate − スケジュールがアクティブになる日付。
- starttime − スケジュール・ジョブの開始時刻。
- enddate − スケジュールが非アクティブになる日付。
- endtime − その日のうちにスケジュールが非アクティブになる時刻。
- days − 曜日をコロンで区切ったリスト。フルネーム、またはサーバのロケールでの省略形が使用できます。
- dates − 日付 (1 〜 31) をコロンで区切ったリスト。32 は月の末日を表します。

**注意**　曜日と日付の両方の値を指定しようとすると、エラーが発生します。

**戻り値**

- 新しいジョブのジョブ ID。
- 新しいスケジュールのスケジュール ID。
- 新しいスケジュール・ジョブのスケジュール・ジョブ ID。
- エラー・コード。

**例**

**例 1** 次の例は、ストアド・プロシージャを実行するための “find_old_logins” という新しいジョブの作成方法を示します。

```
sp_sjobcreate @name='jname=find_old_logins',
@option='jcmd=exec sp_find_old_logins,jproperties=shared'
```

**例 2** 次の例は、午前 1 時に有効になり、毎日繰り返す新しいスケジュールの作成方法を示します。

```
sp_sjobcreate @name='sname=daily 01:00am',
@option='repeats=1day,starttime=01:00am, endtime=02:00am'
```

**例 3** 次の例は、既存のジョブ “find_old_logins” とスケジュール “daily 01:00am” を使用して、サーバ “dev1” で実行する新しいスケジュール・ジョブの作成方法を示します。

```
sp_sjobcreate @name='dev1_old_logins',
@option='server=dev1,jname=find_old_logins, sname=daily 01:00am'
```

**例 4** 次の例は、“load_sales_data” という新しいジョブと毎週月曜、水曜、金曜の 23：00 に実行する新しいスケジュールを使用して、“evening_sales_report” という新しいスケジュールを作成する方法を示します。このスケジュールには、スケジュール・テーブルの ID 値を基にしたデフォルト名が付きます。

```
sp_sjobcreate @name='evening_sales_report',
@option='server=reports, jname=load_sales_data,
jcmd=exec sp_new_sales_data,
starttime=23:00,endtime=23:00,days=Monday:Wednesday:Friday'
```

**使用法**

- days と dates の両方の値を使用したスケジュールは作成できません。ただし、days と dates をそれぞれ使用した複数のスケジュールを 1 つのジョブにバインドすることはできます。
- repeats 値が 1 日を超えるスケジュールでは、days と dates の値は無効です。
- repeats 値が 1 日のスケジュールでは、days と dates の値は無効です。repeats 値が 1 日の場合、デフォルトではすべての曜日が有効です。
- repeats 値が 1 日未満で days 値が NULL のスケジュールは、デフォルトではすべての曜日で有効です。
- startdate が enddate と同じであるときに、endtime を指定する場合は、endtime 値に starttime 値と同じ時刻またはそれ以降の時刻を指定してください。
- repeats 値が 1 日以上のスケジュールでは、endtime 値は意味がないため無視されます。
- endtime の指定を省略すると、デフォルトの午前 0 時が適用されます。
- starttime の指定を省略すると、デフォルトの午前 0 時が適用されます。

# sp_sjobcmd

**説明**

ジョブの SQL テキストを操作できます。@option='list' を指定すると、SQL テキストがリストされます。

デフォルトでは、@name 引数はスケジュール・ジョブの名前または ID です。ジョブの名前または ID を指定するには、@name 引数にプレフィクスとして jname を付けます。

**構文**

sp_sjobcmd @name='...', @option='...', @text='...'

**パラメータ**

*name*

スケジュール・ジョブまたはジョブの名前または ID。プレフィクスとして jname が付いている場合は、ジョブの名前または ID です。

*option*

ジョブの SQL テキストに対する操作を add (追加)、list (リスト)、drop (削除) から選択するためのオプション。

- drop − 指定されたジョブの SQL テキストをすべて削除します。
- list − データベースに格納されているジョブ・コマンド・テキストを表示します。
- add − 指定されたジョブの既存の SQL テキストに追加します。これにより、大きな SQL バッチを複数のまとまりに分けて格納できます。JS Agent がジョブを実行するときに、ジョブの SQL テキストが連結されます。

*text*

格納する SQL テキスト。

**戻り値**

0 またはエラー・コードを返します。

**例**

**例 1** 次の例は、“svr1_check_stats” というスケジュール・ジョブによって参照されるジョブの SQL テキストのリスト方法を示します。

```
sp_sjobcmd 'sjname=svr1_check_stats', 'list'
```

**例 2** 次の例では、“load_sales_data” ジョブの既存の SQL テキストを削除してから、新しい SQL テキストを格納する方法を示します。

```
sp_sjobcmd 'jname=load_sales_data', 'drop'
sp_sjobcmd 'jname=load_sales_data', 'add',
@text='truncate table  sales_report_data'
go
'
```

**使用法**

Job Scheduler は、ジョブ・テキストの末尾に新しい行を暗黙的に追加しません。したがって、sp_sjobcmd でジョブを作成する場合は、呼び出しの後に新しい行を明示的に追加してください。

```
sp_sjobcmd 'jname=load_sales_data', 'drop'
sp_sjobcmd 'jname=load_sales_data', 'add',
@text='truncate table  sales_report_data'
go
```

# sp_sjobmodify

**説明**

スケジュール・ジョブ、ジョブ、スケジュールの任意のフィールドを修正できます。

**構文**

sp_sjobmodify @name='...', @option='...'

**パラメータ**

*name*

修正するスケジュール・ジョブ、ジョブ、スケジュールの名前または ID。

*option*

修正するフィールドの名前と新しい値をカンマで区切ったリスト。

- sjname − スケジュール・ジョブの新しい名前。ユニークな名前を指定します。新しい ID は指定できません。
- enable − スケジュール・ジョブを有効にするには 1 を、無効にするには 0 を指定します。
- sjproperties − スケジュール・ジョブのプロパティ。
- sjowner − スケジュール・ジョブの所有者。所有者を変更するには、呼び出し元ユーザに js_admin_role が必要です。
- server − スケジュール・ジョブを実行するサーバ。
- timeout − スケジュール・ジョブとサーバについて分単位で指定するタイムアウト値。
- locale − スケジュール・ジョブの実行時にクライアント接続で使用するロケール。
- jname − ジョブの新しい名前。ユニークな名前を指定します。新しい ID は指定できません。
- jdesc − ジョブを説明するコメント。
- jcmd − SQL テキスト。SQL テキストを直接渡しやすい単純なジョブで使用します。既存のすべての SQL テキストを置換します。
- jproperties − ジョブ、スケジュール、またはスケジュール・ジョブのプロパティ。それぞれ jproperties、sproperties、sjproperties。次のプロパティがあります。
  - jproperties (ジョブ・プロパティ )
    - multi_task
    - run_as_owner
    - no_job_header
    - no_sql_batching
    - no_conn_redirection
    - shared

    - sjproperties (スケジュール・ジョブのプロパティ )
        - shared_run
        - only_at_starttime
        - disable_on_faillure
        - delete_on_completion
        - no_output_log
        - shared － (スケジュール・プロパティのみ)
- jowner － ジョブの所有者。所有者を変更するには、呼び出し元ユーザに js_admin_role が必要です。
- default_timeout － ジョブのタイムアウト値。分単位で指定します。
- sname － スケジュールの新しい名前。ユニークな名前を指定します。新しい ID は指定できません。
- sdesc － スケジュールを説明するコメント。
- sowner － 所有者を変更するには js_admin_role が必要です。
- sproperties － スケジュールのプロパティ。
- reset － “true” を指定すると、スケジュールのタイミング情報がリセットされます。クリアされるのは、repeat、units、startdate、starttime、enddate、endtime、days、dates です。
- repeats － スケジュールを繰り返す間隔。NULL または 0 の場合、スケジュールを繰り返しません。値には、数値の後ろに次のいずれかの単位を指定します。
    - day または d
    - days または dd
    - hour または h
    - hours または hh
    - minute または m
    - minutes または mm
- startdate － スケジュールがアクティブになる日付。時刻の部分は無視されます。
- enddate － スケジュールが非アクティブになる日付。時刻の部分は無視されます。
- starttime － アクティブなスケジュールが操作を開始する時刻。日付の部分は無視されます。starttime の指定を省略すると、デフォルトの午前 0 時が適用されます。

- endtime – アクティブなスケジュールが操作を終了する時刻。日付の部分は無視されます。endtime の指定を省略すると、デフォルトの午前 0 時が適用されます。
- days – 曜日をコロンで区切ったリスト。フルネーム、またはサーバのロケールでの省略形が使用できます。
- dates – 日付をコロンで区切ったリスト。

**戻り値**

成功した場合は 0 を、それ以外の場合はエラー・コードを返します。

**例**

**例 1** 次の例は、“svr1_clean_stats” というスケジュール・ジョブを修正して、10:00 から 16:00 まで 2 時間ごとに繰り返すようスケジュールを更新する方法を示します。スケジュール・ジョブを実行する日付または曜日は変更しません。

```
sp_sjobmodify @name='svr1_clean_stats',
@option='repeats=2hours,starttime=10:00,endtime=16:00'
```

**例 2** 次の例は、ジョブ “backup_db5” を修正して、デフォルトのタイムアウト値を 120 分に更新する方法を示します。

```
sp_sjobmodify @name='jname=backup_db5',
@option='default_timeout=120'
```

**例 3** 次の例は、ジョブ “orders_picked_report” を修正し、共有プロパティを削除して所有者を mary に変更する方法を示します。ジョブの所有者を変更するには、呼び出し元ユーザに js_admin_role が必要です。

```
sp_sjobmodify @name='jname=orders_picked_report',
@option='jproperties=shared:false,owner=mary'
```

**使用法**

デフォルトでは、@name 引数はスケジュール・ジョブの名前または ID です。ジョブまたはスケジュールの名前または ID を指定するには、@name 引数にプレフィクスとして jname または sname を付けます。

次は、@jname の使用例です。

```
@name= 'jname=run_update_stats'
```

次は、@sname の使用例です。

```
@name= 'sname=daily_schedule'
```

# sp_sjobdrop

**説明**

ジョブ、スケジュール、またはスケジュール・ジョブを削除します。

デフォルトでは、@name 引数はスケジュール・ジョブの名前または ID です。ジョブまたはスケジュールの名前または ID を指定するには、@name 引数にプレフィクスとして jname または sname を付けます。

**構文**

sp_sjobdrop @name='...', @option='...'

**パラメータ**

*name*
削除するスケジュール・ジョブ、ジョブ、スケジュールの名前または ID。

*option*
コマンドのオプションのリスト。

- all − ジョブまたはスケジュールでこのオプションを使用すると、呼び出し元ユーザが所有するスケジュール・ジョブのうち、指定したジョブまたはスケジュールが使用されているものすべてが削除されます。スケジュール・ジョブでこのオプションを使用すると、指定したスケジュール・ジョブに関連付けられているジョブとスケジュールも、他のスケジュール・ジョブから参照されていない限り削除されます。
- all_users − js_admin_role ユーザに、別のユーザが所有するスケジュール・ジョブ、ジョブ、スケジュールに対する sp_sjobdrop の使用を許可します。
- show − この sp_sjobdrop の呼び出しによって削除されるジョブ、スケジュール、およびスケジュール・ジョブを表示してから、実際に削除します。
- force − 実行中のジョブもデータベースから削除します。

(これらのパラメータの詳細については、「使用法」を参照してください)。

**戻り値**

成功した場合は 0 を、それ以外の場合はエラー・コードを返します。

**例**

**例 1** 次の例は、“svr1_clean_stats” というスケジュール・ジョブの削除方法を示します。

```
sp_sjobdrop 'svr1_clean_stats'
```

**例 2** 次の例は、“daily 01:00am” というスケジュールの削除方法を示します。

```
sp_sjobdrop 'sname=daily 01:00am'
```

**例 3** 次の例は、“load_sales_data” というジョブの削除方法を示します。

```
sp_sjobdrop 'jname=load_sales_data'
```

**例 4** 次の例は、“load_sales_data” というジョブと、そのジョブが使用されている、呼び出し元ユーザが所有するスケジュール・ジョブの削除方法を示します。

```
sp_sjobdrop @name='jname=load_sales_data', @option='all'
```

**使用法**

- force オプションを指定せずに実行中のスケジュール・ジョブを削除しようとすると、エラーが発生します。
- デフォルトでは、sp_sjobdrop プロシージャは、@name 引数で指定されたスケジュール・ジョブ、ジョブ、またはスケジュールだけを削除対象とします。

ジョブとスケジュールの全般的な管理用に、all オプションと all_users オプションが用意されています。これらのオプションについて次に説明します。

- all オプションも all_users オプションも使用せずにスケジュール・ジョブを指定すると、指定されたスケジュール・ジョブだけが削除されます。
- all オプションを使用してスケジュール・ジョブを指定すると、そのスケジュール・ジョブが削除されます。ジョブへの参照が他になければ、ジョブも削除されます。同様に、スケジュールへの参照が他になければ、スケジュールも削除されます。
- all オプションも all_users オプションも使用せずにジョブまたはスケジュールを指定すると、そのジョブまたはスケジュールを参照しているスケジュール・ジョブがある場合にエラーが発生します。
- all_users オプションを使用してジョブまたはスケジュールを指定し、そのジョブまたはスケジュールを参照するスケジュール・ジョブの所有者が呼び出し元ユーザである場合、指定したジョブまたはスケジュールとそれを参照するスケジュール・ジョブが削除されます。
- all_users オプションを使用してジョブまたはスケジュールを指定し、呼び出し元ユーザが js_admin_role である場合、指定したジョブまたはスケジュールとそれを参照するすべてのスケジュール・ジョブが削除されます。

show オプションは、sp_dropjob の呼び出しによってどのスケジュール・ジョブ、ジョブ、スケジュールが削除されるかを確認できるように用意されています。実際の削除操作は行われません。

# sp_sjobhelp

**説明**

呼び出し元ユーザが参照できるすべてのスケジュール・ジョブ、ジョブ、または実行中のジョブに関するリストやレポート、または名前とオプションの値に基づいて限定されたセットに関するリストやレポートを生成します。

**構文**

sp_sjobhelp @name='...', @option='...'

**パラメータ**

*name*

スケジュール・ジョブやジョブの名前、ID、または runid。@name 引数を使用すると、スコープを単一のスケジュール・ジョブ、特定のジョブを使用するすべてのスケジュール・ジョブ、または単一のスケジュール・ジョブ実行に限定できます。デフォルトでは、@name はスケジュール・ジョブの名前 ID です。ジョブの名前、ID、または runid を指定するには、@name 引数にプレフィックスとして jname または runid を付けます。

*option*

オプション、実行するアクションを定義する option_name=option_value の組、返された情報を絞り込むフィルタをカンマで区切ったリストを指定します。

- report − レポート形式にフォーマットします。
- list − リスト形式にフォーマットします。
- running − 実行中のジョブのレポートを生成します。
- scheduled − スケジュール・ジョブのレポートを生成します。
- unscheduled − スケジュールされていないジョブのレポートを生成します。
- all_users − すべてのユーザの情報を含めます。
- owner − このユーザが所有する情報に限定します。
- jobs − ジョブに関する情報を提供します。
- schedules − スケジュールに関する情報を提供します。
- user − このユーザに代わって実行されたスケジュール・ジョブの情報に限定します。実行中のジョブにのみ適用されます。

**戻り値**　成功した場合は 0 を、それ以外の場合はエラー・コードを返します。

**例**　**例 1** 次の例は、実行中のすべてのスケジュール・ジョブのリストの作成方法を示します。

```
sp_sjobhelp  @option='list,all_users,running'
```

**例 2** 次の例は、呼び出し元ユーザによってスケジューリングされたすべてのジョブを表示するレポートの作成方法を示します。

```
sp_sjobhelp @option='report,scheduled'
```

**例 3** 次の例は、どのスケジュール・ジョブでも使用されていないジョブとスケジュールのリストの作成方法を示します。

```
sp_sjobhelp  @option='list,all_users,unscheduled'
```

**使用法**　レポートは、スケジュール・ジョブ、スケジュールされていないジョブ、スケジュールを示す複数のセクションに分かれています。

## sp_sjobcontrol

**説明**　実行中のジョブを制御するためのインタフェースを定義します。

**構文**　sp_sjobcontrol @name='...', @option='...'

**パラメータ**

*name*

制御するスケジュール・ジョブやジョブの名前、ID、または runid。

*option*

オプションと、実行するアクションを定義する option_name=option_value の組をカンマで区切ったリストを指定します。

- terminate − 指定された名前、ID、または runid を持つジョブのインスタンスを終了します。
- enable − 指定された名前または ID を持つスケジュール・ジョブのインスタンスを有効にします。
- disable − 指定された名前または ID を持つスケジュール・ジョブのインスタンスを無効にします。現在実行中のジョブには影響しません。
- run_now − 指定された名前または ID を持つジョブを直ちに実行するようにスケジューリングします。ただし、対応するジョブ用に作成された重複しないスケジュール ジョブが必要です。これがない場合は、sp_sjobcontrol はエラーを返します。
- all_users − 役割 js_admin_role を持つユーザに、他のユーザが所有するスケジュール・ジョブの管理を許可します。
- start_js − Job Scheduler を開始します。
- stop_js − Job Scheduler を停止して、実行中のジョブをすべて終了します。
- stop_js_wait − 実行中のジョブが完了するのを待ってから Job Scheduler を停止します。
- stop_js_timeout=*minutes* − 指定された時間 ( 分 ) だけ待ってから Job Scheduler を停止して、実行中のジョブをすべて終了します。

**戻り値**

成功した場合は 0 を、それ以外の場合はエラー・コードを返します。

**例**

**例 1** 次の例は、runid が 1532 である実行中のジョブを終了するよう JS Task と JS Agent に要求する方法を示します。

```
sp_sjobcontrol @name='runid=1532', @option='terminate'
```

**例 2** 次の例は、"svr1_load_sales_data" というスケジュール・ジョブを直ちに実行できるように設定する方法を示します。既存のスケジュールは変更されません。

```
sp_sjobcontrol @name='svr1_load_sales_data',
@option='run_now'
```

**例 3** 次の例は、呼び出し元ユーザのスケジュール・ジョブのうち、"check_user_logins" を使用するものをすべて無効にする方法を示します。

```
sp_sjobcontrol @name='jname=check_user_logins',
@option='disable'
```

**使用法**

デフォルトでは、@name はスケジュール・ジョブの名前または ID です。ジョブの名前、ID、または runid を指定するには、@name 引数にプレフィックスとして jname または runid を付けます。

## sp_sjobhistory

**説明**

ジョブ履歴とジョブ出力のログをリストするか削除します。このプロシージャを使用すると、完了したスケジュール・ジョブの結果を確認できます。ジョブ履歴とジョブ出力のログ管理にも使用できます。

**構文**

sp_sjobhistory @name='...', @option='...'

**パラメータ**

*name*

スケジュール・ジョブやジョブの名前、ID、または runid。

@name 引数を使用すると、履歴のスコープを単一のスケジュール・ジョブ、特定のジョブを使用するすべてのスケジュール・ジョブ、または単一のスケジュール・ジョブ実行に限定できます。デフォルトでは、@name はスケジュール・ジョブの名前または ID です。ジョブの名前、ID、または runid を指定するには、@name 引数にプレフィックスとして `jname` または `runid` を付けます。

*option*

オプションと、ジョブ履歴およびジョブ出力のフィルタされたセットに対して実行するアクションを定義する option_name=option_value の組をカンマで区切ったリストを指定します。

- list − フィルタ条件に一致するジョブのジョブ履歴フィールドをリストします。
- list_short − フィルタ条件に一致するジョブのジョブ履歴フィールドの一部をリストします。
- drop − フィルタ条件に一致するジョブのジョブ履歴エントリとジョブ出力エントリを削除します。
- list_output − フィルタ条件に一致するジョブのジョブ出力をリストします。
- drop_output − フィルタ条件に一致するジョブのジョブ出力エントリを削除します。対応するジョブ履歴エントリは削除されません。
- all_users − すべてのユーザのジョブをスコープに含めます。呼び出し元ユーザに js_admin_role が必要です。
- user − スコープを、このユーザが実行したジョブまたはこのユーザの代わりに実行されたジョブに限定します。ここで指定するユーザ名が呼び出し元ユーザのサーバ・ユーザ名でない場合は、呼び出し元ユーザに js_admin_role が必要です。

- owner − スコープを、このサーバ・ユーザ名を使用して実行されたジョブに限定します。ここで指定するユーザ名が呼び出し元ユーザのサーバ・ユーザ名でない場合は、呼び出し元ユーザに js_admin_role が必要です。
- age − スコープを、この日数より前に記録されたジョブに限定します。
- minsize − スコープを、出力バイト数が minsize を超えるジョブに限定します。
- force − 実行中のジョブについても履歴と出力の削除を許可します。

**戻り値**

成功した場合は 0 を、それ以外の場合はエラー・コードを返します。

**例**

**例 1** 次の例は、呼び出し元ユーザのために実行された “orders_processed_report” というジョブの簡潔な履歴をリストする方法を示します。

```
sp_sjobhistory @name='jname=orders_processed_report',
@option='list_short'
```

**例 2** 次の例は、12389 という runid で実行されたスケジュール・ジョブの履歴とジョブ出力を削除する方法を示します。

```
sp_sjobhistory @name='runid=12389', @option='drop'
```

**例 3** 次の例は、呼び出し元ユーザのジョブ履歴のうち、ジョブ出力が 10,000 バイトを超えるものをリストする方法を示します。

```
sp_sjobhistory @option='list,minsize=10000'
```

**例 4** 次の例は、ジョブ出力が 20,000 バイトを超える、呼び出し元ユーザのジョブ履歴とジョブ出力をすべて削除する方法を示します。

```
sp_sjobhistory @option='drop,minsize=20000'
```

**例 5** 次の例は、任意のユーザによって実行された “load_sales_data” の履歴をリストする方法を示します。

```
sp_sjobhistory @name='jname=load_sales_data',
@option='list,all_users'
```

**例 6** 次の例は、“mary” というユーザが実行したすべてのジョブの簡潔な履歴をリストする方法を示します。

```
sp_sjobhistory @option='list_short,user=mary'
```

**使用法**

- ジョブ履歴やジョブ出力のログが誤って削除されないようにするために、フィルタ引数を指定せず drop オプションを指定して sp_sjobhistory を呼び出すと、エラーが発生します。その場合、フィルタ引数を少なくとも 1 つ指定してください。
- sp_sjobhistory は、現在実行中のジョブについてジョブ出力ログのエントリを削除しません。runid 引数で実行中のジョブのエントリを指定しない限り、これらのエントリは削除プロセスで無視され、メッセージも表示されません。ただし、実行中のジョブのエントリを指定した場合、sp_sjobhistory によってエラーが返されます。

次の表に、list 引数を指定した場合の js_history テーブルからの結果を示します。

**表 5-2: list 引数で返される出力**

| カラム | 型 |
|---|---|
| job_runid | int |
| job_name | JS_DESC |
| job_state | char(2) |
| job_end | datetime |
| job_user_code | int |
| job_os_code | int |
| job_user_req | SUSER_NAME |
| job_long_message | JS_LMSG |
| sjob_id | JS_NAME_ID |
| sched_name | JS_NAME_ID |
| job_start | datetime |
| job_exit_code | int |
| job_atat_error | int |
| job_user_run | SUSER_NAME |
| job_short_message | JS_SMSG |
| job_size | int |

次の表に、list_short 引数を使用した場合の js_history テーブルからの結果を示します。

**表 5-3: list _引数で返される出力**

| カラム | 型 |
|---|---|
| job_runid | int |
| job_name | JS_DESC |
| job_state | char(2) |
| job_end | datetime |
| job_user_code | int |
| job_size | int |
| sjob_id | JS_NAME_ID |
| sched_name | JS_NAME_ID |
| job_start | datetime |
| job_exit_code | int |
| job_atat_error | int |

次の表に、list_output 引数を使用した場合の js_history テーブルからの結果を示します。

**表 5-4: list 引数で返される出力**

| カラム | 型 |
|---|---|
| job_runid | int |
| job_size | int |
| job_name | JS_DESC |
| job_output | JS_OUTPUT |

第 6 章

# Sybase Central の ASE プラグインでの Job Scheduler の管理

この章では、Sybase Central グラフィカル管理ツールの ASE プラグインで Job Scheduler を使用する方法について説明します。

**注意** スケジュール・ジョブを実行するには、あらかじめ「Job Scheduler ユーザの設定」(9 ページ ) と「ターゲット・サーバへのアクセスの設定」(10 ページ) に記載されている設定作業を完了してください。



## スケジュール・ジョブの追加

❖ **スケジュール・ジョブの追加**

1 ジョブを管理する Adaptive Server の [定期ジョブ] フォルダを選択します。

2 ウィンドウ上部の [定期ジョブ] ボタンをクリックします。スケジュール・ジョブ作成ウィザードが表示されます。

3 ウィザードで指示される手順に従います。

**注意** スケジュール・ジョブの名前を作成するときは、必ず英字で始めてください。数字で始まる名前を作成すると、エラーが発生します。

**警告！** ジョブをスケジューリングするときは、ASE Job Scheduler がインストールされているサーバの時間を基準にしてください。ジョブは、該当のサーバの時間に従って実行されます。

Job Scheduler テンプレートからスケジュール・ジョブを作成できます。Job Scheduler テンプレートを使用するには、最初にテンプレートをインストールします。テンプレートの詳細については、「第 3 章 テンプレートを使用したジョブの作成」を参照してください。

# ジョブの追加

❖ **ジョブの追加**

1 ジョブを追加する Adaptive Server の [定期ジョブ] フォルダを選択します。

2 [ジョブ] フォルダをダブルクリックします。

3 ウィンドウ上部の [ジョブ] ボタンをクリックします。ジョブ作成ウィザードが表示されます。

4 ウィザードで指示される手順に従います。

**注意** ジョブの名前を作成するときは、必ず英字で始めてください。数字で始まる名前を作成すると、エラーが発生します。

Job Scheduler テンプレートからスケジュール・ジョブを作成できます。Job Scheduler テンプレートを使用するには、最初にテンプレートをインストールします。テンプレートの詳細については、「第 3 章 テンプレートを使用したジョブの作成」を参照してください。

## スケジュールの追加

❖ **スケジュールの追加**

1. スケジュールを追加する Adaptive Server の [定期ジョブ] フォルダを選択します。
2. [スケジュール] フォルダをダブルクリックします。
3. ウィンドウ上部の [ スケジュール ] ボタンをクリックします。スケジュール作成ウィザードが表示されます。
4. ウィザードで指示される手順に従います。

**注意** スケジュールの名前を作成するときは、必ず英字で始めてください。数字で始まる名前を作成すると、エラーが発生します。

## 既存のジョブのスケジューリング

❖ **既存のジョブのスケジューリング**

1. 既存のジョブをスケジューリングする Adaptive Server の [定期ジョブ] フォルダを選択します。
2. [ジョブ] フォルダをダブルクリックします。
3. スケジュールするジョブを選択します。
4. [ファイル] - [スケジュール] を選びます。Job Scheduler ウィザードが表示されます。
5. ウィザードで指示される手順に従います。

## ジョブの履歴の表示

❖ **スケジュール・ジョブの履歴の表示**

1. 履歴を表示する Adaptive Server の [定期ジョブ] フォルダを選択します。
2. [ジョブ履歴] フォルダをダブルクリックします。
3. スケジュール・ジョブの履歴を表示します。
4. ウィザードで指示される手順に従います。

# ジョブの履歴の消去

❖ **ジョブ履歴の消去**

1 履歴を消去する Adaptive Server の [定期ジョブ] フォルダを選択します。

2 [ジョブ履歴] フォルダをダブルクリックします。

3 [消去の履歴] をダブルクリックします。ジョブ履歴消去ウィザードが表示されます。

4 ウィザードで指示される手順に従います。

# Job Scheduler の管理

❖ **Job Scheduler の管理**

1 ジョブを管理する Adaptive Server の [定期ジョブ] フォルダを右クリックします。[Job Scheduler 管理] ダイアログ・ボックスが開きます。

2 [タスク設定] タブでは、次のことができます。

- Job Scheduler Task が実行中かどうかを確認する。
- Job Scheduler タスクの開始または停止。
- [**ジョブを終了する**] プロパティを設定する。このプロパティを設定すると、何秒後にジョブを終了するかを指定するプロパティを設定できます。
- 起動時に Job Scheduler を有効にするプロパティを設定する。
- Job Scheduler 間隔の分単位での設定。
- 同時ジョブの最大数の設定。
- ジョブの最大出力サイズを設定する。

3 [データベース設定] タブでは、次のことができます。

- ジョブの履歴用に使用するデータベースの割合の指定。クリーンアップ・タスクが実行されると、ここで指定した領域が確保されるまで古いレコードが削除されます。
- 出力用に使用するデータベースの割合の指定。
- 履歴用に割り付けるために確保する空き領域の割合の指定。
- 出力用に割り当てた領域のうち、空けておく割合を定義する。

## すべてのユーザの表示

❖ **すべてのユーザの表示**

1. ユーザを表示する Adaptive Server の [定期ジョブ] フォルダを選択します。
2. 表示するユーザに対応するフォルダをダブルクリックします。
   a. [ジョブ] フォルダをダブルクリックします。すべてのユーザのジョブが右ウィンドウ枠に表示されます。
   b. [スケジュール] フォルダをダブルクリックします。すべてのユーザのスケジュールが右ウィンドウ枠に表示されます。
   c. [ ジョブ履歴 ] フォルダをダブルクリックします。すべてのユーザのジョブの履歴が右ウィンドウ枠に表示されます。

## スケジュール・ジョブの管理

❖ **スケジュール・ジョブの管理**

1. スケジュール・ジョブを管理する Adaptive Server の [定期ジョブ] フォルダを選択します。
2. 管理するジョブのジョブ ID を右クリックします。スケジュール・ジョブの管理用のサブメニューが表示されます。
3. 次の管理オプションのいずれかを選択します。
   - [ 無効化 ] − スケジュール・ジョブを一時的に無効にする。エントリは [ スケジュール・ジョブ ] リストに残り、ジョブとスケジュールの情報もそのまま残りますが、無効のフラグが付きます。スケジュール・ジョブの実行は次回以降、ジョブが再び有効化されるまで中断されます。実行中のジョブは、完了するまで続行されます。
   - [有効化] − スケジュール・ジョブを有効にする。
   - [ いますぐ実行 ] − ジョブを直ちに実行します。ただし、通常のスケジュールによって既に実行されている場合は除きます。
   - [終了] − 実行中のジョブを強制終了します。
   - [削除] − ジョブをスケジュール・ジョブから削除する。

# プロパティの編集

❖ **ジョブ、スケジュール、スケジュール・ジョブのプロパティの変更**

1 プロパティを編集する Adaptive Server の [定期ジョブ] フォルダを選択します。

2 プロパティの編集対象のスケジュール・ジョブ、ジョブ、またはスケジュールを右クリックします。サブメニューが表示されます。

3 サブメニューから [プロパティ ] を選択します。選択したオブジェクトのプロパティ・シートが表示されます。

スケジュール・ジョブについて編集できるプロパティは次のとおりです。

- 所有者
- ターゲット ASE
- タイムアウト (分単位)
- ログ出力の有効化／無効化
- 障害発生時に無効にする
- 完了時に削除する

スケジュールについて編集できるプロパティは次のとおりです。

- 名前
- 所有者
- 説明
- 他のユーザによる使用を許可する
- 開始時刻を設定する
- 開始日付を設定する
- 終了日付を選択しない
- 終了日付を設定する
- 反復を設定する

**注意** スケジュールの変更は、そのスケジュールを使用しているすべてのジョブに影響します。

ジョブについて編集できるプロパティは次のとおりです。

- 名前
- 所有者
- 説明
- 使用テンプレート
- ジョブ定義
- 複数ジョブの同時実行を許可する (ジョブの前回の実行がまだ完了していないときに、次の実行を開始できるかどうかを指定します)
- 他のユーザによる実行を許可する
- ジョブ所有者による実行のみを許可する
- タイムアウト (分単位)

# Job Scheduler オブジェクトの削除

❖ **Job Scheduler オブジェクトの削除**

1. Adaptive Server の [定期ジョブ] フォルダを選択し、削除するスケジュール・ジョブ、スケジュール、またはジョブを選択します。
2. メニュー・バーから [編集] - [削除] を選びます。Job Scheduler オブジェクトが削除されます。

---

**注意** 現在実行されるようにスケジュールされているジョブまたはスケジュールは削除できません。

---

第 7 章

# トラブルシューティング

この章では、スケジュール・ジョブの作成時と実行時のエラー・ロギングとトラブルシューティングについて説明します。



## エラー・メッセージのログ

新しい Adaptive Server を設定すると、インストール・プログラムが自動的にエラー・ログのロケーションを設定します。その後、Adaptive Server は、起動されるたびにローカル・エラー・ログ・ファイルに起動情報を書き込みます。Job Scheduler をインストールすると、JS Agent が独自のログ・ファイルを同じディレクトリに作成します。エラー・ログのデフォルトのロケーションとファイル名については、使用するプラットフォームの『設定ガイド』を参照してください。

Adaptive Server からのエラー・メッセージの多くはユーザの端末にだけ表示されます。ただし、致命的なエラー・メッセージ (重大度レベル 19 以上)、カーネル・エラー・メッセージ、Adaptive Server からの情報メッセージはエラー・ログ・ファイルに記録されます。

Adaptive Server は、サーバ・プロセスが停止されるまではエラー・ログ・ファイルをオープンにした状態に保ちます。古くなったメッセージを削除してエラー・ログのサイズを減らすには、その前に Adaptive Server プロセスを停止してください。

---

**注意** Windows NT など一部のプラットフォームでは、Adaptive Server はオペレーティング・システムのイベント・ログにもエラー・メッセージを記録します。エラー・ログの詳細については、Adaptive Server のインストールと設定についてのガイドを参照してください。

---

## ジョブがスケジューリングした時間に実行されない

スケジュールやスケジュール・ジョブを作成するときは、ASE Job Scheduler がインストールされているサーバの時間を使用してください。自身のコンピュータのローカル時間を使用してジョブをスケジューリングした場合、その時間は Job Scheduler がインストールされているサーバの時間と必ずしも一致するとは限りません。

## テンプレートから作成したスケジュール・ジョブを実行できない

テンプレートから作成したスケジュール・ジョブが実行できない場合は、ジョブ履歴にエラー・メッセージがないか調べてください。ジョブの実行が失敗すると、その理由がジョブ履歴に記録されます。

テンプレートで使用されているストアド・プロシージャを、ジョブを実行するターゲット・サーバにインストールしていない場合は、次のようなメッセージが記録されています。

```
Procedure 'dump_databases' not found.  Specify
owner.objectname or use sp_help to check whether the object
exists (sp_help may produce lots of output).
```

詳細については、「ターゲット・サーバへのストアド・プロシージャのインストール」(14 ページ) を参照してください。

## sp_sjobcmd を複数回呼び出すジョブを実行できない

sp_sjobcmd を複数回呼び出すジョブを作成する場合は、それぞれの呼び出しの後で改行してください。

```
go
```

## ストアド・プロシージャを実行できない

Job Scheduler は、ジョブを開始するときに master データベースで開始します。ジョブで別のデータベース上のストアド・プロシージャを呼び出す場合、そのストアド・プロシージャが存在するデータベースの名前をプレフィックスとして追加することが必要な場合があります。

# 索引

## T



## W



## X



## あ



## い



## う



## え



## お



## か



## き



## こ



## さ

## し



## す



## せ



## た



## つ



## て

## と



## ふ



## め



## や



## ゆ



## り



## れ



## ろ