Actian Zen v16 の主な新機能

株式会社エージーテック

2024 年 12 月 20 日

免責事項

株式会社エージーテックは本書の使用を、利用者またはその会社に対して「現状のまま」でのみ許諾するものです。株式会社エージーテックは、いかなる場合にも本書に記載された内容に関するその他の一切の保証を、明示的にも黙示的にも行いません。本書の内容は予告なく変更される場合があります。

商標

Actian、Actian DataCloud、Actian DataConnect、Actian X、Avalanche、Versant、PSQL、Actian Zen、Actian Director、Actian Vector、DataFlow、Ingres、OpenROAD、および Vectorwise は、Actian Corporation およびその子会社の商標または登録商標です。本資料で記載される、その他すべての商標、名称、サービスマークおよびロゴは、所有各社に属します。

Actian Zen v16 の主な新機能

最終更新：2024 年 12 月 20 日

Actian Zen v16 では主に次の機能が追加されています。

◆Zen エンジンが無い環境で実行可能なリビルド・ユーティリティ
◆1024 バイトまで拡張されたキー
◆Btrieve API に対応したユニバーサルトリガー
◆SQL のログ機能
◆簡易レプリケーション
◆CPU コア数に対応したライセンス

本資料では、Actian Zen v16 で追加された主な機能について概要および使い方について説明します。

１． Zen エンジンが無い環境で実行可能なリビルド・ユーティリティ

従来、ファイルのリビルドを行うには、Zen エンジンがインストールされていることが必要でした。
しかしサイズが大きな複数のファイルをリビルドするには、高い負荷がかかり、長い時間が必要でした。
この問題を解決するため、Zen v16 では Zen 製品がインストールされていない環境でもスタンドアロンで実行可能なオフライン・リビルド・ユーティリティが提供されています。
※オフライン・リビルド・ユーティリティは、Zen 製品のインストーラーには含まれず、別途 WEB からダウンロードが必要です。
オフライン・リビルド・ユーティリティを使用すれば、Zen 製品がインストールされていない環境でのファイル再構築が可能となり、複数のマシンで並行して再構築が可能です。
これにより Zen エンジンを実行しているサーバーに負荷を掛けることなく、短時間での再構築が可能となります。

２． 1024 バイトまで拡張されたキー

文字型のキー長が最大 255 から 1024 に拡張されました。
従来、256 バイトを超える文字列項目をキーに設定する場合、部分キーとする必要があり、異なるデータが同じキー値として読み込まれていました。
Zen v16 では、1024 バイトまでのデータを一意に識別することが可能です。
※この機能を使用するには、ファイル形式を 16 形式にする必要があります。

３． Btrieve API に対応したユニバーサルトリガー

データベースにトリガーを設定した場合、Actian Zen v15 までは、トリガーの実行を引き起こすオペレーションを行うと、ステータス 149 でブロックされました。
Actian Zen v16 では、Btrieve API 実行時にもトリガーが実行されるよう拡張されました。

例えば、入退室を記録するテーブル（入退室テーブルとします）と、累積滞在時間を記録するテーブル（累

積滞在時間テーブルとします）があり、入退室テーブルにレコード（退室時のレコード）が追加された際、トリガーで累積滞在時間を加算するトリガーを紹介します。
Actian Zen v15 までは、Btrieve API でレコードの追加を行うと、ステータス 149 が返りレコードの追加が失敗していましたが、Actian Zen v16 では Btrieve API でレコードの追加ができ、かつトリガーにより累積滞在時間の加算も行えるようになりました。
この例では、次のようなテーブルを使用します。

入退室テーブル

項目名	データ型	長さ	備考
レコード番号	BigIdentity	8	
ID	Integer	4	
入退室時間	TimeStamp2	8	
入退室区分	Integer	4	1：入室　2：退室
入室中フラグ	Integer	4	0：入室中 1：退室済

滞在時間テーブル

項目名	データ型	長さ	備考
ID	BigIdentity	8	
Staytime	Integer	4	秒単位の累積滞在時間

次の SQL 文で検証いただくための環境が構築できます。

```
Create Table iot (
    rno Bigidentity,
    ID Integer not null,
    inouttime TIMESTAMP2 not null,
    inouttype Integer not null,
    Flg Integer not null);

Create Table staytime (
    ID Integer not null,
    staytime Integer not null);

insert into staytime values(1,0);
insert into staytime values(2,0);
insert into staytime values(3,0);
insert into staytime values(4,0);
insert into staytime values(5,0);
```

```
Create Trigger inoutIns
    after Insert on iot
    for each row
    begin
      declare :stime Integer;
      declare :starttime timestamp;
      declare :endtime timestamp;
      update iot set inouttime = NOW() where rno = new.rno;
      if (new.inouttype = 2) then
          select staytime into :stime from staytime where id = new.id;
          select inouttime into :starttime from iot where id = new.id and inouttype = 1 and flg = 0;
          set :endtime = NOW();
          set :stime = :stime + datediff(second,:starttime,:endtime);
          update staytime set staytime = :stime where id = new.id;
          update iot set flg = 1;
      end if;
    end;
```

iot テーブルは、入退室のデータを保存します。

staytime テーブルは、ID 毎の累積滞在時間を保存します。

inoutIns トリガーは iot テーブルにレコードが追加された際に実行され、inouttime に現在時刻を登録します。（Btrieve API でタイムスタンプ型のデータを登録するのは大変なため、トリガーで設定します）
データが退室（inouttype ＝ 2）だった場合、入室時の時間との差から滞在時間を求め、staytime テーブルの累積滞在時間（staytime）に加算します。
また、入室中のフラグを 1 に変更します。

staytime テーブルは、あらかじめ使用する ID と累積滞在時間の初期値の設定が必要です。例では、ID が 1 から 5 までのレコードを設定しています。
テスト時の ID は、1 から 5 を指定します。

テストは Function Executer で iot テーブルに入退室データを登録することで行います。

入室時のデータは次のような内容を挿入します。

rno	ID	入退室時間	入退室区分	フラグ
0	1 ～ 5 のいずれか	0	1	0

※rno は、Identity 型なので、0 を設定することで自動的に最大値 + 1 の値が設定されます。

退室時のデータは次のような内容を挿入します。

rno	ID	入退室時間	入退室区分	フラグ
0	入室時と同じ値	0	2	0

フラグは、0 が入室中であることを示します。（退室時のフラグは使用しません）

入室データと退室データを入力すると、staytime テーブルに滞在時間が加算されます。

４． SQL のログ機能

Actian Zen v15 までは、ODBC アクセスでのみ SQL 実行時のログを採取できましたが、Actian Zen v16 では、使用するインターフェイスに関わらず、ログの採取が可能となりました。
この機能により、プログラムで生成した SQL 文の確認が簡単に可能となります。
SQL の実行時間を確認するのにも役に立ちます。

```
================================================================================
System User:                Administrator
DB User:                    Master
DB Name:                     DEMODATA
Client Host:                Zenv16Server
Network Address:            Shared Memory
SRDE Task Number:             32781
Session Start Time:         2024-07-29T13:44:55.324
================================================================================
2024-07-29T13:44:58.777: Start:         SQLPrepare                      Handle: 0x272ff9e0f80
SQL Query text:              "select * from "Billing""
2024-07-29T13:44:58.996: End:           SQLPrepare          Retcode:0   Handle: 0x272ff9e0f80
================================================================================
2024-07-29T13:44:58.996: Start:         SQLExecute                      Handle: 0x272ff9e0f80
2024-07-29T13:44:58.996: End:           SQLExecute          Retcode:0   Handle: 0x272ff9e0f80
================================================================================
```

５． 簡易レプリケーション

Actian Zen v16 には、簡易レプリケーションを行うための、ツールが付属しています。
このツールは、Insert と Update オペレーションに対応し、簡単な設定で片方向のレプリケーション機能を提供します。
また、Server Edition に限らず Workgroup でも使え、プラットフォームも限定されません。
Delete を含めてレプリケーションが必要なケースで、Windows 用の Server Edition をご使用であれば、Data Exchenge をご使用ください。

簡易レプリケーション機能では、JSON ファイルに複製元と複製先の情報を指定します。

```
{
        "version":      1,
        "settings":     {
                "log_file":     "c:/test/easysync.log",
                "log_level":    "verbose",
                "polling_interval_sec":     10,
                "resume_on_error":      true
                },
        "files":    [{
                "source_file":      "btrv:///synctest?dbfile=住所録.mkd",
                "source_username_enc":    '',
                "source_password_enc":    '',
                "destination_file":     "btrv://bkserver/synctest?dbfile=住所録.mkd",
                "destination_username_enc":     '',
                "destination_password_enc":     '',
                "unique_key":     0,
                "create_destination":       true,
                "_last_copied_record_timestamp":  "0000000000000000",
                "_last_existing_transaction_time": "0000000000000000",
                "_last_file_timestamp":    "0000000000000000"
                }]
}
```

コマンドプロンプトで次のコマンドを実行することで、同期が行えます。

Easysync JSON ファイル

◆注意点◆

・複製先で Insert、Update、Delete を行っても複製元には反映されません。

・複製元で Delete を行っても、複製先には反映されません。

・ツール実行時、複製先にファイルが存在するとステータス 5 が発生します。

・ファイルには SystemData2 が設定されていることが必要です。

６． CPU コア数に対応したライセンス

Actian Zen v16 では、Cloud Server Edition に、ファイルサイズにより制限されるライセンスに加え、CPU の物理コア数に対応したライセンスが追加されました。これによりシステム納品後のサイズ拡大を心配する必要が無くなります。

注意点としては、適用するライセンスが物理 CPU コア数以上であることが必要なことです。物理 CPU コア数よりライセンスのコア数が少ない場合、ライセンスが無効状態となります。

Cloud Server Edition のライセンスは 4 コア単位で切り上げとなります。

例えば、インテル® Xeon® E-2336 CPU の場合、6 コア ですから、8 コア分のライセンスが必要となります。

※仮想マシンでは、仮想マシンに割り当てられた論理コア数以上のライセンスが必要となります。