# アサーション・エラー

# 50213 または 200601 dbunload 固定プロシージャ

テクニカル・サポートに報告されるデータベース破損のタイプの 1 つは、主にアサーション・エラー 50213 "Page number on page does not match page requested." に関連しています。Adaptive Server Anywhere 6、7、または 8 で実行している場合には、アサーションは 200601 "Page for requested record not a table page or record not present on page." になる可能性があります。ただし、最も報告が多かったのは、バージョン 5.x で作成されたデータベースがソフトウェアのアップグレードの前に破損してしまうということでした。

最初に失敗した操作によっては、別のアサーションが報告されることもあります。

どのようなデータベース破損の場合でも、有効なバックアップを見つけてバックアップ時以降のログ・ファイルをすべて適用することが望ましい解決法となります。Adaptive Server Anywhere 8.0.x でのこのプロセスの詳細は、『Adaptive Server Anywhere データベース管理ガイド』の第 11 章「バックアップとデータ・リカバリ」を参照してください。

影響を受けているデータベース・ページをアクセスした場合にのみ、このような特殊なタイプの破損が明らかになるため、特に厄介です。そのページのデータをめったに使用しない場合は、お客様の手元にあるバックアップの多数または全部で破損が生じている可能性が十分にあります。

一部のデータが失われる可能性はありますが、次の手順でデータベースをリカバリできることもあります。リカバリしたデータベースのデータの有効性と完全性を確認するのは、データベース管理者であるお客様の責任となります。

この作業では dbunload コマンドを使用します。dbunload のオプションを確認するには、コマンド・プロンプトで、dbunload /? と入力します。次の表は dbunload のオプションのリストとその説明です。表のように指定の小文字を使用します。

| **オプション** | **説明** |
|---|---|
| -ac “keyword=value;…” | 再ロードのデータベース接続パラメータを指定する |
| -an <file> | 新しいデータベースを作成し、再ロードする |
| -ar [log dir] | データベースを再構築して置き換える |
| -c “keyword=value;…” | データベース接続パラメータを指定する |
| -d | データのみアンロードする |
| -e <list> | リストされたテーブルに対するデータ出力なし |
| -ea <alg> | 新しいデータベースに暗号化アルゴリズムを指定する (デフォルトは AES。-ek または –ep が必要) |
| -ek <key> | 新しいデータベースの暗号化キーを指定する (-an を使用) |

| | |
|---|---|
| -ep | 新しいデータベースの暗号化キーの入力を求めるプロンプトを表示する(-an を使用) |
| -ii | 内部アンロード、内部再ロード (デフォルト) |
| -j <count> | ビュー作成文の繰り返し回数を指定する |
| -jr | アンロードまたは再ロードの時に、Java を無視する |
| -m | レプリケートするデータベースのユーザ ID を保存しない |
| -n | データなし ― スキーマ定義のみ |
| -o <file> | 出力メッセージのログをファイルに取る |
| -p <char> | エスケープ文字(デフォルトは ¥ (バックスラッシュ)) |
| -q | クワイエット：メッセージの出力やウィンドウの表示を行わない |
| -r <file> | 生成された再ロード Interactive SQL コマンド・ファイルの名前を指定する (デフォルトは reload.sql) |
| -t <list> | リストされたテーブルだけをアンロードする |
| -u | データを順序付けしない |
| -v | 冗長メッセージ |
| -xi | 外部アンロード、内部再ロード |
| -xx | 外部アンロード、外部再ロード |
| -y | 確認しないでコマンド・ファイルを置き換える |

**注意：**外部アンロードを使用しない場合は、エンジン (またはサーバ) にとって意味のあるパスとして <directory> を指定してください。

このドキュメントでは、データベースの再構成の方法として、3 ステップの手動プロセスを説明します。dbunload オプションで再構成のフェーズとして組み込まれている、-ac、-an、-ar オプションはデータ修復作業には適応しません。例では、Adaptive Server Anywhere 8 に付属している asademo.db サンプル・データベースを使用しています。

1. 作業ディレクトリ (たとえば c:¥working) とアンロード・ディレクトリ (たとえば c:¥unload) を作成します。これらのディレクトリをルートから分離して配置すると、作業が容易になります。データベース・ファイルとログ・ファイルを作業ディレクトリにコピーします。作業ディレクトリでコマンド・プロンプトを開きます。次のコマンド・ラインを実行してください (すべて 1 行に入力)。

   ```
   dbunload -c "uid=dba;pwd=sql;dbf=c:¥working¥asademo.db" -u c:¥unload
   ```

   出力は次のようになります。

   ```
   Adaptive Server Anywhere Unload Utility Version 8.0.2.3601
   Unloading "rs_systabgroup"."rs_lastcommit" into c:¥unload¥376.dat
   (relative to server)
   ```

```
Unloading "rs_systabgroup"."rs_threads" into c:¥unload¥377.dat (relative
to server)
Unloading "dbo"."ul_file" into c:¥unload¥378.dat (relative to server)
Unloading "dbo"."ul_statement" into c:¥unload¥379.dat (relative to
server)
Unloading "dbo"."ul_variable" into c:¥unload¥380.dat (relative to server)
・・・
```

テクニカル・サポートの事例では、この時点で上記例にアサーション・エラー・メッセージが表示されます。customer テーブルによってアサーション・エラーが発生したと仮定します。

2. 次のコマンドを入力します (すべて 1 行に入力)。

```
dbunload -c "uid=dba;pwd=sql;dbf=c:¥working¥asademo.db" -u c:¥unload -e
dba.customer
```

出力は次のようになります。

```
C:¥workingdbunload -c "uid=dba;pwd=sql;dbf=c:¥working¥asademo.db" -u
c:¥unload -e dba.customer
Adaptive Server Anywhere Unload Utility Version 8.0.2.3601
Unloading "rs_systabgroup"."rs_lastcommit" into c:¥unload¥376.dat
(relative to server)
Unloading "rs_systabgroup"."rs_threads" into c:¥unload¥377.dat (relative
to server)
Unloading "dbo"."ul_file" into c:¥unload¥378.dat (relative to server)
Unloading "dbo"."ul_statement" into c:¥unload¥379.dat (relative to
server)
Unloading "dbo"."ul_variable" into c:¥unload¥380.dat (relative to server)
・・・
Unloading "DBA"."employee" into c:¥unload¥417.dat (relative to server)
Overwrite file "reload.sql"? (Y/N) y
```

このプロセスは、正常に完了するか、または別の破損テーブルを生成します。アンロードが正常に完了するまで、除外するテーブルのカンマ区切りリストを使用してプロセスを再実行してください。

3. 破損したテーブルを除き、必要なデータを含む一連の *.DAT ファイルと、データベースのスキーマを含む reload.sql というファイルが生成されました。このデータベースの完全なスキーマが必要となります。RELOAD1.SQL を作成するためには、次のように入力します (すべて 1 行に入力)。

```
dbunload" -c "uid=dba;pwd=sql;dbf=c:¥working¥asademo.db" -n -r
RELOAD1.SQL
```

4. 任意のテキスト・エディタ (メモ帳など) を使用して reload.sql と RELOAD1.SQL を開き、”RELOAD DATA (データの再ロード)” を検索します。このセクションの内容全体を reload.sql から RELOAD1.SQL の同じセクションにコピーします。

   RELOAD DATA セクションには、次のような LOAD TABLE パラグラフが含まれます。

   ```
   LOAD TABLE "DBA"."employee"
   FROM 'c:¥¥unload¥¥417.dat'
   FORMAT 'ASCII'
   QUOTES ON ESCAPES ON STRIP OFF
   CHECK CONSTRAINTS OFF COMPUTES OFF
   DELIMITED BY ','
   go
   ```

   細かい部分は、アンロードを実行するエンジンのバージョンによって若干異なります。

5. 破損したテーブルごとに、これらのパラグラフのいずれかを複製します。テーブル名 ("DBA"."CONTACT") を、破損したテーブルの名前 ("DBA"."CUSTOMER") に置き換えてください。また、DAT ファイルの名前を、同じディレクトリの一意のファイル名 (たとえば 'c:¥¥unload¥¥customer.dat') に置き換えます。

   ```
   LOAD TABLE "DBA"."customer"
   FROM 'c:¥¥unload¥¥customer.dat'
   FORMAT 'ASCII'
   QUOTES ON ESCAPES ON STRIP OFF
   CHECK CONSTRAINTS OFF COMPUTES OFF
   DELIMITED BY ','
   go
   ```

6. RELOAD1.SQL ファイルを保存します。

7. 破損したテーブルから有効なデータを回復する必要があります。スタンドアロン・エンジンでデータベースを起動し、Interactive SQL から接続します。次の構文を使用して、テーブルのデータを選択して出力します。便宜上、データはプライマリ・キー順になります。

   ```
   SELECT * FROM customer ORDER BY id ASC >># c:¥unload¥customer.dat
   ```

   これにより、エンジンがアサートするまで、できるかぎり多くのデータがテーブルから customer.dat に出力されます。

8. エンジンを再起動し、Interactive SQL から再接続して、ソート順を反転します。

```
SELECT * FROM customer ORDER BY id DESC >># c:¥unload¥customer.dat
```

二重の > 記号を使用すると、新しい出力が最初のセットに結合されます。エンジンは、再度アサートします。書き込まれたローの数を期待数と比較するか、customer.dat を開いて最初のパスの最終ローのプライマリ・キー値と 2 番目のパスの最終ローのプライマリ・キー値を比較して、どれだけのデータが失われているかを確認します。ある程度のデータが失われるのは避けられませんが、かなりの数のローが失われている場合は、次のような構文を使用してプロセスを継続できます。

```
SELECT * FROM customer where id between 1600 and 20000 ORDER BY id ASC
>># c:¥unload¥customer.dat
```

複数のローが破損していて、それらのローの位置が大きく離れている場合は、この処理に時間がかかることがあります。幸いこれらのローはプライマリ・キーで頻繁にグループ化されています。グループ化されていない場合は、インデックスを持つ別のカラムによってソートするか、このテーブルのインデックスをすべて削除してからデータの回復を行います。

**注意:** オプティマイザがテーブルの逐次スキャンを実行するように決定すると、ORDER BY 句を使用していても、常に同じポイントで失敗が起きます。このような時には、ユーザ推定を明示的に指定して、オプティマイザがインデックスを使用するようにします。しかし、必ずオプティマイザがインデックスを使用するかは保証できません。データ回復のもっとも有効な手段としては、適切なバックアップとリカバリ方法をとることです。

9. できるかぎり多くのデータを回復したら、dbinit コマンドを使用して新しいデータベースを作成する必要があります。オリジナルの照合順、ブランク埋め込み設定、大文字と小文字の区別、ページ・サイズに一致していることを確認してください。大量のキャッシュを備えたパーソナル・サーバで新しいデータベースを起動し、Interactive SQL を使用してその新しいデータベースに接続して、次のように入力します。

   ```
   READ C:¥WORKING¥RELOAD1.SQL
   ```

   これにより、元のデータベースから回復できたスキーマ情報とデータがすべて読み込まれます。唯一直面する可能性がある別の問題は、いつ外部キー関係が適用されるかということです。親レコードがないのに子レコードが存在する場合は、外部キー関係が確立されません。これを機能させるには、子レコードを削除するか、親レコードの置換／ダミー作成を行う必要があります。

10. オリジナルとは異なる名前を使用して新しいデータベースを作成した場合は、DOS／エクスプローラを使用してファイルの名前を変更し、dblog を実行してログ・ファイルの名前を変更します。

    ```
    DBLOG -t asademo.log asademo.db
    ```

データが完全で正確であることを確認するのは、お客様の責任となります。