日日是Oracle APEX

Oracle APEXを使った作業をしていて、気の付いたところを忘れないようにメモをとります。

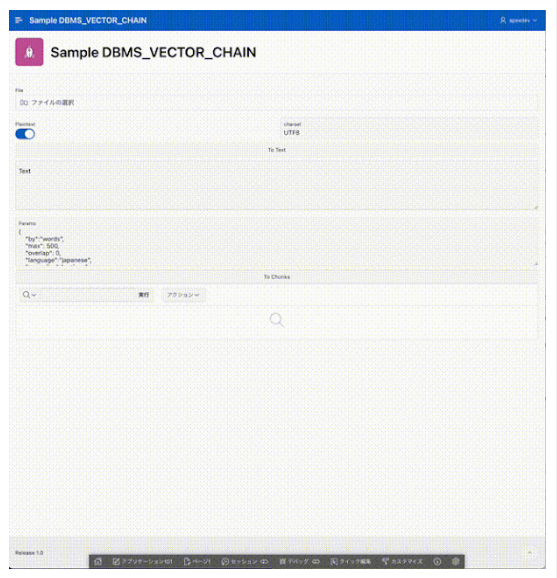
2024年5月14日火曜日

DBMS_VECTOR_CHAINのUTL_TO_TEXTとUTL_TO_CHUNKSの出力を確認する

Always FreeのAutonomous DatabaseとしてOracle Database 23aiを選択できるようになりました。23aiより新しく追加されたパッケージDBMS_VECTOR_CHAINに含まれるUTL_TO_TEXTとUTL_TO_CHUNKSの出力を確認するため、簡単なAPEXアプリケーションを作成してみました。

テキストの取り出しとチャンク分割を行うファイルをアプリケーションにアップロードし、DBMS_VECTOR_CHAINのUTL_TO_TEXTおよびUTL_TO_CHUNKSを呼び出します。UTL_TO_TEXTについては指定可能なパラメータが少ないため、plaintextおよびcharsetをそれぞれ指定できるようにしています。UTL_TO_CHUNKSは色々なパラメータを指定できるため、パラメータをJSONのまま与えるようにしています。

指定できるパラメータについては23aiのSQL Language Referenceの、VECTOR_CHUNKSで説明されています。

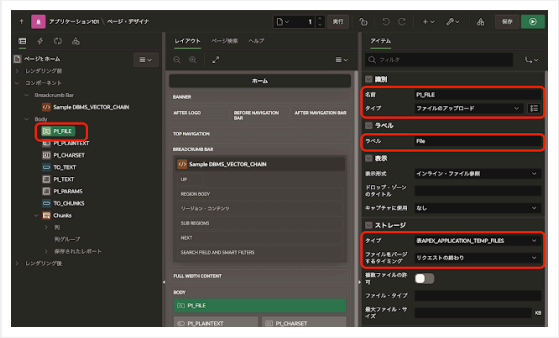


作成したアプリケーションのエクスポートを以下に置きました。
https://github.com/ujnak/apexapps/blob/master/exports/sample-dbms-vector-chain.zip

以下にアプリケーションの実装のポイントを紹介します。

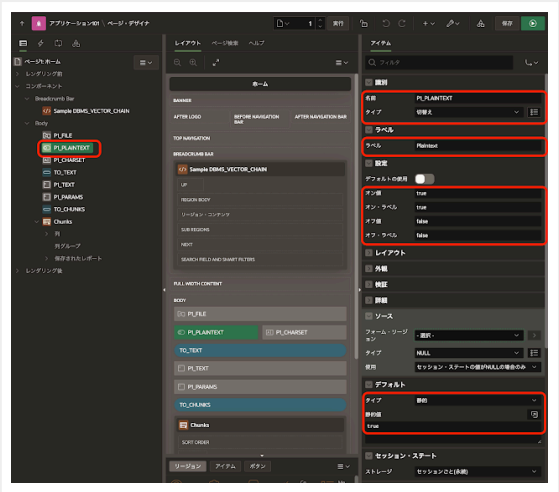
テキストの取り出しおよびチャンク分割の対象とするファイルを選択するページ・アイテムは**P1_FILE**として作成しています。ページ・アイテムの**タイプ**は**ファイルのアップロード**です。

ストレージの**タイプ**に**表APEX_APPLICATION_TEMP_FILES**を選択し、**ファイルをパージするタイミング**は**リクエストの終わり**としています。アップロードしたファイルからテキストが取り出し、ページ・アイテム**P1_TEXT**に保存します。その後はアップロードしたファイルを参照することはないため、リクエストが終了した時点でファイルはパージします。

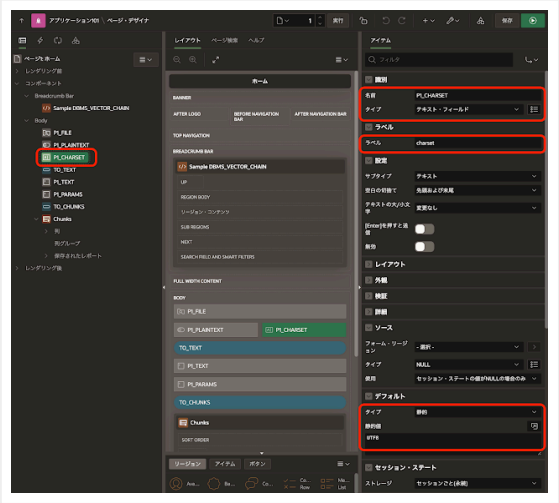


UTL_TO_TEXTのパラメータのひとつである**plaintext**を設定するページ・アイテムを**P1_PLAINTEXT**として作成しています。指定可能な値は**true**または**false**なので、ページ・アイテムの**タイプ**に**切替え**を選択しています。

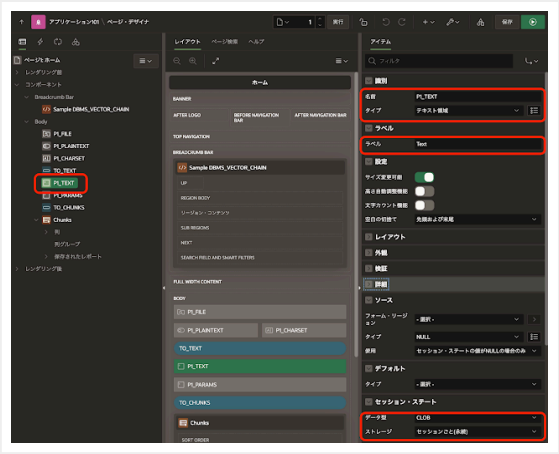
設定の**デフォルトの使用**を**オフ**にし、**オン値**にtrue、**オフ値**にfalseとしています。**デフォルト**は**true**です。



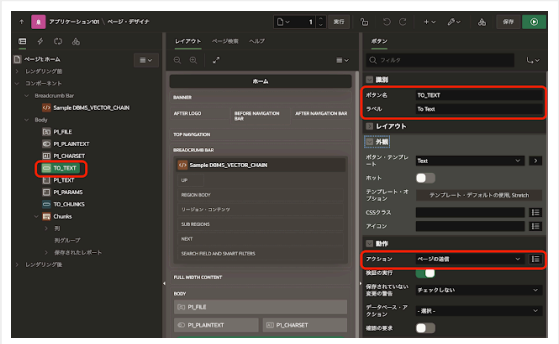
UTL_TO_TEXTに与えることができるもうひとつのパラメータである**charset**は、ページ・アイテム**P1_CHARSET**で指定します。ただし、このパラメータは**UTF8**以外受け取らないため、**タイプ**を**テキスト・フィールド**、**デフォルト**を**UTF8**としています。



UTL_TO_TEXTを呼び出した結果を保持するページ・アイテムを**P1_TEXT**として作成しています。結果はCLOBであるため、**セッション・ステート**の**データ型**として**CLOB**を選択します。また、**ストレージ**は**セッションごと(永続)**を選択し、この後の処理から参照できるようにします。



UTL_TO_TEXTを呼び出すボタン**TO_TEXT**を作成しています。**動作**の**アクション**は**ページの送信**です。



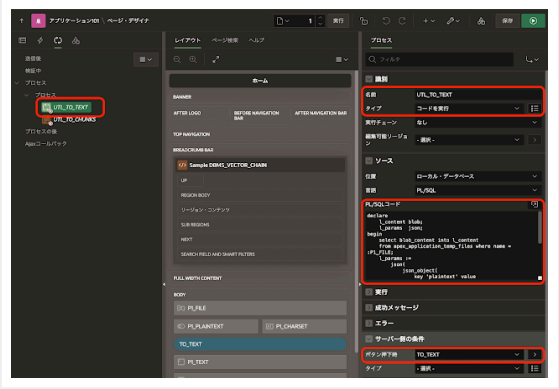
ボタンTO_TEXTを押したときに実行されるプロセスをUTL_TO_TEXTとして作成しています。

実行するコードは以下です。DBMS_VECTOR_CHAIN.UTL_TO_TEXTを呼び出しています。ドキュメントのExampleでは、"plaintext": "true" との指定が記述されていますが、このコードでは"plaintext": true （文字列ではなくboolean）としています。

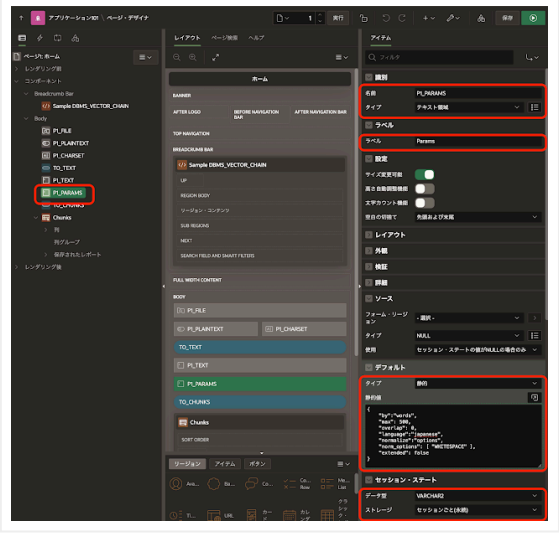
```
declare
    l_content blob;
```

```
    l_params  json;
begin
    select blob_content into l_content
    from apex_application_temp_files where name = :P1_FILE;
    l_params :=
        json(
            json_object(
                key 'plaintext' value to_boolean(:P1_PLAINTEXT)
                ,key 'charset'  value :P1_CHARSET
            )
        );
    apex_debug.info('params for utl_to_text: %s', json_serialize(l_params));
    :P1_TEXT := dbms_vector_chain.utl_to_text(
        data => l_content
        ,params => l_params
    );
end;
```

sample_utl_to_text.sql hosted with ❤️ by GitHub view raw



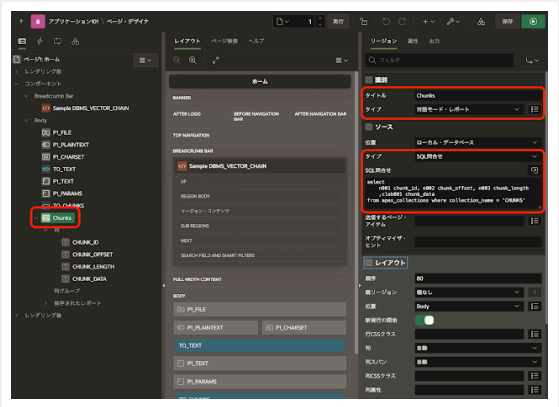
UTL_TO_CHUNKSの呼び出しの際に与えるパラメータを指定するページ・アイテムを**P1_PARAMS**として作成しています。ページ・アイテムの**タイプ**を**テキスト領域**とし、JSONを記述するようにしています。



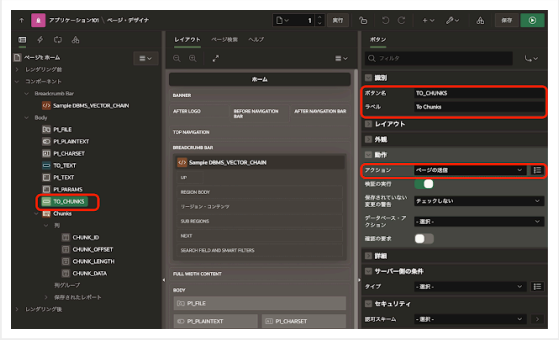
UTL_TO_CHUNKSを呼び出し分割されたチャンクは、対話モード・レポートで表示します。チャンク自体はAPEXコレクションに保存します。

リージョンのソースとして以下のSQLを記述します。

```
select
   n001 chunk_id, n002 chunk_offset, n003 chunk_length
   ,clob001 chunk_data
from apex_collections where collection_name = 'CHUNKS'
```



UTL_TO_CHUNKSを呼び出すボタンTO_CHUNKSを作成しています。**動作**の**アクション**は**ページの送信**です。



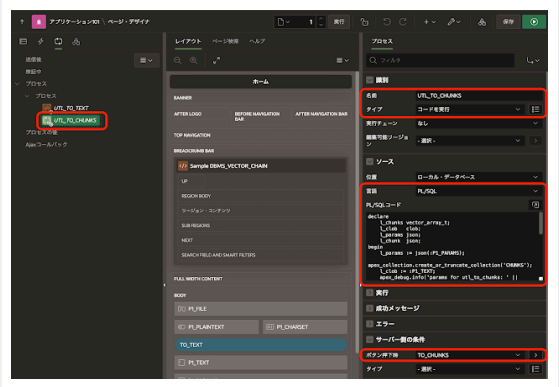
ボタンTO_CHUNKSを押したときに実行されるプロセスをUTL_TO_CHUNKSとして作成しています。

実行するコードは以下です。DBMS_VECTOR_CHAIN.UTL_TO_CHUNKSを呼び出しています。ドキュメントのExampleでは、"max": "100", "overlap":"0" との指定が記述されていますが、本来は"max": 100, "overlap": 0（文字列ではなく数値）ではないかと思います。

```
declare
    l_chunks vector_array_t;
    l_clob   clob;
    l_params json;
    l_chunk  json;
begin
    l_params := json(:P1_PARAMS);
    apex_collection.create_or_truncate_collection('CHUNKS');
    l_clob := :P1_TEXT;
    apex_debug.info('params for utl_to_chunks: %s', json_serialize(l_params));
```

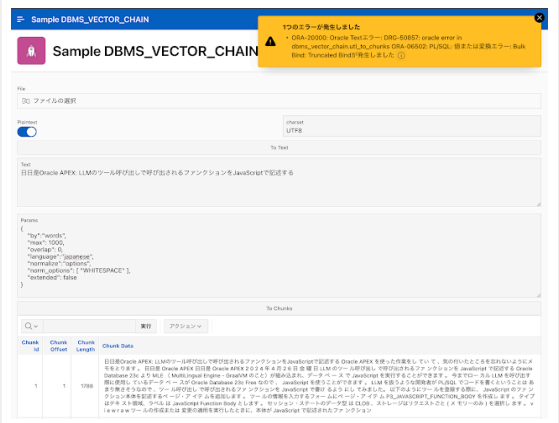
```
    l_chunks := dbms_vector_chain.utl_to_chunks(
        data => l_clob
        ,params => l_params
    );
    for i in l_chunks.FIRST .. l_chunks.LAST
    loop
        l_chunk := json(l_chunks(i));
        apex_collection.add_member(
            p_collection_name => 'CHUNKS'
            ,p_n001 => json_value(l_chunk, '$.chunk_id'      returning number)
            ,p_n002 => json_value(l_chunk, '$.chunk_offset'  returning number)
            ,p_n003 => json_value(l_chunk, '$.chunk_length'  returning number)
            ,p_clob001 => json_value(l_chunk, '$.chunk_data' returning clob)
        );
    end loop;
end;
```

sample_utl_to_chunks.sql hosted with ❤️ by GitHub view raw



アプリケーションの実装の紹介は以上です。

PowerPointやPDFのファイルを対象にテキスト取り出しとチャンク分割を行なってみましたが、設定によっては「ORA-20000: Oracle Textエラー: DRG-50857: oracle error in dbms_vector_chain.utl_to_chunks ORA-06502: PL/SQL: 値または変換エラー: Bulk Bind: Truncated Bindが発生しました」というエラーが発生しました。



これは分割されたチャンクのバイト数が4000を超えるときに発生するように見えます。byに characterを指定した際のmaxの上限は4000、wordsの場合は1000ですが、文字コードがUTF8であるためバイト数としては4000を超えることはあります。

VECTOR_CHUNKSでは、データベースの初期化パラメータのmax_string_sizeがextendedの場合は、チャンクのバイト長が32767に拡張されるとの説明があります。これはVECTOR_CHUNKSの場合に限定されていてDBMS_VECTOR_CHAIN.UTL_TO_CHUNKSには適用されないようです。UTL_TO_CHUNKSの戻り値のタイプであるVECTOR_ARRAY_Tの要素のJSONとして"chunk_data":"VARCHAR2(4000)"と記載されていて、max_string_sizeがextendedでも32767に拡張されないようです。

DBMS_VECTOR_CHAIN.UTL_TO_CHUNKSをVECTOR_CHUNKSに書き直すと、以下のようなコードになります。chunk_specは文字列として指定する必要があります。

```
declare
    l_chunks vector_array_t;
    l_clob   clob;
    l_params json;
    l_chunk  json;
begin
    l_params := json(:P1_PARAMS);
    apex_collection.create_or_truncate_collection('CHUNKS');
    for r in (
        select rownum, t.chunk_offset, t.chunk_length, t.chunk_text
        from vector_chunks(
            :P1_TEXT
            by words
            max 500
            overlap 0
            split by recursively
            language japanese
            normalize all
        ) t
    )
    loop
        apex_collection.add_member(
            p_collection_name => 'CHUNKS'
            ,p_n001 => r.rownum
            ,p_n002 => r.chunk_offset
            ,p_n003 => r.chunk_length
            ,p_clob001 => r.chunk_text
        );
    end loop;
end;
```

sample_vector_chunks.sql hosted with ❤️ by GitHub view raw

VECTOR_CHUNKSの呼び出しではチャンクの長さが4000バイトを超えても「ORA-06502: PL/SQL: 値または変換エラー: Bulk Bind: Truncated Bindが発生しました」のエラーは発生しません。

今回の記事は以上になります。

Oracle APEXのアプリケーション作成の参考になれば幸いです。

完

Yuji N. 時刻: 17:50

共有

自己紹介

Yuji N.

日本オラクル株式会社に勤務していて、Oracle APEXのGroundbreaker Advocateを拝命しました。
こちらの記事につきましては、免責事項の参照をお願いいたします。

詳細プロフィールを表示