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。

2021年4月22日木曜日

Excelファイルのアップロード - その２ - ユーザーの表を使う方法

こちらの記事の続きです。同じページを表APEX_APPLICATION_TEMP_FILESではなく、自分で作成した表を使って実装します。

アップロードしたExcelファイルを保存する表を作成します。クイックSQLの以下のモデルから表FUP_FILESを作成します。

```
# prefix: fup
# semantics: default
files
    content file
```

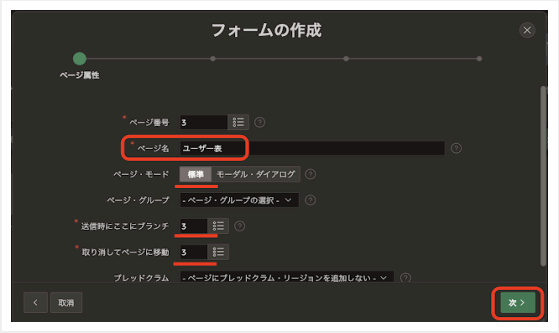
生成されるDDLは以下になります。生成したDDLを実行し、表を作成します。

```
create table fup_files (
    id                          number generated by default on null as identity
                                constraint fup_files_id_pk primary key,
    content                     blob,
    content_filename            varchar2(512),
    content_mimetype            varchar2(512),
    content_charset             varchar2(512),
    content_lastupd             date
)
;
```

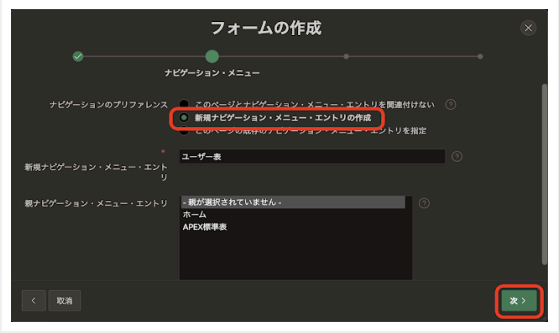
すでに作成済みのアプリケーション、ファイルのアップロードにページを追加します。**アプリケーション・ビルダー**より**ページの作成**を実行し、**ページ作成ウィザード**で**フォーム**をクリックします。

フォームをクリックします。

ページ名を**ユーザー表**とします。**ページ・モード**は**標準**、**送信時にここにブランチ**、**取り消してページに移動**の双方とも、自ページのページ番号(この場合は**3**)を指定します。**次**に進みます。



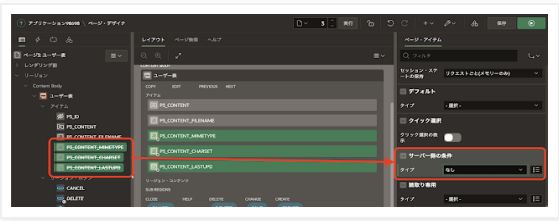
ナビゲーションのプリファレンスとして、**新規ナビゲーション・メニュー・エントリの作成**を選択します。**次**に進みます。



データ・ソースは**ローカル・データベース**とし、**表/ビューの名前**に**FUP_FILES(表)**を選択します。**次**に進みます。

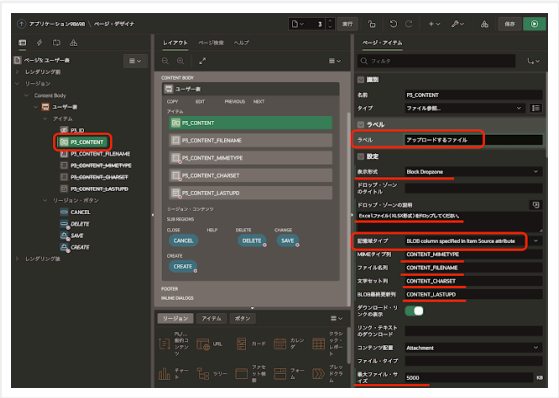
主キー型は**主キー列の選択**とし、**主キー列**として**ID(Number)**を選びます。**作成**をクリックします。

ページが作成されます。最初に**P3_CONTENT_MIMETYPE**、**P3_CONTENT_CHARSET**、**P3_CONTENT_LASTUPD**を表示させないため、これらのページ・アイテムを選択して、**サーバー側の条件**を**なし**に変更します。

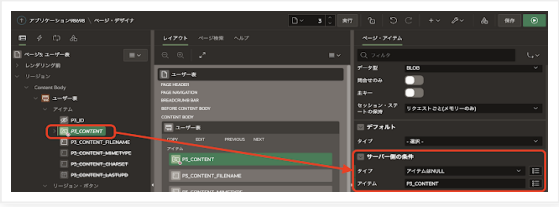


ファイル参照のページ・アイテム**P3_CONTENT**を選択し、前の記事のP2_FILEと同様の動作になる設定を行なっていきます。

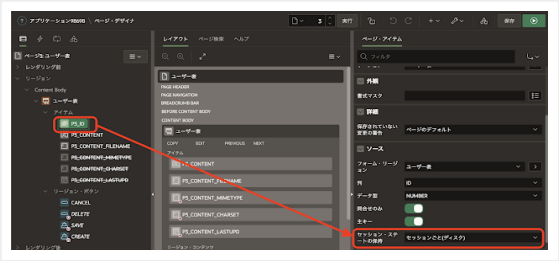
ラベルは**アップロードするファイル**、**設定**の**表示形式**はBlock Dropzone、**ドロップ・ゾーンの説明**は**Excelファイル(XLSX形式)をドロップしてください。**とします。**記憶域タイプ**は**BLOB column specified in Item Source attribute**とします。**MIMEタイプ列**は**CONTENT_MIMETYPE**、**ファイル名列**は**CONTENT_FILENAME**、**文字セット列**は**CONTENT_CHARSET**、**BLOB最終更新列**は**CONTENT_LASTUPD**を指定します。**最大ファイル・サイズ**は**5000KB**とします。



一旦ファイルが選択されたら、その後にページ・アイテムが変更されることを防ぐため、**サーバー側の条件**で**タイプ**に**アイテムがNULL**を選択し、アイテムに**P3_CONTENT**を指定します。この設定によりファイルが未選択のときに限り、ファイル参照が表示されます。

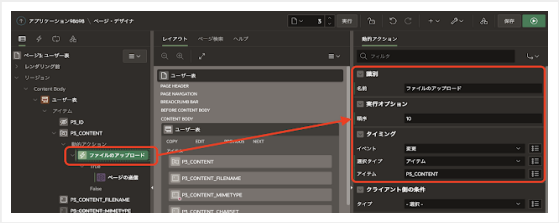


主キーであるページ・アイテム**P3_ID**を、ページ送信後に設定されたIDの値を維持するよう、**セッション・ステートの保持**を**セッションごと(ディスク)**に変更します。

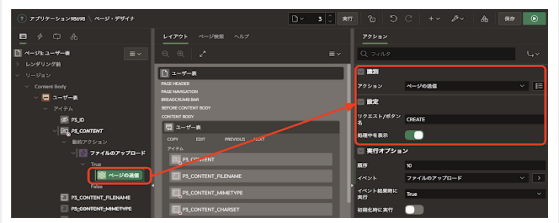


ファイルを選択した時点でアップロードを行うよう、ページ・アイテム**P3_CONTENT**に**動的アクションの作成**を行います。

動的アクションの**名前**を**ファイルのアップロード**とします。**タイミング**はデフォルトで**イベント**が**変更**、**選択タイプ**は**アイテム**、**アイテム**は**P3_CONTENT**となり、P3_CONTENTが変更されるとアクションが実行されます。



Trueアクションを選択し、**アクション**として**ページの送信**を選択します。**設定**の**リクエスト/ボタン名**として**CREATE**を入力します。この指定により、動的アクションから実行される**ページの送信**で、ボタンCREATE(ラベルは作成)を押した時と同じ動作、つまり表へのデータの挿入が行われます。



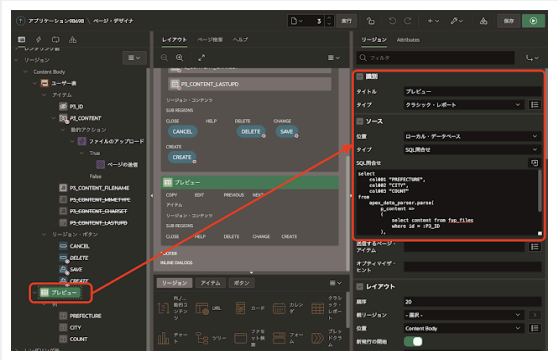
この時点でファイルのアップロードまでは実装できています。

アップロードされた結果をプレビューするために、クラシック・レポートのリージョンを作成します。

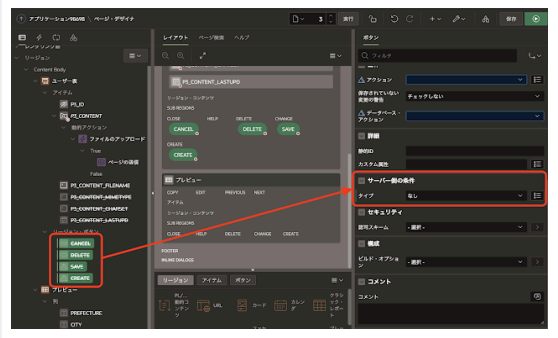
リージョンを作成し、**名前**を**プレビュー**、**タイプ**を**クラシック・レポート**とします。**ソース**の**位置**は**ローカル・データベース**、**タイプ**を**SQL問合せ**とし、**SQL問合せ**に以下のSELECT文を記載します。

```
select
    col001 "PREFECTURE",
    col002 "CITY",
    col003 "COUNT"
from
    apex_data_parser.parse(
```

```
        p_content =>
        (
            select content from fup_files
            where id = :P3_ID
        ),
        p_file_name => :P3_CONTENT_FILENAME,
        p_skip_rows => 1,
        p_file_charset => 'AL32UTF8',
        p_max_rows => 10
    )
```



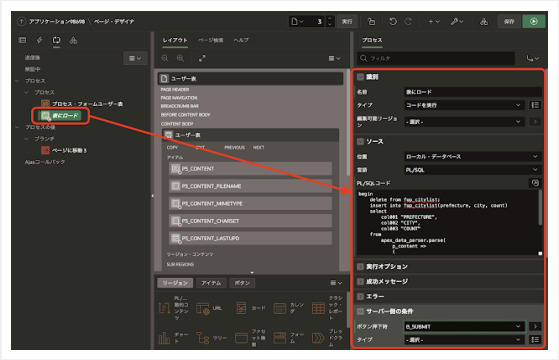
ページ作成ウィザードによって生成されたボタンはすべて使用しないので、ボタン**CANCEL**、**DELETE**、**SAVE**、**CREATE**を選択し、**サーバー側の条件**の**タイプ**を**なし**にします。



表FUD_CITYLISTにデータを投入するプロセスを作成します。データの送信を行うボタンを作成します。**ボタン名**を**B_SUBMIT**、**ラベル**を**送信**とします。

続いて、ボタンB_SUBMITが押された時に実行されるプロセスを作成します。**名前**を**表にロード**とし、**タイプ**は**コードを実行**、**ソースの位置**は**ローカル・データベース**、**言語**は**PL/SQL**を選択し、**PL/SQLコード**として以下を記述します(プレビューではないのでp_max_rowsの指定を除いています)。**サーバー側の条件**として、**ボタン押下時**に**B_SUBMIT**を指定することにより、**送信**ボタンを押した時のみ実行されるようにします。

```
begin
    delete from fup_citylist;
    insert into fup_citylist(prefecture, city, count)
    select
        col001 "PREFECTURE",
        col002 "CITY",
        col003 "COUNT"
    from
        apex_data_parser.parse(
            p_content =>
            (
                select content from fup_files
                where id = :P3_ID
            ),
            p_file_name => :P3_CONTENT_FILENAME,
            p_skip_rows => 1,
            p_file_charset => 'AL32UTF8'
        );
end;
```



上記のコードは、送信をクリックした際に、すでに保存されているデータをすべて削除し、新たにアップロードしたデータで入れ替えています。追加やマージといった動作にしたい場合は、PL/SQLのコードを変更することになります。

以上で表へのデータのロードも実行されるようになりました。動作確認をもう少し簡単にするため、ページを初期化するボタンと表FUD_CITYLISTの内容を表示するレポートを追加します。この作業は前の記事とまったく同じです。

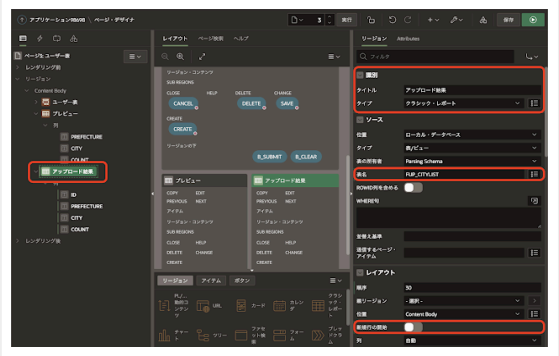
ボタンの作成を行い、**ボタン名**をB_CLEAR、**ラベル**を**クリア**とします。

ボタンB_CLEARを押した時に実行されるプロセスを作成します。作成したプロセスの**名前**を**ページの初期化**とし、**タイプ**に**セッション・ステートのクリア**を選択します。**サーバー側の条件**として、

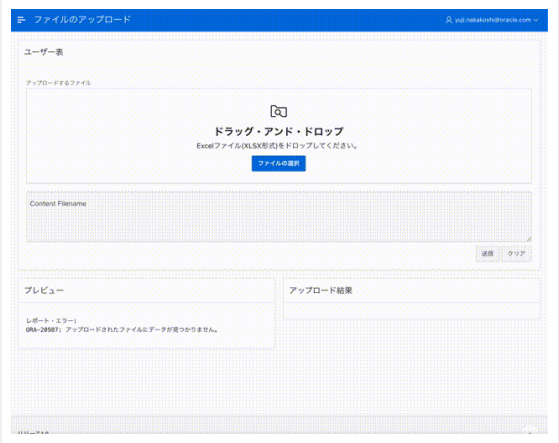
ボタン押下時に**B_CLEAR**を選択します。これでクリアのボタンを押した時に、ページが初期化されます。

アップロード結果を表示するクラシック・レポートを作成します。新規に**リージョンの作成**を行います。

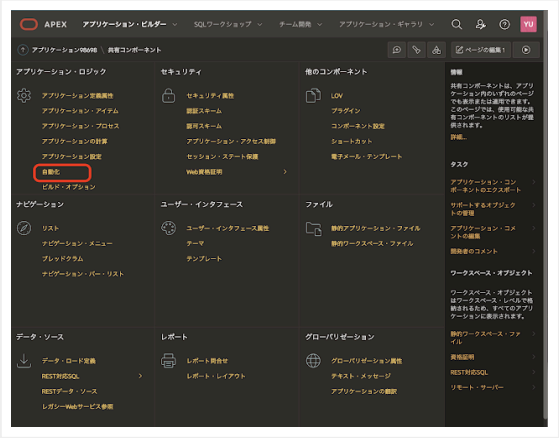
名前を**アップロード結果**とし、**タイプ**は**クラシック・レポート**を選択します。**ソース**の**表名**に**FUP_CITYLIST**を選択し、プレビューと横並びになるよう、**レイアウト**の**新規行の開始**を**OFF**とします。



以上で完成です。ページを実行し、実装を確認します。以下の動作になります。



ユーザーが作成した表にアップロードしたファイルを保存しているので、不要になったデータを削除する処理を実装する必要があります。**共有コンポーネント**の**アプリケーション・ロジック**に含まれる**自動化**を、そのような用途に使うことができます。



以上でデータ・ロードに関する説明は終了です。

Oracle APEX 21.1では、これらの作業のほとんどがページ作成ウィザードによって行われる予定です。ですので、いちからページを作成することはありませんが、自動的に生成されたコンポーネントをカスタマイズする際には、役に立つ知識かと思います。

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/excelfileupload.sql
サンプルのデータはこちらです。
https://github.com/ujnak/apexapps/blob/master/exports/citylist.xlsx

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: 13:40

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