

# 日々是Oracle APEX

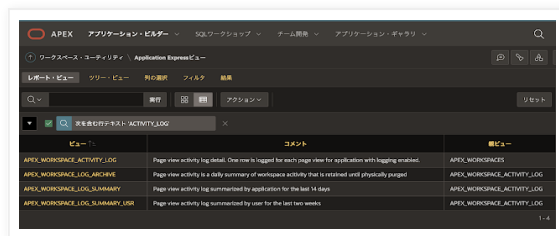
Oracle APEXを使った作業をしていて、気の付いたところを忘れないようにメモをとります。

2021年2月17日 水曜日

## 自律トランザクションを使ってデータベース操作のログを取得する

Oracle APEXのアプリケーションで、操作のログはどのように残すのですか？と質問を受けました。

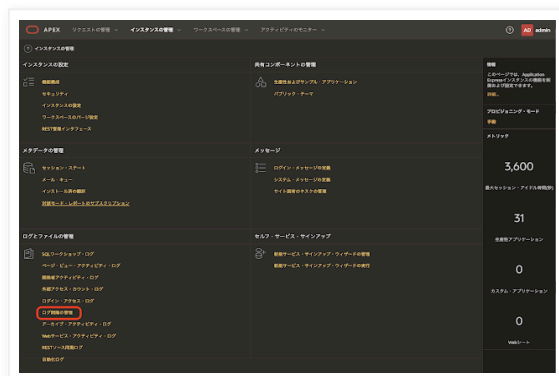
アプリケーションに依存したもので無ければ、Oracle APEXのアプリケーションへのアクセスは、**ワークスペース・ユーティリティのApplication Expressビュー**として提供されている **APEX\_WORKSPACE\_ACTIVITY\_LOG**から参照できます。



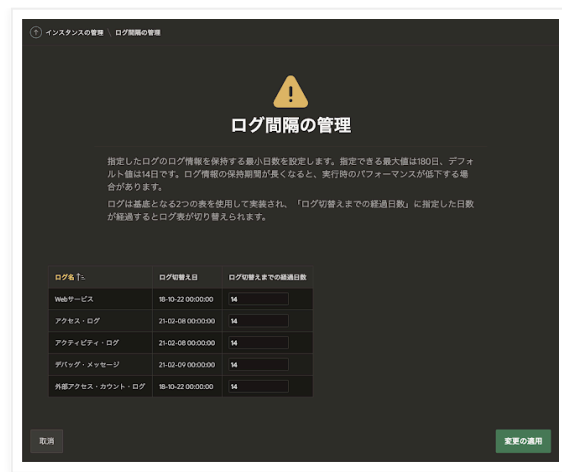
ログの保持期間はデフォルトで28日(ログを保存する2つの表が14日毎に切り替わるため)で、パフォーマンスに影響が出るため延長は推奨されていません。とはいえ、変更方法ですが、オンプレのインストールの場合はOracle APEXの**管理アプリケーション**にサインインし、**インスタンスの管理**を開きます。



ログとファイルの管理よりログ間隔の管理を開きます。

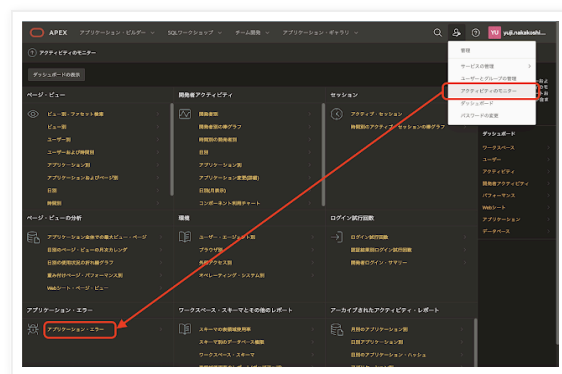


ログ情報を保持する最小日数の指定ができます。



対応する画面がない環境では、APIの呼び出しによって変更が可能です。APEX\_INSTANCE\_ADMINパッケージに含まれるSET\_LOG\_SWITCH\_INTERVALプロシージャを使用します。

アプリケーションで発生したエラーも標準でログに記録されていて、アクティビティのモニターからアプリケーション・エラーを開くことで参照できます。



いつ、誰が、どのページで、どのコンポーネントで、どのようなエラーが発生したのか、などを確認することができます。



標準のログについては以上です。

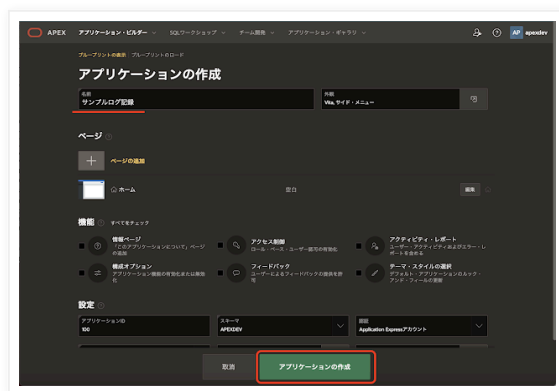
本記事の本題である、カスタムのログ取得について紹介します。

確認のためにクイックSQLの以下のモデルにて、表TTX\_ACTIVITY\_LOGとTTX\_OPERATIONSを作成します。

```
# prefix: ttx
# semantics: default
activity_log
  operation vc400
  operation_date date /default sysdate

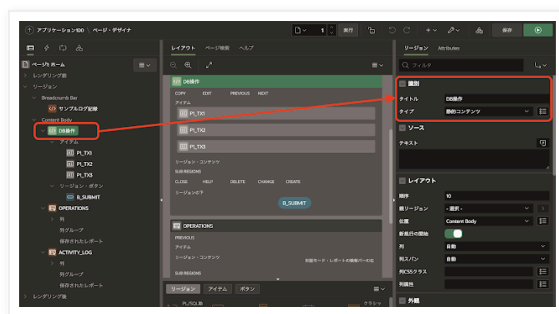
operations
  name vc20
  value vc20
```

アプリケーション・ビルダーより、空のアプリケーションを作成します。名前はサンプルログ記録としました。

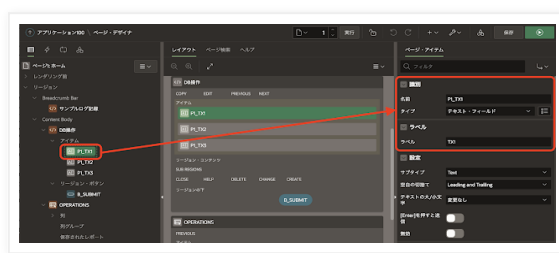


アプリケーションが作成できたら、ホーム・ページをページ・デザイナーで開き、3つのリージョンをContent Bodyに追加します。

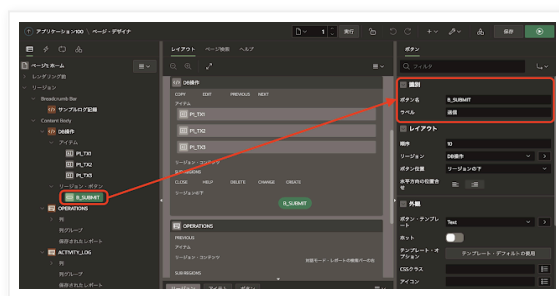
最初にタイトルをDB操作、タイプを静的コンテンツとしてリージョンを作成します。



リージョン内に3つのページ・アイテムを作成します。名前をそれぞれP1\_TX1、P1\_TX2、P1\_TX3とし、ラベルもそれぞれTX1、TX2、TX3とします。P1\_TX1での設定は以下になります。

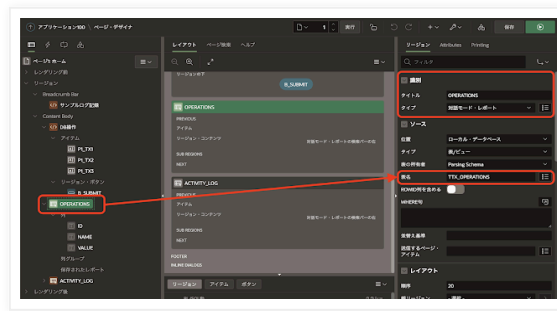


ページ・アイテムを送信するためのボタンを作成します。ボタン名をB\_SUBMITとし、ラベルを送信にします。

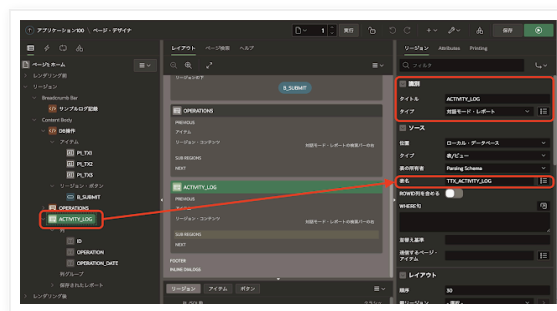


静的コンテンツのリージョンについては以上で作成完了です。

続いて対話モード・レポートのリージョンを作成します。**タイトル**として**OPERATIONS**を設定し、**タイプ**は**対話モード・レポート**、**表名**として**TTX\_OPERATIONS**を選択します。



同様に**タイトル**が**ACTIVITY\_LOG**、**タイプ**が**対話モード・レポート**、**表名**が**TTX\_ACTIVITY\_LOG**のリージョンを作成します。

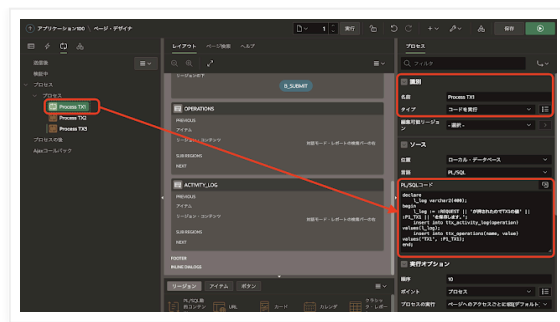


画面の開発は以上で完了です。ページを実行すると以下の画面が表示されます。



ページ・デザイナーに戻り、プロセスを3つ追加します。以下のコードを実行します。**名前**は**Process TX1**とし、**タイプ**は**コードを実行**を選びます。TX2、TX3については、それぞれTX1の部分をTX2、TX3に変更します。

```
declare
    l_log varchar2(400);
begin
    l_log := :REQUEST || 'が押されたのでTX1の値' || :P1_TX1 || 'を保存します。';
    insert into ttx_activity_log(operation) values(l_log);
    insert into ttx_operations(name, value) values('TX1', :P1_TX1);
end;
```



ページ・アイテムP1\_TX1、P1\_TX2、P1\_TX3の値を受け取って、表TTX\_OPERATIONSにデータを挿入し、同時に操作を表TTX\_ACTIVITY\_LOGに書き込むようになります。

この状態で実行します。TX1、TX2、TX3に値を指定し、送信ボタンをクリックします。

以下のようにレポートが表示されます。データが保存され、操作も記録されています。きちんと動いているように見えます。

ID	Name	Value
1	TX1	first
2	TX2	second
3	TX3	third

ID	Operation	Operation Date
1	表TTX_OPERATIONSにデータを挿入しました。	2023/02/07
2	表TTX_ACTIVITY_LOGにデータを挿入しました。	2023/02/07
3	表TTX_ACTIVITY_LOGにデータを挿入しました。	2023/02/07

今度はTX3に20バイトを超えるデータを指定します。想定どおりエラーORA-12899: 列"APEXDEV"."TTX\_OPERATIONS"."VALUE"の値が大きすぎます(実際: 180、最大: 20)が発生し、データベースへの書き込みは行われません。

表の内容をみるとページ・アイテムP1\_TX1、P1\_TX2、P1\_TX3の値が保存されていないのは想定どおりなのですが、操作も記録されていません。操作の記録はエラーの発生によらず、記録されている必要があります。

ID	Name	Value
1	TX1	first
2	TX2	second
3	TX3	third

ID	Operation	Operation Date
1	表TTX_OPERATIONSにデータを挿入しました。	2023/02/07
2	表TTX_ACTIVITY_LOGにデータを挿入しました。	2023/02/07
3	表TTX_ACTIVITY_LOGにデータを挿入しました。	2023/02/07

そのため、ログを保存する操作は自律トランザクションとして、本体とは異なるトランザクションを開始する必要があります。

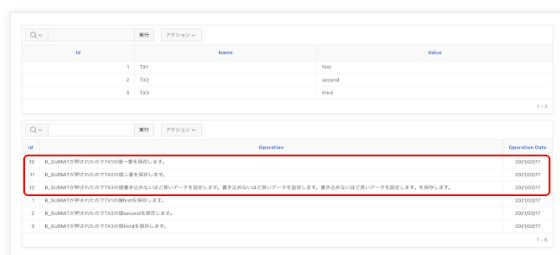
次のプロシージャを作成します。**pragma autonomous\_transaction**句を含めることで、独立したトランザクションが開始されます。プロシージャの終了前に**commit**することで、このプロシージャを呼び出したトランザクションが**rollback**しても、この書き込みは永続化されます。

```
create or replace procedure ttx_log_operation
(
    p_operation in varchar2
)
as
pragma autonomous_transaction;
begin
    insert into ttx_activity_log(operation) values(p_operation);
    commit;
end ttx_log_operation;
```

プロセスに記述したコードを以下に変更します。表TTX\_ACTIVITY\_LOGへのINSERT文をプロシージャTTX\_LOG\_OPERATIONの呼び出しに変更します。TX2、TX3についても同様の変更を行います。

```
declare
    l_log varchar2(400);
begin
    l_log := :REQUEST || 'が押されたのでTX1の値' || :P1_TX1 || 'を保存します。';
    ttx_log_operation(l_log);
    insert into ttx_operations(name, value) values('TX1', :P1_TX1);
end;
```

先ほど失敗したエラーになった操作を再実行すると、レポートの表示は以下に変わります。



id	tx1	tx2	tx3
1	TX1		
2	TX2		
3	TX3		

id	name	value	operation
1	TX1		ttx_log_operation('が押されたのでTX1の値'    :P1_TX1    'を保存します。');
2	TX2		ttx_log_operation('が押されたのでTX2の値'    :P1_TX2    'を保存します。');
3	TX3		ttx_log_operation('が押されたのでTX3の値'    :P1_TX3    'を保存します。');

データが保存されていないのは同様ですが、Operationには、プロセス自体が呼び出されていることが分かる操作ログを確認できます。

以上で、自律トランザクションを使ったデータベースへのログ書き込み方法の紹介は終了です。

確認に使用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/samplelogging.sql>

Oracle APEXのアプリケーション開発の一助になれば幸いです。

完

[ウェブ バージョンを表示](#)

#### 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](#)

Powered by Blogger.

---