

White Paper: Migrating Call Notes from AR to Advanced Collections using JTF_NOTES_PUB API

Author: Anil Patil
Created: May 2, 2007

Product: Advanced Collections
Version: 11.5.10 RUP3.1

Objective

The Objective of this document is to explain how we migrated Call Notes from AR to Advanced Collections

Business Requirement

Our client is implementing Advanced Collections. The Client is currently using AR Collections Workbench to manage its Collections Activities. As such the Call notes are currently maintained in AR in the Customer Calls form as a Response Note or a Call Topic Note . These notes are either at the account level or a transaction level. When the Collectors switch to the Collections form in Advanced Collections, the AR Call notes cannot be viewed in the Notes Tab. Hence there is a need to migrate the AR Call notes from AR to Advanced Collections

Benefits

The AR Call Notes can be viewed from the Collections form without the need to navigate to the AR Customer Calls Form. With this call notes migration, the collectors do not lose the Call Notes history information that they previously maintained using the AR Collections workbench.

Solution

The Notes entered in Advanced Collections are stored using the Oracle Common Application Notes Component. Hence to migrate the AR Call Notes to Advanced Collections, we used the `jtf_notes_pub.create_note` API. We created a custom concurrent program with a 'From Date' and a 'To Date' parameter. This program selects all the AR Call Notes data within the date parameter and passes it as a parameter to the API. The API then creates a jtf note based on the parameters passed. The jtf note could be at the Account level or the Invoice level depending on how the note was created in AR.

Business Rules Implemented

If a particular Call Note is migrated, it should not be migrated again even if the user runs the concurrent program with the same or overlapping date parameters

High-Level Algorithm for the Concurrent Program

1. Select the source data that needs to be migrated
2. Check if the Source data has already been migrated.

3. If 'YES', return to End . Else go to '4'
4. Provide the source data as input parameters to the JTF_NOTES_PUB API and execute it.
5. Capture the return status flag from the API
6. Provide the output of the Notes migrated (Success / failure) in the Output file.

Queries to select the source data

1. Query to select the response note from customer calls. These notes are at the account level and hence have no reference to the transaction

```

SELECT acc_customer_id "CUST_ACCOUNT_ID"
      , ano_text note_text
      , rcu_customer_number account_number
      , 'RESPONSE' note_type
      , last_updated_by
      , last_update_date
      , last_update_login
      , created_by
      , creation_date
FROM ar_customer_calls_v
WHERE trunc(creation_date) >= trunc(&p_date_from)
AND trunc(creation_date) <= trunc(&p_date_to) ;

```

2. Query to select the customer call topic notes. If the customer_trx_id is NULL, it means that the note is at an account level. Else the note is at a transaction level.

```

SELECT cct_customer_trx_id customer_trx_id
      , cct_customer_id cust_account_id
      , ano_text note_text
      , 'CALL_NOTE' note_type
      , last_updated_by
      , last_update_date
      , last_update_login
      , created_by
      , creation_date
FROM ar_customer_call_topics_v
WHERE trunc(creation_date) >= trunc(&p_date_from)
AND trunc(creation_date) <= trunc(&p_date_to) ;

```

Check if Source data is already migrated

1. Query for checking if the account level note is already migrated

```

SELECT DISTINCT COUNT (*)
      INTO x_count
FROM jtf_notes_b jtfn
      , jtf_notes_tl jtfn_tl
WHERE
      jtfn.source_object_code = 'IEX_ACCOUNT'
AND jtfn.source_object_id = note_rec.cust_acct_id

```

```

AND jtfn_tl.jtf_note_id      = jtfn.jtf_note_id
AND jtfn_tl.source_lang     = 'US'
AND UPPER(jtfn_tl.notes)    = UPPER(note_rec.note_text)
AND jtfn.creation_date      = note_rec.creation_date
AND jtfn.created_by         = note_rec.created_by
AND jtfn.last_update_date   = note_rec.last_update_date
AND jtfn.last_updated_by    = note_rec.last_updated_by;

```

In the above query, note_rec is a record_type consisting the source data. If the x_count variable is not zero, it means that the note is already migrated.

2. Query for checking if the invoice level note is already migrated

```

SELECT DISTINCT COUNT (*)
INTO x_count
FROM jtf_notes_b jtfn
     , jtf_notes_tl jtfn_tl
WHERE
    jtfn.source_object_code = 'IEX_INVOICES'
  AND jtfn.source_object_id = note_rec.payment_schedule_id
  AND jtfn_tl.jtf_note_id   = jtfn.jtf_note_id
  AND jtfn_tl.source_lang   = 'US'
  AND UPPER(jtfn_tl.notes)  = UPPER(note_rec.note_text)
  AND jtfn.creation_date    = note_rec.creation_date
  AND jtfn.created_by       = note_rec.created_by
  AND jtfn.last_update_date = note_rec.last_update_date
  AND jtfn.last_updated_by  = note_rec.last_updated_by;

```

In the above query, note_rec is a record_type consisting the source data. If the x_count variable is not zero, it means that the note is already migrated.

JTF_NOTES_PUB API

1) Account Level Notes – The Account level notes are migrated using the following code

```

Declare
l_notes_detail          CLOB;
l_note_type             VARCHAR(30) := 'IEX_HIST';
l_note_status          VARCHAR2 (1) := 'I';
l_return_status        VARCHAR2 (1);
l_msg_count            NUMBER;
l_msg_data             VARCHAR2 (2000);
l_note_id              NUMBER ;
l_msg_index_out        NUMBER;
BEGIN
jtf_notes_pub.create_note
  (p_api_version        => 1.0,
  p_init_msg_list      => 'T',
  p_commit             => 'T',
  p_jtf_note_id        => NULL,
  p_validation_level   => 100,
  p_source_object_id   => note_rec.cust_account_id,

```

```

p_source_object_code      => 'IEX_ACCOUNT',
p_notes                   => note_rec.note_text ,
p_notes_detail            => l_notes_detail,
p_entered_by              => note_rec.created_by,
p_entered_date            => note_rec.creation_date,
p_last_update_date        => note_rec.last_update_date,
p_last_updated_by         => note_rec.last_updated_by,
p_creation_date           => note_rec.creation_date,
p_created_by              => note_rec.created_by,
p_last_update_login       => fnd_global.login_id,
p_note_type               => l_note_type,
p_note_status             => l_note_status,
x_jtf_note_id             => l_note_id,
x_return_status           => l_return_status,
x_msg_count               => l_msg_count,
x_msg_data                => l_msg_data
);

IF (l_return_status <> 'S')
THEN
  fnd_file.put_line(fnd_file.LOG,'l_return_status <> S ');
  IF (fnd_msg_pub.count_msg > 0)
  THEN
    FOR i IN 1 .. fnd_msg_pub.count_msg
    LOOP
      fnd_msg_pub.get
        (p_msg_index      => i,
         p_encoded         => 'F',
         p_data            => l_msg_data,
         p_msg_index_out   => l_msg_index_out
        );
      DBMS_OUTPUT.put_line ('Error:' || l_msg_data);
      fnd_file.put_line(fnd_file.LOG,'ERROR :' ||
        l_msg_data);
    END LOOP;
  END IF;
END IF;
COMMIT;
END ;

```

In the above query, note_rec is a record_type consisting the source date. The value of l_note_type should be a valid lookup_code for lookup_type JTF_NOTE_TYPE

```

SELECT LOOKUP_CODE,MEANING,DESCRIPTION,
TAG,START_DATE_ACTIVE,END_DATE_ACTIVE,ENABLED_FLAG,
LOOKUP_TYPE
FROM FND_LOOKUP_VALUES_VL WHERE (nvl('', territory_code) =
territory_code or territory_code is null)
AND lookup_type = 'JTF_NOTE_TYPE'
order by LOOKUP_CODE ;

```

2) Invoice Level Notes – For migrating the Invoice level notes, the code remains the same except the value passed to the parameters p_source_object_id and p_source_object_code . The values that needs to be passed to these parameters are

```
p_source_object_id      => note_rec.payment_schedule_id ,  
p_source_object_code    => 'IEX_INVOICES'
```

Summary

This document details one approach for migrating Call Notes from AR to Advanced Collections. These notes can then be viewed in the Notes Tab of the Collections Form.

References

Oracle Common Applications Components – API Reference Guide

Oracle Advanced Collections Implementation Guide

Oracle Advanced Collections User Guide