

# EIM ETL Validation Script Guidelines

*Bureau of Information Systems*

<Project Name>

## TABLE OF CONTENTS

Revision History	3
1. Introduction	4
2. Validation Script Guidelines	5
2.1 Post ETL Validation Scripts/Checks (BAL 800)	5
2.2 Move from TFP to Production Moves Scripts/Checks (BAL_600)	6

## REVISION HISTORY

Date	Version	Description	Author
<dd/mm/yy>	<x.x>	<details>	<name>
2/23/2016	1.1	Reviewed for content; changed DPW to DHS.	Don Pidich - EKMS

## 1. INTRODUCTION

*The purpose of this Extract Transform and Load (ETL) Validation Script Guidelines Document is to detail the information needed to create post ETL load data integrity/data quality checks. This may also include checks to verify the complete move of data from a TFP (Test for Production) to Production environments.*

## 2. VALIDATION SCRIPT GUIDELINES

*Validation scripts should be utilized to verify that there were no problems with the ETL process. Checks can be used to verify such things as: min and max range of record counts, dates are in a correct range of values, identify values not found in supporting code/decode dimension tables, identify possible duplicate records, identify incorrect data (i.e. pregnant males) along with many others.*

### 2.1 Post ETL Validation Scripts/Checks (BAL 800)

The template 'EIM ETL Validation Script\_Guidelines.xlsx' will be need to be completed with the below information for each new or modified check that is being requested.

1. **Check Name** (maximum of 50 characters) should specify a name for the check which should clearly state what the check is trying to accomplish
2. **Table** should specify the fully qualified table name to include schema and the database instance
3. **Error Message** should specify what it means when a validation check fails, what the steps to fix it should be and contact information
4. **Benchmark Minimum** should specify the lower value that the validation check will be compared against.
5. **Benchmark Maximum** should specify the upper value that the validation check will be compared against.
6. **Benchmark Type** specifies what type of data the check will return. The two supported types are listed below:
  - a. **Numeric** this will return a numeric value. This is the most common and preferred type of check. Benchmark Min and Max values need to be numeric.
  - b. **String** this will return a string value. This is normally associated with an SQL DECODE function. Benchmark Min and Max values need to be strings.
7. **Free Form** the actual SQL statement used to run when the check is implemented. The statement needs to return only a single row. See the example below:

```
select count(*) from (select t.cde_drg from edw.t_drg_dim t where IND_EFFV_CURR = 'Y' group by t.cde_drg having count(*) > 1)
```
8. **Group** specify what group or program the check belongs. If it belongs to an existing group already please specify it.
9. **Date Start** date when the check should go live
10. **Date End** fill in if the check has a defined period it is expected to run

11. **Status** indicate if this is a new check, update to an existing check or one that is to be retired
12. **Initiation** specify how the check will be called such as via a workflow command or manually along with when it will be called.

## 2.2 Move from TFP to Production Moves Scripts/Checks (BAL\_600)

These checks can be used to verify moves from the test for production (TFP) to production environments. The checks will need to be designed keeping in mind if TFP is either the full universe of data or only a subset.

1. **Table** should specify the fully qualified table name to include schema and the database instance
2. **SQL COUNT COMPARE TO PROD** SQL statement which should be derived to ensure that all the records found in TFP were successfully moved to production. See below example:

```
select count(*) from T_CLAIM_CONDTN_HELPER prd,  
T_CLAIM_CONDTN_HELPER@edw_tfp tfp where tfp.idn_system_claim =  
prd.idn_system_claim and prd.cde_condtn = tfp.cde_condtn and prd.cde_seq_condtn =  
tfp.cde_seq_condtn
```

3. **SQL COUNT OF TFP** SQL statement which will identify the record count in TFP for comparison. See below example: `select count(*) from T_CLAIM_CONDTN_HELPER`
4. **Group** specify what group or program the check belongs. If it belongs to an existing group already please specify it.
5. **Date Start** date when the check should go live
6. **Date End** fill in if the check has a defined period it is expected to run
7. **Status** indicate if this is a new check, update to an existing check or one that is to be retired
8. **Scheduled Move** specify when the table is scheduled to be moved. For example the table is to be moved weekly every Monday night at 8 PM.
9. **On Error Message** specify contact information to be used if a problem is identified