**ILEBCC DB PATCHING INSTRUCTIONS**

Table of Contents

[1.0 PRE-PATCH Validation Steps 2](#_Toc68705501)

[1.1 Access Requirements 2](#_Toc68705502)

[1. 2 Check for any running jobs 2](#_Toc68705503)

[1.3 Turn-off all CRONS 2](#_Toc68705504)

[2.0 POST-PATCH validation Steps 3](#_Toc68705505)

[2.1 Turn-On all CRONS 3](#_Toc68705506)

[2.2 Check all scheduled load\_production\_planning jobs are running 3](#_Toc68705507)

[2.3 Validate ETL Job Management Tables 3](#_Toc68705508)

[2.3.1 Check if ETL job management scripts are getting executed 3](#_Toc68705509)

[2.3.2 Check for stuck jobs in ETL Job Management tables 4](#_Toc68705510)

[2.3.3 Reset stuck jobs in ETL Job Management tables 4](#_Toc68705511)

[2.3.4 Verify reset jobs are running fine 4](#_Toc68705512)

[2.4 Validate scheduled ***Contact Center*** job run 4](#_Toc68705513)

[2.4.1 Check if scheduled ***Contact Center*** job ran successfully 4](#_Toc68705514)

[2.4.2 Schedule an adhoc ***Contact Center*** job 5](#_Toc68705515)

[2.4.3 Check if adhoc ***Contact Center*** job is unsuccessful 5](#_Toc68705516)

[2.4.4 Add an entry in scheduled job table to avoid duplicate run 6](#_Toc68705517)

# 1.0 PRE-PATCH Validation Steps

## 1.1 Access Requirements

* None as we assume DBA is running validation and has all requisite table accesses.
* ILEBCC Production database url is uvacpmmetl03md.maxcorp.maximus; port 1531

## 1. 2 Check for any running jobs

Check if all jobs have completed processing – run the following query to check.

**SELECT** *\**

**FROM** maxdat\_cc.cc\_a\_scheduled\_job

**WHERE is\_running** = 1;

**SELECT \***

**FROM maxdat\_cc.ETL\_JOB\_STATUS**

**WHERE JOB\_RUNNABLE = 'N';**

If the query returns no rows, then no jobs are running and we can proceed.

## 1.3 Turn-off all CRONS

Work with the app admin to make sure all crons are turned off. The app admins are provided a pre-patching document with detailed steps on what to check/verify before turning the crons off.

# 2.0 POST-PATCH validation Steps

## 2.1 Turn-On all CRONS

Work with app admin to ensure crons are turned back on after patching is completed.

## 2.2 Check all scheduled load\_production\_planning jobs are running

After confirmation of crons being turned back on - Verify if the scheduled load\_production\_planning jobs (that run at the start of the hour) are **running** and **completing** successfully as expected.

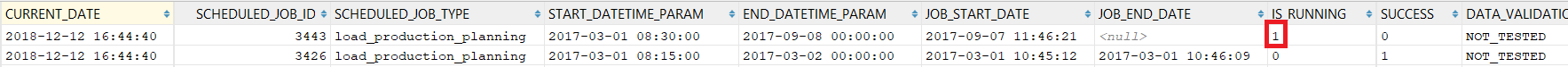
**SELECT** *CURRENT\_DATE*, **a**.*\**

**FROM** MAXDAT\_CC.CC\_A\_SCHEDULED\_JOB **a**

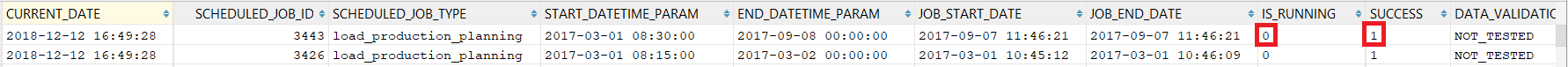
**WHERE SCHEDULED\_JOB\_TYPE** = **'load\_production\_planning'**

**ORDER BY SCHEDULED\_JOB\_ID DESC**;

IS\_RUNNING should == 1 when the job is **running**.



After the job successfully completes, IS\_RUNNING == 0 and SUCCESS == 1



If load\_production\_planning entries aren’t being created every 60 minutes, then check section 2.3.

If the load\_production\_planning fails with entries of IS\_SUCCESS = 0 and IS\_RUNNING = 0 (jobs have completed but not successfully) – this needs to be looked into for any database errors.

## 2.3 Validate ETL Job Management Tables

### 2.3.1 Check if ETL job management scripts are getting executed

Check in the ETL job management run table if any jobs started running after CRONs are enabled, by executing the below query.

If not, reach out to App Admins to check if the script for ETL job management jobs (etl\_job\_control.bash) is enabled in the CRON. Once App admins confirms the jobs are running, re-execute this step 2.3.1. If the jobs are still not running after 30 minutes, check 2.3.2.

**SELECT \***

**FROM MAXDAT\_CC.ETL\_JOB\_RUN**

**WHERE RUN\_START\_DT >= '2020-11-02 14:00:00';**

**-- The above date/time should be corresponding to when CRONs were enabled after the patch**

### 2.3.2 Check for stuck jobs in ETL Job Management tables

If the ETL job management script is enabled in the CRON, but the jobs are still not running, check if any jobs in ETL Job management tables are stuck by running the below query. This query will return the JOB\_ID for any jobs stuck on that particular day. If this query returned no rows and jobs are still not running, reach out to a developer. If there are any stuck jobs, go to step 2.3.3.

**SELECT JOB\_ID**

**FROM MAXDAT\_CC.ETL\_JOB\_LOG**

**WHERE LOG\_DESC = 'JOB STUCK - RESET MAY BE REQUIRED'**

**AND LOG\_UPD\_DT >= '2020-11-02 14:00:00';**

**-- The above date/time should be corresponding to when CRONs were enabled after the patch**

### 2.3.3 Reset stuck jobs in ETL Job Management tables

Reset each stuck job by executing the RESET\_ETL\_JOB procedure in MAXDAT\_CC schema for each JOB\_ID returned in the previous step. Replace “?” below with each JOB\_ID returned in the previous step.

**BEGIN**

**ETL\_JOB.RESET\_ETL\_JOB (?);**

**END;**

### 2.3.4 Verify reset jobs are running fine

ETL Job management shell script is usually scheduled to run every 5 or 10 minutes. So, wait for the next execution of the ETL job management scripts and check if the stuck jobs are started running. If not, reach out to the developer to have a look at the issue.

## 2.4 Validate scheduled ***Contact Center*** job run

### 2.4.1 Check if scheduled ***Contact Center*** job ran successfully

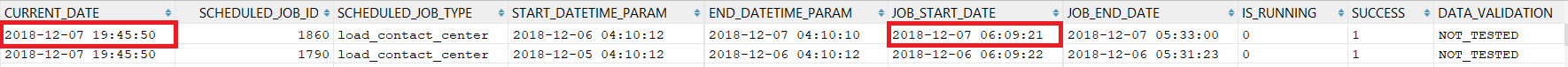
Verify if the scheduled contact center job (that runs once daily) is **ran that day** and **completed** successfully as expected – Please check if the date stamp matches to current date.

**SELECT** *CURRENT\_DATE*, **a**.*\**

**FROM** MAXDAT\_CC.CC\_A\_SCHEDULED\_JOB **a**

**WHERE SCHEDULED\_JOB\_TYPE** = **'load\_contact\_center'**

**ORDER BY SCHEDULED\_JOB\_ID DESC**;



### 2.4.2 Schedule an adhoc ***Contact Center*** job

If the job was not run that day, then insert the job entry into adhoc table for adhoc run using **yesterday’s** date as **START\_DATETIME\_PARAM** and **today’s** date as **END\_DATETIME\_PARAM**.

For example, if today is **12/07/2018** and load\_contact\_center missed its run today, then the insert query will be

**INSERT INTO** MAXDAT\_CC.CC\_A\_ADHOC\_JOB (

**adhoc\_job\_type**,

**start\_datetime\_param**,

**end\_datetime\_param**,

**is\_pending**,

**ACD\_SOURCE**,

**WFM\_SOURCE**)  
**VALUES** (

**'load\_contact\_center'**,

**'2018-12-06 00:00:00'**,

**'2018-12-07 00:00:00'**,

1,

**'AVAYA'**,

**'NA'**);

**COMMIT**;

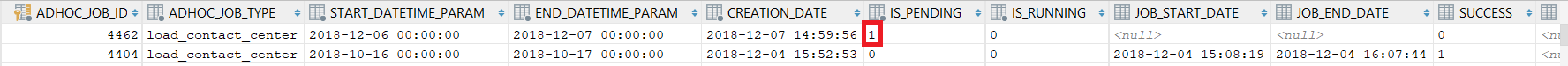
Check for the adhoc job to run (the job kicks off at the 15th minute of the hour):

**SELECT** *\**

**FROM** MAXDAT\_CC. CC\_A\_ADHOC\_JOB

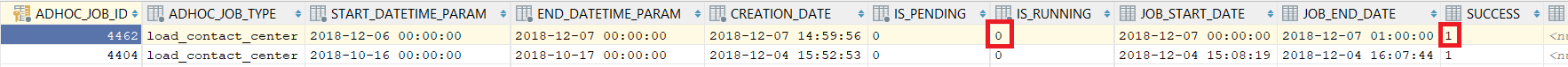
**WHERE ADHOC\_JOB\_TYPE** = **'load\_contact\_center'**

**ORDER BY ADHOC\_JOB\_ID DESC**;



Once the job kicks off, IS\_PENDING == 0 and IS\_RUNNING == 1

When the job finishes successfully the entry would be IS\_RUNNING == 0 and SUCCESS == 1



### 2.4.3 Check if adhoc ***Contact Center*** job is unsuccessful

If the job is unsuccessful where IS\_RUNNING == 0 and SUCCESS == 0, please reach out to app admin to investigate logs. App admin has instructions on how to check adhoc logs for errors. If required to resolve the error, please reach out to available MAXDAT developer.

### 2.4.4 Add an entry in scheduled job table to avoid duplicate run

Add a new entry in scheduled table for the date we just ran, so the job doesn’t run again next day for the same day. If today is **12/07/2018**, the **START\_DATETIME\_PARAM** will be of yesterday’s date and **END\_DATETIME\_PARAM** will be of today’s date and **JOB\_START\_DATE** and **JOB\_END\_DATE** will come from the previous query’s **JOB\_START\_DATE** and **JOB\_END\_DATE**.

**INSERT INTO** MAXDAT\_CC.CC\_A\_SCHEDULED\_JOB (

**scheduled\_job\_type**,

**start\_datetime\_param**,

**end\_datetime\_param**,

**job\_start\_date**,

**job\_end\_date**,

**success**)  
**VALUES** (

**'load\_contact\_center'**,

**'2018-12-06 00:00:00'**,

**'2018-12-07 00:00:00'**,

*to\_date*(**'2018-12-07 00:00:00'**, **'YYYY-MM-DD HH24:MI:SS'**),   
 *to\_date*(**'2018-12-07 01:00:00'**, **'YYYY-MM-DD HH24:MI:SS'**),

1);

**COMMIT**;