Voyager to Alma Migration Query: Course Reserves Cleanup

Created By: Laura Guy (contact)
Created on: 9/18/2020

When migrating from Voyager to Alma it may be desirable to review and cleanup Course Reserves data. The following queries may assist you with that.

These queries (a variety of queries are presented here) can be run using Voyager Prepackaged Access Reports. They each list various aspects of Voyager Course Reserves information.

Note that Course Reserves data are often messy and complicated. You should review the results of these queries carefully as there might be inaccuracies due to data vagaries.

Be aware that some of the queries presented below are pass through queries, and are clearly identified as such.

Active Reserve Lists with Course Name and Titles (Only Lists linked to a Course are included; all titles whether on or off reserve are included)

```
SELECT RESERVE_LIST.LIST_TITLE, utf8to16([bib_text].[TITLE]) AS Title,
COURSE.COURSE_NAME
FROM ((((RESERVE_LIST
INNER JOIN RESERVE_LIST_ITEMS ON RESERVE_LIST.RESERVE_LIST_ID =
RESERVE_LIST_ITEMS.RESERVE_LIST_ID)
INNER JOIN ITEM ON RESERVE_LIST_ITEMS.ITEM_ID = ITEM.ITEM_ID)
INNER JOIN RESERVE_LIST_COURSES ON RESERVE_LIST.RESERVE_LIST_ID =
RESERVE_LIST_COURSES.RESERVE_LIST_ID)
INNER JOIN COURSE ON RESERVE_LIST_COURSES.COURSE_ID = COURSE.COURSE_ID)
INNER JOIN (BIB_TEXT INNER JOIN BIB_ITEM ON BIB_TEXT.BIB_ID = BIB_ITEM.BIB_ID)
ON ITEM.ITEM_ID = BIB_ITEM.ITEM_ID
ORDER BY RESERVE_LIST.LIST_TITLE,
utf8to16([bib_text].[TITLE]), COURSE.COURSE_NAME;
```

All Physical Items on both Active and Expired Reserve Lists (Includes items that are linked to reserve lists which are not linked to a course)

Subquery: Save this Subquery with the name AllNonEItemsOnReserveSubquery and DO NOT run it.

```
SELECT RESERVE_LIST.RESERVE_LIST_ID, ITEM.ITEM_ID,
RESERVE_LIST.LIST_TITLE, RESERVE_LIST.EFFECT_DATE,
RESERVE_LIST.EXPIRE_DATE, utf8to16([bib_text].[TITLE]) AS Title,
utf8to16([bib_text].[AUTHOR]) AS Author,
ITEM_BARCODE.ITEM_BARCODE, RESERVE_LIST_COURSES.DEPARTMENT_ID,
```

```
RESERVE_LIST_COURSES.INSTRUCTOR_ID, RESERVE_LIST_COURSES.COURSE_ID
FROM (BIB_TEXT INNER JOIN (((MFHD_MASTER INNER JOIN MFHD_ITEM
ON MFHD_MASTER.MFHD_ID = MFHD_ITEM.MFHD_ID)
INNER JOIN (ITEM INNER JOIN ((RESERVE_LIST
LEFT JOIN RESERVE_LIST_COURSES
ON RESERVE_LIST.RESERVE_LIST_ID = RESERVE_LIST_COURSES.RESERVE_LIST_ID)
INNER JOIN RESERVE_LIST_ITEMS
ON RESERVE_LIST.RESERVE_LIST_ID = RESERVE_LIST_ITEMS.RESERVE_LIST_ID)
ON ITEM.ITEM_ID = RESERVE_LIST_ITEMS.ITEM_ID)
ON MFHD_ITEM.ITEM_ID = ITEM.ITEM_ID) INNER JOIN BIB_MFHD
ON MFHD_MASTER.MFHD_ID = BIB_MFHD.MFHD_ID)
ON BIB_TEXT.BIB_ID = BIB_MFHD.BIB_ID) LEFT JOIN ITEM_BARCODE
ON ITEM.ITEM_ID = ITEM_BARCODE.ITEM_ID
ORDER BY RESERVE_LIST.LIST_TITLE, RESERVE_LIST.EXPIRE_DATE;
```

Main Query: Save this Main query with the name AllNonEItemsOnReserveMainQuery and DO run it.

```
SELECT AllNonEItemsOnReserveSubquery.RESERVE_LIST_ID,
AllNonEItemsOnReserveSubquery.ITEM_ID,
AllNonEItemsOnReserveSubquery.LIST_TITLE,
AllNonEItemsOnReserveSubquery.EFFECT_DATE,
AllNonEItemsOnReserveSubquery.EXPIRE_DATE,
AllNonEItemsOnReserveSubquery.Title,
AllNonEItemsOnReserveSubquery.Author,
AllNonEItemsOnReserveSubquery.ITEM_BARCODE,
COURSE.COURSE_NAME, COURSE.COURSE_NUMBER,
DEPARTMENT.DEPARTMENT_NAME, DEPARTMENT.DEPARTMENT_CODE,
INSTRUCTOR.LAST_NAME AS [Instructor LAST_NAME],
INSTRUCTOR.FIRST_NAME AS [Instructor FIRST_NAME]
FROM ((AllNonEItemsOnReserveSubquery LEFT JOIN DEPARTMENT
ON AllNonEItemsOnReserveSubquery.DEPARTMENT_ID = DEPARTMENT.DEPARTMENT_ID)
LEFT JOIN INSTRUCTOR
ON AllNonEItemsOnReserveSubquery.INSTRUCTOR_ID = INSTRUCTOR.INSTRUCTOR_ID)
LEFT JOIN COURSE
ON AllNonEItemsOnReserveSubquery.COURSE_ID = COURSE.COURSE_ID
ORDER BY AllNonEItemsOnReserveSubquery.LIST_TITLE,
AllNonEItemsOnReserveSubquery.EXPIRE_DATE, COURSE.COURSE_NAME,
COURSE.COURSE_NUMBER, INSTRUCTOR.LAST_NAME, INSTRUCTOR.FIRST_NAME;
```

All eItems on both Active and Expired Reserve Lists (Includes eItems that are linked to reserve lists which are not linked to a course)

Subquery: Save the subquery with the name AllEItemsOnReserveSubquery, and DO NOT run it.

```
SELECT RESERVE_LIST.LIST_TITLE AS [Reserve List Name],
RESERVE_LIST.CREATE_DATE, RESERVE_LIST.EFFECT_DATE,
RESERVE_LIST.EXPIRE_DATE, BIB_TEXT.BIB_ID, EITEM.MFHD_ID,
EITEM.EITEM_ID, BIB_TEXT.TITLE_BRIEF,
```

RESERVE_LIST.RESERVE_LIST_ID,
RESERVE_LIST_COURSES.DEPARTMENT_ID,
RESERVE_LIST_COURSES.INSTRUCTOR_ID,
RESERVE_LIST_COURSES.COURSE_ID,
RESERVE_LIST_COURSES.SECTION_ID
FROM (((EITEM INNER JOIN (BIB_MFHD
INNER JOIN BIB_TEXT ON BIB_MFHD.BIB_ID = BIB_TEXT.BIB_ID)
ON EITEM.MFHD_ID = BIB_MFHD.MFHD_ID) INNER JOIN RESERVE_LIST_EITEMS
ON EITEM.EITEM_ID = RESERVE_LIST_EITEMS.EITEM_ID)
INNER JOIN RESERVE_LIST
ON RESERVE_LIST_EITEMS.RESERVE_LIST_ID = RESERVE_LIST.RESERVE_LIST_ID)
LEFT JOIN RESERVE_LIST_COURSES
ON RESERVE_LIST.RESERVE_LIST_ID = RESERVE_LIST_COURSES.RESERVE_LIST_ID
ORDER BY RESERVE_LIST.LIST_TITLE;

Main Query: Save the main query with the name AllEItemsOnReserveQuery and DO run it.

SELECT AllEItemsOnReserveSubquery.RESERVE_LIST_ID,
AllEItemsOnReserveSubquery.[Reserve List Name],
AllEItemsOnReserveSubquery.CREATE_DATE,
AllEItemsOnReserveSubquery.EFFECT_DATE,
AllEItemsOnReserveSubquery.EXPIRE_DATE,
AllEItemsOnReserveSubquery.BIB_ID,
AllEItemsOnReserveSubquery.MFHD_ID,
AllEItemsOnReserveSubquery.EITEM_ID,
AllEItemsOnReserveSubquery.TITLE_BRIEF,
COURSE.COURSE_NAME, COURSE.COURSE_NUMBER,
DEPARTMENT.DEPARTMENT_NAME, INSTRUCTOR.LAST_NAME
AS [Instructor LAST_NAME], INSTRUCTOR.FIRST_NAME
AS [Instructor FIRST_NAME]
FROM ((AllEItemsOnReserveSubquery LEFT JOIN DEPARTMENT
ON AllEItemsOnReserveSubquery.DEPARTMENT_ID = DEPARTMENT.DEPARTMENT_ID)
LEFT JOIN COURSE
ON AllEItemsOnReserveSubquery.COURSE_ID = COURSE.COURSE_ID)
LEFT JOIN INSTRUCTOR
ON AllEItemsOnReserveSubquery.INSTRUCTOR_ID = INSTRUCTOR.INSTRUCTOR_ID
ORDER BY AllEItemsOnReserveSubquery.[Reserve List Name],
AllEItemsOnReserveSubquery.EXPIRE_DATE, COURSE.COURSE_NAME,
COURSE.COURSE_NUMBER, DEPARTMENT.DEPARTMENT_NAME,
INSTRUCTOR.LAST_NAME, INSTRUCTOR.FIRST_NAME;

Pass through Access SQL query to identify reserve lists not linked to courses

Note that the following is a pass through SQL query

```
SELECT *
FROM reserve_list
WHERE reserve_list_id not in
(SELECT reserve_list_id FROM reserve_list_courses);
```

Pass through Access SQL queries to identify duplicate list names, courses or departments

Note that the following are pass through SQL queries:

```
SELECT reserve_list_id, list_title
FROM reserve_list
WHERE list_title in
(SELECT list_title FROM reserve_list
GROUP BY list_title having (count(list_title) > 1))
ORDER BY list_title;
```

```
SELECT course_id, course_name
FROM course
WHERE course_name in
(SELECT course_name
FROM course
GROUP BY course_name having (count(course_name) > 1))
ORDER BY course_name;
```

```
SELECT course_id, course_number
FROM course
WHERE course_number in
(SELECT course_number
FROM course
GROUP BY course_number having (count(course_name) > 1))
ORDER BY course_number;
```

```
SELECT department_id, department_name
FROM department
WHERE department_name in
(SELECT department_name
FROM department
GROUP BY department_name having (count(department_name) > 1))
ORDER BY department_name;
```

```
SELECT department_id, department_code
FROM department
WHERE department_code in
(SELECT department_code
FROM department
GROUP BY department_code having (count(department_code) > 1))
ORDER BY department_code;
```

For other queries that may be useful, see my "Surviving Almanado: tips for a successful pre-implementation" presentation's Accompanying Materials.

Posted as is. If you need assistance in running custom SQL queries in Prepackaged Access Reports, consult the Voyager Customer Listserv.

Report