

Cache and Release: Capturing and Using Sierra's Temporary SQL Data

Ray Voelker

<https://github.com/rayvoelker/2019-iug>



THE
PUBLIC
LIBRARY
of Cincinnati
and
Hamilton County

QUICK BACKGROUND

- Sierra's Direct SQL Access feature allows us to:
 - Quickly and efficiently **target** and **extract** real-time data from the Sierra ILS.
 - **Organize data** in logical and useful ways

QUICK BACKGROUND (CONT.)

- Why **save / preserve** data from Sierra?
 1. Data **transformations** and **integrations** for specific use cases

[wikipedia.org/wikipedia.org/wikipedia.org/wiki/Data_transformation](https://en.wikipedia.org/wikipedia.org/wikipedia.org/wiki/Data_transformation)

[wikipedia.org/wikipedia.org/wikipedia.org/wiki/Data_integration](https://en.wikipedia.org/wikipedia.org/wikipedia.org/wiki/Data_integration)

- For example, adding **geocoding** information to patron address data:

[wikipedia.org/wikipedia.org/wikipedia.org/wiki/Geocoding](https://en.wikipedia.org/wikipedia.org/wikipedia.org/wiki/Geocoding)

QUICK BACKGROUND (CONT.)

- Why save / preserve data from Sierra? (cont.)
 2. Cache Sierra data:
 - For use in an application instead of running an “expensive query” to deliver content
 - For use in an application where holding onto data that may otherwise be destroyed or transformed by the Sierra application itself

UNDERSTANDING SIERRA'S DATABASE VIEWS AND DATA

- Some data in sierra stays more **static**
(think of a “receipt”, or log of transactions)

circ_trans

Each row of circ_trans contains information about a circulation transaction.

Column	Data Type	Not NULL?	Comment
id	int	false	System-generated sequential ID.
transaction_gmt	timestamptz	false	Transaction date in UNIX format.
application_name	varchar	false	<p>The name of the program that generated the transaction. Valid program names are:</p> <ul style="list-style-type: none">• circ (includes transactions made using PC Circ)• circa (for transactions written by selfcheckwebserver and in-house use [transaction codes 'u' and 's'], which use webpac to execute transactions.)• milcirc• milmyselfcheck• readreq• selfcheck
source_code	varchar	false	<p>The transaction source. Possible values are:</p> <ul style="list-style-type: none">• local• INN-Reach• ILL
op_code	varchar	false	Type of transaction: i = checkin o = checkout

circ_trans

Each row of circ_trans contains information about a circulation transaction.

Column	Data Type	Not NULL?	Comment														
id	int	false	System-generated sequential ID.														
transaction_gmt	timestamptz	false	Transaction date in UNIX format.														
application_name	varchar	false	The name of the program that generated the transaction. Valid program names are: <ul style="list-style-type: none">• circ (includes transactions made using PC Circ)• circa (for transactions written by selfcheckwebserver and in-house use [transa• milcirc• milmyselfcheck• readreq• selfcheck														
source_code	varchar	false	The transaction source. Possible values are: <ul style="list-style-type: none">• local• INN-Reach• ILL														
op_code	varchar	false	Type of transaction: <table border="1"><tr><td>o = checkout</td><td>i = checkin</td></tr><tr><td>n = hold</td><td>h = hold with r</td></tr><tr><td>nb = bib hold</td><td>hb = hold reca</td></tr><tr><td>ni = item hold</td><td>hi = hold reca</td></tr><tr><td>nv = volume hold</td><td>hv = hold reca</td></tr><tr><td>f = filled hold</td><td>r = renewal</td></tr><tr><td>b = booking</td><td>u = use count</td></tr></table>	o = checkout	i = checkin	n = hold	h = hold with r	nb = bib hold	hb = hold reca	ni = item hold	hi = hold reca	nv = volume hold	hv = hold reca	f = filled hold	r = renewal	b = booking	u = use count
o = checkout	i = checkin																
n = hold	h = hold with r																
nb = bib hold	hb = hold reca																
ni = item hold	hi = hold reca																
nv = volume hold	hv = hold reca																
f = filled hold	r = renewal																
b = booking	u = use count																
patron_record_id	bigint	false	Foreign key to patron_record.														
item_record_id	bigint	false	Foreign key to item_record.														

UNDERSTANDING SIERRA'S DATABASE VIEWS AND DATA (CONT.)

- Circulation transactions are **created** in the database table and **remain static**
- **Rows are deleted** from the table after a certain period of time (2 weeks is the default, but this can be extended by iii upon request)

```
-- get some info about our circ_trans dates
SELECT
    NOW() ::TIMESTAMP WITH TIME ZONE as now_gmt,
    MAX(c.transaction_gmt) ::TIMESTAMP WITH TIME ZONE AS max,
    MIN(c.transaction_gmt) ::TIMESTAMP WITH TIME ZONE AS min,
    AGE(MIN(c.transaction_gmt)) AS earliest_transaction_age
FROM
    sierra_view.circ_trans as c
```

	now_gmt timestamp with time zone	max_circ_gmt timestamp with time zone	min_circ_gmt timestamp with time zone	earliest_transaction_age interval
1	2019-04-10 08:37:03.856585-04	2019-04-10 08:36:38-04	2019-03-26 21:43:24-04	14 days 02:16:36

UNDERSTANDING SIERRA'S DATABASE VIEWS AND DATA (CONT.)

- Other data is more **variable** or is a direct representation that describes a particular state of a record or process in the ILS.

hold

Each row of hold describes a bibliographic, item, or volume hold.

Column	Data Type	Not NULL?	Comment
id	bigint	false	System-generated sequential ID.
patron_record_id	bigint	false	Foreign key to patron_record.
record_id	bigint	false	Foreign key to record.
placed_gmt	timestamp	false	Date the hold was placed.
is_frozen	boolean	false	Specifies whether the hold is frozen (suspended).
delay_days	int	false	Stores the "not wanted before" date as a number of days after the date the hold was placed. The maximum value is "180". If a "not wanted before" date was not specified, the value is '0'.
location_code	varchar	false	For bib or volume-level holds, the branch location from which to fill the hold, if the hold is set for 'Limit to Location'. Does not apply to item-level holds (blank).
expires_gmt	timestamp	false	"Not needed after" date.
status	char	false	Hold status.

hold

Each row of hold describes a bibliographic, item, or volume hold.

Column	Data Type	Not NULL?	Comment						
id	bigint	false	System-generated sequential ID.						
patron_record_id	bigint	false	Foreign key to patron_record.						
record_id	bigint	false	Foreign key to record.						
placed_gmt	timestamp	false	Date the hold was placed.						
is_frozen	boolean	false	Specifies whether the hold is frozen (suspended).						
delay_days	int	false	Stores the "not wanted before" date as a number of days after the date the hold was specified, the value is '0'.						
location_code	varchar	false	For bib or volume-level holds, the branch location from which to fill the hold, if the ho						
expires_gmt	timestamp	false	"Not needed after" date.						
status	char	false	Hold status. <table border="1"><thead><tr><th>CODE</th></tr></thead><tbody><tr><td>0</td></tr><tr><td>b</td></tr><tr><td>j</td></tr><tr><td>i</td></tr><tr><td>t</td></tr></tbody></table>	CODE	0	b	j	i	t
CODE									
0									
b									
j									
i									
t									
is_ir	boolean	false	Specifies whether the hold is an INN-Reach hold.						
pickup_location_code	varchar	false	Pickup location code.						

UNDERSTANDING SIERRA'S DATABASE VIEWS AND DATA (CONT.)

- The **state of the hold** is defined in the Sierra database
- **Data changes** depending on the state or status of the hold, and is then **removed from the database** when the hold is deleted, filled or expires

```
-- this will select Ray Voelker's hold information from the
-- Sierra SQL database
SELECT
h.id,
h.patron_record_id,
h.record_id,
h.status,
h.pickup_location_code

FROM
sierra_view.hold as h

WHERE
h.patron_record_id = 481038535591;
```

	id bigint	patron_record_id bigint	record_id bigint	status character(1)	pickup_location_code character varying(5)
1	37661683	481038535591	450975488540	b	1
2	37332240	481038535591	420910217875	0	1

I checkout the item,
and the hold data
goes away after a
second query.



	id bigint	patron_record_id bigint	record_id bigint	status character(1)	pickup_location_code character varying(5)
1	37332240	481038535591	420910217875	0	1

HOW TO CACHE / TRANSFORM / PRESERVE DATA FROM SIERRA?

- No shortage of options!
 - pgAdmin is a popular choice for a desktop client
www.pgadmin.org
 - "Execute query, write result to file"
Creates a .csv file from the results

Query - iii on xrvoelke@sierra-db.cincinnatilibrary.org:1032 *

File Edit Query Favourites Macros View Help

SQL Editor Graphical Query Builder

Previous queries Delete Delete All

```
SELECT
r.id as bib_record_id,
r.record_num as bib_record_num,
r.deletion_date_gmt,
r.creation_date_gmt,
r.record_last_updated_gmt
|
FROM
sierra_view.record_metadata as r

WHERE
r.record_type_code || r.campus_code = 'b'
AND deletion_date_gmt::DATE = (NOW() - INTERVAL '1 day')::DATE
;
```

Output pane

Data Output Explain Messages History

	bib_record_id bigint	bib_record_num integer	deletion_date_gmt date	creation_date_gmt timestamp with time zone	record_last_updated_gmt timestamp with time zone
1	420910107544	3312536	2019-04-15	2017-10-26 15:05:54-04	2019-04-15 15:11:28.075-04
2	420910243879	3448871	2019-04-15	2019-04-15 13:28:26-04	2019-04-15 13:38:46-04
3	420910243865	3448857	2019-04-15	2019-04-15 12:16:34-04	2019-04-15 12:30:29-04
4	420910243878	3448870	2019-04-15	2019-04-15 13:11:26-04	2019-04-15 13:41:53-04
5	420910243866	3448858	2019-04-15	2019-04-15 12:24:28-04	2019-04-15 13:43:40-04
6	420910126230	3331222	2019-04-15	2018-01-29 12:15:40-05	2019-04-15 15:11:56.498-04
7	420910243885	3448877	2019-04-15	2019-04-15 15:05:28-04	2019-04-15 15:08:41-04
8	420909774705	2979697	2019-04-15	2014-07-16 10:30:35-04	2019-04-15 15:09:33.16-04
9	420910033145	3238137	2019-04-15	2016-12-19 16:09:50-05	2019-04-15 15:10:01.774-04
10	420910037920	3242912	2019-04-15	2017-01-27 13:01:09-05	2019-04-15 15:10:53.751-04

OK. Unix Ln 7, Col 1, Ch 131 30 rows. 6.4 s...

HOW TO CACHE / TRANSFORM / PRESERVE DATA FROM SIERRA? (CONT.)

- Many programming languages provide access to PostgreSQL via their libraries:
 - `php-pgsql`: PHP PostgreSQL driver
www.php.net/manual/en/bookpgsql.php
 - `psycopg2`: Python PostgreSQL adapter
initd.org/psycopg/docs/

HOW TO CACHE / TRANSFORM / PRESERVE DATA FROM SIERRA? (CONT.)

- My method consists of the following overview:
 1. Use **Python** to connect to Sierra's database
 2. Issue **SQL statement** on Sierra's database to target and compile the data for **extraction**

HOW TO CACHE / TRANSFORM / PRESERVE DATA FROM SIERRA? (CONT.)

- My method consists of the following overview (cont.):
 3. **Export result data** to either a **.csv file**, and/or directly into a **SQLite database**
 - **.CSV** files are easy to later load into an SQLite database, spreadsheet, or other data warehouse tool

HOW TO CACHE / TRANSFORM / PRESERVE DATA FROM SIERRA? (CONT.)

SQLite Database: sqlite.org

*"SQLite is a C-language library that implements a **small, fast, self-contained, high-reliability, full-featured, SQL database engine.**"*

HOW TO CACHE / TRANSFORM / PRESERVE DATA FROM SIERRA? (CONT.)

SQLite Database (cont.)

- SQLite database engine is a great tool for caching data:
 1. Stores and organizes large amounts of data quickly and efficiently
 2. You don't have to set up and maintain a server (data is portable; entire database is contained in a single, cross-platform file)

HOW TO CACHE / TRANSFORM / PRESERVE DATA FROM SIERRA? (CONT.)

SQLite Database (cont.)

3. Ability and flexibility to **build SQL queries** and applications that directly use the data that you've collected
4. Has a useful desktop tool: sqlitebrowser.org
5. It's included in the **Python Standard Library!**

```
# Python sample code to create database,
# create a table, and then insert some data
# note: sqlite3 is part of the Python Standard Library
import sqlite3

# create the database
conn = sqlite3.connect('example.db')
cursor = conn.cursor()

# create the table
cursor.execute("""CREATE TABLE IF NOT EXISTS `data` (
    `id` INTEGER PRIMARY KEY AUTOINCREMENT,
    `input` TEXT )""")

# insert some data
cursor.execute("""INSERT INTO data (`input`)
    VALUES ('sample text')""")

# commit and close the connection
conn.commit()
conn.close()
```

EXAMPLES:

1. Collection Analysis / Circulation Data Analysis

- intended for use with the **CollectionHQ** service,
but can be adapted for local use

2. Patron Savings Calculator

- intended to display information on the Encore
patron account information page about **how
much money has been "saved"** by using the
library

EXAMPLES: (CONT.)

3. Hold Shelf Delivery Matrix Report

- intended to produce a spreadsheet report of items for **hold shelf locations** (ready for patron pickup) and **from where they originated**

4. Mapping Geo Data from Patron Address Data and Circulation Transaction Data

- intended to take geocoded patron address data and produce a **visualization** by plotting it **on a map**

EXAMPLE 1

COLLECTION ANALYSIS / CIRCULATION DATA ANALYSIS

- github.com/plch/collection-analysis
- This application was built for the purpose of automatically preserving and **sending data** from the Sierra database to the **CollectionHQ** service via FTP

COLLECTION ANALYSIS / CIRCULATION DATA ANALYSIS (CONT.)

- Three main groups of data are targeted for export:
 1. **Bibliographic Record Data:**
[`temp_table-bib_data.sql`](#)
Fields exported include: bib record number, control numbers (var fields tagged 'o'), isbn, call numbers, and a few other metadata fields related to the bib record

COLLECTION ANALYSIS / CIRCULATION DATA ANALYSIS (CONT.)

- Three main groups of data are targeted for export:
(cont.)

2. Item Record Data:

[`temp_table-item_data.sql`](#)

Fields exported include: item record number, bib record number, circulation information, price, and a few other metadata fields related to the item record

COLLECTION ANALYSIS / CIRCULATION DATA ANALYSIS (CONT.)

- Three main groups of data are targeted for export:
(cont.)

3. Hold Data:

[`temp_table-hold_data.sql`](#)

Weekly snapshot of bib-level holds. It is organized by title (bib record number), and then each title has a list of holds with related metadata for each hold (patron number, pickup location, created date, expiration date, etc)

COLLECTION ANALYSIS / CIRCULATION DATA ANALYSIS (CONT.)

- Overview of the process:
 1. Create and open .csv files to hold data output and/or open local database (if it's useful to place data into local SQLite database)
 2. Connect to remote Sierra database, and create the temporary tables that will be used for exporting

COLLECTION ANALYSIS / CIRCULATION DATA ANALYSIS (CONT.)

- Overview of the process (cont.):
 3. Generate data output from the Sierra database temporary tables, and write the output to a .csv file and/or to a local SQLite database
 4. Send data via FTP
- full export script can be found in the public github repository for this project

EXAMPLE 2

PATRON SAVINGS CALCULATOR

- github.com/plch/patron-savings-calculator
- This application was designed to work with the iii discovery layer Encore's account detail page, to display how much a patron has "saved" by borrowing from the library

Encore - Mozilla Firefox

Encore



https://catalog.cincinnatilibrary.org/iii/encore/



Search



My Book Cart (0 items) | [i](#) | My Account | [Logout](#)

Search



[Advanced Search](#)

[Suggest a Purchase](#)

[Back to previous page](#)

Voelker, Ray L

Home Library:

Email:

Main Library

ray.voelker@gmail.com

Approximate Savings*:

\$2109.28 (since 8/22/2018)

[Edit account](#)

[Edit pin](#)

Checkouts (21)

[Print](#)

Holds (1)

[Sort by Checkout](#)

[Renew All](#)

[Renew Marked](#)

My OverDrive eBooks (0)

Fines/Fees (\$0.00)

Reading History

Preferred Searches

My Lists

RSS Feed

21 items checked out

RENEW	TITLE	BARCODE	STATUS	CALL NUMBER
<input type="checkbox"/>	Spider-Man. The complete clone saga epic / writers, Tom DeFalco [and others] ; pencils, Mark Bagley [and others] ; inkers, Fred Fredericks [and others]. v.03	A000052935020	DUE 04-13-19 Renewed 5 times	741.5 qD313sc 2016
<input type="checkbox"/>	Everybody sleeps (but not Fred) / Josh Schneider.	A000046864542	DUE 04-16-19	Easy
<input type="checkbox"/>	Your pal Mo Willems presents Leonardo the terrible	A000014405799	DUE 04-16-19	Easy

PATRON SAVINGS CALCULATOR (CONT.)

- Cached “**savings**” information is based on the patron record number and can be output in multiple formats (based on this application's custom URL endpoints):
 - JSON
 - /api/v1/patron_savings/2198439

```
{  
    "count_titles": 118,  
    "min_date_epoch": 1534957500,  
    "patron_record_num": 2198439,  
    "total_savings": 2109.279999999997  
}
```

PATRON SAVINGS CALCULATOR (CONT.)

- Cached “**savings**” information is based on the patron record number and can be output in multiple formats (based on this application's custom URL endpoints):

- PNG

/api/v1/patron_savings/img/2198439

By using The Public Library
of Cincinnati and Hamilton County,
you have saved approximatly
\$2109.28 Since 08/22/2018
Cha-ching!!!

PATRON SAVINGS CALCULATOR (CONT.)

- About this very simple RESTful API
 - Written as a Python / Flask application:
 - github.com/plch/patron-savings-calculator/blob/master/app.py
 - www.palletsprojects.com/p/flask

PATRON SAVINGS CALCULATOR (CONT.)

- About this very simple RESTful API (cont)
 - Hosted on an Apache Web Server via the WSGI module
 - A good tutorial on how this can be set up can be found here:
www.digitalocean.com/community/tutorials/how-to-deploy-a-flask-application-on-an-ubuntu-vps

PATRON SAVINGS CALCULATOR (CONT.)

- About this very simple RESTful API (cont)
 - Draws data from the SQLite database that has been **caching relevant data**
 - SQLite database, `patron_savings.db` is updated frequently (every 5 minutes) via a **Python update script**:
github.com/plch/patron-savings-calculator/blob/master/update.py

PATRON SAVINGS CALCULATOR (CONT.)

About the `update.py` script:

- Script starts by looking at the **last entry** it received from the **Sierra database**
- A query is constructed to **extract relevant data from the Sierra database** that is **more recent** than that **last entry** in the local database
- The local database is updated with the **fresh data** from the Sierra database

PATRON SAVINGS CALCULATOR (CONT.)

A note about privacy / protection / obfuscation of this data:

- No title information is saved in the local database, other than a hashed bib record id
- Hashed bib record id is stored to avoid duplicating the price information when a title is checked out multiple times, and to differentiate titles from one another in the local database

PATRON SAVINGS CALCULATOR (CONT.)

A note about obfuscation with this data (cont.):

- No patron information, other than the patron record number, is stored in the local database
- Only price and number of titles checked out are surfaced via the RESTful API

EXAMPLE 3

HOLD SHELF DELIVERY MATRIX REPORT

- github.com/plch/plch-holds-shelf
- The purpose of this application is to create a spreadsheet that displays items delivered to a hold shelf location, and from what location they came from

HOLD SHELF DELIVERY MATRIX REPORT (CONT.)

HOLD SHELF DELIVERY MATRIX REPORT (CONT.)

SELECT FROM `sierra_view.hold` WHERE `status` =

code definition

"b" Bib hold ready for pickup

"j" Volume hold ready for pickup

"i" Item hold ready for pickup

HOLD SHELF DELIVERY MATRIX REPORT (CONT.)

- Source location:
 - `checkin_statistics_group_code_num`
found in `sierra_view.item_record` table view

HOLD SHELF DELIVERY MATRIX REPORT (CONT.)

- Destination location:
 - `pickup_location_code`
found in `sierra_view.hold`

HOLD SHELF DELIVERY MATRIX REPORT (CONT.)

- Note: the **source** value of `location_code` comes from
``sierra_view.statistic_group_myuser``
using ``checkin_statistics_group_code_num``
effectively giving us the **pickup location** from the
stat group code num

HOLD SHELF DELIVERY MATRIX REPORT (CONT.)

- Do not insert duplicate rows of hold data...
 - We could build a complicated set of comparisons of the remotely selected data to our local data ...
 - Or, we could **let the databases do all the work!**
 - Create a **hash of the entire hold row**, use that value as the **unique primary key** in the local database table

HOLD SHELF DELIVERY MATRIX REPORT (CONT.)

Local SQLite table creation (simplified)

```
-- local SQLite table creation (simplified) ...
CREATE TABLE IF NOT EXISTS "data" (
    `hash_row` TEXT UNIQUE PRIMARY KEY
    -- more columns created below ...
);
```

HOLD SHELF DELIVERY MATRIX REPORT (CONT.)

Remote Sierra database query (simplified)

```
-- remote Sierra database query (simplified) ...
SELECT
MD5(CAST((h.*) AS TEXT)) AS hash_row
-- more columns of data selected below
FROM
sierra_view.hold AS h
WHERE
h.status IN(
    'b', 'j', 'i'
);
```

HOLD SHELF DELIVERY MATRIX REPORT (CONT.)

Remote Sierra database query (simplified) output:

	hash_row
	text
1	59752190f26bda0ec17167010cc6619f
2	ce971535718d416fd09e9266797c9374
3	d190eedbd12bdb434f6eaf8fdcaa9c2a5
4	dc2befdf5f6c9e356ef4f7d0885775a55
5	21915024b2b26de216edd6f6a9fc572a
6	0Ef5a22a562aa0E7120a6d29b00aabb2a

HOLD SHELF DELIVERY MATRIX REPORT (CONT.)

Local SQLite table inserting the retrieved data
(simplified):

```
INSERT OR IGNORE INTO data (
    hash_row
)
VALUES ('59752190f26bda0ec17167010cc6619f') ,
       ('59752190f26bda0ec17167010cc6619f') ,
       ('ce971535718d416fd09e9266797c9374');

SELECT * FROM DATA;
```

	hash_row
1	59752190f26bda0ec17167010cc6619f
2	ce971535718d416fd09e9266797c9374

HOLD SHELF DELIVERY MATRIX REPORT (CONT.)

- Overview of the process: (this should look familiar)
 1. Connect to our local database, and create table if it doesn't exist. Also, establish connection to the remote Sierra database
 2. Query the Sierra 'hold' table for rows that have status of 'i', 'j', or 'b'; this indicates that there is a item hold, volume hold, or bib hold ready for pickup

HOLD SHELF DELIVERY MATRIX REPORT (CONT.)

- Overview of the process: (cont.)
 3. Insert retrieved rows (or ignore duplicate rows as explained previously) to the local SQLite database, then close all connections
 4. Set the update of local data to happen frequently (every 5 minutes via CRON is a good method for doing this)

HOLD SHELF DELIVERY MATRIX REPORT (CONT.)

- Exporting / Producing the **report**
 1. Query the local SQLite database, then export the results to a .csv file:

github.com/plch/plch-holds-shelf/blob/master/export_csv.sh

github.com/plch/plch-holds-shelf/blob/master/export.sql

HOLD SHELF DELIVERY MATRIX REPORT (CONT.)

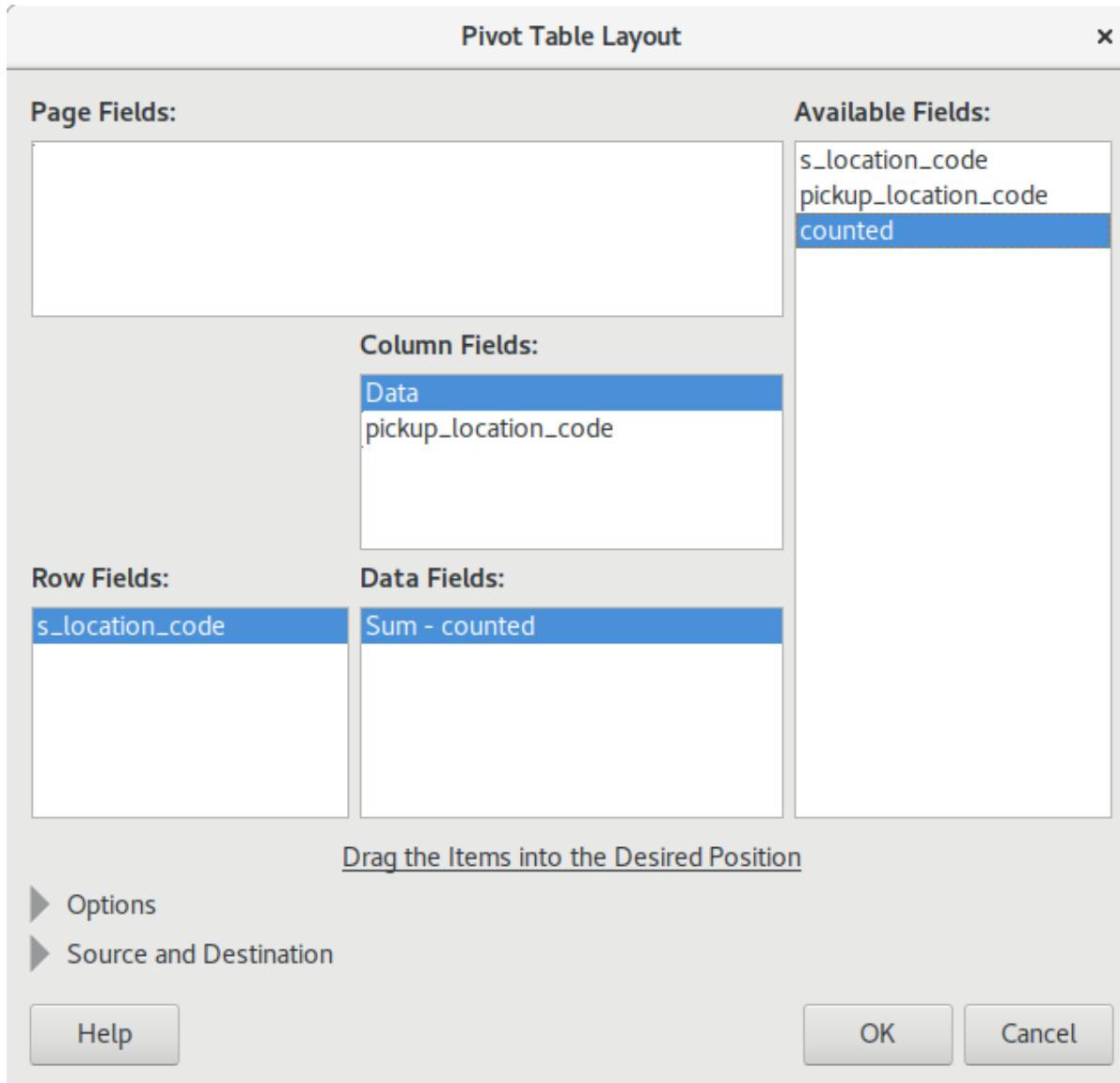
- Exporting / Producing the report (cont.)
 2. Import the .csv file into LibreOffice Calc (or Excel) and perform a pivot on the data:

HOLD SHELF DELIVERY MATRIX REPORT (CONT.)

- Exporting / Producing the report (cont.)

HOLD SHELF DELIVERY MATRIX REPORT (CONT.)

- Exporting / Producing the report (cont.)



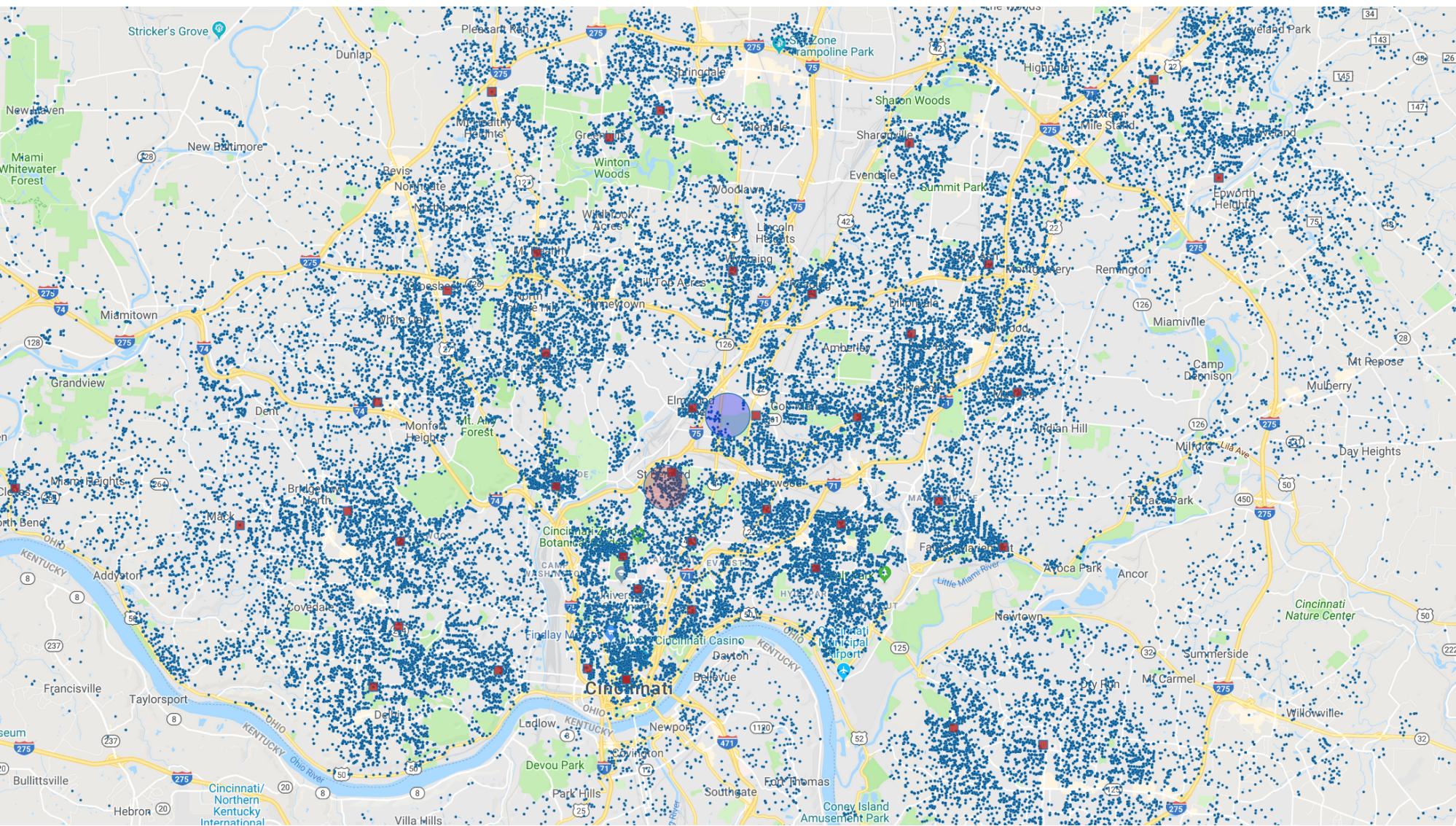
HOLD SHELF DELIVERY MATRIX REPORT (CONT.)

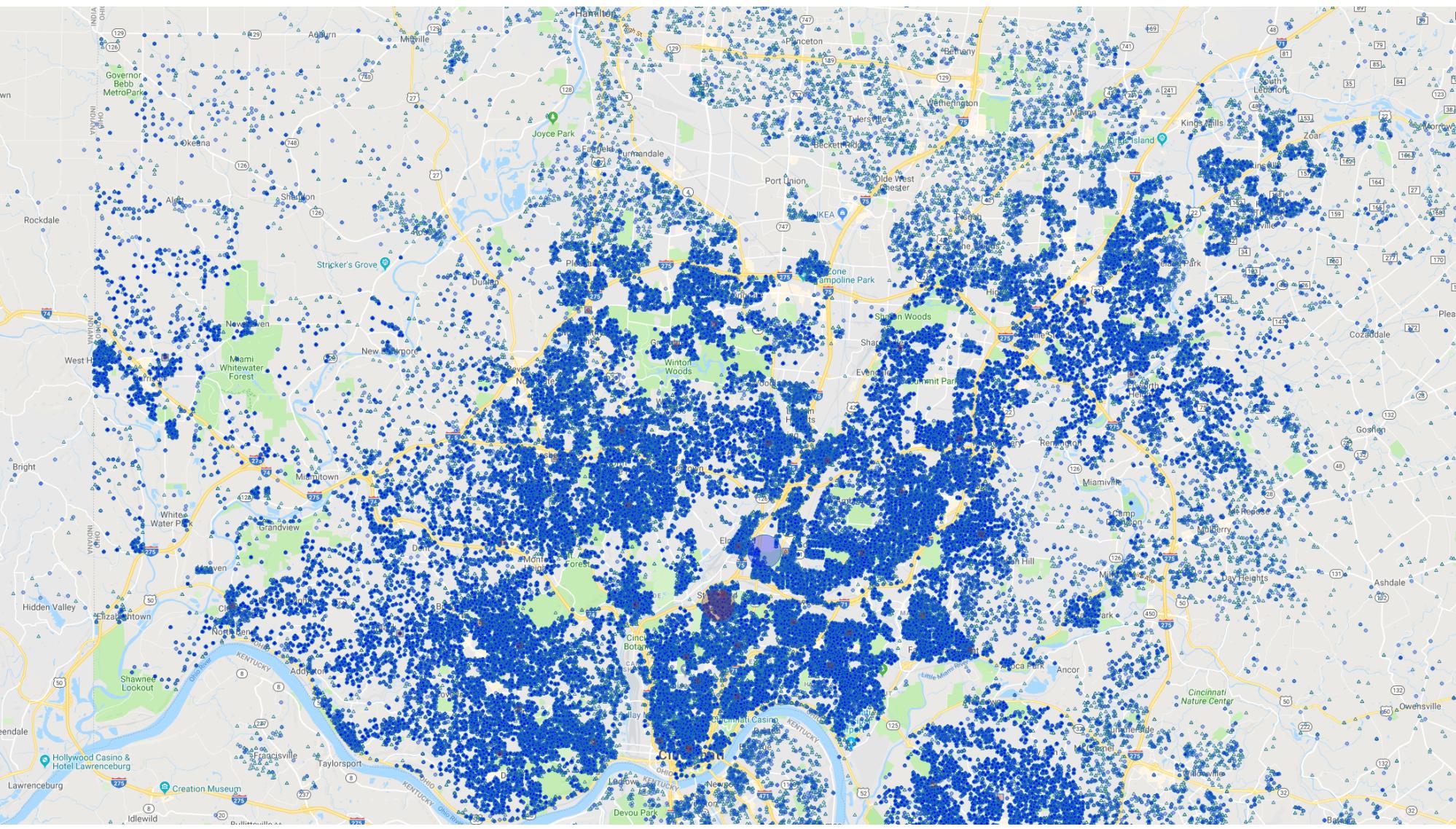
- Exporting / Producing the report (cont.)

EXAMPLE 4

MAPPING GEO DATA FROM PATRON ADDRESS DATA AND CIRCULATION TRANSACTION DATA

- The purpose of this process is to plot patron locations and branch locations on a map based on latitude / longitude coordinates derived from mailing address data
- This is a **work in progress!**





MAPPING GEO DATA (CONT.)

- Cached data is contained in an SQLite database
 - Circulation data
(weekly export of `sierra_view.circ_trans` table view)
 - Patron data
(weekly export of relevant patron information from multiple table views)

MAPPING GEO DATA (CONT.)

- Geocoding Patron Street Addresses:
 - Patron address data (**patron_record_id**, street number, street name, city, zip) are exported to a .csv file

MAPPING GEO DATA (CONT.)

- Possible **Geocoding Services**:

- Census.gov

- www.census.gov/data/developers/data-sets/Geocoding-services.html

- Google

- developers.google.com/maps/documentation/java

- **SmartyStreets**

- smartystreets.com/products/list

MAPPING GEO DATA (CONT.)

- **SmartyStreets** has some very user-friendly services for bulk upload / download of address data for verification and geocoding
- SmartyStreets may be able to offer a discounted / free service to libraries that allow for bulk verification / geocoding as well as on-the-fly verification and auto-corrected address inputs for things such as web input forms

MAPPING GEO DATA (CONT.)

- SmartyStreets list service returns .csv data back with relevant address information, identified by the unique ID (``patron_record_id``) that was provided
- We may easily load this into the local SQLite database with the import csv feature (from the GUI)

MAPPING GEO DATA (CONT.)

- There are very good data analysis / visualization tools available for Python:
 - PyViz
pyviz.org
 - A Conda metapackage "pyviz"
 - Makes data visualization in Python easier to use

jupyter_notebook_example.pdf