HOW TO

USE SQL SERVER

FOR PUBLIC LIFE STUDIES

SF Planning Department

SVETHA AMBATI
PUBLIC LIFE INTERN
SUMMER 2017

ACKNOWLEDGEMENTS

The implementation of the new Public Life Study Database and visualization tools would not be possible without support from Robin Abad Ocubillo, Mike Wynn, Mike Webster, Michael Sanders, Genta Yoshikawa, Teresa Ojeda, and Scott Edmondson of the Planning Department.

My gratitude also goes to previous interns who put together the foundation on which the new database is built.

San Francisco Planning

SEAL OF THE CITY AND COUNTY OF SAN FRANCISCO

TABLE OF CONTENTS

LIST OF FIGURES

[PHOTO CREDIT: MISSION PUBLIC LIFE STUDY]

WHAT IS A PUBLIC LIFE STUDY?

Public Life Studies are critical to our understanding of how public spaces function. Through careful and systematic observation we are able to understand if public spaces serve the needs of people, including dimensions of comfort, safety, and ease of mobility for pedestrians. Typical Public Life Studies involve counting pedestrians and cyclists, and an inventory of stationary activities and behaviors. The findings of these surveys and observations inform strategies to change the public realm, as well as help us understand the impacts of changes. The data gathered from studies provides insight into when, where, and why people are using public spaces. Understanding this basic information can lead to ideas about how the space can function better to support a lively atmosphere, and how to improve the quality of the space.

TYPICAL PUBLIC LIFE STUDY PHASES

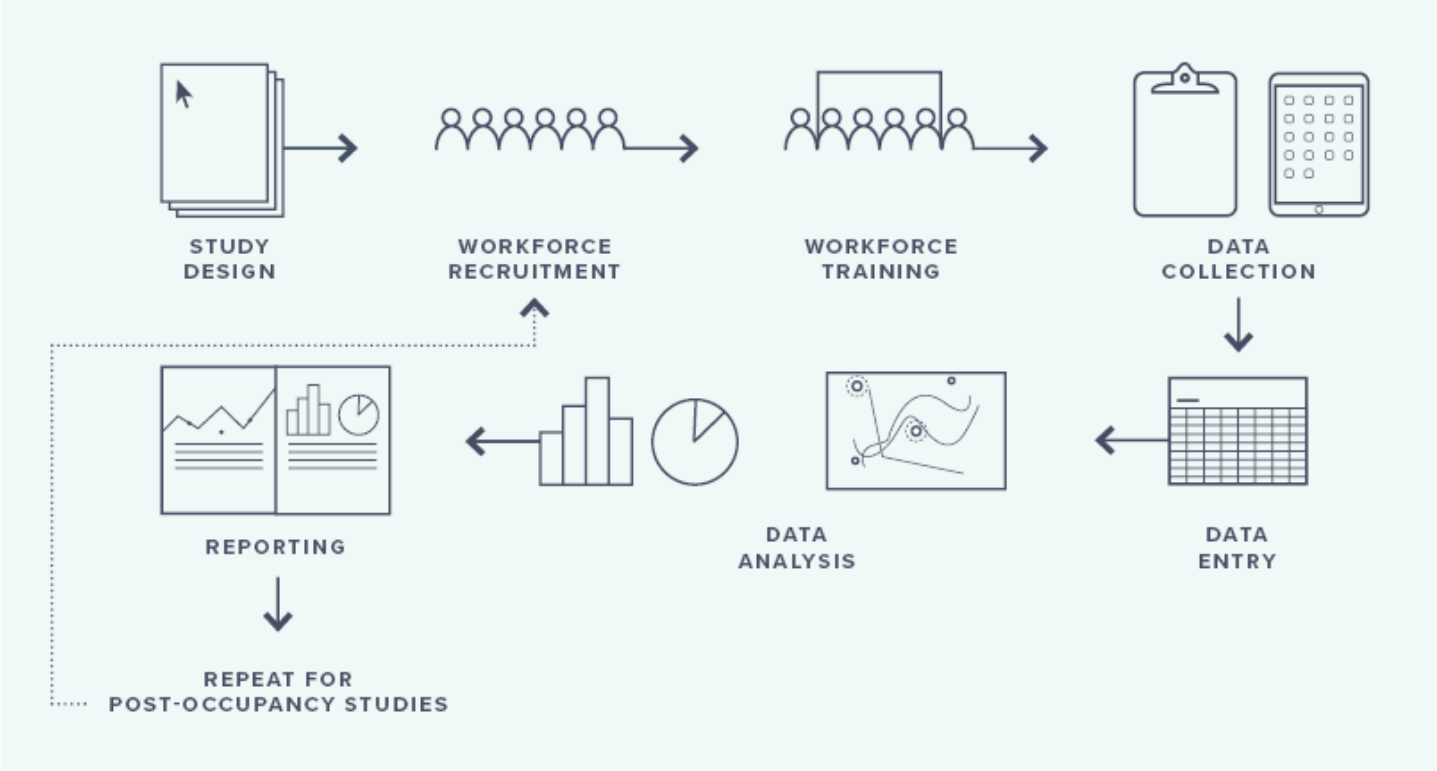


FIGURE 1: TYPICAL PUBLIC LIFE STUDY PHASES

PUBLIC LIFE STUDY DATABASE

Public Life Studies are stored in a database to ensure that data collection follows a standard format. The database also allows planners to join spatial data for pedestrian and bicycle counts, activity maps, and user intercept surveys with other data in order to analyze trends.

Public Life Studies have been collected by the City Planning Division since 2007. Since then, the standards of data collection have been transformed and updated in order to reflect consistency and adaptability for various types of studies. Consistency of data collection helps planners compare studies done in the same areas but different years, and also lends data integrity to the information collected.

The following diagrams illustrate the “old-school” workflow of collected data for public life studies and importing the new data into a Microsoft Access Database, followed by a newer workflow illustrating flexibility of data collection methods and automated importing of data to a Microsoft SQL Server on an Amazon Web Server.

FIGURE 2: “OLD-SCHOOL” WORKFLOW

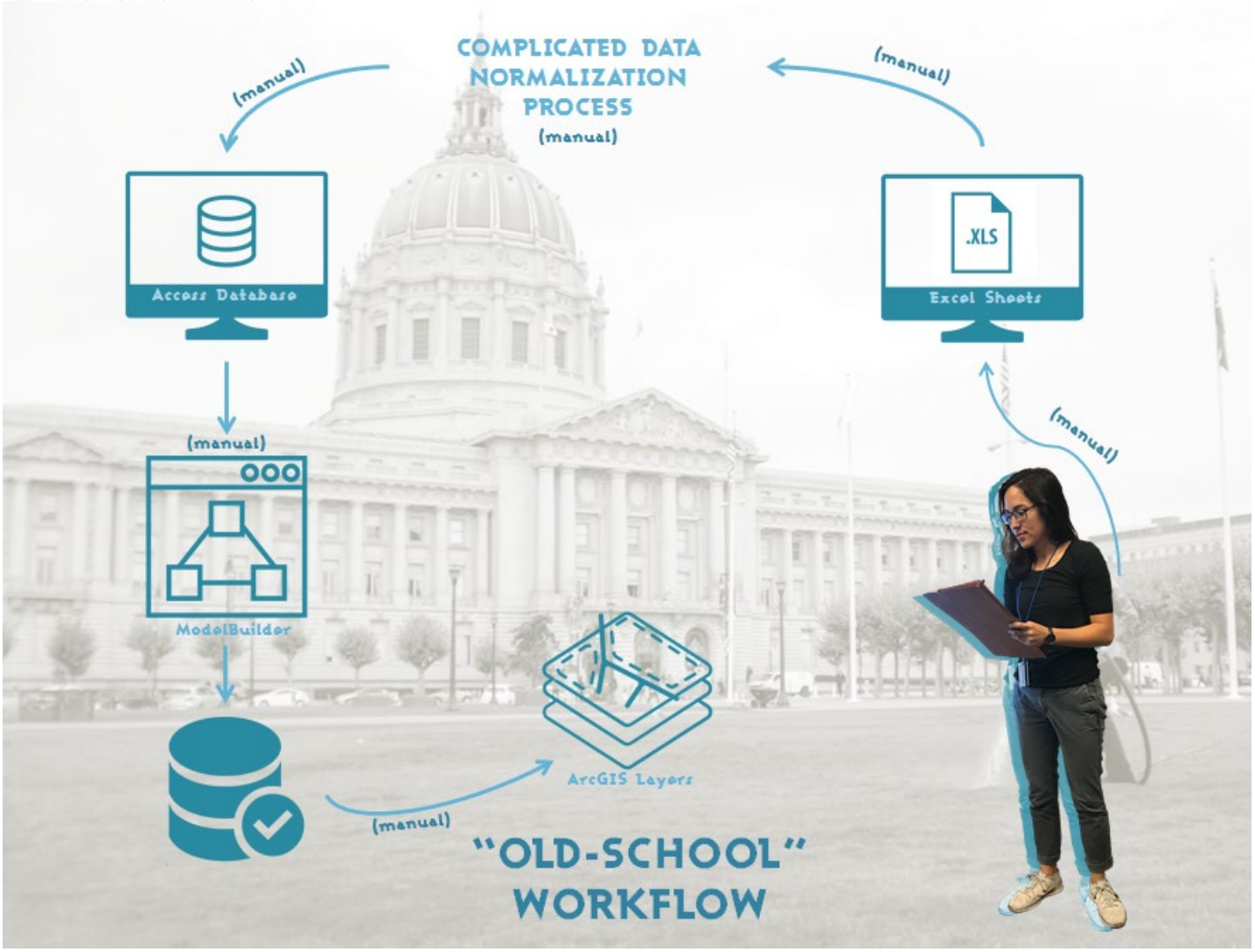


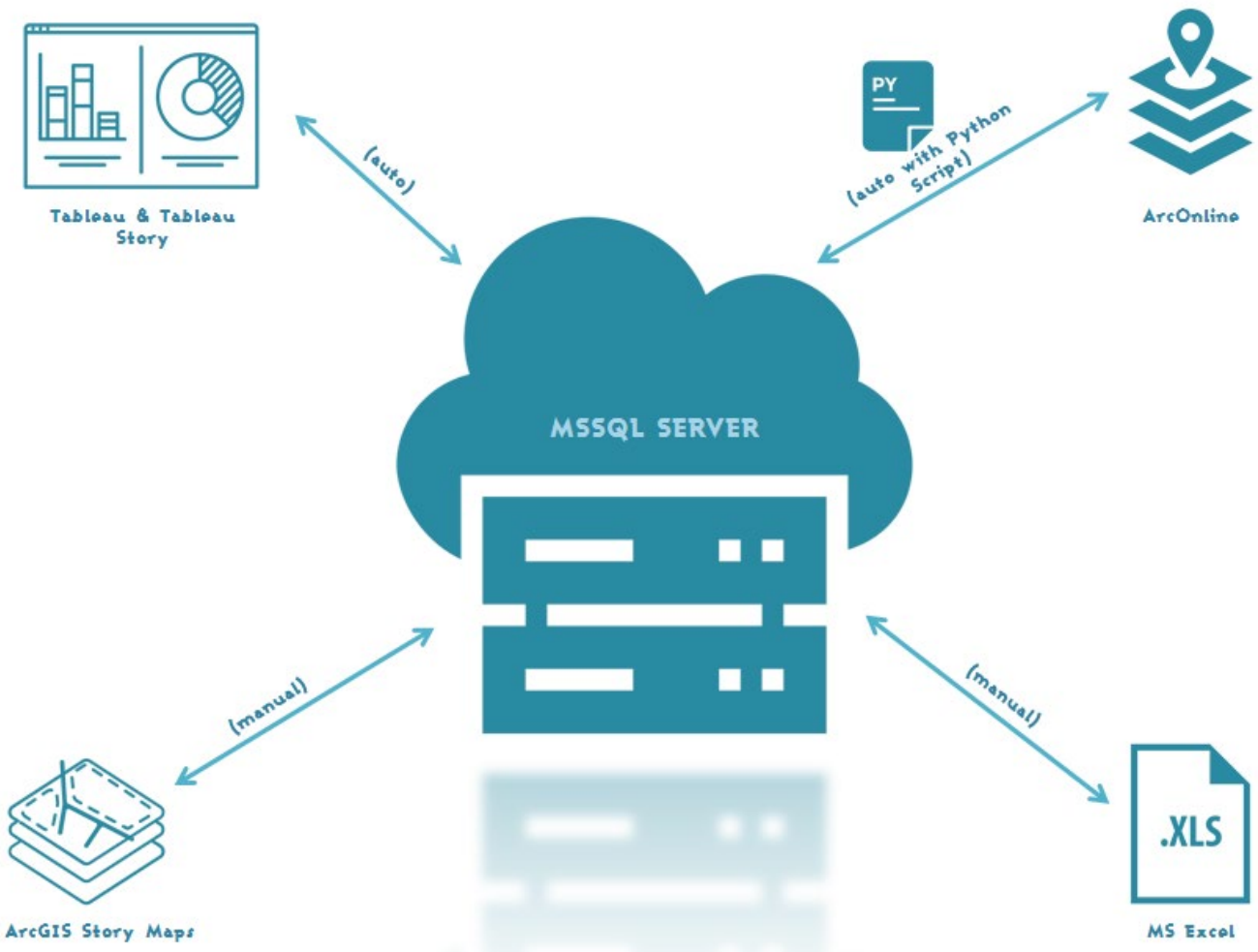
FIGURE 3: NEW WORKFLOW

BENEFITS OF THE PLS DATABASE

Public Life Studies are currently stored in a local Microsoft Access database. The Access database only allows one person to alter the database at a time, therefore restricting how many people can input data simultaneously after a public life study has been completed. Additionally, it is a local copy only accessible from an internal network. The online SQL Server database allows planners to access the database from a website, and to efficiently input data multiple people at a time to produce faster turnaround time for analysis.

The following diagram illustrates how the online database introduces automatic functions for processes that currently rely on manual operations. For example, for data analysis purposes, the online database automatically links to data visualization software such as Tableau, and the connection can be set up to refresh automatically to reflect any new data input. With the creation of a Python script, a similarly automatic connection can be set up with ArcOnline to input data collected by ArcGIS services such as Survey123 or Mobile Collector. A planner can also manually download Microsoft Excel files to conduct analysis within Excel, or manually append data to an existing ArcGIS feature class layer for use in ArcMap.

FIGURE 4: INTERACTIONS WITH THE SERVER DATABASE

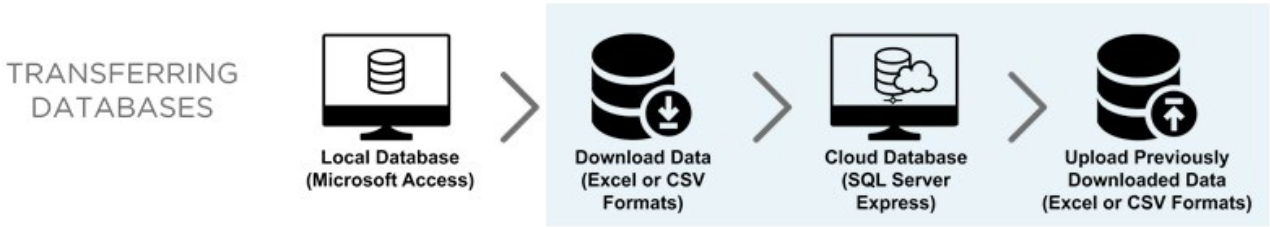


[PHOTO CREDIT: MISSION PUBLIC LIFE STUDY]

SETTING UP THE PLS DATABASE

The new Public Life Study Database is located on an Amazon Web Server (AWS) hosted on a remote Windows desktop. In setting up the new SQL Server database, the following implementation steps are followed:

STEP ONE



The initial step in setting up the SQL Server database is to insert data from the existing Microsoft Access Database. The following workflow details the steps taken to create the SQL Server database:

1 CREATE SQL SERVER EXPRESS DATABASE

1. Open up SQL Server Express

2. Right-click on “Databases”

3. Enter name of database under “Database Name” field

2 CREATE NEW ADMIN LOGIN

1. Expand the “Security” folder

2. Right-click on “Logins”

3. Select “New Login”

4. Enter name under “Login Name” field

5. Select “SQL Server Authentication”

6. Enter password under “Password” field

7. Go to “Server Roles” on left-hand panel

8. Select “All Server Roles”

9. Go to “User Mapping”

10, Select the database you created

11. Select “Database Role Memberships”

12. Go to “Securables” and make sure the “Connect SQL” permission is granted under the Permissions window

13. Go to “Status” and make sure permission is granted to connect to the database engine and that the login is enabled

3 CONFIGURE SERVER TO ACCEPT REMOTE CONNECTIONS

[ENABLE REMOTE CONNECTIONS ON SQL SERVER INSTANCE]

1. Open SQL Management Studio

2. Right-click on server name

3. Select “Properties”

4. Go to “Connections” on the left-hand panel

5. Check “Allow Remote Connections to this Server”

[CONFIGURE SQL SERVER TO LISTEN ON STATIC PORT]

6. Open SQL Server Configuration Manager

7. Go to “SQL Server Services” in the left-hand panel

8. In the center pane, look for the PID in the row for SQL Server

9. Open up Command Line Prompt

10. Enter the following prompt: “nestat-ano | find/I <enter PID here>”

11. If the prompt returns Port 1433, it has been configured

12. If there are no results, or Port 1433 is not returned, follow these steps:

- Click on SQL Server Network Configuration in the SQL Server Configuration Manager

- Right-click “TCP/IP protocol” from the center pane and select “Enable”

- Right-click the SQL Server and select “Restart”

- Retry “nestat-ano | find/I <enter PID here>” in Command Line Prompt

13. If prompt does not return 1433 then follow these steps:

- Click on SQL Server Network Configuration in the SQL Server Configuration Manager

- Right-click “TCP/IP protocol” from the center pane

- Select “Properties”

- Go to IP Address tab and scroll to the APAll section

- If 1433 is not entered in the TCP Port section, replace the number there with “1433”

- Right-click the SQL Server and select “Restart”

- Retry “nestat-ano | find/I <enter PID here>” in Command Line Prompt

[TURN ON SQL SERVER BROWSER SERVICE]

14. Open SQL Server Configuration Manager

15. Click on “SQL Server Services” in the left-hand panel

16. Right-click “SQL Server Browser Service”

17. Select “Properties” and go to the “Service” tab

18. Under “Start Mode Option,” select “Automatic”

19. Click “Start”

20. Confirm that the state has been changed to “Running” for the SQL Server Browser in the center pane

[CONFIGURE FIREWALL TO ALLOW SQL SERVER NETWORK TRAFFIC]

21. Open up Windows Firewall from Desktop

22. Create TCP Rule:

- Select “New Rule” and then “Port”

- Select “TCP” and enter Port 1433

- Allow the Connection, and choose all three profiles (Domain, Private, Public)

- Name the rule “SQL-TCP 1433”

23. Create UDP Rule:

- Select “New Rule” and then “Port”

- Select “UDP” and enter Port 1433

- Allow the Connection, and choose all three profiles (Domain, Private, Public)

- Name the rule “SQL-UDP 1433”

24. Create program exception:

- Select “New Rule” and then “Program”

- Browse to sqlservr.exe in the location field

- Allow the Connection, and choose all three profiles (Domain, Private, Public)

- Name the rule “SQL-sqlservr.exe”

4 ENABLE SQL SERVER ODBC DATA SOURCE

1. Open up ODBC Data Source Administrator from Desktop

2. Go to “User DSN” tab or “System DSN” tab

3. Click “Add”

4. Select “SQL Server Native Client 11.0” to select the driver

5. Click “Finish”

6. Follow instruction prompts in the wizard pop-up

5 ADD DATA TO SQL SERVER DATABASE

1. Download the Office System Driver:

2. Start the SQL Import and Export Wizard

3. Choose the Excel spreadsheet as the Data Source (drop-down menu of “Choose a Data Source” page)

4. Select the version in the Excel Version drop-down list

5. Make sure that the “First row has column names” field is checked

6. Click “Next”

7. Choose the SQL Database as the destination (drop-down menu of “Destination”)

8. Enter server IP address in the “Server Name” field

9. Check “Use SQL Server Authentication” under the “Authentication” section

10. Enter the new user and password created in the “Create New Admin” steps

11. In the “Database” drop-down menu, choose the name of the database

12. Click “Next” and select the option to “Copy data from one or more table views”

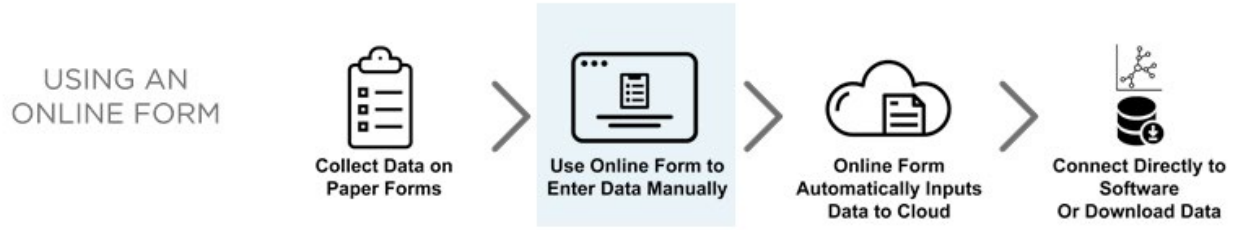
13. Click “Next” again and select the source table by checking the sheet name or number

14. Click “Next”

15. Check “Run Immediately” and click “Next”

16. Verify that the package executed successfully

STEP TWO



The second step in setting up the SQL Server database is to build online forms for data entry. The following workflow details the steps taken to create the forms on Visual Basic Studio 2017:

1 SETTING UP VISUAL BASIC STUDIO

1. Open up Visual Basic Studio 2017

2. Select “Project Templates” to choose the template

3. Under the C# menu, select “Windows Form”

4. A Windows Form template will open up in the window with a default solution name

2 CONNECTING TO SQL SERVER

1. In the Solution page of Visual Basic Studio, expand the Server Explorer tab in the left-hand panel

2. Right-click “Data Connections” and select “Add Connection”

3. Under Data Source, select “Microsoft SQL Server”

4. Select the server under “Server Name”

5. Under Authentication, select “SQL Server Authentication” and enter the admin user name and password

6. Select the database name under the “Connect to a Database” section and click “Test Connection”

7. If connection is successful, click “OK”

3 CONNECTING TO DATA SOURCE

1. In the Solution page of Visual Basic Studio, expand the Data Sources tab in the left-hand panel

2. Click on the Database Configuration Wizard icon

3. Select “Database” and click “Next”

4. Select “Dataset” and click “Next”

5. Under data connection, select the connection string to the database selected in the “Connecting to SQL Server” step

6. Expand the “Connection string that you will save...” and make sure the User ID and Password are correct

7. Click “Next”

8. In the database objects window, select the tables to connect to

9. Click “Finish”

4 BUILDING WINDOWS FORMS

[ADD FORM OBJECT TO SOLUTION]

1. Right-click the Project name (below the Solution name) and select “Add”

2. Under the “Add” menu, select “New Item”

3. In the left-hand panel, select the “Windows Forms” category

4. Under the “Windows Forms” category, select “Windows Form”

5. Click “Add”

6. In the Solution Explorer in the right-hand panel, click on the form added (usually named Form1.cs)

7. The tab titled Form1.cs [Design] opens in the center pane

8. Expand the Data Source tab in the left-hand panel

9. Expand the tab for the Dataset created in the previous “Connecting to Data Source” step

10. Expand the tab for the form to format the newly created Windows Form after

11. In the drop-down arrow next to the Table title, select “Details”

12. For date fields or list fields, select the drop-down arrow to change the button to the appropriate format

13. Click on the Table title and drag over to the center pane

14. The Form1.cs window should now be populated with buttons for the selected Table

15. Re-arrange the buttons by selecting and dragging to different areas of the Windows Form

16. Double-click on the Form1.cs object item in the Solution Explorer right-hand panel

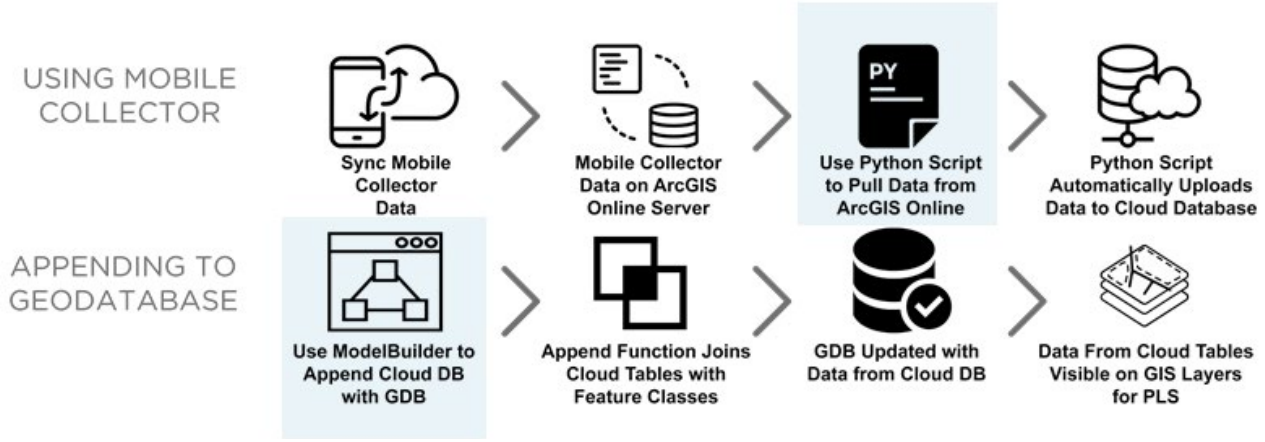
17. In the Properties window below the Solution Explorer, change the “Text” field where Form1 is listed to the name of the form for data entry purposes (i.e. Pedestrian Volume Data Entry Form)

{Forms for Pedestrian Volume, Bicycle Volume, Activity Scan, and User Intercept Surveys are completed and can be found under:

C:Users\sambati\Documents\Visual Studio2017\Projects\WindowsFormsApp2 }

[PHOTO CREDIT: COLUMBUS PUBLIC LIFE STUDY]

NEXT STEPS



The next steps in implementing the new workflow include:

1 PUBLISHING ONLINE FORMS

Using Adobe Dreamweaver to build and publish an online website hosting online forms for data entry.

Using PHP, the online forms can automatically import data entered on the online forms to corresponding data tables on the SQL Server.

2 CONNECTING SQL SERVER TO ARCONLINE

Writing a Python script to push information that is collected through ArcGIS softwares Survey123 and Mobile Collector from the online storage in ArcOnline to the SQL Server.

Steps were taken to receive a script to automate this process - follow up with contacts at ESRI to establish procedure of pushing data from ArcOnline to online server.

3 APPENDING SQL SERVER DATA TO GEODATABASE

Creating a new ModelBuilder to append data stored in the SQL Server to the geodatabase containing feature classes for public life study data.

The current ModelBuilder is located in the PLS-Geocode folder in ArcCatalog under the PublicLifeTools.tbx tools. All ModelBuilder data collection types need to be reformatted for use with the new server.

4 UPDATING DATA ARCHITECTURE

Create a prioritization roadmap for reorganizing the PLS database (see Appendix D for proposed architecture changes).

Create linked tables for easy querying methods.

5 SETTING UP POSTGRESQL

Download and set up PostgreSQL and use the included pgadmin4 to set up geospatial data tables to link to the SQL server.

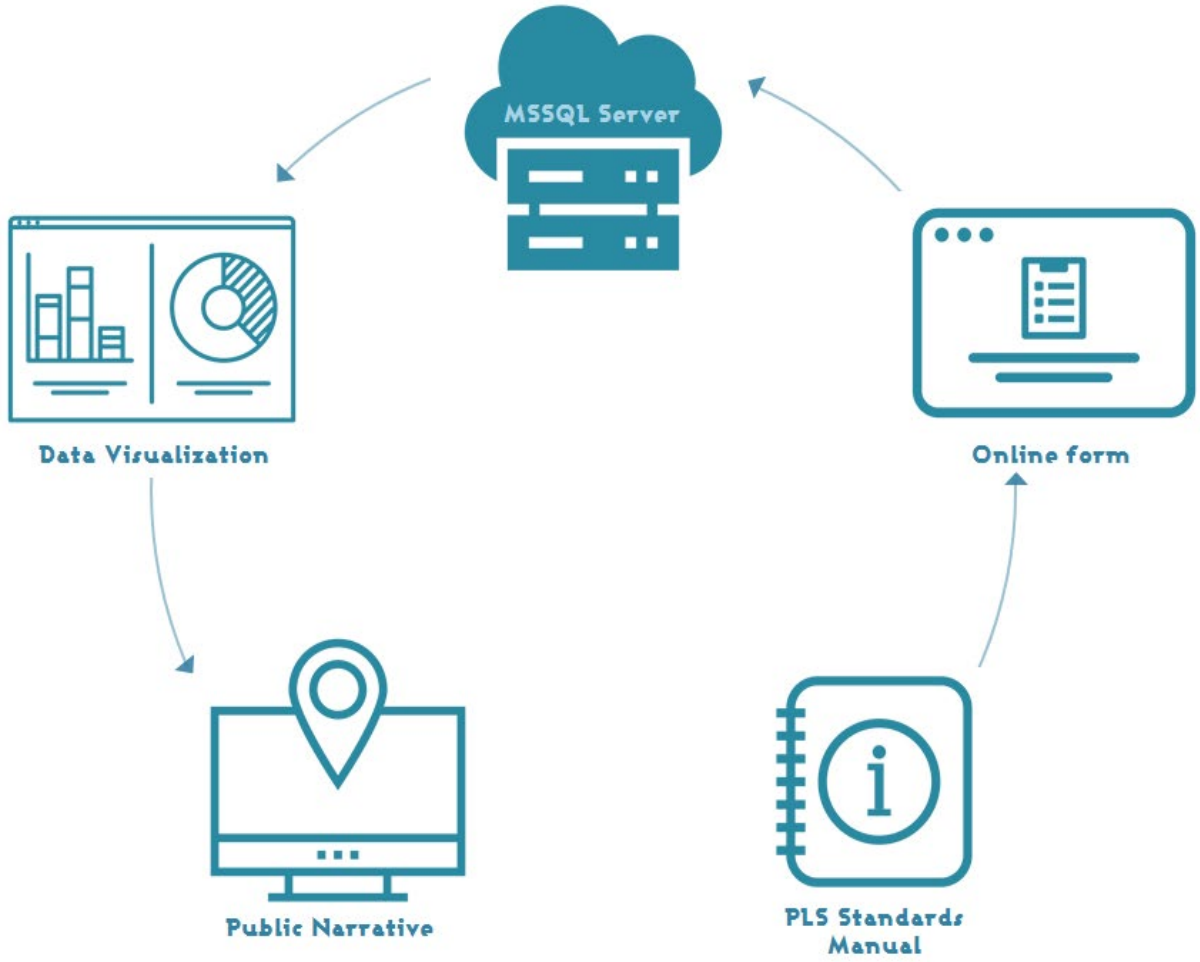
6 LINK POSTGIS DATA TO EXISTING SERVER

Download the ODBC driver for PSQL and use SQL Server Import and Export Wizard to set up the data connection to PostgreSQL data.

A PART OF THE BIGGER PICTURE

The new PLS SQL Server database allows planners to form consistency in data collection methods and data storage. The SQL Server also influences the workflow of creating and analyzing PLS data by allowing planners to connect data with other relevant datasets and data visualization services with ease. The following diagram illustrates how the use of the new SQL Server interacts with other aspects of the Public Life program, such as the Public Life Standards Manual and the public narrative:

FIGURE 5: PLS DATABASE BIGGER PICTURE INTERACTIONS



APPENDIX A:
LOGINS AND PASSWORDS

APPENDIX B:
LOGGING INTO REMOTE DESKTOP

1. Click on the Start icon on your Desktop PC
2. Click on “All Programs”
3. Click on “Accessories”
4. Select “Remote Desktop Connection”
5. Add the IP Address as the Computer (52.203.113.168)
6. Enter Username and Password
7. If Warnings Pop-up, click “Continue”

APPENDIX C:
SAMPLE SQL QUERIES

Querying for a study area:

Select* from <table name> where studyarea = '<study area name>';

Querying for volume at a certain hour:

Select studyarea, SUM(PedVol.total_ped or BikeVol.totalcyclistcount) as totalforhour from <table name> where hourblock = '<1-23>' group by studyarea;

Sorting study area by highest volume by day:

Select studyarea, surveydate, SUM(PedVol.total_ped or BikeVol.totalcyclistcount) as thecount from <table name> group by studyarea order by surveydate, thecount DESC;

Sorting study area results by top records for day or hour:

Select studyarea, COUNT(PedVol.totalped or BikeVol.totalcyclistcount) as thecount from <table name> group by studyarea, <hourblock or dayofweek> order by thecount DESC;

Returning the average volumes by study area and day of week:

Select studyarea, AVG(totalped or total) as avgcount, dayofweek from <table name> group by studyarea

APPENDIX D:
PROPOSED DATA ARCHITECTURE

STUDY CONTEXT TABLES

STUDY
ID (primary key)
study_id
study_area
study_contact
version

LOCATION
ID (primary key)
spatial_id
study_id
unit
unitside
unitaddress

CONTEXT
ID (primary key)
study_id
surveydate
day_type
weather
temperature
starttime
endtime
hourblock
duration
collector
enterdate

DATAMETHOD
ID (primary key)
study_id
study_instrument

COUNT TABLES

PEDVOL
ID (primary key)
count_id
total_ped
lr_ped
rl_ped
lr_male
rl_male
lr_female
rl_female
age6
age7
age16
age30
age31
age64
age65
arunn
aplay
aplyx
aneed
obstroll
obcart
notes

BIKEVOL
ID (primary key)
count_id
total_bike
lr_bike
rl_bike
lr_male
rl_male
lr_female
rl_female
age6
age7
age16
age30
age31
age64
age65
bcntr
bsdwk
bnoht
notes

ACTSCAN
ID (primary key)
count_id
total_ped
male
female
age10
age15
age16
age30
age31
age64
age6
pair
group
pstnd
pphys
psitf
psitc
psitp
psitm
psitw
psits
psiti
psitg
plean
plyng
acult
acome
acomi
arunn
aplay
aplyi
aplyx
aeatd
aelec
atalk
awtch
aidle
atrns
axwlk
awpet
nsmok
nintx
nslep
npanh
obpee
oblit
obbag
obstroll
obcart
obpet
notes

USERINT
ID (primary key)
count_id
trvl_mode
trvl_y
trvl_time
frequency
visit_len
visit_y
here_y
here_hood
fave_in
fave_in_y
fave_out
fave_out_y
res_city
res_zip
res_x
res_year
orig_x
des_x
spend
o_noise
r_noise
y_noise
o_clean
r_clean
o_clean
r_clean
y_clean
o_cond
r_cond
y_cond
o_car
r_car
y_car
o_person
r_person
y_person
o_talk
r_talk
y_talk
o_opps
r_opps
y_opps
o_opublic
r_opublic
y_opublic
o_oprivate
r_oprivate
y_oprivate
o_attract
r_attract
y_attract
o_walk
r_walk
y_walk
o_shop
r_shop
y_shop
o_weather
r_weather
y_weather
o_over
r_over
y_over
accomp16
accomp65
accompdis
accompfam
yearborn
gender
ethnic
race
notes

ACTSCAN_BV
ID (primary key)
count_id
brack
bempt
bothr
bcorr
bcntr
bsdwk
bnoht
bplet
bplem
vmotor
vcars
vvans
vtruc
vload
vdprk
vempt
viprk

Using existing fields in the SQL server database, the chart to the left displays new architecture for the database in order to have higher performance.