Australia, Land of Toilets: The Final Write-Up
By Pete Wells, Stuart Rimel
6/2/2022, CS-586: Intro to Database Management Systems

DESCRIPTION OF ETL

Our dataset includes over 22k public toilets throughout Australia, with each toilet containing 47 data points regarding all the attributes of the facilities (see TABLE CONTENT section). We have split up this data into 14 relational tables (see below CREATE TABLE STATEMENTS section), wich allows multiple perspectives on toilet data throughout the land down under!

Our ETL application will load all 22k+ toilets split among 14 relational tables in about 45 minutes on the school's linux servers. The ETL application starts off by prompting for database credentials to login to our DB. Then, the it creates a connection by using python's psycopg2 library. This allows the application to create instances of our database such that we can run sql commands via python functions. The ETL then proceeds to create all the relational tables that will be utilized in our database (see SCHEMA DIAGRAM attachment). After each table is created (fully automated) a confirmation message is output. Once all the tables are created then the ETL invokes the data wrangling from our source csv file into dataframes using python's pandas library. This process allows our ETL to automate the organization of the source dataset into 14 dataframes that correspond to our 14 relational tables that were previously created by the application. Once the ETL wrangles the dataset into 14 dataframes, then it will insert each row of each dataframe into it's respective relational table in sql via the to_sql() function from the pandas library. With that, the entirety of the dataset is wrangled and inserted into newly created tables within our database.

Overall, the entire process of creating this database and inserting all data into was successfully automated with this ETL application. If we had more time, we would potentially re-organize how some of our data is normalized to prevent excessive joins and try to optimize the amount of time it takes to load the data into the database. There is definitely room for efficiency improvements!

Link to our ETL application repo -->
https://github.com/srimel/land_of_toilets

CREATE TABLE STATEMENTS

We created 14 relational tables for our dataset. See the following create table statements that we used in our ETL:

```
# Create toilets table
def create_toilets_table(db_conn):
    cur = db_conn.cursor()
    create_stmt = "CREATE TABLE toilets(" \
                " FacilityID INT," \
                " URL VARCHAR(256)," \
                " Name VARCHAR(128)," \
                " Male BOOL,"\
                " Female BOOL,"\
                " Unisex BOOL,"\
                " AllGender BOOL,"\
                " ToiletNote VARCHAR(1024),"\
                " DrinkingWater BOOL,"\
                " Shower BOOL,"\
                " PRIMARY KEY (FacilityID));"
    cur.execute(create_stmt)


# Create handicap table
def create_handicap_table(db_conn):
    cur = db_conn.cursor()
    create_stmt = "CREATE TABLE handicap(" \
                " FacilityID INT," \
                " BYOSling BOOL,"\
                " Ambulant BOOL,"\
                " LHTransfer BOOL,"\
                " RHTransfer BOOL,"\
                " PRIMARY KEY (FacilityID),"\
                " CONSTRAINT handicap_fk FOREIGN KEY(FacilityID) "\
                " REFERENCES toilets(FacilityID));"
    cur.execute(create_stmt)


# Create changing table
def create_changing_table(db_conn):
    cur = db_conn.cursor()
    create_stmt = "CREATE TABLE changing(" \
                " FacilityID INT," \
                " BabyChange BOOL,"\
                " BabyCareRoom BOOL,"\
```

```
                " BabyChangeNote VARCHAR(400),"\
                " ACShower BOOL,"\
                " AdultChange BOOL,"\
                " AdultChangeNote VARCHAR(400),"\
                " ChangingPlaces BOOL,"\
                " PRIMARY KEY (FacilityID),"\
                " CONSTRAINT changing_fk FOREIGN KEY(FacilityID) "\
                " REFERENCES toilets(FacilityID));"
    cur.execute(create_stmt)


# Create access table
def create_access_table(db_conn):
    cur = db_conn.cursor()
    create_stmt = "CREATE TABLE access(" \
                " FacilityID INT," \
                " KeyRequired BOOL,"\
                " AccessNote VARCHAR(400),"\
                " PaymentRequired BOOL,"\
                " MLAK24 BOOL,"\
                " MLAKAfterHours BOOL,"\
                " OpeningHours VARCHAR(256),"\
                " OpeningHoursNote VARCHAR(400),"\
                " Accessible BOOL,"\
                " Parking BOOL,"\
                " ParkingAccessible BOOL,"\
                " ParkingNote VARCHAR(400),"\
                " PRIMARY KEY (FacilityID),"\
                " CONSTRAINT access_fk FOREIGN KEY(FacilityID) "\
                " REFERENCES toilets(FacilityID));"
    cur.execute(create_stmt)


# Create disposal table
def create_disposal_table(db_conn):
    cur = db_conn.cursor()
    create_stmt = "CREATE TABLE disposal(" \
                " FacilityID INT," \
                " SharpsDisposal BOOL,"\
                " SanitaryDisposal BOOL,"\
                " MensPadDisposal BOOL,"\
                " PRIMARY KEY (FacilityID),"\
                " CONSTRAINT disposal_fk FOREIGN KEY(FacilityID) "\
                " REFERENCES toilets(FacilityID));"
    cur.execute(create_stmt)
```

```
# Create dump_points table
def create_dump_points_table(db_conn):
    cur = db_conn.cursor()
    create_stmt = "CREATE TABLE dump_points(" \
                " FacilityID INT," \
                " DPWashout BOOL,"\
                " DPAfterHours BOOL,"\
                " DumpPointNote VARCHAR(400),"\
                " PRIMARY KEY (FacilityID),"\
                " CONSTRAINT dump_points_fk FOREIGN KEY(FacilityID) "\
                " REFERENCES toilets(FacilityID));"
    cur.execute(create_stmt)


# Create facility_types table
def create_facility_types_table(db_conn):
    cur = db_conn.cursor()
    create_stmt = "CREATE TABLE facility_types(" \
                " TypeID INT," \
                " Name VARCHAR(128),"\
                " PRIMARY KEY (TypeID));"
    cur.execute(create_stmt)


# Create facility_rel table
def create_facility_rel(db_conn):
    cur = db_conn.cursor()
    create_stmt = "CREATE TABLE facility_rel (" \
                  "  FacilityID  INT," \
                  "  TypeID   INT," \
                  "  PRIMARY KEY (FacilityID, TypeID)," \
                  "  CONSTRAINT facility_rel_fk FOREIGN KEY(TypeID) " \
                  "  REFERENCES facility_types(TypeID));"
    cur.execute(create_stmt)


# Create locations table
def create_locations(db_conn):
    cur = db_conn.cursor()
    create_stmt = "CREATE TABLE locations (" \
                  "  LocID  INT," \
                  "  Address1   VARCHAR(256)," \
                  "  Latitude   FLOAT," \
                  "  Longitude   FLOAT," \
                  "  AddressNote   VARCHAR(400)," \
```

```
                "  PRIMARY KEY (LocID));"
    cur.execute(create_stmt)


# Create location_rel table
def create_location_rel(db_conn):
    cur = db_conn.cursor()
    create_stmt = "CREATE TABLE location_rel (" \
                "  FacilityID  INT," \
                "  LocID  INT," \
                "  PRIMARY KEY (FacilityID, LocID)," \
                "  CONSTRAINT fk_loc_id FOREIGN KEY(LocID) " \
                "  REFERENCES locations(LocID));"
    cur.execute(create_stmt)


# Create states table
def create_states(db_conn):
    cur = db_conn.cursor()
    create_stmt = "CREATE TABLE states (" \
                "  StateID  INT," \
                "  State  VARCHAR(16)," \
                "  PRIMARY KEY (StateID));"
    cur.execute(create_stmt)


# Create state_rel table
def create_state_rel(db_conn):
    cur = db_conn.cursor()
    create_stmt = "CREATE TABLE state_rel (" \
                "  LocID  INT," \
                "  StateID  INT," \
                "  PRIMARY KEY (LocID, StateID)," \
                "  CONSTRAINT fk_loc_id FOREIGN KEY(LocID) " \
                "  REFERENCES locations(LocID));"
    cur.execute(create_stmt)


# Create towns table
def create_towns(db_conn):
    cur = db_conn.cursor()
    create_stmt = "CREATE TABLE towns (" \
                "  TownID  INT," \
                "  Town  VARCHAR(128)," \
                "  PRIMARY KEY (TownID));"
    cur.execute(create_stmt)
```

```
# Create town_rel table
def create_town_rel(db_conn):
    cur = db_conn.cursor()
    create_stmt = "CREATE TABLE town_rel (" \
                  "  LocID  INT," \
                  "  TownID  INT," \
                  "  PRIMARY KEY (LocID, TownID)," \
                  "  CONSTRAINT fk_loc_id FOREIGN KEY(LocID) " \
                  "  REFERENCES locations(LocID));"
    cur.execute(create_stmt)
```

QUERIES AND RESULTS

```
--1. How many toilets in Cooma have parking?
SELECT COUNT(*) AS "# Toilets in Cooma with Parking"
FROM location_rel JOIN locations USING(locID) JOIN town_rel USING(LocID)
JOIN towns USING(TownID) JOIN access USING(facilityID)
WHERE Town='Cooma' AND parking=True;

 # Toilets in Cooma with Parking
--------------------------------
                               4
(1 row)


--2. How many public toilets are there in Australia?
SELECT COUNT(*) AS "# Toilets Down Under" FROM toilets;

 # Toilets Down Under
----------------------
                22031
(1 row)

--3. What are all the sporting facility toilets that are also dump points?
// REWORD: How many sporting facility toilets are also dump points?
SELECT COUNT(*) AS "# Sporting Facilities That Double As Dumps"
FROM facility_rel JOIN facility_types FT USING(typeid)
JOIN dump_points USING (facilityID)
WHERE FT.name='Sporting facility';
```

```
 # Sporting Facilities That Double As Dumps
-------------------------------------------
                                       1872
(1 row)

--4. Which city has the most public toilets?
SELECT town, COUNT(facilityID) AS "Toilet Count" FROM
location_rel JOIN locations USING(locID) JOIN town_rel USING(locID)
JOIN towns T USING(townID)
GROUP BY town ORDER BY town;

               town                | Toilet Count
-----------------------------------+--------------
 Melbourne                         |           50
 Sydney                            |           49
 Adelaide                          |           45
 Dubbo                             |           45
 Coffs Harbour                     |           40
 Hamilton                          |           36
 Manly                             |           35
 Muswellbrook                      |           34
 Orange                            |           32
 Mildura                           |           31
 ...
(6326 rows)
For whole list see q4.txt

--5. What percentage of toilets are open 24 hours a day?
SELECT 100*(SELECT COUNT(*) FROM toilets JOIN access USING(facilityID) WHERE
openinghours='OPEN: 24 hours')/COUNT(*)
AS "% Austrailian Toilets Open 24 Hours a Day" FROM toilets;

 % Austrailian Toilets Open 24 Hours a Day
-------------------------------------------
                                        46
(1 row)

--6. How many toilets in WA require payment?
SELECT COUNT(*) AS "# Toilets in WA That Require Money" FROM access JOIN
location_rel USING(facilityID) JOIN locations USING(locID)
JOIN state_rel USING(locID) JOIN states USING(stateID)
WHERE state='WA' AND paymentrequired=TRUE;

 # Toilets in WA That Require Money
-----------------------------------
```

```
(1 row)

--7. How many unisex toilets are in a park or reserve?
SELECT count(*) AS "# Unisex Toilets in Parks and Reserves" FROM toilets
JOIN facility_rel USING(FacilityID) JOIN facility_types T USING(typeID)
WHERE T.name='Park or reserve' AND unisex=True;

 # Unisex Toilets in Parks and Reserves
---------------------------------------
                                  2077
(1 row)

--8. How many toilets are on 1 Bay Street in Glebe?
SELECT COUNT(*) AS "# Toilets on 1 Bay Street in Glebe" FROM
location_rel JOIN locations USING(locID) JOIN town_rel USING(locID) JOIN
towns USING(townid)
WHERE town='Glebe' AND address1='1 Bay Street';

 # Toilets on 1 Bay Street in Glebe
-----------------------------------
                                  4
(1 row)

--9. What percentage of public toilets are free?
SELECT 100*(SELECT COUNT(*) FROM access WHERE
paymentrequired=False)/COUNT(*) AS  "Percentage of the Free" FROM toilets;

 Percentage of the Free
------------------------
                     99
(1 row)

--10. What percentage of public toilets with baby changing stations are
free?
SELECT 100*(SELECT COUNT(*) FROM access JOIN changing USING(facilityID)
WHERE paymentrequired=False and babychange=True)/COUNT(*) AS  "Percentage of
the Free... With Babies"
FROM toilets JOIN changing USING(facilityID) WHERE babychange=True;

 Percentage of the Free... With Babies
---------------------------------------
                                    99
(1 row)

--11. What percentage of men’s toilets contain baby changing stations?
SELECT 100*(SELECT COUNT(*) FROM toilets JOIN changing USING(facilityID)
```

```
WHERE male=True AND babychange=True)/COUNT(*) AS "% Men's Restrooms with
Baby Changing Stations"
FROM toilets JOIN changing USING (facilityID) WHERE male=True AND
babychange=False;

 % Men's Restrooms with Baby Changing Stations
-----------------------------------------------
                                              15
(1 row)

--11.5 What percentage of women's toilets contain baby changing stations?
SELECT 100*(SELECT COUNT(*) FROM toilets JOIN changing USING(facilityID)
WHERE female=True AND babychange=True)/COUNT(*) AS "% Women's Restrooms with
Baby Changing Stations"
FROM toilets JOIN changing USING (facilityID) WHERE female=True AND
babychange=False;

 % Women's Restrooms with Baby Changing Stations
-------------------------------------------------
                                                15
(1 row)

--11.75 What percentage of unisex toilets contain baby changing stations?
SELECT 100*(SELECT COUNT(*) FROM toilets JOIN changing USING(facilityID)
WHERE unisex=True AND babychange=True)/COUNT(*) AS "% Unisex Restrooms with
Baby Changing Stations"
FROM toilets JOIN changing USING (facilityID) WHERE unisex=True AND
babychange=False;

 % Unisex Restrooms with Baby Changing Stations
-------------------------------------------------
                                                26
(1 row)

--12. How many carpark dump points are there?
SELECT COUNT(*) as "Number of Carpark dumpoints"
FROM toilets t join dump_points using(facilityid) join facility_rel
using(facilityid) join facility_types ft using(typeid)
WHERE ft.name = 'Car park';

 Number of Carpark dumpoints
-----------------------------
                         664
(1 row)

--13. What is the average number of toilets per city within each state?
```

```
SELECT state, AVG(c.count) as "Average Number of toilets per city"
FROM (
        SELECT town, COUNT(facilityid)
        FROM toilets natural join location_rel natural join town_rel natural
join towns
        GROUP BY (town)
     ) as c join towns using(town) join town_rel using(townid) join
location_rel using(locid) join state_rel using(locid) join states
using(stateid)
GROUP BY(state);

 state | Average Number of toilets per city
-------+------------------------------------
 VIC   |                 9.0436767917411157
 SA    |                 6.7060409924487594
 ACT   |                 7.1206030150753769
 QLD   |                 6.7565006075334143
 WA    |                 6.8379866299646087
 NSW   |                 9.4648702031602709
 TAS   |                 6.2079207920792079
 NT    |                 5.4755244755244755
(8 rows)

--14. Which public toilets with showers also have a fee in New South Wales?
SELECT facilityid, name, state
FROM toilets natural join location_rel natural join state_rel natural join
states natural join access
WHERE paymentrequired = true AND state = 'NSW';

 facilityid |                 name                 | state
------------+--------------------------------------+-------
       3188 | Tourist & Travellers Centre          | NSW
      10963 | Hyde Park - North 1                  | NSW
      33491 | Deniliquin Dump Point                | NSW
      33559 | Tenterfield Showground Dump Point    | NSW
      33567 | Wagga Wagga Showgrounds              | NSW
      50335 | St Ives Shopping Village 1           | NSW
      55813 | Stocko                               | NSW
      56023 | Packsaddle Roadhouse                 | NSW
      56508 | Glendora Campground                  | NSW
      56810 | Brackens Hut                         | NSW
      57531 | Bombala Caravan Park Dump Point      | NSW
      57538 | Cessnock Showground Dump Point       | NSW
      57567 | Gundagai Cabins & Tourist Park       | NSW
      57583 | Kyogle Showground Dump Point         | NSW
      57586 | Lismore Showgrounds Dump Point       | NSW
```

```
      57597 | Mittagong Caravan Park Dump Point    | NSW
      57645 | Camp Blackman Dump Point            | NSW
      59195 | Clemton Park Village Shopping Centre | NSW
(18 rows)

--15. How many toilets with sharp disposals are in every state?
SELECT state, COUNT(*) as "Toilets with sharp disposal"
FROM toilets natural join disposal natural join state_rel join states
using(stateid)
WHERE  sharpsdisposal = true
GROUP BY(state);

 state | Toilets with sharp disposal
-------+-----------------------------
 VIC   |                     18980684
 SA    |                      6987726
 ACT   |                       750031
 QLD   |                     15494359
 WA    |                      9573260
 NSW   |                     26620447
 TAS   |                      3426021
 NT    |                      1077934
(8 rows)

--16. How many toilets that are accessible have parking?
SELECT COUNT(*) as "Number of accessible toilets with parking"
FROM toilets natural join access
WHERE accessible = true AND parking = true;

 Number of accessible toilets with parking
-------------------------------------------
                                       7673
(1 row)

--17. Which toilets in VIC are ambulant?
SELECT facilityid, name, state
FROM toilets natural join location_rel natural join state_rel natural join
states natural join handicap
WHERE ambulant = true AND state = 'VIC';

 facilityid |   name                  | state
------------+---------------------------------
        253 | Lloyd Street            | VIC
        270 | Great Western           | VIC
        548 | Mortlake Tea Tree Lake  | VIC
        557 | Woorndoo                | VIC
```

```
        562 | Dudley W Cornell Park   | VIC
        563 | May Park                | VIC
        572 | Horsham Town Hall       | VIC
       1087 | Central Park            | VIC
       1239 | Centenary Park          | VIC
       1338 | Princes Park 1          | VIC
       ...
(210 rows)

--18. How many restrooms have an accessible toilet?
SELECT COUNT(*) as "Numbe of toilets that are accessible"
FROM toilets natural join access
WHERE accessible = true;

 Numbe of toilets that are accessible
--------------------------------------
                                11779
(1 row)

--19. How many toilets have drinking water and showers?
SELECT COUNT(*) as "Number of toilets with water fountain and showers"
FROM toilets
WHERE drinkingwater = true AND shower = true;

 Number of toilets with water fountain and showers
---------------------------------------------------
                                                400
(1 row)

--20. How many toilets require the master locksmith’s access key (MLAK) to
enter?
 SELECT COUNT(mlak24) as "Toilets requiring master locksmith's access key"
 FROM toilets join access using(facilityid)
 WHERE mlak24 = false;

 Toilets requiring master locksmith's access key
-------------------------------------------------
                                           21639
(1 row)



TABLE CONTENTS
------------------------------------------------------------------------------
-------------------------------------------------------------
```

```
                  List of relations
    Schema    |       Name         | Type  |    Owner
--------------+--------------------+-------+--------------
 spr2022bdb58 | access             | table | spr2022bdb58
 spr2022bdb58 | changing           | table | spr2022bdb58
 spr2022bdb58 | disposal           | table | spr2022bdb58
 spr2022bdb58 | dump_points        | table | spr2022bdb58
 spr2022bdb58 | facility_rel       | table | spr2022bdb58
 spr2022bdb58 | facility_types     | table | spr2022bdb58
 spr2022bdb58 | handicap           | table | spr2022bdb58
 spr2022bdb58 | location_rel       | table | spr2022bdb58
 spr2022bdb58 | locations          | table | spr2022bdb58
 spr2022bdb58 | state_rel          | table | spr2022bdb58
 spr2022bdb58 | states             | table | spr2022bdb58
 spr2022bdb58 | toilets            | table | spr2022bdb58
 spr2022bdb58 | town_rel           | table | spr2022bdb58
 spr2022bdb58 | towns              | table | spr2022bdb58


                      Table "spr2022bdb58.access"
      Column       |           Type           | Collation | Nullable | Default

-------------------+--------------------------+-----------+----------+--------
-
 facilityid        | integer                  |           | not null |
 keyrequired       | boolean                  |           |          |
 accessnote        | character varying(400)   |           |          |
 paymentrequired   | boolean                  |           |          |
 mlak24            | boolean                  |           |          |
 mlakafterhours    | boolean                  |           |          |
 openinghours      | character varying(256)   |           |          |
 openinghoursnote  | character varying(400)   |           |          |
 accessible        | boolean                  |           |          |
 parking           | boolean                  |           |          |
 parkingaccessible | boolean                  |           |          |
 parkingnote       | character varying(400)   |           |          |
Indexes:
    "access_pkey" PRIMARY KEY, btree (facilityid)
Foreign-key constraints:
    "access_fk" FOREIGN KEY (facilityid) REFERENCES toilets(facilityid)


                     Table "spr2022bdb58.changing"
     Column      |           Type           | Collation | Nullable | Default
-----------------+--------------------------+-----------+----------+---------
 facilityid      | integer                  |           | not null |
```

```
 babychange      | boolean                |           |          |
 babycareroom    | boolean                |           |          |
 babychangenote  | character varying(400) |           |          |
 acshower        | boolean                |           |          |
 adultchange     | boolean                |           |          |
 adultchangenote | character varying(400) |           |          |
 changingplaces  | boolean                |           |          |
Indexes:
    "changing_pkey" PRIMARY KEY, btree (facilityid)
Foreign-key constraints:
    "changing_fk" FOREIGN KEY (facilityid) REFERENCES toilets(facilityid)


                Table "spr2022bdb58.disposal"
      Column      |  Type   | Collation | Nullable | Default
------------------+---------+-----------+----------+---------
 facilityid       | integer |           | not null |
 sharpsdisposal   | boolean |           |          |
 sanitarydisposal | boolean |           |          |
 menspaddisposal  | boolean |           |          |
Indexes:
    "disposal_pkey" PRIMARY KEY, btree (facilityid)
Foreign-key constraints:
    "disposal_fk" FOREIGN KEY (facilityid) REFERENCES toilets(facilityid)



                   Table "spr2022bdb58.dump_points"
    Column     |          Type          | Collation | Nullable | Default
---------------+------------------------+-----------+----------+---------
 facilityid    | integer                |           | not null |
 dpwashout     | boolean                |           |          |
 dpafterhours  | boolean                |           |          |
 dumppointnote | character varying(400) |           |          |
Indexes:
    "dump_points_pkey" PRIMARY KEY, btree (facilityid)
Foreign-key constraints:
    "dump_points_fk" FOREIGN KEY (facilityid) REFERENCES toilets(facilityid)


           Table "spr2022bdb58.facility_rel"
   Column   |  Type   | Collation | Nullable | Default
------------+---------+-----------+----------+---------
 facilityid | integer |           | not null |
 typeid     | integer |           | not null |
Indexes:
```

```
    "facility_rel_pkey" PRIMARY KEY, btree (facilityid, typeid)
Foreign-key constraints:
    "facility_rel_fk" FOREIGN KEY (typeid) REFERENCES facility_types(typeid)


               Table "spr2022bdb58.facility_types"
 Column |          Type          | Collation | Nullable | Default
--------+------------------------+-----------+----------+---------
 typeid | integer                |           | not null |
 name   | character varying(128) |           |          |
Indexes:
    "facility_types_pkey" PRIMARY KEY, btree (typeid)
Referenced by:
    TABLE "facility_rel" CONSTRAINT "facility_rel_fk" FOREIGN KEY (typeid)
REFERENCES facility_types(typeid)


             Table "spr2022bdb58.handicap"
   Column   |  Type   | Collation | Nullable | Default
------------+---------+-----------+----------+---------
 facilityid | integer |           | not null |
 byosling   | boolean |           |          |
 ambulant   | boolean |           |          |
 lhtransfer | boolean |           |          |
 rhtransfer | boolean |           |          |
Indexes:
    "handicap_pkey" PRIMARY KEY, btree (facilityid)
Foreign-key constraints:
    "handicap_fk" FOREIGN KEY (facilityid) REFERENCES toilets(facilityid)


           Table "spr2022bdb58.location_rel"
   Column   |  Type   | Collation | Nullable | Default
------------+---------+-----------+----------+---------
 facilityid | integer |           | not null |
 locid      | integer |           | not null |
Indexes:
    "location_rel_pkey" PRIMARY KEY, btree (facilityid, locid)
Foreign-key constraints:
    "fk_loc_id" FOREIGN KEY (locid) REFERENCES locations(locid)


                   Table "spr2022bdb58.locations"
   Column    |          Type          | Collation | Nullable | Default
-------------+------------------------+-----------+----------+---------
 locid       | integer                |           | not null |
```

```
 address1    | character varying(256) |           |          |
 latitude    | double precision       |           |          |
 longitude   | double precision       |           |          |
 addressnote | character varying(400) |           |          |
Indexes:
    "locations_pkey" PRIMARY KEY, btree (locid)
Referenced by:
    TABLE "location_rel" CONSTRAINT "fk_loc_id" FOREIGN KEY (locid)
REFERENCES locations(locid)
    TABLE "state_rel" CONSTRAINT "fk_loc_id" FOREIGN KEY (locid) REFERENCES
locations(locid)
    TABLE "town_rel" CONSTRAINT "fk_loc_id" FOREIGN KEY (locid) REFERENCES
locations(locid)


           Table "spr2022bdb58.state_rel"
 Column  |  Type   | Collation | Nullable | Default
---------+---------+-----------+----------+---------
 locid   | integer |           | not null |
 stateid | integer |           | not null |
Indexes:
    "state_rel_pkey" PRIMARY KEY, btree (locid, stateid)
Foreign-key constraints:
    "fk_loc_id" FOREIGN KEY (locid) REFERENCES locations(locid)


                  Table "spr2022bdb58.states"
 Column  |         Type          | Collation | Nullable | Default
---------+-----------------------+-----------+----------+---------
 stateid | integer               |           | not null |
 state   | character varying(16) |           |          |
Indexes:
    "states_pkey" PRIMARY KEY, btree (stateid)


                     Table "spr2022bdb58.toilets"
    Column     |          Type           | Collation | Nullable | Default
---------------+-------------------------+-----------+----------+---------
 facilityid    | integer                 |           | not null |
 url           | character varying(256)  |           |          |
 name          | character varying(128)  |           |          |
 male          | boolean                 |           |          |
 female        | boolean                 |           |          |
 unisex        | boolean                 |           |          |
 allgender     | boolean                 |           |          |
 toiletnote    | character varying(1024) |           |          |
```

```
 drinkingwater | boolean                |           |          |
 shower        | boolean                |           |          |
Indexes:
    "toilets_pkey" PRIMARY KEY, btree (facilityid)
Referenced by:
    TABLE "access" CONSTRAINT "access_fk" FOREIGN KEY (facilityid)
REFERENCES toilets(facilityid)
    TABLE "changing" CONSTRAINT "changing_fk" FOREIGN KEY (facilityid)
REFERENCES toilets(facilityid)
    TABLE "disposal" CONSTRAINT "disposal_fk" FOREIGN KEY (facilityid)
REFERENCES toilets(facilityid)
    TABLE "dump_points" CONSTRAINT "dump_points_fk" FOREIGN KEY (facilityid)
REFERENCES toilets(facilityid)
    TABLE "handicap" CONSTRAINT "handicap_fk" FOREIGN KEY (facilityid)
REFERENCES toilets(facilityid)


           Table "spr2022bdb58.town_rel"
 Column |  Type   | Collation | Nullable | Default
--------+---------+-----------+----------+---------
 locid  | integer |           | not null |
 townid | integer |           | not null |
Indexes:
    "town_rel_pkey" PRIMARY KEY, btree (locid, townid)
Foreign-key constraints:
    "fk_loc_id" FOREIGN KEY (locid) REFERENCES locations(locid)


                    Table "spr2022bdb58.towns"
 Column |          Type          | Collation | Nullable | Default
--------+------------------------+-----------+----------+---------
 townid | integer                |           | not null |
 town   | character varying(128) |           |          |
Indexes:
    "towns_pkey" PRIMARY KEY, btree (townid)
```