```
--################################
--Notices
--################################

-- Copyleft Ed Egan, 2022
-- Note(s):
        -- PortCoPoints, PortcoMaster, and Round are pre-loaded from VCDB20.
        See https://www.edegan.com/wiki/VCDB20.
        -- CPI table is pre-loaded (source: https://www.bls.gov/data/)

--################################
--Enable extensions
--################################

create extension postgis;

--####################################
--CBSA
--####################################

DROP TABLE TigerCBSA;
CREATE TABLE TigerCBSA
(
        statecode varchar(100),
        statename varchar(100),
        gid integer,
        statefp varchar(2),
        placefp varchar(5),
        placens varchar(8),
        geoid varchar(7),
        name varchar(100),
        namelsad varchar(100),
        lsad varchar(2),
        classfp varchar(2),
        pcicbsa varchar(1),
        pcinecta varchar(1),
        mtfcc varchar(5),
        funcstat varchar(1),
        aland numeric,
        awater numeric,
        intptlat varchar(11),
        intptlon varchar(12),
        geom geometry(MultiPolygon,4326)
);

\COPY TigerCBSA FROM 'TigerCBSA.txt' WITH DELIMITER AS E'\t' HEADER NULL AS ''
CSV

DROP TABLE IF EXISTS TigerCBSAgeom;
CREATE TABLE TigerCBSAgeom AS
SELECT name as CBSA, geoid,
CASE WHEN lsad = 'M1' THEN 1 ELSE 0 END as Metro,
ST_SetSRID(wkb_geometry,4326) as geom
FROM TigerCBSA;

CREATE INDEX cbsa_geom_idx on TigerCBSAgeom USING GIST (geom);

--########################
--#Make PortCoPointCBSA
--########################

DROP TABLE IF EXISTS PortCoPointCBSA;
CREATE TABLE PortCoPointCBSA AS
SELECT cbsa, coname, statecode, datefirstinv,
lat4, long4,
geoid as cbsageoid,
pcpoint
FROM PortCoPoints, TigerCBSAgeom
WHERE addr1 IS NOT NULL AND ST_Intersects(geom, PortCoPoints.pcpoint);

DROP TABLE IF EXISTS PortCoHCACBSABlowOut;
CREATE TABLE PortCoHCACBSABlowOut AS
SELECT cbsa, year, lat4, long4, A.coname, A.datefirstinv, A.statecode,
cbsageoid
FROM PortCoPointCBSA AS A
JOIN portcomaster AS B ON A.coname=B.coname AND A.datefirstinv=B.datefirstinv
and A.statecode=B.statecode
JOIN year95to20 ON aliveyear <= year AND (deadyear >= year OR deadyear IS
NULL)
WHERE hadgrowth=1
ORDER BY cbsa, year, coname;

DROP TABLE IF EXISTS PortCoHCACBSASummary;
CREATE TABLE PortCoHCACBSASummary AS
SELECT cbsa, year, COUNT(coname) as numactive
FROM PortCoHCACBSABlowOut
GROUP BY cbsa, year
ORDER BY COUNT(coname) DESC;

DROP TABLE IF EXISTS CBSAWithGT10Active;
CREATE TABLE CBSAWithGT10Active AS
SELECT cbsa, max(numactive) as maxnumactive
FROM PortCoHCACBSASummary WHERE numactive >=10
GROUP BY cbsa
ORDER BY cbsa;

DROP TABLE IF EXISTS PortCoHCACBSA;
CREATE TABLE PortCoHCACBSA AS
SELECT A.cbsa, A.statecode, A.year, A.lat4, A.long4, A.coname, A.datefirstinv,
ST_SetSRID(ST_MakePoint(long4, lat4),4326)::geography as point
FROM PortCoHCACBSABlowOut AS A
JOIN CBSAWithGT10Active AS B ON A.cbsa=B.cbsa;

DROP TABLE IF EXISTS PortCoHCACBSAInput;
CREATE TABLE PortCoHCACBSAInput AS
WITH B AS (
        SELECT cbsa, year, count(*) as obs
        FROM PortCoHCACBSA GROUP BY cbsa, year HAVING COUNT(*) > 1
)
SELECT A.*
FROM PortCoHCACBSA AS A
JOIN B ON A.cbsa=B.cbsa AND A.year=B.year;

\COPY (SELECT cbsa, year, lat4, long4, coname, datefirstinv, statecode FROM
PortCoHCACBSAInput) TO 'CBSAYearForHCA.txt' WITH DELIMITER AS E'\t' HEADER
NULL AS '' CSV

--########################
--#PortCoHCACBSALookup
--########################

DROP INDEX IF EXISTS PortCoHCACBSA_idx;
CREATE INDEX PortCoHCACBSA_idx ON PortCoHCACBSA(cbsa,year,lat4,long4);

DROP TABLE IF EXISTS PortCoHCACBSADist;
CREATE TABLE PortCoHCACBSADist AS
SELECT A.cbsa, A.year, A.lat4 as source_lat, A.long4 as source_long, A.point
as source_point,
B.lat4 as dest_lat, B.long4 as dest_long, B.point as dest_point,
ST_Distance(A.point, B.point)/1000 as dist_km
FROM PortCoHCACBSA AS A
JOIN PortCoHCACBSA AS B ON A.cbsa=B.cbsa AND A.year=B.year;

DROP TABLE IF EXISTS PortCoHCACBSALookup;
CREATE TABLE PortCoHCACBSALookup AS
SELECT source_lat, source_long, dest_lat, dest_long, dist_km
FROM PortCoHCACBSADist
GROUP BY source_lat, source_long, dest_lat, dest_long, dist_km
ORDER BY source_lat, source_long, dest_lat, dest_long;

\COPY PortCoHCACBSALookup TO 'PortCoHCACBSALookup.txt' WITH DELIMITER AS E'\t'
HEADER NULL AS '' CSV


--########################
--#Make HclCBSA
--########################

DROP TABLE IF EXISTS hclCBSA;
CREATE TABLE hclCBSA (
        cbsa varchar(255),
        year int,
        layer  int,
        cluster  int,
        lat  decimal,
        long decimal,
        coname  varchar(255),
        datefirstinv date,
        portcostatecode varchar(2)
);

\COPY hclCBSA FROM 'CBSAYearForHCA_results.txt' WITH DELIMITER AS E'\t' HEADER
NULL AS '' CSV

CREATE INDEX hclCBSA_idx ON hclCBSA(cbsa,year,layer,cluster);

DROP TABLE IF EXISTS HCACBSAInduLookup;
CREATE TABLE HCACBSAInduLookup AS
SELECT A.*, B.code
FROM hclCBSASource AS A
JOIN portcoMaster AS B ON A.coname=B.coname AND A.portcostatecode=B.statecode
AND A.datefirstinv=B.datefirstinv;

CREATE INDEX HCACBSAIndu_idx ON HCACBSAIndu(cbsa,year,layer,cluster);

DROP TABLE IF EXISTS HCACBSAIndu;
CREATE TABLE HCACBSAIndu AS
SELECT A.*,
CASE WHEN (B.code/1000) =1 THEN 1 ELSE 0 END AS it,
CASE WHEN (B.code/1000) =2 THEN 1 ELSE 0 END AS ls,
CASE WHEN (B.code/1000) =3 THEN 1 ELSE 0 END AS ht,
CASE WHEN (B.code/1000) =4 THEN 1 ELSE 0 END AS other
FROM hclCBSASource AS A
JOIN portcoMaster AS B ON A.coname=B.coname AND A.portcostatecode=B.statecode
AND A.datefirstinv=B.datefirstinv;

CREATE INDEX HCACBSAIndu_idx ON HCACBSAIndu(cbsa,year,layer,cluster);

--########################
--#Calculate the cluster variances
--########################
```

```
DROP TABLE IF EXISTS HCLCBSApoints;
CREATE TABLE HCLCBSApoints AS
SELECT cbsa, year, layer, cluster, coname, datefirstinv,
ST_Transform(ST_SetSRID(ST_Makepoint(long,lat),4326),4326) AS pointgeom
FROM HCLCBSA;

DROP INDEX IF EXISTS HCLCBSApoints_idx;
CREATE INDEX HCLCBSApoints_idx ON HCLCBSApoints(cbsa,year,layer,cluster);

DROP TABLE IF EXISTS HCLCBSAclustercentroids;
CREATE TABLE HCLCBSAclustercentroids AS
SELECT cbsa, year, layer, cluster, count(coname) AS littlen,
ST_Centroid(ST_Collect(pointgeom)::geography) as centroidgeog
FROM HCLCBSApoints
GROUP BY cbsa, year, layer, cluster;

DROP TABLE IF EXISTS HCLCBSAoverallcentroids;
CREATE TABLE HCLCBSAoverallcentroids AS
SELECT cbsa, year, layer, max(cluster)+1 AS bigK,
ST_Centroid(ST_Collect(pointgeom)::geography) as overallgeog
FROM HCLCBSApoints
GROUP BY cbsa, year, layer;

DROP TABLE IF EXISTS HCLCBSAbetweenbase;
CREATE TABLE HCLCBSAbetweenbase AS
SELECT  A.cbsa, A.year, A.layer, A.cluster, A.littlen, A.centroidgeog,
B.bigK, B.overallgeog,
ST_Distance(A.centroidgeog,B.overallgeog)/100 as betweendisthm,
(ST_Distance(A.centroidgeog,B.overallgeog)/100)^2 as betweendisthmsq
FROM HCLCBSAclustercentroids AS A
JOIN HCLCBSAoverallcentroids AS B ON A.cbsa=B.cbsa AND A.year=B.year AND
A.layer=B.layer;

DROP TABLE IF EXISTS HCLCBSAbetween;
CREATE TABLE HCLCBSAbetween AS
SELECT cbsa, year, layer, bigk,
CASE WHEN bigk>1 THEN sum(littlen::numeric*(betweendisthm::numeric^2))/
(bigK::numeric-1) ELSE 0::numeric END AS betweenvarhm,
CASE WHEN bigk>1 THEN sum(littlen::numeric*(betweendisthm::numeric^2)) ELSE
0::numeric END AS betweenss
FROM HCLCBSAbetweenbase
GROUP BY cbsa, year, layer, bigk;

DROP TABLE IF EXISTS HCLCBSAwithininnerbase;
CREATE TABLE HCLCBSAwithininnerbase AS
SELECT A.cbsa, A.year, A.layer, A.cluster,
ST_Distance(pointgeom::geography,centroidgeog)/100 as withindisthm
FROM HCLCBSApoints AS A
JOIN HCLCBSAclustercentroids AS B ON A.cbsa=B.cbsa AND A.year=B.year AND
A.layer=B.layer AND A.cluster=B.cluster;

DROP INDEX IF EXISTS HCLCBSAwithininnerbase_idx;
CREATE INDEX HCLCBSAwithininnerbase_idx ON
HCLCBSAwithininnerbase(cbsa,year,layer,cluster);

DROP TABLE IF EXISTS HCLCBSAwithininner;
CREATE TABLE HCLCBSAwithininner AS
SELECT cbsa, year, layer, cluster, count(*) as littlen,
sum(withindisthm^2) AS totwithindisthmsq
FROM HCLCBSAwithininnerbase
GROUP BY cbsa, year, layer, cluster;

DROP TABLE IF EXISTS HCLCBSAwithin;
CREATE TABLE HCLCBSAwithin AS
```

```
SELECT cbsa, year, layer,
CASE WHEN sum(littlen)-(max(cluster)+1) = 0 THEN NULL ELSE
sum(totwithindisthmsq)/(sum(littlen)-(max(cluster)+1)) END as withinvarhmadj,
sum(totwithindisthmsq) AS withinss,
max(cluster)+1 AS bigk,
sum(littlen) as bigN
FROM HCLCBSAwithininner
GROUP BY cbsa, year, layer;

DROP TABLE IF EXISTS  HCLCBSAvariance;
CREATE TABLE HCLCBSAvariance AS
SELECT A.cbsa, A.year, A.layer,
A.betweenvarhm, A.betweenss, A.bigk as betweenbigk,
B.withinvarhmadj, withinss, B.bigk AS withinnbigk, B.bigN,
A.betweenss+B.withinss as totalss,
CASE WHEN B.withinvarhmadj >0 THEN A.betweenvarhm/B.withinvarhmadj ELSE
0::numeric END AS Fstat,
CASE WHEN (A.betweenss+B.withinss) >0 THEN A.betweenss/(A.betweenss
+B.withinss) ELSE 0::numeric END AS r2
FROM HCLCBSAbetween AS A
JOIN HCLCBSAwithin AS B ON A.cbsa=B.cbsa AND A.year=B.year AND A.layer=B.layer
ORDER BY A.cbsa, A.year, A.layer;

--########################
--#Select the Elbow Layer
--########################

DROP TABLE IF EXISTS HCLCBSAVarianceDiffs;
CREATE TABLE HCLCBSAVarianceDiffs AS
SELECT A.cbsa, A.year, A.layer, A.r2 as varex0,
CASE WHEN B.r2 IS NOT NULL THEN B.r2 ELSE 1 END AS varex1,
CASE WHEN C.r2 IS NOT NULL THEN C.r2 ELSE 1 END AS varex2,
CASE WHEN B.r2 IS NOT NULL THEN B.r2-A.r2 ELSE 1-A.r2 END AS firstdiff,
CASE WHEN C.r2 IS NOT NULL AND B.r2 IS NOT NULL THEN C.r2-B.r2
 WHEN C.r2 IS NULL AND B.r2 IS NOT NULL THEN 1-B.r2
 WHEN C.r2 IS NULL AND B.r2 IS NULL THEN 0 END AS nextfirstdiff,
CASE WHEN A.layer=0 THEN 0::int
 WHEN B.r2 IS NOT NULL THEN
        CASE WHEN B.r2 >= A.r2 THEN 1::int ELSE 0::int END
 WHEN B.r2 IS NULL THEN 1::int END AS pass,
CASE WHEN A.layer=0 THEN 0::int
 WHEN C.r2 IS NOT NULL AND B.r2 IS NOT NULL THEN
        CASE WHEN C.r2 >= B.r2 THEN 1::int ELSE 0::int END
 WHEN C.r2 IS NULL AND B.r2 IS NOT NULL THEN 1::int
 WHEN C.r2 IS NULL AND B.r2 IS NULL THEN 1::int END AS nextpass,
CASE WHEN B.r2 IS NOT NULL AND C.r2 IS NOT NULL THEN A.r2 - (2*B.r2) + C.r2
 WHEN B.r2 IS NOT NULL AND C.r2 IS NULL THEN A.r2 - (2*B.r2) + 1
 WHEN B.r2 IS NULL AND C.r2 IS NULL THEN A.r2 - 1
END AS seconddiff
FROM HCLCBSAvariance AS A
LEFT JOIN HCLCBSAvariance AS B ON A.cbsa=B.cbsa AND A.year=B.year AND (A.layer
+1)=B.layer
LEFT JOIN HCLCBSAvariance AS C ON A.cbsa=C.cbsa AND A.year=C.year AND (A.layer
+2)=C.layer;

DROP TABLE IF EXISTS HCLCBSAVariancePassBlowout;
CREATE TABLE HCLCBSAVariancePassBlowout AS
SELECT A.*, B.layer AS layerbase, B.pass as PassBase
FROM HCLCBSAVarianceDiffs AS A
LEFT JOIN HCLCBSAVarianceDiffs AS B ON A.cbsa=B.cbsa AND A.year=B.year AND
A.layer<=B.layer;

DROP TABLE IF EXISTS HCLCBSAVariancePass;
CREATE TABLE HCLCBSAVariancePass AS
```

```
SELECT cbsa, year, layer, varex0, firstdiff, seconddiff,
Count(layerBase) AS forwardlayers, sum(PassBase) as forwardpasses,
max(layerbase) as finallayer
FROM HCLCBSAVariancePassBlowout
GROUP BY cbsa, year, layer, varex0, firstdiff, seconddiff
ORDER BY cbsa, year, layer;

DROP TABLE IF EXISTS HCLCBSAVarianceEligible;
CREATE TABLE HCLCBSAVarianceEligible AS
SELECT  cbsa, year, layer, varex0, firstdiff, seconddiff, forwardlayers,
finallayer
FROM HCLCBSAVariancePass WHERE forwardlayers=forwardpasses;

DROP TABLE IF EXISTS HCLCBSAVarianceSelect;
CREATE TABLE HCLCBSAVarianceSelect AS
WITH Minchoice AS (
 SELECT cbsa, year, min(seconddiff) AS minseconddiff, max(layer) as lastlayer
 FROM HCLCBSAVarianceEligible GROUP BY cbsa, year
)
SELECT A.cbsa, A.year, (A.layer+1) AS ElbowLayer, finallayer
FROM HCLCBSAVarianceEligible AS A
JOIN Minchoice AS B ON A.cbsa=B.cbsa AND A.year=B.year AND
A.seconddiff=B.minseconddiff
WHERE (A.layer+1)<=B.lastlayer AND B.lastlayer>2;

--########################
--#Make the HclCBSA geometries
--########################

DROP TABLE IF EXISTS HclCBSAcount;
CREATE TABLE HclCBSAcount as
SELECT cbsa, year, layer, cluster, count(coname) as clcount, count(DISTINCT
lat) as dlatcount, count(DISTINCT long) as dlongcount
FROM HclCBSA GROUP BY cbsa, year, layer, cluster;

DROP INDEX IF EXISTS HclCBSAcount_idx;
CREATE INDEX HclCBSAcount_idx ON HclCBSAcount(cbsa,year,layer,cluster);

DROP TABLE IF EXISTS HclCBSAsingleset;
CREATE TABLE HclCBSAsingleset as
SELECT A.*, clcount, 1::int as singleton, 0::int AS multiton, 0::int as Hull
FROM HclCBSA AS A
LEFT JOIN HclCBSAcount AS B ON A.cbsa=B.cbsa AND A.year=B.year AND
A.layer=B.layer AND A.cluster=B.cluster
where clcount=1;

DROP TABLE IF EXISTS HclCBSAmultiset;
CREATE TABLE HclCBSAmultiset as
SELECT A.*, clcount, 0::int AS singleton, 1::int AS multiton, 0::int as Hull
FROM HclCBSA AS A
LEFT JOIN HclCBSAcount AS B ON A.cbsa=B.cbsa AND A.year=B.year AND
A.layer=B.layer AND A.cluster=B.cluster
WHERE dlatcount=1 AND dlongcount=1 AND clcount>1;

DROP TABLE IF EXISTS HclCBSAHullset;
CREATE TABLE HclCBSAHullset as
SELECT A.*, clcount, 0::int AS singleton, 0::int AS multiton, 1::int as Hull
FROM HclCBSA AS A
LEFT JOIN HclCBSAcount AS B ON A.cbsa = B.cbsa  AND A.year=B.year AND
A.layer=B.layer AND A.cluster=B.cluster
where (dlatcount>1 OR dlongcount>1) AND clcount>1;

DROP TABLE IF EXISTS HclCBSAsingletons;
CREATE TABLE HclCBSAsingletons AS
```

```
SELECT cbsa, year, layer, cluster, long, lat, clcount, singleton, multiton,
hull,
geography(ST_Transform(ST_SetSRID(ST_Makepoint(long,lat),4326),4326)) AS geog
FROM HclCBSAsingleset;

DROP TABLE IF EXISTS HclCBSAmultitons;
CREATE TABLE HclCBSAmultitons AS
SELECT cbsa, year, layer, cluster, avg(long) as long, avg(lat) as lat,
avg(clcount) as clcount,
0::int as singleton, 1::int as multiton, 0::int as hull,
geography(ST_Transform(ST_SetSRID(ST_Makepoint(avg(long),avg(lat)),4326),4326)
) AS geog
FROM  GROUP BY cbsa, year, layer, cluster;

DROP TABLE IF EXISTS HclCBSAhullpoints;
CREATE TABLE HclCBSAhullpoints AS
SELECT cbsa, year, layer, cluster, clcount, lat, long,
geometry(ST_Transform(ST_SetSRID(ST_Makepoint(long,lat),4326),4326)) AS
pointgeom
FROM HclCBSAHullset;

DROP TABLE IF EXISTS HclCBSAhullbase;
CREATE TABLE HclCBSAhullbase AS
SELECT cbsa, year, layer, cluster, avg(clcount) as clcount, avg(lat) as lat,
avg(long) as long,
CASE WHEN ST_GeometryType(ST_ConvexHull(ST_Collect(pointgeom)))='ST_Polygon'
THEN geography(ST_ConvexHull(ST_Collect(pointgeom))) ELSE NULL END AS
hullgeog,
CASE WHEN ST_GeometryType(ST_ConvexHull(ST_Collect(pointgeom)))='ST_Polygon'
THEN 0::int ELSE 1::int END AS pair,
CASE WHEN
ST_GeometryType(ST_ConvexHull(ST_Collect(pointgeom)))='ST_LineString' THEN
geography(ST_MakeLine(pointgeom)) ELSE NULL END AS pairgeog
FROM HclCBSAhullpoints
GROUP BY cbsa, year, layer, cluster;

DROP TABLE IF EXISTS HclCBSAHulls;
CREATE TABLE HclCBSAHulls AS
SELECT cbsa, year, layer, cluster,
CASE WHEN pair=0 THEN ST_Y(ST_Centroid(hullgeog)::geometry) ELSE lat END AS
lat,
CASE WHEN pair=0 THEN ST_X(ST_Centroid(hullgeog)::geometry) ELSE long END AS
long,
clcount,
0::int as singleton, 0::int as multiton,
CASE WHEN pair=1 THEN 0 ELSE 1::int END as hull,
pair,
CASE WHEN pair=1 THEN ST_Length(pairgeog)/100 ELSE NULL END AS pairlength,
CASE WHEN pair=0 THEN ST_Area(hullgeog)/10000 ELSE NULL END as hullarea,
CASE WHEN pair=0 THEN hullgeog WHEN pair=1 THEN pairgeog END as geog,
CASE WHEN pair=0 THEN geography(ST_Centroid(hullgeog)::geometry) ELSE
geography(ST_Transform(ST_SetSRID(ST_Makepoint(long,lat),4326),4326)) END AS
pointgeog
FROM HclCBSAhullbase;

DROP INDEX IF EXISTS HclCBSAsingletons_idx;
DROP INDEX IF EXISTS HclCBSAcount_idx;
DROP INDEX IF EXISTS HclCBSAcount_idx;
CREATE INDEX HclCBSAsingletons_idx ON HclCBSAHulls(cbsa,year,layer,cluster);
CREATE INDEX HclCBSAmultitons_idx ON HclCBSAHulls(cbsa,year,layer,cluster);
CREATE INDEX HclCBSAHulls_idx ON HclCBSAHulls(cbsa,year,layer,cluster);
--92532922 row(s) updated - 9m 24s

--########################
```

```
--Make HclCBSAMain
--########################

DROP TABLE IF EXISTS HclCBSAmain;
CREATE TABLE HclCBSAmain AS
SELECT cbsa, year, layer, cluster, lat, long, clcount, singleton, multiton,
hull, 0::int as pair, NULL::numeric as pairlength, NULL::numeric as hullarea,
geog
FROM HclCBSAsingletons
UNION
SELECT cbsa, year, layer, cluster, lat, long, clcount, singleton, multiton,
hull, 0::int as pair, NULL::numeric as pairlength, NULL::numeric as hullarea,
geog
FROM HclCBSAmultitons
UNION
SELECT cbsa, year, layer, cluster, lat, long, clcount, singleton, multiton,
hull, pair, pairlength, hullarea, geog
FROM HclCBSAHulls;

DROP INDEX IF EXISTS HclCBSAmain_idx;
CREATE INDEX HclCBSAmain_idx ON HclCBSAmain(cbsa,year,layer);

--########################
--Make HclCBSALayer
--########################

DROP TABLE IF EXISTS HclCBSAlayer;
CREATE TABLE HclCBSAlayer AS
WITH MAIN AS (
        SELECT cbsa, year, layer,
        sum(clcount) as lcount,
        sum(singleton) as nosingleton, sum(multiton) as nomultiton, sum(pair)
        as nopair, sum(hull) as nohull,
        COALESCE(sum(CASE WHEN singleton <> 0 THEN clcount ELSE NULL END),0)
        AS totsingletoncount,
        COALESCE(sum(CASE WHEN multiton <> 0 THEN clcount ELSE NULL END),0) AS
        totmultitoncount,
        COALESCE(avg(CASE WHEN multiton <> 0 THEN clcount ELSE NULL END),0) AS
        avgmultitoncount,
        COALESCE(sum(CASE WHEN pair <> 0 THEN clcount ELSE NULL END),0) AS
        totpaircount,
        COALESCE(avg(CASE WHEN pair <> 0 THEN clcount ELSE NULL END),0) AS
        avgpaircount,
        CASE WHEN sum(pairlength) IS NOT NULL THEN sum(pairlength) ELSE 0 END
        AS totpairlength,
        CASE WHEN avg(pairlength) IS NOT NULL THEN avg(pairlength) ELSE 0 END
        AS avgpairlength,
        CASE WHEN min(pairlength) IS NOT NULL THEN min(pairlength) ELSE 0 END
        AS minpairlength,
        CASE WHEN max(pairlength) IS NOT NULL THEN max(pairlength) ELSE 0 END
        AS maxpairlength,
        COALESCE(sum(CASE WHEN hull <> 0 THEN clcount ELSE NULL END),0) AS
        tothullcount,
        COALESCE(avg(CASE WHEN hull <> 0 THEN clcount ELSE NULL END),0) AS
        avghullcount,
        CASE WHEN sum(hullarea) IS NOT NULL THEN sum(hullarea) ELSE 0 END AS
        tothullarea,
        CASE WHEN avg(hullarea) IS NOT NULL THEN avg(hullarea) ELSE 0 END AS
        avghullarea,
        CASE WHEN min(hullarea) IS NOT NULL THEN min(hullarea) ELSE 0 END AS
        minhullarea,
        CASE WHEN max(hullarea) IS NOT NULL THEN max(hullarea) ELSE 0 END AS
        maxhullarea,
        CASE WHEN sum(hullarea) >0 THEN sum(clcount*hull)/sum(hullarea) ELSE 0
```

```sql
        END AS tothulldensity,
        CASE WHEN avg(hullarea) >0 THEN avg(CASE WHEN hullarea > 0 THEN
        ((clcount*hull)/hullarea) ELSE NULL END) ELSE 0 END AS avghulldensity
        FROM HclCBSAmain
        GROUP BY cbsa, year, layer ORDER BY cbsa, year, layer
)
SELECT A.*,
avghuldisthm
FROM Main AS A
LEFT JOIN HCLCBSAHullCentroidDist AS B ON A.cbsa=B.cbsa AND A.year=B.year AND
A.layer=B.layer;

--####################################
--Create some additional variables!
--####################################

DROP TABLE IF EXISTS HCACBSAPortCoYearVCBase;
CREATE TABLE HCACBSAPortCoYearVCBase AS
        SELECT A.cbsa, A.coname, A.statecode, A.datefirstinv,
        B.roundyear AS year,
        rndamtestm AS totalamnt,
        rndamtestm*B.seedflag AS seedamnt,
        rndamtestm*B.earlyflag AS earlyamnt,
        rndamtestm*B.laterflag AS lateramnt,
        rndamtestm*B.growthflag AS selamnt,
        rndamtestm*GREATEST(B.seedflag,B.earlyflag) AS seedearlyamnt,
        rndamtestm*B.transactionflag AS transamnt,
        B.growthflag*B.dealflag AS seldeal,
        B.dealflag, B.seedflag, B.earlyflag, B.laterflag, B.growthflag,
        B.transactionflag,
        GREATEST(B.seedflag,B.earlyflag) as seedearlyflag
        FROM (
                SELECT cbsa, coname, statecode, datefirstinv
                FROM PortCoHCACBSAInput
                GROUP BY cbsa, coname, statecode, datefirstinv
                ) AS A
        JOIN Round AS B ON A.coname=B.coname AND A.statecode=B.statecode AND
        A.datefirstinv=B.datefirstinv;

DROP TABLE IF EXISTS HCACBSAPortCoYearVC;
CREATE TABLE HCACBSAPortCoYearVC AS
SELECT cbsa, year, coname, statecode, datefirstinv,
sum(selamnt) as growthinv
FROM HCACBSAPortCoYearVCBase
GROUP BY cbsa, year, coname, statecode, datefirstinv;

DROP TABLE IF EXISTS HCACBSAYearNumstartups;
CREATE TABLE HCACBSAYearNumstartups AS
    SELECT  cbsa, year, count(*) as numstartups FROM PortCoHCACBSAInput GROUP
    BY cbsa, year;

DROP TABLE IF EXISTS HCACBSAyearvc;
CREATE TABLE HCACBSAyearvc AS SELECT
A.cbsa, A.year,
numstartups,
COALESCE(sum(growthinv), 0) AS  growthinv
FROM HCACBSAYearNumstartups AS A
LEFT JOIN HCACBSAPortCoYearVC AS B ON A.cbsa = B.cbsa  AND A.year=B.year
GROUP BY A.cbsa, A.year, numstartups
ORDER BY A.cbsa, A.year;

DROP TABLE IF EXISTS HCLCBSAHullClusters;
CREATE TABLE HCLCBSAHullClusters AS
SELECT cbsa, year, layer, cluster FROM HclCBSAHulls WHERE hull=1;
```

```
DROP INDEX IF EXISTS HCLCBSAHullClusters_idx;
CREATE INDEX HCLCBSAHullClusters_idx ON
HCLCBSAHullClusters(cbsa,year,layer,cluster);

DROP TABLE IF EXISTS HCLCBSAClusterPortCos;
CREATE TABLE HCLCBSAClusterPortCos AS
SELECT A.cbsa, A.year, A.layer, A.cluster, A.coname, A.datefirstinv,
A.portcostatecode as statecode
FROM hclCBSA AS A
JOIN HCLCBSAHullClusters AS B ON A.cbsa=B.cbsa AND A.year=B.year AND
A.layer=B.layer AND A.cluster=B.cluster;

DROP INDEX IF EXISTS HCLCBSAClusterPortCos_idx;
CREATE INDEX HCLCBSAClusterPortCos_idx ON
HCLCBSAClusterPortCos(cbsa,year,coname, datefirstinv, statecode);

DROP TABLE IF EXISTS HCLCBSAClusterPortCosVC;
CREATE TABLE HCLCBSAClusterPortCosVC AS
SELECT A.*, COALESCE(growthinv, 0) AS growthinv
FROM HCLCBSAClusterPortCos AS A
LEFT JOIN HCACBSAPortCoYearVC AS B ON A.cbsa=B.cbsa AND A.year=B.year
        AND A.coname=B.coname AND A.datefirstinv=B.datefirstinv AND
        A.statecode=B.statecode;

DROP TABLE IF EXISTS HCLCBSAClusterVC;
CREATE TABLE HCLCBSAClusterVC AS
SELECT  A.cbsa, A.year, A.layer, COALESCE(sum(growthinv), 0) as
growthinvcluster
FROM HCLCBSAClusterPortCosVC AS A
GROUP BY A.cbsa, A.year, A.layer;

DROP TABLE IF EXISTS HCACBSATigerArea;
CREATE TABLE HCACBSATigerArea AS
WITH data as (
        SELECT cbsa, numstartups,
        ln(numstartups)::int as lognumberofstartups,
        CASE WHEN numstartups <= 10 THEN 10
                WHEN numstartups <= 20 THEN 20
                WHEN numstartups <= 50 THEN 50
                WHEN numstartups <= 100 THEN 100
                WHEN numstartups <= 200 THEN 200
                WHEN numstartups <= 500 THEN 500
                WHEN numstartups <= 800 THEN 800
                WHEN numstartups <= 1200 THEN 1200
                WHEN numstartups <= 1900 THEN 1900
                ELSE NULL END AS startupcountgroup
        FROM mastercbsalayers WHERE year=2020 and Layer=0
)
SELECT A.cbsa, ST_Area(ST_CurveToLine(geom)::geography)/10000 as cbsaarea,
ST_CurveToLine(geom)::geography as cbsageog,
numstartups, lognumberofstartups,startupcountgroup
FROM (SELECT DISTINCT cbsa FROM HCACBSAYearNumstartups) AS A
LEFT JOIN TigerCBSAgeom AS B ON A.cbsa=B.cbsa
LEFT JOIN Data as C ON A.cbsa=C.cbsa;

--####################################
--HclCBSALayerCounts
--####################################

DROP TABLE IF EXISTS HclCBSALayerStats;
CREATE TABLE HclCBSALayerStats AS
SELECT A.cbsa, A.year,
A.Numlayers as numlayersthisyear,
B.minnumlayers as minnumlayersallyears,
B.maxnumlayers as maxnumlayersallyears
FROM (
         SELECT cbsa, year, count(layer) AS NumLayers
         FROM HclCBSAlayer
         GROUP BY cbsa, year
     ) AS A
JOIN (
        SELECT cbsa, min(numlayers) as minnumlayers, max(numlayers) as
        maxnumlayers
                FROM (
                        SELECT cbsa, year, count(layer) AS NumLayers
                        FROM HclCBSAlayer GROUP BY cbsa, year
                ) AS T GROUP BY cbsa
    ) AS B ON A.cbsa = B.cbsa ;

--##################################
--Hulls based analysis
--##################################

DROP TABLE IF EXISTS HullsCBSABase;
CREATE TABLE HullsCBSABase AS
    SELECT cbsa, year, layer, count(cluster) as numclusters
    FROM HclCBSAmain WHERE hull=1 GROUP BY cbsa, year, layer ORDER BY cbsa,
    year, layer;

DROP TABLE IF EXISTS HullsCBSAFirst;
CREATE TABLE HullsCBSAFirst AS
SELECT cbsa, year, numclusters, min(layer) as firstlayer
FROM HullsCBSABase GROUP BY cbsa, year, numclusters;

DROP TABLE IF EXISTS HullsCBSAwPrev;
CREATE TABLE HullsCBSAwPrev AS
    SELECT A.cbsa, A.year, A.layer, A.numclusters, B.Layer AS Blayer,
    B.numclusters as Bnumclusters,
        CASE WHEN A.numclusters IS NOT NULL AND A.numclusters=B.numclusters
        THEN 1::int ELSE 0::int END AS sameasprev
    FROM HullsCBSABase AS A
    LEFT JOIN HullsCBSABase AS B ON A.cbsa = B.cbsa AND A.year=B.year AND
    A.layer=(B.layer+1)
    ORDER BY A.cbsa, A.year, A.layer;

DROP SEQUENCE IF EXISTS HullsCBSAset;
CREATE SEQUENCE HullsCBSAset START 1;

DROP TABLE IF EXISTS HullsCBSASeq;
CREATE TABLE HullsCBSASeq AS
    SELECT A.cbsa, A.year, A.layer, A.numclusters, A.sameasprev,
           CASE WHEN sameasprev=0 THEN nextval('HullsCBSAset') WHEN
           sameasprev=1 THEN currval('HullsCBSAset') END as HullsCBSAeq
    FROM HullsCBSAwPrev AS A ORDER BY A.cbsa, A.year, A.layer;

DROP TABLE IF EXISTS HullsCBSAHighest;
CREATE TABLE HullsCBSAHighest AS
    SELECT cbsa, year, numclusters, max(layer) as highestlayer
    FROM HullsCBSASeq GROUP BY cbsa, year, numclusters;

DROP TABLE IF EXISTS HullsCBSAHighestSeq;
CREATE TABLE HullsCBSAHighestSeq AS
    SELECT A.cbsa, A.year, A.layer, A.numclusters, A.HullsCBSAeq
    FROM HullsCBSASeq AS A
    JOIN HullsCBSAHighest AS B ON A.cbsa = B.cbsa AND A.year=B.year AND
    A.layer=B.highestlayer;
```

```
DROP TABLE IF EXISTS HullsCBSALowestHighest;
CREATE TABLE HullsCBSALowestHighest AS
    SELECT cbsa, year, numclusters, HullsCBSAeq, min(layer) as
    lowesthighestlayer
    FROM (
             SELECT A.cbsa, A.year, A.layer, A.numclusters, A.HullsCBSAeq
             FROM HullsCBSASeq AS A
                      JOIN HullsCBSAHighestSeq AS B
                           ON A.cbsa = B.cbsa AND A.year = B.year AND
                              A.HullsCBSAeq = B.HullsCBSAeq
        ) AS T GROUP BY cbsa, year, numclusters, HullsCBSAeq
        ORDER BY cbsa, year, min(layer);

DROP TABLE IF EXISTS HullsCBSAMax;
CREATE TABLE HullsCBSAMax AS
    SELECT cbsa, year, max(numclusters) as maxnumclusters
    FROM HullsCBSALowestHighest GROUP BY cbsa, year;

DROP TABLE IF EXISTS HullsCBSAMaxLayer;
CREATE TABLE HullsCBSAMaxLayer AS
WITH BASE AS (
        SELECT A.cbsa, A.year, A.layer, B.maxnumclusters
        FROM HullsCBSABase AS A
        JOIN HullsCBSAMax AS B ON A.cbsa = B.cbsa AND A.year = B.year
        WHERE A.numclusters=B.maxnumclusters)
SELECT cbsa, year, min(layer) as maxhulllayer, max(maxnumclusters) as
maxnumclusters FROM Base
GROUP BY cbsa, year ORDER BY cbsa, year;

DROP TABLE IF EXISTS HullsCBSAStats;
CREATE TABLE HullsCBSAStats AS
    SELECT A.cbsa, A.year, A.layer, A.numclusters, B.lowesthighestlayer,
    CASE WHEN A.layer=B.lowesthighestlayer THEN 1::int ELSE 0::int END AS
    lowesthighestflag,
    C.maxnumclusters, lastlayer,
    E.maxhulllayer,
    firstlayer,
    CASE WHEN A.layer=F.firstlayer THEN 1::int ELSE 0::int END AS
    firstlayerflag
    FROM HullsCBSABase AS A
    LEFT JOIN HullsCBSALowestHighest AS B ON A.cbsa = B.cbsa AND A.year =
    B.year AND A.layer=B.lowesthighestlayer
    LEFT JOIN HullsCBSAMax AS C on A.cbsa = C.cbsa AND A.year = C.year
    LEFT JOIN (
        SELECT cbsa, year, max(layer) as lastlayer FROM HullsCBSABase GROUP BY
        cbsa, year
        ) AS D on A.cbsa = D.cbsa AND A.year = D.year
    LEFT JOIN HullsCBSAMaxLayer AS E ON A.cbsa = E.cbsa AND A.year = E.year
    LEFT JOIN HullsCBSAFirst AS F ON A.cbsa = F.cbsa AND A.year = F.year AND
    A.numclusters=F.numclusters;

--###################################
--Distance between Hull Centroids (For specific layers only)
--###################################

DROP TABLE IF EXISTS HCLCBSAHullCentroidRelevant;
CREATE TABLE HCLCBSAHullCentroidRelevant AS
SELECT cbsa, year, lowesthighestlayer AS layer
FROM HullsCBSALowestHighest
UNION SELECT cbsa, year, elbowlayer as layer
FROM HCLCBSAVarianceSelect;

DROP TABLE IF EXISTS HCLCBSAHullCentroidBase;
CREATE TABLE HCLCBSAHullCentroidBase AS
```

```
SELECT A.cbsa, A.year, A.layer, A.cluster, A.pointgeog
FROM HclCBSAHulls AS A
JOIN HCLCBSAHullCentroidRelevant AS B ON A.cbsa = B.cbsa AND A.year = B.year
AND A.layer=B.layer
WHERE A.hull=1;

DROP INDEX IF EXISTS HCLCBSAHullCentroidBase_idx;
CREATE INDEX HCLCBSAHullCentroidBase_idx ON
HCLCBSAHullCentroidBase(cbsa,year,layer);

DROP TABLE IF EXISTS HCLCBSAHullCentroidBlowout;
CREATE TABLE HCLCBSAHullCentroidBlowout AS
SELECT A.cbsa, A.year, A.layer, A.cluster as clusterA, B.cluster as clusterB,
ST_Distance(A.pointgeog,B.pointgeog)/100 AS hulldisthm
FROM HCLCBSAHullCentroidBase AS A
JOIN HCLCBSAHullCentroidBase AS B ON A.cbsa=B.cbsa AND A.year=B.year AND
A.layer=B.layer;

DROP TABLE IF EXISTS HCLCBSAHullCentroidDist;
CREATE TABLE HCLCBSAHullCentroidDist AS
SELECT cbsa, year, layer, avg(hulldisthm) as avghulldisthm
FROM HCLCBSAHullCentroidBlowout
GROUP BY cbsa, year, layer;

--##################################
--Load and process the max r2 representative layer
--##################################

DROP TABLE IF EXISTS HCLlayerCBSAchosenhull;
CREATE TABLE HCLlayerCBSAchosenhull
(
    cbsa            varchar(255),
    year             int,
    chosenhulllayer int,
    chosenhullcount  int
);

\COPY HCLlayerCBSAchosenhull FROM 'ChosenHullCBSALayersv20-2.txt' WITH
DELIMITER AS E'\t' HEADER NULL AS '' CSV

DROP TABLE IF EXISTS HCLlayerCBSAchosenhullcount;
CREATE TABLE HCLlayerCBSAchosenhullcount AS
    SELECT cbsa, max(chosenhullcount) as chosenhullcountselected,
    stddev(chosenhullcount::numeric) AS chosenhullcountsd
    FROM HCLlayerCBSAchosenhull GROUP BY cbsa;

DROP TABLE IF EXISTS HCLLayerCBSAMaxR2Base;
CREATE TABLE HCLLayerCBSAMaxR2Base AS
SELECT A.cbsa, A.year, A.numclusters,
       CASE WHEN C.chosenhulllayer IS NOT NULL then C.chosenhulllayer ELSE
       A.lowesthighestlayer END AS bestlayercandidate,
       CASE WHEN C.chosenhulllayer IS NOT NULL then 0::int ELSE 1::int END AS
       usinghighestlowest,
       A.lowesthighestlayer,
       B.chosenhullcountselected AS targetnumclusters,
       C.chosenhulllayer, C.chosenhullcount,
       CASE WHEN B.chosenhullcountselected=A.numclusters THEN 1::int ELSE
       0::int END AS bestischosen
    FROM HullsCBSALowestHighest AS A
    LEFT JOIN HCLlayerCBSAchosenhullcount AS B ON A.cbsa=B.cbsa
    LEFT JOIN HCLlayerCBSAchosenhull AS C ON A.cbsa=C.cbsa AND A.year=C.year
    WHERE B.chosenhullcountselected IS NOT NULL AND A.numclusters
    <=B.chosenhullcountselected;
--Updated Rows  14589
```

```
DROP TABLE IF EXISTS HCLLayerCBSAMaxR2;
CREATE TABLE HCLLayerCBSAMaxR2 AS
    WITH maxclusters AS (
        SELECT cbsa, year, max(numclusters) as maxnumclusters FROM
        HCLLayerCBSAMaxR2Base GROUP BY cbsa, year
    )
    SELECT A.cbsa, A.year, numclusters, targetnumclusters, chosenhulllayer,
           bestlayercandidate as besthulllayer,
           usinghighestlowest as besthulllayerisadded,
           bestischosen
    FROM HCLLayerCBSAMaxR2Base AS A
    JOIN maxclusters AS B ON A.cbsa=B.cbsa AND A.year=B.year AND
    A.numclusters=B.maxnumclusters
    ORDER BY cbsa, year;

--####################################
--Load and process the quadratic max r2 layer
--####################################

DROP TABLE IF EXISTS HCLlayerCBSACubicHull;
CREATE TABLE HCLlayerCBSACubicHull
(
    cbsaid          int,
        hsolve  int,
        hpick  int,
        method   varchar(255),
        hmin   int,
        hmax   int,
    cbsa  varchar(255)
);

\COPY HCLlayerCBSACubicHull FROM 'cubicresults95_processed.txt' WITH DELIMITER
AS E'\t' HEADER NULL AS '' CSV

DROP TABLE IF EXISTS HCLLayerCBSACubicMax;
CREATE TABLE HCLLayerCBSACubicMax AS
SELECT A.cbsa, A.year, A.lowesthighestlayer as cubicmaxlayer, B.hsolve AS
cubichullcount
FROM HullsCBSALowestHighest AS A
JOIN HCLlayerCBSACubicHull AS B ON A.cbsa=B.cbsa AND A.numclusters=B.hsolve;

DROP TABLE IF EXISTS HCLLayerCBSACubicMaxHulls;
CREATE TABLE HCLLayerCBSACubicMaxHulls AS
SELECT A.cbsa, A.year, A.layer, A.cluster, A.geog, A.clcount
FROM HclCBSAmain AS A
JOIN HCLLayerCBSACubicMax AS B ON A.cbsa=B.cbsa AND A.year=B.year AND
A.layer=B.cubicmaxlayer
WHERE A.hull=1;

DROP TABLE IF EXISTS HCACBSACluster2000to2000Base;
CREATE TABLE HCACBSACluster2000to2000Base AS
WITH y2000 AS (
        SELECT * FROM HCLLayerCBSACubicMaxHulls WHERE year=2000
), y2020 AS (
        SELECT * FROM HCLLayerCBSACubicMaxHulls WHERE year=2020
), Blowout AS (
        SELECT A.cbsa, A.layer as l00, A.cluster as c00, A.geog as g00,
        A.clcount as s00,
        B.layer as l20, B.cluster AS c20, B.geog as g20, B.clcount as s20,
        CASE WHEN ST_INTERSECTS(A.geog,B.geog) IS TRUE THEN 1 ELSE 0 END AS
        intersects
        FROM y2000 AS A
        LEFT JOIN y2020 AS B ON A.cbsa=B.cbsa
)
SELECT cbsa, l00, c00, s00,
CASE WHEN (sum(intersects)>0) THEN 1 ELSE 0 END AS ExistsIn20,
sum(intersects) AS CountC20,
CASE WHEN (sum(intersects)>0) THEN sum(s20*intersects) ELSE 0 END AS s20,
CASE WHEN (sum(intersects)>0) THEN sum(s20*intersects)-s00 ELSE 0 END AS
s_add,
CASE WHEN (sum(intersects)>0) THEN (sum(s20*intersects)-s00)/s00 ELSE 0 END AS
s_pcchange
FROM Blowout GROUP BY cbsa, l00, c00, s00
ORDER BY cbsa, l00, c00;

--###################################
--And put it all together
--###################################

DROP TABLE IF EXISTS MasterCBSALayers;
CREATE TABLE MasterCBSALayers AS
SELECT A.*,
        ElbowLayer, Finallayer,
        A.Layer::numeric/Finallayer::numeric AS Layerindex,
        round(A.Layer::numeric/Finallayer::numeric,2) as layerindex2dg,
        r2, Fstat,
        D.numstartups,
        growthinv,
        cpi, inflator20,
        H.numclusters, lowesthighestlayer, lowesthighestflag, maxnumclusters,
        maxhulllayer,  firstlayer, firstlayerflag,
        numlayersthisyear, minnumlayersallyears, maxnumlayersallyears,
        J.avghulldisthm,
        targetnumclusters, chosenhulllayer, besthulllayer,
        besthulllayerisadded, bestischosen,
        COALESCE(growthinvcluster,0) AS growthinvcluster,
        cubicmaxlayer,cubichullcount
FROM HCLCBSALayer AS A
LEFT JOIN HCLCBSAVarianceSelect AS B ON A.cbsa=B.cbsa AND A.year=B.year
LEFT JOIN hclCBSAvariance AS C ON A.cbsa=C.cbsa AND A.year=C.year AND
A.layer=C.layer
LEFT JOIN HCACBSAyearvc AS D ON A.cbsa=D.cbsa AND A.year=D.year
LEFT JOIN HCACBSATigerArea AS E ON A.cbsa=E.cbsa
LEFT JOIN cpi AS G ON A.year=G.year
LEFT JOIN HullsCBSAStats AS H ON A.cbsa=H.cbsa AND A.year=H.year AND
A.layer=H.layer
LEFT JOIN HCLCBSALayerStats AS I ON A.cbsa=I.cbsa AND A.year=I.year
LEFT JOIN HCLCBSAHullCentroidDist AS J ON A.cbsa=J.cbsa AND A.year=J.year AND
A.layer=J.layer
LEFT JOIN HCLLayerCBSAMaxR2 AS K ON A.cbsa=K.cbsa AND A.year=K.year
LEFT JOIN HCLCBSAClusterVC AS L ON A.cbsa=L.cbsa AND A.year=L.year AND
A.layer=L.layer
LEFT JOIN HCLLayerCBSACubicMax AS M ON A.cbsa=M.cbsa AND A.year=M.year;

\COPY MasterCBSALayers TO 'MasterCBSALayersv20-7.txt' WITH DELIMITER AS E'\t'
HEADER NULL AS '' CSV
```