USE [pacs\_oltp]

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[ExportPropertyAccess] Script Date: 9/16/2024 2:49:07 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

/\*

\* This procedure builds the web\_internet\_<countyname> database for the

\* True Automation PropertyAccess website. Formerly ClientDB.

\*/

CREATE PROCEDURE [dbo].[ExportPropertyAccess]

@input\_database\_name varchar(50) = '',

@input\_num\_years int = -1,

@input\_min\_bill\_years int = 2

--WITH RECOMPILE

AS

SET NOCOUNT ON

declare @sql varchar(max)

declare @sql2 varchar(max)

declare @log\_id int

declare @error\_flag int

declare @texas varchar(10)

declare @washington varchar(10)

declare @status varchar (250)

declare @start\_date datetime

set @texas = 'TX'

set @washington = 'WA'

set @start\_date = getdate()

if (len(@input\_database\_name) = 0)

begin

declare @county\_name varchar(20)

select @county\_name = lower(isnull(county\_name, 'cad')) from system\_address where system\_type = 'A'

set @input\_database\_name = 'web\_internet\_' + isnull(@county\_name, 'cad') + '\_auto'

end

if @input\_min\_bill\_years < 2

set @input\_min\_bill\_years = 2

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

-- clean temp tables

if object\_id('tempdb..#commercial\_dor') is not null

drop table #commercial\_dor

if object\_id('tempdb..#farm\_dor') is not null

drop table #farm\_dor

if object\_id('tempdb..#imprv\_adj') is not null

drop table #imprv\_adj

if object\_id('tempdb..#land\_detail') is not null

drop table #land\_detail

if object\_id('tempdb..#layer\_assoc') is not null

drop table #layer\_assoc

if object\_id('tempdb..#residential\_dor') is not null

drop table #residential\_dor

if object\_id('tempdb..#tables') is not null

drop table #tables

if object\_id('tempdb..#tmpPropVal') is not null

drop table #tmpPropVal

if object\_id('tempdb..#tmpDetail') is not null

drop table #tmpDetail

if object\_id('tempdb..#tmpImprvDetail') is not null

drop table #tmpImprvDetail

if object\_id('tempdb..#tmpImpDetVal') is not null

drop table #tmpImpDetVal

if object\_id('tempdb..#tmpMainArea') is not null

drop table #tmpMainArea

if object\_id('tempdb..#tmpDups') is not null

drop table #tmpDups

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

--CREATE NEW WEB DATABASE IF IT DOESN'T EXIST

declare @db\_path varchar(255)

select @db\_path = filename from master..sysdatabases where name = db\_name()

set @db\_path = reverse(@db\_path)

set @db\_path = right(@db\_path, len(@db\_path) - charindex('\', @db\_path, 1) + 1) --'escape text fix

set @db\_path = reverse(@db\_path)

-- Create Database if not exists

if not exists (select \* from master..sysdatabases where name = @input\_database\_name)

begin

set @sql = 'CREATE DATABASE ' + @input\_database\_name + ' ON

(

NAME = ' + @input\_database\_name + ',

SIZE = 5GB,

FILEGROWTH = 256MB,

FILENAME = ''' + @db\_path + @input\_database\_name + '.MDF' + '''' + '

)

LOG ON

(

NAME = ' + @input\_database\_name + '\_log,

SIZE = 3GB,

FILEGROWTH = 128MB,

FILENAME = ''' + @db\_path + @input\_database\_name + '.LDF' + '''' + '

)'

exec(@sql)

print ' Done Creating Database '+ @input\_database\_name + ' at ' + convert(varchar(30), getdate(), 109)

end

set @sql = 'alter database ' + @input\_database\_name + ' set recovery simple'

exec(@sql)

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

if (select category from master..sysdatabases where name = @input\_database\_name) > 0

begin

print 'Dropping Replication'

-- Start: drop publications

-- Drop all subscriptions if they exist.

set @sql = '

if exists (

select \*

from ' + @input\_database\_name + '..syssubscriptions

where dest\_db = ''' + @input\_database\_name + '''

)

begin

exec ' + @input\_database\_name + '..sp\_dropsubscription @publication = N''all'', @article = N''all'', @subscriber = N''all'', @destination\_db = N''all''

end

'

exec (@sql)

-- Dropping the snapshot publication

-- Drop publications only if they exists

set @sql = '

if exists (

select \*

from ' + @input\_database\_name + '..syspublications

where name = ''' + @input\_database\_name + '''

)

begin

exec ' + @input\_database\_name + '..sp\_droppublication @publication = N''all''

end

'

exec (@sql)

-- Stop: drop publications

print ' Done Dropping Replication at ' + convert(varchar(30), getdate(), 109)

end

-- Start: drop objects

if exists (select \* from master..sysdatabases where name = @input\_database\_name)

begin

print 'Dropping Objects in ' + @input\_database\_name

DECLARE @obj\_name varchar(255)

DECLARE @type char (2)

declare @order\_by Integer

DECLARE @key\_tablename varchar(128)

set @sql = 'declare cur\_object insensitive cursor for select so.name, so.xtype, lOrderByDummy = case when so.xtype = ''F'' then 0 else 1 end, stables.name

from ' + @input\_database\_name + '..sysobjects as so

left outer join ' + @input\_database\_name + '..sysconstraints as sc on sc.constid = so.id

left outer join ' + @input\_database\_name + '..sysobjects as stables on stables.id = sc.id

where so.xtype in (''U'',''P'',''FN'',''V'',''F'', ''TF'')

--bitwise - detect non system object

and (so.category & 2) = 0

order by lOrderByDummy

for read only'

execute(@sql)

DECLARE @drop\_obj varchar(255)

OPEN cur\_object

FETCH NEXT FROM cur\_object

INTO @obj\_name, @type, @order\_by, @key\_tablename

WHILE @@FETCH\_STATUS = 0

BEGIN

set @sql = 'use ' + @input\_database\_name + ' '

set @drop\_obj =

CASE @type

When 'P' Then ' drop procedure ' + @obj\_name

When 'V' Then ' drop view ' + @obj\_name

When 'U' Then ' truncate table ' + @obj\_name + '; drop table ' + @obj\_name

When 'TF' Then ' drop function ' + @obj\_name

When 'F' Then ' alter table ' + @key\_tablename + ' drop constraint ' + @obj\_name

END

set @sql = 'use ' + @input\_database\_name + ' ' + @drop\_obj

exec (@sql)

FETCH NEXT FROM cur\_object

INTO @obj\_name, @type, @order\_by, @key\_tablename

END

CLOSE cur\_object

DEALLOCATE cur\_object

print ' Done Dropping Droping Objects in '+ @input\_database\_name + ' at ' + convert(varchar(30), getdate(), 109)

end

-- Stop: drop objects

--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

-- Start: Export Process

print 'Exporting!'

declare @use\_col\_owner\_id bit

select

@use\_col\_owner\_id = case szConfigValue when 'T' then 1 else 0 end

from pacs\_config with (nolock)

where szGroup = 'Property Access' and szConfigname = 'col\_owner\_id'

set @use\_col\_owner\_id = isnull(@use\_col\_owner\_id, 0)

declare @region varchar(5)

select @region = szConfigValue

from core\_config with (nolock) where szConfigName = 'REGION'

if(@region is null) set @region = 'WA'

declare @collections\_only bit

select

@collections\_only = case szConfigValue when 'T' then 1 else 0 end

from pacs\_config with (nolock)

where szGroup = 'Property Access' and szConfigname = 'Collections Only'

set @collections\_only = isnull(@collections\_only, 0)

declare @hide\_minerals bit

select

@hide\_minerals = case szConfigValue when 'T' then 1 else 0 end

from pacs\_config with (nolock)

where szGroup = 'Property Access' and szConfigname = 'hide\_minerals'

set @hide\_minerals = isnull(@hide\_minerals, 0)

declare @township\_enabled bit

select

@township\_enabled = case szConfigValue when 'T' then 1 else 0 end

from pacs\_config with (nolock)

where szGroup = 'SYSTEM' and szConfigname = 'Township Enabled'

set @township\_enabled = isnull(@township\_enabled, 0)

declare @all\_imprv\_features bit

select

@all\_imprv\_features = case szConfigValue when 'T' then 1 else 0 end

from pacs\_config with (nolock)

where szGroup = 'Property Access' and szConfigname = 'All Improvement Features'

set @all\_imprv\_features = isnull(@all\_imprv\_features, 0)

declare @include\_adjustments bit

select

@include\_adjustments = case szConfigValue when 'T' then 1 else 0 end

from pacs\_config with (nolock)

where szGroup = 'Property Access' and szConfigname = 'Include Prior Year ADJ'

set @include\_adjustments = isnull(@include\_adjustments, 0)

set @error\_flag = 0

set @sql = ''

exec DropPATempTables

if not(exists(select id from dbo.sysobjects where id = object\_id(N'[dbo].[\_clientdb\_log]') and OBJECTPROPERTY(id, N'IsUserTable') = 1))

begin

create table \_clientdb\_log

(

[id] int not null,-- identity(1,1),

[start\_dt] datetime null,

[finish\_dt] datetime null,

[status] varchar(500) null,

[error] int null

)

set @log\_id = 0

end

else

begin

select @log\_id = max(id) from \_clientdb\_log

set @log\_id = @log\_id + 1

end

set @status = 'Starting Export'

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

--Year History - Export Info

create table \_clientdb\_pacs\_year

(

tax\_yr numeric(4,0) not null,

certification\_dt datetime null,

prev\_reappraised\_yr numeric(4,0) null

)

declare @max\_year numeric(4,0)

declare @sys\_tax\_year numeric(4,0)

declare @min\_year numeric(4,0)

select @max\_year = max(tax\_yr)

from pacs\_year with (nolock)

select @sys\_tax\_year = tax\_yr

from pacs\_system with (nolock)

if @input\_num\_years < 0

begin

insert \_clientdb\_pacs\_year

(tax\_yr, certification\_dt, prev\_reappraised\_yr)

select tax\_yr, certification\_dt, prev\_reappraised\_yr

from pacs\_year with (nolock)

order by tax\_yr

end

else

begin

set @min\_year = @max\_year - @input\_num\_years

insert \_clientdb\_pacs\_year

(tax\_yr, certification\_dt, prev\_reappraised\_yr)

select tax\_yr, certification\_dt, prev\_reappraised\_yr

from pacs\_year with (nolock)

where tax\_yr > @min\_year

order by tax\_yr

end

create nonclustered index IDX\_\_clientdb\_pacs\_year\_tax\_yr

on \_clientdb\_pacs\_year (tax\_yr)

with fillfactor = 90

-- Temp Table to store most recent ACCEPTED Supplement on a

-- property in a given year

set @status = 'Layer Assoc'

create table #layer\_assoc

(

owner\_tax\_yr numeric(4,0) not null,

sup\_num int not null,

prop\_id int not null,

primary key clustered (owner\_tax\_yr, sup\_num, prop\_id)

with fillfactor = 100

)

insert #layer\_assoc (owner\_tax\_yr, prop\_id, sup\_num)

select distinct pv.prop\_val\_yr, pv.prop\_id, max(pv.sup\_num)

from property\_val as pv with(nolock)

left outer join supplement as s with(nolock) on

s.sup\_tax\_yr = pv.prop\_val\_yr and

s.sup\_num = pv.sup\_num

left outer join sup\_group as sg with(nolock) on

sg.sup\_group\_id = s.sup\_group\_id

where (sg.status\_cd is null or sg.status\_cd in ('A','BC'))

group by pv.prop\_val\_yr, pv.prop\_id

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

--Deed History - Export Info

print 'Exporting Deed History... at ' + convert(varchar(30), getdate(), 109)

set @status = 'Deed History'

if @region = @texas

begin

set @sql = '

select

coopa.prop\_id,

coo.chg\_of\_owner\_id,

coopa.seq\_num,

convert(varchar(10), coo.deed\_dt, 101) as deed\_dt,

coo.deed\_type\_cd,

dt.deed\_type\_desc,

case when asell.confidential\_flag = ''T''

then (select IsNull(confidential\_file\_as\_name, ''Confidential'') confidential\_file\_as\_name from pacs\_system)

else isnull(asell.file\_as\_name, coo.grantor\_cv)

end as grantor,

case when abuy.confidential\_flag = ''T''

then (select IsNull(confidential\_file\_as\_name, ''Confidential'') confidential\_file\_as\_name from pacs\_system)

else isnull(abuy.file\_as\_name, coo.grantee\_cv)

end as grantee,

coo.deed\_book\_id,

coo.deed\_book\_page,

coo.coo\_sl\_dt as sale\_date,

coo.consideration as sale\_price,

null as excise\_number,

coo.deed\_num

into \_clientdb\_deed\_history\_detail

from chg\_of\_owner\_prop\_assoc as coopa with (nolock)

join chg\_of\_owner as coo with (nolock)

on coopa.chg\_of\_owner\_id = coo.chg\_of\_owner\_id

left outer join deed\_type as dt with (nolock)

on coo.deed\_type\_cd = dt.deed\_type\_cd

left outer join buyer\_assoc as ba with (nolock)

on coopa.chg\_of\_owner\_id = ba.chg\_of\_owner\_id

left outer join seller\_assoc as sa with (nolock)

on coopa.chg\_of\_owner\_id = sa.chg\_of\_owner\_id

and coopa.prop\_id = sa.prop\_id

left outer join account as abuy with (nolock)

on ba.buyer\_id = abuy.acct\_id

left outer join account as asell with (nolock)

on sa.seller\_id = asell.acct\_id

where coopa.seq\_num < 3 -- Last three sales 0, 1, 2

'

end

else --Washington

begin

set @sql = '

select

coopa.prop\_id,

coo.chg\_of\_owner\_id,

coopa.seq\_num,

convert(varchar(10), coo.deed\_dt, 101) as deed\_dt,

coo.deed\_type\_cd,

dt.deed\_type\_desc,

case when asell.confidential\_flag = ''T''

then (select IsNull(confidential\_file\_as\_name, ''Confidential'') confidential\_file\_as\_name from pacs\_system)

else isnull(asell.file\_as\_name, coo.grantor\_cv)

end as grantor,

case when abuy.confidential\_flag = ''T''

then (select IsNull(confidential\_file\_as\_name, ''Confidential'') confidential\_file\_as\_name from pacs\_system)

else isnull(abuy.file\_as\_name, coo.grantee\_cv)

end as grantee,

coo.deed\_book\_id,

coo.deed\_book\_page,

coo.coo\_sl\_dt as sale\_date,

s.adjusted\_sl\_price as sale\_price,

coo.excise\_number,

coo.deed\_num

into \_clientdb\_deed\_history\_detail

from chg\_of\_owner\_prop\_assoc as coopa with (nolock)

join chg\_of\_owner as coo with (nolock)

on coopa.chg\_of\_owner\_id = coo.chg\_of\_owner\_id

left outer join deed\_type as dt with (nolock)

on coo.deed\_type\_cd = dt.deed\_type\_cd

left outer join buyer\_assoc as ba with (nolock)

on coopa.chg\_of\_owner\_id = ba.chg\_of\_owner\_id

left outer join seller\_assoc as sa with (nolock)

on coopa.chg\_of\_owner\_id = sa.chg\_of\_owner\_id

and coopa.prop\_id = sa.prop\_id

left outer join account as abuy with (nolock)

on ba.buyer\_id = abuy.acct\_id

left outer join account as asell with (nolock)

on sa.seller\_id = asell.acct\_id

left outer join sale as s with (nolock)

on s.chg\_of\_owner\_id = coopa.chg\_of\_owner\_id

'

end

print @sql

exec(@sql) --populate \_clientdb\_deed\_history\_detail

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print ' Done at ' + convert(varchar(30), getdate(), 109)

--Improvement History - Export Info

print 'Exporting Improvement Building Detail... at ' + convert(varchar(30), getdate(), 109)

set @status = 'Improvement Details'

-- rgoolsby performance changes, put propert\_val and imprv\_detail

-- fields needed in insert statement in temp tables to speed things up

select pv.prop\_val\_yr,pv.sup\_num,pv.prop\_id

,pv.appr\_method,isnull(imprv\_non\_hstd\_val, 0) as imprv\_non\_hstd\_val

INTO #tmpPropVal

from property\_val as pv with (nolock)

join \_clientdb\_pacs\_year as y with (nolock)

on pv.prop\_val\_yr = y.tax\_yr

join #layer\_assoc as psa with (nolock)

on pv.prop\_val\_yr = psa.owner\_tax\_yr

and pv.sup\_num = psa.sup\_num

and pv.prop\_id = psa.prop\_id

select i.prop\_val\_yr, i.sup\_num, i.sale\_id, i.prop\_id, i.imprv\_id, i.imprv\_det\_id

,i.imprv\_det\_area, i.imprv\_det\_type\_cd, i.imprv\_det\_class\_cd, i.imprv\_det\_sub\_class\_cd, i.yr\_built, i.imprv\_det\_val

into #tmpImprvDetail -- all improvement details

from imprv\_detail as i with (nolock)

join \_clientdb\_pacs\_year as y with (nolock)

on i.prop\_val\_yr = y.tax\_yr

join #layer\_assoc as psa with (nolock)

on i.prop\_val\_yr = psa.owner\_tax\_yr

and i.sup\_num = psa.sup\_num

and i.prop\_id = psa.prop\_id

create index idx\_tmpImprvDetail on #tmpImprvDetail (prop\_val\_yr, sup\_num, prop\_id, imprv\_id, imprv\_det\_id, imprv\_det\_val, imprv\_det\_area, imprv\_det\_type\_cd)

set @sql = 'select

i.prop\_id,

i.prop\_val\_yr,

i.imprv\_id,

it.imprv\_type\_desc,

i.imprv\_state\_cd,

(select sum(isnull(impd.imprv\_det\_area, 0)) as living\_area

from #tmpImprvDetail as impd with (nolock)

join imprv\_det\_type as impt with (nolock)

on impd.imprv\_det\_type\_cd = impt.imprv\_det\_type\_cd

where impd.prop\_id = i.prop\_id

and impd.prop\_val\_yr = i.prop\_val\_yr

and impd.sup\_num = i.sup\_num

and impd.imprv\_id = i.imprv\_id

and isnull(impt.main\_area, ''F'') = ''T''

) as living\_area,

-- HS 44190 Kevin Lloyd

imprv\_val =

CASE pv.appr\_method

WHEN ''I'' THEN -1

WHEN ''D'' THEN isnull(i.dist\_val,0)

WHEN ''C'' THEN isnull(i.imprv\_val,0)

WHEN ''A'' THEN isnull(i.arb\_val,0)

ELSE 0

END,

ii.imprv\_det\_id,

ii.imprv\_det\_type\_cd,

idt.imprv\_det\_typ\_desc,

ii.imprv\_det\_class\_cd,

ii.imprv\_det\_sub\_class\_cd,

ia.i\_attr\_val\_cd,

ii.yr\_built,

ii.imprv\_det\_area as area,

-- HS 44190 Kevin Lloyd

isnull(pv.imprv\_non\_hstd\_val, 0) as imprv\_non\_hstd\_val,

''T'' as show\_values

into \_clientdb\_improvement\_building\_detail

from imprv as i with (nolock)

join #layer\_assoc as psa with (nolock)

on i.prop\_id = psa.prop\_id

and i.prop\_val\_yr = psa.owner\_tax\_yr

and i.sup\_num = psa.sup\_num

join \_clientdb\_pacs\_year as y with (nolock)

on i.prop\_val\_yr = y.tax\_yr

-- HS 44190 Kevin Lloyd

join #tmpPropVal as pv with (nolock)

on pv.prop\_val\_yr = i.prop\_val\_yr

and pv.sup\_num = i.sup\_num

and pv.prop\_id = i.prop\_id

join imprv\_type as it with (nolock)

on i.imprv\_type\_cd = it.imprv\_type\_cd

left outer join #tmpImprvDetail as ii with (nolock)

on i.prop\_val\_yr = ii.prop\_val\_yr

and i.sup\_num = ii.sup\_num

and i.prop\_id = ii.prop\_id

and i.imprv\_id = ii.imprv\_id

left outer join imprv\_attr as ia with (nolock)

on ii.imprv\_id = ia.imprv\_id

and ii.prop\_id = ia.prop\_id

and ii.imprv\_det\_id = ia.imprv\_det\_id

and ii.prop\_val\_yr = ia.prop\_val\_yr

and ii.sup\_num = ia.sup\_num

and ia.i\_attr\_val\_id = 3

left outer join imprv\_det\_type as idt with (nolock)

on ii.imprv\_det\_type\_cd = idt.imprv\_det\_type\_cd

where i.sale\_id = 0'

exec(@sql)

drop table #tmpPropVal

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print 'Exporting Improvement Features... at ' + convert(varchar(30), getdate(), 109)

set @status = 'Improvement Features'

create table \_clientdb\_improvement\_features

(

prop\_id int not null,

prop\_val\_yr numeric(4,0) not null,

imprv\_id int not null,

imprv\_det\_id int null,

imprv\_attr\_desc varchar(50),

i\_attr\_val\_cd varchar(75)

)

-- TODO: replace with Robbie's changes to remove duplicates

select prop\_val\_yr, sup\_num, sale\_id, prop\_id, imprv\_id, imprv\_det\_id, imprv\_det\_val

into #tmpMainArea -- all main areas

from #tmpImprvDetail as tID with (nolock)

join imprv\_det\_type as impt with (nolock)

on tID.imprv\_det\_type\_cd = impt.imprv\_det\_type\_cd

where isnull(impt.main\_area, 'F') = 'T'

-- now create temp table with highest value info per main area improvement

select ima.prop\_val\_yr, ima.sup\_num, ima.sale\_id, ima.prop\_id, ima.imprv\_id, max(ima.imprv\_det\_val) as imprv\_det\_val

into #tmpImpDetVal -- Highest valued improvement details on main areas

from #tmpMainArea as ima with (nolock)

join imprv\_attr as ia with (nolock)

on ia.prop\_id = ima.prop\_id

and ia.prop\_val\_yr = ima.prop\_val\_yr

and ia.sup\_num = ima.sup\_num

and ia.sale\_id = ima.sale\_id

and ia.imprv\_id = ima.imprv\_id

and ia.imprv\_det\_id = ima.imprv\_det\_id

group by ima.prop\_val\_yr, ima.sup\_num, ima.sale\_id, ima.prop\_id, ima.imprv\_id

select i.prop\_val\_yr, i.sup\_num, i.sale\_id, i.prop\_id, i.imprv\_id, max(i.imprv\_det\_id) as imprv\_det\_id

into #tmpDetail -- Highest valued improvement detail (single) on main areas (by year, prop\_id, etc)

from #tmpMainArea as i with (nolock)

join #tmpImpDetVal as v

on i.prop\_val\_yr = v.prop\_val\_yr

and i.sup\_num = v.sup\_num

and i.sale\_id = v.sale\_id

and i.prop\_id = v.prop\_id

and i.imprv\_id = v.imprv\_id

and i.imprv\_det\_val = v.imprv\_det\_val

group by i.prop\_val\_yr, i.sup\_num, i.sale\_id, i.prop\_id, i.imprv\_id

if @all\_imprv\_features = 0

begin

insert \_clientdb\_improvement\_features

(prop\_id, prop\_val\_yr, imprv\_id, imprv\_det\_id, imprv\_attr\_desc, i\_attr\_val\_cd)

select i.prop\_id, i.prop\_val\_yr, i.imprv\_id, i.imprv\_det\_id, attr.imprv\_attr\_desc, ia.i\_attr\_val\_cd

-- i.imprv\_det\_val, i.\*

from #tmpImprvDetail as i with (nolock)

join

#tmpDetail d

on i.prop\_val\_yr = d.prop\_val\_yr

and i.sup\_num = d.sup\_num

and i.sale\_id = d.sale\_id

and i.prop\_id = d.prop\_id

and i.imprv\_id = d.imprv\_id

and i.imprv\_det\_id = d.imprv\_det\_id

join imprv\_attr as ia with (nolock)

on ia.prop\_id = i.prop\_id

and ia.prop\_val\_yr = i.prop\_val\_yr

and ia.sup\_num = i.sup\_num

and ia.sale\_id = i.sale\_id

and ia.imprv\_id = i.imprv\_id

and ia.imprv\_det\_id = i.imprv\_det\_id

join attribute as attr with (nolock)

on ia.i\_attr\_val\_id = attr.imprv\_attr\_id

and attr.imprv\_attr\_desc is not null

and attr.web\_export = 1

end

else

begin

insert \_clientdb\_improvement\_features

(prop\_id, prop\_val\_yr, imprv\_id, imprv\_det\_id, imprv\_attr\_desc, i\_attr\_val\_cd)

select i.prop\_id, i.prop\_val\_yr, i.imprv\_id, i.imprv\_det\_id, attr.imprv\_attr\_desc, ia.i\_attr\_val\_cd

-- i.imprv\_det\_val, i.\*

from #tmpImprvDetail as i with (nolock)

join imprv\_attr as ia with (nolock)

on ia.prop\_val\_yr = i.prop\_val\_yr

and ia.sup\_num = i.sup\_num

and ia.sale\_id = i.sale\_id

and ia.prop\_id = i.prop\_id

and ia.imprv\_id = i.imprv\_id

and ia.imprv\_det\_id = i.imprv\_det\_id

join attribute as attr with (nolock)

on ia.i\_attr\_val\_id = attr.imprv\_attr\_id

and attr.imprv\_attr\_desc is not null

and attr.web\_export = 1

end

drop table #tmpMainArea

drop table #tmpImprvDetail

drop table #tmpImpDetVal

drop table #tmpDetail

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print ' Done at ' + convert(varchar(30), getdate(), 109)

--Improvement Detail Sketch - Export Info

print 'Exporting Sketch info... at ' + convert(varchar(30), getdate(), 109)

set @status = 'Sketch'

create table \_clientdb\_imprv\_det\_sketch

(

prop\_id int not null,

prop\_val\_yr numeric(4,0) not null,

imprv\_det\_type\_cd varchar(10) null,

imprv\_det\_typ\_desc varchar(50) null,

imprv\_det\_area numeric(18,1) null,

sketch\_cmds varchar(1800) null,

living\_area numeric(18,1) null

)

insert \_clientdb\_imprv\_det\_sketch

(prop\_id, prop\_val\_yr, imprv\_det\_type\_cd, imprv\_det\_typ\_desc, imprv\_det\_area,

sketch\_cmds, living\_area)

select

i.prop\_id,

i.prop\_val\_yr,

i.imprv\_det\_type\_cd,

idt.imprv\_det\_typ\_desc,

i.imprv\_det\_area,

i.sketch\_cmds,

case when isnull(main\_area, 'F') = 'T' then i.imprv\_det\_area else 0 end as living\_area

from imprv\_detail as i with (nolock)

join imprv\_det\_type as idt with (nolock)

on i.imprv\_det\_type\_cd = idt.imprv\_det\_type\_cd

join #layer\_assoc as psa with (nolock)

on i.prop\_val\_yr = psa.owner\_tax\_yr

and i.sup\_num = psa.sup\_num

and i.prop\_id = psa.prop\_id

join \_clientdb\_pacs\_year as y with (nolock)

on i.prop\_val\_yr = y.tax\_yr

and i.sale\_id = 0

and i.imprv\_id =

(

select min(imprv\_id)

from imprv with (nolock)

where prop\_val\_yr = psa.owner\_tax\_yr

and sup\_num = psa.sup\_num

and sale\_id = 0

and prop\_id = psa.prop\_id

)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

-------------------------

if @region = @washington

begin

set @status = '\_clientdb\_property\_sketch'

set @sql = '

create table '+ @input\_database\_name + '.dbo.\_clientdb\_property\_sketch

(

id int identity(1,1) not null,

prop\_id int not null,

prop\_val\_yr numeric(4,0) not null,

imprv\_id int not null,

image\_path varchar(255) null

)

insert into ' + @input\_database\_name + '.dbo.\_clientdb\_property\_sketch

(prop\_id, prop\_val\_yr, imprv\_id, image\_path)

select img.ref\_id as prop\_id, img.ref\_year as prop\_val\_yr, img.ref\_id1 as imprv\_id,

location as image\_path

from pacs\_image img with (nolock)

join ( select min(i.imprv\_id) minID, i.prop\_id, i.prop\_val\_yr, i.sup\_num, i.sale\_id

from imprv i with (nolock)

join #layer\_assoc la with (nolock)

on i.prop\_val\_yr = la.owner\_tax\_yr

and i.sup\_num = la.sup\_num

and i.prop\_id = la.prop\_id

group by i.prop\_id, i.prop\_val\_yr, i.sup\_num, i.sale\_id ) imp on

img.ref\_year = imp.prop\_val\_yr

and img.ref\_id2 = imp.sup\_num

and img.ref\_id = imp.prop\_id

and img.ref\_id3 = imp.sale\_id

and img.ref\_id1 = imp.minID

where img.ref\_type = ''SKTCH''

and img.image\_type = ''SKETCH\_LG''

--sale\_id

and img.ref\_id3 = 0'

exec (@sql)

set @sql = 'use ' + @input\_database\_name + ' ;checkpoint'

exec (@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = 'Property Image'

set @sql = '

create table '+ @input\_database\_name + '.dbo.\_clientdb\_property\_image

(

id int identity(1,1) not null,

prop\_id int not null,

year numeric(4, 0) not null,

image\_path varchar(255) null,

image\_nm varchar(64) null,

image\_type char(10) null,

sub\_type char(10) null,

rec\_type char(10) null,

comment varchar(255) null

)

insert into ' + @input\_database\_name + '.dbo.\_clientdb\_property\_image

(prop\_id, image\_path, image\_nm, year, image\_type, sub\_type, rec\_type, comment)

select img.ref\_id as prop\_id, location as image\_path, img.image\_nm as image\_nm,

img.ref\_year as year, img.image\_type as image\_type, img.sub\_type as sub\_type,

img.rec\_type as rec\_type, img.comment as comment

from pacs\_image img with (nolock)

join sub\_type as st with (nolock)

on img.image\_type = st.image\_type

and img.rec\_type = st.rect\_type

and img.sub\_type = st.sub\_type

where img.ref\_type in (''P'' , ''PP'' , ''PI'')

and isNull(st.allow\_website\_images, 0) = 1

'

exec (@sql)

set @sql = 'use ' + @input\_database\_name + ' ;checkpoint'

exec (@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = '\_clientdb\_payments'

set @sql = '

create table '+ @input\_database\_name + '.dbo.\_clientdb\_payments

(

prop\_id int not null,

year numeric(4, 0) not null,

statement\_id int not null,

paid bit not null default (0),

primary key clustered (prop\_id, year, statement\_id)

with fillfactor = 100

)'

exec (@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

end

print ' Done at ' + convert(varchar(30), getdate(), 109)

--Land Detail - Export Info

print 'Exporting Land Detail... at ' + convert(varchar(30), getdate(), 109)

set @status = 'Land Detail'

set @sql = 'select

l.prop\_id,

l.prop\_val\_yr,

l.land\_seg\_id,

l.land\_type\_cd,

lt.land\_type\_desc,

l.size\_acres,

l.size\_square\_feet,

l.effective\_front,

l.effective\_depth,

isnull(l.land\_seg\_mkt\_val, 0) as land\_seg\_mkt\_val,

isnull(l.ag\_val, 0) as ag\_val,

l.num\_lots,

''T'' as show\_values

into \_clientdb\_land\_detail from land\_detail as l with (nolock)

join #layer\_assoc as psa with (nolock)

on l.prop\_val\_yr = psa.owner\_tax\_yr

and l.sup\_num = psa.sup\_num

and l.prop\_id = psa.prop\_id

join land\_type as lt with (nolock)

on l.land\_type\_cd = lt.land\_type\_cd

join \_clientdb\_pacs\_year as y with (nolock)

on l.prop\_val\_yr = y.tax\_yr

where l.sale\_id = 0'

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print ' Done at ' + convert(varchar(30), getdate(), 109)

--Abstract/Subdv/MH - Export Info

print 'Exporting Abstract/Subdivision info... at ' + convert(varchar(30), getdate(), 109)

set @status = 'Abstract/Subdivision'

set @sql = 'select abs\_subdv\_ind, abs\_subdv\_cd, abs\_subdv\_desc into ' + @input\_database\_name +

'.dbo.\_clientdb\_abs\_subdv from abs\_subdv with (nolock)

where abs\_subdv\_yr =

(

select max(abs\_subdv2.abs\_subdv\_yr)

from abs\_subdv as abs\_subdv2 with (nolock)

where abs\_subdv.abs\_subdv\_ind = abs\_subdv2.abs\_subdv\_ind

)

order by abs\_subdv\_ind, abs\_subdv\_cd'

exec(@sql)

set @sql = 'use ' + @input\_database\_name + ' ;checkpoint'

exec (@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print ' Done at ' + convert(varchar(30), getdate(), 109)

print 'Exporting Neighborhood Info... at ' + convert(varchar(30), getdate(), 109)

set @status = 'Neighborhood'

set @sql = 'select hood\_cd, hood\_name into ' + @input\_database\_name +

'.dbo.\_clientdb\_neighborhood from neighborhood with (nolock)

where hood\_yr =

(

select max(n.hood\_yr)

from neighborhood as n with (nolock)

)

order by hood\_cd'

exec(@sql)

set @sql = 'use ' + @input\_database\_name + ' ;checkpoint'

exec (@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print ' Done at ' + convert(varchar(30), getdate(), 109)

-- Property info

print 'Exporting Property Info... at ' + convert(varchar(30), getdate(), 109)

set @status = 'Exporting Property Info - Improvement Adj'

create table \_clientdb\_property

(

prop\_id int not null,

prop\_val\_yr numeric(4,0) not null,

geo\_id varchar(50) null,

prop\_type\_cd varchar(5) null,

prop\_type\_desc varchar(50) null,

dba\_name varchar(50) null,

legal\_desc varchar(255) null,

appraised\_val numeric(14,0) null,

abs\_subdv\_cd varchar(50) null,

mapsco varchar(20) null,

map\_id varchar(20) null,

udi\_parent\_prop\_id int null,

agent\_cd varchar(10) null,

situs\_display varchar(255) null,

situs\_num varchar(15) null,

situs\_street varchar(50) null,

street\_name varchar(75) null,

situs\_city varchar(30) null,

hood\_cd varchar(10) null,

hood\_name varchar(100) null,

owner\_name varchar(70) null,

addr\_line1 varchar(60) null,

addr\_line2 varchar(60) null,

addr\_line3 varchar(60) null,

addr\_city varchar(50) null,

addr\_state varchar(50) null,

addr\_zip varchar(10) null,

country\_cd varchar(5) null,

owner\_id int null,

pct\_ownership numeric(13,10) null,

udi\_child\_prop\_id int null,

percent\_type varchar(5) null,

exemptions varchar(100) null,

state\_cd varchar(10) null,

jurisdictions varchar(100) null,

image\_path varchar(255) null,

show\_values varchar(1) null,

tax\_area\_id int null,

tax\_area varchar(300),

dor\_use\_code varchar(10),

open\_space varchar(1) default 'N',

dfl varchar(1) default 'N',

historic varchar(1) default 'N',

remodel varchar(1) default 'N',

multi\_fam varchar(1) default 'N',

township\_code varchar(20) null,

range\_code varchar(20) null,

township\_section varchar(50) null,

legal\_acreage numeric(14,4) null,

non\_taxed\_mkt\_val numeric (14,0) null,

is\_leased\_land\_property bit not null,

property\_use\_cd varchar(10) null,

secondary\_use\_cd varchar(10) null

)

-- Load imprv\_adj and land\_detail for use in sub queries

select distinct ia.prop\_val\_yr, ia.prop\_id, iat.imprv\_adj\_type\_patype

into #imprv\_adj from imprv\_adj ia with (nolock)

join imprv\_adj\_type iat with (nolock)

on ia.prop\_val\_yr = iat.imprv\_adj\_type\_year

and ia.imprv\_adj\_type\_cd = iat.imprv\_adj\_type\_cd

join \_clientdb\_pacs\_year as y with (nolock)

on ia.prop\_val\_yr = y.tax\_yr

join #layer\_assoc as psa with (nolock)

on ia.prop\_val\_yr = psa.owner\_tax\_yr

and ia.prop\_id = psa.prop\_id

and ia.sup\_num = psa.sup\_num

where ia.sale\_id = 0

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = 'Exporting Property Info - #land\_detail'

select distinct ld.prop\_val\_yr, ld.prop\_id, au.dfl, au.timber, au.ag, au.osp

into #land\_detail

from land\_detail as ld with (nolock)

join \_clientdb\_pacs\_year as y with (nolock)

on ld.prop\_val\_yr = y.tax\_yr

join #layer\_assoc as psa with (nolock)

on ld.prop\_val\_yr = psa.owner\_tax\_yr

and ld.sup\_num = psa.sup\_num

and ld.prop\_id = psa.prop\_id

join ag\_use as au with (nolock)

on ld.ag\_use\_cd = au.ag\_use\_cd

where ld.sale\_id = 0

and ld.ag\_apply = 'T'

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = 'Exporting Property Info'

set @sql = '

declare @region varchar(5)

select @region = szConfigValue from core\_config with (nolock) where szConfigName = ''REGION''

if(@region is null) set @region = ''WA''

insert \_clientdb\_property

(prop\_id, prop\_val\_yr, geo\_id, prop\_type\_cd, prop\_type\_desc, dba\_name, legal\_desc,

appraised\_val, abs\_subdv\_cd, mapsco, map\_id, udi\_parent\_prop\_id,

situs\_display, situs\_num, situs\_street, street\_name, situs\_city,

hood\_cd, hood\_name, owner\_name, addr\_line1, addr\_line2, addr\_line3,

addr\_city, addr\_state, addr\_zip, country\_cd, owner\_id, pct\_ownership,

udi\_child\_prop\_id, percent\_type, exemptions, state\_cd, jurisdictions, image\_path,

dor\_use\_code, open\_space, dfl, historic, remodel, multi\_fam,

township\_code, range\_code, township\_section, legal\_acreage, non\_taxed\_mkt\_val, is\_leased\_land\_property,

property\_use\_cd, secondary\_use\_cd

)

select p.prop\_id,

pv.prop\_val\_yr,

p.geo\_id,

p.prop\_type\_cd,

pt.prop\_type\_desc,

p.dba\_name,

pv.legal\_desc,

isnull(pv.appraised\_val, 0) as appraised\_val,

case when p.prop\_type\_cd = ''MH'' then pv.mbl\_hm\_park else pv.abs\_subdv\_cd end,

pv.mapsco,

pv.map\_id,

pv.udi\_parent\_prop\_id,

s.situs\_display,

s.situs\_num,

s.situs\_street,

ltrim(rtrim(

(case

when isnull([situs\_street\_prefx],'''') = '''' then ''''

else rtrim(ltrim([situs\_street\_prefx]))+'' ''

end) +

(case

when [situs\_street] IS NULL then ''''

else rtrim(ltrim([situs\_street]))+'' ''

end) +

(case

when [situs\_street\_sufix] IS NULL then ''''

else rtrim(ltrim([situs\_street\_sufix]))

end)

))

as street\_name,

s.situs\_city,

n.hood\_cd,

n.hood\_name,

a.file\_as\_name as owner\_name,

ad.addr\_line1,

ad.addr\_line2,

ad.addr\_line3,

ad.addr\_city,

ad.addr\_state,

ad.addr\_zip,

ad.country\_cd,

'

if @use\_col\_owner\_id = 0

begin

set @sql2 = 'o.owner\_id,'

end

else

begin

set @sql2 = 'p.col\_owner\_id,'

end

set @sql = @sql + @sql2 + '

o.pct\_ownership,

o.udi\_child\_prop\_id,

o.percent\_type,

dbo.fn\_getExemptions(o.prop\_id, o.owner\_tax\_yr, o.sup\_num) as exemptions,

rtrim(pp.state\_cd) as state\_cd,

case

when @region = ''WA''

then null

else dbo.fn\_getEntities(o.prop\_id, o.owner\_tax\_yr, o.sup\_num)

end as jurisdictions,

pv.image\_path,

pu.dor\_use\_code,

open\_space =

case when exists (select prop\_id from #land\_detail as ld with (nolock)

where (ld.prop\_id = pv.prop\_id and ld.prop\_val\_yr = pv.prop\_val\_yr)

and (ld.timber = 1 or ld.ag = 1 or ld.osp = 1)

)

then ''Y''

else ''N''

end,

dfl =

case when exists (select prop\_id from #land\_detail as ld with (nolock)

where (ld.prop\_id = pv.prop\_id and ld.prop\_val\_yr = pv.prop\_val\_yr)

and ld.dfl = 1

)

then ''Y''

else ''N''

end,

historic =

case when exists (select prop\_id from #imprv\_adj as ia with (nolock)

where ia.prop\_id = pv.prop\_id and ia.prop\_val\_yr = pv.prop\_val\_yr

and ia.imprv\_adj\_type\_patype = 1)

then ''Y''

else ''N''

end,

remodel =

case

when exists (select prop\_id from #imprv\_adj as ia with (nolock)

where ia.prop\_id = pv.prop\_id and ia.prop\_val\_yr = pv.prop\_val\_yr

and ia.imprv\_adj\_type\_patype = 0)

then ''Y''

else ''N''

end,

multi\_fam =

case

when exists (select prop\_id from #imprv\_adj as ia with (nolock)

where ia.prop\_id = pv.prop\_id and ia.prop\_val\_yr = pv.prop\_val\_yr

and ia.imprv\_adj\_type\_patype = 3)

then ''Y''

else ''N''

end,

ts.township\_code,

rr.range\_code,

pv.township\_section,

pv.legal\_acreage,

pv.non\_taxed\_mkt\_val,

is\_leased\_land\_property =

case when isnull(pst.imp\_leased\_land, 0) = 1 and p.prop\_type\_cd = ''R''

then 1 else 0

end,

pv.property\_use\_cd,

pv.secondary\_use\_cd

from property\_val as pv with (nolock)

join property as p with (nolock)

on pv.prop\_id = p.prop\_id

join owner as o with (nolock)

on pv.prop\_val\_yr = o.owner\_tax\_yr

and pv.sup\_num = o.sup\_num

and pv.prop\_id = o.prop\_id

join #layer\_assoc as psa with (nolock)

on pv.prop\_val\_yr = psa.owner\_tax\_yr

and pv.sup\_num = psa.sup\_num

and pv.prop\_id = psa.prop\_id

join \_clientdb\_pacs\_year as y with (nolock)

on pv.prop\_val\_yr = y.tax\_yr

left outer join property\_profile as pp with (nolock)

on pv.prop\_val\_yr = pp.prop\_val\_yr

and pv.prop\_id = pp.prop\_id

'

if @use\_col\_owner\_id = 0

begin

set @sql2 = '

join account as a with (nolock)

on o.owner\_id = a.acct\_id

join address as ad with (nolock)

on o.owner\_id = ad.acct\_id

'

end

else

begin

set @sql2 = '

join account as a with (nolock)

on p.col\_owner\_id = a.acct\_id

join address as ad with (nolock)

on p.col\_owner\_id = ad.acct\_id

'

end

set @sql = @sql + @sql2 + '

and ad.primary\_addr = ''Y''

join property\_type as pt with (nolock)

on p.prop\_type\_cd = pt.prop\_type\_cd

join pacs\_year as py with (nolock)

on pv.prop\_val\_yr = py.tax\_yr

left outer join situs as s with (nolock)

on p.prop\_id = s.prop\_id

and s.primary\_situs = ''Y''

left outer join neighborhood as n with (nolock)

on pv.hood\_cd = n.hood\_cd

and pv.prop\_val\_yr = n.hood\_yr

left outer join property\_use as pu with (nolock)

on pv.property\_use\_cd = pu.property\_use\_cd

left join township as ts with (nolock)

on pv.township\_code = ts.township\_code

and pv.prop\_val\_yr = ts.township\_year

left join prop\_range rr with (nolock)

on pv.range\_code = rr.range\_code

and pv.prop\_val\_yr = rr.range\_year

left outer join property\_sub\_type pst with (nolock)

on pv.sub\_type = pst.property\_sub\_cd

where ((pv.prop\_inactive\_dt is null) or

(pv.collections\_only = 1 and ' + cast(@collections\_only as varchar) + ' = 1))

'

if @hide\_minerals = 1

begin

set @sql = @sql + '

and p.prop\_type\_cd <> ''MN''

'

end

exec(@sql)

drop table #imprv\_adj

drop table #land\_detail

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = 'Exporting Property Info - UDI Parents'

-- HS 50020 Kevin Lloyd - col\_owner\_id mod not required here, since this query deals with UDI parents.

set @status = 'UDI Parents'

set @sql = '

declare @region varchar(5)

select @region = szConfigValue from core\_config with (nolock) where szConfigName = ''REGION''

if(@region is null) set @region = ''WA''

insert \_clientdb\_property

(prop\_id, prop\_val\_yr, geo\_id, prop\_type\_cd, prop\_type\_desc, dba\_name, legal\_desc,

appraised\_val, abs\_subdv\_cd, mapsco, map\_id, udi\_parent\_prop\_id,

situs\_display, situs\_num, situs\_street, street\_name, situs\_city,

hood\_cd, hood\_name, owner\_name, addr\_line1, addr\_line2, addr\_line3,

addr\_city, addr\_state, addr\_zip, country\_cd, owner\_id, pct\_ownership,

udi\_child\_prop\_id, percent\_type, exemptions, state\_cd, jurisdictions, image\_path,

property\_use\_cd, secondary\_use\_cd

)

select p.prop\_id,

pv.prop\_val\_yr,

p.geo\_id,

p.prop\_type\_cd,

pt.prop\_type\_desc,

p.dba\_name,

pv.legal\_desc,

isnull(pv.appraised\_val, 0) as appraised\_val,

case when p.prop\_type\_cd = ''MH'' then pv.mbl\_hm\_park else pv.abs\_subdv\_cd end,

pv.mapsco,

pv.map\_id,

pv.udi\_parent\_prop\_id,

s.situs\_display,

s.situs\_num,

s.situs\_street,

ltrim(rtrim(

(case

when isnull([situs\_street\_prefx],'''') = '''' then ''''

else rtrim(ltrim([situs\_street\_prefx]))+'' ''

end) +

(case

when [situs\_street] IS NULL then ''''

else rtrim(ltrim([situs\_street]))+'' ''

end)+

(case

when [situs\_street\_sufix] IS NULL then ''''

else rtrim(ltrim([situs\_street\_sufix]))

end)

))

as street\_name,

s.situs\_city,

n.hood\_cd,

n.hood\_name,

a.file\_as\_name as owner\_name,

ad.addr\_line1,

ad.addr\_line2,

ad.addr\_line3,

ad.addr\_city,

ad.addr\_state,

ad.addr\_zip,

ad.country\_cd,

o.owner\_id,

o.pct\_ownership,

o.udi\_child\_prop\_id,

o.percent\_type,

dbo.fn\_getExemptions(o.prop\_id, o.owner\_tax\_yr, o.sup\_num) as exemptions,

rtrim(pp.state\_cd) as state\_cd,

case

when @region = ''WA''

then null

else dbo.fn\_getEntities(o.prop\_id, o.owner\_tax\_yr, o.sup\_num)

end as jurisdictions,

pv.image\_path,

pv.property\_use\_cd,

pv.secondary\_use\_cd

from property\_val as pv with (nolock)

join property as p with (nolock)

on pv.prop\_id = p.prop\_id

join owner as o with (nolock)

on pv.prop\_val\_yr = o.owner\_tax\_yr

and pv.sup\_num = o.sup\_num

and pv.prop\_id = o.prop\_id

join #layer\_assoc as psa with (nolock)

on pv.prop\_val\_yr = psa.owner\_tax\_yr

and pv.sup\_num = psa.sup\_num

and pv.prop\_id = psa.prop\_id

join \_clientdb\_pacs\_year as y with (nolock)

on pv.prop\_val\_yr = y.tax\_yr

left outer join property\_profile as pp with (nolock)

on pv.prop\_val\_yr = pp.prop\_val\_yr

and pv.prop\_id = pp.prop\_id

join account as a with (nolock)

on o.owner\_id = a.acct\_id

join address as ad with (nolock)

on o.owner\_id = ad.acct\_id

and ad.primary\_addr = ''Y''

join property\_type as pt with (nolock)

on p.prop\_type\_cd = pt.prop\_type\_cd

join pacs\_year as py with (nolock)

on pv.prop\_val\_yr = py.tax\_yr

left outer join situs as s with (nolock)

on p.prop\_id = s.prop\_id

and s.primary\_situs = ''Y''

left outer join neighborhood as n with (nolock)

on pv.hood\_cd = n.hood\_cd

and pv.prop\_val\_yr = n.hood\_yr

where udi\_parent = ''T''

'

if @hide\_minerals = 1

begin

set @sql = @sql + '

and p.prop\_type\_cd <> ''MN''

'

end

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = 'Tax Area Updates'

if @region = @washington

begin --Populate Tax Area Information

set @sql = '

update \_clientdb\_property set

tax\_area\_id = pta.tax\_area\_id,

tax\_area = ta.tax\_area\_number + '' - '' + ta.tax\_area\_description

from \_clientdb\_property as dbP with (nolock)

join property\_tax\_area pta with (nolock) on

dbP.prop\_val\_yr = pta.year

and dbP.prop\_id = pta.prop\_id

join tax\_area ta with (nolock) on

ta.tax\_area\_id = pta.tax\_area\_id

'

exec(@sql)

end

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print ' Agent Update... at ' + convert(varchar(30), getdate(), 109)

set @status = 'Agent'

-- do agents separately, subquery is too slow

update c

set agent\_cd = case when ag.agent\_cd is null then 'ID:' + cast (ag.agent\_id as varchar(7)) else ag.agent\_cd end

from \_clientdb\_property as c with(tablock)

join agent\_assoc as aa with (nolock)

on c.prop\_val\_yr = aa.owner\_tax\_yr

and c.prop\_id = aa.prop\_id

and c.owner\_id = aa.owner\_id

join agent as ag with (nolock)

on aa.agent\_id = ag.agent\_id

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print ' Done at ' + convert(varchar(30), getdate(), 109)

-- Roll Value history

print 'Exporting Roll History Values... at ' + convert(varchar(30), getdate(), 109)

set @status = 'Roll History'

if @region = @texas

begin

set @sql = '

select

pv.prop\_id,

pv.prop\_val\_yr,

isnull(pv.imprv\_hstd\_val,0) + isnull(pv.imprv\_non\_hstd\_val,0) as improvements,

isnull(pv.land\_hstd\_val,0) + isnull(pv.land\_non\_hstd\_val,0) + isnull(pv.ag\_market,0) + isnull(pv.timber\_market,0) as land\_market,

isnull(pv.ag\_use\_val,0) + isnull(pv.timber\_use,0) as ag\_valuation,

isnull(pv.appraised\_val, 0) as appraised\_val,

pv.ten\_percent\_cap,

isnull(pv.assessed\_val, 0) as assessed\_val,

''T'' as show\_values

into \_clientdb\_roll\_value\_history\_detail

from property\_val as pv with (nolock)

join #layer\_assoc as psa with (nolock)

on pv.prop\_val\_yr = psa.owner\_tax\_yr

and pv.sup\_num = psa.sup\_num

and pv.prop\_id = psa.prop\_id

join \_clientdb\_pacs\_year as y with (nolock)

on pv.prop\_val\_yr = y.tax\_yr

join \_clientdb\_property as pv2 with (nolock)

on pv2.prop\_id = pv.prop\_id

and pv2.prop\_val\_yr = pv.prop\_val\_yr

'

end

else --Washington

begin

set @sql = '

select

pv.prop\_id,

pv.prop\_val\_yr,

isnull(pv.imprv\_hstd\_val,0) + isnull(pv.imprv\_non\_hstd\_val,0) as improvements,

isnull(pv.land\_hstd\_val,0) + isnull(pv.land\_non\_hstd\_val,0) + isnull(pv.ag\_market,0) + isnull(pv.timber\_market,0)

+ isnull(pv.ag\_hs\_mkt\_val,0) + isnull(pv.timber\_hs\_mkt\_val,0) as land\_market,

isnull(pv.ag\_use\_val,0) + isnull(pv.timber\_use,0) + isnull(pv.ag\_hs\_use\_val,0) + isnull(pv.timber\_hs\_use\_val,0) as ag\_valuation,

isnull(wpov.appraised, 0) as appraised\_val,

pv.ten\_percent\_cap,

isNull(wpov.taxable\_classified, 0) + isNull(wpov.taxable\_non\_classified, 0) as assessed\_val,

''T'' as show\_values

into \_clientdb\_roll\_value\_history\_detail

from property\_val as pv with (nolock)

join #layer\_assoc as psa with (nolock)

on pv.prop\_val\_yr = psa.owner\_tax\_yr

and pv.sup\_num = psa.sup\_num

and pv.prop\_id = psa.prop\_id

join \_clientdb\_pacs\_year as y with (nolock)

on pv.prop\_val\_yr = y.tax\_yr

join \_clientdb\_property as pv2 with (nolock)

on pv2.prop\_id = pv.prop\_id

and pv2.prop\_val\_yr = pv.prop\_val\_yr

left join wash\_prop\_owner\_val wpov with (nolock)

on pv.prop\_val\_yr = wpov.year

and pv.sup\_num = wpov.sup\_num

and pv.prop\_id = wpov.prop\_id

'

end

exec(@sql) --\_clientdb\_roll\_value\_history\_detail

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print ' Done at ' + convert(varchar(30), getdate(), 109)

-- Taxing Jurisdiction detail

print 'Exporting Taxing Jurisdiction Detail... at ' + convert(varchar(30), getdate(), 109)

set @status = 'Taxing Jurisdiction'

create table \_clientdb\_taxing\_jurisdiction\_detail

(

prop\_id int not null,

owner\_prop\_id int not null,

sup\_yr numeric(4,0) not null,

owner\_id int not null,

entity\_id int null,

owner\_name varchar(70) null,

pct\_ownership numeric(13,10) null,

total\_market numeric(18,0) null,

total\_appraised\_val numeric(18,0) null,

total\_assessed\_val numeric(18,0) null,

entity\_cd varchar(5) null,

file\_as\_name varchar(70) null,

tax\_rate numeric(15,10) null,

appraised\_val numeric(18,0) null,

assessed\_val numeric(18,0) null,

taxable\_val numeric(18,0) null,

freeze\_ceiling numeric(14,2) null,

show\_values varchar(1) null,

homesite\_val numeric(18,0) null, --classified

nonhomesite\_val numeric(18,0) null, --non-classified

tax\_area\_id int null,

tax\_district\_id int,

levy\_rate numeric(13, 10),

levy\_cd varchar(10) null,

levy\_description varchar(50) null,

taxes numeric (14, 2) null,

taxes\_wo\_ex numeric (14, 2) null

)

-- HS 50020 Kevin Lloyd

if @region = @washington

begin

set @sql = '

insert \_clientdb\_taxing\_jurisdiction\_detail

(prop\_id, owner\_prop\_id, sup\_yr, owner\_id, owner\_name, pct\_ownership,

total\_market, total\_appraised\_val, total\_assessed\_val, appraised\_val, assessed\_val,

taxable\_val, homesite\_val, nonhomesite\_val, tax\_area\_id, tax\_district\_id, levy\_cd, levy\_description)

select o.prop\_id,

wpov.prop\_id as owner\_prop\_id,

wpov.[year],

wpov.owner\_id,

a.file\_as\_name as owner\_name,

o.pct\_ownership,

wpov.market as total\_market,

isnull(wpov.appraised,0) as total\_appraised\_val,

isnull(wpov.taxable\_classified, 0) + isnull(wpov.taxable\_non\_classified, 0) as total\_assessed\_val,

isnull(wpov.appraised,0) as appraised\_val,

isnull(wpov.taxable\_classified, 0) + isnull(wpov.taxable\_non\_classified, 0) as assessed\_val,

null,

isnull(wpov.land\_hstd\_val,0) + isnull(wpov.imprv\_hstd\_val,0) as homesite\_val,

isnull(wpov.land\_non\_hstd\_val,0) + isnull(wpov.imprv\_non\_hstd\_val,0) as nonhomesite\_val,

pta.tax\_area\_id,

tafa.tax\_district\_id,

tafa.levy\_cd,

levy.levy\_description

from wash\_prop\_owner\_val as wpov with (nolock)

join owner as o with (nolock)

on wpov.[year] = o.owner\_tax\_yr

and wpov.sup\_num = o.sup\_num

and wpov.prop\_id = isnull(o.udi\_child\_prop\_id, o.prop\_id)

and wpov.owner\_id = o.owner\_id

join #layer\_assoc as psa with (nolock)

on psa.owner\_tax\_yr = wpov.year

and psa.sup\_num = wpov.sup\_num

and psa.prop\_id = wpov.prop\_id

join property\_val as pv with (nolock)

on wpov.year = pv.prop\_val\_yr

and wpov.sup\_num = pv.sup\_num

and wpov.prop\_id = pv.prop\_id

join property\_tax\_area as pta with (nolock)

on wpov.year = pta.year and

wpov.sup\_num = pta.sup\_num and

wpov.prop\_id = pta.prop\_id

join tax\_area\_fund\_assoc as tafa with (nolock)

on pta.tax\_area\_id = tafa.tax\_area\_id

and pta.[year] = tafa.[year]

join levy with (nolock)

on tafa.year = levy.year and

tafa.tax\_district\_id = levy.tax\_district\_id

and tafa.levy\_cd = levy.levy\_cd

'

if @use\_col\_owner\_id = 0

begin

set @sql2 = '

join account as a with (nolock)

on o.owner\_id = a.acct\_id

join \_clientdb\_pacs\_year as y with (nolock)

on wpov.[year] = y.tax\_yr

'

end

else

begin

set @sql2 = '

join property as p with (nolock)

on wpov.prop\_id = p.prop\_id

join account as a with (nolock)

on p.col\_owner\_id = a.acct\_id

join pacs\_year as y with (nolock)

on wpov.[year] = y.tax\_yr

'

end

end

else -- Texas Region

begin

set @sql = '

insert \_clientdb\_taxing\_jurisdiction\_detail

(prop\_id, owner\_prop\_id, sup\_yr, owner\_id, entity\_id, owner\_name, pct\_ownership,

total\_market, total\_appraised\_val, total\_assessed\_val, appraised\_val, assessed\_val,

taxable\_val, freeze\_ceiling, homesite\_val, nonhomesite\_val)

select o.prop\_id,

poev.prop\_id as owner\_prop\_id,

poev.sup\_yr,

poev.owner\_id,

poev.entity\_id,

a.file\_as\_name as owner\_name,

o.pct\_ownership,

pv.market as total\_market,

isnull(pv.appraised\_val,0) as total\_appraised\_val,

isnull(pv.assessed\_val, 0) as total\_assessed\_val,

isnull(poev.appraised\_val,0) as appraised\_val,

isnull(poev.assessed\_val, 0) as assessed\_val,

isnull(poev.taxable\_val, 0) as taxable\_val,

poev.freeze\_ceiling,

isnull(poev.land\_hstd\_val,0) + isnull(poev.imprv\_hstd\_val,0) - isnull(poev.ten\_percent\_cap,0) as homesite\_val,

isnull(poev.land\_non\_hstd\_val,0) + isnull(poev.imprv\_non\_hstd\_val,0) as nonhomesite\_val

from prop\_owner\_entity\_val as poev with (nolock)

join owner as o with (nolock)

on poev.sup\_yr = o.owner\_tax\_yr

and poev.sup\_num = o.sup\_num

and poev.prop\_id = isnull(o.udi\_child\_prop\_id, o.prop\_id)

and poev.owner\_id = o.owner\_id

join #layer\_assoc as psa with (nolock)

on psa.owner\_tax\_yr = poev.sup\_yr

and psa.sup\_num = poev.sup\_num

and psa.prop\_id = poev.prop\_id

join property\_val as pv with (nolock)

on poev.sup\_yr = pv.prop\_val\_yr

and poev.sup\_num = pv.sup\_num

and poev.prop\_id = pv.prop\_id

'

if @use\_col\_owner\_id = 0

begin

set @sql2 = '

join account as a with (nolock)

on o.owner\_id = a.acct\_id

join \_clientdb\_pacs\_year as y with (nolock)

on poev.sup\_yr = y.tax\_yr

'

end

else

begin

set @sql2 = '

join property as p with (nolock)

on poev.prop\_id = p.prop\_id

join account as a with (nolock)

on p.col\_owner\_id = a.acct\_id

join pacs\_year as y with (nolock)

on poev.sup\_yr = y.tax\_yr

'

end

end

--Execute taxing jurisdiction query

set @sql = @sql + @sql2

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print ' Tax Rate Update... at ' + convert(varchar(30), getdate(), 109)

if @region = @washington

begin

set @status = 'Taxing Rate Update: Step 1'

create clustered index IDX\_\_clientdb\_taxing\_jurisdiction\_detail\_sup\_yr\_levy\_cd

on \_clientdb\_taxing\_jurisdiction\_detail (sup\_yr, prop\_id, tax\_district\_id, levy\_cd)

with fillfactor = 90

-- Use @sql so that a reference to the wash\_prop\_owner\_val\_tax\_vw does not break

set @sql = '

update ct

set levy\_rate = case when levy\_exemption is null or

levy\_exemption not in ( select exmpt\_type\_cd

from property\_exemption with (nolock)

where owner\_tax\_yr = ct.sup\_yr

and prop\_id = ct.prop\_id

and sup\_num = psa.sup\_num)

then tax.levy\_rate\_non\_classified

else tax.levy\_rate\_classified

end,

taxable\_val = tax.taxable,

taxes = tax.tax\_amt,

taxes\_wo\_ex = tax.tax\_wout\_ex\_amt

from \_clientdb\_taxing\_jurisdiction\_detail as ct with(tablock)

join #layer\_assoc as psa with (nolock)

on psa.owner\_tax\_yr = ct.sup\_yr

and psa.prop\_id = ct.prop\_id

join wash\_prop\_owner\_val\_tax\_vw as tax with (nolock)

on tax.year = ct.sup\_yr

and tax.prop\_id = ct.prop\_id

and tax.tax\_district\_id = ct.tax\_district\_id

and tax.levy\_cd = ct.levy\_cd

and tax.sup\_num = psa.sup\_num

'

exec(@sql)

-- rgoolsby removed update of file\_as\_name from previous statement for performane

declare @AcctFileName table

(acct\_id int NOT NULL,

file\_as\_name varchar(70) NOT NULL

)

insert into @AcctFileName(acct\_id,file\_as\_name)

select a.acct\_id,a.file\_as\_name

from account as a with(nolock)

inner join

( select distinct tax\_district\_id

from \_clientdb\_taxing\_jurisdiction\_detail

) as ct

on a.acct\_id = ct.tax\_district\_id

update ct

set file\_as\_name = a.file\_as\_name

from \_clientdb\_taxing\_jurisdiction\_detail as ct

inner join

@AcctFileName as a

on ct.tax\_district\_id = a.acct\_id

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = 'Taxing Rate Update: Step 2'

end

else --Texas

begin

set @status = 'Taxing Rate Update'

create nonclustered index IDX\_\_clientdb\_taxing\_jurisdiction\_detail\_sup\_yr\_entity\_id

on \_clientdb\_taxing\_jurisdiction\_detail (entity\_id, sup\_yr)

with fillfactor = 90

update ct

set entity\_cd = e.entity\_cd,

file\_as\_name = ae.file\_as\_name,

tax\_rate = isnull(tr.m\_n\_o\_tax\_pct, 0) + isnull(tr.i\_n\_s\_tax\_pct, 0) + isnull(tr.prot\_i\_n\_s\_tax\_pct,0)

from \_clientdb\_taxing\_jurisdiction\_detail as ct with(tablock)

join entity as e with (nolock)

on ct.entity\_id = e.entity\_id

join account as ae with (nolock)

on ct.entity\_id = ae.acct\_id

join tax\_rate as tr with (nolock)

on ct.entity\_id = tr.entity\_id

and ct.sup\_yr = tr.tax\_rate\_yr

end

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print ' Done at ' + convert(varchar(30), getdate(), 109)

-- Values Detail

print 'Exporting Values Detail... at ' + convert(varchar(30), getdate(), 109)

set @status = 'Values Update'

if @region = @texas

begin

select pv.prop\_id as prop\_id,

pv.prop\_val\_yr as prop\_val\_yr,

isnull(pv.imprv\_hstd\_val, 0) as imprv\_hstd\_val,

isnull(pv.imprv\_non\_hstd\_val, 0) as imprv\_non\_hstd\_val,

isnull(pv.land\_hstd\_val, 0) as land\_hstd\_val,

isnull(pv.land\_non\_hstd\_val, 0) as land\_non\_hstd\_val,

isnull(pv.ag\_use\_val, 0) as ag\_use\_val,

pv.timber\_use as timber\_use,

isnull(pv.ag\_market, 0) as ag\_market,

isnull(pv.timber\_market, 0) as timber\_market,

isnull(pv.market, 0) as market,

isnull(pv.appraised\_val, 0) as appraised\_val,

pv.ten\_percent\_cap as ten\_percent\_cap,

isnull(pv.assessed\_val, 0) as assessed\_val,

0 as current\_hs\_use\_val,

0 as current\_nhs\_use\_val,

0 as current\_hs\_mkt\_val,

0 as current\_nhs\_mkt\_val,

0 as productivity\_loss,

0 as exmpt\_value,

0 as snr\_exempt\_loss,

'T' as show\_values

into \_clientdb\_values\_detail

from property\_val as pv with (nolock)

join #layer\_assoc as psa with (nolock)

on pv.prop\_val\_yr = psa.owner\_tax\_yr

and pv.sup\_num = psa.sup\_num

and pv.prop\_id = psa.prop\_id

join \_clientdb\_pacs\_year as y with (nolock)

on pv.prop\_val\_yr = y.tax\_yr

end

else --Washington

begin

select pv.prop\_id as prop\_id,

pv.prop\_val\_yr as prop\_val\_yr,

isnull(pv.imprv\_hstd\_val, 0) as imprv\_hstd\_val,

isnull(pv.imprv\_non\_hstd\_val, 0) as imprv\_non\_hstd\_val,

isnull(pv.land\_hstd\_val, 0) as land\_hstd\_val,

isnull(pv.land\_non\_hstd\_val, 0) as land\_non\_hstd\_val,

isnull(pv.ag\_use\_val, 0) + isnull(pv.ag\_hs\_use\_val, 0) as ag\_use\_val,

isNull(pv.timber\_use, 0) + isNull(pv.timber\_hs\_use\_val, 0) as timber\_use,

isnull(pv.ag\_market, 0) + isnull(pv.ag\_hs\_mkt\_val, 0) as ag\_market,

isnull(pv.timber\_market, 0) + isNull(timber\_hs\_mkt\_val, 0) as timber\_market,

isnull(pv.market, 0) as market,

isnull(wpv.appraised\_classified, 0) + isnull(wpv.appraised\_non\_classified, 0) as appraised\_val,

pv.ten\_percent\_cap as ten\_percent\_cap,

isNull(wpov.taxable\_classified, 0) + isNull(wpov.taxable\_non\_classified, 0) as assessed\_val, --isnull(pv.assessed\_val, 0) as assessed\_val,

isNull(pv.ag\_hs\_use\_val, 0) + isNull(pv.timber\_hs\_use\_val, 0) as current\_hs\_use\_val,

isNull(pv.ag\_use\_val, 0) + isNull(pv.timber\_use, 0) as current\_nhs\_use\_val,

isNull(pv.ag\_hs\_mkt\_val, 0) + isNull(pv.timber\_hs\_mkt\_val, 0) as current\_hs\_mkt\_val,

isNull(pv.ag\_market, 0) + isNull(pv.timber\_market, 0) as current\_nhs\_mkt\_val,

isNull(pv.ag\_loss, 0) + isNull(pv.ag\_hs\_loss,0) +

isNull(pv.timber\_loss, 0) + isNull(pv.timber\_hs\_loss, 0) as productivity\_loss,

exmpt.exmpt\_value as exmpt\_value,

wpv.snr\_exempt\_loss as snr\_exempt\_loss,

isNull(wpv.appraised\_classified, 0) as appraised\_classified,

isNull(wpv.appraised\_non\_classified, 0) as appraised\_non\_classified,

isnull(pv.non\_taxed\_mkt\_val, 0) as non\_taxed\_mkt\_val,

'T' as show\_values

into \_clientdb\_values\_detail

from property\_val as pv with (nolock)

join #layer\_assoc as psa with (nolock)

on pv.prop\_val\_yr = psa.owner\_tax\_yr

and pv.sup\_num = psa.sup\_num

and pv.prop\_id = psa.prop\_id

join wash\_property\_val wpv with (nolock)

on pv.prop\_val\_yr = wpv.prop\_val\_yr

and pv.sup\_num = wpv.sup\_num

and pv.prop\_id = wpv.prop\_id

join \_clientdb\_pacs\_year as y with (nolock)

on pv.prop\_val\_yr = y.tax\_yr

--Non Senior Exemption Loss

left join ( select sum(exempt\_value) exmpt\_value, prop\_id, year, sup\_num

from wash\_prop\_owner\_exemption with (nolock)

where exmpt\_type\_cd <> 'SNR/DSBL'

group by prop\_id, year, sup\_num) as exmpt

on pv.prop\_val\_yr = exmpt.year

and pv.sup\_num = exmpt.sup\_num

and pv.prop\_id = exmpt.prop\_id

--Taxable Val

left join wash\_prop\_owner\_val wpov with (nolock)

on pv.prop\_val\_yr = wpov.year

and pv.sup\_num = wpov.sup\_num

and pv.prop\_id = wpov.prop\_id

end

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print ' Done at ' + convert(varchar(30), getdate(), 109)

set @status = 'Bills'

--Bills - Export Info

if @region = @washington

begin

if @include\_adjustments = 1 begin -- 'new' way

set @sql = '

if exists (select name from sysobjects WHERE id = OBJECT\_ID(''exportYears''))

begin

DROP TABLE exportYears

end

CREATE TABLE exportYears (prop\_id int, year numeric(4, 0))

CREATE UNIQUE CLUSTERED INDEX IX\_1 on exportYears (prop\_id, year)

insert into exportYears

select distinct prop\_id, year

from bill with (nolock)

left outer join payout\_agreement\_bill\_assoc as paba

on bill.bill\_id = paba.bill\_id

where year <= ' + cast((@sys\_tax\_year - @input\_min\_bill\_years) as varchar) + '

and ((current\_amount\_due <> amount\_paid)

or (paba.payout\_agreement\_id is not null))

and isNull(bill.is\_active, 0) = 1

insert into exportYears

select distinct vw.prop\_id, f.year

from fee\_property\_vw vw with (nolock)

join fee f with (nolock)

on f.fee\_id = vw.fee\_id

left join exportYears ex with (nolock)

on ex.prop\_id = vw.prop\_id

and ex.year = f.year

where f.year <= ' + cast((@sys\_tax\_year - @input\_min\_bill\_years) as varchar) + '

and current\_amount\_due <> amount\_paid

and isNull(ex.prop\_id, -1) = -1

select bill\_id ,b.prop\_id ,b.year ,sup\_num ,owner\_id ,initial\_amount\_due

,current\_amount\_due ,amount\_paid ,bill\_type ,effective\_due\_date ,earliest\_collection\_date

,statement\_id ,code ,is\_active ,last\_modified ,adj\_effective\_dt ,adj\_expiration\_dt

,comment ,payment\_status\_type\_cd ,created\_by\_type\_cd ,rollback\_id ,payment\_group\_id

,display\_year ,cnv\_xref ,is\_overpaid, case when current\_amount\_due > amount\_paid then 0 else 1 end as taxes\_paid

into ' + @input\_database\_name + '.dbo.bill

from bill as b with (nolock)

left join exportYears ex with (nolock)

on b.year = ex.year

and b.prop\_id = ex.prop\_id

where isNull(b.is\_active, 0) = 1

--1/26/11 TFS 18620 Added to prevent exporting bills where the statement\_id is null

and isNull(b.statement\_id, -1) > -1

--Is in the current or previous year OR

and (b.year > ' + cast((@sys\_tax\_year - @input\_min\_bill\_years) as varchar) + ' or isNull(ex.prop\_id, 0) > 0)

'

end

else -- 'old' way

set @sql = '

if exists (select name from sysobjects WHERE id = OBJECT\_ID(''exportYears''))

begin

DROP TABLE exportYears

end

CREATE TABLE exportYears (prop\_id int, year numeric(4, 0))

CREATE UNIQUE CLUSTERED INDEX IX\_1 on exportYears (prop\_id, year)

insert into exportYears

select distinct prop\_id, year

from bill with (nolock)

left outer join payout\_agreement\_bill\_assoc as paba

on bill.bill\_id = paba.bill\_id

where year <= ' + cast(@sys\_tax\_year as varchar) + ' - 2

and (current\_amount\_due - amount\_paid > 0

or paba.payout\_agreement\_id is not null)

and isNull(bill.is\_active, 0) = 1

insert into exportYears

select distinct vw.prop\_id, f.year

from fee\_property\_vw vw with (nolock)

join fee f with (nolock)

on f.fee\_id = vw.fee\_id

left join exportYears ex with (nolock)

on ex.prop\_id = vw.prop\_id

and ex.year = f.year

where f.year <= ' + cast(@sys\_tax\_year as varchar) + ' - 2

and current\_amount\_due - amount\_paid > 0

and isNull(ex.prop\_id, -1) = -1

select bill\_id ,b.prop\_id ,b.year ,sup\_num ,owner\_id ,initial\_amount\_due

,current\_amount\_due ,amount\_paid ,bill\_type ,effective\_due\_date ,earliest\_collection\_date

,statement\_id ,code ,is\_active ,last\_modified ,adj\_effective\_dt ,adj\_expiration\_dt

,comment ,payment\_status\_type\_cd ,created\_by\_type\_cd ,rollback\_id ,payment\_group\_id

,display\_year ,cnv\_xref ,is\_overpaid, case when current\_amount\_due > amount\_paid then 0 else 1 end as taxes\_paid

into ' + @input\_database\_name + '.dbo.bill

from bill as b with (nolock)

left join exportYears ex with (nolock)

on b.year = ex.year

and b.prop\_id = ex.prop\_id

where isNull(b.is\_active, 0) = 1

--1/26/11 TFS 18620 Added to prevent exporting bills where the statement\_id is null

and isNull(b.statement\_id, -1) > -1

--Is in the current or previous year OR

and (b.year >= ' + cast(@sys\_tax\_year as varchar) + ' - 1 or isNull(ex.prop\_id, 0) > 0)

'

end

else -- region = Texas

begin

CREATE TABLE [\_clientdb\_bill] (

[bill\_id] [int] NOT NULL ,

[sup\_tax\_yr] [numeric](4, 0) NOT NULL ,

[sup\_num] [int] NOT NULL ,

[entity\_id] [int] NOT NULL ,

[prop\_id] [int] NOT NULL ,

[owner\_id] [int] NOT NULL ,

[adjustment\_code] [varchar] (10) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[adj\_effective\_dt] [datetime] NULL ,

[adj\_expiration\_dt] [datetime] NULL ,

[adj\_comment] [varchar] (500) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[rollback\_id] [int] NULL ,

[coll\_status\_cd] [char] (5) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[bill\_type] [char] (5) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[effective\_due\_dt] [datetime] NULL ,

[bill\_m\_n\_o] [numeric](14, 2) NULL ,

[bill\_i\_n\_s] [numeric](14, 2) NULL ,

[bill\_prot\_i\_n\_s] [numeric](14, 2) NULL ,

[bill\_late\_ag\_penalty] [numeric](14, 2) NULL ,

[bill\_m\_n\_o\_pd] [numeric](14, 2) NULL ,

[bill\_i\_n\_s\_pd] [numeric](14, 2) NULL ,

[penalty\_m\_n\_o\_pd] [numeric](14, 2) NULL ,

[penalty\_i\_n\_s\_pd] [numeric](14, 2) NULL ,

[interest\_m\_n\_o\_pd] [numeric](14, 2) NULL ,

[interest\_i\_n\_s\_pd] [numeric](14, 2) NULL ,

[attorney\_fees\_pd] [numeric](14, 2) NULL ,

[bill\_assessed\_value] [numeric](14, 2) NULL ,

[bill\_taxable\_val] [numeric](14, 2) NULL ,

[stmnt\_id] [numeric](18, 0) NULL ,

[discount\_mno\_pd] [numeric](14, 2) NULL ,

[discount\_ins\_pd] [numeric](14, 2) NULL ,

[prev\_bill\_id] [int] NULL ,

[new\_bill\_id] [int] NULL ,

[create\_dt] [datetime] NULL ,

[ref\_id1] [varchar] (20) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[ref\_id2] [varchar] (20) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[ref\_id3] [varchar] (20) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[ref\_id4] [varchar] (20) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[ref\_id5] [varchar] (20) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[discount\_offered] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[levy\_group\_id] [int] NULL ,

[levy\_run\_id] [int] NULL ,

[active\_bill] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[q1\_amt] [numeric](14, 2) NULL ,

[q1\_paid] [numeric](14, 2) NULL ,

[q1\_due\_dt] [datetime] NULL ,

[q2\_amt] [numeric](14, 2) NULL ,

[q2\_paid] [numeric](14, 2) NULL ,

[q2\_due\_dt] [datetime] NULL ,

[q3\_amt] [numeric](14, 2) NULL ,

[q3\_paid] [numeric](14, 2) NULL ,

[q3\_due\_dt] [datetime] NULL ,

[q4\_amt] [numeric](14, 2) NULL ,

[q4\_paid] [numeric](14, 2) NULL ,

[q4\_due\_dt] [datetime] NULL ,

[q\_bill] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[q\_create\_dt] [datetime] NULL ,

[q\_remove\_dt] [datetime] NULL ,

[q\_created\_by] [int] NULL ,

[q\_removed\_by] [int] NULL ,

[bill\_adj\_m\_n\_o] [numeric](14, 2) NULL ,

[bill\_adj\_i\_n\_s] [numeric](14, 2) NULL ,

[refund\_m\_n\_o\_pd] [numeric](14, 2) NULL ,

[refund\_i\_n\_s\_pd] [numeric](14, 2) NULL ,

[refund\_pen\_m\_n\_o\_pd] [numeric](14, 2) NULL ,

[refund\_pen\_i\_n\_s\_pd] [numeric](14, 2) NULL ,

[refund\_int\_m\_n\_o\_pd] [numeric](14, 2) NULL ,

[refund\_int\_i\_n\_s\_pd] [numeric](14, 2) NULL ,

[refund\_atty\_fee\_pd] [numeric](14, 2) NULL ,

[underage\_mno\_pd] [numeric](14, 2) NULL ,

[underage\_ins\_pd] [numeric](14, 2) NULL ,

[overage\_mno\_pd] [numeric](14, 2) NULL ,

[overage\_ins\_pd] [numeric](14, 2) NULL ,

[refund\_disc\_mno\_pd] [numeric](14, 2) NULL ,

[refund\_disc\_ins\_pd] [numeric](14, 2) NULL ,

[ia\_id] [int] NULL ,

[pay\_type] [varchar] (5) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[pay1\_amt] [numeric](14, 2) NULL ,

[pay1\_paid] [numeric](14, 2) NULL ,

[pay1\_due\_dt] [datetime] NULL ,

[pay2\_amt] [numeric](14, 2) NULL ,

[pay2\_paid] [numeric](14, 2) NULL ,

[pay2\_due\_dt] [datetime] NULL ,

[pay3\_amt] [numeric](14, 2) NULL ,

[pay3\_paid] [numeric](14, 2) NULL ,

[pay3\_due\_dt] [datetime] NULL ,

[pay4\_amt] [numeric](14, 2) NULL ,

[pay4\_paid] [numeric](14, 2) NULL ,

[pay4\_due\_dt] [datetime] NULL ,

[pay\_created\_dt] [datetime] NULL ,

[pay\_removed\_dt] [datetime] NULL ,

[pay\_created\_by] [int] NULL ,

[pay\_removed\_by] [int] NULL ,

[refund\_underage\_mno\_pd] [numeric](14, 2) NULL ,

[refund\_underage\_ins\_pd] [numeric](14, 2) NULL ,

[refund\_overage\_mno\_pd] [numeric](14, 2) NULL ,

[refund\_overage\_ins\_pd] [numeric](14, 2) NULL ,

[taxes\_paid] [bit] NULL,

CONSTRAINT [CPK\_\_clientdb\_bill] PRIMARY KEY CLUSTERED

(

[bill\_id],

[sup\_tax\_yr],

[sup\_num],

[entity\_id],

[prop\_id],

[owner\_id]

) WITH FILLFACTOR = 90 ON [PRIMARY]

) ON [PRIMARY]

set @sql = '

insert \_clientdb\_bill

(bill\_id, sup\_tax\_yr, sup\_num, entity\_id, prop\_id, owner\_id, adjustment\_code, adj\_effective\_dt,

adj\_expiration\_dt, adj\_comment, rollback\_id, coll\_status\_cd, bill\_type, effective\_due\_dt,

bill\_m\_n\_o, bill\_i\_n\_s, bill\_prot\_i\_n\_s, bill\_late\_ag\_penalty, bill\_m\_n\_o\_pd, bill\_i\_n\_s\_pd,

penalty\_m\_n\_o\_pd, penalty\_i\_n\_s\_pd, interest\_m\_n\_o\_pd, interest\_i\_n\_s\_pd, attorney\_fees\_pd,

bill\_assessed\_value, bill\_taxable\_val, stmnt\_id, discount\_mno\_pd, discount\_ins\_pd,

prev\_bill\_id, new\_bill\_id, create\_dt, ref\_id1, ref\_id2, ref\_id3, ref\_id4, ref\_id5,

discount\_offered, levy\_group\_id, levy\_run\_id, active\_bill, q1\_amt, q1\_paid, q2\_amt,

q2\_paid, q2\_due\_dt, q3\_amt, q3\_paid, q3\_due\_dt, q4\_amt, q4\_paid, q4\_due\_dt,

q\_bill, q\_create\_dt, q\_remove\_dt, q\_created\_by, q\_removed\_by, bill\_adj\_m\_n\_o,

bill\_adj\_i\_n\_s, refund\_m\_n\_o\_pd, refund\_i\_n\_s\_pd, refund\_pen\_m\_n\_o\_pd, refund\_pen\_i\_n\_s\_pd,

refund\_int\_m\_n\_o\_pd, refund\_int\_i\_n\_s\_pd, refund\_atty\_fee\_pd, underage\_mno\_pd,

underage\_ins\_pd, overage\_mno\_pd, overage\_ins\_pd, refund\_disc\_mno\_pd, refund\_disc\_ins\_pd,

ia\_id, pay\_type, pay1\_amt, pay1\_paid, pay1\_due\_dt, pay2\_amt, pay2\_paid, pay2\_due\_dt,

pay3\_amt, pay3\_paid, pay3\_due\_dt, pay4\_amt, pay4\_paid, pay4\_due\_dt, pay\_created\_dt,

pay\_removed\_dt, pay\_created\_by, pay\_removed\_by,

refund\_underage\_mno\_pd, refund\_underage\_ins\_pd, refund\_overage\_mno\_pd, refund\_overage\_ins\_pd, taxes\_paid)

select b.bill\_id, b.sup\_tax\_yr, b.sup\_num, b.entity\_id, b.prop\_id, b.owner\_id, b.adjustment\_code,

b.adj\_effective\_dt, b.adj\_expiration\_dt, b.adj\_comment, b.rollback\_id, b.coll\_status\_cd,

b.bill\_type, b.effective\_due\_dt, b.bill\_m\_n\_o, b.bill\_i\_n\_s, b.bill\_prot\_i\_n\_s,

b.bill\_late\_ag\_penalty, b.bill\_m\_n\_o\_pd, b.bill\_i\_n\_s\_pd, b.penalty\_m\_n\_o\_pd, b.penalty\_i\_n\_s\_pd,

b.interest\_m\_n\_o\_pd, b.interest\_i\_n\_s\_pd, b.attorney\_fees\_pd, b.bill\_assessed\_value,

b.bill\_taxable\_val, b.stmnt\_id, b.discount\_mno\_pd, b.discount\_ins\_pd, b.prev\_bill\_id, b.new\_bill\_id,

b.create\_dt, b.ref\_id1, b.ref\_id2, b.ref\_id3, b.ref\_id4, b.ref\_id5, b.discount\_offered,

b.levy\_group\_id, b.levy\_run\_id, b.active\_bill, b.q1\_amt, b.q1\_paid, b.q2\_amt,

b.q2\_paid, b.q2\_due\_dt, b.q3\_amt, b.q3\_paid, b.q3\_due\_dt, b.q4\_amt, b.q4\_paid, b.q4\_due\_dt,

b.q\_bill, b.q\_create\_dt, b.q\_remove\_dt, b.q\_created\_by, b.q\_removed\_by, b.bill\_adj\_m\_n\_o,

b.bill\_adj\_i\_n\_s, b.refund\_m\_n\_o\_pd, b.refund\_i\_n\_s\_pd, b.refund\_pen\_m\_n\_o\_pd, b.refund\_pen\_i\_n\_s\_pd,

b.refund\_int\_m\_n\_o\_pd, b.refund\_int\_i\_n\_s\_pd, b.refund\_atty\_fee\_pd, b.underage\_mno\_pd,

b.underage\_ins\_pd, b.overage\_mno\_pd, b.overage\_ins\_pd, b.refund\_disc\_mno\_pd, b.refund\_disc\_ins\_pd,

b.ia\_id, b.pay\_type, b.pay1\_amt, b.pay1\_paid, b.pay1\_due\_dt, b.pay2\_amt, b.pay2\_paid, b.pay2\_due\_dt,

b.pay3\_amt, b.pay3\_paid, b.pay3\_due\_dt, b.pay4\_amt, b.pay4\_paid, b.pay4\_due\_dt, b.pay\_created\_dt,

b.pay\_removed\_dt, b.pay\_created\_by, b.pay\_removed\_by,

b.refund\_underage\_mno\_pd, b.refund\_underage\_ins\_pd, b.refund\_overage\_mno\_pd, b.refund\_overage\_ins\_pd, 0

from bill as b with (nolock)

-- HS 43670 Kevin Lloyd - Import all bills, not just those existing for years in pacs\_year

where isnull(b.active\_bill, ''T'') = ''T''

and b.coll\_status\_cd <> ''RS''

'

end

exec(@sql) --Populate Clientdb\_bill

set @sql = 'use ' + @input\_database\_name + ' ;checkpoint'

exec (@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print ' Bills Done at ' + convert(varchar(30), getdate(), 109)

set @status = 'Tax Updates'

if @region = @washington

begin

--these tables are needed for the tax due engine

if @include\_adjustments = 1 begin -- 'new' way

set @sql = '

select distinct bpd.\*

into ' + @input\_database\_name + '.dbo.[bill\_payments\_due]

from bill\_payments\_due as bpd with (nolock)

join bill as b with (nolock)

on bpd.bill\_id = b.bill\_id

left join exportYears ex with (nolock)

on ex.year = b.year

and ex.prop\_id = b.prop\_id

where isNull(b.is\_active, 0) = 1

--1/26/11 TFS 18620 Added to prevent exporting bills where the statement\_id is null

and isNull(b.statement\_id, -1) > -1

and (b.year > ' + cast((@sys\_tax\_year - @input\_min\_bill\_years) as varchar) + ' or isNull(ex.prop\_id, 0) > 0)

select distinct lb.\*

into ' + @input\_database\_name + '.dbo.[levy\_bill]

from levy\_bill as lb with (nolock)

join bill as b with (nolock)

on lb.bill\_id = b.bill\_id

left join exportYears ex with (nolock)

on ex.year = b.year

and ex.prop\_id = b.prop\_id

where isNull(b.is\_active, 0) = 1

--1/26/11 TFS 18620 Added to prevent exporting bills where the statement\_id is null

and isNull(b.statement\_id, -1) > -1

and (b.year > ' + cast((@sys\_tax\_year - @input\_min\_bill\_years) as varchar) + ' or isNull(ex.prop\_id, 0) > 0)

select distinct ab.\*

into ' + @input\_database\_name + '.dbo.[assessment\_bill]

from assessment\_bill as ab with (nolock)

join bill as b with (nolock)

on ab.bill\_id = b.bill\_id

left join exportYears ex with (nolock)

on ex.year = b.year

and ex.prop\_id = b.prop\_id

where isNull(b.is\_active, 0) = 1

--1/26/11 TFS 18620 Added to prevent exporting bills where the statement\_id is null

and isNull(b.statement\_id, -1) > -1

and (b.year > ' + cast((@sys\_tax\_year - @input\_min\_bill\_years) as varchar) + ' or isNull(ex.prop\_id, 0) > 0)

select distinct ct.\*

into ' + @input\_database\_name + '.dbo.[coll\_transaction]

from coll\_transaction as ct with (nolock)

join bill as b with (nolock)

on ct.trans\_group\_id = b.bill\_id

left join exportYears ex with (nolock)

on ex.year = b.year

and ex.prop\_id = b.prop\_id

where isNull(b.is\_active, 0) = 1

--1/26/11 TFS 18620 Added to prevent exporting bills where the statement\_id is null

and isNull(b.statement\_id, -1) > -1

and (b.year > ' + cast((@sys\_tax\_year - @input\_min\_bill\_years) as varchar) + ' or isNull(ex.prop\_id, 0) > 0)

union all

select ct.\*

from coll\_transaction as ct with (nolock)

join fee with (nolock)

on ct.trans\_group\_id = fee.fee\_id

where isNull(fee.is\_active, 0) = 1

select distinct fee.\*

into ' + @input\_database\_name + '.dbo.[fee]

from fee with (nolock)

join fee\_property\_vw fpv with (nolock)

on fee.fee\_id = fpv.fee\_id

left join exportYears ex with (nolock)

on ex.year = fee.year

and ex.prop\_id = fpv.prop\_id

where isNull(fee.is\_active, 0) = 1

and (fee.year > ' + cast((@sys\_tax\_year - @input\_min\_bill\_years) as varchar) + ' or isNull(ex.prop\_id, 0) > 0)

select distinct fpd.\*

into ' + @input\_database\_name + '.dbo.[fee\_payments\_due]

from fee\_payments\_due as fpd with (nolock)

join fee with (nolock)

on fpd.fee\_id = fee.fee\_id

join fee\_property\_vw fpv with (nolock)

on fee.fee\_id = fpv.fee\_id

left join exportYears ex with (nolock)

on ex.year = fee.year

and ex.prop\_id = fpv.prop\_id

where isNull(fee.is\_active, 0) = 1

and (fee.year > ' + cast((@sys\_tax\_year - @input\_min\_bill\_years) as varchar) + ' or isNull(ex.prop\_id, 0) > 0)

select distinct pta.\*

into ' + @input\_database\_name + '.dbo.[payment\_transaction\_assoc]

from [payment\_transaction\_assoc] as pta with (nolock)

join coll\_transaction as ct with (nolock)

on ct.transaction\_id = pta.transaction\_id

join bill as b with (nolock)

on ct.trans\_group\_id = b.bill\_id

left join exportYears ex with (nolock)

on ex.year = b.year

and ex.prop\_id = b.prop\_id

where isNull(b.is\_active, 0) = 1

and isNull(b.is\_overpaid, 0) = 1

--1/26/11 TFS 18620 Added to prevent exporting bills where the statement\_id is null

and isNull(b.statement\_id, -1) > -1

and (b.year > ' + cast((@sys\_tax\_year - @input\_min\_bill\_years) as varchar) + ' or isNull(ex.prop\_id, 0) > 0)

select distinct rta.\*

into ' + @input\_database\_name + '.dbo.[refund\_transaction\_assoc]

from [refund\_transaction\_assoc] as rta with (nolock)

join coll\_transaction as ct with (nolock)

on ct.transaction\_id = rta.transaction\_id

join bill as b with (nolock)

on ct.trans\_group\_id = b.bill\_id

left join exportYears ex with (nolock)

on ex.year = b.year

and ex.prop\_id = b.prop\_id

where isNull(b.is\_active, 0) = 1

--1/26/11 TFS 18620 Added to prevent exporting bills where the statement\_id is null

and isNull(b.statement\_id, -1) > -1

and (b.year > ' + cast((@sys\_tax\_year - @input\_min\_bill\_years) as varchar) + ' or isNull(ex.prop\_id, 0) > 0)

select distinct p.\*

into ' + @input\_database\_name + '.dbo.[payment]

from [payment] as p with (nolock)

join ' + @input\_database\_name + '.dbo.[payment\_transaction\_assoc] as pta with (nolock)

on p.payment\_id = pta.payment\_id

select distinct paba.\*

into ' + @input\_database\_name + '.dbo.[payout\_agreement\_bill\_assoc]

from payout\_agreement\_bill\_assoc as paba with (nolock)

join bill as b with (nolock)

on paba.bill\_id = b.bill\_id

left join exportYears ex with (nolock)

on ex.year = b.year

and ex.prop\_id = b.prop\_id

where isNull(b.is\_active, 0) = 1

--1/26/11 TFS 18620 Added to prevent exporting bills where the statement\_id is null

and isNull(b.statement\_id, -1) > -1

and (b.year > ' + cast((@sys\_tax\_year - @input\_min\_bill\_years) as varchar) + ' or isNull(ex.prop\_id, 0) > 0)

select distinct pafa.\*

into ' + @input\_database\_name + '.dbo.[payout\_agreement\_fee\_assoc]

from payout\_agreement\_fee\_assoc as pafa with (nolock)

join fee with (nolock)

on pafa.fee\_id = fee.fee\_id

join fee\_property\_vw fpv with (nolock)

on fee.fee\_id = fpv.fee\_id

left join exportYears ex with (nolock)

on ex.year = fee.year

and ex.prop\_id = fpv.prop\_id

where isNull(fee.is\_active, 0) = 1

and (fee.year > ' + cast((@sys\_tax\_year - @input\_min\_bill\_years) as varchar) + ' or isNull(ex.prop\_id, 0) > 0)

select distinct wts.\*

into ' + @input\_database\_name + '.dbo.[wa\_tax\_statement]

from wa\_tax\_statement as wts with (nolock)

left join exportYears ex with (nolock)

on ex.year = wts.year

and ex.prop\_id = wts.prop\_id

and isNull(wts.statement\_id, -1) > -1

where (wts.year > ' + cast((@sys\_tax\_year - @input\_min\_bill\_years) as varchar) + ' or isNull(ex.prop\_id, 0) > 0)

select \*

into ' + @input\_database\_name + '.dbo.[payout\_agreement]

from payout\_agreement with (nolock)

select \*

into ' + @input\_database\_name + '.dbo.[payout\_agreement\_schedule]

from payout\_agreement\_schedule with (nolock)

select \*

into ' + @input\_database\_name + '.dbo.[special\_assessment\_statement\_options]

from special\_assessment\_statement\_options with (nolock)

drop table exportYears

'

end

else -- 'old' way

set @sql = '

select distinct bpd.\*

into ' + @input\_database\_name + '.dbo.[bill\_payments\_due]

from bill\_payments\_due as bpd with (nolock)

join bill as b with (nolock)

on bpd.bill\_id = b.bill\_id

left join exportYears ex with (nolock)

on ex.year = b.year

and ex.prop\_id = b.prop\_id

where isNull(b.is\_active, 0) = 1

--1/26/11 TFS 18620 Added to prevent exporting bills where the statement\_id is null

and isNull(b.statement\_id, -1) > -1

and (b.year >= ' + cast(@sys\_tax\_year as varchar) + ' - 1 or isNull(ex.prop\_id, 0) > 0)

select distinct lb.\*

into ' + @input\_database\_name + '.dbo.[levy\_bill]

from levy\_bill as lb with (nolock)

join bill as b with (nolock)

on lb.bill\_id = b.bill\_id

left join exportYears ex with (nolock)

on ex.year = b.year

and ex.prop\_id = b.prop\_id

where isNull(b.is\_active, 0) = 1

--1/26/11 TFS 18620 Added to prevent exporting bills where the statement\_id is null

and isNull(b.statement\_id, -1) > -1

and (b.year >= ' + cast(@sys\_tax\_year as varchar) + ' - 1 or isNull(ex.prop\_id, 0) > 0)

select distinct ab.\*

into ' + @input\_database\_name + '.dbo.[assessment\_bill]

from assessment\_bill as ab with (nolock)

join bill as b with (nolock)

on ab.bill\_id = b.bill\_id

left join exportYears ex with (nolock)

on ex.year = b.year

and ex.prop\_id = b.prop\_id

where isNull(b.is\_active, 0) = 1

--1/26/11 TFS 18620 Added to prevent exporting bills where the statement\_id is null

and isNull(b.statement\_id, -1) > -1

and (b.year >= ' + cast(@sys\_tax\_year as varchar) + ' - 1 or isNull(ex.prop\_id, 0) > 0)

select distinct ct.\*

into ' + @input\_database\_name + '.dbo.[coll\_transaction]

from coll\_transaction as ct with (nolock)

join bill as b with (nolock)

on ct.trans\_group\_id = b.bill\_id

left join exportYears ex with (nolock)

on ex.year = b.year

and ex.prop\_id = b.prop\_id

where isNull(b.is\_active, 0) = 1

--1/26/11 TFS 18620 Added to prevent exporting bills where the statement\_id is null

and isNull(b.statement\_id, -1) > -1

and (b.year >= ' + cast(@sys\_tax\_year as varchar) + ' - 1 or isNull(ex.prop\_id, 0) > 0)

union all

select ct.\*

from coll\_transaction as ct with (nolock)

join fee with (nolock)

on ct.trans\_group\_id = fee.fee\_id

where isNull(fee.is\_active, 0) = 1

select distinct fee.\*

into ' + @input\_database\_name + '.dbo.[fee]

from fee with (nolock)

join fee\_property\_vw fpv with (nolock)

on fee.fee\_id = fpv.fee\_id

left join exportYears ex with (nolock)

on ex.year = fee.year

and ex.prop\_id = fpv.prop\_id

where isNull(fee.is\_active, 0) = 1

and (fee.year >= ' + cast(@sys\_tax\_year as varchar) + ' - 1 or isNull(ex.prop\_id, 0) > 0)

select distinct fpd.\*

into ' + @input\_database\_name + '.dbo.[fee\_payments\_due]

from fee\_payments\_due as fpd with (nolock)

join fee with (nolock)

on fpd.fee\_id = fee.fee\_id

join fee\_property\_vw fpv with (nolock)

on fee.fee\_id = fpv.fee\_id

left join exportYears ex with (nolock)

on ex.year = fee.year

and ex.prop\_id = fpv.prop\_id

where isNull(fee.is\_active, 0) = 1

and (fee.year >= ' + cast(@sys\_tax\_year as varchar) + ' - 1 or isNull(ex.prop\_id, 0) > 0)

select distinct pta.\*

into ' + @input\_database\_name + '.dbo.[payment\_transaction\_assoc]

from [payment\_transaction\_assoc] as pta with (nolock)

join coll\_transaction as ct with (nolock)

on ct.transaction\_id = pta.transaction\_id

join bill as b with (nolock)

on ct.trans\_group\_id = b.bill\_id

left join exportYears ex with (nolock)

on ex.year = b.year

and ex.prop\_id = b.prop\_id

where isNull(b.is\_active, 0) = 1

and isNull(b.is\_overpaid, 0) = 1

--1/26/11 TFS 18620 Added to prevent exporting bills where the statement\_id is null

and isNull(b.statement\_id, -1) > -1

and (b.year >= ' + cast(@sys\_tax\_year as varchar) + ' - 1 or isNull(ex.prop\_id, 0) > 0)

select distinct rta.\*

into ' + @input\_database\_name + '.dbo.[refund\_transaction\_assoc]

from [refund\_transaction\_assoc] as rta with (nolock)

join coll\_transaction as ct with (nolock)

on ct.transaction\_id = rta.transaction\_id

join bill as b with (nolock)

on ct.trans\_group\_id = b.bill\_id

left join exportYears ex with (nolock)

on ex.year = b.year

and ex.prop\_id = b.prop\_id

where isNull(b.is\_active, 0) = 1

--1/26/11 TFS 18620 Added to prevent exporting bills where the statement\_id is null

and isNull(b.statement\_id, -1) > -1

and (b.year >= ' + cast(@sys\_tax\_year as varchar) + ' - 1 or isNull(ex.prop\_id, 0) > 0)

select distinct p.\*

into ' + @input\_database\_name + '.dbo.[payment]

from [payment] as p with (nolock)

join ' + @input\_database\_name + '.dbo.[payment\_transaction\_assoc] as pta with (nolock)

on p.payment\_id = pta.payment\_id

select distinct paba.\*

into ' + @input\_database\_name + '.dbo.[payout\_agreement\_bill\_assoc]

from payout\_agreement\_bill\_assoc as paba with (nolock)

join bill as b with (nolock)

on paba.bill\_id = b.bill\_id

left join exportYears ex with (nolock)

on ex.year = b.year

and ex.prop\_id = b.prop\_id

where isNull(b.is\_active, 0) = 1

--1/26/11 TFS 18620 Added to prevent exporting bills where the statement\_id is null

and isNull(b.statement\_id, -1) > -1

and (b.year >= ' + cast(@sys\_tax\_year as varchar) + ' - 1 or isNull(ex.prop\_id, 0) > 0)

select distinct pafa.\*

into ' + @input\_database\_name + '.dbo.[payout\_agreement\_fee\_assoc]

from payout\_agreement\_fee\_assoc as pafa with (nolock)

join fee with (nolock)

on pafa.fee\_id = fee.fee\_id

join fee\_property\_vw fpv with (nolock)

on fee.fee\_id = fpv.fee\_id

left join exportYears ex with (nolock)

on ex.year = fee.year

and ex.prop\_id = fpv.prop\_id

where isNull(fee.is\_active, 0) = 1

and (fee.year >= ' + cast(@sys\_tax\_year as varchar) + ' - 1 or isNull(ex.prop\_id, 0) > 0)

select \*

into ' + @input\_database\_name + '.dbo.[payout\_agreement]

from payout\_agreement with (nolock)

select \*

into ' + @input\_database\_name + '.dbo.[payout\_agreement\_schedule]

from payout\_agreement\_schedule with (nolock)

select \*

into ' + @input\_database\_name + '.dbo.[special\_assessment\_statement\_options]

from special\_assessment\_statement\_options with (nolock)

drop table exportYears

'

exec (@sql)

set @sql = 'use ' + @input\_database\_name + ' ;checkpoint'

exec (@sql)

print ' Tax Due Done at ' + convert(varchar(30), getdate(), 109)

end

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

--Tax Rate - Export Info

CREATE TABLE [\_clientdb\_tax\_rate] (

[entity\_id] [int] NOT NULL ,

[tax\_rate\_yr] [numeric](4, 0) NOT NULL ,

[discount\_dt] [datetime] NULL ,

[late\_dt] [datetime] NULL ,

[attorney\_fee\_dt] [datetime] NULL ,

[bills\_created\_dt] [datetime] NULL ,

[m\_n\_o\_tax\_pct] [numeric](13, 10) NULL ,

[i\_n\_s\_tax\_pct] [numeric](13, 10) NULL ,

[prot\_i\_n\_s\_tax\_pct] [numeric](13, 10) NULL ,

[sales\_tax\_pct] [numeric](13, 10) NULL ,

[levy\_start\_rct\_num] [numeric](18, 0) NULL ,

[supp\_start\_rct\_num] [numeric](18, 0) NULL ,

[stmnt\_dt] [datetime] NULL ,

[collect\_for] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[appraise\_for] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[ready\_to\_certify] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[special\_inv\_entity] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[ready\_to\_create\_bill] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[PLUS\_1\_INT\_PCT] [numeric](13, 10) NULL ,

[PLUS\_1\_PENALTY\_PCT] [numeric](13, 10) NULL ,

[PLUS\_2\_INT\_PCT] [numeric](13, 10) NULL ,

[PLUS\_2\_PENALTY\_PCT] [numeric](13, 10) NULL ,

[PLUS\_3\_INT\_PCT] [numeric](13, 10) NULL ,

[PLUS\_3\_PENALTY\_PCT] [numeric](13, 10) NULL ,

[PLUS\_4\_INT\_PCT] [numeric](13, 10) NULL ,

[PLUS\_4\_PENALTY\_PCT] [numeric](13, 10) NULL ,

[PLUS\_5\_INT\_PCT] [numeric](13, 10) NULL ,

[PLUS\_5\_PENALTY\_PCT] [numeric](13, 10) NULL ,

[PLUS\_6\_INT\_PCT] [numeric](13, 10) NULL ,

[PLUS\_6\_PENALTY\_PCT] [numeric](13, 10) NULL ,

[PLUS\_7\_INT\_PCT] [numeric](13, 10) NULL ,

[PLUS\_7\_PENALTY\_PCT] [numeric](13, 10) NULL ,

[PLUS\_8\_INT\_PCT] [numeric](13, 10) NULL ,

[PLUS\_8\_PENALTY\_PCT] [numeric](13, 10) NULL ,

[PLUS\_9\_INT\_PCT] [numeric](13, 10) NULL ,

[PLUS\_9\_PENALTY\_PCT] [numeric](13, 10) NULL ,

[attorney\_fee\_pct] [numeric](4, 2) NULL ,

[effective\_due\_dt] [datetime] NULL ,

[collect\_option] [char] (5) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[weed\_control] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[weed\_control\_pct] [numeric](4, 2) NULL ,

[ptd\_option] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[apply\_bpp\_attorney\_fees] [bit] NOT NULL CONSTRAINT [DF\_\_clientdb\_tax\_rate\_\_apply\_\_\_74F30FE8] DEFAULT (0),

[bpp\_attorney\_fee\_dt] [datetime] NULL ,

CONSTRAINT [CPK\_\_clientdb\_tax\_rate] PRIMARY KEY CLUSTERED

(

[entity\_id],

[tax\_rate\_yr]

) WITH FILLFACTOR = 90 ON [PRIMARY] ,

) ON [PRIMARY]

print ' Tax Rate Done at ' + convert(varchar(30), getdate(), 109)

if @region = @texas

begin

set @status = 'Tax Rate (Entity)'

insert \_clientdb\_tax\_rate

(entity\_id, tax\_rate\_yr, discount\_dt, late\_dt, attorney\_fee\_dt, bills\_created\_dt, m\_n\_o\_tax\_pct,

i\_n\_s\_tax\_pct, prot\_i\_n\_s\_tax\_pct, sales\_tax\_pct, levy\_start\_rct\_num, supp\_start\_rct\_num,

stmnt\_dt, collect\_for, appraise\_for, ready\_to\_certify, special\_inv\_entity, ready\_to\_create\_bill,

PLUS\_1\_INT\_PCT, PLUS\_1\_PENALTY\_PCT, PLUS\_2\_INT\_PCT, PLUS\_2\_PENALTY\_PCT, PLUS\_3\_INT\_PCT,

PLUS\_3\_PENALTY\_PCT, PLUS\_4\_INT\_PCT, PLUS\_4\_PENALTY\_PCT, PLUS\_5\_INT\_PCT, PLUS\_5\_PENALTY\_PCT,

PLUS\_6\_INT\_PCT, PLUS\_6\_PENALTY\_PCT, PLUS\_7\_INT\_PCT, PLUS\_7\_PENALTY\_PCT, PLUS\_8\_INT\_PCT,

PLUS\_8\_PENALTY\_PCT, PLUS\_9\_INT\_PCT, PLUS\_9\_PENALTY\_PCT, attorney\_fee\_pct, effective\_due\_dt,

collect\_option, weed\_control, weed\_control\_pct, ptd\_option, apply\_bpp\_attorney\_fees,

bpp\_attorney\_fee\_dt)

select entity\_id, tax\_rate\_yr, discount\_dt, late\_dt, attorney\_fee\_dt, bills\_created\_dt, m\_n\_o\_tax\_pct,

i\_n\_s\_tax\_pct, prot\_i\_n\_s\_tax\_pct, sales\_tax\_pct, levy\_start\_rct\_num, supp\_start\_rct\_num,

stmnt\_dt, collect\_for, appraise\_for, ready\_to\_certify, special\_inv\_entity, ready\_to\_create\_bill,

PLUS\_1\_INT\_PCT, PLUS\_1\_PENALTY\_PCT, PLUS\_2\_INT\_PCT, PLUS\_2\_PENALTY\_PCT, PLUS\_3\_INT\_PCT,

PLUS\_3\_PENALTY\_PCT, PLUS\_4\_INT\_PCT, PLUS\_4\_PENALTY\_PCT, PLUS\_5\_INT\_PCT, PLUS\_5\_PENALTY\_PCT,

PLUS\_6\_INT\_PCT, PLUS\_6\_PENALTY\_PCT, PLUS\_7\_INT\_PCT, PLUS\_7\_PENALTY\_PCT, PLUS\_8\_INT\_PCT,

PLUS\_8\_PENALTY\_PCT, PLUS\_9\_INT\_PCT, PLUS\_9\_PENALTY\_PCT, attorney\_fee\_pct, effective\_due\_dt,

collect\_option, weed\_control, weed\_control\_pct, ptd\_option, apply\_bpp\_attorney\_fees,

bpp\_attorney\_fee\_dt

from tax\_rate with (nolock)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

end

--Refund Due Trans - Export Info

CREATE TABLE [\_clientdb\_refund\_due\_trans] (

[transaction\_id] [int] NOT NULL ,

[batch\_id] [int] NOT NULL ,

[bill\_id] [int] NOT NULL ,

[mno\_amt] [numeric](14, 2) NULL ,

[ins\_amt] [numeric](14, 2) NULL ,

[penalty\_mno\_amt] [numeric](14, 2) NULL ,

[penalty\_ins\_amt] [numeric](14, 2) NULL ,

[interest\_mno\_amt] [numeric](14, 2) NULL ,

[interest\_ins\_amt] [numeric](14, 2) NULL ,

[atty\_fee\_amt] [numeric](14, 2) NULL ,

[payment\_trans\_id] [int] NULL ,

[adjust\_id] [int] NULL,

[discount\_mno\_amt] [numeric](14, 2) NULL,

[discount\_ins\_amt] [numeric](14, 2) NULL,

[underage\_mno\_amt] [numeric](14, 2) NULL,

[underage\_ins\_amt] [numeric](14, 2) NULL,

[overage\_mno\_amt] [numeric](14, 2) NULL,

[overage\_ins\_amt] [numeric](14, 2) NULL,

CONSTRAINT [CPK\_\_clientdb\_refund\_due\_trans] PRIMARY KEY CLUSTERED

(

[transaction\_id],

[batch\_id],

[bill\_id]

) WITH FILLFACTOR = 100 ON [PRIMARY]

) ON [PRIMARY]

set @status = 'Refund'

if @region = @texas

begin

insert \_clientdb\_refund\_due\_trans

(transaction\_id, batch\_id, bill\_id, mno\_amt, ins\_amt, penalty\_mno\_amt, penalty\_ins\_amt,

interest\_mno\_amt, interest\_ins\_amt, atty\_fee\_amt, payment\_trans\_id,

adjust\_id, discount\_mno\_amt, discount\_ins\_amt, underage\_mno\_amt, underage\_ins\_amt, overage\_mno\_amt, overage\_ins\_amt)

select transaction\_id, batch\_id, bill\_id, mno\_amt, ins\_amt, penalty\_mno\_amt, penalty\_ins\_amt,

interest\_mno\_amt, interest\_ins\_amt, atty\_fee\_amt, payment\_trans\_id,

adjust\_id, discount\_mno\_amt, discount\_ins\_amt, underage\_mno\_amt, underage\_ins\_amt, overage\_mno\_amt, overage\_ins\_amt

from refund\_due\_trans with (nolock)

end

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

--Bill Adjustment Codes - Export Info

set @status = 'Bill Adj Codes'

CREATE TABLE [\_clientdb\_bill\_adjust\_code] (

[adjust\_cd] [varchar] (10) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NOT NULL ,

[adjust\_desc] [varchar] (50) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NOT NULL ,

[deferral\_cd] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[alert\_user] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[use\_penalty] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[penalty\_rate] [numeric](4, 0) NULL ,

[use\_interest] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[interest\_rate] [numeric](4, 0) NULL ,

[use\_attorney\_fee] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[attorney\_fee\_rate] [numeric](4, 0) NULL ,

[use\_range] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[begin\_range] [numeric](4, 0) NULL ,

[end\_range] [numeric](4, 0) NULL ,

[sys\_flag] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[judgement\_cd] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

CONSTRAINT [CPK\_\_clientdb\_bill\_adjust\_code] PRIMARY KEY CLUSTERED

(

[adjust\_cd]

) WITH FILLFACTOR = 100 ON [PRIMARY]

) ON [PRIMARY]

if @region = @texas

begin

set @sql = '

insert \_clientdb\_bill\_adjust\_code

(adjust\_cd, adjust\_desc, deferral\_cd, alert\_user, use\_penalty, penalty\_rate, use\_interest,

interest\_rate, use\_attorney\_fee, attorney\_fee\_rate, use\_range, begin\_range, end\_range,

sys\_flag, judgement\_cd)

select adjust\_cd, adjust\_desc, deferral\_cd, alert\_user, use\_penalty, penalty\_rate, use\_interest,

interest\_rate, use\_attorney\_fee, attorney\_fee\_rate, use\_range, begin\_range, end\_range,

sys\_flag, judgement\_cd

from bill\_adjust\_code with (nolock)

'

exec(@sql)

end

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

--Payment

set @status = 'Payment'

CREATE TABLE [\_clientdb\_payment] (

[payment\_id] [int] NOT NULL ,

[batch\_id] [int] NULL ,

[amt\_due] [numeric](14, 2) NULL ,

[check\_num] [varchar] (50) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[mo\_num] [varchar] (50) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[check\_amt] [numeric](14, 2) NULL ,

[cash\_amt] [numeric](14, 2) NULL ,

[mo\_amt] [numeric](14, 2) NULL ,

[payment\_type] [char] (5) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[payment\_code] [char] (5) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[rcpt\_num] [int] NULL ,

[payee\_id] [int] NULL ,

[operator\_id] [int] NULL ,

[post\_date] [datetime] NULL ,

[date\_paid] [datetime] NULL ,

[dl\_number] [varchar] (50) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[dl\_state] [varchar] (2) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[dl\_exp\_date] [datetime] NULL ,

[void\_payment] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[void\_date] [datetime] NULL ,

[void\_by\_id] [int] NULL ,

[void\_reason] [varchar] (255) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[void\_batch\_id] [int] NULL ,

[new\_payment\_id] [int] NULL ,

[prev\_payment\_id] [int] NULL ,

[paid\_by] [varchar] (50) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[cc\_type] [varchar] (5) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[cc\_amt] [numeric](14, 2) NULL ,

[cc\_fee] [numeric](14, 2) NULL ,

[cc\_last\_four\_digits] [varchar] (4) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[cc\_auth] [varchar] (20) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

CONSTRAINT [CPK\_\_clientdb\_payment] PRIMARY KEY CLUSTERED

(

[payment\_id]

) WITH FILLFACTOR = 90 ON [PRIMARY] ,

) ON [PRIMARY]

if @region = @texas

begin

set @sql = '

insert \_clientdb\_payment

(payment\_id, batch\_id, amt\_due, check\_num, mo\_num, check\_amt, cash\_amt, mo\_amt, payment\_type,

payment\_code, rcpt\_num, payee\_id, operator\_id, post\_date, date\_paid, dl\_number, dl\_state,

dl\_exp\_date, void\_payment, void\_date, void\_by\_id, void\_reason, void\_batch\_id, new\_payment\_id,

prev\_payment\_id, paid\_by, cc\_type, cc\_amt, cc\_fee, cc\_last\_four\_digits, cc\_auth)

select payment\_id, batch\_id, amt\_due, check\_num, mo\_num, check\_amt, cash\_amt, mo\_amt, payment\_type,

payment\_code, rcpt\_num, payee\_id, operator\_id, post\_date, date\_paid, dl\_number, dl\_state,

dl\_exp\_date, void\_payment, void\_date, void\_by\_id, void\_reason, void\_batch\_id, new\_payment\_id,

prev\_payment\_id, paid\_by, cc\_type, cc\_amt, cc\_fee, cc\_last\_four\_digits, cc\_auth

from payment with (nolock)

'

exec(@sql)

end

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

--Payment Trans

set @status = 'Payment Trans'

CREATE TABLE [\_clientdb\_payment\_trans] (

[transaction\_id] [int] NOT NULL ,

[payment\_id] [int] NOT NULL ,

[prop\_id] [int] NULL ,

[bill\_id] [int] NULL ,

[fee\_id] [int] NULL ,

[trans\_type] [varchar] (5) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[fee\_amt] [numeric](14, 2) NULL ,

[mno\_amt] [numeric](14, 2) NULL ,

[ins\_amt] [numeric](14, 2) NULL ,

[penalty\_mno\_amt] [numeric](14, 2) NULL ,

[penalty\_ins\_amt] [numeric](14, 2) NULL ,

[interest\_mno\_amt] [numeric](14, 2) NULL ,

[interest\_ins\_amt] [numeric](14, 2) NULL ,

[attorney\_fee\_amt] [numeric](14, 2) NULL ,

[q1\_amt] [numeric](14, 2) NULL ,

[q2\_amt] [numeric](14, 2) NULL ,

[q3\_amt] [numeric](14, 2) NULL ,

[q4\_amt] [numeric](14, 2) NULL ,

[mno\_due] [numeric](14, 2) NULL ,

[ins\_due] [numeric](14, 2) NULL ,

[penalty] [numeric](14, 2) NULL ,

[interest] [numeric](14, 2) NULL ,

[attorney\_fee] [numeric](14, 2) NULL ,

[fee\_due] [numeric](14, 2) NULL ,

[fiscal\_year] [varchar] (10) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[fiscal\_month] [int] NULL ,

[fiscal\_entity\_id] [int] NULL ,

[discount\_mno\_amt] [numeric](14, 2) NULL ,

[discount\_ins\_amt] [numeric](14, 2) NULL ,

[underage\_mno\_amt] [numeric](14, 2) NULL ,

[underage\_ins\_amt] [numeric](14, 2) NULL ,

[overage\_mno\_amt] [numeric](14, 2) NULL ,

[overage\_ins\_amt] [numeric](14, 2) NULL ,

[refund\_mno\_amt] [numeric](14, 2) NULL ,

[refund\_ins\_amt] [numeric](14, 2) NULL ,

[void\_trans] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[void\_date] [datetime] NULL ,

[void\_by\_id] [int] NULL ,

[void\_reason] [varchar] (255) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[void\_batch\_id] [int] NULL ,

[prev\_transaction\_id] [int] NULL ,

[prev\_payment\_id] [int] NULL ,

CONSTRAINT [CPK\_\_clientdb\_payment\_trans] PRIMARY KEY CLUSTERED

(

[transaction\_id],

[payment\_id]

) WITH FILLFACTOR = 90 ON [PRIMARY] ,

) ON [PRIMARY]

if @region = @texas

begin

set @sql = '

insert \_clientdb\_payment\_trans

(transaction\_id, payment\_id, prop\_id, bill\_id, fee\_id, trans\_type, fee\_amt, mno\_amt, ins\_amt,

penalty\_mno\_amt, penalty\_ins\_amt, interest\_mno\_amt, interest\_ins\_amt, attorney\_fee\_amt,

q1\_amt, q2\_amt, q3\_amt, q4\_amt, mno\_due, ins\_due, penalty, interest, attorney\_fee,

fee\_due, fiscal\_year, fiscal\_month, fiscal\_entity\_id, discount\_mno\_amt, discount\_ins\_amt,

underage\_mno\_amt, underage\_ins\_amt, overage\_mno\_amt, overage\_ins\_amt, refund\_mno\_amt,

refund\_ins\_amt, void\_trans, void\_date, void\_by\_id, void\_reason, void\_batch\_id,

prev\_transaction\_id, prev\_payment\_id)

select transaction\_id, payment\_id, prop\_id, bill\_id, fee\_id, trans\_type, fee\_amt, mno\_amt, ins\_amt,

penalty\_mno\_amt, penalty\_ins\_amt, interest\_mno\_amt, interest\_ins\_amt, attorney\_fee\_amt,

q1\_amt, q2\_amt, q3\_amt, q4\_amt, mno\_due, ins\_due, penalty, interest, attorney\_fee,

fee\_due, fiscal\_year, fiscal\_month, fiscal\_entity\_id, discount\_mno\_amt, discount\_ins\_amt,

underage\_mno\_amt, underage\_ins\_amt, overage\_mno\_amt, overage\_ins\_amt, refund\_mno\_amt,

refund\_ins\_amt, void\_trans, void\_date, void\_by\_id, void\_reason, void\_batch\_id,

prev\_transaction\_id, prev\_payment\_id

from payment\_trans with (nolock)'

exec(@sql)

end

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

--Entity Codes - Export Info

set @status = 'Entity'

CREATE TABLE [\_clientdb\_entity] (

[entity\_id] [int] NOT NULL ,

[entity\_cd] [char] (5) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NOT NULL ,

[entity\_type\_cd] [char] (5) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NOT NULL ,

[entity\_disb\_bal] [numeric](14, 2) NULL ,

[taxing\_unit\_num] [varchar] (50) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[mbl\_hm\_submission] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[freeports\_allowed] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[ptd\_multi\_unit] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[appr\_company\_entity\_cd] [char] (5) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[refund\_default\_flag] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[weed\_control] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[fiscal\_begin\_date] [datetime] NULL ,

[fiscal\_end\_date] [datetime] NULL ,

[fiscal\_year] [varchar] (10) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[county\_taxing\_unit\_ind] [varchar] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[collector\_id] [int] NULL ,

[rendition\_entity] [bit] NULL ,

[enable\_timber\_78] [bit] NULL CONSTRAINT [DF\_\_clientdb\_entity\_enable\_timber\_78] DEFAULT (0),

CONSTRAINT [CPK\_\_clientdb\_entity] PRIMARY KEY CLUSTERED

(

[entity\_id]

) WITH FILLFACTOR = 90 ON [PRIMARY] ,

) ON [PRIMARY]

if @region = @texas

begin

insert \_clientdb\_entity

(entity\_id, entity\_cd, entity\_type\_cd, entity\_disb\_bal, taxing\_unit\_num, mbl\_hm\_submission,

freeports\_allowed, ptd\_multi\_unit, appr\_company\_entity\_cd, refund\_default\_flag, weed\_control,

fiscal\_begin\_date, fiscal\_end\_date, fiscal\_year, county\_taxing\_unit\_ind, collector\_id,

rendition\_entity, enable\_timber\_78)

select entity\_id, entity\_cd, entity\_type\_cd, entity\_disb\_bal, taxing\_unit\_num, mbl\_hm\_submission,

freeports\_allowed, ptd\_multi\_unit, appr\_company\_entity\_cd, refund\_default\_flag, weed\_control,

fiscal\_begin\_date, fiscal\_end\_date, fiscal\_year, county\_taxing\_unit\_ind, collector\_id,

rendition\_entity, enable\_timber\_78

from entity with (nolock)

end

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

--Account - Export Info

CREATE TABLE [\_clientdb\_account] (

[acct\_id] [int] NOT NULL ,

[first\_name] [varchar] (30) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[last\_name] [varchar] (30) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[file\_as\_name] [varchar] (70) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[dl\_num] [varchar] (10) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[dl\_state] [char] (2) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[dl\_expir\_dt] [datetime] NULL ,

[merged\_acct\_id] [char] (5) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[acct\_create\_dt] [datetime] NULL ,

[opening\_balance] [money] NULL ,

[comment] [varchar] (2048) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[misc\_code] [varchar] (50) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[ref\_id1] [varchar] (50) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[source] [varchar] (30) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[ref\_acct\_id] [int] NULL ,

[confidential\_flag] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[confidential\_file\_as\_name] [varchar] (70) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[confidential\_first\_name] [varchar] (30) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[confidential\_last\_name] [varchar] (30) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[dist\_m\_n\_o] [int] NULL ,

[dist\_i\_n\_s] [int] NULL ,

[dist\_pi] [int] NULL ,

[dist\_atty\_fees] [int] NULL ,

[dist\_overages] [int] NULL ,

[dist\_tax\_cert\_fees] [int] NULL ,

[dist\_misc\_fees] [int] NULL ,

[dist\_vit] [int] NULL ,

[email\_addr] [varchar] (50) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[web\_addr] [varchar] (50) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[ftp\_addr] [varchar] (50) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[update\_dt] [datetime] NULL ,

[web\_suppression] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[appr\_company\_id] [int] NULL ,

CONSTRAINT [CPK\_\_clientdb\_account] PRIMARY KEY CLUSTERED

(

[acct\_id]

) WITH FILLFACTOR = 90 ON [PRIMARY]

) ON [PRIMARY]

print ' Begin Account ' + convert(varchar(30), getdate(), 109)

set @status = 'Account'

insert \_clientdb\_account

(acct\_id, first\_name, last\_name, file\_as\_name, dl\_num, dl\_state, dl\_expir\_dt, merged\_acct\_id,

acct\_create\_dt, opening\_balance, comment, misc\_code, ref\_id1, source, ref\_acct\_id,

confidential\_flag, confidential\_file\_as\_name, confidential\_first\_name, confidential\_last\_name,

dist\_m\_n\_o, dist\_i\_n\_s, dist\_pi, dist\_atty\_fees, dist\_overages, dist\_tax\_cert\_fees, dist\_vit,

email\_addr, web\_addr, ftp\_addr, update\_dt, web\_suppression, appr\_company\_id)

select acct\_id, first\_name, last\_name, file\_as\_name, dl\_num, dl\_state, dl\_expir\_dt, merged\_acct\_id,

acct\_create\_dt, opening\_balance, comment, misc\_code, ref\_id1, source, ref\_acct\_id,

confidential\_flag, confidential\_file\_as\_name, confidential\_first\_name, confidential\_last\_name,

dist\_m\_n\_o, dist\_i\_n\_s, dist\_pi, dist\_atty\_fees, dist\_overages, dist\_tax\_cert\_fees, dist\_vit,

email\_addr, web\_addr, ftp\_addr, update\_dt, web\_suppression, appr\_company\_id

from account with (nolock)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print 'Exporting Exemptions Info... at ' + convert(varchar(30), getdate(), 109)

set @status = '\_clientdb\_exmpt\_type'

set @sql = 'select exmpt\_type\_cd, exmpt\_desc into ' + @input\_database\_name +

'.dbo.\_clientdb\_exmpt\_type from exmpt\_type with (nolock) order by exmpt\_type\_cd'

exec(@sql)

set @sql = 'use ' + @input\_database\_name + ' ;checkpoint'

exec (@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print 'Exporting Improvement Detail Type Info... at ' + convert(varchar(30), getdate(), 109)

set @status = '\_clientdb\_imprv\_det\_type'

set @sql = 'select imprv\_det\_type\_cd, imprv\_det\_typ\_desc into ' + @input\_database\_name +

'.dbo.\_clientdb\_imprv\_det\_type from imprv\_det\_type with (nolock) where is\_permanent\_crop\_detail = 0

order by imprv\_det\_type\_cd'

exec(@sql)

set @sql = 'use ' + @input\_database\_name + ' ;checkpoint'

exec (@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print 'Exporting Improvement Detail Sub Class Info... at ' + convert(varchar(30), getdate(), 109)

set @status = 'imprv\_det\_sub\_class'

set @sql = 'select imprv\_det\_sub\_cls\_cd, imprv\_det\_sub\_cls\_desc into ' + @input\_database\_name +

'.dbo.\_clientdb\_imprv\_det\_sub\_class from imprv\_det\_sub\_class with (nolock) where ltrim(rtrim(imprv\_det\_sub\_cls\_cd)) not like ''\*'' and

is\_permanent\_crop\_detail = 0 order by imprv\_det\_sub\_cls\_cd'

exec(@sql)

set @sql = 'use ' + @input\_database\_name + ' ;checkpoint'

exec (@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print ' Done at ' + convert(varchar(30), getdate(), 109)

print 'Exporting Improv Detail Class Code Info... at ' + convert(varchar(30), getdate(), 109)

set @status = '\_clientdb\_imprv\_det\_class'

set @sql = 'select imprv\_det\_class\_cd, imprv\_det\_cls\_desc into ' + @input\_database\_name +

'.dbo.\_clientdb\_imprv\_det\_class from imprv\_det\_class with (nolock) where ltrim(rtrim(imprv\_det\_class\_cd)) not like ''\*'' and

is\_permanent\_crop\_detail = 0 order by imprv\_det\_class\_cd'

exec(@sql)

set @sql = 'use ' + @input\_database\_name + ' ;checkpoint'

exec (@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print ' Done at ' + convert(varchar(30), getdate(), 109)

/\*

\* This was done originally for Bexar. If there's a property

\* group code of X25.19A on a property, do NOT show values.

\* This is so values will show for properties that have had

\* their appraisal card printed.

\*/

set @status = 'UpdatePropertyAccessShowValues'

if exists(select id from syscomments where object\_name(id) like 'UpdatePropertyAccessShowValues')

begin

exec UpdatePropertyAccessShowValues

end

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

-- Start: Create all tables, views and procedures

/\*

\* Create middle tier tables

\*/

-- HS 60280 Kevin Lloyd

set @status = 'table\_cache\_status'

set @sql = 'use ' + @input\_database\_name + ' '

set @sql = @sql + 'create table table\_cache\_status

(

[szTableName] [varchar] (128) not null,

[lDummy] [int] not null,

[ts] [timestamp] not null,

CONSTRAINT [CPK\_table\_cache\_status] PRIMARY KEY CLUSTERED

(

[szTableName]

) ON [PRIMARY]

)'

exec(@sql)

set @sql = 'insert ' + @input\_database\_name + '.dbo.table\_cache\_status(szTableName,lDummy) select szTableName,lDummy from table\_cache\_status with (nolock)'

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = 'penalty\_and\_interest'

set @sql = 'select \* into ' + @input\_database\_name + '.dbo.penalty\_and\_interest from dbo.penalty\_and\_interest with (nolock)'

exec(@sql)

set @sql = 'use ' + @input\_database\_name + ' ;checkpoint'

exec (@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

--More middle-tier stuff

set @sql = @input\_database\_name + '.dbo.sp\_executesql N''

set ansi\_nulls on

set ansi\_padding on

set ansi\_warnings on

set arithabort on

set concat\_null\_yields\_null on

set quoted\_identifier on

set numeric\_roundabort off'''

exec(@sql)

set @status = '\_clientdb\_improvement\_features'

set @sql = @input\_database\_name + '.dbo.sp\_executesql N''

create table \_clientdb\_improvement\_features

(

prop\_id int not null,

prop\_val\_yr numeric(4,0) not null,

imprv\_id int not null,

imprv\_det\_id int null,

imprv\_attr\_desc varchar(50),

i\_attr\_val\_cd varchar(75)

) '''

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = '\_clientdb\_imprv\_det\_sketch'

set @sql = @input\_database\_name + '.dbo.sp\_executesql N''

create table \_clientdb\_imprv\_det\_sketch

(

prop\_id int not null,

prop\_val\_yr numeric(4,0) not null,

imprv\_det\_type\_cd varchar(10) null,

imprv\_det\_typ\_desc varchar(50) null,

imprv\_det\_area numeric(18,1) null,

sketch\_cmds varchar(1800) null,

living\_area numeric(18,1) null

) '''

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = '\_clientdb\_pacs\_year'

set @sql = @input\_database\_name + '.dbo.sp\_executesql N''

create table \_clientdb\_pacs\_year

(

tax\_yr numeric(4,0) not null,

certification\_dt datetime null,

prev\_reappraised\_yr numeric(4,0) null

) '''

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

-- HS 55281 Kevin Lloyd

set @status = '\_clientdb\_taxing\_jurisdiction\_detail'

set @sql = @input\_database\_name + '.dbo.sp\_executesql N''

create table \_clientdb\_taxing\_jurisdiction\_detail

(

prop\_id int not null,

owner\_prop\_id int not null,

sup\_yr numeric(4,0) not null,

owner\_id int not null,

entity\_id int null,

owner\_name varchar(70) null,

pct\_ownership numeric(13,10) null,

total\_market numeric(18,0) null,

total\_appraised\_val numeric(18,0) null,

total\_assessed\_val numeric(18,0) null,

entity\_cd varchar(5) null,

file\_as\_name varchar(70) null,

tax\_rate numeric(15,10) null,

appraised\_val numeric(18,0) null,

assessed\_val numeric(18,0) null,

taxable\_val numeric(18,0) null,

freeze\_ceiling numeric(14,2) null,

show\_values varchar(1) null,

homesite\_val numeric(18,0) null,

nonhomesite\_val numeric(18,0) null,

tax\_area\_id int null,

tax\_district\_id int,

levy\_rate numeric(13, 10),

levy\_cd varchar(10) null,

levy\_description varchar(50) null,

taxes numeric (14, 2) null,

taxes\_wo\_ex numeric (14, 2) null

) '''

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = '\_clientdb\_values\_detail'

set @sql = @input\_database\_name + '.dbo.sp\_executesql N''

create table \_clientdb\_values\_detail

(

prop\_id int not null,

prop\_val\_yr numeric(4,0) not null,

imprv\_hstd\_val numeric(14,0) null,

imprv\_non\_hstd\_val numeric(14,0) null,

land\_hstd\_val numeric(14,0) null,

land\_non\_hstd\_val numeric(14,0) null,

ag\_use\_val numeric(14,0) null,

timber\_use numeric(14,0) null,

ag\_market numeric(14,0) null,

timber\_market numeric(14,0) null,

market numeric(14,0) null,

appraised\_val numeric(14,0) null,

ten\_percent\_cap numeric(14,0) null,

assessed\_val numeric(14,0) null,

current\_hs\_use\_val numeric(14,0) null,

current\_nhs\_use\_val numeric(14,0) null,

current\_hs\_mkt\_val numeric(14,0) null,

current\_nhs\_mkt\_val numeric(14,0) null,

productivity\_loss numeric(14,0) null,

exmpt\_value numeric(14,0) null,

snr\_exempt\_loss numeric(14,0) null,

appraised\_classified numeric (14,0) null,

appraised\_non\_classified numeric (14,0) null,

non\_taxed\_mkt\_val numeric (14,0) null,

show\_values varchar(1) null

) '''

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = '[account]'

set @sql = @input\_database\_name + '.dbo.sp\_executesql N''

CREATE TABLE [account] (

[acct\_id] [int] NOT NULL ,

[first\_name] [varchar] (30) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[last\_name] [varchar] (30) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[file\_as\_name] [varchar] (70) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[dl\_num] [varchar] (10) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[dl\_state] [char] (2) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[dl\_expir\_dt] [datetime] NULL ,

[merged\_acct\_id] [char] (5) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[acct\_create\_dt] [datetime] NULL ,

[opening\_balance] [money] NULL ,

[comment] [varchar] (2048) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[misc\_code] [varchar] (50) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[ref\_id1] [varchar] (50) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[source] [varchar] (30) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[ref\_acct\_id] [int] NULL ,

[confidential\_flag] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[confidential\_file\_as\_name] [varchar] (70) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[confidential\_first\_name] [varchar] (30) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[confidential\_last\_name] [varchar] (30) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[dist\_m\_n\_o] [int] NULL ,

[dist\_i\_n\_s] [int] NULL ,

[dist\_pi] [int] NULL ,

[dist\_atty\_fees] [int] NULL ,

[dist\_overages] [int] NULL ,

[dist\_tax\_cert\_fees] [int] NULL ,

[dist\_misc\_fees] [int] NULL ,

[dist\_vit] [int] NULL ,

[email\_addr] [varchar] (50) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[web\_addr] [varchar] (50) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[ftp\_addr] [varchar] (50) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[update\_dt] [datetime] NULL ,

[web\_suppression] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[appr\_company\_id] [int] NULL ,

CONSTRAINT [CPK\_\_account] PRIMARY KEY CLUSTERED

(

[acct\_id]

) WITH FILLFACTOR = 90 ON [PRIMARY]

) ON [PRIMARY] '''

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

if @region = @texas

begin

set @status = 'Texas [bill]'

set @sql = @input\_database\_name + '.dbo.sp\_executesql N''

CREATE TABLE [bill] (

[bill\_id] [int] NOT NULL ,

[sup\_tax\_yr] [numeric](4, 0) NOT NULL ,

[sup\_num] [int] NOT NULL ,

[entity\_id] [int] NOT NULL ,

[prop\_id] [int] NOT NULL ,

[owner\_id] [int] NOT NULL ,

[adjustment\_code] [varchar] (10) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[adj\_effective\_dt] [datetime] NULL ,

[adj\_expiration\_dt] [datetime] NULL ,

[adj\_comment] [varchar] (500) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[rollback\_id] [int] NULL ,

[coll\_status\_cd] [char] (5) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[bill\_type] [char] (5) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[effective\_due\_dt] [datetime] NULL ,

[bill\_m\_n\_o] [numeric](14, 2) NULL ,

[bill\_i\_n\_s] [numeric](14, 2) NULL ,

[bill\_prot\_i\_n\_s] [numeric](14, 2) NULL ,

[bill\_late\_ag\_penalty] [numeric](14, 2) NULL ,

[bill\_m\_n\_o\_pd] [numeric](14, 2) NULL ,

[bill\_i\_n\_s\_pd] [numeric](14, 2) NULL ,

[penalty\_m\_n\_o\_pd] [numeric](14, 2) NULL ,

[penalty\_i\_n\_s\_pd] [numeric](14, 2) NULL ,

[interest\_m\_n\_o\_pd] [numeric](14, 2) NULL ,

[interest\_i\_n\_s\_pd] [numeric](14, 2) NULL ,

[attorney\_fees\_pd] [numeric](14, 2) NULL ,

[bill\_assessed\_value] [numeric](14, 2) NULL ,

[bill\_taxable\_val] [numeric](14, 2) NULL ,

[stmnt\_id] [numeric](18, 0) NULL ,

[discount\_mno\_pd] [numeric](14, 2) NULL ,

[discount\_ins\_pd] [numeric](14, 2) NULL ,

[prev\_bill\_id] [int] NULL ,

[new\_bill\_id] [int] NULL ,

[create\_dt] [datetime] NULL ,

[ref\_id1] [varchar] (20) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[ref\_id2] [varchar] (20) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[ref\_id3] [varchar] (20) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[ref\_id4] [varchar] (20) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[ref\_id5] [varchar] (20) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[discount\_offered] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[levy\_group\_id] [int] NULL ,

[levy\_run\_id] [int] NULL ,

[active\_bill] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[q1\_amt] [numeric](14, 2) NULL ,

[q1\_paid] [numeric](14, 2) NULL ,

[q1\_due\_dt] [datetime] NULL ,

[q2\_amt] [numeric](14, 2) NULL ,

[q2\_paid] [numeric](14, 2) NULL ,

[q2\_due\_dt] [datetime] NULL ,

[q3\_amt] [numeric](14, 2) NULL ,

[q3\_paid] [numeric](14, 2) NULL ,

[q3\_due\_dt] [datetime] NULL ,

[q4\_amt] [numeric](14, 2) NULL ,

[q4\_paid] [numeric](14, 2) NULL ,

[q4\_due\_dt] [datetime] NULL ,

[q\_bill] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[q\_create\_dt] [datetime] NULL ,

[q\_remove\_dt] [datetime] NULL ,

[q\_created\_by] [int] NULL ,

[q\_removed\_by] [int] NULL ,

[bill\_adj\_m\_n\_o] [numeric](14, 2) NULL ,

[bill\_adj\_i\_n\_s] [numeric](14, 2) NULL ,

[refund\_m\_n\_o\_pd] [numeric](14, 2) NULL ,

[refund\_i\_n\_s\_pd] [numeric](14, 2) NULL ,

[refund\_pen\_m\_n\_o\_pd] [numeric](14, 2) NULL ,

[refund\_pen\_i\_n\_s\_pd] [numeric](14, 2) NULL ,

[refund\_int\_m\_n\_o\_pd] [numeric](14, 2) NULL ,

[refund\_int\_i\_n\_s\_pd] [numeric](14, 2) NULL ,

[refund\_atty\_fee\_pd] [numeric](14, 2) NULL ,

[underage\_mno\_pd] [numeric](14, 2) NULL ,

[underage\_ins\_pd] [numeric](14, 2) NULL ,

[overage\_mno\_pd] [numeric](14, 2) NULL ,

[overage\_ins\_pd] [numeric](14, 2) NULL ,

[refund\_disc\_mno\_pd] [numeric](14, 2) NULL ,

[refund\_disc\_ins\_pd] [numeric](14, 2) NULL ,

[ia\_id] [int] NULL ,

[pay\_type] [varchar] (5) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[pay1\_amt] [numeric](14, 2) NULL ,

[pay1\_paid] [numeric](14, 2) NULL ,

[pay1\_due\_dt] [datetime] NULL ,

[pay2\_amt] [numeric](14, 2) NULL ,

[pay2\_paid] [numeric](14, 2) NULL ,

[pay2\_due\_dt] [datetime] NULL ,

[pay3\_amt] [numeric](14, 2) NULL ,

[pay3\_paid] [numeric](14, 2) NULL ,

[pay3\_due\_dt] [datetime] NULL ,

[pay4\_amt] [numeric](14, 2) NULL ,

[pay4\_paid] [numeric](14, 2) NULL ,

[pay4\_due\_dt] [datetime] NULL ,

[pay\_created\_dt] [datetime] NULL ,

[pay\_removed\_dt] [datetime] NULL ,

[pay\_created\_by] [int] NULL ,

[pay\_removed\_by] [int] NULL ,

[refund\_underage\_mno\_pd] [numeric](14, 2) NULL ,

[refund\_underage\_ins\_pd] [numeric](14, 2) NULL ,

[refund\_overage\_mno\_pd] [numeric](14, 2) NULL ,

[refund\_overage\_ins\_pd] [numeric](14, 2) NULL ,

[taxes\_paid] [bit] NULL,

CONSTRAINT [CPK\_bill] PRIMARY KEY CLUSTERED

(

[bill\_id],

[sup\_tax\_yr],

[sup\_num],

[entity\_id],

[prop\_id],

[owner\_id]

) WITH FILLFACTOR = 90 ON [PRIMARY]

) ON [PRIMARY] '''

exec(@sql)

end

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = 'Texas [bill\_adjust\_code]'

set @sql = @input\_database\_name + '.dbo.sp\_executesql N''

CREATE TABLE [bill\_adjust\_code] (

[adjust\_cd] [varchar] (10) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NOT NULL ,

[adjust\_desc] [varchar] (50) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NOT NULL ,

[deferral\_cd] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[alert\_user] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[use\_penalty] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[penalty\_rate] [numeric](4, 0) NULL ,

[use\_interest] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[interest\_rate] [numeric](4, 0) NULL ,

[use\_attorney\_fee] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[attorney\_fee\_rate] [numeric](4, 0) NULL ,

[use\_range] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[begin\_range] [numeric](4, 0) NULL ,

[end\_range] [numeric](4, 0) NULL ,

[sys\_flag] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[judgement\_cd] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

CONSTRAINT [CPK\_bill\_adjust\_code] PRIMARY KEY CLUSTERED

(

[adjust\_cd]

) WITH FILLFACTOR = 100 ON [PRIMARY]

) ON [PRIMARY] '''

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = 'Texas [entity]'

set @sql = @input\_database\_name + '.dbo.sp\_executesql N''

CREATE TABLE [entity] (

[entity\_id] [int] NOT NULL ,

[entity\_cd] [char] (5) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NOT NULL ,

[entity\_type\_cd] [char] (5) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NOT NULL ,

[entity\_disb\_bal] [numeric](14, 2) NULL ,

[taxing\_unit\_num] [varchar] (50) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[mbl\_hm\_submission] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[freeports\_allowed] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[ptd\_multi\_unit] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[appr\_company\_entity\_cd] [char] (5) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[refund\_default\_flag] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[weed\_control] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[fiscal\_begin\_date] [datetime] NULL ,

[fiscal\_end\_date] [datetime] NULL ,

[fiscal\_year] [varchar] (10) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[county\_taxing\_unit\_ind] [varchar] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[collector\_id] [int] NULL ,

[rendition\_entity] [bit] NULL ,

[enable\_timber\_78] [bit] NULL CONSTRAINT [DF\_entity\_enable\_timber\_78] DEFAULT (0),

CONSTRAINT [CPK\_entity] PRIMARY KEY CLUSTERED

(

[entity\_id]

) WITH FILLFACTOR = 90 ON [PRIMARY] ,

) ON [PRIMARY] '''

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

if @region = @texas

begin

set @status = 'Texas [payment]'

set @sql = @input\_database\_name + '.dbo.sp\_executesql N''

CREATE TABLE [payment] (

[payment\_id] [int] NOT NULL ,

[batch\_id] [int] NULL ,

[amt\_due] [numeric](14, 2) NULL ,

[check\_num] [varchar] (50) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[mo\_num] [varchar] (50) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[check\_amt] [numeric](14, 2) NULL ,

[cash\_amt] [numeric](14, 2) NULL ,

[mo\_amt] [numeric](14, 2) NULL ,

[payment\_type] [char] (5) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[payment\_code] [char] (5) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[rcpt\_num] [int] NULL ,

[payee\_id] [int] NULL ,

[operator\_id] [int] NULL ,

[post\_date] [datetime] NULL ,

[date\_paid] [datetime] NULL ,

[dl\_number] [varchar] (50) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[dl\_state] [varchar] (2) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[dl\_exp\_date] [datetime] NULL ,

[void\_payment] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[void\_date] [datetime] NULL ,

[void\_by\_id] [int] NULL ,

[void\_reason] [varchar] (255) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[void\_batch\_id] [int] NULL ,

[new\_payment\_id] [int] NULL ,

[prev\_payment\_id] [int] NULL ,

[paid\_by] [varchar] (50) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[cc\_type] [varchar] (5) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[cc\_amt] [numeric](14, 2) NULL ,

[cc\_fee] [numeric](14, 2) NULL ,

[cc\_last\_four\_digits] [varchar] (4) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[cc\_auth] [varchar] (20) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

CONSTRAINT [CPK\_payment] PRIMARY KEY CLUSTERED

(

[payment\_id]

) WITH FILLFACTOR = 90 ON [PRIMARY] ,

) ON [PRIMARY] '''

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

end

set @status = 'Texas [payment\_trans]'

set @sql = @input\_database\_name + '.dbo.sp\_executesql N''

CREATE TABLE [payment\_trans] (

[transaction\_id] [int] NOT NULL ,

[payment\_id] [int] NOT NULL ,

[prop\_id] [int] NULL ,

[bill\_id] [int] NULL ,

[fee\_id] [int] NULL ,

[trans\_type] [varchar] (5) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[fee\_amt] [numeric](14, 2) NULL ,

[mno\_amt] [numeric](14, 2) NULL ,

[ins\_amt] [numeric](14, 2) NULL ,

[penalty\_mno\_amt] [numeric](14, 2) NULL ,

[penalty\_ins\_amt] [numeric](14, 2) NULL ,

[interest\_mno\_amt] [numeric](14, 2) NULL ,

[interest\_ins\_amt] [numeric](14, 2) NULL ,

[attorney\_fee\_amt] [numeric](14, 2) NULL ,

[q1\_amt] [numeric](14, 2) NULL ,

[q2\_amt] [numeric](14, 2) NULL ,

[q3\_amt] [numeric](14, 2) NULL ,

[q4\_amt] [numeric](14, 2) NULL ,

[mno\_due] [numeric](14, 2) NULL ,

[ins\_due] [numeric](14, 2) NULL ,

[penalty] [numeric](14, 2) NULL ,

[interest] [numeric](14, 2) NULL ,

[attorney\_fee] [numeric](14, 2) NULL ,

[fee\_due] [numeric](14, 2) NULL ,

[fiscal\_year] [varchar] (10) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[fiscal\_month] [int] NULL ,

[fiscal\_entity\_id] [int] NULL ,

[discount\_mno\_amt] [numeric](14, 2) NULL ,

[discount\_ins\_amt] [numeric](14, 2) NULL ,

[underage\_mno\_amt] [numeric](14, 2) NULL ,

[underage\_ins\_amt] [numeric](14, 2) NULL ,

[overage\_mno\_amt] [numeric](14, 2) NULL ,

[overage\_ins\_amt] [numeric](14, 2) NULL ,

[refund\_mno\_amt] [numeric](14, 2) NULL ,

[refund\_ins\_amt] [numeric](14, 2) NULL ,

[void\_trans] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[void\_date] [datetime] NULL ,

[void\_by\_id] [int] NULL ,

[void\_reason] [varchar] (255) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[void\_batch\_id] [int] NULL ,

[prev\_transaction\_id] [int] NULL ,

[prev\_payment\_id] [int] NULL ,

CONSTRAINT [CPK\_payment\_trans] PRIMARY KEY CLUSTERED

(

[transaction\_id],

[payment\_id]

) WITH FILLFACTOR = 90 ON [PRIMARY] ,

) ON [PRIMARY] '''

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = 'Texas [refund\_due\_trans]'

set @sql = @input\_database\_name + '.dbo.sp\_executesql N''

CREATE TABLE [refund\_due\_trans] (

[transaction\_id] [int] NOT NULL ,

[batch\_id] [int] NOT NULL ,

[bill\_id] [int] NOT NULL ,

[mno\_amt] [numeric](14, 2) NULL ,

[ins\_amt] [numeric](14, 2) NULL ,

[penalty\_mno\_amt] [numeric](14, 2) NULL ,

[penalty\_ins\_amt] [numeric](14, 2) NULL ,

[interest\_mno\_amt] [numeric](14, 2) NULL ,

[interest\_ins\_amt] [numeric](14, 2) NULL ,

[atty\_fee\_amt] [numeric](14, 2) NULL ,

[payment\_trans\_id] [int] NULL ,

[adjust\_id] [int] NULL,

[discount\_mno\_amt] [numeric](14, 2) NULL,

[discount\_ins\_amt] [numeric](14, 2) NULL,

[underage\_mno\_amt] [numeric](14, 2) NULL,

[underage\_ins\_amt] [numeric](14, 2) NULL,

[overage\_mno\_amt] [numeric](14, 2) NULL,

[overage\_ins\_amt] [numeric](14, 2) NULL,

CONSTRAINT [CPK\_refund\_due\_trans] PRIMARY KEY CLUSTERED

(

[transaction\_id],

[batch\_id],

[bill\_id]

) WITH FILLFACTOR = 100 ON [PRIMARY]

) ON [PRIMARY] '''

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = 'Texas [tax\_rate]'

set @sql = @input\_database\_name + '.dbo.sp\_executesql N''

CREATE TABLE [tax\_rate] (

[entity\_id] [int] NOT NULL ,

[tax\_rate\_yr] [numeric](4, 0) NOT NULL ,

[discount\_dt] [datetime] NULL ,

[late\_dt] [datetime] NULL ,

[attorney\_fee\_dt] [datetime] NULL ,

[bills\_created\_dt] [datetime] NULL ,

[m\_n\_o\_tax\_pct] [numeric](13, 10) NULL ,

[i\_n\_s\_tax\_pct] [numeric](13, 10) NULL ,

[prot\_i\_n\_s\_tax\_pct] [numeric](13, 10) NULL ,

[sales\_tax\_pct] [numeric](13, 10) NULL ,

[levy\_start\_rct\_num] [numeric](18, 0) NULL ,

[supp\_start\_rct\_num] [numeric](18, 0) NULL ,

[stmnt\_dt] [datetime] NULL ,

[collect\_for] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[appraise\_for] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[ready\_to\_certify] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[special\_inv\_entity] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[ready\_to\_create\_bill] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[PLUS\_1\_INT\_PCT] [numeric](13, 10) NULL ,

[PLUS\_1\_PENALTY\_PCT] [numeric](13, 10) NULL ,

[PLUS\_2\_INT\_PCT] [numeric](13, 10) NULL ,

[PLUS\_2\_PENALTY\_PCT] [numeric](13, 10) NULL ,

[PLUS\_3\_INT\_PCT] [numeric](13, 10) NULL ,

[PLUS\_3\_PENALTY\_PCT] [numeric](13, 10) NULL ,

[PLUS\_4\_INT\_PCT] [numeric](13, 10) NULL ,

[PLUS\_4\_PENALTY\_PCT] [numeric](13, 10) NULL ,

[PLUS\_5\_INT\_PCT] [numeric](13, 10) NULL ,

[PLUS\_5\_PENALTY\_PCT] [numeric](13, 10) NULL ,

[PLUS\_6\_INT\_PCT] [numeric](13, 10) NULL ,

[PLUS\_6\_PENALTY\_PCT] [numeric](13, 10) NULL ,

[PLUS\_7\_INT\_PCT] [numeric](13, 10) NULL ,

[PLUS\_7\_PENALTY\_PCT] [numeric](13, 10) NULL ,

[PLUS\_8\_INT\_PCT] [numeric](13, 10) NULL ,

[PLUS\_8\_PENALTY\_PCT] [numeric](13, 10) NULL ,

[PLUS\_9\_INT\_PCT] [numeric](13, 10) NULL ,

[PLUS\_9\_PENALTY\_PCT] [numeric](13, 10) NULL ,

[attorney\_fee\_pct] [numeric](4, 2) NULL ,

[effective\_due\_dt] [datetime] NULL ,

[collect\_option] [char] (5) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[weed\_control] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[weed\_control\_pct] [numeric](4, 2) NULL ,

[ptd\_option] [char] (1) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL ,

[apply\_bpp\_attorney\_fees] [bit] NOT NULL CONSTRAINT [DF\_\_clientdb\_tax\_rate\_\_apply\_\_\_74F30FE8] DEFAULT (0),

[bpp\_attorney\_fee\_dt] [datetime] NULL ,

CONSTRAINT [CPK\_tax\_rate] PRIMARY KEY CLUSTERED

(

[entity\_id],

[tax\_rate\_yr]

) WITH FILLFACTOR = 90 ON [PRIMARY] ,

) ON [PRIMARY] '''

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

-- 9.0 MT Stuff

print 'Creating Middle Tier Specific Tables'

set @status = 'abs\_subdv'

set @sql = 'use ' + @input\_database\_name + ' '

set @sql = @sql + 'create table abs\_subdv

(

[abs\_subdv\_cd] varchar(10) not null,

[abs\_subdv\_yr] numeric(4,0) not null,

[abs\_subdv\_desc] varchar(60) null,

[abs\_land\_pct] numeric(5,2) not null,

[abs\_imprv\_pct] numeric(5,2) not null,

[abs\_subdv\_ind] char(1) null,

[sys\_flag] char(1) null,

[changed\_flag] char(1) null,

[cInCounty] char(1) not null,

[bActive] bit null,

[ls\_id] int null,

[active\_year] numeric(4,0) null,

[create\_date] datetime null

)'

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

if @region = @washington

begin

declare @commercial\_codes varchar(200);

declare @farm\_codes varchar(20);

declare @residential\_codes varchar(200);

set @commercial\_codes = '12,13,15,16,17,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,41,42,43,44,45,46,47,48,49,50,51,52,53,54,55,56,57,58,59,61,62,63,64,65,66,67,68,69,71,72,73,74,75,76,77,78,79';

set @farm\_codes = '81,83';

set @residential\_codes = '11,14,18,19,88,94,95,82,84,85,89,91,92,93,96,97,98,99';

CREATE TABLE #commercial\_dor(dor\_use\_cd varchar(10))

INSERT INTO #commercial\_dor

SELECT Id FROM dbo.fn\_ReturnTableFromCommaSepValues(@commercial\_codes)

CREATE TABLE #farm\_dor(dor\_use\_cd varchar(10))

INSERT INTO #farm\_dor

SELECT Id FROM dbo.fn\_ReturnTableFromCommaSepValues(@farm\_codes)

CREATE TABLE #residential\_dor(dor\_use\_cd varchar(10))

INSERT INTO #residential\_dor

SELECT Id FROM dbo.fn\_ReturnTableFromCommaSepValues(@residential\_codes)

--Sales Search

set @status = 'Sales Search'

-- set year value for testing

declare @input\_year numeric(4,0)

select @input\_year = max(tax\_yr) from \_clientdb\_pacs\_year

-- select count(\*) from \_clientdb\_sales -- 5669 , with tax\_area\_fund 111727

-- drop tables needed for processing if they exists

if object\_id('\_clientdb\_sales') is not null -- final outcome table

begin

truncate table \_clientdb\_sales

drop table \_clientdb\_sales

end

if object\_id('\_PA\_buyers') is not null -- work table for buyer name info

begin

truncate table \_PA\_buyers

drop table \_PA\_buyers

end

if object\_id('\_PA\_sellers') is not null -- work table for seller name info

begin

truncate table \_PA\_sellers

drop table \_PA\_sellers

end

if object\_id('\_PA\_owners') is not null -- work table for owner name info

begin

truncate table \_PA\_owners

drop table \_PA\_owners

end

-- permanent tables are built for seller,buyer,owner names

-- so function to creat comma delimited output

-- has real table to work on -- functions do not allow temp tables

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

get BUYER info, accounting for multiple buyers per sale with comma delimited string

broken into several steps to make the processing faster

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

create table \_PA\_buyers (chg\_of\_owner\_id int,buyer varchar(1000),multi\_buyer bit)

insert into \_PA\_buyers(chg\_of\_owner\_id,buyer)

select

b.chg\_of\_owner\_id,

case

when a.confidential\_flag = 'T' or a.web\_suppression = '1'

then (select IsNull(confidential\_file\_as\_name, 'Confidential') as confidential\_file\_as\_name

from pacs\_system)

else a.file\_as\_name

end as buyer

from account as a with(nolock) -- pk cluster acct\_id

inner join

buyer\_assoc as b with(nolock) -- pk cluster chg\_of\_owner\_id, buyer\_id

on a.acct\_id = b.buyer\_id

create index idx\_tmp on \_PA\_buyers(chg\_of\_owner\_id,multi\_buyer)

-- update multi buyer value

update b

set multi\_buyer = 1

from \_PA\_buyers b

join

(select

chg\_of\_owner\_id

from \_PA\_buyers

group by chg\_of\_owner\_id

having count(\*) > 1

) as dups

on b.chg\_of\_owner\_id = dups.chg\_of\_owner\_id

update \_PA\_buyers

set buyer = dbo.fn\_PA\_GetFileAsName\_CommaDelimited(chg\_of\_owner\_id,'B',0)

where multi\_buyer = 1

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

get SELLER info, accounting for multiple sellers per sale with comma delimited string

broken into several steps to make the processing faster

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

create table \_PA\_sellers (chg\_of\_owner\_id int,prop\_id int, seller varchar(1000),multi\_seller bit)

insert into \_PA\_sellers(chg\_of\_owner\_id,prop\_id,seller)

select

s.chg\_of\_owner\_id,

s.prop\_id,

case

when a.confidential\_flag = 'T' or a.web\_suppression = '1'

then (select IsNull(confidential\_file\_as\_name, 'Confidential') as confidential\_file\_as\_name

from pacs\_system)

else a.file\_as\_name

end as seller

from account as a with(nolock) -- pk cluster acct\_id

inner join

seller\_assoc as s with(nolock) -- pk cluster chg\_of\_owner\_id, prop\_id, seller\_id

on a.acct\_id = s.seller\_id

create index idx\_tmp on \_PA\_sellers(chg\_of\_owner\_id,prop\_id,multi\_seller)

-- update multi seller value

update s

set multi\_seller = 1

from \_PA\_sellers s

join

(select

chg\_of\_owner\_id

from \_PA\_sellers

group by chg\_of\_owner\_id

having count(\*) > 1

) as dups

on s.chg\_of\_owner\_id = dups.chg\_of\_owner\_id

update \_PA\_sellers

set seller = dbo.fn\_PA\_GetFileAsName\_CommaDelimited(chg\_of\_owner\_id,'S',prop\_id)

where multi\_seller = 1

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

get CURRENT OWNER info, accounting for multiple owners comma delimited string

broken into several steps to make the processing faster

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

-- determine owner name

create table \_PA\_owners (prop\_id int, owner\_name varchar(1000),multi\_owner bit)

insert into \_PA\_owners(prop\_id,owner\_name)

select

o.prop\_id,

case

when a.confidential\_flag = 'T' or a.web\_suppression = '1'

then (select IsNull(confidential\_file\_as\_name, 'Confidential') as confidential\_file\_as\_name

from pacs\_system)

else a.file\_as\_name

end as owner\_name

from account as a with(nolock) -- pk cluster acct\_id

inner join

owner as o with(nolock) -- pk cluster owner\_tax\_yr, sup\_num, prop\_id, owner\_id

on o.owner\_tax\_yr = @input\_year

and a.acct\_id = o.owner\_id

inner join

prop\_supp\_assoc as psa with(nolock) -- pk cluster owner\_tax\_yr, sup\_num, prop\_id

on psa.owner\_tax\_yr = @input\_year

and o.owner\_tax\_yr = psa.owner\_tax\_yr

and o.sup\_num = psa.sup\_num

and o.prop\_id = psa.prop\_id

create index idx\_tmp on \_PA\_owners(prop\_id,multi\_owner)

-- update multi owner value

update o

set multi\_owner = 1

from \_PA\_owners o

join

(select

prop\_id

from \_PA\_owners

group by prop\_id

having count(\*) > 1

) as dups

on o.prop\_id = dups.prop\_id

update \_PA\_owners

set owner\_name = dbo.fn\_PA\_GetFileAsName\_CommaDelimited(0,'O',prop\_id)

where multi\_owner = 1

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

select into final outcome table

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

select

sale.chg\_of\_owner\_id,

p.prop\_id,

p.prop\_type\_cd,

case

when (p.prop\_type\_cd = 'R' and pu.dor\_use\_code in (select dor\_use\_cd from #commercial\_dor))

then 'CM'

when (p.prop\_type\_cd = 'R' and pu.dor\_use\_code in (select dor\_use\_cd from #farm\_dor))

then 'FM'

when (p.prop\_type\_cd = 'R' and pu.dor\_use\_code in (select dor\_use\_cd from #residential\_dor))

then 'RS'

when (p.prop\_type\_cd = 'MH')

then 'MH'

else

null

end as property\_type,

pp.state\_cd,

pp.school\_id,

pp.city\_id ,

pp.class\_cd as imprv\_class,

pp.yr\_blt as actual\_yr\_built,

pp.living\_area as living\_area\_sqft,

pp.land\_type\_cd,

sale.sl\_dt as sale\_dt,

sale.adjusted\_sl\_price as sl\_price,

sale.adjusted\_sl\_price as sl\_adj\_price,

sale.sl\_type\_cd,

sale.land\_only\_sale,

--pv.last\_appraiser\_id,

case when (pv.prop\_inactive\_dt is null or pv.udi\_parent = 'T')

then IsNull(sale.include\_no\_calc, 'F')

else 'T'

end as include\_no\_calc,

sale.sl\_ratio\_type\_cd as sl\_ratio\_cd,

pp.eff\_yr\_blt as eff\_yr\_built,

IsNull(sale.include\_reason, '') as include\_reason,

p.geo\_id,

p.simple\_geo\_id,

-- pv.map\_id,

CASE WHEN sale.sl\_price <> sale.adjusted\_sl\_price

THEN sale.sl\_adj\_rsn

ELSE ''

END as sl\_adj\_reason,

sale.sl\_price as true\_sl\_price,

-- p.dba\_name,

pp.property\_use\_cd as local\_dor\_code,

case when isnull(pp.living\_area,0) >0

then isnull(pp.land\_total\_sqft,0) / pp.living\_area

else 0

end as living\_area\_sqft2,

isnull(pp.living\_area,0) as living\_area,

pp.imprv\_det\_sub\_class\_cd as imprv\_sub\_class,

pp.condition\_cd,

pp.heat\_ac\_code,

isnull(pp.land\_total\_sqft,0) as land\_total\_sqft,

isnull(pp.land\_total\_acres,0) as land\_total\_acres,

isnull(pp.imprv\_add\_val,0) as additive\_val,

isnull(pp.percent\_complete,0) as percent\_complete,

pp.sub\_market\_cd ,

pp.imprv\_type\_cd,

pp.imprv\_det\_meth\_cd\_highvalueimprov as imprv\_det\_meth\_cd,

pp.imprv\_det\_sub\_class\_cd,

pu.dor\_use\_code as state\_dor\_code,

ta.tax\_area\_number,

ta.tax\_area\_id,

pp.characteristic\_zoning1 as zoning,

pp.mbl\_hm\_make as mh\_make,

pp.mbl\_hm\_model as mh\_model,

pp.mbl\_hm\_sn as mh\_serial,

pp.mbl\_hm\_hud\_num as mh\_hud,

pp.mbl\_hm\_title\_num as mh\_title,

cast('0' as bit) as multi\_prop\_sale, -- initially set to false, updated later

pv.market,

coo.excise\_number,

coo.deed\_type\_cd,

coo.deed\_num,

coo.deed\_book\_id,

coo.deed\_book\_page,

coo.deed\_dt,

coo.grantor\_cv,

coo.grantee\_cv

,sell.seller

,buy.buyer

,own.owner\_name as current\_owner

,webp.prop\_type\_desc

,webp.situs\_display

,webp.legal\_desc

,webp.owner\_name

,webp.tax\_area

,webp.prop\_val\_yr

,webp.show\_values

,webp.abs\_subdv\_cd

into \_clientdb\_sales

from

property as p WITH (NOLOCK) -- pk cluster prop\_id

inner join

property\_profile as pp WITH (NOLOCK) -- pk cluster prop\_val\_yr, prop\_id

on

pp.prop\_val\_yr = @input\_year

and p.prop\_id = pp.prop\_id

inner join

prop\_supp\_assoc as psa WITH (NOLOCK) -- pk cluster owner\_tax\_yr, sup\_num, prop\_id

on

psa.owner\_tax\_yr = @input\_year

and pp.prop\_val\_yr = psa.owner\_tax\_yr

and pp.prop\_id = psa.prop\_id

inner join

property\_val as pv WITH (NOLOCK) -- pk cluster prop\_val\_yr, sup\_num, prop\_id

on

pv.prop\_val\_yr = @input\_year

and psa.owner\_tax\_yr = pv.prop\_val\_yr

and psa.sup\_num = pv.sup\_num

and psa.prop\_id = pv.prop\_id

left outer join

property\_use as pu with (nolock)

on pv.property\_use\_cd = pu.property\_use\_cd

inner join

property\_tax\_area as pta WITH (NOLOCK) -- pk cluster year, sup\_num, prop\_id

on pv.prop\_val\_yr = pta.year

and pv.sup\_num = pta.sup\_num

and pv.prop\_id = pta.prop\_id

inner join

tax\_area as ta WITH (NOLOCK) -- pk cluster tax\_area\_id

on pta.tax\_area\_id = ta.tax\_area\_id

inner join

chg\_of\_owner\_prop\_assoc copa WITH (NOLOCK) -- pk non cluster chg\_of\_owner\_id, prop\_id; cluster on prop\_id

on

copa.prop\_id = pp.prop\_id

inner join

sale WITH (NOLOCK) -- pk cluster chg\_of\_owner\_id

on

copa.chg\_of\_owner\_id = sale.chg\_of\_owner\_id

inner join

chg\_of\_owner coo with (nolock) -- pk cluster chg\_of\_owner\_id

on

copa.chg\_of\_owner\_id = coo.chg\_of\_owner\_id

left join

(select distinct chg\_of\_owner\_id ,buyer

from \_PA\_buyers

)as buy

on

copa.chg\_of\_owner\_id = buy.chg\_of\_owner\_id

left join

(select distinct chg\_of\_owner\_id , prop\_id, seller

from \_PA\_sellers

)as sell

on

copa.chg\_of\_owner\_id = sell.chg\_of\_owner\_id

and copa.prop\_id = sell.prop\_id

left join

(select distinct prop\_id, owner\_name

from \_PA\_owners

)as own

on

copa.prop\_id = own.prop\_id

join \_clientdb\_property as webp with (nolock)

on pp.prop\_id = webp.prop\_id

and pp.prop\_val\_yr = webp.prop\_val\_yr

where

(sale.suppress\_on\_ratio\_rpt\_cd = 'F' or sale.suppress\_on\_ratio\_rpt\_cd is null)

and sale.adjusted\_sl\_price > 0 and sale.adjusted\_sl\_price is not null

and (pv.prop\_inactive\_dt is null or pv.udi\_parent = 'T')

and (sale.confidential\_sale is null or sale.confidential\_sale = 'F')

create index idx\_\_clientdb\_sales\_chg\_of\_owner\_id on \_clientdb\_sales(chg\_of\_owner\_id)

drop table #commercial\_dor

drop table #farm\_dor

drop table #residential\_dor

-- \_clientdb\_property\_tax\_district\_assoc

select

pv.prop\_id,

pv.prop\_val\_yr,

td.tax\_district\_cd

into \_clientdb\_property\_tax\_district\_assoc

from property\_val as pv with (nolock)

join #layer\_assoc as psa with (nolock)

on pv.prop\_val\_yr = psa.owner\_tax\_yr and

pv.sup\_num = psa.sup\_num and

pv.prop\_id = psa.prop\_id

join property\_tax\_area as pta with (nolock) -- pk cluster year, sup\_num, prop\_id

on pv.prop\_val\_yr = pta.year and

pv.sup\_num = pta.sup\_num and

pv.prop\_id = pta.prop\_id

join (select distinct year, tax\_district\_id, tax\_area\_id from

tax\_area\_fund\_assoc with (nolock)) as tafa

on pta.year = tafa.year and

pta.tax\_area\_id = tafa.tax\_area\_id

join tax\_district as td with (nolock)

on tafa.tax\_district\_id = td.tax\_district\_id

create index idx\_\_clientdb\_property\_tax\_district\_assoc

on \_clientdb\_property\_tax\_district\_assoc(prop\_val\_yr, prop\_id, tax\_district\_cd)

-- find sales with multi properties

select

chg\_of\_owner\_id

into #tmpDups

from \_clientdb\_sales

group by chg\_of\_owner\_id

having count(\*) > 1

create index idx\_tmp on #tmpDups(chg\_of\_owner\_id)

-- update multi\_prop\_sale value

update h

set multi\_prop\_sale = 1

from \_clientdb\_sales h

join

#tmpDups as dups

on h.chg\_of\_owner\_id = dups.chg\_of\_owner\_id

-- clear and drop work tables

drop table #tmpDups

if object\_id('\_PA\_buyers') is not null

begin

truncate table \_PA\_buyers

drop table \_PA\_buyers

end

if object\_id('\_PA\_sellers') is not null

begin

truncate table \_PA\_sellers

drop table \_PA\_sellers

end

if object\_id('\_PA\_owners') is not null

begin

truncate table \_PA\_owners

drop table \_PA\_owners

end

create index idx\_\_clientdb\_sales\_sale\_dt on \_clientdb\_sales(sale\_dt)

CREATE CLUSTERED INDEX IX\_\_clientdb\_sales\_multi\_prop\_sale ON dbo.\_clientdb\_sales

(

prop\_id,

multi\_prop\_sale

) ON [PRIMARY]

--Sales Search

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

end

drop table #layer\_assoc

--create a new temp table and populate it

create table #tables(tableName varchar(128), populateData bit, buildIndices bit)

insert into #tables(tableName, populateData)

select szTableName, 0

from table\_cache\_status with (nolock)

-- New Tables for ALL regions

insert into #tables(tableName, populateData)

values ('pacs\_config', 1)

-- Populate for ALL regions

update #tables

set populateData = 1

where tableName in ('pacs\_system')

-- Region specific tables

if @region = @washington

begin

update #tables

set populateData = 1

where tableName in ('refund\_type', 'bill\_fee\_code', 'fee\_type', 'payout\_agreement\_type',

'payout\_agreement\_status\_code', 'payment\_terms\_type', 'imprv\_type',

'penalty\_interest\_frequency\_type', 'penalty\_interest\_ref\_date\_type',

'penalty\_interest\_ref\_type', 'penalty\_interest\_type', 'tax\_district',

'tax\_district\_type', 'special\_assessment\_agency', 'payout\_agreement\_status\_code',

'payment\_terms\_type', 'tax\_area', 'property\_use', 'imprv\_det\_meth', 'dor\_use\_code',

'holiday\_schedule', 'special\_assessment')

insert into #tables(tableName, populateData)

values ('bill', 0)

insert into #tables(tableName, populateData)

values ('bill\_payments\_due', 0)

insert into #tables(tableName, populateData)

values ('bill\_payments\_due', 0)

insert into #tables(tableName, populateData)

values ('levy\_bill', 0)

insert into #tables(tableName, populateData)

values ('assessment\_bill', 0)

insert into #tables(tableName, populateData)

values ('coll\_transaction', 0)

insert into #tables(tableName, populateData)

values ('fee', 0)

insert into #tables(tableName, populateData)

values ('fee\_payments\_due', 0)

insert into #tables(tableName, populateData)

values ('payout\_agreement\_bill\_assoc', 0)

insert into #tables(tableName, populateData)

values ('payout\_agreement\_fee\_assoc', 0)

insert into #tables(tableName, populateData)

values ('payout\_agreement', 0)

insert into #tables(tableName, populateData)

values ('payout\_agreement\_schedule', 0)

insert into #tables(tableName, populateData)

values ('special\_assessment\_statement\_options', 0)

insert into #tables(tableName, populateData)

values ('trans\_group', 0)

insert into #tables(tableName, populateData)

values ('next\_unique\_id', 0)

insert into #tables(tableName, populateData)

values ('user\_input\_query', 0)

insert into #tables(tableName, populateData)

values ('user\_input\_query\_idlist', 0)

insert into #tables(tableName, populateData)

values ('pending\_coll\_transaction', 0)

insert into #tables(tableName, populateData)

values ('tax\_due\_calc\_list', 0)

insert into #tables(tableName, populateData)

values ('tax\_due\_calc\_bill', 0)

insert into #tables(tableName, populateData)

values ('tax\_due\_calc\_bill\_payments\_due', 0)

insert into #tables(tableName, populateData)

values ('tax\_due\_calc\_fee', 0)

insert into #tables(tableName, populateData)

values ('tax\_due\_calc\_fee\_payments\_due', 0)

insert into #tables(tableName, populateData)

values ('bill\_fee\_assoc', 1)

insert into #tables(tableName, populateData)

values ('fee\_prop\_assoc', 1)

insert into #tables(tableName, populateData)

values ('reet\_fee\_assoc', 0)

insert into #tables(tableName, populateData)

values ('fee\_acct\_assoc', 0)

insert into #tables(tableName, populateData)

values ('property', 1)

insert into #tables(tableName, populateData)

values ('levy\_link', 1)

insert into #tables(tableName, populateData)

values ('next\_unique\_id', 1)

insert into #tables(tableName, populateData)

values ('property\_payout\_agreement', 1)

insert into #tables(tableName, populateData)

values ('payment\_transaction\_assoc', 0)

insert into #tables(tableName, populateData)

values ('payment', 0)

insert into #tables(tableName, populateData)

values ('wa\_tax\_statement', 0)

insert into #tables(tableName, populateData)

values ('core\_config', 1)

insert into #tables(tableName, populateData)

values ('imprv\_det\_sub\_class', 1)

insert into #tables(tableName, populateData)

values ('\_clientdb\_sales', 1)

insert into #tables(tableName, populateData)

values ('\_clientdb\_property\_tax\_district\_assoc', 1)

insert into #tables(tableName, populateData)

values ('chg\_of\_owner\_prop\_assoc', 1)

insert into #tables(tableName, populateData)

values ('seller\_assoc', 1)

insert into #tables(tableName, populateData)

values ('buyer\_assoc', 1)

insert into #tables(tableName, populateData)

values ('\_clientdb\_deed\_history\_detail', 1)

insert into #tables(tableName, populateData)

values ('\_clientdb\_improvement\_building\_detail', 1)

insert into #tables(tableName, populateData)

values ('\_clientdb\_land\_detail', 1)

insert into #tables(tableName, populateData)

values ('\_clientdb\_roll\_value\_history\_detail', 1)

insert into #tables(tableName, populateData)

values ('security\_fields', 1)

insert into #tables(tableName, populateData)

values ('\_clientdb\_property', 1)

insert into #tables(tableName, populateData)

values ('pacs\_user', 1)

insert into #tables(tableName, populateData)

values ('user\_right\_user\_assoc', 1)

insert into #tables(tableName, populateData)

values ('user\_role\_user\_assoc', 1)

insert into #tables(tableName, populateData)

values ('user\_role\_right\_assoc', 1)

insert into #tables(tableName, populateData)

values ('user\_role', 1)

insert into #tables(tableName, populateData)

values ('penalty\_interest\_property\_type', 1)

insert into #tables(tableName, populateData)

values ('penalty\_interest\_property\_type\_dor\_use\_code', 1)

insert into #tables(tableName, populateData)

values ('penalty\_interest\_property\_type\_imprv\_det\_type\_cd', 1)

if @township\_enabled = 1

begin

insert into #tables(tableName, populateData)

values ('township', 1)

insert into #tables(tableName, populateData)

values ('prop\_range', 1)

end

end

DECLARE @szTableName varchar(128)

DECLARE @szKeys varchar(8000)

DECLARE @szIndexes varchar(8000)

DECLARE @buildData bit

DECLARE @buildIndices bit

DECLARE tabe\_cache CURSOR READ\_ONLY

FOR

SELECT distinct tableName, populateData

FROM #tables

OPEN tabe\_cache

FETCH NEXT FROM tabe\_cache

INTO @szTableName, @buildData

WHILE @@FETCH\_STATUS = 0

BEGIN

set @sql2 = '';

exec sp\_ScriptTable @szTableName, @sql2 output, @sql output

exec sp\_ScriptSingleTableIndexes @szTableName, @szIndexes output

exec sp\_ScriptSingleTablePrimaryKey @szTableName, @szKeys output

if len(@sql2) > 0

begin

set @start\_date = getdate()

set @status = 'From OLTP: ' + @szTableName

set @sql = 'use ' + @input\_database\_name + ' '

set @sql = @sql + '

if not exists (select name from sysobjects WHERE id = OBJECT\_ID('''+ @szTableName + '''))

begin ' + @sql2 + '

end'

print 'Creating web\_internet table: ' + @szTableName

exec (@sql)

if @buildData = 1

begin

set @sql = 'insert into ' + @input\_database\_name + '.dbo.' + @szTableName + '

select \* from ' + @szTableName + ' with (nolock)'

exec(@sql)

set @sql = 'use ' + @input\_database\_name + ' ;checkpoint'

exec (@sql)

end

set @sql = 'use ' + @input\_database\_name + ' ' + @szKeys

exec (@sql)

set @sql = 'use ' + @input\_database\_name + ' ' + @szIndexes

exec (@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

end

FETCH NEXT FROM tabe\_cache

INTO @szTableName, @buildData

END

CLOSE tabe\_cache

DEALLOCATE tabe\_cache

if object\_id('tempdb..#tables') is not null

drop table #tables

--app exception log

set @status = 'app\_exception\_log'

set @sql = @input\_database\_name + '.dbo.sp\_executesql N''

CREATE TABLE app\_exception\_log (

id int IDENTITY(1,1) NOT NULL,

date\_exception datetime NOT NULL,

machine\_name varchar(23) NOT NULL,

app\_login\_id int NOT NULL,

dll\_class\_method varchar(255) NULL,

transaction\_id bigint NULL,

transaction\_input\_params varbinary(max) NULL,

exception\_data varbinary(max) NOT NULL,

exception\_text1 varchar(max) NOT NULL,

exception\_text2 varchar(max) NOT NULL,

exception\_text3 varchar(max) NOT NULL,

exception\_text4 varchar(max) NOT NULL,

exception\_callstack varchar(max) NOT NULL,

app\_state varchar(max) NULL,

exception\_type varchar(max) NULL,

app\_name varchar(max) NULL,

server\_local\_date\_exception datetime NOT NULL,

client\_app\_version varchar(max) NULL,

CONSTRAINT CPK\_app\_exception\_log PRIMARY KEY CLUSTERED (id ASC)

)'''

exec (@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print 'Done Creating Middle Tier Specific Tables'

-- 9.0 MT Stuff

/\*

\* Create views

\*/

set @status = 'clientdb\_subdivision\_vw'

set @sql = @input\_database\_name + '.dbo.sp\_executesql N''

create view dbo.clientdb\_subdivision\_vw

as

select abs\_subdv\_ind, rtrim(abs\_subdv\_cd) as code, abs\_subdv\_desc as description

from \_clientdb\_abs\_subdv with (nolock) '''

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = 'clientdb\_neighborhood\_vw'

set @sql = @input\_database\_name + '.dbo.sp\_executesql N''

create view dbo.clientdb\_neighborhood\_vw

as

select rtrim(hood\_cd) as code, hood\_name as description

from \_clientdb\_neighborhood with (nolock) '''

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = 'clientdb\_property\_vw'

set @sql = @input\_database\_name + '.dbo.sp\_executesql N''

create view dbo.clientdb\_property\_vw

as

select p.\*, a.web\_suppression, a.confidential\_flag, y.certification\_dt,

case when tmpvw.prop\_id is null then 1 else 0 end as all\_taxes\_paid,

sale.sale\_date,

ts.township\_desc as township, rr.range\_desc as Range

from \_clientdb\_property as p with (nolock)

join \_clientdb\_pacs\_year as y with (nolock)

on p.prop\_val\_yr = y.tax\_yr

join account as a with (nolock)

on a.acct\_id=p.owner\_id

left outer join (

select distinct prop\_id

from bill with (nolock)

where isnull(taxes\_paid,0) = 0

) as tmpvw on tmpvw.prop\_id = p.prop\_id

left outer join (

select sale\_date, prop\_id

from \_clientdb\_deed\_history\_detail with (nolock)

where seq\_num = 0

) as sale on sale.prop\_id = p.prop\_id

left join township as ts with (nolock)

on p.township\_code = ts.township\_code

and p.prop\_val\_yr = ts.township\_year

left join prop\_range rr with (nolock)

on p.range\_code = rr.range\_code

and p.prop\_val\_yr = rr.range\_year

where

(a.web\_suppression = ''''0'''' or a.web\_suppression Is Null) '''

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = 'property\_leased\_land\_vw'

set @sql = @input\_database\_name + '.dbo.sp\_executesql N''

CREATE VIEW dbo.property\_leased\_land\_vw

AS

SELECT DISTINCT

p.prop\_id,

p.prop\_val\_yr,

is\_leased\_land\_property

from \_clientdb\_property as p with (nolock) '''

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = 'clientdb\_taxing\_jurisdiction\_detail\_vw'

set @sql = @input\_database\_name + '.dbo.sp\_executesql N''

create view dbo.clientdb\_taxing\_jurisdiction\_detail\_vw

as

select t.\*, y.certification\_dt, max\_freeze, a.confidential\_flag,

ta.tax\_area\_number + '''' - '''' + ta.tax\_area\_description as tax\_area

from \_clientdb\_taxing\_jurisdiction\_detail as t with (nolock)

join (select prop\_id, sup\_yr, max(isnull(freeze\_ceiling,-1)) as max\_freeze

from \_clientdb\_taxing\_jurisdiction\_detail with (nolock) group by prop\_id, sup\_yr)

as t1 on

t.prop\_id = t1.prop\_id and

t.sup\_yr = t1.sup\_yr

join \_clientdb\_pacs\_year as y with (nolock)

on t.sup\_yr = y.tax\_yr

join account as a with (nolock)

on a.acct\_id=t.owner\_id

left join tax\_area as ta with (nolock)

on t.tax\_area\_id = ta.tax\_area\_id '''

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = 'clientdb\_roll\_value\_history\_detail\_vw'

set @sql = @input\_database\_name + '.dbo.sp\_executesql N''

create view dbo.clientdb\_roll\_value\_history\_detail\_vw

as

select r.\*, y.certification\_dt

from \_clientdb\_roll\_value\_history\_detail as r with (nolock)

join \_clientdb\_pacs\_year as y with (nolock)

on r.prop\_val\_yr = y.tax\_yr '''

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = 'clientdb\_map\_vw'

set @sql = @input\_database\_name + '.dbo.sp\_executesql N''

create view clientdb\_map\_vw

as

select prop\_id,

prop\_val\_yr,

geo\_id,

prop\_type\_cd,

prop\_type\_desc,

dba\_name,

case when y.certification\_dt is null then ''''N/A'''' when ISNULL(p.show\_values,''''T'''') = ''''F'''' then ''''N/A'''' else ''''$'''' + convert(varchar(20), appraised\_val) end as appraised\_val,

abs\_subdv\_cd,

mapsco,

map\_id,

agent\_cd,

hood\_cd,

hood\_name,

owner\_name,

owner\_id,

pct\_ownership,

exemptions,

state\_cd,

legal\_desc,

replace(replace(replace(situs\_display, char(10), ''''''''), char(13), '''' ''''), '''' '''', '''' '''') as situs,

jurisdictions

from \_clientdb\_property as p with (nolock)

join \_clientdb\_pacs\_year as y with (nolock)

on p.prop\_val\_yr = y.tax\_yr '''

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = 'clientdb\_map\_export\_vw'

set @sql = @input\_database\_name + '.dbo.sp\_executesql N''

create view clientdb\_map\_export\_vw

as

select prop\_id,

prop\_val\_yr,

geo\_id,

prop\_type\_cd,

prop\_type\_desc,

convert(varchar(50), replace(replace(dba\_name, ''''&'''', ''''&amp;''''), '''''''''''''''', ''''&apos;'''')) as dba\_name,

case when y.certification\_dt is null then ''''N/A'''' when ISNULL(p.show\_values,''''T'''') = ''''F'''' then ''''N/A'''' else ''''$'''' + convert(varchar(20), appraised\_val) end as appraised\_val,

abs\_subdv\_cd,

mapsco,

map\_id,

agent\_cd,

hood\_cd,

convert(varchar(50), replace(replace(hood\_name, ''''&'''', ''''&amp;''''), '''''''''''''''', ''''&apos;'''')) as hood\_name,

convert(varchar(80), replace(replace(owner\_name, ''''&'''', ''''&amp;''''), '''''''''''''''', ''''&apos;'''')) as owner\_name,

owner\_id,

pct\_ownership,

exemptions,

state\_cd,

convert(varchar(255), replace(replace(replace(replace(legal\_desc, ''''&'''', ''''&amp;''''), '''''''''''''''', ''''&apos;''''), ''''"'''', ''''&quot;''''), ''''<'''', ''''&lt;'''')) as legal\_desc,

convert(varchar(255), replace(replace(replace(situs\_display, char(10), ''''''''), char(13), '''' ''''), '''' '''', '''' '''')) as situs,

jurisdictions,

convert(varchar(80), replace(replace(addr\_line1, ''''&'''', ''''&amp;''''), '''''''''''''''', ''''&apos;'''')) as owner\_address1,

convert(varchar(80), replace(replace(addr\_line2, ''''&'''', ''''&amp;''''), '''''''''''''''', ''''&apos;'''')) as owner\_address2,

convert(varchar(80), replace(replace(addr\_line3, ''''&'''', ''''&amp;''''), '''''''''''''''', ''''&apos;'''')) as owner\_address3,

convert(varchar(80), replace(replace(addr\_city, ''''&'''', ''''&amp;''''), '''''''''''''''', ''''&apos;'''')) as city,

convert(varchar(20), replace(replace(addr\_state, ''''&'''', ''''&amp;''''), '''''''''''''''', ''''&apos;'''')) as state,

convert(varchar(20), replace(replace(addr\_zip, ''''&'''', ''''&amp;''''), '''''''''''''''', ''''&apos;'''')) as zip,

convert(varchar(20), rtrim(replace(replace(country\_cd, ''''&'''', ''''&amp;''''), '''''''''''''''', ''''&apos;''''))) as country

from \_clientdb\_property as p with (nolock)

join \_clientdb\_pacs\_year as y with (nolock)

on p.prop\_val\_yr = y.tax\_yr '''

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = 'clientdb\_exmpt\_type\_vw'

set @sql = @input\_database\_name + '.dbo.sp\_executesql N''

create view dbo.clientdb\_exmpt\_type\_vw

as

select rtrim(exmpt\_type\_cd) as code, exmpt\_desc as description

from ' + @input\_database\_name + '.dbo.\_clientdb\_exmpt\_type with (nolock) '''

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = '[fee\_property\_vw]'

set @sql = @input\_database\_name + '.dbo.sp\_executesql N''

create view [dbo].[fee\_property\_vw]

as

select bfa.fee\_id, b.prop\_id

from bill\_fee\_assoc as bfa with(nolock)

join bill as b with(nolock) on

b.bill\_id = bfa.bill\_id

union

select fee\_id, prop\_id

from fee\_prop\_assoc as fpa with(nolock)

'''

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

-- Stop: Create all tables, views and procedures

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

print 'Create all tables, views and procedures...'

if @region = @texas

begin

set @start\_date = getdate()

--Add necessary tax due procedures

declare @sp\_exists varchar(2000)

declare @sp\_text1 varchar(8000)

declare @sp\_text2 varchar(8000)

declare @sp\_text3 varchar(8000)

declare @sp\_text4 varchar(8000)

declare @sp\_text5 varchar(8000)

declare @sp\_text6 varchar(8000)

declare @sp\_text7 varchar(8000)

declare @sp\_text8 varchar(8000)

declare @sp\_text9 varchar(8000)

declare @sp\_text10 varchar(8000)

declare @procedure\_index int

set @status = 'GetPenaltyInterestRate'

set @sp\_text1 = ''

set @sp\_text2 = ''

set @sp\_text3 = ''

set @sp\_text4 = ''

set @sp\_text5 = ''

set @sp\_text6 = ''

set @sp\_text7 = ''

set @sp\_text8 = ''

set @sp\_text9 = ''

set @sp\_text10 = ''

select @sp\_text1 = case when colid = 1 then text else isnull(@sp\_text1, '') end,

@sp\_text2 = case when colid = 2 then text else isnull(@sp\_text2, '') end,

@sp\_text3 = case when colid = 3 then text else isnull(@sp\_text3, '') end,

@sp\_text4 = case when colid = 4 then text else isnull(@sp\_text4, '') end,

@sp\_text5 = case when colid = 5 then text else isnull(@sp\_text5, '') end,

@sp\_text6 = case when colid = 6 then text else isnull(@sp\_text6, '') end,

@sp\_text7 = case when colid = 7 then text else isnull(@sp\_text7, '') end,

@sp\_text8 = case when colid = 8 then text else isnull(@sp\_text8, '') end,

@sp\_text9 = case when colid = 9 then text else isnull(@sp\_text9, '') end,

@sp\_text10 = case when colid = 10 then text else isnull(@sp\_text10, '') end

from syscomments

with (nolock)

where object\_name(id) = 'GetPenaltyInterestRate'

order by colid

set @sp\_text1 = replace(@sp\_text1, '''', '''''')

set @sp\_text2 = replace(@sp\_text2, '''', '''''')

set @sp\_text3 = replace(@sp\_text3, '''', '''''')

set @sp\_text4 = replace(@sp\_text4, '''', '''''')

set @sp\_text5 = replace(@sp\_text5, '''', '''''')

set @sp\_text6 = replace(@sp\_text6, '''', '''''')

set @sp\_text7 = replace(@sp\_text7, '''', '''''')

set @sp\_text8 = replace(@sp\_text8, '''', '''''')

set @sp\_text9 = replace(@sp\_text9, '''', '''''')

set @sp\_text10 = replace(@sp\_text10, '''', '''''')

set @sp\_exists = 'if exists(select id from syscomments where object\_name(id) = ''''GetPenaltyInterestRate'''')' + char(13) + char(10)

set @sp\_exists = @sp\_exists + 'drop procedure [GetPenaltyInterestRate]' + char(13) + char(10)

set @sql = @input\_database\_name + '.dbo.sp\_executesql N'''

exec(@sql + @sp\_exists + '''')

set @sql = @input\_database\_name + '.dbo.sp\_executesql N'''

exec(@sql + @sp\_text1 + @sp\_text2 + @sp\_text3 + @sp\_text4 + @sp\_text5 + @sp\_text6 +

@sp\_text7 + @sp\_text8 + @sp\_text9 + @sp\_text10 + '''')

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = 'GetQHBillPenaltyInterest'

set @sp\_text1 = ''

set @sp\_text2 = ''

set @sp\_text3 = ''

set @sp\_text4 = ''

set @sp\_text5 = ''

set @sp\_text6 = ''

set @sp\_text7 = ''

set @sp\_text8 = ''

set @sp\_text9 = ''

set @sp\_text10 = ''

select @sp\_text1 = case when colid = 1 then text else isnull(@sp\_text1, '') end,

@sp\_text2 = case when colid = 2 then text else isnull(@sp\_text2, '') end,

@sp\_text3 = case when colid = 3 then text else isnull(@sp\_text3, '') end,

@sp\_text4 = case when colid = 4 then text else isnull(@sp\_text4, '') end,

@sp\_text5 = case when colid = 5 then text else isnull(@sp\_text5, '') end,

@sp\_text6 = case when colid = 6 then text else isnull(@sp\_text6, '') end,

@sp\_text7 = case when colid = 7 then text else isnull(@sp\_text7, '') end,

@sp\_text8 = case when colid = 8 then text else isnull(@sp\_text8, '') end,

@sp\_text9 = case when colid = 9 then text else isnull(@sp\_text9, '') end,

@sp\_text10 = case when colid = 10 then text else isnull(@sp\_text10, '') end

from syscomments

with (nolock)

where object\_name(id) = 'GetQHBillPenaltyInterest'

order by colid

set @sp\_text1 = replace(@sp\_text1, '''', '''''')

set @sp\_text2 = replace(@sp\_text2, '''', '''''')

set @sp\_text3 = replace(@sp\_text3, '''', '''''')

set @sp\_text4 = replace(@sp\_text4, '''', '''''')

set @sp\_text5 = replace(@sp\_text5, '''', '''''')

set @sp\_text6 = replace(@sp\_text6, '''', '''''')

set @sp\_text7 = replace(@sp\_text7, '''', '''''')

set @sp\_text8 = replace(@sp\_text8, '''', '''''')

set @sp\_text9 = replace(@sp\_text9, '''', '''''')

set @sp\_text10 = replace(@sp\_text10, '''', '''''')

set @sp\_exists = 'if exists(select id from syscomments where object\_name(id) = ''''GetQHBillPenaltyInterest'''')' + char(13) + char(10)

set @sp\_exists = @sp\_exists + 'drop procedure [GetQHBillPenaltyInterest]' + char(13) + char(10)

set @sql = @input\_database\_name + '.dbo.sp\_executesql N'''

exec(@sql + @sp\_exists + '''')

set @sql = @input\_database\_name + '.dbo.sp\_executesql N'''

exec(@sql + @sp\_text1 + @sp\_text2 + @sp\_text3 + @sp\_text4 + @sp\_text5 + @sp\_text6 +

@sp\_text7 + @sp\_text8 + @sp\_text9 + @sp\_text10 + '''')

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = 'GetPenaltyInterest'

set @sp\_text1 = ''

set @sp\_text2 = ''

set @sp\_text3 = ''

set @sp\_text4 = ''

set @sp\_text5 = ''

set @sp\_text6 = ''

set @sp\_text7 = ''

set @sp\_text8 = ''

set @sp\_text9 = ''

set @sp\_text10 = ''

select @sp\_text1 = case when colid = 1 then text else isnull(@sp\_text1, '') end,

@sp\_text2 = case when colid = 2 then text else isnull(@sp\_text2, '') end,

@sp\_text3 = case when colid = 3 then text else isnull(@sp\_text3, '') end,

@sp\_text4 = case when colid = 4 then text else isnull(@sp\_text4, '') end,

@sp\_text5 = case when colid = 5 then text else isnull(@sp\_text5, '') end,

@sp\_text6 = case when colid = 6 then text else isnull(@sp\_text6, '') end,

@sp\_text7 = case when colid = 7 then text else isnull(@sp\_text7, '') end,

@sp\_text8 = case when colid = 8 then text else isnull(@sp\_text8, '') end,

@sp\_text9 = case when colid = 9 then text else isnull(@sp\_text9, '') end,

@sp\_text10 = case when colid = 10 then text else isnull(@sp\_text10, '') end

from syscomments

with (nolock)

where object\_name(id) = 'GetPenaltyInterest'

order by colid

set @sp\_text1 = replace(@sp\_text1, '''', '''''')

set @sp\_text2 = replace(@sp\_text2, '''', '''''')

set @sp\_text3 = replace(@sp\_text3, '''', '''''')

set @sp\_text4 = replace(@sp\_text4, '''', '''''')

set @sp\_text5 = replace(@sp\_text5, '''', '''''')

set @sp\_text6 = replace(@sp\_text6, '''', '''''')

set @sp\_text7 = replace(@sp\_text7, '''', '''''')

set @sp\_text8 = replace(@sp\_text8, '''', '''''')

set @sp\_text9 = replace(@sp\_text9, '''', '''''')

set @sp\_text10 = replace(@sp\_text10, '''', '''''')

set @sp\_exists = 'if exists(select id from syscomments where object\_name(id) = ''''GetPenaltyInterest'''')' + char(13) + char(10)

set @sp\_exists = @sp\_exists + 'drop procedure [GetPenaltyInterest]' + char(13) + char(10)

set @sql = @input\_database\_name + '.dbo.sp\_executesql N'''

exec(@sql + @sp\_exists + '''')

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @sql = @input\_database\_name + '.dbo.sp\_executesql N'''

exec(@sql + @sp\_text1 + @sp\_text2 + @sp\_text3 + @sp\_text4 + @sp\_text5 + @sp\_text6 +

@sp\_text7 + @sp\_text8 + @sp\_text9 + @sp\_text10 + '''')

set @status = 'GetBillTaxDue'

set @sp\_text1 = ''

set @sp\_text2 = ''

set @sp\_text3 = ''

set @sp\_text4 = ''

set @sp\_text5 = ''

set @sp\_text6 = ''

set @sp\_text7 = ''

set @sp\_text8 = ''

set @sp\_text9 = ''

set @sp\_text10 = ''

select @sp\_text1 = case when colid = 1 then text else isnull(@sp\_text1, '') end,

@sp\_text2 = case when colid = 2 then text else isnull(@sp\_text2, '') end,

@sp\_text3 = case when colid = 3 then text else isnull(@sp\_text3, '') end,

@sp\_text4 = case when colid = 4 then text else isnull(@sp\_text4, '') end,

@sp\_text5 = case when colid = 5 then text else isnull(@sp\_text5, '') end,

@sp\_text6 = case when colid = 6 then text else isnull(@sp\_text6, '') end,

@sp\_text7 = case when colid = 7 then text else isnull(@sp\_text7, '') end,

@sp\_text8 = case when colid = 8 then text else isnull(@sp\_text8, '') end,

@sp\_text9 = case when colid = 9 then text else isnull(@sp\_text9, '') end,

@sp\_text10 = case when colid = 10 then text else isnull(@sp\_text10, '') end

from syscomments

with (nolock)

where object\_name(id) = 'GetBillTaxDue'

order by colid

set @sp\_text1 = replace(@sp\_text1, '''', '''''')

set @sp\_text2 = replace(@sp\_text2, '''', '''''')

set @sp\_text3 = replace(@sp\_text3, '''', '''''')

set @sp\_text4 = replace(@sp\_text4, '''', '''''')

set @sp\_text5 = replace(@sp\_text5, '''', '''''')

set @sp\_text6 = replace(@sp\_text6, '''', '''''')

set @sp\_text7 = replace(@sp\_text7, '''', '''''')

set @sp\_text8 = replace(@sp\_text8, '''', '''''')

set @sp\_text9 = replace(@sp\_text9, '''', '''''')

set @sp\_text10 = replace(@sp\_text10, '''', '''''')

set @sp\_exists = 'if exists(select id from syscomments where object\_name(id) = ''''GetBillTaxDue'''')' + char(13) + char(10)

set @sp\_exists = @sp\_exists + 'drop procedure [GetBillTaxDue]' + char(13) + char(10)

set @sql = @input\_database\_name + '.dbo.sp\_executesql N'''

exec(@sql + @sp\_exists + '''')

set @sql = @input\_database\_name + '.dbo.sp\_executesql N'''

exec(@sql + @sp\_text1 + @sp\_text2 + @sp\_text3 + @sp\_text4 + @sp\_text5 + @sp\_text6 +

@sp\_text7 + @sp\_text8 + @sp\_text9 + @sp\_text10 + '''')

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = 'PopulatePropertyAccessBills'

set @sp\_text1 = ''

set @sp\_text2 = ''

set @sp\_text3 = ''

set @sp\_text4 = ''

set @sp\_text5 = ''

set @sp\_text6 = ''

set @sp\_text7 = ''

set @sp\_text8 = ''

set @sp\_text9 = ''

set @sp\_text10 = ''

select @sp\_text1 = case when colid = 1 then text else isnull(@sp\_text1, '') end,

@sp\_text2 = case when colid = 2 then text else isnull(@sp\_text2, '') end,

@sp\_text3 = case when colid = 3 then text else isnull(@sp\_text3, '') end,

@sp\_text4 = case when colid = 4 then text else isnull(@sp\_text4, '') end,

@sp\_text5 = case when colid = 5 then text else isnull(@sp\_text5, '') end,

@sp\_text6 = case when colid = 6 then text else isnull(@sp\_text6, '') end,

@sp\_text7 = case when colid = 7 then text else isnull(@sp\_text7, '') end,

@sp\_text8 = case when colid = 8 then text else isnull(@sp\_text8, '') end,

@sp\_text9 = case when colid = 9 then text else isnull(@sp\_text9, '') end,

@sp\_text10 = case when colid = 10 then text else isnull(@sp\_text10, '') end

from syscomments

with (nolock)

where object\_name(id) = 'PopulatePropertyAccessBills'

order by colid

set @sp\_text1 = replace(@sp\_text1, '''', '''''')

set @sp\_text2 = replace(@sp\_text2, '''', '''''')

set @sp\_text3 = replace(@sp\_text3, '''', '''''')

set @sp\_text4 = replace(@sp\_text4, '''', '''''')

set @sp\_text5 = replace(@sp\_text5, '''', '''''')

set @sp\_text6 = replace(@sp\_text6, '''', '''''')

set @sp\_text7 = replace(@sp\_text7, '''', '''''')

set @sp\_text8 = replace(@sp\_text8, '''', '''''')

set @sp\_text9 = replace(@sp\_text9, '''', '''''')

set @sp\_text10 = replace(@sp\_text10, '''', '''''')

set @sp\_exists = 'if exists(select id from syscomments where object\_name(id) = ''''PopulatePropertyAccessBills'''')' + char(13) + char(10)

set @sp\_exists = @sp\_exists + 'drop procedure [PopulatePropertyAccessBills]' + char(13) + char(10)

set @sql = @input\_database\_name + '.dbo.sp\_executesql N'''

exec(@sql + @sp\_exists + '''')

set @sql = @input\_database\_name + '.dbo.sp\_executesql N'''

exec(@sql + @sp\_text1 + @sp\_text2 + @sp\_text3 + @sp\_text4 + @sp\_text5 + @sp\_text6 +

@sp\_text7 + @sp\_text8 + @sp\_text9 + @sp\_text10 + '''')

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

end -- only for texas

if @region = @washington

begin

set @start\_date = getdate()

set @status = 'GetUniqueID'

--Add necessary tax due procedures

set @sp\_text1 = ''

set @sp\_text2 = ''

set @sp\_text3 = ''

set @sp\_text4 = ''

set @sp\_text5 = ''

set @sp\_text6 = ''

set @sp\_text7 = ''

set @sp\_text8 = ''

set @sp\_text9 = ''

set @sp\_text10 = ''

select @sp\_text1 = case when colid = 1 then text else isnull(@sp\_text1, '') end,

@sp\_text2 = case when colid = 2 then text else isnull(@sp\_text2, '') end,

@sp\_text3 = case when colid = 3 then text else isnull(@sp\_text3, '') end,

@sp\_text4 = case when colid = 4 then text else isnull(@sp\_text4, '') end,

@sp\_text5 = case when colid = 5 then text else isnull(@sp\_text5, '') end,

@sp\_text6 = case when colid = 6 then text else isnull(@sp\_text6, '') end,

@sp\_text7 = case when colid = 7 then text else isnull(@sp\_text7, '') end,

@sp\_text8 = case when colid = 8 then text else isnull(@sp\_text8, '') end,

@sp\_text9 = case when colid = 9 then text else isnull(@sp\_text9, '') end,

@sp\_text10 = case when colid = 10 then text else isnull(@sp\_text10, '') end

from syscomments

with (nolock)

where object\_name(id) = 'GetUniqueID'

order by colid

set @sp\_text1 = replace(@sp\_text1, '''', '''''')

set @sp\_text2 = replace(@sp\_text2, '''', '''''')

set @sp\_text3 = replace(@sp\_text3, '''', '''''')

set @sp\_text4 = replace(@sp\_text4, '''', '''''')

set @sp\_text5 = replace(@sp\_text5, '''', '''''')

set @sp\_text6 = replace(@sp\_text6, '''', '''''')

set @sp\_text7 = replace(@sp\_text7, '''', '''''')

set @sp\_text8 = replace(@sp\_text8, '''', '''''')

set @sp\_text9 = replace(@sp\_text9, '''', '''''')

set @sp\_text10 = replace(@sp\_text10, '''', '''''')

set @sp\_exists = 'if exists(select id from syscomments where object\_name(id) = ''''GetUniqueID'''')' + char(13) + char(10)

set @sp\_exists = @sp\_exists + 'drop procedure [GetUniqueID]' + char(13) + char(10)

set @sql = @input\_database\_name + '.dbo.sp\_executesql N'''

exec(@sql + @sp\_exists + '''')

set @sql = @input\_database\_name + '.dbo.sp\_executesql N'''

exec(@sql + @sp\_text1 + @sp\_text2 + @sp\_text3 + @sp\_text4 + @sp\_text5 + @sp\_text6 +

@sp\_text7 + @sp\_text8 + @sp\_text9 + @sp\_text10 + '''')

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

--GetUniqueIDRS

set @status = 'GetUniqueIDRS'

set @sp\_text1 = ''

set @sp\_text2 = ''

set @sp\_text3 = ''

set @sp\_text4 = ''

set @sp\_text5 = ''

set @sp\_text6 = ''

set @sp\_text7 = ''

set @sp\_text8 = ''

set @sp\_text9 = ''

set @sp\_text10 = ''

select @sp\_text1 = case when colid = 1 then text else isnull(@sp\_text1, '') end,

@sp\_text2 = case when colid = 2 then text else isnull(@sp\_text2, '') end,

@sp\_text3 = case when colid = 3 then text else isnull(@sp\_text3, '') end,

@sp\_text4 = case when colid = 4 then text else isnull(@sp\_text4, '') end,

@sp\_text5 = case when colid = 5 then text else isnull(@sp\_text5, '') end,

@sp\_text6 = case when colid = 6 then text else isnull(@sp\_text6, '') end,

@sp\_text7 = case when colid = 7 then text else isnull(@sp\_text7, '') end,

@sp\_text8 = case when colid = 8 then text else isnull(@sp\_text8, '') end,

@sp\_text9 = case when colid = 9 then text else isnull(@sp\_text9, '') end,

@sp\_text10 = case when colid = 10 then text else isnull(@sp\_text10, '') end

from syscomments

with (nolock)

where object\_name(id) = 'GetUniqueIDRS'

order by colid

set @sp\_text1 = replace(@sp\_text1, '''', '''''')

set @sp\_text2 = replace(@sp\_text2, '''', '''''')

set @sp\_text3 = replace(@sp\_text3, '''', '''''')

set @sp\_text4 = replace(@sp\_text4, '''', '''''')

set @sp\_text5 = replace(@sp\_text5, '''', '''''')

set @sp\_text6 = replace(@sp\_text6, '''', '''''')

set @sp\_text7 = replace(@sp\_text7, '''', '''''')

set @sp\_text8 = replace(@sp\_text8, '''', '''''')

set @sp\_text9 = replace(@sp\_text9, '''', '''''')

set @sp\_text10 = replace(@sp\_text10, '''', '''''')

set @sp\_exists = 'if exists(select id from syscomments where object\_name(id) = ''''GetUniqueIDRS'''')' + char(13) + char(10)

set @sp\_exists = @sp\_exists + 'drop procedure [GetUniqueIDRS]' + char(13) + char(10)

set @sql = @input\_database\_name + '.dbo.sp\_executesql N'''

exec(@sql + @sp\_exists + '''')

set @sql = @input\_database\_name + '.dbo.sp\_executesql N'''

exec(@sql + @sp\_text1 + @sp\_text2 + @sp\_text3 + @sp\_text4 + @sp\_text5 + @sp\_text6 +

@sp\_text7 + @sp\_text8 + @sp\_text9 + @sp\_text10 + '''')

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

-- Sales Search

set @status = 'PA\_GetSaleInfo'

set @sp\_text1 = ''

set @sp\_text2 = ''

set @sp\_text3 = ''

set @sp\_text4 = ''

set @sp\_text5 = ''

set @sp\_text6 = ''

set @sp\_text7 = ''

set @sp\_text8 = ''

set @sp\_text9 = ''

set @sp\_text10 = ''

select @sp\_text1 = case when colid = 1 then text else isnull(@sp\_text1, '') end,

@sp\_text2 = case when colid = 2 then text else isnull(@sp\_text2, '') end,

@sp\_text3 = case when colid = 3 then text else isnull(@sp\_text3, '') end,

@sp\_text4 = case when colid = 4 then text else isnull(@sp\_text4, '') end,

@sp\_text5 = case when colid = 5 then text else isnull(@sp\_text5, '') end,

@sp\_text6 = case when colid = 6 then text else isnull(@sp\_text6, '') end,

@sp\_text7 = case when colid = 7 then text else isnull(@sp\_text7, '') end,

@sp\_text8 = case when colid = 8 then text else isnull(@sp\_text8, '') end,

@sp\_text9 = case when colid = 9 then text else isnull(@sp\_text9, '') end,

@sp\_text10 = case when colid = 10 then text else isnull(@sp\_text10, '') end

from syscomments

with (nolock)

where object\_name(id) = 'PA\_GetSaleInfo'

order by colid

set @sp\_text1 = replace(@sp\_text1, '''', '''''')

set @sp\_text2 = replace(@sp\_text2, '''', '''''')

set @sp\_text3 = replace(@sp\_text3, '''', '''''')

set @sp\_text4 = replace(@sp\_text4, '''', '''''')

set @sp\_text5 = replace(@sp\_text5, '''', '''''')

set @sp\_text6 = replace(@sp\_text6, '''', '''''')

set @sp\_text7 = replace(@sp\_text7, '''', '''''')

set @sp\_text8 = replace(@sp\_text8, '''', '''''')

set @sp\_text9 = replace(@sp\_text9, '''', '''''')

set @sp\_text10 = replace(@sp\_text10, '''', '''''')

set @sp\_exists = 'if exists(select id from syscomments where object\_name(id) = ''''PA\_GetSaleInfo'''')' + char(13) + char(10)

set @sp\_exists = @sp\_exists + 'drop procedure [PA\_GetSaleInfo]' + char(13) + char(10)

set @sql = @input\_database\_name + '.dbo.sp\_executesql N'''

exec(@sql + @sp\_exists + '''')

set @sql = @input\_database\_name + '.dbo.sp\_executesql N'''

exec(@sql + @sp\_text1 + @sp\_text2 + @sp\_text3 + @sp\_text4 + @sp\_text5 + @sp\_text6 +

@sp\_text7 + @sp\_text8 + @sp\_text9 + @sp\_text10 + '''')

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = 'fn\_ReturnTableFromCommaSepValues'

set @sp\_text1 = ''

set @sp\_text2 = ''

set @sp\_text3 = ''

set @sp\_text4 = ''

set @sp\_text5 = ''

set @sp\_text6 = ''

set @sp\_text7 = ''

set @sp\_text8 = ''

set @sp\_text9 = ''

set @sp\_text10 = ''

select @sp\_text1 = case when colid = 1 then text else isnull(@sp\_text1, '') end,

@sp\_text2 = case when colid = 2 then text else isnull(@sp\_text2, '') end,

@sp\_text3 = case when colid = 3 then text else isnull(@sp\_text3, '') end,

@sp\_text4 = case when colid = 4 then text else isnull(@sp\_text4, '') end,

@sp\_text5 = case when colid = 5 then text else isnull(@sp\_text5, '') end,

@sp\_text6 = case when colid = 6 then text else isnull(@sp\_text6, '') end,

@sp\_text7 = case when colid = 7 then text else isnull(@sp\_text7, '') end,

@sp\_text8 = case when colid = 8 then text else isnull(@sp\_text8, '') end,

@sp\_text9 = case when colid = 9 then text else isnull(@sp\_text9, '') end,

@sp\_text10 = case when colid = 10 then text else isnull(@sp\_text10, '') end

from syscomments

with (nolock)

where object\_name(id) = 'fn\_ReturnTableFromCommaSepValues'

order by colid

set @sp\_text1 = replace(@sp\_text1, '''', '''''')

set @sp\_text2 = replace(@sp\_text2, '''', '''''')

set @sp\_text3 = replace(@sp\_text3, '''', '''''')

set @sp\_text4 = replace(@sp\_text4, '''', '''''')

set @sp\_text5 = replace(@sp\_text5, '''', '''''')

set @sp\_text6 = replace(@sp\_text6, '''', '''''')

set @sp\_text7 = replace(@sp\_text7, '''', '''''')

set @sp\_text8 = replace(@sp\_text8, '''', '''''')

set @sp\_text9 = replace(@sp\_text9, '''', '''''')

set @sp\_text10 = replace(@sp\_text10, '''', '''''')

set @sp\_exists = 'if exists(select id from syscomments where object\_name(id) = ''''fn\_ReturnTableFromCommaSepValues'''')' + char(13) + char(10)

set @sp\_exists = @sp\_exists + 'drop function [fn\_ReturnTableFromCommaSepValues]' + char(13) + char(10)

set @sql = @input\_database\_name + '.dbo.sp\_executesql N'''

exec(@sql + @sp\_exists + '''')

set @sql = @input\_database\_name + '.dbo.sp\_executesql N'''

exec(@sql + @sp\_text1 + @sp\_text2 + @sp\_text3 + @sp\_text4 + @sp\_text5 + @sp\_text6 +

@sp\_text7 + @sp\_text8 + @sp\_text9 + @sp\_text10 + '''')

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

--Sales Search

--Create a Fake PopulatePropertyAccessBills sp

--Create a Fake PopulatePropertyAccessBills sp

set @status = 'Fake PopulatePropertyAccessBills'

set @sql = '

if exists (

select name

from sysobjects

WHERE id = OBJECT\_ID(''PopulatePropertyAccessBills''))

begin

drop procedure PopulatePropertyAccessBills

end'

exec(@sql)

set @sql = '

CREATE PROCEDURE [dbo].[PopulatePropertyAccessBills]

@input\_prop\_id int,

@input\_effective\_date varchar(10),

@input\_year int = 0,

@input\_total\_due bit = 0

AS

set nocount on

'

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

-- BuildSQLBill (modified PA version)

set @status = 'BuildSQLBill (PA)'

set @sql = @input\_database\_name + '.dbo.sp\_executesql N''

IF EXISTS (SELECT \* FROM sys.objects WHERE object\_id = OBJECT\_ID(N''''[dbo].[BuildSQLBill]'''') AND type in (N''''P'''', N''''PC''''))

BEGIN

DROP PROCEDURE BuildSQLBill

END

'''

exec(@sql)

set @sql = @input\_database\_name + '.dbo.sp\_executesql N''

-- Alternate version of BuildSQLBill for SP Property Access.

-- Do not deploy this SP normally. SP ExportPropertyAccess will create it in PA databases.

CREATE PROCEDURE dbo.BuildSQLBill

@dataset\_id int,

@segment\_id int

AS

set nocount on

if object\_id(''''tempdb..#result'''') is not null

drop table #result

if object\_id(''''tempdb..#prop\_use'''') is not null

drop table #prop\_use

if object\_id(''''tempdb..#prop\_detail'''') is not null

drop table #prop\_detail

create table #result

(

bill\_id int primary key,

year numeric(4,0),

prop\_id int,

current\_amount\_due numeric(14,2),

statement\_id int,

is\_partial char(1),

owner\_id int,

prop\_type\_cd char(5),

tax\_district\_id int,

rollback\_id int,

agency\_id int,

payout\_agreement\_id int,

use\_bond\_interest bit,

override\_penalty\_and\_interest bit,

effective\_due\_date datetime,

bond\_interest\_percentage numeric(14,4),

bond\_interest\_begin\_date datetime,

bond\_interest\_end\_date datetime,

code varchar(10),

bond\_interest\_frequency varchar(10),

payment\_terms\_type\_cd varchar(10),

is\_leased\_land\_property bit,

calc\_penalty\_on\_bond\_interest bit,

calc\_interest\_on\_bond\_interest bit,

penalty\_interest\_property\_type\_cd varchar(10),

priority int

)

create index py on #result (prop\_id, year)

create table #prop\_use

(

prop\_id int not null,

year numeric(4,0) not null,

penalty\_interest\_property\_type\_cd varchar(10) not null,

primary key (prop\_id, year, penalty\_interest\_property\_type\_cd)

)

create table #prop\_detail

(

prop\_id int not null,

year numeric(4,0) not null,

penalty\_interest\_property\_type\_cd varchar(10) not null,

primary key (prop\_id, year, penalty\_interest\_property\_type\_cd)

)

-- Everything else but the PIP codes

insert #result

(bill\_id, year, prop\_id, current\_amount\_due, statement\_id, is\_partial, owner\_id, prop\_type\_cd, tax\_district\_id,

rollback\_id, agency\_id, payout\_agreement\_id, use\_bond\_interest, override\_penalty\_and\_interest, effective\_due\_date,

bond\_interest\_percentage, bond\_interest\_begin\_date, bond\_interest\_end\_date, code, bond\_interest\_frequency,

payment\_terms\_type\_cd, is\_leased\_land\_property, calc\_penalty\_on\_bond\_interest, calc\_interest\_on\_bond\_interest,

penalty\_interest\_property\_type\_cd, priority)

select

b.bill\_id, b.year, b.prop\_id, b.current\_amount\_due, isnull(b.statement\_id, 0) statement\_id,

case when ab.bill\_id is not null then

case when ppra.payout\_agreement\_id is not null then ''''F''''

when isNull(saso.eligible\_for\_partial\_pay, 0) = 1 and b.effective\_due\_date > ''''4/25/2023'''' then ''''T''''

when isnull(bfc.partial\_payment\_indicator, ''''F'''') = ''''T'''' then ''''T''''

when isnull(pc.szConfigValue,0) = 1 then ''''T''''

else ''''F''''

end

when isnull(bfc.partial\_payment\_indicator, ''''F'''') = ''''T'''' then ''''T''''

when isnull(pc.szConfigValue,0) = 1 and ppra.payout\_agreement\_id is null then ''''T''''

else ''''F''''

end is\_partial,

isnull(statement\_owner.owner\_id, isnull(p.col\_owner\_id, 0)) owner\_id, p.prop\_type\_cd,

lb.tax\_district\_id, b.rollback\_id, ab.agency\_id, paba.payout\_agreement\_id,

isnull(pa.use\_bond\_interest, 0) use\_bond\_interest, isnull(pa.override\_penalty\_and\_interest, 0) override\_penalty\_and\_interest,

b.effective\_due\_date, pa.bond\_interest\_percentage,

pa.bond\_interest\_begin\_date, pa.bond\_interest\_end\_date, b.code,pa.bond\_interest\_frequency, pa.payment\_terms\_type\_cd,

isnull(pll.is\_leased\_land\_property, 0) is\_leased\_land\_property,

pa.calc\_penalty\_on\_bond\_interest, pa.calc\_interest\_on\_bond\_interest,

null, -- penalty\_interest\_property\_type\_cd

99999 -- priority, lowest number wins

from tax\_due\_calc\_list as tdcl with(nolock)

join bill as b with(nolock) on

b.bill\_id = tdcl.tax\_due\_id

join property as p with(nolock) on

p.prop\_id = b.prop\_id

left outer join levy\_bill as lb with(nolock) on

lb.bill\_id = b.bill\_id

left outer join assessment\_bill as ab with(nolock) on

ab.bill\_id = b.bill\_id

left outer join (

select bill\_id, max(payout\_agreement\_id) as payout\_agreement\_id

from payout\_agreement\_bill\_assoc with(Nolock)

group by bill\_id

) as paba on

paba.bill\_id = b.bill\_id

left join (

select max(payout\_agreement\_id) payout\_agreement\_id, year, sup\_num, prop\_id, is\_primary from property\_payout\_agreement with(nolock) group by year, sup\_num, prop\_id, is\_primary

) ppra on

p.prop\_id = ppra.prop\_id and b.year = ppra.year and b.sup\_num = ppra.sup\_num

left outer join payout\_agreement as pa with(nolock) on

pa.payout\_agreement\_id = paba.payout\_agreement\_id

left outer join bill\_fee\_code as bfc with(nolock) on

bfc.bill\_fee\_cd = b.code

left outer join special\_assessment\_statement\_options as saso with (nolock) on

saso.agency\_id = ab.agency\_id and

saso.year = ab.year

left join pacs\_config pc on

pc.szgroup = ''''Payment'''' and szConfigName = ''''Allow Universal Partial Payments''''

left outer join property\_leased\_land\_vw as pll with(nolock) on

pll.prop\_id = p.prop\_id and pll.prop\_val\_yr = b.year

outer apply (

select max(wts.run\_id) max\_run\_id

from wa\_tax\_statement wts with(nolock)

where wts.year = b.year

and wts.prop\_id = b.prop\_id

and wts.statement\_id = b.statement\_id

) max\_run

outer apply (

select top 1 owner\_id

from wa\_tax\_statement wts with(nolock)

where wts.year = b.year

and wts.prop\_id = b.prop\_id

and wts.statement\_id = b.statement\_id

and wts.run\_id = max\_run.max\_run\_id

order by copy\_type

) statement\_owner

where tdcl.dataset\_id = @dataset\_id and tdcl.segment\_id = @segment\_id

order by b.bill\_id asc

-- HB 1410 Penalty and Interest (PIP) codes

-- personal properties

update r

set penalty\_interest\_property\_type\_cd = x.penalty\_interest\_property\_type\_cd, [priority] = x.priority

from #result r

outer apply (

select top 1 pipt.penalty\_interest\_property\_type\_cd, pipt.priority

from \_clientdb\_property pv with(nolock)

join property p with(nolock)

on p.prop\_id = pv.prop\_id

join penalty\_interest\_property\_type pipt with(nolock)

on pipt.personal = 1

join penalty\_and\_interest pai with(nolock)

on pai.penalty\_interest\_property\_type\_cd = pipt.penalty\_interest\_property\_type\_cd

and pai.year = pv.prop\_val\_yr

where pv.prop\_id = r.prop\_id

and pv.prop\_val\_yr = r.year

and ((r.tax\_district\_id is null) or (pai.ref\_id = r.tax\_district\_id))

and p.prop\_type\_cd in (''''P'''',''''MN'''')

and pv.prop\_id = r.prop\_id

and pv.prop\_val\_yr = r.year

order by pipt.priority

)x

where x.priority is not null

-- vacant properties

update r

set penalty\_interest\_property\_type\_cd = x.penalty\_interest\_property\_type\_cd, [priority] = x.priority

from #result r

outer apply (

select top 1 pipt.penalty\_interest\_property\_type\_cd, pipt.priority

from \_clientdb\_property pv with(nolock)

join property p with(nolock)

on p.prop\_id = pv.prop\_id

join property\_use pu with(nolock) on

(pu.property\_use\_cd = pv.property\_use\_cd or pu.property\_use\_cd = pv.secondary\_use\_cd)

join dor\_use\_code duc with(nolock)

on duc.sub\_cd = pu.dor\_use\_code

join penalty\_interest\_property\_type\_dor\_use\_code piptduc with(nolock)

on piptduc.sub\_cd = duc.sub\_cd

join penalty\_interest\_property\_type pipt with(nolock)

on pipt.penalty\_interest\_property\_type\_cd = piptduc.penalty\_interest\_property\_type\_cd

and pipt.personal = 0

join penalty\_and\_interest pai with(nolock)

on pai.penalty\_interest\_property\_type\_cd = pipt.penalty\_interest\_property\_type\_cd

and pai.year = pv.prop\_val\_yr

where pv.prop\_id = r.prop\_id

and pv.prop\_val\_yr = r.year

and ((r.tax\_district\_id is null) or (pai.ref\_id = r.tax\_district\_id))

and p.prop\_type\_cd not in (''''P'''',''''MN'''')

and not exists (

select 1 from \_clientdb\_improvement\_building\_detail id with(nolock)

where id.prop\_id = pv.prop\_id

and id.prop\_val\_yr = pv.prop\_val\_yr

and id.imprv\_det\_id is not null

)

and pv.prop\_id = r.prop\_id

and pv.prop\_val\_yr = r.year

and pipt.priority < r.priority

order by pipt.priority

)x

where x.priority is not null

-- Properties with a matching use code, improvement detail type, and PIP type

-- PIP codes that match by property use

insert #prop\_use (prop\_id, year, penalty\_interest\_property\_type\_cd)

select distinct pv.prop\_id, pv.prop\_val\_yr year, piptduc.penalty\_interest\_property\_type\_cd

from

(

select distinct prop\_id, year

from #result

)py

join \_clientdb\_property pv with(nolock)

on pv.prop\_id = py.prop\_id

and pv.prop\_val\_yr = py.year

join property\_use pu with(nolock) on

(pu.property\_use\_cd = pv.property\_use\_cd or pu.property\_use\_cd = pv.secondary\_use\_cd)

join dor\_use\_code duc with(nolock)

on duc.sub\_cd = pu.dor\_use\_code

join penalty\_interest\_property\_type\_dor\_use\_code piptduc with(nolock)

on piptduc.sub\_cd = duc.sub\_cd

-- PIP codes that match by improvement detail

insert #prop\_detail (prop\_id, year, penalty\_interest\_property\_type\_cd)

select distinct pv.prop\_id, pv.prop\_val\_yr year, piptidtc.penalty\_interest\_property\_type\_cd

from

(

select distinct prop\_id, year

from #result

)py

join \_clientdb\_property pv with(nolock)

on pv.prop\_id = py.prop\_id

and pv.prop\_val\_yr = py.year

join \_clientdb\_improvement\_building\_detail id with(nolock)

on pv.prop\_id = id.prop\_id

and pv.prop\_val\_yr = id.prop\_val\_yr

join penalty\_interest\_property\_type\_imprv\_det\_type\_cd piptidtc with(nolock)

on piptidtc.imprv\_det\_type\_cd = id.imprv\_det\_type\_cd

-- Combine queries

update r

set penalty\_interest\_property\_type\_cd = x.penalty\_interest\_property\_type\_cd, [priority] = x.priority

from #result r

outer apply (

select top 1 pipt.penalty\_interest\_property\_type\_cd, pipt.priority

from #prop\_use pu

join #prop\_detail pd

on pd.prop\_id = pu.prop\_id

and pd.year = pu.year

and pd.penalty\_interest\_property\_type\_cd = pu.penalty\_interest\_property\_type\_cd

join penalty\_interest\_property\_type pipt with(nolock)

on pipt.penalty\_interest\_property\_type\_cd = pu.penalty\_interest\_property\_type\_cd

join penalty\_and\_interest pai with(nolock)

on pai.penalty\_interest\_property\_type\_cd = pu.penalty\_interest\_property\_type\_cd

and pai.year = pu.year

and ((r.tax\_district\_id is null) or (pai.ref\_id = r.tax\_district\_id))

where pu.prop\_id = r.prop\_id

and pu.year = r.year

and pipt.priority < r.priority

order by pipt.priority

)x

where x.priority is not null

-- output

select

bill\_id,

year,

prop\_id,

current\_amount\_due,

statement\_id,

is\_partial,

owner\_id,

prop\_type\_cd,

tax\_district\_id,

rollback\_id,

agency\_id,

payout\_agreement\_id,

use\_bond\_interest,

override\_penalty\_and\_interest,

effective\_due\_date,

bond\_interest\_percentage,

bond\_interest\_begin\_date,

bond\_interest\_end\_date,

code,

bond\_interest\_frequency,

payment\_terms\_type\_cd,

is\_leased\_land\_property,

calc\_penalty\_on\_bond\_interest,

calc\_interest\_on\_bond\_interest,

penalty\_interest\_property\_type\_cd

from #result

'''

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

end

--Copy tables to the new database

print 'Copy tables to the new database...'

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print 'Populating \_clientdb\_township\_section'

set @status = 'Copy \_clientdb\_township\_section'

set @sql = 'select distinct township\_section into ' + @input\_database\_name + '.dbo.\_clientdb\_township\_section from \_clientdb\_property'

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print 'Populating \_clientdb\_street\_name'

set @status = 'Copy \_clientdb\_street\_name'

set @sql = '

select distinct street =

ltrim(rtrim(

(case

when isnull([situs\_street\_prefx],'''') = '''' then ''''

else rtrim(ltrim([situs\_street\_prefx]))+'' ''

end) +

(case

when [situs\_street] IS NULL then ''''

else rtrim(ltrim([situs\_street]))+'' ''

end)+

(case

when [situs\_street\_sufix] IS NULL then ''''

else rtrim(ltrim([situs\_street\_sufix]))

end)

))

into '

set @sql = @sql + @input\_database\_name + '.dbo.\_clientdb\_street\_name from situs with (nolock) where situs\_street is not null'

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print 'Populating \_clientdb\_city'

set @status = 'Copy \_clientdb\_city'

set @sql = 'select distinct city = cast(situs\_city as varchar) into '

set @sql = @sql + @input\_database\_name + '.dbo.\_clientdb\_city from situs with (nolock) where situs\_city is not null'

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print 'Populating \_clientdb\_improvement\_features'

set @status = 'Copy \_clientdb\_improvement\_features'

set @sql = 'insert ' + @input\_database\_name + '.dbo.\_clientdb\_improvement\_features select \* from \_clientdb\_improvement\_features'

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print 'Populating \_clientdb\_taxing\_jurisdiction\_detail'

set @status = 'Copy \_clientdb\_taxing\_jurisdiction\_detail'

set @sql = 'insert ' + @input\_database\_name + '.dbo.\_clientdb\_taxing\_jurisdiction\_detail select \* from \_clientdb\_taxing\_jurisdiction\_detail'

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print 'Populating \_clientdb\_values\_detail'

set @status = 'Copy \_clientdb\_values\_detail'

set @sql = 'insert ' + @input\_database\_name + '.dbo.\_clientdb\_values\_detail select \* from \_clientdb\_values\_detail'

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print 'Populating \_clientdb\_pacs\_year'

set @status = 'Copy \_clientdb\_pacs\_year'

set @sql = 'insert ' + @input\_database\_name + '.dbo.\_clientdb\_pacs\_year select \* from \_clientdb\_pacs\_year'

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print 'Populating \_clientdb\_imprv\_det\_sketch'

set @status = 'Copy \_clientdb\_imprv\_det\_sketch'

set @sql = 'insert ' + @input\_database\_name + '.dbo.\_clientdb\_imprv\_det\_sketch select \* from \_clientdb\_imprv\_det\_sketch'

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print 'Populating \_clientdb\_bill'

set @status = 'Copy \_clientdb\_bill'

if @region = @texas

begin

set @sql = 'insert ' + @input\_database\_name + '.dbo.bill select \* from \_clientdb\_bill'

exec(@sql)

end

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print 'Populating \_clientdb\_tax\_rate'

set @status = 'Copy \_clientdb\_tax\_rate'

set @sql = 'insert ' + @input\_database\_name + '.dbo.tax\_rate select \* from \_clientdb\_tax\_rate'

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

if @region = @texas

begin

print 'Populating \_clientdb\_payment'

set @status = 'Copy \_clientdb\_payment'

set @sql = 'insert ' + @input\_database\_name + '.dbo.payment select \* from \_clientdb\_payment'

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

end

print 'Populating \_clientdb\_payment\_trans'

set @status = 'Copy \_clientdb\_payment\_trans'

set @sql = 'insert ' + @input\_database\_name + '.dbo.payment\_trans select \* from \_clientdb\_payment\_trans'

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print 'Populating \_clientdb\_refund\_due\_trans'

set @status = 'Copy \_clientdb\_refund\_due\_trans'

set @sql = 'insert ' + @input\_database\_name + '.dbo.refund\_due\_trans select \* from \_clientdb\_refund\_due\_trans'

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print 'Populating \_clientdb\_bill\_adjust\_code'

set @status = 'Copy \_clientdb\_bill\_adjust\_code'

set @sql = 'insert ' + @input\_database\_name + '.dbo.bill\_adjust\_code select \* from \_clientdb\_bill\_adjust\_code'

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print 'Populating \_clientdb\_entity'

set @status = 'Copy \_clientdb\_entity'

set @sql = 'insert ' + @input\_database\_name + '.dbo.entity select \* from \_clientdb\_entity'

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print 'Populating \_clientdb\_account'

set @status = 'Copy \_clientdb\_account'

set @sql = 'insert ' + @input\_database\_name + '.dbo.account select \* from \_clientdb\_account'

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

/\*\*\*\*\*\*\*\*\*\*/

--Add Indexes

set @status = 'Adding Indexes'

-- HS 55281 Kevin Lloyd -

set @sql = 'use ' + @input\_database\_name

set @sql = @sql + '

CREATE CLUSTERED INDEX IX\_\_clientdb\_pacs\_year ON dbo.\_clientdb\_pacs\_year

(

tax\_yr

) ON [PRIMARY]

CREATE CLUSTERED INDEX IX\_\_clientdb\_deed\_history\_detail ON dbo.\_clientdb\_deed\_history\_detail

(

prop\_id

) ON [PRIMARY]

CREATE CLUSTERED INDEX IX\_\_clientdb\_roll\_value\_history\_detail ON dbo.\_clientdb\_roll\_value\_history\_detail

(

prop\_val\_yr,

prop\_id

) ON [PRIMARY]

CREATE CLUSTERED INDEX IX\_\_clientdb\_land\_detail ON dbo.\_clientdb\_land\_detail

(

prop\_val\_yr,

prop\_id

) ON [PRIMARY]

CREATE CLUSTERED INDEX IX\_\_clientdb\_imprv\_det\_sketch ON dbo.\_clientdb\_imprv\_det\_sketch

(

prop\_val\_yr,

prop\_id

) ON [PRIMARY]

CREATE CLUSTERED INDEX IX\_\_clientdb\_improvement\_building\_detail ON dbo.\_clientdb\_improvement\_building\_detail

(

prop\_val\_yr,

prop\_id,

imprv\_id,

imprv\_det\_id

) ON [PRIMARY]

CREATE CLUSTERED INDEX IX\_\_clientdb\_improvement\_features ON dbo.\_clientdb\_improvement\_features

(

prop\_val\_yr,

prop\_id,

imprv\_id

) ON [PRIMARY]

CREATE CLUSTERED INDEX IX\_\_clientdb\_abs\_subdv ON dbo.\_clientdb\_abs\_subdv

(

abs\_subdv\_ind,

abs\_subdv\_cd

) ON [PRIMARY]

CREATE CLUSTERED INDEX IX\_\_clientdb\_taxing\_jurisdiction\_detail ON dbo.\_clientdb\_taxing\_jurisdiction\_detail

(

sup\_yr,

prop\_id

) ON [PRIMARY]

CREATE CLUSTERED INDEX IX\_\_clientdb\_values\_detail ON dbo.\_clientdb\_values\_detail

(

prop\_val\_yr,

prop\_id

) ON [PRIMARY]

CREATE CLUSTERED INDEX IX\_\_clientdb\_property\_prop\_id\_prop\_val\_yr ON dbo.\_clientdb\_property

(

prop\_val\_yr,

prop\_id

) ON [PRIMARY]

CREATE NONCLUSTERED INDEX IX\_\_clientdb\_property\_owner\_name\_prop\_val\_yr ON dbo.\_clientdb\_property

(

prop\_val\_yr,

owner\_name

) ON [PRIMARY]

CREATE NONCLUSTERED INDEX IX\_\_clientdb\_property\_prop\_val\_yr\_abs\_subdv\_cd ON dbo.\_clientdb\_property

(

prop\_val\_yr,

abs\_subdv\_cd

) ON [PRIMARY]

CREATE NONCLUSTERED INDEX IX\_\_clientdb\_property\_prop\_val\_yr\_hood\_cd ON dbo.\_clientdb\_property

(

prop\_val\_yr,

hood\_cd

) ON [PRIMARY]

CREATE NONCLUSTERED INDEX IX\_\_clientdb\_property\_prop\_val\_yr\_geo\_id on dbo.\_clientdb\_property

(

prop\_val\_yr,

geo\_id

)

CREATE NONCLUSTERED INDEX IX\_\_clientdb\_property\_prop\_val\_yr\_dba\_name on dbo.\_clientdb\_property

(

prop\_val\_yr,

dba\_name

)

CREATE NONCLUSTERED INDEX IX\_\_clientdb\_property\_prop\_val\_yr\_mapsco on dbo.\_clientdb\_property

(

prop\_val\_yr,

mapsco

)

CREATE NONCLUSTERED INDEX IX\_\_clientdb\_property\_prop\_val\_yr\_state\_cd on dbo.\_clientdb\_property

(

prop\_val\_yr,

state\_cd

)

CREATE NONCLUSTERED INDEX IX\_\_clientdb\_property\_prop\_val\_yr\_situs\_num on dbo.\_clientdb\_property

(

prop\_val\_yr,

situs\_num

)

CREATE NONCLUSTERED INDEX IX\_\_clientdb\_property\_prop\_val\_yr\_situs\_street on dbo.\_clientdb\_property

(

prop\_val\_yr,

situs\_street

)

CREATE NONCLUSTERED INDEX IX\_\_clientdb\_property\_prop\_val\_yr\_street\_name on dbo.\_clientdb\_property

(

prop\_val\_yr,

street\_name

)

CREATE NONCLUSTERED INDEX IX\_\_clientdb\_property\_prop\_val\_yr\_situs\_city on dbo.\_clientdb\_property

(

prop\_val\_yr,

situs\_city

)

create nonclustered index IDX\_\_clientdb\_taxing\_jurisdiction\_detail\_sup\_yr

on dbo.\_clientdb\_taxing\_jurisdiction\_detail (sup\_yr)

with fillfactor = 90

create nonclustered index IDX\_\_clientdb\_roll\_value\_history\_detail\_prop\_val\_yr

on \_clientdb\_roll\_value\_history\_detail (prop\_val\_yr)

with fillfactor = 90

CREATE NONCLUSTERED INDEX IX\_payment\_trans\_payment\_id ON dbo.payment\_trans

(

payment\_id

) ON [PRIMARY]

CREATE NONCLUSTERED INDEX IX\_payment\_trans\_prop\_id ON dbo.payment\_trans

(

prop\_id

) ON [PRIMARY]

CREATE NONCLUSTERED INDEX IX\_payment\_trans\_bill\_id ON dbo.payment\_trans

(

bill\_id

) ON [PRIMARY]

CREATE NONCLUSTERED INDEX IX\_refund\_due\_trans ON dbo.refund\_due\_trans

(

bill\_id

) ON [PRIMARY]

CREATE NONCLUSTERED INDEX IX\_\_clientdb\_street\_name ON dbo.\_clientdb\_street\_name

(

street

) ON [PRIMARY]

CREATE NONCLUSTERED INDEX IX\_\_clientdb\_city ON dbo.\_clientdb\_city

(

city

) ON [PRIMARY]

CREATE NONCLUSTERED INDEX IX\_\_clientdb\_township\_section ON dbo.\_clientdb\_township\_section

(

township\_section

) ON [PRIMARY]

CREATE NONCLUSTERED INDEX IX\_\_clientdb\_property\_image ON dbo.\_clientdb\_property\_image

(

prop\_id

) ON [PRIMARY]

ALTER TABLE dbo.\_clientdb\_property\_image ADD PRIMARY KEY (id)

CREATE NONCLUSTERED INDEX IX\_\_clientdb\_property\_sketch ON dbo.\_clientdb\_property\_sketch

(

prop\_id

) ON [PRIMARY]

ALTER TABLE dbo.\_clientdb\_property\_sketch ADD PRIMARY KEY (id)

'

exec(@sql)

if @region = @texas

begin

set @status = 'Bill Index '

set @sql = 'use ' + @input\_database\_name

set @sql = @sql + '

CREATE NONCLUSTERED INDEX IX\_bill ON dbo.bill

(

bill\_id

) ON [PRIMARY]

CREATE NONCLUSTERED INDEX IX\_bill\_1 ON dbo.bill

(

prop\_id

) ON [PRIMARY]

'

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

end

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

/\*\*\*\*\*\*\*\*\*/

set @status = 'SHRINKDATABASE and Log'

set @sql = 'use ' + @input\_database\_name + '

declare @db\_file varchar(100)

-- Shrink Log File

select @db\_file = name from sys.database\_files where type = 1

print @db\_file

dbcc shrinkfile (@db\_file, 5)

-- Shrink DB file

select @db\_file = name from sys.database\_files where type = 0

print @db\_file

dbcc shrinkfile (@db\_file, 5)

'

exec(@sql)

set @sql = 'DBCC SHRINKDATABASE (' + @input\_database\_name + ', 5)'

exec(@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = '\_clientdb\_log'

set @sql = 'select \* into ' + @input\_database\_name + '.dbo.\_clientdb\_log from \_clientdb\_log'

exec(@sql)

set @sql = 'use ' + @input\_database\_name + ' ;checkpoint'

exec (@sql)

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = 'DropPATempTables'

exec DropPATempTables

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

set @status = 'Export Complete!'

insert into \_clientdb\_log (id, start\_dt, finish\_dt, status, error)

values (@log\_id, @start\_date, getdate(), @status, @@error)

set @start\_date = getdate()

print ' Done Exporting at ' + convert(varchar(30), getdate(), 109)

-- Stop: Export Process

set @sql = 'use ' + @input\_database\_name + '

if not exists (

select \*

from master.dbo.syslogins

where name = ''pacsnonprivy''

)

begin

exec master.dbo.sp\_addlogin ''pacsnonprivy'', ''xi4b]ftx1p.w''

exec master.dbo.sp\_grantdbaccess ''pacsnonprivy''

end

if exists (

select \*

from sysusers

where name = ''pacsnonprivy''

)

begin

exec sp\_dropuser ''pacsnonprivy''

end

exec sp\_grantdbaccess ''pacsnonprivy''

exec sp\_addrolemember ''db\_datawriter'', ''pacsnonprivy''

exec sp\_addrolemember ''db\_datareader'', ''pacsnonprivy''

exec(''grant insert on user\_input\_query to pacsnonprivy'')

exec(''grant insert on user\_input\_query\_idlist to pacsnonprivy'')

'

exec(@sql)

set @sql = '

if exists (

select \*

from sysobjects

where name = ''db\_info\_pa'' and xtype = ''U''

)

begin

drop table db\_info\_pa

end

create table db\_info\_pa (

version varchar(23) not null,

export\_dt datetime not null,

region varchar(10)

)

insert db\_info\_pa (version, export\_dt, region)

values (''1.2.4.00'', getdate(), '''+ @region +''')

'

set @sql2 = 'exec ' + @input\_database\_name + '.dbo.sp\_executesql N''' + replace(@sql, '''', '''''') + ''''

exec (@sql2)

--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

DECLARE @subscriber\_server varchar(255)

select @subscriber\_server = szConfigValue from pacs\_config where szGroup = 'Property Access' and szConfigName = 'Subscriber'

if (select category from master..sysdatabases where name = @input\_database\_name) > 0 and

(@subscriber\_server is not null)

begin

-- Start: add publications

print 'Adding Publications for ' + @input\_database\_name

DECLARE @name varchar(255)

-- Enabling the replication database

set @sql = 'master..sp\_replicationdboption @dbname = N''' + @input\_database\_name+ ''', @optname = N''publish'', @value = N''true'''

exec (@sql)

set @sql = @input\_database\_name + '..'

set @sql = @sql + 'sp\_addpublication @publication = N''' + @input\_database\_name + ''', @restricted = N''false'', @sync\_method = N''native'', @repl\_freq = N''snapshot'', @description = N''Property Access Automatic Snapshot publication of ' + @input\_database\_name + ' database.'', @status = N''active'', @allow\_push = N''true'', @allow\_pull = N''true'', @allow\_anonymous = N''false'', @enabled\_for\_internet = N''false'', @independent\_agent = N''false'', @immediate\_sync = N''false'', @allow\_sync\_tran = N''false'', @autogen\_sync\_procs = N''false'', @retention = 336, @allow\_queued\_tran = N''false'', @snapshot\_in\_defaultfolder = N''true'', @compress\_snapshot = N''false'', @ftp\_port = 21, @ftp\_login = N''anonymous'', @allow\_dts = N''false'', @allow\_subscription\_copy = N''false'', @add\_to\_active\_directory = N''false'''

exec (@sql)

set @sql = @input\_database\_name + '..'

set @sql = @sql + 'sp\_addpublication\_snapshot @publication = N''' + @input\_database\_name + ''',@frequency\_type = 1, @frequency\_interval = 1, @frequency\_relative\_interval = 1, @frequency\_recurrence\_factor = 0, @frequency\_subday = 1, @frequency\_subday\_interval = 1, @active\_start\_date = 0, @active\_end\_date = 0, @active\_start\_time\_of\_day = 0, @active\_end\_time\_of\_day = 0'

exec (@sql)

set @sql = 'declare cur\_object insensitive cursor for select so.name, so.xtype

from ' + @input\_database\_name + '..sysobjects as so

where (so.category & 2) = 0 and so.xtype in (''P'', ''V'', ''U'', ''TF'')

and so.name not in (''table\_cache\_status'', ''user\_input\_query'')'

execute(@sql)

OPEN cur\_object

FETCH NEXT FROM cur\_object

INTO @name, @type

WHILE @@FETCH\_STATUS = 0

BEGIN

declare @pre\_creation\_cmd varchar(10)

set @sql = @input\_database\_name + '..'

if (@type = 'P')

begin

set @sql = @sql + 'sp\_addarticle @publication = N''' + @input\_database\_name + ''', @article = N''' + @name + ''', @source\_owner = N''dbo'', @source\_object = N''' + @name + ''', @destination\_table = N''' + @name + ''', @type = N''proc schema only'', @creation\_script = null, @description = null, @pre\_creation\_cmd = N''drop'', @schema\_option = 0x0000000000002001, @status = 16'

end

if (@type = 'V')

begin

set @sql = @sql + 'sp\_addarticle @publication = N''' + @input\_database\_name + ''', @article = N''' + @name + ''', @source\_owner = N''dbo'', @source\_object = N''' + @name + ''', @destination\_table = N''' + @name + ''', @type = N''view schema only'', @creation\_script = null, @description = null, @pre\_creation\_cmd = N''drop'', @schema\_option = 0x0000000000002101, @status = 16'

end

if (@type = 'U')

begin

set @pre\_creation\_cmd = 'drop'

set @sql = @sql + 'sp\_addarticle @publication = N''' + @input\_database\_name + ''', @article = N''' + @name + ''', @source\_owner = N''dbo'', @source\_object = N''' + @name + ''', @destination\_table = N''' + @name + ''', @type = N''logbased'', @creation\_script = null, @description = null, @pre\_creation\_cmd = N''' + @pre\_creation\_cmd + ''', @schema\_option = 0x000000000000FFF1, @status = 16, @vertical\_partition = N''false'', @ins\_cmd = N''SQL'', @del\_cmd = N''SQL'', @upd\_cmd = N''SQL'', @filter = null, @sync\_object = null'

end

if (@type = 'TF')

begin

set @sql = @sql + 'sp\_addarticle @publication = N''' + @input\_database\_name + ''', @article = N''' + @name + ''', @source\_owner = N''dbo'', @source\_object = N''' + @name + ''', @destination\_table = N''' + @name + ''', @type = N''func schema only'', @creation\_script = null, @description = null, @pre\_creation\_cmd = N''drop'', @schema\_option = 0x0000000008000001, @status = 16'

end

exec (@sql)

FETCH NEXT FROM cur\_object

INTO @name, @type

END

CLOSE cur\_object

DEALLOCATE cur\_object

set @sql = @input\_database\_name + '..'

set @sql = @sql + 'sp\_addsubscription @publication = N''' + @input\_database\_name + ''', @article = N''all'', @subscriber = N''' + @subscriber\_server + ''', @destination\_db = N''' + @input\_database\_name + ''', @sync\_type = N''automatic'', @update\_mode = N''read only'', @offloadagent = 0, @dts\_package\_location = N''distributor'''

exec (@sql)

-- Stop: add publications

end

GO