GSTN Data Analysis

Flow Of Postgres Model

CITY WISE PROCESS

RAW DATA TABLE STRUCTURE

Table Name Format: “raw_”+city_name		
Column Name	**Type**	**Description**
raw_name	text	raw area name

MASTER DB TABLE STRUCTURE

Table Name Format: city_name+”_ADDR_ADMIN_R”		
Column Name	**Type**	**Description**
NAME	text	Combination of SSLC_NME +SUBL_NME+LOC_NME
LOC_ID	integer	Locality id
LOC_NME	text	Locality name
SUBL_NME	text	Sub-locality name
SUBL_ID	integer	Sub-locality id
SSLC_NME	text	Sub sub locality name
SSLC_ID	integer	Sub sub locality id
CITY_ID	integer	City name id
STT_ID	integer	State code
TYPE	text	
SP_GEOMETRY	geometry	

OUTPUT TABLE STRUCTURE

Table Name Format: ”gstn_output_”+city_name		
Column Name	**Type**	**Description**
srno	integer	Serial number
raw_name	text	raw area name
M_LOC_NME	text	Master Locality Name
M_LOC_ID	integer	Master Locality ID
SUBL_NME	character varying	Sub locality name
SUBL_ID	integer	Sub locality id
SSLC_NME	character varying	Sub sub locality name
SSLC_ID	integer	Sub sub locality id

ID	integer	
Unmatch_String	text	Remaining raw area name after all match
Match_String	text	Matched raw area name
Replace_String	character varying	Unwanted string like city_name,state_name etc.
status	text	Match Status
N_LOC	text	Near by locality name
N_LOC_MATCHED	text	Near by locality matched
N_LOC_MATCHED_ID	integer	Near by locality matched id
N_SUBL_MATCHED	text	Near by sub locality matched
N_SUBL_MATCHED_ID	integer	Near by sub locality matched id
N_SSLC_MATCHED	text	Near by sub sub locality matched
N_SSLC-MATCHED_ID	integer	Near by sub sub locality matched id
M_N_LOC	text	Main near by locality
M_N_LOC_MATCHED_ID	integer	Main near by locality matched id

N_LOC DICTIONARY TABLE STRUCTURE

Table Name Format: city_name+”_nloc_dictionary”		
Column Name	**Type**	**Description**
M_LOC	character varying	Master Locality Name
M_LOC_ID	integer	Master Locality ID
N_LOC	character varying	Near By Locality Of Master Locality
N_LOC_ID	integer	Near By Locality ID

N_SUBL DICTIONARY TABLE STRUCTURE

Table Name Format: city_name+”_sloc_dictionary”		
Column Name	**Type**	**Description**
M_LOC	character varying	Master Locality Name
M_LOC_ID	integer	Master Locality ID
N_LOC	character varying	Near By Locality Of Master Locality
N_LOC_ID	integer	Near By Locality ID
N_SUBL_NME	character varying	Near By Sub Locality Of

		Near By Locality
N_SUBL_ID	integer	Near By Sub Locality ID

N_SSLC DICTIONARY TABLE STRUCTURE

Table Name Format: city_name+"_sslcloc_dictionary"		
Column Name	**Type**	**Description**
M_LOC	character varying	Master Locality Name
M_LOC_ID	integer	Master Locality ID
N_LOC	character varying	Near By Locality Of Master Locality
N_LOC_ID	integer	Near By Locality ID
N_SLCL_NME	character varying	Near By Sub Sub Locality Of Near By Locality
N_SLCL_ID	integer	Near By Sub Sub Locality ID

FILTERATION PROCESS: Below steps are following during filter of gstn data.

SAMPLE RAW TABLE NAME : raw_chandigarh <input table>
SAMPLE MASTER DB TABLE : CH_ADDR_ADMIN_R <admin table>
SAMPLE OUTPUT TABLE NAME: gstn_output_chandigarh

STEP 1: Insert raw data from table <raw_chandigarh> into output table <gstn_output_chandigarh>.

STEP2: Create city wise master table<Chandigarh_ADDR_ADMIN_R> from <DL_ADDR_ADMIN_R> using CITY_ID.

STEP3: Exact match raw_name with master table name that contains combination of (SSLC_NME+", "+SUBL_NME+", "+LOC_NME)

STEP4: Locality match raw_name with master table name <chandigarh_ADDR_ADMIN_R> .

STEP5: Update status of not matched in output table name<gstn_output_chandigarh>.

STEP6: Sub Locality match raw_name with master table name<chandigarh_ADDR_ADMIN_R> .

STEP7: Sub Sub Locality match raw_name with master table name<chandigarh_ADDR_ADMIN_R> .

STEP8: Create neighbour loc dictionary table<city_name+_nloc_dictionary>by using of master table name<chandigarh_ADDR_ADMIN_R>.

STEP9: Create sub neighbour loc dictionary table<city_name+_nloc_dictionary>by using of master table name<chandigarh_ADDR_ADMIN_R>.

STEP10: Create sub sub neighbour loc dictionary table<city_name+_nloc_dictionary>by using of master table name<chandigarh_ADDR_ADMIN_R>.

STEP11: Update neighbour locality with <raw_name>.

STEP12: Update neighbour sub locality with <raw_name>.

STEP12: Update neighbour sub sub locality with <raw_name>.

STEP13: Update count of table <raw_chandigarh> into output table <gstn_output_chandigarh>.

GSTN DATA ANALYSIS

Approach

Step1 : Input data provided by ram.

Step2: Input has a standard format with standard reference.

Step3: First level of cleaning from input data.

Cleaning process:

Step1: Input has hold house_no, trade,floor,house_name, street_name, area_nme, pincode.

Step2: We removed house number,trade ,floor from input data by using geocoded data then remaining useful column such as house_name, street_area, area_nme,pincode are used for further process.

Step3: If input data has street_name,poi,house_name, city_name,Sub_district_name,state_name then remove from the input data using master reference.

Step4: Now we find the unmatched area_nme.

Step5: Process on unmatched area_nme by using postgres model.

Input Data Structure GSTN DATA

We are use these input for process:

Id	
CHECKSUM	
BLDG_NAM	INPUT DATA
STREET_NAM	
AREA_NAM	
PIN_CD	
Address	GEOCODED ADDRESS
houseName	STANDARDIZE REFERENCE
Poi	
Street	
subSubLocality	
subLocality	
Locality	
Village	
subDistrict	
District	
City	
state	

1. level cleaning process

Step1: Area name as raw_name is input .

Step2: On the basis of standard references such as (POI+STREET+SSLC+SUBL+LOC+VILLAGE+SUBDISTRICT+DISTRICT) are removed in AREA_NAM columns by using of these standard references.

Approach For GSTN DATA

Objective to identified Admin Names

STEP1: Match with output token and cleaned the string matching with output token.

STEP2: Pick particular column name in input data such as <AREA_NAM>.

STEP3: Match this <AREA_NAM> by using of current postgres process on iteration 1 level.

STEP4: If <AREA_NAM> is match with <LOC_NME> or <SUBL_NME> or <SSLC_NME> then status is matched.

STEP5: If <AREA_NAM> is not match then process on remaining Unmatched String.

STEP6: In this remaining Unmatched String try to identified and parse POI and STREETS separately.

Steps For New Approach

Step1: On the basis of standard references such as (sslc+loc+district……) are removed in input token .

Step2: Apply Cleansing Process model.

Step3: Unique cleaned data.

Step4: Apply Postgres Model In unique data .

Step5: Find not matched data from gstn output table.

Step6: Clean Not matched data by using of Cleaning Process model.

Step7: Group by Not matched data.

Step8: Split group by Not matched data by using Space

Step9: Make Combination of Unique Not matched data.

Step10: Create table of combination not matched data.

STEP11: Match this combination not matched data with LOC_NME, SUBL_NME,SSLC_NME.

Step12: Using of Soundex algorithm in LOC_NME,SUBL_NME,SSLC_NME.

Adding New Approach for gstn data

Step1: Find Unmatch data from gstn output table.

Step2: Match with this Unmatch data in Admin table by using pin code reference.

Step3: If raw table data pin code is matched with admin table then change the status of Unmatch data.(Direct matching with admin table)

Step4: Step3 is apply for LOC_NME,SUBL_NME,SSLC_NME matched.

Step5: Remaining Unmatch data matched with soundex algorithm by using of pin code reference

Cleaning Process for raw data

1. Remove special character from starting and ending of Area_Name.
2. Clean by cleansing_ref table.
3. Remove only numeric digits.
4. Remove less than or equal to 3.
5. Remove district name.
6. Remove village name.
7. Check poi street…. and remove.

Output data count of Lucknow

Total data	137679
Cleaned data	31220
Unique data	13135
LOC_MATCHED	290
COMB_LOC_MATCHED,COMB_SUBL_MATCHED	486
COMB_LOC_MATCHED,COMB_SSLC_MATCHED	74
SSLC_MATCHED	1
FUZZY_LOC_MATCHED	1342
EXACT_MATCH	203
COMB_LOC_MATCHED	2573
NOT_MATCHED	7389
N_SUB_LOC_MATCHED	12
N_LOC_MATCHED	10
SUBL_MATCHED	8
N_SSLC_MATCHED	2
REF_FUZZY_SUBL_MATCHED	313
REF_SUBL_MATCHED	205
REF_FUZZY_SSLC_MATCHED	32
REF_SSLC_MATCHED	56
REF_FUZZY_LOC_MATCHED	140

Process for not matched data by using of soundex and fuzzy matched.

1. Find the soundex of the table
2. Find the fuzzy match of the table