{

"workflow": [

{

"COMMENT": "WALMART LOCATION FINDER"

},

{

"SYSTEM\_MISSING": {

"entity\_type": "LOCATION",

"parent\_entity\_id": "b6cd2908-4c9a-4d98-9676-5c1bad7c9007",

"ultimate\_parent\_entity\_id": "",

"state": "ACTIVE",

"location\_type": "STORE"

}

},

{

"FOR": {

"ITEM": "</$year\_item$/>",

"IN\_RANGE": {

"FIRST": 2020,

"LAST": 2023

},

"END\_TAG": "YEAR"

}

},

{

"COMMENT": {

"FOR": {

"ITEM": "</$month\_item$/>",

"IN\_LIST": "</$calendar\_month\_list$/>",

"END\_TAG": "PERIOD"

}

}

},

{

"FOR": {

"ITEM": "</$state\_item$/>",

"IN\_LIST": "</$state\_code\_list$/>",

"END\_TAG": "STATE"

}

},

{

"VERBOSE": "WALMART LOCATION FINDER INTERIOR: </$state\_item$/> </$month\_item$/> of </$year\_item$/>"

},

{

"AI": {

"ENGINE": "PERPLEXITY",

"PROMPT": "List all Walmart stores that opened in the state of </$state\_item$/> in </$year\_item$/>. Respond only in a JSON format and do not include an explanation. Do not wrap the json codes in JSON markers. In the JSON format where each location found has a unique item key identifying the location, and a value that is a python dictionary where the key of 'entity\_name' holds the Walmart identification for this location, the key of 'address\_line\_1' holds the location street address, the key of 'address\_city' holds the location city, the key of 'address\_state' holds the location two-character state code, the key of 'address\_zip' holds the location zip code, the key of 'location\_established\_date' that holds the opening date that that this location in the format of 'YYYY-MM-DD', the key of 'location\_closed\_date' that holds the date that that this location was permanently closed in the format of 'YYYY-MM-DD', the key of 'longitude' that holds the longitude of this location with 4 decimal precision, the key of 'latitude' that holds the latitude of this location with 4 decimals of precision, the key of 'county' that holds the county of this location. If a value is missing or unknown for a given dictionary key then exclude that key and value from the JSON element response.",

"RESPONSE\_TYPE": "JSON",

"RESPONSE\_VAR": "</$$ai\_response$$/>"

}

},

{

"FOR": {

"FOR\_END": "AI\_RESULT\_LOOP",

"KEY": "</$ai\_key$/>",

"VALUE": "</$ai\_row$/>",

"JSON": "</$$ai\_response$$/>"

}

},

{

"DROP\_VAR\_VALUE": {

"KEYS": [

"location\_established\_date",

"location\_closed\_date",

"longitude",

"latitude",

"county"

]

}

},

{

"SET\_VAR": {

"PARTITION": "</$ai\_row$/>"

}

},

{

"COMMENT": "Processing each found store to see if it exists"

},

{

"SQL": {

"QUERY": "SELECT payload->>'longitude' as longitude, payload->>'latitude' as latitude, id FROM signal.dev\_entity\_metadata where entity\_type = 'LOCATION' and payload->>'longitude' = '</longitude/>%' and payload->>'latitude' = '</latitude/>%' ",

"DATAFRAME": "SQL\_DF"

}

},

{

"IF": {

"CONDITION": "</SQL\_DF/>.shape[0] == 0",

"TAG": "NO\_STORE\_FOUND"

}

},

{

"RECORD\_RESEARCH": {

"TARGET\_TABLE": "</$entity\_metadata$/>",

"TARGET\_JSON": {

"JSON\_COLUMN": "payload",

"JSON\_KEYS": [

"location\_established\_date",

"location\_closed\_date",

"location\_type",

"address\_line\_1",

"address\_city",

"address\_state",

"address\_zip",

"longitude",

"latitude",

"county",

"location\_type"

]

},

"TABLE\_COLUMNS": [

"parent\_entity\_id",

"ultimiate\_parent\_entity\_id",

"entity\_type",

"entity\_name",

"payload",

"state"

]

}

},

{

"END": "NO\_STORE\_FOUND"

},

{

"END": "AI\_RESULT\_LOOP"

},

{

"END": "STATE"

},

{

"END": "PERIOD"

},

{

"END": "YEAR"

}

]

}