# I – DATABASE

Refer file venturaDB.sql

**Pre-req**:

1. PostgreSql server / PgAdmin tool
2. Need to know following directories
   1. **inputdir** : my case it is c:\\zunk\\lite\\input

this is root directory where other directories are created

* 1. **workdir**: my case it is c:\\zunk\\lite\\work

this is root directory where output files are created

* 1. **systemdir**: my sql file has “c:\\users\\spadte\\source\\repos\\padtes\\DotNetCorePOC\\ddl\_sql”

this is where config json files, word source templates are saved

1. for each Courier
   1. Need series -start and end of series
   2. Need name, 3 char code and 2 char code
2. Need Printer 3 char code and 2 char code: I used in script XYZ as 3 char code, 51 as 2 char code.

Steps:

1. Refer file venturaDB.sql (also in git in folder ddl\_sql)
2. Create Schema. In my scripts I have assumed **ventura**. – No Script to create Schema.

From script copy/paste in pgAdmin / run:

* 1. CREATE TABLE ventura.system\_param
  2. AS PER **DIRECTORIES** ABOVE - insert into **ventura.system\_param** after modifying the insert sql ALSO there is printer\_code3 and printer\_code2
  3. CREATE TABLE ventura.fileinfo
  4. CREATE TABLE ventura.filedetails
  5. CREATE TABLE ventura.filedetail\_actions
  6. CREATE TABLE ventura.filetypemaster

insert into ventura.filetypemaster (isactive,biztype,module\_name,file\_def\_json\_fName)

values('1','lite\_inp','lite','lite\_input.json');

insert into ventura.filetypemaster (isactive,biztype,module\_name,file\_def\_json\_fName, fname\_pattern)

values('1','lite\_resp','lite','lite\_imm\_resp.json','PRN{{sys\_param(printer\_code)}}RES{{now\_ddmmyy}}{{serial\_no}}.txt');

insert into ventura.filetypemaster (isactive,biztype,module\_name,file\_def\_json\_fName, fname\_pattern)

values('1','lite\_stat','lite','lite\_status\_rep.json','PRN{{sys\_param(printer\_code)}}STS{{now\_ddmmyy}}{{serial\_no}}.txt');

insert into ventura.filetypemaster (isactive,biztype,module\_name,file\_def\_json\_fName, fname\_pattern)

values('1','lite\_word\_apy','lite','apy\_letter.json','apyLetter{{courier\_cd}}\_{{serial\_no}}.docx');

insert into ventura.filetypemaster (isactive,biztype,module\_name,file\_def\_json\_fName, fname\_pattern)

values('1','lite\_word\_nps','lite','NPS\_Lite\_letter.json','npsLiteLetter{{courier\_cd}}\_{{serial\_no}}.docx');

* 1. CREATE TABLE ventura.counters
     1. insert into ventura.counters(isactive,counter\_name,parent\_id,descript) values ('1','couriers',0, 'master rec for couriers')
     2. insert into ventura.counters(isactive,counter\_name,parent\_id) values ('1','generic',0)
     3. **NOTE** : from the **couriers record** inserted above in couters, you will need to add a “child” record per courier, with parent\_id = 1 (assuming that record was inserted first and id is created as 1.
  2. CREATE TABLE ventura.states
  3. CREATE TABLE ventura.countries
  4. create table ventura.reject\_reasons
  5. CREATE TABLE ventura.couriers
     1. Insert into ventura.couriers(isactive, code, code3, name) … make your entries here
     2. In the future as new couriers are added, this record is needed before processing input
  6. CREATE OR REPLACE FUNCTION ventura.get\_serial\_number
  7. CREATE OR REPLACE FUNCTION ventura.lock\_counter

1. See script file for
   1. insert into ventura.states
   2. insert into ventura.countries
   3. insert into ventura.reject\_reasons

from git, get all the .json files and put in

# II – Executable

## Pre-req:

1. Database setup with system\_param, counters and couriers etc. records added
2. Dot net core 3.1 or later
3. Directories – full access (add / modify / delete files and directories for the user that will run the exe
4. Json files: config for various functions – see git ddl\_sql directory
   1. Note: ddl\_sql has all sorts of files, not all are needed. Only Scriban script as .txt and .json are needed
   2. This is configured as “system” directory (Refer Database setup) and files are expected here.
5. There are 3 parameters in appSettings.json
   1. Schema name
   2. Connection string
   3. **Log file** location/name – actual files are named for the system date.
   4. **NOTE: Log files** do not have personally Identifiable data or sensitive data. **Do not delete** files till things start going smooth.

for ex.

"logFileName": "C:\\Zunk\\POC\_Log.txt",

"pgConnection": "Server=localhost; Port=5433; Database=postgres; User Id=userventura; Password=passwordventura; ",

"pgSchema": "ventura"

Install Exe on the machine where **.net core** is installed.

**Copy the files**:

1. NpsApy.exe
2. NpsApy.pdb
3. \*.dll
4. appSettings.json – REVIEW and UPDATE 3 parameters
5. etc.

**Parameters**:

Parameters are named, not positional. Parameters are typically as follows, start with dash “-”and given as key=value

-moduleName=LITE -op=READ

1. moduleName: Lite or Reg
2. op: operation is
   1. read
   2. write
   3. report
   4. updstat
3. runFor: directory of the day as yyyymmdd for ex. 20210620 for 20th June 2021
4. file: within report or write operation, file takes various values

Parameters for **specific task**:

1. Read data:

|  |  |
| --- | --- |
| -moduleName=LITE -op=READ | |
|  | Read all the directories under input directory and process data |
| -moduleName=LITE -op=READ -runfor=**20210620** | |
|  | Read and process data for one directory **20210620 - variable** |

Testing/sanity check

1. Check logs for error / warning printed anywhere
2. See database
   1. Select \* from ventura.filedetails
   2. select filedetails.\* from ventura.filedetails

join ventura.fileinfo on fileinfo.id = filedetails.fileinfo\_id

where fileinfo.isdeleted='0'

and fileinfo.module\_name = 'lite'

and fileinfo.biztype = 'lite\_inp'

* 1. same as above: + and fileinfo.fpath like '%\\20210620\\%'

1. Immediate response file:

|  |  |
| --- | --- |
| -moduleName=LITE -op=write -file=resp -runfor=**20210620** | |
|  | Write immediate response for **20210620 – variable**  **-runfor** is required |

Testing/sanity check

1. Check logs for error / warning printed anywhere
2. Compare with the input.
   1. Import in Excel to confirm header and details
   2. Verify file name
3. Note: Same parameters rerun will result in no records. To Rerun – sql script is needed – such as
   1. delete from ventura.filedetail\_actions where action\_void = '0' **and** **action\_done = '1IMM\_RESP'**
   2. or update ventura.filedetail\_actions set action\_void = '1' where…
4. Simple Reports:

|  |  |
| --- | --- |
| -moduleName=LITE -op=report -file=all -runfor=20210620  -moduleName=LITE -op=report -file=im\_resp\_todo -runfor=20210620  -moduleName=LITE -op=report -file=update\_todo -runfor=20210620  -moduleName=LITE -op=report -file=status\_todo -runfor=20210620  -moduleName=LITE -op=report -file=status\_done -runfor=20210620 | |
|  | report for **20210620 – variable**  **-runfor** is required |
| file = all | All the records read for that directory |
| file = im\_resp\_todo | Immediate response is yet to be done |
| file = update\_todo | Records are not updated as Ok or Printed or error (see updstat) |
| file = status\_todo | Final Status not yet sent |
| file = status\_done | Final Status file was run |

Testing/sanity check

1. Check logs for error / warning printed anywhere
2. Compare with the input.
3. Reports can be run as many times as needed.
4. Reports are all same format and can be used for updstat to update status, open in Excel and save with proper status
5. Update status:

|  |  |
| --- | --- |
| -moduleName=LITE -op=updstat -file=**C:\Zunk\Lite\work\20210620\nps\_lite\stat\_todo\_2021Jul14\_17\_24.csv** | |
|  | Update status of the records.  Take output of – in particular – update-todo (Refer simple reports)  Open in Excel.  SAVE  Update print date or pick date or error codes.  Error codes are saved and printed as comma(,) separated BUT IN THIS FILE USE + (plus) if there are multiple codes  SAVE  Keep CSV when saved  Now run this update.  Once updated, next time the records if update stat is attempted, will be ignored  Use super\_update if you mean to re-update (**untested** feature)  If file is invalid – such as incorrect error code, whole file is rejected |

Testing/sanity check

1. Check logs for error / warning printed anywhere
2. Run report all or update\_todo to verify
3. NPS / ~~APY~~ Letters:

|  |  |
| --- | --- |
| -moduleName=LITE -op=write -file=letter -runfor=20210620 | |
|  | Write word letters for **20210620 – variable**  **-runfor** is required |

Testing/sanity check

1. Check logs for error / warning printed anywhere
2. Compare with the input.
   1. See data, formatting and tags **{{**
   2. Confirm pagination
3. Final Status:

|  |  |
| --- | --- |
| -moduleName=LITE -op=write -file=stat -runfor=20210620 | |
|  | Write status file for **20210620 – variable**  **-runfor** is required |

Testing/sanity check

1. Check logs for error / warning printed anywhere
2. Compare with the input.
   1. Import in Excel to confirm header and details
   2. Verify file name
3. Note: Same parameters rerun will result in no records. To Rerun – sql script is needed – such as
   1. delete from ventura.filedetail\_actions where action\_void = '0' **and** **action\_done = '**3STATUS\_REP**'**
   2. or update action\_void=’1’ …
4. Letters cannot be printed after final status of the report

**Third party libraries**:

Json parsing –by **Newtonsoft**

**Scriban** for dynamic script

**DotNetZip** for zipping – end result word docx

**pgSQL** connector