In [1]:

```
import pandas as pd
```

In [2]:

```
# df = pd.read_csv("../data/dry_etch.csv")

# As the columns cell data contains 'whitespace'. So, just trim it while fetching data.
df = pd.read_csv("../data/dry_etch.csv",
                 converters={'EquipID': str.strip,
                             'RecipeUsedOnEquip': str.strip,
                             'ActualDateIn': str.strip,
                             'LotId': str.strip})
```

In [3]:

```
# show the column headers
df.columns.to_list()
```

Out[3]:

```
['EquipID', 'RecipeUsedOnEquip', 'ActualDateIn', 'LotId']
```

In [4]:

```
# first 5 rows
df.head()
```

Out[4]:

| | EquipID | RecipeUsedOnEquip | ActualDateIn | LotId |
|---|---|---|---|---|
| **0** | ASBE1 | 2 | 09-03-2021 08:39 | F20280002.F1 |
| **1** | ASBE1 | 2 | 09-03-2021 05:44 | F20350002.F1 |
| **2** | ASBE1 | 2 | 09-03-2021 04:53 | F20200003.F1 |
| **3** | ASBE1 | #2 | 08-03-2021 12:41 | F20370001.F1 |
| **4** | ASBE1 | #2 | 08-03-2021 10:25 | F20280002.F1 |

In [5]:

```
# last 5 rows
df.tail()
```

Out[5]:

| | EquipID | RecipeUsedOnEquip | ActualDateIn | LotId |
|---|---|---|---|---|
| **105583** | | | | |
| **105584** | | | | |
| **105585** | | | | |
| **105586** | | | | |
| **105587** | | | | |

In [6]:

```
# dropping rows where at least one element is missing
nan_val = float("NaN")

# replace "" with "NaN"
df.replace("", nan_val, inplace=True)

# drop rows with column - "EquipID" values as "NaN"
df.dropna(subset = ["EquipID"], inplace=True)
```

In [7]:

```
df.tail()
```

Out[7]:

| | EquipID | RecipeUsedOnEquip | ActualDateIn | LotId |
|---|---|---|---|---|
| **17278** | RESP1B | Wf 6. Z-VIA+ASH.1MIN; WF 7. Z-VA+ASH1MIN150W; ... | 24-09-2013 12:02 | S13360004.F6 |
| **17279** | RESP1B | Z-VIA+ASH | 21-09-2013 11:51 | S13360004.F5 |
| **17280** | RESP1B | BARC+ME120SEC | 20-09-2013 15:20 | S13360004.F4 |
| **17281** | RESP1B | See the Remark | 20-09-2013 15:18 | S13360004.F2 |
| **17282** | RESP1B | SPACER-ETCH | 20-04-2013 12:09 | T005547H1.F1 |

In [8]:

```
# create new CSV file
# purpose: to view & apply filter while viewing in Excel
df.to_csv('out.csv', index=False)
```

In [9]:

```
# Fetch all the unique equip-ids provided in the data using `set()`
equip_list_unique = list(set(df["EquipID"].to_list()))
equip_list_unique
```

Out[9]:

```
['REML1A',
 'REOX1A',
 'REPL1A',
 'REML1C,ASBE1',
 'REOX1C',
 'RESP1A',
 'CLMUV1',
 'ASFE1,ASBE1',
 'CLREOX1',
 'REOX1B',
 'CLREPL1',
 'ASBE1,REML1C',
 'ASFE1',
 'RESP1B',
 'REML1C',
 'CLFRBG1',
 'CLRESP1',
 'ASBE1',
 'REPL1B',
 'CLREML1']
```

## Recipes run in ASBE1

### Objectives

- show the recipes run in lots
- frequency of each recipe

In [10]:

```
# show the equip names by which the entry has been done in DMIS
asbe1_equip_names = [e for e in equip_list_unique if "ASBE1" in e]
asbe1_equip_names
```

Out[10]:

```
['REML1C,ASBE1', 'ASFE1,ASBE1', 'ASBE1,REML1C', 'ASBE1']
```

In [11]:

```
# create df for "EquipID" col values as `*_equip_names` list
df_asbe1 = df[df["EquipID"].isin(asbe1_equip_names)]

# first 5 rows
df_asbe1.head()
```

Out[11]:

| | EquipID | RecipeUsedOnEquip | ActualDateIn | LotId |
|---|---|---|---|---|
| **0** | ASBE1 | 2 | 09-03-2021 08:39 | F20280002.F1 |
| **1** | ASBE1 | 2 | 09-03-2021 05:44 | F20350002.F1 |
| **2** | ASBE1 | 2 | 09-03-2021 04:53 | F20200003.F1 |
| **3** | ASBE1 | #2 | 08-03-2021 12:41 | F20370001.F1 |
| **4** | ASBE1 | #2 | 08-03-2021 10:25 | F20280002.F1 |

In [27]:

```
# get the shape (row, col) of dataframe
df_asbe1.shape
```

Out[27]:

```
(1013, 4)
```

In [12]:

```
# fetch all the unique recipes in this equipment/chamber by `*_equip_names` list
recipes_asbe1 = list(set(df_asbe1["RecipeUsedOnEquip"].to_list()))
recipes_asbe1
```

Out[12]:

```
['# 2',
 'TS18SL STRIP',
 '#1',
 'Recipe # D.',
 'Recipe#7(300sec) in ASFE1,Recipe:TS18SLStrip in ASP Chamber',
 'See Remark',
 '2',
 '#d',
 'Recipe: Recipe # D',
 '#6',
 'Recipe 1',
 'TS18SLSTRIP-REML1',
 'D',
 '# D.',
 'TS18SL_STRIP',
 'Recipe # 4',
 'Recipe#6',
 '# D',
```

## Recipes run in ASFE1

### Objectives

- show the recipes run in lots
- frequency of each recipe

In [13]:

```
# show the equip names by which the entry has been done in DMIS
asfe1_equip_names = [e for e in equip_list_unique if "ASFE1" in e]
asfe1_equip_names
```

Out[13]:

```
['ASFE1,ASBE1', 'ASFE1']
```

In [14]:

```
# create df for "EquipID" col values as `*_equip_names` list
df_asfe1 = df[df["EquipID"].isin(asfe1_equip_names)]

# first 5 rows
df_asfe1.head()
```

Out[14]:

| | EquipID | RecipeUsedOnEquip | ActualDateIn | LotId |
|---|---|---|---|---|
| **1011** | ASFE1 | 2 | 16-03-2021 12:07 | F20370001.F1 |
| **1012** | ASFE1 | #2 | 15-03-2021 21:14 | F19410001.F1 |
| **1013** | ASFE1 | #2 | 15-03-2021 19:33 | F21040001.F1 |
| **1014** | ASFE1 | #2 | 15-03-2021 16:41 | F20280002.F1 |
| **1015** | ASFE1 | D | 15-03-2021 10:13 | F21110001.F1 |

In [26]:

```
# get the shape (row, col) of dataframe
df_asfe1.shape
```

Out[26]:

```
(5027, 4)
```

In [15]:

```
# fetch all the unique recipes in this equipment/chamber by `*_equip_names` list
recipes_asfe1 = list(set(df_asfe1["RecipeUsedOnEquip"].to_list()))
recipes_asfe1
```

Out[15]:

```
['ASHER1--CNS Ash',
 'Recipe  3',
 '# 2',
 'Recipe# 2',
 'Recipe # D.',
 'Recipe  D',
 'e#2',
 'Recipe 2',
 'ASHER1',
 '2',
 'D B',
 '6',
 'd',
 'B D',
 '#d',
 'Recipe: Recipe # D',
 'Recipe:2',
 '#6',
```

## Recipes run in REML1

### Objectives

- show the recipes run in lots
- frequency of each recipe

In [16]:

```
# show the equip names by which the entry has been done in DMIS
reml1_equip_names = [e for e in equip_list_unique if "REML1" in e]
reml1_equip_names
```

Out[16]:

```
['REML1A', 'REML1C,ASBE1', 'ASBE1,REML1C', 'REML1C', 'CLREML1']
```

In [17]:

```
# create df for "EquipID" col values as `*_equip_names` list
df_reml1 = df[df["EquipID"].isin(reml1_equip_names)]

# first 5 rows
df_reml1.head()
```

Out[17]:

| | EquipID | RecipeUsedOnEquip | ActualDateIn | LotId |
|---|---|---|---|---|
| **1010** | ASBE1,REML1C | TS18SLSTRIP | 20-06-2014 14:19 | F13360003.F1 |
| **6040** | CLREML1 | Z-CCD-ML-ETCH1 | 17-03-2021 15:11 | F21020001.F7 |
| **6041** | CLREML1 | Z-TSL-M1-OE-55 | 17-03-2021 13:04 | F19460002.F1 |
| **6042** | CLREML1 | Z-TSL-M1-OE-55 | 15-03-2021 19:05 | F19320003.F1 |
| **6043** | CLREML1 | METAL Ti 600 | 15-03-2021 16:48 | F21020001.F7 |

In [25]:

```
# get the shape (row, col) of dataframe
df_reml1.shape
```

Out[25]:

```
(4872, 4)
```

In [18]:

```
# fetch all the unique recipes in this equipment/chamber by `*_equip_names` list
recipes_reml1 = list(set(df_reml1["RecipeUsedOnEquip"].to_list()))
recipes_reml1
```

Out[18]:

```
['Z-M1-90-75',
 'Z-CCD-ML-ETCH1',
 'TS18SLSTRIP(i',
 'TS18SL STRIP',
 'Z-M2-ASP-LSH20',
 'ts18sl strip',
 'TSL18SLSTRIP',
 'z-MT-ETCH-DOE3',
 'Z-M1-10-70;Z-M1-20-70;Z-M1-20-80',
 'e: Recipe # B(if in ASBE1) Recipe:TS18SLSTRIP(if in ASP Chamber of CLREM
L1).',
 'MT-ETCH',
 'Z-TSL-WOBARC',
 'MIM CE ETCH-1)',
 'wf-8(METAL ETCH 11 chuck 35 0C)  wf-9(Z-M1-ETCH-STSTEP))',
 'Z-M1-20-70',
 'TS18SLSTRIP(if in ASP',
 'Z-TSL-M1-OE-60',
```

## Recipes run in REOX1

In [20]:

```
# show the equip names by which the entry has been done in DMIS
reox1_equip_names = [e for e in equip_list_unique if "REOX1" in e]
reox1_equip_names
```

Out[20]:

```
['REOX1A', 'REOX1C', 'CLREOX1', 'REOX1B']
```

In [21]:

```
# create df for "EquipID" col values as `*_equip_names` list
df_reox1 = df[df["EquipID"].isin(reox1_equip_names)]

# first 5 rows
df_reox1.head()
```

Out[21]:

| | EquipID | RecipeUsedOnEquip | ActualDateIn | LotId |
|---|---|---|---|---|
| **8439** | CLREOX1 | Z-CCD-VL-DOE1 | 18-03-2020 21:19 | S20130002.F1 |
| **8440** | CLREOX1 | Z-ARCCCDPAD-DOE4 | 04-04-2019 15:13 | F18180001.F7 |
| **8441** | CLREOX1 | Z-CCD-PD-ETCH | 05-03-2019 17:32 | R17380003.F1 |
| **8442** | CLREOX1 | Z-CCD-PD-ETCH | 05-03-2019 15:47 | F18090002.F5 |
| **8443** | CLREOX1 | Z-CCD-PD-ETCH | 04-03-2019 13:35 | F18090003.F5 |

In [24]:

```
# get the shape (row, col) of dataframe
df_reox1.shape
```

Out[24]:

```
(1384, 4)
```

In [28]:

```
# fetch all the unique recipes in this equipment/chamber by `*_equip_names` list
recipes_reox1 = list(set(df_reox1["RecipeUsedOnEquip"].to_list()))
recipes_reox1
```

```
 'CS Etch',
 'C:RAA ETCH;',
 'RAA SOI, RAA ETCH',
 'Z-ARCCCDPAD-DOE1',
 ':Z-CCD-ER 20 50.',
 'B:PASSIVAION ETCH',
 'Z?-?CCD?-?CS?-?15?0?S?',
 'RAA PARTICLES CHECK',
 'Z-CCD-CS-7000',
 'Z-ARCCCDPAD-DOE4',
 'B:PASSIVA. ETCH',
 'Z-CCD-CS-170s.',
 'Sequence: Z-TTM-375 Recipe: Z-TTM-375',
 'C:RAA ETCH; R',
 'Z-CCD-ER 20 50',
 'TS18SLSTRIP',
 'A:PASSIVA.ETCH250',
 'PASS-ETCH',
 'PASSIVATION ETCH :B',
 'Z-HARD-ETCH'
```

## Recipes run in REPL1

In [29]:

```
# show the equip names by which the entry has been done in DMIS
repl1_equip_names = [e for e in equip_list_unique if "REPL1" in e]
repl1_equip_names
```

Out[29]:

```
['REPL1A', 'CLREPL1', 'REPL1B']
```

In [30]:

```
# create df for "EquipID" col values as `*_equip_names` list
df_repl1 = df[df["EquipID"].isin(repl1_equip_names)]

# first 5 rows
df_repl1.head()
```

Out[30]:

| | EquipID | RecipeUsedOnEquip | ActualDateIn | LotId |
|---|---|---|---|---|
| **8616** | CLREPL1 | Z-C1D-GC-OE-15(1,2) and Z-C1D-GC-HESET(wf 3-15)) | 19-09-2018 09:28 | S17390001.F1 |
| **8617** | CLREPL1 | Z-C1D-GC-OE-15 | 16-09-2018 12:04 | F18090003.F1 |
| **8618** | CLREPL1 | Z-C1D-GC-OE-15 | 30-08-2018 09:40 | R17380003.F1 |
| **8619** | CLREPL1 | Z-C1D-GC-OE-15 | 29-08-2018 00:38 | F18090002.F1 |
| **8620** | CLREPL1 | Z-C1D-AA-DOE2 | 30-05-2018 17:30 | F18090003.F1 |

In [31]:

```
# get the shape (row, col) of dataframe
df_repl1.shape
```

Out[31]:

```
(1737, 4)
```

In [32]:

```
# fetch all the unique recipes in this equipment/chamber by `*_equip_names` list
recipes_repl1 = list(set(df_repl1["RecipeUsedOnEquip"].to_list()))
recipes_repl1
```

Out[32]:

```
[nan,
 'Z-CCD-AA-DOE1-63',
 'Z-STI-DOE1',
 'Z-CCD-P1-DOE2',
 'z-ccd-aa etch',
 'A:Z-CCDAADOE1-63',
 'Z-STI64-AA ETCH',
 'Z-SCMP-GC-ETCH',
 'Z-SOI-AA WO SCS',
 'Z-CCD-RAA-DOE1.',
 'DEMOS-AA-ETCH',
 'GC ETH',
 'Default',
 'B: AA-ETCH',
 'AS per Split Plan',
 'Z-C1D-GC-ETCH2',
 'Z-GC-ETCH-ME1-45 (BT 12 sec)',
 'Z-LDMOS-OAA',
```

## Recipes run in RESP1

In [33]:

```
# show the equip names by which the entry has been done in DMIS
resp1_equip_names = [e for e in equip_list_unique if "RESP1" in e]
resp1_equip_names
```

Out[33]:

```
['RESP1A', 'RESP1B', 'CLRESP1']
```

In [34]:

```
# create df for "EquipID" col values as `*_equip_names` list
df_resp1 = df[df["EquipID"].isin(resp1_equip_names)]

# first 5 rows
df_resp1.head()
```

Out[34]:

| | EquipID | RecipeUsedOnEquip | ActualDateIn | LotId |
|---|---|---|---|---|
| **8667** | CLRESP1 | Done at TSL | 07-04-2016 17:18 | S16060002.F4 |
| **8668** | CLRESP1 | barch etch | 07-04-2016 14:20 | S16060002.F5 |
| **8669** | CLRESP1 | 1(BARCER1),5(BARCER2),9(BARCER9)) | 05-04-2016 12:18 | S16060002.F2 |
| **8670** | CLRESP1 | 1(BARCER1),5(BARCER2),9(BARCER9)) | 05-04-2016 12:15 | S16060002.F2 |
| **8671** | CLRESP1 | 1(BARCER1),5(BARCER2),9(BARCER9)) | 05-04-2016 12:11 | S16060002.F2 |

In [35]:

```
# get the shape (row, col) of dataframe
df_resp1.shape
```

Out[35]:

```
(3251, 4)
```

In [36]:

```
# fetch all the unique recipes in this equipment/chamber by `*_equip_names` list
recipes_resp1 = list(set(df_resp1["RecipeUsedOnEquip"].to_list()))
recipes_resp1
```

Out[36]:

```
['Z-CCD-CS-BC100',
 'SPACER_OE_13s',
 'barch etch',
 'BARCER1-28S',
 'Z-VL-13-NOASH',
 ':Z-VIA-DOE13',
 'Z-VIA_DOE13',
 'Z-RAA-ETCH',
 '1(BARCER1),5(BARCER2),9(BARCER9))',
 'Recipe:Z-CS-DOE1',
 'VIA ETCH',
 'z via last mim',
 'AS per Split Plan',
 '(wF# 18, DOE-Z-CS-DOE7)(wF# 23, DOE-Z-CS-DOE8) (wF# 24, DOE-Z-CS-DOE9)(w
F# 25, DOE-Z-CS-DOE10)',
 'SPACER-C1D',
 ': BARCER1-33S',
 'Z-CCD-VL-DOE1',
```