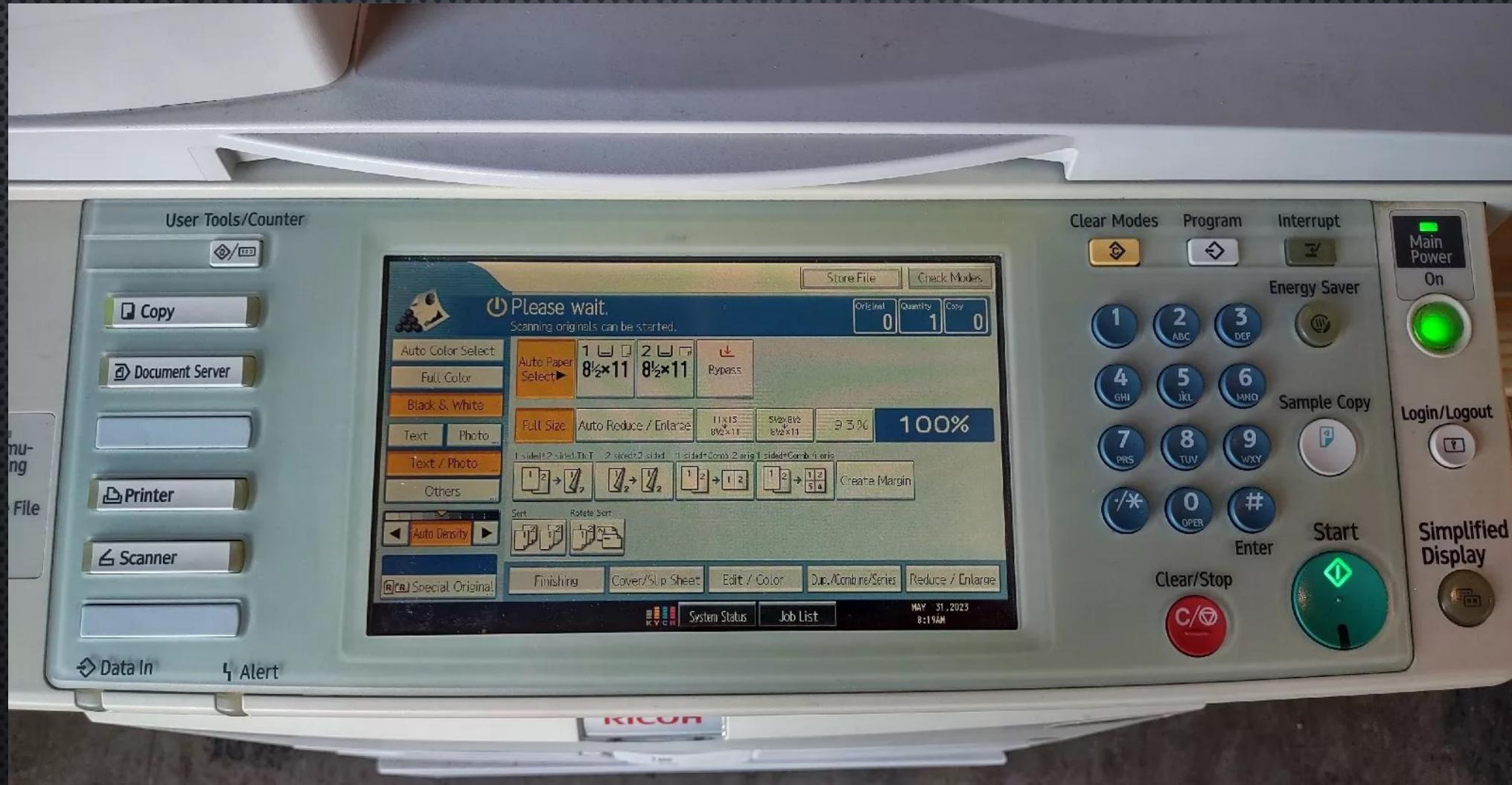


# **análisis de ciertos datos de facturación y gastos en la Xunta de Galicia**

Marcos Fernández Arias

Xunta de Galicia

# Orígenes de datos: contadores de uso de las impresoras



# Orígenes de datos: contadores de uso de las impresoras

Printers		All	Toners	Counters	Info	Search...	
Date	MAC Address	Serial	Model	Black	Cyan	Magenta	Yellow
2019-12-05 12:39:47	00-15-88-48-48-8B	Z51P6LACT0005SA	Samsung ML-45...	79%			
2020-03-03 14:24:41	★ 00-15-88-48-48-27	Z51P6LACT00036B	Samsung ML-45...	58%			
2020-03-03 14:24:40	★ 00-15-88-48-48-06	Z51P6LACT0005AH	Samsung ML-45...	1%			
2020-03-03 14:25:23	★ 00-15-88-48-48-CA	E6100042053681	Brother HL-41...	57%	88%	60%	68%
2020-03-03 14:25:18	★ 00-15-88-48-48-A2	A7100127000344	Generic 36BW-8	89%			
2020-03-03 14:25:28	★ 00-15-88-48-48-F4	A7100127000458	Generic 36BW-8	87%			
2020-03-03 14:25:28	★ 00-15-88-48-48-31	A1100041000799	KONICA MINOLT...	94%			
2020-03-03 14:24:47	★ 00-15-88-48-48-04	A5100121000677	KONICA MINOLT...	43%	18%	2%	82%
2020-03-03 14:24:52	★ 00-15-88-48-48-BB	A5100121000855	KONICA MINOLT...	2%	24%	16%	29%
2020-03-03 14:24:38	★ 00-15-88-48-48-14	A5100121000191	KONICA MINOLT...	34%	57%	69%	60%

Total rows: 78   ★ New Printer   ● Online   ● Offline   ● Other

1 2 3 4 5 6 7 8

# Procesamiento de un fichero CSV que contiene errores de formato y no resulta directamente legible

asimetrías en el cierre de comillas

15425	20240912,	18,	10.50.8.69,	LEXMARK_CO_01,	"75288270F29DC",	"Lexmark XC4150",	33850 ,	135052
15426	20240912,	18,	10.51.106.47,	LEXMARK_CO_01,	"75289020F3LGN",	"Lexmark XC4150",	57262 ,	82500
15427	20240912,	18,	10.51.106.83,	LEXMARK_CO_01,	"75289020F3LFX",	"Lexmark XC4150",	73538 ,	68387
15428	20240912,	18,	10.51.106.87,	LEXMARK_CO_01,	"50288330F31XV",	"Lexmark C4150",	84302 ,	100575
15429	20240912,	18,	10.51.121.207,	RICOH_BN_01,	"5212P701945",	"P501",	23983 ,	
15430	20240912,	18,	10.51.149.123,	LEXMARK_BN_01,	"70156PHH16R5D",	"Lexmark XM1145",	115930 ,	
15431	20240912,	18,	10.51.149.53,	LEXMARK_BN_01,	"70156PHH1659W",	"Lexmark XM1145",	575960 ,	
15432	20240912,	18,	10.51.149.76,	LEXMARK_BN_01,	"70166PHH09F6D",	"Lexmark XM3150",	394598 ,	
15433	20240912,	18,	10.51.15.204,	LEXMARK_CO_01,	"5028652010Y32",	"Lexmark C4150",	20234 ,	76313
15434	20240912,	18,	10.51.15.224,	LEXMARK_BN_01,	"7018830001V61",	"Lexmark XM3250",	22225 ,	
15435	20240912,	18,	10.51.151.45,	LEXMARK_BN_01,	"45147PHH08WB9C",	"Lexmark M1145",	5929 ,	
15436	20240912,	18,	10.51.151.52,	LEXMARK_BN_01,	"70166PHH099YB",	"Lexmark XM3150",	216924 ,	
15437	20240912,	18,	10.51.151.61,	LEXMARK_BN_01,	"70156PHH165BF",	"Lexmark XM1145",	371128 ,	
15438	20240912,	18,	10.51.151.64,	LEXMARK_BN_01,	"70156PHH165ND",	"Lexmark XM1145",	2446 ,	
15439	20240912,	18,	10.51.151.66,	LEXMARK_BN_01,	"70156PHH165LV",	"Lexmark XM1145",	20561 ,	
15440	20240912,	18,	10.51.152.111,	LEXMARK_BN_01,	"7017820101N41",	"Lexmark XM1246",	11480 ,	
15441	20240912,	18,	10.51.152.118,	LEXMARK_BN_01,	"70166PHH09921",	"Lexmark XM3150",	220565 ,	
15442	20240912,	18,	10.51.152.56,	LEXMARK_BN_01,	"70166PHH099Y4",	"Lexmark XM3150",	447764 ,	
15443	20240912,	18,	10.51.152.74,	LEXMARK_BN_01,	"70166PHH09F42",	"Lexmark XM3150",	108769 ,	
15444	20240912,	18,	10.51.152.79,	LEXMARK_BN_01,	"70156PHH16R3C",	"Lexmark XM1145",	45434 ,	
15445	20240912,	18,	10.51.152.80,	LEXMARK_BN_01,	"70156PHH16T80",	"Lexmark XM1145",	56687 ,	
15446	20240912,	18,	10.51.152.82,	LEXMARK_BN_01,	"45147PHH08WB8L",	"Lexmark M1145",	7774 ,	
15447	20240912,	18,	10.51.152.93,	LEXMARK_BN_01,	"46008210057VV",	"Lexmark M1246",	15841 ,	
15448	20240912,	18,	10.51.154.108,	LEXMARK_BN_01,	"7018039008BX65",	"Lexmark XM3250",	24283 ,	
			10.51.154.123,	LEXMARK_BN_01,	"70166PHH09921",	"Lexmark XM3150",	165283 ,	

# Procesamiento de un fichero CSV que contiene errores de formato y no resulta directamente legible

atributos que contienen comas o comillas

1099426	20240926,	20,	10.51.82.201,	BROTHER_BN_01,	"E70650E5N122050",	"BRN30055C837240",	07840 ,
1099427	20240926,	20,	10.51.82.202,	BROTHER_BN_01,	"E70650E5N122701",	"BRN30055C837249",	105185 ,
1099428	20240926,	20,	10.51.82.203,	BROTHER_BN_01,	"E70650E5N122704",	"BRN30055C837222",	13968 ,
1099429	20240926,	20,	10.51.82.204,	BROTHER_BN_01,	"E70650E5N122705",	"BRN30055C83722F",	55064 ,
1099430	20240926,	20,	10.51.82.205,	BROTHER_BN_01,	"E70650E5N122706",	"BRN30055C8370EF",	59505 ,
1099431	20240926,	20,	10.51.82.206,	RICOH_BN_01,	"5212P702039",	"P 501",	5717 ,
1099432	20240926,	20,	10.51.82.250,	RICOH_BN_01,	"4452RB40031",	"IM 5000",	29981 ,
1099433	20240926,	20,	10.51.85.102,	XEROX_CO_01,	"3185025046",	"Xerox Phaser 6600DN; Net 90.66, ESS 201305012027, IOT 02.00.01",	93772 , 257800
1099434	20240926,	20,	10.51.85.103,	RICOH_CO_01,	"Q7088602013",	"SP C410DN",	299289 , 211205
1099435	20240926,	20,	10.51.85.104,	SAMSUNG_BN_01,	"ZDGSB8GJ4A013KA",	"SL-M4020ND",	205092 ,
1099436	20240926,	20,	10.51.85.106,	SAMSUNG_BN_01,	"ZDGSB8GJ4A012KY",	"SL-M4020ND",	29869 ,
1099437	20240926,	20,	10.51.85.86,	KONICA_BN_01,	"AA6T021000911",	"KONICA MINOLTA bizhub 558e",	181941 ,
1099438	20240926,	20,	10.51.86.46,	RICOH_BN_01,	"3382P150679",	"IM 350",	1821 ,
1099439	20240926,	20,	10.51.93.200,	RICOH_CO_01,	"3132M600061",	"IM C5500",	51436 , 28853
1099440	20240926,	20,	10.51.93.45,	RICOH_CO_01,	"5322X529886",	"P C600",	7987 , 7882
1099441	20240926,	20,	10.51.93.46,	RICOH_BN_01,	"3382P950564",	"IM 350",	3398 ,
1099442	20240926,	20,	10.51.99.110,	RICOH_CO_01,	"C748JA00338",	"MP C5504ex",	185603 ,
1099443	20240926,	20,	10.51.99.111,	RICOH_BN_01,	"T588H601063",	"SP C410DN",	299289 , 211205

Se realizan dos lecturas:

- una mediante readr::read\_csv
- otra mediante readr::read\_lines

Se identifican las líneas que no han sido correctamente leídas mediante readr::read\_csv.

```
1589 # pacman::p_load(archive)
1590 snmp_0 <- archive::archive_read(
1591   "origen/SNMP/snmp-2024-09.7z",
1592   file = "snmp-2024-09.txt") %>%
1593   read_csv(col_names = FALSE) %>%
1594   set_names(
1595     c("fecha", "hora", "ip", "name", "serial", "model", "c_bn", "c_color")
1596   ) %>%
1597   rowid_to_column() %>%
1598   filter(!is.na(fecha) & !is.na(serial)) %>%
1599   mutate(
1600     fecha = ymd(fecha),
1601     hora = as.integer(hora),
1602     # c_bn = as.integer(c_bn),
1603     # c_color = as.integer(c_color)
1604   )
1605
1606 lines <- archive::archive_read(
1607   "origen/SNMP/snmp-2024-09.7z",
1608   file = "snmp-2024-09.txt") %>%
1609   read_lines() %>%
1610   as_tibble_col(column_name = "txt") %>%
1611   rowid_to_column()
1612 lines
1613
1614 # líneas que no contienen datos válidos sino un mensaje de error ====
1615 lines %>%
1616   slice(setdiff(1:nrow(lines), snmp_0$rowid)) %>%
1617   count(txt)
1618
1619
1620 # líneas que no se han parseado bien por contener comas dentro de strings ====
1621 rowids_necesario_repetir_parseo <-
1622   snmp_0 %>%
1623   filter(!is.na(c_color) & is.na(parse_integer(c_color))) %>%
1624   pull(rowid)
1625 length(rowids_necesario_repetir_parseo)
1626 rowids_necesario_repetir_parseo
1627
```

Se realizan dos lecturas:

- una mediante `readr::read_csv`
- otra mediante `readr::read_lines`

Se identifican las líneas que no han sido correctamente leídas mediante `readr::read_csv`.

Y para las erróneas, se realiza un nuevo procesamiento mediante otra librería (`base::read.csv`) y distintas especificaciones.

```
1645 lines %>%
1646   filter(rowid %in% rowids_necesario_repetir_parseo) %>%
1647   head() %>%
1648   print(width = Inf)
1649 lines_reparseadas <- read.csv(
1650   lines %>%
1651   filter(rowid %in% rowids_necesario_repetir_parseo) %>%
1652   pull(txt) %>%
1653   textConnection(),
1654   header = FALSE
1655 ) %>%
1656 as_tibble() %>%
1657 set_names(
1658   c("fecha", "hora", "ip", "name", "serial", "model", "c_bn", "c_color")
1659 ) %>%
1660 mutate(
1661   across(where(is.character), str_squish),
1662   fecha = ymd(fecha),
1663   hora = as.integer(hora),
1664   serial = as.character(serial),
1665   rowid = rowids_necesario_repetir_parseo
1666 ) %>%
1667 relocate(rowid)
1668 lines_reparseadas
1669
```

Se realizan dos lecturas:

- una mediante `readr::read_csv`
- otra mediante `readr::read_lines`

Se identifican las líneas que no han sido correctamente leídas mediante `readr::read_csv`.

Y para las erróneas, se realiza un nuevo procesamiento mediante otra librería (`base::read.csv`) y distintas especificaciones.

Finalmente se unen los dataframes mediante `dplyr::bind_rows`.

```
1670 snmp1 <- snmp_0 %>%
1671   filter(
1672     !rowid %in% rowids_necesario_repetir_parseo
1673   ) %>%
1674   mutate(
1675     serial = str_replace(serial, "-.*$", ""),
1676     c_bn = parse_number(c_bn),
1677     c_color = parse_number(c_color),
1678     model = model %>%
1679       str_remove(regex("\\bmultifunction.*$", ignore_case = TRUE)) %>%
1680       str_remove(regex("\\bseries\\b", ignore_case = TRUE)) %>%
1681       str_remove(";.*$") %>%
1682       str_squish()
1683   ) %>%
1684   bind_rows(
1685     lineas_reparseadas
1686   ) %>%
1687   relocate(c(serial, model), .after = hora) %>%
1688   filter(!str_detect(name, "_TOT$")) %>%
1689   filter(
1690     length(unique(name)) == 1 |
1691     ! name %in% c("RICOH_BN_01", "HP_TO_TOT", "BROTHER_TO_TOT",
1692                "SAMSUNG_BN_02"),
1693     .by = serial
1694   ) %>%
1695   filter(c_bn > 0) %>%
1696   verify(any(serial == "G195J700012")) %>%
1697   distinct(across(-rowid), .keep_all = TRUE) %>%
1698   arrange(fecha, hora, serial, name) %>%
1699   mutate(
1700     model = last(model),
1701     .by = serial
1702   )
```

Se realizan dos lecturas:

- una mediante `readr::read_csv`
- otra mediante `readr::read_lines`

Se identifican las líneas que no han sido correctamente leídas mediante `readr::read_csv`.

Y para las erróneas, se realiza un nuevo procesamiento mediante otra librería (`base::read.csv`) y distintas especificaciones.

Finalmente se unen los dataframes mediante `dplyr::bind_rows`.

Con lo que llegamos al conjunto de datos pretendido: la lectura del fichero CSV a pesar de los errores de formato.

```
> snmpl
# A tibble: 1,309,306 × 8
  fecha      hora serial      model ip      name      c_bn c_color
  <date>    <int> <chr>      <chr> <chr> <chr>    <dbl> <dbl>
1 2024-09-03 12 0AQEBJEJC0001NW C3010ND 10.71.54.95 SAMSUNG_CO_01 3123 12176
2 2024-09-03 12 0AQEBJEK10001ZE C3010ND 10.50.20.45 SAMSUNG_CO_01 754 645
3 2024-09-03 12 0AQEBJEK100024F C3010ND 10.50.36.168 SAMSUNG_CO_01 1217 1104
4 2024-09-03 12 0AQEBJEK10002GE C3010ND 10.51.64.200 SAMSUNG_CO_01 4468 3890
5 2024-09-03 12 0AQEBJEK10002JH C3010ND 10.61.26.91 SAMSUNG_CO_01 6025 21742
6 2024-09-03 12 0AQEBJEK100031K C3010ND 10.50.36.172 SAMSUNG_CO_01 1822 6636
7 2024-09-03 12 0AQEBJEK1000AXJ C3010ND 10.61.8.119 SAMSUNG_CO_01 9223 45843
8 2024-09-03 12 0AQEBJEM300003K C3010ND 10.50.25.48 SAMSUNG_CO_01 1805 7574
9 2024-09-03 12 0AQEBJEM300006P C3010ND 10.50.70.47 SAMSUNG_CO_01 1189 5015
10 2024-09-03 12 0AQEBJEM300007B C3010ND 10.50.66.113 SAMSUNG_CO_01 1686 5734
11 2024-09-03 12 3109R130240 IM C3000 10.56.3.136 RICOH_CO_01 169607 167680
12 2024-09-03 12 3109R431579 IM C3000 10.51.20.241 RICOH_CO_01 32693 136513
13 2024-09-03 12 3109R431593 IM C3000 10.51.20.197 RICOH_CO_01 30772 145435
14 2024-09-03 12 3109R830761 IM C3000 10.50.13.157 RICOH_CO_01 254890 164172
15 2024-09-03 12 3109R830772 IM C3000 10.50.19.198 RICOH_CO_01 294738 305252
16 2024-09-03 12 3109R930499 IM C3000 10.51.20.208 RICOH_CO_01 86376 202413
17 2024-09-03 12 3122MA10434 IM C4500 10.65.105.170 RICOH_CO_01 24629 24471
18 2024-09-03 12 3122M110450 IM C4500 10.54.176.45 RICOH_CO_01 1943 208
19 2024-09-03 12 3122M110536 IM C4500 10.55.244.45 RICOH_CO_01 3692 4625
20 2024-09-03 12 3122M110559 IM C4500 10.51.32.47 RICOH_CO_01 29749 33648
21 2024-09-03 12 3122M110567 IM C4500 10.52.118.45 RICOH_CO_01 15707 20267
22 2024-09-03 12 3122M110587 IM C4500 10.57.23.46 RICOH_CO_01 5790 5191
23 2024-09-03 12 3122M910625 IM C4500 10.232.50.174 RICOH_CO_01 1492 4646
24 2024-09-03 12 3130MC00070 IM C5500 10.55.8.200 RICOH_CO_01 115536 87685
25 2024-09-03 12 3130MC00071 IM C5500 10.56.7.119 RICOH_CO_01 26674 30267
26 2024-09-03 12 3130MC00079 IM C5500 10.55.2.248 RICOH_CO_01 65152 95459
27 2024-09-03 12 3130MC00081 IM C5500 10.65.136.62 RICOH_CO_01 21976 17936
28 2024-09-03 12 3130MC00090 IM C5500 10.55.2.183 RICOH_CO_01 84862 18811
29 2024-09-03 12 3130MC00091 IM C5500 10.66.167.57 RICOH_CO_01 68363 94138
30 2024-09-03 12 3130MC00106 IM C5500 10.55.12.51 RICOH_CO_01 67222 15070
31 2024-09-03 12 3130MC00107 IM C5500 10.56.188.55 RICOH_CO_01 24857 20847
32 2024-09-03 12 3130MC00116 IM C5500 10.55.8.126 RICOH_CO_01 49281 9935
33 2024-09-03 12 3130MC00119 IM C5500 10.55.8.224 RICOH_CO_01 82641 24709
34 2024-09-03 12 3130MC00233 IM C5500 10.63.145.85 RICOH_CO_01 60903 2303
35 2024-09-03 12 3130MC00383 IM C5500 10.53.187.57 RICOH_CO_01 49224 15888
36 2024-09-03 12 3130MC00393 IM C5500 10.51.28.205 RICOH_CO_01 54328 15314
37 2024-09-03 12 3130MC00396 IM C5500 10.50.19.235 RICOH_CO_01 11483 6589
38 2024-09-03 12 3130MC00408 IM C5500 10.55.188.47 RICOH_CO_01 55466 21105
39 2024-09-03 12 3130MC00411 IM C5500 10.53.188.47 RICOH_CO_01 67790 43604
40 2024-09-03 12 3130MC00069 IM C5500 10.53.188.47 RICOH_CO_01 10394 11347
```

Cada uno de estos dataframes contienen millones de filas.

Por lo que, su almacenamiento en disco ya no se realiza en texto plano sino en formato Parquet, con compresión zstd.

```
> glimpse(snmp1)
Rows: 1,309,306
Columns: 8
$ fecha    <date> 2024-09-03, 2024-09-03, 2024-09-03, 2024-09-03, 2024-09-03, 2024-09-03, 2
$ hora     <int> 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12
$ serial   <chr> "0AQEBJEJC0001NW", "0AQEBJEK10001ZE", "0AQEBJEK100024F", "0AQEBJEK10002GE"
$ model    <chr> "C3010ND", "C3010ND", "C3010ND", "C3010ND", "C3010ND", "C3010ND", "C3010ND"
$ ip       <chr> "10.71.54.95", "10.50.20.45", "10.50.36.168", "10.51.64.200", "10.61.26.91
$ name     <chr> "SAMSUNG_CO_01", "SAMSUNG_CO_01", "SAMSUNG_CO_01", "SAMSUNG_CO_01", "SAMSU
$ c_bn     <dbl> 3123, 754, 1217, 4468, 6025, 1822, 9223, 1805, 1189, 1686, 169607, 32693,
$ c_color  <dbl> 12176, 645, 1104, 3890, 21742, 6636, 45843, 7574, 5015, 5734, 167680, 1365
> pryr::object_size(snmp1)
73.91 MB
> nanoparquet::write_parquet(
+   snmp1,
+   "origen/SNMP/snmp-2024-10.parquet",
+   compression = "zstd"
+ )
```

# Procesamiento de múltiples fichero CSV que tienen el mismo formato

 2024.csv	7.563.851
 2023-a.csv	16.362.625
 2023-b.csv	1.140.651
 2022.csv	16.003.944
 2021-20.csv	13.503.126
 2019.csv	7.744.241
 2018.csv	9.952.735
 2017.csv	11.091.356
 2016.csv	11.339.763
 2024-05.csv	1.953.667
 2024-06.csv	1.291.536
 2024-07.csv	1.659.247
 2024-08.csv	1.119.061
 2024-09.csv	1.967.134

Cod factura SEF	Ano	Ano-Mes	Día	Número de factura	Importe factura	Descripción
8104222	2024	2024/09	2024-08-30	24SF10214008	2.65	Dotación
8104222	2024	2024/09	2024-08-30	24SF10214008	2.65	Dotación
8162960	2024	2024/09	2024-09-26	24SF10232948	3.90	
8104362	2024	2024/09	2024-08-30	24SF10214010	6.88	Dotación
8104362	2024	2024/09	2024-08-30	24SF10214010	6.88	Dotación
8164733	2024	2024/09	2024-09-27	24SF10236999	19.34	
8104353	2024	2024/09	2024-08-30	24SF10213967	26.20	Dotación
8099235	2024	2024/09	2024-08-30	24SF10206562	28.47	
8140045	2024	2024/09	2024-03-25	24SF10066959	30.24	Dotación
8140045	2024	2024/09	2024-03-25	24SF10066959	30.24	Dotación
8139924	2024	2024/09	2024-09-17	24SF10222156	30.54	DOTACION
8139924	2024	2024/09	2024-09-17	24SF10222156	30.54	DOTACION
8108483	2024	2024/09	2024-08-30	24SF10216629	41.43	Dotación
8108483	2024	2024/09	2024-08-30	24SF10216629	41.43	Dotación
8139923	2024	2024/09	2024-09-17	24SF10222153	56.99	DOTACION
8139923	2024	2024/09	2024-09-17	24SF10222153	56.99	DOTACION
8104240	2024	2024/09	2024-08-30	24SF10213995	57.87	Dotación
8139922	2024	2024/09	2024-09-17	24SF10222155	62.41	DOTACION
8139922	2024	2024/09	2024-09-17	24SF10222155	62.41	DOTACION
8164736	2024	2024/09	2024-09-27	24SF10237358	64.94	
8164735	2024	2024/09	2024-09-27	24SF10237354	76.48	
8104221	2024	2024/09	2024-08-30	24SF10214005	84.80	Dotación
8104221	2024	2024/09	2024-08-30	24SF10214005	84.80	Dotación

# Procesamiento de múltiples fichero CSV que tienen el mismo formato

2024.csv	7.563.851
2023-a.csv	16.362.625
2023-b.csv	1.140.651
2022.csv	16.003.944
2021-20.csv	13.503.126
2019.csv	7.744.241
2018.csv	9.952.735
2017.csv	11.091.356
2016.csv	11.339.763
2024-05.csv	1.953.667
2024-06.csv	1.291.536
2024-07.csv	1.659.247
2024-08.csv	1.119.061
2024-09.csv	1.967.134

```
744 # *****
745 # leer ficheros de contadores y desglose facturación ===
746 # declarados por las empresas prestadoras del servicio
747
748 # fs::dir_ls("origen/detalles_facturacion", type = "file")
749 detalles_facturas <- fs::dir_map(
750   # "origen/detalles_facturacion",
751   "origen/detalles_facturacion/2024-07-16",
752   # "origen/detalles_facturacion/2024-09-16",
753   type = "file",
754   \(x) data.table::fread(x, #verbose = TRUE,
755                          showProgress = TRUE,
756                          data.table = FALSE) %>%
757     janitor::clean_names() %>%
758     as_tibble() %>%
759     mutate(
760       numero_factura = as.character(numero_factura),
761       fichero = fs::path_file(x)
762     ) %>%
763     select(-any_of("id_fecha_lectura"))
764 )
765 ) %>%
766 bind_rows() %>%
```

# Procesamiento de múltiples fichero CSV que tienen el mismo formato

```
> detalles_facturas %>% select(fichero, numero_factura:acuerdo_marco)
# A tibble: 66,368 x 7
  fichero                numero_factura fecha_factura oficina_contable organo_gestor unidad_tramitadora acuerdo_marco
  <chr>                  <chr>          <date>         <chr>              <chr>          <chr>          <chr>
1 Xustiza historico_202405.csv 23SF226506    2023-11-30    A12009376         A12009361     A12009361     99/2021
2 Xustiza historico_202405.csv 23SF193619    2023-10-25    A12009376         A12009361     A12009361     99/2021
3 xustiza_202408.csv          24SF10213964 2024-08-30    A12009376         A12009361     A12009361     99/2021
4 Xustiza historico_202405.csv 23SF193615    2023-10-25    A12009376         A12009361     A12009361     99/2021
5 Xustiza historico_202405.csv 23SF193615    2023-10-25    A12009376         A12009361     A12009361     99/2021
6 xustiza_202408.csv          24SF10213964 2024-08-30    A12009376         A12009361     A12009361     99/2021
7 Xustiza historico_202405.csv 23SF165917    2023-09-25    A12009376         A12009361     A12009361     99/2021
8 Derivados historico_202405.csv 23SF237169    2023-12-27    A12009376         A12024248     A12024248     62/2016
9 xustiza_202407.csv          24SF10188585 2024-07-31    A12009376         A12009361     A12009361     99/2021
10 Xustiza historico_202405.csv 23SF165916    2023-09-25    A12009376         A12009361     A12009361     99/2021
11 Xustiza historico_202405.csv 23SF193617    2023-10-25    A12009376         A12009361     A12009361     99/2021
12 xustiza_202408.csv          24SF10213964 2024-08-30    A12009376         A12009361     A12009361     99/2021
13 Derivados historico_202405.csv 23SF193462    2023-10-25    A12009376         A12024986     GE0010889     62/2016
14 Xustiza historico_202405.csv 23SF140281    2023-07-31    A12009376         A12009361     A12009361     99/2021
15 Xustiza historico_202405.csv 23SF193617    2023-10-25    A12009376         A12009361     A12009361     99/2021
16 Xustiza historico_202405.csv 24SF10134197 2024-05-31    A12009376         A12009361     A12009361     99/2021
17 Derivados historico_202405.csv 23SF193462    2023-10-25    A12009376         A12024986     GE0010889     62/2016
18 Derivados historico_202405.csv 23SF248769    2023-12-29    A12009376         A12024991     GE0011910     62/2016
19 Derivados historico_202405.csv 23SF193612    2023-10-25    A12009376         A12024248     A12024248     62/2016
20 Derivados_202406.csv        24SF10161628 2024-06-28    A12009376         A12034492     A12034492     62/2016
21 Derivados historico_202405.csv 23SF208364    2023-11-20    A12009376         A12025001     GE0017979     62/2016
22 Xustiza historico_202405.csv 23SF193617    2023-10-25    A12009376         A12009361     A12009361     99/2021
23 Derivados historico_202405.csv 23SF193592    2023-10-25    A12009376         A12009360     GE0010870     62/2016
24 Derivados historico_202405.csv 23SF221526    2023-11-28    A12009376         A12009360     GE0010870     62/2016
25 Xustiza historico_202405.csv 24SF10107056 2024-04-30    A12009376         A12009361     A12009361     99/2021
26 Xustiza historico_202405.csv 844155480    2023-02-22    A12009376         GE0017979     GE0017979     37/2019
27 Xustiza historico_202405.csv 23SF1936641    2024-01-01    A12009376         A12009361     A12009361     99/2021
```

# Gráficos de apoyo explicativos

```
888 group_by(numero_serie) %>%
891 mutate(
892   eval_bn =
893     case_when(
894       lectura_inicial_bn > lectura_final_bn ~ "-",
895       !is.na(lead(fecha_factura)) &
896         fecha_factura > lead(fecha_factura) + days(3) ~ "rectif",
897       is.na(lag(lectura_final_bn)) ~ "",
898       lag(lectura_final_bn) == lectura_final_bn &
899         lectura_final_bn > lectura_inicial_bn ~ "dup",
900       bn > 0 & lag(lectura_final_bn) < lectura_inicial_bn ~ "<",
901       bn > 0 & lag(lectura_final_bn) > lectura_inicial_bn ~ ">",
902       bn > 0 & lectura_final_bn < lead(lectura_inicial_bn) ~ "<.",
903       bn > 0 & lectura_final_bn > lead(lectura_inicial_bn) ~ ">.",
904       .default = ""
905     ),
906   eval_color =
907     case_when(
908       lectura_inicial_color > lectura_final_color ~ "-",
909       !is.na(lead(fecha_factura)) &
910         fecha_factura > lead(fecha_factura) + days(3) ~ "rectif",
911       is.na(lag(lectura_final_color)) ~ "",
912       lag(lectura_final_color) == lectura_final_color &
913         lectura_final_color > lectura_inicial_color ~ "dup",
914       color > 0 & lag(lectura_final_color) < lectura_inicial_color ~ "<",
915       color > 0 & lag(lectura_final_color) > lectura_inicial_color ~ ">",
916       color > 0 & lectura_final_color < lead(lectura_inicial_color) ~ "<.",
917       color > 0 & lectura_final_color > lead(lectura_inicial_color) ~ ">.",
918       .default = ""
919     )
920 ) %>%
921 ungroup()
```

Uso de funciones ventana (dplyr::lag y dplyr::lead) para detectar irregularidades en las mediciones de los contadores

# Gráficos de apoyo explicativos

```
1365 g_1 <- datos %>%
1366   filter(!is.na(bn)) %>%
1367   mutate(
1368     lectura_final_bn = if_else(lectura_inicial_bn == lectura_final_bn,
1369                               lectura_final_bn + 1, lectura_final_bn),
1370     bn_category = case_when(
1371       eval_bn == "" ~ "gray40",
1372       eval_bn == "-" ~ "yellow",
1373       eval_bn == "<." ~ "darkgreen",
1374       eval_bn == ">." ~ "darkblue",
1375       eval_bn == "<." ~ "green",
1376       eval_bn == ">." ~ "blue",
1377       eval_bn == "rectif" ~ "darkorchid",
1378       eval_bn == "dup" ~ "red",
1379       .default = "pink"
1380     )
1381   ) %>%
1382   ggplot(aes(y = fecha_lectura_inicio, x = lectura_inicial_bn)) +
1383   geom_segment(aes(xend = lectura_final_bn,
1384                  yend = fecha_lectura_fin,
1385                  color = bn_category),
1386              alpha = .8,
1387              linewidth = 1.1) +
1388   geom_point(aes(color = bn_category), size = 2) +
1389   geom_text(aes(label = numero_factura),
1390            hjust = -.15, vjust = 0,
1391            # angle = 30,
1392            size = 3) +
1393   scale_x_continuous(
1394     labels = label_comma(big.mark = ".", decimal.mark = ","),
1395     # minor_breaks = seq(7000000, 70000000, by = 10000000),
1396     # major_breaks = seq(70000000, 700000000, by = 100000000),
1397     # minor_labels = "",
1398     # major_labels = ""
1399   )
```

# Gráficos de apoyo explicativos

irregularidades en contador de páginas en bn

