In [1]:

```
import pandas as pd
```

In [2]:

```
df = pd.read_excel("trains-props.xlsx")
```

In [3]:

```
df
```

Out[3]:

	third_ac	arrival	from_station_code	name	zone	chair_car	first_class	duration_m	sl
0	0	12:15:00	JAT	Jammu Tawi Udhampur Special	NR	0	0	35.0	
1	0	08:35:00	UHP	UDHAMPUR JAMMUTAWI DMU	NR	0	0	50.0	
2	0	17:50:00	JAT	JAT UDAHMPUR DMU	NR	0	0	35.0	
3	0	19:50:00	UHP	UDHAMPUR JAMMUTAWI DMU	NR	0	0	30.0	
4	1	12:30:00	BDTS	Mumbai BandraT-Bikaner SF Special	NWR	0	0	55.0	
...	...	...	...	...	...	...	...	...	
5203	0	19:20:00	NDLS	New Delhi Palwal Ladies Special	?	0	0	30.0	
5204	0	23:25:00	VSKP	Visakhapatnam Rajahmundry Pass	?	0	0	15.0	
5205	0	20:20:00	MAO	Madgaon Karwar Passenger	?	0	0	20.0	
5206	0	04:30:00	DNR	Danapur Giridih Express	?	0	0	30.0	
5207	0	12:30:00	BRWD	Tribeni Link Express	?	0	0	0.0	

5208 rows × 20 columns

In [4]:

```
df["total time"] = df["duration_m"]/60 + df["duration_h"]
df
```

Out[4]:

	third_ac	arrival	from_station_code	name	zone	chair_car	first_class	duration_m	sl
0	0	12:15:00	JAT	Jammu Tawi Udhampur Special	NR	0	0	35.0	
1	0	08:35:00	UHP	UDHAMPUR JAMMUTAWI DMU	NR	0	0	50.0	
2	0	17:50:00	JAT	JAT UDAHMPUR DMU	NR	0	0	35.0	
3	0	19:50:00	UHP	UDHAMPUR JAMMUTAWI DMU	NR	0	0	30.0	
4	1	12:30:00	BDTS	Mumbai BandraT-Bikaner SF Special	NWR	0	0	55.0	
...	...	...	...	...	...	...	...	...	
5203	0	19:20:00	NDLS	New Delhi Palwal Ladies Special	?	0	0	30.0	
5204	0	23:25:00	VSKP	Visakhapatnam Rajahmundry Pass	?	0	0	15.0	
5205	0	20:20:00	MAO	Madgaon Karwar Passenger	?	0	0	20.0	
5206	0	04:30:00	DNR	Danapur Giridih Express	?	0	0	30.0	
5207	0	12:30:00	BRWD	Tribeni Link Express	?	0	0	0.0	

5208 rows × 21 columns

In [5]:

```
df.drop(["duration_m", "duration_h"],axis=1)
df
```

Out[5]:

	third_ac	arrival	from_station_code	name	zone	chair_car	first_class	duration_m	sl
0	0	12:15:00	JAT	Jammu Tawi Udhampur Special	NR	0	0	35.0	
1	0	08:35:00	UHP	UDHAMPUR JAMMUTAWI DMU	NR	0	0	50.0	
2	0	17:50:00	JAT	JAT UDAHMPUR DMU	NR	0	0	35.0	
3	0	19:50:00	UHP	UDHAMPUR JAMMUTAWI DMU	NR	0	0	30.0	
4	1	12:30:00	BDTS	Mumbai BandraT-Bikaner SF Special	NWR	0	0	55.0	
...	...	...	...	...	...	...	...	...	
5203	0	19:20:00	NDLS	New Delhi Palwal Ladies Special	?	0	0	30.0	
5204	0	23:25:00	VSKP	Visakhapatnam Rajahmundry Pass	?	0	0	15.0	
5205	0	20:20:00	MAO	Madgaon Karwar Passenger	?	0	0	20.0	
5206	0	04:30:00	DNR	Danapur Giridih Express	?	0	0	30.0	
5207	0	12:30:00	BRWD	Tribeni Link Express	?	0	0	0.0	

5208 rows × 21 columns

In [6]:

```
df = df.drop(["duration_m", "duration_h"],axis=1)
df
```

Out[6]:

	third_ac	arrival	from_station_code	name	zone	chair_car	first_class	sleeper	from_
0	0	12:15:00	JAT	Jammu Tawi Udhampur Special	NR	0	0	0	
1	0	08:35:00	UHP	UDHAMPUR JAMMUTAWI DMU	NR	0	0	0	
2	0	17:50:00	JAT	JAT UDAHMPUR DMU	NR	0	0	0	
3	0	19:50:00	UHP	UDHAMPUR JAMMUTAWI DMU	NR	0	0	0	
4	1	12:30:00	BDTS	Mumbai BandraT-Bikaner SF Special	NWR	0	0	1	MUA
...	...	...	...	...	...	...	...	...	
5203	0	19:20:00	NDLS	New Delhi Palwal Ladies Special	?	0	0	0	
5204	0	23:25:00	VSKP	Visakhapatnam Rajahmundry Pass	?	0	0	0	VISI
5205	0	20:20:00	MAO	Madgaon Karwar Passenger	?	0	0	0	
5206	0	04:30:00	DNR	Danapur Giridih Express	?	0	0	1	
5207	0	12:30:00	BRWD	Tribeni Link Express	?	0	0	1	I

5208 rows × 19 columns

In [7]:

```
df = df.dropna()
df
```

Out[7]:

	third_ac	arrival	from_station_code	name	zone	chair_car	first_class	sleeper	from_
0	0	12:15:00	JAT	Jammu Tawi Udhampur Special	NR	0	0	0	
1	0	08:35:00	UHP	UDHAMPUR JAMMUTAWI DMU	NR	0	0	0	
2	0	17:50:00	JAT	JAT UDAHMPUR DMU	NR	0	0	0	
3	0	19:50:00	UHP	UDHAMPUR JAMMUTAWI DMU	NR	0	0	0	
4	1	12:30:00	BDTS	Mumbai BandraT-Bikaner SF Special	NWR	0	0	1	MUM
...	...	...	...	...	...	...	...	...	
5201	0	20:45:00	VG	Viramgam Valsad Passenger	WR	0	1	0	V
5202	0	08:10:00	CWA	Panchvalley Passenger Slip1	SCR	0	0	1	Chhin
5203	0	19:20:00	NDLS	New Delhi Palwal Ladies Special	?	0	0	0	
5204	0	23:25:00	VSKP	Visakhapatnam Rajahmundry Pass	?	0	0	0	VISH
5205	0	20:20:00	MAO	Madgaon Karwar Passenger	?	0	0	0	

4593 rows × 19 columns

In [8]:

```
df["zone"].value_counts()
```

Out[8]:

```
NR      597
SR      498
WR      433
NER     356
CR      333
ER      321
SCR     284
ECR     251
SER     234
NWR     224
SWR     218
NFR     201
?       196
NCR     129
ECoR    116
SECR    108
WCR      73
KR       21
Name: zone, dtype: int64
```

In [12]:

```
df1 = df[df["zone"].str.contains("?")==False]
df1
```

```
---------------------------------------------------------------------------
error                                     Traceback (most recent call l
ast)
Input In [12], in <cell line: 1>()
----> 1 df1 = df[df["zone"].str.contains("?")==False]
      2 df1

File /usr/lib/python3.10/site-packages/pandas/core/strings/accessor.py:
125, in forbid_nonstring_types.<locals>._forbid_nonstring_types.<locals
>.wrapper(self, *args, **kwargs)
    120     msg = (
    121         f"Cannot use .str.{func_name} with values of "
    122         f"inferred dtype '{self._inferred_dtype}'."
    123     )
    124     raise TypeError(msg)
--> 125 return func(self, *args, **kwargs)

File /usr/lib/python3.10/site-packages/pandas/core/strings/accessor.py:
1214, in StringMethods.contains(self, pat, case, flags, na, regex)
   1089 @forbid_nonstring_types(["bytes"])
   1090 def contains(self, pat, case=True, flags=0, na=None, regex=True
):
   1091     r"""
   1092     Test if pattern or regex is contained within a string of a
 Series or Index.
   1093
   (...)
   1212     dtype: bool
   1213     """
-> 1214     if regex and re.compile(pat).groups:
   1215         warnings.warn(
   1216             "This pattern is interpreted as a regular expressio
n, and has "
   1217             "match groups. To actually get the groups, use str.
extract.",
   1218             UserWarning,
   1219             stacklevel=find_stack_level(),
   1220         )
   1222     result = self._data.array._str_contains(pat, case, flags, n
a, regex)

File /usr/lib/python3.10/re.py:251, in compile(pattern, flags)
    249 def compile(pattern, flags=0):
    250     "Compile a regular expression pattern, returning a Pattern
 object."
--> 251     return _compile(pattern, flags)

File /usr/lib/python3.10/re.py:303, in _compile(pattern, flags)
    301 if not sre_compile.isstring(pattern):
    302     raise TypeError("first argument must be string or compiled p
attern")
--> 303 p = sre_compile.compile(pattern, flags)
    304 if not (flags & DEBUG):
    305     if len(_cache) >= _MAXCACHE:
    306         # Drop the oldest item
```

```
File /usr/lib/python3.10/sre_compile.py:764, in compile(p, flags)
    762 if isstring(p):
    763     pattern = p
--> 764     p = sre_parse.parse(p, flags)
    765 else:
    766     pattern = None

File /usr/lib/python3.10/sre_parse.py:948, in parse(str, flags, state)
    945 state.str = str
    947 try:
--> 948     p = _parse_sub(source, state, flags & SRE_FLAG_VERBOSE, 0)
    949 except Verbose:
    950     # the VERBOSE flag was switched on inside the pattern.  to b
e
    951     # on the safe side, we'll parse the whole thing again...
    952     state = State()

File /usr/lib/python3.10/sre_parse.py:443, in _parse_sub(source, state,
verbose, nested)
    441 start = source.tell()
    442 while True:
--> 443     itemsappend(_parse(source, state, verbose, nested + 1,
    444                        not nested and not items))
    445     if not sourcematch("|"):
    446         break

File /usr/lib/python3.10/sre_parse.py:668, in _parse(source, state, ver
bose, nested, first)
    666     item = None
    667 if not item or item[0][0] is AT:
--> 668     raise source.error("nothing to repeat",
    669                        source.tell() - here + len(this))
    670 if item[0][0] in _REPEATCODES:
    671     raise source.error("multiple repeat",
    672                        source.tell() - here + len(this))

error: nothing to repeat at position 0
```

In [13]:

```
df[~df.C.str.contains("?")]
df
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Input In [13], in <cell line: 1>()
----> 1 df[~df.C.str.contains("?")]
      2 df

File /usr/lib/python3.10/site-packages/pandas/core/generic.py:5583, in NDFrame.__getattr__(self, name)
   5576 if (
   5577     name not in self._internal_names_set
   5578     and name not in self._metadata
   5579     and name not in self._accessors
   5580     and self._info_axis._can_hold_identifiers_and_holds_name(name)
   5581 ):
   5582     return self[name]
-> 5583 return object.__getattribute__(self, name)

AttributeError: 'DataFrame' object has no attribute 'C'
```

In [18]:

```
df = df[~df.zone.str.contains("?")]
df
```

```
---------------------------------------------------------------------------
error                                     Traceback (most recent call l
ast)
Input In [18], in <cell line: 1>()
----> 1 df = df[~df.zone.str.contains("?")]
      2 df

File /usr/lib/python3.10/site-packages/pandas/core/strings/accessor.py:
125, in forbid_nonstring_types.<locals>._forbid_nonstring_types.<locals
>.wrapper(self, *args, **kwargs)
    120     msg = (
    121         f"Cannot use .str.{func_name} with values of "
    122         f"inferred dtype '{self._inferred_dtype}'."
    123     )
    124     raise TypeError(msg)
--> 125 return func(self, *args, **kwargs)

File /usr/lib/python3.10/site-packages/pandas/core/strings/accessor.py:
1214, in StringMethods.contains(self, pat, case, flags, na, regex)
   1089 @forbid_nonstring_types(["bytes"])
   1090 def contains(self, pat, case=True, flags=0, na=None, regex=True
):
   1091     r"""
   1092     Test if pattern or regex is contained within a string of a
 Series or Index.
   1093
   (...)
   1212     dtype: bool
   1213     """
-> 1214     if regex and re.compile(pat).groups:
   1215         warnings.warn(
   1216             "This pattern is interpreted as a regular expressio
n, and has "
   1217             "match groups. To actually get the groups, use str.
extract.",
   1218             UserWarning,
   1219             stacklevel=find_stack_level(),
   1220         )
   1222     result = self._data.array._str_contains(pat, case, flags, n
a, regex)

File /usr/lib/python3.10/re.py:251, in compile(pattern, flags)
    249 def compile(pattern, flags=0):
    250     "Compile a regular expression pattern, returning a Pattern
 object."
--> 251     return _compile(pattern, flags)

File /usr/lib/python3.10/re.py:303, in _compile(pattern, flags)
    301 if not sre_compile.isstring(pattern):
    302     raise TypeError("first argument must be string or compiled p
attern")
--> 303 p = sre_compile.compile(pattern, flags)
    304 if not (flags & DEBUG):
    305     if len(_cache) >= _MAXCACHE:
    306         # Drop the oldest item
```

```
File /usr/lib/python3.10/sre_compile.py:764, in compile(p, flags)
    762 if isstring(p):
    763     pattern = p
--> 764     p = sre_parse.parse(p, flags)
    765 else:
    766     pattern = None

File /usr/lib/python3.10/sre_parse.py:948, in parse(str, flags, state)
    945 state.str = str
    947 try:
--> 948     p = _parse_sub(source, state, flags & SRE_FLAG_VERBOSE, 0)
    949 except Verbose:
    950     # the VERBOSE flag was switched on inside the pattern.  to b
e
    951     # on the safe side, we'll parse the whole thing again...
    952     state = State()

File /usr/lib/python3.10/sre_parse.py:443, in _parse_sub(source, state,
verbose, nested)
    441 start = source.tell()
    442 while True:
--> 443     itemsappend(_parse(source, state, verbose, nested + 1,
    444                        not nested and not items))
    445     if not sourcematch("|"):
    446         break

File /usr/lib/python3.10/sre_parse.py:668, in _parse(source, state, ver
bose, nested, first)
    666     item = None
    667 if not item or item[0][0] is AT:
--> 668     raise source.error("nothing to repeat",
    669                        source.tell() - here + len(this))
    670 if item[0][0] in _REPEATCODES:
    671     raise source.error("multiple repeat",
    672                        source.tell() - here + len(this))

error: nothing to repeat at position 0
```

In [20]:

```
df.to_excel("until-totaltime.xlsx", index=False)
```

In [23]:

```
df = pd.read_excel("until-totaltime.xlsx")
df
```

Out[23]:

	third_ac	arrival	from_station_code	name	zone	chair_car	first_class	sleeper	from_
0	0	12:15:00	JAT	Jammu Tawi Udhampur Special	NR	0	0	0	
1	0	08:35:00	UHP	UDHAMPUR JAMMUTAWI DMU	NR	0	0	0	
2	0	17:50:00	JAT	JAT UDAHMPUR DMU	NR	0	0	0	
3	0	19:50:00	UHP	UDHAMPUR JAMMUTAWI DMU	NR	0	0	0	
4	1	12:30:00	BDTS	Mumbai BandraT-Bikaner SF Special	NWR	0	0	1	MUM
...	...	...	...	...	...	...	...	...	
4588	0	20:45:00	VG	Viramgam Valsad Passenger	WR	0	1	0	V
4589	0	08:10:00	CWA	Panchvalley Passenger Slip1	SCR	0	0	1	Chhin
4590	0	19:20:00	NDLS	New Delhi Palwal Ladies Special	NaN	0	0	0	
4591	0	23:25:00	VSKP	Visakhapatnam Rajahmundry Pass	NaN	0	0	0	VISH
4592	0	20:20:00	MAO	Madgaon Karwar Passenger	NaN	0	0	0	

4593 rows × 19 columns

In [24]:

```
df.head(20)
```

Out[24]:

	third_ac	arrival	from_station_code	name	zone	chair_car	first_class	sleeper	from_st
0	0	12:15:00	JAT	Jammu Tawi Udhampur Special	NR	0	0	0	J
1	0	08:35:00	UHP	UDHAMPUR JAMMUTAWI DMU	NR	0	0	0	
2	0	17:50:00	JAT	JAT UDAHMPUR DMU	NR	0	0	0	J
3	0	19:50:00	UHP	UDHAMPUR JAMMUTAWI DMU	NR	0	0	0	
4	1	12:30:00	BDTS	Mumbai BandraT-Bikaner SF Special	NWR	0	0	1	MUM
5	0	00:25:00	SSA	Sirsa - Hisar Passenger Special	NaN	0	0	0	
6	0	06:10:00	HSR	Hisar - Sirsa Passenger Special	NaN	0	0	0	
7	0	23:45:00	DNA	Rewari Degana Merta Special	NWR	0	0	0	
8	0	04:40:00	MTD	Degana Merta Special	NWR	0	0	0	MEF
9	1	17:50:00	JAT	JAMMU TAWI - BANDRA TERMINUS AC Suvidha Specia...	WR	0	0	1	J
10	1	21:55:00	DR	Chalukya Express	CR	0	0	1	MUM
11	1	04:25:00	CSTM	Konark Express	CR	0	0	1	M
12	1	03:55:00	BBS	Konark Express	CR	0	0	1	BHU
13	1	06:05:00	CSTM	Sahyadri Express	CR	0	0	1	M
14	1	11:55:00	KOP	Sahyadri Express	CR	0	0	1	C SI KOLH
15	0	12:05:00	MMR	Manmad Pune Express(Via Nasik)	CR	1	0	0	

	third_ac	arrival	from_station_code	name	zone	chair_car	first_class	sleeper	from_st
16	0	21:30:00	PUNE	Pune Manmad Express	CR	1	0	0	
17	1	04:55:00	CSTM	Mumbai CST-Chennai Mail	CR	0	0	1	M
18	0	20:25:00	CSTM	Koyna Express	CR	1	0	0	M
19	0	20:22:00	KOP	Koyna Express	CR	1	0	0	C SI KOLH

In [25]:

```
df.dropna()
```

Out[25]:

	third_ac	arrival	from_station_code	name	zone	chair_car	first_class	sleeper	from_
0	0	12:15:00	JAT	Jammu Tawi Udhampur Special	NR	0	0	0	
1	0	08:35:00	UHP	UDHAMPUR JAMMUTAWI DMU	NR	0	0	0	
2	0	17:50:00	JAT	JAT UDAHMPUR DMU	NR	0	0	0	
3	0	19:50:00	UHP	UDHAMPUR JAMMUTAWI DMU	NR	0	0	0	
4	1	12:30:00	BDTS	Mumbai BandraT-Bikaner SF Special	NWR	0	0	1	MUM
...	...	...	...	...	...	...	...	...	
4585	0	12:25:00	KDJR	Kendujhargarh-Bhubaneswar Fast Passenger	ECoR	0	0	0	
4586	0	03:10:00	KOTA	Kota Vadodara Passenger	WR	0	0	0	
4587	0	15:20:00	ASN	Asansol Dhanbad Passenger	ER	0	0	0	
4588	0	20:45:00	VG	Viramgam Valsad Passenger	WR	0	1	0	V
4589	0	08:10:00	CWA	Panchvalley Passenger Slip1	SCR	0	0	1	Chhin

4397 rows × 19 columns

until total time and no NaN values

In [ ]: