```
[1] LOAD DATASET raw_data AS csv FROM dataset_77.csv @ artifact file 82
```

Console | Timing | Datasets | Charts

raw_data (**2823** rows)

Views

	ORDERNUMBER (short)	QUANTITYORDERED (short)	PRICEEACH (float)	ORDERLINENUMBER (short)	SALES (float)	ORDERDATE (string)	STATUS (string)	QTR_ID
0	10107	30	95.69999694824219	2	2871	2/24/2003 0:00	Shipped	1
1	10121	34	81.3499984741211	5	2765.89990234375	5/7/2003 0:00	Shipped	2
2	10134	41	94.73999786376953	2	3884.340087890625	7/1/2003 0:00	Shipped	3
3	10145	45	83.26000213623047	6	3746.699951171875	8/25/2003 0:00	Shipped	3
4	10159	49	100	14	5205.27001953125	10/10/2003 0:00	Shipped	4
5	10168	36	96.66000366210938	1	3479.760009765625	10/28/2003 0:00	Shipped	4
6	10180	29	86.12999725341797	9	2497.77001953125	11/11/2003 0:00	Shipped	4
7	10188	48	100	1	5512.31982421875	11/18/2003 0:00	Shipped	4
8	10201	22	98.56999969482422	2	2168.5400390625	12/1/2003 0:00	Shipped	4
9	10211	41	100	14	4708.43994140625	1/15/2004 0:00	Shipped	1
10	10223	37	100	1	3965.659912109375	2/20/2004 0:00	Shipped	1
11	10237	23	100	7	2333.1201171875	4/5/2004 0:00	Shipped	2
12	10251	28	100	2	3188.639892578125	5/18/2004 0:00	Shipped	2
13	10263	34	100	2	3676.760009765625	6/28/2004 0:00	Shipped	2
14	10275	45	92.83000183105469	1	4177.35009765625	7/23/2004 0:00	Shipped	3
15	10285	36	100	6	4099.68017578125	8/27/2004 0:00	Shipped	3
16	10299	23	100	9	2597.389892578125	9/30/2004 0:00	Shipped	3
17	10309	41	100	5	4394.3798828125	10/15/2004 0:00	Shipped	4
18	10318	46	94.73999786376953	1	4358.0400390625	11/2/2004 0:00	Shipped	4
19	10329	42	100	1	4396.14013671875	11/15/2004 0:00	Shipped	4

- The raw sales dataset consist of Order Details places for given products.
- The dataset consist of **25 Order Attributes** and **2823 Row Orders**
- The datatypes in the dataset includes INT, FLOAT, STRING, STRING, DATETIME

Column Name	Data Type	Short Description	Notes
ORDERNUMBER	int	Order Identifier	
QUANTITYORDERED	int	# of Units ordered	
PRICEEACH	float	Unit Price of Each	This can be pushed to Product Table, and references here
ORDERLINENUMBER	int	?	Can be dropped
SALES	float	Sale Price of the order	
ORDERDATE	datetime	Date on which order was placed	Remove time component
STATUS	string	Status of the order	
QTR_ID	int	Quarter Number	Can be dropped since we can derive it from Order Date
MONTH_ID	int	Month Number	Can be dropped since we can derive it from Order Date
YEAR_ID	int	Year Number	Can be dropped since we can derive it from Order Date
PRODUCTCODE	string	Product Identifier	Create Product Table using this ID
CUSTOMERNAME	string	Name of the Company	Create Customer Table
PHONE	string	Customer Phone Number	Push to Customer Table
ADDRESSLINE1	string	Customer Physical Address	Push to Customer Table
ADDRESSLINE2	string	Customer Physical Address Extended	Push to Customer Table
CITY	string	Name of the City	Standardize
STATE	string	Name of the State	Standardize, Default for Country with no State
POSTALCODE	string	ZipCode	Standardize
COUNTRY	string	Name of the country	
TERRITORY	string	Name of the territory	
CONTACTLASTNAME	string	Representative Last name of Customer	Push to Customer Table
CONTACTFIRSTNAME	string	Representative First name of Customer	Push to Customer Table
DEALSIZE	string	size of order	Can be populated using automatic function

[4]

```sql
-- Get Account of Dateoutliers & Date Range
SELECT
ORDERDATE,
COUNT(1)
FROM raw_data
GROUP BY ORDERDATE
ORDER BY ORDERDATE DESC;
```

Console | Timing | Datasets | Charts

temporary_dataset (**252** rows)

Views

	ORDERDATE (string)	count(1) (long)
0	9/9/2004 0:00	9
1	9/8/2004 0:00	26
2	9/7/2004 0:00	2
3	9/5/2003 0:00	11
4	9/30/2004 0:00	11
5	9/3/2004 0:00	4
6	9/3/2003 0:00	2
7	9/28/2003 0:00	13
8	9/27/2004 0:00	2

[6]

```python
# import dataset as pandas dataframe
ds = vizierdb.get_data_frame("raw_data")

print("Null Values by Column")
print(ds.isna().sum())
```

Console | Timing | Datasets | Charts

```
Null Values by Column
ORDERNUMBER          0
QUANTITYORDERED      0
PRICEEACH            0
ORDERLINENUMBER      0
SALES                0
ORDERDATE            0
STATUS               0
QTR_ID               0
MONTH_ID             0
YEAR_ID              0
PRODUCTLINE          0
MSRP                 0
PRODUCTCODE          0
CUSTOMERNAME         0
PHONE                0
ADDRESSLINE1         0
ADDRESSLINE2      2521
CITY                 0
STATE             1486
POSTALCODE         541
COUNTRY              0
TERRITORY            0
```

[7]
```
LOAD DATASET phone_number AS csv FROM dataset_81.csv @ artifact file 87
```

Console | Timing | Datasets | Charts

phone_number (**2823** rows)

Views

	PHONE (string)	COUNTRY (string)
0	2125557818	USA
1	26.47.1555	France
2	+33 1 46 62 7555	France
3	6265557265	USA
4	6505551386	USA
5	6505556809	USA
6	20.16.1555	France
7	+47 2267 3215	Norway
8	6505555787	USA
9	(1) 47.55.6555	France
10	03 9520 4555	Australia
11	2125551500	USA
12	2015559350	USA
13	2035552570	USA
14	40.67.8555	France
15	6175558555	USA
16	90-224 8555	Finland
17	07-98 9555	Norway
18	2155551555	USA
19	2125557818	USA

[11]
```
import pycountry

# Standardize Country Name
def clean_country_name(country_name, alpha=2):
    cleaned_output = pycountry.countries.search_fuzzy(country_name)[0].alpha_2
    return cleaned_output

ds = vizierdb.get_data_frame('phone_number')

ds['COUNTRY_A2'] = ds['COUNTRY'].apply(lambda x: clean_country_name(x))
print(ds.head(5))
```

Console | Timing | Datasets | Charts

```
  PHONE COUNTRY COUNTRY_A2
0        2125557818     USA         US
1        26.47.1555  France         FR
2  +33 1 46 62 7555  France         FR
3        6265557265     USA         US
4        6505551386     USA         US
```

[12]

```
import phonenumbers
import pycountry
import re


def clean_phone_numbers(phone_number, country):
    '''Reformat's passed argument with corrected phone number format based on the country.'''
    formated_phone = phonenumbers.parse(phone_number, country)
    formated_phone = phonenumbers.format_number(formated_phone, phonenumbers.PhoneNumberFormat.INTERNATIONAL)

    return formated_phone

#phonenumbers.format_number(phonenumbers.parse("8006397663", 'US'), phonenumbers.PhoneNumberFormat.NATIONAL)
# get dataframe
ds = vizierdb.get_dataset('phone_number')
#print(phonenumbers.parse("8006397663", 'US'))
print("|PREV FORMAT| NEW FORMAT| COUNTRY")
for row in ds.rows[:10]:
    phone = re.sub("[^0-9]","",row[0])
    country = pycountry.countries.search_fuzzy(row[1])[0].alpha_2
    print(phone,clean_phone_numbers(phone, country), country)
```

Console ▾ Timing Datasets ▾ Charts ▾

```
|PREV FORMAT| NEW FORMAT| COUNTRY
2125557818 +1 212-555-7818 US
26471555 +33 26471555 FR
33146627555 +33 1 46 62 75 55 FR
6265557265 +1 626-555-7265 US
6505551386 +1 650-555-1386 US
6505556809 +1 650-555-6809 US
20161555 +33 20161555 FR
4722673215 +47 22 67 32 15 NO
6505555787 +1 650-555-5787 US
147556555 +33 1 47 55 65 55 FR
```

```
SELECT PRODUCTCODE,
LAST(PRODUCTLINE) AS PRODUCTLINE,
MAX(PRICEEACH) AS MAX_UNIT_PRICE,
MIN(PRICEEACH) AS MIN_UNIT_PRICE,
MAX(MSRP) AS MAX_RETAIL_PRICE
-- MIN(MSRP) AS MIN_RETAIL_PRICE MSRP IS SAME
FROM raw_data
GROUP BY PRODUCTCODE
```

Console ▾ Timing Datasets ▾ Charts ▾

```
SELECT DISTINCT
CUSTOMERNAME,
CONTACTFIRSTNAME,
CONTACTLASTNAME,
PHONE,
ADDRESSLINE1,
ADDRESSLINE2,
CITY,
STATE,
POSTALCODE,
COUNTRY
FROM raw_data
ORDER BY CUSTOMERNAME
```

Console ▾ Timing Datasets ▾ Charts ▾