

Combining data of some columns into one column

CLEANING DATA IN SQL SERVER DATABASES

SQL

Miriam Antona
Software Engineer

Dataset - paper shop sales

vendors

vendor_id	vendor_name	vendor_surname
-----	-----	-----
1	Eric	Mendoza
2	Wu	Fengmian
3	Jaime	Furtado
4	Carol	NULL
...

Dataset - paper shop sales

clients

client_id	client_name	client_surname	city	state
-----	-----	-----	-----	-----
1	Miriam	Antona	Las Vegas	Nevada
2	Astrid	Harper	Chicago	Illinois
3	David	Madden	Phoenix	Arizona
4	Hiroki	Konoe	Orlando	NULL
...

Dataset - paper shop sales

paper_shop_daily_sales

product_name	units	year_of_sale	month_of_sale	day_of_sale	vendor_id	client_id
notebooks	2	2019	1	1	1	1
notebooks	3	2019	5	12	1	2
notebooks	1	2019	8	31	1	3
pencils	2	2019	5	2	2	1
pencils	5	2019	6	7	2	2
pencils	1	2019	9	11	3	3
crayons	1	2019	4	15	1	1
...

Dataset - paper shop sales

paper_shop_monthly_sales

product_name	units	year_of_sale	month_of_sale
-----	-----	-----	-----
notebooks-150	2018	1	1
notebooks-200	2019	1	2
notebooks-30	2019	2	3
pencils-100	2018	1	1
pencils-50	2018	2	2
pencils-130	2019	1	3
...

Combining name and surname

vendor_id	vendor_name	vendor_surname
1	Eric	Mendoza
2	Wu	Fengmian
3	Jaime	Furtado
4	Carol	NULL
...

vendor_id	full_name
1	Eric Mendoza
2	Wu Fengmian
3	Jaime Furtado
4	Carol
...	...

Combining name and surname

CONCAT

```
SELECT vendor_name,  
       vendor_surname,  
       CONCAT(vendor_name, ' ', vendor_surname) AS full_name  
FROM vendors
```

vendor_name	vendor_surname	full_name
Eric	Mendoza	Eric Mendoza
Wu	Fengmian	Wu Fengmian
Jaime	Furtado	Jaime Furtado
Carol	NULL	Carol
...

- `CONCAT` ignores the `NULL` value

Combining name and surname

+ operator

```
SELECT vendor_name,  
       vendor_surname,  
       vendor_name + ' ' + vendor_surname AS full_name  
FROM vendors
```

vendor_name	vendor_surname	full_name
Eric	Mendoza	Eric Mendoza
Wu	Fengmian	Wu Fengmian
Jaime	Furtado	Jaime Furtado
Carol	NULL	NULL
...

Combining name and surname

```
SELECT vendor_name,  
       vendor_surname,  
       vendor_name + ISNULL(' ' + vendor_surname, '') AS full_name  
FROM vendors
```

vendor_name	vendor_surname	full_name
Eric	Mendoza	Eric Mendoza
Wu	Fengmian	Wu Fengmian
Jaime	Furtado	Jaime Furtado
Carol	NULL	Carol
...

Combining dates

paper_shop_daily_sales

product_name	units	year_of_sale	month_of_sale	day_of_sale	...
-----	-----	-----	-----	-----	-----
notebooks	2	2019	1	1	...
notebooks	3	2019	5	12	...
notebooks	1	2019	8	31	...
pencils	2	2019	5	2	...
pencils	5	2019	6	7	...
crayons	2	2019	10	NULL	...
...

Combining dates

`DATEFROMPARTS` -> since SQL Server 2012

```
SELECT
    product_name,
    units,
    DATEFROMPARTS(
        year_of_sale,
        month_of_sale,
        day_of_sale) AS complete_date
FROM paper_shop_daily_sales
```

product_name	units	complete_date
-----	-----	-----
notebooks	2	2019-01-01
notebooks	3	2019-05-12
notebooks	1	2019-08-31
pencils	2	2019-05-02
pencils	5	2019-06-07
crayons	2	NULL
...

Let's practice!

CLEANING DATA IN SQL SERVER DATABASES

Splitting data of one column into more columns

CLEANING DATA IN SQL SERVER DATABASES

SQL

Miriam Antona
Software Engineer

Splitting products and units

```
paper_shop_monthly_sales
```

```
| product_name_units | year_of_sale |
|-----|-----|
| notebooks-150      | 2018         |
| notebooks-200      | 2019         |
| notebooks-30       | 2019         |
| pencils-100        | 2018         |
| pencils-50         | 2018         |
| pencils-130        | 2019         |
| crayons-80         | 2018         |
| ...               | ...         |
```

Using SUBSTRING and CHARINDEX

```
| product_name_units |  
|-----|  
| notebooks-150      |
```

```
| product_name | units |  
|-----|-----|  
| notebooks   | 150   |
```

SUBSTRING(string, **start**, **length**)

CHARINDEX(substring, string [, **start**])

Using SUBSTRING and CHARINDEX

```
SELECT SUBSTRING('notebooks - 150', 1, CHARINDEX('-', 'notebooks - 150') - 1) AS product_name
```


Using SUBSTRING and CHARINDEX

```
SELECT SUBSTRING('notebooks - 150' 1, CHARINDEX('-', 'notebooks - 150') - 1) AS product_name
```

Using SUBSTRING and CHARINDEX

```
SELECT SUBSTRING('notebooks - 150', 1, CHARINDEX('-', 'notebooks - 150') - 1) AS product_name
```

Using SUBSTRING and CHARINDEX

```
SELECT SUBSTRING('notebooks - 150', 1, CHARINDEX('-', 'notebooks - 150') - 1) AS product_name
```

Using SUBSTRING and CHARINDEX

```
SELECT SUBSTRING('notebooks - 150', 1, CHARINDEX('-', 'notebooks - 150') - 1) AS product_name
```

```
| product_name |  
|-----|  
| notebooks   |
```

Using SUBSTRING and CHARINDEX

```
SELECT CAST(  
    SUBSTRING('notebooks-150', CHARINDEX('-', 'notebooks-150') + 1, LEN('notebooks-150'))  
    AS INT) units
```

Using SUBSTRING and CHARINDEX

```
SELECT CAST(  
    SUBSTRING('notebooks-150', 11CHARINDEX('-', 'notebooks-150') + 1, LEN('notebooks-150'))  
    AS INT) units
```

Using SUBSTRING and CHARINDEX

```
SELECT CAST(  
    SUBSTRING('notebooks-150', CHARINDEX('-', 'notebooks-150') + 1, 13LEN('notebooks-150'))  
    AS INT) units
```

```
| units |  
|-----|  
| 150   |
```

Using SUBSTRING and CHARINDEX

```
SELECT
    SUBSTRING('notebooks-150', 1, CHARINDEX('-', 'notebooks-150') - 1) product_name,
    CAST
        (SUBSTRING('notebooks-150', CHARINDEX('-', 'notebooks-150') + 1, LEN('notebooks-150')))
    AS INT) units
```

```
| product_name | units |
|-----|-----|
| notebooks   | 150   |
```


Using LEFT, RIGHT and REVERSE

```
LEFT(string, number_of_chars)
```

- Gets a number of characters from the left of a given string

```
RIGHT(string, number_of_chars)
```

- Gets a number of characters from the right of a given string

```
REVERSE(string_expression)
```

- Reverses a string

Using LEFT, RIGHT and REVERSE

```
SELECT  
    LEFT('notebooks-150', CHARINDEX('-', 'notebooks-150') - 1) AS product_name,  
    RIGHT('notebooks-150', CHARINDEX('-', REVERSE('notebooks-150')) - 1) AS units
```

Using LEFT, RIGHT and REVERSE

```
SELECT
```

9

```
LEFT('notebooks-150', CHARINDEX('-', 'notebooks-150') - 1) AS product_name,  
RIGHT('notebooks-150', CHARINDEX('-', REVERSE('notebooks-150')) - 1) AS units
```

Using LEFT, RIGHT and REVERSE

```
SELECT
  LEFT('notebooks-150', CHARINDEX('-', 'notebooks-150') - 1) AS product_name,
  RIGHT('notebooks-150', CHARINDEX('-', REVERSE('notebooks-150')) - 1) AS units
```

Using LEFT, RIGHT and REVERSE

```
SELECT  
  LEFT('notebooks-150', CHARINDEX('-', 'notebooks-150') - 1) AS product_name,  
  RIGHT('notebooks-150', CHARINDEX('-', REVERSE('notebooks-150')) - 1) AS units
```

4

Using LEFT, RIGHT and REVERSE

```
SELECT
  LEFT('notebooks-150', CHARINDEX('-', 'notebooks-150') - 1) AS product_name,
  RIGHT('notebooks-150', CHARINDEX('-', REVERSE('notebooks-150')) - 1) AS units
```

product_name	units
-----	-----
notebooks	150

Let's practice!

CLEANING DATA IN SQL SERVER DATABASES

Transforming rows into columns and vice versa

CLEANING DATA IN SQL SERVER DATABASES

SQL

Miriam Antona
Software Engineer

Pivot tables in spreadsheets

- Really common
- Allow to group data based of a specific set of columns
- Compute statistics of other columns

Using PIVOT

PIVOT : turns the unique values from one column into multiple columns.

Using PIVOT - Turn product names into columns

```
SELECT * FROM paper_shop_monthly_sales
```

product_name_units	year_of_sale	month_of_sale
-----	-----	-----
notebooks-150	2018	1
notebooks-200	2019	1
notebooks-30	2019	2
pencils-100	2018	1
pencils-50	2018	2
pencils-130	2019	1
crayons-80	2018	1
...

Using PIVOT - Turn product names into columns

Change

```
| product_name_units | year_of_sale | month_of_sale |
|-----|-----|-----|
| notebooks-150      | 2018         | 1              |
| notebooks-200      | 2019         | 1              |
| pencils-50         | 2018         | 2              |
| crayons-80         | 2018         | 1              |
| ...                | ...          | ...            |
```

to

```
| year_of_sale | notebooks | pencils | crayons |
|-----|-----|-----|-----|
| 2018         | 150       | 150     | 80      |
| 2019         | 230       | 130     | 170     |
```

Using PIVOT - Turn product names into columns

```
SELECT
    year_of_sale,
    notebooks,
    pencils,
    crayons
FROM
    (SELECT
        year_of_sale,
        SUBSTRING(product_name_units, 1, charindex('-', product_name_units)-1) AS product_name,
        CAST(SUBSTRING(product_name_units,
            charindex('-', product_name_units)+1, len(product_name_units)) AS INT) AS units
        FROM paper_shop_monthly_sales) AS sales
PIVOT (SUM(units)
FOR product_name IN (notebooks, pencils, crayons))
AS paper_shop_pivot
```

Using PIVOT - Turn product names into columns

```
SELECT
```

```
year_of_sale,  
notebooks,  
pencils,  
crayons
```

```
FROM
```

```
(SELECT
```

```
year_of_sale,  
SUBSTRING(product_name_units, 1, charindex('-', product_name_units)-1) AS product_name,  
CAST(SUBSTRING(product_name_units,  
    charindex('-', product_name_units)+1, len(product_name_units)) AS INT) AS units  
FROM paper_shop_monthly_sales) AS sales
```

```
PIVOT (SUM(units)
```

```
FOR product_name IN (notebooks, pencils, crayons))
```

```
AS paper_shop_pivot
```

Using PIVOT - Turn product names into columns

```
SELECT
    year_of_sale,
    notebooks,
    pencils,
    crayons
FROM
    (SELECT
        year_of_sale,
        SUBSTRING(product_name_units, 1, charindex('-', product_name_units)-1) AS product_name,
        CAST(SUBSTRING(product_name_units,
            charindex('-', product_name_units)+1, len(product_name_units)) AS INT) AS units
        FROM paper_shop_monthly_sales) AS sales
PIVOT (SUM(units)
FOR product_name IN (notebooks, pencils, crayons))
AS paper_shop_pivot
```

Using PIVOT - Turn product names into columns

year_of_sale	product_name	units
-----	-----	-----
2018	notebooks	150
2019	notebooks	200
2019	notebooks	30
2018	pencils	100
2018	pencils	50
2019	pencils	130
2018	crayons	80
2019	crayons	90
2019	crayons	80

Using PIVOT - Turn product names into columns

```
SELECT
    year_of_sale,
    notebooks,
    pencils,
    crayons
FROM
    (SELECT
        year_of_sale,
        SUBSTRING(product_name_units, 1, charindex('-', product_name_units)-1) AS product_name,
        CAST(SUBSTRING(product_name_units,
            charindex('-', product_name_units)+1, len(product_name_units)) AS INT) AS units
        FROM paper_shop_monthly_sales) AS sales
PIVOT (SUM(units)
FOR product_name IN (notebooks, pencils, crayons))
AS paper_shop_pivot
```

Using PIVOT - Turn product names into columns

```
SELECT
    year_of_sale,
    notebooks,
    pencils,
    crayons
FROM
    (SELECT
        year_of_sale,
        SUBSTRING(product_name_units, 1, charindex('-', product_name_units)-1) AS product_name,
        CAST(SUBSTRING(product_name_units,
            charindex('-', product_name_units)+1, len(product_name_units)) AS INT) AS units
        FROM paper_shop_monthly_sales) AS sales
PIVOT (SUM(units)
FOR product_name IN (notebooks, pencils, crayons))
AS paper_shop_pivot
```

Using PIVOT - Turn product names into columns

```
SELECT
    year_of_sale,
    notebooks,
    pencils,
    crayons
FROM
    (SELECT
        year_of_sale,
        SUBSTRING(product_name_units, 1, charindex('-', product_name_units)-1) AS product_name,
        CAST(SUBSTRING(product_name_units,
            charindex('-', product_name_units)+1, len(product_name_units)) AS INT) AS units
        FROM paper_shop_monthly_sales) AS sales
PIVOT (SUM(units)
FOR product_name IN (notebooks, pencils, crayons))
AS paper_shop_pivot
```

Using PIVOT - Turn product names into columns

```
SELECT
    year_of_sale,
    notebooks,
    pencils,
    crayons
FROM
    (SELECT
        year_of_sale,
        SUBSTRING(product_name_units, 1, charindex('-', product_name_units)-1) AS product_name,
        CAST(SUBSTRING(product_name_units,
            charindex('-', product_name_units)+1, len(product_name_units)) AS INT) AS units
        FROM paper_shop_monthly_sales) AS sales
PIVOT (SUM(units)
FOR product_name IN (notebooks, pencils, crayons))
AS paper_shop_pivot
```

Using PIVOT - Turn product names into columns

year_of_sale	notebooks_units	pencils_units	crayons_units
2018	150	150	80
2019	230	130	170

Using UNPIVOT

UNPIVOT : Turns columns into rows.

```
SELECT * FROM pivot_sales
```

year_of_sale	notebooks	pencils	crayons
2018	150	150	80
2019	230	130	170

Using UNPIVOT - Turn product names into rows

```
SELECT * FROM pivot_sales
UNPIVOT
    (units FOR product_name IN (notebooks, pencils, crayons)
) AS unpvt
```

year_of_sale	units	product_name
2018	150	notebooks
2018	150	pencils
2018	80	crayons
2019	230	notebooks
2019	130	pencils
2019	170	crayons

Using UNPIVOT - Turn product names into rows

```
SELECT * FROM pivot_sales
UNPIVOT
  (units FOR product_name IN (notebooks, pencils, crayons)
) AS unpvt
```

year_of_sale	units	product_name
2018	150	notebooks
2018	150	pencils
2018	80	crayons
2019	230	notebooks
2019	130	pencils
2019	170	crayons

Using UNPIVOT - Turn product names into rows

```
SELECT * FROM pivot_sales
UNPIVOT
  (units FOR product_name IN (notebooks, pencils, crayons)
) AS unpvt
```

year_of_sale	units	product_name
2018	150	notebooks
2018	150	pencils
2018	80	crayons
2019	230	notebooks
2019	130	pencils
2019	170	crayons

Using UNPIVOT - Turn product names into rows

```
SELECT * FROM pivot_sales
UNPIVOT
  (units FOR product_name IN (notebooks, pencils, crayons)
) AS unpvt
```

year_of_sale	units	product_name
2018	150	notebooks
2018	150	pencils
2018	80	crayons
2019	230	notebooks
2019	130	pencils
2019	170	crayons

Let's practice!

CLEANING DATA IN SQL SERVER DATABASES

Congratulations!

CLEANING DATA IN SQL SERVER DATABASES



Miriam Antona
Software Engineer

Chapter 1

- Why cleaning data is important
- Removing blank spaces at the beginning and end of a string
- Filling numbers with leading zeros
- Unifying strings
- Similarity between strings

Chapter 2

- Deal with missing data
- Avoid duplicate data
- Work with different formats of dates

Chapter 3

- Deal with out of range values and inaccurate data
- Converting data with different types
- Matching patterns

Chapter 4

- Combine data of some columns into one
- Split data of one column into more columns
- Transform rows into columns and vice versa

Thank you!

CLEANING DATA IN SQL SERVER DATABASES