



Cleaning Data in SQL Server

Cleaning data fetched from various datasets.



Unifying flight name formats using Replicate()

- You need to get every register with more than 100 delays from the flight_statistics table. In a unique column, you have to concatenate the carrier_code, registration_code, and airport_code, having a similar format to this one: "AA - 000000119, JFK".

query.sql

```
1 SELECT
2 Concat(carrier_code,
3 ' - ',
4 Replicate('0', 9-len(registration_code)),
5 registration_code,
6 ', ',
7 airport_code) AS registration_code
8
9 FROM flight_statistics
10 WHERE delayed>100;
```

query result

flight_statistics

registration_code

AA - 000000119, JFK

B6 - 000000120, JFK

AA - 000000127, JFK

Trim() strings

- Examine the content of the airports table, and use the appropriate function to remove the leading and trailing spaces.

```
1 SELECT
2 airport_code,
3 TRIM(airport_name) AS airport_name,
4 TRIM(airport_city) AS airport_city,
5 TRIM(airport_state) AS airport_state
6 FROM airports;
```



Run Code

Submit Answer

query result

airports

airport_code	airport_name	airport_city	airport_state
MSP	Minneapolis-St Paul International	Minneapolis	Minnesota
JFK	John F. Kennedy International	New York City	New York
LAX	Los Angeles International	Los Angeles	California
DFW	Dallas/Fort Worth International	Dallas/Fort Worth	Texas

Unifying name formats with Case(),Replace()

- There are inconsistent values for 'Chicago' in the airport_city column, with values such as 'ch'. You will treat these inconsistent values by replacing them.*

```
1 SELECT
2   airport_code,
3   airport_name,
4   CASE
5     WHEN airport_city <> 'Chicago'
6     THEN REPLACE(airport_city,'ch','Chicago')
7     ELSE airport_city
8   END as airport_city,
9   airport_state
10  FROM airports
11  WHERE airport_code in ('ORD','MDW');
```



Run Code

Submit Answer

query result

airports



airport_code

airport_name

airport_city

airport_state

ORD

Chicago O'Hare International

Chicago

Illinois

MDW

Chicago Midway International

Chicago

Illinois

Comparing names with SOUNDEX()

- Some `statistician_name` and `statistician_surname` are written in a different way, such as Miriam Smith and Myriam Smyth. You think about comparing with `SOUNDEX()` the names of the statisticians. If the result of `SOUNDEX()` is the same, but the texts you are comparing are different, you will find the data you need to clean.

```
13 SELECT DISTINCT
14 S1.statistician_name, S1.statistician_surname
15 FROM flight_statistics S1
16     INNER JOIN flight_statistics S2
17         ON SOUNDEX(S1.statistician_name)=SOUNDEX(S2.statistician_name)
18         AND SOUNDEX(S1.statistician_surname)=SOUNDEX(S2.
19 statistician_surname)
20 WHERE (S1.statistician_name <> S2.statistician_name)
      OR (S1.statistician_surname <> S2.statistician_surname);
```



Run Code

Submit Answer

query result	flight_statistics
Brian	Page
Bryan	Page
Miriam	Smith
Miriam	Smyth

Filling missing values using ISNULL()

- *Replace the missing values for airport_city and airport_state column with the 'Unknown' string.*

```
1 SELECT
2   airport_code,
3   airport_name,
4   ISNULL(airport_city, 'Unknown') AS airport_city,
5   ISNULL(airport_state, 'Unknown') AS airport_state
6 FROM
7   airports
```



Run Code

Submit Answer

query result	airports		
BWI	Baltimore/Washington International Thurgood Marshall	Baltimore	Maryland
EWR	Newark Liberty International	Newark	New Jersey
SEA	Seattle/Tacoma International	Unknown	Unknown
PHL	Philadelphia International	Philadelphia	Unknown
SLC	Salt Lake City International	Unknown	Utah
MCO	Orlando International	Orlando	Florida

Filling missing values using COALESCE()

- Now, you want to create a new column, location, that returns the values of the airport_city column, and in case it has NULL values, return the value of airport_state. Finally, if airport_state is also NULL, you want to return the string 'Unknown'.

```
1 SELECT
2   airport_name,
3   airport_city,
4   airport_state,
5   COALESCE(airport_city,airport_state,'Unknown') AS location
6 FROM airports
```



Run Code

Submit Answer

query result	airports			
Seattle/Tacoma International		null	null	Unknown
Philadelphia International		Philadelphia	null	Philadelphia
Salt Lake City International		null	Utah	Utah
Orlando International		Orlando	Florida	Orlando
Tampa International		Tampa	Fl	Tampa

Treating duplicates with Row_number()

- Get all the rows without duplicates. You consider that the repeating group for this table is formed by the columns `airport_code`, `carrier_code`, and `registration_date`.

```
1 WITH CTE AS
2 (SELECT
3  *,
4  ROW_NUMBER() OVER
5    ( PARTITION BY airport_code, carrier_code, registration_date
6      ORDER BY airport_code, carrier_code, registration_date )
7    AS row_num
8  FROM flight_statistics)
9
10 SELECT * FROM CTE
11 WHERE row_num = 1
```



Run Code

Submit Answer

query result		flight_statistics							
airport_code	carrier_code	canceled	on_time	delayed	diverted	statistician_name	statistician_surname	registration_date	row_num
JFK	AA	74	819	233	13	Miriam	Smith	2014-01-31	1
JFK	AA	59	764	212	14	Brian	Page	2014-02-28	1
JFK	B6	438	1865	1010	29	Myrlam	Smith	2014-01-31	1
JFK	B6	181	2089	831	16	Carol	York	2014-02-28	1

Convert() Dates Format

- The format of the `registration_date` column is `yyyy-mm-dd`, and you want to show the results in the format of `mm/dd/yyyy`, which is hardcoded as `101`, using the `CONVERT()` function. Notice that the type of the `registration_date` column is `VARCHAR(10)` and not a date.

```
1 SELECT
2   airport_code,
3   carrier_code,
4   canceled,
5   Convert(VARCHAR(10),CAST(registration_date AS DATE),101) AS
   registration_date
6 FROM flight_statistics
7 WHERE Convert(VARCHAR(10),CAST(registration_date AS DATE),101)
8   BETWEEN '01/01/2014' AND '06/30/2014';
```



Run Code

Submit

query result

flight_statistics

airport_code	carrier_code	canceled	registration_date
JFK	AA	74	01/31/2014
JFK	B6	438	01/31/2014
JFK	HA	null	01/31/2014

Excluding inaccurate data from a TV Series dataset

- In a TV series dataset there are contradictory values, some rows with a TRUE value in its is_adult column have a number smaller than 18 in its min_age column. Can you find these rows with inaccurate data?*

```
1
2 SELECT
3 id,name,rating,is_adult,min_age,summary
4 FROM series
5 WHERE NOT (is_adult=1 AND min_age<18)
```



Run Code

Submit Answer

query result

episodes

series

id	name	rating	is_adult	min_age	summary
----	------	--------	----------	---------	---------

1	Adventure Time	8.4	false	10	Adventure Time's unlikely heroes Finn and Jake are budding
---	----------------	-----	-------	----	--

3	Futurama	9.2	false	13	Futurama follows pizza guy Philip J. Fry, who reawakens
---	----------	-----	-------	----	---

5	Homeland	8.3	false	17	The winner of 6 Emmy Awards including Outstanding Dr
---	----------	-----	-------	----	--

6	Westworld	8.6	true	18	Westworld is a dark odyssey about the dawn of artificial
---	-----------	-----	------	----	--

7	Silicon Valley	4.4	false	16	In the high-tech mold of modern Silicon Valley, the
---	----------------	-----	-------	----	---

Converting Data Types Using CAST() or CONVERT()

- *Tables could store data with different types than you want. Sometimes you will need to convert these types to the correct ones to perform the operations you want.*
- *The num_ratings stores integer numbers, but this time it was designed as VARCHAR(5).*

```
1  -- Using CAST()
2  SELECT
3  AVG( CAST(num_ratings AS INT) )
4  FROM series
5  WHERE CAST(num_ratings AS INT) BETWEEN 0 AND 5000;
6
7  -- Using CONVERT()
8  SELECT
9  AVG( CONVERT(INT,num_ratings) )
10 FROM series
11 WHERE CONVERT(INT,num_ratings) BETWEEN 0 AND 5000;
```



Run Code

Submit Answer

query result

series

3009

The TV series with most episodes

- *In the episodes table, there is a column named number. It stores the number of each episode within a season for every series. This column was designed as VARCHAR(5), but it actually stores numbers.*
- *Can you guess which is the series with most episodes within a season?*

query.sql

Light Mode

```
1 WITH CTE AS
2 ( SELECT
3     s.name series_name, season,e.name episode_name,
4     CAST(number AS INT) AS number
5 FROM episodes e
6     INNER JOIN series s
7     ON e.series_id = s.id
8 )
9 SELECT series_name,season,episode_name,number
10 FROM CTE
11 WHERE number = [(SELECT MAX(number) FROM CTE )]
```

 Run Code

query result

episodes

series

▼

series_name	season	episode_name	number
Adventure Time	1	The Gut Grinder	26

Find Wrong URLs using Like()

- Prepare a script that checks every `official_site` value from the `series` table to analyze possible wrong URLs (not starting with `www.`).

query.sql

```
1
2 SELECT
3 name,
4 official_site
5 FROM series
6 WHERE official_site NOT LIKE 'www.%';
```

 Run Code

query result	series
name	official_site
Adventure Time	wwq.cartoonnetwork.com/video/adventuretime/
Dexter	ww.sho.com/sho/dexter/home

Checking Phone Numbers using Like()

- *prepare a script that checks every contact_number value from the series table to get those numbers which are not of the format 555-xxx-xxxx.*

```
query.sql
1
2 SELECT
3 name,
4 contact_number
5 FROM series
6 WHERE contact_number NOT LIKE '555-____-____';
```


Run Code

query result	series
name	contact_number
The Good Doctor	000-930-1274

Concatenating cities and states with Concat()

- Concatenate the names of the cities with the states using the `CONCAT()` function, while using a `CASE` statement that returns blank when state is `NULL` and performs a normal concatenation otherwise.

```
1
2 SELECT
3 client_name,client_surname,
4 CONCAT(city,
5     CASE
6     WHEN state IS NULL THEN ''
7     ELSE CONCAT(', ',state)
8     END
9     ) AS city_state
10 FROM clients
```

 [Run Code](#)

query result		clients
client_name	client_surname	city_state
Miriam	Antona	Las Vegas, Nevada
Astrid	Harper	Chicago, Illinois
David	Madden	Phoenix, Arizona

Split Column with Substring() and Charindex()

- You need to split this city_state column into two new columns, one for the city and the other one for the state.*

```
1 SELECT
2 client_name,client_surname,city_state,
3 SUBSTRING(city_state,1,CHARINDEX(', ',city_state)-1) AS city,
4 SUBSTRING(city_state,CHARINDEX(', ',city_state)+1,
5 LEN(city_state)) AS state
6 FROM clients_split
7
```



Run Code

Submit Answer

query result		clients_split		
client_name	client_surname	city_state	city	state
Miriam	Antona	Las Vegas, Nevada	Las Vegas	Nevada
Astrid	Harper	Chicago, Illinois	Chicago	Illinois
David	Madden	Phoenix, Arizona	Phoenix	Arizona
Hiroki	Konoe	Orlando, Florida	Orlando	Florida

Combine Different Parts of a Date: DATEFROMPARTS()

- Combine the columns, `year_of_sale`, `month_of_sale`, and `day_of_sale`, in the `paper_shop_daily_sales` table.

```
1 SELECT
2   product_name,
3   units,
4   DATEFROMPARTS(year_of_sale, month_of_sale, day_of_sale) complete_date
5 FROM paper_shop_daily_sales;
```



Run Code

query result

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

Turning rows into columns: Pivot()

- The structure of the table `paper_shop_monthly_sales` is not appropriate for the report. You want to generate a report with this appearance:

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

```
1 SELECT
2   year_of_sale, notebooks, pencils, crayons
3 FROM
4   (SELECT
5     SUBSTRING(product_name_units,1,charindex('-',product_name_units)-1)
6     AS product_name,
7     CAST(SUBSTRING(product_name_units,charindex('-',product_name_units)+1,
8              len(product_name_units)) AS INT) AS units,
9     year_of_sale
10    FROM paper_shop_monthly_sales) sales
11   PIVOT( SUM(units) FOR product_name IN (notebooks,pencils,crayons) )
12        AS paper_shop_pivot
```

 [Run Code](#) [Submit Answer](#)

query result	paper_shop_monthly_sales		
year_of_sale	notebooks	pencils	crayons
2018	150	150	80
2019	230	130	170

Turning columns into rows: Unpivot()

- Suppose you stored the result from the previous exercise in a new table called `pivot_sales`, and now you want to turn the columns `notebooks`, `pencils`, and `crayons` into row values.

```
1 SELECT
2 *
3 FROM pivot_sales
4 UNPIVOT(units FOR product_name
5         IN (notebooks,pencils,crayons) )
6         AS unpivot_sales;
```



Run Code

query result

pivot_sales

year_of_sale	units	product_name
2018	150	notebooks
2018	150	pencils
2018	80	crayons