

Introduction to data cleaning

CLEANING DATA IN POSTGRESQL DATABASES

SQL

Darryl Reeves, Ph.D.

Industry Assistant Professor, New York
University

Why is cleaning data important?

- Data is messy
- Before it can be analyzed, often needs cleaning
- Helpful to utilize column type constraints
- Course focuses on when defensive approaches not available/applicable

Cleaning string data

- String data: abundant, flexible, and often messy

name	grade	inspection_type	census_tract	...
...
EMPANADAS MONUMENTAL	B	Cycle Inspection / Re-inspection	26900	...
ALPHONSO'S PIZZERIA & TRATTORIA	A	Cycle Inspection / Initial Inspection	202	...
THE SPARROW TAVERN	A	Cycle Inspection / Initial Inspection	12500	...
BURGER KING	A	Cycle Inspection / Re-inspection	86400	...
ASTORIA PIZZA	B	Cycle Inspection / Re-inspection	6300	...
...

Cleaning string data

name	grade	inspection_type	census_tract	...
...
EMPANADAS MONUMENTAL	B	Cycle Inspection / Re-inspection	26900	...
ALPHONSO'S PIZZERIA & TRATTORIA	A	Cycle Inspection / Initial Inspection	202	...
THE SPARROW TAVERN	A	Cycle Inspection / Initial Inspection	12500	...
BURGER KING	A	Cycle Inspection / Re-inspection	86400	...
ASTORIA PIZZA	B	Cycle Inspection / Re-inspection	6300	...
...

1. Restrict capitalization in `name`
2. Remove extra divider space in `inspection_type`
3. Make `census_tract` values have a uniform length

Cleaning string data

name	grade	inspection_type	census_tract	...
...
EMPANADAS MONUMENTAL	B	Cycle Inspection / Re-inspection	26900	...
ALPHONSO'S PIZZERIA & TRATTORIA	A	Cycle Inspection / Initial Inspection	202	...
THE SPARROW TAVERN	A	Cycle Inspection / Initial Inspection	12500	...
BURGER KING	A	Cycle Inspection / Re-inspection	86400	...
ASTORIA PIZZA	B	Cycle Inspection / Re-inspection	6300	...
...

name	grade	inspection_type	census_tract	...
...
Empanadas Monumental	B	Cycle Inspection / Re-inspection	026900	...
Alphonso's Pizzeria & Trattoria	A	Cycle Inspection / Initial Inspection	000202	...
The Sparrow Tavern	A	Cycle Inspection / Initial Inspection	012500	...
Burger King	A	Cycle Inspection / Re-inspection	086400	...
Astoria Pizza	B	Cycle Inspection / Re-inspection	006300	...
...

Using the INITCAP() function

INITCAP(input_string) - fixing capitalization

```
SELECT INITCAP('HELLO FRIEND!');
```

```
Hello Friend!
```

Using the REPLACE() function

`REPLACE(input_string, to_replace, replacement)` - replacing one text value with another

```
SELECT REPLACE('180 Main Street', 'Street', 'St');
```

```
180 Main St
```

Using the LPAD() function

`LPAD(input_string, length [, fill_value])` - prepending text values to a string

```
SELECT LPAD('123', 7, 'X');
```

```
XXXX123
```


Building the string cleaning query

```
SELECT
  INITCAP(name) as name,
  grade,
  REPLACE(inspection_type, ' / ', ' / ') as inspection_type,
  LPAD(census_tract, 6, '0') as census_tract
FROM
  restaurant_inspection;
```

name	grade	inspection_type	census_tract	...
...
Empanadas Monumental	B	Cycle Inspection / Re-inspection	026900	...
Alphonso'S Pizzeria & Trattoria	A	Cycle Inspection / Initial Inspection	000202	...
The Sparrow Tavern	A	Cycle Inspection / Initial Inspection	012500	...
Burger King	A	Cycle Inspection / Re-inspection	086400	...
Astoria Pizza	B	Cycle Inspection / Re-inspection	006300	...
...

Let's practice!

CLEANING DATA IN POSTGRESQL DATABASES

Pattern matching

CLEANING DATA IN POSTGRES SQL DATABASES



Darryl Reeves, Ph.D.

Industry Assistant Professor, New York
University

Identifying patterns: an example

camis	name	inspection_date	score	nta	...
...
41659848	LA BRISA DEL CIBAO	01/30/2018	20	QN26	...
40961447	MESON SEVILLA RESTAURANT	03/19/2019	50	MN15	...
50063071	WA BAR	05/23/2018	15	MN17	...
50034992	EMPANADAS MONUMENTAL	06/21/2019	17	MN35	...
50095871	ALPHONSO'S PIZZERIA & TRATTORIA	01/16/2020	10	MN28	...
41104041	THE SPARROW TAVERN	09/17/2019	13	QN72	...
50016937	BURGER KING	09/14/2018	12	QN55	...
50066469	DARBAR'S CHICKEN & RIBS	08/07/2017	11	QN55	...
41195691	F & J PINE RESTAURANT	05/02/2019	26	BX49	...
50015706	EL RINCONCITO DE LOS SABORES	12/18/2019	20	QN35	...
...

¹ <https://www1.nyc.gov/site/planning/data-maps/open-data/dwn-nynta.page>

Identifying patterns: an example

Valid Prefixes to NTA Code

- MN - Manhattan
- BK - Brooklyn
- BX - Bronx
- QN - Queens
- SI - Staten Island

Identifying patterns: an example

camis	name	inspection_date	score	nta	...
...
50058910	HUNGER PANG	06/15/2017	13	BK42	...
40376029	MOMS LUNCHEONETTE	03/14/2020	13	QN544	...
50019128	IKI MODERN JAPANESE CUISINE	07/13/2017	10	QN22	...
50000458	BEVERLEY PIZZA & CAFE	07/08/2019	12	BK41	...
50002521	JADE PALACE	05/14/2018	11	BX13	...
...

The LIKE operator

```
SELECT * FROM restaurant_inspection WHERE nta LIKE 'QN544';
```

```
SELECT * FROM restaurant_inspection WHERE nta = 'QN544';
```

The LIKE Operator

Pattern Matching Characters

- `%` - matches any sequence of zero or more characters
- `_` (underscore) - matches a single character

```
SELECT
```

```
*
```

```
FROM
```

```
restaurant_inspection
```

```
WHERE nta LIKE 'QN%';
```

```
SELECT
```

```
*
```

```
FROM
```

```
restaurant_inspection
```

```
WHERE nta LIKE 'QN%'
```

```
AND nta NOT LIKE 'QN__';
```


Regular expressions (REs)

- Pattern matching with `LIKE` is limited
- More specific patterns can be useful
- Regular Expressions (REs) enable more expressive pattern matching

The SIMILAR TO operator

- Provides additional pattern matching functionality

```
SELECT
```

```
  camis, name, inspection_date, score, nta
```

```
FROM
```

```
  restaurant_inspection
```

```
WHERE nta SIMILAR TO 'QN%' AND nta NOT SIMILAR TO 'QN__';
```

Basics of REs

Metacharacter	Usage	Example RE	Example Match
<code>\d</code>	matches a digit (0-9)	<code>\d\d\d</code>	'345'
<code>?</code>	matches 0 or 1 of previous character	<code>x\d?</code>	'x5'
<code>+</code>	matches one or more of previous character	<code>\d+</code>	'10'
<code>*</code>	matches any character 0 or more times	<code>\d*</code>	'3081'
<code>[]</code>	matches any character inside of the brackets	<code>[a-z]</code>	'f'

Using REs with SIMILAR TO

```
SELECT
  camis, name, inspection_date, score, nta
FROM
  restaurant_inspection
WHERE nta SIMILAR TO 'QN%' AND nta NOT SIMILAR TO 'QN__';
```

```
SELECT
  camis, name, inspection_date, score, nta
FROM
  restaurant_inspection
WHERE
  nta NOT SIMILAR TO '[A-Z][A-Z]\d\d';
```

camis	name	inspection_date	score	nta
41659848	LA BRISA DEL CIBAO	01/30/2018	20	Q26
41104041	THE SPARROW TAVERN	09/17/2019	13	QN723

Let's practice!

CLEANING DATA IN POSTGRESQL DATABASES

Matching similar strings

CLEANING DATA IN POSTGRESQL DATABASES

SQL

Darryl Reeves, Ph.D.

Industry Assistant Professor, New York University

Similar strings (example)

- Delivery address: 121 Fontainebleau Drive
- Fountainbleau , Fontainbleu , Fontainblue

The Soundex algorithm

- Words represented by sound
- Encodes words using 4 characters
- Fountainbleau , Fontainbleu , Fontainblue → F535

SOUNDEX() in PostgreSQL

- Available through `fuzzystrmatch` module

```
CREATE EXTENSION fuzzystrmatch;
```

`SOUNDEX(input_string)` → 4 character code

```
SELECT
  SOUNDEX('Fountainbleau') as sd1,
  SOUNDEX('Fontainebleau') as sd2,
  SOUNDEX('Fontaineblue')  as sd3;
```

sd1		sd2		sd3
-----+-----+-----				
F535		F535		F535

The DIFFERENCE() function

`DIFFERENCE(string1, string2)` → 0, 1, 2, 3, or 4

```
SELECT SOUNDEX('pair') AS sd_pair, SOUNDEX('pear') AS sd_pear;
```

```
sd_pair | sd_pear  
-----+-----  
P600   | P600
```

```
SELECT DIFFERENCE('pair', 'pear') AS diff;
```

```
diff  
----  
4
```

The DIFFERENCE() function

```
SELECT SOUNDEX('bow') AS sd_bow, SOUNDEX('bough') AS sd_bough;
```

```
sd_bow | sd_bought  
-----+-----  
B300   | B230
```

```
SELECT DIFFERENCE('bout', 'bought') AS diff
```

```
diff  
----  
    2
```

Using DIFFERENCE()

name	boro	building	street
ATOMIC WINGS	Manhattan	2090	FREDERICK DOUGLASS BOULEVARD
BARAKA BUFFET	Manhattan	2546	FREDERICK DOUGLASS BOULEVARD
CHOCOLAT	Manhattan	2217	FREDERICK DOUGLASS BOULEVARD
ESO	Manhattan	2906	FREDERICK DOUGLASS BOULEVARD
HOP HOUSE HARLEM	Manhattan	2224	FREDERICK DOUGLASS BOULEVARD
HOT POT UNDER DE' TREE	Manhattan	2839	FREDERICK DOUGLAS BOULEVARD
LIDO	Manhattan	2168	FREDERICK DOUGLAS BOULEVARD
MESS HALL	Manhattan	2194	FRDRCK DGLS BLVD
VINATERIA	Manhattan	2211	FREDERICK DOUGLAS BOULEVARD

Using DIFFERENCE()

name	boro	building	street	sd_street
ATOMIC WINGS	Manhattan	2090	FREDERICK DOUGLASS BOULEVARD	F636
BARAKA BUFFET	Manhattan	2546	FREDERICK DOUGLASS BOULEVARD	F636
CHOCOLAT	Manhattan	2217	FREDERICK DOUGLASS BOULEVARD	F636
ESO	Manhattan	2906	FREDERICK DOUGLASS BOULEVARD	F636
HOP HOUSE HARLEM	Manhattan	2224	FREDERICK DOUGLASS BOULEVARD	F636
HOT POT UNDER DE' TREE	Manhattan	2839	FREDERICK DOUGLAS BOULEVARD	F636
LIDO	Manhattan	2168	FREDERICK DOUGLAS BOULEVARD	F636
MESS HALL	Manhattan	2194	FRDRCK DGLS BLVD	F636
VINATERIA	Manhattan	2211	FREDERICK DOUGLAS BOULEVARD	F636

```
SELECT
  name, boro, building, street
FROM
  restaurant_inspections
WHERE
  DIFFERENCE(street, 'Frederick Douglass Boulevard') = 4;
```

Updating the recordings

UPDATE

table_name

SET

column_name = value

WHERE

condition

UPDATE

restaurant_inspection

SET

street = 'Frederick Douglass Boulevard'

WHERE

DIFFERENCE(street, 'Frederick Douglass Boulevard') = 4;

UPDATE 10

Let's practice!

CLEANING DATA IN POSTGRESQL DATABASES