

# **How Do We Solve the World's Spreadsheet Problem?**

**Alex Rasmussen**

@alexras



# Hi, I'm Alex!

@alexras  
[alexras.info](http://alexras.info)

freenome  
[freenome.com](http://freenome.com)



**BITS ON DISK**

[bitsondisk.com](http://bitsondisk.com)





# My Background

**2009-2013:** really fast sorting

**2013-2016:** data wrangling

**2017-2018:** cancer-fighting robots

I **think**/**worry** a lot  
about spreadsheets.

Today's focus:  
**spreadsheet data**  
(for compute, [feliennne.com](https://feliennne.com))

# **This talk:**

- 1. Spreadsheets are great**
- 2. Spreadsheets are a problem**
- 3. How we can fix it**

# **This talk:**

- 1. Spreadsheets are great**
2. Spreadsheets are a problem
3. How we can fix it

What's so **great**  
about spreadsheets?



Spreadsheets are  
**Ubiquitous**

**1.2 billion** Office users (~16% of humans)

**1.2 billion** Office users (~16% of humans)

**60 million** Office 365 customers

**1.2 billion** Office users (~16% of humans)

**60 million** Office 365 customers

**>5 million businesses** use Google Apps

Spreadsheets are  
**Approachable**





Spreadsheets are

**Flexible**

# Data grids

# Data grids

# Graphs

**Data grids**

**Graphs**

**Anything tabular**



**Data grids**

**Graphs**

**Anything tabular**

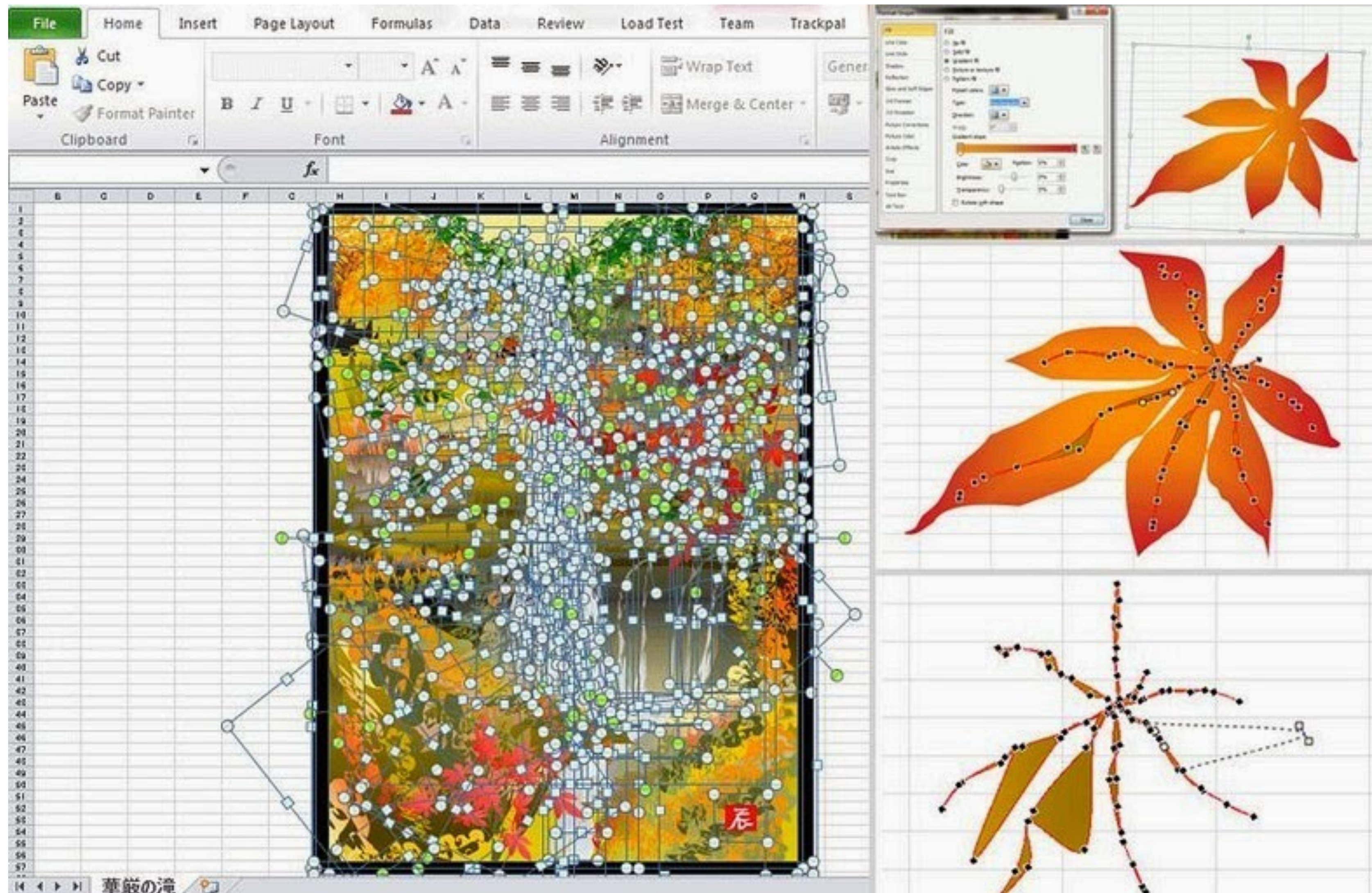
**Full-scale “apps”**





**Tatsuo Horiuchi (b. 1940)**  
*Kegon Falls, 2007*  
**AutoShape on canvas**





<https://pasokonga.com/>



So if spreadsheets are  
**ubiquitous, approachable,**  
and **flexible,**  
what's the problem?

# **This talk:**

1. Spreadsheets are great
- 2. Spreadsheets are a problem**
3. How we can fix it



# Problem #1:

## Data Types



**GSV Arson Kite**

@Phylan



ME: \*makes typo while entering a number\*

EXCEL: WAS THAT A DATE

ME: no I meant t-

EXCEL: THAT WAS A  DATE

ME: it doesn't even make sen-

EXCEL: MAY 12TH 1382. LOOK I EVEN FORMATTED IT.  
IT IS THIS FOREVER

9:00 AM - Feb 1, 2018

♡ 64.8K 💬 17.5K people are talking about this



Automatic type  
conversion can cause  
**serious problems.**

**DEC1**

**DEC1**

**12/1**



**RIKEN Identifier**

**23100009E13**

**RIKEN Identifier**

**23100009E13**

**2.31E+13**

“We confirmed gene name errors in **987** supplementary files from **704** published articles (**19.6% of all articles**).”

[https://genomebiology.biomedcentral.com/  
articles/10.1186/s13059-016-1044-7](https://genomebiology.biomedcentral.com/articles/10.1186/s13059-016-1044-7)

# False Equivalence

$$\begin{aligned} & 000123 \\ &= 00123 \\ &= 123 \end{aligned}$$

True if they're integers,  
but **what if they're strings?**

# Enumerated Types

# Enumerated Types

**“Prostate Cancer”**

# Enumerated Types

**“Prostate Cancer”**

**“prostate cancer”**

# Enumerated Types

**“Prostate Cancer”**

**“prostate cancer”**

**“prostatecancer”**



# Enumerated Types

**“Prostate Cancer”**

**“prostate cancer”**

**“prostatecancer”**

**“PC”**

# Enumerated Types

**“Prostate Cancer”**

**“prostate cancer”**

**“prostatecancer”**

**“PC”**

**“prostate”**

# Enumerated Types

**“Prostate Cancer”**

**“prostate cancer”**

**“prostatecancer”**

**“PC”**

**“prostate”**

**“prostrate”**

# List Validations? Sheet Protection?

- 😊 Easy to add
- 😞 Easy to remove by accident
- 😡 Hard to enforce

Data loss!

False equivalence!

Ontological chaos!

**Mass hysteria!**

**Problem #2:**

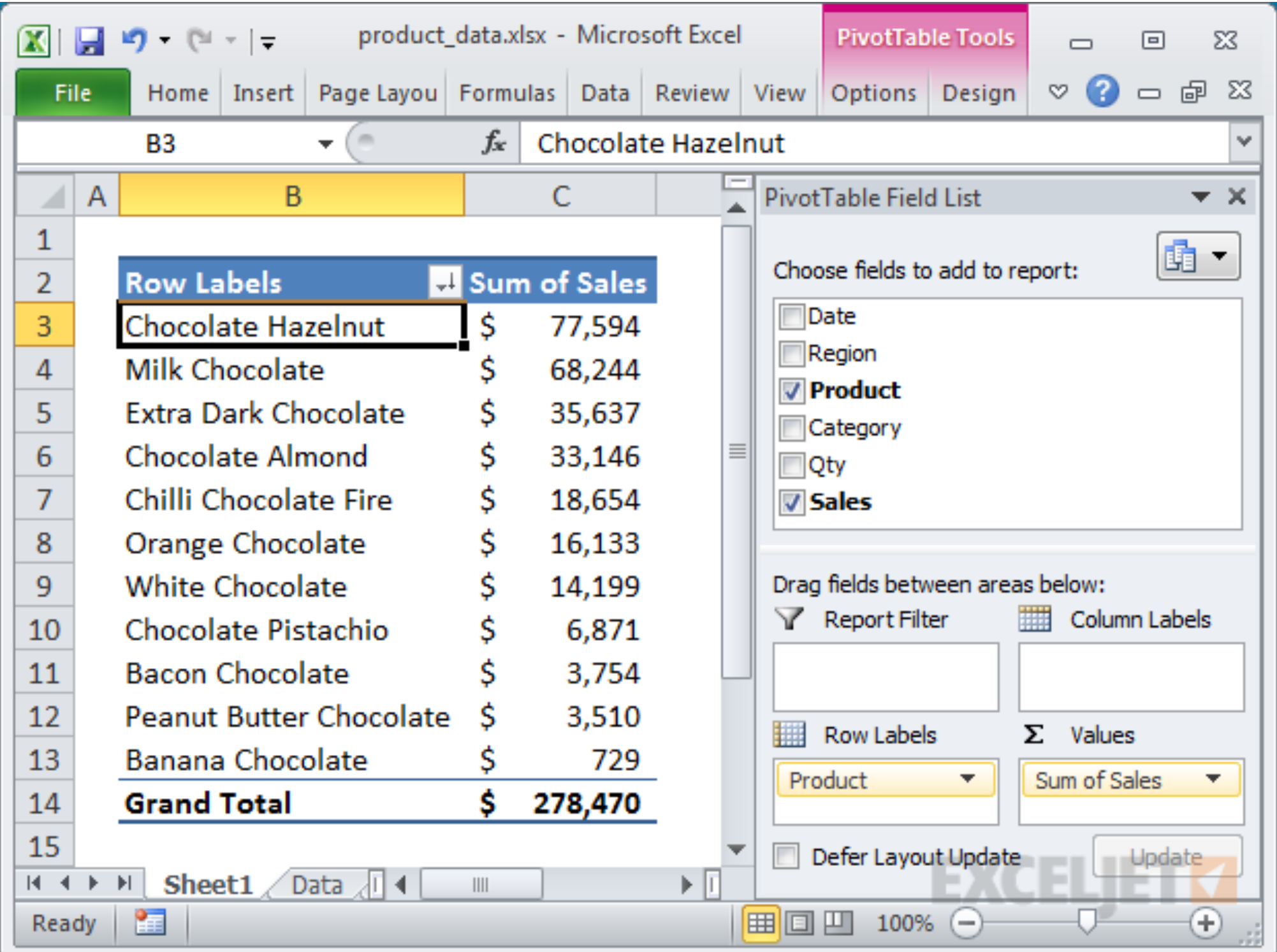
**Queryability**

Inside a  
spreadsheet, things  
are pretty good!

Formulas!

Pivot Tables!

Filters!



The screenshot shows a Microsoft Excel window with the file name "product\_data.xlsx". The "PivotTable Tools" task pane is active, showing the "Options" tab. The PivotTable is located in the worksheet "Sheet1" and is filtered by "Chocolate Hazelnut". The PivotTable has two columns: "Row Labels" and "Sum of Sales". The data is as follows:

Row Labels	Sum of Sales
Chocolate Hazelnut	\$ 77,594
Milk Chocolate	\$ 68,244
Extra Dark Chocolate	\$ 35,637
Chocolate Almond	\$ 33,146
Chilli Chocolate Fire	\$ 18,654
Orange Chocolate	\$ 16,133
White Chocolate	\$ 14,199
Chocolate Pistachio	\$ 6,871
Bacon Chocolate	\$ 3,754
Peanut Butter Chocolate	\$ 3,510
Banana Chocolate	\$ 729
<b>Grand Total</b>	<b>\$ 278,470</b>

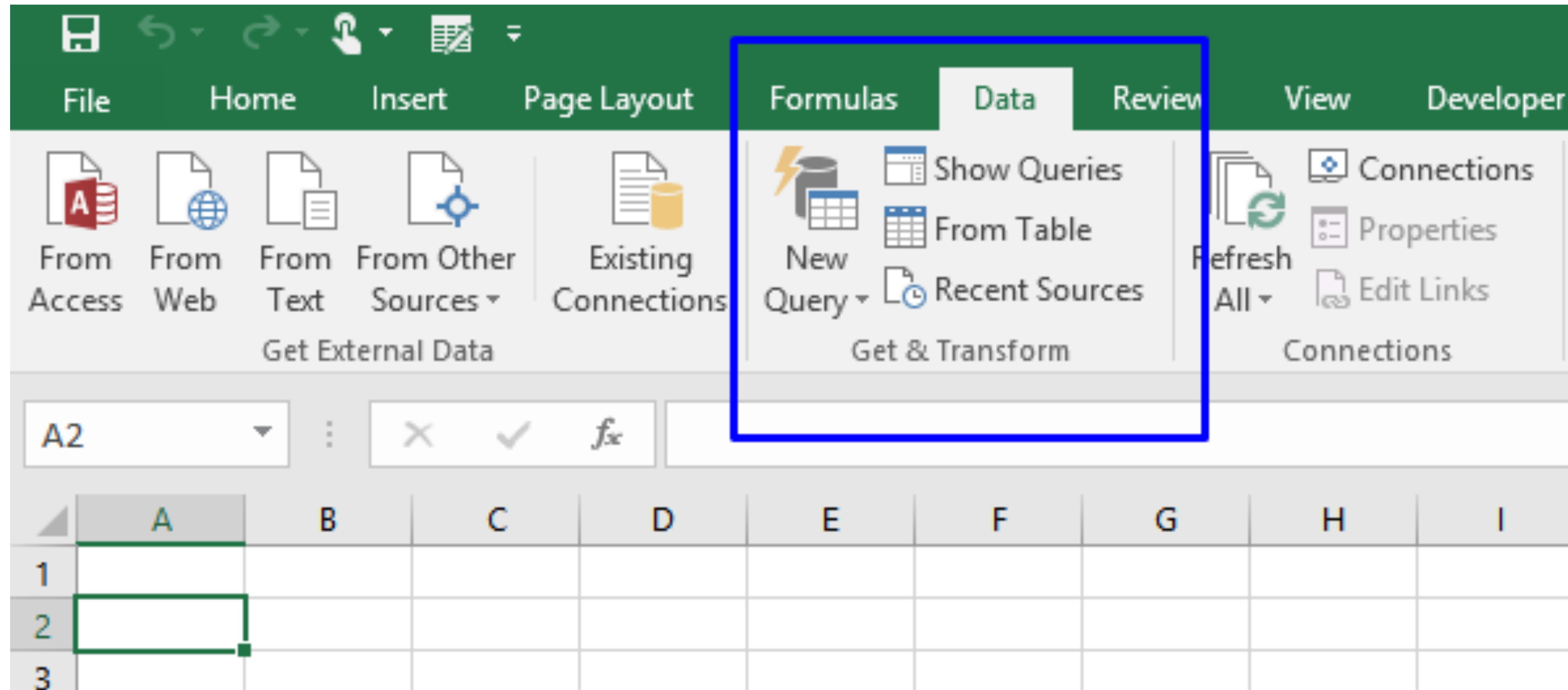
The "PivotTable Field List" task pane on the right shows the following fields:

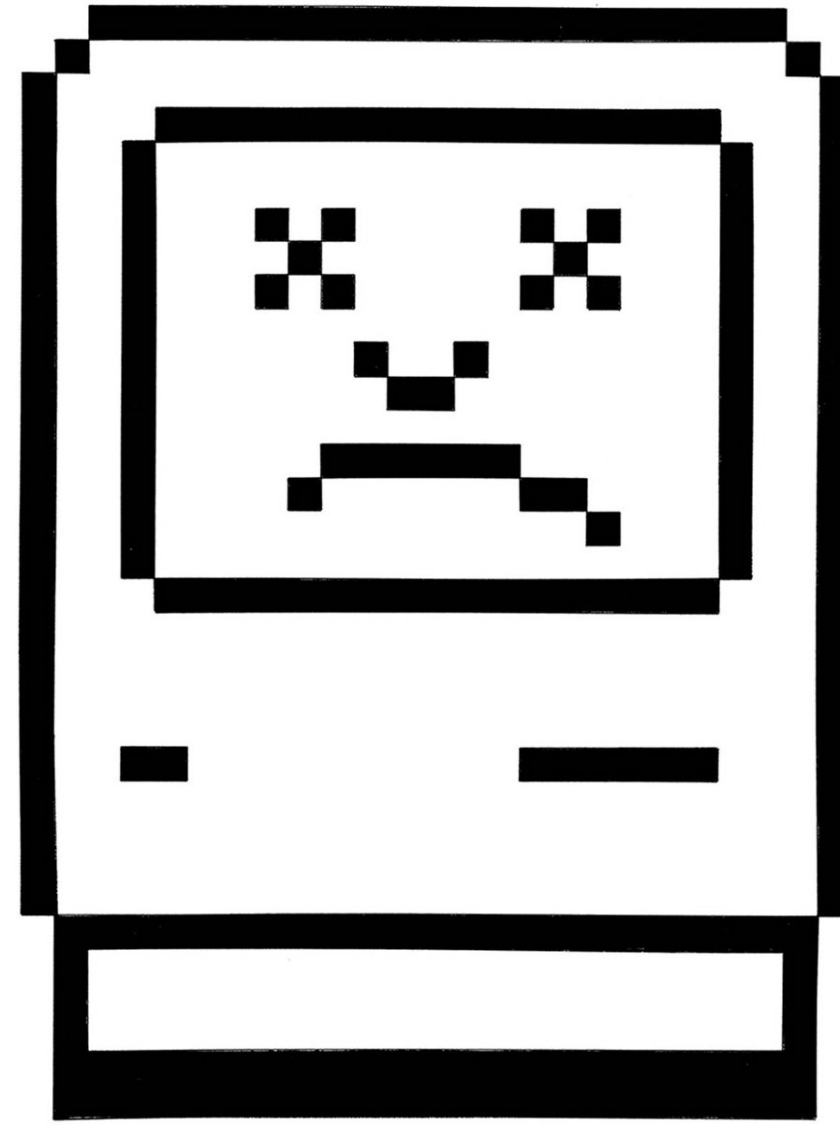
- Choose fields to add to report:
  - ☐ Date
  - ☐ Region
  - ☒ **Product**
  - ☐ Category
  - ☐ Qty
  - ☒ **Sales**
- Drag fields between areas below:
  - Report Filter: (empty)
  - Column Labels: (empty)
  - Row Labels: Product
  - Values: Sum of Sales
- Defer Layout Update: ☐
- Update: [button]



What about  
querying **across**  
spreadsheets?

# Get and Transform





**No Mac support.**

Structure changes?

Type changes?

Column Renames?

**Have fun re-loading.**

And what  
about **joins**?

# There's VLOOKUP

`=VLOOKUP ("Product 1",  
Prices!$A$2:$B$9,2,FALSE)`

... but, like, eww.

Data **inside** a  
spreadsheet is hard to  
connect to data **outside**  
that spreadsheet.

**Summary:**

**Spreadsheets are**

**bad at types and**

**hard to query**



# **This talk:**

1. Spreadsheets are great
2. Spreadsheets are a problem
- 3. How we can fix it**



What about **databases**?

Databases are **great**  
in ways that  
spreadsheets **aren't**.

Databases are great at  
data type **definition**  
and **enforcement**.

# So Many Types of Types!

Numeric

Enumerated

XML

Monetary

Geometric

JSON

Character

Network Address

Arrays

Binary

Bit String

Composite

Date/Time

Text Search

Range

Boolean

UUID

Pseudo-Types



Databases are  
**purpose-built** for  
queries and joins.

**BUT**

Databases  
aren't as **approachable**  
as spreadsheets.

phpMyAdmin

Current Server: phpMyAdmin demo - MySQL

(Recent tables) ...

filter databases by name

<< < 3

- Usuarios
- Usuarios1
- VSet
- VsetiAdmin
- Xss
- uam
- uam2
- ube\_db
- victoria\_base
- vseti
- world
  - New
  - City
  - Country
  - CountryLanguage

Structure SQL Search Query Export Import Operations Privileges More

world.City

- ID : int(11)
- Name : char(35)
- CountryCode : char(3)
- District : char(20)
- Population : int(11)

world.Country

- Code : char(3)
- Name : char(52)
- Continent : enum('Asia','Europe','North America','Africa','Oceania','Antarctica','South America')
- Region : char(26)
- SurfaceArea : float(10,2)
- IndepYear : smallint(6)
- Population : int(11)
- LifeExpectancy : float(3,1)
- GNP : float(10,2)
- GNPOld : float(10,2)
- LocalName : char(45)
- GovernmentForm : char(45)
- HeadOfState : char(60)
- Capital : int(11)
- Code2 : char(2)

world.CountryLanguage

- CountryCode : char(3)
- Language : char(30)
- IsOfficial : enum('T','F')
- Percentage : float(4,1)

```
$ psql -d postgres
psql (10.4, server 9.6.9)
Type "help" for help.

postgres=#
```

Databases  
aren't as **flexible**  
as spreadsheets.

Databases are good at  
**storing** and **querying** data.

**But that's it.**



Spreadsheets and  
databases have  
**complementary**  
skillsets.

So, what do we  
**do** about it?

# How to **Solve** Your Spreadsheet Problem

1. Identify the use case.
2. Stop the spread.
3. Backfill.

**1. Identify the use case.**

2. Stop the spread.

3. Backfill.

Every spreadsheet  
solves a **problem**.

What is that problem?

What's the **business need**?

**How much** data is there?

**How fast** does it change?

**How frequent** are additions?



1. Identify the use case.

**2. Stop the spread.**

3. Backfill.

Give new data a  
**structured home.**



**Ragic!**

No custom apps.

**At least at first.**

Optimize for  
**Speed**

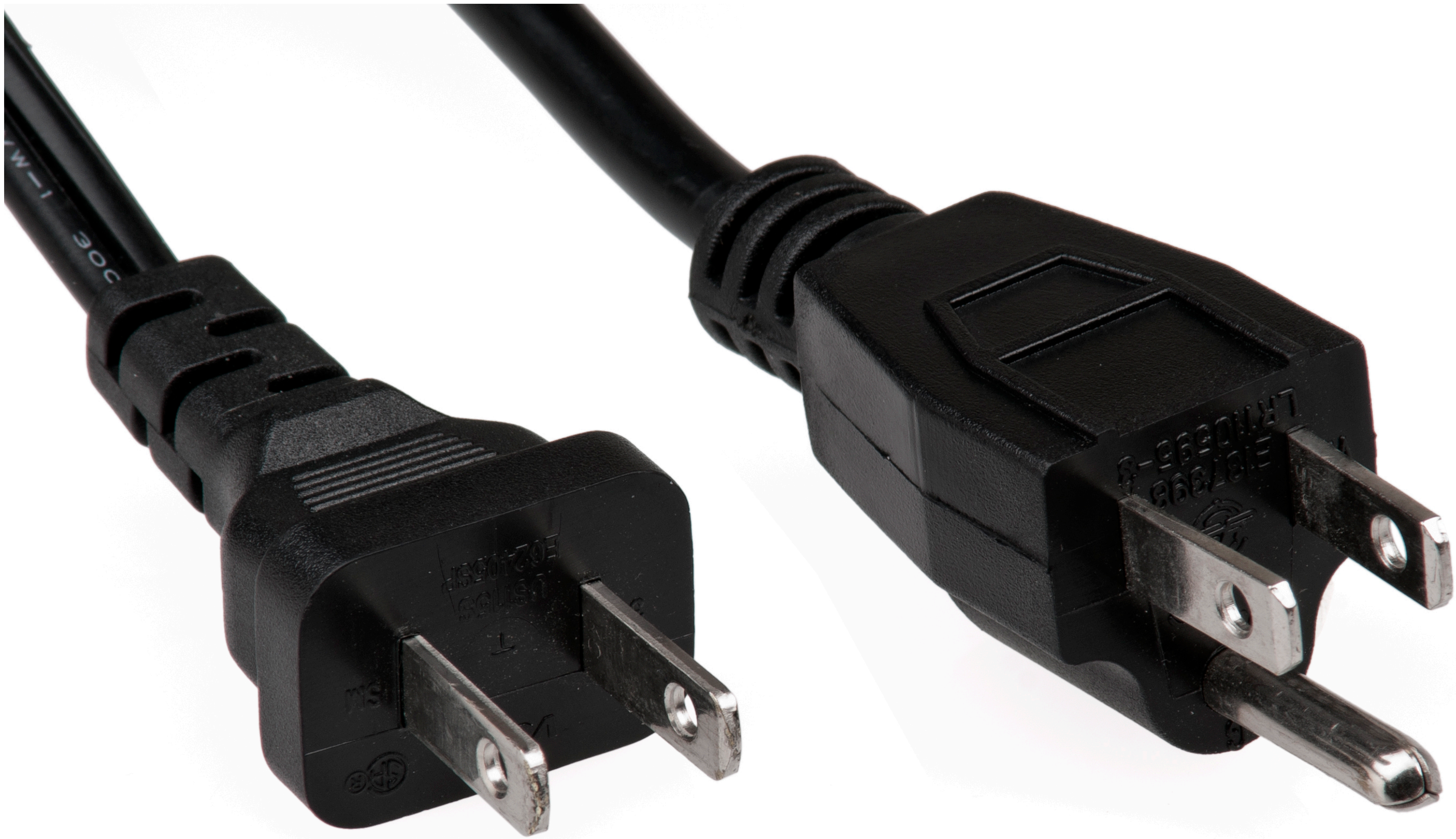
1. Identify the use case.
2. Stop the spread.
- 3. Backfill.**

It's time for some  
**Data Wrangling.**

(yee-haw 🙄)

Writing one-off  
scripts is *sometimes*  
the best option.







Sample 1 - First 500KB

11 Columns2,980 Rows2 Data Types

Grid

Columns: AllTransformed - 2 Columns  
Rows: AllTransformed - 2,979 Rows

Filter in grid

Source

Preview

ABC	ID	ABC	column3	ABC	column4	ABC	column1	ABC	column5	ABC	column6	ABC	
2,839 Categories		258 Categories		101 Categories		1 Category		2,475 Categories		2,475 Categories		2,959 Categories	
1	customer_id		first_name		last_name				SSN		credit_card		address
2	"4abe6b808c96e647239677f2a9f247fd"		Julian		"Russell"		"		"451-59-0366"		"4516009576471550"		"2166.Cedar.Lane
3	"d8983c31f0c1031ca2837f42852fbf24"		Nathan		"Davis"		"		"308-61-6226"		"4407614812304060"		"7475.Madison.St
4	"f8ebc4a9d5c03b7e2f934019fc10e9d4"		Elijah		"Wright"		"		"593-19-3579"		"5584472636741872"		"48533.2nd.Stree
5	"7d9c5c49ad12e8233c558dd88ed3c143"		Cole		"Thomas"		"		"177-74-6463"		"4257017589440200"		"2366.Linden.Str
6	"334ae2126c83dffbe28bd8a13d4ae50b"		Andrew		"Green"		"		"557-30-0305"		"5477064333168580"		"4252.10th.ST,.I
7	"af67b3a6f43ff02dfedb81ee94cd0bf5"		Adam		"Howard"		"		"076-69-2166"		"5409820014340117"		"278.Orange.Stre
8	"12f35d87b7c6e54aec5593e4c19b9824"		Andrew		"Price"		"		"457-96-9416"		"4087559818775316"		"953.Prospect.ST
9	"c98623c793c911e68e9c4b7502429983"		Erin		"Barnes"		"		" "		" "		"370.Dogwood.Dri
10	"4435b2c7c3712c154bca9d76427ba72f"		Daniel		"Perez"		"		"329-36-9209"		"5290545373364620"		"6268.Fairway.Dr
11	"30a359c8d57c68de61eb5be6128d8d37"		Blake		"Bell"		"		"071-17-4141"		"4477504255299526"		"7709.Holly.Driv

SUGGESTIONS

Extract on: ````

ABC	column4	ABC	column1
	last_name		
	"Russell"		"
	"Davis"		"

Affects 1 column, 2979 rowsCreates 1 column

Countpattern on: ````

ABC	column4	#	column1
	last_name	0	
	"Russell"	2	
	"Davis"	2	

Affects 1 column, 2979 rowsCreates 1 column

Extractlist on: `{any}+` delimiter: ````

ABC	column4		column1
	last_name		["last_name"]
	"Russell"		["","Russell",""]
	"Davis"		["","Davis",""]

Affects 1 column, all rowsCreates 1 column

CancelModifyAdd to Script

<https://www.trifacta.com/start-wrangling/>

Sample 1 - First 500KB 11 Columns 2,980 Rows 2 Data Types Grid Columns: All Transformed - 2 Columns Rows: All Transformed - 2,979 Rows Filter in grid

Source	Preview
ABC ID	ABC column3
ABC column4	ABC column1
ABC column5	ABC column6
ABC	ABC

2,839 Categories 258 Categories 101 Categories 1 Category 2,475 Categories 2,4 Categories 2,959 Categories

1 customer\_id 2 "4abe6b808c96e647239677f2a9f2" 3 "d8983c31f0c1031ca2837f42852f" 4 "f8ebc4a9d5c03b7e2f934019fc10" 5 "7d9c5c49ad12e8233c558dd88ed3c143" 6 "334ae2126c83dffbe28bd8a13d4ae50b" 7 "af67b3a6f43ff02dfedb81ee94cd0bf5" 8 "12f35d87b7c6e54aec5593e4c19b9824" 9 "c98623c793c911e68e9c4b750242" 10 "4435b2c7c3712c154bca9d76427ba72f" 11 "30a359c8d57c68de61eb5be6128d8d37" 12

2,839 Categories 258 Categories 101 Categories 1 Category 2,475 Categories 2,4 Categories 2,959 Categories

1 customer\_id 2 "4abe6b808c96e647239677f2a9f2" 3 "d8983c31f0c1031ca2837f42852f" 4 "f8ebc4a9d5c03b7e2f934019fc10" 5 "7d9c5c49ad12e8233c558dd88ed3c143" 6 "334ae2126c83dffbe28bd8a13d4ae50b" 7 "af67b3a6f43ff02dfedb81ee94cd0bf5" 8 "12f35d87b7c6e54aec5593e4c19b9824" 9 "c98623c793c911e68e9c4b750242" 10 "4435b2c7c3712c154bca9d76427ba72f" 11 "30a359c8d57c68de61eb5be6128d8d37" 12

SUGGESTIONS

Extract on: `` last\_name column4 column1 "Russell" "Davis" Affects 1 column 2979 rows

Countpattern on: `` last\_name column4 # column1 last\_name 0 2 2 Affects 1 column 2979 rows

Extractlist on: `{any}+` delimiter: `` last\_name column4 ["last\_name"] ["Russell",""] ["","Davis",""] Affects 1 column all rows

Cancel Modify Add to Script

# Infer wrangle "recipe" from high-level actions.

<https://www.trifacta.com/start-wrangling/>

# **Another Option:**

## **Programming**

### **By Example**

# FlashRelate

	A	B	C	D	E	...	R
<b>1</b>		value	year	value	year		Comments
<b>2</b>	Albania	1,000	1950	930	1981		FRA 1
<b>3</b>	Austria	3,139	1951	3,177	1955		FRA 3
<b>4</b>	Belgium	541	1947	601	1950		
<b>5</b>	Bulgaria	2,964	1947	3,259	1958		FRA 1
<b>6</b>	Czech ...	2,416	1950	2,503	1960		NC

...  
(a)

	A	B	C	D
<b>1</b>	Albania	1,000	1950	FRA 1
<b>2</b>	Albania	930	1981	FRA 1

...

<b>5</b>	Austria	3,139	1951	FRA 3
<b>6</b>	Austria	3,177	1955	FRA 3

...

<b>9</b>	Belgium	541	1947	
<b>10</b>	Belgium	601	1950	

...

(b)

Provide example rows,  
synthesize *layout* transformations.

<https://github.com/microsoft/prose>



# Foofah

	A	B	C	D	E	F	G
1	Description	9/14/2009	9/15/2009	9/16/2009	9/17/2009	9/18/2009	
2	Item 4	900	0	1800	1800		
3	Item 6					1200	
4	Item 8		1800				
5							

	A	B	C	D
1	Item 4	9/14/2009	900	
2	Item 4	9/15/2009	0	
3	Item 4	9/16/2009	1800	
4	Item 4	9/17/2009	1800	
5	Item 6	9/18/2009	1200	
6	Item 8	9/15/2009	1800	
7				

Provide input/output sample, synthesize  
*layout* and *syntactic* transformations.

<https://github.com/umich-dbggroup/foofah>

# How to **Solve** Your Spreadsheet Problem

1. Identify the use case.
2. Stop the spread.
3. Backfill.

What about  
the **future**?



Spreadsheets aren't  
going anywhere,  
**for good reason.**

**Learn** from the  
spreadsheet.

Meet the users  
**where they are.**

# Thank you.

@alexras

**Consulting Inquiries:** [contact@bitsondisk.com](mailto:contact@bitsondisk.com)