DAX Physics 101:
Demystifying DAX Evaluation Context

Featuring:
Mr. Brian Grant
Sr. Analytics Consultant
skypoint.ai

skypoint

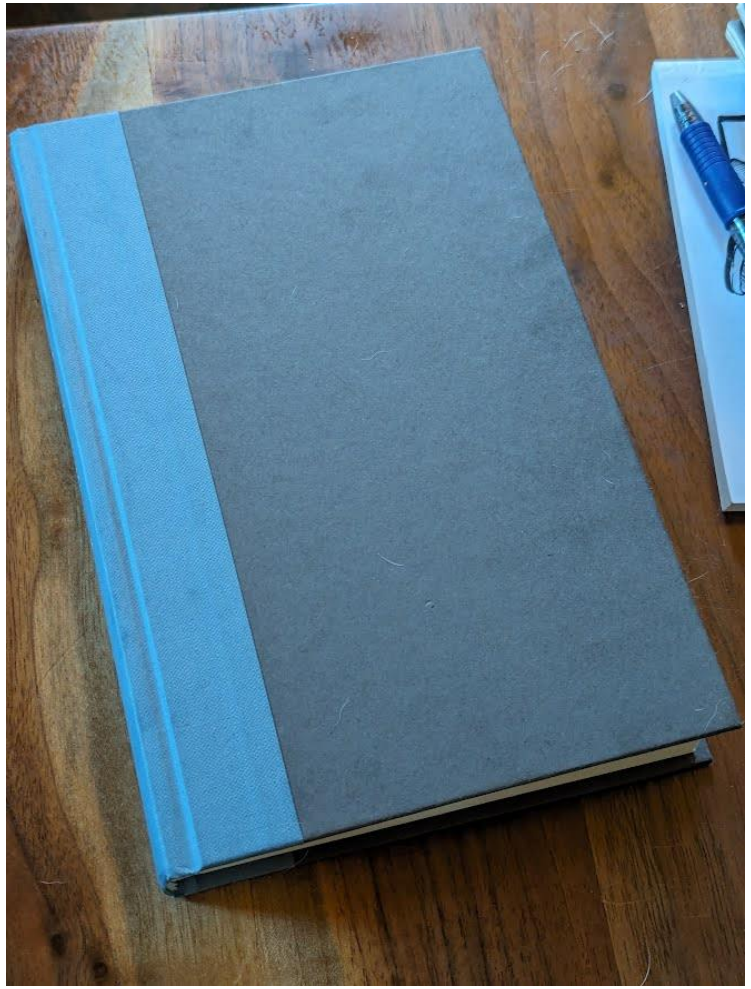
A Little Background

People have been asking me to write a book for forever.

I decided to create a blog to act as a first draft.

This let me get ideas out on paper (sort of) while working out specifics about where/how certain ideas get introduced.

This presentation summarizes a big, big chunk of the content of my blog / proto-book.



Evaluation Context

Evaluation Context is terrifying.

EVALUATION CONTEXT

Not so much the concept, mind you, mainly the name.

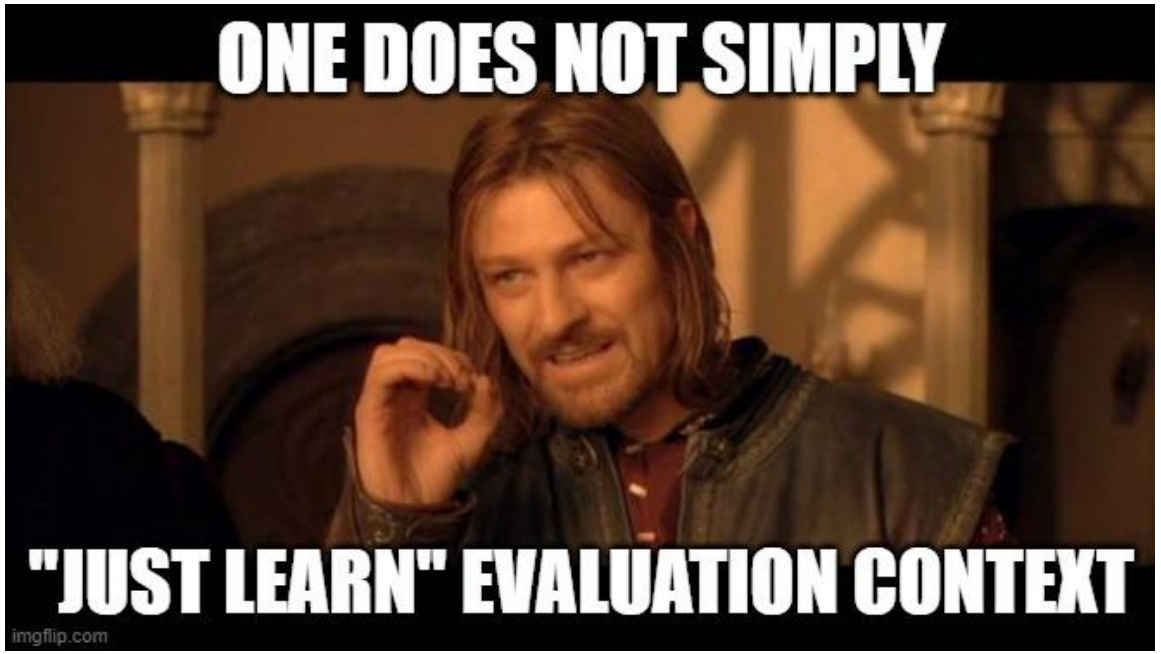
Don’t get me wrong, it is tricky, but mere mortals (like you and me) can absolutely understand it.
If you can understand a list of groceries, you can understand Evaluation Context.

Evaluation Context

This is a core concept of DAX, and if we can do the work to understand it,
we will start to see DAX not as a giant angry, spiteful god, prone to whims of fury;
but instead see it as a simple machine that moves tables around for us.

Evaluation Context

There’s a small problem though…



Before we can understand Evaluation Context,
we have to understand a couple other related concepts first.

What We'll Learn Today

```
Total Transactions =
COUNTROWS( Sale )
```

To start we'll talk about **Table References** and how, in general, when you type in the name of a table what you get is probably different than you expect.

```
SUMX(
    Sale,
    Sale[Qty] * Sale[Price]
)
```

Next, we'll review the concept of **Sub-Formulas** ("sub-expressions") and how what often looks like one big formula is actually several smaller formulas chained together.

Row Context

Filter Context

With all that done we'll be able to get a solid understanding of **Evaluation Context** and how it gets used by DAX sub-formulas to produce the answers we see in our Power BI reports.

Where Are We On Mount DAX?

Advanced DAX

Deep mechanical understanding.
Must learn hard ideas.

Very powerful
Very difficult

Advanced DAX Prep

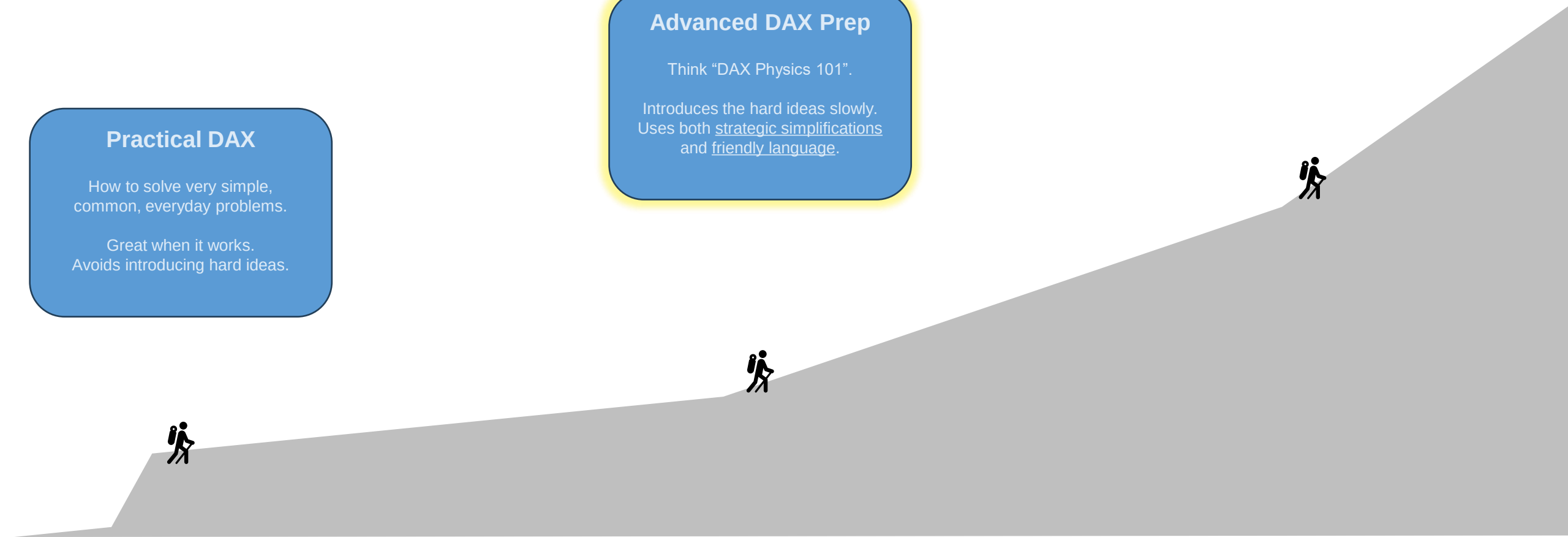
Think “DAX Physics 101”.

Introduces the hard ideas slowly.
Uses both strategic simplifications and friendly language.

Practical DAX

How to solve very simple, common, everyday problems.

Great when it works.
Avoids introducing hard ideas.



Understanding Table References

(Setting Things Up Part 1)

The Data Model: What We Load Into Power BI

Sale

"All the sales transactions"

Sale[Shift]	Sale[Qty]	Sale[Price]	Sale[Dish]
Lunch	2	$10	Pasta
Lunch	1	$8	Burger
Lunch	1	$10	Pasta
Lunch	3	$8	Burger
Dinner	2	$8	Burger
Dinner	1	$12	Salad
Dinner	2	$10	Pasta

M:1
(LU↗)

Dish

"All the dishes on the menu"

Dish[ID]	Dish[Type]	Dish[Cost]
Pasta	Regular	$4
Burger	H. Special	$5
Salad	Regular	$6

A very simple Data Model with two tables and a relationship between them.

The Four Tables of DAX (and Power BI)

Model Table
"Physical Table of the Data Model"

Sale

Shift	Qty	Price	Dish
Lunch	2	$10	Pasta
Lunch	1	$8	Burger
Lunch	1	$10	Pasta
Lunch	3	$8	Burger
Dinner	2	$8	Burger
Dinner	1	$12	Salad
Dinner	2	$10	Pasta

Very easy to see
(Data Viewer in Power BI)

Un-summarized data

Just a few DAX functions work with these.

Temp Table
"Table Value"

Sale[Shift]	Sale[Qty]	Sale[Price]	Sale[Dish]	Dish[ID]	Dish[Type]	Dish[Cost]
Lunch	1	$8	Burger	Burger	H. Special	$5
Lunch	3	$8	Burger	Burger	H. Special	$5

Impossible to see directly
(CONCATONATEX, TOCSV)

Temporary (logical) copies used in the process of summarizing data

Most DAX functions are designed to work with these.

Summary Table
"Result of a DAX Query"

Results

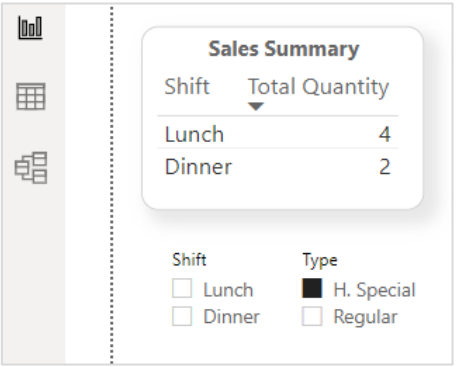
Shift	Total Qty
Lunch	4
Dinner	2

Output | Results | Query History

Fairly easy to see
(DAX Studio/Query Panel)

Summarization of the data in the Data Model

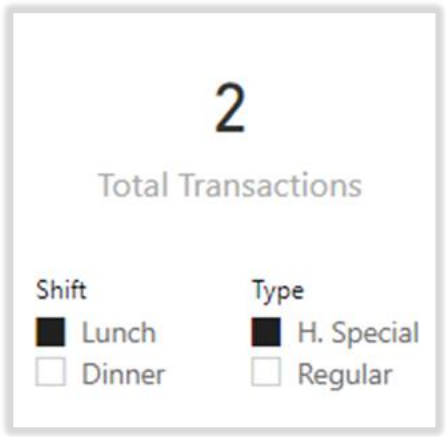
Table Visual
"Table Visual"



Very easy to see
(Power BI Page)

Rendering of Summary Table with fonts, titles, etc.

Who's Counting Anyways?




```
Total Transactions =
COUNTROWS( Sale )
```

What are you counting the rows of?

This Right?

Sale

Shift	Qty	Price	Dish
Lunch	2	$10	Pasta
Lunch	1	$8	Burger
Lunch	1	$10	Pasta
Lunch	3	$8	Burger
Dinner	2	$8	Burger
Dinner	1	$12	Salad
Dinner	2	$10	Pasta

The table called "Sale" in your Date Model

7 Rows / 4 Col

Nope, This:

Sale[Shift]	Sale[Qty]	Sale[Price]	Sale[Dish]	Dish[ID]	Dish[Type]	Dish[Cost]
Lunch	1	$8	Burger	Burger	H. Special	$5
Lunch	3	$8	Burger	Burger	H. Special	$5

A (logical) copy of that table with columns added & filters applied

2 Rows/ 7 Col

By default, table references in DAX give you a filtered copy of that table.

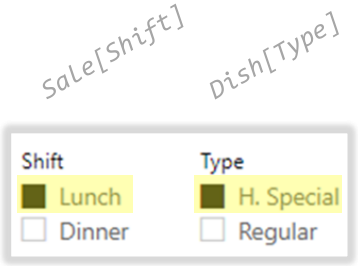
Filtered (Logical) Copy

```
Total Transactions =
COUNTROWS( Sale )
```

The copy is filtered so that the measure "responds" to slicer selection.

Sale[Shift]	Sale[Qty]	Sale[Price]	Sale[Dish]	Dish[ID]	Dish[Type]	Dish[Cost]
Lunch	1	$8	Burger	Burger	H. Special	$5
Lunch	3	$8	Burger	Burger	H. Special	$5

Columns from the related table of Dish have been added to allow filters on those related columns (like Type) to work.

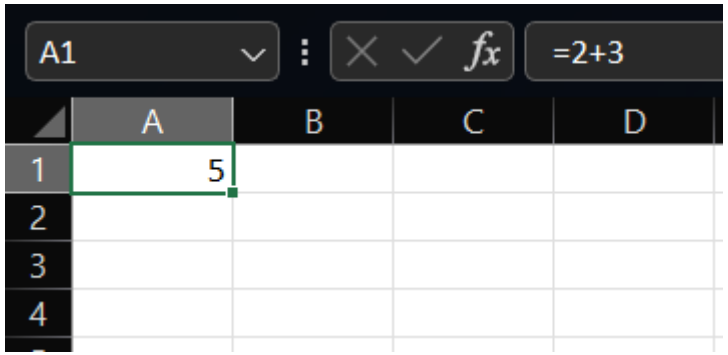


Picture Perfect

This is less weird than it sounds.

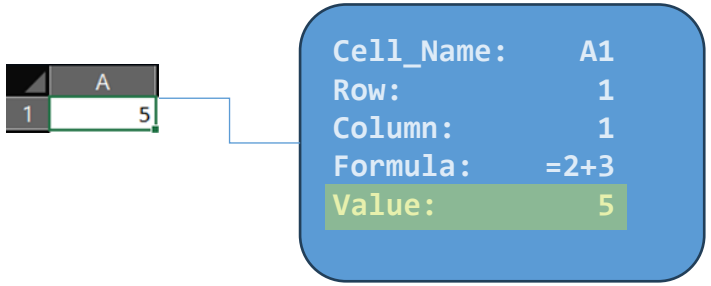
In fact, it is very Excel like…

Back to Excel

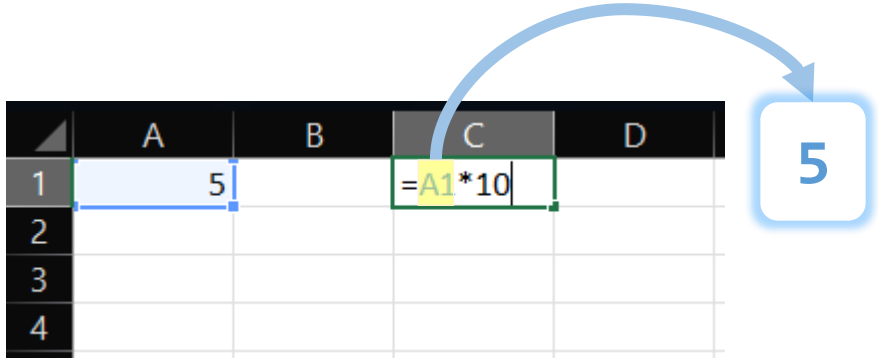


Let's start with a very simple cell.

Cell References Automatically Give You Values



The cell A1 has lots of things in it…



… but reference A1 in a formula and Excel will assume you want the cell's value.

99% of the time this is what you want.

Only specialty functions (ROW, COLUMN, FORMULATEXT) will use the reference to grab the other things.

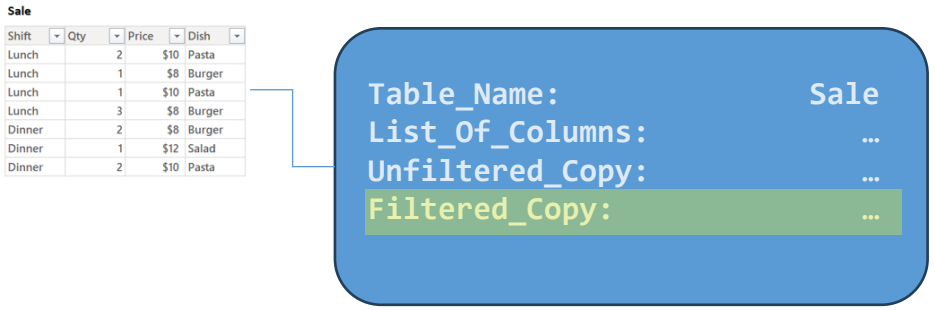
Back to DAX

Sale

Shift	Qty	Price	Dish
Lunch	2	$10	Pasta
Lunch	1	$8	Burger
Lunch	1	$10	Pasta
Lunch	3	$8	Burger
Dinner	2	$8	Burger
Dinner	1	$12	Salad
Dinner	2	$10	Pasta

Let's start with a very simple Data Model table.

Table References Automatically Give You Filtered Copies



The Model Table "Sale" has lots of things in it…

```
Total Transactions =
COUNTROWS( Sale )
```

Sale[Shift]	Sale[Qty]	Sale[Price]	Sale[Dish]	Dish[ID]	Dish[Type]	Dish[Cost]
Lunch	1	$8	Burger	Burger	H. Special	$5
Lunch	3	$8	Burger	Burger	H. Special	$5

… but reference "Sale" in a formula and DAX will assume you want a filtered copy.

99% of the time this is what you want.

Only specialty functions (REMOVEFILTERS, ALL, ISFILTERED) will use the reference to grab the other things.

Getting the Filtered Copy

```
Total Transactions =
COUNTROWS( Sale )
```

But how does DAX get you that “Filtered Copy”?

It preforms 3 simple steps…

Getting the Filtered Copy

Sale

Shift	Qty	Price	Dish
Lunch	2	$10	Pasta
Lunch	1	$8	Burger
Lunch	1	$10	Pasta
Lunch	3	$8	Burger
Dinner	2	$8	Burger
Dinner	1	$12	Salad
Dinner	2	$10	Pasta

M:1 (LU↗)

Dish

ID	Type	Cost
Pasta	Regular	$4
Burger	H. Special	$5
Salad	Regular	$6

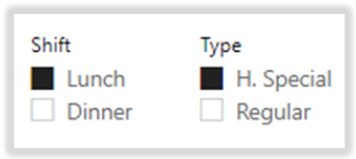
Sale

Shift	Qty	Price	Dish
Lunch	2	$10	Pasta
Lunch	1	$8	Burger
Lunch	1	$10	Pasta
Lunch	3	$8	Burger
Dinner	2	$8	Burger
Dinner	1	$12	Salad
Dinner	2	$10	Pasta

M:1 (LU↗)

Dish

ID	Type	Cost
Pasta	Regular	$4
Burger	H. Special	$5
Salad	Regular	$6



(Slicers put filters in the Filter Context)

Filter Context:

Sale[Shift]
Lunch

Dish[Type]
H. Special

(List of current filters)

Sale[Shift]	Sale[Qty]	Sale[Price]	Sale[Dish]
Lunch	2	$10	Pasta
Lunch	1	$8	Burger
Lunch	1	$10	Pasta
Lunch	3	$8	Burger
Dinner	2	$8	Burger
Dinner	1	$12	Salad
Dinner	2	$10	Pasta

The Simple Copy

Make a copy as a Temp Table.

*(Like SELECT * FROM Sale)*

Sale[Shift]	Sale[Qty]	Sale[Price]	Sale[Dish]	Dish[ID]	Dish[Type]	Dish[Cost]
Lunch	2	$10	Pasta	Pasta	Regular	$4
Lunch	1	$8	Burger	Burger	H. Special	$5
Lunch	1	$10	Pasta	Pasta	Regular	$4
Lunch	3	$8	Burger	Burger	H. Special	$5
Dinner	2	$8	Burger	Burger	H. Special	$5
Dinner	1	$12	Salad	Salad	Regular	$6
Dinner	2	$10	Pasta	Pasta	Regular	$4

The Super Lookup

Use Relationships to lookup columns.
AKA “Table Expansion”

(Like LEFT JOIN Dish ON Relationship)

Sale[Shift]	Sale[Qty]	Sale[Price]	Sale[Dish]	Dish[ID]	Dish[Type]	Dish[Cost]
Lunch	2	$10	Pasta	Pasta	Regular	$4
Lunch	1	$8	Burger	Burger	H. Special	$5
Lunch	1	$10	Pasta	Pasta	Regular	$4
Lunch	3	$8	Burger	Burger	H. Special	$5
Dinner	2	$8	Burger	Burger	H. Special	$5
Dinner	1	$12	Salad	Salad	Regular	$6
Dinner	2	$10	Pasta	Pasta	Regular	$4

Auto Filtering

Apply all filters in the Filter Context.

(Like WHERE EXISTS filters aka Semi Join)

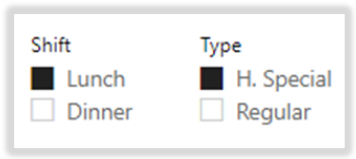
Sale[Shift]	Sale[Qty]	Sale[Price]	Sale[Dish]	Dish[ID]	Dish[Type]	Dish[Cost]
Lunch	1	$8	Burger	Burger	H. Special	$5
Lunch	3	$8	Burger	Burger	H. Special	$5

Getting the Filtered Copy

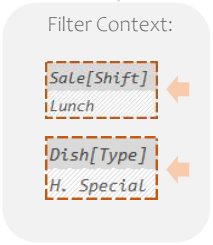
Filter Context

The name sounds intimidating, but it's nothing more than:

"The list of filters to apply during Auto-Filtering"



(Slicers put filters in the Filter Context)



(List of current filters)

Sale[Shift]	Sale[Qty]	Sale[Price]	Sale[Dish]	Dish[ID]	Dish[Type]	Dish[Cost]
Lunch	2	$10	Pasta	Pasta		$4
Lunch	1	$8	Burger	Burger	H. Special	$5
Lunch	1	$10	Pasta	Pasta		$4
Lunch	3	$8	Burger	Burger	H. Special	$5
	2	$8	Burger	Burger	H. Special	$5
	1	$12	Salad	Salad		$6
	2	$10	Pasta	Pasta		$4

Auto Filtering

Apply all filters in the Filter Context.

(Like WHERE EXISTS filters aka Semi Join)

In DAX, filters are tables.
For humans, this is weird.
For a database, this makes total sense
(and is very fast).

Getting the Filtered Copy

Sale

Shift	Qty	Price	Dish
Lunch	2	$10	Pasta
Lunch	1	$8	Burger
Lunch	1	$10	Pasta
Lunch	3	$8	Burger
Dinner	2	$8	Burger
Dinner	1	$12	Salad
Dinner	2	$10	Pasta

M:1 (LU↗)

Dish

ID	Type	Cost
Pasta	Regular	$4
Burger	H. Special	$5
Salad	Regular	$6

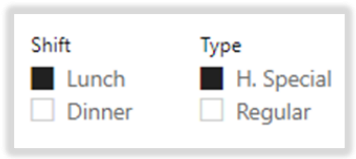
Sale

Shift	Qty	Price	Dish
Lunch	2	$10	Pasta
Lunch	1	$8	Burger
Lunch	1	$10	Pasta
Lunch	3	$8	Burger
Dinner	2	$8	Burger
Dinner	1	$12	Salad
Dinner	2	$10	Pasta

M:1 (LU↗)

Dish

ID	Type	Cost
Pasta	Regular	$4
Burger	H. Special	$5
Salad	Regular	$6



(Slicers put filters in the Filter Context)

Filter Context:

Sale[Shift]
Lunch

Dish[Type]
H. Special

(List of current filters)

Sale[Shift]	Sale[Qty]	Sale[Price]	Sale[Dish]
Lunch	2	$10	Pasta
Lunch	1	$8	Burger
Lunch	1	$10	Pasta
Lunch	3	$8	Burger
Dinner	2	$8	Burger
Dinner	1	$12	Salad
Dinner	2	$10	Pasta

The Simple Copy

Make a copy as a Temp Table.

*(Like SELECT * FROM Sale)*

Sale[Shift]	Sale[Qty]	Sale[Price]	Sale[Dish]	Dish[ID]	Dish[Type]	Dish[Cost]
Lunch	2	$10	Pasta	Pasta	Regular	$4
Lunch	1	$8	Burger	Burger	H. Special	$5
Lunch	1	$10	Pasta	Pasta	Regular	$4
Lunch	3	$8	Burger	Burger	H. Special	$5
Dinner	2	$8	Burger	Burger	H. Special	$5
Dinner	1	$12	Salad	Salad	Regular	$6
Dinner	2	$10	Pasta	Pasta	Regular	$4

The Super Lookup

Use Relationships to lookup columns.
AKA "Table Expansion"

(Like LEFT JOIN Dish ON Relationship)

Sale[Shift]	Sale[Qty]	Sale[Price]	Sale[Dish]	Dish[ID]	Dish[Type]	Dish[Cost]
Lunch	2	$10	Pasta	Pasta	Regular	$4
Lunch	1	$8	Burger	Burger	H. Special	$5
Lunch	1	$10	Pasta	Pasta	Regular	$4
Lunch	3	$8	Burger	Burger	H. Special	$5
Dinner	2	$8	Burger	Burger	H. Special	$5
Dinner	1	$12	Salad	Salad	Regular	$6
Dinner	2	$10	Pasta	Pasta	Regular	$4

Auto Filtering

Apply all filters in the Filter Context.

(Like WHERE EXISTS filters aka Semi Join)

Sale[Shift]	Sale[Qty]	Sale[Price]	Sale[Dish]	Dish[ID]	Dish[Type]	Dish[Cost]
Lunch	1	$8	Burger	Burger	H. Special	$5
Lunch	3	$8	Burger	Burger	H. Special	$5

Any function that responds to slicers, uses the exact same three steps above to do so.

Think SUM, MIN, MAX, AVERAGE, DISTINCTCOUNT, VALUES

Take Aways: Table References

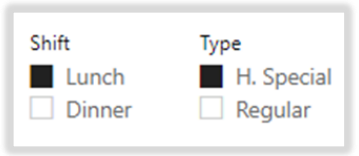
Most table references in DAX create a **filtered copy** of that table.

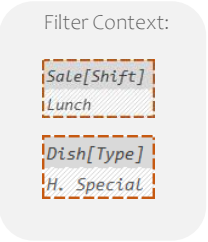
Sale

Shift	Qty	Price	Dish
Lunch	2	$10	Pasta
Lunch	1	$8	Burger
Lunch	1	$10	Pasta
Lunch	3	$8	Burger
Dinner	2	$8	Burger
Dinner	1	$12	Salad
Dinner	2	$10	Pasta

Sale[Shift]	Sale[Qty]	Sale[Price]	Sale[Dish]	Dish[ID]	Dish[Type]	Dish[Cost]
Lunch	1	$8	Burger	Burger	H. Special	$5
Lunch	3	$8	Burger	Burger	H. Special	$5

The list of filters that get applied is called the **Filter Context.**





The Sub-Formulas of DAX

(Setting Things Up Part 2)

DAX Is A Language of Sub-Formulas

Most DAX formulas actually contain one or more **Sub-Formulas** in them.

These are just a small formulas inside the larger formula.

(Technical terms: “expressions” and “sub-expressions”)

DAX Is A Language of Sub-Formulas

There are just two kinds of sub-formulas we care about:

Per Row Formula

"A sub-formula that runs once per row of a Temp Table."

New Filters Formula

"A sub-formula that runs with a new set of filters."

The Per Row Formula of Iterators

Sale[Shift]	Sale[Qty]	Sale[Price]	Sale[Dish]	Dish[ID]	Dish[Type]	Dish[Cost]
Lunch	1	$8	Burger	Burger	H. Special	$5
Lunch	3	$8	Burger	Burger	H. Special	$5

```
SUMX(
    Sale,
    Sale[Qty] * Sale[Price]
)
```

Dish[ID]	Dish[Type]	Dish[Cost]
Pasta	Regular	$4
Burger	H. Special	$5
Salad	Regular	$6

```
AVERAGEX(
    ALL( Dish ),
    IF( Dish[Cost] >= 5, 1, 0 )
)
```

[Value]
99.6
100.4
200.1

```
MAXX(
    { 99.6, 100.4, 200.1 },
    ROUND( [Value], 0 )
)
```

Argument 1:

Instructions for creating a Temp Table

Argument 2:

Formula to run for each row of that Temp Table

("Per Row Formula")

The New Filters Formula of CALCULATE

```
CALCULATE(
    COUNTROWS( Sale ),
    REMOVEFILTERS( Sale[Shift] )
)
```

"Please remove any filters on Sale[Shift]."

```
CALCULATE(
    SUM( Sale[Qty] ),
    TREATAS( { "Pasta" }, Sale[Dish] )
)
```

Sale[Dish]
Pasta

"Please add this as a filter."

```
CALCULATE(
    DISTINCTCOUNT( Sale[Dish] ),
    TREATAS( { "Lunch" }, Sale[Shift] )
)
```

Sale[Shift]
Lunch

"Please add this as a filter."

Argument 1:

Formula to run with a new set of filters

("New Filters Formula")

Argument 2:

How you want the filters to be different

(What filters do you want to remove or add?)

The Two Kinds of DAX Sub-Formulas

Per Row Formula

(Argument 2 of Iterators)

"A sub-formula that runs once per row of a Temp Table."

```
SUMX(
    Sale,
    Sale[Qty] * Sale[Price]
)
```

```
AVERAGEX(
    ALL( Dish ),
    IF( Dish[Cost] >= 5, 1, 0 )
)
```

```
MAXX(
    { 99.6, 100.4, 200.1 },
    ROUND( [Value], 0 )
)
```

Argument 1 is the instructions for building the Temp Table our sub-formula will run "for each row" of.

New Filters Formula

(Argument 1 of CALCULATE)

"A sub-formula that runs with a new set of filters."

```
CALCULATE(
    COUNTROWS( Sale ),
    REMOVEFILTERS( Sale[Shift] )
)
```

```
CALCULATE(
    SUM( Sale[Qty] ),
    TREATAS( { "Pasta" }, Sale[Dish] )
)
```

```
CALCULATE(
    DISTINCTCOUNT( Sale[Dish] ),
    TREATAS( { "Lunch" }, Sale[Shift] )
)
```

Argument 2 is the instructions for how we want the filters to be changed. We can tell CALCULATE to remove filters on certain columns or give it instructions for building a Temp Table we want added as a filter.

What's With the Non-Technical Names?

What do I tell students they're writing?

```
SUMX(
    Sale,
    ???
)
```

Understandable:

"A Per Row Formula"

Accurate:

"A Multi-Row Contextualized Scalar Value Sub-Expression"

```
CALCULATE(
    ???,
    REMOVEFILTERS( Sale[Dish] )
)
```

Understandable:

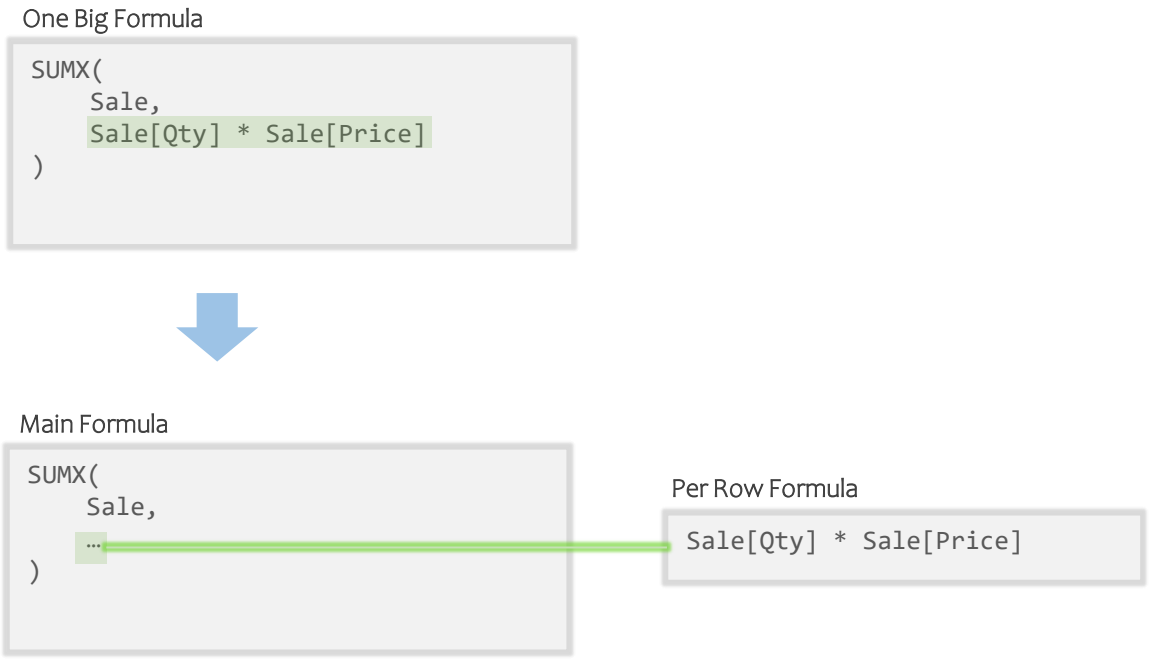
"A New Filters Formula"

Accurate:

"A Filter Contextualized Scalar Value Sub-Expression"

Formula Decomposition

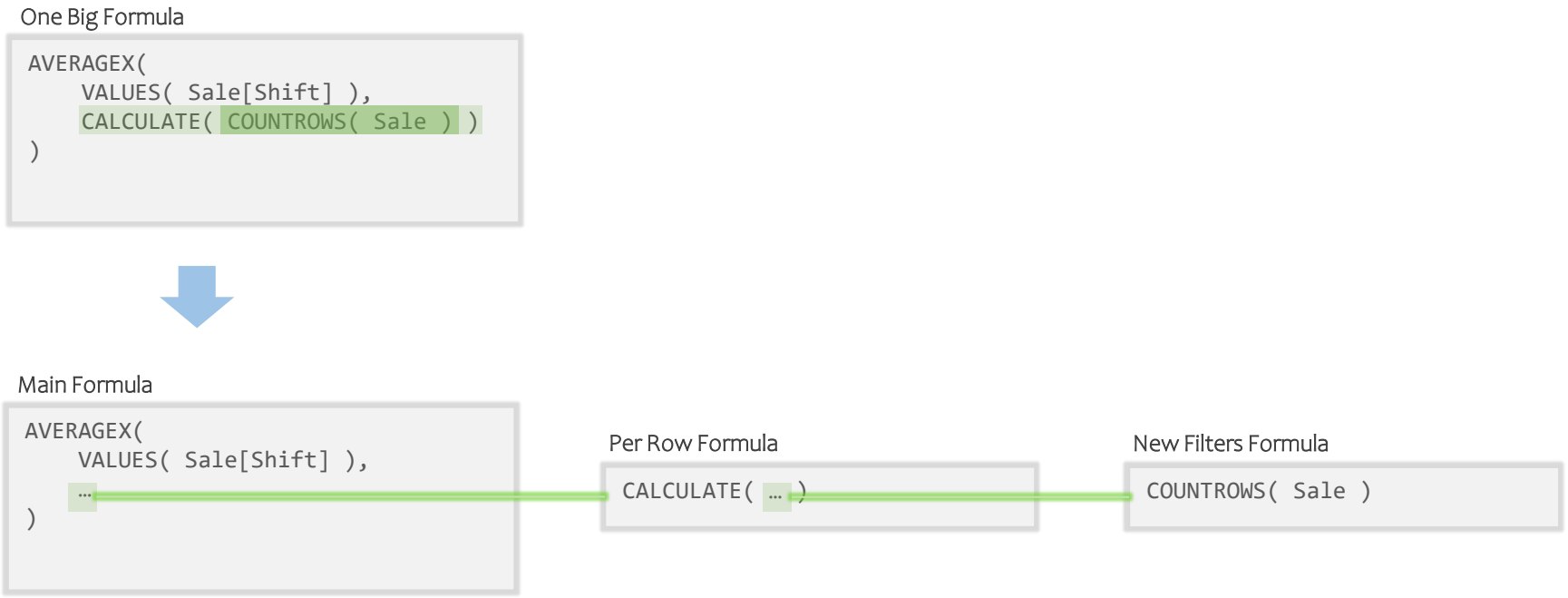
To help help us understand the sub-formulas, we can visually out into their own little boxes.



This **Decomposition** process, makes the sub-formulas bite-sized and easier to understand.

Formula Decomposition

This is especially helpful when dealing with a longer chain of sub-formulas:



Each piece is bite-sized and easier to read. There's another reason this is useful though…

Take Aways: Sub-Formulas

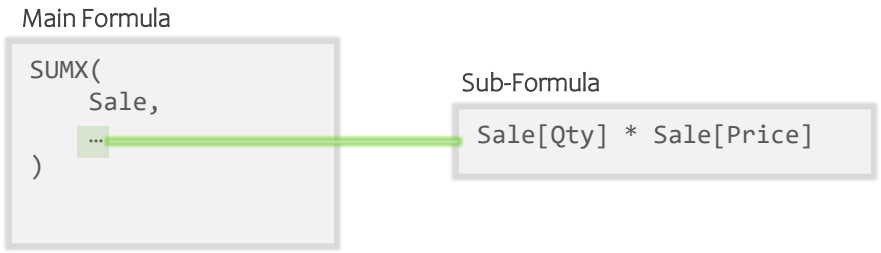
The **Per Row Formula** is a sub-formula that runs for each row of a Temp Table. (Argument 2 of Iterators)

```
SUMX(
    Sale,
    Sale[Qty] * Sale[Price]
)
```

The **New Filters Formula** is a sub-formula that runs with some changes made to the filters. (Argument 1 of CALCULATE)

```
CALCULATE(
    COUNTROWS( Sale ),
    REMOVEFILTERS( Sale[Shift] )
)
```

Decomposition is visually pulling the sub-formulas out into their own boxes to better understand the chain of formulas.

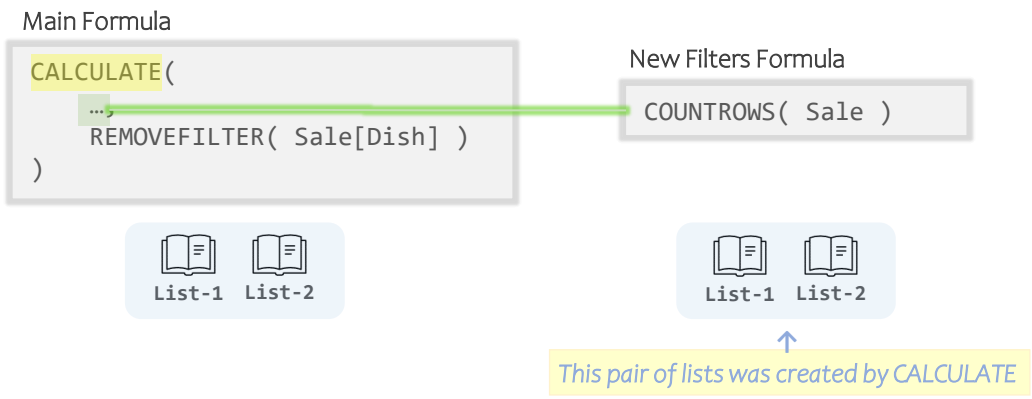


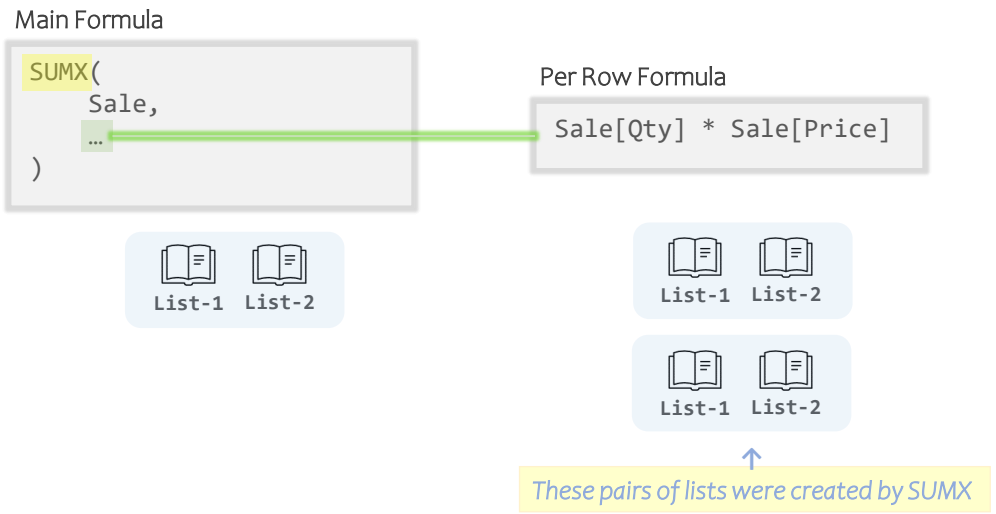
Evaluation Context

(Finally)

DAX Formulas and Sub-Formulas

Each time a sub-formula (or formula) runs, it does so with at least one pair of lists in place.





These lists hold important information that the sub-formulas might need.

Each pair of lists is given to the sub-formula by its parent function.

DAX Formulas and Sub-Formulas

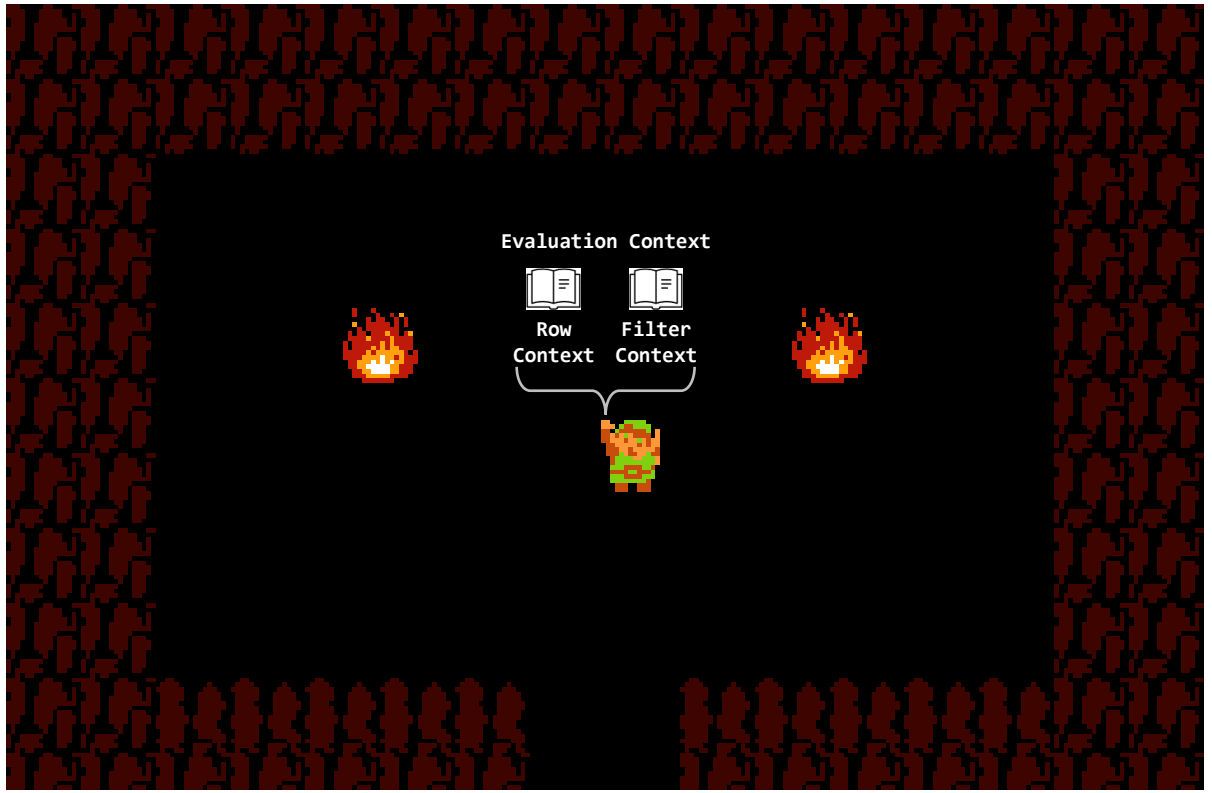
The parent function “hands down” important information to the sub-formula through these lists.

DAX Formulas and Sub-Formulas

The parent function “hands down” important information to the sub-formula through these lists.

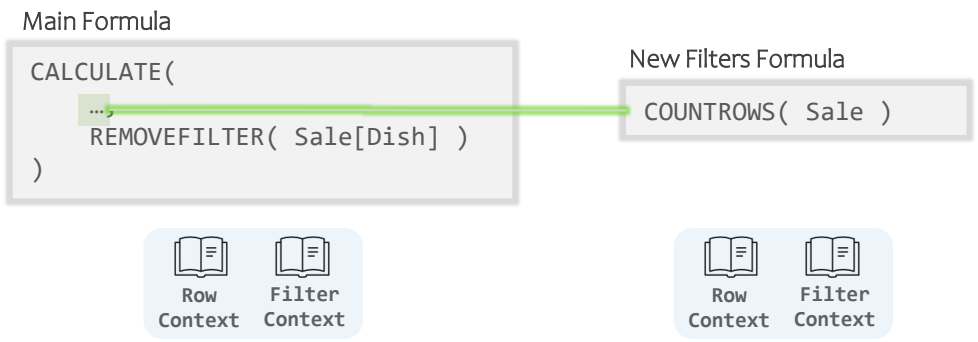
DAX Formulas and Sub-Formulas

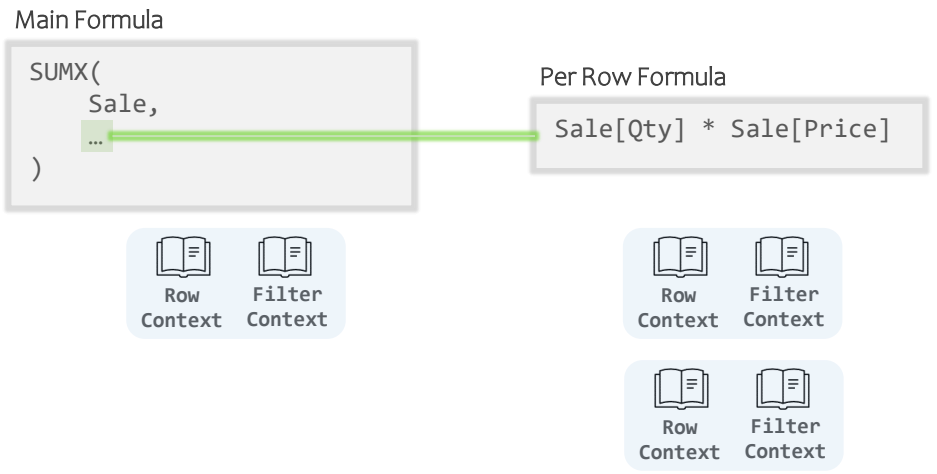
The parent function “hands down” important information to the sub-formula through these lists.



DAX Formulas and Sub-Formulas

Each time a sub-formula (or formula) runs, it does so with at least one pair of lists in place.





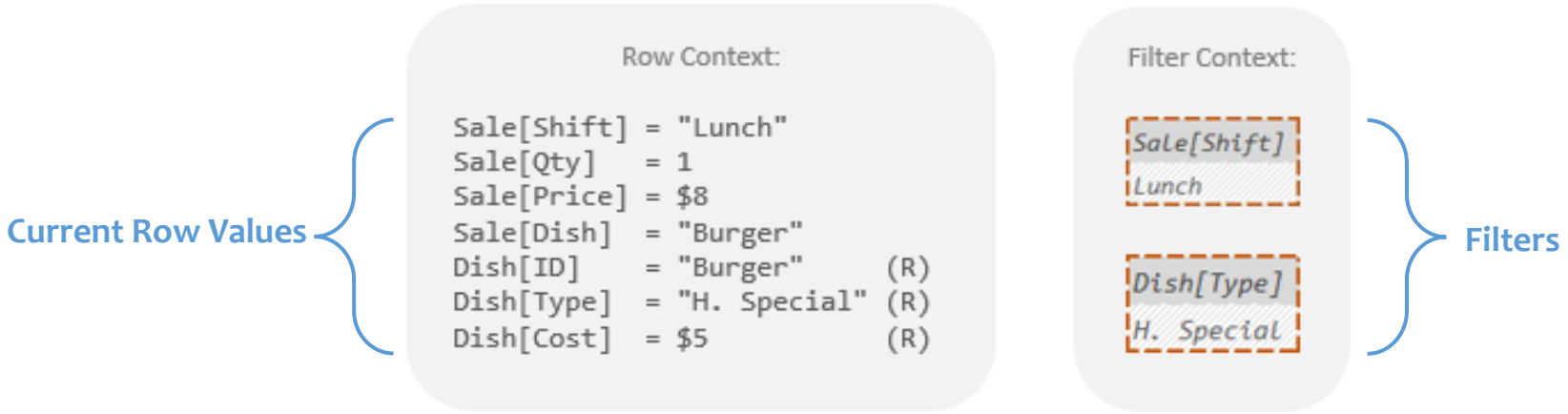
These pairs of lists hold important information that the sub-formulas might need.

Each Evaluation Context of lists is given to the sub-formula by its parent function.

DAX Evaluation Context

Evaluation Context

A pair of lists where important information is stored for a sub-formula to use.

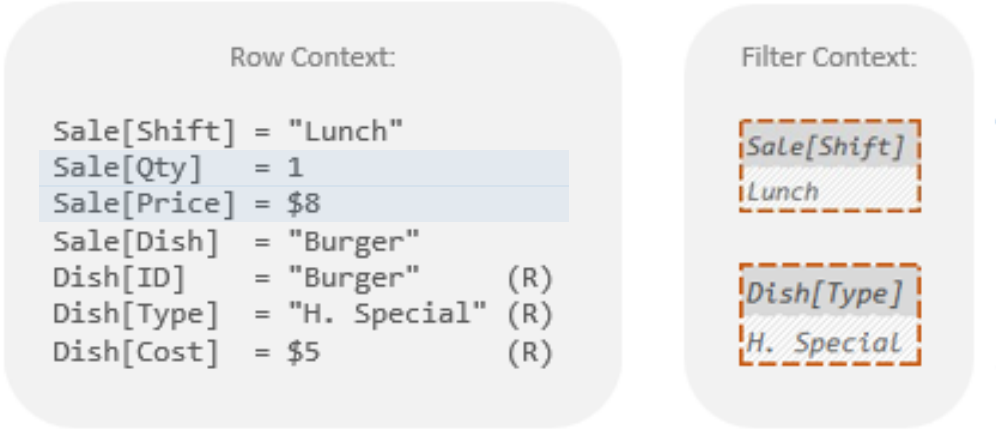


"List of all the values in the current row"

"List of all the current filters"

(When I say "list" I don't mean the computer science kind, I mean the kind that's on your fridge.)

How Does “Row Context” Get Used?



Sub-Formula

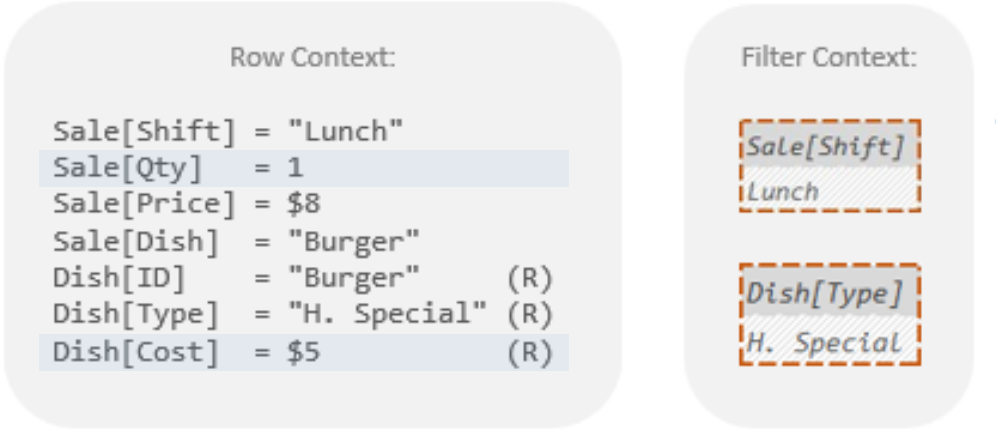
```
Sale[Qty] * Sale[Price]
```

1 * $8 → $8

The sub-formula can grab values from the Row Context and use them like a numbers/text/etc.

(Values from related columns have a little (R) and can be grabbed with the RELATED function.)

How Does “Row Context” Get Used?



Sub-Formula

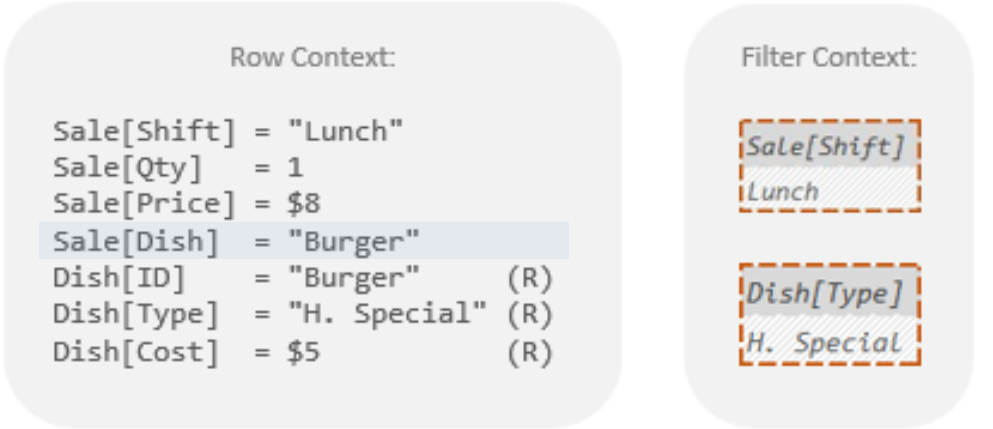
```
Sale[Qty] * RELATED( Dish[Cost] )
```

1 * $5 → $5

The sub-formula can grab values from the Row Context and use them like a numbers/text/etc.

(Values from related columns have a little (R) and can be grabbed with the RELATED function.)

How Does “Row Context” Get Used?



Sub-Formula

```
UPPER( Sale[Dish] )
```

UPPER(“Burger”) → “BURGER”

Generally, you add or multiply the row’s numbers, but you can do most Excel things here.

How Does “Filter Context” Get Used?

Row Context:

```
Sale[Shift] = "Lunch"
Sale[Qty]   = 1
Sale[Price] = $8
Sale[Dish]  = "Burger"
Dish[ID]    = "Burger"      (R)
Dish[Type]  = "H. Special"  (R)
Dish[Cost]  = $5            (R)
```

Filter Context:

Sale[Shift]
Lunch

Dish[Type]
H. Special

Sub-Formula

```
COUNTROWS( Sale )
```

Sale[Shift]	Sale[Qty]	Sale[Price]	Sale[Dish]	Dish[ID]	Dish[Type]	Dish[Cost]
Lunch	1	$8	Burger	Burger	H. Special	$5
Lunch	3	$8	Burger	Burger	H. Special	$5

The Sub-Formula might include a table reference or function that performs Auto-Filtering.

Evaluation Context: A Simple Pair of Lists

Row Context:

```
Sale[Shift] = "Lunch"
Sale[Qty]   = 1
Sale[Price] = $8
Sale[Dish]  = "Burger"
Dish[ID]    = "Burger"      (R)
Dish[Type]  = "H. Special"  (R)
Dish[Cost]  = $5            (R)
```

"Numbers and Text for Excel style use"

Filter Context:

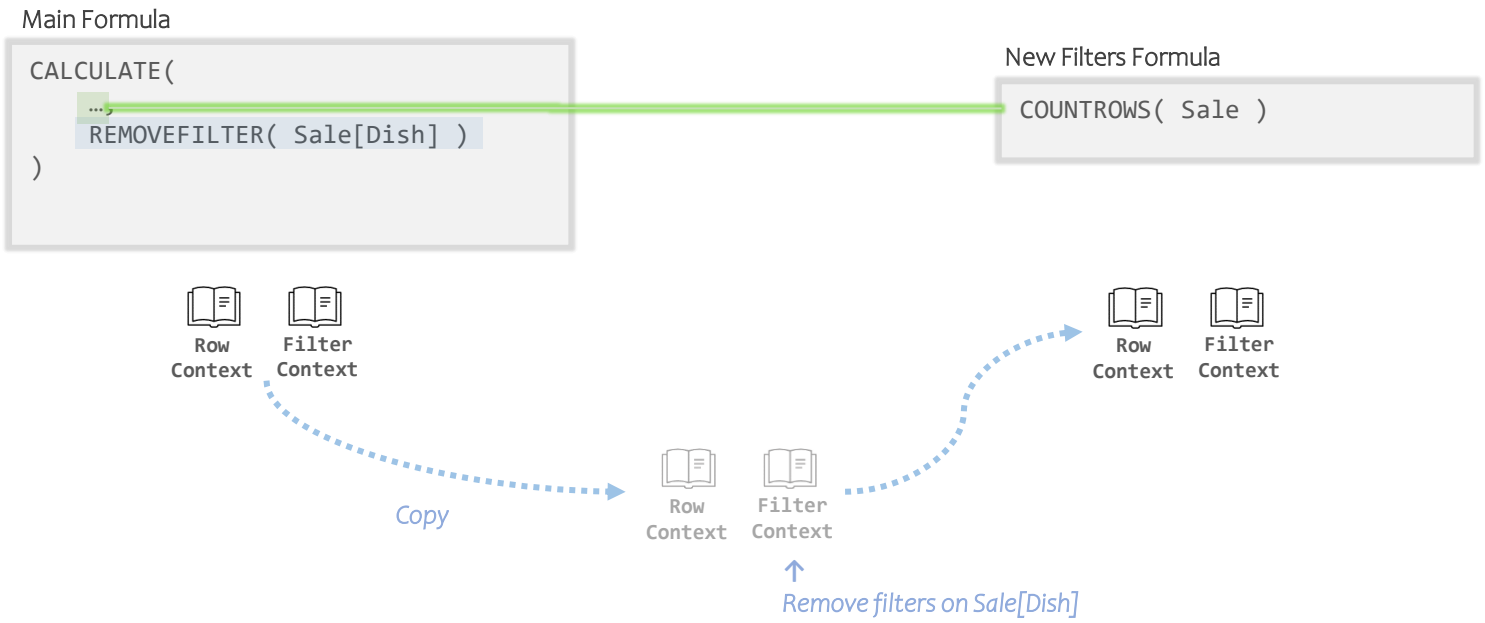
"Filters for Auto-Filtering"

Evaluation Context: A Simple Pair of Lists

OK, fine. But how does stuff get put into the lists?

Lemme show you…

How CALCULATE Changes Filter Context



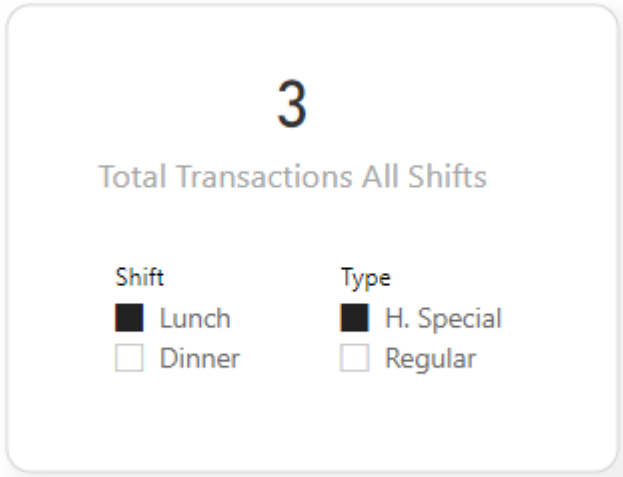
CALCULATE makes a copy of the Evaluation Context it was called in.

It makes whatever change was requested in argument 2 of CALCULATE.

Then it hands the new Evaluation Context to the sub-formula to run in.

How CALCULATE Changes Filter Context

Let's make this more concrete.




```
1  Total Transactions All Shifts =
2  CALCULATE(
3      COUNTROWS( Sale ),
4      REMOVEFILTERS( Sale[Shift] )
5  )
```

How CALCULATE Changes Filter Context

Total Transactions All Shifts

```
CALCULATE(
    COUNTROWS( Sale ),
    REMOVEFILTERS( Sale[Shift] )
)
```

How CALCULATE Changes Filter Context

Sale[Shift]	Sale[Qty]	Sale[Price]	Sale[Dish]	Dish[ID]	Dish[Type]	Dish[Cost]
Lunch	1	$8	Burger	Burger	H. Special	$5
Lunch	3	$8	Burger	Burger	H. Special	$5
Dinner	2	$8	Burger	Burger	H. Special	$5

COUNTROWS returns 3

Total Transactions All Shifts

```
CALCULATE(
    ...,
    REMOVEFILTERS( Sale[Shift] )
)
```

New Filters Formula

```
COUNTROWS( Sale )
```

3

Total Transactions All Shifts

Evaluation Context for Measure

Row Context:

(Empty)

Filter Context:

Sale[Shift]
Lunch

Dish[Type]
H. Special

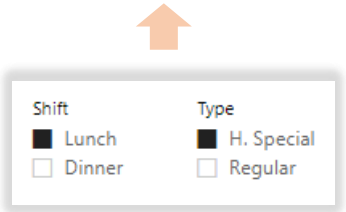
Evaluation Context for New Filters Formula

Row Context:

(Empty)

Filter Context:

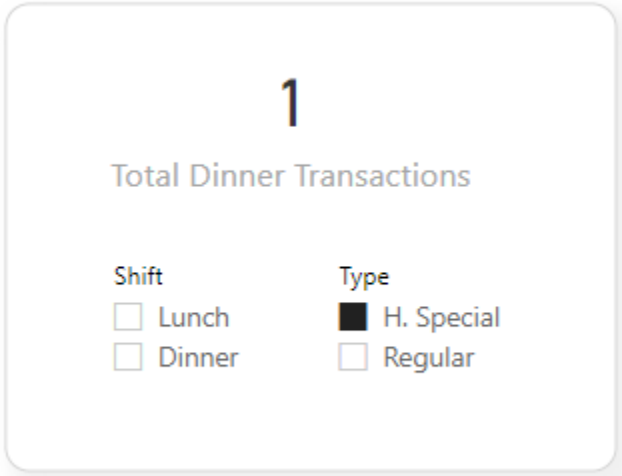
Dish[Type]
H. Special



How About Adding A Filter?

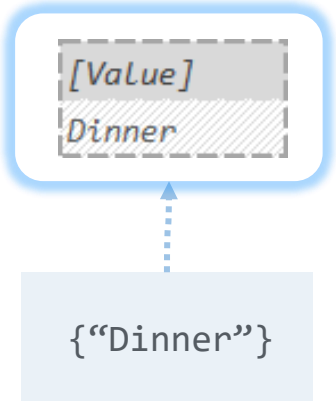
Yeah, how about adding filters? How do you do that?

Lemme show you...

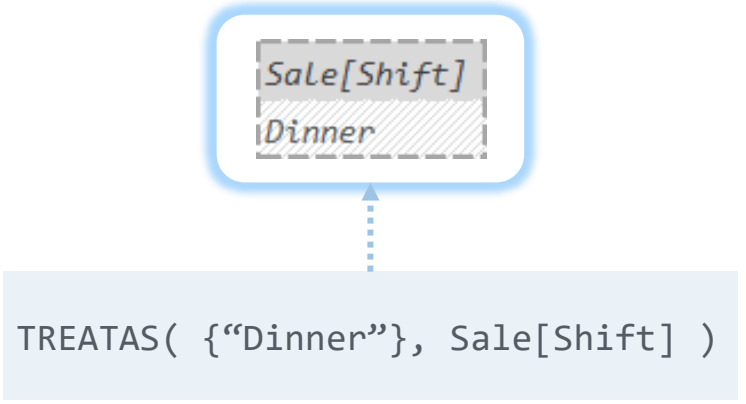



```
1  Total Dinner Transactions =
2  CALCULATE(
3      COUNTROWS( Sale ),
4      TREATAS( {"Dinner"}, Sale[Shift] )
5  )
```

A Brief Aside



We can use curly braces to quickly make a Temp Table…



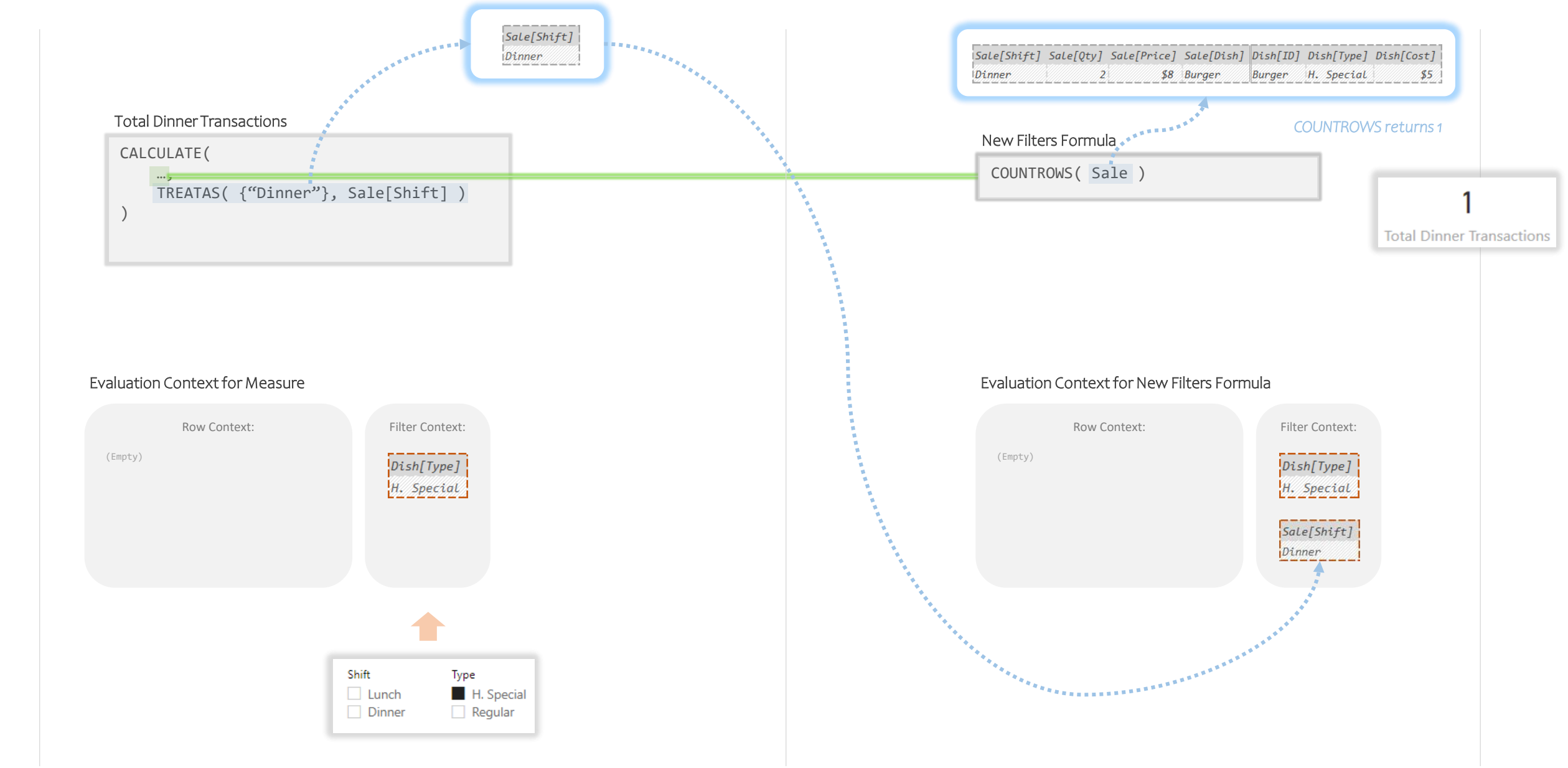
…then use TREATAS to “rename” its column so it will filter the column we want.

How CALCULATE Changes Filter Context

Total Dinner Transactions

```
CALCULATE(
    COUNTROWS( Sale ),
    TREATAS( {“Dinner”}, Sale[Shift] )
)
```

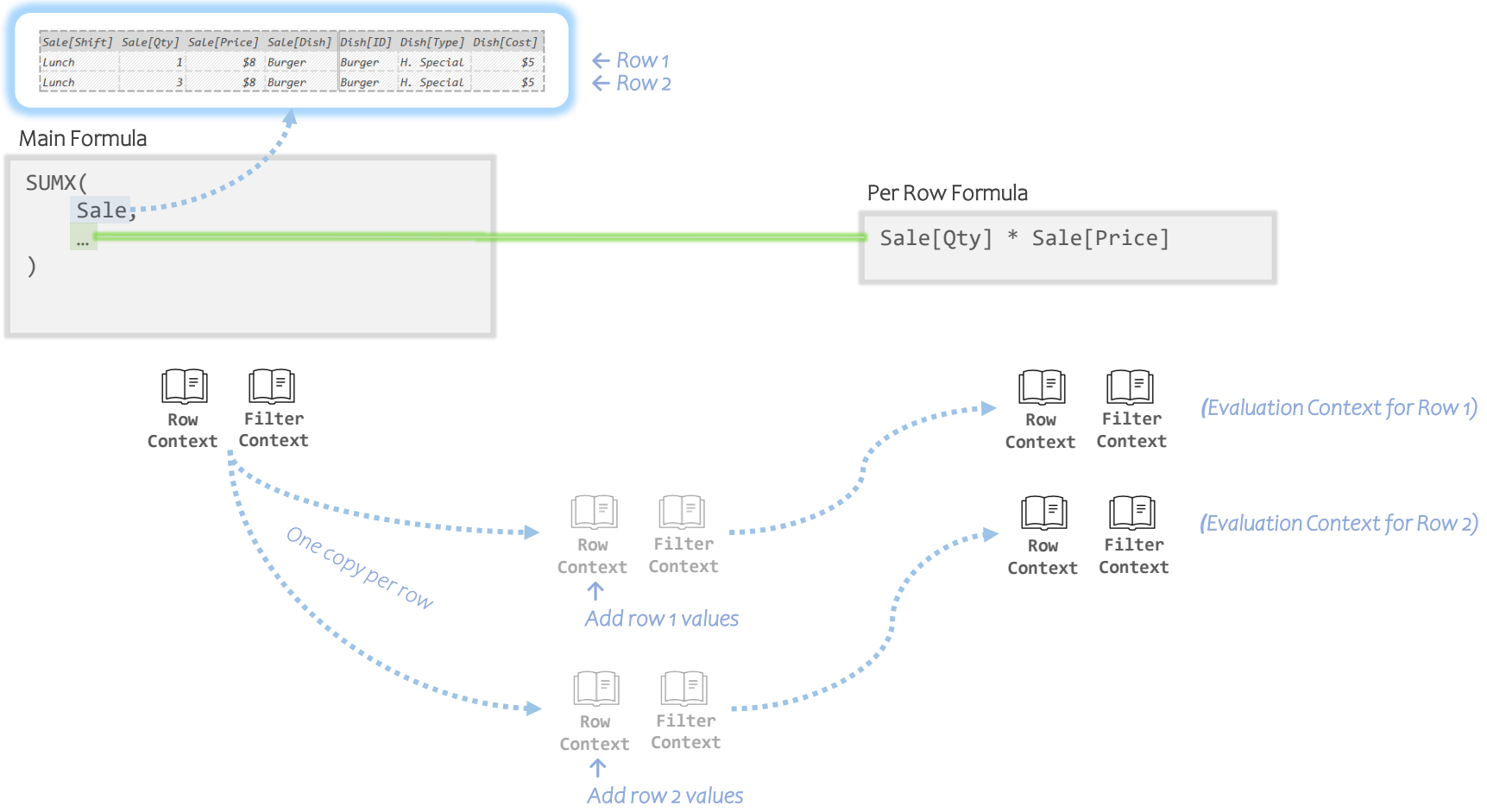
How CALCULATE Changes Filter Context



Iterators

Now let's look at Iterators…

How Iterators Set Row Context



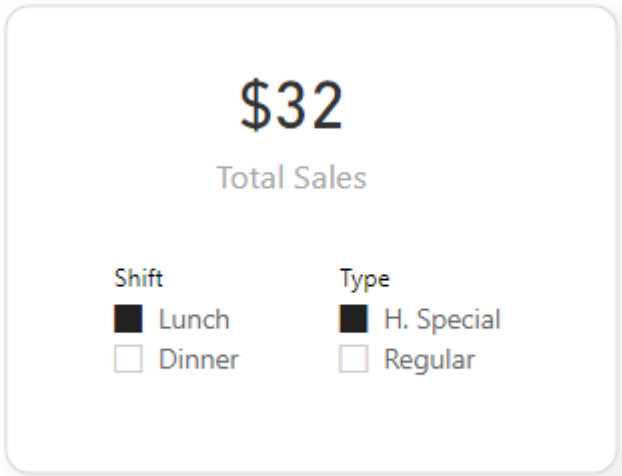
SUMX makes several copies of the Evaluation Context it was called in.

Each row's values get written into one of the Row Contexts.

Then it hands the set of new Evaluation Contexts to the sub-formula to run in.

How SUMX Changes Row Context

Let's make this more concrete.




```
1  Total Sales =
2  SUMX(
3      Sale,
4      Sale[Qty] * Sale[Price]
5  )
```

How SUMX Changes Row Context

Total Sales

```
SUMX(
    Sale,
    Sale[Qty] * Sale[Price]
)
```

How SUMX Changes Row Context

Sale[Shift]	Sale[Qty]	Sale[Price]	Sale[Dish]	Dish[ID]	Dish[Type]	Dish[Cost]
Lunch	1	$8	Burger	Burger	H. Special	$5
Lunch	3	$8	Burger	Burger	H. Special	$5

Total Sales

```
SUMX(
    Sale,
    ...
)
```

1 * $8 → $8 Row 1

3 * $8 → $24 Row 2

$8 + $24 = $32

Per Row Formula

```
Sale[Qty] * Sale[Price]
```

Evaluation Context for Measure

Row Context:

(Empty)

Filter Context:

Sale[Shift]
Lunch

Dish[Type]
H. Special

Shift: Lunch, Dinner

Type: H. Special, Regular

Evaluation Context for Per Row Formula (Row 1)

Row Context:

```
Sale[Shift] = "Lunch"
Sale[Qty]   = 1
Sale[Price] = $8
Sale[Dish]  = "Burger"
Dish[ID]    = "Burger"      (R)
Dish[Type]  = "H. Special"  (R)
Dish[Cost]  = $5            (R)
```

Filter Context:

Sale[Shift]
Lunch

Dish[Type]
H. Special

Evaluation Context for Per Row Formula (Row 2)

Row Context:

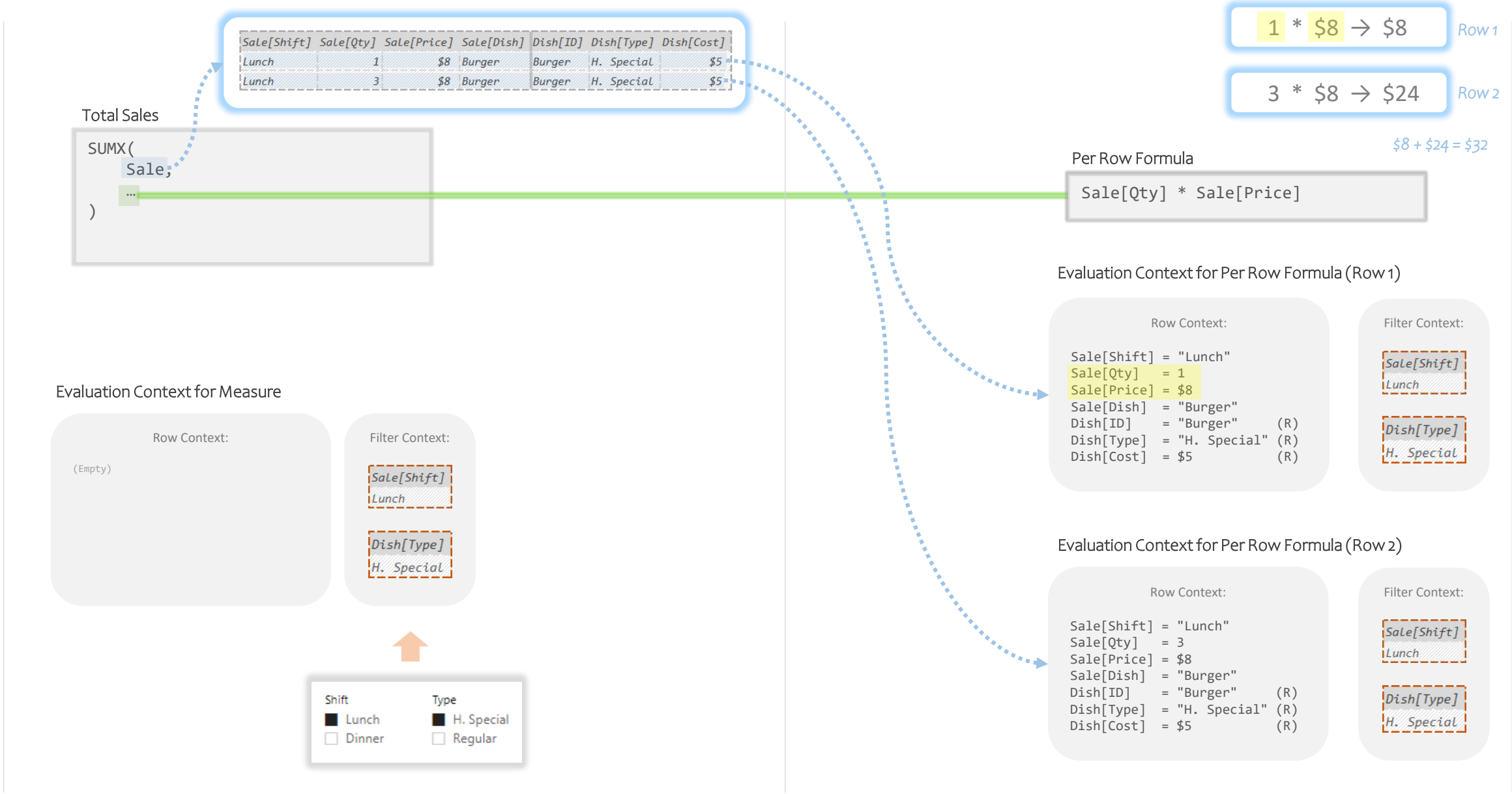
```
Sale[Shift] = "Lunch"
Sale[Qty]   = 3
Sale[Price] = $8
Sale[Dish]  = "Burger"
Dish[ID]    = "Burger"      (R)
Dish[Type]  = "H. Special"  (R)
Dish[Cost]  = $5            (R)
```

Filter Context:

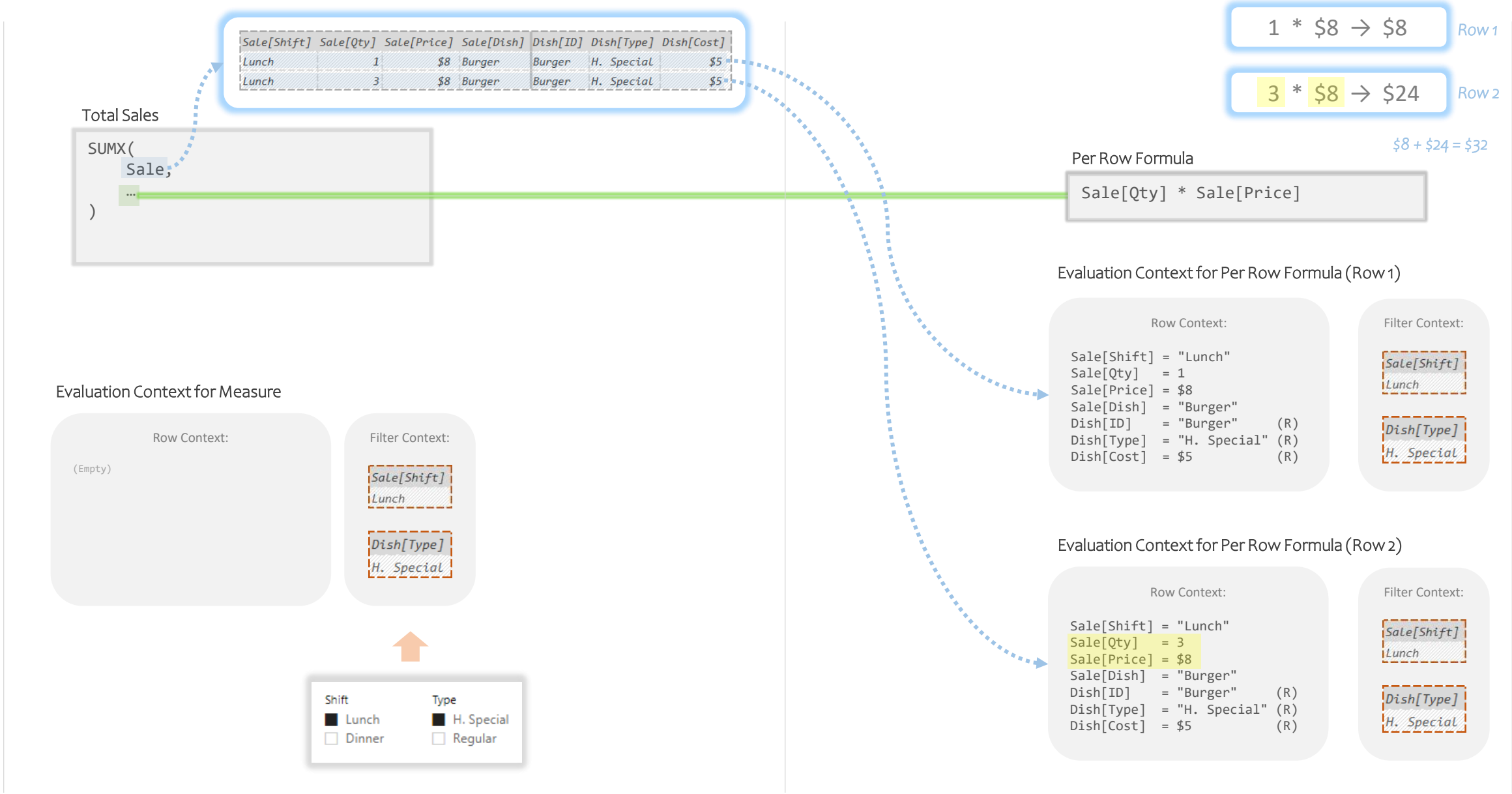
Sale[Shift]
Lunch

Dish[Type]
H. Special

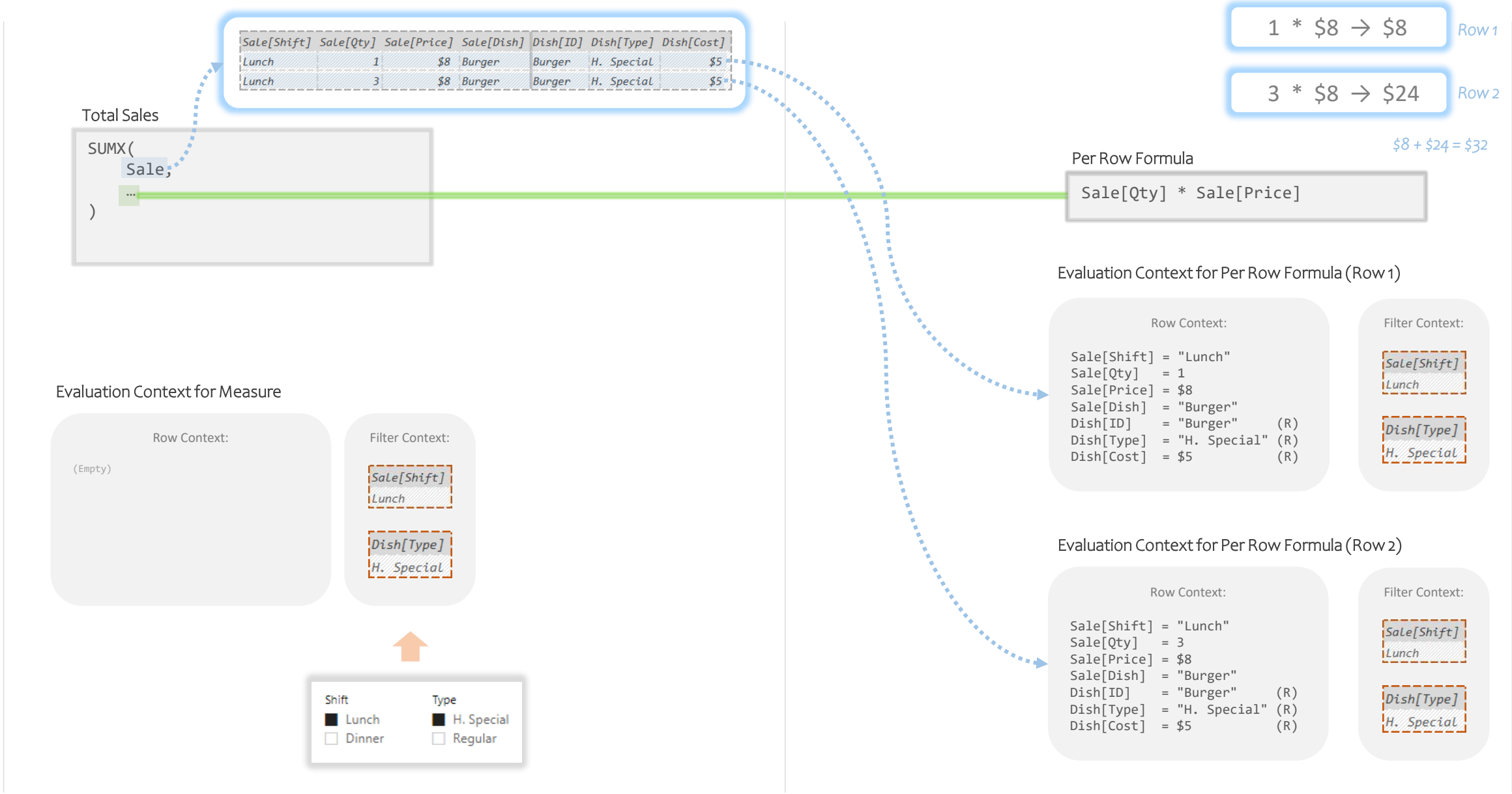
How SUMX Changes Row Context



How SUMX Changes Row Context

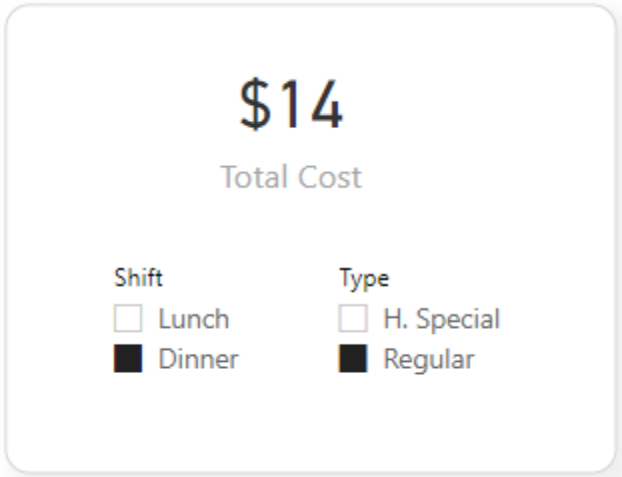


How SUMX Changes Row Context



How SUMX Changes Row Context (Again)

Just One More Example (If There's Time)



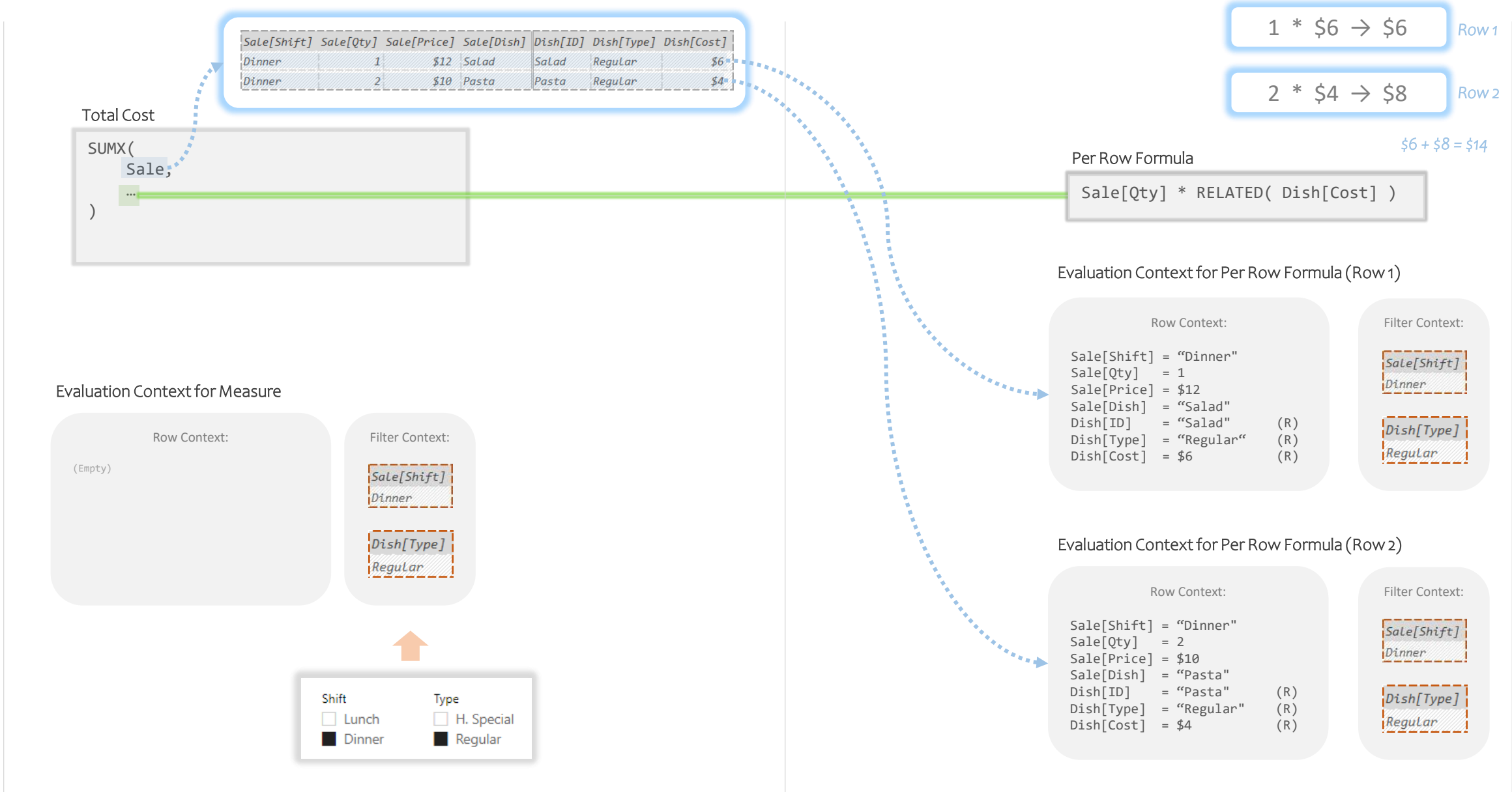

```
1  Total Cost =
2  SUMX(
3      Sale,
4      Sale[Qty] * RELATED( Dish[Cost] )
5  )
```

How SUMX Changes Row Context (Again)

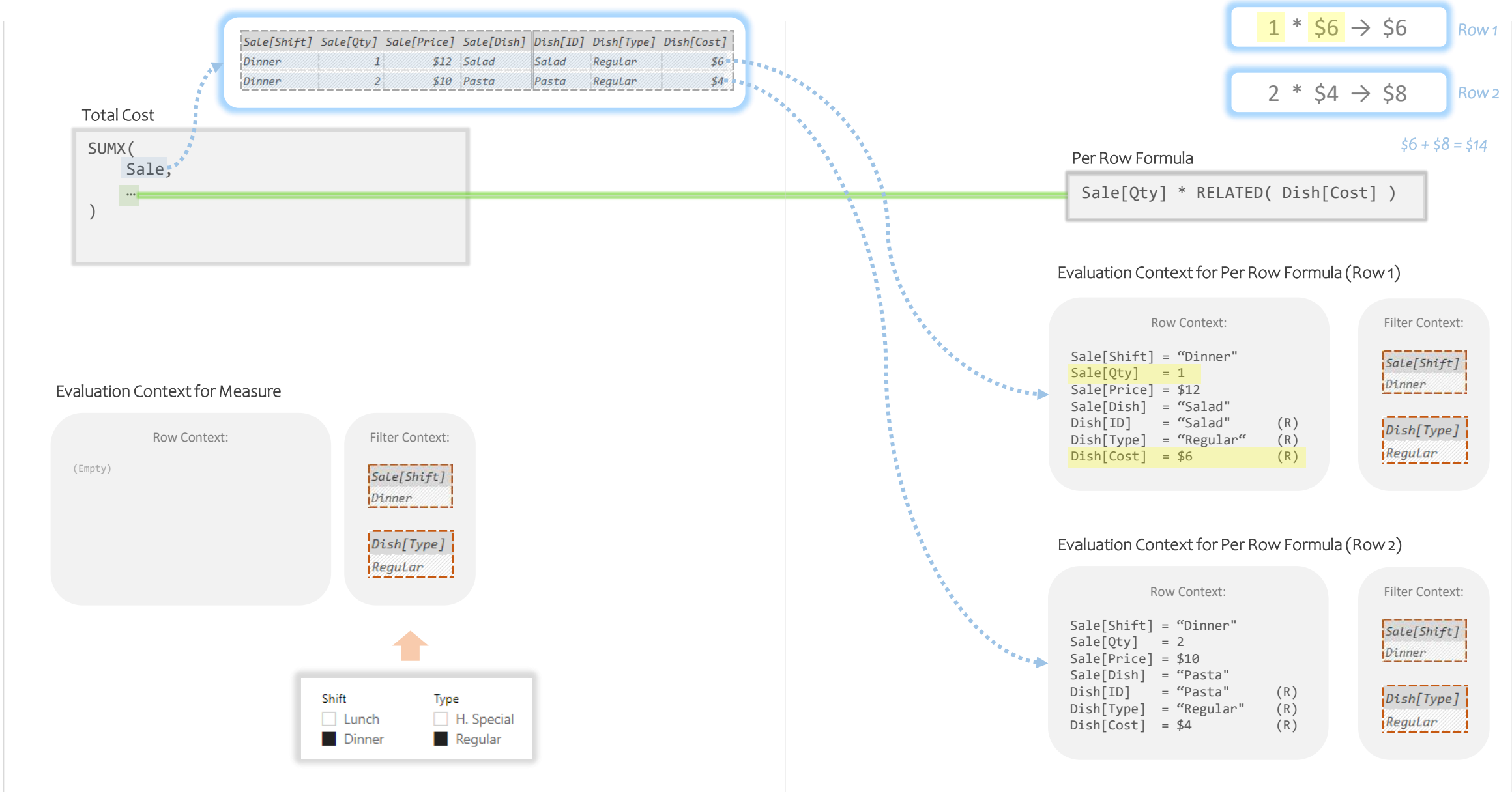
Total Cost

```
SUMX(
    Sale,
    Sale[Qty] * RELATED( Dish[Cost] )
)
```

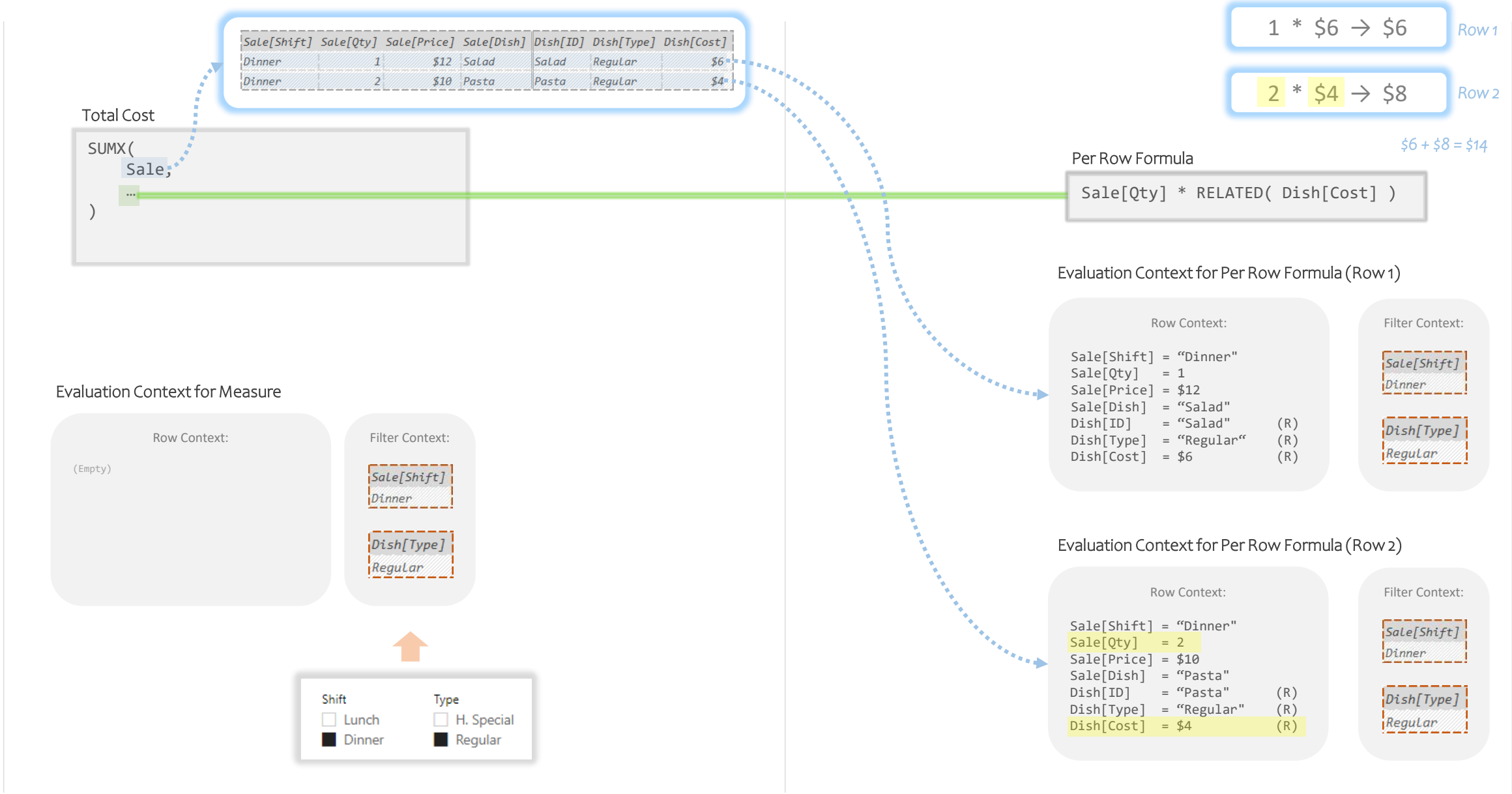
How SUMX Changes Row Context (Again)



How SUMX Changes Row Context (Again)



How SUMX Changes Row Context (Again)



How SUMX Changes Row Context (Again)

Sale[Shift]	Sale[Qty]	Sale[Price]	Sale[Dish]	Dish[ID]	Dish[Type]	Dish[Cost]
Dinner	1	$12	Salad	Salad	Regular	$6
Dinner	2	$10	Pasta	Pasta	Regular	$4

Total Cost

```
SUMX(
    Sale,
    …
)
```

Per Row Formula

```
Sale[Qty] * RELATED( Dish[Cost] )
```

1 * $6 → $6 Row 1

2 * $4 → $8 Row 2

$6 + $8 = $14

$14
Total Cost

Evaluation Context for Measure

Row Context:

(Empty)

Filter Context:

Sale[Shift]
Dinner

Dish[Type]
Regular

Shift: Lunch, Dinner

Type: H. Special, Regular

Evaluation Context for Per Row Formula (Row 1)

Row Context:

```
Sale[Shift] = "Dinner"
Sale[Qty]   = 1
Sale[Price] = $12
Sale[Dish]  = "Salad"
Dish[ID]    = "Salad"      (R)
Dish[Type]  = "Regular"    (R)
Dish[Cost]  = $6           (R)
```

Filter Context:

Sale[Shift]
Dinner

Dish[Type]
Regular

Evaluation Context for Per Row Formula (Row 2)

Row Context:

```
Sale[Shift] = "Dinner"
Sale[Qty]   = 2
Sale[Price] = $10
Sale[Dish]  = "Pasta"
Dish[ID]    = "Pasta"      (R)
Dish[Type]  = "Regular"    (R)
Dish[Cost]  = $4           (R)
```

Filter Context:

Sale[Shift]
Dinner

Dish[Type]
Regular

Anything Else?

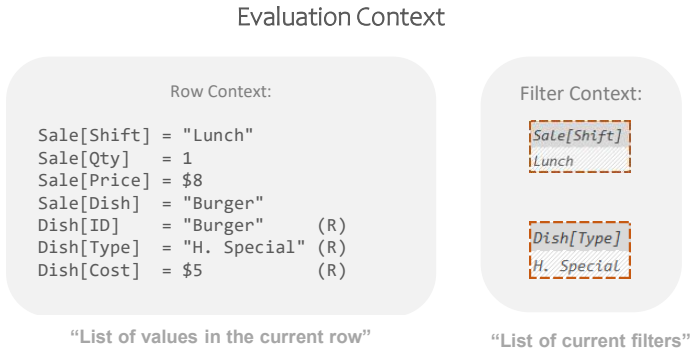
Is there anything else to know about Evaluation Context?

Sure! But not in this presentation, we've already covered plenty.

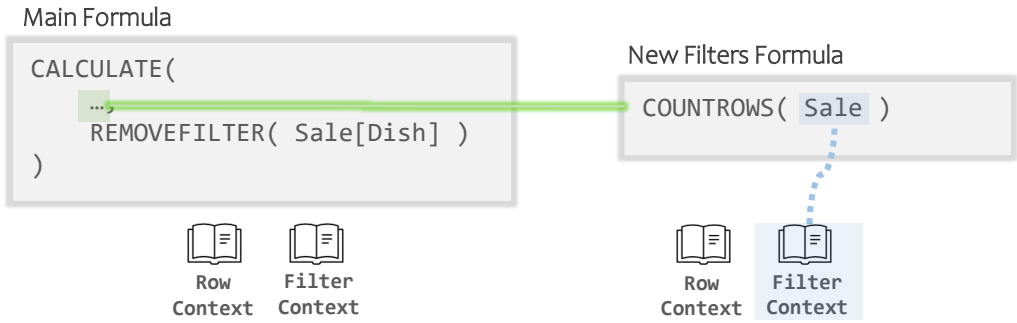
That's it for the basics. One last "Take Away" slide…

Take Aways: Evaluation Context

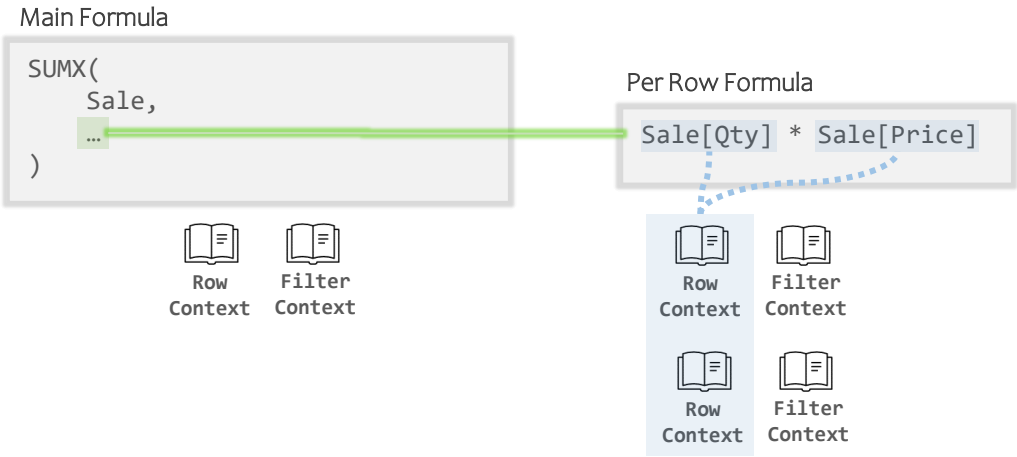
Every Sub-Formula runs with at least one **Evaluation Context** in place. These are pairs of lists that holds useful information for the sub-formula.



CALCULATE makes changes to the **Filter Context**; this affects things like table references, which need to know which filters to apply.



Iterators create multiple Evaluation Contexts, each with the values of one row added to the **Row Context**. This is where the Per Row Formula goes to look each row's values to use like basic number/text/etc.



All Done!

And that's it. Thanks for joining me!

You Just Finished Watching:

DAX Physics 101:
Demystifying DAX Evaluation Context

Any Questions?

To learn more please visit: **briangrantbi.blog**
(Start from the beginning, posts are sequential)

Featuring:
Mr. Brian Grant
Sr. Analytics Consultant
Skypoint

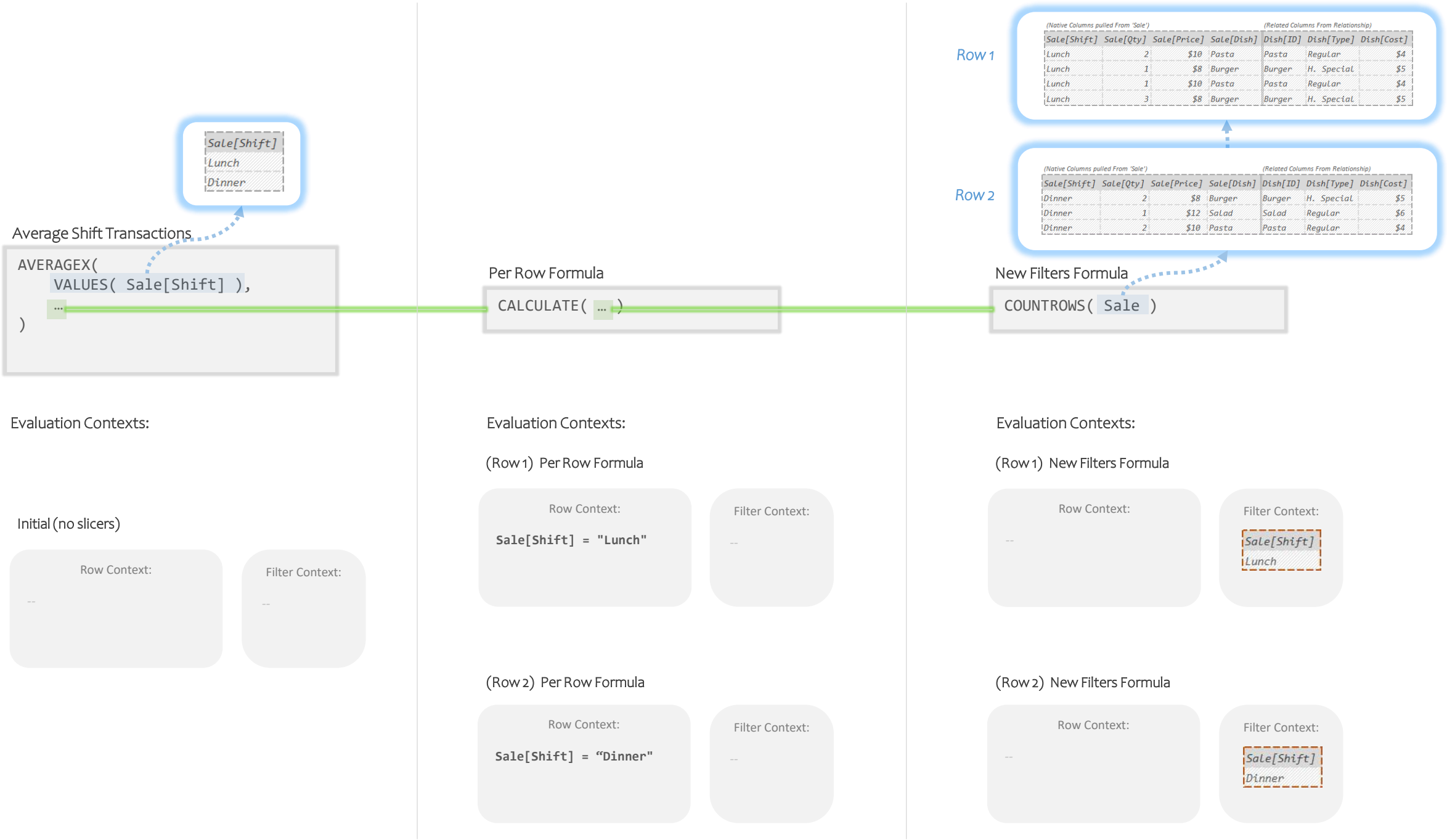
skypoint

Coming Attractions!

(If we somehow have time for them)

Context Transition

CALCULATE in disguise


Sale[Shift]
Lunch
Dinner
Average Shift Transactions
AVERAGEX(
VALUES(Sale[Shift]),
…
)
Per Row Formula
CALCULATE(…)
New Filters Formula
COUNTROWS(Sale)
(Native Columns pulled From 'Sale')
(Related Columns From Relationship)
Sale[Shift] Sale[Qty] Sale[Price] Sale[Dish] Dish[ID] Dish[Type] Dish[Cost]
Lunch 2 $10 Pasta Pasta Regular $4
Lunch 1 $8 Burger Burger H. Special $5
Lunch 1 $10 Pasta Pasta Regular $4
Lunch 3 $8 Burger Burger H. Special $5
Row 1
(Native Columns pulled From 'Sale')
(Related Columns From Relationship)
Sale[Shift] Sale[Qty] Sale[Price] Sale[Dish] Dish[ID] Dish[Type] Dish[Cost]
Dinner 2 $8 Burger Burger H. Special $5
Dinner 1 $12 Salad Salad Regular $6
Dinner 2 $10 Pasta Pasta Regular $4
Row 2
Evaluation Contexts:
Initial (no slicers)
Row Context:
--
Filter Context:
--
Evaluation Contexts:
(Row 1) Per Row Formula
Row Context:
Sale[Shift] = "Lunch"
Filter Context:
--
(Row 2) Per Row Formula
Row Context:
Sale[Shift] = "Dinner"
Filter Context:
--
Evaluation Contexts:
(Row 1) New Filters Formula
Row Context:
--
Filter Context:
Sale[Shift]
Lunch
(Row 2) New Filters Formula
Row Context:
--
Filter Context:
Sale[Shift]
Dinner

Context Transition

Per Row Formula

```
CALCULATE(…)
```

New Filters Formula

```
COUNTROWS( Sale )
```

Evaluation Context for Per Row Formula (Row 1)

Row Context:

```
Sale[Shift] = "Lunch"
Sale[Qty]   = 1
Sale[Price] = $8
Sale[Dish]  = "Burger"
Dish[ID]    = "Burger"      (R)
Dish[Type]  = "H. Special"  (R)
Dish[Cost]  = $5            (R)
```

Filter Context:

(empty)

Evaluation Context for New Filters Formula (Row 1)

Row Context:

(empty)

Filter Context:

Sale[Shift]	Sale[Qty]	Sale[Price]	Sale[Dish]	Dish[ID]	Dish[Type]	Dish[Cost]
Dinner	1	$12	Salad	Salad	Regular	$6

Evaluation Context for Per Row Formula (Row 2)

Row Context:

```
Sale[Shift] = "Lunch"
Sale[Qty]   = 3
Sale[Price] = $8
Sale[Dish]  = "Burger"
Dish[ID]    = "Burger"      (R)
Dish[Type]  = "H. Special"  (R)
Dish[Cost]  = $5            (R)
```

Filter Context:

(empty)

Evaluation Context for New Filters Formula (Row 2)

Row Context:

(empty)

Filter Context:

Sale[Shift]	Sale[Qty]	Sale[Price]	Sale[Dish]	Dish[ID]	Dish[Type]	Dish[Cost]
Dinner	2	$10	Pasta	Pasta	Regular	$4

Measures

CALCULATE in disguise

Sale[Shift]
Lunch
Dinner

Average Shift Transactions

```
AVERAGEX(
    VALUES( Sale[Shift] ),
    …
)
```

Per Row Formula

```
[Total Transactions]
```

New Filters Formula

```
COUNTROWS( Sale )
```

Row 1

(Native Columns pulled From 'Sale')				(Related Columns From Relationship)		
Sale[Shift]	Sale[Qty]	Sale[Price]	Sale[Dish]	Dish[ID]	Dish[Type]	Dish[Cost]
Lunch	2	$10	Pasta	Pasta	Regular	$4
Lunch	1	$8	Burger	Burger	H. Special	$5
Lunch	1	$10	Pasta	Pasta	Regular	$4
Lunch	3	$8	Burger	Burger	H. Special	$5

Row 2

(Native Columns pulled From 'Sale')				(Related Columns From Relationship)		
Sale[Shift]	Sale[Qty]	Sale[Price]	Sale[Dish]	Dish[ID]	Dish[Type]	Dish[Cost]
Dinner	2	$8	Burger	Burger	H. Special	$5
Dinner	1	$12	Salad	Salad	Regular	$6
Dinner	2	$10	Pasta	Pasta	Regular	$4

Evaluation Contexts:

Initial (no slicers)

Row Context:

--

Filter Context:

--

Evaluation Contexts:

(Row 1) Per Row Formula

Row Context:

`Sale[Shift] = "Lunch"`

Filter Context:

--

(Row 2) Per Row Formula

Row Context:

`Sale[Shift] = “Dinner"`

Filter Context:

--

Evaluation Contexts:

(Row 1) New Filters Formula

Row Context:

--

Filter Context:

Sale[Shift]
Lunch

(Row 2) New Filters Formula

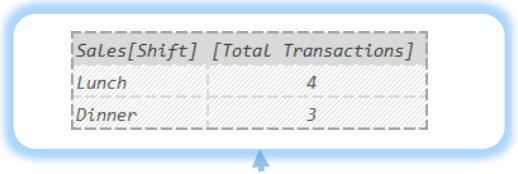
Row Context:

--

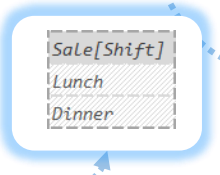
Filter Context:

Sale[Shift]
Dinner

Decomposing a Summary Query



Sales[Shift]	[Total Transactions]
Lunch	4
Dinner	3



Sale[Shift]
Lunch
Dinner

Summary Query

```
ADDCOLUMNS(
    VALUES( Sale[Shift] ),
    "Total Transactions", …
)
```

Per Row Formula

```
[Total Transactions]
```

New Filters Formula

```
COUNTROWS( Sale )
```

Row 1

(Native Columns pulled From 'Sale') | (Related Columns From Relationship)

Sale[Shift]	Sale[Qty]	Sale[Price]	Sale[Dish]	Dish[ID]	Dish[Type]	Dish[Cost]
Lunch	2	$10	Pasta	Pasta	Regular	$4
Lunch	1	$8	Burger	Burger	H. Special	$5
Lunch	1	$10	Pasta	Pasta	Regular	$4
Lunch	3	$8	Burger	Burger	H. Special	$5

→ 4

Row 2

(Native Columns pulled From 'Sale') | (Related Columns From Relationship)

Sale[Shift]	Sale[Qty]	Sale[Price]	Sale[Dish]	Dish[ID]	Dish[Type]	Dish[Cost]
Dinner	2	$8	Burger	Burger	H. Special	$5
Dinner	1	$12	Salad	Salad	Regular	$6
Dinner	2	$10	Pasta	Pasta	Regular	$4

→ 3

Evaluation Contexts:

Initial (no slicers)

Row Context:

--

Filter Context:

--

Evaluation Contexts:

(Row 1) Per Row Formula

Row Context:

`Sale[Shift] = "Lunch"`

Filter Context:

--

(Row 2) Per Row Formula

Row Context:

`Sale[Shift] = "Dinner"`

Filter Context:

--

Evaluation Contexts:

(Row 1) New Filters Formula

Row Context:

--

Filter Context:

Sale[Shift]
Lunch

(Row 2) New Filters Formula

Row Context:

--

Filter Context:

Sale[Shift]
Dinner

The Hidden Contexts of DAX

4 Lists

Evaluation Context

Row Context:

```
Sale[Shift] = "Lunch"
Sale[Qty]   = 1
Sale[Price] = $8
Sale[Dish]  = "Burger"
Dish[ID]    = "Burger"      (R)
Dish[Type]  = "H. Special" (R)
Dish[Cost]  = $5            (R)
```

Filter Context:

Sale[Shift]
Lunch

Dish[Type]
H. Special

Relationship Context:
List of Active Relationships (used in "Super Lookup"/"Table Expansion")

Relationship Context:

- Sale[Dish] to Dish[ID] (1-M, B.D.)

Rephrasing Context:

```
(1) SELECTEDMEASURE() + 2
(2) SELECTEDMEASURE() * 10
```

Rephrasing Context:
How should the New Filters Formula be rewritten (rephrased) before running. Populated by Calculation Items.