EXPLORE | DIGITAL SKILLS

DAX Formulas, Columns and Measures

Learning Objectives

After this train, you should be able to:

- Understand the different command functions in Power BI
- Identify different DAX formulas
- Differentiate between calculated columns and measures
- Apply DAX formulas to creating calculated columns and measures

EXPLORE DIGITAL SKILLS

Topics to be covered

The following material will be covered in this train.

1. Overview of commands and queries in Power BI
Understand the purpose of commands in Power BI

2. DAX formula categories
Different types of DAX formula functions

3. Calculated columns and measures
Understand and identify differences between the two

4. Creating measures for IPL data
Create both calculated columns and measures using the IPL data

EXPLORE | DIGITAL SKILLS

Overview of Power BI Commands and Queries

Power BI offers the functionality to employ commands and queries for data analysis

What makes Power BI commands and queries so powerful?

DAX Formulas

DAX formulas are powerful functions used to manipulate, aggregate and calculate metrics for a Power BI dashboard

Calculated Columns

Calculated columns assist to create new columns or fields from columns that have current dependencies or are related in some way; this allows a user to create columns needed to obtain the results for analysis

Measures

Creating measures makes it easier to apply calculations on the dataset; you are able to finetune the input and output data

DAX Formulas

Categories of DAX Formulas

AGGREGATE

MIN
MAX
AVERAGE
SUM
SUMX

COUNTING

DISTINCTCOUNT
COUNT
COUNTA
COUNTROWS
COUNTBLANK

LOGICAL

AND
OR
NOT
IF
IFERROR

TEXT

REPLACE
SEARCH
UPPER/LOWER
FIXED
CONCATENATE

TEMPORAL

DATE
HOUR
WEEKDAY
NOW
EOMONTH

INFORMATION

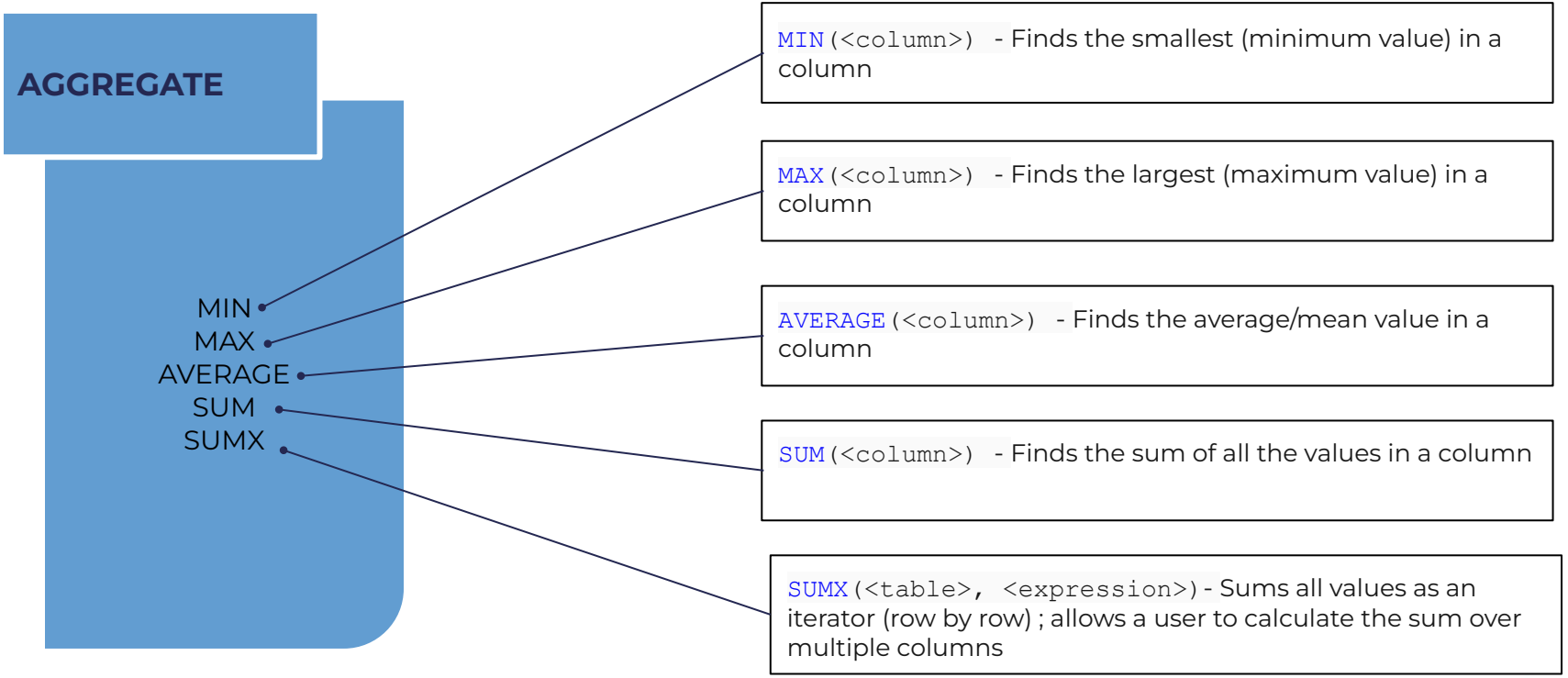
ISBLANK
ISNUMBER
ISTEXT
ISNONTEXT
ISERROR

FILTER

ALL
CALCULATE
CALCULATETABLE
FILTER
LOOKUPVALUE

DAX Formulas

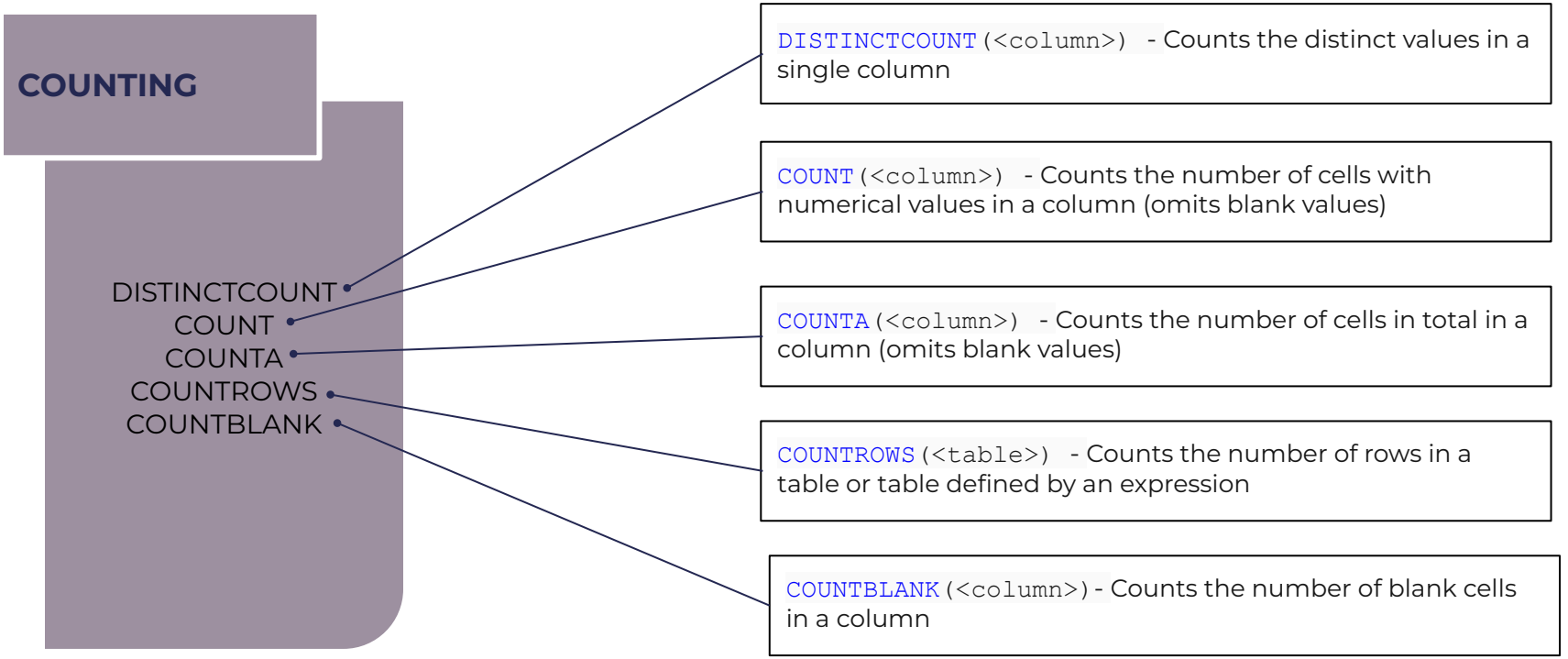
Categories of DAX Formulas - Aggregate



Note: Aggregations require the data type to be a number data type

DAX Formulas

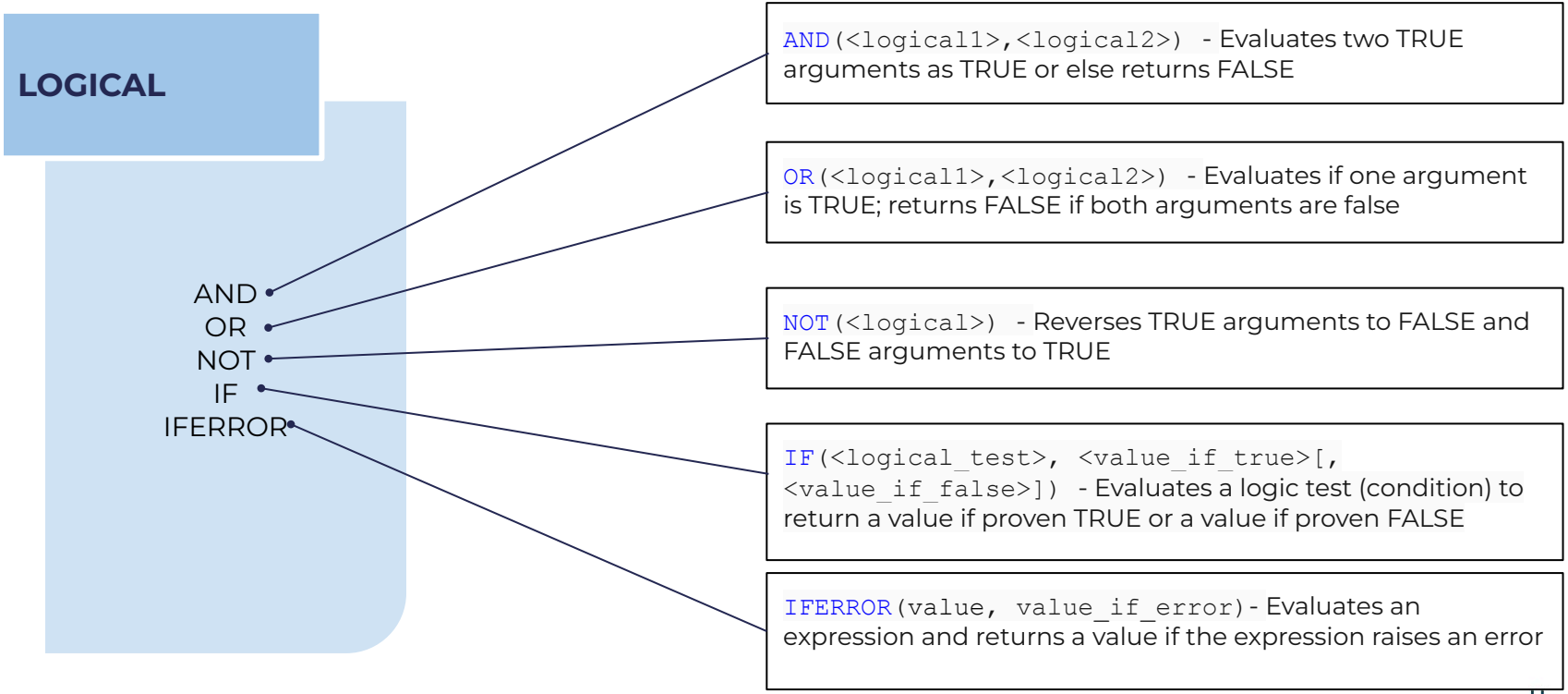
Categories of DAX Formulas - Counting



EXPLORE | DIGITAL SKILLS

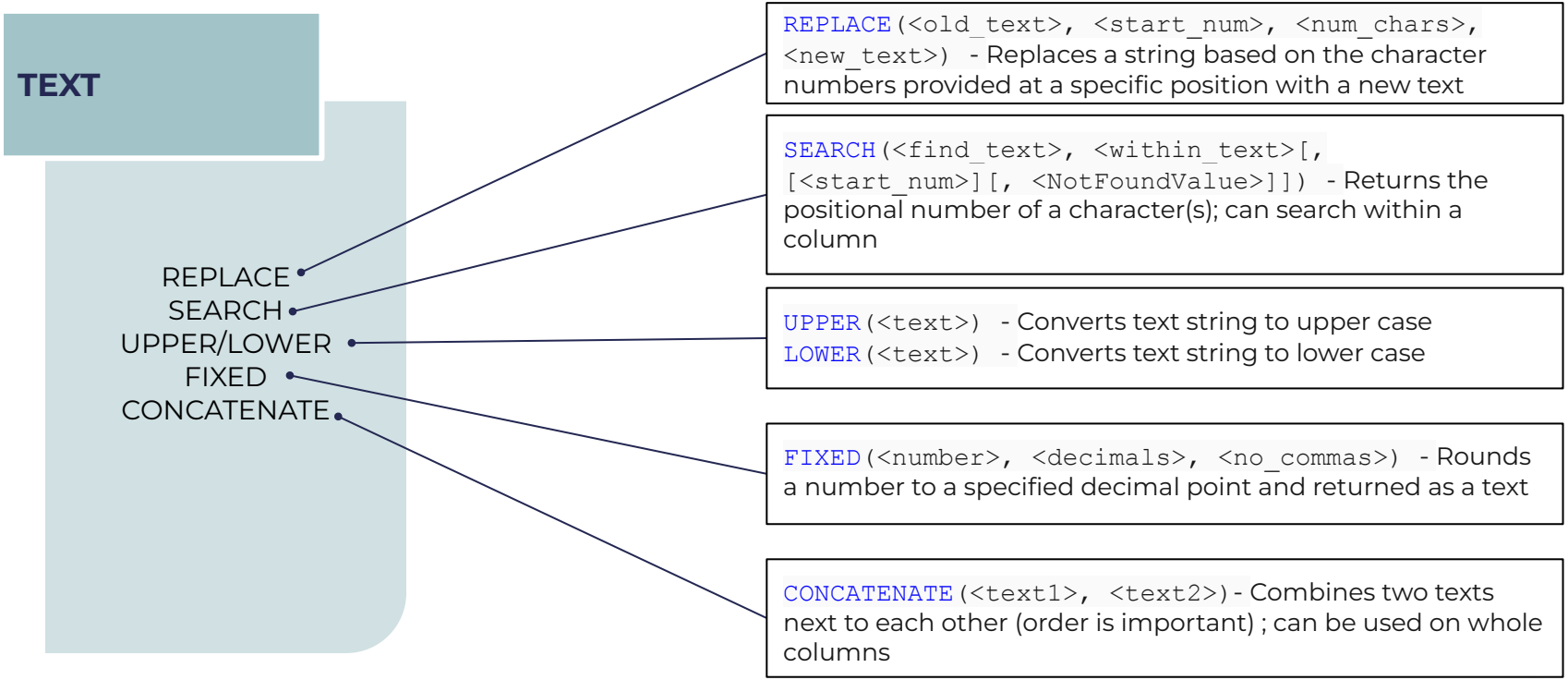
DAX Formulas

Categories of DAX Formulas - Logical



DAX Formulas

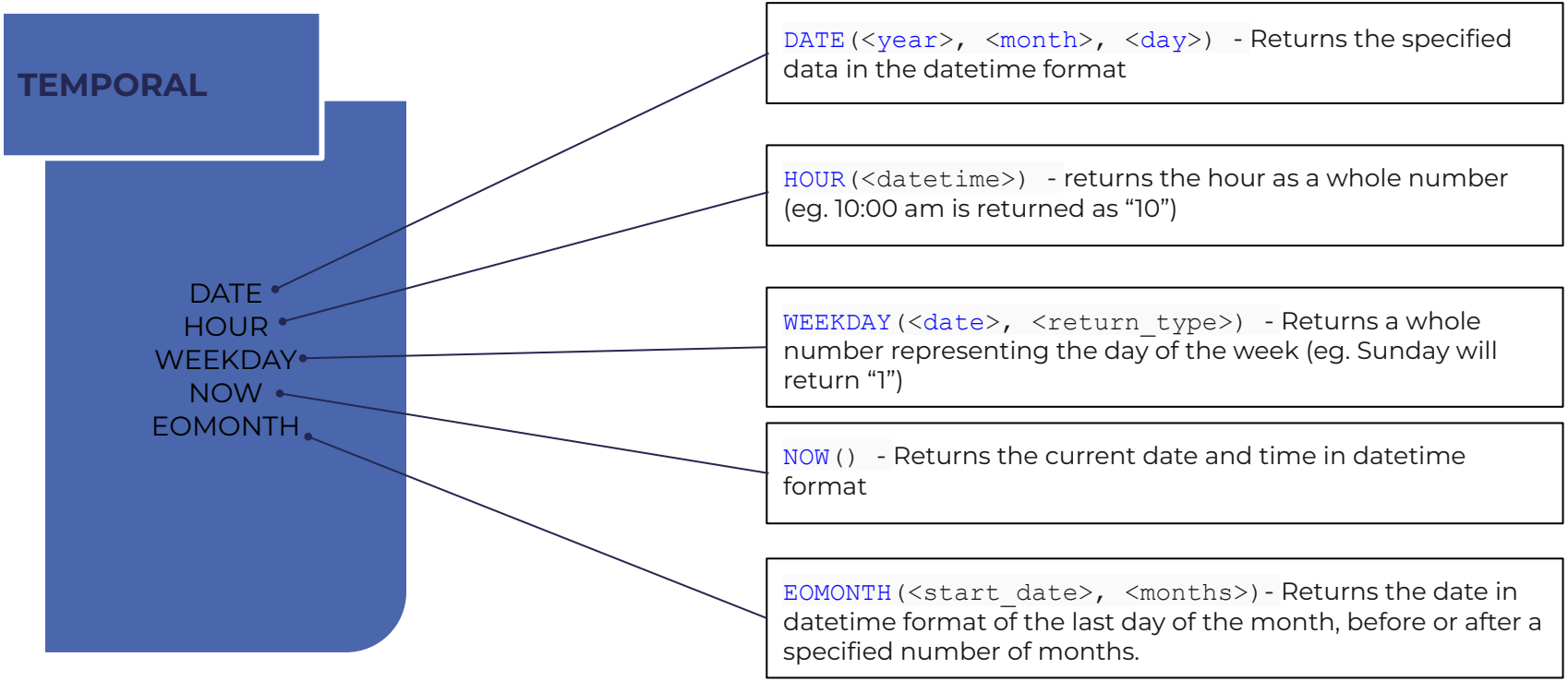
Categories of DAX Formulas - Text



EXPLORE | DIGITAL SKILLS

DAX Formulas

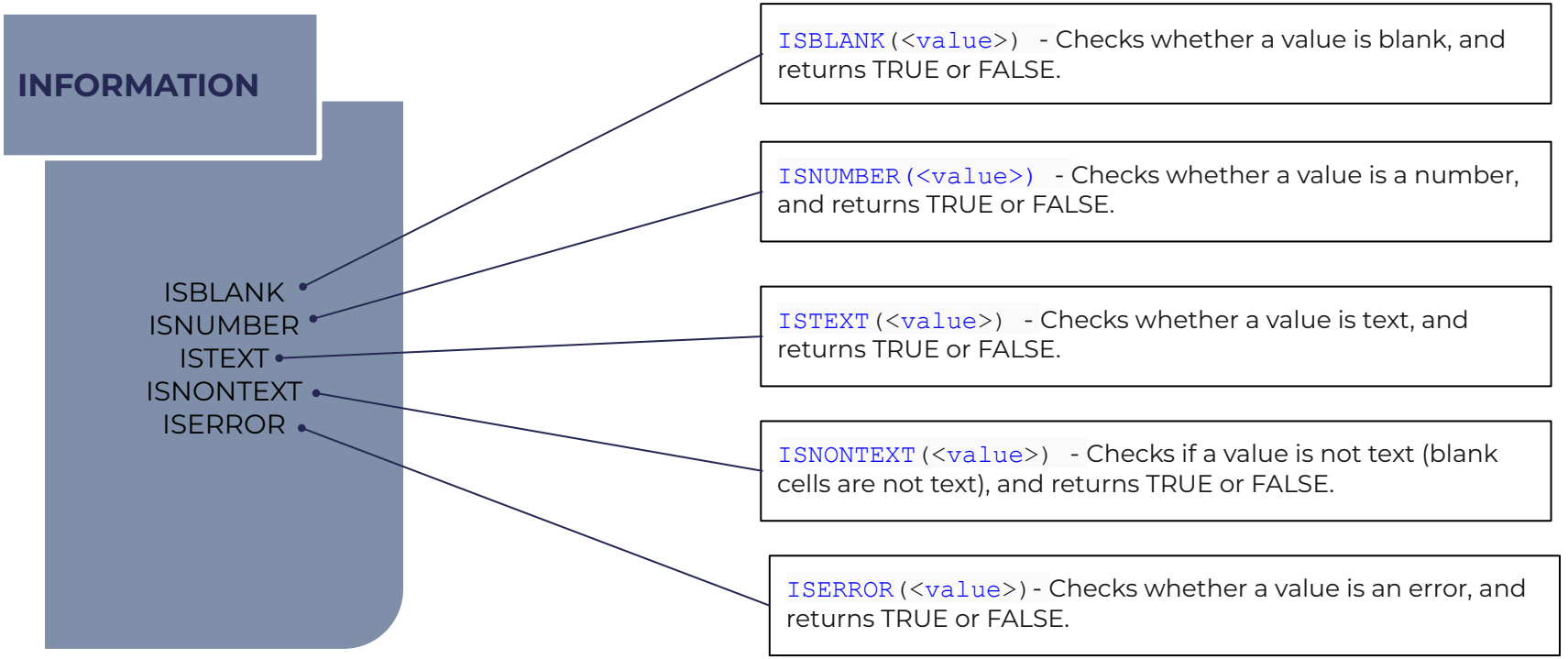
Categories of DAX Formulas - Temporal



EXPLORE | DIGITAL SKILLS

DAX Formulas

Categories of DAX Formulas - Information

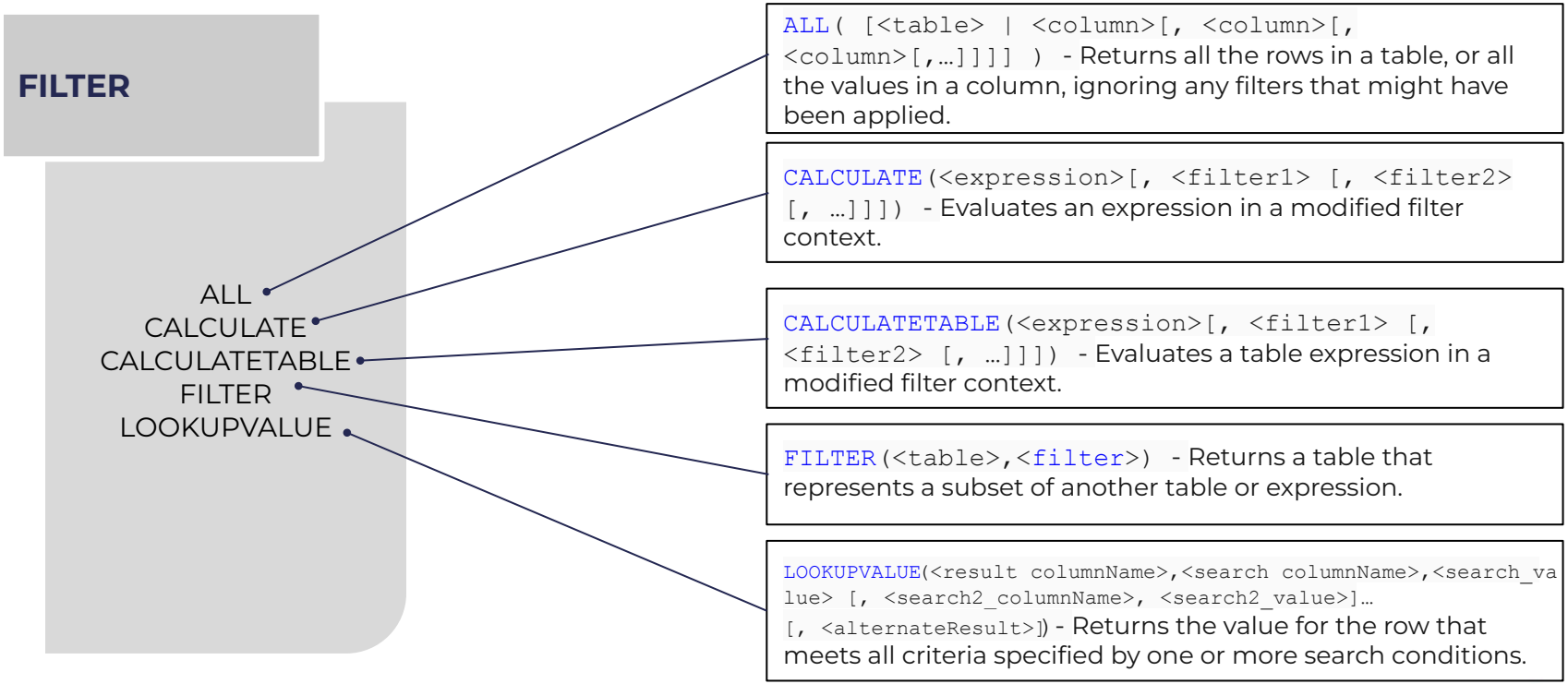


These are usually used in conjunction with conditional formulas

EXPLORE | DIGITAL SKILLS

DAX Formulas

Categories of DAX Formulas - Filter



EXPLORE | DIGITAL SKILLS

Calculated columns vs measures

Here we will cover the differences between **calculated columns** and **measures** and when to use which.

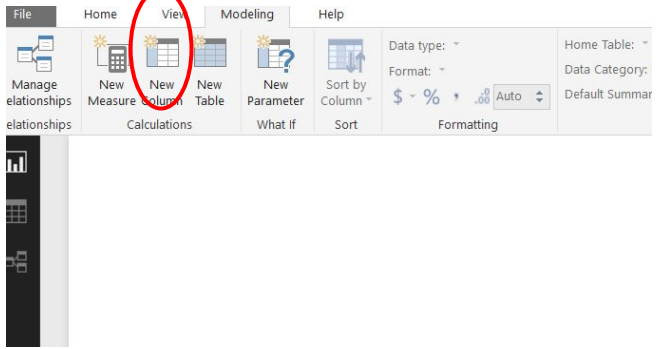
Calculated column	Measure
A calculated column is a column like any other, created in the table. **Features:** ● Row by row calculation ● Stored in memory **When to use:** If the calculation is performed row-by-row based on a combination of other columns	A measure is usually a calculation that works on an aggregated level basis. **Features:** ● Calculated based on filters ● Not stored in memory or pre-calculated **When to use:** If the calculation is an aggregation or will be affected by filter criteria in the report

Calculated columns vs measures

Here we will show where in Power BI we create columns and measures.

Calculated column

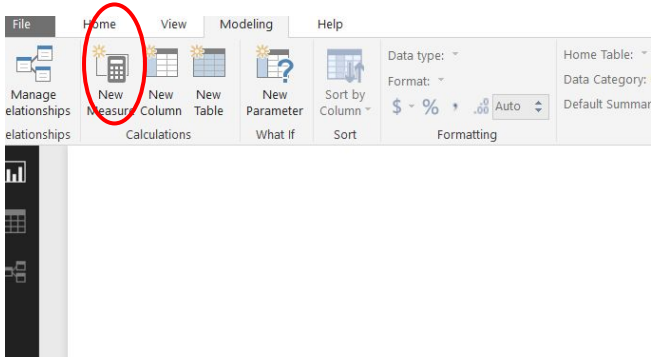
To create a new column, first, select the **Modeling** tab from the top panel. Then select **New Column**.



NB: Make sure you are currently in the correct table when you create the column.

Measure

To create a new measure, first, select the **Modeling** tab from the top panel. Then select **New Measure**.



EXPLORE | DIGITAL SKILLS

Creating calculated columns for IPL data

Here we will create the **RCB_involved** calculated column. The brief is as follows:

1. We need columns that indicate whether a team (let's say Royal Challengers Bangalore, RCB) was **involved** in a match and who their **opposition** was for that match ; because currently, a team can be involved if they are either in the matches[team1] column or the matches[team2] column.

2. We also want to be able to determine whether the match was a **home or away match**.

Let's start by creating a column called "**RCB_involved**" - which indicates whether RCB is involved in the match ("1" if they were, and "0" if they were not). We want to do this to filter out all games that RCB was not involved in.

RCB_involved = *IF*(*OR*(matches[team1] = "Royal Challengers Bangalore", matches[team2] = "Royal Challengers Bangalore"), 1 , 0)

The matches dataset should now look like the snippet on the right.

venue	umpire1	umpire2	RCB_involved
Maharashtra Cricket Association Stadium	A Nand Kishore	S Ravi	0
Saurashtra Cricket Association Stadium	A Nand Kishore	S Ravi	0
Maharashtra Cricket Association Stadium	AY Dandekar	NJ Llong	0
M Chinnaswamy Stadium	AK Chaudhary	C Shamshuddin	1
Eden Gardens	NJ Llong	S Ravi	0
Feroz Shah Kotla	M Erasmus	Nitin Menon	0
Rajiv Gandhi International Stadium, Uppal	KN Ananthapadmanabhan	M Erasmus	0
M Chinnaswamy Stadium	MR Benson	IL Howell	1
Eden Gardens	BF Bowden	AV Jayaprakash	0
Rajiv Gandhi International Stadium, Uppal	BR Doctrove	RB Tiffin	0
MA Chidambaram Stadium, Chepauk	MR Benson	RB Tiffin	0
Dr DY Patil Sports Academy	DJ Harper	RE Koertzen	0
Rajiv Gandhi International Stadium, Uppal	BG Jerling	AM Saheba	0
Kingsmead	BR Doctrove	TH Wijewardene	1

Creating calculated columns for IPL data

Here we will create the **RCB_opposition_team** calculated column. The brief is as follows:

Let's create a column called "**RCB_opposition_team**" - which is the name of the team playing **against** RCB.
We want to do this in order to **analyse** performance against specific teams.

RCB_opposition_team = *IF*(matches[team1] = "Royal Challengers Bangalore", matches[team2], *IF*(matches[team2] = "Royal Challengers Bangalore", matches[team1], "-"))

The matches dataset should now look like the snippet below.

umpire1	umpire2	RCB_involved	RCB_opposition_team
A Nand Kishore	S Ravi	0	-
A Nand Kishore	S Ravi	0	-
AY Dandekar	NJ Llong	0	-
AK Chaudhary	C Shamshuddin	1	Gujarat Lions
NJ Llong	S Ravi	0	-
M Erasmus	Nitin Menon	0	-
KN Ananthapadmanabhan	M Erasmus	0	-
MR Benson	IL Howell	1	Rajasthan Royals
BF Bowden	AV Jayaprakash	0	-
BR Doctrove	RB Tiffin	0	-
MR Benson	RB Tiffin	0	-
DJ Harper	RE Koertzen	0	-

Creating calculated columns for IPL data

Here we will create the **RCB_venue** calculated column. The brief is as follows:

Let's create a column called "**RCB_venue**" - which is either "**Home**" or "**Away**", depending on the stadium name. RCB's home stadium is "M Chinnaswamy Stadium".

We want to do this in order to analyse RCB's match performance **away and at home**.

RCB_venue = *IF*(matches[venue] = "M Chinnaswamy Stadium", "Home", "Away")

The matches dataset should now look like the snippet below.

umpire2	RCB_involved	RCB_opposition_team	RCB_venue
S Ravi	0	-	Away
S Ravi	0	-	Away
NJ Llong	0	-	Away
C Shamshuddin	1	Gujarat Lions	Home
S Ravi	0	-	Away
Nitin Menon	0	-	Away
M Erasmus	0	-	Away
IL Howell	1	Rajasthan Royals	Home
AV Jayaprakash	0	-	Away
RB Tiffin	0	-	Away
RB Tiffin	0	-	Away
RE Koertzen	0	-	Away

Creating measures for IPL data

Here we will create some measures for **team statistics**. The brief is as follows:

We want to have an idea of team performance over time. An **aggregation** is required, therefore we need to create a measure.

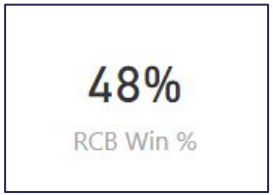
Let's create a measure that calculates the match winning percentage given certain filters.

RCB Win % = *CALCULATE*(*COUNTA*(matches[winner]), matches[winner] = "Royal Challengers Bangalore") / *SUM*(matches[RCB_involved])

In the calculation above we count how many matches were won and divide it by the total number of matches played.

Before visualising, highlight the measure and head to Modeling and select "**%**" in the **Formatting** section.

The measure (with no filters applied) should look as follow on a **card**:



Creating measures for IPL data

Here we will create some measures for **team statistics**. The brief is as follows:

Let's create a measure that calculates RCB's run rate. **Run rate is the number of runs divided the number of overs (six balls in one over)**.

Every row in the deliveries dataset represents a ball, however, it contains some extras (balls that needed to be re-bowled). We only want to count the number of **legitimate** deliveries (exclude the extra balls).

Run Rate =
SUM(deliveries[total_runs]) / *CALCULATE*(*COUNT*(deliveries[ball]), (deliveries[wide_runs] = 0),
(deliveries[noball_runs] = 0)) * 6

In the calculation above, we sum all the runs and divide it by the total number of legitimate balls bowled (i.e. not a wide or no-ball), then multiply it by 6 to get the run rate per over.

The measure (with no filters applied) should look as follow on a **card**:

Creating measures for IPL data

Here we will create some interim measures for **batting statistics.** The brief is as follows:

We want to create some measures that will help us analyse batting performance. Good metrics are:

- Runs
- Strike Rate - runs per 100 balls
- Average - runs per dismissal

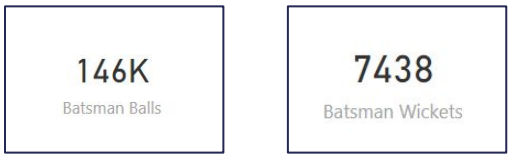
Looking at the metrics above, we need some **interim measures** - balls faced and number of dismissals. We can write the following queries to create these interim measures:

Batsman Balls = *CALCULATE*(*COUNT*(deliveries[ball]), deliveries[wide_runs] = 0)

We don't count the wides for balls faced, since they don't count towards the batsman's balls.

Batsman Wickets =
CALCULATE(*COUNTA*(deliveries[player_dismissed]), deliveries[player_dismissed] <> "-")

The measures (with no filters applied) should look as follow on a **card**:



Creating measures for IPL data

Creating batting statistics - continued

Now that the interim measures are created, we can create the **batting indicators**:

Batsman Runs = *SUM*(deliveries[batsman_runs])

Batsman Strike Rate = [Batsman Runs] / [Batsman Balls] * 100

Batting Average = [Batsman Runs] / [Batsman Wickets]

The measures (with no filters applied) should look as follow on a **card**:

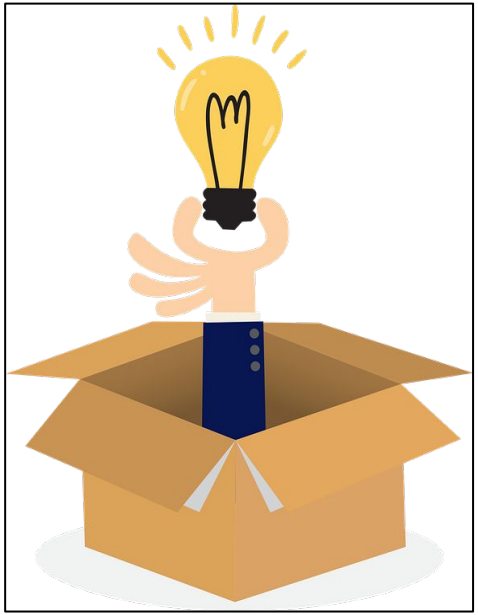
See if you can create the bowling measures on your own!

EXPLORE DIGITAL SKILLS

Conclusion

In this train we have covered the following:

- Understand the purposes of DAX formulas, calculated columns and measures
- Defining and identify different DAX formulas
- Define and differentiate between calculated columns and measures
- Apply DAX formulas and create calculated measures and columns



EXPLORE DIGITAL SKILLS

Appendix

[Official Microsoft Documentation for Power BI]
[A Free PDF to Begin with DAX and Power BI]

Additional Material
[Quickstart with DAX in Power BI]

EXPLORE | DIGITAL SKILLS