Lab Instructions

The instructions below will take you through 3 individual labs resulting in a hexbin plot, pareto chart, trellis chart and if time allows a bonus sentiment analysis wordcloud!

Viel Glück!

Tricia & Paul

PREPARATION TO ACCESS THE DATA

TIME: 5-10 MINUTES

PREREQUISITE:

1. TabPy Server is running
2. Tableau Desktop is running
3. Tableau Analytics Extension is running

---

1. Connect to CSV file

> Instructions: CSV file

- From Tableau Desktop connect to a CSV file

- File-location: …**Desktop/Data**

- Select one of these CSV files:
  -- **Airmovie..**
  -- **Data23..**

1. Use Table Extension object

> Instructions: Table Extension object

- Clean up Data Model canvas:
  Remove the CSV table from Data Model canvas in order to have a blank canvas
- Now place the **Table Extension object** on the Data Model canvas

- **Save** the workbook on Desktop

> Note: Table Extension / Code / Final data model

- For the Table Extension object the **python code** will be **provided during the Lab exercise**.

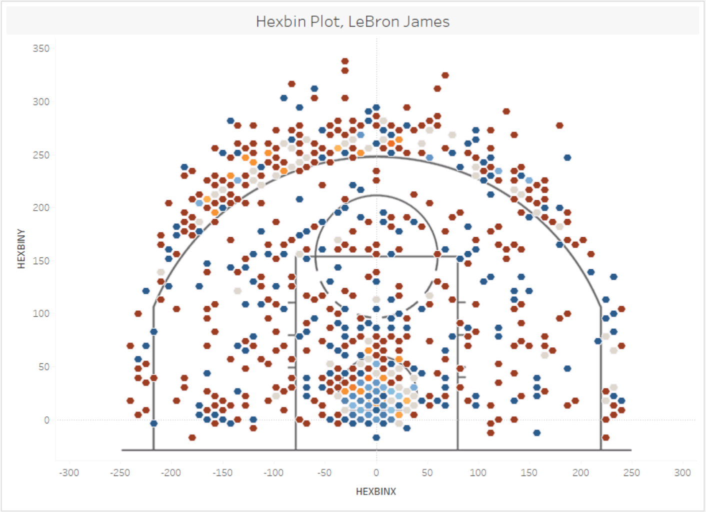
- For the NBA Lab exercises we will use **NO Input table** for the Table Extension object.
- For **each lab exercise** we will use a **new Table Extension object** which will be equipped with specific python code provided during the lab exercise.
- Each Table Extension object will generate one **Output table**.
- In the end we will create a **data model with three Table Extension objects** linked through Relationships based on the numeric field **Team ID**.
  Example of final data model:

Introduction NBA Data

Here is some information and links about the NBA Stats API and python library nba-api we used to request static data (player and teams), along with different endpoints like shotchart for player, teams or league data.
The nba_api package is Open Source with an MIT License
Glossary
PTS - Points:
The number of points scored
FGM - Field Goals Made:
The number of field goals that a player or team has made (both 2 and 3 pointers)
FTM - Free Throws Made:
Number of free throws that a player or team has made
LOC_X or LOC_Y:
The location coordinate of the player on the basketball court during game event
SHOT_MADE_FLAG:
Flag: Provides information if shot was successful or not
WL:
Flag: Provides information if a team has won or lost a game
Slack or Stack Overflow Community
Github

- Static
- Endpoints

Lab 1: Shotchart - Player



Time: 20 minutes
Prerequisite:

1. Table Extension object is on Data Model canvas
2. Rename Data Source: **NBA**
3. Rename Table Extension object: **Players**

---

1. Request NBA data for Players

> Instructions: Players

- Open with Note++ Editor Script: **Player_Shotchart_Hexbin.py**
- File-location: …**Desktop/Script**
- Copy and **paste script code** into Table Extension Editor
- Pres **Apply** to request data
- Optional - Check retrieved data:
  -- On the Output Table section press Update Now
- Optional - Update list of requested Players:
  -- Add e.g. "Nikola Jokic" to the list — In case Errors appear, please check spelling

Note: Python code explanation

- **ATTENTION**: In case you are having **Problems** to request the data:
- Use datadump Players
  -- Drop Table Extension object from canvas
- Connect to CSV file:
  -- File-location: …**Desktop/Data/Emergency data**
  -- File: player_data_hexbin.csv

1. Create Shotchart

> Instructions: Shotchart

- Create a view with the name: Shotchart detail
- Create a Scatterplot visualization using following fields:

- LOC_X:
  -- LOC_X: Convert it to Dimension
  -- LOC_X: Convert it to Continous
  -- Columns: LOC_X

- LOC_Y:
  -- LOC_Y: Convert it to Dimension
  -- LOC_Y: Convert it to Continous
  -- Rows: LOC_Y

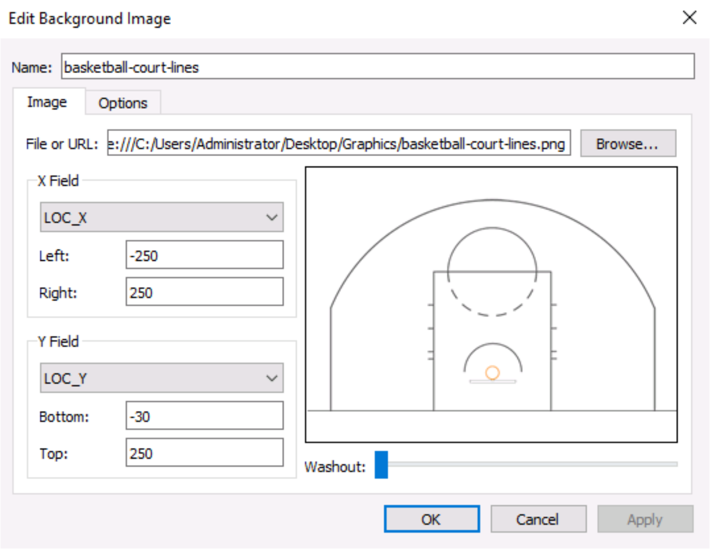
- SHOT_MADE_FLAG:
  -- SHOT_MADE_FLAG: Convert it to Dimension
  -- Color: SHOT_MADE_FLAG

- Optional: Add Player name to Filter
  -- Add PLAYER_NAME field to Filters
  -- Display it as Single value list

- Optional: Adjust color palette:
  -- Select: Orange-Blue Diverging

- Optional: Update Title:
  -- Add PLAYER_NAME to Title
  -- Center Title — Change Background to grey

- Optional: Add Background map:
  -- Navigate to Menu: Map → Background Images → Here select the Data Source NBA
  -- Select: Add Image:
  -- Select: Browse
  -- Browse to Folder: …Desktop/Graphics
  -- Select Image: **basketball-court-lines.png**
  -- Make following settings for the Image and hit OK



Edit Background Image

Name: basketball-court-lines

Image | Options

File or URL: e:///C:/Users/Administrator/Desktop/Graphics/basketball-court-lines.png

X Field: LOC_X

Left: -250

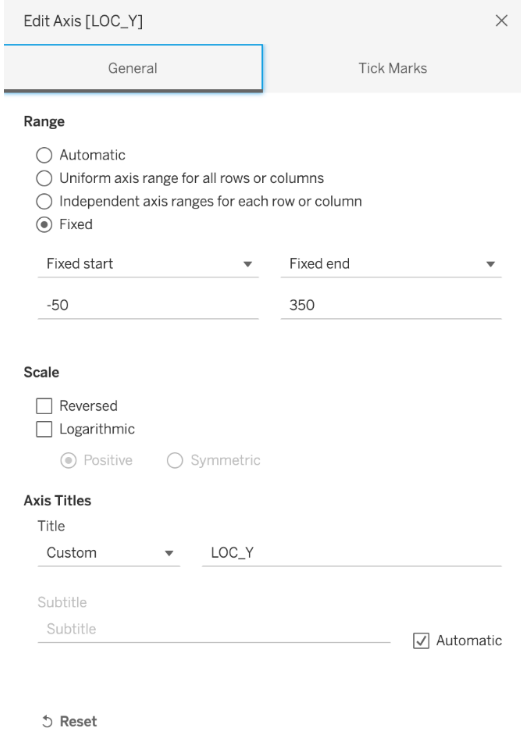
Right: 250

Y Field: LOC_Y

Bottom: -30

Top: 250

-- If needed, Edit the X and Y axes



Edit Axis [LOC_Y]

General | Tick Marks

Range

○ Automatic
○ Uniform axis range for all rows or columns
○ Independent axis ranges for each row or column
● Fixed

Fixed start: -50

Fixed end: 350

Scale

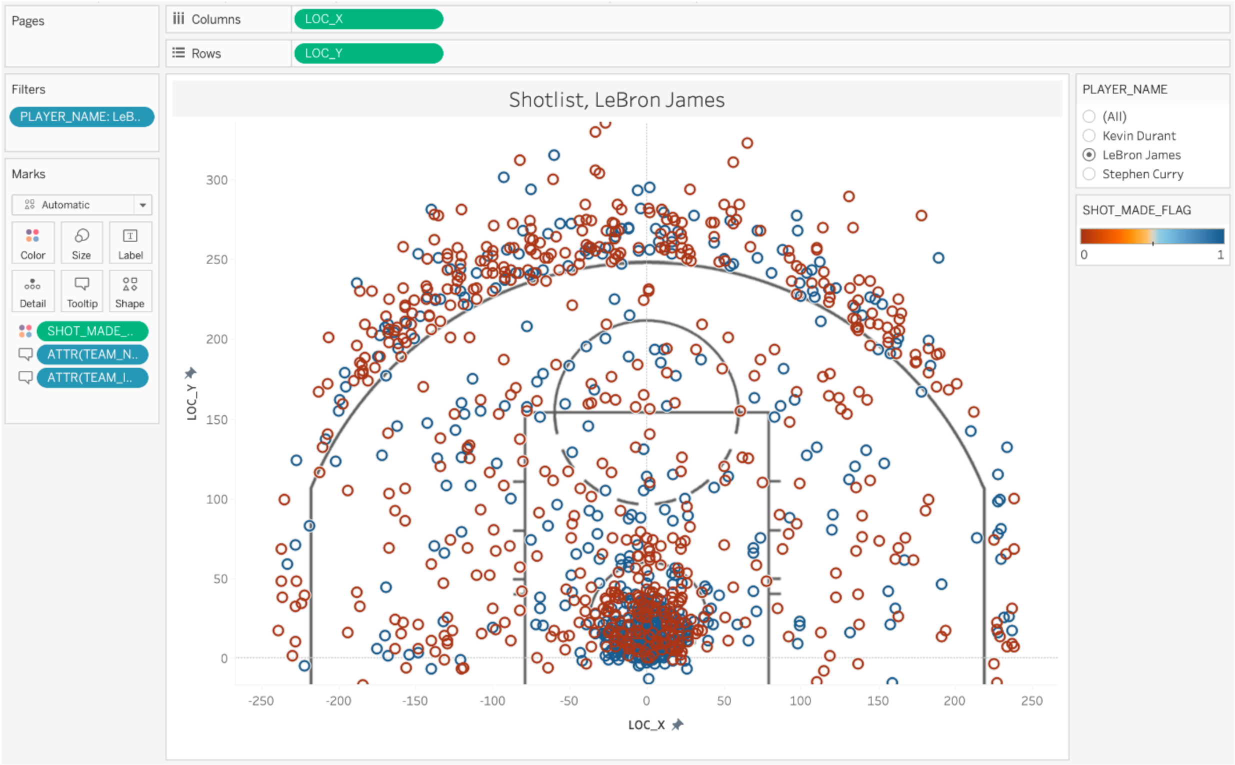
□ Reversed
□ Logarithmic
● Positive ○ Symmetric

Axis Titles

Title: Custom LOC_Y

Subtitle: Subtitle ☑ Automatic

Final Shotchart



1. Create Hexbin plot

> Instructions: Hexbin plotThe Hexbin (Scatter-)plot is a **Density** Scatterplot - We use it to visualize patterns or trends in dense data with many overlapping marks. Let' start!

- **Duplicate** the view: Shotchart detail
- Rename the new view: **Hexbin plot**

Place Hex graphic in MyTableauRepository

- Navigate to Hex graphic: **Hex_solid.png**
- File-location: …**Desktop/Graphics**
- **Copy** Hex_solid graphic
- Navigate to …Documents/My Tableau Repository/Shapes
- Here create a **new folder: Hex**
- **Paste** Hex_solid graphic into Hex folder
- Go back to Tableau Desktop

Calculations:

- Create a numeric **Parameter**:
  -- Name: `Hexbin Size`
  -- Data Type: Integer
  -- Range: 1..5
  -- Show Parameter
  -- Set Parameter to 4

- Create a **calculated field** for HexbinY:
  -- name: `HEXBINY`
  -- Formula:
  `HEXBINY([LOC_X]/[Hexbin Size], [LOC_Y]/[Hexbin Size]) * [Hexbin Size]`
  -- Convert it to Dimension
  -- Convert it to Continous

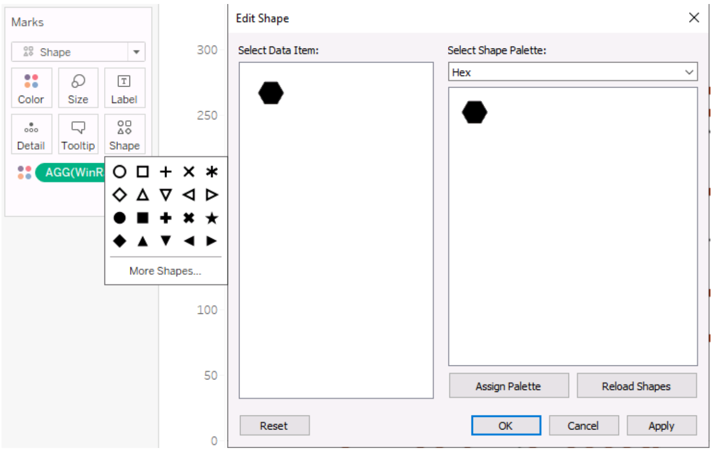
- Create a **calculated field** for HexbinX:
  -- name: `HEXBINX`
  -- Formula:
  `HEXBINX([LOC_X]/[Hexbin Size], [LOC_Y]/[Hexbin Size]) * [Hexbin Size]`
  -- LOC_X: Convert it to Dimension
  -- LOC_X: Convert it to Continous

Visualization:

- Update Scatterplot:
  -- Columns: Replace LOC_X by **HEXBINX**
  -- Rows: Replace LOC_Y by **HEXBINY**
  -- If needed rotate the Viz using 'Swap Rows and Columns'
  -- Color: SHOT_MADE_FLAG

- Update Scatterplot-Marks:
  -- Select Shape
  -- From Shape, select **More Shapes**
  -- First select: **Reload Shapes**
  -- Here select Shape Palette: **Hex**
  -- Finally, select Hex graphic
  -- Apply selection and confirm with OK

Hexbin shape selection

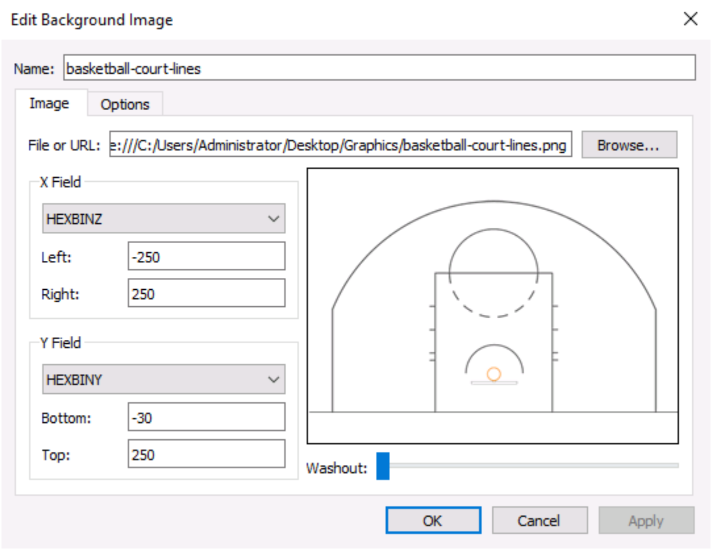


- Optional - Calculate WinRatio:

- Create a **calculated field** for Number of shots:
  -- name: `No_Player`
  -- Formula: `INT([PLAYER_NAME]=[PLAYER_NAME])`

- Create a **calculated field** for WinRatio:
  -- name: `WinRatio`
  -- Formula: `SUM([SHOT_MADE_FLAG])/SUM([No_Player])`

- Put WinRatio on color
- Adjust color palette:
  -- Orange-Blue Diverging
- Optional - Add Background map:
  -- Navigate to Menu: Map → Background Images → Here select the Data Source NBA
  -- Add Image:
  -- Browse to Folder: …Desktop/Graphics
  -- Select Image: **basketball-court-lines.png**
  -- Make following settings for the Image
  — If needed, Edit the X and Y axis accordingly to the image settings



Edit Background Image

Name: basketball-court-lines

Image | Options

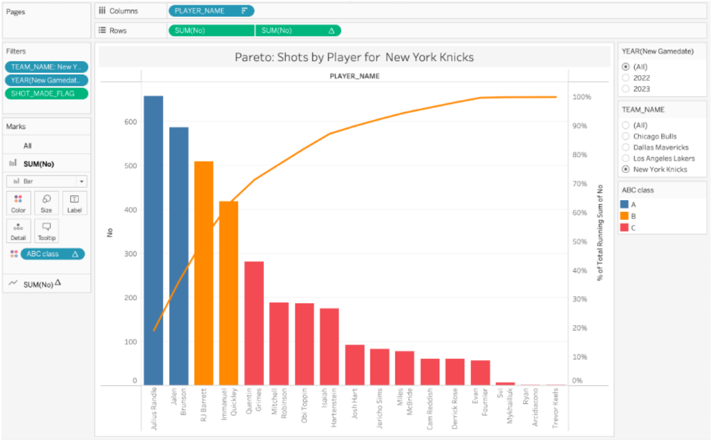
File or URL: e:///C:/Users/Administrator/Desktop/Graphics/basketball-court-lines.png

X Field: HEXBINZ
Left: -250
Right: 250

Y Field: HEXBINY
Bottom: -30
Top: 250

Final Hexbin plot

Lab 2: ABC Analysis - Team



Columns: PLAYER_NAME
Rows: SUM(No) | SUM(No) △

Filters: TEAM_NAME: New Y.. | YEAR(New Gamedat.. | SHOT_MADE_FLAG

Marks: All | SUM(No) | Bar | ABC class △ | SUM(No) △

YEAR(New Gamedate): ● (All) ○ 2022 ○ 2023

TEAM_NAME: ○ (All) ○ Chicago Bulls ○ Dallas Mavericks ○ Los Angeles Lakers ● New York Knicks

Time: 20 minutes
Prerequisite:

1. Second empty Table Extension object is on Data Source canvas

---

1. Request NBA data for Teams

> Instructions: Teams

- Open with Note++ Editor Script: **Team_Shotchart_Pareto.py**

- File-location: …**Desktop/Script**

- Copy and **paste script code** into Table Extension Editor
- Pres **Apply** to request data

- Optional - Check retrieved data:
  -- On the Output Table section press Update Now
- Optional - Update list of requested Teams:
  -- Add e.g. "San Antonio Spurs" to the list
  -- In case Errors appear, please check spelling

Note: Python code explanation

- **ATTENTION**: In case you are having **Problems** to request the data:

- Use datadump Team
  -- Drop Table Extension object from canvas

- Connect to CSV file:
  -- File-location: …**Desktop/Data/Emergency data**
  -- File: team_data_pareto.csv

---

1. Update Data Model

> Instructions: Data model

- For Relationship between both Table Extension objects use field: **TEAM ID**
- Rename Table Extension object: Teams

Note: Data Model

1. Create Pareto Diagram

> Instructions: Pareto

- Create a view with the name: ABC Analysis

- In the Data Pane minimize the Players folder

- Create a **Barchart** using following fields from **TEAMS folder**:
  -- Columns: PLAYER_NAME
  -- Rows: Teams(Count)
  -- Sort PLAYER_NAME by Field in descending order
  -- Add TEAM_NAME to Filters and show as Single Value list

  Players of one Team are now displayed and sorted by shot attempts

---

- Optional: Show scored points
  -- Add SHOT_MADE_FLAG to Filters and select 1

- Create a combo chart:

- Select a second time Teams(Count) from Data pane and place it to the Rows:
  -- Columns: PLAYER_NAME
  -- Rows: Teams(CNT) **Teams(Count)**

- Add Table Calculations:
  -- Right click in Rows on **second** field CNT(Teams) and select: **Add Table Calculations**
  -- As primary table calculation: **Running Total**
  -- Select checkbox: **Add secondary calculation**
  -- As secondary table calculation: **Percent of Total**
  -- Close the Table calculation window

- Update Visualization:
  -- For the second CNT(Teams) field change Mark Type to **Line**
  -- Right click in Rows on the second CNT(Teams) field and select **Dual Axis chart**
  -- If needed, update first CNT(Teams) Mark Type to **Bars**
  -- If needed, update colors

  Players of one Team are displayed and sorted by shot attempts, additionally the Running Total of shot attempts is calculated and displayed as Percent of Total.

Final Pareto diagram

1. Add ABC Analysis

> Instructions: ABCCalculations:

- From folder Teams duplicate field **PLAYER_NAME**:
  -- Name: `PLAYER_NAME_Teams`

- Create a **calculated field** for number of records respectively shot attempts:
  -- Name: `No_Teams`

-- Formula:
-- `INT([PLAYER_NAME_Teams]=[PLAYER_NAME_Teams])`

It's a calculated field that provides the number of records and mirrors the required level of detail for the ABC Analysis.

- Create a **calculated field** for Running Sum of Total records respectively shot attempts:
  -- Name: `No Running Total`
  -- Formula:
  -- `RUNNING_SUM(Sum([No_Teams]))`

- Create a **calculated field** for % of Running Total:
  -- Name: `% No Running Total`
  -- Formula:
  -- `[No Running Total]/ TOTAL(Sum([No_Teams]))`

- Create a **calculated field** for ABC classification:
  -- Name: `ABC`
  -- Formula:
  -- `IF ZN(LOOKUP([% No Running Total], -1)) < .2 THEN 'A’ ELSEIF ZN(LOOKUP([% No Running Total], -1)) < .6 THEN 'B’ ELSE 'C’ END`

  This calculation checks the position of the current Player in the number of throws range and place the Player into the corresponding ABC category. Tableau LOOKUP function is used to evaluate whether the total number of throws of all the higher ranked Player above the current Player is < 20% of total number of throws of all Player, if so, the current Player falls into the category ‘A’, if the total number of throws of all the higher ranked Player > or = 20%, that means the current Player does not consist of the top 20% Player throws and it will falls into the category ‘B’ or ‘C’. Then we can use the same approach to find which Player falls into category ‘B’ and the remaining ones will be in category ‘C’.

Visualization:

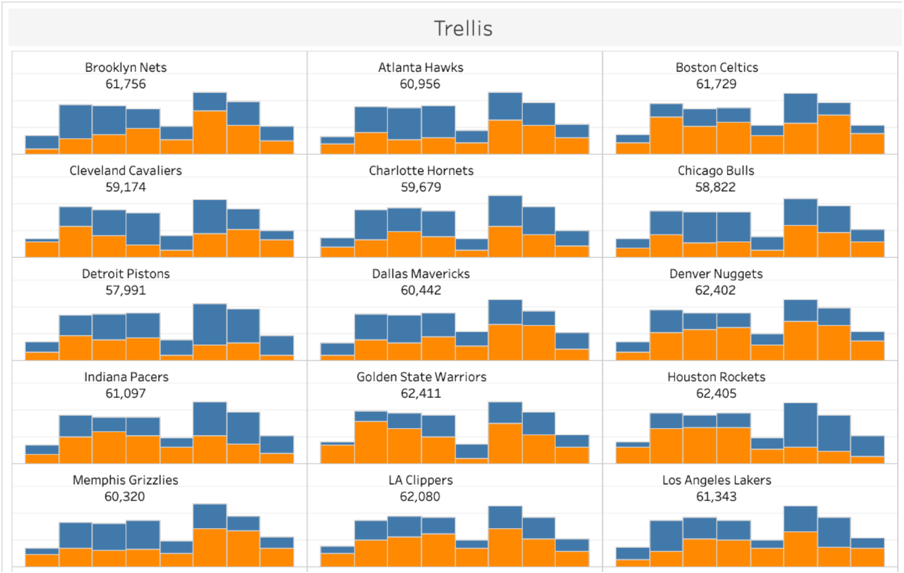
- Add ABC field to color of Barchart

- Optional: Update Title:
  -- Add TEAM_NAME to Title
  -- Center Title
  -- Change Background to grey

- Hide Field Labels for Columns

Final ABC Analysis

Lab 3: Trellis Chart - League



Time: 25 minutes
Prerequisite:

1. Third empty Table Extension object is on Data Source canvas

---

1. Request NBA data for League

> Instructions: League

- Open with Note++ Editor Script: **Leaguegamelog_Trellis.py**

- File-location: …**Desktop/Script**

- Copy and **paste script code** into Table Extension Editor
- Pres **Apply** to request data

- Optional - Check retrieved data:
  -- On the Output Table section press Update Now

- **ATTENTION**: In case you are having **Problems** to request the data:

- Use datadump League
  -- Drop Table Extension object from canvas

- Connect to CSV file:
  -- File-location: …**Desktop/Data/Emergency data**
  -- File: league_data_trellis.csv

1. Update Data Model

> Instructions: Data model

- For Relationship between both Table Extension objects use field: **TEAM ID**
- Rename Table Extension object: League

Note: Data Model

1. Create grid for Trellis chart

> Instructions: Grid

- Create a view with the name: Trellis

- In the Data Pane minimize the Players/Teams folder
  Calculations:
- Create a **calculated field** to index the values (Teams):
  -- Name: `Index`
  -- Formula:
  -- `Index()`

- Create a numeric **Parameter** to determine the number of columns to display:
  -- Name: `No of columns`
  -- Data Type: Integer
  -- Range: 1..5
  -- Show Parameter

- Create a **calculated field** for columns:
  -- Name: `Columns`
  -- Formula:
  -- `Index % [No of columns]`
  -- Convert to: **Discrete**
  Columns are calculated using Mudula Division
- Create a **calculated field** for rows:
  -- Name: `Rows`
  -- Formula:
  -- `INT((Index-1) / [No of columns])`
  -- Convert to: **Discrete**

- From League folder duplicate TEAM_NAME:
  -- Name: `League_Team`
  These fields will be used to create the grid for our Trellis chart.

Visualization: Grid

- Create a visualization using following fields:
  -- Detail mark: League_Team
  -- Columns: Columns
  -- **Compute** Columns field **Using** League_Team
  -- Rows: Rows
  -- **Compute** Rows field **Using** League_Team
  Congratulations: That is the grid for the Trellis chart.

Note: Grid

1. Create Trellis chart

> Instructions: Trellis chartCalculations:

- From League folder duplicate GAME_DATE and change Data Type:
  -- Name: `New_GAME_DATE`
  -- Change Data Type: Date
  -- Convert to **Continous**

Visualization

- Update the visualization using following fields:
  -- Columns: Columns **New_GAME_Date**
  -- Rows: Rows **SUM(PTS)**
  Please make sure you have adjusted aggregation of PTS to SUM.
  Congratulations: This is already a basic Trellis chart.

-- Optional: Use SUM(FTM) or SUM(FGM) as Measure for Rows
FTM: Number of free throws that a player or team has made
FGM: The number of field goals that a player or team has made (both 2 and 3 pointers)
-- Optional: Change color palette
Note: Trellis chart

1. Create Trellis chart label

> Instructions: Trellis labelCalculations:

- Create a **calculated field** for Total Team PTS:
  -- Name: `Total Team PTS`
  -- Formula:
  -- `WINDOW_SUM(SUM([PTS]))`
  This calculation will provide for each Team the Total PTS.
- Create a **calculated field** (nested LOD) to determine the max PTS:
  -- Name: `Max PTS`
  -- Formula:
  -- `{MAX({ FIXED [League_Team], YEAR([New_GAME_DATE]): SUM([PTS])})}`
  This calculation is used for aesthetics. It will help us to find a fix position for the Trellis label above the highest (Max

PTS) bar. Though it overlaps with other marks it will be avoided.

- Create a **calculated field** to display Total Team PTS:
  -- Name: `Halfway Mark`
  -- Formula:
  -- `IF [Index]=INT(SIZE()/2) THEN SUM([Max PTS]) END`
  Get the midpoint the halfway mark between the first and the last point

---

Visualization

- Update the visualization using following fields:
  -- Columns: Columns New_GAME_Date
  -- Rows: Rows SUM(PTS) **Halfway Mark**

- Change appearance of **Halfway Mark**:
  -- Select Circle — Size: Minimize it completly
  -- Color: Opacity = 0%
  -- Text: League_Team
  -- Text: Total Team PTS

- Create Dual Axis chart
  -- Right click in Rows on Halfway and select Dual Axis

- Synchronize Y Axis
  -- Right click on right Y Axis and select Synchronize Axis

- Remove Measure Names
  -- Remove Measure Names from Halfway Mark
  -- Remove Measure Names from SUM(PTS)

- Overlapping: Update Max PTS and add Padding (*1.5)
  -- Name: `Max PTS`
  -- Formula:
  -- `{MAX({ FIXED [League_Team], YEAR([New_GAME_DATE]): SUM([PTS])})}*1.5`

- Change appearance of SUM(PTS):
  -- Mark Type: Bar
  Congratulations: That is already a better Trellis chart with labels.
  But you can do more.

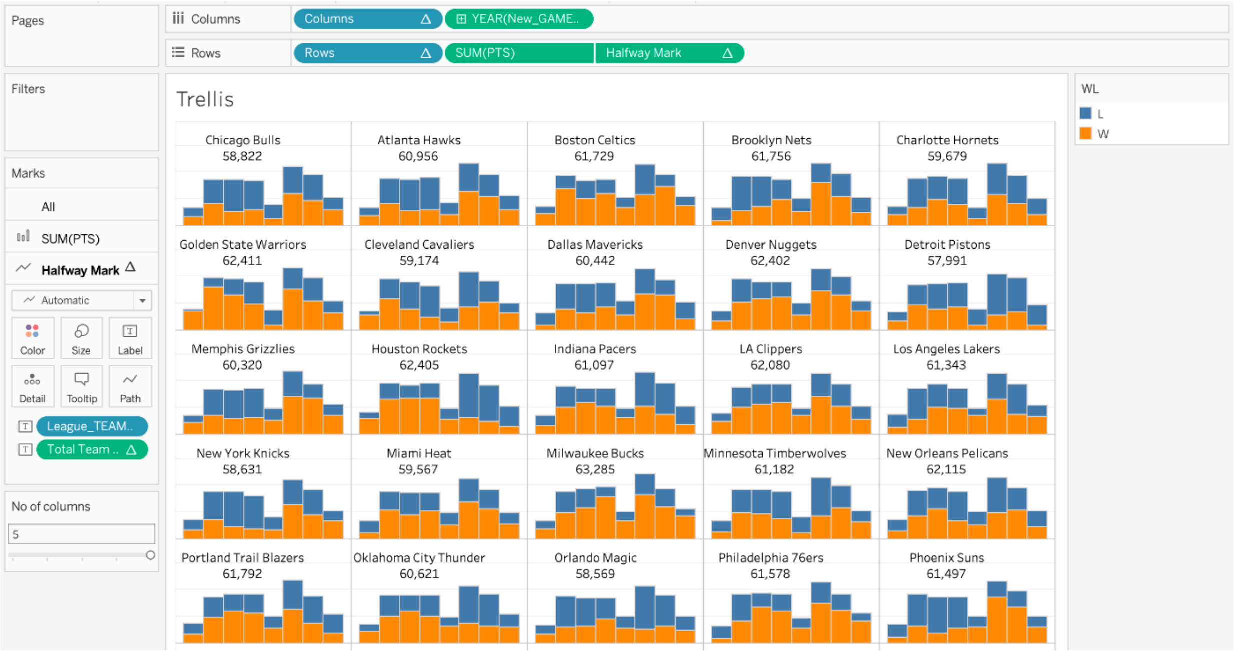
- Optional: Add WL
  -- Add WL to color of SUM(PTS)

- Optional: Remove Axes
  -- Right click on right Y Axis and deselect Show Header
  -- (2x) Right click on both left Y Axes and deselect Show Header

-- Right click on bottom X Axis and deselect Show Header
-- Right click on top X Axis and deselect Show Header

- Note: 210 Null values
  These Null values result from the Halfway Mark. We are showing just one mark for Halfway Mark for each (Sub-) Trellis chart respectively Team. That means we are showing 30 Teams * 1 Mark = 30 Marks. But there could be in total 240 Marks because 30 Teams * 8 years = 240 Marks. Finally 210 Marks are Null values (240-30=210). Please keep that Null Values.

Note: Trellis chart label



Columns: Columns Δ, YEAR(New_GAME..

Rows: Rows Δ, SUM(PTS), Halfway Mark Δ

Marks: All, SUM(PTS), Halfway Mark Δ, Automatic

League_TEAM..

Total Team .. Δ

No of columns
5