DAX Queries

DAX Measures for Real-Estate-Analytics Dashboard

This document provides a detailed breakdown of the DAX (Data Analysis Expressions) measures used to create the Key Performance Indicators (KPIs) in the Real Estate Analytics Power BI dashboard. Each measure is explained with its purpose, syntax, and the functions used.

1. Last 12 Month Sales

This measure calculates the total sales revenue from the last 12 months from the most recent date in the dataset.

```
Last 12 Month Sales =
CALCULATE(
    SUM(Housing[purchase_price]),
    DATESINPERIOD(
        Housing[date],
        MAX(Housing[date]),
        -12,
        MONTH
    )
)
```

Explanation:
The formula sums the `purchase_price` for all dates within the last 12-month period. This is a rolling calculation that dynamically updates as new data is added.

Function Breakdown:

- `CALCULATE`: Modifies the context in which the data is evaluated. Here, it applies a time-based filter to the `SUM` expression.
- `SUM(...)`: Aggregates the total value of the `purchase_price` column.

- **`DATESINPERIOD(...)`**: Returns a table containing a column of dates that begins from the `MAX(Housing[date])` and goes back 12 months.

2. Median Sales Price Change

This measure calculates the percentage change in the median sales price between the current year and the previous year.

```
Median Sales Price Change =
VAR CurrMedianPrice =
    MEDIANX(
        FILTER(Housing, YEAR(Housing[date].[Date]) = YEAR(MAX(Housing[date].[Date]))),
        Housing[purchase_price]
    )
VAR PreMedianPrice =
    MEDIANX(
        FILTER(Housing, YEAR(Housing[date].[Date]) = YEAR(MAX(Housing[date].[Date])) - 1),
        Housing[purchase_price]
    )
RETURN
    IF(PreMedianPrice <> 0, (CurrMedianPrice - PreMedianPrice) / PreMedianPrice, BLANK())
```

Explanation:
The formula first calculates the median sales price for the most recent full year (`CurrMedianPrice`) and the year prior to that (`PreMedianPrice`). It then calculates the percentage difference. Using `MEDIAN` instead of `AVERAGE` makes this metric more robust to outliers.

Function Breakdown:

- **`VAR ... RETURN`**: A best practice for storing intermediate results (`CurrMedianPrice` and `PreMedianPrice`) to improve readability and performance.

- `MEDIANX(...)`: Returns the median number of an expression evaluated for each row in a table.
- `FILTER(...)`: Returns a subset of the `Housing` table where the year matches the specified condition (current year or previous year).
- `IF(...)`: Prevents a division-by-zero error if there were no sales in the previous year.

3. Units Sold in Last Year & Quarter

This measure counts the distinct number of houses sold in the most recent year and quarter available in the data.

```
Units Sold in Last Year & Quater =
CALCULATE(
    DISTINCTCOUNT(Housing[house_id]),
    YEAR(Housing[date]) = YEAR(MAX(Housing[date])) &&
    QUARTER(Housing[date]) = QUARTER(MAX(Housing[date]))
)
```

Explanation:
The formula filters the `Housing` table to include only records from the latest year and quarter and then performs a distinct count of `house_id` to get the number of unique properties sold.

Function Breakdown:

- `DISTINCTCOUNT(...)`: Counts the number of unique values in the `house_id` column.
- `YEAR(MAX(Housing[date]))`: Dynamically finds the most recent year in the dataset.
- `QUARTER(MAX(Housing[date]))`: Dynamically finds the most recent quarter in the dataset.

4. YOY Sales Growth

This measure calculates the Year-over-Year (YOY) growth rate for total sales revenue.

```
YOY Sales Growth =
VAR CurrentYearSales =
    CALCULATE(
        SUM(Housing[purchase_price]),
        YEAR(Housing[date]) = YEAR(MAX(Housing[date]))
    )
VAR PrevYearSales =
    CALCULATE(
        SUM(Housing[purchase_price]),
        YEAR(Housing[date]) = YEAR(MAX(Housing[date])) - 1
    )
RETURN
    IF(PrevYearSales <> 0, (CurrentYearSales - PrevYearSales) / PrevYearSales, BLANK())
```

Explanation:
This formula compares the total sales of the most recent year (`CurrentYearSales`) with the total sales of the previous year (`PrevYearSales`) to determine the growth percentage.

Function Breakdown:

- This measure uses a similar `VAR...RETURN` structure to the `Median Sales Price Change` measure for clarity and efficiency.
- The core logic is based on filtering the `SUM` of `purchase_price` for the two respective years and then calculating the percentage change.

5. Average Price SQM

Calculates the overall average price per square meter across all properties in the current filter context.

```
Average Price SQM = AVERAGE(Housing[sqm_price])
```

Explanation:
A straightforward measure that computes the arithmetic mean of the `sqm_price` column. It can be sliced and diced by other dimensions (like region or house type) in the report.

Function Breakdown:

- `AVERAGE(...)`: Returns the average of all the numbers in a column.

6. Offer to SQM Ratio

This measure calculates the ratio of the total offer price to the total square meterage of properties. It essentially represents the average offer price per square meter.

```
Offer to SQM Ratio = DIVIDE(SUM(Housing[Offer Price]), SUM(Housing[sqm]))
```

Explanation:
Instead of averaging the per-unit `sqm_price`, this measure calculates the ratio based on the grand totals of offer price and area. This can give a different, more macro-level insight into pricing.

Function Breakdown:

- `DIVIDE(...)`: A safe division function that handles division-by-zero cases automatically (returning `BLANK` by default).
- `SUM(...)`: Used here to aggregate the total offer price and total square meters before the division occurs.

7. Sales by Region

Calculates the total purchase price for a region, ignoring any filters that might be applied to other columns within the `Housing` table except for the region itself.

```
Sales by Region = CALCULATE(SUM(Housing[purchase_price]), ALLEXCEPT(Housing, Housing[region]))
```

Explanation:
This is useful for creating visuals where you want to show the total sales for a region as a percentage of the grand total, or to ensure that filtering by other attributes (like `house_type` or `sales_type`) doesn't affect the regional total calculation.

Function Breakdown:

- `ALLEXCEPT(...)`: Removes all context filters from the `Housing` table except for the filters that have been applied to the `region` column.

8. Total YTD Sales

Calculates the cumulative total of sales from the beginning of the current year up to the latest date available in the dataset.

```
Total YTD Sales = TOTALYTD(SUM(Housing[purchase_price]), Housing[date].[Date])
```

Explanation:
A standard Year-to-Date (YTD) calculation. This is a powerful time intelligence function for tracking performance over the course of a year.

Function Breakdown:

- `TOTALYTD(...)`: Evaluates the specified expression (`SUM(Housing[purchase_price])`) over the interval that begins on the first day of the year and ends with the last date in the specified date column, after applying filters.