

# Decision Support Systems

## Laboratory of Data Science Project

Simone Di Luna

Department of Computer Science  
University of Pisa

July 2023

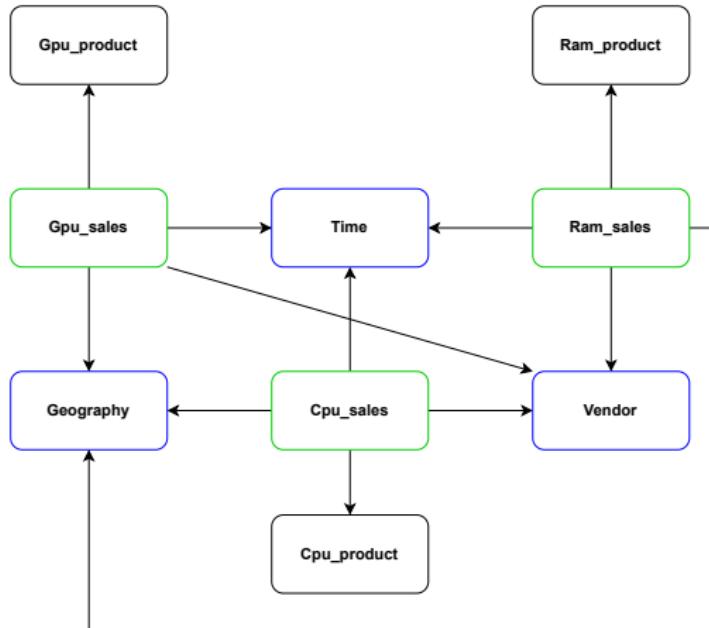


# Table of Contents

- 1 Data warehouse design and population
- 2 ETL process
- 3 SSAS database design
- 4 MDX query resolution
- 5 Power BI dashboard

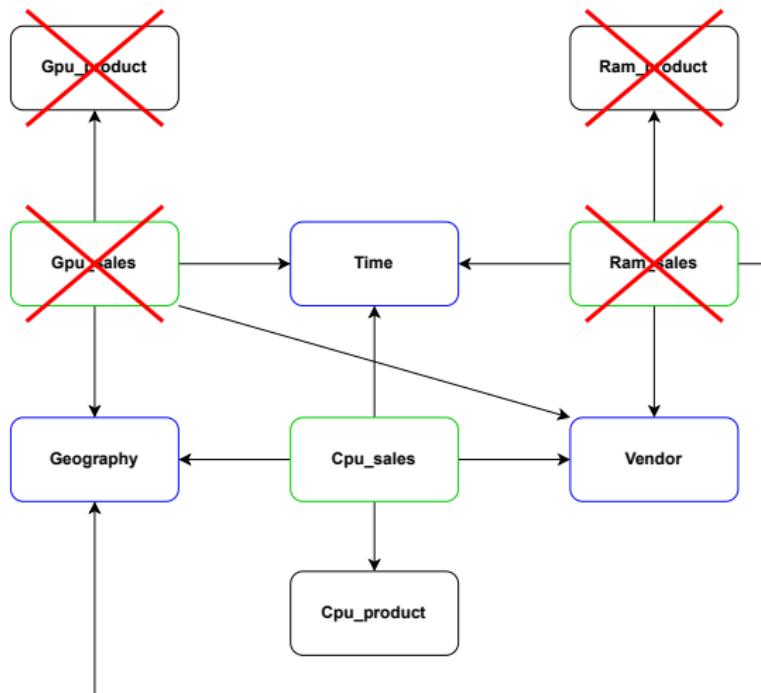
# From the data warehouse . . .

- Fact tables
- Conformed dimension tables
- Simple dimension tables
- Referential constraints (arrows)



Constellation schema

# ... to the CPU data mart



Star schema

# Composition of CSV files

**fact.csv**

- Id
- gpu\_code
- cpu\_code
- ram\_code
- time\_code
- geo\_code
- vendor\_code
- sales\_uds
- sales\_currency

**cpu.csv**

- cpu\_code
- brand
- series
- name
- n\_cores
- socket

**geography.csv**

- geo\_code
- continent
- country
- region
- currency

**time.csv**

- time\_code
- year
- month
- day
- week

**vendor.csv**

- vendor\_code
- name

# Preliminary analysis

For each CSV file and for each variable in the file, we are interested to discover the following information:

- Data type.
- Domain.
- Character encoding (if qualitative).
- Number of missing values and form in which they appear.

Why?

# Preliminary analysis

For each CSV file and for each variable in the file, we are interested to discover the following information:

- Data type.
- Domain.
- Character encoding (if qualitative).
- Number of missing values and form in which they appear.

## Why?

# Preliminary analysis

- To help us choose between different data types and arguments in MS SQL Server:
  - char/varchar or nchar/nvarchar?
  - What about the length?
  - int, smallint, tinyint, money,...?
- To check the semantic accuracy.

# simonediluna\_DB: Diagram

- Surrogate keys
- Data types
- Not null constraints
- Default constraints
- Unique constraints

Column Name	Condensed Type	Identity	Nullable	Default Value
geo_id	int	<input checked="" type="checkbox"/>	No	
geo_code	int	<input type="checkbox"/>	No	
continent	varchar(7)	<input type="checkbox"/>	No	
country	varchar(30)	<input type="checkbox"/>	No	
region	varchar(32)	<input type="checkbox"/>	No	
currency	char(3)	<input type="checkbox"/>	No	

Column Name	Condensed Type	Identity	Nullable	Default Value
cpu_id	int	<input checked="" type="checkbox"/>	No	
cpu_code	int	<input type="checkbox"/>	No	
brand	varchar(7)	<input type="checkbox"/>	No	
series	varchar(25)	<input type="checkbox"/>	No	
cpu_name	varchar(39)	<input type="checkbox"/>	No	
n_cores	smallint	<input type="checkbox"/>	No	
socket	varchar(12)	<input type="checkbox"/>	No	

Column Name	Condensed Type	Identity	Nullable	Default Value
fact_id	int	<input checked="" type="checkbox"/>	No	
cpu_id	int	<input type="checkbox"/>	No	
time_id	int	<input type="checkbox"/>	No	
geo_id	int	<input type="checkbox"/>	No	
vendor_id	int	<input type="checkbox"/>	No	
sales_usd	money	<input type="checkbox"/>	No	((0))
sales_currency	money	<input type="checkbox"/>	No	((0))
cost	money	<input type="checkbox"/>	No	((0))

Column Name	Condensed Type	Identity	Nullable	Default Value
vendor_id	int	<input checked="" type="checkbox"/>	No	
vendor_code	int	<input type="checkbox"/>	No	
vendor_name	nvarchar(32)	<input type="checkbox"/>	No	

- Attribute renaming
- New attributes:
  - the\_quarter
  - day\_name
  - cost

Column Name	Condensed Type	Identity	Nullable	Default Value
time_id	int	<input type="checkbox"/>	No	
the_year	smallint	<input type="checkbox"/>	No	
month_of_year	tinyint	<input type="checkbox"/>	No	
day_of_month	tinyint	<input type="checkbox"/>	No	
week_of_year	tinyint	<input type="checkbox"/>	No	
the_quarter	char(2)	<input type="checkbox"/>	No	
day_name	char(9)	<input type="checkbox"/>	No	

# simonediluna\_DB: Indexes

Before populating the database

Clustered indexes on primary keys (auto-generated).

After populating the database

- Non-clustered indexes on foreign keys of the fact table.
- Non-clustered columnar indexes on the attributes that was assumed to be used most frequently in the WHERE/GROUP BY clauses (1 index for each dimension table).

# simonediluna\_DB: Indexes

Before populating the database

Clustered indexes on primary keys (auto-generated).

After populating the database

- Non-clustered indexes on foreign keys of the fact table.
- Non-clustered columnar indexes on the attributes that was assumed to be used most frequently in the WHERE/GROUP BY clauses (1 index for each dimension table).

# Missing values

Which attributes are affected?

Attribute	File	Type	Number
series	cpu	<i>"Unknown"</i>	1
cpu_code	fact	""	4 409 487
gpu_code	fact	""	5 017 522
ram_code	fact	""	2 602 347

# Missing values

## How to treat them?

Missing values in fact.csv

Skip the rows where cpu\_code = "" and remove the gpu\_code and ram\_code columns.

Missing value in cpu.csv

Infer the series of the CPU by looking at its name:

cpu_code	brand	series	name	n_cores	socket
467	AMD	Unknown	Amd E-Series E2-3200	2	FM1

# Missing values

## How to treat them?

Missing values in fact.csv

Skip the rows where `cpu_code = ""` and remove the `gpu_code` and `ram_code` columns.

Missing value in cpu.csv

Infer the series of the CPU by looking at its name:

cpu_code	brand	series	name	n_cores	socket
467	AMD	Unknown	Amd E-Series E2-3200	2	FM1

# Missing values

## How to treat them?

Missing values in fact.csv

Skip the rows where cpu\_code = "" and remove the gpu\_code and ram\_code columns.

Missing value in cpu.csv

Infer the series of the CPU by looking at its name:

cpu_code	brand	series	name	n_cores	socket
467	AMD	Unknown	Amd E-Series E2-3200	2	FM1

# Other inconsistencies in the cpu.csv file

cpu_code	brand	series	name	n_cores	socket
1	AMD	Amd 2650	Amd Sempron 2650	2	AMD Socket AM1
2	AMD	Amd 3850	Amd Sempron 3850	4	AMD Socket AM1
3	AMD	Amd 5150	Amd Athlon 5150	4	AMD Socket AM1
4	AMD	Amd 5350	Amd Athlon 5350	4	AMD Socket AM1
187	AMD	Amd E2-3200	Amd E2-3200	2	AMD Socket FM1
200	AMD	Amd E-Series E-450	Amd E-Series E-450	2	Intel Socket BGA413
240	AMD	Amd Opteron	Amd Opteron 2373 Ee	4	Intel Socket LGA3647
467	AMD	Unknown	Amd E-Series E2-3200	2	AMD Socket FM1

Domain knowledge

# Other inconsistencies in the cpu.csv file

## How to solve them?

- Infer the correct value from the other fields, if possible.
- For the socket field remove redundant information (i.e., brand + "Socket").
- Check the data type and convert it to the appropriate one (e.g., integer).

# Other inconsistencies in the cpu.csv file

## How to solve them?

- Infer the correct value from the other fields, if possible.
- For the socket field remove redundant information (*i.e.*, brand + “Socket”).
- Look for information on the manufacturer’s official website.

# Other inconsistencies in the cpu.csv file

## How to solve them?

- Infer the correct value from the other fields, if possible.
- For the socket field remove redundant information (*i.e.*, brand + “Socket”).
- Look for information on the manufacturer’s official website.

# Other inconsistencies in the cpu.csv file

How to solve them?

- Infer the correct value from the other fields, if possible.
- For the socket field remove redundant information (*i.e.*, brand + “Socket”).
- Look for information on the manufacturer’s official website.

## Computation of new attributes: Quarters

The quarter in which a fact occurred can be easily inferred from the month of the year:

```
def compute_quarter(month: str):
    return ["Q1", "Q2", "Q3", "Q4"][(int(month)-1) // 3]
```

# Computation of new attributes: Name of the day of the week

The name of the day of the week on which a particular fact occurred can be inferred from the `time_id`. Specifically, it is necessary to **fix a point in time where the name of the day is known** and then **count the days elapsed from there to the date of the fact**.

What time reference point to choose?

The Unix epoch: Thursday, January 1, 1970.

The earliest date in the time dimension table is March 22, 2013.

# Computation of new attributes: Name of the day of the week

The name of the day of the week on which a particular fact occurred can be inferred from the `time_id`. Specifically, it is necessary to **fix a point in time where the name of the day is known** and then **count the days elapsed** from there to the date of the fact.

## What time reference point to choose?

The Unix epoch: Thursday, January 1, 1970.

The earliest date in the time dimension table is March 22, 2013.

# Computation of new attributes: Name of the day of the week

The name of the day of the week on which a particular fact occurred can be inferred from the `time_id`. Specifically, it is necessary to **fix a point in time where the name of the day is known** and then **count the days elapsed** from there to the date of the fact.

## What time reference point to choose?

The **Unix epoch**: Thursday, January 1, 1970.

The earliest date in the time dimension table is March 22, 2013.

# Computation of new attributes: Cost

$C = \frac{R}{k+1}$ , where  $k = \frac{R-C}{C}$  is the markup which was randomly drawn from a uniform distribution.

```
def compute_cost(
    revenue: str,
    markup_range=(0.15, 0.4)
) -> float:
    rnd = random.random()
    delta = markup_range[1] - markup_range[0]
    markup = rnd * delta + markup_range[0] # min-max norm.
    return float(revenue) / (1 + markup) # C
```

# Data warehouse population

The chronological steps:

- ① Populate the dimension tables.
- ② For each dimension table in the DB, compute a dictionary mapping the values of the old PK to the auto-generated values of the surrogate key.
- ③ Replace the old FKS in the fact.csv file with the new FKS referencing the auto-generated PKs.
- ④ Populate the fact table.

# Data warehouse population

The chronological steps:

- ① Populate the dimension tables.
- ② For each dimension table in the DB, compute a dictionary mapping the values of the old PK to the auto-generated values of the surrogate key.
- ③ Replace the old FKS in the fact.csv file with the new FKS referencing the auto-generated PKs.
- ④ Populate the fact table.

# Data warehouse population

The chronological steps:

- ① Populate the dimension tables.
- ② For each dimension table in the DB, compute a dictionary mapping the values of the old PK to the auto-generated values of the surrogate key.
- ③ Replace the old FKS in the fact.csv file with the new FKS referencing the auto-generated PKs.
- ④ Populate the fact table.

# Data warehouse population

The chronological steps:

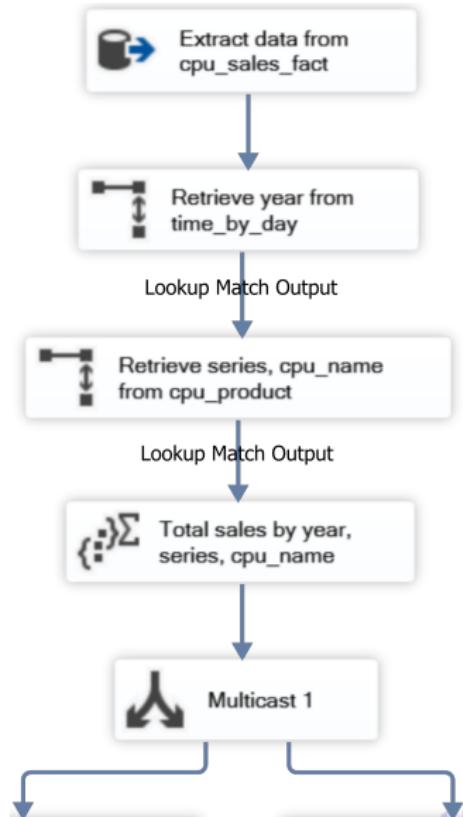
- ① Populate the dimension tables.
- ② For each dimension table in the DB, compute a dictionary mapping the values of the old PK to the auto-generated values of the surrogate key.
- ③ Replace the old FKS in the fact.csv file with the new FKS referencing the auto-generated PKs.
- ④ Populate the fact table.

# ETL task

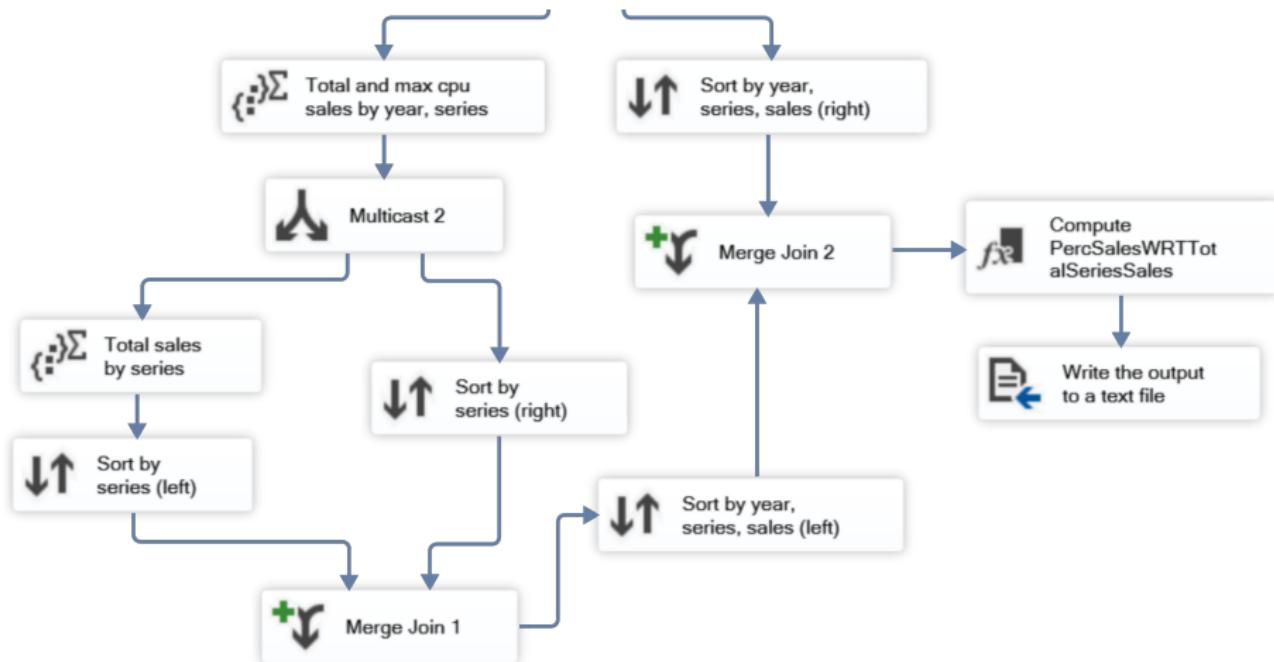
## ETL workflow in Visual Studio

For each year and CPU series, the name of the CPU that sold the most that year and the percentage in sales with respect to the total sales of the corresponding CPU series.

# SSIS solution part 1



## SSIS solution part 2



# Preliminary settings

Server: `http://lds.di.unipi.it/olap/msmdpump.dll`

Database: `simonediluna_SSAS_DB`

Conn. string: `Provider=SQLOLEDB.1;`

`Data Source=lds.di.unipi.it;`

`Password=*****;`

`User ID=simonediluna;`

`Initial Catalog=simonediluna_DB`

# Dimensions

The cube has 4 dimensions, each based solely on one table in the data warehouse:

- Time
- Vendor
- CPU
- Geography

# Measures

Two group of measures:

## CPU Sales Fact

- Sales Usd (Sum, Currency)
- Sales Currency (Sum, Currency)
- Cost (Sum, Currency)
- Cpu Sales Fact Count (Count, Standard)
  - Gross Profit (Currency)
  - Margin (Percent)
  - Markup (Percent)

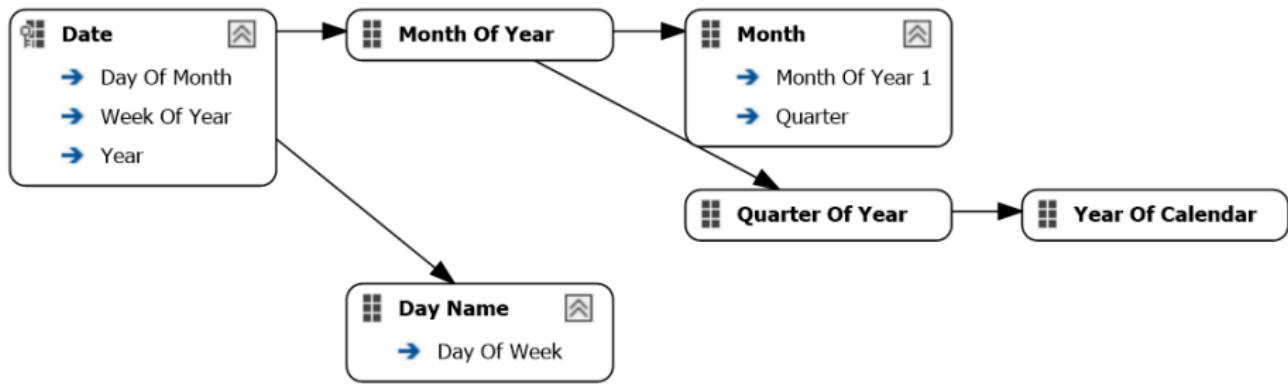
## Vendor

- NVendors (DistinctCount, Standard)

# Time dimension: Structure

Attributes	Hierarchies	Data Source View
 Time <ul style="list-style-type: none"> <li> Date</li> <li> Day Name</li> <li> Day Of Month</li> <li> Day Of Week</li> <li> Month</li> <li> Month Of Year</li> <li> Month Of Year 1</li> <li> Quarter</li> <li> Quarter Of Year</li> <li> Week Of Year</li> <li> Year</li> <li> Year Of Calendar</li> </ul>	 Calendar <ul style="list-style-type: none"> <li>▪ Year</li> <li>▪▪ Quarter               <ul style="list-style-type: none"> <li>➔ Year Of Calendar</li> </ul> </li> <li>▪ Month               <ul style="list-style-type: none"> <li>➔ Month</li> <li>➔ Quarter Of Year</li> </ul> </li> <li>▪▪ Date               <ul style="list-style-type: none"> <li>➔ Day Name</li> <li>➔ Day Of Month</li> <li>➔ Month Of Year</li> <li>➔ Week Of Year</li> <li>➔ Year</li> <li>&lt;new level&gt;</li> </ul> </li> </ul>	 time_by_day <ul style="list-style-type: none"> <li> time_id</li> <li> the_year</li> <li> month_of_year</li> <li> day_of_month</li> <li> week_of_year</li> <li> the_quarter</li> <li> day_name</li> <li> day_of_week</li> <li> month_name</li> </ul>

# Time dimension: Attribute Relationships



# Geography dimension

## Attributes

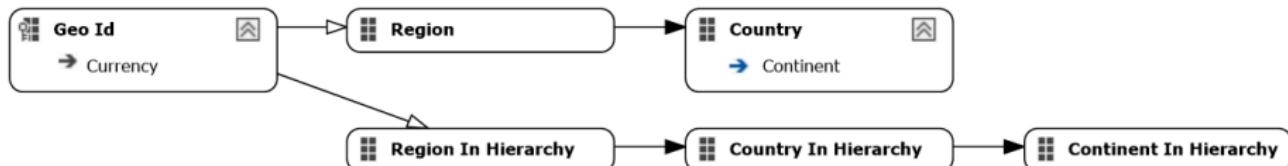
- Geography
- Continent
- Continent In Hierarchy
- Country
- Country In Hierarchy
- Currency
- Geo Id
- Region
- Region In Hierarchy

## Hierarchies

- GeoHierarchy
- Continent
  - Country
    - ➔ Continent In Hierarchy
  - Region
    - ➔ Country In Hierarchy
    - <new level>

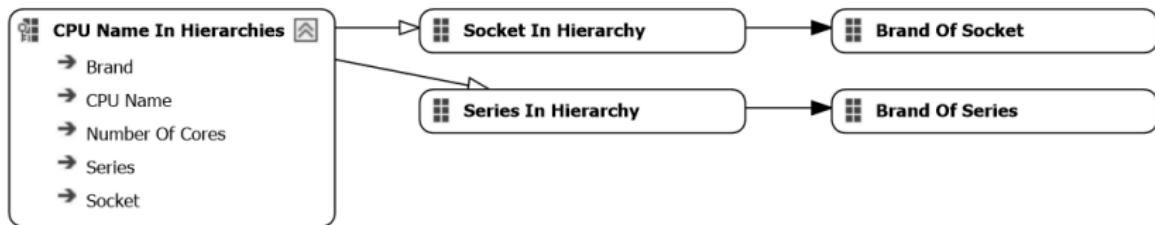
## Data Source View

- geography
- geo\_id
  - geo\_code
  - continent
  - country
  - region
  - currency



# CPU dimension

Attributes	Hierarchies	Data Source View								
<ul style="list-style-type: none"> <li><input checked="" type="checkbox"/> CPU           <ul style="list-style-type: none"> <li>Brand</li> <li>Brand Of Series</li> <li>Brand Of Socket</li> <li>CPU Name</li> <li>CPU Name In Hierarchies</li> <li>Number Of Cores</li> <li>Series</li> <li>Series In Hierarchy</li> <li>Socket</li> <li>Socket In Hierarchy</li> </ul> </li> </ul>	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <th style="text-align: center;">BrandSeriesCpu</th> <th style="text-align: center;"></th> </tr> <tr> <td style="padding: 5px;"> <ul style="list-style-type: none"> <li>▪ Brand</li> <li>▪▪ Series               <ul style="list-style-type: none"> <li>→ Brand Of Series</li> </ul> </li> <li>▪ CPU Name               <ul style="list-style-type: none"> <li>→ Brand</li> <li>→ CPU Name</li> <li>→ Number Of Cores</li> <li>→ Series</li> <li>→ Series In Hierarchy</li> <li>→ Socket</li> <li>→ Socket In Hierarchy</li> </ul> </li> </ul> </td> <td style="text-align: center; padding: 5px;"> </td> </tr> <tr> <th style="text-align: center;">BrandSocketCpu</th> <th style="text-align: center;"></th> </tr> <tr> <td style="padding: 5px;"> <ul style="list-style-type: none"> <li>▪ Brand</li> <li>▪▪ Socket               <ul style="list-style-type: none"> <li>→ Brand Of Socket</li> </ul> </li> <li>▪ CPU Name               <ul style="list-style-type: none"> <li>→ Brand</li> <li>→ CPU Name</li> <li>→ Number Of Cores</li> <li>→ Series</li> <li>→ Series In Hierarchy</li> <li>→ Socket</li> <li>→ Socket In Hierarchy</li> </ul> </li> </ul> </td> <td style="text-align: center; padding: 5px;"> </td> </tr> </table>	BrandSeriesCpu		<ul style="list-style-type: none"> <li>▪ Brand</li> <li>▪▪ Series               <ul style="list-style-type: none"> <li>→ Brand Of Series</li> </ul> </li> <li>▪ CPU Name               <ul style="list-style-type: none"> <li>→ Brand</li> <li>→ CPU Name</li> <li>→ Number Of Cores</li> <li>→ Series</li> <li>→ Series In Hierarchy</li> <li>→ Socket</li> <li>→ Socket In Hierarchy</li> </ul> </li> </ul>		BrandSocketCpu		<ul style="list-style-type: none"> <li>▪ Brand</li> <li>▪▪ Socket               <ul style="list-style-type: none"> <li>→ Brand Of Socket</li> </ul> </li> <li>▪ CPU Name               <ul style="list-style-type: none"> <li>→ Brand</li> <li>→ CPU Name</li> <li>→ Number Of Cores</li> <li>→ Series</li> <li>→ Series In Hierarchy</li> <li>→ Socket</li> <li>→ Socket In Hierarchy</li> </ul> </li> </ul>		<p>cpu_product</p> <ul style="list-style-type: none"> <li>cpu_id</li> <li>cpu_code</li> <li>brand</li> <li>series</li> <li>cpu_name</li> <li>n_cores</li> <li>socket</li> </ul>
BrandSeriesCpu										
<ul style="list-style-type: none"> <li>▪ Brand</li> <li>▪▪ Series               <ul style="list-style-type: none"> <li>→ Brand Of Series</li> </ul> </li> <li>▪ CPU Name               <ul style="list-style-type: none"> <li>→ Brand</li> <li>→ CPU Name</li> <li>→ Number Of Cores</li> <li>→ Series</li> <li>→ Series In Hierarchy</li> <li>→ Socket</li> <li>→ Socket In Hierarchy</li> </ul> </li> </ul>										
BrandSocketCpu										
<ul style="list-style-type: none"> <li>▪ Brand</li> <li>▪▪ Socket               <ul style="list-style-type: none"> <li>→ Brand Of Socket</li> </ul> </li> <li>▪ CPU Name               <ul style="list-style-type: none"> <li>→ Brand</li> <li>→ CPU Name</li> <li>→ Number Of Cores</li> <li>→ Series</li> <li>→ Series In Hierarchy</li> <li>→ Socket</li> <li>→ Socket In Hierarchy</li> </ul> </li> </ul>										



# Storage mode

The default storage mode was not modified: **pure MOLAP without proactive caching**.

# MDX query

## Analysis to perform with MDX

For each CPU brand, the percentage variation of profit (sales - cost) with respect to the previous year, for each year.

## Resolution method 1: PREV MEMBER

```
WITH MEMBER [% Profit Variation] AS
    ([Profit] - ([Time].[Year].PREV MEMBER, [Profit]))
        / ([Time].[Year].PREV MEMBER, [Profit]),
    FORMAT_STRING = 'Percent'
SELECT
    [% Profit Variation] ON 0,
    ([CPU].[Brand].[Brand], [Time].[Year].[Year]) ON 1
FROM [CPU Sales];
```

## Resolution method 2: LAG

```
WITH MEMBER [% Profit Variation] AS
    ([Profit] - ([Time].[Year].CURRENTMEMBER.LAG(1), [Profit]))
        / ([Time].[Year].CURRENTMEMBER.LAG(1), [Profit]),
    FORMAT_STRING = 'Percent'
SELECT
    [% Profit Variation] ON 0,
    ([CPU].[Brand].[Brand], [Time].[Year].[Year]) ON 1
FROM [CPU Sales];
```

## Resolution method 3: PARALLELPERIOD

WITH

```
MEMBER [Profit Prev Year] AS
(
    PARALLELPERIOD(
        [Time].[Calendar].[Year],
        1,
        [Time].[Calendar].CURRENTMEMBER
    ),
    [Profit]
)
MEMBER [% Profit Variation] AS
    ([Profit] - [Profit Prev Year]) / [Profit Prev Year],
    FORMAT_STRING = 'Percent'
SELECT
    [% Profit Variation] ON 0,
    ([CPU].[Brand].[Brand], [Time].[Calendar].[Year]) ON 1
FROM [CPU Sales];
```

# Power BI dashboard

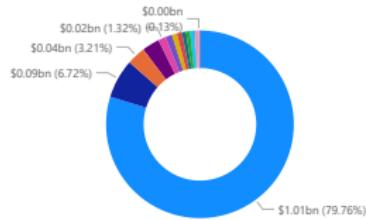
Brand, Series, CPU Name

- AMD
- INTEL

Gross Profit, Sales Usd and Cost by Continent, Country and Series



Sales Usd by Brand and Series



Brand Series

- INTEL Intel Xeon
- INTEL Intel Core i7
- INTEL Intel Core i5
- AMD Amd Opteron
- INTEL Intel Core i3
- INTEL Intel Core i9
- AMD Amd Ryzen
- AMD Amd Epyc

Sales Usd and Cost by Year and Quarter

