Università di Pisa

UNIVERSITÀ DI PISA

Department of Computer Science
MSc in Data Science & Business Informatics

GROUP_ID_778

LABORATORY OF DATA SCIENCE PROJECT

Michele Dicandia 657494

m.dicandia1@studenti.unipi.it

Academic year 2023/2024

CONTENTS

1. INTRODUZIONE

Il presente report documenta il processo e i risultati del progetto di analisi dati basato sul dataset fornito, composto principalmente dal file computer_sales.csv delle vendite di computer e dal file geography.xml contenente informazioni geografiche correlate alle vendite. Gli obiettivi principali del progetto includono la costruzione di un datawarehouse, lo sviluppo di un processo ETL, la creazione di un Cubo dei Dati, l'implementazione di query MDX e la realizzazione di una dashboard per la visualizzazione dei dati.

Il progetto si è proposto di raggiungere una serie di obiettivi chiave:

1. Costruzione di un Data Warehouse per la gestione efficiente dei dati.
2. Sviluppo di un processo ETL (Extract, Transform, Load) per l'elaborazione dei dati.
3. Creazione di un Cubo dei Dati per l'analisi multidimensionale.
4. Implementazione di query MDX per l'estrazione di informazioni rilevanti.
5. Realizzazione di una dashboard intuitiva per la visualizzazione dei dati.

Il dataset principale, rappresentato dal file computer sales.csv, fornisce una panoramica completa delle vendite di computer dal marzo 2013 all'aprile 2018, includendo dettagli sulle transazioni di vendita e sulle specifiche hardware dei PC venduti. L'aggiunta del file geography.xml arricchisce ulteriormente il dataset con informazioni geografiche pertinenti, le quali possono essere collegate alla tabella dei fatti principale mediante le rispettive chiavi primarie.

2. COSTRUZIONE DEL DATA WAREHOUSE

2.1 ASSEGNAZIONE 1

Per costruire il data warehouse seguendo lo star schema fornito nel file PDF di riferimento, sono state inizialmente create le varie tabelle su Sql Server Management Studio. Ogni tabella è stata definita specificando la natura degli attributi e stabilendo le appropriate relazioni con le chiavi primarie delle tabelle delle dimensioni tramite chiavi esterne nella tabella dei fatti. Successivamente, è stato creato un diagramma per visualizzare chiaramente i collegamenti dello schema a stella, come mostrato nella *Figura 1*.

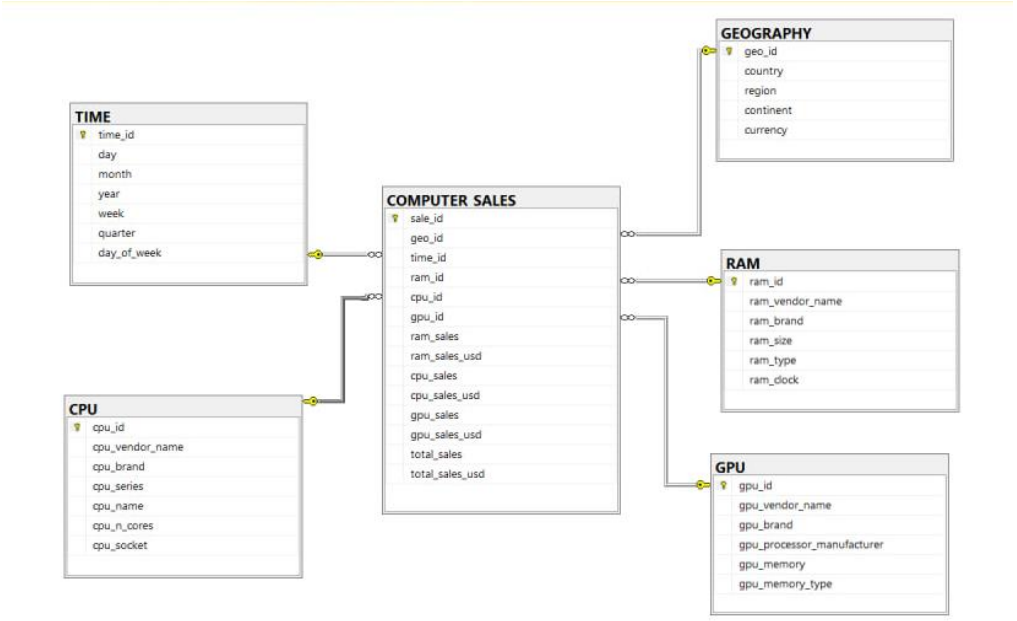


Figura 1

Per completare questa fase, è stato utilizzato Python per caricare i due file "computer_sales.csv" e "geography.xml". Sono state implementate alcune funzioni al fine di gestire e creare le tabelle dello star schema in modo più efficiente. Ad esempio, la funzione "gen_table_distinct" è stata utilizzata per estrarre valori unici dalle colonne specificate e restituirli come una tabella di valori distinti ordinati, la funzione "set_primary_key" ha generato chiavi primarie per le tabelle, o la funzione add_foreign_key che prendendo in input due data frame aggiunge una chiave esterna al primo data frame basandosi sui valori corrispondenti nelle colonne comuni tra il primo e il secondo data frame.

Il dataset "computer_sales" è composto da 3.412.325 righe e 25 colonne, e non presenta attributi con valori nulli.

Sono state create le tabelle "time_table", "cpu_table", "gpu_table", "ram_table", "geography_table" e "computer_sales_table".

Per la realizzazione della tabella "time_table", sono stati estratti i valori delle variabili 'day', 'month' e 'year' dalla stringa "time_code" nel formato 'yyyymmdd'. Inoltre, è stata derivata la variabile "quarter" dal mese utilizzando la funzione "get_quarter", mentre le variabili "week" e "day_of_week" sono state ottenute tramite la funzione "to_datetime" della libreria Pandas.

Successivamente, è stato creato un oggetto che funge da dizionario denominato "dict_sales" mediante la funzione "dict_from_header", dove le chiavi rappresentano i nomi delle colonne e i valori corrispondono alle colonne dei dati. Da questo dizionario sono state estratte le colonne rilevanti, alle quali è stata applicata la funzione "gen_distinct_value", seguita dalla funzione "set_primary_key" per generare chiavi univoche per le tuple della tabella.

Le tabelle "cpu_table", "gpu_table" e "ram_table" sono state create semplicemente estraendo le variabili di interesse dal dizionario "dict_sales" e applicando le funzioni per generare righe distinte e inserire le chiavi primarie.

Per la tabella "geography_table", è stato importato il secondo file "geography.xml" e unito al dataset principale tramite la chiave "geo_id". Il nuovo dataset è stato quindi trasformato in dizionario, e sono state applicate le medesime funzioni alle variabili di interesse. Inoltre, la natura della variabile "geo_id" è stata modificata in stringa per garantire una coerenza nel tipo di chiavi.

Per quanto riguarda la creazione della tabella "computer_sales", sono state introdotte nuove variabili come "cpu_sales_usd", "gpu_sales_usd", "ram_sales_usd" e "total_sales_usd", derivanti dalle rispettive variabili che identificavano le vendite nella valuta del paese di origine, attraverso dei coefficienti di cambio. L'obiettivo di queste nuove variabili è consentire il confronto delle vendite nella stessa valuta, ovvero con la stessa unità di misura. Successivamente, sono state mappate le chiavi esterne nell'intero dataset e, utilizzando le stesse funzioni, sono state estratte le variabili di interesse e creata la chiave primaria.

In conclusione, le tabelle create sono le seguenti:

- "time_table": 1840 righe e 7 colonne
- "cpu_table": 5321 righe e 7 colonne
- "gpu_table": 3159 righe e 6 colonne
- "ram_table": 11990 righe e 6 colonne
- "geography_table": 75 righe e 5 colonne
- "computer_sales": 3.381.715 righe e 14 colonne

.

2.2 ASSEGNAZIONE 2

Per la seconda fase dell'assegnazione, relativa al caricamento dei dati su Sql Server Management Studio, sono state importate le librerie Python necessarie, quali pyodbc, os, re e tqdm. È stata inoltre effettuata una conversione del valore "Mike’s computer shop" in "Mikes computer shop" negli attributi "sales_vendor_name", "gpu_vendor_name" e "cpu_vendor_name" a causa della presenza di apici nella stringa SQL, al fine di consentire l'inserimento dei dati nella tabella remota.

Successivamente, è stata creata la classe "Upload_table". Utilizzando le credenziali per l'accesso a Sql Server Management Studio, questa classe si è occupata di inserire i dati nelle tabelle, dopo aver eliminato eventuali contenuti preesistenti. È stata adottata una strategia di mini-batch, che ha comportato l'esecuzione di un commit periodico ogni 100 righe, contribuendo a ridurre il tempo complessivo di esecuzione dell'operazione di inserimento e minimizzando il rischio di perdita di dati in caso di errori.

3. PROCESSO ETL

In questo capitolo viene descritta la soluzione di una domanda aziendale mediante l’utilizzo di Sql Server Integration Services (SSIS) utilizzando Visual Studio, è stato sviluppato un workflow ETL per condurre un'analisi specifica

3.1 ASSEGNAZIONE 3

La domanda di business richiesta è: **“For each year and region, identify the computer IDs associated with the highest sales of CPUs. Furthermore, augment the result by including the percentage of sales w.r.t. to the total sales of all computers within the same CPU series.”**

Per affrontare la problematica assegnata, si è proceduto avviando l'ambiente di sviluppo Visual Studio e creando un progetto SSIS. All'interno del flusso di controllo (control flow) di tale progetto, è stata inserita l'operazione "data flow". All'interno di questa operazione, i dati sono stati estratti dalla tabella di origine mediante l'utilizzo della trasformazione "origine OLE DB", specificando la connessione al server e la tabella di provenienza. Questa trasformazione è stata successivamente denominata "caricamento COMPUTER SALES".

Successivamente, è stata applicata una procedura di ordinamento della tabella in base alla chiave "time_id" mediante la trasformazione di ordinamento. Tale operazione è stata ripetuta anche per la tabella "TIME". L'ordinamento era essenziale per l'applicazione della trasformazione di merge join tra le due tabelle, utilizzando la chiave "time_id". Durante questa fase, sono state incluse le variabili necessarie per le future analisi, ovvero "geo_id" e "cpu_id", le quali avrebbero rappresentato le chiavi di collegamento con altre tabelle. Sono state aggiunte "year" per l'analisi e "total_sales_usd" per il calcolo di "max_sales" e "percentage_sales", con esclusione di tutte le altre variabili non rilevanti ai fini dell'analisi.

Successivamente, è stata estratta la tabella "GEOGRAPHY" mediante un procedimento analogo, ordinando entrambe le tabelle secondo la variabile "geo_id" e unendo le due tabelle mediante merge join. Durante questa fase, sono state incluse le variabili "year", "cpu_id", "total_sales" e sono state aggiunte "region" per scopi analitici, con esclusione di "geo_id" poiché non più necessario.

Lo stesso approccio è stato seguito per l'estrazione della tabella "CPU", ordinando entrambe le tabelle secondo "cpu_id" e unendole mediante merge join, includendo le variabili "year", "region", "total_sales", "cpu_id" e "cpu_series".

Una volta ottenuta una tabella contenente tutte le variabili necessarie, è stata eseguita una trasformazione di raggruppamento per "year", "region", "cpu_series", "cpu_id" e "total_sales_usd" impostandolo come valore massimo. Ciò ha permesso di ottenere una tabella ridotta senza duplicati, passando da 3.381.715 a 5.399 righe.

Successivamente, è stata applicata una trasformazione di multicast per riutilizzare la stessa tabella in varie trasformazioni. Successivamente, è stata applicata un'altra trasformazione di raggruppamento, questa volta solo per "year", "region", "cpu_series" e "total_sales_usd" impostandolo come valore massimo e rinominando la colonna come "max_sales". In questo modo, sono stati identificati i "cpu_id" con le massime vendite per "year", "region" e "cpu_series".

Dopodiché, è stata eseguita un'altra trasformazione di raggruppamento per "year", "region", "cpu_series", calcolando la somma dei "total_sales_usd" e rinominandola come "sum_sales". Le tabelle sono state quindi ordinate secondo "year", "region", "cpu_series" e svolto in join, ottenendo così "max_sales" e "sum_sales" nella stessa tabella.

Utilizzando una trasformazione di colonna derivata, è stato calcolato il rapporto tra "max_sales" e "sum_sales", creando una nuova colonna denominata "sales_percentage". È stato quindi eseguito un merge join di questa nuova tabella con la tabella completa dal multicast, ordinata per "year", "region" e "cpu_series", identificando così il "cpu_id" per ogni combinazione di "year", "region" e "cpu_series", insieme alla relativa "max_sales" e "sales_percentage".

Infine, è stata eseguita un'ultima trasformazione di destinazione OLE DB, caricando la tabella risultante in una nuova tabella su Sql Server Management Studio, appositamente creata per ospitare i dati elaborati dal Data Flow in SSIS.

In *figura 2* è mostrato il codice sql per creare la tabella dove vengono esportati i dati dal workflow in visual studio

In *figura 3* è mostrato il Data flow della procedura effettuata con visual studio, visibile e riproducibile eliminando i dati nella tabella SSIS_table

```sql
--CREAZIONE TABELLA SSIS
CREATE TABLE SSIS_TABLE (
    year INT,
    region VARCHAR(255),
    cpu_series VARCHAR(255),
    cpu_id VARCHAR(255),
    max_sales DECIMAL(20,2),
    sales_percentage DECIMAL(20,2),
    PRIMARY KEY (year, region, cpu_series)
);
```

Figura 2

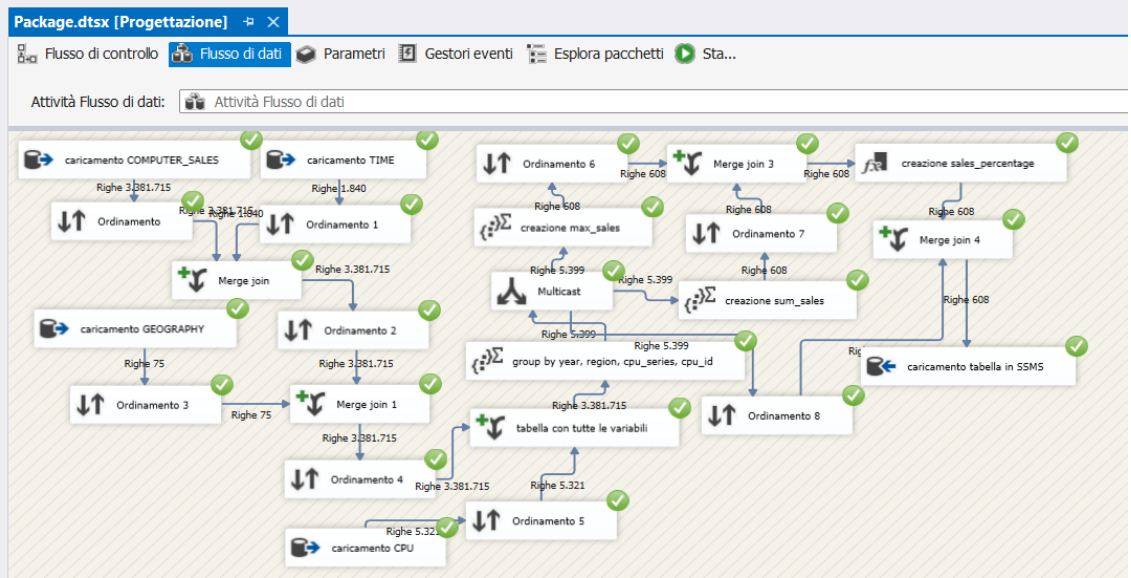


Figura 3

4. ANALISI DATI MULTIDIMENSIONALI

Questa sezione si focalizza sulla risoluzione di specifiche domande aziendali utilizzando un datacube che sarà generato a partire dal database già esistente. Per affrontare tali problematiche, ci avvarremo di MultiDimensional eXpression (MDX) all'interno di SQL Server Management Studio.

4.1 ASSEGNAZIONE 4

Il quarto compito implica la costruzione di un data cube partendo dalle tabelle dei dati e l'integrazione delle gerarchie appropriate. Il processo inizia creando un nuovo progetto di Analysis Services in Visual

Studio, dove si configura un'origine dati con le tabelle importate da SQL Server Management Studio. Successivamente si procede con la creazione del cubo, definendo le dimensioni necessarie con le relative gerarchie per rispondere alle domande aziendali.

Dopo l'importazione dei dati, si crea una vista origine dati per "Group_ID_778_DB", seguita dalla definizione delle dimensioni con le rispettive gerarchie. La dimensione CPU include (cpu_id → cpu_brand) [*figura 4*], la dimensione GPU include (gpu_id → gpu_brand) [*figura 5*], la dimensione RAM include (ram_id → ram_brand) [*figura 6*], mentre la dimensione GEOGRAPHY comprende (geo_id → region → country → continent) [*figura 7*]. Inoltre, la dimensione TIME viene arricchita con nuove variabili come "month_str", che combina il numero del mese con il suo nome (ad esempio, 01-January, 02-February, ...) e "day_str", ottenuta unendo il mese attuale in formato carattere con il giorno (ad esempio, January-16, March-24, ...). Queste variabili serviranno per creare una nuova gerarchia utile per l'analisi successiva tramite il cubo MDX SQL. È stata creata un ulteriore gerarchia con month_hier → year con l'attributo month_hier creato dall'unione tra la variabile year e la varaibile month (es. 2016-08, ...) che servirà per una futura analsi nell'assegnazione 6.

La dimensione TIME avrà quindi la seguente struttura: (time_id → day_str → month_str); (month_hier → year) [*figura 8*]. Non sono state create ulteriori gerarchie come time_id → day → week → month → quarter → year poiché considerate superflue per le esigenze analitiche.

Dopo la definizione delle dimensioni, si procede con la creazione del cubo, che include le misure come "ram_sales_usd", "cpu_sales_usd", "gpu_sales_usd" e le dimensioni TIME, GEOGRAPHY, CPU, GPU, RAM.

Infine, il progetto viene elaborato e caricato sul server, completando l'operazione con la compilazione della destinazione, specificando la stringa di connessione al server e il nome del database "GROUP_ID_778_DB", tramite il comando elabora.

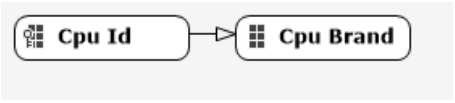


Figura 4

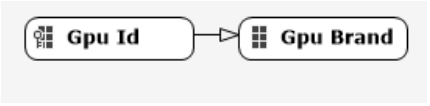


Figura 5

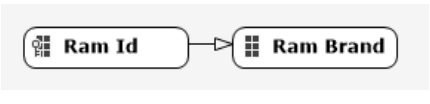


Figura 6

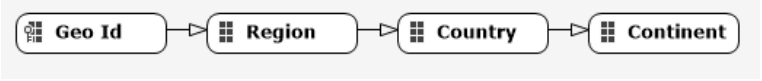


Figura 7

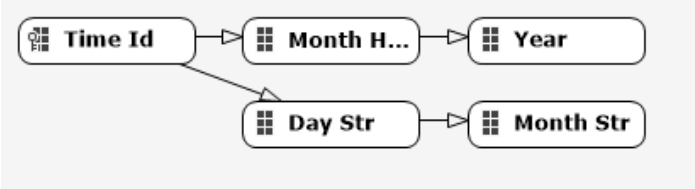


Figura 8

4.2 ASSEGNAZIONE 5

Una volta completata la progettazione del cubo, siamo stati in grado di rispondere ad alcune domande aziendali utilizzando SQL Server Analysis Services (SSAS) e MultiDimensional eXpressions (MDX) in SQL Server Management Studio.

La domanda aziendale era la seguente: **"Produce the following analysis using an MDX query: Show the top 5 cpu, ram, and gpu brands w.r.t the monthly average sales for each region in Europe."**

Per affrontare questa sfida, sono state sviluppate tre diverse query MDX:

1. Trovare la vendita media mensile dei primi 5 marchi di CPU per ogni regione in Europa.
2. Trovare la vendita media mensile dei primi 5 marchi di GPU per ogni regione in Europa.
3. Trovare la vendita media mensile dei primi 5 marchi di RAM per ogni regione in Europa.

Il primo problema è stato risolto con la seguente query:

```
WITH
  MEMBER [Measures].[AvgMonthlyCPUSales] AS
    Avg(
      Descendants(
         [TIME].[Gerarchia].CurrentMember,
         [TIME].[Gerarchia].[Month Str]
      ),
       [Measures].[Cpu Sales Usd]
   )
SELECT
   [Measures].[AvgMonthlyCPUSales] ON COLUMNS,
   NONEMPTY(
      GENERATE(
         ([TIME].[Month Str].[Month Str], [GEOGRAPHY].[Region].[Region]),
         TOPCOUNT(
            ([TIME].[Month Str].CURRENTMEMBER,
                    [GEOGRAPHY].[Region].CURRENTMEMBER,
                    [CPU].[Cpu Brand].[Cpu Brand]),
            5,
            [Measures].[AvgMonthlyCPUSales]
         )
      )
   )ON ROWS
FROM [Group ID 778 DB]
WHERE [GEOGRAPHY].[Gerarchia].[Continent].&[Europe]
```

Inizialmente, è stata calcolata la misura [AvgMonthlyCPUSales] nel blocco "WITH" utilizzando la funzione AVG per determinare la media mensile delle vendite di CPU ([Cpu Sales Usd]) su tutti i mesi.

Successivamente, è stata selezionata la misura calcolata [AvgMonthlyCPUSales] sull'asse delle colonne.

Utilizzando la funzione GENERATE, è stato creato un insieme di tuple combinando gli elementi delle gerarchie [TIME].[Month Str] e [GEOGRAPHY].[Region].[Region].

La funzione TOPCOUNT ha restituito i primi 5 membri della gerarchia [CPU].[Cpu Brand].[Cpu Brand] in base alla misura [AvgMonthlyCPUSales].

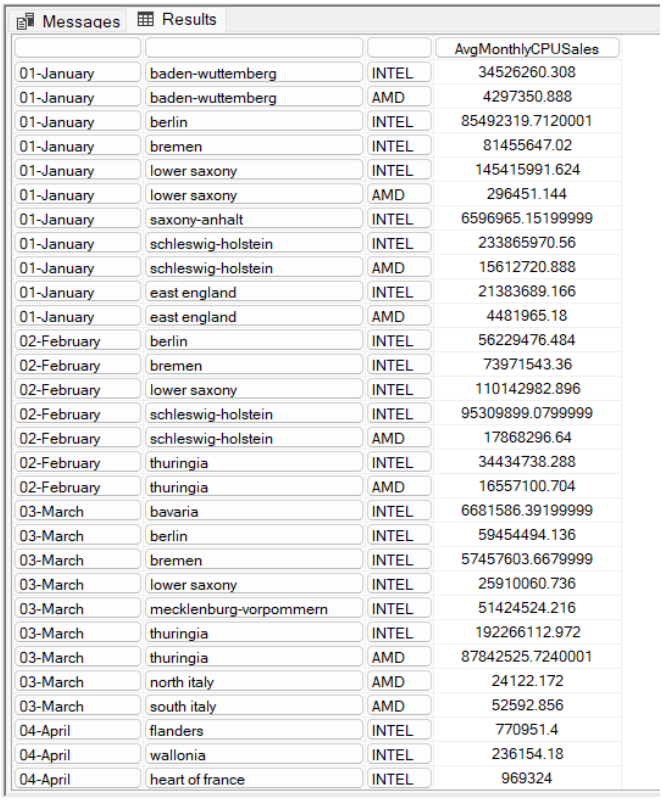
Per assicurare di filtrare solo le tuple valide, ovvero quelle con una misura non nulla, è stata utilizzata la funzione NONEMPTY.

In sintesi, il secondo blocco di codice ha restituito i top 5 cpu_brand per ogni mese e per ogni regione sulle righe.

Il blocco 'FROM [Group ID 778 DB]' indica la provenienza delle misure e delle dimensioni.

Infine, il blocco di codice 'WHERE [GEOGRAPHY].[Hierarchy].[Continent].&[Europe]' limita la selezione della gerarchia ai soli valori appartenenti al continente Europa..

In *figura 9* è mostrato il riultato di questa prima query.



Messages | Results

			AvgMonthlyCPUSales
01-January	baden-wuttemberg	INTEL	34526260.308
01-January	baden-wuttemberg	AMD	4297350.888
01-January	berlin	INTEL	85492319.7120001
01-January	bremen	INTEL	81455647.02
01-January	lower saxony	INTEL	145415991.624
01-January	lower saxony	AMD	296451.144
01-January	saxony-anhalt	INTEL	6596965.15199999
01-January	schleswig-holstein	INTEL	233865970.56
01-January	schleswig-holstein	AMD	15612720.888
01-January	east england	INTEL	21383689.166
01-January	east england	AMD	4481965.18
02-February	berlin	INTEL	56229476.484
02-February	bremen	INTEL	73971543.36
02-February	lower saxony	INTEL	110142982.896
02-February	schleswig-holstein	INTEL	95309899.0799999
02-February	schleswig-holstein	AMD	17868296.64
02-February	thuringia	INTEL	34434738.288
02-February	thuringia	AMD	16557100.704
03-March	bavaria	INTEL	6681586.39199999
03-March	berlin	INTEL	59454494.136
03-March	bremen	INTEL	57457603.6679999
03-March	lower saxony	INTEL	25910060.736
03-March	mecklenburg-vorpommern	INTEL	51424524.216
03-March	thuringia	INTEL	192266112.972
03-March	thuringia	AMD	87842525.7240001
03-March	north italy	AMD	24122.172
03-March	south italy	AMD	52592.856
04-April	flanders	INTEL	770951.4
04-April	wallonia	INTEL	236154.18
04-April	heart of france	INTEL	969324

ù

Figura 9

La tabella nella Figura 9 presenta il risultato della query descritta in precedenza, mostrando i top 5 cpu_brand con le rispettive medie di vendita per ogni combinazione di mese e regioni europee. Poiché le modalità della variabile cpu_brand erano solo "Intel" e "AMD", la tabella mostra solo queste due categorie.

Ad esempio, nel mese di gennaio, nella regione Baden-Württemberg, la categoria Intel ha registrato una media di vendite pari a 34,526,260.38 USD. Allo stesso modo, nel mese di marzo, nella regione Nord Italia, la categoria AMD ha registrato una media di vendite pari a 24,122.72 USD.

Per la selezione dei top 5 brand di GPU e RAM, è stato utilizzato lo stesso approccio utilizzato per i CPU. Sono state utilizzate le misure gpu_sales_usd e ram_sales_usd, calcolando la media delle vendite su base mensile, e rinominandola rispettivamente AvgMonthlyGPUSales e AvgMonthlyRAMSales. Nel codice, è stata sostituita la gerarchia [CPU].[Cpu Brand].[Cpu Brand] con [GPU].[Gpu Brand].[Gpu Brand] per i top 5 brand di GPU e [RAM].[Ram Brand].[Ram Brand] per i top 5 brand di RAM.

4.3 ASSEGNZIONE 6

Per analizzare i dati disponibili nel cubo, è stato utilizzato Power BI per creare un plot/dashboard significativo. I dati sono stati recuperati dal cubo tramite la stringa di connessione al server. Un'analisi importante è stata focalizzata sulle vendite totali per continente, rivelando che il 93,58% delle vendite complessive (tra CPU, GPU e RAM) proviene dall'Europa. Successivamente, è stata ampliata la gerarchia per visualizzare le vendite totali per ogni paese del mondo. È emerso che il 77,85% delle vendite complessive proviene dalla Germania.

In *figura 10* sono mostrati i due grafici: il primo grafico ad anello presenta le vendite totali per continente, il secondo grafico a torta presenta la vendita totale per paese.

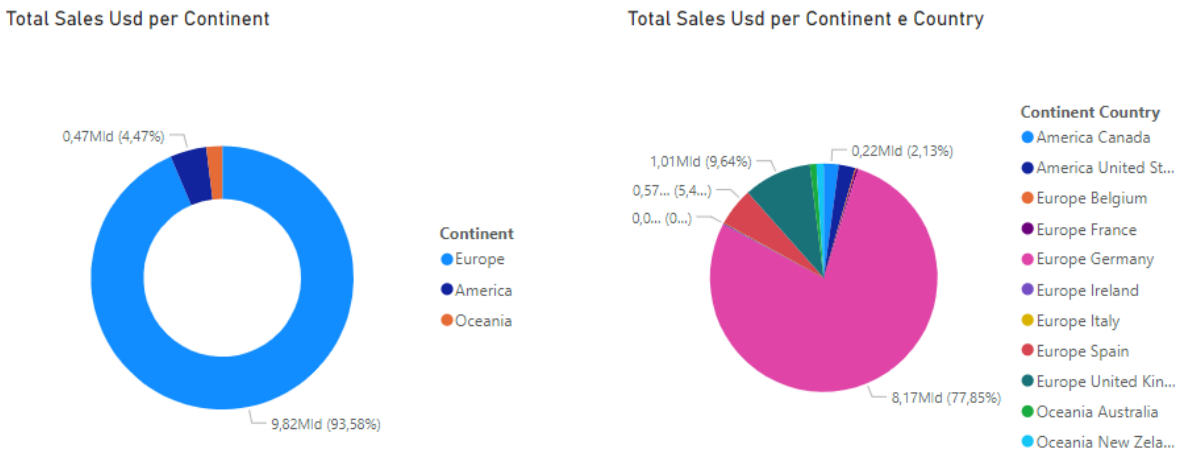


Figura 10

Un'altra analisi importante è stata quella di valutare la proporzione delle vendite totali destinate alle CPU, GPU e RAM. Attraverso un grafico a torta, rappresentato in *figura 11*, è stata evidenziata la percentuale totale di vendite per ciascuna categoria. È emerso che la vendita destinata alle CPU prevale, occupando il 64,49% delle vendite totali, seguita dalla GPU con il 29,52% delle vendite totali.

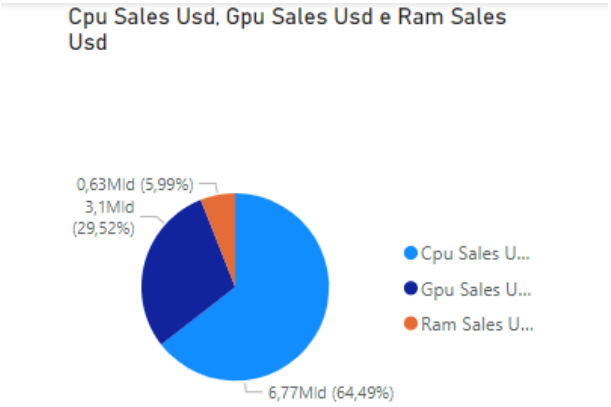


Figura 11

Successivamente, sono presentate le vendite di CPU, GPU e RAM per ciascun brand attraverso tre istogrammi nella *figura 12*.

Il primo istogramma mostra che circa 6,41 miliardi di USD sono destinati alla vendita delle CPU Intel, mentre solo 0,35 miliardi di USD sono destinati alla vendita delle CPU AMD.

Il secondo istogramma rivela che la maggior parte delle vendite di GPU proviene dalla GPU di tipo PNY, con esattamente 1,63 miliardi di USD.

Il terzo istogramma illustra le vendite di RAM per brand, dove circa 158 milioni di USD delle vendite di RAM provengono dalla vendita della RAM G.Skill, seguita da Kingston con 145 milioni di USD.

.

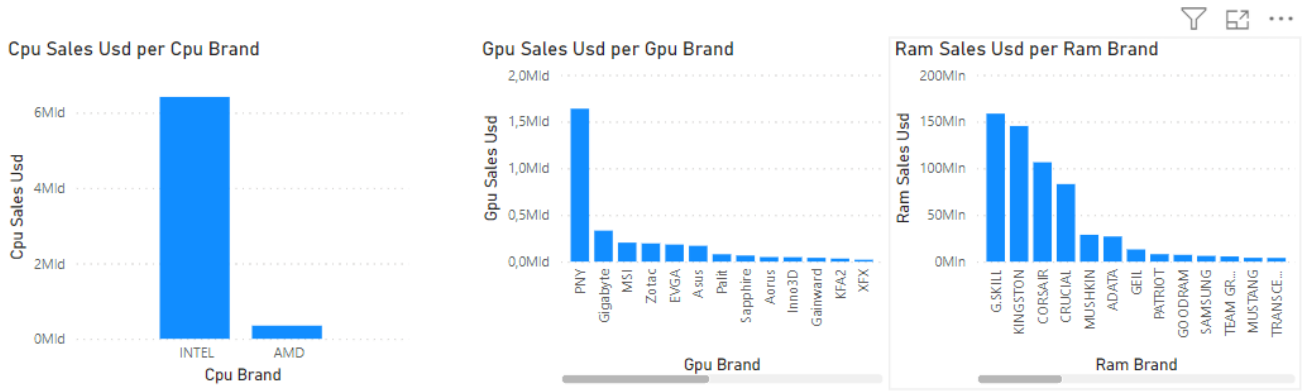


Figura 12

Infine, un'ultima analisi riguarda la vendita totale nel tempo, utilizzando un time plot dove sull'asse delle ascisse abbiamo la nuova variabile creata month_hier e sulle ordinate la vendita totale, come mostrato in *figura 13*.

Si nota che da marzo 2013 fino ad aprile 2018 c'è stato un trend positivo delle vendite totali, soprattutto con un'esplosione delle vendite da marzo 2016 fino ad aprile 2017, dove la media passa da 150 milioni di USD a 320 milioni di USD per questo intervallo di tempo. Successivamente, si osserva

un drastico calo dalle vendite a partire da maggio 2017 in poi con una media di 170 milioni di USD per il periodo da maggio 2017 ad aprile 2018.

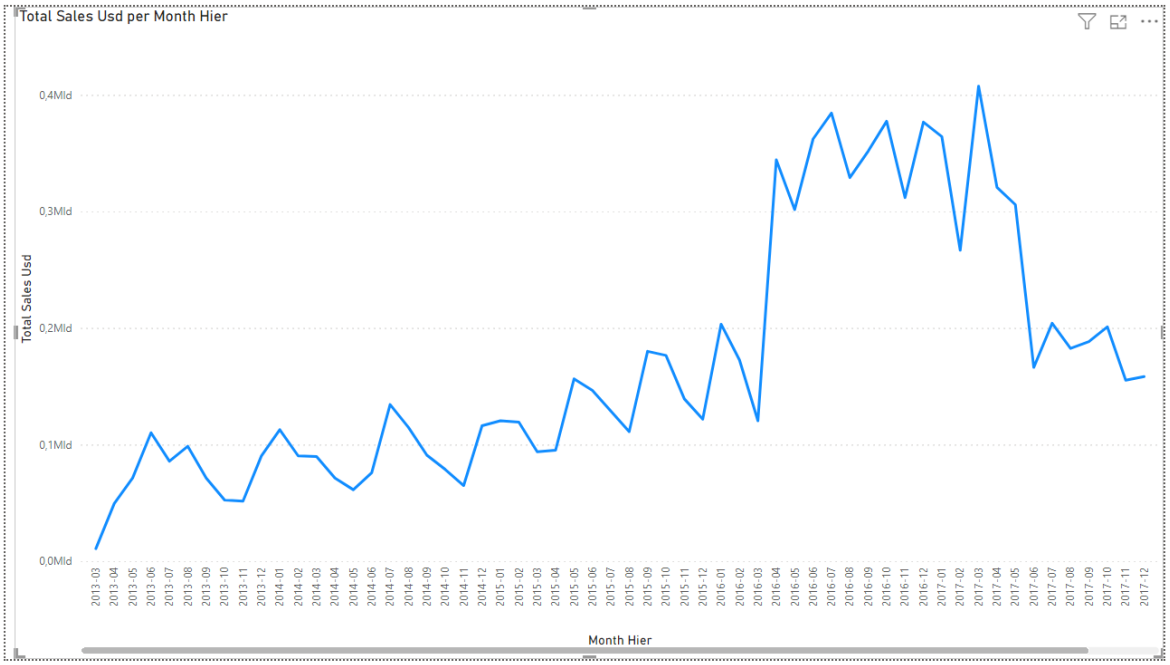


Figura 13