UNIVERSITÀ DI PISA

Laboratory of Data Science Report

Federica Di Pasquale (493195)
email: f.dipasquale1@studenti.unipi.it

Ferri Lorenzo (607828)
email: l.ferri11@studenti.unipi.it

Anno accademico 2020/2021

Indice

1 Part 1

1.1 Assignment 0

Il Constellation Schema assegnato è stato riprodotto mediante la query presente nel file "01_DB_creation.sql". Valori nulli sono ammessi solo per gli attributi diversi dalle Chiavi Primarie presenti nelle Dimension Tables. Nelle tre Fact Tables non sono ammessi valori nulli per le Chiavi Esterne, al fine di non perdere ennuple a seguito di uno Star-Join.
I valori della Chiave Primaria della Dimension Table Time, "time_code", sono stati trasformati nel formato yyyy-mm-dd per renderli compatibili con il Data Type "Date" di SQL Server. Sono stati imposti vincoli di Chiave Primaria nelle Dimension Tables e di Chiave Esterna nelle Fact Tables, dunque i file vengono forniti nell'ordine in cui devono essere eseguiti per riempire prima le Dimensions e poi i Facts.

1.2 Assignment 1

Il file "fact.csv" contenente l'intera Fact Table, è stato suddiviso in tre Fact Tables separate, ciascuna per ogni linea di prodotto (cpu, gpu, ram), tramite il file "04_fact.py".
Date le dimensioni complessive della Fact Table, si è scelto di non creare tre file ".csv" ma di suddividere e caricare immediatamente il loro contenuto nel Constellation Schema.
A tale scopo è stato fatto uno scan dell'intero file durante il quale si individua per ogni record la Fact Table corrispondente; si è scelto di non caricare nel DW un record per volta, ma di salvare il loro contenuto in tre liste separate e solo al termine effettuare tre chiamate "executemany(sql, *params)".
Tale scelta si è rivelata molto più efficiente rispetto all'esecuzione di "execute(sql, *params)" effettuata su ogni singolo record; è necessario tuttavia notare che è stato possibile procedere in questo modo solo perché le dimensioni della Fact Table in esame non sono così elevate, mentre in generale sarebbe necessario bilanciare il contenuto caricato in memoria e le chiamate al DB.

1.3 Assignment 2

Le Dimension Tables sono state riempite con i due file: "02_dimensions.py", per tutte le dimensioni tranne Time, e "03_time_dimension.py", per la sola dimensione Time.

- **Dimensions:**
  Una volta effettuata la connessione al DB e definita una query parametrica, sono stati letti i file .csv contenenti le Dimension Tables tramite la funzione "reader" della libreria standard "csv", caricando uno per volta i record così ottenuti.

- **Time Dimension:**
  Sono state preparate le funzioni getQuarter(month) e getDayOfWeek(day, month, year), che ottengono rispettivamente il quarter dato il mese e il giorno della settimana data la

data del giorno. La seconda funzione ha l'obiettivo di calcolare il resto della divisione di x/7, dove x è dato dalla formula:

$$x = year + (year - 1)//4 - (year - 1)//100 + (year - 1)//400 + t \tag{1}$$

questo resto sarà l'indice del giorno della settimana partendo da 'Sabato'. Per ottenere x c'è bisogno di calcolare t, cioè il numero di giorni trascorsi dall'inizio dell'anno. Per farlo si è diviso il calcolo in:

$$t = daysToMonth(month, year) + day \tag{2}$$

dove la funzione daysToMonth(month, year) è una recursione che, dato il mese e l'anno (che serve per verificare se l'anno è bisestile), calcola il numero di giorni trascorsi fino all'inizio del mese corrente sommando i giorni dei mesi a ritroso lungo l'anno.

Il riempimento della tabella Time è avvenuto leggendo il contenuto del file "time.csv" e assegnandolo ad una Dictionary con la funzione DictReader(). Dopo essersi connessi al DB ed aver definito la query parametrica di inserzione, si è iterato sulle righe presenti nella Dictionary per ottenere i valori da passare ai parametri della query che ha riemepito il DB, applicandovi prima le funzioni sopra citate quando necessario.

2 Part 2

Si vuole rispondere ad alcune *business questions* utilizzando *Sequel Server Integration Services* (SSIS) e sviluppando delle soluzioni che calcolino i risultati dal lato client. In particolare è stata sviluppata un' unica soluzione contenente tre progetti relativi alle tre *business questions* assegnate.

2.1 Assignement 0

Business Question: For every year, the brand of gpu ordered by sales.

Tramite il nodo "Origine OLE DB" sono state lette solo le colonne necessarie della tabella Gpu_sales (gpu_code, time_code, sales_usd). Con il nodo "Ricerca" è stata operata la giunzione (tramite indice) con la colonna "brand" della tabella Gpu_product. Anziché operare una nuova ricerca per l'attributo "year" nella tabella Time, si è sfruttato il nodo "Colonna derivata" con cui si è estratto l'anno dal "time_code": quest'operazione permette un risparmio computazionale ed è stata resa possibile dalla trasformazione dell'attributo time_code effettuata al momento del popolamento della DW. E' stato poi eseguito il raggruppamento per gli attributi "year" e "brand", calcolando la funzione di aggregazione SUM(sales_usd) con nome "tot_sales_usd". Infine si è ordinato il risultato per (year, tot_sales_usd), in modo da avere per ogni anno la somma dei sales ordinata, come da richiesta. Il risultato finale viene scritto su un file di testo.

2.2 Assignement 1

Business Question: For any given country, a product is said to have full regional spread if it was sold in all the regions of that country. List all the AMD brand cpus that do not have full regional spread in Germany.

Per rispondere a questa domanda, è necessario accedere ai dati presenti nel DW ed in particolare alle tre tabelle Cpu_product, Cpu_Sales e Geography.
Inizialmente è stato fatto un accesso alla tabella Cpu_Sales tramite un nodo "Origine OLE DB" selezionando solo le colonne necessarie. Successivamente, tramite un nodo "Ricerca" è stata recuperata la colonna relativa al "brand" delle Cpu dalla tabella Cpu_Product. In questo modo è stato possibile selezionare solo le cpu relative al brand "AMD" attraverso un nodo "Conditional Split".
Dal momento che si cercano le cpu che non hanno "full regional spread" è stato necessario recuperare anche le informazioni relative al luogo di vendita. A questo scopo, tramite un ulteriore nodo "Ricerca", sono state recuperate le colonne "Country" e "Region" dalla tabella Geography (utilizzando geo_code) e sono state selezionate solo le cpu vendute nella nazione Germania. Successivamente il flusso di dati è stato suddiviso tramite un nodo "Multicast" in modo da poter calcolare nei due branch rispettivamente il numero di regioni distinte per ogni nazione e, per ogni cpu e nazione, il numero di regioni distinte in cui è stata venduta tale cpu. Una volta che sono state calcolate tutte le informazioni necessarie, è stato possibile selezionare le cpu che non hanno "full regional spread" facendo una Join fra i risultati dei due flussi di dati precedenti (preventivamente ordinati rispetto a "Country") e infine selezionando solo le cpu per cui il numero di regioni distinte in cui sono state vendute è minore del numero totale di regioni distinte per nazione. Infine il risultato è stato scritto in un file tramite un nodo "Destinazione file flat".
Nel caso specifico di questa Business question, l'ultimo ordinamento rispetto a "Country" risulta superfluo, in quanto è stata precedentemente selezionata solo la regione Germania; tuttavia si è preferito costruire una soluzione più generale che funzioni anche nel caso in cui non ci si voglia restringere ad una sola nazione.

2.3 Assignement 2

Business Question: Calculate which processor manufacturer yields the most sales, for each country and year.

Analogamente ai casi visti fin quì, dalla Fact Table "Gpu_sales" si ottengono gli attributi geo_code, time_code, gpu_code e sales_usd. country e processor_manufacturer si ottengono mediante una ricerca con Indice, mentre year si ottiene di nuovo con una funzione applicata su time_code. A questo punto sono necessarie due aggregazioni successive: una per (country, year, processor_manufacturer), per calcolare la somma dei sales; e un'altra per (country, year), in modo da ottenere il valore massimo della somma dei sales già calcolata, per ogni country e year. La prima aggregazione viene calcolata subito e dopo di essa la tabella viene duplicata con un nodo di "Multicast". In questo modo la seconda aggregazione può essere prodotta solo su una delle due tabelle identiche, lasciando l'altra invariata. Le

due tabelle vengono poi ricongiunte con un nodo "Merge join" su uguaglianza degli attributi country e year (previo ordinamento per tali attributi). La tabella risultante conterrà sia la somma dei sales per l'aggregazione (country, year, processor_manufacturer) che il valore della somma dei sales massima ripetuto per ogni country e year distinti. Basterà quindi imporre un'uguaglianza tra queste due metriche per ottenere solo le ennuple dove la somma dei sales è effettivamente pari a quella massima. Il risultato viene scritto su file di testo.