Alma Mater Studiorum · Università di Bologna

FACOLTÀ DI SCIENZE MATEMATICHE, FISICHE E NATURALI
Corso di Laurea in Informatica

Cloud Database: Google BigQuery

Tesi di Laurea in Basi di Dati

Relatore:
Chiar.mo Prof.
DANILO MONTESI

Presentata da:
MICHEL BUTERA

Sessione I
Anno Accademico 2010/2011

Indice

Introduzione

I dati sono l'anima del commercio. Garantire che i dati siano protetti, disponibili e facilmente accessibili sono esigenze fondamentali di qualsiasi reparto IT. Fattore più importante, garantire che i dati vengano utilizzati nel modo corretto (per gestire i processi, per informare i decision maker e intervenire in modo intelligente a circostanze mutevoli).

Il modo in cui le aziende garantiscono la disponibilità dei dati è in rapido mutamento.

Il Cloud computing ha fatto registrare una crescita impressionante negli ultimi anni, sia come concetto che come componente pratico dell'infrastruttura IT.

Una soluzione di Cloud computing particolarmente interessante è Google Storage.

Google Storage è un nuovo servizio per gli sviluppatori per archiviare e accedere ai dati nella cloud di Google.

Google Storage è stato presentato alla conferenza Google I/O 2010 assieme a Google BigQuery, un servizio web pensato per permettere di eseguire query su set di dati di grandi dimensioni, ad esempio è in grado di eseguire query di selezione e di aggregazione su tabelle con miliardi di record in pochi secondi, sarebbe quindi un buon passo in avanti per ottenere in modo interattivo informazioni che prima impiegavano anche giorni per essere calcolate.

In questa sede presenterò Google BigQuery e Google Storage, descriverò i passi per creare o trasferire un proprio database in Google Storage e per gestirlo attraverso Google BigQuery.

A questo scopo verranno illustrati anche alcuni strumenti, come GSUtil per la gestione dei dati su Google Storage e BQ per tutto quello che riguarda BigQuery.

Capitolo 1

Cos'è BigQuery

Con questo strumento Google vuole mettersi in primo piano sugli altri servizi di Cloud Computing permettendo ai propri utenti di ottenere informazioni dai propri dati in maniera più veloce. Lo scopo è quello di poter eseguire query su tabelle con un set di dati di miliardi di record con tempi di risposta di qualche secondo.

Essendo BigQuery strettamente collegato a Google Storage, questa può essere una buona scelta se si ha necessità di condividere e di elaborare dati molto velocemente.

Le principali caratteristiche di questo strumento sono:

- Scalabilità: uno dei vantaggi intrinseci del Cloud computing è la capacità di ampliare l'infrastruttura su richiesta, garantendo una scalabilità dinamica della capacità applicativa in base all'aumento delle esigenze. Tale aspetto si rivela particolarmente utile quando il livello di utilizzo di picco delle applicazioni ospitate cambia in modo consistente con il trascorrere del tempo (ad esempio, le applicazioni utilizzate nel settore della vendita al dettaglio durante i periodi di festività e così via).

- Interattività: riesce ad eseguire query di selezione o di raggruppamento su miliardi di record in pochi secondi.

- Familiarità: utilizza un dialetto SQL per la scrittura delle query.

Permette inoltre un'ottima condivisione dei dati, l'utilizzo di Google Storage consente di creare un hub di collaborazione. Ogniqualvolta si presentasse la necessità di condividere i propri dati con altri utenti è possibile dare l'accesso alle informazioni disponibili a chi si vuole impostando opportunamente gli ACL.

BigQuery contiene metodi che permettono sia di creare, popolare e cancellare tabelle, sia di eseguire query sopra di esse.

La scrittura delle query in BigQuery è possibile utilizzando un dialetto SQL, in questo dialetto sono stati modificati alcuni metodi SQL per rendere più veloce l'esecuzione di

alcune query, in quei casi in cui non è indispensabile la precisione dei risultati, si basano infatti su stime statistiche e restituiscono un valore indicativo.

Sia Google BigQuery che Google Storage sono in anteprima e sono aperti ad un numero limitato di imprese e sviluppatori. È quindi necessario avere almeno un account Google (ad esempio gmail), registrarsi alla lista d'attesa sia per Google Storage che per BigQuery, ed aspettare l'invito che si riceve per e-mail.

1.1 Utilità

In qualunque campo commerciale si ha che fare con grandi set di dati che con l'avanzare del tempo continuano ad aumentare in dimensione e complessità. Sono questi a fare il business delle aziende e il meglio che si può chiedere è di riuscire ad analizzarli nel minor tempo possibile. Google quindi mette a disposizione le sue infrastrutture hardware e software per tutte quelle aziende e privati che hanno bisogno di una grande potenza di calcolo, ma non vogliono comprare e gestire costosi macchinari o creare reti complesse, e lo fa attraverso Google Storage per la gestione dei dati e Google BigQuery per la gestione del database.

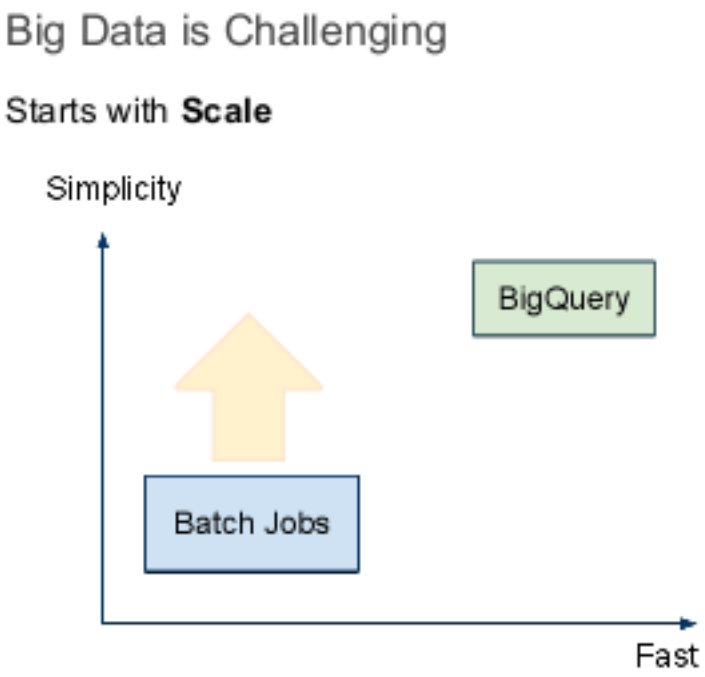

Figura 1.1: BigQuery (semplicità & velocità)

I casi di utilizzo possono essere svariati: dallo sviluppo di tool interattivi, controllo dello spam, rilevamento di trend, presentazioni via web, ottimizzazioni delle reti ecc..

Per comprendere meglio come questo prodotto ci possa essere utile per la sua velocità nell'analizzare set di dati enormi può essere quello di pensare di voler testare alcuni parametri di una rete di computer, effettuando dei ping fra le macchine e registrando tutto quello che avviene, alla fine del test ci ritroveremmo con miliardi di record da analizzare, ed è proprio in un caso come questo che potremmo aver bisogno di uno strumento potente come BigQuery, sempre che non si voglia lanciare query ed aspettare giorni per ottenere un risultato.

Capitolo 2

Concetti e terminologia di Google Storage

Google Storage permette di salvare, condividere e gestire i dati nello storage di Google.

È possibile utilizzare Google Storage per gestire qualsiasi tipo di dato di qualsiasi dimensione.

Per utilizzare al meglio questo strumento bisogna conoscere i concetti sui quali è costruito, questi concetti definiscono come vengono salvati i nostri dati in Google Storage.

2.1 Bucket

I bucket sono i contenitori che contengono i dati.

Tutto ciò che viene salvato in Google Storage deve essere contenuto all'interno di un bucket, corrispondono alle cartelle, è possibile utilizzarli per organizzare e controllare l'accesso ai dati, ma a differenza delle cartelle, non è possibile creare sotto-bucket.

2.2 Oggetti

Gli oggetti sono i singoli dati che vengono salvati in Google Storage.

Possono essere file di qualsiasi tipo, estensione e dimensione, sono considerati oggetti anche le tabelle create con BigQuery.

2.2.1 Immutabilità dei dati

Gli oggetti sono immutabili, il che significa che un oggetto non può essere modificato direttamente in Google Storage.

Non è possibile effettuare nessun tipo di modifica al contenuto di un oggetto, se si vuole modificare un oggetto memorizzato in Google Storage lo si può solo sostituire, se

non si vuole eliminarlo e ricaricarlo è possibile fare la stessa cosa con un'operazione sola sovra-scrivendolo.

2.3 Gerarchia

Google Storage utilizza una struttura gerarchica piatta per memorizzare bucket e oggetti. Tutti i bucket risiedono in quest'unico ambiente, e tutti gli oggetti si trovano anch'essi all'interno di una gerarchia piatta all'interno di un certo bucket.

2.3.1 NameSpace

Proprio per il fatto che Google Storage utilizza una gerarchia piatta, esiste un solo NameSpace, questo significa che ogni bucket deve avere un nome univoco, quindi non possiamo utilizzare un nome che è già stato scelto da un altro utente.

Mentre gli oggetti devono avere un nome univoco solo all'interno di un dato bucket, possiamo quindi avere più bucket che contengono sempre un oggetto con lo stesso nome.

2.4 Nomi degli oggetti

I nomi degli oggetti possono contenere una combinazione di caratteri Unicode (codifica UTF-8) con una lunghezza minore di 1024 byte.

Per eliminare le limitazioni date dalla gerarchia piatta è quella di utilizzare il carattere slash (/) nei nomi degli oggetti.

Ad esempio, si potrebbe nominare un oggetto /europe/france/paris.jpg e un altro /europe/italy/rome.jpg.

In questo modo è possibile organizzare gli oggetti come se fossere all'interno di directory e gestirli come tali (se ad volessimo visualizzare solo le immagini francesi potremmo usare il comando: gsutil ls gs://europe/france/).

Tuttavia, Google Storage vede gli oggetti come oggetti indipendenti che non hanno alcun rapporto gerarchico.

2.5 Nomi dei bucket

I nomi dei bucket hanno più restrizioni dei nomi degli oggetti, perché tutti i bucket risiedono nell'unico namespace di Google Storage.

I nomi dei bucket devono rispettare queste regole:

- devono essere lunghi da 3 a 63 caratteri.
- devono iniziare e terminare con un numero o una lettera.
- devono contenere solo lettere minuscole, numeri, punti e trattini (.) (-).
- non possono essere rappresentati come un indirizzo IP in notazione decimale puntata (ad esempio 192.168.5.4).
- non possono iniziare con il prefisso "GOOG".
- non possono contenere 2 punti adiacenti o un trattino prima o dopo di un punto. Ad esempio, ".." o "-." o ".-" non sono accettabili.

Capitolo 3

Requisiti di utilizzo

Per poter iniziare ad utilizzare BigQuery bisogna conoscere ed installare nel proprio computer gli strumenti necessari per gestire Google Storage e per poter eseguire le query sulle tabelle.

3.1 GSUtil

GSUtil è un'applicazione Python che permette di accedere a Google Storage da linea di comando.

GSUtil gira su sistemi operativi UNIX-based, come Linux o Mac OS X, e sui sistemi Windows. Per utilizzare GSUtil, è necessario avere una versione di Python 2.5.x o superiore, installata nel computer. Python è installato di default nella maggior parte delle distribuzioni di Linux e Mac OS X, mentre non lo è nei sistemi operativi Windows.

Questo strumento risulta molto semplice, se si ha familiarità con la shell dei sistemi operativi Unix-based, questo perchè i comandi sono gli stessi che si utilizzano in questa shell per la gestione dei file.

Proprio come nella shell è possibile utilizzare i caratteri jolly, che ci consentono una gestione veloce e pratica dei file nella sintassi di un comando.

3.1.1 Autenticazione a Google Storage

La prima volta che si prova ad eseguire gsutil, ci verrà segnalato che il file di configurazione non contiene le credenziali di accesso.

Le credenziali di accesso sono formate da una chiave di accesso e una "password".

La chiave d'accesso è una stringa alfanumerica di 20 caratteri, ed è collegata al proprio account Google.

Un esempio di chiave d'accesso è il seguente: GOOGTS7C7FUP3AIRVJTE.

La password invece è una stringa di 40 caratteri ed è collegata ad una specifica chiave d'accesso.

È necessario utilizzare questa password per firmare tutte le richieste come parte del processo di autenticazione (dopo il primo accesso viene immessa automaticamente).

Un esempio di password è:
bGoa+V7g/yqDXvKRqq+JTFn4uQZbPiQJo4pf9RzJ

È possibile creare e gestire queste chiavi di sviluppo utilizzando il gestore delle chiavi di Google Storage raggiungibile all'indirizzo: https://sandbox.google.com/storage/m/manage, che in modo molto semplice (premendo un pulsante) ci assegna automaticamente una chiave di sviluppo.

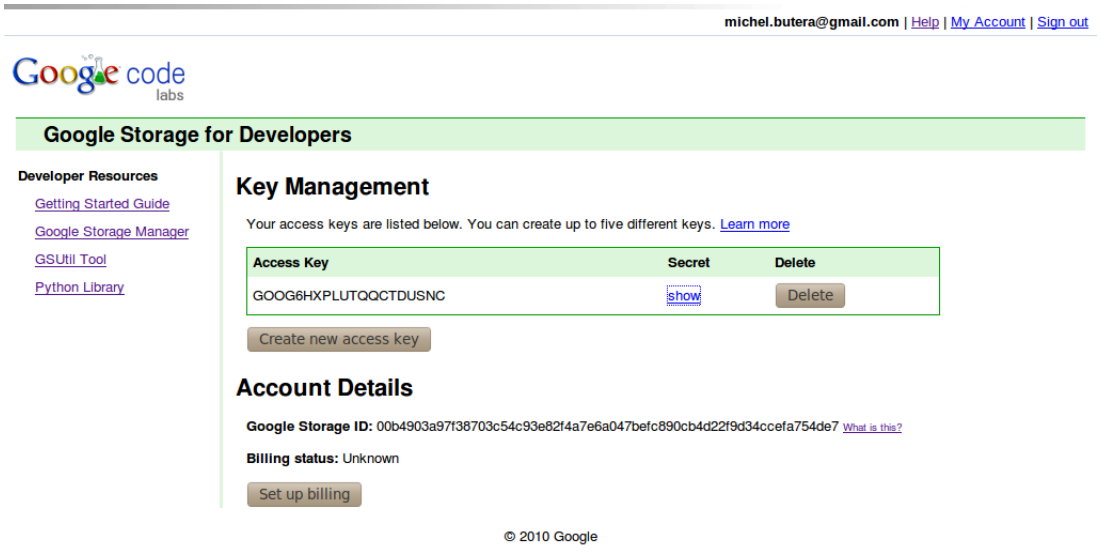

Figura 3.1: Gestore delle chiavi

Si possono creare fino a 5 set di chiavi di sviluppo, questo è utile se si lavora su diversi progetti e si vuole usare per ogni progetto una chiave di sviluppo diversa.

Queste chiavi di sviluppo devono essere nella sezione riguardante le credenziali di accesso del file .boto che viene creato automaticamente nella propria cartella /home/NomeUtente in seguito all'installazione di gsutil.

In questo modo:

[Credentials]
# google credentials ("gs://" URIs):
gs_access_key_id = GOOG6HXPLUTQQCTDUSNC
gs_secret_access_key = yEw+Le4P9ZCPlftYyVQpqVQqx5eyNucantYaIiFy

3.1.2 Creare un bucket

Per creare un bucket si utilizza il comando mb.

Il comando:

gsutil mb gs://NomeBucket1 gs://NomeBucket2

crea 2 bucket, uno nominato NomeBucket1 e l'altro NomeBucket2.

È importante ricordare che i nomi dei bucket devono essere univoci all'interno dell'intero namespace di Google Storage.

Se un altro utente ha già creato un bucket con il nome che vogliamo utilizzare, dobbiamo scegliere un altro nome.

3.1.3 Caricare oggetti in un bucket

Una volta creati i bucket, possiamo importare gli oggetti all'interno di questi.

Con il comando cp possiamo copiare i file dal nostro computer in Google Storage,

il comando:

gsutil cp image.jpg gs://NomeBucket/immagini/img.jpg

copia l'immagine image.jpg dalla cartella in cui siamo posizionati con la shell nel bucket NomeBucket rinominando l'oggetto in "immagini/img.jpg"

3.1.4 Elencare i bucket e gli oggetti

Se si vogliono elencare i nostri bucket o gli oggetti contenuti in essi si deve utilizzare il comando ls.

Il comando "gsutil ls" elenca tutti i bucket che abbiamo creato.

Mentre il comando:

gsutil ls gs://NomeBucket

elenca gli oggetti contenuti nel NomeBucket.

3.1.4.1 Opzione -L

È possibile usare l'opzione -L del comando ls per avere più informazioni sugli oggetti e sui bucket.

Il seguente comando:

gsutil ls -L gs://NomeBucket

fornisce informazioni circa la dimensione degli oggetti, la data dell'ultima modifica, il tipo di dato e gli ACL di tutti gli oggetti contenuti nel bucket NomeBucket.

Mentre il comando "gsutil ls -L" fornisce informazioni sui nostri bucket, come il numero di oggetti contenuti, la dimensione totale e gli ACL di tutti i nostri bucket.

3.1.5 Stampare il contenuto di un oggetto

È possibile utilizzare GSUtil per stampare il contenuto di un oggetto sullo standard output del terminale utilizzando il comando cat.

Il seguente comando stampa il contenuto del file nome.txt:

gsutil cat gs://NomeBucket/nome.txt

per concatenare il contenuto di più oggetti contemporaneamente, è possibile ad esempio usare questo comando:

gsutil cat gs://NomeBucket/file.txt gs//NomeBucket/sito.html

Oltre a visualizzare il contenuto degli oggetti (testuali) è anche possibile utilizzare il comando cat per stampare l'output in un file sul proprio computer.

Ad esempio questo comando:

gsutil cat gs://NomeBucket/*.txt > logs.txt

concatena tutti i file di testo contenuti nel bucket NomeBucket e stampa l'output nel file log.txt (nella cartella locale in cui siamo posizionati con la shell)

3.1.6 Trasferire un oggetto

Per trasferire un oggetto da un bucket ad un altro si utilizza il comando mv, questo comando può essere utilizzato anche per rinominare un oggetto.

Ad esempio il comando:

gsutil mv gs://NomeBucket1/image.jpg gs://NomeBucket2/

trasferisce l'immagine image dal bucket NomeBucket1 al bucketNomeBucket2.

Mentre il comando:

gsutil mv gs://NomeBucket/image.jpg gs://NomeBucket/img.jpg

rinomina l'immagine image in img.

3.1.7 Scaricare un oggetto

È possibile scaricare i propri oggetti, cioè copiarli da un bucket in Google Storage ad una directory sul proprio computer usando il comando cp.

Ad esempio il comando

gsutil cp gs://NomeBucket/*.jpg file://image

copia tutte le immagini JPG dal bucket NomeBucket nella propria directory image del proprio computer.

3.1.8 Condividere oggetti e bucket

È possibile condividere bucket ed oggetti con altri utenti modificando i loro ACL, questo è possibile usando i comandi getacl e setacl.

Gli ACL definiscono chi ha il permesso di accedere ai bucket e agli oggetti.

Il comando getacl restituisce un documento XML che elenca gli ACL di un bucket o di un oggetto. È possibile modificare questo documento con un normale editor di testo aggiungendo e rimuovendo gli ACL che ci interessano.

Il comando setacl applica gli ACL descritti nel documento XML sui bucket o oggetti interessati.

Ad esempio, il seguente comando:

gsutil getacl gs://NomeBucket/image.jpg > acl.txt

restituisce gli ACL relativi all'oggetto image.jpg e salva questi ACL nel file di testi acl.txt.

Se andiamo a modificare questo file con un editor di testo, vedremo un documento XML che descrive gli ACL, come questo:

```
<?xml version="1.0" ?>
 <AccessControlList>
  <Owner>
   <ID>your_google_storage_id</ID>
  </Owner>
  <Entries>
   <Entry>
    <Scope type="UserById">
     <ID>your_google_storage_id</ID>
    </Scope> <Permission>FULL_CONTROL</Permission>
   </Entry>
  </Entries>
 </AccessControlList>
```

Modificando gli ACL in questo documento è possibile condividere l'oggetto image.jpg con altri utenti.

Ad esempio, aggiungendo i seguenti ACL (le aggiunte sono in grassetto) al documento XML, si permette a Jane e a tutti i membri del gruppo Google Storage per gli sviluppatori di scaricare l'oggetto image.jpg:

```
<?xml version="1.0" ?>
 <AccessControlList>
  <Owner>
   <ID>your_google_storage_id</ID>
```

```
</Owner>
<Entries>
 <Entry>
  <Scope type="UserById">
   <ID>your_google_storage_id</ID>
  </Scope>
  <Permission>FULL_CONTROL</Permission>
 </Entry>
 <Entry>
  <Scope type="GroupByEmail">
   <EmailAddress>gs-discussion@googlegroups.com</EmailAddress>
  </Scope>
  <Permission>READ</Permission>
 </Entry>
 <Entry>
  <Scope type="UserByEmail">
   <EmailAddress>jane@gmail.com</EmailAddress>
  </Scope>
  <Permission>READ</Permission>
 </Entry>
</Entries>
</AccessControlList>
```

È possibile assegnare permessi di accesso agli altri utenti utilizzando l'indirizzo dell'account Gmail, l'ID di Google Storage o utilizzando l'indirizzo email di un gruppo di Google considerando quindi tutti quelli che ne fanno parte, per considerare invece tutti gli utenti che hanno un qualsiasi account Google si può utilizzare la parola chiave "AllAuthenticatedUsers".

Dopo aver effettuato le modifiche necessarie agli ACL è possibile eseguire il comando setacl per aggiornare i nuovi ACL sul nostro oggetto:

gsutil setacl acl.txt gs://NomeBucket/image.jpg

In questo modo passiamo come parametro il file di testo contenente i nuovi ACL e l'oggetto sua quale devono essere impostati.

Con il comando setacl è anche possibile applicare ai bucket e agli oggetti degli ACL predefiniti, senza modificare manualmente nessun file:

- private: imposta gli ACL in modo che solo il proprietario di un certo bucket o di un oggetto abbia i permessi di lettura/scrittura (questo è l'ACL che viene impostato di default quando si crea un bucket o si carica un oggetto)
- public-read: imposta gli ACL in modo che qualsiasi utente anche non autenticato abbia i permessi di lettura in un certo bucket o oggetto.

- public-read-write: imposta gli ACL in modo che qualsiasi utente anche non autenticato abbia i permessi di lettura e scrittura in un certo bucket (questo non è applicabile agli oggetti).
- authenticated-read: imposta gli ACL in modo che qualsiasi utente autenticato abbia i permessi di lettura in un certo bucket o oggetto.
- bucket-owner-read: imposta gli ACL in modo che il proprietario del bucket abbia i permessi di lettura sugli oggetti.

Possiamo ad esempio utilizzare il comando setacl nel seguente modo:
gsutil setacl private gs://NomeBucket/image.jpg
per reimpostare gli ACL di default.

3.1.9 Eliminare oggetti e bucket

È possibile eliminare oggetti con il comando rm ed eliminare bucket con il comando rb.
Ad esempio il seguente comando:
gsutil rm gs://NomeBucket/image.jpg
elimina l'immagine image.jpg contenuto nel bucket NomeBucket.
Mentre il comando:
gsutil rb gs://NomeBucket
elimina il bucket NomeBucket.
Per eliminare un bucket è necessario rimuovere tutti gli oggetti che contiene (è possibile sia eliminarli con il comando rm che trasferirli in un altro bucket con il comando mv).

3.2 BQ

BQ è uno strumento a riga di comando per accedere al servizio BigQuery.

Dopo aver effettuato l'accesso è possibile utilizzare bq per creare, importare ed eliminare le tabelle, così come eseguire query o ottenere lo schema delle tabelle.

L'operazione di importazione in tabelle BigQuery può durare anche molti minuti, quindi tramite il comando "bq import_status" possiamo controllare lo stato dell'importazione per verificare che nel frattempo non ci siano stati errori.

3.2.1 Autenticazione a Google BigQuery

La prima volta che viene lanciato bq verrà richiesto di eseguire l'accesso, e questo è possibile tramite il comando

bq login –user_email user@gmail.com

il flag –user_mail serve per indicare l'indirizzo email del proprio account Google, se non viene utilizzato di default viene considerata l'email che ha come nome utente il nickname di chi stà utilizzando il computer.

In seguito all'esecuzione di questo comando ci viene richiesta la password dell'account che abbiamo inserito.

3.2.2 Creare una tabella

Il primo passo per creare una tabella è creare un file che ne descrive lo schema, questo file deve contenere un array di parametri in formato JSON.

Ad esempio:

```
[
{ "id": "country", "type": "string" },
{ "id": "year", "type": "integer", "mode": "REQUIRED" }
]
```

Il campo "id" è il nome dell'attributo, "type" indica il tipo dell'attributo (accetta i tipi string, integer, float e boolean) e "mode" che è opzionale accetta i valori NULLABE e REQUIRED, il primo indica che quel campo può avere NULL, mentre il secondo indica che bisogna inserire un valore per quel campo.

Dopo aver creato questo file utilizziamo il comando "bq create" in questo modo:

bq create NomeBucket/Tabelle/NomeTabella file_schema

andando cioè a creare la tabella chiamata /Tabelle/NomeTabella con lo schema descritto nel file file_schema.

3.2.3 Eliminare una tabella

Per eliminare una tabella si utilizza il comando:

bq delete NomeBucket/Tabelle/NomeTabella

questo comporta l'eliminazione permanente sia della tabella che dei dati che conteneva.

3.2.4 Ottenere lo schema di una tabella

Se per caso volessimo sapere lo schema di una tabella, che può essere utile nel caso non ci ricordassimo tutti i campi che contiene o nel caso dovessimo andare a lavorare su una tabella che non è la nostra, possiamo utilizzare il comando:

bq describe NomeBucket/Tabelle/NomeTabella

Ci verranno restituiti tutti i dati descritti nel file con cui è stato descritto lo schema la prima volta che è stata creata.

3.2.5 Importare dati in una tabella

Per importare i record di una nostra tabella in una tabella BigQuery dobbiamo inserirli in un file in formato CSV (con i valori separati da virgole) e trasferire questo file in Google Storage.

Utilizzando poi il comando:

bq import NomeBucket/Tabelle/NomeTabella NomeBucket/record.csv

vengono importati tutti i record contenuti nel file record.csv nella tabella Tabelle/NomeTabell.

Questo è un processo che può prendere molto tempo se si considerano tabelle con miliardi di record, ed è per questo che appena lanciamo il comando ci viene restituito un messaggio:

{
"table": "NomeBucket/Tabelle/NomeTabella",
"kind": "bigquery#import_id",
"import": "2c212c1f31a32fa7"
}

contenente il tipo di operazione, l'oggetto sul quale stiamo lavorando e anche l'id di questa operazione, che ci può essere utile nel caso volessimo andare a controllare lo stato di questa operazione.

3.2.5.1 File CSV

Comma-separated values ("valori separati da virgole") è un formato di file basato su file di testo utilizzato per l'importazione ed esportazione (ad esempio da fogli elettronici o database) di una tabella di dati.

In questo formato, ogni riga della tabella (o record della base dati) è normalmente rappresentata da una linea di testo, che a sua volta è divisa in campi (le singole colonne) separati da un apposito carattere separatore (quello usato più comunemente è la virgola), ciascuno dei quali rappresenta un valore.

Ad esempio:

Opera	Autore	Casa Editrice
I Robot e l'Impero	Isaac Asimov	Mondadori
Il lungo meriggio della Terra	Brian W. Aldiss	Minotauro
Absolute OpenBSD "2d Edition"	Michael W. Lucas	No Starch Press
I mercanti dello spazio	Frederik Pohl, C. M. Kornbluth	Mondadori

L'esempio qui sopra si potrebbe rappresentare in CSV come:

OPERA,AUTORE,CASA EDITRICE
I Robot e l'Impero,Isaac Asimov,Mondadori
Il lungo meriggio della Terra,Brian W. Aldiss,Minotauro
Absolute OpenBSD ""2d Edition""",Michael W. Lucas,No Starch Press
I mercanti dello spazio,"Frederik Pohl, C. M. Kornbluth",Mondadori

3.2.6 Controllare lo stato di importazione

Per controllare lo stato d'importazione dei dati in una tabella BigQuery si deve utilizzare il parametro import_status dello strumento bq.

Il seguente comando:

bq import_status NomeBucket/Tabelle/NomeTabella 2c212c1f31a32fa7

che prende come parametro l'indirizzo della tabella e l'id dell'operazione di importazione su quella tabella, può restituire uno dei seguenti valori:

- IMPORT_QUEUED: l'importazione non è ancora iniziata
- IMPORT_RUNNING: l'importazione è ancora in esecuzione
- IMPORT_DONE: l'importazione è stata eseguita Data is ready to be used. It is possible that there might be errors importing individual records
- IMPORT_ABORTED: sono stati riscontrati degli errori nel file CSV contenente i dati, durante l'importazione dei dati e quindi i dati non sono stati importati (anche un solo errore fa terminare l'importazione dei dati)

3.2.7 Eseguire le query

Quando tutte le tabelle sono state popolate è possibile iniziare ad eseguire le query.

Per eseguire le query si utilizza sempre lo strumento bq, ma questa volta andremo ad usare il metodo query, passando come parametro la query stessa.

Un semplice esempio di query è:

bq query "SELECT nomi FROM [NomeBucket/Tabelle/NomeTabella]"

questa query restituisce tutti i valori dei record dell'attibuto "nomi" contenuti nella tabella "Tabelle/NomeTabella".

Capitolo 4

Sintassi SQL in BigQuery

Le query in BigQuery sono formulate in un dialetto SQL.

Questo dialetto supporta una variante della clausola SELECT del SQL standard, insieme ad una grande varietà di funzioni come COUNT, espressioni aritmetiche, funzioni di stringa, e così via.

Tutte le query in BigQuery sono delle istruzioni SELECT che devono avere questa forma:

```
SELECT expr1 [AS alias1], expr2 [AS alias2], ...
[FROM table_name1|(subselect1)] [, table_name2|(subselect2), ...]
[WHERE condition]
[GROUP BY field1|alias1, field2|alias2, ...]
[ORDER BY field1|alias1 [DESC|ASC], field2|alias2 [DESC|ASC], ...]
[LIMIT n] ;
```

(Le parole chiave sono case-insensitive, ma per chiarezza verranno scritte in maiuscolo).

Questo capitolo

descrive la sintassi della clausola SELECT di BigQuery e le funzioni che possono essere utilizzate all'interno della clausola SELECT.

4.1 Clausole e metodi

4.1.1 Clausola SELECT

Nella clausola SELECT si specifica il set di valori che deve essere restituito dalla query. Le espressioni expr1 nella clasuola SELECT possono essere campi, lettarali o funzioni che che operano sui campi o letterali. Le espressioni devono essere separate da una virgola.

Ad esempio:
SELECT word AS w, LENGTH(word), 3*2-1
FROM [bigquery/samples/shakespeare]
WHERE word CONTAINS "th";

4.1.1.1 Alias

Le espressione possono avere un alias fornendo un nome diverso per un espressione, usando il comando AS. Gli alias sono utili per riferirsi ai risultati delle funzioni, gli ALIAS possono riferirsi solo alle clausole GROUP BY e ORDER BY.

DA notare che che la query SELECT * FROM ... che in SQL e molti altri dialetti SQL ritorna tutti i campi di una tabella, non è supporta.

4.1.2 Clausola FROM

Nella clausola FROM si dichiara la tabella da cui ottenere i dati (da notare che al momento non è possibile effettuare nessun tipo di join, quindi nella clausola FROM ci sarà sempre e solo una tabella).

4.1.2.1 Identificare una tabelle

I nomi delle tabelle in BigQuery non sono identificatori SQL validi, quindi devono essere indicati tra parentesi quadre [] nelle query.

Esempio:
SELECT COUNT(*)
FROM [bigquery/samples/shakespeare]
WHERE word CONTAINS "th";

In questo caso andiamo a identificare la tabella nominata “samples/shakespeare” all’interno del bucket “bigquery”.

4.1.2.2 Subquery

Una subquery è una query all’interno di un’altra query.

È possibile specificare una subquery all’interno delle parentesi nella clausola FROM. La subquery viene valutata e il risultato viene trattato come se fosse una tabella, quindi il risultato deve avere uno schema adeguato alla SELECT della query esterna.

Se la query include più sottoquery o una combinazione di sottoquery e tabelle, tutte le tabelle e subquery devono contenere tutti i campi della clausola SELECT della query principale. Le regole che si applicano alle query si applicano anche alle sottoquery, salvo che le subquery non terminano con un punto e virgola.

4.1.3 Clausola HAVING

The clausola HAVING in questo dialetto SQL non è supportata.

Una soluzione è utilizzare una subquery in modo da avere lo stesso effetto, ad esempio questa query:

```
SELECT f1, SUM(f2)
FROM [Table]
GROUP BY f1
HAVING SUM(f2) > 1000;
```

può essere riscritta utilizzando una subquery al posto della clausola HAVING:

```
SELECT f1, total
FROM (SELECT f1, SUM(f2) AS total FROM [Table] GROUP BY f1)
WHERE total > 1000;
```

4.1.4 Clausola WHERE

La clausola WHERE, chiamata anche predicato, consente di filtrare i dati della query selezionando solo gli elementi che soddisfano determinati criteri, agisce come un filtro inquanto rimuove dai valori restituiti dalla query quei valori che non soddisfano le condizioni richieste. È possibile avere condizioni multiple usando gli operatori logici AND o OR. Le condizioni nella clausola WHERE sono basate su funzioni di confronto, che restituiscono vero o falso. I record sono inclusi nel set di risultati se e solo se la condizione WHERE è soddisfatta.

Le funzioni di aggregazione non possono essere utilizzati nella clausola WHERE.

4.1.5 Clausola GROUP BY

La clausola GROUP BY viene utilizzata, come dice il nome, per suddividere in gruppi le righe estratte mediante istruzione SELECT, consente di raggruppare le righe che hanno gli stessi valori per un dato campo. Sarà quindi possibile eseguire funzioni di aggregazione su ciascuno dei gruppi. Ad esempio è possibile raggruppare le righe che hanno lo stesso valore per il campo f1 e trovare la somma (SUM) dei valori di f2 per ogni gruppo:

```
SELECT f1, SUM(f2)
FROM [Table]
GROUP BY f1;
```

La clausola GROUP BY e la clausola SELECT sono collegate per 2 motivi, per prima cosa i campi nella clausola GROUP BY devono essere elencati nella clausola SELECT,

e secondo motivo, i campi non aggregati nella clausola SELECT devono essere elencati nella clausola GROUP BY.

Ad esempio, questa query non è valida perché il campo f3 non è elencato nella clausola SELECT:

```
SELECT f1, COUNT(f2)
FROM [Table]
GROUP BY f1, f3;
```

Mentre quest'altra non è valida perché il campo f3 non è elencato nella clausola GROUP BY:

```
SELECT f1, COUNT(f2), f3
FROM [Table]
GROUP BY f1;
```

Il raggruppamento per i campi con valori double o float non è supportato, perché la funzione di uguaglianza per questi tipi non è ben definita.

Poiché il sistema è interattivo, alcune query che producono un gran numero di raggruppamenti potrebbero restituire qualche errore (ad esempio se impiegano troppo tempo o troppa memoria), per questo è stato introdotto un metodo (TOP) da utilizzare al posto di GROUP BY che permette di risolvere questo problema.

4.1.6 Metodo TOP

La funzione TOP è una semplice alternativa per le query che seguono lo schema:
... GROUP BY ... ORDER BY ... LIMIT ....

TOP è specifico per BigQuery ed è progettato per approssimare i risultati usando i valori più significativi, ha un esecuzione più veloce rispetto all'intera sintassi GROUP BY, ORDER BY, LIMIT, ma i risultati possono essere indicativi.

L'esempio seguente mostra 2 query equivalenti, la prima utilizza la sintassi GROUP BY, ORDER BY, LIMIT, mentre l'altra utilizza solo il comando TOP.

Questa query considerando una tabella con un attributo word estrae da questa tutte le parole contenenti "th", raggruppa le paroli uguali, le conta, e restituisce (in ordine crescente) le 10 parole che compaiono più spesso e il numero di volte che compaiono nella tabella:

```
SELECT word, COUNT(*) AS cnt
FROM [Table]
```

```
WHERE word CONTAINS 'th'
GROUP BY word
ORDER BY cnt
DESC LIMIT 10;
```

Utilizzando TOP, possiamo riscrivere la stessa query in questo modo:

```
SELECT TOP(word, 10), COUNT(*)
FROM [Table]
WHERE word CONTAINS 'th';
```

Una clausola SELECT che include la funzione TOP deve contenere anche COUNT(*).

4.1.7 Clausola ORDER BY

La clausola ORDER BY ordina i risultati di una query in ordine crescente o decrescente. Si utilizza DESC (decrescente) o ASC (crescente) per specificare il tipo di ordinamento che vogliamo (di default è impostato il valore ASC).

Per ordinare in base a più campi o alias, basta elencare un campo dopo l'altro, separati da virgole. I risultati verranno ordinati partendo dal primo campo o alias, in seguito verranno ordinati in base al secondo campo e così via, ma sempre seguendo l'ordinazione precedente.

4.1.8 Clausola LIMIT

La clausola LIMIT limita il numero di righe nel set di risultati restituiti.

Ad esempio, la seguente query restituisce solo i primi 5 risultati:

```
SELECT COUNT(*), word
FROM [Table]
WHERE word CONTAINS 'th'
GROUP_BY word
LIMIT 5;
```

La clausola LIMIT non può contenere funzioni, prende solo costanti numeriche.

Questa clausola è utile quando non si vuole visualizzare l'intero set di risultati, ma sono sufficienti alcune righe.

4.2 Funzioni

All'interno delle query è possibile fare uso di molti tipi di funzioni, sono supportate funzioni di aggregazione, aritmetiche e matematiche, sui bit, di casting, di confronto, altre per gestire gli indirizzi IP, per eseguire operazioni con operatori logici, sulle stringhe, sui timestamp e altre ancora.

4.2.1 Funzioni di aggregazione

Le funzioni di aggregazione restituiscono valori che rappresentano la sintesi di grandi insiemi di dati. Una funzione di aggregazione opera nei confronti di un insieme di valori, può agire su tutti i record di una tabella o solo su alcuni valori, selezionando un gruppo di record usando la clausola GROUP BY.

- Aggregazione su una tabella: utilizza una funzione di aggregazione per riassumere tutte le righe che vogliamo considerare nella tabella. Ad esempio:
  SELECT COUNT(f1) FROM Table;

- Aggregazione di gruppo: utilizza una funzione di aggregazione e una clausola GROUP BY che specifica un campo non aggregato per riassumere le righe di un certo gruppo. Ad esempio:
  SELECT COUNT(f1) FROM Table GROUP BY b1;
  (la funzione TOP rappresenta un caso specializzato di aggregazione di gruppo)

La tabella seguente mostra le funzioni di aggregazione supportate da BigQuery:

Sintassi	Descrizione
COUNT(*)	Restituisce il numero totale di valori (NULL e non) nel campo di applicazione della funzione. A meno che non si usi COUNT (*) in combinazione alla funzione TOP, è meglio specificare esplicitamente il campo da contare.
COUNT([DISTINCT] field [, n])	Restituisce il numero totale di valori non NULL nel campo di applicazione della funzione. Se si utilizza la parola chiave DISTINCT, la funzione restituisce il numero di valori univoci nel campo specificato. Quando si utilizza la parola chiave DISTINCT, è anche possibile specificare un valore n (con n>0) e se il conteggio è minore di n la funzione restituisce un risultato esatto, mentre se il conteggio è maggiore di n la funzione restituisce un approssimazione statistica. Il valore predefinito per n è 1000. Valori troppo grandi per n potrebbero aumentare il tempo di esecuzione della query. In generale, è meglio utilizzare COUNT([DISTINCT] field [, n]) invece di COUNT (*), perchè specificando il campo non c'è confusione sul campo su cui si vuole contare, ad esempio quando sono presenti campi ripetuti.

Sintassi	Descrizione
MAX(field)	Restituisce il valore massimo tra i valori del campo passato come parametro.
MIN(field)	Restituisce il valore minimo tra i valori del campo passato come parametro.
SUM(field)	Restituisce la somma dei valori del campo passato come parametro. Funziona solo con dati di tipo numerico.
TOP(field, [numeric_const], [multiplier])	Restituisce i campi che compaiono con più frequenza. TOP è una scorciatoia nei casi in cui si deve utilizzare una query con lo schema GROUP BY ... ORDER BY ... LIMIT. Il valore numeric_const ha lo stesso effetto della clausola LIMIT. È facoltativo, e il valore predefinito è 20. L'argomento moltiplicatore deve essere un valore di tipo numerico, moltiplica i valori di ogni riga per quel valore.

Tabella 4.1: Funzioni di aggregazione

La funzione per il calcolo della media (AVG) non è implementata (la si può calcolare tramite SUM / COUNT).

4.2.2 Funzioni aritmetiche e matematiche

Le funzioni aritmetiche e le funzioni matematiche si possono utilizzare solo con tipi di dati numerici. Prendono uno o due argomenti di input di tipo numeric_expr e restituiscono un valore dello stesso tipo. Gli argomenti possono essere valori numerici inseriti direttamente o valori numerici recuperati da una query. Se una funzione aritmetica o matematica si trova a dover restituire un risultato indefinito (come 5 / 0), viene restituito il valore NULL.

Sintassi	Descrizione
numeric_expr + numeric_expr	Restituisce la somma dei 2 valori, come numeric_expr.
numeric_expr - numeric_expr	Restituisce la differenza tra i 2 valori, come numeric_expr. Può restituire anche valori negativi.
numeric_expr * numeric_expr	Restituisce il prodotto dei 2 valori, come numeric_expr.
numeric_expr / numeric_expr	Restituisce il risultato della divisione tra i 2 valori, come numeric_expr.
numeric_expr % numeric_expr	Restituisce il resto della divisione tra i 2 valori, come numeric_expr.
ABS(numeric_expr)	Restituisce il valore assoluto dell'argomento come numeric_expr.
ACOS(numeric_expr)	Restituisce l'arco coseno dell'argomento come numeric_expr.
ACOSH(numeric_expr)	Restituisce il coseno iperbolico dell'argomento come numeric_expr.
ASIN(numeric_expr)	Restituisce l'arco seno dell'argomento come numeric_expr.
ASINH(numeric_expr)	Restituisce il seno iperbolico dell'argomento come numeric_expr.
ATAN(numeric_expr)	Restituisce l'arco tangente dell'argomento come numeric_expr.
ATANH(numeric_expr)	Restituisce l'arco tangente iperbolico dell'argomento come numeric_expr.
ATAN2(num_expr1, num_expr2)	Restituisce l'arco tangente dei 2 argomenti come numeric_expr.
CEIL(numeric_expr)	Arrotonda l'argomento fino al numero intero più vicino e restituisce il valore arrotondato come un numeric_expr.
COS(numeric_expr)	Restituisce il coseno dell'argomento come numeric_expr.
COSH(numeric_expr)	Restituisce il coseno iperbolico dell'argomento come numeric_expr
COT(numeric_expr)	Restituisce il cotangente dell'argomento come un numeric_expr.
DEGREES(numeric_expr)	Restituisce l'argomento, convertito da radianti a gradi.

Sintassi	Descrizione
FLOOR(numeric_expr)	Arrotonda l'argomento per difetto e restituisce il valore arrotondato come un numeric_expr.
LN(numeric_expr) LOG(numeric_expr)	Restituiscono il logaritmo naturale dell'argomento come numeric_expr.
LOG2(numeric_expr)	Restituiscono il logaritmo in base 2 dell'argomento come numeric_expr.
LOG10(numeric_expr)	Restituiscono il logaritmo in base 10 dell'argomento come numeric_expr.
PI()	Restituisce la costante pi-greco (non richiede nessun argomento passato come parametro). È possibile utilizzare PI () come una costante in funzioni matematiche e aritmetiche.
POW(num_expr1, num_expr2)	Restituisce come numeric_expr, il risultato di num_expr1 elevato alla potenza di num_expr2.
RADIANS(numeric_expr)	Restituisce l'argomento convertito da gradi a radianti.
ROUND(numeric_expr)	Arrotonda l'argomento al numero intero più vicino e restituisce il valore arrotondato come un numeric_expr.
SIN(numeric_expr)	Restituisce il seno dell'argomento come numeric_expr.
SINH(numeric_expr)	Restituisce il seno iperbolico dell'argomento come numeric_expr.
SQRT(numeric_expr)	Restituisce la radice quadrata dell'espressione come numeric_expr.
TAN(numeric_expr)	Restituisce la tangente dell'argomento come numeric_expr.
TANH(numeric_expr)	Restituisce la tangente iperbolica dell'argomento come numeric_expr.

Tabella 4.2: Funzioni aritmetiche e matematiche

4.2.3 Funzioni bit a bit

Le funzioni bit a bit operano a livello di bit individuale. Gli argomenti di queste funzioni devono essere numerici.

Sintassi	Descrizione
numeric_expr & numeric_expr	Restituisce come numeric_expr, il risultato dell'AND bit a bit tra le 2 numeric_expr.
numeric_expr | numeric_expr	Restituisce come numeric_expr, il risultato dell'OR bit a bit tra le 2 numeric_expr.
numeric_expr ^ numeric_expr	Restituisce come numeric_expr, il risultato dell'XOR bit a bit tra le 2 numeric_expr.
num_expr1 << num_expr2	Restituisce come numeric_expr, il risultato di shiftare i bit di num_expr1 a sinistra di num_expr2 spazi.
num_expr1 >> num_expr2	Restituisce come numeric_expr, il risultato di shiftare i bit di num_expr1 a destra di num_expr2 spazi.
~ numeric_expr	Restituisce come numeric_expr, l'inversione di tutti i bit dell'espressione numerica.

Tabella 4.3: Funzioni bit a bit

4.2.4 Funzioni di casting

È possibile utilizzare le funzioni di casting per cambiare il tipo di un dato in un'espressione numerica. Le funzioni di casting sono particolarmente utili per assicurarsi che gli argomenti di una funzione di confronto abbiano lo stesso tipo di dati.

Sintassi	Descrizione
BOOLEAN(numeric_expr)	Restituisce • true se expr non è ne 0 ne NULL • false se expr è 0 • NULL se numeric_expr è NULL.
FLOAT(expr)	Restituisce expr convertito nel tipo double, (se viene passato un dato non numerico la funzione restituisce NULL).
HEX_STRING(num_expr)	Restituisce num_expr convertito in una stringa esadecimale.
INTEGER(expr)	Restituisce expr convertito in un intero a 64 bit, expr può essere una stringa come '45', '0x2D' (esadecimale) o '055' (ottale). La funzione restituisce NULL se si passano come parametro valori non convertibili.
STRING(numeric_expr)	Restituisce numeric_expr convertito in una stringa.

Tabella 4.4: Funzioni di casting

4.2.5 Funzioni di confronto

Le funzioni di confronto restituiscono i valori vero o falso, sulla base di diversi tipi di confronto:

- Confronto tra due espressioni
- Controllare che un'espressione un insieme di espressioni rispettino un criterio specifico, ad esempio che il valore di un certo campo sia contenuto in una certa lista o che a un campo opzionale sia stato impostato un valore o meno.

È possibile utilizzare espressioni numeriche o di stringa come argomenti per le funzioni di confronto (le stringhe devono essere racchiuse tra singoli o doppi apici).

Sintassi	Descrizione
expr1 = expr2	Restituisce true se le espressioni sono uguali
expr1 != expr2 expr1 <> expr2	Restituisce true se le espressioni non sono uguali
expr1 > expr2	Restituisce true se expr1 è maggiore di expr2
expr1 < expr2	Restituisce true se expr1 è minore di expr2
expr1 >= expr2	Restituisce true se expr1 è maggiore o uguale di expr2
expr1 <= expr2	Restituisce true se expr1 è minore o uguale di expr2
expr IN(expr1, expr2, ...)	Restituisce true se expr corrisponde a expr1, expr2, o a qualsiasi valore tra parentesi. La parola chiave IN è una scorciatoia efficace per evitare condizione estese come: (expr = expr1 | expr = expr2 | || ...). Le espressioni usate con la parola chiave IN devono essere costanti e devono essere dello stesso tipo di expr.
IS_NULL(expr)	Restituisce true se expr è null.
IS_EXPLICITLY_DEFINED(expr)	Restituisce true se è stato definito esplicitamente un valore per il campo expr (expr deve essere un campo opzionale).

Tabella 4.5: Funzioni di confronto

4.2.6 Funzioni IP

Le funzioni IP convertono gli indirizzi IP in un formato leggibile e viceversa.

Sintassi	Descrizione
FORMAT_IP(integer_value)	Converte i 32 bit meno significativi di integer_value in una stringa contenente l'indirizzo in formato IPv4. Ad esempio, FORMAT_IP(1) restituirà la stringa '0 .0.0.1 '.
PARSE_IP(readable_ip)	Converte una stringa che rappresenta un indirizzo IPv4 in un valore intero senza segno. Ad esempio, PARSE_IP ('0.0.0.1 ') restituirà 1. Se la stringa non è un indirizzo IPv4 valido, PARSE_IP restituirà NULL.

Tabella 4.6: Funzioni IPv4

Sono supportati indirizzi IP sia in formato IPv4 che in formato Ipv6.
Queste funzioni operano solo su campi di tipo stringa che contengono indirizzi IP.

Sintassi	Descrizione
FORMAT_PACKED_IP(packed_ip)	Restituisce un indirizzo IP in un formato leggibile, nel formato 10.1.5.23 oppure 2620:0:1009:1:216:36ff:feef:3f. Ad esempio, FORMAT_PACKED_IP('0123456789@ABCDE') restituirà la stringa '3031:3233:3435:3637:3839:4041:4243:4445' mentre FORMAT_PACKED_IP('0123') restituirà la stringa '48.49.50.51'.
PARSE_PACKED_IP(readable_ip)	Restituisce un indirizzo IP in formato binario in una stringa. Ad esempio, PARSE_PACKED_IP ('48 .49.50.51 ') restituirà la stringa '0123', mentre PARSE_PACKED_IP('3031:3233:3435:3637:3839: 4041:4243:4445 ') restituisce la stringa: '0123456789 @ ABCDE'. Se la stringa di input non è un indirizzo IPv4 o Ipv6 valido, PARSE_PACKED_IP restituirà NULL.

Tabella 4.7: Funzioni IPv6

4.2.7 Funzioni con operatori logici

Gli operatori logici permettono di utilizzare una logica binaria o ternaria sulle espressioni. La logica binaria restituisce solo i valori true o false, mentre la logica ternaria può restituire true, false, o NULL.

Sintassi	Descrizione
expr AND expr	Restituisce: • true se entrambe le espressioni sono vere • false se una o entrambe le espressioni sono false • NULL se entrambe le espressioni sono NULL o se una espressione è vera e l'altra è NULL.
expr OR expr	Restituisce: • true se una o entrambe le espressioni sono vere • false se entrambe le espressioni sono false • NULL se entrambe le espressioni sono NULL o se una espressione è false e l'altra è NULL.
NOT expr	Restituisce: • true se l'espressione è vera • false se l'espressione è falsa • NULL se l'espressione è NULL. È possibile usare NOT con altre funzioni come operatore di negazione, ad esempio: NOT IN(expr1, expr2) o IS NOT NULL.

Tabella 4.8: Funzioni con operatori logici

4.2.8 Funzioni sulle stringhe

Le funzioni sulle stringhe operano solo sui dati di tipo stringa. Le stringhe devono essere racchiuse tra singoli o doppi apici. Alcune di queste funzioni operano solo su stringhe in codifica Latin-1, ad esempio le funzioni sulle stringhe sono case-sensitive di defalut, per

abilitare la modalità case-insensitive è possibile aggiungere IGNORE CASE alla fine di una query, ma questa funziona solo con le stringhe in formato LATIN-1.

Sintassi	Descrizione
CONCAT('str1', 'str2')	Restituisce la concatenazione delle 2 stringhe. Ad esempio, se str1 è Java e str2 è Script, CONCAT restituisce JavaScript.
expr CONTAINS 'str'	Restituisce true se expr contiene la stringa specificata.
LEFT('str', numeric_expr)	Restituisce una sottostringa di str, iniziando dal primo carattere di sinistra fino al carattere definito con numeric_expr. Ad esempio, LEFT('seattle', 3) restituisce sea.
LENGTH('str')	Restituisce un valore numerico che corrisponde alla lunghezza della stringa. Ad esempio, se str è '123456', LENGTH restituisce 6.
LOWER('str)	Restituisce la stringa originale con tutti i caratteri in minuscolo. Funziona solo con i caratteri in formato LATIN-1.
LPAD('str1', numeric_expr, 'str2')	Restituisce la stringa str2 concatenata alla stringa str1, se la stringa da restituire è più lunga di numeric_expr caratteri viene troncata la stringa str2. Ad esempio, LPAD ('ator', 7, 'senior') restituisce senator.
RIGHT('str', numeric_expr)	Restituisce una sottostringa di str di numeric_expr caratteri, considerando la stringa a partire da destra. Ad esempio, RIGHT('kirkland', 4) restituisce land.
RPAD('str1', numeric_expr, 'str2')	Restituisce la stringa str1 concatenata alla stringa str2, se la stringa da restituire è più lunga di numeric_expr caratteri viene troncata la stringa str2. Ad esempio, RPAD('west ', 12, 'seattleite') restituisce west seattle
SUBSTR('str', const_from, const_len)	Restituisce una sottostringa di str di lunghezza const_len caratteri, e che inizia dal punto specificato da const_from. Il conteggio inizia da 1, quindi il primo carattere della stringa è in posizione 1 (non zero). Se const_from è 5, la sottostringa inizia con il quinto carattere da sinistra in str. Se const_from è -4, la sottostringa inizia con il quarto carattere da destra in str. Ad esempio, SUSBTR('awesome', -4, 4) restituisce la sottostringa some.
UPPER('str')	Restituisce la stringa originale con tutti i caratteri in maiuscolo. Funziona solo con i caratteri in formato LATIN-1.

Tabella 4.9: Funzioni sulle stringhe

4.2.8.1 Caratteri speciali nelle stringhe

Per inserire dei caratteri speciali all'interno delle stringhe, si deve utilizzare uno dei seguenti metodi:

- utilizzare la notazione '\xDD', dove '\x' è seguita da 2 cifre che rappresentano la rappresentazione esadecimale del carattere
- utilizzare uno slash davanti a singoli e doppi apici
- utilizzare sequenze in stile C ('\ a', '\ b', '\ f', '\ n', '\ r', '\ t', e '\ v').

Degli esempi di inserimento di caratteri speciali possono essere:
'Questo è uno spazio: \x20'
'Questa stringa è tra \'singoli apici\''
'prima linea \n seconda linea'

4.2.9 Funzioni sui timestamp

In queste funzioni quando si indica il parametro timestamp_usec si intende un intero a 64 bit senza segno pari al timestamp di Unix con una precisione al microsecondo.

Queste funzioni operano sul fuso-orario locale del server:

Sintassi	Descrizione
FORMAT_TIME_USEC(timestamp_usec)	Restituisce il timestamp nella data corrispondente, nel formato YYYY-MM-DD HH:MM:SS[.uuuuu].
PARSE_TIME_USEC(readable_timestamp)	Restituisce il readable_timestamp nel formato timestamp_usec. Il readable_timestamp fornito come argomento deve essere nel formato YYYY-MM-DD HH:MM:SS[.uuuuuu]. La parte frazionaria del secondo può essere fino a 6 cifre o può essere omessa.
TIME_USEC_TO_DAY(timestamp_usec)	Restituisce il timestamp_usec spostato all'inizio del giorno a cui appartiene. Ad esempio, se timestamp_usec è 1207929480000000 (questo corrisponde a 2008-04-11 08:58:00.000000), TIME_USEC_TO_DAY restituisce 1207897200000000 (che corrisponde a 2008-04-11 00:00:00.000000).
TIME_USEC_TO_HOUR(timestamp_usec)	Restituisce il timestamp_usec spostato all'inizio dell'ora a cui appartiene. Ad esempio, se timestamp_usec è 1207929480000000 (che corrisponde a 2008-04-11 08:58:00.000000), TIME_USEC_TO_HOUR restituisce 1207926000000000 (che corrisponde a 2008-04-11 08:00:00.000000).
TIME_USEC_TO_MONTH(timestamp_usec)	Restituisce il timestamp_usec spostato all'inizio del mese a cui appartiene. Ad esempio, se timestamp_usec è 1207929480000000 (che corrisponde a 2008-04-11 08:58:00.000000), TIME_USEC_TO_MONTH restituisce 1207033200000000 (che corrisponde a 2008-04-01 00:00:00.000000).
TIME_USEC_TO_YEAR(timestamp_usec)	Restituisce il timestamp_usec spostato all'inizio dell'anno a cui appartiene. Ad esempio, se timestamp_usec è 1207929480000000 (che corrisponde a 2008-04-11 08:58:00.000000), TIME_USEC_TO_YEAR restituisce 1199174400000000 (che corrisponde a 2008-01-01 00:00:00.000000).

Tabella 4.10: Funzioni basate sull'orario locale del server

Queste altre, sono identiche a quelle sopra, ma lavorano con l'orario UTC[1]:

Sintassi
FORMAT_UTC_USEC(timestamp_usec)
PARSE_UTC_USEC(readable_timestamp)
UTC_USEC_TO_DAY(timestamp_usec)
UTC_USEC_TO_HOUR(timestamp_usec)
UTC_USEC_TO_MONTH(timestamp_usec)
UTC_USEC_TO_YEAR(timestamp_usec)

Tabella 4.11: Funzioni basate sull'orario UTC

4.2.10 Altre funzioni

Sintassi	Descrizione
IF(condition, true_return, false_return)	Restituisce true_return o false_return, a seconda se la condizione è vera o falsa.

Tabella 4.12: Altre funzioni

[1]Il Tempo Coordinato Universale (Coordinated Universal Time - UTC), conosciuto anche come tempo civile, è il fuso orario di riferimento da cui tutti gli altri fusi orari del mondo sono calcolati.

Capitolo 5

Importare un database in BigQuery

Se si vuole trasferire un proprio database in BigQuery quello che si deve fare è portare ogni singola tabella in un file di testo in formato CSV.

Per automatizzare questa lunga operazione è possibile utilizzare il programma freeware SQL2CSV, questo strumento permette di convertire i record selezionati da una tabella (anche tutti) di un database in formato CSV e salvare il tutto in un file di testo.

SQL2CSV supporta solo i sistemi operativi Microsoft Windows a 32 bit (Windows 2000/XP/Server 2003/Vista) con il .NET Framework 2.0 installato, ma permette di lavorare con quasi tutti i tipi di database:

- Microsoft SQL Server (System.Data.SqlClient)
- ODBC Connections (System.Data.Odbc)
- Oracle (System.Data.OracleClient)
- Firebird (FirebirdSql.Data.FirebirdClient)
- MySQL (MySql.Data.MySqlClient) - PostgreSQL (Npgsql)
- SQLite (System.Data.SQLite)

In ogni caso esistono in circolazione applicazioni per ogni tipo di database e sistema operativo che permettono di esportare dati in formato CSV dalle tabelle.

5.1 Utilizzo di SQL2CSV

SQL2CSV (come tutti i programmi di esportazione di dati da un database) esporta i dati che vengono selezionati da una tabella attraverso una query.

In questo caso è sufficiente lanciare il comando:

Sql2Csv <Connectionname> <SQL-Query> <Filename>

che in un caso reale potrebbe essere:

Sql2Csv "Sample (SQLite Database File)" "SELECT * FROM Products" "C:\Products.csv"

questo comando apre il database Sample che utilizza la libreria SQLite (quindi contiene tutto in unico file), esegue una query sulla tabella Products selezionando tutti i record e li salva nel file di testo Products.csv in formato CSV (con i valori separati da una virgola).

Capitolo 6

Esempio pratico

In questo capitolo mostrerò un esempio pratico di utilizzo di Google BigQuery, i dati che verranno utilizzati per popolare le tabelle sono stati scaricati dal sito della Social Security Online, in cui è possibile trovare un archivio di tutti i nomi più usati dati alla nascita ai propri figli dal 1980.

È quindi possibile trovare un file zippato contenente 120 file (uno per ogni anno) dove ognuno contiene i 1000 nomi più usati per quell'anno.

Sono stati scelti questi dati perchè sono già in formato CSV e quindi pronti per essere usati da BigQuery.

6.1 Creare il file di dati

I file che andiamo a considerare sono già in formato CSV con i valori separati da virgole, quindi sono già nel formato adatto per Google BigQuery.

Questi file sono formattati in 3 colonne: nome, sesso ("M" o "F") e numero di bambini con quel nome.

Altrimenti avremmo dovuto esportare i dati nelle nostre tabelle in formato CSV (tramite qualche applicazione) o nel caso di una tabella non ancora esistente avremmo dovuto creare e popolare da zeeo i file CSV.

6.2 Creare un bucket

Chiameremo il nostro bucket "tesi", per farlo entriamo nella shell e lanciamo il seguente comando:

gsutil mb gs://tesi

Verrà mostrato il seguente messaggio:

Creating gs://tesi/

che ci indica Google Storage sta eseguendo il nostro comando.

Uno dei possibili errori che può avvenire oltre ai problemi legati alla connessione è che qualcuno abbia già nominato un bucket con questo nome.

Nel caso il nome scelto per il nostro bucket sia già stato usato da un altro utente ci verrà restituito questo errore:

Failure: GSCreateError: 409 Conflict
<?xml version='1.0'encoding='UTF-8'?>
<Error>
<Code>BucketNameUnavailable</Code>
<Message>
The requested bucket name is not available.
The bucket namespace is shared by all users of the system.
Please select a different name and try again.
</Message>
</Error>.

Mentre se siamo stati noi ad aver già creato un bucket con quel nome verrà restituito questo messaggio:

"Your previous request to create the named bucket succeeded and you already own it".

Se non ci sono stati errori, il nostro bucket verrà creato con successo e possiamo accertarcene con il comando gsutil ls, che ci restituirà l'elenco dei bucket creati il nostro account.

6.3 Caricare i dati

Per caricare i dati nel bucket che abbiamo appena creato, utilizziamo il comando cp, che copia i dati dalla nostra cartella locale a Google Storage.

Ad esempio:

gsutil cp yob1880.txt gs://tesi/tables/babynames/1880.csv

copia il file di testo yob1880.txt nel bucket "tesi", rinominandolo in tables/babynames/1880.csv.

Per controllare che l'oggetto sia stato caricato con successo possiamo lanciare il comando: "gsutil ls gs://tesi/" ottenendo così una lista di tutti gli oggetti contenuti in quel bucket.

6.4 Creare una tabella

Prima di passare alla creazione della tabella vera e propria si deve creare un file di testo che ne descrive lo schema.

Quindi aprendo un editor di testo, descriviamo lo schema che nel nostro caso è:

```
[
{ "id": "name", "type": "string", "mode": "REQUIRED" },
{ "id": "genere", "type": "string", "mode": "REQUIRED" },
{ "id": "totale", "type": "integer", "mode": "REQUIRED" }
]
```

e salviamo tutto in un file chiamato: baby_schema.

A questo punto, dopo aver caricato sia il file di dati che lo schema della tabella, possiamo creare la nostra prima tabella con il comando:

bq create tesi/tables/babynames/tblNames baby_schema

che andrà a creare una tabella nominata "tables/babynames/tblNames" usando lo schema descritto nel file "baby_schema".

Se non ci sono stati errori viene restituito il seguente messaggio:

```
{
"kind": "bigquery#table",
"name": "tesi/tables/babynames/tblNames"
}
```

che ci indica il tipo e il nome dell'oggetto che è stato appena creato.

6.5 Importare i dati nella tabella

Per importare i dati nella tabella utilizziamo il comando import di bq in questo modo:

bq import tesi/tables/babynames/tblNames tesi/tables/babynames/1880.csv

che importa il file 1880.csv nella tabella tblNames.

Come risposta ci verrà stampato questo messaggio:

```
{
"table": "tesi/tables/babynames/tblNames",
"kind": "bigquery#import_id",
"import": "2c212c1f31a32fa7"
}
```

Nel nostro caso i file contengono al massimo 1000 record, quindi l'importazione durerà pochi secondi, nel caso invece si debba importare molti più dati, nel messaggio di risposta troveremo un ID che corrisponde all'operazione di importazione che abbiamo appena eseguito.

6.6 Controllare lo stato dell'importazione

Per controllare lo stato dell'importazione è possibile utilizzare il metodo import_status dello strumento bq.

Ad esempio il comando:

bq import_status tesi/tables/babynames/tblNames 2c212c1f31a32fa7

che prende come parametro il nome della tabella (compreso di indirizzo) e l'ID dell'operazione di importazione su quella tabella restituisce un messaggio come questo:

```
{
"status": "IMPORT_DONE",
"sources": [ { "uri": "tesi/tables/babynames/1880.csv" } ],
"kind": "bigquery#import"
}
```

6.7 Esecuzione delle query

Se l'importazione di tutti i dati che devono popolare la tabella è terminata con successo, avremo la tabella completamente popolata con i relativi dati, possiamo quindi iniziare a eseguire le query sui nostri dati.

Per eseguire le query si utilizza sempre lo strumento bq, ma questa volta andremo ad usare il metodo query, passando come parametro la query stessa.

Se volessimo selezionare i 5 nomi femminili usati più di frequente tra quelli inseriti nella tabella tblNames potremmo usare questa query:

```
SELECT name,count
FROM [tesi/tables/babynames/tblNames]
WHERE gender = 'F'
```

ORDER BY count DESC
LIMIT 5

In questo modo:

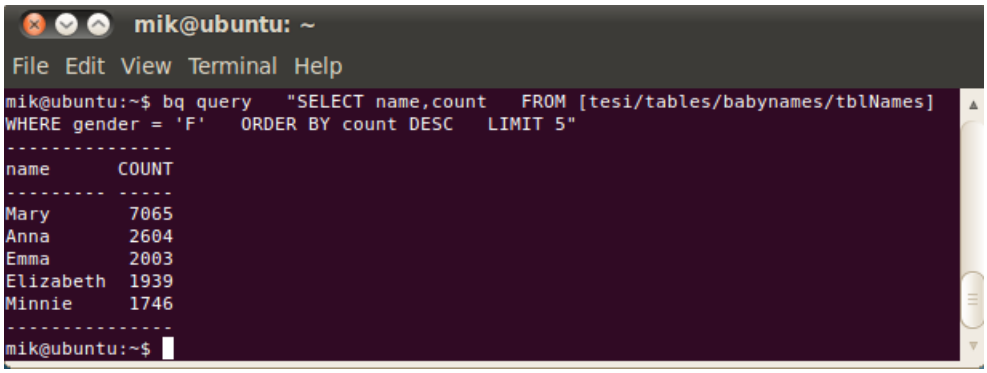

Figura 6.1: I 5 nomi femminili più usati negli Stati Uniti

Mentre se volessimo ottenere i 5 nomi maschili utilizzati più di frequente, avremmo una schermata come questa:

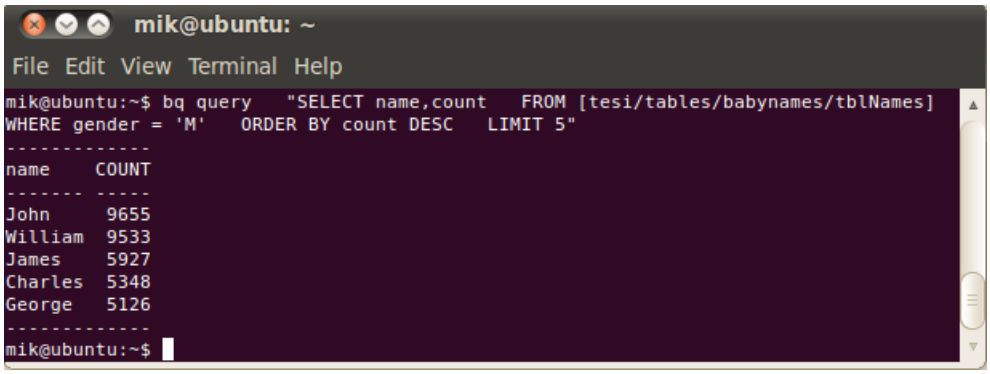

Figura 6.2: I 5 nomi maschili più usati negli Stati Uniti

Capitolo 7

Test su dataset d'esempio

In questo capitolo proveremo a testare un po di più l'efficenza di BigQuery eseguendo query su set di dati un po' più complessi.

Google mette a disposizione alcune tabelle su cui è possibile eseguire query, quella che ritengo più interessante riguarda Wikipedia, contiene tutte le informazioni sugli articoli che si possono trovare in Wikipedia, contiene 314 milioni di record.

Questa tabella registra la storia completa di ogni articolo, tiene traccia di tutte le modifiche fatte per data, per utente ecc..

È possibile trovarla nel bucket nominato "bigquery" con il nome "samples/wikipedia")

7.1 Esaminare una tabella

Prima di cominciare è bene controllare quali dati contiene la tabella che vogliamo analizzare, possiamo quindi ottenerne lo schema usando il comando describe dello strumento bq.

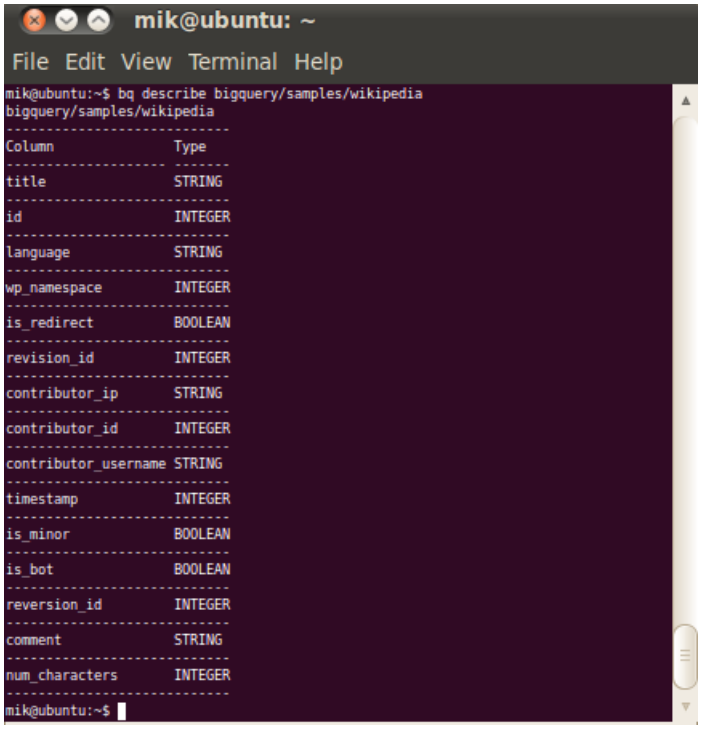

Figura 7.1: Schema della tabelle di Wikipedia

Questa tabella contiene quindi per ogni articolo contenuto nel database di Wikipedia, tutti i suoi dettagli tra cui il titolo, i commenti, la data di inserimento, il namespace[1] in cui viveva ecc...

Se vogliamo essere certi della dimensione del set di dati, possiamo lanciare la query "SELECT COUNT(*) FROM [bigquery/samples/wikipedia]" usando il comando bq.

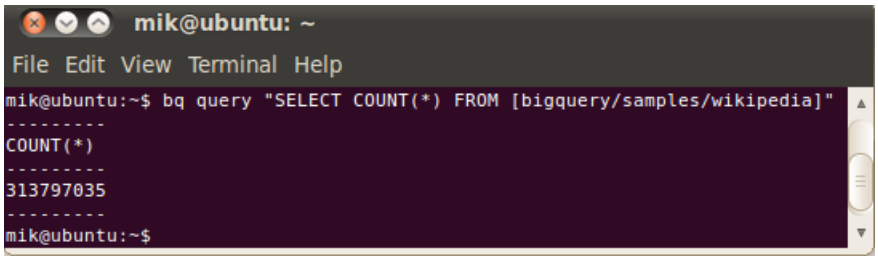

Figura 7.2: Numero di record della tabella di Wikipedia

[1]Lo scopo dei namespace è quello di evitare confusione ed equivoci nel caso siano necessarie molte entità con nomi simili, fornendo il modo di raggruppare i nomi per categorie

Possiamo quindi vedere che si stà lavorando con un set di dati di 313 milione di righe (questo dato è stato ottenuto in meno di un secondo).

7.2 Eseguire delle query

Avendo a disposizione una tabella che riguarda gli articoli di Wikipedia, un'informazione interessante da ricavare è quella di sapere quali sono gli articoli più modificati.

Ricaviamo i 5 articoli più modificati e il numero di volte che sono stati modificati, Wikipedia divide gli articoli in namespace e per considerare quello relativo agli articoli che ci interessano dobbiamo selezionare il namespace 0.

Eseguiamo quindi il comando:

bq query "SELECT TOP(title, 5), COUNT(*) FROM [bigquery/samples/wikipedia] WHERE wp_namespace=0"

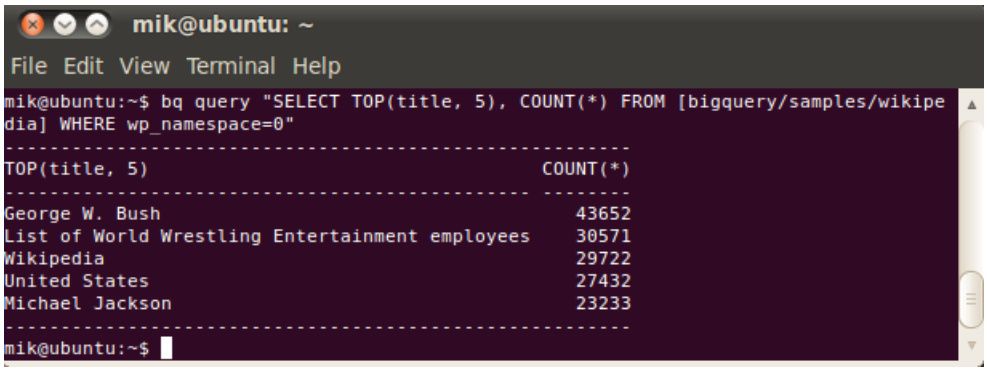

Figura 7.3: Articoli più modificati in Wikipedia

Da questi risultati si può vedere che l'articolo più modificato da quando esiste Wikipedia è l'articolo riguardante George Bush, mentre il secondo articolo più modificato riguarda il wrestling e così via.

Un'altra informazione da estrarre, potrebbe essere quella di sapere quante persone hanno creato o modificato un articolo in Wikipedia, questo si può fare utilizzando questa query:

```
"SELECT Month, Contributors
  FROM (SELECT LEFT(FORMAT_TIME_USEC(timestamp * 1000000), 7) Month, COUNT(DISTINCT contributor_id) Contributors
          FROM [bigquery/samples/wikipedia]
          GROUP BY Month)
  WHERE RIGHT(Month, 2) IN ('01', '04', '07', '10')
  ORDER BY Month"
```

Questa query prima di tutto crea una lista contenente anno/mese e numero di contribuenti distinti che hanno fatto modifiche in quella data (tramite la subquery), fa questo trasformando l'attributo timestamp (moltiplicato per 1 milione) della tabella in una data corrispondente (nel formato YYYY-MM-DD hh:mm:ss [.uuuuu]).

Da questa data utilizzando la funzione LEFT vengono estratti solo i primi 7 caratteri, cioè anno e mese (YYYY-MM).

Nella query esterna vengono mostrati i 2 campi ottenuti dalla subquery e con la clausola WHERE per semplificare i risultati consideriamo solo i mesi di Gennaio, Aprile, Luglio ed Ottobre (usando la funzione RIGHT che estra solo gli ultimi 2 caratteri che corrispondono al mese).

Andiamo quindi a selezionare il numero di utenti distinti che hanno contribuito a Wikipedia in ogni mese iniziando a contare da quando da quando esiste Wikipedia (nella tabella sono contenuti i dati fino a Gennaio 2010).

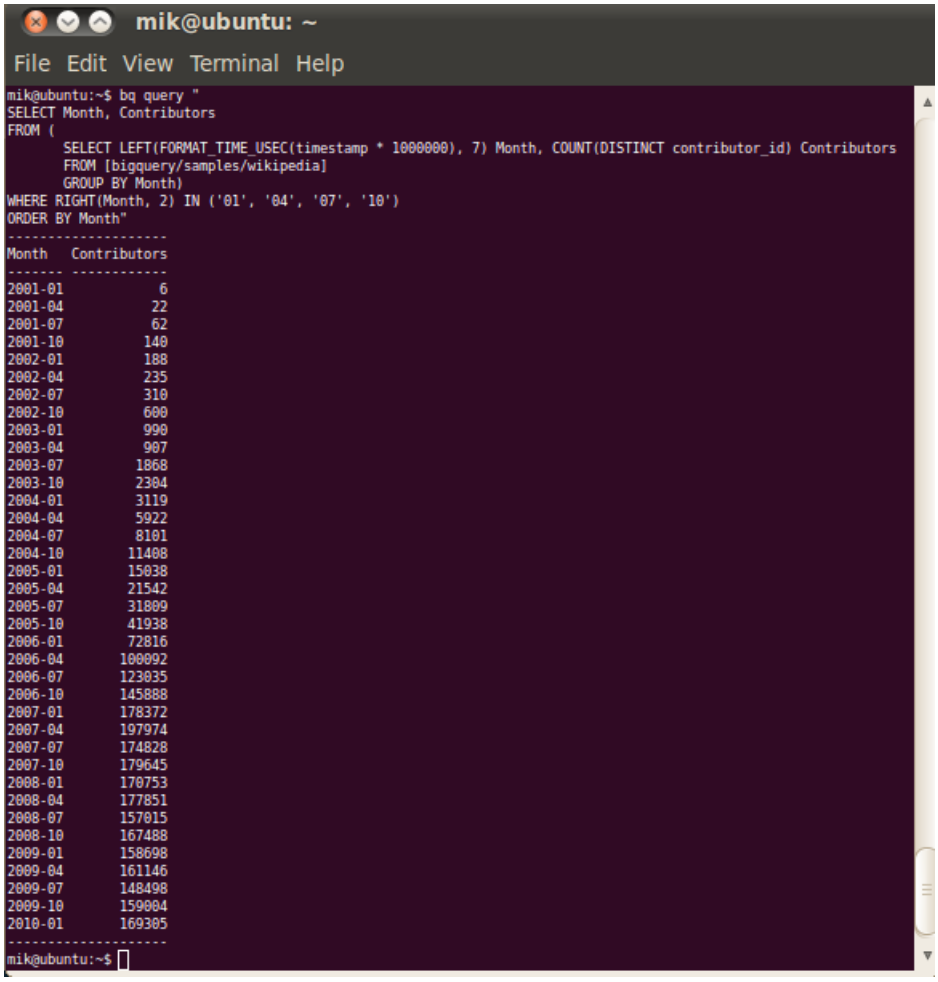

Figura 7.4: Numero di utenti che effettuano modifiche in Wikipedia

Si può notare il continuo aumento di contribuenti distinti nell'avanzare di Wikipedia, anche se è stato un successo con momenti caldi e freddi.

Ora che sappiamo che l'articolo più modificato è quello su George Bush e sappiamo come ottenere il numero di utenti che hanno effettuato modifiche, possiamo riutilizzare questa query e limitarla a soltanto gli articoli che contengono la parola "bush" nel testo, per vedere in quali periodi sono stati modificato più spesso.

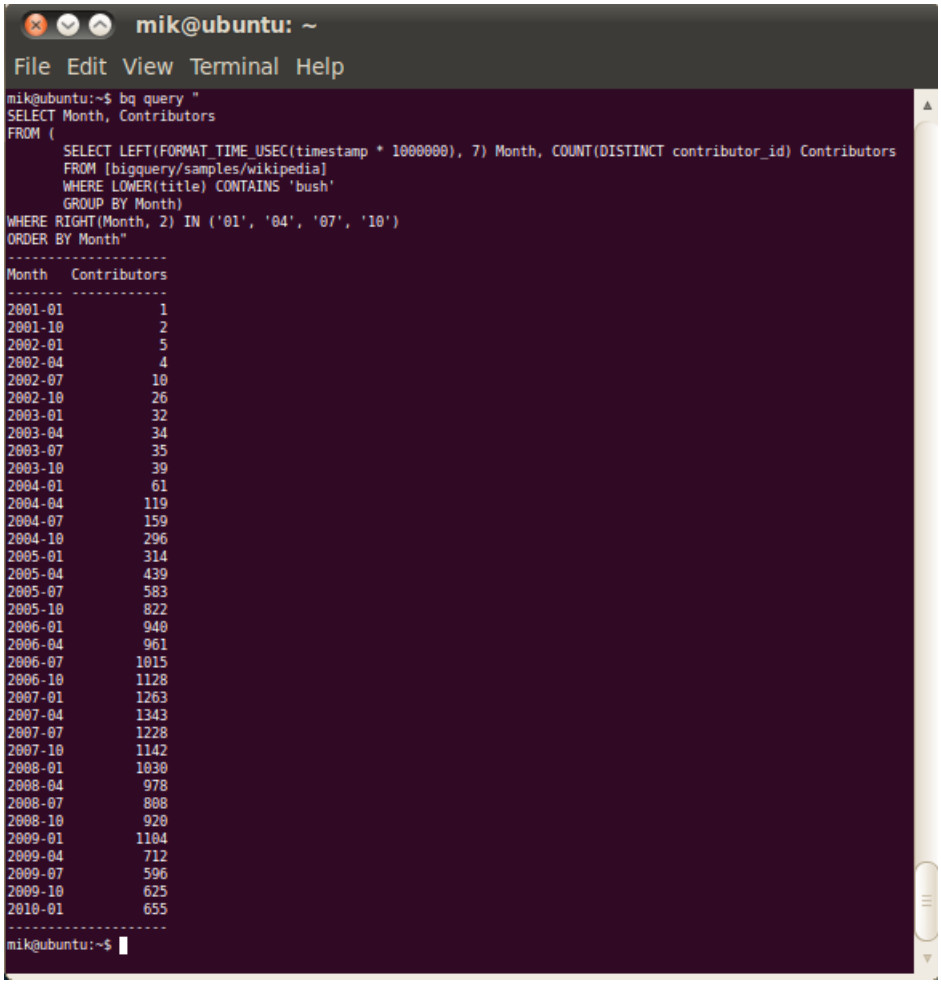

Figura 7.5: Modifiche effettuate agli articoli che contengono la parola 'bush' nel titolo

Da questi dati si può vedere l'aumento di modifiche al suo articolo tra il 2006 e il 2007, cioè quando e' diventato di nuovo presidente.

Una cosa da evidenziare è che non è importante in quale modo si analizzano i dati, ma che tutte queste query si riescono ad eseguire in pochissimi secondi.

L'innovazione data da BigQuery è proprio il fatto di poter eseguire interattivamente query che analizzando milioni di record riescono a darci informazioni molto utili, lasciandosi indietro tempi di risposta anche molto lunghi, che potevano durare anche giorni.

Capitolo 8

Prezzi

I prezzi per questi servizi vengono calcolati in base all'utilizzo che ne viene fatto.

Prima di tutto viene considerata la quantità di dati che archiviamo in Google Storage, la spesa è di 0,17 dollari per GB ogni mese.

Per ogni upload che effettuiamo dobbiamo corrispondere 0,10 dollari per ogni GB, mentre per i download la spesa và dai 0,15 dollari al GB per l'America, l'Europa, il Medio Oriente e l'Africa ai 0,30 dollari per il sud-est asiatico.

Invece per quanto riguarda le operazioni sui dati, queste vengono suddivise in richieste:

- PUT, POST, del costo di 0.01 dollari ogni 1.000 richieste
- GET, HEAD, del costo di 0.01 dollari ogni 10,000 richieste

Questa tabella riassume quali operazioni è possibile fare con queste richieste:

Richiesta	Descrizione
GET Service	Elencare tutti i bucket che si è creato
GET Bucket	Elencare il contenuto di un bucket o scaricare gli ACL che gli sono applicati
GET Object	Scaricare un oggetto o scaricare gli ACL che gli sono applicati
PUT Bucket	Creare un bucket o reimpostare gli ACL su quel bucket
PUT Object	Caricare un oggetto o reimpostare gli ACL su quell'oggetto
DELETE Bucket	Eliminare un bucket
DELETE Object	Eliminare un oggetto
HEAD Object	Elencare i metadati di un oggetto
POST Object	Caricare un oggetto usando il formato HTML

Tabella 8.1: Richieste in Google Storage

Durante l'anteprima, ogni utente riceve fino a 100 GB al mese di storage senza dover spendere niente (se necessario è possibile richiedere un aumento del limite di archiviaziane fino a 1000 GB, ma in questo caso si dovrà pagare la differenza).

Per quanto riguarda le operazioni che si possono eseguire con BigQuery, tutte quelle presentate fino ad ora saranno sempre gratuite, ma in futuro è possibile che verranno presentate nuove funzioni a pagamento.

Capitolo 9

SQL Azure

9.1 Cos'è SQL Azure?

SQL Azure fa parte della piattaforma Windows Azure, un sistema operativo Microsoft (variante di Windows Server 2008) rivolto ad aziende e sviluppatori. Windows Azure mette a disposizione servizi a pagamento come SQL Azure che offre le funzionalità di un database relazionale completo come SQL Server, ma offre anche le funzionalità di un servizio di Cloud computing, ospitato presso i datacenter di Microsoft in tutto il mondo.

Come Google BigQuery è una scelta ideale se si ha necessità di servizi di database che permettano una buona gestione del carico di lavoro, di collaborare con utenti in tutto il mondo e che siano scalabili in base alle nostre necessità, ed essendo servizi a pagamento che permettano un'efficenza in termini di costi a seconda del carico di lavoro di cui si ha bisogno, il cosi detto modello "a consumo" offerto dal Cloud computing che fornisce l'approccio più efficiente in termini di costi.

9.2 Introduzione a SQL Azure

Iniziare ad utilizzare SQL Azure è semplice, specialmente se si ha familiarità con gli strumenti di gestione di SQL Server classici come SQL Server Management Studio[1] (che può essere utilizzato per gestire le istanze di SQL Azure).

Il primo passo consiste nell'impostare un account per la piattaforma Windows Azure che fornisce l'accesso ai relativi servizi tra cui Windows Azure e SQL Azure. Una volta configurato l'account, è possibile accedere al pannello di controllo di SQL Azure.

[1] SQL Server Management Studio è un ambiente integrato per l'accesso, la configurazione, la gestione, l'amministrazione e lo sviluppo di tutti i componenti di SQL Server.

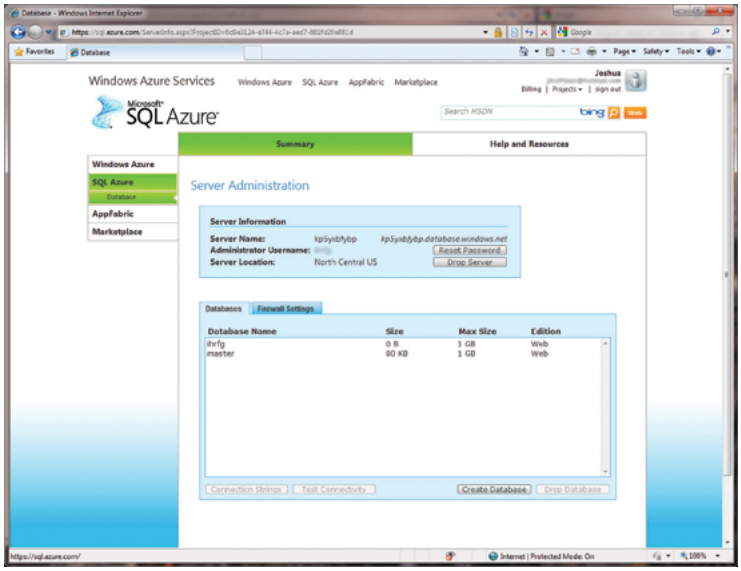

Figura 9.1: Pannello di controllo

Il pannello di controllo di SQL Azure consente di creare nuovi database all'interno del proprio account. È possibile creare 2 tipi di database, il database Web Edition che può supportare fino a un massimo di 5 GB di dati, e il database Business Edition che supporta fino a un massimo di 50 GB. Per la Business Edition vengono utilizzati incrementi di fatturazione pari a 10 GB: 10 GB, 20 GB, 30 GB, 40 GB e 50 GB.

Nel pannello di controllo è inoltre riportato un nome server dell'istanza SQL Azure, nonché stringhe di connessione ADO.NET o ODBC da utilizzare per la connessione al database (basata sulla modalità di autenticazione SQL, da notare che in SQL Azure non è supportata la modalità di autenticazione Windows). Da questa posizione, è possibile utilizzare una delle stringhe di connessione all'interno dell'applicazione, creare un oggetto ODBC o associare le informazioni del server a uno strumento di gestione come SQL Server Management Studio.

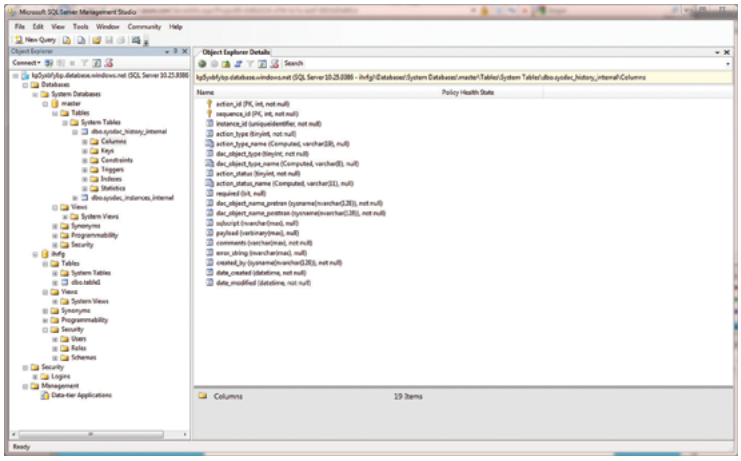

Figura 9.2: SQL Server Management Studio connesso a un database SQL Azure

Come ulteriore misura di sicurezza, all'interno del pannello di controllo di SQL Azure si noterà la finestra "Impostazioni firewall" in cui è possibile specificare gli indirizzi IP dai quali saranno accessibili i database.

L'interazione con i dati archiviati in SQL Azure è identica alla gestione di SQL Server. Mediante SQL Server Management Studio è possibile eseguire tutte le classiche attività di SQL Server come la creazione e la gestione di tabelle, l'importazione di dati e l'esecuzione di istruzioni SQL.

9.3 Integrazione con SQL Server

Esistono diversi modi per integrare SQL Azure con l'infrastruttura SQL Server esistente. La maggior parte dei modi tradizionali per l'integrazione e la migrazione di dati tra server sono disponibili anche per SQL Azure, tra cui SQL Server Integration Services che consente l'importazione, l'esportazione e la trasformazione dei dati. SQL Azure supporta inoltre l'importazione e l'esportazione guidata, integrata nel set di strumenti di SQL Server Management Studio per copiare i database e migrare i dati verso e da SQL Azure.

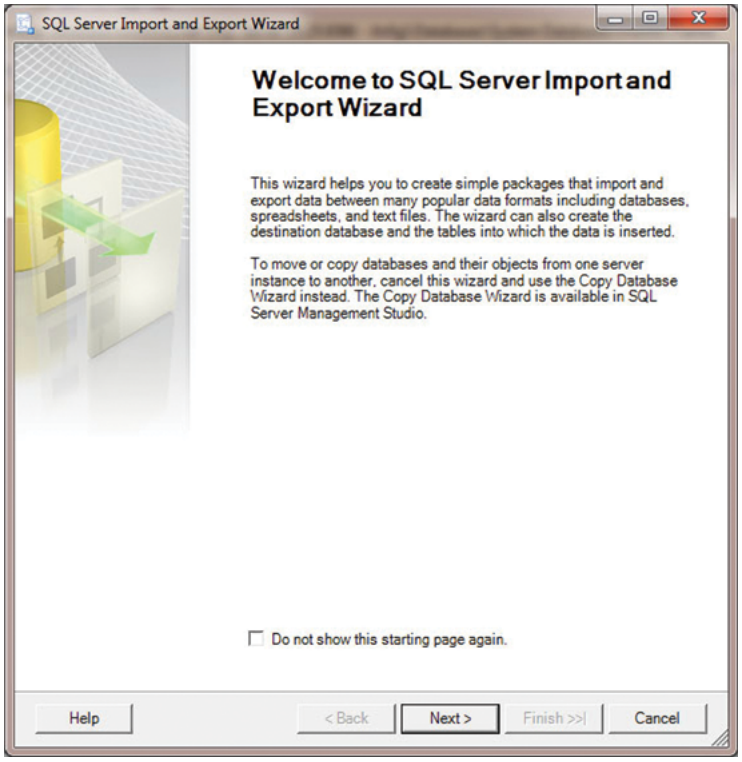

Figura 9.3: Utilizzo di Importazione/Esportazione guidata SQL Server

9.4 Prezzi

Microsoft ha previsto una spesa di 12 centesimi di dollaro all'ora per il computing, 15 centesimi per gigabyte per lo spazio disco e 10 centesimi ogni 10 mila operazioni su disco. Per la banda di rete il prezzo è di 10-15 centesimi per gigabyte trasferito.

9.5 Google BigQuery Vs SQL Azure

Google BigQuery e SQL Azure a parità di servizi offerti sono allo stesso livello, ma se si analizza più a fondo, il prodotto di Microsoft in questo caso ha qualcosa in più, partendo dall'interfaccia grafica molto più intuitiva che in Google BigQuery corrisponde alla linea di comando passando alla gestione dei database veri e propri, con SQL Azure è possibile trasferire interi database in pochi passi, mentre utilizzando Google BigQuery dovremmo trasferire ogni tabella in file CSV caricando e scaricando ogni file sempre da linea di comando.

Tutti questi aspetti per una azienda sono molto importanti, sia in termini di velocità in quanto il lavoro viene svolto in molto meno tempo sia in termini di semplicità perchè

lavorando in un ambiente come quello che propone SQL Azure non si devono conoscere molti aspetti di programmazione che vengono appunto nascosti dall'interfaccia.

Conclusioni

Google BigQuery è di sicuro uno strumento molto potente, e proprio per questo ne consiglierei l'utilizzo solo a chi ha necessità di elaborare grandi quantità di dati dovendo ottenere dei risultati in tempo brevi.

Il cloud computing offre alcuni indubbi vantaggi. Tanto per cominciare, non ci sono costi di manutenzione: i server e la sicurezza dei dati sono affidati all'azienda che gestisce il servizio. Un file affidato ad un servizio cloud è salvato su diversi data center, e ne esistono diverse copie di sicurezza, continuamente aggiornate e monitorate. Di fatto, i dati andrebbero persi solo se tutti i data center dell'azienda andassero in tilt nello stesso momento, un'eventualità più impossibile che remota.

Tra i vantaggi si annovera anche la possibilità e la facilità di condividere un documento, o un file di altro genere, con colleghi e amici, e l'estrema semplicità del lavoro di squadra. Un file online, infatti, può essere modificato da più persone allo stesso tempo, e le modifiche sono disponibili immediatamente per tutti gli utenti collegati.

L'ultimo importante vantaggio legato al cloud computing, in stretta relazione con i costi di manutenzione, è relativo al costo dell'hardware. Nella maggior parte dei casi, i servizi cloud non richiedono PC potenti, quindi sarà possibile usare anche un vecchio PC o un PC di fascia molto bassa, senza alcun problema.

Passando ai lati negativi, il punto che suscita più perplessità è legato alla riservatezza e alla legalità. Per usare un servizio cloud è necessario prima di tutto accettarne i termini d'uso, e poi caricare i propri dati sui server. La conseguenza diretta è che se i server vengono violati, l'hacker avrà accesso ai vostri dati, come se avesse violato il vostro computer.

Ad oggi non si sono mai registrati attacchi seri ai servizi cloud, né si sono registrati furti d'informazioni. Il rischio però esiste, e diventa più concreto mano a mano che cresce la quantità di dati affidati ai server cloud.

Per quanto riguarda i costi, si tratta di spese trascurabili, se comparate con quelle delle licenze software per ogni computer, come nel caso di Microsoft o comunque se si vuole utilizzare un servizio così potente senza comprare e dover gestire l'attrezzatura adeguata, è vero anche che Google in questo modo si crea un business non indifferente centralizzando il servizio e avendo clienti in tutto il mondo.

Occorreranno comunque ancora almeno un paio d'anni prima di assistere al pieno decollo delle tecnologie di cloud computing, le aziende dovranno in primo luogo abituarsi all'idea di dover condividere tutti i loro dati con i provider di servizi.

Bibliografia

[1] Google, documentazione e video della conferenza Google I/O 2010, Google BigQuery, http://code.google.com/apis/bigquery/, 20/11/2010

[2] Google, Documentazione Google Storage, http://code.google.com/apis/storage/docs/, 20/11/2010

[3] Tim Taylor, SQL2CSV, http://sourceforge.net/projects/sql2csv/, 10/11/2010

[4] Wikipedia, articoli vari, http://www.wikipedia.org/, 25/11/2010