ALMA MATER STUDIORUM – UNIVERSITÀ DI BOLOGNA

CAMPUS DI CESENA
SCUOLA DI INGEGENERIA E ARCHITETTURA CORSO DI LAUREA IN INGEGNERIA BIOMEDICA

TITOLO DELL'ELABORATO

Sistemi cloud per l'analisi di big data: BigQuery, la soluzione proposta da Google

Elaborato in Calcolatori elettronici

Sessione Unica

Anno Accademico 2018/2019

Relatore:
Chiar.mo Prof. Luca Roffia

Presentata da:
Gabriele Boretti

Correlatore:

Ing. Cristiano Aguzzi

Indice

Indice delle tabelle

Indice delle figure

ABSTRACT

Lo scopo di questa tesi è quello di analizzare il servizio BigQuery offerto da Google Cloud, strumento utilizzato per l'archiviazione, l'analisi e il monitoring di grandi database. Per fare ciò vengono prima descritte le componenti che lo circondano, offrendo una panoramica completa sul Cloud Computing, partendo dai requisiti minimi per il suo funzionamento, come Internet, fino ad arrivare ai più complessi utilizzi di quest'ultimo, come le intelligenze artificiali. Essendo BigQuery uno strumento per l'analisi di grandi volumi di dati (Big Data), viene fornito un quadro di riferimento sullo sviluppo e utilizzo di quest'ultimi, per porre così le basi per una migliore comprensione di quello che verrà di seguito approfondito. Tutta la tesi è genericamente improntata in ambito biomedico. Gli argomenti affrontati sono tutti ricollegabili all'ambito sanitario, a partire dalle fondamenta della tesi riguardo Internet, collegato con l'Internet of Things (IoT) e con la Telemedicina, dei quali viene mostrato lo stretto collegamento con il Cloud Computing. Anche l'utilizzo delle intelligenze artificiali verrà affrontato in ambito clinico, vedendo i vari utilizzi e vantaggi, fino ad arrivare all'argomento dei data set di carattere medico. La tesi quindi analizza le varie possibilità offerte dall'analisi di dati e per quanto riguarda l'ambito sanitario affronta inevitabilmente l'argomento della privacy e della sicurezza. A questo punto si viene a chiudere il ciclo Internet-Database-Cloud con BigQuery. Dopo una generica descrizione degli altri servizi di offerti dal Google Cloud, necessaria per comprendere come BigQuery possa integrarsi con questi ultimi, la tesi si addentra nell'analisi del suo funzionamento. Viene fornita una descrizione della struttura e dei metodi di analisi disponibili, riportando inoltre alcuni casi d'uso, citando vari articoli e fonti. L'ultima parte della tesi fornisce i risultati derivati dalla sperimentazione dello strumento su diversi dataset, sia contenti dati clinici, che forniti da Google.

Introduzione

Questo capitolo mira a spiegare l'importanza della connessione a Internet, la sua nascita e il suo funzionamento, necessario per l'utilizzo del Cloud, basilare quindi per questa tesi. Il Web è diventato fondamentale in svariate applicazioni e ha facilitato lo sviluppo sociale, in ambito comunicativo e non solo. Dalla sua nascita, quando la connessione si aveva solamente tra reti di computer, si è successivamente ampliata anche a oggetti ad uso diverso, il cosiddetto Internet of Things. In particolare, andando a vederlo in ambito medico, si affronta l'argomento della Telemedicina, necessaria per l'avanzamento e la semplificazione di operazioni sanitarie. Si conclude infine con lo sviluppo più moderno della connessione, applicata per il funzionamento delle intelligenze artificiali, il cui utilizzo sarà ripreso successivamente per un'analisi approfondita e "intelligente" di dati.

Internet e Web

Internet (parola derivata da Interconnected Network) è il sistema globale di reti di computer interconnessi che utilizzano la suite di protocolli Internet (TCP / IP) per collegare dispositivi in tutto il mondo. Si tratta di una rete di reti, che comprende reti private, pubbliche, accademiche, aziendali e governative di portata locale a globale, collegate da un'ampia gamma di tecnologie di rete elettroniche, wireless e ottiche. I mezzi di comunicazione più tradizionali, tra cui telefono, radio, televisione, posta cartacea e giornali sono stati rimodellati, ridefiniti o addirittura surclassati da Internet, dando vita a nuovi servizi come e-mail, chiamate Skype, musica online, giornali digitali e siti web di streaming video. Giornali, libri e altre pubblicazioni di stampa si stanno adattando alla tecnologia dei siti Web o sono rimodellati in blog, feed Web o semplici bacheche di notizie online. Internet ha consentito e accelerato nuove forme di interazione personale attraverso la messaggistica istantanea, i forum su Internet e il social networking. Lo shopping online è cresciuto in modo esponenziale sia per i principali rivenditori che per le piccole imprese e gli imprenditori, in quanto consente alle aziende di estendere la loro area d'azione per servire un mercato più ampio o persino vendere beni e servizi interamente online. Internet non ha un'unica sede centralizzata e di conseguenza neanche una regolamentazione globale nell'implementazione tecnologica o nelle politiche di accesso e utilizzo; ogni rete costituente stabilisce le proprie politiche [1]. Molte persone usano erroneamente i termini Internet e World Wide Web in modo intercambiabile, ma i due termini non sono sinonimi. Internet è l'infrastruttura tecnologica che permette di trasferire dati online e offre servizi quali app per telefono o tablet, app per social media, World Wide Web, posta elettronica, giochi online multiplayer, telefonia via Internet e servizi di condivisione file e streaming. Il World Wide Web (WWW), comunemente noto come Web, è un sistema di informazioni in cui documenti e altre risorse Web sono identificati dagli Uniform Resource Locator (URL), che possono essere interconnessi

dall'ipertesto (link) e accessibili su Internet [2]. Le risorse del WWW sono accessibili agli utenti da un'applicazione software chiamata browser web. Il WWW è stato fondamentale per lo sviluppo dell'Era dell'informazione ed è il principale strumento che miliardi di persone utilizzano per interagire su Internet [3] [4] [5]. Il WWW è accessibile tramite le pagine Web, supporti ipertestuali formattati in Hypertext Markup Language (HTML) [6]. Tale formattazione consente collegamenti ipertestuali incorporati che contengono URL e consentono agli utenti di navigare verso altre risorse Web. Oltre al testo, le pagine Web possono contenere immagini, video, audio e componenti software che vengono visualizzati nel browser Web dell'utente come contenuti multimediali. Più risorse Web con un tema comune, un dominio comune, o entrambi, costituiscono un sito Web. I siti Web originano da speciali computer chiamati server Web. Questi ultimi mantengono online le risorse sopra descritte e sono deputati a rispondere alle richieste effettuate dagli utenti (client) tramite i browser Web. Il sito Web può essere pubblicato da un unico ente o può avere una struttura dinamica e interattiva, nella quale gli utenti contribuiscono e modificano continuamente il contenuto. I siti Web possono essere forniti per una varietà di motivi informativi, di intrattenimento, commerciali, governativi o non governativi. Nonostante il suo crescente ruolo nella comunicazione, il World Wide Web rimane incontrollato: ogni individuo o istituzione può creare un sito Web con qualsiasi numero di documenti e collegamenti. Questa crescita non regolata porta a un Web enorme e complesso, che diventa un grande grafo diretto i cui vertici sono documenti e i cui bordi sono collegamenti (URL) che puntano da un documento all'altro [7]. La topologia di questo grafo determina la connettività del Web e di conseguenza con quale efficacia possiamo localizzare le informazioni su di esso. In questo universo di informazione libero e caotico diventa di primaria importanza riuscire a sviluppare processi di ricerca e indicizzazione in grado di ottenere le informazioni ricercate di volta in volta, ignorando allo stesso tempo la grande quantità di dati non pertinenti. A questa necessità hanno risposto i cosiddetti motori di ricerca come Yahoo! e Google. Un motore di ricerca è un sistema software progettato per eseguire ricerche Web in modo sistematico, per ottenere informazioni specifiche indicate in una query di ricerca testuale. Gli esiti della ricerca sono generalmente presentati come una serie di risultati, ordinati in funzione dell'attinenza riscontrata con le parole chiave utilizzate. Le informazioni ottenute possono essere collegamenti a pagine Web, immagini, video, infografica, articoli, documenti di ricerca e altri tipi di file. Alcuni motori di ricerca hanno anche funzioni di data mining nei database e nelle directory accessibili. Questa funzione si rivelerà basilare per l'argomentazione di questa tesi e sarà ripresa in seguito, poichè rappresenta il processo di analisi di dati, in particolare le tecniche di estrapolazione di informazioni utili da grandi set di dati attraverso anche metodi di intelligence, funzione coperta da BigQuery ed i sistemi con cui interagisce. Il contenuto Internet che non è in grado di essere cercato da un motore di ricerca Web è generalmente descritto come Deep Web. Dal

momento della sua nascita fino ad oggi, il WWW ha attraversato grandi cambiamenti volti a migliorare ed espandere le sue funzionalità; se inizialmente i protocolli necessari al funzionamento del Web supportavano unicamente pagine HTML "statiche", oggi vengono utilizzate soprattutto pagine HTML dinamiche. Mentre le prime vengono immagazzinate nel server e visualizzate in seguito sui client tali e quali, le seconde vengono generate in tempo reale da applicazioni Web attraverso JavaScript. Conseguentemente, le pagine Web dinamiche permettono una maggiore interattività e fluidità durante l'utilizzo. A fianco agli sviluppi delle funzionalità HTML, vi sono stati progressi anche sulla qualità di elaborazione dei server; infatti, sono nati molteplici linguaggi impiegati dal lato server (ad esempio JSP e PHP) che hanno migliorato la qualità di elaborazione, dando vita agli attuali application server. Il linguaggio HTML ha quindi giocato un ruolo fondamentale nello sviluppo del Web come lo conosciamo; eppure, guardando al futuro, sembra presentare delle limitazioni basilari che impediscono di realizzare importanti progressi. L'HTML ha infatti funzionalità puramente di presentazione, senza la minima capacità di interpretazione di input o output. In altre parole, l'HTML permette di visualizzare una pagina Web elaborando correttamente ogni stringa di codice ma durante questo processo non vi è nessuna consapevolezza del materiale trattato, né capacità di distinguere gli elementi del codice stesso. Una soluzione a questo problema è offerta dal progetto del cosiddetto Web Semantico [8] . Mentre HTML descrive i documenti e i collegamenti tra di essi, i linguaggi del Web Semantico descrivono oggetti arbitrari quali persone, date, eventi. L'idea che muove questo progetto è quella di creare un Web di applicazioni intelligenti, ossia in grado di guidare l'utente umano verso le informazioni ricercate e rendere completamente automatiche le operazioni meno complesse. Il Web Semantico mira a rendere le applicazioni in grado di verificare l'attendibilità delle informazioni, navigare di pagina in pagina seguendo processi logici e comprendere almeno a livello basilare il significato degli elementi contenuti in rete. Il tutto finalizzato a guidare e facilitare le attività dell'utente in funzione delle sue esigenze.

IoT e applicazioni mediche

Internet of Things (IoT) è un neologismo riferito all'estensione della connettività di Internet ai dispositivi fisici e agli oggetti di uso quotidiano. Analisi in tempo reale, sistemi intelligenti in grado di apprendere, sensori wireless e controllo remoto pongono le basi per una rete interattiva e dinamica, capace di integrare informazioni e raggiungere scopi specifici [9] [10] [11] [12] [13]. Il termine "Internet-of-Things" è usato come parola chiave a ombrello per coprire vari aspetti legati all'estensione di Internet e del Web nel mondo fisico, mediante la diffusione su larga scala di dispositivi distribuiti spazialmente con identificazione, rilevamento e / o capacità di attuazione. IoT

prevede un futuro in cui le entità digitali e fisiche possano essere collegate, mediante appropriate tecnologie di informazione e comunicazione, per consentire una nuova classe di applicazioni e servizi. Riguardo allo sviluppo di IoT finalizzato all'uso da parte dei consumatori, si ricordano veicoli connessi, domotica, tecnologia indossabile, connected health e apparecchiature con funzionalità di monitoraggio remoto [14], [15]. IoT è strettamente associato al concetto di "sistema intelligente", un'entità virtuale programmata per sfruttare in modo efficiente ogni dispositivo elettronico disponibile nell'ambiente per facilitare ogni tipo di attività umana. Di conseguenza troviamo impieghi di IoT nelle "smart cities", città intelligenti, in forma di energy management, videosorveglianza e monitoraggio del traffico e dell'ambiente. IoT è applicato anche in agricoltura, dove promuove l'automatizzazione delle tecniche di coltivazione e la presa di decisioni informate che minimizzino rischi e sprechi. In ambito biomedicale questa nuova tecnologia porta enormi vantaggi sotto vari punti di vista, sia a livello economico che a livello operativo. Infatti, numerose tecnologie possono ridurre i costi complessivi per la prevenzione o la gestione delle malattie croniche. Queste includono i dispositivi che monitorano costantemente gli indicatori di salute, quelli che eseguono l'auto-somministrazione di terapie e quelli che tracciano i dati sanitari in tempo reale quando un paziente gestisce da solo una terapia. Grazie alla maggiore accessibilità a una buona connessione a Internet raggiunta negli ultimi anni, molti pazienti hanno iniziato a utilizzare applicazioni mobili (app) per gestire vari bisogni di salute. Tali dispositivi e app mobili sono sempre più utilizzati e integrati con la telemedicina tramite l'Internet of Things medico (IoMT). L'IoMT è un'applicazione dell'IoT specificamente per scopi medici e sanitari; ad esempio, monitoraggio e raccolta di dati e analisi per la ricerca [16] [17] [18] [19]. Il progetto "Smart Healthcare" [20], ha portato per esempio alla creazione di un sistema sanitario digitalizzato, che collega le risorse mediche disponibili e i servizi sanitari [21]. Infatti, l'accesso al Web rende possibile la condivisione di grandi quantità di dati per analisi statistiche a livello mondiale, fondamentali per la prevenzione ed il miglioramento della salute umana. Negli ultimi anni, per facilitare la connessione tra dispositivi medici eterogenei e diversi tra loro, è stato introdotto "DICOM" [22] (Digital Imaging and Communications), uno standard per la comunicazione dei dati di "Imaging Diagnostico". Di conseguenza, i vari dispositivi medici risultano uniformati a norme comuni (come la DICOM 3.0) che gli permettono di interfacciarsi tra loro, migliorando così il passaggio di informazioni nel sistema sanitario. DICOM è diventato difatti il linguaggio usato nel PACS [23] (Picture archiving and communication system), il software ospedaliero necessario per l'archiviazione e la trasmissione di immagini digitali, che si trova quindi ad interfacciarsi con gli altri sistemi ospedalieri, come RIS (Radiology Information System) [24] o HIS (Hospital Information System), i quali invece usano lo standard HL7 [25]. L'utilizzo di reti

centralizzate tipiche dell'IoT vede largo uso anche nelle sale ospedaliere, dando modo di monitorare molteplici pazienti contemporaneamente [26].

Telemedicina

Telemedicina, o "In absentia cura" è probabilmente esistita da più di 1500 anni. Questa pratica deve la sua perenne longevità:

- Alla necessità, la convenienza e, in alcuni casi, l'avidità dei medici.
- Ai trasferimenti difficoltosi (esigenza dei pazienti che non potevano recarsi a vedere un medico in persona).
- Per arrivare a una diagnosi, l'esame fisico si era visto come meno importante che sentire la storia del paziente. Così, il contatto personale, sebbene auspicabile, poteva non essere essenziale.

La telemedicina [27] non è assolutamente un fenomeno nuovo: il medico che pratica la telemedicina è semplicemente una variazione tecnologica su un tema vecchio come la medicina stessa. La telemedicina è la consegna a distanza di assistenza sanitaria, ad esempio valutazioni o consulenze sulla salute, tramite l'infrastruttura delle telecomunicazioni. Essa consente agli operatori sanitari di valutare, diagnosticare e trattare i pazienti utilizzando tecnologie comuni, quali videoconferenze e smartphone, senza la necessità di una visita di persona. La telemedicina è uno strumento a servizio del paziente, che gli permette di consultarsi con un professionista per esigenze mediche minori o non urgenti, invece di rivolgersi all'ufficio di un medico di base o al pronto soccorso. Si può ottenere una classificazione basata sul tempo:

- **Real time**. Richiede la presenza di entrambe le parti allo stesso tempo e un canale di comunicazione tra loro che permetta di dare luogo a una interazione in tempo reale. Le apparecchiature per video-conferenza sono una delle forme più comuni di tecnologia utilizzate in tale scopo.
- **Store-and-forward.** Comporta l'acquisizione di dati medici (ad esempio immagini mediche o dati acquisiti da sensori biomedici) e quindi la trasmissione di questi dati ad un medico o a uno specialista in un tempo utile per la valutazione offline. Non richiede la presenza di entrambe le parti allo stesso tempo.

Riguardo alla situazione in Italia, l'evoluzione della dinamica demografica ha portato negli ultimi anni a un forte aumento della popolazione anziana e conseguentemente della necessità di assistenza sanitaria per trattare malattie croniche o non urgenti. La telemedicina sfrutta le moderne tecnologie

di comunicazione per spostare il fulcro dell'assistenza sanitaria dagli ospedali al territorio, facilitando l'accesso alle prestazioni di specialisti lontani dal paziente e permettendo la diffusione di protocolli diagnostico-terapeutici all'interno della comunità scientifica. Un punto di forza della telemedicina è costituito dalla varietà di dispositivi e sensori che permettono di rilevare informazioni specifiche sullo stato di salute del paziente (incluse eventuali emergenze) e recapitarle al medico in tempo reale tramite Internet. I dispositivi IoT di questo tipo vanno dai monitor della pressione arteriosa e della frequenza cardiaca agli strumenti avanzati in grado di monitorare impianti specializzati, come pacemaker, braccialetti elettronici Fitbit [28] o apparecchi acustici avanzati. Similmente, l'avvento di applicazioni che permettono all'utente di gestire la propria salute (ad esempio l'ipertensione arteriosa e le condizioni diabetiche) promuove una cultura del benessere e della prevenzione tramite l'uso di dispositivi comuni come smartphone e tablet. Un esempio di applicazione della telemedicina si ha nell'utilizzo della cartella clinica digitale. Gli obiettivi di una cartella clinica sono: facilitare il processo di cura del paziente, la raccolta cronologica degli eventi caratterizzanti il processo di cura, la comunicazione fra il personale coinvolto, la raccolta dei dati a fini medico/legali e la possibilità di compiere ricerche statistich [29]. Sezioni della cartella clinica:

- *Anagrafica*: contiene i dati anagrafici del paziente, come nome, cognome, data di nascita, indirizzo, professione e recapiti. Dati utili per la stratificazione statistica dei dati.
- *Accettazione*: assegnazione del codice al paziente, registrazione del reparto in cui il paziente sarà degente e il motivo del ricovero.
- *Anamnesi*: storia clinica del paziente.
- *Esame obiettivo*: informazioni acquisite durante la visita ambulatoriale.
- *Diario Clinico Giornaliero*: insieme delle registrazioni che accadono durante il ricovero.
- *Richiesta accertamenti*: come visite specialistiche o esami diagnostici.
- *Dimissione*: scheda di dismissione ospedaliera (SDO) e lettera di dismissione.

La cartella clinica cartacea è divenuta sempre più voluminosa, con contributi provenienti da moltissime fonti differenti, aumentando quindi la difficoltà di trovare tempestivamente le informazioni necessarie. La distinzione proposta dal Servizio Sanitario Inglese (NHS), tra le due soluzioni estreme all'interno dei sistemi informativi clinici è la seguente:

- La **"Cartella clinica elettronica locale"**, cioè limitata ad una singola struttura sanitaria; questa soluzione viene chiamata "Electronic Patient Record";
- Il **"Fascicolo Sanitario Personale"**, cioè forme più complete di servizio che prevedono una qualche modalità di integrazione e di accesso in rete, su dati provenienti da applicazioni

cliniche eterogenee. Queste varie forme vengono denominate genericamente “Electronic Health Record”.

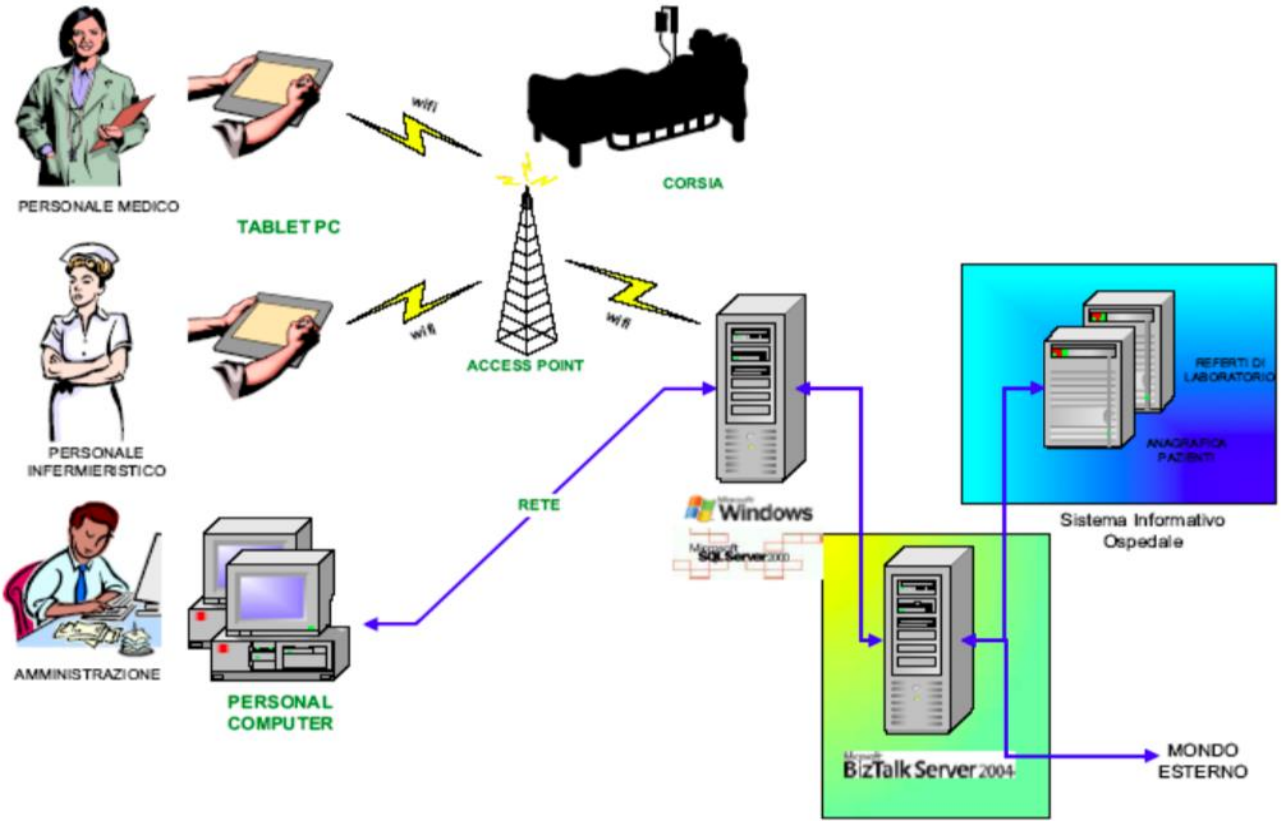

Figura 1 Rappresentazione schematica della cartella clinica elettronica digitale. Le informazioni su trascorsi medici, esami, e prescrizioni vengono integrate da molteplici canali di entrata. I dati provenienti da personale medico, infermieristico e amministrativo vengono convogliati attraverso dispositivi di vario tipo e risultano facilmente accessibili all’occorrenza. L’immagine evidenzia anche come l’utilizzo della cartella clinica permetta una comoda trasmissione e condivisione dei dati sanitari, eliminando la necessità delle cartelle cartacee (e quindi il rischio di perdere dati importanti). [L’immagine riportata è presa dalle slide del corso di Informatica Medica e Reti di Telemedicina del corso di Ingegneria Biomedica UNIBO]

Un esempio di un sistema più completo che sfrutta l’utilizzo di Internet in ambito medico si ha nel progetto “IoT per le attività di telemedicina abilitate da un'applicazione AndroidTM con integrazione del sistema cloud” [30]. I dispositivi medici collegati stanno già aprendo la strada verso la realizzazione di Smart Hospitals [31] in cui la cura del paziente è migliorata grazie alle soluzioni Internet of Things (IoT). Si propone una rete basata su IoT per il monitoraggio on-line dell’anestesia. Questa architettura consente all'anestesista di rimanere collegato contemporaneamente a tutti i pazienti sedati tramite un'app Android. Inoltre, i dati medici dei pazienti possono essere condivisi su una soluzione Cloud accessibile da un'applicazione Web che consente il teleconsulto. Una definizione di Cloud e la presentazione di soluzioni tra loro concorrenti saranno date nel seguito. Accedendo al Cloud gli specialisti in medicina possono consultare i dati condivisi da qualsiasi luogo e in qualsiasi momento. La flessibilità e la portabilità di questa architettura di monitoraggio consentono la possibilità di interfacciarsi con qualsiasi dispositivo medico che può inviare i dati misurati in modalità

wireless. Pertanto, possono essere indirizzate anche altre applicazioni di monitoraggio medico.

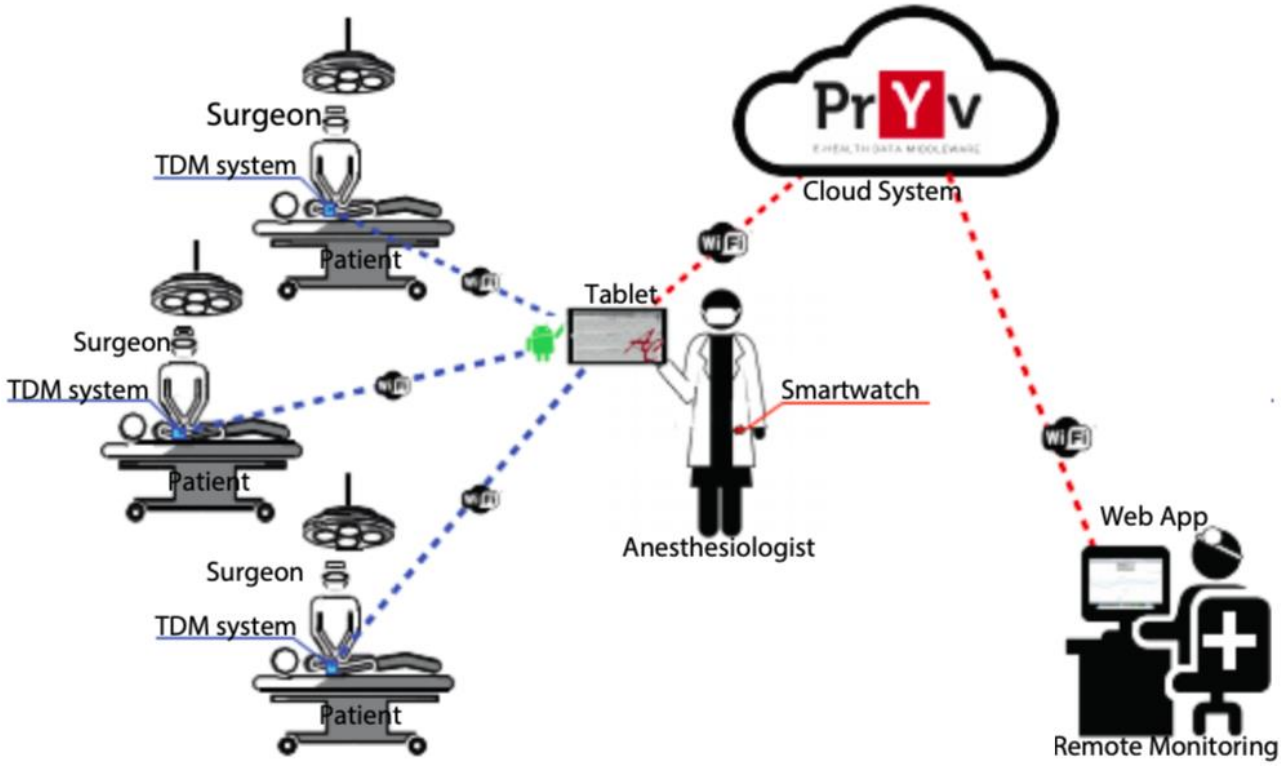

Figura 2 Rappresentazione della rete di monitoraggio IoT basata sul cloud per la pratica anestetica. Il supporto digitale fornito dal tablet consente allo specialista di avere contemporaneamente sotto controllo i parametri fondamentali di molteplici pazienti. La scelta di un sistema operativo vastamente diffuso come Android garantisce un'ottima accessibilità al progetto. I dati registrati in questo modo vengono successivamente caricati sul Cloud, ciò permette di consultare in remoto le informazioni raccolte, ma anche di studiarle e analizzare l'efficacia delle tecniche utilizzate. [L'immagine illustrata è presa dal conference proceedings di F. Stradolini, N. Tamburrano, T. Modoux, A. Tuoheti, D. Demarchi e S. Carrara, «IoT for telemedicine practices enabled by an Android™ application with cloud system integration.,» 2018 IEEE international symposium on circuits and systems (ISCAS), *pp. 1-5, 2018.]*

Un altro esempio di progetto che vede l'utilizzo dell'IoT è il progetto HABITAT [32], che sfrutta queste nuove tecnologie per ottenere abitazioni "intelligenti", che si adattano alle esigenze di chi le fruisce e che sono quindi riconfigurabili a seconda delle necessità dei diversi inquilini. In particolare, il progetto utilizza tecnologie dell'IoT per arricchire oggetti di uso comune (come radio, poltrone, orologi ecc..) rendendoli così "smart objects", tra loro interoperabili. Viene creato così un sistema "trasparente" che monitora i comportamenti quotidiani dei soggetti. Il fine ultimo del progetto è quello di dare la possibilità a soggetti bisognosi di assistenza, in particolare a persone anziane, di continuare a vivere nella propria abitazione in completa sicurezza, aiutandole a svolgere in autonomia la maggior parte delle attività legate allo svolgimento dei bisogni primari quotidiani. Il progetto vede l'integrazione delle più avanzate tecnologie come identificazione a radio-frequenza (RF-ID), *wearable electronics*, *wireless sensor networks* (WSN) e intelligenza artificiale (IA), di cui parleremo in seguito. Un esempio di "smart object" è la poltrona intelligente ERGOTEK, dotata di sensori di pressione, un sistema per l'identificazione della persona basato su TAG RF-ID [33] e attuatori intelligenti per il supporto dell'alzata del soggetto.

Intelligenza artificiale

Per sfruttare la mole di dati che si trovano sul web si utilizzano software che elaborano dati, calcolatori elettronici che ricavano informazioni specifiche. L'informatica definisce la ricerca di Intelligenza Artificiale (IA) come lo studio di "agenti intelligenti": qualsiasi dispositivo che percepisca il suo ambiente e intraprenda azioni che massimizzino le sue possibilità di raggiungere con successo i suoi obiettivi [34]. Una definizione più elaborata si riferisce all'intelligenza artificiale come alla “capacità di un sistema di interpretare correttamente i dati esterni, di apprendere da tali dati e di utilizzare tali apprendimenti per raggiungere obiettivi e compiti specifici attraverso un adattamento flessibile" [35]. Le forme più moderne di intelligenza artificiale sono in grado di generare una rappresentazione cognitiva del mondo e utilizzare l'apprendimento basato sull'esperienza passata per informare le decisioni future. Inoltre, presentano elementi dell'intelligenza emotiva; ad esempio, comprendere le emozioni umane e considerarle nel loro processo decisionale o essere consapevoli delle interazioni con gli altri. Nell’ultimo secolo, le tecniche di IA hanno sperimentato una rinascita in seguito a concomitanti progressi nella potenza del computer, grandi quantità di dati ed evoluzione a livello teorico. Le tecnologie di IA sono diventate una parte essenziale dell'industria tecnologica, contribuendo a risolvere molti problemi complessi in informatica, ingegneria del software e ricerca operativa [36], [37]. L'intelligenza artificiale trova impieghi per qualsiasi compito intellettuale. Paradossalmente, quando una tecnica IA diventa di uso quotidiano, non viene più considerata intelligenza artificiale; questo fenomeno è conosciuto come l'effetto IA. Esempi di IA attualmente rilevanti includono veicoli autonomi (come droni e auto a guida autonoma), diagnosi medica, dimostrazione di teoremi matematici, giochi (come scacchi), motori di ricerca (come Google), assistenti online (come Siri), riconoscimento delle immagini nelle fotografie, filtro antispam, previsione dei ritardi dei voli, previsione delle decisioni giudiziarie e targeting di pubblicità online. I sistemi di diagnosi medica intelligenti verranno ripresi successivamente, riportando vari esempi, poichè alcuni tra di essi sono basati sull’analisi intelligente di grandi quantità di dati, che possono variare da dati di immagini (presi da sistemi di imaging diagnostico) ad altri dati (come valori di analisi del sangue o di altre analisi). Con l’avvento dei social media e il progressivo superamento di media come la TV, le organizzazioni giornalistiche diventano sempre più dipendenti dalle piattaforme social per generare distribuzione [38]; di conseguenza, i principali editori ora usano la tecnologia di intelligenza artificiale per pubblicare storie e tweet in modo mirato, generando maggiori volumi di traffico [39]. In ambito clinico, l’IA ha introdotto grandi miglioramenti sia nella gestione dei pazienti in ambito domestico (prevalentemente grazie all’uso di robot assistenti) che nelle sale ospedaliere, dove software intelligenti supportano gli specialisti nell’interpretazione delle lastre, nella formulazione di diagnosi accurate e nel corso di attività estremamente delicate di chirurgia. Un

elemento chiave della moderna IA è costituito dalla capacità di apprendimento, grazie al quale le esperienze pregresse vengono immagazzinate e costituiscono informazioni utili per affrontare situazioni simili in futuro, proprio come accade negli esseri umani. Questa branca dell'IA prende il nome di Machine Learning, o Apprendimento Automatico. Gli algoritmi di apprendimento automatico costruiscono un modello matematico basato su dati campione per fare previsioni senza essere programmati esplicitamente per eseguire attività specifiche [40]. L'apprendimento automatico è quindi strettamente correlato alle statistiche computazionali. Nello specifico, esistono due principali approcci all'apprendimento automatico. Il primo si fonda sui meccanismi di apprendimento per rinforzo [41], il quale prevede una rete di commutazione connessa casualmente e delle routine di apprendimento basate sul rinforzo; la condotta che ha portato alla soluzione di un problema viene rinforzata mentre quella che ha fallito in tale compito sarà esclusa in futuro. Il secondo metodo è basato sulla riproduzione di una rete altamente organizzata [35] che apprenda in modo mirato solo alcune attività. Tale approccio risulta più efficiente ma necessita di supervisione e richiede una riprogrammazione ogni volta che viene applicato a un nuovo ambito. Entrambi i due metodi di apprendimento automatico, e in particolare l'apprendimento per rinforzo, saranno ripresi in seguito per affrontare il servizio offerto da Google di BigQuery ML, basato su avanzate tecnologie di Machine Learning e portatore di grandi vantaggi sotto diversi ambiti. Gli algoritmi di apprendimento automatico sono utilizzati in un un'ampia varietà di situazioni, come il filtraggio delle e-mail e la visione artificiale [42]. Quest'ultima è impiegata nel riconoscimento facciale e nell'elaborazione del linguaggio naturale, comprendente del contenuto semantico, dell'espressione, dell'intonazione e in generale della comunicazione sia verbale che non. La visione artificiale è alla base di intelligenze artificiali conosciute come Alexa e Siri. Recentemente è stato condotto uno studio che indagava la possibilità di diagnosticare la depressione basandosi sulla grande quantità di foto pubblicate sui social; tale ricerca è resa possibile dall'applicazione di un software di riconoscimento facciale che analizza le espressioni e le situazioni, offrendo giudizi sulle condizioni delle persone ritratte in foto [43]. Rimanendo sull'argomento dell'analisi delle espressioni, si ricorda lo sviluppo di recenti applicazioni capaci di valutare lo stato d'animo delle persone. In queste analisi vengono considerati molteplici parametri rilevabili quali comunicazione non verbale, respirazione, quantità di attività fisica e perfino il numero di volte in cui l'utente ha toccato lo schermo dello smartphone o del tablet. Anche nelle dipendenze e nella gestione del diabete sono state introdotte app di questo tipo e gli ambiti applicativi vengono rinnovati continuamente. Proprio riguardo la gestione del diabete, patologia sempre più diffusa nella società moderna (in particolare nei paesi più sviluppati), saranno citati successivamente alcuni approfondimenti che vedono l'integrazione di BigQuery per la lotta contro questa malattia. Le ricerche sopra descritte potrebbero apparire poco rilevanti sul piano clinico,

ma giocano un ruolo importante nello sviluppo di tecniche di prevenzione e cura di patologie psichiatriche e disturbi psicologici. Le applicazioni di questo tipo sono definite famaci digiceutici [44] e si propongono come vere e proprie terapie digitali. Sempre nel settore clinico, un forte impatto del machine learning è stato riscontrato nell'analisi e nella gestione dell'enorme mole di dati sulla salute (Big data); infatti, i software di IA deputati alle funzioni di data mining (letteralmente "estrazione di dati") permettono di scoprire nuovi pattern tra i dati già esistenti, introducendo nuove tecniche di ricerca e possibilità di prevedere lo sviluppo di patologie e comorbidità nei pazienti. I Big Data sono moli di dati, eterogenee, destrutturate, difficili da gestire attraverso tecnologie tradizionali, che hanno reso necessario lo sviluppo di potenti sistemi di analisi proprio come BigQuery. L'analisi di Big data a fini di previsioni strategiche è largamente sfruttata anche nel settore economico, dove prende il nome di Business Intelligence (BI). La BI comprende le strategie e le tecnologie utilizzate dalle imprese per l'analisi dei dati delle informazioni aziendali [45]. Le tecnologie di BI forniscono una visione storica, attuale e predittiva delle operazioni aziendali. Funzioni comuni delle tecnologie di business intelligence includono reporting, elaborazione analitica online, analisi, data mining, estrazione di processi, elaborazione di eventi complessi, gestione delle prestazioni aziendali, benchmarking, text mining, analisi predittiva e analisi prescrittiva. Le tecnologie di BI possono gestire grandi quantità di dati strutturati e talvolta non strutturati per aiutare a identificare, sviluppare e creare altrimenti nuove opportunità strategiche di business. Mirano a consentire una facile interpretazione di grandi quantità di dati. Le informazioni acquisite dai Big data permettono di identificare nuove opportunità e implementare strategie efficaci, offrendo alle aziende un vantaggio competitivo sul mercato e una stabilità a lungo termine [46]. La BI è impiegata pertanto in una varietà di attività commerciali, ad esempio, le metriche sulle prestazioni e il benchmarking informano i dirigenti aziendali sui progressi verso gli obiettivi aziendali (gestione dei processi aziendali), le analisi dei dati e dei pattern in essi contenuti permettono di elaborare decisioni ottimali e minimizzare i rischi e i report aziendali possono utilizzare i dati BI per informare la strategia da adottare. La BI può inoltre facilitare la collaborazione all'interno e all'esterno dell'azienda consentendo la condivisione e lo scambio dei dati elettronici. In sintesi, la BI unisce le tecnologie di raccoglimento dei dati, immagazzinamento dei dati e gestione delle conoscenze per fornire un supporto a una grande varietà di processi decisionali in ambito commerciale.

Database

La connessione ad Internet da parte dei vari dispositivi e la possibilità da parte di chiunque della creazione di moltitudini di pagine Web affrontati nel capitolo precedente, ha portato, oltre a vari vantaggi in ambito di sviluppo sociale, ad una grandissima quantità di dati vaganti nel Web. I Big Data rappresentano un grosso potenziale di sviluppo sotto diversi ambiti, tra cui anche quello sanitario. In questa tesi proponiamo BigQuery come mezzo per trasformare i dati in informazioni utili, così da migliorare vari aspetti dell'analisi, fino ad arrivare, grazie all'integrazione con intelligenze artificiali, ad applicazioni avanzate come quella della diagnosi accurata di malattie o quella di previsione di vendite commerciali. In questo capitolo si vuole analizzare il funzionamento e lo sviluppo che i database hanno subito negli anni, le varie tipologie in cui si possono trovare ed i diversi metodi per studiarli ed analizzarli, soppesando le differenze tra le due grosse tipologie di set di dati, quelli SQL (analizzabili tramiti BigQuery) e quelli NoSQL.

Introduzione

Si definisce un Database come:

"Archivio elettronico di dati correlati, registrati nella memoria di un computer e organizzati in modo da poter essere facilmente, rapidamente e selettivamente rintracciabili uno per uno, oppure per gruppi determinati, mediante appositi programmi di gestione e di ricerca (chiamati anch'essi data base, ma più propriamente denominati data base management system, in sigla DBMS)."

Cit. Treccani

Un database quindi è l'insieme degli strumenti per memorizzare e manipolare dati in modo "efficiente ed efficace" allo scopo di ottenere informazioni utili in un preciso contesto. Risulta fondamentale distinguere il concetto di dato dal concetto di informazione:

- I dati sono fatti conosciuti, che possono essere memorizzati e che hanno significato implicito, che diventa esplicito solo in seguito ad un determinato contesto o ad una elaborazione specifica.
- Le informazioni, invece, sono tutto ciò che modifica la nostra conoscenza, hanno quindi significato esplicito.

Un dato rappresenta un'informazione di interesse solo a coloro per i quali la base di dati è stata costruita. Per poter interpretare i dati e ricavare quindi informazioni, è necessario l'utilizzo di un modello. I modelli definiscono il nostro modo di conoscere il mondo, interpretare ciò che vediamo e applicare la nostra conoscenza per realizzare dei cambiamenti, attraverso azioni e tecnologia. Quindi

un modello dipende dai presupposti di partenza e dallo scopo per il quale è stato costruito. Si basa su due presupposti fondamentali:

1. Non si può sviluppare un modello unicamente veridico di un oggetto.

2. Non esiste un progetto generale adatto ad ogni scopo.

Più un modello ci fornisce informazioni sulla fonte e sul contenuto dei dati, più il potenziale interpretativo è ricco. Considerato un insieme di dati, modelli diversi possono dare informazioni diverse, quindi alcuni saranno più adatti per certi fini rispetto ad altri. Esistono varie tipologie di modelli, tra cui il gerarchico, il reticolare e il relazionale. Le banche dati possono essere composte da dati relativi a svariati campi, da informazioni sanitarie e identificative di persone, a codici economici o informatici, utili ai vari calcolatori per poter operare nei rispettivi ambiti. Database più rinomati e utili per la navigazione sul Web e per l'associazione di informazioni sono ad esempio l'ImagineNet e il WordNet [47]. Quest'ultimo in particolare permette la "comprensione" di un testo da parte del calcolatore. Dato che le frasi significative sono composte da parole significative, qualsiasi sistema che spera di elaborare le lingue naturali, come per le persone, deve avere informazioni sulle parole e sui loro significati. Queste informazioni sono tradizionalmente fornite tramite dizionari, e i dizionari a lettura automatica sono ora ampiamente disponibili. Ma le voci del dizionario si sono evolute per la comodità dei lettori umani, non per le macchine. WordNet offre una combinazione più efficace di informazioni lessicografiche tradizionali e calcolo moderno. È un database lessicale online progettato per essere utilizzato sotto il controllo di un programma. I nomi, i verbi, gli aggettivi e gli avverbi inglesi sono organizzati in gruppi di sinonimi, ognuno dei quali rappresenta un concetto lessicalizzato. Le relazioni semantiche collegano i gruppi di sinonimi. Per operare sui database si utilizzano i DBMS (Data Base Management System): collezione di programmi che consentono agli utenti di creare e gestire più basi di dati. Lo scopo di un DBMS è facilitare il processo di definizione, costruzione e manipolazione di basi di dati relative a differenti applicazioni. Si ottiene tramite:

- Individuazione dei tipi di dati che devono essere memorizzati ed una dettagliata descrizione di ogni tipo di dato;
- Memorizzazione dei dati su un certo supporto;
- Interrogazione della base di dati per ritrovare alcune informazioni;
- Aggiornamento della base di dati per riflettere i cambiamenti osservati nella realtà che la base di dati descrive,
- Generazione di tabulati costruiti sui dati memorizzati.

Un DBMS deve avere le caratteristiche di:

- *Indipendenza fisica*: la riorganizzazione fisica dei dati non deve comportare effetti collaterali sui programmi applicativi / è possibile modificare la struttura fisica dei dati senza dover apportare cambiamenti alla loro descrizione al livello concettuale
- *Indipendenza logica:* modifiche allo schema logico/concettuale non comportano aggiornamenti degli schemi esterni e delle applicazioni

Gli utenti di un DB sono suddivisibili in diverse tipologie, a cui vanno pertanto associate autorizzazioni distinte, ad esempio:

- uno studente può leggere i propri dati, ma non quelli di altri studenti; inoltre non può modificare l'elenco degli esami sostenuti
- un docente può leggere i dati dei soli studenti del proprio corso; non può modificare l'elenco degli esami già sostenuti da uno studente, ma può registrare esami del proprio corso
- la segreteria può leggere i dati di tutti e può registrare nuovi studenti

La gestione delle autorizzazioni può essere oltremodo complessa, per questo motivo sono previste specifiche figure di Data Base Administrator (DBA) che conferiscono agli utenti i "giusti" privilegi. Un DBMS deve garantire che gli accessi ai dati, da parte di diverse applicazioni, non interferiscano tra loro. Al fine di conservare l'integrità dei dati è pertanto necessario far ricorso a opportuni meccanismi di controllo della concorrenza. Quindi un DBMS si occupa dei seguenti aspetti:

1. Gestione di grandi moli di dati (pazienti all'interno di una divisione ospedaliera, dati clinici ed amministrativi, storico, immagini)
2. Gestione della persistenza (il tempo di vita dei dati va oltre quello previsto dall'applicazione)
3. Condivisione dei dati (molti e differenti utenti con finalità e modalità di accesso, controllo di concorrenza per la modifica contemporanea dei dati per evitare inconsistenze)
4. Controllo della ridondanza (dati non duplicati)
5. Operazioni di salvataggio e ripristino (affidabilità)
6. Privatezza dei dati (gestione di accessi autorizzati)

Un DBMS è una rappresentazione concettuale della base di dati, che può essere vista come una tabella caratterizzata da:

- *Schema (componente intensionale)*: è la descrizione della struttura con la quale sono memorizzati i dati.

- *Istanza (componente estensionale)*: è il contenuto della base di dati in uno specifico momento.

Negli ultimi anni la creazione di dati è incrementata incredibilmente, a partire dal 2012, ogni giorno vengono creati circa 2,5 exabyte di dati e tale numero raddoppia ogni 40 mesi circa. Più dati attraversano Internet ogni secondo di quanto non siano stati memorizzati su Internet solo 20 anni fa. Ciò offre alle aziende l'opportunità di lavorare con molti petabytes di dati in un unico set di dati, e non solo da Internet. Ad esempio, si stima che Walmart raccolga più di 2,5 petabyte di dati ogni ora dalle sue transazioni con i clienti [48]. Un petabyte è pari a un quadrilione di byte, ovvero l'equivalente di circa 20 milioni di documenti di archiviazione. Un exabyte è 1.000 volte tale importo, o un miliardo di gigabyte. Queste grandissime quantità di dati, come abbiamo precedente accennato, prendono il nome di Big Data. I Big Data assumono la forma di messaggi, aggiornamenti e immagini postati sui social network; lettura da sensori; segnali GPS da telefoni cellulari e altro. Molte delle fonti più importanti di Big Data sono relativamente nuove. Con lo sviluppo della società si vedono sempre più necessari quindi potenti strumenti di analisi di dati, senza di questi l'analisi di data set di dimensioni di exabyte potrebbero richiedere mesi, non rendere le informazioni ricercate o addirittura fornirne di sbagliate. Le enormi quantità di informazioni provenienti dai social network, ad esempio, sono vecchie quanto le reti stesse; Facebook è stato lanciato nel 2004, Twitter nel 2006. Lo stesso vale per gli smartphone e gli altri dispositivi mobili che ora forniscono enormi flussi di dati legati a persone, attività e luoghi. Come spiegano W. Edwards Deming e Peter Drucker "Non si può gestire ciò che non si può misurare" [49].

Database relazionali (SQL)

Si propone di superare il problema della mancanza di indipendenza tra l'analisi teorica e il sistema di gestione di basi di dati adottato. In questo formato si rappresenta la base di dati come una collezione di tabelle legate ("Join") tra loro in maniera univoca; ogni tabella corrisponde al concetto di relazione. Le caratteristiche che questa tipologia di database presenta sono:

- semplicità
- esistenza di numerosi strumenti per la gestione della base di dati
- ricca e completa trattazione teorica
- esistenza di potenti linguaggi di interrogazione accettati e diffusi
- i concetti fondamentali del modello relazionale dei dati sono quelli di relazione, tupla (entità) e attributo

La novità introdotta da questo tipo di database è il collegamento che viene fatto tra tabelle diverse, che quindi possono dare differenti informazioni, tramite codici che identificano in maniera univoca una certa entità, i quali prendono il nome di chiave (Key Value). Una relazione è un insieme di tuple composte a loro volta da valori di attributi, con le seguenti proprietà:

- non esiste alcun criterio di ordinamento tra tuple di una relazione
- non possono esistere tuple identiche in una relazione
- non esiste ordinamento fra gli attributi di una relazione

In una relazione vi sono dei sottoinsiemi di attributi che conservano questa proprietà.

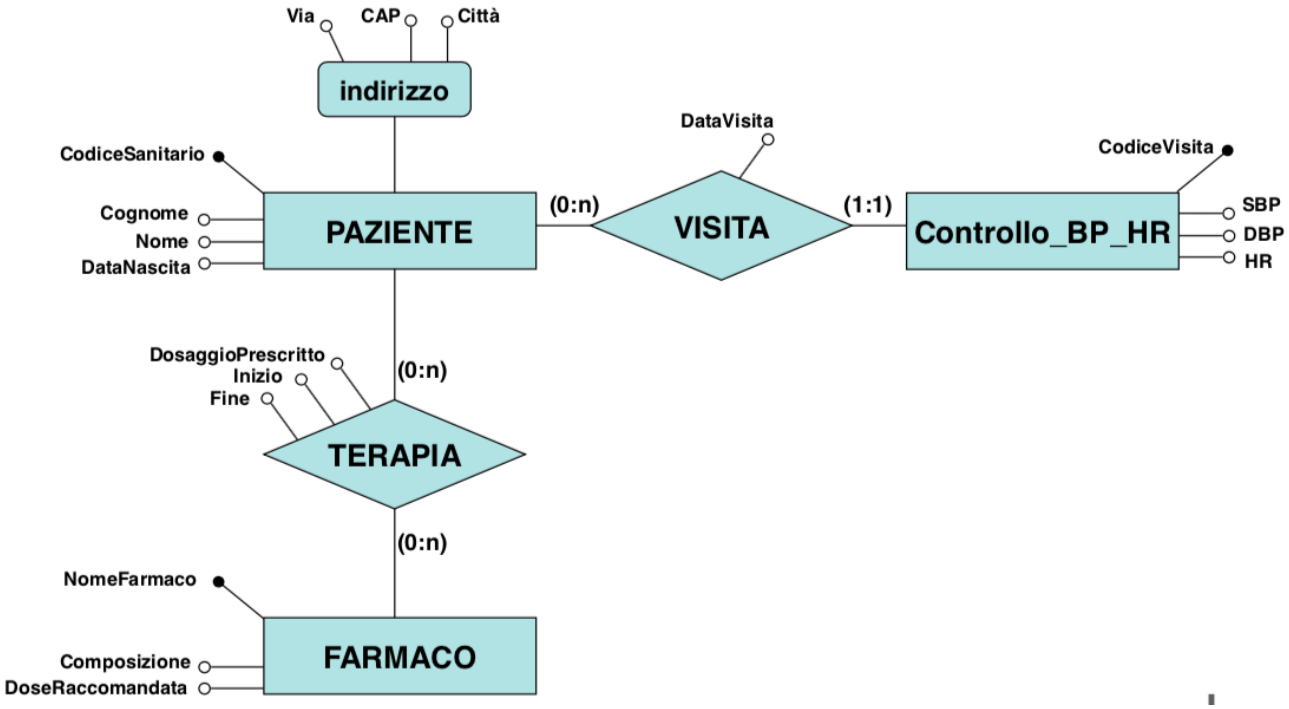

Figura 3 La figura mostra un esempio di schema entità-relazione. Farmaco, Paziente, Indirizzo e Controllo sono le entità, mentre Terapia e Visita sono le relazioni. Le voci illustrate al di fuori del diagramma rappresentano gli attributi, in particolare quelli contrassegnati dal cerchio pieno sono i Key-Values. I valori riportati tra parentesi indicano i vincoli di cardinalità. Dallo schema emerge come uno specifico farmaco (NomeFarmaco) venga assegnato ad un paziente (CodiceSanitario) in base ad una precisa terapia. Sfruttando le informazioni riportate in figura è possibile costruire la tabella relazionale corrispondente. [Lo schema in figura è preso dalle slide del corso di Basi di Dati di Ingegneria Biomedica UNIBO]

Per la creazione di database relazionali si fa riferimento ad una progettazione concettuale, il modello entità-relazione [50], che è uno strumento descrittivo e non implementativo della base di dati. Quindi l'entità è l'insieme (classe) di oggetti della realtà di interesse che possiedono caratteristiche comuni e che hanno un'esistenza autonoma e l'istanza di una entità è uno specifico oggetto appartenente a quella entità.

La relazione invece rappresenta il legame tra più entità: ogni legame è una istanza (relationship istance) di un certo tipo di relazione (relationship type), caratterizzata e vincolata dalla cardinalità (cardinality), la coppia di numeri (N, M) che specificano il numero minimo e massimo di entità che possono partecipare ad una istanza di quella relazione.

Quindi:

• se il concetto è significativo per il contesto applicativo: ENTITA'

• se il concetto è marginale e descrivibile in modo semplice: ATTRIBUTO

• se il concetto definisce un legame tra entitá: RELAZIONE (o ASSOCIAZIONE)

Per indicare un Database Menagement System basato sul modello relazionale si usa il termine *relational database management system* (RDBMS). Questa struttura obbliga la frammentazione delle informazioni fra differenti tabelle, anche quando i dati descrivono un medesimo oggetto**.**

Per identificare le informazioni appartenenti a uno stesso oggetto si utilizzano tutta una serie di operazioni logiche come il "join", che basano i propri calcoli sulle chiavi esterne o foreign keys che mettono in collegamento una tabella all'altra. Per questo motivo, utilizzare tabelle SQL troppo grandi è fortemente sconsigliato: sono di difficile gestione anche con sistemi computazionali molto potenti. Questa forte frammentazione prescrive anche un rigido controllo sulle relazioni e sulla validità dei dati residenti nelle differenti tabelle in modo da preservare l'integrità del database, a scapito della flessibilità. È anche per questo motivo che i database relazionali si basano su schemi tabellari (schemi entità-relazione) predefiniti a monte e che difficilmente possono essere riadattati a nuove situazioni, a meno di non correre il rischio di una corruzione dei dati.

```
1. CREATE TABLE Person (
2. ID INT,
3. name CHAR(10),
4. addr INT,
5. FOREIGN KEY(addr) REFERENCES Address(ID)
6. )
```

Figura 4 La figura riporta un esempio di codice SQL necessario alla creazione di una tabella "Person". La tabella sarà composta da tre campi; "ID" e "addr" conterranno valori interi, mentre "name" conterrà una stringa di massimo 10 caratteri. Il comando "FOREIGN KEY" caratterizza le tabelle relazionali. In questo caso collega la tabella "Person" e la tabella "Address", riportata in seguito (Tabella 2).

Person

ID	name	addr
7	Bob	18
8	Alice	*NULL*

Tabella 1 La struttura della tabella illustrata è il risultato della query rappresentata in Figura 4. Si nota come non sia conosciuto l'indirizzo di Alice. La possibilità di riportare valori nulli è tipica dei database relazionali.

```
1. CREATE TABLE Address(
2. ID INT,
3. city CHAR(10),
4. state CHAR(2)
5. )
```

Figura 5 La figura riporta un esempio di codice SQL necessario alla creazione di una tabella "Address". La tabella sarà composta da tre campi; "city" e "state" conterranno stringhe di lunghezza massima rispettivamente di 10 e 2 caratteri, mentre "ID" conterrà valori interi.

Address

ID	city	state
18	Cambridge	MA

Tabella 2 La struttura della tabella illustrata è il risultato della query rappresentata in Figura 5 . Il valore contenuto nel campo "ID" è il corrispondente dell'indirizzo di Bob; infatti, "Address" rappresenta la chiave esterna che collega questa tabella alla Tabella 1.

Il codice SQL definisce la struttura delle due tabelle, specificando la tipologia di valori ammissibili all'interno dei campi e le relazioni (foreign key) tra le due. Questo consente al database relazionale di acquisire le relazioni che legano le entità del mondo reale (le cose) rappresentate nel database. I valori riportati non sono legati con il linguaggio SQL, che difatti non è un linguaggio basata sull'oggetto, e indicano che Bob vive a Cambridge, in Massachusetts, ma che non sappiamo dove vive Alice. I numeri 7, 8 e 18 sono stati ideati appositamente per identificare le righe e catturare i collegamenti tra di loro. Le query SQL su queste relazioni generalmente non menzionano i numeri specifici ma ricapitoleranno i vincoli della relazione, ovvero Person.addr = Address.ID.

Database non relazionali (NoSQL)

Un NoSQL (originariamente riferito a database "non SQL" o "non relazionale") [51] fornisce un meccanismo per la memorizzazione e il recupero di dati modellati in mezzi diversi dalle relazioni tabulari utilizzate nei database relazionali. I database SQL utilizzano le query SQL per definire e manipolare i dati organizzati in tabelle in maniera ordinata; se da un lato questo metodo è estremamente efficiente e trova moltissime applicazioni, dall'altro risulta restrittivo. Infatti, si

incontra una importante limitazione durante l'analisi di dati che non presentano una struttura rigida, uno schema definito (come quello entità-relazione). I database NoSQL offrono una soluzione a questo problema in quanto utilizzano schemi dinamici per dati non strutturati. I database NoSQL risultano particolarmente efficaci quando si devono trattare flussi di dati originati da molteplici sorgenti eterogenee, primo fra tutti il caso del Web. In queste condizioni i dati risultano irregolari, sparsi e privi di ordine, pertanto richiedono flessibilità nei metodi di archiviazione e recupero impiegati. I database non relazionali sono esistiti fin dalla fine degli anni '60, ma non hanno ottenuto il soprannome di "NoSQL" fino a quando non si è verificato un aumento di popolarità all'inizio del XXI secolo, [52] innescato dalle esigenze delle aziende del Web [53]. I database NoSQL sono sempre più utilizzati nei Big Data e nelle applicazioni Web in tempo reale [54]. Le motivazioni per questo approccio includono: semplicità di progettazione, scalabilità orizzontale più semplice a gruppi di macchine (che è un problema per i database relazionali), controllo più preciso sulla disponibilità e limitazione del disadattamento di impedenza relazionale dell'oggetto [55]. Con "scalabilità orizzontale" ci si riferisce alla capacità del cloud di soddisfare le necessità degli utenti in maniera efficiente, incrementando le capacità di gestione di traffico all'aumentare della richiesta tramite l'aggiunta di nodi al sistema. Questo aumento di unità operative permette di suddividere il carico di lavoro sui nodi, che di conseguenza lavorano in parallelo e lo rendono più rapido. Oltre a velocizzare i tempi di esecuzione, questo approccio aumenta l'affidabilità dell'intero database, poiché l'eventuale malfunzionamento di un nodo non pregiudica gli esiti finali. Le strutture di dati utilizzate dai database NoSQL (ad esempio valore-chiave, colonna estesa, grafico o documento) sono diverse da quelle utilizzate di default nei database relazionali, rendendo alcune operazioni più veloci. In particolare, questi sono caratterizzati da una struttura orientata all'oggetto, quindi risultano molto più utili per scopi e situazioni specifiche. Inoltre, i database NoSQL non richiedono le dispendiose operazioni di combinazione delle tuple (Join) necessarie invece nei database SQL. Allo scopo di ottenere un miglioramento prestazionale nella gestione e nell'interrogazione dei dati, i database non relazionali scelgono di impiegare un sistema distribuito. Tuttavia, alcuni sistemi NoSQL presentano forme di perdita di dati [56], motivo per cui a volte sfruttano tecniche quali la registrazione write-ahead per evitare questo inconveniente. La perdita di dati conseguente al partizionamento del sistema trova spiegazione nel cosiddetto Teorema CAP, pubblicato nel 2000 da Eric Brewer [57]. Infatti, Brewer individua tre proprietà fondamentali dei database distribuiti, postulando che sia impossibile per un sistema di questo tipo presentare contemporaneamente più di due tra esse [58]:

-Coerenza: ogni lettura riceve la scrittura più recente o un errore

-Disponibilità: ogni richiesta riceve una risposta (non di errore) senza la garanzia che contenga la scrittura più recente

-Tolleranza delle partizioni: il sistema continua a funzionare nonostante un numero arbitrario di messaggi interrotti (o ritardati) dalla rete tra i nodi.
In particolare, il teorema CAP implica che in presenza di una partizione di rete, si debba scegliere tra coerenza e disponibilità.
Il mondo digitale sta crescendo molto velocemente e diventa più complesso nel volume (da terabyte a petabyte), varietà (strutturata e non strutturata e ibrida), velocità (alta velocità in crescita) e in natura, portando alla creazione di grandi quantità di dati come appunto i Big Data. Per gestire questo problema, gli RDBMS tradizionali sono completati da una serie di DBMS alternativi appositamente progettati, come NoSQL e NewSQL, allo scopo di fornire classificazione, caratteristiche e valutazione dei database NoSQL nell'analisi dei Big Data [59].

Name	Language(s)	Notes
AllegroGraph	SPARQL	RDF triple store
Amazon Neptune	Gremlin, SPARQL	Graph database
ArangoDB	AQL, JavaScript, GraphQL	Multi-model DBMS Document, Graph database and Key-value store
DEX/Sparksee	C++, Java, .NET, Python	Graph database
FlockDB	Scala	Graph database
IBM DB2	SPARQL	RDF triple store added in DB2 10
InfiniteGraph	Java	Graph database
MarkLogic	Java, JavaScript, SPARQL, XQuery	Multi-model document database and RDF triple store
Neo4j	Cypher	Graph database
OpenLink Virtuoso	C++, C#, Java, SPARQL	Middleware and database engine hybrid
Oracle	SPARQL 1.1	RDF triple store added in 11g
OrientDB	Java, SQL	Multi-model document and graph database
OWLIM	Java, SPARQL 1.1	RDF triple store
Profium Sense	Java, SPARQL	RDF triple store
Sqrrl Enterprise	Java	Graph database

Figura 6 La tabella riportata elenca alcuni tra i principali database NoSQL, nonché i linguaggi di programmazione da essi utilizzati. [presa dal file di Wikipedia.eng con url https://en.wikipedia.org/wiki/NoSQL]

Esistono svariati modi per classificare i database NoSQL, differenziati in base alle strutture di dati con cui si rappresentano i record di dato; tra questi si ricordano quattro approcci principali:

- ***Database Column-oriented (database orientato alle colonne):*** I database colonnari utilizzano tabelle, righe e colonne, ma a differenza di un database relazionale, i nomi e il formato delle colonne possono variare da riga a riga nella stessa tabella. Un ampio archivio di colonne può essere interpretato come un archivio di Key-Values bidimensionale. Questo approccio è tipicamente adottato nell'ambito della memorizzazione distribuita dei dati. HBase

e Cassandra utilizzano questa tipologia di classificazione [60]. L'organizzazione colonnare è utilizzata anche da BigQuery per velocizzare l'estrazione delle informazioni e ridurre i costi esecutivi in termini di CPU. Questo argomento sarà approfondito ulteriormente in seguito.

- ***Key-Value store (database di valori-chiave)*** [61]***:*** Gli archivi di valori-chiave utilizzano l'array associativo (noto anche come mappa o dizionario) come modello di dati fondamentale. In questo modello, i dati sono rappresentati come una raccolta di coppie chiave-valore, in modo tale che ogni possibile chiave appaia al massimo una volta nella raccolta. Esempi di database che impiegano questa tecnica sono BerkeleyDB, Project Voldemort [62]. Questo approccio è il più semplice da implementare ma presenta delle limitazioni quando si rende necessario modificare parzialmente uno o più elementi archiviati; infatti, il sistema permette di operare con gli elementi immagazzinati solo per intero.
- ***Document-oriented database (database orientato al documento):*** Il concetto centrale di questo approccio è costituito dalla nozione di "documento". Sebbene ogni implementazione dei database di questo tipo differisca nei particolari da questa definizione, in generale, tutti presuppongono che i documenti incapsulino e codifichino dati (o informazioni) in alcuni formati standard. Le codifiche in uso includono principalmente XML, YAML e JSON, ma anche moduli binari come BSON [63]. I documenti sono catalogati nel database tramite una chiave univoca che rappresenta uno specifico documento. Il documento può contenere un qualunque numero di campi di lunghezza illimitata. Un altro vantaggio di questa tipologia di database è che, oltre alla ricerca della chiave effettuata da un database di key-values, offre anche un linguaggio API o di query che recupera i documenti in base al loro contenuto. Esempi noti di questa tipologia di database sono il MongoDB [64] e il CouchDB.
- ***Graph database (database a grafo):*** Questo tipo di database è progettato per i dati le cui relazioni sono ben rappresentate tramite un grafo costituito da elementi interconnessi da un numero finito di relazioni. Dati di questo tipo si ritrovano per esempio nelle relazioni sociali, nei collegamenti di trasporto pubblico, nelle mappe stradali e nelle topologie di rete. Neo4J e Titan sono esempi di graph database [65].

In informatica, il graph database [66] è un archivio che utilizza strutture a grafo basate sui concetti di nodo, arco e proprietà per rappresentare e archiviare dati. Le relazioni consentono ai dati immagazzinati di essere collegati direttamente e, in molti casi, recuperati con una sola operazione. I database a grafo mantengono le relazioni tra i dati come priorità. Interrogare le relazioni all'interno di questi è veloce, poichè sono permanentemente memorizzate all'interno del database stesso. Le

relazioni possono essere visualizzate in modo intuitivo utilizzando un database a grafo, rendendoli utili per dati fortemente interconnessi [67].

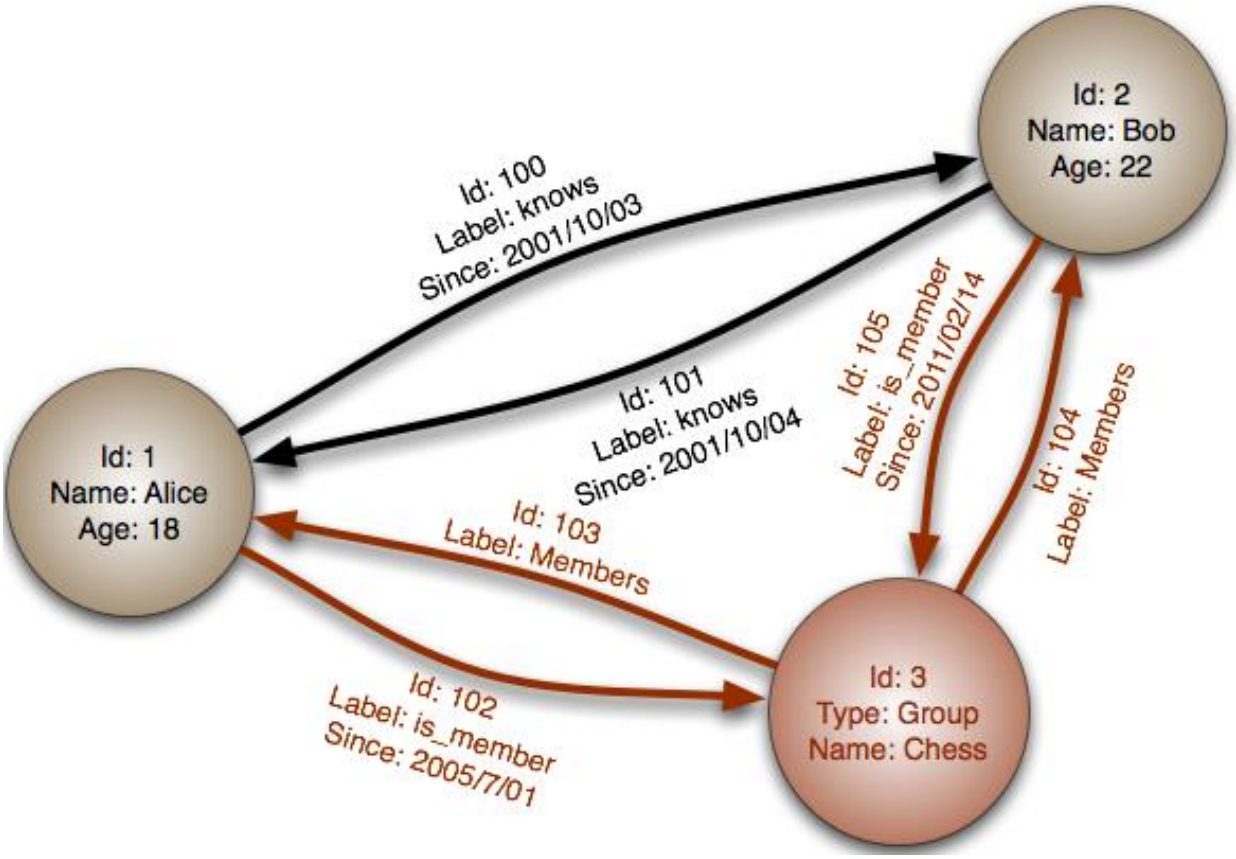

Figura 7 Il grafo in figura mostra come i database di questa tipologia mettano in evidenza le relazioni tra gli elementi interconnessi. Nell'esempio rappresentato, è nota (e indicata) la data in cui si sono conosciuti Bob e Alice, nonché l'appartenenza di entrambi al gruppo di giocatori di scacchi. [immagine presa Wikipedia https://it.wikipedia.org/wiki/Base_di_dati_a_grafo]

La possibilità di elaborare grandi quantità di dati rapidamente, la scalabilità orizzontale garantita dall'aggiunta di nuovi server, la capacità di analisi di dati non strutturati e i costi relativamente contenuti rispetto ai tradizionali database relazionali hanno rinnovato l'interesse per i database NoSQL (e in particolare per quelli a grafo) anche da parte di organizzazioni importanti quali Facebook e Twitter. Infatti, entrambe gestiscono enormi quantità di informazioni eterogenee provenienti dal Web e interrelate tra loro (in quanto mappe virtuali simili relazioni umane). Lo stesso Web Semantico [68] si può vedere come un enorme database a grafo, grazie alla rappresentazione della grande quantità di dati tramite il formato RDF qui di seguito descritto. L'aumento del numero di informazioni presenti sul Web fa sì che gli utenti, quando vogliono cercare informazioni specifiche, si trovano a dover compiere una grossa selezione, poiché il motore di ricerca restituisce, seppure in un tempo brevissimo, tanti risultati contenenti le parole chiavi cercate, non facendo fede però al significato dell'informazione cercata. Per ovviare a ciò, il W3C (World Wide Web Consortium) già nel 1999 propone l'utilizzo del Resource Description Framework (RDF), un modello astratto di descrizione per la codifica, lo scambio e il riutilizzo di metadati strutturati. Questi ultimi sono anch'essi dei dati, che svolgono la funzione di descrivere e classificare il significato di altri dati, al

fine di rendere la loro accessibilità più semplice e immediata, garantendo l'interoperabilità semantica tra le diverse applicazioni che condividono la medesima informazione sul Web.

La rappresentazione semantica dei dati: il modello RDF

Il modello dei dati proposti dal Resource Description Framework (RDF) [69] si basa su 2 tipi di oggetti: **Risorse (resources):** Una risorsa può essere rappresentata da un qualsiasi elemento che abbia un Universal Resource Identifier (URI), cioè un meccanismo di identificazione.

Asserzioni (statements): Rappresenta l'unità base per rappresentare un'informazione in RDF. Un'asserzione è costituita dall'insieme di una risorsa, un predicato, e uno specifico valore per quel predicato, rispettivamente chiamate Soggetto-Predicato-Oggetto. Uno statement RDF descrive sia le caratteristiche di una risorsa, che le relazioni con altre risorse.

L'RDF è un modello di descrizione astratto che non pone vincoli sulla sintassi, ci sono quindi vari modi per rappresentarlo. Uno di questi è il linguaggio Turtle [70](Terse RDF Triple Language) successivamente esposto, ma ulteriori tipi di serializzazioni come per esempio RDF/XML [71] possono essere impiegate. Recentemente un linguaggio di serializzazione dei dati RDF che sta prendendo molto piede è JSON-LD [72].

Di seguito vengono riportati alcuni esempi sulla rappresentazione RDF.

```
1. <PersonA> a <Person> .
2. <AddressB> a <Address> .
3. <PersonA> <Person#name> “Bob” .
4. <AddressB> <Address#city> “Cambridge” .
5. <PersonA> <Person#addr> <AddressB> .
6. <AddressB> <Address#state> “MA” .
```

Figura 8 La figura riporta un esempio rappresentazione Turtle relativo ad un grafo basato sul soggetto "Bob". Si mette in risalto la differenza col linguaggio SQL; infatti, Turtle, come del resto le strutture grafo, è finalizzato alla descrizione di un soggetto e delle sue caratteristiche. In particolare, si nota come l'oggetto "Bob" sia collegato all'indirizzo di residenza di "Cambridge, MA".

```
1. <PersonF> a <Person> .
2. <PersonF> <Person#name> “Alice” .
```

Figura 9 La figura riporta un esempio di rappresentazione Turtle relativo ad un grafo basato sul soggetto "Alice". Volendo rappresentare le stesse informazioni fornite dalle tabelle relazionali precedentemente illustrate (Tabella 1 e Tabella 2), si nota come l'oggetto "Alice" non sia collegato a nessun indirizzo. A differenza del linguaggio SQL, Turtle non determina la struttura di una tabella, bensì le informazioni riguardanti un certo soggetto.

Ricollegandosi all'esempio precedente di un database relazionale, nel caso RDF acquisisco sia gli attributi che le relazioni tra entità. Negli esempi rimportati è evidente come il modello RDF sia basato sul soggetto (Bob o Alice), a differenza del modello SQL che è basato sulla struttura della tabella e le relazioni che la legano ad altre. Il linguaggio utilizzato è Turtle, poichè ha la caratteristica di avere una sintassi adatta a SPARQL (che sarà utilizzato successivamente per interrogare il grafo). Chiamiamo i tre termini in ogni dichiarazione RDF come soggetto, predicato e oggetto. I termini

utilizzati nelle dichiarazioni di cui sopra, sono URL relativi tra parentesi angolari (<>), letterali tra virgolette (""), mentre la parola chiave "a" è solo un'abbreviazione per un URI predefinito dal modello RDF (http://www.w3.org/1999/02/22-rdf-syntax-ns#type), che sta ad indicare la classe di appartenenza di una certa risorsa secondo quello che è RDFS [73]. Negli esempi mostrati in precedente, questo permette di asserire che gli URI <PersonA> e <PersonF> sono istanze delle classe <Person>, mentre per esempio l'URI <AddressB> è un'istanza della classe <Address>. Non esiste alcun concetto in RDF corrispondente a NULL di SQL in quanto non esiste alcun requisito RDF corrispondente al vincolo strutturale di SQL che ogni riga in un database relazionale deve essere conforme allo stesso schema. L'oggetto di un'asserzione, ad esempio <AddressB> sopra, può essere il soggetto, oppure può essere l'oggetto di altre asserzioni. In questo modo, un insieme di statement RDF si connette per creare un "grafo" (in senso matematico), in particolare un grafo orientato e labellato in cui i nomi degli archi corrispondono agli URI dei predicati, mentre i nodi possono sia corrispondere a URI (per esempio <PersonC>) o a valori letterali (per esempio "Bob"). Si trova comunemente la situazione in cui questi grafi possano essere ciclici, ad esempio affermando che Bob vive in un posto in cui è anche il proprietario.

```
1. <PersonC> <Person#homeAddress> <AddressK> .
2. <PersonC> <Person#name> "Bob" .
3. <AddressK> <Address#owner> <PersonC> .
```

Figura 10 La figura riporta un esempio di codice Turtle necessario alla creazione di un grafo. In questo caso si rende ciclico il grafo illustrato precedentemente (Figura 8), collegando l'elemento residenza all'elemento proprietario.

Confronto tra due linguaggi di query: SQL e SPARQL

L'operazione di estrapolazione di informazioni e di interrogazione di un database prende il nome di query. A seconda della tipologia del DB le query potranno essere eseguite tramite diversi linguaggi informatici. I due più conosciuti e utilizzati sono SQL (Structured Query Language) e SPARQL [74]. Entrambi questi linguaggi consentono all'utente di creare, combinare e utilizzare dati strutturati. SQL lo fa accedendo alle tabelle in database relazionali e SPARQL lo fa accedendo a una rete di dati collegati (RDF).

Le caratteristiche principali di SQL sono:

- Linguaggio indipendente dal particolare sistema adottato che permette all'utente di interagire in modo omogeneo con differenti sistemi basati sul modello relazionale.
- Standard de facto con funzionalità tipiche per la definizione della base dei dati e per la manipolazione della base di dati.
- Linguaggio dichiarativo: permette di specificare le caratteristiche del risultato che si vuole ottenere e non la sequenza delle operazioni per ottenerlo.

```
SELECT Person.name, Address.city
FROM Person, Address
WHERE Person.addr=Address.ID
AND Address.state=”MA”
```

Figura 11 La figura riporta un esempio di query SQL che fa riferimento all’esempio illustrato in Tabella 1 e Tabella 2. Nello specifico si interrogano le tabelle sopra citate (clausola FROM) per ottenere l’indirizzo e il nome (clausola SELECT) di ogni persona residente in Massachussets (MA). Si nota come la clausola WHERE renda necessaria la corrispondenza della chiave esterna che mette in relazione le due tabelle interpellate nella query.

Lo scopo della query è quello di ottenere gli indirizzi di ogni persona che vive in MA. Concettualmente, stiamo selezionando (SELECT) un elenco di attributi da (FROM) un insieme di tabelle dove (WHERE) sono soddisfatti alcuni vincoli. Questi vincoli catturano le relazioni implicite nello schema, Person.addr = Addresses.ID e i criteri di selezione, ad esempio Address.state =”MA”.

```
SELECT ?name ?city
WHERE {
?who <Person#name> ?name ;
<Person#addr> ?adr .
?adr <Address#city> ?city ;
<Address#state> “MA”
}
```

Figura 12 La figura riporta un esempio di query SPARQL riferita ai grafi in Figura 8 e Figura 9. Con questa interrogazione si vuole ottenere il medesimo risultato di quella effettuata in liguaggio SQL (Figura 11).

In un’analoga query SPARQL le variabili sono associate al modello del grafo (dove con grafo intendo un insieme di affermazioni potenzialmente interconnesse). Il modello a grafo assomiglia alle dichiarazioni dei dati, ma il soggetto, il predicato o l'oggetto possono essere sostituiti con una variabile (termini che iniziano con un "?"). Il modello sopra trova tutti i valori di ?Name, ?City, ?Who e ?Addr che corrispondono ai dati, proiettando solo ?Name e ?City. Compie quindi una ricerca completa all’interno del database di tutte le possibili associazioni richieste, quindi il risultato saranno i nomi e la città di tutte le persone che vivono nel Massachusetts. Si nota come questo linguaggio sia simile al precedente sotto certi aspetti, in particolare SPARQL utilizza parole chiave presenti anche nel linguaggio SQL, come “Select”, “From”, “Where” visti precedentemente, ma anche altri come “Group by”, “Union”, “Having”, utilizzati per query più articolate che mirano ad ottenere informazioni più dettagliate [75]. I servizi SPARQL variano in base al fatto che abbiano o meno un database RDF predeterminato. La query di esempio presuppone che sia disponibile un set di triple per l'interrogazione, che ci si aspetterebbe dal servizio che interroga un database già popolato. Se il database RDF era vuoto per impostazione predefinita, o non conteneva i dati necessari per la query, avremmo dovuto specificare dove caricare tali dati. SPARQL riutilizza la parola chiave "FROM", come SQL, per identificare le risorse Web caricate per completare una query, operazione che non avrei potuto fare con l’altro tipo di linguaggio, poiché il Web è popolato da grandi quantità di dati

solitamente non organizzate in modo relazionale. Per entrambe le interrogazioni il risultato sarà analogo.

name	**city**
Bob	Cambridge

Tabella 3 La tabella illustrata rappresenta il risultato delle query presentate in Figura 11 e Figura 12.

Una differenza che si ha però nel risultato finale dei due diversi modi di porre la medesima query è che SPARQL quando non trova l'associazione tra due entità non dà come risultato "NULL", come invece accade nel caso SQL.

Cloud computing

Entriamo nel vivo della tesi, avvicinandoci a BigQuery ed andando a vedere gli aspetti essenziali del servizio di Cloud Computing, tra cui i possibili modelli di servizio e i modelli di implementazione. Saranno genericamente esposti i vari utilizzi possibili per cui le piattaforme online sono utilizzate, dall'archiviazione di dati alla possibilità di programmazione. Si affronta in particolare il vantaggio che il suo utilizzo porta nell'ambito sanitario, e le relative norme sulla privacy a cui devono sottostare le aziende che offrono il servizio in questo settore. Dato che il Cloud Computing è proposto da vari provider, si analizzano e comparano i principali tra questi, approfondendo Google ed andando a vedere i vari servizi di Cloud Computing che l'azienda propone. Alcuni tra i diversi servizi proposti saranno successivamente ripresi in maniera più approfondita, poichè vedremo come interagendo con BigQuery permettono complessi processi operativi complementari tra di loro.

Definizione

Il cloud computing è la disponibilità su richiesta delle risorse del sistema informatico, in particolare l'archiviazione dei dati e la potenza di calcolo, senza una gestione attiva diretta da parte dell'utente. Il termine viene generalmente utilizzato per descrivere i data center disponibili per molti utenti su Internet. I grandi Cloud, oggi predominanti, hanno spesso funzioni distribuite su più sedi dai server centrali. Se la connessione all'utente è relativamente vicina, può essere designata come edge server [76]. La definizione del NIST [77] (National Institute of Standards and Technology) spiega come il cloud computing sia un modello per abilitare l'accesso alla rete onnipresente, conveniente e on-demand a un pool condiviso di risorse di calcolo configurabili (ad esempio reti, server, storage, applicazioni e servizi) che possono essere rapidamente fornite e rilasciate con il minimo sforzo di gestione o interazione con il fornitore di servizi. Questo modello cloud è composto da cinque caratteristiche essenziali, tre modelli di servizio e quattro modelli di implementazione:

- ***Self-service su richiesta*:** Il sistema consente all'utente di usufruire liberamente delle funzionalità del Cloud (come l'archiviazione online) in funzione delle proprie esigenze e in modo automatico, ossia senza necessitare dell'assistenza di un operatore dal lato server.
- ***Ampio accesso alla rete*:** Tutte le funzioni rese disponibili possono essere utilizzate tramite internet attraverso processi standardizzati a prescindere dal dispositivo utilizzato; in altre parole viene promosso l'uso di piattaforme client eterogenee (ad es. Telefoni cellulari, tablet, laptop e workstation).
- ***Condivisione delle risorse*:** Le risorse fondamentali quali memoria, elaborazione e larghezza della banda di rete vengono suddivise e gestite in modo dinamico ed efficiente; tali risorse

vengono riassegnate continuamente in funzione della domanda riscontrata tra i consumatori del servizio. Questo modello, composto da risorse fisiche e risorse virtuali differenti, permette all'utente di specificare una posizione di utilizzo generica (per es. Paese, stato o banca dati) senza però avere alcun controllo o conoscenza sull'origine delle risorse fornite.

- ***Elasticità rapida*:** L'utente ha la sensazione di avere a disposizione capacità di elaborazione illimitate poiché il sistema è in grado di aumentare o diminuire rapidamente la quantità in cui queste ultime vengono erogate al variare della domanda. Questo concetto è definito scalabilità orizzontale.
- ***Servizio misurato*:** Per controllare e ottimizzare l'utilizzo delle risorse, i sistemi Cloud sfruttano processi di misurazione ad un livello di astrazione appropriato al tipo di servizio (ad es. archiviazione, elaborazione, larghezza di banda e numero di account utente attivi). Inoltre, in questo modo è possibile tener traccia con precisione delle risorse erogate, garantendo trasparenza sia da parte del provider che dell'utente.

Di seguito sono riportati i modelli di servizio correntemente utilizzati:

- **Software come servizio (SaaS):** L'utente ottiene l'accesso a una varietà di applicazioni messe a disposizione dal provider su un'infrastruttura Cloud; per interagire con queste ultime è sufficiente un client, un'interfaccia locale che può essere costituita da un browser o da un software. In questo modo il consumatore non ha nessuna capacità di controllo o gestione su rete, server, sistema operativo e archivio, o più in generale sull'infrastruttura del Cloud. Piuttosto, all'utente sono concesse alcune limitate funzionalità di configurazione e impostazione del servizio all'interno delle specifiche applicazioni utilizzate.
- **Piattaforma come servizio (PaaS):** All'utente viene garantita la possibilità di implementare nell'infrastruttura del Cloud applicazioni generate o acquisite attraverso l'uso librerie, linguaggi di programmazione, servizi e strumenti supportati dal provider. Ancora una volta l'utente non ottiene facoltà di gestire o modificare l'infrastruttura del Cloud, bensì funzionalità controllate, limitate alle applicazioni distribuite e alla configurazione per l'ambiente di hosting delle stesse.
- **Infrastruttura come servizio (IaaS):** L'utente ottiene accesso alla rete del Cloud, alla capacità di elaborazione e di archiviazione e ad altre risorse fondamentali che permettono la distribuzione e l'esecuzione arbitraria di software (inclusi sistemi operativi e applicazioni). Anche in questo caso i permessi di gestione dell'utente sono strettamente limitati alle funzionalità sopra descritte (e, in alcuni casi, ad alcune componenti di rete come il firewall). L'utente non ha quindi accesso all'infrastruttura del Cloud sottostante.

In conclusione, si ricordano i diversi modelli di distribuzione:

- ***Cloud privato.*** L'infrastruttura del Cloud privato è fornita ad uso esclusivo di un'organizzazione che comprende un gruppo di utenti, spesso un'azienda, e viene utilizzata dai membri della suddetta organizzazione. La proprietà e la gestione del Cloud privato possono essere interne all'azienda o esterne ad essa (a carico di terzi) o una combinazione di tali opzioni. Può essere "on-premises" o "SaaS".
- ***Community Cloud.*** In questo modello di distribuzione l'infrastruttura del Cloud è messa al servizio di una comunità di utenti che condividono alcune necessità, ad esempio riguardo alla sicurezza dei dati o alla policy. Il Community Cloud può essere di proprietà di una o più organizzazioni parte della comunità o di terze parti e lo stesso vale per la sua gestione. Come il Cloud privato può essere "on-premises" o "SaaS".
- ***Cloud pubblico.*** L'infrastruttura del Cloud pubblico è aperta all'uso di qualunque generico utente pubblico. La gestione e la proprietà dell'infrastruttura possono essere a carico di organizzazioni governative, accademiche o aziendali o di una combinazione di esse. Il Cloud pubblico è accessibile solamente in locale (on-premises) dal provider del Cloud.
- ***Cloud ibrido.*** Il Cloud ibrido affianca due tra le modalità di distribuzione sopra descritte (privato, community e pubblico); le due infrastrutture rimangono entità distinte ma allo stesso tempo accomunate da tecnologia proprietaria o standardizzata, la quale permette la portabilità dei dati e delle applicazioni. Ad esempio, il Cloud bursting è una tecnica utilizzata dal Cloud ibrido per bilanciare il carico di lavoro gestito dai due Cloud, facendo subentrare la seconda infrastruttura quando la prima è sovraccaricata.

Utilizzo e funzionamento

Alcune delle funzionalità più importanti di un computer sono costituite dalla possibilità di archiviare e recuperare dati su vari tipi di unità di memoria. Alle unità fisse quali hard disk interni e RAM, utilizzabili solamente in locale, vengono affiancati dispositivi mobili in grado di spostare dati tra più macchine, ad esempio CD, DVD e chiavette USB. Nel caso di computer interconnessi da LAN o WAN (rispettivamente reti locali e geografiche), queste funzionalità vengono estese a tutti i computer collegati alla rete, permettendo di effettuare operazioni su dispositivi in remoto. Il Cloud Computing permette di ampliare ulteriormente il campo di influenza di un operatore; infatti, ogni utente connesso al provider del Cloud potrà sfruttare le funzionalità di elaborazione, archiviazione e recupero dei dati completamente online attraverso un browser. L'accesso a uno spazio di archiviazione online permette

di svolgere molte attività; ad esempio, utilizzare in remoto un programma non installato in locale, consultare grandi quantità di dati, sfruttare servizi messi a disposizione dal provider e via dicendo. Il Cloud nasce principalmente come un servizio con lo scopo di "storage", o meglio come un servizio che dà la possibilità di avere uno spazio virtuale su cui poter salvare i propri file, che possono variare da foto, a testi, ad applicazioni o altri tipi di dati. Oggigiorno abbiamo oramai la possibilità di accedere ad Internet tramite svariati dispositivi, come tablet, smartphone, computer o workstation. Spesso può succedere però che lo spazio di archiviazione disponibile sul dispositivo utilizzato non sia sufficiente, quindi si potrebbe fare un back up su un hard disk esterno o su una chiavetta usb, che potrebbero però rompersi, come potrebbe accadere anche al dispositivo stesso. Oppure può succedere di aver salvato dei documenti su un dispositivo, ma di avere la necessità di poterlo avere su altri. Proprio per venire incontro a questi bisogni, nasce il Cloud storage. Questo offre infatti uno spazio di archiviazione personale, disponibile in qualsiasi momento e su qualsiasi dispositivo tramite l'accesso ad una linea Internet. I file salvati sul Cloud possono inoltre essere condivisi in modo semplice, veloce ed efficace, con altri utenti, su richiesta del cliente. La quantità di memoria messa a disposizione è potenzialmente infinita, o meglio è altamente scalabile, ad un prezzo che varia in base allo spazio richiesto dal cliente. Il Cloud computing offre inoltre un alto livello di sicurezza, che rende i file personali potenzialmente più al "sicuro" che su altri dispositivi fisici. Il principio di funzionamento è simile a quello della posta elettronica, anch'essa un servizio accessibile dal cliente dove e quando lo necessita. Noti esempi di questo tipo di servizio sono Google Drive [78], ICloud [79], Dropbox, WOW Space e Mega, solitamente offerti gratuitamente in prova per diverse quantità di spazio a seconda del provider, per far capire al cliente la comodità dell'archiviazione online. I Cloud offrono solitamente anche altri servizi oltre a quello di "storage", tra cui server, database, rete, software, analisi e intelligence, il tutto tramite Internet, per offrire innovazione rapida, risorse flessibili ed economie di scala. Paghi solo per i servizi cloud che usi e risparmi sui costi operativi, esegui l'infrastruttura in modo più efficiente e ridimensioni le risorse in base all'evoluzione delle esigenze aziendali. Oltre all'archiviazione e alla condivisione di file il Cloud permette di:

-*Creare nuovi servizi:* quindi di creare, distribuire e ridimensionare velocemente le applicazioni in qualsiasi piattaforma, che possono essere API, destinate al Web o ai dispositivi mobili. Permette di accedere alle risorse necessarie per soddisfare i requisiti di prestazioni, sicurezza e conformità.

-*Fornire software on demand:* conosciuto anche come Software as a Service (SaaS), è quindi un software su richiesta del cliente, che permette di offrire le versioni e gli aggiornamenti più recenti ai consumatori, sempre e ovunque si trovino.

-*Testare e compilare le applicazioni:* sfruttanto le infrastrutture Cloud è possibile ridurre costi e tempi di sviluppo delle applicazioni, poichè consentono di aumentare o ridurre facilmente e velocemente le prestazioni in base alle esigenze.
-*Analizzare i dati:* rende possibile l'unificazione di dati eterogenei provenienti da diverse fonti, come team, divisioni e sedi nel Cloud. Si potrà utilizzare quindi servizi come Machine Learning e intelligenza artificiale, col fine di acquisire informazioni dettagliate e prendere decisioni più pesate.
-*Incorporare l'intelligence:* utilizzare cioè modelli intelligenti per l'analisi di molteplici dati, per coinvolgere i clienti e raccogliere importanti e dettagliate informazioni dai dati acquisiti.
I vantaggi che questi servizi (ad esempio AWS o Azure) [80] offrono sono evidenti sotto vari punti di vista: la scalabilità orizzontale offerta garantisce da un lato grandi risorse di elaborazione (e quindi una velocità di esecuzione efficiente indipendentemente dalla mole dei dati interessati) e dall'altro la possibilità di mantenere costi contenuti; infatti, ridimensionare costantemente le risorse allocate in funzione delle necessità permette di evitare sprechi di capacità di elaborazione in modo molto più economico rispetto alla scalabilità verticale (anch'essa potenzia le capacità di elaborazione ma lo fa migliorando le risorse hardware, rimanendo quindi soggetta ai suoi limiti). La grande potenza di calcolo offerta dalla tecnologia del Cloud ha surclassato anche i database locali proprio per l'incapacità di superare i limiti imposti dall'hardware; il sistema di elaborazione in parallelo permette di assemblare una potenza virtualmente infinita (poiché non c'è limite al numero di server che possono essere impiegati in parallelo). Tuttavia, se da una parte i servizi Cloud risultano molto affidabili (anche considerando le funzionalità di protezione e ripristino dei dati spesso implementate) dall'altra è necessario ricordare che l'infrastruttura Cloud si basa su una rete estesa che comprende una grande varietà di dispositivi, il che espone a potenziali minacce informatiche provenienti da fonti diverse. L'enorme quantità di dati privati e sensibili rende necessario lo sviluppo di accurate misure di sicurezza, in seguito saranno approfondite quelle relative al caso specifico di Google Cloud.

Cloud più rinomati

Il cloud computing esiste dal 2000. Nell'agosto 2006, Amazon ha creato una filiale Amazon Web Services e ha introdotto il suo Elastic Compute Cloud (EC2) [81]. Nell'aprile 2008, Google ha rilasciato Google App Engine in versione beta. All'inizio del 2008, OpenNebula della NASA [82], arricchito dal progetto finanziato dalla Commissione europea RESERVOIR, è diventato il primo software open source per la distribuzione di cloud privati e ibridi e per la federazione di cloud. Nel mese di febbraio 2010, Microsoft ha rilasciato Microsoft Azure, annunciato nell'ottobre 2008. Nel luglio 2010, Rackspace Hosting e NASA hanno lanciato congiuntamente un'iniziativa di software

cloud open source nota come OpenStack [83]. Il progetto OpenStack intendeva aiutare le organizzazioni che offrono servizi di cloud computing in esecuzione su hardware standard.
A maggio 2012, Google Compute Engine è stato rilasciato in anteprima, prima di essere implementato in General Availability a dicembre 2013. Diversi studi mirano a confrontare queste offerte di open so urce sulla base di una serie di criteri. Quindi quale provider scegliere?

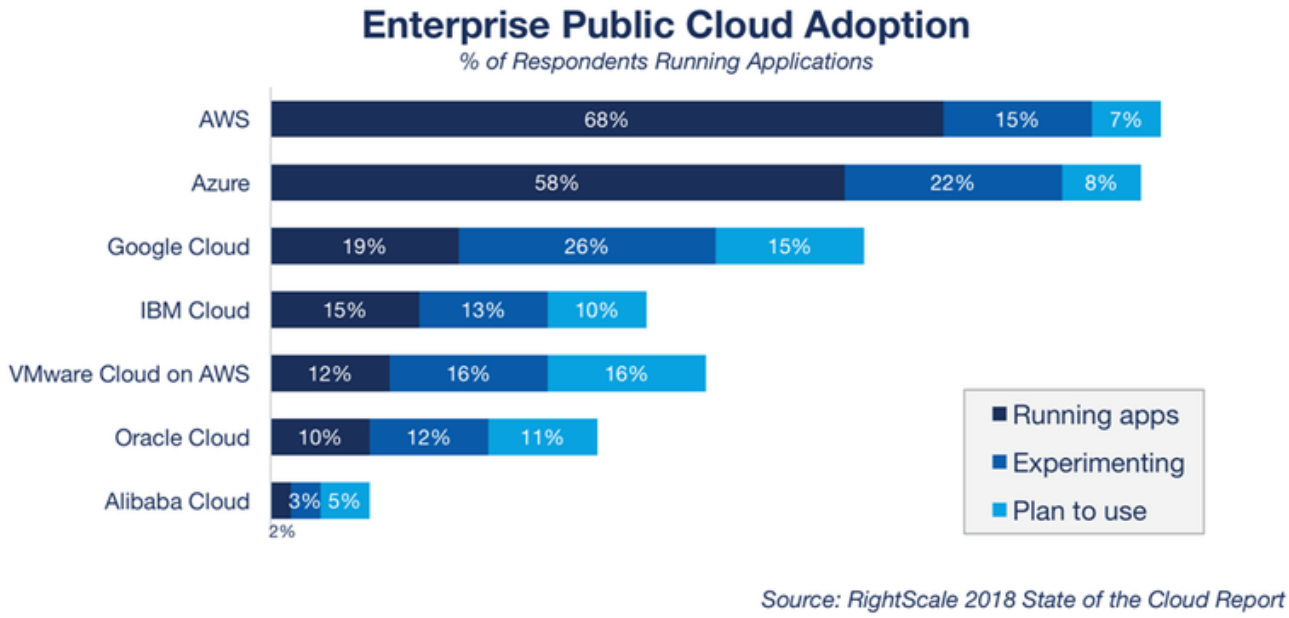

Figura 13 Rappresentazione grafica riguardo l'utilizzo dei vari Cloud nell' anno 2018. Le tonalità di colore utilizzate distinguono l'utilizzo "corrente", "in fase di prova" e "pianificato" da parte degli utenti. [istogramma preso dal Technical Report "RightScale 2018 State of the Cloud Report"]

Fondamentalmente la scelta dipende dal tipo di esigenze del cliente o dell'azienda. Oltre al servizio Cloud offerto da Google, di cui parleremo in seguito, gli altri provider più rinomati sono AWS [84] di Amazon e Azure [85] della Microsoft.

Cloud in medicina

"Immagina un futuro, in cui un dispositivo medico campiona semplicemente i tuoi organi vitali e carica le tue informazioni sanitarie, il dottore analizza le stesse basi sanitarie, scarichi una prescrizione di medicina di precisione dal medico, ordini il farmaco online e consegna a domicilio. Questo futuro dell'assistenza sanitaria è qui, dove stiamo entrando nell'era digitale con un notevole potenziale di medicina di precisione per salvare e migliorare la vita" [86]. L'utilizzo del Cloud Computing in ambito biomedicale si sta diffondendo enormemente in questi ultimi anni, tanto da rappresentare il futuro dell'assistenza sanitaria. I moderni fornitori di servizi sanitari stanno sfruttando il cloud e implementando il concetto di 'Cloud Medicine' su modelli basati su SaaS per

fornire assistenza sanitaria di qualità a tutti, ovunque e in qualsiasi momento. Nei paesi in via di sviluppo come l'India, il settore sanitario ha iniziato a utilizzare le più recenti tecnologie emergenti come il mobile computing e il cloud computing. Un grande volume di dati viene raccolto, archiviato, elaborato e recuperato in dati multimediali digitali del paziente denominati Electronic Health Records (EHR). Le cartelle cliniche elettroniche sono costituite da immagini di pazienti che presentano un elevato livello di sicurezza. La disponibilità della cartella clinica durante ogni visita all'ospedale migliora quindi la qualità del trattamento. Il cloud computing utilizzato nel settore sanitario riduce il costo dello storage, la disponibilità e la qualità in qualsiasi momento [87].
La combinazione di cloud computing e telemedicina introduce nuove opportunità per trasformare l'erogazione dell'assistenza sanitaria in modo più efficace e sostenibile. Numerose applicazioni di telemedicina sono state studiate e sviluppate sul cloud, come il telemonitoraggio e la teleconsultazione, tutte dimostrano pienamente il potenziale della telemedicina nella promozione di un'assistenza sanitaria più economica e di migliore qualità attraverso l'adozione di tecnologie cloud e mobili emergenti [88].

Utilizzi e vantaggi

Utilizzando questo tipo di formato, i dati sanitari, tra cui imaging diagnostici, referti, risultati di laboratorio e informazioni della storia clinica del paziente o anagrafiche, possano circolare molto meglio all'interno di un ospedale (ad esempio tra due medici di due reparti diversi) e all'interno delle aziende sanitarie (per analisi più globali, usate solitamente per prevenzione di epidemie). Quindi l'uso di cartelle cliniche elettroniche, oltre ad eliminare il problema della circolazione all'interno degli ospedali di cartacei, dei vari contenitori e di cartelle reali (aiutando a mantenere ovviamente un maggior livello di ordine e durabilità dei dati), facilita la possibilità della condivisione di queste informazioni, ovviamente solo con chi di dovere. Inoltre, rappresenta un risparmio per l'azienda ospedaliera, sia di tempo che di denaro, così da dare la possibilità al personale di passare più tempo con i pazienti. L'utilizzo del servizio di Cloud Computing all'interno degli ospedali sfrutta principalmente la funzione di "storage" e condivisione di dati. La funzione di analisi dei dati sarà invece sfruttata di più nelle aziende sanitarie per analisi sul territorio. La questione di analisi dei dati ricavati dalle aziende sanitarie, sarà toccata con mano successivamente nella parte sperimentale della tesi, dove i dati clinici saranno analizzati con BigQuery. Ovviamente per sfruttare a pieno le potenzialità del Cloud si necessita di strutture che hanno una buona connessione internet, requisito oramai facilmente realizzabile. Se inoltre l'azienda sanitaria è dotata di dispositivi "smart", il Cloud potrà fornire ulteriori vantaggi connettendosi direttamente con essi e dando quindi la possibilità di

analisi combinate in qualsiasi momento ad un prezzo economico e con un'ottima efficienza. Un servizio basato proprio su questi principi è offerto da Google, prende il nome di IoT Core e verrà ripreso successivamente.

Privacy e sicurezza dei dati sanitari

Nell'ambito della sicurezza dei dati sanitari e della tutela della privacy sono state svolte varie ricerche. Le informazioni sensibili sulla condizione di salute di una persona appartengono ai cosiddetti dati di categorie speciali e sono attentamente protette ai sensi dell'articolo 9 del GDPR (Regolamento generale sulla protezione dei dati in Unione Europea). Purtroppo, alcune ricerche nel settore mostrano come tali dati siano in realtà accessibili più o meno direttamente da vari terzi. I risultati di una ricerca realizzata da Cybot, società danese che fornisce soluzioni di consenso per il tracciamento dei cookie come Cookiebot, sensibilizzano circa la necessità di proteggere gli utenti che frequentano le pagine Web, in particolare quelle relative alla sanità. Cybot ha compiuto un'analisi nelle maggiori città europee, con la ricerca *Cookiebot Report 2019. Ad Tech Surveillance on the Public Sector Web* [89]. Lo studio è stato condotto ponendo domande sensibili riguardanti la salute (query) nei motori di ricerca di 6 paesi diversi, allo scopo di identificare le esatte pagine di destinazione che i cittadini dell'UE avrebbero realisticamente visitato per ottenere consigli governativi ufficiali sui servizi sanitari. Ciò ha prodotto 15 pagine Web per ciascun paese, ciascuno analizzato con la tecnologia di scansione di Cookiebot. Gli esiti riportati individuano il 52% delle pagine di destinazione scansionate come ospitante di cookie. Il servizio sanitario irlandese è risultato il peggiore, con il 73% delle pagine di destinazione contenenti tracker. Le pagine Web analizzate in Regno Unito, Spagna, Francia e Italia sono risultate monitorate rispettivamente nel 60%, 53%, 47% e 47% dei casi. La Germania si è classificata al livello più basso tra i Paesi presi in considerazione, e nonostante questo un terzo delle pagine Web tedesche contenevano tracker. L'attività dei cittadini ignari su questi siti può essere utilizzata per dedurre informazioni sensibili sulle loro condizioni di salute e sulla loro situazione di vita comportando quindi una fuga di dati personali teoricamente protetti. I dati ricavati dalla profilazione possono includere: età, sesso, orientamento sessuale, abitudini, interessi, professione, reddito, credenze politiche, religione, malattie e molto altro ancora. Questo "furto" di dati avviene tramite un metodo assimilabile a quello del "cavallo di Troia". Infatti, i siti Web includono spesso script di terze parti per abilitare funzionalità come lettori video, widget per la condivisione sui social, analisi dei dati Web e commenti. Oltre ad espletare la funzione per la quale sono stati creati, tali script generano vulnerabilità nel codice del sito Web, e attraverso queste ultime le compagnie pubblicitarie possono inserire i propri tracker. I dati ricavati vengono poi elaborati e spesso rivenduti dal settore

della tecnologia pubblicitaria. In questo modo, le informazioni sensibili sono sfruttate per sviluppare ulteriori annunci pubblicitari in modo mirato per ogni utente, influenzando potenzialmente i risultati economici, come i punteggi del rischio assicurativo. Per prevenire questo crimine informatico, esistono norme di legge che regolamentano la gestione dei dati privati. Lo Health Insurance Portability and Accountability Act del 1996 (HIPAA) [90] è una legge statunitense che mira a tutelare la copertura assicurativa dei cittadini che si trovano a cambiare impiego o a perdere il lavoro. HIPAA promuove la transizione al formato digitale e l'adozione della cartella elettronica per migliorare la circolazione dei documenti, la loro accessibilità e reperibilità. Oltre ai generici benefici forniti dalla cartella elettronica di cui si è discusso in precedenza (in particolare nel capitolo Telemedicina, a pagina 11), l'introduzione delle nuove tecnologie in ambito sanitario consente una gestione efficiente delle grandi quantità di dati relativi ai pazienti. HIPAA si impegna a regolamentare tale processo, garantendo protezione sui dati che consentono l'identificazione dei pazienti, nonché sulle informazioni relative a diagnosi, esami e situazione economica degli stessi (a quest'ultima si può risalire attraverso i codici di esenzione). Il paziente che vede violati i suoi diritti di privacy ha pertanto il diritto a essere risarcito. Le tutele garantite dall'HIPAA non riguardano soltanto i pazienti, ma abbracciano anche ogni altro ente coinvolto: ospedali, fornitori di servizi sanitari, compagnie assicurative ecc. La normativa HIPAA è stata ampliata ulteriormente nel 2009 dalla legge HITECH (Health Information Technology for Economic and Clinical Health Act). Con l'entrata in vigore di HITECH, le disposizioni HIPAA sono state integrate e aggiornate, introducendo anche un aumento della pena conseguente alle violazioni delle norme precedentemente descritte. HIPAA e HITECH stabiliscono, ad esempio, norme relative all'uso e alla divulgazione delle informazioni sanitarie tutelate, oltre a introdurre degli standard per la protezione di tali dati, per i diritti di privacy individuali e per le responsabilità amministrative. Per accorpare le varie leggi sulla sicurezza dei dati è nato l'HITRUST CSF (Health Information Trust Alliance Common Security Framework) [91]: un framework certificabile e conforme alle norme sopra descritte, utilizzabile da ogni organizzazione che crea, utilizza, immagazzina o trasferisce dati sanitari protetti. Il corrispettivo dell'Unione Europea dei modelli statunitensi di HIPAA e HITECH è costituito dal precedentemente citato GDPR; quest'ultimo utilizza una regolamentazione più dettagliata, offrendo ai pazienti maggiori garanzie. Nel modello GDPR, ad esempio, si richiede il consenso rilasciato attivamente dal paziente sul trattamento delle informazioni personali, e quest'ultimo ha una durata strettamente limitata all'interazione del paziente col servizio sanitario. Il trattamento delle informazioni sanitarie e la regolamentazione sulla sicurezza delle stesse sono di fondamentale importanza per i sistemi di Cloud, proprio a causa delle grandi opportunità di analisi e archiviazione dei dati offerte da questi ultimi.

Infatti, i dati riguardanti la sanità figurano tra le principali applicazioni delle nuove tecnologie di gestione dei BigData.

Google Cloud

Google è un'azienda statunitense con sede in California che offre servizi online, fondata il 4 settembre 1998 da Larry Page e Sergey Brin a Menlo Park, circa un anno dopo la nascita del motore di ricerca Google Search. L'azienda è partita infatti come società di ricerca online, con la funzione di catalogare e indicizzare le varie risorse del Web. In pochi anni diventa il sito più visitato al mondo, tanto che al momento si stima che oltre il 70% delle ricerche online in tutto il mondo venga gestito da Google. Questo l'ha portata ad ingigantirsi fino ad arrivare a rappresentare una delle quattro aziende leader nel mercato high-tech, insieme ad Apple, IBM e Microsoft. Oltre al motore di ricerca, l'azienda ora offre più di 50 servizi o prodotti Internet, come il sistema operativo Android e il sistema operativo Chrome OS, o servizi Web come YouTube, Play Store, Gmail, Google Maps, Google Scholar, Google Earth e molti altri. In particolare, il 7 aprile 2008 l'azienda ha lanciato sul mercato il servizio di Cloud Computing, chiamato Google Cloud, che fornisce infrastruttura come servizio (IaaS), piattaforma come servizio (PaaS) e software come servizio (SaaS), i 3 metodi di servizio di cui abbiamo già trattato nel capitolo a pagina 33. La grossa potenzialità del Google Cloud, è che si basa sulla stessa rete utilizzata dallo stesso Google Search, che è considerata la rete esistente più potente e affidabile. Inoltre, i servizi che offre sono gli stessi che l'azienda ha utilizzato in questi anni e che l'hanno portata al colosso mondiale quale è diventata, sono infatti estremamente efficienti ed affidabili. Il primo servizio reso disponibile, a partire dal 2011, è stato App Engine, una piattaforma per lo sviluppo e il test di applicazioni Web nei data center gestiti da Google, seguito successivamente da molti altri [92]. Come gli altri provider, Google offre vari servizi, tra cui quello di storage, calcolo e analisi di dati.
I servizi offerti dalla piattaforma si possono dividere in 13 macrocategorie:

1. IA e machine learning: basati su reti neurali e caratterizzati da un alto livello di accuratezza. Le stesse applicazioni Google, tra cui Foto (ricerca di immagini), Traduttore, Inbox (Risposta rapida) [93] e l'app Google (ricerca vocale), utilizzano i servizi di machine learning di Google Cloud, gli stessi disponibili come servizio cloud, caratterizzati quindi da un'alta velocità e scalabilità.
2. Gestione delle API: servizio che prende il nome di Apigee. Offre servizi per la progettazione, l'analisi e la pubblicazione delle API (Application Programming Interface) con la massima scalabilità e sicurezza (Apigee Sense).
3. Cloud Services Platform: servizio che prende il nome di Anthos, permette la creazione e la gestione di moderne applicazioni ibride (applicazioni scritte in un linguaggio cross-platform, in

genere Javascript + HTML5, come le web app, che però possono essere incapsulate tramite Web View nel linguaggio nativo di una certa piattaforma).

4. Compute: offre macchine virtuali altamente personalizzabili con funzionalità avanzate, un modello di prezzi basato sull'utilizzo effettivo e la possibilità di eseguire il deployment del codice direttamente o tramite container. Vi sono servizi come Google Kubernetes Engine che consente di utilizzare cluster Kubernetes completamente gestiti per eseguire il deployment, gestire e orchestrare i container su larga scala, o Google App Engine è una Platform as a Service flessibile, che consente di delegare i dettagli operativi del deployment e della gestione dell'infrastruttura.
5. Analisi dei dati: permette l'estrazione in modo efficiente di informazioni aziendali strategiche con i prodotti end-to-end, tra cui il servizio di BigQuery, Cloud Dataflow, Cloud Dataprep, Cloud Composer o Genomics
6. Database: insieme di servizi che permettono l'archiviazione e la gestione di grandi quantità di dati, sia relazionali che NoSQL, che prevedono servizi di database gestiti e non, tra cui Cloud SQL, Cloud Bigtable, Cloud Datastore (NoSQL) o Cloud Firestore, permettendo così l'eliminazione della necessità di un DBMS o di un DBA. Inoltre, i servizi di Database permettono all'aziende di evitare tutta la parte di previsione e di quantificazione della potenza di calcolo necessaria, poiché come gli altri servizi di Google, questi sono altamente scalabili, dotati di una grande potenziale potenza attivabile solo nel bisogno che quindi permette anche un grosso risparmio di tempo, che per le grandi aziende corrisponde a grandi cifre di denaro.
7. Strumenti per sviluppatori: servizio molto utile per gli informatici dell'azienda, che fornisce strumenti e librerie permettendo di sviluppare nuovi programmi più velocemente. Dato che anche Google è un servizio online offerto e creato da informatici, la messa a disposizione degli strumenti utilizzati dall'azienda stessa permettono grandi vantaggi in campo di sviluppo informatico.
8. Archiviazione: uno dei servizi più utilizzati dai clienti. Il servizio prevede spazio di archiviazione potenzialmente illimitato, scalabile a seconda delle necessità, che non risente delle grandi quantità di carico. L'azienda mette a disposizione 15 GB di spazio di archiviazione gratuito sul servizio di Google Drive per chiunque possegga l'account Google.
9. Internet of Things: servizi riguardanti l'interconnessione del servizio di Cloud con dispositivi diversi. Cloud IoT Core permette ad esempio di connettere, gestire e importare dati in tempo reale da milioni di dispositivi dislocati in tutto il mondo.

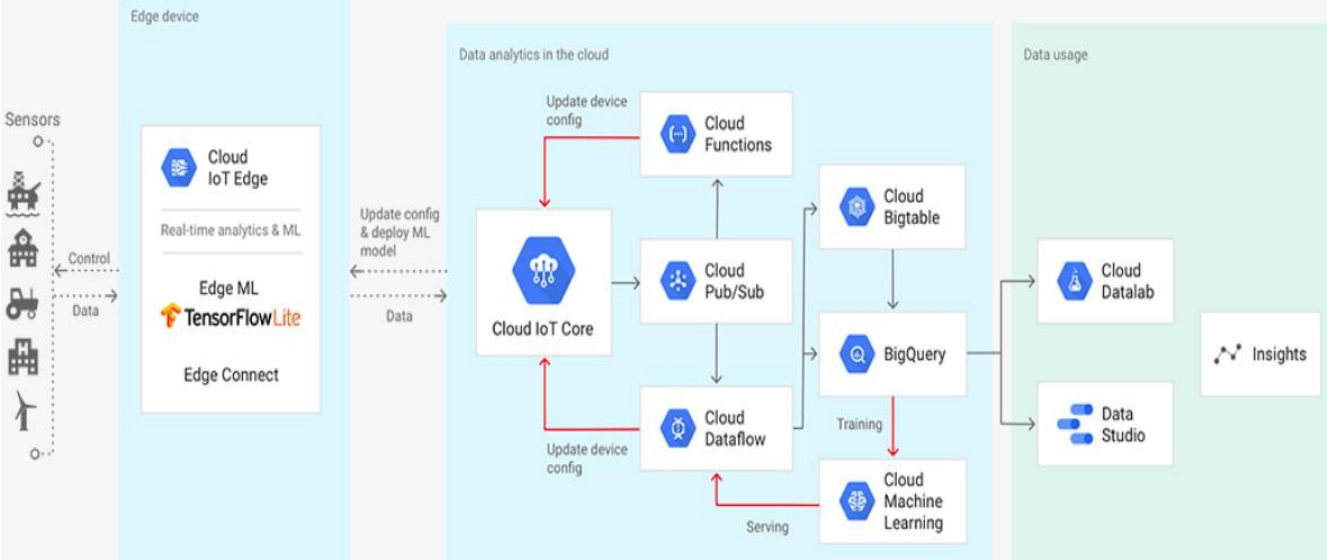

Figura 14 Rappresentazione dell'interazione del servizio di IoT Core con gli altri servizi offerti dal Cloud e con i vari dispositivi. Offre una soluzione completa per la raccolta, l'elaborazione, l'analisi e la visualizzazione di dati IoT in tempo reale, garantendo una maggiore efficienza operativa. [immagine presa dal Cloud di Google, con URL: https://cloud.google.com/iot-core/]

10. Strumenti di gestione: servizio che prende il nome di Stackdriver. Permette il monitoraggio e la gestione real time delle varie applicazioni su GCP e AWS, tramite dispositivi fissi o mobili, fornendo informazioni dettagliate e interfacce molto chiare riguardo le loro prestazioni. Vi sono servizi tra cui Cloud Endpoints (per gestire API) o Deployment Menager (per semplificare il lavoro di gestione).
11. Migrazione: strumenti per il trasferimento e la riorganizzazione di grandi quantità di dati o informazioni, personalizzando il cloud a seconda delle necessità.
12. Networking: basato sulla rete privata di Google, collegata a più di 100 POP (Point of Presence), permettendo il collegamento delle applicazioni da un'area geografica all'altra.
13. Sicurezza: i servizi riguardanti la sicurezza sono vari, e mettono in evidenza la grande affidabilità del Cloud in campo di privacy e visibilità dei dati. Permette di condividere progetti, dati o applicazioni solo con chi e quando si vuole, e di vedere tutti gli utenti che accedono a queste informazioni, tramite semplici operazioni di gestione.

Confrontando Google con gli altri provider, è evidente come il Google Cloud sia in pieno sviluppo e che il suo utilizzo stia avendo in questi ultimi anni un grande incremento. Si colloca tra le migliori opzioni tra i vari Cloud grazie a straordinarie funzionalità di Intelligence, protezione avanzata e svariate opzioni di SaaS.

Una classificazione oggettiva delle potenzialità offerte dal Google Cloud può essere stimata dalla metodologia "The Forrester Wave" [94], che posiziona il servizio offerto al primo posto in ambito di sicurezza, di Database NoSQL e di Database come servizio (Database-as-a-Service).

L’azienda statunitense si è allargata inoltre anche nel settore sanitario, fondando nel 2013 Calico [95], un’azienda indipendente di ricerca e sviluppo nel campo delle biotecnologie, focalizzata sulla ricerca sull’invecchiamento e sulle terapie rivolte alle malattie legate all’età, tra cui il cancro e la demenza senile.

BigQuery

Arriviamo al fulcro della tesi, con il servizio per l'analisi di Big Data offerto dal Google Cloud, chiamato BigQuery. Come è stato esposto precedentemente, l'influenza del Web sulla società moderna ed il suo sviluppo negli ultimi due decenni ha portato alla creazione di grandi quantità di dati, che però causano grosse difficoltà nell'essere riordinati ed analizzati. Memorizzare e interrogare enormi set di dati può essere dispendioso in termini di tempo e denaro senza l'hardware e l'infrastruttura corretti. BigQuery rappresenta la risposta a queste esigenze, fornendo grandi vantaggi in vari ambiti operativi, tra cui anche quello sanitario. Andiamo allora a vedere il procedimento da effettuare per utilizzare questo potente strumento di analisi, dalla creazione o trasferimento del database alla sua interrogazione, e i requisiti che richiedono i vari passaggi. Analizzeremo inoltre le tecnologie che forniscono la sua potenza di calcolo. Saranno affrontate successivamente le sue integrazioni con gli avanzati servizi di intelligence proposti da Google e le recenti ricerche in questo ambito.

Introduzione

BigQuery è il data warehouse [96] (magazzino di dati) aziendale di Google, messo a disposizione per i clienti del Cloud. Solamente pensare al fatto che Google renda disponibile il servizio di analisi di dati utilizzato da lui stesso fa capire le potenzialità di questo servizio. L'azienda statunitense ha infatti gestito fino al giorno d'oggi Big Data (che abbiamo trovato e affrontato già più volte in questa tesi, a partire dal capitolo a pagina 15) ogni secondo di ogni giorno, per fornire servizi come Ricerca, YouTube, Gmail e Google Documenti, che l'hanno portata al grande sviluppo che ha compiuto in questi anni. BigQuery ha le fondamentali caratteristiche di essere serverless e altamente scalabile. Esistono tre modi principali per interagire con BigQuery:

- Importazione ed esportazione dei dati
- Interrogazione e visualizzazione dei dati
- Gestione dei dati

Il servizio permette all'azienda (o al cliente) di rimediare alla mancanza di un potente strumento hardware per l'analisi di grandi data set, e abilita il suo utilizzo in qualsiasi momento senza tempi di attesa. Il suo scopo è quello di facilitare il compito degli analisti di dati, così da aumentarne la loro produttività, con un ottimo rapporto qualità/prezzo. Permette inoltre all'azienda di togliere la figura dell'amministratore di database e di tralasciare l'onere della pianificazione di potenzialità e spazi di archiviazione, necessari per il caricamento e l'interrogazione di dati. BigQuery ha 4 caratteristiche fondamentali che lo descrivono:

1. Operatività immediata: non richiede tempi di attesa per accedere al servizio e compiere le varie operazioni. Il data warehouse è sempre accessibile, indipendentemente dal luogo o dall'orario, ed ha inoltre la possibilità di mantenere i dati in analisi aggiornati in tempo reale.
2. Scalabilità totale (argomento spiegato nel capitolo a pagina 24): potenzialmente il servizio non ha limiti di potenza di calcolo o archiviazione, questi infatti "scalano" (aumentano o diminuiscono) a seconda della richiesta del cliente, senza porre limiti.
3. Potenti strumenti di analisi per l'estrazione di informazioni strategiche: permette l'analisi dei dati archiviati in modo veloce ed efficace tramite l'utilizzo di query SQL. Supporta inoltre potenti strumenti basati su intelligenze artificiali, come Machine Learning e Business Intelligence, per permettere analisi incrociate, dettagliate e affidabili che non sarebbero possibili altrimenti.
4. Protezione degli investimenti e dei dati di business: un grosso punto di forza di questo servizio, come lo è anche della maggior parte degli altri servizi del Google Cloud, è la sicurezza, data da un controllo granulare della gestione di identità e accessi. I dati circolanti su BigQuery, inoltre, che siano in situazione di inattività o di transito, sono sempre criptati.

Grazie a questo servizio, Google ha acquisito varie certificazioni, che lo posizionano tra i migliori provider in circolazione per la gestione e l'analisi di dati.

Gartner, società multinazionale statunitense, leader mondiale nella consulenza strategica, ricerca e analisi nel campo della tecnologia dell'informazione, nomina Google come azienda Leader per la categoria Data Management Solutions for Analytics (DMSA) per il 2019, e lo posiziona nel "Magic Quadrant di Gartner" [97] [98].

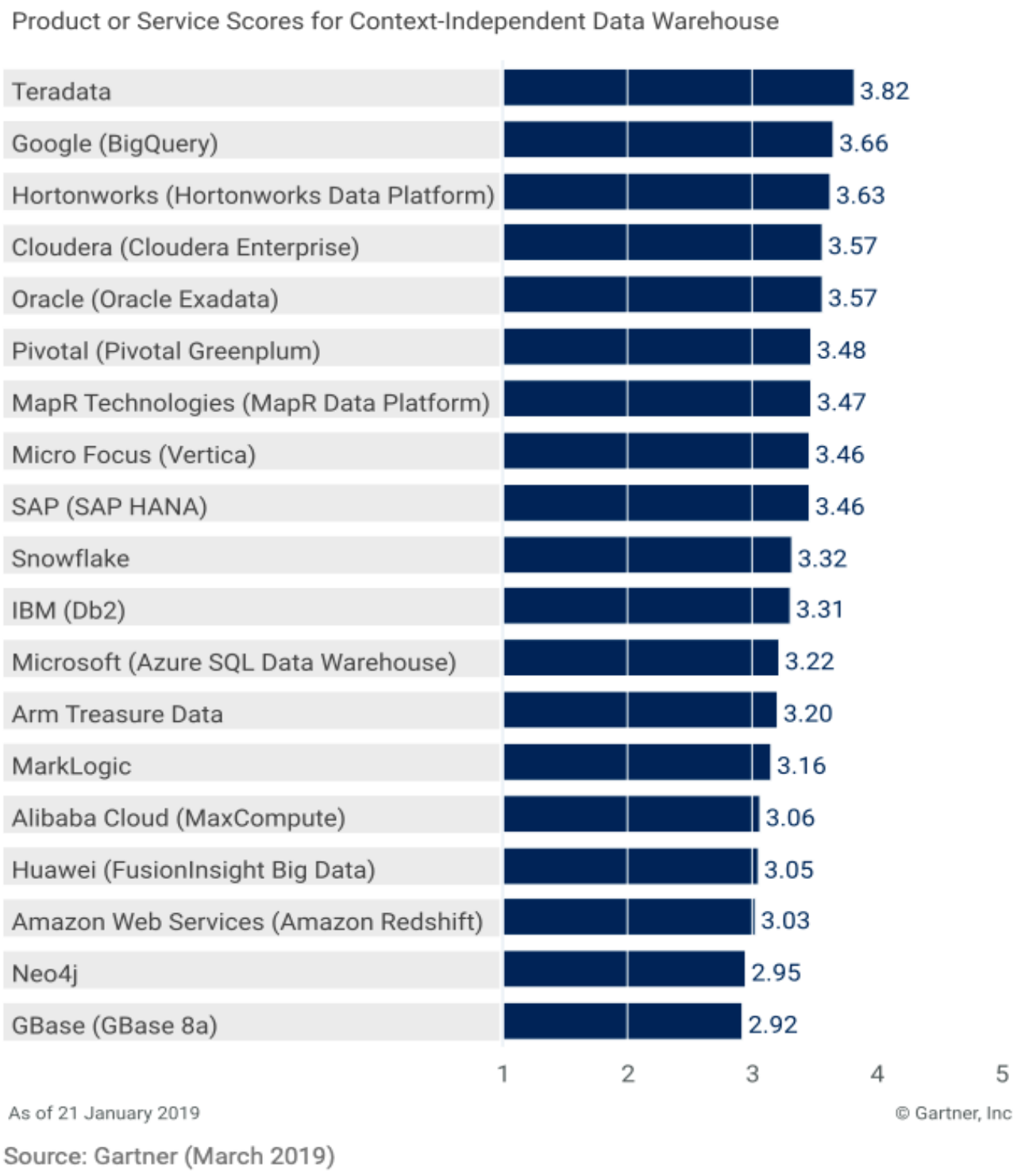

Figura 15 Il grafico illustrato rappresenta il punteggio e la classificazione fornita da Gartner nel marzo 2019 relativamente ai provider in circolazione. Tale classifica si basa sull'efficienza delle operazioni di archiviazione, analisi, trasmissione e condivisione di dati indipendentemente dal contesto di utilizzo. [L'istogramma è stato preso dalla pagina wab teradata, poichè non disponibile sul sito ufficiale di Gartner, con URL: http://www.teradata.com/Press-Releasis/2019/Teradata-Ranked-Highest-in-2019-Gartner]

Un'altra importante certificazione è quella fornita da The Forrester Wave ™, società americana di ricerche di mercato, che fornisce consulenza sull'impatto esistente e potenziale della tecnologia, ai suoi clienti e al pubblico. Anche questa socità nomina Google Cloud come Leader, valutando come ottime le offerte proposte dal Cloud, tanto che il servizio ha ottenuto il punteggio più alto nella

categoria “Strategia” [99]. Confrontando inoltre Google Cloud in ambito di analisi di dati con gli altri servizi di Cloud Computing più rinomati al momento, si nota come Google assuma una posizione rilevante, e come la sua crescita negli ultimi anni sia esponenziale [100].

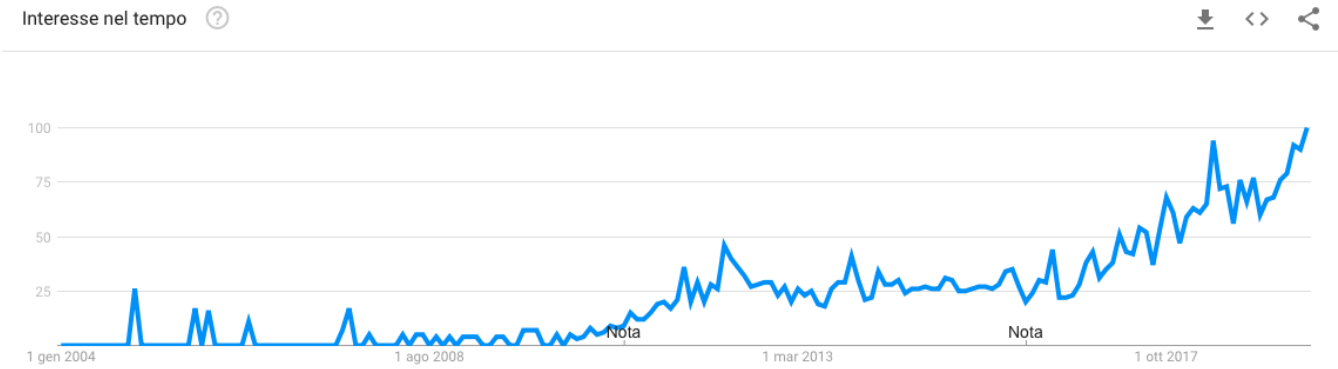

Figura 16 Il grafico in figura rappresenta la crescita dell’utilizzo del Google Cloud in Italia a partire dal 2004 fino ad oggi. [Grafico ricavato da Google Trends, impostando Google Cloud in Italy come analisi]

Il grosso sviluppo e la continua innovazione del Cloud lasciano presagire grandi orizzonti per il futuro prossimo dell’azienda statunitense. Inoltre, nel 2019, Thomas Kurian (ex dirigente Oracle), è stato nominato nuovo dirigente CEO di Google Cloud, promettendo grande competizione per i prossimi anni [101].

Funzionamento

Con BigQuery è possibile creare, popolare e cancellare tabelle, per poi eseguire delle query in linguaggio SQL su di esse. Per poter interrogare un database tramite BigQuery, quindi, è necessario avere una base di dati in formato relazionale. L’utilizzo di BigQuery vede la sua integrazione con altri servizi offerti dal Google Cloud, in particolare con Google Storage, che consente di archiviare e condividere grandi quantità di dati [102]. Tutte le operazioni eseguite tramite l’interfaccia utente (UI) in BigQuery possono essere eseguite anche tramite la riga di comando basata su browser, chiamata Cloud Shell. Per poter iniziare ad usare il servizio di analisi messo a disposizione da Google Cloud, la prima cosa da fare è memorizzare all’interno di esso il database da analizzare (creare cioè la tabella). Per far ciò vi sono varie alternative. Nel caso in cui non si disponga già una tabella specifica da interrogare, BigQuery permette di crearne una partendo da zero. Se i dati sono contenuti in un’applicazione SaaS, tramite l’utilizzo di BigQuery Data Transfer Service [103], è possibile il trasferimento automatico da origini dati esterne come Google Marketing Platform, Google Ads (servizio online di pubblicità) o YouTube. Nell’alternativa in cui si abbia già un database relazionale, se è di dimensioni inferiori a 10 MB è possibile caricarlo direttamente su BigQuery dal computer. Se

il database supera i 10 MB bisogna utilizzare gli altri servizi di archiviazione di Google Cloud, come Google Cloud Bigtable [104], Drive o Google Storage, per poi spostare il file archiviato su BigQuery per poterlo analizzare. Google Storage è il servizio del Cloud che ha la funzione di archiviazione. Può essere utilizzato sia attraverso applicazioni che permettono di accedere a Google Storage da linea di comando (come da terminale), come GSUtil (applicazione Python), sia attraverso la più semplice e intuitiva interfaccia che propone il Google Cloud. Per archiviare qualcosa nel Google Storage è necessario prima di tutto creare un "bucket", che letteralmente significa secchio, e rappresenta infatti un contenitore (proprio come una cartella). Tutto ciò che viene salvato in questo servizio deve essere contenuto in un bucket. A questo punto andiamo a caricare la tabella nel bucket. Dato che è la stessa tabella che successivamente analizzeremo tramite BigQuery, conviene che sia già nel formato giusto, cioè in CSV (comma-separated values). Questo è un formato di file basato su file di testo, utilizzato appositamente per l'importazione e l'esportazione di una tabella da fogli elettronici o database, che converte la tabella in valori separati da virgole (come si intuisce dal nome), trasformando ogni record (riga) della base di dati in una linea di testo, divisa nei rispettivi campi (colonne) rappresentate appunto dalle virgole. Per trasferire il database relazionale su BigQuery, è necessario ridefinire i vari campi della tabella. Per far ciò è possibile utilizzare sia l'interfaccia offerta dal servizio, che scrivere manualmente il codice, assegnando i nomi, la tipologia (data type) e le modalità dei vari campi. Una volta trasferito il database su BigQuery non resta che analizzarlo, sfruttando semplici query in formato SQL. Il servizio supporta due dialetti SQL: SQL standard e SQL legacy [105]. In precedenza, BigQuery eseguiva le query utilizzando un dialetto SQL non standard noto come SQL BigQuery. Con il lancio di BigQuery 2.0, BigQuery ha rilasciato il supporto per SQL standard e ha rinominato BigQuery SQL in SQL legacy. SQL standard è il dialetto SQL preferito per l'interrogazione dei dati memorizzati in BigQuery, ma non è obbligatoria la conversione da SQL legacy a SQL standard, anche se consigliata, poichè lo standard presenta vari vantaggi. SQL standard è conforme allo standard SQL ANSI:2011 e dispone di estensioni che supportano l'esecuzione di query su dati nidificati e ripetuti [106]. Per analizzare i dati, come per le altre operazioni sul Google Cloud, è possibile utilizzare il Cloud Shell, oppure utilizzare l'editor proposto da BigQuery, che nel caso di errori di compilazione rimanderà ad essi per poterli correggere. BigQuery fornisce inoltre driver ODBC e JDBC permettendo l'interazione con applicazioni esterne.

Potenza di analisi

BigQuery rappresenta l'implementazione esterna di una delle tecnologie principali dell'azienda il cui nome in codice è Dremel [107]. Dremel è stato in questi anni lo strumento utilizzato dall'azienda

statunitense per l'analisi di Big Data. Ha un'altissima potenza di calcolo che si traduce in una velocità incomparabile rispetto agli altri provider. Questa potenza è possibile grazie al servizio di query basato sul cloud, che condivide l'infrastruttura di Google, in modo da poter parallelizzare ogni query ed eseguirla su decine di migliaia di server contemporaneamente. Dremel riesce a parallelizzare in maniera così efficiente grazie a due tecnologie basilari.

1. *Architettura ad albero* [108].

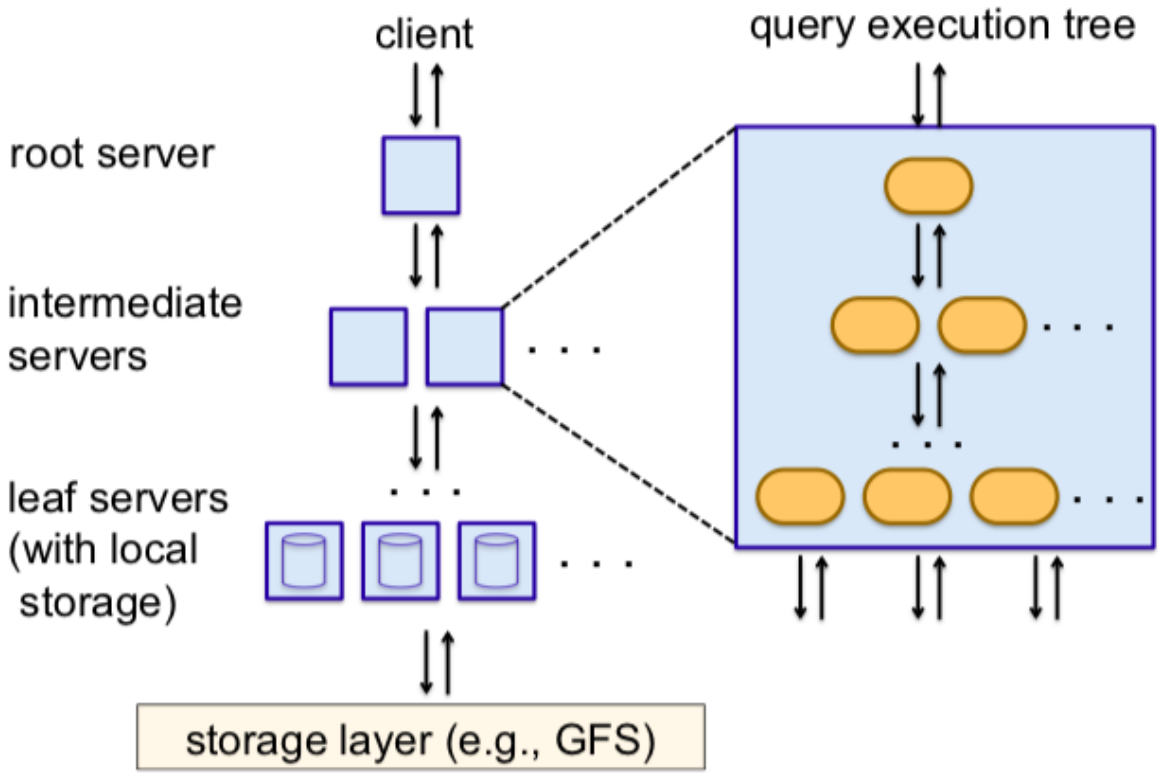

Figura 17 Il grafico riportato rappresenta uno schema semplificato dell'architettura informatica ad albero di Dremel. Nello specifico si vede come la richiesta del cliente venga partizionata nei vari livelli intermedi e come ognuno di questi esegua query semplificate andando a selezionare le informazioni ricercate (contenute nei server foglia). In questo modo, il tempo necessario per ottenere i dati richiesti è ridotto drasticamente. [Illustrazione presa dal S. Melnik, A. Gubarev, J. J. Long, G. Romer, S. Shivakumar, M. Tolton e T. Vassilakis, «Dremel: Interactive Analysis of Web-Scale Datasets,» Proceedings of the VLDB Endowment, *vol. 3, n. 1-2, pp. 330-339, 2010.]*

Utilizzata per inviare query e aggregare i risultati su migliaia di macchine in pochi secondi. In Dremel, i dati vengono archiviati come relazioni nidificate. Lo schema per una relazione è un albero, di cui tutti i nodi (rami) sono attributi e i cui attributi (foglie) contengono valori. Un root server riceve le query in entrata, legge i metadati dalle tabelle e indirizza le query al livello successivo nell'albero di servizio [109]. I server foglia comunicano con il livello di archiviazione o accedono ai dati sul disco locale. Quando il root server riceve una query generica, determina tutti i "tablets", cioè le partizioni orizzontali della tabella, e riscrive la query come una somma di tantissime unioni ("UNION") ordinate ("GROUP BY") che

rappresentano la query iniziale partizionata. A questo punto ogni livello di servizio esegue una riscrittura simile. Infine, le domande raggiungono le foglie con il contenuto richiesto dalla query, che scansionano le partizioni della tabella in parallelo. Durante l'ascesa, i server intermedi eseguono un'agglomerazione parallela dei “tablets”. Il modello di esecuzione presentato sopra è adatto per le query di aggregazione che restituiscono risultati di piccole e medie dimensioni, che sono una classe molto comune di query interattive. Grandi aggregazioni e altre classi di query potrebbero dover contare su meccanismi di esecuzione noti da DBMS e MR (Map-Reduce) paralleli [110]. Al contrario di Pig e Hive (servizio di analisi di dati di AWS), Dremel esegue le query in modo nativo senza tradurle in lavori MapReduce [111] [112] [113]. MR è la tecnologia utilizzata da Google fino al 2006, prima dell’introduzione di Dremel, ed è integrata in Dremel con algoritmi specifici, per aumentarne le potenzialità. Rappresenta un framework progettato per l’analisi di dati su larga scala (petabyte). Poiché l'input di un tipico calcolo di MapReduce è ampio, uno dei requisiti chiave del framework è che l'input non può essere memorizzato su una singola macchina ma deve essere elaborato in parallelo. Una tecnica di progettazione algoritmica generale nel framework MapReduce è chiamata filtering [114]. L'idea principale alla base del filtering è ridurre la dimensione dell'input in modo distribuito, in modo che l'istanza risultante, molto più piccola, possa essere risolta su una singola macchina. In tutti questi casi, gli algoritmi sono parametrizzati in base alla quantità di memoria disponibile sulle macchine, permettendo compromessi tra la memoria disponibile e il numero di round di MapReduce.

Il dialetto Dremel di SQL fornisce quindi query di filtro e aggregazione. Le complicazioni sorgono a causa di attributi ripetuti, cioè attributi a cui è consentito avere più di un valore. La classe comune di query che vengono elaborate su dati archiviati in colonne (di cui parleremo in seguito) si traduce in tempo di elaborazione della query lineare sulla dimensione dei dati rilevanti, ovvero i dati nelle colonne che partecipano alla query. I concetti di contesto di ripetizione e semi-appiattimento giocano un ruolo centrale nella comprensione di questa classe di query e dei loro algoritmi.

2. *Archiviazione colonnare* [115]. Modalità di archiviazione dei dati [116], in particolare un formato di archiviazione condiviso, che consente di ottenere un rapporto di compressione elevato e un’ottima capacità di trasmissione (THR) [117] [118]. Questa tecnologia è fondamentale per la gestione di dati interoperabili. Lo storage colonnare si è rivelato efficace per i dati relazionali piatti, ma per renderlo funzionante per Google è stato necessario adattarlo a un modello di dati nidificati, integrabile con il metodo esposto precedentemente di analisi con architettura ad albero.

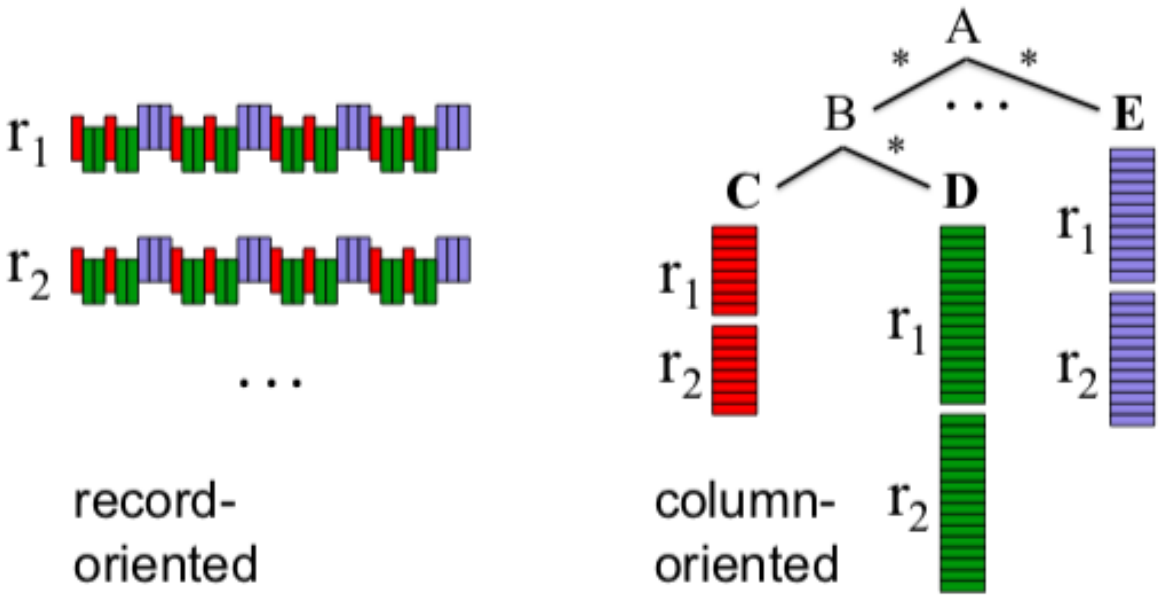

Figura 18 Rappresentazione di dati nidificati in organizzazione lineare e colonnare. La figura illustra un concetto chiave: tutti i valori di un campo nidificato come A.B.C vengono memorizzati in modo contiguo. Quindi, A.B.C può essere recuperato senza leggere A.E, A.B.D, ecc. La sfida affrontata è come conservare tutte le informazioni strutturali ed essere in grado di ricostruire i record da un sottoinsieme arbitrario di campi. È pertanto evidente come l'archiviazione di dati colonnare permetta un'estrazione di dati più veloce rispetto a quella lineare. [Illustrazione presa da S. Melnik, A. Gubarev, J. J. Long, G. Romer, S. Shivakumar, M. Tolton e T. Vassilakis, «Dremel: Interactive Analysis of Web-Scale Datasets,» Proceedings of the VLDB Endowment, *vol. 3, n. 1-2, pp. 330-339, 2010.]*

La rappresentazione di archiviazione a colonne, utilizzata da Dremel, consente di leggere meno dati dallo storage secondario e ridurre i costi della CPU a causa di una compressione più economica [119]. Gli archivi di colonne sono stati adottati per analizzare i dati relazionali, ma al meglio delle nostre conoscenze non sono stati estesi ai modelli di dati nidificati (NoSQL). Il formato di archiviazione colonnare è supportato da molti strumenti di elaborazione dati di Google, tra cui MR, Sawzall e FlumeJava [120] [121].

Un'altra caratteristica di Dremel è quella di essere un sistema multiutente, infatti, possono essere eseguite più query contemporaneamente. Un dispatcher di query pianifica le query in base alle loro priorità e bilancia il carico [122]. L'altro suo ruolo importante consiste nel fornire tolleranza di errore quando un server diventa molto più lento di altri, o quando la replica di un "tablet" diventa inaccessibile.

BigQuery fornisce il set principale delle funzionalità disponibili in Dremel a sviluppatori di terze parti. Lo fa tramite un'API REST, un'interfaccia a riga di comando, un'interfaccia utente Web, il controllo dell'accesso e altro ancora, mantenendo inalterate le prestazioni di Dremel. In questo modo,

BigQuery mira a garantire una comoda e veloce interrogazione tramite query SQL di grandi quantità di dati.

Sicurezza

Un altro punto di forza del sistema di Cloud Computing offerto da Google è quello della sicurezza. L'azienda è stata difatti intitolata Leader in questo ambito, da The Forrester Wave ™, nel secondo semestre del 2019, con un rapporto che riconosce come Google metta al centro della sua strategia proprio la sicurezza dei dati [123]. Questo grazie a servizi come Cloud Data Loss Prevention (DLP) [124], che aiutano a scoprire, classificare e redigere i dati sensibili all'interno della propria organizzazione. Lo scopo dell'azienda è quello di fornire ai clienti metodi facili da adottare per aumentare la visibilità sull'uso dei dati, la condivisione e la protezione nei loro ambienti cloud. Ciò include il centro di comando di Cloud Security per Google Cloud Platform (GCP) e Security Center per G Suite [125], prodotti che aiutano a far emergere informazioni strategiche sulla sicurezza. Inoltre, l'azienda ha creato appositamente un team a tempo pieno, chiamato Project Zero [126], allo scopo di prevenire attacchi mirati, segnalando bug ai fornitori dei software e archiviandoli in un database esterno. Google poi intrattiene da tempo uno stretto rapporto con la comunità di esperti della sicurezza. Il programma "Vulnerability Reward Program" [127] incoraggia i ricercatori a segnalare i problemi di progettazione e implementazione che potrebbero mettere a rischio i dati dei clienti, offrendo premi da decine di migliaia di dollari. Essendo il Cloud Computing una piattaforma connessa a più dispositivi, che offre servizi sotto diversi aspetti, la sicurezza si trova ad essere divisa in più livelli, ognuno di questi fondamentale e complementare agli altri:

- Sicurezza dell'infrastruttura: il livello più alto [128] [129] [130].
- Sicurezza della rete: ha lo scopo di far rispettare i limiti della rete perimetrale, consentendo la segmentazione della rete, l'accesso remoto e la difesa dagli attacchi DoS [131].
- Sicurezza degli endpoint (dispositivi in grado di connettersi alla rete aziendale centrale): per prevenire le compromissioni, è basata sulla protezione avanzata dei dispositivi, sulla loro gestione e sulla gestione di patch e vulnerabilità [132].
- Sicurezza dei dati: ottenuta tramite l'individuazione dei dati, i controlli per prevenire perdite, fuoriuscite e esfiltrazioni e la governance dei dati.
- Gestione dell'accesso e delle identità: le identità degli utenti sono protette gestendo il ciclo di vita, l'autenticazione e la sicurezza degli utenti, oltre che gestendo l'accesso al sistema e alle applicazioni [133] [134] [135] [136] .

- Sicurezza delle applicazioni: ha lo scopo di proteggere e gestire le applicazioni aziendali con test delle applicazioni, scansioni e funzionalità di sicurezza [137] [138].
- Monitoraggio della sicurezza e operazioni: permette di monitorare le attività dannose, di gestire gli incidenti di sicurezza e supportare processi operativi che prevengono, rilevano e rispondono alle minacce [139] [140] [141].

Per ottenere l'alto livello di sicurezza il provider utilizza vari metodi, tra cui la chiave per autenticazione utenti, in modo tale da avere il controllo completo di chi accede ai progetti del Cloud, permettendo una semplice e sicura condivisione di questi ultimi e un monitoraggio accurato di chi può accedervi, e il concetto di "Zero trust" (non fidarsi di nessuno), attraverso il modello BeyondCorp e Security Center [142] [143] [144] [145]. All'interno di BigQuery sono attivi i sistemi per la crittografia e sicurezza dei dati, sia che siano inattivi che in transito, permettendo di avere il controllo di chi può accedere ai dati archiviati nel servizio, tramite "Cloud Identity & Access Menagement" (IAM). Il controllo è basato sul ruolo per le API tramite l'integrazione con Cloud IAM.

Il servizio offre poi un controllo granulare degli accessi, grazie alla separazione delle funzioni di archiviazione e calcolo, è possibile controllare l'accesso a ciascuna delle due. Per ridurre i disagi derivanti dalle modifiche impreviste dei dati, BigQuery replica automaticamente i dati e conserva la cronologia delle modifiche per sette giorni, permettendo di ripristinare facilmente i dati iniziali e di confrontare dati di periodi diversi.
Numerose sono le certificazioni, le normative e gli standard in ambito di sicurezza di cui l'azienda ha ottenuto la conformità. Si citano ISO 27001 [146], SOC 1 [147], PCI DSS [148], CSA STAR [149], FedRAMP [150], FISC (Giappone) [151], GDPR (UE) [152], FIPS 140-2 [153], POPI (Sud Africa) [154], MPAA (https://cloud.google.com/security/compliance/mpaa/) , HITRUST e HIPAA (precedentemente citati) che coprono aree geografiche comprendenti UE e USA, fino ad arrivare al Giappone e al Sud Africa.

Interazione con gli altri servizi collegati

Le interrogazioni dei grandi Database relazionali tramite BigQuery possono essere integrate tramite due servizi di grande sviluppo tecnologico integrati con BigQuery: BigQuery ML e BigQuery GIS.

BigQuery ML

Il servizio consente agli utenti di creare ed eseguire modelli di apprendimento automatico (ML) in BigQuery utilizzando query SQL standard [155]. Il Machine Learning fa parte delle IA, già affrontate in questa tesi nel capitolo a pagina 15, come anche il principio di SQL introdotto invece nel capitolo dei database relazionali a pagina 21. Dato che l'apprendimento automatico su grandi set di dati richiede una vasta programmazione e conoscenza dei framework ML, lo sviluppo della soluzione è limitato solitamente a un gruppo molto ristretto di persone all'interno di un'azienda. BigQuery ML consente agli analisti di dati di utilizzare l'apprendimento automatico attraverso strumenti e competenze SQL esistenti, così che gli analisti possano creare e valutare modelli ML in BigQuery. In questo modo gli analisti non devono più esportare piccole quantità di dati su fogli di calcolo o altre applicazioni e non necessitano più dell'intervento di un team di informatica. BigQuery ML aumenta quindi la velocità di sviluppo, eliminando la necessità di spostare i dati. Alcune tipologie di modelli supportati da BigQuery ML sono:

- Regressione lineare per la previsione. Ad esempio, le vendite di un articolo in un dato giorno.
- Regressione logistica binaria per la classificazione. Ad esempio, determinare se un cliente effettuerà un acquisto.
- Regressione logistica multiclasse. Modelli utilizzati per prevedere più valori possibili, ad esempio, per prevedere se un imput è ad alto, medio o basso valore.

In BigQuery ML, un modello può essere utilizzato con i dati provenienti dai molteplici dataset di BigQuery, sia per il "training" (formazione) del modello, che per la previsione [156]. Vengono addestrati e consultati in BigQuery utilizzando SQL. I modelli, inoltre, sono già creati e messi a disposizione, non è necessario quindi programmare manualmente una soluzione.

BigQuery GIS

Il servizio di BigQuery GIS, permette di sfruttare le informazioni sulla posizione, molto comuni in un data warehouse come BigQuery [157]. I dati sulla posizione sono molto importanti per alcune tipologie di aziende, come quelle per i trasporti, o anche per le aziende di analisi statistiche. Con BigQuery GIS è possibile ad esempio, registrare la latitudine e la longitudine dei veicoli o dei pacchetti in consegna nel tempo [158]. Permette inoltre la registrazione delle transizioni dei clienti e l'unione dei dati di tabelle diverse con i dati sulla posizione. Inoltre, si possono integrare con questo servizio modelli di analisi, per ottenere informazioni di previsione. Ad esempio, diventa possibile determinare quando è probabile che un pacco arrivi, oppure rilevare quali clienti sono più vicini ad

un determinato punto vendita e quindi a chi mandare una mail pubblicitaria. BigQuery GIS consente di analizzare e visualizzare i dati geospaziali in BigQuery utilizzando le semplici funzioni geografiche SQL standard.

Applicazioni in ambito sanitario

Il potente strumento di analisi offerto da Google vede applicazioni anche nel campo medico. Precedentemente è stato affrontato l'utilizzo del Cloud Computing nelle aziende sanitarie, la sua integrazione con i dispositivi medici e con l'IoT, ed i vantaggi che portano, tutto questo nel capitolo a pagina 38. Il Google Cloud si è sviluppato ed è molto sfruttato nell'ambito dell'Healthcare, sia per le potenze di analisi che fornisce, sia per la grande sicurezza che offre, fondamentale per il trattamento dei dati sanitari sensibili (argomento affrontato a pagina 40). Esempi di aziende sanitarie con cui collabora sono Portal Telemedicina, Cleveland Clinic, Hunterdon Healthcare [159], Lahey Health e Colorado Center for Personalized Medicine [160]. In particolare, l'utilizzo di BigQuery, integrato con la scoperta di dati self-service e l'analisi visiva da Tableau, consente a ricercatori e medici di utilizzare Health Data Compass [161], con il fine di identificare e comprendere rapidamente i modelli dei dati genomici per migliorare la qualità, ridurre i costi e accelerare la fornitura di assistenza per risultati terapeutici migliori.

Difatti il settore sanitario necessita di un'analisi veloce di grandi volumi di dati e di ricavare da set di dati diversi informazioni significative. Oltre al semplice ma efficace e fondamentale servizio di analisi, BigQuery integrato alle intelligenze artificiali del Google Cloud, offre vantaggi enormi per il campo sanitario. Uno di questi è quello proposto dalla Foundation of Precision Medicine [162], associazione che si occupa dell'addestramento di algoritmi di apprendimento automatico (ML) con lo scopo di agevolare le previsioni di pattern e relazioni che potrebbero indicare una diagnosi di Alzheimer, con un anticipo di mesi o addirittura anni rispetto alla manifestazione dei sintomi di demenza. Si nota quindi quanto sia rivoluzionaria e innovativa l'applicazione del Machine Learning e delle IA all'analisi dei dati sanitari. Con Cloud Machine Learning Engine [163] e TensorFlow [164], diventa possibile la creazione e l'addestramento di modelli personalizzati per identificare popolazioni, la rilevazione automatica di pattern nei dati di pazienti con il fine di compiere previsioni di risultati clinici. L'integrazione del machine learning con gli studi di imaging diagnostico e il rilevamento automatico di malattie rappresentano il futuro per il campo medico. Tra le ricerche dell'azienda statunitense che sfruttano queste tecnologie vi è lo studio sulla diagnosi di retinopatia diabetica, pubblicato su JAMA [165], dove si tratta della creazione di un algoritmo per il rilevamento automatico della retinopatia diabetica e dell'edema maculare diabetico dalle fotografie del fondo oculare (imaging diagnostico). Un altro esempio di studio dello stesso genere è la ricerca

dell'Università di Stanford sulla diagnosi dei melanomi [166]. L'azienda statunitense si è operata anche nell'ambito per la ricerca contro il cancro [167]. Per identificare nuovi modelli nelle immagini di patologia digitale, l'American Cancer Society [168] ha collaborato con Slalom (partner di Google Cloud specializzato per l'analisi di dati) e ha utilizzato Cloud ML Engine sulla piattaforma di Google Cloud per migliorare la tempestività e l'accuratezza.

Recentemente si è sviluppata la branca della biologia molecolare chiamata "genomica", che si occupa difatti dello studio e dell'evoluzione del genoma (totalità del DNA o dell'RNA) degli organismi viventi. Le variazioni del genoma sono collegabili a specifiche patologie. Il campo sanitario si vede popolato da Big Data riguardanti i genomi, la loro analisi è quindi resa possibile tramite l'utilizzo di potenti processori proprio come BigQuery, che se integrati con processi di ML riescono a prevedere lo sviluppo di patologie. La soluzione che offre il Google Cloud è Google Genomics [169], con il fine di aiutare la community delle scienze biologiche ad organizzare le informazioni genomiche mondiali e a renderle utili ed accessibili [170] [171].

Uno dei progetti a cui Google collabora è MSSNG [172], che ha lo scopo di identificare molti sottotipi di autismo, per sviluppare così trattamenti più personalizzati ed efficaci. L'obiettivo è quello di sequenziare il DNA di 10.000 famiglie affette da autismo, per poter rispondere alle molte domande che ancora vi sono su questo disturbo.

Parte sperimentale

Per verificare le informazioni esposte riguardo al Cloud Computing proposto da Google è stato creato un account di prova. Quest'ultimo consente di testare i vari servizi offerti per un periodo di tempo ridotto, con risorse di calcolo limitate. A tale scopo è stato registrato un nuovo account Google, lo stesso richiesto anche per utilizzare altri servizi offerti dall'azienda statunitense come Gmail o Google Maps. Una volta effettuato l'accesso in Google Cloud, la piattaforma propone varie guide e video-tutorial di presentazione e introduzione. Il primo passo effettuato è stato quello di creare un nuovo progetto, chiamato ProjectTesi2019, su cui sono stati condotti i test riguardo ai servizi di calcolo e storage precedentemente descritti.

Elementi propedeutici per l'interrogazione dei dati: il linguaggio SQL

Per affrontare e capire la successiva parte della tesi in cui saranno compiute delle analisi di database occorre richiamare alcuni concetti già accennati, in particolare mirandoli a ciò che sarà poi utilizzato nella parte pratica della tesi. Per poter analizzare i database relazionali tramite BigQuery non occorrono grandi competenze di informatica, è difatti un servizio che rende semplice ciò che in realtà a livello computazionale sarebbe enormemente complesso. L'unica conoscenza fondamentale per l'utilizzo di questo servizio è quella del linguaggio SQL. Si espongono di seguito alcune tra le basi del linguaggio necessarie per la compilazione delle query che saranno poi poste ai database.

Interrogazioni elementari: rappresentano le interrogazioni che non richiedono ulteriori operazioni da parte del software oltre a quella di ricerca ed estrazione dei dati richiesti. Sono le più utilizzate e sono indispensabili per comporre una query. Hanno una struttura del tipo seguente:

select EspressioneColonna [[**as**] NuovoNomeColonna]
from NomeTabella [[**as**] alias]
where Condizione

-La clausola **select** specifica i componenti dello schema della tabella risultante, quindi si elencheranno a fianco ad essa tutti i campi (colonne) che si vorranno vedere nel risultato della query. Tramite **as** possiamo inoltre assegnare il nome del campo che si avrà nella tabella risultante.

-La clausola **from** elenca tutte le tabelle che devono essere considerate nell'interrogazione, nel nostro caso interrogheremo solitamente una tabella alla volta, avremo quindi accanto a **where** solamente il nome della tabella presa in considerazione. Stavolta non utilizzeremo **as**, poichè serve per quando vengono interrogate varie tabelle, infatti se queste hanno dei campi che si chiamano nello stesso modo per distinguerli vengono aggiunti nella clausola **select** prima del nome del campo l'alias che riconosce la tabella da cui proviene la colonna. L'alias consente quindi anche di riferirsi alle tabelle senza doverne esplicitare per intero il nome e di considerare più volte la stessa tabella nella clausola **from**.

-La clausola **where** specifica la condizione in base alla quale vengono composti i record (righe) della tabella risultante dell'interrogazione. Per la specificazione delle condizioni richieste dalla query si utilizzano i simboli: =, != (diverso. Attenzione a non usare <>), <, >, <=, >=, and, or, not.
La clausola **where** può essere associata a vari comandi che permettono di interrogare più agevolmente i dati. Ne citiamo alcuni che saranno utili per la parte successiva della tesi.
Se il campo selezionato nella clausola **from** è di tipologia "date", "datetime" o "timestamp", è possibile utilizzare il comando year|month|day[NomeColonna]=year|month|day.
In questo modo è possibile porre una query con delle condizioni riferite esclusivamente all'anno, al mese o al giorno del campo preso in considerazione. Questo comando non è valido per tutte le colonne che non hanno una data come tipologia di valore (come ad esempio "string", "integer", "float").
Si fa presente inoltre la condizione utilizzata per selezionare un valore o una specifica stringa all'interno di un campo, metodo che sarà utilizzato più volte. La topografia del comando, per lo standard SQL supportato da BigQuery, per cercare una certa parola all'interno di un campo è:
where NomeColonna=("%ParolaCercata%").
Dato che una tabella può avere righe uguali, il formato SQL standard permette l'eliminazione dei duplicati tramite il comando **distinct**, posto nella clausola **select,** prima del nome del nome del campo di cui si vuole evitare la ripetizione.
SQL mette a disposizione anche la clausola **order by**, che permette di stabilire dei criteri di ordinamento sulle righe del risultato di una interrogazione:
order by NomeColonna [asc|desc].
Operatori aggregati: a differenza delle interrogazioni elementari considerano delle condizioni non legate alle singole righe delle tabelle, sono **count, sum, max, min, avg**. Il primo di questi è molto utilizzato e lo vedremo nelle query che compileremo successivamente. Permette di contare il numero dei record di una colonna, inoltre, se utilizzato assieme a **distinct** conterà il numero di valori diversi nelle colonne specificate, se invece si utilizza l'asterisco si contano tutti i valori della colonna, con all si contano tutte le righe che hanno valori diversi da NULL:
count (< *****| [**distinct**|**all**] ListaColonne >).
Interrogazioni con raggruppamento: offrono la possibilità di considerare gruppi omogenei di righe di una tabella. La sintassi **select** si arricchisce delle clausole **group by** e **having**. La prima delle due serve per individuare un gruppo di righe, mentre la seconda serve per esprimere delle condizioni suglio operatori aggregati. Spesso le due sono utilizzate insieme per esprimere delle condizioni sulle righe di un gruppo individuato da **group by.**
Riassumendo, la sintassi di una generica query SQL è la seguente:
select EspressioneColonna [[**as**] Nuovo Nome Colonna]

from NomeTabella [[**as**] alias]
[**where** Condizione]
[**group by** NomeColonna]
[**having** CondizioneRigheAggregate]
[**order by** NomeColonna [asc|desc]]

Il linguaggio SQL permette poi di comporre interrogazioni di tipo insiemistico e interrogazioni nidificate, per comporre sub-query su livelli multipli, supportate quindi da BigQuery. Con SQL è anche possibile creare e modificare tabelle relazionali, ma BigQuery, per semplificare questa parte, abilita anche la creazione di tabelle tramite interfacce permettendo così di evitare questa parte di progettazione. Altre informazioni utili riguardo il linguaggio SQL standard che utilizzeremo nella successiva parte sono:

-**REGEXP_REPLACE**. Comando che restituisce l'oggetto della stringa con tutte le occorrenze del modello di espressione regolare sostituito dalla stringa sostitutiva. Se non viene trovata alcuna occorrenza, l'oggetto viene restituito così com'è.

La stringa di sostituzione può avere riferimenti secondari alle sottoespressioni nella forma | N, dove N è un numero compreso tra 1 e 9.

-**JOIN**. Comando utilizzato per l'unione di informazioni proveniente da tabelle diverse, che però essendo relazionali sono collegabili avendo un campo identico (spesso indicato con nome diverso per ciascuna tabella, ma contenente i medesimi dati), che rappresenta la "foreign key". Le tabelle legate dal **JOIN** sono indicate nella clausola **from**. Le query utilizzano questo comando sono del tipo:

select EspressioneColonna [[**as**] NuovoNomeColonna]
from NomeTabella [[**as**] alias]
    join NomeTabella [[**as**] alias] on CondizioneJoin
where Condizione.

Applicazione sperimentale per la verifica dell'accessibilità e dell'utilità

Nei capitoli precedenti abbiamo esposto le potenzialità e le molteplici applicazioni del data warehouse proposto da Google. In questa sezione andremo a presentare una valutazione concreta di tale strumento, soffermandoci in particolare sugli aspetti di semplicità e accessibilità della piattaforma. L'unico prerequisito che daremo per scontato consiste in una conoscenza basilare del linguaggio SQL, solitamente comune a tutti gli analisti di dati. Ai fini di testare utilità, efficienza e semplicità del servizio offerto da Google, abbiamo scelto di utilizzare BigQuery per l'analisi di dati clinici, usufruendo unicamente di un comune computer ed una connessione a Internet. Le molteplici applicazioni del servizio BigQuery sono state affrontate precedentemente a pagina 57, in questa

sezione verranno presi in considerazione solo gli aspetti inerenti alle applicazioni in ambito clinico. In particolare ci chiediamo se sia possibile ottenere, da una base di dati contenente prescrizioni dematerializzate, alcune informazioni utili al fine pratico. Miriamo ad ottenere una panoramica del lavoro svolto dai medici all'interno di un'azienda sanitaria, e un'analisi della diffusione di alcune patologie, con il fine di dimostrare l'utilità e la semplicità di utilizzo dello strumento di analisi di cui abbiamo parlato fino ad ora. Per far ciò utilizziamo un database relazionale in formato CSV chiamato DBSANITA. La base di dati contiene le prescrizioni dematerializzate, promemoria della ricetta elettronica, di cui abbiamo trattato nel capitolo delle applicazioni mediche del Cloud Computing a pagina 38. Le informazioni sono raccolte da un'azienda sanitaria (che quindi comprende più sedi ospedaliere) dell'Emilia Romagna, di cui però non citeremo il nome per ragioni di privacy. Infatti, come abbiamo precedentemente illustrato, i dati sanitari sono vincolati da rigide norme che abbiamo accennato nel relativo paragrafo a pagina 40. Per compiere l'analisi sono state omesse le generalità dei pazienti che avrebbero potuto in qualche modo ricondurre alla loro identità. Per caricare su BigQuery la tabella da analizzare abbiamo utilizzato Cloud Storage, servizio messo a disposizione da Google proprio per l'archiviazione e la trasmissione di grandi quantità di dati. Per prima cosa abbiamo creato il bucket, e lo abbiamo chiamato bucket_tesi2019. Successivamente siamo andati ad inserire il DBSANITA all'interno del bucket, passaggio che non ha causato difficoltà, poichè il file caricato si trovava già nel formato CSV richiesto da Google Storage. Successivamente abbiamo iniziato il processo di trasferimento della tabella DBSANITA sulla piattaforma. Per prima cosa è stato necessario creare un set di dati, a cui abbiamo dato il nome di DatiClinici. Poi abbiamo selezionato "Crea Tabella", abbiamo estratto DBSANITA da Google Storage e lo abbiamo inserito nella nuova tabella in BigQuery. Per far ciò sono stati necessari alcuni accorgimenti. In primo luogo abbiamo dovuto selezionare il giusto formato di esportazione CSV, in seguito abbiamo assegnato il nome, la tipologia e la modalità di ciascun campo della tabella. Questa operazione è possibile sia utilizzando il formato JSON come mostrato in Figura 20, sia utilizzando l'interfaccia proposta da BigQuery. I campi della tabella sono 19 e sono rappresentati nella Figura 19. Accorgimento importante, senza il quale la creazione della tabella non poteva avvenire, è stato saltare la prima riga della tabella da importare, poichè rappresentante la riga con i nomi dei campi. Infine abbiamo cambiato il delimitatore di campo riconosciuto di default da BigQuery, che è la virgola (tipica dei file traferiti in CSV), nel punto e virgola, che invece fa al caso nostro.

```
[
    {
        "name": "Data_Prescrizione",
        "type": "DATETIME",
        "mode": "REQUIRED"
    },
    {
        "name": "Codice_Prescrizione",
        "type": "STRING",
        "mode": "REQUIRED"
    },
    {
        "name": "Programma_Prestazioni",
        "type": "STRING",
        "mode": "NULLABLE"
    },
    {
        "name": "Prescrittore",
        "type": "STRING",
        "mode": "NULLABLE"
    },
    {
        "name": "Paziente",
        "type": "STRING",
        "mode": "NULLABLE"
    },
    {
        "name": "Data_Nascita",
        "type": "DATETIME",
        "mode": "REQUIRED"
    },
    {
        "name": "Sesso",
        "type": "STRING",
        "mode": "REQUIRED"
    },
    {
        "name": "Residenza",
        "type": "STRING",
        "mode": "REQUIRED"
    },
    {
        "name": "Classe_Priorita",
        "type": "STRING",
        "mode": "NULLABLE"
    },
    {
        "name": "Esenzione_Reddito",
```

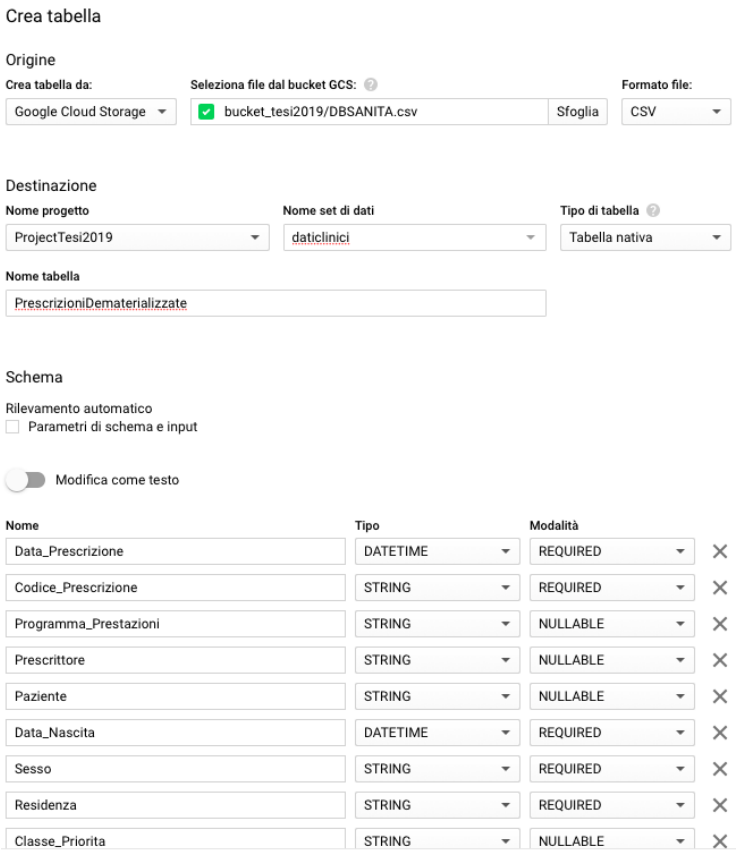

Figura 19 Nell'immagine sovrastante è rappresentata come appare l'interfaccia proposta da BigQuery per la creazione della tabella. Si vede come l'origine della tabella sia Google Cloud Storage ed il formato sia CSV. Nella parte inferiore dell'immagine si vedono elencati i vari campi della tabella, il nome, il tipo e la modalità, inseriti manualmente in base alla tipologia dei dati raccolti nel database di partenza.

Figura 20 Nalla figura di sinistra è rappresentata l'altra possibile soluzione per scrivere la struttura della tabella che risulterà su BigQuery. In particolare il formato è JSON. Si notano specificate le caratteristiche dei campi, le stesse viste in Figura 19.

Una volta creata la tabella, a cui abbiamo dato il nome di PrescrizioniDematerializzate, è stato possibile visualizzarla in anteprima, come si mostra in Figura 21, così da capire in pratica cosa contenesse. Il database considerato in particolare è composto da 1.042.106 record ed occupa uno spazio di archiviazione di 199,05 MB. Una volta compiute queste operazioni è stato possibile interrogare il database. Le prime query che abbiamo posto, riportate assieme ai rispettivi risultati, sono state di carattere generico, in particolare ci siamo chiesti quanti fossero i pazienti che avessero ricevuto prescrizioni dematerializzate per il mese di marzo 2019, come si vede in Figura 22, e tra questi quante fossero le femmine e quanti fossero i maschi, come mostrato nelle corrispettive Figura

25 e Figura 24. Inoltre abbiamo riportato il tempo impiegato da BigQuery per eseguire la prima query, come si può osservare in Figura 23, così da poterlo confrontare in successive analisi.

Schema Dettagli Anteprima

Riga	Data_Prescrizione	Codice_Prescrizione	Programma_Prestazioni	Prescrittore	Paziente	Data_Nascita	Sesso	Residenza	Classe_Priorita	Esenzione_Reddito	Codice_Esen
1	1/4/2019 00:00:00	080A05080055516	1	NL61C1	RC77C1	14/3/1977 00:00:00	M	FC	D	null	RFG060
2	1/4/2019 00:00:00	080A05080055520	1	NL61C1	RC77C1	14/3/1977 00:00:00	M	FC	D	null	RFG060
3	1/4/2019 00:00:00	080A05080056391	1	MR60H5	NT52R3	30/10/1952 00:00:00	M	RA	D	null	009
4	1/4/2019 00:00:00	080A05080056397	1	NT51H2	RZ51S6	22/11/1951 00:00:00	F	RA	D	null	0A31
5	1/4/2019 00:00:00	080A05080056889	1	LL56B6	NO72H6	27/6/1972 00:00:00	F	FC	D	null	044
6	1/4/2019 00:00:00	080A05080056906	1	null	RN41C4	6/3/1941 00:00:00	F	FC	D	null	025
7	1/4/2019 00:00:00	080A05080056921	1	NN62D2	NE71R5	16/10/1971 00:00:00	F	RA	D	null	025
8	1/4/2019 00:00:00	080A05080056921	2	NN62D2	NE71R5	16/10/1971 00:00:00	F	RA	D	null	025
9	1/4/2019 00:00:00	080A05080056921	3	NN62D2	NE71R5	16/10/1971 00:00:00	F	RA	D	null	025
10	1/4/2019 00:00:00	080A05080056921	4	NN62D2	NE71R5	16/10/1971 00:00:00	F	RA	D	null	025
11	1/4/2019 00:00:00	080A05080056921	5	NN62D2	NE71R5	16/10/1971 00:00:00	F	RA	D	null	025
12	1/4/2019 00:00:00	080A05080056921	6	NN62D2	NE71R5	16/10/1971 00:00:00	F	RA	D	null	025
13	1/4/2019 00:00:00	080A05080056925	1	NN62D2	NE71R5	16/10/1971 00:00:00	F	RA	D	null	0A31
14	1/4/2019 00:00:00	080A05080057141	1	null	RA89C6	20/3/1989 00:00:00	F	RA	B	null	M50

Righe per pagina: 100 1 - 100 di 1042106 Prima pagina Ultima pagina

Figura 21 Visualizzazione in anteprima della tabella creata su BigQuery. Si evidenzia così il contenuto dei vari campi, osservando il significato pratico di ciascuno di essi. Si nota inoltre il numero di righe (in basso). Il campo Data_Prescrizione ha lo stesso contenuto per ogni record, poichè la tabella rappresenta le prescrizioni dematerializzate dell'azienda sanitaria in riferimento al mese di Marzo 2019, quindi il sistema ha aggiornato tutto mettendo tutto nella stessa data.

```
SELECT
  COUNT(DISTINCT Paziente) AS NPazienti
FROM
  `projecttesi2019.daticlinici.PrescrizioniDematerializzate`
```

Riga	NPazienti
1	141282

Figura 22 A sinistra query SQL in cui viene interrogata la tabella in Figura 21 al fine di ottenere il numero totale di pazienti che hanno avuto una prescrizione. A destra si ha la tabella risultante della rispettiva query.

Tempistica worker

Fasi		Attesa	Lettura	Calcolo	Scrittura		Righe
S00: Input	Media:	1 ms	43 ms	476 ms	9 ms	Input:	1.042.106
	Max:	1 ms	43 ms	476 ms	9 ms	Output:	141.282
S01: Aggregate	Media:	0 ms	0 ms	85 ms	2 ms	Input:	141.282
	Max:	0 ms	0 ms	85 ms	2 ms	Output:	1
S02: Output	Media:	1 ms	0 ms	5 ms	5 ms	Input:	1
	Max:	1 ms	0 ms	5 ms	5 ms	Output:	1

Figura 23 Viene mostrata la tempistica impiegata da BigQuery per compiere la query in Figura 22, diviso nelle tre fasi di Input, Aggregazione e Output. Si nota come il tempo totale impiegato sia dato principalmente dalla somma dei tempi di lettura e di calcolo nella fase di Input e dal calcolo della fase di aggregazione.Vediamo come costituiscano tempi irrisori, che sommati non arrivano ad un secondo. A destra vediamo inoltre il numero di righe in ingresso (Input) analizzate al fine di rispondere all'interrogazione posta

(corrispondenti al numero totale di record della tabella), e il numero di righe in uscita (Output), che corrisponde al numero di pazienti che hanno avuto una prescrizione (risultato della query). Questo numero è stato poi l'Input della fase di aggregazione, data dal comando nella query "Count", così da restituire un'unica riga in uscita contenente l'informazione ricercata.

```
SELECT
  COUNT(DISTINCT Paziente) AS NPazienti
FROM
  `projecttesi2019.daticlinici.PrescrizioniDematerializzate`
WHERE
  Sesso="M"
```

Riga	NPazienti
1	62603

Figura 24 Si riporta la query (a sinistra) ed il corrispettivo risultato (a destra) posta col fine di ottenere il numero di pazienti maschili che hanno ricevuto almeno una prescrizione nel mese di marzo.

```
SELECT
  COUNT(DISTINCT Paziente) AS NPazienti
FROM
  `projecttesi2019.daticlinici.PrescrizioniDematerializzate`
WHERE
  Sesso="F"
```

Riga	NPazienti
1	78679

Figura 25 L'immagine mostra la query ed il corrispettivo risultato per interrogare il database sul numero di pazienti femminili che hanno ricevuto una prescrizione. Si nota come il risultato ottenuto sia maggiore di quello mostrato in Figura 24, verosimile con le analisi statistiche sulla popolazione. Si può vedere inoltre come la somma di queste ultime due query sia uguale al numero di pazienti totale, calcolato con la query in Figura 22.

Successivamente, abbiamo interrogato il database per ottenere informazioni più utili in ambito medico. Spesso si è soliti monitorare il lavoro effettivo compiuto da ciascun medico. Quindi, per avere un'idea sul quantitativo di lavoro compiuto, ci siamo chiesti quanti fossero i pazienti visitati da ogni medico nel mese di marzo.

```
SELECT
  COUNT(DISTINCT Paziente) AS NPazienti,
  Prescrittore
FROM
  `projecttesi2019.daticlinici.PrescrizioniDematerializzate`
WHERE
  Prescrittore != "NULL"
  AND Paziente != "NULL"
GROUP BY
  Prescrittore
ORDER BY
  Npazienti DESC
LIMIT
  10
```

Riga	NPazienti	Prescrittore
1	436	FN88L0
2	357	BA53C1
3	334	GR64A6
4	322	RZ55H1
5	316	RD64M0
6	306	RZ61P6
7	304	DR57C3
8	304	TN51T1
9	289	ZE56L2
10	287	NN53L2

Figura 26 A sinistra si mostra la query posta per trovare il numero di pazienti visitati da ogni prescrittore nel mese di marzo. Si nota come sia stata compilata selezionando il numero di pazienti ed il corrispettivo prescrittore, ordinando il risultato in ordine decrescente, così da avere come risultato il codice dei primi 10 prescrittori con un maggior numero di pazienti. A destra è esposto il risultato dell'interrogazione spiegata.

Infine abbiamo voluto interrogare il database con un'analisi di carattere più scientifico, così da mostrare una potenziale utilità di BigQuery nell'ambito della ricerca. Ci siamo chiesti quante fossero le persone affette da diverse tipologie di polmonite residenti a Forlì-Cesena (FC).

```
SELECT
  COUNT(Quesito_Diagnostico) AS NPolmonite
FROM
  `projecttesi2019.daticlinici.PrescrizioniDematerializzate`
WHERE
  Residenza="FC"
  AND Quesito_Diagnostico LIKE ("%polmonite%")
```

Riga	NPolmonite
1	160

Figura 27 A sinistra si mostra la query utilizzata per interrogare il database sul numero di persone a cui è stata diagnosticata una probabile malattia polmonare. Si nota come la ricerca effettuata nel campo "Quesito_Diagnostico" non sia stata specifica per una tipologia di polmonite, ma anzi, grazie al comando standard SQL "LIKE", siano state ricercati ed infine contati tutti i pazienti affetti da diverse tipologie di polmonite. A destra è mostrato la tabella risultante di questa query.

```
SELECT
  COUNT(Quesito_Diagnostico) AS NPolmonite
FROM
  `projecttesi2019.daticlinici.PrescrizioniDematerializzate`
WHERE
  Residenza="RA"
  AND Quesito_Diagnostico LIKE ("%polmonite%")
```

Riga	NPolmonite
1	126

Figura 28 La query rappresentata è molto simile a quella mostrata in Figura 27, con la differenza che in questa la residenza dei pazienti è Ravenna. A destra si evidenzia come il risultato di pazienti affetti da malattie polmonari sia minore.

Si nota come il numero di persone affette da malattie di questo tipo sia differente a seconda della residenza. Da queste query sarebbe possibile trarre conclusioni in riferimento alla diffusione della malattia in una determinata zona, e le relative cause. I dati analizzati però non sono però sufficienti a stimare conclusioni di carattere statistico, poichè il risultato trovato potrebbe variare considerevolmente in base a molteplici aspetti, come ad esempio, nel caso considerato, la popolazione totale di Ravenna e di Forlì-Cesena. Difatti risulta che la popolazione di Forlì-Cesena sia enormemente maggiore (più del doppio) rispetto a quella di Ravenna, ciò spiegherebbe come vi siano più prescrizioni per pazienti con residenza FC. Un'analisi di questo tipo, se ampliata a grandi zone e ripetuta nel corso del tempo, può portare a conclusioni importanti. Analizzando la residenza delle persone con malattie alle vie respiratorie, in particolare polmonari, si possono trarre conclusioni di ben più alto impatto che il semplice numero di persone affette da polmonite in una determinata città. Ad esempio si possono legare le malattie respiratorie all'inquinamento atmosferico delle varie località geografiche, oppure si può monitorare la diffusione di epidemie in certe zone. Analisi di questo genere possono essere svolte anche in base al sesso o all'età, per ottenere ad esempio la distribuzione di

diverse patalogie su diverse fasce d'età. Per fare questo occore però analizzare database molto più grandi di quello che abbiamo appena interrogato, che arrivano ad occupare spazi di archiviazione dell'ordine dei GB o addirittura dei TB.

Applicazione sperimentale per la verifica delle potenzialità

In quest'ultima parte della tesi ci chiediamo quanto effettivamente lo strumento di analisi sia potente, se sia cioè in grado di soddisfare le esigenze esposte alla fine dello scorso paragrafo. Il database precedentemente interrogato, nonostante abbia al suo interno ben 19.800.014 dati (rappresentanti ciascuno una potenziale informazione) riproduce una base di dati relativamente piccola, che, come mostrato in Figura 23, BigQuery riesce ad analizzare in tempi irrisori. Evidenzia quindi l'utilità e la semplicità di BigQuery sotto l'aspetto pratico, ma non mette in piena luce le sue potenzialità. Come abbiamo già illustrato nei capitoli precedenti, a partire già dal capitolo delle intelligenze artificiali a pagina 15, la società moderna a cui apparteniamo è popolata da grandi quantità di dati, i Big Data, che molti software di analisi fanno fatica ad interrogare. Queste moli di dati, se analizzate e trasformate in informazioni, permettono di arrivare a conclusioni fondamentali sotto il punto di vista scientifico, coprendo svariati ambiti. Con l'integrazione del machine learning permettono addiruttura di rivoluzionare il campo della ricerca medica, aprendo varchi che prima non sarebbero stati possibili, come abbiamo accennato nel paragrafo chiamato BigQuery ML.

Ci chiediamo se BigQuery sia davvero in grado di analizzare i BigData e di permettere le operazioni citate. Per far ciò si sfruttano le basi di dati fornite da Google Cloud. Abbiamo già visto le prestazioni dello strumento per database di modeste dimensioni, andiamo allora scalando fino ad arrivare ad enormi dataset, così da confrontare le diverse prestazioni dello strumento al variare del lavoro richiesto, e tramite i quali si vuole testare l'effettiva potenza di analisi del data warehouse. Per sfruttare i data set messi a disposizione dall'azienda statunitense già su BigQuery, composti quindi da tabelle relazionali già nel giusto formato e pronte per essere interrogate, è necessario creare un nuovo progetto con localizzazione US, che andiamo a chiamare ProjectPotenza. I database che possiamo utilizzare per testare le capacità dello strumento sono molti, e variano da raccolte di dati sportivi, a dati raccolti da FDA (Food and Drug Administration), a censimenti, fino ad arrivare a dati relativi al campo della criminologia, dei bitcoin e molti altri.

Il primo database che prendiamo in considerazione è situato nel data set chiamato "samples", in particolare vogliamo analizzare il database "Wikipedia". Questo è un insieme di dati rappresentato come una tabella relazionale, costituito da una singola riga di dati per ogni revisione o aggiornamento di Wikipedia. Tra i campi della tabella vi sono inclusi il cognome (contributor_username) di chi ha

aggiornato l'articolo e il titolo (title) dell'articolo stesso. Il database occupa uno spazio di archiviazione di 35,69 GB, ed ha ben 313.797.035 righe.

ID tabella	bigquery-public-data:samples.wikipedia
Dimensione tabella	35,69 GB
Dimensione spazio di archiviazione a lungo termine	35,69 GB
Numero di righe	313.797.035
Data/ora creazione	14 mar 2016, 18:16:47
Scadenza tabella	Mai
Ultima modifica	14 mar 2016, 18:16:47
Località dei dati	US

Figura 29 L'immagine mostra le caratteristiche del database "wikipedia".

Vogliamo selezionare le voci di Wikipedia contenenti la parola "Italy", e ci chiediamo il numero di aggiornamenti o revisioni di queste ultime. Quindi interroghiamo la base di dati per vedere come si comporta BigQuery nell'analizzare un database di queste dimensioni, quanto tempo ci mette e se compie degli errori di overflow.

Figura 30 A sinistra è raffigurata la query posta al database, che lo interroga riguardo le voci contenenti la parola "Italy", sfruttando la stessa clausola utilizzata nella query mostrata in figura 28. A destra si ha la tabella risultante, caratterizzata dal campo "title", in si nota essere presente la parola "Italy", e il numero di revisioni/aggiornamenti effettuati.

```
SELECT
  title,
  COUNT(title) AS num_revisioni
FROM
  `bigquery-public-data.samples.wikipedia`
WHERE
  title LIKE ("%Italy%")
GROUP BY
  title
ORDER BY
  num_revisioni DESC
LIMIT
  10
```

Riga	title	num_revisioni
1	Italy	10768
2	Italy national football team	4271
3	Kingdom of Italy (1861–1946)	2264
4	Culture of Italy	1740
5	User:AlexNewArtBot/ItalySearchResult	1452
6	History of Italy	1431
7	Talk:Italy	1424
8	Music of Italy	1382
9	Military history of Italy during World War II	998
10	Economy of Italy	993

Tempo trascorso	Tempo utilizzato da slot	Byte ordinati in modo sistematico	Byte con overflow su disco
2,9 sec	2 min 44,688 sec	3,03 MB	0 B

Figura 31 L'immagine mostra il tempo di esecuzione della query, che stavolta supera la soglia del secondo, ma che non ha comunque byte di overflow e restituisce il risultato cercato in 2,9 secondi. Il tempo utilizzato da slot rappresenta il tempo utilizzato da un singolo slot per elaborare la query. Uno slot è un'unità di capacità di calcolo richiesta per eseguire le query SQL. Il numero di slot utilizzato da BigQuery varia a seconda della dimensione e della complessità della query.

Come abbiamo esposto nel capitolo Potenza di analisi, il funzionamento di BigQuery è basato sulla metodologia di analisi ad albero e sulla tipologia di archiviazione colonnare, che permette uno "scanning" veloce dei dati utili per compilare la query posta, rendendo ottime prestazioni anche per interrogazioni su più colonne di grandi database. Per verificare queste informazioni poniamo una query che richieda l'analisi del database "Wikipedia" su più colonne. In particolare vogliamo sapere il cognome, il titolo e il numero di revisioni/correzioni che ciascun contributore ha fatto su un file di Wikipedia contenente nella voce il termine "Italy". Inoltre abbiamo richiesto che il risultato fosse riportato in ordine decrescente.

```sql
SELECT
  contributor_username,
  title,
  COUNT(contributor_username) AS num_revisions
FROM
  `bigquery-public-data.samples.wikipedia`
WHERE
  title LIKE ("%Italy%")
GROUP BY
  contributor_username,
  title
HAVING
  num_revisions > 50
ORDER BY
  num_revisions DESC
```

Figura 32 A sinistra si può osservare la query posta al database "wikipedia", con il fine di testare le prestazioni di BigQuery su un'analisi ramificata di un database di 35,69 GB. Nel codice si nota come sia stata utilizzata la clausola HAVING, che ancora non era stata utilizzata nelle query precedenti, così da avere nella tabella risultante soltanto i "contributor_username" che abbiano fatto almeno 50 revisioni.

Riga	contributor_username	title	num_revisions
1	AlexNewArtBot	User:AlexNewArtBot/ItalySearchResult	1437
2	R-41	Kingdom of Italy (1861–1946)	999
3	AlexNewArtBot	User:AlexNewArtBot/ItalyLog	967
4	Conte di Cavour	Italy	726
5	Jeffmatt	Music of Italy	708
6	WP 1.0 bot	Wikipedia:Version 1.0 Editorial Team/Italy articles by quality log	324
7	WP 1.0 bot	Wikipedia:Version 1.0 Editorial Team/Italy articles by quality	316
8	Schnurrbart	List of rivers of Italy	308
9	WP 1.0 bot	Wikipedia:Version 1.0 Editorial Team/Italy articles by quality statistics	298
10	Lib3rtarian	Libertarian Movement (Italy)	276
11	Theologiae	Italy	275
12	Sicilianmandolin	Italy	261

Figura 33 La figura rappresenta una parte della tabella risultante della query sovracitata. La query infati non poneva un limite alla tabella risultante, richiedeva solamente che il numero di revisioni fosse maggiore di 50, perciò il risultato della query è in realtà una tabella composta da molte più righe, che per comodità non abbiamo riportato. Dal risultato si nota come i contributori che hanno fatto più revisioni siano bot, informazione interessante che non ci si aspettava di rilevare.

Tempo trascorso	Tempo utilizzato da slot	Byte ordinati in modo sistematico	Byte con overflow su disco
11,2 sec	2 min 54,903 sec	4,87 MB	0 B

		Tempistica worker					
	Fasi		Attesa	Lettura	Calcolo	Scrittura	Righe
✓	S00: Input	Media:	835 ms	357 ms	225 ms	6 ms	Input: 313.797.035
		Max:	1067 ms	656 ms	303 ms	149 ms	Output: 98.148
✓	S01: Aggregate+	Media:	27 ms	0 ms	9 ms	13 ms	Input: 98.148
		Max:	29 ms	0 ms	14 ms	44 ms	Output: 126
✓	S02: Output	Media:	5 ms	0 ms	6 ms	7 ms	Input: 126
		Max:	5 ms	0 ms	6 ms	7 ms	Output: 126

Figura 34 L'immagine mostra la tempistica di lavoro che è stata necessaria a BigQuery per analizzare il database "wikipedia", col fine di ricavare le informazioni richieste dalla query in Figura 33. Si nota come il tempo trascorso sia di 11,2 secondi, contro i 2,9 impiegati per risolvere la query precedente posta sullo stesso database.

Andiamo adesso a vedere il comportamento della macchina di analisi su database di tutt'altre dimensioni. Per eseguire questo test si cercano tra i dataset proposti da BigQuery quelli caratterizzati da uno spazio di archiviazione maggiore. Notiamo come i database più pesanti siano proprio quelli che abbiamo introdotto nel capitolo delle applicazioni in ambito sanitario, a pagina 57, dove abbiamo accennato l'utilizzo di BigQuery per studi sulla genetica, analisi di Big Data contenenti informazioni riguardanti miliardi di persone. Ci chiediamo se BigQuery possa svolgere query di carico così elevato, e coprire così un ruolo fondamentale per lo sviluppo della ricerca medica. I database trovati riguardo all'ambito della genomica si suddividono in tabelle partizionate e non, che arrivano a pesare anche oltre i 5 TB. Un esempio tra i database sulla genomica umana messo a disposizione da Google è:

ID tabella	bigquery-public-data:human_genome_variants.1000_genomes_phase_3_variants_20150220
Dimensione tabella	5,38 TB
Dimensione spazio di archiviazione a lungo termine	5,38 TB
Numero di righe	84.801.880
Data/ora creazione	23 ago 2018, 23:16:32
Scadenza tabella	Mai
Ultima modifica	21 set 2018, 17:23:26
Località dei dati	US
Tipo di tabella	Partizionata
Partizionata in base a	Day
Partizionata nel campo	partition_date_please_ignore
Filtro partizione	Non richiesto
Raggruppato in cluster da	reference_name, start_position, end_position

Figura 35 L'immagine mostra le caratteristiche della tabella 1000_genomes_phase_3_variants_20150220 appartenente al data set messo a disposizione in BigQuery chiamato human_genome_variants. Si nota che la tipologia della tabella è "partizionata" e che la dimensione dello spazio di archiviazione del database è ben 5,38 TB.

Per valutare le potenzialità di BigQuery a fronte di una mole di dati maggiore di quelle testate finora, si è scelto di interrogare un database delle dimensioni di 1,5 TB. In particolare, abbiamo fatto uso della tabella "Rice3K_DeepVariant_Os_Nipponbare_Reference_IRGSP_1_0", uno dei database forniti gratuitamente dal servizio di Google ad uso di ogni account, situato nel data set "genomics_rice". Questa tabella è un valido esempio di BigData, in quanto contiene i genomi di oltre 3000 tipologie di riso, catalogati in più di 12 miliardi di righe. L'utilizzo delle query di ricerca nell'ambito della genetica costituiscono un perfetto esempio delle potenzialità di applicazione di questa tecnologia; infatti, le operazioni di analisi e ricerca in un database così esteso richiederebbero un lavoro umano talmente lungo da renderlo totalmente irrealizzabile. In questo caso sfrutteremo BigQuery per interrogare il dataset circa la distribuzione di genotipi di riso nei campioni classificati, valutando le frequenze di alleli in relazione al principio di Hardy–Weinberg (ossia la legge che postula la costanza delle frequenze dei genotipi di una popolazione attraverso le generazioni).

ID tabella	bigquery-public-data:genomics_rice.Rice3K_DeepVariant_Os_Nipponbare_Reference_IRGSP_1_0
Dimensione tabella	1,5 TB
Dimensione spazio di archiviazione a lungo termine	1,5 TB
Numero di righe	12.186.710.727
Data/ora creazione	4 ott 2018, 15:16:37
Scadenza tabella	Mai
Ultima modifica	4 ott 2018, 15:16:37
Località dei dati	US

Figura 36 La tabella mostra i dettagli del database che andiamo ad interrogare. In particolare notiamo l'importante spazio di archiviazione occupato, di 1,5 TB, caratteristico dei Big Data. Si vuole inoltre evidenziare il numero di righe che la costituiscono, equivalente a 12.186.710.727. I campi principali che cosituiscono la tabella sono "reference_name" (nome di riferimento), "start_position" (inizio della posizione di DNA considerato), "end_position", "reference_bases" e "alternate_bases.alt" (basi azotate, timina, adenina, citosina, guanina e uracile, che caratterizzano ogni tipologia di riso).

```
-- La seguente query elabora l'equilibrio di Hardy-Weinberg per le varianti alleliche.
--
WITH variants AS (
  SELECT reference_name, start_position, end_position, reference_bases, alt,
  SUM(HOM_REF) AS HOM_REF,
  SUM(HOM_ALT) AS HOM_ALT,
  SUM(HET) AS HET
  FROM (
  SELECT
    reference_name,
    start_position,
    end_position,
    reference_bases,
    alt.alt,
    -- Conteggio del numero di campioni HOM_REF/HOM_ALT/HET entro ogni variante.
    (SELECT SUM(CAST((SELECT LOGICAL_AND(gt = 0)                      FROM UNNEST(call.genotype) gt) AS INT64)) FROM v.call) AS HOM_REF,
    (SELECT SUM(CAST((SELECT LOGICAL_AND(gt = 1)                      FROM UNNEST(call.genotype) gt) AS INT64)) FROM v.call) AS HOM_ALT,
    (SELECT SUM(CAST((SELECT LOGICAL_OR(gt = 0) AND LOGICAL_OR(gt = 1) FROM UNNEST(call.genotype) gt) AS INT64)) FROM v.call) AS HET
  FROM
    `bigquery-public-data.genomics_rice.Rice3K_DeepVariant_Os_Nipponbare_Reference_IRGSP_1_0` v
    JOIN UNNEST(alternate_bases) AS alt
  WHERE
    --reference_name = 'Chr9' AND start_position = 4691919
    reference_bases IN ('A','C','G','T')
    AND ARRAY_LENGTH(alternate_bases) = 1
    AND alt.alt IN ('A','C','G','T')
  )
  AS x
  GROUP BY reference_name, start_position, end_position, reference_bases, alt
),

```

```
observations AS (
  SELECT
    reference_name,
    start_position,
    reference_bases,
    alt,
    HOM_REF AS OBS_HOM1,
    HET AS OBS_HET,
    HOM_ALT AS OBS_HOM2,
    HOM_REF + HET + HOM_ALT AS SAMPLE_COUNT,
    3024 - (HOM_REF + HET + HOM_ALT) AS MISSING_COUNT
  FROM variants
),

expectations AS (
  SELECT
    reference_name,
    start_position,
    reference_bases,
    alt,
    OBS_HOM1,
    OBS_HET,
    OBS_HOM2,

    # utilizziamo F come coefficiente di consanguinità.
    # per tornare allo standard Hardy-Weiberg F=0, ma dato che noi consideriamo una popolazione di riso autofecondata dobbiamo usare un coefficiente pari a 0.95
    # Expected AA
    # p^2
    # ((COUNT(AA) + (COUNT(Aa)/2) / SAMPLE_COUNT) ^ 2) * SAMPLE_COUNT
    POW((MISSING_COUNT + OBS_HOM1 + OBS_HET/2) / (MISSING_COUNT + SAMPLE_COUNT), 2) * (SAMPLE_COUNT + MISSING_COUNT)
      + 0.95 * ((MISSING_COUNT + OBS_HOM1 + OBS_HET/2) / (MISSING_COUNT + SAMPLE_COUNT)) * ((OBS_HOM2 + OBS_HET/2) / (SAMPLE_COUNT)) * (MISSING_COUNT + SAMPLE_COUNT)
    AS E_HOM1,

    # Expected Aa
    # 2pq
    # 2 * (COUNT(AA) + (COUNT(Aa)/2) / SAMPLE_COUNT) * (COUNT(aa) + (COUNT(Aa)/2) / SAMPLE_COUNT) * SAMPLE_COUNT
    (1.0 - 0.95) * 2
      * ((MISSING_COUNT + OBS_HOM1 + OBS_HET/2) / (MISSING_COUNT + SAMPLE_COUNT))
      * ((OBS_HOM2 + (OBS_HET/2)) / SAMPLE_COUNT) * SAMPLE_COUNT AS E_HET,

    # Expected aa
    # q^2
    # (COUNT(aa) + (COUNT(Aa)/2) / SAMPLE_COUNT) ^ 2 * SAMPLE_COUNT
    POW((OBS_HOM2 + (OBS_HET/2)) / SAMPLE_COUNT, 2) * SAMPLE_COUNT
      + 0.95
        * ((MISSING_COUNT + OBS_HOM1 + OBS_HET/2) / (MISSING_COUNT + SAMPLE_COUNT))
        * ((OBS_HOM2 + (OBS_HET/2)) / SAMPLE_COUNT) * SAMPLE_COUNT
    AS E_HOM2

  FROM observations
  WHERE SAMPLE_COUNT > 0
),

stats AS (
  SELECT
    reference_name,
    start_position,
    reference_bases,
    alt,
    OBS_HOM1,
    OBS_HET,
    OBS_HOM2,
    E_HOM1,
    E_HET,
    E_HOM2,

    # Calcolo di Chi quadro
    # SUM(((Observed - Expected)^2) / Expected )
    ROUND((POW(OBS_HOM1 - E_HOM1, 2) / E_HOM1)
    + (POW(OBS_HET - E_HET, 2) / E_HET)
    + (POW(OBS_HOM2 - E_HOM2, 2) / E_HOM2), 6)
    AS ChiSq,

    # Determino se il valore di Chi quadro è significativo
    # Il punteggio chi quadro corrisponde al valore nominale P-value of 0.05
    # for a table with 2 degrees of freedom is 5.991.
    IF((POW(OBS_HOM1 - E_HOM1, 2) / E_HOM1)
    + (POW(OBS_HET - E_HET, 2) / E_HET)
    + (POW(OBS_HOM2 - E_HOM2, 2) / E_HOM2)
    > 5.991, TRUE, FALSE) AS PVALUE_SIG

  FROM expectations
  WHERE
    E_HOM1 > 0 AND E_HET > 0 AND E_HOM2 > 0
    -- Aggiungo la proposizione opzionale al fine di vincolare i risultati.
)

SELECT pt.ref,pt.bin,pt.n AS t,pf.n AS f,pt.n/(pt.n+pf.n) AS pct_t,pf.n/(pt.n+pf.n) AS pct_f
FROM
(SELECT reference_name AS ref, CAST(start_position/10000 AS INT64) AS bin, COUNT(1) AS n
FROM stats
WHERE PVALUE_SIG IS TRUE
GROUP BY ref, bin
) AS pt,
(SELECT reference_name AS ref, CAST(start_position/10000 AS INT64) AS bin, COUNT(1) AS n
FROM stats
WHERE PVALUE_SIG IS FALSE
GROUP BY ref, bin
) AS pf
WHERE pt.ref = pf.ref AND pt.bin = pf.bin
ORDER BY ref, bin
```

Figura 37 La query in esempio è stata presa dalla ricerca compiuta da Allen Day e Ryan Poplin (appartenenti al Google team), chiamata "Analyzing 3024 rice genomes characterized by DeepVariant" [173], con il fine ultimo di testare le capacità di calcolo di BigQuery in un database di dimensioni maggiori di un Tera . Il codice raffigurato nella seguenza delle quattro immagini sovrastanti rappresenta la query posta per l'applicazione pratica della teoria di Hardy-Weinberg. Dalla figura vediamo come la prima parte della query sia stata elaborata col fine di essegnare dei valori alle varianti che caratterizzano l'equazione di Hardy-Weinberg. Successivamente si hanno le prove per le tre diverse possibili distribuzioni dei due presi in considerazione, p^2, pq e q^2. Infine si ha la verifica della validità del risultato della query, tramite il metodo del Chi quadro, che verifica la validità di un risultato statistico ottenuto se il quadrato della percentuale di errore risultante è minore di 0,05, cioè del 5%.

La teoria di Hardy-Weinberg, su cui si basa la query in Figura 37, postula il mantenimento di un equilibrio nella frequenza dei genotipi osservabili in una popolazione nel susseguirsi delle generazioni, senza considerare gli effetti di mutazioni, selezione o altri fattori esterni in grado di influenzare la distribuzione allelica di base. Stimando le distribuzioni di genomi attese in funzione della legge Hardy-Weinberg nelle popolazioni di riso (opportunamente modificate per un coefficiente di consanguineità dovuto alla tendenza del riso ad auto-fecondarsi), e comparandole con i risultati della query precedentemente proposta, otterremo una serie di porzioni di genoma che si discostano dalle previsioni. Tali porzioni corrispondono alle regioni sottoposte ad alta pressione selettiva e possono essere studiate e analizzate per trarne conclusioni nell'ambito della genetica. Il tempo di elaborazione della query proposta è mostrato nella seguente figura. Queste conclusioni sono state tratte dalla ricerca svolta sul genotipo del riso.

Tempo trascorso	Tempo utilizzato da slot	Byte ordinati in modo sistematico	Byte con overflow su disco
1 min 41 sec	4 giorni 3 h	2,29 TB	0 B

		Tempistica worker					
Fasi		Attesa	Lettura	Calcolo	Scrittura		Righe
S00: Input	Media:	12085 ms	402 ms	28259 ms	402 ms	Input:	12.186.710.727
	Max:	16756 ms	875 ms	38321 ms	8669 ms	Output:	9.622.745.063
S01: Input	Media:	16451 ms	196 ms	27264 ms	331 ms	Input:	12.186.710.727
	Max:	23884 ms	822 ms	39848 ms	12832 ms	Output:	9.622.745.063
S02: Repartition	Media:	6483 ms	0 ms	3196 ms	391 ms	Input:	157.441.783
	Max:	7399 ms	0 ms	4334 ms	5628 ms	Output:	157.441.783
S03: Repartition	Media:	12852 ms	0 ms	3241 ms	129 ms	Input:	155.720.681
	Max:	12980 ms	0 ms	4266 ms	2288 ms	Output:	155.720.681
S04: Repartition	Media:	7389 ms	0 ms	3595 ms	418 ms	Input:	207.525.471
	Max:	7787 ms	0 ms	4225 ms	3842 ms	Output:	207.525.471

Figura 38 L'immagine rappresenta la tempistica di lavoro impiegata da BigQuery per elaborare l'interrogazione mostrata nella Figura 37. Si nota come le fasi che lo strumento ha dovuto compiere siano 21, di cui quelle che hanno richiesto maggior tempo sono state la fase di input iniziale, in cui BigQuery ha dovuto analizzare varie colonne composte da più di 12 miliardi di record, e alcune fasi di ripartizione, necessarie per le sub-query annidate che abbiamo posto. Si nota in oltre quanto sia aumentato il tempo totale trascorso rispetto alle query poste negli esempi precedenti, che ha superato il minuto arrivando a 1 min e 42 secondi. Il numero di byte ordinati per la creazione della tabella risultante è stato 2,29 TB.

Riga	ref	bin	t	f	pct_t	pct_f
1	Chr1	25	1933	6	0.9969056214543579	0.0030943785456420837
2	Chr1	216	2186	1	0.9995427526291724	4.572473708276177E-4
3	Chr1	308	1145	3	0.9973867595818815	0.002613240418118467
4	Chr1	339	2457	1	0.999593165174939	4.068348250610252E-4
5	Chr1	344	2673	1	0.9996260284218399	3.7397157816005983E-4
6	Chr1	687	2007	1	0.9995019920318725	4.9800796812749E-4
7	Chr1	792	2793	3	0.9989270386266095	0.001072961373390558
8	Chr1	1006	2351	4	0.9983014861995754	0.0016985138004246285
9	Chr1	1091	2507	4	0.9984070091596974	0.0015929908403026682
10	Chr1	1120	2552	3	0.998825831702544	0.0011741682974559687
11	Chr1	1242	1958	3	0.998470168281489	0.0015298317185109638
12	Chr1	1270	2264	1	0.9995584988962473	4.415011037527594E-4
13	Chr1	1543	1840	3	0.9983722192078134	0.0016277807921866521
14	Chr1	1641	1710	2	0.9988317757009346	0.0011682242990654205

Figura 39 Rappresentazione visiva del risultato della query posta nella Figura 37. La tabella risultante aveva molte più righe, 542 per l'esattezza. In quella raffigurata ne sono mostrate solo 14 per comodità rappresentativa, al solo scopo di far vedere il risultato ottenuto da una query di Big Data. Notiamo come i numeri rappresentati nel campo pct_f siano spesso composti da 20 cifre.

Applicazione sperimentale per l'integrazione con GIS

Vogliamo integrare il risultato delle analisi dei dati con un risultato visivo. In particolare siamo interessati a provare ad ottenere una rappresentazione delle informazioni, ottenute grazie all'analisi di BigQuery, su una interfaccia proposta dal Google Cloud, chiamata BigQuery Geo Viz. Quest'ultima utilizza le API di Google Maps per tracciare geopunti in tutto il mondo. Per far ciò abbiamo intenzione di analizzare alcuni database forniti da Google, tramite query create appositamente per questo scopo. Una volta ottenuti i dati si vuole vedere come il servizio di BigQuery si integra con quello di BigQuery GIS, la cui funzione è quella di convertire le colonne di latitudine e longitudine in punti geografici, che dovranno essere poi visualizzabili in BigQuery Geo Viz. Per far ciò riprendiamo in mano il database "Wikipedia" e andiamo a vedere come BigQuery possa fornire risultati raffigurati su una mappa geografica. Tra i campi del database considerato, oltre a "title" e "contributor_username", utilizzati nella query mostrata in Figura 32, vi è anche un altro campo chiamato "contributor_ip", che indica l'Internet Protocol address, collegabile in forma anonima alla posizione del contributore. Vogliamo quindi utilizzare questo campo per vedere quali sono le aree geografiche che raccolgono il maggior numero di contributori, stavolta a tutte le voci di Wikipedia (non solo a quelle con "Italy" come nel caso precedente). Per ottenere dall'indirizzo IP informazioni sulla posizione geografica sarà necessario sfruttare un'altra tabella proposta da BigQuery tra i data set messi a disposizione sulla piattaforma online. Il database in questione, mostrato in Figura 40, si chiama "ipv4_city_locations", ed è situato nel data set "geolite2".

Riga	geoname_id	locale_code	continent_code	continent_name	country_iso_code	country_name	subdivision_1_iso_code
1	49518	es	AF	África	RW	Ruanda	*null*
2	51537	es	AF	África	SO	Somalia	*null*
3	69543	es	AS	Asia	YE	Yemen	*null*
4	99237	es	AS	Asia	IQ	Irak	*null*
5	102358	es	AS	Asia	SA	Arabia Saudí	*null*
6	130758	es	AS	Asia	IR	Irán	*null*
7	146669	es	EU	Europa	CY	Chipre	*null*
8	149402	es	AF	África	TZ	Tanzania	*null*
9	149590	es	AF	África	TZ	Tanzania	*null*
10	163843	es	AS	Asia	SY	Siria	*null*
11	174982	es	AS	Asia	AM	Armenia	*null*
12	192950	es	AF	África	KE	Kenia	*null*
13	203312	es	AF	África	CD	Congo Democrático	*null*

Figura 40 La tabella in figura mostra una porzione del database "ipv4_city_locations", così da avere un'idea concreta di quali informazioni andiamo ad unire alla tabella "wikipedia" nella seguente query. Notiamo come questo database rappresenti una specifica tabella per le localizzazioni geografiche, dove ogni località è caratterizzata da varie proprietà, tra cui un codice identificativo, indicato nel campo "geoname_id".

Per unire le informazioni ottenute tramite la query sul database "wikipedia" con quelle ottenute dal database "ipv4_city_locations", utilizzeremo il comando SQL standard "JOIN".

```sql
WITH source_of_ip_addresses AS (
  SELECT REGEXP_REPLACE(contributor_ip, 'xxx', '0')  ip, COUNT(*) c
  FROM `publicdata.samples.wikipedia`
  WHERE contributor_ip IS NOT null
  GROUP BY 1
)

SELECT country_name, SUM(c) c
FROM (
  SELECT ip, country_name, c
  FROM (
    SELECT *, NET.SAFE_IP_FROM_STRING(ip) & NET.IP_NET_MASK(4, mask) network_bin
    FROM source_of_ip_addresses, UNNEST(GENERATE_ARRAY(9,32)) mask
    WHERE BYTE_LENGTH(NET.SAFE_IP_FROM_STRING(ip)) = 4
  )
  JOIN `fh-bigquery.geocode.201806_geolite2_city_ipv4_locs`
  USING (network_bin, mask)
)
GROUP BY 1
ORDER BY 2 DESC
```

Riga	country_name	c
1	United States	40289843
2	United Kingdom	10111247
3	Canada	5012007
4	Australia	3403443
5	India	1675270
6	Germany	1418025
7	Philippines	766862
8	Netherlands	669967
9	Ireland	651958
10	France	608999
11	New Zealand	598979
12	Brazil	567661

Figura 41 A sinistra dell'immagine è mostrato il codice SQL per compiere la query. Si nota come si siano resi necessari vari comandi non utilizzati fino a questo momento. Nel primo frammento di codice in particolare è stato utilizzato REGEXP_REPLACE per modificare la stringa "contributor_ip" in 3 variabili seguite da uno 0. Inoltre, proprio per la funzione di geolocalizzazione svolta da questa query, si vede come è stato inserita la funzione "NET", funzioni di byte e di rete migliorate appositamente da BigQuery per la conversione in dati geografici. A destra si nota il risultato ottenuto dalla query, che mostra il nome delle città con accanto il numero di "contributori" alle voci di Wikipedia, ordinato in ordine decrescente. La query è stata elaborata in meno di 15 secondi ed ha prodotto la tabella mostrata sulla destra, contenente 256 record, che nell'immagine è stata tagliata per questioni di comodità.

Vogliamo adesso visualizzare questi risultati nella mappa geografica messa a disposizione dal servizio di BigQuery Geo Viz. Per far ciò occorre ottenere delle coordinate geografiche, in particolare una colonna che chiamiamo “point” in cui inseriamo le precise coordinate di latitudine e longitudine in riferimento alle diverse località. Per fare questa operazione è stata creata una query simile alla precedente, raffigurata nella seguente figura.

```sql
WITH source_of_ip_addresses AS (
  SELECT REGEXP_REPLACE(contributor_ip, 'xxx', '0')  ip, COUNT(*) c
  FROM `publicdata.samples.wikipedia`
  WHERE contributor_ip IS NOT null
  GROUP BY 1
)


SELECT city_name, SUM(c) c, ST_GeogPoint(AVG(longitude), AVG(latitude)) point
FROM (
  SELECT ip, city_name, c, latitude, longitude, geoname_id
  FROM (
    SELECT *, NET.SAFE_IP_FROM_STRING(ip) & NET.IP_NET_MASK(4, mask) network_bin
    FROM source_of_ip_addresses, UNNEST(GENERATE_ARRAY(9,32)) mask
    WHERE BYTE_LENGTH(NET.SAFE_IP_FROM_STRING(ip)) = 4
  )
  JOIN `fh-bigquery.geocode.201806_geolite2_city_ipv4_locs`
  USING (network_bin, mask)
)
WHERE city_name  IS NOT null
GROUP BY city_name, geoname_id
ORDER BY c DESC
LIMIT 5000
```

Figura 42 L’immagine mostra la query in linguaggio standard SQL, posta tramite BigQuery ai due database “wikipedia” e “ipv4_city_locations”, visualizzabile in Figura 40. Con questa query si vuole creare un campo contenente le localizzazioni geografiche, prese da ST_GeogPoint, corrispondenti ai valori trovati nella query precedente in Figura 41.

Riga	city_name	c	point
1	New York	620153	POINT(-73.9774716883187 40.7649325291159)
2	Seattle	496045	POINT(-122.316075640286 47.5747049185338)
3	Toronto	461451	POINT(-79.4157699490502 43.6818338707703)
4	Brooklyn	365231	POINT(-73.9530768746045 40.6598734576039)
5	Chicago	359766	POINT(-87.6670091484614 41.8841176087964)
6	Singapore	310970	POINT(103.8558 1.2931)
7	Los Angeles	294791	POINT(-118.31400152413 34.0424667212694)
8	London	264715	POINT(-0.0994497370164024 51.5147004246698)
9	Philadelphia	250254	POINT(-75.151949423869 39.9918175168147)
10	San Jose	220753	POINT(-121.893028985091 37.3468602239403)
11	Atlanta	213687	POINT(-84.372594521586 33.8210334500696)
12	Washington	211046	POINT(-77.0306693476913 38.9111268569329)
13	Auckland	208737	POINT(174.760975229358 -36.8686598964854)
14	Portland	205145	POINT(-122.654843583892 45.5114857132979)

Figura 43 Nell’immagine mostrata vediamo come la query compilata sia riuscita con successo. Difatti siamo riusciti ad ottenere una tabella risultante composta dalla colonna “point” contenente le varie coordinate geografiche.

Una volta aver constato che la query compila, e dopo aver ottenuto i risultati di localizzazione grafica che cercavamo, proviamo ad aprire BigQuery Geo Viz e ad inserire la query rappresentata in Figura 42. L'interfaccia proposta da Google per questo servizio risulta chiara, difatti vi è una sorta di tutorial che guida alle operazioni da fare, affiancato ad un planisfero diviso da una linea rappresentante l'equatore. La prima operazione richiesta è quella di inserire la query. Quindi copiamo l'interrogazione SQL compilata precedentemente su BigQuery e vediamo che cosa succede.

city_name	c	point
New York	620153	{ "type": "Point", "coord...
Seattle	496045	{ "type": "Point", "coord...
Toronto	461451	{ "type": "Point", "coord...
Brooklyn	365231	{ "type": "Point", "coord...
Chicago	359766	{ "type": "Point", "coord...
Singapore	310970	{ "type": "Point", "coord...
Los Angeles	294791	{ "type": "Point", "coord...
London	264715	{ "type": "Point", "coord...
Philadelphia	250254	{ "type": "Point", "coord...
San Jose	220753	{ "type": "Point", "coord...

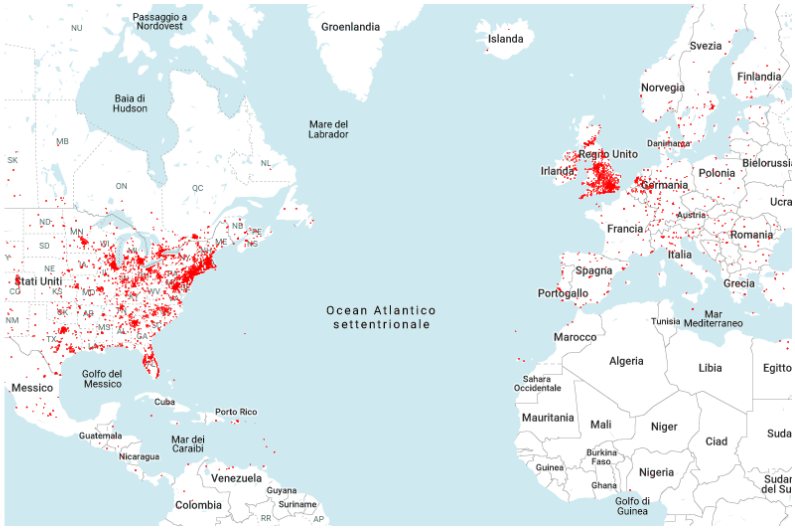

Figura 44 L'immagine raffigurata mostra il risultato della query posta su BigQuery Geo Viz. Quella in figura è infatti l'interfaccia proposta dal servizio. A sinistra vi è lo spazio per scrivere ed eseguire la query, oltre che per modificare la rappresentazione dei risultati sulla mappa. A destra invece vediamo il planisfero che abbiamo citato precedentemente. La figura in particolare rappresenta il risultato della query rappresentata in Figura 42, ottenuto in meno di 20 secondi. A sinistra si nota infatti la stessa tabella di Figura 43, mentre a destra la localizzazione geografica delle prime 5000 città con il maggior numero di "contributori" per le voci di Wikipedia, che in realtà sono gli stessi dati della tabella visualizzati sulla mappa.

Vogliamo adesso fare un ultimo test della visualizzazione geografica di BigQuery Geo Viz. Stavolta siamo interessati a visualizzare non solo il riferimento geografico dei dati, ma anche la loro distribuzione nel tempo. Ci chiediamo se BigQuery sia in grado di rispondere a queste diverse esigenze. Per testare queste funzioni prendiamo in considerazione un fenomeno atmosferico che è decsrivibile sulla mappa geografica in funzione del tempo: l'uragano. Tra i data set che Google mette a disposizione ne abbiamo trovato uno chiamato proprio "noaa_hurricanes". NOAA è una sigla che indica "National Oceanic and Atmospheric Administration", l'agenzia federale statunitense che si occupa proprio dei fenomeni atmosferici, tra cui gli uragani. Il database che abbiamo scelto da analizzare si chiama proprio "hurricanes" ed è una tabella composta da ben più di 100 campi e da 681.199 record, per uno spazio di archiviazione risultante di 162,26 MB. Il numero elevato di colonne è giustificato dal fatto che all'interno di questo file sono riportate le coordinate degli uragani, le distanze di questi da più città diverse, la distanza da terra, le pressioni e le forze dei venti generati in

molteplici città, tra cui New York e Tokyo. Inoltre, per ogni uragano vi sono più record; i cui valori contenuti nei campi rappresentanti le coordinate spaziali e temporali variano, mentre non cambiano i valori nei campi contenenti le informazioni identificative dell'uragano. Questo significa che per ogni catastrofe atmosferica sono riportate più informazioni variabili nel corso del tempo di propagazione dell'uragano. Ci chiediamo lo sviluppo ed il percorso che ha compiuto l'uragano "Katrina" del 2005, con l'obiettivo di visualizzarlo tramite il servizio di BigQuery Geo Viz. Andiamo allora ad eseguire la query. Considerato che il database contiene già le informazioni per la localizzazione geografica dell'uragano, non ci sarà bisogno di unire i risultati della query con quelli del data set "geolite2", anzi faremo in modo tale di ottenere una colonna "point" con le coordinate di latitudine e longitudine.

```
SELECT
  ST_GeogPoint(longitude, latitude) AS point,
  name,
  iso_time,
  dist2land,
  usa_wind,
  usa_pressure
FROM
  `bigquery-public-data.noaa_hurricanes.hurricanes`
WHERE
  name LIKE '%KATRINA%'
  AND season = '2005'
ORDER BY
  iso_time ASC
```

Figura 45 L'immagine mostrata fa vedere direttamente l'interfaccia di BigQuery Geo Viz. A sinistra vediamo raffigurato il codice necessario per selezionare dal database "hurricanes" l'uragano Katrina che siamo interessati a visualizzare. Vediamo infatti come le informazioni della query corrispondano. Notiamo inoltre una funzione non ancora vista in questa tesi, ST_GeogPoint, necessaria per convertire le colonne "longitude" e "latitude" del database, in informazioni geografiche (points), così da ottenere la colonna richiesta dal BigQuery Geo Viz per la rappresentazione sulla mappa geografica del risultato della query. A destra è invece raffigurata la rappresentazione grafica dell'uragano Katrina

Conclusioni

Possiamo concludere, dopo aver esposto una panoramica riguardo agli argomenti relativi a BigQuery, di aver mostrato il completo funzionamento e utilizzo dello strumento. Abbiamo voluto ampliare il concetto basilare di analisi di dati su cui si basa BigQuery e le possibili applicazioni, in particolare in ambito medico. Partendo dallo sviluppo di Internet e dalla creazione dei dati sul Web, siamo arrivati alla nascita dei Big Data, dei quali abbiamo visto i possibili utilizzi e le potenzialità legate alla loro analisi. La ricerca condotta in questo studio mette in evidenza il ruolo chiave del servizio fornito da BigQuery in tale processo. Abbiamo mostrato le varie caratteristiche del Cloud Computing e le sue diverse applicazioni, così da permettere di capire la possibile integrazione dello strumento di analisi. Con la serie di test effettuati, abbiamo voluto valutare il funzionamento di BigQuery. In primo luogo, è stata verificata l'accessibilità dello strumento di analisi proposto da Google Cloud. Abbiamo illustrato le varie operazioni da eseguire, concentrandoci sui passaggi più importanti. I risultati hanno confermato le aspettative circa la semplicità di utilizzo, poichè il procedimento si è dimostrato lineare e privo di imprevisti. Ovviamente non si possono trarre conclusioni concrete sulla semplicità di uno strumento testato da una sola persona, ma si può affermare che le conoscenze base del linguaggio SQL sono sufficienti per sfruttare questo strumento. Ciò è permesso dall'interfaccia proposta da Google per l'utilizzo di questo software, che consente l'utilizzo di uno strumento complesso come BigQuery anche a utenti privi di solide competenze nell'area del linguaggio computazionale. Riguardo alle potenzialità di BigQuery, abbiamo fornito un quadro generale e spiegato come le applicazioni di un così potente strumento di analisi possano essere molteplici e arrivino a coinvolgere più settori, da quello sanitario a quello economico. Nella seconda parte abbiamo invece testato le vere e proprie capacità di calcolo del data warehouse aziendale di Google. L'obiettivo era proprio valutare le variazioni nei tempi impiegati per l'analisi delle tabelle, ponendo a BigQuery database crescenti scalarmente. Abbiamo messo alla prova il meccanismo sfruttato da Dremel di cui abbiamo parlato negli ultimi capitoli. Riguardo all'analisi portata a termine circa le prestazioni di BigQuery, possiamo dire che lo strumento di calcolo ha superato le aspettative, riuscendo a gestire anche i carichi di lavoro più grandi in modo estremamente rapido. La velocità con cui i compiti vengono portati a termine è particolarmente impressionante se ci si sofferma a valutare le enormi dimensioni dei database analizzati, tanto da rendere il Cloud Computing proposto da Google competitivo con gli altri Cloud Leader in circolazione. Infatti, le prove effettuate hanno mostrato come BigQuery riesca ad elaborare 1.5 TB di dati genomici in 101 secondi. Tale velocità è resa possibile non solo dalla potenza di calcolo di Dremel, ma anche dalle ottime prestazioni di compressione e disposizione dei database elaborati.
Infine abbiamo testato il provider integrato con BigQuery GIS, per valutare le capacità di visualizzazione geografica dei dati ottenuti. Il servizio offerto da Google, integrato recentemente con

Google BigQuery Geo Viz, è risultato interessante, ma sicuramente questa funzionalità ci ha stupito meno della potenza di calcolo fornita dallo strumento vero e proprio. Vogliamo inoltre mettere in evidenza come la combinazione di semplicità e capacità di analisi consenta di aumentare esponenzialmente l'efficienza di BigQuery. Infatti, se la grande capacità di calcolo rappresenta già di per se l'enorme velocità di scanning e analisi dello strumento, la sua semplicità lo rende accessibile a un numero di utenti molto maggiore rispetto ad altri software similari; in questo modo non soltanto si ottiene uno strumento potente, ma si incrementa anche il numero di operatori che possono interagire con esso, garantendo un'enorme incremento della produttività. Inoltre, si ricorda che per compiere i test svolti in questa tesi è stato utilizzato l'account gratuito in prova. Possiamo concludere che BigQuery, il servizio di "data analysis" proposto da Google, è un eccellente strumento di analisi, con grandi possibilità di sviluppo nel prossimo futuro.

Ringraziamenti

Lo svolgimento di questa tesi è stato sicuramente un percorso che mi ha personalmente fatto crescere ed avvicinare alla materia presa in considerazione.
La sua realizzazione non sarebbe stata possibile senza l'aiuto ed il sostegno di molte persone che hanno deciso di affiancarmi in questo periodo del mio percorso universitario.
Ringrazio la gentilissima Professoressa Cristiana Corsi, che mi ha fornito il DBSANITA.csv e che si è mostrata disponibile, gentile e pronta a dare preziosi consigli.
Ringrazio il mio relatore, il Professore Luca Roffia, e il correlatore, l'Ing. Cristiano Aguzzi, per la fiducia che hanno avuto in me, per il sostegno, per la disponibilità e per le brillanti idee che hanno contribuito a rendere la tesi più interessante.
Infine ringrazio il mio coinquilino Giacomo e la mia famiglia che mi hanno aiutato e sostenuto in questo lavoro.

BIBLIOGRAFIA

[1] K. Mossberger, C. J. Tolbert e R. S. McNeal, Digital citizenship: The Internet, society, and participation, MIT Press, 2007.

[2] T. Berners-Lee, R. Cailliau, J. Groff e B. Pollermann, «World-Wide Web: The Information Universe,» *Internet Research,* vol. 2, n. 1, pp. 52-58, 1992.

[3] «World Wide Web Timeline,» 11 03 2014. [Online]. Available: https://www.pewinternet.org/2014/03/11/world-wide-web-timeline/. [Consultato il giorno 13 luglio 2019].

[4] D. A. Ferguson e M. P. Elizabeth, «The World Wide Web as a functional alternative to television,» *Journal of broadcasting & electronic media,* vol. 44, n. 2, pp. 155-174, 2000.

[5] S. N. Dorogovtsev e J. F. Mendes, Evolution of networks: From biological nets to the Internet and WWW, OUP Oxford, 2013.

[6] D. Raggett, A. Le Hors e I. Jacobs, «HTML 4.01 Specification,» W3C recommendation, 1999.

[7] R. Albert, H. Jeong e A.-L. Barabási, «Internet: Diameter of the world-wide web,» *nature,* vol. 401, n. 6749, p. 130, 1999.

[8] A. Bernstein e J. Hendler, «A new look of the Semantic Web,» *Communications of the ACM,* vol. 59, n. 9, pp. 35-37, 2016.

[9] J. Gubbi, R. Buyya, S. Marusic e M. Palaniswami, «Internet of Things (IoT): A vision, architectural elements, and future directions,» *Future Generation Computer Systems,* vol. 29, n. 7, pp. 1645-1660, 2013.

[10] A. Zanella, N. Bui, A. Castellani, L. Vangelista e M. Zorzi, «Internet of things for smart cities,» *IEEE Internet of Things journal,* vol. 1, n. 1, pp. 22-32, 2014.

[11] «Internet of Things Global Standards Initiative,» [Online]. Available: https://www.itu.int/en/ITU-T/gsi/iot/Pages/default.aspx. [Consultato il giorno 13 luglio 2019].

[12] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari e M. Ayyash, «Internet of things: A survey on enabling technologies, protocols, and applications.,» *IEEE communications surveys & tutorials* , vol. 17, n. 4, pp. 2347-2376, 2015.

[13] L. Atzori, A. Iera e G. Morabito, «The Internet of Things: A survey,» *Computer Networks,* vol. 54, n. 15, pp. 2787-2805, 2010.

[14] A. Ometov, S. V. Bezzateev, J. Kannisto, J. Harju, S. Andreev e Y. Koucheryavy, «Facilitating the delegation of use for private devices in the era of the internet of wearable things.,» *IEEE Internet of Things Journal*, vol. 4, n. 4, pp. 843-854, 2016.

[15] D. Miorandi, S. Sicari, F. De Pellegrini e I. Chlamtac, «Internet of things: Vision, applications and research challenges,» *Ad Hoc Networks*, vol. 10, n. 7, pp. 1497-1516, 2012.

[16] C. A. da Costa, C. F. Pasluosta, B. Eskofier, D. B. da Silva e R. da Rosa Righi, «Internet of Health Things: Toward intelligent vital signs monitoring in hospital wards.,» *Artificial intelligence in medicine*, vol. 89, pp. 61-69, 2018.

[17] L. J. Kricka, «History of disruptions in laboratory medicine: what have we learned from predictions?,» *Clinical Chemistry and Laboratory Medicine (CCLM)*, vol. 57, n. 3, pp. 308-311, 2019.

[18] A. Gatouillat, Y. Badr, B. Massot e E. Sejdić, «Internet of Medical Things: A Review of Recent Contributions Dealing With Cyber-Physical Systems in Medicine,» *IEEE Internet of Things Journal*, vol. 5, n. 5, pp. 3810-3822, 2018.

[19] E. Topol, The Patient Will See You Now: The Future of Medicine Is in Your Hands, Basic Books, 2015.

[20] N. Dey, A. E. Hassanien, C. Bhatt, A. S. Ashour e S. C. Satapathy, Internet of things and big data analytics toward next-generation intelligence, Berlin: Springer, 2018.

[21] G. J. Joyia, R. M. Liaqat, A. Farooq e S. Rehman, «Internet of Medical Things (IOMT): applications, benefits and future challenges in healthcare domain.,» *Journal of Communications*, vol. 12, n. 4, pp. 240-247, 2017.

[22] P. Mildenberger, M. Eichelberg e E. Martin, «Introduction to the DICOM standard.,» *European Radiology*, vol. 12, n. 4, pp. 920-927, 2002.

[23] R. H. Choplin, J. Boehme e C. Maynard, «Picture archiving and communication systems: an overview.,» *Radiographics*, vol. 12, n. 1, pp. 127-129, 1992.

[24] R. Haux, E. Ammenwerth, A. Winter e B. Brigl, Strategic information management in hospitals: an introduction to hospital information systems, Springer Science & Business Media, 2004.

[25] J. J. Rodrigues, Health information systems: concepts, methodologies, tools, and applications: concepts, methodologies, tools, and applications. Vol. 1, Igi Global, 2009.

[26] D. V. Dimitrov, «Medical internet of things and big data in healthcare,» *Healthcare informatics research*, vol. 22, n. 3, pp. 156-163, 2016.

[27] W. H. Organization., Telemedicine: opportunities and developments in member states. Report on the second global survey on eHealth., World Health Organization, 2010.

[28] N. Ridgers, M. A. McNarry e K. A. Mackintosh, «Feasibility and effectiveness of using wearable activity trackers in youth: a systematic review.,» *JMIR mHealth and uHealth,* vol. 4, n. 4, p. e129, 2016.

[29] A. Rossi Mori, R. Maceratini e F. Ricci, «La cartella clinica elettronica (Electronic Patient Record).,» in *Il medico on-line. Manuale di informatica medica*, Roma, Verduci, 2009.

[30] F. Stradolini, N. Tamburrano, T. Modoux, A. Tuoheti, D. Demarchi e S. Carrara, «IoT for telemedicine practices enabled by an Android™ application with cloud system integration.,» *2018 IEEE international symposium on circuits and systems (ISCAS),* pp. 1-5, 2018.

[31] A. Holzinger, C. Röcker e M. Ziefle, «From smart health to smart hospitals.,» in *Smart Health*, Springer, Cham, 2015, pp. 1-20.

[32] E. Borelli, G. Paolini, F. Antoniazzi, M. Barbiroli, F. Benassi, F. Chesani, L. Chiari, M. Fantini, F. Fuschini, A. Galassi, G. A. Giacobone, S. Imbesi, M. Licciardello, D. Loreti, M. Marchi, D. Masotti, P. Mello, S. Mellone, S. Mincolelli, C. Raffaelli, L. Roffia, T. Salmon Cinotti, C. Tacconi, C. Tamburini, M. Zoli e A. Costanzo, «HABITAT: An IoT Solution for Independent Elderly,» *Sensors*, vol. 19, n. 5, p. 1258, 2019.

[33] R. Want, «An introduction to RFID technology.,» *IEEE pervasive computing,* vol. 1, pp. 25-33, 2006.

[34] S. Legg e M. Hutter, «A collection of definitions of intelligence.,» *Frontiers in Artificial Intelligence and applications*, vol. 157, p. 17, 2007.

[35] S. Russell e P. Norvig, Artificial intelligence: a modern approach., Malaysia: Pearson Education Limited, 2016.

[36] N. J. Nilsson, Principles of artificial intelligence, Morgan Kaufmann, 2014.

[37] D. E. O'Leary, «Artificial intelligence and big data.,» *IEEE Intelligent Systems,* vol. 28, n. 2, pp. 96-99, 2013.

[38] W. G. Mangold e D. J. Faulds, «Social media: The new hybrid element of the promotion mix.,» *Business Horizons,* vol. 52, n. 4, pp. 357-365, 2009.

[39] «So you think you chose to read this article?,» [Online]. Available: https://www.bbc.com/news/business-36837824. [Consultato il giorno 13 luglio 2019].

[40] C. M. Bishop, Pattern Recognition and Machine Learning, Springer, 2006.

[41] L. P. Kaelbling, M. L. Littman e A. W. Moore, «Reinforcement Learning: A Survey,» *Journal of artificial intelligence research*, vol. 4, pp. 237-285, 1996.

[42] M. I. Jordan e T. M. Mitchell, «Machine learning: Trends, perspectives, and prospects.,» *Science*, vol. 349, n. 6245, pp. 255-260, 2015.

[43] J. C. Eichstaedt, R. J. Smith, R. M. Merchant, L. H. Ungar, P. Crutchley, D. Preoţiuc-Pietro, D. A. Asch e A. Schwartz, «Facebook language predicts depression in medical records,» *Proceedings of the National Academy of Sciences*, vol. 115, n. 44, pp. 11203-11208, 2018.

[44] «Digital Therapeutics Have Huge Promise And They Are Real Today,» [Online]. Available: https://www.forbes.com/sites/toddhixon/2015/12/09/digital-therapeutics-have-huge-promise-and-they-are-real-today/#5961681526f0. [Consultato il giorno 2019 luglio 13].

[45] N. Dedić e C. Stanier, «Measuring the Success of Changes to Existing Business Intelligence Solutions to Improve Business Intelligence Reporting,» in *International Conference on Research and Practical Issues of Enterprise Information Systems*, 2016.

[46] O. P. Rud, Business Intelligence Success Factors: Tools for Aligning Your Business in the Global Economy. Vol. 18, John Wiley & Sons, 2009.

[47] G. A. Miller, «WordNet: a lexical database for English,» *Communications of the ACM*, vol. 38, n. 11, pp. 39-41, 1995.

[48] R. L. Daft, J. Murphy e H. Willmott, Organization theory and design., Cengage learning EMEA, 2010.

[49] A. McAfee, E. Brynjolfsson, T. Davenport, D. J. Patil e D. Barton, «Big data: the management revolution,» *Harvard business review*, vol. 90, n. 10, pp. 60-68, 2012.

[50] P. P.-S. Chen, «The Entity-Relationship Model - Toward a Unified View of Data,» *ACM Transactions on Database Systems (TODS)*, vol. 1, n. 1, pp. 9-36, 1976.

[51] «NoSQL Database,» [Online]. Available: http://nosql-database.org/. [Consultato il giorno 2019 luglio 13].

[52] N. Leavitt, «Will NoSQL databases live up to their promise?,» *Computer*, vol. 43, n. 2, pp. 12-14, 2010.

[53] J. Han, G. L. Haihong e J. Du, «Survey on NoSQL database,» in *2011 6th international conference on pervasive computing and applications*, 2011.

[54] J. Pokorny, «NoSQL databases: a step to database scalability in web environment,» *International Journal of Web Information Systems*, vol. 9, n. 1, pp. 69-82, 2013.

[55] P. J. Sadalage e M. Fowler, NoSQL distilled: a brief guide to the emerging world of polyglot persistence, Pearson Education, 2013.

[56] J. Schindler, «Profiling and analyzing the I/O performance of NoSQL DBs,» *ACM SIGMETRICS Performance Evaluation Review*, vol. 41, n. 1, pp. 389-390, 2013.

[57] E. Brewer, «A certain freedom: thoughts on the CAP theorem,» in *Proceedings of the 29th ACM SIGACT-SIGOPS symposium on Principles of distributed computing.*, 2010.

[58] S. Gilbert e N. Lynch, «Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services,» *Acm Sigact News*, vol. 33, n. 2, pp. 51-59, 2002.

[59] A. Moniruzzaman e S. A. Hossain, «Nosql database: New era of databases for big data analytics-classification, characteristics and comparison.,» *International Journal of Database Theory and Application* , vol. 6, n. 4, 2013.

[60] «Apache Cassandra,» [Online]. Available: http://cassandra.apache.org/. [Consultato il giorno 13 luglio 2019].

[61] M. Seeger e S. Ultra-Large-Sites, «Key-Value stores: a practical overview.,» Computer Science and Media, Stuttgart, 2009.

[62] «Project Voldemort,» [Online]. Available: https://www.project-voldemort.com/voldemort/. [Consultato il giorno 13 luglio 2019].

[63] «BSON,» [Online]. Available: http://bsonspec.org. [Consultato il giorno 13 luglio 2019].

[64] «MongoDB,» [Online]. Available: https://www.mongodb.com/. [Consultato il giorno 13 luglio 2019].

[65] «Titan,» [Online]. Available: https://titan.thinkaurelius.com/. [Consultato il giorno 13 luglio 2019].

[66] N. G. Bourbakis, Artificial Intelligence and Automation, World Scientific, 1998.

[67] B.-H. Yoon, S.-K. Kim e S.-Y. Kim, «Use of Graph Database for the Integration of Heterogeneous Biological Data,» *Genomics & informatics*, vol. 15, n. 1, p. 19, 2017.

[68] T. Berners-Lee, J. Hendler e O. Lassila, «The Semantic Web,» *Scientific American*, vol. 284, n. 5, pp. 28-37, 2001.

[69] «RDF 1.1 Concepts and Abstract Syntax,» [Online]. Available: https://www.w3.org/TR/turtle/. [Consultato il giorno 13 luglio 2019].

[70] «RDF 1.1 Turtle,» [Online]. Available: https://www.w3.org/TR/turtle/. [Consultato il giorno 13 luglio 2019].

[71] «RDF 1.1 XML Syntax,» [Online]. Available: https://www.w3.org/TR/rdf-syntax-grammar/. [Consultato il giorno 13 luglio 2019].

[72] «JSON-LD 1.0,» [Online]. Available: https://www.w3.org/TR/json-ld/. [Consultato il giorno 13 luglio 2019].

[73] «RDF Schema 1.1,» [Online]. Available: https://www.w3.org/TR/rdf-schema/. [Consultato il giorno 13 luglio 2019].

[74] R. Cattell, «Scalable SQL and NoSQL data stores,» *Acm Sigmod Record*, vol. 39, n. 4, pp. 12-27, 2011.

[75] J. Pérez, M. Arenas e C. Gutierrez, «Semantics and complexity of SPARQL,» in *International semantic web conference.*, Berlin, Heidelberg, 2006.

[76] H. Wang, W. He e F.-K. Wang, «Enterprise cloud service architectures,» *Information Technology and Management,* vol. 13, n. 4, pp. 445-454, 2012.

[77] P. Mell e T. Grance, «The NIST definition of cloud computing,» 2011.

[78] «Google Drive,» [Online]. Available: https://www.google.com/drive/. [Consultato il giorno 13 luglio 2019].

[79] «ICloud,» [Online]. Available: https://www.icloud.com/. [Consultato il giorno 13 luglio 2019].

[80] «Microsoft Azure,» [Online]. Available: https://azure.microsoft.com/it-it/. [Consultato il giorno 13 luglio 2019].

[81] S. Ostermann, A. Iosup, N. Yigitbasi, R. Prodan, T. Fahringer e D. Epema, «A performance analysis of EC2 cloud computing services for scientific computing,» in *International Conference on Cloud Computing*, Berlin, Heidelberg, 2009.

[82] «Open Nebula,» [Online]. Available: https://opennebula.org/. [Consultato il giorno 13 luglio 2019].

[83] O. Sefraoui, M. Aissaoui e M. Eleuldj, «OpenStack: toward an open-source solution for cloud computing.,» *International Journal of Computer Applications,* vol. 55, n. 3, pp. 38-42, 2012.

[84] «Amazon AWS,» [Online]. Available: https://aws.amazon.com/it/. [Consultato il giorno 13 luglio 2019].

[85] B. Calder, J. Wang, A. Ogus, N. Nilakantan, A. Skjolsvold, S. McKelvie, Y. Xu e e. al., «Windows Azure Storage: a highly available cloud storage service with strong consistency,» in *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles.*, 2011.

[86] A. Tiwari, «How Cloud Is Playing An Important Role In Medicine,» [Online]. Available: http://www.businessworld.in/article/How-Cloud-Is-Playing-An-Important-Role-In-Medicine-/07-04-2019-168878/. [Consultato il giorno 13 luglio 2019].

[87] R. Daman, M. M. Tripathi e S. K. Mishra, «Cloud Computing for Medical Applications & Healthcare Delivery: Technology, Application, Security and Swot Analysis.,» in *ACEIT Conference Proceeding 2016*, 2016.

[88] Z. Jin e Y. Chen, «Telemedicine in the cloud era: Prospects and challenges.,» *IEEE Pervasive Computing*, vol. 14, n. 1, pp. 54-61, 2015.

[89] «CookieBot,» [Online]. Available: https://www.cookiebot.com/media/1136/cookiebot-report-2019-ad-tech-surveillance-2.pdf. [Consultato il giorno 13 luglio 2019].

[90] «HIPAA,» [Online]. Available: https://www.hhs.gov/hipaa/for-professionals/security/laws-regulations/index.html. [Consultato il giorno 13 luglio 2019].

[91] «HITRUST,» [Online]. Available: https://hitrustalliance.net/. [Consultato il giorno 13 luglio 2019].

[92] «Google Inc. - Other Services,» [Online]. Available: https://www.britannica.com/topic/Google-Inc/Other-services. [Consultato il giorno 13 luglio 2019].

[93] D. Aberdeen, A. Slater e O. Pacovsky, «The Learning Behind Gmail Priority Inbox,» in *LCCC : NIPS 2010 Workshop on Learning on Cores, Clusters and Clouds*, 2010.

[94] «The Forrester Wave™ Methodology Guide,» [Online]. Available: https://go.forrester.com/policies/forrester-wave-methodology/. [Consultato il giorno 13 luglio 2019].

[95] «Calico,» [Online]. Available: https://www.calicolabs.com/news/2019/06/03/. [Consultato il giorno 13 luglio 2019].

[96] L. A. Barroso e U. Hölzle, «The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines,» *Synthesis lectures on computer architecture*, vol. 4, n. 1, pp. 1-108, 2009.

[97] «Gartner,» [Online]. Available: https://www.gartner.com/en. [Consultato il giorno 13 luglio 2019].

[98] «BigQuery riconosciuta Leader nel Magic Quadrant di Gartner,» [Online]. Available: https://cloud.google.com/gartner-magic-quadrant-for-dmsa/?hl=it. [Consultato il giorno 13 luglio 2019].

[99] «Google Cloud riconosciuto Leader nell'ambito dei data warehouse,» [Online]. Available: https://cloud.google.com/forrester-cloud-data-warehouse/. [Consultato il giorno 13 luglio 2019].

[100] «Critical Capabilities for Data Management Solutions for Analytics,» [Online]. Available: https://www.gartner.com/doc/reprints?id=1-6KF3VTY&ct=190424&st=sb. [Consultato il giorno 13 luglio 2019].

[101] «New Google cloud boss Thomas Kurian warns, 'You will see us competing much more aggressively',» [Online]. Available: https://www.cnbc.com/2019/02/12/google-cloud-ceo-thomas-kurian-google-will-compete-more-aggressively.html. [Consultato il giorno 13 luglio 2019].

[102] «Google Storage,» [Online]. Available: https://cloud.google.com/storage/. [Consultato il giorno 13 luglio 2019].

[103] «BigQuery Data Transfer Service,» [Online]. Available: https://cloud.google.com/bigquery/docs/dts?hl=it. [Consultato il giorno 13 luglio 2019].

[104] «BigTable,» [Online]. Available: https://cloud.google.com/bigtable/. [Consultato il giorno 13 luglio 2019].

[105] M. Habringer, M. Moser e J. Pichler, «Reverse engineering PL/SQL legacy code: an experience report,» in *2014 IEEE International Conference on Software Maintenance and Evolution.*, 2014.

[106] F. Zemke, «What's new in SQL: 2011,» *ACM SIGMOD Record,* vol. 41, n. 1, pp. 67-73, 2012.

[107] S. Melnik, A. Gubarev, J. J. Long, G. Romer, S. Shivakumar, M. Tolton e T. Vassilakis, «Dremel: Interactive Analysis of Web-Scale Datasets,» *Proceedings of the VLDB Endowment,* vol. 3, n. 1-2, pp. 330-339, 2010.

[108] F. N. Afrati, D. Delorey, M. Pasumansky e J. D. Ullman, «Storing and Querying Tree-Structured Records in Dremel,» *Proceedings of the VLDB Endowment,* vol. 7, n. 12, pp. 1131-1142, 2014.

[109] «Root Servers,» [Online]. Available: https://www.iana.org/domains/root/servers. [Consultato il giorno 13 luglio 2019].

[110] S. Lattanzi, B. Moseley, S. Suri e S. Vassilvitskii, «Filtering: a method for solving graph problems in MapReduce.,» in *Proceedings of the twenty-third annual ACM symposium on Parallelism in algorithms and architectures.*, 2011.

[111] C. Olston, B. Reed, U. Srivastava, R. Kumar e A. Tomkins, «Pig latin: a not-so-foreign language for data processing.,» in *Proceedings of the 2008 ACM SIGMOD international conference on Management of data.*, 2008.

[112] C.-a. Chan, J. Vit, R. Clark, T. Kenter e V. Wan, «CHiVE: Varying Prosody in Speech Synthesis with a Linguistically Driven Dynamic Hierarchical Conditional Variational Network,» in *International Conference on Machine Learning*, 2019.

[113] J. Tordable, «MapReduce for Integer Factorization,» *https://arxiv.org/abs/1001.0421*, 2010.

[114] F. Belletti, L. Karthik, W. Krichene, N. Mayoraz, Y.-F. Chen, J. Anderson, T. Robie, T. Oguntebi, D. Shirron e A. Bleiwess, «Scaling Up Collaborative Filtering Data Sets through Randomized Fractal Expansions,» *https://arxiv.org/abs/1905.09874*, 2019.

[115] D. J. Abadi, S. R. Madden e M. C. Ferreira, «Integrating compression and execution in column-oriented database systems,» in *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, 2006.

[116] M. Stonebraker, D. J. Abadi, A. Batkin, X. Chen, M. Cherniack, E. Lau, A. Lin, S. Madden, E. O'Neil, P. O'Neil, A. Rasin, N. Tran e S. Zdonik, «C-store: a column-oriented DBMS,» in *VLDB '05 Proceedings of the 31st international conference on Very large data bases*, 2005.

[117] B. R. Iyer e D. Wilhite, «Data compression support in databases,» *VLDB*, vol. 94, pp. 695-704, 1994.

[118] Z. Chen, J. Gehrke e F. Korn, «Query optimization in compressed database systems.,» *ACM SIGMOD Record*, vol. 30, n. 2, pp. 271-282, 2001.

[119] D. Lemire, O. Kaser e E. Gutarra, «Reordering Rows for Better Compression: Beyond the Lexicographic Order,» *ACM Transactions on Database Systems (TODS)*, vol. 37, n. 3, p. 20, 2012.

[120] R. Pike, S. Dorward, R. Griesemer e S. Quinlan, «Interpreting the Data: Parallel Analysis with Sawzall,» *Scientific Programming*, vol. 13, n. 4, pp. 277-298, 2005.

[121] C. Chambers, A. Raniwala, F. Perry, S. Adams, R. R. Henry, R. Bradshaw e N. Weizenbaum, «FlumeJava: easy, efficient data-parallel pipelines,» *ACM Sigplan Notices*, vol. 45, n. 6, pp. 363-375, 2010.

[122] A. Brooks, E. Lu, D. Reicher, C. Spirakis e B. Weihl, «Demand Dispatch - Using Real-Time Control of Demand to help Balance Generation and Load,» *IEEE Power and Energy magazine*, pp. 20-29, 2010.

[123] H. Shey, S. Balaouras, M. Flug e P. Dostie, «The Forrester Wave™: Data Security Portfolio Vendors,» The Forrester Wave™, 2019.

[124] «Cloud Data Loss Prevention,» [Online]. Available: https://cloud.google.com/dlp/. [Consultato il giorno 13 luglio 2019].

[125] «G Suite,» [Online]. Available: https://gsuite.google.it/intl/it/products/admin/security-center/. [Consultato il giorno 13 luglio 2019].

[126] «Project Zero,» [Online]. Available: https://googleprojectzero.blogspot.com/. [Consultato il giorno 13 luglio 2019].

[127] «Google Vulnerability Reward Program (VRP),» [Online]. Available: https://www.google.com/about/appsecurity/reward-program/?hl=it. [Consultato il giorno 13 luglio 2019].

[128] «Google Infrastructure Security Design Overview,» [Online]. Available: https://cloud.google.com/security/infrastructure/design/. [Consultato il giorno 13 luglio 2019].

[129] «Sicurezza dei container,» [Online]. Available: https://cloud.google.com/containers/security/. [Consultato il giorno 13 luglio 2019].

[130] «VM schermate,» [Online]. Available: https://cloud.google.com/shielded-vm/. [Consultato il giorno 13 luglio 2019].

[131] «Virtual Private Cloud (VPC),» [Online]. Available: https://cloud.google.com/vpc/. [Consultato il giorno 13 luglio 2019].

[132] «Google Safe Browsing,» [Online]. Available: https://safebrowsing.google.com/. [Consultato il giorno 13 luglio 2019].

[133] «Titan Security Key,» [Online]. Available: https://cloud.google.com/titan-security-key/. [Consultato il giorno 13 luglio 2019].

[134] «Resource Manager,» [Online]. Available: https://cloud.google.com/resource-manager/. [Consultato il giorno 13 luglio 2019].

[135] «Identity-Aware Proxy,» [Online]. Available: https://cloud.google.com/iap/. [Consultato il giorno 13 luglio 2019].

[136] «Identity Platform,» [Online]. Available: https://cloud.google.com/identity-platform/. [Consultato il giorno 13 luglio 2019].

[137] «Cloud Security Scanner,» [Online]. Available: https://cloud.google.com/security-scanner/. [Consultato il giorno 13 luglio 2019].

[138] «Autorizzazione Binaria,» [Online]. Available: https://cloud.google.com/binary-authorization/. [Consultato il giorno 13 luglio 2019].

[139] «Stackdriver Logging,» [Online]. Available: https://cloud.google.com/logging/. [Consultato il giorno 13 luglio 2019].

[140] «Cloud Security Command Center,» [Online]. Available: https://cloud.google.com/security-command-center/. [Consultato il giorno 13 luglio 2019].

[141] «Trasparenza degli accessi,» [Online]. Available: https://cloud.google.com/access-transparency/. [Consultato il giorno 13 luglio 2019].

[142] R. Ward e B. Beyer, «BeyondCorp: A New Approach to Enterprise Security,» *;login:*, vol. 39, n. 6, pp. 6-11, 2014.

[143] B. Osborn, J. McWilliams, B. Beyer e M. Saltonstall, «BeyondCorp: Design to Deployment at Google,» *;login:*, vol. 41, n. 1, pp. 28-34, 2016.

[144] V. M. Escobedo, F. Zyzniewski e M. Saltonstall, «BeyondCorp: The User Experience,» *;login:*, vol. 42, n. 3, 2017.

[145] B. Beyer, C. McCormick Beske, J. Peck e M. Saltonstall, «Migrating to BeyondCorp: Maintaining Productivity While Improving Security,» *;login:*, vol. 42, n. 2, 2017.

[146] «ISO 27001,» [Online]. Available: https://cloud.google.com/security/compliance/iso-27001/. [Consultato il giorno 13 luglio 2019].

[147] «SOC 1,» [Online]. Available: https://cloud.google.com/security/compliance/soc-1/. [Consultato il giorno 13 luglio 2019].

[148] «PCI DSS,» [Online]. Available: https://cloud.google.com/security/compliance/pci-dss/. [Consultato il giorno 13 luglio 2019].

[149] «CSA STAR,» [Online]. Available: https://cloud.google.com/security/compliance/csa-star/. [Consultato il giorno 13 luglio 2019].

[150] «FedRAMP,» [Online]. Available: https://cloud.google.com/security/compliance/fedramp/. [Consultato il giorno 13 luglio 2019].

[151] «FISC (Giappone),» [Online]. Available: https://cloud.google.com/security/compliance/fisc/. [Consultato il giorno 13 luglio 2019].

[152] «GDPR,» [Online]. Available: https://cloud.google.com/security/compliance/gdpr/. [Consultato il giorno 13 luglio 2019].

[153] «Modulo convalidato FIPS 140-2,» [Online]. Available: https://cloud.google.com/security/compliance/fips-140-2-validated/. [Consultato il giorno 13 luglio 2019].

[154] «POPI (Sudafrica),» [Online]. Available: https://cloud.google.com/security/compliance/south-africa-popi/. [Consultato il giorno 13 luglio 2019].

[155] B. Hutchinson e M. Mitchell, «50 Years of Test (Un) fairness: Lessons for Machine Learning,» in *Proceedings of the Conference on Fairness, Accountability, and Transparency*, 2019.

[156] A. Li, O. Spyra, S. Perei, V. Dalibard, M. Jaderberg, C. Gu, D. Budden, T. Harley e P. Gupta, «A Generalized Framework for Population Based Training,» *https://arxiv.org/abs/1902.01894*, 2019.

[157] «GIS,» [Online]. Available: https://cloud.google.com/bigquery/docs/gis-intro. [Consultato il giorno 13 luglio 2019].

[158] M. Grupp, A. Gaschler, M. Ferrati, S. Christoph e M. Tenorth, «Cloud-based Mapping and Localization in Dynamic Warehouse Environments,» in *ROSCon 2018*, Madrid, Spain, 2018.

[159] «Hunterdon Healthcare,» [Online]. Available: https://gsuite.google.com/customers/hunterdon-healthcare-gsuite.html. [Consultato il giorno 13 luglio 2019].

[160] «Colorado Center for Personalized Medicine,» [Online]. Available: https://cloud.google.com/customers/colorado-center-for-personalized-medicine/. [Consultato il giorno 13 luglio 2019].

[161] «Health Data Compass,» [Online]. Available: https://www.healthdatacompass.org. [Consultato il giorno 13 luglio 2019].

[162] «Foundation for Precision Medicine,» [Online]. Available: https://cloud.google.com/customers/fpm/. [Consultato il giorno 13 luglio 2019].

[163] «Cloud Machine Learning Engine,» [Online]. Available: https://cloud.google.com/ml-engine/. [Consultato il giorno 13 luglio 2019].

[164] «TensorFlow,» [Online]. Available: https://www.tensorflow.org/. [Consultato il giorno 13 luglio 2019].

[165] V. Gulshan, L. Peng, M. Coram, M. C. Stumpe, D. Wu, A. Narayanaswamy, S. Venugopalan e e. al, «Development and Validation of a Deep Learning Algorithm for Detection of Diabetic Retinopathy in Retinal Fundus Photographs,» *Jama* , vol. 316, n. 22, pp. 2402-2410, 2016.

[166] «Deep learning algorithm does as well as dermatologists in identifying skin cancer,» [Online]. Available: https://news.stanford.edu/2017/01/25/artificial-intelligence-used-identify-skin-cancer/. [Consultato il giorno 13 luglio 2019].

[167] «Analyzing breast cancer images faster and better with machine learning,» [Online]. Available: https://cloud.google.com/customers/american-cancer-society/. [Consultato il giorno 13 luglio 2019].

[168] «American Cancer Society,» [Online]. Available: https://www.cancer.org. [Consultato il giorno 13 luglio 2019].

[169] «Google Genomics,» [Online]. Available: https://cloud.google.com/genomics/. [Consultato il giorno 13 luglio 2019].

[170] C. Pan, G. McInnes, N. Deflaux, M. Snyder, J. Bingham, S. Datta e P. S. Tsao, «Cloud-based interactive analytics for terabytes of genomic variants data,» *Bioinformatics,* vol. 33, n. 23, pp. 3709-3715, 2017.

[171] S. Gopaulakrishnan, S. Pollack, B. Stubbs, H. Pagès, J. Readey, S. Davis, L. Waldron, M. Morgan e V. Carey, «restfulSE: A semantically rich interface for cloud-scale genomics with Bioconductor,» *F1000Research,* vol. 8, p. 21, 2019.

[172] «MSSNG,» [Online]. Available: https://www.mss.ng. [Consultato il giorno 13 luglio 2019].

[173] «Analyzing 3024 rice genomes characterized by DeepVariant,» [Online]. Available: https://cloud.google.com/blog/products/data-analytics/analyzing-3024-rice-genomes-characterized-by-deepvariant. [Consultato il giorno 13 luglio 2019].

[174] «HITECH,» [Online]. Available: https://www.hhs.gov/hipaa/for-professionals/security/laws-regulations/index.html. [Consultato il giorno 13 luglio 2019].