

Integrità delle Query

Parte II

Indice

1	Integrità della computazione	2
1.1	Esempi	2
1.2	Integrità di storage e computazione	4
2	Approcci deterministici	6

Capitolo 1

Integrità della computazione

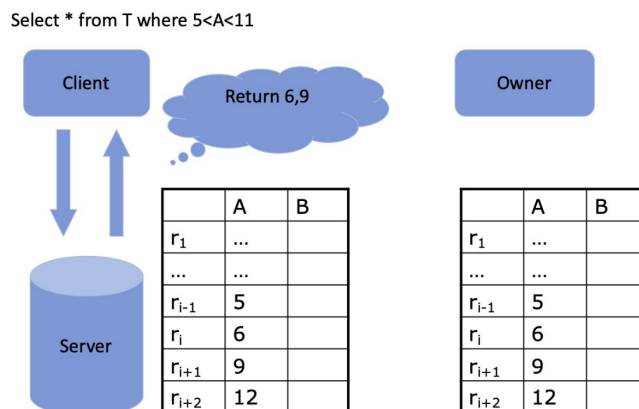
Nel nostro scenario di riferimento potremo avere più *data owner*; queste sono entità di cui ci fidiamo.

Il problema è che questi dati potrebbero essere affidati a dei *cloud provider* esterni, e che possano essere soggetti a delle *computazioni*; questo potrebbe essere un problema sia in termini di confidenzialità che in termini di **integrità**: *"chi mi dice che la tua computazione sia integra?"*.

1.1 Esempi

Esempio di una query

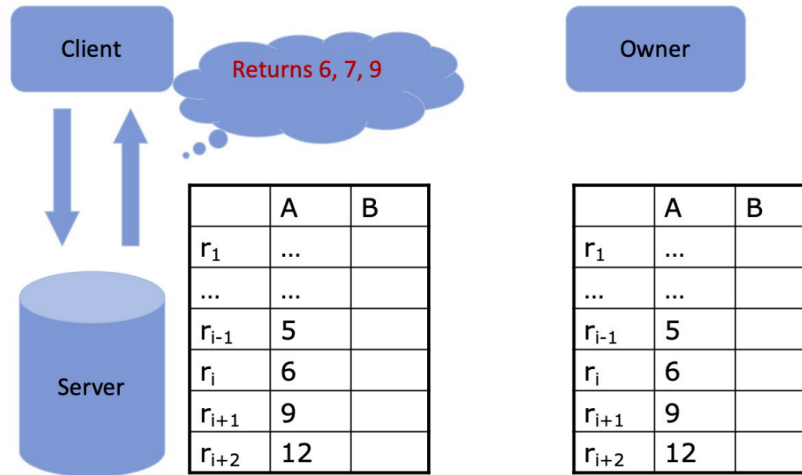
Abbiamo l'owner che affida i propri dati ad un provider esterno; abbiamo poi un client che effettua una query.



Esempio di query: iniezione

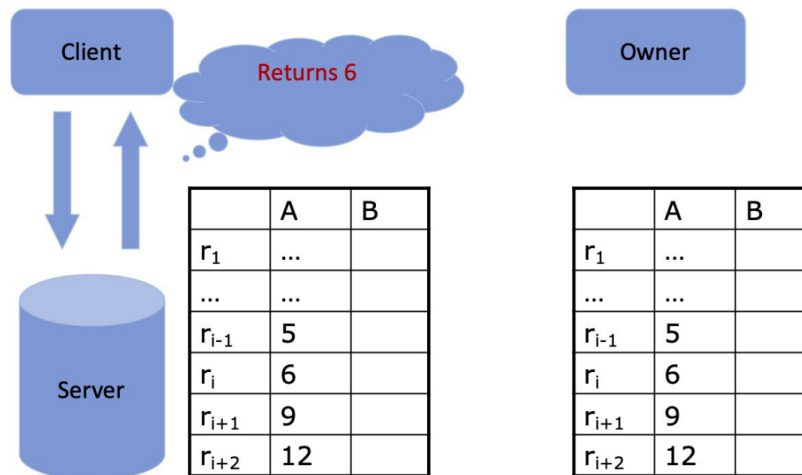
Viene iniettata un'informazione fasulla; *magari mi conviene dirti una cosa piuttosto che un'altra, le tue azioni dipendono da quello che ti dico...*

Select * from T where 5<A<11



Esempio di query: drop

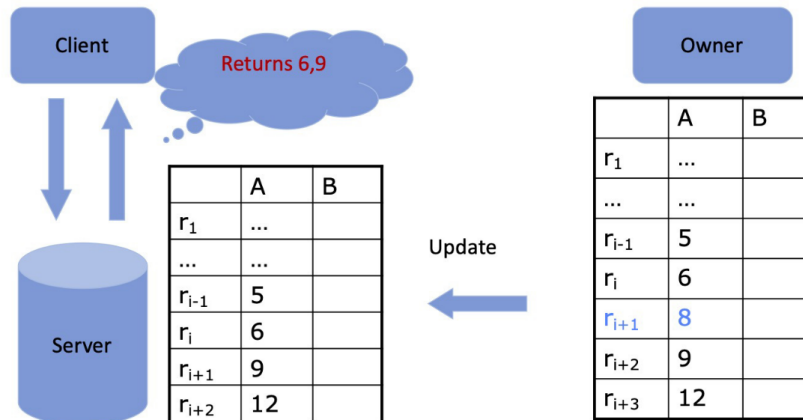
Select * from T where 5<A<11



Esempio di query: omissione

I dati potrebbero essere dinamici, dunque potrebbero essere richieste delle operazioni di update.

Select * from T where 5<A<11



1.2 Integrità di storage e computazione

Il data owner e gli utenti necessitano di meccanismi che assicurino l'integrità dei risultati delle query. Una query è integra se rispetta:

- **Correttezza:** il risultato viene calcolato sui dati veri dell'owner (primo esempio)
- **Completezza:** il risultato calcolati su tutti i dati (secondo esempio)
- **Freschezza:** il risultato è calcolato sull'ultima versione dei dati che l'owner ha dato (terzo esempio)

Ci sono due diversi tipi di approcci per rispondere al problema di integrità, ciascuno con i suoi vantaggi e svantaggi:

- **Deterministico:** *se il risultato di una computazione è integro, sono sicuro al 100% che sia integro*

Queste tecniche vengono implementate in modo che l'owner dà al provider, oltre ai dati da gestire, anche delle strutture ausiliarie che vengono sfruttate per verificare l'integrità della computazione

- **Probabilistico:** *ti dico sempre se è integro o no, ma non con certezza assoluta ma con una certa probabilità; c'è della probabilità di fare degli errori*

Perché si usano questi approcci? Sono tecniche che hanno lo svantaggio di non avere la certezza assoluta ma che hanno altri vantaggi (che vedremo

più avanti); il fatto che non avere certezza sia un problema dipende da caso a caso.

In queste tecniche il *qualcosa di ausiliario* sono dei "dati finti" (marcatori) che "aggiungo" ai dati veri; dalla presenza o meno capisco se la query è integra o no (se prima c'erano e poi non ci sono più, probabilmente c'è un errore)

Capitolo 2

Approcci deterministici

L'idea è che il proprietario *dà fuori* i dati e una struttura da lui calcolata. Quando il client vuole fare una computazione, restituisce oltre al risultato anche un *qualcosa in più* usando la struttura dati; questo prende il nome di **verification object**: è ciò che permette di verificare se il risultato della query è integro.

