Progetto a.a 2021/2022

Gruppo 30: Chiara Repetti

RELAZIONE - PARTE III

PROGETTAZIONE FISICA

Per la progettazione fisica della base di dati , sotto consiglio delle docenti, è stato creato un nuovo file sql ‘dummysocialmarket’. In questo file si è rielaborato il precedente ‘socialmarket’ aggiungendo un campo ‘dummy’ di tipo text in ogni relazione scelta, poiché esso serve per raggiungere più velocemente l’occupazione di spazio disco indicata. Nella specifica del progetto le relazioni coinvolte nel carico di lavoro dovevano avere un numero di tuple equivalenti a qualche decina di pagine (ogni pagina ha dimensione 8k), per fare ciò è stato utilizzato il generatore automatico di dati *datanamic.*

CARICO DI LAVORO:

Il carico di lavoro che è stato individuato è il seguente:

1. Selezionare i clienti che hanno reddito superiore a 6000 e inferiore a 2500 appartenenti a un determinato ente, con data maggiore del 18 luglio 2023
2. Selezionare il prodotto con una data quantità di uscita
3. Selezionare i volontari che hanno appuntamento

Tabella che, per ogni tabella coinvolta nelle operazioni del carico di lavoro, riporta il numero di tuple inserite e la dimensione in blocchi della tabella:

Relname	Relpages	kB	Reltuples
Volontari	39	312 kB	1500
Clienti	35	280 kB	1500
Appuntamenti	33	264 kB	1500
Prodottiuscita	31	248 kB	1500

I Query

```
--seleziono i clienti che hanno ente Renault il reddito minore di 2500 o
maggiore di 6000 e con data inizio maggiore del 18 luglio 2023

SELECT Cod_Fisc
FROM clienti
WHERE ente='Renault' AND (reddito < 2500 or reddito> 6000) AND
data_inizio>='07/18/2023';
```

Query complete 00:00:00.076

Il sistema senza indice esegue una scansione sequenziale

clienti

→ Seq Scan on clienti as clienti
Filter: ((((ente)::text = 'Renault'::text) AND ((cod_fam)::text > '30'::text) AND ((cod_fam)::text < '2000'::text)) OR (reddito < '2500'::numeric))

Nel contesto del piano fisico, si opta per la creazione di un indice basato sull'attributo "ente", presente in uno dei fattori booleani della query.
L'indice `ind_ente` è un indice ordinato e clusterizzato sull'attributo ente in quanto il numero di enti è di quantità nettamente inferiore rispetto agli altri attributi (16 diversi tipi di enti), perciò l'utilizzo di un indice clusterizzato in questo caso è nettamente più efficiente. Inoltre si può pensare che nel social market l'ente sarà un attributo alquanto frequente nelle clausole di ricerca delle queery.

```
--indice su ente
CREATE INDEX ind_ente
ON clienti(ente);
CLUSTER clienti USING ind_ente;
```

Dopo aver eseguito la queery:

Query complete 00:00:00.049

Dopo la creazione dell'indice il sistema sceglie di effettuare la queery con un index scan

ind_ente clienti

→ Bitmap Heap Scan on clienti as clienti (rows=76 loops=1)
Filter: ((data_inizio >= '2023-07-18'::date) AND ((reddito < '2500'::numeric) OR (reddito > '6000'::numeric)))
Rows Removed by Filter: 17
Recheck Cond: ((ente)::text = 'Renault'::text)
Heap Blocks: exact=3

→ Bitmap Index Scan using ind_ente (rows=93 loops=1)
Index Cond: ((ente)::text = 'Renault'::text)

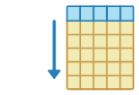
II Query

```
-- Il prodotto con quantità in uscita = 3

SELECT Cod_Prod
FROM ProdottiUscita
WHERE quantità = 3;
```

Query complete 00:00:00.074

Il sistema senza indice esegue una scansione sequenziale



prodottiuscita

→ Seq Scan on prodottiuscita as prodottiuscita (rows=49 loops=1)
Filter: ("quantità" = '3'::numeric)
Rows Removed by Filter: 1451

Come piano fisico è stato scelto un indice ordinato sull'attributo quantità. In questo caso si è deciso di non clusterizzare l'indice poiché la presenza di un gran numero di prodotti ridurrà la quantità di dati da recuperare e manipolare, rendendo meno necessaria l'organizzazione dei dati tramite clusterizzazione.

```
--indice su quantità di prodottiuscita

CREATE INDEX ind_quant
ON ProdottiUscita(quantità);
```

Dopo aver eseguito la queery:

Query complete 00:00:00.046

Dopo la creazione dell'indice il sistema sceglie di effettuare la queery con un index scan:

ind_quant prodottiuscita

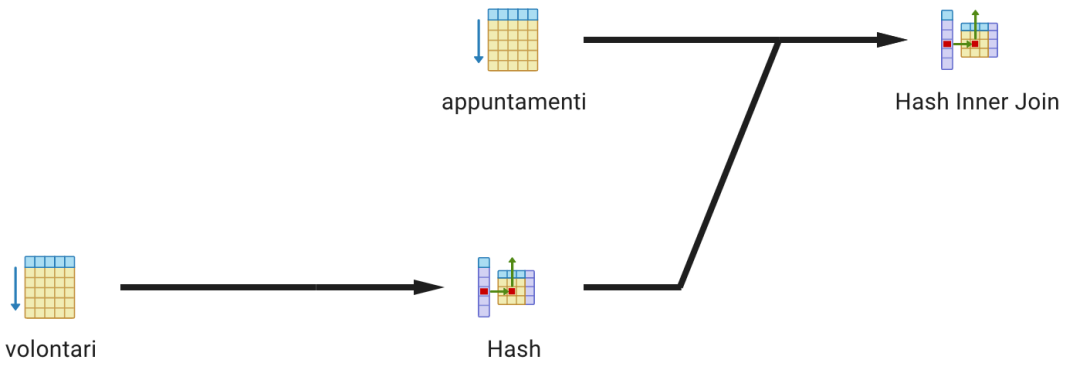
→ Bitmap Heap Scan on prodottiuscita as prodottiuscita (rows=49 loops=1)
Recheck Cond: ("quantità" = '3'::numeric)
Heap Blocks: exact=26

→ Bitmap Index Scan using ind_quant (rows=49 loops=1)
Index Cond: ("quantità" = '3'::numeric)

III Query

```
-- I codici fiscali dei volontari che hanno appuntamento

SELECT Volontari.id_Vol
FROM Volontari
JOIN Appuntamenti ON Appuntamenti.id_Vol=Volontari.id_Vol;
```



→ Hash Inner Join (rows=1500 loops=1)
Hash Cond: ((appuntamenti.id_vol)::text = (volontari.id_vol)::text)

→ Seq Scan on appuntamenti as appuntamenti (rows=1500 loops=1)

→ Hash (rows=1500 loops=1)
Buckets: 2048 **Batches**: 1 **Memory Usage**: 81 kB

→ Seq Scan on volontari as volontari (rows=1500 loops=1)

Query complete 00:00:00.086

L'indice scelto per eseguire la queery è prevedibilmente id_vol, nonostante ciò il sistema ha deciso di utilizzare un hashjoin poiché è stato stimato come un'opzione più efficiente dal punto di vista delle prestazioni, rispetto che l'index nested loop.

Successivamente, tramite la creazione degli indici su id_vol delle due relazioni: appuntamenti e volontari, il piano di esecuzione fisico utilizzato è il merge join. Gli indici permettono una scansione efficiente e ordinata delle tabelle, in quanto il merge join richiede che le tabelle siano ordinate in base all'attributo di join. Inoltre, il merge join presenta vantaggi in termini di risorse e tempo di esecuzione rispetto all'hash join, poiché quest'ultimo richiede un'allocazione di memoria maggiore per creare la tabella hash.

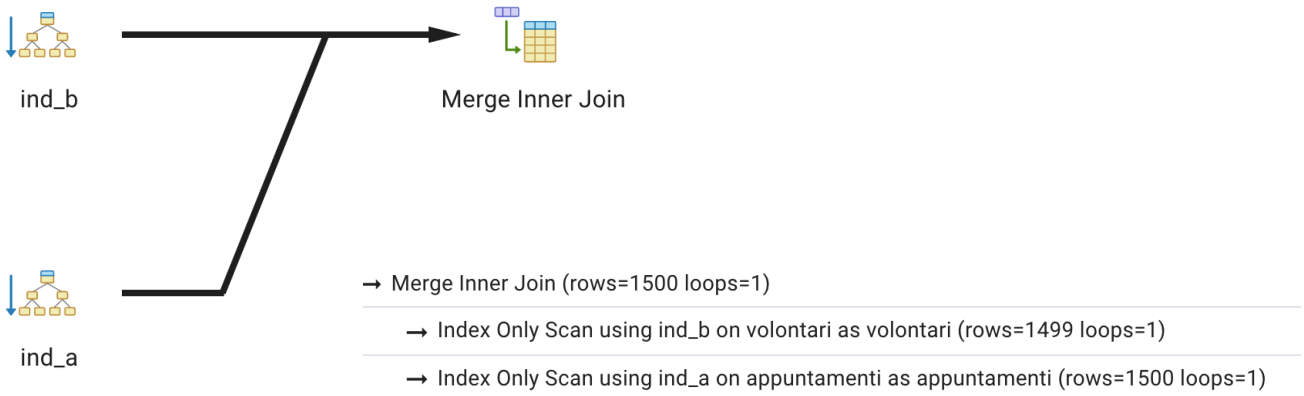
```
--creazione inidice sul volontario in appuntamenti
CREATE INDEX ind_a
ON Appuntamenti(id_Vol);

--creazione indice sul volontario in volontari
CREATE INDEX ind_b
ON Volontari(id_Vol);
```

Dopo aver eseguito la queery:

Query complete 00:00:00.050

Dopo la creazione dell'indice il sistema sceglie di effettuare la queery con il merge inter join:



TRANSAZIONI

La transazione è stata eseguita sullo schema *socialmarket*

La transazione che è stata scelta è la seguente:

1. Selezionare i crediti di tutte le famiglie
2. Aggiungere 10 punti al credito della famiglia f0008
3. Selezionare le informazioni della famiglia f0008 per verificarne l'adeguatezza (per il vincolo check non può essere maggiore di 60)

Codice :

```
BEGIN;
SET CONSTRAINTS ALL IMMEDIATE;
SET TRANSACTION ISOLATION LEVEL READ COMMITTED ;
-- guardo i crediti mensili delle famiglie
SELECT Cod_Fam, punti_mensili
FROM Nuclei_Familiari
GROUP BY Cod_Fam;

-- Operazione di scrittura
-- vado ad aggiornare i punti con il valore nella famiglia con codice f0008
UPDATE Nuclei_Familiari
SET Punti_mensili = Punti_mensili+10
WHERE Cod_Fam = 'f0008';

-- Leggo dati aggiornati
SELECT *
FROM Nuclei_Familiari
WHERE Cod_Fam = 'f0008';

COMMIT;
```

LIVELLO DI ISOLAMENTO

Il livello di isolamento scelto è "READ COMMITTED" (in Postgresql è equivalente a uncomitted), esso garantisce che una transazione legga solo dati che sono stati confermati (committed) da altre transazioni. Ciò significa che le modifiche apportate da altre transazioni non ancora confermate (non ancora committate) non saranno visibili all'interno della transazione corrente.
Questo livello non consente letture di dati non valide (dirty reads), poiché protegge la transazione dalle modifiche non confermate effettuate da altre transazioni.
Nell' esempio specifico, se una transazione A inizia e legge alcuni dati dalla tabella Nuclei_Familiari per il codice famiglia 'f0008', che ha un valore di punti_mensili pari a 30.

Nel frattempo, transazione B inizia e aggiorna il valore di punti_mensili per il codice famiglia 'f0008' a 40. La transazione A continua la sua lettura e legge il valore di punti_mensili come 30, poiché il livello di isolamento READ COMMITTED impedisce alla transazione A di leggere dati non confermati. Successivamente, transazione B conferma l'aggiornamento e il valore di punti_mensili diventa 40. La transazione A ha letto un dato non valido (30 anziché 40) perché non ha avuto accesso ai dati aggiornati della transazione B durante la sua lettura (dirty read). Tuttavia, con il livello di isolamento scelto, una volta che la transazione A ha letto il valore, non vedrà le modifiche apportate da altre transazioni durante la stessa lettura. Se la transazione A avesse letto nuovamente i dati dopo che la transazione B aveva confermato l'aggiornamento, avrebbe visto il valore corretto di punti_mensili (40).
La transizione eseguita effettua una sola lettura della tabella Nuclei_Familiari tra l'inizio e il COMMIT della transazione. Poiché non ci sono altre query nella transazione che coinvolgono la stessa tabella, le letture fantasma non possono verificarsi in questo caso, quindi non serve un livello “più restrittivo”.
Inoltre, il livello READ COMMITTED offre un buon equilibrio tra prestazioni e isolamento, in quanto permette alle transazioni concorrenti di accedere ai dati letti senza essere bloccate.

Politica controllo accesso

Codice dei permessi:

```
CREATE USER Alice PASSWORD 'gestore';
CREATE USER Roberto PASSWORD 'volontario';
GRANT USAGE ON SCHEMA "socialmarket" TO Alice;
GRANT USAGE ON SCHEMA "socialmarket" TO Roberto;

--permessi di Alice
-- Essendo gestore avrà abbastanza permessi
-- Le revoco la delete sulle persone, clienti, nuclei, volontari, inventario e enti
GRANT SELECT, INSERT, UPDATE, DELETE ON ALL TABLES IN SCHEMA "socialmarket" TO
Alice;
REVOKE DELETE ON Persone, Nuclei_Familiari, Clienti, Volontari, Inventario, Enti
FROM Alice;

--permessi di Roberto
-- do i diritti necessari affinche' possa svolgere l'attività di volontario al meglio
-- può vedere tutto ma ha delle limitazioni essendo un volontario sull' update delete e
insert
GRANT SELECT, UPDATE ON ALL TABLES IN SCHEMA "socialmarket" TO Roberto;
REVOKE UPDATE ON Persone, Volontari, Enti, Donazioni, ScaricoProdotti FROM
Roberto;
GRANT DELETE ON Appuntamenti, Trasporti, ProdottiUscita, ProdottiEntrata TO
Roberto;
GRANT INSERT ON Inventario, ProdottiUscita, ProdottiEntrata TO Roberto;
```

Tabella dei permessi di Alice (Gestore Market):

	SELECT	INSERT	DELETE	UPDATE
PERSONE	SI	SI	NO	SI
NUCLEI_FAMILIARI	SI	SI	NO	SI
CLIENTI	SI	SI	NO	SI
VOLONTARI	SI	SI	NO	SI
APPUNTAMENTI	SI	SI	SI	SI
INVENTARIO	SI	SI	NO	SI
ENTI	SI	SI	NO	SI
DONAZIONI	SI	SI	SI	SI
PRODOTTI ENTRATA	SI	SI	SI	SI
PRODOTTI USCITA	SI	SI	SI	SI
TRASPORTI	SI	SI	SI	SI
SCARICO PRODOTTI	SI	SI	SI	SI

Tabella dei permessi di Roberto (Volontario):

	SELECT	INSERT	DELETE	UPDATE
PERSONE	SI	NO	NO	NO
NUCLEI_FAMILIARI	SI	NO	NO	SI
CLIENTI	SI	NO	NO	SI
VOLONTARI	SI	NO	NO	NO
APPUNTAMENTI	SI	NO	SI	SI
INVENTARIO	SI	SI	NO	SI
ENTI	SI	NO	NO	NO
DONAZIONI	SI	NO	NO	NO
PRODOTTI ENTRATA	SI	SI	SI	SI
PRODOTTI USCITA	SI	SI	SI	SI
TRASPORTI	SI	NO	SI	SI
SCARICO PRODOTTI	SI	NO	NO	NO

PERMESSI DI ALICE (GESTORE)

Alice, essendo un gestore del market, avrà bisogno di avere un insieme di privilegi e permessi più ampio rispetto ad altri ruoli, poiché sarà coinvolta nella gestione e nel controllo di diverse attività. Questo potrebbe includere la possibilità di leggere, inserire, aggiornare e cancellare dati da diverse tabelle, oltre a svolgere altre operazioni amministrative necessarie per la gestione del market. I suoi privilegi devono essere adeguatamente concessi per consentirle di accedere a tutte le informazioni rilevanti e svolgere le sue funzioni in modo efficace e sicuro.

Le sono stati assegnati ampi privilegi che includono la capacità di inserire, selezionare e aggiornare dati. Questo è essenziale perché, in qualità di gestore, è responsabile della supervisione e del controllo delle diverse attività all'interno del market. Dovendo prendere decisioni e autorizzare varie operazioni, le è stato concesso un ampio spettro di privilegi per assicurarsi che tutte le azioni siano eseguite attraverso di lei, garantendo così una gestione efficiente e coerente del market.

È stato considerato opportuno revocare il permesso di cancellazione riguardante le informazioni delle persone, inclusi volontari, clienti e donatori, così come le donazioni, compresi gli enti, e l'inventario. Questa decisione è stata presa per garantire una maggiore sicurezza e integrità dei dati. Mantenere un registro completo di tutte le attività svolte nel socialmarket può rivelarsi estremamente utile per scopi di tracciabilità e monitoraggio. Inoltre, proteggere le informazioni sensibili dalle cancellazioni accidentali o non autorizzate contribuisce a mantenere l'integrità dei dati a lungo termine, soprattutto per quanto riguarda l'inventario, che rappresenta una parte essenziale delle operazioni del market. Revocando il permesso di cancellazione, si garantisce che le informazioni preziose siano conservate e che le operazioni di gestione del market siano svolte in modo accurato e controllato. Tuttavia, per quanto riguarda gli appuntamenti, è stato deciso di mantenere il

permesso di cancellazione. Questa decisione è stata presa perché gli appuntamenti possono essere soggetti a modifiche o cancellazioni, a seguito di cambiamenti negli orari o nelle disponibilità dei volontari e dei clienti. La possibilità di cancellare gli appuntamenti consente una maggiore flessibilità nella gestione delle prenotazioni, permettendo di adattarsi alle esigenze dei partecipanti e delle risorse disponibili.

PERMESSI DI ROBERTO (VOLONTARIO)

Roberto ha il permesso di SELECT su tutte le tabelle poiché i dati non sono considerati segreti o sensibili. Essendo un volontario, è importante che abbia accesso alle informazioni necessarie per svolgere efficacemente le sue attività senza restrizioni. La possibilità di SELECT su tutte le tabelle gli consente di visualizzare e consultare i dati pertinenti, come gli appuntamenti, i trasporti e le informazioni sui prodotti, facilitando il suo coinvolgimento nelle operazioni del socialmarket.
A Roberto è stata revocata la possibilità di eseguire l'operazione di INSERT su tutte le tabelle, ad eccezione delle tabelle dei prodotti. Questa decisione è stata presa per garantire che le altre tabelle, che richiedono un controllo più rigoroso, siano gestite e controllate esclusivamente dal gestore del market. Limitando il privilegio di inserimento su queste tabelle, si assicura che le operazioni di aggiunta di dati vengano effettuate solo da una figura autorizzata, come il gestore, per garantire l'integrità e la coerenza dei dati. Inoltre, consentendo a Roberto di effettuare inserimenti nelle tabelle dei prodotti, si offre la possibilità di contribuire al catalogo dei prodotti disponibili, mantenendo al contempo un controllo appropriato sulle altre informazioni sensibili nel socialmarket.
È stato concesso il permesso di DELETE sulle tabelle dei prodotti, degli appuntamenti e dei trasporti. Questa decisione è stata presa considerando la flessibilità necessaria nelle operazioni quotidiane del socialmarket. Ad esempio, Roberto potrebbe dover cancellare un appuntamento a causa di imprevisti o indisponibilità, oppure potrebbe essere necessario cancellare informazioni sui trasporti se il mezzo a sua disposizione non è più disponibile. Consentendo a Roberto di effettuare cancellazioni in queste tabelle, si offre un livello di autonomia e adattabilità nell'esecuzione delle attività, permettendogli di gestire al meglio le situazioni impreviste che possono verificarsi nel contesto del suo coinvolgimento come volontario.
Il privilegio di eseguire operazioni di UPDATE è stato concesso solo su alcune tabelle specifiche, tra cui clienti, nuclei familiari, appuntamenti, inventario, prodotti (entrata e uscita) e trasporti; tenendo conto delle responsabilità e delle attività che spettano a un volontario all'interno del socialmarket.
Immaginiamo una situazione in cui Roberto sta gestendo gli appuntamenti dei clienti. Potrebbe accadere che un cliente richieda di cambiare l'orario o la data dell'appuntamento per motivi personali. Grazie al privilegio di UPDATE sulla tabella degli appuntamenti, Roberto avrà la possibilità di apportare tali modifiche in modo tempestivo e accurato, garantendo la soddisfazione del cliente e la pianificazione efficiente delle attività.
Concedendo a Roberto il privilegio di UPDATE solo su tabelle specifiche, si garantisce che sia coinvolto nelle attività chiave del socialmarket e possa svolgere il suo ruolo con flessibilità e prontezza. Allo stesso tempo, le altre tabelle contenenti informazioni più sensibili e critiche saranno sotto il controllo del gestore del market.