Progetto Basi di Dati 2024-25
“FANTASANREMO”
Parte III

[37]

[Riccardo Malchiodi s5680500, Andrei Dorin Sandu s6483324, Giulia Ponassi s6145273]

1. PROGETTAZIONE FISICA

1A+1C - CARICO DI LAVORO

Q1 - QUERY CON SINGOLA SELEZIONE E NESSUN JOIN

LINGUAGGIO NATURALE

Mostrare tutte le informazioni delle squadre che occupano la prima posizione nel campionato mondiale.

SQL

*SELECT * FROM Squadra_CL WHERE posizioneCampionatoMondiale = 1;*

Q2 - QUERY CON CONDIZIONE DI SELEZIONE COMPLESSA E NESSUN JOIN

LINGUAGGIO NATURALE

Selezionare gli utenti che hanno più di 5 Baudi e il cui username inizia con la lettera A.

SQL

*SELECT * FROM Utenti_CL WHERE numeroBaudi > 5 AND username LIKE 'A%';*

Q3 - QUERY CON ALMENO UN JOIN E ALMENO UNA CONDIZIONE DI SELEZIONE

LINGUAGGIO NATURALE

Visualizzare l’email degli utenti, il nome della squadra e il punteggio delle squadre create dagli utenti, ma solo se il punteggio è maggiore di 50

SQL

SELECT u.email, s.nome, s.punteggioFantasanremo

FROM Utenti_CL u

JOIN Crea_CL c ON u.email = c.email_Utenti

JOIN Squadra_CL s ON c.nome_Squadra = s.nome

WHERE s.punteggioFantasanremo > 50;

1D - PROGETTO FISICO

[Riportare nella seguente tabella l'elenco degli indici che si intendono creare per: (1) ciascuna query del carico di lavoro individualmente; (2) l'insieme delle query del carico di lavoro, motivando opportunamente, in modo sintetico, le scelte effettuate]

Id query	*Relazione*	*Chiave di ricerca*	*Tipo (ordinato/hash, clusterizzato/non clusterizzato)*	*Motivazione*
Q1	*Squadra_CL*	*posizioneCampionatoMondiale*	*indice ordinato non cluster*	*Permette di trovare velocemente le squadre in una certa posizione*
Q2	*Utenti_CL*	*(numeroBaudi, username)*	*indice ordinato non cluster*	*Ottimizza l'accesso combinato su più condizioni*
Q3	*Crea_CL*	*email_Utenti*	*indice ordinato non cluster*	*Ottimizza il join con Utenti_CL*
Q3	*Squadra_CL*	*nome*	*indice ordinato non cluster*	*Ottimizza il join con Squadra_CL*

Schema fisico complessivo per il carico di lavoro	*Motivazione*
CREATE INDEX idx_posizione ON Squadra_CL(posizioneCampionatoMondiale); *CREATE INDEX idx_username_baudi ON Utenti_CL(numeroBaudi, username);* *CREATE INDEX idx_email_squadra ON Crea_CL(email_Utenti);* *CREATE INDEX idx_squadra_nome ON Squadra_CL(nome);*	*Gli indici sono stati scelti per ottimizzare le prestazioni del carico di lavoro. Sono stati indicizzati gli attributi usati come filtri nelle WHERE e quelli utilizzati nei join.* *L'uso degli indici ha ridotto i tempi di esecuzione, evitando scansioni complete delle tabelle.*

1G-ANALISI PIANI DI ESECUZIONE SCELTI DAL SISTEMA

Q1 - QUERY CON SINGOLA SELEZIONE E NESSUN JOIN

PIANO DI ESECUZIONE SCELTO DAL SISTEMA PRIMA DELLA CREAZIONE DELLO SCHEMA FISICO

NOTA: Abbiamo utilizzato lensQL

```
DROP INDEX IF EXISTS idx_posizione;

EXPLAIN ANALYZE
SELECT * FROM Squadra_CL WHERE posizioneCampionatoMondiale = 1;
```

```
Seq Scan on squadra_cl  (cost=0.00..10.50 rows=1 width=1774) (actual time=0.021..0.026 rows=2 loops=1)
  Filter: (posizionecampionatomondiale = 1)
  Rows Removed by Filter: 14
Planning Time: 0.190 ms
Execution Time: 0.055 ms
```

PIANO DI ESECUZIONE SCELTO DAL SISTEMA DOPO DELLA CREAZIONE DELLO SCHEMA FISICO

```
CREATE INDEX idx_posizione ON Squadra_CL(posizioneCampionatoMondiale);

EXPLAIN ANALYZE

SELECT * FROM Squadra_CL WHERE posizioneCampionatoMondiale = 1;
```

```
Seq Scan on squadra_cl  (cost=0.00..1.20 rows=1 width=1774) (actual time=0.017..0.022 rows=2 loops=1)
  Filter: (posizionecampionatomondiale = 1)
  Rows Removed by Filter: 14
Planning Time: 0.100 ms
Execution Time: 0.050 ms
```

CONFRONTO TRA I DUE PIANI

Tempo esecuzione, PRIMA	*Tempo esecuzione DOPO*	*Motivazione*
0.055ms	*0.050ms*	*Anche dopo la creazione dell'indice idx_posizione, LensQL ha scelto una Seq Scan perché la tabella è piccola. L'indice non è stato utilizzato, ma diventerebbe utile con tabelle più grandi e filtri più selettivi.*

Q2 - QUERY CON CONDIZIONE DI SELEZIONE COMPLESSA E NESSUN JOIN

PIANO DI ESECUZIONE SCELTO DAL SISTEMA PRIMA DELLA CREAZIONE DELLO SCHEMA FISICO

```
EXPLAIN ANALYZE

SELECT * FROM Utenti_CL WHERE numeroBaudi > 5 AND username LIKE 'A%';
```

```
Seq Scan on utenti_cl  (cost=0.00..11.20 rows=1 width=956) (actual time=0.025..0.031 rows=4 loops=1)

  Filter: ((numerobaudi > 5) AND ((username)::text ~~ 'A%'::text))

  Rows Removed by Filter: 16

Planning Time: 1.531 ms

Execution Time: 0.061 ms
```

PIANO DI ESECUZIONE SCELTO DAL SISTEMA DOPO DELLA CREAZIONE DELLO SCHEMA FISICO

```
CREATE INDEX idx_username_baudi ON Utenti_CL(numeroBaudi, username);

EXPLAIN ANALYZE

SELECT * FROM Utenti_CL WHERE numeroBaudi > 5 AND username LIKE 'A%';
```

```
Seq Scan on utenti_cl  (cost=0.00..11.20 rows=1 width=956) (actual time=0.019..0.025 rows=4 loops=1)
```

Filter: ((numerobaudi > 5) AND ((username)::text ~~ 'A%'::text))

Rows Removed by Filter: 16

Planning Time: 0.099 ms

Execution Time: 0.050 ms

CONFRONTO TRA I DUE PIANI

[Riportare nella seguente tabella i tempi di esecuzione per i piani ottenuti prima e dopo la creazione dello schema fisico complessivo, giustificando i piani e i tempi ottenuti]

Tempo esecuzione PRIMA	*Tempo esecuzione DOPO*	*Motivazione*
0.061ms	*0.050ms*	*Stessa motivazione di Q1*

Q3 - QUERY CON ALMENO UN JOIN E ALMENO UNA CONDIZIONE DI SELEZIONE

PIANO DI ESECUZIONE SCELTO DAL SISTEMA PRIMA DELLA CREAZIONE DELLO SCHEMA FISICO

```
DROP INDEX IF EXISTS idx_email_squadra;

DROP INDEX IF EXISTS idx_squadra_nome;

EXPLAIN ANALYZE

SELECT u.email, s.nome, s.punteggioFantasanremo

FROM Utenti_CL u

JOIN Crea_CL c ON u.email = c.email_Utenti

JOIN Squadra_CL s ON c.nome_Squadra = s.nome

WHERE s.punteggioFantasanremo > 50;
```

Hash Join (cost=12.06..23.21 rows=5 width=1036) (actual time=0.134..0.161 rows=38 loops=1)

  Hash Cond: ((u.email)::text = (c.email_utenti)::text)

  -> Seq Scan on utenti_cl u (cost=0.00..10.80 rows=80 width=516) (actual time=0.010..0.014 rows=20 loops=1)

  -> Hash (cost=12.00..12.00 rows=5 width=1036) (actual time=0.109..0.111 rows=20 loops=1)

        Buckets: 1024 Batches: 1 Memory Usage: 10kB

-> Hash Join (cost=1.26..12.00 rows=5 width=1036) (actual time=0.071..0.096 rows=20 loops=1)

Hash Cond: ((c.nome_squadra)::text = (s.nome)::text)

-> Seq Scan on crea_cl c (cost=0.00..10.50 rows=50 width=1032) (actual time=0.009..0.013 rows=24 loops=1)

-> Hash (cost=1.20..1.20 rows=5 width=520) (actual time=0.028..0.029 rows=6 loops=1)

Buckets: 1024 Batches: 1 Memory Usage: 9kB

-> Seq Scan on squadra_cl s (cost=0.00..1.20 rows=5 width=520) (actual time=0.011..0.017 rows=6 loops=1)

Filter: (punteggiofantasanremo > 50)

Rows Removed by Filter: 10

Planning Time: 0.406 ms

Execution Time: 0.225 ms

PIANO DI ESECUZIONE SCELTO DAL SISTEMA DOPO DELLA CREAZIONE DELLO SCHEMA FISICO

```
CREATE INDEX idx_email_squadra ON Crea_CL(email_Utenti);

CREATE INDEX idx_squadra_nome ON Squadra_CL(nome);

EXPLAIN ANALYZE

SELECT u.email, s.nome, s.punteggioFantasanremo

FROM Utenti_CL u

JOIN Crea_CL c ON u.email = c.email_Utenti

JOIN Squadra_CL s ON c.nome_Squadra = s.nome

WHERE s.punteggioFantasanremo > 50;
```

Hash Join (cost=12.06..23.21 rows=5 width=1036) (actual time=0.030..0.038 rows=38 loops=1)

Hash Cond: ((u.email)::text = (c.email_utenti)::text)

-> Seq Scan on utenti_cl u (cost=0.00..10.80 rows=80 width=516) (actual time=0.004..0.005 rows=20 loops=1)

-> Hash (cost=12.00..12.00 rows=5 width=1036) (actual time=0.022..0.023 rows=20 loops=1)

Buckets: 1024 Batches: 1 Memory Usage: 10kB

-> Hash Join (cost=1.26..12.00 rows=5 width=1036) (actual time=0.013..0.019 rows=20 loops=1)

Hash Cond: ((c.nome_squadra)::text = (s.nome)::text)

-> Seq Scan on crea_cl c (cost=0.00..10.50 rows=50 width=1032) (actual time=0.002..0.003 rows=24 loops=1)

-> Hash (cost=1.20..1.20 rows=5 width=520) (actual time=0.008..0.008 rows=6 loops=1)

Buckets: 1024 Batches: 1 Memory Usage: 9kB

-> Seq Scan on squadra_cl s (cost=0.00..1.20 rows=5 width=520) (actual time=0.003..0.005 rows=6 loops=1)

Filter: (punteggiofantasanremo > 50)

Rows Removed by Filter: 10

Planning Time: 0.092 ms

Execution Time: 0.056 ms

CONFRONTO TRA I DUE PIANI

[Riportare nella seguente tabella i tempi di esecuzione per i piani ottenuti prima e dopo la creazione dello schema fisico complessivo, giustificando i piani e i tempi ottenuti]

Tempo esecuzione PRIMA	*Tempo esecuzione DOPO*	*Motivazione*
0.225ms	*0.056ms*	*Stessa motivazione di Q1*

2. CONTROLLO DELL'ACCESSO

GERARCHIA TRA I RUOLI

GERARCHIA

amministratore_fantasanremo > amministratore_lega > proprietario_lega > utente_premium

MOTIVAZIONE GERARCHIA

La gerarchia proposta rispecchia i livelli crescenti di responsabilità e privilegi. L'utente premium ha accesso base. Il proprietario di lega può creare e gestire leghe. L'amministratore di lega ha più poteri, ad esempio su squadre e utenti della lega. L'amministratore del Fantasanremo ha privilegi completi per gestire l'intero sistema.

ASSEGNAZIONE PRIVILEGI SPECIFICI AI RUOLI

[Riportare nella prima colonna della seguente tabella le relazioni considerate e in ciascuna altra cella (i,j) i privilegi specifici (quindi non acquisiti tramite la gerarchia) che si intendono assegnare al ruolo j sulla tabella i].

RELAZIONE	*Amministratore del FantaSanremo*	*Utente premium*	*Amministratore lega*	*Proprietario lega*
Utenti_CL		*SELECT, INSERT, UPDATE*		
Squadra_CL		*SELECT, INSERT, UPDATE*		
Crea_CL	*ALL PRIVILEGES*	*SELECT, INSERT, UPDATE*		*INSERT*