

Basi di Dati
a.a. 2022/2023
Progetto “Orti scolastici”

PARTE III

Indice

| | |
|---|--------|
| 9. Progetto fisico | |
| a) Interrogazioni carico di lavoro | pag. 2 |
| b) Elenco indici | pag. 2 |
| c) Dimensionamento tabelle | pag. 3 |
| d) Piani e tempi d'esecuzione | pag. 4 |
| 10. Controllo dell'accesso | pag. 7 |

9. Interrogazioni del carico di lavoro

a) Descrizione in linguaggio naturale e codice SQL per interrogazioni carico di lavoro:

1) « Selezionare il nome scientifico e la data della prima installazione delle piante messe a dimora nell'anno scolastico 2006/2007 »

```
SELECT DISTINCT gruppo.nomescientifico, MIN(gruppo.data)
FROM gruppo
WHERE data BETWEEN '2006-09-01' AND '2007-08-31'
GROUP BY nomescientifico ORDER BY nomescientifico;
```

2) « Selezionare tutti i sensori di rilevamento umidità e i rispettivi orti in cui sono alloggiati »

```
SELECT orto.id, orto.nome, sensore.num, sensore.tiposensore
FROM sensore JOIN orto ON sensore.idorto = orto.id
WHERE tiposensore = 'umidita' ORDER BY orto.id, sensore.num
```

3) « Selezionare le date delle rilevazioni, in ordine cronologico, nelle quali sono presenti informazioni di temperatura estrema (maggiore di 45°C) »

```
SELECT rilevazione.dataoraoss, informazione.tipoinformazione,
informazione.valore
FROM informazione JOIN rilevazione ON informazione.idrilevazione =
rilevazione.id
WHERE tipoinformazione = 'temperatura' AND valore > 45
ORDER BY rilevazione.dataoraoss
```

b) Elenco indici creati per gestire i carichi di lavoro sopra descritti:

- 1) Indice ad albero su attributo Data della relazione Gruppo, non clusterizzato; scelto in quanto carico di lavoro concentrato su attributo Data e indice su di essa precedentemente non presente; inoltre essendo la richiesta di tipo a intervallo, un indice hash non avrebbe avuto effetto.
- 2) Indice di tipo hash su attributo TipoSensore della relazione Sensore, non clusterizzato; scelto in quanto presente condizione di uguaglianza su di esso, e avente cardinalità del dominio dell'ordine di poche unità (ossia, potendo assumere pochi valori differenti, un indice hash permetterebbe di partizionare in base ai suoi valori le tuple in *buckets* tutti diversi).
- 3) Indice ad albero su attributo Valore di Informazione, clusterizzato; scelto in quanto condizione di richiesta di tipo a intervallo, quantità di tuple all'interno della relazione significativamente elevata, e dominio dell'attributo appartenente a un intervallo di numeri decimali (reali).







c) Rilevazione numero tuple e dimensionamenti tabelle coinvolte:

Per conoscere il dimensionamento delle tabelle presenti nel database dopo tutti gli inserimenti effettuati, è stata eseguita la seguente interrogazione:

```
SELECT N.nspname, C.relname, C.relpages, C.reltuples, C.relnatts,
C.relhas triggers
FROM pg_namespace N JOIN pg_class C ON N.oid = C.relnamespace
WHERE N.nspname = 'progetto52'
AND relname NOT LIKE '%\_%' OR relname IN ('tabella_log')
ORDER BY relpages DESC
```

nella quale è stata inserita una condizione specifica in modo da visualizzare solo le tabelle create appositamente per il progetto (senza quelle automaticamente create dal DBMS).

L'output della precedente interrogazione restituisce una tabella contenente per ogni relazione nel database il numero di pagine che occupa e il numero di tuple presenti:

| |  nspname name |  relname name |  relpages integer |  reltuples real |  relnatts smallint |  relhas triggers boolean |
|----|---|---|---|---|--|--|
| 1 | progetto52 | sensore | 342 | 40000 | 4 | true |
| 2 | progetto52 | rilevazione | 193 | 10000 | 8 | true |
| 3 | progetto52 | replica | 178 | 20000 | 3 | true |
| 4 | progetto52 | informazione | 148 | 20000 | 5 | true |
| 5 | progetto52 | collocazione | 89 | 20000 | 2 | true |
| 6 | progetto52 | tabella_log | 47 | 3004 | 5 | false |
| 7 | progetto52 | gruppo | 32 | 1000 | 8 | true |
| 8 | progetto52 | classe | 25 | 2000 | 6 | true |
| 9 | progetto52 | orto | 21 | 2000 | 8 | true |
| 10 | progetto52 | persona | 12 | 1000 | 5 | true |
| 11 | progetto52 | referente | 10 | 1000 | 3 | true |
| 12 | progetto52 | scuola | 8 | 1000 | 5 | true |
| 13 | progetto52 | progetto | 2 | 200 | 1 | true |
| 14 | progetto52 | specie | 1 | 100 | 3 | true |

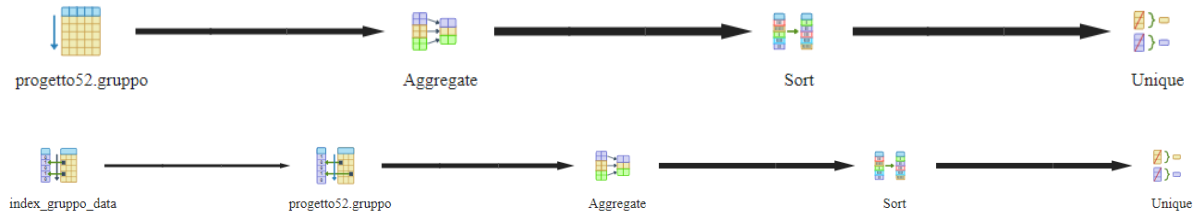
Dove

- nspname indica il nome dello schema;
- relname indica il nome della relazione di riferimento;
- relpages indica il numero di pagine occupate dalla tabella corrispondente;
- reltuples indica il numero di tuple contenute nella relazione;
- relnatts indica il numero di attributi di cui è composta la relazione;
- relhas triggers indica se la relazione è menzionata in un qualsiasi *trigger*.

d) Descrizione e visualizzazione di piani e tempi d'esecuzione:

(le immagini proposte riguardano il piano fisico e i tempi d'esecuzione prima e dopo la creazione degli indici scelti, rispettivamente)

1) Interrogazione 1:



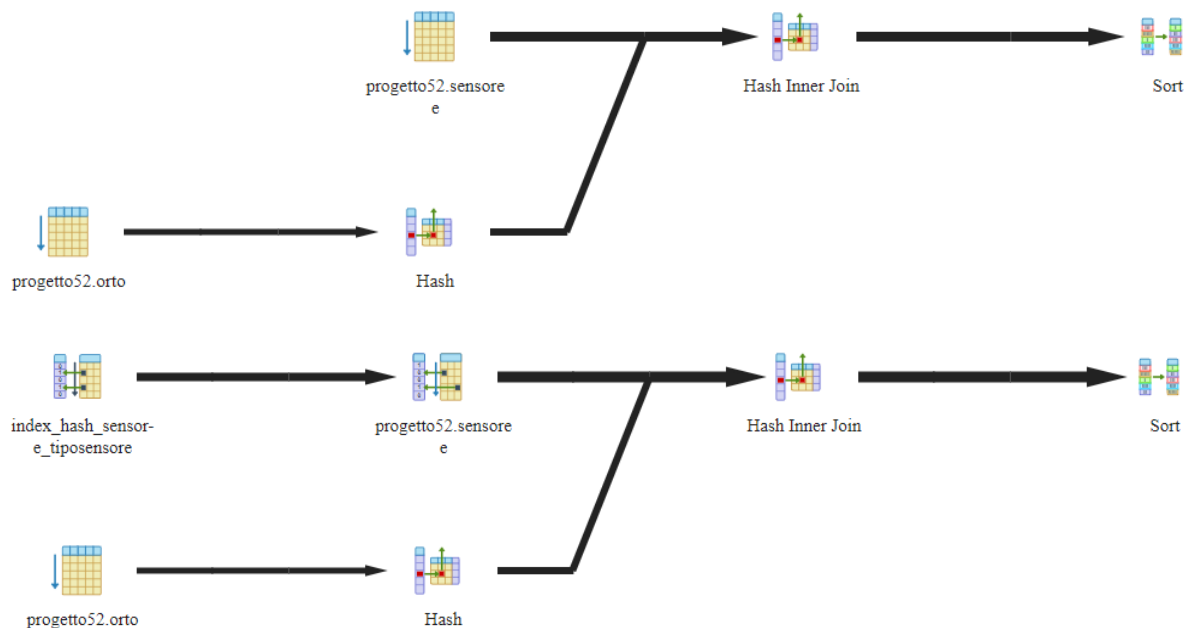
| # | Node | Timings | | Rows | | | |
|----|---|-----------|-----------|--------|--------|------|-------|
| | | Exclusive | Inclusive | Rows X | Actual | Plan | Loops |
| 1. | → Unique (cost=48.71..49.01 rows=40 width=22) (actual=0.15..0.155 rows=38 loops=1) | 0.004 ms | 0.155 ms | 1 1.06 | 38 | 40 | 1 |
| 2. | → Sort (cost=48.71..48.81 rows=40 width=22) (actual=0.15..0.152 rows=38 loops=1) | 0.043 ms | 0.152 ms | 1 1.06 | 38 | 40 | 1 |
| 3. | → Aggregate (cost=47.25..47.65 rows=40 width=22) (actual=0.106..0.109 rows=38 ...) | 0.015 ms | 0.109 ms | 1 1.06 | 38 | 40 | 1 |
| 4. | → Seq Scan on progetto52.gruppo as gruppo (cost=0..47 rows=50 width=22) (... Filter: ((gruppo.data >= '2006-09-01'::date) AND (gruppo.data <= '2007-08-31'::date))) Rows Removed by Filter: 949 | 0.094 ms | 0.094 ms | 1 1.02 | 51 | 50 | 1 |

| # | Node | Timings | | Rows | | | |
|----|---|-----------|-----------|--------|--------|------|-------|
| | | Exclusive | Inclusive | Rows X | Actual | Plan | Loops |
| 1. | → Unique (cost=40.43..40.73 rows=40 width=22) (actual=0.084..0.089 rows=38 loops=1) | 0.004 ms | 0.089 ms | 1 1.06 | 38 | 40 | 1 |
| 2. | → Sort (cost=40.43..40.53 rows=40 width=22) (actual=0.084..0.086 rows=38 loops=1) | 0.044 ms | 0.086 ms | 1 1.06 | 38 | 40 | 1 |
| 3. | → Aggregate (cost=38.97..39.37 rows=40 width=22) (actual=0.038..0.042 rows=38 loops=1) | 0.019 ms | 0.042 ms | 1 1.06 | 38 | 40 | 1 |
| 4. | → Bitmap Heap Scan on progetto52.gruppo as gruppo (cost=4.79..38.72 rows=50 width=... Recheck Cond: (((gruppo.data >= '2006-09-01'::date) AND (gruppo.data <= '2007-08-31'::date))) Heap Blocks: exact=11 | 0.014 ms | 0.024 ms | 1 1.02 | 51 | 50 | 1 |
| 5. | → Bitmap Index Scan using index_gruppo_data (cost=0..4.78 rows=50 width=0) (act... Index Cond: (((gruppo.data >= '2006-09-01'::date) AND (gruppo.data <= '2007-08-31'::date))) | 0.01 ms | 0.01 ms | 1 1.02 | 51 | 50 | 1 |

Prima: scansione sequenziale della tabella Gruppo, con filtro di tipo intervallo su attributo Data; esecuzione algoritmi per aggregazione (clausola `GROUP BY`), ordinamento (`ORDER BY`) e rimozione duplicati (`SELECT DISTINCT`).

Dopo: scansione indicizzata della tabella Gruppo, con condizione di indice su intervallo valori posti per attributo Data, consistente in una prima scansione dell'albero dell'indice creato in modo da ordinare le pagine e velocizzare la successiva scansione per il ritrovamento effettivo dei dati; esecuzione algoritmi per aggregazione, ordinamento e rimozione duplicati in modo analogo all'esecuzione senza indice.

2) Interrogazione 2:



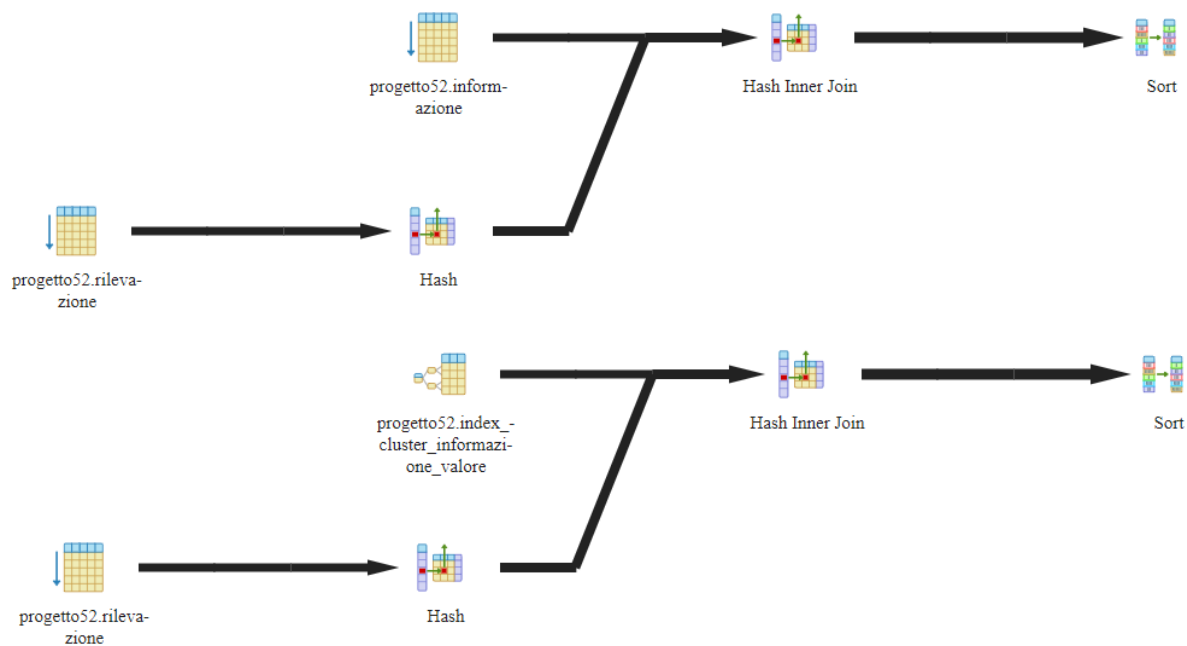
| # | Node | Timings | | Rows | | | |
|----|---|-----------|-----------|--------|--------|------|-------|
| | | Exclusive | Inclusive | Rows X | Actual | Plan | Loops |
| 1. | → Sort (cost=1534.49..1554.42 rows=7975 width=25) (actual=6.508..6.869 rows=7981 loops=1) | 2.524 ms | 6.869 ms | ↓ 1.01 | 7981 | 7975 | 1 |
| 2. | → Hash Inner Join (cost=66..1017.66 rows=7975 width=25) (actual=0.324..4.345 rows=7981 loops=1) Hash Cond: (sensore.idorto = orto.id) | 1.034 ms | 4.345 ms | ↓ 1.01 | 7981 | 7975 | 1 |
| 3. | → Seq Scan on progetto52.sensore as sensore (cost=0..842 rows=7975 width=18) (actual=0.307..0.307 rows=2000 loops=1) Filter: ((sensore.tiposensore)::text = 'umidita':text) Rows Removed by Filter: 32019 | 3.004 ms | 3.004 ms | ↓ 1.01 | 7981 | 7975 | 1 |
| 4. | → Hash (cost=41..41 rows=2000 width=11) (actual=0.307..0.307 rows=2000 loops=1) Buckets: 2048 Batches: 1 Memory Usage: 101 kB | 0.162 ms | 0.307 ms | ↑ 1 | 2000 | 2000 | 1 |
| 5. | → Seq Scan on progetto52.orto as orto (cost=0..41 rows=2000 width=11) (actual=0.0..0.0 rows=2000 loops=1) | 0.145 ms | 0.145 ms | ↑ 1 | 2000 | 2000 | 1 |

| # | Node | Timings | | Rows | | | |
|----|--|-----------|-----------|--------|--------|------|-------|
| | | Exclusive | Inclusive | Rows X | Actual | Plan | Loops |
| 1. | → Sort (cost=1484.43..1504.48 rows=8017 width=25) (actual=5.073..5.431 rows=7981 loops=1) | 2.527 ms | 5.431 ms | ↑ 1.01 | 7981 | 8017 | 1 |
| 2. | → Hash Inner Join (cost=412.13..964.58 rows=8017 width=25) (actual=0.708..2.904 rows=7981 loops=1) Hash Cond: (sensore.idorto = orto.id) | 1.071 ms | 2.904 ms | ↑ 1.01 | 7981 | 8017 | 1 |
| 3. | → Bitmap Heap Scan on progetto52.sensore as sensore (cost=346.13..788.34 rows=8017 width=18) (actual=0.347..0.347 rows=2000 loops=1) Recheck Cond: ((sensore.tiposensore)::text = 'umidita':text) Heap Blocks: exact=331 | 1.157 ms | 1.486 ms | ↑ 1.01 | 7981 | 8017 | 1 |
| 4. | → Bitmap Index Scan using index_hash_sensore_tiposensore (cost=0..344.13 rows=8017 width=11) (actual=0.347..0.347 rows=2000 loops=1) Index Cond: ((sensore.tiposensore)::text = 'umidita':text) | 0.329 ms | 0.329 ms | ↑ 1.01 | 7981 | 8017 | 1 |
| 5. | → Hash (cost=41..41 rows=2000 width=11) (actual=0.347..0.347 rows=2000 loops=1) Buckets: 2048 Batches: 1 Memory Usage: 101 kB | 0.187 ms | 0.347 ms | ↑ 1 | 2000 | 2000 | 1 |
| 6. | → Seq Scan on progetto52.orto as orto (cost=0..41 rows=2000 width=11) (actual=0.0..0.0 rows=2000 loops=1) | 0.16 ms | 0.16 ms | ↑ 1 | 2000 | 2000 | 1 |

Prima: scansione sequenziale tabelle Sensore e Orto, con filtro su tabella Sensore derivante da condizione posta nell'interrogazione; successivo Hash Inner Join (possibile poiché casistica equi-join anche senza indici specifici) grazie a un partizionamento di tipo hash eseguito preventivamente sulla tabella Orto (con utilizzo di 2048 *buckets*); successivo ordinamento (`ORDER BY`).

Dopo: scansione sequenziale tabella Orto e suo successivo partizionamento tramite funzione di hash, analogamente a quanto sopra; parallelamente, lettura tabella Sensore tramite utilizzo di indice hash con condizione su attributo TipoSensore posta dall'interrogazione richiesta; Hash Inner Join e algoritmo di ordinamento successivi in modalità analoghe alle precedenti.

3) Interrogazione 3:



| # | Node | Timings | | Rows | | | | Loops |
|----|---|-----------|-----------|--------|--------|-------|--|-------|
| | | Exclusive | Inclusive | Rows X | Actual | Plan | | |
| 1. | → Sort (cost=1014.18..1019.55 rows=2145 width=21) (actual=3.913..3.936 rows=573 loops=1) | 0.113 ms | 3.936 ms | 1 3.75 | 573 | 2145 | | 1 |
| 2. | → Hash Inner Join (cost=418.895..49 rows=2145 width=21) (actual=3.045..3.823 rows=573 L... Hash Cond: (informazione.idrilevazione = rilevazione.id) | 0.154 ms | 3.823 ms | 1 3.75 | 573 | 2145 | | 1 |
| 3. | → Seq Scan on progetto52.informazione as informazione (cost=0.448 rows=2145 width... Filter: ((informazione.tipoinformazione)::text = 'temperatura'::text)) Rows Removed by Filter: 19427 | 1.988 ms | 1.988 ms | 1 3.75 | 573 | 2145 | | 1 |
| 4. | → Hash (cost=293.293 rows=10000 width=12) (actual=1.682..1.682 rows=10000 loops... Buckets: 16384 Batches: 1 Memory Usage: 558 kB) | 0.867 ms | 1.682 ms | 1 1 | 10000 | 10000 | | 1 |
| 5. | → Seq Scan on progetto52.rilevazione as rilevazione (cost=0.293 rows=10000 width...) | 0.815 ms | 0.815 ms | 1 1 | 10000 | 10000 | | 1 |

| # | Node | Timings | | Rows | | | | Loops |
|----|---|-----------|-----------|--------|--------|-------|--|-------|
| | | Exclusive | Inclusive | Rows X | Actual | Plan | | |
| 1. | → Sort (cost=810.07..815.43 rows=2145 width=21) (actual=2.509..2.531 rows=573 loops=1) | 0.103 ms | 2.531 ms | 1 3.75 | 573 | 2145 | | 1 |
| 2. | → Hash Inner Join (cost=418.29..691.38 rows=2145 width=21) (actual=1.62..2.429 rows=57... Hash Cond: (informazione.idrilevazione = rilevazione.id) | 0.123 ms | 2.429 ms | 1 3.75 | 573 | 2145 | | 1 |
| 3. | → Index Scan using index_cluster_informazione_valore on progetto52.informazione as I... Filter: ((informazione.tipoinformazione)::text = 'temperatura'::text) Index Cond: (informazione.valore > '45'::numeric) Rows Removed by Filter: 5508 | 0.715 ms | 0.715 ms | 1 3.75 | 573 | 2145 | | 1 |
| 4. | → Hash (cost=293.293 rows=10000 width=12) (actual=1.591..1.591 rows=10000 loops... Buckets: 16384 Batches: 1 Memory Usage: 558 kB) | 0.731 ms | 1.591 ms | 1 1 | 10000 | 10000 | | 1 |
| 5. | → Seq Scan on progetto52.rilevazione as rilevazione (cost=0.293 rows=10000 width...) | 0.86 ms | 0.86 ms | 1 1 | 10000 | 10000 | | 1 |

Prima: scansione sequenziale tabelle Informazione e Rilevazione, con filtro sui dati della tabella Informazione derivanti dalla condizione dell'interrogazione posta (su due suoi attributi); successivo Hash Inner Join, grazie a un partizionamento preventivo della tabella inner Rilevazione (tramite utilizzo di 16384 *buckets*), possibile per casistica di equi-join; esecuzione algoritmo di ordinamento del risultato (`ORDER BY`).

Dopo: modalità per la maggior parte analoghe alle precedenti, con la differenza dell'utilizzo dell'indice ad albero clusterizzato per la scansione della tabella Informazione, con predicato di ricerca su attributo di clusterizzazione Valore e ulteriore filtro su seconda condizione posta dall'interrogazione su attributo Tipoinformazione.

- In tutti e 3 i casi, si può notare come il sistema abbia deciso di scegliere piani fisici che implementassero lo schema fisico proposto, andando effettivamente a migliorare i tempi di esecuzione.

10. Descrizione delle politiche di controllo dell'accesso

I ruoli proposti da implementare all'interno della base di dati sono stati realizzati secondo una gerarchia "a catena" tra di essi; in particolare:

- Gestore globale \geq Referente istituto
- Referente istituto \geq Referente scuola
- Referente scuola \geq Insegnante
- Insegnante \geq Studente

Questo in base alla seguente semantica:

- **Studente**: ruolo con meno privilegi, rappresenta uno studente della scuola che (se la scuola partecipa) prende parte alle attività inerenti il progetto botanico;
- **Insegnante**: ruolo rappresentante un docente all'interno di una scuola, con le necessità di valutare, organizzare e moderare l'operato degli studenti, anche relativamente all'eventuale progetto botanico;
- **Referente scuola**: rappresentante della scuola di riferimento, intesa come struttura fisica, con le necessità di poter gestire e aggiornare informazioni relative alle classi e alle altre infrastrutture all'interno della singola scuola;
- **Referente istituto**: rappresentante della scuola di riferimento, intesa come complesso di scuole dello stesso istituto comprensivo, con le necessità di dover gestire situazioni più generali e comuni alle strutture dello stesso istituto;
- **Gestore globale**: gestore totale e amministratore della base di dati; in quanto unico a detenere tutti i permessi su tutte le tabelle, è abilitato anche alla delegabilità del ruolo tramite opzione `GRANT OPTION`.

Nella tabella alla pagina seguente sono descritti schematicamente i privilegi concessi a ogni ruolo per ogni relazione contenuta nella base di dati.

I permessi concessi, proprio in virtù della gerarchia implementata, sono cumulativi: vengono schematizzati per ogni ruolo solo i nuovi privilegi ad esso concessi.

Legenda:

S permesso di SELECT
 I permesso di INSERT
 U permesso di UPDATE
 D permesso di DELETE

| Ruolo sottostante eredita permessi da | | studente | insegnante | referente scuola | referente istituto |
|--|-----------------|-------------------|-------------------------|---------------------------|------------------------|
| Ruolo → Tabella ↓ | studente | insegnante | referente scuola | referente istituto | gestore globale |
| classe | S | | I U D | | |
| collocazione | S | I U | D | | |
| gruppo | S | I U D | | | |
| informazione | S I U | D | | | |
| orto | S | | U | I D | |
| persona | | | S U | I D | |
| progetto | S | | | | I U D |
| referente | | S | | I U D | |
| replica | S | I U D | | | |
| rilevazione | S I U | D | | | |
| scuola | | | S | I U D | |
| sensore | S | | U | I D | |
| specie | S | | | | I U D |
| tabella_log | | | | | S I U D |
| riassunto biomonitoraggio (vista) | S | | | | |

- Sulla tabella “di servizio” `tabella_log`, nonostante sia possibile accedervi in qualsiasi modalità solamente ai detentori del ruolo di “gestore globale”, è necessario che sia possibile anche per gli utenti abilitati con i ruoli inferiori generare su di essa messaggi automatizzati all'accadimento di situazioni di warning o di errore: per questo alla definizione delle funzioni che potrebbero avere la necessità di fare ciò è stata aggiunta l'opzione `SECURITY DEFINER`, che permette proprio questo senza la necessità di concedere nessun altro permesso a nessun altro ruolo.
- Sulla vista `riassunto_biomonitoraggio` è possibile gestire solo il permesso di `SELECT` in quanto essendo comprensiva di funzioni di aggregazione, è impossibile effettuare qualsiasi altra operazione.