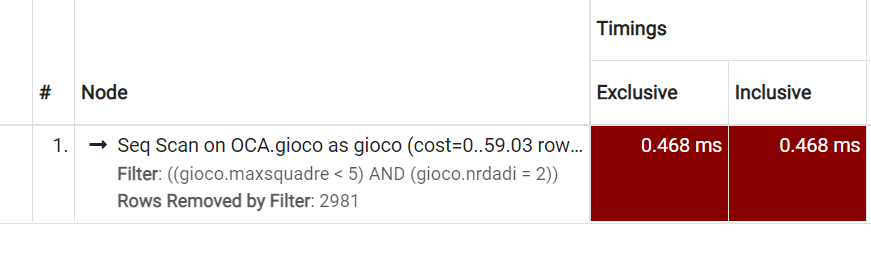
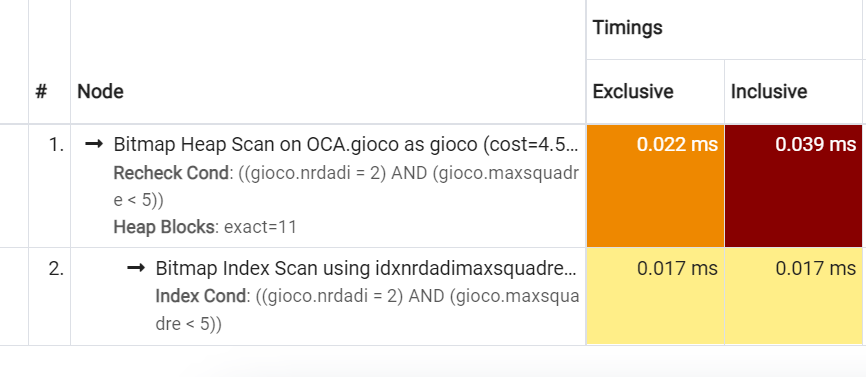
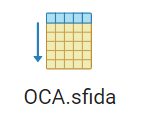
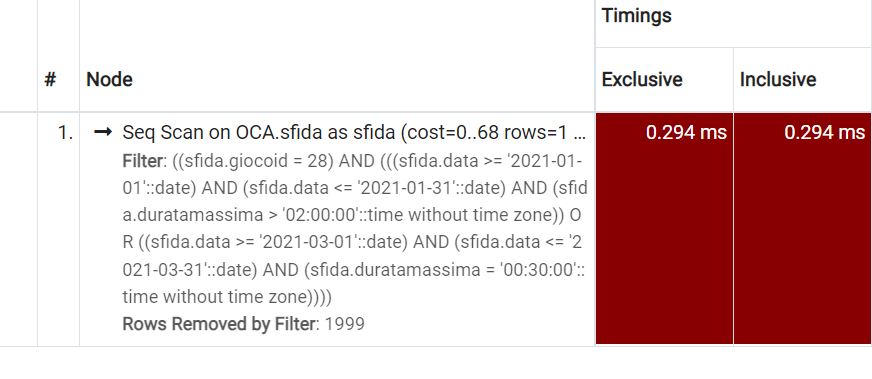
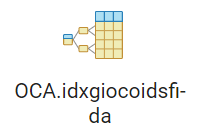
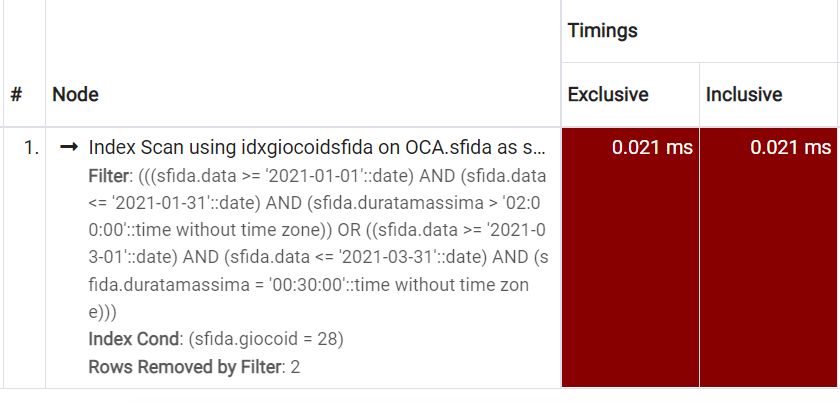
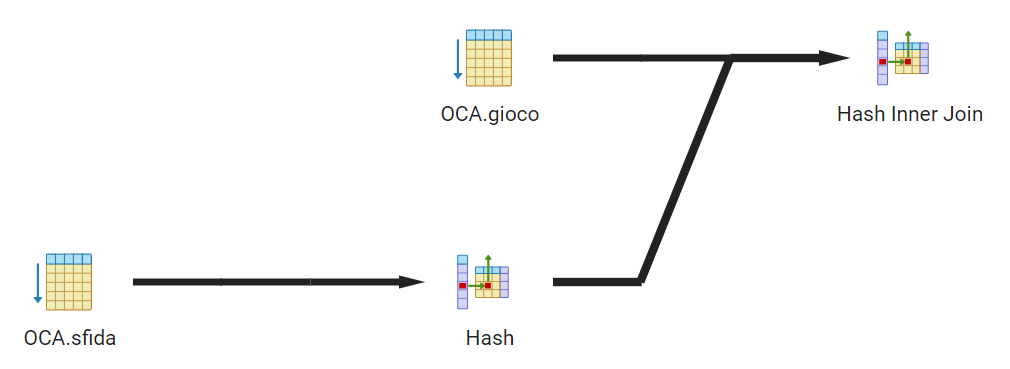
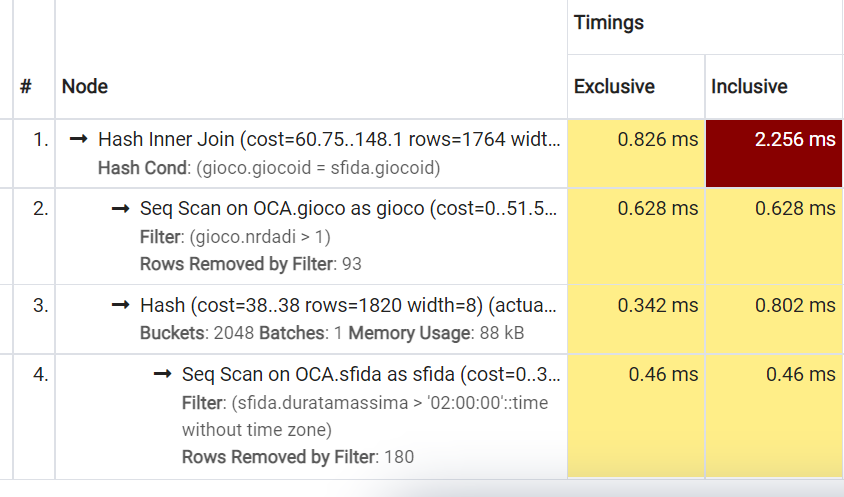
[D] Descrizione dei piani di esecuzione scelti dal sistema (prima e dopo la creazione dello schema fisico) per le interrogazioni contenute nel carico di lavoro e giustificazione di tale scelta sulla base dello schema fisico realizzato e della dimensione della base di dati.  
  
======Prima Query:  
Prima dello schema fisico:  
Il piano scelto dal sistema

  
  
  
  
Una scansione sequenziale con filtro sulle condizioni della query. Viene scelto questo approccio perché in assenza di indici non rimane altro che scorrere tutta la tabella sequenzialmente.  
  
Dopo lo schema fisico:  
  
  
Il tempo di esecuzione della query è passato da 0.47 circa ms a 0.04 circa ms.  
Il sistema sceglie di usare l’indice perché la tabella è più grande di una o poche pagine e il numero di accessi disco mediante indice è minore rispetto a quello con la scansione sequenziale. Minore perché   
i fattori di carico stimati durante la progettazione fisica sono vicini a zero.  
Dato che le condizioni sono più selettive il sistema ha bisogno di accedere a meno tuple, seguendo i puntatori delle foglie del B-+ tree, e quindi meno accessi a disco. Notare che il sistema non usa solamente l’indice dato che i fattori di carico non sono troppo piccoli, ma una struttura ausiliaria cioè una bitmap per rendere la query più efficiente.  
  
======Seconda Query  
Prima dello schema fisico:  
Piano scelto dal sistema:  
  
  
Senza indici il sistema non ha altra scelta che fare una scansione sequenziale.  
Dopo lo schema fisico:

  
  
Grazie all’indice siamo passati da 0.3 ms circa a 0.02 ms circa.  
Il sistema considera il nostro indice ordinato su GiocoId della tabella Sfida, come detto durante la progettazione fisica, perché la tabella è più grande di una o poche pagine e il fattore di carico F(GiocoId = 28) è molto vicino a 0. Ciò significa che la condizione di uguaglianza è molto selettiva e che quindi il numero di accessi disco seguendo i rid dell’indice è minore degli accessi con la scansione sequenziale.  
  
  
======Terza Query:  
  
  
  
E’ stato scelto l’hash join a presciendere dagli indici perché:  
- la condizione di join è una uguaglianza  
- non richiede né indici né tabelle ordinate per funzionare efficacemente  
- le due figlie del join sono tabelle grandi quindi il nested loop join non viene preso in considerazione  
- la hashtable ci sta tutta in memoria primaria  
- le due colonne coinvolte nell’hash join non sono ordinate quindi il merge join viene scartato, ma anche se lo ordinassimo con degli indici clusterizzati rispetto agli attributi di join la differenza non sarebbe molta dato che i due algoritmi hanno costo B(R) + B(S) con R e S argomenti del join.  
Mentre per quanto riguarda l’applicazione delle condizioni viene fatta comunque una scansione sequenziale perché i due fattori di carico sono molto vicini a 1 e quindi poco selettivi portando a non preferire gli indici ma la scansione sequenziale.