UNIVERSITÀ DEGLI STUDI DI PISA

CORSO DI LAUREA MAGISTRALE IN

DATA SCIENCE AND BUSINESS INFORMATICS

BASI DI DATI

PROF. GIORGIO GHELLI

PROF. GIOVANNA ROSONE

Attacco la Kamchatka!

Alessandro Andriotto 600975

Marco Ciompi 537856

Francesco Salerno 534622

20/6/2020

Anno Accademico 2019/2020

1. DOMINIO DEL DISCORSO

Un TORNEO ha un id identificativo, un montepremi, il numero iscritti. Ad ogni torneo possono essere iscritti più utenti, o nessuno nel caso il torneo sia appena creato.

Ogni torneo è composto da più partite, giocate su mappa standard.

Un torneo può essere in corso o terminato. Nel caso sia terminato interessa rilevare l'id del giocatore vincitore.

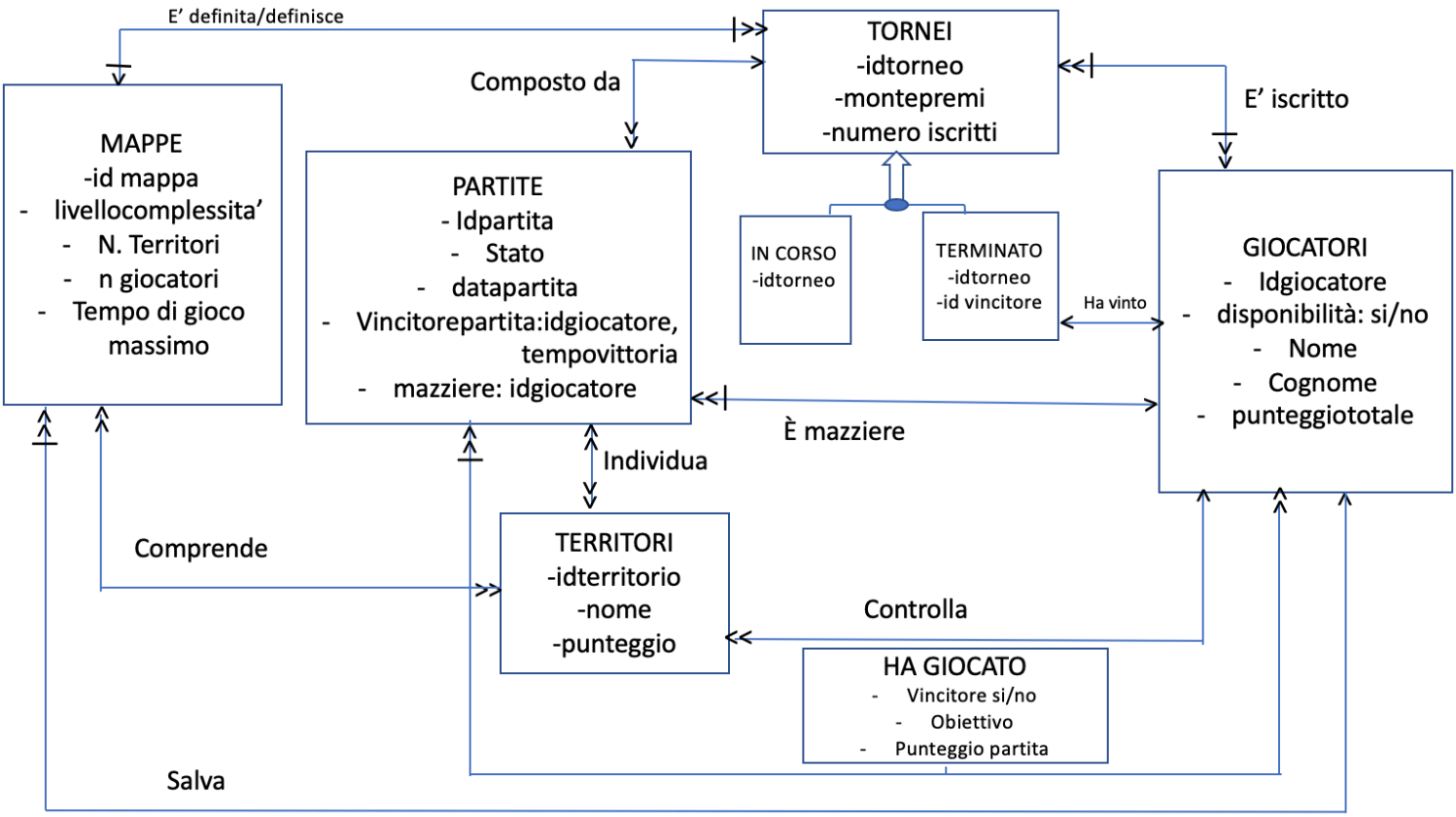
Una PARTITA ha un id, una data, uno stato, il vincitore e il mazziere. Il mazziere è unico ed è scelto random tra i giocatori della partita. Ogni giocatore può giocare più partite. Nel caso raggiunga l'obiettivo vince la partita ottenendo il bonus.

Una MAPPA è identificata da un id e dal livello di complessità, che ne determina il numero di territori, il numero di giocatori e il tempo massimo di gioco.

Di ogni GIOCATORE interessa l'id identificativo, nonché nome, cognome. Un giocatore deve dare la propria disponibilità a giocare. Ad ogni giocatore è assegnato un punteggio cumulativo in base ai territori conquistati durante le partite. Se il giocatore raggiunge l'obiettivo prima della scadenza del tempo massimo di gioco concesso dalla mappa, accede ad un bonus di 50 punti. Ogni giocatore può salvare una nuova configurazione di mappa. Di ogni giocatore interessa rilevare se è stato vincitore o meno della partita, e il suo obiettivo.

I TERRITORI sono compresi nella mappa e ogni territorio ha un id, un nome e un punteggio assegnato. Ogni territorio di una mappa è controllato da un giocatore.

2. SCHEMA CONCETTUALE



VINCOLI NON CATTURATI GRAFICAMENTE

- Il giocatore che vince entro il tempo limite della partita si vede accreditati 50 punti in più per il raggiungimento dell'obiettivo e lo stato della partita cambia in «terminata».
- In ogni mappa devono comunque essere presenti i due territori «Kamchatka» e «Čita».
- Può giocare la partita solo chi alla data della partita stessa è disponibile
- Chi fa il mazziere in una partita non gioca. All'interno di ogni torneo i giocatori avranno giocato lo stesso numero di partite.
- Alla fine di ogni torneo il punteggio di ogni giocatore è azzerato
- OSSERVAZIONI:
- Un giocatore deve essere obbligatoriamente registrato.

VINCOLI

VINCITORIPARTITE(Idpartita PRIMARYKEY, FOREIGNKEY;
vincitore PRIMARYKEY)

PARTITE(Idpartita PRIMARYKEY
Mazziere FOREIGN KEY
Idtorneo FOREIGN KEY
Id mappa FOREIGN KEY
stato NOT NULL

PARTITEGIOCATORI(Idpartita PRIMARY KEY, FOREIGN KEY
IdGiocatore PRIMARYKEY, FOREIGN KEY)

TORNEI(Id torneo PRIMARY KEY,
Vincitoretorneo FOREIGN KEY
Numeroiscritti NOT NULL)

TORNEIGIOCATORI(Idtorneo PRIMARY KEY FOREIGN KEY,
Idgiocatore PRIMARYKEY FOREIGNKEY)

GIOCATORI (Idgiocatore PRIMARY KEY,
Nome NOT NULL
Cognome NOT NULL
Disponibilità NOT NULL
Punteggiototale NOTNULL)

MAPPE(Idmappa PRIMARYKEY
Livellocomplessità PRIMARYKEY
N-territori NOT NULL
Tempodigiocomassimo CHECK (<= 90 minuti)
n giocatori CHECK (2<n<6)

MAPPETERRITORI(Id mappa PRIMARYKEY FOREIGN KEY,

Id territorio PRIMARYKEY FOREIGN KEY)

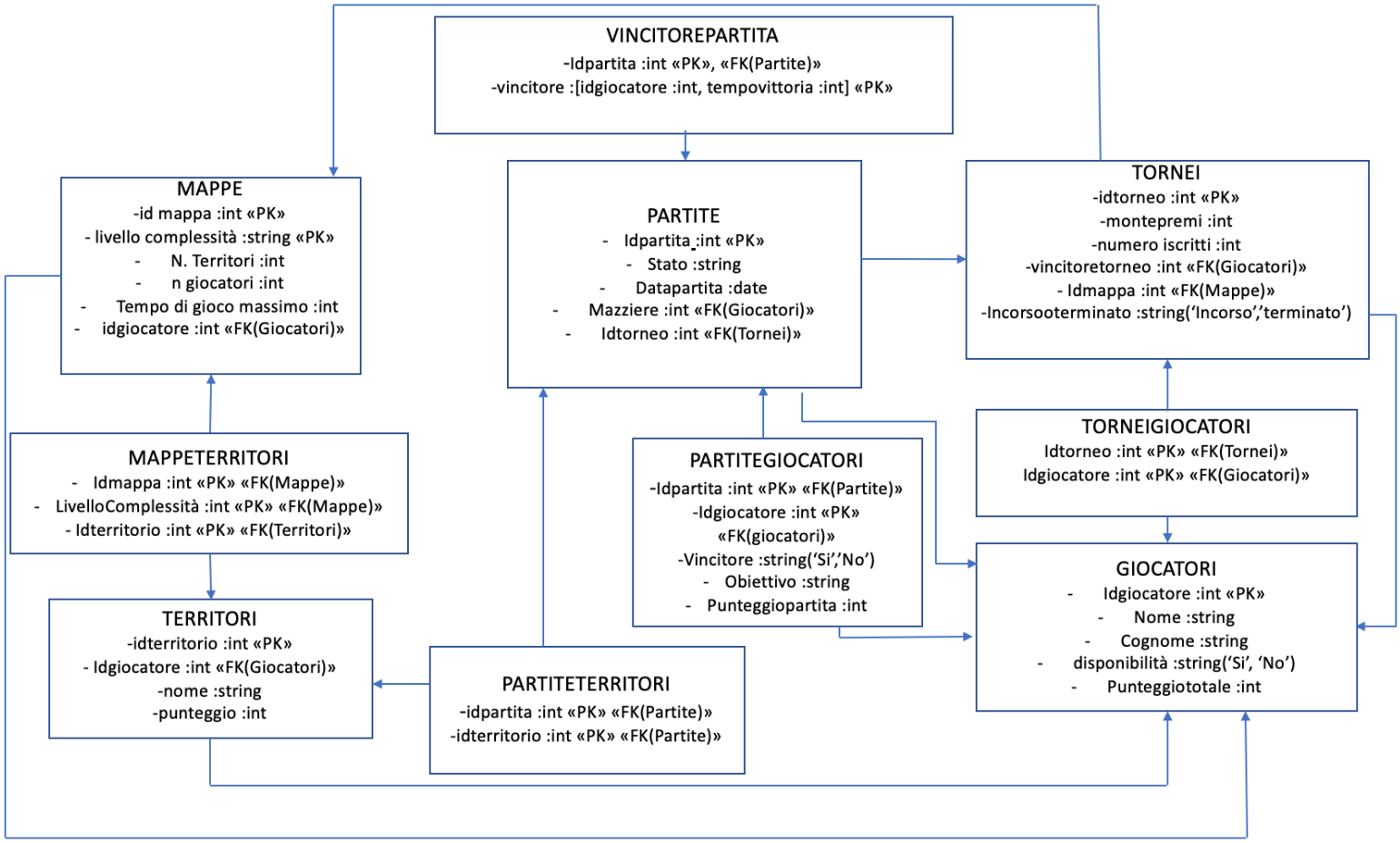
TERRITORI (Id territorio PRIMARYKEY

Idgiocatore FOREIGN KEY

Nome (condizione su Kamchatka e Čita)

punteggio NOT NULL)

3. SCHEMA RELAZIONALE



SCHEMA RELAZIONALE IN FORMATO TESTUALE

VINCITORIPARTITE(Idpartita* , vincitore)

PARTITE(Idpartita, Mazziere*, Idtorneo*, stato, datapartita)

PARTITEGIOCATORI(Idpartita*, Idgiocatore*, vincitore, obiettivo, punteggiopartita)

TORNEI (Idtorneo, vincitoretorneo*, idmappa*, montepremi, numero iscritti, incorsooterminato)

TORNEIGIOCATORI(Idtorneo*, Idgiocatore*)

GIOCATORI (Idgiocatore, nome, cognome, disponibilità, punteggiototale)

MAPPE(Idmappa, Livellocomplessità, N.territori, tempodigiocomassimo, ngiocatori,idgiocatore*)

MAPPETERRITORI(Idmappa*, Idterritorio*,livellocomplessità*)

TERRITORI(Idterritorio, Idgiocatore*, nome, punteggio)

PARTITETERRITORI(idpartita*,idterritorio*)

DIPENDENZE FUNZIONALI

MAPPE (IdMappA, LivelloComplessità → N.territori, N.giocatorI, Tempomassimodigioco ,Idgiocatore)

TERRITORI (Idterritorio→Idgiocatore,nome,punteggio)

MAPPETERRITORI ()

PARTITE (Idpartita→ Stato, DataPartita,Mazziere,Idtorneo)

TORNEIGIOCATORI()

PARTITETERRITORI()

PARTITEGIOCATORI(Idpartita,Idgiocatotore →Vincitore,Obiettivo,PunteggioPartita)

GIOCATORI (Idgiocatore→,Nome,Cognome,Disponibilità,Punteggiototale)

TORNEI (idtorneo → montepremi, numero iscritti, vincitoretorneo, Idmappa , Incorsooterminato)

VINCITOREPARTITA ()

LE DIPENDENZE FUNZIONALI RISPETTANO LA FORMA NORMALE DI BOYCE CODD. POICHE' PER OGNI DIPENDENZA FUNZIONALE X→Y L'X FA PARTE DI UNA CHIAVE.

4. INTERROGAZIONI IN SQL

a) uso di proiezione, join e restrizione
" Trovare le partite (idpartita) dei tornei con più di 10 iscritti"

```
SELECT  P.Idpartita
FROM  Partite P, Tornei T
WHERE   P.Idtorneo=T.Idtorneo  AND  T.Numeroiscritti >10
```

b) uso di group by con having, where e sort
"trovare id torneo dei tornei con piu di 5 partite terminate. ordinare per numero partite decrescente."

```
SELECT Idtorneo, COUNT (*) AS Numeropartite
FROM Partite
WHERE Stato = ''Terminato''
GROUPBY Idtorneo
HAVING Numeropartite >5
ORDER BY Numeropartite DESC
```

c) uso di join, group by con having e where
"Trovare gli obiettivi raggiunti più di 5 volte da giocatori con punteggio complessivo sotto i 200 punti e il numero di successi per ogni obiettivo"

```
SELECT PG.Obiettivo, COUNT(*) AS Numerosuccessi
FROM PartiteGiocatori PG, Giocatori G
WHERE PG.Vincitore = 'Si' AND G.Punteggiototale < 200 AND
PG.IdGiocatore = PG.IdGiocatore
GROUPBY PG.Obiettivo
HAVING Numerosuccessi > 5
```

d) uso di select annidata con quantificazione esistenziale
"Trovare Nome e cognome dei giocatori che hanno vinto almeno una partita"

```
SELECT G.Nome, G.Cognome
FROM Giocatori G
WHERE EXIST (SELECT *
                FROM GiocatoriPartite GP
                 WHERE G.Idgiocatore = GP.Idgiocatore AND GP.Vincitore = 'Si')
```

e) uso di select annidata con quantificazione universale
"Trovare il punteggio totale dei giocatori che in un torneo non hanno mai posseduto il territorio «Kamchatka»"

```
SELECT  G.PunteggioTotale
FROM  Giocatori G
WHERE NOT EXIST (SELECT *
                 FROM Territori T
                WHERE T.IdGiocatore = G.IdGiocatore AND T.Nome = 'Kamchatka')
```

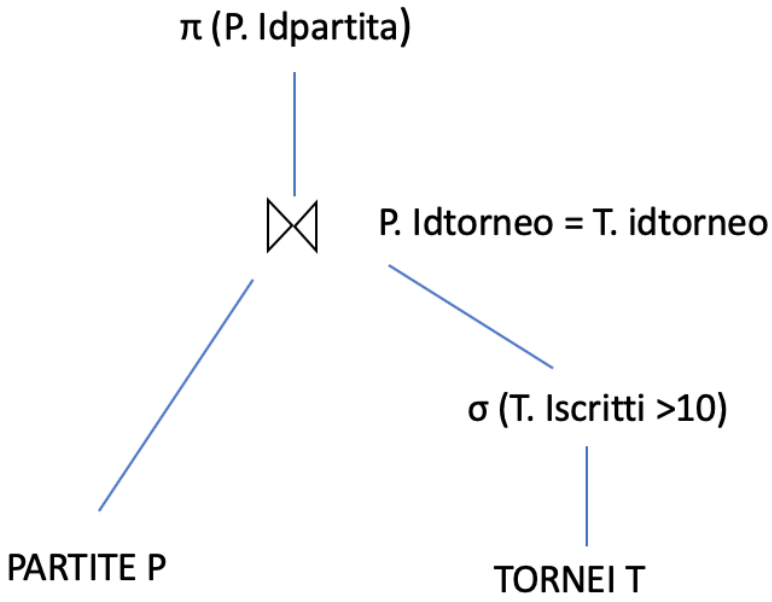
f) uso di subquery di confronto quantificato usando una subquery
"Trovare i giocatori il cui punteggio totale è maggiore della somma dei punteggi dei territori posseduti dal giocatore stesso (trovare i giocatori che hanno ricevuto almeno un bonus, tramite il sistema punteggi)"

```
SELECT G.IdGiocatore
FROM Giocatore G
WHERE G.Punteggiototale > (SELECT SUM(T.Punteggio)
                           FROM Territori T
                           WHERE G.IdGiocatore = T.IdGiocatore
                           GROUPBY T.IdGiocatore)
```

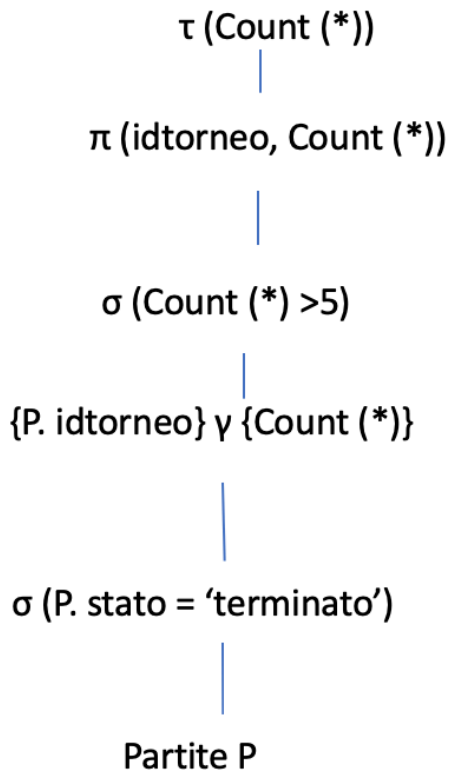
5. PIANI DI ACCESSO

Piani di accesso logico delle query a,b,c

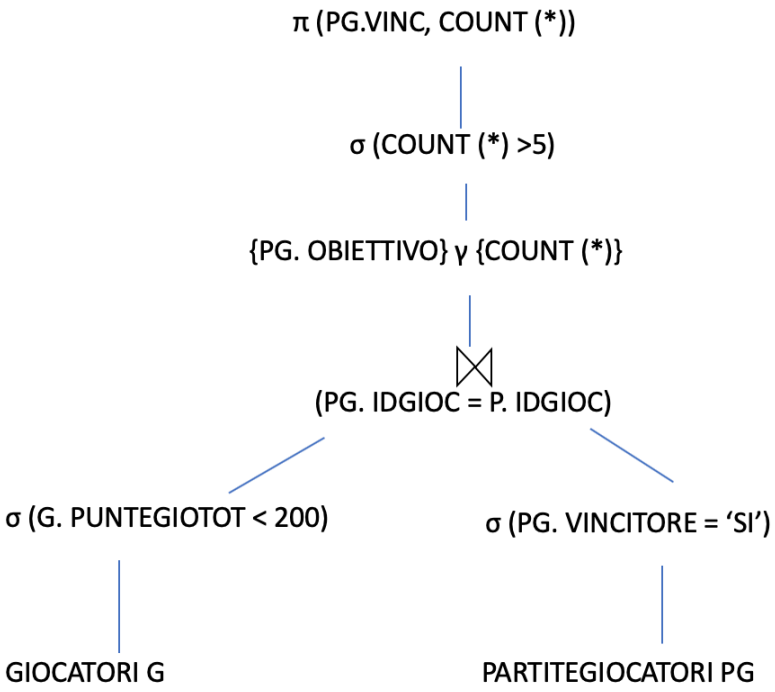
a)



b)

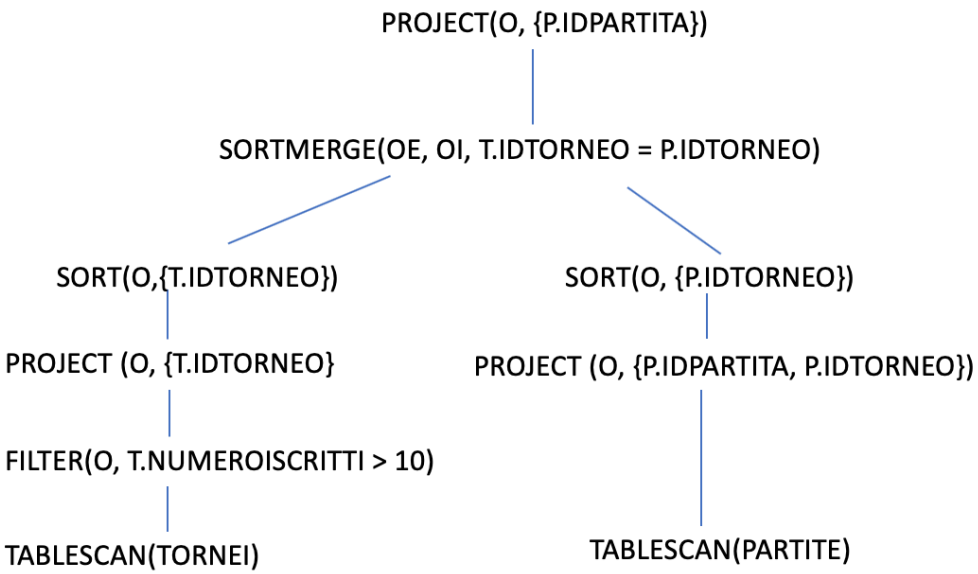


c)



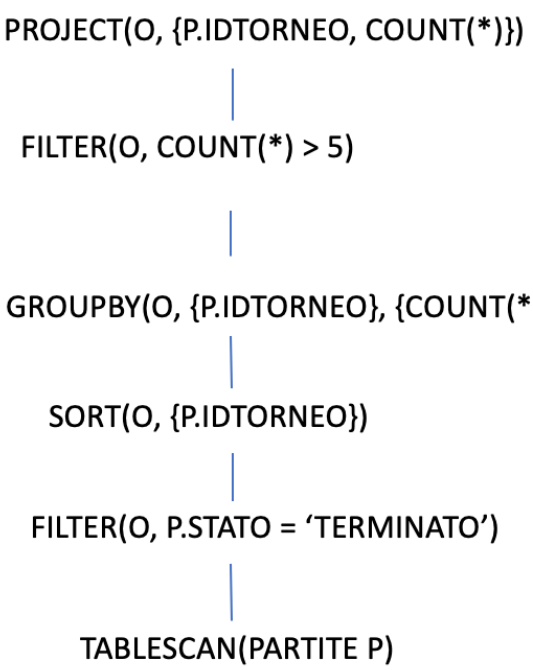
Piani di accesso fisici senza indici

a)

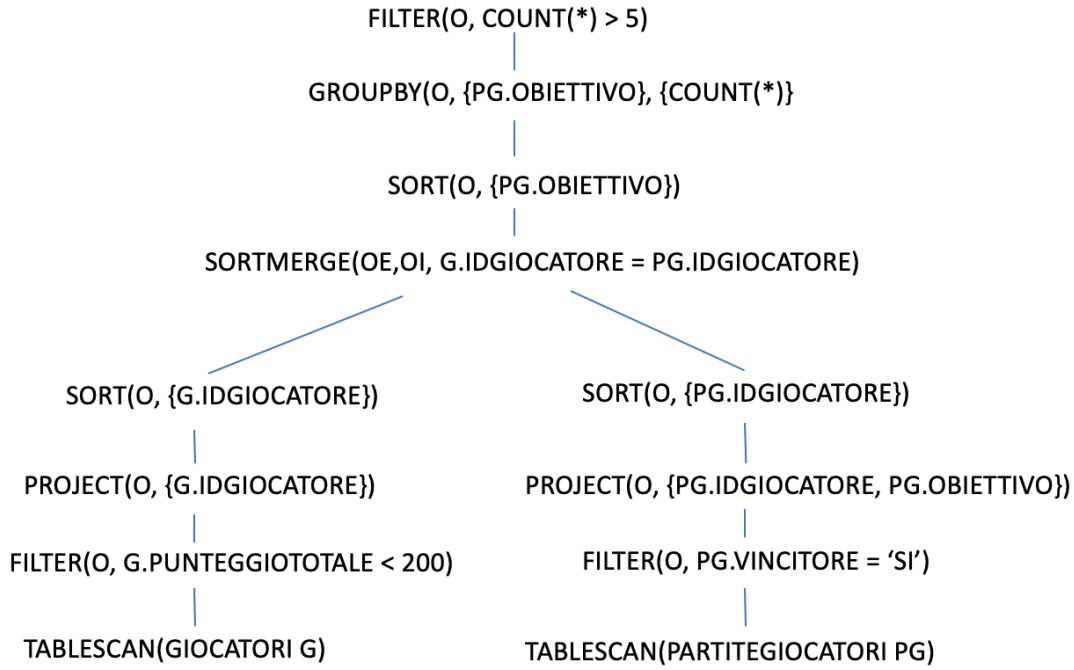


b)

Il sort prima del groupby deve essere effettuato in quanto l'operatore groupby si aspetta record ordinati sull'attributo di raggruppamento.



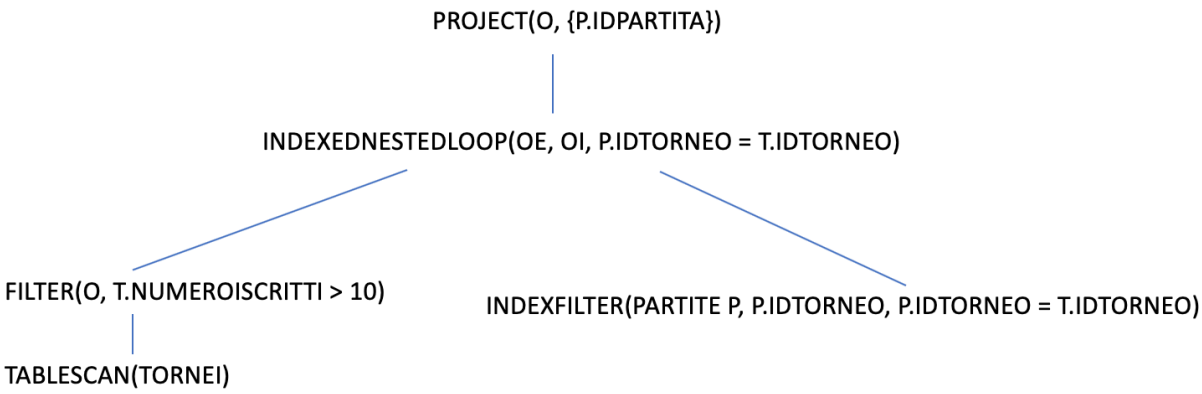
c)



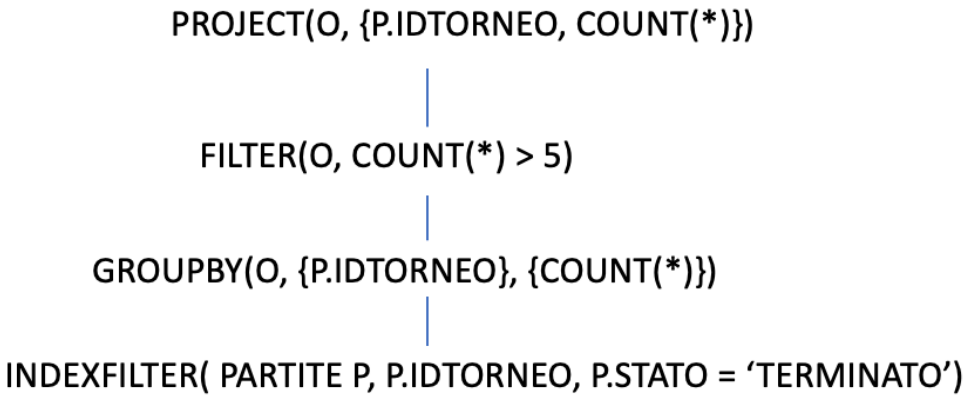
Il sort prima del groupby è necessario perché il raggruppamento viene effettuato in un attributo diverso rispetto a quello di giunzione.

Piani di accesso fisici con indici

a)

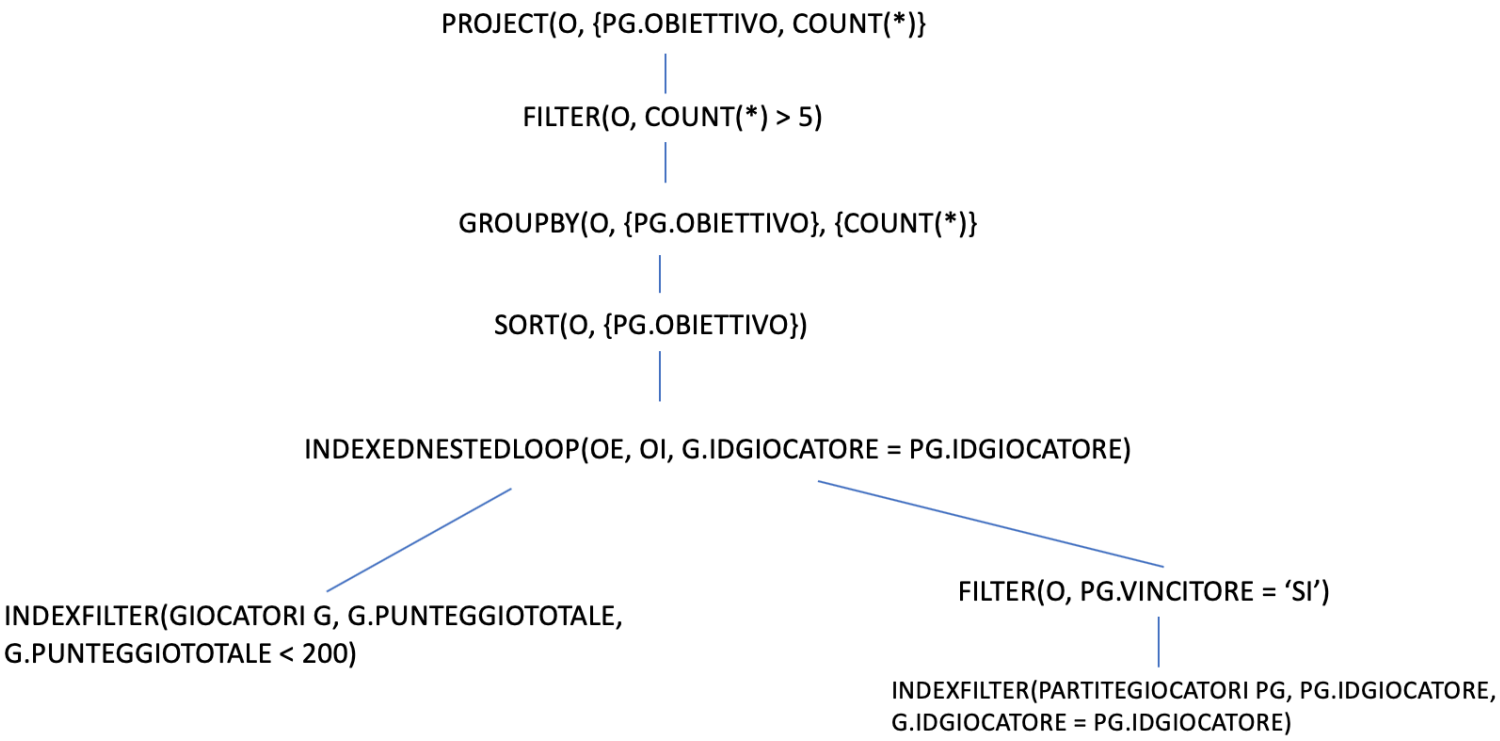


b)



Il sort prima del groupby non è necessario in quanto essendo il raggruppamento effettuato sullo stesso attributo in cui è posizionato l'indice della indexfilter, questo risulta già ordinato.

c)



Il sort prima del groupby è necessario in quanto la giunzione è effettuata ed ordinata rispetto ad un altro attributo della groupby.