Progetto di Basi di Dati

Calvo Giulia 544434
Curci Giulia 598202
Di Luzio Francesco 596212
Rossi Elvis 561394

June 25, 2021

Contents

1 Descrizione del dominio

La fabbrica di marchingegni è interessata a creare un sistema che tenga sotto controllo quelli che sono gli eventi riguardanti la produzione dei propri marchingegni, la loro relativa vendita e la vita professionale dei dipendenti. Abbiamo deciso di cominciare la nostra analisi focalizzandoci sulle persone che fanno parte della vita dell'azienda.

Per ogni persona interessa il nome, il cognome, il numero di telefono, il domicilio e il codice fiscale. La specializzazione effettuata divide questa categoria in clienti e dipendenti, che sono una partizione. I clienti devono essersi registrati per poter effettuare acquisti e ci interessa sapere la data in cui è avvenuta la registrazione. Per i dipendenti ci interessa la data in cui sono stati assunti e il ruolo che hanno all'interno della fabbrica. Il ruolo può assumere una tra quattro posizioni predefinite, ovvero: Capoprogetto, Operaio, Addetto alle vendite, Gestore delle risorse umane. Per ogni dipendente vengono registrate le presenze: nello specifico la data, l'ora di entrata e l'ora di uscita.

Ogni dipendente sottoscrive mensilmente un contratto, basato sulle ore lavorative e la paga. Tutti i contratti vengono gestiti tramite uno storico dove vengono registrate la data di inizio e la data di fine contratto. Un contratto viene approvato da un gestore delle risorse umane e il contratto personale di un gestore delle risorse umane dev'essere approvato da un altro gestore. Un dipendente non ha due o più contratti attivi nello stesso mese.

Più operai si occupano di costruire marchingegni, mentre un solo capoprogetto registra il numero di serie. Ogni marchingegno costruito è basato su un modello di un marchingegno di cui si conoscono il nome e il prezzo.

Per quanto riguarda i clienti, prima di acquistare un marchingegno, devono aver effettuato una prenotazione della quale devono essere registrate la data di consegna ultima, il prezzo totale dei marchingegni selezionati e l'ora in cui è stata effettuata la prenotazione per stabilire le 48 ore entro le quali è possibile richiedere la cancellazione dell'ordine gratuitamente. La prenotazione dev'essere visionata da un addetto alle vendite; una volta approvata, quest'ultimo deve registrare l'importo versato dal cliente che deve essere pari al 40% del prezzo totale.

2 Schema concettuale

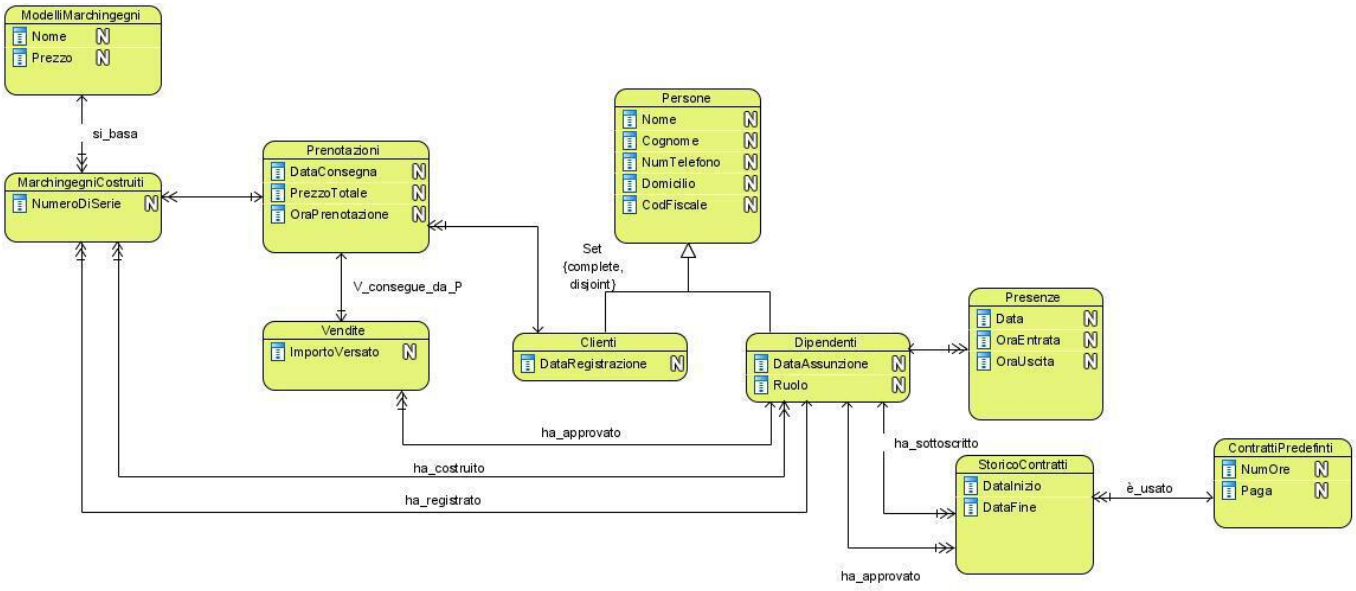


Figure 1: Schema concettuale

StoricoContratti:
Non esiste a, b appartenenti a StoricoContratti tale che a.IdPersona = b.IdPersona && (data(a) $\cap$ data(b)

$\neq \emptyset$).

MarchingegniCostruiti:
Il Dipendente che ha_registrato deve avere ruolo Capoprogetto. I Dipendenti che ha_costruito devono avere ruolo Operaio.

Vendite:
Il Dipendente che ha_approvato deve avere ruolo AddettoVendite.

StoricoContratti:
Il Dipendente che ha_approvato deve avere ruolo GestoreRisorseUmane. Il Dipendente che ha_sottoscritto un Contratto nello StoricoContratti non deve avere ha_approvato come stessa persona.

3 Schema logico relazionale

Come scelta progettuale abbiamo deciso di introdurre degli Id come chiavi per ogni tabella, perchè nei casi in cui c'era una chiara scelta per la chiave, come NumeroDiSerie in MarchingegniCostruiti, è più semplice l'implementazione e comunque senza ridondanze per BCNF. Negli altri casi era necessaria.

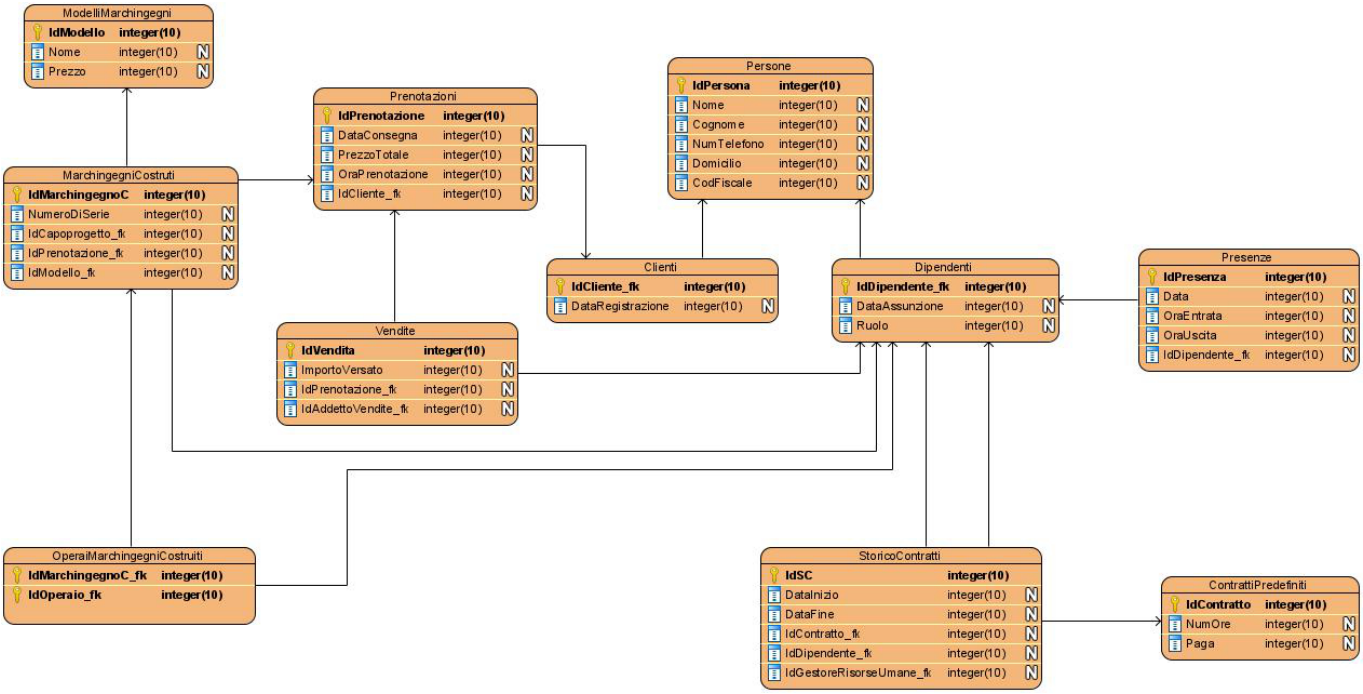


Figure 2: Schema logico relazionale

Abbiamo usato _fk *per simboleggiare una chiave esterna dato che il programma non ci permetteva di usare* *.

Schema logico relazionale testuale:

ModelliMarchingegni(IdModello, Nome, Prezzo)
MarchingegniCostruiti(IdMarchingegnoC, NumeroDiSerie, IdCapoprogetto*, IdPrenotazione*, IdModello*)
OperaiMarchingegniCostruiti(IdMarchingegnoC*, IdOperaio*)
Prenotazioni(IdPrenotazione, DataConsegna, PrezzoTotale, OraPrenotazione, IdCliente*)
Vendite(IdVendita, ImportoVersato, IdPrenotazione*, IdAddettoVendite*)
Persone(IdPersona, Nome, Cognome, NumTelefono, Domicilio, CodFiscale)
Clienti(IdCliente*, DataRegistrazione)
Dipendenti(IdDipendente*, DataAssunzione)
Presenze(IdPresenza, Data, OraEntrata, OraUscita, IdDipendente*)
StoricoContratti(IdSC, DataInizio, DataFine, IdContratto*, IdDipendente*, IdGestoreRisorseUmane*)
ContrattiPredefiniti(IdContratto, NumOre, Paga)

Vicoli:

ModelliMarchingegni:
IdModello → Nome,Prezzo

MarchingegniCostruiti:
IdMarchingegnoC → NumeroDiSerie, IdCapoprogetto*, IdPrenotazione*, IdModello*
NumeroDiSerie → IdMarchingegnoC, IdPrenotazione*, IdModello*

Prenotazioni:
IdPrenotazione → DataConsegna, IdCliente*, PrezzoTotale, OraPrenotazione

Vendite:
IdVendita → ImportoVersato, IdPrenotazione*, IdAddettoVendite*
IdPrenotazione* → IdVendita, ImportoVersato, IdAddettoVendite*

Persone:
IdPersona → Nome, Cognome, NumTelefono, Domicilio, CodFiscale

Presenze:
IdPresenza → Data, OraEntrata, OraUscita, IdDipendente*
Data, OraEntrata, IdDipendete* → OraUscita
Data, OraUscita, IdDipendete* → OraEntrata
Data, OraEntrata, OraUscita, IdDipendente* → IdPresenza

StoricoContratti:
IdSC → DataInizio, DataFine, IdContratto*, IdDipendente*, IdGestoreRisorseUmane*
DataInizio, DataFine, IdContratto*, IdDipendente* → IdGestoreRisorseUmane*, IdSC

ContrattiPredefiniti:
IdContratto → NumOre, Paga

Le dipendenze rispettano tutte la BCNF perché tutti gli R$\langle$T, F$\rangle$ vale che per ogni X→A in F non banale, X è una superchiave.

F non è una copertura canonica, dato che ci sono alcuni vincoli ridondanti in MarchingegniCostruiti, Vendite e StoricoContratti.

4 Interrogazioni in SQL

a) "L'id di prenotazione, il prezzo totale e il prezzo versato di tutte le prenotazioni con data di consegna oggi"

```
SELECT P.IdPrenotazione, P.PrezzoTotale, V.ImportoVersato
FROM Prenotazioni P
  JOIN Vendite V ON (P.IdPrenotazione = V.IdPrenotazione_fk)
WHERE P.DataConsegna = CURDATE()
```

b) "Prendere l'id del capoprogetto e la quantità di progetti registrati tali che abbia più di 5 progetti e il prezzo dei singoli progetti sia sopra 10000, in ordine crescente per numero di progetti"

```
SELECT MC.IdCapoprogetto_fk, COUNT(MC.IdMarchingegnoC) as NumeroProgetti
FROM ModelliMarchingegni MM
  JOIN MarchingegniCostruiti MC ON (MM.IdModello = MC.IdModello_fk)
WHERE MM.Prezzo > 10000
GROUP BY MC.IdCapoprogetto_fk
HAVING COUNT(MC.IdMarchingegnoC) > 5
ORDER BY NumeroProgetti
```

c) "L'idCliente, il nome, il cognome di tutti i clienti che abbiano fatto acquisti, con data di consegna antecedente al 1 maggio 2020, in media sopra a 20000"

```
SELECT C.IdCliente_fk, Pe.Nome, Pe.Cognome
FROM Persone Pe
  JOIN Clienti C ON (Pe.IdPersona = C.IdCliente_fk)
  JOIN Prenotazioni Pr ON (C.IdCliente_fk = Pr.IdCliente_fk)
WHERE Pr.DataConsegna < '2020-05-01'
GROUP BY C.IdCliente_fk, Pe.Nome, Pe.Cognome
HAVING AVG(Pr.PrezzoTotale) > 20000
```

d) "Il seriale di tutti i marchingegni a cui ha lavorato almeno un operaio a cui è scaduto il contratto il 23-01-2014"

```
SELECT MC.NumeroDiSerie
FROM MarchingegniCostruiti MC
WHERE EXISTS (
  SELECT *
  FROM OperaiMarchingegniCostruiti OMC
    JOIN Dipendenti D ON (OMC.IdOperaio_fk = D.IdDipendente_fk)
    JOIN StoricoContratti SC ON (D.IdDipendente_fk = SC.IdDipendente_fk)
  WHERE MC.IdMarchingegnoC = OMC.IdMarchingegnoC_fk
    AND SC.DataFine = '2014-01-23'
)
```

e) "Restituire l'id di tutti gli operai che hanno lavorato solo a marchingegni di costo superiore a 10000"

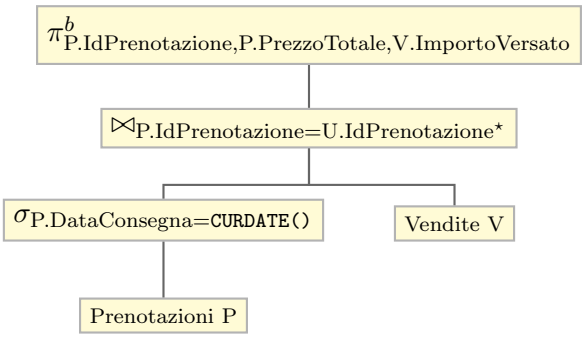
```
SELECT OMC.IdOperaio_fk
FROM OperaiMarchingegniCostriti OMC
WHERE NOT EXISTS (
  SELECT *
  FROM  MarchingegniCostruiti MC
    JOIN ModelliMarchingegni MM ON (MC.IdModello_fk = MM.IdModello)
  WHERE OMC.IdMarchingegnoC_fk = MC.IdMarchingengoC
    AND MM.Prezzo <= 10000
)
```

f) "Tutti i dipendenti che hanno partecipato alla costruzione di più di due marchingegni e con paga minore di 1000"

```sql
SELECT D1.IdPersona_fk
FROM Dipendenti D1
  JOIN StoricoContratti SC ON (D.IdDipendente_fk = SC.IdDipendente_fk)
  JOIN ContrattiPredefiniti CP ON (SC.IdContratto_fk = CP.IdContratto)
WHERE CP.Paga < 1000
  AND 2 < (
    SELECT COUNT(*)
    FROM Dipendenti D2
      JOIN OperaiMarchingegniCostruiti OMC ON (D2.IdDipendente_fk = OMC.
         ↪ IdOperaio_fk)
    WHERE D2.IdPersona_fk = D1.IdPersona_fk
  )
```

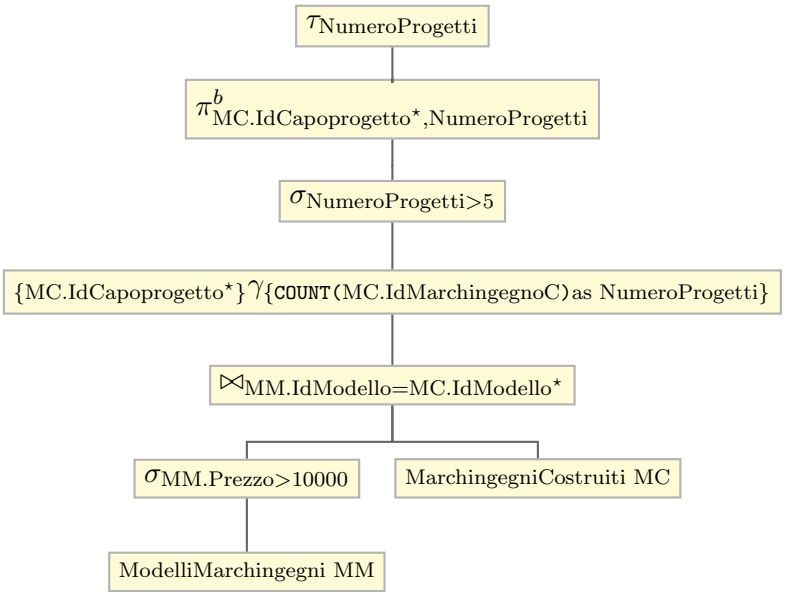
5 Piani di accesso

I) a)



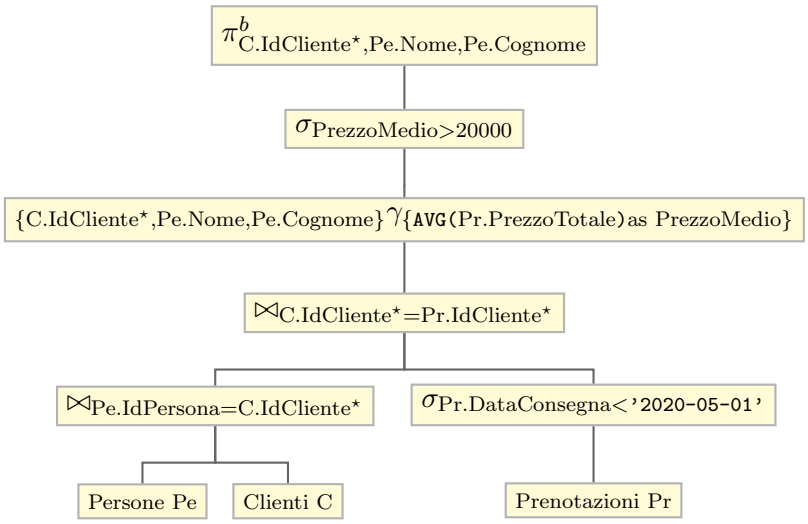
Abbiamo fatto commutare la giunzione con la restrizione.

b)



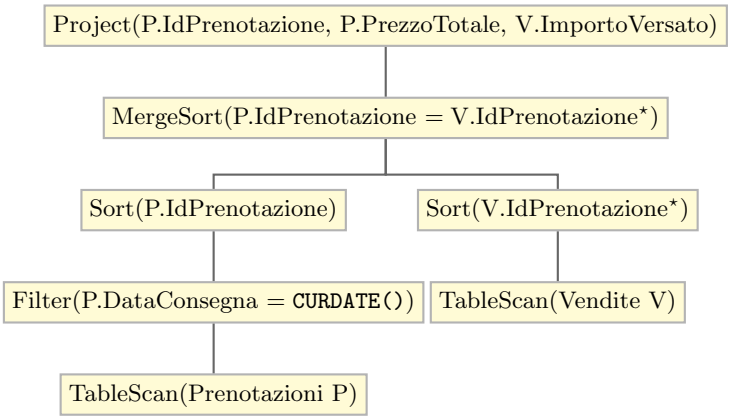
Abbiamo fatto commutare la giunzione con la restrizione MM.Prezzo $>$ 10000.

c)

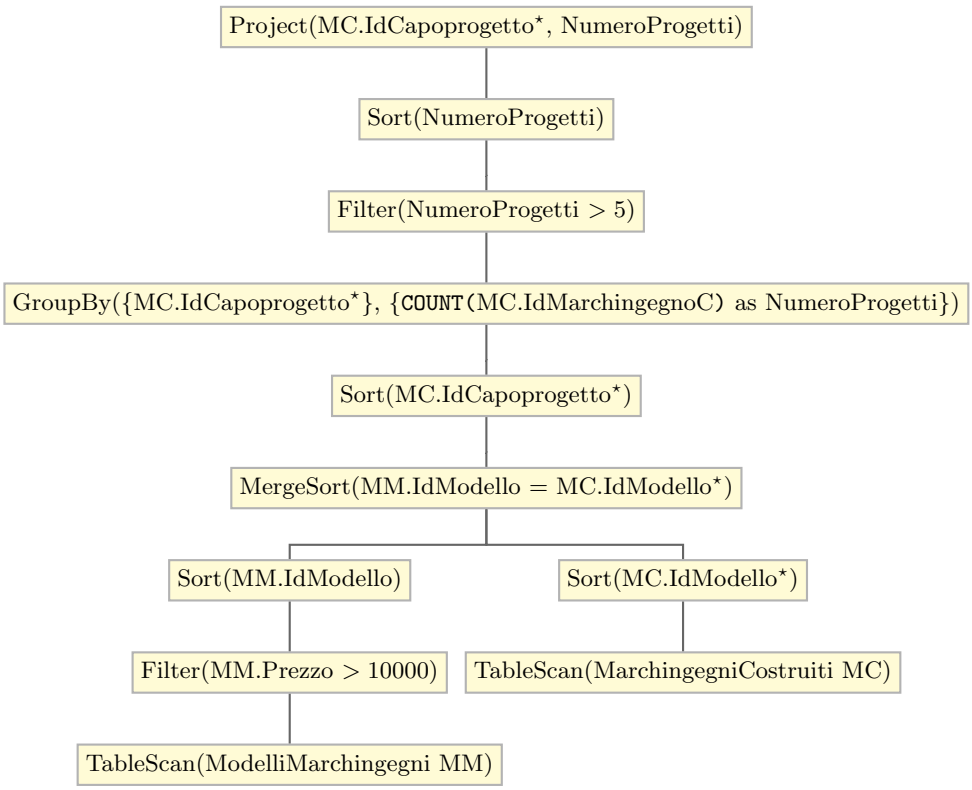


Abbiamo fatto commutare la restrizione Pr.DataConsegna < `'2020-05-01'` con la giunzione su IdCliente.

II) a)

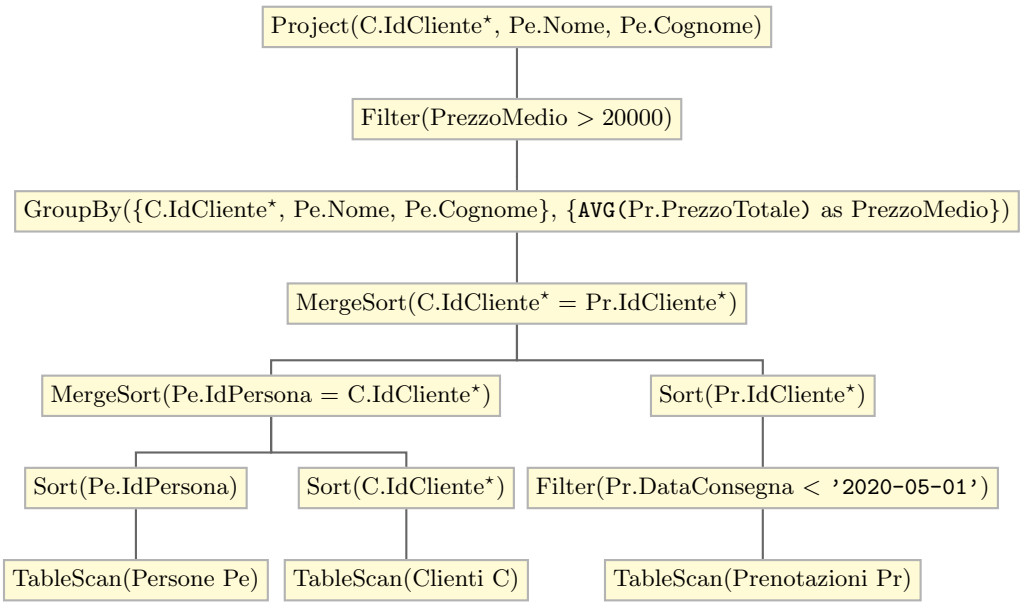


b)



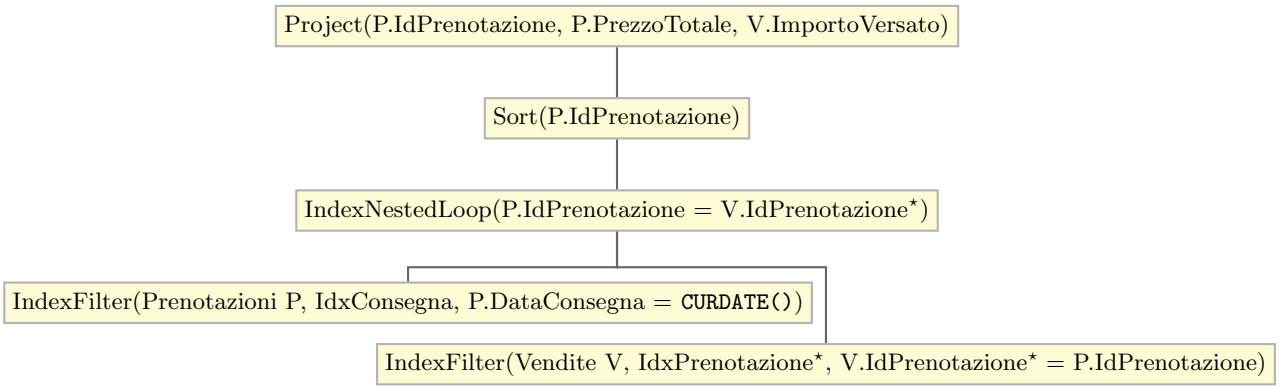
Il sort prima della group by non può essere rimosso in quanto i dati che arrivano dalla MergeSort sono ordinati per IdModello (o equivalentemente per IdModello*).

c)



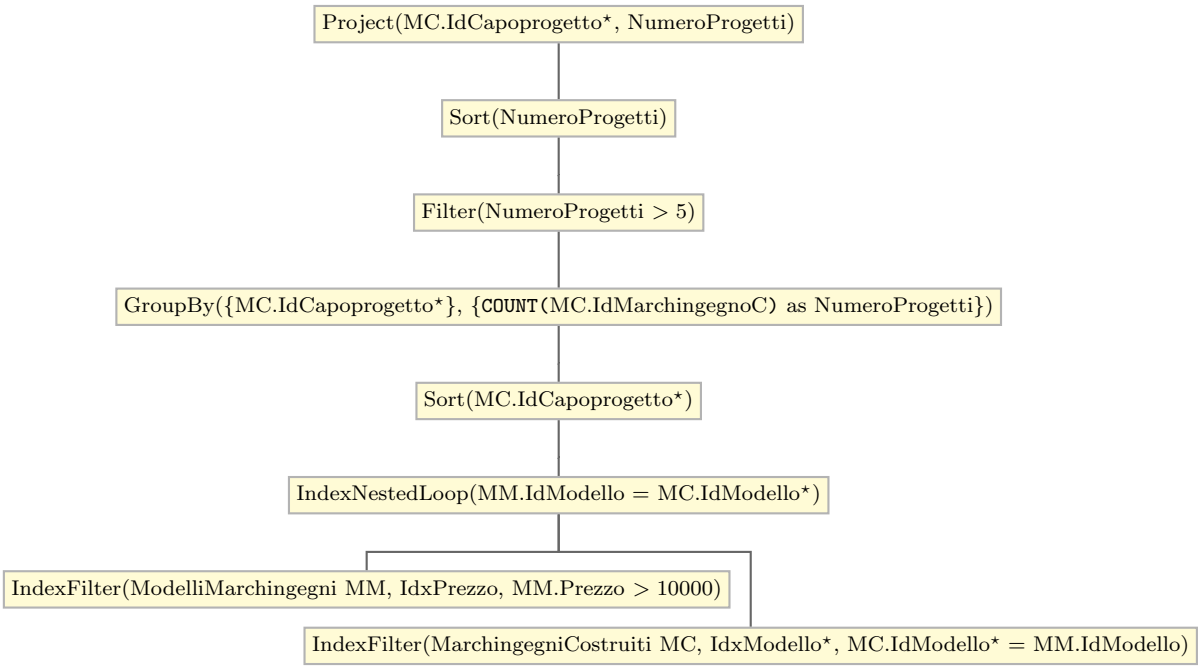
Il sort prima della group by non è necessario in quanto i dati arrivano ordinati su IdCliente*, dato che questa è una chiave avremmo automaticamente che i dati arrivano ordinati lessicograficamente rispetto a (C.IdCliente*, Pe.Nome, Pe.Cognome).

III) a) Piano di accesso fisico sulla query a) con indici su IdPrenotazione* (IdxPrenotazione*) e DataConsegna (IdxConsegna).



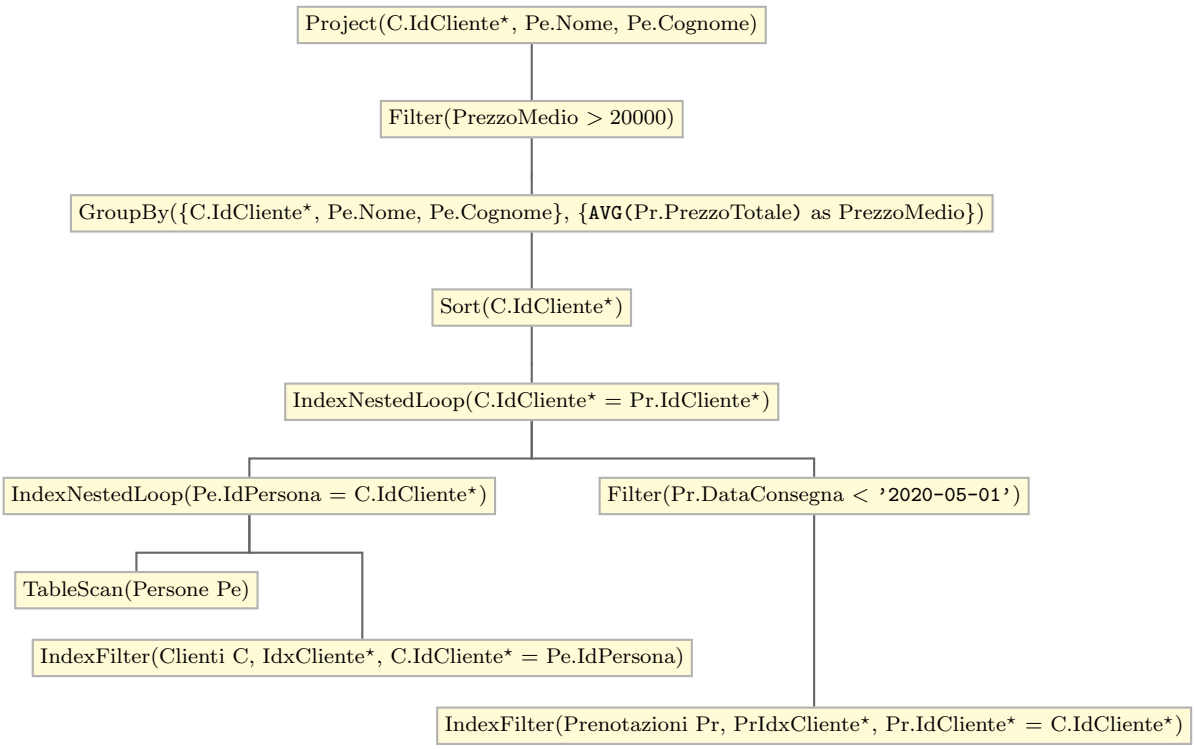
Non abbiamo fatto commutare Sort e IndexNestedLoop perchè per quanto la proprietà di ordinamento è conservata da IndexNestedLoop, si ordinerebbero più elementi prima della giunzione. L'IndexFilter su Prenotazioni è particolarmente efficiente dato che il filtro è molto selettivo. L'altro IndexFilter sarà molto selettivo di conseguenza.

b) Piano di accesso fisico sulla query b) con indici su Prezzo (IdxPrezzo) e IdModello* (IdxModello*).



Il sort prima della group by non può essere rimosso in quanto i dati che arrivano dall'IndexNestedLoop non sono sicuramente ordinati per IdCapoprogetto*.

c) Piano di accesso fisico sulla query c) con indici su IdCliente* per la tabella Clienti (IdxCliente*) e IdCliente* per la tabella Prenotazioni (PrIdxCliente*).



Il sort prima della group by non può essere rimosso in quanto i dati che arrivano dall'IndexNestedLoop non sono necessariamente ordinati per IdCliente*.