Basi di dati prof. Ghelli

"Una cervecita fresca"

Davide Testa 613565

2020-05-30

0 Descrizione di massima del dominio (testo)

Introduzione La birra fatta in casa è un'attività che riceve crescente attenzione da parte degli appassionati. Ogni birraio amatoriale possiede un'attrezzatura per il processo di produzione della birra su piccola scala (bollitori, fermentatori, tubi, ecc.) con una certa capacità massima di fermentazione: il numero di litri che l'attrezzatura è in grado di gestire in un unico "lotto". La preparazione della birra richiede anche ingredienti, le cui quantità effettive variano da una ricetta all'altra, questi sono vari tipi di malto, luppolo, lieviti e zuccheri (e, naturalmente, acqua).

Ai birrai piace registrare le proprie ricette per riferimento futuro e mantenere un elenco aggiornato degli ingredienti disponibili per fare acquisti prima della successiva produzione.

L'obiettivo di questo progetto è quello di sviluppare un'applicazione per i birrai domestici che consenta loro di mantenere un elenco di ricette e adattare quelle esistenti. L'applicazione deve anche:

- mantenere un elenco di ingredienti disponibili;
- aggiornare questo elenco dopo un ciclo di produzione e quando vengono acquistati nuovi ingredienti;
- produrre liste della spesa per il lotto successivo;
- guidare il birraio nel processo di produzione.

Descrizione del progetto "Una cervecita fresca" è un'applicazione che consente ai produttori amatoriali di birra di mantenere un database organizzato delle loro ricette di birra. L'applicazione consente agli utenti di creare, archiviare e modificare ricette, e successivamente eliminarle, se l'utente desidera farlo. L'applicazione è destinata solo ai produttori di birra con metodo all-grain, e quindi tutte le ricette sono per questo tipo di birre (le birre "estratto" non sono supportate).

Ogni birrificio domestico dispone di un'attrezzatura specifica, le cui caratteristiche portano a una particolare "dimensione del lotto": il numero massimo di litri che possono essere prodotti in una singola produzione. Le ricette prevedono, oltre all'acqua:

- malti
- luppolo

- lieviti
- zuccheri
- additivi

Mentre i produttori di birra preferiscono creare ricette riferendosi a valori concreti, come chilogrammi di un particolare malto o grammi di un particolare luppolo, l'applicazione deve memorizzare queste ricette in una misura "assoluta", che consente una conversione diretta della ricetta quando l'apparecchiatura, e di conseguenza la dimensione del lotto, è diversa. Ad esempio, una possibilità è esprimere la quantità di malto in percentuale del totale e usare i grammi per litro di miscuglio (mash) per il luppolo.

Oltre alle ricette, l'applicazione deve conservare le **istanze** della ricetta, ovvero singole produzioni basate su una ricetta; queste istanze possono essere accompagnate da note per fare riferimento a problemi che possono influire sulla birra risultante, note che i produttori di birra vorrebbero rimanessero memorizzate. Un particolare tipo di nota sono le note di degustazione, che consentono ai birrai di tenere traccia delle opinioni su una birra di un dato lotto.

Oltre a queste funzionalità più tradizionali, l'applicazione "Una cervecita fresca", mantiene un elenco di ingredienti disponibili. Ciò consente ai birrai di avere la lista degli ingredienti mancanti per la prossima produzione. Un'istanza della ricetta, ovvero una produzione di birra, dovrebbe consentire agli utenti di aggiornare l'elenco degli ingredienti disponibili, sottraendo gli ingredienti usati da quelli disponibili.

Sarà inoltre possibile per i birrai vendere la birra prodotta. L'applicazione deve offrire un'interfaccia web per la prenotazione e la vendita. Un cliente registrato può prenotare un lotto di birra in produzione, oppure parte di esso. Quando il lotto è stato prodotto, il birraio può confermare le prenotazioni e procedere con la vendita oppure, se non è soddisfatto del prodotto, cancellarle, per non danneggiare il proprio buon nome. La birra non prenotata può essere messa in vendita e comprata da utenti registrati.

Scopo dell'applicazione Il sistema deve implementare le funzionalità sopra descritte, ovvero creazione, modifica e cancellazione di ricette, creazione di istanze di ricette (birre), supporto per le note sulle birre, controllo degli ingredienti disponibili, supporto alla produzione con allarmi, supporto alla vendita.

Scopo del progetto per quanto riguarda Basi di Dati Si integrano i requisiti già specificati con le seguenti ulteriori informazioni:

- le ricette sono relative ad un solo birrificio ma possono essere condivise tra diversi birrai che sono autorizzati al loro utilizzo;
- gli ingredienti possono essere acquistati da più fornitori (registrati).

1 Descrizione del dominio

L'applicazione "Una cervecita fresca" deve fornire supporto ai birrai e alle birraie artigianali nella produzione delle loro birre fatte in casa con il metodo all-grain.

Ogni utente dell'app (birraio o birraia) può lavorare per uno o più birrifici, e può visualizzare le ricette dei birrifici per cui lavora. Del* birrai* sono rilevanti il nome, il cognome, il soprannome, l'indirizzo email, il codice fiscale. Ogni birrificio ha un nome, un anno di fondazione, un motto e uno stemma. Il birrificio ha inoltre una capacità produttiva, cioè il numero massimo di litri che può produrre in un singolo ciclo produttivo. Per ogni birrificio possono lavorare più persone. Ogni birrificio può lavorare a una sola produzione per volta, mettendo a disposizione tutta la sua capacità produttiva, o solo una parte di questa.

Ogni birrificio ha accesso a una o più ricette: ogni ricetta ha un nome, un creatore o creatrice, una data di creazione, una eventuale ricetta madre (ovvero la ricetta che è stata modificata per elaborarla) e la quantità relativa di ciascun ingrediente. Oltre alla creatrice o creatore di una ricetta, anche le birraie e i birrai di un birrificio possono vedere le ricette del birrificio in cui lavorano. La ricetta può essere archiviata o eliminata dal creatore o creatrice: lo stato della ricetta può essere attiva, archiviata, eliminata.

Gli ingredienti sono moltissimi, ma di solo 5 tipi: malti, luppoli, lieviti, zuccheri e additivi. Ciascun tipo ha un nome e la sua unità di misura appropriata (per esempio, i luppoli vengono espressi in *mash*, ovvero grammi per litro di miscuglio, mentre i malti sono espressi in peso percentuale sugli ingredienti secchi). Ciascun ingrediente ha un tipo e una descrizione. In ogni ricetta, è indicata la quantità di ciascun ingrediente come numero puro: l'unità di misura (mash, peso percentuale, ...) dipende dal tipo di ingrediente. La quantità di acqua è ricavabile dagli altri ingredienti e non occorre quindi memorizzarla: si miscelano in proporzione gli ingredienti secchi, dal *mash* si capisce quale volume finale deve raggiungere la soluzione.

La visualizzazione delle ricette terrà conto della quantità di prodotto che si vuole produrre (che dev'essere inferiore o uguale alla capacità produttiva del birrificio) per mostrare le quantità assolute dei vari ingredienti in kg e L; nel database invece le quantità verranno memorizzate in termini relativi come descritto sopra.

Una produzione è caratterizzata da una data di produzione, un numero di lotto che la identifica univocamente all'interno del birrificio, il numero di bottiglie da 500 mL prodotte (questo è l'unico possibile formato di produzione) e uno stato di preparazione (in corso, completa, annullata). È prodotta seguendo una ricetta di un birrificio.

A ogni produzione si possono accompagnare alcune note. Ogni nota ha un testo. Esistono particolari note, dette di degustazione, che esprimono anche un giudizio da 1 a 10 sulla qualità del prodotto.

Il birrificio tiene un registro degli acquisti, conservando i dati della fattura e specificando per ogni ingrediente acquistato la quantità. Ogni fattura registrata dal birrificio è caratterizzata da una data, un numero di fattura, un importo e un fornitore. I fornitori hanno una ragione sociale, una partita IVA e un indirizzo. L'inventario mostra, per ogni ingrediente, la quantità disponibile e quella totale (compresi cioè gli ingredienti "prenotati" da preparazioni in corso).

Oltre che come birrario o birraia, ci si può anche registrare come cliente, specificando un indirizzo di spedizione. I clienti sono caratterizzati, come chi produce birra, da nome, cognome, email e codice fiscale, ma non hanno un soprannome.

La clientela può effettuare prenotazioni per una quantità di bottiglie di un dato lotto di produzione. Ogni prenotazione ha uno stato, che rimane in sospeso fino al termine della produzione, quando il birrificio può impostarlo su 'confermato' se intende procedere alla vendita oppure 'annullato' se il prodotto non è soddisfacente.

2 Schema concettuale

La figura 1 mostra lo schema concettuale in formato grafico.

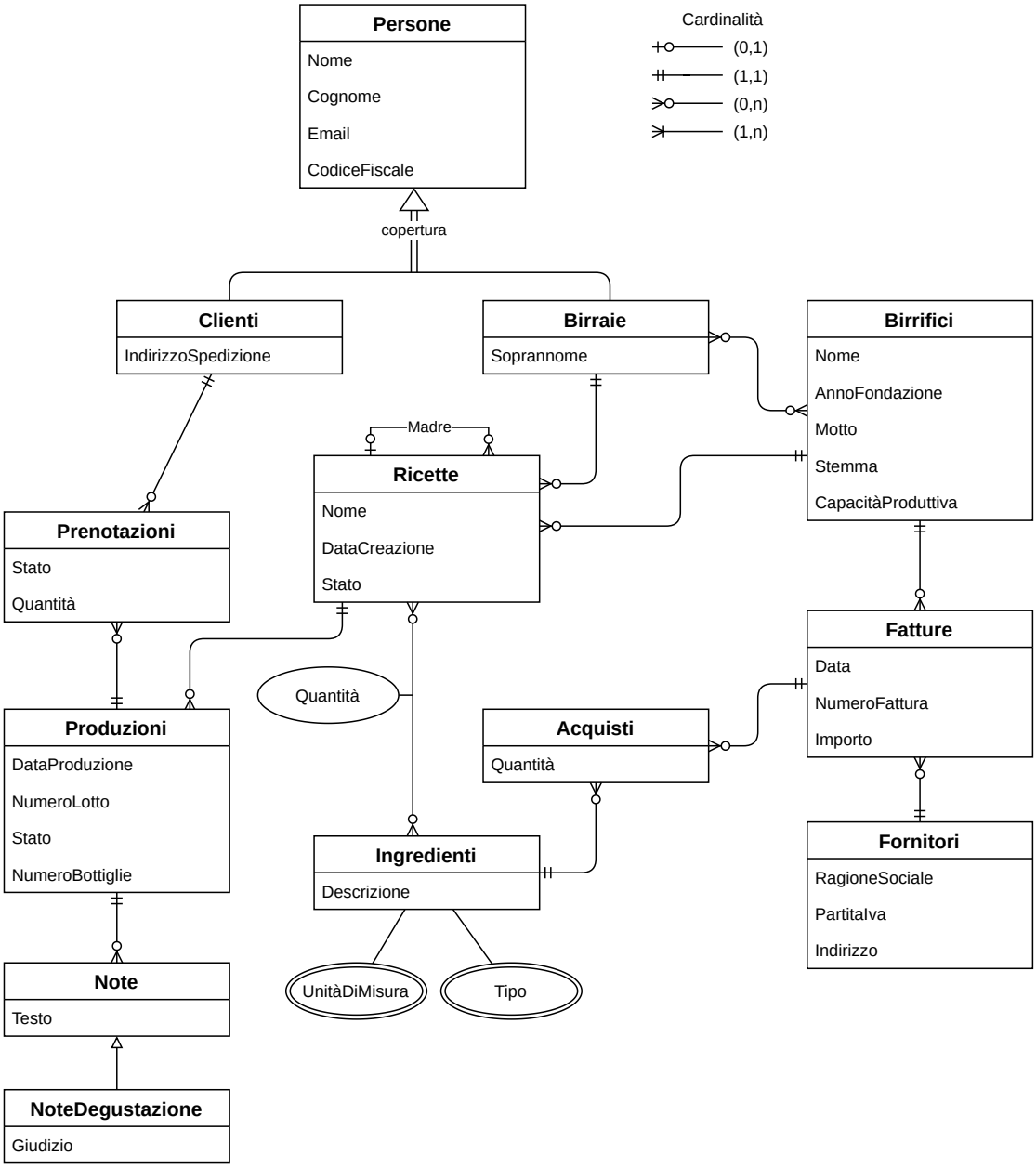


Figura 1: Schema concettuale in formato grafico

Elenco degli ingredienti disponibili *l* committente ha chiesto espressamente l'elenco degli ingredienti disponibili. Tuttavia, questo è calcolabile a partire dagli acquisti e dalle produzioni. Ho convinto *l* committente a non memorizzare separatamente l'inventario degli ingredienti, rassicurandol* che avrei fornito una vista logica "Inventario".

```
CREATE VIEW IngredientiAcquistatiTotali (IdIngrediente, Ingrediente,
                                         Totale)
AS SELECT i.IdIngrediente IdIngrediente, i.Descrizione Ingrediente,
          SUM(a.Quantità) Totale
   FROM Acquisti a
   JOIN Ingredienti i ON i.IdIngrediente = a.IdIngrediente
   GROUP BY i.IdIngrediente, i.Descrizione;

CREATE VIEW IngredientiInUso (IdIngrediente, Ingrediente, InUso)
AS SELECT i.IdIngrediente IdIngrediente, i.Descrizione Ingrediente,
          SUM(ir.Quantità) InUso
   FROM IngredientiRicette ir
   JOIN Ingredienti i ON i.IdIngrediente = ir.IdIngrediente
   JOIN Produzioni p ON p.IdRicetta = ir.IdRicetta
   WHERE p.Stato IS NULL
   GROUP BY i.IdIngrediente, i.Descrizione;

CREATE VIEW IngredientiUsati (IdIngrediente, Ingrediente,
                              Usati)
AS SELECT i.IdIngrediente IdIngrediente, i.Descrizione Ingrediente,
          SUM(ir.Quantità) Usati
   FROM IngredientiRicette ir
   JOIN Ingredienti i ON i.IdIngrediente = ir.IdIngrediente
   JOIN Produzioni p ON p.IdRicetta = ir.IdRicetta
   WHERE p.Stato = 0
   GROUP BY i.IdIngrediente, i.Descrizione;

CREATE VIEW Inventario (IdIngrediente, Ingrediente, QuantitàTotale,
                        QuantitàDisponibile)
AS SELECT iat.IdIngrediente IdIngrediente,
          iat.Ingrediente Ingrediente,
          (iat.Totale - COALESCE(iu.Usati, 0)) QuantitàTotale,
          (iat.Totale - COALESCE(iu.Usati, 0)
            - COALESCE(iiu.InUso, 0)) QuantitàDisponibile
   FROM IngredientiAcquistatiTotali iat
   LEFT JOIN IngredientiUsati iu
             ON iu.IdIngrediente = iat.IdIngrediente
   LEFT JOIN IngredientiInUso iiu
             ON iiu.IdIngrediente = iat.IdIngrediente
   WHERE iat.Totale - COALESCE(iu.Usati, 0) > 0;
```

Ogni volta che si inizia una produzione, l'applicazione controlla che la quantità di ingredienti disponibili superi la quantità degli ingredienti necessari alla preparazione.

L'applicazione può anche mostrare una "lista della spesa" basandosi su ricette che si vogliono preparare e sulla vista inventario.

L'applicazione mostrerà ad ogni birrai* solo le ricette di cui è aut*r* o di un birrificio per cui lavora.

Vincoli intra-relazionali

- Non possono esistere due persone con lo stesso codice fiscale.
- Non possono esistere due birrai* con lo stesso soprannome.
- Non possono esistere due fornitori con la stessa partita IVA né con la stessa ragione sociale.
- Il tipo di ingrediente determina l'unità di misura. Esiste un breve elenco di tipi ingredienti disponibili con la relativa unità di misura. Non ho creato una classe "TipiIngredienti" per contenere il numero di classi, ma in effetti il tipo determina funzionalmente l'unità di misura ed esistono pochi tipi, mentre ci sono molti ingredienti per ogni tipo.

Vincoli inter-relazionali

- Alla registrazione, l'utente deve inserire un soprannome e/o un indirizzo di spedizione: il vincolo di copertura impone che l'unione di Clienti e Birraie sia Persone, non devono esistere persone che non sono né clienti né birrai*.
- In ogni produzione, il NumeroBottiglie diviso per 0.5 non deve superare la CapacitàProduttiva del birrificio.
- Ogni produzione deve iniziare con stato 'in corso'; non può iniziare una produzione se un altra è 'in corso' nello stesso birrificio.
- Ogni nota deve fare riferimento a una produzione.
- Ogni prenotazione deve fare riferimento a una produzione.
- Ogni prenotazione deve fare riferimento ad un* cliente.
- La quantità di bottiglie di un produzione prenotate non deve superare il numero di bottiglie prodotte.
- Ogni produzione deve seguire una ricetta.
- Due produzioni di uno stesso birrificio non devono avere lo stesso lotto.
- Ogni ricetta deve avere un* creat*r* e un birrificio di riferimento.
- Ogni fattura deve fare riferimento ad un birrificio e un fornitore.
- Ogni acquisto deve riferirsi a una fattura e un ingrediente.
- Ogni ricetta deve avere almeno un ingrediente per ciascuno dei seguenti tipi: malto, luppolo, lievito.

3 Schema logico relazionale

La figura 2 mostra lo schema logico relazionale in formato grafico.

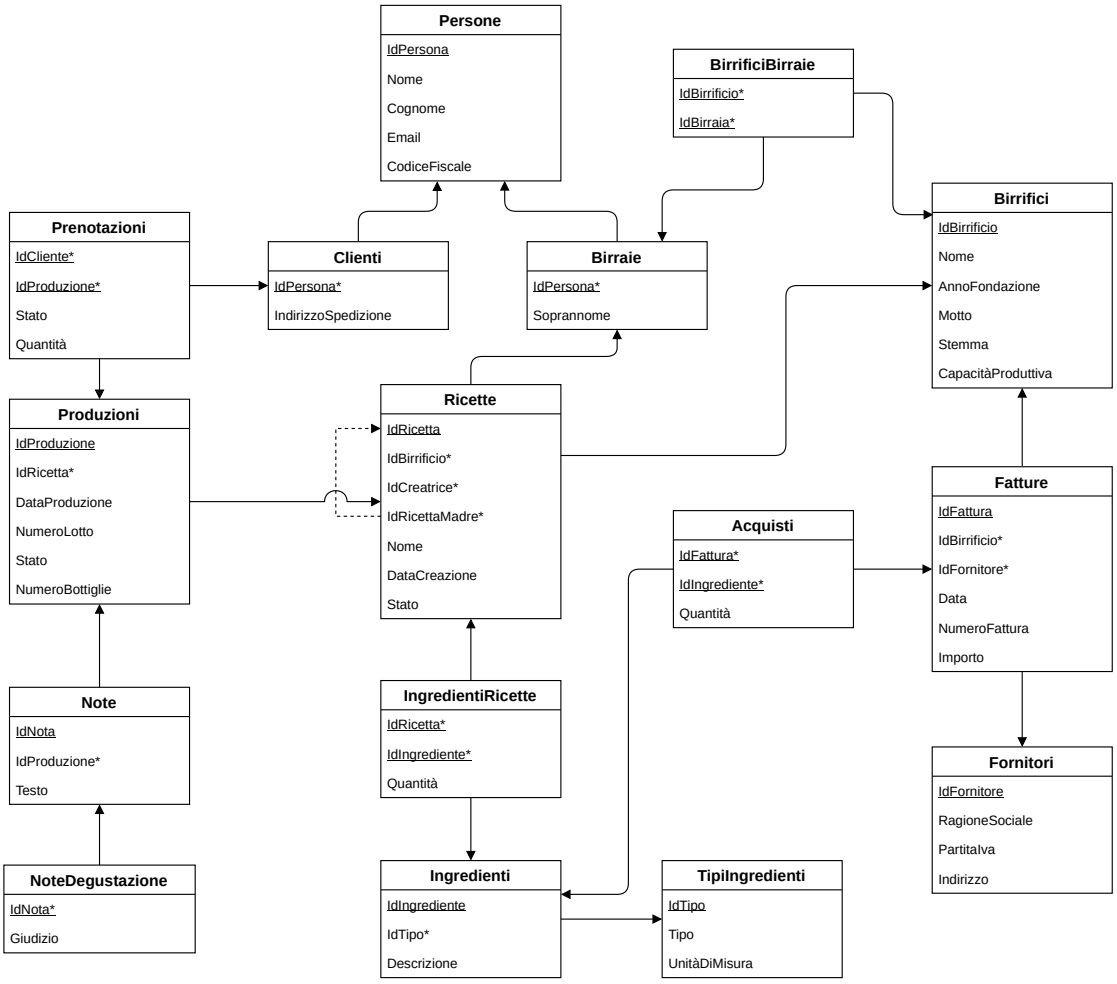


Figura 2: Schema logico relazionale in formato grafico

Schema logico relazionale in formato testuale

```
Persone(IdPersona, Nome, Cognome, Email, CodiceFiscale)
Clienti(IdPersona*, IndirizzoSpedizione)
Birraie(IdPersona*, Soprannome)
Birrifici(IdBirrificio, Nome, AnnoFondazione, Motto, Stemma,
          CapacitàProduttiva)
BirrificiBirraie(IdBirrificio*, IdBirraia*)
Fornitori(IdFornitore, RagioneSociale, PartitaIva, Indirizzo)
Fatture(IdFattura, IdBirrificio*, IdFornitore*, Data,
        NumeroFattura, Importo)
TipiIngredienti(IdTipo, Tipo, UnitàDiMisura)
Ingredienti(IdIngrediente, IdTipo*, Descrizione)
Acquisti(IdFattura*, IdIngrediente*, Quantità)
Ricette(IdRicetta, IdBirrificio*, IdCreatrice*, IdRicettaMadre*,
```

```
        Nome, DataCreazione, Stato)
IngredientiRicette(IdRicetta*, IdIngrediente*, Quantità)
Produzioni(IdProduzione, IdRicetta*, DataProduzione, NumeroLotto,
        Stato, NumeroBottiglie)
Prenotazioni(IdCliente*, IdProduzione*, Stato, Quantità)
Note(IdNota, IdProduzione*, Testo)
NoteDegustazione(IdNota*, Giudizio)
```

Dipendenze funzionali

- Per ogni tabella la chiave primaria (sottolineata) determina ciascuno degli attributi della tabella.

  `IdPersona` → `Nome`, `IdPersona` → `Cognome`, `IdPersona` → `Email`, `IdPersona` → `CodiceFiscale`, …

- Nella tabella `Persone`, `CodiceFiscale` è chiave naturale e determina tutti gli altri attributi. Ho ritenuto prudente aggiungere una chiave artificiale perché, se è vero che due persone diverse non avranno mai lo stesso codice fiscale, è vero anche che ci possono essere errori umani nell'inserimento di un CF e voglio riservarmi la possibilità di correggere un CF senza minare l'affidabiltà della base di dati.
- Stesso discorso per la `RagioneSociale` e la `PartitaIva` nella tabella `Fornitori`: ciascuno è chiave separatamente.
- Nella tabella `Fatture`, la coppia di attributi {`IdFornitore`, `NumeroFattura`} è chiave.
- Nella tabella `Produzioni`, il NumeroLotto non è chiave, in quanto birrifici diversi possono avere lotti uguali, è solo all'interno del birrificio che il lotto identifica univocamente la produzione.

Uno schema R, avente insieme di attributi T e insieme di dipendenze funzionali F, R<T, F>, è in forma normale di Boyce-Codd (BCNF) se ogni dipendenza funzionale della chiusura di F o è banale o ha come determinante una superchiave di T. Esiste un teorema che semplifica il calcolo, asserendo che se la condizione di cui sopra vale per una qualsiasi copertura di F allora vale per l'intera chiusura di F.

Nella copertura di F che ho descritto sopra (che peraltro è canonica: ogni dipendenza ha un solo attributo come determinato, nessuna dipendenza è ridondante e non sono presenti attributi estranei, in quanto ogni determinante è chiave), ogni dipendenza funzionale ha come determinante o la chiave primaria o una chiave naturale che non è stata scelta come primaria, in ogni caso una superchiave. La BCNF è pertanto rispettata.

Alcuni vincoli intra-relazionali possono essere tradotti in trigger in modo da mantenere la coerenza del database con l'inserimento di record. Per esempio:

```
CREATE TRIGGER IntegritàReferenzialeNote
   BEFORE INSERT ON Note
BEGIN
    SELECT
        CASE WHEN NEW.IdProduzione NOT IN (
            SELECT IdProduzione FROM Produzioni)
        THEN RAISE (ABORT,'Produzione non valida!')
        END;
END;
```

4 Interrogazioni

a. Uso di proiezione, join e restrizione.

   Mostrare l'IdRicetta e il Nome delle ricette create da birrai di nome Giovanni.

```
SELECT r.IdRicetta, r.Nome
FROM Ricette r
JOIN Persone p ON p.IdPersona = r.IdCreatrice
WHERE p.Nome = 'Giovanni'
```

b. Uso di group by con having, where e sort.

   Per ogni birrificio che abbia fatto almeno un acquisto quest'anno, riportare l'IdBirrificio e il numero di diversi fornitori da cui ha acquistato quest'anno, se questo numero è almeno di 3. Ordinare il risultato dal birrificio che ha avuto più fornitori a quello che ne ha avuti meno.

```
SELECT fa.IdBirrificio,
       COUNT(DISTINCT fa.IdFornitore) DiversiFornitori
FROM Fatture fa
WHERE fa.Data >= '2020-01-01'
GROUP BY fa.IdBirrificio
HAVING COUNT(DISTINCT fa.IdFornitore) >= 3
ORDER BY COUNT(DISTINCT fa.IdFornitore) DESC
```

c. Uso di join, group by con having e where.

   Dei fornitori da cui ha ordinato il birrificio 'Pirati Rossi', mostrare la ragione sociale, l'importo totale e l'importo medio delle fatture, purché l'importo totale sia superiore a 10 euro.

```
SELECT fo.RagioneSociale, SUM(fa.Importo) ImportoTotale,
       AVG(fa.Importo) ImportoMedio
FROM Fornitori fo
JOIN Fatture fa ON fa.IdFornitore = fo.IdFornitore
JOIN Birrifici b ON b.IdBirrificio = fa.IdBirrificio
WHERE b.Nome = 'Pirati Rossi'
GROUP BY fo.IdFornitore, fo.RagioneSociale
HAVING SUM(fa.Importo) > 10
```

d. Uso di select annidata con quantificazione esistenziale.

   Mostrare il soprannome de* birrai* che siano aut*r* di almeno una ricetta.

```
SELECT b.Soprannome
FROM Birraie b
WHERE EXISTS (SELECT *
              FROM Ricette r
              WHERE r.IdCreatrice = b.IdPersona)
```

e. Uso di select annidata con quantificazione universale.

Mostrare il nome e il cognome de* clienti che hanno ordinato da un solo birrificio.

Traduco in notazione insiemistica:

```
{p1.Nome, p1.Cognome | (p1 ∈ Persone, pre1 ∈ Prenotazioni,
                        pre1.IdCliente = p1.IdPersona,
                        pro1 ∈ Produzioni,
                        pro1.IdProduzione = pre1.IdProduzione,
                        r1 ∈ Ricette,
                        r1.IdRicetta = pro1.IdRicetta) .
    ∀ (pre2 ∈ Prenotazioni, pre2.IdCliente = pre1.IdCliente
       pro2 ∈ Produzioni, pro2.IdProduzione = pre2.IdProduzione,
       r2 ∈ Ricette, r2.IdRicetta = pro2.IdRicetta) .
       (r2.IdBirrificio = r1.IdBirrificio)}
```

Sostituisco il $\forall x.P$ con $\neg\exists x.\neg P$

```
{p1.Nome, p1.Cognome | (p1 ∈ Persone, pre1 ∈ Prenotazioni,
                        pre1.IdCliente = p1.IdPersona,
                        pro1 ∈ Produzioni,
                        pro1.IdProduzione = pre1.IdProduzione,
                        r1 ∈ Ricette,
                        r1.IdRicetta = pro1.IdRicetta) .
    ¬∃ (pre2 ∈ Prenotazioni, pre2.IdCliente = pre1.IdCliente
        pro2 ∈ Produzioni, pro2.IdProduzione = pre2.IdProduzione,
        r2 ∈ Ricette, r2.IdRicetta = pro2.IdRicetta) .
        (r2.IdBirrificio ≠ r1.IdBirrificio)}
```

Scrivo quindi la query, inserendo l'`IdPersona` e la parola chiave `DISTINCT` per rimuovere i duplicati (ma non le persone omonime).

```
SELECT DISTINCT p1.IdPersona, p1.Nome, p1.Cognome
FROM Persone p1
JOIN Prenotazioni pre1 ON pre1.IdCliente = p1.IdPersona
JOIN Produzioni pro1 ON pro1.IdProduzione = pre1.IdProduzione
JOIN Ricette r1 ON r1.IdRicetta = pro1.IdRicetta
WHERE NOT EXISTS (SELECT *
                  FROM Prenotazioni pre2
                  JOIN Produzioni pro2
                       ON pro2.IdProduzione = pre2.IdProduzione
                  JOIN Ricette r2 ON r2.IdRicetta = pro2.IdRicetta
                  WHERE pre2.IdCliente = pre1.IdCliente
                        AND r2.IdBirrificio <> r1.IdBirrificio)
```

f. Uso di subquery di confronto quantificato.

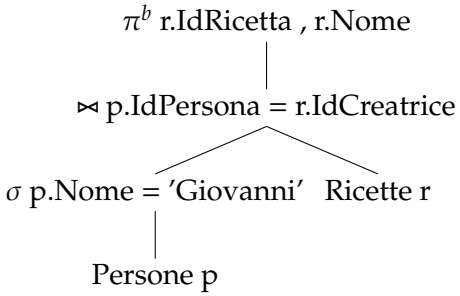
Per ogni birrificio, mostrare l'IdBirrificio e l'ultimo NumeroLotto prodotto in quel birrificio (sapendo che il NumeroLotto è progressivo).

```
SELECT r1.IdBirrificio, pro1.NumeroLotto
FROM Produzioni pro1
JOIN Ricette r1 ON r1.IdRicetta = pro1.IdRicetta
WHERE pro1.NumeroLotto >= ANY (SELECT pro2.NumeroLotto
                               FROM Produzioni pro2
                               JOIN Ricette r2
                                    ON r2.IdRicetta = pro2.IdRicetta
                               WHERE r2.IdBirrificio = r1.IdBirrificio)
```

5 Piani di accesso

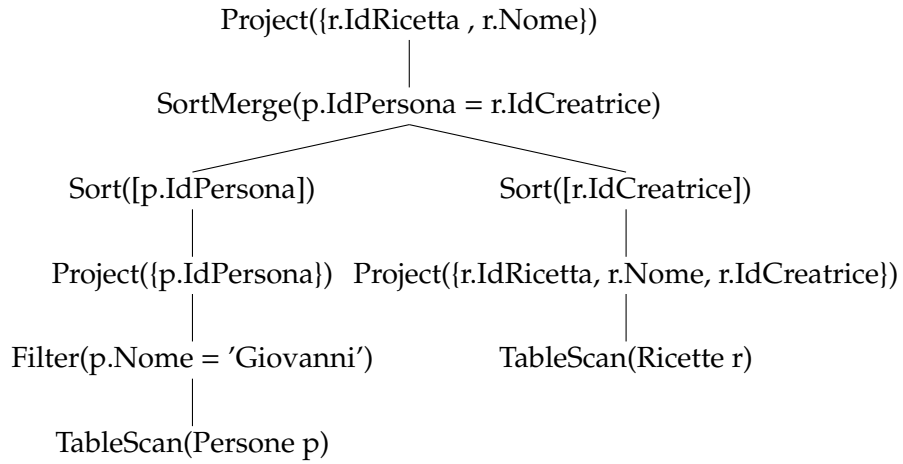
5.1 Query a

Piano di accesso logico della query a

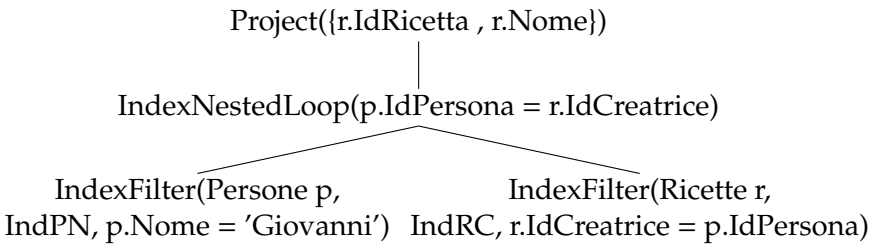


Non c'è in questo caso differenza tra π^b e π: non ci possono essere duplicati.

Piano di accesso fisico della query a senza indici



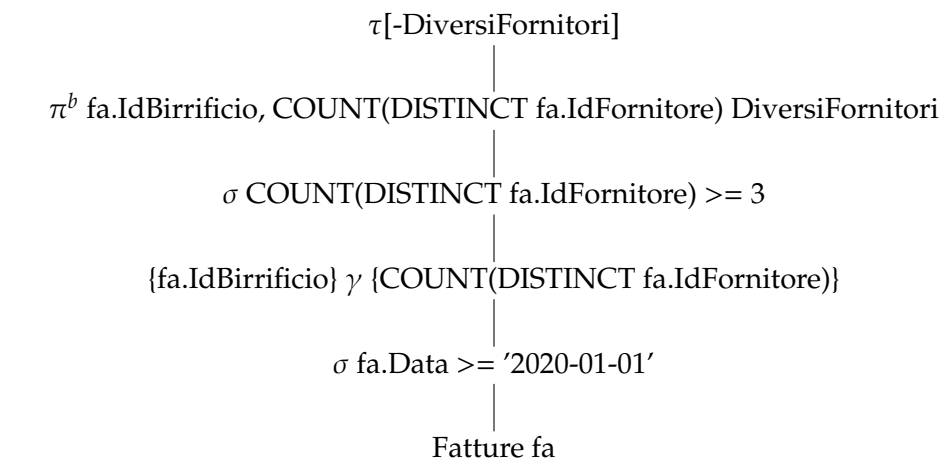
Piano di accesso fisico della query a con due indici



Indici necessari: `IndPN` (indice della tabella Persone sull'attributo Nome) e `IndRC` (indice della tabella Ricette sull'attributo Creatrice).

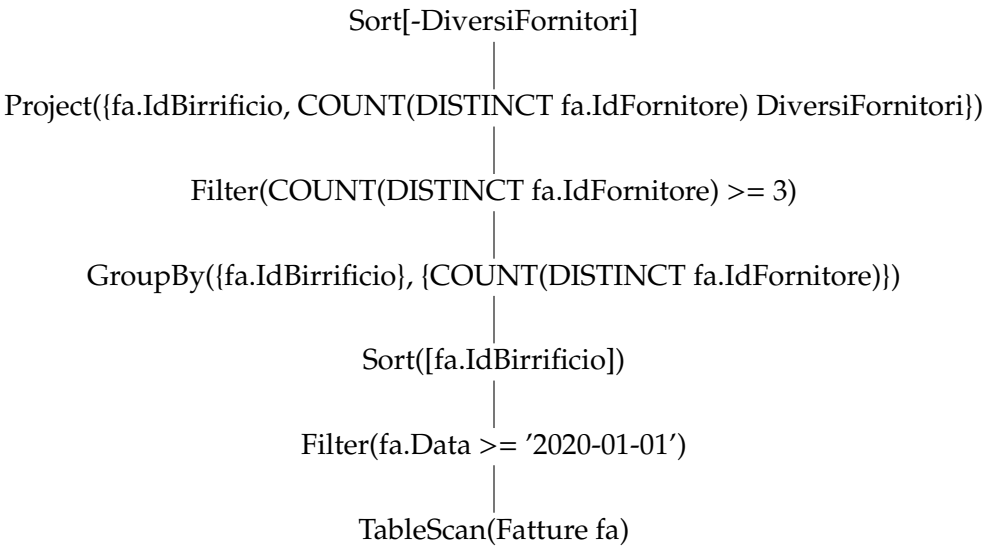
5.2 Query b

Piano di accesso logico della query b



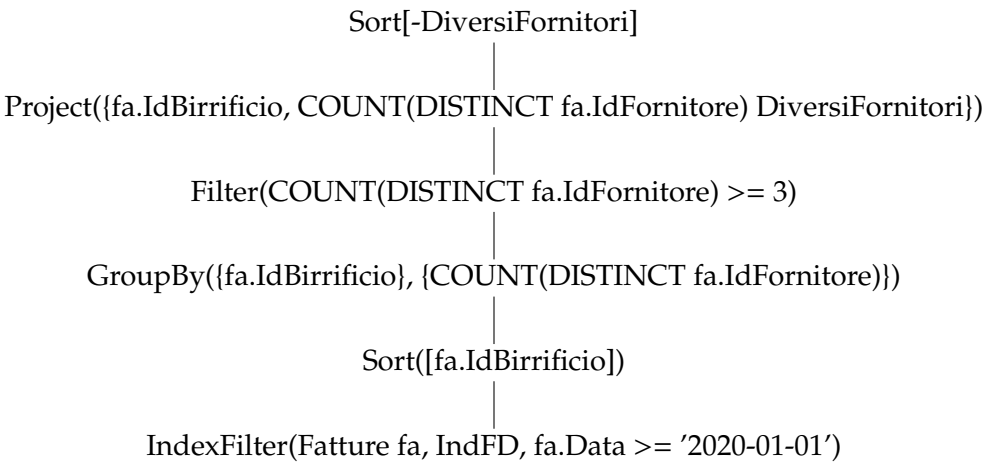
Non c'è in questo caso differenza tra π^b e π: non ci possono essere duplicati, in quanto la GROUP BY raggruppa per IdBirrificio.

Piano di accesso fisico della query b senza indici



Il sort sull'attributo dimensione di analisi prima della GroupBy è necessario, in quanto non è garantito che i record della tabella Fatture siano raggruppati per IdBirrificio. Lo sarebbero se l'organizzazione primaria della tabella fosse sequenziale proprio su questo attributo, il che è estremamente poco probabile.

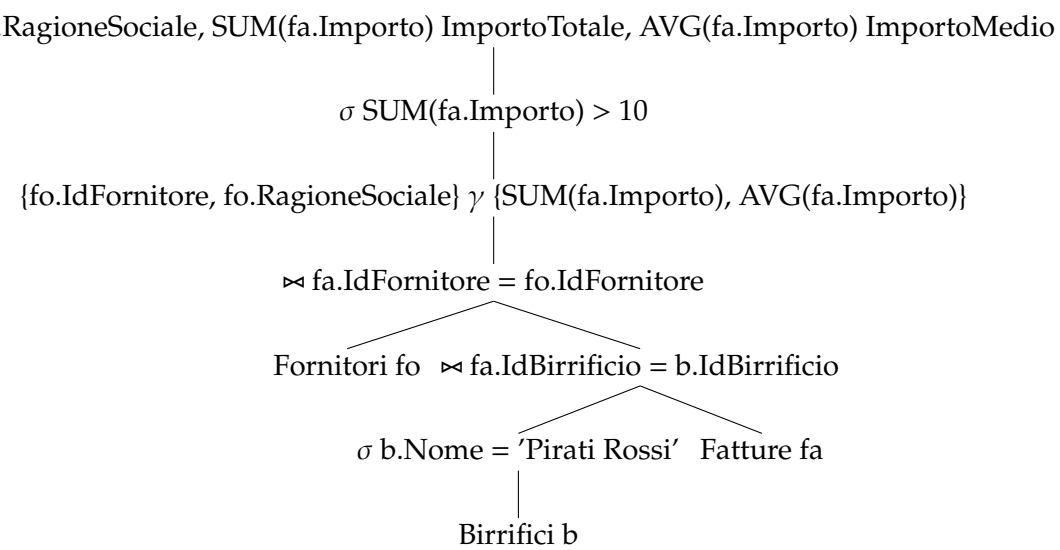
Piano di accesso fisico della query b con un indice



Indice necessario: `IndFD` (indice della tabella Fatture sull'attributo Data). Il sort sull'attributo IdBirrificio prima della GroupBy è necessario, in quanto i record in input sono ordinati per data, il che non ci garantisce che siano raggruppati per IdBirrificio (che è dimensione di analisi).

5.3 Query c

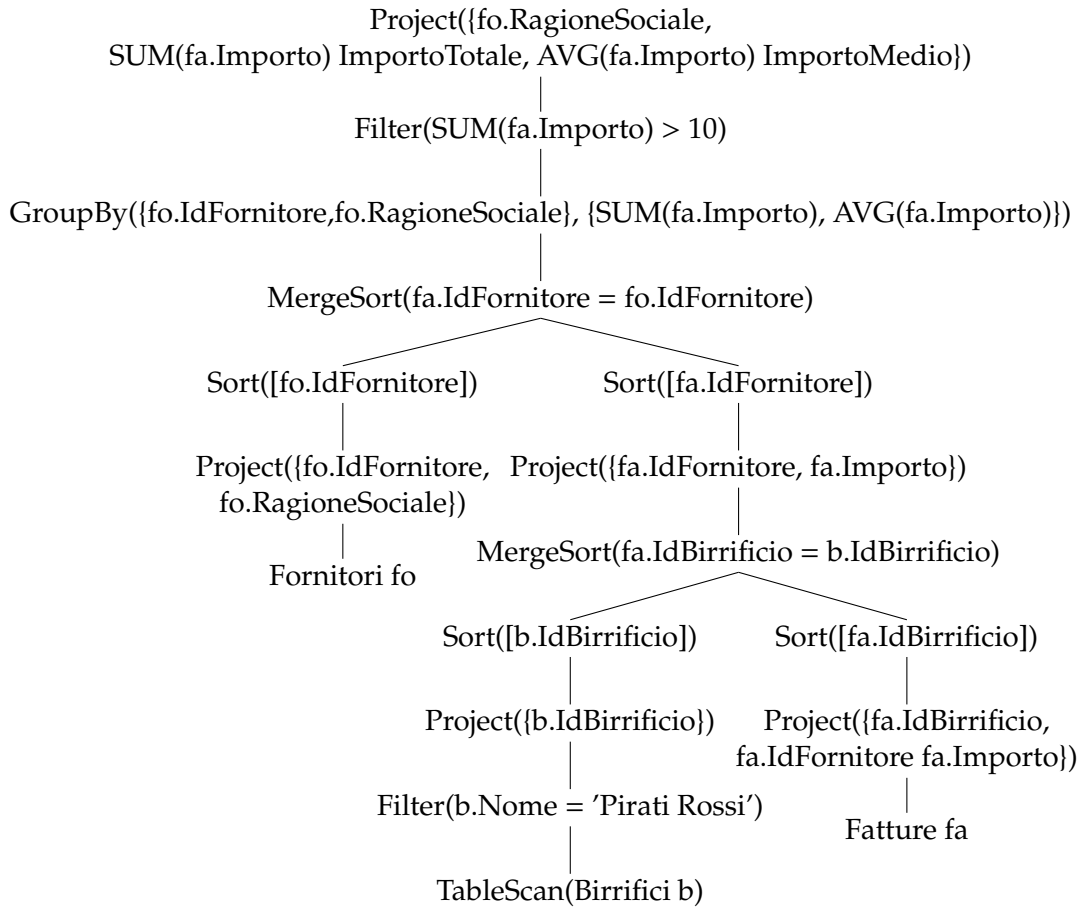
Piano di accesso logico della query c



In questo caso non ci dovrebbe essere differenza tra π^b e π: non ci devono essere due fornitori con la stessa ragione sociale (la ragione sociale è chiave naturale); è comunque possibile un errore di inserimento se non ho impostato un vincolo di unicità anche su questo attributo, che non ho scelto come chiave primaria della tabella: ecco perché ho raggruppato anche per IdFornitore e non solo per RagioneSociale.

Ho scelto l'ordine di giunzione in modo da avere la restrizione il più distale possibile.

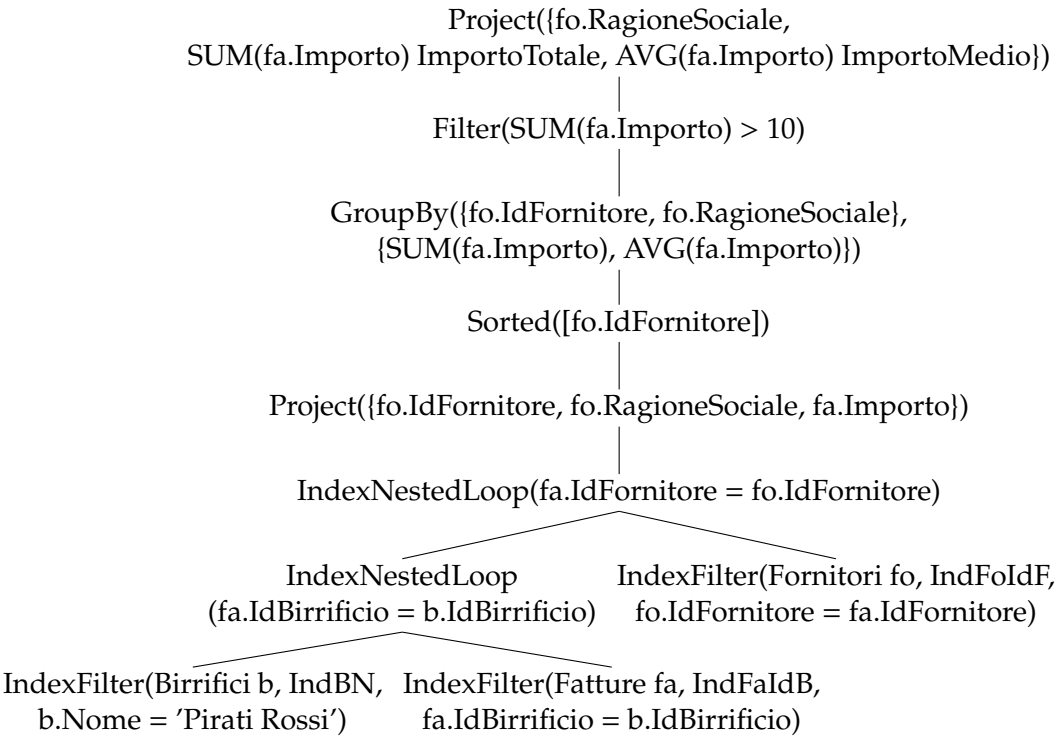
Piano di accesso fisico della query c senza indici



Non è necessario ordinare per `[fo.IdFornitore, fo.RagioneSociale]` prima della GroupBy: per costruzione, l'ordine dell'operatore esterno della SortMerge viene mantenuto nell'output, e questo ordine è sull'attributo fo.IdFornitore, che a sua volta determina funzionalmente l'altra dimensione di analisi, fo.RagioneSociale.

Pertanto, è garantito che l'input della GroupBy sarà già raggruppato per gli attributi che sono dimensione di analisi e non occorre un ordinamento preventivo.

Piano di accesso fisico della query c con tre indici



Indici necessari: `IndBN` (indice della tabella Birrifici sull'attributo Nome), `IndFaIdB` (indice della tabella Fatture sull'attributo IdBirrificio) e `IndFoIdF` (indice della tabella Fornitori sull'attributo IdFornitore).

Occorre ordinare per IdFornitore prima della `GroupBy`, in quanto l'output della IndexNestedLoop è ordinato come l'operatore esterno, ovvero per nome del birrificio.

Potrei spostare l'ordinamento tra le due giunzioni con IndexNestedLoop, tanto ogni fattura ha un fornitore e l'output non andrà a decrecere dopo la seconda giunzione (anzi, si arricchirà di campi). Il sort andrebbe fatto con il minor numero possibile di dati, dato l'alto costo dell'algoritmo, eliminando i campi superflui con una project prima.

Il vantaggio dell'IndexNestedLoop sul SortMerge si ha solo se la condizione è sufficientemente restrittiva da essere soddisfatta da una piccola minoranza di record. In questo caso, la restrizione sul nome del birrificio dovrebbe essere abbastanza restrittiva (se ci sono abbastanza birrifici, il numero di birrifici con il nome 'Pirati Rossi' sarà trascurabile rispetto al totale) ed è ragionevole che le fatture che riguardano quel birrificio siano una esigua minoranza rispetto al totale delle fatture. Se così non fosse, pur avendo i tre indici a disposizione, converrebbe utilizzare comunque il SortMerge.

Codice sorgente e test: https://gogs.davte.it/Davte/basi_di_dati