

## HOMEWORK 3 – Filippo Iacobelli 582898

### → PROCESSO

Si tratta di analizzare dati relativi a ricette/prescrizioni mediche.

### → GRANA DEI FATTI

La grana opportuna è rappresentata dal singolo elemento di ricetta, si tratta infatti di una transaction fact table atomica.

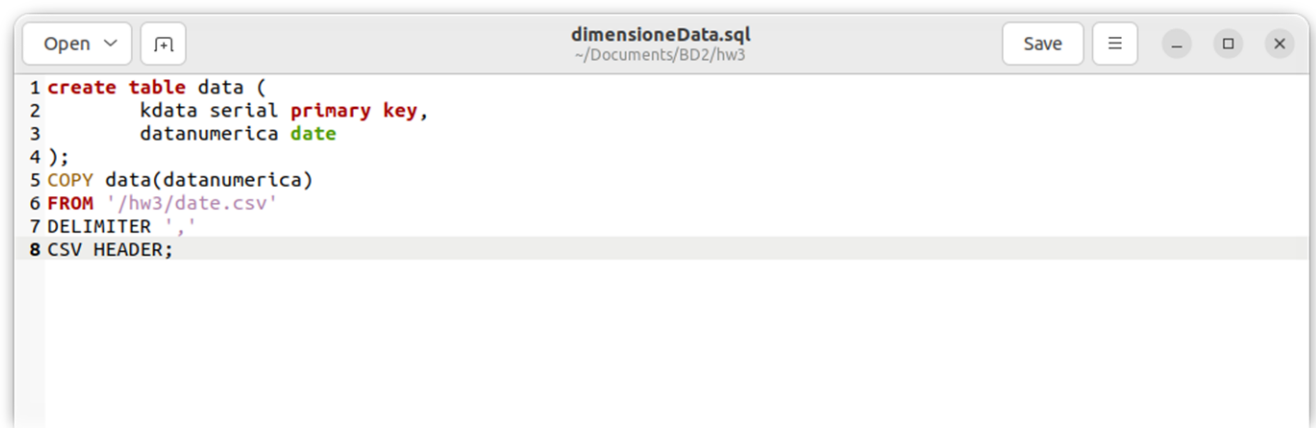
### → SCHEMA DIMENSIONALE

- dimFarmaco (kVersioneFarmaco, kFarmaco, codFarmaco, descrizioneFarmaco, codMolecola, descrizioneMolecola, codCasa, nomeCasa, fascia)
- dimEta (kEta, anni, fascia)
- dimData (kData, data)
- dimAslPaziente (kAslPaziente, codiceAslPaziente, nomeAslPaziente)
- dimAslFarmacia (kAslFarmacia, codiceAslFarmacia, nomeAslFarmacia)
- dimRicetta(kRicetta, codRicetta)
- fatti(codiceAslFarmacia, codiceAslPaziente, data, eta, codiceFarmaco, codiceRicetta, fascia, prezzocomplessivo, quantita)

### → POPOLAZIONE DELLE DIMENSIONI

#### dimData

Tramite un foglio di calcolo Excel sono state generate tutte le date possibili comprese fra 01/01/2014 e 31/12/2015. Queste date sono state poi inserite nella tabella tramite una COPY.



```
1 create table data (
2     kdata serial primary key,
3     datanumerica date
4 );
5 COPY data(datanumerica)
6 FROM '/hw3/date.csv'
7 DELIMITER ','
8 CSV HEADER;
```

## dimEta

Tramite uno script Python le varie età sono state associate alle relative fasce. L'output è stato inserito in un file CSV in quale si può facilmente caricare in una tabella (sempre grazie ad una COPY).



```
1 for x in range(0, 121):
2     if x <=3 :
3         print(str(x)+",0-3",file="out.csv")
4     elif x <=17 :
5         print(str(x)+",3-17")
6     elif x <=30 :
7         print(str(x)+",18-30")
8     elif x <=50 :
9         print(str(x)+",31-50")
10    elif x <=70 :
11        print(str(x)+",51-70")
12    elif x <=90 :
13        print(str(x)+",71-90")
14    else:
15        print(str(x)+",91-120")
```

Python 2 ▾ Tab Width: 8 ▾ Ln 1, Col 1 ▾ INS



```
1 create table eta (
2     keta serial primary key,
3     anni integer,
4     fascia varchar(10)
5 );
6 COPY eta(anni,fascia)
7 FROM 'hw3/out.csv'
8 DELIMITER ','
9 CSV HEADER;
```

## dimFarmaco

Poiché esistono diverse versioni dello stesso farmaco sono state introdotte due chiavi surrogate, la prima relativa alle versioni e la seconda ai farmaci veri e propri. Per realizzare queste chiavi surrogate si è utilizzata una tabella di appoggio con lo scopo di ottenere valori numerici progressivi, in corrispondenza uno a uno con codiceFarmaco.

Si è inoltre resa necessaria, nella staging area, un'operazione di join fra le tabelle farmaco, molecola e casafarmaceutica con opportune ridenominazioni.

```
dimensioneFarmaco.sql
~/Documents/BD2/hw3

1 CREATE TABLE KKVersioniFarmaci (
2   KFarmaco serial primary key,
3   Codice integer,
4   Fascia character(1)
5 );
6 INSERT INTO KKVersioniFarmaci(Codice,Fascia)
7 SELECT F.Codice, F.Fascia
8 FROM Farmaci F
9 WHERE (F.Codice,F.Fascia)
10 NOT IN(SELECT Codice, Fascia FROM KKVersioniFarmaci);
11
12 CREATE TABLE KKFarmaci (
13   KFarmaco serial primary key,
14   Codice integer
15 );
16 INSERT INTO KKFarmaci(Codice)
17 SELECT F.Codice
18 FROM Farmaci F
19 WHERE F.Codice NOT IN(SELECT Codice FROM KKFarmaci);
20
21 CREATE TABLE farmaco
22   kversionefarmaco integer,
23   kfarmaco integer,
24   codfarmaco integer,
25   descrizionefarmaco character varying,
26   codmolecola integer,
27   descrizionemolecola character varying,
28   codcasa integer,
29   nomecasa character varying,
30   fascia character varying
31 );
32
33 INSERT INTO farmaco(kversionefarmaco,kfarmaco,codfarmaco,descrizionefarmaco,codmolecola,descrizionemolecola,codcasa,nomecasa,fascia)
34 SELECT KKF.kfarmaco as kversionefarmaco, KF.kfarmaco, F.Codice as codfarmaco, F.Descrizione as descrizionefarmaco, F.CodMolecola,
35   M.Descrizione as descrizionemolecola,
36   C.CodCasa, C.Nome as nomecasa, F.Fascia
37 FROM Farmaci F join Molecole M on F.CodMolecola=M.CodMolecola natural join Casefarmaceutiche C natural join kkfarmaci KF, kkversionifarmaci
38 WHERE KKF.codice = F.codice AND KKF.fascia = F.fascia
```

## dimRicetta

Si tratta in realtà di una dimensione degenera, contiene solamente le chiavi surrogate inserite con lo stesso sistema utilizzato per i farmaci.

## dimAslpaziente & dimaslFarmacia

Per la popolazione delle due dimensioni relative alle asl dei pazienti e delle farmacie sono stati effettuati i join fra le tabelle Territorio e ASL e poi rispettivamente con Pazienti e Farmacie. Anche in questo caso per la gestione della chiave surrogata si è utilizzata una tabella di appoggio.

```
dimensioneRicetta.sql
~/Documents/BD2/hw3

1 CREATE TABLE ricetta (
2   Kricetta serial primary key,
3   Codricetta integer
4 );
5 INSERT into ricetta(Codricetta)
6 SELECT Numero
7 FROM Ricette
8 WHERE Numero NOT IN(SELECT Codricetta FROM ricetta);
```

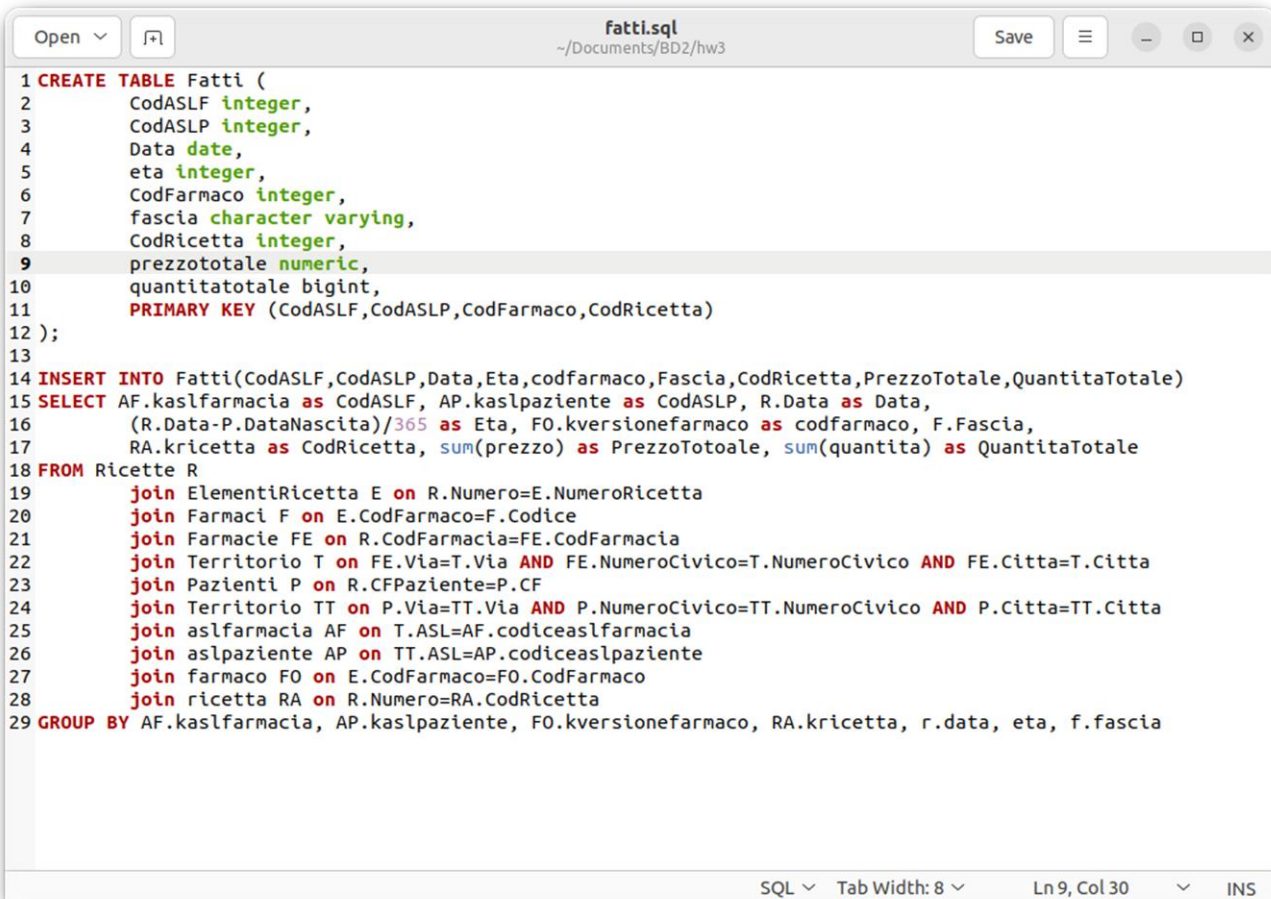
```
dimensioneAsl.sql
~/Documents/BD2/hw3

1 CREATE TABLE KKAsl (
2   Kasl serial primary key,
3   Codice integer
4 );
5 INSERT into KKAsl(Codice)
6 SELECT Codice
7 FROM Asl
8 WHERE Codice NOT IN(SELECT Codice FROM KKAsl);
9
10 CREATE TABLE aslpaziente (
11   kaslpaziente integer,
12   codiceaslpaziente integer,
13   nomeaslpaziente character varying
14 );
15
16 CREATE TABLE aslfarmacia (
17   kaslfarmacia integer,
18   codiceaslfarmacia integer,
19   nomeaslfarmacia character varying
20 );
21
22 INSERT into aslpaziente(kaslpaziente,codiceaslpaziente,nomeaslpaziente)
23 SELECT KKA.kasl as kaslpaziente, A.Codice as codiceaslpaziente, A.Nome as nomeaslpaziente
24 FROM Pazienti P natural join Territorio T join Asl A on A.Codice=T.ASL natural join kkasl KKA;
25
26
27 INSERT into aslfarmacia(kaslfarmacia,codiceaslfarmacia,nomeaslfarmacia)
28 SELECT KKA.kasl as kaslfarmacia, A.Codice as codiceaslfarmacia, A.Nome as nomeaslfarmacia
29 FROM Farmacie F natural join Territorio T join Asl A on A.Codice=T.ASL natural join kkasl KKA
```

SQL Tab Width: 8 Ln 24, Col 95 INS

## → COSTRUZIONE DELLA TABELLA DEI FATTI

È stata generata una tabella dei fatti provvisoria grazie al join fra ElementiRicetta, Ricette, Farmaci, Farmaci, Pazienti. A partire da questa tabella è stato effettuato un doppio join con Territorio, così da ottenere i riferimenti delle ASL sia dei Pazienti che delle Farmacie. È stata calcolata l'età del paziente con una sottrazione fra la data della ricetta e quella di nascita del paziente. Sono state poi calcolate le misure relative ai prezzi complessivi e le quantità. In conclusione, sono stati sostituiti gli identificatori con le chiavi surrogate delle dimensioni.



```
1 CREATE TABLE Fatti (
2     CodASLF integer,
3     CodASLP integer,
4     Data date,
5     eta integer,
6     CodFarmaco integer,
7     fascia character varying,
8     CodRicetta integer,
9     prezzototale numeric,
10    quantitatotale bigint,
11    PRIMARY KEY (CodASLF,CodASLP,CodFarmaco,CodRicetta)
12 );
13
14 INSERT INTO Fatti(CodASLF,CodASLP,Data,Eta,codFarmaco,Fascia,CodRicetta,PrezzoTotale,QuantitaTotale)
15 SELECT AF.kaslfarmacia as CodASLF, AP.kaslpaziente as CodASLP, R.Data as Data,
16        (R.Data-P.DataNascita)/365 as Eta, FO.kversionefarmaco as codfarmaco, F.Fascia,
17        RA.kricetta as CodRicetta, sum(prezzo) as PrezzoTotoale, sum(quantita) as QuantitaTotale
18 FROM Ricette R
19     join ElementiRicetta E on R.Numero=E.NumeroRicetta
20     join Farmaci F on E.CodFarmaco=F.Codice
21     join Farmacie FE on R.CodFarmacia=FE.CodFarmacia
22     join Territorio T on FE.Via=T.Via AND FE.NumeroCivico=T.NumeroCivico AND FE.Citta=T.Citta
23     join Pazienti P on R.CFPaziente=P.CF
24     join Territorio TT on P.Via=TT.Via AND P.NumeroCivico=TT.NumeroCivico AND P.Citta=TT.Citta
25     join aslfarmacia AF on T.ASL=AF.codiceaslfarmacia
26     join aslpaziente AP on TT.ASL=AP.codiceaslpaziente
27     join farmaco FO on E.CodFarmaco=FO.CodFarmaco
28     join ricetta RA on R.Numero=RA.CodRicetta
29 GROUP BY AF.kaslfarmacia, AP.kaslpaziente, FO.kversionefarmaco, RA.kricetta, r.data, eta, f.fascia
```