Date

venerdì 18 novembre 2016 19:01

1. Confronto
   ```
   WHERE Data < '2005-07-15'
   ```
2. Intervallo temporale
   ```
   WHERE Data BETWEEN '2001-01-01' AND '2005-12-31'
   ```
3. Cinque anni fa
   ```
   WHERE YEAR(Data) = YEAR(CURRENT_DATE) - 5
   ```
4. Differenza

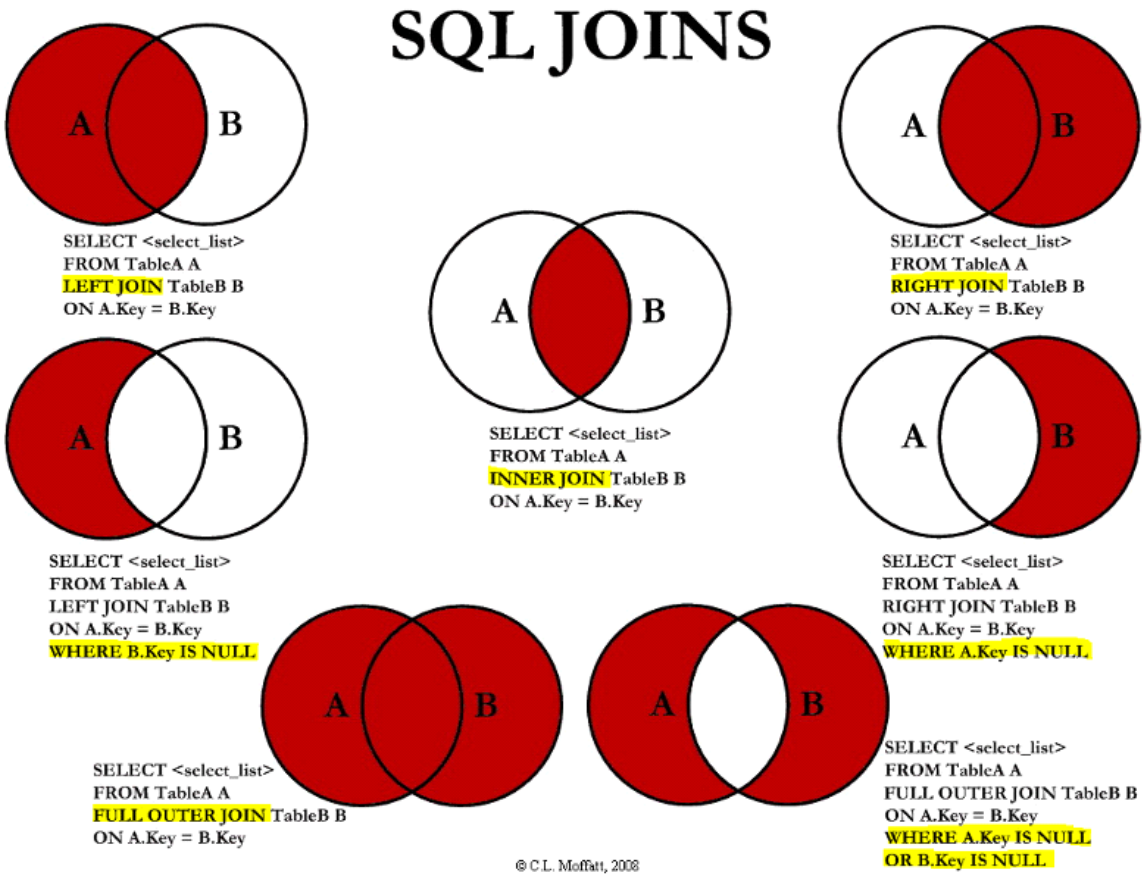
○ Numero di giorni `DATEDIFF(Data1, Data2)`	Età di una persona `SELECT FLOOR(DATEDIFF(CURRENT_DATE, '1994-11-30')/365)`

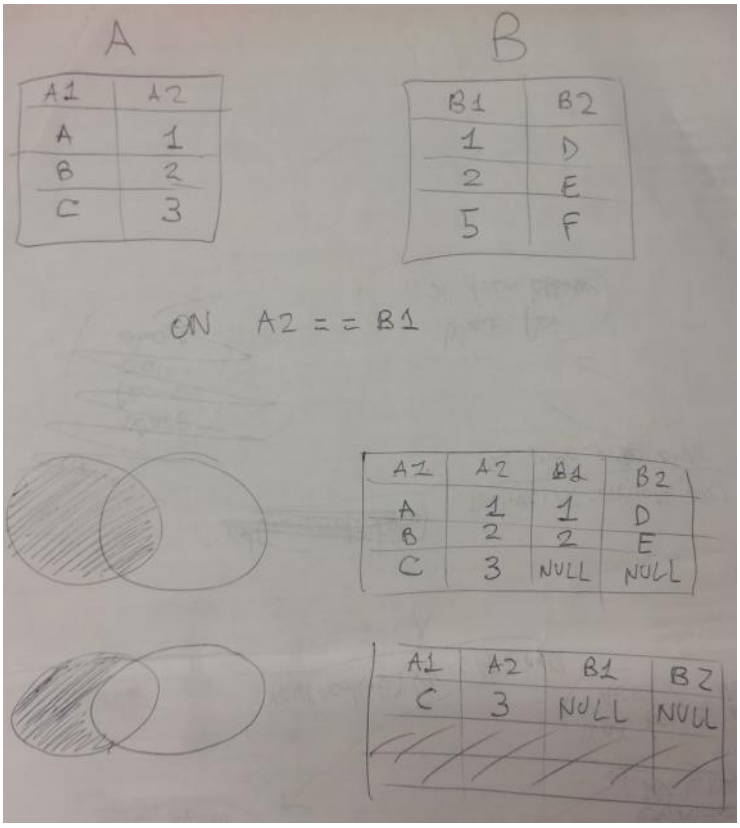
   ○ Numero di mesi
   ```
   PERIOD_DIFF(Data1, Data2)
   ```
5. Formattazione
   ```
   DATE_FORMAT(Data, '%Y%m')
   ```
6. Somma
   ```
   DATE_ADD(Data, INTERVAL 5 YEAR)
   ```
   oppure
   ```
   Data + INTERVAL 5 YEAR -- Controllo età
   ```
7. Sottrazione
   ```
   DATE_SUB(Data, INTERVAL 5 YEAR)
   ```
   oppure
   ```
   Data - INTERVAL 5 YEAR
   ```
8. Giorno della settimana
   ```
   DAYOFWEEK(Data)
   ```

Join

giovedì 20 marzo 2014 13:47

- **Join naturale:** fa l'unione di due tabelle unendo i valori uguali sugli attributi che hanno lo stesso nome.
  `T1 NATURAL JOIN T2`
- **Theta join:** fa l'unione di due tabelle su una condizione.
  `T1 INNER JOIN T2 ON A1 = A2`
  - **Using**: Fa un join naturale solo su determinati attributi omonimi elencati.
    `T1 INNER JOIN T2 USING(AttributiElencati)`
  - **Self join:** combina le tuple della stessa tabella in base a certe condizioni.
    `Tabella T1 INNER JOIN Tabella T2 ON T2.Data < T1.Data`
- **Prodotto cartesiano:** prodotto cartesiano senza condizione.
  `T1 CROSS JOIN T2`
- **Join esterni:** se non c'è corrispondenza, mantiene la riga specificata (LEFT o RIGHT) e riempie la tabella non indicata con valori NULL.
  `T1 LEFT OUTER JOIN T2 ON A1 = A2`
  `T1 RIGHT OUTER JOIN T2 ON A1 = A2`
- **Join multiplo**
  `INNER JOIN … ON … INNER JOIN … ON …`




A
A1 A2
A 1
B 2
C 3
B
B1 B2
1 D
2 E
5 F
ON A2 = = B1
A1 A2 B1 B2
A 1 1 D
B 2 2 E
C 3 NULL NULL
A1 A2 B1 B2
C 3 NULL NULL

Views, temporary tables, …

mercoledì 18 gennaio 2017 10:21

	Persistente	Ricalcolo all'utilizzo	Utilizzo	Creazione

View	No	Sì	Scomposizione di query.	`CREATE VIEW ViewName AS`
				`  SELECT *`
				`  FROM Table;`

Temporary table	No	No
		Tre tipi di refresh:
		• *Immediate: trigger*
		• *Deferred: temporal trigger (event)*
		• *On demand: stored procedure*
			Risultati multi-tupla di funzioni.	`-- Uso per risultato multi-tupla`
				`DROP PROCEDURE IF EXISTS ProcedureName;`
				`DELIMITER $$`
				`CREATE PROCEDURE ProcedureName([…])`
				`BEGIN`
				`  CREATE TEMPORARY TABLE IF NOT EXISTS _TempTableName(`
				`  Attribute1 CHAR(50) NOT NULL,`
				`  Attribute2 INT(11) NOT NULL DEFAULT 0,`
				`  PRIMARY KEY(Attribute1)`
				`  ) ENGINE=InnoDB DEFAULT CHARSET=latin1;`
				
				`  -- oppure`
				`  CREATE TEMPORARY TABLE IF NOT EXISTS _TempTableName`
				`  LIKE AnotherSampleTable;`
				
				`  TRUNCATE _TempTableName;`
				
				`  INSERT INTO _TempTableName`
				`  SELECT *`
				`  FROM Table;`
				`END $$`
				`DELIMITER ;;`
				
				`-- Chiamata e visualizzazione del risultato`
				`CALL ProcedureName([…]);`
				`SELECT * FROM _TempTableName;`

Materialized view (*tabella in MySQL*) (*snapshot*)	Sì	No
		Tre tipi di refresh:
		• *Immediate: trigger*
		• *Deferred: temporal trigger (event)*
		• *On demand: stored procedure*
			Per ottenere una risposta veloce, anche non aggiornata.
			La query che riempie la materialized view di solito è lenta e complessa.
				`CREATE TABLE MAT_VIEW_NAME(`
				`  Attribute1 CHAR(50) NOT NULL,`
				`  Attribute2 INT(11) NOT NULL DEFAULT 0,`
				`  PRIMARY KEY(Attribute1)`
				`) ENGINE=InnoDB DEFAULT CHARSET=latin1;`

Procedures & functions

lunedì 30 gennaio 2017 17:11

	Sintassi	Dove può essere chiamata
Stored procedures	DELIMITER $$ DROP PROCEDURE IF EXISTS *procedureName*; CREATE PROCEDURE *procedureName*(IN *param1* INT(11), OUT *param2* CHAR(50)) BEGIN Variabili DECLARE *var1* INTEGER DEFAULT 0; DECLARE *var2* VARCHAR(255) DEFAULT ''; Cursori DECLARE *cursorName* CURSOR FOR -- *Query*; OPEN *cursorName*; Loops *loopName*: LOOP FETCH *cursorName* INTO *var2*; -- ... END LOOP *loopName*; CLOSE *cursorName*; END $$ DELIMITER ;	Fuori da altri pezzi di codice.
Functions	DELIMITER $$ DROP FUNCTION IF EXISTS *functionName*; CREATE FUNCTION *functionName*(*param1* INT, *param2* CHAR(50)) RETURNS VARCHAR(6) [NOT] DETERMINISTIC BEGIN Variabile risultato DECLARE *result* VARCHAR(6) DEFAULT ''; RETURN (*result*); END $$ DELIMITER ;	• Query • Stored procedure • Trigger • Event

Events & triggers

domenica 05 febbraio 2017 17:05

	Sintassi

Triggers	DROP TRIGGER IF EXISTS *TriggerName*;
	DELIMITER $$
	CREATE TRIGGER *TriggerName*
	{BEFORE, AFTER} {INSERT, UPDATE, DELETE} ON *TableName* FOR EACH ROW
	BEGIN
	Errori
	IF(*Condition*) THEN
	SIGNAL SQLSTATE '45000'
	SET MESSAGE_TEXT = '*Message*';
	END IF;
	END $$
	DELIMITER ;

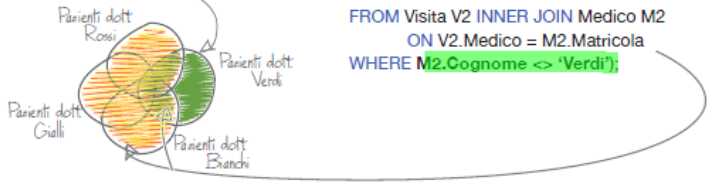
Events (*temporal triggers*)	DROP EVENT IF EXISTS *EventName*; DELIMITER $$ CREATE EVENT *EventName* ON SCHEDULE EVERY 3 {DAY, MONTH, …} DO BEGIN END $$ DELIMITER ;

Analisi del testo

mercoledì 25 gennaio 2017 11:22

	Proposizioni	Operatori
Appartenenza	• i medici **che**... • il numero di visite **effettuate** dai medici di Pisa... • i pazienti **aventi** il reddito più alto del reddito medio...	• IN • INNER JOIN • WHERE >, <, …
Esclusione	• i medici **che non**... • il numero di visite **non effettuate** dai medici di Pisa... • i pazienti **che non hanno superato** dieci visite...	• NOT IN • OUTER JOIN • WHERE >, <, …
Unione	• i pazienti visitati dal dott. Verdi **o** dal dott. Rossi • i medici che hanno visitato **sia** un paziente di Pisa **che** un paziente di Siena • i medici che non visitano il giovedì **oppure** il sabato	• OR • UNION (elimina i duplicati) I pazienti visitati dal dott. Verdi **o** dal dott. Rossi La disgiunzione **non** esclude l'intersezione • UNION ALL (conserva i duplicati)
Differenza	• tranne • eccetto che • solamente • esclusivamente • ma non	• NOT IN • OUTER JOIN
Esclusività	• solamente • soli/e	• Pazienti visitati **solamente** dal dott. Verdi • Dose giornaliera media dei farmaci indicati per la cura di **sole** patologie intestinali

```
SELECT DISTINCT V.Paziente
FROM Visita V INNER JOIN Medico M
        ON V.Medico = M.Matricola
WHERE M.Cognome = 'Verdi'
        AND V.Paziente NOT IN (
                SELECT  Paziente
                FROM Visita V2 INNER JOIN Medico M2
                        ON V2.Medico = M2.Matricola
                WHERE M2.Cognome <> 'Verdi');
```




```
SELECT AVG(I1.DoseGiornaliera)
FROM Indicazione I1
WHERE I1.Farmaco NOT IN (
     SELECT I2.Farmaco
     FROM Indicazione I2 INNER JOIN Patologia P
          ON I2.Patologia = P.Nome
     WHERE P.ParteCorpo <> 'Intestino'
);
```

Tutti gli esercizi 1

mercoledì 01 febbraio 2017 09:55

L'ultimo esame dell'anno accademico 2013/2014 è quello del 25 Febbraio 2015.
Negli esami dal 10 Giugno 2015 in poi, vanno considerati gli esercizi dell' "a.a. precedente".

Database

PAZIENTE(CodFiscale, Cognome, Nome, Sesso, DataNascita, Citta, Reddito)
MEDICO(Matricola, Cognome, Nome, Specializzazione, Parcella, Citta)
FARMACO(NomeCommerciale, PrincipioAttivo, Costo, Pezzi)
PATOLOGIA(Nome, ParteCorpo, SettoreMedico, Invalidita, PercEsenzione)
INDICAZIONE(Farmaco, Patologia, DoseGiornaliera, NumGiorni, AVita)
VISITA(Medico, Paziente, Data, Mutuata)
ESORDIO(Paziente, Patologia, DataEsordio, DataGuarigione, Gravita, Cronica)
TERAPIA(Paziente, Patologia, DataEsordio, Farmaco, DataInizioTerapia, DataFineTerapia, Posologia)

Esercizio 1

Scrivere una query che, considerata ciascuna parte del corpo, ne restituisca il nome, e i principi attivi contenuti in farmaci indicati solamente per la cura di patologie a carico di tale parte del corpo.

```
SELECT DISTINCT F.PrincipioAttivo,
                P.ParteCorpo
FROM Indicazione I
     INNER JOIN
     Patologia P ON I.Patologia = P.Nome
     INNER JOIN
     Farmaco F ON I.Farmaco = F.NomeCommerciale
GROUP BY I.Farmaco
HAVING COUNT(DISTINCT P.ParteCorpo) = 1;
```

Esercizio 1

Scrivere una query che restituisca, se esiste, la città dalla quale proviene il maggior numero di pazienti che hanno contratto l'acufene un numero di volte maggiore o uguale a quello degli altri pazienti della loro città.

```sql
CREATE OR REPLACE VIEW EsordiAcufene AS
SELECT P.CodFiscale,
       P.Citta,
       COUNT(*) AS NumeroEsordi
FROM Esordio E
     INNER JOIN
     Paziente P ON E.Paziente = P.CodFiscale
WHERE E.Patologia = 'Acufene'
GROUP BY P.CodFiscale, P.Citta;

CREATE OR REPLACE VIEW PazientiCitta
SELECT EA.Citta,
       COUNT(*) AS NumeroPazienti
FROM EsordiAcufene EA
WHERE EA.NumeroEsordi >= ALL
      (
       SELECT EA2.NumeroEsordi
       FROM EsordiAcufene EA2
       WHERE EA2.Citta = EA.Citta
      )
GROUP BY EA.Citta;

SELECT PC.Citta
FROM PazientiCitta PC
WHERE PC.NumeroPazienti > ALL
      (
       SELECT PC2.NumeroPazienti
       FROM PazientiCitta PC2
       WHERE PC2.Citta <> PC.Citta
      );
```

Esercizio 1

Scrivere una query che restituisca le patologie curate sempre con il farmaco meno costoso fra tutti quelli indicati. Se, data una patologia, esiste più di un farmaco meno costoso, questi possono essere stati usati intercambiabilmente.

```
CREATE OR REPLACE VIEW CostiMinimi AS
SELECT I.Patologia,
       MIN(F.Costo) AS Costo
FROM Indicazione I
     INNER JOIN
     Farmaco F ON I.Farmaco = F.NomeCommerciale
GROUP BY I.Patologia;

SELECT T.Patologia
FROM Terapia T
     INNER JOIN
     Farmaco F ON T.Farmaco = F.NomeCommerciale
     NATURAL JOIN
     CostiMinimi CM
GROUP BY T.Patologia
HAVING COUNT(*) = (SELECT COUNT(*)
                   FROM Terapia T2
                   WHERE T2.Patologia = T.Patologia);
```

Esercizio 1

Scrivere una query che elimini tutti gli esordi di otite contratta e curata con successo prima di cinque anni fa, relativi ai soli pazienti che hanno contratto nuovamente, negli ultimi cinque anni, la stessa patologia.

```
DELETE E.*
FROM Esordio E
     NATURAL JOIN
     (
      SELECT *
      FROM Esordio E2
      WHERE E2.Patologia = 'Otite'
            AND E2.DataGuarigione IS NOT NULL
            AND YEAR(E2.DataGuarigione) < YEAR(CURRENT_DATE) - 5
            AND EXISTS
                (
                 SELECT *
                 FROM Esordio E3
                 WHERE E3.Paziente = E2.Paziente
                       AND E3.Patologia = E2.Patologia
                       AND YEAR(E3.DataEsordio) > YEAR(CURRENT_DATE) - 5
                )
     ) AS Target;
```

Esercizio 1

Scrivere una query che restituisca nome e cognome del medico che, al 31/12/2014, aveva visitato un numero di pazienti superiore a quelli visitati da ciascun medico della sua stessa specializzazione.

```
CREATE OR REPLACE VIEW PazientiVisitati AS
SELECT V.Medico, M.Specializzazione,
       COUNT(DISTINCT V.Paziente) AS NumeroPazienti
FROM Visita V INNER JOIN Medico M ON V.Medico = M.Matricola
WHERE V.Data < '2014-12-31'
GROUP BY V.Medico;

SELECT M.Nome, M.Cognome
FROM Medico M INNER JOIN PazientiVisitati PV
     ON M.Matricola = PV.Medico
WHERE PV.NumeroPazienti >= ALL
      (SELECT PV2.NumeroPazienti
       FROM PazientiVisitati PV2
       WHERE PV2.Specializzazione = PV.Specializzazione
      );
```

Esercizio 2

Scrivere una query che restituisca per ciascun principio attivo, il nome del principio attivo e il nome commerciale di ogni farmaco utilizzato almeno una volta per tutte le patologie per le quali è indicato. Il risultato è formato da row (PrincipioAttivo, NomeCommerciale), una per ogni farmaco che rispetta la condizione.

```
SELECT F.PrincipioAttivo, T.Farmaco
FROM Terapia T INNER JOIN Farmaco F ON T.Farmaco = F.NomeCommerciale
GROUP BY T.Farmaco, F.PrincipioAttivo
HAVING COUNT(DISTINCT T.Patologia) =
         (SELECT COUNT(*)
          FROM Indicazione I
          WHERE I.Farmaco = T.Farmaco
         )
ORDER BY F.PrincipioAttivo;

/* ORDER BY non è necessario, ma aiuta a interpretare meglio il risultato
richiesto a livello visivo */
```

Esercizio 1

Scrivere una query che restituisca la dose giornaliera media dei farmaci indicati per la cura di sole patologie intestinali.

```
SELECT AVG(I1.DoseGiornaliera)
FROM Indicazione I1
WHERE I1.Farmaco NOT IN(
      SELECT I2.Farmaco
      FROM Indicazione I2 INNER JOIN Patologia P ON I2.Patologia = P.Nome
      WHERE P.ParteCorpo <> 'Intestino'
);
```

Esercizio 2

Scrivere una query che restituisca, per il sesso maschile e per quello femminile, rispettivamente, il numero di pazienti attualmente affetti da ipertensione, trattata con lo stesso farmaco da più di venti anni.

```
SELECT P.Sesso, COUNT(DISTINCT P.CodFiscale)
FROM Esordio E INNER JOIN Paziente P ON E.Paziente = P.CodFiscale
WHERE E.DataGuarigione IS NULL
      AND E.Patologia = 'Ipertensione'
      AND EXISTS(
           SELECT *
           FROM Terapia T
           WHERE T.Paziente = E.Paziente
                 AND T.Patologia = E.Patologia
                 AND T.DataEsordio = E.DataEsordio
                 AND T.DataFineTerapia IS NULL
                 AND T.DataInizioTerapia < CURRENT_DATE - INTERVAL 20 YEAR
          )
GROUP BY P.Sesso;
```

Esercizio 1

Scrivere una query che restituisca la parte del corpo maggiormente colpita da patologie con invalidità superiore al 70%. In caso di pari merito, restituire tutte le parti del corpo.

```
CREATE OR REPLACE VIEW PatologieAltaInvalidita AS
SELECT Nome AS Patologia, ParteCorpo
FROM Patologia
WHERE Invalidita > 70;

SELECT PAI.ParteCorpo
FROM Esordio E NATURAL JOIN PatologieAltaInvalidita PAI
GROUP BY PAI.ParteCorpo
HAVING COUNT(*) >= ALL (
       SELECT COUNT(*) AS NumeroEsordi
       FROM Esordio E2 NATURAL JOIN PatologieAltaInvalidita PAI2
       GROUP BY PAI2.ParteCorpo);
```

Esercizio 2

Scrivere una query che restiuisca il numero di terapie iniziate da ciascun paziente in ogni mese dell'anno. Nel risultato devono comparire tutti i pazienti e tutti i mesi dell'anno.

```
SELECT PM.CodFiscale, PM.Mese, IF(P.CodFiscale IS NULL, 0, COUNT(*))
FROM Terapia T RIGHT OUTER JOIN Paziente P ON T.Paziente = P.CodFiscale
     RIGHT OUTER JOIN (SELECT P.CodFiscale, D.Mese
                       FROM Paziente P CROSS JOIN
                          (SELECT DISTINCT MONTH(DataInizioTerapia) AS Mese
                           FROM Terapia) AS D
                      ) AS PM
     ON (PM.CodFiscale = P.CodFiscale
         AND PM.Mese = MONTH(T.DataInizioTerapia))
GROUP BY PM.CodFiscale, PM.Mese;
```

Esercizio 1

Senza fare uso di join né viste, scrivere una query che restituisca nome e cognome dei medici che hanno visitato tutti i pazienti di Roma.

```
SELECT M.Nome, M.Cognome
FROM Medico M
WHERE NOT EXISTS(
      SELECT *
      FROM Paziente P
      WHERE P.Citta = 'Roma'
            AND NOT EXISTS(
                SELECT *
                FROM Visita V
                WHERE V.Paziente = P.CodFiscale
                      AND V.Medico = M.Matricola
            )
     );
```

Esercizio 2

Scrivere una query che restituisca le patologie che nel 2011 hanno colpito, con tasso d'incidenza complessivo superiore al 90%, bambini di età inferiore a 5 anni o anziani di età superiore a 75 anni.

```
SELECT E.Patologia
FROM Paziente P INNER JOIN Esordio E
     ON P.CodFiscale = E.Paziente
WHERE YEAR(E.DataEsordio) = 2011
      AND (P.DataNascita + INTERVAL 5 YEAR > E.DataEsordio
           OR P.DataNascita + INTERVAL 75 YEAR < E.DataEsordio)
GROUP BY E.Patologia
HAVING COUNT(DISTINCT E.Paziente) >
       0.9*(SELECT COUNT(*)
            FROM Paziente P
            WHERE YEAR(P.DataNascita + INTERVAL 5 YEAR) > 2011
                  OR YEAR(P.DataNascita + INTERVAL 75 YEAR) > 2011
            );
```

Esercizio 1

Scrivere una query che, relativamente a ciascun mese del 2013, restituisca il mese (come intero da 1 a 12) e il numero medio di terapie per esordio effettuate dai pazienti per combattere il dolore.

```
SELECT MONTH(D.DataEsordio) AS Mese, AVG(NumeroTerapie) AS TerapieMedie
FROM(
      SELECT E.Paziente, E.DataEsordio, COUNT(*) AS NumeroTerapie
      FROM Esordio E NATURAL JOIN Terapia T
      WHERE YEAR(E.DataEsordio) = 2013
            AND E.Patologia = 'Dolore'
      GROUP BY E.Paziente, E.DataEsordio
    ) AS D
GROUP BY MONTH(D.DataEsordio);
```

Esercizio 2

Scrivere una query che restituisca nome e cognome dei pazienti che, esclusivamente per le patologie ortopediche, hanno assunto, man mano nella vita, tutti i farmaci a base di nimesulide.

```
CREATE OR REPLACE VIEW PazientiSettoreMedicoNimesulide AS
SELECT E.Paziente, P.SettoreMedico
FROM Esordio E NATURAL JOIN Terapia T
     INNER JOIN Patologia P ON E.Patologia = P.Nome
     INNER JOIN Farmaco F ON T.Farmaco = F.NomeCommerciale
WHERE F.PrincipioAttivo = 'Nimesulide'
GROUP BY E.Paziente, P.SettoreMedico
HAVING COUNT(DISTINCT T.Farmaco) = (SELECT COUNT(*)
                                    FROM Farmaco
                                    WHERE PrincipioAttivo = 'Nimesulide');

SELECT P.Nome, P.Cognome
FROM PazientiSettoreMedicoNimesulide PSMN INNER JOIN Paziente P
     ON PSMN.Paziente = P.CodFiscale
WHERE PSMN.SettoreMedico = 'Ortopedia'
      AND NOT EXISTS (SELECT *
                      FROM PazientiSettoreMedicoNimesulide PSMN2
                      WHERE PSMN2.Paziente = PSMN.Paziente
                            AND PSMN2.SettoreMedico <> PSMN.SettoreMedico);
```

Esercizio 1

Scrivere una query che, considerate le sole terapie finalizzate alla cura di patologie cardiache, restituisca, per ciascuna di esse, il nome della patologia e il farmaco più utilizzato per curarla. La soluzione proposta deve presupporre che, data una patologia cardiaca, tale farmaco possa non essere unico.

```
CREATE OR REPLACE VIEW TotaliTerapie AS
SELECT T.Patologia, T.Farmaco, COUNT(*) AS NumeroTerapie
FROM Terapia T INNER JOIN Patologia P ON T.Patologia = P.Nome
WHERE P.ParteCorpo = 'Cuore'
GROUP BY T.Patologia, T.Farmaco;

SELECT TT.Patologia, TT.Farmaco
FROM TotaliTerapie TT
WHERE TT.NumeroTerapie >= ALL (SELECT TT2.NumeroTerapie
                               FROM TotaliTerapie TT2
                               WHERE TT2.Patologia = TT.Patologia);
```

Esercizio 2 [stessa soluzione per l'esercizio 3, anno accademico precedente]

Scrivere una query che restituisca nome, cognome e reddito dei pazienti di sesso femminile che al 15 Giugno 2010 risultavano affetti, oltre alle eventuali altre, da un'unica patologia cronica, con invalidità superiore al 50%, e non l'avevano mai curata con alcun farmaco fino a quel momento.

```
SELECT P.Nome, P.Cognome, P.Reddito
FROM  Esordio E INNER JOIN Paziente P ON E.Paziente = P.CodFiscale
      INNER JOIN Patologia PA ON E.Patologia = PA.Nome
WHERE E.DataEsordio < '2010-06-15'
      AND PA.Invalidita > 50
      AND E.DataGuarigione IS NULL -- ridondante, la malattia è cronica
      AND E.Cronica = 'si'
      AND NOT EXISTS (SELECT *
                      FROM Terapia T
                      WHERE T.Paziente = E.Paziente
                            AND T.Patologia = E.Patologia
                            AND T.DataInizioTerapia < '2010-06-15'
                     )
GROUP BY E.Paziente, P.Nome, P.Cognome, P.Reddito
HAVING COUNT(*) = 1;
```

Esercizio 1

Scrivere una query che restituisca il nome commerciale di ciascun farmaco utilizzato da almeno un paziente per curare tutte le patologie per le quali è indicato.

```
SELECT DISTINCT T.Farmaco
FROM Terapia T
GROUP BY T.Paziente, T.Farmaco
HAVING COUNT(DISTINCT T.Patologia) = (SELECT COUNT(*)
                                      FROM Indicazione I
                                      WHERE I.Farmaco = T.Farmaco);
```

Esercizio 2 (7 punti)

Scrivere una query che restituisca il numero di pazienti visitati solo da medici specializzati in cardiologia o neurologia, almeno due volte per ciascuna delle due specializzazioni. Si scriva la query senza usare viste.

```
SELECT FLOOR(COUNT(*)/2) AS NumeroPazienti
FROM(
      SELECT Specializzazione, Paziente
      FROM Visita V INNER JOIN Medico M ON V.Medico = M.Matricola
      WHERE V.Paziente NOT IN(
            SELECT V2.Paziente
            FROM Visita V2 INNER JOIN Medico M2
                 ON V2.Medico = M2.Matricola
            WHERE M2.Specializzazione <> 'Cardiologia'
                  AND M2.Specializzazione <> 'Neurologia')
     GROUP BY M.Specializzazione, V.Paziente
     HAVING COUNT(*) >= 2
    ) AS D
GROUP BY Paziente
HAVING COUNT(*) = 2;
```

Esercizio 1

Considerato ciascun principio attivo, indicarne il nome e il costo medio al pezzo fra tutti i farmaci che lo contengono.

```
SELECT F.PrincipioAttivo, AVG(F.Costo/F.Pezzi) AS CostoMedioPezzo
FROM Farmaco F
GROUP BY F.PrincipioAttivo;
```

Esercizio 2

Indicare nome e cognome dei pazienti che hanno contratto almeno due volte tutte le patologie intestinali.

```
CREATE OR REPLACE VIEW DoppiEsordi AS
SELECT E.Patologia, E.Paziente
FROM Patologia PA INNER JOIN Esordio E
     ON PA.Nome = E.Patologia
WHERE PA.ParteCorpo = 'Intestino'
GROUP BY E.Patologia, E.Paziente
HAVING COUNT(*) >= 2;

SELECT P.Nome, P.Cognome
FROM DoppiEsordi DE INNER JOIN Paziente P
     ON DE.Paziente = P.CodFiscale
GROUP BY DE.Paziente, P.Nome, P.Cognome
HAVING COUNT(*) = (SELECT COUNT(*)
                   FROM Patologia
                   WHERE ParteCorpo = 'Intestino');
```

Esercizio 3

Indicare le patologie esordite esclusivamente in forma cronica, curate con il farmaco *Lyrica*.

```
SELECT DISTINCT E.Patologia
FROM Esordio E NATURAL JOIN Terapia T
WHERE T.Farmaco = 'Lyrica'
      AND E.Patologia NOT IN( SELECT E2.Patologia
                              FROM Esordio E2
                              WHERE E2.Cronica = 'no');
```

Esercizio 1

Indicare nome e cognome di ciascun medico che ha visitato tutti i pazienti della sua città.

```
SELECT M.Nome, M.Cognome
FROM Medico M
WHERE EXISTS(SELECT *
             FROM Paziente P1
             WHERE P1.Citta = M.Citta)
      AND NOT EXISTS(SELECT *
                     FROM Paziente P2
                     WHERE P2.Citta = M.Citta
                           AND NOT EXISTS(
                               SELECT *
                               FROM Visita V
                               WHERE V.Medico = M.Matricola
                                     AND V.Paziente = P2.CodFiscale)
                    );
```

Esercizio 3

Indicare il reddito massimo fra quelli di tutti i pazienti che, nell'anno 2011, hanno effettuato esattamente tre visite, ognuna delle quali con un medico avente specializzazione diversa dagli altri.

```
SELECT MAX(D.Reddito)
FROM(
      SELECT P.Reddito
      FROM Visita V INNER JOIN Paziente P ON V.Paziente = P.CodFiscale
           INNER JOIN Medico M ON V.Medico = M.Matricola
      WHERE YEAR(V.Data) = 2011
      GROUP BY P.CodFiscale, P.Reddito
      HAVING COUNT(*) = 3 AND COUNT(DISTINCT M.Specializzazione) = 3
    ) AS D;
```

Esercizio 1

Considerato ogni principio attivo, indicarne il nome e il numero medio di giorni per cui sono indicati i farmaci che lo contengono.

```
SELECT F.PrincipioAttivo, AVG(I.NumGiorni) NumeroMedioGiorni
FROM Farmaco F INNER JOIN Indicazione I
     ON F.NomeCommerciale = I.Farmaco
WHERE I.AVita IS NULL
GROUP BY F.PrincipioAttivo;
```

Esercizio 2

Indicare nome e cognome dei pazienti che, per curare gli esordi di almeno una patologia, hanno complessivamente assunto tutti i farmaci assunti da almeno un paziente per curare tale patologia.

```
SELECT DISTINCT P.Nome, P.Cognome
FROM Terapia T INNER JOIN Paziente P
     ON T.Paziente = P.CodFiscale
GROUP BY T.Paziente, T.Patologia
HAVING COUNT(DISTINCT T.Farmaco) = (SELECT COUNT(DISTINCT T2.Farmaco)
                                   FROM Terapia T2
                                   WHERE T2.Patologia = T.Patologia);
```

Esercizio 1

Indicare nome e cognome dei pazienti che hanno contratto o sono guariti da almeno una patologia almeno una volta in ciascun mese dell'anno 2005.

```
SELECT P.Nome, P.Cognome
FROM Paziente P INNER JOIN(
     SELECT E.Paziente, MONTH(E.DataEsordio) AS Mese
     FROM Esordio E
     WHERE YEAR(E.DataEsordio) = 2005
     UNION
     SELECT E.Paziente, MONTH(E.DataGuarigione) AS Mese
     FROM Esordio E
     WHERE YEAR(E.DataGuarigione) = 2005) AS D ON D.Paziente = P.CodFiscale
GROUP BY P.CodFiscale, P.Nome, P.Cognome
HAVING COUNT(DISTINCT Mese) = 12;
```

Esercizio 2

Per ciascun medico di Firenze che ha effettuato almeno una visita, indicare il suo nome e cognome, e il numero di pazienti visitati di ciascuna città.

```
SELECT M.Nome, M.Cognome, P.Citta,
       COUNT(DISTINCT P.CodFiscale) AS QuantiPazienti
FROM Visita V INNER JOIN Medico M ON V.Medico = M.Matricola
     INNER JOIN Paziente P ON V.Paziente = P.CodFiscale
WHERE M.Citta = 'Firenze'
GROUP BY M.Matricola, M.Nome, M.Cognome, P.Citta;
```

Esercizio 2

Per ciascuna specializzazione, indicarne il nome, e il nome e cognome del medico che ha visitato più pazienti mai visitati in precedenza da altri medici della stessa specializzazione.

```
CREATE OR REPLACE VIEW PrimeVisite AS
SELECT V1.Medico, M1.Nome, M1.Cognome, M1.Specializzazione,
       COUNT(DISTINCT V1.Paziente) AS QuantiPazienti
FROM Visita V1 INNER JOIN Medico M1 ON V1.Medico = M1.Matricola
WHERE NOT EXISTS( SELECT *
                  FROM Visita V2 INNER JOIN  Medico M2
                       ON V2.Medico = M2.Matricola
                  WHERE V2.Paziente = V1.Paziente
                        AND M2.Specializzazione = M1.Specializzazione
                        AND V2.Data < V1.Data)
GROUP BY V1.Medico, M1.Nome, M1.Cognome, M1.Specializzazione;

SELECT PV.Specializzazione, PV.Nome, PV.Cognome
FROM PrimeVisite PV INNER JOIN(
         SELECT Specializzazione, MAX(QuantiPazienti) AS MassimoPrimeVisite
         FROM PrimeVisite
         GROUP BY Specializzazione) AS Massimi
     ON PV.Specializzazione = Massimi.Specializzazione
WHERE PV.QuantiPazienti = Massimi.MassimoPrimeVisite;
```

Esercizio 3

Relativamente ad ogni farmaco indicato per la cura dell'insonnia, indicarne il nome commerciale e la differenza percentuale fra la posologia media di tutte le terapie effettuate con ciascuno di essi, rispetto alla rispettiva dose giornaliera consigliata nelle indicazioni.

```
SELECT T.Farmaco, ((AVG(T.Posologia)/(SELECT DoseGiornaliera
                                      FROM Indicazione
                                      WHERE Farmaco = T.Farmaco))*100)-100
FROM Terapia T
WHERE T.Patologia = 'Insonnia'
GROUP BY T.Farmaco;
```

Esercizio 4

Considerata ciascuna patologia a carico dello stomaco, indicarne il nome, e il principio attivo contenuto in più farmaci indicati per la cura della stessa patologia.

```
CREATE OR REPLACE VIEW FarmaciStomaco AS
SELECT P.Nome, F.PrincipioAttivo, COUNT(*) AS QuantiFarmaci
FROM Patologia P INNER JOIN Indicazione I ON P.Nome = I.Patologia
     INNER JOIN Farmaco F ON I.Farmaco = F.NomeCommerciale
WHERE P.ParteCorpo='Stomaco'
GROUP BY P.Nome, F.PrincipioAttivo;

SELECT FS1.Nome, FS1.PrincipioAttivo
FROM FarmaciStomaco FS1
WHERE FS1.QuantiFarmaci = (SELECT MAX(FS2.QuantiFarmaci)
                           FROM FarmaciStomaco FS2
                           WHERE FS2.Nome = FS1.Nome
);
```

Esercizio 1

Indicare il nome dei farmaci mai assunti prima dei venti anni d'età.

```
SELECT DISTINCT T1.Farmaco
FROM Terapia T1
WHERE T1.Farmaco NOT IN(
      SELECT T2.Farmaco
      FROM Terapia T2 INNER JOIN Paziente P ON T2.Paziente = P.CodFiscale
      WHERE T2.DataInizioTerapia <= P.DataNascita + INTERVAL 20 YEAR
);
```

Esercizio 2

Indicare nome e cognome dei pazienti che hanno curato sempre la stessa patologia con lo stesso farmaco, per tutte le patologie contratte.

```
SELECT Nome, Cognome
FROM Paziente
WHERE CodFiscale NOT IN(SELECT E.Paziente
                        FROM Esordio E NATURAL JOIN Terapia T
                        GROUP BY E.Paziente, E.Patologia
                        HAVING COUNT(DISTINCT T.Farmaco) > 1
);
```

Solo/solamente/esclusivamente

lunedì 06 febbraio 2017 09:37

Database

PAZIENTE(CodFiscale, Cognome, Nome, Sesso, DataNascita, Citta, Reddito)
MEDICO(Matricola, Cognome, Nome, Specializzazione, Parcella, Citta)
FARMACO(NomeCommerciale, PrincipioAttivo, Costo, Pezzi)
PATOLOGIA(Nome, ParteCorpo, SettoreMedico, Invalidita, PercEsenzione)
INDICAZIONE(Farmaco, Patologia, DoseGiornaliera, NumGiorni, AVita)
VISITA(Medico, Paziente, Data, Mutuata)
ESORDIO(Paziente, Patologia, DataEsordio, DataGuarigione, Gravita, Cronica)
TERAPIA(Paziente, Patologia, DataEsordio, Farmaco, DataInizioTerapia, DataFineTerapia, Posologia)

Esercizio 1

Scrivere una query che, considerata ciascuna parte del corpo, ne restituisca il nome, e i principi attivi contenuti in farmaci indicati solamente per la cura di patologie a carico di tale parte del corpo.

```
SELECT DISTINCT F.PrincipioAttivo,
                P.ParteCorpo
FROM Indicazione I
     INNER JOIN
     Patologia P ON I.Patologia = P.Nome
     INNER JOIN
     Farmaco F ON I.Farmaco = F.NomeCommerciale
GROUP BY I.Farmaco
HAVING COUNT(DISTINCT P.ParteCorpo) = 1;
```

Esercizio 1

Scrivere una query che elimini tutti gli esordi di otite contratta e curata con successo prima di cinque anni fa, relativi ai soli pazienti che hanno contratto nuovamente, negli ultimi cinque anni, la stessa patologia.

```
DELETE E.*
FROM Esordio E
     NATURAL JOIN
     (
      SELECT *
      FROM Esordio E2
      WHERE E2.Patologia = 'Otite'
            AND E2.DataGuarigione IS NOT NULL
            AND YEAR(E2.DataGuarigione) < YEAR(CURRENT_DATE) - 5
            AND EXISTS
                (
                 SELECT *
                 FROM Esordio E3
                 WHERE E3.Paziente = E2.Paziente
                       AND E3.Patologia = E2.Patologia
                       AND YEAR(E3.DataEsordio) > YEAR(CURRENT_DATE) - 5
                )
     ) AS Target;
```

Esercizio 1

Scrivere una query che elimini tutti gli esordi di otite contratta e curata con successo prima di cinque anni fa, relativi ai soli pazienti che hanno contratto nuovamente, negli ultimi cinque anni, la stessa patologia.

```
DELETE FROM Esordio
WHERE (Paziente, Patologia, DataEsordio) IN (
    SELECT * FROM (
        SELECT E.Paziente, E.Patologia, E.DataEsordio
        FROM Esordio E
        WHERE E.Patologia = 'Otite'
            AND E.DataGuarigione IS NOT NULL
            AND YEAR(E.DataGuarigione) < YEAR(CURRENT_DATE) - 5
            AND EXISTS (
                SELECT *
                FROM Esordio E2
                WHERE E2.Patologia = E.Patologia
                    AND E2.Paziente = E.Paziente
                    AND YEAR(E2.DataEsordio) > YEAR(CURRENT_DATE) - 5
            )
        ) AS D
);
```

Esercizio 1

Scrivere una query che restituisca la dose giornaliera media dei farmaci indicati per la cura di sole patologie intestinali.

```
SELECT AVG(I1.DoseGiornaliera)
FROM Indicazione I1
WHERE I1.Farmaco NOT IN(
      SELECT I2.Farmaco
      FROM Indicazione I2 INNER JOIN Patologia P ON I2.Patologia = P.Nome
      WHERE P.ParteCorpo <> 'Intestino'
);
```

Esercizio 3

Scrivere una query che aumenti di un'unità il dosaggio giornaliero indicato di tutti i farmaci per la cura di sole patologie a carico della tiroide che sono sempre stati assunti, tranne nei casi di patologie croniche, a dosaggi superiori rispetto a quello indicato.

```
UPDATE Indicazione I NATURAL JOIN
       (SELECT I2.Farmaco
        FROM Indicazione I2 INNER JOIN Patologia P ON I2.Patologia = P.Nome
        WHERE P.ParteCorpo = 'Tiroide'
              AND I2.Farmaco NOT IN(
                  SELECT I3.Farmaco
                  FROM Indicazione I3 INNER JOIN Patologia P2
                       ON I3.Patologia = P2.Nome
                  WHERE P2.ParteCorpo <> 'Tiroide')
       ) AS D
SET I.DoseGiornaliera = I.DoseGiornaliera + 1
WHERE NOT EXISTS (SELECT *
                  FROM Terapia T NATURAL JOIN Esordio E
                  WHERE T.Farmaco = I.Farmaco
                        AND T.Posologia <= I.DoseGiornaliera
                        AND E.Cronica = 'no');
```

Esercizio 2

Scrivere una query che restituisca nome e cognome dei pazienti che, esclusivamente per le patologie ortopediche, hanno assunto, man mano nella vita, tutti i farmaci a base di nimesulide.

```
CREATE OR REPLACE VIEW PazientiSettoreMedicoNimesulide AS
SELECT E.Paziente, P.SettoreMedico
FROM Esordio E NATURAL JOIN Terapia T
     INNER JOIN Patologia P ON E.Patologia = P.Nome
     INNER JOIN Farmaco F ON T.Farmaco = F.NomeCommerciale
WHERE F.PrincipioAttivo = 'Nimesulide'
GROUP BY E.Paziente, P.SettoreMedico
HAVING COUNT(DISTINCT T.Farmaco) = (SELECT COUNT(*)
                                    FROM Farmaco
                                    WHERE PrincipioAttivo = 'Nimesulide');

SELECT P.Nome, P.Cognome
```

```
HAVING COUNT(DISTINCT T.Farmaco) = (SELECT COUNT(*)
                                   FROM Farmaco
                                   WHERE PrincipioAttivo = 'Nimesulide');

SELECT P.Nome, P.Cognome
FROM PazientiSettoreMedicoNimesulide PSMN INNER JOIN Paziente P
     ON PSMN.Paziente = P.CodFiscale
WHERE PSMN.SettoreMedico = 'Ortopedia'
      AND NOT EXISTS (SELECT *
                      FROM PazientiSettoreMedicoNimesulide PSMN2
                      WHERE PSMN2.Paziente = PSMN.Paziente
                            AND PSMN2.SettoreMedico <> PSMN.SettoreMedico);
```

Esercizio 2 (7 punti)

Scrivere una query che restituisca il numero di pazienti visitati solo da medici specializzati in cardiologia o neurologia, almeno due volte per ciascuna delle due specializzazioni. Si scriva la query senza usare viste.

```
SELECT FLOOR(COUNT(*)/2) AS NumeroPazienti
FROM(
     SELECT Specializzazione, Paziente
     FROM Visita V INNER JOIN Medico M ON V.Medico = M.Matricola
     WHERE V.Paziente NOT IN(
           SELECT V2.Paziente
           FROM Visita V2 INNER JOIN Medico M2
                ON V2.Medico = M2.Matricola
           WHERE M2.Specializzazione <> 'Cardiologia'
                 AND M2.Specializzazione <> 'Neurologia')
     GROUP BY M.Specializzazione, V.Paziente
     HAVING COUNT(*) >= 2
    ) AS D
GROUP BY Paziente
HAVING COUNT(*) = 2;
```

Sempre

lunedì 06 febbraio 2017 10:03

Esercizio 1

Scrivere una query che restituisca le patologie curate sempre con il farmaco meno costoso fra tutti quelli indicati. Se, data una patologia, esiste più di un farmaco meno costoso, questi possono essere stati usati intercambiabilmente.

```sql
CREATE OR REPLACE VIEW CostiMinimi AS
SELECT I.Patologia,
       MIN(F.Costo) AS Costo
FROM Indicazione I
     INNER JOIN
     Farmaco F ON I.Farmaco = F.NomeCommerciale
GROUP BY I.Patologia;

SELECT T.Patologia
FROM Terapia T
     INNER JOIN
     Farmaco F ON T.Farmaco = F.NomeCommerciale
     NATURAL JOIN
     CostiMinimi CM
GROUP BY T.Patologia
HAVING COUNT(*) = (SELECT COUNT(*)
                   FROM Terapia T2
                   WHERE T2.Patologia = T.Patologia);
```

```sql
-- Torna
CREATE OR REPLACE VIEW FarmaciIndicati AS
    SELECT  I.Patologia,
            I.Farmaco,
            F.Costo
    FROM    Indicazione I
            INNER JOIN
            /* Solo per inserire il costo, per costruire
            così la tabella dei farmaci più economici */
            Farmaco F  ON I.Farmaco = F.NomeCommerciale
    ORDER BY I.Patologia, I.Farmaco, F.Costo;

CREATE OR REPLACE VIEW FarmaciEconomici AS
    SELECT *
    FROM FarmaciIndicati FI
    WHERE FI.Costo = (
        SELECT MIN(FI2.Costo)
        FROM FarmaciIndicati FI2
        WHERE FI2.Patologia = FI.Patologia
    );

SELECT DISTINCT T1.Patologia
FROM Terapia T1
WHERE NOT EXISTS (
    SELECT *
    FROM Terapia T2
    WHERE T2.Patologia = T1.Patologia
        AND T2.Farmaco NOT IN (
            SELECT FE.Farmaco
            FROM FarmaciEconomici FE
            WHERE FE.Patologia = T2.Patologia
        )
);
```

Esercizio 2

Scrivere una query che restituisca, per il sesso maschile e per quello femminile, rispettivamente, il numero di pazienti attualmente affetti da ipertensione, trattata con lo stesso farmaco da più di venti anni.

```sql
SELECT P.Sesso, COUNT(DISTINCT P.CodFiscale)
FROM Esordio E INNER JOIN Paziente P ON E.Paziente = P.CodFiscale
WHERE E.DataGuarigione IS NULL
      AND E.Patologia = 'Ipertensione'
      AND EXISTS(
           SELECT *
           FROM Terapia T
           WHERE T.Paziente = E.Paziente
                 AND T.Patologia = E.Patologia
                 AND T.DataEsordio = E.DataEsordio
                 AND T.DataFineTerapia IS NULL
                 AND T.DataInizioTerapia < CURRENT_DATE - INTERVAL 20 YEAR
          )
GROUP BY P.Sesso;
```

Esercizio 3

Scrivere una query che aumenti di un'unità il dosaggio giornaliero indicato di tutti i farmaci per la cura di sole patologie a carico della tiroide che sono sempre stati assunti, tranne nei casi di patologie croniche, a dosaggi superiori rispetto a quello indicato.

```sql
UPDATE Indicazione I NATURAL JOIN
      (SELECT I2.Farmaco
       FROM Indicazione I2 INNER JOIN Patologia P ON I2.Patologia = P.Nome
       WHERE P.ParteCorpo = 'Tiroide'
             AND I2.Farmaco NOT IN(
                 SELECT I3.Farmaco
                 FROM Indicazione I3 INNER JOIN Patologia P2
                      ON I3.Patologia = P2.Nome
                 WHERE P2.ParteCorpo <> 'Tiroide')
      ) AS D
SET I.DoseGiornaliera = I.DoseGiornaliera + 1
WHERE NOT EXISTS (SELECT *
                  FROM Terapia T NATURAL JOIN Esordio E
                  WHERE T.Farmaco = I.Farmaco
                        AND T.Posologia <= I.DoseGiornaliera
                        AND E.Cronica = 'no');
```

Mai

martedì 07 febbraio 2017 12:04

Esercizio 2 [stessa soluzione per l'esercizio 3, anno accademico precedente]

Scrivere una query che restituisca nome, cognome e reddito dei pazienti di sesso femminile che al 15 Giugno 2010 risultavano affetti, oltre alle eventuali altre, da un'unica patologia cronica, con invalidità superiore al 50%, e non l'avevano mai curata con alcun farmaco fino a quel momento.

```
SELECT P.Nome, P.Cognome, P.Reddito
FROM  Esordio E INNER JOIN Paziente P ON E.Paziente = P.CodFiscale
      INNER JOIN Patologia PA ON E.Patologia = PA.Nome
WHERE E.DataEsordio < '2010-06-15'
      AND PA.Invalidita > 50
      AND E.DataGuarigione IS NULL -- ridondante, la malattia è cronica
      AND E.Cronica = 'si'
      AND NOT EXISTS (SELECT *
                      FROM Terapia T
                      WHERE T.Paziente = E.Paziente
                            AND T.Patologia = E.Patologia
                            AND T.DataInizioTerapia < '2010-06-15'
                     )
GROUP BY E.Paziente, P.Nome, P.Cognome, P.Reddito
HAVING COUNT(*) = 1;
```

Tutti

lunedì 06 febbraio 2017 09:50

A.A. precedente (2013-2014)

Scrivere una query che elenchi nome e cognome dei pazienti oggi maggiorenni che, al 5 Settembre 2015 erano stati visitati da tutti gli oculisti della clinica.

```
CREATE OR REPLACE VIEW Oculisti AS
SELECT M.Matricola,
       M.Nome,
       M.Cognome
FROM Medico M
WHERE M.Specializzazione = 'Oculistica';

SELECT P.Nome,
       P.Cognome
FROM Visita V
     INNER JOIN
     Oculisti O ON V.Medico = O.Matricola
     INNER JOIN
     Paziente P ON V.Paziente = P.CodFiscale
WHERE P.DataNascita + INTERVAL 18 YEAR < CURRENT_DATE
GROUP BY V.Paziente
HAVING COUNT(DISTINCT V.Medico) =
    (
     SELECT COUNT(*)
     FROM Oculisti
    );
```

Esercizio 2

Scrivere una query che restituisca per ciascun principio attivo, il nome del principio attivo e il nome commerciale di ogni farmaco utilizzato almeno una volta per tutte le patologie per le quali è indicato. Il risultato è formato da row (PrincipioAttivo, NomeCommerciale), una per ogni farmaco che rispetta la condizione.

```
SELECT F.PrincipioAttivo, T.Farmaco
FROM Terapia T INNER JOIN Farmaco F ON T.Farmaco = F.NomeCommerciale
GROUP BY T.Farmaco, F.PrincipioAttivo
HAVING COUNT(DISTINCT T.Patologia) =
         (SELECT COUNT(*)
          FROM Indicazione I
          WHERE I.Farmaco = T.Farmaco
         )
ORDER BY F.PrincipioAttivo;

/* ORDER BY non è necessario, ma aiuta a interpretare meglio il risultato
richiesto a livello visivo */
```

Esercizio 2

Scrivere una query che restituisca il numero di terapie iniziate da ciascun paziente in ogni mese dell'anno. Nel risultato devono comparire tutti i pazienti e tutti i mesi dell'anno.

```
SELECT PM.CodFiscale, PM.Mese, IF(P.CodFiscale IS NULL, 0, COUNT(*))
FROM Terapia T RIGHT OUTER JOIN Paziente P ON T.Paziente = P.CodFiscale
     RIGHT OUTER JOIN (SELECT P.CodFiscale, D.Mese
                       FROM Paziente P CROSS JOIN
                          (SELECT DISTINCT MONTH(DataInizioTerapia) AS Mese
                           FROM Terapia) AS D
                      ) AS PM
     ON (PM.CodFiscale = P.CodFiscale
         AND PM.Mese = MONTH(T.DataInizioTerapia))
GROUP BY PM.CodFiscale, PM.Mese;
```

Esercizio 1

Senza fare uso di join né viste, scrivere una query che restituisca nome e cognome dei medici che hanno visitato tutti i pazienti di Roma.

```
SELECT M.Nome, M.Cognome
FROM Medico M
WHERE NOT EXISTS(
       SELECT *
       FROM Paziente P
       WHERE P.Citta = 'Roma'
             AND NOT EXISTS(
                 SELECT *
                 FROM Visita V
                 WHERE V.Paziente = P.CodFiscale
                       AND V.Medico = M.Matricola
             )
      );
```

```
SELECT  M.Nome,
        M.Cognome
FROM    Medico M
WHERE   NOT EXISTS(
            SELECT  *
            FROM    Paziente P
            WHERE   P.Citta = 'Roma'
                    AND NOT EXISTS(
                        SELECT  *
                        FROM    Visita V
                        WHERE   V.Paziente = P.CodFiscale
                                AND V.Medico = M.Matricola
                    )
        );
```

Esercizio 2

Scrivere una query che restituisca nome e cognome dei pazienti che, esclusivamente per le patologie ortopediche, hanno assunto, man mano nella vita, tutti i farmaci a base di nimesulide.

```
CREATE OR REPLACE VIEW PazientiSettoreMedicoNimesulide AS
SELECT E.Paziente, P.SettoreMedico
FROM Esordio E NATURAL JOIN Terapia T
     INNER JOIN Patologia P ON E.Patologia = P.Nome
     INNER JOIN Farmaco F ON T.Farmaco = F.NomeCommerciale
WHERE F.PrincipioAttivo = 'Nimesulide'
GROUP BY E.Paziente, P.SettoreMedico
HAVING COUNT(DISTINCT T.Farmaco) = (SELECT COUNT(*)
                                    FROM Farmaco
                                    WHERE PrincipioAttivo = 'Nimesulide');

SELECT P.Nome, P.Cognome
FROM PazientiSettoreMedicoNimesulide PSMN INNER JOIN Paziente P
     ON PSMN.Paziente = P.CodFiscale
WHERE PSMN.SettoreMedico = 'Ortopedia'
      AND NOT EXISTS (SELECT *
                      FROM PazientiSettoreMedicoNimesulide PSMN2
                      WHERE PSMN2.Paziente = PSMN.Paziente
                            AND PSMN2.SettoreMedico <> PSMN.SettoreMedico);
```

Esercizio 1

Scrivere una query che restituisca il nome commerciale di ciascun farmaco utilizzato da almeno un paziente per curare tutte le patologie per le quali è indicato.

```
SELECT DISTINCT T.Farmaco
FROM Terapia T
GROUP BY T.Paziente, T.Farmaco
HAVING COUNT(DISTINCT T.Patologia) = (SELECT COUNT(*)
                                      FROM Indicazione I
                                      WHERE I.Farmaco = T.Farmaco);
```

Tutti o tutti tranne uno

lunedì 06 febbraio 2017 09:50

Esercizio 2

Scrivere una query che elenchi nome e cognome dei pazienti oggi maggiorenni che, al 5 Settembre 2015, erano stati visitati da tutti gli oculisti della clinica, tranne eventualmente uno, e, qualora esista, il cognome di tale oculista.

```
CREATE OR REPLACE VIEW Oculisti AS
SELECT M.Matricola,
       M.Nome,
       M.Cognome
FROM Medico M
WHERE M.Specializzazione = 'Oculistica';

SET@num_oculisti = (
                    SELECT COUNT(*)
                    FROM Oculisti
                   );

CREATE OR REPLACE VIEW PazientiVisitatiTuttiQuasiOculisti AS
SELECT V.Paziente,
       COUNT(DISTINCT V.Medico) AS Quanti
FROM Visita V
     INNER JOIN
     Oculisti O ON V.Medico = O.Matricola
     INNER JOIN Paziente P ON V.Paziente = P.CodFiscale solo per la condizione di pazienti maggiorenni
WHERE P.DataNascita + INTERVAL 18 YEAR < CURRENT_DATE
GROUP BY V.Paziente
HAVING COUNT(DISTINCT V.Medico) >=
    (
     SELECT COUNT(*) - 1 non può usare @num_oculisti (le views vietano l'utilizzo di variabili e parametri)
     FROM Oculisti
    );

CREATE OR REPLACE VIEW MappingPazienteOculista AS
SELECT DISTINCT V.Medico,
       V.Paziente
FROM Visita V
     INNER JOIN
     Oculisti O ON V.Medico = O.Matricola;

CREATE OR REPLACE VIEW Combinazioni AS
SELECT O.Matricola AS Medico,
       PVTQO.Paziente,
       PVTQO.Quanti
FROM Oculisti O
     CROSS JOIN
     PazientiVisitatiTuttiQuasiOculisti PVTQO;
```

```sql
SELECT P.Nome,
       P.Cognome,
       O.Cognome
FROM
    (
     SELECT DISTINCT C.Paziente,
            C.Quanti,
            IF(MPO.Medico IS NULL, C.Medico, NULL) AS Medico
     FROM Combinazioni C
          LEFT OUTER JOIN
          MappingPazienteOculista MPO
          ON C.Paziente = MPO.Paziente
             AND C.Medico = MPO.Medico
    ) AS D
    INER JOIN
    Paziente P ON D.Paziente = P.CodFiscale
    LEFT OUTER JOIN
    Oculisti O ON D.Medico = O.Matricola
WHERE (
       D.Quanti = @num_oculisti
       AND D.Medico IS NULL
      )
      OR
      (
       D.Quanti = @num_oculisti - 1
       AND D.Medico IS NOT NULL
      );
```

Numero di volte maggiore (ALL)

lunedì 06 febbraio 2017 09:54

Database

PAZIENTE(CodFiscale, Cognome, Nome, Sesso, DataNascita, Citta, Reddito)
MEDICO(Matricola, Cognome, Nome, Specializzazione, Parcella, Citta)
FARMACO(NomeCommerciale, PrincipioAttivo, Costo, Pezzi)
PATOLOGIA(Nome, ParteCorpo, SettoreMedico, Invalidita, PercEsenzione)
INDICAZIONE(Farmaco, Patologia, DoseGiornaliera, NumGiorni, AVita)
VISITA(Medico, Paziente, Data, Mutuata)
ESORDIO(Paziente, Patologia, DataEsordio, DataGuarigione, Gravita, Cronica)
TERAPIA(Paziente, Patologia, DataEsordio, Farmaco, DataInizioTerapia, DataFineTerapia, Posologia)

Esercizio 1

Scrivere una query che restituisca, se esiste, la città dalla quale proviene il maggior numero di pazienti che hanno contratto l'acufene un numero di volte maggiore o uguale a quello degli altri pazienti della loro città.

```
CREATE OR REPLACE VIEW EsordiAcufene AS
SELECT P.CodFiscale,
       P.Citta,
       COUNT(*) AS NumeroEsordi
FROM Esordio E
     INNER JOIN
     Paziente P ON E.Paziente = P.CodFiscale
WHERE E.Patologia = 'Acufene'
GROUP BY P.CodFiscale, P.Citta;

CREATE OR REPLACE VIEW PazientiCitta
SELECT EA.Citta,
       COUNT(*) AS NumeroPazienti
FROM EsordiAcufene EA
WHERE EA.NumeroEsordi >= ALL
      (
       SELECT EA2.NumeroEsordi
       FROM EsordiAcufene EA2
       WHERE EA2.Citta = EA.Citta
      )
GROUP BY EA.Citta;

SELECT PC.Citta
FROM PazientiCitta PC
WHERE PC.NumeroPazienti > ALL
      (
       SELECT PC2.NumeroPazienti
       FROM PazientiCitta PC2
       WHERE PC2.Citta <> PC.Citta
      );
```

Esercizio 1

Scrivere una query che restituisca nome e cognome del medico che, al 31/12/2014, aveva visitato un numero di pazienti superiore a quelli visitati da ciascun medico della sua stessa specializzazione.

```
CREATE OR REPLACE VIEW PazientiVisitati AS
SELECT V.Medico, M.Specializzazione,
       COUNT(DISTINCT V.Paziente) AS NumeroPazienti
FROM Visita V INNER JOIN Medico M ON V.Medico = M.Matricola
WHERE V.Data < '2014-12-31'
GROUP BY V.Medico;

SELECT M.Nome, M.Cognome
FROM Medico M INNER JOIN PazientiVisitati PV
     ON M.Matricola = PV.Medico
WHERE PV.NumeroPazienti >= ALL
      (SELECT PV2.NumeroPazienti
       FROM PazientiVisitati PV2
       WHERE PV2.Specializzazione = PV.Specializzazione
      );
```

Esercizio 1

Scrivere una query che restituisca la parte del corpo maggiormente colpita da patologie con invalidità superiore al 70%. In caso di pari merito, restituire tutte le parti del corpo.

```
CREATE OR REPLACE VIEW PatologieAltaInvalidita AS
SELECT Nome AS Patologia, ParteCorpo
FROM Patologia
WHERE Invalidita > 70;

SELECT PAI.ParteCorpo
FROM Esordio E NATURAL JOIN PatologieAltaInvalidita PAI
GROUP BY PAI.ParteCorpo
HAVING COUNT(*) >= ALL (
       SELECT COUNT(*) AS NumeroEsordi
       FROM Esordio E2 NATURAL JOIN PatologieAltaInvalidita PAI2
       GROUP BY PAI2.ParteCorpo);
```

```
CREATE OR REPLACE VIEW CasiParteCorpo AS
    SELECT  P.ParteCorpo,
            COUNT(*) AS NumeroEsordi
    FROM    Esordio E
            INNER JOIN
            Patologia P ON E.Patologia = P.Nome
    WHERE   P.Invalidita > 70
    GROUP BY P.ParteCorpo;

SELECT  CPC.ParteCorpo
FROM    CasiParteCorpo CPC
WHERE   CPC.NumeroEsordi = (
            SELECT  MAX(CPC2.NumeroEsordi)
            FROM    CasiParteCorpo CPC2
        );
```

Incidenza

martedì 07 febbraio 2017 10:49

Esercizio 2

Scrivere una query che restituisca le patologie che nel 2011 hanno colpito, con tasso d'incidenza complessivo superiore al 90%, bambini di età inferiore a 5 anni o anziani di età superiore a 75 anni.

```
SELECT E.Patologia
FROM Paziente P INNER JOIN Esordio E
     ON P.CodFiscale = E.Paziente
WHERE YEAR(E.DataEsordio) = 2011
      AND (P.DataNascita + INTERVAL 5 YEAR > E.DataEsordio
           OR P.DataNascita + INTERVAL 75 YEAR < E.DataEsordio)
GROUP BY E.Patologia
HAVING COUNT(DISTINCT E.Paziente) >
       0.9*(SELECT COUNT(*)
            FROM Paziente P
            WHERE YEAR(P.DataNascita + INTERVAL 5 YEAR) > 2011
                  OR YEAR(P.DataNascita + INTERVAL 75 YEAR) > 2011
            );
```

Mediamente più

lunedì 06 febbraio 2017 11:21

Database

PAZIENTE(CodFiscale, Cognome, Nome, Sesso, DataNascita, Citta, Reddito)
MEDICO(Matricola, Cognome, Nome, Specializzazione, Parcella, Citta)
FARMACO(NomeCommerciale, PrincipioAttivo, Costo, Pezzi)
PATOLOGIA(Nome, ParteCorpo, SettoreMedico, Invalidita, PercEsenzione)
INDICAZIONE(Farmaco, Patologia, DoseGiornaliera, NumGiorni, AVita)
VISITA(Medico, Paziente, Data, Mutuata)
ESORDIO(Paziente, Patologia, DataEsordio, DataGuarigione, Gravita, Cronica)
TERAPIA(Paziente, Patologia, DataEsordio, Farmaco, DataInizioTerapia, DataFineTerapia, Posologia)

Esercizio 1 (10 punti)

Considerate tutte le città di provenienza dei pazienti, scrivere una query che restituisca la patologia mediamente più contratta, fra tutte le città, da pazienti al di sotto dei venti anni d'età. In caso di pari merito, restituire tutti gli ex aequo.

```
/* Patologia, Citta, NumeroCasi */
CREATE OR REPLACE VIEW Casi AS
    SELECT  E.Patologia,
            P.Citta,
            COUNT(*) AS NCasi
    FROM Esordio E INNER JOIN Paziente P
        ON E.Paziente = P.CodFiscale
    WHERE E.DataEsordio < P.DataNascita + INTERVAL 20 YEAR
    GROUP BY E.Patologia, P.Citta;

/* Patologia, Media dei casi tra TUTTE le città */
CREATE OR REPLACE VIEW Medie AS
    SELECT  SUM(C.NCasi)/(
                /*  +++++ IMPORTANTISSIMO +++++ */
                /*  La media va effettuata tra
                    TUTTE le città di provenienza
                    dei pazienti. */
                SELECT COUNT(DISTINCT P.Citta)
                FROM Paziente P
            ) AS Media,
            C.Patologia
    FROM Casi C
    GROUP BY C.Patologia;

SELECT M1.Patologia
FROM Medie M1
WHERE M1.Media = (
    SELECT MAX(M2.Media)
    FROM Medie M2
);
```

Esercizio 2 (anni accademici precedenti senza la parte fra parentesi quadre)

Scrivere una query che, considerati i soli pazienti affetti da ipertensione cronica da almeno dieci anni trattata al massimo con due farmaci diversi, indichi il nome commerciale del farmaco mediamente più utilizzato per curare le altre patologie cardiache croniche. [In caso di pari merito, il risultato deve essere vuoto.]

Interpretazione:

- **Mediamente** per patologia
- Più **utilizzato** per numero di pazienti diversi

Soluzione senza parentesi quadre:

```
/* Pazienti target, che faranno da filtro alla
query finale, tramite NATURAL JOIN con Terapia */
CREATE OR REPLACE VIEW PazientiTarget AS
    SELECT  DISTINCT E.Paziente
    FROM    Esordio E
            NATURAL JOIN
            Terapia T
    WHERE   E.Patologia = 'Ipertensione'
            AND E.Cronica = 'si'
            AND E.DataEsordio + INTERVAL 10 YEAR < CURRENT_DATE
            -- "...considerati i soli pazienti AFFETTI da..."
            -- il prof NON ha messo
            -- AND E.DataGuarigione IS NULL
    GROUP BY E.Paziente, E.DataEsordio
    HAVING COUNT(DISTINCT T.Farmaco) <= 2;

/* Indica, per ogni farmaco, su quanti pazienti
è stato usato per ogni patologia */
CREATE OR REPLACE VIEW TotaliUtilizzi AS
    SELECT  T.Patologia,
            T.Farmaco,
            COUNT(DISTINCT T.Paziente) NPazienti
    FROM    -- patologie croniche
            Esordio E
            NATURAL JOIN
            -- considerati i soli pazienti...
            PazientiTarget PT
            NATURAL JOIN
            Terapia T
            INNER JOIN
            -- patologie cardiologiche
            Patologia P ON T.Patologia = P.Nome
    WHERE   P.ParteCorpo = 'Cuore'
            AND P.Nome <> 'Ipertensione'
            AND E.Cronica = 'si'
    GROUP BY T.Patologia, T.Farmaco;

/* Per ogni farmaco, si fa la media
degli utilizzi sui diversi pazienti per
ogni patologia (cardiaca non ipertensione).
Viene mostrato il farmaco che ha la media
più alta. */
SELECT TU.Farmaco
FROM TotaliUtilizzi TU
GROUP BY TU.Farmaco
HAVING AVG(TU.NPazienti) >= ALL (
    SELECT AVG(TU2.NPazienti)
    FROM TotaliUtilizzi TU2
    GROUP BY TU2.Farmaco
);
```

Semplice media

martedì 07 febbraio 2017 11:05

Esercizio 1

Scrivere una query che, relativamente a ciascun mese del 2013, restituisca il mese (come intero da 1 a 12) e il numero medio di terapie per esordio effettuate dai pazienti per combattere il dolore.

```
SELECT MONTH(D.DataEsordio) AS Mese, AVG(NumeroTerapie) AS TerapieMedie
FROM(
      SELECT E.Paziente, E.DataEsordio, COUNT(*) AS NumeroTerapie
      FROM Esordio E NATURAL JOIN Terapia T
      WHERE YEAR(E.DataEsordio) = 2013
            AND E.Patologia = 'Dolore'
      GROUP BY E.Paziente, E.DataEsordio
    ) AS D
GROUP BY MONTH(D.DataEsordio);
```

Eliminazione

lunedì 06 febbraio 2017 10:55

Esercizio 1

Scrivere una query che elimini tutti gli esordi di otite contratta e curata con successo prima di cinque anni fa, relativi ai soli pazienti che hanno contratto nuovamente, negli ultimi cinque anni, la stessa patologia.

```
DELETE E.*
FROM Esordio E
     NATURAL JOIN
     (
      SELECT *
      FROM Esordio E2
      WHERE E2.Patologia = 'Otite'
            AND E2.DataGuarigione IS NOT NULL
            AND YEAR(E2.DataGuarigione) < YEAR(CURRENT_DATE) - 5
            AND EXISTS
                (
                 SELECT *
                 FROM Esordio E3
                 WHERE E3.Paziente = E2.Paziente
                       AND E3.Patologia = E2.Patologia
                       AND YEAR(E3.DataEsordio) > YEAR(CURRENT_DATE) - 5
                )
     ) AS Target;
```

Esercizio 1

Scrivere una query che elimini tutti gli esordi di otite contratta e curata con successo prima di cinque anni fa, relativi ai soli pazienti che hanno contratto nuovamente, negli ultimi cinque anni, la stessa patologia.

```
DELETE FROM Esordio
WHERE (Paziente, Patologia, DataEsordio) IN (
    SELECT * FROM (
        SELECT E.Paziente, E.Patologia, E.DataEsordio
        FROM Esordio E
        WHERE E.Patologia = 'Otite'
            AND E.DataGuarigione IS NOT NULL
            AND YEAR(E.DataGuarigione) < YEAR(CURRENT_DATE) - 5
            AND EXISTS (
                SELECT *
                FROM Esordio E2
                WHERE E2.Patologia = E.Patologia
                    AND E2.Paziente = E.Paziente
                    AND YEAR(E2.DataEsordio) > YEAR(CURRENT_DATE) - 5
            )
        ) AS D
);
```

Aggiornamenti

martedì 07 febbraio 2017 10:57

Esercizio 3

Scrivere una query che aumenti di un'unità il dosaggio giornaliero indicato di tutti i farmaci per la cura di sole patologie a carico della tiroide che sono sempre stati assunti, tranne nei casi di patologie croniche, a dosaggi superiori rispetto a quello indicato.

```
UPDATE Indicazione I NATURAL JOIN
       (SELECT I2.Farmaco
        FROM Indicazione I2 INNER JOIN Patologia P ON I2.Patologia = P.Nome
        WHERE P.ParteCorpo = 'Tiroide'
              AND I2.Farmaco NOT IN(
                  SELECT I3.Farmaco
                  FROM Indicazione I3 INNER JOIN Patologia P2
                       ON I3.Patologia = P2.Nome
                  WHERE P2.ParteCorpo <> 'Tiroide')
       ) AS D
SET I.DoseGiornaliera = I.DoseGiornaliera + 1
WHERE NOT EXISTS (SELECT *
                  FROM Terapia T NATURAL JOIN Esordio E
                  WHERE T.Farmaco = I.Farmaco
                        AND T.Posologia <= I.DoseGiornaliera
                        AND E.Cronica = 'no');
```

Join-equivalente

lunedì 06 febbraio 2017 09:56

Esercizio 2

Data la seguente query, descriverne il risultato e scriverne la versione join-equivalente senza usare view.

```
SELECT T1.Farmaco
FROM Terapia T1
WHERE NOT EXISTS(
        SELECT *
        FROM Terapia T2
        WHERE T2.Farmaco = T1.Farmaco
              AND EXISTS(
                   SELECT *
                   FROM Terapia T3
                   WHERE T3.DataInizioTerapia < T2.DataInizioTerapia
                         AND T3.Farmaco = T2.Farmaco
                  )
        );
```

```
/*
Nome commerciale dei farmaci assunti in un'unica terapia.
*/

SELECT T.Farmaco
FROM Terapia T
GROUP BY T.Farmaco
HAVING COUNT(*) = 1;
```

Report / snapshot / tab. ridondante

lunedì 06 febbraio 2017 10:00

A.A. precedente

Con cadenza imprevedibile, la direzione della clinica è interessata a conoscere, per ciascuna specializzazione, il numero di nuovi pazienti visitati, il medico che effettua il numero minore di prime visite, e il numero di città diverse dalle quali provengono i pazienti visitati per la prima volta. Si desidera creare una tabella REPORT contenente le precedenti informazioni. Dopo aver creato la tabella, scrivere il codice per popolarla.

```
CREATE TABLE Report(
    Specializzazione CHAR(100) NOT NULL,
    PazientiNuovi INTEGER DEFAULT 0,
    MedicoInf CHAR(255) DEFAULT '',
    CittaDiverse INTEGER DEFAULT 0
);

CREATE TABLE LastReport(
    DataUltimoReport DATE DEFAULT CURRENT_DATE - INTERVAL 1 MONTH);  /* soggettivo */
);

SET @last_report_date =
   (
    SELECT DataUltimoReport
    FROM LastReport
   );

CREATE OR REPLACE VIEW  VisiteNuoviPazienti AS
SELECT V.Medico,
       M.Specializzazione,
       V.Paziente
FROM Visita V
     INNER JOIN
     Medico M ON V.Medico = M.Matricola
WHERE V.Data >= @last_report_date
      AND NOT EXISTS
        (
         SELECT *
         FROM Visita V2
         WHERE V2.Medico = V.Medico
               AND V2.Paziente = V.Paziente
               AND V2.Data < @last_report_date
        );

CREATE OR REPLACE VIEW NuoviPazientiNuoveCitta AS
SELECT VNP.Medico,
       VNP.Specializzazione,
       COUNT(DISTINCT NPPS.Paziente) AS NuoviPazienti,
       COUNT(DISTINCT P.Citta) AS NuoveCitta
```

```
FROM VisiteNuoviPazienti VNP
     INNER JOIN
     Paziente P ON VNP.Paziente = P.CodFiscale
GROUP BY VNP.Medico, VNP.Specializzazione;


CREATE OR REPLACE VIEW MinimoNuoviPazientiSpecializzazione AS
SELECT NPPC.Specializzazione,
       MIN(NuoviPazienti) AS MinimoNuoviPazienti
FROM NuoviPazientiNuoveCitta NPNC
GROUP BY NPPC.Specializzazione;


/* La dipendenza funzionale Specializzazione->Medico è garantita dalla clausola HAVING*/
CREATE OR REPLACE VIEW MediciPeggiori AS
SELECT NPPC.Specializzazione,
       NPPC.Medico,
       MNPS.MinimoNuoviPazienti
FROM NuoviPazientiNuoveCitta NPPC
     NATURAL JOIN
     MinimoNuoviPazientiSpecializzazione MNPS
GROUP BY NPPC.Specializzazione
HAVING COUNT(*) = 1;

/*
   Il contesto di transazione (wrapper START TRANSACTION-COMMIT) benché necessario,
   non era richiesto nell'esercizio
*/

START TRANSACTION;

    TRUNCATE TABLE Report;

    INSERT INTO Report
    SELECT NPNC.Specializzazione,
           NPNC.NuoviPazienti,
           MP.Medico,
           NPNC.NuoveCitta
    FROM NuoviPazientiNuoveCitta NPNC
         NATURAL JOIN
         MediciPeggiori MP;

    UPDATE LastReport
    SET DataUltimoReport = CURRENT_DATE;

COMMIT;
```

A.A. precedente (2013-2014)

Scrivere una stored procedure che implementi l'on demand refresh di una tabella EsordiPrecedenti contenente, per ogni record in ESORDIO, il numero di esordi precedenti di patologie dello stesso paziente relative allo stesso settore medico, curati con successo, e il numero medio di terapie necessario per la guarigione.

```
CREATE TABLE EsordiPrecedenti (
      Paziente CHAR(100) NOT NULL,
      Patologia CHAR(100)  NOT NULL,
      DataEsordio DATE NOT NULL,
```

"per ogni record in Esordio" -> chiave di Esordio

A.A. precedente (2013-2014)

Scrivere una stored procedure che implementi l'on demand refresh di una tabella EsordiPrecedenti contenente, per ogni record in ESORDIO, il numero di esordi precedenti di patologie dello stesso paziente relative allo stesso settore medico, curati con successo, e il numero medio di terapie necessario per la guarigione.

```
CREATE TABLE EsordiPrecedenti (
      Paziente CHAR(100) NOT NULL,
      Patologia CHAR(100)  NOT NULL,
      DataEsordio DATE NOT NULL,
      EsordiPrecedenti INTEGER,
      TerapieMedie DOUBLE,
      PRIMARY KEY (Paziente, Patologia, DataEsordio)
);

DROP PROCEDURE IF EXISTS refreshEsordiPrecedenti;
DELIMITER $$
CREATE PROCEDURE refreshEsordiPrecedenti()
BEGIN
      DECLARE _paziente CHAR(100) DEFAULT '';
      DECLARE _patologia CHAR(100) DEFAULT '';
      DECLARE _dataEsordio DATE DEFAULT NULL;
      DECLARE _settoreMedico CHAR(100) DEFAULT '';
      DECLARE _esordiPrecedenti INTEGER DEFAULT 0;
      DECLARE finito INTEGER DEFAULT 0;
      DECLARE _terapieMedie INTEGER DEFAULT 0;
```

"per ogni record in Esordio" -> chiave di Esordio

```
DECLARE esordi CURSOR FOR
SELECT E.Paziente,
       E.Patologia,
       E.DataEsordio
FROM Esordio E;

DECLARE CONTINUE HANDLER
FOR NOT FOUND
SET finito = 1;

OPEN esordi;

scan: LOOP

      FETCH esordi INTO _paziente, _patologia, _dataEsordio;

      IF finito = 1 THEN
            LEAVE scan;
      END IF;

      SET _settoreMedico :=
            (SELECT P.SettoreMedico
             FROM Patologia P
             WHERE P.Nome = _patologia
            );

      SET _esordiPrecedenti :=
        (SELECT COUNT(*)
         FROM Esordio E
              INNER JOIN
              Patologia P ON E.Patologia = P.Nome
         WHERE E.Paziente = _paziente
               AND P.SettoreMedico = _settoreMedico
                     AND E.DataEsordio = < _dataEsordio
                     AND E.DataGuarigione IS NOT NULL
        );

     SET _terapieMedie :=
       (SELECT AVG(NTerapie)
        FROM(
             SELECT COUNT(*) AS NTerapie
             FROM Terapia T
                  INNER JOIN
                  Patologia P ON T.Patologia = P.Nome
             WHERE T.Paziente = _paziente
                   AND P.SettoreMedico = _settoreMedico
                   AND T.DataEsordio = < _dataEsordio
             GROUP BY T.Paziente, T.Patologia, T.DataEsordio
            ) AS D
       );

     REPLACE INTO EsordiPrecedenti
     VALUES
     (_paziente, _patologia, _dataEsordio, _esordiPrecedenti,
       _terapieMedie
     );
```

```
    END LOOP scan;

END $$

DELIMITER ;
```

Query complesse

lunedì 06 febbraio 2017 10:03

Database

PAZIENTE(CodFiscale, Cognome, Nome, Sesso, DataNascita, Citta, Reddito)
MEDICO(Matricola, Cognome, Nome, Specializzazione, Parcella, Citta)
FARMACO(NomeCommerciale, PrincipioAttivo, Costo, Pezzi)
PATOLOGIA(Nome, ParteCorpo, SettoreMedico, Invalidita, PercEsenzione)
INDICAZIONE(Farmaco, Patologia, DoseGiornaliera, NumGiorni, AVita)
VISITA(Medico, Paziente, Data, Mutuata)
ESORDIO(Paziente, Patologia, DataEsordio, DataGuarigione, Gravita, Cronica)
TERAPIA(Paziente, Patologia, DataEsordio, Farmaco, DataInizioTerapia, DataFineTerapia, Posologia)

Esercizio 3

Considerati i soli pazienti di Pisa e Roma attualmente affetti da al più tre patologie gastroenterologiche croniche, ognuno di essi visitato, negli ultimi dieci anni, almeno tre volte da un gastroenterologo di città diversa dalla sua, scrivere una query che restituisca il numero di tali pazienti che, dopo essersi fatti visitare, negli anni, da almeno un altro gastroenterologo, hanno effettuato l'ultima visita nuovamente dal gastroenterologo iniziale, trascorso un tempo inferiore a sei mesi dalla prima visita.

```
CREATE OR REPLACE VIEW VisiteTarget AS
SELECT V.Paziente, V.Medico, V.Data
FROM Paziente P
     INNER JOIN
     Visita V ON P.CodFiscale = V.Paziente
     INNER JOIN
     Medico M ON V.Medico = M.Matricola
WHERE YEAR(V.Data) BETWEEN YEAR(CURRENT_DATE) - 9 AND YEAR(CURRENT_DATE)
      AND(
          P.Citta = 'Pisa'
          OR P.Citta = 'Roma'
         )
      AND P.Citta <> M.Citta
      AND M.Specializzazione = 'Gastroenterologia';

CREATE OR REPLACE VIEW PazientiTarget AS
SELECT E.Paziente
FROM Esordio E
     INNER JOIN
     Patologia PA ON E.Patologia = PA.Nome
WHERE PA.SettoreMedico = 'Gastroenterologia'
      AND E.DataGuarigione IS NULL
      AND E.Cronica = 'si'
      AND E.Paziente IN(
           SELECT VT.Paziente
           FROM VisiteTarget VT
           GROUP BY VT.Paziente
           HAVING COUNT(*) >= 3
                  AND COUNT(DISTINCT VT.Data) = COUNT(*) /*nota a pag. 7*/
         )
GROUP BY E.Paziente
HAVING COUNT(*) <= 3;
```

```sql
CREATE OR REPLACE VIEW PrimaUltimaVisita AS
SELECT VT.Paziente,
       MIN(VT.Data) AS DataPrimaVisita,
       MAX(VT.Data) AS DataUltimaVisita
FROM VisiteTarget VT
GROUP BY VT.Paziente
HAVING MAX(VT.Data) < MIN(VT.Data) + INTERVAL 6 MONTH;

SELECT COUNT(*)
FROM(
       SELECT VT.Paziente
       FROM VisiteTarget VT
            NATURAL JOIN
            PrimaUltimaVisita PUV
            NATURAL JOIN
             (
              SELECT PT.Paziente
              FROM PazientiTarget PT
                   NATURAL JOIN
                   VisiteTarget VT2
              GROUP BY PT.Paziente
              HAVING COUNT(DISTINCT VT2.Medico) >= 2  /* almeno 2 gastro */
             ) AS D
       GROUP BY VT.Paziente
       HAVING COUNT(DISTINCT VT.Medico) = 1
    ) AS DD;


/*
NOTA:

La condizione COUNT(DISTINCT VT.Data) = COUNT(*) permette di non considera-
re il caso, improbabile ma possibile, che un paziente target, nello stesso
giorno, sia visitato da più di un gastroenterologo: data la chiave primaria
di Visita, ciò è possibile. Questa eventualità deve essere scartata perché
altrimenti sarebbe insensato chiedersi quali sono i pazienti che, dopo tut-
ta la trafila, sono tornati dal primo gastroenterologo, poiché tale primo
gastroenterologo potrebbe non essere unico (come già detto, nello stesso
giorno, il paziente potrebbe essere stato visitato da più di un gastroente-
rologo). Inoltre, anche l'ultimo gastroenterologo potrebbe essere non de-
terminabile per la stessa motivazione.

Questa condizione, un po' insidiosa, non era necessaria per ottenere il
punteggio completo dell'esercizio. In questo contesto, si può supporre che,
per una business rule della clinica, dato un giorno, un paziente possa ef-
fettuare al massimo una visita per specializzazione.

*/
```

A.A. precedenti al 2013-2014

Un'analisi interna alla clinica ha lo scopo di esaminare il fenomeno del rischio di cronicizzazione delle patologie. Nell'ambito dell'analisi, un farmaco è considerato tanto potente quante patologie ad alta invalidità (superiore al 70%) è in grado di curare. Data una patologia, il rischio dicronicità di un paziente per tale patologia è espressa, utilizzando l'insieme $\mathcal{T}$ delle terapie del paziente per curare la patologia, come la media normalizzata in [0,1] dei prodotti $r_t = posologia_t \cdot potenza\_farmaco_t \cdot durata_t$, dove $t \in \mathcal{T}$. Scrivere una query che, considerato ciascun paziente e ciascuna patologia da egli/ella contratta almeno una volta, restituisca il codice fiscale del paziente, il nome della patologia e il relativo rischio di cronicità.

```
CREATE OR REPLACE VIEW PotenzaFarmaco AS
SELECT I.Farmaco, COUNT(*) AS Potenza
FROM Patologia P
     INNER JOIN
     Indicazione I ON P.Nome = I.Patologia
WHERE P.Invalidita > 70
GROUP BY I.Farmaco;

CREATE OR REPLACE VIEW Prodotti AS
SELECT E.Paziente,
       E.Patologia,
       T.Posologia*PF.Potenza
       *DATEDIFF(
                 IFNULL(T.DataFineTerapia, CURRENT_DATE),
                 T.DataInizioTerapia
                ) AS TassoRischio
FROM Esordio E
     NATURAL JOIN
     Terapia T
     NATURAL JOIN
     PotenzaFarmaco PF
WHERE E.Cronica = 'no'
      AND E.DataGuarigione IS NOT NULL;

CREATE OR REPLACE VIEW MaxMin AS
SELECT PR.Paziente,
       PR.Patologia,
       MAX(PR.TassoRischio) AS Massimo,
       MIN(PR.TassoRischio) AS Minimo
FROM Prodotti PR
GROUP BY PR.Paziente, PR.Patologia;

SELECT PR.Paziente,
       PR.Patologia,
       SUM(
            (PR.TassoRischio - MM.Minimo)
            /
            (MM.Massimo - MM.Minimo)
           )
          /COUNT(*) AS RischioCronicita
FROM Prodotti PR
     NATURAL JOIN
     MaxMin MM
GROUP BY PR.Paziente, PR.Patologia
HAVING COUNT(*) > 1;
```

"SQLizzare"

giovedì 02 febbraio 2017 14:22

	Italiano	Italiano in forma SQL	SQL
Tutti	Medici che hanno visitato tutti i pazienti di Roma.	Per ogni medico, non esiste paziente romano che non abbia visitato.	• Doppio NOT EXISTS • Si filtra sulla condizione e si impone il numero di record prodotti uguale al numero di oggetti presenti nella condizione