Quaderno 3: Linguaggio SQL

1. Sono date le seguenti relazioni (le chiavi primarie sono sottolineate, gli attributi opzionali sono indicati con *):

AUTORE (CodAutore, Nome, Cognome, Dipartimento, Università)
ARTICOLO (CodArticolo, Titolo, Argomento)
AUTORI_ARTICOLO (CodArticolo, CodAutore)
EDIZIONI_CONFERENZA (Conferenza, Edizione, NomeEdizione, DataInizio, DataFine, Editore)
AUTORE_PRESENTA_ARTICOLO (CodAutore, Data, OraInizio, OraFine, Sala, CodArticolo, Conferenza, Edizione)

Esprimere la seguente interrogazione in linguaggio SQL

- Per ciascun autore che ha presentato almeno 4 articoli di argomento 'Data Mining', ma che non ha mai presentato articoli di argomento 'Deep Learning' o 'CyberSecurity', visualizzare il nome, il cognome, l'università di afferenza dell'autore e il numero totale di articoli presentati dall'autore in ciascuna edizione di ogni conferenza.

```
SELECT Nome, Cognome, Università, Conferenza, Edizione, COUNT(*)
FROM AUTORE AU, AUTORE_PRESENTA_ARTICOLO APA
WHERE AU.CodAutore = APA.CodAutore
AND AU.CodAutore IN (SELECT AA1.CodAutore
                     FROM AUTORE_PRESENTA_ARTICOLO AA1, ARTICOLO AR1
                     WHERE AR1.CodArticolo = AA1.CodArticolo
                     AND AR1.Argomento = 'Data Mining'
                     GROUP BY AA1.CodAutore
                     HAVING COUNT(*) >=4)
AND AU.CodAutore NOT IN (SELECT AU2.CodAutore
                         FROM AUTORE_PRESENTA_ARTICOLO AA2, ARTICOLO AR2
                         WHERE AR2.CodArticolo = AA2.CodArticolo
                         AND (AR1.Argomento = 'Deep Learning OR
                                              AR1.Argomento='CyberSecurity'))
GROUP BY AU.CodAutore, Nome, Cognome, Università, Conferenza, Edizione
```

2. Sono date le seguenti relazioni (le chiavi primarie sono sottolineate, gli attributi opzionali sono indicati con *):

STUDENTE (MatricolaS, Nome, Cognome, CorsodiLaurea)
HOMEWORK_DA_CONSEGNARE (CodHW, Titolo, Argomento, DataScadenzaPrevista)
DOCENTE (CodDocente, Nome, Cognome, Dipartimento)
VALUTAZIONE_HOMEWORK_CONSEGNATI (MatricolaS, CodHW, CodDocente, DataConsegna, DataValutazione, Valutazione)

Esprimere la seguente interrogazione in linguaggio SQL

- Per ciascuno studente, visualizzare il nome e il cognome dello studente, e il titolo e l'argomento di ciascun homework per cui lo studente ha ottenuto una valutazione superiore alla valutazione media conseguita su quell'homework da tutti gli studenti

Sol. 1 – Uso di Condizione di Correlazione

```
SELECT Nome, Cognome, Titolo, Argomento
FROM STUDENTE S, HOMEWORK_DA_CONSEGNARE HDC, VALUTAZIONE_HOMEWORK_CONSEGNATI VHC
WHERE S.MatricolaS = VHC.MatricolaS AND HDC.CodHW = VHC.CodHW
AND Valutazione > (SELECT AVG(Valutazione)
                   FROM VALUTAZIONE_HOWMEWORK_CONSEGNATI VHC2
                   WHERE VHC2.CodHW = VHC.CodHW)  ---condizione di correlazione
```

Sol 2 – alternativa – Uso di Tabella derivata

```
SELECT Nome, Cognome, Titolo, Argomento
FROM STUDENTE S, VALUTAZIONE_HOMEWORK_CONSEGNATI VHC,
     HOMEWORK_DA_CONSEGNARE HDC,
                (SELECT CodHW, AVG(Valutazione) AS Avg_Val
                 FROM VALUTAZIONE_HOMEWORK_CONSEGNATI VHC2
                 GROUP BY CodHW) AS HW_AVG
WHERE S.MatricolaS = VHC.MatricolaS  AND HDC.CodHW = VHC.CodHW
AND VHC.CodHW = HW_AVG.CodHW AND Valutazione > Avg_Val
```

Sol 3 – alternativa – Uso di CTE

```
WITH HW_AVG AS
    (SELECT CodHW, AVG(Valutazione) AS Avg_Val
     FROM VALUTAZIONE_HOMEWORK_CONSEGNATI VHC2
     GROUP BY CodHW)

SELECT Nome, Cognome, Titolo, Argomento
```

```
FROM STUDENTE S, VALUTAZIONE_HOWMEWORK_CONSEGNATI VHC, HW_AVG, HOMEWORK_DA_CONSEGNARE HDC
WHERE S.MatricolaS = VHC.MatricolaS AND VHC.CodHW = HW_AVG.CodHW
AND HDC.CodHW = VHC.CodHW AND Valutazione > Avg_Val
```

3. Sono date le seguenti relazioni (le chiavi primarie sono sottolineate, gli attributi opzionali sono indicati con *):

STUDENTE (MatricolaS, Nome, Cognome, CorsodiLaurea)
HOMEWORK_DA_CONSEGNARE (CodHW, Titolo, Argomento, DataScadenzaPrevista)
DOCENTE (CodDocente, Nome, Cognome, Dipartimento)
VALUTAZIONE_HOMEWORK_CONSEGNATI (MatricolaS, CodHW, CodDocente, DataConsegna, DataValutazione, Valutazione)

Esprimere la seguente interrogazione in SQL

- Per ogni studente che ha consegnato *tutti* gli homework di argomento 'basi di dati', e sempre *prima* della data di consegna prevista (DataConsegna < DataScadenzaPrevista), visualizzare il cognome dello studente e, relativamente agli homework di argomento 'basi di dati' consegnati, la valutazione media ricevuta, il numero complessivo di docenti diversi che hanno effettuato le valutazioni, ed il numero medio di giorni in cui lo studente ha consegnato gli homework in anticipo rispetto alla data di consegna prevista (DataScadenzaPrevista – DataConsegna)

```
SELECT S.Cognome, AVG(VHC.Valutazione),  AVG (HDC.DataScadenzaPrevista-VHC.DataConsegna),
                                                    COUNT(DISTINCT VHC.CodDocente)
FROM STUDENTE S, VALUTAZIONE_HOMEWORK_CONSEGNATI VHC, HOMEWORK_DA_CONSEGNARE HDCW
WHERE HDC.Argomento = 'basi di dati '
AND VHC.CodHW = HDC.CodHW
AND VHC.DataConsegna < HDC.DataScadenzaPrevista
AND S.MatricolaS = VHC.MatricolaS
GROUP BY S.MatricolaS, S.Cognome
HAVING COUNT(*) = (SELECT COUNT(*)
                   FROM HOMEWORK_DA_CONSEGNARE HDC2
                   WHERE Argomento = 'basi di dati')
```

4. Sono date le seguenti relazioni (le chiavi primarie sono sottolineate, gli attributi opzionali sono indicati con *):

CLIENTE (CodFiscale, NomeC, Cognome, DataNascita, Città)
RISTORANTE (CodR, NomeR, Indirizzo, Città, Cucina)
ORDINI (CodO, CodFiscale, Data, Ora, Prezzo, CodR)
DIPENDENTE (MatrD, NomeD, Cognome, Qualifica)
LAVORA_IN (MatrD, Data, CodR)

Esprimere la seguente interrogazione in SQL

- Per ogni ristorante che ha ricevuto complessivamente il maggior numero di ordini tra i ristoranti presenti nella sua stessa città, visualizzare il nome del ristorante e l'incasso totale ottenuto dal ristorante in ciascuna data.

Sol 1-con Tabelle derivate

```
SELECT NomeR, Data, SUM(Prezzo)
FROM RISTORANTE R, ORDINE O
WHERE R.CodR = O.CodR
AND R.CodR IN (SELECT O1.CodR
               FROM ORDINE O1, RISTORANTE R1
               WHERE O1.CodR=R1.CodR
               GROUP BY O1.CodR, O1.Città
               HAVING COUNT(*) = (SELECT MAX(TotRisto)
                                  FROM (SELECT O2.CodR, Città, COUNT(*) AS TotRisto
                                        FROM ORDINI O2, RISTORANTE R2
                                        WHERE O2.CodR=R2.CodR
                                        GROUP BY O2.CodR, Città) AS OrdiniCittà
                                  WHERE OrdiniCittà.Città = O1.Città)
GROUP BY R.CodR, NomeR, Data
```

Sol 2 – alternativa con CTE

```
WITH
OrdiniRisto AS
              (SELECT O.CodR, Città, COUNT(*) AS TotRisto
               FROM ORDINI O, RISTORANTE R
               WHERE O.CodR=R.CodR
               GROUP BY O.CodR, Città)
OrdiniCittà AS
              (SELECT Città, MAX(TotRisto) As TotCittà
               FROM OrdiniRisto
               GROUP BY Città)
```

```
SELECT NomeR, Data, SUM(Prezzo)
FROM RISTORANTE R, ORDINE O
WHERE R.CodR = O.CodR AND R.CodR IN
                                (SELECT CodR
                                FROM OrdiniRisto OR, OrdiniCittà OC
                                WHERE OR.Città=OC=Città AND TotRisto > TotCittà
)
GROUP BY R.CodR, NomeR, Data
```

5. Sia dato il seguente schema relazionale (le chiavi primarie sono sottolineate, gli attributi opzionali sono indicati con '*'):

HOTEL (CodH, Nome, Categoria, Indirizzo, Città)
RECENSIONE_HOTEL (CodR, DataRecensione, CodH, Punteggio, Commento)
RIASSUNTO_RECENSIONI (CodH, NumeroRecensioni, PunteggioComplessivo)
NOTIFICA_RECENSIONI_COMPLESSIVE (CodH, DataRecensione,
PunteggioComplessivoHotel, PunteggioMedioCategoria)

Si scriva il trigger per gestire le recensioni di hotel raccolte attraverso un portale web.

La tabella HOTEL contiene l'elenco degli hotel per cui è possibile inviare una recensione. La tabella RIASSUNTO_RECENSIONI contiene, per ogni hotel, il *numero complessivo* di recensioni ricevute e il *punteggio complessivo* assegnato a ciascun hotel mediante le recensioni. Si consideri che un hotel è presente nella tabella RIASSUNTO_RECENSIONI solo se è stata inserita almeno una recensione per quell'hotel.

Viene inserita attraverso il portale una nuova recensione per un hotel (inserimento di un record nella tabella RECENSIONE_HOTEL). Il trigger deve svolgere le seguenti operazioni.

Si deve aggiornare la tabella RIASSUNTO_RECENSIONI tenendo conto della recensione appena inoltrata. Si consideri anche il caso che questa sia la prima recensione inserita per l'hotel.

Si deve quindi inserire un nuovo record nella tabella NOTIFICA_ RECENSIONI_COMPLESSIVE con le informazioni sul *punteggio complessivo assegnato all'hotel* (attributo PunteggioComplessivoHotel). Deve inoltre essere notificato il *punteggio medio assegnato per categoria* (attributo PunteggioMedioCategoria) calcolato come punteggio complessivo medio per gli hotel della stessa categoria dell'hotel che ha ricevuto la recensione.

Indicazioni per lo svolgimento dell'esercizio:

Si chiede di scrivere il trigger per gestire le recensioni di hotel secondo le modalità sopra riportate.

Se necessario, usare la funzione raise_application_error (…) per segnalare un errore. Non è richiesto di specificare i parametri passati alla funzione.

```
CREATE TRIGGER GestioneRecensioni
AFTER INSERT INTO RECENSIONE_HOTEL
FOR EACH ROW
DECLARE
X number; TotP number; MediaCategoria number;
Categoriahotel char (10);
BEGIN
---- Verifico se è la prima recensione per l'Hotel
SELECT COUNT(*) into X
FROM RIASSUNTO_RECENSIONI
WHERE CodH = :NEW.CodH;

IF X =0 THEN
        INSERT INTO RIASSUNTO_RECENSIONI (CodH, NumeroRecensioni, PunteggioComplessivo)
        VALUES (:NEW.CodH, 1, :NEW.Punteggio);

        TotP := :New.Punteggio;
ELSE
        UPDATE RIASSUNTO_RECENSIONI
        SET NumeroRecensioni = NumeroRecensioni +1,
        PunteggioComplessivo = PunteggioComplessivo + :NEW.Punteggio
        WHERE CodH = :NEW.CodH;

       SELECT PunteggioComplessivo INTO TotP
       FROM RIASSUNTO_RECENSIONI
       WHERE CodH = :NEW.CodH;

END IF;

SELECT Categoria INTO CategoriaHotel
FROM HOTEL
WHERE CodH = :NEW.CodH:

SELECT AVG (PunteggioMedioComplessivo) INTO  MediaCategoria
FROM RIASSUNTO_RECENSIONI RR, HOTEL H
WHERE Categoria = CategoriaHotel AND RR.CodH = H.CodH;
```

```
[---soluzione alternativa
SELECT AVG (PunteggioMedioComlessivo)
FROM RIASSUNTO_RECENSIONI RR, HOTEL H
WHERE Categoria IN (SELECT Categoria
                    FROM HOTEL
                    WHERE CodH = :NEW.CodH)
AND RR.CodH = H.CodH; ]

SELECT COUNT(*) INTO X
FROM NOTIFICA_ESITO_RECENSIONI
WHERE CodH = :NEW.CodH AND DataRecensione =:NEW.DataRecensione;

IF X=0 THEN
        INSERT INTO NOTIFICA_ESITO_RECENSIONI (CodH, DataRecensione, PunteggioComplessivoHotel,
                                                PunteggioMedioCategoria)
        VALUES (:NEW.CodH, :NEW.DataRecensione, TotP, MediaCategoria);
ELSE
        UPDATE NOTIFICA_ESITO_RECENSIONI
        SET PunteggioComplessivoHotel = TotP, PunteggioMedioCategoria = MediaCategoria
        WHERE CodH = :NEW.CodH AND DataRecensione =:NEW.DataRecensione
END IF;

END;
```