Unità D1.4. – Interrogazioni complesse e viste	**MODULO T2**

(A) CONOSCENZA TERMINOLOGICA

Dare una breve descrizione dei termini introdotti:

- Equivalenza di espressioni
- Relazioni derivate
- Viste o relazioni virtuali

(B) CONOSCENZA E COMPETENZA

Rispondere alle seguenti domande producendo anche qualche esempio

B1) Conoscenza

1. Cosa vuol dire che due espressioni in algebra relazionale sono equivalenti?
2. Che differenza c'è tra viste materializzate e virtuali?

B2) Competenza

1. In un'interrogazione composta, comprendente proiezione e selezione, è indifferente che venga eseguita prima l'una o l'altra?
2. Perché una vista semplifica la scrittura di altre interrogazioni?

(C) ESERCIZI DI COMRENSIONE

1. L'equivalenza tra espressioni relazionali è importante perché il DBMS cerca di eseguire espressioni …….. a quelle date, ma meno "pesanti" in termini di …… di elaborazione.
2. Se A è un attributo di R1, si consideri l'interrogazione

   $$\mathbf{SEL}_{A=10}\ (R1\ \mathbf{JOIN}\ R2) = R1\ \mathbf{JOIN}\ \mathbf{SEL}_{A=10}\ (R2)$$

   le due espressioni sono …… ma quella sulla …….riduce in modo significativo la dimensione del risultato intermedio e quindi il costo dell'operazione.
3. Operando sulle ……. che costituiscono il DB, si ottengono, grazie alle interrogazioni, altre relazioni, dette relazioni …….., il cui contenuto è funzione del contenuto delle prime, che costituiscono lo schema …….Le relazioni …….. dette anche …….., possono essere ……., e in tal caso sono memorizzate nella base di dati, o possono essere …… .
4. Considerata la relazione R(A, B, C, D, E), indicare quali delle seguenti relazioni hanno lo stesso numero di tuple di R, ponendo una crocetta, rispettivamente, nella colonna **Sì** o **No**.
5. Indicare, mettendo una crocetta nella colonna apposita, se ciascuna delle seguenti affermazioni risulti vera o falsa.

	Sì	**No**
PROJ $_{A, B, C, D}$ (R)		
PROJ $_{A, C}$ (R)		
PROJ $_{B, C}$ (R)		
PROJ $_{C, D}$ (R)		

	Vero	**Falso**
Ogni relazione ha almeno una chiave		
Ogni relazione ha esattamente una chiave		
Ogni attributo appartiene al massimo ad una chiave		
Possono esistere attributi che non appartengono a nessuna chiave		
Una chiave può essere sottoinsieme di un'altra		
Può esistere una chiave che coinvolge tutti gli attributi		
Può succedere che esistano pi`u chiavi e che una di esse coinvolga tutti gli attributi		

(D) ESERCIZI DI APPLICAZIONE

1. **Esercizio risolto**. Si consideri una base di dati che contiene informazioni sugli impiegati, i progetti e le sedi di una azienda, con le partecipazioni degli impiegati ai progetti e le sedi di svolgimento dei progetti stessi; essa contiene le seguenti relazioni:
   b. *Impiegati(Matricola,Cognome,Nome,Progetto)*, con vincolo di integrità referenziale fra *Progetto* e la relazione *Progetti*
   c. *Progetti(Codice,Titolo)*
   d. *Sedi(Nome,Città, Indirizzo)*
   e. *Svolgimento(Progetto,Sede)*, con vincoli di integrità referenziale fra *Progetto* e la relazione *Progetti* fra *Sede* e la relazione *Sedi*

   a. Formulare in algebra relazionale un'interrogazione per visualizzare Codice, Titolo e Sede dei progetti svolti nelle sedi di Roma.
   PROJ Codice, Titolo, Nome (**SEL** $_{\text{Codice = Progetto, and Sede = Nome and Citta = 'Roma'}}$ (Svolgimento **JOIN** Sedi) **JOIN** Progetti
   b. Formulare in SQL l'interrogazione di cui sopra:
   SELECT Codice, Titolo, nome
   FROM Progetti, Svolgimento, Sedi
   WHERE Codice = Progetto
   AND Sede = Nome
   AND Citta = 'Roma';

c. Formulare in SQL, l'interrogazione che conta, per ciascun progetto, gli impiegati che lavorano ad esso:

Se interessano solo i codici dei progetti:
SELECT Progetto, count (*)
FROM Impiegati
GROUP BY Progetto;

Se invece interessano anche i titoli:
SELECT Progetto, Titolo, count (*)
FROM Impiegati, Progetti
WHERE Progetto = Codice
GROUP BY Progetto, Titolo;

Se ci sono progetti cui non lavora nessun impiegato:
SELECT Codice, Titolo, count(matricola)
FROM Progetti **LEFT JOIN** Impiegati **ON** Codice = Progetto
GROUP BY Codice;

Se ci sono progetti cui non lavora nessun impiegato, ma il sistema non supporta il join esterno:
SELECT Codice, Titolo, count(Matricola)
FROM Progetti **JOIN** Impiegati *ON* Codice = Progetto
GROUP BY Codice, Titolo
UNION
SELECT Codice, Titolo, 0
FROM Progetti
WHERE Codice **NOT IN** (**SELECT** Progetto
FROM Impiegati);

6. **Esercizio risolto**. Con riferimento ad una relazione
Soci(Codice, Cognome, Nome, Categoria, Età)
scrivere le interrogazioni SQL che calcolano l'età media dei soci di ciascuna categoria, nei due casi seguenti:
   a. se l'età non è nota si usa per essa il valore nullo;
      SELECT Categoria, **AVG**(Eta) AS EtaMedia
      FROM Soci
      GROUP BY Categoria
   b. se l'età non è nota si usa per essa il valore 0.
      SELECT Categoria, **AVG**(Eta) AS EtaMedia
      FROM Soci
      WHERE Eta <> 0
      GROUP BY Categoria;

7. **Esercizio risolto**. Scrivere un'interrogazione che calcola la differenza di due relazioni R e S definite entrambe sugli attributi A e B.
SELECT *
FROM R
WHERE NOT EXISTS (**SELECT** *
FROM S
WHERE S.A = R.A **AND** S.B = R.B)

8. **Esercizio risolto**. Considerare le relazioni *R1(A,B,C)* e *R2(D,E, F)* aventi rispettivamente cardinalità n1 e n2. Assumere che sia definito un vincolo di integrità referenziale fra l'attributo C di R1 e la chiave D di R2. Indicare la cardinalità di ciascuno dei seguenti join (specificando l'intervallo nel quale essa pu`o variare):
   a. R1 **JOIN** $_{A=D}$ R2 (compresa fra 0 e il minimo fra n1 e n2)
   b. R1.**JOIN** $_{C=D}$ R2 (esattamente n1)
   c. R1.**JOIN** $_{A=F}$ R2 (compresa fra 0 e n2)
   d. R1 **JOIN** $_{B=E}$ R2 (compresa fra 0 e n1 × n2)

9. **Esercizio risolto**. Considerare una base di dati relativa a studenti ed esami da essi superati:
Studenti(Matricola, Cognome, Nome)
Esami(Studente,Materia,Voto,Data)
con vincolo di integrità referenziale fra l'attributo *Studente* di *Esami* e la relazione *Studenti*.
Formulare in algebra relazionale le seguenti interrogazioni:
   a. Elencare matricola, cognome e nome degli studenti che hanno preso almeno un 30:
      PROJ $_{Matricola,Cognome,Nome}$ (Studenti **JOIN** $_{Matricola=Studente}$ **SEL** $_{Voto=30}$ (Esami))
   b. Elencare matricola, nome e nome degli studenti che hanno superato almeno un esame dopo il 1/1/2000.
      PROJ $_{Matricola,Cognome,Nome}$ (Studenti **JOIN** $_{Matricola=Studente}$ **SEL** $_{Data>1/1/2000}$ (Esami))
   c. Elencare i numeri di matricola degli studenti che hanno superato almeno due esami dopo il 1/1/2000.
      PROJ $_{Matricola}$
      (**SEL** Data1>1/1/2000 **AND** Data2>1/1/2000 AND Materia1<> Materia2
      (Esami **JOIN** $_{Studente1=Studente2}$
      (REN $_{Studente2, Materia2, Data2, Voto2 \leftarrow Studente1, Materia1, Data1, Voto1}$ (Esami))))
   d. Elencare matricola, cognome e nome degli studenti che hanno preso tutti 30.
      PROJ $_{Matricola, Cognome, Nome}$ (Studenti **JOIN** $_{Matricola=Studente}$ **SEL** $_{Voto=30}$ (Esami))−
      PROJ $_{Matricola, Cognome, Nome}$ (Studenti **JOIN** $_{Matricola=Studente}$ **SEL** $_{Voto<30}$ (Esami))

      Notare che, definendo una vista StudEsami per mezzo dell'espressione:
      Studenti **JOIN** $_{Matricola=Studente}$ Esami
      le interrogazioni si possono scrivere in modo più compatto, come segue:

a. **PROJ** $_{\text{Matricola, Cognome, Nome}}$ (**SEL** $_{\text{Voto=30}}$ (StudEsami))
b. **PROJ** $_{\text{Matricola, Cognome, Nome}}$ (**SEL** _Data>1/1/2000 (StudEsami))
c. Inalterata
d. **PROJ** $_{\text{Matricola, Cognome, Nome}}$ (**SEL** $_{\text{Voto=30}}$ (StudEsami))–
PROJ $_{\text{Matricola, Cognome, Nome}}$ (**SEL** $_{\text{Voto<30}}$ (StudEsami))

10. Considerare le relazioni
Dipartimenti(Codice, Direttore)
Impiegati(Matricola, Nome, Stipendio, Direttore)
Scrivere interrogazioni per:
a. Calcolare la media degli stipendi degli impiegati;
b. Calcolare la media degli stipendi degli impiegati aventi come direttore D.