1) Dato il seguente schema di una **base di dati contenente dati relativi a trasporti extraurbani**

FERMATA(#id, nome, via, comune)

TRATTA(#id, #idp, #ida)

PERCORSO(#idt, #idf, numero)

ORARIO(#idt, orapart)

1a) Trovare id della tratta, nome e via della fermata di partenza, nome e via della fermata di arrivo di tratte che partono dal Comune di Guidonia e arrivano nel comune di Roma con partenza tra le ore 8:00 e le ore 9:00

1b) Trovare id della tratta e nome della fermata di arrivo per le tratte che hanno la prima partenza prima delle 6:00 del mattino

1 a) $TP := \Pi_{TRATTA.ID}(\sigma_{COMUNE = GUIDONIA}(FERMATA \bowtie_{FERMATA.ID = IDP}(TRATTA)))$ $\quad TA := \Pi_{TRATTA.ID}(\sigma_{COMUNE = ROMA}(FERMATA \bowtie_{FERMATA.ID = IDA}(TRATTA)))$

$TROMAGUID = \Pi_{TRATTA.ID}(\sigma_{ORAPART \geq 8.00 \wedge ORAPART \leq 9.00}((TP \cap TA) \bowtie_{TRATTA.ID = IDT} ORARIO))$ ← id delle fermate in questione!

$Fc := \rho_{HID \to IDC, NOME \to NOMEC, VIA \to VIAC}(FERMATA)$ $\quad Q = \Pi_{TRATTA.ID, NOME, VIA, NOMEC, VIAC}((FERMATA \bowtie_{FERMATA.ID = IDP}(TRATTA \bowtie TROMAGUID)) \bowtie_{IDA = IDC} FC)$

1b) $OC = \rho_{ORAPART \to ORAPARTC}(ORARIO)$ $\quad COMP = \Pi_{HIDT, ORAPART}(ORARIO \bowtie OC - \sigma_{ORAPART > ORAPARTC}(ORARIO \bowtie OC))$

in COMP ho tutte le tratte con l'orario della loro prima partenza! $BEFORE6 = \sigma_{ORAPART < 6:00}(COMP)$

Query finale: $Q = \Pi_{IDT, NOME}((TRATTA \bowtie_{HIDA = FERMATA.ID}(FERMATA)) \bowtie_{TRATTA.ID = HIDT} BEFORE6)$

2a) Dati lo schema di relazione R=ABCDE, l'insieme di dipendenze funzionali F={AB→C, AD→E, C→BE, BD→E} e la decomposizione ρ={ABCE,CDE} di R,

dire se ρ preserva F e illustrare il procedimento seguito per giungere alla risposta

2b) Dati lo schema di relazione R=ABCDEG, l'insieme di dipendenze funzionali

F={ A→G, GA→DB, B→AC, BE→A, AD→EB} e la decomposizione ρ={ABG, ADE, CDE} di R,

dire se ρ ha un join senza perdita e illustrare il procedimento seguito per giungere alla risposta

2 a) Necessito di controllare se $F \equiv G$, dove $G = \bigcup_{i=1}^{2} \Pi_{R_i}(F)$, $\forall X \to Y \in F$, devo calcolare la chiusura di X rispetto a G, e controllare che ci sia Y. Applico l'algoritmo, escludendo le $X \to Y \in F$ tali che $XY \subseteq R_i \in \rho$.

- $AD \to E$, $Z_0 = AD$ $\quad S_0 = \begin{matrix} (AD \cap ABCE)^+_F \cap ABCE \\ \cup \\ (AD \cap CDE)^+_F \cap CDE \end{matrix} = \begin{matrix} A^+_F \cap ABCE \\ \cup \\ D^+_F \cap CDE \end{matrix} = \begin{matrix} A \\ \cup \\ D \end{matrix}$ $\quad U = AD \Rightarrow S_0 \subseteq Z_0$? SI $\Rightarrow AD^+_G = \{AD\} \not\ni E \Rightarrow$ non preserva F.

2 b) Data una decomposizione, applico l'algoritmo che costruisce la tabella:

	A	B	C	D	E	G
ABG	a	a	b_1	b_1	b_1	a
ADE	a	b_2	b_2	a	a	b_2
CDE	b_3	b_3	a	a	a	b_3

$A \to G$ ⟹

	A	B	C	D	E	G
ABG	a	a	b_1	b_1	b_1	a
ADE	a	b_2	b_2	a	a	a
CDE	b_3	b_3	a	a	a	b_3

$GA \to DB$ ⟹

	A	B	C	D	E	G
ABG	a	a	b_1	a	b_1	a
ADE	a	a	b_2	a	a	a
CDE	b_3	b_3	a	a	a	b_3

$B \to AC$ ⟹

	A	B	C	D	E	G
ABG	a	a	b_2	a	b_1	a
ADE	a	a	b_2	a	a	a
CDE	b_3	b_3	a	a	a	b_3

$AD \to EB$ ⟹

	A	B	C	D	E	G
ABG	a	a	b_2	a	a	a
ADE	a	a	b_2	a	a	a
CDE	b_3	b_3	a	a	a	b_3

⇒ l'algoritmo non fa modifiche e termina, non vi è una riga di sole "a", quindi non ha un Join senza perdita.

---

3) Supponiamo di avere un file di 12.000.000 record. Ogni record occupa 275 byte, di cui 25 per il campo chiave. Ogni blocco contiene 2048 byte. Un puntatore a blocco occupa 5 byte. Usiamo una organizzazione B-tree **in cui ogni blocco del file principale punta al prossimo blocco nel file principale (B⁺-tree). I blocchi sia del file principale sono pieni al massimo mentre i blocchi del file indice sono pieni al minimo. Calcolare:**

- il numero di blocchi del file principale
- il numero di blocchi del file indice
- il numero di accessi necessari per ricercare un record del file principale

Da un blocco del MainFile, escludo 5 byte di puntatore, ogni blocco contiene $\lfloor 2043/275 \rfloor = 7$ record quindi, si ha che il numero di blocchi del MainFile è: $\lceil \frac{12\,000\,000}{7} \rceil = 1714286$. Una coppia key-pointer pesa 30 byte, in un blocco riempito al minimo ce ne sono $\lceil \frac{1024-5}{30} \rceil = 33$, quindi vi sono 34 puntatori.

LIV1: $\lceil \frac{17142867}{34} \rceil = 50421$ blocchi $\quad$ LIV2: $\lceil \frac{50421}{34} \rceil = 1483$ blocchi $\quad$ LIV3: $\lceil \frac{1483}{34} \rceil = 44$ $\quad$ LIV4: $\lceil \frac{44}{34} \rceil = 2$ $\quad$ LIV5: 1 (RADICE)

Nel file indice ci sono $50421 + 1483 + 44 + 2 + 1 = 51951$ blocchi. Supponendo che la radice non sia caricata in RAM, per accedere ad un record sono necessari 6 accessi.