# Basi di dati (nuovo ordinamento) — 19 luglio 2005 — Compito A

Tempo a disposizione: due ore. Libri chiusi.

**Cognome:** _____ **Nome:** _____ **Matricola:** _____

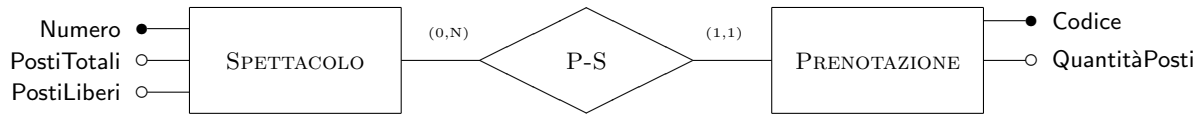
**Domanda 1** (25%) Mostrare uno schema concettuale che rappresenti una realtà i cui dati siano organizzati per mezzo del seguente schema relazionale.

- CICLISTA(Codice, Cognome, Nome, Squadra)
- COMPETIZIONE(Codice, Nome, Organizzatore, KmTotali)
- TAPPA(Numero, Competizione, Partenza, Arrivo, Km) con vincolo di integrità referenziale fra Competizione e la relazione COMPETIZIONE
- CLASSIFICATAPPA(NumeroTappa, Competizione, Ciclista, Posizione, Distacco) con vincoli di integrità referenziale fra gli attributi NumeroTappa,Competizione e la relazione TAPPA e fra Ciclista e la relazione CICLISTA
- CLASSIFICAGENERALE(NumeroTappa, Competizione, Ciclista, Posizione, Distacco) con vincoli di integrità referenziale fra gli attributi NumeroTappa,Competizione e la relazione TAPPA e fra Ciclista e la relazione CICLISTA

**Domanda 2** (20%) Modificare lo schema ottenuto in risposta alla domanda precedente, assumendo che

- ciascuna competizione si ripeta ogni anno, con lo stesso organizzatore ma diverso numero di Km totali.
- per ogni località di partenza e arrivo interessi memorizzare l'altitudine
- ogni squadra abbia una sigla, un nome e un capitano (che è un ciclista)

**Domanda 3** (15%) Lo schema concettuale seguente rappresenta un insieme di spettacoli e un insieme di prenotazioni ognuna delle quali fa riferimento (attraverso la relazione P-S) ad uno spettacolo. In particolare, l'attributo PostiLiberi di una occorrenza di SPETTACOLO è pari alla differenza fra il valore di PostiTotali per lo stesso SPETTACOLO e la somma del numero di posti prenotati per quello spettacolo (cioè alla somma dei valori dell'attributo QuantitàPosti delle occorrenze dell'entità PRENOTAZIONE cui l'occorrenza di SPETTACOLO è correlata tramite P-S).




Valutare se convenga o meno mantenere la ridondanza, tenendo conto del fatto che le cardinalità delle due entità sono $N_{\mathrm{SP}} = 1.000$ e $N_{\mathrm{PRE}} = 1.000.000$ e che le operazioni più importanti sono:

OP$_1$ lettura del numero di posti disponibili per uno spettacolo, con frequenza $f_1 = 100$
OP$_2$ inserimento di una prenotazione, con con frequenza $f_2 = 10.000$

Assumere che il costo di una lettura e quello di una scrittura siano uguali e che non vi sia costo associato alla lettura o scrittura della relationship P-S.

**Domanda 4** (25%) Considerare la base di dati relazionale definita per mezzo delle seguenti istruzioni:

```
create table impiegati (
   cf numeric not null primary key,
   titolo char(5),
   cognome char(20) not null,
   nome char(20) not null,
   stipendio numeric not null,
   dip numeric not null references dipartimenti(codice)
   );
create table Dipartimenti (
   codice numeric not null primary key,
   nomeDip char(20) not null unique,
   indirizzo char(30) not null );
```

Formulare

1. in SQL l'interrogazione che trova i nomi dei dipartimenti che non hanno impiegati
2. in algebra l'interrogazione che trova i nomi dei dipartimenti che hanno almeno due impiegati
3. in SQL l'interrogazione che trova, per ogni dipartimento, lo stipendio medio degli impiegati, mostrando codice del dipartimento e stipedio medio
4. in SQL l'interrogazione che trova gli impiegati che guadagnano più della media degli impiegati del proprio dipartimento; mostrare i dati dell'impiegato e lo stipendio medio del dipartimento.

**Domanda 5** (15%) Con riferimento alla base di dati già utilizzata nella domanda precedente, considerare il metodo mostrato sotto, che avrebbe lo scopo di stampare, per ciascun dipartimento, le informazioni sintetiche (nome e indirizzo) e, subito dopo, la lista degli impiegati.

Riguardo a tale metodo:

1. il codice SQL contiene un errore, a causa del quale la stampa può non avvenire correttamente; individuare tale errore e correggerlo
2. modificare il metodo in modo che stampi anche le informazioni sui dipartimenti che non hanno impiegati (che sono invece ignorati nella versione attuale)

```
static void stampaDati(Connection connection) throws SQLException {
    Statement statement = connection.createStatement();
    String query = "select * from dipartimenti JOIN impiegati ON codice = dip " ;
    ResultSet resultSet = statement.executeQuery(query);
    boolean primo = true ;
    int codice = 0;
    int codicePrec = 0;
    while (resultSet.next()){
        primo = false ;
        codice = resultSet.getInt("codice");
        if (primo || !(codice==codicePrec)){
            System.out.println("Dipartimento: " + codice +
                " Nome: " + resultSet.getString("NomeDip") +
                " Indirizzo: " + resultSet.getString("Indirizzo") +
                "\nImpiegati");
        }
        String titolo = resultSet.getString("titolo");
        // Nota bene (dalla documentazione di Java)
        // boolean wasNull(): reports whether the last column read had a value of SQL NULL.
        if ((resultSet.wasNull())){ titolo = "     ";}
        System.out.println(" " + resultSet.getInt("cf") + " " + titolo +
            resultSet.getString("Nome") + " " + resultSet.getString("Cognome"));
        codicePrec = codice ;
    }
}
```

# Basi di dati (nuovo ordinamento) — 19 luglio 2005 — Compito B

Tempo a disposizione: due ore. Libri chiusi.

**Cognome:** _____ **Nome:** _____ **Matricola:** _____

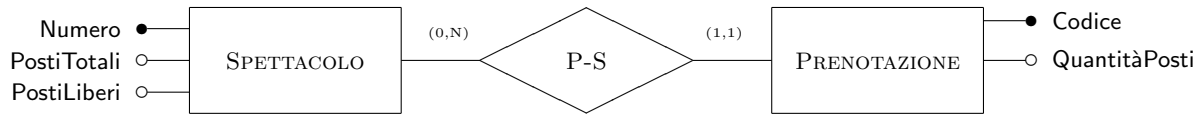
**Domanda 1** (25%) Mostrare uno schema concettuale che rappresenti una realtà i cui dati siano organizzati per mezzo del seguente schema relazionale.

- CICLISTA(Codice, Cognome, Nome, Squadra)
- COMPETIZIONE(Codice, Nome, Organizzatore, NumeroDiTappe)
- TAPPA(Numero, Competizione, Partenza, Arrivo, Km) con vincolo di integrità referenziale fra Competizione e la relazione COMPETIZIONE
- CLASSIFICATAPPA(NumeroTappa, Competizione, Ciclista, Posizione, Distacco) con vincoli di integrità referenziale fra gli attributi NumeroTappa,Competizione e la relazione TAPPA e fra Ciclista e la relazione CICLISTA
- CLASSIFICAGENERALE(NumeroTappa, Competizione, Ciclista, Posizione, Distacco) con vincoli di integrità referenziale fra gli attributi NumeroTappa,Competizione e la relazione TAPPA e fra Ciclista e la relazione CICLISTA

**Domanda 2** (20%) Modificare lo schema ottenuto in risposta alla domanda precedente, assumendo che

- ciascuna competizione si ripeta ogni anno, con lo stesso organizzatore ma diverso numero di tappe.
- per ogni località di partenza e arrivo interessi memorizzare l'altitudine
- ogni squadra abbia una sigla, un nome e un capitano (che è un ciclista)

**Domanda 3** (15%) Lo schema concettuale seguente rappresenta un insieme di spettacoli e un insieme di prenotazioni ognuna delle quali fa riferimento (attraverso la relazione P-S) ad uno spettacolo. In particolare, l'attributo PostiLiberi di una occorrenza di SPETTACOLO è pari alla differenza fra il valore di PostiTotali per lo stesso SPETTACOLO e la somma del numero di posti prenotati per quello spettacolo (cioè alla somma dei valori dell'attributo QuantitàPosti delle occorrenze dell'entità PRENOTAZIONE cui l'occorrenza di SPETTACOLO è correlata tramite P-S).




Valutare se convenga o meno mantenere la ridondanza, tenendo conto del fatto che le cardinalità delle due entità sono $N_{\mathrm{SP}} = 1.000$ e $N_{\mathrm{PRE}} = 1.000.000$ e che le operazioni più importanti sono:

OP$_1$ inserimento di una prenotazione, con frequenza $f_1 = 100$
OP$_2$ lettura del numero di posti disponibili per uno spettacolo, con con frequenza $f_2 = 10.000$

Assumere che il costo di una lettura e quello di una scrittura siano uguali e che non vi sia costo associato alla lettura o scrittura della relationship P-S.

**Domanda 4** (25%) Considerare la base di dati relazionale definita per mezzo delle seguenti istruzioni:

```
create table impiegati (
   cf numeric not null primary key,
   titolo char(5),
   cognome char(20) not null,
   nome char(20) not null,
   stipendio numeric not null,
   dip numeric not null references dipartimenti(codice)
   );
create table Dipartimenti (
   codice numeric not null primary key,
   nomeDip char(20) not null unique,
   indirizzo char(30) not null );
```

Formulare

1. in algebra relazionale l'interrogazione che trova i nomi dei dipartimenti che non hanno impiegati
2. in SQL l'interrogazione che trova i nomi dei dipartimenti che hanno almeno due impiegati
3. in SQL l'interrogazione che trova, per ogni dipartimento, lo stipendio medio degli impiegati, mostrando codice del dipartimento e stipedio medio
4. in SQL l'interrogazione che trova gli impiegati che guadagnano più della media degli impiegati del proprio dipartimento; mostrare i dati dell'impiegato e lo stipendio medio del dipartimento.

**Domanda 5** (15%) Con riferimento alla base di dati già utilizzata nella domanda precedente, considerare il metodo mostrato sotto, che avrebbe lo scopo di stampare, per ciascun dipartimento, le informazioni sintetiche (nome e indirizzo) e, subito dopo, la lista degli impiegati.

Riguardo a tale metodo:

1. il codice SQL contiene un errore, a causa del quale la stampa può non avvenire correttamente; individuare tale errore e correggerlo
2. modificare il metodo in modo che stampi anche le informazioni sui dipartimenti che non hanno impiegati (che sono invece ignorati nella versione attuale)

```
static void stampaDati(Connection connection) throws SQLException {
    Statement statement = connection.createStatement();
    String query = "select * from dipartimenti JOIN impiegati ON codice = dip " ;
    ResultSet resultSet = statement.executeQuery(query);
    boolean primo = true ;
    int codice = 0;
    int codicePrec = 0;
    while (resultSet.next()){
        primo = false ;
        codice = resultSet.getInt("codice");
        if (primo || !(codice==codicePrec)){
            System.out.println("Dipartimento: " + codice +
                " Nome: " + resultSet.getString("NomeDip") +
                " Indirizzo: " + resultSet.getString("Indirizzo") +
                "\nImpiegati");
        }
        String titolo = resultSet.getString("titolo");
        // Nota bene (dalla documentazione di Java)
        // boolean wasNull(): reports whether the last column read had a value of SQL NULL.
        if ((resultSet.wasNull())){ titolo = "     ";}
        System.out.println(" " + resultSet.getInt("cf") + " " + titolo +
            resultSet.getString("Nome") + " " + resultSet.getString("Cognome"));
        codicePrec = codice ;
    }
}
```