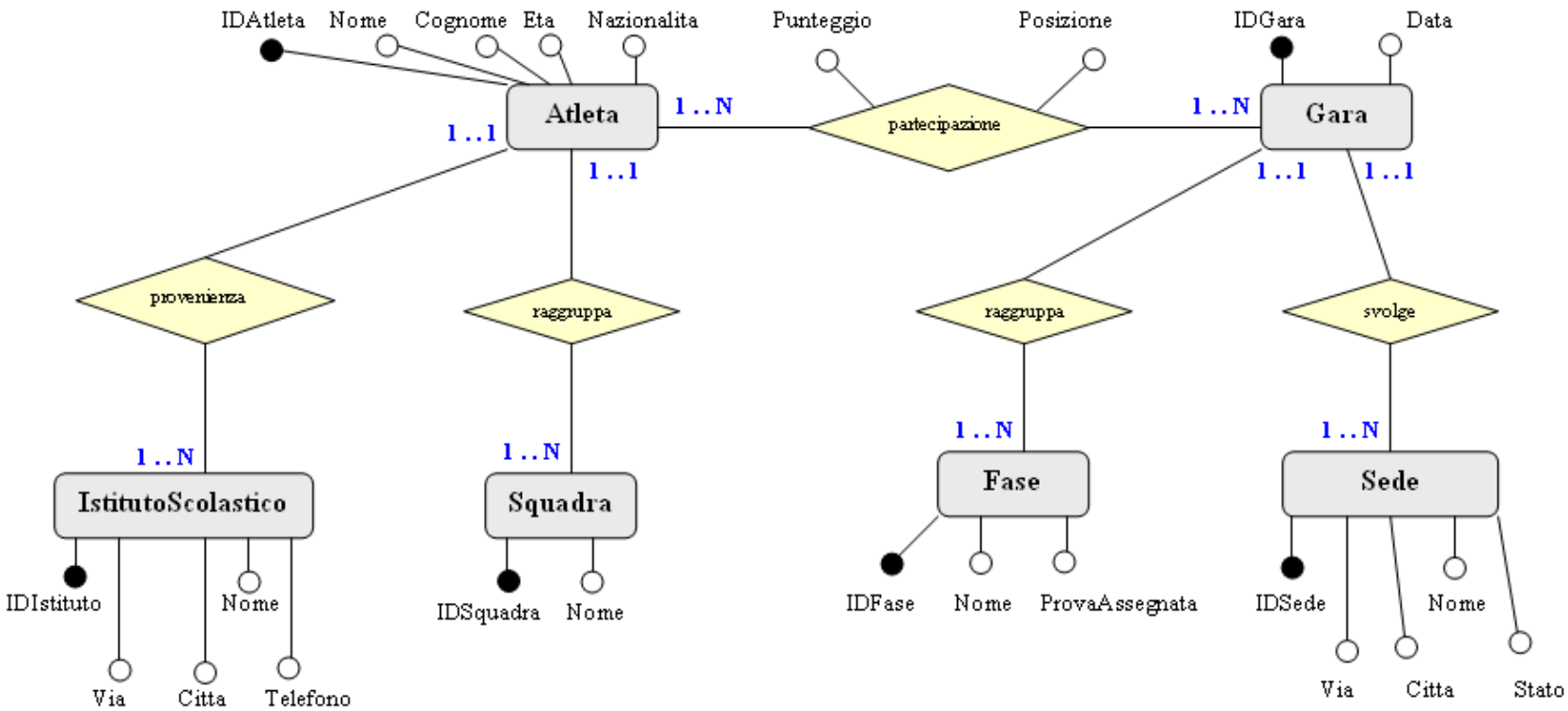
1. Analisi

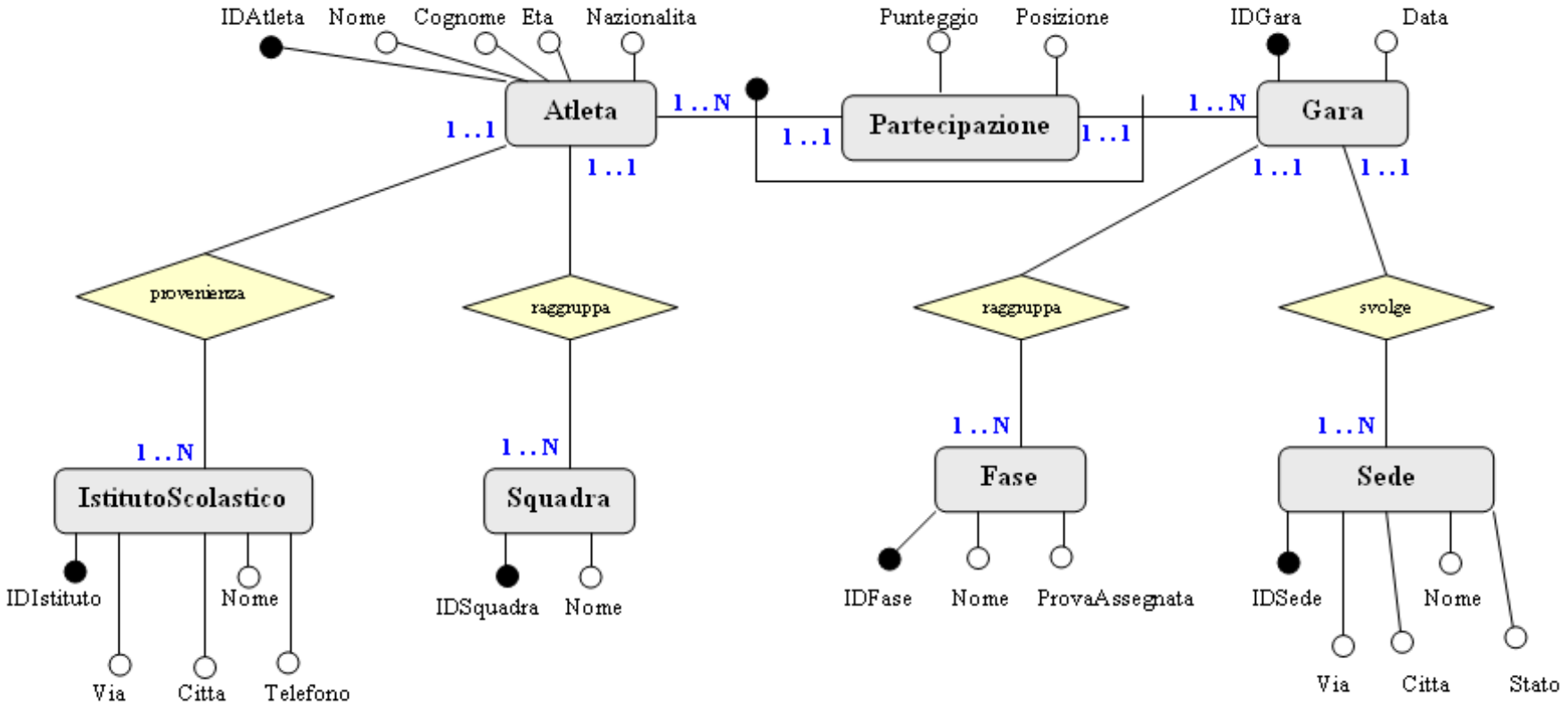
NOTA : *La soluzione che segue si riferisce al testo relativo all'Esame di Stato di Informatica per l'indirizzo Informatica (a.s. 2007/2008), modificato in alcune interrogazioni SQL. Quella che segue è una possibile soluzione, versione semplificata che ipotizza l'appartenenza sempre e comunque di un atleta ad una squadra. Nei paragrafi 2.3 e 2.4 è proposta un'altra soluzione.*

2. Schema Concettuale

2.1. Primo diagramma E/R

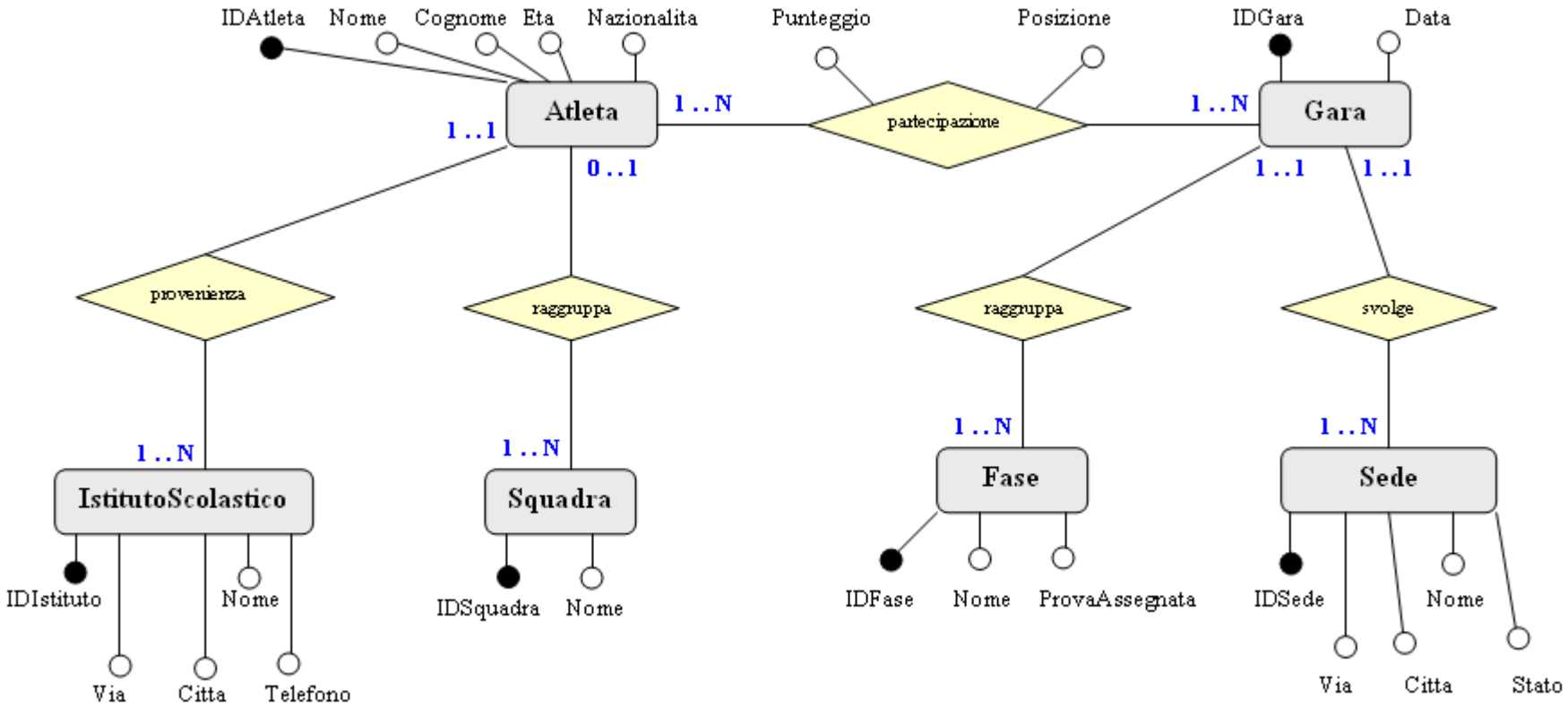


2.2. Secondo diagramma E/R



Viene ristrutturata l'associazione n..n tra *Atleta* e *Gara*. Come chiave primaria della tabella *Partecipazione*, si sceglie la coppia formata dalle due chiavi esterne (*FKAtleta* ed *FKGara*), data l'univocità della coppia per ogni istanza di *Partecipazione*: infatti, un atleta potrà partecipare soltanto una volta a quella particolare gara che si riferisce ad una determinata fase.

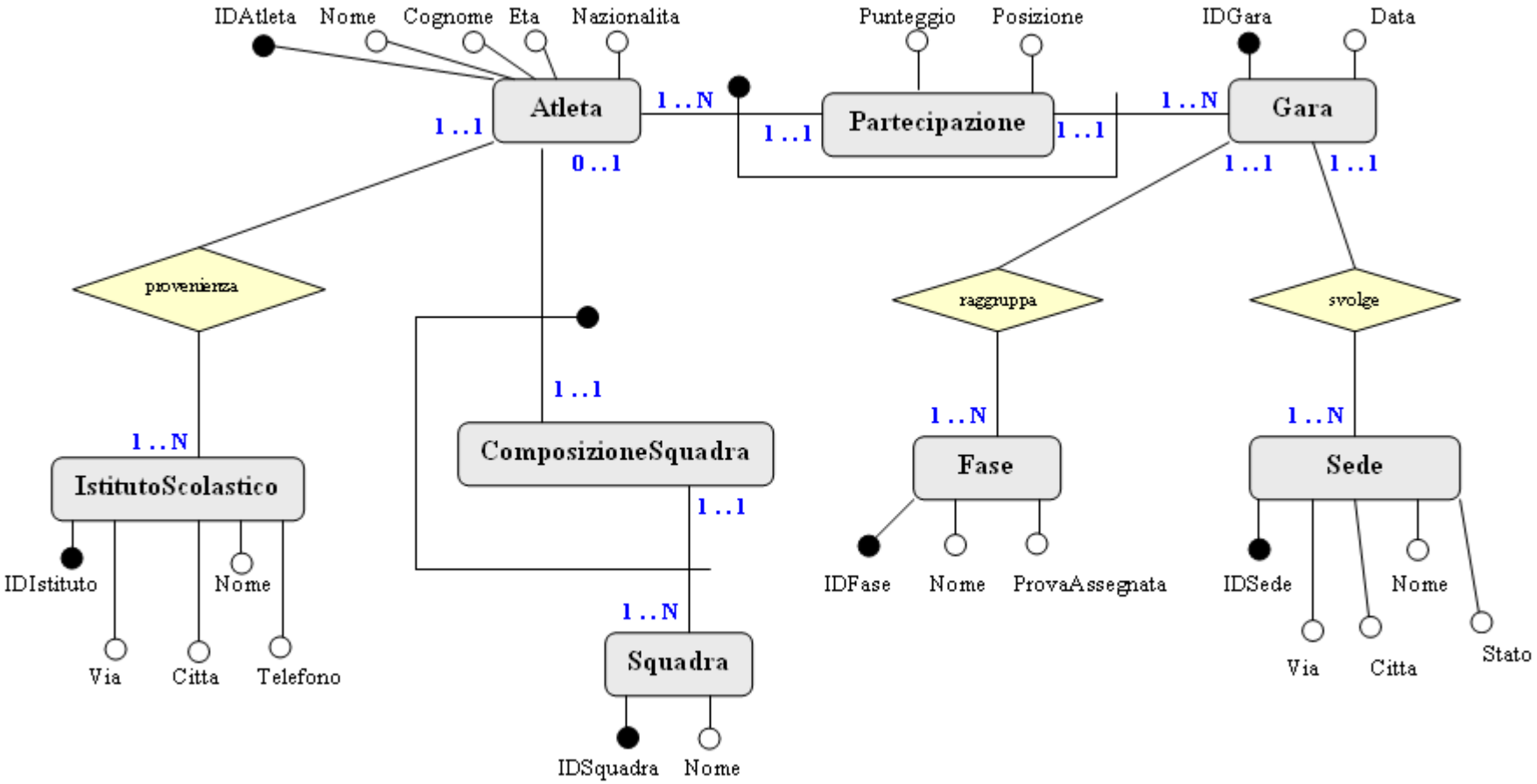
2.3. Seconda Soluzione - Primo diagramma E/R



In questa soluzione si ipotizza che l'atleta possa o meno far parte di una squadra (la cardinalità della relazione uscente da *Atleta* verso *Squadra* è 0..1).

È a questo modello E/R che si riferisce il resto della soluzione (modello logico, DDL ed interrogazioni SQL).

2.4. Seconda Soluzione - Secondo diagramma E/R



Per evitare problemi di valori *null* nella chiave esterna *FKSquadra* che avrebbe avuto la tabella Atleta, si inserisce un'ulteriore tabella ***ComposizioneSquadra***, la cui chiave primaria è costituita dalle due foreign key *FKAtleta* ed *FKSquadra*. I valori di questi due campi saranno sempre *not null*, in quanto verranno catalogati in essa soltanto gli atleti che sono stati assegnati ad una squadra. Per la ristrutturazione dell'associazione n..n tra *Atleta* e *Gara*, vale quanto detto nel paragrafo 2.2 .

3. Schema Logico

Tabella	Campo	Formato	Not Null	Chiave	Relazione
IstitutoScolastico	IDIstituto	Integer - Autoincrementale	si	Primaria	
	Nome	String (25)	si		
	Via	String (25)	si		
	Citta	String (25)	si		
	Telefono	String (15)	si		
Squadra	IDSquadra	Integer - Autoincrementale	si	Primaria	
	Nome	String (25)	si		
Atleta	IDAtleta	Integer - Autoincrementale	si	Primaria	
	Nome	String (25)	si		
	Cognome	String (25)	si		
	Eta	Integer	si		
	Nazionalita	String (25)	si		
	FKIstituto	Integer	si	Esterna	Istituto.IDIstituto
ComposizioneSquadra	FKAtleta	Integer	si	Primaria / Esterna	Atleta.IDAtleta
	FKSquadra	Integer	si	Primaria / Esterna	Squadra.IDSquadra
Fase	IDFase	Integer - Autoincrementale	si	Primaria	
	Nome	String (25)	si		
	ProvaAssegnata	String (50)	si		
Sede	IDSede	Integer - Autoincrementale	si	Primaria	
	Nome	String (25)	si		
	Via	String (25)	si		
	Citta	String (25)	si		
	Stato	String (25)	si		
Gara	IDGara	Integer - Autoincrementale	si	Primaria	
	Data	Date	si		
	FKFase	Integer	si	Esterna	Fase.IDFase
	FKSede	Integer	si	Esterna	Sede.IDSede
Partecipazione	FKAtleta	Integer	si	Primaria / Esterna	Atleta.IDAtleta
	FKGara	Integer	si	Primaria / Esterna	Gara.IDGara
	Punteggio	Integer	si		
	Posizione	Integer	si		

Per il campo Fase.ProvaAssegnata, si è scelto il formato String, andando pertanto a memorizzare l'indirizzo del file che conterrà la prova.

4. Definizione Relazioni

DBMS di riferimento : ORACLE

TABELLA ISTITUTO

```
CREATE TABLE Istituto
(    IDIstituto int not null,
     Nome varchar(25) not null,
     Via varchar(25) not null,
     Citta varchar(25) not null,
     Telefono varchar(15) not null);
```

Definizione Chiave Primaria

```
ALTER TABLE Istituto ADD CONSTRAINT PKIstituto PRIMARY KEY (IDIstituto);
```

Definizione formato autoincrementale per il campo IDIstituto (si crea una sequenza gestita tramite un trigger)

```
CREATE SEQUENCE SeqIstituto
INCREMENT BY 1
START WITH 1;


CREATE OR REPLACE TRIGGER T_Seq_Istituto
BEFORE INSERT ON Istituto FOR EACH ROW
BEGIN
      SELECT SeqIstituto.nextval INTO :new.IDIstituto FROM DUAL;
END;
```

TABELLA SQUADRA

```
CREATE TABLE Squadra
(    IDSquadra int not null,
     Nome varchar(25) not null);
```

Definizione Chiave Primaria

```
ALTER TABLE Squadra ADD CONSTRAINT PKSquadra PRIMARY KEY (IDSquadra);
```

Definizione formato autoincrementale per il campo IDSquadra (si crea una sequenza gestita tramite un trigger)

```
CREATE SEQUENCE SeqSquadra
INCREMENT BY 1
START WITH 1;


CREATE OR REPLACE TRIGGER T_Seq_Squadra
BEFORE INSERT ON Squadra FOR EACH ROW
BEGIN
      SELECT SeqSquadra.nextval INTO :new.IDSquadra FROM DUAL;
END;
```

TABELLA ATLETA

```
CREATE TABLE Atleta
(    IDAtleta int not null,
     Nome varchar(25) not null,
     Cognome varchar(25) not null,
     Eta int not null,
     Nazionalita varchar(15) not null,
     FKIstituto int not null);
```

Definizione Chiave Primaria

```
ALTER TABLE Atleta ADD CONSTRAINT PKAtleta PRIMARY KEY (IDAtleta);
```

Definizione Chiavi Esterne

```
ALTER TABLE Atleta ADD CONSTRAINT FKAtletaIstituto FOREIGN KEY (FKIstituto) REFERENCES Istituto(IDIstituto);
```

Definizione formato autoincrementale per il campo IDAtleta (si crea una sequenza gestita tramite un trigger)

```
CREATE SEQUENCE SeqAtleta
INCREMENT BY 1
START WITH 1;


CREATE OR REPLACE TRIGGER T_Seq_Atleta
BEFORE INSERT ON Atleta FOR EACH ROW
BEGIN
      SELECT SeqAtleta.nextval INTO :new.IDAtleta FROM DUAL;
END;
```

TABELLA COMPOSIZIONESQUADRA

```
CREATE TABLE ComposizioneSquadra
(    FKAtleta int not null,
     FKSquadra int not null);
```

Definizione Chiave Primaria

```
ALTER TABLE ComposizioneSquadra ADD CONSTRAINT PKComposizioneSquadra PRIMARY KEY (FKAtleta, FKSquadra);
```

Definizione Chiavi Esterne

```
ALTER TABLE ComposizioneSquadra ADD CONSTRAINT CompSq_Atleta FOREIGN KEY (FKAtleta) REFERENCES Atleta(IDAlteta);
ALTER TABLE ComposizioneSquadra ADD CONSTRAINT CompSq_Squadra FOREIGN KEY (FKSquadra) REFERENCES Squadra(IDSquadra);
```

TABELLA FASE

```
CREATE TABLE Fase
(    IDFase int not null,
     Nome varchar(25) not null,
     ProvaAssegnata varchar(50) not null);
```

Definizione Chiave Primaria

```
ALTER TABLE Fase ADD CONSTRAINT PKFase PRIMARY KEY (IDFase);
```

Definizione formato autoincrementale per il campo IDFase (si crea una sequenza gestita tramite un trigger)

```
CREATE SEQUENCE SeqFase
INCREMENT BY 1
START WITH 1;


CREATE OR REPLACE TRIGGER T_Seq_Fase
BEFORE INSERT ON Fase FOR EACH ROW
BEGIN
      SELECT SeqFase.nextval INTO :new.IDFase FROM DUAL;
END;
```

TABELLA SEDE

```
CREATE TABLE Sede
(    IDSede int not null,
     Nome varchar(25) not null,
     Via varchar(25) not null,
     Citta varchar(25) not null,
     Stato varchar(25) not null);
```

Definizione Chiave Primaria

```
ALTER TABLE Sede ADD CONSTRAINT PKSede PRIMARY KEY (IDSede);
```

Definizione formato autoincrementale per il campo IDSede (si crea una sequenza gestita tramite un trigger)

```
CREATE SEQUENCE SeqSede
INCREMENT BY 1
START WITH 1;


CREATE OR REPLACE TRIGGER T_Seq_Sede
BEFORE INSERT ON Sede FOR EACH ROW
BEGIN
      SELECT SeqSede.nextval INTO :new.IDSede FROM DUAL;
END;
```

TABELLA GARA

```
CREATE TABLE Gara
(    IDGara int not null,
     Data date not null,
     FKFase int not null,
     FKSede int not null);
```

Definizione Chiave Primaria

```
ALTER TABLE Gara ADD CONSTRAINT PKGara PRIMARY KEY (IDGara);
```

Definizione Chiavi Esterne

```
ALTER TABLE Atleta ADD CONSTRAINT FKGaraFase FOREIGN KEY (FKFase) REFERENCES Fase(IDFase);
ALTER TABLE Atleta ADD CONSTRAINT FKGaraSede FOREIGN KEY (FKSede) REFERENCES Sede(IDSede);
```

Definizione formato autoincrementale per il campo IDGara (si crea una sequenza gestita tramite un trigger)

```
CREATE SEQUENCE SeqGara
INCREMENT BY 1
START WITH 1;


CREATE OR REPLACE TRIGGER T_Seq_Gara
BEFORE INSERT ON Gara FOR EACH ROW
BEGIN
      SELECT SeqGara.nextval INTO :new.IDGara FROM DUAL;
END;
```

TABELLA PARTECIPAZIONE

```
CREATE TABLE Partecipazione
(    FKAtleta int not null,
     FKGara int not null,
     Punteggio int not null,
     Posizione int not null);
```

Definizione Chiave Primaria

```
ALTER TABLE Partecipazione ADD CONSTRAINT PKPartecipazione PRIMARY KEY (FKAtleta, FKGara);
```

Definizione Chiavi Esterne

```
ALTER TABLE Partecipazione ADD CONSTRAINT FKPartecipazioneAtleta FOREIGN KEY (FKAtleta) REFERENCES Atleta(IDAtleta);
ALTER TABLE Partecipazione ADD CONSTRAINT FKPartecipazioneGara FOREIGN KEY (FKGara) REFERENCES Gara(IDGara);
```

5. Interrogazioni SQL

a. Stampare l'elenco degli atleti raggruppati per squadre per ogni singola fase

Realizzata vedendo la fase come parametro di input :

```
SELECT Squadra.Nome, Atleta.Cognome, Atleta.Nome
FROM Atleta, Squadra, ComposizioneSquadra c, Partecipazione, Gara, Fase
WHERE Squadra.IDSquadra = ComposizioneSquadra.FKSquadra
AND Atleta.IDAtleta = ComposizioneSquadra.FKAtleta
AND Atleta.IDAtleta = Partecipazione.FKAtleta
AND Gara.IDGara = Partecipazione.FKGara
AND Fase.IDFase = Gara.FKFase
AND Fase.Nome = :valoreNomeFase
ORDER BY Squadra.Nome, Atleta.Cognome, Atleta.Nome;
```

b. Dato il nome di un atleta stampare i risultati ottenuti nelle diverse gare alle quali ha partecipato

```
SELECT f.Nome, p.Punteggio, p.Posizione
FROM Partecipazione p, Atleta a, Gara g, Fase f
WHERE a.IDAtleta = p.FKAtleta
AND f.IDFase = g.FKFase
AND g.IDGara = p.FKGara
AND a.Nome = :valoreNomeAtleta
AND a.Cognome = :valoreCognomeAtleta
ORDER BY f.Nome;
```

c. Stampare il calendario delle gare

Realizzata mostrando i giorni in cui si svolgono le gare relative alle varie fasi.

```
SELECT f.Nome, s.Nome, g.Data
FROM Fase f, Gara g, Sede s
WHERE f.IDFase = g.FKFase
AND s.IDSede = g.FKSede
ORDER BY f.Nome, s.Nome;
```

d. Stampare una scheda informativa (cognome, nome, istituto scolastico di provenienza, nazionalità) del vincitore e della squadra vincitrice

```
SELECT a.Cognome, a.Nome, a.Nazionalita, i.Nome, s.Nome
FROM Atleta a, Squadra s, ComposizioneSquadra c, IstitutoScolastico i, Partecipazione p, Gara g, Fase f
WHERE i.IDIstituto = a.FKIstituto
AND s.IDSquadra = c.FKSquadra
AND a.IDAtleta = c.FKAtleta
AND a.IDAtleta = p.FKAtleta
AND g.IDGara = p.FKGara
AND f.IDFase = g.FKFase
AND f.Nome = “Internazionale”
AND p.Punteggio = ( SELECT MAX (Punteggio)
                    FROM Partecipazione, Gara, Fase
                    WHERE Gara.IDGara = Partecipazione.FKGara
                    AND Fase.IDFase = Gara.FKFase
                    AND Fase.Nome = “Internazionale”);
```

e. Stampare la classifica per ciascuna gara (a parità di punteggio vengono privilegiati gli atleti più giovani)

```
SELECT f.Nome, s.Nome, a.Cognome, a.Nome, p.Punteggio
FROM Atleta a, Partecipazione p, Gara g, Fase f, Sede s
WHERE a.IDAtleta = p.FKAtleta
AND g.IDGara = p.FKGara
AND f.IDFase = g.FKFase
AND s.IDSede = g.FKSede
ORDER BY f.Nome, p.Punteggio DESC, a.Eta ASC;
```

f. Aggiornare al 23/04/2009 la data delle gare relative alla fase scolastica

```
UPDATE Gara
SET data = ‘04/23/2009’
WHERE FKFase = (  SELECT IDFase
                  FROM Fase
                  WHERE Nome = ”Scolastica”);
```

g. Calcolare il punteggio medio ottenuto durante la prima selezione, per ciascun istituto scolastico

```
SELECT i.Nome, AVG (p.Punteggio) AS 'Punteggio Medio'
FROM Partecipazione p, Atleta a, IstitutoScolastico i, Gara g, Fase f
WHERE i.IDIstituto = a.FKIstituto
AND a.IDAtleta = p.FKAtleta
AND g.IDGara = p.FKGara
AND f.IDFase = g.FKFase
AND f.Nome = "Scolastica"
GROUP BY i.Nome
ORDER BY i.Nome;
```

h. Stampare per ciascuna squadra il numero di "atleti" partecipanti e l'età media

```
SELECT s.Nome, COUNT(*) AS 'Numero Atleti Partecipanti', AVG (a.Eta) AS 'Eta Media'
FROM Squadra s, ComposizioneSquadra c, Atleta a
WHERE s.IDSquadra = c.FKSquadra
AND a.IDAtleta = c.FKAtleta
GROUP BY s.Nome
ORDER BY s.Nome;
```

i. Inserire le informazioni di un nuovo atleta

```
INSERT INTO Atleta
(Nome, Cognome, Eta, Nazionalita, FKIstituto)
VALUES
("Luca", "Boschi", 34, "Italiana", 1);
```