CH5 STUDENT PROJECTS

Creating and Manipulating a Relational Database for the Student Project

Read the sample project steps (The Art Gallery) for this chapter and apply the same techniques to the student project that you are developing.

For the relational schema you developed at the end of Chapter 4 for the project you have chosen, carry out the following steps to implement the design using a relational database management system such as Oracle, SQLServer, or MySQL:

- Step 5.1 Update the data dictionary and list of assumptions as needed. For each table, write the table name and write out the names, data types, and sizes of all the data items, and identify any constraints, using the conventions of the DBMS you will use for implementation.

donorCircle : circletype varchar, minAmount NUMBER, maxAmount NUMBER
Donor1: donorId NUMBER GENERATED ALWAYS AS IDENTITY,
PK, name VARCHAR NOT NULL, DOB DATE, address VARCHAR NOT NULL, affiliation VARCHAR, preferredPhone VARCHAR, circleType VARCHAR
Pledge1: pledgeId NUMBER GENERATED ALWAYS AS IDENTITY, donorId NUMBER(8,2) NOT NULL, pledgeAmount Number(10,2) CHECK (pledgeAmount > 0), pledgeDate DATE NOT NULL,
Payment1: Payment1, paymentId NUMBER GENERATED ALWAYS AS IDENTITY, pledgeId NUMBER(8,2) NOT NULL, amountPaid NUMBER(10,2), paymentType VARCHAR2(30), cardNumber VARCHAR2(16), paymentTiming VARCHAR2(30), CONSTRAINT Payment_paymentId_pk1 PRIMARY KEY (paymentID), CONSTRAINT Payment_pledgeId_fk1 Foreign KEY (pledgeId) REFERENCES Pledge1(pledgeId) ON DELETE SET NULL
Event1: eventId NUMBER GENERATED ALWAYS AS IDENTITY,, eventdonorId, name VARCHAR2(200) NOT NULL, eventDate DATE NOT NULL, eventCost NUMBER(10,2) NOT NULL, CONSTRAINT Event_eventId_pk1 PRIMARY KEY (eventId), CONSTRAINT Event_eventdonorId, name_uk1 UNIQUE (eventdonorId, name)
EventParticipation1: eventId NUMBER GENERATED ALWAYS AS IDENTITY,
,donorId NUMBER(8,2) NOT NULL, donationAtEvent NUMBER, isDonor CHAR(1) DEFAULT 'Y' CHECK (isDonor IN ('Y', 'N'))
MatchingGift1:matchingGiftId VARCHAR2(6) NOT NULL, donorId NUMBER(8,2) NOT NULL, employerdonorId, name VARCHAR2(50) NOT NULL,employerPhone VARCHAR2(13),amount Number(10,2) CHECK (amount >0),

- Step 5.2 Write and execute SQL statements to create all tables needed to implement the design.

```
CREATE TABLE DonorCircle1(
   circleType VARCHAR2(50),
   minAmount NUMBER(10,2),
   maxAmount NUMBER(10,2),
   CONSTRAINT DonorCircle1_pk PRIMARY KEY (circleType)
);

CREATE TABLE Donor1(
   donorId NUMBER GENERATED ALWAYS AS IDENTITY,
   name VARCHAR2(50) NOT NULL,
   DOB DATE,
   address VARCHAR2(100) NOT NULL,
   affiliation VARCHAR2(100),
   preferredPhone VARCHAR2(16),
   circleType VARCHAR2(50),
   CONSTRAINT Donor1_pk PRIMARY KEY (donorId),
   CONSTRAINT Donor1_circleType_fk FOREIGN KEY (circleType)
      REFERENCES DonorCircle1(circleType) ON DELETE SET NULL
);

CREATE TABLE Pledge1(
   pledgeId NUMBER GENERATED ALWAYS AS IDENTITY,
   donorId NUMBER(8) NOT NULL,
   pledgeAmount Number(10,2) CHECK (pledgeAmount > 0),
   pledgeDate DATE NOT NULL,
   CONSTRAINT Pledge1_pk PRIMARY KEY (pledgeID),
   CONSTRAINT Pledge1_donorId_fk FOREIGN KEY (donorId)
      REFERENCES Donor1(donorId) ON DELETE CASCADE
);

CREATE TABLE Payment1(
   paymentId NUMBER GENERATED ALWAYS AS IDENTITY,
   pledgeId NUMBER NOT NULL,
   amountPaid NUMBER(10,2) CHECK (amountPaid >= 0),
   paymentType VARCHAR2(30),
   cardNumber VARCHAR2(16),
   paymentTiming VARCHAR2(30),
   CONSTRAINT Payment1_pk PRIMARY KEY (paymentID),
   CONSTRAINT Payment1_pledgeId_fk FOREIGN KEY (pledgeId)
      REFERENCES Pledge1(pledgeId) ON DELETE CASCADE
);

CREATE TABLE Event1(
   eventId NUMBER GENERATED ALWAYS AS IDENTITY,
   eventName VARCHAR2(200) NOT NULL,
   eventDate DATE NOT NULL,
   eventCost NUMBER(10,2) CHECK (eventCost >= 0),
   CONSTRAINT Event1_pk PRIMARY KEY (eventId)
);
```

```
CREATE TABLE EventParticipation1(
   eventId NUMBER NOT NULL,
   donorId NUMBER NOT NULL,
   donationAtEvent NUMBER(10,2) CHECK(donationAtEvent > 0),
   isDonor CHAR(1) DEFAULT 'Y' CHECK (isDonor IN ('Y', 'N')),
   CONSTRAINT EventParticipation1_pk PRIMARY KEY (donorId, eventId),
   CONSTRAINT EventParticipation1_donorId_fk FOREIGN KEY (donorId)
      REFERENCES Donor1(donorId) ON DELETE CASCADE,
   CONSTRAINT EventParticipation1_eventId_fk FOREIGN KEY (eventId)
      REFERENCES Event1(eventId) ON DELETE CASCADE
);

CREATE TABLE MatchingGift1(
   matchingGiftId NUMBER GENERATED ALWAYS AS IDENTITY,
   donorId NUMBER NOT NULL,
   employerName VARCHAR2(50) NOT NULL,
   employerPhone VARCHAR2(13),
   amount Number(10,2) CHECK (amount > 0),
   CONSTRAINT MatchingGift1_pk PRIMARY KEY (matchingGiftId),
   CONSTRAINT MatchingGift1_donorId_fk FOREIGN KEY (donorId)
      REFERENCES Donor1(donorId) ON DELETE CASCADE
);
```

- Step 5.3 Insert at least five records in each table, preserving all constraints. Put in enough data to demonstrate how the database will function.

```
INSERT INTO DonorCircle VALUES ('Student');
INSERT INTO DonorCircle1 VALUES ('Student', 0, NULL);
INSERT INTO DonorCircle1 VALUES ('Alumni', 200, 487.99);
INSERT INTO DonorCircle1 VALUES ('Friend', 199, 395.99);

INSERT INTO Donor1(name, DOB, address, affiliation, preferredPhone, circleType)
VALUES('Liya', TO_DATE('2025-07-12','YYYY-MM-DD'),'oak st','Alumni','099-111','Alumni');
INSERT INTO Donor1(name, DOB, address, affiliation, preferredPhone, circleType)
VALUES('Lisa', TO_DATE('2005-09-27','YYYY-MM-DD'), 'brook st','Friend','009-234','Friend');
INSERT INTO Donor1(name, DOB, address, affiliation, preferredPhone, circleType)
VALUES('Mark', TO_DATE('2003-03-13','YYYY-MM-DD'),'pine ave','Student','785-675','Student');
INSERT INTO Donor1(name, DOB, address, affiliation, preferredPhone, circleType)
VALUES('Patricia', TO_DATE('2007/07/17','YYYY-MM-DD'),'river st','Alumni','256-756','Alumni');
INSERT INTO Donor1(name, DOB, address, affiliation, preferredPhone, circleType)
VALUES('Maria', TO_DATE('2007/08/19','YYYY-MM-DD'),'hills rd','Friend','023-465','Friend');


INSERT INTO Pledge1(donorId, pledgeAmount, pledgeDate)
VALUES(1, 200, TO_DATE('2025-07-12','YYYY-MM-DD'));
INSERT INTO Pledge1(donorId, pledgeAmount, pledgeDate)
VALUES(2, 400, TO_DATE('2025-06-12','YYYY-MM-DD'));
INSERT INTO Pledge1(donorId, pledgeAmount, pledgeDate)
VALUES(3, 600, TO_DATE('2025-05-12','YYYY-MM-DD'));
INSERT INTO Pledge1(donorId, pledgeAmount, pledgeDate)
```

```
VALUES(4, 800, TO_DATE('2025-03-12','YYYY-MM-DD'));
INSERT INTO Pledge1(donorId, pledgeAmount, pledgeDate)
VALUES(5, 900, TO_DATE('2025-01-12','YYYY-MM-DD'));

INSERT INTO Payment1 (pledgeId, amountPaid, paymentType, cardNumber, paymentTiming)
VALUES (1, 100.00, 'Credit', '123456789012', 'Monthly');
INSERT INTO Payment1 (pledgeId, amountPaid, paymentType, cardNumber, paymentTiming)
VALUES (2, 250.00, 'Cash', '434868338357', 'One-time');
INSERT INTO Payment1 (pledgeId, amountPaid, paymentType, cardNumber, paymentTiming)
VALUES (3, 150.00, 'Debit', '987654321012', 'One-time');
INSERT INTO Payment1 (pledgeId, amountPaid, paymentType, cardNumber, paymentTiming)
VALUES (4, 100.00, 'Credit', '555555555555', 'Monthly');
INSERT INTO Payment1 (pledgeId, amountPaid, paymentType, cardNumber, paymentTiming)
VALUES (5, 300.00, 'Cash', '376493902556', 'One-time');

INSERT INTO Event1 (eventName, eventDate, eventCost)
VALUES ('Charity Gala', TO_DATE('2025-06-20','YYYY-MM-DD'), 50.00);
INSERT INTO Event1 (eventName, eventDate, eventCost)
VALUES ('Fundraiser Dinner', TO_DATE('2025-07-10','YYYY-MM-DD'), 75.00);
INSERT INTO Event1 (eventName, eventDate, eventCost)
VALUES ('Volunteer Fair', TO_DATE('2025-08-01','YYYY-MM-DD'), 1.00);
INSERT INTO Event1 (eventName, eventDate, eventCost)
VALUES ('Alumni Meetup', TO_DATE('2025-09-12','YYYY-MM-DD'), 25.00);
INSERT INTO Event1 (eventName, eventDate, eventCost)
VALUES ('Donor Appreciation Night', TO_DATE('2025-10-05','YYYY-MM-DD'), 100.00);

INSERT INTO EventParticipation1 VALUES (1, 1, 50.00, 'Y');
INSERT INTO EventParticipation1 VALUES (2, 2, 100.00, 'Y');
INSERT INTO EventParticipation1 VALUES (3, 3, 1.00, 'N');
INSERT INTO EventParticipation1 VALUES (4, 4, 20.00, 'Y');
INSERT INTO EventParticipation1 VALUES (5, 5, 150.00, 'Y');

INSERT INTO MatchingGift1 (donorId, employerName, employerPhone, amount)
VALUES (2, 'FoodService', '555-7777', 500.00);
INSERT INTO MatchingGift1 (donorId, employerName, employerPhone, amount)
VALUES (3, 'MedicalCenter', '555-8888', 150.00);
INSERT INTO MatchingGift1 (donorId, employerName, employerPhone, amount)
VALUES (5, 'EductaionPurpose', '555-9999', 300.00);
INSERT INTO MatchingGift1 (donorId, employerName, employerPhone, amount)
VALUES (1, 'StudentAid', '555-1212', 200.00);
INSERT INTO MatchingGift1 (donorId, employerName, employerPhone, amount)
VALUES (4, 'FoodDrive', '555-3434', 100.00);
```

- Step 5.4 Write SQL statements that will process five non-routine requests for information from the database just created. For each, write the request in English, followed by the corresponding SQL command.
    - Total amount pais by every donor

        - SELECT d.name, SUM(p.amountPaid) AS totalPaid

          FROM Donor1 d

          JOIN Pledge1 p1 ON d.donorId = p1.donorID

          JOIN Payment1 p ON p1.pledgeId =p.pledgeId

          GROUP BY d.name;

- Pledged more than 500
    - SELECT d.donorId, d.name , p.pledgeAmount

      FROM Donor1 d

      JOIN Pledge1 p ON d.donorId = p.donorId

      WHERE p.pledgeAmount >500;
- Event and total amount donated during that event
    - SELECT e.eventName,

            SUM(ep.donationAtEvent) AS totalDonations

      FROM Event1 e

      JOIN EventParticipation1 ep ON e.eventId= ep.eventId

      GROUP BY e.eventName;
- Attend event but did not donate
    - SELECT d.donorId, d.name , e.eventName

      FROM EventParticipation1 ep

      JOIN Donor1 d ON ep.donorId =d.donorId

      JOIN Event1 e ON ep.eventId = e.eventId

      WHERE ep.isDonor = ‘N’
- Donors who made a matching gift contribution

- SELECT d.name AS donorName, m.employerName

  FROM Donor1 d

  JOIN MatchingGift1 m ON d.donorId = m.donorId;