Scenario 1: SALESPEOPLE / SALES / CUSTOMER

CREATE TABLE SALESPEOPLE (Sid INT PRIMARY KEY, Sname VARCHAR(50), City VARCHAR(50), Commission DECIMAL(5,2));

CREATE TABLE CUSTOMER (Cno INT PRIMARY KEY, Cname VARCHAR(50), City VARCHAR(50), Order_amount DECIMAL(10,2), Sid INT, FOREIGN KEY (Sid) REFERENCES SALESPEOPLE(Sid));

CREATE TABLE SALES (Sale_date DATE, Product_id INT, Qty INT, Price DECIMAL(10,2), Sid INT, FOREIGN KEY (Sid) REFERENCES SALESPEOPLE(Sid));

INSERT INTO SALESPEOPLE VALUES (101, 'Alice', 'New York', 12.5);

SELECT Sname, COUNT(*) FROM SALESPEOPLE JOIN SALES USING(Sid) GROUP BY Sname;

CREATE VIEW TopSalespeople AS SELECT Sname, SUM(Price * Qty) AS TotalSales FROM SALES JOIN SALESPEOPLE USING(Sid) GROUP BY Sname;

CREATE TRIGGER trg_check_commission BEFORE INSERT ON SALESPEOPLE FOR EACH ROW BEGIN IF NEW.Commission < 0 THEN SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Invalid commission'; END IF; END;

CREATE PROCEDURE GetCustomersBySalesperson(IN sid INT) BEGIN SELECT * FROM CUSTOMER WHERE Sid = sid; END;

DECLARE cur CURSOR FOR SELECT Cname FROM CUSTOMER;

MongoDB: db.salespeople.find({ city: 'New York' })

Scenario 2: ACCOUNT / TRANSACTION

CREATE TABLE ACCOUNT (Ac_No INT PRIMARY KEY, Ac_Name VARCHAR(50), Openbal DECIMAL(10,2), Opentype VARCHAR(10), Currentbal DECIMAL(10,2));

CREATE TABLE TRANSACTION (Tran_id INT PRIMARY KEY, Ac_No INT, Trans_Date DATE, Trans_Type VARCHAR(10), Trans_Amount DECIMAL(10,2), FOREIGN KEY (Ac_No) REFERENCES ACCOUNT(Ac_No));

INSERT INTO ACCOUNT VALUES (1001, 'John', 5000, 'Savings', 5000);

SELECT Ac_Name, SUM(Trans_Amount) FROM TRANSACTION JOIN ACCOUNT USING(Ac_No) GROUP

BY Ac_Name;

CREATE VIEW AccountBalance AS SELECT Ac_Name, Currentbal FROM ACCOUNT;

CREATE TRIGGER trg_balance_update AFTER INSERT ON TRANSACTION FOR EACH ROW BEGIN UPDATE ACCOUNT SET Currentbal = CASE WHEN NEW.Trans_Type = 'Deposit' THEN Currentbal + NEW.Trans_Amount ELSE Currentbal - NEW.Trans_Amount END WHERE Ac_No = NEW.Ac_No; END;

CREATE PROCEDURE GetTransactions(IN ac_no INT) BEGIN SELECT * FROM TRANSACTION WHERE Ac_No = ac_no; END;

MongoDB: db.transaction.find({ Trans_Type: 'Deposit' })

Scenario 3: STUDENT / MARKS

CREATE TABLE STUDENT (Student_id INT PRIMARY KEY, Student_Name VARCHAR(50), Course_Name VARCHAR(50));

CREATE TABLE MARKS (Student_id INT, Score INT, Grade CHAR(1), FOREIGN KEY (Student_id) REFERENCES STUDENT(Student_id));

INSERT INTO STUDENT VALUES (1, 'Tom', 'Math');

SELECT Student_Name, AVG(Score) FROM STUDENT JOIN MARKS USING(Student_id) GROUP BY Student_Name;

CREATE VIEW HighScorers AS SELECT Student_Name, Score FROM STUDENT JOIN MARKS USING(Student_id) WHERE Score > 90;

CREATE TRIGGER trg_score_check BEFORE INSERT ON MARKS FOR EACH ROW BEGIN IF NEW.Score < 0 THEN SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Invalid score'; END IF; END;

CREATE PROCEDURE GetGradesByStudent(IN sid INT) BEGIN SELECT * FROM MARKS WHERE Student_id = sid; END;

MongoDB: db.student.find({ Course_Name: 'Math' })

Scenario 4: INSTRUCTOR / COURSE_OFFERING

CREATE TABLE INSTRUCTOR (Instructor_id INT PRIMARY KEY, Instructor_Name VARCHAR(50), Department VARCHAR(50), Subject_Name VARCHAR(50));

CREATE TABLE COURSE_OFFERING (Course_No INT PRIMARY KEY, Year INT, Semester VARCHAR(10), Section VARCHAR(10), Instructor INT, FOREIGN KEY (Instructor) REFERENCES INSTRUCTOR(Instructor_id));

INSERT INTO INSTRUCTOR VALUES (1, 'Dr. Smith', 'CS', 'AI');

SELECT Instructor_Name, COUNT(*) FROM INSTRUCTOR JOIN COURSE_OFFERING ON INSTRUCTOR.Instructor_id = COURSE_OFFERING.Instructor GROUP BY Instructor_Name;

CREATE VIEW OfferedCourses AS SELECT * FROM COURSE_OFFERING WHERE Semester = 'Fall';

CREATE PROCEDURE GetCoursesByInstructor(IN iid INT) BEGIN SELECT * FROM COURSE_OFFERING WHERE Instructor = iid; END;

MongoDB: db.course_offering.find({ Semester: 'Fall' })

Scenario 5: EMPLOYEE / INCENTIVES

CREATE TABLE EMPLOYEE (Employee_id INT PRIMARY KEY, First_name VARCHAR(30), Last_name VARCHAR(30), Salary DECIMAL(10,2), Joining_date DATE, Department VARCHAR(50));

CREATE TABLE INCENTIVES (Employee_id INT, Incentive_date DATE, Incentive_amount DECIMAL(10,2), FOREIGN KEY (Employee_id) REFERENCES EMPLOYEE(Employee_id));

INSERT INTO EMPLOYEE VALUES (1, 'John', 'Doe', 60000, '2020-01-01', 'HR');

SELECT First_name, SUM(Incentive_amount) FROM EMPLOYEE JOIN INCENTIVES USING(Employee_id) GROUP BY First_name;

CREATE VIEW IncentiveSummary AS SELECT Employee_id, SUM(Incentive_amount) AS Total FROM INCENTIVES GROUP BY Employee_id;

CREATE PROCEDURE GetIncentives(IN emp_id INT) BEGIN SELECT * FROM INCENTIVES WHERE Employee_id = emp_id; END;

MongoDB: db.employee.find({ Department: 'HR' })