Experiment 2

```
CREATE TABLE Department (
    DeptID INT PRIMARY KEY,
    DeptName VARCHAR(50) NOT NULL
);

CREATE TABLE Employee (
    EmployeeID INT PRIMARY KEY,
    FirstName VARCHAR(50) NOT NULL,
    LastName VARCHAR(50) NOT NULL,
    Email VARCHAR(100) UNIQUE,
    Salary DECIMAL(10, 2) CHECK (Salary > 0),
    HireDate DATE DEFAULT CURRENT_DATE,
    DeptID INT REFERENCES Department(DeptID),
    PhoneNumber VARCHAR(15) NULL
);

INSERT INTO Department (DeptID, DeptName) VALUES (1, 'Engineering');
INSERT INTO Department (DeptID, DeptName) VALUES (2, 'Marketing');

INSERT INTO Employee (EmployeeID, FirstName, LastName, Email, Salary, DeptID)
VALUES (101, 'John', 'Doe', 'john.doe@example.com', 60000.00, 1);

INSERT INTO Employee (EmployeeID, FirstName, LastName, Email, Salary, DeptID,
PhoneNumber)
VALUES (102, 'Jane', 'Smith', 'jane.smith@example.com', 50000.00, 2, '555-1234');
```

Experiment 3

```
CREATE TABLE Department (
    DeptID INT PRIMARY KEY,
    DeptName VARCHAR(50) NOT NULL
);

CREATE TABLE Employee (
    EmployeeID INT PRIMARY KEY,
    FirstName VARCHAR(50) NOT NULL,
    LastName VARCHAR(50) NOT NULL,
    Email VARCHAR(100) UNIQUE,
    Salary DECIMAL(10, 2),
    HireDate DATE DEFAULT CURRENT_DATE,
    DeptID INT,
    FOREIGN KEY (DeptID) REFERENCES Department(DeptID)
);

INSERT INTO Department (DeptID, DeptName) VALUES (1, 'Engineering'), (2, 'Marketing');

INSERT INTO Employee (EmployeeID, FirstName, LastName, Email, Salary, DeptID)
VALUES (101, 'John', 'Doe', 'john.doe@example.com', 60000.00, 1),
       (102, 'Jane', 'Smith', 'jane.smith@example.com', 50000.00, 2);

SELECT CONCAT(FirstName, ' ', LastName) AS FullName, Salary * 1.10 AS AdjustedSalary
FROM Employee
WHERE Salary > 50000;

UPDATE Employee
SET Salary = Salary * 1.05
```

```
WHERE DeptID = 1;

DELETE FROM Employee
WHERE EmployeeID = 102;

SELECT E.FirstName, E.LastName, D.DeptName,
    CASE WHEN E.Salary > 55000 THEN 'High' ELSE 'Low' END AS SalaryLevel
FROM Employee E
JOIN Department D ON E.DeptID = D.DeptID;

SELECT FirstName, LastName FROM Employee WHERE DeptID = 1
UNION
SELECT FirstName, LastName FROM Employee WHERE DeptID = 2;
```

Experiment 4

```
CREATE TABLE Department (
   DeptID INT PRIMARY KEY,
   DeptName VARCHAR(50) NOT NULL
);

CREATE TABLE Employee (
   EmployeeID INT PRIMARY KEY,
   FirstName VARCHAR(50) NOT NULL,
   LastName VARCHAR(50) NOT NULL,
   Email VARCHAR(100) UNIQUE,
   Salary DECIMAL(10, 2),
   DeptID INT,
   FOREIGN KEY (DeptID) REFERENCES Department(DeptID)
);

CREATE VIEW EmployeeView AS
SELECT FirstName, LastName, Salary, DeptName
FROM Employee E
JOIN Department D ON E.DeptID = D.DeptID;

CREATE INDEX idx_email ON Employee (Email);

CREATE TABLE EmployeeWithSequence (
   EmployeeID INT AUTO_INCREMENT PRIMARY KEY,
   FirstName VARCHAR(50),
   LastName VARCHAR(50),
   Email VARCHAR(100) UNIQUE,
   Salary DECIMAL(10, 2),
    DeptID INT,
    FOREIGN KEY (DeptID) REFERENCES Department(DeptID)
);

CREATE VIEW EmpSynonym AS SELECT * FROM Employee;
```

Experiment 5

```
CREATE TABLE Department (
    DeptID INT PRIMARY KEY,
    DeptName VARCHAR(50) NOT NULL
);

CREATE TABLE Employee (
    EmployeeID INT PRIMARY KEY,
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    Salary DECIMAL(10, 2),
    DeptID INT,
    FOREIGN KEY (DeptID) REFERENCES Department(DeptID)
);

INSERT INTO Department (DeptID, DeptName) VALUES (1, 'Engineering'), (2, 'Marketing');

INSERT INTO Employee (EmployeeID, FirstName, LastName, Salary, DeptID)
VALUES (101, 'John', 'Doe', 60000, 1),
       (102, 'Jane', 'Smith', 50000, 2),
       (103, 'Michael', 'Brown', 70000, 1),
       (104, 'Alice', 'Johnson', 45000, 2);

SELECT COUNT(*) AS TotalEmployees FROM Employee;

SELECT SUM(Salary) AS TotalSalary FROM Employee;

SELECT MAX(Salary) AS HighestSalary FROM Employee;
SELECT MIN(Salary) AS LowestSalary FROM Employee;

SELECT AVG(Salary) AS AverageSalary FROM Employee;

START TRANSACTION;

INSERT INTO Employee (EmployeeID, FirstName, LastName, Salary, DeptID)
VALUES (105, 'Chris', 'Evans', 65000, 1);

SAVEPOINT SavePoint1;

INSERT INTO Employee (EmployeeID, FirstName, LastName, Salary, DeptID)
VALUES (106, 'Emily', 'Blunt', 75000, 2);

ROLLBACK TO SavePoint1;

COMMIT;

SELECT * FROM Employee;
```

Experiment 6

```
CREATE TABLE Department (
    DeptID INT PRIMARY KEY,
    DeptName VARCHAR(50) NOT NULL
);

CREATE TABLE Employee (
    EmployeeID INT PRIMARY KEY,
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    Salary DECIMAL(10, 2),
    DeptID INT,
    FOREIGN KEY (DeptID) REFERENCES Department(DeptID)
);

INSERT INTO Department (DeptID, DeptName) VALUES (1, 'Engineering'), (2, 'Marketing'), (3,
'Sales');

INSERT INTO Employee (EmployeeID, FirstName, LastName, Salary, DeptID)
VALUES (101, 'John', 'Doe', 60000, 1),
       (102, 'Jane', 'Smith', 50000, 2),
       (103, 'Michael', 'Brown', 70000, 1),
       (104, 'Alice', 'Johnson', 45000, 3),
       (105, 'Chris', 'Evans', 65000, NULL);

SELECT E.FirstName, E.LastName, D.DeptName
FROM Employee E
INNER JOIN Department D ON E.DeptID = D.DeptID;
```

```sql
SELECT E.FirstName, E.LastName, D.DeptName

FROM Employee E

LEFT JOIN Department D ON E.DeptID = D.DeptID;


SELECT E.FirstName, E.LastName, D.DeptName

FROM Employee E

RIGHT JOIN Department D ON E.DeptID = D.DeptID;


SELECT E.FirstName, E.LastName, D.DeptName

FROM Employee E

LEFT JOIN Department D ON E.DeptID = D.DeptID

UNION

SELECT E.FirstName, E.LastName, D.DeptName

FROM Employee E

RIGHT JOIN Department D ON E.DeptID = D.DeptID;


SELECT FirstName, LastName

FROM Employee

WHERE Salary > (SELECT AVG(Salary) FROM Employee);


SELECT DeptName

FROM Department

WHERE DeptID IN (SELECT DeptID FROM Employee WHERE Salary > 60000);
```

Experiment 7

```
DECLARE
    TYPE ScoreArray IS TABLE OF NUMBER INDEX BY PLS_INTEGER;
    scores ScoreArray;
    grades ScoreArray;
    total_students PLS_INTEGER := 10;

    PROCEDURE CalculateGrade(score NUMBER, grade OUT CHAR) IS
    BEGIN
        IF score >= 90 THEN
            grade := 'A';
        ELSIF score >= 80 THEN
            grade := 'B';
        ELSIF score >= 70 THEN
            grade := 'C';
        ELSIF score >= 60 THEN
            grade := 'D';
        ELSE
            grade := 'F';
        END IF;
    END;

BEGIN
    scores(1) := 85;
    scores(2) := 92;
    scores(3) := 76;
    scores(4) := 64;
    scores(5) := 58;
    scores(6) := 89;
    scores(7) := 73;

    scores(8) := 91;

    scores(9) := 87;

    scores(10) := 77;


    FOR i IN 1..total_students LOOP

        CalculateGrade(scores(i), grades(i));

    END LOOP;


    DBMS_OUTPUT.PUT_LINE('Student Scores and Grades:');

    FOR i IN 1..total_students LOOP

        DBMS_OUTPUT.PUT_LINE('Score: ' || scores(i) || ', Grade: ' || grades(i));

    END LOOP;


END;
```

Experiment 7

```
DECLARE
    total_count NUMBER;

    CURSOR emp_cursor IS
        SELECT EmployeeID, FirstName, LastName, Salary FROM Employee;

    emp_record emp_cursor%ROWTYPE;

    CURSOR emp_salary_cursor(p_min_salary NUMBER) IS
        SELECT EmployeeID, FirstName, LastName FROM Employee WHERE Salary >
p_min_salary;

BEGIN
    SELECT COUNT(*) INTO total_count FROM Employee;
    DBMS_OUTPUT.PUT_LINE('Total number of employees: ' || total_count);

    OPEN emp_cursor;
    LOOP
        FETCH emp_cursor INTO emp_record;
        EXIT WHEN emp_cursor%NOTFOUND;
        DBMS_OUTPUT.PUT_LINE('Employee ID: ' || emp_record.EmployeeID || ', Name: ' ||
emp_record.FirstName || ' ' || emp_record.LastName || ', Salary: ' || emp_record.Salary);
    END LOOP;
    CLOSE emp_cursor;

    DBMS_OUTPUT.PUT_LINE('Employees with salary greater than 60000:');
    FOR emp IN emp_salary_cursor(60000) LOOP
        DBMS_OUTPUT.PUT_LINE('Employee ID: ' || emp.EmployeeID || ', Name: ' ||
emp.FirstName || ' ' || emp.LastName);
```

```
    END LOOP;


    DBMS_OUTPUT.PUT_LINE('All Employees:');
    FOR emp IN (SELECT EmployeeID, FirstName, LastName FROM Employee) LOOP
        DBMS_OUTPUT.PUT_LINE('Employee ID: ' || emp.EmployeeID || ', Name: ' ||
emp.FirstName || ' ' || emp.LastName);
    END LOOP;


END;
/
```

Experiment 9

```
db.employees.insertMany([
  { empId: 1, name: 'Clark', dept: 'Sales' },
  { empId: 2, name: 'Dave', dept: 'Accounting' },
  { empId: 3, name: 'Ava', dept: 'Sales' },
  { empId: 4, name: 'Ella', dept: 'Marketing' },
  { empId: 5, name: 'James', dept: 'Sales' }
]);


db.employees.find({ dept: 'Sales' });


db.employees.findOne({ empId: 2 });


db.employees.updateOne(
  { empId: 3 },
  { $set: { dept: 'HR' } }
);


db.employees.deleteOne({ empId: 1 });


db.employees.find({});
```