DBMS LAB - ALL EXPERIMENTS (ORACLE)

Below are all experiments E01–E11 with full code and sample data, compatible with Oracle Database.

E01 — Library Tables

```
CREATE TABLE Book (
BookID NUMBER PRIMARY KEY,
Title VARCHAR2(100),
Author VARCHAR2(60),
Price NUMBER(6,2)
);

CREATE TABLE Member (
MemberID NUMBER PRIMARY KEY,
Name VARCHAR2(60),
Email VARCHAR2(80),
Phone VARCHAR2(15)
);

CREATE TABLE Loan (
LoanID NUMBER PRIMARY KEY,
BookID NUMBER REFERENCES Book(BookID),
MemberID NUMBER REFERENCES Member(MemberID),
LoanDate DATE,
DueDate DATE
);
```

Sample Inserts:

```
INSERT INTO Book VALUES (1, 'DBMS Concepts', 'Korth', 550);
INSERT INTO Member VALUES (101, 'Aditya Patil', 'adi@mail.com', '9876543210');
```

```
INSERT INTO Loan VALUES (1001, 1, 101, SYSDATE, SYSDATE+14);
```

E02 — DDL

```
CREATE TABLE Employee (
EmpID NUMBER PRIMARY KEY,
EmpName VARCHAR2(50),
Dept VARCHAR2(50),
Salary NUMBER(10,2)
);
ALTER TABLE Employee ADD HireDate DATE;
ALTER TABLE Employee MODIFY EmpName VARCHAR2(80);
ALTER TABLE Employee DROP COLUMN HireDate;
TRUNCATE TABLE Employee;
DROP TABLE Employee;
```

E03 — DML

```
CREATE TABLE Students (
Roll NUMBER PRIMARY KEY,
Name VARCHAR2(50),
Marks NUMBER(3)
);

INSERT INTO Students VALUES (1, 'Amit', 85);
UPDATE Students SET Marks = 95 WHERE Roll = 3;
DELETE FROM Students WHERE Roll = 1;
SELECT * FROM Students;
```

E04 — Functions

```
SELECT ABS(-10), ROUND(123.456,2), CEIL(12.1) FROM dual;
SELECT UPPER('hello'), SUBSTR('DATABASE',1,4) FROM dual;
SELECT SYSDATE, ADD_MONTHS(SYSDATE,1) FROM dual;
SELECT TO_CHAR(SYSDATE,'DD-MON-YYYY') FROM dual;
```

E05 — GROUP BY, HAVING, INDEX

```
SELECT Dept, COUNT(*), AVG(Salary)
FROM Employee
GROUP BY Dept
HAVING AVG(Salary) > 35000;

CREATE INDEX idx_emp_dept ON Employee(Dept);
```

E06 — Set Operations & Joins

```
SELECT * FROM A UNION SELECT * FROM B;
SELECT * FROM A INTERSECT SELECT * FROM B;
SELECT * FROM A MINUS SELECT * FROM B;

SELECT e.Name, d.DeptName
FROM Employee e
JOIN Dept d ON e.Dept = d.DeptID;
```

E07 — Subqueries & Views

```
SELECT Name FROM Employee
WHERE Salary = (SELECT MAX(Salary) FROM Employee);

CREATE OR REPLACE VIEW vw_highsal AS
SELECT Name, Salary FROM Employee WHERE Salary > 40000;
```

E08 — Transactions

```
UPDATE Account SET Balance = Balance - 500 WHERE AccNo = 101;
SAVEPOINT sp1;
UPDATE Account SET Balance = Balance + 500 WHERE AccNo = 102;
ROLLBACK TO sp1;
COMMIT;
```

E09 — Procedure & Function

```
CREATE OR REPLACE PROCEDURE give_bonus(p_emp NUMBER, p_bonus NUMBER) IS
BEGIN
UPDATE Employee SET Salary = Salary + p_bonus WHERE EmpID = p_emp;
END;
/

CREATE OR REPLACE FUNCTION yearly_salary(p_emp NUMBER)
RETURN NUMBER IS
sal NUMBER;
BEGIN
SELECT Salary INTO sal FROM Employee WHERE EmpID = p_emp;
RETURN sal * 12;
END;
/
```

E10 — Trigger & Cursor

```
CREATE OR REPLACE TRIGGER trg_insert_audit
AFTER INSERT ON Employee
FOR EACH ROW
BEGIN
```

```
INSERT INTO Emp_Audit VALUES (:NEW.EmpID, 'INSERT', SYSDATE);
END;
/

DECLARE
CURSOR cur IS SELECT Name, Salary FROM Employee;
v_name Employee.Name%TYPE;
v_sal Employee.Salary%TYPE;
BEGIN
OPEN cur;
LOOP
FETCH cur INTO v_name, v_sal;
EXIT WHEN cur%NOTFOUND;
DBMS_OUTPUT.PUT_LINE(v_name || ' earns ' || v_sal);
END LOOP;
CLOSE cur;
END;
/
```

E11 — JDBC Code

Java JDBC code included in previous messages.