**BASIC:**

CREATE TABLE Employee

(

Id int,

Name varchar(30),

City varchar(30),

Dob date

)

INSERT INTO Employee(Id, Name, City, Dob) values('1', 'A', 'Mumbai','19-JUNE-2023')

SELECT \* FROM Employee

**BASIC CRUD:**

CREATE TABLE Student (

ID int,

Age int,

CGPA float,

Name varchar(255),

Gender varchar(255)

)

INSERT INTO Student(ID, Age, CGPA, Name, Gender) VALUES ('1','20','9.9','Disha','F')

INSERT INTO Student(ID, Age, CGPA, Name, Gender) VALUES ('2','19','8.8','A','M')

INSERT INTO Student(ID, Age, CGPA, Name, Gender) VALUES ('3','20','7.7','B','F')

INSERT INTO Student(ID, Age, CGPA, Name, Gender) VALUES ('4','19','6.6','C','M')

INSERT INTO Student(ID, Age, CGPA, Name, Gender) VALUES ('5','20','5.5','D','F')

SELECT \* FROM Student

DELETE FROM Student WHERE Gender='M'

DELETE FROM Student WHERE ID=5

UPDATE Student SET CGPA = 7.9 WHERE ID = 3

**DDL, DML, OPERATORS, DATE:**

CREATE TABLE Employee (

id int PRIMARY KEY,

name varchar(25),

phone int

)

INSERT INTO Employee (id, name, phone) VALUES (1, 'A', 12345)

INSERT INTO Employee (id, name, phone) VALUES (2, 'B', 67890)

INSERT INTO Employee (id, name, phone) VALUES (3, 'C', 98765)

INSERT INTO Employee (id, name, phone) VALUES (4, 'D', 43210)

SELECT \* FROM Employee

ALTER TABLE Employee ADD Salary float

ALTER TABLE Employee ADD Designation varchar(25)

ALTER TABLE Employee ADD DOJ DATE

ALTER TABLE Employee ADD Email varchar(25)

INSERT INTO Employee (id, name, phone, Salary, Designation, DOJ, Email) VALUES

(5, 'E', 24680, 99999.99, 'HOD',DATE '2000-12-06', 'eeeee@gmail.com')

UPDATE Employee SET Salary=39998, Designation='CEO', DOJ= DATE '2003-06-07', Email='aaaa@gmail.com' WHERE id=1

UPDATE Employee SET Salary=43252, Designation='CFO', DOJ= DATE '2004-03-09', Email='bbbb@gmail.com' WHERE id=2

UPDATE Employee SET Salary=94675, Designation='HOI', DOJ= DATE '2001-02-04', Email='cccc@gmail.com' WHERE id=3

UPDATE Employee SET Salary=76548, Designation='MANAGER', DOJ= DATE '2008-01-01', Email='dddd@gmail.com' WHERE id=4

SELECT \* FROM Employee WHERE Salary>50000

SELECT \* FROM Employee ORDER BY id DESC

ALTER TABLE Employee DROP COLUMN Designation

ALTER TABLE Employee RENAME COLUMN Name to fName

SELECT \* FROM EMPLOYEE WHERE Salary = (SELECT MIN(Salary) FROM EMPLOYEE)

SELECT \* FROM EMPLOYEE WHERE Salary = (SELECT MAX(Salary) FROM EMPLOYEE)

SELECT AVG(Salary) FROM EMPLOYEE WHERE Designation='Manager'

SELECT ID, Name, Salary FROM EMPLOYEE WHERE Designation='Programmer'

SELECT \* FROM EMPLOYEE WHERE SALARY>(SELECT MAX(Salary) FROM EMPLOYEE WHERE Designation='Programmer')

SELECT \* FROM EMPLOYEE WHERE SALARY>(SELECT MAX(Salary) FROM EMPLOYEE WHERE Designation='Programmer')

SELECT \* FROM EMPLOYEE WHERE Designation='Programmer' OR Designation='DS'

SELECT \* FROM EMPLOYEE WHERE Department='IT' OR Department='HR'

SELECT ID, Name FROM EMPLOYEE WHERE Salary>10000 AND Salary<20000

SELECT \* FROM EMPLOYEE WHERE Salary = (SELECT MAX(Salary) FROM EMPLOYEE) AND Designation='Programmer' AND Department='CSE'

SELECT \* FROM EMPLOYEE WHERE Salary = (SELECT MAX(Salary) FROM EMPLOYEE WHERE Designation='Programmer' AND Department='CSE')

SELECT COUNT(\*) FROM EMPLOYEE WHERE Department='CSE'

SELECT ID, Name FROM EMPLOYEE WHERE Salary BETWEEN 10000 AND 20000

SELECT COUNT(DISTINCT 'CSE') FROM EMPLOYEE

SELECT COUNT(DISTINCT 'CSE'), COUNT(DISTINCT 'IT') FROM EMPLOYEE

SELECT COUNT(Designation), Designation as Designation from EMPLOYEE WHERE Designation='P' OR Designation='DS' OR Designation='M' GROUP BY Designation

ALTER TABLE Employee RENAME TO HRfaculty;

SELECT \* FROM HRfaculty;

**ALTER, LIKE, AS:**

CREATE TABLE BOOK (

ACC\_NO int PRIMARY KEY,

TITLE varchar(25) UNIQUE,

CATEGORY varchar(25) DEFAULT 'Book',

PRICE float NOT NULL,

YEAR\_OF\_PUB varchar(10),

PAGE\_COUNT int

)

CREATE TABLE PUBLISHER (

PUB\_ID int PRIMARY KEY,

NAME varchar(25) NOT NULL,

CITY varchar(10),

STATE\_ varchar(10) DEFAULT '-',

PH\_NO int UNIQUE

)

SELECT \* FROM PUBLISHER

SELECT \* FROM BOOK

SELECT \* FROM BOOK ORDER BY ACC\_NO ASC

SELECT \* FROM PUBLISHER ORDER BY PUB\_ID ASC

INSERT INTO BOOK VALUES ('1', 'ABC', DEFAULT, 2000, 2003, 150)

INSERT INTO BOOK VALUES ('2', 'DEF', 'SCI-FI', 2000, 2003, 150)

INSERT INTO BOOK VALUES ('3', 'GHI', 'SPACE WARS', 2000, 2005, 200)

INSERT INTO BOOK VALUES ('4', 'JKL', 'FAN-FIC', 2000, 2006, 350)

INSERT INTO BOOK VALUES ('5', 'MNO', 'SCI-FI', 2000, 2007, 770)

INSERT INTO BOOK VALUES ('6', 'PQR','SPACE WARS', 2008, 2003, 800)

INSERT INTO BOOK VALUES ('7', 'STU', DEFAULT, 2000, 2009, 1500)

INSERT INTO BOOK VALUES ('8', 'VW', 'SCI-FI', 2000, 2010, 120)

INSERT INTO BOOK VALUES ('9', 'XY', 'FAN-FIC', 2000, 2011, 190)

INSERT INTO BOOK VALUES ('10', 'Z', 'SCI-FI', 2000, 2012, 200)

INSERT INTO PUBLISHER VALUES ('1', 'Q', 'W', DEFAULT, 12345)

INSERT INTO PUBLISHER VALUES ('2', 'W', 'S', DEFAULT, 6789)

INSERT INTO PUBLISHER VALUES ('3', 'E', 'D', DEFAULT, 1011)

INSERT INTO PUBLISHER VALUES ('4', 'R', 'F', DEFAULT, 1213)

INSERT INTO PUBLISHER VALUES ('5', 'T', 'G', DEFAULT, 1415)

INSERT INTO PUBLISHER VALUES ('6', 'Y', 'H', DEFAULT, 1617)

INSERT INTO PUBLISHER VALUES ('7', 'U', 'I', DEFAULT, 1819)

INSERT INTO PUBLISHER VALUES ('8', 'I', 'N', DEFAULT, 2021)

INSERT INTO PUBLISHER VALUES ('9', 'O', 'M', DEFAULT, 2223)

INSERT INTO PUBLISHER VALUES ('10', 'P', 'X', DEFAULT, 2425)

INSERT INTO PUBLISHER VALUES ('11', 'A', 'Z', DEFAULT, 2627)

INSERT INTO BOOK VALUES ('1', 'ABC', DEFAULT, 299.99, 2003, 150)

INSERT INTO BOOK VALUES ('2', 'DEF', 'Novel', 2199.99, 2004, 1000)

INSERT INTO BOOK VALUES ('3', 'GHI', 'Comic', 699.09, 2005, 70)

INSERT INTO BOOK VALUES ('4', 'JKL', 'Fiction', 99.29, 1999, 275)

INSERT INTO BOOK VALUES ('5', 'MNO', 'Non-Fiction', 5009.99, 2014, 3630)

INSERT INTO BOOK VALUES ('6', 'PQR', DEFAULT, 399.99, 2023, 880)

INSERT INTO BOOK VALUES ('7', 'STU', 'Textbook', 1499.99, 2000, 90)

INSERT INTO BOOK VALUES ('8', 'VWX', 'Novel', 7008.59, 1989, 5000)

INSERT INTO BOOK VALUES ('9', 'YZA', 'Textbook', 70.05, 2001, 120)

INSERT INTO BOOK VALUES ('10', 'BCD', 'Comic', 600.00, 1882, 15)

INSERT INTO PUBLISHER VALUES (123, 'ZYX', 'Pune', 'Maharashtra', 1111)

INSERT INTO PUBLISHER VALUES (456, 'WVU', 'Sambhajinagar', 'Maharashtra', 2222)

INSERT INTO PUBLISHER VALUES (789, 'TSR', 'Mumbai', 'Mharashtra', 3333)

INSERT INTO PUBLISHER VALUES (1011, 'QPO', 'Delhi', DEFAULT, 4444)

INSERT INTO PUBLISHER VALUES (1213, 'NML', 'Chennai', 'Tamil Nadu', 5555)

INSERT INTO PUBLISHER VALUES (1415, 'KJI', 'Hyderabad', DEFAULT, 6666)

INSERT INTO PUBLISHER VALUES (1617, 'HGF', 'Banglore', 'Karnataka', 7777)

INSERT INTO PUBLISHER VALUES (1819, 'EDC', 'Gandhinagar', 'Gujrat', 8888)

INSERT INTO PUBLISHER VALUES (2021, 'BAZ', 'Nagpur', 'Maharashtra', 9999)

INSERT INTO PUBLISHER VALUES (2223, 'YXV', 'Ahmedabad', 'Gujrat', 1010)

ALTER TABLE PUBLISHER ADD EMAIL\_ID varchar(15)

SELECT NAME AS PUBLISHER\_NAME, CITY, PH\_NO FROM PUBLISHER WHERE STATE\_='Mh'

SELECT ACC\_NO, TITLE, PRICE FROM BOOK WHERE PAGE\_COUNT>600

SELECT TITLE AS BOOK\_TITLE, PRICE FROM BOOK WHERE CATEGORY='Comic'

UPDATE BOOK SET PRICE = (PRICE + 0.1\*PRICE) WHERE CATEGORY='Novel'

SELECT TITLE AS BOOK\_TITLE, PRICE FROM BOOK WHERE CATEGORY='Comic'

SELECT \* FROM BOOK WHERE TITLE LIKE 'D%'

SELECT \* FROM PUBLISHER WHERE NAME LIKE 'A%

SELECT \* FROM PUBLISHER WHERE NAME LIKE 'Y%'

SELECT \* FROM PUBLISHER WHERE NAME LIKE 'C%'

SELECT \* FROM PUBLISHER WHERE NAME LIKE 'D%' AND STATE\_='GJ'

DELETE FROM BOOK WHERE PAGE\_COUNT<100

ALTER TABLE BOOK DROP COLUMN YEAR\_OF\_PUB

SELECT \* FROM BOOK WHERE TITLE NOT LIKE 'D%'

ALTER TABLE PUBLISHER DROP COLUMN EMAIL\_ID

ALTER TABLE PUBLISHER ADD EMAIL\_ID int

ALTER TABLE PUBLISHER MODIFY EMAIL\_ID varchar(10)

SELECT \* FROM BOOK WHERE TITLE LIKE '%o'

SELECT \* FROM PUBLISHER WHERE NAME LIKE '%M%'

**CONSTRAINTS, AGGREGATE FUNCTIONS:**

CREATE TABLE STUDENT (

Std\_ID int PRIMARY KEY,

Std\_Name varchar(10) NOT NULL,

DOB timestamp,

Ass\_ID int UNIQUE,

Due\_Date DATE DEFAULT DATE '2023-06-15',

Sub\_Date timestamp,

Adm\_Date DATE,

Batch int DEFAULT '2021'

)

SELECT \* FROM STUDENT

SELECT \* FROM STUDENT ORDER BY Std\_ID ASC

INSERT INTO STUDENT VALUES (1, 'ABC', DATE '2003-03-04', 123, DEFAULT, DATE '2023-04-20',DATE '2021-08-25', DEFAULT)

INSERT INTO STUDENT VALUES (2, 'DEF',timestamp '2004-02-02 08:04:11', 456, DEFAULT, CURRENT\_TIMESTAMP, DATE '2021-08-15', DEFAULT)

INSERT INTO STUDENT VALUES (3, 'GHI',timestamp '2002-01-06 13:07:15', 789, DEFAULT, CURRENT\_TIMESTAMP, DATE '2021-07-25', DEFAULT)

INSERT INTO STUDENT VALUES (4, 'JKL',timestamp '2003-04-08 12:06:00', 101, DATE '2023-06-25', DATE '2023-04-20',DATE '2021-06-20', '2022')  
INSERT INTO STUDENT VALUES (5, 'MNO',timestamp '2004-05-10 01:00:08', 121, DATE '2023-05-25', CURRENT\_TIMESTAMP, DATE '2021-08-02', DEFAULT)

INSERT INTO STUDENT VALUES (6, 'PQR',timestamp '2002-06-12 02:18:17', 131, DEFAULT, DATE '2023-04-20',DATE '2021-08-17', '2022')

INSERT INTO STUDENT VALUES (7, 'STU',timestamp '2001-07-14 09:04:59', 141, DATE '2023-06-13', CURRENT\_TIMESTAMP, DATE '2021-08-19', DEFAULT)

INSERT INTO STUDENT VALUES (8, 'VWX',timestamp '2004-08-08 10:00:44', 151, DEFAULT, DATE '2023-04-20',DATE '2021-07-20', '2020')

INSERT INTO STUDENT VALUES (9, 'YZA',timestamp '2003-09-04 08:18:33', 161, DEFAULT, DATE '2023-04-20',DATE '2021-08-19', '2022')

INSERT INTO STUDENT VALUES (10, 'BCD',timestamp '2002-10-06 18:08:51', 171, DATE '2023-03-15', CURRENT\_TIMESTAMP,DATE '2021-07-30', DEFAULT)  
INSERT INTO STUDENT VALUES (11, 'EFG',timestamp '2001-11-05 23:09:00', 181, DEFAULT, DATE '2023-04-20', DATE '2021-08-18', DEFAULT)

INSERT INTO STUDENT VALUES (12, 'HIJ',timestamp '2003-12-03 16:16:01', 191, DATE '2023-06-22', CURRENT\_TIMESTAMP ,DATE '2021-08-19', '2023')

INSERT INTO STUDENT VALUES (13, 'KLM',timestamp '2003-01-01 22:45:17', 201, DATE '2023-06-21', DATE '2023-04-20',DATE '2021-07-20', DEFAULT)  
INSERT INTO STUDENT VALUES (14, 'NOP',timestamp '2004-03-07 15:02:10', 301, DATE '2023-06-30', CURRENT\_TIMESTAMP,DATE '2021-09-12','2022')  
INSERT INTO STUDENT VALUES (15, 'QRS',timestamp '2002-02-05 03:01:11', 401, DEFAULT, DATE '2023-04-20',DATE '2021-08-21', DEFAULT)

UPDATE STUDENT SET Due\_Date=Due\_Date+5

SELECT COUNT(Due\_Date) FROM STUDENT WHERE Due\_Date= DATE '2023-06-15

SELECT TO\_CHAR(Adm\_Date, 'DD-MM-YYYY') AS NEW\_FORMAT FROM STUDENT

SELECT TO\_CHAR(Adm\_Date, 'DD-MM-YYYY') AS NEW\_FORMAT FROM STUDENT

SELECT Std\_ID, Std\_Name, EXTRACT(MONTH FROM Adm\_Date) AS ADMISSION\_MONTH, EXTRACT(YEAR FROM Adm\_Date) AS ADMISSION\_YEAR FROM STUDENT

ALTER TABLE STUDENT ADD Department varchar(20)

UPDATE STUDENT SET Department='CSE'

SELECT Adm\_Date, COUNT(\*) AS Number\_Of\_Students FROM STUDENT GROUP BY Adm\_Date

SELECT Adm\_Date, COUNT(\*) AS Number\_Of\_Students FROM STUDENT GROUP BY Adm\_Date

**DISTINCT, GROUP BY, HAVING, EXTRACT, TO\_CHAR:**

CREATE TABLE SALES (

Cust\_ID int PRIMARY KEY,

Cust\_Name varchar(10) NOT NULL,

Salesman\_ID int,

City varchar(20),

Order\_Date DATE,

Order\_ID int UNIQUE,

Purchase\_Amt float DEFAULT 500.00

)

SELECT \* FROM SALES ORDER BY Cust\_ID ASC

INSERT INTO SALES VALUES (1, 'A', 123, 'Pune', DATE '2023-08-07', 1111, DEFAULT)  
INSERT INTO SALES VALUES (2, 'B', 456, 'Mumbai', DATE '2023-07-06', 2222, 225.9)  
INSERT INTO SALES VALUES (3, 'C', 789, 'Delhi', DATE '2023-06-05', 3333, 144)

INSERT INTO SALES VALUES (4, 'D', 123, 'Pune', DATE '2023-05-04', 4444, 1022.05)

INSERT INTO SALES VALUES (5, 'E', 1011, 'Mumbai', DATE '2023-03-02', 5555, DEFAULT)

INSERT INTO SALES VALUES (6, 'F', 1213, 'Delhi', DATE '2023-01-10', 6666, 678)

INSERT INTO SALES VALUES (7, 'A', 123, 'Pune', DATE '2022-12-23', 7777, 991.22)

INSERT INTO SALES VALUES (8, 'B', 456, 'Mumbai', CURRENT\_DATE, 8888, 10000)  
INSERT INTO SALES VALUES (9, 'A', 1415, 'Delhi', CURRENT\_DATE, 9999, 2002.99)  
INSERT INTO SALES VALUES (10, 'F', 123, 'Pune', CURRENT\_DATE, 1010, DEFAULT)

SELECT DISTINCT Salesman\_ID FROM SALES

SELECT DISTINCT Cust\_Name FROM SALES

SELECT DISTINCT Salesman\_ID AS DISTINCT\_SALESMAN FROM SALES

SELECT DISTINCT Cust\_Name AS DISTINCT\_CUSTOMERS FROM SALES

SELECT DISTINCT Count(Salesman\_ID) AS DISTINCT\_SALESMAN FROM SALES

SELECT DISTINCT Count(Cust\_Name) AS DISTINCT\_CUSTOMERS FROM SALES

SELECT DISTINCT(Count(Cust\_Name)) AS DISTINCT\_CUSTOMERS FROM SALES

SELECT DISTINCT Count(Salesman\_ID) AS DISTINCT\_SALESMAN FROM SALES

SELECT Count(DISTINCT Salesman\_ID) AS DISTINCT\_SALESMAN FROM SALES

SELECT Count(DISTINCT Cust\_Name) AS DISTINCT\_CUSTOMERS FROM SALES

SELECT DISTINCT Order\_Date AS DISTINCT\_ORDER\_DATES FROM SALES

SELECT Cust\_Name, Order\_Date, Max(Purchase\_Amt) FROM SALES GROUP BY Cust\_Name, Order\_Date

SELECT Salesman\_ID, Order\_Date, Min(Purchase\_Amt) FROM SALES WHERE Order\_Date = DATE '2023-08-07' GROUP BY Salesman\_ID, Order\_Date

SELECT DISTINCT Cust\_Name, Order\_Date, Max(Purchase\_Amt) FROM SALES GROUP BY Cust\_Name, Order\_Date

SELECT Cust\_Name, Order\_Date, Max(Purchase\_Amt) FROM SALES GROUP BY Cust\_Name, Order\_Date HAVING Max(Purchase\_Amt)>2000

SELECT Cust\_Name, Order\_Date, Max(Purchase\_Amt) FROM SALES GROUP BY Cust\_Name, Order\_Date HAVING Max(Purchase\_Amt) IN(2002.99, 10000)



SELECT Cust\_Name, Order\_Date, Max(Purchase\_Amt) FROM SALES GROUP BY Cust\_Name, Order\_Date HAVING Max(Purchase\_Amt) IN(2002.99, 10000)

SELECT Cust\_Name, Cust\_ID, Max(Purchase\_Amt) FROM SALES WHERE Cust\_ID BETWEEN 1000 AND 1500 GROUP BY Cust\_Name, Cust\_ID HAVING Max(Purchase\_Amt)>2000

SELECT Cust\_Name, Cust\_ID, Max(Purchase\_Amt) FROM SALES WHERE Cust\_ID BETWEEN 100 AND 1000 GROUP BY Cust\_Name, Cust\_ID HAVING Max(Purchase\_Amt)>2000

SELECT Cust\_Name, Cust\_ID, Max(Purchase\_Amt) FROM SALES WHERE Cust\_ID BETWEEN 1 AND 5 GROUP BY Cust\_Name, Cust\_ID HAVING Max(Purchase\_Amt)>20

SELECT Cust\_Name, Order\_Date, Max(Purchase\_Amt) FROM SALES GROUP BY Cust\_Name, Order\_Date HAVING Max(Purchase\_Amt) BETWEEN 500 AND 10000

SELECT Cust\_Name, Cust\_ID, Count(Salesman\_ID) FROM SALES GROUP BY Cust\_Name, Cust\_ID HAVING Count(Salesman\_ID)>1  
SELECT Cust\_Name, Cust\_ID, Count(Salesman\_ID) FROM SALES GROUP BY Cust\_Name, Cust\_ID HAVING Count(Salesman\_ID)>2  
SELECT City, Cust\_ID, Count(Salesman\_ID) FROM SALES GROUP BY City, Cust\_ID HAVING Count(Salesman\_ID)>2

SELECT City, Count(Salesman\_ID) FROM SALES GROUP BY City HAVING Count(Salesman\_ID)>2

SELECT City, Count(DISTINCT Salesman\_ID) FROM SALES GROUP BY City HAVING Count(DISTINCT Salesman\_ID)>2

SELECT City, Count(DISTINCT Salesman\_ID) FROM SALES GROUP BY City HAVING Count(DISTINCT Salesman\_ID)>1

SELECT TO\_CHAR(Adm\_Date, 'Day') AS DayOfWeek , COUNT(\*) AS NumberOfStudents FROM STUDENT GROUP BY TO\_CHAR(Adm\_Date, 'Day')

SELECT EXTRACT(YEAR FROM DOB), EXTRACT(MONTH FROM DOB), EXTRACT(DAY FROM DOB) FROM STUDENT

SELECT EXTRACT(YEAR FROM DOB) AS YEAR\_, EXTRACT(MONTH FROM DOB) AS MONTH\_, EXTRACT(DAY FROM DOB) AS DAY\_ FROM STUDENT



**ORDER BY:**

CREATE TABLE SALES (

Cust\_ID int PRIMARY KEY,

Cust\_Name varchar(10) NOT NULL,

Salesman\_ID int,

City varchar(20),

Order\_Date DATE,

Order\_ID int UNIQUE,

Purchase\_Amt float DEFAULT 500.00

)

INSERT INTO SALES VALUES (1, 'A', 123, 'Pune', DATE '2023-08-07', 1111, DEFAULT)

INSERT INTO SALES VALUES (2, 'B', 456, 'Mumbai', DATE '2023-07-06', 2222, 225.9)

INSERT INTO SALES VALUES (3, 'C', 789, 'Delhi', DATE '2023-06-05', 3333, 144)

INSERT INTO SALES VALUES (4, 'D', 123, 'Pune', DATE '2023-05-04', 4444, 1022.05)

INSERT INTO SALES VALUES (5, 'E', 1011, 'Mumbai', DATE '2023-03-02', 5555, DEFAULT)

INSERT INTO SALES VALUES (6, 'F', 1213, 'Delhi', DATE '2023-01-10', 6666, 678)

INSERT INTO SALES VALUES (7, 'A', 123, 'Pune', DATE '2022-12-23', 7777, 991.22)

INSERT INTO SALES VALUES (9, 'A', 1415, 'Delhi', CURRENT\_DATE, 9999, 2002.99)

INSERT INTO SALES VALUES (8, 'B', 456, 'Mumbai', CURRENT\_DATE, 8888, 10000)

INSERT INTO SALES VALUES (10, 'F', 123, 'Pune', CURRENT\_DATE, 1010, DEFAULT)

SELECT \* FROM SALES ORDER BY Cust\_ID ASC

SELECT \* FROM SALES ORDER BY Purchase\_Amt

SELECT \* FROM SALES ORDER BY Purchase\_Amt DESC

SELECT \* FROM SALES ORDER BY Cust\_Name, City

SELECT \* FROM SALES ORDER BY Order\_Date

SELECT \* FROM SALES ORDER BY Salesman\_ID ASC, Order\_Date DESC

SELECT Cust\_Name, City FROM SALES ORDER BY Cust\_Name, City

SELECT \* FROM SALES ORDER BY EXTRACT(MONTH FROM Order\_Date)

SELECT \* FROM SALES ORDER BY EXTRACT(YEAR FROM Order\_Date)

**SET OPERATORS:**

CREATE TABLE SALESMAN (

Salesman\_ID int NOT NULL,

Salesman\_Name varchar(20),

City varchar(30),

Comission float DEFAULT '100.00'

);

CREATE TABLE CUSTOMER (

Cust\_ID int PRIMARY KEY,

Cust\_Name varchar(30),

City varchar(35),

Salesman\_ID int NOT NULL,

Order\_Amt float,

Order\_Date DATE DEFAULT CURRENT\_DATE

);

INSERT INTO SALESMAN VALUES (1, 'A', 'CSN', DEFAULT);

INSERT INTO SALESMAN VALUES (2, 'B', 'CSN', 200);

INSERT INTO SALESMAN VALUES (3, 'C', 'Pune', 340);

INSERT INTO SALESMAN VALUES (4, 'D', 'Pune', DEFAULT);

INSERT INTO SALESMAN VALUES (5, 'E', 'Pune', 150);

INSERT INTO SALESMAN VALUES (6, 'F', 'Delhi', 1005);

INSERT INTO SALESMAN VALUES (7, 'G', 'Delhi', 900);

SELECT \* FROM SALESMAN ORDER BY Salesman\_ID;

INSERT INTO CUSTOMER VALUES (111, 'H', 'CSN', 1, 2000, DEFAULT);

INSERT INTO CUSTOMER VALUES (222, 'I', 'CSN', 2, 1233, DATE '2023-07-09');

INSERT INTO CUSTOMER VALUES (333, 'J', 'Pune', 3, 500, DATE '2023-12-22');

INSERT INTO CUSTOMER VALUES (444, 'K', 'Pune', 5, 10000, DEFAULT);

INSERT INTO CUSTOMER VALUES (555, 'L', 'Delhi', 7, 1300,DATE '2023-08-14');

INSERT INTO CUSTOMER VALUES (666, 'M', 'CSN', 1, 700, DATE '2022-01-10');

INSERT INTO CUSTOMER VALUES (777, 'N', 'CSN', 1, 805, DATE '2021-05-19');

INSERT INTO CUSTOMER VALUES (888, 'O', 'Pune', 3, 445, DATE '2019-06-05');

INSERT INTO CUSTOMER VALUES (999, 'P', 'Delhi', 7, 672, DEFAULT);

SELECT \* FROM CUSTOMER ORDER BY Cust\_ID;

SELECT Salesman\_ID, Salesman\_Name, City, Comission, NULL AS Cust\_ID, NULL AS Cust\_Name, NULL AS Order\_Amt, NULL AS Order\_Date FROM SALESMAN UNION SELECT Salesman\_ID, NULL AS Salesman\_Name, City, NULL AS Comission, Cust\_ID, Cust\_Name, Order\_Amt, Order\_Date FROM Customer;

SELECT Salesman\_ID, Salesman\_Name, City, Comission, NULL AS Cust\_ID, NULL AS Cust\_Name, NULL AS Order\_Amt, NULL AS Order\_Date FROM SALESMAN UNION ALL SELECT Salesman\_ID, NULL, City, NULL, Cust\_ID, Cust\_Name, Order\_Amt, Order\_Date FROM CUSTOMER

SELECT Salesman\_ID, City FROM Salesman INTERSECT SELECT Salesman\_ID, City FROM Customer;

SELECT Salesman\_ID, City FROM Salesman MINUS SELECT Salesman\_ID, City FROM Customer;

SELECT Salesman\_Name, NULL AS Customer\_Name, City FROM SALESMAN WHERE City='Delhi' UNION SELECT NULL, Cust\_Name, City FROM CUSTOMER WHERE City='Delhi';

SELECT Salesman\_Name AS NAME, 'Salesman' AS ROLE FROM Salesman UNION SELECT Cust\_Name, 'Customer' AS ROLE FROM Customer;

SELECT Salesman\_ID FROM SALESMAN INTERSECT SELECT Salesman\_ID FROM CUSTOMER WHERE Order\_Amt=(SELECT Max(Order\_Amt) FROM CUSTOMER WHERE Order\_Date=DATE'2023-08-14');

SELECT Salesman\_ID FROM SALESMAN INTERSECT SELECT Salesman\_ID FROM CUSTOMER WHERE Order\_Amt=(SELECT Min(Order\_Amt) FROM CUSTOMER WHERE Order\_Date=DATE'2019-06-05');

SELECT Salesman\_ID AS ID, City FROM SALESMAN INTERSECT SELECT Salesman\_ID, City FROM CUSTOMER;

SELECT Salesman\_ID FROM SALESMAN MINUS SELECT Salesman\_ID FROM CUSTOMER;

**JOIN:**

CREATE TABLE SAILER(S\_ID INT PRIMARY KEY,S\_NAME VARCHAR(50),RATING INT,AGE INT);

CREATE TABLE BOAT(B\_ID INT PRIMARY KEY,B\_NAME VARCHAR(50),COLOUR VARCHAR(50));

CREATE TABLE RESERVATION(S\_ID INT,B\_ID INT,R\_DATE DATE,FOREIGN KEY(S\_ID) REFERENCES SAILER(S\_ID),FOREIGN KEY(B\_ID) REFERENCES BOAT(B\_ID));

INSERT INTO SAILER VALUES(22,'A',7,45);

INSERT INTO SAILER VALUES(29,'B',1,33);

INSERT INTO SAILER VALUES(31,'C',8,55);

INSERT INTO SAILER VALUES(32,'D',8,25);

INSERT INTO SAILER VALUES(58,'E',10,35);

INSERT INTO SAILER VALUES(64,'F',7,35);

INSERT INTO SAILER VALUES(71,'G',10,16);

INSERT INTO SAILER VALUES(74,'H',9,40);

INSERT INTO SAILER VALUES(85,'I',3,25);

INSERT INTO SAILER VALUES(95,'J',3,63);

SELECT \* FROM SAILER ORDER BY S\_ID;

INSERT INTO BOAT VALUES(101,'INTERLAKE','BLUE');

INSERT INTO BOAT VALUES(102,'INTERLAKE','RED');

INSERT INTO BOAT VALUES(103,'CLIPER','GREEN');

INSERT INTO BOAT VALUES(104,'MARINE','RED');

SELECT \* FROM BOAT ORDER BY B\_ID;

INSERT INTO RESERVATION VALUES(22,101,DATE '2022-10-10');

INSERT INTO RESERVATION VALUES(22,102,DATE '2021-10-10');

INSERT INTO RESERVATION VALUES(22,103,DATE '2014-08-15');

INSERT INTO RESERVATION VALUES(22,104,DATE '2015-07-10');

INSERT INTO RESERVATION VALUES(31,102,DATE '2009-10-11');

INSERT INTO RESERVATION VALUES(31,103,DATE '2008-06-11');

INSERT INTO RESERVATION VALUES(31,104,DATE '2008-12-11');

INSERT INTO RESERVATION VALUES(64,101,DATE '2010-05-09');

INSERT INTO RESERVATION VALUES(64,102,DATE '2008-08-09');

INSERT INTO RESERVATION VALUES(74,103,DATE '2009-08-09');

SELECT \* FROM RESERVATION ORDER BY S\_ID;

UPDATE RESERVATION SET S\_ID=31 WHERE R\_DATE=DATE '2009-10-10';

UPDATE RESERVATION SET S\_ID=31 WHERE R\_DATE=DATE '2008-06-11';

UPDATE RESERVATION SET S\_ID=31 WHERE R\_DATE=DATE '2008-12-11';

UPDATE RESERVATION SET S\_ID=64 WHERE R\_DATE=DATE '2010-05-09';

UPDATE RESERVATION SET S\_ID=64 WHERE R\_DATE=DATE '2008-08-09';

UPDATE RESERVATION SET S\_ID=74 WHERE R\_DATE=DATE '2009-08-09';

SELECT S\_NAME,AGE FROM SAILER;

SELECT \* FROM SAILER WHERE RATING>7;

SELECT SAILER.S\_NAME FROM SAILER, RESERVATION WHERE SAILER.S\_ID=RESERVATION.S\_ID AND B\_ID=103;

SELECT S\_ID FROM BOAT,RESERVATION WHERE RESERVATION.B\_ID=BOAT.B\_ID AND COLOUR='RED';

SELECT COLOUR FROM BOAT,RESERVATION,SAILER WHERE RESERVATION.B\_ID=BOAT.B\_ID AND SAILER.S\_ID=RESERVATION.S\_ID AND S\_NAME='C';

SELECT S\_NAME FROM BOAT,RESERVATION,SAILER WHERE RESERVATION.B\_ID=BOAT.B\_ID AND SAILER.S\_ID=RESERVATION.S\_ID AND COLOUR='RED';

**TYPES OF JOINS:**

CREATE TABLE STUDENT(

S\_ROLL INT,

S\_NAME VARCHAR(20),

D\_ID INT

);

CREATE TABLE DEPARTMENT(

D\_ID INT,

D\_NAME VARCHAR(20),

HOD VARCHAR(20)

);

INSERT INTO STUDENT VALUES (101, 'P', 10);

INSERT INTO STUDENT VALUES (102, 'Q', 10);

INSERT INTO STUDENT VALUES (103, 'R', 20);

INSERT INTO STUDENT VALUES (104, 'S', 40);

INSERT INTO STUDENT VALUES (105, 'X', 50);

INSERT INTO DEPARTMENT VALUES (10, 'CSE', 'A');

INSERT INTO DEPARTMENT VALUES (30, 'MCA', 'C');

INSERT INTO DEPARTMENT VALUES (20, 'IT', 'B');

INSERT INTO DEPARTMENT VALUES (40, 'ECT', 'D');

SELECT \* FROM STUDENT ORDER BY S\_ROLL;

SELECT \* FROM DEPARTMENT ORDER BY D\_ID;

SELECT \* FROM STUDENT INNER JOIN DEPARTMENT ON STUDENT.D\_ID=DEPARTMENT.D\_ID;

SELECT \* FROM STUDENT LEFT JOIN DEPARTMENT ON STUDENT.D\_ID=DEPARTMENT.D\_ID;

SELECT \* FROM STUDENT JOIN DEPARTMENT ON STUDENT.D\_ID=DEPARTMENT.D\_ID;

SELECT \* FROM STUDENT JOIN DEPARTMENT ON STUDENT.D\_ID!=DEPARTMENT.D\_ID;

SELECT \* FROM STUDENT NATURAL JOIN DEPARTMENT;

SELECT \* FROM STUDENT NATURAL JOIN DEPARTMENT;

SELECT \* FROM STUDENT JOIN DEPARTMENT ON STUDENT.D\_ID!=DEPARTMENT.D\_ID;

SELECT \* FROM STUDENT FULL JOIN DEPARTMENT ON STUDENT.D\_ID=DEPARTMENT.D\_ID;

SELECT \* FROM STUDENT INNER JOIN DEPARTMENT ON STUDENT.D\_ID=DEPARTMENT.D\_ID;

SELECT \* FROM STUDENT INNER JOIN DEPARTMENT ON STUDENT.D\_ID=DEPARTMENT.D\_ID;

SELECT \* FROM STUDENT NATURAL JOIN DEPARTMENT;

SELECT \* FROM STUDENT FULL JOIN DEPARTMENT ON STUDENT.D\_ID=DEPARTMENT.D\_ID;

SELECT \* FROM STUDENT NATURAL JOIN DEPARTMENT;

SELECT \* FROM STUDENT INNER JOIN DEPARTMENT ON STUDENT.D\_ID=DEPARTMENT.D\_ID;

**FOREIGN\_KEY:**

CREATE TABLE EMPLOYEE (

Emp\_ID int PRIMARY KEY,

Emp\_Name varchar(25)

);

INSERT INTO EMPLOYEE VALUES (101, 'John');

INSERT INTO EMPLOYEE VALUES (103, 'Smith');

INSERT INTO EMPLOYEE VALUES (104, 'Lal');

INSERT INTO EMPLOYEE VALUES (106, 'Byron');

INSERT INTO EMPLOYEE VALUES (107, 'Ivan');

INSERT INTO EMPLOYEE VALUES (110, 'Drew');

INSERT INTO EMPLOYEE VALUES (112, 'Smith');

SELECT \* FROM EMPLOYEE ORDER BY Emp\_ID;

CREATE TABLE PROJECT (

P\_ID varchar(20) PRIMARY KEY,

P\_Name varchar(20),

Chief int

);

INSERT INTO PROJECT VALUES ('COMP231', 'Pascal', 107);

INSERT INTO PROJECT VALUES ('COMP278', 'Pascal', 110);

INSERT INTO PROJECT VALUES ('COMP353', 'Database', 107);

INSERT INTO PROJECT VALUES ('COMP354', 'OS', 104);

INSERT INTO PROJECT VALUES ('COMP453', 'Database', 101);

SELECT \* FROM PROJECT ORDER BY Chief;

CREATE TABLE ASSIGNED\_TO (

P\_ID varchar(20),

Emp\_ID int,

FOREIGN KEY(P\_ID) REFERENCES PROJECT(P\_ID),

FOREIGN KEY(Emp\_ID) REFERENCES EMPLOYEE(Emp\_ID)

);

INSERT INTO ASSIGNED\_TO VALUES ('COMP453', 101);

INSERT INTO ASSIGNED\_TO VALUES ('COMP354', 103);

INSERT INTO ASSIGNED\_TO VALUES ('COMP354', 104);

INSERT INTO ASSIGNED\_TO VALUES ('COMP231', 106);

INSERT INTO ASSIGNED\_TO VALUES ('COMP278', 106);

INSERT INTO ASSIGNED\_TO VALUES ('COMP353', 106);

INSERT INTO ASSIGNED\_TO VALUES ('COMP354', 106);

INSERT INTO ASSIGNED\_TO VALUES ('COMP453', 106);

INSERT INTO ASSIGNED\_TO VALUES ('COMP231', 107);

INSERT INTO ASSIGNED\_TO VALUES ('COMP353', 107);

INSERT INTO ASSIGNED\_TO VALUES ('COMP278', 110);

INSERT INTO ASSIGNED\_TO VALUES ('COMP353', 112);

INSERT INTO ASSIGNED\_TO VALUES ('COMP354', 112);

SELECT \* FROM ASSIGNED\_TO ORDER BY Emp\_ID;

SELECT Emp\_ID FROM Assigned\_To WHERE P\_ID='COMP353';

SELECT \* FROM EMPLOYEE,ASSIGNED\_TO WHERE EMPLOYEE.EMP\_ID=ASSIGNED\_TO.EMP\_ID AND P\_ID='COMP354';

SELECT EMP\_ID, EMP\_NAME, P\_ID FROM EMPLOYEE, PROJECT WHERE EMPLOYEE.EMP\_ID = PROJECT.CHIEF AND P\_NAME='OS';

SELECT \* FROM EMPLOYEE, PROJECT, ASSIGNED\_TO WHERE EMPLOYEE.EMP\_ID=ASSIGNED\_TO.EMP\_ID AND PROJECT.P\_ID=ASSIGNED\_TO.P\_ID AND P\_NAME='Database';

SELECT COUNT(PROJECT.P\_ID) FROM EMPLOYEE, ASSIGNED\_TO, PROJECT WHERE EMPLOYEE.EMP\_ID=ASSIGNED\_TO.EMP\_ID AND PROJECT.P\_ID=ASSIGNED\_TO.P\_ID AND EMP\_NAME='Byron';

SELECT DISTINCT P\_NAME FROM PROJECT;

SELECT EMP\_NAME FROM EMPLOYEE ORDER BY EMP\_NAME DESC;

SELECT EMPLOYEE.EMP\_NAME FROM EMPLOYEE, PROJECT, ASSIGNED\_TO WHERE EMPLOYEE.EMP\_ID=ASSIGNED\_TO.EMP\_ID AND PROJECT.P\_ID=ASSIGNED\_TO.P\_ID AND P\_NAME='Pascal';

ALTER TABLE PROJECT ADD DURATION INT;

UPDATE PROJECT SET DURATION=20;

SELECT SUM(DURATION) AS TOTAL\_DURATION\_OF\_PROJECT FROM PROJECT;

SELECT \* FROM EMPLOYEE, ASSIGNED\_TO, PROJECT WHERE EMPLOYEE.EMP\_ID=ASSIGNED\_TO.EMP\_ID AND PROJECT.P\_ID=ASSIGNED\_TO.P\_ID AND EMP\_NAME='Lal';

SELECT COUNT(EMP\_ID) FROM EMPLOYEE;

UPDATE PROJECT SET P\_NAME='BUISNESS APPLICATION' WHERE P\_Name='Pascal';

ALTER TABLE ASSIGNED\_TO DROP COLUMN P\_ID;

ALTER TABLE ASSIGNED\_TO ADD P\_ID varchar(20);

ALTER TABLE ASSIGNED\_TO ADD FOREIGN KEY(P\_ID) REFERENCES PROJECT (P\_ID) ON DELETE CASCADE;

DELETE FROM Project WHERE p\_name='OS'

**INDEXING AND SEQUENCING:**

CREATE TABLE College (

PRN int PRIMARY KEY,

Roll int UNIQUE,

Name varchar(20) NOT NULL,

Course varchar(10),

Grade varchar(10) DEFAULT '-',

Age int,

CHECK (Age>=18)

);

CREATE INDEX I1 ON College (PRN, Roll, Name, Course, Grade);

CREATE INDEX I2 ON College (PRN, Age);

CREATE SEQUENCE s1

START WITH 101

INCREMENT BY 1

MINVALUE 0

MAXVALUE 115

CYCLE ;

INSERT INTO College VALUES (3157, 104, 'D', 'CSE', 'C+', 20);

INSERT INTO College VALUES (3154, 101, 'A', 'CSE', 'A+', 22);

INSERT INTO College VALUES (3155, 102, 'B', 'CSE', 'A', 20);

INSERT INTO College VALUES (3156, 103, 'C', 'IT', 'B+', 19);

INSERT INTO College VALUES (3158, 105, 'E', 'IT', 'F', 22);

INSERT INTO College VALUES (3159, 106, 'F', 'CSE', 'D+', 21);

INSERT INTO College VALUES (3160, 107, 'G', 'IT', 'A+', 20);

SELECT \* FROM College;

INSERT INTO College VALUES (s1.NEXTVAL, 108, 'H', 'CSE', 'C+', 22);

SELECT \* FROM College;

CREATE SEQUENCE s

START WITH 3154

INCREMENT BY 1

MINVALUE 0

MAXVALUE 3200

CYCLE ;

TRUNCATE TABLE College

SELECT \* FROM College;

SELECT \* FROM College;

INSERT INTO College VALUES (s.NEXTVAL, s1.NEXTVAL, 'A', 'CSE', 'A+', 22);

SELECT \* FROM College;

SELECT \* FROM College;

SELECT \* FROM College;

INSERT INTO College VALUES (s.NEXTVAL, s1.NEXTVAL, 'B', 'CSE', 'A', 20)

SELECT \* FROM College;

TRUNCATE TABLE College

INSERT INTO College VALUES (s.NEXTVAL, 101, 'A', 'CSE', 'A+', 22);

INSERT INTO College VALUES (s.NEXTVAL, s1.NEXTVAL, 'B', 'CSE', 'A', 20);

SELECT \* FROM College;

TRUNCATE TABLE College

CREATE SEQUENCE seq

START WITH 101

INCREMENT BY 1

MINVALUE 101

MAXVALUE 200

CYCLE ;

INSERT INTO College VALUES (s.NEXTVAL, seq.NEXTVAL, 'A', 'CSE', 'A+', 22);

INSERT INTO College VALUES (s.NEXTVAL, seq.NEXTVAL, 'B', 'CSE', 'A', 20);

SELECT \* FROM College;

CREATE SEQUENCE seq1

START WITH 3154

INCREMENT BY 1

MINVALUE 3154

MAXVALUE 3200

CYCLE ;

CREATE SEQUENCE seq2

START WITH 101

INCREMENT BY 1

MINVALUE 101

MAXVALUE 200

CYCLE ;

TRUNCATE TABLE College

INSERT INTO College VALUES (seq1.NEXTVAL, seq2.NEXTVAL, 'B', 'CSE', 'A', 20);

INSERT INTO College VALUES (seq1.NEXTVAL, seq2.NEXTVAL, 'A', 'CSE', 'A+', 22);

SELECT \* FROM College;

**DATE FUNCTION:**

SELECT TO\_CHAR(DOJ, 'DD-MON-YYYY') AS DATE\_ FROM EMPLOYEE;

SELECT TO\_CHAR(DOJ, 'DD-MONTH-YEAR') AS DATE\_ FROM EMPLOYEE;

SELECT DOJ, TO\_CHAR(DOJ, 'DAY') AS DAY\_ FROM EMPLOYEE GROUP BY DOJ;

SELECT LISTAGG(TO\_CHAR(SALARY, '999,999,999,990.00'), ',') WITHIN GROUP (ORDER BY SALARY) AS CSV\_SALARIES FROM EMPLOYEE;

SELECT TO\_NUMBER(EMP\_ID) FROM EMPLOYEE;

SELECT LISTAGG(TO\_CHAR(SALARY, '999,999,999,990.00'), ',') AS CSV\_SALARIES FROM EMPLOYEE;

SELECT TO\_CHAR(SALARY, '999,999,999,990.00') AS CSV\_SALARIES FROM EMPLOYEE;

CREATE TABLE EmployeeInfo (

EmpID INT PRIMARY KEY,

EmpName VARCHAR(255),

Project VARCHAR(255),

Address VARCHAR(255),

DoB DATE,

Gender VARCHAR(10)

);

CREATE TABLE EmployeePosition (

EmpID INT PRIMARY KEY,

Position VARCHAR(255),

DoJ DATE,

Salary FLOAT,

FOREIGN KEY (EmpID) REFERENCES EmployeeInfo(EmpID)

);  
INSERT INTO EmployeeInfo VALUES (1, 'John Smith', 'Project A', '123 Main St, City', DATE '1990-05-15', 'Male');

INSERT INTO EmployeeInfo VALUES (2, 'Jane Doe', 'Project B', '456 Elm St, Town', DATE '1988-09-20', 'Female');

INSERT INTO EmployeeInfo VALUES (3, 'Michael Johnson', 'Project A', '789 Oak St, Village', DATE '1995-02-10', 'Male');

INSERT INTO EmployeeInfo VALUES (4, 'Emily Brown', 'Project C', '987 Maple St, Town', DATE '1992-12-01', 'Female');

INSERT INTO EmployeeInfo VALUES (5, 'David Wilson', 'Project B', '654 Pine St, City', DATE '1991-08-08', 'Male');

INSERT INTO EmployeePosition VALUES (2, 'Developer', DATE '2019-07-01', 60000.00);

INSERT INTO EmployeePosition VALUES (1, 'Manager', DATE '2018-03-15', 80000.00);

INSERT INTO EmployeePosition VALUES (5, 'Developer', DATE '2016-05-05', 65000.00);

INSERT INTO EmployeePosition VALUES (3, 'Analyst', DATE '2020-01-10', 55000.00);

INSERT INTO EmployeePosition VALUES (4, 'Designer', DATE '2017-11-20', 58000.00);

SELECT \* FROM EmployeeInfo;

SELECT \* FROM EmployeePosition;

SELECT EmpName, DoB, Gender FROM EmployeeInfo i, EmployeePosition p WHERE i.EmpID = p.EmpID AND p.Salary BETWEEN 50000 AND 100000;

SELECT \* FROM EmployeeInfo WHERE EmpName LIKE 'J%\_\_\_\_';

SELECT Position, COUNT(\*) AS EmployeeCount FROM EmployeePosition GROUP BY Position;

SELECT MIN(Salary) AS MINIMUM\_SALARY, MAX(Salary) AS MAXIMUM\_SALARY FROM EmployeePosition;

UPDATE EmployeePosition SET Salary = Salary \* 1.2 WHERE Position = 'Manager';

SELECT \* FROM EmployeeInfo i, EmployeePosition p WHERE i.EmpID = p.EmpID AND Position = 'Manager' AND Project = 'Project A';

SELECT \* FROM EmployeeInfo i, EmployeePosition p WHERE i.EmpID = p.EmpID ORDER BY p.Salary DESC;

SELECT \* FROM EmployeeInfo WHERE DOB = (SELECT MAX(DOB) FROM EmployeeInfo);

CREATE TABLE Employee (

Emp\_No int PRIMARY KEY,

Emp\_Name varchar(25) NOT NULL,

Skill varchar(25),

Pay\_rate float

);

INSERT INTO Employee VALUES (56, 'Ron', 'Waiter', 7.50);

INSERT INTO Employee VALUES (57, 'John', 'Lab Attendant', 8.79);

INSERT INTO Employee VALUES (58, 'Don', 'Bus Boy', 4.70);  
INSERT INTO Employee VALUES (59, 'Abc', 'Clerk', 4.90);

INSERT INTO Employee VALUES (60, 'Smith', 'Day Boy', 4.70);

INSERT INTO Employee VALUES (72, 'John', 'Chief', 14.00);

INSERT INTO Employee VALUES (61, 'Abc', 'Chief', 14.00);

INSERT INTO Employee VALUES (71, 'Day', 'Chief', 14.50);

SELECT \* FROM Employee ORDER BY Emp\_No;

CREATE TABLE Duty\_Allocation (

Emp\_No int,

Day\_ DATE,

Shifts\_ varchar(10),

FOREIGN KEY(Emp\_No) REFERENCES Employee(Emp\_No)

);

INSERT INTO Duty\_Allocation VALUES (56, DATE '2015-08-15', '1st');  
INSERT INTO Duty\_Allocation VALUES (57, DATE '2015-08-20', '2nd');

INSERT INTO Duty\_Allocation VALUES (61, DATE '2015-08-22', '2nd');

INSERT INTO Duty\_Allocation VALUES (58, DATE '2015-08-20', '1st');

INSERT INTO Duty\_Allocation VALUES (71, DATE '2015-08-23', '1st');

INSERT INTO Duty\_Allocation VALUES (61, DATE '2015-08-25', '2nd');

INSERT INTO Duty\_Allocation VALUES (58, DATE '2015-08-21', '3rd');

INSERT INTO Duty\_Allocation VALUES (71, DATE '2015-08-19', '2nd');

SELECT \* FROM Duty\_Allocation ORDER BY Emp\_No;

SELECT Avg(Pay\_rate) AS Average\_Pay\_Rate FROM Employee WHERE Skill='Chief';

SELECT COUNT(DISTINCT Pay\_rate) AS Count\_Of\_Distinct\_Pay\_Rates FROM Employee;

SELECT \* FROM Employee e, Duty\_Allocation d WHERE e.Emp\_No = d.Emp\_No AND (d.DAY\_=DATE '2015-08-21' OR d.DAY\_=DATE '2015-08-20');

SELECT \* FROM Employee e, Duty\_Allocation d WHERE e.Emp\_No = d.Emp\_No AND e.Pay\_Rate = (SELECT MIN(Pay\_Rate) FROM Employee);

SELECT DISTINCT Emp\_Name, Pay\_Rate FROM Employee e, Duty\_Allocation d WHERE e.Emp\_No = d.Emp\_No;

SELECT Emp\_Name, Pay\_Rate FROM Employee WHERE Emp\_No NOT IN (SELECT Emp\_No FROM Duty\_Allocation);

SELECT \* FROM Employee e, Duty\_Allocation d WHERE e.Emp\_No = d.Emp\_No AND d.Shifts\_='3rd';

SELECT COUNT(\*) FROM Duty\_Allocation WHERE DAY\_ BETWEEN DATE '15-08-22' AND DATE '15-08-25';

UPDATE Employee SET Pay\_Rate = Pay\_Rate\*1.1 WHERE Skill='Bus Boy';

SELECT \* FROM Employee WHERE Emp\_Name LIKE '\_a\_' ;

**NESTED QUERIES:**

CREATE TABLE Employee (

emp\_id int PRIMARY KEY,

name varchar(20) NOT NULL,

phone\_no number UNIQUE,

salary float,

designation varchar(20),

doj DATE,

email\_id varchar(20) DEFAULT '-',

CHECK (salary > 0)

);

INSERT INTO Employee VALUES (1, 'John Doe', 1234567890, 50000.00, 'Manager', DATE '2023-09-25', 'john.doe@example.com');

INSERT INTO Employee VALUES (2, 'Jane Smith', 9876543210, 45000.00, 'Developer', DATE '2023-08-25', 'js@example.com');

INSERT INTO Employee VALUES (3, 'Robert Johnson', 5551234567, 55000.00, 'Designer', DATE '2023-07-25', 'rj@example.com');

INSERT INTO Employee VALUES (4, 'Mary Brown', 4445556666, 60000.00, 'Intern', DATE '2023-06-25', 'mb@example.com');

INSERT INTO Employee VALUES (5, 'David Lee', 9998887777, 60000.00, 'Manager', DATE '2023-05-25', DEFAULT);

INSERT INTO Employee VALUES (6, 'Sarah White', 1112223333, 48000.00, 'Developer', DATE '2023-04-25', 'sw@example.com');

INSERT INTO Employee VALUES (7, 'Michael Johnson', 7778889999, 52000.00, 'Designer', DATE '2023-03-25', DEFAULT);

INSERT INTO Employee VALUES (8, 'Elizabeth Windsor', 964201357, 70000.00, 'Manager', DATE '2023-02-25', 'queen@example.com');

INSERT INTO Employee VALUES (9, 'Thomas Anderson', 12121212, 60000.00, 'Developer', DATE '2023-01-25', 'neo@example.com');

INSERT INTO Employee VALUES (10, 'Grace Kelly', 8887776666, 58000.00, 'Intern', DATE '2023-10-25', DEFAULT);

SELECT \* FROM Employee ORDER BY emp\_id;

SELECT MIN(salary) AS LOWEST\_SALARY, MAX(salary) AS HIGHEST\_SALARY FROM Employee;

SELECT SUM(salary) AS TOTAL\_PAYABLE\_SALARY FROM Employee;

SELECT SUM(salary) AS TOTAL\_PAYABLE\_SALARY FROM Employee WHERE designation='Manager';

SELECT designation,COUNT(\*) FROM Employee GROUP BY designation;

ALTER TABLE Employee ADD department varchar(20);

UPDATE Employee SET department='HR' WHERE emp\_id=5 OR emp\_id=8;

SELECT AVG(salary) AS AVG\_SAL\_OF\_CSE\_EMP FROM Employee WHERE department='CSE';

SELECT MAX(salary) AS HIGHEST\_SALARY, MIN(salary) AS LOWEST\_SALARY, SUM(salary) AS TOTAL\_SALARY, AVG(salary) AS AVERAGE\_SALARY FROM Employee WHERE department='CSE' AND designation='Developer';

SELECT \* FROM Employee WHERE salary=(SELECT MIN(salary) FROM Employee);

SELECT \* FROM Employee WHERE salary=(SELECT MAX(salary) FROM Employee) AND designation='Manager' AND department='HR';

SELECT AVG(salary) AS AVERAGE\_SAL\_OF\_MANAGERS FROM Employee WHERE designation='Manager';

SELECT \* FROM Employee WHERE salary>(SELECT MAX(salary) FROM Employee WHERE designation='Developer');

SELECT COUNT(\*) AS COUNT\_OF\_DEVS\_AND\_DES FROM Employee WHERE designation IN('Designer','Developer');

**VIEWS AND SUBQUERIES:**

CREATE TABLE FACULTY (

F\_ID int PRIMARY KEY,

F\_Name varchar(20),

L\_Name varchar(20),

Mail varchar(20),

Phone\_No number,

Salary float,

Dept varchar(10),

Project\_ID varchar(10),

DOJ DATE

);

INSERT INTO FACULTY VALUES (2, 'C', 'D', 'CD@example.com', 22222, 40000, 'IT', NULL, DATE'2004-08-15');

INSERT INTO FACULTY VALUES (5, 'I', 'J', 'IJ@example.com', 55555, 70000, 'CSE', 'COMP123', DATE'2007-11-18');

DROP TABLE FACULTY;

CREATE TABLE FACULTY (

F\_ID int PRIMARY KEY,

F\_Name varchar(20) NOT NULL,

L\_Name varchar(20),

Mail varchar(20),

Phone\_No number UNIQUE,

Salary float CHECK (Salary>=10000.00),

Dept varchar(10),

Project\_ID varchar(10),

DOJ DATE DEFAULT date '2023-01-01'

);

INSERT INTO FACULTY VALUES (1, 'A', 'B', 'AB@example.com', 11111, 30000, 'CSE', 'COMP123', DATE'2003-07-14');

INSERT INTO FACULTY VALUES (2, 'C', 'D', 'CD@example.com', 22222, 40000, 'IT', NULL, DEFAULT);

INSERT INTO FACULTY VALUES (3, 'E', 'F', 'EF@example.com', 33333, 50000, 'CSE', 'COMP321', DATE'2005-09-16');

INSERT INTO FACULTY VALUES (5, 'I', 'J', 'IJ@example.com', 55555, 70000, 'CSE', 'COMP123', DEFAULT);

INSERT INTO FACULTY VALUES (4, 'G', 'H', 'GH@example.com', 44444, 60000, 'IT', NULL, DATE'2006-10-17');

INSERT INTO FACULTY VALUES (7, 'M', 'N', 'MN@example.com', 77777, 90000, 'CSE', 'COMP321', DATE'2003-01-20');

INSERT INTO FACULTY VALUES (6, 'K', 'L', 'KL@example.com', 66666, 80000, 'IT', NULL, DATE'2007-12-19');

SELECT \* FROM FACULTY;

SELECT \* FROM FACULTY ORDER BY F\_ID;

SELECT AVG(Salary) FROM FACULTY;

SELECT \* FROM FACULTY WHERE Salary>(SELECT AVG(Salary) FROM FACULTY);

SELECT Count(Project\_ID) FROM FACULTY;

SELECT \* FROM FACULTY WHERE Salary<ANY(SELECT MIN(Salary) FROM FACULTY);

SELECT \* FROM FACULTY WHERE Salary>ANY(SELECT MIN(Salary) FROM FACULTY);

SELECT \* FROM FACULTY WHERE Salary<ANY(SELECT MIN(Salary) FROM FACULTY WHERE Dept='CSE');

SELECT \* FROM FACULTY WHERE Salary<ANY(SELECT MIN(Salary) FROM FACULTY WHERE Dept='IT');

SELECT MAX(Salary) FROM FACULTY WHERE DOJ IN (DATE '2023-01-01', DATE '2003-07-14', DATE '2006-10-17');

SELECT MAX(Salary) FROM FACULTY WHERE Salary<(SELECT MAX(Salary) FROM FACULTY);

SELECT \* FROM FACULTY WHERE Project\_ID <> NULL;

SELECT \* FROM FACULTY WHERE Project\_ID IS NOT NULL;

SELECT F\_Name FROM FACULTY WHERE PROJECT\_ID IS NULL;

SELECT \* FROM FACULTY WHERE SALARY IN (SELECT MIN(salary) FROM FACULTY GROUP BY Dept);

CREATE VIEW F AS

SELECT \*

FROM Faculty;

SELECT \* FROM F;

CREATE VIEW F2 AS

SELECT F\_ID, F\_NAME, L\_NAME, MAIL

FROM Faculty;

CREATE VIEW F3 AS

SELECT Salary

FROM Faculty WHERE Salary=(SELECT Max(Salary) FROM Faculty);

SELECT \* FROM F3;

CREATE VIEW F4 AS

SELECT DEPT FROM Faculty;

SELECT \* FROM F4;

SELECT \* FROM F4 GROUP BY DEPT;

UPDATE F2 SET MAIL='JI@example.com' WHERE MAIL='IJ@example.com';

SELECT \* FROM F2;

SELECT F\_ID, F\_NAME FROM F2;

RENAME F To F1

DROP VIEW F2;

DELETE FROM FACULTY WHERE F\_ID=6;

DELETE FROM F1 WHERE F\_ID=7;

SELECT \* FROM Faculty;

SELECT \* FROM F1;

**USER MANAGEMENT:**

SHOW DATABASES;

CREATE DATABASE DB1;

USE DB1;

CREATE TABLE T1 (c1 int, c2 varchar(10));

INSERT INTO T1 VALUES (1, 'A');

CREATE TABLE T2 (col float);

INSERT INTO T1 VALUES (999.99);

SELECT \* FROM T1;

SELECT \* FROM T2;

CREATE USER 'U1'@'localhost' IDENTIFIED BY 'PASS1';

CREATE USER 'U2'@'localhost' IDENTIFIED BY 'PASS2';

SELECT USER FROM MYSQL.USER;

SELECT user, host FROM mysql.user;

RENAME USER 'U2'@'localhost' TO 'User'@'localhost';

DROP USER 'User'@'localhost';

ALTER USER 'U1'@'localhost' IDENTIFIED BY 'PASSWORD';

UPDATE MYSQL.USER SET AUTHENTICATION\_STRING='NEW\_PASS' WHERE USER='U1';

SET PASSWORD = 'FINAL';

GRANT ALL PRIVILEGES ON \*.\* TO 'U1'@'localhost';

REVOKE DROP, DELETE ON \*.\* FROM 'U2'@'localhost';

CREATE USER user1 IDENTIFIED BY 'p1';

CREATE USER user2 IDENTIFIED BY 'p2';

GRANT ALL PRIVILEGES ON DB.Emp TO user1;

GRANT ALL PRIVILEGES ON DB.Dept TO user2;

GRANT ALL PRIVILEGES ON DB.Emp TO user2;

REVOKE ALL PRIVILEGES ON DB.Emp FROM user2;

GRANT SELECT, UPDATE ON DB.Dept TO user1;

REVOKE SELECT, UPDATE ON DB.Dept FROM user1;

**TRANSACTION:**

SHOW DATABASES;

CREATE DATABASE BankTransaction;

USE BankTransaction;

CREATE TABLE Customer (name varchar(20), ifsc varchar(20), balance int);

START TRANSACTION;

BEGIN;

INSERT INTO Customer VALUES ('C1', 'ABC123', 2000);

SAVEPOINT A;

UPDATE Customer SET balance=balance+500;

SET SQL\_SAFE\_UPDATES = 0;

INSERT INTO Customer VALUES ('C2', 'DEF456', 5000);

SAVEPOINT B;

INSERT INTO Customer VALUES ('C3', 'GHI789', 10000);

SAVEPOINT C;

INSERT INTO Customer VALUES ('C4', 'JKL1011', 1200);

SAVEPOINT D;

INSERT INTO Customer VALUES ('C5', 'MNO1213', 4500);

SAVEPOINT E;

SELECT \* FROM Customer;

ROLLBACK TO B;

COMMIT;

ROLLBACK TO A;

**PLSQL:**

DECLARE

id INTEGER;

name VARCHAR(20);

dob DATE;

type int\_arr IS VARRAY(5) OF INTEGER;

arr int\_arr;

BEGIN

id:=1;

name:='ABC';

dob:= TO\_DATE('2003-03-04','YYYY-MM-DD');

arr:=int\_arr(1, 2, 3, 4, 5);

dbms\_output.put\_line('ID: ' || id);

dbms\_output.put\_line('Name: ' || name);

dbms\_output.put\_line('DOB: ' || dob);

FOR i in 1 .. 5 LOOP

dbms\_output.put\_line(arr(i));

END LOOP;

END;

DECLARE

n NUMBER := 1634;

r NUMBER;

BEGIN

r := MOD(n, 2);

dbms\_output.Put\_line('The number is: ' || n);

IF r = 0 THEN

dbms\_output.Put\_line(n || ' is Even');

ELSE

dbms\_output.Put\_line(n || ' is Odd');

END IF;

END;

DECLARE

type int\_arr IS VARRAY(10) OF INTEGER;

arr int\_arr;

BEGIN

arr:=int\_arr(1, 2, 3, 4, 5, 6, 7, 8, 9, 10);

FOR i in 1 .. 10 LOOP

IF arr(i)<5

THEN

dbms\_output.put\_line('No. ' || arr(i) || ' is in loop');

ELSE

dbms\_output.put\_line('No. ' || arr(i) || ' is not in loop');

END IF;

END LOOP;

END;

DECLARE

e\_id Employee.id%type;

e\_dept Employee.dept%type;

e\_salary Employee.salary%type;

ex\_invalid\_salary EXCEPTION;

BEGIN

IF e\_salary <= 0 THEN

RAISE ex\_invalid\_salary;

ELSE

SELECT id, dept, salary INTO e\_id, e\_dept, e\_salary

FROM Employee

WHERE id = e\_id;

DBMS\_OUTPUT.PUT\_LINE ('Dept: ' || e\_dept);

DBMS\_OUTPUT.PUT\_LINE ('Salary: ' || e\_salary);

END IF;

EXCEPTION

WHEN ex\_invalid\_salary THEN

dbms\_output.put\_line('Salary must be greater than zero!');

WHEN no\_data\_found THEN

dbms\_output.put\_line('No such customer!');

WHEN others THEN

dbms\_output.put\_line('Error!');

END;

DECLARE

e\_id Employee.id%type := 2;

e\_dept Employee.dept%type;

e\_salary Employee.salary%type;

ex\_invalid\_salary EXCEPTION;

BEGIN

IF e\_salary <= 0 THEN

RAISE ex\_invalid\_salary;

ELSE

SELECT id, dept, salary INTO e\_id, e\_dept, e\_salary

FROM Employee

WHERE id = e\_id;

DBMS\_OUTPUT.PUT\_LINE ('Dept: ' || e\_dept);

DBMS\_OUTPUT.PUT\_LINE ('Salary: ' || e\_salary);

END IF;

EXCEPTION

WHEN ex\_invalid\_salary THEN

dbms\_output.put\_line('Salary must be greater than zero!');

WHEN no\_data\_found THEN

dbms\_output.put\_line('No such customer!');

WHEN others THEN

dbms\_output.put\_line('Error!');

END;

CREATE OR REPLACE TRIGGER display\_salary\_changes

BEFORE DELETE OR INSERT OR UPDATE ON Employee

FOR EACH ROW

WHEN (NEW.ID > 0)

DECLARE

sal\_diff number;

BEGIN

sal\_diff := :NEW.salary - :OLD.salary;

dbms\_output.put\_line('Old salary: ' || :OLD.salary);

dbms\_output.put\_line('New salary: ' || :NEW.salary);

dbms\_output.put\_line('Salary difference: ' || sal\_diff);

END;

create or replace procedure(p\_id int, p\_sub OUTT NUMBER)

AS v\_total INT,

BEGIN

SELECT total INTO v\_total FROM student WHERE std\_id = p\_id;

p\_sub := (v\_total/500 )\*100;

END;

DECLARE

v\_emp id INT := 3;

v\_percentage NUMBER;

BEGIN

calculate\_percentage (v\_std\_id, v\_percentage);

dbms\_output.put\_line('Percentage for std\_id '||v\_emp||':'||v\_percentage||'%');

END;

DECLARE

id INTEGER;

name VARCHAR(20);

dob DATE;

type int\_arr IS VARRAY(5) OF INTEGER;

arr int\_arr;

BEGIN

id:=1;

name:='ABC';

dob:= TO\_DATE('2003-03-04','YYYY-MM-DD');

arr:=int\_arr(1, 2, 3, 4, 5);

dbms\_output.put\_line(id || name || dob);

END;

/

DECLARE

id INTEGER;

name VARCHAR(20);

dob DATE;

type int\_arr IS VARRAY(5) OF INTEGER;

arr int\_arr;

BEGIN

id:=1;

name:='ABC';

dob:= TO\_DATE('2003-03-04','YYYY-MM-DD');

arr:=int\_arr(1, 2, 3, 4, 5);

dbms\_output.put\_line(id);

dbms\_output.put\_line(name);

dbms\_output.put\_line(dob);

END;

/

DECLARE

id INTEGER;

name VARCHAR(20);

dob DATE;

type int\_arr IS VARRAY(5) OF INTEGER;

arr int\_arr;

BEGIN

id:=1;

name:='ABC';

dob:= TO\_DATE('2003-03-04','YYYY-MM-DD');

arr:=int\_arr(1, 2, 3, 4, 5);

dbms\_output.put\_line('ID' || id);

dbms\_output.put\_line(name);

dbms\_output.put\_line(dob);

END;

/

DECLARE

id INTEGER;

name VARCHAR(20);

dob DATE;

type int\_arr IS VARRAY(5) OF INTEGER;

arr int\_arr;

BEGIN

id:=1;

name:='ABC';

dob:= TO\_DATE('2003-03-04','YYYY-MM-DD');

arr:=int\_arr(1, 2, 3, 4, 5);

dbms\_output.put\_line('ID: ' || id);

dbms\_output.put\_line('Name: ' || name);

dbms\_output.put\_line('DOB: ' || dob);

END;

/

DECLARE

id INTEGER;

name VARCHAR(20);

dob DATE;

type int\_arr IS VARRAY(5) OF INTEGER;

arr int\_arr;

BEGIN

id:=1;

name:='ABC';

dob:= TO\_DATE('2003-03-04','YYYY-MM-DD');

arr:=int\_arr(1, 2, 3, 4, 5);

dbms\_output.put\_line('ID: ' || id);

dbms\_output.put\_line('Name: ' || name);

dbms\_output.put\_line('DOB: ' || dob);

FOR i in 1 .. 5 LOOP

dbms\_output.put\_line(arr(i));

END LOOP;

END;

/

DECLARE

n NUMBER := 1634;

r NUMBER;

BEGIN

r := MOD(n, 2);

IF r = 0 THEN

dbms\_output.Put\_line('Even');

ELSE

dbms\_output.Put\_line('Odd');

END IF;

END;

/

DECLARE

n NUMBER := 1634;

r NUMBER;

BEGIN

r := MOD(n, 2);

dbms\_output.Put\_line('The number is: ' || n);

IF r = 0 THEN

dbms\_output.Put\_line(n || ' is Even');

ELSE

dbms\_output.Put\_line(n || ' is Odd');

END IF;

END;

/

DECLARE

n NUMBER := 3154;

r NUMBER;

BEGIN

r := MOD(n, 2);

dbms\_output.Put\_line('The number is: ' || n);

IF r = 0 THEN

dbms\_output.Put\_line(n || ' is Even');

ELSE

dbms\_output.Put\_line(n || ' is Odd');

END IF;

END;

/

DECLARE

type int\_arr IS VARRAY(10) OF INTEGER;

arr int\_arr;

BEGIN

arr:=int\_arr(1, 2, 3, 4, 5, 6, 7, 8, 9, 10);

FOR i in 1 .. 10 LOOP

IF arr(i)<5

THEN

dbms\_output.put\_line('No. ' || arr(i) || ' is in loop');

ELSE

dbms\_output.put\_line('No. ' || arr(i) || ' is not in loop');

END IF;

END LOOP;

END;

/

CREATE TABLE EMPLOYEE (id int, dept varchar(20), salary number);

INSERT INTO EMPLOYEE VALUES (1, 'CSE', 25000);

INSERT INTO EMPLOYEE VALUES (2, 'IT', 50000);

INSERT INTO EMPLOYEE VALUES (3, 'CSE', 75000);

INSERT INTO EMPLOYEE VALUES (4, 'IT', 100000);

INSERT INTO EMPLOYEE VALUES (5, 'CSE', 125000);

BEGIN

dbms\_output.put\_line('Employee ' || 'SELECT id FROM Employee ' );

END;

/

DECLARE

id INTEGER;

name VARCHAR(20);

dob DATE;

type int\_arr IS VARRAY(5) OF INTEGER;

arr int\_arr;

BEGIN

id:=1;

name:='ABC';

dob:= TO\_DATE('2003-03-04','YYYY-MM-DD');

arr:=int\_arr(1, 2, 3, 4, 5);

dbms\_output.put\_line(id || name || dob);

END;

/

DECLARE

id INTEGER;

name VARCHAR(20);

dob DATE;

type int\_arr IS VARRAY(5) OF INTEGER;

arr int\_arr;

BEGIN

id:=1;

name:='ABC';

dob:= TO\_DATE('2003-03-04','YYYY-MM-DD');

arr:=int\_arr(1, 2, 3, 4, 5);

dbms\_output.put\_line(id);

dbms\_output.put\_line(name);

dbms\_output.put\_line(dob);

END;

/

DECLARE

id INTEGER;

name VARCHAR(20);

dob DATE;

type int\_arr IS VARRAY(5) OF INTEGER;

arr int\_arr;

BEGIN

id:=1;

name:='ABC';

dob:= TO\_DATE('2003-03-04','YYYY-MM-DD');

arr:=int\_arr(1, 2, 3, 4, 5);

dbms\_output.put\_line('ID' || id);

dbms\_output.put\_line(name);

dbms\_output.put\_line(dob);

END;

/

DECLARE

id INTEGER;

name VARCHAR(20);

dob DATE;

type int\_arr IS VARRAY(5) OF INTEGER;

arr int\_arr;

BEGIN

id:=1;

name:='ABC';

dob:= TO\_DATE('2003-03-04','YYYY-MM-DD');

arr:=int\_arr(1, 2, 3, 4, 5);

dbms\_output.put\_line('ID: ' || id);

dbms\_output.put\_line('Name: ' || name);

dbms\_output.put\_line('DOB: ' || dob);

END;

/

DECLARE

id INTEGER;

name VARCHAR(20);

dob DATE;

type int\_arr IS VARRAY(5) OF INTEGER;

arr int\_arr;

BEGIN

id:=1;

name:='ABC';

dob:= TO\_DATE('2003-03-04','YYYY-MM-DD');

arr:=int\_arr(1, 2, 3, 4, 5);

dbms\_output.put\_line('ID: ' || id);

dbms\_output.put\_line('Name: ' || name);

dbms\_output.put\_line('DOB: ' || dob);

FOR i in 1 .. 5 LOOP

dbms\_output.put\_line(arr(i));

END LOOP;

END;

/

DECLARE

n NUMBER := 1634;

r NUMBER;

BEGIN

r := MOD(n, 2);

IF r = 0 THEN

dbms\_output.Put\_line('Even');

ELSE

dbms\_output.Put\_line('Odd');

END IF;

END;

/

DECLARE

n NUMBER := 1634;

r NUMBER;

BEGIN

r := MOD(n, 2);

dbms\_output.Put\_line('The number is: ' || n);

IF r = 0 THEN

dbms\_output.Put\_line(n || ' is Even');

ELSE

dbms\_output.Put\_line(n || ' is Odd');

END IF;

END;

/

DECLARE

n NUMBER := 3154;

r NUMBER;

BEGIN

r := MOD(n, 2);

dbms\_output.Put\_line('The number is: ' || n);

IF r = 0 THEN

dbms\_output.Put\_line(n || ' is Even');

ELSE

dbms\_output.Put\_line(n || ' is Odd');

END IF;

END;

/

DECLARE

type int\_arr IS VARRAY(10) OF INTEGER;

arr int\_arr;

BEGIN

arr:=int\_arr(1, 2, 3, 4, 5, 6, 7, 8, 9, 10);

FOR i in 1 .. 10 LOOP

IF arr(i)<5

THEN

dbms\_output.put\_line('No. ' || arr(i) || ' is in loop');

ELSE

dbms\_output.put\_line('No. ' || arr(i) || ' is not in loop');

END IF;

END LOOP;

END;

/

CREATE TABLE EMPLOYEE (id int, dept varchar(20), salary number);

INSERT INTO EMPLOYEE VALUES (1, 'CSE', 25000);

INSERT INTO EMPLOYEE VALUES (2, 'IT', 50000);

INSERT INTO EMPLOYEE VALUES (3, 'CSE', 75000);

INSERT INTO EMPLOYEE VALUES (4, 'IT', 100000);

INSERT INTO EMPLOYEE VALUES (5, 'CSE', 125000);

BEGIN

dbms\_output.put\_line('Employee ' || 'SELECT id FROM Employee ' );

END;

/

begin

for i in(select id, dept, salary from employee) loop

dbms\_output.put\_line('Employee'||' '||i.id||' '||'from'||' '||i.dept||' '||'earns'||' '||i.salary);

end loop;

end;

/

SELECT \* FROM Employee;

declare

length int;

begin

for i in(select length(dept) as length, dept, id from employee) loop

dbms\_output.put\_line('Salary length of Employee '||i.id||' from '||i.dept||' is '||i.length);

end loop;

end;

/

declare

length int;

begin

for i in(select length(salary) as length, dept, id from employee) loop

dbms\_output.put\_line('Salary length of Employee '||i.id||' from '||i.dept||' is '||i.length);

end loop;

end;

/

DECLARE

id INTEGER;

name VARCHAR(20);

dob DATE;

type int\_arr IS VARRAY(5) OF INTEGER;

arr int\_arr;

BEGIN

id:=1;

name:='ABC';

dob:= TO\_DATE('2003-03-04','YYYY-MM-DD');

arr:=int\_arr(1, 2, 3, 4, 5);

dbms\_output.put\_line(id || name || dob);

END;

/

DECLARE

id INTEGER;

name VARCHAR(20);

dob DATE;

type int\_arr IS VARRAY(5) OF INTEGER;

arr int\_arr;

BEGIN

id:=1;

name:='ABC';

dob:= TO\_DATE('2003-03-04','YYYY-MM-DD');

arr:=int\_arr(1, 2, 3, 4, 5);

dbms\_output.put\_line(id);

dbms\_output.put\_line(name);

dbms\_output.put\_line(dob);

END;

/

DECLARE

id INTEGER;

name VARCHAR(20);

dob DATE;

type int\_arr IS VARRAY(5) OF INTEGER;

arr int\_arr;

BEGIN

id:=1;

name:='ABC';

dob:= TO\_DATE('2003-03-04','YYYY-MM-DD');

arr:=int\_arr(1, 2, 3, 4, 5);

dbms\_output.put\_line('ID' || id);

dbms\_output.put\_line(name);

dbms\_output.put\_line(dob);

END;

/

DECLARE

id INTEGER;

name VARCHAR(20);

dob DATE;

type int\_arr IS VARRAY(5) OF INTEGER;

arr int\_arr;

BEGIN

id:=1;

name:='ABC';

dob:= TO\_DATE('2003-03-04','YYYY-MM-DD');

arr:=int\_arr(1, 2, 3, 4, 5);

dbms\_output.put\_line('ID: ' || id);

dbms\_output.put\_line('Name: ' || name);

dbms\_output.put\_line('DOB: ' || dob);

END;

/

DECLARE

id INTEGER;

name VARCHAR(20);

dob DATE;

type int\_arr IS VARRAY(5) OF INTEGER;

arr int\_arr;

BEGIN

id:=1;

name:='ABC';

dob:= TO\_DATE('2003-03-04','YYYY-MM-DD');

arr:=int\_arr(1, 2, 3, 4, 5);

dbms\_output.put\_line('ID: ' || id);

dbms\_output.put\_line('Name: ' || name);

dbms\_output.put\_line('DOB: ' || dob);

FOR i in 1 .. 5 LOOP

dbms\_output.put\_line(arr(i));

END LOOP;

END;

/

DECLARE

n NUMBER := 1634;

r NUMBER;

BEGIN

r := MOD(n, 2);

IF r = 0 THEN

dbms\_output.Put\_line('Even');

ELSE

dbms\_output.Put\_line('Odd');

END IF;

END;

/

DECLARE

n NUMBER := 1634;

r NUMBER;

BEGIN

r := MOD(n, 2);

dbms\_output.Put\_line('The number is: ' || n);

IF r = 0 THEN

dbms\_output.Put\_line(n || ' is Even');

ELSE

dbms\_output.Put\_line(n || ' is Odd');

END IF;

END;

/

DECLARE

n NUMBER := 3154;

r NUMBER;

BEGIN

r := MOD(n, 2);

dbms\_output.Put\_line('The number is: ' || n);

IF r = 0 THEN

dbms\_output.Put\_line(n || ' is Even');

ELSE

dbms\_output.Put\_line(n || ' is Odd');

END IF;

END;

/

DECLARE

type int\_arr IS VARRAY(10) OF INTEGER;

arr int\_arr;

BEGIN

arr:=int\_arr(1, 2, 3, 4, 5, 6, 7, 8, 9, 10);

FOR i in 1 .. 10 LOOP

IF arr(i)<5

THEN

dbms\_output.put\_line('No. ' || arr(i) || ' is in loop');

ELSE

dbms\_output.put\_line('No. ' || arr(i) || ' is not in loop');

END IF;

END LOOP;

END;

/

CREATE TABLE EMPLOYEE (id int, dept varchar(20), salary number);

INSERT INTO EMPLOYEE VALUES (1, 'CSE', 25000);

INSERT INTO EMPLOYEE VALUES (2, 'IT', 50000);

INSERT INTO EMPLOYEE VALUES (3, 'CSE', 75000);

INSERT INTO EMPLOYEE VALUES (4, 'IT', 100000);

INSERT INTO EMPLOYEE VALUES (5, 'CSE', 125000);

BEGIN

dbms\_output.put\_line('Employee ' || 'SELECT id FROM Employee ' );

END;

/

DECLARE

e\_id Employee.id%type;

e\_dept Employee.dept%type;

e\_salary Employee.salary%type;

BEGIN

SELECT id, dept, salary INTO e\_id, e\_dept, e\_salary

FROM Employee

WHERE id = e\_id;

DBMS\_OUTPUT.PUT\_LINE ('Dept: ' || e\_dept);

DBMS\_OUTPUT.PUT\_LINE ('Salary: ' || e\_salary);

EXCEPTION

WHEN no\_data\_found THEN

dbms\_output.put\_line('No such customer!');

WHEN others THEN

dbms\_output.put\_line('Error!');

END;

/

DECLARE

e\_id Employee.id%type;

e\_dept Employee.dept%type;

e\_salary Employee.salary%type;

BEGIN

SELECT id, dept, salary INTO e\_id, e\_dept, e\_salary

FROM Employee

WHERE id = e\_id;

DBMS\_OUTPUT.PUT\_LINE ('Dept: ' || e\_dept);

DBMS\_OUTPUT.PUT\_LINE ('Salary: ' || e\_salary);

EXCEPTION

WHEN no\_data\_found THEN

dbms\_output.put\_line('No such employee!');

WHEN others THEN

dbms\_output.put\_line('Error!');

END;

/

DECLARE

e\_id Employee.id%type;

e\_dept Employee.dept%type;

e\_salary Employee.salary%type;

ex\_invalid\_salary EXCEPTION;

BEGIN

IF e\_salary <= 0 THEN

RAISE ex\_invalid\_salary;

ELSE

SELECT id, dept, salary INTO e\_id, e\_dept, e\_salary

FROM Employee

WHERE id = e\_id;

DBMS\_OUTPUT.PUT\_LINE ('Dept: ' || e\_dept);

DBMS\_OUTPUT.PUT\_LINE ('Salary: ' || e\_salary);

END IF;

EXCEPTION

WHEN ex\_invalid\_salary THEN

dbms\_output.put\_line('Salary must be greater than zero!');

END;

/

DECLARE

e\_id Employee.id%type;

e\_dept Employee.dept%type;

e\_salary Employee.salary%type;

ex\_invalid\_salary EXCEPTION;

BEGIN

IF e\_salary <= 0 THEN

RAISE ex\_invalid\_salary;

ELSE

SELECT id, dept, salary INTO e\_id, e\_dept, e\_salary

FROM Employee

WHERE id = e\_id;

DBMS\_OUTPUT.PUT\_LINE ('Dept: ' || e\_dept);

DBMS\_OUTPUT.PUT\_LINE ('Salary: ' || e\_salary);

END IF;

EXCEPTION

WHEN ex\_invalid\_salary THEN

dbms\_output.put\_line('Salary must be greater than zero!');

END;

/

DECLARE

e\_id Employee.id%type;

e\_dept Employee.dept%type;

e\_salary Employee.salary%type;

ex\_invalid\_salary EXCEPTION;

BEGIN

IF e\_salary <= 0 THEN

RAISE ex\_invalid\_salary;

ELSE

SELECT id, dept, salary INTO e\_id, e\_dept, e\_salary

FROM Employee

WHERE id = e\_id;

DBMS\_OUTPUT.PUT\_LINE ('Dept: ' || e\_dept);

DBMS\_OUTPUT.PUT\_LINE ('Salary: ' || e\_salary);

END IF;

EXCEPTION

WHEN ex\_invalid\_salary THEN

dbms\_output.put\_line('Salary must be greater than zero!');

WHEN no\_data\_found THEN

dbms\_output.put\_line('No such customer!');

WHEN others THEN

dbms\_output.put\_line('Error!');

END;

/

DECLARE

e\_id Employee.id%type := 9;

e\_dept Employee.dept%type;

e\_salary Employee.salary%type;

BEGIN

SELECT id, dept, salary INTO e\_id, e\_dept, e\_salary

FROM Employee

WHERE id = e\_id;

DBMS\_OUTPUT.PUT\_LINE ('Dept: ' || e\_dept);

DBMS\_OUTPUT.PUT\_LINE ('Salary: ' || e\_salary);

EXCEPTION

WHEN no\_data\_found THEN

dbms\_output.put\_line('No such employee!');

WHEN others THEN

dbms\_output.put\_line('Error!');

END;

/

DECLARE

e\_id Employee.id%type := 2;

e\_dept Employee.dept%type;

e\_salary Employee.salary%type;

ex\_invalid\_salary EXCEPTION;

BEGIN

IF e\_salary <= 0 THEN

RAISE ex\_invalid\_salary;

ELSE

SELECT id, dept, salary INTO e\_id, e\_dept, e\_salary

FROM Employee

WHERE id = e\_id;

DBMS\_OUTPUT.PUT\_LINE ('Dept: ' || e\_dept);

DBMS\_OUTPUT.PUT\_LINE ('Salary: ' || e\_salary);

END IF;

EXCEPTION

WHEN ex\_invalid\_salary THEN

dbms\_output.put\_line('Salary must be greater than zero!');

WHEN no\_data\_found THEN

dbms\_output.put\_line('No such customer!');

WHEN others THEN

dbms\_output.put\_line('Error!');

END;

/

DECLARE

e\_id Employee.id%type := 1;

e\_dept Employee.dept%type;

e\_salary Employee.salary%type;

ex\_invalid\_salary EXCEPTION;

BEGIN

IF e\_salary <= 50000 THEN

RAISE ex\_invalid\_salary;

ELSE

SELECT id, dept, salary INTO e\_id, e\_dept, e\_salary

FROM Employee

WHERE id = e\_id;

DBMS\_OUTPUT.PUT\_LINE ('Dept: ' || e\_dept);

DBMS\_OUTPUT.PUT\_LINE ('Salary: ' || e\_salary);

END IF;

EXCEPTION

WHEN ex\_invalid\_salary THEN

dbms\_output.put\_line('Salary must be greater than zero!');

WHEN no\_data\_found THEN

dbms\_output.put\_line('No such customer!');

WHEN others THEN

dbms\_output.put\_line('Error!');

END;

/

DECLARE

e\_id Employee.id%type := 3;

e\_dept Employee.dept%type;

e\_salary Employee.salary%type;

ex\_invalid\_salary EXCEPTION;

BEGIN

IF e\_salary <= 50000 THEN

RAISE ex\_invalid\_salary;

ELSE

SELECT id, dept, salary INTO e\_id, e\_dept, e\_salary

FROM Employee

WHERE id = e\_id;

DBMS\_OUTPUT.PUT\_LINE ('Dept: ' || e\_dept);

DBMS\_OUTPUT.PUT\_LINE ('Salary: ' || e\_salary);

END IF;

EXCEPTION

WHEN ex\_invalid\_salary THEN

dbms\_output.put\_line('Salary must be greater than zero!');

WHEN no\_data\_found THEN

dbms\_output.put\_line('No such customer!');

WHEN others THEN

dbms\_output.put\_line('Error!');

END;

/

DECLARE

e\_id Employee.id%type := 0;

e\_dept Employee.dept%type;

e\_salary Employee.salary%type;

ex\_invalid\_salary EXCEPTION;

BEGIN

IF e\_id <= 0 THEN

RAISE ex\_invalid\_salary;

ELSE

SELECT id, dept, salary INTO e\_id, e\_dept, e\_salary

FROM Employee

WHERE id = e\_id;

DBMS\_OUTPUT.PUT\_LINE ('Dept: ' || e\_dept);

DBMS\_OUTPUT.PUT\_LINE ('Salary: ' || e\_salary);

END IF;

EXCEPTION

WHEN ex\_invalid\_salary THEN

dbms\_output.put\_line('Salary must be greater than zero!');

WHEN no\_data\_found THEN

dbms\_output.put\_line('No such customer!');

WHEN others THEN

dbms\_output.put\_line('Error!');

END;

/

DECLARE

e\_id Employee.id%type := 0;

e\_dept Employee.dept%type;

e\_salary Employee.salary%type;

ex\_invalid\_salary EXCEPTION;

BEGIN

IF e\_id <= 0 THEN

RAISE ex\_invalid\_salary;

ELSE

SELECT id, dept, salary INTO e\_id, e\_dept, e\_salary

FROM Employee

WHERE id = e\_id;

DBMS\_OUTPUT.PUT\_LINE ('Dept: ' || e\_dept);

DBMS\_OUTPUT.PUT\_LINE ('Salary: ' || e\_salary);

END IF;

EXCEPTION

WHEN ex\_invalid\_salary THEN

dbms\_output.put\_line('ID must be greater than zero!');

WHEN no\_data\_found THEN

dbms\_output.put\_line('No such customer!');

WHEN others THEN

dbms\_output.put\_line('Error!');

END;

/

CREATE OR REPLACE PROCEDURE Hike

AS

BEGIN

dbms\_output.put\_line('Hello World!');

END;

/

DECLARE

e\_id Employee.id%type := 1;

e\_dept Employee.dept%type;

e\_salary Employee.salary%type;

CURSOR c\_emp is

SELECT id, dept, salary FROM Employee;

BEGIN

OPEN c\_emp;

LOOP

FETCH c\_emp into e\_id, e\_dept, e\_salary;

EXIT WHEN c\_emp%notfound;

dbms\_output.put\_line(e\_id || ' ' || e\_dept || ' ' || e\_salary);

END LOOP;

CLOSE c\_emp;

END;

/

CREATE OR REPLACE PROCEDURE greetings

AS

BEGIN

dbms\_output.put\_line('Hello World!');

END;

/

EXEC greetings

CREATE OR REPLACE TRIGGER display\_salary\_changes

BEFORE DELETE OR INSERT OR UPDATE ON Employee

FOR EACH ROW

WHEN (NEW.ID > 0)

DECLARE

sal\_diff number;

BEGIN

sal\_diff := :NEW.salary - :OLD.salary;

dbms\_output.put\_line('Old salary: ' || :OLD.salary);

dbms\_output.put\_line('New salary: ' || :NEW.salary);

dbms\_output.put\_line('Salary difference: ' || sal\_diff);

END;

/

SELECT \* FROM Employee;

INSERT INTO Employee VALUES (6, 'IT', 150000 );

UPDATE Employee SET salary=salary+1000;

CREATE TABLE student (emp\_id INT PRIMARY KEY,total INT,no\_sub INT);

CREATE OR REPLACE PROCEDURE calculate\_percentage(

p\_id INT,

sub out number

)

AS

v\_total INT;

BEGIN

SELECT total, no\_sub INTO v\_total, sub FROM student WHERE emp\_id = p\_id;

sub := ((v\_total / 500)\*100);

END;

DECLARE

v\_emp\_id INT := 3176;

v\_percentage NUMBER;

BEGIN

calculate\_percentage(v\_emp\_id, v\_percentage);

DBMS\_OUTPUT.PUT\_LINE('Percentage for emp\_id ' || v\_emp\_id || ': ' || v\_percentage || '%');

END;