**5. Implementing Access Control in Relational Database**

**AIM:**

To Implement access control in relational databases is crucial for ensuring the security and integrity of data.

**ALGORITHM:**

 **Download MS SQL Server**: Obtain the latest version of MS SQL Server from the official Microsoft website.

 **Table Creation**: Define a table to store customer data.

 **Security Predicate Function**: Implement a function that filters rows based on the current user.

 **Security Policy**: Apply the predicate function to enforce row-level security.

 **Create Users**: Define users with login credentials.

 **Grant Permissions**: Assign access permissions to the users for the table.

 **Policy Enforcement**: Ensure each user can only view or modify their own data based on their name.

**PROGRAM:**

Role-Based Access Control (RBAC)

Row-Level Security

Step 2: Create a new table with a security policy

CREATE TABLE Customers (

CustomerID int,

Name varchar(50),

Email varchar(100)

);

Step 3: Create a Security Predicate Function

CREATE FUNCTION dbo.fn\_securitypredicate(@Name AS varchar(50))

RETURNS TABLE

WITH SCHEMABINDING

AS

RETURN SELECT 1 AS result

WHERE @Name = USER\_NAME();

Step 4: Create a Security Policy

CREATE SECURITY POLICY CustomerSecurityPolicy

ADD FILTER PREDICATE dbo.fn\_securitypredicate(Name) ON dbo.Customers

WITH (STATE = ON);

Step 5: Create the User1

CREATE LOGIN user1 WITH PASSWORD = 'root@123';

CREATE USER user1 FOR LOGIN user1;

Step 5: Create the User2

CREATE LOGIN user2 WITH PASSWORD = 'root@456';

CREATE USER user2 FOR LOGIN user2;

Step 6: Grant Permissions

GRANT SELECT, INSERT, UPDATE ON dbo.Customers TO user1;

Step 6: Grant Permissions

GRANT SELECT, INSERT, UPDATE ON dbo.Customers TO user2;

**OUTPUT**:

-- Create roles

CREATE ROLE manager;

CREATE ROLE employee;

-- Grant privileges

GRANT SELECT, UPDATE ON employees TO manager;

GRANT SELECT ON employees TO employee;

-- Assign roles to users

GRANT manager TO 'user1';

GRANT employee TO 'user2';

-- Grant privileges to specific users

GRANT SELECT, INSERT ON employees TO 'user3';

-- Revoke access

REVOKE INSERT ON employees FROM 'user3';

BEGIN

DBMS\_RLS.ADD\_POLICY(

object\_schema => 'HR',

object\_name => 'employees',

policy\_name => 'dept\_policy',

function\_schema => 'HR',

policy\_function => 'department\_policy');

END;

-- Create an encrypted table in MySQL

CREATE TABLE sensitive\_data (

id INT,

name VARCHAR(255) NOT NULL,

ssn VARBINARY(255) NOT NULL

) ENCRYPTION='Y';

-- Encrypt and decrypt data

INSERT INTO sensitive\_data (id, name, ssn)

VALUES (1, 'John Doe', AES\_ENCRYPT('123-45-6789', 'encryption\_key'));

SELECT id, name, AES\_DECRYPT(ssn, 'encryption\_key') AS ssn FROM sensitive\_data;

END;

-- Enable auditing in Oracle

AUDIT SELECT TABLE, INSERT TABLE, DELETE TABLE BY ACCESS;

-- View audit trail

SELECT \* FROM DBA\_AUDIT\_TRAIL;

**RESULT:**

Hence the implementation of access control in relational databases is crucial for ensuring the security and integrity of data is executed successfully.