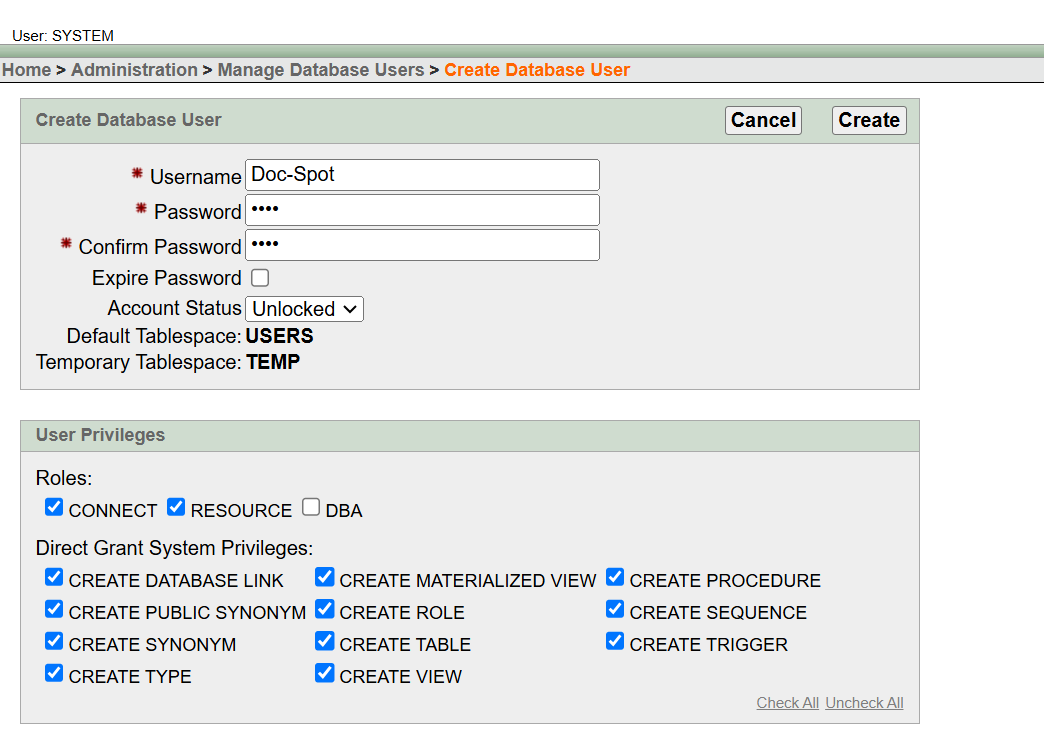


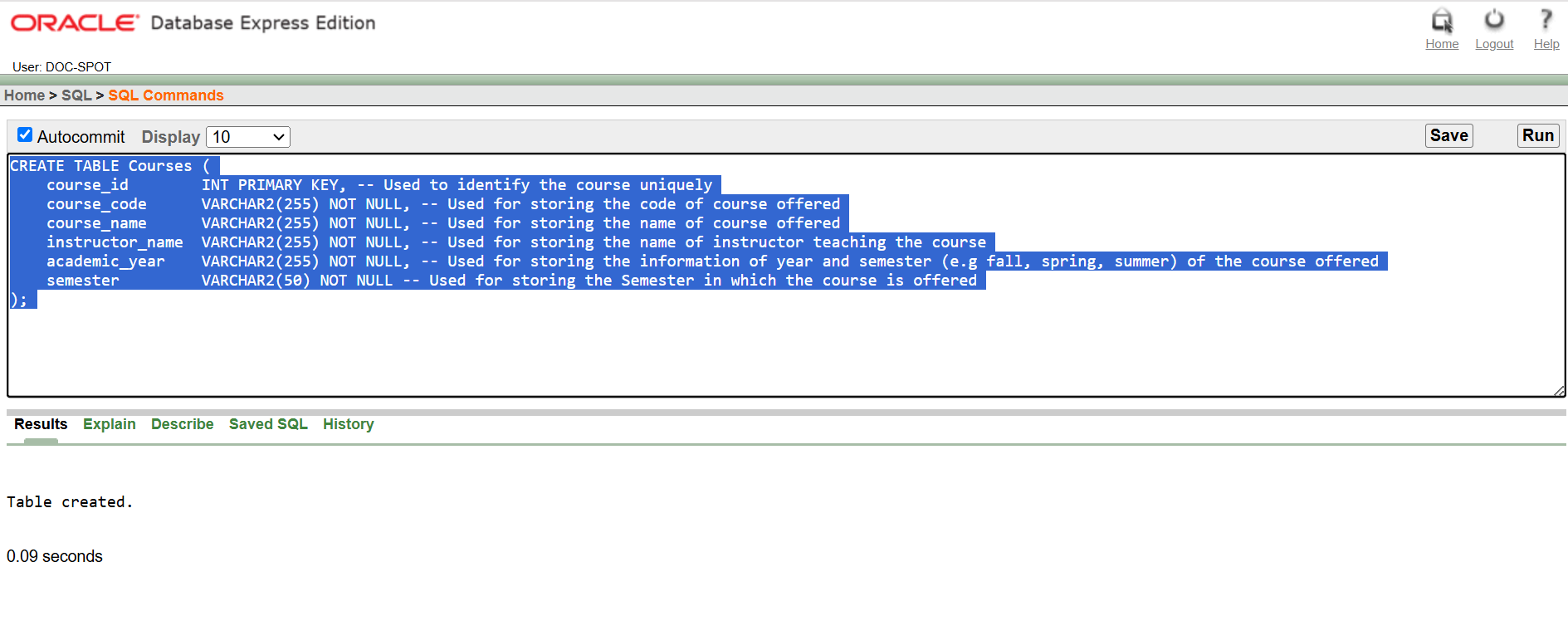
**Project Members:**

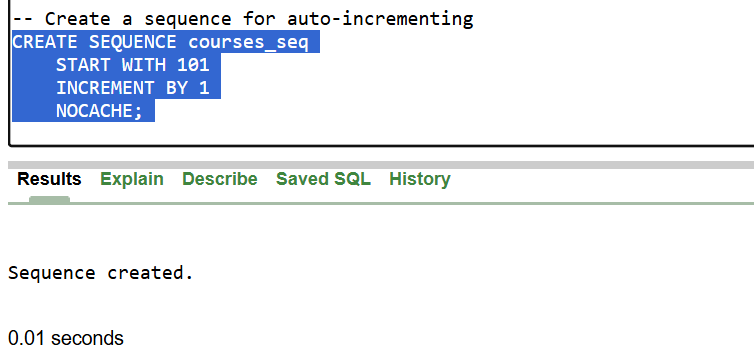
       **Muhammad Abdullah (53457)**

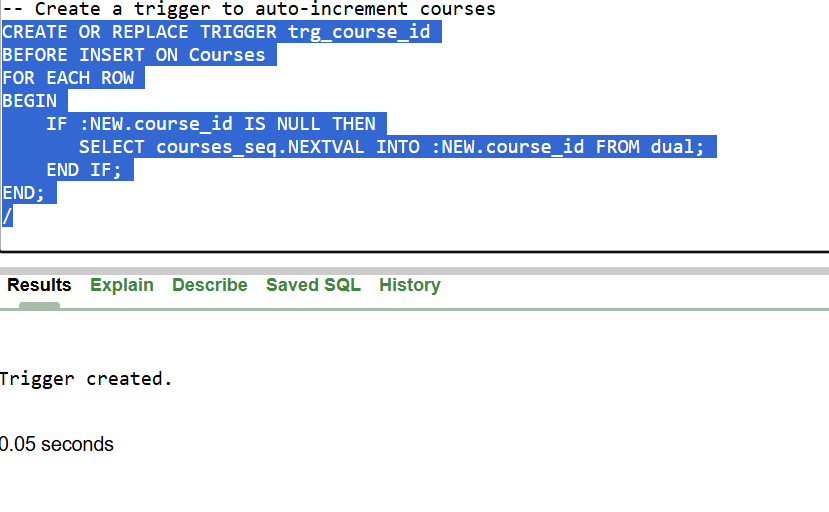
**Creating Schema User**



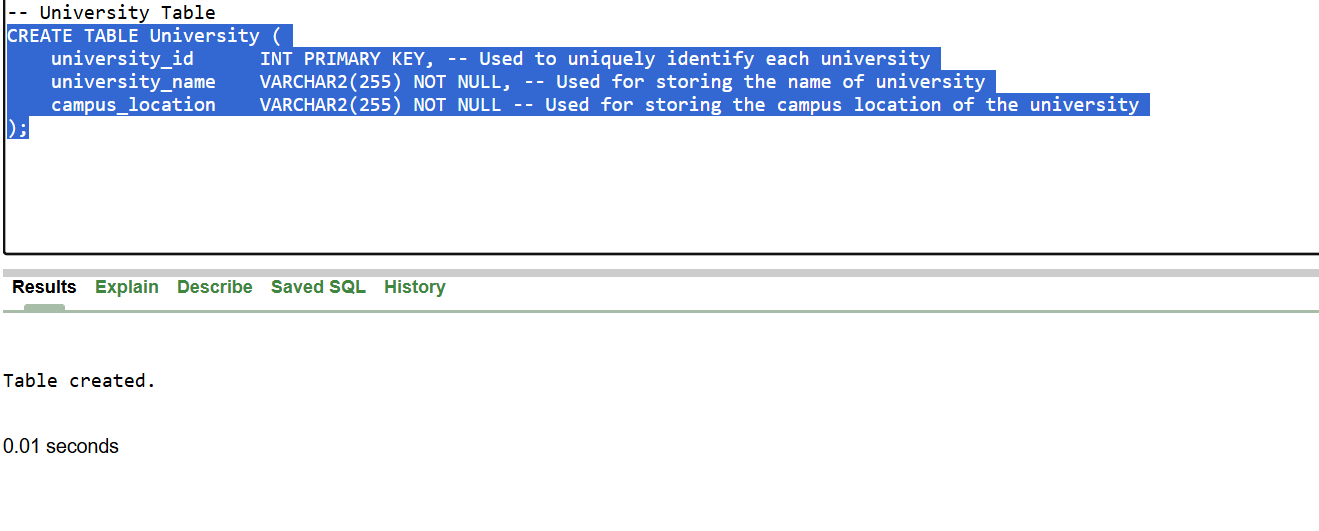
1. **Courses Table**

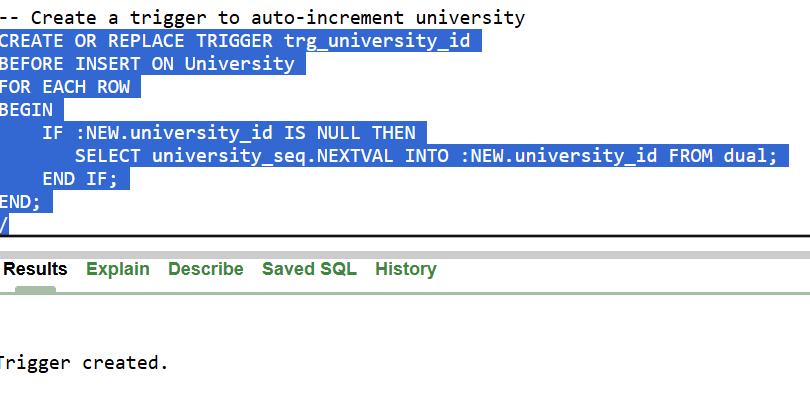




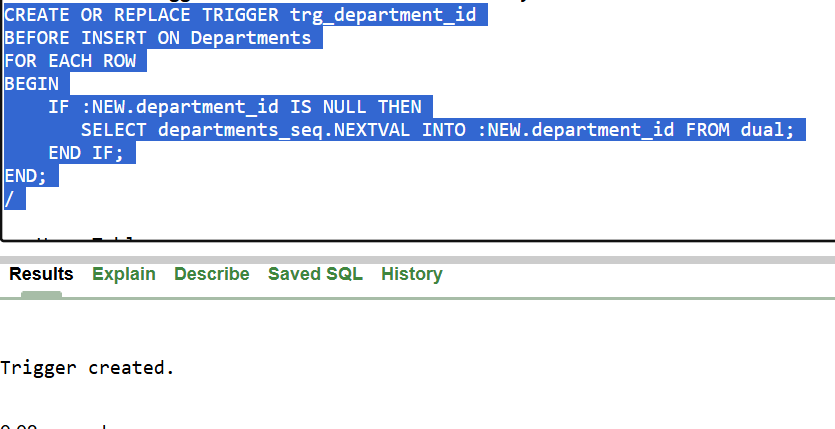
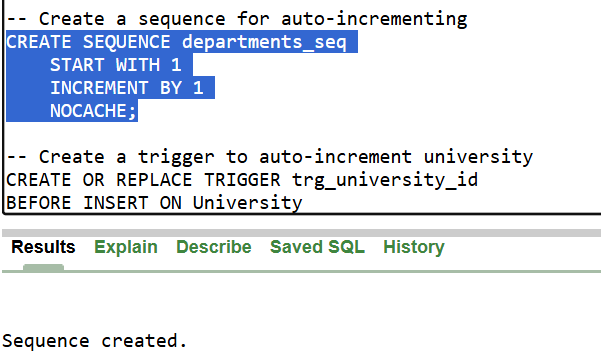
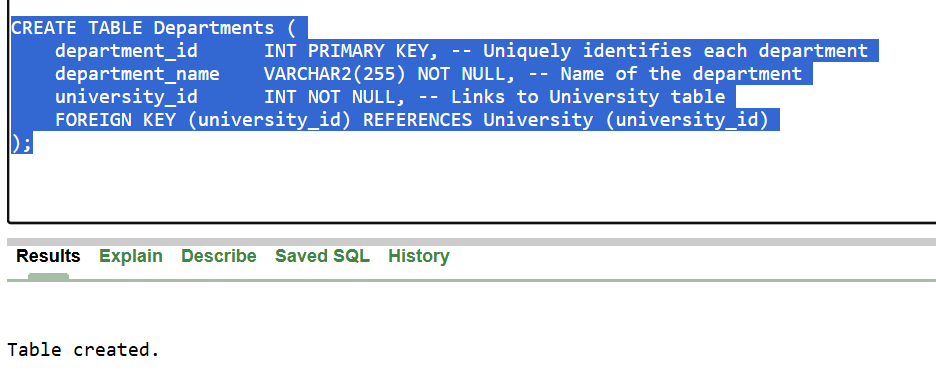


1. **University Table**

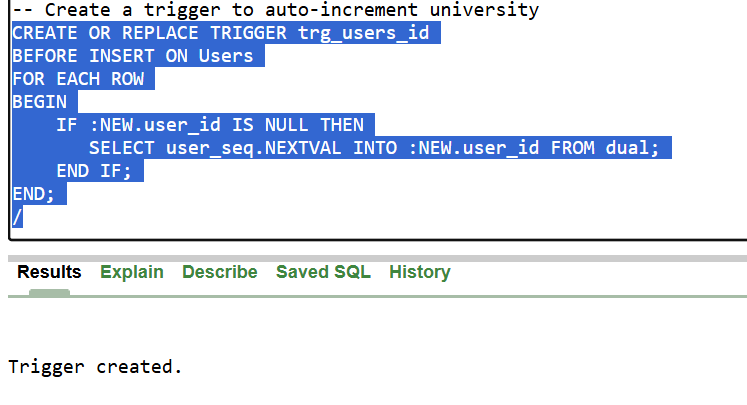
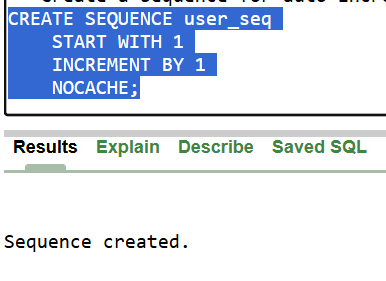
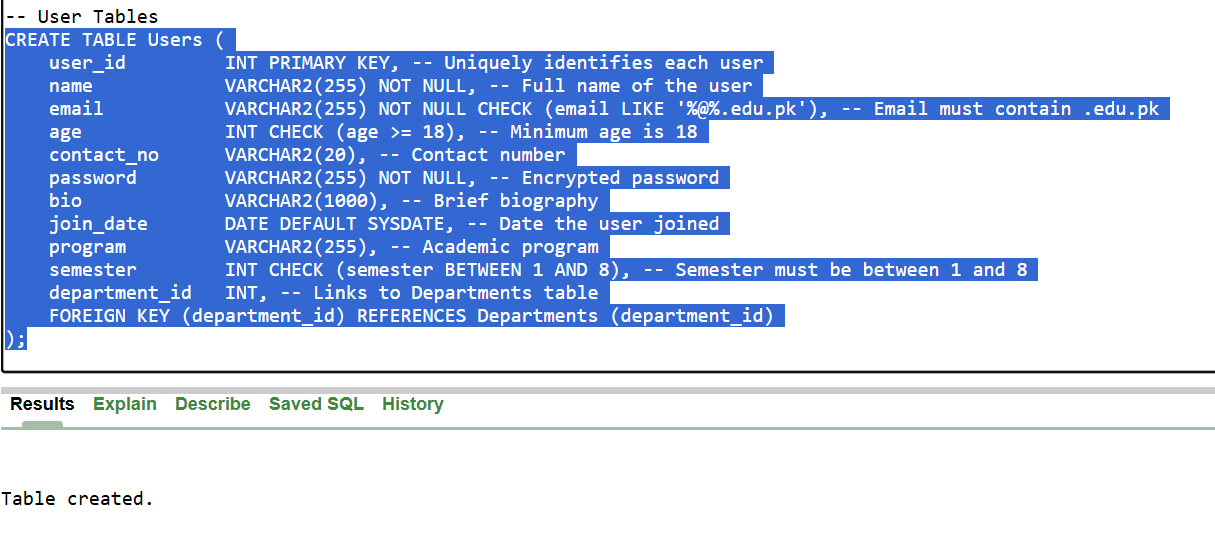




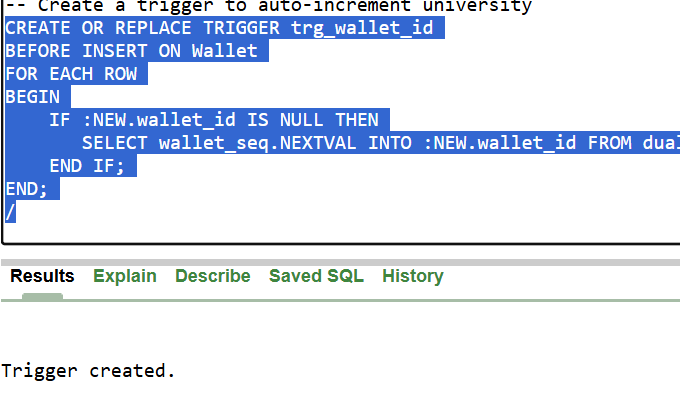
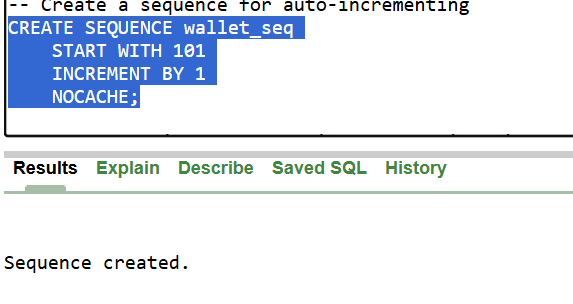
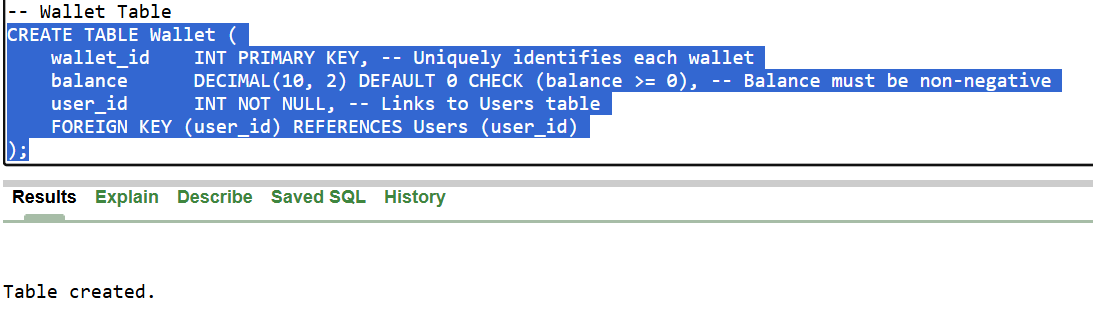
1. **Departments Table**

****

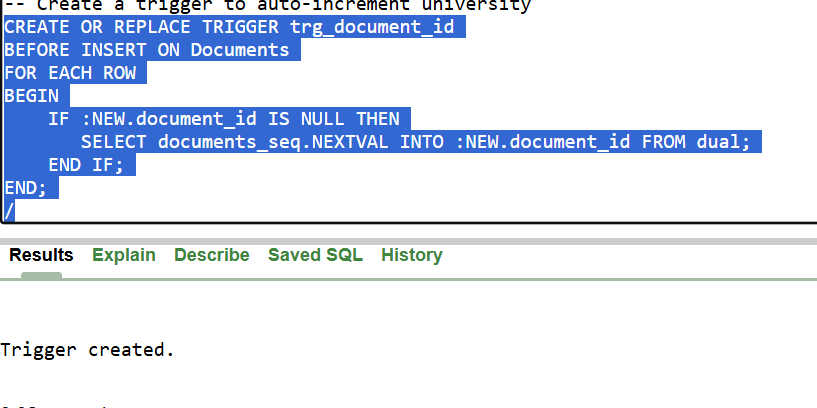
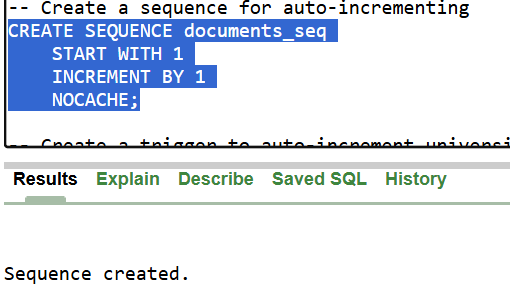
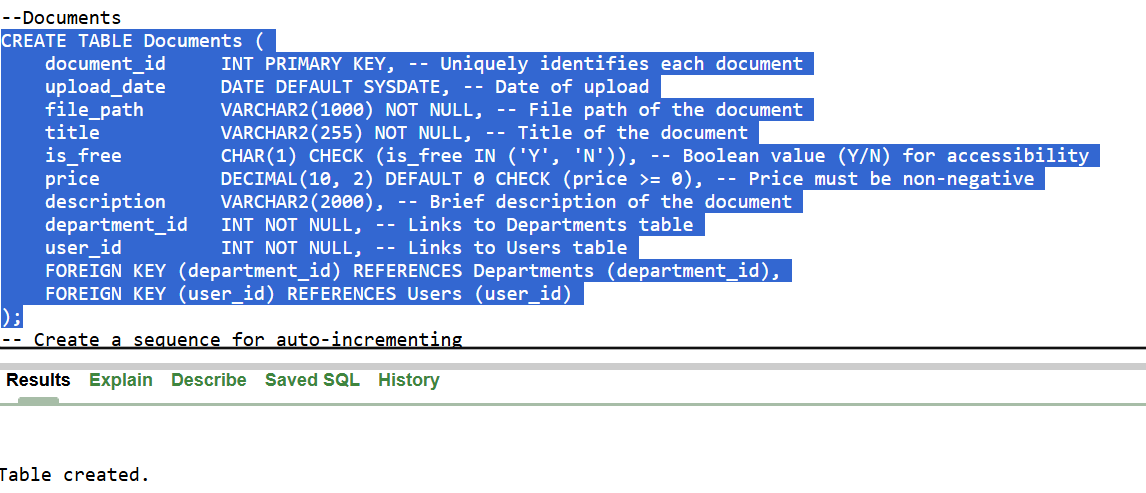
1. **Users Table**

****

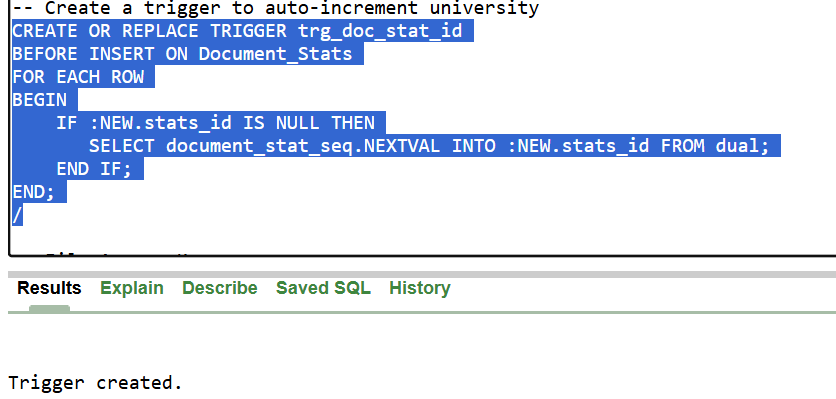
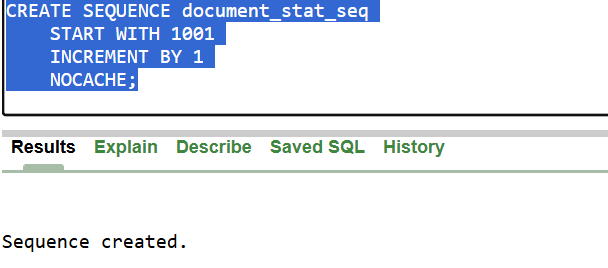
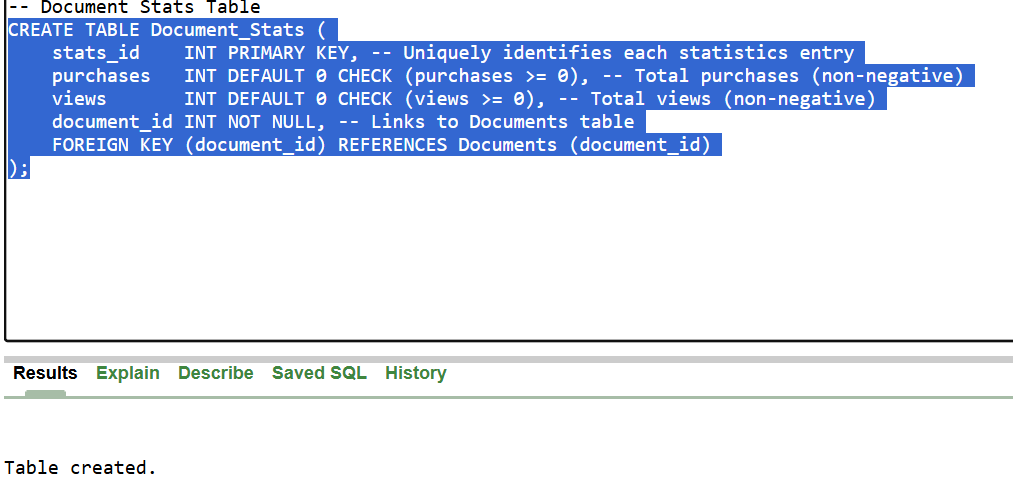
1. **Wallet Table**

****

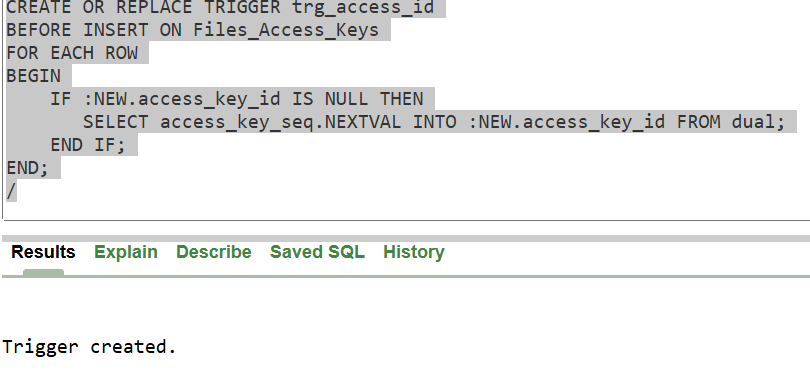
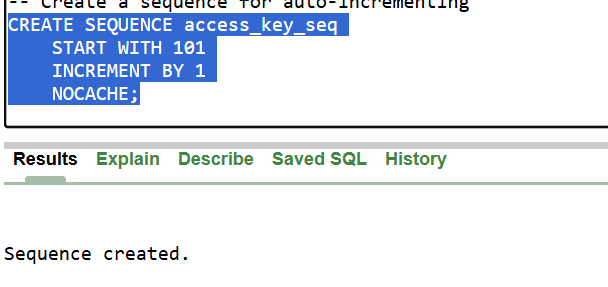
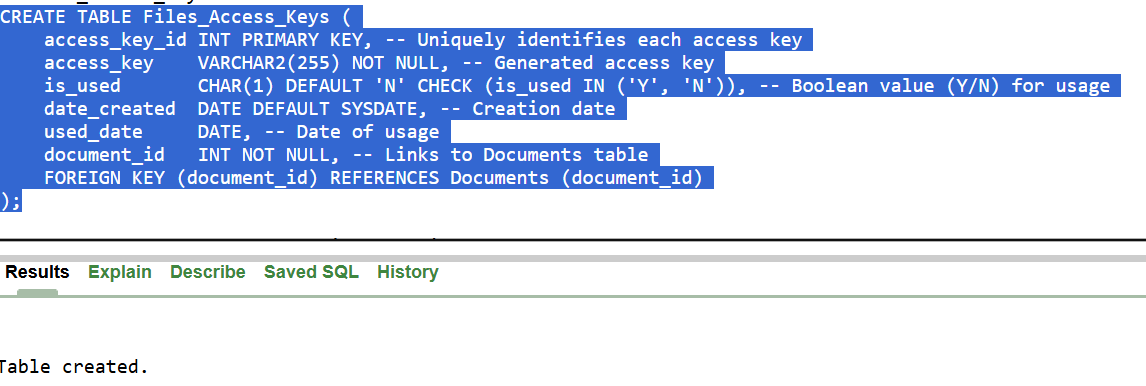
1. **Documents Table**

****

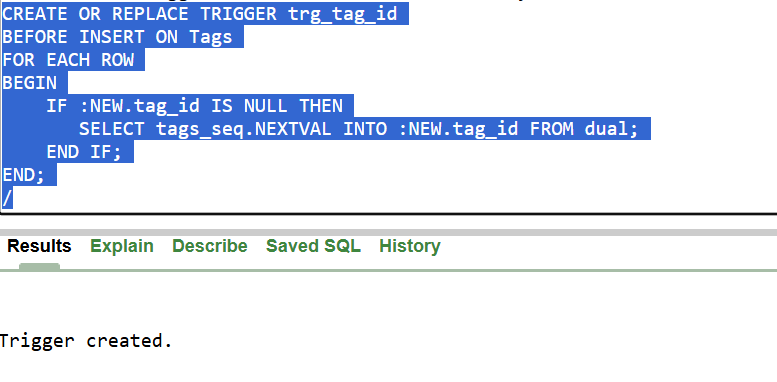
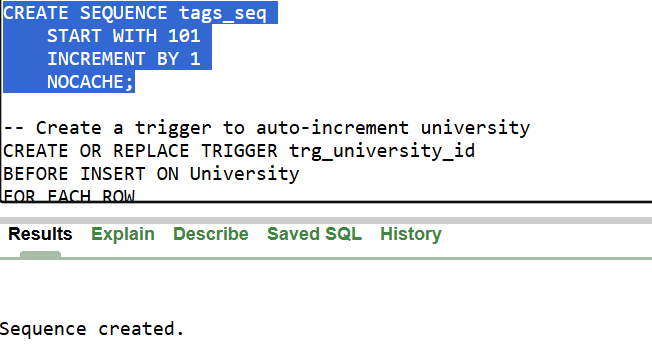
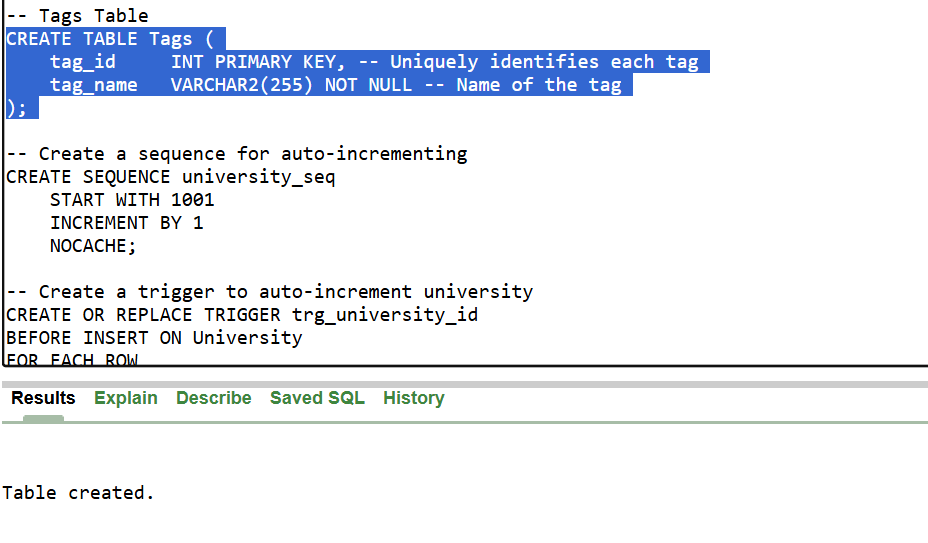
1. **Document\_Stats Table**

****

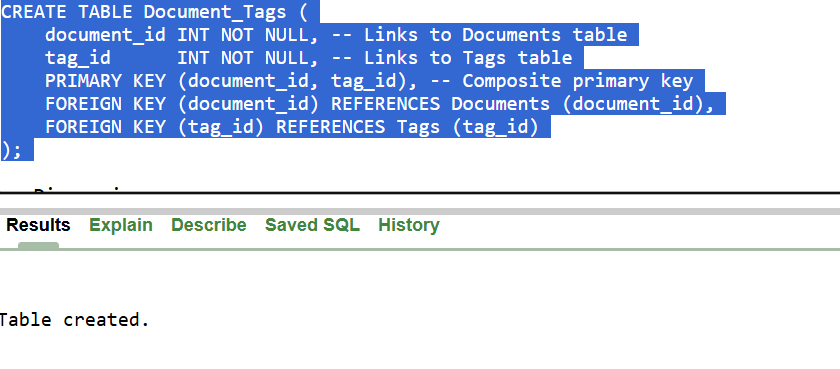
1. **File\_Access\_Keys Table**

****

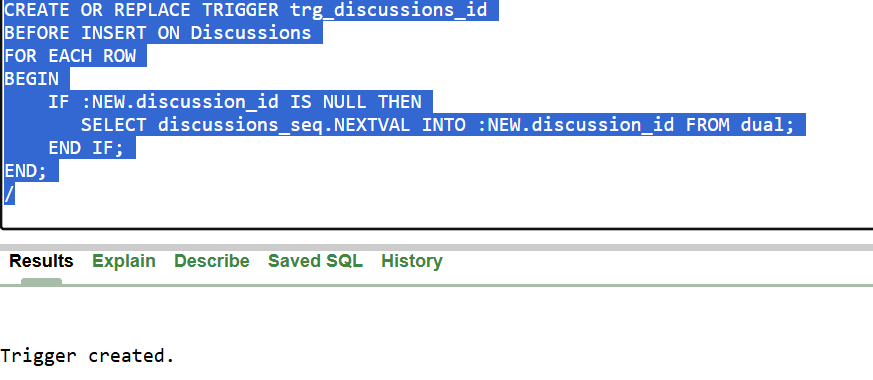
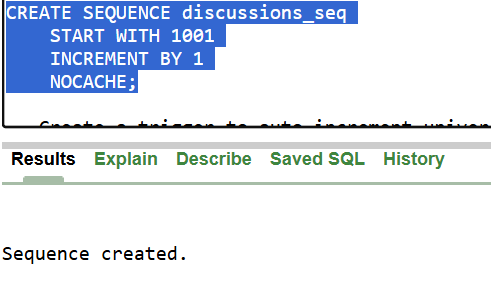
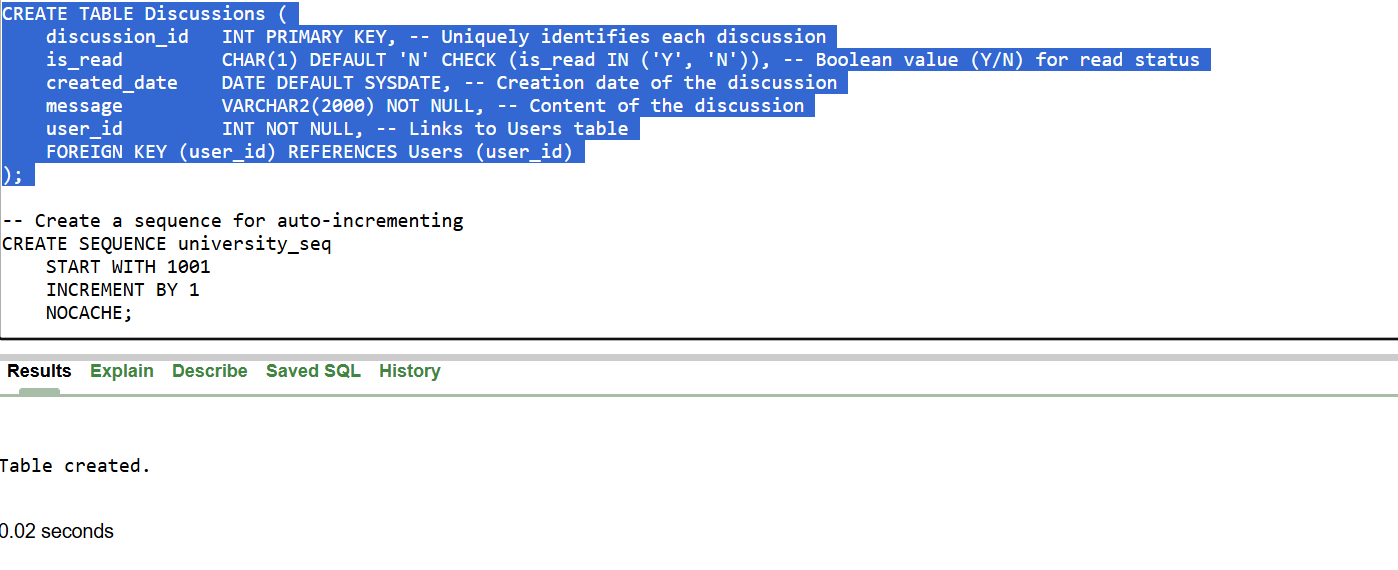
1. **Tags Table**

****

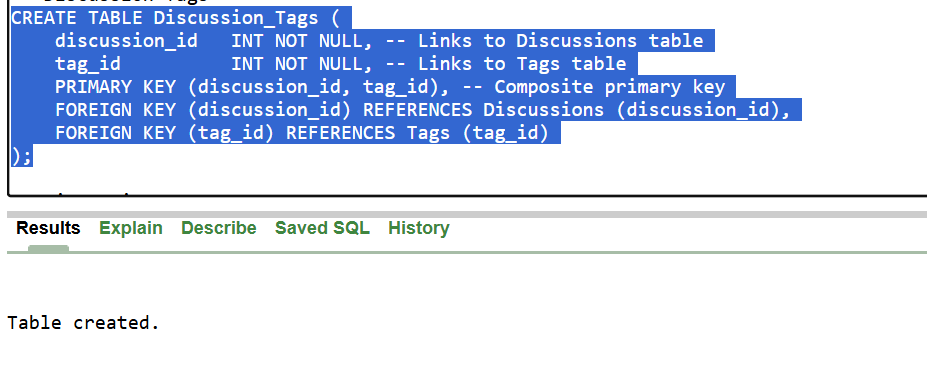
1. **Documet\_Tags Table**

****

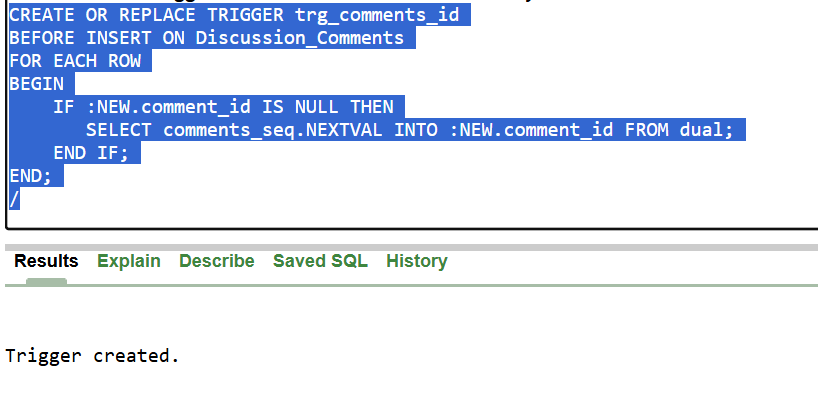
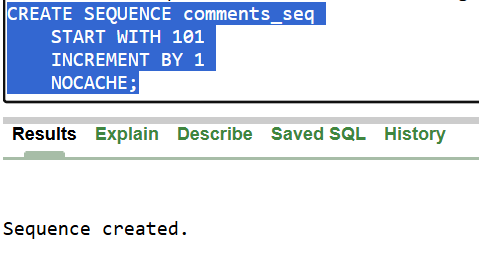
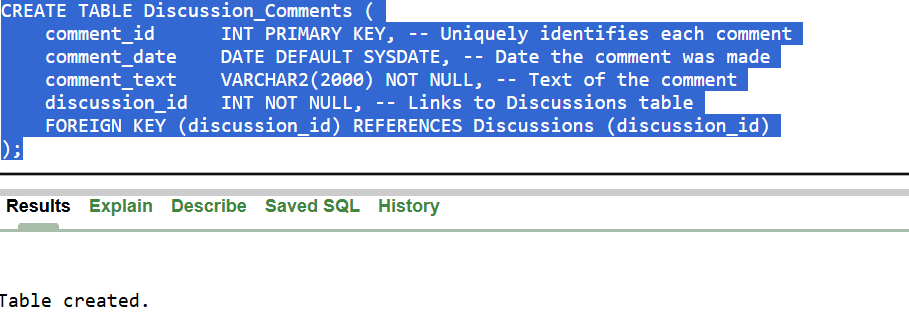
1. **Discussions Table**

****

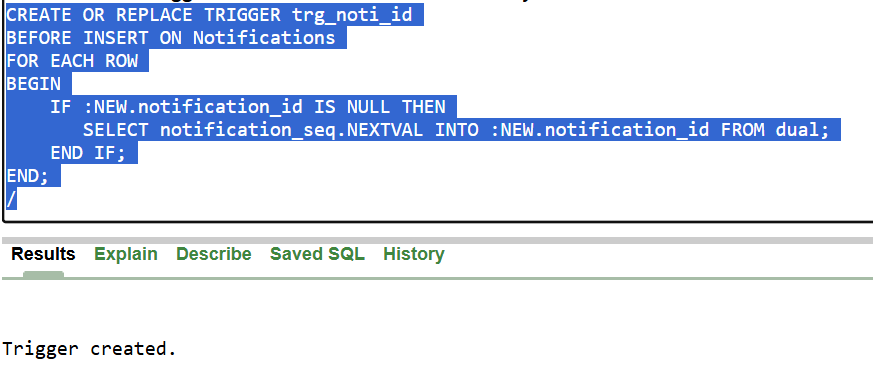
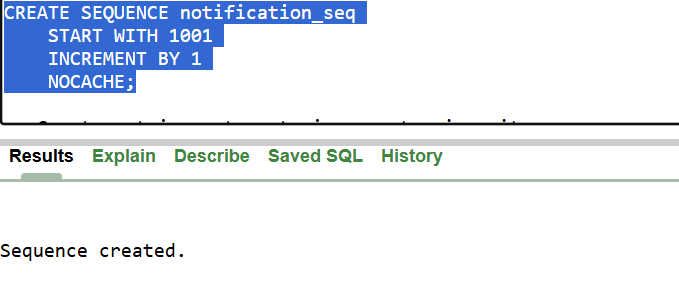
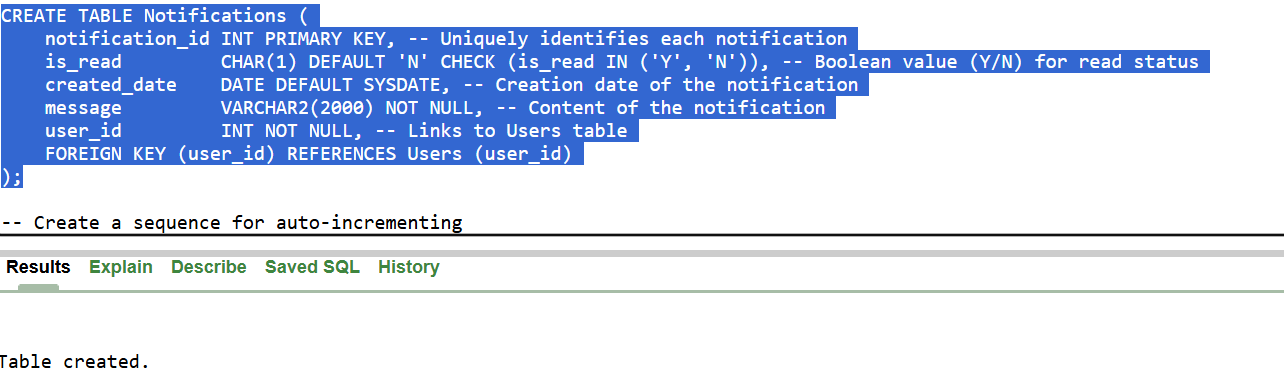
1. **Discussion\_Tags Table**

****

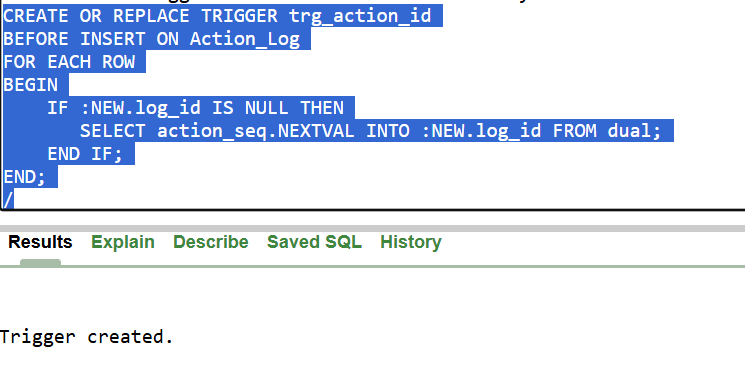
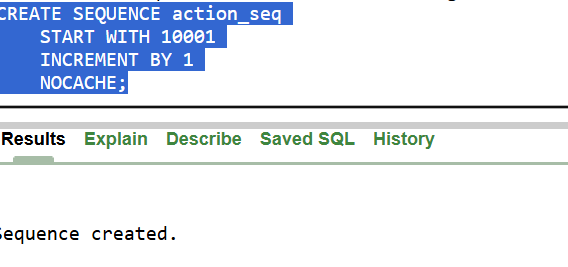
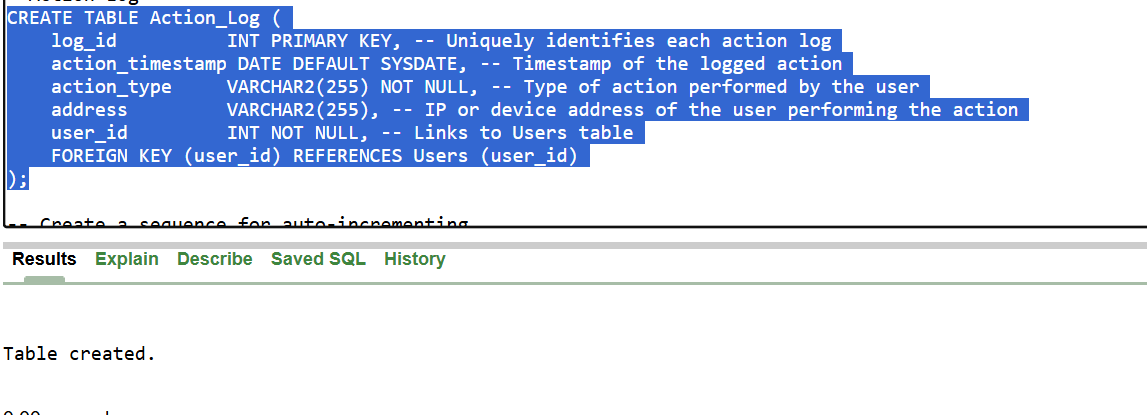
1. **Discussion\_Comments Table**

****

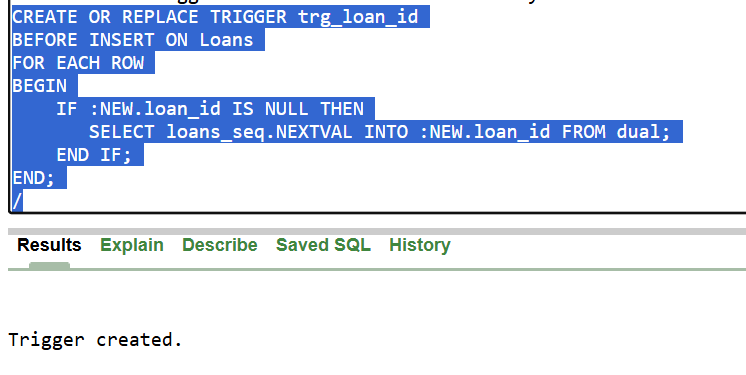
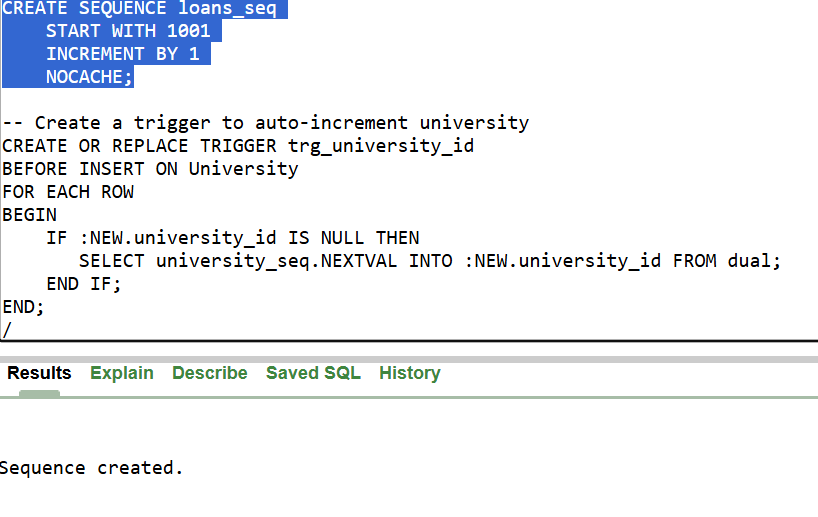
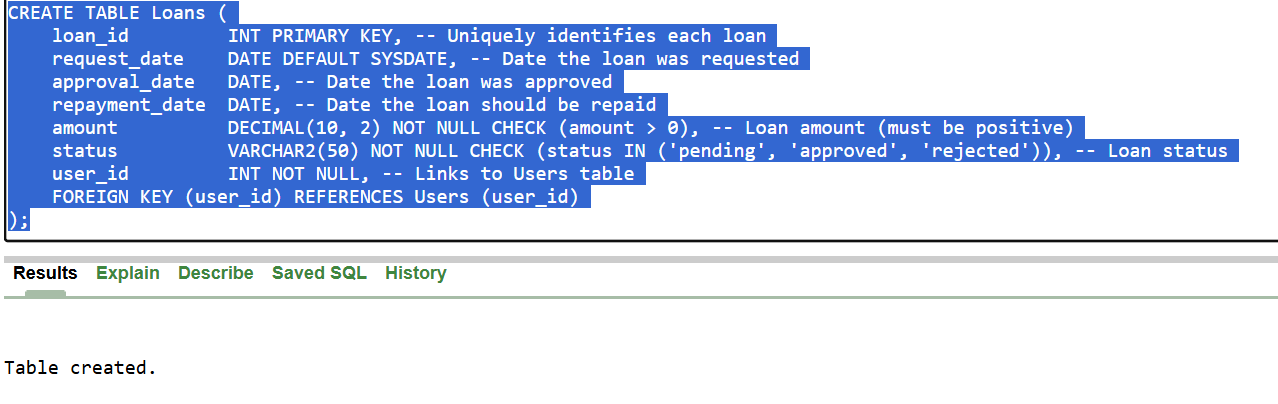
1. **Notifications Table**

****

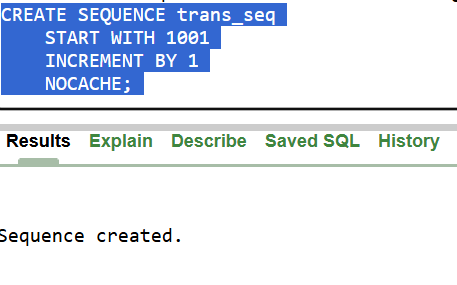
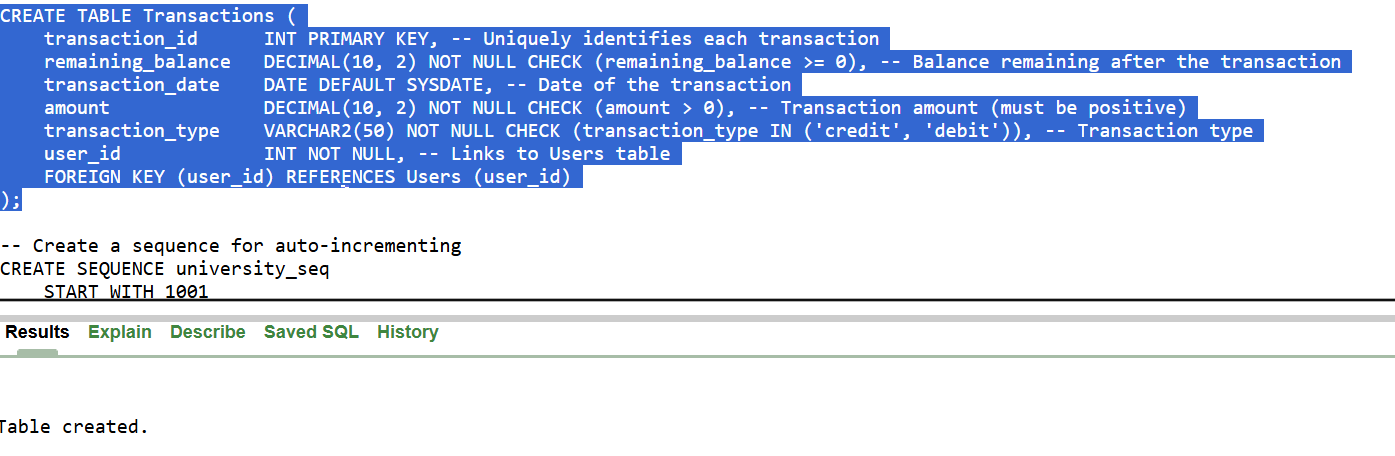
1. **Action\_Log Table**

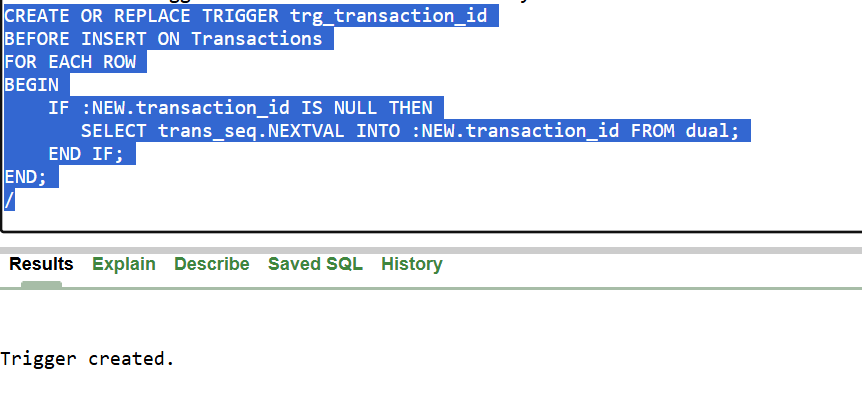
****

1. **Loans Table**

****

1. **Transactions Table**

****

****

1. **Code**

|  |
| --- |
| ----Courses Table  CREATE TABLE Courses (  course\_id INT PRIMARY KEY, -- Used to identify the course uniquely  course\_code VARCHAR2(255) NOT NULL, -- Used for storing the code of course offered  course\_name VARCHAR2(255) NOT NULL, -- Used for storing the name of course offered  instructor\_name VARCHAR2(255) NOT NULL, -- Used for storing the name of instructor teaching the course  academic\_year VARCHAR2(255) NOT NULL, -- Used for storing the information of year and semester (e.g fall, spring, summer) of the course offered  semester VARCHAR2(50) NOT NULL -- Used for storing the Semester in which the course is offered  );  -- Create a sequence for auto-incrementing  CREATE SEQUENCE courses\_seq  START WITH 101  INCREMENT BY 1  NOCACHE;  -- Create a trigger to auto-increment courses  CREATE OR REPLACE TRIGGER trg\_course\_id  BEFORE INSERT ON Courses  FOR EACH ROW  BEGIN  IF :NEW.course\_id IS NULL THEN  SELECT courses\_seq.NEXTVAL INTO :NEW.course\_id FROM dual;  END IF;  END;  /  -- University Table  CREATE TABLE University (  university\_id INT PRIMARY KEY, -- Used to uniquely identify each university  university\_name VARCHAR2(255) NOT NULL, -- Used for storing the name of university  campus\_location VARCHAR2(255) NOT NULL -- Used for storing the campus location of the university  );  -- Create a sequence for auto-incrementing  CREATE SEQUENCE university\_seq  START WITH 1001  INCREMENT BY 1  NOCACHE;  -- Create a trigger to auto-increment university  CREATE OR REPLACE TRIGGER trg\_university\_id  BEFORE INSERT ON University  FOR EACH ROW  BEGIN  IF :NEW.university\_id IS NULL THEN  SELECT university\_seq.NEXTVAL INTO :NEW.university\_id FROM dual;  END IF;  END;  /  -- Departments  CREATE TABLE Departments (  department\_id INT PRIMARY KEY, -- Uniquely identifies each department  department\_name VARCHAR2(255) NOT NULL, -- Name of the department  university\_id INT NOT NULL, -- Links to University table  FOREIGN KEY (university\_id) REFERENCES University (university\_id)  );  -- Create a sequence for auto-incrementing  CREATE SEQUENCE departments\_seq  START WITH 1  INCREMENT BY 1  NOCACHE;  -- Create a trigger to auto-increment university  CREATE OR REPLACE TRIGGER trg\_department\_id  BEFORE INSERT ON Departments  FOR EACH ROW  BEGIN  IF :NEW.department\_id IS NULL THEN  SELECT departments\_seq.NEXTVAL INTO :NEW.department\_id FROM dual;  END IF;  END;  /  -- User Tables  CREATE TABLE Users (  user\_id INT PRIMARY KEY, -- Uniquely identifies each user  name VARCHAR2(255) NOT NULL, -- Full name of the user  email VARCHAR2(255) NOT NULL CHECK (email LIKE '%@%.edu.pk'), -- Email must contain .edu.pk  age INT CHECK (age >= 18), -- Minimum age is 18  contact\_no VARCHAR2(20), -- Contact number  password VARCHAR2(255) NOT NULL, -- Encrypted password  bio VARCHAR2(1000), -- Brief biography  join\_date DATE DEFAULT SYSDATE, -- Date the user joined  program VARCHAR2(255), -- Academic program  semester INT CHECK (semester BETWEEN 1 AND 8), -- Semester must be between 1 and 8  department\_id INT, -- Links to Departments table  FOREIGN KEY (department\_id) REFERENCES Departments (department\_id)  );  -- Create a sequence for auto-incrementing  CREATE SEQUENCE user\_seq  START WITH 1  INCREMENT BY 1  NOCACHE;  -- Create a trigger to auto-increment university  CREATE OR REPLACE TRIGGER trg\_users\_id  BEFORE INSERT ON Users  FOR EACH ROW  BEGIN  IF :NEW.user\_id IS NULL THEN  SELECT user\_seq.NEXTVAL INTO :NEW.user\_id FROM dual;  END IF;  END;  /  -- Wallet Table  CREATE TABLE Wallet (  wallet\_id INT PRIMARY KEY, -- Uniquely identifies each wallet  balance DECIMAL(10, 2) DEFAULT 0 CHECK (balance >= 0), -- Balance must be non-negative  user\_id INT NOT NULL, -- Links to Users table  FOREIGN KEY (user\_id) REFERENCES Users (user\_id)  );  -- Create a sequence for auto-incrementing  CREATE SEQUENCE wallet\_seq  START WITH 101  INCREMENT BY 1  NOCACHE;  -- Create a trigger to auto-increment university  CREATE OR REPLACE TRIGGER trg\_wallet\_id  BEFORE INSERT ON Wallet  FOR EACH ROW  BEGIN  IF :NEW.wallet\_id IS NULL THEN  SELECT wallet\_seq.NEXTVAL INTO :NEW.wallet\_id FROM dual;  END IF;  END;  /  --Documents  CREATE TABLE Documents (  document\_id INT PRIMARY KEY, -- Uniquely identifies each document  upload\_date DATE DEFAULT SYSDATE, -- Date of upload  file\_path VARCHAR2(1000) NOT NULL, -- File path of the document  title VARCHAR2(255) NOT NULL, -- Title of the document  is\_free CHAR(1) CHECK (is\_free IN ('Y', 'N')), -- Boolean value (Y/N) for accessibility  price DECIMAL(10, 2) DEFAULT 0 CHECK (price >= 0), -- Price must be non-negative  description VARCHAR2(2000), -- Brief description of the document  department\_id INT NOT NULL, -- Links to Departments table  user\_id INT NOT NULL, -- Links to Users table  FOREIGN KEY (department\_id) REFERENCES Departments (department\_id),  FOREIGN KEY (user\_id) REFERENCES Users (user\_id)  );  -- Create a sequence for auto-incrementing  CREATE SEQUENCE documents\_seq  START WITH 1  INCREMENT BY 1  NOCACHE;  -- Create a trigger to auto-increment university  CREATE OR REPLACE TRIGGER trg\_document\_id  BEFORE INSERT ON Documents  FOR EACH ROW  BEGIN  IF :NEW.document\_id IS NULL THEN  SELECT documents\_seq.NEXTVAL INTO :NEW.document\_id FROM dual;  END IF;  END;  /  -- Document Stats Table  CREATE TABLE Document\_Stats (  stats\_id INT PRIMARY KEY, -- Uniquely identifies each statistics entry  purchases INT DEFAULT 0 CHECK (purchases >= 0), -- Total purchases (non-negative)  views INT DEFAULT 0 CHECK (views >= 0), -- Total views (non-negative)  document\_id INT NOT NULL, -- Links to Documents table  FOREIGN KEY (document\_id) REFERENCES Documents (document\_id)  );  -- Create a sequence for auto-incrementing  CREATE SEQUENCE document\_stat\_seq  START WITH 1001  INCREMENT BY 1  NOCACHE;  -- Create a trigger to auto-increment university  CREATE OR REPLACE TRIGGER trg\_doc\_stat\_id  BEFORE INSERT ON Document\_Stats  FOR EACH ROW  BEGIN  IF :NEW.stats\_id IS NULL THEN  SELECT document\_stat\_seq.NEXTVAL INTO :NEW.stats\_id FROM dual;  END IF;  END;  /  -- File\_Access\_Keys  CREATE TABLE Files\_Access\_Keys (  access\_key\_id INT PRIMARY KEY, -- Uniquely identifies each access key  access\_key VARCHAR2(255) NOT NULL, -- Generated access key  is\_used CHAR(1) DEFAULT 'N' CHECK (is\_used IN ('Y', 'N')), -- Boolean value (Y/N) for usage  date\_created DATE DEFAULT SYSDATE, -- Creation date  used\_date DATE, -- Date of usage  document\_id INT NOT NULL, -- Links to Documents table  FOREIGN KEY (document\_id) REFERENCES Documents (document\_id)  );  -- Create a sequence for auto-incrementing  CREATE SEQUENCE access\_key\_seq  START WITH 101  INCREMENT BY 1  NOCACHE;  -- Create a trigger to auto-increment university  CREATE OR REPLACE TRIGGER trg\_access\_id  BEFORE INSERT ON Files\_Access\_Keys  FOR EACH ROW  BEGIN  IF :NEW.access\_key\_id IS NULL THEN  SELECT access\_key\_seq.NEXTVAL INTO :NEW.access\_key\_id FROM dual;  END IF;  END;  /  -- Tags Table  CREATE TABLE Tags (  tag\_id INT PRIMARY KEY, -- Uniquely identifies each tag  tag\_name VARCHAR2(255) NOT NULL -- Name of the tag  );  -- Create a sequence for auto-incrementing  CREATE SEQUENCE tags\_seq  START WITH 101  INCREMENT BY 1  NOCACHE;  -- Create a trigger to auto-increment university  CREATE OR REPLACE TRIGGER trg\_tag\_id  BEFORE INSERT ON Tags  FOR EACH ROW  BEGIN  IF :NEW.tag\_id IS NULL THEN  SELECT tags\_seq.NEXTVAL INTO :NEW.tag\_id FROM dual;  END IF;  END;  /  -- Document Tags Table  CREATE TABLE Document\_Tags (  document\_id INT NOT NULL, -- Links to Documents table  tag\_id INT NOT NULL, -- Links to Tags table  PRIMARY KEY (document\_id, tag\_id), -- Composite primary key  FOREIGN KEY (document\_id) REFERENCES Documents (document\_id),  FOREIGN KEY (tag\_id) REFERENCES Tags (tag\_id)  );  -- Discussions  CREATE TABLE Discussions (  discussion\_id INT PRIMARY KEY, -- Uniquely identifies each discussion  is\_read CHAR(1) DEFAULT 'N' CHECK (is\_read IN ('Y', 'N')), -- Boolean value (Y/N) for read status  created\_date DATE DEFAULT SYSDATE, -- Creation date of the discussion  message VARCHAR2(2000) NOT NULL, -- Content of the discussion  user\_id INT NOT NULL, -- Links to Users table  FOREIGN KEY (user\_id) REFERENCES Users (user\_id)  );  -- Create a sequence for auto-incrementing  CREATE SEQUENCE discussions\_seq  START WITH 1001  INCREMENT BY 1  NOCACHE;  -- Create a trigger to auto-increment university  CREATE OR REPLACE TRIGGER trg\_discussions\_id  BEFORE INSERT ON Discussions  FOR EACH ROW  BEGIN  IF :NEW.discussion\_id IS NULL THEN  SELECT discussions\_seq.NEXTVAL INTO :NEW.discussion\_id FROM dual;  END IF;  END;  /  -- Discussion Tags  CREATE TABLE Discussion\_Tags (  discussion\_id INT NOT NULL, -- Links to Discussions table  tag\_id INT NOT NULL, -- Links to Tags table  PRIMARY KEY (discussion\_id, tag\_id), -- Composite primary key  FOREIGN KEY (discussion\_id) REFERENCES Discussions (discussion\_id),  FOREIGN KEY (tag\_id) REFERENCES Tags (tag\_id)  );  -- Discussion Comments  CREATE TABLE Discussion\_Comments (  comment\_id INT PRIMARY KEY, -- Uniquely identifies each comment  comment\_date DATE DEFAULT SYSDATE, -- Date the comment was made  comment\_text VARCHAR2(2000) NOT NULL, -- Text of the comment  discussion\_id INT NOT NULL, -- Links to Discussions table  FOREIGN KEY (discussion\_id) REFERENCES Discussions (discussion\_id)  );  -- Create a sequence for auto-incrementing  CREATE SEQUENCE comments\_seq  START WITH 101  INCREMENT BY 1  NOCACHE;  -- Create a trigger to auto-increment university  CREATE OR REPLACE TRIGGER trg\_comments\_id  BEFORE INSERT ON Discussion\_Comments  FOR EACH ROW  BEGIN  IF :NEW.comment\_id IS NULL THEN  SELECT comments\_seq.NEXTVAL INTO :NEW.comment\_id FROM dual;  END IF;  END;  /  -- Notifications Table  CREATE TABLE Notifications (  notification\_id INT PRIMARY KEY, -- Uniquely identifies each notification  is\_read CHAR(1) DEFAULT 'N' CHECK (is\_read IN ('Y', 'N')), -- Boolean value (Y/N) for read status  created\_date DATE DEFAULT SYSDATE, -- Creation date of the notification  message VARCHAR2(2000) NOT NULL, -- Content of the notification  user\_id INT NOT NULL, -- Links to Users table  FOREIGN KEY (user\_id) REFERENCES Users (user\_id)  );  -- Create a sequence for auto-incrementing  CREATE SEQUENCE notification\_seq  START WITH 1001  INCREMENT BY 1  NOCACHE;  -- Create a trigger to auto-increment university  CREATE OR REPLACE TRIGGER trg\_noti\_id  BEFORE INSERT ON Notifications  FOR EACH ROW  BEGIN  IF :NEW.notification\_id IS NULL THEN  SELECT notification\_seq.NEXTVAL INTO :NEW.notification\_id FROM dual;  END IF;  END;  /  --Action Log  CREATE TABLE Action\_Log (  log\_id INT PRIMARY KEY, -- Uniquely identifies each action log  action\_timestamp DATE DEFAULT SYSDATE, -- Timestamp of the logged action  action\_type VARCHAR2(255) NOT NULL, -- Type of action performed by the user  address VARCHAR2(255), -- IP or device address of the user performing the action  user\_id INT NOT NULL, -- Links to Users table  FOREIGN KEY (user\_id) REFERENCES Users (user\_id)  );  -- Create a sequence for auto-incrementing  CREATE SEQUENCE action\_seq  START WITH 10001  INCREMENT BY 1  NOCACHE;  -- Create a trigger to auto-increment university  CREATE OR REPLACE TRIGGER trg\_action\_id  BEFORE INSERT ON Action\_Log  FOR EACH ROW  BEGIN  IF :NEW.log\_id IS NULL THEN  SELECT action\_seq.NEXTVAL INTO :NEW.log\_id FROM dual;  END IF;  END;  /  -- Loans Table  CREATE TABLE Loans (  loan\_id INT PRIMARY KEY, -- Uniquely identifies each loan  request\_date DATE DEFAULT SYSDATE, -- Date the loan was requested  approval\_date DATE, -- Date the loan was approved  repayment\_date DATE, -- Date the loan should be repaid  amount DECIMAL(10, 2) NOT NULL CHECK (amount > 0), -- Loan amount (must be positive)  status VARCHAR2(50) NOT NULL CHECK (status IN ('pending', 'approved', 'rejected')), -- Loan status  user\_id INT NOT NULL, -- Links to Users table  FOREIGN KEY (user\_id) REFERENCES Users (user\_id)  );  -- Create a sequence for auto-incrementing  CREATE SEQUENCE loans\_seq  START WITH 1001  INCREMENT BY 1  NOCACHE;  -- Create a trigger to auto-increment university  CREATE OR REPLACE TRIGGER trg\_loan\_id  BEFORE INSERT ON Loans  FOR EACH ROW  BEGIN  IF :NEW.loan\_id IS NULL THEN  SELECT loans\_seq.NEXTVAL INTO :NEW.loan\_id FROM dual;  END IF;  END;  /  -- Transactions Table  CREATE TABLE Transactions (  transaction\_id INT PRIMARY KEY, -- Uniquely identifies each transaction  remaining\_balance DECIMAL(10, 2) NOT NULL CHECK (remaining\_balance >= 0), -- Balance remaining after the transaction  transaction\_date DATE DEFAULT SYSDATE, -- Date of the transaction  amount DECIMAL(10, 2) NOT NULL CHECK (amount > 0), -- Transaction amount (must be positive)  transaction\_type VARCHAR2(50) NOT NULL CHECK (transaction\_type IN ('credit', 'debit')), -- Transaction type  user\_id INT NOT NULL, -- Links to Users table  FOREIGN KEY (user\_id) REFERENCES Users (user\_id)  );  -- Create a sequence for auto-incrementing  CREATE SEQUENCE trans\_seq  START WITH 1001  INCREMENT BY 1  NOCACHE;  -- Create a trigger to auto-increment university  CREATE OR REPLACE TRIGGER trg\_transaction\_id  BEFORE INSERT ON Transactions  FOR EACH ROW  BEGIN  IF :NEW.transaction\_id IS NULL THEN  SELECT trans\_seq.NEXTVAL INTO :NEW.transaction\_id FROM dual;  END IF;  END;  / |