LU06.S03 - SQL-DDL: Constraint Management

Assignments & Solutions

A: PRIMARY KEY

Create a table books that has a book_id as a unique identifier for each book, with book_id as the primary key. Include columns for book_title (VARCHAR) and author_name (VARCHAR). The book_id should be an integer and cannot be NULL.

```
CREATE TABLE books (
  book_id INT PRIMARY KEY,
  book_title VARCHAR(100),
  author_name VARCHAR(100)
);
```

B: FOREIGN KEY

Create two tables: one called departments and the other called employees. Each department has a department_id as its primary key. In the employees table, include a column called department_id as a foreign key that references the departments table. Ensure that every employee is linked to a department.

```
CREATE TABLE departments (
  department_id INT PRIMARY KEY,
  department_name VARCHAR(50)
);
```

```
CREATE TABLE employees (
  employee_id INT PRIMARY KEY,
  employee_name VARCHAR(100),
  department_id INT,
  FOREIGN KEY (department_id) REFERENCES departments(department_id)
);
```

C: NOT NULL

Create a table students that includes a student_id (INT) and a student_name (VARCHAR). Ensure that the student_name column cannot have a NULL value by applying the NOT NULL constraint.

```
CREATE TABLE students (
  student_id INT PRIMARY KEY,
  student_name VARCHAR(50) NOT NULL
);
```

D: AUTO INCREMENT

Create a table products where each product has an automatically generated, unique product_id using the AUTO_INCREMENT feature. Include columns for product_name and price.

```
CREATE TABLE products (
  product_id INT AUTO_INCREMENT PRIMARY KEY,
  product_name VARCHAR(100),
  price DECIMAL(10,2)
);
```

E: UNIQUE

Create a table users that has a user_id (INT) and email (VARCHAR). Ensure that no two users can have the same email address by applying the UNIQUE constraint to the email column.

```
CREATE TABLE users (
  user_id INT PRIMARY KEY,
  email VARCHAR(100) UNIQUE
);
```

Vocabulary

English	German
...	...
...	...
