|  |  |
| --- | --- |
|  |  |

**Faculty of Technology and Engineering**

**Chandubhai S. Patel Institute of Technology (CSPIT)**

**Department of Computer Science & Engineering**

Date: / /

**Laboratory Manual**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Academic Year | : | 2024-25 | Semester | : | 4 |
| Course code | : | CSE206 | Course name | : | DATABASE MANAGEMENT SYSTEM |

|  |
| --- |
| **Practical - 7** |
| **Aim:** As a database administrator for a university, you are tasked with designing and implementing a database schema in the MS Access tool. This database should efficiently manage relationships between departments, courses, students, and their academic performance. To ensure data integrity and consistency, you will define master-slave relationships with appropriate integrity constraints.  **maintains the following schemas:**   1. Department (Master Table):   o Stores information about university departments.  o Attributes: Dept\_ID (Primary Key), Dept\_Name (Not Null, Unique).   1. Course (Slave Table):   o Stores courses offered by departments.  o Attributes: Course\_ID (Primary Key), Course\_Name (Not Null), Dept\_ID (Foreign Key referencing Department).   1. Student (Slave Table):   o Stores details of enrolled students.  o Attributes: Student\_ID (Primary Key), Student\_Name (Not Null), Dept\_ID (Foreign Key referencing Department).   1. Enrollment (Slave Table):   o Tracks student enrollments in courses.  o Attributes: Enrollment\_ID (Primary Key), Student\_ID (Foreign Key referencing Student), Course\_ID (Foreign Key referencing Course), Grade. |

**Tasks:-**

1. Create Master Table (Department):

o Design a Department table with Dept\_ID as the Primary Key.

o Enforce the following constraints:

▪ Dept\_Name must be unique.

▪ No null values in Dept\_Name.

o Test Case: Verify that duplicate Dept\_Name values cannot be inserted.

1. Create Slave Table (Course):

o Design a Course table with Course\_ID as the Primary Key.

o Establish a relationship with the Department table using the Dept\_ID foreign key.

o Enforce referential integrity with the following rules:

▪ Cascade updates: If a Dept\_ID is updated in the Department table, the corresponding Dept\_ID in the Course table should update automatically.

▪ Restrict deletions: Prevent deleting a department if courses are linked to it.

o Test Case: Verify that deleting a department linked to courses is restricted.

1. Create Slave Table (Student):

o Design a Student table with Student\_ID as the Primary Key.

o Establish a relationship with the Department table using the Dept\_ID foreign key.

o Enforce referential integrity to ensure that each student belongs to a valid department.

o Test Case: Verify that a student cannot be added with a Dept\_ID that does not exist in the Department table

1. Create Slave Table (Enrollment):

o Design an Enrollment table with Enrollment\_ID as the Primary Key.

o Establish relationships:

▪ Student\_ID as a foreign key referencing the Student table.

▪ Course\_ID as a foreign key referencing the Course table.

o Enforce referential integrity for cascading updates and restricting deletions:

▪ Cascade updates: If a Student\_ID or Course\_ID is updated in their respective tables, the changes should reflect in the Enrollment table.

▪ Restrict deletions: Prevent deleting a student or course if enrollment records exist.

o Test Case: Verify that deleting a student or course linked to enrollments is restricted.

1. Data Validation Rules:

o Ensure that grades in the Enrollment table only accept valid values (A, B, C, D, F).

o Test Case: Verify that invalid grades (e.g., E or empty values) cannot be inserted into the Grade field.

1. Data Entry:

o Populate the tables with sample data for departments, courses, students, and enrollments

Test Case: Verify that the inserted sample data adheres to all constraints and relationships.







