Lecture – 4

Contents:
- Constraints in MySQL
  - NOT NULL
  - UNIQUE
  - PRIMARY KEY
  - FOREIGN KEY
  - CHECK
  - DEFAULT
  - CREATE INDEX

3.1 Constraints:

Constraints are rules and restrictions in tables that ensure the implementation of accuracy, reliability, and integrity in the data being kept.

The following constraints are commonly used in SQL:

- ❖ NOT NULL - Ensures that a column cannot have a NULL value
- ❖ UNIQUE - Ensures that all values in a column are different
- ❖ PRIMARY KEY - A combination of a NOT NULL and UNIQUE. Uniquely identifies each row in a table
- ❖ AUTO_INCREMENT: Automatically maintain the primary key for a table.
- ❖ FOREIGN KEY - Prevents actions that would destroy links between tables
- ❖ CHECK - Ensures that the values in a column satisfies a specific condition
- ❖ DEFAULT - Sets a default value for a column if no value is specified
- ❖ CREATE INDEX - Used to create and retrieve data from the database very quickly

3.2 NOT NULL:

- ❖ Ensures that no NULL value exists in a column.

- ❖ This ensures that the field in the table has a value for each record.

Example of NOT NULL Constraints

In **Department** table, there are all column or record are filled with specific value, there is no any **NULL** value.

Department:

id	name	location	hod	phone
1	CSE	4th Floor	MMA	019...
2	Physics	3rd Floor	SR	013...
3	English	1st Floor	MAH	017....

3.3 UNIQUE:

- ❖ Makes sure that each value in a column is distinct.

- ❖ Unlike the primary key, a table can have multiple unique constraints.

- ❖ A unique column can accept NULL values; however, when it contains a non-NULL value, the value must be unique.

Example of UNIQUE Constraints

In **Department** table, value of column **id** and **phone** must be **UNIQUE** means each row's value are differents.

Department:

id	name	location	hod	phone
1	CSE	4th Floor	MMA	019...
2	Physics	3rd Floor	SR	013...
3	English	1st Floor	MAH	017....

3.4 PRIMARY KEY:

- ❖ Uniquely identifies each record in a table.

- ❖ Only one primary key is allowed in any table, and it can have one or more columns.

- ❖ The PRIMARY KEY column(s) accepts only NOT NULL values.

Example of PRIMARY KEY Constraints

In a **Department** table, **phone** columns is **PRIMARY KEY** constraints. We can also use **id** column as **PRIMARY KEY.** But we can only use one column as **PRIMARY KEY** in a table.
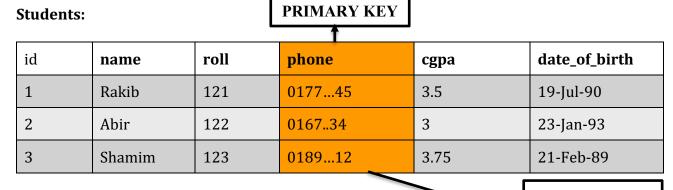
Department:

id	name	location	hod	phone
1	CSE	4th Floor	MMA	019...
2	Physics	3rd Floor	SR	013...
3	English	1st Floor	MAH	017....

3.5 FOREIGN KEY:

❖ Establishes relation between two tables.

❖ The foreign key in one table references the primary key in another table.

❖ Ensures referential integrity, meaning a value in the column of a foreign key will either correspond to a value in the column of a primary key that is referenced, or it will be NULL.

Example of FOREIGN KEY Constraints

Students:

PRIMARY KEY

id	name	roll	phone	cgpa	date_of_birth
1	Rakib	121	0177...45	3.5	19-Jul-90
2	Abir	122	0167..34	3	23-Jan-93
3	Shamim	123	0189...12	3.75	21-Feb-89

FOREIGN KEY

Department:

id	name	location	hod	phone	phone
1	CSE	4th Floor	MMA	019...	0177...45
2	Physics	3rd Floor	SR	013...	0167..34
3	English	1st Floor	MAH	017....	0189...12

3.6 CHECK:

❖ Validates the data entered into a column based on a specific condition or expression.

❖ For instance, a check constraint can specify that a particular column in a table has only those values that are greater than 18.

In a table **Cyber_Cafe**, these person will enter those aged 18 or over. Otherwise they are not allowed.

Example of CHECK Constraints

Cyber_Cafe:

Name	Mobile	Age	Address
Rifat	0178299	23	Dhaka
Zakir	01578299	21	Khulna

3.7 DEFAULT:

❖ It assigns a default value to a column if no value is specified in an insert. It can save you from many NULL values in the columns by giving default data.

Example of DEFAULT Constraints

In a **Students** table, the value of department column which is CSE are **DEFAULT** value.

Students:

id	name	roll	phone	cgpa	department
1	Rakib	121	0177…45	3.5	CSE
2	Abir	122	0167..34	3	CSE
3	Shamim	123	0189…12	3.75	CSE

3.8 CREATE INDEX:

❖ Not a strict data constraint, but indexes are used to speed up the retrieval of data from the database. Though in one sense this is not a real constraint, an integrity

constraint, indexes can and often do work hand-in-hand with other kinds of constraints to help performance.

❖

Practice the following Database:

A complete database named **Library Management System** is given below with details of all tables.

Databse name: **lms**

1. Books Table

This table stores information about the books available in the library.

Column Name	Data Type	Description
BookID	INT	Primary key, unique identifier for each book
Title	VARCHAR	Title of the book
AuthorID	INT	Foreign key referencing AuthorID in the Authors table
PublisherID	INT	Foreign key referencing PublisherID in the Publishers table
ISBN	VARCHAR	International Standard Book Number
YearPublished	INT	Year the book was published

2. Authors Table

This table stores information about the authors of the books.

Column Name	Data Type	Description
AuthorID	INT	Primary key, unique identifier for each author
Name	VARCHAR	Name of the author
Country	VARCHAR	Country of origin

3. Publishers Table

This table stores information about the publishers who publish the books.

Column Name	Data Type	Description
PublisherID	INT	Primary key, unique identifier for each publisher
Name	VARCHAR	Name of the publisher
Location	VARCHAR	Location of the publisher

4. Members Table

This table stores information about the library members who borrow books.

Column Name	Data Type	Description
MemberID	INT	Primary key, unique identifier for each member
Name	VARCHAR	Name of the member
Address	VARCHAR	Address of the member
PhoneNumber	VARCHAR	Contact number of the member
Email	VARCHAR	Email address of the member

5. BorrowedBooks Table

This table records the books that have been borrowed by members.

Column Name	Data Type	Description
BorrowID	INT	Primary key, unique identifier for each borrowed record
MemberID	INT	Foreign key referencing MemberID in the Members table
BookID	INT	Foreign key referencing BookID in the Books table
BorrowDate	DATE	Date when the book was borrowed
DueDate	DATE	Date when the book is due for return
ReturnDate	DATE	Date when the book was returned (can be null if not returned)

Relations Between Tables (Set Foreign Key):

1. **Books Table**:

   o   AuthorID is a foreign key referencing the **Authors Table**.

   o   PublisherID is a foreign key referencing the **Publishers Table**.

2. **BorrowedBooks Table**:

   o   MemberID is a foreign key referencing the **Members Table**.

   o   BookID is a foreign key referencing the **Books Table**.