I

E & C

T

IIT Guwahati

Advanced Certification Programme in Data Science Business Analytics

Week 5
Constraints in SQL

Topics Covered

- SQL Constraints
- Primary Key and Foreign Key
- Unique, Not Null, Check and Default Constraints
- Case Study
- Solving Constraint-Related Issues
- Best Practices for Managing Constraints
- Q & A

SQL Constraints

Constraints in MySQL

Ensuring Data Integrity, Consistency and Accuracy

- Enforce data integrity, consistency and accuracy
- Ensure data follows specific rules, reducing errors
- Prevent invalid data from entering the table
- Maintain meaningful relationships between tables

Types of Constraints

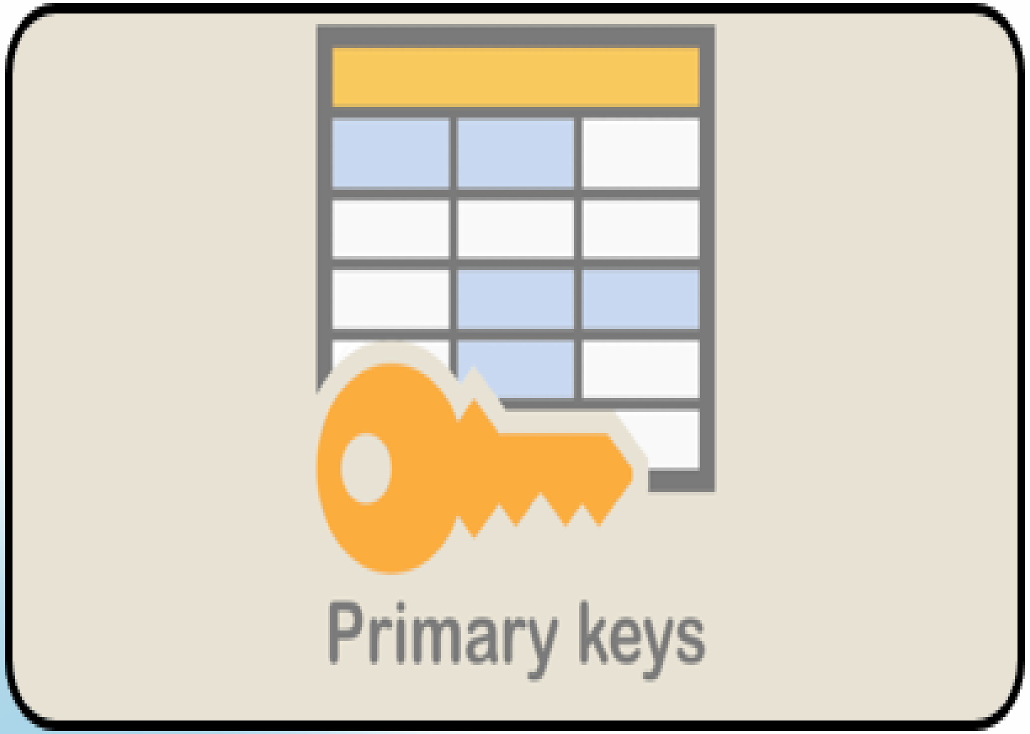
Ensuring Validity and Consistency in Data

1. Primary key
2. Foreign key
3. Unique key
4. Not Null key
5. Check key
6. Default key

Primary Key and Foreign key

SQL Constraints: Primary Key

Unique Identifier for Each Record in a Table



Definition

A unique identifier for each record in a database table

Usage

- Identifies each record uniquely in a table
- Prevents duplicate entries in key columns
- Enforces entity integrity

Need

- Maintains data integrity with unique records
- Enables efficient data retrieval and indexing

Create Primary Key

Defining a Unique Identifier in a Table

While creating a table

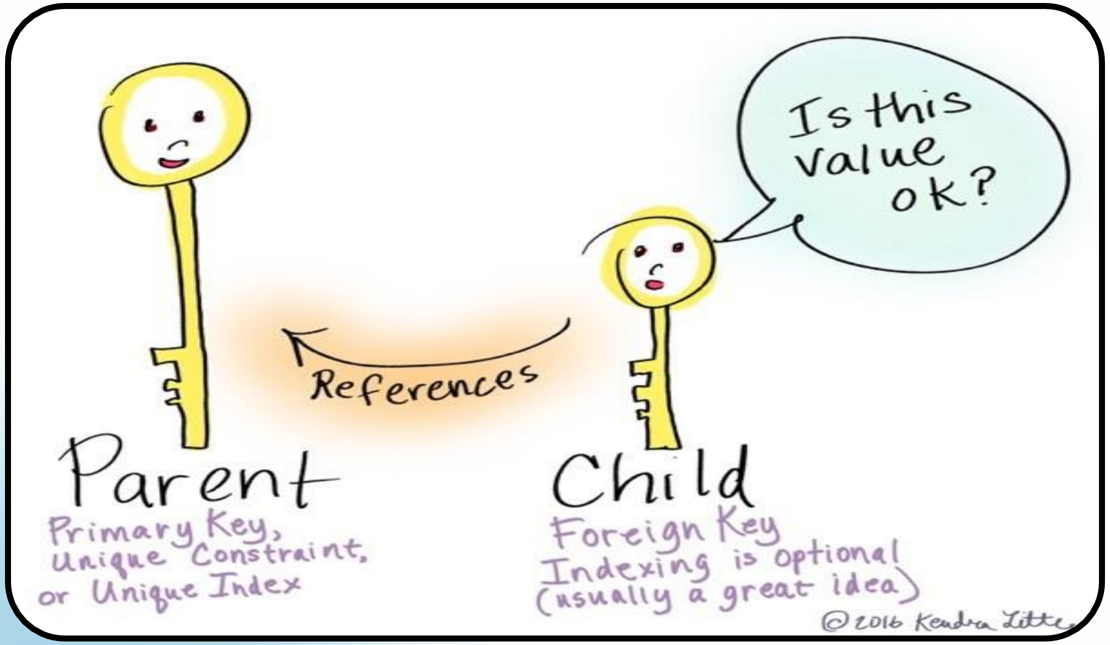
```
CREATE TABLE Customers ( CustomerID
INT NOT NULL, FirstName VARCHAR(50),
LastName VARCHAR(50), Email
VARCHAR(100), PRIMARY KEY
(CustomerID) );
```

On an existing table

```
ALTER TABLE Customers ADD PRIMARY
KEY (CustomerID);
```

SQL Constraints: Foreign Key

Ensuring Data Integrity and Relationships


Is this value ok?
References
Parent
Primary Key, Unique Constraint, or Unique Index
Child
Foreign Key
Indexing is optional (usually a great idea)
©2016 Kendra Litte

Definition

Links a column to the primary key of another table, ensuring data consistency

Usage

- Links related data across tables
- Enforces referential integrity

Need

- Maintains data consistency
- Supports complex queries and retrieval

Creating a Foreign Key in SQL

Define Relationships Between Tables for Data Integrity

While creating a table

```
CREATE TABLE Orders ( OrderID INT NOT
NULL, OrderDate DATE, CustomerID INT,
PRIMARY KEY (OrderID), FOREIGN KEY
(CustomerID) REFERENCES
Customers(CustomerID) );
```

On an existing table

```
ALTER TABLE Orders ADD CONSTRAINT
FK_CustomerOrder FOREIGN KEY
(CustomerID) REFERENCES
Customers(CustomerID);
```

Cascading Actions in Foreign Keys

Maintain Data Consistency with Delete and Update Cascades

On delete cascade

```
FOREIGN KEY (CustomerID)
REFERENCES Customers(CustomerID)
ON DELETE CASCADE;
```

On update cascade

```
FOREIGN KEY (CustomerID)
REFERENCES Customers(CustomerID)
ON UPDATE CASCADE;
```

Unique, Not Null, Check and Default Constraints

Unique, Not Null, Check & Default Constraints

Ensure Data Integrity and Consistency with SQL Constraints

Constraint type	Purpose	Behaviour	NULL values	Example use case
Unique	Ensures values in a column (or combination) are distinct	Prevents duplicate entries	Allows one NULL (in most SQL databases)	Unique email addresses
Not Null	Prevents NULL values in a column	Require a value for every row	Not allowed	Mandatory fields like first and last names
Check	Enforces a condition on column values	Validates data before inserting or updating	Allows NULL unless restricted	Ensuring prices are greater than zero
Default	Assigns a default value if none is provided	Auto-fills column with a specified value	Allows a NULL or a set value	Status column defaulting to 'Pending'

Essential SQL Constraints

Key Rules for Data Consistency

- **Unique:** Ensures distinct values, allows one NULL
- **Not Null:** Prevents NULL entries in a column
- **Check:** Enforces conditions on values, NULL allowed unless restricted
- **Default:** Assigns a preset value if none is provided, can be NULL or set value

Case Study

Managing a Bookstore Database

Ensuring Data Accuracy with Constraints

- Track books, customers and orders efficiently
- Maintain correct and consistent data
- Apply database constraints for accuracy
- Prevent duplicate or missing entries

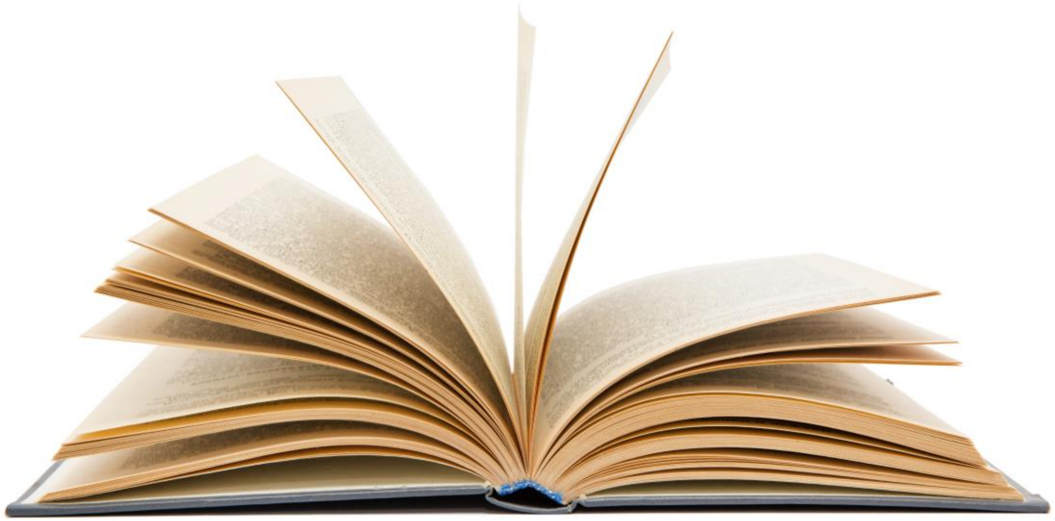
Scenario: Creating a Bookstore Database

Structuring Data for Efficiency and Accuracy

Table to create	Description
Books	To store information about each book
Customers	To keep track of customer details
Orders	To record customer purchases

Setting Up the Books Table

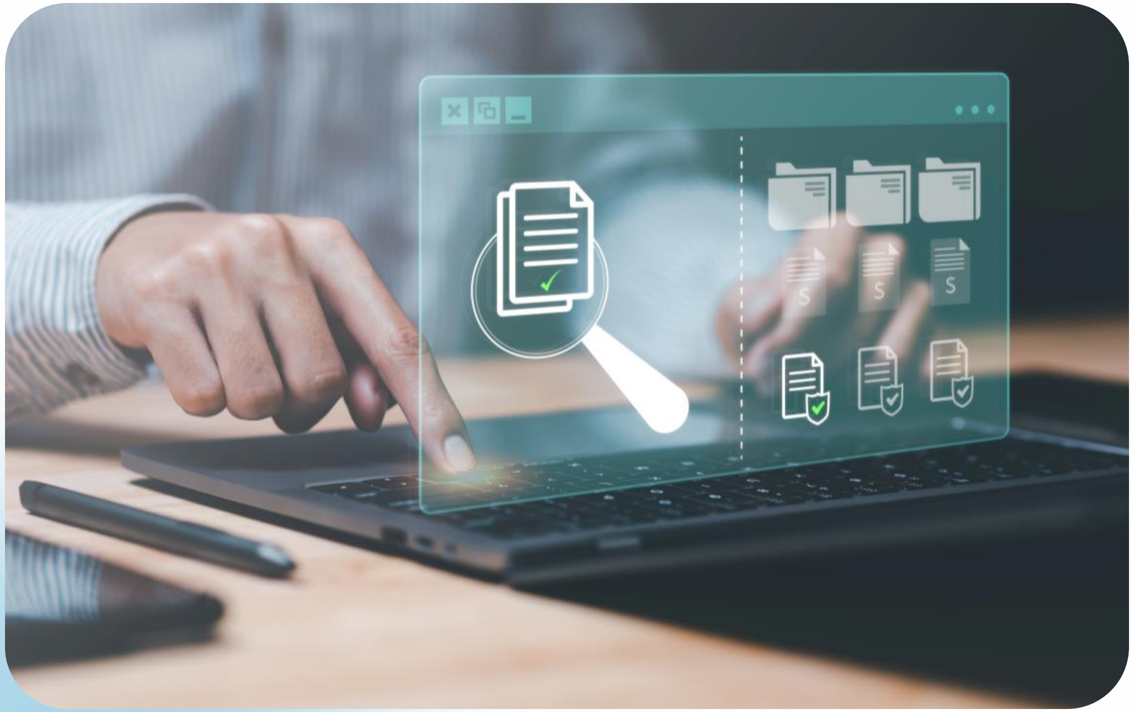
Key Requirements for Data Integrity



- Ensure that every book has a unique BookID for identification
- Prevent the title of the book from being left empty
- Ensure the price remains above zero at all times
- Keep the stock value from dropping below zero

Creating the Customers Table

Essential Rules for Customer Data



- Assign a unique CustomerID for identification
- Ensure the email is unique, preventing duplication among customers
- Ensure FirstName and LastName are not empty

Creating the Orders Table

Key Requirements for Order Records

- Assign a unique OrderID for tracking purposes
- Require a CustomerID to associate each order with a customer
- Set the OrderDate to the current date by default if not specified

Inserting Data into Tables

Populating the Database with Records

Table type	SQL code
Books	INSERT INTO Books (Title, Author, Price, Stock) VALUES ('The Great Gatsby', 'F. Scott Fitzgerald', 10.99, 5);
Orders	INSERT INTO Customers (FirstName, LastName, Email) VALUES ('Alice', 'Smith', 'alice@example.com');
Customers	INSERT INTO Orders (CustomerID, BookID) VALUES (1, 1);

Verifying Data Accuracy

Run Queries to Check Your Tables

Data accuracy

- Retrieve all records from the tables to verify data entries
- Ensure all stored information is accurate and complete
- Validate relationships between tables for consistency

Queries

```
SELECT * FROM Books;
SELECT * FROM Customers;
SELECT * FROM Orders;
```

Solving Constraint-Related Issues

Managing Constraint Issues in Databases

Maintaining Data Integrity with Constraints

Common database constraints

- Complex databases use constraints to ensure accuracy
- Constraints include PRIMARY KEY, FOREIGN KEY, UNIQUE, CHECK and DEFAULT

Resolving constraint issues

- Manage constraints properly to prevent data inconsistencies and errors
- Understand constraints to effectively prevent and resolve issues

Common Constraint Issues

Violation of UNIQUE Constraints

Issue

- Inserting duplicate values in a column with a UNIQUE constraint

Example

- Adding two customers with the same email in the customers' table

Solution

- Check for existing values using a SELECT query before inserting
- Implement error handling to manage duplicates smoothly

Common Constraint Issues

FOREIGN KEY Violations

Issue

- Inserting a record with a FOREIGN KEY that has no match in the parent table

Example

- Adding an order for a non-existent customer in the customer's table

Solution

- Ensure the referenced record exists before insertion
- Use cascading options like CASCADE or SET NULL when defining constraints

Common Constraint Issues

CHECK Constraint Violations

Issue

Inserting or updating a record that does not meet the CHECK constraint conditions

Example

Adding a book with a negative price in the books table, where the CHECK constraint enforces a price greater than zero

Solution

- Validate values before insertion to meet CHECK constraints
- Adjust application logic to prevent invalid entries

Common Constraint Issues

NOT NULL Constraint Violations

Issue

Inserting a NULL value into a column that requires a value

Example

Adding a customer without a name in the customer's table where first name is NOT NULL

Solution

- Ensure all required fields are filled before insertion
- Implement validation rules to detect and handle missing values early

Common Constraint-Related Issues

CASCADING Issues

Issue

Deleting a record from a parent table while related records still exist in the child table

Example

Removing a customer who has existing orders in the orders table

Solution

- Define cascading actions like CASCADE DELETE or SET NULL for foreign keys
- Choose to auto-delete related records or block deletion to prevent orphan records

Best Practices for Managing Constraints

Best Practices for Data Integrity

Ensuring Effective Constraint Management

- Define constraints clearly
- Use transactions
- Implement error handling
- Maintain proper documentation
- Validate data regularly

Q & A

Thank you