Module 5: Implementing Data Integrity
Overview

- Types of Data Integrity
- Enforcing Data Integrity
- Defining Constraints
- Types of Constraints
- Disabling Constraints
- Using Defaults and Rules
- Deciding Which Enforcement Method to Use
Types of Data Integrity

- **Domain Integrity** (columns)
- **Entity Integrity** (rows)
- **Referential Integrity** (between tables)
Enforcing Data Integrity

- **Declarative Data Integrity**
  - Criteria defined in object definitions
  - SQL Server enforces automatically
  - Implement by using constraints, defaults, and rules

- **Procedural Data Integrity**
  - Criteria defined in script
  - Script enforces
  - Implement by using triggers and stored procedures
Defining Constraints

- Determining Which Type of Constraint to Use
- Creating Constraints
- Considerations for Using Constraints
# Determining Which Type of Constraint to Use

<table>
<thead>
<tr>
<th>Type of integrity</th>
<th>Constraint type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain</td>
<td>DEFAULT</td>
</tr>
<tr>
<td></td>
<td>CHECK</td>
</tr>
<tr>
<td></td>
<td>REFERENTIAL</td>
</tr>
<tr>
<td>Entity</td>
<td>PRIMARY KEY</td>
</tr>
<tr>
<td></td>
<td>UNIQUE</td>
</tr>
<tr>
<td>Referential</td>
<td>FOREIGN KEY</td>
</tr>
<tr>
<td></td>
<td>CHECK</td>
</tr>
</tbody>
</table>
Creating Constraints

- Use CREATE TABLE or ALTER TABLE
- Can Add Constraints to a Table with Existing Data
- Can Place Constraints on Single or Multiple Columns
  - Single column, called column-level constraint
  - Multiple columns, called table-level constraint
Considerations for Using Constraints

- Can Be Changed Without Recreating a Table
- Require Error-Checking in Applications and Transactions
- Verify Existing Data
Types of Constraints

- DEFAULT Constraints
- CHECK Constraints
- PRIMARY KEY Constraints
- UNIQUE Constraints
- FOREIGN KEY Constraints
- Cascading Referential Integrity
DEFAULT Constraints

- Apply Only to INSERT Statements
- Only One DEFAULT Constraint Per Column
- Cannot Be Used with IDENTITY Property or rowversion Data Type
- Allow Some System-supplied Values

USE Northwind
ALTER TABLE dbo.Customers
ADD
CONSTRAINT DF_contactname DEFAULT 'UNKNOWN'
FOR ContactName
CHECK Constraints

- Are Used with INSERT and UPDATE Statements
- Can Reference Other Columns in the Same Table
- Cannot:
  - Be used with the `rowversion` data type
  - Contain subqueries

```sql
USE Northwind
ALTER TABLE dbo.Employees
ADD CONSTRAINT CK_birthdate
CHECK (BirthDate > '01-01-1900' AND BirthDate < getdate())
```
PRIMARY KEY Constraints

- Only One PRIMARY KEY Constraint Per Table
- Values Must Be Unique
- Null Values Are Not Allowed
- Creates a Unique Index on Specified Columns

USE Northwind
ALTER TABLE dbo.Customers
ADD
CONSTRAINT PK_Customers
    PRIMARY KEY NONCLUSTERED (CustomerID)
UNIQUE Constraints

- Allow One Null Value
- Allow Multiple UNIQUE Constraints on a Table
- Defined with One or More Columns
- Enforced with a Unique Index

USE Northwind
ALTER TABLE dbo.Suppliers
ADD CONSTRAINT U_CompanyName
    UNIQUE NONCLUSTERED (CompanyName)
FOREIGN KEY Constraints

- Must Reference a PRIMARY KEY or UNIQUE Constraint
- Provide Single or Multicolumn Referential Integrity
- Do Not Automatically Create Indexes
- Users Must Have SELECT or REFERENCES Permissions on Referenced Tables
- Use Only REFERENCES Clause Within Same Table

USE Northwind
ALTER TABLE dbo.Orders
ADD CONSTRAINT FK_Orders_Customers
    FOREIGN KEY (CustomerID)
    REFERENCES dbo.Customers(CustomerID)
### Cascading Referential Integrity

#### NO ACTION

1. INSERT new CustomerID

2. UPDATE old CustomerID to new CustomerID

3. DELETE old CustomerID

#### CASCADE

1. UPDATE CustomerID

2. CASCADE

3. Customers

<table>
<thead>
<tr>
<th>CustomerID (PK)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
</tbody>
</table>

4. Orders

<table>
<thead>
<tr>
<th>CustomerID (FK)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
</tbody>
</table>

5. Customers

<table>
<thead>
<tr>
<th>CustomerID (PK)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
</tr>
</tbody>
</table>

6. Orders

<table>
<thead>
<tr>
<th>CustomerID (FK)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
</tr>
</tbody>
</table>

7. Customers

<table>
<thead>
<tr>
<th>CustomerID (PK)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
</tr>
</tbody>
</table>

8. Orders

<table>
<thead>
<tr>
<th>CustomerID (FK)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
</tr>
</tbody>
</table>
Disabling Constraints

- Disabling Constraint Checking on Existing Data
- Disabling Constraint Checking When Loading New Data
Disabling Constraint Checking on Existing Data

- Applies to CHECK and FOREIGN KEY Constraints
- Use WITH NOCHECK Option When Adding a New Constraint
- Use if Existing Data Will Not Change
- Can Change Existing Data Before Adding Constraints

USE Northwind
ALTER TABLE dbo.Employees
WITH NOCHECK
ADD CONSTRAINT FK_Employees_Employees
FOREIGN KEY (ReportsTo)
REFERENCES dbo.Employees(EmployeeID)
Disabling Constraint Checking When Loading New Data

- Applies to CHECK and FOREIGN KEY Constraints
- Use When:
  - Data conforms to constraints
  - You load new data that does not conform to constraints

USE Northwind
ALTER TABLE dbo.Employees
   NOCHECK
      CONSTRAINT FK_Employees_Employees
Using Defaults and Rules

- As Independent Objects They:
  - Are defined once
  - Can be bound to one or more columns or user-defined data types

```
CREATE DEFAULT phone_no_default
    AS '(000)000-0000'
GO
EXEC sp_bindefault phone_no_default,
    'Customers.Phone'

CREATE RULE regioncode_rule
    AS @regioncode IN ('IA', 'IL', 'KS', 'MO')
GO
EXEC sp_bindrule regioncode_rule,
    'Customers.Region'
```
Deciding Which Enforcement Method to Use

<table>
<thead>
<tr>
<th>Data integrity components</th>
<th>Functionality</th>
<th>Performance costs</th>
<th>Before or after modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constraints</td>
<td>Medium</td>
<td>Low</td>
<td>Before</td>
</tr>
<tr>
<td>Defaults and rules</td>
<td>Low</td>
<td>Low</td>
<td>Before</td>
</tr>
<tr>
<td>Triggers</td>
<td>High</td>
<td>Medium-High</td>
<td>After</td>
</tr>
<tr>
<td>Data types, Null/Not Null</td>
<td>Low</td>
<td>Low</td>
<td>Before</td>
</tr>
</tbody>
</table>
Recommended Practices

- Use Constraints Because They Are ANSI-compliant
- Use Cascading Referential Integrity Instead of Triggers
Lab A: Implementing Data Integrity
Review

- Types of Data Integrity
- Enforcing Data Integrity
- Defining Constraints
- Types of Constraints
- Disabling Constraints
- Using Defaults and Rules
- Deciding Which Enforcement Method to Use