Data Integrity

IT 4153 Advanced Database

J.G. Zheng
Spring 2012

SPSU SOUTHERN POLYTECHNIC STATE UNIVERSITY

Overview

- Three basic types of data integrity
- Integrity implementation and enforcement
  - Database constraints
  - Transaction
  - Trigger

Data Integrity

- Data integrity:
  - Completeness, accuracy/correctness, consistency, validity, accessibility, currency
- Relational databases ensure data integrity in 3 aspects:
  - Entity Integrity: no empty or duplicate rows in a table.
    - Use primary key constraints
  - Referential integrity: rows cannot be deleted or modified, if they are referenced by other records in other tables
    - Use foreign key constraints
  - Domain Integrity: valid entries for a given column is restricted by the type, the format, or the range of possible values
    - Use "Check" rules on columns

3

Entity Integrity

- Entity Integrity
  - No empty or duplicate rows in a table.
- Implementation
  - Primary key
  - Unique key

4

Domain Integrity

- Domain Integrity
  - Valid entries for a given column is restricted by the type, the format, or the range of possible values
- Implementation
  - Use "Check" constraints on columns
    - Range of values
    - Finite number of choices
    - Patterns, regex
    - XML

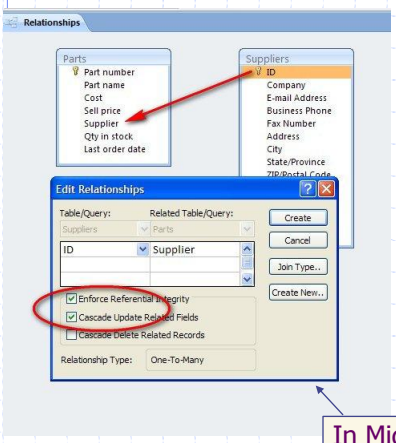
5

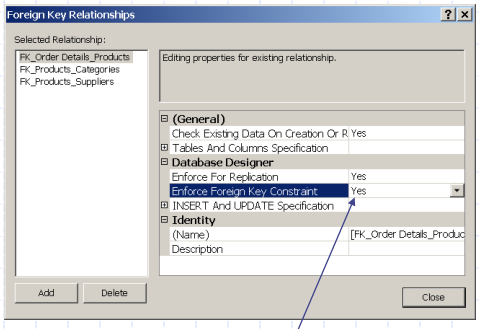
Referential Integrity

- Every value of a foreign key must match a value of the referenced primary key or be "NULL"
- Implementation
  - Foreign key constraints

6

Enforcing Referential Integrity in Databases





In Microsoft Access

In SQL Server

Referential Actions

- Actions when a dependent row (primary key) is updated or deleted
  - CASCADE
  - SET NULL
  - SET DEFAULT
  - NO ACTION
  - RESTRICT

Transaction

- A serials of database actions (operations) defined as a single logical unit (atomic)
  - They should either all succeed, or all fail, in case of system errors
- General process

Begin Transaction
Execute a set of data manipulations and/or queries;
(Previous changes are "uncommitted data")
If no error, *commit* changes;
Otherwise, *rollback* changes;
(End Transaction)

9

Transaction Example

- Update Nancy's hire date to 1992/6/1, and Jack's hire date 1 day after Nancy's hire date.

```
BEGIN TRANSACTION

DECLARE @newDate date='8/1/1992';

update Employees set HireDate=@newDate
where EmployeeID=1;

update Employees set HireDate=dateadd(DAY, 1, @newDate)
where EmployeeID=10;

COMMIT;
```

10

Error Handling

- Use @@ERROR and @@ROWCOUNT to validate the operation of a DML statement
- @@ERROR is checked for any indication of an error
  - Returns 0 if no errors.
  - Because @@ERROR is cleared and reset on each statement executed, check it immediately following the statement being verified, or save it to a local variable that can be checked later.
- @@ROWCOUNT is the number of rows that were affected in the last modification

11

Transaction with Error Handling

```
BEGIN TRANSACTION

DECLARE @newDate date='10/1/1992';

update Employees set HireDate=@newDate
where EmployeeID=1;

if @@ERROR <> 0 or @@ROWCOUNT<>1
begin
        rollback tran;
        return;
end

update Employees set HireDate=dateadd(DAY, 1, @newDate)
where EmployeeID=11;

if @@ERROR = 0 and @@ROWCOUNT=1
        commit;
else
        rollback;
```

Check for errors after each statement.

RETURN is immediate and complete and can be used at any point to exit from a procedure, batch, or statement block. Statements that follow RETURN are not executed.

12

Trigger

- A trigger is a special kind of stored procedure that automatically executes when an event occurs in the DBMS.
- DML triggers
  - execute when a user tries to modify data through a data manipulation language (DML) event (INSERT, UPDATE, or DELETE statements)
  - These triggers fire when any valid event is fired, regardless of whether or not any table rows are affected.
- DDL triggers
  - execute in response to a variety of data definition language (DDL) events (primarily correspond to T-SQL CREATE, ALTER, and DROP statements, and certain system stored procedures that perform DDL-like operations)
- Logon triggers
  - fire in response to the LOGON event that is raised when a user sessions is being established.

13

DML Triggers

- DML triggers are frequently used for enforcing business rules and data integrity.
- The trigger and the statement that fires it are treated as a single transaction, which can be rolled back from within the trigger.
- Types
  - AFTER Triggers: executed after the action of the INSERT, UPDATE, or DELETE statement is performed. AFTER triggers can be specified only on tables.
  - INSTEAD OF Triggers: executed in place of the usual triggering action. INSTEAD OF triggers can also be defined on views with one or more base tables.
  - http://msdn.microsoft.com/en-us/library/ms190267.aspx

14

Create an AFTER Trigger

- Special tables used in DML triggers

```
CREATE TRIGGER TestTrigger
on Products
AFTER UPDATE
as
select * from deleted;
select * from inserted;
rollback transaction;
go
```

This is an AFTER UPDATE trigger

These two temporary tables are used to hold uncommitted data.

Ignore all modifications if conditions are not met.

```
UPDATE Products set UnitPrice=20
WHERE ProductName='Chai'
```

15

AFTER Trigger Practical Example

```
CREATE TRIGGER PriceUpdateLog
ON Products
AFTER UPDATE
AS
Select  SYSTEM_user as Who,
        deleted.UnitPrice as OldValue,
        inserted.UnitPrice as NewValue,
        getdate() as ChangeDateTime
from deleted inner join inserted on
        deleted.ProductID = inserted.ProductID
go
```

Log all price changes for the products table.

These can be inserted into a log table if we created one.

16

DML Triggers and Constraints

- Constraints and DML triggers each have benefits that make them useful in special situations. The primary benefit of DML triggers is that they can contain complex processing logic that uses Transact-SQL code.
- If constraints exist on the trigger table, they are checked after the INSTEAD OF trigger execution and before the AFTER trigger execution.
- If the constraints are violated, the INSTEAD OF trigger actions are rolled back and the AFTER trigger is not fired.

17

Summary

- Key concepts
  - Data integrity
  - Domain integrity, entity integrity, referential integrity
  - Transaction, commit, rollback
  - Trigger, after trigger, instead of trigger
- Key skills
  - Declare explicit transactions
  - Create database DML triggers

18