Dynamic SQL and SQL Injection

- Dynamic SQL
  - Concatenate strings to create dynamic SQL statements
- SQL Injection
  - Hackers make substitution for variables from front-end application
- SQL injection can provide a back door for hackers trying to access or destroy data

SQL Injection Risks

- Hackers may be able to use SQL injection to gain administrator permissions
- Retrieve and modify data stored in server
- Take control of server
- Access network resources

SQL Injection Example

```
DECLARE @product_name nvarchar(50) =
N'Mountain';

DECLARE @sql_stmt NVARCHAR(128) = N'SELECT
ProductID, Name ' +
N'FROM Production.Product ' +
N'WHERE Name LIKE ''' +
@product_name + N'%''';

EXECUTE (@sql_stmt);
```

SQL Injection Example

```
DECLARE @product_name nvarchar(50) =
N''; DROP TABLE Production.Product; --‘

SELECT ProductID, Name
FROM Production.Product
WHERE Name LIKE '';
DROP TABLE Production.Product; --%'
```

How To Prevent SQL Injection

- Always validate the input data
- Disallow apostrophes, semicolons, parentheses, and double hyphens (––) in the input if possible
- Reject strings that contain binary data, escape sequences, and multiline comment markers (/* and */)
- Validate XML input data against an XML schema when possible