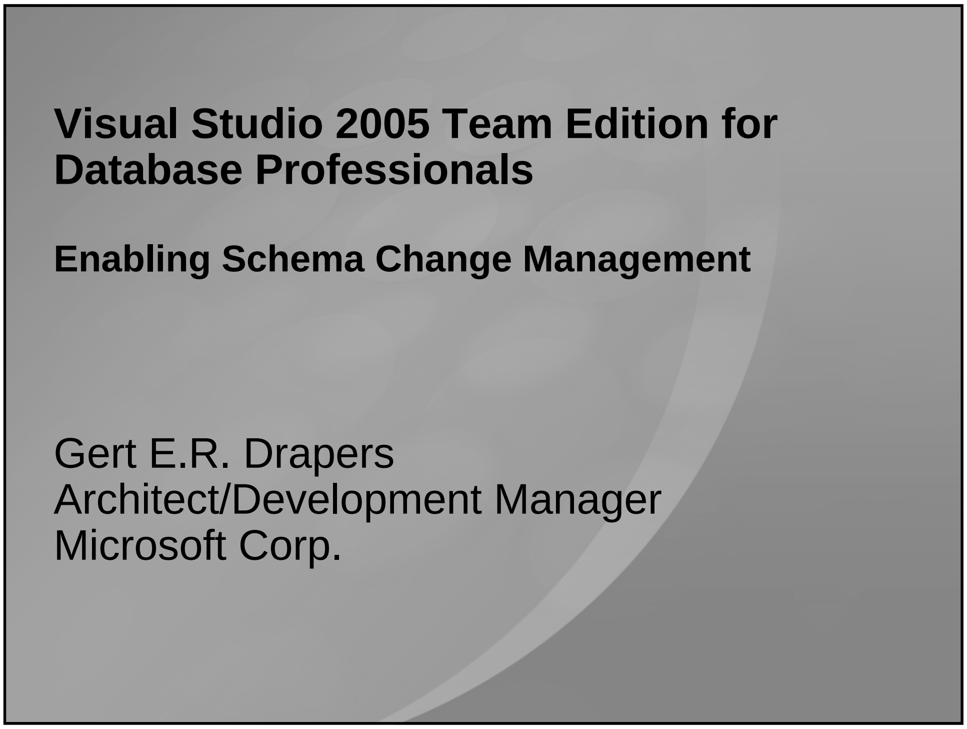
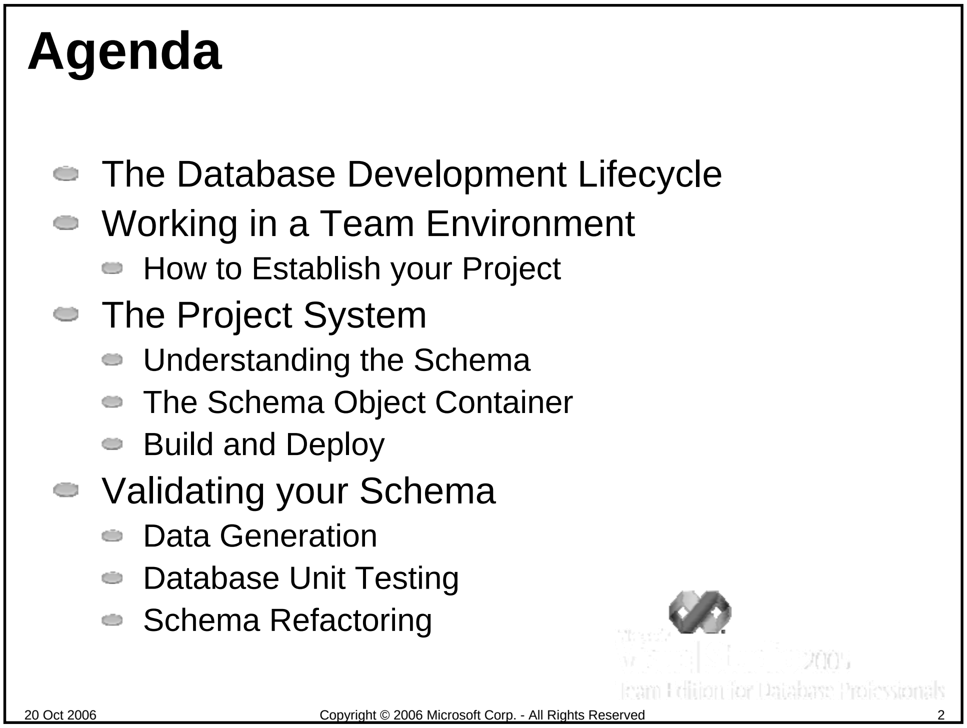


Visual Studio 2005 Team Edition for Database Professionals

Enabling Schema Change Management

Gert E.R. Drapers
Architect/Development Manager
Microsoft Corp.



Agenda

- The Database Development Lifecycle
- Working in a Team Environment
  - How to Establish your Project
- The Project System
  - Understanding the Schema
  - The Schema Object Container
  - Build and Deploy
- Validating your Schema
  - Data Generation
  - Database Unit Testing
  - Schema Refactoring

20 Oct 2006 Copyright © 2006 Microsoft Corp. - All Rights Reserved 2

This presentation is for informational purposes only. Microsoft makes no warranties, express or implied, in this summary.

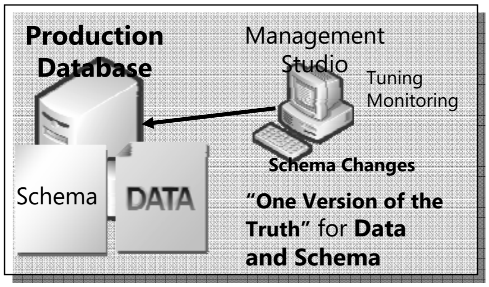
Microsoft Visual Studio 2005 Team Edition for Database Professionals

Incorporate the Database Professional into the software lifecycle and provide them with a foundation for change management and process integration.

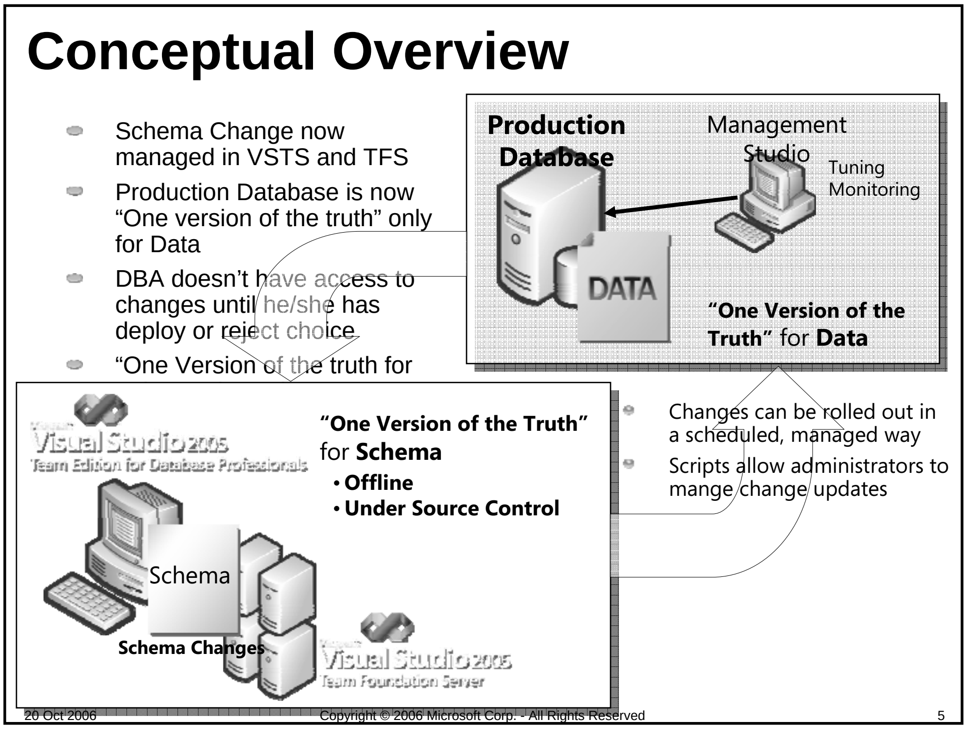
- Change Management
  - Project Based Development
    - Project Model that represents schema as objects providing a “personal sandbox” for offline development that lives within a Visual Studio Solution
    - Team Collaboration with Work Item and Process Integration with Team Foundation Server
  - Automated Change Support
    - Rename Refactoring with the ability to preview pending changes prior to execution
    - Comparison Tools (Schema & Data Compare) allow comparisons & synchronization of schema and data with design/test/production databases
    - Source/Version Control of all database objects with the ability to reverse engineer a database to bring it under Source Control
  - Database Unit Testing
    - Leverages the Test Project Infrastructure
    - Generate “Real and Meaningful” Data Values through the ability to import information such as Row Counts and histograms from a real database
    - Data Generator provides Repetitive Dataset Generation for tests based on saved settings
- Build / Deployment
  - MSBuild Integration for Database Deployments/Builds based on Projects
  - Either Create a new Database at the target location or Update an Existing Schema

Conceptual Overview

- Difficult to Manage Change to the schema
- Production Database is one version of the truth for Data and Schema
- DBA doesn’t have access to changes until he/she has deploy or reject choice
- Changes often made to production database and not rolled back into test



This presentation is for informational purposes only. Microsoft makes no warranties, express or implied, in this summary.



Conceptual Overview

- Schema Change now managed in VSTS and TFS
- Production Database is now “One version of the truth” only for Data
- DBA doesn’t have access to changes until he/she has deploy or reject choice
- “One Version of the truth for

Production Database

Management Studio

Tuning
Monitoring

DATA

“One Version of the Truth” for **Data**

Visual Studio 2005
Team Edition for Database Professionals

“One Version of the Truth”
for **Schema**

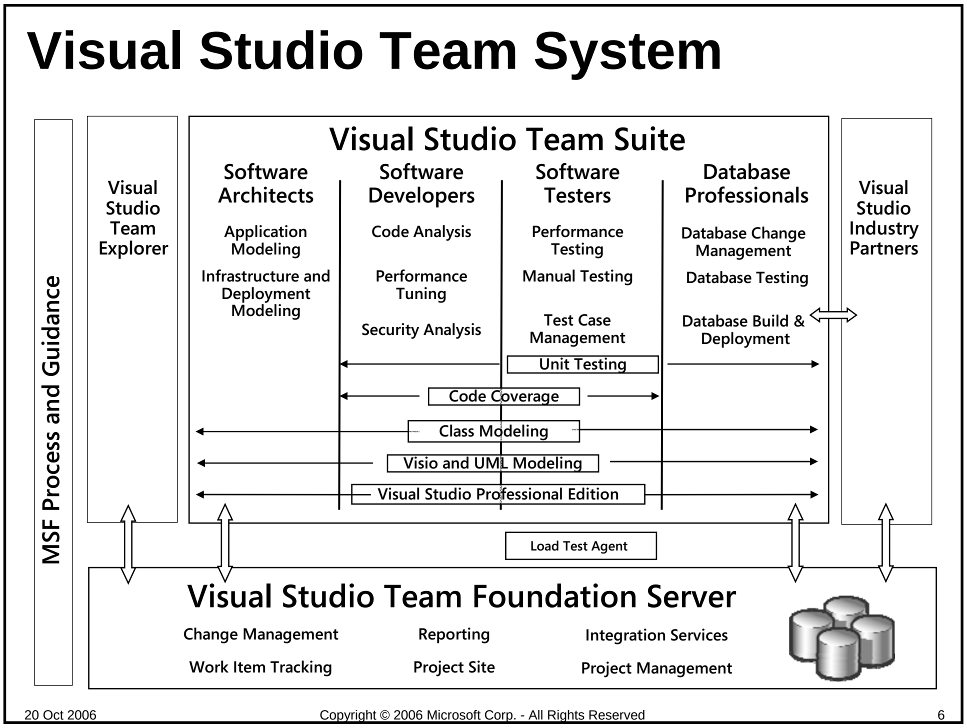
- **Offline**
- **Under Source Control**

Schema

Schema Changes

Visual Studio 2005
Team Foundation Server

- Changes can be rolled out in a scheduled, managed way
- Scripts allow administrators to mange change updates



Visual Studio Team System

MSF Process and Guidance

Visual Studio Team Explorer

Visual Studio Team Suite

Software Architects	Software Developers	Software Testers	Database Professionals
Application Modeling	Code Analysis	Performance Testing	Database Change Management
Infrastructure and Deployment Modeling	Performance Tuning	Manual Testing	Database Testing
	Security Analysis	Test Case Management	Database Build & Deployment

Unit Testing

Code Coverage

Class Modeling

Visio and UML Modeling

Visual Studio Professional Edition

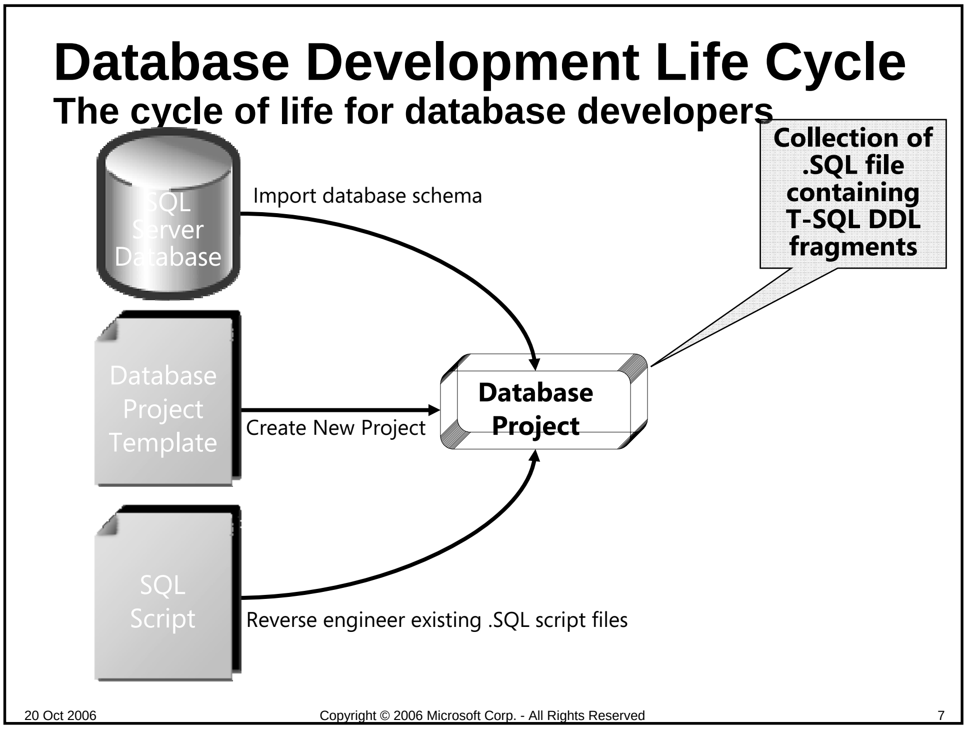
Visual Studio Industry Partners

Load Test Agent

Visual Studio Team Foundation Server

Change Management
Reporting
Integration Services
Work Item Tracking
Project Site
Project Management

This presentation is for informational purposes only. Microsoft makes no warranties, express or implied, in this summary.



Database Development Life Cycle

The cycle of life for database developers

SQL Server Database

Import database schema

Database Project Template

Create New Project

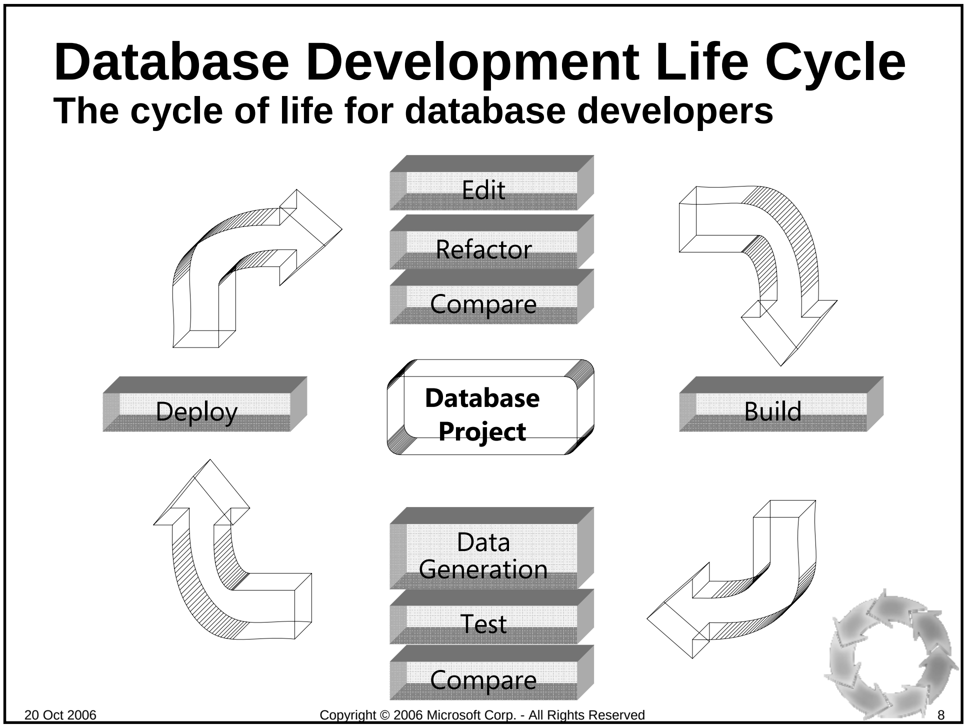
SQL Script

Reverse engineer existing .SQL script files

Database Project

Collection of .SQL file containing T-SQL DDL fragments

20 Oct 2006 Copyright © 2006 Microsoft Corp. - All Rights Reserved 7



Database Development Life Cycle

The cycle of life for database developers

Edit

Refactor

Compare

Deploy

Database Project

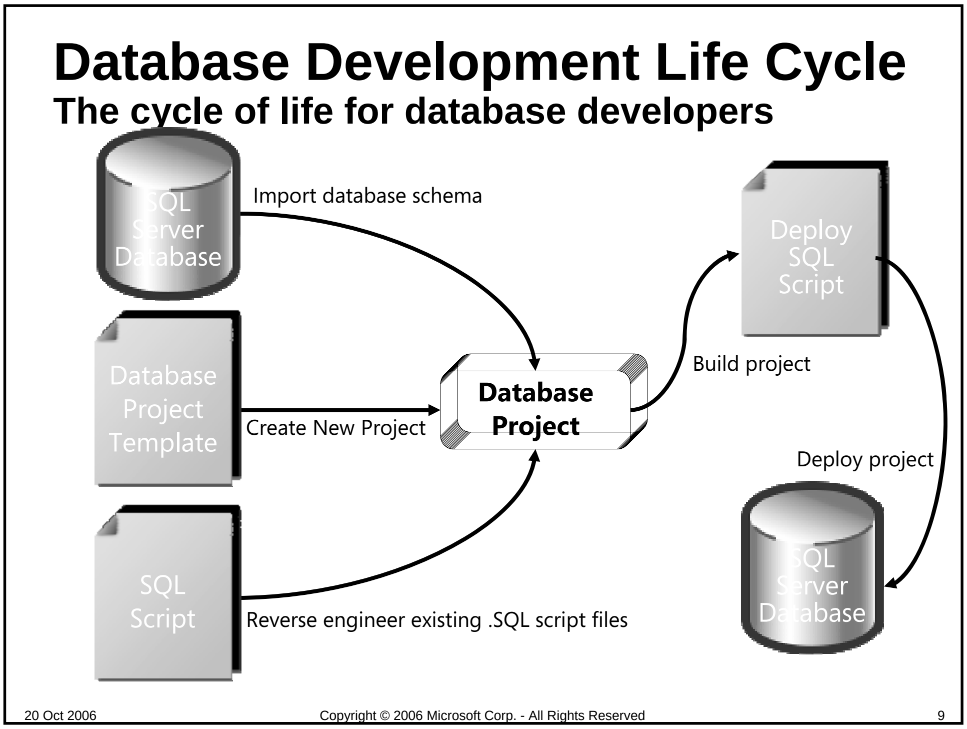
Build

Data Generation

Test

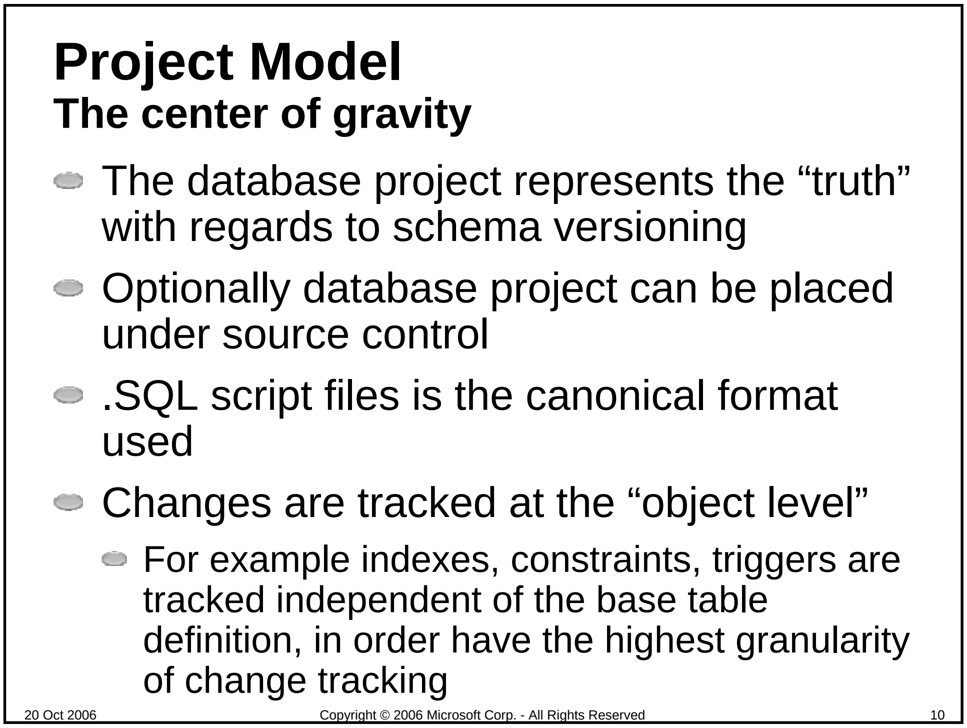
Compare

20 Oct 2006 Copyright © 2006 Microsoft Corp. - All Rights Reserved 8



Database Development Life Cycle

The cycle of life for database developers

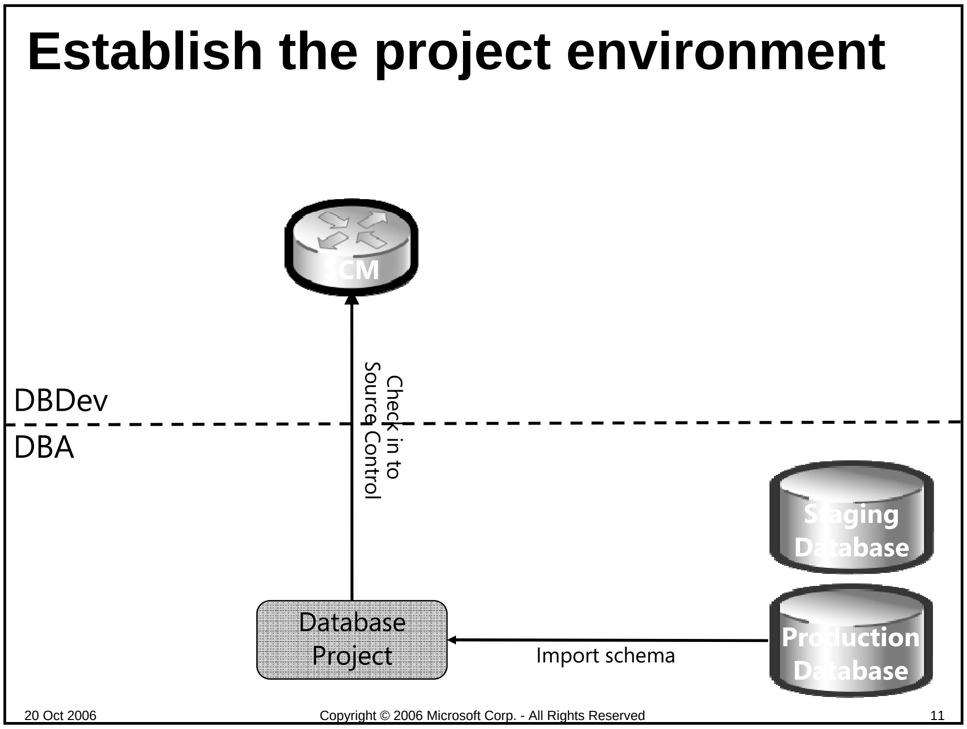


Project Model

The center of gravity

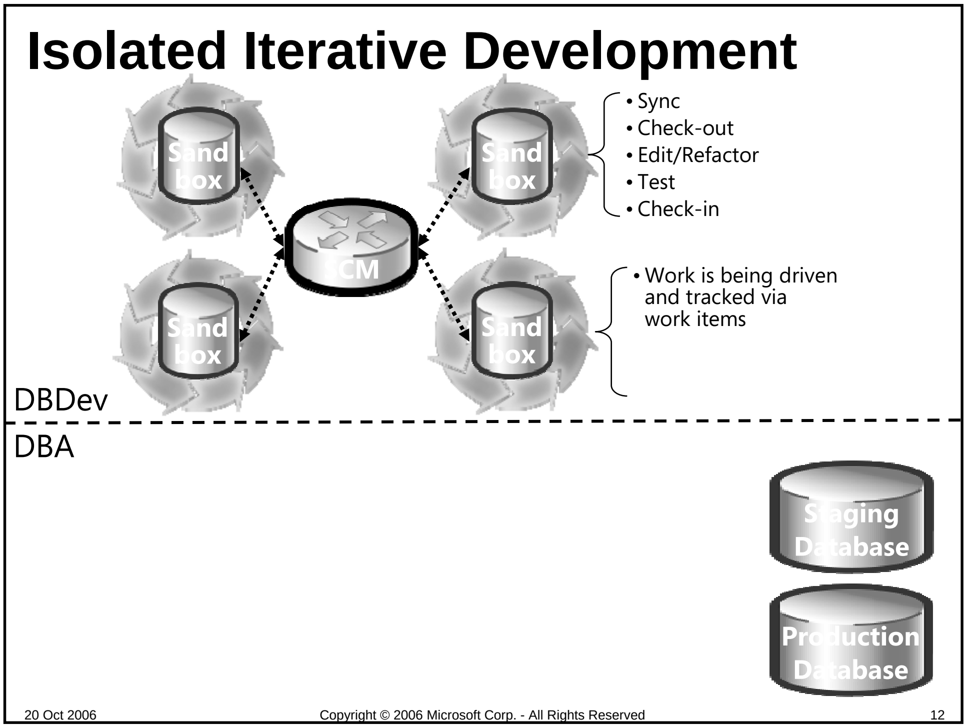
- The database project represents the “truth” with regards to schema versioning
- Optionally database project can be placed under source control
- .SQL script files is the canonical format used
- Changes are tracked at the “object level”
  - For example indexes, constraints, triggers are tracked independent of the base table definition, in order have the highest granularity of change tracking

This presentation is for informational purposes only. Microsoft makes no warranties, express or implied, in this summary.



Establish the project environment

20 Oct 2006 Copyright © 2006 Microsoft Corp. - All Rights Reserved 11



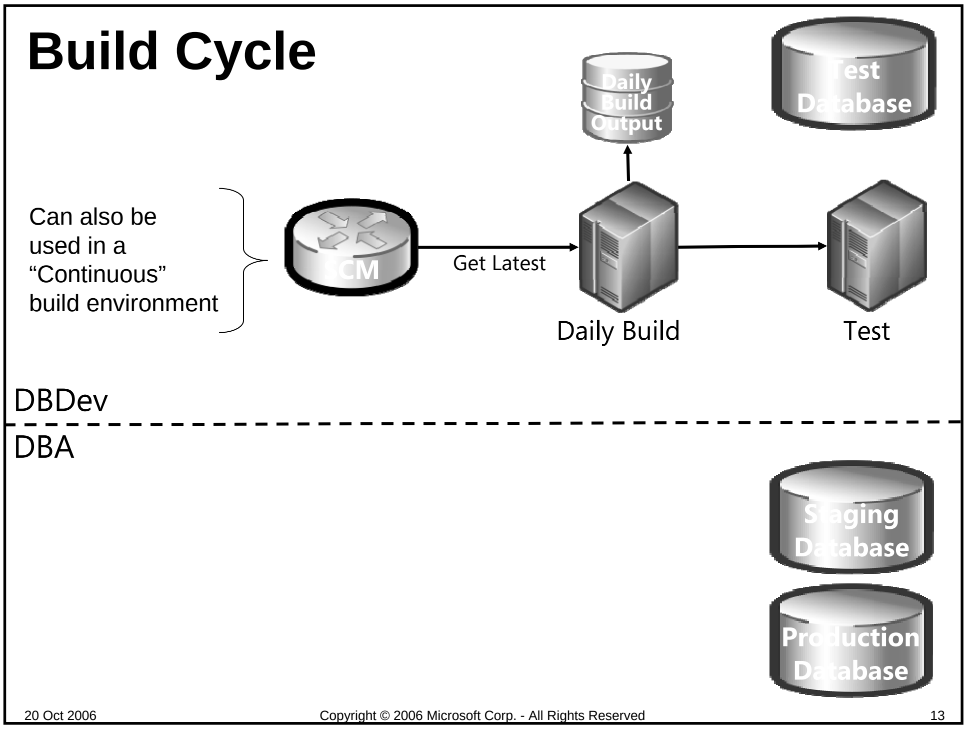
Isolated Iterative Development

- Sync
- Check-out
- Edit/Refactor
- Test
- Check-in

- Work is being driven and tracked via work items

20 Oct 2006 Copyright © 2006 Microsoft Corp. - All Rights Reserved 12

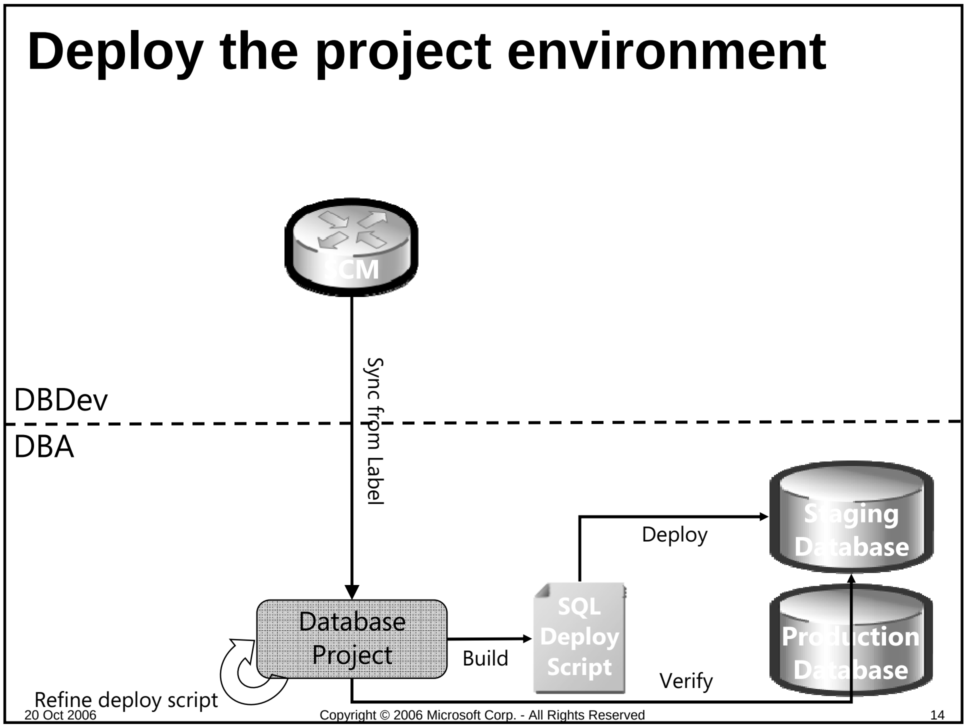
This presentation is for informational purposes only. Microsoft makes no warranties, express or implied, in this summary.



Build Cycle

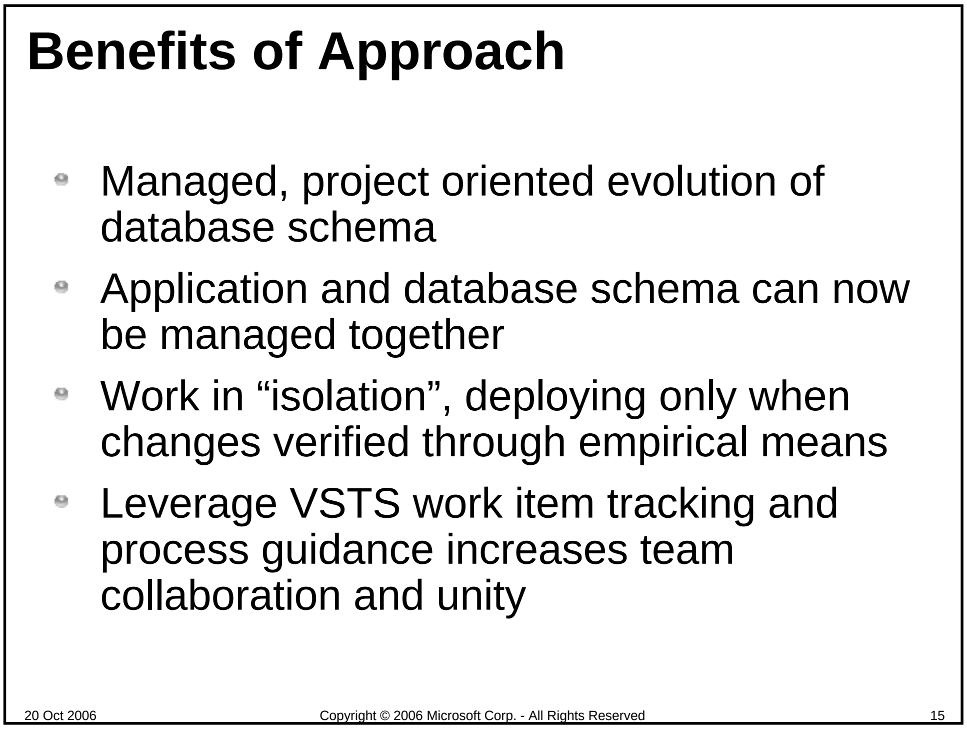
Can also be used in a “Continuous” build environment

20 Oct 2006 Copyright © 2006 Microsoft Corp. - All Rights Reserved 13



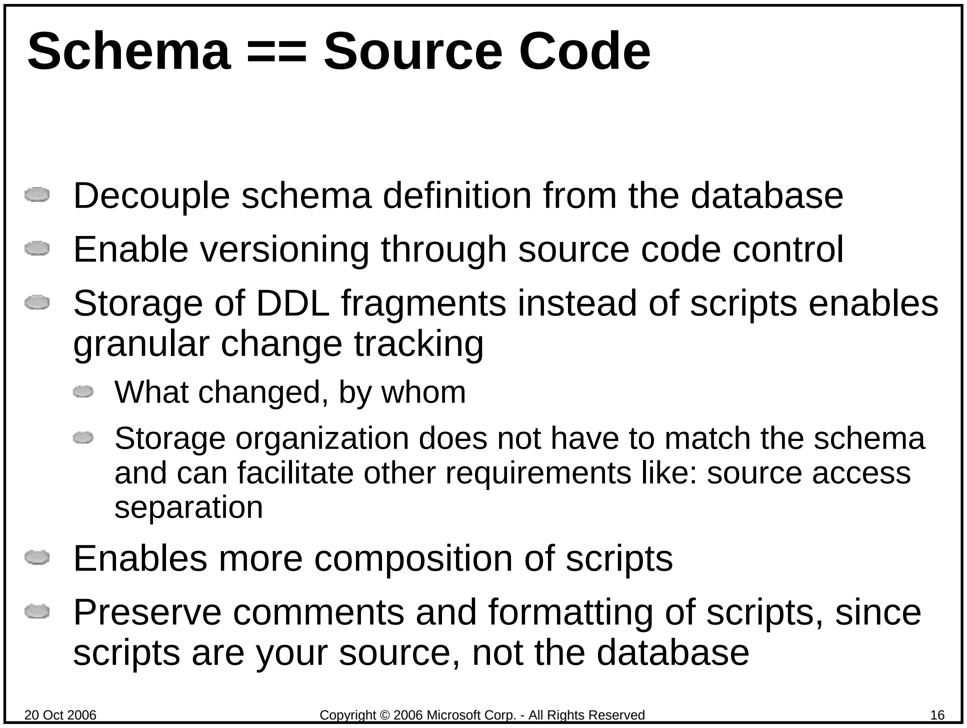
Deploy the project environment

20 Oct 2006 Copyright © 2006 Microsoft Corp. - All Rights Reserved 14



Benefits of Approach

- Managed, project oriented evolution of database schema
- Application and database schema can now be managed together
- Work in “isolation”, deploying only when changes verified through empirical means
- Leverage VSTS work item tracking and process guidance increases team collaboration and unity



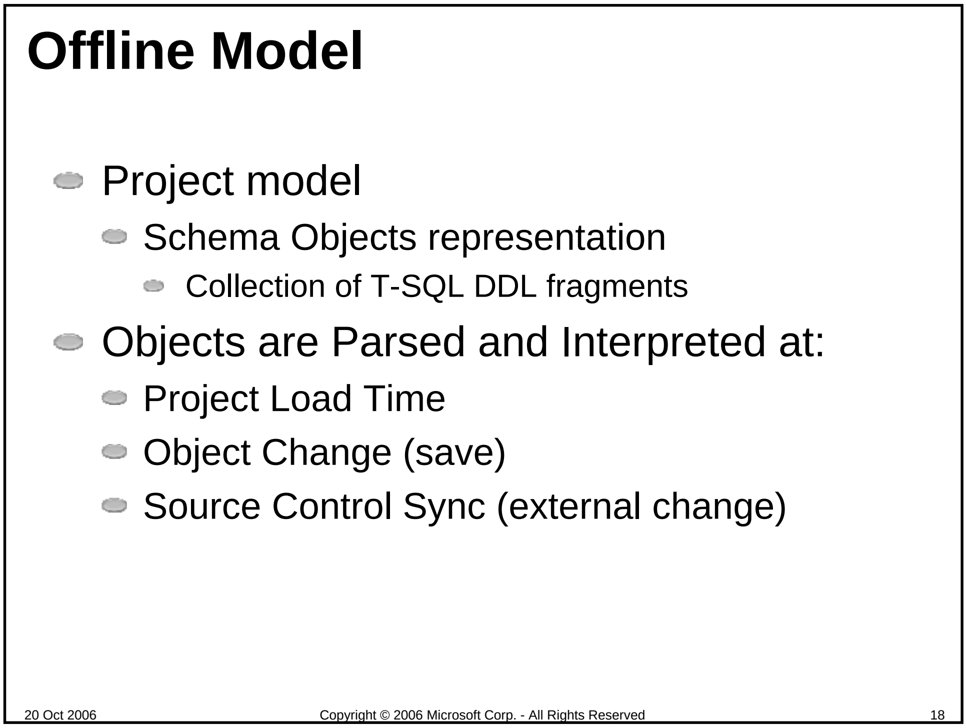
Schema == Source Code

- Decouple schema definition from the database
- Enable versioning through source code control
- Storage of DDL fragments instead of scripts enables granular change tracking
  - What changed, by whom
  - Storage organization does not have to match the schema and can facilitate other requirements like: source access separation
- Enables more composition of scripts
- Preserve comments and formatting of scripts, since scripts are your source, not the database

This presentation is for informational purposes only. Microsoft makes no warranties, express or implied, in this summary.

demo

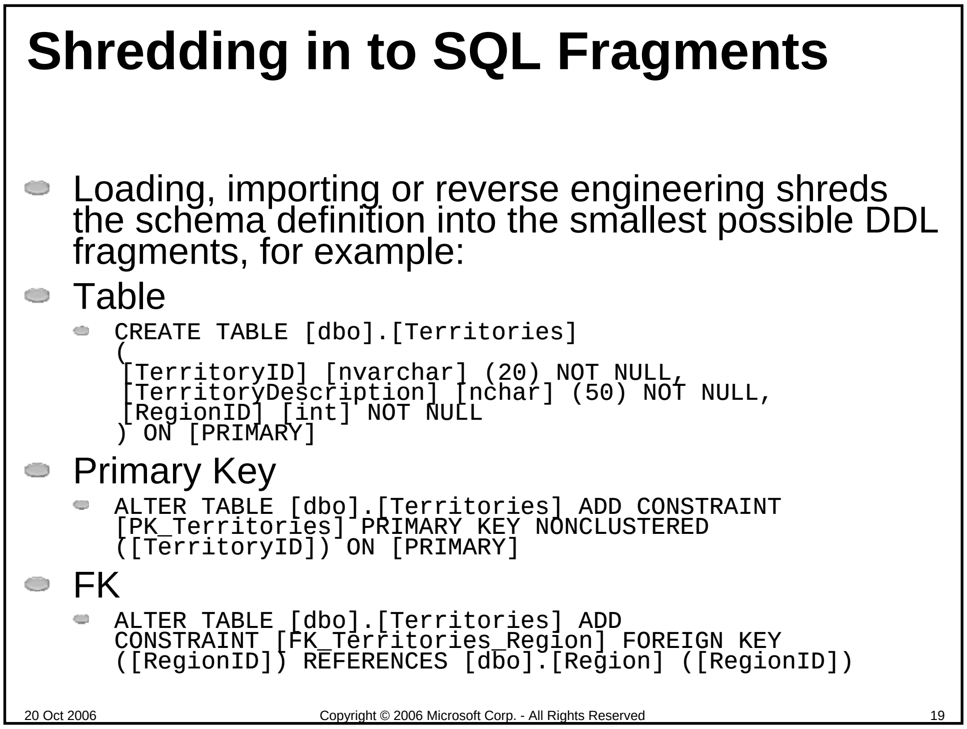
Establish the Project



Offline Model

- Project model
  - Schema Objects representation
    - Collection of T-SQL DDL fragments
- Objects are Parsed and Interpreted at:
  - Project Load Time
  - Object Change (save)
  - Source Control Sync (external change)

This presentation is for informational purposes only. Microsoft makes no warranties, express or implied, in this summary.



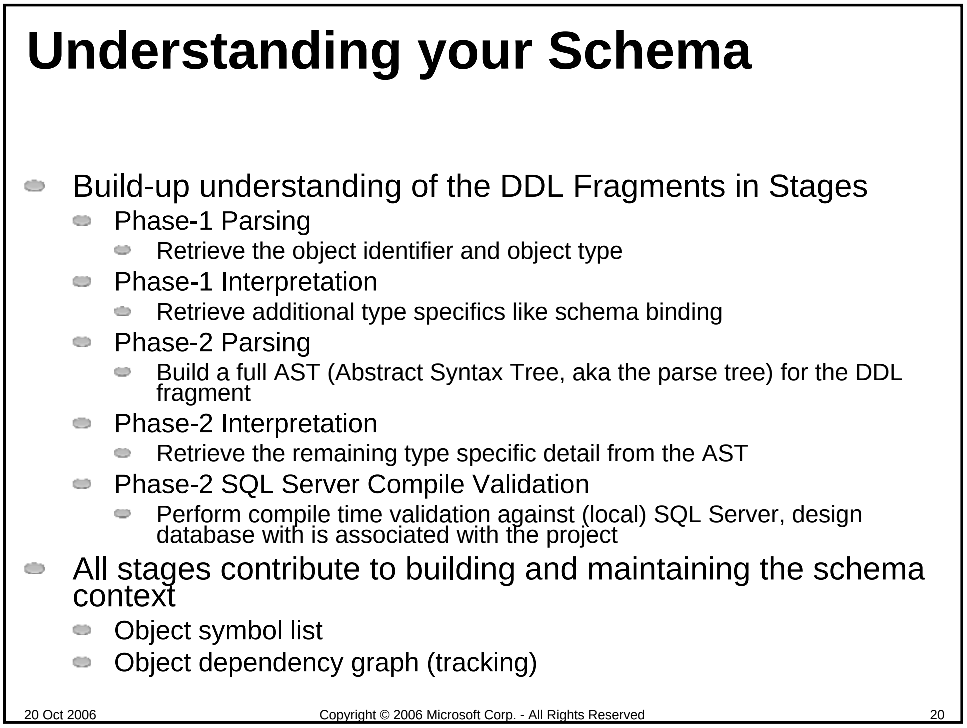
Shredding in to SQL Fragments

- Loading, importing or reverse engineering shreds the schema definition into the smallest possible DDL fragments, for example:
- Table

  -
    ```
    CREATE TABLE [dbo].[Territories]
    (
     [TerritoryID] [nvarchar] (20) NOT NULL,
     [TerritoryDescription] [nchar] (50) NOT NULL,
     [RegionID] [int] NOT NULL
    ) ON [PRIMARY]
    ```
- Primary Key

  -
    ```
    ALTER TABLE [dbo].[Territories] ADD CONSTRAINT
    [PK_Territories] PRIMARY KEY NONCLUSTERED
    ([TerritoryID]) ON [PRIMARY]
    ```
- FK

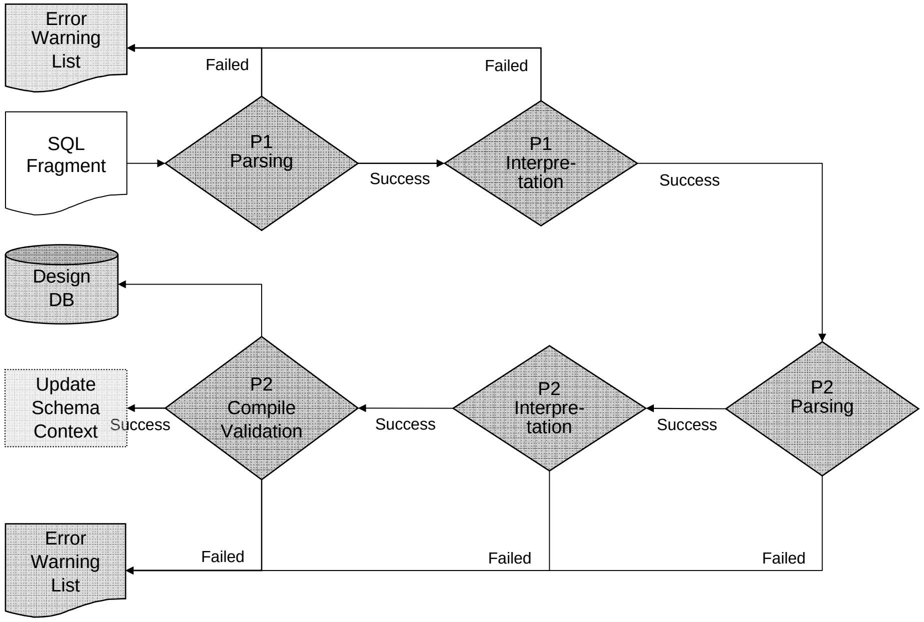
  -
    ```
    ALTER TABLE [dbo].[Territories] ADD
    CONSTRAINT [FK_Territories_Region] FOREIGN KEY
    ([RegionID]) REFERENCES [dbo].[Region] ([RegionID])
    ```



Understanding your Schema

- Build-up understanding of the DDL Fragments in Stages
  - Phase-1 Parsing
    - Retrieve the object identifier and object type
  - Phase-1 Interpretation
    - Retrieve additional type specifics like schema binding
  - Phase-2 Parsing
    - Build a full AST (Abstract Syntax Tree, aka the parse tree) for the DDL fragment
  - Phase-2 Interpretation
    - Retrieve the remaining type specific detail from the AST
  - Phase-2 SQL Server Compile Validation
    - Perform compile time validation against (local) SQL Server, design database with is associated with the project
- All stages contribute to building and maintaining the schema context
  - Object symbol list
  - Object dependency graph (tracking)

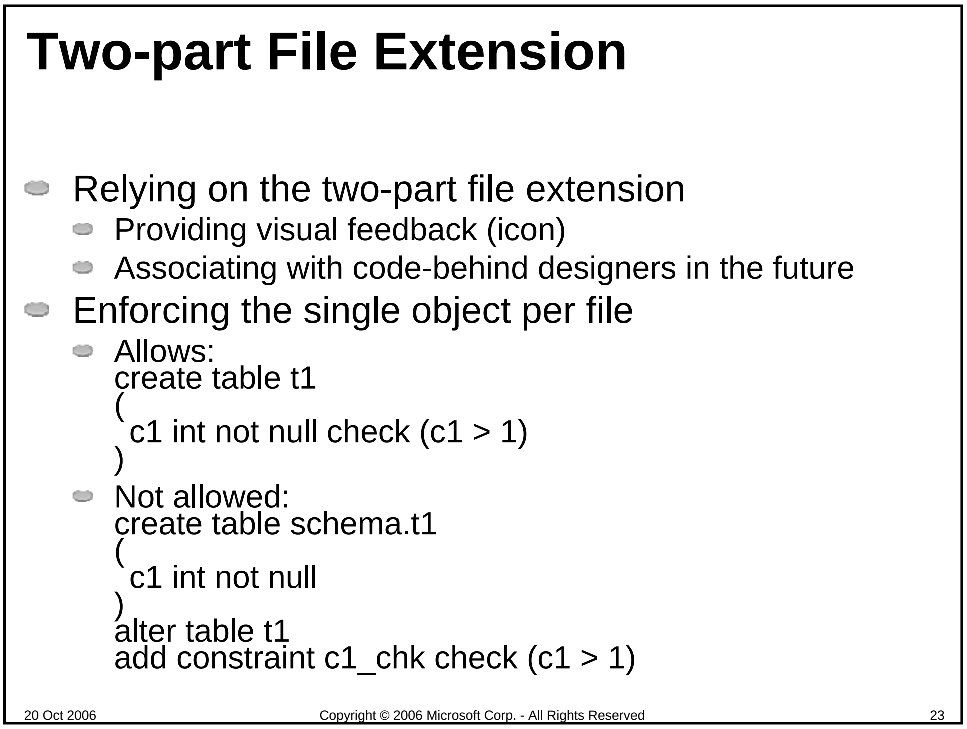
Understanding Schema Objects



File Naming & Extension Scheme

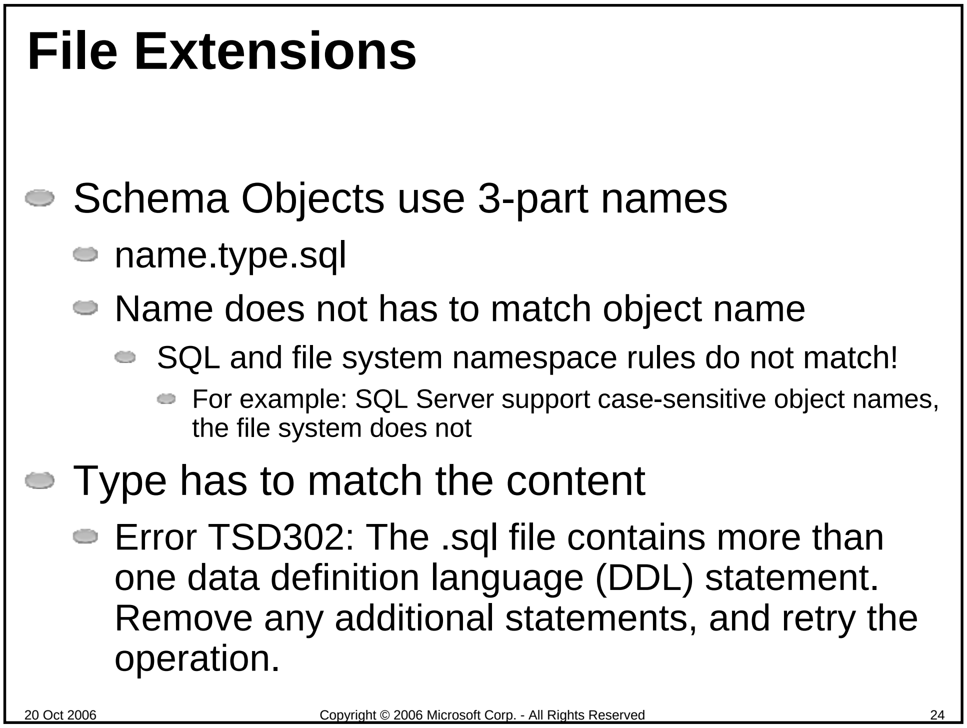
- Everything is a .SQL file
  - Associated with the T-SQL editor
- Using a two part naming scheme to identify types
  - This is not required, but helps identification of types
- By default the file name encodes the object name
  - Not required
    - Filename do not have to match the containing type name
    - Required since SQL Server namespace restrictions do not match the file system naming restrictions

This presentation is for informational purposes only. Microsoft makes no warranties, express or implied, in this summary.



Two-part File Extension

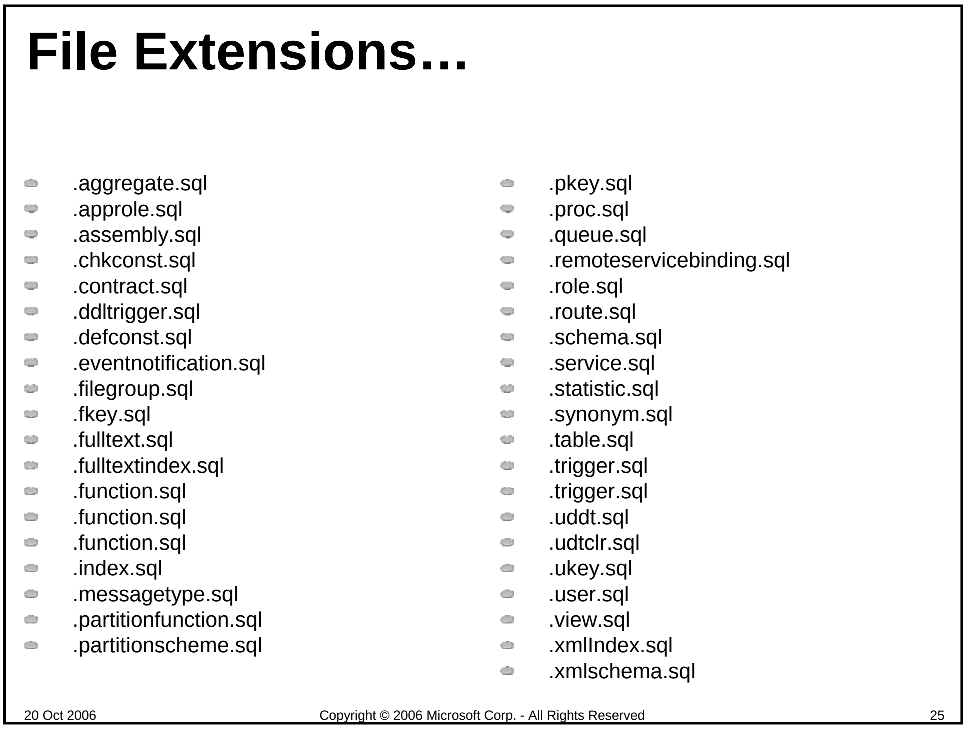
- Relying on the two-part file extension
  - Providing visual feedback (icon)
  - Associating with code-behind designers in the future
- Enforcing the single object per file
  - Allows:
    ```
    create table t1
    (
     c1 int not null check (c1 > 1)
    )
    ```
  - Not allowed:
    ```
    create table schema.t1
    (
     c1 int not null
    )
    alter table t1
    add constraint c1_chk check (c1 > 1)
    ```



File Extensions

- Schema Objects use 3-part names
  - name.type.sql
  - Name does not has to match object name
    - SQL and file system namespace rules do not match!
      - For example: SQL Server support case-sensitive object names, the file system does not
- Type has to match the content
  - Error TSD302: The .sql file contains more than one data definition language (DDL) statement. Remove any additional statements, and retry the operation.

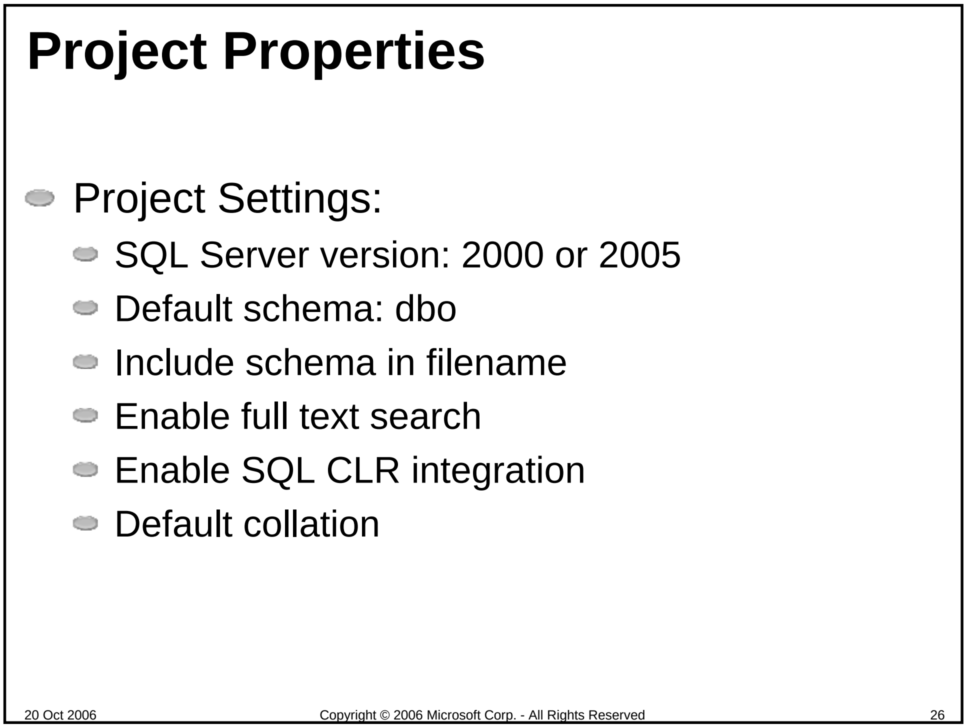
This presentation is for informational purposes only. Microsoft makes no warranties, express or implied, in this summary.



File Extensions…

- .aggregate.sql
- .approle.sql
- .assembly.sql
- .chkconst.sql
- .contract.sql
- .ddltrigger.sql
- .defconst.sql
- .eventnotification.sql
- .filegroup.sql
- .fkey.sql
- .fulltext.sql
- .fulltextindex.sql
- .function.sql
- .function.sql
- .function.sql
- .index.sql
- .messagetype.sql
- .partitionfunction.sql
- .partitionscheme.sql
- .pkey.sql
- .proc.sql
- .queue.sql
- .remoteservicebinding.sql
- .role.sql
- .route.sql
- .schema.sql
- .service.sql
- .statistic.sql
- .synonym.sql
- .table.sql
- .trigger.sql
- .trigger.sql
- .uddt.sql
- .udtclr.sql
- .ukey.sql
- .user.sql
- .view.sql
- .xmlIndex.sql
- .xmlschema.sql

20 Oct 2006 Copyright © 2006 Microsoft Corp. - All Rights Reserved 25



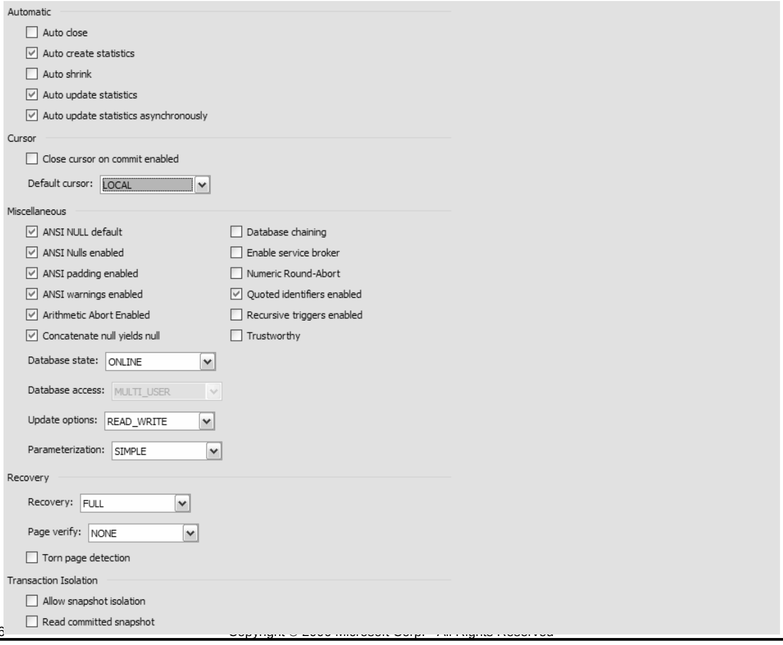
Project Properties

- Project Settings:
  - SQL Server version: 2000 or 2005
  - Default schema: dbo
  - Include schema in filename
  - Enable full text search
  - Enable SQL CLR integration
  - Default collation

20 Oct 2006 Copyright © 2006 Microsoft Corp. - All Rights Reserved 26

This presentation is for informational purposes only. Microsoft makes no warranties, express or implied, in this summary.

Project Properties…



Automatic

- ☐ Auto close
- ☑ Auto create statistics
- ☐ Auto shrink
- ☑ Auto update statistics
- ☑ Auto update statistics asynchronously

Cursor

- ☐ Close cursor on commit enabled
- Default cursor: LOCAL

Miscellaneous

- ☑ ANSI NULL default
- ☑ ANSI Nulls enabled
- ☑ ANSI padding enabled
- ☑ ANSI warnings enabled
- ☑ Arithmetic Abort Enabled
- ☑ Concatenate null yields null
- ☐ Database chaining
- ☐ Enable service broker
- ☐ Numeric Round-Abort
- ☑ Quoted identifiers enabled
- ☐ Recursive triggers enabled
- ☐ Trustworthy
- Database state: ONLINE
- Database access: MULTI_USER
- Update options: READ_WRITE
- Parameterization: SIMPLE

Recovery

- Recovery: FULL
- Page verify: NONE
- ☐ Torn page detection

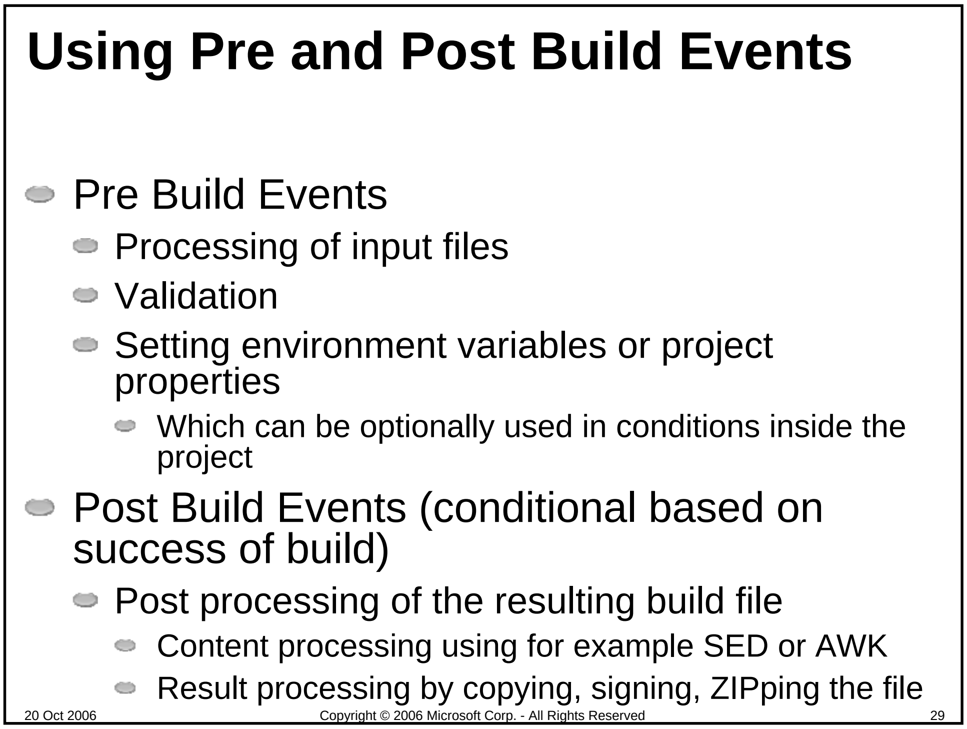
Transaction Isolation

- ☐ Allow snapshot isolation
- ☐ Read committed snapshot

Project Properties…

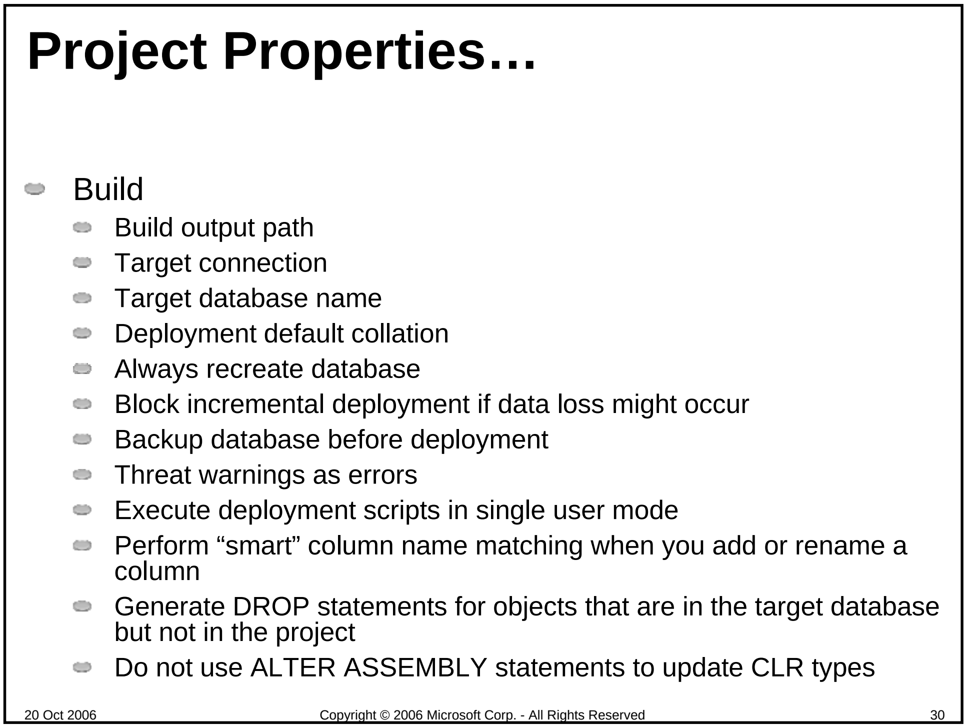
- Build Events
  - Pre-build event command line
  - Post-build event command line
  - Run the post-build events on: success **|** always

This presentation is for informational purposes only. Microsoft makes no warranties, express or implied, in this summary.



Using Pre and Post Build Events

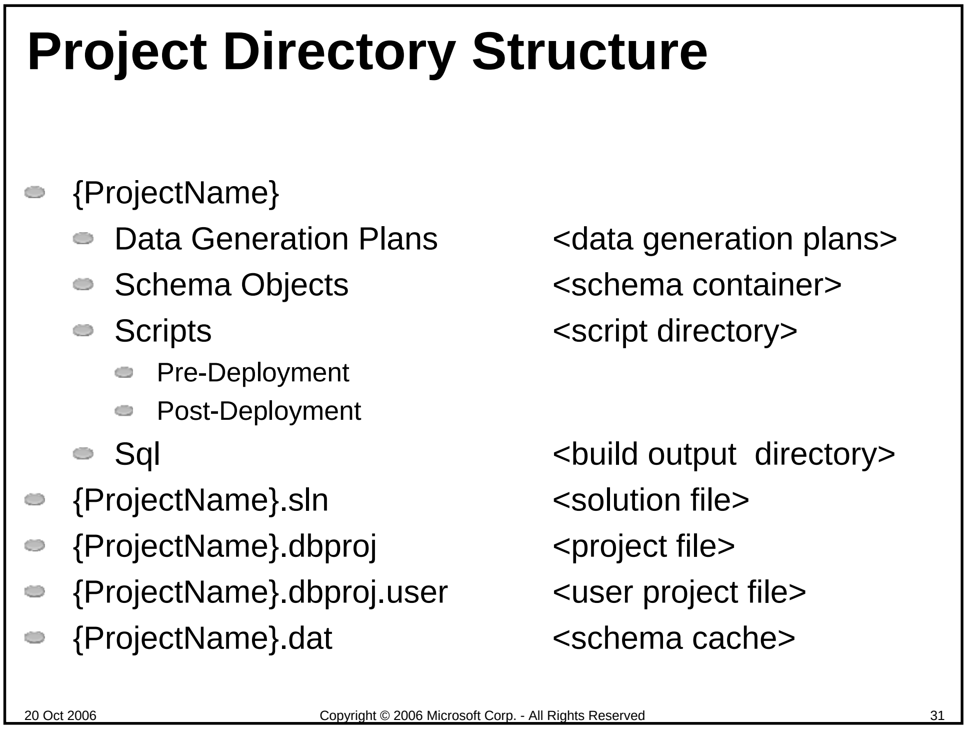
- Pre Build Events
  - Processing of input files
  - Validation
  - Setting environment variables or project properties
    - Which can be optionally used in conditions inside the project
- Post Build Events (conditional based on success of build)
  - Post processing of the resulting build file
    - Content processing using for example SED or AWK
    - Result processing by copying, signing, ZIPping the file



Project Properties…

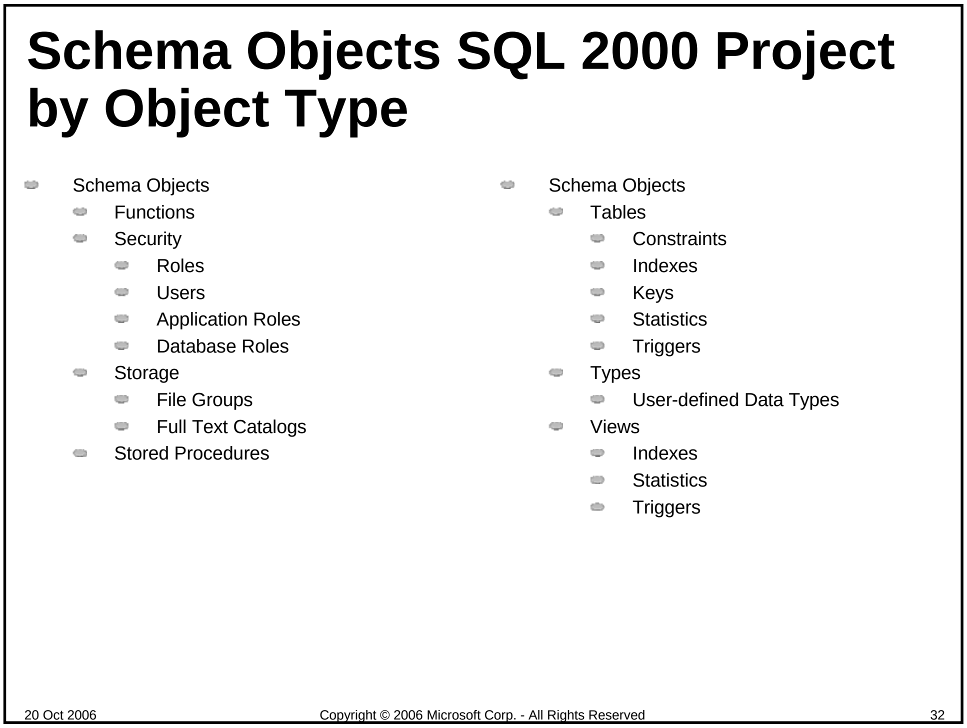
- Build
  - Build output path
  - Target connection
  - Target database name
  - Deployment default collation
  - Always recreate database
  - Block incremental deployment if data loss might occur
  - Backup database before deployment
  - Threat warnings as errors
  - Execute deployment scripts in single user mode
  - Perform “smart” column name matching when you add or rename a column
  - Generate DROP statements for objects that are in the target database but not in the project
  - Do not use ALTER ASSEMBLY statements to update CLR types

This presentation is for informational purposes only. Microsoft makes no warranties, express or implied, in this summary.



Project Directory Structure

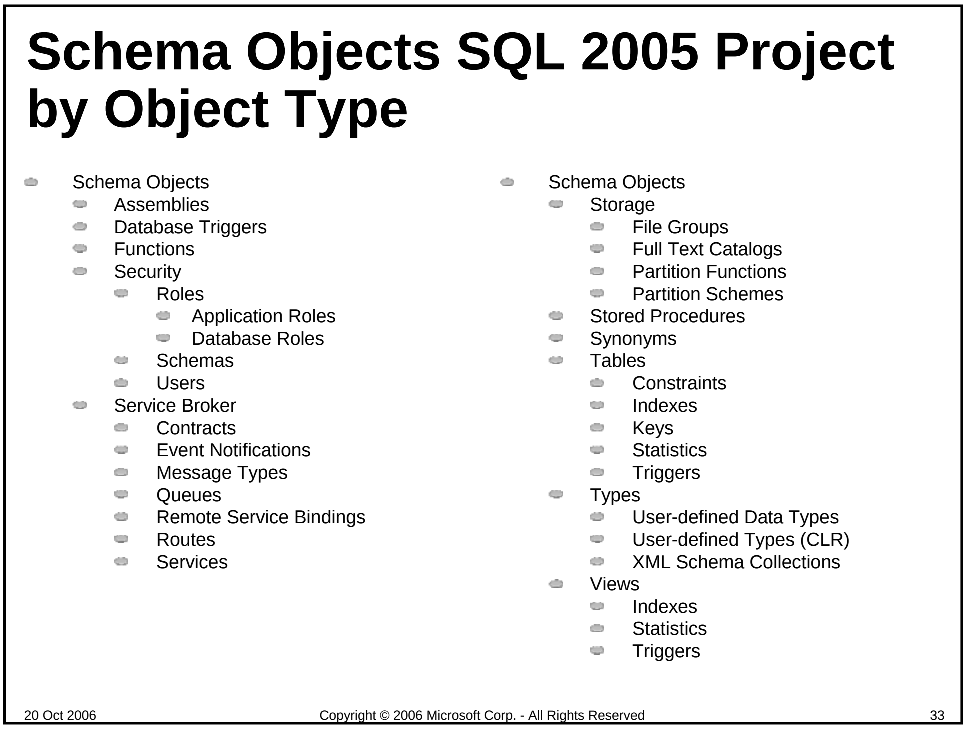
- {ProjectName}
  - Data Generation Plans <data generation plans>
  - Schema Objects <schema container>
  - Scripts <script directory>
    - Pre-Deployment
    - Post-Deployment
  - Sql <build output directory>
- {ProjectName}.sln <solution file>
- {ProjectName}.dbproj <project file>
- {ProjectName}.dbproj.user <user project file>
- {ProjectName}.dat <schema cache>



Schema Objects SQL 2000 Project by Object Type

- Schema Objects
  - Functions
  - Security
    - Roles
    - Users
    - Application Roles
    - Database Roles
  - Storage
    - File Groups
    - Full Text Catalogs
  - Stored Procedures

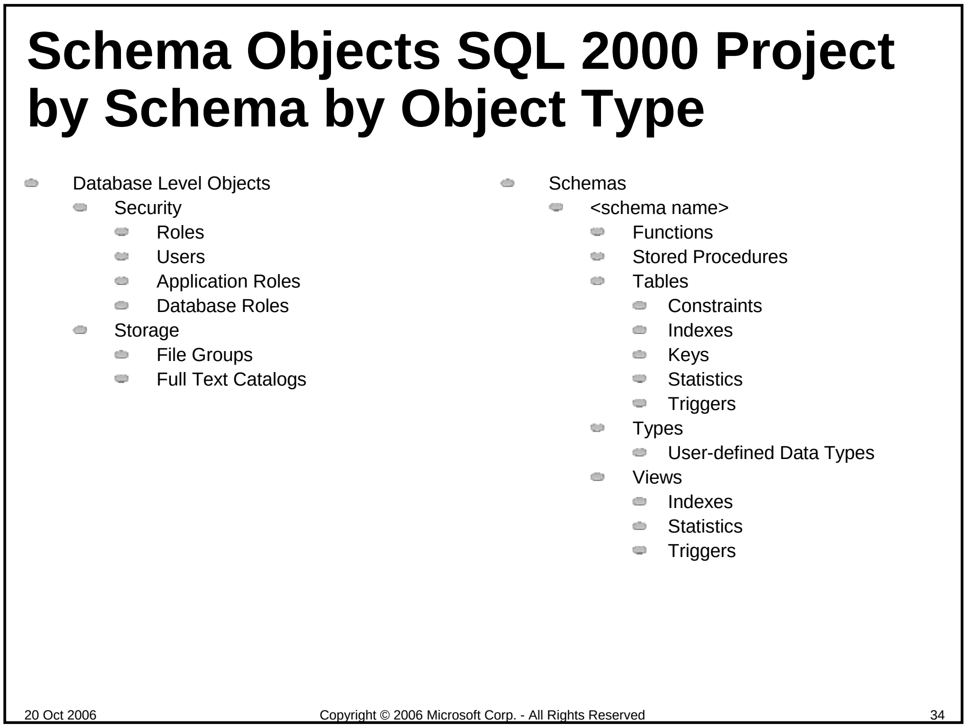
- Schema Objects
  - Tables
    - Constraints
    - Indexes
    - Keys
    - Statistics
    - Triggers
  - Types
    - User-defined Data Types
  - Views
    - Indexes
    - Statistics
    - Triggers



Schema Objects SQL 2005 Project by Object Type

- Schema Objects
  - Assemblies
  - Database Triggers
  - Functions
  - Security
    - Roles
      - Application Roles
      - Database Roles
    - Schemas
    - Users
  - Service Broker
    - Contracts
    - Event Notifications
    - Message Types
    - Queues
    - Remote Service Bindings
    - Routes
    - Services

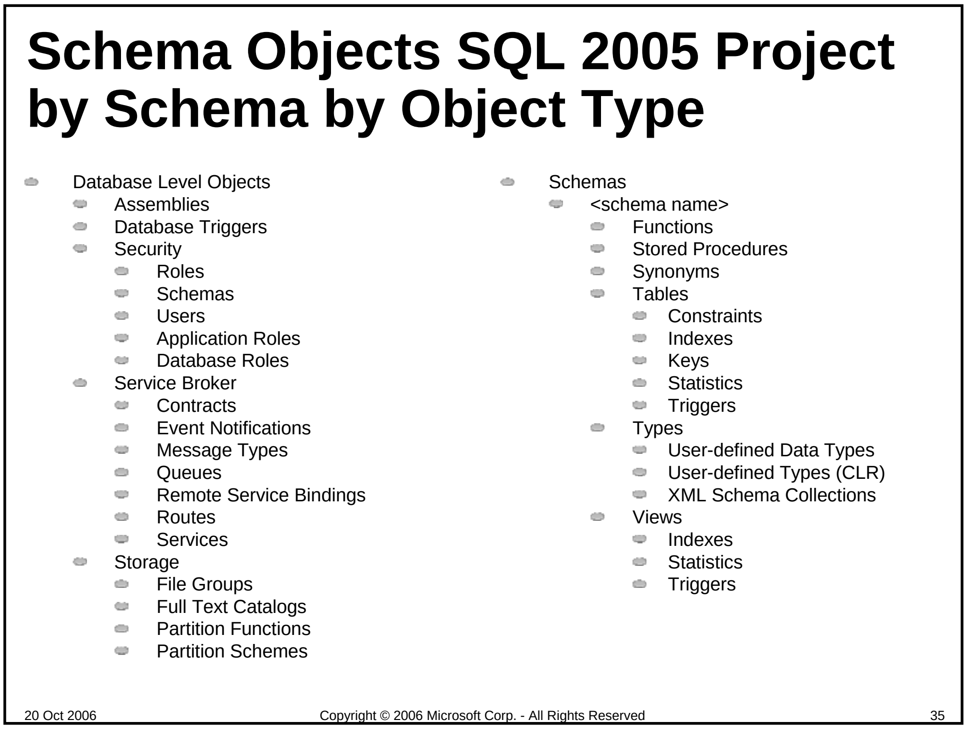
- Schema Objects
  - Storage
    - File Groups
    - Full Text Catalogs
    - Partition Functions
    - Partition Schemes
  - Stored Procedures
  - Synonyms
  - Tables
    - Constraints
    - Indexes
    - Keys
    - Statistics
    - Triggers
  - Types
    - User-defined Data Types
    - User-defined Types (CLR)
    - XML Schema Collections
  - Views
    - Indexes
    - Statistics
    - Triggers



Schema Objects SQL 2000 Project by Schema by Object Type

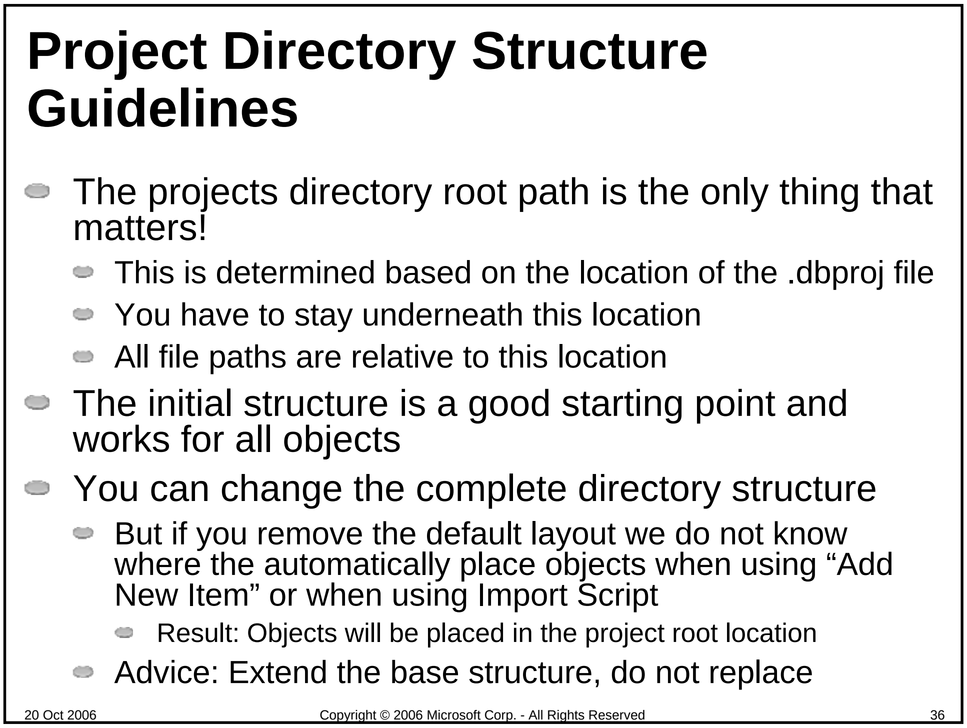
- Database Level Objects
  - Security
    - Roles
    - Users
    - Application Roles
    - Database Roles
  - Storage
    - File Groups
    - Full Text Catalogs

- Schemas
  - <schema name>
    - Functions
    - Stored Procedures
    - Tables
      - Constraints
      - Indexes
      - Keys
      - Statistics
      - Triggers
    - Types
      - User-defined Data Types
    - Views
      - Indexes
      - Statistics
      - Triggers



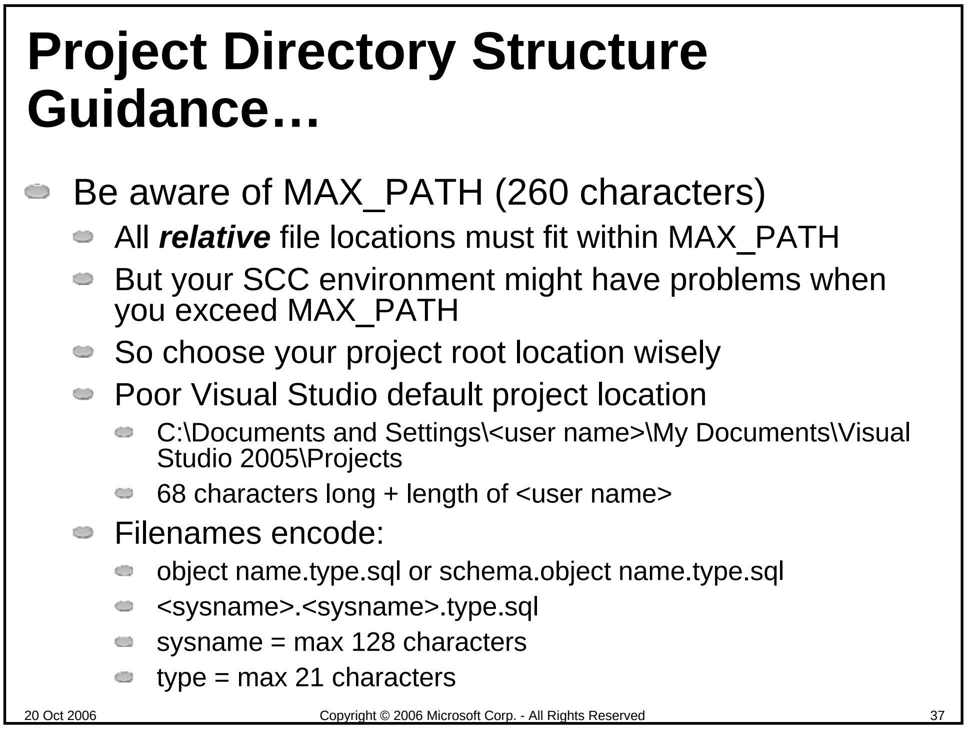
Schema Objects SQL 2005 Project by Schema by Object Type

- Database Level Objects
  - Assemblies
  - Database Triggers
  - Security
    - Roles
    - Schemas
    - Users
    - Application Roles
    - Database Roles
  - Service Broker
    - Contracts
    - Event Notifications
    - Message Types
    - Queues
    - Remote Service Bindings
    - Routes
    - Services
  - Storage
    - File Groups
    - Full Text Catalogs
    - Partition Functions
    - Partition Schemes
- Schemas
  - <schema name>
    - Functions
    - Stored Procedures
    - Synonyms
    - Tables
      - Constraints
      - Indexes
      - Keys
      - Statistics
      - Triggers
    - Types
      - User-defined Data Types
      - User-defined Types (CLR)
      - XML Schema Collections
    - Views
      - Indexes
      - Statistics
      - Triggers



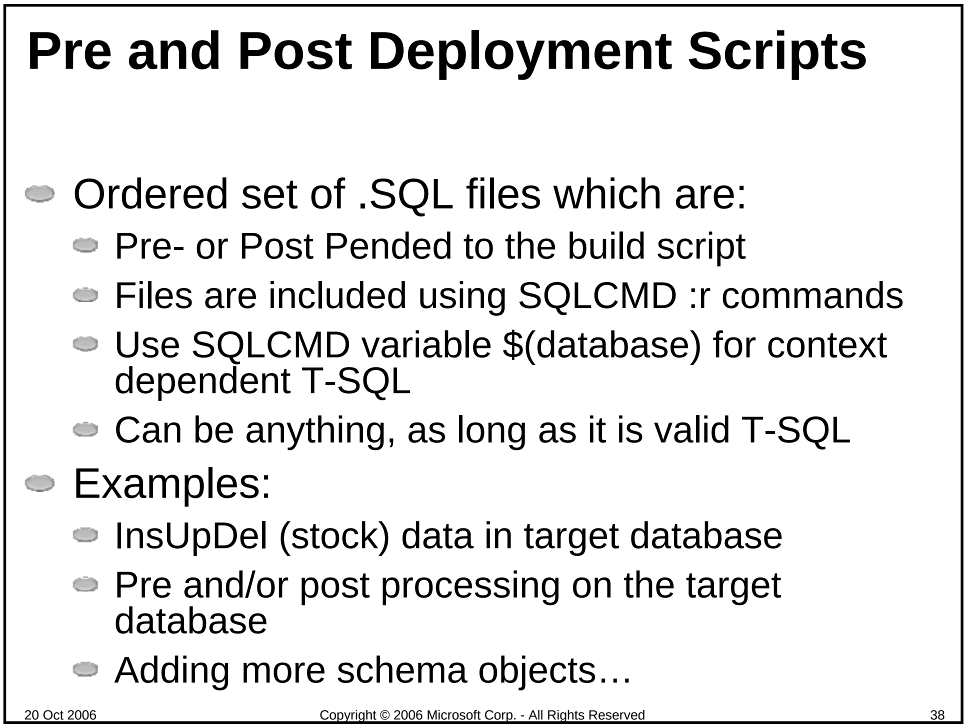
Project Directory Structure Guidelines

- The projects directory root path is the only thing that matters!
  - This is determined based on the location of the .dbproj file
  - You have to stay underneath this location
  - All file paths are relative to this location
- The initial structure is a good starting point and works for all objects
- You can change the complete directory structure
  - But if you remove the default layout we do not know where the automatically place objects when using “Add New Item” or when using Import Script
    - Result: Objects will be placed in the project root location
  - Advice: Extend the base structure, do not replace



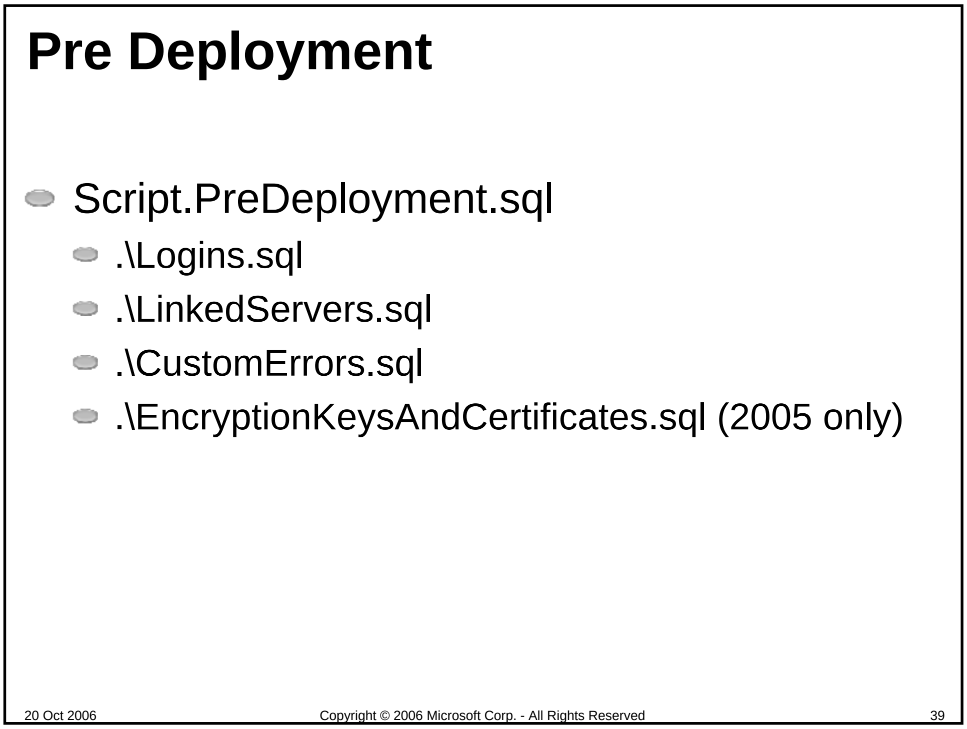
Project Directory Structure Guidance…

- Be aware of MAX_PATH (260 characters)
  - All ***relative*** file locations must fit within MAX_PATH
  - But your SCC environment might have problems when you exceed MAX_PATH
  - So choose your project root location wisely
  - Poor Visual Studio default project location
    - C:\Documents and Settings\<user name>\My Documents\Visual Studio 2005\Projects
    - 68 characters long + length of <user name>
  - Filenames encode:
    - object name.type.sql or schema.object name.type.sql
    - <sysname>.<sysname>.type.sql
    - sysname = max 128 characters
    - type = max 21 characters



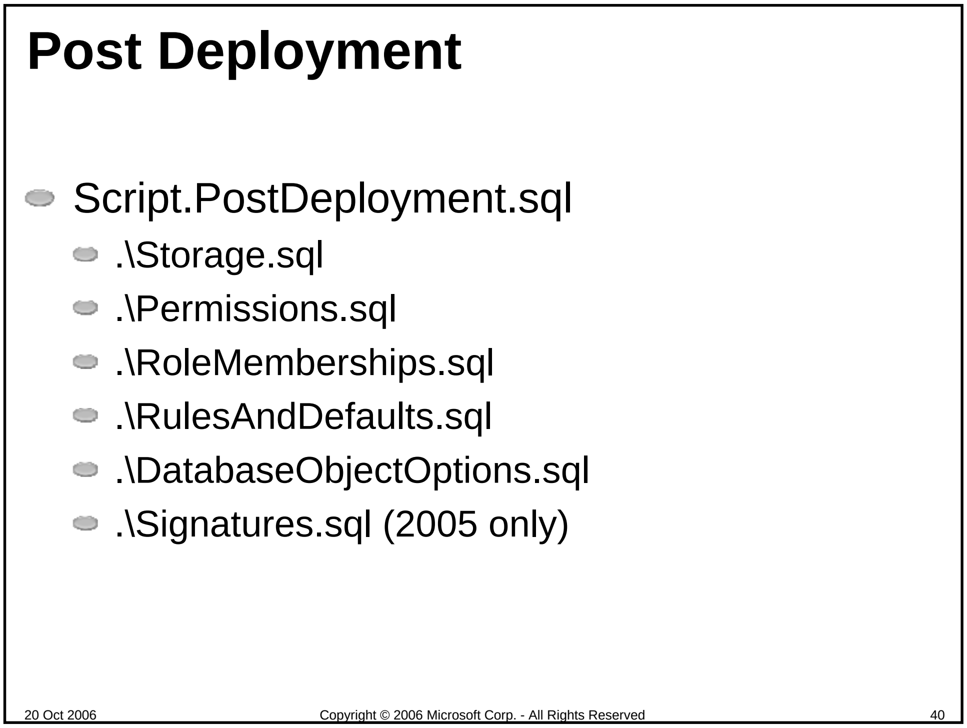
Pre and Post Deployment Scripts

- Ordered set of .SQL files which are:
  - Pre- or Post Pended to the build script
  - Files are included using SQLCMD :r commands
  - Use SQLCMD variable $(database) for context dependent T-SQL
  - Can be anything, as long as it is valid T-SQL
- Examples:
  - InsUpDel (stock) data in target database
  - Pre and/or post processing on the target database
  - Adding more schema objects…



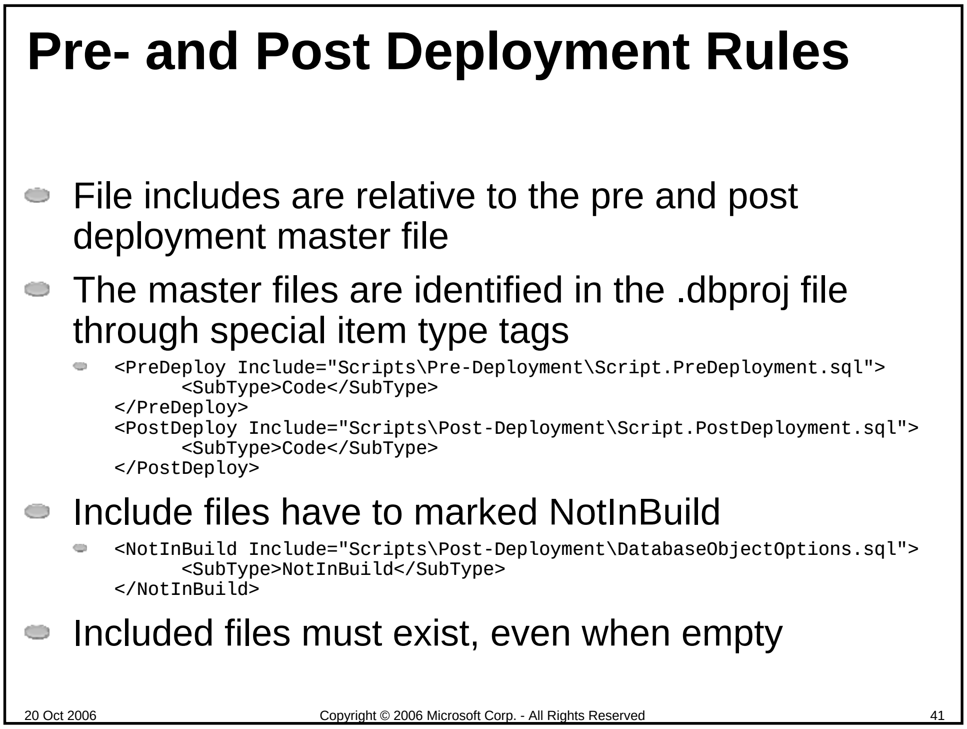
Pre Deployment

- Script.PreDeployment.sql
  - .\Logins.sql
  - .\LinkedServers.sql
  - .\CustomErrors.sql
  - .\EncryptionKeysAndCertificates.sql (2005 only)



Post Deployment

- Script.PostDeployment.sql
  - .\Storage.sql
  - .\Permissions.sql
  - .\RoleMemberships.sql
  - .\RulesAndDefaults.sql
  - .\DatabaseObjectOptions.sql
  - .\Signatures.sql (2005 only)

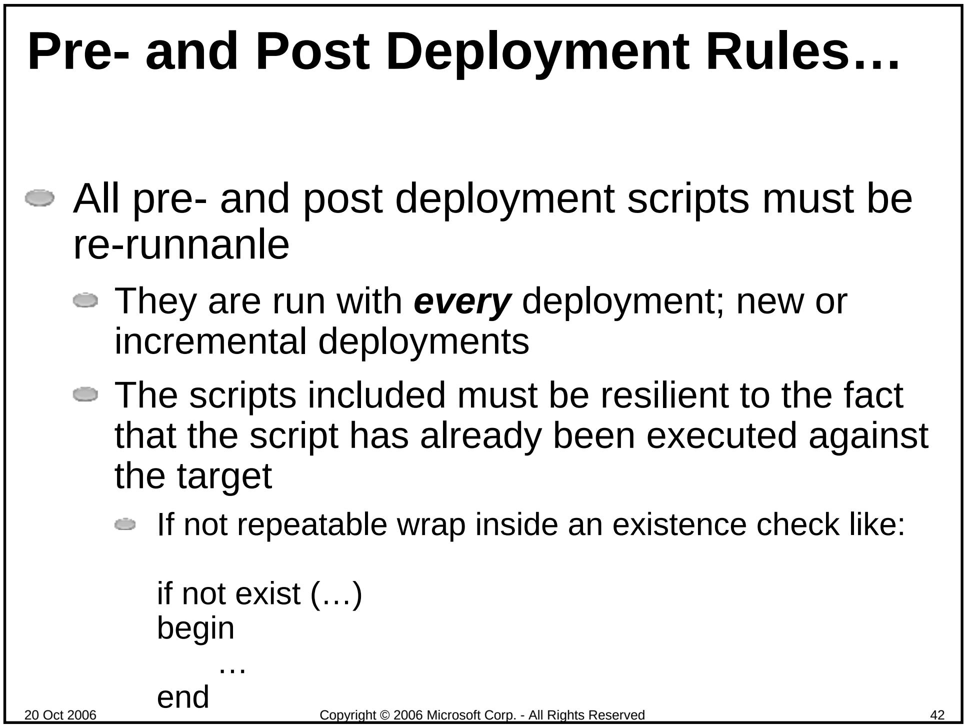


Pre- and Post Deployment Rules

- File includes are relative to the pre and post deployment master file
- The master files are identified in the .dbproj file through special item type tags

  -
    ```
    <PreDeploy Include="Scripts\Pre-Deployment\Script.PreDeployment.sql">
          <SubType>Code</SubType>
    </PreDeploy>
    <PostDeploy Include="Scripts\Post-Deployment\Script.PostDeployment.sql">
          <SubType>Code</SubType>
    </PostDeploy>
    ```
- Include files have to marked NotInBuild

  -
    ```
    <NotInBuild Include="Scripts\Post-Deployment\DatabaseObjectOptions.sql">
          <SubType>NotInBuild</SubType>
    </NotInBuild>
    ```
- Included files must exist, even when empty

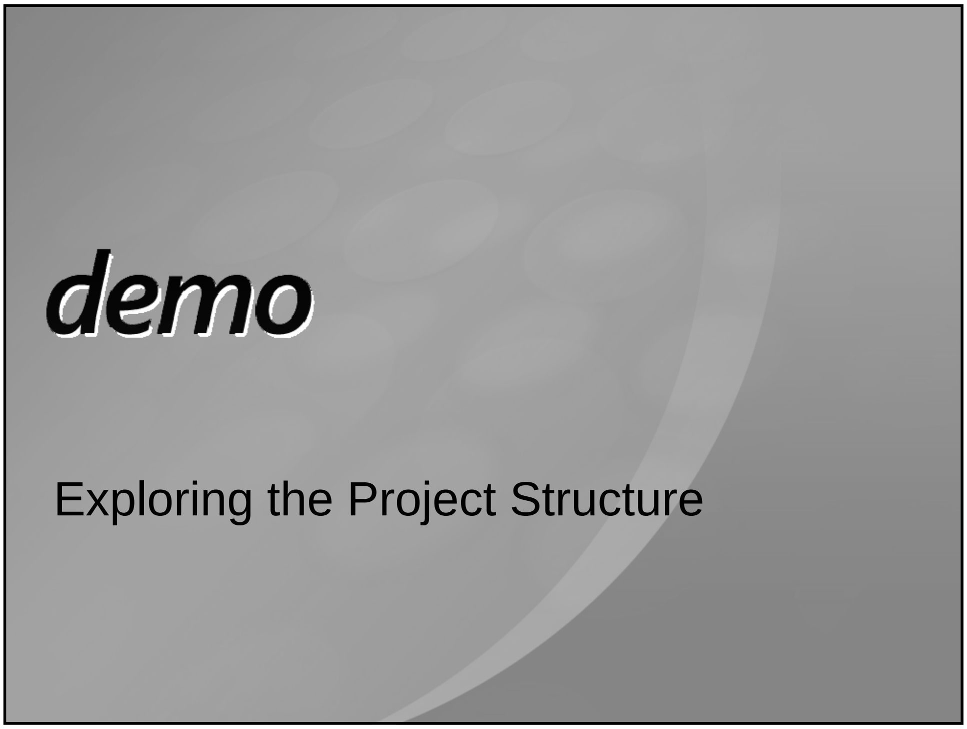


Pre- and Post Deployment Rules…

- All pre- and post deployment scripts must be re-runnanle
  - They are run with ***every*** deployment; new or incremental deployments
  - The scripts included must be resilient to the fact that the script has already been executed against the target
    - If not repeatable wrap inside an existence check like:

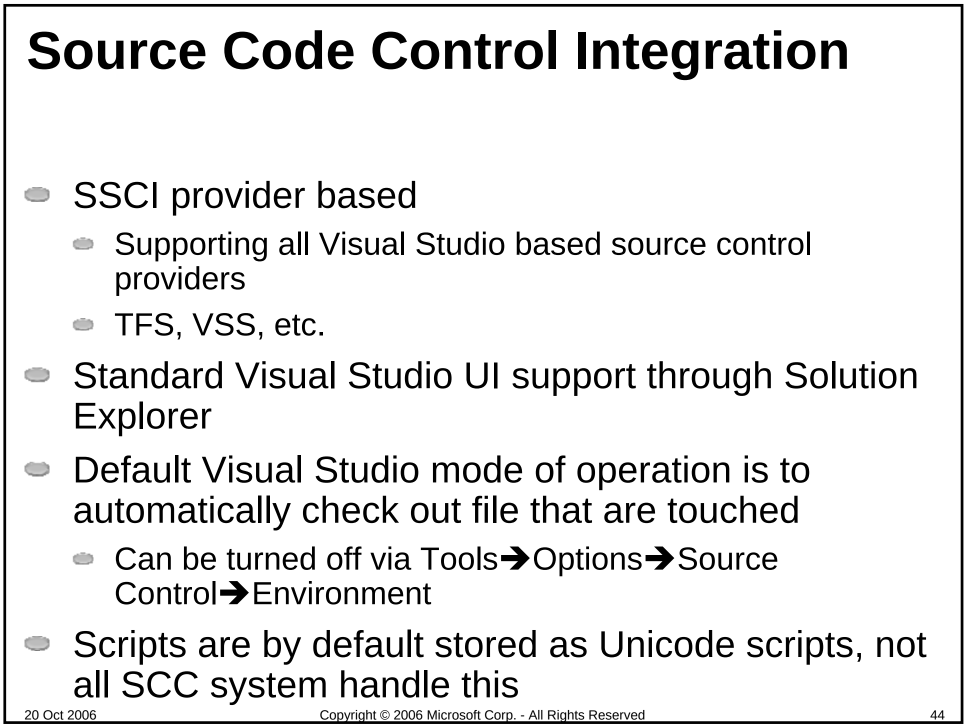
      ```
      if not exist (…)
      begin
           …
      end
      ```

This presentation is for informational purposes only. Microsoft makes no warranties, express or implied, in this summary.



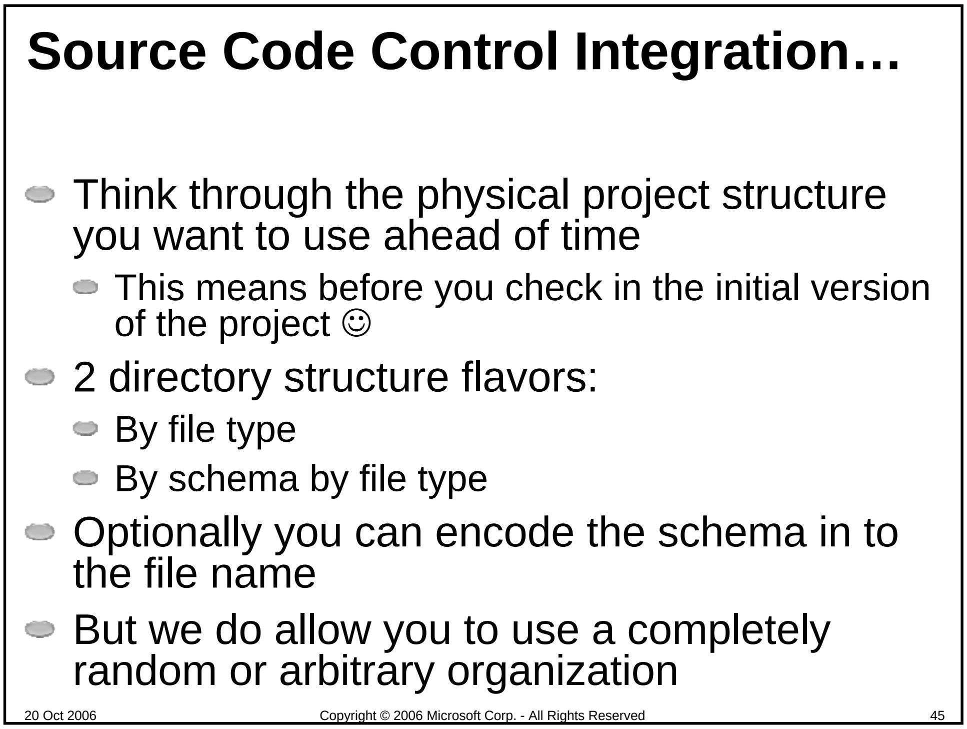
demo

Exploring the Project Structure



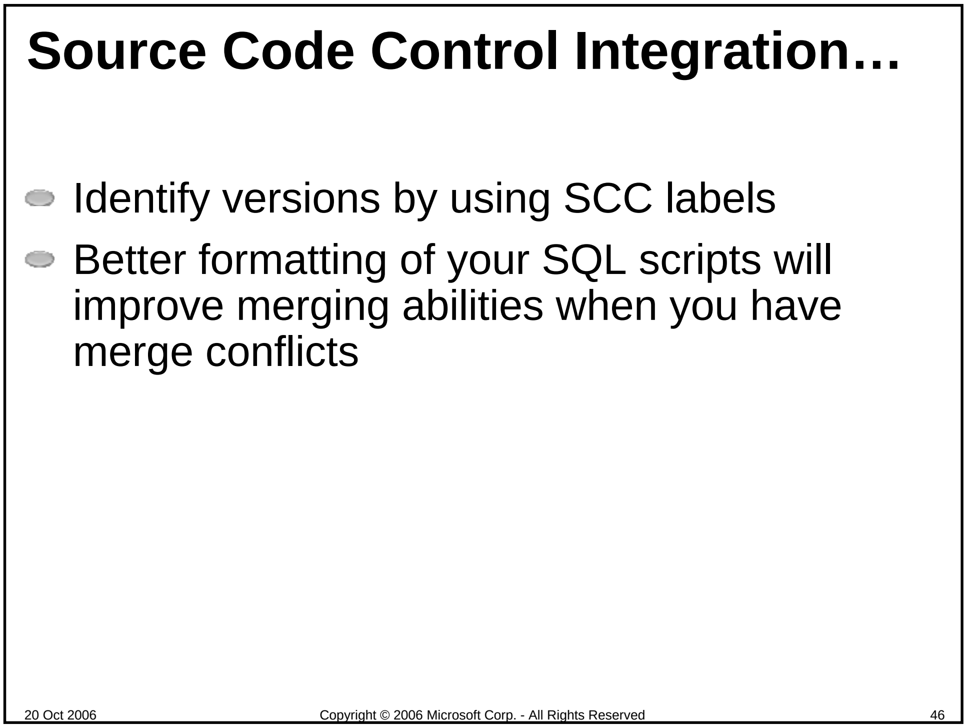
Source Code Control Integration

- SSCI provider based
  - Supporting all Visual Studio based source control providers
  - TFS, VSS, etc.
- Standard Visual Studio UI support through Solution Explorer
- Default Visual Studio mode of operation is to automatically check out file that are touched
  - Can be turned off via Tools➔Options➔Source Control➔Environment
- Scripts are by default stored as Unicode scripts, not all SCC system handle this



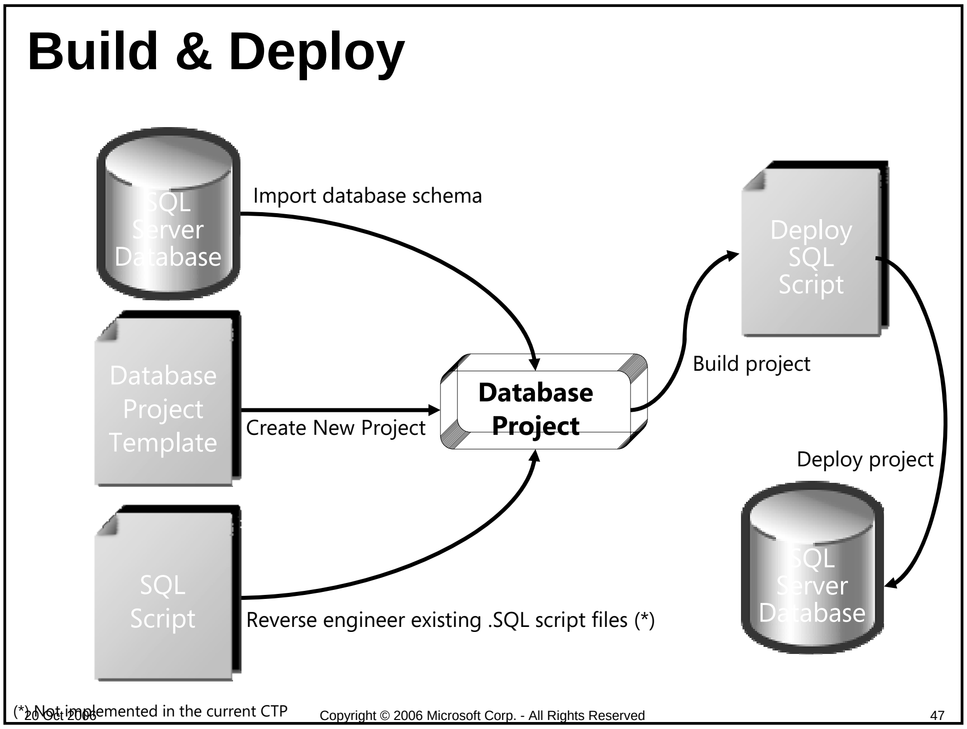
Source Code Control Integration…

- Think through the physical project structure you want to use ahead of time
  - This means before you check in the initial version of the project ☺
- 2 directory structure flavors:
  - By file type
  - By schema by file type
- Optionally you can encode the schema in to the file name
- But we do allow you to use a completely random or arbitrary organization



Source Code Control Integration…

- Identify versions by using SCC labels
- Better formatting of your SQL scripts will improve merging abilities when you have merge conflicts



Build & Deploy

SQL Server Database

Import database schema

Database Project Template

Create New Project

SQL Script

Reverse engineer existing .SQL script files (*)

Database Project

Build project

Deploy SQL Script

Deploy project

SQL Server Database

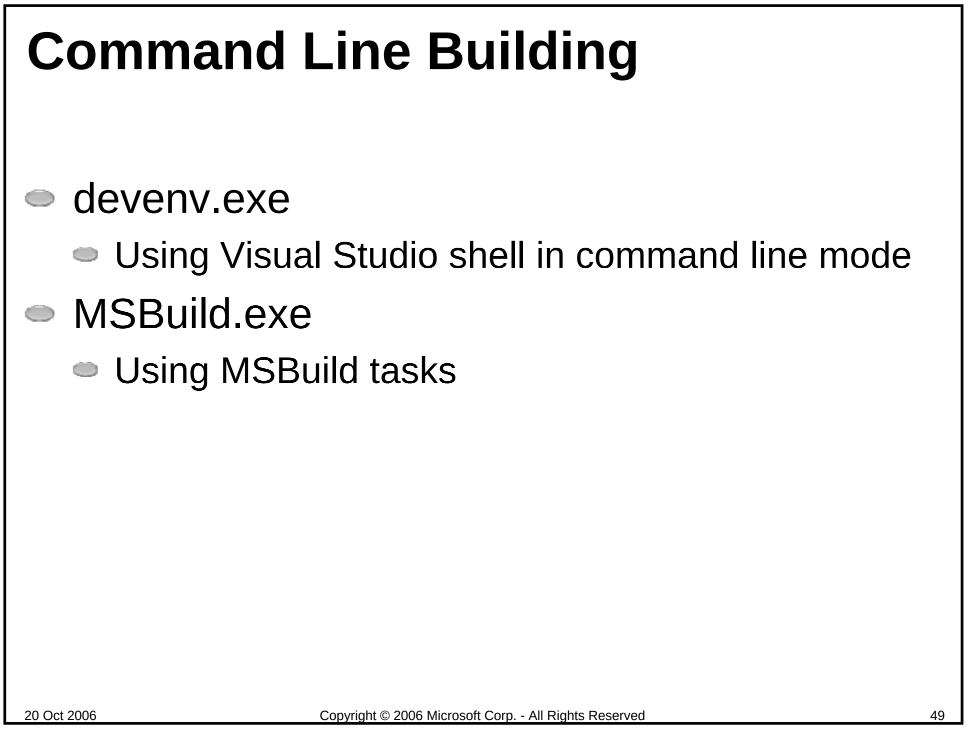
(*) Not implemented in the current CTP

20 Oct 2006 Copyright © 2006 Microsoft Corp. - All Rights Reserved 47

demo

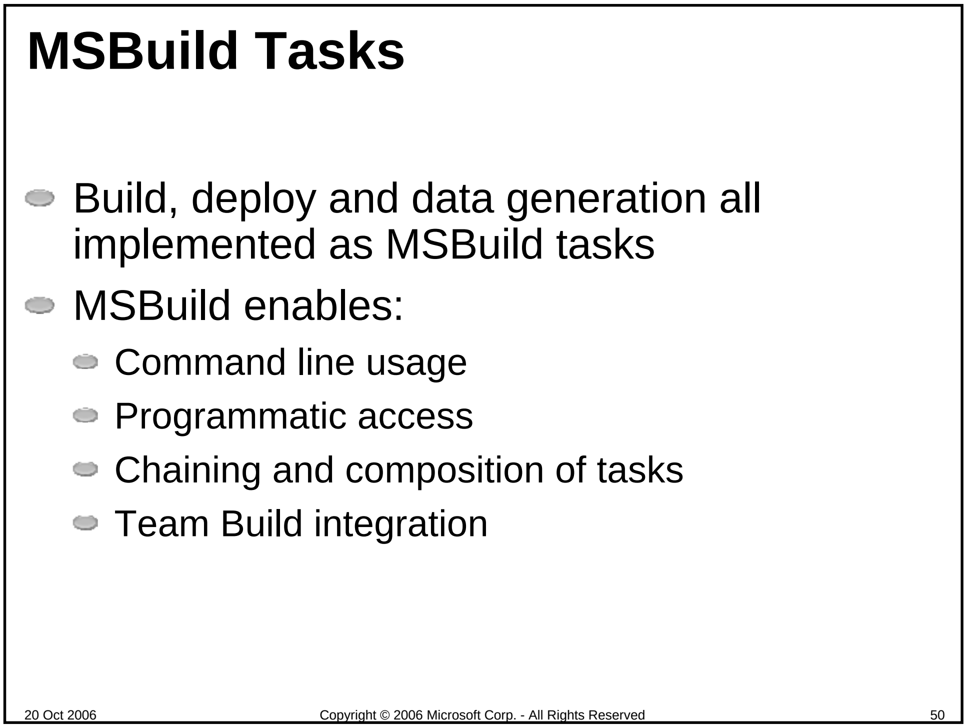
Building and Deploying the Project

This presentation is for informational purposes only. Microsoft makes no warranties, express or implied, in this summary.



Command Line Building

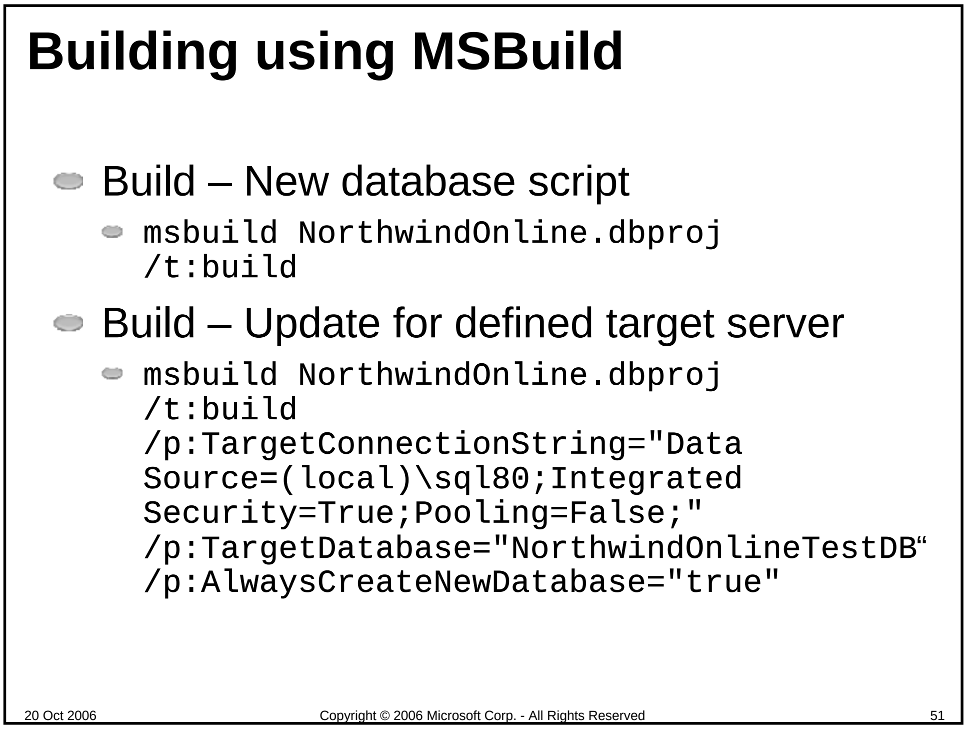
- devenv.exe
  - Using Visual Studio shell in command line mode
- MSBuild.exe
  - Using MSBuild tasks



MSBuild Tasks

- Build, deploy and data generation all implemented as MSBuild tasks
- MSBuild enables:
  - Command line usage
  - Programmatic access
  - Chaining and composition of tasks
  - Team Build integration

This presentation is for informational purposes only. Microsoft makes no warranties, express or implied, in this summary.

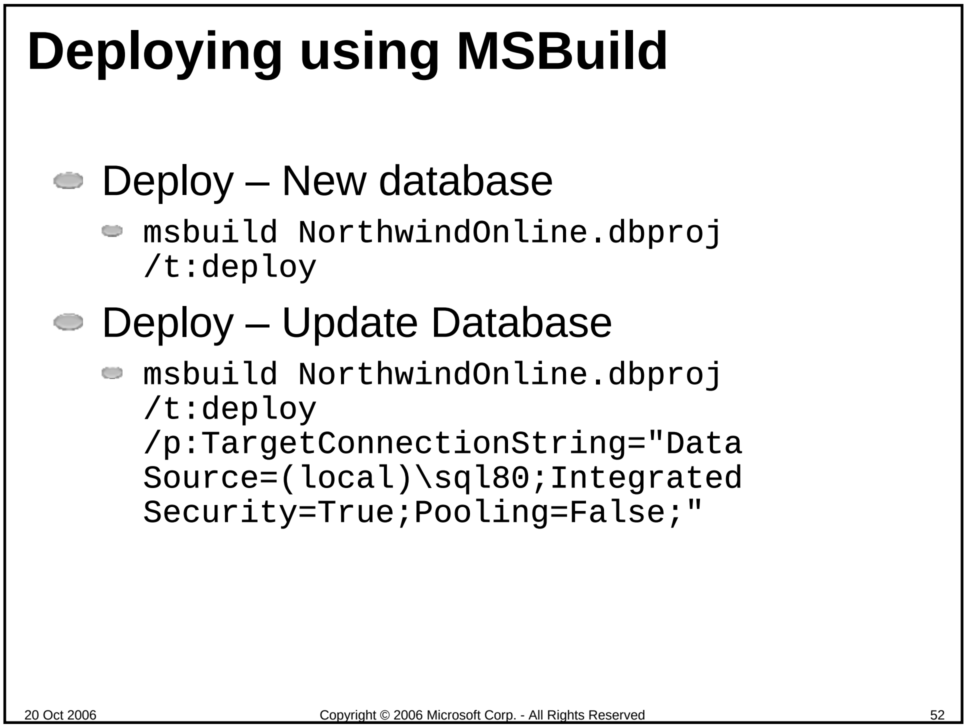


Building using MSBuild

- Build – New database script

  -
    ```
    msbuild NorthwindOnline.dbproj
    /t:build
    ```
- Build – Update for defined target server

  -
    ```
    msbuild NorthwindOnline.dbproj
    /t:build
    /p:TargetConnectionString="Data
    Source=(local)\sql80;Integrated
    Security=True;Pooling=False;"
    /p:TargetDatabase="NorthwindOnlineTestDB“
    /p:AlwaysCreateNewDatabase="true"
    ```



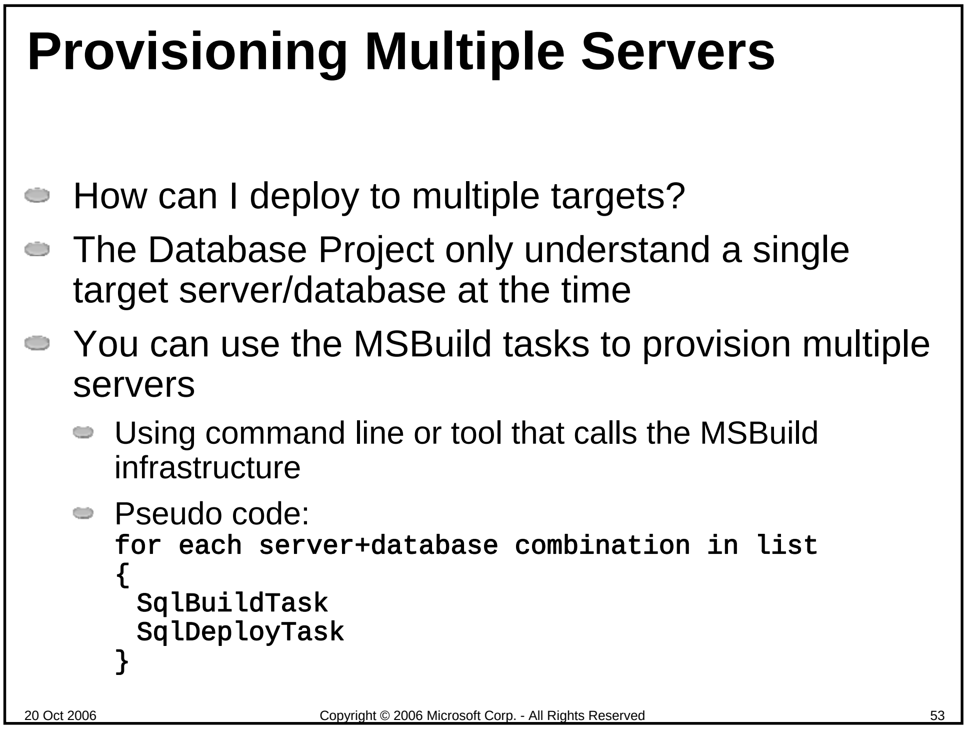
Deploying using MSBuild

- Deploy – New database

  -
    ```
    msbuild NorthwindOnline.dbproj
    /t:deploy
    ```
- Deploy – Update Database

  -
    ```
    msbuild NorthwindOnline.dbproj
    /t:deploy
    /p:TargetConnectionString="Data
    Source=(local)\sql80;Integrated
    Security=True;Pooling=False;"
    ```

This presentation is for informational purposes only. Microsoft makes no warranties, express or implied, in this summary.

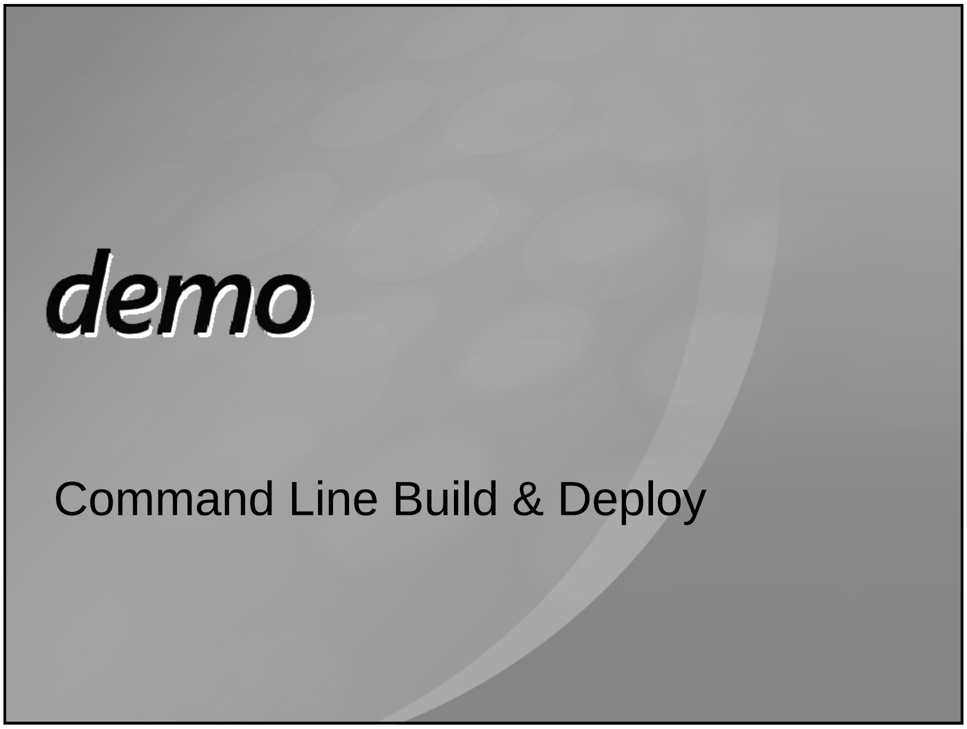


Provisioning Multiple Servers

- How can I deploy to multiple targets?
- The Database Project only understand a single target server/database at the time
- You can use the MSBuild tasks to provision multiple servers
  - Using command line or tool that calls the MSBuild infrastructure
  - Pseudo code:

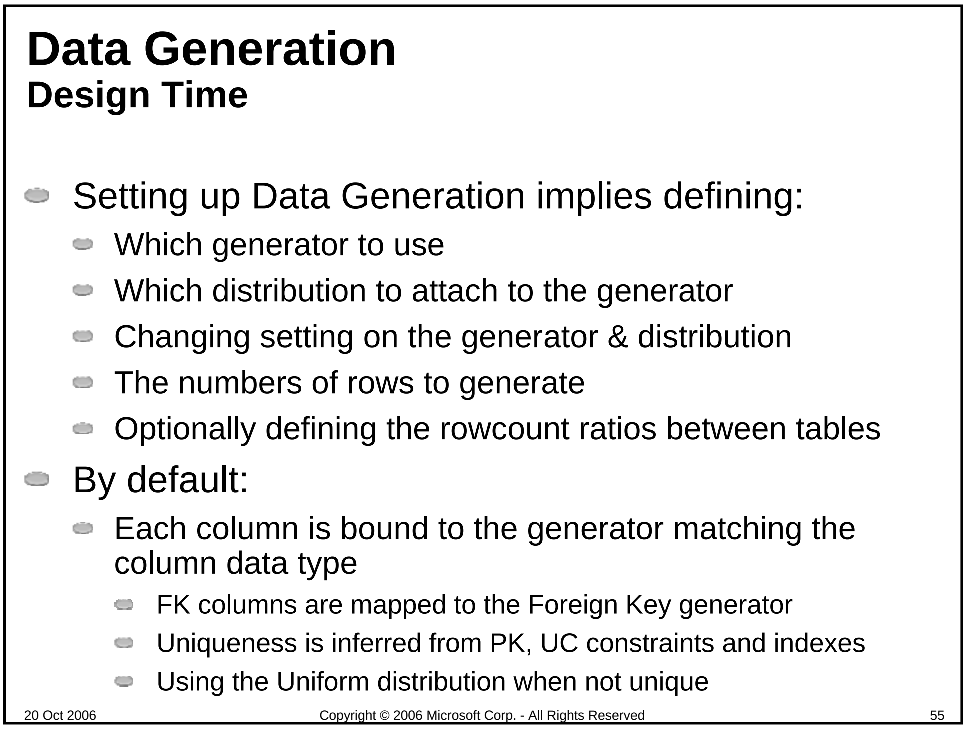
```
for each server+database combination in list
{
  SqlBuildTask
  SqlDeployTask
}
```

20 Oct 2006 Copyright © 2006 Microsoft Corp. - All Rights Reserved 53



demo

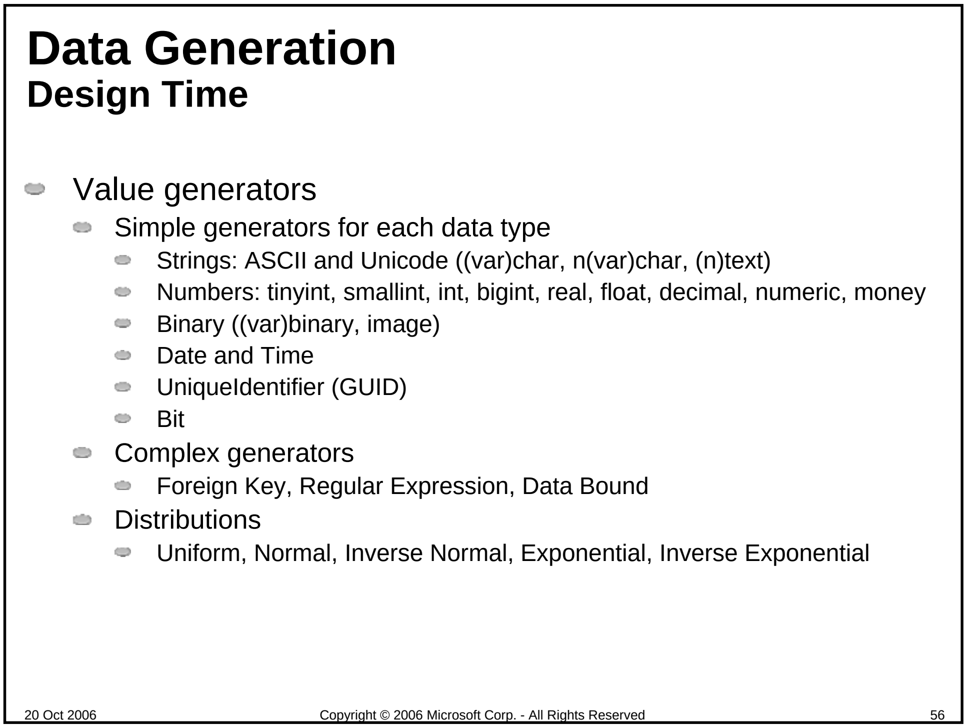
Command Line Build & Deploy



Data Generation
Design Time

- Setting up Data Generation implies defining:
  - Which generator to use
  - Which distribution to attach to the generator
  - Changing setting on the generator & distribution
  - The numbers of rows to generate
  - Optionally defining the rowcount ratios between tables
- By default:
  - Each column is bound to the generator matching the column data type
    - FK columns are mapped to the Foreign Key generator
    - Uniqueness is inferred from PK, UC constraints and indexes
    - Using the Uniform distribution when not unique

20 Oct 2006 Copyright © 2006 Microsoft Corp. - All Rights Reserved 55

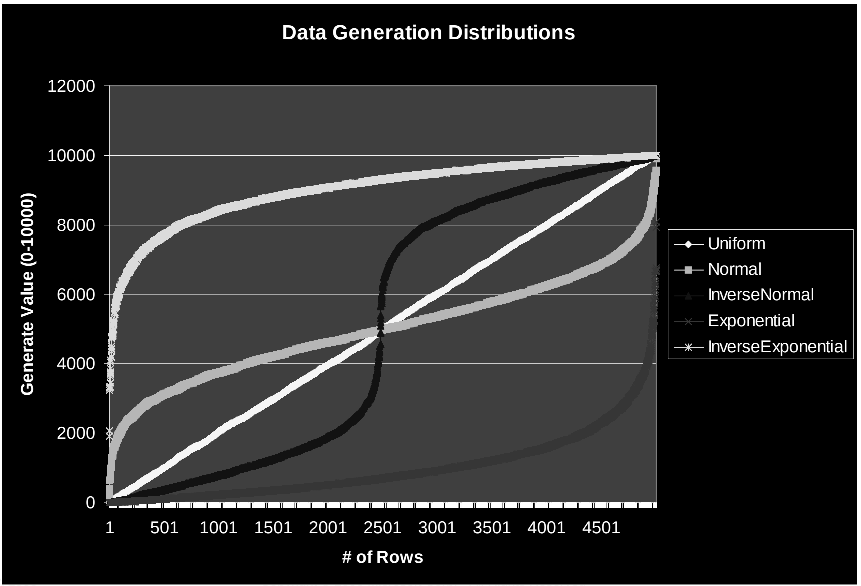


Data Generation
Design Time

- Value generators
  - Simple generators for each data type
    - Strings: ASCII and Unicode ((var)char, n(var)char, (n)text)
    - Numbers: tinyint, smallint, int, bigint, real, float, decimal, numeric, money
    - Binary ((var)binary, image)
    - Date and Time
    - UniqueIdentifier (GUID)
    - Bit
  - Complex generators
    - Foreign Key, Regular Expression, Data Bound
  - Distributions
    - Uniform, Normal, Inverse Normal, Exponential, Inverse Exponential

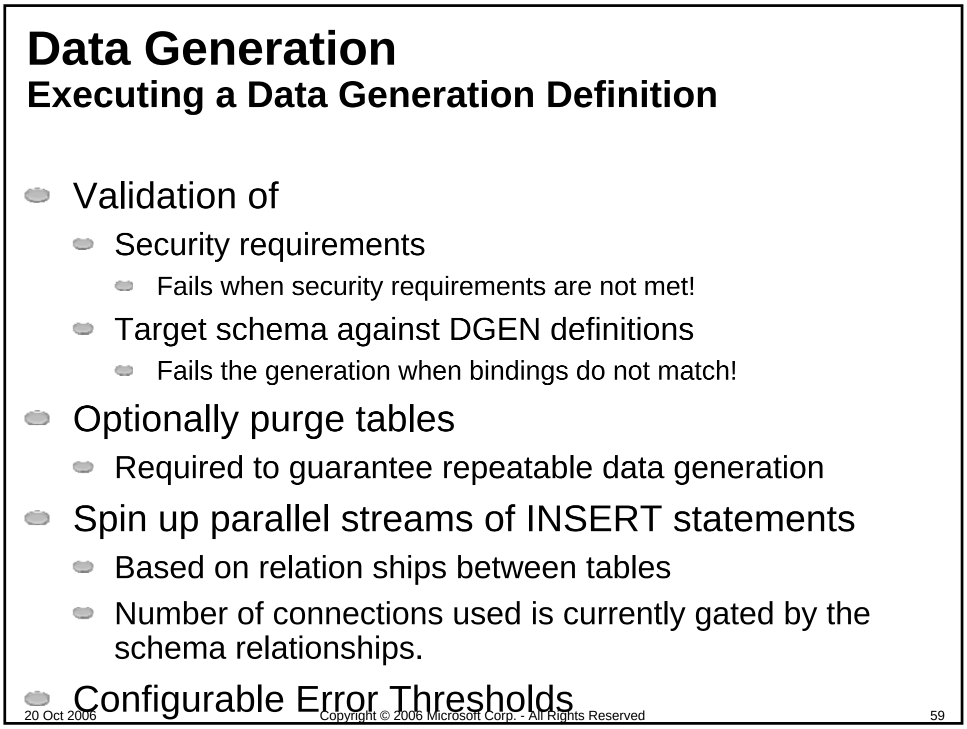
20 Oct 2006 Copyright © 2006 Microsoft Corp. - All Rights Reserved 56

Data Generation
Distributions @ Work



Data Generation
Design Time

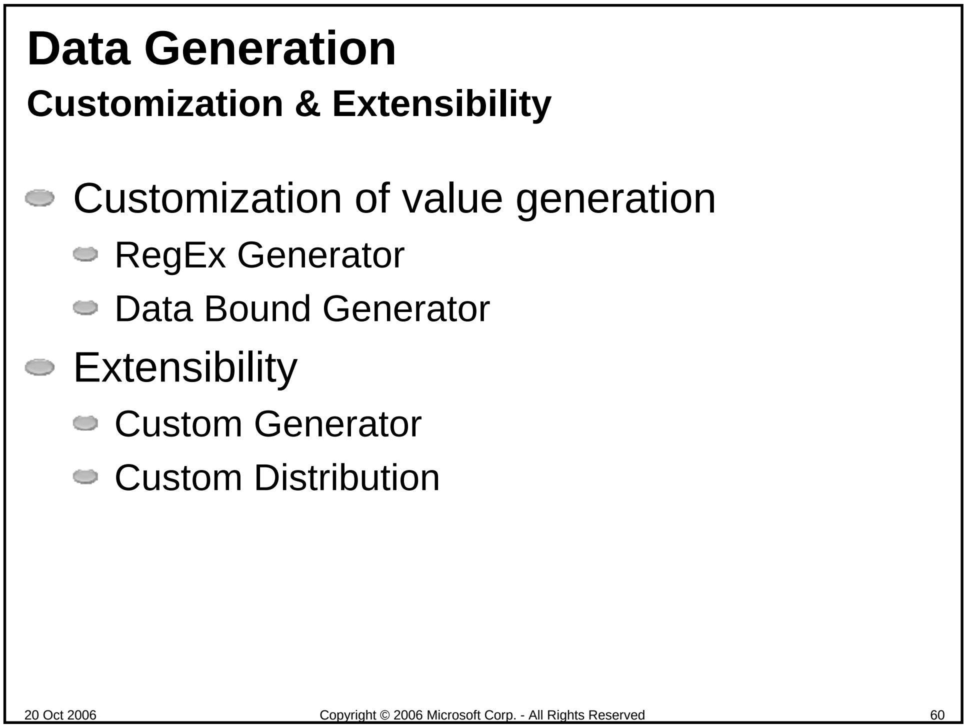
- Understand domain constraints
  - Check constraints (min/max)
- Table cardinality
  - Enforce table ratios
- Column value distribution



Data Generation

Executing a Data Generation Definition

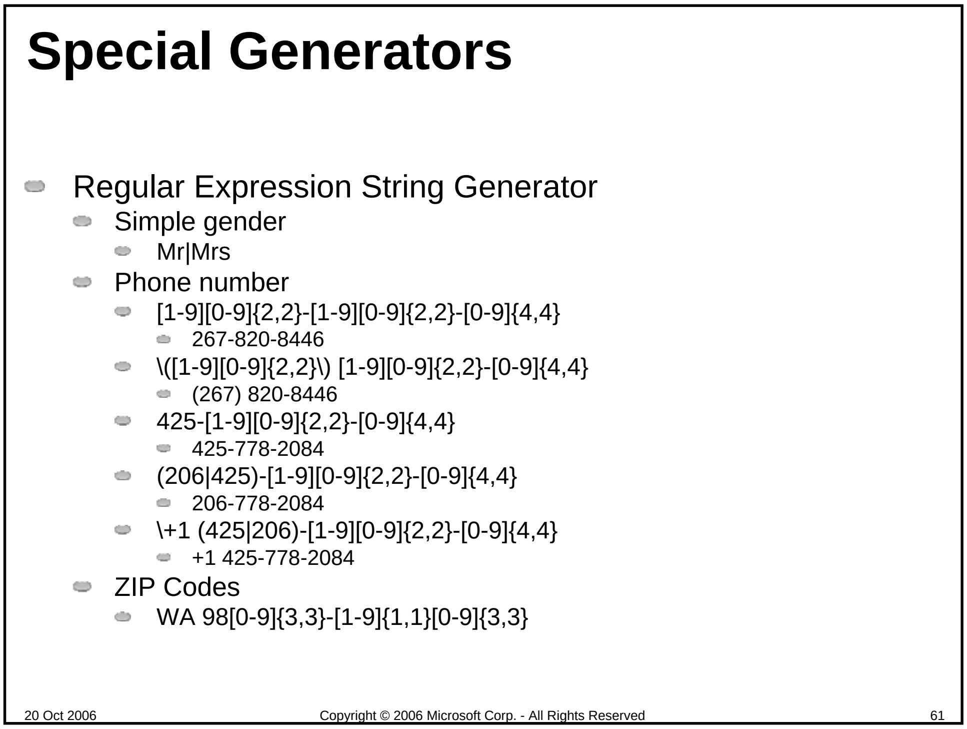
- Validation of
  - Security requirements
    - Fails when security requirements are not met!
  - Target schema against DGEN definitions
    - Fails the generation when bindings do not match!
- Optionally purge tables
  - Required to guarantee repeatable data generation
- Spin up parallel streams of INSERT statements
  - Based on relation ships between tables
  - Number of connections used is currently gated by the schema relationships.
- Configurable Error Thresholds



Data Generation

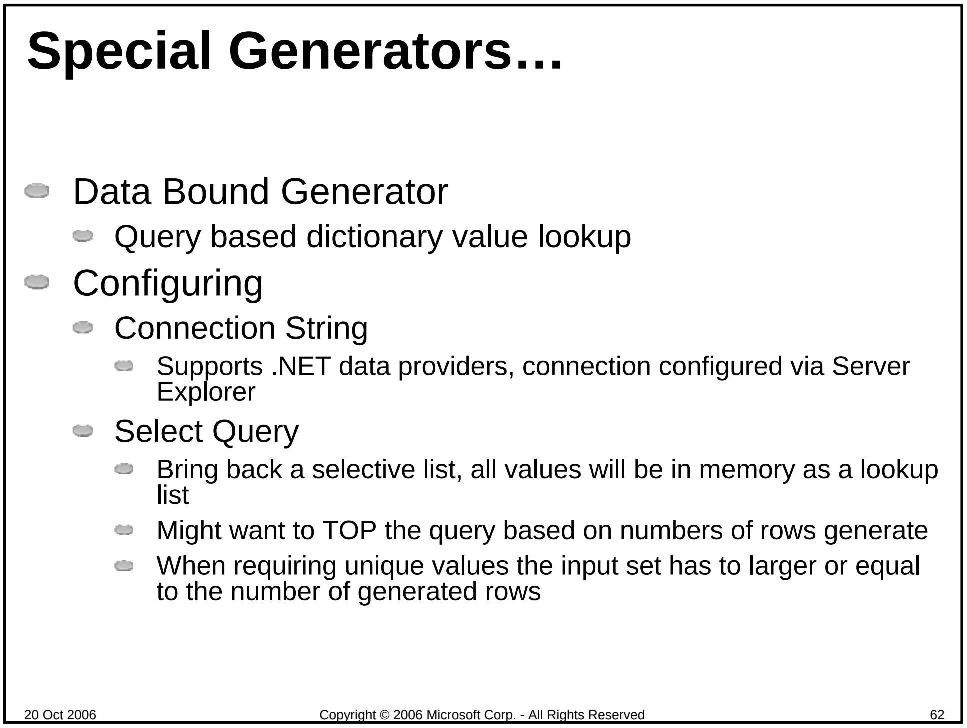
Customization & Extensibility

- Customization of value generation
  - RegEx Generator
  - Data Bound Generator
- Extensibility
  - Custom Generator
  - Custom Distribution



Special Generators

- Regular Expression String Generator
  - Simple gender
    - Mr|Mrs
  - Phone number
    - [1-9][0-9]{2,2}-[1-9][0-9]{2,2}-[0-9]{4,4}
      - 267-820-8446
    - \([1-9][0-9]{2,2}\) [1-9][0-9]{2,2}-[0-9]{4,4}
      - (267) 820-8446
    - 425-[1-9][0-9]{2,2}-[0-9]{4,4}
      - 425-778-2084
    - (206|425)-[1-9][0-9]{2,2}-[0-9]{4,4}
      - 206-778-2084
    - \+1 (425|206)-[1-9][0-9]{2,2}-[0-9]{4,4}
      - +1 425-778-2084
  - ZIP Codes
    - WA 98[0-9]{3,3}-[1-9]{1,1}[0-9]{3,3}

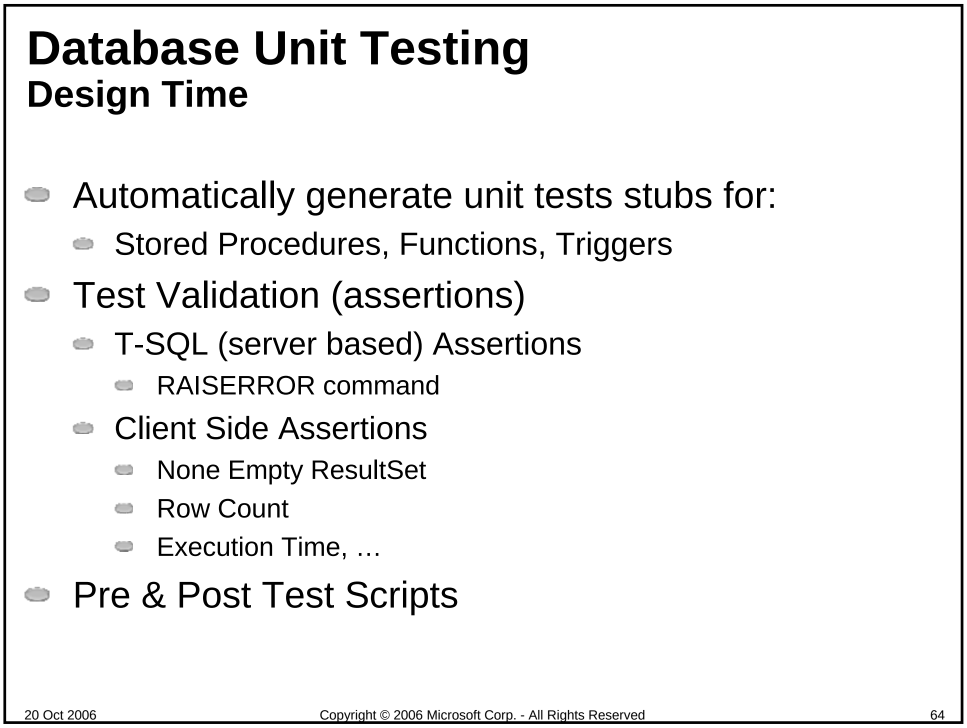


Special Generators…

- Data Bound Generator
  - Query based dictionary value lookup
- Configuring
  - Connection String
    - Supports .NET data providers, connection configured via Server Explorer
  - Select Query
    - Bring back a selective list, all values will be in memory as a lookup list
    - Might want to TOP the query based on numbers of rows generate
    - When requiring unique values the input set has to larger or equal to the number of generated rows

demo

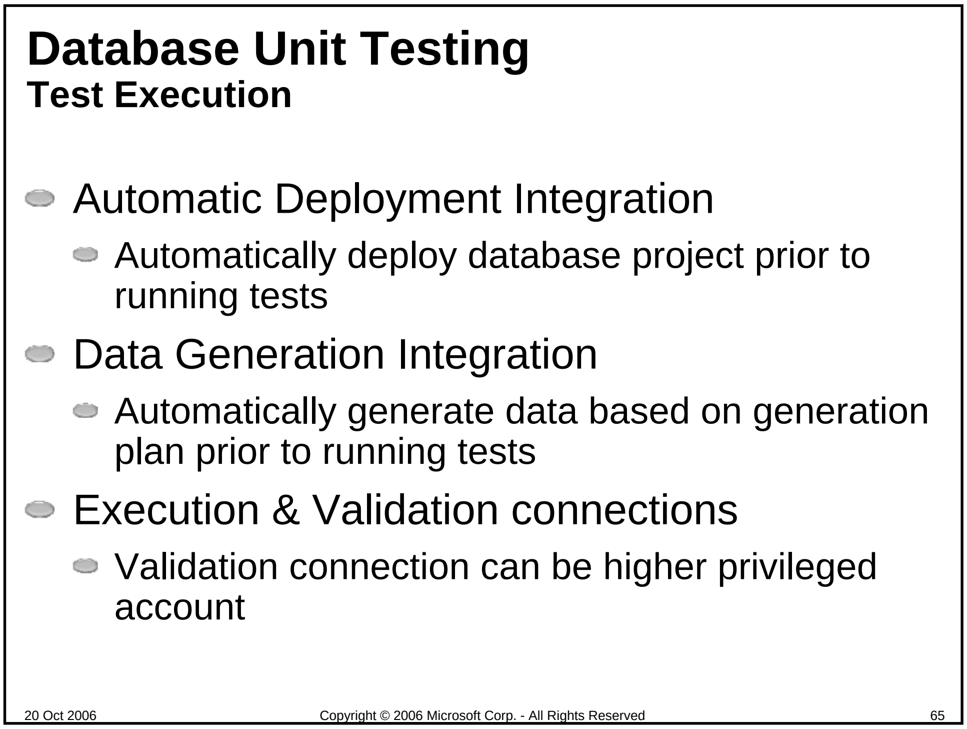
Data Generation



Database Unit Testing
Design Time

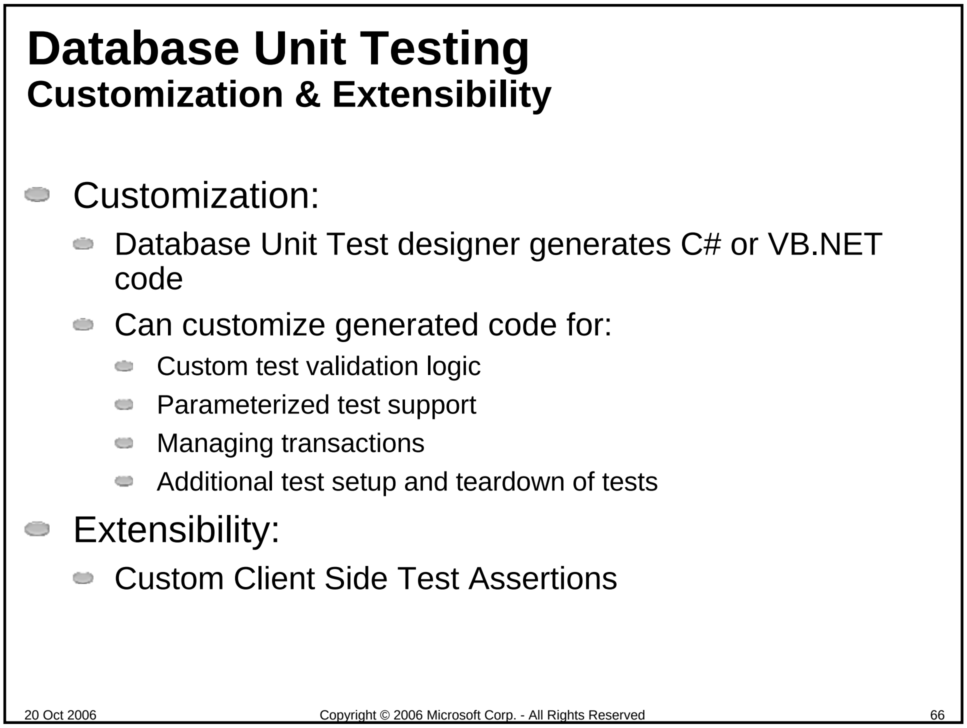
- Automatically generate unit tests stubs for:
  - Stored Procedures, Functions, Triggers
- Test Validation (assertions)
  - T-SQL (server based) Assertions
    - RAISERROR command
  - Client Side Assertions
    - None Empty ResultSet
    - Row Count
    - Execution Time, …
- Pre & Post Test Scripts

This presentation is for informational purposes only. Microsoft makes no warranties, express or implied, in this summary.



Database Unit Testing
Test Execution

- Automatic Deployment Integration
  - Automatically deploy database project prior to running tests
- Data Generation Integration
  - Automatically generate data based on generation plan prior to running tests
- Execution & Validation connections
  - Validation connection can be higher privileged account

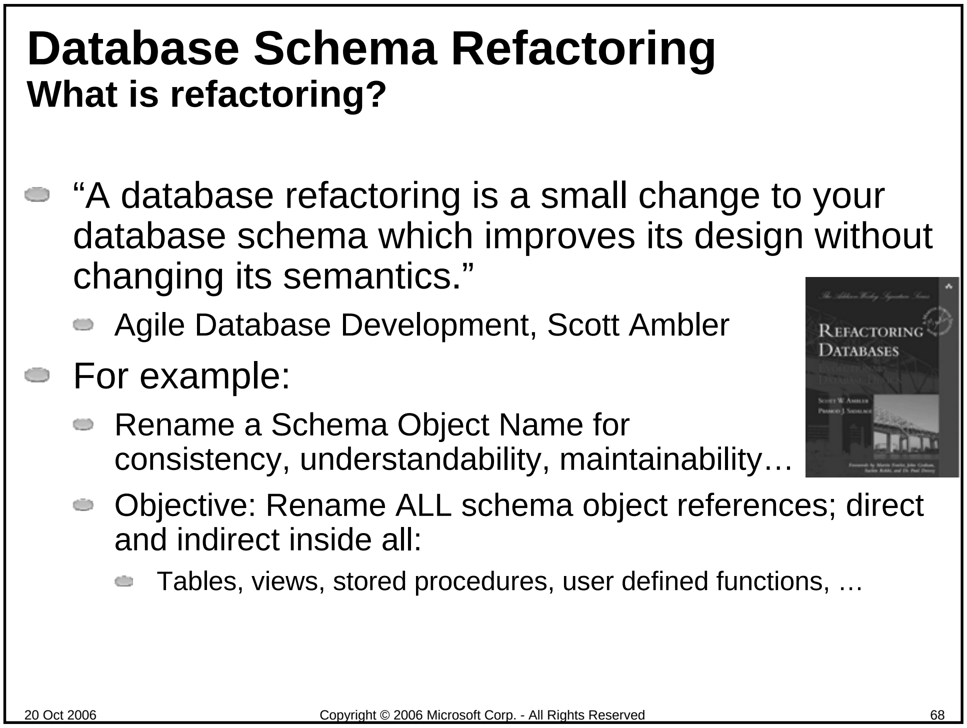


Database Unit Testing
Customization & Extensibility

- Customization:
  - Database Unit Test designer generates C# or VB.NET code
  - Can customize generated code for:
    - Custom test validation logic
    - Parameterized test support
    - Managing transactions
    - Additional test setup and teardown of tests
- Extensibility:
  - Custom Client Side Test Assertions

demo

Database Unit Testing

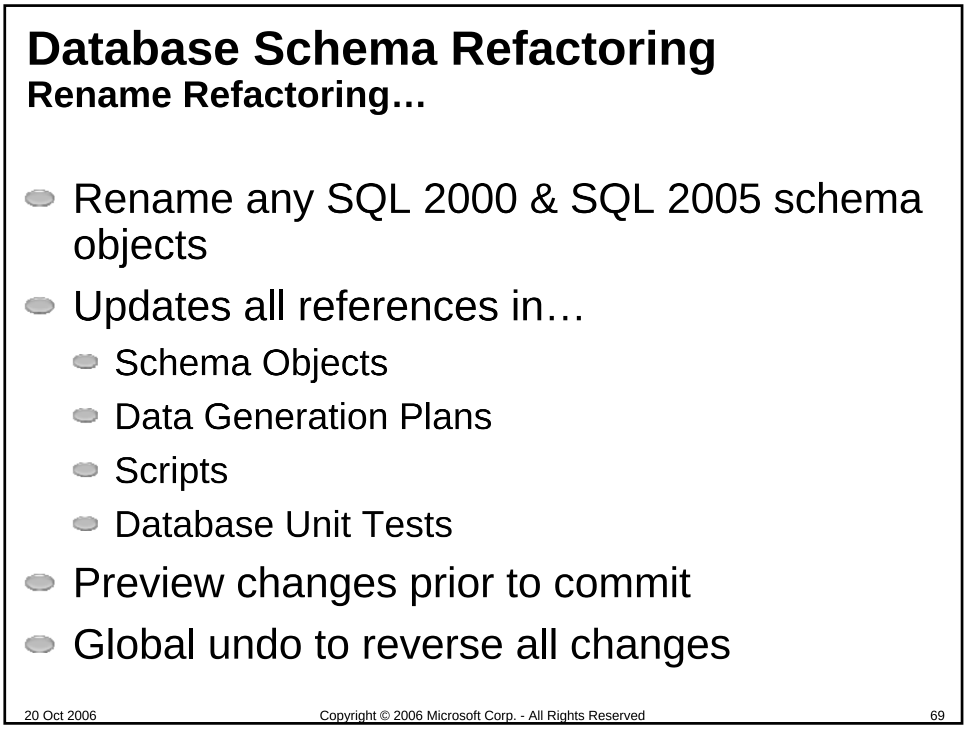


REFACTORING DATABASES

Database Schema Refactoring
What is refactoring?

- "A database refactoring is a small change to your database schema which improves its design without changing its semantics."
  - Agile Database Development, Scott Ambler
- For example:
  - Rename a Schema Object Name for consistency, understandability, maintainability…
  - Objective: Rename ALL schema object references; direct and indirect inside all:
    - Tables, views, stored procedures, user defined functions, …

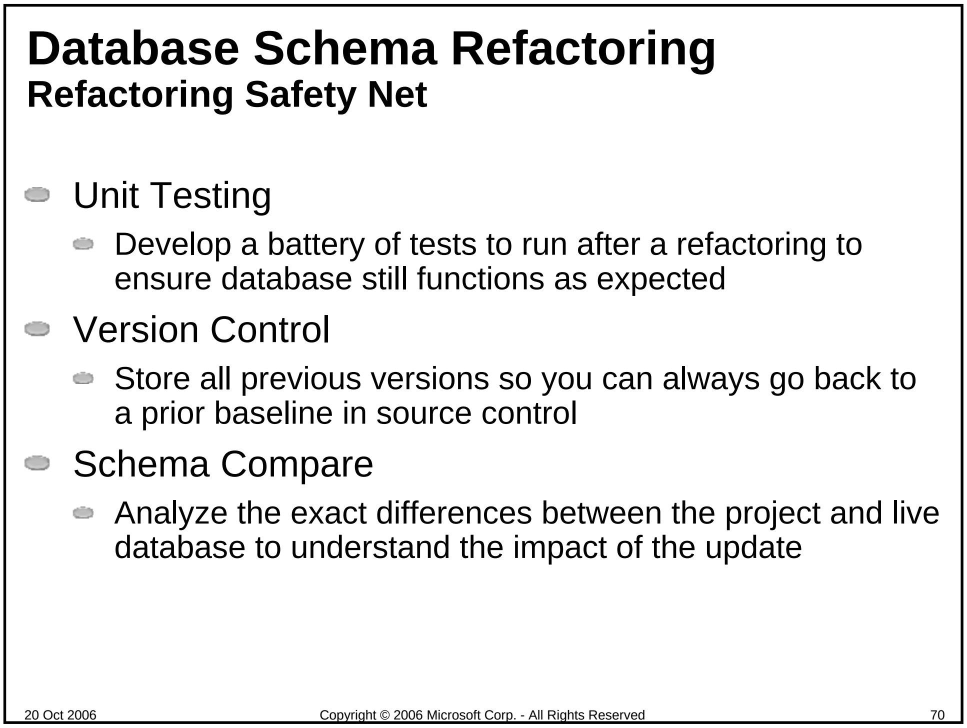
This presentation is for informational purposes only. Microsoft makes no warranties, express or implied, in this summary.



Database Schema Refactoring

Rename Refactoring…

- Rename any SQL 2000 & SQL 2005 schema objects
- Updates all references in…
  - Schema Objects
  - Data Generation Plans
  - Scripts
  - Database Unit Tests
- Preview changes prior to commit
- Global undo to reverse all changes



Database Schema Refactoring

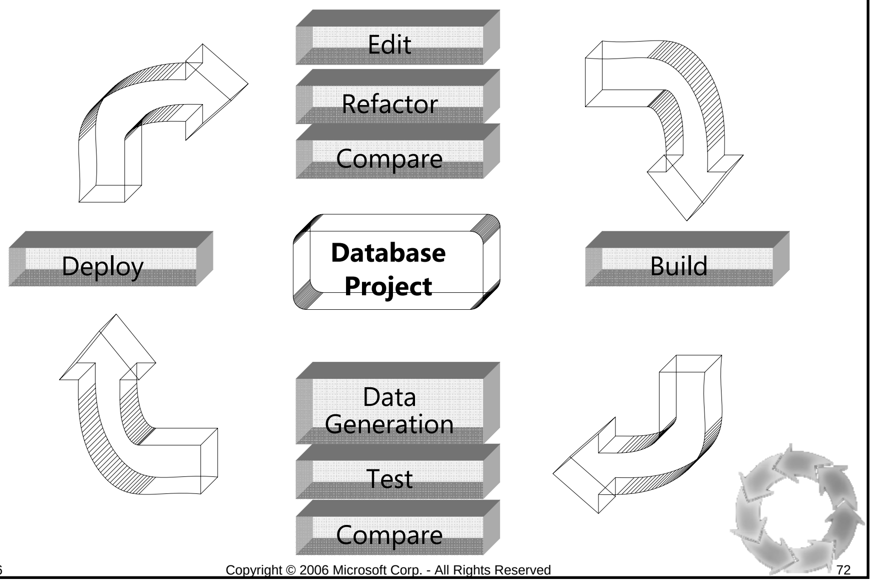
Refactoring Safety Net

- Unit Testing
  - Develop a battery of tests to run after a refactoring to ensure database still functions as expected
- Version Control
  - Store all previous versions so you can always go back to a prior baseline in source control
- Schema Compare
  - Analyze the exact differences between the project and live database to understand the impact of the update

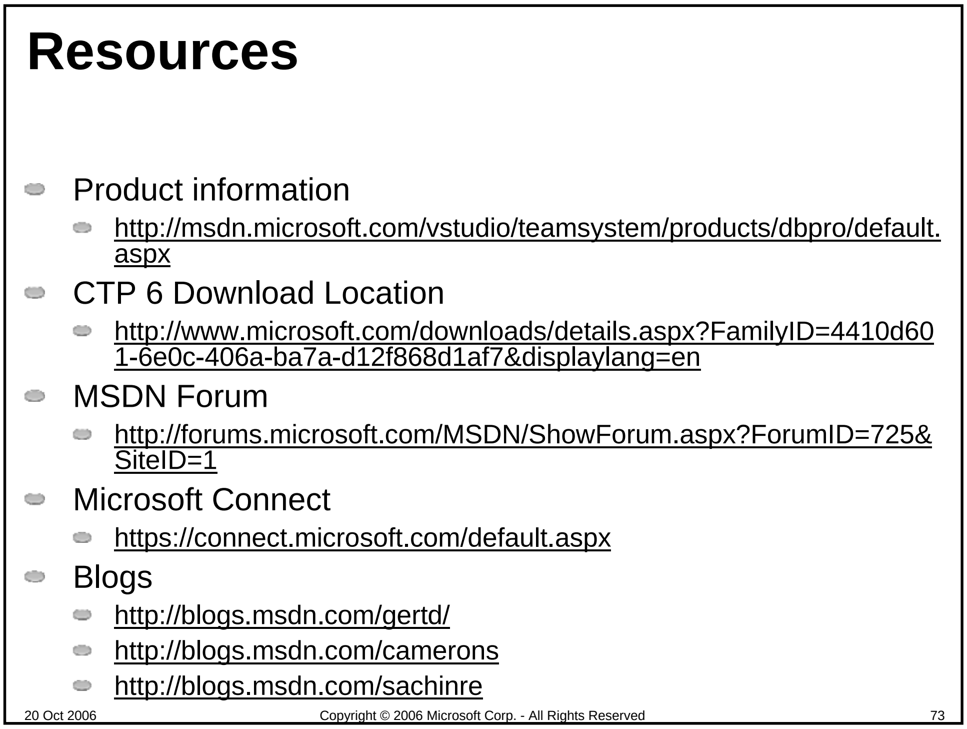
demo

Database Schema Refatoring

Integrating In To The Cycle of Life



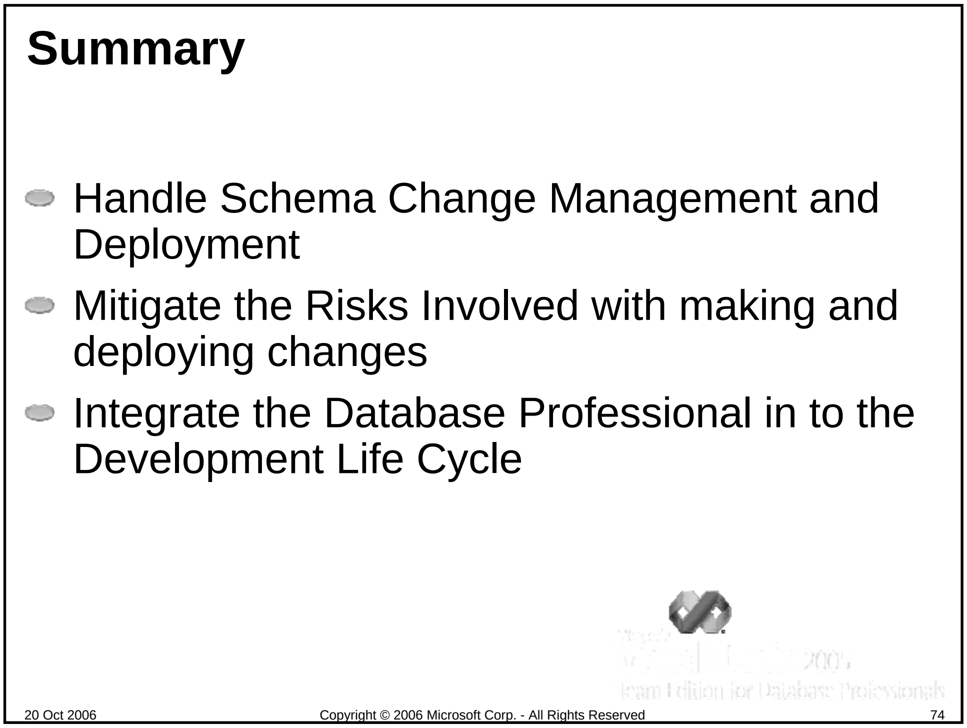
This presentation is for informational purposes only. Microsoft makes no warranties, express or implied, in this summary.



Resources

- Product information
  - http://msdn.microsoft.com/vstudio/teamsystem/products/dbpro/default.aspx
- CTP 6 Download Location
  - http://www.microsoft.com/downloads/details.aspx?FamilyID=4410d601-6e0c-406a-ba7a-d12f868d1af7&displaylang=en
- MSDN Forum
  - http://forums.microsoft.com/MSDN/ShowForum.aspx?ForumID=725&SiteID=1
- Microsoft Connect
  - https://connect.microsoft.com/default.aspx
- Blogs
  - http://blogs.msdn.com/gertd/
  - http://blogs.msdn.com/camerons
  - http://blogs.msdn.com/sachinre

20 Oct 2006 Copyright © 2006 Microsoft Corp. - All Rights Reserved 73



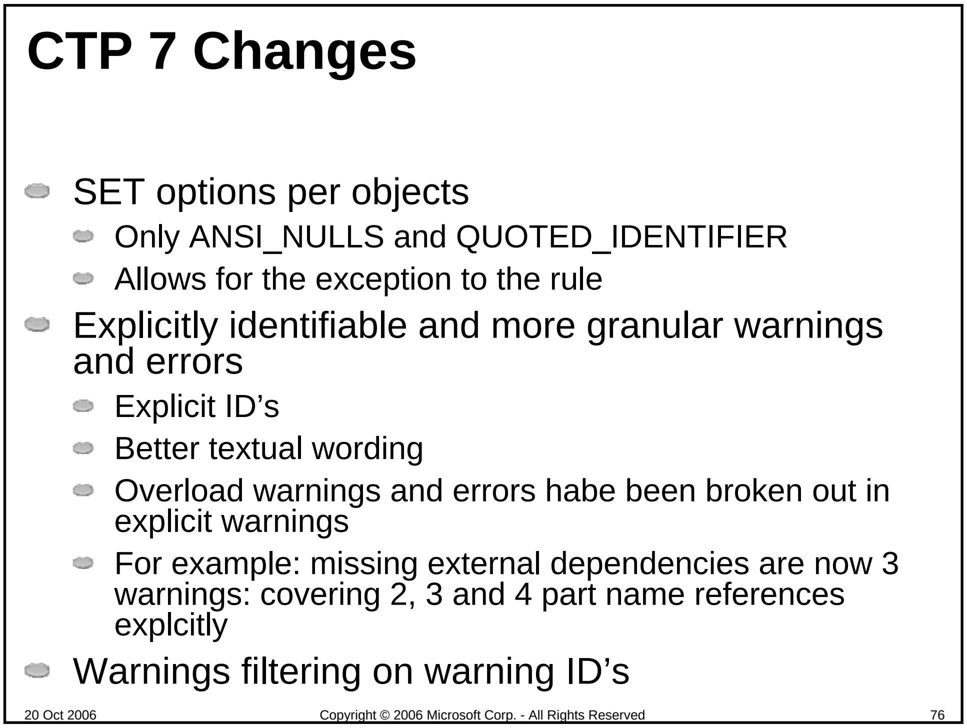
Summary

- Handle Schema Change Management and Deployment
- Mitigate the Risks Involved with making and deploying changes
- Integrate the Database Professional in to the Development Life Cycle

20 Oct 2006 Copyright © 2006 Microsoft Corp. - All Rights Reserved 74

This presentation is for informational purposes only. Microsoft makes no warranties, express or implied, in this summary.

Microsoft®



CTP 7 Changes

- SET options per objects
  - Only ANSI_NULLS and QUOTED_IDENTIFIER
  - Allows for the exception to the rule
- Explicitly identifiable and more granular warnings and errors
  - Explicit ID’s
  - Better textual wording
  - Overload warnings and errors habe been broken out in explicit warnings
  - For example: missing external dependencies are now 3 warnings: covering 2, 3 and 4 part name references explcitly
- Warnings filtering on warning ID’s

This presentation is for informational purposes only. Microsoft makes no warranties, express or implied, in this summary.