SQL Server Database Development Best Practices

Grant Fritchey, Red Gate Software
Grant.Fritchey@red-gate.com

Jeremy Kadlec, Edgewood Solutions
jeremyk@edgewoodsolutions.com
Agenda

• Introductions
• Best Practices
  – Design, Build, Test, Deploy, Tools
• Demos
• Q&A and Thank You
• Special offer for all attendees
Grant Fritchey

• Product Evangelist for Red Gate Software
• Microsoft SQL Server MVP
• Author:
  – SQL Server Execution Plans
  – SQL Server 2008 Query Performance Tuning Distilled
  – SQL Server MVP Deep Dives II
Jeremy Kadlec

• Edgewood Solutions - CTO
  • jeremyk@edgewoodsolutions.com
  • 410.591.4683

• MSSQLTips.com - Co-Founder
  • Twitter - @MSSQLTips
  • LinkedIn - http://www.linkedin.com/groups/MSSQLTips-2320891

• Baltimore SQL Server Users Group - Co-Leader
  • www.bssug.org

• SQL Server MVP
Design Best Practices

- Build a Strong Foundation
- Comprehensive Data Modeling
- Right Technology @ Right Time
#1 – Build a Strong Foundation

### Forward Thinking
- Business Goals
- Long Term Needs

### Scope & Support
- Managerial Support
- Reasonable Expectations

### Cohesive Team
- Correct Skills and Training
- Accountability

### Stable Platform
- Appropriate Technology
- Meet Performance Needs

### Complete Requirements
- Correct Level of Detail
- Agreement
#2 – Comprehensive Data Model

- Set the standard - Build a Data Modeling Checklist for your company
- Make the Data Dictionary mandatory
  - Store meta data in your SQL Server database
  - Include object, column, relationships, sample data, etc.
  - Maintain with your releases
- Include in code reviews and source control
#3 - Right Technology @ Right Time

- Holistic view
- Build technical plan
- Seek alternatives
- Mitigate risk with new technologies
  - Proof of concept
- Pull from experience
- Document and discuss with team
Build Best Practices

- Leverage Your Sandbox Environment
- Build on Your Experience
- Prioritize Security
#4 – Leverage Your Sandbox Environment

- Functional environment to match production
- Ability to setup quickly and begin development
- Intended for personal or small group usage
- Ability to build and recreate as needed
- Isolated so no impact to other team members
#5 – Build on Your Experience

- Naming conventions
- Comment code
- Parameters in tables
- Consolidated error handling
- Referential integrity, data types, indexes, etc.
- Select correct technology
- Review query plans
- Code reviews
- Check-in procedures
- No functions in WHERE clause
- Avoid cursors, temp tables, triggers, etc. when a better solution is available
- No views on top of views on top of views
- Don’t throw hardware at a database design problem
- Verify code before check-in
- No code susceptible to SQL Injection
#6 – Prioritize Security

- “From 2005 through to September 20, 2011, SQL injection has been responsible for 83% of successful hacking-related data breaches.”

- “…from 2005 to September 20, 2011. There were 312,437,487 data records lost due to hacking with about 262 million records from various breaches including TJMax, RockYou and Heartland, all of which were SQL injection attacks.”

- “In the wild, it has been noted that applications experience, on average, 71 attempts an hour.[1] When under direct attack, some applications occasionally came under aggressive attacks and at their peak, were attacked 800–1300 times per hour.”
Test Best Practices

- Coordinate Parallel Development
- Close the Loop
#7 - Coordinate Parallel Development

- Coordination requires communication, comments and notation
- Prevent “Code Jambalaya” from causing chaos and frustration
#8 - Close the Loop

- Review Requirements
- Build Test Cases
- Promote to Test
- Development
- Verify Requirements
- Track Changes
- User Involvement
Deployment Best Practices

Avoid Deployment Catastrophes
#9 – Avoid Deployment Catastrophes

- Don’t deploy suspect code, apply the brakes
- Don’t let any changes sneak into the environment
- Don’t assume things will just work out
- Don’t take someone’s word that code is accurate

✔ Build rollback scripts as your insurance policy
✔ Deploy the same scripts to Test as Production
  – Rollback deployment and re-deploy until error free
✔ Double check your deployment scripts for accuracy
  – Remember “Code Jambalaya”
✔ Ensure deployment is verified, do not find out from users
Tools Best Practices

- Build Re-Useable Templates
- Standardize Tools and Methods
#10 – Build Re-Useable Templates

- **T-SQL Templates**
  - Integrate with SSMS
  - Template projects
  - Data dictionary
- **Integration Services**
  - Import processes
  - Export processes
- **Reporting Services**
  - Connection objects
  - Headers, footers, etc.

- **Benefits**
  - No “blank slate” syndrome
  - Standardization
  - Set expectations
  - Time saver
#11 – Standardize Tools and Methods

- **Methods**
  - Data Access
  - Comments
  - Deployment
  - Rollback
  - Communication

- **SQL Server Tools**
  - Development
  - Continuous Integration
  - Source Control
  - Refactoring
Next Steps

Offer

For all webinar registrants

✓ SQL Source Control - $99
✓ SQL Developer Bundle - $200 discount

Email michael.francis@red-gate.com
Quote code “webinarmssqltipsmarch2012”
One offer per organization