Lehr- und Forschungseinheit Wirtschaftsinformatik 2
Informationsmanagement & Office Systeme
Fakultät für Wirtschaftswissenschaften
Prof. Dr. Ludwig Nastansky

Style Guide
zur Datenbankerstellung im Praktikum
Workgroup Computing 1

Stand: 1. Juli 2003

Inhaltsverzeichnis

1 Allgemein

Dieser Style Guide beschreibt Anforderungen, die bei der Erstellung der Abschlussdatenbanken im Praktikum Workgroup Computing 1 zu beachten sind. Die Einhaltung sämtlicher dieser Anforderungen ist notwendige Voraussetzung für das Bestehen des Kurses.
Darüber hinaus soll er eine Orientierungshilfe für die Programmierung darstellen, indem einerseits auf Fehler hingewiesen wird, die zu Notenabwertungen führen können, indem aber andererseits auch Hinweise gegeben werden, wie eine Notenaufwertung erreicht werden kann.

In den nächsten Kapiteln werden verschiedene Bereiche getrennt voneinander betrachtet. Die dort verwendeten Symbole haben folgende Bedeutung:

- Verbindliche Anforderungen. Nicht-Einhaltung führt automatisch zur Abwertung.
- Positive Eigenschaften, die zu einer Aufwertung führen können.
- Negative Eigenschaften, die zu einer Abwertung führen können.

Die unten aufgelisteten Hinweise sind zwangsweise sehr allgemein gehalten. Grundsätzlich muss jede Gruppe ihr Aufgabenszenario genauestens analysieren und darauf aufbauend die Abschlussdatenbank entwickeln. Daraus folgt unmittelbar, dass für jede Datenbank einzeln entschieden werden muss, welche Objekte zu betrachten sind, wie viele und welche Ansichten nötig sind, welche Funktionalitäten angeboten werden müssen, welche Access Control List (ACL) am geeignetsten ist und wie das Benutzerinterface auszusehen hat.

Die wichtigste Regel lautet:
Die fertige Datenbank muss geeignet sind, die ihr zugedachte Aufgabe im operativen Einsatz vollständig zu erfüllen.

2 Masken und Felder

- Ausreichende Anzahl verschiedenartiger Masken (mindestens 3). Betrachtung mehrerer Objekte, also z. B. Kunden, Lieferanten und Produkte.
- Ausreichende Anzahl Felder in jeder Maske. Insgesamt Nutzung möglichst vieler Feldtypen (z. B. berechnet, berechnet beim Speichern etc.) und Datentypen (z. B. Autor-, Datums-, Auswahl-Felder etc.).
- Help Descriptions für alle Felder, deren Benutzung nicht intuitiv verständlich ist (z. B. Angaben zu gewünschten Formatierungen und Einheiten, Auflistung möglicher Wertebereiche).
- Alle Input Validations und Input Translations, die die ordnungsgemäße Nutzung der Datenbank sicherstellen. Die Speicherung „leerer“ Dokumente muss z. B. verhindert werden, ebenso die Speicherung einer Rechnung ohne Angabe eines Produktes oder das doppelte Vorhandensein einer Kundennummer. Ebenso müssen vom Benutzer eingegebene Zahlen und Daten validiert werden, falls diese entscheidend sind.
- Fenstertitel für alle Masken.

- Verwendung von Subforms.
- Verwendung von Shared Fields.
- Verwendung von Layern.
- Vererbung zwischen Dokumenten.
- Dynamisch berechnete Fenstertitel.
- Zusätzliche Input Validations zur Sicherstellung sinnvoller Eingaben in allen Feldern (z. B. richtige Formate von Postleitzahlen, Telefonnummern, E-Mail-Adressen).
- Zusätzliche sinnvolle Input Translations.
- Vorgabe sinnvoller Standardwerte.

- Auswahl eines falschen Feldtyps (z. B. berechnet beim Speichern, obwohl die Daten bei jeder Modifizierung aktualisiert werden müssen) oder Datentyps (z. B. ein Textfeld, obwohl immer eine Zahl gespeichert wird).
- Verwendung einer Maske, obwohl die Inhalte völlig statisch sind.
- Vorhandensein von Testdaten, die mit älteren Versionen einer Maske erstellt wurden und mit der endgültigen Version nicht mehr angezeigt werden können.
- Speicherung von Dokumenten mittels Masken, für die keine Speicherung vorgesehen ist (z. B. Masken mit einem einzigen Auswahlfeld, das die Inhalte einer eingebetteten Ansicht steuert).

3 Ansichten

- Ausreichende Anzahl verschiedenartiger Ansichten. Zwar sind auch ähnliche Ansichten zu einem Objekt möglich, jedoch sollte für alle betrachteten Objekte zumindest eine eigene Ansicht existieren (z. B. für Produkte, Kunden und Lieferanten).
- Ausreichende Anzahl Spalten für jede Ansicht.
- Eine View Selection für jede Ansicht. Diese kann auch über den Ansichtserstellungsdialog erzeugt werden.

- Komplexere View Selections, die mithilfe von @Functions erzeugt wurden.
- Verketten von Spaltenwerten mithilfe von @Functions.
- Bereitstellung reichhaltiger Auswertungsmöglichkeiten mithilfe von Ansichten (z. B. Kategorisierung nach bestimmten Kriterien, Verknüpfung mehrerer Objekte, Darstellung bestimmter Teilmengen der Datensätze).

- Verwendung einer großen Anzahl von Ansichten für eine Objektart, obwohl eine einzige alle Aufgaben erfüllen könnte (z. B. Sortierung nach unterschiedlichen Spalten).
- Ungeschickte Wahl der Sortierung bzw. Kategorisierung.

4 Seiten

- Ausreichende Anzahl verschiedenartiger Seiten (mindestens 3).
- Fenstertitel für jede Seite.

5 Aktionen

- Ausreichende Anzahl sinnvoller Aktionen (mindestens 5; eingebettet in Masken, Ansichten oder Seiten).

- Bereitstellung aller vom Benutzer benötigten Aktionen an den entsprechenden Stellen.
- Verwendung von Shared Actions.

- Falsch implementierte Aktionen, die ihren Zweck nicht erfüllen.
- Bereitstellung von Aktionen in Kontexten, die zu Fehlern führen (z. B. eine Bearbeiten-Schaltfläche im Bearbeitungsmodus eines Dokumentes).

6 Agenten

- Verwendung mindestens eines sinnvollen Agenten.

- Falsch implementierte Agenten, die ihren Zweck nicht erfüllen.
- Verwendung von Private Agents (nur im Rahmen der Veranstaltung ist die Nutzung dieses Features verboten).

7 Zugriff und Sicherheit

- Folgende vier Gruppen müssen als Manager in die Access Control List (ACL) eingetragen werden:
  WI2LEHRER (Person Group)
  WI2MANAGER (Person Group)
  WI2MIT (Person Group)
  WI2SERVER (Server Group)
- Falls mit Rollen gearbeitet wird, müssen diese vier Gruppen alle Berechtigungen erhalten.
- Festlegung einer fiktiven Access Control List (ACL), die im operativen Einsatz der Datenbank zu empfehlen wäre (inklusive Default- und Anonymous-Zugriff).

- Verwendung von Reader und Author Fields.
- Einsatz von Rollen, um den Zugriff auf bestimmte Elemente zu kontrollieren.
- Verstecken von Elementen, um die Übersichtlichkeit zu steigern (z. B. mithilfe von Rollen oder in Abhängigkeit von Zuständen, beispielsweise Bearbeitungs-modus / Lesemodus, neues / altes Dokument).

- Aktivierung von "Enforce a consistent ACL across all replicas of this database" (nur im Rahmen der Veranstaltung ist die Nutzung dieses Features verboten)
- Mangelhaftes Zugriffsmanagement (z. B. Designer-Rechte für Default).
- Vorhandensein von Rollen, die an keiner Stelle verwendet werden.

8 Layout und Benutzer-Interface

- Eigenständiges Layout für alle Bereiche der Datenbank (inklusive eigenem Datenbank-Symbol).
- Durchdachte Benutzerführung, geeignetes Navigationskonzept.

- Ergonomische optische Präsentation (farbig, aber nicht bunt; genügend Platz lassen; Farben im Kontext verwenden, an Alltagskonventionen halten: Grün für „ja“, Rot für "nein"; starke Kontraste vermeiden usw.).
- Erstellung eigener Grafiken für Schaltflächen, Logos usw.
- Verwendung einer Gliederung.
- Verwendung eines Framesets.

- Mehrfaches Öffnen des Framesets (hervorgerufen durch ungeschickte Target-Anweisungen).

9 Sonstiges

- Eine Hilfe, die dem Benutzer der Datenbank deren Anwendung erklärt. Zumindest ein hilfreiches Using Document (inklusive Screenshots).
- Aussagekräftiges About Document, das Sinn und Zweck der Datenbank erläutert, das Programmiererteam mit Matrikelnummern vorstellt und Kontaktmöglichkeiten auflistet.
- Einführung generischer Namen (Alias) für alle Gruppen von Designelementen, die an anderer Stelle referenziert werden (für gewöhnlich Masken, Ansichten, Seiten, Agenten, Gliederungen, Framesets usw.).
- Genügend sinnvolle Beispieldaten. Es müssen sämtliche Funktionalitäten getestet werden können, ohne vorher neue Testdaten zu erzeugen.

- Ausführliche selbst implementierte Hilfe (z. B. durch einen View mit Hilfedokumenten oder ein Frameset mit einem Angebot an statischen Hilfeseiten)
- Verknüpfung und Nutzung bestehender Daten (z. B. durch Vererbung, eingebettete Ansichten, Verwendung von @DBColumn und @DBLookup). Als Beispiel: Statt bei jedem Auftrag die Kundendaten zu erfassen, könnte man auch eine Kundemaske erstellen und dann bei neuen Aufträgen lediglich Zeiger auf vorhandene Kunden speichern.
- Verwendung von Computed Text.

- Gelöschte Designelemente, die an anderer Stelle noch aufgerufen werden.
- Verknüpfungen, deren Ziel nicht mehr existiert.
- Unbrauchbare Benennung von Designelementen (z. B. Felder mit Namen „test1“ und „test2“ statt „Vorname“ und „Nachname“).
- Nicht vollständige / nicht sinnvolle / veraltete Beispieldaten.