F. Unterrichtsbezogenes Datenbankpraktikum

Die im unterrichtsbezogenen Datenbankpraktikum (UDBP) zu erbringende Leistung stellt den Bewertungsteil des zweiten Semesters der Vorlesung zu Datenbanksystemen dar. Er besteht in der schriftlichen Dokumentation der Praktikumsergebnisse incl. für die Datenbank verwendbarer Dateien. Das Praktikum wird in kleinen Gruppen durchgeführt (2-4 Personen), idealerweise zu dritt.

Aufgabenstellung:	27.03.2023
Praktikumsbeginn:	17.04.2023
Abgabe der Ausarbeitung:	**19.06.2023**
Präsentationen:	ab 26.06.2023

Eine Abgabe nach dem angegebenen Abgabetermin führt zur Bewertung *nicht bestanden*! Es muss zudem eine Zuordnung einzelner Teile der schriftlichen Ausarbeitung zu den Gruppenmitgliedern möglich sein. Bei fehlender Zuordnung wird davon ausgegangen, dass alle Mitglieder zu gleichen Teilen für alle Abschnitte der Ausarbeitung verantwortlich sind.

Damit die Orientierung in fremdem Material später für alle Gruppenteilnehmer schnell und einfach möglich ist, ist es wichtig, dass Sie sich beim Zusammenstellen der Dokumente an eine einheitliche Ordnerstruktur für Ihre Ausarbeitung halten (siehe Seite F–6). Sinnvoll ist die Erstellung eines Git-Archivs.
Wünschenswert wäre, die Ausarbeitungen aller Gruppen zum Zweck der allgemeinen Nutzung zur Verfügung zu stellen.

F.1. Einordnung

DBS-Miniwelten modellieren den Kern eines größeren realen Anwendungssystems, das wegen seiner Komplexität aus informatischer Sicht nicht im Unterricht anzuwenden ist[109]. Die Modellierung der Miniwelt kann als ein Beispiel und Leitline bei der Einführung und Erarbeitung von fachlichen Konzepten des Themenkreises „Datenbanksysteme“ im Unterricht dienen.

Die Erfahrung lehrt, dass Informatiklehrer viele schöne Projektideen für den Unterricht aus reinem Zeitmangel liegen lassen. Die Praktikumsaufgabe setzt hier an, indem sie die zeitfressenden Vorbereitungsaufgaben für die Durchführung eines Projekts erledigt. Die Umsetzung in den Unterricht wird durch die Herstellung einer guten und vollständigen Dokumentation erleichtert. Die Arbeitsergebnisse dieses Praktikums können und sollen als Grundlage für zukünftige Unterrichtsprojekte dienen. Eine konkrete Umsetzung in Unterrichtsplanung (schülergerecht aufbereitetes Material, Arbeitsblätter, Tests, . . .) gehört nicht mehr zur Aufgabenstellung.

[109] Was nicht heißt, dass einige Kollegen nicht auf den Gedanken kommen, das „große reale System“ aus Anwendersicht im Unterricht zu behandeln. Das ist möglich, kostet viel Zeit, kann erstaunliche Einsichten erzeugen, der Nutzen für die Realisierung tiefergehender informatischer Lernziele ist jedoch begrenzt.

Modellierung

Die Kunst bei der Erfindung einer geeigneten Miniwelt liegt unter anderem darin,

- Welten zu finden, die dem Erfahrungsschatz der Schülerinnen und Schüler nahestehen oder zumindest mit geringem Aufwand erfasst werden können,
- eine knappe und präzise textuelle Beschreibung der Miniwelt zu entwerfen,
- verständlich zu machen, welcher Detaillierungsgrad (nicht) erreicht werden soll,
- zu berücksichtigen, welche Teile der Miniwelt nur lose oder gar nicht spezifiziert werden,
- eine Modellierbarkeit im ER-Modell mit 5-7 Entitäten zu gewährleisten,
- eine kleine, aber 'artenreiche' Population der Miniwelt zu generieren
- und eine spezielle Sicht auf die Daten zu ermöglichen, die modular mit Go oder PHP realisierbar ist.

F.2. Teilaufgaben

(a) Präzise textuelle Beschreibung des zu modellierenden Weltausschnitts bzw. der Anforderungen an die Datenbank ohne Vorgriff auf Entwurfsentscheidungen des Designers
(b) Erstellung eines ER-Modells mit vollständiger Attributierung und Angabe von Schlüsseln und Komplexitäten
(c) Angabe von statischen und ggf. dynamischen Integritätsbedingungen, die nicht im ERM erfasst werden
(d) Transformation des ER-Modells in eine Menge von Relationen (incl. Schlüssel, Fremdschlüssel, abstrakte Wertebereiche für Attribute)
(e) Angabe der verwendeten Transformationsregeln
(f) Angabe von funktionalen Abhängigkeiten in den Relationen
(g) Konkrete Umsetzung des Relationenentwurfs in einen Datenentwurf für die Implementierung in SQL (`postgreSQL`),
    (i) Umsetzung des Relationenentwurfs in DDL und DML, die als Dateien in die Datenbank eingelesen werden können
    (ii) Begründung der Wahl besonderer Attributdatentypen (z. B. NUMERIC, BOOL oder konkrete VARCHAR-Längen)
    (iii) Umsetzung der Integritätsbedingungen, wo es möglich ist und nicht zu komplex wird
(h) Entwurf einer Beispielpopulation der Tabellen: realitätsnah, variationsreich, aber klein
(i) Konstruktion einer Aufgabensequenz von insgesamt ca. 10 einfachen bis zu schwierigen Anfragen für eine Umsetzung in
    (i) Relationenalgebra,
    (ii) TRC und DRC (optional),
    (iii) Datalog (optional)
    (iv) und SQL.

    Eine Reihe von Aufgabenstellungen werden nur in SQL sinnvoll sein (Aggregatfunktionen, Gruppierung etc.), so dass für die anderen Anfragesprachen eine kleinere Auswahl getroffen werden kann.
(j) Erstellung von Musterlösungen zu diesen Anfragen in DCL der genannten Sprachen
(k) ggf. auch Alternativlösungen
(l) Dokumentation der Anfageergebnisse

(m) Auswahl und Implementierung einer Sicht auf die erstellten Daten in Form einer interaktiven Anwendung (in Go oder webbasiert in PHP), dazu die

(i) Begründung und Motivation der Sicht
(ii) Beschreibung der interaktiven Anwendungsmöglichkeiten
(iii) Beschreibung optionaler Möglichkeiten der Erweiterung
(iv) Erstellung einer datenbankgestützten, interaktiven Anwendung in Go oder PHP auf der Basis der erstellten Daten

Es geht *nicht* darum, lediglich die SQL-Anfragen in Go auszuführen. Es geht darum, eine kleine, praktikable Anwendung für den Endbenutzer zu erstellen, der die Datenbank nutzen soll.

Anforderung an die Miniwelten

Die Konstruktion eigener Miniwelten nach den bereits genannten Kriterien ist ausdrücklich erwünscht, sollte aber bitte vor Beginn der Arbeiten mit der Projektleitung abgesprochen werden. Bei der verbalen Formulierung kann man sich hinsichtlich des Umfanges und der Detailgenauigkeit an den vielen Beispielen der Vorlesung leiten lassen. Es sei noch einmal darauf hingewiesen, dass die Miniwelt der Erfahrungswelt der Schülerinnen und Schüler möglichst nahe stehen sollte.

Da es hier oft zu Missverständnissen kommt: Die Beschreibung der Miniwelt ist kein Vorgriff auf das ER- oder Relationenmodell, sondern aus der Perspektive des Äuftraggebers"die Beschreibung seiner Situation (z. B. der Garten-AG für den Schulgarten) für die eine Datenbank erstellt werden soll – also *bevor* ein ER-Modell entworfen wird. Es werden dem Designer alle Informationen gegeben, die in der Datenbank festgehalten werden sollen. Sie können sich dabei auch überlegen, welche Nachfragen ein Designer an die Garten-AG stellen würde, um die Datenbank korrekt zu entwerfen, und diese Information in die Miniwelt einbauen. Hier ist aber noch nicht von SSchlüsselattributen", ternären Beziehungen, dem Begriff Ëntitätenö. ä. die Rede, wenn es diese im Schulgarten nicht gibt. Ebenso wird z. B. vermutlich nicht für jedes Ding eine ID-Nummer existieren, diese mglw. einzuführen ist Entscheidung des Designers. Beispiele für Miniweltbeschreibungen sind den zahlreichen Übungen der Vorlesung zu Datenbanksystemen zu entnehmen.

Anmerkungen zur Bearbeitung

Es geht nicht um ein möglichst detailliertes, realitätsnahes Modell, vielmehr soll von zu vielen Details und sehr seltenen Sonderfällen abgesehen werden, um ein möglichst übersichtliches und nachvollziehbares (aber natürlich wiederum nicht *zu* kleines) Modell zu erhalten. Einiger zeitlicher Aufwand geht in die Konstruktion der Aufgabensequenz (Anfragen, Lösungen, Ergebnisse). Angestrebt ist eine graduell ansteigende Schwierigkeit der Aufgaben mit breiter Vielfalt in den Lösungsmethoden. Schön wäre, wenn die meisten Anfragen jeweils einen neuen Aspekt gegenüber den vorherigen Anfragen demonstrierten. Besonders soll die Ausarbeitung möglichst viele Konzepte von SQL verdeutlichen. Hierfür sind Angaben von Alternativlösungen besonders sinnvoll.

Auch der programmiertechische Teil (GO oder PHP) des UDBP soll explizit nicht jedes mögliche in der Realität vorkommende Detail abbilden bzw. fordern. Es können allerdings

zusätzliche Erweiterungsmöglichkeiten angeregt werden. Dieser Teil liefert für den Unterricht die Möglichkeit, weitere binnendifferenzierte Aufgaben zu stellen. Wichtig ist bei der Konzeption eine möglichst modulare Struktur der Anwendungsschicht zu erreichen, so dass Schülerinnen und Schüler in jedem Fall zu einem fertigen Produkt kommen können. Wichtig im Web-Zusammenhang ist eine saubere Trennung von Design und Struktur (wenn Sie darin Erfahrungen haben, sollten CSS-Dateien verwendet werden) der datenbankunabhängige Zugriff mittels PDO sowie eine übersichtliche und gut dokumentierte Implementierung.

F.3. Schriftliche Ausarbeitung

Die Ausarbeitung des Berichts sollte folgende Punkte umfassen:[110]

Modellierung

(a) Knappe, präzise textuelle Beschreibung der Miniwelt in einfachen Aussagesätzen. Dokument: `<db>-Welt.txt`, eine halbe bis eine Seite Text.

(b) Das Entity-Relationship-Modell in klassischen Notation unter Angabe aller Attribute, der Schlüssel und der Beziehungskomplexitäten. Achten Sie darauf, dass unter den Attributen mindestens ein numerisches ist, so dass Aggregierungsanfragen möglich werden.
Dokument: Computererstellte Grafik oder übersichtliche Handzeichnung in Form einer Datei `<db>-ERModell.<xxx>` (mit `xxx = png, pdf, eps,` o. ä.).

(c) Verbale Angabe von semantischen Bezügen bzw. Integritätsbedingungen, die sich im Modell nicht wiederfinden. (`<db>-SemBed.txt`).

(d) Ein Relationenmodell dazu in dritter Normalform mit (informeller) Angabe von Schlüsseln, Fremdschlüsseln und umsetzbaren Integritätsbedingungen in einer Datei `<db>-RelMod.txt`.

(e) Verbale Kurzbeschreibung des Transformationswegs vom ER-Modell der Datenbank zum Relationenmodell unter Angabe der benutzten Transformationsregeln

(f) Eine Angabe der im Modell vorkommenden funktionalen Abhängigkeiten in einer Datei `<db>-FAen.txt`. Bei fraglichen Abhängigkeiten geben Sie einen kurzen Hinweis z. B. Pflanzen-ID->Pflanzenname, aber Pflanzenname->Pflanzen-ID? (`<db>-Trafo.txt`).

(g) Eine kleine, gut überlegte Sammlung von weltnahen Beispieldaten zu allen Relationen in der Datenbank `<db>`. Die verwendeten Namen sollen plausibel, realitätsnah und paradigmatisch für eine größere Datenmenge sein. Und sie sollte mit einigen Sonderfällen auch komplexere Anfragen ermöglichen. *Format*: `<db>-Daten.txt` und optional auch `<db>-Daten-<rel>.csv` in Tabellenform. Für das csv-Format wird jede Relation jeweils in einer eigenen Datei abgelegt

(h) Zur Konstruktion der Anfragen: Anordnung nach steigendem Schwierigkeitsgrad. Jede Anfrage soll möglichst eine neue Komplexitätsstufe, Methode oder Implementierungstechnik demonstrieren. Insbesondere sollen auch Anfragen mit Negation, Unteranfragen, Aggregatfunktionen etc. vertreten sein.
Mit anderen Worten: der Konstruktionsprozess der Anfragen orientiert sich weniger an der Semantik des Miniweltmodells, sondern vielmehr an der Demonstration der jeweiligen Abfragemittel.
Der Entwicklungs- und Ausleseprozess der Anfragen wird vermutlich eng mit den Implementierungen gekoppelt sein

[110] Alle Textdateien dabei bitte in UTF-8-Kodierung.

Dokumentationsformat: `<db>-Anfragen.txt` als eine zusammenfassende Textdatei mit allen Anfragen in fortlaufender Nummerierung.

Relationenalgebra mit DES

Im Einzelnen sind gesucht:

(a) Das Erstellen und Befüllen der Relationen für DES in einer DDL/DML-Datei in simplem SQL (s. F.5).
(b) Die Implementierung der Anfragen in einer kommentierten ra-Datei `<db>-query.ra`. Nicht alle Anfragen werden im SQL-fernen Kontext sinnvoll sein (solche mit Aggregatfunktionen z. B.), sie können weggelassen werden. Die Aufgabenstellung steht jeweils als Kommentar über der auswertbaren Anfrage.
(c) Das Protokoll aller Anfrageergebnisse in einer Datei `<db>-query.ans`. Das geht z. B. mit einer einfachen Funktion, die alle Anfragen sequentiell auswertet und dabei die Ausgabe mit `&>` in eine Datei umlenkt.

Implementierung im Datenbankystem (PostgreSQL)

Im Einzelnen sind gesucht:

(a) *Der DDL-Teil*: eine dokumentierte Datei `Create-<db>.sql` zur Anlage der Tabellenstruktur einer Datenbank (oder eines Schemas) `<db>` in PostgreSQL.
(b) Eine dokumentierte Datei `Drop-<db>.sql` zur Beseitigung der Datenbank `<db>`.
(c) *Die Daten*: zwei Dateien `Insert-<db>-data.sql` und `Delete-<db>-data.sql`, die die Probedaten mit generischen SQL-Befehlen (`INSERT, DELETE`) in `<db>` einfügen bzw. aus ihr löschen.
(d) *Die Anfragen*: Die Implementierung aller Anfragen (numeriert) als `<db>-query.sql`. Die Aufgabe steht jeweils als Kommentar über der ausführbaren Anfrage. Oft sind Alternativ-Implementierungen möglich, deren Varianten in dieser Datei ebenfalls angegeben werden können, z. B. mit a- und b-Varianten. Ein begleitender didaktisch-methodischer Kommentar zu den Anfragen und den Lösungsvarianten ist in der zusammenfassenden Ausarbeitung erwünscht, z. B. wenn ein neuer Befehl verwendet wird.
(e) Die Protokollierung der Anfrageergebnisse in den Dateien `<db>-query<nn>.ans`, z. B. mittels `psql -o` oder mittels des Schalters `o <datei>`.

Didaktisch-Methodisches

Würdigen Sie unter den Aspekten Schwierigkeitsgrad, Umfang, Zeitbedarf und Bedeutsamkeit im Kontext des Informatikunterrichts ihre bis hierher geleistete Arbeit. Welche Teile sind unverzichtbar, welche ggf. fakultativ für ein Schülerprojekt? Haben Sie Ideen zur methodischen Umsetzung? Was wären in einer Weiterführung Ihres Projektes sinnvolle Fragestellungen? Die Antworten gehen in eine Datei `<db>-did.txt`

Ausarbeitung

Erwünscht ist eine zusammenfassende Darstellung Ihrer Praktikumsausarbeitung als PDF-Dokument. Die textliche Ausarbeitung steht in einer Druckfassung `<db>.pdf` und ggf. weiteren Formaten wie `<db>.txt` (formatiertes ASCII, UTF8), LaTeX-Format `<db>.tex`, `<db>.odt` oder `<db>.docx`. In jedem Fall wird eine kurze Beschreibung des Projektes zur Orientierung eines unbeteiligten Lesers geliefert (z. B. "Dies ist ein Datenbankprojekt der Lehrerweiterbildung Informatik, das ..."). Es handelt sich um eine kurze Einleitung im Format `README.txt`.

Alle Texte, Aufgaben und Lösungen sollen in einer *logischen Struktur* bereitgestellt werden (s. u.), die es leicht macht, für Unterrichtszwecke geeignete Teile zu übernehmen. Stellen Sie alle Praktikumsergebnisse in einer Archiv-Datei `<db>.tgz` mit folgender Binnenstruktur zusammen:

```
<db>
|-- doc                     (A) Bündelung aller Ausarbeitungen in einer Datei
|   |-- README.txt              ASCII-Text mit Kurzbeschreibung des DB-Projektes
|   |-- <db>.tex                optional :-)
|   |-- <db>.odt                optional
|   |-- <db>.doc[x]             optional
|   `-- <db>.pdf                in jedem Fall
|
|-- modell                  (B) Modelldokumente
|   |-- <db>-Welt.txt           Miniwelt
|   |-- <db>-ERModell.png       (oder .pdf, .eps, .jpg, .svg)
|   |-- <db>-SemBed.txt         zusätzliche sematische bzw. Integrationsbedingungen
|   |-- <db>-FAen.txt           vorhandene Funktionale Abhängigkeiten
|   |-- <db>-RelMod.txt         Relationenmodell
|   |-- <db>-Trafo.txt          angewandte Transformationsregeln
|   |-- <db>-Daten.txt          übersichtliche Tabelle der Beispielpopulation (kein SQL)
|   `-- <db>-Daten-<rel>.csv    Daten der jeweiligen Relation (ggf.)
|
|-- relational              (C) RALG/TRC/DRC/DL (in DES-kompatiblem Format)
|   |-- <db>-DDL.ddl            Erstellen und befüllen der Relationen in simplem SQL
|   |-- <db>-query.pdf          Alle RALG-Anfragen in originärer Syntax (Pi, Sigma usw.)
|   |-- <db>-query.ra           Alle RALG-Anfragen nummeriert mit Kommentar
|   |-- <db>-query.trc          Alle Tupelkalkül-Anfragen mit Kommentar
|   |-- <db>-query.drc          Alle Domänenkalkül-Anfragen mit Kommentar
|   |-- <db>-query.dl           Alle Datalog-Anfragen mit Kommentar
|   |-- <db>-query-ra.ans       Alle RALG-Anfrageergebnisse
|   |-- <db>-query-trc.ans      Alle Tupelkalkül-Anfrageergebnisse
|   |-- <db>-query-drc.ans      Alle Domänenkalkül-Anfrageergebnisse
|   `-- <db>-query-dl.ans       Alle Datalog-Anfrageergebnisse
|
|-- sql                     (D) SQL (PostgreSQL)
|   |-- Create-<db>.sql         Erstellen der Relationen
|   |-- Drop-<db>.sql           Löschen der Relationen
|   |-- Insert-<db>-data.sql    Befüllen der Relationen mit der Beispielpopulation
|   |-- Delete-<db>-data.sql    Löschen aller Inhalte der Relationen
|   |-- <db>-query.sql          Alle Anfragen nummeriert mit Kommentar
|   |-- <db>-query.ans          Anfrageergebnisse
|
|-- did                     (E) Didaktisch-Methodisches
|   |-- <db>-did.txt            Didaktisch-methodische Hinweise
|   `-- ...                     Ggf. Weiteres Material (Arbeitsblätter, ...)
|
`-- view                    (F) Implementierung eines Views in Go/PHP (Auswahl)
    |-- <db>-view.txt           Motivation, Anwendung und mgl. Erweiterungen des Views
    |-- <db>-htdocs             Webserver-Verzeichnisinhalt (nur PHP)
    |   |-- index.html/php      Startdatei im Webserververzeichnis
    |   |-- <db>.css            CSS-Datei (ggf.)
    |   `-- ...
    `-- <db>-go                 Go-Dateien (nur Go)
        |-- Pakete              Verwendete externe Go-Pakete (ggf.)
        |   |-- <pak1>
        |   `-- ...
        `-- <db>.go             Hauptprogramm-Datei
```

Präsentation

Bei der Präsentation wird eine kurze Einführung in die modellierte Miniwelt erwartet, wobei Entitäten und Beziehungen mit ihren Komplexitäten anhand eines ER-Diagramms vorzustellen sind. Die Relevanz des Weltausschnittes für SuS muss deutlich werden. Anschließend ist eine kleine Auswahl an Anfragen an das Relationenalgebra- bzw. SQL-System vorzuführen sowie der programmiertechnische Teil vorzustellen. Dabei soll insbesondere die Realitätsnähe des gewählten Views erläutert werden. Zentrale Teile des Codes sollen vorgestellt werden. Die Präsentation soll eine Dauer von 30 Minuten nicht überschreiten.

F.4. Bewertung des UDBP

Kandidat A: _____ Kandidat C: _____

Kandidat B: _____ Kandidat D: _____

Ermittlung des Gesamtergebnisses

Jeder Kandidat muss erkennbar an **mindestens zwei Bewertungskriterien der schriftlichen Ausarbeitung** (Qualität der Miniwelt, Modellierung, Entwurf und Population, Güte der Anfragen (SQL/RA) und Anwendungsprogramm) mitgewirkt haben. Für diese Kriterien erfolgt eine individuelle Bewertung. Zusammen mit der Bewertung der allgemeinen Aspekte ergeben sich daraus die individuellen Punktsummen und daraus die Anteile für die schriftliche Ausarbeitung. Bei fehlender Zuordnung wird davon ausgegangen, dass alle Mitglieder zu gleichen Teilen für alle Abschnitte der Ausarbeitung verantwortlich sind. Die Zuordnung des Gesamtanteils a zur Note erfolgt nach dem in der gymnasialen Oberstufe üblichen Schlüssel. Bei $a < 45\%$ gilt das Datenbankpraktikum als nicht bestanden.

Bedeutung und Bewertung der Merkmalsausprägungen

Symbol	textliche Beschreibung	Punkte
$++$	sehr ausgeprägt	5
$+$	ausgeprägt	4
$+/\circ$	im Allgemeinen ausgeprägt	3

Symbol	textliche Beschreibung	Punkte
$\circ/-$	teilweise ausgeprägt	2
$-$	kaum ausgeprägt	1
$--$	nicht vorhanden	0

Individuelle Gesamtergebnisse

	A	B	C	D
erreichter Anteil a für die schriftliche Ausarbeitung				
erreichte Note				

Bewertungsraster

schriftliche Ausarbeitung		++	+	+/○	○/−	−	−−
Qualität Miniwelt und ERM (Relevanz, Beschreibung, angemessene Komplexität, Umsetzung in das ER-Modell)	A						
	B						
	C						
	D						
Modellierung (Transformation ins RA-Modell, Begründung von Entwurfsentscheidungen, Funktionale Abhängigkeiten)	A						
	B						
	C						
	D						
Entwurf und Population (DDL-Qualität, Berücksichtigung von Integritätsbedingungen, Güte der Population)	A						
	B						
	C						
	D						
Güte der Anfragen (SQL/RA) (Progression, Vielfalt und Motivation, Alternativen, Korrektheit und Vollständigkeit, Ausnutzen von SQL-Spezifika)	A						
	B						
	C						
	D						
Anwendungsprogramm (Lauffähigkeit, Fehlertoleranz, Übersichtlichkeit und Wiederverwendbarkeit des Codes, Benutzerfreundlichkeit, Modularisierung für Unterrichtszwecke)	A						
	B						
	C						
	D						
Allgemeine Aspekte (Vollständigkeit und Übersichtlichkeit der Dokumentation, Einhalten der vorgegebenen Binnenstruktur, Erleichterung der Fremdnutzung des Projektes)							

F.5. Praktische Hinweise zum UDBP

Während der Arbeit im unterrichtsbezogenen Praktikum fallen viele Aufgaben an, die praktischer Natur sind und in der laufenden Vorlesung nicht oder nur peripher angesprochen wurden. In diesem Kapitel werden einige solcher Probleme kurz angesprochen, um den Start in die selbständige Arbeit zu erleichtern.

Hinweise zur Verwendung von DES

Datentypen Will man mit DES die Daten des `postgreSQL`-Datenbankservers verwenden, dürfen nur folgende Datentypen verwendet werden:

- `VARCHAR (n)`
- `VARCHAR`
- `INTEGER`
- `REAL`

Insbesondere sind Zeit- und Datums-Datentypen sowie Boolean-Datentypen **nicht** erlaubt. Um dieses Manko zu umgehen, könnte man für Boolean-Typen einen Integer-Typ verwenden, den man auf die Werte 0 und 1 beschränkt und Zeit- und Datumsangaben in Zeichenketten-Typen abspeichern. Eine syntaktische Prüfung auf solcherlei Datentypen zu realisieren wäre allerdings ein Overkill.

Integration in Geany Um ähnlich wie für `postgreSQL` auch ra-Dateien für DES in Geany durch einen Druck auf F5 in DES verarbeiten zu lassen und die Ausgabe gleich im Terminalfenster zu bekommen, muss eine Datei `des.sh` irgendwo im Pfad (z. B. in `/home/lewein/bin`) untergebracht werden, die folgenden Inhalt hat:[111]

```
1  #!/bin/bash
2
3  # HERE-Dokument - erzeugt die passende des.ini
4  cat > des.ini <<DES
5  /RA
6  /p $1
7  /q
8  DES
9
10 # startet des, das die des.ini einliest und die Ausgabe
11 # auf dem Terminal darstellt.
12 des
13 # automatisch erzeugte des.ini löschen
14 rm des.ini
```

Zusätzlich muss über das Menu in Geany das Kommando zu Erstellen konfiguriert werden. Unter `Ausführen` trägt man bei geöffneter ra-Datei ein

```
des.sh %f
```

Nun wird bei Betätigen der Taste F5 eine temporäre ini-Datei erzeugt, die DES beim Start einliest und die aktuelle ra-Datei verarbeitet. Die Ausgabe von DES erscheint dann im sich öffnenden Terminalfenster. Soll die Ausgabe – etwa zur weiteren Verwendung in der UDBP-Doku – noch zusätzlich in einer LOG-Datei protokolliert werden, muss die ra-Datei mit

[111] Die Datei findet man auch auf der Freigabe **udbp** im Ordner *Geany*.

```
/log <Dateiname>
```

beginnen und mit

```
/nolog
```

enden.

ODBC-Anbindung Für den Zugriff von DES auf bestehende Datenbanken – z. B. auf welche, die der postgreSQL-Server beheimatet – muss ggf. noch die ODBC-Unterstützung (*open database connection*) installiert werden. Dafür sind die Pakete

- `postgresql-odbc` und
- `libiodbc`

auszuwählen und zu installieren. Anschließend editiert man (als `root`(!)) im Ordner `/etc` die Datei `odbcinst.ini` fügt – falls sie noch nicht existieren – die Zeilen

```
1  [PostgreSQL]
2  Description = ODBC for PostgreSQL
3  Driver      = /usr/lib/psqlodbcw.so
4  Setup       = /usr/lib/libodbcpsqlS.so
5  Driver64    = /usr/lib64/psqlodbcw.so
6  Setup64     = /usr/lib64/libodbcpsqlS.so
7  FileUsage   = 1
8
9  [MySQL]
10 Description = ODBC for MySQL
11 Driver      = /usr/lib/libmyodbc5.so
12 Setup       = /usr/lib/libodbcmyS.so
13 Driver64    = /usr/lib64/libmyodbc5.so
14 Setup64     = /usr/lib64/libodbcmyS.so
15 FileUsage   = 1
```

ein. Danach erstellt man als `root` die Datei `/etc/odbc.ini` und fügt dort die Zeilen

```
1  [LWB]
2  Description = Tabellen der Lehrerweiterbildung
3  Driver      = PostgreSQL
4  Database    = lewein
5  Servername  = 127.0.0.1
6  User        = lewein
7
8  [PI]
9  Description = Datenbank von lewein auf fernem Pi
10 Driver      = PostgreSQL
11 Database    = pi
12 Servername  = 192.168.2.13
13 User        = lewein
14 Password    = niewel
```

ein. Nun können in DES mittels des Kommandos

```
/open_db 'LWB'
```

z. B. alle Tabellen der Datenbank `lewein` auf dem lokalen Rechner verfügbar gemacht werden.