WIRTSCHAFTS
UNIVERSITÄT
WIEN VIENNA
UNIVERSITY OF
ECONOMICS
AND BUSINESS

Datenorientierte Systemanalyse

07/05/2013

Axel Polleres

Datenorientierte Systemanalyse/ Datenanalyse

WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

- **Stundenwiederholung**
  - **Linux: Verzeichnisse & Dateien anlegen, loeschen, verschieben?**
  - **Eine einfache Webseite erstellen?**
- Erste Schritte mit PostgreSQL
- Entity-Relationship-Model & richtiges Strukturieren von Tabellen in einer Datenbank
- Daten filtern und abfragen mit SQL

Datenorientierte Systemanalyse/ Datenanalyse

WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

- Stundenwiederholung
- **Erste Schritte mit PostgreSQL**
  - **Wie legt man Tabellen und Datensätze in psql an?**
  - **Wie ändert man Datensätze?**
  - **Wie importiert man eine CSV Tabelle?**
- Entity-Relationship-Model & richtiges Strukturieren von Tabellen in einer Datenbank
- Daten filtern und abfragen mit SQL

PostgreSQL

WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

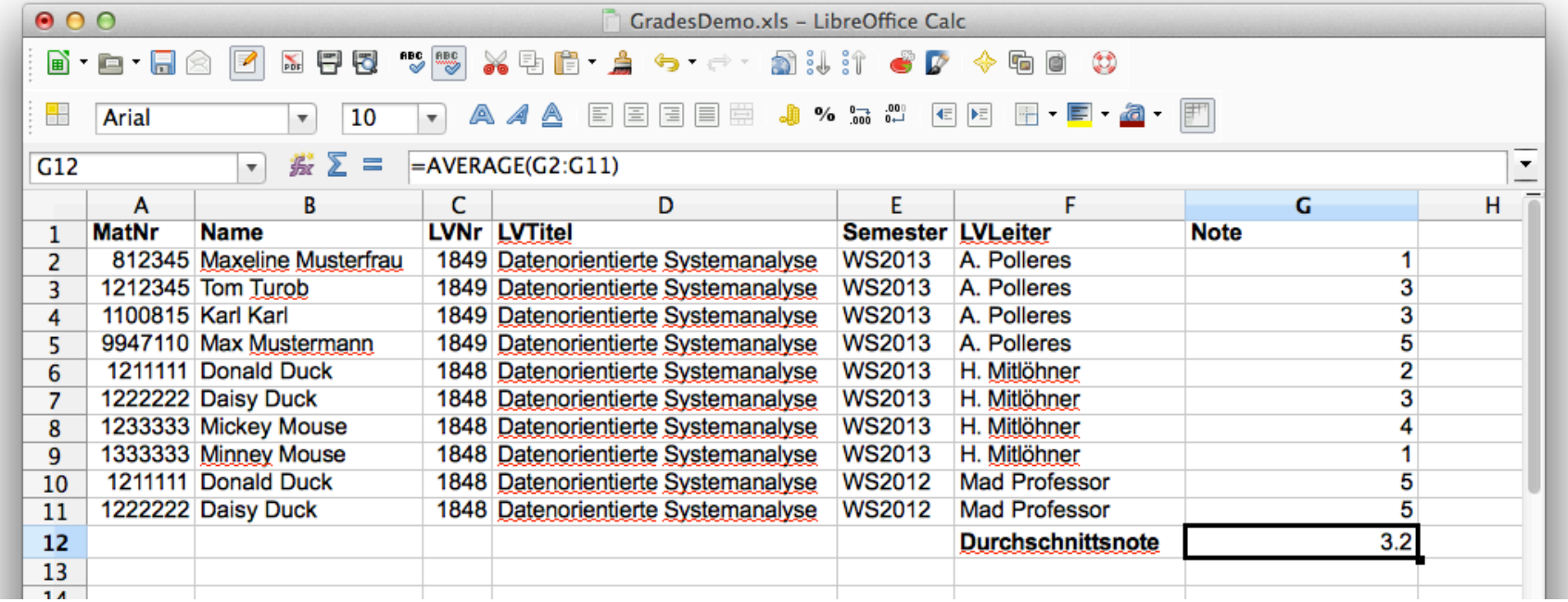
- Postgres über die Commandline starten:
  - `psql`
- Wichtige commands:
  - \? ... help
  - \d ... List all tables
  - \q ... Quit (oder CTRL-D)

- Ein Backup Ihrer Datenbank anlegen:
  - `pg_dump > backupfile.sql`
- Backup wieder in Ihre Datenbank einspielen:
  - `psql < backup-file.sql`

- Andere PostgreSQL Kommandos & Einführung in SQL:
  http://mitloehner.net/lehre/sql/all.html
  ... und im PostgreSQL Manual:
  http://www.postgresql.org/docs/manuals/
  - Viele andere Quellen & Tutorials online!

EFMD EQUIS ACCREDITED

Wie bekomme ich diese Tabelle in meine Datenbank?



	A	B	C	D	E	F	G	H
1	**MatNr**	**Name**	**LVNr**	**LVTitel**	**Semester**	**LVLeiter**	**Note**	
2	812345	Maxeline Musterfrau	1849	Datenorientierte Systemanalyse	WS2013	A. Polleres	1	
3	1212345	Tom Turob	1849	Datenorientierte Systemanalyse	WS2013	A. Polleres	3	
4	1100815	Karl Karl	1849	Datenorientierte Systemanalyse	WS2013	A. Polleres	3	
5	9947110	Max Mustermann	1849	Datenorientierte Systemanalyse	WS2013	A. Polleres	5	
6	1211111	Donald Duck	1848	Datenorientierte Systemanalyse	WS2013	H. Mitlöhner	2	
7	1222222	Daisy Duck	1848	Datenorientierte Systemanalyse	WS2013	H. Mitlöhner	3	
8	1233333	Mickey Mouse	1848	Datenorientierte Systemanalyse	WS2013	H. Mitlöhner	4	
9	1333333	Minney Mouse	1848	Datenorientierte Systemanalyse	WS2013	H. Mitlöhner	1	
10	1211111	Donald Duck	1848	Datenorientierte Systemanalyse	WS2012	Mad Professor	5	
11	1222222	Daisy Duck	1848	Datenorientierte Systemanalyse	WS2012	Mad Professor	5	
12						**Durchschnittsnote**	3.2	
13								

Die Datei finden Sie hier:
https://ai.wu.ac.at/~polleres/teaching/DOSA_2014/20140505/

PostgreSQL

- Tabellen anlegen:

```
CREATE TABLE tabellenname( spaltenname datentyp, ...);
```

  - http://www.postgresql.org/docs/8.4/static/sql-createtable.html

- Example:

```
CREATE TABLE grades_raw ( MatNr integer, Name varchar(50), LVNr integer,
LVTitel varchar(100), Semester char(6), LVLeiter varchar(50), Note integer );
```

  - Für jede Spalte in der Tabelle muss ein Datentyp definiert werden, siehe:
    http://www.postgresql.org/docs/8.4/static/datatype.html

PostgreSQL

- Tabelleninhalt anzeigen:

  `SELECT * FROM Tabellenname ;`

- Einen Datensatz einfügen:

  `INSERT INTO Tabellenname VALUES ( WertSpalte1, WertSpalte2, ...) ;`

  http://www.postgresql.org/docs/8.4/static/dml-insert.html

- Datensätze ändern:

  http://www.postgresql.org/docs/8.4/static/dml-update.html

- Datensätze löschen:

  http://www.postgresql.org/docs/8.4/static/dml-delete.html

  Finden Sie selbst weitere Befehle heraus:

  Tabelle löschen?

  Spalte in Tabelle hinzufügen?

  Constraints/Checks auf Spalten definieren?

Datenorientierte Systemanalyse/ Datenanalyse

WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

- Stundenwiederholung
- Erste Schritte mit PostgreSQL
- **Entity-Relationship-Model & richtiges Strukturieren von Tabellen in einer Datenbank**
- Daten filtern und abfragen mit SQL

Entity-Relationship-Model & richtiges Strukturieren von Tabellen in einer Datenbank

WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

- Wiederholung: Das Entity-Relationship-Modell

  http://mitloehner.net/lehre/erm/all.html

- Anlegen von entsprechenden Tabellen:
  - Wichtig:
    - PRIMARY KEY constraints, um Schlüssel anzulegen
    - REFERENCES (oder FOREIGN KEYs) um auf Schlüssel in anderen Tabellen zu referenzieren
  - http://www.postgresql.org/docs/8.1/static/ddl-constraints.html

EFMD EQUIS ACCREDITED

Entity-Relationship-Model & richtiges Strukturieren von Tabellen in einer Datenbank

WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

- Richtiges Strukturieren von Tabellen im ER-Modell:

  Folgende Probleme, wenn beim Anlegen von Tabellen Entitäten und deren Abhaengigkeiten untereinander nicht richtig erkannt/zerlegt werden:
  - Update Anomalien
  - Insert Anomalien
  - Delete Anomalien

- Hängt zusammen mit Datenbank-**Normalisierung**

  Hier eine vereinfachte Definition von 3. Normalform:

„Every non-prime attribute is non-transitively dependent on every candidate key in the table. The attributes that do not contribute to the description of the primary key are removed from the table. In other words, no transitive dependency is allowed.[within a table]"

EFMD EQUIS ACCREDITED

Beispiel nichtnormalisierte „Datenbank“:

- Es bestehen etliche interne Abhängigkeiten:

	A	B	C	D	E	F	G
1	**MatNr**		**LVNr**	**LV-Titel**	**Semester**	**LV-Leiter**	**Note**
2	812345	Maxeline Musterfrau	1849	Datenorientierte Systemanalyse	WS2013	A. Polleres	1
3	1212345	Tom Turob	1849	Datenorientierte Systemanalyse	WS2013	A. Polleres	3
4	1100815	Karl Karl	1849	Datenorientierte Systemanalyse	WS2013	A. Polleres	3
5	9947110	Max Mustermann	1849	Datenorientierte Systemanalyse	WS2013	A. Polleres	5
6	1211111	Donald Duck	1848	Datenorientierte Systemanalyse	WS2013	H. Mitlöhner	2
7	1222222	Daisy Duck	1848	Datenorientierte Systemanalyse	WS2013	H. Mitlöhner	3
8	1233333	Mickey Mouse	1848	Datenorientierte Systemanalyse	WS2013	H. Mitlöhner	4
9	1333333	Minney Mouse	1848	Datenorientierte Systemanalyse	WS2013	H. Mitlöhner	1
10	1211111	Donald Duck	1848	Datenorientierte Systemanalyse	WS2012	Mad Professor	5
11	1222222	Daisy Duck	1848	Datenorientierte Systemanalyse	WS2012	Mad Professor	5
12							
13							
14							
15							
16						**Durchschnittsnote:**	**2.75**

- MatNr → Name
- LVNr → LVTitel
- LVNr, Semester → LVLeiter
- MatNr, LVNr, Semester → Note

Beispiel **Update Anomalie:**

WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

- Update eines Datensatzes erzeugt Inkonsistenz im Bezug auf tatsächliche Abhängigkeiten:

	A	B	C	D	E	F	G
1	**MatNr**	**Name**	**LVNr**	**LVTitel**	**Semester**	**LVLeiter**	**Note**
2	812345	Maxeline Musterfrau	1849	Datenorientierte Systemanalyse	WS2013	A. Polleres	1
3	1212345	Tom Turob	1849	Datenorientierte Systemanalyse	WS2013	A. Polleres	3
4	1100815	Karl Karl	1849	**Datenanalyse**	WS2013	A. Polleres	3
5	9947110	Max Mustermann	1849	Datenorientierte Systemanalyse	WS2013	A. Polleres	5
6	1211111	Donald Duck	1848	Datenorientierte Systemanalyse	WS2013	H. Mitlöhner	2
7	1222222	Daisy Duck	1848	Datenorientierte Systemanalyse	WS2013	H. Mitlöhner	3
8	1233333	Mickey Mouse	1848	Datenorientierte Systemanalyse	WS2013	H. Mitlöhner	4
9	1333333	Minney Mouse	1848	Datenorientierte Systemanalyse	WS2013	H. Mitlöhner	1
10	1211111	Donald Duck	1848	Datenorientierte Systemanalyse	WS2012	Mad Professor	5
11	1222222	Daisy Duck	1848	Datenorientierte Systemanalyse	WS2012	Mad Professor	5

- MatNr → Name
- **LVNr → LVTitel**
- LVNr, Semester → LVLeiter
- MatNr, LVNr, Semester → Note

EFMD EQUIS ACCREDITED

Beispiel **Insert Anomalie:**

- Nicht möglich neue Studenten ohne Note oder ohne Kursanmeldung zu speichern:

	A	B	C	D	E	F	G
1	**MatNr**	**Name**	**LVNr**	**LVTitel**	**Semester**	**LVLeiter**	**Note**
2	812345	Maxeline Musterfrau	1849	Datenorientierte Systemanalyse	WS2013	A. Polleres	1
3	1212345	Tom Turob	1849	Datenorientierte Systemanalyse	WS2013	A. Polleres	3
4	1100815	Karl Karl	1849	Datenorientierte Systemanalyse	WS2013	A. Polleres	3
5	9947110	Max Mustermann	1849	Datenorientierte Systemanalyse	WS2013	A. Polleres	5
6	1211111	Donald Duck	1848	Datenorientierte Systemanalyse	WS2013	H. Mitlöhner	2
7	1222222	Daisy Duck	1848	Datenorientierte Systemanalyse	WS2013	H. Mitlöhner	3
8	1233333	Mickey Mouse	1848	Datenorientierte Systemanalyse	WS2013	H. Mitlöhner	4
9	1333333	Minney Mouse	1848	Datenorientierte Systemanalyse	WS2013	H. Mitlöhner	1
10	1211111	Donald Duck	1848	Datenorientierte Systemanalyse	WS2012	Mad Professor	5
11	1222222	Daisy Duck	1848	Datenorientierte Systemanalyse	WS2012	Mad Professor	5

- MatNr → Name
- LVNr → LVTitel
- LVNr, Semester → LVLeiter
- MatNr, LVNr, Semester → Note

9011111	Dagobert Duck	?

Beispiel **Delete Anomalie:**

- Wenn ich die Anmeldung von „Tom Turbo“ lösche, ist der Student damit „komplett verschwunden“:

	A	B	C	D	E	F	G
1	**MatNr**	**Name**	**LVNr**	**LVTitel**	**Semester**	**LVLeiter**	**Note**
2	812345	Maxeline Musterfrau	1849	Datenorientierte Systemanalyse	WS2013	A. Polleres	1
3	1212345	Tom Turob	1849	Datenorientierte Systemanalyse	WS2013	A. Polleres	3
4	1100815	Karl Karl	1849	Datenorientierte Systemanalyse	WS2013	A. Polleres	3
5	9947110	Max Mustermann	1849	Datenorientierte Systemanalyse	WS2013	A. Polleres	5
6	1211111	Donald Duck	1848	Datenorientierte Systemanalyse	WS2013	H. Mitlöhner	2
7	1222222	Daisy Duck	1848	Datenorientierte Systemanalyse	WS2013	H. Mitlöhner	3
8	1233333	Mickey Mouse	1848	Datenorientierte Systemanalyse	WS2013	H. Mitlöhner	4
9	1333333	Minney Mouse	1848	Datenorientierte Systemanalyse	WS2013	H. Mitlöhner	1
10	1211111	Donald Duck	1848	Datenorientierte Systemanalyse	WS2012	Mad Professor	5
11	1222222	Daisy Duck	1848	Datenorientierte Systemanalyse	WS2012	Mad Professor	5

?

- MatNr → Name
- LVNr → LVTitel
- LVNr, Semester → LVLeiter
- MatNr, LVNr, Semester → Note

Ziel der Normalisierung:

- Zerlegen von Tabellen, sodass
  - keine Daten redundant gespeichert werden
  - Anomalien wo immer möglich vermieden werden

Vereinfachte Definition von 3. Normalform:

„Every non-prime attribute is non-transitively dependent on every candidate key in the table. The attributes that do not contribute to the description of the primary key are removed from the table. In other words, no transitive dependency is allowed.[within a table]"

- **Übung**: Zerlegen Sie die Tabelle der vorigen Folien in Ihre Entities und zeichne ein ER-Diagramm:
  - Student
  - Course
  - Course instance
  - Teacher

- Legen Sie die daraus entstehenden Tabellen (Hinweis: 5) in Ihrer Datenbank mit CREATE TABLE an...

Constraints

- `NOT NULL` ... Spalte darf nicht leer sein
- `PRIMARY KEY (`*`Column1`*`,...)` ... Eindeutiger Schlüssel, kann aus mehreren Elementen bestehen
- `FOREIGN KEY (`*`Column1`*`,...) REFERENCES` *`Reftable`* `(`*`Refcolumn1`*`, ...)` ... Spalte(n) die Fremdschlüssel von anderen Tabellen enthalten müssen.
- `CHECK (` *`Bedingung`* `)`

- Constraints können entweder direkt bei C REATE TABLE angegeben werden, oder nachträglich mit ALTER TABLE eingefügt werden, siehe:
- http://www.postgresql.org/docs/8.1/static/sql-createtable.html
- http://www.postgresql.org/docs/8.1/static/sql-altertable.html
- Beispiel:

```
ALTER TABLE grades_raw ADD CHECK ( semester similar to '(WS|SS)[0-9][0-9][0-9][0-9]' );
```

ER-Diagram:

WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

1) Identify **entities** (and dependencies)
2) Think about **relations** between those entities
3) For 1:n relations think about which side of the relation is optional.
4) If there are m:n relations, decompose them into a new intermediate relation.

- *Beispiel am whiteboard/flipchart...*

- Diskussion: Oft ist es in praktischer Modellierung ein **trade-off** wie weit man bei der normalisierung geht! Manchmal sind Tabellen mit NULL Werten eine Alternative.

EFMD EQUIS ACCREDITED

Lösung:

- Siehe https://ai.wu.ac.at/~polleres/teaching/DOSA_2014/20140507/GradesDemoCreateTables.txt

- Nächster Schritt:
- Tabellen mit Daten befüllen ...?
  - Händisch:
    - `INSERT INTO TabelleZiel VALUES ( ... );`
  - Aus einer bestehenden Tabelle:
    - `INSERT INTO TabelleZiel SELECT Spalte1, Spalte 2 FROM TabelleQuelle ;`
    - Um Duplikate zu vermeiden verwenden Sie das Schlüsselwort DISTINCT:
    - `INSERT INTO TabelleZiel SELECT DISTINCT Spalte1, Spalte 2 FROM TabelleQuelle ;`

- Übung: Füllen Sie die Tabellen, die sie angelegt haben mit den Werten aus der CSV-Datei https://ai.wu.ac.at/~polleres/teaching/DOSA_2014/20140505/GradesDemo.csv

Lösung sollte so aussehen:

WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

```
apollere=> SELECT * FROM Student;
  matnr  |         name
---------+----------------------
  812345 | Maxeline Musterfrau
 1100815 | Karl Karl
 1211111 | Donald Duck
 1212345 | Tom Turob
 1222222 | Daisy Duck
 1233333 | Mickey Mouse
 1333333 | Minney Mouse
 9947110 | Max Mustermann
```

```
apollere=> SELECT * FROM Course;
 lvnr |             lvtitel
------+---------------------------------
 1848 | Datenorientierte Systemanalyse
 1849 | Datenorientierte Systemanalyse
```

```
apollere=> SELECT * FROM CourseInstance;
 lvnr | semester |   lvleiter
------+----------+---------------
 1848 | WS2012   | Mad Professor
 1848 | WS2013   | H. Mitlöhner
 1849 | WS2013   | A. Polleres
```

```
apollere=> SELECT * FROM Grades;
 lvnr | semester |  matnr  | note
------+----------+---------+------
 1848 | WS2012   | 1211111 |    5
 1848 | WS2012   | 1222222 |    5
 1848 | WS2013   | 1211111 |    2
 1848 | WS2013   | 1222222 |    3
 1848 | WS2013   | 1233333 |    4
 1848 | WS2013   | 1333333 |    1
 1849 | WS2013   |  812345 |    1
 1849 | WS2013   | 1100815 |    3
 1849 | WS2013   | 1212345 |    3
 1849 | WS2013   | 9947110 |    5
```

```
apollere=> SELECT * FROM Teacher;
     name
--------------
 A. Polleres
 H. Mitlöhner
 Mad Professor
```

Abfragen?

- Weitere Beispiele fuer SQL Abfragen sollten Sie schon gesehen haben, wenn Sie sich das Tutorial angesehen haben:
  - http://xmdimrill.ai.wu-wien.ac.at/~mitloehn/lehre/sql/all.html
- Wir gehen diese nun exemplarisch anhand unseres Beispiels durch.

Filtern von Datensaetzen mittels WHERE und sortieren (ORDER BY)

- *Matrikelnummern von Studierende, deren Name mit 'M' beginnt:*

```
SELECT MatNr FROM Student WHERE Name LIKE 'M%';
```

  - siehe http://www.postgresql.org/docs/8.1/static/functions-matching.html

- Namen von Studierende, die vor 2008 zu studieren begonnen haben, sortiert nach Namen absteigend *(Hinweis: Matrikelnummern von 2014 beginnen mit 14)*

```
SELECT Name FROM Student
WHERE (MatNr < 900000) OR (MatNr > 1500000) ORDER BY Name DESC;
```

Abfragen von Datensätzen meherer Tabellen (Join)

- Namen (ohne Duplikate) aller Studierenden die jemals eine negative Note erhalten haben...

```
SELECT Name from Student, Grades WHERE Student.MatNr =
Grades.MatNr AND Note = 5;
```

- ... inkl. der Namen der Profs und LVTitel:

```
SELECT S.Name, Ci.LVLeiter, C.LVTitel
FROM Student S, Grades G, Course C, CourseInstance Ci
WHERE S.MatNr = G.MatNr
  AND Ci.lvnr = G.lvnr
  AND Ci.semester = G.semester
  AND Ci.lvnr = C.lvnr
  AND G.Note = 5;
```

Vereinigugn der Ergebnisse mehrerer Anfragen (UNION)

- Namen von Studierenden ODER LVLeiter/inn/en sortiert nach Name:

```
SELECT Name FROM Student UNION SELECT Name
from Teacher ORDER BY Name;
```

Aggregation & Gruppierung:

- Durchschnittsnote (über alle vergebenen Noten):

```
SELECT AVG(Note)
FROM Grades;
```

or

```
SELECT AVG(Note) AS Durschschnittsnote
FROM Grades;
```

- ... pro LV-Instanz:

```
SELECT LVNr, Semester, AVG(Note)
FROM Grades G
GROUP BY LVNr, Semester;
```

Aggregation & Gruppierung:

- Wie oft hat jede/r Prof pro LV-Instanz eine negative Note vergeben?

```
SELECT Ci.LVLeiter, Ci.LVNr, Ci.Semester,
COUNT(G.Note)
FROM Grades G, Course C, CourseInstance Ci
WHERE Ci.lvnr = G.lvnr
   AND Ci.semester = G.semester
   AND Ci.lvnr = C.lvnr
   AND G.Note = 5
GROUP BY Ci.LVLeiter, Ci.LVNr, Ci.Semester;
```

Aggregation & Gruppierung:

- Wie oft hat jede/r Prof pro LV-Instanz eine negative Note vergeben absteigend sortiert nach Anzahl der „Fünfer“?

```
SELECT Ci.LVLeiter, Ci.LVNr, Ci.Semester,
COUNT(G.Note) AS Fuenfer
FROM Grades G, Course C, CourseInstance Ci
WHERE Ci.lvnr = G.lvnr
   AND Ci.semester = G.semester
   AND Ci.lvnr = C.lvnr
   AND G.Note = 5
GROUP BY Ci.LVLeiter, Ci.LVNr, Ci.Semester
ORDER BY Fuenfer DESC;
```

Hausübung (nächste Woche!):

WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

- Überlegen Sie sich ein **eigenes** praktisches Datenbank-Schema und Erstellen Sie ein ER-Modell für Ihre Datenbank
  - Verwenden Sie PRIMARY KEY constraints und REFERENCES constraints richtig
- Formulieren Sie SQL-Abfragen über Ihre Datenbank:
  - Mind. eine Abfage, die mehrere Tabellen verknüpft (Join)
  - Filtern von Datensätzen nach verschiedenen Kriterien (WHERE)
  - Mind. eine UNION query
  - Mind. eine Abfrage, die Sortierung und Aggregation verwendet (ORDER BY, GROUP BY, COUNT, AVG, SUM...)
- Letzmögliche Abgabe: Sonntag 18.05.2014, 20:00
- What's next?:
  - Mehr SQL, ein einfaches Web-Interface erstellen.
  - Andere Datenformate abfragen, importieren/exportieren (CSV, RDF)
  - Eine Webseite mit Diagrammen erstellen

EFMD EQUIS ACCREDITED