Hasso Plattner Institut

IT Systems Engineering | Universität Potsdam

Datenbankentwurf & Datenbankzugriff mit JDBC

Georg Köster

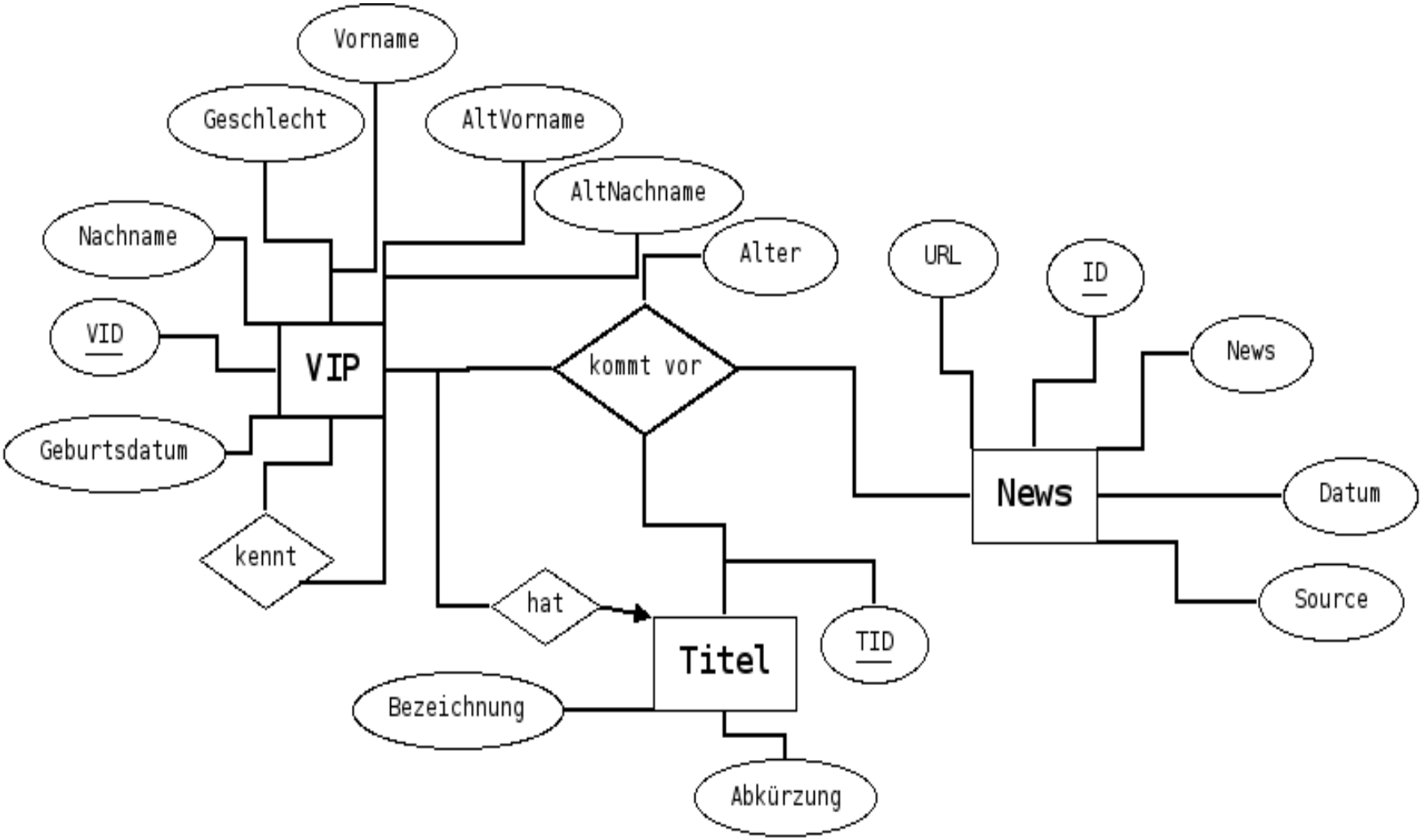
Sven Wagner-Boysen

6. November 2007

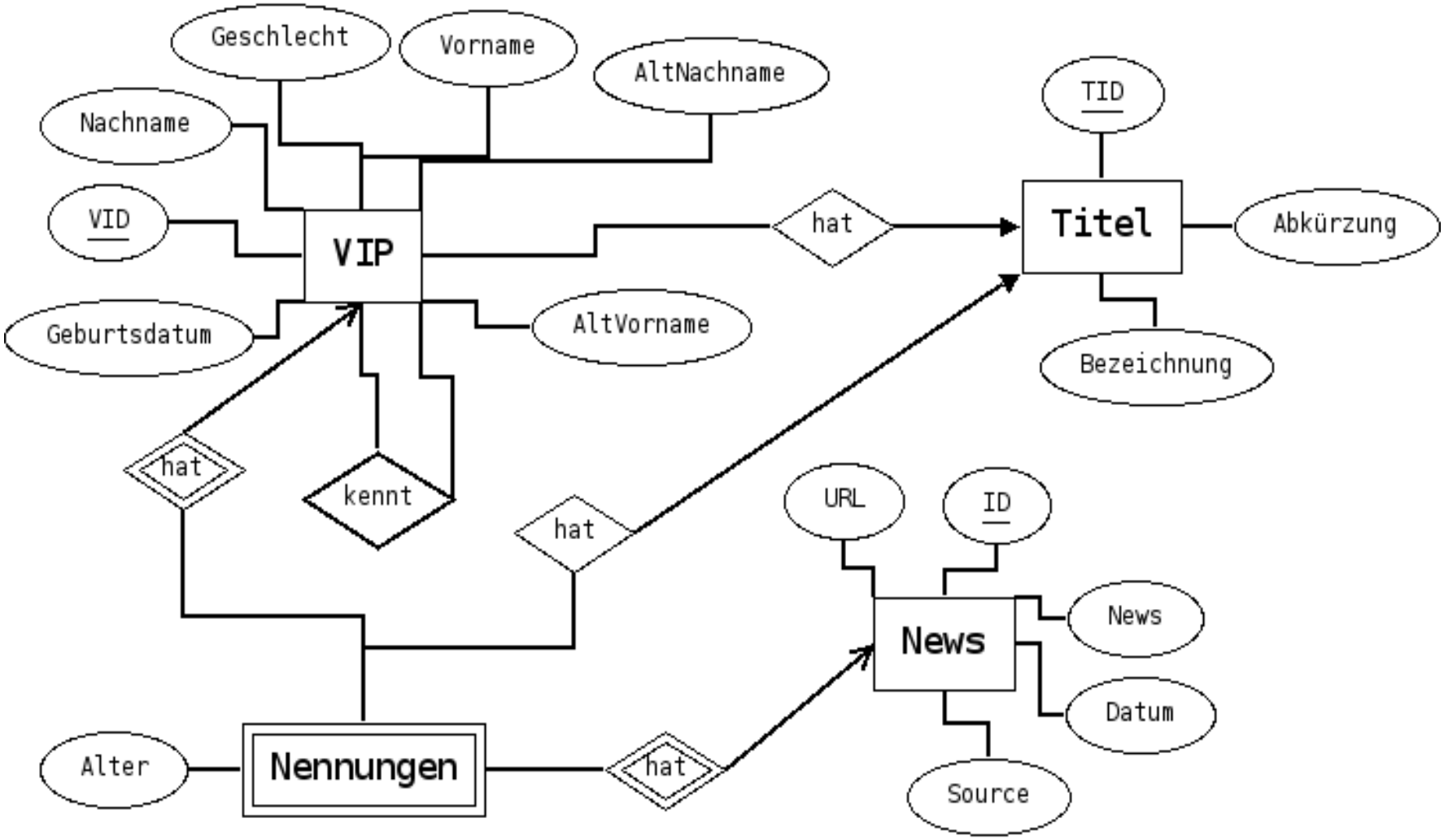
Gliederung

- Datenbankentwurf für ProminentPeople.info
  - ER-Modell
  - Relationaler Entwurf
  - Normalisierung
- Datenbankzugriff mit JDBC
  - Einführung
  - Verbindungsaufbau-/abbau
  - Datenbankabfrage
  - Ergebnisverarbeitung
- Vorstellung der Java-Bibliothek für Datenbankzugriff

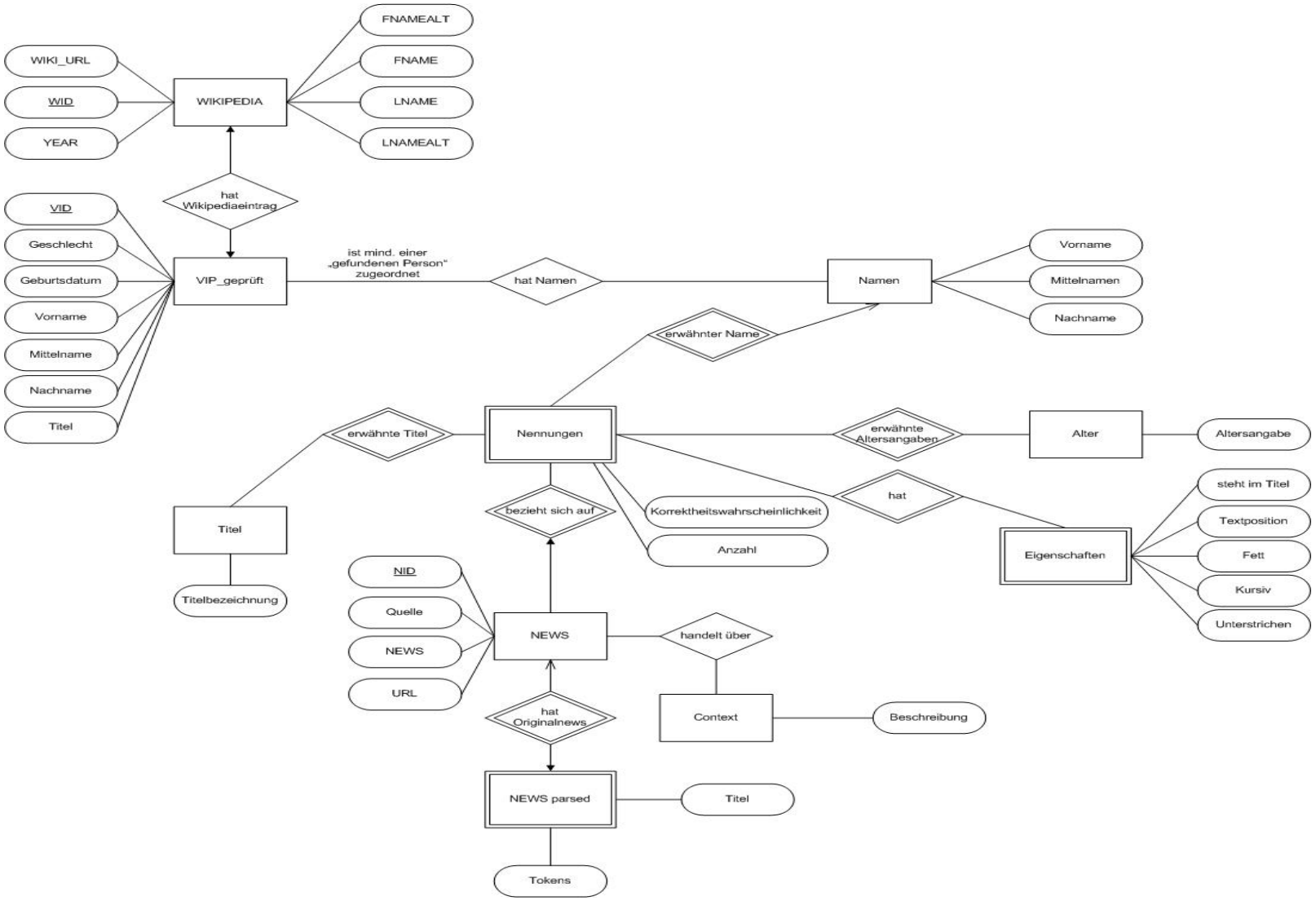
www.ProminentPeople.info
ER – Modell Vorschlag 1



www.ProminentPeople.info
ER – Modell Vorschlag 2



www.ProminentPeople.info
ER – Modell Vorschlag 2



www.ProminentPeople.info
Relationales Modell

VIP(<u>VID</u>, firstname, lastname, altfirstname, altlastname, gender, <u>title</u>, birthdate)

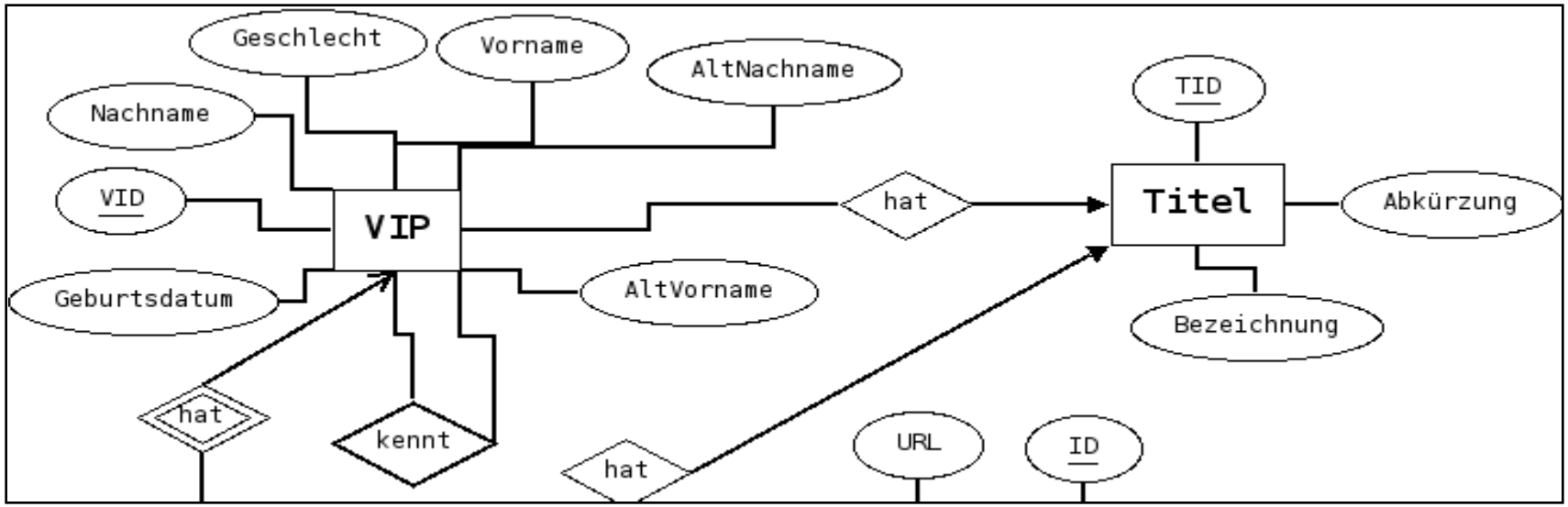
firstname, lastname NOT NULL

TITLE(<u>TID</u>, title, short)

title, short NOT NULL

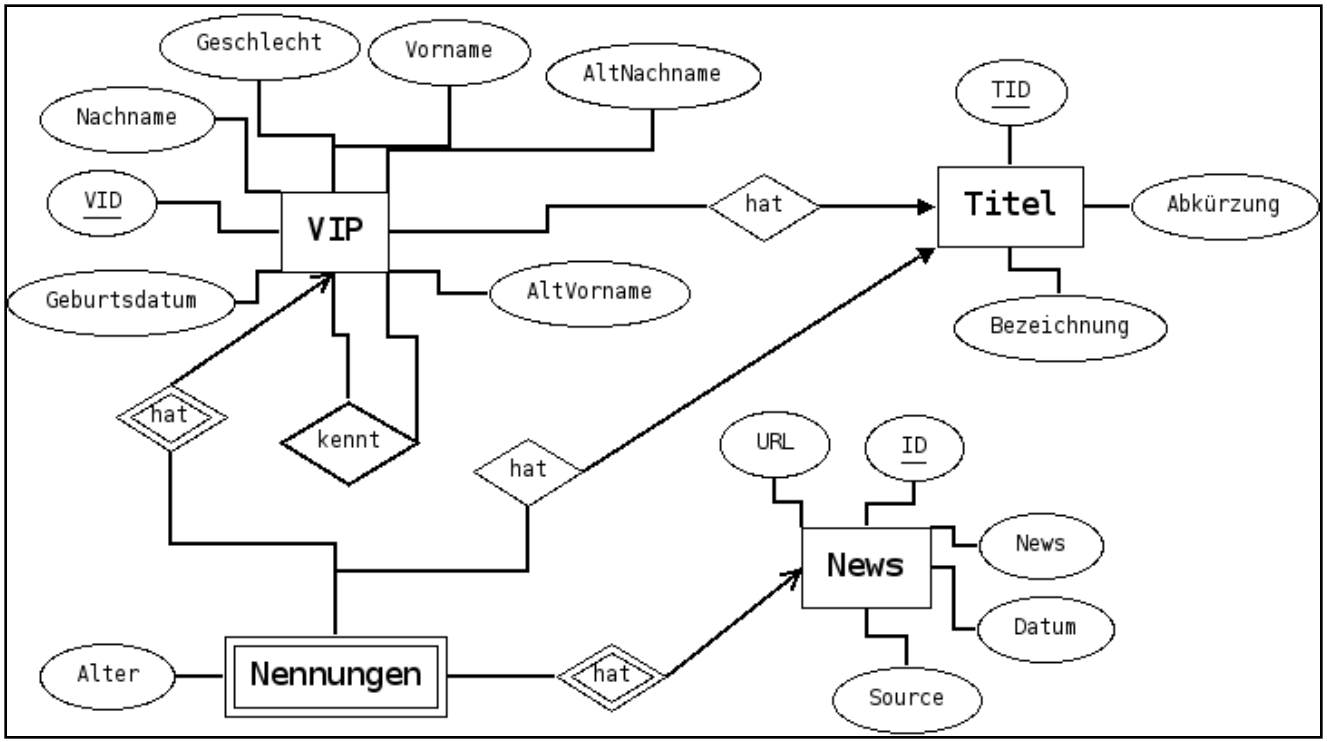
KNOWS(VID1, VID2)

VID1, VID2 NOT NULL



www.ProminentPeople.info
Relationales Modell

MENTION(VID, NID, TID, AGE)



Normalisierung

- Erste Normalform
  - liegt vor wenn alle Attribute nur atomare Werte beinhalten
  - keine Wiederholungsgruppen

NEWS_ID	VIP_MIT_ALTER
1200	{Sven 20, Georg 21, Alexander 21}
1201	Fabian 21
1202	Christiana 20

  - Lösung: Attribut aufteilen und neue Zeilen einfügen

NEWS_ID	VIP	Alter
1200	Sven	20
1200	Georg	21
1200	Alexander	21
1201	Fabian	21
1202	Christiana	20

Normalisierung

- Zweite Normalform
  - Relation befindet sich in der ersten Normalform
  - jedes Nicht-Schlüssel-Attribut ist funktional abhängig vom Gesamtschlüssel
  - aber nicht von einzelnen Schlüsselteilen

VID	**NID**	**Datum**	**Alter**
1	5	06.11.2007	30
3	5	06.11.2007	25

Normalisierung

- Lösung: Aufteilen der Daten in 2 Tabellen

VID	NID	Alter
1	5	30
3	5	25

NID	Datum
5	06.11.2007

Normalisierung

VIP(VID, firstname, lastname, altfirstname, altlastname, gender, title, birthdate)

(1, Georg, Köster, NULL, NULL, m, 1, 12.09.1986)

TITLE(TID, title, short)

(1, Herr, Hr)

KNOWS(VID1, VID2)

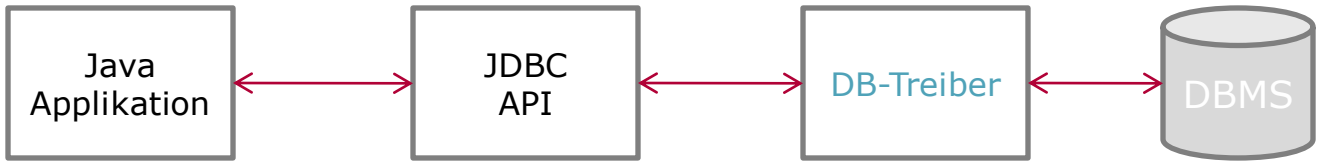
(1, 20)

MENTION(VID, NID, TID, AGE)

(34, 1000, 2, 40)

Datenbankzugriff mit JDBC - Einführung

- JDBC Abk. für Java Database Connectivity
- Satz von Schnittstellen zum Zugriff von Java Programmen auf relationale Datenbanken
- Ziel: Abstraktion von der konkret eingesetzten Datenbank
- objektorientierter Ansatz von Java verfolgt
- Kommunikationsschema:



- http://developers.sun.com/product/jdbc/drivers

JDBC - Treibertypen

Typ1: JDBC – ODBC – Brücke:

- nutzt ODBC-Standard von Microsoft; weitverbreitet auf Windowssystemen
- Performanceschwächen, plattformabhängig, ungeeignet für den Einsatz im Netz

Typ2: Native plattformeigene JDBC – Treiber

- JDBC-Aufrufe werden direkt in Aufrufe des DB-API übersetzt

Typ3: universelle JDBC – Treiber

- Kommunikation über Middleware; komprimiertes Protokoll
- günstig für Internetdienste

Typ4: direkte Netzwerktreiber

- direkte Verbindung zur DB über offenen IP-Port

JDBC – Schritte zur Datenbankabfrage
Treiber laden und Verbindungsaufbau

import java.sql.* //JDBC - Klassen laden

1. Installieren der JDBC – Datenbanktreiber

```
static Class forName(String className)
    throws ClassNotFoundException
```

Bsp.: `Class.forName("com.ibm.db2.jcc.DB2Driver");`

   – Java6 ist in der Lage Treiber selbst zu ermitteln

2. Verbindungsaufbau mit den Klassen DriverManager, Connection

```
static Connection getConnection(String URL, String user, String password)
    throws SQLExecption
```

URL-Format: jdbc:<Subprotokoll>:<Datenbankquelle>

Bsp.:

```
Connection con = DriverManager.getConnection(
    "jdbc:db2://HPI:12345/PEOPLE",
    "prominent","2007");
```

JDBC – Schritte zur Datenbankabfrage Statement - Objekt

3. SQL – Anweisung erzeugen und ausführen

```
Statement <Connection>.createStatement() throws SQLException
```

Bsp.: `Statement stmt = con.createStatement();`

Ausführen von "SELECT" – Anfragen mit

```
ResultSet executeQuery(String query)throws SQLException
```

"INSERT" "UPDATE" "DELETE" mit `int executeUpdate(String sql)`

Bsp.: `ResultSet rs = stmt.executeQuery("SELECT * FROM NEWS");`

- Einfügen und Ändern großer Datenmengen mittels Batch-Update

```
stmt.addBatch(String sql)
stmt.addBatch("INSERT INTO NEWS VALUES(....));
                              ...
stmt.executeBatch();
```

- Fehlerbehandlung über die BatchUpdateException

JDBC – Schritte zur Datenbankabfrage PreparedStatement - Objekt

- Anfrage bereits für das interne DB-Format vorbereite

```
PreparedStatement <Connection>.createPreparedStatement(String SQL)
  throws SQLException
```

Bsp.:
```
PreparedStatement pstmt = con.createPreparedStatement("UPDATE
  NEWS SET QUELLE = ? WHERE QUELLE Like ?");
```

- Geschwindigkeitsvorteil durch den Einsatz von Platzhaltern ("?") und Schleifen

```
pstmt.addString(1, "Bunte");
pstmt.addString(2, "BunteOnline");
pstmt.executeUpdate();
```

JDBC – Schritte zur Datenbankabfrage
Ergebnis der Anfrage verarbeiten

- Verarbeiten von ResultSets nach Aufruf von executeQuery()
  - Aufruf von `Boolean next()` im ResultSet setzt internen Cursor auf erste Ergebniszeile, dann jeweils auf die folgende
  - falls es keine weitere Zeile gibt, gibt `next() false` zurück
  - Auswertung des jeweiligen Spalteneintrags mit

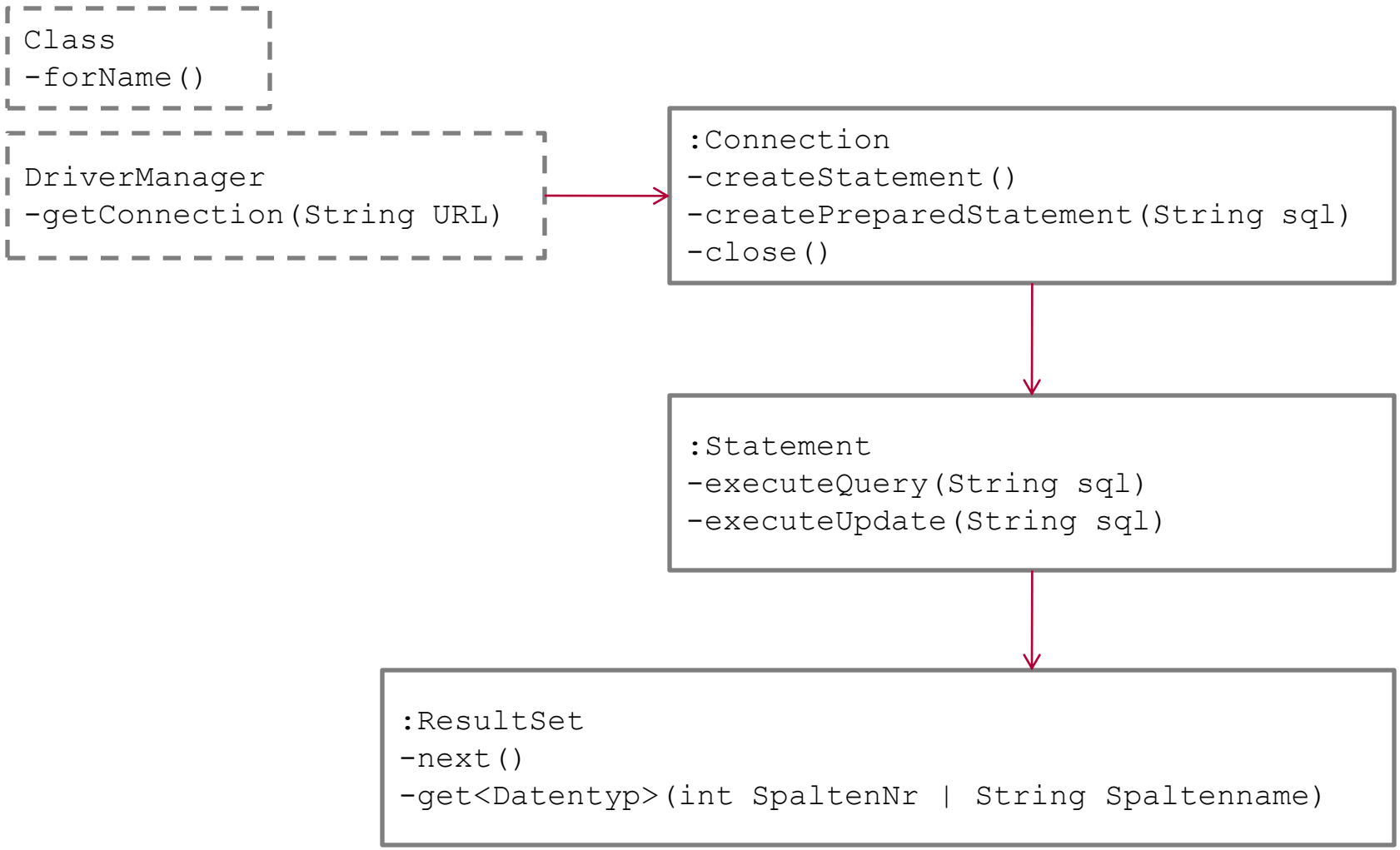
```
get<Datentyp der Spalte>(int SpaltenNr | String Spaltenname)
```

Bsp.:

```
ResultSet rs = stmt.executeQuery("SELECT * FROM NEWS");
while(rs.next()) {
   System.out.println(rs.getString("QUELLE");
}
```

- Verbindungsabbau mit <Connection>.close();

JDBC – Klassen-/Objektübersicht



Fehlerbehandlung mit JDBC

- Basisklasse ist SQLException
  - enthält Fehlerbeschreibung, sowie eine weitere Beschreibung des SQL-Status, und eine Ganzzahl vom Datenbanktreiber

- SQLWarning enthält keine kritischen Fehler, sondern Warnung
  - bereitgestellt von ResultSet-, Connection-, Statementobjekten
  - muss explizit ausgegeben werden über

```
SQLWarning getWarnings()
```

  - andernfalls wird das SQLWarning überschrieben

Quellen

- Galileo openbook - Java ist auch eine Insel
- Database Systems – The Complete Book
- Handbuch der Java-Programmierung
- JDBC-Einführung
  http://web.f4.fhtw-berlin.de/hartwig/JDBC/jdbc.html
- http://de.wikipedia.org/wiki/Normalisierung_(Datenbank)
- Vorlesung Datenbanksysteme 1