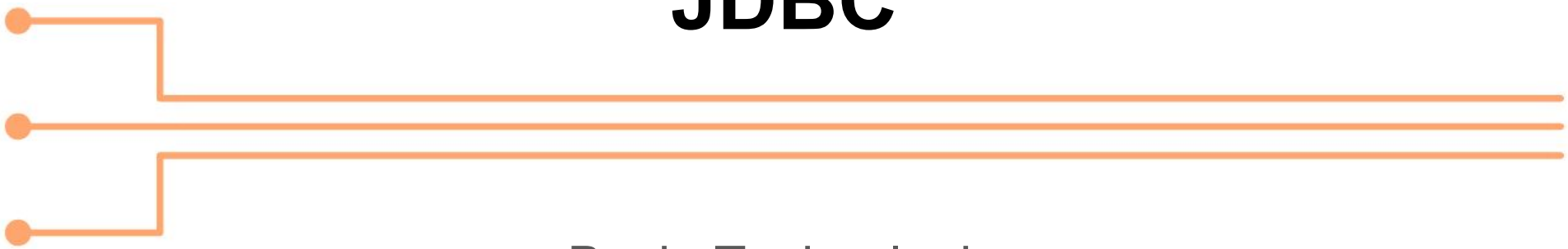
Datenbankanwendungen mit **JDBC**



Basis-Technologie

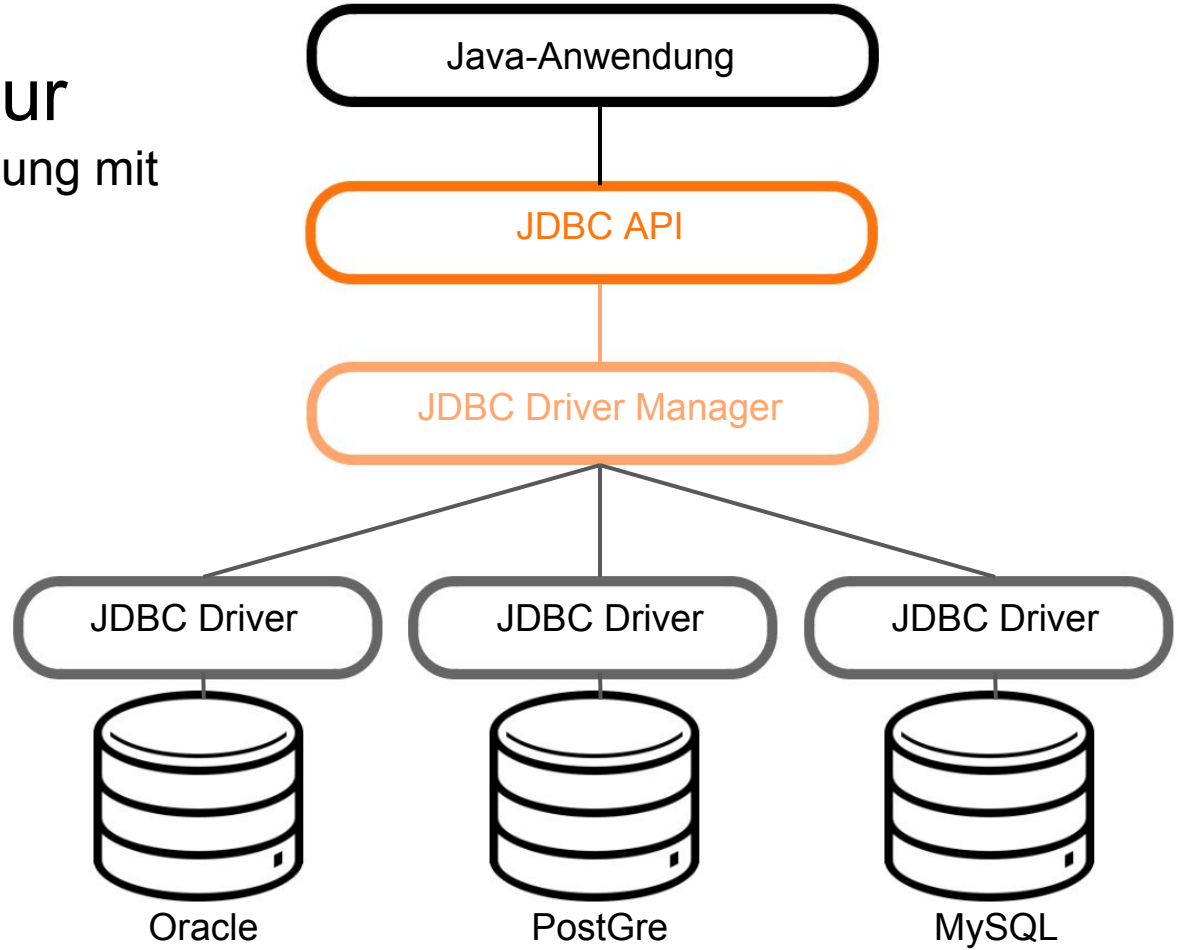
Datenmanagement

JDBC

- JDBC = **J**ava **D**ata**b**ase **C**onnectivity
- API für Java
- Herstellerunabhängige Verbindung von Java-Anwendungen zu Datenbanken
- JDBC Kernfunktionalitäten
  - Verbindungsmanagement
  - SQL-Queries
  - Verarbeitung von Ergebnismengen

Architektur

Java-Anwendung mit JDBC



[...]

JDBC Treiber

- MYSQL
  http://www.mysql.de/products/connector/
  https://mvnrepository.com/artifact/mysql/mysql-connector-java (Maven Dependency)
- PostGre
  http://jdbc.postgresql.org/
  https://mvnrepository.com/artifact/org.postgresql/postgresql (Maven Dependency)
- Oracle
  http://www.oracle.com/technetwork/database/features/jdbc/index-091264.html

- **Connection**
  Verbindungen zum Server bzw. zur DB aufbauen
- **Statement**
  Abfragen generieren und zum Server schicken
- **ResultSet**
  Durch Ergebnismengen browsen

Dokumentation & Tutorial

https://docs.oracle.com/javase/7/docs/api/java/sql/package-summary.html
https://docs.oracle.com/javase/tutorial/jdbc/basics/index.html

Verbindung

- Datenbankverbindung aufbauen
  mit `DriverManager` und einem speziellen Connection-String

```
Connection c = DriverManager.getConnection("jdbc:postgresql://localhost:5432/myDB",
                                           "user", "password");
```

- JRE findet und lädt autom. den richtigen Treiber
  - Connection-String gibt Treiber an
  - Treiber ist im Buildpath referenziert postgresql[...].jar

Abfragen

- `Statement` für SQL Abfragen

```
Statement stmt = c.createStatement();
ResultSetrs= stmt.executeQuery("SELECT persno, fname, lnameFROM person");
```

- `ResultSet` zum Lesen und durchlaufen der Ergebnismenge

```
while (rs.next()) {
     Person p = new
     Person();
     p.setPersno(rs.getInt(“persno"));
     p.setFname(rs.getString(“fname”));
     p.setLname(rs.getString(“lname”));
     System.out.println(p.toString());
}
```

Ergebnismengen

- `ResultSet` = Ergebnismenge
- Methoden zur Navigation im ResultSet

```
next()
previous()
first()
last()
beforeFirst()
afterLast()
relative()
absolute()
```

- Methoden zum Abfragen und Manipulieren von Spalten

```
getX(“key”);
updateX(“key”);
```

(Wert vom Typ X aus Spalte mit Name “key” lesen)

(Wert vom Typ X in Spalte mit Name “key” verändern)

Arbeiten mit Ergebnismengen

- Aktualisierbare Statements

```
Statement stmt = con.createStatement(
                         ResultSet.TYPE_SCROLL_SENSITIVE,
                         ResultSet.CONCUR_UPDATABLE);
ResultSet result = stmt.executeQuery("SELECT * FROM person");
```

- Updates auf speziellen Zellen durchführen

```
result.next();
result.updateString(„fname“,„Franz“);
result.cancelRowUpdates();
result.updateString(„fname“, „Peter“);
```

- Updates für einzelne Zeilen committen

```
result.updateRow();
```

Prepared Statements

- `PreparedStatement` ist wie Lückentext für SQL

```
PreparedStatement stmt = con.prepareStatement(
                                „SELECT * FROM person WHERE persno = ?“);
```

- Werte eintragen

```
stmt.setInt(0, 40);
```

- Query ausführen

```
stmt.executeQuery();
```

Transaktionen

- JDBC-Connections verwenden standardmäßig `AUTO_COMMIT`
- Transaktionen mit JDBC

```
c.setAutoCommit(false);
Statment stmt = c.createStatement(
                    „UPDATE person SET fname = „Franz“ WHERE persno = 70“);
stmt.ExecuteUpdate();
//do some more stuff here
c.commit();
c.setAutoCommit(true);
```

- Savepoints

```
Savepoint beforeUpdate = con.setSavepoint(„BeforeUpdate“);
// do some stuff here
c.rollback(beforeUpdate);
```

```
Connection con = DriverManager.getConnection(
                    "jdbc:mysql://localhost:3306",
                    "myLogin",
                    "myPassword");

Statement stmt = con.createStatement();

ResultSet rs= stmt.executeQuery("SELECT persno, fname, lname FROM person");

while (rs.next()) {

     Person p = new Person();
     p.setPersno(rs.getInt(“persno"));
     p.setFname(rs.getString(“fname”));
     p.setLname(rs.getString(“lname”));

     System.out.println(p.toString());

}
```

Datenbankverbindung aufbauen

Statement erstellen und Query abschicken...

… Ergebnisse durchlaufen und POJO befüllen