

Molekularbiologische Datenbanken

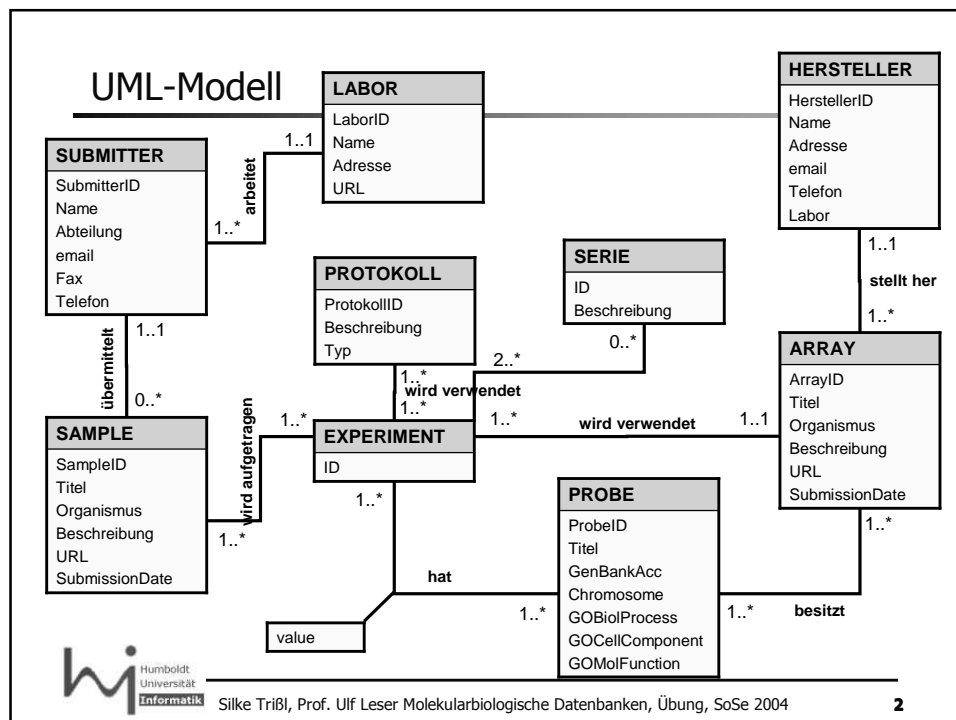
Übungen
Einleitung 3



Silke Trißl
Prof. Ulf Leser



Wissensmanagement in der
Bioinformatik



Relationale Datenbank

- Zusammenhänge sichtbar machen
 - Welcher Submitter verwendet welche Protokolle?
 - Welche Proben haben in unterschiedlichen Versuchen extrem abweichende Werte?
 - Wie viele Proben hat ein Array?
 - Gibt es Proben, die auch auf mehreren Arrays sind?

⇒ SQL – Anfragen formulieren

Dataload – CSV files

- Plattform
 - Array-Submitter.csv
 - Angaben über Array Submitter
 - Array.csv
 - Alle bis jetzt genutzten Arrays, wobei die ID = GPLxx ist.
 - GPLxx.csv
 - Jede Datei steht für einen Array. In diesen Dateien ist angegeben, welche
 - Proben auf dem Array sind und noch die zusätzlichen Daten über
 - GenBank-Accession und GeneOntology.

Dataload – CSV files – 2 –

- Samples
 - Organisation.csv
 - Daten über die beteiligten Labors
 - Submitter.csv
 - Daten über den Submitter von Samples und Meßergebnissen
 - Sample.csv
 - Daten über die Samples, wobei ID = GSMxxxx.csv ist.

Dataload – CSV files – 3 –

- Samples
 - Sample-Protokoll.csv
 - Welche Probe ist welchem Protokoll behandelt worden
 - GSMxxxx.csv
 - Jede Datei steht für ein Sample. In diesen Dateien sind die ID der Probe
 - und die Meßwerte für jede Probe auf dem Array angegeben.

Dataload – CSV files – 4 –

- Protokolle
 - Protokolle.csv
 - Daten über Protokoll-ID und die dazugehörigen Angaben.
- Aufgabe:
 - Laden der Daten (verfügbar über web-page)
 - SQL-Queries

PostgreSQL 7.4

- Download von postgresql-7.4.2.tar.gz über <http://www.postgresql.org/>
- Installation unter Linux (schwieriger bei Windows)
 - Installationsanleitung im WWW

Arbeiten auf dem WBI-Rechner

- Lokale Installation
 - jeder auf dem eigenen Rechner
- Möglichkeit, die Datenbank auf paprika, einem Rechner bei WBI zu nutzen
 - Zugang nur von den Rechnerpools der Informatik
 - Login auf paprika.informatik.hu-berlin.de nur mit Informatiklogin möglich
 - e-mail an Herrn Mielke (mielke@informatik.hu-berlin.de)
 - mit Vor- und Nachname und Benutzername

Arbeiten mit PostgreSQL

- Shell
 - > `psql -d <db_name> -U <user_name>`
- AquaDataStudio von AquaFold
 - kann sich mit PostgreSQL, Oracle, DB2, MySQL, ... verbinden
 - Verfügbar unter www.aquafold.com
 - oder auf paprika unter `/local/software/datstudio/datstudio.sh`
 - Anleitung im WWW

SQL

```
CREATE TABLE test
(t_id int NOT NULL,
 some_text varchar,
 foreign_id int NOT NULL,
 CONSTRAINT t_id PRIMARY KEY (subid));
```

```
ALTER TABLE test
ADD FOREIGN KEY (foreign_id)
REFERENCES foreign (id);
```

SQL – 2 –

```
SELECT count(*) FROM array
WHERE array_id LIKE '%34%';
```

```
INSERT INTO protokoll (prot_id, des)
VALUES ('EMBL12', 'some text');
```

```
DELETE FROM submitter
WHERE name ~* 'smith';
```

PostgreSQL Eigenheiten

- COPY
 - Möglichkeit, große Datenmengen in die Datenbank zu bringen
 - Tab (Komma, ...) delimited file
 - > **COPY** <table_name> **FROM** '<file>';

Meta-commands

- \h help
- \dT list data types
- \i <file> read and execute queries from <file>
- \d{t|i|s|v} list tables/indices/sequences/views
- \d <table> describe table (or index, ...)
- \q quit psql

User defined functions

- Funktionen werden in der Sprache plpgsql geschrieben
 - PostgreSQL eigene Sprache
 - kann externe Programme ersetzen (Java, Perl, ...)
 - Alles wird innerhalb der Datenbank abgearbeitet

```
CREATE FUNCTION max(int, int) returns int AS '  
DECLARE  
  a ALIAS FOR $1;  
  b ALIAS FOR $2;  
BEGIN  
  IF a > b THEN  
    RETURN a;  
  ELSE  
    RETURN b;  
  END IF;  
END  
' LANGUAGE 'plpgsql';
```

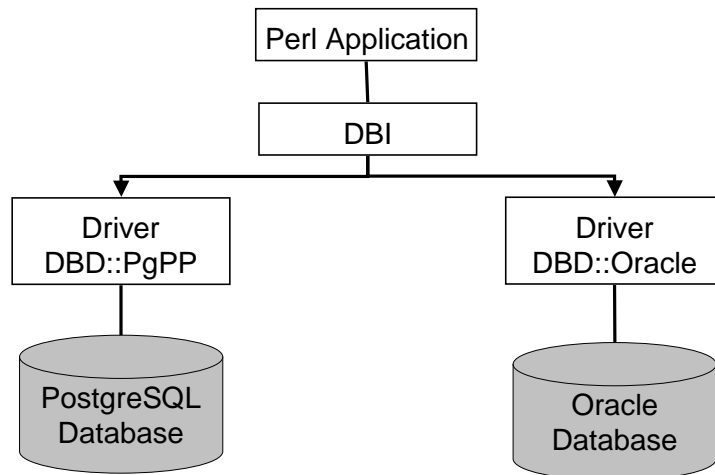


Eigenheit von PostgreSQL

- Bei Verwendung von Anführungszeichen ("):
 - CREATE TABLE "SOME" ("S_Id" int, ...
 - SELECT "SOME". "S_Id", ... FROM "SOME", ...
- Besser:
 - CREATE TABLE SOME (S_Id int, ...
 - SELECT SOME.S_Id, ... FROM SOME, ... oder
 - SELECT some.s_id, ... FROM some, ...



Perl - PostgreSQLPerlExample.pl



connect to database

```
my $dbname = "mdb";  
my $user = "u_mdb";  
my $password = "mdb_user";  
  
my $dbh = DBI-  
>connect("dbi:PgPP:dbname=$dbname", $user, $password)  
        or die "Can not connect to database $DBI::errstr  
($DBI::err)\n";  
$dbh->disconnect();
```

query database with query \$q

```
my $sth = $dbh->prepare( $q )  
        or die "Could not prepare statement: " . $dbh->errstr;  
  
$sth->execute( )  
        or die "Could not execute statement: " . $sth->errstr;
```

```

# print results of $sth

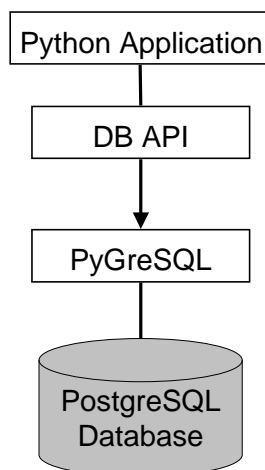
my $field_count = $sth->{NUM_OF_FIELDS};
my $names = $sth->{NAME};

# header
for (my $col = 0; $col < $field_count; $col++ ) {
    print "$names->[$col] | \t";
}

# result set
print "\n";
while( my $vals = $sth->fetchrow_arrayref() ) {
    foreach my $val ( @$vals ) {
        print $val." |\t";
    }
    print "\n";
}

```

Python - PostgreSQLPythonExample.py



```
# connect to database
```

```
try:  
    connection = _pg.connect( dbname= 'mdb',  
                               user = 'u_mdb', passwd= 'mdb_user' )  
except Exception, e:  
    print str( e )  
    exit  
connection.close()  
exit
```

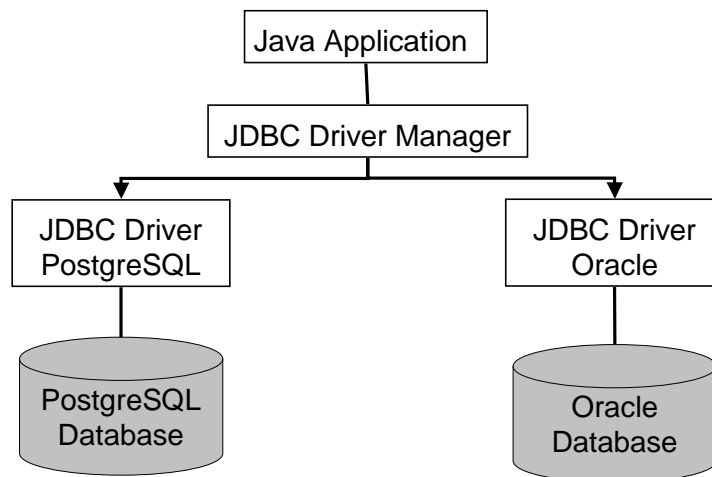
```
# query database with query
```

```
try:  
    print "EXECUTE QUERY ..."  
    cur = connection.query( "SELECT * FROM test" )  
  
except StandardError, e:  
    print str( e )
```

```
# print results of cur
```

```
for row in cur.getresult():  
  
    for col in row:  
        print str(col) + " |\t",  
    print
```

Java - PostgreSQLJavaExample.java



connect to database

```
String database = "mdb";
String driver = "org.postgresql.Driver";
String url = "jdbc:postgresql:" + database;
String user = "u_mdb";
String pwd = "mdb_user";

Class.forName(driver);
Connection con = DriverManager.getConnection(url, user, pwd);

System.out.println("Connection complete");
```

query database with query

```
try {
    Statement stmt = con.createStatement();
    //System.out.println(query);
    return (stmt.executeQuery(query));
} catch (SQLException e) {
    System.out.println("Query failed - " + e.getMessage());
    return (null);
}
```

```

// print results of query
try {
    ResultSetMetaData rsmd = res.getMetaData();
    int colCount = rsmd.getColumnCount();

    // drucken der Kopfzeile
    for (int i = 1; i <= colCount; i++) {
        System.out.print(rsmd.getColumnLabel(i) + " | \t");
    }
    System.out.println();

    while (res.next()) {
        for (int i=1; i<=colCount; i++){
            System.out.print(res.getString(i) + " | \t");
        }
        System.out.println();
    }
}
catch (SQLException e) {
    System.out.println("Fetch Failed: " + e.getMessage());
}

```



Literatur

- PostgreSQL-Homepage:
 - <http://www.postgresql.org/docs/>
 - Administrators Guide
 - Users Guide
 - Programmers Guide
- Bücher:
 - Douglas & Douglas: PostgreSQL, Developers Library 2003



Fragen?

- Aufgabe 3 bei Goya oder auf der Web-page
- Daten auf der Web-page
- Lösung der SQL-Queries bis 25.05., 17 Uhr

