Beleg Praktikum

Igor Geiger
Hochschule Zittau/Görlitz
FB Informatik
IIb07

31. August 2010

Inhaltsverzeichnis

1 Aufgabenstellung **3**

2 Das Geschäftsprozessmodell **3**

3 Modellierung der USE-Cases **5**

3.1 USE-Cases für den Client Ticketverkauf 5

3.2 USE-Cases für den Client Produktverkauf 5

3.3 USE-Cases für den Client Administrator 6

4 Das Datenbankschema der Datenbank auf dem Server **7**

4.1 SQL-Anweisungen des Clients Ticketverkauf 14

4.2 SQL-Anweisungen des Clients Produktverkauf 15

5 Die GUI des Clients Administrator **16**

5.1 Verwalten der Kinos 17

5.2 Verwalten von Sälen 17

5.3 Verwalten von Filmen 18

5.4 Verwalten von Vorführungen 19

5.5 Verwalten von Tickets 20

5.6 Verwalten von Produkten 21

6 Die GUI des Clients Ticketverkauf **22**

7 Die GUI des Clients Produktverkauf **22**

8 Tabellendarstellung der Daten **24**

9 Zusammenfassung **25**

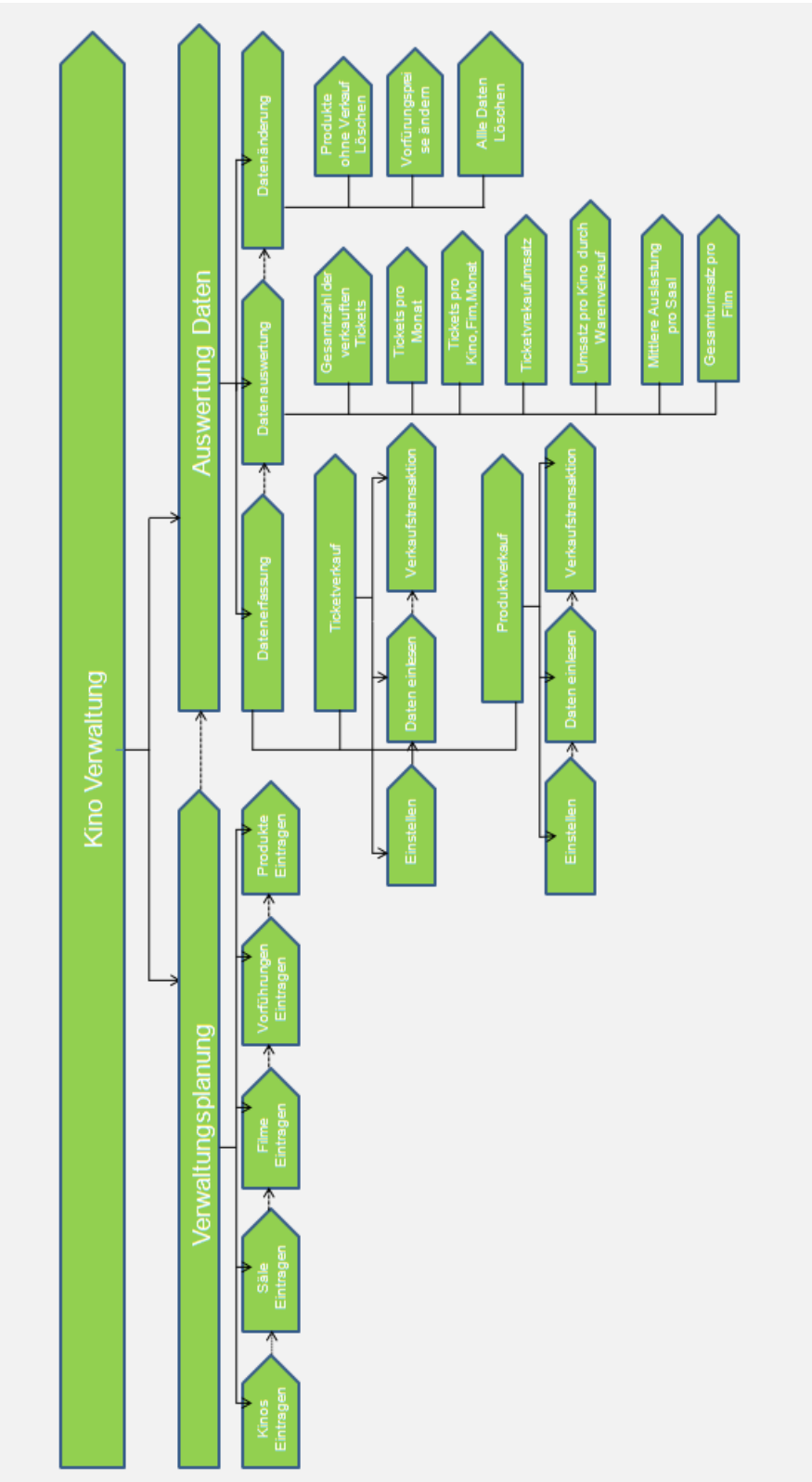
10 Anlagen **26**

1 Aufgabenstellung

Die Aufgabe des Praktikums besteht darin eine Client-Server Aplikation auf Basis eines Datenbankschemas zu entwickeln und in einem Wiki zu Dokumentieren,diese soll einem realen Geschäftsprozess entsprechen.Das verwendete Datenbankschema ,sollte dem Schema aus einer Klausur im Fachbereich Datenbanken 1 von Prof. Klaus ten Hagen entsprechen.Das von mir umgesetze Schema entspricht daher der Klausur vom Sommersemester 2009.Es ging in dem Schema darum,eine Datenverwaltung anhand einer Datenbank,für eine Kinokette zu realisieren.Desweiteren gehörte zur Klausur auch die Implementation von, für den allgemeinen Geschäftsprozess notwendigen SQL-Anweisungen,die in der Aplikation als Bestandtteile der Anwendungsfälle des Geschäftsprozesses realisirt werden sollten.Die realisierte Applikation besteht aus drei Clients die unabhängig voneinander auf eine Server-Datenbank des Apache-Derby Servers zugreifen.Die verwendete Programmiersprache ist Java und die dazugehörige Java-SQL API.Es gibt einen Client der den Ticketverkauf als Geschäftsprozess realisiert und einen Client der den Produktverkauf eines Kinos realisiert.Desweiteren einen Client zum Administrieren der gesamten Datenbank der Kinokette,er realisert die Gesamte Datenverwaltung und Auswertung für die Kinokette.Jeweils ein Kino besitzt daher für die Umsetzung seines Geschäftsprozesses ,den Client für den Ticketverkauf und den Client für den Produktverkauf in einfacher oder mehrfacher Anzahl aber mit den gleichen Netzwerk und Verkaufseinstellungen.Das Praktikum wurde von Prof. Klaus ten Hagen betreut.

2 Das Geschäftsprozessmodell

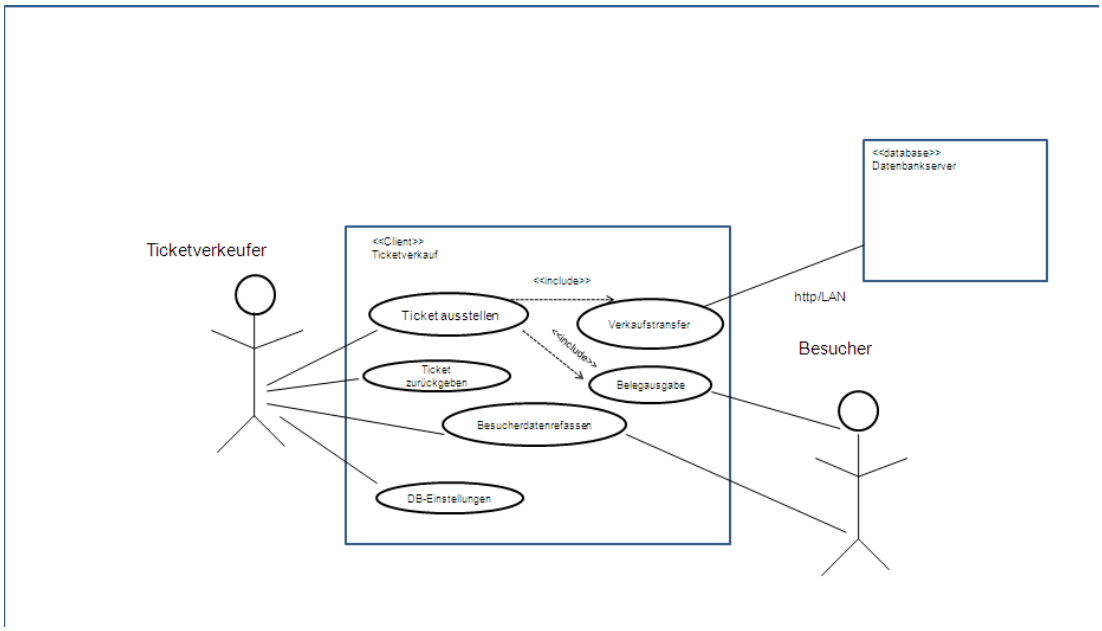
Das Geschäftsprozessmodell verdeutlicht den gesamten Geschäftsprozess einer Kinokette als Datenbankverwaltung.Dieser besteht aus der Verwaltungsplanung und dem Auswerten der Daten .Zur Verwaltungsplanung gehört es die bestehenden Kinodaten erstmal nacheinader mit gegenseitigem Bezug einzutragen ,in die angelegte Datenbank.Zur Auswertung der Daten gehören die Prozesse der Datenerfassung,Datenauswertung und Datenänderung .Die Datenerfassung erfolgt durch die Clients Ticketverkauf und Produktverkauf Parallel .Die Datenauswertung erfolgt anhand der erfassten Daten und liefert dem Administrator bzw. dem Manager Daten als Anschaugsobjekte in Tabellenform.Die Datenänderung erfolgt optional für die vom Administrator oder Manager ausgewählten und erfassten Datenobjekte.


Kino Verwaltung
Verwaltungsplanung
Kinos Eintragen
Säle Eintragen
Filme Eintragen
Vorführungen Eintragen
Produkte Eintragen
Auswertung Daten
Datenerfassung
Einstellen
Daten einlesen
Ticketverkauf
Verkaufstransaktion
Einstellen
Daten einlesen
Produktverkauf
Verkaufstransaktion
Datenauswertung
Datenänderung
Produkte ohne Verkauf Löschen
Vorführungspreise ändern
Alle Daten Löschen
Gesamtzahl der verkauften Tickets
Tickets pro Monat
Tickets pro Kino,Film,Monat
Ticketverkaufumsatz
Umsatz pro Kino durch Warenverkauf
Mittlere Auslastung pro Saal
Gesamtumsatz pro Film

3 Modellierung der USE-Cases

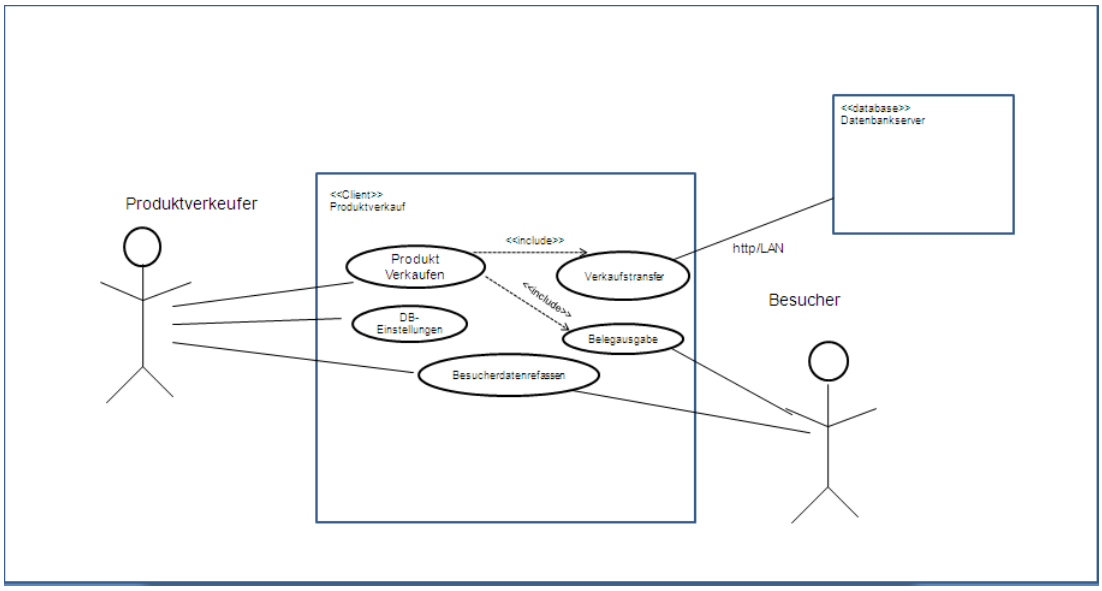
3.1 USE-Cases für den Client Ticketverkauf

Der Ticketverkauf umfasst das Erfassen der Besucherdaten den Verbindunsaufbau zum Datenbankserver,das Ausstellen der Tickets durch den Verkeufer und den Verkaufstransfer der Daten zum Datenbankserver ,wobei nebenbei eine Belegausgabe der Tickets für den Besucher erfolgt.Desweiteren ermöglicht der Client eine Rückgabe der Tickets ,anhand erfasster Besucherdaten.



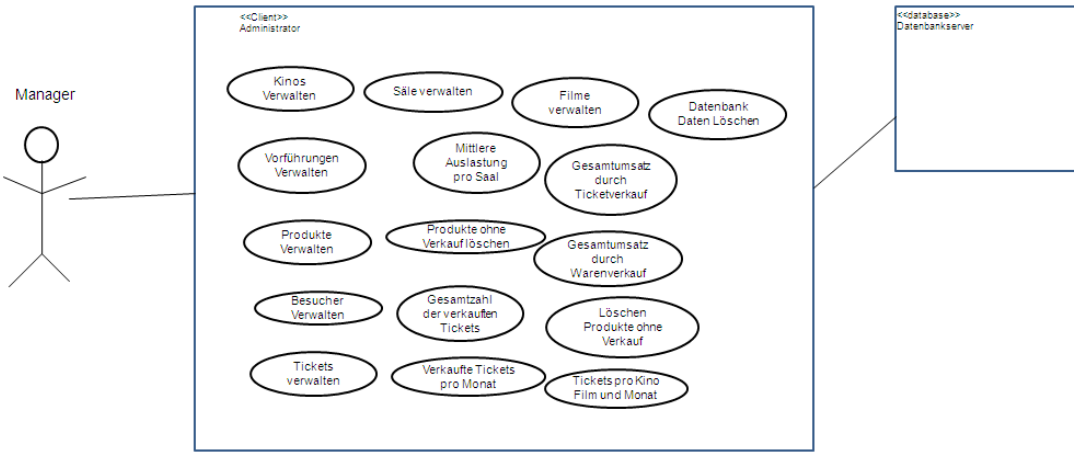
3.2 USE-Cases für den Client Produktverkauf

Der Produktverkauf umfasst das Erfassen der Besucherdaten und Datenbankeinstellungen zur jeweiligen Vorführung,durch den Produktverkeufer.Der Produktverkeufer initiiert den Produktverkauf an den Besucher und den Verkaufsdatentransfer an die Datenbank sowie die Belegausgabe an den jeweiligen Besucher.



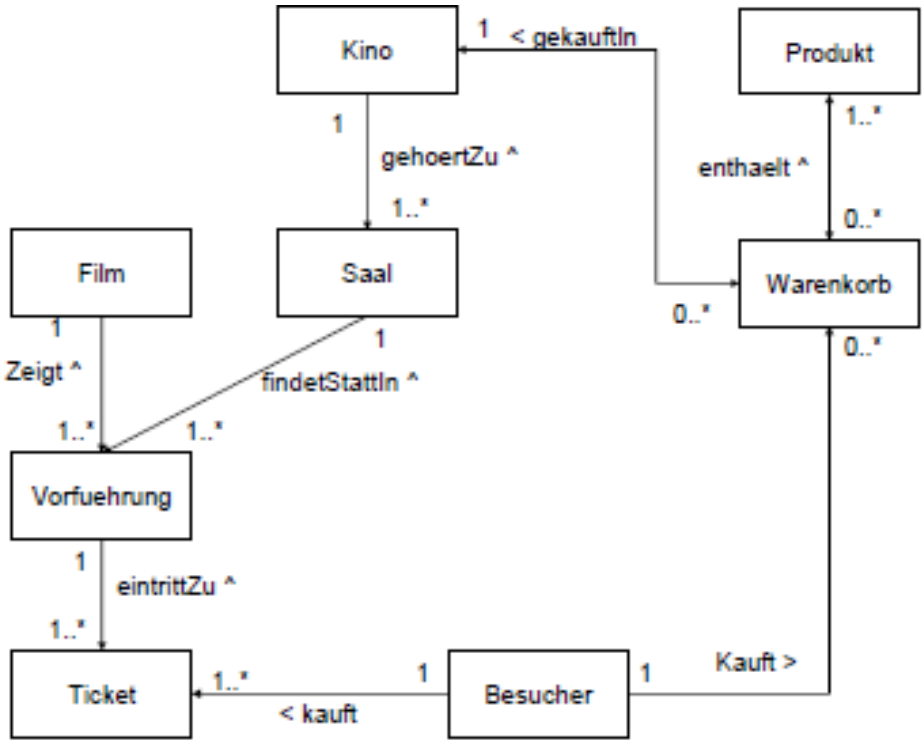
3.3 USE-Cases für den Client Administrator

Der Administrator Verwaltet und legt die Datenbank an.Er verwaltet alle dem Datenbankschema entsprechenden Datentabellen.Und wertet die Daten nach bestimmten Kriterien aus indem er sich ausgewälte Daten anzeigen lassen kann bzw. löschen kann.



4 Das Datenbankschema der Datenbank auf dem Server

Die Clients Administrator,Ticketverkauf und Produktverkauf bauen zur folgenden Datenbank eine Verbindung während des Geschäftsprozesses auf.



Folgende Verbindungseinstellungen werden dabei von den Clients verwendet:

```
static String driver = "org.apache.derby.jdbc.ClientDriver";

static  String dbName="Kino";

static  String connectionURL = "jdbc:derby://localhost:1527/
"+dbName+";create=true";
```

Das Datenbankschema besteht aus mehreren Tabellen,diese werden innerhalb des Clients Administrator, beim Aufruf erzeugt.Wenn aber schon eine Datenbank auf dem Server besteht,so wird diese nicht nochmal erzeugt.Sondern es wird die erzeugte Datenbank mit den gespeicherten Daten verwendet.Die SQL -Anweisungen zur erzeugung sind als Stringvariablen im Client Administrator folgendermassen definiert:

```
public static String CKino="CREATE TABLE Kino
(KID INTEGER PRIMARY KEY,Name VARCHAR(30) UNIQUE )";

public static String CSaal="CREATE TABLE Saal
( SID INTEGER PRIMARY KEY,Name VARCHAR(30)
UNIQUE,KID INTEGER REFERENCES Kino,Plaetze INTEGER )";

public static String CFilm="CREATE TABLE Film
(FID INTEGER PRIMARY KEY,Name VARCHAR(70) UNIQUE)";

public static String CVorfuehrung="CREATE TABLE Vorfuehrung
(VID INTEGER PRIMARY KEY,Zeit TIMESTAMP ,
SID INTEGER  REFERENCES Saal(SID),
FID INTEGER  REFERENCES Film(FID),
UNIQUE(Zeit,SID,FID),Preis DECIMAL(10,2) )";

public static String CBesucher="CREATE TABLE Besucher
(BID INTEGER PRIMARY KEY,Name VARCHAR(100) )";

public static String CTicket="CREATE TABLE Ticket
(TID INTEGER PRIMARY KEY, Nummer VARCHAR(200) ,
 VID INTEGER REFERENCES Vorfuehrung(VID),
 BID INTEGER REFERENCES Besucher(BID),
 Preis DECIMAL(10,2) )";

public static String CProdukt="CREATE TABLE Produkt
(PID INTEGER PRIMARY KEY, Name VARCHAR(30) UNIQUE,
Preis DECIMAL(10,2) )";

public static String CWarenkorb="CREATE TABLE Warenkorb
(WID INTEGER PRIMARY KEY, BID INTEGER REFERENCES Besucher(BID)
, KID INTEGER REFERENCES Kino(KID), Zeit TIMESTAMP)";

public static String CProdInKorb="CREATE TABLE ProdInKorb
(PID INTEGER REFERENCES Produkt(PID),
WID INTEGER REFERENCES Warenkorb(WID), Anzahl INTEGER, Preis DECIMAL(10,2))";
```

Zu jeder Tabelle definiert der Client Administrator alle notwendigen
SQL-Anweisungen innerhalb von Methoden.
Eine Anweisungsfolge zum Einfgen von Kinos:

```
String d="SELECT COUNT(KID) FROM KINO";
int kidi=0;
dateidk=s.executeQuery(d);


while (dateidk.next())
 {
kidi=dateidk.getInt(1);
kidi=kidi+1;
SetInKino(kidi,name);


}


s.close();
conn.close();
```

Löschen von Kinos mit einem bestimmten Namen:

```
s = conn.createStatement();
s.execute("DELETE FROM Kino WHERE Name=  "+"'"+name+"'";
```

Eine Anweisungsfolge zum Löschen der Verkaufsdaten :

```
conn = DriverManager.getConnection(connectionURL);
s = conn.createStatement();

s.execute("DELETE FROM ProdInKorb");
s.close();
conn.close();
conn = DriverManager.getConnection(connectionURL);
s = conn.createStatement();
s.execute("DELETE FROM Warenkorb");
s.close();
conn.close();
conn = DriverManager.getConnection(connectionURL);

s = conn.createStatement();
```

```
s.execute("DELETE FROM  Ticket");
s.close();
conn.close();

conn = DriverManager.getConnection(connectionURL);

s = conn.createStatement();

s.execute("DELETE FROM  Besucher");
s.close();
conn.close();
```

Der Umsatz zum Warenverkauf pro Kino:

```
Date t,end;

" SELECT Kino.Name, SUM(ProdInKorb.Anzahl * ProdInKorb.Preis)
AS Umsatz FROM Kino , Warenkorb , ProdInKorb WHERE Kino.KID
= Warenkorb.KID AND Warenkorb.WID = ProdInKorb.WID AND
DATE(Warenkorb.Zeit) BETWEEN "+"'"+t+"'"+" AND "+" '"+
end+"'"+" GROUP BY Kino.KID,Kino.Name";
```

Der Gesamtumsatz pro Film:

```
Date t,end;


 String sum=" SELECT Film.Name,(SELECT
 SUM(ProdInKorb.Anzahl * ProdInKorb.Preis) AS
 Umsatz FROM  Warenkorb , ProdInKorb,Film,Vorfuehrung
 WHERE Film.FID=Vorfuehrung.FID AND Vorfuehrung.Zeit=Warenkorb.Zeit
 AND Warenkorb.WID = ProdInKorb.WID AND DATE(Warenkorb.Zeit)
 BETWEEN "+"'"+t+"'"+" AND "+" '"+end+"'"+")+(SELECT SUM(Ticket.Preis)
  FROM  Ticket,Film,Vorfuehrung WHERE Film.FID=Vorfuehrung.FID
  AND Ticket.VID=Vorfuehrung.VID AND DATE(Vorfuehrung.Zeit)
  BETWEEN "+"'"+t+"'"+" AND "+" '"+end+"'"+" )FROM FILM  ";
```

Mittlere Auslastung pro Saal:

```
String d="SELECT Saal.Name , ( COUNT(Ticket.TID)*100 /
(Saal.Plaetze * COUNT(DISTINCT Vorfuehrung.VID))) AS
Auslastung FROM Saal , Vorfuehrung , Ticket WHERE Saal.SID
= Vorfuehrung.SID AND Vorfuehrung.VID = Ticket.VID AND
DATE(Vorfuehrung.Zeit) BETWEEN  "+"'"+t+"'"+" AND "+"
'"+end+"'"+" GROUP BY Saal.Name,Saal.SID,Saal.Plaetze
ORDER BY Auslastung DESC";
```

Saal mit einem bestimmten Namen Löschen:

```
s.execute("DELETE FROM Saal WHERE Name=  "+"'"+name+"'")
```

Einfügen eines Films:

```
String d="SELECT COUNT(FID) FROM Film";
int kidi=0;
dateidk=s.executeQuery(d);


while (dateidk.next())
{
kidi=dateidk.getInt(1);
kidi=kidi+1;
SetInFilm(kidi,name);


}
```

Löschen eines Films:

```
s.execute("DELETE FROM Film WHERE Name=  "+"'"+name+"'");
```

Einfügen einer Vorführung:

```
s = conn.createStatement();

String d="SELECT COUNT(VID) FROM Vorfuehrung";
int kidi=0;
dateidk=s.executeQuery(d);


while (dateidk.next())
{
kidi=dateidk.getInt(1);
kidi=kidi+1;
SetInVorf(kidi,Zeit,sid,fid,pr);


}
```

Update des Preises für die Vorführungen eines Films:

```
String d="UPDATE Vorfuehrung  SET Preis = "+quot+"
WHERE FID = (SELECT FID FROM Film WHERE FID = "+film+")";
```

Löschen einer bestimmten Vorführung:

```
s = conn.createStatement();
s.execute("DELETE FROM Vorfuehrung WHERE VID=  "+id);
s.close();
conn.close();
```

Verkaufte Tickets in einem bestimmten Zeitraum:

```
String d="SELECT COUNT(*)  FROM Ticket,Vorfuehrung
WHERE DATE(Vorfuehrung.Zeit) BETWEEN "+"'"+an+"'"+"
AND "+" '"+en+"'"+" AND Ticket.VID=Vorfuehrung.VID    ";
```

Verkaufte Tickets pro Monat:

```
String d=" SELECT MONTH(Zeit)as Monat,COUNT(TID)as
Tickets FROM  Vorfuehrung JOIN Ticket ON
Vorfuehrung.VID=Ticket.VID GROUP BY MONTH(Zeit) ";
```

Gesamtumsatz durch Ticketverkauf:

```
String d=" SELECT SUM(Ticket.Preis) FROM TICKET,
Vorfuehrung WHERE Ticket.VID=Vorfuehrung.VID AND
DATE(Vorfuehrung.Zeit) BETWEEN "+"'"+t+"'"+" AND "+" '"+end+"'";
```

Verkaufte Tickets pro Kino,Film und Monat:

```
String d=" SELECT Kino.Name,Film.Name,
MONTH(Vorfuehrung.Zeit)as Monat,COUNT(Ticket.TID)as
Tickets FROM Kino,Film,Saal,Vorfuehrung,Ticket WHERE
Vorfuehrung.SID=Saal.SID AND Saal.KID=Kino.KID AND
Vorfuehrung.FID=Film.FID AND Ticket.VID=Vorfuehrung.VID
GROUP BY Kino.Name,Film.Name,MONTH(Vorfuehrung.Zeit)
ORDER BY Kino.Name,Film.Name,MONTH(Vorfuehrung.Zeit)";
```

Einfügen eines Produktes:

```
String d="SELECT COUNT(PID) FROM Produkt";
int kidi=0;
dateidk=s.executeQuery(d);


while (dateidk.next())
{
kidi=dateidk.getInt(1);
kidi=kidi+1;
SetInProdkt(kidi,name,pr);


}
```

Löschen eines bestimmten Produktes:

```
s.execute("DELETE FROM Produkt WHERE Produkt.Name=  "+"'"+name+"'");
```

Löschen von Produkten ohne Verkauf:

```
s.execute("DELETE FROM Produkt  WHERE Produkt.PID
 NOT IN((Select ProdInKorb.PID From ProdInKorb))");
```

4.1 SQL-Anweisungen des Clients Ticketverkauf

Der Verkaufstransfer bei einem Ticketverkauf geschieht nachdem die Besucherdaten und die Ticketdaten erfasst worden sind.Es wird bei jedem Ticketkauf ,eines Besuchers zu einer Vorührung ,ein Warenkorb ,nebenbei angelegt.Und Inserts in die Tabellen Besucher,Ticket und Warenkorb gemacht,dazu wir aber vorher geprüft ob es genug freie Plätze im Saal gibt ,dazu wird die Ticketanzahl mit der bestimmten Anzahl der freien Platze im betroffenem Saal vergliechen,sollte es freie Plätze geben so wird ein Insert in die Besucher und Tickettabelle gemacht,wobei die Ticketnummer aus den Daten zum Ticketkauf generiert wird.

Folgende SQL-Anweisungen werden beim Ticketkauf verwendet:

```
String plat="SELECT Saal.Plaetze  FROM Film ,Vorfuehrung , Saal
, Kino  WHERE Film.FID = Vorfuehrung.FID AND Vorfuehrung.SID
= Saal.SID AND Saal.KID = Kino.KID AND  Kino.Name =
"+"'"+kin+"'"+" " +" AND " +" Film.Name = "+"'"+film+"'"+"
AND "+"Vorfuehrung.Zeit = "+"'"+vr+"'";

String d="SELECT COUNT(Ticket.TID)  FROM Film , Ticket
, Vorfuehrung , Saal , Kino  WHERE Film.FID = Vorfuehrung.FID
AND Vorfuehrung.SID = Saal.SID AND Saal.KID = Kino.KID
AND Vorfuehrung.VID = Ticket.VID   AND Kino.Name =
"+"'"+kin+"'"+" " +" AND " +" Film.Name = "+"'"+film+"'"+"
AND "+"Vorfuehrung.Zeit = "+"'"+vr+"'";

String bs="SELECT COUNT(*) FROM Besucher";

String vpr="SELECT Vorfuehrung.VID,Vorfuehrung.Preis
FROM  Vorfuehrung,Film,Saal,Kino WHERE Vorfuehrung.Zeit =
"+"'"+vr+"'" +  " AND "+"Vorfuehrung.FID=Film.FID AND
Vorfuehrung.SID=Saal.SID AND Film.Name= "+"'"+film+"'" +"
AND Saal.KID = Kino.KID AND Kino.Name = "+"'"+kin+"'";
String at="SELECT COUNT(*) FROM Ticket";
String kida="SELECT KID FROM Kino WHERE Name= "+"'"+kin+"'";
```

Bei der Rückgabe der Tickets werden die Tickets zum Besucher und Vorführung aus der Tickettabelle gelöscht.Dazu werden folgende SQL-Anweisungen verwendet:

```
String quir="SELECT WID FROM WARENKORB,BESUCHER "+"WHERE
Zeit = "+"'"+vr+"'"+" AND Besucher.BID=Warenkorb.BID
AND Besucher.Name= "+"'"+bn+"'";

String d="DELETE FROM Ticket WHERE
NUMMER ="+"'"+bn+"-"+kin+"-"+film+vr+"'";
```

4.2 SQL-Anweisungen des Clients Produktverkauf

Beim Produktverkauf wird jeweils in die Tabelle ProdInKorb eingetragen und eine Rechnung im Client Produktverkauf ausgegeben.Zu jedem Kunden wird beim Kauf eine Rechnung dynamisch erstellt und ausgegeben,am Ende wird immer die Gesamtsumme ausgegeben.

Dazu werden folgende SQL-Anweisungen verwendet:

```
String apid="SELECT PID FROM PRODUKT WHERE Name = "+"'"+prn+"'";

String awid="SELECT WID FROM WARENKORB,BESUCHER,Ticket,Vorfuehrung
WHERE WARENKORB.BID=BESUCHER.BID AND BESUCHER.NAME =
"+"'"+bn+"'"+" AND "+"Ticket.BID =Besucher.BID AND
Ticket.VID=Vorfuehrung.VID AND
Vorfuehrung.Zeit="+"'"+vz+"'";

String apr="SELECT Preis FROM PRODUKT WHERE Name = "+"'"+prn+"'";

//Rechnung

String sum="SELECT Produkt.Name,ProdInKorb.Anzahl,
ProdInKorb.Preis FROM Produkt, ProdInKorb,Warenkorb,
Besucher,Ticket,Vorfuehrung WHERE Produkt.PID=ProdInKorb.PID
AND ProdInKorb.WID=Warenkorb.WID AND Warenkorb.BID=Ticket.BID
AND Ticket.VID=Vorfuehrung.VID AND Vorfuehrung.Zeit=Warenkorb.Zeit
AND Besucher.Name = "+"'"+bn+"'"+" AND Warenkorb.wid= "+wk ;
```

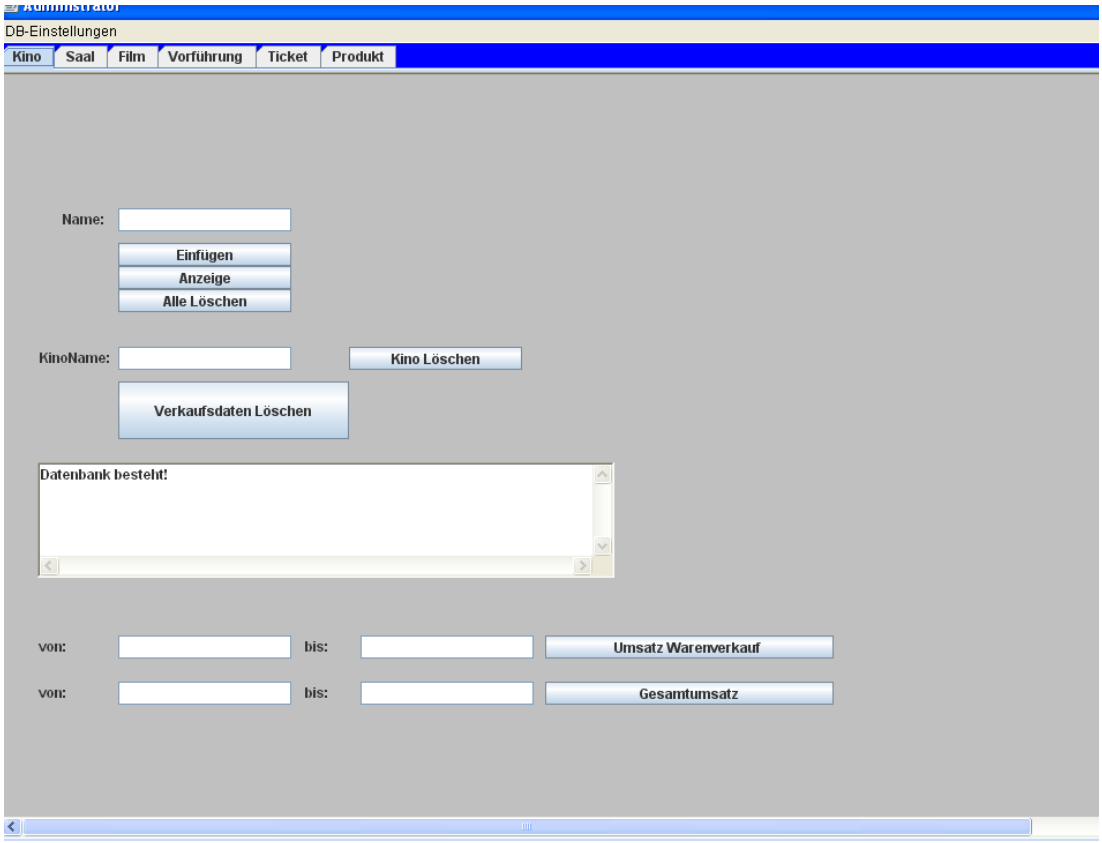
```
Datenpr=s.executeQuery(sum);
int Anzahl;
double prsi;
String nami;
double waren;
while (Datenpr.next())
       {
nami=Datenpr.getString(1);
Anzahl=Datenpr.getInt(2);
prsi=Datenpr.getDouble(3);
waren=prsi*Anzahl;
tex.setText(nami+"--"+Anzahl+"--"+prsi+"--"+waren+'\n');



       }
```

5 Die GUI des Clients Administrator

Die Graphische Benuzeroberfläche des Clients Administrator ermöglicht es dem Manager oder dem Administrator der Kinokette alle Verkaufsdaten der einzelnen Kinos zu verwalten.Zur Verwaltung der Daten sind nur die Tabellen Kino,Film,Vorführung,Ticket,Saal interessant da nur diese Tabellen direkt vom Administrator bei der Verwaltung angesprochen und beschrieben werden.Weitere Tabellen dienen nur zur Speicherung der Zwischenergebnisse.Zur GUI des Clients Administrator gehören die Klassen Administrator,AdminTable,EinstAdmin,WwdUtils.Die Klasse Administrator Enthält die eigentliche Benutzeroberfläche.Diese besteht aus einzelnen TabedPane's zu den einzelnen Tabellen.Zu jedem Tabed Pane gibt es ein Panel auf dem die einzelnen Bedienelemente wie Buttons und Textfelder angeordnet sind. Auf die Klasse AdminTable wird zugegriffen um die Daten der Tabellen darzustellen oder um einzelne Werte aus diesen sichtbar auszuwählen durch linksklick der Maustaste.Zu Begin des Geschäftsprzesses werden die Tabellen Kino,Saal,Film,Vorführung mit Daten nacheinander beschrieben ,da z.Bsp. das Beschreiben der Tabelle Vorführung nur durch Auswahl der Daten aus den Tabellen Saal und Film uber die Methoden der Klasse AdminTable geht.Vor jeder Bedienung muss aber einmalig die IP des Datenbankservers im Menüpunkt Einstellungen gesetzt werden.

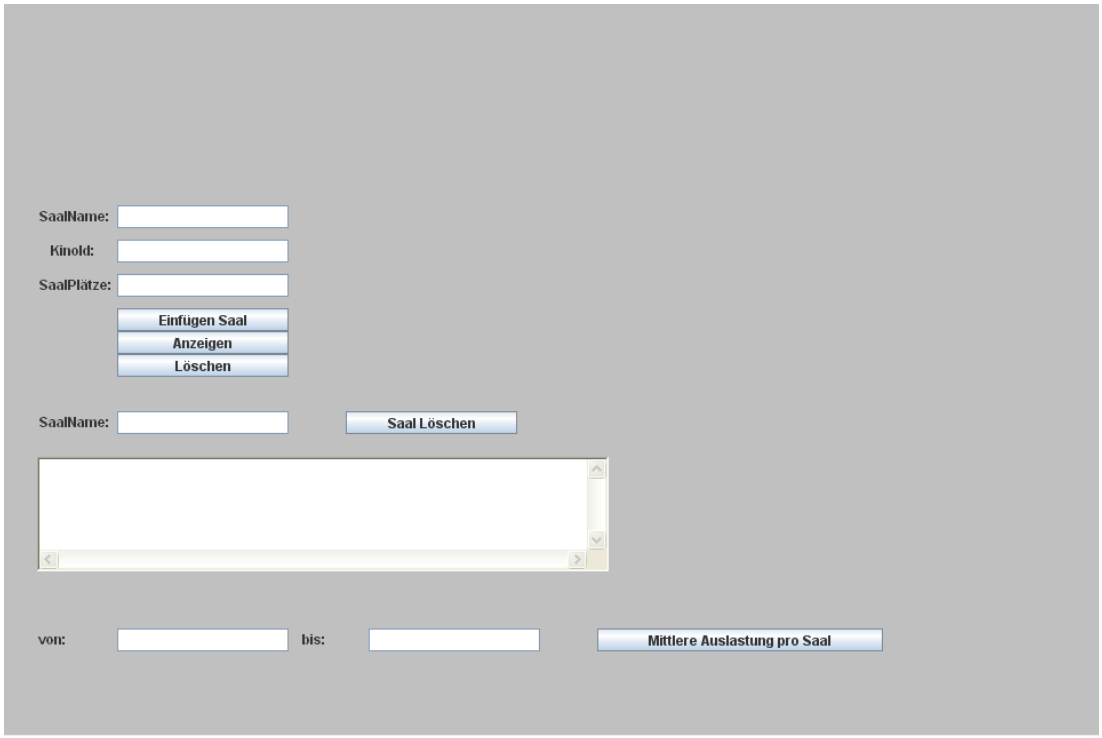
5.1 Verwalten der Kinos

Kinos können eingefügt und gelöscht werden.Zum Einfügen wird der Name in ein Textfeld eingetragen,danach wird der Button Einfügen gedrückt.Das Löschen funktioniert analog. Weitere Buttons führen die auf ihnen stehenden Funktionen aus.Diverse Fehlermeldungen werden im groen Textfeld angezeigt.



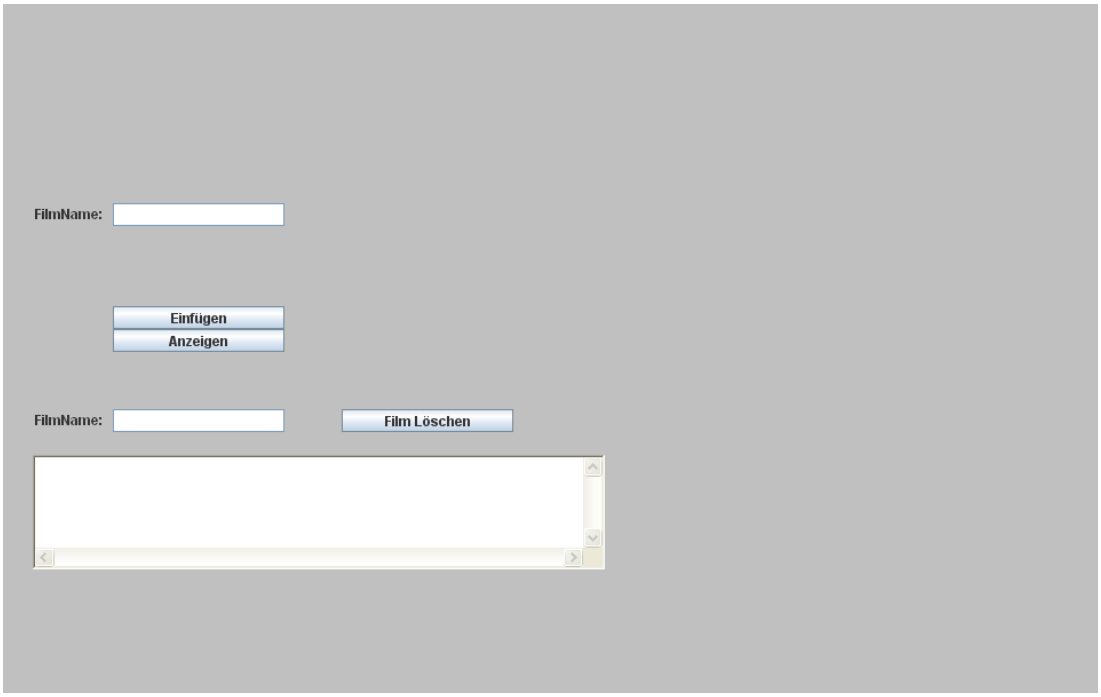
5.2 Verwalten von Sälen

Die Attribute eines Saal werden nacheinander eingetragen in die Textfelder danach wird das Button mir dem aufgeschriebenem Befehl gedrückt.In die Textfelder für Datumswerte kommen nur Datewerte ohne Zeitangabe rein.



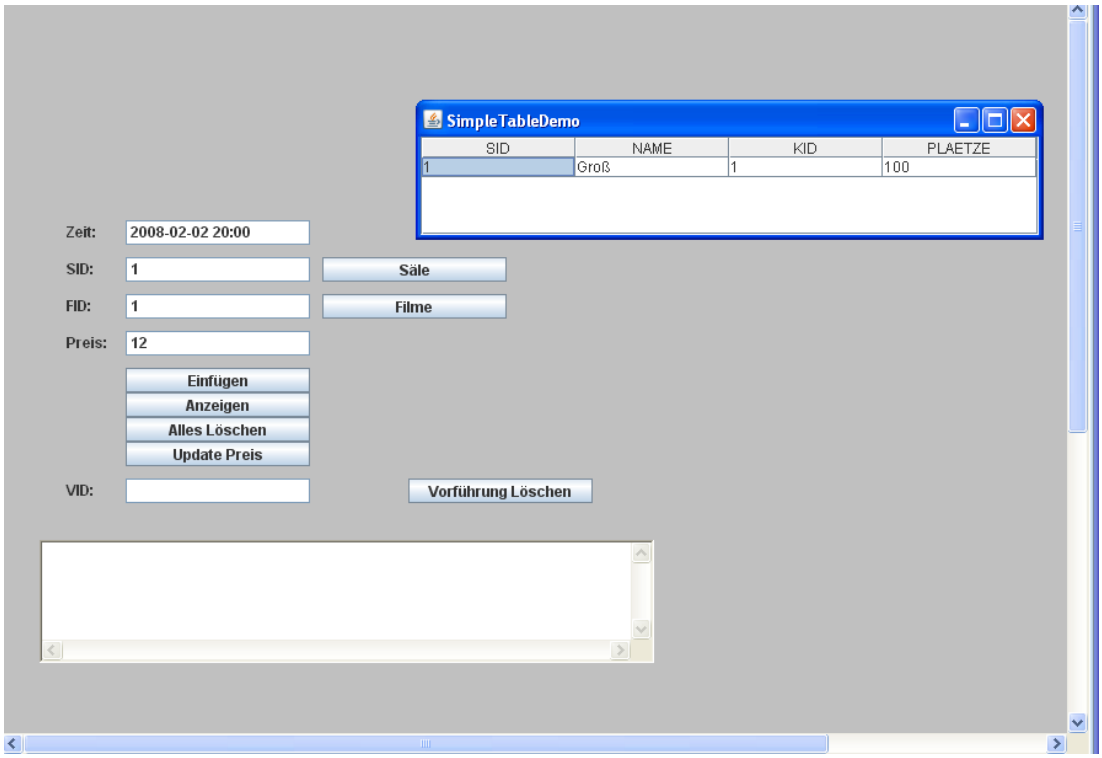
5.3 Verwalten von Filmen

Das Verwalten von Filmen benötigt nur den Eintrag eines Filmnamens in die benötigten Textfelder.



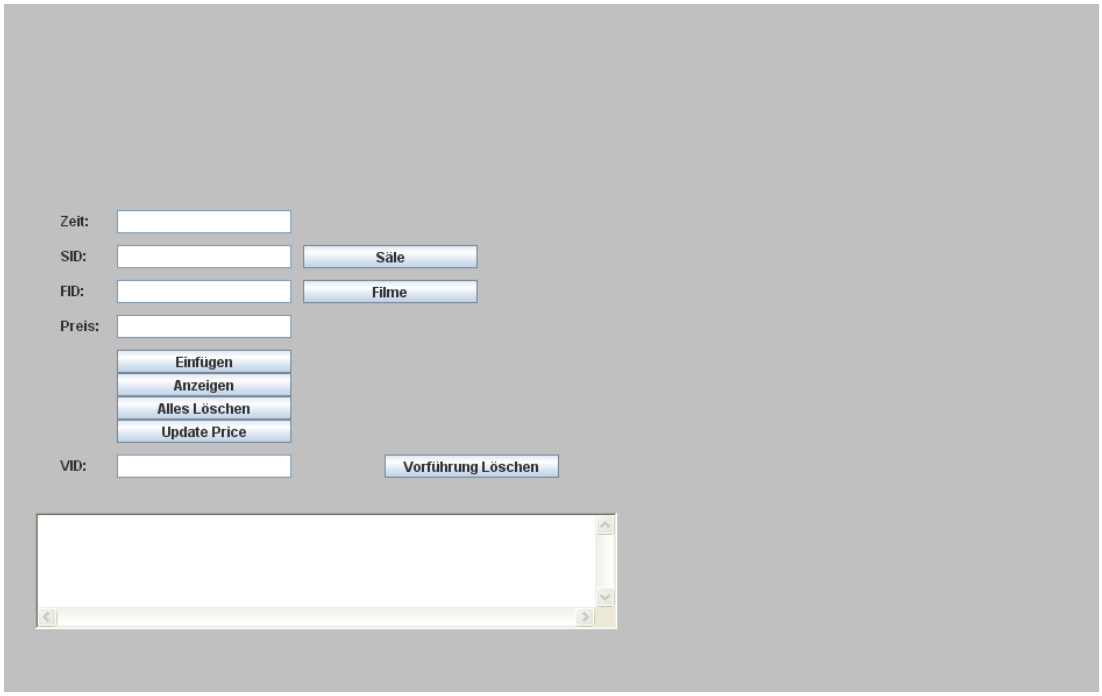
5.4 Verwalten von Vorführungen

Zum Verwalten von Vorführungen sind Eintragungen in die beschriebenen Textfelder ötig.Über die Buttons Säle werden die SID's per Mausklick aus der geöffneten Tabelle gewählt,genau so wie die FID's mit dem Button Filme.Zum Update der Preise für die Vorführungen eines Film ist die SID,FID und die neue aktuelle Preisnagabe notwendig.



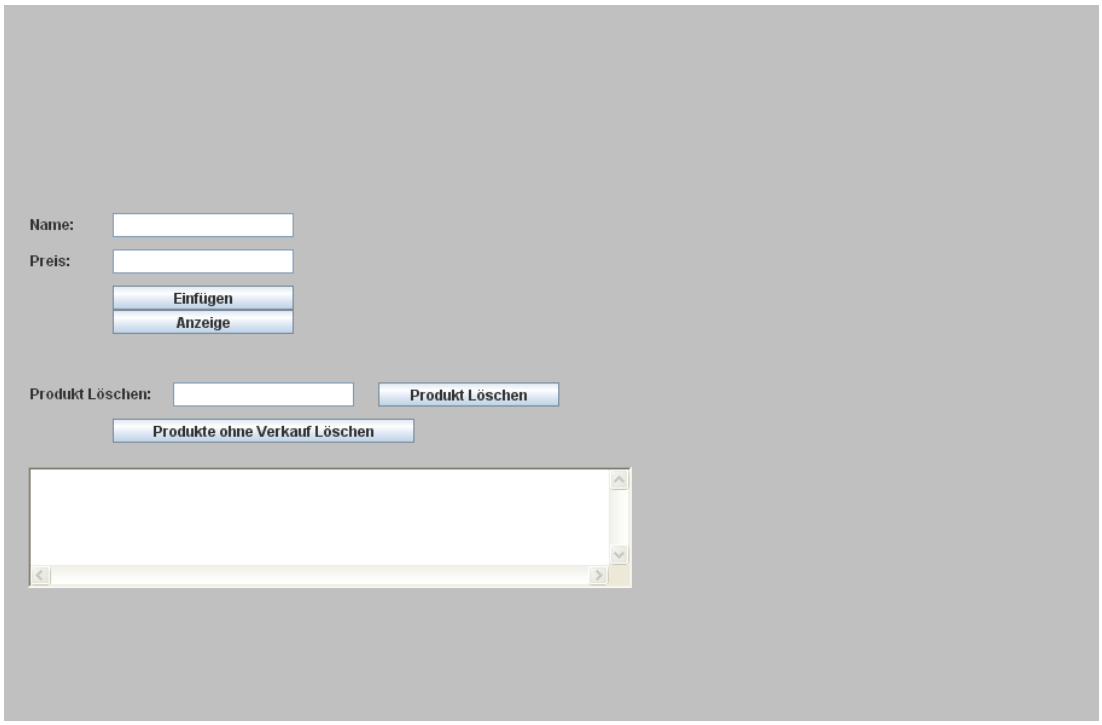
5.5 Verwalten von Tickets

Die Verkauften Tickets in einem bestimmten Zeitraum können mittels Button Anzeige angezeigt werden.Und der Gesamtumsatz des Ticketverkaufs kann mit dem Button Gesamtumsatz zu einem bestimmte Zeitraum angezeigt werden.



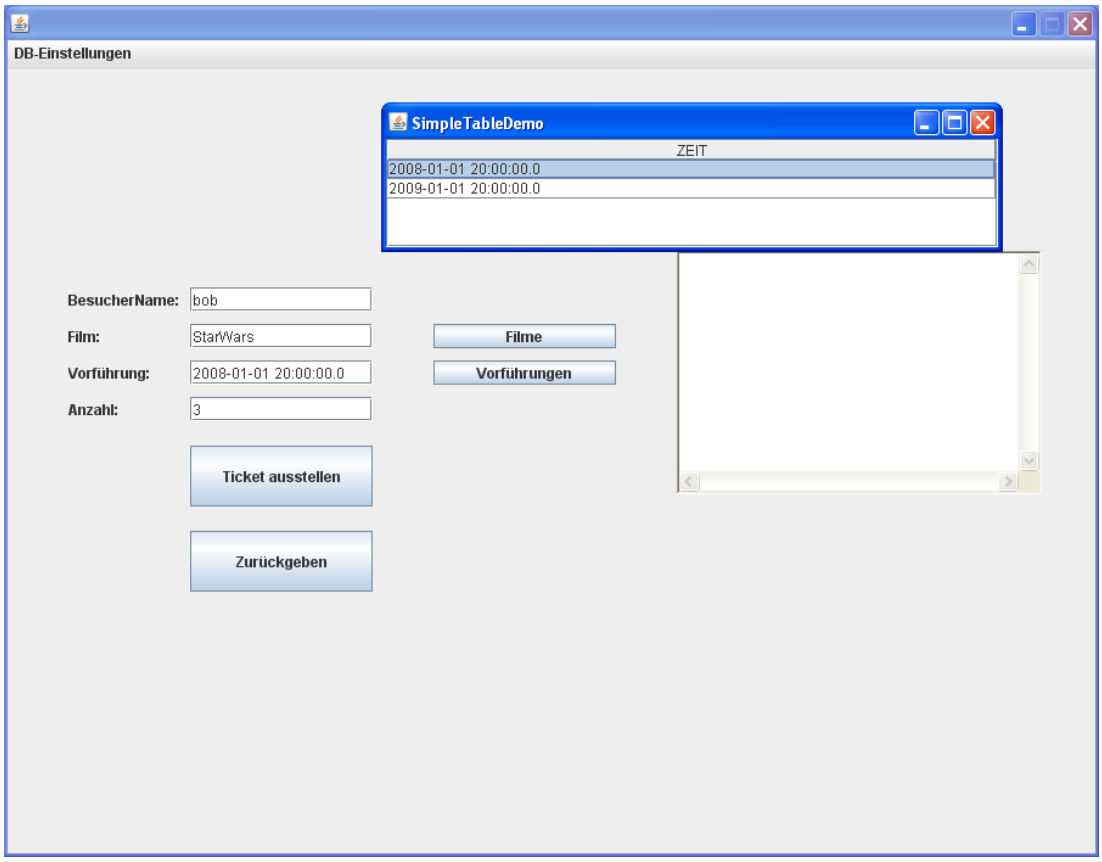
5.6 Verwalten von Produkten

Produkte können angezeigt,gelöscht und eingefügt werden.Zum Löschen ist die Eingabe des Namens notwendig.



6 Die GUI des Clients Ticketverkauf

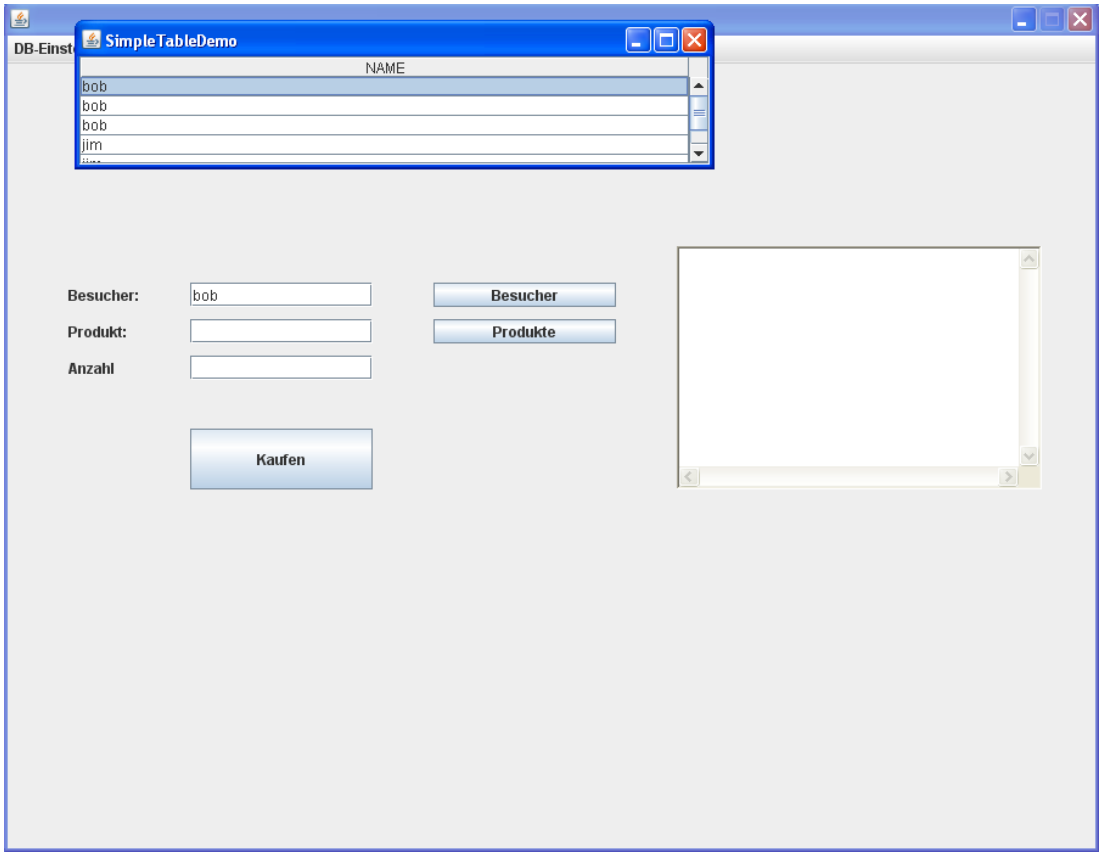
Zur GUI Des Clients Ticketverkauf gehören die Klassen Ticketverkauf,StartTV,EinstTV und TicketTable.Mit der Klasse Ticketverkauf ruft der der Benutzer alle anderen Klassen auf ,fals erforderlich.Zu Beginn des Ticketverkaufs stellt der Benutzer im Menüpunkt DB-Einstellungen die Verbindungseinstellungen zum Server und seinen Kinonamen ein.Anhand folgender Benutzeroberfläche werden die Daten des Besuchers erfasst ,die Tickets ausgestellt oder zurückgegeben,beim Zurückgeben ist die Angabe der Ticketanzahl unnötig. Die Klasse Tickettable dient zum Darstellen der für den Verkauf notwendigen Ticketdaten.Die Klasse StartTV dient zum Starten der GUI.



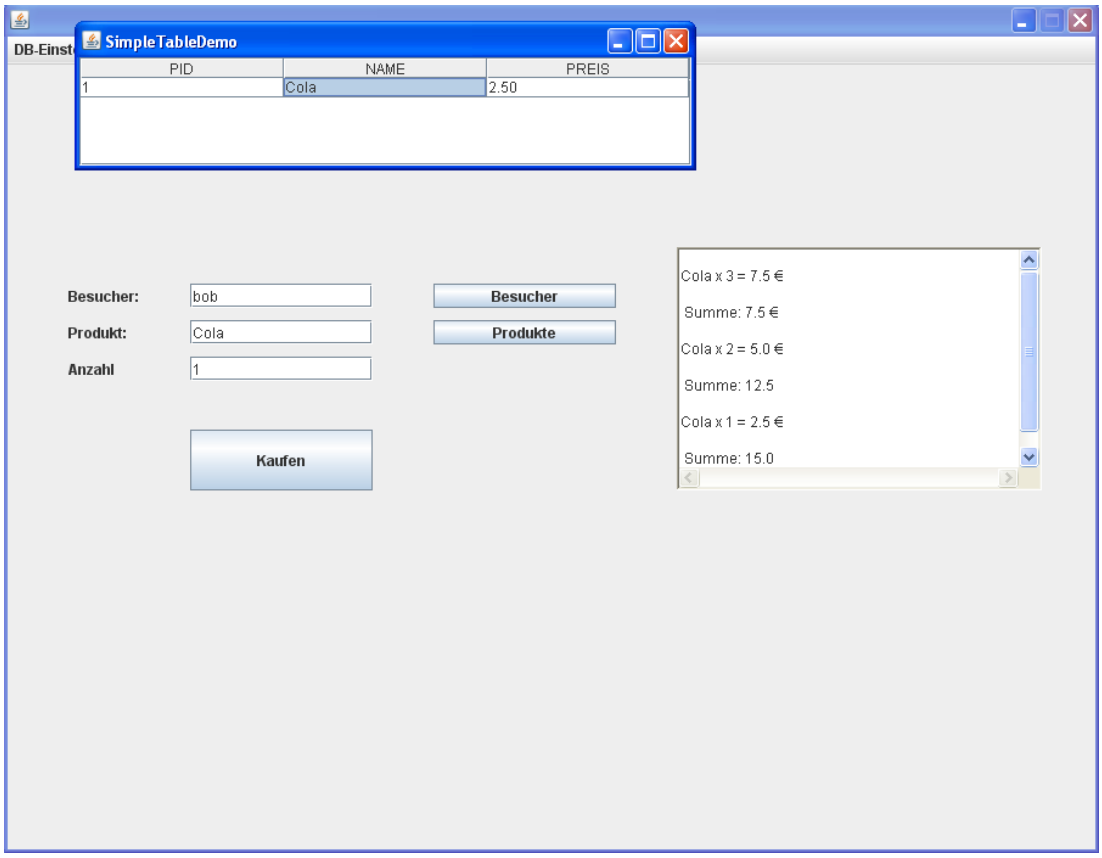
7 Die GUI des Clients Produktverkauf

Zur GUI des Clients Produktverkauf gehören die Klassen Produktverkauf,ProduktTable,StartPV,TicketEinst.Mit der Klasse Produktverkauf ruft der Benutzer alle anderen Klassen fals erforderlich auf.Die

Klasse StartPV dient zum Starten der GUI.Ist die Benuzeroberfläche sichtbar so werden im Menüpunkt DB-Einstellungen alle notwendigen Einstellungen gemacht wie die Enstellung der aktuellen Vorführung und des Kinonamens.Beim Produktverkauf wird der Name des Besuchers der die aktuellen Tickets vorweist erfasst,um den Produktkauf für diesen zu machen.Die Produkte werden danach ausgewält anhand der GUI von der Klasse ProduktTable und damit,mit dem Klicken der Linken Maustaste eingetragen.Beim Produktkauf wird nebenbei die aktuelle Rechnung im Textfeld erstellt.



Darstellung eines Verkaufs an den Kunden bob mit seiner Rechnung als Ausgabe:



8 Tabellendarstellung der Daten

Alle Cleints verwenden eine Tabellendarstellung der Daten zur Ansicht oder zum Auswählen bestimmter Objekte.Diese Tabellendarstellung beruht bei allem auf ein und demselben Prinzip.Das Zugreifen auf Zellen einer Graphischen Tabelle wird ermöglicht und der MouseListener der JTable dieser Klassen wird aktiviert.Somit kann auf die Inhalte der Zellen durch Mausklicks zugegriffen werden und eine Auswahl der Inhalte als Stringobjekte erfolgen.Ein Ausschnitt der Klasse ProduktTable verdeutlicht dieses Prinzip.

```
final JTable table = new JTable(dat, columnNames);
table.setCellSelectionEnabled(true);
table.setPreferredScrollableViewportSize(new Dimension(500, 70));
table.setFillsViewportHeight(true);

table.addMouseListener(new MouseAdapter(){
public void mouseClicked(MouseEvent bm){
```

```
int u= table.getSelectedRow();
int p=table.getSelectedColumn();

System.out.println(u+"--"+p);

Object t=dat.get(u);
Object e=table.getValueAt(u,p);
String h=e.toString();

if(Tabelle.equals("Vorfuehrung")){
ProdEinst.vze.setText(h);
ProdEinst.zeit=Timestamp.valueOf(h);}
if(Tabelle.equals("Besucher")){
Produktverkauf.Besucher=h;
Produktverkauf.Be.setText(h);
}
if(Tabelle.equals("Produkt")){
Produktverkauf.Produkt=h;
Produktverkauf.Pr.setText(h);
}



       }});
```

9 Zusammenfassung

Es wurde eine Client-Server Applikation erstellt die das vorgegebene Datenbankschema anwendet.Die Funktionsweise entsprischt dem vorliegenden Geschäftsprozess einer Kinokette.Deweiteren wurden unabhängig von der Klausur weitere elementare Funktionen hinzugefügt,um der Gebrauchstaugligkeit zu entsprechen.Die Anlagen enthalten den gesamten Quellcode und Workspaces und eine ausführbare Version des Programms sowie den Quellcode des Wikis,dieses ist unter folgender Adresse zu finden:

`http://www.michael-hielscher.de/wiki/index.php/Client-Server_Aplikation_mit_Java`

10 Anlagen auf CD

- Administrator
- beleg.pdf
- Ticketverkauf und Produktverkauf
- Workspace:Admins
- Workspace:Verkauf
- Wiki.html