Georg Habeler

YPL Bilddatenbank der Proteinlokalisierung in Hefezellen

Diplomarbeit

Institut für Elektro- und Biomedizinische Technik
Technische Universität Graz
Inffeldgasse 18, A - 8010 Graz

Vorstand:
Univ.-Prof. Dipl.-Ing. Dr. techn. Gert Pfurtscheller

Betreuer:
Ao. Univ.-Prof. Dipl.-Ing. Dr. techn. Zlatko Trajanoski
Dipl.-Ing. Gerhard G. Thallinger

Begutachter:
Ao. Univ.-Prof. Dipl.-Ing. Dr. techn. Zlatko Trajanoski

Graz, November 2001

Kurzfassung

YPL.db (Yeast Protein Localization Database) enthält Information über die Verteilungsmuster spezifischer Proteine in Hefezellen. Das Gen des zu untersuchende Proteins wird mit dem Gen eines fluoreszierenden Proteins fusioniert. Von den Hefezellen, die sich aus dem manipulierten Erbmaterial entwickeln, werden konfokalmikroskopisch Schichtbilder aufgenommen. Die Lokalisierung, die experimentellen Daten und die Bilder werden in einer relationalen Datenbank (Oracle 8) archiviert. Eine Oracle Forms 6 Applikation ermöglicht die Verwaltung der Datenbank. Die Eingabe der Daten wird durch stapelweises Importieren der Bilder in die Datenbank und durch eine flexible Festlegung von Voreinstellungen beschleunigt. Eine benutzerspezifische Berechtigungsvergabe verhindert die Manipulation kritischer Daten durch Unbefugte. Vordefinierte Stammdaten standardisieren die Einträge und schaffen eine zuverlässige Datenkonsistenz. Genname, Proteinlokalisierung, Wachstums- bedingungen und eine Reihe anderer Attribute können als Abfragekriterien eingesetzt werden. Eine dynamische Webanbindung mittels Application Server ermöglicht den Zugriff auf die Datenbank unter http://ypl.tugraz.at. Genselektive Links zu MIPS, YPD und SGD ermöglichen in der Webanwendung einen schnellen Zugriff zu anderen relevanten Daten, die nicht in YPL.db verfügbar sind.

Schlagwörter: Konfokalmikroskopie, Protein, Lokalisierung, Oracle, Bioinformatik, Bilddatenbank, dynamische Webseite

Abstract

The YPL.db (Yeast Protein Location Database) contains information about the localization patterns of yeast proteins resulting from microscopic analyses. The data and parameters of the experiments to obtain the location information together with images from confocal or video microscopy are stored in a relational database, building an archive of and the documentation for all experiments. The database can be queried based on gene name, protein location, growth conditions and a number of additional parameters. All experiment parameters are selectable from predefined lists to ensure database integrity and conformity across different investigators. The database provides a structure reference resource to allow better characterization of unknown or ambiguous localization patterns. YPL.db is available via the World Wide Web at http://ypl.tu-graz.ac.at. Links to MIPS, YPD and SGD databases are provided to allow fast access to further information not contained in the location database itself.

Keywords: Confocal microskopy, Protein, Localization, Oracle, Bioinformatics, Image database, dynamic Website

Inhaltsverzeichnis

Abbildungsverzeichnis

Glossar

ADA Ada Byron 1815-1852. Modular aufgebaute Programmiersprache.

API Application Programming Interface

Allele Die alternativen Formen eines Gens, die an einem bestimmten Chromosomenort sitzen. Durch Mutation können neue Allele entstehen.

Application Server Ein zentraler Server, der Applikationslogik verarbeitet. Die Präsentation erfolgt über das Internet im Browser des Klienten.

Basenpaar Basen, wie Adenin und Thymin oder Guanin und Cytosin, werden durch eine schwache Wasserstoffbrücke zusammengehalten.

BFILES Methode zur Einbindung von Dateien (zB. Bilddatei), die im Verzeichnissystem liegen, in eine Datenbank.

Bioinformatik Interdisziplinäre Wissenschaft Verbindung Molekularbiologie und Datenverarbeitung

BLOBS (Binary Large Object) Einbindung von Dateien (zB. Bilddatei) unmittelbar in eine Datenbank.

Chromosomen sind fadenförmige Gebilde im Zellkern jeder Zelle, die die aus DNA bestehenden Gene tragen und für die Übertragung der verschiedenen, im Erbmaterial festgelegten Eigenschaften von der sich teilenden Zelle auf die beiden Tochterzellen verantwortlich sind.

DB Datenbank

DBMS Database Management System

DDL Data Definition Language

DML Data Manipulation Language

DNA auch DNS, Desoxyribonukleinsäure: in allen Lebewesen vorhandener Träger der genetischen Information; besteht aus zwei spiralig angeordneten Ketten von Nukleotiden, die durch 4 verschiedene, sich in unterschiedlicher Reihenfolge wiederholender Basen über Wasserstoff-brücken (in der Kopplung Adenin-Thymin und Guanin-Cytosin) miteinander verbunden sind

Fluoreszenz Leuchten von Materie bei Bestrahlung mit Licht.

GFP Grün Fluoreszierendes Protein

GUI Graphical User Interface: Grafische Benutzeroberfläche

Gen Erbeinheit: Ein Gen bestimmt die Ausbildung eines bestimmten Protein bzw. einer Polypeptidkette. Gene sind in Chromosomen angeordnet und sind Teil der DNA.

Genom Die Gesamtheit aller Gene eines Organismus.

HTML Hypertext Markup Language: Bezeichnung einer Beschreibungssprache zur Gestaltung von Inhalten im World Wide Web.

HTTP Hypertext Transfer Protocol: Das Datenübertragungsprotokoll des World Wide Web zur Anforderung und Übertragung von Daten zwischen Computern.

LOB`s Large Objects: Direkte Speicherung von Dateien (zB. Bilddatei) in einer Datenbank.

MIPS Munich Information Center for Protein Sequences

Molkeularbiologie Erforschung der molekularen (DNA, Polypeptidketten, ...) Mechanismen in Lebewesen.

RDBMS Relational Database Management System

RNA auch RNS, Ribonukleinsäure: Die RNA liegt im Gegensatz zur DNA nicht in Form von Doppelsträngen vor (außer bei Viren). Sie enthält den Zucker Ribose und anstatt der Base Thymin Uracil.

SGD Saccharomyces Genome Database

SQL Structured Query Language

SSP Security Service Protocol

WWW World Wide Web: Bezeichnung für einen einfach zu bedienenden Dienst im Internet, der mittels Hyperlinks den einfachen Zugriff auf multimediale Informationen entfernter Datenbanken ermöglicht, ohne an eine bestimmte genormte Dateistruktur gebunden zu sein.

YPD Yeast Proteome Database

1 Einleitung

Durch molekularbiologische (siehe Glossar) Forschung werden Erkenntnisse grundlegender Mechanismen aller Lebewesen gewonnen. Man erwartet, daraus neue Ansätze für präventivmedizinische, diagnostische und therapeutische Methoden zu finden.

Biologie ist heute die Wissenschaft, die am meisten Publikationen liefert. Neue Technologien hochparallelisierter Analyseverfahren produzieren enorme Datenmengen. Diese Daten können nur dann sinnvoll genutzt werden, wenn sie strukturiert archiviert sind und durch leistungsfähige Dataretrivalstrategien (Datenwiedergewinnung) wirksam durchsuchbar sind. Durch die enorme Datenflut aus der modernen molekularbiologischen Forschung werden hohe Ansprüche an die Datenverwaltung gestellt. Das Internet in Verbindung mit Datenbanken ermöglicht einen einfachen Zugang und eine sehr schnelle Verbreitung wissenschaftlicher Erkenntnisse.

1.1 Biologie

1.1.1 Die Zelle

Die Zellen höherer Lebewesen enthalten, im Gegensatz zu Bakterien und Blaualgen, einen vom Cytoplasma abgegrenzten Zellkern, der die Erbinformation enthält. Dieser höher entwickelte Zelltyp wird eukaryotisch (siehe Abbildung 1) genannt, der Zelltyp von Bakterien und Blaualgen dagegen prokaryotisch. Eukaryotische Zellen sind viel größer und, vom Standpunkt der Evolution betrachtet, viel weiter entwickelt. Sie zeichnen sich weiterhin dadurch aus, daß sie Zellorganellen mit eigenen Membranen besitzen, wie die Mitochondrien, das Endoplasmatische Reticulum und den Golgi-Apparat. Die Stoffwechselreaktionen finden nicht mehr alle im Cytoplasma statt, sondern sind auf die verschiedenen Zellorganellen aufgeteilt. Die Zellteilung erfolgt in der Regel durch Mitose.

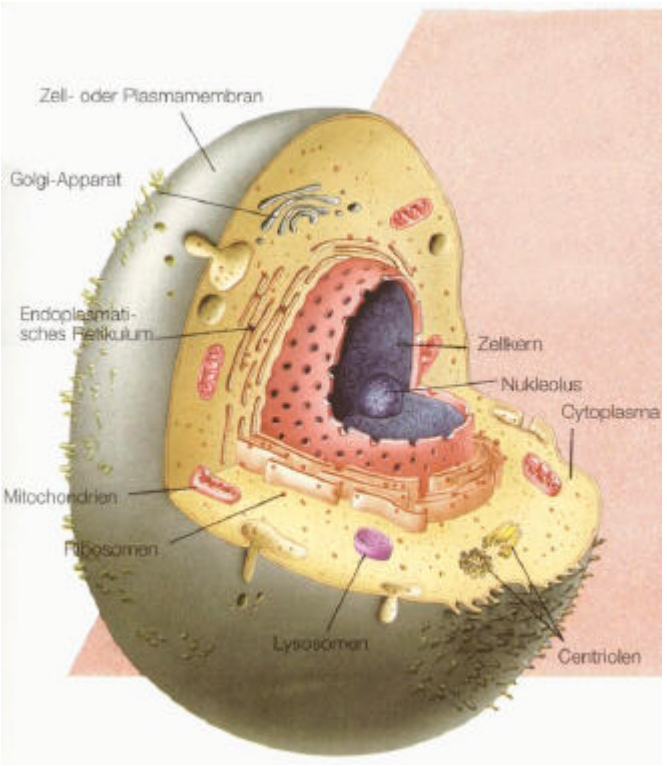

Abbildung 1: Eukaryontische Zelle mit deren Organellen.

Betrachtet man die Struktur der Zellen, so besitzen Menschen, Tier- und Pflanzenzellen trotz gewisser Parallelen große Unterschiede. Besonderheiten der Pflanzenzelle sind eine dicke Zellwand aus Zellulose, große Vakuolen und spezifische Zellorganellen, die Plastiden, in denen das Chlorophyll enthalten ist.

1.1.2 Proteine

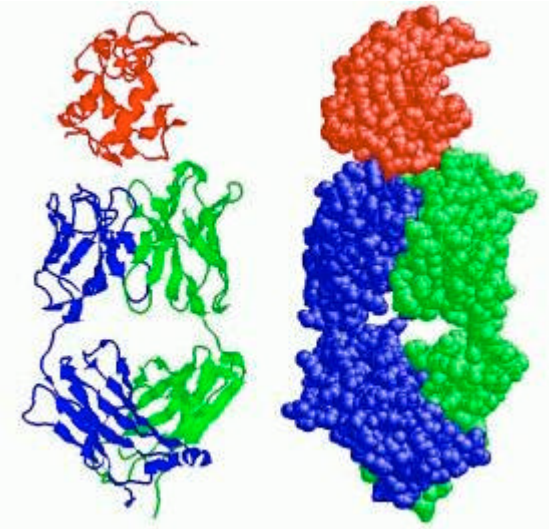

Abbildung 2: Dreidimensioale Veranschaulichung von drei interagierenden Proteinen.

Die Proteine (Eiweiße) sind große Moleküle (siehe Abbildung 2), die aus den Elementen Kohlenstoff, Wasserstoff und Stickstoff, teilweise auch aus Schwefel und Phosphor bestehen. Gene beschreiben den Bauplan der Proteine. Sie sind aus einfachen Untereinheiten, den Aminosäuren, zusammengesetzt. In der Zelle erfüllen die Proteine vielfache Aufgaben:

- **Strukturfunktion:** Gemeinsam mit den Fetten bilden sie Membranen und andere Strukturen, die dem Organismus seine Form und Stabilität geben (zB::Kollagen, Keratin, Elastin).
- **Katalytische Funktion:** Alle biologischen Reaktionen werden von bestimmten Proteinen, den Enzymen, gesteuert.Transportfunktion: Das Hämoglobin des Blutes transportiert den Sauerstoff im Körper.
- **Bewegung:** z.B. bei den Proteinen der Muskeln(Aktin, Myosin).Ernährungs- und Reservefunktion.. z.B. bei den Albuminen, dem Kasein der Milch oder beim Ferritin, das als Speicherprotein für Eisen dient.

- **Abwehrfunktion:** durch Immunglobuline oder durch Antikörper Regulierende Funktion, denn viele Hormone sind Eiweiße.
- **Andere Funktionen:** Beispielsweise die Gefrierschutzproteine, die bei Fischen in den Polarmeeren das Einfrieren verhindern etc.

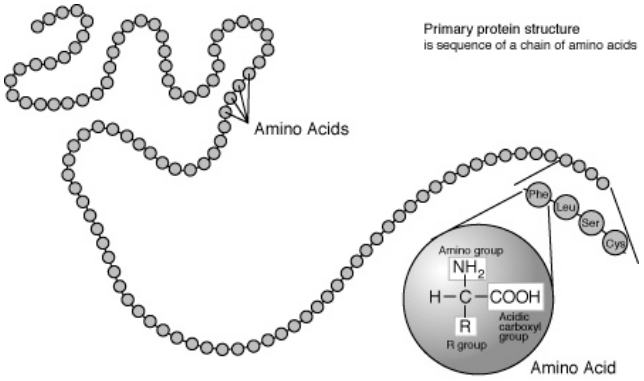

Abbildung 3: Atomare Struktur von Aminosäuren und deren Verkettung in Proteinen.

Die Aminosäurensequenz (siehe Abbildung 3) aller Proteine ist genetisch vorprogrammiert und ist in der Erbsubstanz DNA verschlüsselt. Oft sind die Aminosäuresequenzen typisch für bestimmte Organismengruppen, so daß man die Unterschiede zwischen den Sequenzen als Grad für die Verwandtschaft verwenden und so Stammbäume aufstellen kann. Aminosäuren sind Derivate von organischen Säuren, die zusätzlich eine Aminogruppe aufweisen. Die Proteine werden im Wesentlichen bei allen Organsimen aus zwanzig Aminosäuren gebildet. Durch die ihnen eigene COOH-Gruppe und NH2-Grupppe haben sie gleichzeitig sauren und basischen Charakter.

Die konkreten Eigenschaften und Funktionen [9] einzelner Proteine herauszufinden ist einer der Schwerpunkte biochemischer Forschung. Eine Gruppe von Wissenschaftlern am Institut für Biochemie der Technischen Universität Graz untersucht die Lokalisierung spezifischer Proteine [4] in der Hefezelle, um daraus Rückschlüsse auf deren Funktion ziehen zu können. Die Lokalisation der Proteine kann durch das selektive Markieren (siehe Abbildung 4) eines bestimmten Proteintyps mit einem fluoreszierenden Protein unter dem Mikroskop sichtbar gemacht werden. Weiters werden durch Vitalfärbung die Organellen in der Zelle eindeutig erkennbar. Aus

diesen Bildinformationen können Wechselwirkungen im Zellinneren in bestimmten Entwicklungsstadien der Zellen beobachtet werden.

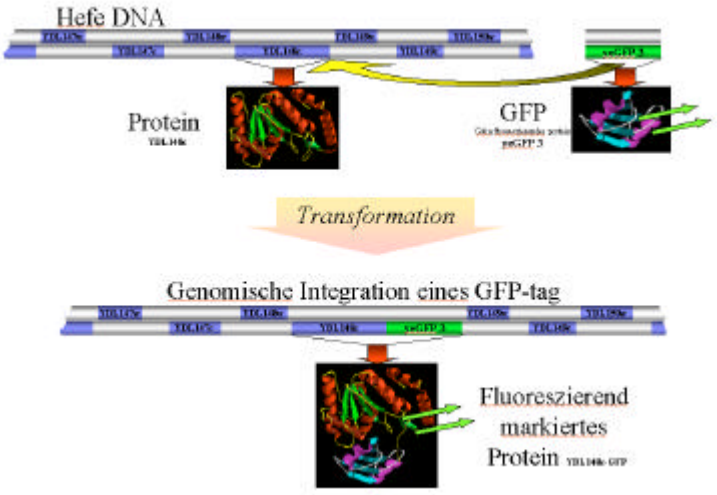

Abbildung 4: Prinzip der Proteinmarkierung mit fluoreszierenden Proteinen.

1.1.3 Hefe

Hefe "Saccharomyces Cerevisiae" ist ein wichtiger Modellorganismus für zellbiologische Forschung. Er war der erste eukaryontische Organismus dessen Genom vollständig sequenziert [5] wurde. Seither ist Hefe ein wichtiger Modellorganismus der Genomforschung. Viele menschliche Krankheitsgene haben Homologien in der Hefe [13]. Aufgrund weitestgehender Übereinstimmung der Stoffwechselfunktion ist Hefe für zellbiologische Studien hervorragend geeignet.

1.1.4 Fluoreszenz Konfokal Mikroskopie

Wegen der kleinen Abmessungen der Zellen, zwischen 5 und 10 Mikrometer, ist die lichtmikroskopische Analyse der subzellulären Strukturen in lebenden Hefezellen [7] experimentell aufwendig. Dennoch sind fluoreszenzmikroskopische Methoden für Hefe gut geeignet. Da die Größenordnung der Hefezellen ungefähr der Wellenlänge sichtbaren Lichtes entspricht, gerät die mikroskopische Analyse an ihre Grenzen.

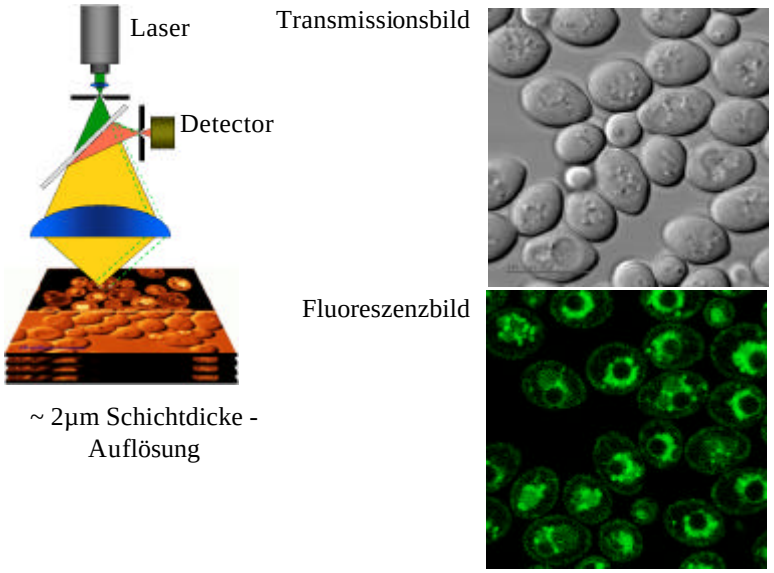

Abbildung 5: Schematische Darstellung konfokaler Lasermikroskopie [6].

Probleme, die sich daraus ergeben sind:

- Die maximale Auflösung bei der Wellenlänge des emittierten Lichtes entspricht etwa 0,25 µm.
- Die Fluoreszenzemission der Bereiche der Zelle, die nicht in der fokalen Fläche liegen, produzieren Unschärfen.
- Interferenzen durch Beugung entstehen durch die starke Hefezellwand.
- Die Bewegung der gelösten Zellen und die interne Dynamik der Strukturen in der Zelle ergeben weitere Beschränkungen.

Wie bei allen mikroskopischen Techniken, wird die Bildqualität durch die Vergrößerung, die Wellenlänge, die Signalintensität und die Dauer der mikroskopischen Beobachtung begrenzt (Bilddatenerfassungszeit). Die technischen Artefakte müssen im Zusammenhang mit experimentellen Rahmenbedingungen betrachtet werden, die durch das lebende Probenmaterial und mögliche Störungen der Einfärbungsmethoden einher gehen. Fixierung der Zelle und Temperaturüberwachungstechniken erlauben eine fast störungsfreie Beobachtung der Hefezellen. Durch die Visualisierung vieler Zellen können die Lokalisationsmuster statistisch gut erfaßt werden. Anhand der Hefe kann die Homogenität eines

eingefärbten Musters durch Beobachtung einer großen Population von Zellen bestimmt werden. Fortschritte in der elektronischen Lichtmikroskopie durch Video-mikroskopie und Konfokaler Laserabtastmikroskopie (siehe Abbildung 5) haben eine wesentliche Verbesserung bei Aufnahmen der Hefezellen ergeben. Als Folge dieser technologischen Entwicklungen wurde Hefe ein interessantes System, nicht nur für strukturelle und morphologische Studien, sondern auch für die Analyse der Organellendynamik und der Vererbung.

Folgende Punkte liefern Einblicke in die Struktur der Hefezellen und erweitern die experimentellen Möglichkeiten.

- Organellenspezifische Leuchtstofffärbungen (Vital Staining),
- Die Messung physiologischer Parameter (z.B. pH, Ca2+, Membrane Potential),
- Die Anwendung des grünen Leuchtstoffproteins (GFP) und seine verschiedenen spektralen Ableitungen als Fluoreszenzmarker zur Analyse der Protein-lokalisierung in lebenden Zellen.

Da GFP nur aus 10 bis 12 Aminosäuren bestehen ist unwahrscheinlich, daß die eigentliche Proteinfunktion verändert wird, obgleich diese selbstverständlich von den Eigenschaften des markierten Proteins und der c -, oder n - terminalen Bindungsseite des Markers abhängt. Um die Leuchtintensität zu erhöhen, können mehrere GFP Marker integriert werden. Daraus resultiert die Verstärkung des Signals aber auch die Vergfößerung des Proteins und infolgedessen die mögliche strukturelle Änderung des markierten Proteins. In jedem Fall ist die Funktion des markierten Proteins durch andere Experimente Sicherzustellen.

1.2 Datenbanken

Zur Speicherung, Manipulation, Wiedergewinnung (Data Retrieval) und Veröffentlichung (Internetanbindung) großer Mengen von Daten sind Computer besonders geeignet. Dateisysteme auf permanenten Speichermedien sind dafür die einfachste Möglichkeit. Anforderungen bezüglich der Integrität der Daten, der

assoziativen Suche nach Daten (Assoziative Suche umfasst insbesondere die Suche nach Werten und nach Beziehungen innerhalb einer Sammlung gespeicherter Daten) und der Synchronisation (schreibender Mehrbenutzerzugriff) werden Dateisysteme jedoch nicht gerecht.

In Datenbanksystemen werden diese Aspekte berücksichtigt. Die Datenverwaltung geschieht im Rahmen eines eigenständigen Datenmodells, das im Idealfall:

- Vollständig die realen Gegebenheiten erfaßt.
- Assoziative Suche durch eine Abfragesprache (SQL Structured Query Language) zulässt.
- Datenkonsistenz durch Integritätsregeln sichert.
- Datenredundanz vermeiden hilft.

Die Integritätssicherung der Daten umfaßt sowohl operationale Sicherheit (Speichermedienfehler, Synchronisation paralleler Zugriffe) als auch die Festlegung semantischer Konsistenzbedingungen (z.B. referentielle Integrität, Schlüsseleigenschaft).

In den 70er Jahren begann die Entwicklung von Datenbanken. Seither wurden unterschiedliche Ansätze bezüglich der Datenmodellierung verwendet. Durchgesetzt hat sich das auf mengenorientierten Zugriff basierende relationale Datenmodell gegenüber dem Netzartigen Modell und dem Hierarchischen Datenmodell. Systeme wie DB2, System R, Ingres, Informix, PostgreSQL, Interbase, Oracle und andere setzten das relationale Modell seit Mitte der 80er Jahre sehr erfolgreich um. Für den Entwurf von Datenmodellen eignet sich das Entity-Relationship Modell (siehe Seite 11).

1.2.1 Begriffe und Konzepte

Datenbanksysteme stellen eine Technologie dar, die Gegenstand der Forschung verschiedenster Disziplinen ist. Trotz dieser Vielfalt sind jedoch im Laufe der Jahre grundsätzliche Konventionen über Begriffe, Zielsetzungen und Grundkonzepte dieser

Technologie entstanden, die in mehreren abgeschlossenen und laufenden Standardisierungen festgelegt wurden.

Grundsätzlich wird zwischen zwei Aspekten unterschieden:

- **Datenbeschreibungsfunktionen:** Data Definition Language (DDL)

  Durch Nutzung der DDL wird im wesentlichen festgelegt, welche Daten in das Datenbanksystem aufgenommen werden dürfen, welche speziellen Konsistenzbedingungen für die Daten bestehen müssen, und wie sie physikalisch organisiert werden sollen. Die mit der DDL getroffenen Deklarationen machen das Schema einer Datenbank aus, das die Grundlage für eine Integritätssicherung aller Einfüge-, Lösch- und Änderungsoperationen ist.

- **Datenzugangsfunktionen:** Data Manipulation Language (DML).

  Durch Benutzung der DML werden Daten in die Datenbank eingebracht und dort verwaltet. Die DML erlaubt in der Regel das Einfügen, Ändern und Löschen sowie die assoziative Suche nach Werten. Alle Operationen der DML stellen dabei sicher, daß die im Schema abgelegten Deklarationen nicht verletzt werden.

1.3 Ziele

Die "Yeast Genetics and Molecular Biology Group TU Graz" am Institut für Biochemie der Technischen Universität Graz untersucht in Kooperation mit Universitäten in USA und Großbritannien in zahlreichen Experimenten die Lokalisierung der ca. 6200 unterschiedlichen Proteine in der Hefezelle. Um die erforschten Daten, einschließlich der mikroskopischen Bilder, verwalten zu können und diese für andere Wissenschaftler zugänglich zu machen, sollte eine relationale Datenbank mit entsprechenden Werkzeugen für die Dateneingabe und Abfrage erstellt werden.

Das Projekt umfaßt folgende Teilbereiche:

- Die Datenbank
  - Die Modellierung eines relationalen Datenbankschemas um die experimentellen Daten vollständig und redundanzfrei zu erfassen.
  - Die Implementierung des Schemas auf einer Oracle 8 Datenbank.
- Die Verwaltungswerkzeuge

  Die Erstellung einer Multiform Anwendung in Oracle Forms 6 im Client/Server Modus, mit einer Benutzerverwaltung für die Eingabe von Experiment- und Bilddaten in die Datenbank, für die Bestimmung der Daten die im Internet veröffentlicht werden sollen und zur spezifischen Abfrage des Datenbestandes.

- Der Internetzugang

  Die Anbindung der Datenbank an das Internet in Form einer dynamischen HTML-Anwendung mit Apache Webserver, Oracle Internet Application Server und PL/SQL Webtoolkit mit einfachen Abfragemöglichkeiten bezüglich der Gen- bzw. Proteinbezeichnung und deren Lokalisierung und gen- bzw. proteinspezifischen Links zu den Datenbanken YPD, SGD und MIPS.

- Die Datenimportfunktion

  Die Erstellung eines PL/SQL-Packages für den Import von Daten und Bildern aus einer Tabelle in das relationale Datenmodell.

2 Methoden

In diesem Kapitel wird auf Konzepte, Programmiersprachen, Entwicklungswerkzeuge und auf Lösungswege eingegangen.

2.1 Datenmodellierung, Schemaimplementierung

2.1.1 Das relationale Datenmodell

1970 begründete E.F. Codd [1] das relationale Datenmodell, das die konzeptionelle Grundlage relationaler Datenbanken wurde. Wegen der mengenorientierten Datenmanipulation, die eines der wesentlichsten Merkmale des relationalen Modells ist sowie der einfachen Datenstruktur, die auch die Modellierung sehr umfangreicher Strukturen ermöglicht, ist es heute das am weitesten verbreitete Datenmodell. Das relationale Modell wird durch eine mathematische Grundlage, die Relationale Algebra fundamentiert. Die beliebige Kombination neun grundlegender Operationen auf Relationen ergibt stets wieder eine Relation. Die Abbildung eines realen Objektes entspricht der Relation im Modell. Der Relation im Modell wiederum entspricht die Tabelle in der Datenbank. Spalten der Tabelle werden Attribute im Tabellenkopf zugeordnet, Zeilen bzw. Tupel sind die Ausprägungen.

2.1.2 Schlüssel, Fremdschlüssel und Referentielle Integrität

- Eine Attributgruppe X einer Relation R heißt Schlüssel von R, falls alle Tupel unter X verschieden sind (Schlüsseleigenschaft).
- X heißt minimaler Schlüssel, wenn die Schlüsseleigenschaft nur für X, aber keine (echte) Teilmenge von X gilt.
- Kein Tupel darf in einer Relation existieren, bei dem Tupelkomponenten des Primärschlüssels mit NULL belegt sind oder die Schlüsseleigenschaft verletzt ist.
- Für jedes Tupel einer Relation R, die den Primärschlüssel einer (nicht notwendig verschiedenen) Relation S als Fremdschlüssel enthält, gilt, daß entweder alle

Tupelwerte des Fremdschlüssels mit NULL belegt sind oder aber ein Tupel in S mit gleichen Primärschlüsselwerten existiert.

Das Schlüsselkonzept des relationalen Datenmodells erlaubt die tabellarische Darstellung von Information. Ein Schlüssel ist ein eindeutiger Identifikator eines Tupels in einer Relation. Durch das Schlüsselkonzept wird ausgeschlossen, daß Tupel mehrfach in einer Relation vorkommen. Grundsätzlich ist die Reihenfolge, bzw. die Position sowohl von Tupeln, wie auch die von Attributen nicht von Bedeutung. Festgelegt ist daß jede Tupelkomponente atomar (nicht zerlegbar) ist. Zumindest die Gesamtmenge aller Attribute einer Relation müssen einen minimalen Schlüssel bilden. Die Angabe eines Primärschlüssels ist daher in jedem Fall möglich. Dieser Primärschlüssel wird praktisch zur eindeutigen Adressierung eines einzelnen Tupels, gleichgültig an welcher Position er sich befindet, verwendet. Ein aus wenigen Attributen zusammengesetzter Primärschlüssel erhöht die Übersicht in großen Schemata. Die Beziehung zwischen Entitäten im relationalen Modell werden durch Primärschlüssel in der Einen und eines sich darauf beziehenden Fremdschlüssels in der anderen Relation abgebildet. Referentielle Integrität, eine wesentliche Sicherstellung der Datenkonsistenz, wird durch den Zusammenhang zwischen Primärschlüssel und Fremdschlüssel erreicht. Die Durchsetzung der referentiellen Integrität ist eine Kernaufgabe eines RDBMS (Relationales Datenbankmanagementsystem).

2.1.3 Normalformen

Normalformentheorie entstand im Rahmen der Entwicklung des Relationenmodells. Sie bestrebt die Vermeidung von Datenredundanzen und schließt damit Dateninkonsistenz beim Einfügen, Ändern und Löschen von Tupeln aus. Eine Kernbereich des Datenbankentwurfs ist die Erstellung des konzeptuellen Relationenschemas. Die Normalformentheorie wird in dieser Modellierungsphase eingesetzt. Nach der Identifizierung relevanter Daten erfolgt durch Normalisierung die systematische Aufteilung der Daten in mehrere Relationen sowie die Einteilung der Relationenspezifischen Attribute. Die Anwendung der Relationaler Algebra setzt die erste Normalform voraus.

Wesentlich sind die ersten drei Normalformen:

- **Erste Normalform**

  Ausschluss von nichtatomaren (teilbaren) Attributwerten innerhalb einer Relation eines Schemas. Vektoren oder Mengen als Tupelwerte werden dadurch ausgeschlossen.

- **Zweite Normalform**

  Eine Relation ist in zweiter Normalform, wenn sie in erster Normalform ist, und wenn kein Nichtschlüsselattribut von einer Teilmenge eines minimalen Schlüssels funktional abhängig ist. Relationen, die nicht in zweiter Normalform sind, werden durch Herauslösen der Attributgruppe, die bereits von einer Teilmenge eines minimalen Schlüssels funktional abhängig sind, in zweite Normalform gebracht. Die herausgelöste Attributgruppe und die Teilmenge des minimalen Schlüssels, von der sie abhängig ist, werden zu einer neuen Relation kombiniert.

- **Dritte Normalform**

  Eine Relation ist in dritter Normalform, wenn sie in zweiter Normalform ist und kein Nichtschlüsselattribut transitiv von einem minimalen Schlüssel abhängig ist. Die dritte Normalform fordert, funktionale Unabhängigkeit, die zwischen Attributgruppen bestehen, die nicht Teil eines minimalen Schlüssels sind.

2.1.4 Transaktionen und operationale Integrität

Als operationale Integrität wird die Bewältigung von zwei speziellen Problemen bezeichnet:

- Gleichzeitiger bzw. quasiparalleler Zugriff mehrerer Benutzer auf gemeinsame Daten muß synchronisiert werden.
- Dateninkonsistenz verursacht durch physikalische Speicherfehler.

Die Maßnahme zur Durchsetzung der operationalen Integrität wird als Transaktionskonzept bezeichnet. Eine Transaktion besteht aus einer

zusammengehörigen Abfolge von DDL oder DML Operationen (zB. Abbuchung von Konto A und Zubuchung auf Konto B). Diese Maßnahmen können zu Datenverlust führen, aber die Entstehung inkonsistenter Datenbestände wird dadurch konsequent verhindert.

- **Atomarität:** Entweder die gesamte Sequenz von Operationen wird ausgeführt (Abschlußbefehl einer Transaktion „Commit“) oder keine (Der Abbruchbefehl einer Transaktion „Rollback“ verwirft alle bis zum letzten „Commit“ eingelangten Befehle und führt dadurch keine der Operationen dieser Transaktion durch).
- **Konsistenz:** Nach Abschluss der Transaktion wird, entsprechend datenmodellspezifischer Konsistenzregeln, die Datenbank in konsistentem Zustand hinterlassen.
- **Isolation:** Durch Transaktionen geänderte Daten oder Datenobjekte sind erst nach deren Abschluss für andere Benutzer sichtbar, und können erst dann durch neue Transaktion manipuliert werden.
- **Dauerhaftigkeit:** Abgeschlossene Transaktionen sind persistent (zB. Systemfehler kann keine Dateninkonsistenz verursachen).

Die Synchronisation von Transaktionen wird durch das RDBMS erfüllt.

2.1.5 Datenbankentwurf mit dem Entity-Relationship Modell

Die Abbildung von Informationsstrukturen der Wirklichkeit auf das Schema einer Datenbank erfolgt indirekt über ein semantisches Datenmodell, das dann auf das konzeptuelle Schema (DDL, Datenbankobjekte) transformiert wird [3]. Das am häufigsten zu diesem Zweck eingesetzte semantische Datenmodell ist das Entity-Relationship Modell, das in der heute gebräuchlichen Form einer Weiterentwicklung des 1976 von P. Chen spezifizierten Modells entspricht. Das Entity-Relationship Modell ist unabhängig vom hierarchischen, netzartigen und relationalen Datenmodell. Es bietet eine anschauliche grafische Darstellungsmethodik. Das Entity-Relationship Modell beschreibt anhand von Objekten (Entities), zwischen denen Beziehungen (Relationships) bestehen, reale Gegebenheiten. Ebenso wie im relationale Modell

werden im Entity-Relationship Modell Attribute zur Beschreibung von Objekten und Beziehungen sowie das Primärschlüsselkonzept verwendet.

2.1.6 SQL Data Definition Language (DDL)

Die Erstellung der Objekte die die Daten enthalten werden, erfolgt in einem Schema, das einem Benutzer zugeordnet ist und eine Reihe von Relationen enthält. Relationen werden in SQL als Tabellen bezeichnet, die aus mehreren benannten Spalten bestehen. Tupel werden in SQL Zeilen genannt. Für die Tabellendefinition steht jeweils eine Erzeuge- (Create), Änder- (Alter) und Löschoperation (Drop) zur Verfügung. Für kürzere Datenzugriffszeiten und die Festlegung von Konsistenzbedingung werden gegebenenfalls Indizes erstellt.

2.2 Architekturen verteilter Anwendungen

2.2.1 Client-Server-Architektur

Bei den ersten Datenbanken auf Großrechnern erfolgte sowohl die Auswertung der Benutzeranfrage wie auch die Generierung des Abfrageergebnisses auf dem Großrechner. Im Gegensatz dazu wird bei LAN (Local Area Network) basierten PC-Arbeitsplätzen der Fileserver nur zur Ablage der Dateien, welche die Daten der Datenbank repräsentieren, genutzt. Die Auswertung der Nutzeraktivitäten und die Durchführung der Datenabfrage erfolgen auf der Arbeitsstation des Nutzers. Diese beiden Methoden sind ineffizient, da im ersten Fall die gesamte Last serverseitig bewältigt werden muß, bzw. im zweiten Fall der gesamte Datenbestand zum PC übertragen werden muß, wodurch eine enorme Netzbelastung und Verzögerungen bei gleichzeitiger Arbeit mehrerer Arbeitsplätze entstehen.

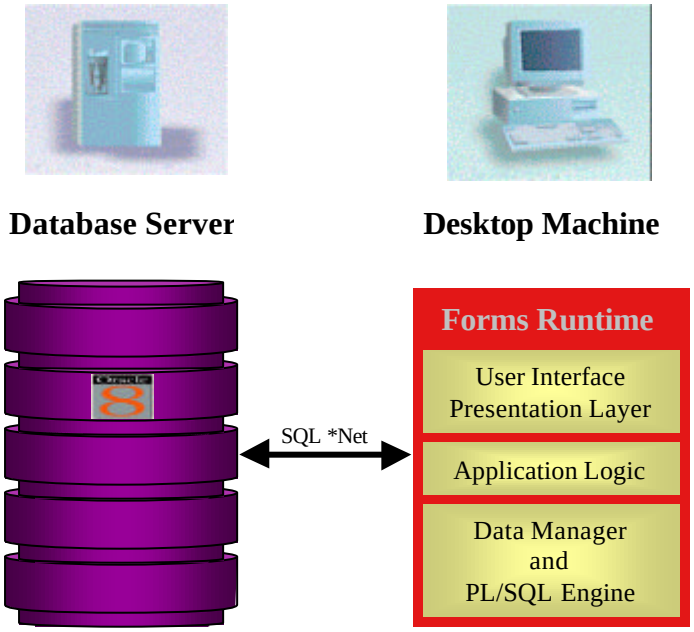

Abbildung 6: Client-Server-Architektur.

Die Lösung liegt in der Verteilung der Lasten bei geringer Netzbelastung. Diese Systemlösungen werden, wegen der Aufteilung der Aufgaben in Nutzerinteraktionen (Client) und die getrennte Dienstleistung (Server) zur Realisierung der Auswertungen, als Client-Server-Systeme (siehe Abbildung 6) bezeichnet. Die Arbeitsstation dient bei diesen Systemen als intelligentes Front End für den Benutzer. Eine Datenabfrage des Anwenders wird meist in ein SQL-Statement umgewandelt und über das Netz an den Datenbankserver geschickt. Der Server wertet die SQL-Anweisungen aus und führt die entsprechenden Datenbankabfragen aus. Die Ergebnisse dieser Abfragen und auch nur diese werden über das Netz zurück an die Arbeitsstation geschickt.

Eigenschaften von Client-Server-Systemen sind:

- Deutlich reduzierte Netzbelastung.
- Die Performanceanforderungen an die Arbeitstationen selbst können heruntergesetzt werden. Ausreichend ist eine befriedigende Leistung bei der Abwicklung der Bedienvorgänge.
- Die Serveranforderungen (meist ein separater Datenbankserver) steigen zwar an, doch verbessert sich bei vergleichbarer Gesamtinvestition die Gesamtleistung des Systems.
- Wesentlich sind auch die Vorteile bei der Sicherheit des Systems. Da bei Client-Server-Systemen die Daten nicht mehr komplett über das Netz transferiert werden müssen und der Datenbankserver Zugriffsrechte selbst verwalten kann, wird ein unberechtigtes Abhören oder Abfragen von Daten deutlich erschwert.

2.2.2 Drei Schicht Architektur, Application Service

In den meisten Client/Server Anwendungen, laufen clientseitig rechenintensive Prozesse, die die Installation eines aufwendigen Programmpaketes erfordern. Obwohl die verwendeten Daten von einem Datenbankserver geladen werden, laufen Anwendungen auf Computern, die häufig begrenzte Rechenleistungen und Speicherkapazitäten haben. In einer "Three Tier Web-Implementierung" wird die Verarbeitung der Applikationslogik auf einen Applicationsserver zwischen Datenbankserver und Klientenrechner verlegt. Auf den Klient Rechner entfällt im wesentlichen nur mehr die Anzeige bzw. die Übermittlung interaktiver Benutzereingaben vom und zum Applicationsserver. Dadurch reduziert sich die clientseitige Installation zur Bewältigung großer Applikationen auf die Installation eines Standardwebbrowsers und eventueller Plugins.

2.3 Applikationsentwicklung

2.3.1 SQL Data Manipulation Language (DML)

Relationale Algebra stellt ein präzise definiertes Instrumentarium für den Umgang mit Relationen dar, ist jedoch für den Einsatz als DML praktisch nicht geeignet. SQL ist eine seit 1988 standardisierte Sprache (ANSI Standard) für relationale Datenbanken, die wichtige Bereiche der relationalen Algebra erfüllt. SQL stellt Elemente zur Datendefinition (DDL) und zur Datenmanipulation (DML) zur Verfügung. Der Einsatz ist sowohl im direkten Datenbankdialog als auch innerhalb von Programmiersprachen (Embedded SQL, Dynamic SQL, PL/SQL) möglich. DML ist auf vier grundlegende Operatoren reduziert. Durch die Kombination mehrerer Tabellen in einer Abfrage können sehr komplexe Konstrukte entstehen. SQL ist ursprünglich aus der Entwicklung relationaler Datenbanksysteme der Firma IBM (SYSTEM R, DB2) hervorgegangen. Mittlerweile bieten nahezu alle Anbieter relationaler Datenbanken eine SQL Schnittstelle an.

2.3.1.1 Datenabfrage

Die Möglichkeit zur Abfrage von Daten wird in SQL durch eine einzige Operation geboten, die in sich alle Möglichkeiten der relationalen Algebra vereint. Eine Abfrage mit dieser Operation hat stets wiederum eine Tabelle zum Ergebnis. Der erste Teil dieser Operation, die SELECT Klausel, bestimmt die Spalten der Ergebnisrelation. Der zweite Teil, die FROM Klausel, bestimmt die Tabellen und/oder Ansichten (Views), aus deren kartesischen Produkt die Ergebnisrelation gebildet wird. Die SELECT und FROM Klausel lassen sich folglich zusammengenommen als kartesisches Produkt, gefolgt von einer Projektion (bei der Projektion wird Eindeutigkeit von Spaltennamen durch optionales Voranstellen des Tabellennamens bzw. eines in der FROM Klausel vergebenen Alias bewirkt), auffassen. Die WHERE Klausel dient zur weiteren Qualifizierung der Ergebnisrelation und kann abhängig von der verwendeten Qualifizierung einen Join, eine Restriktion oder eine Selektion nach sich ziehen. Sie stellt einen logischen Ausdruck (unter Einbeziehung von Konjunktion, Disjunktion und Negation) von Prädikaten über Relationen dar. Dabei gilt insbesondere, daß die Schachtelung von Anfragen möglich ist (also ein Prädikat

angegeben werden kann, das zur Auswertung wiederum eine Anfrage erforderlich macht „Subqueries“). Die ORDER BY Klausel dient schließlich zum Ordnen der Tupel der Ergebnisrelation.

2.3.1.2 Relationenalgebra und SQL

Beschreibung der Grundoperationen der Relationenalgebra und deren Umsetzung in SQL anhand von Beispielen.

- **Vereinigung**
- **Differenz**
- **Kartesisches Produkt**
- **Projektion**
- **Selektion**

Die Relationen R und S

R:

A	B	C
a	b	c
d	a	f
c	b	d

S:

D	E	F
b	g	a
d	a	f

- **Vereinigung**

  Vereinigung ist die Menge aller Tupel bzw. Zeilen, die in R oder S oder in beiden Relationen bzw. Tabellen vorhanden sind (Voraussetzung: Relationen gleichen Grades). In SQL werden zwecks Eindeutigkeit den Spalten Aliase zugewiesen.

  SQL:

```
select A s1, B s2, C s3 from R union select D s1, E s2, F s3 from S
```

s1	s2	s3
a	b	c
d	a	f
c	b	d
b	g	a

- **Differenz**

  Differenz ist die Menge aller Tupel, die in R aber nicht in S enthalten sind. (Voraussetzung: Relationen gleichen Grades)

  SQL:

```
select A s1, B s2, C s3 from R minus select D s1, E s2, F s3 from S
```

s1	s2	s3
a	b	c
c	b	d

- **Kartesisches Produkt**

  Seien R und S Relationen mit Grad k1 und k2. Das kartesische Produkt ist die Menge aller (k1 + k2) gradigen Tupel, deren erste k1 Elemente ein Tupel in R und deren letzte k2 Elemente ein Tupel aus S darstellen.

  SQL:

```
select A, B, C, D, E, F from R, S
```

A	B	C	D	E	F
a	B	c	b	g	a
a	B	c	d	a	f
d	A	f	b	g	a
d	A	f	d	a	f
c	B	d	b	g	a
c	B	d	d	a	f

- **Projektion**

  Bei der Projektion werden einzelne Spalten einer Relation auf eine neue Relation abgebildet. Sei R eine Relation mit Grad k. Dann ist die Abbildung von R auf die Komponenten i1,i2,...,im. Die entstehende Relation ist m-gradig.

  SQL:

```
select A, C  from R
```

A	C
a	c
d	f
c	d

  Da Projektion eine Menge von Tupeln definiert, werden Duplikate eliminiert!

- **Selektion**

  F sei eine Formel. Diese Formel enthält Operanden (Konstante oder Komponente einer Relation) Aritmethische Vergleichsoperatoren <, =, > , <> Logische Operatoren AND, OR, NOT Dann ist die Selektion die Menge aller Tupel t in R, für die die Formel F wahr ist, wenn für jedes Tupel in R die entsprechenden Komponenten von R in F eingesetzt werden.

  Sei F: B = b, dann ist das Resultat
  SQL:

```
select  A, B, C  from R  where B = b
```

A	B	C
a	b	c
c	b	d

  Diese Grundoperationen genügen, um alle anderen Operationen der Relationenalgebra davon abzuleiten. Obwohl sie genügen, sind es nicht die praktischsten Operationen. Daher werden noch zusätzliche Operationen verwendet.

JOIN (Verbund)

Der Join ist die Wichtigste nicht-elementare Operation. Ein Join kann durch die Basis-Operationen wie folgt realisiert werden:

Beispiel:

R:

A	B	C
1	2	3
4	5	6
7	8	9

S:

D	E
3	1
6	2

Dann ist für B < D

SQL:

```
select A, B, C, D, E  from R, S   where B < D
```

A	B	C	D	E
1	2	3	3	1
1	2	3	6	2
4	5	6	6	2

Der häufigste Fall ist der Natürliche Verbund (Natural Join). Man bildet das kartesische Produkt R X S. Für jedes Attribut, das sowohl in R als auch in S erscheint, selektiert man die Tupel, für die die Werte der gleichnamigen Attribute übereinstimmen.

Beispiel:

R:

A	B	C
a	b	c
d	b	c
b	b	f
c	a	d

S.

D	E	F
b	c	d
b	c	e
a	d	b

SQL:

```
select R.A, R.B, R.C, S.D  from R, S  where R.B = S.B and R.C = S.C
```

A	B	C	D
a	b	c	d
a	b	c	e
d	b	c	d
d	b	c	e
c	a	d	b

Durch den Natural Join werden Relationen, die aus entwurfstheoretischen Gründen zerlegt wurden, für die Abfrage wieder kombiniert.

Die meistbenutzten Operationen der Relationenalgebra sind Select, Project und Join. Die Queries, die aus diesen 3 Operationen bestehen, werden als SPJ-Queries bezeichnet. Die performante Abarbeitung von SPJ-Queries ist bei einem relationalen DBMS entscheidend.

2.3.1.3 Einfügen, Ändern und Löschen

SQL Operationen zum Einfügen, Ändern und Löschen können auf einzelne Zeilen oder auf Mengen von Zeilen angewendet werden.

- **Einfügen:**

  einzelne Zeilen einfügen:

```
insert into Tabellenname (Spalte1, Spalte2)
                  values (Wert1  , Wert2  )
```

  mehrere Zeilen einfügen:

```
insert into Tabellenname_A (A1, A2)
                     select B1, B2 from Tabellenname_B
```

  Null-Werte können in Spalten eingefügt werden, sofern sie nicht von der Modelllierung her konsistenzbedingt auf NOT NULL festgelegt wurden, indem ihnen kein Wert zugewiesen wird.

- **Ändern:**

```
Update Tabellenname set Spalte1 = ‘kalt‘ where Spalte2 = ‘Winter‘
```

  In allen Datensätzen, in denen Spalte2 gleich ‘Winter‘ ist, wird Spalte1 gleich ‘kalt‘ gesetzt.

- **Löschen:**

```
Delete from Tabellenname where Spalte1 = ‘kalt‘
```

Allen Zeilen deren Spalte1 gleich ‘kalt‘ gesetzt ist, werden gelöscht. Wird keine einschränkende Klausel angegeben, werden alle Zeilen der Tabelle gelöscht.

2.3.2 PL/SQL

Alle im weiteren erwähnten Entwicklungswerkzeuge verwenden als Programmiersprache PL/SQL [14]. PL/SQL ist die von Oracle entwickelte prozedurale Erweiterung von SQL. Oracle Datenbanken können PL/SQL im RDBMS verarbeiten. Plattformunabhängigkeit liefern Runtime Enivonments, die auch clientseitig kompilierten PL/SQL Metacode verarbeiten können. Die syntaktische Struktur von PL/SQL ist an ADA angelehnt.

2.3.2.1 Eigenschaften von PL/SQL

PL/SQL ist eine systemunabhängige, für Datenbankanwendungen optimierte Programmiersprache.

- **SQL Unterstützung**

  SQL ist der Standard unter den Datenzugriffssprachen. PL/SQL erweitert SQL ohne separate Schnittstelle um prozedurale Funktionalität. In PL/SQL sind SQL Datenmanipulation, Cursor Funktionalität, Transaktionskommandos sowie SQL Funktionen verwendbar. Ebenso unterstützt PL/SQL alle SQL Datentypen, wodurch sich Datenkonvertierung zwischen Datenbank und Anwendung erübrigt.

- **Unterstützung objektorientierter Programmierung**

  Objekttypen erlauben eine realitätsnahe Modellierung. Komplexe Applikationsentwicklung wird durch wiederverwendbare Objekttypen modular und die Wartbarkeit wird dadurch verbessert. Die Kapselung von Operationen mit Daten ermöglicht es, Datenbearbeitung aus SQL-Scripts und PL/SQL Blöcken in Methoden umzulagern. Details der Implementation in Objekttypen sind nach außen hin nicht sichtbar, sodaß Änderungen der Methoden keine Auswirkung auf externe Programme haben.

- **Ausgezeichnete Performance**

  Durch die Übermittlung gesamter Programmblöcke zum DB Server, wird eine Reduktion der Netzwerkbelastung erzielt. PL/SQL kann sowohl klientenseitig in Oracle Applikationen, als auch im Oracle Datenbankserver verarbeitet werden. Dadurch kann bei Client Server Applikationen eine gezielte Lastverteilung erreicht werden.

- **Portabilität**

  Applikationen, die in PL/SQL geschrieben wurden, sind auf allen Betriebssystemen, auf denen Oracle läuft, lauffähig. Programm Bibliotheken können auf unterschiedlichen Umgebungen eingesetzt werden.

- **hohe Produktivität**

  PL/SQL ergänzt die Funktionalität nichtprozeduraler Werkzeuge wie Oracle Forms und Oracle Reports.

- **Integration in Oracle**

  Sowohl PL/SQL und Oracle basieren auf SQL. Daher werden keine separaten Schnittstellen benötigt.

2.4 Entwicklungswerkzeuge

2.4.1 Oracle Developer, Form Builder

Die klassische Oracle Forms [10] Anwendung verwendet die Client-Server-Architektur. Alternativ dazu wird im nächsten Abschnitt der Einsatz von Oracle Forms in einer Drei-Schicht-Architektur erläutert.

"Form Builder" ist ein objektorientiertes Entwicklungswerkzeug zur Erstellung von Anwendungen in der Form von Masken. Diese Masken ermöglichen Benutzern einen Zugang zu Daten, die in einer Datenbank gespeichert sind. Form Builder Module sind eigenständige Programme, in die auch Oracle Reports (Berichterstellung) und Oracle Graphics integriert werden können.

Oracle Form Builder Hauptkomponenten

Form Builder Anwendungen bestehen aus drei Arten von Modulen:

- Formsmodule

  sind Ansammlungen von Objekten, die einem Benutzer interaktiv die Manipulation der Daten in Datenbanktabellen ermöglichen. Sie bilden den Hauptbestandteil einer Form Builder Anwendungen.

- Menüs

  Ein Menümodul besteht aus Menüs und Menücode. Durch Menüs, werden Funktionen der Anwendung durchgeführt.

- Bibliotheken.

  Ein Bibliotheksmodul ist eine Ansammlung des Clientseitigem Programmcode, der von mehreren Anwendungen benutzt werden kann.

Grundsätzliche Entwicklungsschritte

Für jede Aufgabe, die ein Produkt unterstützen soll, sollte ein Ablaufdiagramm der Schritte angefertigt werden, die zur Durchführung der Aufgabe erforderlich sind. Erst wenn alle Aspekte einer bestimmten Aufgabe identifiziert sind, werden Möglichkeiten zur Optimierung deutlich. Sobald der gewünschte Ablauf einer Aufgabe festgelegt ist, werden die folgenden Punkte analysiert, damit die richtige Benutzeroberfläche zur Unterstützung dieser Aufgabe entwickelt werden kann:

- Welche zusammengehörigen Informationen werden zur Durchführung der Aufgabe benötigt? Welche Informationen muß der Benutzer ignorieren?
- Wie häufig werden Daten für den Bildschirm benutzt, und wie umfangreich sind diese Daten?
- Welche Objekte sind am geeignetsten? Wie wird eine möglichst effiziente Dateneingabe ermöglicht?
- Welche Einweisung braucht der Benutzer für bestimmte Aufgaben an der Maske?
- Was geschieht, wenn ein Benutzer einen Fehler bei einem bestimmten Schritt macht oder versucht, einen Schritt zu umgehen? Müssen die Schritte der Aufgabe folgerichtig erfolgen oder können sie parallel ausgeführt werden?
- Welche Entscheidungen muß der Benutzer während der Ausführung der Aufgabe fällen? Was geschieht, wenn diese Entscheidungen nur in Sonderfällen erforderlich sind?
- Mit welchen anderen Werkzeugen sollte der Benutzer vertraut sein? Was sind die Erwartungen der Benutzer an Ihr Produkt basierend auf diesen anderen Werkzeugen?

2.4.2 Webforms

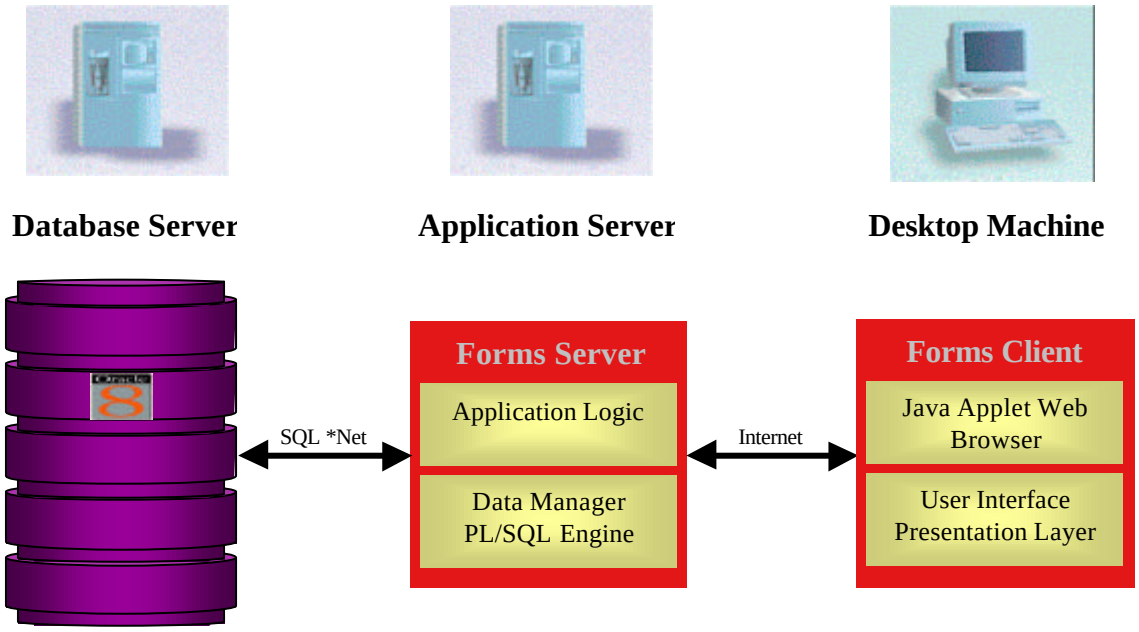

Abbildung 7: Drei Schicht Architektur Webforms.

Der Oracle Developer Server bietet eine Three Tier Architektur (siehe Abbildung 7), die dem Benutzer die Ausführung von komplexen Oracle Forms- und Reportsapplikeationen mit Einbettung von Graphics im Web ermöglicht. Clientseitig wird in einem Browser (zB. Netscape Navigator oder Internet Explorer) ein Plugin (Oracle Jinitiator) ausgeführt, das wiederum ein Applett, das für jede Sitzung vom Application Server zur Verfügung gestellt wird bedient. Dadurch erfüllt der clientseitige Rechner nur die Präsentationsebene. Die gesamte Applikationslogik verarbeitet der Applikation Server in der Mittelschicht.

Vorteile der Webforms-Drei-Schicht-Architekturen:

- Weiterverwendung bestehender Applikationen.
- Unveränderte Applikationsentwicklung: ohne Rücksichtnahme auf webspezifische Details.
- Geringerer Wartungsaufwand: da aufwendige Installation nur auf zentralisierten Applikationsservern notwendig ist.

- "Thin Client" Architektur: Clientseitig wird nur ein Web Browser, auf dem ein Java Runtime Environment als Plugin fungiert, benötigt. Dieses verringert Prozessor- und Speicheranforderungen für die Desktop Maschinen der Endbenutzer.
- Plattformunabhängigkeit: Die einzige systemspezifische Installation ist der Web Browser mit dem Java Runtime Environment.

2.4.3 PL/SQL Webtoolkit

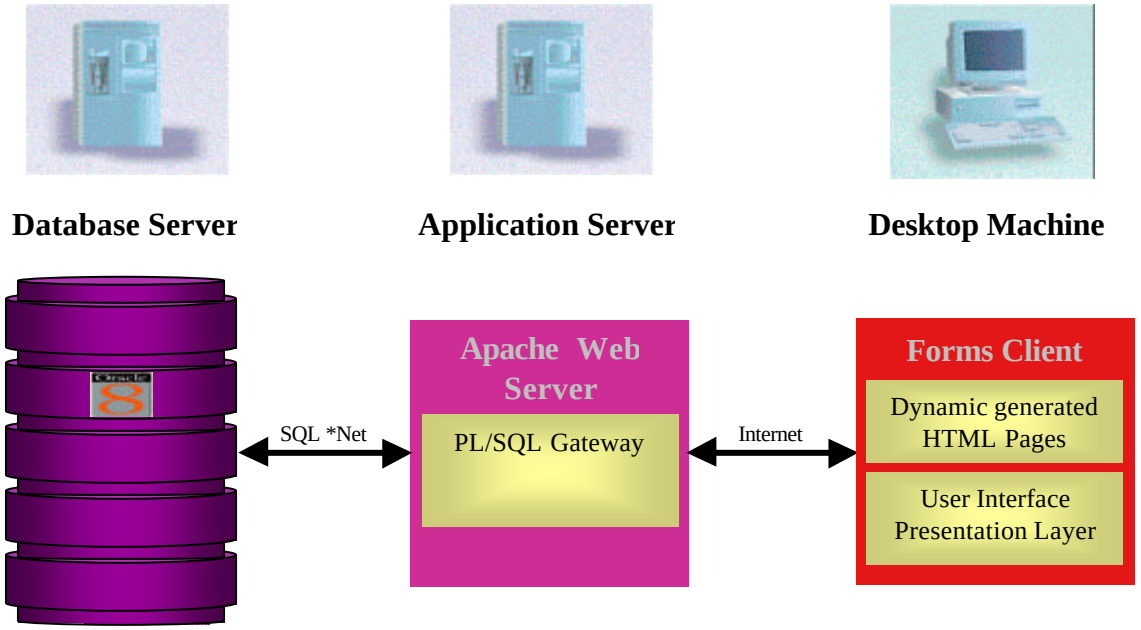

Abbildung 8: Struktur einer Webapplikation mit dem PL/SQL Gateway.

Das PL/SQL Webtoolkit [11] besteht aus PL/SQL Programmpaketen, die auf HTTP requests reagieren und dynamische HTML Seiten, beispielsweise aus dem Datenbestand einer Datenbank, generieren. Die Applikationslogik wird durch den Datenbankserver verarbeitet. In der Zwischenschicht zwischen Client und Server befindet sich ein PL/SQL Gateway im Webserver (siehe Abbildung 8), das die Client Requests an das RDBMS weiterleitet bzw. dynamisch generierte Webseiten aus dem RDBMS an den Client übermittelt. Der wesentliche Unterschied zur vorhin genannten Webforms Methodik besteht darin, daß kein Plugin installiert werden muss, da der Datenaustausch ausschließlich in HTML (mit optional eingebundenem JavaScript) erfolgt. Einschränkungen ergeben sich jedoch bei der Oberflächengestaltung sowie der

inhärenten Logik der HTML Masken gegenüber Webforms. Der clientseitige Rechner erfüllt im Wesentlichen nur mehr die Präsentation der Daten und einfache HTML Formularfunktionalität.

2.5 Die Entwicklung von YPL

Der chronologischen Ablauf des Projekts.

- Modellbildung

  Die Recherche über den Ablauf der Experimente im molekularbiologischen Labor. Ein wesentlicher Teil dabei ist die Auseinandersetzung mit der Terminologie der Molekularbiologie.

- Bildung eines Entity Relationship Modells anhand der realen Abläufe und Gegebenheiten im Labor.
- Erstellung der Datenbankobjekte, Implementierung des Schemas auf einer Oracle 8 Datenbank.
- Entwicklung der Anwendungen
  - Erstellung einer Multiform Anwendung in Oracle Forms 6 (Client/Server Modus) mit einer Benutzerverwaltung zur Eingabe von Experiment- und Bilddaten in die Datenbank zur Bestimmung der Daten, die im Internet veröffentlicht werden sollen und zur spezifischen Abfrage des Datenbestandes.
  - Anbindung der Datenbank an das Internet anhand einer Webforms Applikation.
  - Anbindung der Datenbank an das Internet in Form einer dynamischen HTML-Anwendung mit Apache Webserver, Oracle Internet Application Server und PL/SQL Webtoolkit mit einfachen Abfragemöglichkeiten bezüglich der Gen- bzw. Proteinbezeichnung und deren Lokalisierung und gen- bzw. proteinspezifischen Links zu den Datenbanken YPD, SGD und MIPS.
  - Erstellung eines PL/SQL-Packages das den Import von Daten und Bildern aus einer Tabelle in das relationale Datenmodell ermöglicht.

3 Ergebnisse

3.1 Das Schema

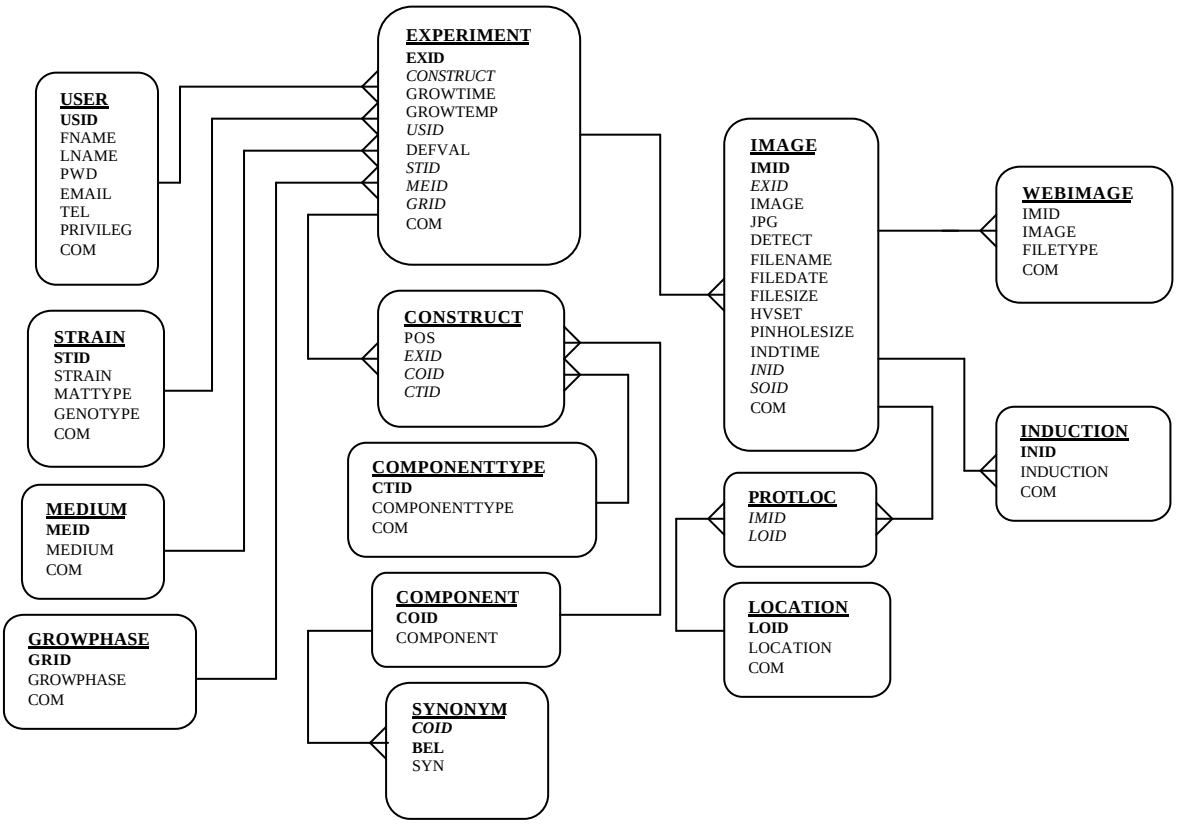

Abbildung 9: Der Kern des Schemas [12] zur Erfassung der Bild- und Experimentdaten.

Die Schlüssel der Tabellen sind mit Ausnahme der Benutzertabelle referentielle Schlüssel. Alle Benutzer, die berechtigt sind, Daten einzugeben, werden in einer Benutzerverwaltung erfaßt. Jedem Benutzer können vorgegebene Berechtigungen erteilt werden. Die zentrale Entität ist das Experiment. Ein Experiment umschreibt die in der Einleitung beschriebene gentechnische Transformation von Hefezellen. Das vorläufige Ergebnis des Experiments ist ein Objektträger, auf dem Hefezellen aufgebracht sind, von denen Schichtbilder im Mikroskop aufgenommen werden. Zu jedem Experiment wird das genetische Konstrukt (Anordnug der Gene in der transformierten DNA und Angabe deren experimenteller Funktionalität), sämtliche Synonyme der Gene, der Stamm, das Wachstumsmedium, die

Temperatur und die Wachstumphase erfaßt. Jedes Experiment kann mehrere Bilder enthalten. Anhand dieser Bilder wird die Lokalisierung des markierten Proteins in der Hefezelle festgestellt. Die Bilder (TIFF Format) werden in einer eigenen Entität erfasst. Um diese beiden zentralen Entitäten gruppieren sich weitere Entitäten. In diesem Schema (siehe Abbildung 9) werden alle Experimentdaten sowie Benutzerdaten gespeichert. Jedem Originalbild können mehrere unterschiedlich modifizierte Bilder für die Webveröffentlichung zugewiesen werden.

Das Schema wurde auf einem Oracle 8.1.6.0.0 Server unter Compaq Tru64 UNIX V5.1 auf einem Compaq Alpha Server implementiert.

3.2 Datenbankverwaltungsmasken

3.2.1 Die Loginmaske

Die Anmeldung erfolgt durch Datenbankname, Username und ein verschlüsseltes Passwort, die beim Maskenstart als Parameter an die Maske übergeben werden. Das verschlüsselte Passwort wird vor dem Login am Schema dekodiert. Die Anmeldung des Benutzers erfolgt unmittelbar vor dem Aufruf der Hauptmaske bzw. der Benutzeradministrationsmaske. Aus der Loginmaske kann die Hauptmaske oder (bei entsprechender Berechtigung) die Benutzeradministrationsmaske gestartet werden. Zum Login wird ein Benutzerkürzel, das sich aus dem ersten Buchstaben des

Vornamens und den ersten beiden Buchstaben des Nachnamens zusammensetzt verwendet. Jeder User kann in der Loginmaske sein Passwort ändern.

3.2.2 Die Hauptmaske

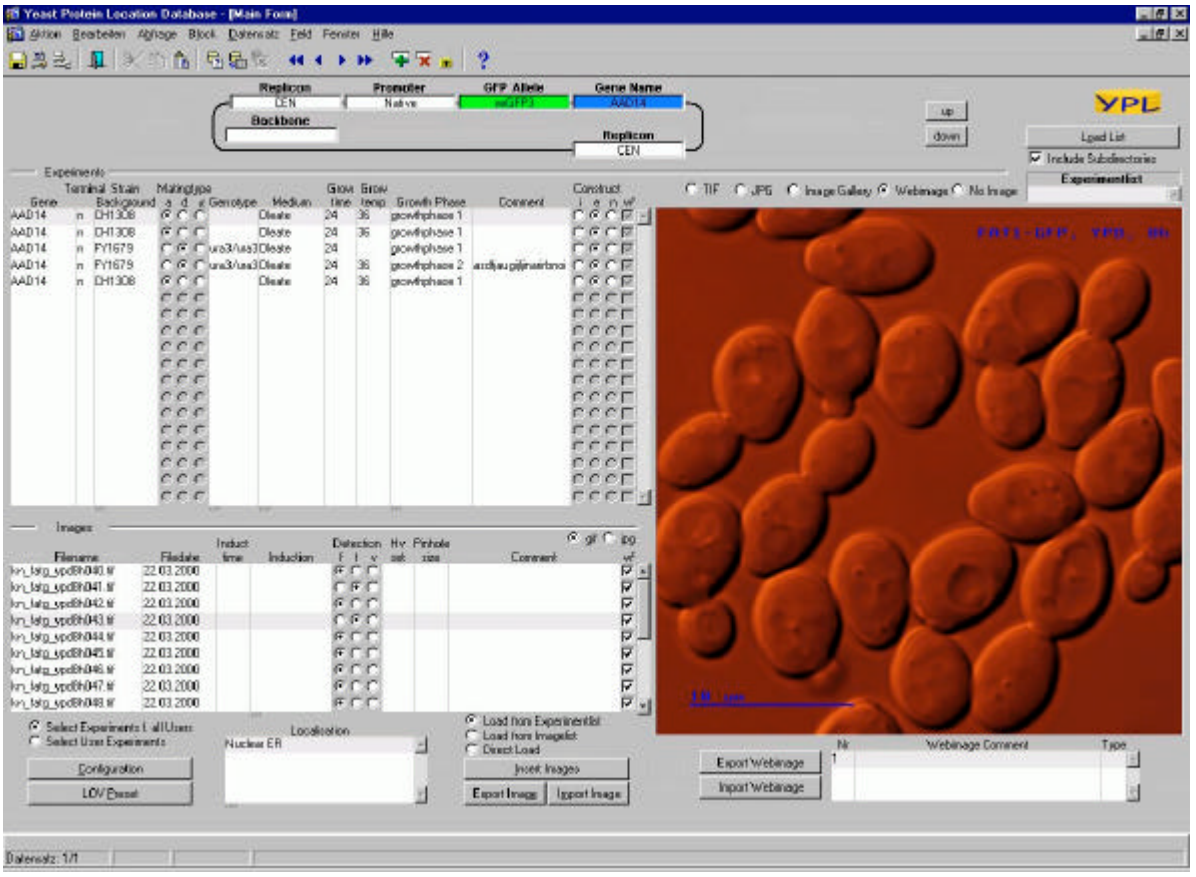

Abbildung 10: Die Hauptmaske zur Verwaltung der Daten.

Die Hauptmaske (siehe Abbildung 10) besteht aus 8 Bereichen.

- Konstruktdaten

  Da drei unterschiedliche Konstrukte (integral, episomal und vital stain) mit unterschiedlichen Hintergrundbildern dargestellt werden, erfolgt die Anordnung der Textfelder auf einer nur teilweise sichtbaren Leinwand, die hinter einem Fenster der Hauptmaske vertikal verschoben wird. Die Textfelder werden dem Konstrukt entsprechend auf dieser Leinwand angeordnet. Zur Auswahl der unterschiedlichen Ausgangskonstrukte sind neben der Konstruktdarstellung zwei Schaltflächen. Unmittelbar nach der Erzeugung eines neuen Experimentdatensatzes kann unter den für den Benutzer vorhandenen Ausgangskonstrukten eines ausgewählt werden. Daten können ausschließlich aus Wertelisten (LOV List Of Values) eingefügt, geändert oder gelöscht werden. Eine Werteliste ist ein Popup-

Fenster mit Bildlaufleiste, das dem Endbenutzer eine ein- oder mehrspaltige Auswahlliste anzeigt.

Wertelisten bieten folgende Funktionalität:

- Wertelisten können mit Daten direkt aus einem Select Statement befüllt werden.
- Wertelisten können auf Anforderung des Benutzers angezeigt werden (sofern eine Werteliste verfügbar ist), wenn der Endbenutzer zu einem Textobjekt mit einer zugehörigen Werteliste navigiert. Sie können auch programmgesteuert unabhängig von einem bestimmten Textobjekt angezeigt werden.
- Mit den automatischen Reduzierungs- und Suchfunktionen von Wertelisten können Endbenutzer bestimmte Werte suchen.
- Werte in Wertelisten, die vom Endbenutzer ausgewählt werden, können je nach den angegebenen Rückgabeobjekten, Formobjekten zugewiesen werden.
- Beim Entwurf kann eine Werteliste einem oder mehreren Textobjekten in der Form zugeordnet werden.
- Werte von Wertelisten werden aus Datensatzgruppen abgeleitet.

- Experimenttabelle

  In der Experimenttabelle wird Datenmanipulation in Textfeldern, die am unteren linken Ende mit LOV gekennzeichnet sind, durch Wertelisten vorgenommen.

- Bilddatentabelle

  Die Bilddatentabelle ist das Detail der Experimenttabelle.

- Lokalisationstabelle

  Die Lokalisationstabelle ist ein Detail zur Bilddatentabelle.

- Bilder

  Sämtliche geladenen Bilder werden als LOB`s in der Datenbank gespeichert. Als weitere Option stünde die Speicherung als BFILES (Pointer in der Datenbank zeigt auf die Bilddatei im Dateisystem) zur Verfügung. Die erste Variante wurde implementiert, da auch die Bilder in den Transaktionsprozess integriert sind und ein geschlossenes Backup der Daten einschließlich der Bilder erstellt werden kann.

Bei der Bildübertragung zwischen Server und Client wird das Netzwerk durch große Datenmengen belastet. Bei leistungsfähigen Netzwerken können immer unkomprimierte TIF Dateien angezeigt werden. Bei weniger leistungsfähigen Netzwerken besteht die Möglichkeit, bei Datenabfragen keine Bilder zu übertragen oder hochkomprimierte JPG Bilder anzuzeigen. Ähnlich wie bei dem Konstruktdatenbereich, wird die Darstellung der unterschiedlichen Quellbilder sowie der Webbilder auf einer horizontal verschiebbaren Leinwand bewerkstelligt. Bei der Anzeige der Webbilder wird unterhalb des Bildes eine Tabelle angezeigt, die die Eingabe eines Kommentars zu jedem Bild ermöglicht.

Die gleichzeitige Darstellung von 9 Bildern eines Experiments in einer Gallerie ist möglich.

- Bilderladebereich

  In der Regel haben die Bilder eine Größe von 512x512 Pixels, eine Farbtiefe von 8 Bit und einen Speicherbedarf von ca. 250 kByte. Die Größe eines Bildpunktes entspricht etwa 60 nm am Mikroskopobjektträger. Die Bilder werden aus lokalen Laufwerken oder Netzwerklaufwerken geladen. Der Benutzer kann ein Verzeichnis angeben in dem sämtliche Unterverzeichnisse nach TIF Bildern durchsucht werden. Die Dateinamen der in den Unterverzeichnissen gefundenen TIF Dateien werden nach bestimmten Kriterien verglichen und Experimenten zugeordnet. Auf diese Weise können aus einer angezeigten Tabelle Experimente mit voreingestellten Ausgangswerten und gleichzeitigem Laden aller zugehörigen Bilder erstellt werden. Anschließend werden die Ausgangswerte entsprechend korrigiert. Die Bilder werden von Forms unkomprimiert an die Datenbank übermittelt und dort gespeichert.

  Die Bilder die im Internet angezeigt werden, werden optional ins GIF- oder JPG-Format konvertiert und anschließend über Telnet und ein SSP Protokoll in das Dateisystem des Datenbankservers übermittelt. Von dort werden die Bilder in die Datenbank importiert.

Sowohl Originale Bilder wie auch Webbilder können in externe Programme für die Bildbearbeitung exportiert und anschließend wieder importiert werden.

- Konfigurationsmaske

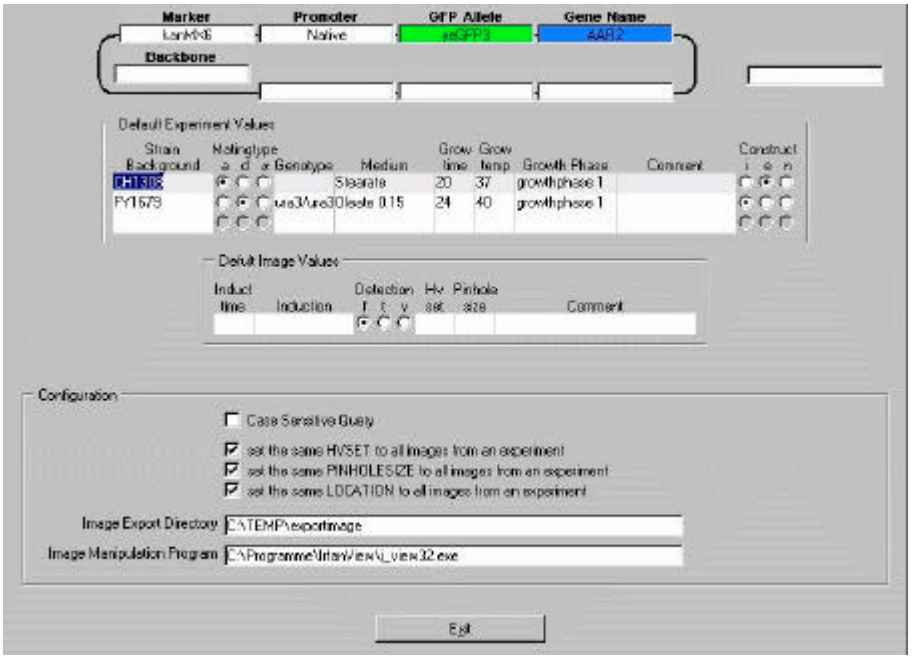

Abbildung 11: Die Konfigurationsmaske zur Einstellung benutzerspezifischer Parameter.

Benutzerspezifische Angaben können in der Konfigurationsmaske (siehe Abbildung 11) eingestellt werden. Jeder Benutzer kann eine beliebige Anzahl von Ausgangsdatensätzen für Experimente definieren. Unmittelbar nach der Erstellung eines neuen Experimentes können die Ausgangswerte eines vordefinierten Experimentes zugeteilt werden. HV Set (Laserintensität des Konfokalmikroskops), Pinholesize (Blende der Mikroskopoptik) und Lokalisation unterscheiden sich nur in seltenen Fällen zwischen den Bildern eines Exeriments. Dafür kann in der Konfigurationsmaske eingestellt werden, ob bei Eingabe eines Wertes allen Bildern dieser Wert zugeordnet werden soll oder nicht. Ebenso kann umgeschalten werden, ob Groß und Kleinschreibung bei der Abfrage berücksichtigt wird oder nicht.

3.2.3 Die Stammdatenmaske

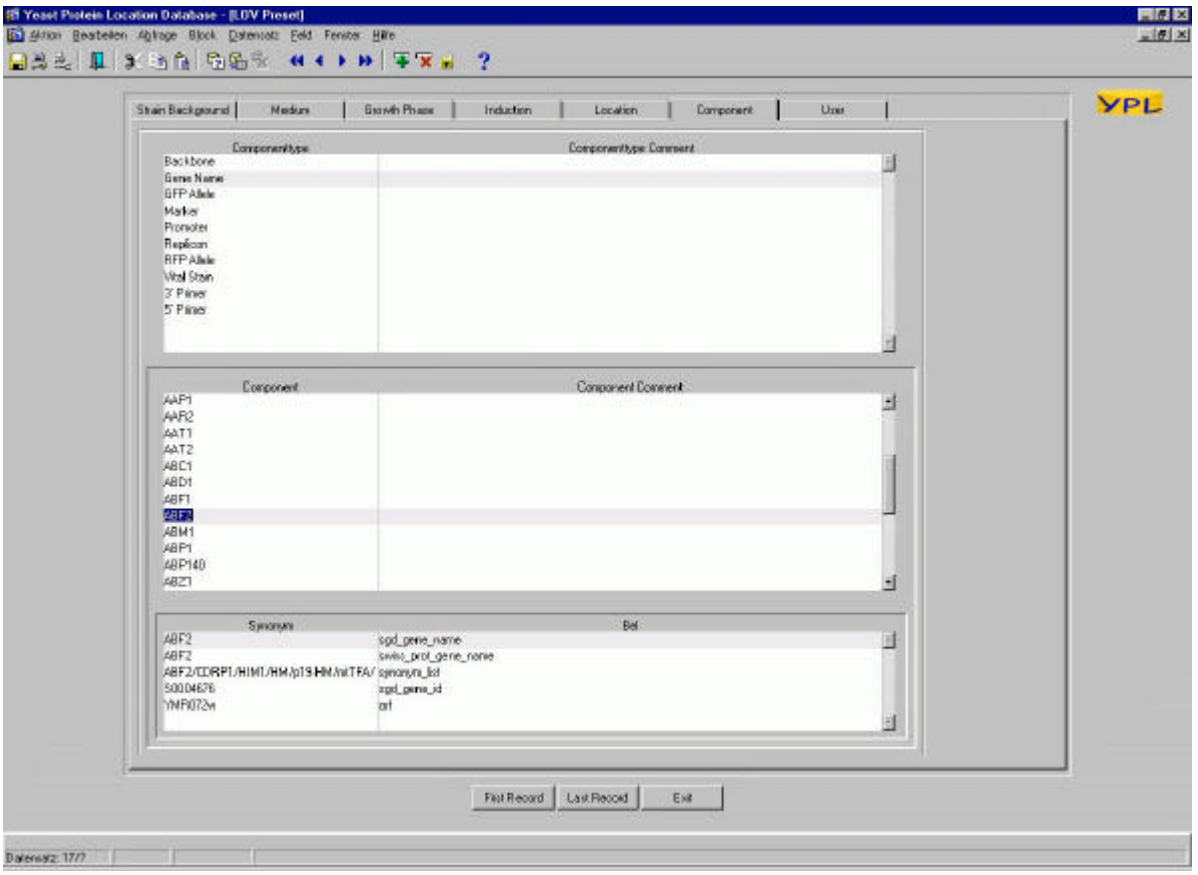

Abbildung 12: Die Stammdatenmaske zur Festlegung einheitlicher Bezeichnungen.

Berechtigte Benutzer können diese Maske (siehe Abbildung 12) aus der Hauptmaske heraus in einer neuen Datenbanksitzung starten. In dieser Maske werden Stammdaten wie mögliche Lokalisationen, Wachstumsmedien, Zellstämme etc. definiert.

3.3 Internetzugang zur Datenbank

3.3.1 Die Webformsmaske

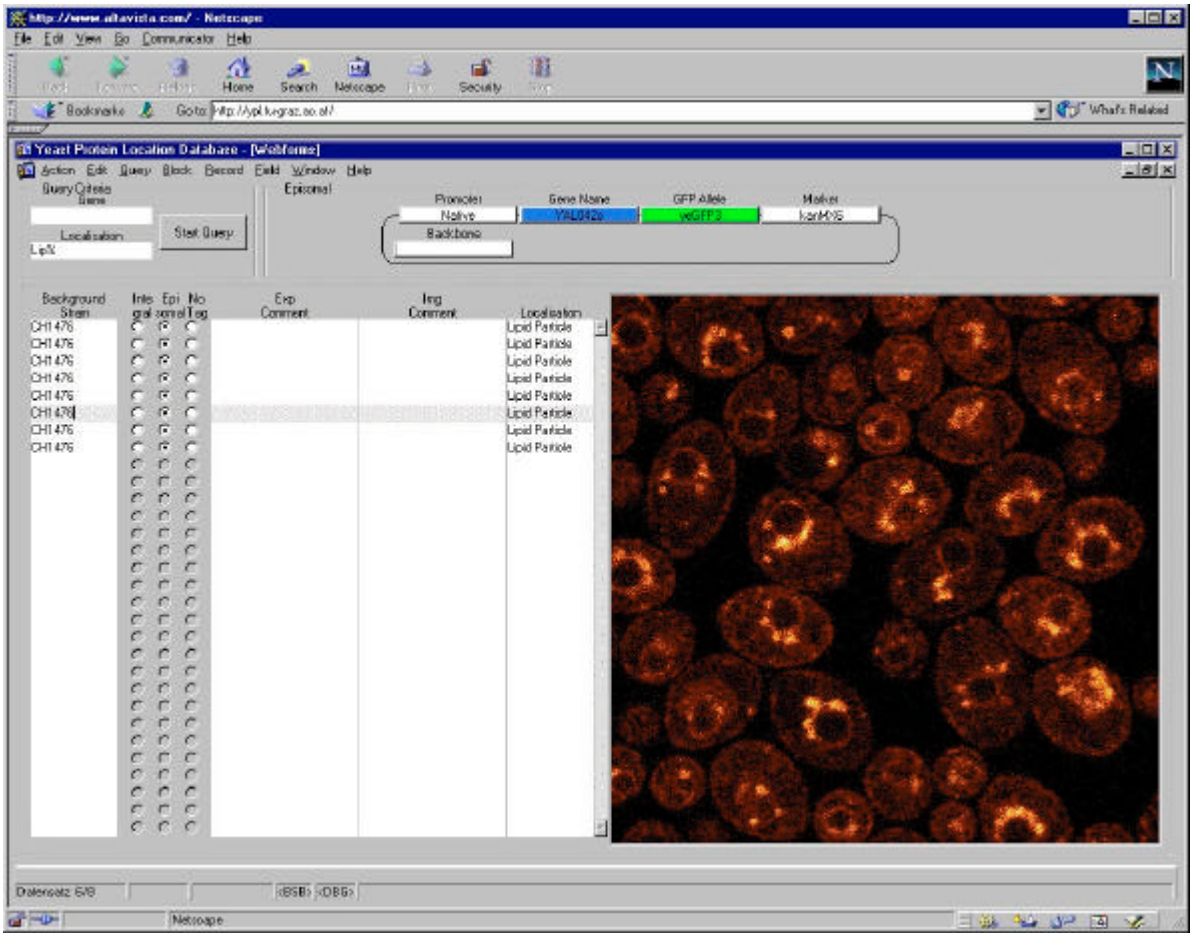

Abbildung 13: Die Webformsmaske zur Abfrage der Daten nach Gen oder Lokalisierung.

Im ersten Anlauf wurde der Internetzugang durch eine Webformsmaske (siehe Abbildung 13) realisiert, die clientseitig als Java Applett im Webbrowser ausführbar ist. Aufgrund von Problemen dieser Technologie auf Macintosh Rechnern und der Notwendigkeit eines einmaligen Downloads eines Plugins (Oracle Jinitiator ca. 8MB ist ein Java Runtime Environment mit speziellen Funktionalitäten für Oracle Developer Anwendungen), fand diese Lösung keine Akzeptanz. Um die genannte Problematik zu umgehen, wurde eine HTML Webapplikation entwickelt, die zur Ausführung keine Plugins benötigt.

3.3.2 Die Dynamische HTML Webmaske

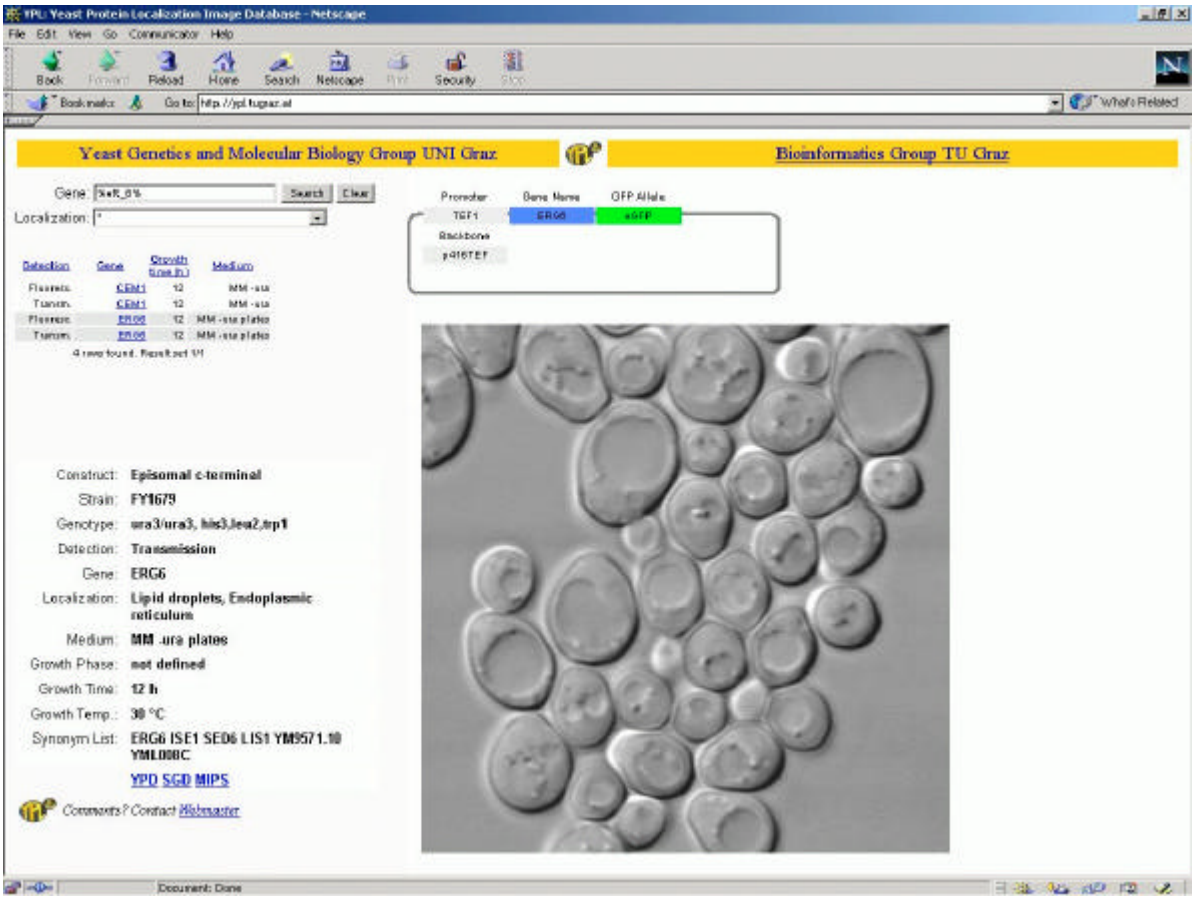

Abbildung 14: Die dymanisch aus dem PL/SQL Webtoolkit generierte HTML Maske.

Der Internetzugang über die dynamische HTML Webmaske (siehe Abbildung 13) unter http:/ypl.tugraz.at bietet die Abfrage nach zwei Suchkriterien:

- Genname:

  Sämtliche gängigen Genbezeichnungen können als Suchbegriff verwendet werden. Weiters können Wildcards (“*“ für eine beliebige Anzahl beliebiger Zeichen und “?“ für ein beliebiges Zeichen) verwendet werden

- Lokalisation:

  Die Proteinlokalisation als Suchkriterium kann aus einer Werteliste ausgewählt werden.

Das Suchergebnis wird in einer Tabelle angezeigt. Der Genname in dieser Tabelle ist als Link ausgeführt und zeigt zu experimentellen Details.

3.3.3 Die Datenimportfunktionen

Um bereits bestehende Sammlungen von elektronischen Daten durchgeführter Experimente in die Datenbank importieren zu können, wurde ein PL/SQL Package entwickelt. Nach der Formatierung der vorhandenen Daten in einer Excel Tabelle werden die Schlüssel generiert und die Daten entsprechend in das Datenbankschema eingelesen. In diesem Fall kann zeitaufwendige Eingabevorgang über die Hauptmaske umgangen werden.

4 Diskussion

In dieser Arbeit wurde die Datenbank YPL zur Archivierung von Experimentdaten und Bildern entwickelt. Sämtliche durch die Molekularbiologie Gruppe der TU Graz anfänglich und während der Entwicklungsphase gestellten Anforderungen an die Datenbank sowie die Applikationen wurden vollständig implementiert.

YPL bietet folgende Funktionalitäten:

- Verwaltung von Benutzern und deren Berechtigungen, die zum Eingeben, Änderung und Löschen von Daten berechtigt sind.
- Stapelweiser Import von Bildern in die Datenbank.
- Festlegung von Stammdaten.
- Einrichtung von Benutzerprofilen.
- Umfangreiche Abfragemöglichkeiten in der Client Server Umgebung.
- Import von Daten aus einer entsprechend formatierten Tabelle.
- Internetzugang zu freigegebenen Experimentdaten und Bildern.
- Abfragemöglichkeiten nach Gen bzw. Protein oder Lokalisation in der Webapplikation.

Besonderer Wert wurde auf die Möglichkeit einer schnellen Dateneingabe gelegt. Durch benutzerspezifische Konfiguration der Ausgangswerte, Tastenkombinationen für häufig verwendete Funktionen, teilweise automatische Übernahme von Werten und dem stapelweisen Import von Bildern wird der Eingabeaufwand auf ein Minimum reduziert. Ein weiterer Schwerpunkt wurde auf Detaillierte Abfragemöglichkeiten gelegt. Im Konstrukt können auch von der Position- bzw. Reihenfolge abhängige Abfragen durchgeführt werden. Durch die Vergabe von Benutzerprofilen können unterschiedliche Berechtigungen für die Datenmanipulation vergeben werden.

Die Datenbank wurde für die Erfassung der Lokalisationsexperimente des gesamten Hefeproteoms (ca. 6200 unterschiedliche Proteine) ausgelegt. Der Speicherbedarf dafür wird voraussichtlich zwischen 15 und 20 Gbyte betragen. Eine Ansammlung anschaulicher Referenzbilder über Proteinlokalisationsmuster ist in einem abgeschlossenen Bereich im Web verfügbar. Zugang zu anderen Datenbanken (MIPS, YPD und SGD), welche proteinspezifische Informationen enthalten, die nicht in YPL verfügbar sind werden durch Links in dem Suchergebnis der YPL Webmaske angeboten.

Perspektiven:

- Künftige Versionen der Proteinlokalisationsdatenbank werden Videosequenzen und Dreidimensionale Präsentationstechniken zur besseren Veranschaulichung anbieten können.
- An Zellen anderer Organismen (Mensch, Maus) werden fluoreszenzmikroskopisch Proteinlokalisationen untersucht.
- Eine Datenbank, die fluoreszenzmikroskopische Lokalisationsuntersuchung mehrerer verschiedenfarbig markierter Proteine erfassen könnte, währe in ähnlicher Weise realisierbar.

Durch YPL können Wissenschaftler weltweit auf Daten aus den Lokalisierungsexperimenten zugreifen und dadurch einen weiteren Aspekt der komplexen intrazellulären Mechanismen in ihre Forschungen mit einbeziehen.

Literaturverzeichnis

[1] Codd, E.F.; "A Relational Model for Large Shared Data Banks", Comm. ACM, Vol. 13, No. 6, June 1970, pp 377-387

[2] Dave Ensor & Ian Stevenson Oracle8 Design Tips O'Reilly

[3] David Kreines & Brian Laskey Oracle Database Administration: The Essential Reference O'Reilly

[4] Ding, D.Q., Tomita, Y., Yamamoto, A., Chikashige, Y., Haraguchi, T., and Hiraoka, Y. (2000) Genes Cells, 5, 169-190.

[5] Goffeau, A., Barrell, B. G., Bussey, H., Davis, R. W., Dujon, B., Feldmann, H., Galibert, F., Hoheisel, J. D., Jacq, C., Johnston, M., et al. (1996) Science, 274, 546, 563-547.

[6] Gustafsson, M. G., Agard, D. A. and Sedat, J. W. (1999) I5M: 3D widefield light microscopy with better than 100 nm axial resolution, J Microsc. 195, 10-16.

[7] Haßek, J. and Streiblová, E. (1996) Fluorescence microscopy methods, Methods Mol Biol. 53, 391-405.

[8] Karl-Heinz Hauer Der Oracle-Entwickler Addison Wesley

[9] Oliver, S. (1996) Trends Genet, 12, 241-242.

[10] Oracle Developer Form Builder RELEASE 6.0 http://otn.oracle.com/docs/products/forms/content.html

[11] Oracle PL/SQL Webtoolkit http://www-wnt.gsi.de/oragsidoc/OAS/plsql_r.pdf

[12] Oracle8 Enterprise Edition Release 8.1.7 http://www-wnt.gsi.de/oragsidoc/doc_817/index.htm

[13] Ploger, R., Zhang, J., Bassett, D., Reeves, R., Hieter, P., Boguski, M. and Spencer, F. (2000) Nucleic Acids Res, 28, 120-122.

[14] Steven Feuerstein with Bill Pribyl Oracle PL/SQL Programming, 2nd Edition O'Reilly

[15] Uwe Herrmann / Dierk Lenz / Günter Unbescheid Oracle8 für den DBA Verwalten, optimieren, vernetzen Addison Wesley

Index

5 Anhang