Towards an efficient management of biological data

Inaugural – Dissertation

zur

Erlangung des Doktorgrades der

Mathematisch-Naturwissenschaftlichen Fakultät

der Heinrich-Heine-Universität Düsseldorf

vorgelegt von

Jochen Kohl

aus Düsseldorf

April 2008

Aus dem Institut für Informatik
der Heinrich-Heine-Universität Düsseldorf

Gedruckt mit der Genehmigung der Mathematisch-Naturwissenschaftlichen
Fakultät der Heinrich-Heine-Universität Düsseldorf

Referent: Prof. Dr. Arndt von Haeseler
Korreferent: Prof. Dr. Martin Lercher

Tag der mündlichen Prüfung: 30.04.2008

Danksagung

Bedanken möchte ich mich zuallererst bei meinem Betreuer Arndt von Haeseler für sein Vertrauen und die Unterstützung, ohne die ich nicht so weit gekommen wäre. Und dann natürlich bei der gesamten Arbeitsgruppe, den Häslis, auf die man immer zählen konnte, und ein gutes Arbeitsklima schufen; im Besonderen bei Ingo P., Thomas S. und L., Nicole, Achim, Ricardo, Stefan, Simone, Tanja und Andrea. Auch dem gesamten Ontoverse-Team; im Besonderen Katrin, Dominic, Indra. Desweiteren danke ich Martin Lercher für die Begutachtung meiner Arbeit und wünsche ihm viel Erfolg in Düsseldorf. Für die finanzielle Unterstützung danke ich der DFG und dem BMBF.

Im Besonderen möchte ich danken:

Meinen Eltern und meinem Bruder, die immer an mich geglaubt haben und Zeit für mich hatten.

Meinem großen und kleinen Schatz, die ich immer lieben werde.

Schluffi, für ehrlich versoffene Nächte und die guten Gespräche beim Kaffee [!!KillerBiene!!].

Ingo, nicht nur für die nächtelangen Korrekturen, sondern für seine Freundschaft.

Achim, der mir die Geheimnisse des Oracles offenbart hat.

Stefan, der Herr der Bäume.

Lutz, für interessante Diskussionen.

Beim Biokolleg PartyPöbel, den drei Cs, Kocky, Stobbe, Kalles und Herrn Alteriiii. Ich sage nur: *Ergo bibamus.*

Bei den guten alten Freunden Andreas, Jörg, Helmut und Brennie.

Alle, die mich durchs Studium begleitet haben.

Zum Schluß noch bei allen, die ich vergessen habe.

Publications

Parts of this thesis have been published in the following articles and conference proceedings:

- Jochen Kohl, Ingo Paulsen, Thomas Laubach, Achim Radtke, Arndt von Haeseler. (2006) HvrBase++: a phylogenetic database for primate species. Nucleic Acids Res., **34**, D700-D704.

Other publications and conference proceedings:

- Jochen Kohl and Arndt von Haeseler. (2005) Book Review: Perl Programming for Biologists by D. C. Jamison. Biometrics, **61(1)**, 320-320
- Benjamin Kilian, Hakan Özkan, Jochen Kohl, Arndt von Haeseler, Francesca Barale, Oliver Deusch, Andrea Brandolini, Cemal Yucel, William Martin, Francesco Salamini. (2006) Haplotype structure at seven barley genes: relevance to gene pool bottlenecks, phylogeny of ear and site of barley domestication. Mol Gen Genomics, **276**, 230-241
- Ingo Paulsen, Dominic Mainz, Katrin Weller, Indra Mainz, Jochen Kohl, Arndt von Haeseler. (2007) Ontoverse: Collaborative Knowledge Management in the Life Sciences Network. In: Proceedings of the Germany eScience Conference 2007, Max Planck Digital Library, ID 316588.0.
- Benjamin Kilian, Hakan Özkan, Oliver Deusch, Siglinde Effgen, Andrea Brandolini, Jochen Kohl, William Martin, Francesco Salamini (2007) Independent Wheat B and G Genome Origins in Outcrossing *Aegilops* Progenitor Haplotypes. Mol. Biol. Evol., **24(1)**, 217-227
- B. Kilian, H. Özkan, A. Walther, Jochen Kohl, T. Dagan, F. Salamini, and W. Martin (2007) Molecular Diversity at 18 Loci in 321 Wild and 92 Domesticate Lines Reveal No Reduction of Nucleotide Diversity During *Triticum monococcum* (Einkorn) Domestication: Implications for the Origin of Agriculture. MBE., Advance Access published Spetember

Abstract

This thesis focuses on the management of biological data and is divided into two parts. The first part deals with the extension and enhancement of a mitochondrial database, called HvrBase. This database handles DNA sequences from two regions of the mitochondrial genome, *hypervariable region* I and II, and corresponding information required for phylogenetic studies of human evolution. To follow trends in evolution history the structure of HvrBase is re-designed to add further genetic loci and to provide new features, like a dynamic tree reconstruction and visualization tool. The improved version is called HvrBase++.

Based on the experiences made with HvrBase++, a general web application is developed to give biologists the opportunity to establish their own sequence collections without deeper knowledge about database design. The challenge, in contrast to the well defined and slowly changing HvrBase++, is that the application and the database design do not restrict and support scientists to define their own related sequence information. Hence, an RDF (resource description framework) like structure was implemented to solve this problem.

.

Contents

Chapter 1

Introduction

The growing amount of biological data makes it necessary to develop concepts of managing and exploring the data using current computer technologies. Biologists mostly manage sequences and corresponding sequence information with office programs. On the other hand professional managed sequence databases usually maintain biological data in relational database management systems (RDBMSs). One goal of the thesis is to improve biological data management for private collections by developing an application that is easily integrated into the workflow of biologists. This application minimizes the technical effort and opens the way for an efficient biological data management. RDBMSs are the state-of-the-art for data management and are utilized to reach the described goal.

A database management system is a piece of software that administrates database storage and access. The database itself is only the collection of data. An RDBMS is a special kind of DBMS that uses the relation model presented by F. Codd (1970, 1972, 1979) to manage data and is the commonly used type of DBMSs, like Oracle, MySQL or SQLite. Data is handled in tables (relations), which can be joined to generate a new table. Furthermore the view of a relation can be restricted. The combination and restriction of relations allow the representation of the same database in alternative forms depending on the task and not on physical storage. This prevents the time

consuming creation of different files with redundant information, which also reduce errors caused by redundant storage. RDBMSs are not commonly used in biology to manage private collections, caused by the handicap of getting familiar with these kind of technique. This handicap should be minimized by the developed application, called TreeDB. To develop such a general data management application the specialized mitochondrial database HvrBase is analyzed and redesigned to work out the general concepts and requirements.

Sequence databases can be roughly divided into general and specialized sequence databases. Generalized databases like GenBank[1] from the National Center for Biotechnology Information (NCBI) provide available sequences for loci and species. Currently, GenBank contains over 61 million publicly available sequences from more than 240,000 named organisms (Benson *et al.*, 2007). In contrast specialized databases are much smaller but focus on special kinds of data or questions and provide customized search tools.

For example HvrBase (Handt *et al.*, 1998) is a mitochondrial sequence database for human history studies. Its name was derived from the two loci *HVR-I* and *HVR-II* and the word database. *Hypervariable regions* (*HVRs*) are located in the non-coding region of the mitochondrial genome and have a high substitution rate with 7×10^{-8} per sites per year (Horai *et al.*, 1995). The human mitochondrial genome is maternal inherited and does not show recombination events which allow the use of simple models to reveal human evolutionary history. However, Mitochondrial sequences present only one perspective of the evolutionary process nowadays more and more Y-chromsomal and autosomal loci are analyzed (Torroni *et al.*, 2006). To follow this trend a new version of HvrBase, called HvrBase++, is designed. Moreover, a revised web interface is created to provide new intuitive searches and visualization features to integrate HvrBase++ into the scientists workflow.

[1] http://www.ncbi.nlm.nih.gov

Thesis outline

Background knowledge, like studying human evolution on the basis of mitochondrial sequences or relational database design is presented in Chapter 2. The extension and re-design of HvrBase is explained in Chapter 3. In this chapter first the existing version of HvrBase is analyzed to extract the requirements for the new HvrBase++. In the following the re-desig of the database and the web application is described. Chapter 4 describes the concepts and the implementation of TreeDB. Last but not least a general conclusion is provided in the last chapter.

Chapter 2

Background

This thesis deals with biological data management with emphasis on mitochondrial DNA (mtDNA) sequences. But also for data management it is vital to understand the biological background to improve the management process. Hence at the beginning of this chapter the function and genetics of mitochondria are briefly explained. Subsequently an overview of molecular phylogeny is given, which leads to a more detailed observation of human history based on mitochondrial sequences. Then the technical part introduces relational data management and aspects of web application design.

2.1 Functionality of mitochondria

Mitochondria are organelles in eukaryotic cells that provide 90% of cellular energy and are therefore indispensable for the development of higher organisms. The number of mitochondria in a cell varies among cell types. Those cells with a high energy turnover, such as muscle and nerve cells, contain thousands of mitochondria. The respiratory chain (oxidative phosphorylation) is the energy generating process and consists of the five protein complexes I-V. Only thirteen subunits of these complexes are synthesized by the mitochondrion protein synthesis apparatus (Sukernik *et al.*, 2002). The remaining proteins are encoded by nuclear genes, synthesized in

the cytoplasm and transferred into mitochondria (Shadel and Clayton, 1997). During oxidative phosphorylation a high-energy electron is passed along the electron-transport chain in the inner membrane. Released energy is used to pump protons out of the mitochondrion matrix, resulting in an electrochemical proton gradient. This proton-motive force drives *complex V* (*ATP synthase*) to synthesize *adenosine triphosphate* (ATP) from *adenosine diphosphate* (ADP) and phosphate. ATP releases that stored energy for work within the body.

Mitochondria as the major power supplier of the cell are linked to a wide range of degenerative diseases, preferentially affecting the central nervous system, heart, muscle, renal and endocrine systems (Sukernik *et al.*, 2002). The impairment of mitochondrial functions is related to mitochondria and nuclear mutations. Hence there is brisk interest of studying mtDNA variations in medical sciences. For example, a correlation between a T-to-C transition at nucleotide 16,189 in mitochondrial genomes with increasing insulin resistance and adult-onset *diabetes mellitus* is described in Liou *et al.* (2007). Other disease-causing mtDNA mutations arise preferentially on specific haplogroups, as a result of synergistic interaction between disease mutation and haplogroup polymorphisms. Like the *lebers hereditary optic neuropathy* (*LHON*), where primary three mtDNA mutations (3460A, 11778A, 14484C) interact with the Western Eurasian haplogroup J (Brown *et al.*, 2002).

2.2 Genome structure and mitochondrial genetics

Mitochondrial genomes are circular and range from 6,000 bp in *Plasmodium falciparum* (Feagin *et al.*, 1991) to more than 350,000 bp in some land plants like *Arabidopsis thaliana* with 366,924 nucleotides and 57 genes (Unseld *et al.*, 1997). Human mitochondrial genomes are around 16,569 bp. A complete mitochondrion genome was first sequenced by Anderson *et al.*

(1981). This sequence is known as Cambridge Reference Sequence (CRS). The nucleotide numbering of CRS is retained and starts in the middle of the control region (Fig. 2.1). The non-coding control region with 1,122 bp (16,024 bp-576 bp) contains both promoters and one origin of replication. Moreover, the two regions commonly used in population studies, the hypervariable region I (HVR-I) and the hypervariable region II (HVR-II), are located there[1]. The substitution rate of non-coding control region with 7.00×10^{-8} per site per year is higher than for coding sequences with 3.89×10^{-8} per site per year (Horai *et al.*, 1995). Coding sequences occupy the major part of the mitochondrial genome, where intercistronic regions only account for a total of 87 bp (Sukernik *et al.*, 2002). There are no introns in the 37 genes of the human mitochondrial genome:

- 13 protein-coding genes
  - 7 subunits of complex I (NADH dehydrogenase)
  - 3 subunits of complex IV (Cytochrome c oxidase)
  - 2 subunits of complex V (ATP synthase)
  - 1 subunit of complex III (Cytochrome b)
- 22 tRNAs
- 2 rRNAs (12S-/16S-rRNA)

The GC content of the two strands is distributed asymmetric. The guanine rich chain is called heavy chain, the other is called light chain. The non-unidirectional and asynchronous replications starts from the heavy strand's origin by forming a displacement loop (D-loop). The replication of the light strand starts when the replication of the heavy strand reaches the origin on the light strand.

[1]Within the scope of this thesis sequences belong to the hypervariable regions I and II if they are located between 16,001 bp and 16,408 bp or between 1 bp and 408 bp according to Anderson numbering

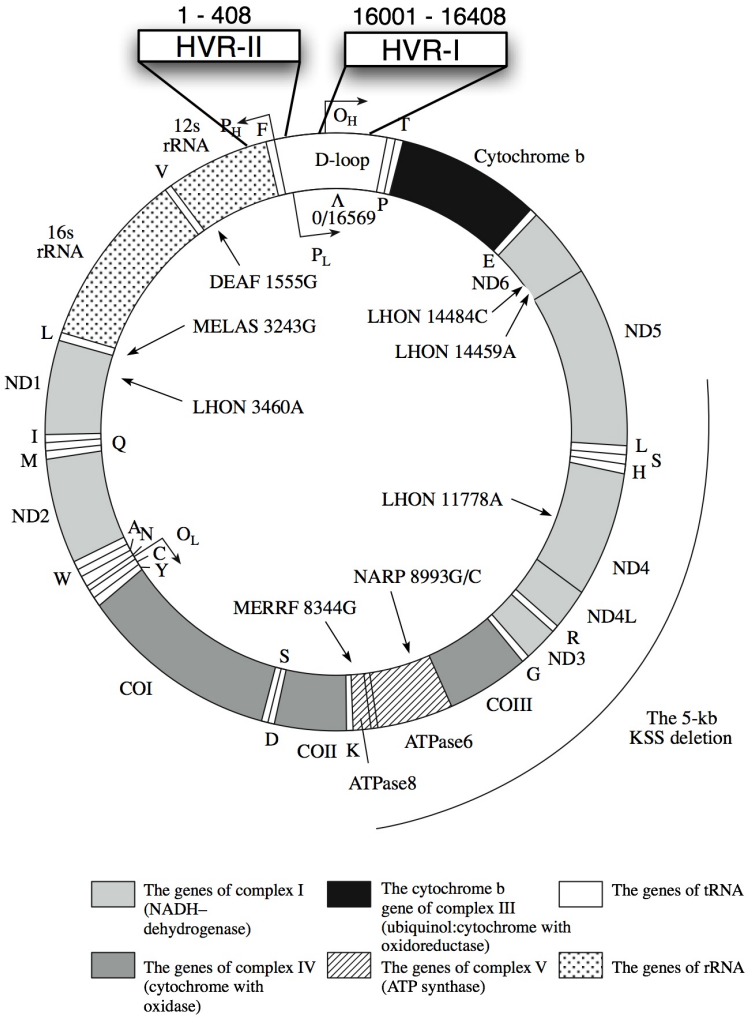


Figure 2.1: Map of the human mitochondrial genome (modified from SUKERNIK *et al.* (2002)). Positions of genes and common pathogenic mutations are indicated. Positions of HVR-I (16,001 bp-16408 bp) and HVR-II (1 bp-408 bp) as used in HvrBase are added to the map.

Each mitochondrion carries one to ten copies of its genome (Satoh and Kuroiwa, 1991). *Oocytes* carry around 100,000 mitochondria, with only one genome per organelle. Human sperms in contrast contain only between 50 to 75 mitochondria. During fertilization paternal mtDNA does normally not enter the offspring (Ankel-Simons and Cummins, 1996). A degradation mechanism of paternal mitochondria is discussed in (Sutovsky *et al.*, 2004). All mitochondrial genomes in a cell are identical (homoplasmic) after fertilization but if mutations occur in the life cycle a mixture of different genomes exist in a cell (heteroplasmic). This means that the mitochondrial genome type can differ between organs and change over time. mtDNA sequence variants arise rapidly between generations, while an oogonia contains only a small number of mitochondria, which acts as a developmental bottleneck for the transmission of mtDNA (Shoubridge and Wai, 2007).

Non-functional mitochondrial genes are also common in eukaryotic nuclear DNA (nDNA). These pseudogenes are called NUMTs or *numt*DNAs and exhibit different degrees of similarity to their mitochondrial counterparts. The fragment length and amount differ from roughly 200 short sequences until 500 bp up to a few sequences over 15 kb (Bensasson *et al.*, 2003). Furthermore NUMTs are evenly distributed within and among chromosomes and can be rearranged and fragmented (Ricchetti *et al.*, 1999). Analyses of 13 eukaryotic species show a high variation of inserted NUMTs ranging from none or few copies in *Anopheles* to more than 500 in *Homo sapiens*, *Oryza*, and *Arabidopsis* (Richly and Leister, 2004). It had been suggested that most NUMTs have been inserted in a primate ancestor but still colonized the nuclear genome (Tourmen *et al.* (2002); Bensasson *et al.* (2003)). Hence, the analysis of NUMTs provides a new insight into population history (Thalmann *et al.*, 2005) as well it provoked a discussion about the reliability of mtDNA data (Thalmann *et al.*, 2004).

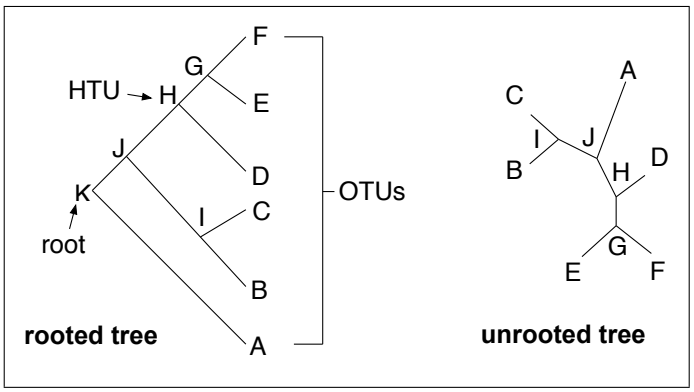


Figure 2.2: Structure of phylogenetic trees. The same topology for a rooted (left) and an unrooted tree (right) are shown. The external nodes A, B, C, D, E, F represent the *OTUs* and the internal nodes G, H, I, J, K are the *HTUs*. The lines between nodes are branches.

2.3 Molecular phylogeny

Phylogeny is the representation of branching history of the routes of inheritance of organisms. To classify species by their evolutionary relationships characters are utilized to distinguish between species. Traditionally morphological characteristics such as brain size or upright walking are examples for these characters. To differentiate between vertebrates and invertebrates presence/absence of a backbone is used. In molecular phylogeny genetic information constitute these characters, such as DNA sequences or restriction fragment length polymorphisms (RFLP). Which characters are deployed depend on the specific problem and the technical opportunities.

Evolutionary relationships are mostly illustrated by phylogenetic trees, which are cycle-free connected graphs. A *binary* tree consists of nodes connected with three nodes (internal node) or one node (external node). The root of a tree is the only node with exactly two connections. In phylogenetics internal and external nodes are also called '*hypothetical taxonomic units*' (*HTUs*) and '*operational taxonomic units*' (*OTUs*), respectively. Trees with a root are called *rooted* trees whereas trees without a root are called *unrooted* trees (Fig. 2.2). The root represents the common ancestor of all *OTUs* and indicates the direction of evolution. A tree can be rooted if at least one *OTU*


```
Human-1  ATCAAACCCCCCCCCCCCCATGCTTACAAGCAAGTAC    37
Human-2  ......a.....t..-..-.................    35
Human-3  ......a.....t....--.................    35
Human-4  ..................-.................    36
Chimp    ......a..tt.a.t..--............c.c..    35
```

Figure 2.3: Multiple alignment of one chimp and four human mitochondrial DNA sequences. Homologue positions are arranged into columns. Nucleotides that match the first sequence (Human-1) are marked as dots (.). Gaps caused by insertion or deletion (Indels) are denoted by a dash (-). The sequence logo shows how conserved positions are.

forms an *outgroup* (most distantly related *OTU*). Typically branch lengths represent the distances between nodes.

The interpretation of such relationships requires the integration of knowledge from other scientific fields like ethology, comparative morphology or palaeontology. For example, the mapping of geographic information onto phylogenetic trees can be utilized to understand the human dispersal.

A prerequisite for sequence based tree reconstruction is a multiple sequence alignment (MSA). Therefore, the generation of alignments is one crucial task in sequence analysis. Each position in a nucleotide sequence presents a phylogenetic character. In the alignment process characters or nucleotides are arranged in columns that are most likely homologous, and the similarity between sequences can be determined. The resulting alignment D is an $n \times l$ matrix with n sequences of length l. Gaps caused by insertion or deletion events are represented by dashes (–). D_{ij} denotes a nucleotide at site j of sequence i. Columns are also called alignment sites (Fig. 2.3). Further information about sequence alignments and tree reconstruction can be found in Waterman (2000).

2.4 Human evolution in the light of mitochondrial DNA

Human mtDNA is widely used to study human history, especially focusing the human origin and pioneer settlement patterns. Because of the maternal inheritance and the absence of recombination, which allow the use of simplified models (Handt *et al.*, 1998). In this section first a review of mitochondrial phylogeny is provided, then the nomenclature of mitochondrial haplogroups and the resulting human migration patterns are explained.

Review of mitochondrial analyses

In 1980 RFLP analyses of 21 mitochondrial genomes from humans with diverse ethnic and geographic background showed group specific patterns of cleavage (Brown, 1980). A clear evidence that mtDNA variation correlated with ethnic and geographic origin of humans came from *Hpa*I RFLP studies with African, Asian and European-American mtDNA. The *Hpa*I restriction site at nucleotide position 3,592 is only found in African individuals. By contrast, approximately 13% of Asians lacked a *Hpa*I restriction site at position 12,406, which was present in all other mtDNAs (Denaro *et al.*, 1981). Further RFLP analyses revealed that there was a single mtDNA tree, with the deepest root occurred in Africa, and that Africans harbored the greatest sequence diversity (Cann *et al.* (1987); Stoneking *et al.* (1990); Merriwether *et al.* (1991)).

At the beginning of the 1990s also mtDNA sequences from the control region (CR), mainly HVR-I and HVR-II, were investigated. Vigilant *et al.* (1991) confirmed for 189 CR sequences, where 121 sequences came from native Africans, that the greatest sequence diversity was found in Africans, that the deepest root was between Africans, and that the coalescence time of mtDNA tree was between 166,000-249,000 YBP. The fast-evolving CR sequences were accompanied by high levels of recurrent mutations, blurring the structure of such trees, while there was insufficient variation to distinguish

important ancient branches (Torroni *et al.*, 2006). Hence, Macaulay *et al.* (1999) used a combination of CR sequence and RFLP analyses to describe humans history. A study of complete mtDNA sequences from three humans (African, European, and Japanese), three African apes (common and pygmy chimpanzees, and gorilla), and one orangutan has shown that European and Japanese sequences were most similar, that the African mtDNA was more divergent, and the closest ape relatives, the chimpanzees, were ten times more divergent from humans than Africans from Asians and Europeans. This study suggested an age for the human mtDNA radiation of 143,000 ± 18,000 YBP and the time for European and Japanese radiation of 70,000 ± 13,000 YBP (Horai *et al.*, 1995). In 2000 a set of 53 human mitochondrial genomes, with excluded control regions, showed a complete separation of Africans and non-Africans. The expansion time for non-African lineages was estimated at 1,925 generations roughly 38.500 years, assuming a generation time of 20 years (Ingman *et al.*, 2000). In 2006 more than 2,000 complete mtDNA sequences were published. Today the basal branching structure of mtDNA variation in many parts of the world is well understood (Torroni *et al.*, 2006).

Cladistic nomenclature for human mtDNA haplogroups

Mitochondrial genomes are characterized by haplogroups or clusters, which are collections of related haplotypes[2]. Haplotypes and mutations are determined according to the 'Cambridge Reference Sequence' which allows the reconstruction of sequences back from haplotypes (Fig. 2.1). The haplogroups represent named clades[3] of the tree which correspond to early human migration patterns or distinct geographical regions and should be monophyletic (Fig. 2.4). The nomenclature of haplogroups is based on a cladistic appreciation of the underlying phylogeny relating the ancestral relationships of

[2]A haplotype refers to the combination of allelic states of polymorphic markers along the same DNA molecule (e.g. mtDNA). Because mtDNA is nonrecombining, haplotype diversity is due only to mutations (Jobling *et al.*, 2004).

[3]All descendants of a single node form a clade.

	2	**3**	**8.1**	**10**	**coded sequence**	**alignment**
reference	T	G	-	C	GTCAAGTTGC	GTGAAGTT–GC
sequence 1	A	*	T	*	GAGAAGTTTGC	GAGAAGTTTGC
sequence 2	*	C	-	*	GTCAAGTTGC	GTCAAGTT–GC
sequence 3	*	C	-	G	GTCAAGTTGG	GTCAAGTT–GG

Table 2.1: Reconstruction of sequences from haplotype tables according to a reference sequence. The table codes three different sequences based on the reference '*GTCAAGTTGC*'. The numbers in the table head define the nucleotide positions according to the reference position. A number before a dot indicates an existing nucleotide position and a number after a dot presents inserted nucleotide after the current nucleotide position. For example, 8.1 marks the first inserted nucleotide between nucleotide 8 and 9 of the reference sequence. A star (*) represents an unchanged nucleotide. Where substituted nucleotides represented by A, T , C or G and a deletion is represented by a dash (-). The sequence alignment is shown in the last column. This kind of table [column 1-5] is called 'Anderson matrix' in this thesis.

haplogroups. Major haplogroups are named by single capital letters. Nested subhaplogroups of major haplogroups are denoted by alternating positive integers and lowercase roman letters. For example: J1b1 $\subset$ J1b $\subset$ J1 $\subset$ J, where '$\subset$' means "*is a subhaplogroup of*". A haplogroup that is composed of a set of named subgroups is referred to by concatenating those names (e.g. HV). To designate a set of mtDNAs that coalesce in an unresolved multifunction but that are not member of any haplogroups branching from that node, an asterisk is appended to the list of those haplogroups. Unnamed clades that enclose haplogroups are indicated by the prefix 'pre-' and the list of enclosed haplogroups (Richards *et al.* (1998); Macaulay *et al.* (1999)).

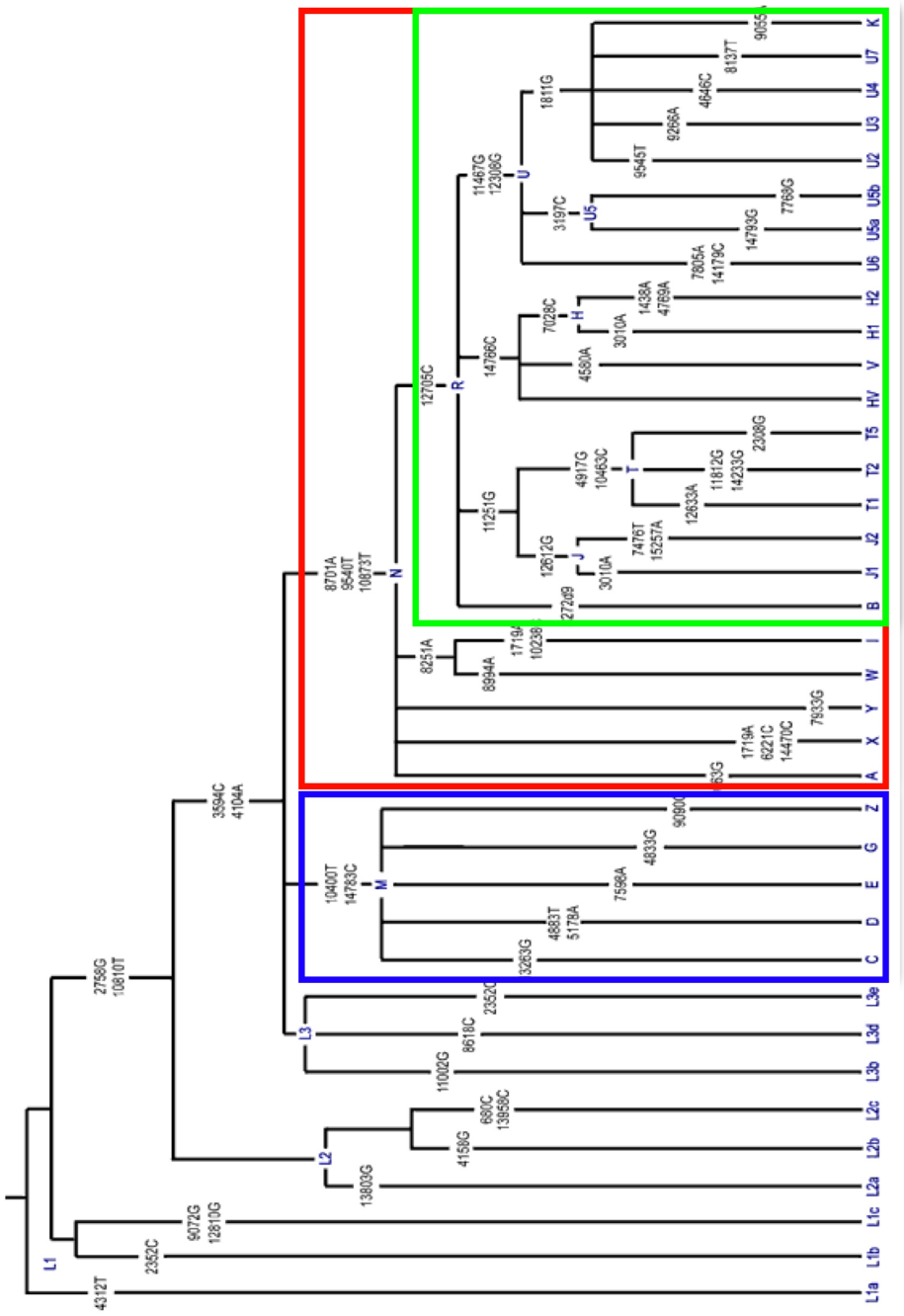


Figure 2.4: A phylogeny of common mtDNA haplogroups. Rectangles indicate three major haplotypes: Blue ≙ M, Red ≙ N, Green ≙ R. Haplogroup L is only found in Africans, where all other haplogroups diverged from the major haplotypes N and M, which diverged from L. The tree was taken from '*www.mitomap.org*' and is extended by the rectangles.

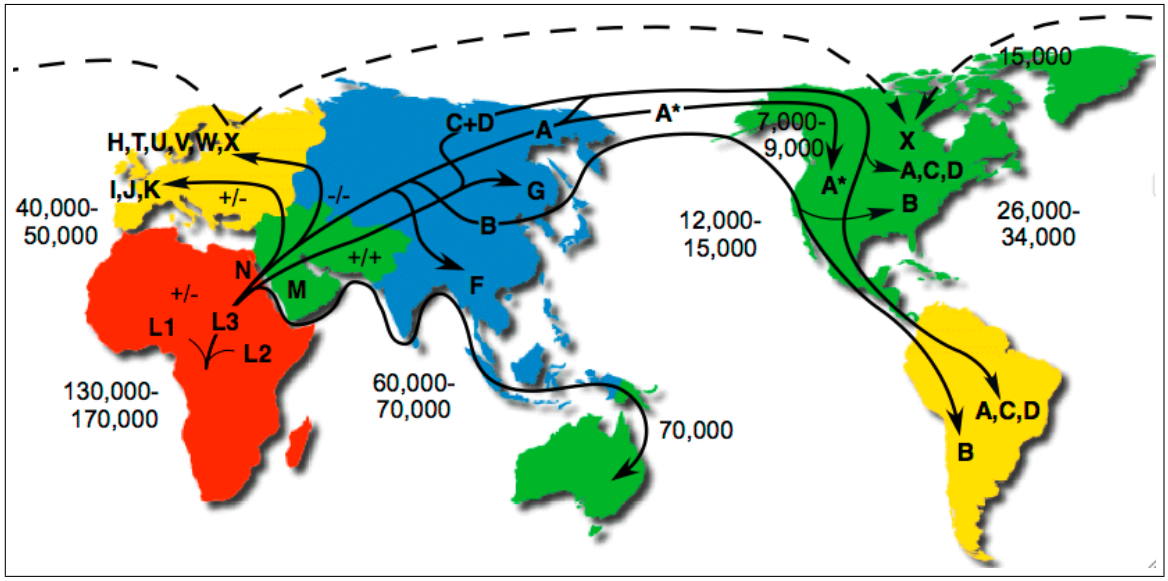


Figure 2.5: Human mtDNA migration by haplogroups. Mutation rate = 2.2-2.9%/MYR. Time estimates are YBP. +/+,+/-,-/- indicate the presence/absence of the restriction sites *Dde*I (10,394)/*Alu*I (10397). * stands for *Rsa*I (16,329). Picture modified from MitoMap (http://www.mitomap.org/mitomap/WorldMigrations.pdf).

Human migration pattern based on mtDNA variation

The results of mtDNA analyses support the '*Out of Africa*' model (Cann *et al.*, 1987) and depict the human dispersal that can be summarized as follows (Fig 2.5): The major haplogroup L consists of the three subhaplogroups L1, L2 and L3, only found in African mtDNA. mtDNA outside Africa fall into the two clades M and N. M diverged into haplogroups C, D, E and G, which are primarily found in South and East Asians and Americans. Haplogroup N is subdivided into A, B, F, H, I, J, K, R, T, U and V and is primarily found in West Asian and European populations (Richards and Macaulay, 2001). Mostly all Sub-Saharan African mtDNA belongs to haplogroup L, whereas in Ethiopian mtDNA up to $\approx 20\%$ of the M variant is found, which identifies Eastern Africans as the source of migration out of Africa (Quintana-Murci *et al.* (1999); Kivisild *et al.* (2004)). It is proposed that the two haplogroups M and N belong to two migration events out of Africa, where M migrated earlier (Jobling *et al.*, 2004).

2.5 General and Mitochondrial Databases

Sequence data is publicly available from general sequence databases or from specialized databases. The differences between both database types are explained and databases similar to HvrBase are presented.

GenBank[4] is one of the major general sequence databases. GenBank contains more than 61 million sequences for more than 240,000 named organisms submitted from laboratories and projects around the world (Benson *et al.*, 2007). A daily synchronization between two other general databases, *EMBL Data Library* in Europe and the *DNA Data Bank of Japan*, ensures equal content in these databases. Specialized sequence databases are smaller, and the sequences they provide are often also available in general databases. But the data is preselected and categorized to match the scope of the collection.

MitoMap[5] is a compendium of polymorphisms and mutations of human mitochondrial DNA. It provides a general and several adaptive search interfaces to explore the database. But only leads to publication references not to DNA sequences directly. MitoMap collects sequences related to human history, forensics and clinical research (Brandon *et al.*, 2005).

mtDB[6] is a collection of 1,865 complete human mitochondrial genomes and 839 human coding regions (*in April 2007*) for population genetics and medical sciences (Ingman and Gyllensten, 2006). mtDB provides two ways of accessing sequence data. Either by choosing sequences directly from geographic or population ordered tables. The latter allows searching for polymorphic sites. As a special feature the user can search for predefined haplogroups by choosing a haplogroup from a haplogroup tree.

[4] http://www.ncbi.nlm.nih.gov
[5] http://www.mitomap.org/
[6] http://www.genpat.uu.se/mtDB/

MamMiBase[7] is a mammalian mitochondrial genome database for phylogenetic analysis. It allows retrieval of nucleotide and protein alignments of the 13 protein coding mitochondrial genes (Vasconcelos *et al.*, 2005). Sequences are selected by a taxonomic tree and then an alignment and a tree is generated on the fly.

The **mtDNA Population Database** from the Federal Bureau of Investigation focused on forensic analyses links sequences to population information and represents statistical information like genetic diversity from Tajima 1989 (Monson *et al.*, 2002).

MitoRes[8], is a resource for mitochondrial genes, transcripts and proteins with 3,180 records for different species (Version 1.4) and **GoBase**[9], an organelle genome database with 350,000 mitochondrial sequences and 118,000 chloroplast sequences, integrates sources from GeneOntology[10] and GenBank to provide own search methods. (Catalano *et al.* (2006); O'Brien *et al.* (2006)).

2.6 Relational database and relational schema design

Biological data management has to cope with the same elementary management challenges known from other scientific fields, such as redundant data storage, inconsistent data, multi-user access and an independence from physically bound data storage is desired. Database management systems (DBMS), like MySQL or Oracle, are commonly employed to address these considerations. A DBMS is a piece of software that manages the database access

[7] http://www.mammibase.lncc.br/
[8] http://www2.ba.itb.cnr.it/MitoRes/
[9] http://gobase.bcm.umontreal.ca/
[10] http://www.geneontology.org

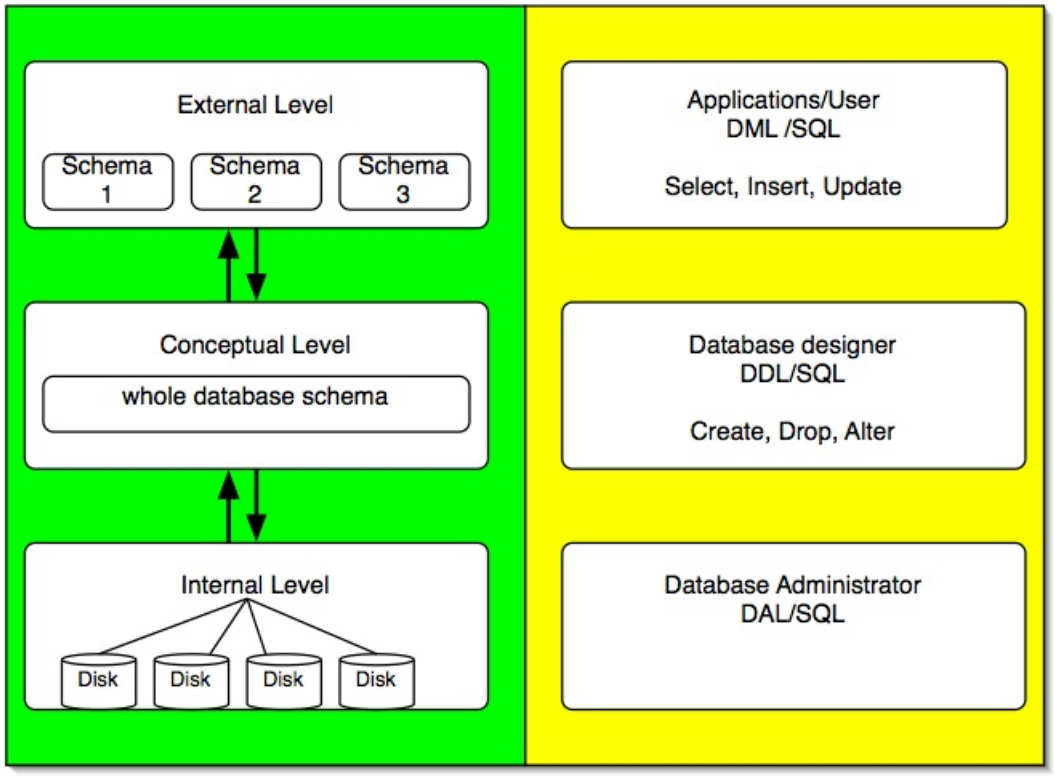


Figure 2.6: ANSI/SPARC DBMS FRAMEWORK: The left side shows the three levels of the DBMS framework. The corresponding box on the right side describes the user and the language mainly used on each level. The *internal* level manages the physical data storage, including indices, storage representation, field orders etc. Controlled by the database administrator who uses the data administration language (DAL). On the *conceptual* level a designer creates the full logical view of the represented data, independent of the physical storage, using the data definition language (DDL). Hence, the designer create tables, views and indices on that external users/applications work on. On the *external* level the data manipulation language (DML) is used to query and modify data.

in a consistent way and forms a layer between the database and programs that work on top of it. It is important to note that the term database only means the collection of data, which can be managed by a DBMS. The DBMS itself is composed of an external, conceptual and internal level (Tsichritzis and Klug, 1978). Applications are only linked to the external level using a standardized language like the structural query language (SQL). Whereas the physical storage is managed by the DBMS (Fig. 2.6). The benefit is that physical data storage is decoupled from data usage. Therefore, data can be accessed by different applications for different tasks. How the data is modeled depends on the DBMS. Today the de-facto-standard for database models is the 'relational model' based on the work of Edgar F. Codd (Codd 1970, 1972, 1979). That variant of DBMS is termed 'relation database management systems' (RDBMS). There are other ways of organizing data, among them the hierarchical or the network model where data is adapted to a tree-like or graph structure (Taylor and Frank (1976); Blackman (1998); Bachman (1969)).

2.6.1 Relational model

The fundamental idea of the relational model is that data are represented by mathematical n-array relations r (Vossen, 2000). A relation is a subset of the cartesian product of n domains (dom), a domain representing all possible or valid values of the attribute A_i, and the relation consisting of a set of n attributes A:

$$r \subseteq \mathrm{dom}(A_1) \times \ldots \times \mathrm{dom}(A_n)$$

A relation with n attributes has the following properties:

- Each row is a distinct n-triple element of r.
- Each value in a tuple is atomic.
- The order of attributes and rows is irrelevant.
- Table and columns are labeled by names.

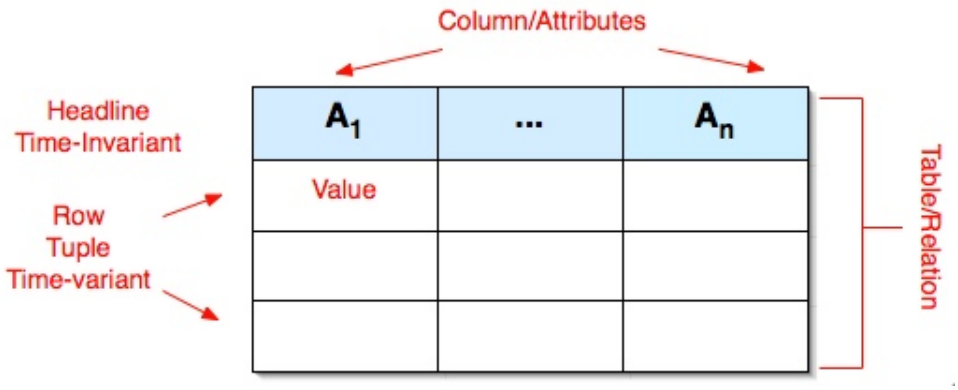


Figure 2.7: Terms of relational databases

A relational database table represents a relation that can be addressed by its table name. The table definition or the headline specify the attributes of a relation. Columns represent values of domain dom(A_i). A row is a single combination of attribute values and is called a tuple (Fig. 2.7).

Attributes or sets of attributes, which unique specify a tuple, are called 'keys'. Every tuple has at least one key consisting of all attributes of the relation. In order to define keys it is important to understand the concept of functional dependency. A functional dependency ($X \rightarrow Y$) between two sets of attributes X and Y which are subsets of the relation r, uniquely defines tuples of Y by attribute values of X. The functional dependency is determined by the semantics of the relation. For example the attribute 'FirstName' of a typical person relation does not functionally determine tuples, as it is possible that two persons share the same forename. Notice that this need not be for the current relation. A key can be defined by one or more attributes for a relation r if those attributes functionally determine all other attributes of the relation and no proper subset of those attributes functionally determines all other attributes of the relation. A set of attributes that contains a key is called a superkey. Thus every key is a superkey, but not every superkey is minimal. Minimal sets of attributes that determine tuples are called candidate keys. If a relation has more than one candidate key, one of this keys is designated as the primary key, which will typically be used as 'foreign key' in other relations to determine tuples of this relation.

The way of data access and manipulation is formally described by 'relational algebra' (Codd 1970, 1972, 1979). Relational algebra consists of a collection of operators (like Select, Project and Join) that operate on a relation or relations and result in a new relation. This means that relational algebra is a closed language and that the result from one operation can used as an input for further operations.

2.6.2 Structured Query Language (SQL)

Nearly all current RDBMSs implement the 'Structured Query Language' (SQL) as their database query language. SQL was originally developed at IBM in the early 1970s and first called 'Structured English Query Language' (SEQUEL) (Chamberlin, 1974). 1987 the first SQL ISO standard is introduced as ISO-8601 (1988), which was subsequently extended. The latest version dates from 2003 (ISO-9075, 2003). Vendors of RDBMS implement different subsets of the SQL standard and provide product specific variations and extensions. Thus SQL code must sometimes be adapted to run an another RDBMSs. Like other database query languages SQL commands can be categorized into three subsets: Data Manipulation Language (DML), Data Definition Language (DDL) and Data Control Language (DCL). The next paragraph shows an overview of the subsets. The examples presented are based on the relational database schema shown in figure 2.8.

DML

DML is mainly used by applications to query or modify data using the following four statements: SELECT, INSERT, DELETE, UPDATE. For example, to get a relation consisting of attributes from the relations *people* and *lessons*, a SELECT-statement is formed that joins both tables and presents only the attributes of interest (*lesson*, *location*, *name*):

```
SELECT lesson, location, name
  FROM   people, lessons
 WHERE person_id = teacher_id;
```

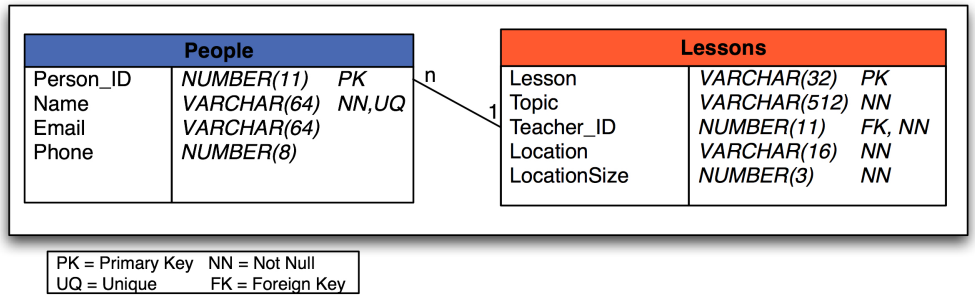


Figure 2.8: Database schema for the relations '*People*' and '*Lessons*'. The headline defines the relation name and each row defines an attribute, where the first column specifies attribute names and the second properties of the attribute.

In the joining process, the Cartesian product of all tuples from both tables is built and only these tuples are chosen for the new relation, for which the join-condition holds ($person\_id = teacher\_id$).

DDL

The DDL SQL statements CREATE, DROP, ALTER define the structure of a database, including tables, columns, rows, views and database specifics. The next example shows how to create the table *people*:

```
CREATE TABLE people (
 person_id   NUMBER(11)   PRIMARY KEY,
 name        VARCHAR(64)  NOT NULL UNIQUE,
 email       VARCHAR(64),
 phone       NUMBER(8)
);
```

In contrast to physical stored data of tables, it is possible to create logic or 'virtual' tables called 'views', which represent relations composed of predefined select statements. Every time a view is queried, the predefined statement is executed to create a temporary relation. With views it is possible to restrict data access or present a convenient way of accessing data. In Oracle a second type of 'view' can be created called materialized view (MV). An

MV keeps preselected data time-persistent in a database management system. MVs help to reduce joining processes for recurrent queries in slowly changing databases. It is also possible to present fixed versions of data based on changing data.

DCL

DCL is a subset of SQL and is used to control database access with the statements GRANT and REVOKE. The right to connect to the database, to select, update, delete, insert data can be granted or revoked to users or roles. In a realistic scenario one would grant students only the right to connect to the database and to select a '*PublicPeopleLessons*' view.

2.6.3 Procedural Language/Structured Query Language (PL/SQL)

PL/SQL is a procedural extension to SQL from the Oracle Corporation. It is not a standalone language and depends on proprietary Oracle products like the DBMS. PL/SQL supports variables, conditions, arrays and exceptions. Since Oracle RDBMS 8 features associated with object-orientation are included. It can be used:

- via databank frontends, where the code is directly executed on the server.
- as predefined code which extends the DBMS functionality. The code can be stored as a subroutine ('stored procedure') or as a collection of subroutines ('stored packages'). In principle every user can execute such code depending on his accession rights.
- to program database triggers.
- as a programming language in the tools Oracle-Forms and Oracle-Reports.

The main benefit of PL/SQL is that the functionality provided by a PL/SQL program is available for different users or applications. Additionally a processing speedup due to reduced network traffic between client and DBMS is gained. Further information found in Owens (1999), Feuerstein and Pribyl (2005).

2.6.4 Aspects of relational schema design

In the last paragraph the programming language SQL was presented. Now problems of designing relational database schemas are discussed. We start with a discussion about relational database *anomalies*, which are essentially erroneous changes to data, more specifically to single records. Common problems, like updating, deletion and insertion anomalies are due to redundancy and can be reduced by splitting a relation into smaller sub-relations (decomposition). This is one way of designing a relational schema starting from a huge relation consisting of all attributes and then decomposing it into smaller relations. This process of minimizing redundancy and anomalies by decomposition is called 'normalization'. The states of normalization can be described by the normal forms (NF) 1-5 and the Boyce-Codd Normal Form (BNF). Normalization is an incremental process, each normal form layer adds to what-ever normal forms have already been applied. For example, 2nd NF can only be applied to tables in the 1st NF, and 3rd NF only applied to tables in the 2nd NF, and so on. Each NF is a reference of the previous one. In the following the first three normal forms are shown:

First normal form: No non-atomic attributes or nested relations are allowed, which means each column must contain only single values. For instance if the '*location*' attribute of the '*lessons*' relation (Fig. 2.8) consists of the attributes '*building*' and '*room number*' it must be split into these two atomic attributes.

Second normal form: Non-prime attributes must fully functionally depend on the key attributes, or in other words no subset of primary key attributes determine non-prime attributes.

Third normal form: There are no non-trivial functional dependencies between non-prime attributes. A violation of 3NF would mean that at least one non-prime attribute is only indirectly dependent (transitively dependent) on a candidate key.

One remaining challenge in relational database design resides: How to avoid anomalies while still building a useable schema that matches the real word characteristic. Ongoing with normalization the complexity of schemas rises and with it the number of joins needed to query data. This can reduce efficiency, especially when the 'query/changes' ratio is high. Extreme cases can be found under data-warehouse conditions, where a huge amount of integrated data is analyzed. One of the simplest data-warehouse schemas is the 'star schema', which consists of a fact table and accompanying dimension tables. The schema resembles a star with the fact table as center, which is directly connected to the dimension tables. A fact table mainly consists of *foreign keys* and is coded predominately in third normal form, where denormalized dimensional tables hold the data. Normalization of the dimension tables leads to the 'snowflake schema'. Further surveys can be found in Inmon (1994) and Lehner (2003).

2.7 Software Design

This section briefly presents the phases of software development and introduces the notions of client-server architectures in respect of the implemented Web application.

The process of software development from an initial idea through maintenance of the completed application can be split into smaller more easily

manageable parts, which aid to increase the quality and the usability of an application. These processes are known as 'systems development life cycle' (SDLC or SLC) techniques (Vossen, 2000). SDLC techniques typically include the phases: initiation, planning, design, development, testing, implementation, and maintenance. The first model conceived was the 'waterfall model', in which each phase is entered exactly once (Winston, 1999). A modification of the waterfall model led to the 'spiral model', where phases can be entered many times (Barry, 1988).

In client-server architectures tasks are separated into client and server processes (nodes) and are almost always connected by a network. The most basic type employs only two types of nodes: A client, which represents the user interface and a server carrying out the process logic, calculations and storing data. An extension is the 'three-tier client-server architecture', which consists of a client, a web server and a database server. The separated database server increases performance and raises the security level of the application.

A web browser requests a web server via the 'unified resource location' (URL), the web server responses to the request and by typically sending back Hyper text markup language (HTML) code. The HTML code get rendered by the web browser. HTML site is static and every update requires a new server request. One can overcome that limitation by means of the scripting language JavaScript, which runs on the client side. It can react on user interaction and can alter HTML sites directly in the web browser. A technique called XMLHttpRequest allows for requesting the server in the background. The received content can be used to modify the web page without reloading the whole page. For that mechanism the term Asynchronous JavaScript and XML (AJAX) was coined, which is widely used to extend the functionality of web applications (Crane *et al.*, 2006).

Chapter 3

Extending HvrBase

This chapter describes the extension of HvrBase from a mitochondrial control region sequences database to a more general database, called HvrBase++, for DNA sequences used for primates phylogenetic studies. First a historical review on HvrBase is given to get familiar with it and determine the state before re-creation. Then the requirements for the new version are specified (3.2). Before HvrBase was extended the quality of the sequence data are checked against corresponding sequences from GenBank, which is described in section 3.3. The following development process is split into three parts: transformation of the database, enhancing the collecting process and the creation of the web interface (3.4-3.6). The last sections presents and discuss the outcomes of HvrBase++ development.

3.1 Historical view on HvrBase

HvrBase has started 1997 as a compilation of 4,079 HVR-I and 969 HVR-II sequences with the purpose to support human mitochondrial phylogenetics (Handt *et al.*, 1998). For that reason not only sequences were collected but also seven properties describing the donor individuals (Tab. 3.1).

All sequences of both HVR regions are manually aligned. For HVR-I region the positions 16,001–16,408 and for the HVR-II region positions 1–408

Properties	**Description**	**Example**
Nr_Seq1	ID of the HVR-I sequence of individuals	no example
Nr_Seq2	ID of the HVR-II sequence of individuals	no example
Name	Original name used in publication or GenBank ID	1.2; Et154; CP-L16; CAMB
Reference	Publication reference	Anderson et al. Nature 290:457-465, 1981
Continent	Continent the individual stems from	Europe; Africa
Country/ Region	Country the individual stems from	German, Rhine area; Africa, N.W., Senegal
Population	Population the individual belongs to	Arabic; Amerind
Language	Language and language phylum of the individual	Bantu; Mbenzele
+/– bpdel	Indicates presence or absence of 9 bp deletion	+9bpdel
Species	Species the individual belongs to	*Pan troglodytes*; *Homo sapiens*

Table 3.1: This table presents the individual properties describing the donor individual in the lastest version of HvrBase. Property '*species*' was added to the original property set 1998. Further the property '*language*' is only defined for humans.

are taken into account. Longer sequences are truncated and shorter sequences are extended by question marks to achieve the length of 408 bp required by the alignment. Also all non-determined nucleotides within a sequence are represented by question marks. A dash indicates an insertion or deletion of a nucleotide. Hence a sequence update or incorporation required a realignment of those two hand made alignments.

Interesting new data sets were found by reading related publications or were offered by scientists. At the beginning of HvrBase most sequences were not accessible in public databases. Instead control region sequences must be extracted from publications. Sometimes tables illustrate only the substitutions found in these sequences with respect to the 'Cambridge Reference Sequence' by which the sequences must be reconstructed. The individual properties were mostly extracted from the text or tables of publications. But sometimes it was also necessary to request for lacking information from the authors or not all properties were available for those sequences. Property entries were taken as they extracted from publication and not standardized.

The whole collection was managed in three ASCII files. One file holds the individual information and the other two files store aligned sequences for HVR-I and HVR-II regions. Identical sequences are stored once and linked to multiple individual information. This version of HvrBase uses a program written in C to retrieve all individual sequences that match a user-defined keyword in the information file. The search results are made available in four files:

- *kw-info*: contains the information about the individuals.
- *kw-I*: contains aligned HVR-I sequences.
- *kw-II*: contains aligned HVR-II sequences.
- *kw-I-II*: contains sequences of individuals from both regions.

In 1998 HvrBase was extended by 295 HVR-I and 13 HVR-II sequences from great apes (*Pan troglodytes*, *Pan paniscus*, *Gorilla gorilla*, *Pongo pygmaeus*)

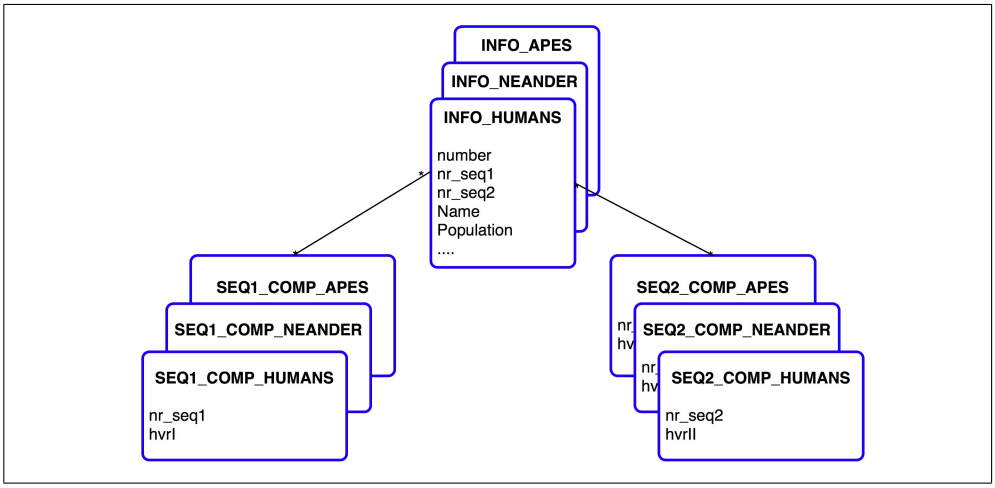


Figure 3.1: Database schema of HvrBase in 2000. For each part of the collection '*Humans*', '*Apes*' and '*Neander*' three tables are defined. The *INFO* tables hold information about the donor individual (Tab. 3.1). Alignments of HVR-I and HVR-II sequences are stored in *SEQ1_COMP* and *SEQ2_COMP* tables.

(Burckhardt *et al.*, 1999). HVR-I and HVR-II alignments were created and handled in two additional ASCII-files and the new property '*species*' was added to the apes information file. Furthermore a self-running FilemakerPro database application with a graphical user interface was developed to retrieve sequences.

In 2000 sequences from neandertaler (Krings *et al.*, 2000) were added to HvrBase and a web application was written (Fig. 3.2). For that reason the collection was migrated from ASCII files to MySQL. The used database schema is analogue to the ASCII files. For each of the categories *Humans*, *Apes* and *Neandertaler* three tables are used to manage the data (Fig. 3.1). One table is used to hold donor individuals information and the two other tables store the HVR-I and HVR-II alignments. The latest version of HvrBase from 2002 contains data about 10,240 individual (9,768 humans, 469 great apes and 3 neandertaler) and 9,860 HVR-I sequence (9,388 humans, 469 apes, 3 neandertaler) and 3,317 HVR-II sequence (3,302 humans, 13 apes, 2 neandertaler).

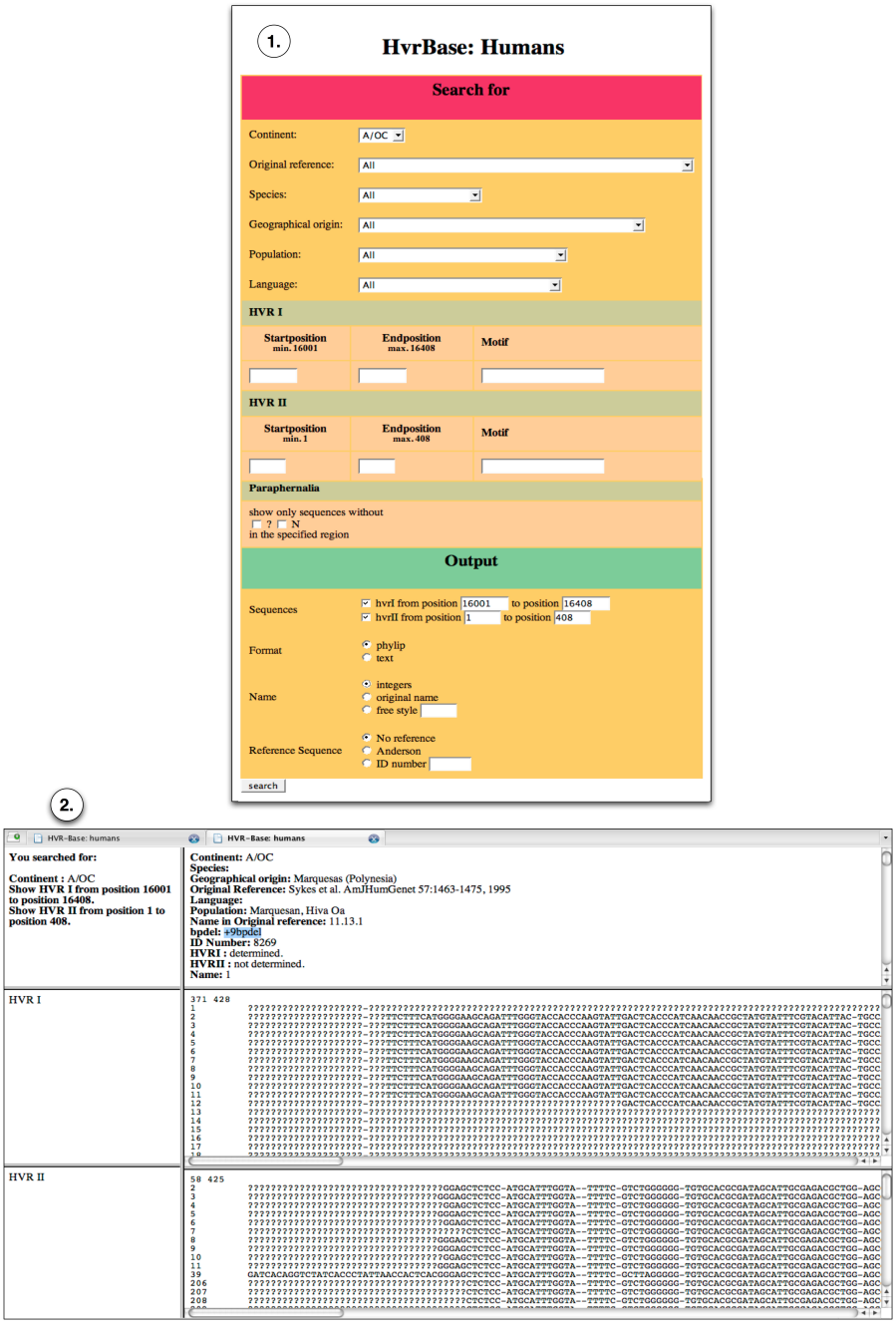


Figure 3.2: This figure shows the *humans* search form (1) and the result window (2). HVR-I and HVR-II sequences can be retrieved by six individual properties and by a motif search. The motif search allows the determination of the motif location. The bottom part of the search form defines the output format. The result window consists of six frames. The top left frame shows the query conditions and the top right frame presents the properties of all retrieved individuals. The middle and buttom right frames show the corresponding sequences. Results can only be downloaded by *copy and paste*. The specific *Apes* and *Neandertaler* windows look similar to these two human windows.

3.2 Requirement analysis for HvrBase++

Before an application is enhanced or upgraded it is necessary to analyze the existing application and the user profiles to find out necessary improvements. Users of HvrBase are mainly biologists interested in population history. The main reason why HvrBase is used, is the benefit that those sequences can simply selected by properties important for this kind of analysis. In other words the specialization of HvrBase in contrast to general public databases, like GenBank or EMBL, is the main cause to use HvrBase. Hence to improve the usability of HvrBase, the search possibilities must be extended and adapted to user workflows.

Geographic information about the individuals are commonly used in this field, therefore geographic maps can be used to make the search more intuitive. These maps can project search results and answer the question, where is an attribute or sequence pattern distributed over the world, at a glance. In a similar way, sequences can be visualized by phylogenetic trees to verify the collection depending on sequence relations. Further a bibliographic search tool which links offered sequences to publications enhance the usability.

HvrBase provides sequence alignments, which give the opportunity to immediately analyze these sequences after downloading. To strengthen the idea of simply using sequence sets, HvrBase++ should provide different sequence formats. Nevertheless a single search rarely ends up in the final collection. Hence, the workflow of HvrBase++ must support the assembly of a user collection consisting of multiple search events. Another benefit, that HvrBase++ must support, is the opportunity to share such an user collection with other co-workers.

A second important point for the acceptance of databases is the integrity and the quality of the data. Following that, the sequences of the existing collection are verified against sequences from GenBank and checked for misleadingly sequenced nuclear mitochondrial pseudogenes (NUMTs). After that new ways of controlling the data must be found to guarantee the quality of the HvrBase++ collection. Moreover, the collection must be ex-

tendable for new available sequences, like complete mitochondrial genomes, to following trends in population history. But not only sequence data must be extended also individual information must be revisited. These information must be standardized by official standards to provide advanced search opportunities. Furthermore, generally used haplogroup information, which clusters sequences to geographic regions or ethnic groups, must be added.

From the technical point of view, HvrBase++ must be completely new designed to fullfil the new requirements. Moreover the handcrafted way of updating the collection must be transfered into a semi-automated process to handle the increasing amount of available sequences.

HvrBase users are scientists, which are not interested in data management and techniques behind. They mainly focus on open questions in population history and need these sequences to find some answers. The idea of an easy to use application remains for HvrBase++ but with new improved search concepts. The following list outlines the new requests:

- More specialized and intuitive search features
- Data visualization (e.g. world maps)
- Low effort to use the application
- Reliable data sets
- Supporting the workflow of scientists
- Successive extensions of important genetic loci

3.3 Controlling sequence data of HvrBase

This section describes the comparison of HVR-I and HVR-II sequences against corresponding sequences from GenBank and check HvrBase sequences for misleadingly sequenced NUMTS.

Comparison between GenBank and HvrBase

Before sequences from both databases can be compared, they must be mapped to each other. This must be guaranteed that both sequences really belongs to the same individual, described in the same publication. But no unique identifier is available for GenBank sequences in HvrBase. Hence, the 13,177 HVR-I and HVR-II sequences can only be mapped by the sequence names and the bibliographic references. The sequence pattern itself can be used to detected possible candidates but it is alone an insufficient criterion. Hence two attempts are applied to find possible sequence in GenBank, which then are manually mapped to HvrBase sequences:

The first attempt fetches for each publication in HvrBase the corresponding sequences set from GenBank via publication ID (PubMed ID). For that reason the PubMed ID for all 91 publications from HvrBase were manually determined with PubMed. After that the NCBI entrez programming utilities (E-Utilities) used in a Perl script to fetch sequences related to the publication. For 34 of the 91 publications 3,108 sequences are available. The second approach tries to identify sequences by BLAST searches for HVR-I and HVR-II sequences from HvrBase against nucleotide sequences from GenBank. Restricting Blast searches with an expect value of 1×10^{-50} we found 5,681 HVR-I and 2,303 HVR-II sequences in GenBank.

Comparing HvrBase properties and GenBank records from both approaches lead to 2,061 mapped sequences. 157 of these 2,061 sequences show discrepancies. The discrepancies are analyzed in more detail and errors caused by HvrBase or variations caused by new sequence versions in GenBank are corrected in HvrBase++. If sequences differ between GenBank and publications and no sequence update occurs, sequences remain unchanged in HvrBase++. In case of doubt those sequences are deleted. See Table 3.2 for detail.

Reference	No. of sequences in HvrBase	No. of sequences in GenBank	No. of deviation	Explanation for the discrepancies
Batista *et al.* 1995	63	44	5	HVR-I sequences for haplotypes K21, K28 had a deletion at position 16,182, which is not reported in publications.
Di-Rienzo *et al.* 1991	69	54	4	Typos in HvrBase
Helgason *et al.* 2000	804	740	0	Not all sequences found
Kolman *et al.* 1995	92	92	46	HVR-II sequences update one year later
Lum *et al.* 1994	75	73	2	Different sequences in paper and GenBank and typo HvrBase
Pult *et al.* 1994	99	74	-	Different sequences names used, not comparable
Sajantila *et al.* 1995	349	336	10	Sequences updated in GenBank
Shields *et al.* 1993	108	56	3	Assignment of sequences names to sequences was wrong in HvrBase
Vigilant *et al.* 1991	374	263	69	Not clear
Ward *et al.* 1991	63	61	2	Different Sequence in paper and GenBank
Watson *et al.* 1996	242	242	4	Typos HvrBase and Sequences names switched
Redd *et al.* 1995	60	60	-	Different sequences names used, not comparable
Morin *et al.* 1994	60	48	1	Typo in HvrBase
Wise *et al.* 1997	49	49	10	HvrBase uses only 'N' for missing bases.
Xu *et al.* 1996	3	3	1	Not clear

Table 3.2: Mitochondrial control region sequence deviation between HvrBase and GenBank. The numbers of sequences found in HvrBase and GenBank and the discrepancies for a publication are shown. The number of found sequences in GenBank is smaller or equal than in HvrBase, because sometimes sequences in GenBank refer to haplotypes and not to individuals. Further sometimes not all sequences of a publication are offered in GenBank.

	Humans		Apes		Neander	
Identity %	**HVR-I**	**HVR-II**	**HVR-I**	**HVR-II**	**HVR-I**	**HVR-II**
< 0.6	2	0	2	0	0	0
0.6 - 0.69	3,009	1	200	0	9	0
0.7 - 0.79	10,035	1,060	45	8	0	3
0.8 - 0.89	3,371	649	5	4	0	0
0.9 - 0.99	1,008	1	0	0	0	0

Table 3.3: Shows the identity of fragments found by BLAST searches of HVR-I and HVR-II sequences from HvrBase against the human genome. The numbers indicate how often this fragment is found over all searches.

Checking for misleadingly sequenced NUMTS

All HVR-I and HVR-II sequences are compared with the human genome (revision 16) by BLAST[1] (Tab. 3.3) and BLAT[2] (Tab. 3.4) searches. 13 chromosomal locations mostly between 30 bp and 150 bp are found for the HVR-I and HVR-II sequences. On chromosome 17 a region of 370 bp was found which covers around 235 bp of HVR-II (Tab. 3.4, 3.3). HVR-I and HVR-II fragments found in the nuclear genome are too short and too diverse to influence the quality of collected HVR sequences.

[1]Basic Local Alignment Search Tool. Find regions of local similarity between sequences
[2]BLAST-like alignment tool

HVR-I

Chromosome	**Position of found region**	**Hits**	**Length min.**	**Length max.**	**Mismatches**
3	43,231,412 - 43,231,535	20	55	56	3-2
4	65,476,525 - 65,476,589	29	44	44	2
5	93,980,571 - 93,980,696	1,492	43	113	2-12
9	34,989,255 - 34,989,507	139	30	79	0-4
11	31,540,968 - 31,541,049	8	59	75	6-7
16	83,486,308 - 83,486,580	1	66	66	6
17	22,165,719 - 22,165,928	1	65	65	2

HVR-II

Chromosome	**Position of found region**	**Hits**	**Length min.**	**Length max.**	**Mismatches**
1	235,151,248 - 235,151,758	29	60	108	1-24
2	43,113,201 - 43,113,370	10	47	47	3
5	80,032,192 - 80,032,260	1,258	31	80	0-4
8	68,542,999 - 68,543,046	1	41	41	4
13	94,046,721 - 94,046,780	525	39	48	2-3
17	22,166,290 - 22,166,525	57	149	239	9-24

Table 3.4: These tables show the results of BLAT searches for 5,873 HVR-I and 1,721 HVR-II sequences against the human genome (built 16). '*Position*' specify the borders reconstructed from all searches. '*Hits*' means how often this region is found over all BLAST searches. '*Length min/max*' defines the minimum and maximum length found for that positions. '*Mismatch*' presents the number of observed mismatches for the position.

3.4 Transforming the database schema

This section describes the database schema and the standardization and extension of HvrBase properties. This is a fundamental step, which affects the later accessibility of data. Further implementation steps depend on that schema and it can restrict the functionality of the application. First the core concept of the new database schema is explained in detail, whereas the transformation process itself is described in figure 3.3. Later on details of the database schema are outlined, while explaining the standardization of HvrBase properties.

3.4.1 Basic database structure

The table *Sample* is the center of the new database schema of HvrBase++, it pools all information available for an individual sequence (Fig. 3.3). Information itself are mostly distributed into additional tables to reduce redundancy to avoid database anomalies (Tab. 3.5). Hence, the only information directly accessible via *Sample* is the sequence name (*name*), the internal sequence id (*sam_id*) and the external GenBank ids (*acc*, *acc_version*). Sequence names are taken from publications or sequence files and are not unique for the HvrBase++ collection. The GenBank id maps sequences to the corresponding source available in GenBank. This allows an automated update mechanisms for sequences available from GenBank. The three foreign keys *seq_id*, *gen_id* and *met_id* integrate all other information from the sequence tables (*Alignment*, *Sequence*, *Manualalignment*), *Gene/Loci* and *Metadata* table, respectively.

The integration of further genetic loci and the rising amount of sequences make it necessary to replace the manual alignments by computer generated alignments. For that reason MAFFT is used (Katoh *et al.*, 2005), because it is fast and can handle huge data sets. The manual alignments are similar to the MAFFT alignments and only kept to offer the original HvrBase alignments for comparability. Further the unaligned sequences are offered, which

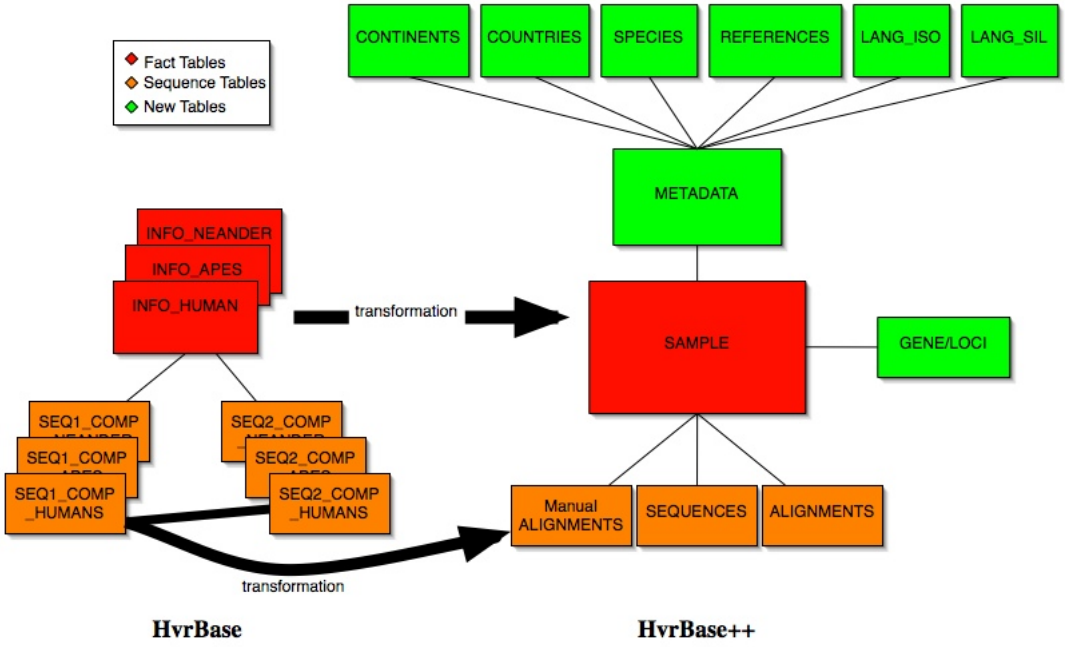


Figure 3.3: Transforming the database schema. Database schema for HvrBase (left side) and after the transformation (right side) are shown. First the three red labeled *Info_x* tables are transformed into table *Sample*, where *x* stands for one of the three sets: Human, Apes, Neander. The attribute '*species*' is now used to distinguish between the three joined sets. In the second step the six orange labeled *Seq1/2_comp_x* tables, which store the aligned haplotype sequences of HVR-I and HVR-II, are integrated into table *Manual Alignments*. After that two additional tables for unaligned haplotype sequence (*Sequences*) and computer generated alignments (*Alignments*) were established. The normalization expands the schema of the green labeled tables. Table *Metadata* keeps properties related to individuals and *Samples* keeps properties devoted to sequences.

make it possible to generate own alignments. The three variations of the sequences are managed in three identical tables (Tab. 3.5). Sequences until 4,000 characters are handled in columns called '*seq_var*' and longer sequences in columns, called '*seq_lob*'. This is done to manage short sequences with the more flexible 'VARCHAR' data type, which is restricted to 4,000 characters and longer sequences with 'CLOB' data type.

Table *Metadata* provides all information about an individual, like geographic origin or spoken language. But nearly all information are defined in additional tables, except *population* and *lang_pap* (language as used in publication) attributes (Tab. 3.5). The attributes of *Metadata* are excluded from table *Sample* because they build an unit and a tuple of *Metadata* can describe a set of tuples in *Samples*. The organization of the missing individual properties are shown in figure 3.4 (green labeled tables) and are explicitly explained in the following section.

The schema described above is mainly used to extent or update the collection. The normalization prevents update anomalies but the sequence retrieval of the web application must join this separated tables for every search. To avoid these *recurrent joins* a 'materialized view' is used to present all individual sequence properties de-normalized in a tuple (A.2). Further this view is used to enhance the security of the application and to distinguish the collection process from the sequence retrieval step offered by the HvrBase++ application (See section 3.6.2).

Moreover, seven administration tables are created (cyan labeled tables Fig. 3.4) to log sequence changes (*History*, *Update_log*, *Alignment_log*), to provide session management (*UserCache*, *Sessions*), information about the mitochondrial genome (*Genome_Map*) and versioning (*Version*).

Sample

Column	Data type	Description
sam_id	number(11)	Primary key
name	varchar2(32)	Sequence name used in publication
acc	varchar2(16)	Accession number used by GenBank
acc_version	number(3)	Version offered by GenBank
seq_id	number(11)	Foreign key to haplotype sequences
gen_id	number(11)	Foreign key to sequence types/genes
met_id	number(11)	Foreign key to individual properties

Metadata

Column	Data type	Description
met_id	number(11)	Primary key
population	varchar2(128)	Population defined by publication
lang_pap	varchar2(64)	Language name used in publication
lang_iso_id	number(11)	Foreign key to ISO language entries
lang_sil_id	number(11)	Foreign key to SIL language entries
ref_id	number(11)	Foreign key to bibliographic entries
cou_id	number(11)	Foreign key to country entries
con_id	number(11)	Foreign key to continents entries
spe_id	number(11)	Foreign key to species entries

Alignment/Seq/Malignment

Column	Data type	Description
seq_id	number(11)	Primary key
seq_var	varchar2(4000)	Sequence until 4,000 characters
seq_lob	clob	Sequence over 4,000 characters

Table 3.5: Column definition of tables *Sample*, *Alignment/Seq/Malignment* and *Metadata*. *Sample* is the fact table, which connects all other tables and represents all information for a sequence. This table holds only the sequence name and external GenBank ids for sequences. All other attributes are foreign keys of other tables. Foreign key '*seq_id*' in *Sample* integrates the haplotype sequences and derive alignment sequences of table *Seq*, *Malignment* and *Alignment*. Foreign key '*met_id*' connects *Sample* with table *Metadata*, which collects individual information.

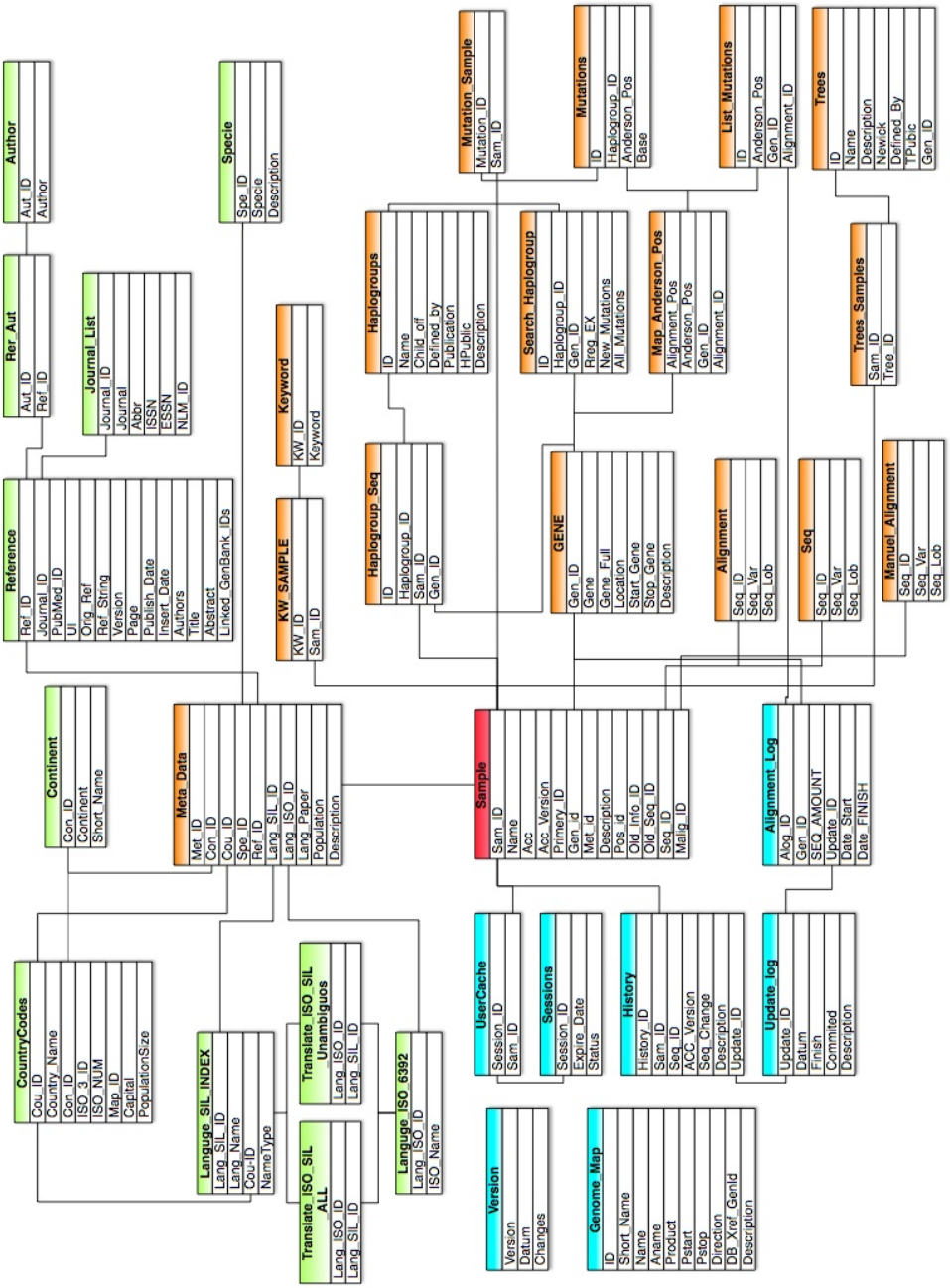


Figure 3.4: HvrBase++ database schema. The red labeled table '*Sample*' connects all facts available about collected sequences. The orange labeled tables present information related to the sequence, whereas the green labeled tables present information about the donor individual. Cyan labeled tables are needed for the management of the data and for the functionality of the web application.

Continent HvrBase	Continent HvrBase++	One letter code
AFRI	Africa	F
ASIA	Asia	L
AMER	North America	N
AMER	South America	S
A/OC	Australia	O
EURO	Europe	E
	Antarctic	A

Table 3.6: Defined continent entries in HvrBase and HvrBase++.

3.4.2 Extending the individual properties

The properties '*continent*', '*country*', '*language*' are standardized by official standards and the bibliographic '*reference string*' is replaced by a set of bibliographic properties. Additionally a new property for mitochondrial haplogroups is integrated into HvrBase++.

Geographic information about a donor consists of the properties '*continent*' and '*country*'. Continent entries were well-defined in HvrBase, but abbreviated. The short form is changed to the full name together with a one-letter code for continent names. America is subdivided into North and South America, and Antarctic was added (Table 3.6).

Entries for '*country*' sometimes contain additional specifications or only an area is described. To standardize these entries country names are mapped to an official list of country names. Table 3.7 shows entries for '*continent*' and '*country*' in HvrBase and HvrBase++. The list also provides general accepted *ISO 3166-1* IDs for countries in two-, three-letter code and a numeric code. These codes are added to the database and facilitate the exchange of country information with external sources.

Furthermore, countries were linked to corresponding continents by political memberships. For example, Canary Island belongs to the sovereign territory of Spain and so it belongs to Europe.

Countries HvrBase	**Countries HvrBase++**	**2 letter code**	**3 letter code**	**Numeric code**
Africa, N.W.	Null	-	-	-
Africa, N.W., Senegal	Senegal	SN	SEN	686
Australes Islands (Polynesia)	French Polynesia	PF	PYF	258
Tahiti (Polynesia)	French Polynesia	PF	PYF	258
German, Rhine area	Germany	DE	DEU	276
Germany, Lower Saxony	Germany	DE	DEU	276
Germany	Germany	DE	DEU	276

Table 3.7: Country entries from HvrBase and HvrBase++ with corresponding extended two-, three-letter and numeric *ISO 3166-1* codes are shown.

These geographic properties are managed in the two tables *Continent* and *CountryCodes*. The primary keys '*con_id*' and '*cou_id*' of *Continent* and *CountryCodes*, respectively, are used as foreign key in table *Metadata* to specify the geographic information (Fig. 3.4).

All **language entries** in HvrBase have been adapted to comply with the SIL (Summer Institute of Linguistics) and ISO/DIS 639-2 language code standards from Ethnologue vol. 14 (Grimes, 2000). In order to avoid information loss and to compensate the incompleteness of any standards, it was necessary to integrate both language codes (Table 3.8). The following example shows the hassle of associating a mother tongue of an individual deduced from a publication with the SIL and/or ISO language codes: It is known that a certain tongue belongs to the Niger-Kordofanian language family. Niger-Kordofanian is a collective language code only used in the ISO standard whose languages can be found throughout Southern and Central Africa as well as in Sub-Saharan Western Africa. Since that language family does not have a SIL code, a more in-depth knowledge about the very tongue (e.g. language name and habitation of a tribe) would be essential to find a suitable SIL code.

Both language standards are managed in the two tables '*Language_SIL_Index*'

SIL	ISO	No. of individuals	Language family or population
Yes	Yes	7,248	English
Yes	No	41	Mandenka (population from Senegal, 'Mandinka' in SIL)
No	Yes	1,951	Bantu (Africa's largest language family)
No	No	454	Mbenzele (population from Central African Republic)
-	-	4,611	Language information missing or not assignable
Total		14,305	

Table 3.8: Assignment of language names to SIL and ISO/DIS 639-2 codes. The first two columns show if an assignment is possible to SIL or ISO. Column three represents the amount of entries match this assignment. The last column shows a typical entry in HvrBase++ and a short explanation.

and '*Language_ISO_6392*' and linked to table '*Metadata*' via primary keys '*lang_sil_id*' and '*lang_iso_id*'. Relations between both standards are provided by two join tables. '*Translate_ISO_SIL_all*' maps all language entries but is ambiguous and '*Translate_ISO_SIL_unambiguous*' maps only unambiguous entries (Fig. 3.4).

Bibliographic information in HvrBase was limited to string referencing a publication. This string consists of the name of the first author, a journal abbreviation, volume and page information and the year of publication. The following list gives an impression how publications are referenced:

- Torroni et al. AmJHum Genet 53:591-608, 1993b
- Torroni et al. AmJHumGenet 53:563-590, 1993a
- Anderson et al. Nature 290:457-465, 1981
- Seo et al. ForSciInt 97:155-164, 1998
- Harris and Hey, Current Biology 2001, 11:774-778
- Sajantila et al, Proc Natl Acad Sci USA

The string format slightly differs between entries, like using a comma instead of a point after author name. Further the way of abbreviating journal names is ambiguous. All these details do not change the information content but make it difficult to search for a single criteria in it. This can be prevented

if data is normalized to the first normal form (1NF) by splitting aggregated information into independent properties. Therefore fifteen properties, including the original reference string are used in HvrBase++ to manage publication data (Tab. 3.9). These properties are based on bibliographic data offered by PubMed[3], a service of the U.S. National Library of Medicine that includes over 17 million citations from life science journals for biomedical articles back to the 1950s. These data are freely accessible and give the opportunity of an automated citation upload into HvrBase++. Moreover a unique author list is provided by HvrBase++ to link authors to publications. As no official or general accepted list is available, the list is generated semi-automatically.

Furthermore a file of about 17,000 biomedical journals also available from PubMed[4] is integrated into HvrBase++ to improve the quality of the database.

51 common **haplogroups** were taken from mtDB and integrated into HvrBase++ (*Detailed list see Fig. A.4*). A tuple of table '*Haplogroups*' defines the haplogroup name, the ancestral haplogroup, a short description and a reference for the haplogroup (Fig. 3.4). The haplogroup substitutions corresponding to the Cambridge reference sequence and a predefined regular expression are managed in table '*Search_Haplogroup*'. The regular expression is used in Oracle SQL-statements to assign sequences to haplogroups. To speed up the haplogroup search the assignments are kept in table '*Haplogroup_Seq*'. Currently only mitochondrial haplogroups are present but the database schema also support further haplogroups, like Y-chromosomal haplogroups.

[3]http://www.ncbi.nlm.nih.gov/sites/entrez?db=PubMed

[4]ftp://ftp.ncbi.nih.gov/pubmed/J_Entrez.gz

Property	Description
Ref_ID	Internal publication ID.
PubMed_ID	IDs used by PubMed to identify publications.
Orig_Ref	Reference string from HvrBase.
Journal_ID	Internal ID for bibliographic information about the journal issue.
Volume	The volume name or number of the journal, including any supplement information.
UI	MEDLINE Unique Identifier.
Page	Pages in the journal.
Publish_Date	Month and year of the publication.
Ref_String	Reference string from PubMed.
Authors	All authors of the publication.
Author	Single author name. Used to create a unique author ID.
Title	Publication title.
Abstract	Abstract of the publication.
Keywords	Terms used to characterize publications.
GenBank_Link	GenBank sequence IDs related to publications. The list is offered by PubMed.

Table 3.9: Bibliographic properties defined for HvrBase++.

3.5 The collection process

First the three phases of the collection process are described (Fig. 3.5) and then an example is given.

- The **retrieval phase** addresses the problem of finding new potential sequences from external sequence databases or publication databases.
- In the **extraction phase** properties are extracted from publications or sequences files.
- In the **transformation and inserting phase** properties are transformed and inserted into HvrBase++. A new global alignment is created for updated sequence sets.

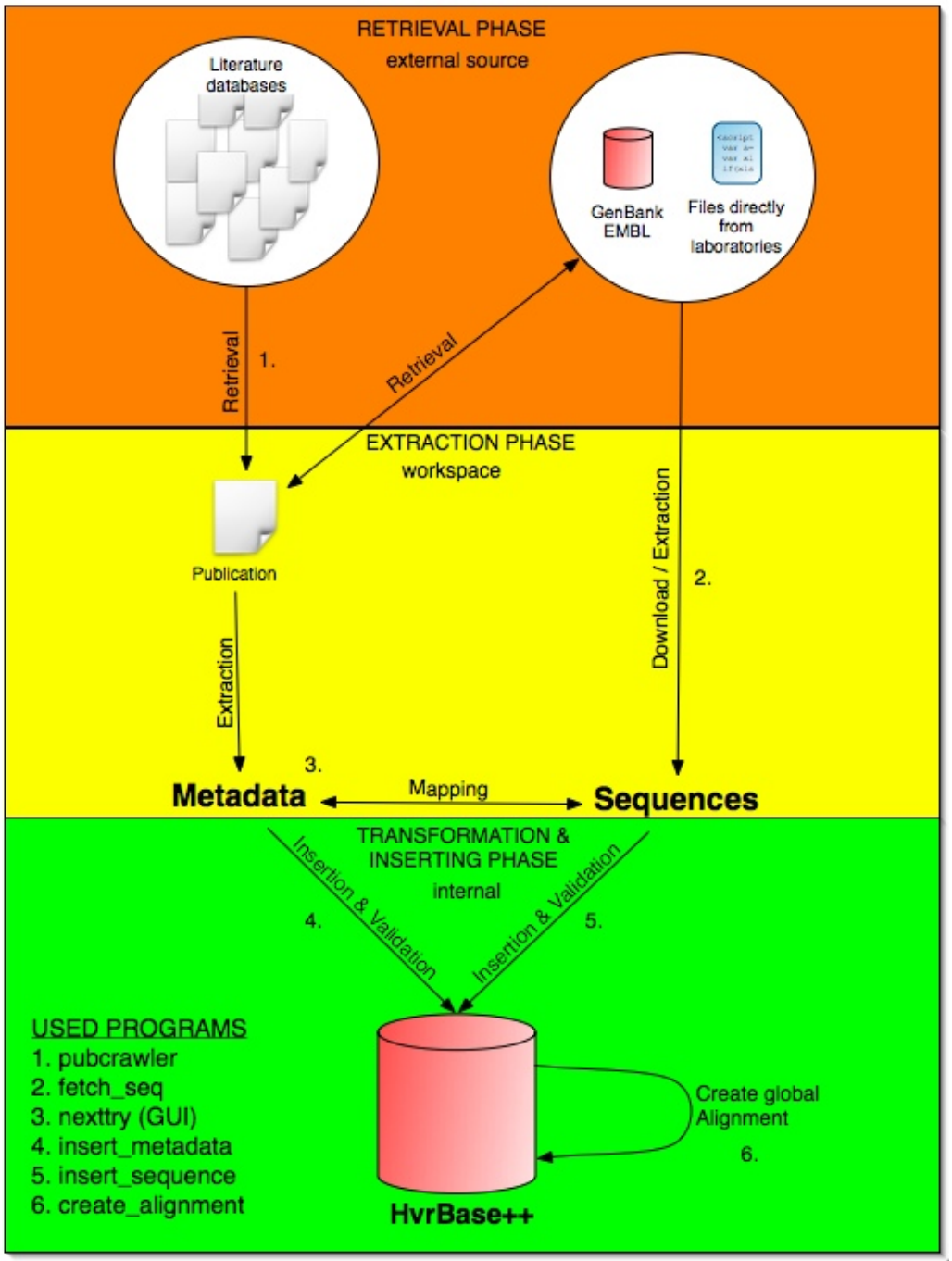


Figure 3.5: This figure shows the three phases of the collection process (colored rectangles). New sequences are mainly found by searching general sequence or literature databases. After retrieving new data, properties (metadata) are extracted. Then metadata is mapped to the corresponding individual sequences. After the insertion process new alignments are created. Used programs are numbered on the order of their application. All programs are selfwritten, except of *pubcrawler*.

3.5.1 Retrieval Phase

The simplest way of retrieving new sequences is, when sequences are submitted from laboratories or by advices from the community. However most new sequence sets are found by searching for publications, which present new insights in population history based on new sequences. For that reason literature databases such as PubMed are checked by different search terms or profiles for new publications. Tools like PubCrawler, a service which scans daily updates to the NCBI Medline (PubMed) databases, support the retrieval process. But only a handcrafted deeper look into those detected publications brings out new sequence sets. Thus publication and supplemental material must be analysed carefully by examing each of them more closely. Alternatively new sequences are found by searching directly in sequence databases like GenBank via keyword searches or BLAST searches. This approach also requires a deeper look to detect proper sequences, if those sequences fullfill the conditions of HvrBase++. For example, sequences related to medical studies typically do not provide information about individuals. Normally sequences are linked to publications and vice versa, so both approaches should achieve the same result.

3.5.2 Extraction Phase

The required properties are extracted from text, tables, captions or legends within publications or are extracted from sequences files. The way of presenting the data change for every data set. Also the best candidate, the rich tagged GenBank sequence format does not provide requested properties in a standardized way. Thus properties are still extracted by hand. If no information about donors are available they are requested from authors.

3.5.3 Transformation and Insertion Phase

The phase is split into two independent steps to uncouple transformation from the insertion process. In the transformation process the extracted prop-

erties and sequences are arranged into HvrBase++ specific input files. These files are then used to insert new entries. The benefit of uncoupling both process are, that the first step can be done by scientists familiar with the material, whereas the second step is executed by the HvrBase++ administrator. To find out an effective way of transforming data into the input formats two guided and one unguided approach are tested. In the guided approach specific HvrBase++ programs provide input masks, which help to create well-defined input files. One guided approach uses text based programs on the command line and the other provides a graphical user interface (GUI) and creates a specific XML input file (Fig. 3.6). Instead, in the unguided approach users create simple text based input files with common tools or programs. At first glance the GUI based approach seems to favorable, because it is easy to handle and the input files are well formatted. This approach is limited to moderately sized data sets. To transfer a set with hundreds of sequences, it is a time consuming job and the XML input file is also not easy to generate with common tools. Hence the current version of HvrBase++ no longer supports the GUI based approach.

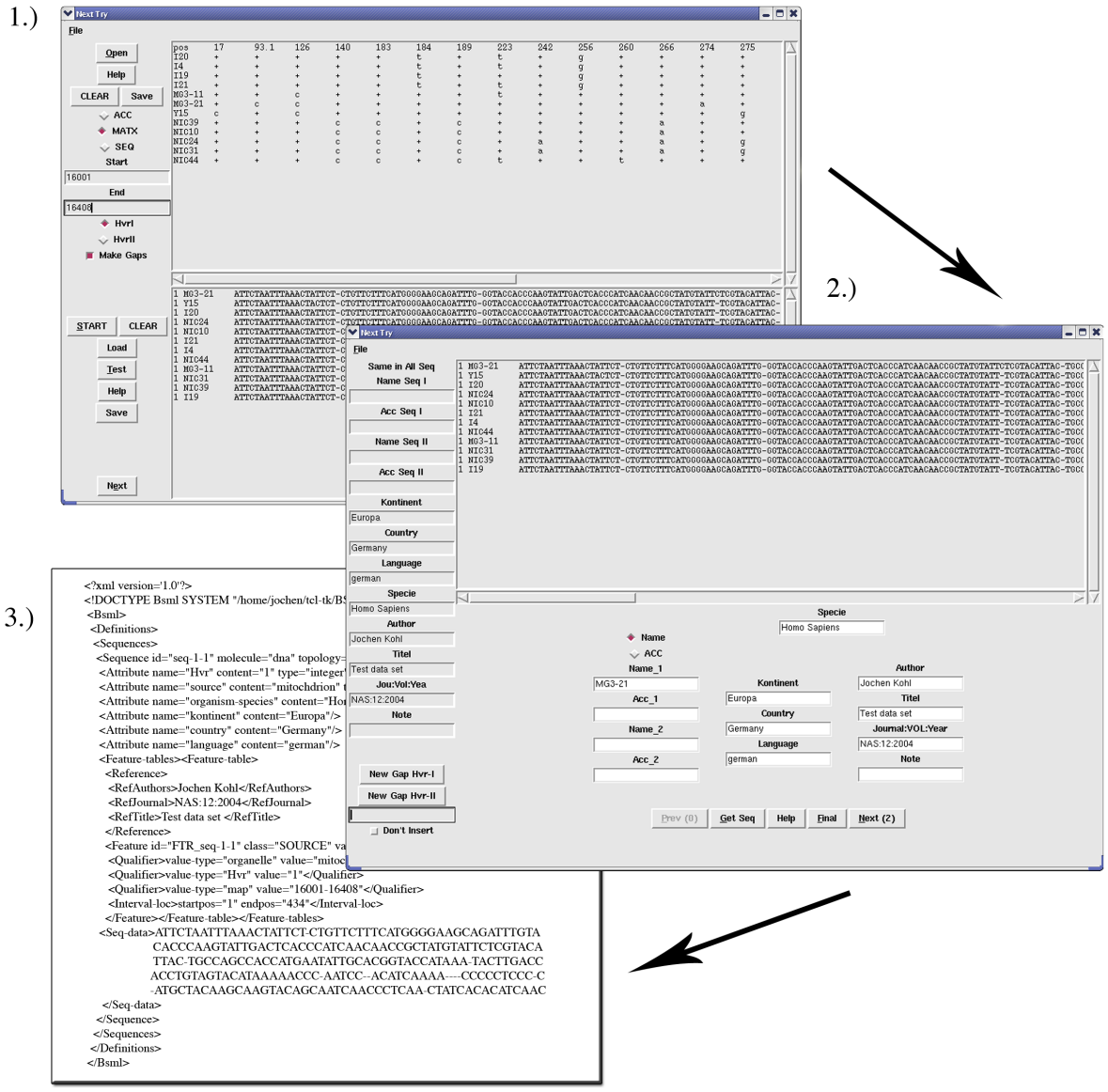


Figure 3.6: GUI-aided transformation processes for new data sets: The program helps to create an XML file which tags sequences with the corresponding properties. The process consists of the following three steps: get sequences, map metadata, generate XML file. 1.) shows the generation of HVR-I sequences from an 'Anderson matrix'. Such matrices can be loaded and edited into the upper frame and are converted into corresponding sequences (lower frame). Alternative sequences can be downloaded from GenBank via accession numbers or loaded directly from sequence files. 2.) shows the mapping process. For every sequence metadata are specified by lower left text fields. The left side allows setting of default values, that are automatically inserted in the lower left text fields. 3.) shows the resulting XML file for a sequence.

3.5.4 Collecting a huge data set with the unguided approach

The collection process is explained by the data set from TANAKA *et al.* (2004). This publication focus on the mitochondrial genome variation in Eastern Asian. A phylogenetic analysis of 942 Asiatic sequences are performed, whereof 672 Japanese mitochondrial genomes are newly sequenced. The only information from publication and supplemental material about donor individuals are, that they are unrelated Japanese from Tokyo (373) and from the Nagoya area (299). The GenBank files provide sequence names by the attribute '*isolate*' and a short distribution of individuals by the attribute '*isolation_source*'. The following cutaway shows an example entry in GenBank format for both attributes:

```
FEATURES             Location/Qualifiers
     source          1..16561
                     /country="Japan"
                     /db_xref="taxon:9606"
                     /mol_type="genomic DNA"
                     /isolation_source="Japanese diabetic patient
                      with angiopathy from Tokyo"
                     /isolate="JDsq0101"
                     /organelle="mitochondrion"
                     /organism="Homo sapiens"
```

Hence available individual information, consisting of geographic, species, population and publication properties, do not change for the whole set. The following table shows the extracted attribute value pairs together with a description.

Attribute	Value	Description
pubmed_id	15466285	PubMed ID
species	*Homo sapiens*	Species
cou_id	JP	Country ID (Japan)
con_id	L	Continent ID (Asia)
popu	Japanese	Population
gen_id	182	Genetic loci (mtDNA)

To convert these properties into the input file format, each attribute/value pair must be delimited by tab-stops and the attribute/value pairs itself must be divided by an equal sign. One line of the input file describes a record, which results in one row in the table *Metadata*. Hence only one line is needed to define one *Metadata* record for all individuals:

Input file:

```
pubmed_id=15466285 cou_id=JP con_id=L popu=Japanese species=Homo sapiens
```

The program INSERT_METADATA validates the properties, fetches the publication data from PubMed and inserts this record into HvrBase++. A resulting primary key is returned and used to map information to sequences by extending the keyword field for all GenBank files. Also the sequences properties, consisting of the genetic locus, the sequence name and a donor description, are coded in the keyword field as a single keyword, which starts with the string 'hvrbase:'. The properties are divided by tab-stops and the attribute/value pair by an equal sign. The upper part of a modified GenBank file is shown in the following:

```
LOCUS    AP008917  16561 bp   DNA  circular PRI 16-JUL-2005
DEFINITION  Homo sapiens mitochondrial DNA, complete genome, isolate:
            JDsq0101.
ACCESSION   AP008917
VERSION     AP008917.1  GI:61288310
KEYWORDS    hvrbase:    name=JDsq0101 met_id=3492   gen_id=182
            sam_desc=Japanese diabetic patient with angiopathy from Tokyo
SOURCE      mitochondrion Homo sapiens (human)
  ORGANISM  Homo sapiens
            .....
```

Then the sequences are inserted via the program INSERT_SEQ. After that the program MAKE_ALIGNMENT is used to create and insert alignments for updated sequence sets.

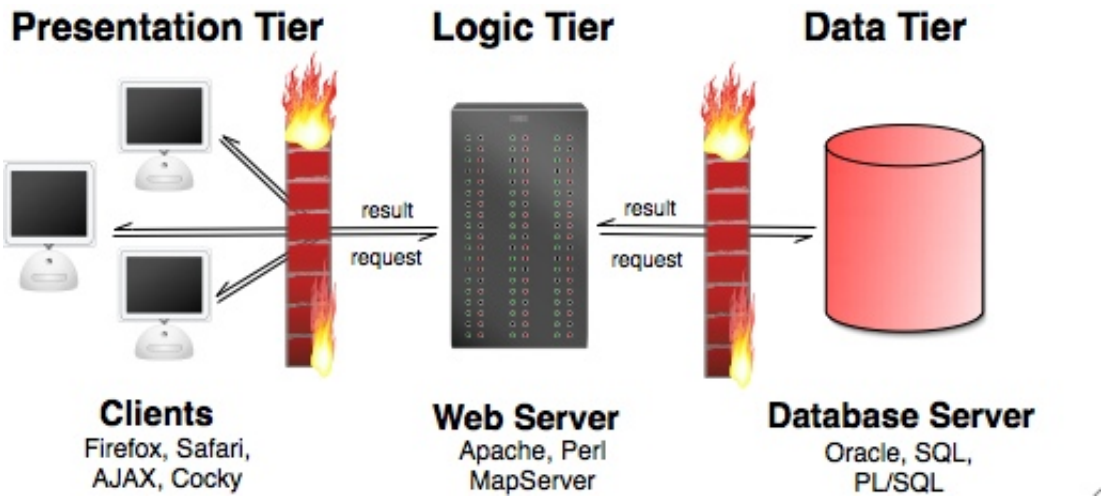


Figure 3.7: Three-tier architecture implemented for HvrBase++. Every tier is separated through a firewall from each other and runs on its own machine.

3.6 Implementation of the web application

The implementation of HvrBase++ followed a three-tier client-server architecture consist of the client, the web server and the database server (Fig. 3.7). In the following the basic concepts of the three nodes are presented and then the implementation of special features are explained.

3.6.1 Client

The web browser requests the web server and receives HTML-, JavaScript code to render the web site. An example for the JavaScript functionality is given for the *Publication search* form (Fig. 3.8). This form provides two search modi (normal/advanced) where a JavaScript function manages switching between both modi without reloading. Query parameters are sent via a background process to a server side script (*pubsearch_response.pl*). This script generates a *select*-statement based on the parameters and queries the database. The processed results are sent back to the client, which then integrate the result into the current web site.

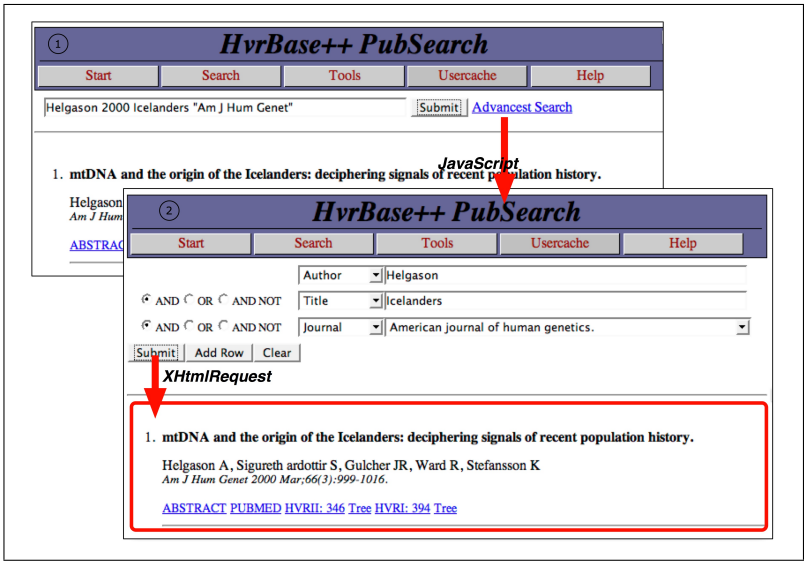


Figure 3.8: Normal (1) and advanced (2) *Publication search* forms. In the normal search mode terms are specified by a single text field. The advanced mode allows sophisticated specification of the search terms. JavaScript is used to change between both modi and to request the search results by a *XHtmlRequest*.

3.6.2 Database server

The database server runs on a separate machine, which is not directly accessible from clients outside the local network. Only the web server has the rights to access the database server through the firewall. One database account is used to collected new data and to retrieve the collection via the web server. But the collection process directly accesses the normalized tables via SQL-statements, whereas the web server mainly works on the de-normalized materialized view over a 'stored procedure' (Fig. 3.9). The materialized view presents the current offered version, which will be updated whenever a new version is released. However, for specific queries and to access tables used for the application management the web server access the database directly via SQL. The implementation of the 'stored procedure' is explained in the following:

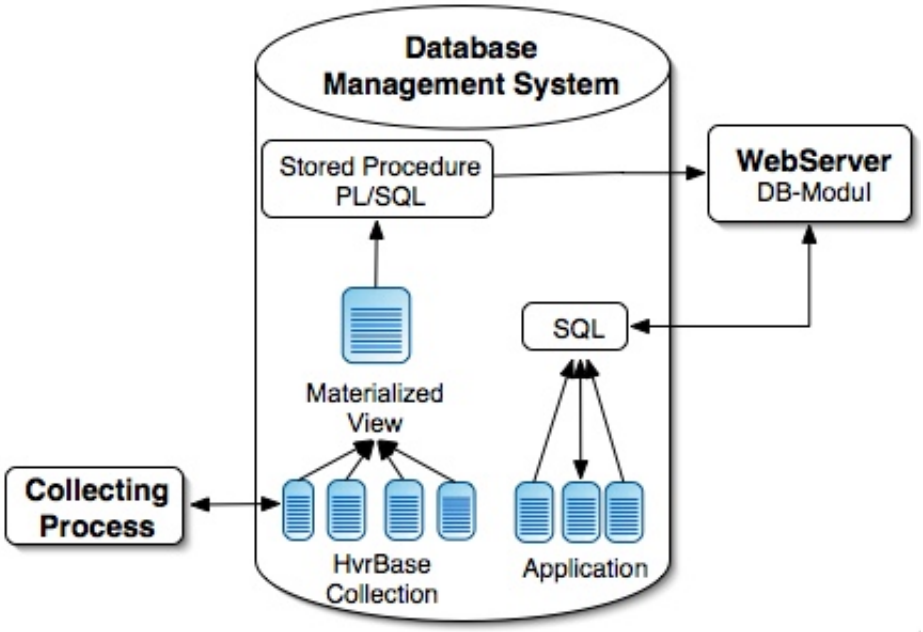


Figure 3.9: Data flow and organization of the database server is shown. New data sets are normalized and inserted into tables by the collecting processes. The web server work on a fixed version of HvrBase++ managed in a materialized view. If a new version is released the materialized view is updated. Moreover the web sever accesses the individual properties not directly instead over a 'stored procedure'. For the application workflow the web server needs read/write access for some tables related to the application. Arrays show the direction of data flow.

Implementation of the stored procedure

This technique is used in HvrBase++ to handle the query for individual information and sequences. The PL/SQL function *next_search* accepts 50 parameters in a specific order to create the *select* statement dynamically. Depending on the parameter the materialized view *Main_Info* is joined with one of the three sequence tables *Seq*, *Alignment* or *Malignment*. Parameters specify the *where-condition* of the query. If a parameter consists of multiple entries, these entries are divided by pipe signs (|). Mostly only few parameters are used to restrict the query and the rest is set to NULL. The work of the PL/SQL function is explained by a short example, called *searchView* (A.3). It requires six parameters, where the first parameter restricts column *country* and the fifth parameter column *gene* of the materialized view *Main_info*. To get properties for all individuals from England and Germany, where HVR-I sequence exists the following function call must be executed as SQL statement:

```
BEGIN
     searchView('England|Germany',NULL,NULL,NULL,'HVRI','NULL');
END;
```

The created and executed SQL statement is the following:

```
SELECT * FROM main_info
 WHERE country in ('England','Germany') AND m.gene in ('HVRI');
```

Such an SQL statement cannot be executed as a trivial SQL statement in Perl. Instead Perl must handle a database cursor, which is something like a pointer or references to the statement call. The function return a cursor, which is used in a second database call to fetch the results:

```
 1 my $function = "BEGIN
 2 :cursor:=searchView('England|Germany',NULL,NULL,NULL,'HVRI',NULL);
 3  END;"
 4
 5 my $sth1 = $dbh->prepare($function);
 6 $sth1->bind_param_inout(":cursor",\$sth2,1,{ ora_type=>ORA_RSET});
 7
 8 $sth1 ->execute();
 9 while (my @row = $sth2->fetchrow_array) {
10   print "@row\n";
11  }
```

This example consists of three major parts. First the function call is defined (Lines 1-3). Then the function call is prepared and the cursor is defined (Lines 5-6). After that the function is executed and the result are fetched from the database server via the cursor (Lines 8-11).

3.6.3 Web Server

HvrBase++ use the 'Apache' web server for the client communication and Perl to create dynamic HTML content. Different Perl packages were written to implement HvrBase++ functionality (Tab. 3.10). In the next two subsections the implementation of the *TreeView* window and the generation of interactive world maps is described.

Implementation of the TreeView window

HvrBase++ gives the opportunity to visualize own phylogenetic trees or automatically reconstructed trees based on the current selection. To generate a phylogenetic tree for a private collection a pipeline of phylogenetic programs is integrated into HvrBase++. Four steps are required:

- Creating an alignment from selected sequences
- Calculating a distance matrix from the alignment
- Reconstruct the tree depending on the distance matrix
- Visualize the tree

Package	Type	Description
Hvrbase::dbi	model	Provide a general database access and a special access to the HvrBase collection.
Hvrbase::weboutput	view	Functions to create web sites and content like sequence files and the world maps.
Hvrbase::scaffold	view	Maintain the headers and tails of web sites.
Hvrbase::config	controller	Hold application parameters like paths.
Hvrbase::hvrbase_object	controller	Receive and manage client parameters.
Hvrbase::_data_array	controller	Provide access functions and data model for the hvrbase_object package.

Table 3.10: List of Perl packages from web server applications.

Outside a web application the time required for each step is not so important and computationally intensive programs can be used. But for web applications fast response times are expected, so that the pipeline is optimized to be fast. The aim is to reconstruct one tree, that should give a first expression of the data.

For the tree calculation, the alignment consists of a subset of the existing sequence alignments of a genetic loci. To compute the distance matrix and the tree the *Phylip* programs '*dnadist*' and '*neighbor*' integrated (Felsenstein, 2005). The Embassy package provides a wrapper to use Phylip programs via command line arguments (Rice *et al.*, 2000). This allows a better integration of the two Phylip applications into the tree reconstruction pipeline. For the distance matrix calculation the F84 substitution model is taken(Kishino and Hasegawa (1989), Felsenstein and Churchill (1996)).

For tree reconstruction the Neighbor-Joining method (Saitou and Nei, 1987) implemented in *neighbor* is used. For tree visualization an adapted version of the Perl package *BIO::Phylo* is utilized to draw a tree in SVG (scalable vector graphic) format. JavaScript is then used to modify SVG graphic in the web browser, like selecting a node or relabel the tree. The internal sequence id (*seq_id*) is used to identify nodes and request properties from the web server.

Integration of the *MapServer* environment to generate world maps

The *MapSearch* window presents dynamically created world maps to interact with the users. For that reason the *MapServer* development environment for building spatially-enabled internet applications is integrated into HvrBase++. This environment is not a full-featured global information system (GIS) system. Instead it renders spatial data like maps, images, and vector data for the web. The underlying program, called *MapServer* uses the configuration file ('*Mapfile*') to render graphics from geospatial vector data. A '*Mapfile*' specifies the vector data and defines layers, which specifies alternative representations of the data (Fig. 3.10). In HvrBase++ layers are used to visualize continents or countries borders and properties. For accessing the geospatial vector data the built-in 'ESRI Shapefile' format is used. The 'ESRI Shapefile' is a popular geospatial vector data format for geographic information systems software. Those 'shapefiles' commonly refers to at least three files with the suffixes '.shp', '.dbf' , '.shx' and a common prefix like 'world'. The '.shp' file stores the feature geometry and the 'shx' stores the index of the feature geometry. The '.dbf' file is an xBase database file that handles properties of geographic features. These properties can be used to render a map. For example in the HvrBase++ xBase file the attribute '*Continent*' is defined for each country. To colorize continents each country belongs to a continent is colored with the same color. For that reason a class for ever continent is defined in a layer of the *Mapfile*. The *EXPRESSION* parameter specifies the value, in this case the continent name, of the attribute defined by the *CLASSITEM* parameter (Fig. 3.10). To draw alternative maps, properties of the database file or entries of the *Mapfile* must change. Because HvrBase++ data is not managed in xBase files, the *Mapfile* must be modified to draw maps based on search results. Hence the wrapper MapScript is used to modify the *Mapfile* dynamically.

If a user selects a geographic region via a map, the web server gets only the click point and not a region identifier. The click point is an XY-coordinate that belongs to the shown image and not the spatial coordinates of the *Shape-*

file. Thus, the coordinates must be converted to the spatial coordinates before getting the region identifier from the *MapServer*. The identifier is then used to query HvrBase++ to represent data based the spatial information. A new layer is created and each region is colorized depending on the number of sequences found for the region.

```
01 MAP
02  NAME WORLD
03  STATUS ON
04  SIZE 1200 700
05  EXTENT -179.999 -89.901 181.796 84.929
06  UNITS DD
07  SHAPEPATH      "../data"
08  IMAGECOLOR     211 228 255
09  LAYER
10    NAME         twoContinents
11    DATA          world_shp.shp
12    STATUS     OFF
13    TYPE          POLYGON
14    CLASSITEM "CONTINENT"
15    LABELITEM "CONTINENT"
16     CLASS
17        EXPRESSION 'Asia'
18        COLOR 232 0 0
19     END
20     CLASS
21        EXPRESSION 'Europe'
22       COLOR 0 232 0
23     END
24  END
25 END
```

Figure 3.10: This box shows a truncated *Mapfile* used to render a *world* map. A *Mapfile* is hierarchically structured and begins with the global *MAP* definition. All other objects like layers or labels are siblings of *MAP*. In the example only the layer 'twoContinents' is defined, which colorizes the two continents Asia and Europa (Lines 09-23). The global parameter *Size* defines the image size and *EXTENT* specifies the visible geographic region based on the shapefile. If an object is rendered or not is specified by the parameter *STATUS*.

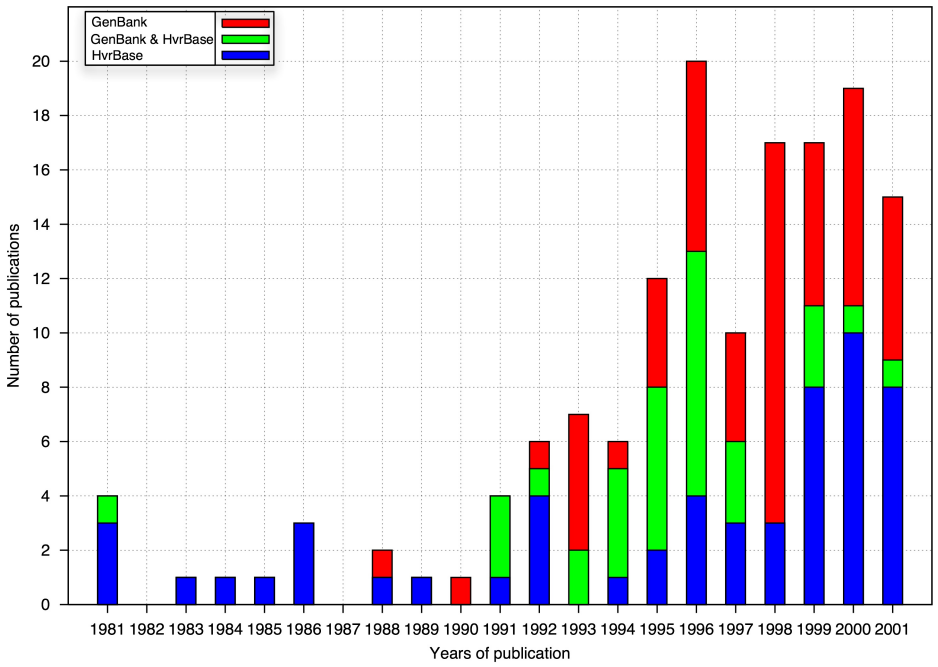


Figure 3.11: Comparison between available HVR-I and HVR-II sequences via publications for GenBank and HvrBase. Publications from GenBank found by BLAST searches for HVR-I and HVR-II sequences. Each bar presents the number of publications, which describe mitochondrial sequences.

3.7 Results

3.7.1 Qualities of HvrBase sequences

To ensure the quality of HvrBase++ sequences from HvrBase were mapped and compared against corresponding sequences from GenBank. Only 2,061 of 13,177 HVR-I and HVR-II sequences could be mapped, because sequences released before 1990 mostly were not found in GenBank; instead these sequences are offered within publications. Since 1991 the amount of available HVR-I and HVR-II sequences increased and more and more sequences are also deposited in GenBank (Fig. 3.11). Comparison of these 2,061 sequences leads to 157 discrepancies between GenBank and HvrBase. Only 13 discrep-

ancies are related to typos caused by HvrBase. The more general problem are discrepancies caused by updated sequence versions in GenBank (56) and unprovable discrepancies (70) caused by the absence of further reference sequences (Fig. 3.2). Based on these findings and the observation that nowadays mostly all sequences are also stored in GenBank, the new collection process concentrates on sequence available by GenBank and an automated update procedure based on GenBank is implemented. That procedure ensures equality of sequences collected in HvrBase++ and corresponding sequences from GenBank.

Moreover HvrBase sequences are checked for misleading sequenced NUMTS. But only short similar regions between 30 bp and 250 bp are found, which are not identical to one of the HVR-I or HVR-II sequences.

3.7.2 Reorganization of the database

The individual properties '*continent*', '*country*' and '*language*' are mapped to official lists and the '*reference*' string is replaced by detailed bibliographic entries. These improvements facilitate data exchange and comparison within HvrBase++ and with other data sources. But it required the redesign of the database schema. The new arrangement for the core data looks like a 'snow flake' schema with one centered table that aggregates all sequence information from surrounding normalized tables. These normalized tables are mainly used for the collection process, where the normalization helps to present a well-defined state. Instead a decomposed 'materialized view' is used to reduce the complexity of queries and speed up queries against a stable version of HvrBase++ from the web application. Further the web application access these data over a stored procedure, which enriches the security of the DBMS by aggravating SQL-injections.

Moreover, it is now possible to define haplogroups and match these haplogroups automatically to collected sequences. These findings can then be used to query sequences or to present the world wide distribution of haplogroups depending on the collection (Fig. 3.14). At the moment 51 mito-

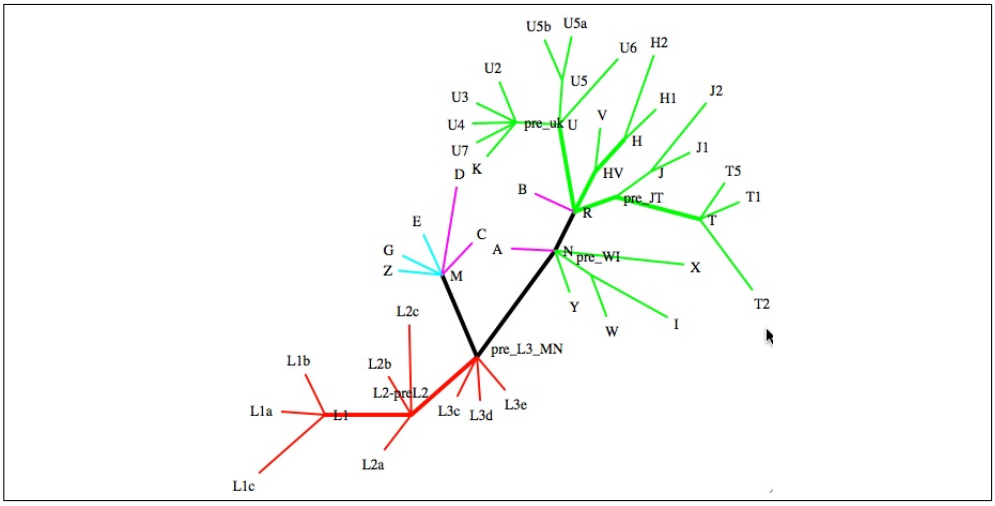


Figure 3.12: Haplogroups defined by HvrBase++. Red labeled branches belong to macrohaplogroup L, which is only found in Africa. Black labeled branches represent the macrohaplogroups M, N and R. M and N are involved from haplogroup L3 in East Africa. All non-African haplogroups are based on these two haplogroups. Green branches belong to haplogroups mostly found in Europa. Haplogroups A, B, C and D (purple) found predominate in native Americans (Wallace, 1995).

chondrial haplogroups are defined. But it is not restricted to mitochondrial haplogroups. Figure 3.12 shows a phylogenetic tree for the defined mitochondrial haplogroups based on 67 defined diverging sites (haplotypes).

To see how the 51 haplogroups are distributed, 749 human and 1 chimpanzee complete mitochondrial sequences are selected. Then, their were trees reconstructed for complete sequences, for HVR-I sequences and for genomes excluding the control region (Fig. 3.13). Human radiation patterns (Fig 2.5) are well preserved in the slowly changing coding regions of the mitochondrial genome, whereas the non-coding HVR-I sequences do not clearly provide these patterns. The higher mutation rate of the control region blur the radiation patterns but studying more recent events the combination of coding and non-coding sequence parts provide a detailed view. Another prospective of the relation between haplogroups and geographic patterns is offered by the new HvrBase++ functionality of mapping sequence properties to geographic maps. Figure 3.14 presents the haplogroup distribution for all complete mi-

tochondrial genomes. The proposed correlation between haplogroups and geographic origin is recognizable but biased by distribution of sequences (see next paragraph). However, it provides an interesting function to explore the collection via geographic maps.

3.7.3 Collection process

The new database schema and the large amount of sequences made it necessary to replace the handcrafted collecting process by an efficient semi-automated process. This process consists of a database independent step, where data is extracted and transformed into a HvrBase++ specific format, and a database dependent insertion step. Different approaches are tested to transform data into HvrBase++ format. The GUI based approach was easy to use but inefficient for huge data sets. The unguided approach provides more flexibility to adapt the specific case. Thus, the new collection process supports the unguided approach.

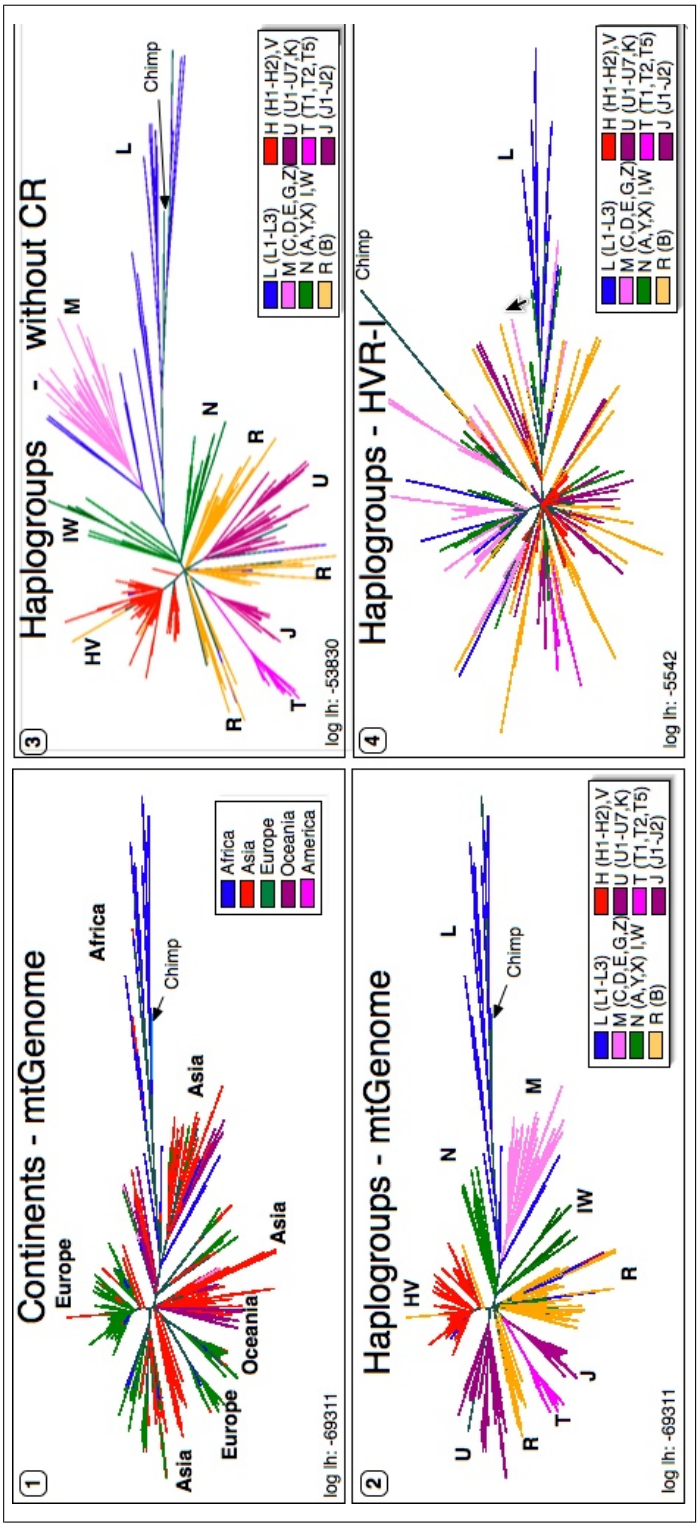


Figure 3.13: Tree 1 and 2 are equal and consist of 750 unique complete mitochondrial genomes (mtGenomes). Tree 3 consists of 670 unique mtGenome sequences extracted from the mtGenome set, where the control region (CR) is excluded. Tree 4 consists of 387 unique HVR-I sequences extracted from the mtGenome set. Tree 1 is colorized by *continent* information. Trees 2-4 are colorized by *haplogroup* information. Sequences sets are aligned with ClustalW (Thompson *et al.*, 1994), trees are reconstructed with pIQPNNI (Minh *et al.*, 2005) and visualized with HyperTree (Bingham and Sudarsanam, 2000). Parameter used by pIQPNNI: substitution model: HKY85, rate heterogeneity: uniform substitution rate, 10.000 iterations.

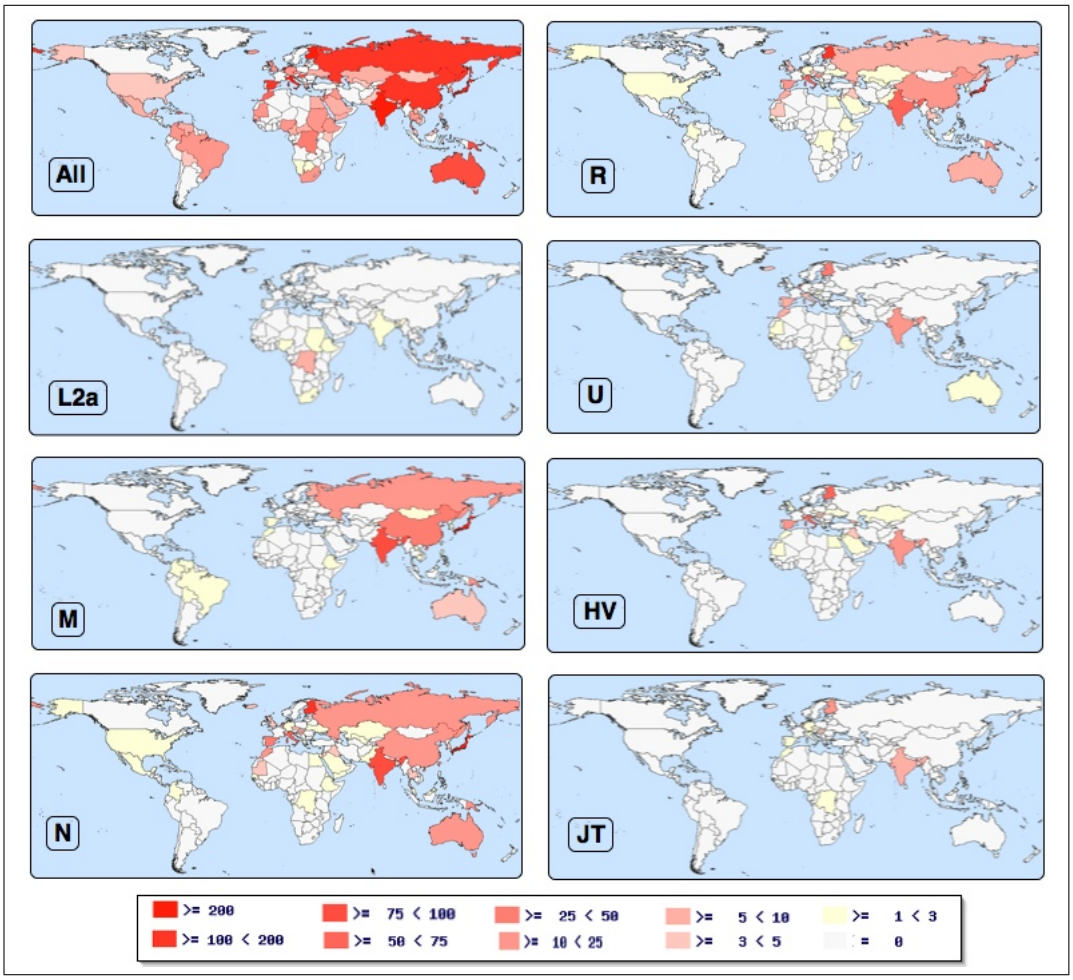


Figure 3.14: Haplogroup World Maps. Shown are the distributions of nine haplogroups (L2a, M, N, R, U, H and V, J and T) by seven world maps generated with HvrBase++. The colors define the amount of sequences belonging to a haplogroup. Label 'All' denotes a map for all haplogroups.

	Continent	Country	Species	Population	Language SIL/ISO	Publication
HVR-I	6	93	15	312	125/88	109
HVR-II	6	65	9	154	89/50	53
mtGenomes	6	63	2	162	62/52	22
ChrX	6	41	4	89	51/48	4
Autosomal	5	30	3	48	24/25	8
Total	6	110	17	515	147/108	132

Table 3.11: The table shows how many different property entries are defined for each loci. Table head defines six property names and the first column the loci names. The numbers are the amount of entries. The amounts for a single property, represented in a row, are not additive.

3.7.4 Current HvrBase++ collection

HvrBase++ supports the dynamic extension of further genetic loci to follow trends in human history studies. One of the observed extension was the integration of mitochondrial genomes (2,416). Sequences from six Y-chromosomal loci (205) and four autosomal loci (204) are added. Present 8,081 new sequences are added into HvrBase++ (Table 3.12).

To give an impression of the expressiveness of the collection the diversity of used property entries is presented in Table 3.11. Although the HVR-I set consists of 13,873 individual sequences, these individuals belong only to 93 countries. Some geographic regions predominate and for other regions only few sequences are available. For example, in the mitochondrial genome set sequences from Japan (679), Finland (181) and India (163) predominate and 44 of 63 countries are presented with less than ten sequences. The high variation within the population property is due to the unstandardized entries reflecting the description used in the publication. A more realistic view on the population level reflects the standardized language properties, which is related to the population. The difference in language variation depends on different granularity of both language standards, with a more specific SIL standard.

Gene/Region	Location	Individuals	Haplotype sequences	Average sequence length
HVR-I	mtGenome	13,873	7,161	362
HVR-II	mtGenome	4,940	2,136	332
Complete mtGenomes	mtGenome	2,416	2,416	16,195
XQ 13.3	ChrX	71	71	10,162
CH1 membrane protein	Chr1	61	61	9,622
Beta globin	Chr11	79	33	2,584
Melanocortin 1 receptor (mc1r)	Chr16	56	56	6,584
Tumor necrosis factor ligand superfamily 5 (tnfsf5)	ChrX	42	42	5,239
Ribonucleotide reductase M2 pseudogene (rrm2p4)	ChrX	42	10	2,387
Factor ix gene, intron 4	ChrX	41	11	3,729
Pyruvate dehydrogenase E1-alpha-subunit (pdh1)	ChrX	8	8	1,762
Lipoprotein lipase gene (LPL)	Chr8	8	8	922
Amelogenin X chromosome	ChrX	1	1	5,323

Table 3.12: The amount of sequences for the different genetic loci in HvrBase++. The column '*Location*' specifies the chromosome (Chr) or mitochondrial genome where the genes/regions belong to. '*Individuals*' represents the number of donor individuals. '*Haplotype sequences*' defines the number of diverting sequences.

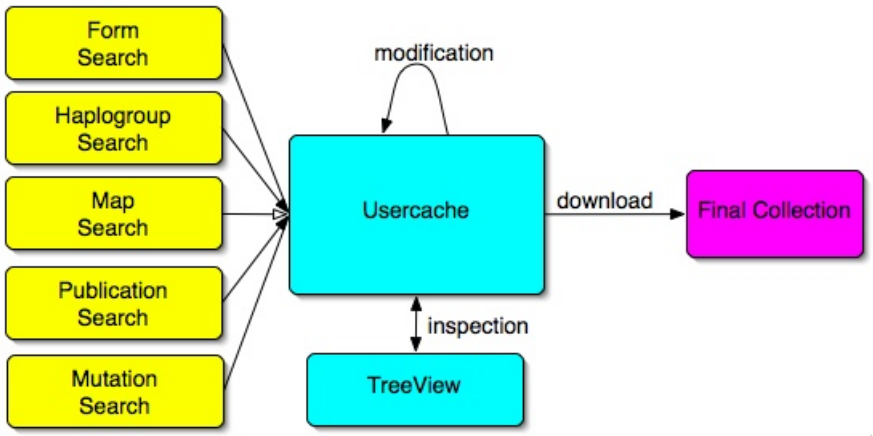
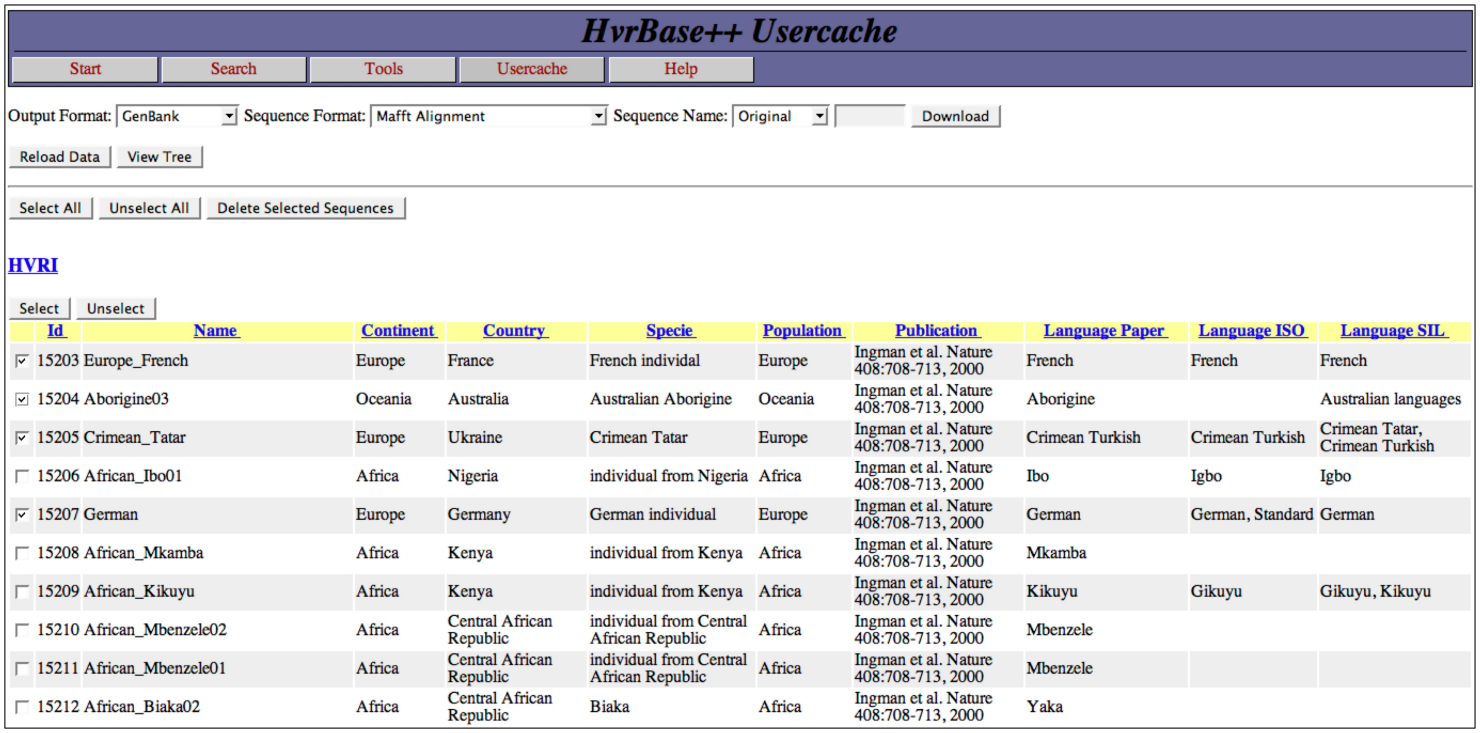


Figure 3.15: The main functionality of the HvrBase++ service. Yellow rectangles specify the five different search modes, the lower cyan rectangle caching and inspections facilities and the magenta rectangle the collection download. Arrows mark the information flow between tasks. A typical workflow consists of independent queries using different search modes, followed by inspection and modification of the collection and results in collection download.

3.7.5 The new Web interface of HvrBase++

The serialized workflow of HvrBase, consisting of query and result, is enhanced by a caching mechanism, which provides the basis of the HvrBase++ web application (Fig. 3.15). Users have now the possibility to build a collection resulting from many independent queries. This private collection is managed in the '*Usercache*' window (Fig. 3.16) and can be shared with co-workers. The '*Usercache*' provides for every genetic loci properties of selected sequences in sortable tables. Sequences can be marked for collection removing and for tree visualization. The whole private collection or parts can be downloaded in six formats (GenBank, Fasta, Phylip, TSV, XML, HTML) from '*Usercache*'. A phylogenetic tree visualization window '*Treeview*' gives an first impression of sequence relations and supports the private collecting process (see next paragraph). The following five search forms are implemented to add sequences to the '*Usercache*':



HvrBase++ Usercache

Start | Search | Tools | Usercache | Help

Output Format: GenBank | Sequence Format: Mafft Alignment | Sequence Name: Original | Download

Reload Data | View Tree

Select All | Unselect All | Delete Selected Sequences

HVRI

Select | Unselect

	Id	Name	Continent	Country	Specie	Population	Publication	Language Paper	Language ISO	Language SIL
☑	15203	Europe_French	Europe	France	French individal	Europe	Ingman et al. Nature 408:708-713, 2000	French	French	French
☑	15204	Aborigine03	Oceania	Australia	Australian Aborigine	Oceania	Ingman et al. Nature 408:708-713, 2000	Aborigine		Australian languages
☑	15205	Crimean_Tatar	Europe	Ukraine	Crimean Tatar	Europe	Ingman et al. Nature 408:708-713, 2000	Crimean Turkish	Crimean Turkish	Crimean Tatar, Crimean Turkish
☐	15206	African_Ibo01	Africa	Nigeria	individual from Nigeria	Africa	Ingman et al. Nature 408:708-713, 2000	Ibo	Igbo	Igbo
☑	15207	German	Europe	Germany	German individual	Europe	Ingman et al. Nature 408:708-713, 2000	German	German, Standard	German
☐	15208	African_Mkamba	Africa	Kenya	individual from Kenya	Africa	Ingman et al. Nature 408:708-713, 2000	Mkamba		
☐	15209	African_Kikuyu	Africa	Kenya	individual from Kenya	Africa	Ingman et al. Nature 408:708-713, 2000	Kikuyu	Gikuyu	Gikuyu, Kikuyu
☐	15210	African_Mbenzele02	Africa	Central African Republic	individual from Central African Republic	Africa	Ingman et al. Nature 408:708-713, 2000	Mbenzele		
☐	15211	African_Mbenzele01	Africa	Central African Republic	individual from Central African Republic	Africa	Ingman et al. Nature 408:708-713, 2000	Mbenzele		
☐	15212	African_Biaka02	Africa	Central African Republic	Biaka	Africa	Ingman et al. Nature 408:708-713, 2000	Yaka		

Figure 3.16: This figure shows the '*Usercache*' window of HvrBase++, which allows the management of the private sequence collections. Sequence properties for genes are presented in sortable tables. In this example the properties for ten HVR-I sequences are shown. Selected sequences can be downloaded, removed from the collection or visualized as phylogenetic tree.

FormSearch

It allows the selection of sequences by HvrBase++ properties. An alternative search form is available that does not require JavaScript to be compatible with old web browser. The alternative FormSearch directly lead to a download window.

HaplogroupSearch

This search form allows the selection of mitochondrial genomes by haplogroups. Sequences from a selected haplogroup are clustered by the geographic properties '*country*' or '*continent*'. Sequences of a cluster can be added to '*Usercache*'.

MutationSearch

This is a specialized search form for mitochondrial sequences, where sequences are found by variable sites. For a specified region, which can be given accordingly to Anderson numbering or by the real alignment positions, all variable sites are shown in a table. From there all sequences belonging to a variation can be added to the private collection.

PublicationSearch

PublicationSearch provides two search modi for sequences by bibliographic information (Fig. 3.8). The query result is a list of publication entries. Further sequences from the publication are presented. These sequences can be added to the private sequence collection.

MapSearch

Search results are visualized in a world map (Fig 3.17). Hence, it is possible to see how a property is distributed over the world. By selecting a country the available properties for the country are shown in detail and sequences can be added to the private collection.

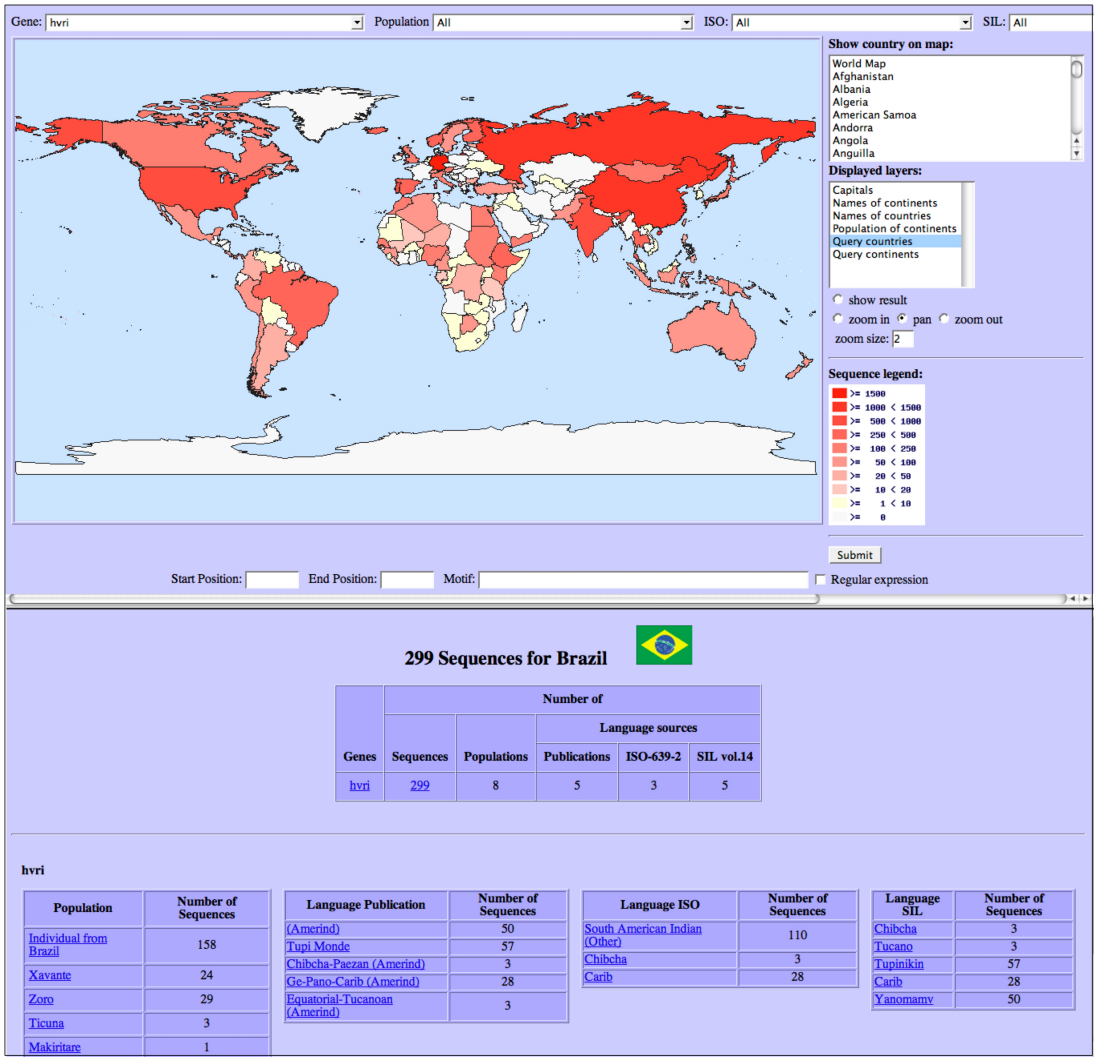


Figure 3.17: Geographical map interface in HvrBase++. The upper frame contains elements for searching sequences, the search results are displayed in the map and at the bottom. A countries color represents the number of sequences for a given gene. The main table shows the results of all available genes for a selected country. Additional information for each gene is displayed in separate tables. Sequences are accessible by selecting them from the table.

Furthermore the sites '*ConvertNumbering*' and '*GenomeMap*' are implemented. The '*ConvertNumbering*' site convert sequence positions between Anderson numbering and real alignment position off HvrBase++ alignments. The '*GenomeMap*' site provides additional facts about the mitochondrial genome.

TreeView – Interactive phylogenetic trees

HvrBase++ gives the opportunity to visualize own phylogenetic trees or automatically reconstructed trees based on the current selection. To work with user trees, these trees must be in Newick format and the used sequence names must be HvrBase++ sequence IDs (*seq_id*). The process of automatically reconstructing trees is optimized to be fast and should only give a first expression of the sequence relations. Existing HvrBase++ properties can be used to relabel tree nodes, which supports the understanding of the tree. Hence, uninteresting sequences can be directly deleted from the private collection by deleting the corresponding node from the tree. The usability and functionality for this window is explained in Figure 3.18.

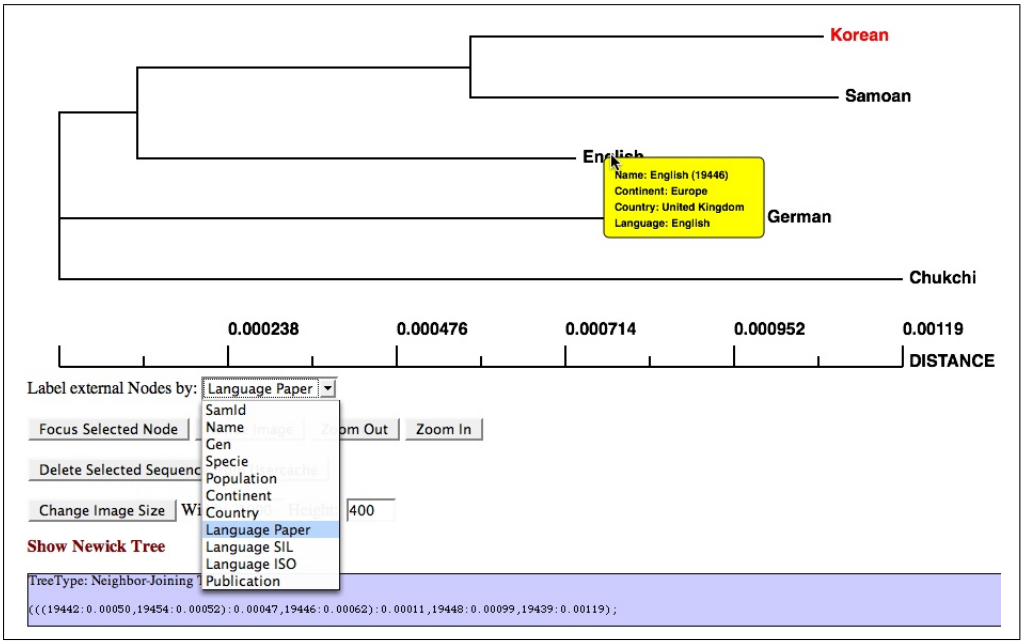


Figure 3.18: The screenshot shows the '*TreeView*' window. A tree of five complete mitochondrial genomes and the distance is shown in the upper part of the window. The five sequences belong to individuals from Korea, Samoa, England, Germany and Russia. Nodes are labeled by the property '*language*'. The red color of node 'Korean' indicates a selected node. A selection can be used to remove the corresponding sequence from private collection or to focus this part of the tree. A tool-tip pops-up when moving over a node label. The tool-tip provides the sequence name, geographic information and the tongue of the individual (*yellow box*). The bottom part of the window provides buttons for user interaction and presents the reconstructed tree in Newick format. (*blue rectangle*).

Chapter 4

TreeDB

The aim of this project, called TreeDB, is the assistance of biological data management in laboratories, enrichment of the cooperation between scientists and visualization of such data. TreeDB gives the opportunity to establish private sequence databases, which can be offered via a web-interface and supports data sets, which presents subsets of the private sequence database. These data sets further provide alignments and phylogenetic trees resulting from analysis based on these sequences. In contrast to similar programs like ARB[1], Geneious[2] or AceDB[3], TreeDB supports a history function to trace back the analyzing process and a complex data structure to describe sequence properties. Experiences made with HvrBase++ are used to work out requirements and needs for TreeDB and help to avoid typical pitfalls. But a completely new attempt was needed to satisfy this more general useable application. For the general case it is impossible to predefine the needed sequence properties nor the number of properties, without limiting usability of the system. Hence TreeDB is designed to handle an unlimited number of free definable properties. The mechanisms of handling sequences are quite similar in both databases applications, the challenge are unclear sequence properties, a more flexible way for building new data collections and new

[1] http://www.arb-home.de
[2] http://www.geneious.com
[3] http://www.acedb.org

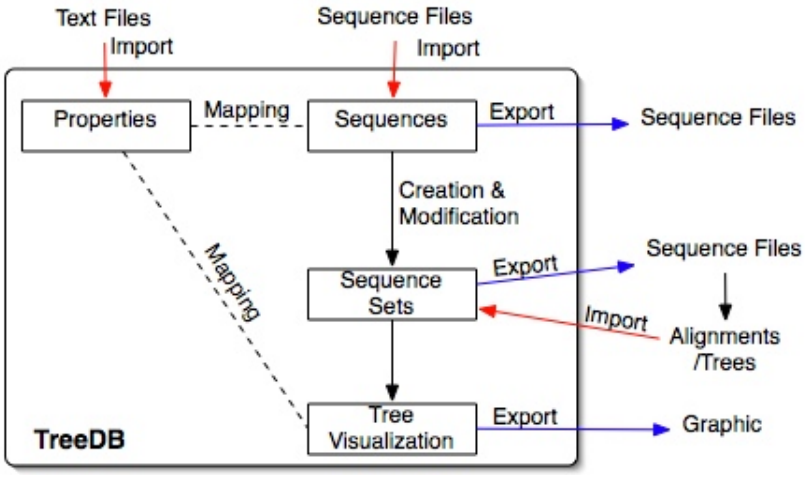


Figure 4.1: Data flow and interaction between user and TreeDB. The large rectangle represents the application. Red arrows indicate import interfaces whereas blue arrows indicate export interface. Dashed lines illustrate the opportunity to map user-defined properties to sequences or node labels of phylogenetic trees.

software requirements to make the application maintainable for biologists. The aims of TreeDB are:

- Professional sequence data management for biologists
- Less configuration and easy to use
- User-defined properties and relations
- Enhance the cooperation between distributed work groups
- Versioning control
- Enhanced tree visualization

4.1 Functionality and data flow

To point out the basic idea behind TreeDB the supported model of sequence set generation and analyses is characterized (Fig. 4.2). Sequence data comes directly from laboratories or from public sequence databases. Typically sequences from general sequence databases are available in a common sequence

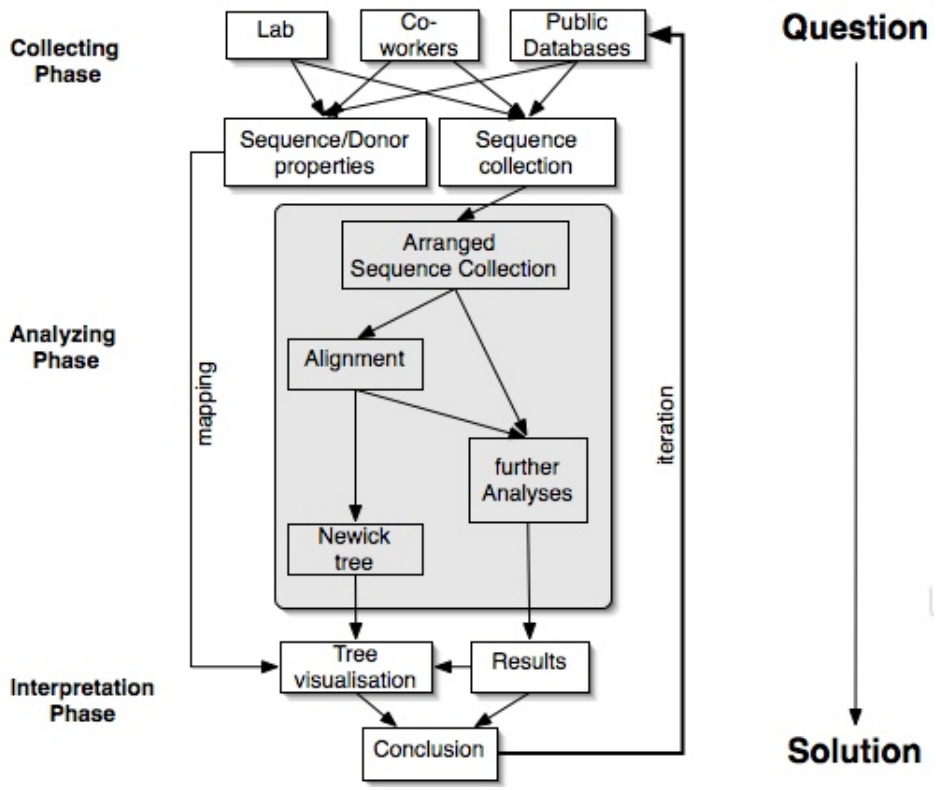


Figure 4.2: This schema represents the proposed sequences analyzing model used to build TreeDB functionality. The whole analysis is motivated by an open question and may end up with a solution. To reach this aim first sequences and additional cognitions about the sequences and/or the donor individual are collected from different sources. After establishing the collection the '*Analyzing phase*' follows and leads to the '*Interpretation phase*'. The model assumes that the whole analysis is an iterative process. This means that the original collection is adapted to intermediated cognitions and reanalyzed. The phylogenetic tree visualizes relations between sequences and the mapping of additional properties to the corresponding nodes can uncover new correlations.

format. Sequences from laboratories came in common sequence formats or in own formats like spread sheets. Then the sequences will be arranged and typically managed by a set of files. Additional information and search results are commonly handled with normal office programs. A resulting collection is typically analyzed via phylogenetic trees and further analysis. The outcomes normally require the adaptation or creation of further collections until the final problem is resolved. TreeDB now provides the collection of sequences and corresponding properties, which can be used to create sequence sets (Fig. 4.1). These sets can be exported in common sequence formats and analyzed. Resulting alignments and trees can be added to sets and build a unit. Further results could not be managed within sets but they can be added as additional sequence properties. These properties can be used to relabel tree nodes while exploring trees. If sequences of a set are altered while updating sequences or adding sequences to a set a new set version is created. Set versions represent different states in the whole analyzing process, are disjoint and comparable in TreeDB.

4.2 Understanding the concept of categories, properties and relations

As introduced above one challenge of TreeDB is the management of user-defined sequence or individual properties. TreeDB solves this problem by the grouping related properties into user defined categories and allowing complex data structures depending on user defined relations between properties.

Categories

TreeDB gives the opportunity to manage properties in groups called categories. The intent or meaning of a category depends on the problem or on the user preferences to structure needed properties. Later on, each category represents its own search interface in TreeDB but this is not the only reason

to structure properties by categories. Well-structured categories are easily re-usable between projects. To make the concept of categories familiar Table 4.1 presents the categories defined to map HvrBase++ data into TreeDB. In this case five general categories *Geographic*, *Language*, *Country2Language*, *Haplogroups* and *Literature* are defined and used in the specialized category *HvrBase++*. Other projects can use this general categories without the need of redefining. To summarize categories are collections of properties, the amount of categories is not limited, categories can integrate other categories but at least one category is needed to manage properties.

Properties

A property consists of a keyword and a value. The following table illustrates this concept:

Keyword	**possible Values**
Continent	Asia, Europa
Country	France, Poland
City	Cologne, Gelsenkirchen
Population Size	166312, 20556, large, small

The specification of a keyword is important to understand the meaning of a property. Hence, it is possible to define a further description for every property of a category. But the user must take care to define meaningful keyword specifications. Additionally it is possible to use strings and numbers to define property entries. Each key/value pair build a unit by which TreeDB separates properties. These units are also called entries or records. As an example TreeDB can distinguish between 'Poland' as a country or as a computer program, if two entries *country/Poland* and *program/Poland* exists.

Property entries are not directly mapped to sequences or individuals when they are defined by categories. So an additional step is required after inserting categories and sequences. However predefined property entries help to

Category	**Meaning**
Geographic	Manages geographic locations like continents, countries and the corresponding ISO-IDs. To represent the dependency between single properties mostly the *partOf* relation is used (e.g. Germany partOf Europa).
Language	Holds over 7,000 languages and the corresponding scope, type and four different kind of ISO-IDs for each language. *Type* describes the status of language for example living or distinct. *Scope* denotes different kinds of languages like 'macro language', which is a collection of related languages. The basic scope is the language itself.
Country2Language	An abstract category, which joins category *Language* and *Geographic* by mapping countries to languages. With this category it is possible to answer questions like 'Which is the capital where a certain language is the mother tongue?' This category depends on *Language* and *Geographic* category.
Literature	Holds properties needed to manage literature information, like author names and publication title. This category is needed for the publication search interface.
Haplogroups	List of 51 common haplogroups and related mutations which determine complete mitochondrial genomes.
HvrBase++	This category consists of seven properties to map HvrBase++ data into TreeDB. It integrates the categories *Geographic*, *Language*, *Literature* and *Haplogroups* by the properties *continent*, *country*, *language*, *ref_string* and *haplogroup*. Also the independent properties *species* and *population* are defined.

Table 4.1: This table shows and explains categories defined to reproduce the HvrBase++ properties in TreeDB. See also Figure 4.3 which shows all properties and relations for these categories.

establish reliable data sets by admitting only the use of predefined entries. Another point for using predefined property entries is the feasibility to build complex data structures (see following paragraph). Otherwise it is also possible to specify only allowed keywords for a category and create the property entries on the fly while mapping entries to sequences. This is needed when data comes from a series of measurements where the results are not predictable.

Relations

Terms can be used to specify relations between two property entries to image complex structures or dependencies into TreeDB. This term is called *predicate* and maps an object to a subject. The construct consisting of a subject, a predicate and an object is called a *triple*. All properties can be used as subject or object. In *Geographic category* all instances of country are assigned to the corresponding continent. In this case the subject is country and the object is continent and the predicate is 'partOf'. There is no restriction on how to build a triple. It is also possible to define continent as subject and country as object and link them to together with the predicate 'consistsOf'. The way how to build a triple depends on the problem or the logic construct mapped into TreeDB. Four common predicates are predefined and can be used to structure categories:

partOf: Subject is a part of an object. (Germany is part of Europe)

isA: Subject is specified by an object. (Berlin is a capital)

has: Subject has a property. (Country has population size)

isEqual: Subject is the same as object. (Country ID is equal to country)

How to structure data is depicted for the geographic category (Fig. 4.3). The different kinds of regions stand in a hierarchical order and the partOf predicate is used to map their relation into the schema. '*Capital*' and '*Region*' are the smallest units located in a country. A '*Country*' is part of a continent

Software	Explanation	
SQLite v3.0	Is a small C library that implements a Self-contained embeddable SQL database engine.	
Web Server	Abyss Web Server X1 (V2.4). A Tiny web server configurable via a web front end.	
Convert v6.3	Converts between graphic formats. Contained in the ImageMagic suite, that create, edit, and compose bitmap images	
Perl v5.8	*P*ractical *E*xtraction and *R*eport *L*anguage is script language.	
Perl modules	Bio::Phylo v0.16	Phylogenetic analysis using Perl. Used to convert Newick trees into SVG graphics.
	SVG v2.32	Extension for generating SVG documents.
	Bio::Perl v1.4	Bioperl is the product of a community effort to produce Perl code which is useful in biology. Used to handle sequence formats.
	CGI v2.18	Common Gateway Interface. Help to manage web sites
	DBD::SQLite v1.14	Self Contained RDBMS in a DBI Driver
	DBI v1.59	Database independent interface for Perl

Table 4.2: Required programs and libraries to install TreeDB

and a continent is a part of a '*Planet*'. The most general property '*Planet*' is labeled with the attribute '*Root*', which describes '*Planet*' as the topmost property. ISO IDs are specified with the attribute ID. They are equal to countries, so that the 'is_equal' predicate is used. At least '*Country*' has the property '*Population size*'. By walking through the graph it is possible to get different information about a property. For instance it is possible to follow that Paris is a part of Europe, which is indirectly specified in the established graph.

4.3 Implemenation

4.3.1 Software requirements

As discussed earlier TreeDB should be a web application with less configuration overhead, which works on the following three operating systems: Mac OS X, Linux and Windows. Thus, the required software components are pick

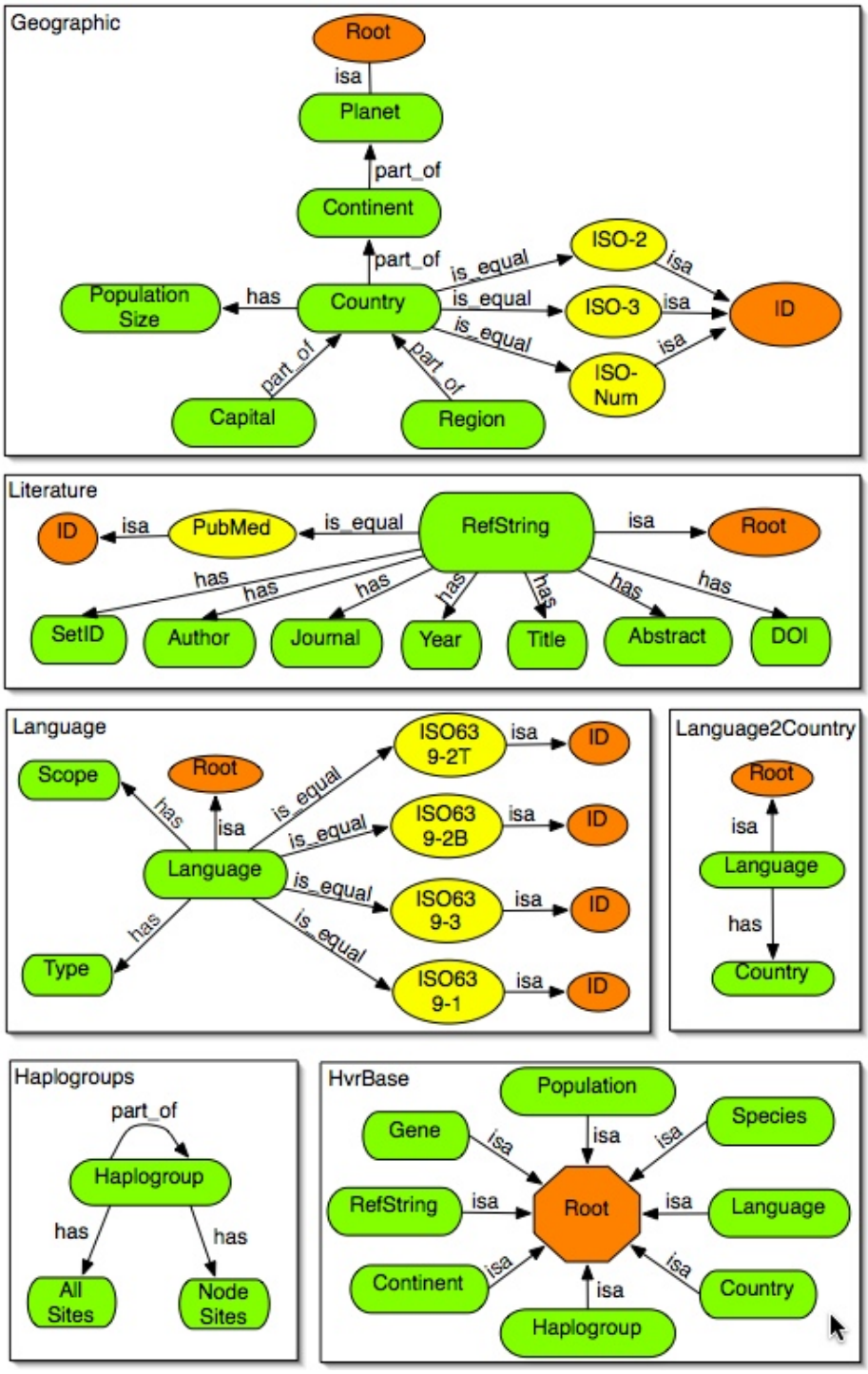


Figure 4.3: Necessary categories to manage HvrBase++ data in TreeDB are shown in this figure. Each of the six boxes represents a category. The colored boxes inside the rectangles stand for defined property entries. Green property entries are the core properties, while yellow ones represent existing IDs for a core property. Orange properties are objects that determinate attributes of properties. Arrows between boxes mark relations between properties. The direction of an arrow points out the way how a triple is defined, from *subject* to *object*. Table 4.1 gives an overview about.

out to fulfill these conditions. Furthermore TreeDB should be easily adaptable by advanced users to their own needs. Hence the scripting language Perl was used. A huge collection of bioinformatics methods and tools are available for Perl by the Bioperl project.

SQLite is chosen as DBMS because its small size (250 kb) with no configuration overhead. The database is managed in a single file, which can be exchanged between systems by simply copying the file. Another advantage of SQLite is, that only basic features of SQL92 are implemented, hence a replacement to another database is not problematic. Abyss is chosen as web server, because it implements the required features, is tiny and can be configured in a comfortable way via a website.

4.3.2 Implementation of TreeEditor window

TreeEditor is a good example to demonstrate how different application layers work together (Fig 4.4). A selected phylogenetic tree is read out from database in Newick format and converted into an SVG image via the Perl modules *Biophylo* and *SVG*. This tree object is then embedded into the XHTML code of the website and sent to the client. JavaScript allow the modification and relabeling of tree nodes. The relabeling step uses XMLHttpRequest to get the required node properties. For this reason the origin node label must code the sequence id (S-XXX), the individual name or id (N-XXX). If the individual name or id is used and more than one sequence is defined for an individual, multiple entries for sequences properties occur. The use of '*sequence_ids*' as node labels is recommended in such cases. The final tree is sent back to the web server and transformed via the command line tool 'convert' into other graphic formats.

4.3.3 Database Schema

The two important parts of sequences management and properties management are point out in this section. The pre-requested underlying database schema is shown in figure 4.5.

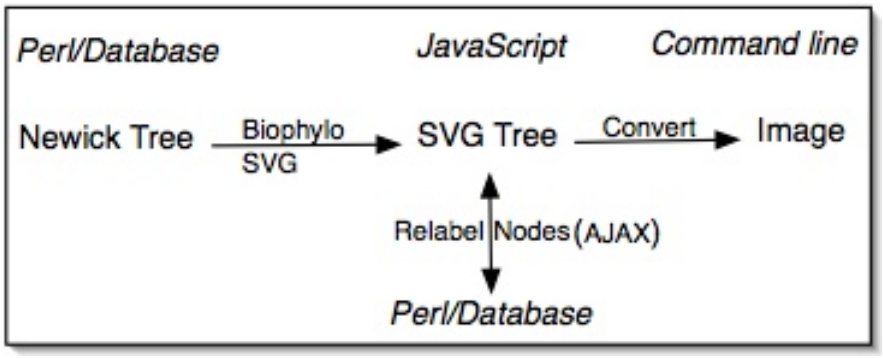


Figure 4.4: Workflow and interaction of the different TreeEditor components.

Sequence management

TreeDB distinguishes between donor individuals and sequences. That make it possible to map sequences from different genetic loci to an individual. Hence, the two tables SEQS and INDIVIDUALS exist. Information about genetic loci are defined in the table GENES and are used as foreign keys in table SEQS. In addition table SEQS stores alternative sequences IDs, a version number and the import date. If a sequence is updated a new record is inserted with an incremented version number. So that '*individual_id*', '*gene_id*' and '*version*' uniquely identify a sequence. The column '*current*' is used as a flag to optimize the search process for a current sequence version.

Sequences of a sequence set are managed via a join table between tables SEQS and SEQSETS. If a sequence update is required also for sequence sets, a new set version is created and an updated collection of '*sequence_ids*' is mapped against the new set version. Hence the schema allows the trace back for sequences and for sequence sets. While the '*seqset_id*' is used to link alignments and Newick trees to sequence sets, it is also possible to trace back these.

Property management

An interesting part of TreeDB is the question how to handle user-defined properties and relations. In general a database schema is optimized for a

well-known problem, which does rarely change over time. The main problem is not to alter the database schema later on but rather the interconnection between the database and the program logic. Thus, changes in database schema lead to changes in the source code of the application. One way to solve the problem is a program that react dynamically on database changes and present general views, like Ruby on Rails does (Thomas and Heinemeier Hansson, 2006). TreeDB instead handles dynamic data in a general way, which does not care on the underlying data structure. Hence the schema does not change but the structure or the logic of the data managed by the application.

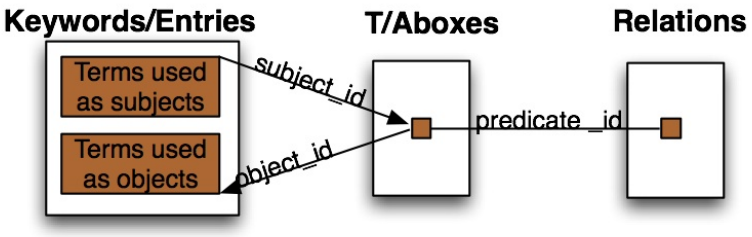
Properties are not handled in one table, where the column names define property and rows represent characteristics or entries. However TreeDB provides two tables Keywords and Entries to define properties and characteristics of properties. The characteristic or the instance of a property is then mapped to the individual. For example, the keyword or concept '*country*' is defined in table Keywords and instances of this concept, like Germany, are defined in table Entries. Individuals from Germany can be linked to this instance or entry via the join table Entries_Individuals. The following SQL statement points out how to use the schema to find all individuals from Germany:

```
SELECT i.name
  FROM individuals i,entries_individuals ei,entries e,keywords k
 WHERE k.name  = 'Country'
       e.entry = 'Germany'
       i.id = ei.individual_id
       e.id = ei.entry_id
       k.id = e.keyword_id;
```

Indeed a join over four tables is necessary but each instance is well-defined and stored once. Moreover this statement is used to find all kinds of properties only by changing the keyword and its corresponding value. Before doing on how relations are handled keep in mind that properties managed by categories via the three tables Categories, Categories_Keywords and CategoryDependencies.

Relations management

TreeDB distinguishes relations between concepts (terminological box or Tbox) and instances (assertional box or Abox), but they are implemented in the same way. As explained triples are used to determine relations. On the concept level keywords are used as subjects or as objects and on the instance level entries are used instead. Predicates are administrated in a separate table RELATIONS for both cases. The triple itself is then defined by three foreign keys and additional information about the triple in table TBOXES or table ABOXES.



The following SQL statements demonstrate the functionality of the implementation. The first statement simply finds the continent to which the country England belongs to:

```
SELECT o.entry
  FROM entries s, entries o, aboxes a, relations r
 WHERE s.entry    = 'England'
   AND r.relation = 'partOf'
   AND s.id = a.sentry_id
   AND o.id = a.oentry_id
   AND r.id  = a.relation_id;
```

Table ENTRIES is used for the subject (s) and the object (o) definition. Hence the denotation of a column depends on the join condition. If the primary key ('*id*') of table ENTRIES is assigned with column '*sentry_id*' of table ABOXES then columns are treated as subject entries. The related object (Europe) is then found by restricting the search by the subject 'England' and the pred-

icate 'partOf'. This example predicts that 'partOf' is defined once for the subject and used to specify the continent. Otherwise the search must also be restricted by the entry specifying keywords ('*country*', '*continent*'). To simplify this case the column '*tbox_id*' can be used to restrict the search with the knowledge of the concept level:

```
SELECT o.entry
  FROM entries s, entries o, aboxes a, tboxes t
 WHERE s.entry = 'England'
   AND t.name = 'Country2Continent'
   AND t.id = a.tbox_id
   AND s.id = a.sentry_id
   AND o.id = a.oentry_id;
```

This reflects the common search process of exploring the concept level and then pick out instances of a concept. The last example shows that also transitions are possible with the database schema. Predicating the *geographic* concept of HvrBase (Fig. 4.3) where capitals belong to countries and countries to continents, it is possible to find out in one SQL statement to which continent a capital (e.g. Berlin) belongs to:

```
SELECT o.entry
  FROM entries s, entries o, aboxes a1, aboxes a2, relations r
 WHERE s.entry = 'Berlin'
   AND r.relation = 'partOF'
   AND s.id = a1.sentry_id
   AND a1.oentry_id = a2.sentry_id
   AND o.id = a2.oentry_id;
   AND a1.relation_id = r.id
   AND a2.relation_id = r.id;
```

In this example two ABOXES (triples) are joined. The object of the first triple (Germany) restricts the subject of the second triple and the object of the second triple (Europe) is selected.

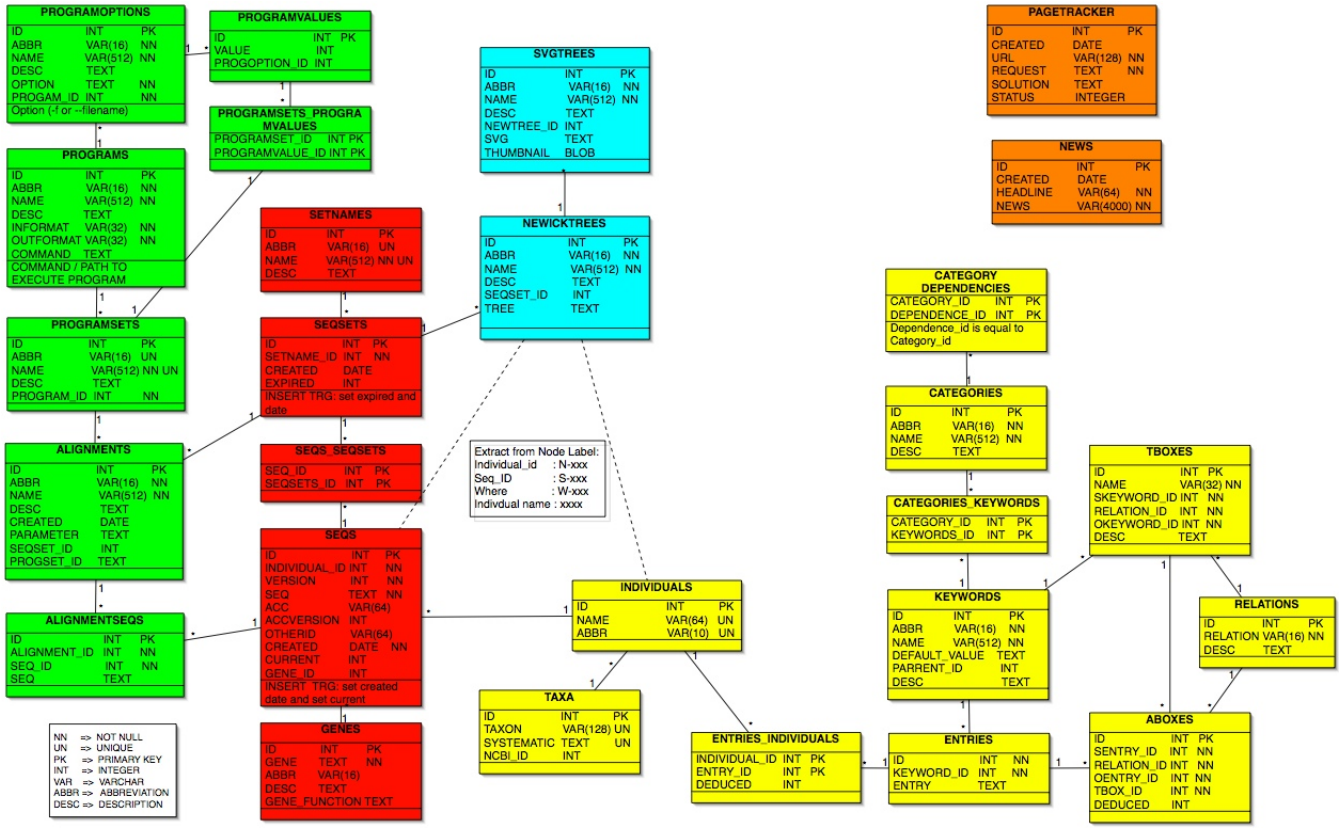


Figure 4.5: The database schema of TreeDB can split into five topics: Sequence and Set management (red), Category, Property and Relation definition (yellow), Alignment (green) and Tree (blue) storage, communication tools (orange). Each box represents a table and lines show the relations between tables. The first entry of a box is the table name. Other lines represent the table definition. The two dashed lines mark possible relations between node labels of a Newick tree and entries of table SEQS and INDIVIDUAL. The 'ActiveRecord' (Object-relation mapping library written in Ruby) way of defining tables and column names is used, so that the schema can easily be imported into a RUBY ON RAILS Project: Table names are defined in plural, each table has the primary key column id, foreign keys consist of the singular of the foreign table and the suffix '*_id*'.

4.4 Working with an existing collection

The achievement of structuring the data is the transparent view on the data. TreeDB provides four different projections to explore collections:

- Focus on general sequence properties.
- Focus on publications.
- Focus on sequence sets.
- Tree visualization and mapping of properties.

The first two projections work on the sequence level. Categories, properties and relations are used to retrieve sequences. The remaining projects work on sequence set level providing features to find, visualize and modify sequence sets. Both levels are discussed in the following two paragraphs. Examples given are related to the HvrBase++ categories and a corresponding data set consisting of 50 mitochondrial genomes offered by (Ingman *et al.*, 2000).

Sequences retrieval by categories, properties and relations

The goal of the *SequenceSearch* projection is the general sequences management via sequence/donor properties (Fig. 4.6). The main challenge for this projection is that it should work with user-defined data structures. Hence a general search form is built to find sequences by properties and relations with the lack of optimized search options. To attenuate the problem TreeDB provide three separate search modes or sites, which are alternative ways of presenting properties. Which mode is the best depends on the definition of the category and on what users are interested in. For example, the default search mode shows only the properties directly mapped to donor individuals and their corresponding '*is_equal*' relations. This means for the language category that only the languages are shown which are spoken by donors and not the whole 7,000 languages defined in this category.

The publication management of TreeDB is an example for a specialized projection for the literature category (Fig. 4.7). The form, the program

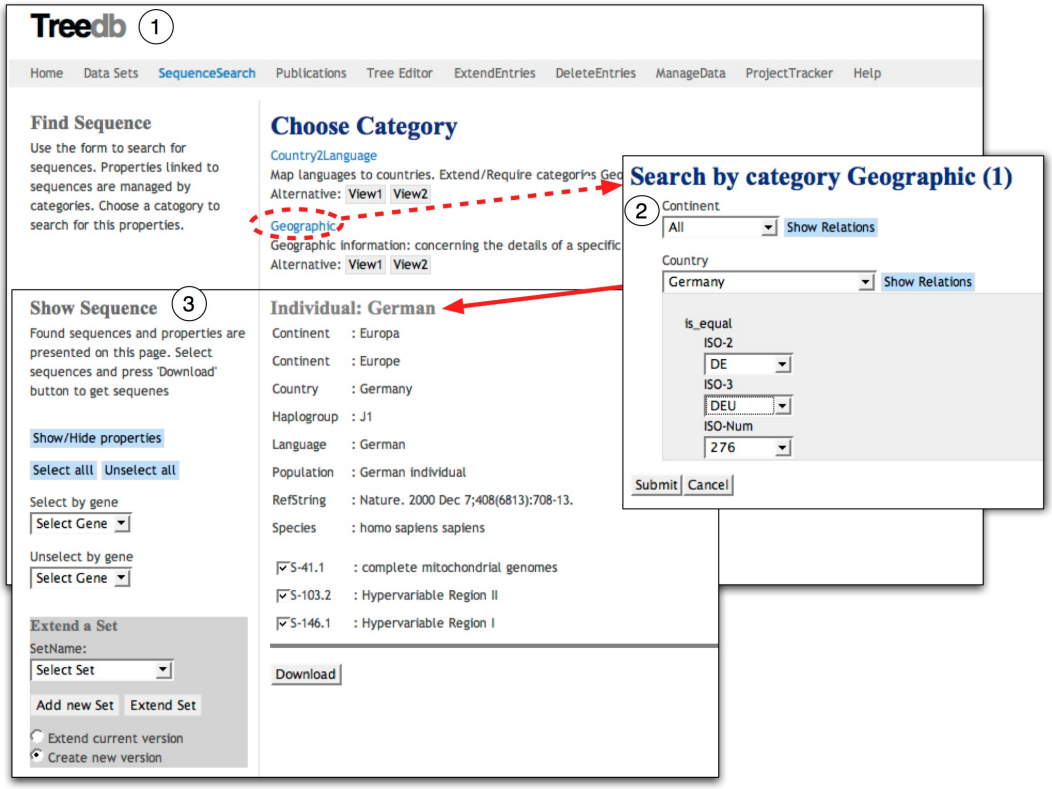


Figure 4.6: This figure gives an example how the generalized *SequenceSearch* projection looks like. Sites are dynamically created depending on defined categories. In this case the HvrBase categories are used by the system (Table 4.1). The *SequenceSearch* projection consists of three websites (circled numbers). 1) First one of the tree alternative property search forms for one of the categories must be selected to search for properties and relations managed in the chosen category. 2) In this example the default search form for the *Geographic* category is selected (red ellipsis). The form shows the two used properties '*continent*' and '*country*' and '*is_equal*' relations of property '*country*' (grey box). The terms defined by '*is_equal*' relations can be used as an alternative way of specifying a property. Hence it is possible to select the property 'German' by the ISO number 276. 3) The website shows the search result and gives the opportunity to download single sequences (right frame) or to add sequences to a sequence set (grey box, bottom left). For the search term 'Germany' tree sequences from one individual are found. All properties directly mapped to the individual are shown. Properties can be hidden to get an overview of all hits (Blue box *Show/Hide properties*). Additional properties can be added by using the *ExtendEntries* projection (not shown see 4.8).

logic and the representation are designed to work with publication data and require the HvrBase++ literature category.

These two examples show that the data structure of TreeDB can be used by a common search form but it is also possible to design optimized forms to solve specific requests. So it is possible to extend the functionality of TreeDB by adding further features without changing the data structure or the rest of the application. This gives bioinformaticians a chance to quickly adapt TreeDB to a specific problem without much effort.

Another interesting aspect is the feasibility to extend mapped properties by relations. The *ExtendEntries* projection presents all relations for mapped properties (Fig. 4.8). The user can choose some of these relations to map corresponding properties to the individuals. Moreover, new properties can be used to add further properties. Hence it is possible to walk through category structure and add needed properties to individuals. This gives the opportunity to enrich own collections by importing foreign categories and to add new properties via corresponding relations.

Figure 4.7: Publication search form of TreeDB. Publications can be found by terms. A term is a word or a collection of words surrounded by commas. In this example one publication for the search term '*ingman*' is shown. *DOI* property is used to link to the original publisher. *PubMedID* is used to link to the specific PubMed service. Related sequences can be uploaded from the NCBI databases. Moreover a PDF can be added and offered for download.

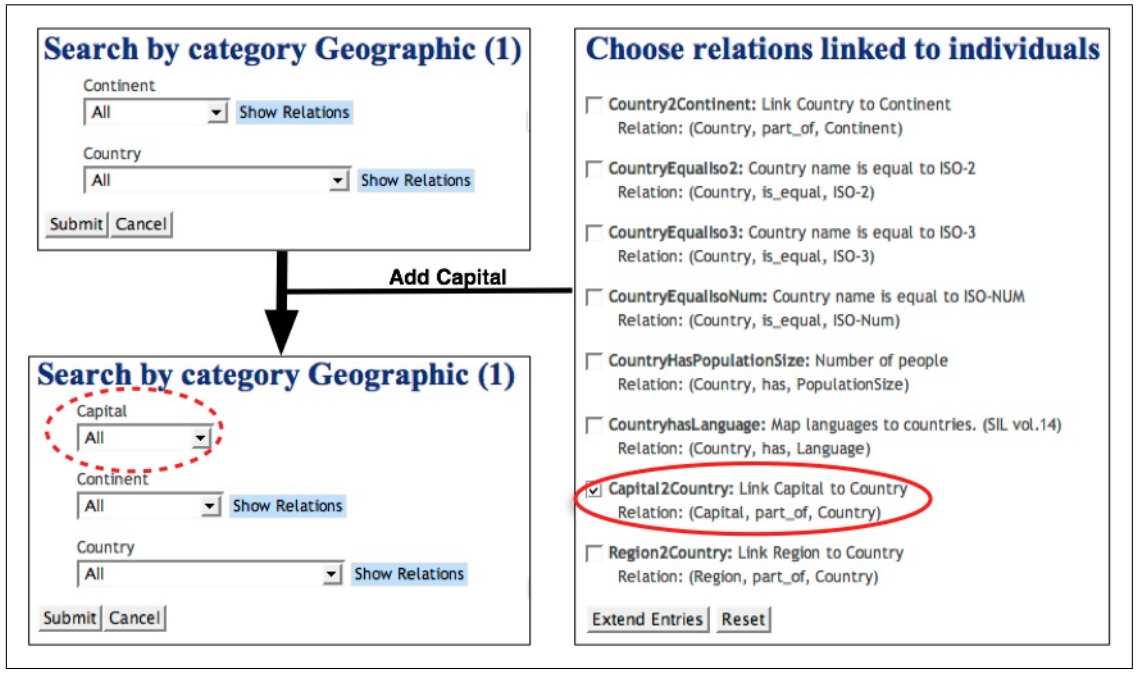


Figure 4.8: The process of mapping properties via relations is shown for the *Geographic* category. On the left side the geographic search form before and after adding '*capital*' properties is shown. On the right side the form for adding properties via '*country*' relations is shown. In this example the property '*country*' is already mapped to individuals, so all properties related to '*country*' can be mapped to sequences.

Dataset and TreeEditor

The *DataSet* and *TreeEditor* projection focus on the management and visualization of data sets. The goal is the assistance of the analyzing process. Each sequence set builds the base for a data set, which can be extended by alignments and phylogenetic trees. Each data set is presented on a single website, that gives an overview about the set and makes the different kinds of data accessible (Fig. 4.9). Furthermore a history is implemented that allows the track back of older versions. For a new version alignments and phylogenetic trees must be created and inserted again. Old data will not be removed and can be revisited and compared. This keeps the process of analyzing transparent. *TreeEditor* visualizes phylogenetic trees and labeled properties to corresponding nodes (Fig. 4.9).

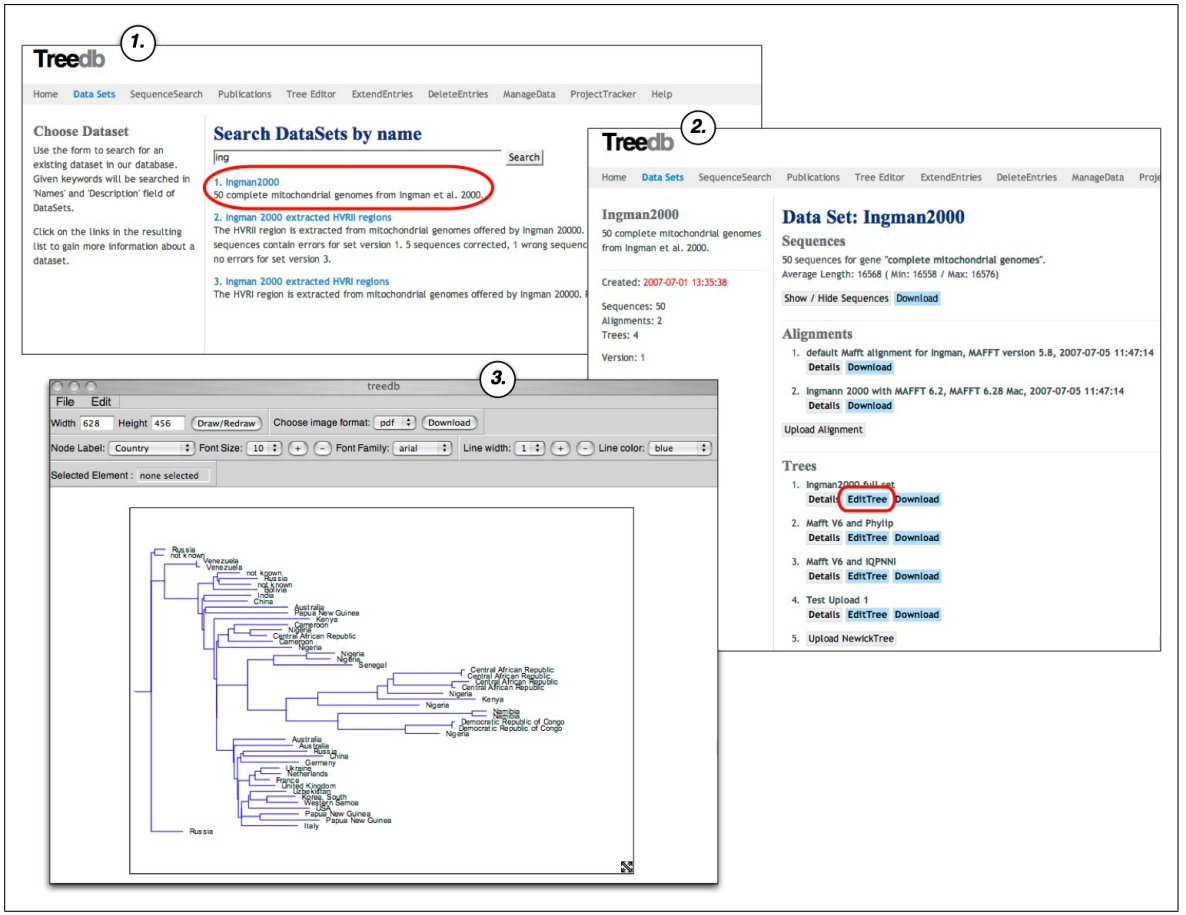


Figure 4.9: This figure shows the organization of data sets in TreeDB. 1.) This window visualizes the *DataSets* search form. Data sets can be found by name or descriptions. Three data sets are found for the search term *ing*. The red marked data set, *Ingman2000*, is selected. 2.) This window presents one data set. The left site summarizes the data set and gives the opportunity to switch between set versions. In this example the *drop down box* for switching between versions is not presented because only one data set is available. On the right side sequences, alignments and trees are organized. Further alignments and trees can be uploaded into this data set version. 3.) Newick trees can be visualized via the *TreeEditor*. The representation of a tree can be modified by altering fonts and size of node labels or colorized edges. The nodes can be relabeled by defined properties (here country). Modified trees can be downloaded as images (PDF, JPEG, SVG, PNG). The size of an image is changeable by resizing the tree.

4.4.1 Establishing a collection

The section before gave an example how TreeDB works with an existing collection, now the process of initializing a collection is explained. The aim is not to give an explicit, stepwise description of the import process, however this section focus on the interesting point of establishing own categories and mapping properties to individuals or sequences. Hence the following list gives a short overview of the single steps needed to create a database and then explains in more detail how to create categories:

- **Install TreeDB** – Install the required software and libraries (Tab. 4.2), unpack TreeDB zip file into the Abyss (web server) folder and start the web server. TreeDB is then available under 'http://localhost:8000'. The zip file contains Perl libraries, HTML-sites and the Abyss configuration file *abyss.conf*. Other web servers must be configured, so that pages with the suffix '*.html*' will be interpreted by Perl.

- **Sequence Insertion** – Sequences can be uploaded via a form in a multitude of formats, like Fasta or GenBank. The upload procedure requires that every sequence is related to a set. It gives the opportunity to build needed sets directly or mark the origin of sequences by using different data sets. Perhaps it is possible to upload all sequences in one set. Furthermore a sequence type like HVR-I region or *ADH* gene and a taxonomic description can be assigned to all sequences in a file.

- **Creating and uploading categories** – A Category with all properties, relations and property entries can be complex, so that a web interface is not a good choice to build categories. Hence categories were defined by creating a tab delimed file outside TreeDB. These files can be created via common spreadsheet programs but also by shell scripts. Furthermore these files can easily be exchanged and reused.

- **Mapping Properties** – The mapping process requires the explicit mapping of every property to every individual. Doing it by hand for

Name	Population	Species	Country	Haplogrp	Continent	RefString
Hausa	Individual Nigeria	Homo sapiens	Nigeria	L1a	Africa	Nature.2000
San01	Namibian individual	Homo sapiens	Namibia	L1a	Africa	Nature.2000
San02	Namibian individual	Homo sapiens	Namibia	L1a	Africa	Nature.2000
Mbuti01	Individual Congo	Homo sapiens	Congo	L1a	Africa	Nature.2000
Mbuti02	Individual Congo	Homo sapiens	Congo	L1a	Africa	Nature.2000
Ibo01	Individual Nigeria	Homo sapiens	Nigeria	L1b	Africa	Nature.2000
Ibo02	Individual Nigeria	Homo sapiens	Nigeria	L1c	Africa	Nature.2000
Effik01	Effik	Homo sapiens	Nigeria	L2a	Africa	Nature.2000
Effik02	Effik	Homo sapiens	Nigeria	L2a	Africa	Nature.2000

Figure 4.10: A tab delimited file, which maps six properties to nine individuals. Shortened and modified example of the *Ingman* data set. The first line specifies keywords for properties mapped in each column. The first column determines individuals by names or individual ids. Allowed keywords for this column are *Name* and *Id*. In all other columns all defined keywords can be used to specify properties. The order or the amount of mapped properties is not restricted. Later on further properties can be added to individuals via the *ExtendEntries* form, which extends mapped properties via relations (Fig. 4.8).

date sets is a time consuming step. For that reason also a tab delimed file, that describes the mapping, is created and uploaded into TreeDB. Figure 4.10 shows an example and explains the format.

- **Modifying or Creating Sets** – After that TreeDB can be used to add sequences to a set, update sequences or properties or create new sets.

Establishing own categories

As explained above categories are defined in tab delimited files. Now the concept of defining a category is illustrated. One important thing is to distinguish between the keyword and the entry of a property. The keyword can be understand as a concept and the entry as the instance of a concept. For example '*country*' is the concept and 'Netherlands' is the instance. Before an instance can be defined the concept must exist. The same is true for relations. A relation first must be defined on the concept level before defining manifestations of these relations. Thus, at first keywords and relations

between keywords are defined and then entries and relations between entries are defined.

The file format is illustrated by Figure 4.11. This example defines four countries and corresponding 'two letter ISO codes'. First of all the category itself is described by a name, an abbreviation of the name and a description. The name of a category must be unique for the established database. Categories can integrate properties from other categories, which are determined in the '*Dependency section*'. But at least each category depends on itself. The two required relations are defined in the '*Relations section*'. In the '*Keywords section*' keywords and corresponding relations are defined. In this example the '*isa*' relation maps the attribute *ID* to the subject *ISO-2*. Further the alternative *ISO-2* term is map to '*country*' via the '*is_equal*' predicate. The relation name of this concept is '*CouEqIso*', which is used to define instances of this relation within the '*Entries section*'. An entry itself is simply defined by a value and the corresponding keyword.

```
[CATEGORY]
#name        abbr   description
Geographic   geo    Defines four countries with ISO codes
[DEPENDENCY]
#requiredCategory
Geographic
[RELATIONS]
#relation   description
isa         Map attributes to subjects.
is_equal    Map alternative terms to subjects.
[KEYWORDS]
#name   abbr    description      defaultValue  relName    relation   objectKeyword   description
Country Country Country concept
ID      ID      Mark ISO-ID as ID
ISO-2   ISO-2   Two letter code               Iso2IsId    isa        ID              Iso2 is a id
Country                                      CouEqIso    is_equal   ISO-2           Country  is equal to ISO-2
[ENTRIES]
#entry     keyword   relName    objectEntry
AD         ISO-2
AO         ISO-2
AI         ISO-2
DE         ISO-2
Andorra    Country    CouEqIso   AD
Angola     Country    CouEqIso   AO
Anguilla   Country    CouEqIso   AI
Germany    Country    ConEqIso   DE
```

Figure 4.11: This example shows a category definition of four countries and corresponding two letter ISO-codes in a tab delimited file. The file can split into the following five sections: Category, Dependency, Relations, Keywords, Entries. Each line of a section specifies a record. Values of a record are separated by tabulator signs (not explicitly shown). Required values depend on the section. These values are described in the first line of a section, which starts with the comment sign (#).

Chapter 5

Conclusion

The way of collecting biological data sets and the management of such data are analyzed for the mitochondrial database HvrBase to get familiar with the problematic and deduce general requirements for biological data management. The analyzing process leads to the re-creation of HvrBase, called HvrBase++, and a data management application for own sequence collections, called TreeDB.

HvrBase++ is now a more general phylogenetic database to explore human history not only concentrated on the two HVR regions. For that reason the database schema and the web application are redesigned to manage also other genetic loci. In the current version of HvrBase++ six Y-chromosomal loci and four autosomal loci and complete mitochondrial genome sequences are added. Overall HvrBase++ consists of sequences from 21,638 donor individuals (Fig. 3.12), whereas the major part of sequences is still related to mitochondrial sequences (21,229 sequences). At the moment an equivalent to the widely used mitochondrial HVR regions in nuclear genome loci is not present and only small sequence sets are available. The nuclear genomic variation is mainly studied by single nucleotide polymorphisms (SNPs) or by short tandem repeats (STR) (Agrafioti and Stumpf, 2007). But outgoing from the development of mitochondrial study, where a shift happened from restriction fragment length polymorphism (RFLP) to sequences studies, an

increase of available sequences can be assumed. Furthermore, the geographic coverage for 2,416 complete mitochondrial genomes shows a predomination of sequences from Japan (679), Finland (181) and India (163) and only for 63 countries sequences are available. Hence, the extension in HvrBase++ was necessary to react on potential new loci but continued updates are needed, to hold the relevance of it. The searching and collecting process is still time consuming although a new semi-automatic process is implemented. Handling the sequences itself is not problem. Most sequences are available from general public databases, like GenBank, which is accessible by the *Entrez Programming Utilities*[1]. The critical parts of the collection process are finding potential new sequences and the extraction of additional sequence information. This is caused by the problem of extracting information automated from text written in natural language. Well-defined bibliographic information help to preselect publications but information extraction in our context is still a hand-crafted job. A solution could be text mining tools like *GATE*[2] (General Architecture for Text Engineering) or methods provided from computational linguistics. Another interesting solution, known from *Web 2.0*[3] is to invite scientists to help to extend the collection. Scientists can (1) propose interesting new sequence sets or (2) can directly insert sequences. The benefit for scientists is to represent their own works and results.

The re-creation of the web interface replaces the simple stateless search form of HvrBase by a caching mechanism to manage and build private collections from many independent queries. These collections can be inspected and modified by visualizing the relationship between sequences with phylogenetic trees. Furthermore, a world map shows the distribution of additional sequences properties and helps to understand geographic relations. These functions give a first expression of the data while selecting them, which makes the collection more transparent and speed up scientific collection processes.

[1] Tools that provide access to Entrez data outside of the regular web query interface. http://eutils.ncbi.nlm.nih.gov/entrez/query/static/eutils_help.html

[2] http://gate.ac.uk

[3] Web 2.0 refers to a perceived second generation of web-based communities and hosted services – such as social-networking sites, wikis and folksonomies

The following list summarizes the development of HvrBase++:

- HvrBase was verified, re-designed and extended.
- A new database schema was created to provide sophisticated queries.
- An improved collecting and updating mechanism based on the NCBI-Tools is implemented.
- The new web-interface offers now five alternative search modes to build private and shareable collections.
- For data visualization (1) a global information system providing dynamic world maps and (2) an interactive phylogenetic tree editing tool is integrated into HvrBase++.

TreeDB is a small and easy to handle application for sequence data managed in life sciences, which assists scientists to manage different kind of data sets consisting of raw sequences, alignments and phylogenetic trees. Furthermore, a large number of user-defined properties and relations can be mapped to sequences or individuals. The relations are managed by triples consisting of a subject, a predicate and an object, which allow the modeling of complex data structures. As special feature these properties can be visualized in phylogenetic trees, which helps to quickly analyze them or find errors in the data set. The functionality of TreeDB is tested with a data set of 50 complete mitochondrial genomes and the corresponding properties taken from HvrBase++. For this reason five categories were defined to manage HvrBase++ properties. Moreover, an adapted search form for the HvrBase++ literature category was written to show the potential of TreeDB's properties management and the possibility to extend TreeDB with self-written forms. To show the flexibility of TreeDB a data set of 27 sequences from 3 genes and 14 corresponding morphological characteristics has been integrated and tested. Further plans are the general improvement and extension of TreeDB such the

management of parameters used for analyzing steps for automated analyzing processes. This provides the opportunity of completely re-creating data sets after sequence updates. Secondly an import/export module for RDF triples allows data exchange with semantic tools, which can improve acceptance of TreeDB.

Chapter 6

Zusammenfassung

Die Mitochondrien Datenbank HvrBase wurde auf ihre Schwachstellen und auf mögliche Verbesserungen hin analysiert und entsprechend neu entwickelt. Die Erkenntnisse wurden genutzt, um eine allgemeine Sequenzdatenbank namens TreeDB zu entwickeln, die es Biologen ermöglicht eigene Sequenzdaten zu verwalten.

Die neue Version von HvrBase, HvrBase++, kann nun neben HVR-I and HVR-II Sequenzen auch DNA Sequenzen von weiteren genetischen Regionen verwalten. HvrBase++ ist somit in der Lage auf neue Trends bei der phylogenetischen Analyse der Entwicklungsgeschichte des Menschen zu reagieren. In der aktuellen Version von HvrBase++ wurden zusätzlich Sequenzen von sechs Y-chromosomalen Regionen, von vier autosomalen Regionen und von kompletten mitochondrialen Genomen aufgenommen. Die Datenbank umfasst nun insgesamt Sequenzen von 21.638 Individuen, wobei der Groß teil Mitochondrien Sequenzen sind (21.229), da momentan nur vereinzelt kerngenomische Sequenzen genutzt werden, um die Entwicklungsgeschichte des Menschen aufzuklären.

Außerdem wurde das Web-Interface komplett überarbeitet. Ein Caching-Mechanismus erlaubt es nun private Sequenzsets anzulegen und zu verwalten. Zur Erstellung eines Sequenzsets werden fünf neue Suchformulare angeboten. Wie die einzelnen Sequenzen eines Sets in Relation stehen, kann durch einen

automatisch generierten phylogenetischen Baum visualisiert werden.

Damit HvrBase++ nicht an Relevanz verliert, ist es notwendig die Datenbank immer auf dem aktuellsten Stand zu halten. Allerdings ist dies trotz des schnelleren halb-automatischen Erweiterungsprozesses immer noch zeitaufwendig. Das Problem liegt nicht bei der Handhabung der Sequenzedaten, sondern bei der Extraktion der Sequenzeigenschaften aus den Publikationen. Dieses Problem ließe sich mit Hilfe von Verfahren aus der Informationsextraktion lösen. Es wäre auch denkbar, HvrBase++ soweit öffentlich zugänglich zu machen, dass es anderen Wissenschaftlern ermöglicht eigene Sequenzen zur Datenbank beizusteuern.

TreeDB ist im Gegensatz eine kleine und gut bedienbare Web-Applikation, die es erlaubt, biologische Sequenzdaten effektiv zu verwalten. Nicht nur Sequenzdaten, sondern auch Alignments und phylogenetische Bäume werden verwaltet. Zu jeder Sequenz können beliebig viele benutzerdefinierte Eigenschaften verwaltet werden. Ein besonderes Merkmal von TreeDB ist die Visualisierung dieser Eigenschaften an den entsprechenden Blättern im phylogenetischen Baum. Die Applikation wurde für einen Datensatz von 50 kompletten mitochondrialen Sequenzen und den entsprechenden Sequenzeigenschaften aus HvrBase++ und einem Set von 27 Sequenzen für 3 Gene mit 14 morphologischen Eigenschaften getestet.

Appendix A

A.1 Used Programs and Libraries

- PubCrawler (Version 1.7) a Perl script for an "alerting" service which scans daily updates to the NCBI Medline (PubMed) and GenBank databases. http://www.gen.tcd.ie/pubcrawler/program.html
- MAFFT (Version 5.733) rapid program for multiple sequence alignments based on fast Fourier transform (Katoh *et al.* (2005)).
- Phylip – Phylogeny Inference Package (Version 3.61) is a package of programs for inferring phylogenies (evolutionary trees). http://evolution.genetics.washington.edu/phylip.html
- EMBOSS EMBASSY Packages (Version 4.1) extent emboss of external programs. Used for Phylip (Version 3.61). http://emboss.sourceforge.net/apps/release/4.1/embassy/index.html
- EMBOSS (Version 4.1) is a free Open Source software analysis package specially developed for the needs of the molecular biology user community. http://emboss.sourceforge.net
- Bio::Phylo (Version 0.16) is a Perl package to manage and draw phylogenetic trees. http://search.cpan.org/ rvosa/Bio-Phylo-0.16
- Bio::Perl (Version 1.4/1.5) is the product of a community effort to produce Perl code which is useful in biology. http://bioperl.org
- DBI (Version 1.59) Database independent interface for Perl.
- DBD::SQlite (Version 1.14) a self contained RDBMS in a DBI Drive.

- DBD::Oracle (Version 1.19) a Oracle database driver for the DBI module
- SVG (Version 2.32) an extension for generating Scalable Vector Graphics (SVG) documents.
- CGI (Version 2.18) a implementation the *common gateway interface* protocol for perl.
- SQLite (Version 3.0) Is a small C library that implements a self-contained, embeddable, zero-configuration SQL database engine. http://www.sqlite.org
- Abyss Web Server X1 (V2.4) s a compact web server available for Windows, MacOS X, Linux and FreeBSD operating systems. http://www.aprelium.com/abyssws
- ImageMagic (Version 6.3) a program suit to create, edit, and compose bitmap images. http://www.imagemagick.org
- Perl (Version 5.8) *P*ractical *E*xtraction and *R*eport *L*anguage is script language
- Apache (Version 1.3) an open-source HTTP server for modern operating systems including UNIX and Windows NT. http://httpd.apache.org
- Oracle 10g is relational database management system. http://www.oracle.com
- MapServer (Version 4) is an open source development environment for building spatially-enabled internet applications. http://mapserver.gis.umn.edu
- BLAST (Basic Local Alignment Search Tool) finds regions of local similarity between sequences. The program compares nucleotide or protein sequences to sequence databases and calculates the statistical significance of matches. http://www.ncbi.nlm.nih.gov/BLAST
- BLAT (BLAST-like alignment tool) http://genome.ucsc.edu

A.2 Materialized view HvrBase++

```
CREATE MATERIALIZED VIEW next_hvrbase.main_info
       USING INDEX TABLESPACE SMALL_IDX
       REFRESH FORCE ON DEMAND
       AS
       SELECT sa.sam_id, sa.name, sa.seq_id, sa.primery_id, sa.acc,
              sa.acc_version, sa.old_info_id, sa.old_seq_id, sa.malig_id,
              me.met_id, me.population, me.lang_pap,
              sp.spe_id, sp.specie,
              si.lang_pri_name, si.lang_id,
              iso.iso_name, iso.iso_id,
              ge.gen, ge.gen_full, ge.gen_id, ge.location,
              ge.start_gen, ge.stop_gen,
              con.continent, con.short_name, con.continent_id,
              cou.country_name, cou.country_id, cou.iso_3_id,
              cou.iso_num,cou.capital, cou.map_id , cou.pop_2005,
              re.orig_ref, re.ref_id, re.authors, re.ref_string,
              re.publish_date, re.page, re.ip, re.pubmed_id,re.title,
              jo.journal, jo.journal_id, jo.abbr, jo.essn, jo.issn, jo.nlm_id,
              pos.pos_id, pos.p_start, pos.p_stop
         from (((((((((
               next_hvrbase.sample sa
               INNER JOIN next_hvrbase.meta_data me ON sa.met_id = me.met_id)
               INNER JOIN next_hvrbase.gen ge ON sa.gen_id = ge.gen_id )
               INNER JOIN next_hvrbase.specie sp ON me.spe_id = sp.spe_id)
               INNER JOIN continent con ON me.con_id = con.continent_id)
               LEFT OUTER JOIN countrycodes cou ON me.cou_id = cou.country_id)
               LEFT OUTER JOIN languagecodes si ON me.lang_sil = si.lang_id)
               LEFT OUTER JOIN lang_iso_unique iso ON me.lang_iso = iso.iso_id)
               LEFT OUTER JOIN seq_pos pos ON sa.pos_id = pos.pos_id)
               LEFT OUTER JOIN (refference re INNER JOIN journal_list jo
               ON re.journal_id = jo.journal_id) ON me.ref_id = re.ref_id);
```

SQL-Statment to create the ‘materialized view‘ *Main_info*.

A.3 PL/SQL function searchView

```
 1 FUNCTION  "searchView" (country IN STRING,.,.,., gen IN STRING,.)
 2   return Types.cursorType
 3  AS query_cursor Types.cursorType;
 4  ....
 5
 6 BEGIN
 7 where_:=' ';
 8 tren:=' where ';
 9 if NOT country IS NULL THEN
10    where_:=where_ || tren || ' country_name in (';
11        temp:=zerlege(country_name,'|');
12           FOR i IN 1..temp.count
13          LOOP
14          where_:=where_ || '''' || temp(i) || '''';
15            if i<temp.count THEN
16           where_:=where_ || ', ';
17         END IF;
18      END LOOP;
19      where_:=where_ || ') ';
20      tren:=' and ';
21  END IF;
22  .....
23   statement := 'SELECT * FROM main_info ' || where ;
24
25 OPEN query_cursor FOR statement
26 return query_cursor;
27 END;
```

The function starts with the definition of six input and one out parameter (Lines 1-4). The *begin* command remark the beginning of the body (Line 6). First all input parameter processed to create the *where* clause. Only the processing of the parameter '*country*' is shown (Lines 9-21). If the parameter is defined then the function '*zerlege*' separate merged values from the parameter string and extent the *where* clause. After processing all parameter the *select*-string is build (Line 23). Then a cursor is created and returned (Lines 24-25).

A.4 Defined Haplogroups

ID	NAME	PARENT_ID	SUBSTITUTIONS/HALPOTYPES
2	L1	-	4312:C,2352:T,9072:A,12810:A,2758:A,10810:C,13803:A,4158:A,650:T,13958:G,3594:T 4104:G,11002:A,8618:T,2352:T,10400:C,14783:T,13263:A,4883:C,5178:C,7598:G,4833:A 9090:T,8701:G,9540:C,10873:C,663:A,1719:G,6221:T,14470:T,7933:A,8251:G,8994:G 1719:G,10238:T,12705:T,8272:T,11251:A,12612:A,3010:G,7476:C,15257:G,4917:A,10463:T 12633:C,11812:A,14233:A,2308:A,14766:T,4580:G,7028:T,3010:G,1438:G,4769:G,11467:A 12308:A,7805:G,14179:A,3197:T,14793:A,7768:A,1811:A,9545:A,9266:G,4646:T,8137:C,9055:G
3	L1a	2	4312:T
4	L1b	2	2352:C
5	L1c	2	9072:G,12810:G
6	L2-preL	2	2758:G,10810:T
7	L2a	6	13803:G
8	L2b	6	4158:G
9	L2c	6	650:C,13958:C
10	pre_L3_MN	6	3594:C,4104:A
11	L3c	10	11002:G
12	L3d	10	8618:C
13	L3e	10	2352:C
15	M	10	10400:T,14783:C
16	C	15	13263:G
17	D	15	4883:T,5178:A
18	E	15	7598:A
19	G	15	4833:G
20	Z	15	9090:C
21	N	10	8701:A,9540:T,10873T
22	A	21	663:G
23	X	21	1719:A,6221:C,14470:C
24	Y	21	7933G
25	pre_WI	21	8251:A
26	W	25	8994:A
27	I	25	1719:A,10238:C
28	R	21	12705:C
29	B	28	8272:A
30	pre_JT	28	11251:G
31	J	30	12612:G
32	J1	31	3010:A
33	J2	31	7476:T,15257:A
34	T	30	4917:G,10463:C
35	T1	34	12633:A
36	T2	34	11812:G,14233:G
37	T5	34	2308:G
38	HV	28	14766:C
39	V	38	4580:A
40	H	38	7028:C
41	H1	40	3010:A
42	H2	40	1438:A,4769:A
43	U	28	11467:G,12308:G
44	U6	43	7805:A,14179:C
45	U5	43	3197:C
46	U5a	45	14793:G
47	U5b	45	7768:G
48	pre_uk	43	1811:G
49	U2	48	9545:T
50	U3	48	9266:A
51	U4	48	4646:C
52	U7	48	8137:T
53	K	48	9055:A

This List show the defined Haplogroups for HvrBase++. Each haplogroup consist of the substations new defined for this haplogroup and the substitutions of the parent haplogroup (*parent_id*). Substitutions according to the Cambridge reference sequence.

Glossary

Abox Assertional box. Describes instance hierarchies (relations between individuals and concepts), 102

Anderson Notation Way of counting mitochondrial nucleotides based on the Cambridge reference sequence, starting in the D-Loop. Inserts are denoted by previous nucleotide position followed by a number characterizing inserted base separated by a dot (e.g. 127.2 denotes the second inserted nucleotide after CRS position 127), 33

Architectural Pattern High level description of software architectures, 33

ATP Adenosine triphosphate. Principal energy carrier of the cell, 5

Client-Server Architecture which separates a client from a server, and is almost always implemented over a computer network, 55

Control Region Noncoding part of the mitochondrial genome from 16024 to 576 bp containing both promoters, HVRI/II and one origin of replication, 6

CRS Cambridge Reference Sequence. First complete sequenced human mitochondrial genome by Anderson 1981 et al., 6

D-Loop A displaced DNA single strand that is created after strand invasion. The structure is formed when a third strand of DNA is taken up by double-stranded DNA, 6

Data-Warehouse A collection of integrated, subject-oriented databases designed to support data analysis, with the goal of aggregating data from different sources and separates data from normal on going business, 25

Database The collection of data that build the database, 21

DBMS A database management systems is software that manages the database, 21

DOI Digital Object Identifier. A permanent identifier given to a document, which is not related to its current location, 95

ESRI Shapefile Popular geospatial vector data format for geographic information systems software. Consists of three files with the extension '.shp', '.shx' and '.dbf', 62

Fasta A simple sequence format, 73

Functional dependency Constraint between two sets of attributes in a relation. Attribute B is functionally dependent on attribute A if, for every valid value of A, that value of A uniquely determines the value of B, 21

GenBank General DNA and Protein database provided by the National Center for Biotechnology Information (NCBI), 33, 73

GIS Geographic information system, 62

Haplogroups Set of variations promoting features such as geographical origin, 15

Heteroplasmic A mixture of genetically different cytoplasms, generally different mitochondria or different chloroplasts in a cell, 8

Homoplastic An organism with only one type of plastide; usually referring to genetic identity of mitochondria or chloroplasts, 8

HTML Hypertext Markup Language is a markup language for the creation of web pages, 33, 73

ISO International Organization for Standardization, 44

KYA Thousand years ago, 15

Multiregional Modern humans involve simultaneous around the Old World from archaic humans. Genetic background will exchanged between all populations. Also known as regional coalescence model, 15

MVC Model-view-controller is an architectural pattern distinguish ing between data (model), user interface (view) and processes responsible to interaction (controller), 33

MYA Million years ago, 15

Non-prime attribute An attribute that does not occur in any candidate key, 21

Normal Forms States of normalization to reduce anomalies by eliminating redundancy, 24

Oocytes The gamete in females. A germ cell that have a haploid chromosome complement, 8

Oogina Cells in females that produce primary oocytes by mitosis, 8

Out of Africa A model supposes that modern humans evolved within the African continent and migrated around the world and replace archaic humans. Also known as replacement model, 15

Phylip Is a free package of programs for inferring phylogenies and the name of the corresponding sequence file format, 73

Primary key A minimal key used as default key, 21

PubMed Publication database provided by National Center for Biotechnology Information (NCBI), 33, 34, 46, 50

RDF Resource Description Framework is a specification from the W3C for modeling data in the web. The model is based on subject-predicate-object expressions, called triples, 107

Respiratory Chain Also known as electron transport chain. Enzymes that generate an electron and proton 'gradient' to generate ATP inside mitochondria, 5

SDLC Systems development life cycle. Describe single phases of system design, 26

Slowly Changing Dimensions Handling data changes over the time, 25

Star Schema Data-Warehouse schemas with one fact table consisting of primary keys columns from dimension tables and additional fact columns, 25

Tbox Terminological box. Describes concept hierarchies (relations between concepts), 102

Three-tier Three-tier is a client-server architecture in which the user interface, functional process logic, data storage are developed and maintained as independent modules, 55

Transitive Dependency A dependency caused of other dependencies like: A determines B, and B determines C, then A determines C, 25

TSV Tab-separated-values is a popular method of data interchange among databases and spreadsheets. It encodes a number of records that may contain multiple fields. Each record is represented as a single line. Each field value is represented as text. Fields in a record are separated from each other by a tab character, 73

xBase Term for programming languages that derive from the original dBASE programming language and database format, 62

XHTML Extensible Hypertext Mark-up Language use XML to define a stricter and cleaner version of HTML, 86

XML Extensible Markup Language is a general-purpose markup language recommended by W3C, 73

Bibliography

Agrafioti, I., and M. Stumpf, 2007 Snpstr: a database of compound microsatellite-snp markers. Nucleic Acids Res **35**: D71–5.

Anderson, S., A. Bankier, B. Barrell, M. de Bruijn, A. Coulson, *et al.*, 1981 Sequence and organization of the human mitochondrial genome. Nature **290**: 457–65.

Ankel-Simons, F., and J. Cummins, 1996 Misconceptions about mitochondria and mammalian fertilization: implications for theories on human evolution. Proc Natl Acad Sci U S A **93**: 13859–63.

Ashburner, M., C. Ball, J. Blake, D. Botstein, H. Butler, *et al.*, 2000 Gene ontology: tool for the unification of biology. the gene ontology consortium. Nat Genet **25**: 25–9.

Bachman, C. W., 1969 Data structure diagrams. DATA BASE **1(2)**: 1(2):4–10.

Barry, W., 1988 A spiral model of software development and enhancement. IEEE WESCON,the Institute of Electrical and Electronics .

Batista, O., C. Kolman, and E. Bermingham, 1995 Mitochondrial dna diversity in the kuna amerinds of panama. Hum Mol Genet **4**: 921–9.

Bensasson, D., M. Feldman, and D. Petrov, 2003 Rates of dna duplication and mitochondrial dna insertion in the human genome. J Mol Evol **57**: 343–54.

Benson, D., I. Karsch-Mizrachi, D. Lipman, J. Ostell, and D. Wheeler, 2007 Genbank. Nucleic Acids Res **35**: D21–5.

Bingham, J., and S. Sudarsanam, 2000 Visualizing large hierarchical clusters in hyperbolic space. Bioinformatics **16**: 660–1.

Blackman, K. R., 1998 Technical note - ims celebrates thirty technical note - ims celebrates thirty years as an ibm product. IBM Systems Journal **37(4)**: 596–603.

Brandon, M., M. Lott, K. Nguyen, S. Spolim, S. Navathe, *et al.*, 2005 Mitomap: a human mitochondrial genome database–2004 update. Nucleic Acids Res **33**: D611–3.

Brown, M., E. Starikovskaya, O. Derbeneva, S. Hosseini, J. Allen, *et al.*, 2002 The role of mtdna background in disease expression: a new primary lhon mutation associated with western eurasian haplogroup j. Hum Genet **110**: 130–8.

Brown, W., 1980 Polymorphism in mitochondrial dna of humans as revealed by restriction endonuclease analysis. Proc Natl Acad Sci U S A **77**: 3605–9.

Burckhardt, F., A. von Haeseler, and S. Meyer, 1999 Hvrbase: compilation of mtdna control region sequences from primates. Nucleic Acids Res **27**: 138–42.

Cann, R., M. Stoneking, and A. Wilson, 1987 Mitochondrial dna and human evolution. Nature **325**: 31–6.

Catalano, D., F. Licciulli, A. Turi, G. Grillo, C. Saccone, *et al.*, 2006 Mitores: a resource of nuclear-encoded mitochondrial genes and their products in metazoa. BMC Bioinformatics **7**: 36.

Chamberlin, Donald D. Boyce, R. F., 1974 Sequel: A structured english query language. Proceed- Proceedings of 1974 ACM SIGFIDET Workshop on Data Description, Access and Control. : 249–264.

Codd, E., 1970 A relational model of data for large shared data banks. Association for Computing Machinery (ACM) **13**: 377–87.

Codd, E., 1972 Relational completeness of data base sublanguage. Database Systems .

Codd, E., 1979 Extending the database relational model to capture more meaning. ACM Transactions on Database Systems (TODS) **4**: 397–434.

Crane, D., E. Pascarello, and J. D., 2006 *Ajax in Action. Das Entwickler für das Web 2.0*. Addison-Weley.

Denaro, M., H. Blanc, M. Johnson, K. Chen, E. Wilmsen, *et al.*, 1981 Ethnic variation in hpa 1 endonuclease cleavage patterns of human mitochondrial dna. Proc Natl Acad Sci U S A **78**: 5768–72.

Di Rienzo, A., and A. Wilson, 1991 Branching pattern in the evolutionary tree for human mitochondrial dna. Proc Natl Acad Sci U S A **88**: 1597–601.

Feagin, J., M. Gardner, D. Williamson, and R. Wilson, 1991 The putative mitochondrial genome of plasmodium falciparum. J Protozool **38**: 243–5.

Felsenstein, J., 2005 *PHYLIP (Phylogeny Inference Package) version 3.6*. Distributed by the author. Department of Genome Sciences, University of Washington, Seattle.

Felsenstein, J., and G. Churchill, 1996 A hidden markov model approach to variation among sites in rate of evolution. Mol Biol Evol **13**: 93–104.

Feuerstein, S., and B. Pribyl, 2005 *Oracle PL/SQL Programming, 4nd Edition*. O'Reily Media, INC.

Grimes, B., 2000 *Ethnologue: Volume 1 Langauges of the World*. SIL International, Dallas, TX.

Handt, O., S. Meyer, and A. von Haeseler, 1998 Compilation of human mtdna control region sequences. Nucleic Acids Res **26**: 126–9.

Helgason, A., S. Sigureth ardottir, J. Gulcher, R. Ward, and K. Stefansson, 2000 mtdna and the origin of the icelanders: deciphering signals of recent population history. Am J Hum Genet **66**: 999–1016.

Horai, S., K. Hayasaka, R. Kondo, K. Tsugane, and N. Takahata, 1995 Recent african origin of modern humans revealed by complete sequences of hominoid mitochondrial dnas. Proc Natl Acad Sci U S A **92**: 532–6.

Ingman, M., and U. Gyllensten, 2006 mtdb: Human mitochondrial genome database, a resource for population genetics and medical sciences. Nucleic Acids Res **34**: D749–51.

Ingman, M., H. Kaessmann, S. Paabo, and U. Gyllensten, 2000 Mitochondrial genome variation and the origin of modern humans. Nature **408**: 708–13.

Inmon, W., 1994 *Using the data warehouse*. Willey, J, first edition.

ISO-8601, 1988 Data elements and interchange formats — information interchange — representation of dates and times. ISO .

ISO-9075, 2003 Information technology – database languages – sql. ISO/IEC .

Jobling, M., M. Hurles, and C. Tyler-Smith, 2004 *Human Evolutionary Genetics*. Garland.

Katoh, K., K. Kuma, H. Toh, and T. Miyata, 2005 Mafft version 5: improvement in accuracy of multiple sequence alignment. Nucleic Acids Res **33**: 511–8.

Kishino, H., and M. Hasegawa, 1989 Evaluation of the maximum likelihood estimate of the evolutionary tree topologies from dna sequence data, and the branching order in hominoidea. J Mol Evol **29**: 170–9.

Kivisild, T., M. Reidla, E. Metspalu, A. Rosa, A. Brehm, *et al.*, 2004 Ethiopian mitochondrial dna heritage: tracking gene flow across and around the gate of tears. Am J Hum Genet **75**: 752–70.

Kolman, C., E. Bermingham, R. Cooke, R. Ward, T. Arias, *et al.*, 1995 Reduced mtdna diversity in the ngobe amerinds of panama. Genetics **140**: 275–83.

Krings, M., C. Capelli, F. Tschentscher, H. Geisert, S. Meyer, *et al.*, 2000 A view of neandertal genetic diversity. Nat Genet **26**: 144–6.

Lehner, W., 2003 *Datenbanktechnologie für Data-Warehouse-Systeme*, volume 1. dpunkt.verlag.

Liou, C., T. Lin, H. Huei Weng, C. Lee, T. Chen, *et al.*, 2007 A common mitochondrial dna variant and increased body mass index as associated factors for development of type 2 diabetes: Additive effects of genetic and environmental factors. J Clin Endocrinol Metab **92**: 235–9.

Ludwig, W., O. Strunk, W. R., L. Richter, H. Meier, *et al.*, 2004 Arb: a software environment for sequence data. Nucleic Acids Res **32**: 1363–1371.

Lum, J., O. Rickards, C. Ching, and R. Cann, 1994 Polynesian mitochondrial dnas reveal three deep maternal lineage clusters. Hum Biol **66**: 567–90.

Macaulay, V., M. Richards, E. Hickey, E. Vega, F. Cruciani, *et al.*, 1999 The emerging tree of west eurasian mtdnas: a synthesis of control-region sequences and rflps. Am J Hum Genet **64**: 232–49.

Merriwether, D., A. Clark, S. Ballinger, T. Schurr, H. Soodyall, *et al.*, 1991 The structure of human mitochondrial dna variation. J Mol Evol **33**: 543–55.

Minh, B., S. Vinh le, A. von Haeseler, and H. Schmidt, 2005 piqpnni: parallel reconstruction of large maximum likelihood phylogenies. Bioinformatics **21**: 3794–6.

Monson, L., K. Miller, M. Wilson, J. DiZinno, and B. Budowle, 2002 The mtdna popuation database: An integrated software and database for forensic comparison. Forensic Science Communications **4**.

Morin, P., J. Moore, R. Chakraborty, L. Jin, J. Goodall, *et al.*, 1994 Kin selection, social structure, gene flow, and the evolution of chimpanzees. Science **265**: 1193–201.

Needleman, S., and C. Wunsch, 1970 A general method applicable to the search for similarities in the amino acid sequence of two proteins. J Mol Biol **48**: 443–53.

O'Brien, E., Y. Zhang, L. Yang, E. Wang, V. Marie, *et al.*, 2006 Gobase–a database of organelle and bacterial genome information. Nucleic Acids Res **34**: D697–9.

Owens, K., 1999 *Oracle PL/SQL - Trigger and Stored Procedures*, volume 2. Prentice Hall PTR.

Pult, I., A. Sajantila, J. Simanainen, O. Georgiev, W. Schaffner, *et al.*, 1994 Mitochondrial dna sequences from switzerland reveal striking homogeneity of european populations. Biol Chem Hoppe Seyler **375**: 837–40.

Quintana-Murci, L., O. Semino, H. Bandelt, G. Passarino, K. McElreavey, *et al.*, 1999 Genetic evidence of an early exit of homo sapiens sapiens from africa through eastern africa. Nat Genet **23**: 437–41.

Redd, A., N. Takezaki, S. Sherry, S. McGarvey, A. Sofro, *et al.*, 1995 Evolutionary history of the coii/trnalys intergenic 9 base pair deletion in human mitochondrial dnas from the pacific. Mol Biol Evol **12**: 604–15.

Ricchetti, M., C. Fairhead, and B. Dujon, 1999 Mitochondrial dna repairs double-strand breaks in yeast chromosomes. Nature **402**: 96–100.

Rice, P., I. Longden, and A. Bleasby, 2000 Emboss: The european molecular biology open software suite (2000). Trends in Genetics **16**: 276–277.

Richards, M., and V. Macaulay, 2001 The mitochondrial gene tree comes of age. Am J Hum Genet **68**: 1315–20.

Richards, M., V. Macaulay, H. Bandelt, and B. Sykes, 1998 Phylogeography of mitochondrial dna in western europe. Ann Hum Genet **62**: 241–60.

Richly, E., and D. Leister, 2004 Numts in sequenced eukaryotic genomes. Mol Biol Evol **21**: 1081–4.

Saitou, N., and M. Nei, 1987 The neighbor-joining method: a new method for reconstructing phylogenetic trees. Mol Biol Evol **4**: 406–25.

Sajantila, A., P. Lahermo, T. Anttinen, M. Lukka, P. Sistonen, *et al.*, 1995 Genes and languages in europe: an analysis of mitochondrial lineages. Genome Res **5**: 42–52.

Satoh, M., and T. Kuroiwa, 1991 Organization of multiple nucleoids and dna molecules in mitochondria of a human cell. Exp Cell Res **196**: 137–40.

Shadel, G., and D. Clayton, 1997 Mitochondrial dna maintenance in vertebrates. Annu Rev Biochem **66**: 409–35.

Shields, G., A. Schmiechen, B. Frazier, A. Redd, M. Voevoda, *et al.*, 1993 mtdna sequences suggest a recent evolutionary divergence for beringian and northern north american populations. Am J Hum Genet **53**: 549–62.

Shoubridge, E., and T. Wai, 2007 Mitochondrial dna and the mammalian oocyte. Curr Top Dev Biol **77**: 87–111.

Smith, T., and M. Waterman, 1981 Identification of common molecular subsequences. J Mol Biol **147**: 195–7.

Stoneking, M., L. Jorde, K. Bhatia, and A. Wilson, 1990 Geographic variation in human mitochondrial dna from papua new guinea. Genetics **124**: 717–33.

Sukernik, R., O. Derbeneva, E. Starikovskaia, N. Volod'ko, I. Mikhailovskaia, *et al.*, 2002 [the mitochondrial genome and human mitochondrial diseases]. Genetika **38**: 161–70.

Sutovsky, P., K. Van Leyen, T. McCauley, B. Day, and M. Sutovsky, 2004 Degradation of paternal mitochondria after fertilization: implications for heteroplasmy, assisted reproductive technologies and mtdna inheritance. Reprod Biomed Online **8**: 24–33.

Tajima, F., 1989 Statistical method for testing the neutral mutation hypothesis by dna polymorphism. Genetics **123**: 585–95.

Tanaka, M., V. Cabrera, A. Gonzalez, J. Larruga, T. Takeyasu, *et al.*, 2004 Mitochondrial genome variation in eastern asia and the peopling of japan. Genome Res **14**: 1832–50.

Taylor, R., and R. Frank, 1976 Codasyl data-base management systems. ACM Computing Surveys (CSUR) **8(1)**.

Thalmann, O., J. Hebler, H. Poinar, S. Paabo, and L. Vigilant, 2004 Unreliable mtdna data due to nuclear insertions: a cautionary tale from analysis of humans and other great apes. Mol Ecol **13**: 321–35.

Thalmann, O., D. Serre, M. Hofreiter, D. Lukas, J. Eriksson, *et al.*, 2005 Nuclear insertions help and hinder inference of the evolutionary history of gorilla mtdna. Mol Ecol **14**: 179–88.

Thomas, D., and D. Heinemeier Hansson, 2006 *Agile Web Development with Rails, Agile Web Development with Rails, Second Edition.*. The Pragmatic Programmers, LLC.

Thompson, J., D. Higgins, and T. Gibson, 1994 Clustal w: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. Nucleic Acids Res **22**: 4673–80.

Torroni, A., A. Achilli, V. Macaulay, M. Richards, and H. Bandelt, 2006 Harvesting the fruit of the human mtdna tree. Trends Genet **22**: 339–45.

Tourmen, Y., O. Baris, P. Dessen, C. Jacques, Y. Malthiery, *et al.*, 2002 Structure and chromosomal distribution of human mitochondrial pseudogenes. Genomics **80**: 71–7.

Tsichritzis, D., and A. Klug, 1978 The ansi/x3/sparc dbms framework report of the study group on database management systems. Information Systems **3**: 173–91.

Unseld, M., J. Marienfeld, P. Brandt, and A. Brennicke, 1997 The mitochondrial genome of arabidopsis thaliana contains 57 genes in 366,924 nucleotides. Nat Genet **15**: 57–61.

Vasconcelos, A., A. Guimaraes, C. Castelletti, C. Caruso, C. Ribeiro, *et al.*, 2005 Mammibase: a mitochondrial genome database for mammalian phylogenetic studies. Bioinformatics **21**: 2566–7.

Vigilant, L., M. Stoneking, H. Harpending, K. Hawkes, and A. Wilson, 1991 African populations and the evolution of human mitochondrial dna. Science **253**: 1503–7.

Vinh le, S., and A. Von Haeseler, 2004 Iqpnni: moving fast through tree space and stopping in time. Mol Biol Evol **21**: 1565–71.

Vossen, G., 2000 *Datenmodelle, Datenbanksprachen und Datenbankmanagementsysteme*. Oldenbourg, fourth edition.

Wallace, D., 1995 Mitochondrial dna variation in human evolution, degenerative disease, and aging. Am J Hum Genet **57**: 201–223.

Ward, R., B. Frazier, K. Dew-Jager, and S. Paabo, 1991 Extensive mitochondrial diversity within a single amerindian tribe. Proc Natl Acad Sci U S A **88**: 8720–4.

Waterman, M., 2000 *Introduction to computational biology - Maps, sequences and genomes*. Chapman & Hall/CRC.

Watson, E., K. Bauer, R. Aman, G. Weiss, A. von Haeseler, *et al.*, 1996 mtdna sequence diversity in africa. Am J Hum Genet **59**: 437–44.

Winston, W., 1999 Managing the development of large software systems. IEEE WESCON,the Institute of Electrical and Electronics : 1–9.

Wise, C., M. Sraml, D. Rubinsztein, and S. Easteal, 1997 Comparative nuclear and mitochondrial genome diversity in humans and chimpanzees. Mol Biol Evol **14**: 707–16.

Xu, X., and U. Arnason, 1996 The mitochondrial dna molecule of sumatran orangutan and a molecular proposal for two (bornean and sumatran) species of orangutan. J Mol Evol **43**: 431–7.

Die hier vorgelegte Dissertation habe ich eigenständig und ohne unerlaubte Hilfe angefertigt. Die Dissertation wurde in der vorgelegten oder in ähnlicher Form noch bei keiner anderen Institution eingereicht. Ich habe bisher keine erfolglosen Promotionsversuche unternommen.

Düsseldorf, den

(Jochen Kohl)