

# Datenbankgestützte Lexika für MALAGA

Matthias Bethke

7. November 2002

# Gliederung

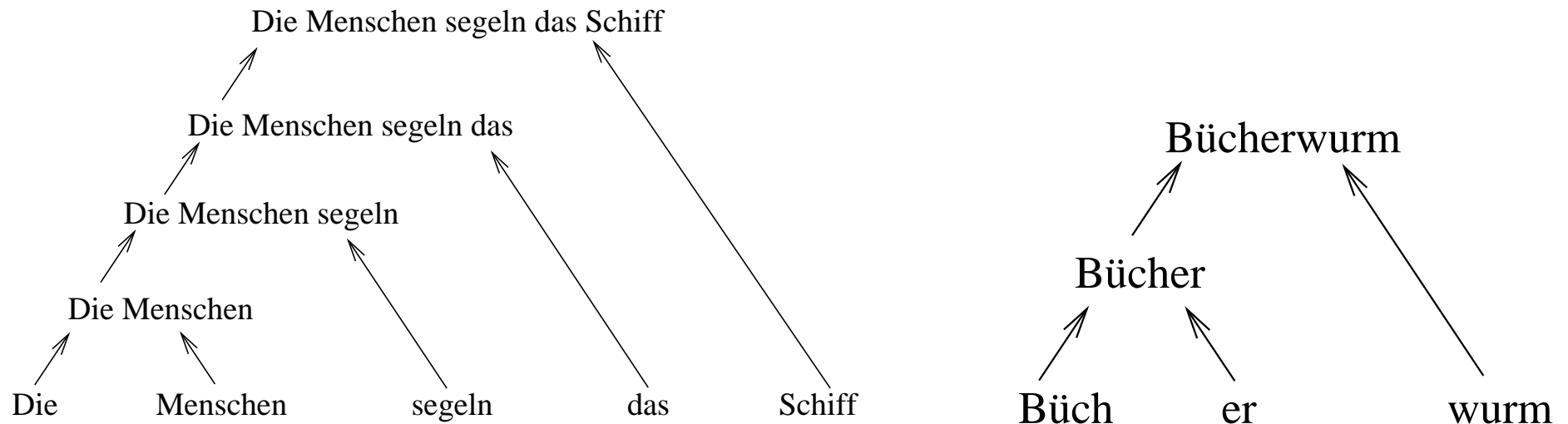
1. Einleitung
2. MALAGA: Stand der Entwicklung, Arbeitsweise, Interna
3. Relationale Datenbanken
4. Anforderungen an eine Lexikondatenbank
5. Tabellenmodell
6. Implementation: Abfragestrategien, Suchtries, Spracherweiterung
7. Praktische Ergebnisse und Probleme

# Einleitung

## Warum Datenbankunterstützung?

- Proprietäre Speicherformate
- Probleme bei verteilter Bearbeitung
- Zeitaufwendige **mallex**-Läufe
- Keine Lexikonerweiterung zur Laufzeit

# LA-Grammatik



$$r_i : cat_1 cat_2 \Rightarrow cat_3 r p_i$$

# Lexika in Malaga

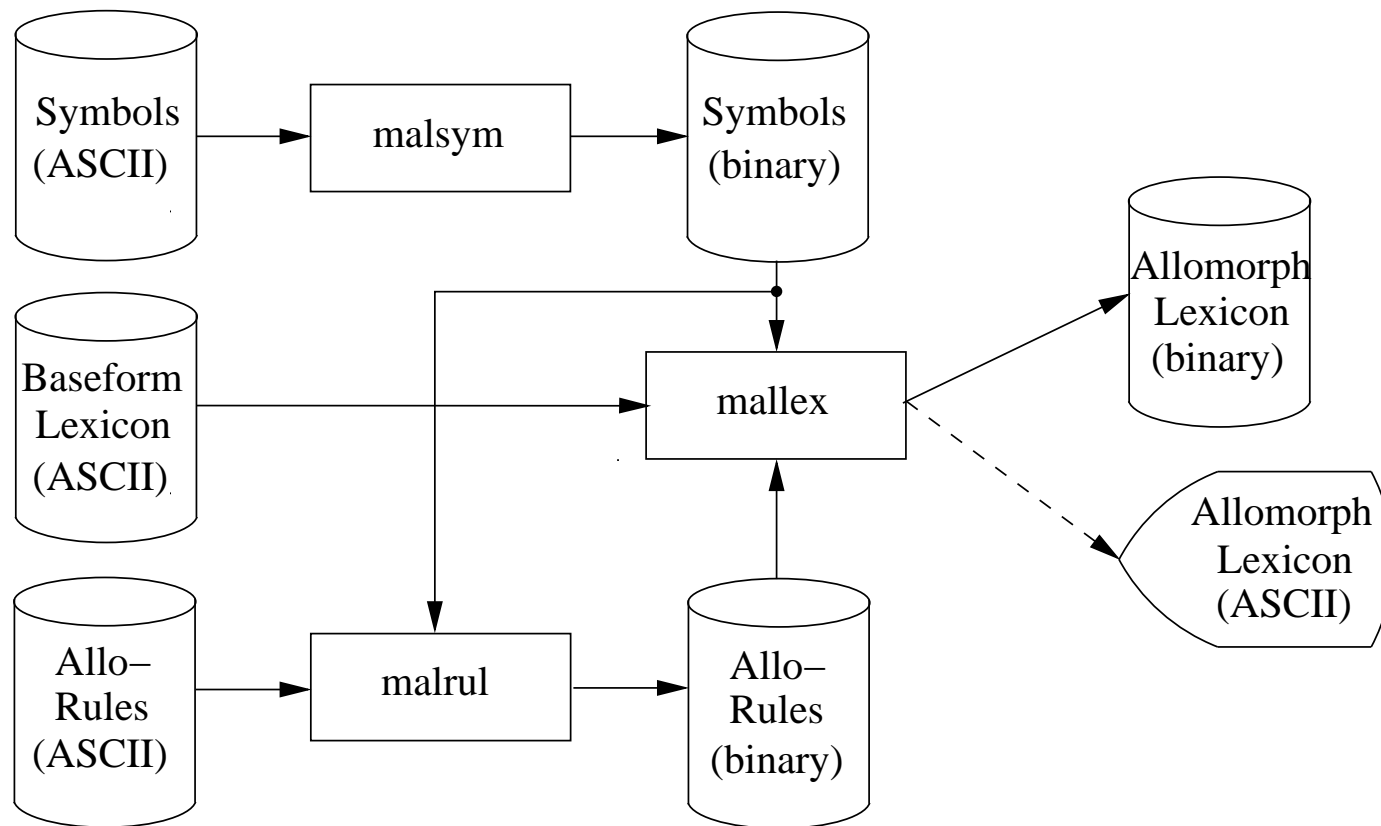
## 1. Grundformlexikon

- Textdatei; speichert Grundformeinträge und ihre Kategorien als Attribut-Werte-Strukturen in spezieller Malaga-Notation.
- Wird von **mallex** durch Anwendung der Allo-Regel auf jeden einzelnen Grundformeintrag kompiliert.

## 2. Allomorphlexikon

- Binärdatei; enthält direkt zur Laufzeit verwendbare kompakte Trie-Struktur.
- Statisch, d.h. bei Änderungen am Grundformlexikon neu zu generieren.

# mallex schematisch



# Morphologie und Lexikon

LA-Maschine/Morphologie: fragt Wortform(teil)oberfläche im Lexikon ab.  
Lexikon: liefert gefundene Allomorphe in Reihenfolge wachsender Länge.

z.B.: „Eisenbahn“

<b>Abfrage</b>	<b>Ergebnis(se)</b>
eisenbahn	e-ei-eis-eisen
senbahn	s-se-sen
enbahn	e-en
nbahn	n
bahn	ba-bahn
hn	

# Warum relationale Datenbanken?

- Verringerung von Redundanz
- Vermeidung von Inkonsistenzen
- Gemeinsame Nutzung von Daten
- Zugriffsbeschränkungen
- Standardisierung

→ CELEX, Verbmobil, BNC/SARA

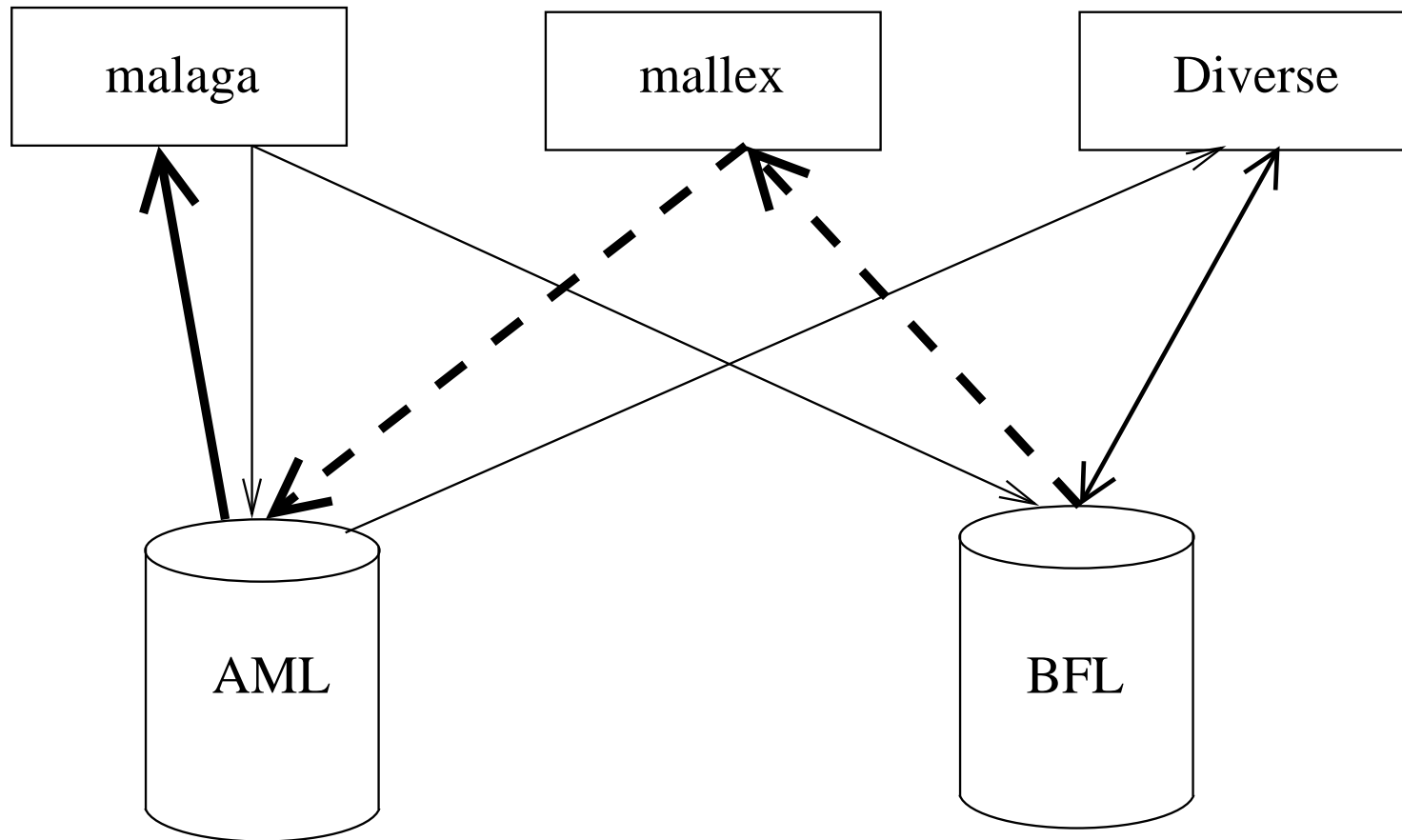
→ Malaga-Datenbank: Scherbaum 1996

# Anforderungen an eine Lexikondatenbank

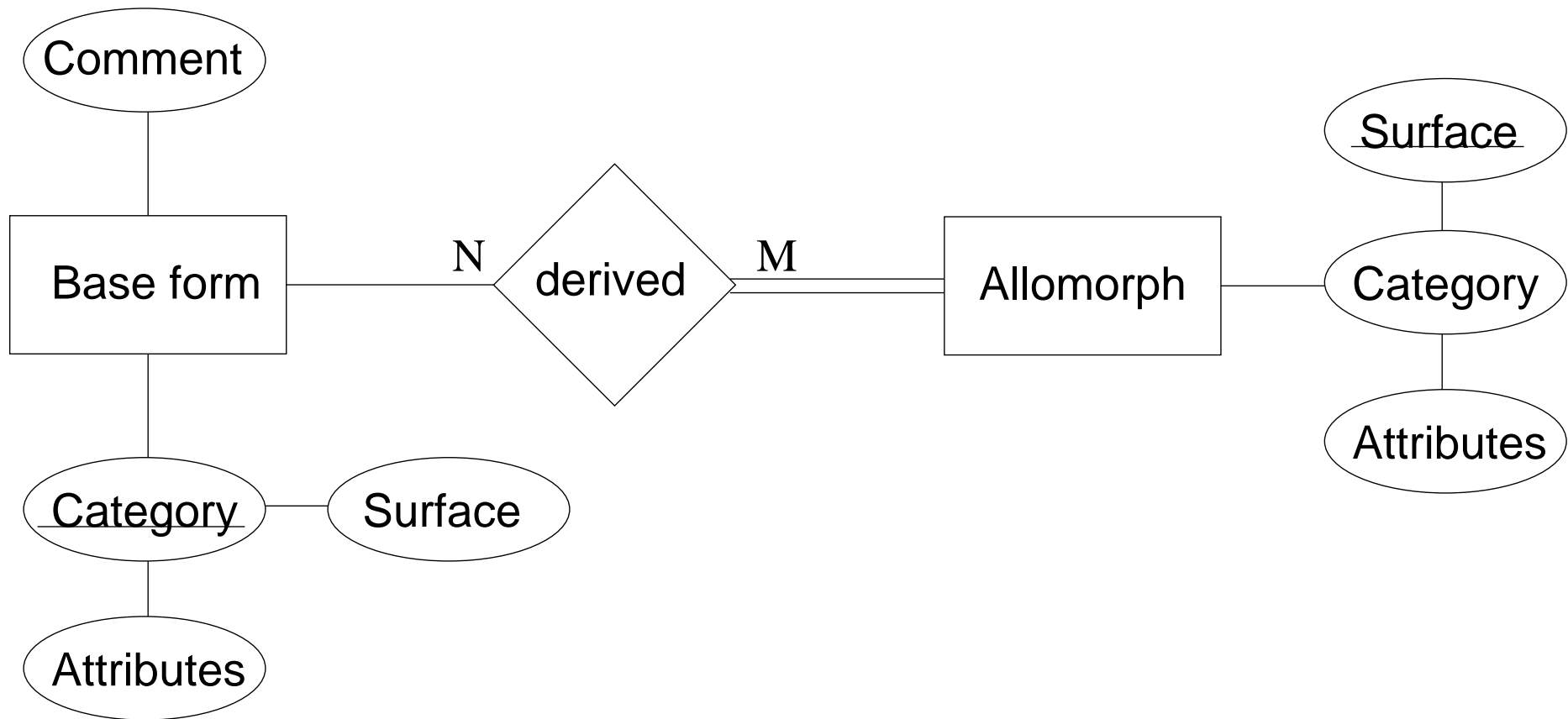
- Aus Anwenderperspektive: stabil, performant, unkompliziert
- Benutzt RDBMS „von der Stange“, Standard-SQL
- Unabhängig von der Sprache oder Details der Morphologie
- Offenes Format erlaubt Abfragen aus anderen Anwendungen
- Erweiterbar zur Laufzeit aus Malaga-Grammatiken

→ Keine lexikographischen Einschränkungen gegenüber dem ASCII-Format  
→ Text- statt Binärdaten für sämtliche Kategorien

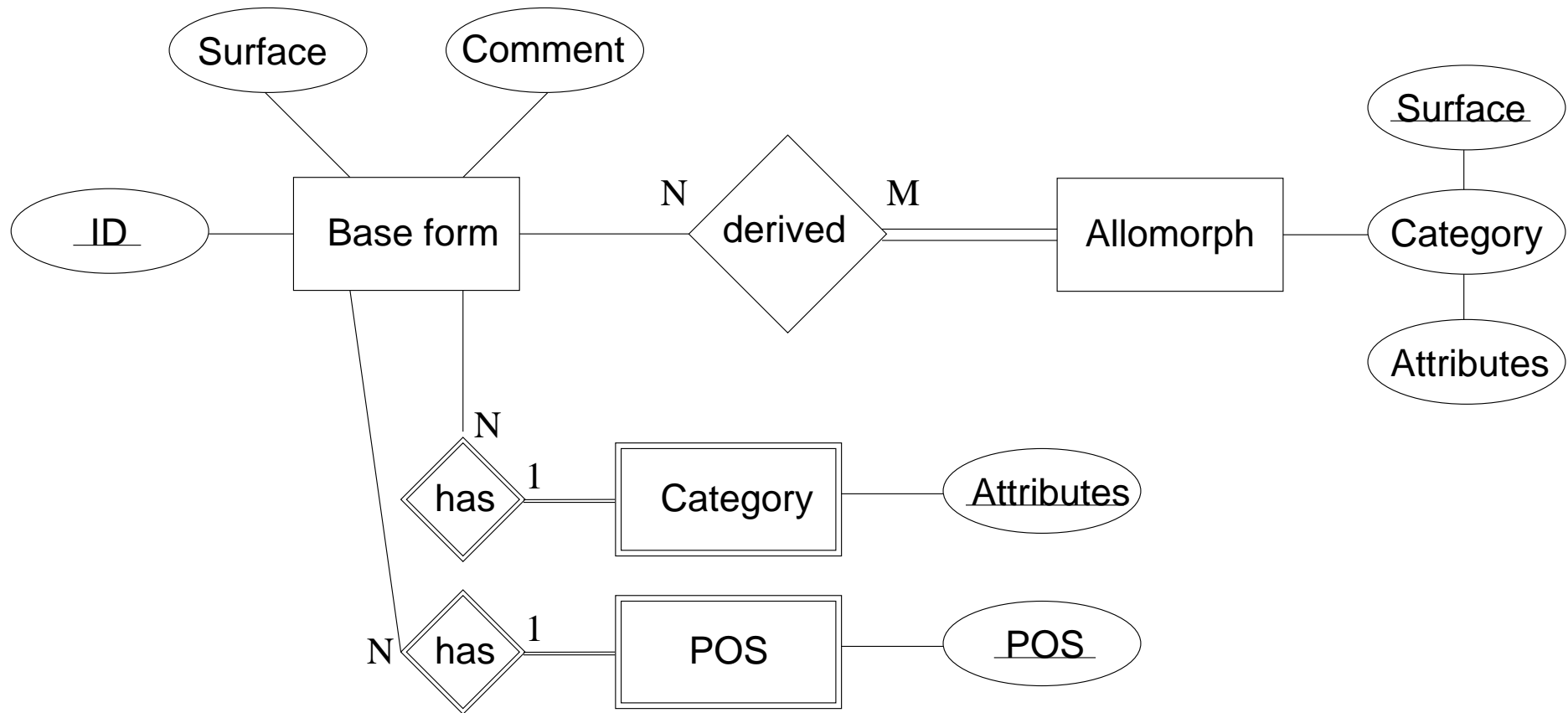
# Datenströme



# ER-Diagramm: Ausgangszustand



# ER-Diagramm: Alternative



# Vergleich der Modelle

## **Modell 1: 3 Tabellen, „flaches“ Grundformlexikon**

- + Minimaler Aufwand
- Wenige Vorteile gegenüber der bisherigen Datei

## **Modell 2: 5-Tabellen-Version**

- + Berücksichtigt lexikographische Anforderungen
- + Nutzt RDB-Eigenschaften
- + Speichersparend → u.U. schneller
- Komplexer, Gefahr der Inkonsistenz
- DB-Schnittstelle muss Details der Lexikonstruktur kennen

# Tabellenmodell

lexicon

LEXID	BFLEX	BFCAT	BFPOS	ALLEX	ABREF	ADATE	BDATE
german	ger_bfl	ger_bfc	ger_bfp	ger_all	ger_abr	20020610120133	20020610115851

ger\_bfl

ID	LEMMA	POSNUM	CATNUM	AG	COMMT
1	"absolut"	1	NULL	0	
2	"all"	2	1	0	
3	"allerlei"	3	2	0	
4	"allzu"	1	NULL	0	

ger\_bfc

ID	CATEGORY
1	[PluralSx: Pl_s, Gender:...] ]
2	[Pronoun: Pl3 Gen Dat Acc, DetFlex: none, ...]

ger\_bfp

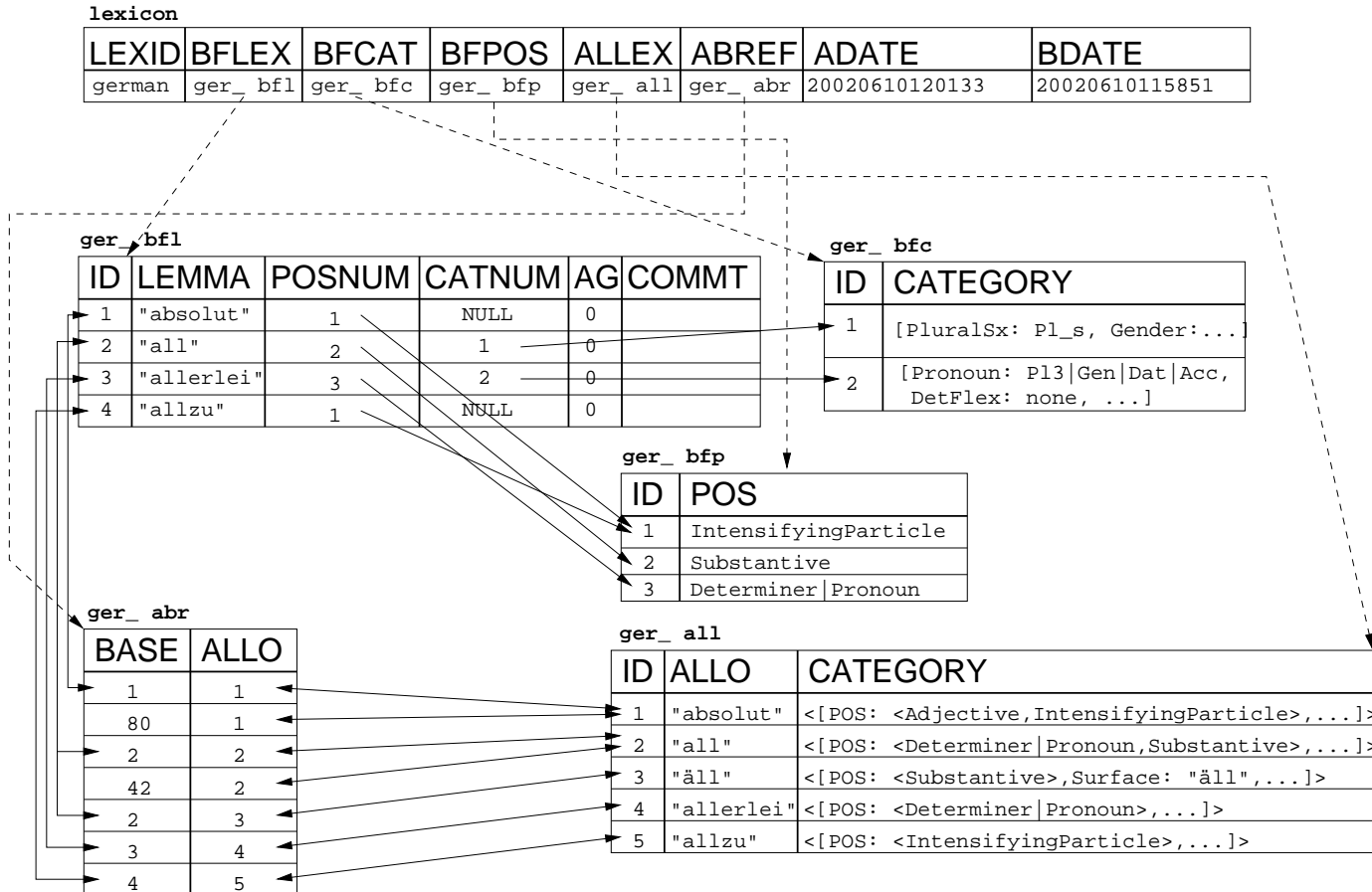
ID	POS
1	IntensifyingParticle
2	Substantive
3	Determiner Pronoun

ger\_abr

BASE	ALLO
1	1
80	1
2	2
42	2
2	3
3	4
4	5

ger\_all

ID	ALLO	CATEGORY
1	"absolut"	<[POS: <Adjective, IntensifyingParticle>, ...]>
2	"all"	<[POS: <Determiner Pronoun, Substantive>, ...]>
3	"äll"	<[POS: <Substantive>, Surface: "äll", ...]>
4	"allerlei"	<[POS: <Determiner Pronoun>, ...]>
5	"allzu"	<[POS: <IntensifyingParticle>, ...]>



# Abfragestrategien

**mallex:** unkritisch, da relativ seltener gebraucht. Lexika müssen komplett gelesen bzw. geschrieben werden.

**malaga:**

- Zeitkritisch beim Lesezugriff auf AML
- Benötigt aus Geschwindigkeitsgründen weiterhin einen Trie
- Seltene Schreibzugriffe auf beide Lexika bei dynamischer Erweiterung
- Problem: Startzeit. Lösungsmöglichkeiten:
  1. Einlesen des kompletten Lexikons beim Start
  2. Inkrementelles Laden mit SQL-Präfixsuche
  3. Trie-Aufbau beim Start, Kategorien nachladen

# Trie-Struktur

- Zur Laufzeit erweiterbar
- Erlaubt Präfixsuche
- Kompakt und schnell

Möglichkeiten: BST, AVL-Tree, TST, Trie, Hybridstrukturen

→ Burst Trie: Einstellbarer Speicherverbrauch; optimal für die Suche nach kurzen Zeichenketten

# Spracherweiterung

**Ziel:** Eintrag einer Worthypothese ins Lexikon aus einer laufenden Grammatik.

**Funktion:** `addlex($surfstr, $possym, $cat)`

Generiert Allomorphe und trägt Grundform+Allomorphe in die Datenbank ein, letztere auch ins Laufzeitlexikon.

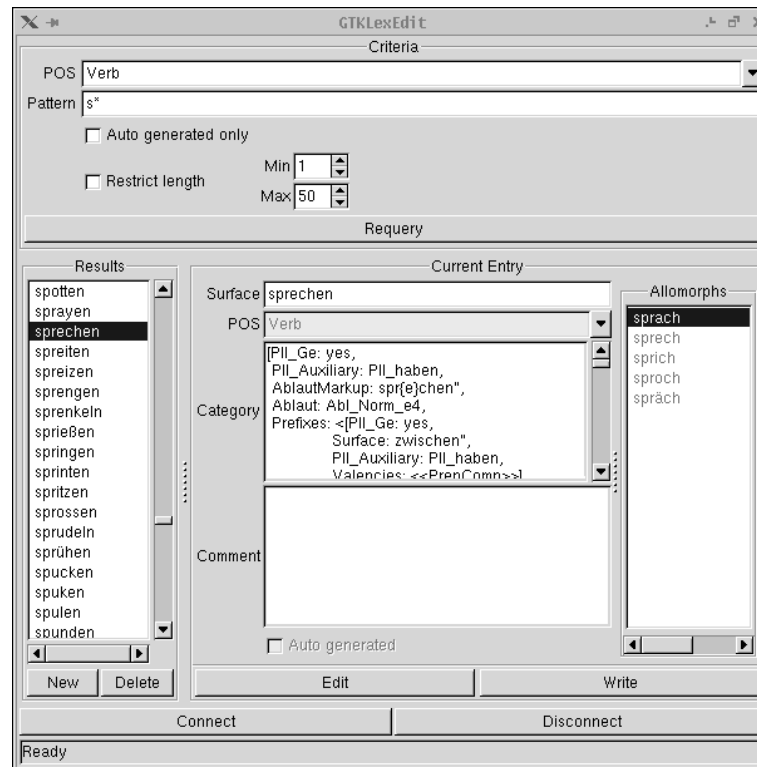
**Problem:** Globale Zustandsvariablen, keine Parallelausführung von weiteren Regelsystemen vorgesehen.

# Hilfsprogramme

- Malaga.pm: Perl-Schnittstelle zu Malaga (Piotrowski 1998)
- MalagaDB.pm: Perl-Modul mit einigen Datenbankfunktionen, basierend auf DBI.pm
- CreateMalagaDB.sh: Legt eine Lexikondatenbank an
- InitMalagaLex.pl: Initialisiert ein neues Lexikon
- MalagaLex2DB.pl: Liest ASCII-Lexika in die Datenbank

# GTKLexEdit

Demonstrationsanwendung: Lexikoneditor mit grafischer Oberfläche.



# Mögliche Änderungen/Erweiterungen

- GTKLexEdit
- Optimierung des Trie-Algorithmus
- Portierung der DB-Schnittstelle
- Optionale Kompression der Kategorien (zlib)
- Integration weiterer Malaga-Komponenten: Symboldatei, Allo-Regel?
- Abstraktion der Programmierschnittstelle