

---

# Aufbereitung unbekannter unstrukturierter Dokumente für Oracle Text

Markus Schwabe, Thomas Pahl  
codework Software GmbH

- Kleines SW-Haus
- Gegründet 1984
- Ursprünglich tätig im IBM MVS und BS2000 Umfeld
- Unix, Linux, Oracle, Security
- Consulting
- Entwicklung
- Technologie/Realisierungs-Partner
- Oracle seit 1987
- Portierung Oracle auf BS2000 (5.0 – 7.1)
- Oracle + Web seit 1995
- 8 Jahre im DOAG-Vorstand, aktuell im Beirat, [www.doag.org](http://www.doag.org)
- PL/SQL, PHP, (Java)
- Oracle Text / Dokumente als wachsender Schwerpunkt

# Inhalte des Vortrags

---

- Aufgabenstellung – Umfeld und Detail
- Warum nicht CMSDK? Gründe einer eigenen Lösung
- Struktur der Dokumentspeicherung
- Der Weg bis zur Recherche – Die Dokumentaufbereitung
- Wechsel von 9.2 auf 10.2 – erste Erfahrungen
- Wünsche an Oracle
- Q&A

# Aufgabenstellung - Anforderungen

---

- Projekt im Bereich deutscher Sicherheitsbehörden
- Schneller Import von (sehr) großen Dateimengen bis zur Recherchefähigkeit, unbekannte Inhalte
- Mehrfache, auch große, Nachimporte
- Text-Dokumente in verschiedenen (nicht bekannten) Sprachen
- Originale unverändert zugreifbar
- Metadaten erschließen (Office, Bilder(Fotos), ...)
- Besondere Auswertung von Email (hier nicht betrachtet)
- Mandanten-fähig (strikte Trennung, übergreifende Recherchen)
- Vorgabe: ca. 500.000 Dokumente je Mandant, >2 TB über alle Mandanten

# Aufgabenstellung - Detail

---

- Dokumente unterschiedlichen Typs:
  - Office-Dokumente verschiedener Hersteller (doc, xls, ppt, sdw, odt etc.)
  - weiterhin pdf, txt, html, xml
  - Multimedia-Dateien wie Bilder (bmp, jpg, gif, png u.a.), Audio- (wav, mp3) und Video-Daten (mpeg, wmf, avi etc.)
  - Mails aus Outlook-Anwendung
  - OCR-scanned Dokumente möglich
- Datei-Extension ist unzuverlässig

mehr..

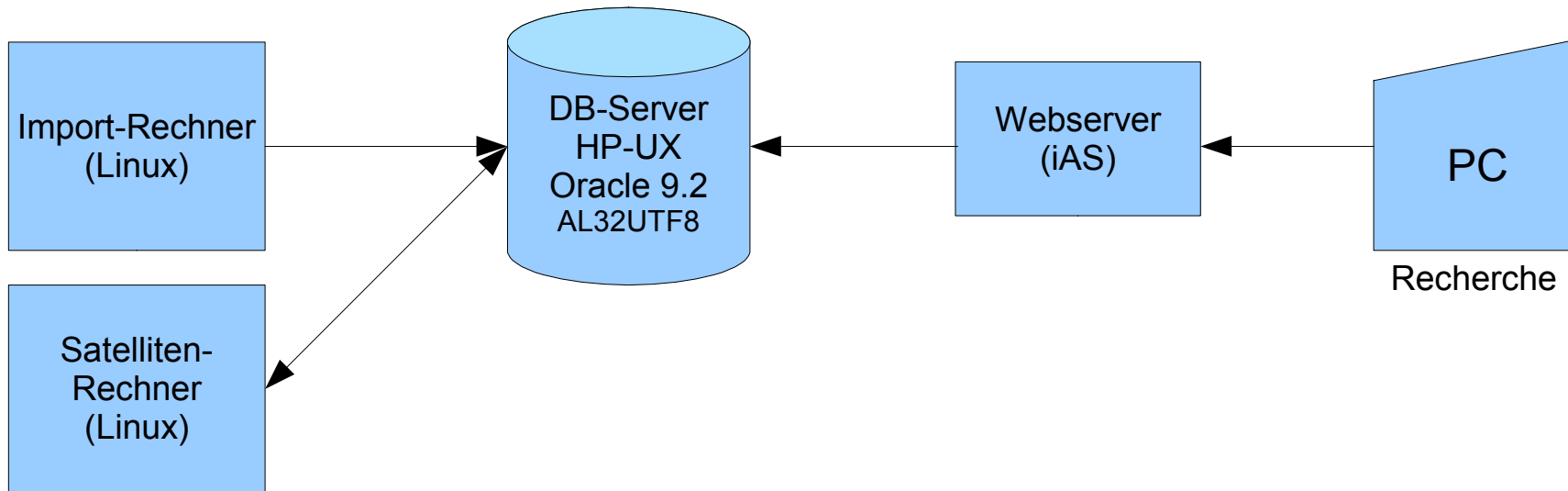
# Aufgabenstellung - Detail

---

- Erste Dokumentaufbereitung findet ausserhalb der Applikations-Kontrolle statt
- Dateien können vom Import ausgeschlossen werden, wenn...
  - es ausführbare Dateien sind
  - die Dateien aufgrund ihres Hashwertes sonst als nicht relevant für eine Recherche gelten (z.B. DLLs, INIs etc. - Hash-Bibliotheken)
  - die Dateien virenverseucht sind
- Archive (.zip usw.) werden entpackt

# Aufgabenstellung - Architektur Anwendung

---



- Software

- Webanwendung pure PL/SQL
- PHP und Linux Tools auf Import/Satellitenrechner

# Warum nicht CMSDK?

## Gründe für eine eigene Lösung

---

- CMSDK hat andere Zielsetzung (CMSDK)
- Zu langsam (Java)
- Speichermanagement
  - X Terabyte Ziel
  - delete 1 Mio LOBs?
  - backup/restore?
- Mehrsprachigkeit problematisch
- Gefordertes aufwendiges Rollensystem (Rechtemanagement)  
nicht ganz einfach

# Struktur der Dokumentenspeicherung

---

- klassische Dokumententabelle mit Format-, Sprache- und Zeichensatzspalte (wichtig für Index-Steuerung)
- Original-Dokumente liegen in BLOBs
- Zusätzliche Text-Spalte (CLOB), die den gefilterten Text aufnimmt
- Dokumententabelle wird dynamisch beim Einrichten eines Mandanten erstellt => unterschiedliche Text-Index-Strategien pro Mandant möglich (Partitionierung keine Lösung, auch nicht List-Partitioning in 11g, da auf gleicher Präferenz-Basis aufgesetzt wird)
- Zusatztabelle für Metadaten (XML)

# Der Weg bis zur Recherche - der Weg zum Volltext-Index

---

## Geradeaus

- Dokumente importieren, alle Dateien unbesehen indizieren
- Indexlauf konnte mehrere Tage dauern, Abbruch war oft fatal
- Keine Informationen, was eigentlich indiziert wird und ob bestimmte Dokumente überhaupt indiziert werden müssen
- Einfach zu realisieren

## Mit Vorab-Bearbeitung

- Mehrstufige Analyse auf Dokumenttyp usw., Text separat filtern + speichern
- Zahl zu filternder und indizierender Dokumente wird minimiert
- Indexerstellung geht schnell und robust
- Gewinnung von Zusatzinformationen (Sprache, Metadaten)
- Mehr Arbeit, die lohnt

# Der Weg bis zur Recherche - Probleme beim Geradeaus-Ansatz

---



- Zeichensatz bei ascii-Text-Dokumenten unbekannt, Filter könnte falsch konvertieren
- Filter muss auch Media- und Müll-Dokumente verarbeiten, obwohl keine Informationen extrahiert werden können
- bei fatalem Indexabbruch (no resume) muss Text-Index neu gestartet werden; Indexlauf kann mehrere Tage dauern
- Sprache unbekannt, Einrichten eines (Multi-)Lexers basiert auf Annahmen, sinnlos ohne Sprachspalte
- Keine Information aus Metadaten aus Office-Dokumenten (Autor, Stichworte etc.) und Bildern (EXIF, IPTC, XMP)
- EMail-Recherche nach Metadaten (From, To etc.) nicht möglich

# Der Weg bis zur Recherche - Texte indizieren/filtern – technische Probleme

---



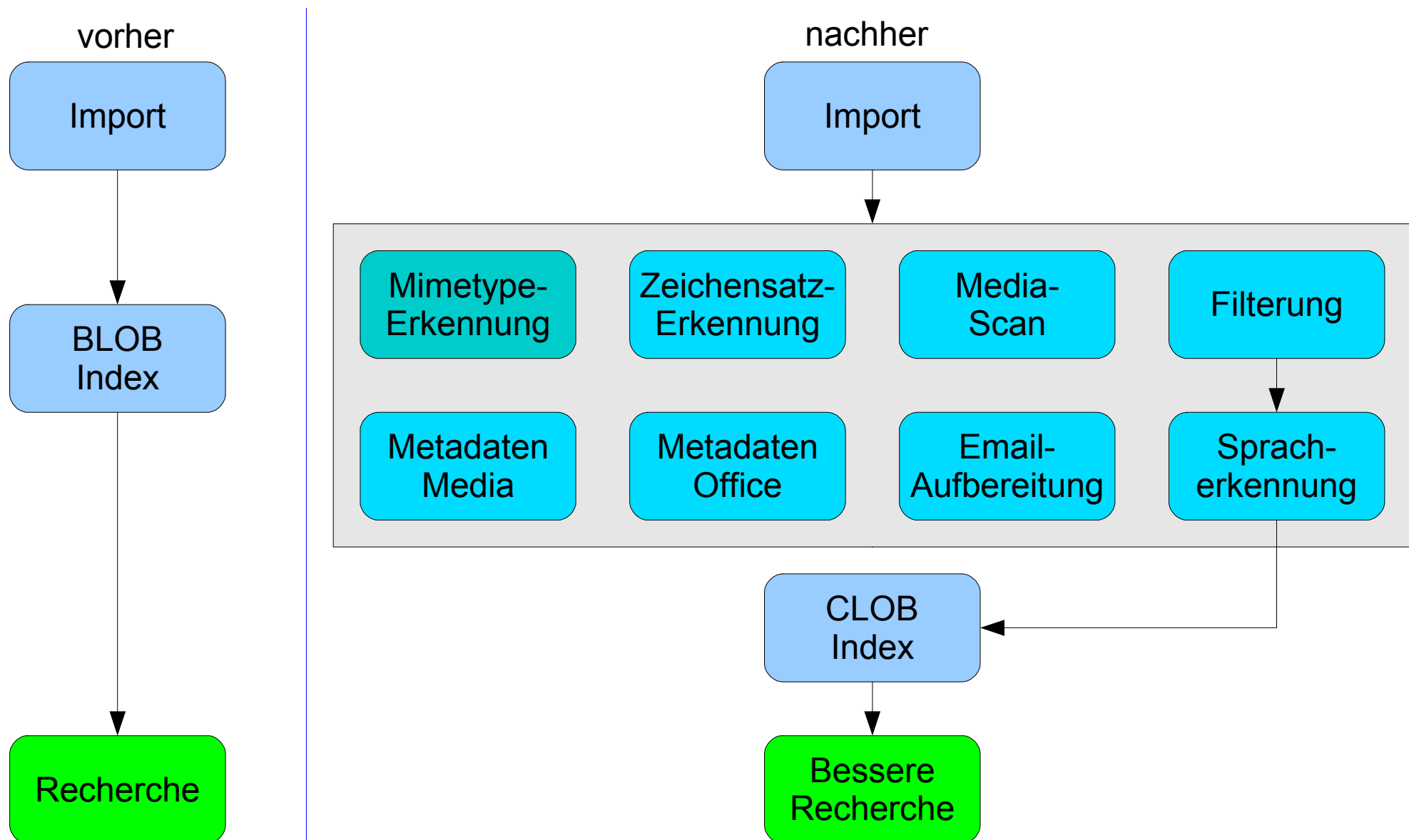
- Problem 9.2: INSO Filter bleibt sporadisch hart hängen (1:100.000), Datei-abhängig (bestimmte HTML, andere)
  - Workaround 1: cron job mit kill -9 auf ctxhx
  - Workaround 2: nur filtern was unbedingt nötig ist (Mimetype-Erkennung vorher)

# Der Weg bis zur Recherche - Dokumentaufbereitung und Analyse vor dem Index

---

- Erweiterung aus Erfahrung - zweite Version:
  - Dokumenttyp wird anhand eindeutigem Mimetype identifiziert
  - Sprache von Texten wird automatisch erkannt
  - Der Zeichensatz für ascii-Text-Dokumente wird bestimmt
  - Metadaten aus Office-Dokumenten und Bildern werden extrahiert
  - EMail-Daten werden aufbereitet
- Vorteile u.a.:
  - Ausschluss von Dokumenten, die nicht indiziert werden müssen
  - automatische Einrichtung eines (Multi-)Lexers anhand erkannter Sprachen; ab 10g Einsatz des World-Lexer als Default-Lexer
  - erweiterte EMail-Recherche und Metadaten-Recherche
  - genaue Statistiken zu allen (wirklichen) Dokumenttypen

# Der Weg bis zur Recherche - die Dokumentaufbereitung



# Der Weg bis zur Recherche - viele kleine Schritte

---

- Mimetype-Erkennung: essentielle Basis einer Dokumentaufbereitung von unbekanntem Dokumenten
- Metadaten-Extraktion: Recherche-Mehrwert
- Zeichensatz-Erkennung: wichtig für die korrekte Umwandlung von binär abgelegten ascii-Text-Dokumenten
- Automatische Erkennung der Sprache des Dokuments: genaue Einstellung von Lexer/Wordlist/Stoplist möglich - Multi-Lexer

# Der Weg bis zur Recherche - Texte filtern



- Aufteilung des Prozesses in seine 2 Teile: filtern und indizieren
  - Dokumente werden mit CTX\_DOC gefiltert, Nettotext (HTML) in CLOB abgelegt (Platz gegen Zeit)
- Vorteile:
  - Filtern „offline“, berührt operativen Index noch nicht
  - Indizierung danach VIEL schneller, Hit Markup ebenfalls
  - Text liegt für weitere Auswertungen bereit (Sprache, Mining)
- Probleme:
  - Filtern non-text gibt Unsinn („IMAGE1“), kostet viel Zeit
  - IFILTER nicht parametrierbar, FILTER braucht Index-Definition (kills direct path load)
  - 10g bietet brauchbarere Funktion an (POLICY\_FILTER)

# Der Weg bis zur Recherche - Mimetype-Erkennung

---



- Was ist drin in der Datei?
- Mimetype-Erkennung in mehreren Schritten – there is no „one serves it all“
- 1) es werden alle Dokumente über das Linux-Kommando „file -i“ interpretiert (magic numbers, pattern rules)
  - findet vor dem Import statt
  - schnell, sicher für ascii-Text-Dokumente
  - Binärdokumente ungenau (Excel ist Word, usw., Heuristik versagt zu oft)
- 2) Mimetype-Zuordnung als Nebennutzen von Metadaten-Extraktion Multimedia-Dateien innerhalb Oracle (s.u.)

mehr..

# Der Weg bis zur Recherche - Mimetype-Erkennung

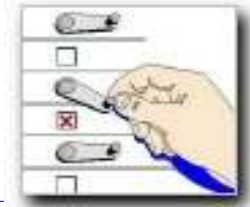
---



- 3) direkter Aufruf von „ctxhx“ (dem Filterprogramm)
  - nur für Dateien, die bis dahin nicht als „text/...“, „image/...“, „audio/...“ identifiziert wurden
  - gleichzeitig wird der gefilterte Text extrahiert und separat abgespeichert; Index wird nun auf Text CLOB statt auf BLOB aufgesetzt
  - Metalink-Note **246178.1** „How to extract Metadata from Documents“ (Hinweis: andere Metadaten nur unzureichend, besser sind externe Tools, s.u.)
  - Natürlich kein „committed“ feature...

# Der Weg bis zur Recherche - Metadaten-Extraktion

---



- Bilder

- Für alle Dokumente, die nicht ascii-Text sind, wird ein Bilder-Scan durchgeführt
- **Hilfsmittel:** `ordsys.ordimage.getProperties`
- schnell
- Erkannt werden jpeg, tif, gif, png, bmp und andere Formate
- Für alle anderen Dokumente wird eine Exception generiert, die ignoriert werden kann (kein Bild)

- Ergebnisse

- Mimetype
- technische Metadaten (Größe, Höhe, Breite, Kompression u.a.)

mehr..

# Der Weg bis zur Recherche - Metadaten-Extraktion, 10g

---



- Bilder, neu in 10g
  - Anschliessend werden EXIF/IPTC-Metadaten aus jpg- und tif-Bildern extrahiert (10g)
  - auch XMP und DICOM möglich (10g)
  - erspart offline-Lösungen (via Linux Satellit)
- Nach dem Bilder-Scan wird gleichartig ein Audio- und ein Video-Scan durchgeführt (10g)
  - Ergebnisse: Mimetype und technische Metadaten
- Extrahierte Metadaten werden als XML abgespeichert und dem Anwender für eine Recherche zur Verfügung gestellt

mehr..

# Der Weg bis zur Recherche - Metadaten-Extraktion

---



- Office-Dokumente
  - Metadaten werden über das Programm „libextractor“ extrahiert  
<http://www.gnu.org/software/libextractor/libextractor.de.html>  
(remote auf Linux Satellit)
  - unterstützte Formate: HTML, PDF, DOC, XLS, PPT, OpenOffice (sxw), StarOffice (sdw) u.a.
  - Metadaten können sein:
    - Mimetype
    - Betriebssystem
    - Titel, Autor, Version, Firmenname
    - Seitenanzahl, Änderungsdatum
    - etc.
  - Ergebnisse werden in XML umgewandelt und abgespeichert

# Der Weg bis zur Recherche - Zeichensatzerkennung

---



- wichtig für binär abgelegte ascii-Text-Dokumente
- Zeichensatz wird genau dann ermittelt, wenn das Dokument als „text/...“ erkannt wurde und noch nicht als CLOB vorliegt
- Erkennung erfolgt über Java-Applikation „ICU“  
<http://icu.sourceforge.net/>, IBM-unterstützt und Open-Source
- ISO-Zeichensatz kann über neues 10g-Package UTL\_I18N in einen Oracle-Zeichensatz gemappt werden
- Dokument bleibt als Original in BLOB-Spalte
- Text wird in UTF8 im CLOB abgelegt
- Diese Dateien werden nicht mehr gefiltert

# Der Weg bis zur Recherche - Mehrsprachigkeit

---



- Spracherkennung
  - Spracherkennung erfolgt über externes Programm, hier „Xtramind“ (Güte abhängig von Trainingssets)
  - Spracherkennung wird nach dem Filtern der Dokumente durchgeführt (auf den CLOBs)
  - Spracherkennung ist nicht eindeutig: best guess wird in CONTENT\_LANGUAGE abgelegt
  - vermutete andere Sprachen werden in separater Metatabelle gespeichert
- Mixed-Language Dokumente werden noch nicht unterstützt (Dokument müsste zur Recherche ggf. in Sprachsegmente zerlegt werden)

mehr..

# Der Weg bis zur Recherche - Mehrsprachigkeit - Multilexer

---



- Manuelle Pflege der Multilexer-Einstellungen überfordert Anwender (trotz Web-Wizard)
- automatisch anhand der Spracherkennung Multilexer erweitern
- World-Lexer (10g)
  - Wertet Unicode aus (nicht Sprach-, sondern Schrifterkennung)
  - Ersetzt nicht Spracherkennung / Multilexer (stemming usw. fehlt)
  - Hier wertvoll: spezielle Features für Arabisch
  - Ideal als Default Lexer im Multilexer Setup
  - (Erfahrungen liegen noch nicht ausreichend vor)

mehr..

# Der Weg bis zur Recherche - Mehrsprachigkeit und Recherche

---



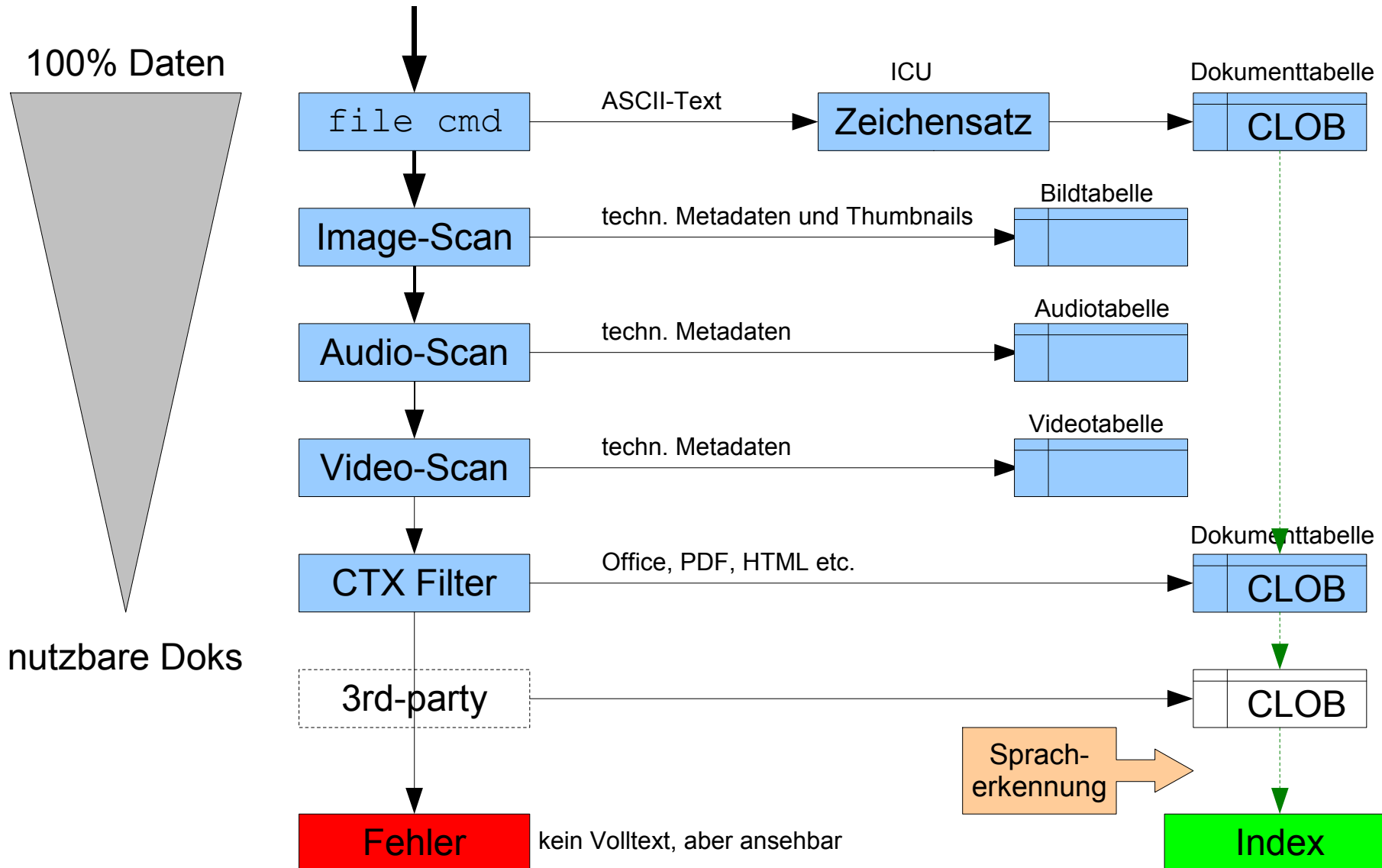
- Multilexer: verfügbare Sprachen reichen nicht (user lexer Markt?)
- Query generiert Such-Tokens für jedes Suchwort – aber leider nur in der session language (doesn't meet user expectations)
- 10g query templates:
  - Man kann sich Query Alternativen mit Sprachangabe generieren
  - Leider wird die Sprachangabe nicht voll unterstützt (kein stemming, fuzzy, ...)
  - Kann bei vielen Sprachen oder komplexen Queries unhandlich werden
- Anwender möchte in der Regel keine Sprache bei seiner Query eingeben oder einschränken

# Der Weg bis zur Recherche - die Dokumentaufbereitung

---

- viele Schritte notwendig, um alle Informationen zu sammeln
- sequentielle Abarbeitung verbraucht mehr Zeit als der Aufbau eines einfachen Volltextindex
  - dem Anwender wird der Zugang zu seinen Daten länger als notwendig verwehrt
  - Akzeptanz des Systems könnte leiden
- Massnahme: parallele Abarbeitung aller Prozessschritte durch eigenen Job-Manager (DBMS\_JOB Überbau)
  - Pending-Tabelle ähnlich wie DR\$PENDING
  - Inhalt: Dokument-IDs, Mandant, (Follow-Up-)Jobs
  - Viele kleine Programme statt ein Monolith, leicht erweiterbar, Drittprogramme integrierbar

# Der Weg bis zur Recherche - die Dokumentaufbereitung



# Wechsel 9.2 auf 10.2 - erste Auswertungen

---



- Testdatenbestand: 4560 Dokumente eines Mandanten (spezifischer Testdatenbestand für verschiedene Dateitypen)
- ca. 2000 mit `ordsys.ordimage` als Bild erkannt
- Filterdatenbestand: 2540 Dateien:
  - ca. 1800 HTML-Dateien
  - 65 PDF
  - 45 DOC, 10 RTF, 8 XLS, 4 MDB
  - 18 TXT, 15 XML
  - 18 MSG (Emails)
  - verschiedenste andere zumeist ältere Binär-Typen: DBase III/IV, Lotus 123, Paradox 5, MS Works, etc.
  - Verschiedene Text-Typen: PHP, CSS, INI, LOG, PostScript etc.

mehr..

# Wechsel 9.2 auf 10.2 - erste Auswertungen



## Oracle 9.2

- Fehlerhaft gefiltert: 222
  - HTML: 170
  - XML: 15
  - DOC: 3
  - MDB: 4
  - MSG: keins
- insgesamt 21 versch. Typen
- distinct \$I tokens: ~1,9Mio

## Oracle 10.2

- Fehlerhaft gefiltert: 149
  - HTML: keins
  - XML: keins
  - DOC: keins
  - MDB: 4
  - MSG: 18
- insgesamt 63 versch. Typen
- distinct \$I tokens: ~380.000

Häufigste  
Fehlermeldung bei  
beiden Versionen:

```
ORA-06502: PL/SQL: numeric or value error  
ORA-06512: at "CTXSYS.DRUE", line 180  
ORA-06512: at "CTXSYS.CTX_DOC", line 1142  
ORA-06502: PL/SQL: numeric or value error
```

mehr..

# Wechsel 9.2 auf 10.2 - erste Auswertungen



- Oracle 9.2 filtert mehr unbekannte Dokumenttypen, aber der gefilterte Text liefert binären Unsinn:

```
SQL> select text from ida_docs_0291 where doc_id = 5637666;
```

```
TEXT
```

```
-----  
<HTML><BODY>  
<p>FWS*jÃ°p*&quot;Ã² &nbsp; <br />  
*C*Ã|Ã¶*Ãµ*Ã¿*]â*wÃ®+â¢Ã¿*Ã¶]â¬*Ã±Ãâα**ÃÃÃ¬ÃÃÃ«3]&quot;Ã»nuÃÃ«***Ãâ°â{%Ã¾  
Ã·âÃ¾4Ã¬Ã·*0Ã£Ã¶Hf*}Ã- ij*0*F@4&nbsp; \Ã¥Ã,)rÃÃÃ Ã¬Ã½  
)Ã¬/Ã-Y]-Ã§&quot;&nbsp; xÃÃ'â¬Ã¹Lâ¬xdh4LâCÃÃ·Z^  
ZÃÃ¬âÃ**T*B*â@Ã½Ã¬Ã¥ÃÃ,99Ã¹Ep*Ã¶Ã*ÃHÃ¹â¢&amp;'M  
ÃâwÃZÃ«`âLÆkHÃ«Ã|Ã»&gt;*.ÃÃ¶FÃ£Ã§*kÃª*^&nbsp; Ã¬ââ&n  
bsp;*ÃfÃÃ»*Ã«Ãp}FrÃ±âÃ®&nbsp; <br />  
Ã¾ÃαÃ³ÃÃ|zeÃ§dEYC\*ÃBÃ¶QmKTHÃÃ£#(ÃÃ*MÃÃªgÃ©Ãµ3ÃT}}ZÃÃ&nbsp;  
<br />
```

- Oracle 10.2 springt mit Fehler raus (sinnvoller)
- insgesamt weniger Fehler bei 10.2, und deutlich bessere Performance

# Wünsche an Oracle

---



- Metadaten-Infos über offizielle Oracle-Schnittstelle (CTX\_DOC?)
- mehr Einflussnahme auf CTX\_DOC.SNIPPET
- Reguläre Ausdrücke im Text-Index (und Stopwords?)
- Bessere Unterstützung Mehrsprachigkeit in Query
- List-Partitioning
- Fortschritt eines Indexlaufs (V\$SESSION\_LONGOPS)
- Indexlauf abbrechen (DBMS\_ALERT?)
- Markup: Sprung zum ersten Treffer (**Metalink:** Bug-ID 902046)
- WORLD\_LEXER: erkannte Sprache ausgeben; Stemming etc.
- Knowledge-Base/Thesaurus für DEUTSCH mitliefern

# Q&A

---

Danke.

Noch Fragen?