Projektbeschreibung
TIFF-Korpus-Analyse

Inhalt

1 Zusammenfassung

Eines der am weitesten verbreiteten Formate, um hochwertige Bilddaten abzuspeichern, ist das TIF-Format. TIFF ist ein flexibles, anpassungsfähiges Dateiformat, das über die Jahre eine Vielzahl von Erweiterungen und Ergänzungen erfahren hat. Daneben bietet es die Möglichkeit, Metadaten in andern Standards (wie IPTC, EXIF oder ICC) einzubetten. Durch diese Flexibilität und Ausprägungen ist TIFF eigentlich als ein komplexes Dateiformat zu betrachten. TIFF ist zurzeit eine offene Spezifikation von Adobe, jedoch kein ISO-Standard. Diese Umstände, die aus archivischer Sicht nicht unbedenklich sind, haben die KOST 2014 bewogen, basierend auf der Baseline-TIFF-Spezifikation eine Empfehlung zu verfassen.

Ein weiterführendes Projekt des *Digital Humanities Lab* Basel und der KOST, zusammen mit der Universität Girona und der Firma Easy Innova (http://ti-a.org/) ist es nun, eine erweiterte Baseline-Spezifikation in eine *ISO Recommendation* zu überführen.

Damit eine solche Empfehlung nicht nur auf theoretischen Überlegungen beruht, sondern sich auf eine fundierte Analyse echter archivischer Daten stützen kann, haben es die KOST und das DHLab Basel unternommen, mehrere Millionen Dateien aus drei Archiven systematisch zu untersuchen. Parallel dazu wurden an diesem Korpus auch etliche bekannte und in der Archivwelt verbreitete Analysetools getestet.

2 Analyse im Archiv

2.1 Korpus

Die Staatsarchive Basel-Stadt und St.Gallen und das Schweizerische Bundesarchiv stellten für die Untersuchung ihre TIFF-Sammlungen von je etwa 12 TB zur Verfügung:

Basel-Stadt		
Typ	*Anzahl*	*Grösse*
2000	1'950	
2001	2'300	
2002	200	
2003	100	
2004	10'000	
2005-2015	750'000	
Total:	**764'550**	**12.4 TB**

Bundesarchiv		
Typ	*Anzahl*	*Grösse*
Archivtiffs	1.7 Mio	5-6 TB
Digitalisate	7300	460 GB
Digitalisate 3te	14 Mio	6 TB
Total:	**15 Mio**	**11-12 TB**

St. Gallen		
Typ	*Anzahl*	*Grösse*
Total:	**870'000**	**12.83 TB**

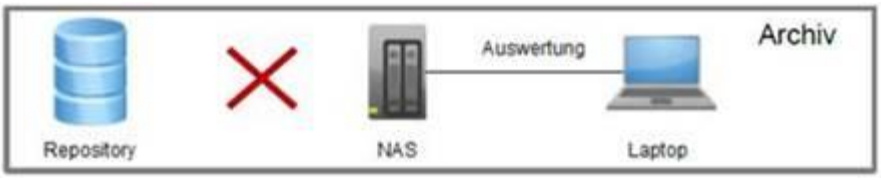
Die Bestände sind, was Alter, Grösse und Ursprung betrifft, in allen Archiven sehr heterogen.

2.2 Korpus zusammentragen

Eine Schwierigkeit im Projekt war, dass die zu untersuchenden Korpora die beteiligten Archive nicht verlassen durfte, weil möglicherweise die Daten noch einer Schutzfrist unterstehen oder urheberrechtlich geschützt sind. Auch sollen über Dateinamen oder Pfadnamen keine Rückschlüsse auf die Archivbestände gezogen werden können.

Wir haben dazu in einem ersten Schritt die TIFF-Dateien aus dem jeweiligen Archivsystem auf USB- oder NAS-Platten kopiert und diese anschliessend für die weitere Untersuchung vom Netzwerk des jeweiligen Archivs abgehängt.

Kopieren der Daten im Archiv



Untersuchen der Daten mit den Analyseprogrammen

Das Kopieren nahm wegen der Datenmenge und wegen organisatorischer Herausforderungen (Rechner, NAS, Disk) etwa 3 Monate in Anspruch.

2.3 Analyse

Für die Analyse der TIFF-Dateien wurden vom Projektteam folgende Programme ausgewählt:

- [MD5-Berechnung
  MD5 (integriert in den Loop)
  Das Berechnen des MD5-Schlüssels gehört nicht zu den Analysemodulen, wird aber vor der Analyse durchgeführt.]
- Formaterkennung
  file
  http://gnuwin32.sourceforge.net/packages/file.htm
  Mit der Formaterkennung werden falsch gelabelte Dateien erkannt.

- Validierung
  JHOVE
  http://jhove.openpreservation.org/
  Die JHOVE-Validierung ermittelt die grundlegende Struktur der TIFF-Datei. Wichtig sind hier *Status* und *InfoMessage*.
- Validierung
  DPF-Manager
  http://www.preforma-project.eu/dpf-manager.html
  Der DPF-Manager ist eine Alternative zu JOHVE aus dem PREFORMA-Projekt.
- Validierung
  checkit_tiff: *a conformance checker for baseline TIFFs*
  https://github.com/SLUB-digitalpreservation/checkit_tiff
  checkit_tiff wurde von der Sächsischen Landesbibliothek – Staats- und Universitätsbibliothek Dresden entwickelt.
- TIFF-Tag-Extraktion
  tiffhist
  http://dhlab.unibas.ch/
  Das vom DHLab entwickelte C++-Programm extrahiert alle TIFF-Tags in eine CSV-Tabelle (TIFF-Tag, Datentyp und Wert)
- EXIF-Extraktion
  ExifTool
  http://owl.phy.queensu.ca/~phil/exiftool/
  Eingebettete EXIF- und XMP-Metadaten werden extrahiert.
- Thumbnail-Generierung
  ImageMagick
  http://www.imagemagick.org/
  Für jede Datei wird mit *ImageMagick* ein sehr kleines Thumbnail generiert. In diesem Schritt wird die Payload oder Bitmap der TIFF-Datei untersucht. Eine erfolgreiche Konvertierung belegt die korrekte Implementierung von Komprimierung und Farbraum.

2.4 Auswertung

Die Analysemodule wurden ohne direkte Auswertung der Log- oder Systemausgabe ausgeführt. Die Log- oder Systemausgabe zu jedem Analyseschritt wurden für die spätere Auswertung festgehalten. Um der Anforderung der vollständigen Anonymisierung gerecht zu werden, wurden die Logdateien beim Schreiben gefiltert und Pfad und Dateinamen entfernt. Ein Abbruch in einem Analyseschritt darf die nächsten Tools nicht beeinflussen. Die eigentliche Auswertung erfolgt anschliessend vollständig *offline*, entweder im Archiv oder ausgelagert.

Die Vorteile dieses Vorgehens sind, dass verschiedene Auswertungen auch zeitlich versetzt möglich sind und dass die Fragestellungen und Auswertungsmethoden während der Arbeit noch verändert werden können.

Für die Auswertung stehen nach der Analyse insgesamt etwa 35 GB Log-Informationen zur Verfügung.

3 Analyseumgebung

Damit die Anforderungen an den Umgang mit grossen Datenmenge, langen Programmlaufzeiten und sicherer Anonymisierung erfüllt werden konnten, wurde beschlossen, die Programmausführung wie auch die Logverwaltung mit einer Datenbank und einem speziellen Analyse-Überwachungsprogramm zu realisieren. Vorbedingung war zudem ein Einsatz im Linux- wie im Windows-Umfeld.

Als Datenbank wurde SQlite und als Programmiersprache für das Überwachungsprogramm Golang gewählt. Alle Programme, Datenbankmodelle, Scripts und SQL-Abfragen sind auf GitHub verfügbar unter https://github.com/KOST-CECO/TiffAnalyseProject.

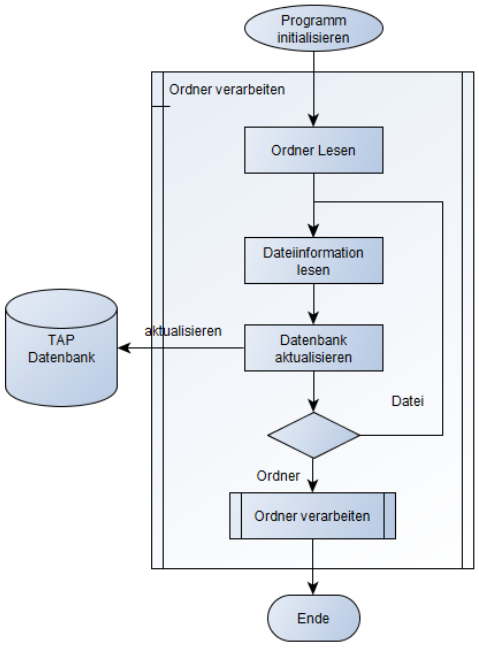
SQlite (https://www.sqlite.org/) hat den Vorteil, dass weder Server noch Administration notwendig sind. Golang (https://golang.org/) ist eine kompilierte Sprache und auf allen Plattformen verfügbar, hat ein API zu SQLite und ist einfacher als C/C++.

3.1 Analyse-Programm

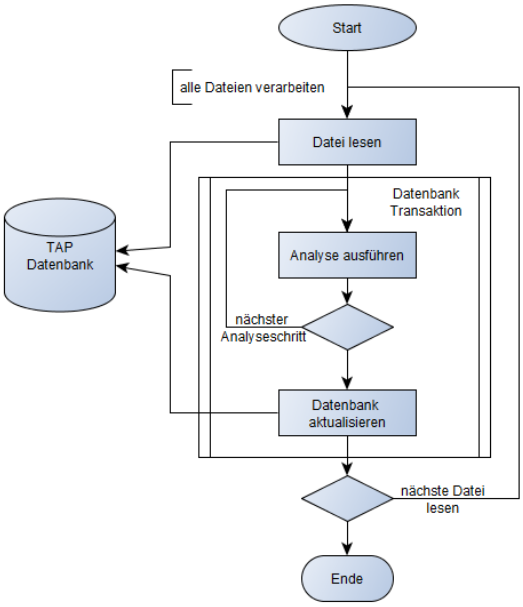
Das Analyse-Loop-Programm liest alle TIFF-Dateien des Korpus vom NAS und führt mit der jeweils gelesenen Datei mehrere Analyseschritte aus durch Aufrufen von externen Programmen. Der Loop-Prozess ist zweiteilig und besteht aus der *Initialisierung* der Prozessdatenbank und der *eigentlichen Analyse*.

Der Initialisierungsschritt erstellt die Datenbank und schreibt für jede TIFF-Datei einen Eintrag mit dem Pfad und Dateinamen als Schlüssel. Die Initialisierung kann mehrfach aufgerufen werden und fügt so neue Verzeichnispfade zur Datenbank hinzu.

Damit für eine spätere Auswertung ausserhalb der Archive problemlos gearbeitet werden kann, werden Dateinamen und Dateipfad, welche allenfalls Rückschlüsse auf den Inhalt der Dateien erlauben, in einer separaten Tabelle *(namefile)* gehalten.



„Initial Loop“ liest sämtliche TIFF-Dateien



„Process Loop“ führt Analyseschritte aus

Im Analysefall werden die Dateieinträge in der Datenbank abgearbeitet und die Analysetools im Kommandozeilenmodus aufgerufen. Durch die Verwendung einer Datenbank ist jederzeit ein Abbrechen und Neustarten der Analyse möglich.

- Dem Analysetool im Kommandozeilenmodus werden der Dateipfad und der Pfad zur Logdatei übergeben.
- Kann das Analysetool kein Logfile im Append Modus öffnen, kann der Log-Output vom Loop-Programm entweder an die Logdatei angehängt oder in die Datenbank geschrieben werden.
- Der aktuelle Offset der Log-Datei wird in der Datenbank gespeichert.
- Der Exit Value des Analysetools wird in der Datenbank festgehalten.
- Es kann festgelegt werden, ob der Systemoutput des Analysetools in eine spezielle Ausgabedatei geschrieben oder in der Datenbank gespeichert werden soll.
- Eine Logrotation verhindert allzu grosse Logdateien.

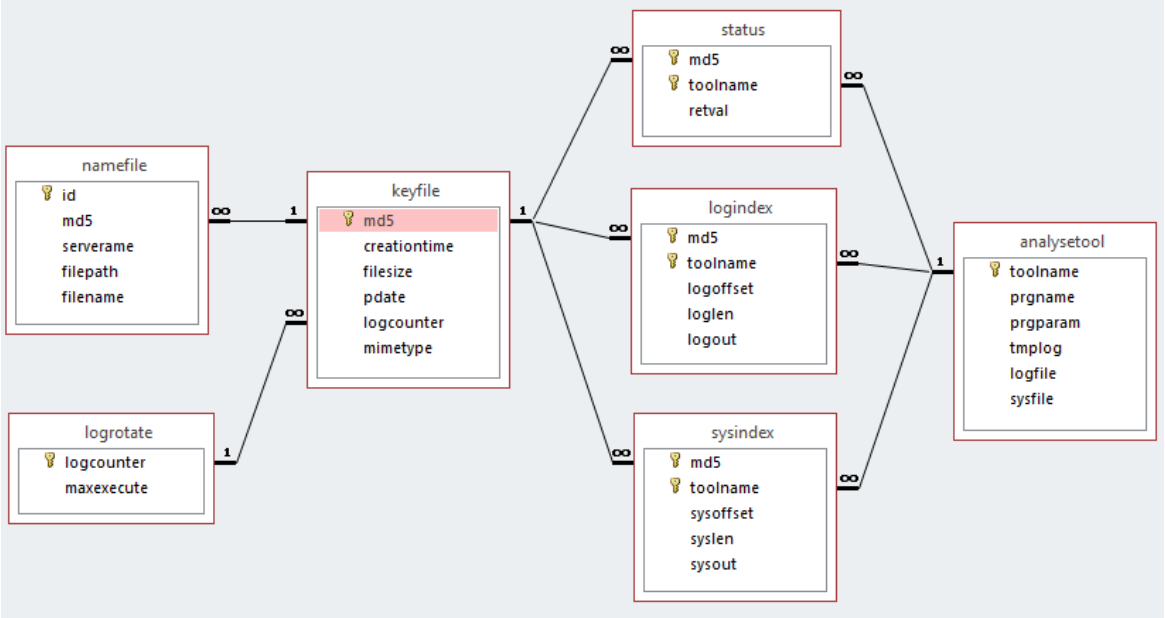
3.2 Datenmodell

Die Tabellen *keyfile* und *namefile* enthalten den primären Verzeichnisscan, also die Namen aller Dateien mit Dateigrösse und *Creation Time*, sowie diese aus dem Lesen der Verzeichnisstrukturen erstellt werden können.

Zum Ausführen der Analysemodule werden die notwendigen Informationen aus der Tabelle *analysetool* ausgelesen, d.i. Programmname und Pfad, Logdatei, Datei bzw. BLOB für den Systemoutput.

Die Tabelle *status* hält den Exit Status des Analyseprogramms fest.

Die Tabellen *logindex* und *outindex* speichern entweder den Dateinamen und den Offset in die jeweilige Logdatei oder den gesamten Output der eben analysierten Datei in ein entsprechendes LOB.



Das Datenmodell in grafischer Darstellung

tablename	name	type	description
analysetool	toolname	CHARACTER VARYING(30)	Name des registrierten Analyseprograms in Kurzform
	prgname	CHARACTER VARYING(255)	Pfad und Dateiname zum Analyseprograms
	prgparam	CHARACTER VARYING(255)	Parameter des Analyseprogramms mit Wildcards %file% und %log%
	tmplog	CHARACTER VARYING(255)	Temporäre Logdatei: ersetzt Wildcards %log% beim Ausführen des Analyseprogramms, Fehlen meint keine Logdatei schreiben
	logfile	CHARACTER VARYING(255)	Pfad und Dateiname der mit diesem Analyseprogramms verbunden Logdatei: Ist kein Logfile definiert, wird in LOB "logout" gespeichert
	sysfile	CHARACTER VARYING(255)	Pfad und Dateiname der mit diesem Analyseprogramms verbunden Ausgabedatei: Ist kein Sysfile definiert, wird in LOB "sysout" gespeichert

keyfile	md5	CHARACTER VARYING(32)	MD5-Hashwert und Referenz zum „namefile“
	creationtime	TIMESTAMP(0)	Entstehungszeitpunkt der Datei laut Dateisystem
	filesize	LONG	Dateigrösse in Byte
	pdate	TIMESTAMP(0)	Zeitpunkt und Flag für den Abschluss der gesamten Analyse
	logcounter	INTEGER	Zähler für "logfile" bzw. "sysfile" beginnend mit Eins
	mimetype	INTEGER	Internet Media Type, auch MIME-Type aufgrund der Magic Number

logindex	md5	CHARACTER VARYING(255)	MD5-Schlüssel der TIFF-Datei
	toolname	CHARACTER VARYING(255)	Kurzname des Tools
	logoffset	INTEGER	Offset in die Ausgabedatei analysetool.logfile
	loglen	INTEGER	Länge des Logausgabe
	logout	CHARACTER LARGE OBJECT	Vollständige Logausgabe des Analysetools oder Name der Logdatei

logrotate	logcounter	INTEGER	Zähler für "logfile" bzw. "sysfile" beginnend mit eins
	maxexecute	INTEGER	Maximale Verarbeitungsschritte pro "logfile" bzw. "sysfile"

namefile	id	INTEGER	Referenz zu "keyfile"
	md5	CHARACTER VARYING(32)	MD5-Hashwert
	serverame	CHARACTER VARYING(255)	Name des NAS-Servers oder des zugeordneten Laufwerkbuchstabens
	filepath	CHARACTER VARYING(255)	Dateipfad
	filename	CHARACTER VARYING(255)	Dateiname mit Dateiextension

status	md5	CHARACTER VARYING(255)	MD5-Schlüssel der TIFF-Datei
	toolname	CHARACTER VARYING(255)	Name des registrierten Analyseprogramms in Kurzform
	retval	CHARACTER VARYING(255)	Rückgabewert des Tools (Exit Status 0 = erfolgreicher Abschluss) http://www.hiteksoftware.com/knowledge/articles/049.htm

sysindex	md5	CHARACTER VARYING(255)	MD5-Schlüssel der TIFF-Datei
	toolname	CHARACTER VARYING(255)	Kurzname des Tools
	sysoffset	INTEGER	Offset in die Ausgabedatei analysetool.sysfile
	sylen	INTEGER	Länge des Konsolenausgabe
	sysout	CHARACTER LARGE OBJECT	Vollständige SystemOut Ausgabe des Analysetools: stderr & stdout oder Name der Logdatei

3.3 Analyseprogramm installieren

Vorbedingung: `sqlite` (www.sqlite.org) und alle Analyseprogramme sind installiert.

Folgende Schritte müssen ausgeführt werden:

1. Script Package in einem Ordner mit entsprechenden Rechten (create/execute) installieren.
2. In Script `load_ToolList.sql` die entsprechenden Analyseprogramme und Logdateien eintragen (siehe Beispiele `load_ToolList.sql`)
3. Mit Script `create_TAP.bat` eine neue Datenbank anlegen
   `usage: create_TAP.bat path/dbname.db`

4. Mit `iniloop.exe` alle Dateien einlesen und in die Datenbank schreiben.
   `iniloop.exe` kann mit mehreren Startordnern mehrfach aufgerufen werden.
   `usage: iniloop.exe folder database`
5. Mit `runloop.exe` die Verarbeitung starten.
   `usage: runloop.exe [options] database`

`runloop.exe` kann mit Ctrl-C unterbrochen oder beendet werden.

Es empfiehlt sich, zu Beginn `runloop.exe` nach kurzer Zeit abzubrechen und zu kontrollieren, ob alle aufgerufenen Analysetools in gewünschter Art und Weise arbeiten und die entsprechenden Logdateien geschrieben werden. Die entstandenen Testergebnisse können mit `clean_TAP.bat` wieder aus der Datenbank entfernt werden, ohne dass die bereits gelesenen Dateiinformationen verloren gehen. Achtung, Logdateien müssen manuell gelöscht werden.

Zwischendurch, wenn auch nicht gleichzeitig, ist es auch möglich, mit `iniloop.exe` weitere Dateien in die Datenbank einzufügen.

4 Einschränkungen bei der Analyse

Nach den ersten Tests hat sich schnell gezeigt, dass die verwendeten Tools ganz unterschiedliche Rechenzeiten pro TIFF-Datei erfordern. Hier die Rechenzeiten pro Tool über 1000 Dateien unterschiedlicher Grösse (mittlere Grösse ~5.5 MB), indexiert gegenüber tiffhist:

Tool	Sec	Prozent	Faktor
tiffhist	257	100%	1.0
dpf-manager	257	100%	1.0
file	266	104%	1.0
exiv2	267	104%	1.0
exif	335	130%	1.3
checkit_tiff	503	196%	2.0
jhove	697	271%	2.7
ImageMagick	3424	1332%	13.3
Total	6006	2337%	23.4

Dieser Umstand hat uns anschliessend bewogen, *ImageMagick* nur über einem sehr kleinen Teilbestand und *JHOVE* nur etwa über der Hälfte der Dateien auszuführen.

Das aus unserer Sicht wichtigste Tool zur Extraktion von TIFF-Tags, *tiffhist* vom *Digital Humanities Lab* Basel, wurde hingegen auf allen Dateien ausgeführt.

5 Auswertung

Die Auswertung der Analyse kann und soll nicht durch die KOST-Geschäftsstelle oder das *Digital Humanities Lab* Basel alleine geleistet werden. Deshalb publizieren und erläutern wir die Resultate auf unserer Website (http://kost-ceco.ch/ftp_space/TIFF-Analyse/) und laden alle Interessierten ein, diese Daten für ihre eigenen Forschungen zu benützen. Besonders angesprochen sind dabei die Hochschulen und Fachhochschulen.

Die Webseite enthält die Analysedatenbank als *SQL Loader Script* tap.sql.gz (md5: 33b406a083472fb3853c1d07169bd640) und die Logdateien in einem TAR-File log.tgz (md5: d44b169f1d8048637c5502be646f8a85). Beide Dateien sind gzip-komprimiert.

Anschliessend einige Beispiele für eine mögliche Auswertung.

5.1 Verteilung TIFF-Komprimierung

In diesem Beispiel werden auf Grund der Tag-Informationen, welche *tiffhist* in die Logdatei schreibt, die Verteilung und Werte des Compression Tags 259 untersucht. Das Tag ist in der TIFF-Spezifikation folgendermassen erläutert:

> **Compression**
> Data can be stored either compressed or uncompressed.
> Tag = 259 (103.H)
> Type = SHORT
> Value = 1 -10, values > 32766 (proprietary values)

1	No compression
2	CCITT 1D
3	Group 3 Fax
4	Group 4 Fax
5	LZW
6	JPEG TIFF/6-.0 marked as deprecated
7	JPEG TIFF TechNote2 1995
8	Adobe Deflate
9	JBIG bw
10	JBIG color
32773	PackBits

Die Verteilung im Korpus nach Aufbereitung mit Microsoft Excel:

Compression	Basel-Stadt	Bundesarchiv	Bundesarchiv extern	St. Gallen	Total	Prozent
none	732'696	560'956	217'843	550'675	2'062'170	**52%**
CCIT 1D	0	564	0	0	564	**0%**
Fax Group 3	0	20'041	0	0	20'041	**1%**
Fax Group 4	22'745	602'571	1'207'376	317	1'833'009	**46%**
LZW	31	19'534	0	15'651	35'216	**1%**
old JPEG	0	0	0	0	0	**0%**
JPEG	0	12'095	0	15'427	27'522	**1%**
Adobe Deflate	0	1'593	0	0	1'593	**0%**
JBIG bw	0	0	0	0	0	**0%**
JBIG color	0	0	0	0	0	**0%**
Pack Bits	0	0	0	0	0	**0%**
other	0	0	0	0	0	**0%**
Summe	755'472	1'217'354	1'425'219	582'070	3'980'115	**100%**

Ein Windows CMD oder Linux Shell Script erzeugt eine CSV-Tabelle (hier die Windowsvariante):

```
@ECHO off
SETLOCAL

SET UNIX_HOME=C:\Tools\PCUnixUtils
SET PERL_HOME=C:\Tools\Perl
SET PATH=%UNIX_HOME%;%PERL_HOME%\bin;%PATH%

REM -------------------------------------------------------------------------
IF [%1]==[] (
     ECHO log path is missing
     ECHO usage: %0 path
     EXIT /B
)

IF NOT EXIST %1 (
     ECHO invalid path: %1
     ECHO usage: %0 path
     EXIT /B
)

SET OUT=c:\tmp\dummy.log
rm -f %OUT%

FOR /F "tokens=*" %%G IN ('dir /b %1\tiffhist*.log') DO (
     ECHO %%G
     cat %1\%%G >> %OUT%
)
ECHO.
ECHO %1
ECHO Total Compression
grep "259$Compression" %OUT% | wc -l
ECHO.

ECHO Compression:1
grep "259$Compression:1" %OUT% | wc -l
ECHO Compression:2
grep "259$Compression:2" %OUT% | wc -l
ECHO Compression:3
grep "259$Compression:3" %OUT% | wc -l
ECHO Compression:4
grep "259$Compression:4" %OUT% | wc -l
ECHO Compression:5
grep "259$Compression:5" %OUT% | wc -l
ECHO Compression:6
grep "259$Compression:6" %OUT% | wc -l
ECHO Compression:7
grep "259$Compression:7" %OUT% | wc -l
ECHO Compression:8
grep "259$Compression:8" %OUT% | wc -l
ECHO Compression:9
grep "259$Compression:9" %OUT% | wc -l

DEL %OUT%
```

5.2 Auswertung JHOVE

Eine einfache Abfrage auf der Datenbank zeigt, dass JHOVE einerseits mit externem Logfile, aber auch mit Speichern der Systemausgabe in der Datenbank ausgeführt worden ist.

```
SELECT DISTINCT toolname FROM logindex;
     dpf-manager
     jhove
     tiffhist

SELECT DISTINCT logout FROM logindex WHERE toolname = "jhove";
     ./log/01_jhove_out.1.log
     ./log/01_jhove_out.2.log
     ./log/01_jhove_out.3.log

SELECT DISTINCT toolname FROM sysindex;
     jhove
     dpf-manager
     file
     tiffhist
     checkit_tiff
     exif
     exiv2
     ImageMagick
```

Hier ein Auszug aus dem Ergebnis mit der Auswahl „PhotoshopProperties“:

```
SELECT DISTINCT sysout FROM sysindex WHERE toolname = "jhove" AND sysout LIKE
"%PhotoshopProperties%";

Jhove (Rel. 1.6, 2011-01-04)
 Date: 2016-07-28 14:39:49 MESZ
 RepresentationInformation: /***********/*********/*******/*********.***
  ReportingModule: TIFF-hul, Rel. 1.5 (2007-10-02)
  LastModified: 2000-07-27 11:51:14 MESZ
  Size: 6738554
  Format: TIFF
  Version: 5.0
  Status: Well-Formed and valid
  SignatureMatches:
   TIFF-hul
  MIMEtype: image/tiff
  Profile: Baseline grayscale (Class G), TIFF/IT-MP (ISO 12639:1998), DLF Benchmark
for Faithful Digital Reproductions of Monographs and Serials: grayscale and white
  TIFFMetadata:
   ByteOrder: little-endian
   IFDs:
    Number: 1
    IFD:
     Offset: 8
     Type: TIFF
     Entries:
      NisoImageMetadata:
       ByteOrder: little_endian
       CompressionScheme: uncompressed
       ImageWidth: 3093
       ImageHeight: 2177
       ColorSpace: black is zero
       Orientation: normal
       SamplingFrequencyUnit: inch
       XSamplingFrequency: 460
       YSamplingFrequency: 460
       BitsPerSample: 8
       BitsPerSampleUnit: integer
       SamplesPerPixel: 1
```

```
      ImageDescription: Sammlung BALAIR%0DOriginal
      NewSubfileType: 0
      SampleFormat: 1
      MinSampleValue: 0
      MaxSampleValue: 255
      Threshholding: 1
      PhotoshopProperties: 56, 66, 73, 77, 4, 4, 0, 0, 0, 0, 0, 116, 28, 2, 0, 0, 2,
0, 2, 28, 2, 120, 0, 24, 83, 97, 109, 109, 108, 117, 110, 103, 32, 66, 65, 76, 65,
73, 82, 13, 79, 114, 105, 103, 105, 110, 97, 108, 28, 2, 105, 0, 12, 66, 65, 76, 65,…
      TIFFITProperties:
       BackgroundColorIndicator: background not defined
       ImageColorIndicator: image not defined
       TransparencyIndicator: no transparency
       PixelIntensityRange: 0, 255
       RasterPadding: 1 byte
       BitsPerRunLength: 8
       BitsPerExtendedRunLength: 16
      TIFFEPProperties:
       IPTCNAA: 540, 469893122, 402683906, 1835884883, 1735292268, 1279345184,
223496513, 1734963791, 1818324585, 6881820, 1279345164, 542263617, 959787808,
671226931, 1666387968, 980315745, 1098142496, 790647618, 35409217, 839385143,
808464432, 473379383, 402682882, 1633776723, 1918989172, 1986619491, 1935753760,
1395485797, 1952735604
```

5.3 Vergleich zweier Tools (exif und exiv2)

Die Auswertung der Ausgaben verschiedener Tools ist bei der Speicherung der Logausgabe in der Datenbank relativ einfach. Das Beispiel vergleicht die Ausgabe von *exiftool* und *exiv2* zur jeweils gleichen Datei.

```
.output exiftool&exiv2.html
.mode asci
SELECT "<!DOCTYPE html><HTML><head><style> table { font-family: arial, sans-serif;
    border-collapse: collapse; width: 100%; } td, th { border: 1px solid #dddddd;
    text-align: left; padding: 8px; } tr:nth-child(even) { background-color:
    #dddddd; }</style></head>
    <BODY><PRE><TABLE>";

      .mode html
      SELECT
      --  sys1.md5,
          sys1.toolname,
          sys1.sysout,
      --  sys2.md5,
          sys2.toolname,
          sys2.sysout
      FROM
          (SELECT md5, toolname, sysout from sysindex WHERE toolname = "exif") sys1
      INNER JOIN
          (SELECT md5, toolname, sysout from sysindex WHERE toolname = "exiv2") sys2
      ON
          sys1.md5 = sys2.md5;

      .mode ascii
      SELECT "</TABLE></PRE></BODY></HTML>";
      .exit
```

Die Ausgabedatei, eine HTML Tabelle, mit einem Browser betrachtet: