CAD-Dokumentation zu GIS mit SpatiaLite migrieren

Christoph Rinne

04. November 2022

Inhaltsverzeichnis

Vorwort

Ziel ist die Überführung von Ausgrabungsplänen aus CAD-Dateien in ein GIS. Ausgang ist die Retrodigitalisierung (2D) einer Papierdokumentation einer über vier Jahre erfolgten Ausgrabung des Kollektivgrabes Odagsen 1, Stadt Einbeck, Ldkr. Northeim. Hierbei geht es nicht um einen schönen, interaktiven Plan in einem GIS am Ende, sondern um die Nachnutzung möglichst vieler Daten für eine räumliche Statistik.

Dieser erste Fall besteht aus vielen DWG-Dateien mit spezifischen, aber sehr weit verbreitete Problemen, u.a. zur falsch deklarierten Einheit in AutoCAD. Die zweite Variante der 2D-Retrodigitalisierung besteht aus einer sehr umfangreichen Datei einschließlich zahlreicher Blöcke mit Attributen, u.a. für Funde und Niv-Werte.

Der dritte Fall ist eine AutoCAD-Datei mit 3D-Objekten, die mit eine Tachymeter und der Software TachyCAD direkt in AutoCAD dokumentiert wurde.

Anmerkungen

- Menüpfade oder Abfolgen von Fenstern werden mit schlichten Pfeilen dargestellt: “Datei > Speichern”.
- Tastaturkürzel, die ich gerne Nutze, stehen in Spitzklammern je Taste: <strg> + <c>.
- Schalter auf Formularen werden in [] gesetzt: [OK]
- Zur Darstellung von Befehlen im Text nutze ich die in Markdown übliche Darstellung von Code oder eben Anweisungen an den Computer: `anweisung`.
- SQL-Anweisungen sind nicht in Großbuchstaben gesetzt, die farbliche Gestaltung macht dies überflüssig. Eine Ausnahme bilden die verwendeten Funktion bei denen der *CamelCase* für die bessere Lesbarkeit beibehalten wird. Auch wird der Anfang einer Anweisungen jeweils groß geschrieben, um aufeinander folgende Anweisungen etwas besser zu trennen. Im vorliegenden Fall wurden Leerzeichen in Objektnamen vermieden, dadurch müssen Tabellen und Feldnamen nicht in “ ” stehen.
- Der Text enthält viele Links die auf Papier nicht funktionieren. Sparen Sie bitte Papier und verzichten Sie auf den Ausdruck.

1 Einführung

1.1 Verwendete Software & Informationen

- OS Windows 10
- QGIS 3.22.4-Białowieża Quelle: [https://qgis.org]
- SpatiaLite SpatiaLite GUI 2.1.0 beta1, SpatiaLite 5.0.0, SQLite 3.33.0, Quelle [http://www.gaia-gis.it]
- AutoCAD 2010, Quelle für aktuelle *kostenlose* Schulversionen: [https://www.autodesk.de/education/edu-software/overview]
- SpatiaLite Cookbook html [http://www.gaia-gis.it/gaia-sins/spatialite-cookbook/index.html]
- SpatiaLite Funktionen [http://www.gaia-gis.it/gaia-sins/spatialite-sql-5.0.0.html]

AutoCAD ist eine sehr komplexe Software und Ausgrabungen können eine komplexe Struktur annehem, die es zu dokumentieren gilt. Erwarten Sie nicht, dass die notwendige Kompetenz beim Erstellen der digitalen Daten stets vorhanden war, auch der Autor (Chr. Rinne) ist hier nur Autodidakt.

Rechnen Sie mit Fehlern im originalen Datenbestand und einer ggf. nicht optimalen Struktur oder erwarten Sie nicht die von Ihnen bevorzugte Struktur. Korrektur von Fehler und Anpassungen der Struktur erfolgen sicher am besten im originalen Arbeitsumfeld, also CAD.

Neben AutoCAD gibt es teils kostengünstigere Alternativen, u.a.:

- BricsCAD [https://www.bricsys.com]
- MegaCAD [https://www.megacad.de/]

1.2 Layer als getrennte DXF-Dateien

Alle GIS trennen nach Punkt, Linie und Polygon. Die in AutoCAD sehr wichtige inhaltliche Trennung in diverse Layer wird in einem GIS beim Import in der Regel nur in einer Attributspalte wiedergegeben. Es mag für die Archivierung als auch die Nachnutzung von CAD-Dateien sinnvoll sein, jeden Layer in eine eigene DXF-Datei zu exportieren. So werden kleinere Einheiten geschaffen und eventuell vorhandene Fehler ggf. eingegrenzt.

Statt Handarbeit empfiehlt sich auch hierfür eine Programmierung, bei AutoCAD in LISP:

```
; Create list of layer names.
; The (not lyr) excludes the current lyr so the next layer is selected
(setq lyrlst nil)
(while (setq lyr (tblnext "layer" (not lyr)))
  ;the keyword for more informations related to the next is "dotted pairs"
  (setq lyrlst (cons (cdr (assoc 2 lyr)) lyrlst))
)

; Select all elements of the lyer and save it to dxf
(foreach lyr lyrlst
  (progn ; allows more than one command in the for loop
    ;switch layer on, thaw and unlock to see and export something when loaded
      (command "-layer" "_on" lyr "_thaw" lyr "_unlock" lyr "")
      ;select everything from the selected layer
    (setq ss (ssget "X" (list (cons 8 lyr))))
    (if (> (sslength ss) 0)
      (progn
        ;concatenate current file path, filename, "_", layername and extension
          (setq pathfile (strcat
                (getvar "dwgprefix")
                (substr (getvar "dwgname") 1 (- (strlen (getvar "dwgname")) 4))
                "_" lyr
                ".dxf"
                );end strcat
          );end setq
          ;export selected features as dxf
          (command "-wblock" pathfile "" "" "0,0" ss "")
      );end progn
    );end if
  );end progn
);end for
```

Das ist eine einfache und ausreichend kommentierte Funktion, also einfach per *copy 'n paste* in den LISP-Editor kopieren und dort ausführen. Den LISP-Editor finden Sie in AutoCAD unter "Extras > AutoLISP > Visual LISP Editor". Mit <Ctrl> + <n> öffnen Sie eine neue Datei, fügen den Code dort ein und wählen dann das Icon mit dem roten Pfeil nach unten ("Aktives Bearbeitungsfenster laden"). Der Export erfolgt in das Verzeichnis der aktiven Zeichnung.

Für den Export wird der Befehl `wblock` verwendet, da dieser die selektierten Objekte exportiert. Damit lässt sich leider keine Entscheidung über die DXF-Version treffen. Eine Möglichkeit ist das pauschale Ändern des Speicherformates in den Optionen: "Extras > Optionen" Register "Öffnen und Speichern". Der alternative Befehl `dxfout` ist keine echte Alternative, da dieser stets den gesammten Dateinhalt exportiert.

1.3 Originaldaten

1.3.1 Ausgrabung

Die Daten stammen von der Ausgrabung und Auswertung des spätneolithischen Kollektivgrabes Ogasen I, Stadt Einbeck, Ldkr. Northeim in Niedersachsen. Die Ausgrabung erfolgte in vier Kampagnen von 1981 bis 1984 als Forschungs- und Lehrgrabung des Institutes für Ur- und Frühgeschichte der Georg-

August-Universität in Niedersachsen (Rinne (2003); Heege und Heege (1989)). In diesen Kampagnen wurden zahlreiche Schnitte und eine wechselnde Anzahl von Plana angelegt als auch die dazwischen ursprünglich belassenen Profilstege sukzsessive abgebaut und auf insgesamt 154, meist einzelnen und neu gerichteten Din A3-Blättern im M 1:20 dokumentiert. Zu den jeweiligen Planblättern wurden Überlieger auf Transparentpapier mit Nivellierwerte und weiteren Angaben angefertigt. Die Einmessung erfolgte mit Theodolit, Nivelliergerät und Maßband.

1.3.2 Digitalisierung in AutoCAD

Die die Digitalisierung der Planzeichnungen erfolgte im Mai und Juni 1997 in AutoCAD Ver. 12 (DOS) und Ver. 13 (Windows 3.1) auf einem Din-A3-Grafiktablett und mit Referenzierung anhand der Koordinatenangaben auf den Blättern. Jede Datei erhielt eine stringent vergeben Namen ODS(chnitt) P(lanum) , z.B. ODS1P102. Für jedes Blatt wurde die Planumsangabe der Zeichnung, die Planumsangabe mit Bezug auf die Angabe der Ausgräberin, die Bearbeitungszeit, die mittlere Angabe der Nivellierwerte zum Fixpunkt der Oberfläche erfasst. Ergänzt wurden nachträglich die gängigen Metadaten der resultierenden Dateien.

datei	pl_lokal	pl_heege	zeit_h	niv_oberfl	byte	datum	uhrzeit	nr
ODS1P101	1	1	0.9	50	47219	29.04.97	11:37	1
ODS1P102	1	1	3.1	38	195786	29.04.97	9:54	2
ODS1P103	1	1	1.6	34	142651	29.04.97	10:06	3
ODS1P106	1	1	0.4	38	11904	27.04.97	7:19	6
ODS1P107	1	1	2.0	40	64278	29.04.97	11:53	7
ODS1P204	1	1	1.6	48	204959	29.04.97	11:36	4
ODS1P205	1	1	3.8	50	484742	29.04.97	11:46	5
ODS1P308	1	2	0.5	58	320690	29.04.97	11:59	8
ODS1P309	1	2	2.8	60	231455	29.04.97	12:06	9

Die Dateien sind einfach strukturiert. Folgende Layer wurden für Informationseinheiten verwendet: BEFUND, BEFUND_UNSICHER, BEFUNDSCHRAFF, FEUER, GRABUNGSGRENZE, GRENZE, KNOCHEN, KNOCHENSCHRAFF, STEINE, STEINESCHRAFF, TPROFIL. Alle Linien wurden als 2D-Polygone digitalisiert, allerdings wurden die Polygone nicht geschlossen, sondern bei der oft dichten Lage von Steinen und Knochen nur sauber an den gemeinsamen Punkten gefangen.

Als Störungen klassifizierte Befunde haben eine horizontale Linienschraffur, Sandsteine erhielten eine Punktschraffur und gebrannte Steine eine diagonale Schraffur auf dem Layer “FEUER”, um diese Information zu vermitteln. Knochen wurden ausschließlich für den optischen Effekt stets schraffiert. Die Schraffuren wurden nicht je Objekt, sondern meist für zahlreiche Objekte angelegt, wodurch diese Schraffuren als ein Objekt über mehrere Steine oder Knochen laufen und der Mittelpunkt dieser Schraffur räumlich nicht mit einem Objekt zusammenhängt. Dies trifft vor allem auf Knochen zu, bei den eher singulären Sandsteinen oder im Verbund gebrannten Steinen ist ein räumlicher Kontext eher gegeben.

Symbole für Holzkohle, Rotlehm und verbrannte Knochen wurden als grafische Blöcke mit den Namen HK, RL, LB eingefügt. Diese können mit dem jeweiligen Datei-, Layer- und Blocknamen als auch den Koordinaten aus allen Zeichnungen eines Ordners in eine Tabelle exportiert werden (s.u. Vorbereitung in AutoCAD).

2 Vorbereitung in AutoCAD

2.1 Export in DXF

Für den Export aller DWG-Datei in DXF kann ein Script geschrieben und als Startoption an AutoCAD innerhalb eines Batch-Scriptes übergeben werden. Die Batch-Datei zum Starten von AutoCAD wird im Ordner der DWG-Dateien aufgerufen, wodurch das Arbeitsvezeichnis hier liegt und die Pfadangaben im Skript (*.scr) entfallen können.

```
REM Command to start ACAD with the script to convert all DWG files to DXF
"c:\Program Files\Autodesk\AutoCAD 2014\acad.exe" /b od_convert_dwg2dxf.scr
```

Das Script für AutoCAD wiederholt die Befehle für jede DWG-Datei und muss mit einer **Leerzeile enden**. Sollten Sie in der DWG Änderungen vornehmen (s.u.) und wollen diese auch speichern ergänzen Sie die den Befehl "*_qsave*".

```
;; Script file for AutoCAD
;; Start AutoCAD on the command line with option: /b script-file.scr"
_open
ODS1P101.DWG
_saveas
dxf
16
ODS1P101.DXF
_close
_open
ODS1P102.DWG
...
<blank line>
```

2.2 Einheiten

AutoCAD kennt Enheiten (inch, mm, m etc) und rechnet diese automatisch ineinander um. Dies wird leider oft ignoriert, so dass DWG-Dateien in der Archäologie zwar in Metern gemessen sind, die Angabe zur Einheit aber auf dem Standard "Millimeter" steht oder sogar eventuell auf Inch (Britisch). Dies kann im Export-Script gleich mit angepasst werden, um die automatische Skalierung um den Faktor 1000 bei einem heterogenem Datenbestand zu vermeiden. Dazu im vorangehenden Code nach dem Öffnen der DWG-Datei und vor dem Speichern (*_saveas*) den folgenden Code einfügen. Hierbei steht die 6 für "Meter", 5 für "Zentimeter" und 4 für "Millimeter".

```
INSUNITS
6
```

Sollten die Zeichnungen darüber hinaus tatsächlich falsch skaliert sein, kann dies in einem Zug mit folgender Befehlsfolge im Script erledigt werden. Die Leerzeile nach "all" beendet die Objektwahl und ".01" ist durch den notwendigen Faktor zu ersetzen. Für die Optik können Sie noch ein "Zoom" "G" (Grenzen) ergänzen.

```
_scale
all

0,0,0
.01
```

2.3 Schraffuren zerlegen

Schraffuren kodieren oft Informationen, sind aber schlecht in ein GIS zu überführen. Werden Schraffuren in die zugehörigen Elemente, z.B. einzelne Linien zerlegt, handelt es sich um den Import einer schlichten Geometrie. In einem GIS kann dann mit eine räumliche Verbindung (*spatial join*) zwischen den unterschiedlichen Objekten hergestellt werden. Im vorliegenden Fall könnten dann alle Steine mit mindestens einem Linienmittelpunkt vom Layer "FEUER" als gebrannt markiert werden. Dazu muss vor dem Speichern (*_saveas*) folgender Code eingefügt werden.

```
(setq SS (ssget "x" '((0 . "hatch") (8 . "FEUER"))))
(if SS
 (progn
  (setq CNT 0)
  (repeat (sslength SS)
   (vl-cmdf "._explode" (ssname SS CNT))
   (setq CNT (1+ CNT))
  )
 )
)
```

```
<blank line>
```

Da dies nicht ganz selbsterklärend ist, eine knappe Erläuterung: Die erste Zeile definiert die Variable "ss" und weist dieser mit ssget aus der gesamten Datei mit "x" alle Objekte zu, die der folgende Liste an Parametern entsprechen (*dotted pairs*, d.h. Attributkennziffer . Wert). Wenn die Variable "ss" Inhalt hat wird eine Abfolge (*progn*) von Anweisungen durchgeführt: 1. ein Zähler mit dem Startwert "0" definiert und dann auf alle Elemente der Auswahl "ss" der Befehl "*explode*" ausgeführt, wobei der jeweilige Objektname anhand des Zählers ermittelt wird.

2.4 Datenextraktion

In AutoCAD können aus einzelnen oder auch vielen Zeichnungen eines Ordners diverse Elemente mit deren Attributen als Liste exportiert werden (*_dataextraction*). Die Befehlsführung ist weitgehend intuitiv. In den einzelnen Fenstern kann die Auswahl an Elemente und Attribute durch entsprechende Anzeigeoptionen bzw. Filter gesteuert werden. Im Beispiel Odagsen "Nur Blöcke Anzeigen" für die Auswahl von "HK", "LB" und "RL". Dann den Kategorienfilter auf "Allgemein", "Geometrie" und "Zeichnung" setzen um dan nur die Attribute "Dateiname", "Layer", "Position x","Position Y" und "Position Z" zu wählen. **Wichtig**: der Export muss wegen der Punkt-Komma-Problematik als CSV-Datei gespeichert werden.

In diesem Fall erkennt der DXF-Import sowohl die Blockdefinitionen als auch die Einfügepunkte und listet diese korrekt (s.u.). Eine Datenextraktion ist deshalb nicht notwendig.

3 SpatiaLite GUI

Starten Sie die SpatiaLite GUI und erstellen Sie eine neue, leere Datenbank. In diesem Fall werden die vielen DXF-Dateien nicht einzeln, sondern der gesamte Ordner importiert: "Menu > Advanced > Import DXF drawings". Wählen Sie dann nur eine DXF-Datei aus und ändern Sie im Importfenster dann die Angabe auf "(x) Import any DXF drawing file from selected folder". Da ein lokales Koordinatensystem verwendet wurde belassen Sie SRID auf "-1". Weitere Angaben: "(x) automatic 2D/3D", "(x) mixed layers (distinct by type)", "(X) none" für das *Ring handling* also das erkennen von sog. Donuts. Nach einer kurzen Wartezeit wurden folgende Tabellen und Sichten erstellt:

- block_line_2d: die Linien der grafischen Blockdefinitionen in jeder Datei.
- insline_layer_2d: Eine Liste der eingefügten Blöcke in jeder Datei, u.a. mit Datei-, Layer und Blocknamen als auch x, y und z-Koordinate des Einfügepunktes.

- insline_layer_2d_view: Die Kombination der beiden vorgenannten Dateien in einer Sicht, die im vorliegenden Fall zwar offensichtlich korrekte Geometrien enthält, in QGIS im Kartenfenster aber dennoch nicht dargestellt wird.
- line_layer_2d: sehr viele Linien der diversen Objekte (Steine, Knochen etc.) mit jeweiligem Datei- und Layernamen.
- polyg_layer_2d: Deutlich weniger Polygone mit jeweiligem Datei- und Layernamen.
- text_layer_2d: Die Texte in den DXF-Dateien, z.B. Befund und Profilnummern, mit dem zugehörigen Einfügepunkt, Datei- und Layernamen.

3.1 Datenkontrolle

Es folgt eine Datenkontrolle mit Überarbeitung, die vor allem die Geometrien betrifft. Hier sind teils durch das unsaubere Digitalisieren doppelte Knoten vorhanden oder einige Polygone überschneiden sich selbst.

3.1.1 line_layer_2d

Zahlreiche **fehlerhafte Geometrien** können über das Kontextmenü repariert werden: Spalte "geometry" Kontextmenü > "Malfomed geometries". Liefert 501 fehlerhafte Geometrien: 1. überwiegend wiederholter Knoten (*repeated vertex*), 2. fehlerhafte Geometrie durch zu wenig Punkte. Bestätigen Sie [Repair], um das erste Problem direkt zu lösen. Sollten Sie die Daten prüfen und von Hand korrigieren wollen, dann führen Sie folgende Befehle nacheinander aus.

```
-- Eine Geometrie mit Fehler zur Kontrolle ansehen
Select AsText(geometry) from line_layer_2d
 where feature_id = 32;
-- Die Korrektur ansehen
Select AsText(SanitizeGeometry(geometry)) from line_layer_2d
 where feature_id = 32;
-- Die Korrektur ausführen
Update line_layer_2d
 set geometry = SanitizeGeometry(geometry)
 where IsValid(geometry) = 0;
```

Es bleiben 78 fehlerhafte Geometrien mit "Repeated vertex. Too few points in geometry . . . ". Die Kontrolle der Geometrie von Feature 412 zeigt drei identische Punkte. Verallgemeinernd können wir mit folgender Abfrage diese fehlerhaften erst finden und dann auch löschen:

```
-- Fehlerhafte Linien aus idetischen Punkten finden.
Select * from line_layer_2d
where ST_Length(geometry) = 0 and IsValid(geometry) <> 1;
-- Löschen dieser Linien
Delete from line_layer_2d
where ST_Length (geometry) = 0 and IsValid(geometry) <> 1;
```

Weitere Merkmale von möglicherweise fehlerhafter Linien können gesucht und ggf. gelöscht werden.

Eine Linie besteht nur aus drei Punkten, wobei Start und Endpunkt identisch sind.

```
Select feature_id, layer, NumPoints(geometry)
from line_layer_2d
where StartPoint(geometry) = EndPoint(geometry) and NumPoints(geometry) = 3;
```

Die Linien könnten alledings als einfach Linie von Bedeutung sein, deshalb wird nur ein Punkt gelöscht. Anmerkung: die Funktion RemovePoint zählt 0-basiert.

```
Update line_layer_2d
set geometry = RemovePoint(geometry, 2)
where StartPoint(geometry) = EndPoint(geometry) and NumPoints(geometry) = 3;
```

Je nach Kontext könnten Linien mit zwei Punkten oder besonders kurze Linien auch weniger plausibel sein.

Abschließend ein Überblick über den Datenbestand:

```
Select Count(*), layer, GeometryType(geometry)
from line_layer_2d
group by 2, 3;
```

3.1.2 poly_layer_2d

Auch hier zuerst **fehlerhafte Geometrien** über das Kontextmenü reparieren: Spalte "geometry" Kontextmenü > "Malfomed geometries": Wiederholte Knoten (*repeated vertex*) reparieren. Ringe mit weniger als 4 Punkten entsprechen den Linien mit identischem Start- und Endpunkt. fehlerhafte Geometrie durch zu wenig Punkte. Bestätigen Sie [Repair], um das erste Problem direkt zu lösen. Die zugehörige Funktion ist "ST_MakeValid(geom)". Sollten Sie die weiteren fehlerhaften Daten prüfen und von Hand korrigieren wollen, dann führen Sie folgende Befehle nacheinander aus.

```
Select feature_id, layer, AsText(geometry)
from polyg_layer_2d
where IsValid(geometry) = 0 and ST_NPoints(geometry) < 4;
-- Löschen mit
Delete from polyg_layer_2d
where IsValid(geometry) = 0 and ST_NPoints(geometry) < 4;
```

Es bleiben noch die sich überschneidenden Polygone, zur visuellen Kontrolle

```
Select feature_id, layer, ST_NPoints(geometry), AsText(geometry)
from polyg_layer_2d
where IsValid(geometry) <> 1;
```

Die Polygone können mit der Funktion “ST_RingsCutAtNodes(geom)” in ein Multilinenobjekt zerteilt werden. Etwas einfacher und eventuell auch erfolgreich ist die Anwendung der Funktion “MakeValid(geometry)” die ein Multipolygon zurückgibt. Um keine gemischten Geometrien zu erhalten, was durch die Prüfroutinen (*trigger*) auch verhindert wird, sollte die Geometrie von polyg_layer_2d vorher angepasst und zu einem Multipolygon verändert werden.

Anmerkung: Durch das Entfernen der Geometrieinformationen erscheinen die Index-Tabellen aus dem Bereich “Spatial Index” nach einem *refresh* jetzt unter den “User Data”. Mit Wiederherstellung des Index werden diese alten Indices gelöscht.

```
-- Prüfroutinen entfernen, die Parameter sind hier Text.
Select DiscardGeometryColumn('polyg_layer_2d', 'geometry');
-- Nur die Konvertierung zu Multipolygon und Reparatur ausführen.
Update polyg_layer_2d
set geometry = CastToMulti(MakeValid(geometry));
-- Prüfroutine wieder einrichten, Parameter als Text.
Select RecoverGeometryColumn('polyg_layer_2d', 'geometry', -1, 'MULTIPOLYGON','XY');
-- Rückgabewert 1 wenn ok.
-- Und den Index der Geometrie neu generieren.
Select CreateSpatialIndex('polyg_layer_2d', 'geometry');
-- Rückgabewert 1 wenn ok, dennoch zur Kontrolle.
Select Count(*), GeometryType(geometry), Srid(geometry), CoordDimension(geometry)
from polyg_layer_2d
group by 2, 3, 4
```

Zur Kontrolle und falls nur einfache Polygone gewünscht sein sollten:

```
Select *, ST_NumGeometries(geometry)
from polyg_layer_2d
where ST_NumGeometries(geometry) > 1;
```

Die Fälle mit genau zwei Subpolygonen können getrennt werden.

```
with recursive cnt(x) as
 (select 1
 union all
 select x+1 from cnt limit 2)
-- Ende Rekursion
Select filename, layer, AsText(CastToMulti(ST_GeometryN(geometry, x))) as geometry
 from cnt,  polyg_layer_2d as b
  where ST_NumGeometries(b.geometry) = 2;
```

Um diese Multipolygone mit genau zwei Subpolygonen getrennt anzufügen muss eine “*insert into*”-Anweisung eingefügt werden. Danach noch die ursprünglichen Polygone löschen und die Tabellen der Datenbank aktualisieren.

```
with recursive cnt(x) as
 (select 1
 union all
 select x+1 from cnt limit 2)
Insert into  polyg_layer_2d (filename, layer, geometry)
select filename, layer, CastToMulti(ST_GeometryN(geometry, x)) as geometry
 from cnt,  polyg_layer_2d as b
  where ST_NumGeometries(b.geometry) = 2;
```

```
Delete from polyg_layer_2d
 where ST_NumGeometries(geometry) = 2;
```

```
Select UpdateLayerStatistics();
```

Zum Abschluss der Prüfung eine Übersicht zum Datenbestand:

```
Select Count(*), layer, GeometryType(geometry)
from polyg_layer_2d
group by 2, 3;
```

3.2 Linien zu Polygone

Sehr viele Steine und Knochen wurden nicht als Polygone erkannt. Im Folgenden werden möglichst viele Linien, die eigentlich Polygone darstellen sollten in diese konvertiert und in die Tabelle der Polygone verschoben (kopieren & löschen). Bei diesem Pauschalen vorgehen können natürlich Ungenauigkeiten und Fehler entstehen, u.a. erneut sich selbst schneidende Polygone. Je nach Aufwand ist also eine Einzelfallprüfung vorzuziehen.

3.2.1 Pauschale Methode

Die Funktion ST_Polygonize aggregiert die Geometrien und versucht dann aus den Linien Polygone zu bilden. Durch das Aggregieren je Datei und Layer werden nachfolgend stets nur die jeweils möglichen räumlichen Bezüge der Linien berücksichtigt. Ergänzend wird eine Liste der jeweils eingehenden "*feature_id*" erstellt (*Group_Concat()*), um die Zugehörigen Linien identifizieren und löschen zu können. Je Datei und Layer wird ein Multipolygon erstellt.

```
Create table tmp_polygonize as
select filename, layer, Group_Concat(feature_id) as fids,
 CastToMulti(ST_Polygonize(geometry)) as geometry
from line_layer_2d
where layer in ('KNOCHEN', 'STEINE')
group by 1, 2;
```

```
Select Count(*), GeometryType(geometry), Srid(geometry), CoordDimension(geometry)
from tmp_polygonize
group by 2, 3, 4
```

```
Delete from tmp_polygonize
where geometry is null
```

```
Select RecoverGeometryColumn('tmp_polygonize', 'geometry', -1, 'MULTIPOLYGON', 'XY')
```

```
Select CreateSpatialIndex('tmp_polygonize', 'geometry');
```

Im nächsten Schritt werden die zahlreichen Steine bzw. Knochen die je Datei als Multipolygon vorliegen getrennt. Die rekursive Anweisung unterscheidet sich deutlich von der vorangehenden Version für genau zwei Polygone in einem Multipolygon.

```
Create table tmp_polygonize_single as
-- Beginn der Rekursion mit zwei Zählern und Geometrie
with recursive liste ( fid, n, single ) as (
 -- Erster Datensatz
 select 1 as fid, 0 as n, null as single
 union all
 -- Alle folgenden Datensätze mit Entscheidung (case)
 select
 case -- Wenn die n-te Geometrie im Multipolygone existiert behalte fid sonst fid + 1
  when (select ST_GeometryN(geometry, n + 1) from tmp_polygonize where rowid = fid) not null
   then fid
  else
   fid + 1
 end as fid, -- end case und Zuweisung
 case -- dito nur n + 1 sonst 0
```

```
  when (select ST_GeometryN(geometry, n + 1) from tmp_polygonize where rowid = fid) not null
   then n + 1
  else
   0
 end as n, -- end case und Zuweisung
 case -- dito nur wird jetzt die Geometrie übernommen
  when (select ST_GeometryN(geometry, n + 1) from tmp_polygonize where rowid = fid) not null
   then (select ST_GeometryN(geometry, n + 1) from tmp_polygonize where rowid = fid)
  else
    null
 end as single
from liste
 -- Limit für die Rekursion.
  where fid <= (select Count(*) from tmp_polygonize))
Select a.fid, a.n, CastToMulti(single) as geometry, b.filename, b.layer from liste as a
 left join tmp_polygonize as b on a.fid = b.rowid
 where single not null;
```

Diese Rekursive Abfrage ist mit den eingefügten Bedingungen (*case*) etwas umfangreicher aber nicht wirklich kompliziert. Die Grundkonstruktion der durch Rekursion gebauten Tabelle ("liste") hat drei Spalten für zwei Zähler ("fid", "n") und eine Geometrie ("single"). Die Tabelle liste startet mit dem Setzen der Zähler, wobei "fid" = 1 und "n" = 0 gesetzt wird. Für alle weiteren Datenzeilen wird per *select*-Anweisung geprüft, ob im Multipolygon der gegebenen "fid" das "n"+1 Polygon existiert. Ist dies vorhanden wird in den drei Bedingungen: 1. die "fid" behalten, 2: "n" + 1 gesetzt und 3. das zugehörige Polygon abgefragt. Alternativ wird für das nächste Multipolygon: 1. "fid" + 1 und 2. "n" zurück auf 0 gesetzt sowie 3. keine Geometrie übergeben. Das Limit für die Rekursion wird auf die Anzahl der vorhandenen Multipolygone gesetzt. Die Abfrage der durch Rekursion erstellten Tabelle "liste" wird per *join* mit den Eingangsdaten verbunden um nachträglich Dateiname und Layer zu erhalten. Zugleich wird das einfache Polygon wieder zu einem Multipolygon für die weitere Bearbeitung umgewandelt und auf die Zeilen mit Geometrie gefiltert. Die knapp 14400 Polygone werden in weniger als 3 Sekunden verarbeitet.

Danach erfolgt das Prüfen, das Wiederherstellen und ggf. die Korrektur der Geometriespalte.

```
Select Count(*), GeometryType(geometry), Srid(geometry), CoordDimension(geometry)
from tmp_polygonize_single
group by 2, 3, 4;

Select RecoverGeometryColumn('tmp_polygonize_single', 'geometry', -1, 'MULTIPOLYGON', 'XY');

Select *, AsText(geometry) from tmp_polygonize_single
 where IsValid(geometry) <> 1;

Update tmp_polygonize_single
 set geometry = ST_MakeValid(geometry)
 where IsValid(geometry) <> 1;
```

Der dann vorliegende saubere Datenbestand kann an die Tabelle polyg_layer_2d angefügt und die Tabelle selbst gelöscht werden.

```
Insert into polyg_layer_2d (filename, layer, geometry)
 select filename, layer, geometry from tmp_polygonize_single;
-- löschen
Drop table if exists tmp_polygonize_single;
```

Wenn gewünscht noch die verarbeitetn Linien aus line_layer_2d löschen, dies erfolgt in zwei Schritten: 1. Wird eine Sicht (*view*) der verarbeiteten ID's mittels Rekursion erstellt und 2. dann mit dieser entsprechenden Einträge in der Tabelle der Linien gelöscht. Eine Ursache dafür ist das in "tmp_polygonize" als ein langer Text und nicht als Liste von Zahlen abgelegte Ergebnis der vorangehenden *Group_Concat*-Anweisung. Die Rekursion arbeitet erneut mit einer *case*-Anweisung, um jeweils das erste Element bis zum ersten "," und den Rest zu trennen. In der abschließenden Abfrage wird noch auf die Elemente

("feature_id") gruppiert, um die doppelten zu eliminieren.

```
-- Einspaltige Tabelle der feature_ids
Create view tmp_delfids as
with recursive liste (element, remainder) as (
 select 0 as element, (select Group_Concat(fids) from tmp_polygonize) as remainder
  union all
 select
  case
   when instr(remainder, ',') > 0 then
    substr(remainder, 0, instr(remainder, ','))
  else
   remainder
  end as element,
  case
   when instr(remainder, ',') > 0 then
    substr(remainder, instr(remainder, ',') + 1)
  else
   null
  end as remainder
from liste
where remainder is not null
)
Select trim(element) as fid from liste where element is not null group by 1;
```

Auf diese zuvor erstellen Sicht der ID's bezieht sich dann die folgende *delete*-Anweisung. Diese führe ich jetzt aber nicht aus, um die problemorientierte Lösung noch durchzuführen.

```
Delete from cp_line_layer_2d
where feature_id in (select fid from tmp_delfids);
```

3.2.2 Polygone aus problemorientierter Lösung

Ein Problem sind die in AutoCAD nicht geschlossen Polygone bei denen Startpunkt und Endpunkt identische Werte haben. Dies ist durch eine *insert*-Anweisung einfach zu lösen. Danach sollte der Index von polyg_layer_2d aktualisiert und die ursprünglichen Linien gelöscht werden.

```
-- Anzahl der betreffenden Linien feststellen.
SELECT Count(*), layer from line_layer_2d
where StartPoint(geometry) = EndPoint(geometry)
group by 2;
-- Diese Linien zu einem Multipolygon konvertieren mit weiteren Attributen
-- in die Tabelle der Polygone schreiben
Insert into  polyg_layer_2d (filename, layer, geometry)
select filename, layer, CastToMulti(ST_BuildArea(geometry)) as geometry
from line_layer_2d where StartPoint(geometry) = EndPoint(geometry)
 and layer in ('KNOCHEN', 'STEINE');
-- Den Index aktualisieren
Select RecoverSpatialIndex(polyg_layer_2d, geometry);
-- Die ursprünglichen Linien löschen, bzw. zur Kontrolle feature_id archivieren
Delete from line_layer_2d
where StartPoint(geometry) = EndPoint(geometry)
 and layer in ('KNOCHEN', 'STEINE');
```

Bei den verbleibenden Linien sind noch sehr viele auf den Layern STEINE und KNOCHEN. Um deren Struktur zu prüfen können einige Abfragen ausgeführt werden:

```
-- Anzahl der optionlen Linien mit mehr als 2 Knoten feststellen.
SELECT Count(*), layer from line_layer_2d
where NumPoints(geometry) > 2
 and layer in ('KNOCHEN', 'STEINE')
```

```
group by 2;
-- Wie viele würden, direkt geschlossen, ein valides Polygon ergeben?
SELECT Count(*), layer, IsValid(ST_BuildArea(geometry))
from line_layer_2d
where NumPoints(geometry) > 2
 and layer in ('KNOCHEN', 'STEINE')
group by 2, 3;
```

Nach einer visuellen Prüfung liegen nicht nur ideale Polylinien vor, bei denen lediglich eine Kante wegen eines benachbarten Polygons fehlt, vielmehr sind die Strukturen deutlich komplexer. Dennoch als Beispiel hier eine mögliche Konstruktion weiterer Polygone durch das Anfügen des Starpunktes als Endpunkt.

```
Create table tmp_line2polyg as
select feature_id, filename, layer,
 CastToMulti(ST_BuildArea(ST_AddPoint(geometry, ST_StartPoint(geometry)))) as geometry
from line_layer_2d
where NumPoints(geometry) > 2
 and layer in ('KNOCHEN', 'STEINE');

Delete from tmp_line2polyg where geometry is null;

Select Count(*), GeometryType(geometry), Srid(geometry), CoordDimension(geometry)
from tmp_line2polyg
group by 2, 3, 4;

Select RecoverGeometryColumn('tmp_line2polyg', 'geometry', -1, 'MULTIPOLYGON', 'XY');

Select Count(*) , ST_IsValid(geometry), ST_SelfIntersections(geometry) from tmp_line2polyg
group by 2, 3;
```

Viele weitere Linien mit nur zwei Punkten sind maximal 13 cm lang oder haben mit drei Punkten eine Länge bis 18 cm. Eine qualifizierte Entscheidung ist hier nicht wirklich zu treffen.

3.2.3 Fazit Linie zu Polygon

Vergleichen wir am Ende die “Erfolgszahlen” schneidet die “pauschale” Methode mit 21.200 verarbeiteten Linien deutlich besser ab als die problemorientierte Lösung mit 14.300 verarbeiteten Linien. Natürlich ist der vorliegende Fall von den Ausgangsdaten eher für die pauschale Version geeignet. Bei den Befunden einer Flächengrabung ist möglicherweise die zweite eher geeignet, ein Ergebnis entsprechend der ursprünglichen Dokumentation zu liefern. Viel hängt auch von der Datenqualität, also der genuinen Strukturierung und Umsetzung der ursprünglichen CAD-Daten ab.

Je effizienter einzelne Lösungswege umgesetzt werden können desdo vielfältiger sind die möglichen Lösungen und umso größer ist auch die Bereitschaft, Ergebnisse auch zu verwerfen. Zudem bieten die aufgezeigten Skripte die Möglichkeit, kleine Variationen einzubauen. Hier liegt der große Vorteil der Automatisierung gegenüber der qualitativ hochwertigen aber langwierigen Einzelfallprüfung und Handarbeit.

Literatur

Heege, E., Heege, A., 1989. Die Häuser Der Toten. Jungsteinzeitliche Kollektivgräber Im Ldkr. Northeim, Wegweiser Zur Vor- Und Frühgeschichte Niedersachsens. Hildesheim.

Rinne, C., 2003. Odagsen Und Großenrode, Ldkr. Northeim. Jungsteinzeitliche Kollektivgräber Im Südlichen Leinetal, Beiträge Zur Archäologie in Niedersachsen. Marie Leidorf, Rahden/Westf.