CAD-Dokumentation zu GIS mit SpatiaLite migrieren

Christoph Rinne

18. Juli 2022

Inhaltsverzeichnis

Vorwort

1 Einführung

Ziel ist die Überführung von Ausgrabungsplänen aus CAD-Dateien in ein GIS. Ausgang ist die gut strukturierte Retrodigitalisierung (2D) einer Papierdokumentation aus 2 Grabungskampagnen auf der [Fundstelle Wittelsberg 7, Ldkr. Marburg Biedenkopf][1].

1.1 Verwendete Software

2 Protokoll von Schritten

2.1 SpatiaLite GUI

- Import des Gesamtbestandes der Ausgrabung (Wit7 retrodigital) als dxf-Datei (Wit7_8990_alles.dxf)
- Ergebnis Tabellen: hatch_layer_2d, hatch_layer_2D_pattern, line_layer_2d, polyg_layer_2d, text_layer_3d, text_layer_3d_attr und eine *view* zu den beiden Texttabellen.

- Erste Sichtung der linien_layer_2d, darin einige wenige Objekte von Befund-Layern (nicht geschlossene Polygone in CAD, anschließende Befundlinien, Binnenlinien eines Befundes auf falschem Layer). Korrektur im CAD und erneuter Import
- Pauschale Kombination der Befundpolygone mit den den Befundnummern der Text-Tabelle in eine *view*.

```
CREATE VIEW Pl01_BefundeNr as
select a.layer, a.number, a.info, b.label, a.geometry
from polyg_layer_2d as a
join text_layer_3d as b on st_within(b.geometry, a.geometry)
where a.layer = 'PL01_Befund'
  and b.layer = 'PL01_BefundNr';
```

Grundsätzlich zwei Wege:

1. Die vorangehende *view* als Vorlage für eine neue Tabelle verwenden und die *geometry* wieder registrieren. Danach mit Handarbeit die Probleme klären.

```
CREATE table Pl01_BefundeNr as
select a.layer, a.number, a.info, b.label, a.geometry
from polyg_layer_2d as a
join text_layer_3d as b on st_within(b.geometry, a.geometry)
where a.layer = 'PL01_Befund'
  and b.layer = 'PL01_BefundNr';
```

Das ist die arbeitsreiche und in der Datenhaltung schwierigere Variante wegen der zunehmend vielen Tabellen. Eigentlich nicht zu empfehlen. Der eigentliche Vorteil: der Lernaufwand liegt nahe NULL.

2. Die Tabelle polyg_layer_2d um sinnvolle Spalten für wichtige Informationen erweitern (Nummer, Info). Unbedingt die folgenden Updates bei exklusivem Zugriff von SpatiaLite ausführen, sonst erscheinen die Spalten nicht in QGIS. Oder rechtsklick auf die geometry-Spalte und 'Update Layer Statistics'.

Anmerkung: SQLite hat *type affinity*, trivialisierte Info: in einem Feld vom Typ *integer* kann auch '8820A' gespeichert werden.

2.1.1 Weg 2. Zentrale Datenhaltung in einer Polygon-Tabelle

```
ALTER TABLE "polyg_layer_2d"
ADD COLUMN number integer,
ADD COLUMN info text;
```

[1] Mit freundlicher Genehmigung des Landesdenmalamtes Hessen, Außenstelle Marburg.

```
UPDATE geometry_columns_statistics set last_verified = 0;
SELECT UpdateLayerStatistics();
```

Danach in die Spalte number die Befundnummer aus der vorangehenden view eintragen.

```
update polyg_layer_2d
set number = (select label
from Pl01_BefundeNr
where geometry = polyg_layer_2d.geometry)
where number is null;
```

Dann wieder Handarbeit für die Problemfälle. Jeweils die 'feature_id' im Plan abfragen und SQL-Update vornehmen:

```
update polyg_layer_2d
set number = '9013'
where feature_id = 111
```

Abschließend die 0-Nummern noch auf NULL setzen.

```
update polyg_layer_2d
set number = NULL
where number = 0
```

Den Vorgang für die weiteren Plana wiederholen.

Für die Darstellung der Befunde über alle Plana in QGIS kann die 'polyg_layer_2d' der Karte hinzugefügt werden. Objektfilter setzen auf `"layer" like 'PL0%_Befund'`, dann bei der Symbologie auf 'Abgestuft' wechseln und als Wert mit `substr("layer",4,1)` nur die Planumsnummer aus dem standardisierten Layernamen "Pl01_Befunde" holen. Alles klassifizieren und bei Legendenformat: "Planum %%2" eintragen.

Befunde mit Schraffur sind **Tiergänge?!** Dabei sind im vorliegenden Fall einige fehlerhafte Geometrien dabei.

```
SELECT *, astext(centroid(geometry))
FROM "hatch_layer_2d"
where GeometryType(geometry) not null;
```

```
update polyg_layer_2d
set info = 'TG'
where layer like '%_Befund' and
feature_id in (select a.feature_id
from polyg_layer_2d as a
inner join hatch_layer_2d as b on st_within(centroid(b.geometry),a.geometry)
where GeometryType(b.geometry) not null);
```

Die Kontrolle auch wegen der falschen Geometrien zeigt eine Fehlzuweisung wegen der Lage des Tierganges in einem anderen Befund, dazu einige fehlende Zuweisungen. Deshalb die folgende Korrekturen:

```
update polyg_layer_2d
set info = NULL
where feature_id = 69;

update polyg_layer_2d
set info = 'NULL'TG'
where feature_id in (71, 72, 34, 31, 47, 22, 28);
```

Bei den **Profilen** müssen die Linien mit Namen versehen werden.

1. Schritt alle Profillinien bekommen die Nummer des Befundpolygons (= zugehörige Befundnummer).

```
update line_layer_2d
set number = (select a.number
 from polyg_layer_2d as a
```

```
 where intersects(geometry,line_layer_2d.geometry) and
 layer like 'PL%_Befund')
where number is null;
```

2. Schritt, die Nagelnamen, z.B. A, B„ werden in die Info-Spalte geschrieben.

```
update line_layer_2d
set info = (select group_concat(b.label, ' - ')
 from line_layer_2d as a
 inner join text_layer_3d as b on intersects(b.geometry,a.geometry)
 where a.layer='PL01_Profil' and b.layer='PL01_ProfilNr'
 group by a.feature_id
 having a.feature_id=line_layer_2d.feature_id)
where info is null
```

Optional für das schnelle Abfragen der Profilinfos in den Layereigenschaften folgende “Anzeige” (pop ups, ballon, Sprechblase) bei HTML-Kartenhinweise definieren:

```
<p>Befund-Nr.: [%"number"%] <br>
Nägel: [%"info"%]<br>
Pl.: [%substr("layer",4,1)%]</p>
```

Bei den **Schnitten** sind es die Layer mit ‘Grabungsgrenze’ oder ‘Schnitt’ die bearbeitet werden müssen. Ersteres sind z.B. die Dokumentationsgrenzen um die jeweiligen Befunde, zweiters sind die “echten” Schnitte z.B. durch den Graben. Beides wird hier erstmal zusammengefasst behandelt. Die Update-Anweisung beinhaltet mehrfach verschachtelte Unterabfragen.

```
update polyg_layer_2d as c
set info = (select b.label
from (select *
   from polyg_layer_2d
   where layer like '%Grabungsgrenze%' or layer like '%Schnitt%') as a
inner join (select *
  from text_layer_3d
  where layer like '%SchnittNr') as b
on st_within(b.geometry,a.geometry)
where c.feature_id = a.feature_id)
where info is null;
```

Zahlreiche **Steine** sind ebenfalls in der Zeichnung als Polygone digitalisiert und liegen je Planum auf einem eigenen Layer. Damit ist die Aktualisierung der Spalte ‘Info’ sehr einfach.

```
update polyg_layer_2d
set info = 'Stein'
where layer like '%Steine' and info is null;
```

Dazu soll aber in der Spalte ‘number’ die zugehörige Befundnummer ergänzt werden.

```
update polyg_layer_2d as c
set number = (select a.number
 from (select number, layer, geometry from polyg_layer_2d
  where layer like '%Befund' and number not null) as a
 inner join (select feature_id, info, geometry
   from polyg_layer_2d where info = 'Stein') as b
 on st_relate(b.geometry, a.geometry)
 where c.feature_id = b.feature_id)
where info = 'Stein';
```

Einige wenige Steine liegen außerhalb eines Befundes in jeweils einem Fall in Planum 2 (Bef. 9011) und 3 (9003). Da es sich beim anstehenden Boden um Löss handelt ist die Abgenzung der Befunde offensichtlich schwierig gewesen und muss bedingt hinterfragt werden. Im Planum 1 können an der Grenze noch Steine des Pflughorizontes vorkommen.

Blöcke mit Attributen in CAD sind ein wichtiges Element in CAD-Zeichnungen mit zahlreichen Optionen und einem enormen Mehrwert, z.B. einem einfachen Export in Tabellen mit zahlreichen weiteren Informationen. Dies ist die Optimale Möglichkeit diese Informationen zu Sichern und Nachzunutzen. Leider sind beim Import der DXF-Datei in der SpatiaLite-GUI diese nicht übernommen worden. Es sind in diesem Fall Blöcke für die Niv-Werte, die Holzkohle, den Rotlehm und die Fundnummern vorhanden.

Blöcke in AutoCAD können eine unterschiedlichen Grad an Komplexität haben, der Import in ein GIS ist deshalb nicht trivial und kann jeweils zu unterschiedlichen Ergebnissen führen. Allgemein werden Objekte in einer Blockdefinition auf dem Layer '0' angelegt. Beim Einfügen werden die Objekte dann dem jeweils aktiven Layer zugewiesen und erhalten auch dessen Attribute, z.B. die Farbe. Beim Import werden diese komplexen Strukturen unterschiedlich aufgelöst aber immer als Punkt-Objekt des Einfügepunktes in AutoCAD behandelt. In QGIS zeigt die Attributtabelle zu den importierten Punktobjekten etwa folgendes:

layer	subclasses	entityhandle	text
PL01_BefundNr	AcDbEntity:AcDbBlockReference	77	
PL01_BefundNr	AcDbEntity:AcDbText:AcDbText	1665	9021
0	AcDbEntity:AcDbText:AcDbAttribute	461	9021
0	AcDbEntity:AcDbText:AcDbAttribute	113F	9005-Keramik
0	AcDbEntity:AcDbText:AcDbAttribute	6F7	03/04/2021
0	AcDbEntity:AcDbText:AcDbAttribute	BD4	207,95
PL01_FundNr	AcDbEntity:AcDbText:AcDbText	2CF	BL

Die ersten drei Zeilen der Tabelle für den Block der Befund-Nr 9021 nach der Spalte *subclasses* 1. die eigentliche Blockreferenz, 2. den mit dem Block auch eingefügten Text und 3. das Attribut des Blockes. Letzteres wird aber aufgelöst auf den ursprünglichen Layer '0' der Layerdefinition. Das gleiche passiert auch mit den Attributen der Fundnummern, den Datumsangaben der Erfassung und den Niv-Werten. Nur die als Text und nicht als Attribut oder grafisches Symbol eingefügten Hinweise auf Brandlehm werden auf dem jeweiligen Layer aufgelöst.

Bis auf die Informationen zur Lage von Brandlehm und Holzkohle sind diese Informationen ohne den Bezug zum Planum bzw. dem ursprünglichen Layer in CAD unbrauchbar. Die DXF-Import-Erweiterung "AnotherDXFImporter" bietet faszinierende Möglichkeiten, ist bei dieser Aufgabe aber auch keine bessere Option.

Lösungen für das Import-Problem CAD-Blöcke:

1. Kontrollieren Sie die vorliegende Dokumentation, wer mit Blöcken in CAD arbeitet weiß meist was er tut und exportiert diese Daten in CSV-Tabellen.
2. Nehmen Sie den Export der Blockinformationen aus AutoCAD selber vor, ggf. unter Verwendung einer kostengünstigeren Alternative wie BricsCAD. AutoCAD gibt es für die Lehre auch 1 Jahr kostenlos und BicsCAD als Demo-Version.
3. Importieren Sie die originale DWG- oder DXF-Datei in Esri ArcMap, dort werden Blöcke mit Attributen korrekt zu Punkten mit entsprechenden Attributen aufgelöst.

Eine in CSV exportierte Blockliste kann so aussehen:

Name	Layer	NUMMER	Position X	Position Y	Position Z
Brandlehm	PL01_FundNr		3490060.7999	5627322.4548	0.0000
Fund	PL01_FundNr	9005-Keramik	3490067.9820	5627321.1025	0.0000

Für jeden Block wird der Name des Blocks angegeben, der Layer, alle Attribute als getrennte Spalten, z.B. Nummer und die Koordinaten. Der Import mit folgender Kartirung und Auswertung sind damit leicht möglich.

Der **Import der CSV-Tabelle** mit Daten ist über das entsprechende Icon einfach möglich. Beachten Sie im folgenden Fenster 1. den Tabellennamen (z.B. block_attr_3d), 2. [x] First line . . . , 3. Column separator

(x) Comma, 4. Decimal separator (x) Point, 5. Charset Encoding (vermutl. UTF-8). Nachfolgend wird eine Geometriespalte ergänzt und aus den Daten gefüllt. Bitte beachten: anders als in GIS sind Punkte sind in CAD immer 3D, nur ggf. mit dem Z-Wert 0.

```
SELECT AddGeometryColumn('block_attr_3d', 'geometry', 31467, 'POINTZ', 'XYZ');

update block_attr_3d as a
set geometry = (select st_geomfromtext(st_astext(st_geomfromtext(
 'POINTZ(' || "Position_X" || ' ' || "Position_Y" || ' ' ||
  "Position_Z" || ')')), 31467)
 from block_attr_3d
 where a.PK_UID = PK_UID)
```