Lekce 14: Atributové operace

Python pro GIS - Práce s tabulkami a poli

Vojtěch Barták, FŽP ČZU Praha

2025-11-19

Table of contents

1 Data a nastavení

Data ke stažení:

Z Lekce 13 (rastrové analýzy): - `katastry_Liberec.shp` - katastry v Libereckém kraji - `okresy.shp` - okresy

Data pro atributové operace (`data_odtoky/`): - `UrbanAtlas_LK.shp` - krajinný pokryv Urban Atlas pro Liberecký kraj - `obce_LK.shp` - obce v Libereckém kraji - `coef.dbf` - tabulka součinitelů odtoku pro typy povrchu

Z ArcČR500: - `Sidla.shp` - sídla v ČR - `Silnice.shp` - silniční síť - `VUSC.shp` - kraje - `Okresy_VUSC.shp` - okresy s kódem kraje - `Sidla_pocty.dbf` - tabulka s počty obyvatel sídel

Nastavení prostředí:

```
import arcpy

# Workspace
arcpy.env.workspace = r"C:\TEMP\Python_GIS\Lekce_14"
arcpy.env.overwriteOutput = True
```

> **i** Pracovní prostředí
>
> **Úlohy 1-20** řešte v **IDLE** nebo v **Python Window** v ArcGIS Pro.
> **Úlohy 21-22** (komplexní analýzy) řešte v **Python Notebooku**.

---

2 Úvod

V této lekci probereme operace, které jste většinou zvyklí v grafickém prostředí ArcGIS provádět "ručně", často kliknutím pravým tlačítkem myši na pole atributové tabulky či vrstvu a výběrem nástroje z kontextového menu. Tyto operace zahrnují například:

- Přidání nového pole do atributové tabulky
- Smazání existujícího pole atributové tabulky
- Výpočet hodnot pole pomocí *Field Calculator*, případně *Calculate Geometry*
- Sumarizaci hodnot pole/polí atributové tabulky
- Propojení tabulek pomocí *Join*
- Atributový výběr prvků pomocí SQL dotazu

Ke všem těmto "ručním" operacím existují odpovídající geoprocessingové nástroje z ArcToolboxu, které umožňují jejich provádění v rámci modelů z Model Builderu či v rámci skriptů v Pythonu. Většina z nich je umístěná v nástrojovém balíčku *Data Management Tools*.

---

3 Práce s vrstvami

Než se pustíme do práce s poli a tabulkami, je důležité pochopit rozdíl mezi **datovou sadou** (data uložená na disku) a **vrstvou** (pohled na data v paměti).

Datová sada (feature class, shapefile) je fyzický soubor s daty uloženými na disku. Když s ní pracujeme v Pythonu, přímo měníme data na disku.

Vrstva (layer) je dočasný pohled na datovou sadu, který existuje pouze v operační paměti. Vrstva může mít definovaný výběr prvků, symbologii, propojení s jinými tabulkami atd. Když zavřeme ArcGIS Pro (nebo smažeme vrstvu), vrstva zmizí, ale původní data zůstávají nedotčená.

3.1 Úloha 1: Vytvoření vrstvy

Zadání: Vytvořte vrstvu z `katastry_Liberec.shp` s názvem "katastry_layer".

Vrstva se vytváří nástrojem **Make Feature Layer** z balíčku *Data Management Tools*. Prvním parametrem je cesta k datové sadě, z níž chceme vrstvu vytvořit, druhým parametrem je název vrstvy. Výsledek volání nástroje (objekt třídy *Result*) můžeme uložit do proměnné, která bude následně vrstvu zastupovat v dalších nástrojích.

```
# Vytvoření vrstvy
katastry_lyr = arcpy.management.MakeFeatureLayer(
    "katastry_Liberec.shp",
    "katastry_layer"
)

print("Vrstva vytvořena")

# Výpis názvů polí vrstvy
print("\nPole v tabulce:")
for field in arcpy.ListFields(katastry_lyr):
    print(f"  {field.name}")
```

> **!** Zamykání dat
>
> Když z datové sady vytvoříte vrstvu pomocí `MakeFeatureLayer`, tato datová sada se "zamkne" - vytvoří se k ní soubor zámku, zamezující jakémukoli programu provádět v datové sadě změny.
> Pokud budete chtít v datové sadě přidávat či mazat pole, nebude to možné, pokud je z ní vytvořena vrstva. Po smazání vrstvy nástrojem `Delete` se soubor zámku smaže a datová sada se odemkne pro úpravy.

3.2 Úloha 2: Atributový výběr

Zadání: Z vrstvy katastrů vyberte pouze ty, které mají v názvu slovo "Liberec".

Atributový výběr se provádí nástrojem **Select Layer By Attribute**. Tento nástroj používá SQL syntaxi pro definici podmínek výběru.

```
# Výběr katastrů s "Liberec" v názvu
arcpy.management.SelectLayerByAttribute(
    katastry_lyr,
    "NEW_SELECTION",
    "NAZEV LIKE '%Liberec%'"
)

# Zjištění počtu vybraných prvků
result = arcpy.management.GetCount(katastry_lyr)
count = int(result.getOutput(0))
print(f"Vybráno katastrů: {count}")
```

Parametry:

- První parametr: vrstva
- Druhý parametr: typ výběru
  - `"NEW_SELECTION"` - nový výběr (smaže předchozí)
  - `"ADD_TO_SELECTION"` - přidá k výběru
  - `"REMOVE_FROM_SELECTION"` - odebere z výběru
  - `"CLEAR_SELECTION"` - smaže výběr
- Třetí parametr: SQL podmínka (where clause)

SQL operátory:

- `LIKE` - hledání vzoru (% = libovolné znaky)

- `=`, `<>`, `>`, `<`, `>=`, `<=` - porovnání
- `AND`, `OR`, `NOT` - logické operátory
- `IN (hodnota1, hodnota2, ...)` - jeden z hodnot

> **Note**
>
> Vytvoření vrstvy s výběrem lze provést v jediném kroku. Nástroj `MakeFeatureLayer` má totiž nepovinný parametr, ve kterém je možné přímo zadat SQL dotaz. Vrstva se pak vytvoří rovnou s výběrem.
> K čemu je tedy vytváření vrstvy v samostatném kroku? Umožňuje např. vytvořit nejprve vrstvu pomocí `MakeFeatureLayer` a následně v ní v cyklu dynamicky měnit výběr pomocí `SelectLayerByAttribute` a provádět nějakou analýzu jen s příslušným výběrem.

3.3 Úloha 3: Kombinace podmínek

Zadání: Z vrstvy obcí vyberte obce s počtem obyvatel větším než 5000 **A ZÁROVEŇ** s rozlohou menší než 20 km².

```
# Nejprve vytvoříme vrstvu z obcí
obce_lyr = arcpy.management.MakeFeatureLayer("obce_LK.shp", "obce_layer")

# Přidáme pole pro plochu (bude potřeba pro podmínku)
arcpy.management.AddField(obce_lyr, "AREA_KM", "DOUBLE")
arcpy.management.CalculateField(
    obce_lyr,
    "AREA_KM",
    '!shape.area!/1000000',
    "PYTHON3"
)

# Výběr s kombinací podmínek
arcpy.management.SelectLayerByAttribute(
    obce_lyr,
    "NEW_SELECTION",
    "OBYVATEL > 5000 AND AREA_KM < 20"
)

# Uložení výběru do nového shapefile
arcpy.management.CopyFeatures(obce_lyr, "obce_velke_kompaktni.shp")

print("Výběr uložen")
```

3.4 Úloha 4: Tabulkový pohled

Zadání: Vytvořte tabulkový pohled na `Sidla_pocty.dbf` obsahující pouze sídla s počtem obyvatel větším než 10000.

Pro samostatné tabulky (ne prostorové vrstvy) se místo `MakeFeatureLayer` používá **Make Table View**:

```
# Vytvoření tabulkového pohledu s filtrem
sidla_view = arcpy.management.MakeTableView(
    "Sidla_pocty.dbf",
    "sidla_view",
    "POCET_OB > 10000"
)

# Zjištění počtu záznamů
result = arcpy.management.GetCount(sidla_view)
print(f"Velkých sídel: {result.getOutput(0)}")
```

---

4 Práce s poli

Nástroje pro práci s poli atributových tabulek, jako je přidání pole, smazání pole, či výpočet hodnot pole, nalezneme v nástrojové sadě *Data Management Tools* → *Fields*.

4.1 Úloha 5: Přidání pole

Zadání: Do `obce_LK.shp` přidejte nové pole "AREA_KM" typu DOUBLE.

Přidání pole se provádí nástrojem **Add Field**. Jeho použití si předvedeme na příkladu přidání číselného pole do tabulky vrstvy obcí.

```
obce = "obce_LK.shp"

# Přidání číselného pole
arcpy.management.AddField(obce, "AREA_KM", "DOUBLE")

print("Pole AREA_KM přidáno")
```

Parametry:

- První parametr: název vstupní tabulky (může být vektorová třída prvků nebo samostatná tabulka)
- Druhý parametr: název přidávaného pole
- Třetí parametr: datový typ pole

Běžné datové typy:

- `"TEXT"` - textový řetězec
- `"DOUBLE"` - desetinné číslo
- `"LONG"` - celé číslo
- `"SHORT"` - krátké celé číslo
- `"DATE"` - datum

> ⚠ Pozor na existující pole!
>
> Při přidávání pole je třeba dávat pozor na to, zda pole s daným názvem již v tabulce není. Pokud ano, nástroj *Add Field* pole přidá a pouze na existenci původního pole se stejným názvem upozorní hláškou typu "Warning". Tím se ovšem původní pole nenávratně smaže.

Existující pole (pokud není povinné, jako např. pole FID či Shape) lze smazat pomocí nástroje **Delete Field**:

```
# Smazání pole
arcpy.management.DeleteField(obce, "AREA_KM")
```

4.2 Úloha 6: Výpočet konstantní hodnoty

Zadání: Vytvořte pole "KRAJ" a vyplňte ho konstantou "Liberecký".

Výpočet hodnot pole tabulky se provádí nástrojem **Calculate Field** z nástrojové sady *Fields*. Prvním parametrem je cesta ke vstupní datové sadě, druhým název pole, které chceme vypočítat. Následuje výraz výpočtu.

```
obce = "obce_LK.shp"

# Přidání textového pole
arcpy.management.AddField(obce, "KRAJ", "TEXT")

# Vyplnění pole konstantou
arcpy.management.CalculateField(obce, "KRAJ", '"Liberecký"')

print("Pole KRAJ vyplněno")
```

Všimněte si, že v případě textového atributu bylo třeba výraz "Liberecký" napsat pomocí obou typů uvozovek: `'"Liberecký"'`. To proto, že součástí výrazu musí v případě textu být i uvozovky, přičemž samotný výraz je nástroji předáván také jako text. Vnější, jednoduché uvozovky ohraničují celý text výrazu, kdežto vnitřní, dvojité uvozovky jsou již součástí výrazu samotného a sdělují, že obsah výrazu je text.

Podobně pro číselné pole:

```
# Přidání číselného pole
arcpy.management.AddField(obce, "CONST", "LONG")

# Vyplnění konstantou
arcpy.management.CalculateField(obce, "CONST", 100)
```

4.3 Úloha 7: Výpočet z jiného pole

Zadání: Vytvořte pole "NAZEV_UPPER" a zkopírujte do něj názvy obcí převedené na velká písmena.

Při vyplňování pole je možné použít hodnoty kteréhokoli jiného, již existujícího pole. Na toto pole se lze odkázat jeho názvem, uvedeným mezi vykřičníky. Je přitom možné hodnoty různých polí libovolně kombinovat, provádět s nimi výpočty apod. Ve výrazu je možné použít jakoukoli funkci či operaci, která je součástí základní výbavy Pythonu. Čtvrtým, volitelným parametrem nástroje *Calculate Field* specifikujeme, že používáme jazyk Python 3.

```
obce = "obce_LK.shp"

# Přidání nového pole
arcpy.management.AddField(obce, "NAZEV_UPPER", "TEXT")

# Vyplnění pole s použitím metody upper()
arcpy.management.CalculateField(
    obce,
    "NAZEV_UPPER",
    '!NAZEV!.upper()',
    "PYTHON3"
)

print("Názvy převedeny na velká písmena")
```

Všimněte si použití vykřičníků pro odkaz na pole "NAZEV" v rámci výrazu, a na použití metody `.upper()` pro převod na velká písmena.

Tip

Užitečné metody pro textové řetězce:

- `.upper()` - převod na velká písmena
- `.lower()` - převod na malá písmena
- `.replace(old, new)` - nahrazení textu
- `.strip()` - odstranění bílých znaků

4.4 Úloha 8: Výpočet plochy

Zadání: Vypočítejte plochu katastrů v km² do pole "AREA_KM".

Velmi častou úlohou je výpočet geometrických vlastností prvků do atributu. "Ruční" řešení by spočívalo v použití nástroje *Calculate Geometry* dostupného v kontextovém menu příslušného pole. V Pythonu se operace provede pomocí nástroje *Calculate Field*, kde se v rámci výrazu odkážeme na speciální pole `!shape.area!`:

```
katastry = "katastry_Liberec.shp"

# Přidání pole pro plochu
arcpy.management.AddField(katastry, "AREA_KM", "DOUBLE")

# Výpočet plochy v km² (shape.area je v m²)
arcpy.management.CalculateField(
    katastry,
    "AREA_KM",
    '!shape.area!/1000000',
    "PYTHON3"
)

print("Plochy vypočítány")
```

4.5 Úloha 9: Výpočet obvodu

Zadání: Vypočítejte obvod polygonů Urban Atlas v km do pole "PERIMETER_KM".

```
urban_atlas = "UrbanAtlas_LK.shp"

# Přidání pole
arcpy.management.AddField(urban_atlas, "PERIMETER_KM", "DOUBLE")
```

```
# Výpočet obvodu v km (shape.length je v m)
arcpy.management.CalculateField(
    urban_atlas,
    "PERIMETER_KM",
    '!shape.length!/1000',
    "PYTHON3"
)

print("Obvody vypočítány")
```

4.6 Úloha 10: Výpočet délky linií

Zadání: Vypočítejte délku silnic v km do pole "LENGTH_KM".

```
silnice = "Silnice.shp"

# Přidání pole
arcpy.management.AddField(silnice, "LENGTH_KM", "DOUBLE")

# Výpočet délky (shape.length u linií)
arcpy.management.CalculateField(
    silnice,
    "LENGTH_KM",
    '!shape.length!/1000',
    "PYTHON3"
)

print("Délky silnic vypočítány")
```

Geometrické vlastnosti:

- **Polygony:**
  - `!shape.area!` - plocha (v m^2)
  - `!shape.length!` - obvod (v m)
- **Linie:**
  - `!shape.length!` - délka (v m)
- **Body:**
  - `!shape.X!` - souřadnice X
  - `!shape.Y!` - souřadnice Y

4.7 Úloha 11: Perimeter-Area Ratio

Zadání: Do `UrbanAtlas_LK.shp` přidejte pole "PA_RATIO" a vypočtěte do něj poměr obvodu a plochy.

Perimeter-Area Ratio je krajinná metrika popisující míru složitosti tvaru krajinných plošek. Čím je hodnota vyšší, tím je tvar složitější (členitější okraj).

```
urban_atlas = "UrbanAtlas_LK.shp"

# Přidání pole
arcpy.management.AddField(urban_atlas, "PA_RATIO", "DOUBLE")

# Výpočet PA_RATIO = obvod / plocha
arcpy.management.CalculateField(
    urban_atlas,
    "PA_RATIO",
    '!shape.length!/!shape.area!',
    "PYTHON3"
)

print("PA_RATIO vypočítán")
```

---

5 Propojování tabulek

Propojení tabulek pomocí metody *join* má za následek přidání pole (či polí) z připojované tabulky do cílové tabulky, a to na základě nějakého společného atributu (tzv. *klíče*). Tento typ propojení je možný v případě, kdy jednomu řádku cílové tabulky odpovídá max. jeden řádek připojované tabulky. Zároveň platí, že jednomu řádku připojované tabulky může odpovídat jeden či více řádků cílové tabulky.

Propojení metodou *join* lze v praxi provést dvěma způsoby:

- jako **dočasné propojení**, které je definované pouze v rámci vrstev
- jako **permanentní** rozšíření cílové tabulky o nové sloupce

5.1 Dočasné propojení

Dočasné propojení je právě ten způsob, který jste pravděpodobně zvyklí provádět v prostředí ArcGIS "ručně".

Jelikož je dočasné propojení definováno na úrovni vrstvy, nikoli datové sady, je pro jeho použití pomocí Pythonu třeba nejprve vytvořit ze zdrojové sady vrstvu. Vrstva se vytvoří v operační paměti a bude uložena do proměnné, pomocí které s ní bude možné pracovat.

5.2 Úloha 12: Dočasné propojení

Zadání: K vrstvě `UrbanAtlas_LK.shp` dočasně připojte tabulku `coef.dbf` podle pole CODE.

```
# Vstupní data
land_cover = "UrbanAtlas_LK.shp"
coef_table = "coef.dbf"

# Vytvoření vrstvy
lyr = arcpy.management.MakeFeatureLayer(land_cover, "ua_layer")

# Připojení tabulky
arcpy.management.AddJoin(lyr, "CODE", coef_table, "CODE")

# Výpis názvů polí připojené vrstvy
print("Pole v připojené tabulce:")
for field in arcpy.ListFields(lyr):
    print(f"  {field.name}")
```

Výsledek:

```
Pole v připojené tabulce:
  UrbanAtlas_LK.FID
  UrbanAtlas_LK.Shape
  UrbanAtlas_LK.CODE
  UrbanAtlas_LK.ITEM
  UrbanAtlas_LK.SHAPE_AREA
  coef.OID
  coef.CODE
  coef.nazev
  coef.ko_vsak
```

Jak je vidět, v tabulce vrstvy jsou názvy sloupců doplněny názvem tabulky, z níž sloupce pochází. Díky tomu se v této vrstvě můžeme odkazovat na sloupce z připojené tabulky.

5.3 Úloha 13: Použití připojených dat

Zadání: Ve vrstvě s připojenou tabulkou vypočítejte pole "ODTOK" jako součin plochy a součinitele odtoku.

```
# Pokračujeme s vrstvou z předchozí úlohy

# Přidání pole pro výpočet
# POZOR: přidáváme do vrstvy, ne do původní datové sady!
arcpy.management.AddField(lyr, "ODTOK", "DOUBLE")

# Výpočet s použitím polí z obou tabulek
# Pozor na správné názvy s prefixem!
arcpy.management.CalculateField(
    lyr,
    "ODTOK",
    '!UrbanAtlas_LK.shape.area! * !coef.ko_vsak!',
    "PYTHON3"
)

print("Odtok vypočítán")
```

Prefixy u připojených polí

Při práci s připojenými poli musíte používat **plné názvy s prefixem**:

- `!NazevTabulky.NazevPole!` pro pole z připojené tabulky
- `!NazevVrstvy.NazevPole!` pro pole z původní vrstvy

Bez prefixu by Python nevěděl, ze které tabulky pole vzít!

5.4 Úloha 14: Odstranění propojení

Zadání: Odstraňte propojení z vrstvy.

Pokud již propojení tabulek dále nepotřebujeme, je třeba je z vrstvy odstranit nástrojem **Remove Join**:

```
# Odstranění propojení
arcpy.management.RemoveJoin(lyr, "coef")

print("Propojení odstraněno")

# Ověření - výpis polí
for field in arcpy.ListFields(lyr):
    print(field.name)
```

Namísto odstranění propojení je možné také vrstvu smazat. Nestačí ovšem použít pythonovský příkaz `del`, je nutné vrstvu smazat nástrojem **Delete**:

```
# Smazání vrstvy
arcpy.management.Delete(lyr)

print("Vrstva smazána")
```

5.5 Trvalé propojení

Pokud chceme připojit pole trvale, máme dvě možnosti:

1. Provedeme nejprve dočasné propojení výše uvedeným způsobem, a pak výslednou vrstvu uložíme do nové datové sady nástrojem **Copy Features**
2. Připojíme pole přímo do tabulky zdrojové datové sady pomocí nástroje **Join Field**

5.6 Úloha 15: Trvalé připojení jednoho pole

Zadání: K tabulce `Sidla.shp` trvale připojte pole POCET_OB z tabulky `Sidla_pocty.dbf`.

Nástroj **Join Field** připojí pole přímo do tabulky zdrojové datové sady. Zde je možné připojit buď všechna pole, nebo jen vybraná:

```
# Trvalé připojení pole
arcpy.management.JoinField(
    in_data="Sidla.shp",
    in_field="KOD",
    join_table="Sidla_pocty.dbf",
    join_field="KOD",
    fields=["POCET_OB"]  # seznam polí k připojení
)
```

```
print("Pole POCET_OB trvale připojeno")
```

> ⚠ Pozor - trvalá změna!
>
> Na rozdíl od `AddJoin`, nástroj `JoinField` **přímo mění** vstupní tabulku! Pole z připojené tabulky se stanou trvalou součástí cílové tabulky.

5.7 Úloha 16: Trvalé připojení více polí

Zadání: K `Okresy_VUSC.shp` připojte název a kód kraje z tabulky `VUSC.shp`.

```
# Připojení více polí najednou
arcpy.management.JoinField(
    in_data="Okresy_VUSC.shp",
    in_field="KOD_VUSC",
    join_table="VUSC.shp",
    join_field="KOD_VUSC",
    fields=["NAZEV_VUSC", "KOD_VUSC"]
)

print("Informace o krajích připojeny")
```

---

6 Sumarizace tabulky

Sumarizace tabulky pomocí nástroje **Statistics** umožňuje vypočítat agregační statistiky (součet, průměr, minimum, maximum, počet) pro vybraná pole. Výsledkem je nová tabulka obsahující vypočítané hodnoty.

6.1 Úloha 17: Základní sumarizace

Zadání: Spočítejte celkovou rozlohu všech katastrů v Libereckém kraji.

```python
katastry = "katastry_Liberec.shp"

# Nejprve musíme mít pole AREA_KM (viz úloha 8)
# Pokud ho nemáme, vytvoříme ho:
arcpy.management.AddField(katastry, "AREA_KM", "DOUBLE")
arcpy.management.CalculateField(
    katastry,
    "AREA_KM",
    '!shape.area!/1000000',
    "PYTHON3"
)

# Sumarizace - celková plocha
arcpy.analysis.Statistics(
    in_table=katastry,
    out_table="stat_katastry_celkem",
    statistics_fields=[["AREA_KM", "SUM"]]
)

print("Statistiky vypočítány")

# Výpis výsledku
with arcpy.da.SearchCursor("stat_katastry_celkem", ["SUM_AREA_KM"]) as cursor:
    for row in cursor:
        print(f"Celková plocha katastrů: {row[0]:.2f} km²")
```

Parametry nástroje Statistics:

- `in_table` - vstupní tabulka
- `out_table` - výstupní tabulka se statistikami
- `statistics_fields` - seznam polí a funkcí: `[["pole", "funkce"], ...]`
- `case_field` - pole pro skupinování (volitelné)

Statistické funkce:

- `"SUM"` - součet
- `"MEAN"` - průměr
- `"MIN"` - minimum
- `"MAX"` - maximum
- `"COUNT"` - počet

6.2 Úloha 18: Sumarizace po skupinách

Zadání: Spočítejte celkovou rozlohu a průměrnou rozlohu katastrů pro každý okres.

```
katastry = "katastry_Liberec.shp"

# Nejprve musíme k datům připojit informaci o okresu
# To provedeme pomocí JoinField z vrstvy okresů
arcpy.management.JoinField(
    in_data=katastry,
    in_field="KOD_OKRESU",  # předpokládáme, že toto pole existuje
    join_table="okresy.shp",
    join_field="KOD",
    fields=["NAZEV_OKRESU"]
)

# Sumarizace podle okresů
arcpy.analysis.Statistics(
    in_table=katastry,
    out_table="stat_katastry_okresy",
    statistics_fields=[
        ["AREA_KM", "SUM"],
        ["AREA_KM", "MEAN"]
    ],
    case_field="NAZEV_OKRESU"  # skupinování podle okresu
)

print("Statistiky po okresech vypočítány")

# Výpis výsledků
with arcpy.da.SearchCursor("stat_katastry_okresy",
                           ["NAZEV_OKRESU", "SUM_AREA_KM", "MEAN_AREA_KM"]) as cursor:
    print(f"\n{'Okres':<30} | {'Celková plocha':>15} | {'Průměrná plocha':>17}")
    print("-" * 70)
    for row in cursor:
        print(f"{row[0]:<30} | {row[1]:>15.2f} | {row[2]:>17.2f}")
```

i Parametr case_field

Když použijete `case_field`, nástroj **Statistics** vytvoří jeden řádek výsledné tabulky pro každou **unikátní hodnotu** v tomto poli. Výsledky jsou tedy **seskupené podle hodnot** v case_field.

> Bez `case_field` se sumarizuje celá tabulka do jednoho řádku.

6.3 Úloha 19: Kombinace funkcí

Zadání: Pro každý typ krajinného pokryvu (Urban Atlas) spočítejte: počet plošek, celkovou plochu, průměrnou plochu, minimální a maximální plochu.

```
urban_atlas = "UrbanAtlas_LK.shp"

# Přidání pole AREA_KM
arcpy.management.AddField(urban_atlas, "AREA_KM", "DOUBLE")
arcpy.management.CalculateField(
    urban_atlas,
    "AREA_KM",
    '!shape.area!/1000000',
    "PYTHON3"
)

# Sumarizace s kombinací funkcí
arcpy.analysis.Statistics(
    in_table=urban_atlas,
    out_table="stat_landcover",
    statistics_fields=[
        ["AREA_KM", "COUNT"],
        ["AREA_KM", "SUM"],
        ["AREA_KM", "MEAN"],
        ["AREA_KM", "MIN"],
        ["AREA_KM", "MAX"]
    ],
    case_field="CODE"
)

print("Statistiky krajinného pokryvu vypočítány")
```

6.4 Úloha 20: Připojení statistik zpět k původní tabulce

Zadání: K původní tabulce katastrů připojte celkovou rozlohu okresu, ve kterém se katastr nachází. Poté vypočítejte podíl rozlohy katastru na rozloze okresu.

```python
katastry = "katastry_Liberec.shp"

# 1. Sumarizace - celková plocha okresů
arcpy.analysis.Statistics(
    in_table=katastry,
    out_table="stat_okresy",
    statistics_fields=[["AREA_KM", "SUM"]],
    case_field="KOD_OKRESU"
)

# 2. Připojení výsledků zpět k původní tabulce
arcpy.management.JoinField(
    in_data=katastry,
    in_field="KOD_OKRESU",
    join_table="stat_okresy",
    join_field="KOD_OKRESU",
    fields=["SUM_AREA_KM"]
)

# 3. Výpočet podílu
arcpy.management.AddField(katastry, "PODIL_OKRESU", "DOUBLE")
arcpy.management.CalculateField(
    katastry,
    "PODIL_OKRESU",
    '!AREA_KM!/!SUM_AREA_KM!*100',
    "PYTHON3"
)

print("Podíl rozlohy na okresu vypočítán")
```

---

7 Komplexní úlohy - Python Notebook

> **!** Python Notebook
>
> **Následující úlohy řešte v Python Notebooku**, ne v Python Window. Tyto úlohy kombinují techniky z celé lekce a vyžadují strukturovanější přístup.

7.1 Úloha 21: Analýza hustoty zalidnění

Zadání: Pro každý okres spočítejte hustotu zalidnění (obyvatel/km²).

Data: - `Okresy_VUSC.shp` - polygony okresů - `Sidla.shp` - bodová vrstva sídel - `Sidla_pocty.dbf` - tabulka s počty obyvatel

Postup:

```
import arcpy

# Nastavení
arcpy.env.workspace = r"C:\TEMP\Python_GIS\Lekce_14"
arcpy.env.overwriteOutput = True

print("=" * 60)
print("ANALÝZA HUSTOTY ZALIDNĚNÍ OKRESŮ")
print("=" * 60)

# 1. Výpočet plochy okresů
print("\n1. Počítám plochu okresů...")
okresy = "Okresy_VUSC.shp"

arcpy.management.AddField(okresy, "AREA_KM", "DOUBLE")
arcpy.management.CalculateField(
    okresy,
    "AREA_KM",
    '!shape.area!/1000000',
    "PYTHON3"
)
print("   OK")

# 2. Připojení počtu obyvatel k sídlům
print("\n2. Připojuji počty obyvatel k sídlům...")
sidla = "Sidla.shp"

arcpy.management.JoinField(
    in_data=sidla,
    in_field="KOD",
    join_table="Sidla_pocty.dbf",
    join_field="KOD",
    fields=["POCET_OB"]
)
```

```
print("   OK")

# 3. Spatial Join - připojení kódu okresu k sídlům
print("\n3. Přiřazuji sídla k okresům (Spatial Join)...")
sidla_okresy = "sidla_s_okresy.shp"

arcpy.analysis.SpatialJoin(
    target_features=sidla,
    join_features=okresy,
    out_feature_class=sidla_okresy,
    join_type="KEEP_COMMON",
    match_option="WITHIN"
)
print("   OK")

# 4. Sumarizace - součet obyvatel pro každý okres
print("\n4. Sumarizuji počty obyvatel podle okresů...")
arcpy.analysis.Statistics(
    in_table=sidla_okresy,
    out_table="stat_obyvatele_okresy",
    statistics_fields=[["POCET_OB", "SUM"]],
    case_field="KOD_OKRESU"
)
print("   OK")

# 5. Připojení počtu obyvatel k okresům
print("\n5. Připojuji počty obyvatel k okresům...")
arcpy.management.JoinField(
    in_data=okresy,
    in_field="KOD",
    join_table="stat_obyvatele_okresy",
    join_field="KOD_OKRESU",
    fields=["SUM_POCET_OB"]
)
print("   OK")

# 6. Výpočet hustoty zalidnění
print("\n6. Počítám hustotu zalidnění...")
arcpy.management.AddField(okresy, "HUSTOTA", "DOUBLE")
arcpy.management.CalculateField(
    okresy,
    "HUSTOTA",
```

```
    '!SUM_POCET_OB!/!AREA_KM!',
    "PYTHON3"
)
print("   OK")

# 7. Výpis výsledků
print("\n" + "=" * 60)
print("VÝSLEDKY - HUSTOTA ZALIDNĚNÍ")
print("=" * 60)

with arcpy.da.SearchCursor(okresy,
                           ["NAZEV_OKRESU", "SUM_POCET_OB", "AREA_KM", "HUSTOTA"],
                           sql_clause=(None, "ORDER BY HUSTOTA DESC")) as cursor:
    print(f"\n{'Okres':<30} | {'Obyvatel':>10} | {'Plocha (km²)':>12} | {'Hustota':>10}")
    print("-" * 70)
    for row in cursor:
        if row[1] is not None:  # kontrola, že má data
            print(f"{row[0]:<30} | {row[1]:>10.0f} | {row[2]:>12.2f} | {row[3]:>10.2f}")

print("\n" + "=" * 60)
print("ANALÝZA DOKONČENA")
print("=" * 60)
```

7.2 Úloha 22: Dávkové zpracování - kategorizace obcí

Zadání: Pro všechny kraje v ČR vytvořte nové shapefily s kategorizací obcí podle velikosti.

Kategorie: - Malé obce: $<$ 500 obyvatel - Střední obce: 500-5000 obyvatel - Velké obce: $>$ 5000 obyvatel

Postup:

```
import arcpy
import os

# Nastavení
arcpy.env.workspace = r"C:\TEMP\Python_GIS\Lekce_14"
arcpy.env.overwriteOutput = True

# Výstupní složka
output_folder = r"C:\TEMP\Python_GIS\Lekce_14\kategorie_obci"
if not os.path.exists(output_folder):
```

```
    os.makedirs(output_folder)

print("=" * 60)
print("KATEGORIZACE OBCÍ PODLE KRAJŮ")
print("=" * 60)

# Data
obce = "obce_LK.shp"  # nebo obecnější data ze všech krajů
kraje = "VUSC.shp"

# Vytvoření vrstvy krajů
kraje_lyr = arcpy.management.MakeFeatureLayer(kraje, "kraje_layer")

# Získání seznamu krajů
kraje_list = []
with arcpy.da.SearchCursor(kraje_lyr, ["NAZEV_VUSC", "KOD_VUSC"]) as cursor:
    for row in cursor:
        kraje_list.append((row[0], row[1]))

print(f"\nNalezeno krajů: {len(kraje_list)}")

# Zpracování každého kraje
for nazev_kraje, kod_kraje in kraje_list:
    print(f"\n{'='*60}")
    print(f"Zpracovávám: {nazev_kraje}")
    print(f"{'='*60}")

    try:
        # 1. Výběr obce v kraji (pomocí Spatial Join nebo Select by Location)
        obce_lyr = arcpy.management.MakeFeatureLayer(obce, "obce_layer")

        # Výběr kraje
        arcpy.management.SelectLayerByAttribute(
            kraje_lyr,
            "NEW_SELECTION",
            f"KOD_VUSC = {kod_kraje}"
        )

        # Výběr obcí v kraji
        arcpy.management.SelectLayerByLocation(
            obce_lyr,
            "WITHIN",
```

```
        kraje_lyr
    )

    # Zjištění počtu obcí v kraji
    count = int(arcpy.management.GetCount(obce_lyr).getOutput(0))
    print(f"  Obcí v kraji: {count}")

    if count == 0:
        print("  Přeskakuji - žádné obce")
        continue

    # 2. Vytvoření shapefilů pro jednotlivé kategorie

    # Malé obce (< 500)
    arcpy.management.SelectLayerByAttribute(
        obce_lyr,
        "NEW_SELECTION",
        "OBYVATEL < 500"
    )
    count_male = int(arcpy.management.GetCount(obce_lyr).getOutput(0))
    if count_male > 0:
        output_male = os.path.join(output_folder,
                                   f"obce_{nazev_kraje}_male.shp")
        arcpy.management.CopyFeatures(obce_lyr, output_male)
        print(f"  Malé obce: {count_male} → {output_male}")

    # Střední obce (500-5000)
    arcpy.management.SelectLayerByLocation(
        obce_lyr,
        "WITHIN",
        kraje_lyr
    )
    arcpy.management.SelectLayerByAttribute(
        obce_lyr,
        "NEW_SELECTION",
        "OBYVATEL >= 500 AND OBYVATEL <= 5000"
    )
    count_stredni = int(arcpy.management.GetCount(obce_lyr).getOutput(0))
    if count_stredni > 0:
        output_stredni = os.path.join(output_folder,
                                      f"obce_{nazev_kraje}_stredni.shp")
        arcpy.management.CopyFeatures(obce_lyr, output_stredni)
```

```
            print(f"  Střední obce: {count_stredni} → {output_stredni}")

        # Velké obce (> 5000)
        arcpy.management.SelectLayerByLocation(
            obce_lyr,
            "WITHIN",
            kraje_lyr
        )
        arcpy.management.SelectLayerByAttribute(
            obce_lyr,
            "NEW_SELECTION",
            "OBYVATEL > 5000"
        )
        count_velke = int(arcpy.management.GetCount(obce_lyr).getOutput(0))
        if count_velke > 0:
            output_velke = os.path.join(output_folder,
                                        f"obce_{nazev_kraje}_velke.shp")
            arcpy.management.CopyFeatures(obce_lyr, output_velke)
            print(f"  Velké obce: {count_velke} → {output_velke}")

        print(f"  OK - Kraj zpracován")

    except Exception as e:
        print(f"  CHYBA: {str(e)}")

    finally:
        # Vyčištění výběrů
        arcpy.management.SelectLayerByAttribute(obce_lyr, "CLEAR_SELECTION")
        arcpy.management.SelectLayerByAttribute(kraje_lyr, "CLEAR_SELECTION")

# Smazání vrstev
arcpy.management.Delete(obce_lyr)
arcpy.management.Delete(kraje_lyr)

print("\n" + "=" * 60)
print("DÁVKOVÉ ZPRACOVÁNÍ DOKONČENO")
print("=" * 60)
print(f"Výstupní složka: {output_folder}")
```

---

8 Shrnutí

8.1 Co jsme se naučili

Práce s vrstvami:

- `MakeFeatureLayer()` - vytváření vrstev v paměti
- `MakeTableView()` - vytváření tabulkových pohledů
- `SelectLayerByAttribute()` - SQL dotazy na vrstvy
- `Delete()` - mazání vrstev (odemknutí dat)

Práce s poli:

- `AddField()` - přidávání polí do tabulky
- `DeleteField()` - mazání polí
- `CalculateField()` - výpočet hodnot polí
  - Konstanty: `100`, `"text"`
  - Odkaz na pole: `!NAZEV!`
  - Geometrie: `!shape.area!`, `!shape.length!`
  - Python výrazy: `.upper()`, `.lower()`, matematika

Propojování tabulek:

- `AddJoin()` - dočasné propojení (v rámci vrstvy)
- `RemoveJoin()` - odstranění dočasného propojení
- `JoinField()` - trvalé připojení polí (mění zdrojová data!)
- Prefixy u připojených polí: `!TabulkaA.pole!`

Sumarizace:

- `Statistics()` - agregační funkce (SUM, MEAN, MIN, MAX, COUNT)
- `case_field` - sumarizace po skupinách
- Kombinace více funkcí v jednom volání
- Připojení výsledků zpět k původní tabulce

SQL syntaxe v ArcGIS:

- Porovnání: `=`, `<>`, `>`, `<`, `>=`, `<=`
- Vzory: `LIKE '%text%'`
- Logika: `AND`, `OR`, `NOT`
- Seznamy: `IN (val1, val2, ...)`

8.2 Klíčové koncepty

1. **Vrstvy vs. datové sady** - vrstva je dočasný pohled v paměti, datová sada je fyzický soubor
2. **Dočasné vs. trvalé propojení** - AddJoin je dočasné (v rámci vrstvy), JoinField je trvalé (mění data)
3. **Uvozovky v CalculateField** - vnější `'` pro celý výraz, vnitřní `"` pro textové konstanty
4. **Prefixy u připojených polí** - `!NazevTabulky.NazevPole!` pro jednoznačnou identifikaci
5. **Statistics vytváří novou tabulku** - výsledek je samostatná tabulka, ne modifikace původní
6. **Zamykání dat vrstvami** - MakeFeatureLayer zamkne data, Delete odemkne

---

9 Domácí úkol

9.1 Varianta A (základní)

Zadání: Analýza silniční sítě podle okresů.

Pro každý okres spočítejte: 1. Celkovou délku silnic (v km) 2. Průměrnou délku silničních úseků 3. Počet silničních úseků

Data: `Silnice.shp`, `Okresy_VUSC.shp`

Postup: 1. Vypočítat délku silnic (LENGTH_KM) 2. Spatial Join - připojit kód okresu k silnicím 3. Statistics - sumarizovat podle okresů 4. Vyexportovat výsledky do CSV

Odevzdejte: Skript `analyza_silnic.py` + CSV s výsledky

9.2 Varianta B (střední)

Zadání: Kategorizace katastrů podle tvaru.

Vytvořte kategorie katastrů podle tvaru (PA_RATIO): - Kompaktní: PA_RATIO < 0.001 - Střední: 0.001 <= PA_RATIO < 0.002 - Členité: PA_RATIO >= 0.002

Pro každou kategorii: 1. Spočítejte počet katastrů 2. Spočítejte celkovou a průměrnou rozlohu 3. Vytvořte samostatný shapefile

Data: `katastry_Liberec.shp`

Odevzdejte: Python Notebook `kategorizace_katastru.ipynb` + 3 shapefily

9.3 Varianta C (pokročilá)

Zadání: Komplexní analýza krajinného pokryvu.

Pro každý okres v Libereckém kraji vytvořte report o krajinném pokryvu:

1. Pro každý typ pokryvu (Urban Atlas):
    - Celková plocha
    - Počet plošek
    - Průměrná velikost plošky
    - PA_RATIO (průměr, min, max)
2. Identifikujte dominantní typ pokryvu v každém okrese (největší plocha)
3. Spočítejte fragmentaci krajiny:
    - Počet plošek / celková plocha
4. Vytvořte HTML report s tabulkami a grafem (matplotlib)

Data: `UrbanAtlas_LK.shp`, `okresy.shp`

Bonusově: - Graf: zastoupení typů pokryvu v každém okrese (stacked bar chart) - Mapa: okresy obarvené podle dominantního typu pokryvu

Odevzdejte: Notebook `analyza_krajiny.ipynb` + HTML report

---

10 Poznámky pro vyučujícího

10.1 Časový plán (90 min)

Čas	Obsah
0-5 min	Úvod, rozdíl vrstvy vs. datová sada
5-20 min	Úlohy 1-4 (MakeFeatureLayer, Select)
20-40 min	Úlohy 5-11 (AddField, CalculateField)
40-60 min	Úlohy 12-16 (Join operace)
60-75 min	Úlohy 17-20 (Statistics)
75-85 min	Úloha 21 v Notebooku - struktura řešení
85-90 min	Shrnutí, domácí úkol

10.2 Klíčové momenty

10.2.1 Úlohy 1-4 (5-20 min):

- **ZDŮRAZNIT:** Rozdíl mezi vrstvou (v paměti) a datovou sadou (na disku)
- **UKÁZAT:** MakeFeatureLayer krok za krokem
- **DŮLEŽITÉ:** Zamykání dat - vrstva zamkne datovou sadu
- **PRAKTICKÉ:** SQL syntaxe v ArcGIS (LIKE, AND, OR)
- **VIZUÁLNÍ:** Kontrola výběrů v ArcGIS Pro

10.2.2 Úlohy 5-11 (20-40 min):

- **ZAČÍT:** Jednoduchý příklad s AddField
- **ZDŮRAZNIT:** Uvozovky v CalculateField (`'"text"'`)
- **UKÁZAT:** Odkaz na pole pomocí `!POLE!`
- **PRAKTICKÉ:** shape.area a shape.length
- **TIP:** Převody jednotek (dělení 1000000 pro km^2)

10.2.3 Úlohy 12-16 (40-60 min):

- **UKÁZAT:** AddJoin krok za krokem
- **DŮLEŽITÉ:** Prefixy u připojených polí
- **ZDŮRAZNIT:** Rozdíl mezi AddJoin (dočasné) a JoinField (trvalé)
- **POZOR:** JoinField mění původní data!
- **PRAKTICKÉ:** Kdy použít který typ propojení

10.2.4 Úlohy 17-20 (60-75 min):

- **UKÁZAT:** Statistics bez case_field
- **POROVNAT:** Statistics s case_field
- **DŮLEŽITÉ:** Statistics vytváří NOVOU tabulku
- **PRAKTICKÉ:** Kombinace Statistics + JoinField
- **TIP:** Použití sql_clause pro řazení výsledků

10.2.5 Úloha 21 (75-85 min):

- **PŘEPNOUT:** Do Python Notebooku
- **UKÁZAT:** Strukturu komplexní analýzy (komentáře, výpisy průběhu)
- **ZDŮRAZNIT:** Kombinace technik z celé lekce
- **NECHAT:** Studenty adaptovat kód pro vlastní analýzu

10.3 Rizika

1. **Studenti zapomínají vytvořit vrstvu před Select**
   - Řešení: Zdůraznit, že Select pracuje s vrstvami, ne datovými sadami
2. **Špatné uvozovky v CalculateField**
   - Řešení: Napsat na tabuli vzor: `'"textová_konstanta"'`
   - Ukázat chybovou hlášku a správné řešení
3. **Nesprávné názvy polí v AddJoin**
   - Řešení: Ukázat ListFields() pro výpis správných názvů s prefixy
4. **Zapomínají smazat vrstvy (zamčená data)**
   - Řešení: Důsledně ukazovat Delete() na konci každé sekce
5. **Nedostatek času na Notebook úlohy**
   - Řešení: Úloha 21 jen struktura, úloha 22 jako domácí úkol
6. **SQL syntaxe - studenti míchají Python a SQL**
   - Řešení: Zdůraznit rozdíl, ukázat příklady

10.4 Tipy

- **Vizuální kontrola:** Po každé úloze zkontrolovat výsledek v atributové tabulce
- **Data:** Mít připravená data, otestovat všechny skripty předem
- **ListFields():** Používat často pro kontrolu názvů polí
- **GetCount():** Používat pro kontrolu počtu vybraných prvků
- **Cheatsheet:** Rozdat vytištěný cheatsheet se SQL syntaxí
- **Propojení s GUI:** Ukázat, jak se operace dělají v GUI vs. Python

10.5 Běžné chyby studentů

```
# 1. Zapomínají vytvořit vrstvu před Select
arcpy.management.SelectLayerByAttribute("shapefile.shp", ...)  # CHYBA!
# Správně:
lyr = arcpy.management.MakeFeatureLayer("shapefile.shp", "layer")
arcpy.management.SelectLayerByAttribute(lyr, ...)

# 2. Špatné uvozovky v CalculateField
arcpy.management.CalculateField(tbl, "POLE", "text")  # CHYBA!
# Správně:
arcpy.management.CalculateField(tbl, "POLE", '"text"')

# 3. Zapomínají prefix u připojených polí
arcpy.management.CalculateField(lyr, "VYSL", '!pole1! * !pole2!')  # CHYBA!
# Správně:
arcpy.management.CalculateField(lyr, "VYSL", '!tabulka1.pole1! * !tabulka2.pole2!')

# 4. Nesprávná jednotka u shape.area
# shape.area je v m², ne v km²!
arcpy.management.CalculateField(tbl, "AREA", '!shape.area!')  # v m²
# Pro km²:
arcpy.management.CalculateField(tbl, "AREA_KM", '!shape.area!/1000000')

# 5. Zapomínají PYTHON3 u CalculateField s Python výrazy
arcpy.management.CalculateField(tbl, "POLE", '!TEXT!.upper()')  # CHYBA!
# Správně:
arcpy.management.CalculateField(tbl, "POLE", '!TEXT!.upper()', "PYTHON3")
```

10.6 Materiály k přípravě

- ☐ Všechna data připravená a otestovaná
- ☐ SQL syntaxe cheatsheet (vytisknout)
- ☐ Diagram: vrstvy vs. datové sady
- ☐ Diagram: AddJoin vs. JoinField
- ☐ Tabulka: datové typy polí
- ☐ Vzorový Notebook pro úlohu 21
- ☐ Řešení všech domácích úkolů (A, B, C)
- ☐ Screenshot: ListFields() s prefixy

---

11 Cheatsheet - Atributové operace

11.1 Vrstvy

```
# Vytvoření vrstvy
lyr = arcpy.management.MakeFeatureLayer("data.shp", "layer_name")

# Tabulkový pohled
view = arcpy.management.MakeTableView("table.dbf", "view_name", "WHERE_CLAUSE")

# Smazání vrstvy (odemkne data)
arcpy.management.Delete(lyr)
```

11.2 Výběry

```
# Nový výběr
arcpy.management.SelectLayerByAttribute(
    lyr,
    "NEW_SELECTION",
    "POLE > 100 AND TEXT LIKE '%vzor%'"
)

# Zjištění počtu vybraných
count = int(arcpy.management.GetCount(lyr).getOutput(0))
```

```
# Vyčištění výběru
arcpy.management.SelectLayerByAttribute(lyr, "CLEAR_SELECTION")
```

11.3 Pole

```
# Přidání pole
arcpy.management.AddField(table, "POLE", "DOUBLE")

# Smazání pole
arcpy.management.DeleteField(table, "POLE")

# Výpočet - konstanta
arcpy.management.CalculateField(table, "POLE", 100)
arcpy.management.CalculateField(table, "TEXT", '"konstanta"')

# Výpočet - z jiného pole
arcpy.management.CalculateField(
    table,
    "VYSLEDEK",
    '!POLE1! + !POLE2!',
    "PYTHON3"
)

# Geometrie
arcpy.management.CalculateField(table, "AREA_KM", '!shape.area!/1000000', "PYTHON3")
arcpy.management.CalculateField(table, "LENGTH_KM", '!shape.length!/1000', "PYTHON3")
```

11.4 Propojení

```
# Dočasné propojení (v paměti)
lyr = arcpy.management.MakeFeatureLayer("data.shp", "layer")
arcpy.management.AddJoin(lyr, "KEY_FIELD", "table.dbf", "KEY_FIELD")
# Práce s připojenými poli: !table.field!
arcpy.management.RemoveJoin(lyr, "table")

# Trvalé připojení (mění data!)
arcpy.management.JoinField(
    "data.shp",
```

```
    "KEY_FIELD",
    "table.dbf",
    "KEY_FIELD",
    ["POLE1", "POLE2"]
)
```

11.5 Sumarizace

```
# Celá tabulka
arcpy.analysis.Statistics(
    "input.shp",
    "output_stats",
    [["POLE", "SUM"], ["POLE", "MEAN"]]
)

# Po skupinách
arcpy.analysis.Statistics(
    "input.shp",
    "output_stats",
    [["POLE", "SUM"]],
    "GROUP_FIELD"
)

# Funkce: SUM, MEAN, MIN, MAX, COUNT
```

11.6 SQL syntaxe

```
# Porovnání
"POLE = 100"
"POLE <> 100"
"POLE > 100"
"POLE <= 100"

# Rozsah
"POLE BETWEEN 10 AND 100"

# Text
"TEXT = 'Praha'"
```

```
"TEXT LIKE '%vzor%'"  # % = libovolné znaky

# Logika
"POLE1 > 100 AND POLE2 < 200"
"TEXT = 'A' OR TEXT = 'B'"
"NOT (POLE IS NULL)"

# Seznam
"POLE IN (1, 2, 3)"
```