## **Multnomah County Attribute Rules**

**Notes on Enable / Disable:** Since in SDE the Attribute Rules are configured / controlled by the “Data Owner”. Cartographers cannot update these rules on the fly. In addition, the rules cannot be suspended for an operation (see disabled) and then turned back on for subsequent actions. Such a process would involve all editors exiting work in the SDE environment, the data owner disabling the rule, then informing the editors that may resume. This is not an issue outside of the SDE environment, Standalone Geodatabases can be manipulated as needed, as they are a single editor environment.

**Notes on Update / Insert Trigger:** many rules that would normally be triggered by Insert and Update are only configured to be triggered by insert. If a rule cannot be enabled / disabled on the fly the flexibility is needed for editing. A value may need be overwritten after insert and not have it revert to an incorrect value through the “update” of the attribute rule.

**Notes in the “Intersect” Arcade Function:** The intersect may find multiple features on intersection (most likely when you are editing on the edge of a MapIndex), the “First” function grabs the feature that is listed first in the intersection array, this may not be the value that is preferred. Rules that include the intersect function have the “Insert” trigger active but will not be triggered by an “update”. This will ensure that editors can enter in the correct value if the intersect brings in an undesired one. (If someone finds a different function to use besides “first”, please let me know)

**Notes on Rules Upload:** Multnomah County is operating on Enterprise 10.8.1. The “Import Attribute Rules” GP tool works well loading rules that are included in arcade release 1.0.0, however the tool does not load Rules utilizing functions included in Arcade 10.5.0 (namely using “Count” to read the length of a string). Our workaround was to load the Arcade 1.0.0 Rules and then manually enter the 1.5.0 rules independently.

Each Rule is listed in Operational Order

## **Annotation Rules (100,200,400,2000):**

*\*Name:* **ProAnno0100\_MapNumber\_Populate\_Insert (Same Statement at each scale)**

*Field:* **MapNumber**

*Description:* **Populating the MapNumber for Annotation Page Query upon creation of feature**

*Trigger:* **Insert**

*Expression:*

**var fsMapIndex = FeatureSetByName($datastore, "Pro\_Mapindex", ["MapNumber"])**

**var fsMapIndexIntersect = Intersects(fsMapIndex, $feature)**

**var mapIndex = First(fsMapIndexIntersect)**

**if (mapIndex == null) return {"errorMessage": "No Map Index Feature Found for Attribute Population"}**

**return MapIndex.MapNumber**

*\*Name:* **ProAnno0100\_MapScale\_Populate\_Insert (Same Statement at each scale)**

*Field:* **MapScale**

*Description:* **Populating the MapScale for Annotation Page Query upon creation of feature**

*Trigger:* **Insert**

*Expression:*

**var fsMapIndex = FeatureSetByName($datastore, "Pro\_Mapindex", ["MapScale"])**

**var fsMapIndexIntersect = Intersects(fsMapIndex, $feature)**

**var mapIndex = First(fsMapIndexIntersect)**

**if (mapIndex == null) return {"errorMessage": "No Map Index Feature Found for Attribute Population"}**

**return right(MapIndex.MapScale,4)**

## **Reference and Cartographic Lines Rules:**

*\*Name:* **Pro\_CartoLines\_MapNumber\_Populate**

*Field:* **MapNumber**

*Description:* **Populating the MapNumber for Cartographic Lines Page Query upon creation of feature**

*Trigger:* **Insert**

*Expression:*

**var fsMapIndex = FeatureSetByName($datastore, "Pro\_Mapindex", ["MapNumber"])**

**var fsMapIndexIntersect = Intersects(fsMapIndex, $feature)**

**var mapIndex = First(fsMapIndexIntersect)**

**if (mapIndex == null) return {"errorMessage": "No Map Index Feature Found for Attribute Population"}**

**return MapIndex.MapNumber**

*\*Name:* **Pro\_CartoLines\_MapScale\_Populate**

*Field:* **MapScale**

*Description:* **Populating the MapScale for Cartographic Lines Query upon creation of feature**

*Trigger:* **Insert**

*Expression:*

**var fsMapIndex = FeatureSetByName($datastore, "Pro\_Mapindex", ["MapScale"])**

**var fsMapIndexIntersect = Intersects(fsMapIndex, $feature)**

**var mapIndex = First(fsMapIndexIntersect)**

**if (mapIndex == null) return {"errorMessage": "No Map Index Feature Found for Attribute Population"}**

**return right(MapIndex.MapScale,4)**

## **Condo\_Units Rules:**

*\*Name:* **New Condo Unit MapNumber Populate**

*Field:* **MAPINDEX**

*Description:* **Populating the Map Number For Condo Unit Features**

*Trigger:* **Insert, Update**

*Expression:*

**var fsMapIndex = FeatureSetByName($datastore, "Pro\_Mapindex", ["MapNumber"])**

**var fsMapIndexIntersect = Intersects(fsMapIndex, $feature)**

**var mapIndex = First(fsMapIndexIntersect)**

**if (mapIndex == null) return {"errorMessage": "No Map Index Feature Found for Attribute Population"}**

**return MapIndex.MapNumber**

*\*Name:* **Condo Unit MTL Concatenation**

*Field:* **MAPTAXLOT**

*Description:* **Concatenating the Map Taxlot Number with the Map Index and the Taxlot Number**

*Trigger:* **Insert**

*Expression:*

**var mapIdx = $feature.MAPINDEX**

**var mapSizeLength = COUNT($feature.MAPINDEX)**

**var mtlSpacer = when(mapSizeLength == 8, " ", mapSizeLength == 7, " ",mapSizeLength == 6, " ", mapSizeLength == 4, " ", "ERROR")**

**return concatenate(mapIdx + mtlSpacer + "-" + $feature.TAXLOT)**

## **SubAccounts Rules:**

*\*Name:* **Populating Taxlot Fields into SubAccount Fields MAPINDEX**

*Field:* **MAPINDEX**

*Description:* **Translating Taxlot Fields into the Child Subaccount Features - MAPINDEX**

*Trigger:* **Insert**

*Expression:*

**var fsMapIndex = FeatureSetByName($datastore, "Pro\_Mapindex", ["MapNumber"])**

**var fsMapIndexIntersect = Intersects(fsMapIndex, $feature)**

**var mapIndex = First(fsMapIndexIntersect)**

**if (mapIndex == null) return {"errorMessage": "No Map Index Feature Found for Attribute Population"}**

**return MapIndex.MapNumber**

*\*Name:* **Populating Taxlot Fields into SubAccount Fields TAXLOT**

*Field:* **TAXLOT**

*Description:* **Translating Taxlot Fields into the Child Subaccount Features - TAXLOT**

*Trigger:* **Insert**

*Expression:*

**var fsPro\_Taxlot = FeatureSetByName($datastore, "Pro\_Taxlot", ["Taxlot"])**

**var fsPro\_TaxlotIntersect = Intersects(fsPro\_Taxlot, $feature)**

**var pro\_Taxlot = First(fsPro\_TaxlotIntersect)**

**if (pro\_Taxlot == null) return {"errorMessage": "No Taxlot Feature Found for Attribute Population"}**

**return Pro\_Taxlot.Taxlot**

*\*Name:* **Subaccount MTL Calculation**

*Field:* **MAPTAXLOT**

*Description:* **Calculating MTL for subaccountrs, (MapNumber, Taxlot, SubType, and Subtype Number)**

*Trigger:* **Insert**

*Expression:*

**var mapIdx = $feature.MAPINDEX**

**var mapSizeLength = count(mapIdx)**

**var mtlSpacer = when(mapSizeLength == 8, " ", mapSizeLength == 7, " ",mapSizeLength == 6, " ", mapSizeLength == 4, " ", "ERROR")**

**var txl = text($feature.Taxlot,'00000')**

**return concatenate(mapIdx + mtlSpacer + "-" + txl + "-" + upper($feature.SUBTYPE) + $feature.SUBTYPE\_NUMBER)**

## **Taxlot Rules:**

*\*Name:* **Taxlots\_MapName\_Populate**

*Field:* **MapNumber**

*Description:* **Populate Map Number**

*Trigger:* **Insert**

*Expression:*

**var fsMapIndex = FeatureSetByName($datastore, "Pro\_Mapindex", ["MapNumber"])**

**var fsMapIndexIntersect = Intersects(fsMapIndex, $feature)**

**var mapIndex = First(fsMapIndexIntersect)**

**if (mapIndex == null) return {"errorMessage": "No Map Index Feature Found for Attribute Population"}**

**return MapIndex.MapNumber**

*\*Name:* **Taxlots\_County\_Populate**

*Field:* **County**

*Description:* **Populating the County Number (26) for the Taxlots feature class**

*Trigger:* **Insert, Update**

*Expression:*

**var county = '26'**

**return county**

*\*Name:* **Taxlots\_Township\_Populate**

*Field:* **Town**

*Description:* **Populating the Township Number for the Taxlots feature class**

*Trigger:* **Insert, Update**

*Expression:*

**var townshipPart = left($feature.MapNumber,1)**

**return townshipPart**

*\*Name:* **Taxlots\_TownshipPart\_Populate**

*Field:* **TownPart**

*Description:* **Populating the Township Part Number for the Taxlots feature class**

*Trigger:* **Insert, Update**

*Expression:*

**var twnPrt = '0'**

**return twnPrt**

*\*Name:* **Taxlots\_TownshipDirectrion\_Populate**

*Field:* **TownDir**

*Description:* **Populating the Township Direction for the Taxlots feature class**

*Trigger***: Insert, Update**

*Expression:*

**var twnDir = mid($feature.MapNumber,1,1)**

**return twnDir**

*\*Name:* **Taxlots\_Range\_Populate**

*Field:* **Range**

*Description:* **Populating the Range Number for the Taxlots feature class**

*Trigger:* **Insert, Update**

*Expression:*

**var rangeNumber = mid($feature.MapNumber,2,1)**

**return rangeNumber**

*\*Name:* **Taxlots\_RangePart\_Populate**

*Field:* **RangePart**

*Description:* **Populating the Range Part Number for the Taxlots feature class**

*Trigger:* **Insert, Update**

*Expression:*

**var rngPrt = '0'**

**return rngPrt**

*\*Name:* **Taxlots\_RangeDirection\_Populate**

*Field:* **RangeDir**

*Description:* **Populating the Range Direction for the Taxlots Feature Class**

*Trigger:* **Insert, Update**

*Expression:*

**var rngDir = mid($feature.MapNumber,3,1)**

**return rngDir**

*\*Name:* **Taxlots\_SectionNumber\_Populate**

*Field:* **SecNumber**

*Description:* **Populating the Section Number for the Taxlots Feature Class**

*Trigger:* **Insert, Update**

*Expression:*

**var mapNumLen = count($feature.MapNumber)  
var secNum = when(mapNumLen > 4, mid($feature.MapNumber,4,2), '0')**

**return secNum**

*\*Name:* **Taxlots\_QuarterValue\_Populate**

*Field:* **Qtr**

*Description:* **Populating the PLSS Quarter Value for the Taxlots feature Class**

*Trigger:* **Insert, Update**

*Expression:*

**var mapNumLen = count($feature.MapNumber)**

**var qtrVal = when(mapNumLen > 6, mid($feature.MapNumber,6,1), '0')**

**return qtrVal**

*\*Name:* **Taxlots\_QuarterQuarterValue\_Populate**

*Field:* **QtrQtr**

*Description:* **Populating the PLSS Quarter Quarter Value for the Taxlots feature Class**

*Trigger:* **Insert, Update**

*Expression:*

**var mapNumLen = count($feature.MapNumber)**

**var qtrQtrVal = when(mapNumLen > 7, mid($feature.MapNumber,7,1), '0')**

**return qtrQtrVal**

*\*Name:* **Taxlots\_PFTypeToTaxlotNONTL**

*Field:* **Taxlot**

*Description:* **Populating the Taxlot with PF Type if is not a Taxlot or Tract**

*Trigger:* **Insert, Update**

*Expression:*

**var pfTp = $feature.PF\_Type**

**if ($feature["Taxlot\_Type"] == 8) {  
return pfTP**

**}**

**else if ($feature["Taxlot\_Type"] == 81) {**

**return pfTP**

**}**

**else if ($feature["Taxlot\_Type"] == 82) {**

**return pfTP**

**}**

**else if ($feature["Taxlot\_Type"] == 14) {**

**return pfTP**

**}**

**else if ($feature["Taxlot\_Type"] == 17) {**

**return pfTP**

**}**

**else if ($feature["Taxlot\_Type"] == 21) {**

**return pfTP**

**}**

**else**

**return text($feature.Taxlot,'00000')**

*\*Name:* **Taxlots\_PFTaxlot\_Populate**

*Field:* **PF\_Type**

*Description:* **Populating various Taxlot Types for PFType Attribute**

*Trigger:* **Insert, Update**

*Expression:*

**if ($feature["Taxlot\_Type"] == 8) {**

**return "ROADS"**

**}**

**else if ($feature["Taxlot\_Type"] == 81) {**

**return "NONTL"**

**}**

**else if ($feature["Taxlot\_Type"] == 82) {**

**return "NONTL"**

**}**

**else if ($feature["Taxlot\_Type"] == 14) {**

**return "RAILS"**

**}**

**else if ($feature["Taxlot\_Type"] == 17) {**

**return "NONTL"**

**}**

**else if ($feature["Taxlot\_Type"] == 22) {**

**return "WATER"**

**}**

**else if ($feature["Taxlot\_Type"] == 221) {**

**return "WATER"**

**}**

*\*Name:* **Taxlots\_MapTaxlot\_Concatenate**

*Field:* **MapTaxlot**

*Description:* **Concatenating the MTL (Map Taxlot) Identifier from other feature attributes**

*Trigger:* **Insert, Update**

*Expression:*

**var mapIdx = $feature.MapNumber**

**var mapSizeLength = count(mapIdx)**

**var mtlSpacer = when(mapSizeLength == 8, " ", mapSizeLength == 7, " ",mapSizeLength == 6, " ", mapSizeLength == 4, " ", "ERROR")**

**var txl = text($feature.Taxlot,'00000')**

**if ($feature.PF\_SubType\_Num > 0) {**

**return concatenate(mapIdx + mtlSpacer + "-" + text(txl,'#####') + "-" + $feature.PF\_SubType +$feature.PF\_SubType\_Num)**

**}**

**else**

**return concatenate(mapIdx + mtlSpacer + "-" + text(txl,'#####'))**

*\*Name:* **Taxlots\_ORMapNumber\_Concatenate**

*Field:* **ORMapNum**

*Description:* **Concatenating the OR Map Number Identifier from other feature attributes**

*Trigger:* **Insert, Update**

*Expression:*

**var county = $feature.County**

**var twp = text($feature.Town,'00')**

**var twpP = text($feature.TownPart,'00')**

**var twpD = $feature.TownDir**

**var rng = text($feature.Range,'00')**

**var rngP = text($feature.RangePart,'00')**

**var rngD = text($feature.RangeDir,'00')**

**var secNum = text($feature.SecNumber, '00')**

**var qtrDes = $feature.Qtr**

**var qtrQtrDes = $feature.QtrQtr**

**return concatenate(county + twp + "." + twpP + twpD + rng + "." + rngP + rngD + secNum + qtrDes + qtrQtrDes + "--0000")**

*\*Name:* **Taxlots\_ORTaxlot\_Concatenate**

*Field:* **ORTaxlot**

*Description:* **Concatenating the OR Taxlot Identifier from other feature attributes**

*Trigger:* **Insert, Update**

*Expression:*

**var orMapNum = $feature.ORMapNum**

**var tl = text($feature.Taxlot,'00000')**

**return concatenate(orMapNum + text(tl,'#####'))**

## **Cancelled Number Rules: (May be Multnomah County Specific)**

*\*Name:* **Extract\_MapNumber\_From\_MTL**

*Field:* **MAPNUMBER**

*Description:* **Extracts the Map Number from the MapTaxlot Entry**

*Trigger:* **Insert, Update**

*Expression:*

**var mapNum = trim(left($feature.MTL,10))**

**return mapNum**

*\*Name:* **COUNT\_MapNumberLength\_Value**

Field: **MapNumLength**

*Description:* **Counts the Length of the Map Number and populates attribute**

*Trigger:* **Insert, Update**

*Expression:*

**var mapLen = count(trim($feature.MAPNUMBER))**

**return mapLen**

*\*Name:* **Extract\_Taxlot\_Value**

*Field:* **CancelledTL**

*Description:* **Extracts the Taxlot Value from the MTL Entry**

*Trigger:* **Insert, Update**

*Expression:*

**var canTL = number(mid($feature.MTL,11,5))**

**return canTL**

*\*Name:* **Extract\_Cancelled\_Number**

Field: **CanNumDisplay**

*Description:* **Extracts the Cancelled Number From the MTL Provided, this will account for cancelled improvement accounts**

*Expression:*

**var canTL = number(mid($feature.MTL,11,5))**

**var mtlCount = count($feature.MTL)**

**var canTLex1 = right($feature.MTL,3)**

**var canTLex2 = right($feature.MTL,4)**

**if ( count($feature["MTL"]) == 19) {**

**return concatenate(canTL + canTLex1)**

**}**

**else if ( count($feature["MTL"]) == 20) {**

**return concatenate(canTL + canTLex2)**

**}**

**else return canTLelse if ( count($feature["MTL"]) == 20) {**

**return concatenate(canTL + canTLex2)**

**}**

**else return canTL**