# MapThat – Project Global Variables

Project Global Variables allow you to utilise one MapThat Layer which references a group of unique records from the same source table, to be shown in separate MapThat Projects.

For example, you may have a spatial table which contains Land Parcels, and a field that defines the organisation that manages that land. Using a Project Global Variable you can now create a MapThat Layer which can be exposed in multiple Projects and only render the Land Parcels for the records associated to that Organisation / Project.

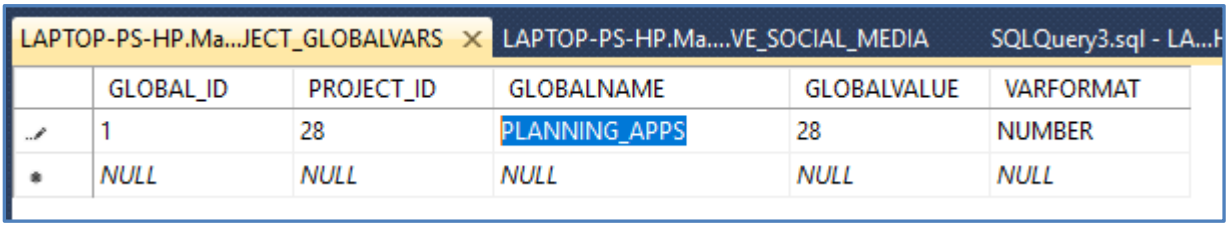
This dramatically reduces the number of MapThat Layers that you need and as your Projects grow you can simply re-use the same layer with minimal administration.

Below we will outline an example of how to implement a Project Global Variable using Planning Applications for different Local Authorities who share the same MapThat Instance and/or spatial data.

**Step 1 – PROJECT_GLOBALVAR table:**

In the MapThat Database, choose to Edit the table called **VE_PROJECT_GLOBALVAR,** adding a new record into the table and completing the values as below:

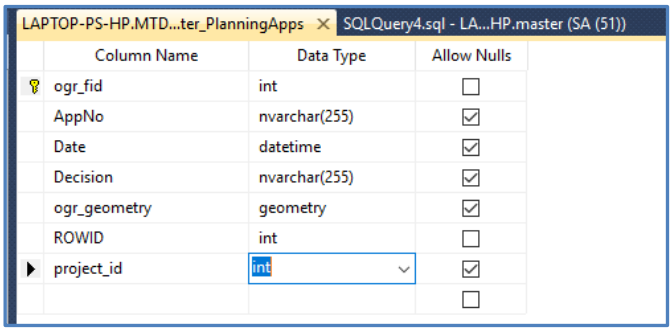
- Global_ID – any unique numeric value
- Project_ID – the ID for the first Project the layer is revealed in
- GlobalName – any value used as the link with your Spatial Table (e.g. Planning Apps)
- GlobalValue – the value used to differentiate which records should be revealed in which MapThat Projects
- VarFormat – defining a number or textual value as the reference in the Spatial Table




| | GLOBAL_ID | PROJECT_ID | GLOBALNAME | GLOBALVALUE | VARFORMAT |
|---|---|---|---|---|---|
| ✎ | 1 | 28 | PLANNING_APPS | 28 | NUMBER |
| * | NULL | NULL | NULL | NULL | NULL |

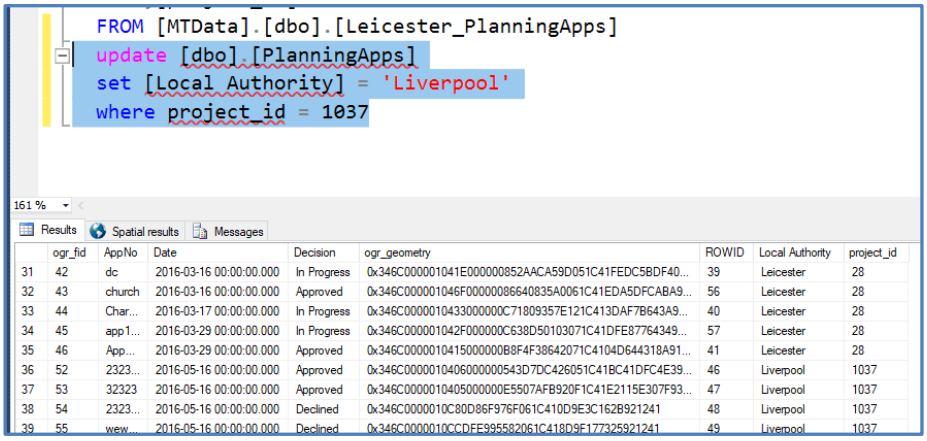
## Step 2 – Update your Spatial Table:

Next you will need to modify the Design of your source table so that there is a field that will store the Project_ID of the Project that each record will be exposed within. Here we have called this new field – Project_id and made the type numeric (int) as per the Project_GlobalVar.VarFormat.




| Column Name | Data Type | Allow Nulls |
|---|---|---|
| ogr_fid | int | ☐ |
| AppNo | nvarchar(255) | ☑ |
| Date | datetime | ☑ |
| Decision | nvarchar(255) | ☑ |
| ogr_geometry | geometry | ☑ |
| ROWID | int | ☐ |
| project_id | int | ☑ |
| | | ☐ |

In your Spatial Table now run an **Update Statement** to update this new field with the corresponding Project_ID. Here we have updated the Liverpool Planning Apps with an ID of 1037 and the Leicester Planning Apps with an ID of 28.



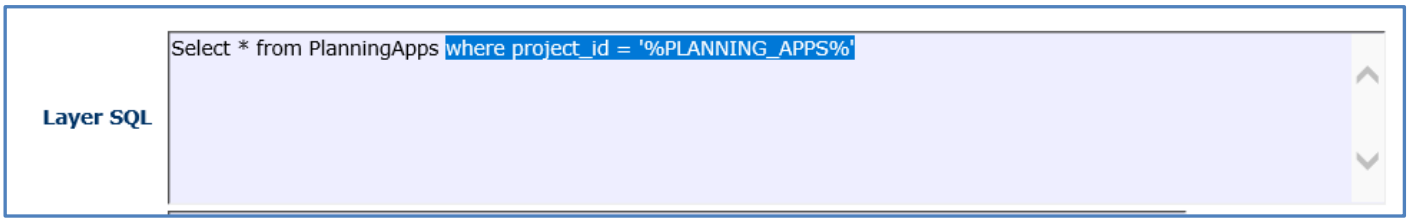
```
FROM [MTData].[dbo].[Leicester_PlanningApps]
update [dbo].[PlanningApps]
set [Local Authority] = 'Liverpool'
where project_id = 1037
```

161 % Results Spatial results Messages

| | ogr_fid | AppNo | Date | Decision | ogr_geometry | ROWID | Local Authority | project_id |
|---|---|---|---|---|---|---|---|---|
| 31 | 42 | dc | 2016-03-16 00:00:00.000 | In Progress | 0x346C000001041E000000852AACA59D051C41FEDC5BDF40... | 39 | Leicester | 28 |
| 32 | 43 | church | 2016-03-16 00:00:00.000 | Approved | 0x346C000001046F00000086640835A0061C41EDA5DFCABA9... | 56 | Leicester | 28 |
| 33 | 44 | Char... | 2016-03-17 00:00:00.000 | In Progress | 0x346C0000010433000000C71809357E121C413DAF7B643A9... | 40 | Leicester | 28 |
| 34 | 45 | app1... | 2016-03-29 00:00:00.000 | In Progress | 0x346C000001042F000000C638D50103071C41DFE87764349... | 57 | Leicester | 28 |
| 35 | 46 | App... | 2016-03-29 00:00:00.000 | Approved | 0x346C0000010415000000B8F4F38642071C4104D644318A91... | 41 | Leicester | 28 |
| 36 | 52 | 2323... | 2016-05-16 00:00:00.000 | Approved | 0x346C0000010406000000543D7DC426051C41BC41DFC4E39... | 46 | Liverpool | 1037 |
| 37 | 53 | 32323 | 2016-05-16 00:00:00.000 | Approved | 0x346C0000010405000000E5507AFB920F1C41E2115E307F93... | 47 | Liverpool | 1037 |
| 38 | 54 | 2323... | 2016-05-16 00:00:00.000 | Declined | 0x346C00000010C80D86F976F061C410D9E3C162B921241 | 48 | Liverpool | 1037 |
| 39 | 55 | wew... | 2016-05-16 00:00:00.000 | Declined | 0x346C00000010CCDFE995582061C418D9F177325921241 | 49 | Liverpool | 1037 |

## Step 3 – Create a MapThat Layer:

You can now either edit an existing or create a new MapThat Layer, which will now include a **WHERE statement** within the SQL clause. Below is the Planning Apps layer that I will use to reference my Global Variable.

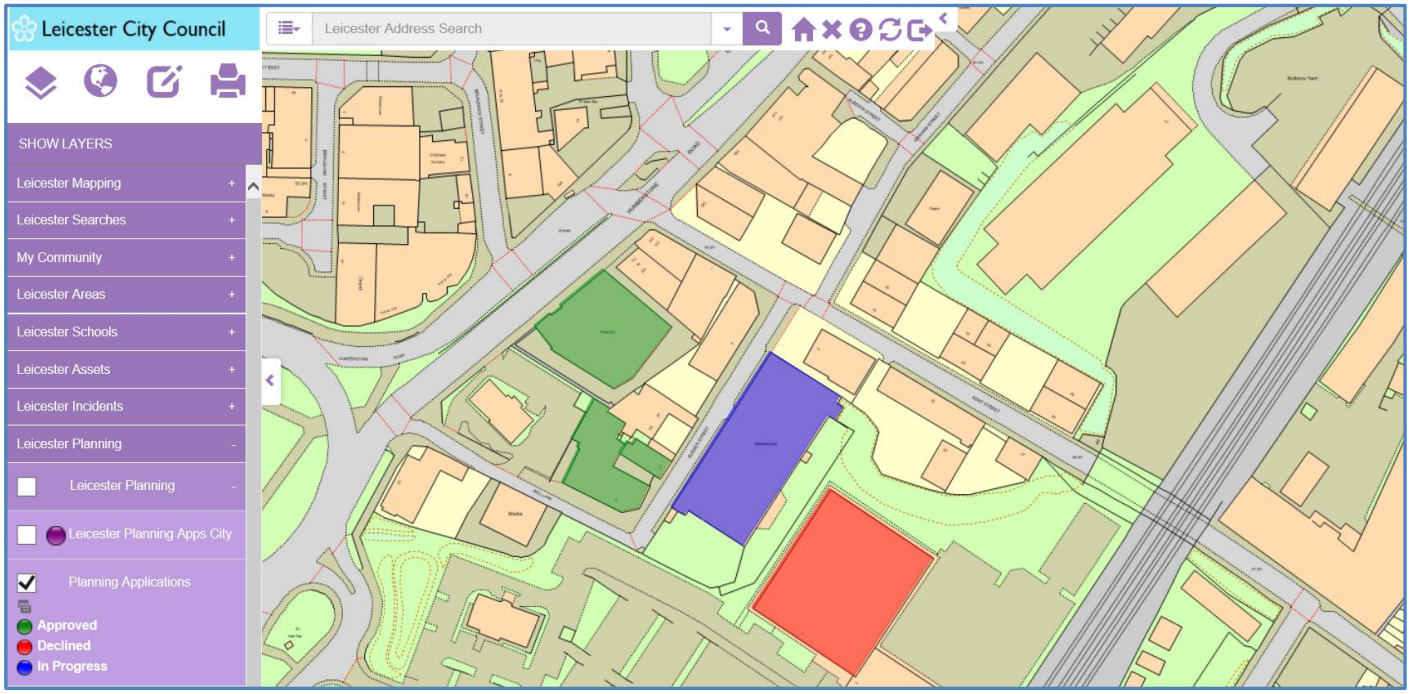



We now have a link between the MapThat Layer – Planning Apps – and the Spatial Table – Planning Apps - using the Global Variable called 'PLANNING_APPS'. Our Project_GlobalVar table will then decide which records from the source table are rendered in each MapThat Project.

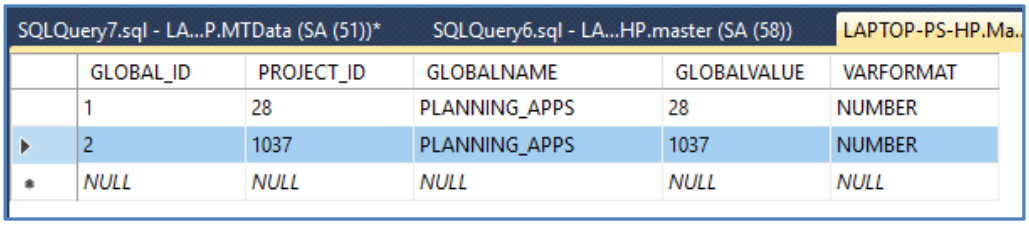
You should now choose to add/expose this Planning Apps layer to each of the Projects, e.g. Liverpool and Leicester.

**Test in MapThat**:

After opening the Leicester Project and choosing to display the Planning Apps layer, you will see that only the Planning Apps with the Project ID 28 are shown in the map.

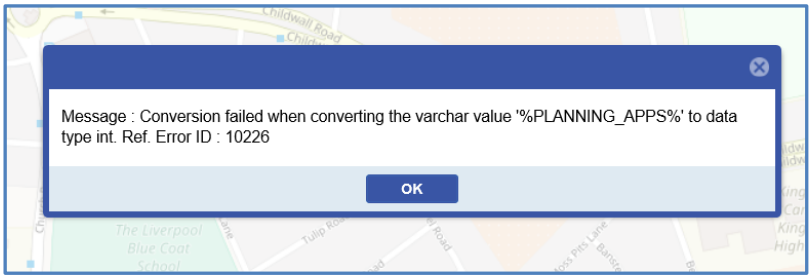


Before we can view the Planning Apps that only relate to our Liverpool Project, we need to add a record into the Project_GlobalVar table which holds the Project_ID for the Liverpool Project. As per the below we have now added a new record for 1037.


SQLQuery7.sql - LA...P.MTData (SA (51))* | SQLQuery6.sql - LA...HP.master (SA (58)) | LAPTOP-PS-HP.Ma..

| | GLOBAL_ID | PROJECT_ID | GLOBALNAME | GLOBALVALUE | VARFORMAT |
|---|---|---|---|---|---|
| | 1 | 28 | PLANNING_APPS | 28 | NUMBER |
| ▶ | 2 | 1037 | PLANNING_APPS | 1037 | NUMBER |
| * | NULL | NULL | NULL | NULL | NULL |

***Note -*** *If you do not add a record for each of your Projects, then the following message will appear when the user tries to display the layer in their project.*




If we now open the Liverpool Project and choose to display the Planning Apps layer, you will see that only the Planning Apps with the Project ID 1037 are shown in the map.