
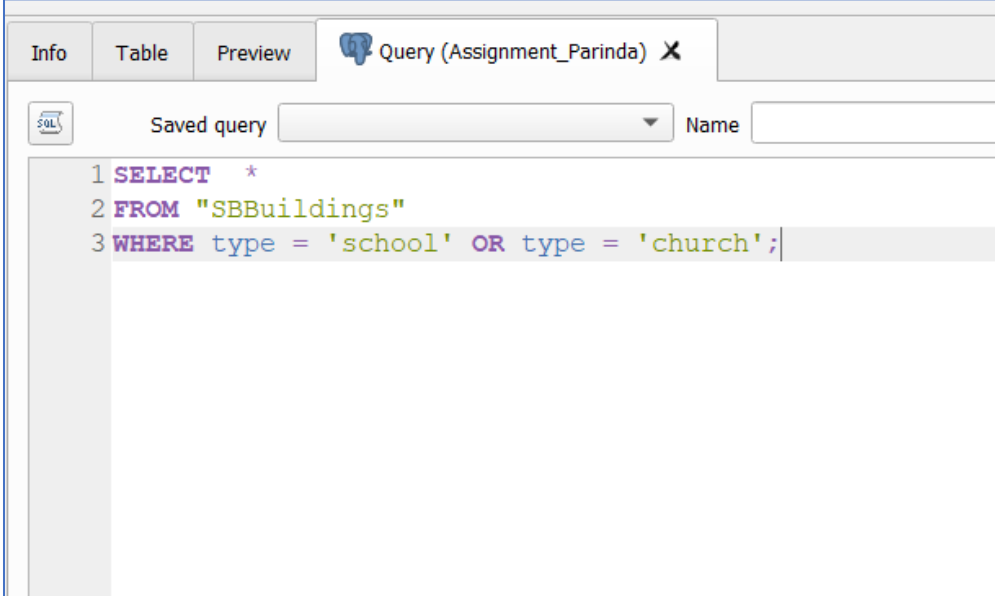


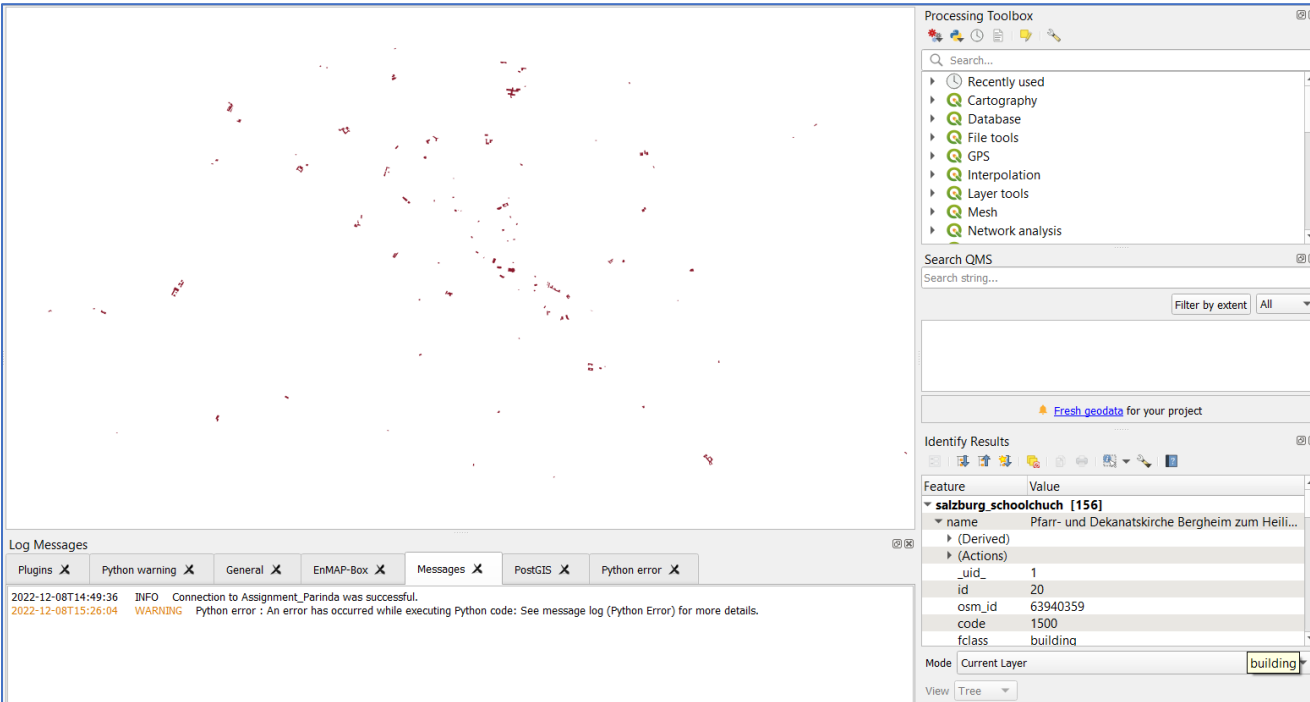
Assignment: Spatial Databases

SQL for spatial queries (1)

 **Task :** Repeat the query with all buildings, which are of type church and then repeat the query to create a layer, which contains churches and schools.



```
1 SELECT *
2 FROM "SBBuildings"
3 WHERE type = 'school' OR type = 'church';
```



The screenshot shows the QGIS interface. The main map area displays a spatial query result as a layer of red points. The Processing Toolbox on the right is open, showing various tool categories. The Identify Results panel is also open, displaying the details of a selected feature:

Feature	Value
salzburg_schoolchuch [156]	
name	Pfarr- und Dekanatskirche Bergheim zum Heili...
id	20
osm_id	63940359
code	1500
fclass	building

The Log Messages panel at the bottom shows the following messages:

```
2022-12-08T14:49:36 INFO Connection to Assignment_Parinda was successful.
2022-12-08T15:26:04 WARNING Python error : An error has occurred while executing Python code: See message log (Python Error) for more details.
```



Task : Extend the query and add a column that contains the calculated area of the houses.

```

Info  Table  Preview  Query (Assignment_Parinda) X
Saved query [dropdown] Name [input]
1 SELECT *, st_area(geom) AS area
2 FROM "SBBuildings"
3 WHERE type = 'house'
4 ORDER BY area ASC;
    
```

The screenshot displays the QGIS interface. The main map area shows a collection of small red and white building footprints. Below the map is a data table window titled "salzburg_house" with the following data:

uid	id	osm_id	code	fclass	name	type	area
1	31375	198177689	1500	building	NULL	house	14.3222684896...
2	34647	201933386	1500	building	NULL	house	14.5825378204...
3	21281	201741752	1500	building	NULL	house	18.3835049589...
4	11650	483007009	1500	building	NULL	house	19.7138754011...
5	1997	443813572	1500	building	NULL	house	23.9828127399...
6	37290	809903627	1500	building	NULL	house	28.7304451592...
7	16488	134510104	1500	building	NULL	house	35.1662812958...
8	39209	195910778	1500	building	NULL	house	35.3274337876...
9	16051	116703327	1500	building	NULL	house	35.5296881251...
10	16052	116703329	1500	building	NULL	house	35.5529939394...
11	16061	116703347	1500	building	NULL	house	35.5529965873...
12	16147	122687846	1500	building	NULL	house	36.0003073701...
13	16060	116703343	1500	building	NULL	house	36.0887964035...
14	16048	116703314	1500	building	NULL	house	36.0953327717...
15	16068	116703363	1500	building	NULL	house	36.1136708627...
16	16066	116703357	1500	building	NULL	house	36.1736380859...
17	16149	122687851	1500	building	NULL	house	36.2229647863...
18	16495	134510112	1500	building	NULL	house	36.3144066301...
19	16053	116703330	1500	building	NULL	house	36.4843199625...
20	16152	122687855	1500	building	NULL	house	36.5859165659...

The Processing Toolbox on the right shows the "Identify Results" panel for the "salzburg_house" layer, displaying details for a selected feature (id: 1, area: 14.3222684896...).



Task : Extend the query as follows to calculate the distance

Info Table Preview Query (Assignment_Parinda) X

Saved query Name


```

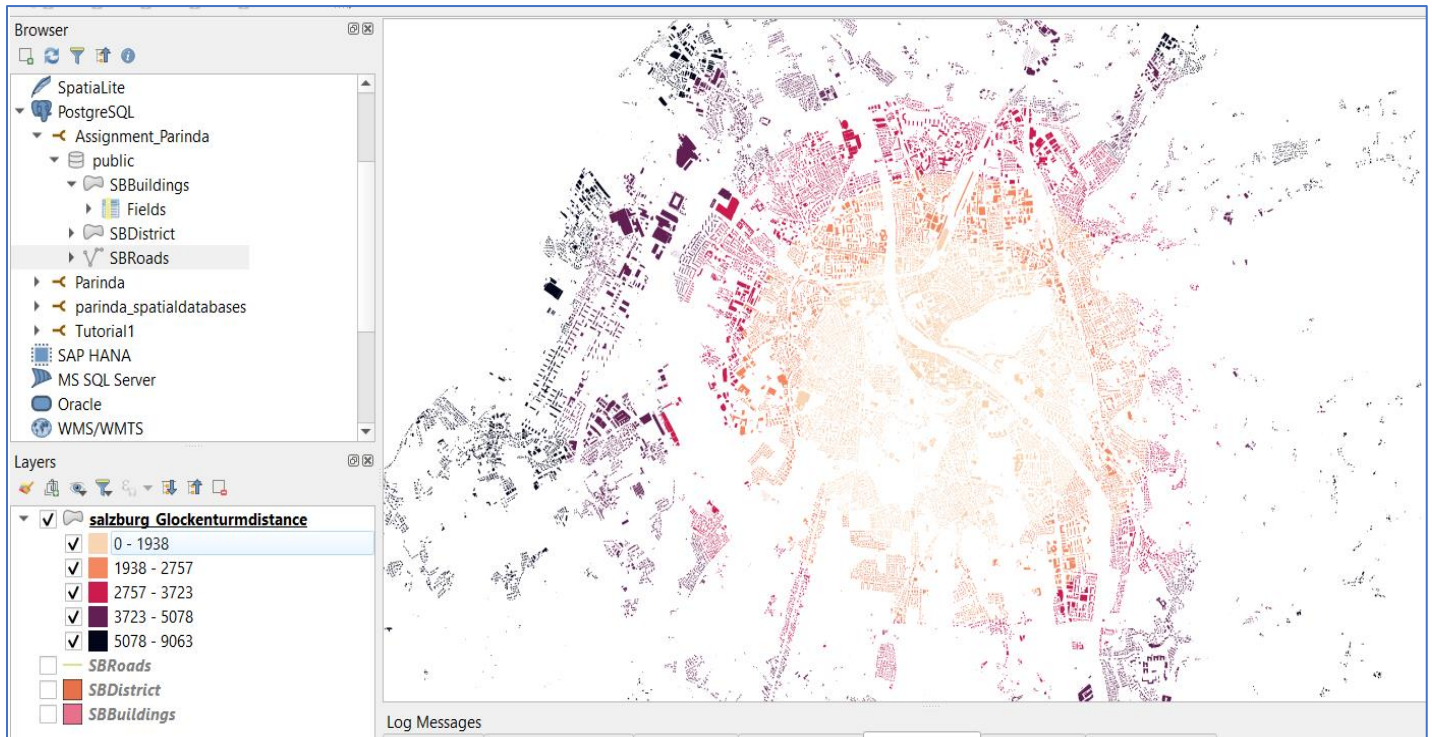
1 SELECT b.*, st_distance(b.geom, g.geom) AS distance
2 FROM "SBBuildings" AS b,
3 (SELECT geom FROM "SBBuildings" WHERE name = 'Glockenturm') AS g;
```


salzburg_Glockenturmdistance — Features Total: 41002, Filtered: 41002, Selected: 0

	id	osm_id	code	fclass	name	type	distance
1	1	24403125	1500	building	NULL	apartments	4242.44604419...
2	2	24403128	1500	building	Gummitechnik ...	commercial	4262.69323172...
3	3	24403133	1500	building	NULL	NULL	4147.69750101...
4	4	24403137	1500	building	NULL	apartments	4266.63698300...
5	5	24403141	1500	building	NULL	apartments	4351.20424624...
6	6	24403147	1500	building	NULL	detached	4054.90087244...
7	7	24403150	1500	building	Tapezierer Land...	retail	4009.79055456...
8	8	62643707	1500	building	NULL	NULL	5206.07689821...
9	9	62643708	1500	building	Spar	NULL	5339.98755286...
10	10	62643829	1500	building	Flüchtlings-Qua...	office	5189.00772588...
11	11	63522288	1500	building	NULL	commercial	5337.73304806...
12	12	63522289	1500	building	NULL	NULL	5204.93370815...
13	13	63576770	1500	building	Raiffeisen Lager...	retail	5205.16523559...
14	14	63576773	1500	building	Schiessl Kältete...	commercial	5354.09511636...
15	15	63576774	1500	building	NULL	NULL	5284.73118093...
16	16	63576775	1500	building	NULL	NULL	5214.94330191...
17	17	63576777	1500	building	NULL	industrial	5260.76256585...
18	18	63940357	1500	building	NULL	NULL	5187.91023277...
19	19	63940358	1500	building	Pfarrhof	NULL	5234.12166699...
20	20	63940359	1500	building	Pfarr- und Dek...	church	5250.19310374...

Show All Features

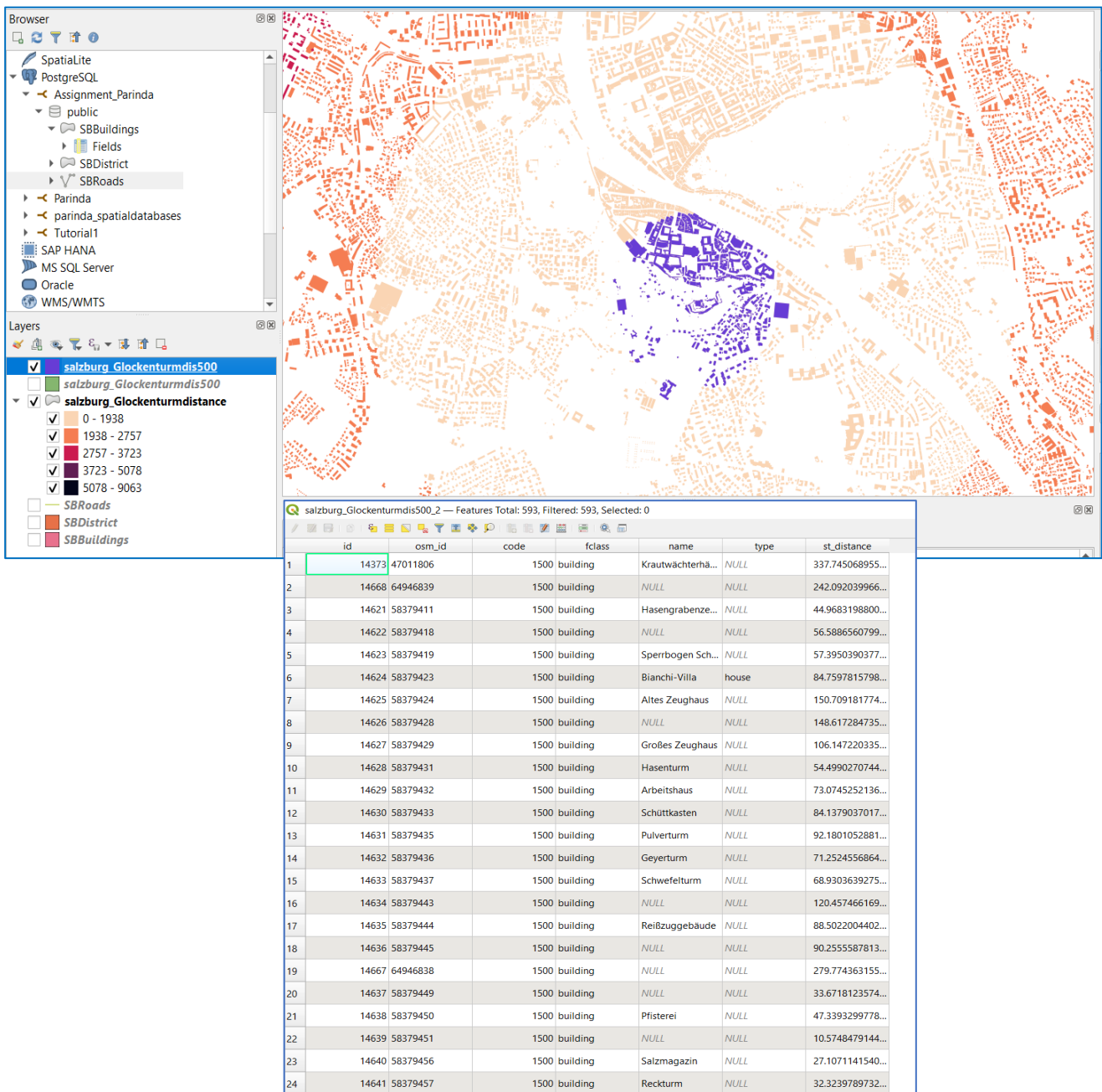
 **Task :** Add this result as new layer in the QGIS Project and style it accordingly (the styling aesthetics will not be graded).



 **Task :** Execute the following, modified query to select all buildings, which are within 500m to the fortress and visualise the result in QGIS.


```

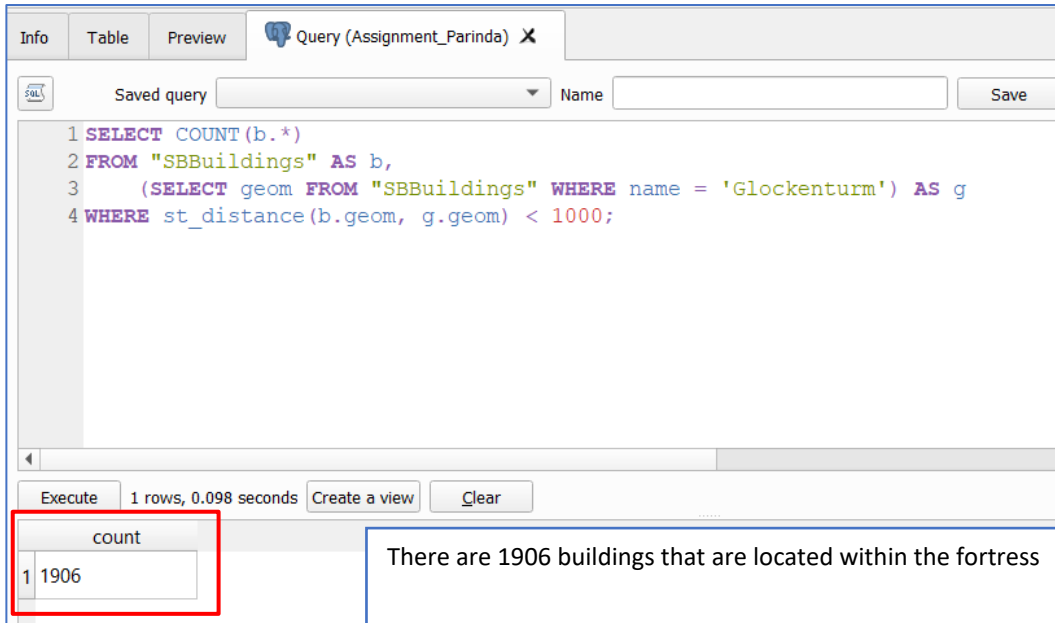
1 SELECT b.*
2 FROM "SBBuildings" AS b,
3 (SELECT geom FROM "SBBuildings" WHERE name = 'Glockenturm') AS g
4 WHERE st_distance(b.geom, g.geom) < 500;
    
```



The screenshot shows the QGIS interface with a map of Salzburg, Austria. The map displays buildings in orange and blue, with a central area highlighted in blue. The query window shows a table with columns: id, osm_id, code, fclass, name, type, st_distance.

id	osm_id	code	fclass	name	type	st_distance	
1	14373	47011806	1500	building	Krautwächterhä...	NULL	
2	14668	64946839	1500	building	NULL	242.092039966...	
3	14621	58379411	1500	building	Hasengrabenze...	44.9683198800...	
4	14622	58379418	1500	building	NULL	56.5886560799...	
5	14623	58379419	1500	building	Sperrbogen Sch...	57.3950390377...	
6	14624	58379423	1500	building	Bianchi-Villa	house	84.7597815798...
7	14625	58379424	1500	building	Altes Zeughaus	NULL	150.709181774...
8	14626	58379428	1500	building	NULL	NULL	148.617284735...
9	14627	58379429	1500	building	Großes Zeughaus	NULL	106.147220335...
10	14628	58379431	1500	building	Hasenturm	NULL	54.4990270744...
11	14629	58379432	1500	building	Arbeitshaus	NULL	73.0745252136...
12	14630	58379433	1500	building	Schüttkasten	NULL	84.1379037017...
13	14631	58379435	1500	building	Pulverturm	NULL	92.1801052881...
14	14632	58379436	1500	building	Geyerturm	NULL	71.2524556864...
15	14633	58379437	1500	building	Schwefelturm	NULL	68.9303639275...
16	14634	58379443	1500	building	NULL	NULL	120.457466169...
17	14635	58379444	1500	building	Reißzuggebäude	NULL	88.5022004402...
18	14636	58379445	1500	building	NULL	NULL	90.2555587813...
19	14667	64946838	1500	building	NULL	NULL	279.774363155...
20	14637	58379449	1500	building	NULL	NULL	33.6718123574...
21	14638	58379450	1500	building	Pfisterei	NULL	47.3393299778...
22	14639	58379451	1500	building	NULL	NULL	10.5748479144...
23	14640	58379456	1500	building	Salzmagazin	NULL	27.1071141540...
24	14641	58379457	1500	building	Reckturm	NULL	32.3239789732...

 **Task :** Unfortunately, an elephant walked through my screen when I did the screenshot of the last example to show how many buildings are within 1km distance. How many buildings are there with this condition?



The screenshot shows a SQL query editor with the following query:


```

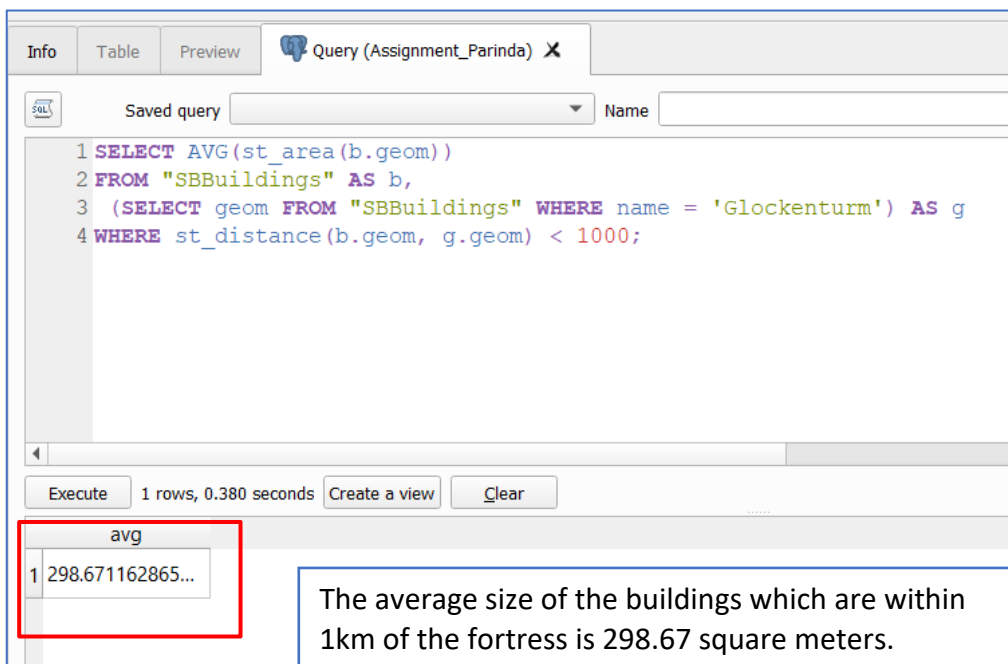
1 SELECT COUNT(b.*)
2 FROM "SBBuildings" AS b,
3     (SELECT geom FROM "SBBuildings" WHERE name = 'Glockenturm') AS g
4 WHERE st_distance(b.geom, g.geom) < 1000;
    
```

The execution status shows "1 rows, 0.098 seconds". The result table has one row with the column "count" and the value "1906".

count
1 1906

There are 1906 buildings that are located within the fortress

 **Task :** Of course it is also possible to combine spatial operations and filters. What is the average size of all buildings, which are within 1km distance to the fortress (including the fortress itself)? Make a screenshot, which shows the query and the result and explain in your own words how the query works.



The screenshot shows a SQL query editor with the following query:

```

1 SELECT AVG(st_area(b.geom))
2 FROM "SBBuildings" AS b,
3     (SELECT geom FROM "SBBuildings" WHERE name = 'Glockenturm') AS g
4 WHERE st_distance(b.geom, g.geom) < 1000;
    
```

The execution status shows "1 rows, 0.380 seconds". The result table has one row with the column "avg" and the value "298.671162865...".

avg
1 298.671162865...

The average size of the buildings which are within 1km of the fortress is 298.67 square meters.

**Answer :**

This query is executed by using average function (AVG) that calculates the average value of a numerical dataset that returns from the SELECT statement. In this query, we need to find an average of building size so we use a spatial function of area (st_area(<geomA>)) inside the average function. St_area function returns the area of the geometry of the table “b” (b.geom).

All the dataset in this query is “SBBuildings” table that was imported from vector layer. To specify which table to be selected, we use FROM command and use AS to assign a new name temporarily (“b”) to a table or even columns for making an easy writing of queries. However, in this query is also in the condition of the distance to the fortress so we need a second geometry column of the fortress, this is a sub query embedded within the WHERE (“Glockenturm”), and we set this subquery to a new short name “g”.

Now, the query has only an average size of building, we need to know *the average size of all buildings which are within 1km distance to the fortress* so we use spatial function of distance(st_distance(<geomA>, <geomB>)) which requires 2 geometry columns, this is why we need to do a sub query of geometry column of the fortress. In this query, the first geometry is location of buildings and the second one is location of the fortress, then the distance function will execute the distance between these 2 geometries that a condition is must be within 1km from the fortress (<1000)

The final result we will get the average size of the buildings which are within 1km of the fortress is 298.67 square meters.