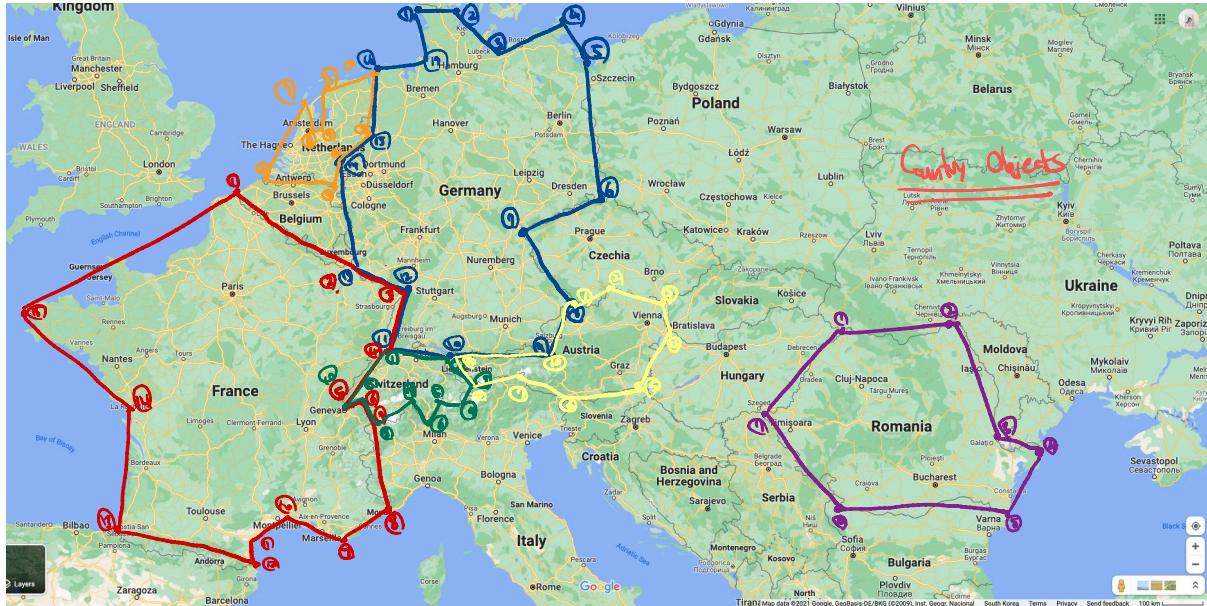


[공간데이터 관리 및 응용 프로젝트 I]

20161575 김연요

1. 실제 지도 좌표 데이터

1) 나라 데이터 (6 개)



유럽 국가 중 제가 이번 프로젝트에서 좌표 데이터로 활용한 여섯 국가는 위의 그림과 같습니다.

1. France (Red)

1. (51.054229, 2.339865)
2. (49.469402, 6.368522)
3. (48.966455, 8.232948)
4. (47.589749, 7.589084)
5. (46.143156, 5.964401)
6. (46.424675, 6.816491)
7. (45.924097, 7.043506)
8. (43.784369, 7.529033)
9. (43.278222, 5.356292)
10. (43.542885, 3.887426)
11. (43.113250, 3.094080)
12. (42.449260, 3.153489)
13. (43.369642, -1.789990)
14. (46.159733, -1.220800)
15. (48.509382, -4.761720)

2. Switzerland (Green)

1. (47.589749, 7.589084)
2. (47.496249, 9.565087)
3. (46.939417, 10.460778)
4. (46.234958, 10.125952)
5. (46.513577, 9.347988)
6. (45.835056, 9.015423)
7. (46.464512, 8.433434)
8. (45.924097, 7.043506)
9. (46.424675, 6.816491)
10. (46.143156, 5.964401)

3. Germany (Blue)

1. (54.881902, 8.660610)
2. (54.729941, 9.943801)
3. (53.914000, 11.339094)
4. (54.565974, 13.594696)
5. (53.706347, 14.249548)
6. (50.858786, 14.807385)

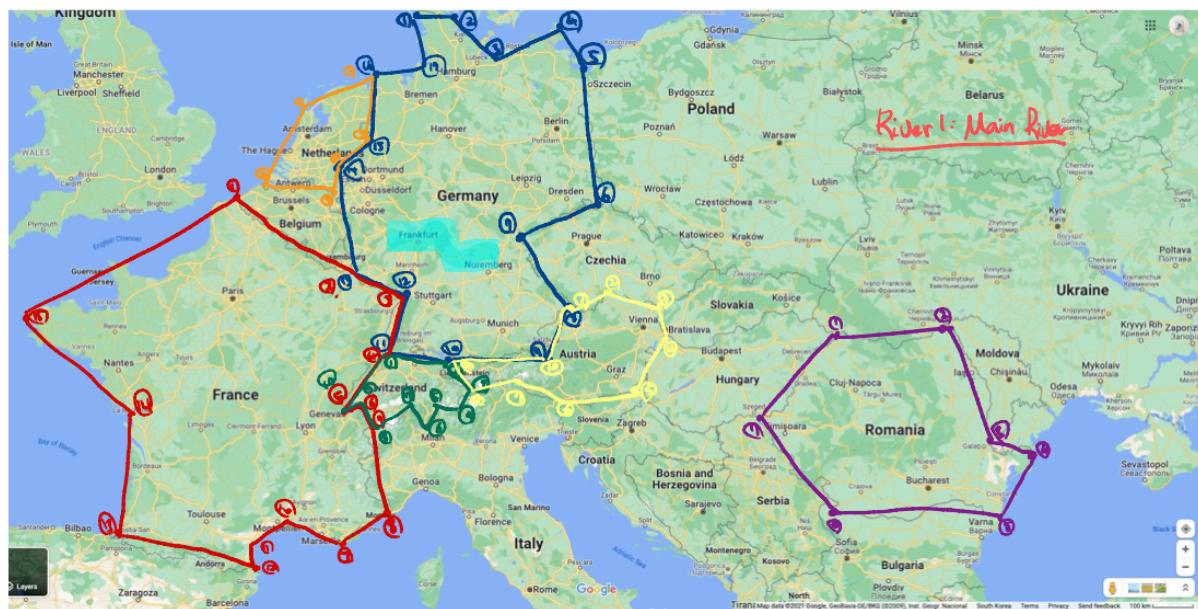
- 7. (50.273379, 12.054581)
 - 8. (48.761658, 13.816679)
 - 9. (47.491295, 13.010570)
 - 10. (47.496249, 9.565087)
 - 11. (47.589749, 7.589084)
 - 12. (48.966455, 8.232948)
 - 13. (49.469402, 6.368522)
 - 14. (51.830706, 5.984595)
 - 15. (52.446698, 6.979074)
 - 16. (53.405149, 7.016784)
 - 17. (53.856341, 8.597183)
4. Austria (Yellow)
- 1. (48.761658, 13.816679)
 - 2. (48.993386, 15.084363)
 - 3. (48.697393, 16.883462)
 - 4. (48.106544, 17.055717)
 - 5. (46.694588, 15.983913)
 - 6. (46.523662, 13.725469)
 - 7. (47.093491, 12.203891)
 - 8. (46.939417, 10.460778)
 - 9. (47.496249, 9.565087)
 - 10. (47.491295, 13.010570)
5. Romania (Purple)
- 1. (47.959995, 22.930931)
 - 2. (48.214695, 26.611841)
 - 3. (45.424993, 28.225357)
 - 4. (45.153307, 29.676849)
 - 5. (43.775358, 28.544951)
 - 6. (43.861832, 22.912095)
 - 7. (46.103132, 20.315388)
6. Netherland (Orange)
- 1. (53.213594, 5.589597)
 - 2. (53.405149, 7.016784)
 - 3. (52.446698, 6.979074)
 - 4. (51.830706, 5.984595)
 - 5. (51.113675, 5.817156)
 - 6. (51.364972, 3.387690)
 - 7. (52.932246, 4.773003)
 - 8. (52.358388, 4.983996)
 - 9. (52.329867, 5.113208)

위에 그림에 각각 나타낸 좌표들에 대한 좌표 정보를 정리해보았습니다.

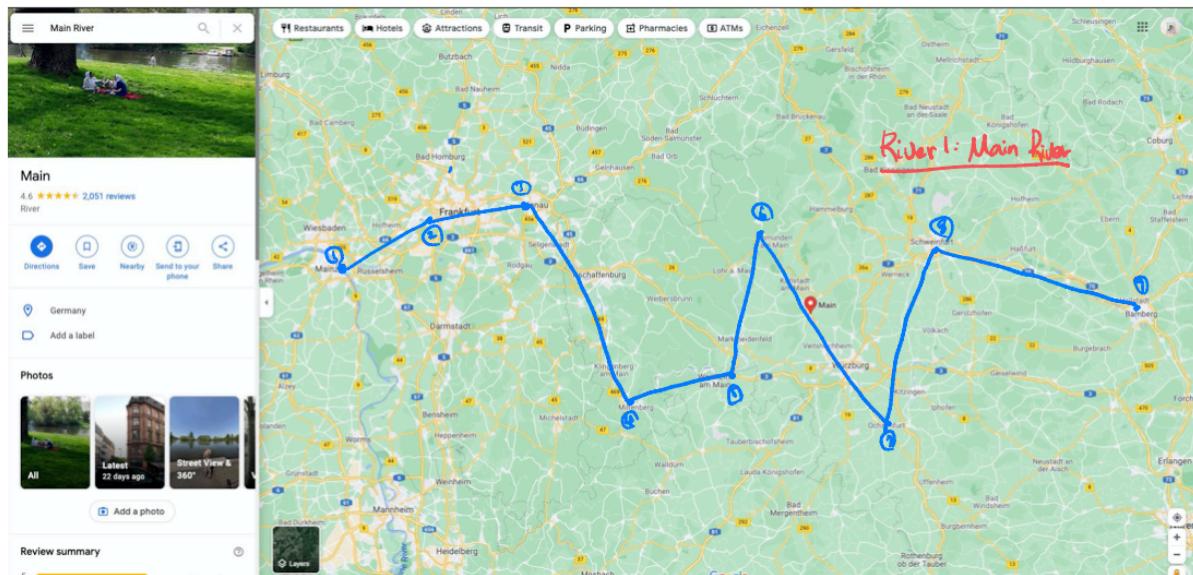
2) 강 데이터 (5 개)

이어서, 강으로 사용한 좌표들은 다음과 같습니다.

1. Main River



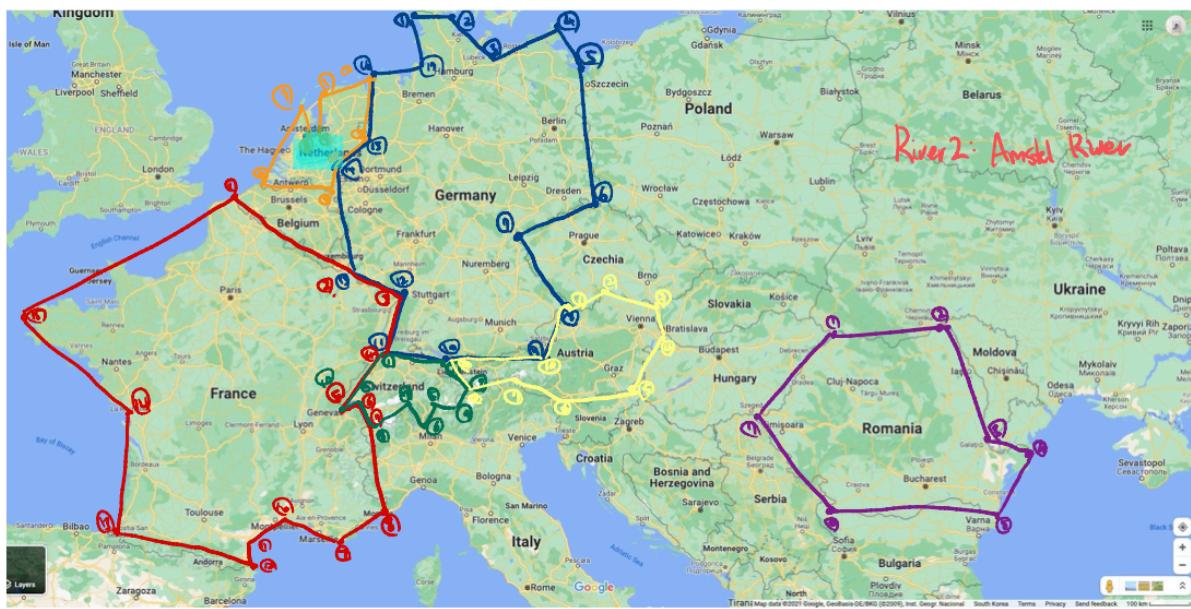
위 지도에서 Main River 를 푸른색 형광펜으로 나타내보았습니다. Germany 내부에 있기 때문에 contain/inside 관계를 얻을 수 있을 것이라고 생각합니다.



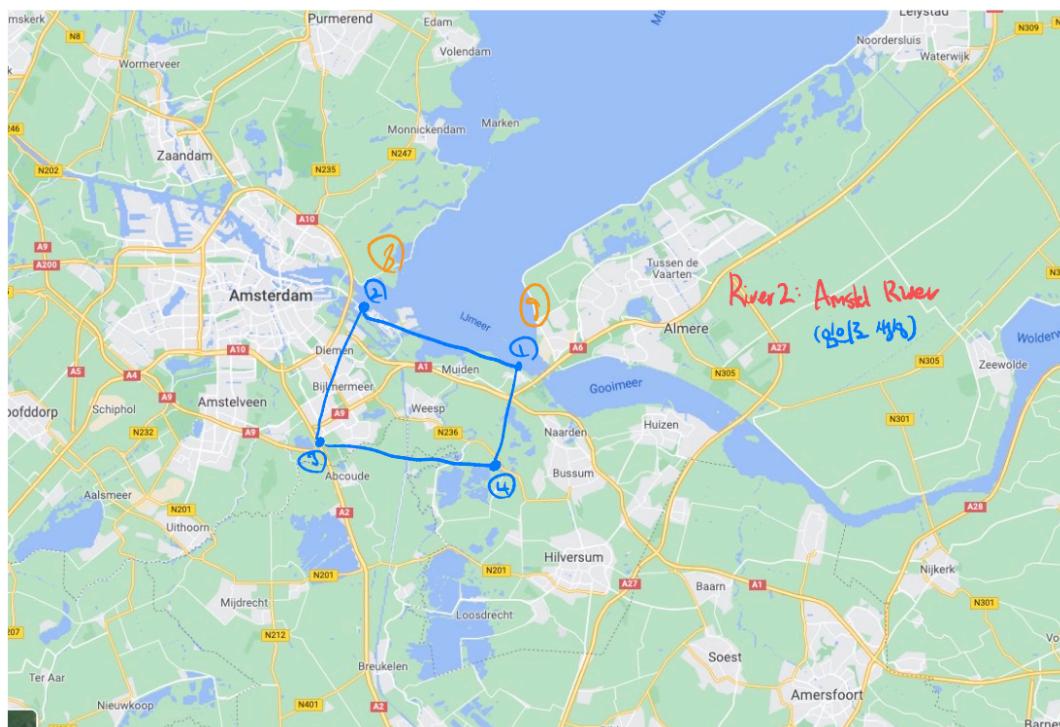
Main River 의 좌표를 정리해보았습니다.

1. (49.993928, 8.288695)
2. (50.099850, 8.558159)
3. (50.1270079, 8.901116)
4. (49.700620, 9.250788)
5. (49.77251, 9.516735)
6. (50.058439, 9.676245)
7. (49.668891, 10.096535)
8. (50.038423, 10.227997)
9. (49.906414, 10.863171)

2. Amstel River



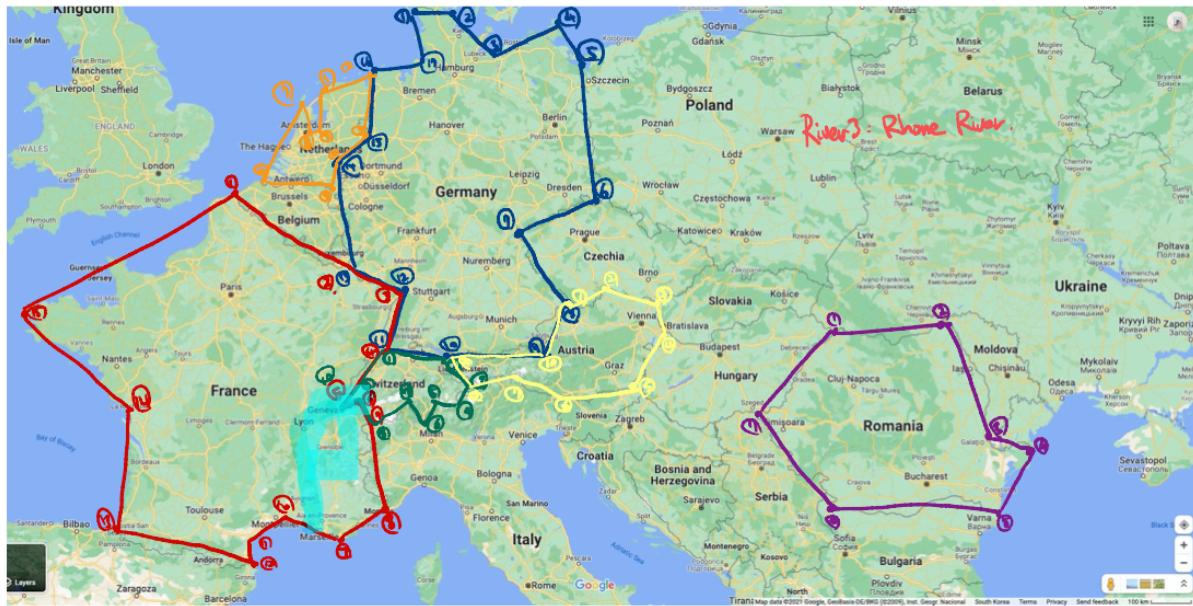
위 지도에서 Amstel River 를 푸른색 형광펜으로 나타내보았습니다. 잘 보이지는 않지만 주황색의 Netherland 내부에 있습니다.



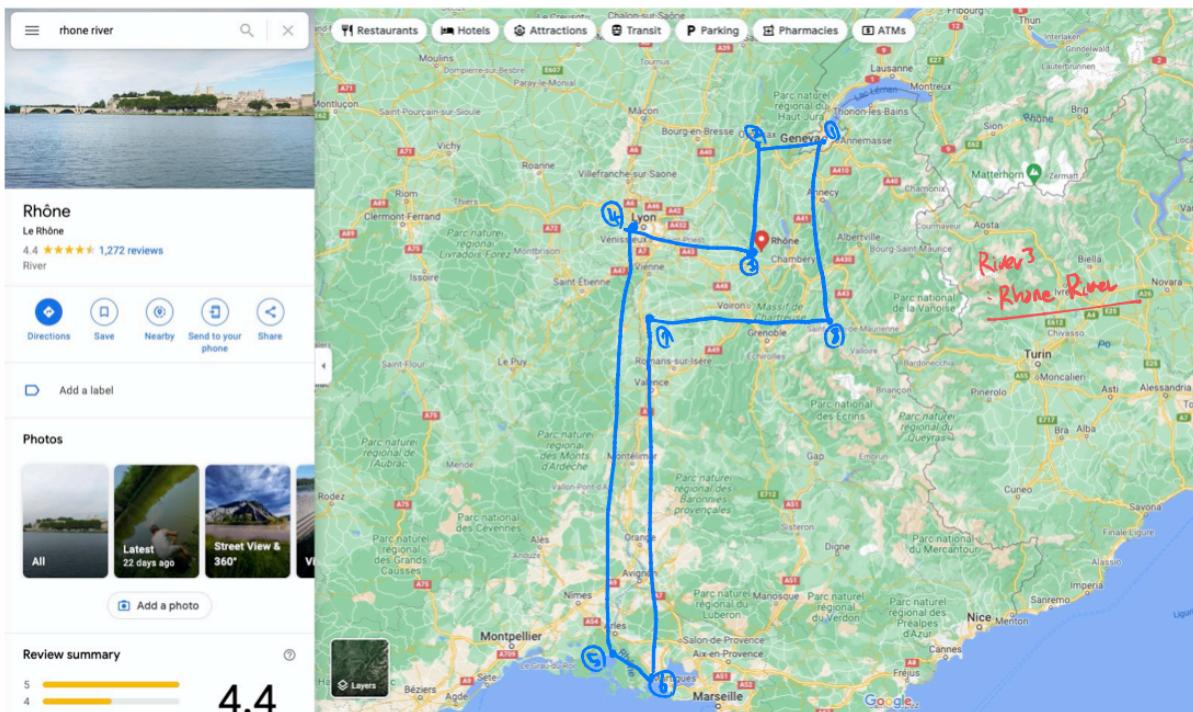
Amstel River 의 경우, 이후에 Cover/CoveredBy 관계를 확인할 수 있도록 임의의 좌표를 생성해서 정리했습니다. River 의 한 쪽 면이 Netherland (Country)의 경계선 중 하나와 닿도록 다음과 같이 설정했습니다.

1. (52.329867, 5.113208)
2. (52.358388, 4.983996)
3. (52.247382, 5.047002)
4. (52.296384, 4.900875)

3. Rhone River



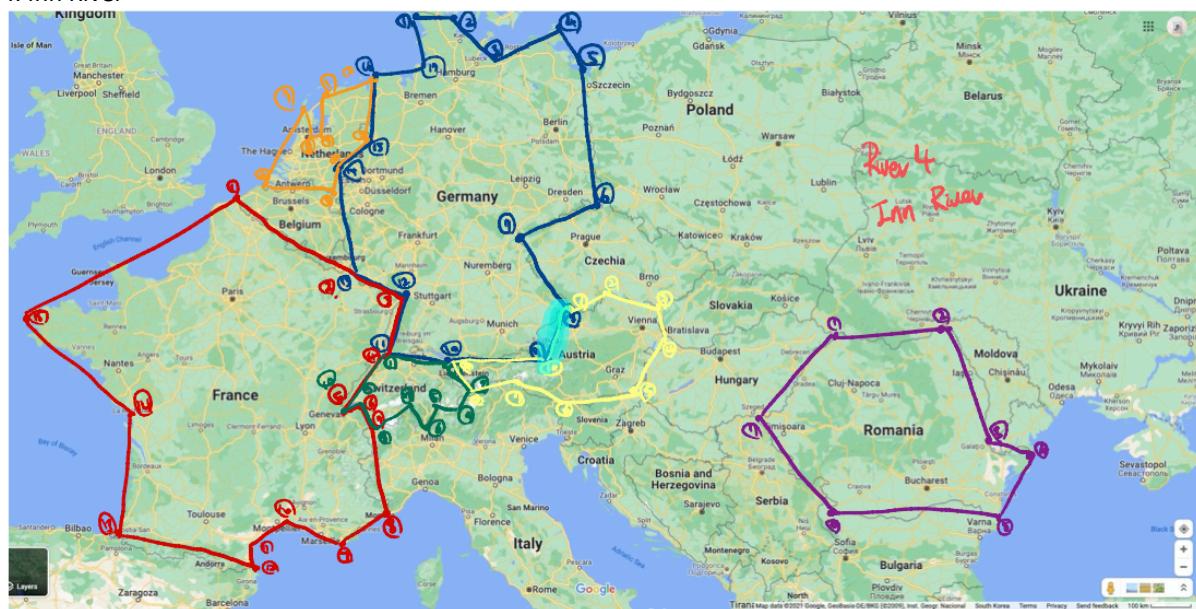
위 지도에서 Rhone River 를 푸른색 형광펜으로 나타내보았습니다. France 와 Switzerland 를 모두 통과하는 강입니다.



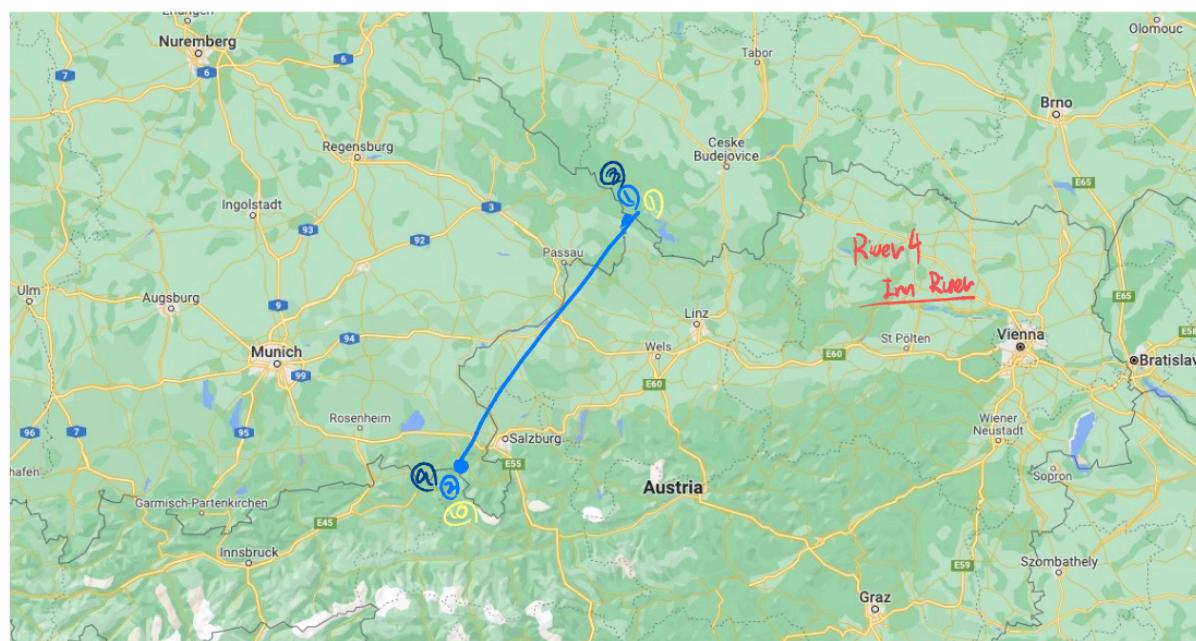
Rhone River 의 좌표를 정리해보았습니다. 두 나라를 overlapping 하는 것을 예측할 수 있습니다.

1. (46.237697, 6.249716)
2. (46.207309, 5.613591)
3. (45.627794, 5.664697)
4. (45.786606, 4.864900)
5. (43.624943, 4.678032)
6. (43.552600, 5.099138)
7. (45.343598, 5.016999)
8. (45.343598, 6.280317)

4. Inn River



위 지도에서 Inn River 를 푸른색 형광펜으로 나타내보았습니다. 강 중에서 Country 와 Touch 관계를 가질 수 있는 강을 찾기 위해 boundary 에 맞닿아 있는 Inn River 를 찾았습니다.



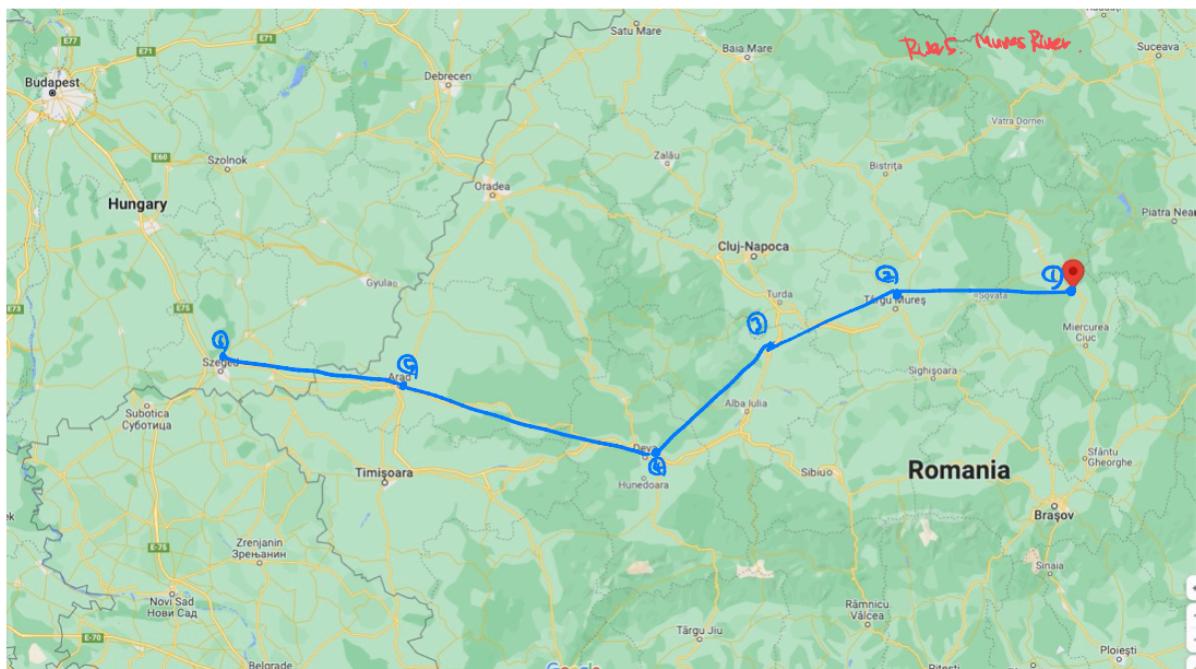
Inn River 의 좌표를 정리해보았습니다. Germany 와 Austria 의 경계선에서 흐르는 강이기 때문에, 이와 함께 설정했습니다.

1. (48.761658, 13.816679)
2. (47.491295, 13.010570)

5. Mures River



위 지도에서 Mures River 를 푸른색 형광펜으로 나타내보았습니다.



Mures River 의 좌표를 정리해보았습니다.

1. (46.621413, 25.715376)
2. (46.553276, 24.553548)
3. (46.324683, 23.743772)
4. (45.882789, 22.928532)
5. (46.179297, 21.332483)
6. (46.259770, 20.156942)

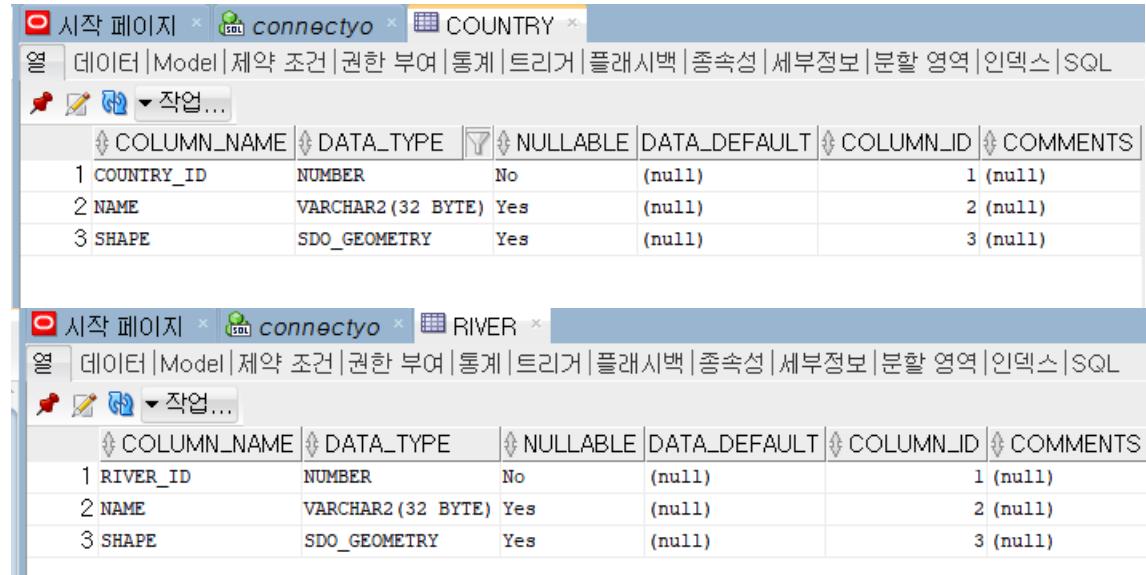
2. SQL

1) 데이터를 관리하는 table 을 생성하는 CREATE 문

```
CREATE TABLE COUNTRY
(
    country_id NUMBER PRIMARY KEY,
    name VARCHAR2(32),
    shape SDO_Geometry
);

CREATE TABLE RIVER
(
    river_id NUMBER PRIMARY KEY,
    name VARCHAR2(32),
    shape SDO_Geometry
);
```

각각 Country table 과 River table 을 생성하는 CREATE 문이다.



The screenshot shows two database structures in Oracle SQL Developer:

COUNTRY Table Structure:

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 COUNTRY_ID	NUMBER	No	(null)	1	(null)
2 NAME	VARCHAR2(32 BYTE)	Yes	(null)	2	(null)
3 SHAPE	SDO_Geometry	Yes	(null)	3	(null)

RIVER Table Structure:

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 RIVER_ID	NUMBER	No	(null)	1	(null)
2 NAME	VARCHAR2(32 BYTE)	Yes	(null)	2	(null)
3 SHAPE	SDO_Geometry	Yes	(null)	3	(null)

위와 같이 생성이 되는 것을 확인했다.

2) 해당 table 에 데이터를 삽입하는 INSERT 문

a. COUNTRY table 에 삽입

```
INSERT INTO COUNTRY VALUES
(
    1,
    'France',
    SDO_Geometry(
    2003,
    NULL,
    NULL,
    SDO_ELEM_INFO_ARRAY(1,1003,1),
    SDO_ORDINATE_ARRAY(51.054229,2.339865, 49.469402, 6.368522, 48.966455, 8.232948, 47.589749, 7.589084, 46.143156, 5.964401, 46.424675, 6.816491, 45.924097, 7.043506,
    43.784369, 7.529033, 43.278222, 5.356292, 43.542885, 3.887426, 43.113250, 3.094080, 42.449260, 3.153489, 43.369642, -1.789990, 46.159733, -1.220800, 48.509382, -4.761720,
    51.054229, 2.339865)
);
```

```
INSERT INTO COUNTRY VALUES
(
    2,
    'Switzerland',
    SDO_Geometry(
    2003,
    NULL,
    NULL,
    SDO_ELEM_INFO_ARRAY(1,1003,1),
    SDO_ORDINATE_ARRAY(47.589749, 7.589084, 47.496249, 9.565087, 46.939417, 10.460778, 46.234958, 10.125952, 46.513577, 9.347988, 45.835056, 9.015423, 46.464512, 8.433434, 45.924097, 7.043506,
    46.424675, 6.816491, 46.143156, 5.964401, 47.589749, 7.589084)
);
```

```

INSERT INTO COUNTRY VALUES
(
  3,
  'Germany',
  SDO_Geometry(
  2003,
  NULL,
  NULL,
  SDO_ELEM_INFO_ARRAY(1,1003,1),
  SDO_ORDINATE_ARRAY(54.881902, 8.660610, 54.729941, 9.943801, 53.914000, 11.339094, 54.565974, 13.594696, 53.706347, 14.249548, 50.858786, 14.807385, 50.273379, 12.054581,
  48.761658, 13.816679, 47.491295, 13.010570, 47.496249, 9.565087, 47.589749, 7.589084, 48.966455, 8.232948, 49.469402, 6.368522, 51.830706, 5.984595, 52.446698, 6.979074,
  53.405149, 7.016784, 53.856341, 8.597183, 54.881902, 8.660610)
);
;

INSERT INTO COUNTRY VALUES
(
  4,
  'Austria',
  SDO_Geometry(
  2003,
  NULL,
  NULL,
  SDO_ELEM_INFO_ARRAY(1,1003,1),
  SDO_ORDINATE_ARRAY(48.761658, 13.816679, 48.993386, 15.084363, 48.697393, 16.883462, 48.106544, 17.055717, 46.694588, 15.983913, 46.523662, 13.725469, 47.093491, 12.203891,
  46.939417, 10.460778, 47.496249, 9.565087, 47.491295, 13.010570, 48.761658, 13.816679)
);
;

INSERT INTO COUNTRY VALUES
(
  5,
  'Romania',
  SDO_Geometry(
  2003,
  NULL,
  NULL,
  SDO_ELEM_INFO_ARRAY(1,1003,1),
  SDO_ORDINATE_ARRAY(47.959995, 22.930931, 48.214695, 26.611841, 45.424993, 28.225357, 45.153307, 29.676849, 43.775358, 28.544951, 43.861832, 22.912095, 46.103132, 20.315388,
  47.959995, 22.930931)
);
;

INSERT INTO COUNTRY VALUES
(
  6,
  'Netherland',
  SDO_Geometry(
  2003,
  NULL,
  NULL,
  SDO_ELEM_INFO_ARRAY(1,1003,1),
  SDO_ORDINATE_ARRAY(53.213594, 5.589597, 53.405149, 7.016784, 52.446698, 6.979074, 51.830706, 5.984595, 51.113675, 5.817156, 51.364972, 3.387690, 52.932246, 4.773003,
  52.358388, 4.983996, 52.329867, 5.113208, 53.213594, 5.589597)
);
;

```

b. RIVER table에 삽입

```

INSERT INTO RIVER VALUES
(
  1,
  'Main River',
  SDO_Geometry(
  2002,
  NULL,
  NULL,
  SDO_ELEM_INFO_ARRAY(1,2,1),
  SDO_ORDINATE_ARRAY(49.993928, 8.288695, 50.099850, 8.558159, 50.1270079, 8.901116, 49.700620, 9.250788, 49.77251, 9.516735, 50.058439, 9.676245, 49.668891, 10.096535, 50.038423, 10.227997)
);
;

INSERT INTO RIVER VALUES
(
  2,
  'Amstel River',
  SDO_Geometry(
  2002,
  NULL,
  NULL,
  SDO_ELEM_INFO_ARRAY(1,1003,1),
  SDO_ORDINATE_ARRAY(52.329867, 5.113208, 52.358388, 4.983996, 52.247382, 5.047002, 52.296384, 4.900875, 52.329867, 5.113208)
);
;
```

```

INSERT INTO RIVER VALUES
(
  3,
  'Rhone River',
  SDO_GEOGRAPHY(
    2002,
    NULL,
    NULL,
    SDO_ELEM_INFO_ARRAY(1,1003,1),
    SDO_ORDINATE_ARRAY(46.237697, 6.249716, 46.207309, 5.613591, 45.627794, 5.664697, 45.786606, 4.864900, 43.624943, 4.678032, 43.552600, 5.099138, 45.343598, 5.016999, 45.343598, 6.280317,
    46.237697, 6.249716
  );
)

INSERT INTO RIVER VALUES
(
  4,
  'Inn River',
  SDO_GEOGRAPHY(
    2002,
    NULL,
    NULL,
    SDO_ELEM_INFO_ARRAY(1,2,1),
    SDO_ORDINATE_ARRAY(48.761658, 13.816679, 47.491295, 13.010570
  )
);
)

INSERT INTO RIVER VALUES
(
  5,
  'Mures River',
  SDO_GEOGRAPHY(
    2002,
    NULL,
    NULL,
    SDO_ELEM_INFO_ARRAY(1,2,1),
    SDO_ORDINATE_ARRAY(46.621413, 25.715376, 46.553276, 24.553548, 46.324683, 23.743772, 45.882789, 22.928532, 46.179297, 21.332483, 46.259770, 20.156942)
);
)

```

c. SELECT 문을 통해 topological relationship 을 모두 구현

1. TOUCH(MEET), EQUAL, DISJOINT

```

SELECT c.name,
       SDO_GEOGRAPHY.RELATE(c.shape, 'determine', c_b.shape, 0.005) relationship
  FROM country c, country c_b WHERE c_b.name = 'Germany';

```

스크립트 출력 x | 질의 결과 x

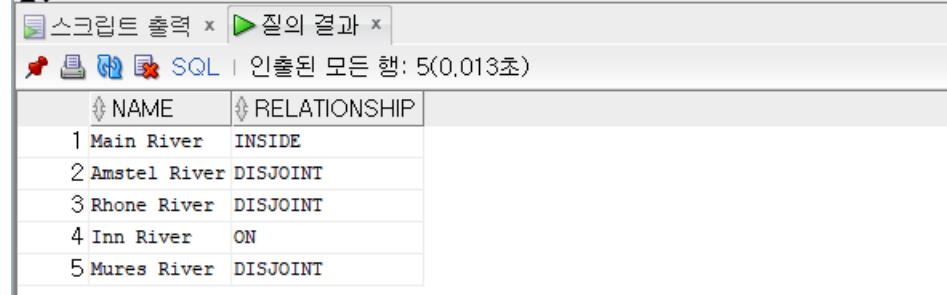
SQL | 인출된 모든 행: 6(0.021초)

NAME	RELATIONSHIP
1 France	TOUCH
2 Switzerland	TOUCH
3 Germany	EQUAL
4 Austria	TOUCH
5 Romania	DISJOINT
6 Netherland	TOUCH

GERMANY 와 다른 국가들간의 관계들을 확인해보았습니다. 인접하는 France, Switzerland, Austria, Netherland 와는 **TOUCH**, 자기 자신과는 **EQUAL** relationship 을 보입니다. Romania 와는 전혀 접점이 없기 때문에 **DISJOINT** 관계를 가집니다.

2. INSIDE, ON

```
SELECT r.name,
       SDO_GEOGRAPHICAL_RELATE(r.shape, 'determine', c_b.shape, 0.005) relationship
  FROM river r, country c_b WHERE c_b.name = 'Germany';
```

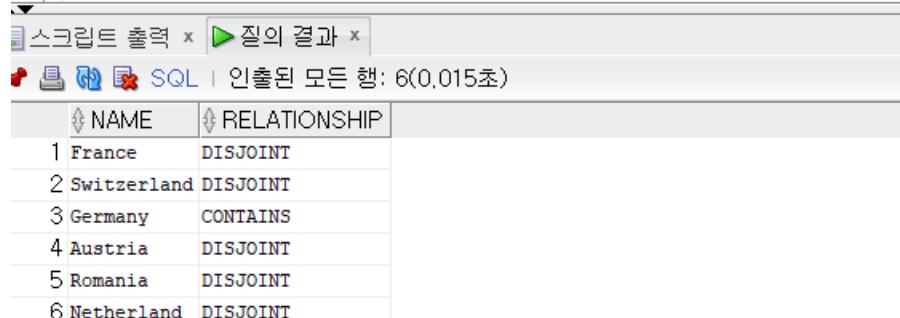


NAME	RELATIONSHIP
1 Main River	INSIDE
2 Amstel River	DISJOINT
3 Rhone River	DISJOINT
4 Inn River	ON
5 Mures River	DISJOINT

이번에는 GERMANY 와 강들 사이의 relationship 을 확인해봤습니다. Main River 의 경우 완전히 안에 있기 때문에 INSIDE relationship 을 갖습니다. 또, GERMANY 의 경계선 위에 존재하는 Inn River 의 경우 ON relationship 을 갖습니다. 이외 다른 River 들과는 Disjoint 관계입니다.

3. CONTAINS

```
SELECT c.name,
       SDO_GEOGRAPHICAL_RELATE(c.shape, 'determine', r_b.shape, 0.005) relationship
  FROM country c, river r_b WHERE r_b.name = 'Main River';
```



NAME	RELATIONSHIP
1 France	DISJOINT
2 Switzerland	DISJOINT
3 Germany	CONTAINS
4 Austria	DISJOINT
5 Romania	DISJOINT
6 Netherland	DISJOINT

Contains 관계를 확인하기 위해 이전에 inside 관계를 보였던 GERMANY – MAIN RIVER 간의 관계를 확인해보았습니다. 이번에는 Main River 의 입장에서 다른 국가와의 관계를 확인해보았고, 역시나 Main River 를 include 했던, Germany 가 Main River 와 Contains 관계를 보이고, 이외의 국가들과는 Disjoint 관계를 갖습니다.

4. COVERS, COVEREDBY

```
SELECT c.name,
       SDO_GEM.RELATE(c.shape, 'determine', r_b.shape, 0.005) relationship
  FROM country c, river r_b WHERE r_b.name = 'Amstel River';

SELECT r.name,
       SDO_GEM.RELATE(r.shape, 'determine', c_b.shape, 0.005) relationship
  FROM river r, country c_b WHERE c_b.name = 'Netherland';
```

스크립트 출력 x 질의 결과 x SQL | 인출된 모든 행: 6(0.018초)

NAME	RELATIONSHIP
1 France	DISJOINT
2 Switzerland	DISJOINT
3 Germany	DISJOINT
4 Austria	DISJOINT
5 Romania	DISJOINT
6 Netherland	COVERS

스크립트 출력 x 질의 결과 x SQL | 인출된 모든 행: 5(0.015초)

NAME	RELATIONSHIP
1 Main River	DISJOINT
2 Amstel River	COVEREDBY
3 Rhone River	DISJOINT
4 Inn River	DISJOINT
5 Mures River	DISJOINT

COVERS 과 COVEREDBY 관계를 확인하기 위해 NETHERLAND 과 Amstel River 의 관계를 확인해볼 수 있습니다. Netherland 의 경계에 한쪽 면이 맞닿아 있도록 좌표를 설정했기 때문에 COVERS, COVEREDBY 관계를 확인할 수 있습니다.

5. OVERLAPBYDISJOINT, OVERLAPBYINTERSECT

```
워크시트 질의 작성기
```

```
SELECT r.name,
       SDO_GEM.RELATE(r.shape, 'determine', c_b.shape, 0.005) relationship
  FROM river r, country c_b WHERE c_b.name = 'Romania';
```

스크립트 출력 x 질의 결과 x SQL | 인출된 모든 행: 5(0.013초)

NAME	RELATIONSHIP
1 Main River	DISJOINT
2 Amstel River	DISJOINT
3 Rhone River	DISJOINT
4 Inn River	DISJOINT
5 Mures River	OVERLAPBYDISJOINT

워크시트 질의 작성기

```
SELECT c.name,
       SDO_GEO_RELATE(c.shape, 'determine', r_b.shape, 0.005) relationship
  FROM country c, river r_b WHERE r_b.name = 'Rhone River';
```

스크립트 출력 x 질의 결과 x

SQL | 인출된 모든 행: 6(0.016초)

NAME	RELATIONSHIP
1 France	OVERLAPBYINTERSECT
2 Switzerland	OVERLAPBYINTERSECT
3 Germany	DISJOINT
4 Austria	DISJOINT
5 Romania	DISJOINT
6 Netherland	DISJOINT

Mures River 는 straight line segment 들로 이루어진 line string 이고, Romania 를 통과한다. 그렇기 때문에 OVERLAPBYDISJOINT 관계가 나타남을 확인할 수 있다. Rhone River 의 경우 straight line segment 로 이루어진 simple polygon 으로 볼 수 있고, France 와 Switzerland 에 걸쳐져있다. 그렇기 때문에 OVERLAPBYINTERSECT 관계가 나타남을 확인할 수 있다.

3. SCHEMA

마지막으로 생성한 table 에 해당하는 schema 를 작성해보았습니다.

COUNTRY

country_id	NUMBER
name	VARCHAR2(32)
shape	SDO_Geometry

RIVER

river_id	NUMBER
name	VARCHAR2(32)
shape	SDO_Geometry

시작 페이지 x connectyo COUNTRY x

열 | 데이터 | Model | 제약 조건 | 권한 부여 | 통계 | 트리거 | 플래시백 | 종속성 | 세부정보 | 분할 영역 | 인덱스 | SQL

작업... | COLUMN_NAME | DATA_TYPE | NULLABLE | DATA_DEFAULT | COLUMN_ID | COMMENTS |

1 COUNTRY_ID	NUMBER	No	(null)	1	(null)
2 NAME	VARCHAR2(32 BYTE)	Yes	(null)	2	(null)
3 SHAPE	SDO_Geometry	Yes	(null)	3	(null)

시작 페이지 x connectyo RIVER x

열 | 데이터 | Model | 제약 조건 | 권한 부여 | 통계 | 트리거 | 플래시백 | 종속성 | 세부정보 | 분할 영역 | 인덱스 | SQL

작업... | COLUMN_NAME | DATA_TYPE | NULLABLE | DATA_DEFAULT | COLUMN_ID | COMMENTS |

1 RIVER_ID	NUMBER	No	(null)	1	(null)
2 NAME	VARCHAR2(32 BYTE)	Yes	(null)	2	(null)
3 SHAPE	SDO_Geometry	Yes	(null)	3	(null)