



2014/12/31

PostGIS 公式サイトにある osm2pgrouting で経路探索

[OpenStreetMap] [pgRouting]

昨日 Windows 7 32bit 上の PostgreSQL 9.4 に PostGIS と pgRouting を入れる途中で、公式サイト
のファイル配布ページ ↓ を見ました。よく見ると extras というフォルダがあり(上から二行
目) osm2pgrouting という OpenStreetMap データを pgRouting にインポートするツールがありました。
今日試したら問題なく動作。これで Windows で OpenStreetMap + pgRouting の経路探索データ
ベースが簡単にできます。以下、手順のメモ。

■ <http://winnie.postgis.net/download/windows/pg94/buildbot/>

12/19/2014 11:02 PM	<dir>	archive
11/27/2014 2:38 PM	<dir>	extras
12/20/2014 11:49 PM	429129	pgrouting-pg94-binaries-2.0.0w32gcc481.zip
12/20/2014 11:54 PM	523758	pgrouting-pg94-binaries-2.0.0w64gcc48.zip
12/20/2014 9:18 PM	497058	pgrouting-pg94-binaries-2.0w32gcc481.zip
12/20/2014 9:13 PM	588551	pgrouting-pg94-binaries-2.0w64gcc48.zip
12/21/2014 12:04 AM	25158905	postgis-bundle-pg9432-2.1.5-2.zip
12/21/2014 12:02 AM	21370528	postgis-bundle-pg9432-setup-2.1.5-2.exe
12/20/2014 11:28 PM	26000197	postgis-bundle-pg9464-2.1.5-2.zip
12/20/2014 11:22 PM	21412122	postgis-bundle-pg9464-setup-2.1.5-2.exe
12/21/2014 3:08 AM	24993121	postgis-pg94-binaries-2.1.5w32gcc481.zip
12/20/2014 2:01 AM	25490482	postgis-pg94-binaries-2.1.5w64gcc48-1.zip
12/22/2014 7:40 AM	24996248	postgis-pg94-binaries-2.1.6devw32gcc481.zip
12/18/2014 6:28 PM	26191493	postgis-pg94-binaries-2.2.0devw32gcc481.zip
12/18/2014 5:06 PM	27081349	postgis-pg94-binaries-2.2.0devw64gcc48.zip

↓ 公式サイトでは Experimental Builds の一つになっています。

■ http://postgis.net/windows_downloads

Windows: Winnie Bot PostGIS and pgRouting Experimental Builds

- for PostgreSQL 9.4 32-bit 64-bit compiled against PostgreSQL 9.4beta3 for PostGIS 2.2 (built with SFCGAL support) (also pgRouting 2.0)

(in [extras folder](#): [osm2pgrouting](#) for pgRouting 2.0
- the osm2pgrouting should work fine with PostgreSQL 9.3 as well)

↓ osm2pgrouting は ZIP で 32 / 64 bit 版それぞれがあり、今回は前者を使用。

■ <http://winnie.postgis.net/download/windows/pg94/buildbot/extras/>

powered by Seesaa

Seesaaブログ

Google でブログ内検索

Search

リンク文字色変更

2015.05~

kenpg.bitbucket.org

で更新中です

タグと個別記事

- PostgreSQL (84)
- PostGIS (38)
- Python (10)
- R (45)
- Qt (19)

- MADlib (5)
- pgAdmin (13)
- pgRouting (5)

- OpenStreetMap (13)
- SoilGrids (3)

- Chart (9)
- MathJax (9)
- SVG (17)
- XML (7)

- CentOS (16)
- VMware (4)

- 統計分析 (8)
- 実行環境 (17)

- Browser (7)
- Firefox (6)
- PC (11)

- Info (23)
- About and contact

2014/09/16 まで

- Top
- PostgreSQL
- PostGIS
- R
- データいろいろ

日本語ドキュメント
(外部リンク)

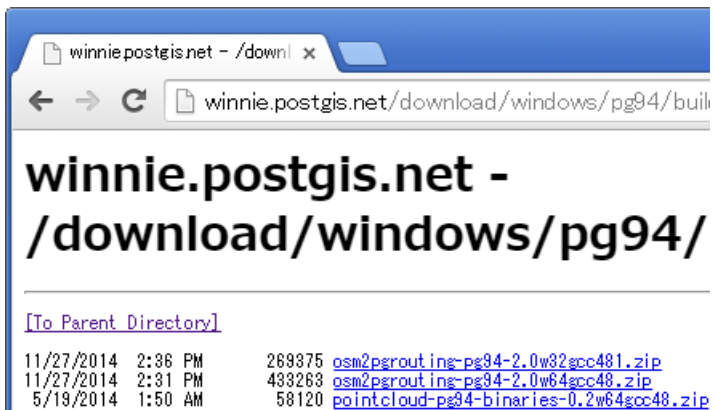
- PostgreSQL
- PostGIS

公式ウェブサイト

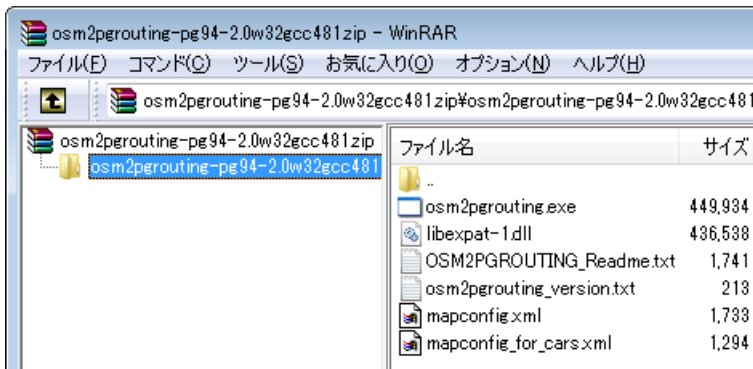
- PostgreSQL
- // 日本ユーザ会
- PostGIS
- Python
- R
- Qt Project
- // 日本ユーザー会
- CentOS

- MADlib
- pgAdmin
- pgRouting
- PL/R

- ・ OpenStreetMap
- ・ // Japan
- ・ OSGeo
- ・ // Japan
- ・ FOSS4G 2015



↓ ZIP の中身は osm2pgrouting 本体、ライブラリ、説明、変換時の設定 XML です。コマンドラインツールなのでインストーラはありません。



解凍したら、説明ファイル (Readme) に Copy the osm2pgrouting.exe and libexpat-1.dll to your PostgreSQL bin folder, other dependencies should already be there とあったので、その通り PostgreSQL のプログラムがある bin フォルダに実行ファイルとライブラリをコピーしました。

次に、適当な OpenStreetMap データを用意します。説明ファイルは親切に [Weekly OSM Metro Extracts](#) というウェブサイトでは世界の主要都市のデータが得られると紹介しており、確かにそうでした。↓ OpenStreetMap のネイティブデータの他、Shapefile や GeoJSON などもあります。osm2pgrouting に使うのは OSM XML です。

Mapzen - Start where you x

https://mapzen.com/metro-extracts/

ALL A B C D E F G H I J K L M
O P Q R S T U V W X Y Z

Sacramento, California

OSM PBF (7.3 MB)	OSM XML (13.9 MB)	OSM2PGSQL SHP (15.7 MB)
OSM2PGSQL GEOJSON (11.2 MB)	IMPOSM SHP (14.4 MB)	IMPOSM GEOJSON (17.4 MB)

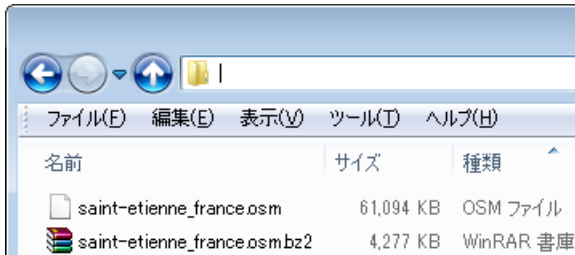
Saguenay, Canada

OSM PBF (1.7 MB)	OSM XML (2.5 MB)	OSM2PGSQL SHP (2.0 MB)
OSM2PGSQL GEOJSON (1.3 MB)	IMPOSM SHP (2.1 MB)	IMPOSM GEOJSON (2.5 MB)

Saint Etienne, France

OSM PBF (2.8 MB)	OSM XML (4.2 MB)	OSM2PGSQL SHP (4.7 MB)
OSM2PGSQL GEOJSON (3.0 MB)	IMPOSM SHP (5.1 MB)	IMPOSM GEOJSON (5.7 MB)

日本の都市もいくつかありますが、サイズが割と大きいので今回は↑フランスの Saint Etienne にしました。ポップグループの *Saint Etienne* の名はこのサーカーチームに由来するとか。bzip2 で圧縮されており、解凍すると ↓ 61 MB ありました。



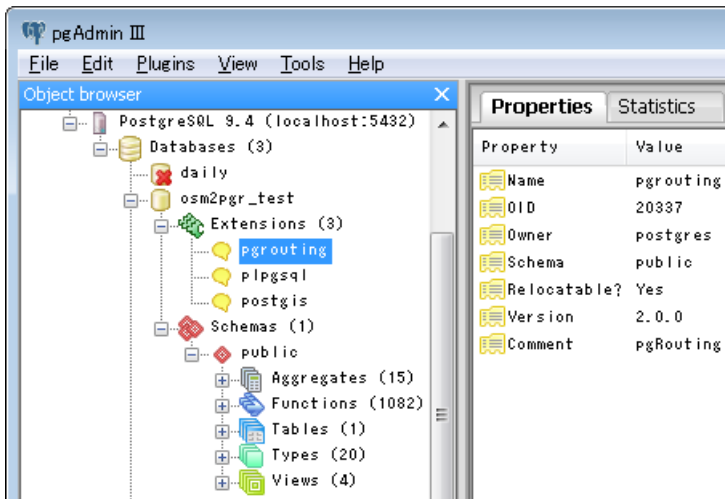
↓ テスト用のインポート先データベースを作り、**昨日**の最後と同様に PostGIS と pgRouting をインストールして、データベース側は準備完了。

```

-- // on DB postgres
CREATE DATABASE osm2pgr_test ;

-- // on DB osm2pgr_test
CREATE EXTENSION postgis ;
CREATE EXTENSION pgrouting ;

```



コマンドを直接打ってもいいですが、今回は↓のようなバッチファイルを作りました。OSM ファイルおよび変換設定用の mapconfig.xml が同じ場所にある前提。なお自動車での最短経路探索に使う場合は mapconfig_for_cars.xml を使います(車の通れない道路は探索対象にならない、などいろいろ設定される)。

```

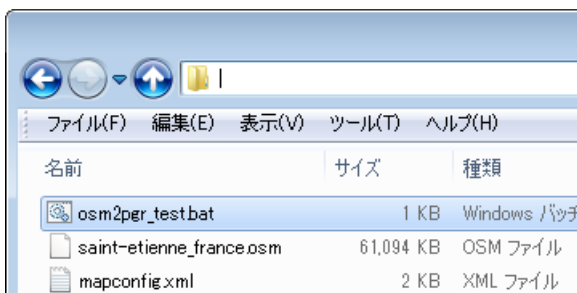
@ECHO OFF
"C:/PostgreSQL/9.4/bin/osm2pgrouting.exe" -file saint-etienne_france.osm
    -conf mapconfig.xml -dbname osm2pgr_test -user postgres -passwd xxxx
    -prefixtables saint_etienne_ (実際は一行)
PAUSE

```

コマンドライン引数の意味は次のとおり。説明ファイルでは passwd が optional なのに対し、自分の環境では必須でした。他にも若干の引数があり、詳しくは説明ファイルを参照。

- file <OSM ファイル>
- conf <変換設定用の XML ファイル>
- dbname <インポート先データベース>
- user <ユーザ名>
- passwd <パスワード>
- prefixtables <インポート先テーブルの接頭辞>

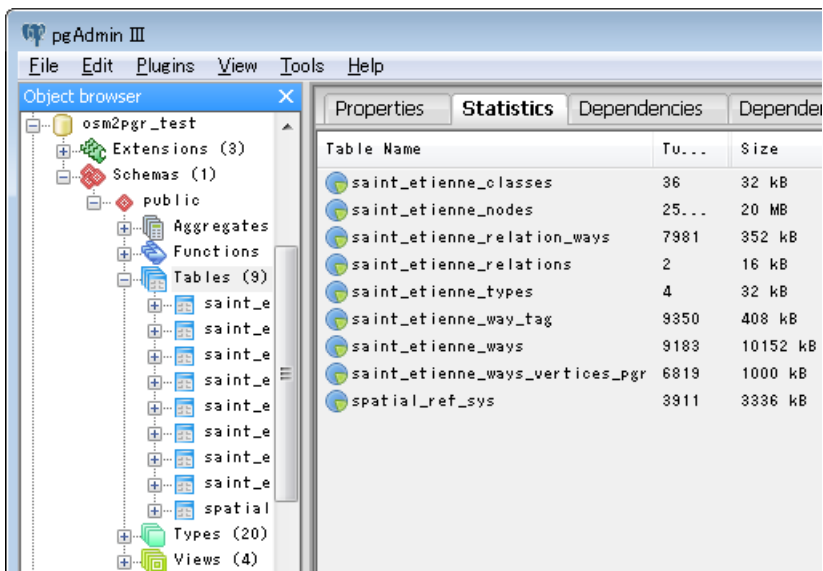
↓ バッチファイルと OSM ファイルおよび変換設定用の mapconfig.xml を同じ場所に置き、バッチを実行します。



```
C:\Windows\system32\cmd.exe
host=127.0.0.1 user=postgres dbname=osm2pgr_test port=5432 password=
connection success
Trying to load config file mapconfig.xml
Trying to parse config
Trying to load data
Trying to parse data
Split ways
Creating tables...
Nodes table created
2create ways failed:
Types table created
Way_tag table created
Relations table created
Relation_ways table created
Classes table created
Adding tag types and classes to database...
Adding relations to database...
Adding nodes to database...
Adding ways to database...
Creating topology...
NOTICE: PROCESSING:
NOTICE: pgr_createTopology('saint_etienne_ways',1e-005,'the_geom','gid','source',
'target',true)
NOTICE: Performing checks, please wait .....
NOTICE: Creating Topology, Please wait...
NOTICE: 1000 edges processed
NOTICE: 2000 edges processed
NOTICE: 3000 edges processed
NOTICE: 4000 edges processed
NOTICE: 5000 edges processed
NOTICE: 6000 edges processed
NOTICE: 7000 edges processed
NOTICE: 8000 edges processed
NOTICE: 9000 edges processed
NOTICE: -----> TOPOLOGY CREATED FOR 9183 edges
NOTICE: Rows with NULL geometry or NULL id: 0
NOTICE: Vertices table for table public.saint_etienne_ways is: public.saint_eti
enne_ways_vertices_pgr
NOTICE: -----
Create Topology success
#####
size of streets: 4522
size of splitted ways : 9183
finished
続行するには何かキーを押してください . . .
```

数秒で無事完了。時間をちゃんと測るのは忘れましたが、最初の方 2create ways failed が気になりますが、全体としては pgRouting で経路探索するためのトポロジーができたようで、とりあえずこのまま使います。

↓新しい8個のテーブルが、指定した接頭辞つきで作成されています。



↓経路探索で基本になるエッジ(リンク)テーブル。インデクスも自動作成されました。

The screenshot shows the pgAdmin III interface. On the left, the 'Object browser' displays a tree view of the database schema. The 'saint_etienne_ways' table is selected, showing its columns: gid, class_id, length, name, x1, y1, x2, y2, reverse_cost, rule, to_cost, maxspeed_forward, maxspeed_backward, osm_id, priority, the_geom, source, and target. Below the columns, there are sections for Constraints (0), Indexes (4), Rules (0), Triggers (0), and a view named 'saint_etienne_ways_vertices_pgr' with 6 columns.

The SQL pane on the right contains the following SQL script:

```
-- Table: saint_etienne_ways
-- DROP TABLE saint_etienne_ways;

CREATE TABLE saint_etienne_ways
(
    gid integer,
    class_id integer NOT NULL,
    length double precision,
    name text,
    x1 double precision,
    y1 double precision,
    x2 double precision,
    y2 double precision,
    reverse_cost double precision,
    rule text,
    to_cost double precision,
    maxspeed_forward integer,
    maxspeed_backward integer,
    osm_id bigint,
    priority double precision DEFAULT
    the_geom geometry(LineString,4326)
    source integer,
    target integer
)
WITH (
    OIDS=FALSE
);
ALTER TABLE saint_etienne_ways
    OWNER TO postgres;

-- Index: saint_etienne_geom_idx
-- DROP INDEX saint_etienne_geom_idx

CREATE INDEX saint_etienne_geom_idx
    ON saint_etienne_ways
    USING gist
    (the_geom);

-- Index: saint_etienne_source_idx
-- DROP INDEX saint_etienne_source_i
```

エッジテーブルの to_cost はすべて NULL だったため、今回は length をコストにします。length の単位が不明でしたが、各エッジのおおよそのメートル長を算出して比べると ↓ ほぼ 1 対 1000 なので km と推測できます。

```
.....
SELECT length, ST_Length(the_geom :: geography)
FROM saint_etienne_ways LIMIT 20 ;
```

Query - osm2pgr_test on postgres@localhost:5432 *

SQL Editor Graphical Query Builder

Previous queries

```
SELECT length, ST_Length(the_geom :: geography)
FROM saint_etienne_ways LIMIT 20 ;
```

Output pane

Data Output Explain Messages History

	length double precision	st_length double precision
1	0.0178131620032084	17.8193095553648
2	0.028161841252673	28.2275225707737
3	0.0180949954231537	18.0982171565458
4	0.0704158945982812	70.4619027412973
5	0.0608548200397201	60.96458442519
6	0.0176922505700015	17.7354535811147
7	0.0322879023265133	32.3469934219947
8	0.857865021627722	882.130460593038
9	0.109369892949941	110.011246364895
10	0.506164954769751	506.675341927143
11	0.625519157616946	664.290783328744
12	0.197481672478701	201.331684880984
13	0.177497292785142	178.445274630108
14	0.269169479578957	269.663830545181
15	0.450145965147217	456.343740342997
16	0.0747944357123131	74.8816039492968

↓ 早速 A star アルゴリズムによる pgr_astar 関数で最短経路を探索した例。起点ノード=1、終点ノード=7です。関数の詳細は[ドキュメント](#)を参照。最右列はウィンドウ関数で距離を累計しており、最終行が全体の道路距離になります。

```
.....
SELECT seq, id1 AS node, id2 AS edge, cost, sum(cost) over w
FROM pgr_astar($$
  SELECT gid AS id, source, target, length AS cost, x1, y1, x2, y2
  FROM saint_etienne_ways
$$, 1, 7, false, false)
WINDOW w AS (ORDER BY seq) ;
```

```

SELECT seq, id1 AS node, id2 AS edge, cost, sum(cost) over w
FROM pgr_astar($$
  SELECT gid AS id, source, target, length AS cost, x1, y1, x2, y2
  FROM saint_etienne_ways
  $$, 1, 7, false, false)
WINDOW w AS (ORDER BY seq) ;

```

	seq integer	node integer	edge integer	cost double precision	sum double precision
1	0	1	7506	0.249694173877369	0.249694173877369
2	1	3874	5895	0.345765552247132	0.595459726124501
3	2	3873	7500	0.442214785627401	1.0376745117519
4	3	4772	10466	0.0897833651902721	1.12745787694217
5	4	6071	12278	0.568513515289882	1.69597139223206
6	5	1933	2620	0.222025652195412	1.91799704442747
7	6	1934	2621	0.0343480559212595	1.95234510034873
8	7	1935	2631	0.0298174469172745	1.982162547266
9	8	1943	12280	0.0298997771641256	2.01206232443013
10	9	4486	6819	0.224405000695605	2.23646732512573
11	10	4485	8321	0.0160610352380467	2.25252836036378
12	11	7	-1	0	2.25252836036378

上で求めた道路距離は 2km ちよつと。試しに直線距離を概算で求めると ↓ 約 1.5km で、やや大回りなルートですが(都市部ではだいたい直線距離×1.3 が平均的)ネットワーク構造によってはあり得ると思います。

```

SELECT ST_Length(ST_MakeLine(the_geom) :: geography)
FROM saint_etienne_ways_vertices_pgr WHERE id IN (1, 7) ;

```

```

SELECT ST_Length(ST_MakeLine(the_geom) :: geography)
FROM saint_etienne_ways_vertices_pgr WHERE id IN (1, 7) ;

```

	st_length double precision
1	1492.93828482434

↓最後に少し汎用的な例。最初の a ブロックで起終点ノードを設定するだけで、A star による最短経路 + 道路距離 + 概算の直線距離を同時に表示します。しよつちゅう道路距離が直線距離の数倍という結果だとネットワーク作成が適切でない可能性があり、その簡易チェック用。

```

WITH a (p1, p2) AS (
  -- // 任意の起終点ノードを指定
  VALUES (1, 9)
), b AS (
  SELECT pgr_astar($$
    SELECT gid AS id, source, target, length AS cost, x1, y1, x2, y2
    FROM saint_etienne_ways
    $$, p1, p2, false, false)
FROM a
), c AS (

```



```

SELECT (pgr_astar).* FROM b
)
SELECT seq, id1 AS node, id2 AS edge, cost, sum(cost) over w
FROM c
WINDOW w AS (ORDER BY seq)
UNION ALL
SELECT NULL, NULL, NULL, NULL
, ST_Length(ST_MakeLine(the_geom):: geography) / 10^3
FROM a, saint_etienne_ways_vertices_pgr WHERE id IN (p1, p2);

```

Query - osm2pgr_test on postgres@localhost:5432 *

SQL Editor

```

WITH a (p1, p2) AS (
VALUES (1, 9)
), b AS (
SELECT pgr_astar($$
SELECT gid AS id, source, target, length AS cost, x1, y1, x2, y2
FROM saint_etienne_ways
$$, p1, p2, false, false)
FROM a
), c AS (
SELECT (pgr_astar).* FROM b
)
SELECT seq, id1 AS node, id2 AS edge, cost, sum(cost) over w
FROM c
WINDOW w AS (ORDER BY seq)
UNION ALL
SELECT NULL, NULL, NULL, NULL
, ST_Length(ST_MakeLine(the_geom):: geography) / 10^3
FROM a, saint_etienne_ways_vertices_pgr WHERE id IN (p1, p2);

```

Output pane

seq	node	edge	cost	sum
integer	integer	integer	double precision	double precision
1	0	1	10430	0.0697899028856267
2	1	3304	10431	0.12701158583374
3	2	4762	10432	0.196752287332029
4	3	3035	10433	0.222532475773425
5	4	6063	10434	0.255580575535053
6	5	6064	10451	0.321943538273227
7	6	3273	10452	0.619298712518521
8	7	6069	10453	0.641348119315663
9	8	1927	12302	0.655203164284214
10	9	6491	11673	0.682529394122868
11	10	3436	4962	0.732504915401643
12	11	3434	9030	0.758252520520487
62	61	5808	11818	3.55736136110911
63	62	2223	6919	3.57611680552518
64	63	4536	6920	3.61601615973901
65	64	4537	9781	3.66879835903582
66	65	2219	3105	3.68007907537217
67	66	2220	9766	3.71714836934363
68	67	5802	9767	3.76395061693419
69	68	9	-1	0
70				3.34210178912306

↑ 起終点ノード(1,9)で経路探索したところ、直線距離 3.3km に対し道路距離 3.8km と、かなり直線に近いです(×1.2 弱)。何度か試した結果、道路距離が直線距離の数倍になるケースはなく OpenStreetMap からの道路ネットワーク作成は大丈夫そう。来年は東京など知っている都市のデータで試し、地図プロットも行って結果を確認する予定です。

- » 次の記事 : [osm2pgrouting が出す道路長は、実際より短いらしい件](#)
 << 前の記事 : [PostgreSQL 9.4 に PostGIS & pgRouting インストール](#)