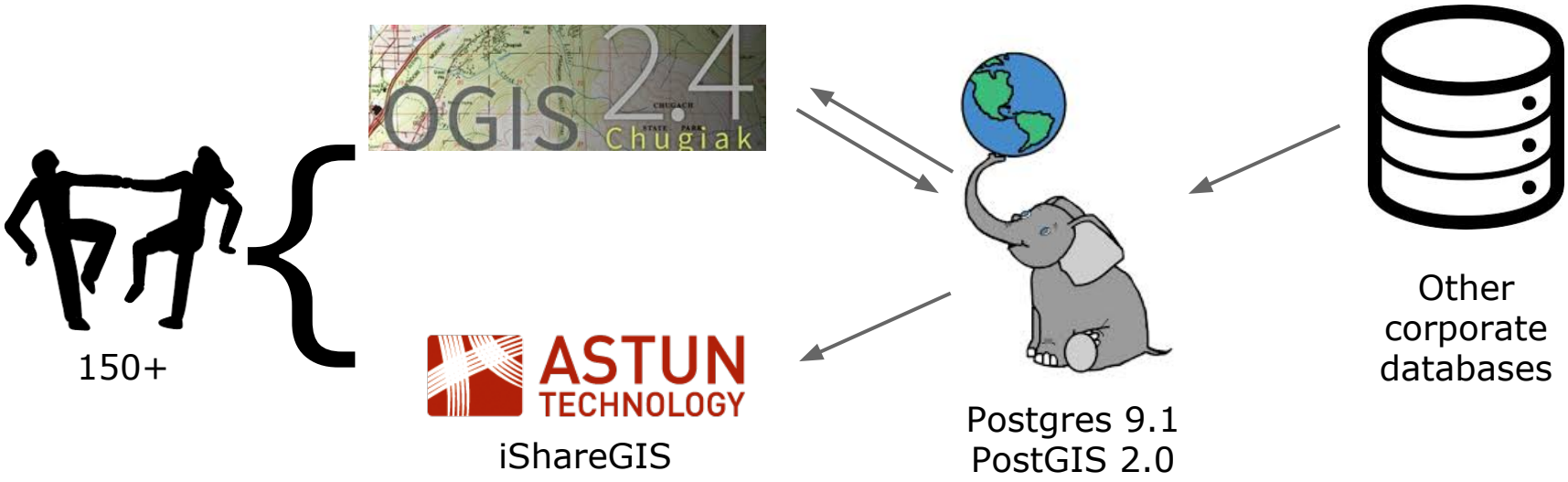
PostGIS @ RBWM

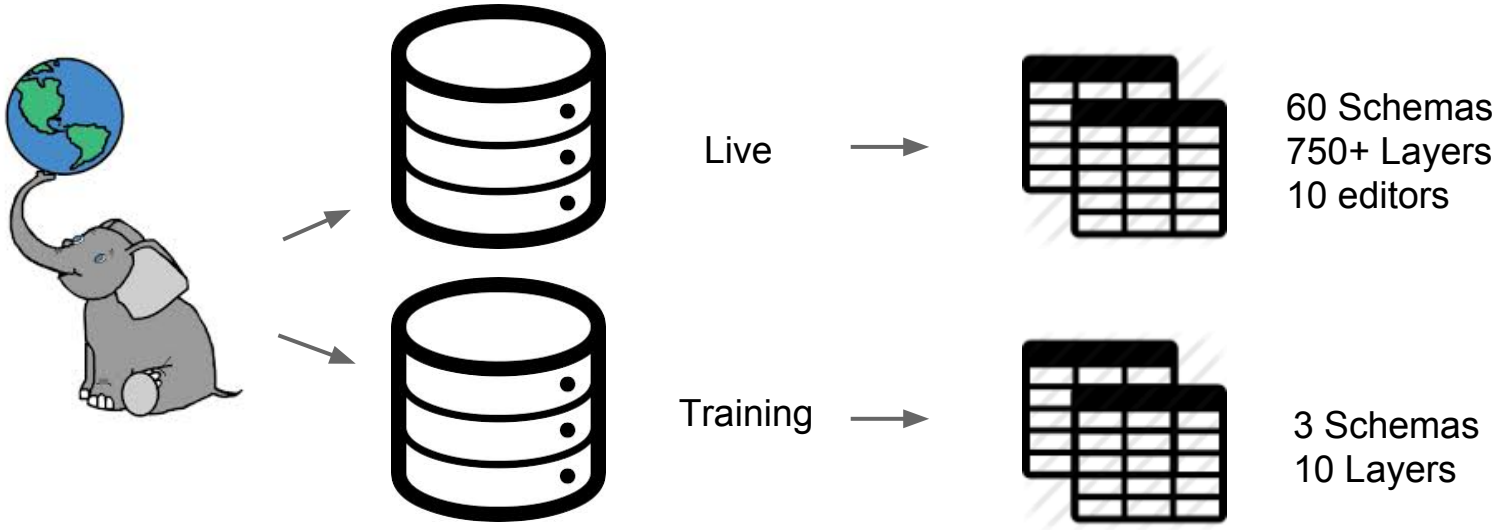
in ten slides!

Simon Miles
@geosmiles
Simon.Miles@rbwm.gov.uk

Our Architecture



Behind the scenes?



Schemas & Tables!

Schema: Simple Names

Addresses
Boundaries
Council Facilities
Planning Applications
Trees
Highways
Highways Private
Community Safety
Flooding

Tables: Simple Names

Postcode Sectors
Ward
Council Buildings
Last 14 days
TPO areas
Highways Record
UKPMS draft
Crime data for January 2014
Flood Zone 3

Previously in ArcSDE: **CORP11_DO:TPO areas** WTF!

Conventions

NAMES

- Always in lower case
- Never starts with a number
- Spaces are replaced with underscores

DEFAULT FIELDS

- lab_x
- lab_y
- lab_rot
- owner
- department
- notes
- status_of_data
- inspire_la
- inspire_region
- inspire_country

Points, Lines & Polygons

Points
- x
- y

Lines
- length

Polygons
- x_centre
- y_centre
- area (acres, hectares)

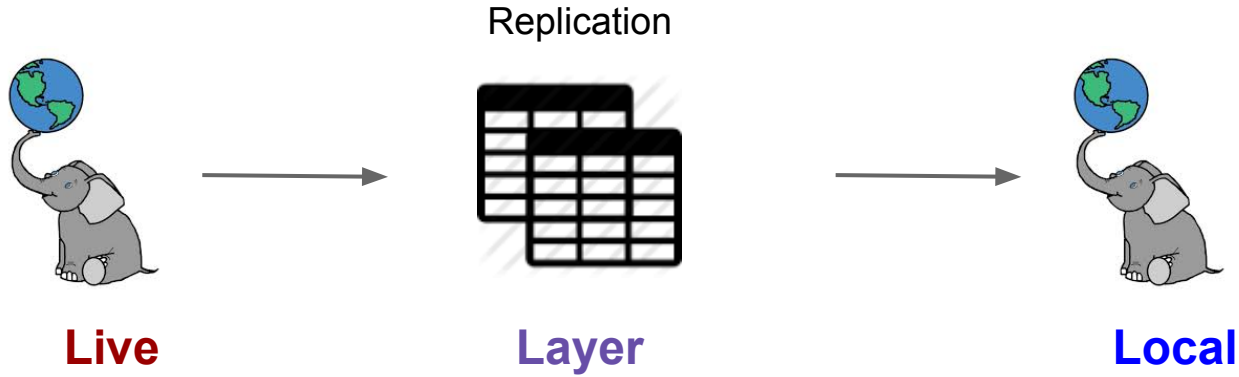
Each feature type has its own set of triggers/ trigger functions:

```
CREATE OR REPLACE FUNCTION public.cal_centroids()
  RETURNS trigger AS
$BODY$
BEGIN
    NEW.x_centre := ST_X(ST_PointOnSurface(NEW.geom));
    NEW.y_centre := ST_Y(ST_PointOnSurface(NEW.geom));
    RETURN NEW;
END;
$BODY$
  LANGUAGE plpgsql VOLATILE
  COST 100;
ALTER FUNCTION public.cal_centroids()
  OWNER TO postgres;
```

Other triggers:

```
CREATE OR REPLACE FUNCTION update_details_asb()
  RETURNS trigger AS
$BODY$
BEGIN
NEW.x := st_x(NEW.geom);
NEW.y := st_y(NEW.geom);

NEW.parish = nm FROM boundaries.parish AS B WHERE ST_Within(NEW.geom, B.geom);

NEW.ward = ward_names FROM boundaries.ward AS B WHERE ST_Within(NEW.geom, B.geom);

NEW.neighbourhood_area = nhname FROM community_safety.neighbourhood_areas AS B WHERE
ST_Within(NEW.geom, B.geom);

NEW.tvp = tvp_name FROM community_safety.thames_valley_police_areas AS B WHERE ST_Within
(NEW.geom, B.geom);
RETURN NEW;
END;
$BODY$
  LANGUAGE plpgsql VOLATILE
  COST 100;
ALTER FUNCTION update_details_asb()
  OWNER TO postgres;
```

Cool stuff?




```
cd /d C:\
cd "C:\OSGeo4W\bin"
ogr2ogr -f PostgreSQL --config PG_USE_COPY YES
PG:"dbname='rbwm_gis_local' host='localhost' port='5432' user='*!@?' password='?@!*'"
PG:"dbname='rbwm_gis' host='server' port='5432' user='@@££!!' password='!!@@FF'"
-overwrite  parking_enforcement.restrictions
exit
```

Cool stuff?

```
import os

############################################################
#                                                          #
#               FOR POLYGON DATA                           #
# For this script to work you require                      #
#python 2.7x                                               #
#                                                          #
#                                                          #
############################################################

path = "D:/batch_files/create_sql" # Where the script is to be created

schema_table = "historic_environment.listed_buildings" # ENTER THE NAME OF THE SCHEMA AND TABLE TO BE ALTERED
geom_field = "geom" # WHAT IS THE geomtery field called
owner = "'Rachel Fletcher'" # Who owns the data
department = "'Planning - Conservation'" # What is the department called
notes = "'This data was captured at 1 to 1250 scale'" # Some notes about the data
status_of_data = "'Register of Listed Buildings within RBWM'" # status of the data
la = "'Royal Borough of Windsor and Maidenhead'" # local authoirty name
region = "'South East England'" # region in the UK
country = "'United Kingdom'" # the county in which we live

os.chdir((path))
file = open((schema_table) + ".sql", 'w')
file.write('ALTER TABLE ' + (schema_table) + ' ADD COLUMN lab_x numeric(10,0);\n')
file.write('ALTER TABLE ' + (schema_table) + ' ADD COLUMN lab_y numeric(10,0);\n')
file.write('ALTER TABLE ' + (schema_table) + ' ADD COLUMN lab_rot numeric(10,0);\n')

file.write('ALTER TABLE ' + (schema_table) + ' ADD x_centre numeric(6,0);\n')
file.write('ALTER TABLE ' + (schema_table) + ' ADD y_centre numeric(6,0);\n')
file.write('UPDATE ' + (schema_table) + ' SET x_centre = ST_X(ST_PointOnSurface(' + (geom_field) +'));\n')
file.write('UPDATE ' + (schema_table) + ' SET y_centre = ST_Y(ST_PointOnSurface(' + (geom_field) +'));\n')

file.write('ALTER TABLE ' + (schema_table) + ' ADD COLUMN owner text;\n')
file.write('ALTER TABLE ' + (schema_table) + ' ALTER COLUMN owner SET DEFAULT '+ (owner) + '::text;\n')
file.write('UPDATE ' + (schema_table) + ' SET owner = '+ (owner) + ';\n')
```

historic_environment.listed_buildings.sql

```
ALTER TABLE historic_environment.listed_buildings ADD COLUMN lab_x nu
ALTER TABLE historic_environment.listed_buildings ADD COLUMN lab_y nu
ALTER TABLE historic_environment.listed_buildings ADD COLUMN lab_rot

ALTER TABLE historic_environment.listed_buildings ADD x_centre numeric
ALTER TABLE historic_environment.listed_buildings ADD y_centre numeric

UPDATE historic_environment.listed_buildings SET x_centre = ST_X(ST_P
UPDATE historic_environment.listed_buildings SET y_centre = ST_Y(ST_P

ALTER TABLE historic_environment.listed_buildings ADD COLUMN owner t
ALTER TABLE historic_environment.listed_buildings ALTER COLUMN owner
UPDATE historic_environment.listed_buildings SET owner = 'Rachel Fletch
```

Using python to quickly create SQL

Fin

Simon Miles
@geosmiles
Simon.Miles@rbwm.gov.uk