# Mapping election results with CartoWeb

# Mapping election results with CartoWeb

- Introduction
    - Project presentation
    - Constraints
    - Slide show
    - Processes
- Pseudo-dynamic Mapfile generation:
    - Automatic generation of multiple layers
    - Automatic adaptation of class intervals
- Load / performance tuning for high traffic:
    - Caching images with CartoWeb
    - Database tuning
- Access statistics
- Possible improvements
- Questions

# Introduction

## Project presentation & software's choice

- The goal of this web mapping project is:
  - to show election results live using thematic maps. This includes evolution maps between 2001 & 2006.

- Constraints:
  - No Plugin for the client -> raster image
  - Open source software to adapt it to our needs
  - Human intervention on the election day should be limited to acquiring the data
  - MicroGIS Know-how

- Software's Choice
  - Mapserver/CartoWeb in relation with PostgreSQL/PostGIS

# Introduction
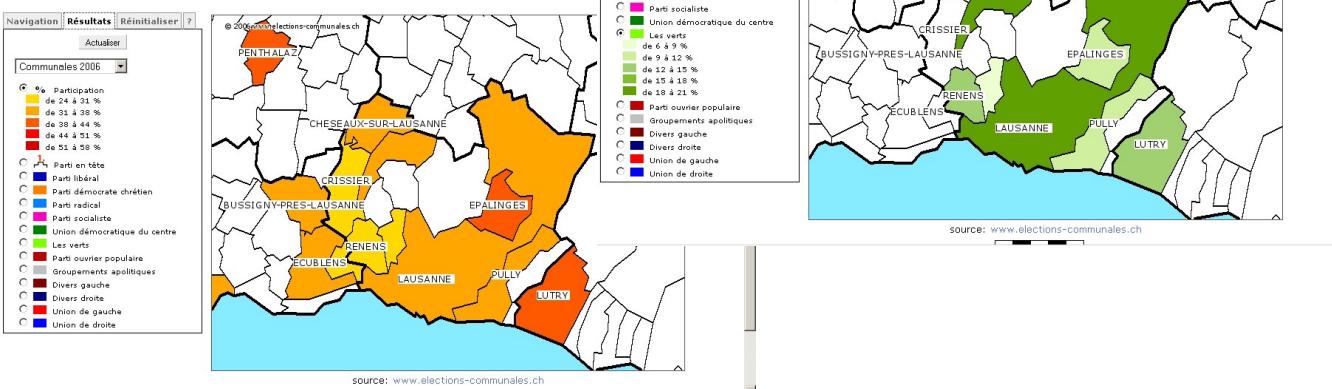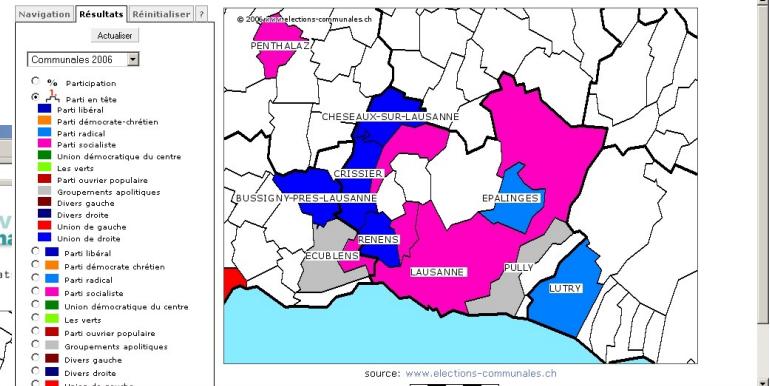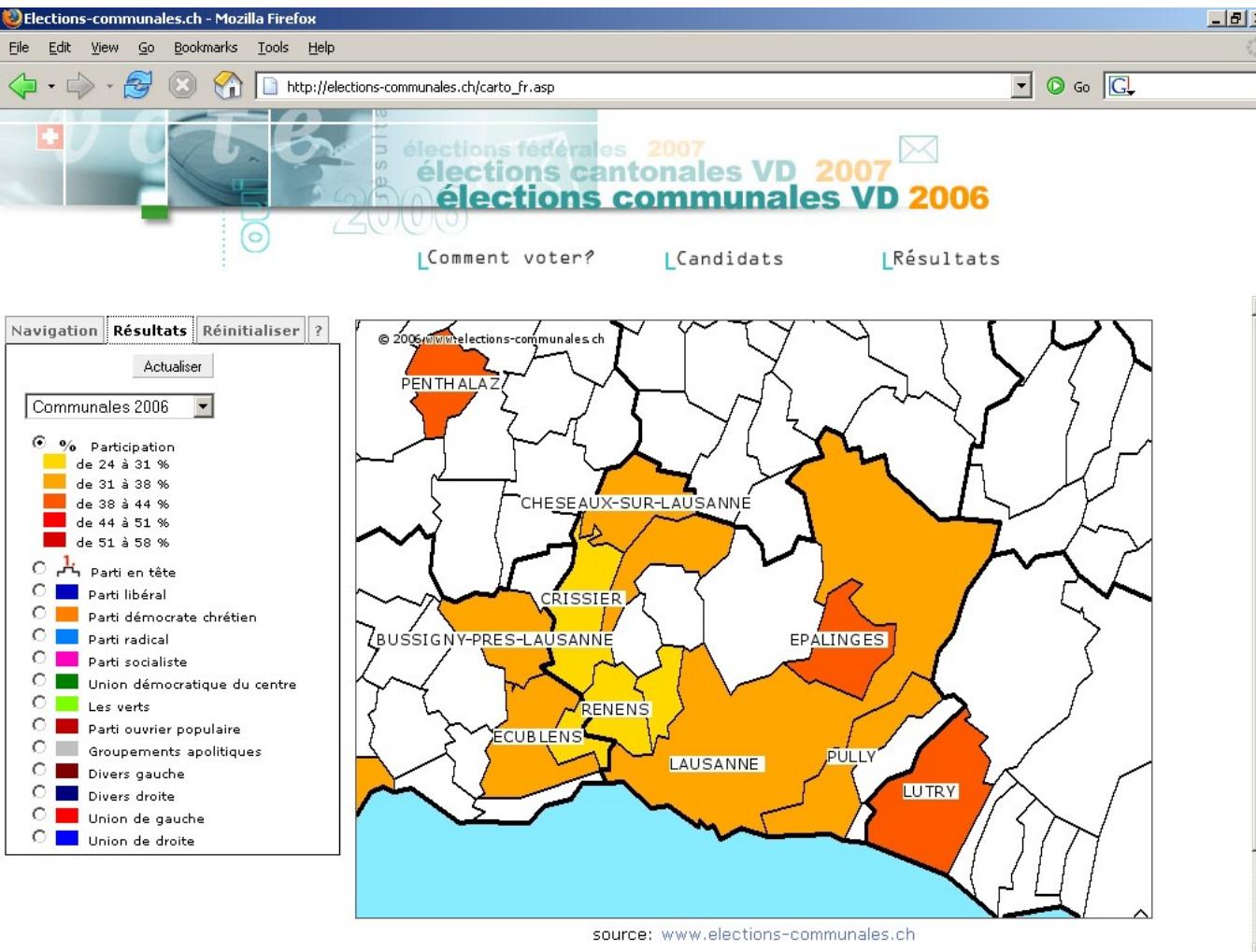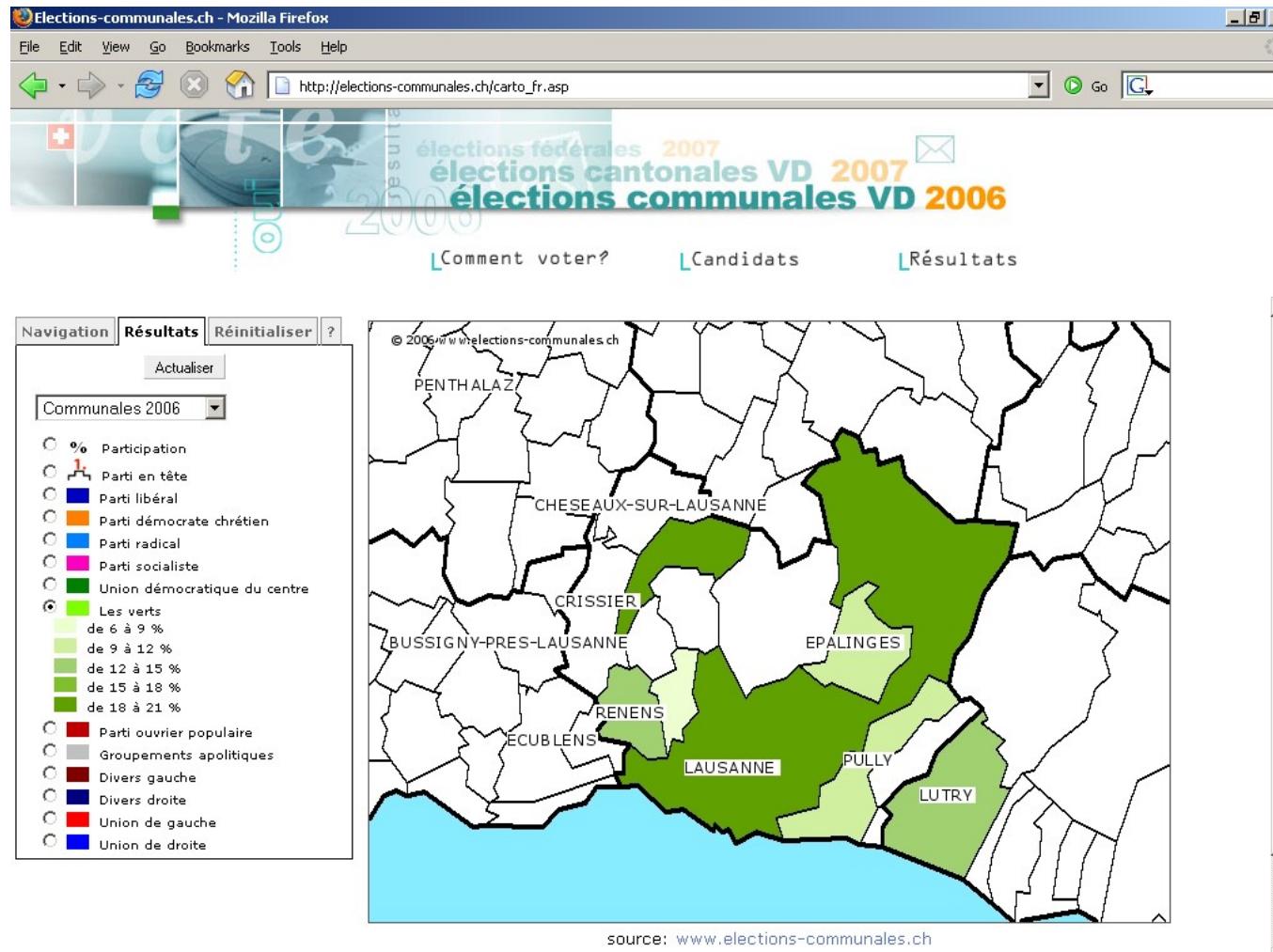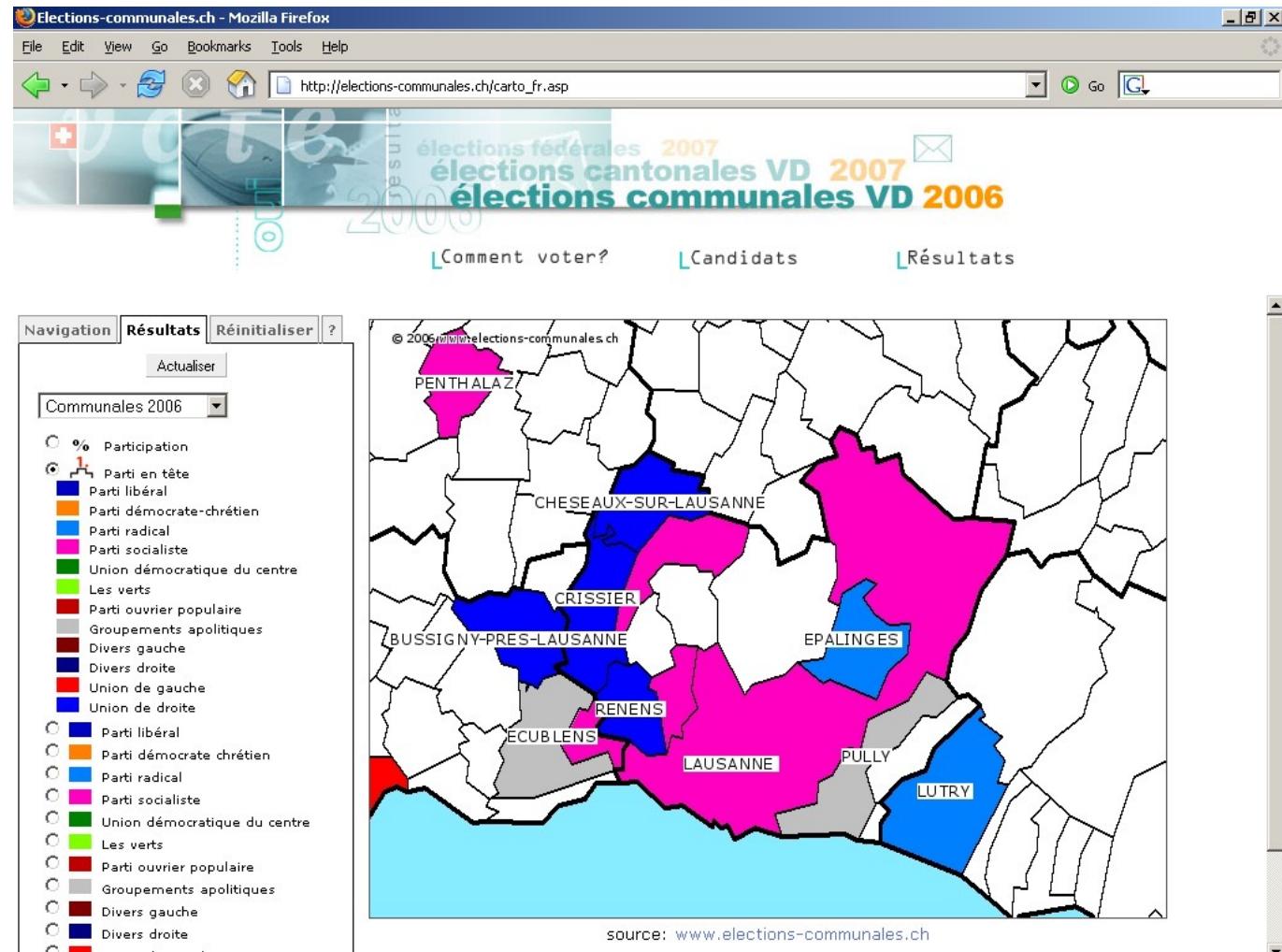
## Slide show: thematics
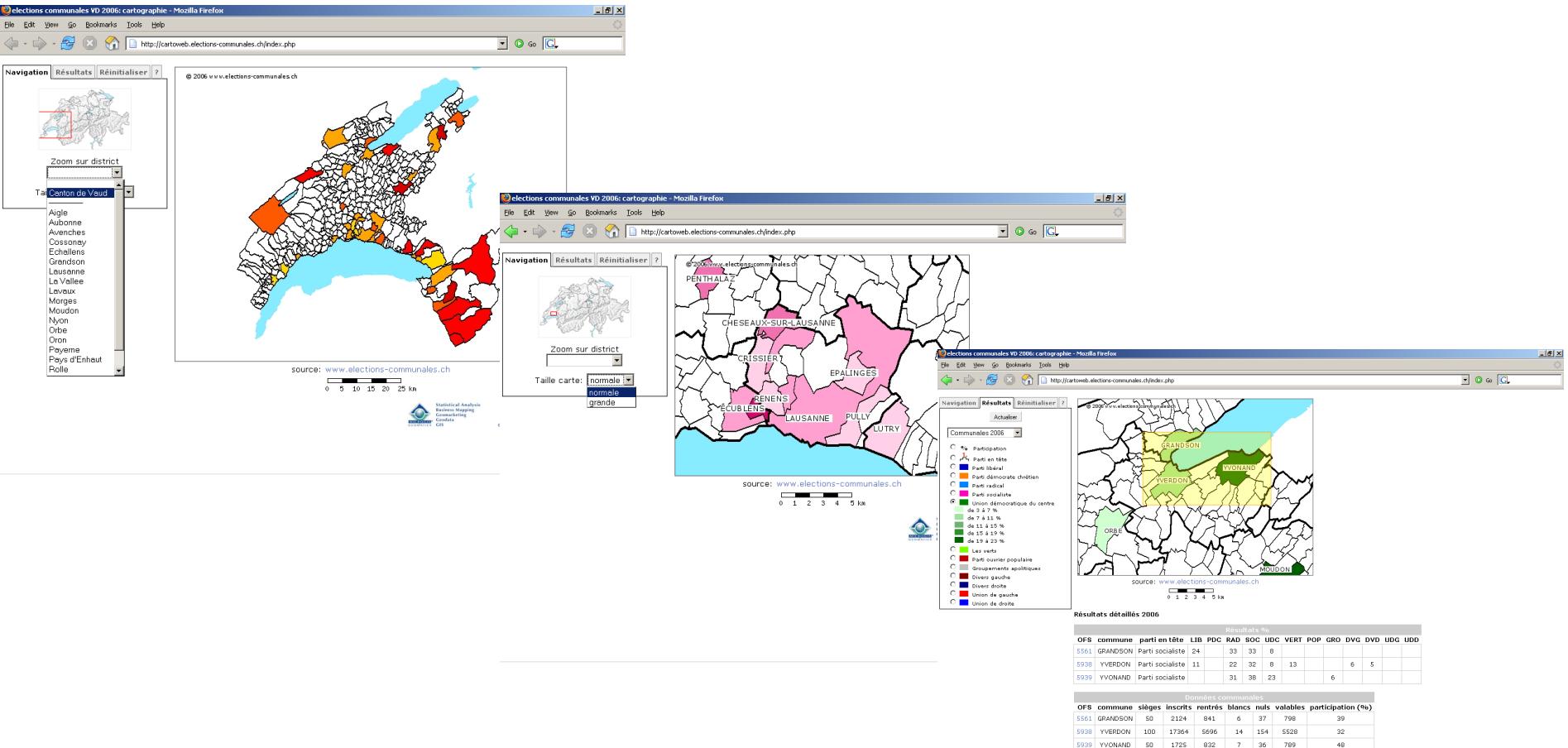
# Introduction
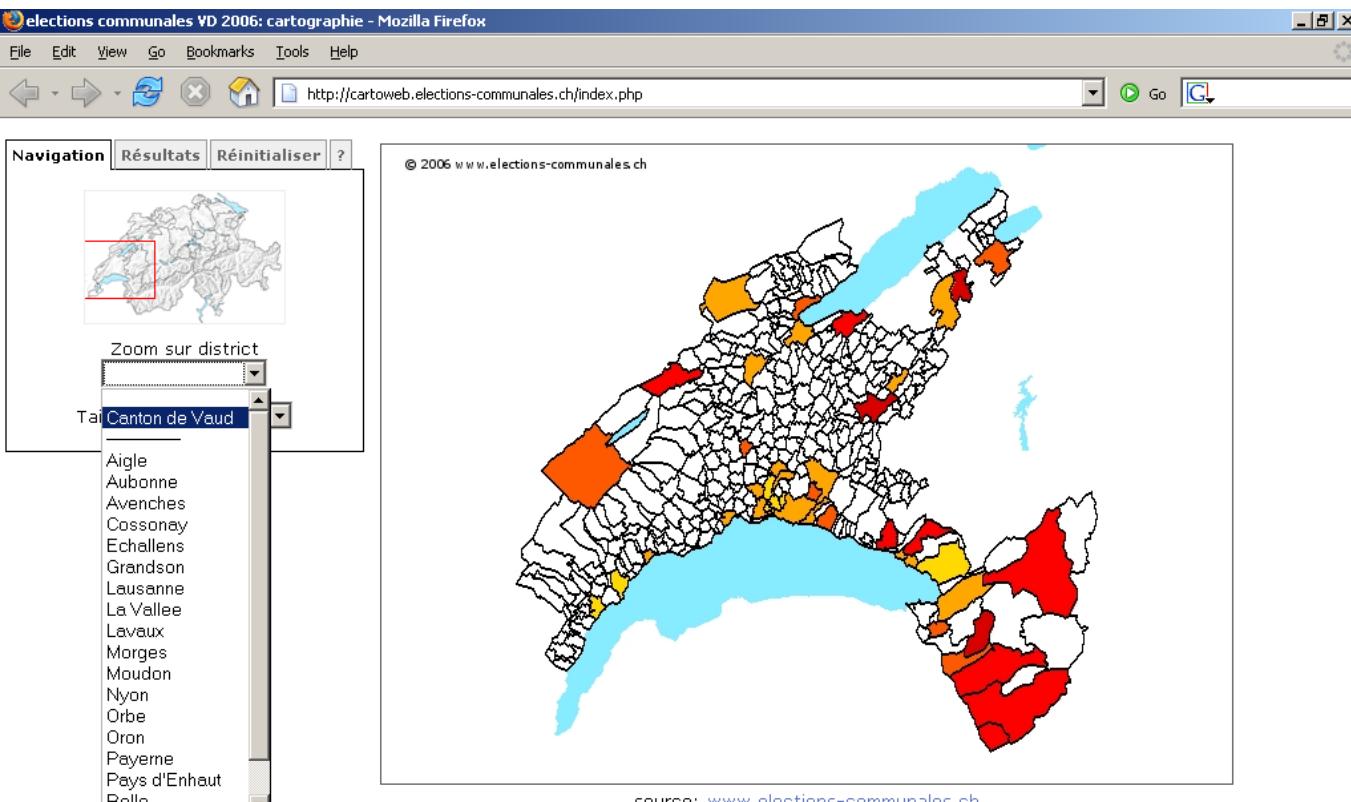
# Introduction

# Introduction

## Slide show: thematics
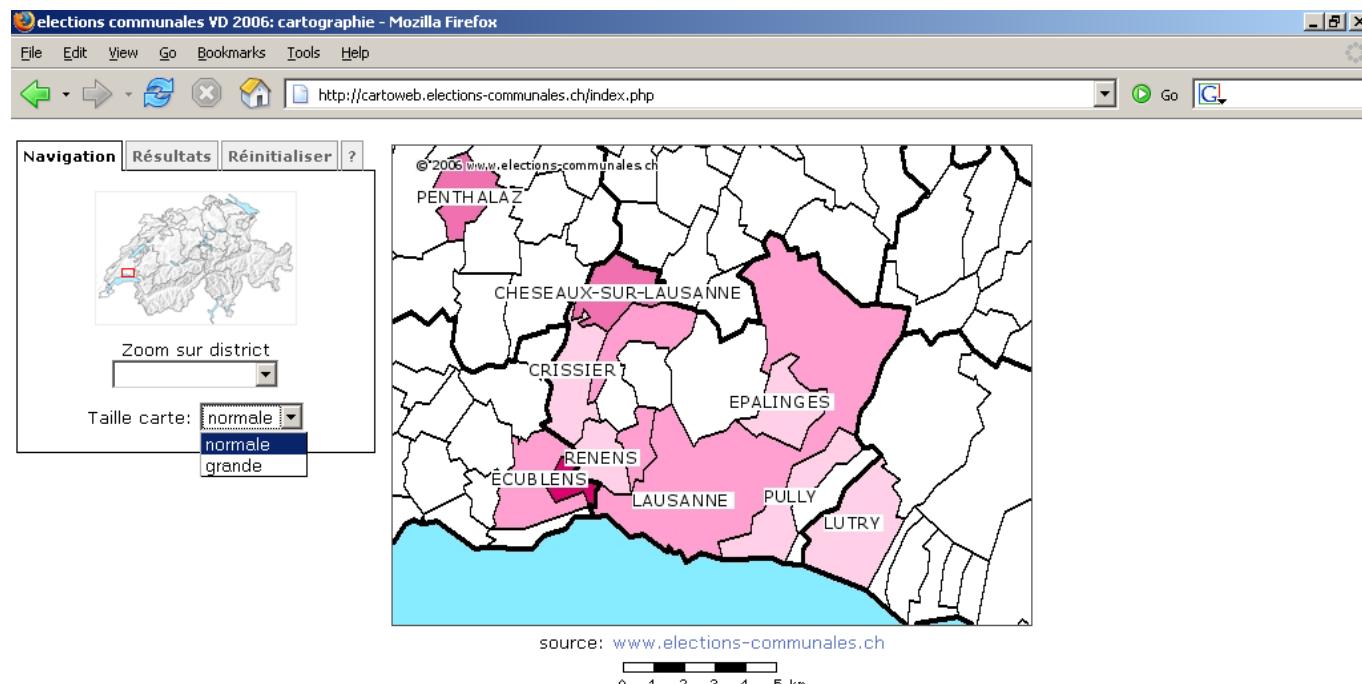
# Introduction

# Introduction
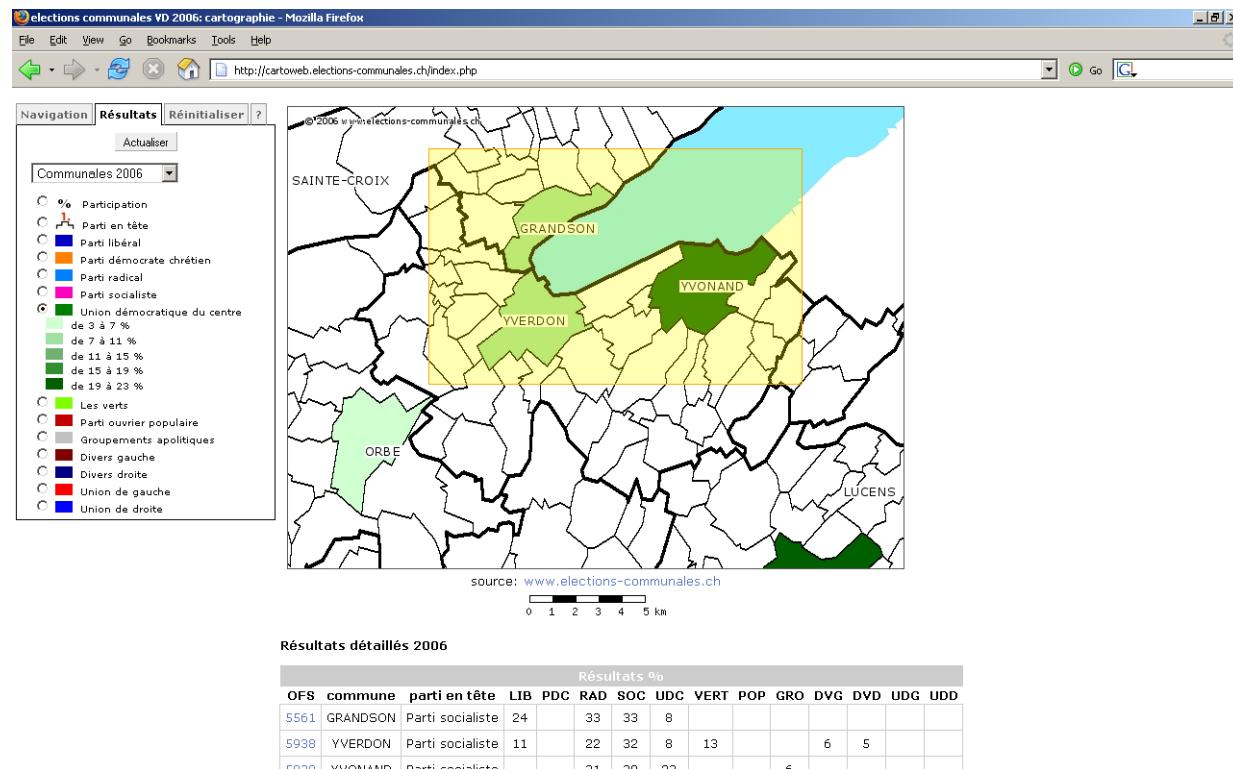
# Introduction

# Introduction

## Processes

- Based on CartoWeb/PostGIS, 4 steps are needed to generate the maps:



HTML     CSV     RDBMS     Mapfile     Image

**AQUISITION**
- Copy&Paste

**STRUCTURING**
- PL/pgSQL scripting
- Materialized views

**CLASSIFICATION**
- autolayers.php
- SQL requests
- makemaps.php
- Mapfile templates

**PRESENTATION**
- IMG Mapserver
- GUI CartoWeb
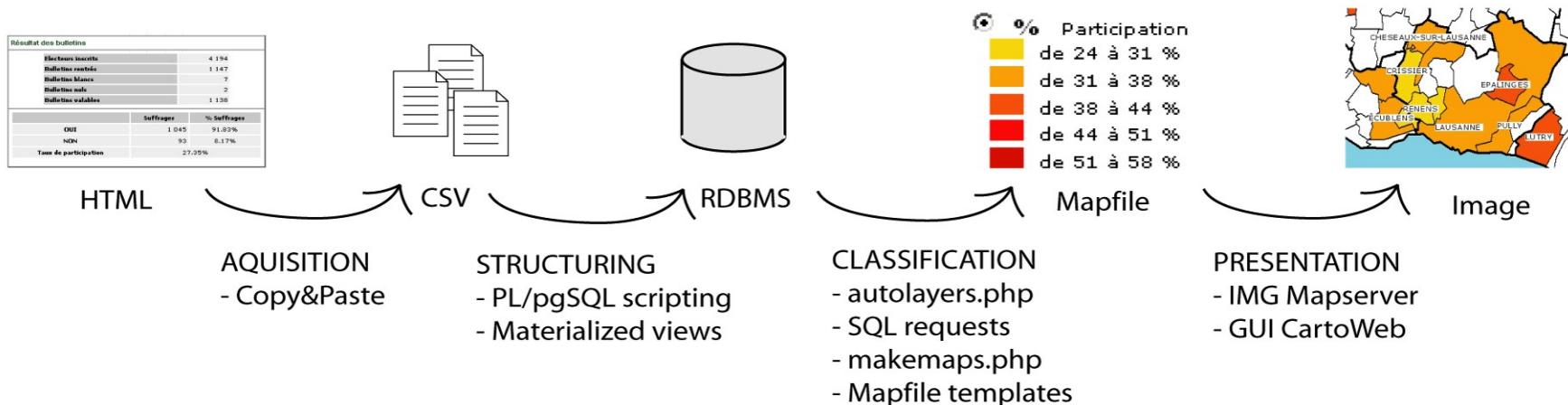
- Generated: 28 thematics; every thematic is automatically divided into 5 class intervals
- The project has to automate these processes and to customize them to serve high traffic.

# Pseudo-dynamic mapfile generation

## Automatic generation of multiple layers

- The **autolayer** functionality from CartoWeb allows to generate multiple layers from a template without any notion of PHP.



| Mapfile template | Descriptions variables | Mapfile | Generated layers |
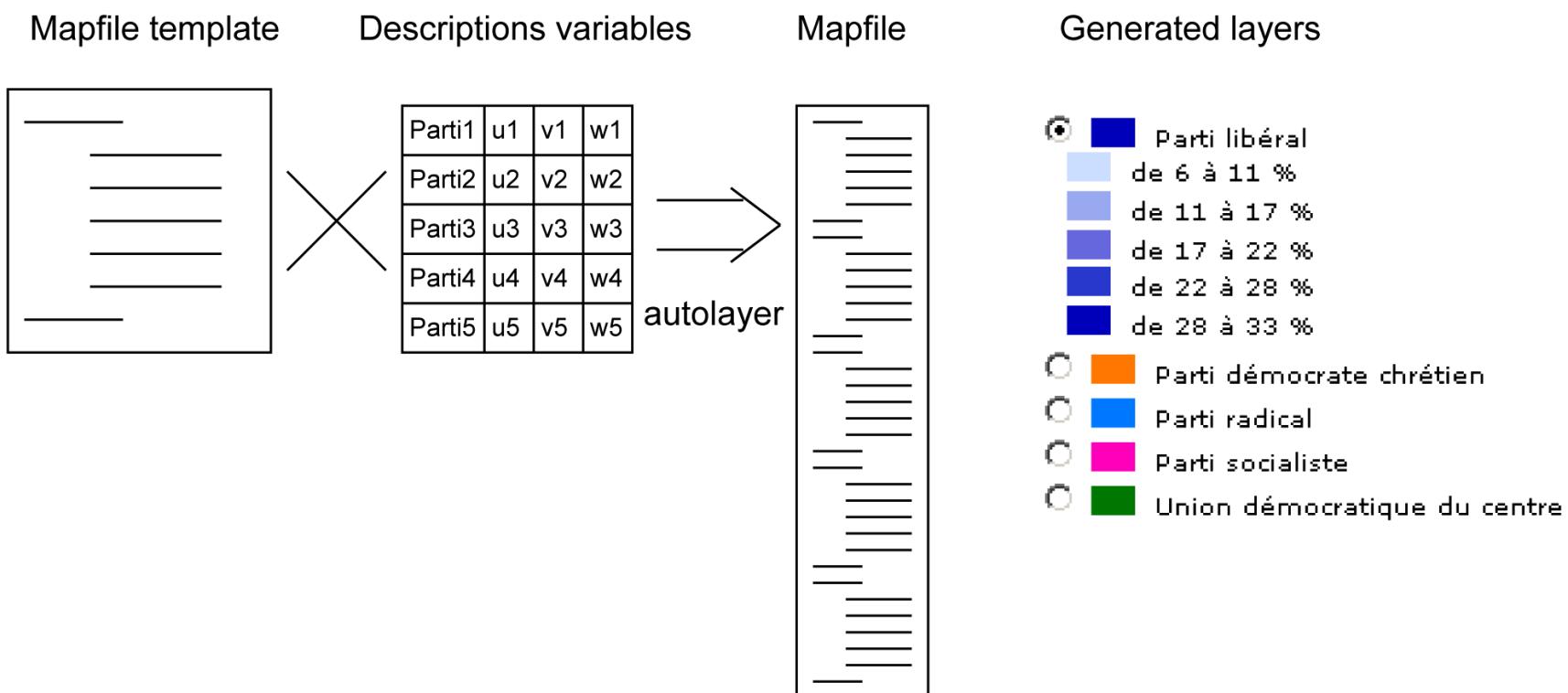
# Pseudo-dynamic mapfile generation

## Automatic adaptation of class intervals

- **makemaps.php** is a script of CartoWeb that allows to parse a PHP template to generate a mapfile.

% Participation

| | |
|---|---|
| 🟨 | de 24 à 31 % |
| 🟧 | de 31 à 38 % |
| 🟧 | de 38 à 44 % |
| 🟥 | de 44 à 51 % |
| 🟥 | de 51 à 58 % |
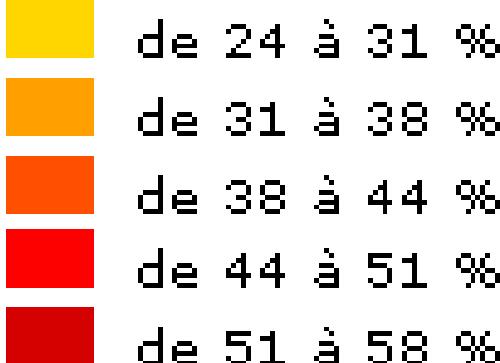
- `<?php`
- `SELECT min(colname), max(colname) FROM layer;`
- `MyFunction computes class intervals: min1, max1, min2, max2, etc`
- `print_f to MAPFILE`
  - `EXPRESSION (([colname] > min1) AND ([colname] <= max1))`
  - `EXPRESSION (([colname] > min2) AND ([colname] <= max2))`
  - `...`

  `?>`

- Whenever new values are added to the database, makemaps.php should be invoked to generate the new adjusted mapfile.
  - `<PHP-INTERPRETER> makemaps.php`

## CLASSIFICATION: pseudo-dynamic Mapfile generation



HTML → CSV → RDBMS → Mapfile → Image

AQUISITION
- Copy&Paste

STRUCTURING
- PL/pgSQL scripting
- Materialized views

CLASSIFICATION
- autolayers.php
- SQL requests
- makemaps.php
- Mapfile templates

PRESENTATION
- IMG Mapserver
- GUI CartoWeb
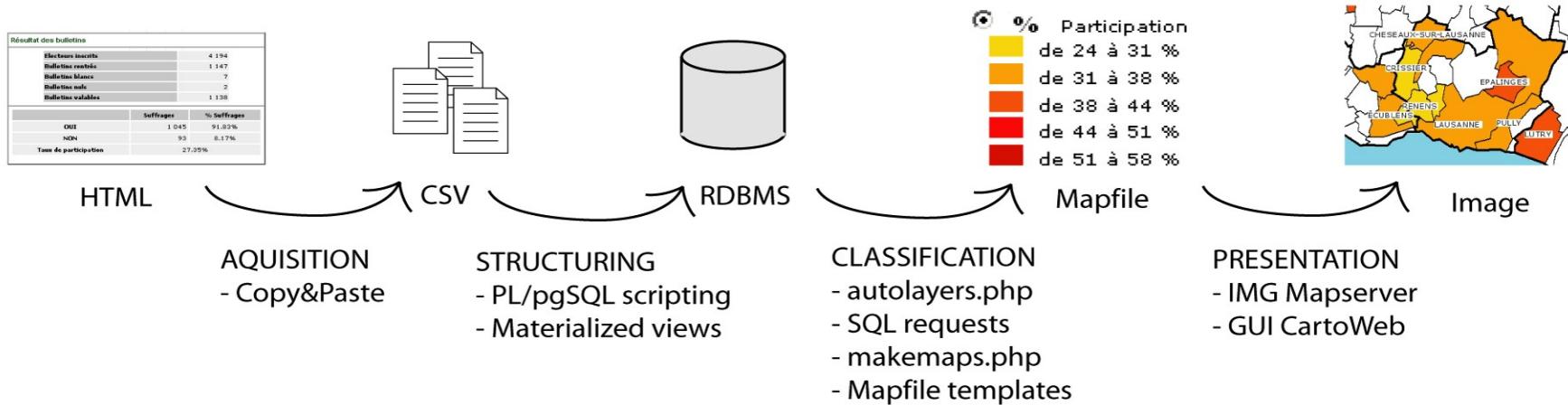
## PRESENTATION: caching images with CartoWeb

# Load / performance tuning for high traffic

## Caching images with CartoWeb  I/II

- Introducing some facts:
    - Mapserver generates an image on request. A list of parameters describes the map.
    - The image is associated to a pseudo unique filename and returned to the browser.

=> Each request, even with identical parameters generates a new image/filename.

- CartoWeb offers a caching mechanism. If the same parameters are invoked, then the cached image is sent again.

=> This is a great functionality, but how can you use it?

## Caching images with CartoWeb  II/II

- To request a cached image, the new request should contain the same parameters as the cached request.

- Which parameters are commonly used in a project?
  - `Mapped layers, Map size & Extent (Zoom&Pan)`

- We have an **infinite quantity of possible maps**, because of free zoom and free pan possibilities.

- For the cache to work, we have to reduce the amount of different maps:
  - 1/ disable free pan
  - 2/ disable free zoom-in, zoom-out

- And to preserve interaction, we have substituted these functionalities with a drop-down menu to access predefined areas. **We reduced the amount of maps to 1120** (14 thematics * 2 years * 20 predefined areas * 2 image sizes).

- Whenever the database id updated, the cache should be cleaned:
  - `<PHP-INTERPRETER> cw3setup.php --clean`

## PRESENTATION: caching images with CartoWeb



HTML → CSV → RDBMS → Mapfile → Image

**AQUISITION**
- Copy&Paste

**STRUCTURING**
- PL/pgSQL scripting
- Materialized views

**CLASSIFICATION**
- autolayers.php
- SQL requests
- makemaps.php
- Mapfile templates

**PRESENTATION**
- IMG Mapserver
- GUI CartoWeb
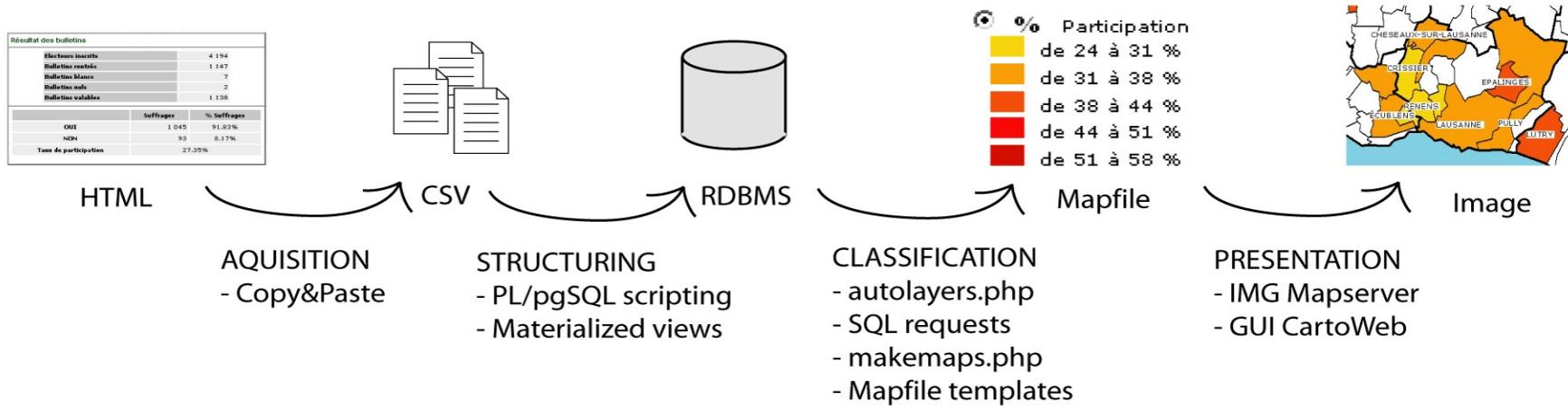
## STRUCTURING: database tuning

## Database tuning I/II

### Original "clean" database design
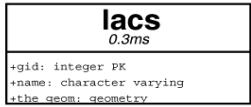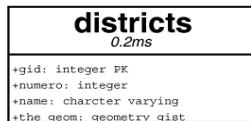
| Static Data | Results to be inserted | Percentages calculation, winner identification | Data joining, evolution calculation |
|---|---|---|---|
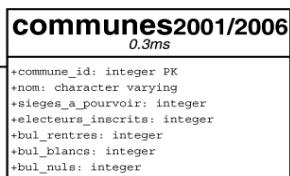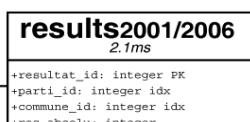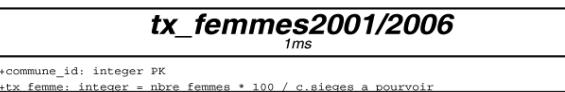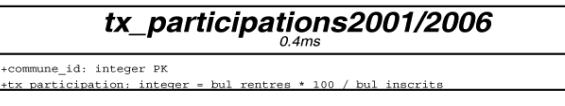


**partis**
*0.2ms*
+parti_id: integer PK
+nom: character varying
+abbreviation: character varying

**communes**
*1.7ms*
+gid: integer PK
+commune_id: integer idx
+the geom: geometry gist

**districts**
*0.2ms*
+gid: integer PK
+numero: integer
+name: character varying
+the geom: geometry gist

**lacs**
*0.3ms*
+gid: integer PK
+name: character varying
+the geom: geometry

**results**2001/2006
*2.1ms*
+resultat_id: integer PK
+parti_id: integer idx
+commune_id: integer idx
+res absolu: integer

**communes**2001/2006
*0.3ms*
+commune_id: integer PK
+nom: character varying
+sieges_a_pourvoir: integer
+electeurs_inscrits: integer
+bul_rentres: integer
+bul_blancs: integer
+bul_nuls: integer
+bul valables: integer

**femmes**2001/2006
*0.3ms*
+commune_id: integer PK
+nbre femmes: integer

*tx_partis2001/2006*
*8.1ms*
+commune_id: integer
+parti_id: integer
+res relatif: integer = res absolu / total suffrages * 100

*parti_en_tetes2001/2006*
*6ms*
+commune_id: integer PK
+parti id: integer = parti id FROM resultats WHERE max(res absolu)

*tx_participations2001/2006*
*0.4ms*
+commune_id: integer PK
+tx participation: integer = bul rentres * 100 / bul inscrits

*tx_femmes2001/2006*
*1ms*
+commune_id: integer PK
+tx femme: integer = nbre femmes * 100 / c.sieges a pourvoir

*tp2001/2006*
+commune_id: integer
+nom: character varying
+tp1: integer
+tp2: integer
+tp3: integer
+tp4: integer
+tp5: integer
+tp6: integer
+tp7: integer
+tp8: integer
+tp9: integer
+tp10: integer
+tp11: integer
+tp12: integer

*evol_partis*
+commune_id: integer
+parti_id: integer
+evolution: integer

*evol_participations*
+commune_id: integer
+evolution: integer

*evol_femmes*
+commune_id: integer
+evolution: integer

table (roman) / *view (italic)*

- Access time to views is a bottleneck => need to optimization
  - Select operations have to be kept simple!

# Load / performance tuning for high traffic

## Database tuning II/II

**fond_query_partis**

+commune_id: integer
+nom: charcter varying
+tp1: integer
+tp2: integer
+tp3: integer
+tp4: integer
+tp5: integer
+tp6: integer
+tp7: integer
+tp8: integer
+tp9: integer
+tp10: integer
+tp11: integer
+tp12: intger
+parti_nom: character varying
+the_geom: geometry GIST

**fond_query_communes**

+commune_id: integer
+nom: charcter varying
+sieges_a_pourvoir: integer
+electeurs_inscrits: integer
+bul_rentres: integer
+bul_blancs: integer
+bul_nuls: integer
+bul_valables: integer
+tx_participation: integer
+the_geom: geometry GIST

**Materialized views used by the application.**

**fond_partis2001/2006**

+commune_id: integer
+nom: character varying
+tp1: integer
+tp2: integer
+tp3: integer
+tp4: integer
+tp5: integer
+tp6: integer
+tp7: integer
+tp8: integer
+tp9: integer
+tp10: integer
+tp11: integer
+tp12: integer
+en_tete: integer
+the_geom: geometry GIST

**fond_communes2001/2006**

+commune_id: integer
+nom: integer
+sieges_a_pourvoir: integer
+electeurs_inscrits: integer
+bul_rentres: integer
+bul_blancs: integer
+bul_nuls: integer
+bul_valables: integer
+tx_participation: integer
+tx_femme: integer
+the_geom: geometry GIST

**fond_evolutions**

+commune_id: integer
+nom: character varying
+tp1: integer
+tp2: intger
+tp3: integer
+tp4: integer
+tp5: integer
+tp6: integer
+tp7: integer
+tp8: integer
+tp9: integer
+tp10: integer
+tp11: integer
+tp12: integer
+participation: integer
+femme: integer
+the_geom: geometry GIST

- No join between tables is required.
- A PL/pgSQL script generates these tables from previous structure.

# Access statistics

- The website is unknown before election day.

- On the election day, the following websites have linked to our site:
    - TSR.ch (state TV)
    - RSR.ch (state Radio)
    - LeTemps.ch (major newspaper)
    - RougeFM.ch (local radio)

- Traffic summary:

|  | Hits | Visitors | Pages Views | Bandwidth | Visit length |
|---|---|---|---|---|---|
| Sunday | 1'614'810 | 5'518 | 50'333 | 6.9 GB | 10:08 |
| Monday | 664'277 | 3'079 | 19'834 | 2.8 GB | 6:58 |

- These figures are equivalent on the Swiss scale to:
    - 17. position in Pages Views
    - 27. position in Visitors                        (source: www.wemf.ch)

# Access statistics

Activity by Hour of Day

dinner & TV evening news

Offices open monday morning

Results are published

Visitors

700
600
500
400
300
200
100
0

0  1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23

Monday AM                    Hour                    Sunday PM
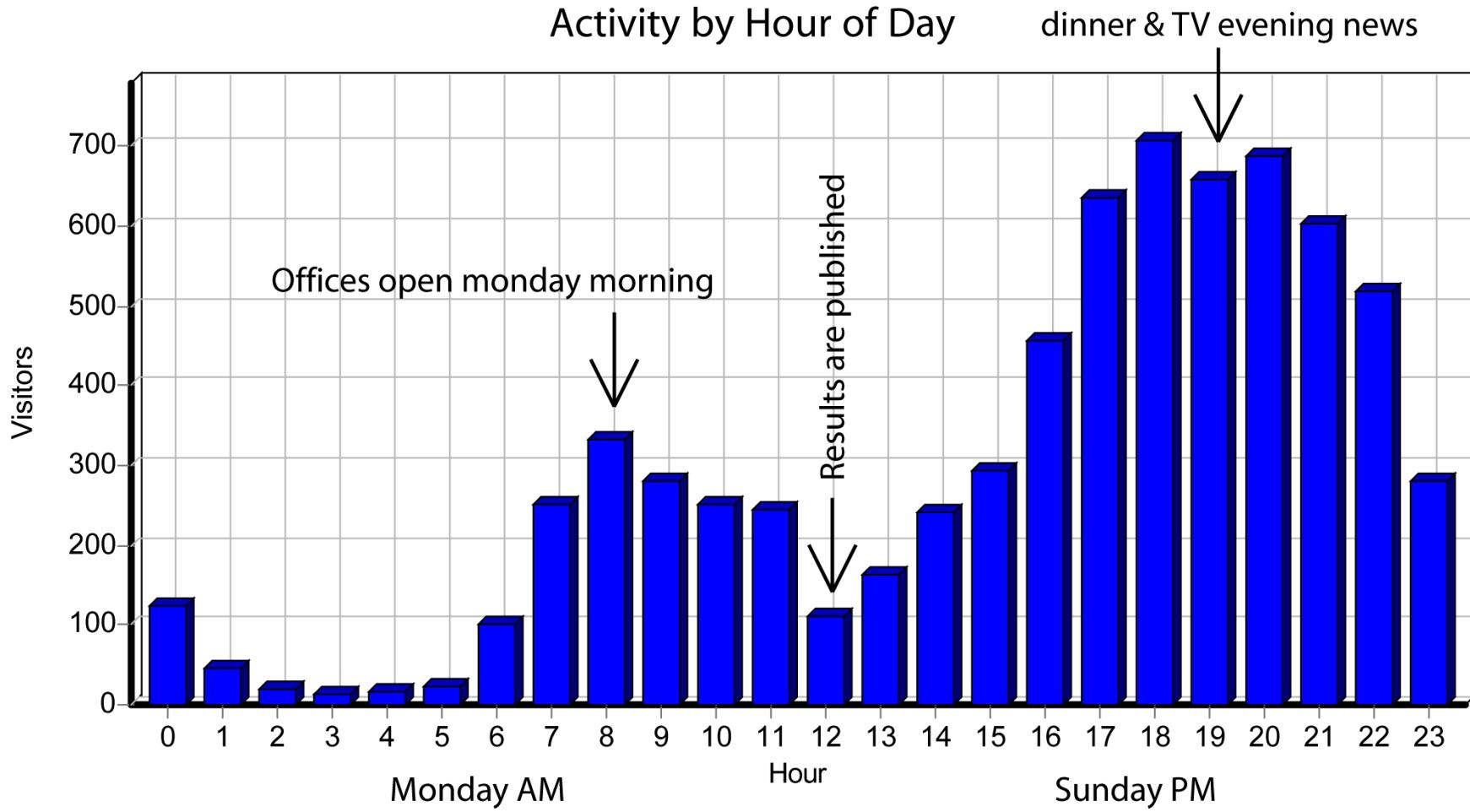
Time range: 12.03.2006 12:00:16 – 13.03.2006 11:59:59

# Conclusions

## Possible improvements

- The webmapping software package CartoWeb/PostGIS is adequate for thematic cartography of news events on a regional basis.

- For larger diffusion, some points should be further evaluated:

  - Reduction of bandwidth use

  - Use of Apache caching mechanism

  - Deployment of a PostgreSQL/PostGIS caching mechanism for queries (info tool)

  - Improvement of CartoWeb caching mechanism: limitation of parallel maps generation

- Richer thematic maps could be implemented (circle proportional, pie charts, etc): see Mapserver or CartoWeb functionalities.

# Mapping election results with CartoWeb

Thank you for your attention!