**Website Classification Design**

Read CSV

DB

API

CSV File

API

no

yes

Finish

Finish

per line

update DB record for line

read API record for line

update\_date + 30 min < current date?

read DB record for line

Start

**DB Design**

Site table

|  |  |  |
| --- | --- | --- |
| Column name | Type | Property |
| URL | string | PK |
| ID | string | Unique |
| category | string |  |
| update\_date | timestamp |  |

Voting table

|  |  |  |
| --- | --- | --- |
| Column name | Type | Property |
| URL | string | PK, FK to site |
| voting\_type | string | PK |
| votes | int |  |

Site\_classification table

|  |  |  |
| --- | --- | --- |
| Column name | Type | Property |
| URL | string | PK, FK to site |
| classification | string | PK |
| votes | int |  |

**AWS implementation**

Since this process writes to a DB, and not returning an on-line response, performance is not a high-priority, but cost is (always is, but sometimes it takes less precedence over other considerations).

This is why I will install the weakest possible EC2 machine to accept the API calls and run the python code, and also store the data on flat files (either CSV or AVRO) on S3, and call them using Athena. Redshift will be an overkill for this, and RDS will be better than S3, but will cost more.