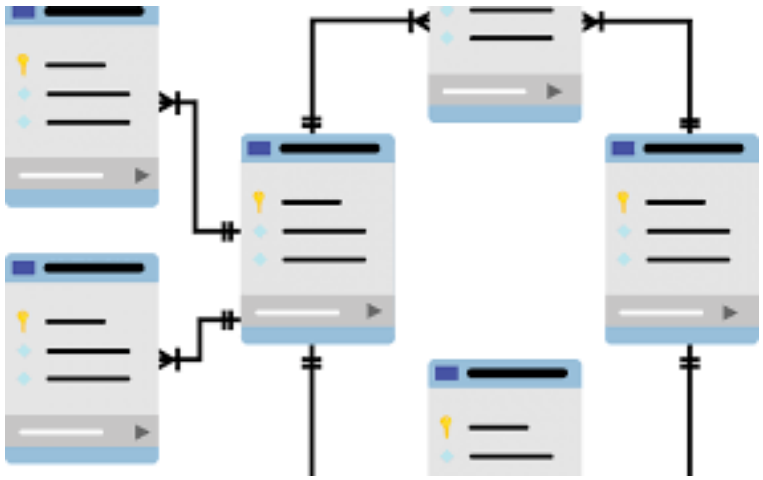
INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES ROUEN NORMANDIE

Rapport de projet de base de données

Base de données de services des citoyens



Diane LEBLANC-ALBAREL
Rand ASSWAD
Génie Mathématique

A l'attentien de :
Mme. Nathalie Chaignaud

21 mai 2019

Table des matières

1 Déscription du projet

1.1 Introduction

Notre projet est une base de données à destination de citoyens et le cadre administratif d'un pays. Le but est de pouvoir répertorier et d'avoir accès aux différents papiers et informations relatifs à un citoyen donné et de pouvoir également faire les démarches en lignes afin d'obtenir des aides, des papiers ou autres, le tout sur la même plateforme.

Actuellement, les plateformes pour percevoir des aides, payer des impôts ou demander certains papiers ou justificatifs sont rarement les mêmes, ce qui peut compliquer les tâches des citoyens et des autorités, qui pourraient avoir besoin de certaines informations sur certains citoyens.

La base de données est adaptée pour deux applications web: l'une est mise à disposition des résidents dans un pays et l'autre pour le corps administratif qui traitera les demandes effectuées sur le premier site.

1.2 Mise-à-jour

Après les retours que vous nous avez fait, nous avons modifié notre modèle et nous l'avons rendu plus concis et fonctionnel.

Le sujet abordé est très complexe, mais grâce à notre nouveau modèle, quelque soit la procédure qu'on souhaite effectuer elle se traduit à quelques requêtes simples.

2 Conception

2.1 Modèle Entité/Association

Après plusieurs discussions nous avons réussi à poser les bases de notre modèle avec les différentes utilisations possibles. Voici donc notre schéma Entité/Association final obtenu.

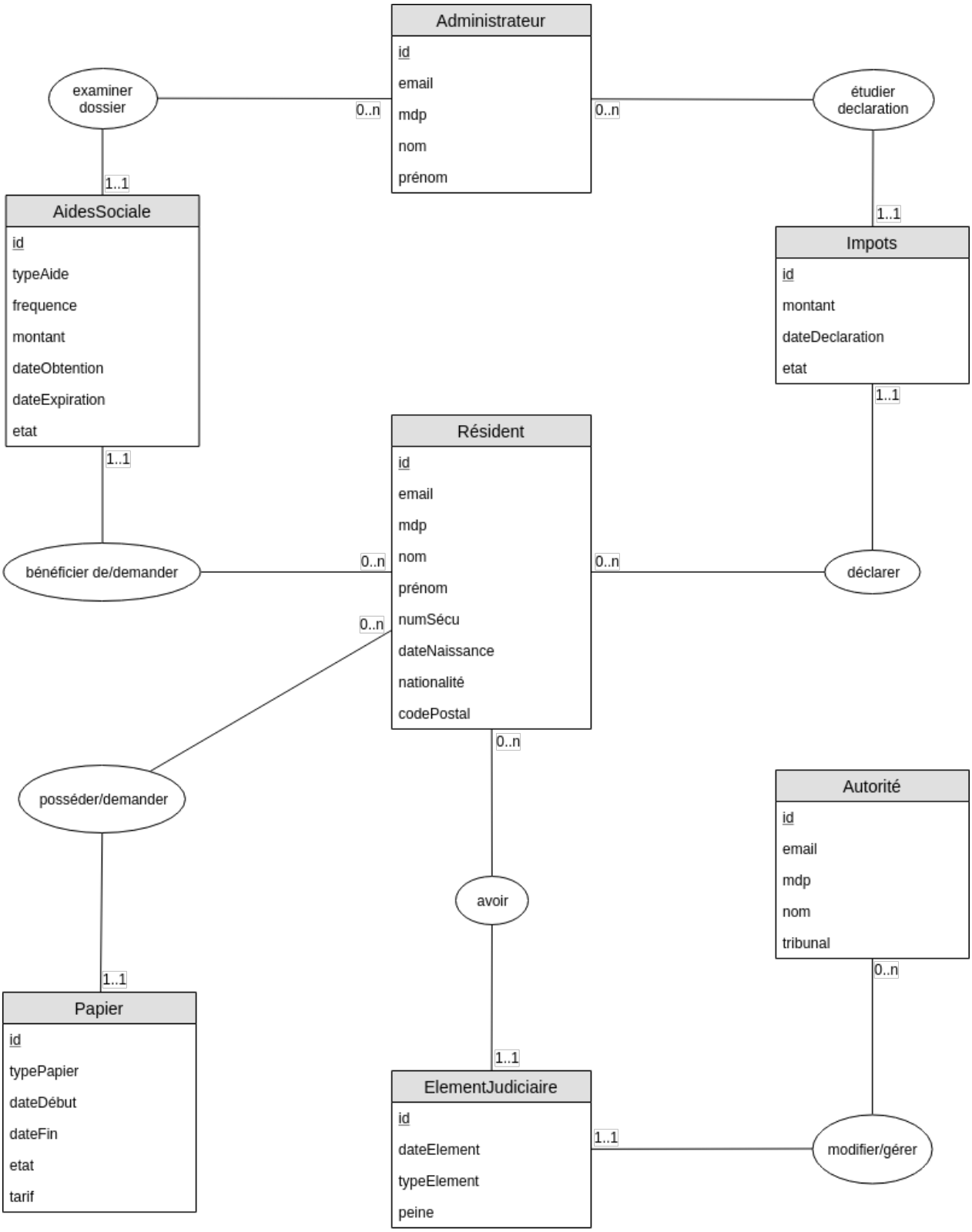


FIGURE 1 – Modèle Entité/Association

Résident

Le résident, comme on le voit sur le diagramme E/A, est le centre de cette application. Il doit pouvoir générer des justificatifs des différents papiers en sa possession (justificatifs de passeport, carte de séjour, carte d'identité, permis de conduire etc.) ainsi que des justificatifs des impôts qu'il auraient pu avoir déjà versé ainsi qu'un justificatif des aides sociales qu'il aurait pu avoir perçu. De plus il doit également être en mesure d'effectuer des démarches pour l'obtention de ces différents papiers et la régularisation de sa situation en faisant différentes déclarations lui permettant de payer ses impôts ou de percevoir des aides. Ainsi le résident peut par exemple effectuer une déclaration de ses revenus qui devra être validé par un administrateur afin qu'il puisse payer les impôts correspondant a sa situation, de même pour les aides sociales.

Administrateur

L'administrateur a pour unique rôle de vérifier les déclarations faites par le résident. Les déclarations concernant ses revenus pour les impôts et les déclarations qu'il auraient effectué pour percevoir certaines bourses ou autres aides sociales.

Autorité

Cette entité peut consulter les informations relatives aux habitants et compléter le casier judiciaire des habitants. Hormis le casier judiciaire des habitant, il ne peut rien modifier mais il a un droit de regard sur toutes les informations concernant les utilisateurs enregistrés en tant que résidents.

2.2 Liens et entités

Papiers

Un résident peut posséder déjà certains papiers, il peut en posséder plusieurs et en possède au moins un : sa carte d'identité. De plus un papier ne peut appartenir qu'à une seule personne. Un résident peut demander un ou plusieurs papiers et sa demande sera muni d'un formulaire. D'où le lien « demander » entre l'entité « Résident » et l'entité « Papier » doublé d'un formulaire. Enfin lors de sa demande de papier ou lorsqu'il consulte les papiers qu'il possède, un résident peut générer une preuve temporaire. Par exemple s'il demande une carte européenne d'assurance maladie il peut demander de générer un papier pouvant remplacer la carte pendant une certaine durée par exemple un mois. Une preuve temporaire ne peut correspondre qu'à un seul papier mais un résident peut demander plusieurs exemplaires de preuves temporaires de papiers.

Impôts et aides sociales

Un résident peut, tout d'abord, générer une preuve ou plusieurs preuves du payements passés de ses impôts, de plus il peut également faire une déclaration en remplissant un formulaire qui devrai être validé par un administrateur afin de pouvoir payer ses impôts. Pour établir une déclaration afin de payer des impôts il faut qu'un Résident remplisse un ou plusieurs formulaires correspondant chacun a un impôt précis (par exemple impôt sur le revenu), formulaires qui devront être validé par un Administrateur pour que la déclaration existe ; Pour les aides sociales on retrouve exactement le même principe qu'avec les impôts pour comprendre cette partie il suffit donc se référer à l'explication ci-dessus

Element judiciaire

Chaque résident possède un casier judiciaire (celui peut être vide ou non). Il peut consulter son casier judiciaire mais ne peut le modifier. Les autorités peuvent consulter le casier judiciaire de chaque résident (comme toutes les autres informations) mais peuvent également le compléter ci besoin en renseignant de nouvelles informations ou en supprimant certaines.

2.3 Schéma relationnel

— Resident(**id**, email, mdp, nom, prenom, numSecu, dateNaissance, nationalite, codePostal)
— Administrateur(**id**, email, mdp, nom, prenom)

— Authorite(**id**, email, mdp, nom, prenom, tribunal)
— Papier(**id**, typePapier, nom, dateDebut, dateFin, etat, tarif, *idResident*)
— AideSociale(**id**, typeAide, frequence, montant, dateObtention, dateExpiration, etat, *idResident*, *idAdmin*)
— Impots(**id**, montant, dateDeclaration, etat, *idResident*, *idAdmin*)
— ElementJudiciaire(**id**, dateElement, typeElement, peine, *idResident*, *idAuthorite*)

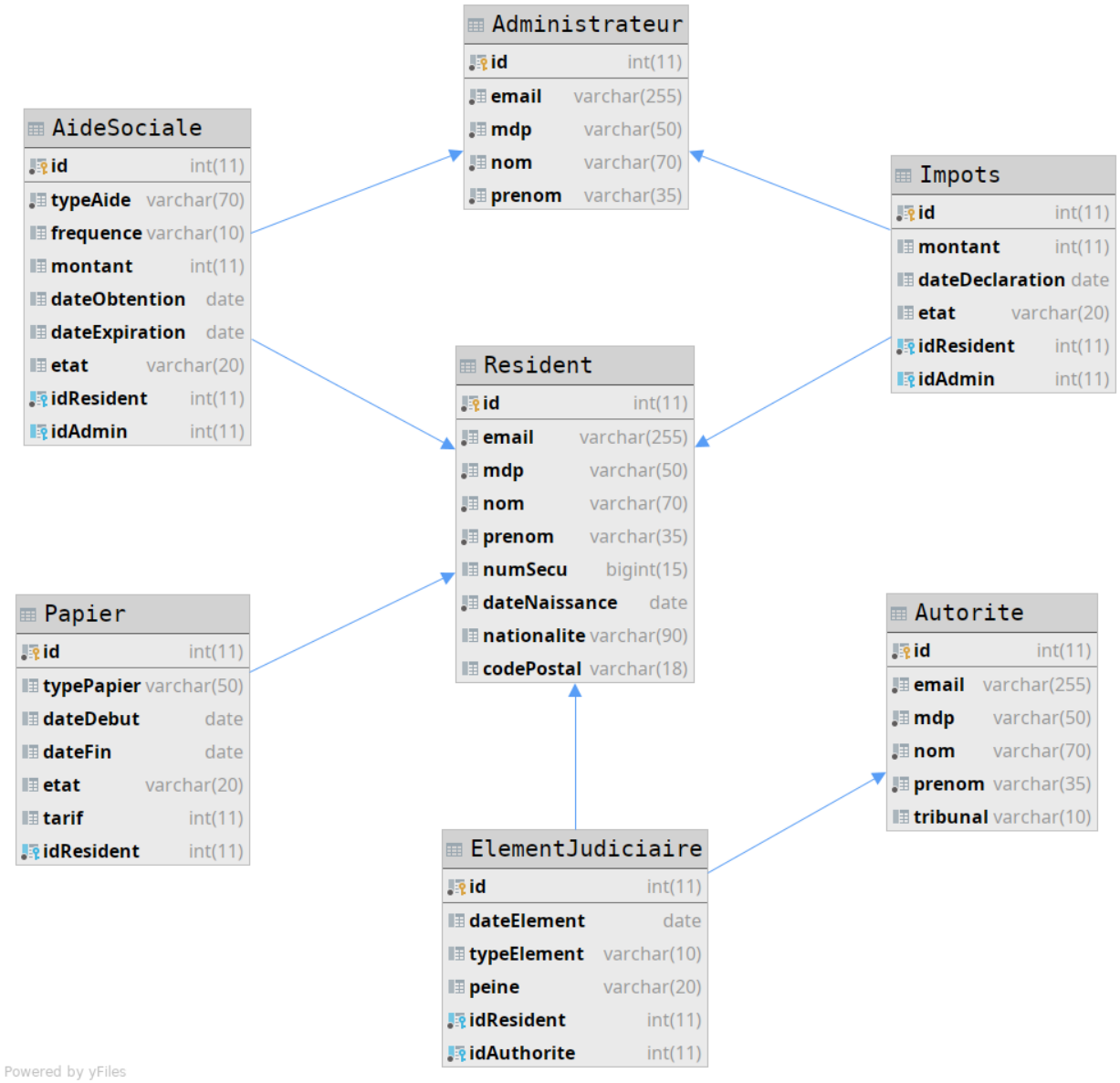


FIGURE 2 – Diagramme du schéma relationnel

3 Implémentation

3.1 Initialisation de la base de donnée

Il s'agit de traduire le schéma relationnel en *MySQL*, voici le fichier `db/tables.sql` permettant de créer la base de données et ses relations.

```
DROP DATABASE IF EXISTS pays;
CREATE DATABASE pays;
USE pays;

CREATE TABLE Resident (
  id INTEGER NOT NULL AUTO_INCREMENT,
  email VARCHAR(255) NOT NULL,
  mdp VARCHAR(50) NOT NULL,
  nom VARCHAR(70) NOT NULL,
  prenom VARCHAR(35) NOT NULL,
  numSecu BIGINT(15),
  dateNaissance DATE NOT NULL,
  nationalite VARCHAR(90),
  codePostal VARCHAR(18),
  PRIMARY KEY (id)
);

CREATE TABLE Administrateur (
  id INTEGER NOT NULL AUTO_INCREMENT,
  email VARCHAR(255) NOT NULL,
  mdp VARCHAR(50) NOT NULL,
  nom VARCHAR(70) NOT NULL,
  prenom VARCHAR(35) NOT NULL,
  PRIMARY KEY (id)
);

CREATE TABLE Autorite (
  id INTEGER NOT NULL AUTO_INCREMENT,
  email VARCHAR(255) NOT NULL,
  mdp VARCHAR(50) NOT NULL,
  nom VARCHAR(70) NOT NULL,
  prenom VARCHAR(35) NOT NULL,
  tribunal VARCHAR(10),
  PRIMARY KEY (id)
);

CREATE TABLE Papier (
  id INTEGER NOT NULL AUTO_INCREMENT,
  typePapier VARCHAR(50),
  dateDebut DATE,
  dateFin DATE,
  etat VARCHAR(20),
  tarif INTEGER,
  idResident INTEGER NOT NULL,
  PRIMARY KEY (id),
  FOREIGN KEY (idResident) REFERENCES Resident(id)
);

CREATE TABLE AideSociale (
  id INTEGER NOT NULL AUTO_INCREMENT,
  typeAide VARCHAR(70) NOT NULL,
  frequence VARCHAR(10),
  montant INT,
```

```
  dateObtention DATE,
  dateExpiration DATE,
  etat VARCHAR(20),
  idResident INTEGER NOT NULL,
  idAdmin INTEGER,
  PRIMARY KEY (id),
  FOREIGN KEY (idResident) REFERENCES Resident(id),
  FOREIGN KEY (idAdmin) REFERENCES Administrateur(id)
);

CREATE TABLE Impots (
  id INTEGER NOT NULL AUTO_INCREMENT,
  montant INT,
  dateDeclaration DATE,
  etat VARCHAR(20),
  idResident INTEGER NOT NULL,
  idAdmin INTEGER,
  PRIMARY KEY (id),
  FOREIGN KEY (idResident) REFERENCES Resident(id),
  FOREIGN KEY (idAdmin) REFERENCES Administrateur(id)
);

CREATE TABLE ElementJudiciaire (
  id INTEGER NOT NULL AUTO_INCREMENT,
  dateElement DATE,
  typeElement VARCHAR(10),
  peine VARCHAR(20),
  idResident INTEGER NOT NULL,
  idAuthorite INTEGER NOT NULL,
  PRIMARY KEY (id),
  FOREIGN KEY (idResident) REFERENCES Resident(id),
  FOREIGN KEY (idAuthorite) REFERENCES Autorite(id)
);
```

Ensuite, nous avons écrit des scripts (en *python* cf. `db/data`) permettant de générer des données aléatoires mais assez pertinentes afin de pouvoir tester nos requêtes et d'avoir un aperçu de l'application du projet.

Après avoir populé la base de données, nous avons sauvegardé les données dans des scripts MySQL sous forme de `INSERT` pour des raisons de portabilité (cf. `db/populate`).

3.2 Environnement

Ce rapport a été écrit en *r markdown* et ensuite traduit en LaTeX et HTML à l'aide de *knitr* et *pandoc*. Cela pour dire qu'à partir de ce point, le code dans ce rapport a été interprété par *knitr* qui rajoute ensuite les résultats obtenus au rapport. Le code source est disponible sur le répértoire github github.com/minikara/bdd-pays.

De plus, nous avons limité les résultats affichés dans le PDF (pour des raisons de visibilité) donc nous recommandons la lecture de cette partie sous sa forme HTML sur minikara.github.io/bdd-pays car les résultats sont complets et plus lisibles.

Commeçons alors par l'établissemnt de la connection avec la base de donnée à l'aide de la librairie R `DBI`

```
library(DBI)
pays <- dbConnect(RMySQL::MySQL(), dbname="pays", username="root")
```

3.3 Visualisation des relations

```
SELECT * FROM Resident;
```

Table 1 – résultats 1..30 (total: 100)

id	email	mdp	nom	prenom	numSecu	dateNaissance	nationalite	codePostal
1	Sed.congue.elit@in.ca	FFO07WWH5BF	Caron	Maëlle	1.668100e+12	1944-06-19	France	3117
2	Aenean@quisdiamPellentesque.edu	ZTP06XOJ5CW	Herve	Kimberley	1.697052e+12	2017-04-10	France	21909
3	molestie@AliquamnislNulla.com	SXH35ZVE0LR	Pons	Quentin	1.600022e+12	1992-05-23	France	12444
4	sollicitudin.commodo.ipsum@Duis.com	TFQ15CRR3TU	Lemoine	Maïlé	1.677051e+12	1992-07-18	France	16092
5	lobortis.mauris.Suspendisse@mauriserateget.org	SXK69NYT4QB	Julien	Éloïse	1.602012e+12	2004-11-20	France	X2S 3T0
6	consequat@Aliquamvulputateullamcorper.com	CQY08SAS5UV	Le gall	Simon	1.629111e+12	1939-06-27	France	66121
7	enim@dolor.com	JBC62RLA0XZ	Daniel	Cédric	1.694102e+12	1951-06-26	Northern Mariana Islands	26098
8	eget@scelerisqueduiSuspendisse.ca	WJE08HZV4OG	Guillot	Adam	1.681053e+12	1957-05-27	France	912415
9	tortor@metus.net	SIB74QPV6XS	Renard	Yanis	1.635022e+12	1979-12-22	France	486292
10	ut@nec.ca	KHY48ULC8IN	Menard	Yasmine	1.681091e+12	1956-01-03	France	81353
11	ac.orci.Ut@Cras.ca	BRW37RHT8OK	Roussel	Corentin	1.643111e+12	1991-12-13	France	P0L 8A1
12	hendrerit.id.ante@gravida.ca	GRG43DQC2OE	Marchand	Félix	1.676022e+12	2018-05-29	Slovenia	63377
13	parturient.montes.nascetur@eu.co.uk	BKB25ZBS8UQ	Gauthier	Gabin	1.628033e+12	1954-05-08	France	40-152
14	dignissim.tempor@tellusPhasellus.org	JYL63MPH8OQ	Gonzalez	Robin	1.690121e+12	1979-11-07	France	01315-619
15	Donec.sollicitudin@consequatdolor.net	DOY46FVH0GG	Brunet	Jade	1.670101e+12	1959-08-11	Venezuela	58871
16	scelerisque.mollis.Phasellus@Phasellusataugue.com	IKW42MEI0TU	Francois	Gaspard	1.672020e+12	1959-01-07	France	5091 PD
17	dis@Curabituregestasnunc.ca	EED22LWZ7KH	Gautier	Salomé	1.613112e+12	1962-04-17	France	7998
18	pharetra.ut.pharetra@sociisnatoque.co.uk	TON13YHZ4TO	Clement	Kilian	1.677062e+12	2005-05-02	France	2588
19	ornare.In@eleifendnunc.edu	VTS30RZM8BM	Lefebvre	Alexia	1.666021e+12	1999-10-05	Åland Islands	50606
20	Phasellus.vitae@aliquamiaculis.edu	TPD40OAY0MX	Henry	Dorian	1.643011e+12	1932-10-12	France	60885-926
21	metus.In.lorem@Cum.edu	LIJ55MEG6EK	Noel	Loane	1.618062e+12	1970-10-09	Holy See (Vatican City State)	4968 EQ
22	dignissim@purussapien.ca	CYW72KEO3XO	Prevost	Anaël	1.645111e+12	1943-03-15	France	R6C 9G7
23	et.magnis@lorem.edu	QZN50SDS4OA	Bouvier	Maxence	1.627042e+12	1974-12-08	France	3578 NY
24	massa@Cras.org	SWT16KVS9DP	Leclerc	Félix	1.687100e+12	2005-06-12	France	98501
25	Duis@maurisSuspendisse.org	XAF34PEM4TK	Millet	Nolan	1.654101e+12	1928-06-08	France	6009
26	elit.pharetra@Aeneaneget.org	QEL53ESM3HL	Huet	Jeanne	1.646030e+12	1992-06-11	France	4137 OZ
27	risus.at.fringilla@adipiscing.edu	BQK30ICH8ZJ	Lucas	Gabin	1.616091e+12	1932-08-22	France	Y3X 7J3
28	gravida.sagittis.Duis@torquentperconubia.net	POC31MVO0IR	Marechal	Émile	1.663031e+12	2008-05-10	France	94149
29	ut@eutellus.com	AJX84YMA3NI	Boucher	Diego	1.691052e+12	1985-06-12	Åland Islands	83889-718
30	cursus.et.eros@volutpat.edu	SRY37BKT3YF	Garnier	Lisa	1.643071e+12	1959-09-01	France	4041

```
SELECT * FROM Administrateur;
```

Table 2 – résultats 1..30 (total: 50)

id	email	mdp	nom	prenom
1	auctor@turpisvitaepurus.org	YQP03FRH8QX	Charpentier	Maéva
2	hendrerit.id.ante@arcu.org	NEG98YKV3HR	Martinez	Bienvenue
3	nunc.Quisque.ornare@pharetra.org	RUE79QBN8LF	Nicolas	Noë
4	enim@mollisduiin.com	GLO71RVC5BG	Dumont	Renaud
5	fames.ac@enim.co.uk	VHR96VIN2WV	Gay	Syrine
6	ipsum@sed.net	QJH37JLU0TJ	Girard	Hugo
7	ultrices.posuere@magnaaneque.org	FLX93PDJ6SS	Marchand	Louis
8	magna@tempus.com	SAI78XKA3JT	Meyer	Alexandre
9	Suspendisse.ac@MorbivehiculaPellentesque.net	INV45QMG2HF	Deschamps	Justine
10	Donec.est.mauris@tinciduntpedeac.co.uk	KIW18CZB1UC	Marchand	Timéo
11	eu.sem.Pellentesque@eget.net	QSM21RGU5BL	Roux	Jeanne
12	dictum.placerat.augue@vitaeerat.ca	MZR63IMM8YX	Pereira	Jade
13	Quisque.libero.lacus@gravidasagittisDuis.ca	CMW89PEW3AR	Royer	Clémence
14	ante@auguescelerisquemollis.edu	TRZ62GVI0VL	Rousseau	Antonin
15	augue.porttitor.interdum@parturientmontesnascetur.edu	BFS90QIQ8CO	Nguyen	Louise
16	ante.lectus.convallis@CurabiturdictumPhasellus.ca	VPY35XRF9ZO	Dupont	Tatiana
17	eu.eleifend@nullaDonecnon.edu	VCB62FJJ7DB	Simon	Marine
18	enim.Nunc.ut@mi.ca	ZHL29TYM3ZF	Guyot	Gilbert
19	cursus.et@maurisid.edu	FXM39DEZ0JJ	Remy	Lorenzo
20	senectus.et.netus@commodotinciduntnibh.net	QNO57JNI9VE	Paul	Lilou
21	odio@ametante.edu	EFO81NVJ0RT	Roger	Justine
22	cursus.vestibulum.Mauris@dictumPhasellus.co.uk	MIU04VMT1JC	Gauthier	Paul
23	neque@Aliquamerat.edu	LMO08PAJ4WC	Collin	Yanis
24	tempus.mauris@aarcuSed.edu	GOU10PJC7MR	Herve	Chaïma
25	fames.ac@dignissimtempor.ca	ANC81ITL9OC	Dumas	Carla
26	arcu@sitametdapibus.ca	XNE62EPQ1BH	Pierre	Elsa
27	ultrices.posuere.cubilia@amet.com	XVH51QPK0TS	Brunet	Anaëlle
28	a.feugiat@enimnisl.ca	VJV75AJY3WN	Joly	Timéo
29	vitae.orci.Phasellus@bibendum.com	ALU19EAC0QK	Moulin	Malik
30	diam@eu.net	OHI81JPV9QN	Bonnet	Malik

```
SELECT * FROM Autorite;
```

Table 3 – résultats 1..30 (total: 50)

id	email	mdp	nom	prenom	tribunal
1	sem@cursusin.edu	RKR16WOX2ZF	Roche	Pauline	null
2	orci@tinciduntaliquam.net	CEF98OTV6ST	Petit	Dimitri	null
3	eget@acturpisegestas.ca	ZGJ79LWH4WC	Dubois	Florian	null
4	mi.enim.condimentum@ut.com	IVH25XVA8DJ	Fabre	Esteban	null
5	congue@eget.ca	VIE14CVG3FD	Renard	Bienvenue	null
6	Aliquam.auctor.velit@sagittislobortismauris.edu	VBR38YEI8AH	Rodriguez	Nathan	null
7	libero.Morbi.accumsan@augue.net	PJV75HCA5JF	Martinez	Émilie	null
8	pulvinar.arcu.et@eratvolutpatNulla.ca	HED17XOK1VA	Guyot	Marine	null
9	dignissim@diamatpretium.co.uk	ESP85TTI8LD	Francois	Loane	null
10	mus.Proin.vel@arcuVestibulumut.edu	DXM42IZZ6AN	Aubry	Maïlé	null
11	dui@tacitisociosquad.co.uk	FQB71SBX2FH	Bailly	Catherine	null
12	neque@Aliquam.ca	WAU78KZO9PY	Meyer	Killian	null
13	montes.nascetur@etultrices.edu	FPA06KMH7ZE	Lebrun	Samuel	null
14	euismod@aliquet.net	OVG14FLL4VZ	Leroux	Lamia	null
15	natoque@elitdictumeu.com	EJQ80RZX3WE	Meyer	Adrien	null
16	eget.venenatis.a@nullaIntincidunt.com	QQS01AAS7DV	Garnier	Florentin	null
17	adipiscing.lacus.Ut@ultricesDuisvolutpat.net	MGT51NEF5GI	Dubois	Émilie	null
18	massa.Quisque@enimnectempus.net	ASF53TRH0PV	Dubois	Jade	null
19	sodales@magna.com	RID84SUD1JD	Michel	Maxime	null
20	Morbi@purusNullamscelerisque.co.uk	FWQ32FIR1TU	Gauthier	Lou	null
21	Nunc.mauris@posuerecubiliaCurae.ca	THC91NZZ9CL	Sanchez	Maéva	null
22	nec.imperdiet.nec@justo.net	GEN44VYY2HD	Robin	Bienvenue	null
23	pede@Fuscemilorem.ca	IJN89VCN7CY	Renaud	Kylian	null
24	aliquet.Proin.velit@vellectusCum.ca	DWB58USM9QG	Le gall	Léo	null
25	mauris.sagittis@fringillaporttitorvulputate.co.uk	TPV18IBF9NM	Royer	Léa	null
26	nunc.nulla.vulputate@aliquamerosturpis.com	DNU62SVZ2VF	Berger	Thomas	null
27	Nunc@amet.net	BHH16BIE2JB	Jacob	Ambre	null
28	vel.sapien@tinciduntduiaugue.ca	CEU50EHX9NU	Bouvier	Lina	null
29	ornare.facilisis.eget@aliquet.com	GHU08AIG1ZT	Petit	Lily	null
30	Cum.sociis@ultricies.com	QQK32VBG2NM	Boucher	Agathe	null

```
SELECT * FROM AideSociale;
```

Table 4 – résultats 1..30 (total: 500)

id	typeAide	frequence	montant	dateObtention	dateExpiration	etat	idResident	idAdmin
1	RSA	m	246	2017-03-05	2018-02-28	Expire	90	41
2	PrimeNaissance	p	562	2017-04-14	2017-04-14	Expire	18	35
3	CMU	a	302	2016-08-08	2017-08-03	Expire	72	41
4	CMU	a	401	2015-08-09	2016-08-03	Expire	36	50
5	RSA	m	284	2016-03-24	2017-03-19	Expire	60	44
6	CROUS	m	402	2017-08-06	2018-08-01	Expire	22	50
7	CAF	m	62	2011-04-28	2012-04-22	Expire	59	26
8	CAF	m	null	2019-02-22	null	EnCoursDeValidation	4	36
9	RSA	m	164	2013-12-27	2014-12-22	Expire	19	5
10	CROUS	m	178	2017-02-02	2018-01-28	Expire	43	3
11	CAF	m	180	2014-05-04	2015-04-29	Expire	77	5
12	CROUS	m	294	2013-04-29	2014-04-24	Expire	30	11
13	CAF	m	156	2013-02-10	2014-02-05	Expire	68	47
14	RSA	m	null	2014-10-18	null	Refuse	17	8
15	PrimeNaissance	p	529	2012-07-02	2012-07-02	Expire	27	48
16	RSA	m	158	2014-08-11	2015-08-06	Expire	38	50
17	CROUS	m	null	2017-02-14	null	Refuse	29	33
18	PrimeNaissance	p	747	2014-11-04	2014-11-04	Expire	7	23
19	CMU	a	258	2012-11-13	2013-11-08	Expire	38	25
20	CAF	m	null	2016-05-23	null	Refuse	69	48
21	CMU	a	403	2018-10-21	2019-10-16	Valide	95	38
22	RSA	m	397	2016-07-07	2017-07-02	Expire	11	47
23	CMU	a	431	2011-03-21	2012-03-15	Expire	89	20
24	PrimeNaissance	p	null	2019-03-11	null	EnCoursDeValidation	75	7
25	CROUS	m	null	2012-10-24	null	Refuse	50	43
26	PrimeNaissance	p	657	2012-04-04	2012-04-04	Expire	99	39
27	CROUS	m	432	2016-10-03	2017-09-28	Expire	87	28
28	CAF	m	null	2019-03-28	null	EnCoursDeValidation	20	33
29	RSA	m	490	2013-08-26	2014-08-21	Expire	78	26
30	CMU	a	null	2013-11-22	null	Refuse	1	36

```
SELECT * FROM Impots;
```

Table 5 – résultats 1..30 (total: 500)

id	montant	dateDeclaration	etat	idResident	idAdmin
1	1767	2015-08-23	Valide	77	8
2	1855	2017-07-02	Valide	27	16
3	455	2018-08-05	Valide	20	3
4	1754	2017-06-29	Valide	74	23
5	3992	2018-05-19	Valide	27	33
6	null	2018-09-28	Refuse	65	29
7	null	2016-01-16	Refuse	80	47
8	203	2018-05-13	Valide	42	48
9	null	2015-11-16	Refuse	57	14
10	364	2016-06-08	Valide	71	28
11	null	2018-08-09	Refuse	45	8
12	3363	2017-03-22	Valide	10	23
13	null	2019-01-10	Refuse	57	18
14	2053	2017-04-15	Valide	71	50
15	null	2018-06-19	Refuse	89	9
16	null	2015-12-13	Refuse	2	47
17	1206	2016-03-03	Valide	57	11
18	1614	2018-01-08	Valide	52	25
19	null	2014-08-23	Refuse	7	22
20	3610	2014-01-17	Valide	31	35
21	null	2016-05-18	Refuse	28	31
22	null	2019-01-31	EnCoursDeValidation	51	9
23	1601	2016-07-04	Valide	51	29
24	111	2015-06-01	Valide	82	12
25	null	2019-03-12	EnCoursDeValidation	3	29
26	null	2019-04-28	EnCoursDeValidation	72	6
27	353	2015-07-02	Valide	28	18
28	1238	2018-11-25	Valide	29	6
29	794	2014-06-25	Valide	63	3
30	2003	2016-09-20	Valide	48	27

```
SELECT * FROM ElementJudiciaire;
```

Table 6 – résultats 1..30 (total: 150)

id	dateElement	typeElement	peine	idResident	idAuthorite
1	2007-03-05	null	NonElligible	68	12
2	2015-04-06	null	NonElligible	45	22
3	2005-04-10	null	NonElligible	50	9
4	2012-02-06	null	NonElligible	24	34
5	2017-09-05	null	EnCoursDExecution	64	3
6	2007-07-05	null	NonElligible	23	43
7	2014-05-16	null	NonElligible	15	25
8	2012-03-12	null	EnCoursDExecution	51	3
9	2012-01-10	null	NonElligible	24	16
10	2001-02-01	null	NonElligible	15	14
11	2001-08-07	null	NonElligible	13	39
12	2017-04-13	null	NonElligible	4	39
13	2000-08-24	null	EnCoursDExecution	24	23
14	2002-05-13	null	NonElligible	60	16
15	2009-12-29	null	NonElligible	47	7
16	2005-11-19	null	Execute	34	39
17	2014-01-24	null	EnCoursDExecution	4	33
18	2018-11-18	null	NonElligible	14	2
19	2007-02-13	null	NonElligible	45	18
20	2001-08-23	null	NonElligible	44	18
21	2008-12-26	null	NonElligible	1	15
22	2008-11-16	null	NonElligible	68	20
23	2002-06-29	null	NonElligible	17	38
24	2006-10-10	null	NonElligible	28	18
25	2018-10-03	null	NonElligible	23	41
26	2006-10-18	null	Execute	26	6
27	2014-05-01	null	NonElligible	61	40
28	2014-09-20	null	NonElligible	24	40
29	2018-01-11	null	NonElligible	12	24
30	2004-04-15	null	Execute	62	4

```
SELECT * FROM Papier;
```

Table 7 – résultats 1..30 (total: 500)

id	typePapier	dateDebut	dateFin	etat	tarif	idResident
1	carteId	2005-09-13	2020-06-26	Valide	0	71
2	carteId	1986-07-28	2001-02-09	Expire	0	58
3	passeport	2011-02-19	2020-07-31	Valide	86	26
4	livretFamille	1962-05-31	null	Valide	0	5
5	livretFamille	1962-10-21	null	Valide	0	12
6	livretFamille	1975-05-30	null	Valide	0	1
7	carteElectorale	2005-10-24	2009-10-03	Expire	0	20
8	livretFamille	1980-01-30	null	Valide	0	30
9	passeport	2012-04-06	2021-02-18	Valide	86	67
10	carteId	2019-04-08	null	EnCoursDeValidation	0	36
11	livretFamille	2009-02-27	null	Valide	0	33
12	livretFamille	1994-06-29	null	Valide	0	88
13	carteElectorale	2019-04-26	null	EnCoursDeValidation	0	12
14	titreDeSejour	2018-04-05	2019-06-29	Valide	47	88
15	carteElectorale	2019-04-25	null	EnCoursDeValidation	0	13
16	livretFamille	1971-07-31	null	Valide	0	42
17	carteId	2004-09-24	2019-07-08	Valide	0	47
18	carteElectorale	2018-10-22	2021-05-09	Valide	0	6
19	passeport	2010-05-30	2019-12-09	Valide	86	91
20	passeport	2010-06-16	2019-08-28	Valide	86	20
21	titreDeSejour	2019-03-19	null	EnCoursDeValidation	47	77
22	livretFamille	1977-11-09	null	Valide	0	42
23	carteElectorale	2010-03-24	2015-02-26	Expire	0	53
24	livretFamille	1968-08-30	null	Valide	0	93
25	livretFamille	1978-12-20	null	Valide	0	74
26	carteId	2006-10-14	2020-12-29	Valide	0	29
27	carteElectorale	2016-03-30	2020-04-08	Valide	0	67
28	passeport	2011-03-15	2020-09-23	Valide	86	13
29	livretFamille	1974-12-23	null	Valide	0	2
30	titreDeSejour	2016-08-08	2019-07-24	Valide	47	62

3.4 Requêtes

Liste des aides sociales que perçoit un résident donné.

- **Reformultion :** Liste de toutes les aides **où** idResident $= X$
- **Algèbre relationelle :**

$$\sigma_{\text{idResident}=<X>}(\text{AideSociale})$$

- **SQL:** On donne une valeur pour X

```
X <- 44
```

```
SELECT * FROM AideSociale WHERE idResident = ?X;
```

TABLE 8 – résultats 1..6 (total: 6)

id	typeAide	frequence	montant	dateObtention	dateExpiration	etat	idResident	idAdmin
119	CROUS	m	null	2019-05-19	null	EnCoursDeValidation	44	14
223	CAF	m	null	2015-09-03	null	Refuse	44	3
292	RSA	m	227	2014-02-21	2015-02-16	Expire	44	39
397	CMU	a	337	2018-08-07	2019-08-02	Valide	44	33
483	RSA	m	360	2017-11-02	2018-10-28	Expire	44	2
488	RSA	m	140	2019-04-22	2020-04-16	Valide	44	7

Combien chaque résident a payé en impôt sur les derniers 10 ans ?

- **Reformulation :**
  - Liste d'impôts *Validés* **où** la date est $\geq$ année en cours -10 (sélections)
  - Le montant payé (projection)
  - La somme des montants de la liste par résident (fonction agrégation)
- **Algèbre relationnelle :**

$$I = \pi_{\text{montant, idResident}}(\sigma_{\text{date}\geq<\text{AnnéeEnCours}>-10 \quad \wedge \quad \text{etat}=\text{"valide"}}(\text{Impots}))$$

  Afin d'obtenir la somme, nous n'avons qu'à faire: ${}_{\text{idResident}}\gamma_{\text{somme(montant)}}(\sigma_{\text{idResident}=<X>}(I))$
- **SQL :**

```
SELECT R.id, R.nom, R.prenom, R.nationalite, SUM(montant) FROM Impots, Resident R
    WHERE etat = "Valide"
    AND idResident = R.id
    AND dateDeclaration >= YEAR(CURDATE()) - 10
    GROUP BY idResident;
```

Table 9 – résultats 1..25 (total: 25)

id	nom	prenom	nationalite	SUM(montant)
1	Caron	Maëlle	France	6364
2	Herve	Kimberley	France	3282
3	Pons	Quentin	France	1200
4	Lemoine	Maïlé	France	974
5	Julien	Éloïse	France	3330
6	Le gall	Simon	France	5376
7	Daniel	Cédric	Northern Mariana Islands	1656
8	Guillot	Adam	France	11184
9	Renard	Yanis	France	5592
10	Menard	Yasmine	France	8880
11	Roussel	Corentin	France	17729
12	Marchand	Félix	Slovenia	6280
13	Gauthier	Gabin	France	2779
14	Gonzalez	Robin	France	17240
15	Brunet	Jade	Venezuela	14150
16	Francois	Gaspard	France	1776
17	Gautier	Salomé	France	19046
19	Lefebvre	Alexia	Åland Islands	894
20	Henry	Dorian	France	9045
21	Noel	Loane	Holy See (Vatican City State)	2818
23	Bouvier	Maxence	France	7412
24	Leclerc	Félix	France	2181
25	Millet	Nolan	France	4328
26	Huet	Jeanne	France	7570
27	Lucas	Gabin	France	7247

Si on cherche pour un résident en particulier, il suffit de faire

$$\gamma_{\text{somme(montant)}}(\sigma_{\text{idResident=<X>}}(I))$$

```
SELECT SUM(montant) FROM Impots
    WHERE etat = "Valide"
    AND idResident = ?X
    AND dateDeclaration >= YEAR(CURDATE()) - 10;
```

Table 10 – résultats 1..1 (total: 1)

x
2779

Est-ce que la demande d'allocation RSA a été validé pour un résident donné ?

- **Reformulation :** état de l'aide sociale **où** idResident = X **et** typeAide = "RSA"
- **Algèbre relationnelle :**

$$\pi_{\text{etat}}(\sigma_{\text{idResident=<}X\text{>}}(\sigma_{\text{typeAide="RSA"}}(\text{AideSociale})))$$

- **SQL :**

```
SELECT etat FROM AideSociale
    WHERE idResident = ?X
    AND typeAide = "RSA";
```

Table 11 – résultats 1..3 (total: 3)

x
Expire
Expire
Valide

Est ce que tel résident purge actuellement une peine ?

- **Reformulation :**
  - liste des elements du casier judiciaire **où** idResident = X **et** peine est en cours d'exécution.
  - si la liste est vide alors la réponse est "non" sinon "oui"
- **Algèbre relationnelle :**

$$\sigma_{\text{idResident}=<X>}(\sigma_{\text{peine}=\text{"EnCoursDExecution"}}(\text{ElementJudicaire})$$

- **SQL :**

```
SELECT * FROM ElementJudiciaire
    WHERE idResident = ?X
    AND peine = "EnCoursDExecution";
```

Table 12 – résultats 1..1 (total: 1)

id	dateElement	typeElement	peine	idResident	idAuthorite
117	2006-01-20	null	EnCoursDExecution	44	9

* (Possibilité de faire une projection sur une peine si on veut simplement avoir la liste des peines 'EnCoursDExecution' cela dépend de l'usage choisit. Afficher tous les éléments si la peine est en cours d'exécution à l'avantage de permettre d'en savoir plus sur la dite peine).

Effectuer une demande de renouvellement d'un passeport

Dans notre vision de cette application il s'agit d'une procédure qui ne demande pas l'intervention d'un *Administrateur* car toutes les informations nécessaires sont déjà dans cette base de données. Par exemple, en supposant qu'un citoyen qui purge une peine n'a pas le droit d'obtenir un passeport il s'agit donc de faire la requête précédente. Pour le cas d'un passeport, le tarif et la durée sont déjà fixés, il reste plus qu'à remplir la base de donné par ces informations récupérées. Une instance sera donc créée avec l'état "EnCoursDeValidation" jusqu'à la vérification du paiement.

```
INSERT INTO Papier (idResident, typePapier, dateDebut, dateFin, etat, tarif)
VALUES (?X, "passeport", CURRENT_DATE(), CURRENT_DATE() + 10, "EnCoursDeValidation", 86);
```

Lister tout les administrateur dans l'ordre de nombre de tâches dont ils sont chargés

On voudrait étendre la relation Administrateur en y rajoutant le nombre de tâches dont chaque administrateur est chargé actuellement, cette requête est très utile dans le cas où on voudrait automatiser l'attribution des tâches.

Petite remarque: on aurait pu rendre l'application plus réaliste en rajoutant un champs *disponibilité* qui indiquera si l'employé est disponible ou pas (en arrêt maladie, congé maternel/paternel, vacances, etc), ce qui exclura les employés non disponible des résultats de cette requête.

Plongeons nous alors dans cette requête !

- Un administrateur est chargé d'examiner les demandes d'aides sociales et des déclarations d'impots, on voudrais donc d'abord séléctionner tous les éléments de ces deux relations dont l'état est *en cours de validation*.
- Ensuite, on fait une jointure externe gauche entre Administrateur et la relation obtenue.
- A l'aide de la fonction d'agrégation de comptage, on compte le nombre d'occurances par administrateur.

$$T = \pi_{\text{id,idAdmin}}(\sigma_{\text{etat}=\text{"EnCoursDeValidation"}}(\text{AideSociale})) \cup \pi_{\text{id,idAdmin}}(\sigma_{\text{etat}=\text{"EnCoursDeValidation"}}(\text{Impots}))$$

$$_{\text{id}}\gamma_{\text{compter(T.id)}}(\text{Administrateur} \underset{\text{id=T.idAdmin}}{\bowtie} T)$$

```sql
SELECT Administrateur.id, email, nom, prenom, COUNT(ALL T.id) as nbTaches
FROM Administrateur
LEFT OUTER JOIN (
    SELECT id, idAdmin FROM AideSociale WHERE etat = "EnCoursDeValidation"
    UNION ALL SELECT id, idAdmin FROM Impots WHERE etat = "EnCoursDeValidation"
) AS T ON T.idAdmin = Administrateur.id
GROUP BY Administrateur.id
ORDER BY nbTaches;
```

Table 13 – résultats 1..30 (total: 50)

id	email	nom	prenom	nbTaches
49	mauris.Integer@erat.ca	Laine	Élise	0
17	eu.eleifend@nullaDonecnon.edu	Simon	Marine	0
38	ullamcorper.Duis.cursus@purusinmolestie.co.uk	Dupuy	Esteban	0
45	malesuada@vitae.net	Philippe	Clémence	0
16	ante.lectus.convallis@CurabiturdictumPhasellus.ca	Dupont	Tatiana	0
30	diam@eu.net	Bonnet	Malik	0
1	auctor@turpisvitaepurus.org	Charpentier	Maéva	0
40	Aliquam.erat.volutpat@asollicitudinorci.edu	Gauthier	Amine	0
47	nonummy.ac.feugiat@ipsum.edu	Nicolas	Arthur	0
22	cursus.vestibulum.Mauris@dictumPhasellus.co.uk	Gauthier	Paul	0
35	risus.at.fringilla@hendrerit.ca	Bourgeois	Gaspard	0
42	cursus.purus@congue.ca	Dupuis	Lilou	1
31	ullamcorper.Duis@velitegestaslacinia.com	Masson	Sara	1
6	ipsum@sed.net	Girard	Hugo	1
13	Quisque.libero.lacus@gravidasagittisDuis.ca	Royer	Clémence	1
34	euismod.enim.Etiam@massaQuisque.com	Chevalier	Juliette	1
41	auctor.Mauris.vel@cursuspurus.co.uk	David	Thibault	1
23	neque@Aliquamerat.edu	Collin	Yanis	1
37	elit@pedeNuncsed.ca	Carre	Gabriel	1
5	fames.ac@enim.co.uk	Gay	Syrine	1
44	semper.pretium.neque@mollisDuis.edu	Louis	Catherine	1
12	dictum.placerat.augue@vitaeerat.ca	Pereira	Jade	1
26	arcu@sitametdapibus.ca	Pierre	Elsa	1
8	magna@tempus.com	Meyer	Alexandre	1
36	Nullam.velit.dui@aliquameu.edu	Boyer	Maéva	1
4	enim@mollisduiin.com	Dumont	Renaud	1
43	justo.Proin@cursuseteros.edu	Albert	Lamia	1
25	fames.ac@dignissimtempor.ca	Dumas	Carla	1
39	Nunc.mauris.sapien@dictumPhasellusin.net	Renault	Lauriane	1
21	odio@ametante.edu	Roger	Justine	1

Donner pour chaque aide sociale attribuée le montant total reçu

Afin de calculer le montant reçu pour une aide sociale il faut prendre en compte sa fréquence, sa date d'obtention et sa date d'expiration.

En effet, on introduit une fonction **M(m,f,deb,fin)** qui calcule le montant total où:

- **m:** montant unitaire.
- **f:** fréquence qui prend l'une des valeurs (annuelle, monsuelle, ou poncuelle).
- **deb:** date d'obtention de l'aide.
- **fin:** date de fin de l'aide.

$$A_T = {}_{\text{id}}\gamma_{\text{M(montant, frequence, dateObtention, dateExpiration)}}(\text{AideSociale})$$

```sql
DROP VIEW IF EXISTS AideTotale;
```

```sql
CREATE VIEW AideTotale AS
SELECT id,
    CASE frequence
        WHEN 'a' THEN montant * (timestampdiff(year, dateObtention, dateExpiration))
        WHEN 'm' THEN montant * (timestampdiff(month, dateObtention, dateExpiration))
```

```
        ELSE montant
    END AS montantTotal
FROM AideSociale;
```

```
SELECT AideSociale.id, typeAide, frequence, montant, montantTotal,
    dateObtention, dateExpiration, idResident
FROM AideSociale, AideTotale
WHERE AideSociale.id = AideTotale.id;
```

TABLE 14 – résultats 1..30 (total: 500)

id	typeAide	frequence	montant	montantTotal	dateObtention	dateExpiration	idResident
1	RSA	m	246	2706	2017-03-05	2018-02-28	90
2	PrimeNaissance	p	562	562	2017-04-14	2017-04-14	18
3	CMU	a	302	0	2016-08-08	2017-08-03	72
4	CMU	a	401	0	2015-08-09	2016-08-03	36
5	RSA	m	284	3124	2016-03-24	2017-03-19	60
6	CROUS	m	402	4422	2017-08-06	2018-08-01	22
7	CAF	m	62	682	2011-04-28	2012-04-22	59
8	CAF	m	null	null	2019-02-22	null	4
9	RSA	m	164	1804	2013-12-27	2014-12-22	19
10	CROUS	m	178	1958	2017-02-02	2018-01-28	43
11	CAF	m	180	1980	2014-05-04	2015-04-29	77
12	CROUS	m	294	3234	2013-04-29	2014-04-24	30
13	CAF	m	156	1716	2013-02-10	2014-02-05	68
14	RSA	m	null	null	2014-10-18	null	17
15	PrimeNaissance	p	529	529	2012-07-02	2012-07-02	27
16	RSA	m	158	1738	2014-08-11	2015-08-06	38
17	CROUS	m	null	null	2017-02-14	null	29
18	PrimeNaissance	p	747	747	2014-11-04	2014-11-04	7
19	CMU	a	258	0	2012-11-13	2013-11-08	38
20	CAF	m	null	null	2016-05-23	null	69
21	CMU	a	403	0	2018-10-21	2019-10-16	95
22	RSA	m	397	4367	2016-07-07	2017-07-02	11
23	CMU	a	431	0	2011-03-21	2012-03-15	89
24	PrimeNaissance	p	null	null	2019-03-11	null	75
25	CROUS	m	null	null	2012-10-24	null	50
26	PrimeNaissance	p	657	657	2012-04-04	2012-04-04	99
27	CROUS	m	432	4752	2016-10-03	2017-09-28	87
28	CAF	m	null	null	2019-03-28	null	20
29	RSA	m	490	5390	2013-08-26	2014-08-21	78
30	CMU	a	null	null	2013-11-22	null	1

3.4.1 Combien chaque résident reçoit en aides sociales ?

On reprend la vue qu'on vient de créer, et ensuite on fait la somme par résident.

- Jointure interne de AideTotale et AideSociale sur le *id*
- Jointure interne avec le résident sur *idResident*
- La somme des montants totaux groupés par résident

Remarque: on aurait pu mettre tous les champs de la relation AideSociale dans la vue AideTotale ce qui nous évitera de faire la première jointure. On justifie notre choix par le fait qu'on a préféré obtenir une vue minimale.

$$_{\text{R.id}}\gamma_{\text{somme(montantTotal)}}(\text{Resident} \underset{\text{R.id=idResident}}{\bowtie} (\text{AideSociale} \underset{\text{id}}{\bowtie} A_T))$$

On rajoute le nom, prénom, et la nationalité du résident pour mieux visualiser.

```
SELECT R.id, R.nom, R.prenom, SUM(ALL montantTotal) as total, R.nationalite
FROM AideTotale, AideSociale
INNER JOIN Resident R on AideSociale.idResident = R.id
WHERE AideTotale.id = AideSociale.id
GROUP BY R.id
ORDER BY total DESC;
```

Table 15 – résultats 1..30 (total: 99)

id	nom	prenom	total	nationalite
11	Roussel	Corentin	19718	France
30	Garnier	Lisa	16906	France
57	Roy	Edwige	14795	Saudi Arabia
17	Gautier	Salomé	14306	France
92	Deschamps	Maxence	13655	Northern Mariana Islands
50	Fabre	Mathis	13556	Somalia
65	Germain	Thomas	13359	France
71	Dufour	Lena	13167	France
93	Rolland	Robin	13151	France
25	Millet	Nolan	12914	France
84	Chevalier	Jasmine	12562	Myanmar
13	Gauthier	Gabin	11924	France
33	Henry	Alice	11528	France
8	Guillot	Adam	10538	France
85	Humbert	Florian	9281	France
72	Barre	Sarah	9207	France
41	Henry	Charlotte	9188	France
83	Dubois	Lina	9108	France
9	Renard	Yanis	9047	France
74	Herve	Anna	8956	France
32	Moreau	Julien	8921	France
100	Boyer	Kevin	8818	France
38	Faure	Enzo	8756	France
75	Chevallier	Yasmine	8630	France
53	Benoit	Amine	8569	France
69	Marchal	Killian	8495	El Salvador
1	Caron	Maëlle	8470	France
55	Marty	Corentin	8338	France
4	Lemoine	Maïlé	8029	France
44	Leveque	Maxime	7997	Togo

3.4.2 Combien on donne au non-français en aides sociales ?

Similairement à la requête précédante, on utilise la vue AideTotale

- Jointure interne de AideTotale et AideSociale sur le *id*
- Jointure interne avec le résident sur *idResident*
- La somme des montants totaux **où** résident est non français.

$$\gamma_{\text{somme(montantTotal)}}(\sigma_{\text{nationalite}\neq\text{"France"}}(\text{Resident} \underset{\text{R.id=idResident}}{\bowtie} (\text{AideSociale} \underset{\text{id}}{\bowtie} A_T)))$$

```
SELECT SUM(ALL montantTotal) AS total
FROM AideTotale, AideSociale
INNER JOIN Resident R on AideSociale.idResident = R.id
WHERE AideTotale.id = AideSociale.id
AND R.nationalite <> "France"
```

Table 16 – résultats 1..1 (total: 1)

x
137893

Il est aussi intéressant de regrouper les montants par nationalité (pour les montants non-nuls).

$$\sigma_{\text{total}\neq 0}(_{\text{nationalite}}\gamma_{\text{somme(montantTotal)}}(\sigma_{\text{nationalite}\neq\text{"France"}}(\text{Resident} \underset{\text{R.id=idResident}}{\bowtie} (\text{AideSociale} \underset{\text{id}}{\bowtie} A_T))))$$

```
SELECT R.nationalite, SUM(ALL montantTotal) AS total
FROM AideTotale, AideSociale
INNER JOIN Resident R on AideSociale.idResident = R.id
WHERE AideTotale.id = AideSociale.id
GROUP BY R.nationalite
HAVING total > 0;
```

Table 17 – résultats 1..16 (total: 16)

nationalite	total
Åland Islands	4467
China	7869
El Salvador	14358
France	456611
Malaysia	11434
Myanmar	12562
Netherlands	7236
Northern Mariana Islands	19770
Saint Helena, Ascension and Tristan da Cunha	4663
Saint Vincent and The Grenadines	5929
Saudi Arabia	14795
Slovenia	4882
Somalia	13556
Togo	7997
Uganda	2307
Venezuela	6068

4 Conclusion

Ce projet nous a permis d'imaginer un service qui pourrait être utile à tous et qui pourrait faciliter la vie des résidents. Un tel système fonctionnel représenterai en effet un gain de temps pour les différents parties. La difficulté fût justement d'imaginer les besoins que pourraient avoir les différents utilisateurs de cette application, afin que celle-ci soit la plus efficace et adaptée possible.

Ce projet nous a également permis (pour la moitié d'entre nous) d'avoir une première approche de la gestion de bases de données et de nous donner une première idée des grandes possibilités qu'apporte leurs utilisation.

Enfin ce projet, nous a de nouveau permis d'améliorer nos capacités de communications et de gestion de projet afin de mener à bien un projet en équipe et non de manière individuel.