

PROJET 3 - DOCUMENT TECHNIQUE

L'ANALYSE DES DONNEES

LE TYPE DE DONNEES

Lors de la récupération d'une base de données, il est nécessaire d'identifier la nature des données, élément essentiel pour créer une base de données saine sur un logiciel de SGBD.

Pour ce faire, pour chaque colonne il faut voir s'il s'agit de nombres entiers, de nombres décimaux, de textes, de dates etc.

Pour l'exemple ci-contre, les données sont de la nature suivante :

- Contrat_ID : nombre entier
- No_voie : nombre entier
- B_T_Q : texte
- Type de voie : texte
- Voie : texte

	A	B	C	D	E
	Contrat_ID	No_voie	B_T_Q	Type_de_voi	Voie
4	100724	711		RUE	DES MORAIN
5	100725	120	B	CHE	DU MARTINE
5	100731	242		RUE	DES VERTES C
7	100734	183		RUE	DU VIEUX BO
8	100747	346		RUE	DE PRE BAILL'
9	100750	348		RUE	DE PRE BAILL'
10	100760	67		RTE	DE GENEVE
11	100761	333		CHE	DES LONGES

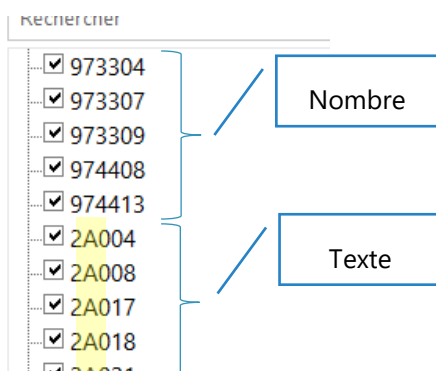
LA DETECTION DES PIEGES

LES FORMATS

Parfois il y a des pièges qui induisent en erreur.

Dans un premier coup d'œil on peut détecter un format qui semble uniforme mais il faut faire attention à l'ensemble des données.

Le cas fut rencontré au niveau de la colonne « Code_dep_code_commune » en survolant la colonne il est facile de croire que les données sont des nombres, mais en utilisant le filtre sur Excel, nous pouvons voir qu'à la fin de la longue liste filtrée, nous trouvons la présence de la lettre A.



Les nombres peuvent facilement être traduits en texte, en revanche, ces valeurs comportant des chiffres et des lettres ne peuvent pas être convertis en nombre. Il faut donc considérer la colonne comme une valeur texte.

Une nuance existe cependant : si sur un fichier ouvert sur Excel des nombres sont en format texte, par défaut alignés à gauche, et s'il n'y a aucune lettre de l'alphabet caché, ils peuvent être convertis en format nombre.

LES VALEURS

Certaines valeurs peuvent également être erronées ou corrompues.

Par exemple, dans la colonne « commune » nous avons des villes dont la commune est « 0 » alors qu'il y a un code postal

	Code_postal	Commune
9	50500	0
2	49120	0

En l'espèce « 50500 » comporte plusieurs villes (Baupte, Auxis, Auvers, etc.) et « 49120 » comporte également plusieurs villes (Chemillé en Anjou, Cossé d'Anjou, La Chapelle Rousselin, etc.). Il semblerait que le code « 0 » s'applique pour les villes ayant le même code postal.

Sachant que la table « Région » ne comporte pas de ville « 0 », il faudra par précaution utiliser une jointure entre ces deux tables et ne pas se fier à la colonne « commune » seule.

LE ROLE DES DONNEES

Quand on analyse un fichier destiné à une base de données, il est toujours utile de comprendre le rôle des données, à quoi elles servent, à quoi elles correspondent, comment elles réagissent entre elles.

No_voie	B_T_Q	Type_de_voi	Voie	Code_dep_cc	Code_postal	Commune
4	C	RUE	NICEPHORE N	38151	38130	ECHIROLLES

En l'espèce, nous voyons bien que dans la table « Contrat » les colonnes qui se suivent, en dehors de la colonne « code_dep_code_commune » correspondent à tous les éléments composants une adresse postale (numéro de rue, code de répétition, type de voie, nom de la voie, code postal et commune)

Code_dep_cc	reg_code	reg_nom	aca_nom	dep_nom	com_nom_maj_court	dep_code	dep_nom_num
1001	84	Auvergne-Rh	Lyon	Ain	LABERGEMENT CLEMENCIAT	1	Ain (01)
1002	84	Auvergne-Rh	Lyon	Ain	LABERGEMENT DE VAREY	1	Ain (01)

Pour la table « Région », nous voyons bien qu'il s'agit de données relatives aux communes, avec leur département, région, et académie.

L'ENCODAGE DES DONNEES

Avant d'importer les données au sein du logiciel de SGBD, il est essentiel de connaître l'encodage des données analysées. Pour ce faire, j'ai utilisé Excel.

J'ai ouvert un nouveau classeur, puis dans le menu « Données », j'ai choisi « A partir d'un fichier texte/CSV »

J'ai sélectionné les deux fichiers, un par un, et en haut de la visualisation, l'encodage est spécifié.

La base de données « Contrat » est en format WIN1252 :

bdd_contrat.csv							
Origine du fichier		Délimiteur		Détection du type de données			
1252: Europe de l'Ouest (Windows)		Point-virgule		Selon les 200 premières lignes			
Contrat_ID	No_voie	B_T_Q	Type_de_voie	Voie	Code_dep_code_commune	Code_postal	C
100773	151		RTE	DE BELLEVILLE	1258	1090	MOI
100611	79		CRS	DE VERDUN	1283	1100	OYO
100645	10		RUE	AMPERE	1283	1100	OYO

La base de données « Région » est au format UTF-8 :

bdd_region.csv							
Origine du fichier		Délimiteur		Détection du type de données			
65001: Unicode (UTF-8)		Point-virgule		Selon les 200 premières lignes			
Code_dep_code_commune	reg_code	reg_nom	aca_nom	dep_nom	com_nom_maj_court	dep_coc	
1001	84	Auvergne-Rhône-Alpes	Lyon	Ain	L ABERGEMENT CLEMENCIAT		
1002	84	Auvergne-Rhône-Alpes	Lyon	Ain	L ABERGEMENT DE VAREY		
1003	84	Auvergne-Rhône-Alpes	Lyon	Ain	AMAREINS		

LA CREATION DU DICTIONNAIRE

Une fois toutes ces informations collectées, il faut ensuite compléter le dictionnaire, qui sert de base de connaissances des différentes données.

On doit y indiquer le nom des colonnes, le format des données, la taille si nécessaire, la clé là où elle existe, et une description sommaire.

	Nom des colonnes	Type de donnée	Taille	Clé	Description
CONTRAT.CSV	Contrat_ID	INTEGER		Clé primaire	Id unique pour les contrats
	No_voie	CHAR	4		Numéro dans la voie pour l'adresse du logement assuré
	B_T_Q	CHAR	1		Indicateur éventuel de répétition pour l'adresse du logement assuré sur un caractère
	Type_de_voie	CHAR	4		Type de voie pour l'adresse du logement assuré: rue, av (Avenue), rte (Route), ...
	Voie	VARCHAR			Libellé de la voie pour l'adresse du logement assuré
	Code_dep_code_commune	CHAR	6	Clé étrangère	Concaténation du code département et code commune pour avoir une clé unique
	Code_postal	INTEGER			Code postal pour l'adresse du logement assuré
	Commune	VARCHAR			Libellé de la commune de l'adresse du logement
	Code_département	INTEGER			Numéro du département
	Surface	INTEGER			Surface du bien immobilier
	Type_local	CHAR	11		Typologie du local (appartement, maison)
	Occupation	CHAR	12		Qualité de l'occupant (locataire, propriétaire)
	Type_contrat	CHAR	20		Le type de contrat (Mise en location, Residence principale ou secondaire)
REGION.CSV	Formule	CHAR	9		Le type de formule (Intégrale ou classique)
	Valeur_declaree_biens	CHAR	12		La fourchette de valeur du bien
	Prix_cotisation_mensuel	INTEGER			Montant de la cotisation mensuelle
	Code_dep_code_commune	CHAR	6	Clé primaire	Concaténation du code département et code commune pour avoir une clé unique
	reg_code	INTEGER			Code de la région
	reg_nom	CHAR	26		Nom de la région
	aca_nom	CHAR	24		Nom de l'académie
	dep_nom	CHAR	43		Nom du département
	com_nom_maj_court	VARCHAR			Nom de la commune
	dep_code	CHAR	3		Numéro du département
	dep_nom_num	CHAR	49		Concaténation du nom du département et du numéro du département entre parenthèses

LE NOM DES TYPES DE DONNEES

En fonction du logiciel de SGBD utilisé, le nom du type de données peut varier.

Données	SQLite	PostgreSQL	MySQL	SQLServer
Nombre entier	INTEGER	INTEGER	INTEGER	INT
Nombre décimal	REAL	REAL	REAL	FLOAT
Texte longueur fixe	TEXT	CHAR(n)	CHAR(n)	CHAR(n)
Texte longueur variable	TEXT	VARCHAR	VARCHAR	VARCHAR
Booléen		BOOLEAN		
Date	NUMERIC	DATE	DATE	DATE
Heure	NUMERIC	TIME	TIME	TIME
Date et Heure		TIMESTAMP	DATETIME	DATETIME

LES CONTRAINTES

Après avoir indiqué les colonnes et le type de données, il faut préciser les contraintes.

LES LONGUEURS DE VALEURS

Pour certaines données, la longueur est fixe et non variable.

C'est notamment le cas de la colonne « B_T_Q » qui précise le code de répétition du numéro de rue. Dans la liste il n'y a que du B pour bis, T pour ter et Q pour quarter.

S'agissant d'une lettre seule, il faut donc indiquer en format « CHAR » car longueur fixe et en contrainte 1 car il n'y a qu'un seul caractère.

Toutes les autres données, que ce soit le nom de rue, la ville, etc. sont des données à longueur variable, il est préférable d'indiquer « VARCHAR » qui confère une longueur illimitée aux données

Certaines données n'ont pas besoin d'un nombre illimité de caractère, c'est le cas des départements, des régions qui restent fixe pendant un grand nombre d'années. Il est donc possible d'indiquer CHAR(nombre maximum de caractère de la colonne)

LES VALEURS VIDES

Les données autorisant ou non des valeurs vides ne sont pas à préciser dans le dictionnaire fourni mais sont à connaître pour la création de l'architecture.

Certaines données peuvent être vides, notamment les codes de répétition, car toute adresse n'a pas forcément de bis, ter ou quarter. Même le numéro de rue peut être vide car certaines adresses sont composées que d'une rue ou d'une place.

Pour ces données, dont les valeurs sont facultatives, il est nécessaire de préciser qu'elles peuvent être vides, pour les autres il faudra les interdire.

L'information sera ensuite visible dans l'architecture par l'info « NOT NULL », c'est-à-dire qui n'accepte pas de données vides. Sans précision, cela veut donc dire que les valeurs vides sont acceptées.

LES CLES

Il est ensuite indispensable de préciser les clés pour chaque table afin que les éventuelles jointures fonctionnent.

Chaque table doit contenir une ou plusieurs données permettant d'identifier un seul enregistrement. Il peut s'agir d'une clé naturelle, en combinant plusieurs données ou d'une clé artificielle, généralement un ID.

En l'espèce, la clé primaire de la table « Contrat » est la colonne « Contrat_ID » et la clé primaire de la table « Region » est l'identifiant unique « Code_dep_code_commune » qui sont tous deux le seul moyen d'identifier un enregistrement unique.

La clé étrangère est potentiellement optionnelle, elle n'existe que si une jointure entre deux tables peut être établie.

En l'espèce, le point commun entre la table « Contrat » et la table « Région » est la clé « Code_dep_code_commune ». Etant également présente dans la table « Contrat » elle est donc la clé étrangère de la table « Contrat »

LA DESCRIPTION

La description est toujours utile en cas de remplacement de personnel ou quand on revient sur une base de données ancienne dont on ne se souvient plus précisément de sa construction.

Elle sert à expliquer à quoi sert l'information de la colonne. Cela peut être très utile s'il s'agit d'un calcul, d'une concaténation de données, ou quand le nom de la colonne n'est pas suffisamment explicite.

LA CREATION DE L'ARCHITECTURE

Suite à la création du dictionnaire, il faut ensuite retranscrire toutes les informations dans une architecture SQL afin de pouvoir générer un code SQL.

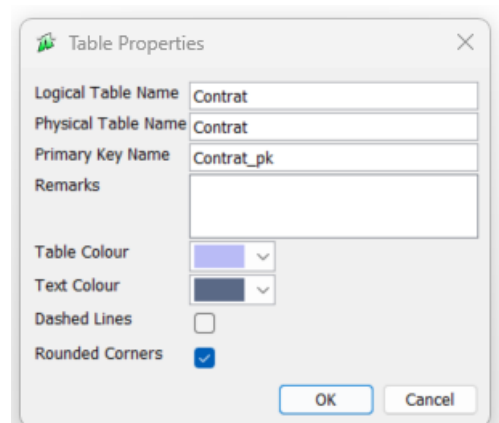
Le logiciel utilisé est SQL Power Architect.

LA CREATION DES TABLES

Dans un premier temps il faut créer la table, en faisant un clic droit au milieu du canevas puis « New Table... ».

Ensuite dessiner un rectangle avec la souris et en relâchant la souris, une fenêtre s'ouvre. Il suffit d'indiquer le nom de la table (par exemple « Contrat », et indiquer la même information sur tous les deux autres champs en dessous.

Il est également possible de personnaliser sa table avec des couleurs de table, de texte, les contours en tirets, et l'arrondi des angles, etc.



L'IMPORT DE CHAQUE COLONNE

Une fois les tables créées il faut ensuite ajouter les colonnes.

En sélectionnant la table, il faut cliquer sur le logo situé sur la droite, dont l'infobulle indique « Insert Column » ou en cliquant sur le raccourci clavier « C »

Une nouvelle fenêtre s'ouvre.

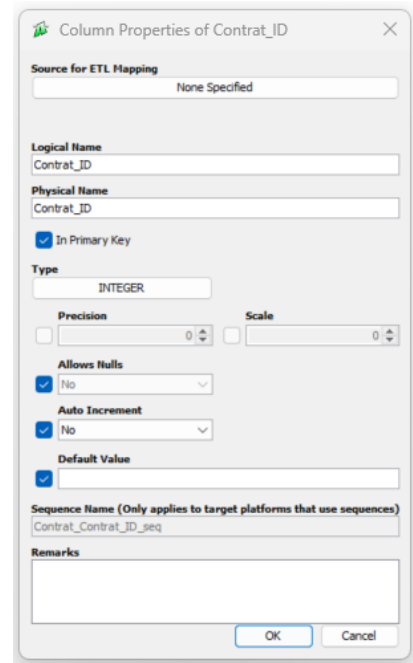
Il faut inscrire le nom de la colonne (à indiquer dans « logical name » et « physical name »)

S'il s'agit d'une clé primaire, il faut cocher la case « In Primary Key »

Préciser le type de données, comme indiqué dans le dictionnaire

Si des données vides sont autorisées, il faut indiquer « Yes » dans « Allows Nulls » sinon, indiquer « No »

Les autres informations peuvent rester par défaut



LA JOINTURE

Comme indiqué précédemment les clés sont indiquées afin de permettre des jointures.

Dans le dictionnaire nous avons indiqué que « Contrat_ID » est la clé primaire de la table « Contrat » et « Code_dep_code_commune » est la clé primaire de la table région. De fait, sur l'architecture, ces deux données doivent être en haut de chaque table avec l'acronyme « PK » entre crochet.

Contrat	Region
Contrat_ID: INTEGER NOT NULL [PK]	Code_dep_code_commune: CHAR(6) NOT NULL [PK]

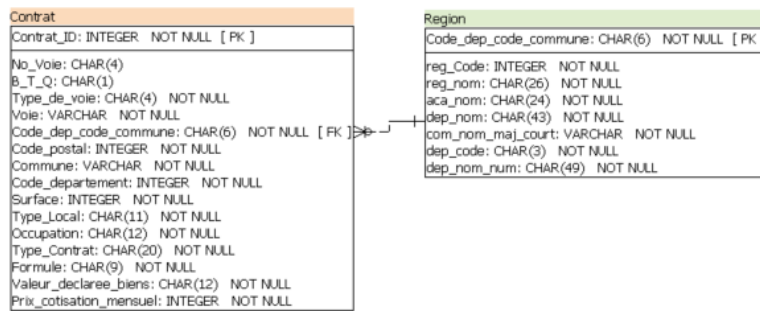
Pour faire la jointure entre les deux tables, il faut sélectionner le logo sur le bandeau de droite dont l'info-bulle est « New identifying Relationship », cliquer ensuite sur la clé primaire de la table « Region » : « Code_dep_code_commune » et finir en cliquant sur la clé étrangère de la table « Contrat » : « code_dep_code_commune ».

Sur la table contrat le « code_dep_code_commune » remonte tout en haut avec, entre crochet, le code « PFK » (primary foreign key)

Contrat	Region
Contrat_ID: INTEGER NOT NULL [PK] Code_dep_code_commune: CHAR(6) NOT NULL [PFK]	Code_dep_code_commune: CHAR(6) NOT NULL [PK] Contrat_ID: INTEGER NOT NULL [FK]

Afin de conserver l'ordre des colonnes, il faut cliquer et faire glisser « Code_dep_code_commune » sous la colonne « Voie ». Le PFK est remplacé par un FK (foreign key).

La jointure faite, nous pouvons considérer l'architecture comme étant terminée.



L'EXPORT SQL

Une fois les tables, les colonnes et les jointures créées, l'architecture est terminée, il ne reste qu'à générer le code SQL.

Pour ce faire, il faut cliquer sur le logo SQL avec la flèche verte présent sur le bandeau supérieur.

Une fenêtre s'ouvre, sur laquelle il faut préciser le langage utilisé, en l'espèce PostgreSQL, puis valider en cliquant sur OK. Le code se génère automatiquement et il est possible de le copier en cliquant sur « Copy »

```

    CREATE TABLE public.Region (
      Code_dep_code_commune CHAR(6) NOT NULL,
      reg_Code INTEGER NOT NULL,
      reg_nom CHAR(26) NOT NULL,
      aca_nom CHAR(24) NOT NULL,
      dep_nom CHAR(43) NOT NULL,
      com_nom_maj_court VARCHAR NOT NULL,
      dep_code CHAR(3) NOT NULL,
      dep_nom_num CHAR(49) NOT NULL,
      CONSTRAINT region_pk PRIMARY KEY
    (Code_dep_code_commune)
    );

    CREATE TABLE public.Contrat (
      Contrat_ID INTEGER NOT NULL,
      No_Voie CHAR(4),
      B_T_Q CHAR(1),
      Type_de_voie CHAR(4) NOT NULL,
      Voie VARCHAR NOT NULL,
      Code_dep_code_commune CHAR(6) NOT NULL,
      Code_postal INTEGER NOT NULL,
      Commune VARCHAR NOT NULL,
      Code_departement INTEGER NOT NULL,
      Surface INTEGER NOT NULL,
      Type_Local CHAR(11) NOT NULL,
      Occupation CHAR(12) NOT NULL,
      Type_Contrat CHAR(20) NOT NULL,
      Formule CHAR(9) NOT NULL,
      Valeur_declaree_biens CHAR(12) NOT NULL,
      Prix_cotisation_mensuel INTEGER NOT NULL,
      CONSTRAINT contrat_pk PRIMARY KEY (Contrat_ID)
    );

    ALTER TABLE public.Contrat ADD CONSTRAINT region_contrat_fk
    FOREIGN KEY (Code_dep_code_commune)
    REFERENCES public.Region (Code_dep_code_commune)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION
    NOT DEFERRABLE;
  
```

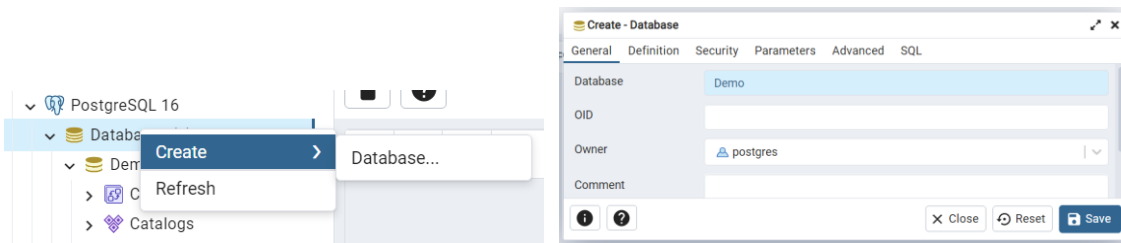
LA CREATION DE LA BASE DE DONNEES

Suite à la génération du Code SQL il est possible d'aller sur le logiciel de SGBD. Ici, il s'agira de PostgreSQL.

CREATION DE LA BASE DE DONNEE

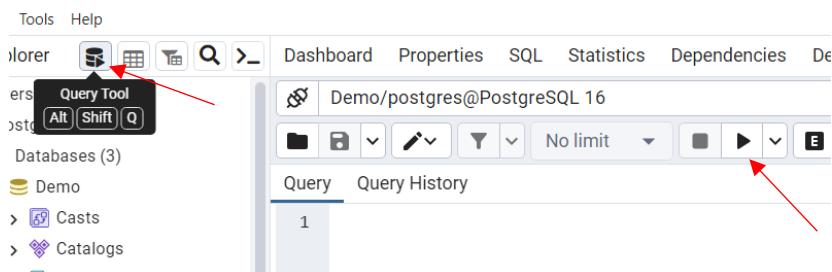
Avant toute chose, il faut créer sa base de données.

En faisant un clic droit sur « Database » puis « Create » puis « Database ». Il ne reste qu'à donner un nom à la base de données



L'IMPORT DES TABLES

Une fois dans la base de données, aller dans l'outil de requête



Coller la requête précédemment générée, sélectionner toute la requête avec le raccourci « Ctrl +A » et exécuter avec le bouton « Execute/Refresh »

Les tables, une fois créées, sont visibles au niveau de l'arborescence suivante : Database / nom_de_la_bdd / Schemas / public / Tables. Il faut parfois faire un clic droit sur Table et cliquer sur Refresh pour faire apparaître les tables

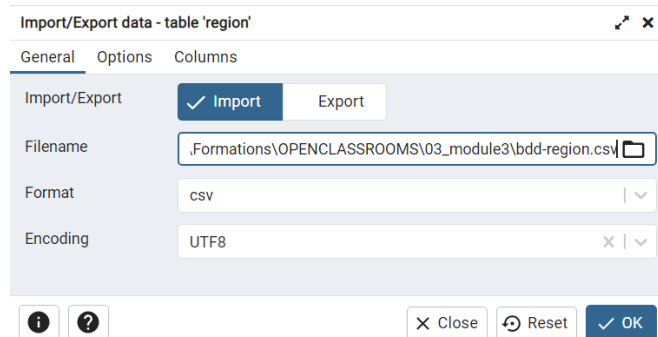


L'IMPORT DES DONNEES

Les tables étant créées, elles sont donc pour le moment vides. Il faut charger les données.

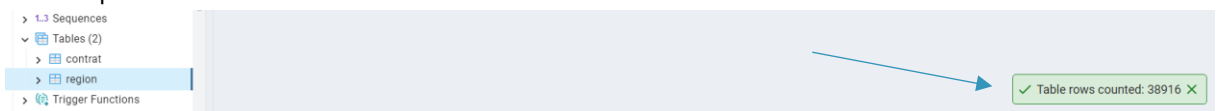
Pour ce faire il faut faire un clic droit sur la table et sélectionner « Import/Export Data... ». Une fenêtre s'ouvre.

Il faut charger le fichier CSV, indiquer le format (ici, csv) et préciser l'encodage qui a été déterminé au moment de l'analyse des données. Ici, comme le fichier Région est en « UTF-8 », je précise donc UTF-8. En chargeant l'autre fichier, il faudra indiquer « WIN1252 »

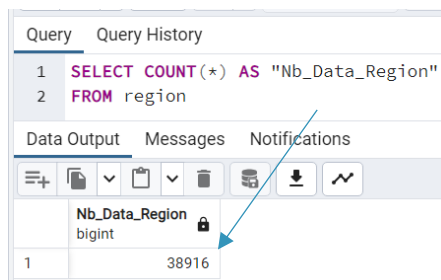


Pour vérifier le bon import des données, il y a deux méthodes :

1. Soit faire un clic droit sur le nom de la table et cliquer sur « Count Rows ». Le résultat apparaît sur une petite fenêtre à droite



2. Soit faire une requête SQL basique : `SELECT COUNT(*) FROM base_de_donnees`



LES REQUETES SQL

LES REQUETES SIMPLES

Les requêtes simples sont des requêtes ne nécessitant qu'une source de données, sans calcul particulier.

Elle est construite par un appel d'enregistrements via la commande SELECT, la source des données, via la commande FROM, des filtres éventuels via les commandes WHERE, et éventuellement AND, un éventuel classement via la commande ORDER BY et une limitation d'un nombre de données via la commande LIMIT

Requête	Résultat																																								
SELECT contrat_id, type_contrat, formule FROM contrat WHERE type_local = 'Maison' AND Code_departement = 71	<table><tr><th></th><th>contrat_id [PK] integer</th><th>type_contrat character varying</th><th>formule character varying</th></tr><tr><td>1</td><td>114768</td><td>Residence principale</td><td>Integral</td></tr><tr><td>2</td><td>114782</td><td>Residence principale</td><td>Classique</td></tr><tr><td>3</td><td>114812</td><td>Residence principale</td><td>Integral</td></tr><tr><td>4</td><td>114779</td><td>Residence principale</td><td>Classique</td></tr></table>		contrat_id [PK] integer	type_contrat character varying	formule character varying	1	114768	Residence principale	Integral	2	114782	Residence principale	Classique	3	114812	Residence principale	Integral	4	114779	Residence principale	Classique																				
	contrat_id [PK] integer	type_contrat character varying	formule character varying																																						
1	114768	Residence principale	Integral																																						
2	114782	Residence principale	Classique																																						
3	114812	Residence principale	Integral																																						
4	114779	Residence principale	Classique																																						
SELECT contrat_ID, surface FROM contrat ORDER BY surface DESC LIMIT 5	<table><tr><th></th><th>contrat_id [PK] integer</th><th>surface integer</th></tr><tr><td>1</td><td>104211</td><td>815</td></tr><tr><td>2</td><td>105463</td><td>742</td></tr><tr><td>3</td><td>130878</td><td>595</td></tr><tr><td>4</td><td>100822</td><td>570</td></tr><tr><td>5</td><td>109872</td><td>559</td></tr></table>		contrat_id [PK] integer	surface integer	1	104211	815	2	105463	742	3	130878	595	4	100822	570	5	109872	559																						
	contrat_id [PK] integer	surface integer																																							
1	104211	815																																							
2	105463	742																																							
3	130878	595																																							
4	100822	570																																							
5	109872	559																																							
SELECT DISTINCT reg_nom FROM region	<table><tr><th></th><th>reg_nom character varying</th></tr><tr><td>1</td><td>Hauts-de-France</td></tr><tr><td>2</td><td>La Réunion</td></tr><tr><td>3</td><td>Bretagne</td></tr><tr><td>4</td><td>Mayotte</td></tr><tr><td>5</td><td>Auvergne-Rhône-Alpes</td></tr><tr><td>6</td><td>Ile-de-France</td></tr><tr><td>7</td><td>Grand Est</td></tr><tr><td>8</td><td>Corse</td></tr><tr><td>9</td><td>Collectivités d'outre-mer</td></tr><tr><td>10</td><td>Guyane</td></tr><tr><td>11</td><td>Bourgogne-Franche-Comté</td></tr><tr><td>12</td><td>Provence-Alpes-Côte d'Azur</td></tr><tr><td>13</td><td>Martinique</td></tr><tr><td>14</td><td>Guadeloupe</td></tr><tr><td>15</td><td>Normandie</td></tr><tr><td>16</td><td>Nouvelle-Aquitaine</td></tr><tr><td>17</td><td>Occitanie</td></tr><tr><td>18</td><td>Pays de la Loire</td></tr><tr><td>19</td><td>Centre-Val de Loire</td></tr></table>		reg_nom character varying	1	Hauts-de-France	2	La Réunion	3	Bretagne	4	Mayotte	5	Auvergne-Rhône-Alpes	6	Ile-de-France	7	Grand Est	8	Corse	9	Collectivités d'outre-mer	10	Guyane	11	Bourgogne-Franche-Comté	12	Provence-Alpes-Côte d'Azur	13	Martinique	14	Guadeloupe	15	Normandie	16	Nouvelle-Aquitaine	17	Occitanie	18	Pays de la Loire	19	Centre-Val de Loire
	reg_nom character varying																																								
1	Hauts-de-France																																								
2	La Réunion																																								
3	Bretagne																																								
4	Mayotte																																								
5	Auvergne-Rhône-Alpes																																								
6	Ile-de-France																																								
7	Grand Est																																								
8	Corse																																								
9	Collectivités d'outre-mer																																								
10	Guyane																																								
11	Bourgogne-Franche-Comté																																								
12	Provence-Alpes-Côte d'Azur																																								
13	Martinique																																								
14	Guadeloupe																																								
15	Normandie																																								
16	Nouvelle-Aquitaine																																								
17	Occitanie																																								
18	Pays de la Loire																																								
19	Centre-Val de Loire																																								

LES REQUETES AVEC AGREGATION

Les requêtes avec agrégation sont construites globalement de la même manière que les requêtes simples. Elles demandent néanmoins des calculs telles que des sommes, des moyennes etc.

Elles sont également composées des commandes SELECT, FROM, WHERE mais au niveau de SELECT il y a les calculs (SUM : somme, AVG : moyenne, COUNT : nombre, MIN : minimum, MAX : maximum)

Les résultats sont soit globaux, en une ligne, soit agrégés par des critères via la commande GROUP BY.

Il peut également y avoir un filtre post-calcul, via la commande HAVING.

Requête	Résultat				
SELECT ROUND(AVG(surface),2) AS "Surface_Moyenne" FROM contrat WHERE code_departement = 75	<table> <thead> <tr> <th></th><th>Surface_Moyenne numeric</th></tr> </thead> <tbody> <tr><td>1</td><td>51.77</td></tr> </tbody> </table>		Surface_Moyenne numeric	1	51.77
	Surface_Moyenne numeric				
1	51.77				
SELECT ROUND(AVG(Prix_cotisation_mensuel),2) AS "Cotisation_moyenne" FROM contrat	<table> <thead> <tr> <th></th><th>Cotisation_moyenne numeric</th></tr> </thead> <tbody> <tr><td>1</td><td>19.33</td></tr> </tbody> </table>		Cotisation_moyenne numeric	1	19.33
	Cotisation_moyenne numeric				
1	19.33				

SELECT Valeur_declaree_biens, COUNT(contrat_ID) AS "Nb_Contrats" FROM contrat GROUP BY Valeur_declaree_biens	<table><tr><th></th><th>valeur_declaree_biens character varying</th><th>Nb_Contrats bigint</th></tr><tr><td>1</td><td>50000-100000</td><td>696</td></tr><tr><td>2</td><td>100000+</td><td>104</td></tr><tr><td>3</td><td>25000-50000</td><td>6815</td></tr><tr><td>4</td><td>0-25000</td><td>22720</td></tr></table>		valeur_declaree_biens character varying	Nb_Contrats bigint	1	50000-100000	696	2	100000+	104	3	25000-50000	6815	4	0-25000	22720																		
	valeur_declaree_biens character varying	Nb_Contrats bigint																																
1	50000-100000	696																																
2	100000+	104																																
3	25000-50000	6815																																
4	0-25000	22720																																
SELECT count(contrat_ID) AS "Nb_Contrats" FROM contrat WHERE Type_contrat = 'Residence principale'	<table><tr><th></th><th>Nb_Contrats bigint</th></tr><tr><td>1</td><td>25620</td></tr></table>		Nb_Contrats bigint	1	25620																													
	Nb_Contrats bigint																																	
1	25620																																	
SELECT Code_departement, ROUND(AVG(prix_cotisation_mensuel),2) AS "prix_moyen" FROM contrat GROUP BY Code_departement ORDER BY prix_moyen DESC LIMIT 10	<table><tr><th></th><th>code_departement integer</th><th>prix_moyen numeric</th></tr><tr><td>1</td><td>75</td><td>36.40</td></tr><tr><td>2</td><td>92</td><td>26.27</td></tr><tr><td>3</td><td>94</td><td>19.82</td></tr><tr><td>4</td><td>78</td><td>18.88</td></tr><tr><td>5</td><td>69</td><td>18.46</td></tr><tr><td>6</td><td>1</td><td>18.24</td></tr><tr><td>7</td><td>6</td><td>18.14</td></tr><tr><td>8</td><td>17</td><td>17.32</td></tr><tr><td>9</td><td>74</td><td>17.16</td></tr><tr><td>10</td><td>20</td><td>17.03</td></tr></table>		code_departement integer	prix_moyen numeric	1	75	36.40	2	92	26.27	3	94	19.82	4	78	18.88	5	69	18.46	6	1	18.24	7	6	18.14	8	17	17.32	9	74	17.16	10	20	17.03
	code_departement integer	prix_moyen numeric																																
1	75	36.40																																
2	92	26.27																																
3	94	19.82																																
4	78	18.88																																
5	69	18.46																																
6	1	18.24																																
7	6	18.14																																
8	17	17.32																																
9	74	17.16																																
10	20	17.03																																

LES REQUETES AVEC JOINTURES

Les requêtes avec jointures fonctionnent de la même manière que les requêtes simples et les requêtes avec agrégations. La seule différence réside sur le fait que ces requêtes récupèrent les informations sur plusieurs sources de données, liées par des clés étrangères et primaires.

Elles se remarques par la présence de commandes de jointures telles que INNER JOIN, LEFT JOIN, RIGHT JOIN, FULL JOIN.

Requête	Résultat															
<pre>SELECT contrat_id, surface FROM contrat ct LEFT JOIN region rg ON ct.Code_dep_code_commune = rg.Code_dep_code_commune WHERE com_nom_maj_court = 'CAEN'</pre>	<table><tr><th></th><th>contrat_id [PK] integer</th><th>surface integer</th></tr><tr><td>1</td><td>103791</td><td>35</td></tr><tr><td>2</td><td>103792</td><td>99</td></tr><tr><td>3</td><td>103793</td><td>40</td></tr><tr><td>4</td><td>103794</td><td>20</td></tr></table>		contrat_id [PK] integer	surface integer	1	103791	35	2	103792	99	3	103793	40	4	103794	20
	contrat_id [PK] integer	surface integer														
1	103791	35														
2	103792	99														
3	103793	40														
4	103794	20														

SELECT COUNT(contrat_ID) AS "Nb_Contrats"

FROM contrat ct

LEFT JOIN region rg ON
ct.Code_dep_code_commune =
rg.Code_dep_code_commune

WHERE formule = 'Integral'

AND reg_nom = 'Pays de la Loire'

	Nb_Contrats bigint
1	561

SELECT com_nom_maj_court AS
"Nom_commune", COUNT(contrat_ID) AS
"Nb_Contrats"

FROM contrat ct

LEFT JOIN region rg ON
ct.Code_dep_code_commune =
rg.Code_dep_code_commune

GROUP BY com_nom_maj_court

HAVING COUNT(contrat_ID) >= 150

ORDER BY "Nb_Contrats" DESC

	Nom_commune character varying	Nb_Contrats bigint
1	PARIS 18	515
2	PARIS 17	468
3	PARIS 15	407
4	PARIS 16	394
5	NICE	387
6	PARIS 11	381
7	BORDEAUX	302
8	PARIS 20	302
9	NANTES	291
10	PARIS 19	266
11	PARIS 10	263
12	PARIS 12	252
13	PARIS 14	222
14	GRENOBLE	220
15	PARIS 9	204
16	TOULOUSE	187
17	TOULON	170
18	COURBEVOIE	163
19	LILLE	161
20	PARIS 3	159

SELECT reg_nom, COUNT(contrat_ID) AS
"Nb_Contrats"

FROM contrat ct

LEFT JOIN region rg ON
ct.Code_dep_code_commune =
rg.Code_dep_code_commune

GROUP BY reg_nom

ORDER BY "Nb_Contrats" DESC

	reg_nom character varying	Nb_Contrats bigint
1	Ile-de-France	13474
2	Provence-Alpes-Côte d'Azur	3287
3	Auvergne-Rhône-Alpes	2972
4	Nouvelle-Aquitaine	2097
5	Occitanie	1837
6	Hauts-de-France	1327
7	Pays de la Loire	1138
8	Bretagne	945
9	Normandie	898
10	Grand Est	806
11	Centre-Val de Loire	804
12	Bourgogne-Franche-Comté	402
13	Corse	247
14	Martinique	60
15	Guyane	37
16	La Réunion	4