Université de Montréal

SCI6005 Information numérique et informatique documentaire (A2021)

Christine Dufour, EBSI, UdeM

A2021 2021-11-23

Cours 10 - Structuration de l'information dans une base de données (1 de 2)

Christine Dufour, EBSI, UdeM

Table des matières

Cours 10 - Structuration de l'information dans une base de données (1 de 2)

1. + Au programme aujourd'hui

- Principes de base des systèmes de gestion de bases de données
- Principaux modèles de données en contexte documentaire
    - Modèle textuel
    - Modèle relationnel (avec des capsules vidéos pour illustrer la création d'une structure de données dans *Base*)
    - Familles NoSQL

2. + Alignement pédagogique

Objectifs visés, matériel du cours et évaluation : Examen final

Lien entre les objectifs, les compétences à développer et le matériel du cours 10

Objectif général : Concevoir et implanter des systèmes et services d'information numérique		
Objectifs spécifiques	**Compétences à développer**	**Matériel associé**
Appliquer une approche systématique pour la mise sur pied de systèmes et de services d'information numérique	Modéliser une base de données	Section *SGBD : Modèle textuel > Modélisation d'une base de données textuelle* Section *SGBD : Modèle relationnel > Modélisation d'une base de données relationnelle* *TP Structuration dans une base de données*
Exploiter la structuration de l'information numérique dans différents contextes	Décrire la structure de données d'une base de données	Section *SGBD : Modèle textuel > Structure des données du modèle textuel* Section *SGBD : Modèle relationnel > Structure des données du modèle relationnel* *TP Structuration dans une base de données*

	Décrire les opérations dans une base de données	Section *SGBD : Modèle textuel > Opérations possibles pour le modèle textuel* Section *SGBD : Modèle relationnel > Opérations possibles pour le modèle relationnel*
	Définir une structure de données dans un SGBD	Section *SGBD : Modèle relationnel > Implantation de la modélisation dans un SGBD (Base)*

Objectifs visés, matière du cours et activités associées

Lien entre les objectifs, la matière du cours 10 et les activités associées

Objectif général : Concevoir et implanter des systèmes et services d'information numérique		
Objectifs spécifiques	**Matière**	**Activités**
Exploiter la structuration de l'information numérique dans différents contextes	Modélisation d'une base de données	*TP Structuration dans une base de données*
	Structure de données dans une base de données	*TP Structuration dans une base de données*

3. Introduction

Les professionnels et professionnelles de l'information font grand usage des **bases de données**, notamment pour faire de la **recherche d'information**. En sus de ce chapeau d'utilisateurs et utilisatrices de bases de données, il est utile de s'attarder à leur **conception** pour bien comprendre comment l'information y est **structurée** et ce que cette structure apporte pour la **réutilisation de l'information**. Cette compréhension est utile tant pour intervenir dans la *conception* des bases de données que dans leur *utilisation*.

Dans ce premier cours sur la structuration de l'information dans les bases de données, nous nous attarderons principalement à comprendre ce qu'est une **base de données** et un **système de gestion de bases de données** ainsi qu'à explorer des **modèles de données** présents en contexte documentaire (modèle textuel, modèle relationnel et familles NoSQL).

Remarque

La matière présentée dans ce cours représente une *introduction aux bases de données*. Il est en effet impossible en seulement deux séances de cours de couvrir en profondeur la richesse notamment du modèle relationnel. Pour celles et ceux qui auraient la piqûre, le cours SCI6306 Bases de données documentaires[1] est pour vous! En attendant, quelques lectures complémentaires ont été précisées dans la section *Ressources* des notes de cours.

[1] https://admission.umontreal.ca/cours-et-horaires/cours/sci-6306/

4. Principes généraux des SGBD

4.1. Définitions

Base de données *Définition*

Dans son sens large, une **base de données** est un ensemble de **données persistantes**, c'est-à-dire des données que l'on veut *conserver pour une certaine durée*. Ces données sont **organisées** par **collections d'items** en fonction de **similitudes de structure** ou selon d'autres critères de regroupement, données qui peuvent être **reliées** entre elles. Une base de données vise à **représenter une certaine réalité** pour pouvoir agir sur elle.

Système de base de données *Définition*

Un *système de base de données* est un système informatique servant à **maintenir des informations** et à les **rendre disponibles à la demande**. Il possède quatre composantes :

1. Les données
2. Le matériel
3. Le logiciel
4. Les utilisateurs

Système de gestion de bases de données (SGBD) *Définition*

Un *système de gestion de bases de données* (SGBD) est la **composante logicielle** la plus importante d'un *système de base de données*. Il prend en charge les **requêtes** pour **accéder à la base de données**, pour y faire des **ajouts**, des **suppressions**, des **mises à jour**, etc.

4.2. Objectifs des SGBD

Un **SGBD** vise à offrir des mécanismes pour :

- S'assurer de la **cohérence** des données
- Assurer la **sécurité** et le **partage** des données
- Assurer l'**indépendance** des données (par rapport au matériel et au logiciel)
- Permettre d'exploiter les **liens** entre les données
- Obtenir une bonne **performance** (vitesse et gestion d'espace)

4.3. Classes d'utilisateurs

On retrouve **différents types d'utilisateurs des SGBD** : les *utilisateurs finaux*, les *responsables du contenu*, les *administrateurs de bases de données* et les *programmeurs d'application* :

Classes d'utilisateurs des SGBD

Utilisateurs finaux	Interagissent avec la base de données principalement pour l'**interroger** et **imprimer des rapports**. Tout dépendant leur niveau de connaissance du langage d'interrogation, ils peuvent soit utiliser des requêtes prédéfinies ou en formuler eux-mêmes.
Responsables du contenu	Interagissent avec la base de données pour **gérer certains contenus** (ajout d'information, mise à jour, suppression).
Administrateurs de bases de données	Assurent la **gestion technique** nécessaire pour **implémenter les SGBD** : définition de la structure conceptuelle et physique, définitions des règles de sécurité, interaction avec les utilisateurs finaux, supervision des performances, etc.
Programmeurs d'application	Programment des **applications** pour **interagir** avec la base de données (par exemple des pages ASP ou PHP pour mettre une base de données en ligne).

Les **professionnels de l'information** interviennent principalement comme **utilisateurs finaux**, **responsables du contenu** ou **administrateurs de bases de données**. Il leur est aussi possible d'agir comme *programmeurs d'application* pour un système à petite échelle.

5. Principaux modèles de SGBD en contexte documentaire

5.1. Introduction

Tout SGBD est basé sur un **modèle de données** constitué de :

- Une façon de **structurer** les données
- Des **opérations** pour agir sur les données

Au fil de l'évolution de la technologie et des besoins pour des bases de données, **différents modèles de données ont vu le jour**. Les modèles présentés dans le cadre du cours sont quelques-uns des principaux modèles que l'on retrouve en contexte documentaire, mais d'autres existent. Nous verrons plus spécifiquement :

- Le *modèle textuel* (fichier plat)
- Le *modèle relationnel* (dans un détail un peu plus grand, comme c'est l'environnement retenu pour le TP)
- Les *familles NoSQL* (pour information)

5.2. Modèle textuel

Les premières utilisations des ordinateurs étaient pour des bases de données simples comme, par exemple, des données de recensement. Le premier modèle de données, dans ces contextes, a été le **modèle de fichier plat**. L'ajout de différentes caractéristiques et fonctionnalités spécifiquement pour les *données textuelle*s l'a fait évolué dans certains cas vers un **modèle textuel**. Inmagic DB/TextWorks[1] et CDS/ISIS[2] sont deux exemples de bases de données basées sur le modèle textuel. Plusieurs *bases de données commerciales accessibles en ligne* sur des serveurs sont gérées par des SGBD textuels.

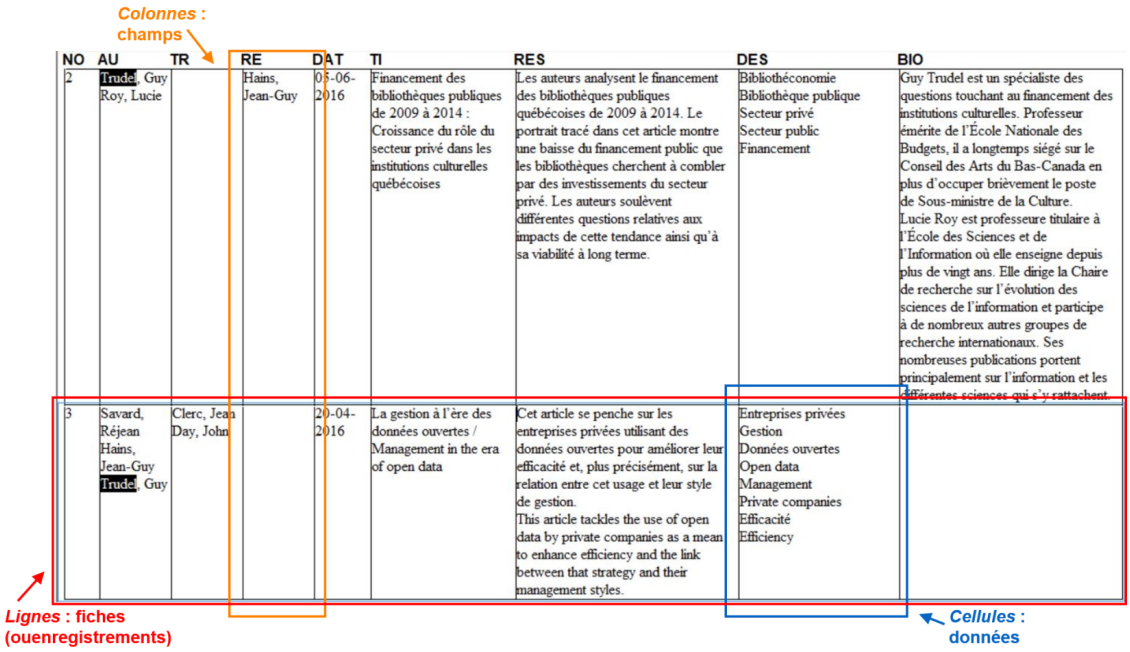
Contexte des exemples ***Remarque***

Afin d'illustrer nos propos, nous utiliserons la **base de données fictive REPSCIE** développée avec le *SGBD textuel DB/TextWorks*. Une version de démonstration de ce SGBD est disponible sur les postes des laboratoires de l'EBSI. La base de données RESPCIE est utilisée par le comité éditorial de la *Revue prestigieuse des sciences de l'information essentielle* pour effectuer le suivi des articles acceptés pour publication de la revue. On y retrouve de l'information sur les auteurs des articles (champs AU pour les noms et BIO pour une courte biographie), sur les réviseurs des articles (RE), sur les traducteurs des articles (champ TR) et sur les articles eux-mêmes (champs NO pour les identifier, TI pour le titre, RES pour un résumé, DAT pour la date de publication et DES pour des descripteurs).

[1] https://lucidea.com/inmagic-dbtextworks/
[2] http://www.unesco.org/new/en/communication-and-information/information-society/open-source-and-low-cost-technologies/information-processing-tools/cdsisis-database-software/

a) Structure des données du modèle textuel

Dans le modèle textuel, la structuration des données se fait sur la base d'**une seule table de données** où chaque **ligne** représente un *enregistrement* (fiche) et chaque **colonne**, un *champ*. Un champ représente une caractéristique de l'enregistrement. Les enregistrements dans cette table ne sont pas reliés entre eux.



NO	AU	TR	RE	DAT	TI	RES	DES	BIO
2	Trudel, Guy Roy, Lucie		Hains, Jean-Guy	05-06-2016	Financement des bibliothèques publiques de 2009 à 2014 : Croissance du rôle du secteur privé dans les institutions culturelles québécoises	Les auteurs analysent le financement des bibliothèques publiques québécoises de 2009 à 2014. Le portrait tracé dans cet article montre une baisse du financement public que les bibliothèques cherchent à combler par des investissements du secteur privé. Les auteurs soulèvent différentes questions relatives aux impacts de cette tendance ainsi qu'à sa viabilité à long terme.	Bibliothéconomie Bibliothèque publique Secteur privé Secteur public Financement	Guy Trudel est un spécialiste des questions touchant au financement des institutions culturelles. Professeur émérite de l'École Nationale des Budgets, il a longtemps siégé sur le Conseil des Arts du Bas-Canada en plus d'occuper brièvement le poste de Sous-ministre de la Culture. Lucie Roy est professeure titulaire à l'École des Sciences et de l'Information où elle enseigne depuis plus de vingt ans. Elle dirige la Chaire de recherche sur l'évolution des sciences de l'information et participe à de nombreux autres groupes de recherche internationaux. Ses nombreuses publications portent principalement sur l'information et les différentes sciences qui s'y rattachent.
3	Savard, Réjean Hains, Jean-Guy Trudel, Guy	Clerc, Jean Day, John		20-04-2016	La gestion à l'ère des données ouvertes / Management in the era of open data	Cet article se penche sur les entreprises privées utilisant des données ouvertes pour améliorer leur efficacité et, plus précisément, sur la relation entre cet usage et leur style de gestion. This article tackles the use of open data by private companies as a mean to enhance efficiency and the link between that strategy and their management styles.	Entreprises privées Gestion Données ouvertes Open data Management Private companies Efficacité Efficiency	

Unité de base : Table de données (DB/TextWorks)

Au croisement d'un enregistrement et d'un champ se retrouve une **cellule** qui contient les *données* pour cet enregistrement et ce champ. Les données peuvent être de différents types : *données textuelles*, *données numériques*, *dates* par exemple.

On retrouve habituellement la possibilité dans ce modèle de données de retrouver plus d'une **occurrence** dans une cellule. Une occurrence est une valeur (entrée) distincte pour un champ et un enregistrement. Par exemple, pour la première fiche ci-dessus, on remarque qu'il y a deux entrées (deux occurrences) pour le champ *AU* comme cet article a deux auteurs : Trudel, Guy, Roy, Lucie. On parlera alors d'un champ à *occurrences multiples*.

Dans d'autres cas, un champ pourrait être défini à *occurrence simple*, c'est-à-dire qu'on ne pourra retrouver qu'une seule occurrence. C'est le cas pour le champ *DAT* comme un article est associé à une seule année de publication dans la base de données.

Finalement, on pourrait obliger la présence d'une occurrence ou plus pour des champs qui sont alors considérés *obligatoires* comme, par exemple, le champ AU comme il doit nécessairement y avoir au moins un(e) auteur(e). À l'opposé, on pourrait permettre l'absence d'occurrence pour un champ qui sera alors *facultatif*. C'est le cas du champ *TR* comme ce ne sont pas tous les articles qui nécessitent une traduction.

b) Opérations possibles pour le modèle textuel

Les bases de données du modèle textuel proposent des fonctionnalités bien adaptées aux **données textuelles**, c'est-à-dire des données composées surtout de phrases ou de mots.

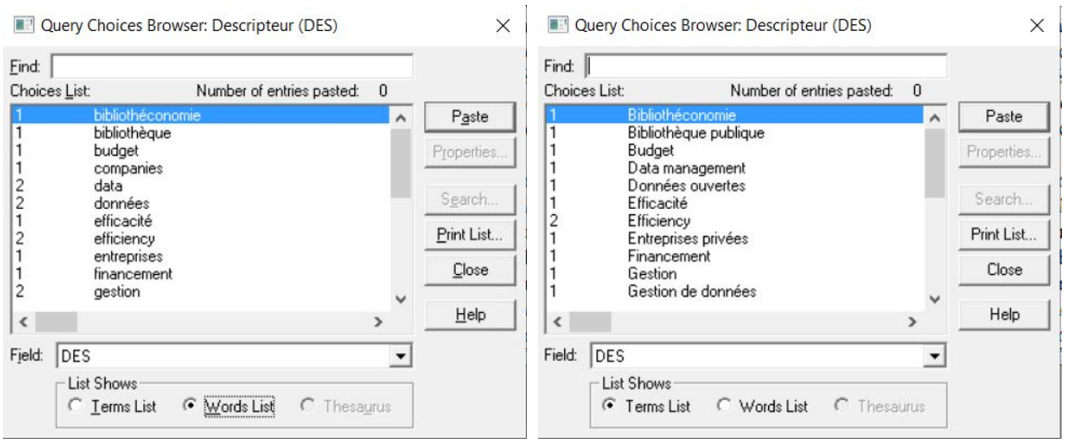
Premièrement, *DB/TextWorks*, en sus des opérateurs booléens (ET, OU, SAUF) pour la recherche, offre des opérateurs particulièrement pertinents à la recherche textuelle, soit des **opérateurs de distance**. Les opérateurs de distance permettent de chercher des mots à une certaine distance les uns des autres, dans l'ordre (par exemple "sciences" à quelques mots de "information" pour repérer "sciences de l'information" et "sciences de la communication et de l'information") ou dans le désordre (par exemple "sciences" à quelques mots de "information" pour retrouver tant "sciences de l'information" que "information science" comme illustré ci-dessous). Ils sont ainsi *fort utiles pour préciser une requête de recherche*.

NO	AU	TR	RE	DAT	TI	RES	DES	BIO
1	Roy, Lucie	Day, John	Hains, Jean Savard, Guy	01-01-2016	Les sciences de l'information aujourd'hui : une discipline à redéfinir / Information science today : the old, the new and what's to come	L'auteure propose un survol de l'état actuel des sciences de l'information ainsi qu'une définition revue des différentes branches qui la composent. Elle défend notamment l'idée que les avancées technologiques forcent une mise au point quant à la distinction entre la gestion de l'information et la gestion de données. The author describes the current state of information science and offers updated definitions of its many fields of study. She puts forward the notion that due to technological advances, we must rethink the boundaries of information management and data management.	Science de l'information Sciences de l'information Information science Information sciences Information management Gestion de l'information Data management Gestion de données	Lucie Roy est professeure titulaire à l'École des Sciences et de l'Information où elle enseigne depuis plus de vingt ans. Elle dirige la Chaire de recherche sur l'évolution des sciences de l'information et participe à de nombreux autres groupes de recherche internationaux. Ses nombreuses publications portent principalement sur l'information et les différentes sciences qui s'y rattachent. / Lucie Roy is a professor at the School of Sciences and Information. Her teaching career spans more than twenty years. She is a member of a number of international research groups and heads the Research Chair on the evolution of information sciences. She has published numerous books and articles on information and its sciences.
4	Clerc, Jean		Roy, Lucie Savard, Guy	25-12-2015	What helps and what doesn't : Efficiency measures in public libraries	Jean Clerc offers a thorough analysis of efficiency measures implemented in public libraries and their success (or lack thereof) in helping maintain a balanced budget. The author first suggests different criteria of success and then applies them to different real-life examples of management strategies in public libraries.	Information science Library science Librarianship Public library Efficiency Budget Management	Jean Clerc is a postdoctoral researcher at the School of Information of Northern Labrador. His PhD thesis The new librarian: A better way to manage libraries, written under the guidance of Lucie Roy, has recently been published by SINL Press. He has authored, coauthored and translated many articles on a wide array of subjects, including librarianship, library management, efficiency, and strategic measures.

Utilisation de l'opérateur de distance dans le désordre (DB/TextWorks)

Deuxièmement, dans cet environnement, la recherche se fait nécessairement à partir des *index construits* sur la base des données présentes dans les champs. On parlera ainsi d'une **recherche indexée**. Un champ qui n'est pas indexé, dans ce SGBD, n'est pas cherchable. Cette base de données textuelle **indexe** les champs de deux manières décrites ci-dessous et illustrées par la suite :

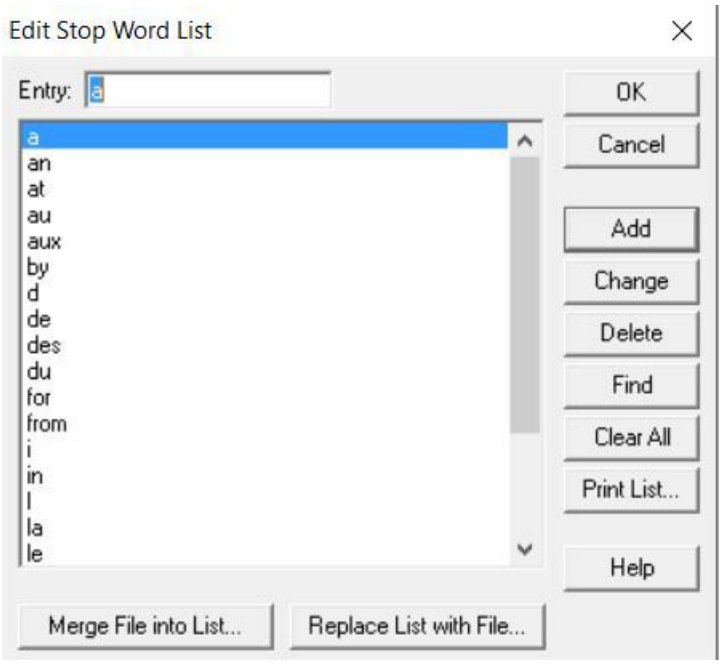
- Les données saisies dans un champ (par exemple, pour le champ *DES*, l'expression "Gestion de données") peuvent être découpées par mot dans un **index mots** (on retrouvera ainsi dans l'index mots une entrée pour "gestion" et une entrée pour "données").
- Une entrée complète dans un champ (par exemple, "Gestion de données") peut se retrouver telle quelle dans un **index occurrences** sans être découpée.



Index mots **Index occurrences**

Sans entrer dans les détails, l'existence de ces deux types d'index couplée au comportement des différents opérateurs de recherche permet des *recherches textuelles plus précises*.

Finalement, on sent très bien la "*sensibilité textuelle*" dans ce SGBD du fait qu'il offre à ses utilisateurs, en sus des opérateurs de distance et des deux types d'index, une *prise intéressante sur les index* comme on peut, d'une part, les consulter comme illustré ci-dessus, mais aussi en *contrôler le vocabulaire* en **rejetant les mots vides de sens** :



Liste des mots vides rejetés lors de la création de l'index mots (DB/TextWorks)

c) Modélisation d'une base de données textuelle

La **modélisation d'une base de données textuelle** passera par les trois étapes habituelles d'une modélisation soit (1) sa conceptualisation, (2) sa représentation et (3) sa *validation*.

Conceptualisation

La **conceptualisation d'une base de données textuelle** consiste principalement à **décider des champs** qui permettront de bien représenter le contexte pour permettre les opérations désirées et à en **définir les caractéristiques**. Les champs dans une base de données représentent ainsi certaines caractéristiques sur lesquelles on désire conserver des informations. Dans l'exemple de la base de données *REPSCIE*, les concepteurs de la base de données ont eu à identifier, avec l'aide du comité éditorial de la revue, les caractéristiques qui leur seraient utiles pour gérer les articles acceptés pour publication. Ainsi, avant de développer une base de données dans un SGBD, il faut commencer par identifier les champs pertinents et leurs caractéristiques, notamment :

- Nom du champ
- Description du contenu du champ
- Type de données (texte, nombre, date, ...)
- Statut obligatoire ou facultatif
- Type d'indexation (index mots et/ou index occurrences)
- Occurrences simples ou multiples
- Validation (par exemple, un masque pour la saisie pour respecter un format précis de date ou la saisie à partir d'une liste de mots prédéfinis)
- Règles d'écriture s'il y a lieu
- Exemples de contenu valide

Représentation

La **représentation de cette conceptualisation** (*deuxième étape de la modélisation*) se fera en consignant l'ensemble des champs et leurs caractéristiques dans le dictionnaire de données de la base de données. Ce dictionnaire servira à l'implantation, par la suite, de la base de données dans l'environnement retenu afin de définir la structure de données.

On y retrouverait, par exemple, les informations suivantes dans le dictionnaire de la base de données *REPSCIE* pour le champ AU :

Caractéristiques du champ AU dans la base de données REPSCIE

Caractéristique	**Valeur**
Nom du champ	AU
Description du contenu	Liste des auteurs de l'article
Type de données	Textuelle
Statut	Obligatoire
Type d'indexation	Index mots et Index occurrences
Occurrences	Occurrences multiples
Validation	Liste de validation des noms d'auteur
Règles d'écriture s'il y a lieu	(Nom de famille, avec majuscule initiale) (virgule) (espace) (prénom au complet, si connu, initiale sinon, avec première lettre en majuscule) suivi, si nécessaire, de : (espace) (particule de nom de famille, telle qu'écrite dans le document) ou de : (espace) (initiale, en majuscule) (point)
Exemples de contenu valide	Roy, Lucie Gardner, Richard K. Rochelière, Luc de la

Validation

La validation consiste à s'assurer que le dictionnaire de données permet de **bien représenter le contexte**. Cela peut se faire entre autres par un *retour auprès des personnes ayant émis le besoin* d'une base de données pour qu'ils en vérifient l'exactitude.

5.3. Modèle relationnel

Le deuxième modèle que l'on retrouve souvent en contexte documentaire est celui des **SGBD relationnels**. Le mot "modèle" prend dans ce cas-ci tout son sens : avant d'être implémenté sur un ordinateur, son créateur, l'informaticien britannique Edgar Frank "Ted" Codd, l'a défini mathématiquement formellement en 1970. Le premier produit basé sur ce modèle a vu le jour à la fin des années 70. De cette manière, il était possible d'éviter les problèmes rencontrés par certains modèles précédents, notamment le modèle réseau et le modèle hiérarchique, qui, n'ayant pas été formellement définis, mais plutôt directement implémentés, n'ont pas passé le test du temps et sont devenus obsolètes.

Il s'agit du modèle encore **le plus utilisé actuellement**. Beaucoup de systèmes intégrés de bibliothèques (SIGB) et autres applications documentaires sont construits sur des SGBD relationnels. Il est aussi très présent au niveau des systèmes d'information Web, quoiqu'il ne soit plus le seul! Ses **limites** pour les systèmes distribués à **grande échelle** sur le Web comme Twitter et Facebook ont conduit à l'apparition des familles de bases de données NoSQL (*Not Only SQL*) qui seront abordées rapidement un peu plus tard.

Contexte des exemples — ***Remarque***

Afin d'illustrer nos propos, nous utiliserons la **base de données fictive Dépenses Papeterie** développée avec le *SGBD relationnel Base de LibreOffice*. Cette base de données, inspirée du *TP Tableur*, contient les informations sur les dépenses en papeterie de la firme *ABC Courtage informationnel*. On y retrouve de l'information sur les items achetés (champs ID_ITEM pour un numéro d'identification, ITEM pour le descriptif de l'item et COMMENTAIRE pour des commentaires sur les items), leur coût unitaire (champ COUT_UNITAIRE), le nombre d'items achetés (champ NBRE), la date d'achat (champ DATE), la ou les personnes ayant effectué l'achat (champ ID_RESPONSABLE pour un numéro d'identification, NOM pour le nom) et des commentaires sur l'achat (champ COMMENTAIRE).

a) Types de SGBD relationnels

On retrouve des SGBD relationnels de **taille variable** selon les contextes et les besoins auxquels ils répondent.

Pointures de SGBD relationnels (SGBDR)

Petite pointure = SGBDR personnels	**Grande pointure = SGBDR corporatifs**
• Des BD relativement **petites**, pour une personne ou un groupe restreint • Un volume des données et/ou d'accès **réduit** • Des situations où la performance n'est **pas un facteur critique** • Ex. : *Access* (*Microsoft Office*), *Base* (*LibreOffice*), *FileMaker Pro*, *MySQL*	• **Haute performance** • Habituellement sur un **serveur dédié** • Permet l'intégration de **toutes** les BD d'une organisation • Rôle de l'**administrateur de bases de données** très important • Ex. : *Oracle*, *SQL Server*

On peut distinguer **deux scénarios principaux d'implantation** des SGBD relationnels :

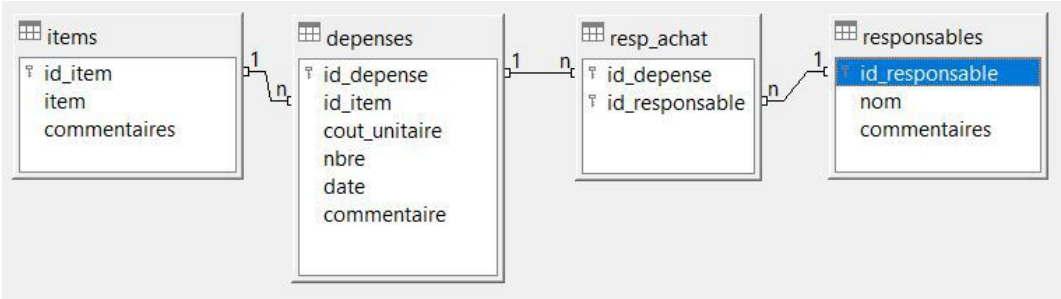
- *Scénario 1* : Base de données relationnelle sur un **ordinateur personnel** (par exemple *Access* (*Microsoft Office*), *Base* (*LibreOffice*), *FileMaker Pro*)
    - *Un seul utilisateur* à la fois
    - *Recherche d'information directement dans le SGBD* via des assistants (recherche simple) ou SQL (recherche experte)
    - *Entrée des données directement dans le SGBD* (pour l'administrateur(trice) de données) ou via un ou des *formulaires préparés dans le SGBD* (pour les responsables de contenu)
    - Présentation des données à partir de *rapports préparés dans le SGBD*
- *Scénario 2* : Base de données relationnelle sur le **Web** (par exemple, le *SGBD MySQL* sur un serveur Web)
    - Architecture distribuée permettant de *multiples utilisateurs* et des *systèmes d'exploitation variés* (accès via un navigateur Web)
    - *Recherche d'information directement dans le SGBD* (pour l'administrateur(trice) de données) *ou via des interfaces Web* (pour le grand public)
    - *Entrée des données directement dans le SGBD* (pour l'administrateur(trice) de données) ou *via un ou des formulaires HTML* (pour les responsables de contenu)
    - Présentation des données à l'aide de *pages Web dynamiques* (par exemple en PHP ou en ASP)

Dans le cadre du *TP Base de données*, vous développerez une base de données de petite pointure selon le premier scénario. Bien que le scénario 2 soit de plus en plus présent, il implique des aspects de développement Web qui vont au-delà des attentes pour ce cours (il s'agit toutefois du scénario retenu pour le cours SCI6306!).

b) Structure de données du modèle relationnel

Les SGBD relationnels ressemblent aux SGBD textuels quant à l'**unité de base** utilisée pour structurer les données, soit la **table de données**. Deux *différences importances* existent cependant à ce niveau.

- Premièrement, une base de données relationnelle peut comporter **plus d'une table de données**, les différentes tables étant **reliées** entre elles.
  - Dans l'exemple utilisé, on retrouve *quatre tables de données* comme illustré ci-dessous, soit (1) une table pour gérer la liste des items achetés (table ITEMS), (2) une table pour gérer le détail des achats (table DEPENSES), (3) une table pour gérer la liste des personnes responsables des achats (table RESPONSABLES) et (4) une table pour faire le lien entre un achat et la ou les personnes qui en sont responsables (table RESP_ACHAT) :



Structure de tables et relation (Base)

- Deuxièmement, une *cellule* dans une table de données ne peut comporter qu'**une et une seule valeur** :
  - On ne peut ainsi y retrouver d'*occurrences multiples*. C'est ce qui explique la table RESP_ACHAT qui permet d'associer plus d'un responsable à un achat. Remarquez dans cette table, pour la dépense #167, qu'on retrouve justement deux lignes, une ligne par responsable.
  - De plus, on ne peut y retrouver de *cellule vide*; pour un champ facultatif, l'absence de données est représentée par la valeur NULL. Les SGBD varient quant à l'affichage des valeurs nulles. Dans l'exemple ci-dessous, *Base* laisse tout simplement la cellule sans contenu. Dans *MySQL*, un autre SGBD relationnel, il indique explicitement *NULL*.

id_item	item	commentaires
0	5000 feuilles 8,5x11	
1	5000 feuilles 8,5x14	
2	Bloc-notes (5/paquet)	
3	Chemises legal (100)	
4	Chemises lettre (100)	
5	Post it (12 par paquet)	
6	Stylo (12 par paquet)	

Table ITEMS

id_depense	id_item	cout_unitaire	nbre	date	commentaire
167	0	55,34	2	2015-01-15	
168	5	23,06	1	2015-01-15	
169	2	5,53	1	2015-01-15	
170	4	23,06	2	2015-01-15	
171	3	25,83	1	2015-01-15	
172	6	2,77	2	2015-01-30	
173	0	55,34	2	2015-02-15	
174	0	55,34	2	2015-03-15	
175	1	92,24	1	2015-03-15	
176	0	55,34	2	2015-04-15	
177	2	5,53	2	2015-04-15	
178	0	55,34	2	2015-05-15	
179	0	55,34	1	2015-06-15	
180	5	23,06	3	2015-06-15	

Table DEPENSES

id_depense	id_responsable
167	6
167	19
168	3
168	7
169	13
170	0
171	9
172	1

Table RESP_ACHAT

id_responsable	nom	commentaires
0	Barnett, Tyqueria	
1	Frazier, Kashayla	
2	Gaudet, Sénard	
3	Hatsuka, Tano	
4	Johnson, Darelle	
5	Jones, Samantha	
6	Juneau, Arthurien	
7	Kiko, Nagashima	
8	Laberge, Tommy	
9	Lacoste, Edward	
10	Laprise, Ophina	
11	Loiselle, Laura	
12	Love, Addrian	
13	Sénécal, Roméric	

Table RESPONSABLES

Contenu des tables (Base)

Ces deux différences font en sorte que leur modélisation se fait différemment de la modélisation pour le modèle textuel, comme nous le verrons un peu plus tard.

c) Opérations possibles pour le modèle relationnel

La "**sensibilité au texte**" du modèle textuel *ne se retrouve pas* dans le modèle relationnel :

- Les *opérateurs de distance* ne s'y retrouvent que sous certaines conditions et sont ainsi *moins accessibles*.
- Les *index* ne sont *pas nécessaires pour la recherche*. La recherche peut en effet se faire tant à partir d'un index (*recherche indexée*) que sans index (*recherche séquentielle*).
- On ne retrouve qu'*un index mots*, index *que l'on ne peut consulter* et sur lequel *peu de contrôle* est donné.

Par contre, le langage d'interrogation **SQL** (*Structured Query Language*) offre une *multitude d'opérateurs* que l'on ne retrouve pas dans le modèle textuel, comme, par exemple :

- Des opérateurs *mathématiques* (addition, soustraction, etc.)
- Des opérateurs *statistiques* (moyenne, médiate, etc.)
- Des opérateurs pour *traiter le texte* (extraction de caractères, longueur d'une chaîne, transformation de la casse, etc.)

Le langage d'interrogation permet de représenter des **besoins d'information très complexes** grâce à ces opérateurs et aux différentes fonctions du langage SQL (fonctions d'agrégation, sous-requêtes, union de requêtes, etc.). Ce que l'on perd en recherche textuelle, on le gagne en richesse des opérateurs et des fonctions disponibles. L'exemple ci-dessous illustre la possibilité de regrouper les données par année et par item afin de calculer le nombre total d'items achetés, leur coût unitaire moyen ainsi que le coût total :

depenses_parannee_paritem - bd_depenses2 - LibreOffice Base : vue de...

Fichier Édition Affichage Insertion Données Outils Fenêtre Aide

Item	Année	Nombre d'items	Coût unitaire moyen	Coût total
5000 feuilles 8,5x11	2015	19	55,34	1051,46
5000 feuilles 8,5x11	2016	19	55,9	1062,1
5000 feuilles 8,5x11	2017	20	57,63	1152,6
5000 feuilles 8,5x11	2018	20	58,21	1164,2
5000 feuilles 8,5x11	2019	20	59,4	1188
5000 feuilles 8,5x11	2020	20	60,6	1212
5000 feuilles 8,5x14	2015	1	92,24	92,24
5000 feuilles 8,5x14	2016	1	93,17	93,17
5000 feuilles 8,5x14	2017	4	96,05	384,2
5000 feuilles 8,5x14	2018	4	97,02	388,08
5000 feuilles 8,5x14	2019	4	99	396
5000 feuilles 8,5x14	2020	4	101	404
5000 feuilles 8,5x14	2021	2	102	204
Bloc-notes (5/paquet)	2015	5	5,53	27,65

Enregistrement de 41 *

Résultat d'une requête SQL permettant de regrouper les données par année et par item pour faire différents calculs (Base)

d) Modélisation d'une base de données relationnelle

La **modélisation sémantique** d'une base de données relationnelle permet d'ajouter à la compréhension d'une base de données et d'ainsi pouvoir répondre plus intelligemment aux interactions de l'utilisateur. Cette modélisation est utile au processus de *conception systématique* des bases de données. Elle se fait habituellement en *dehors du SGBD*.

L'objectif de cette modélisation est de représenter une certaine réalité selon le modèle relationnel pour pouvoir construire, par la suite, une base de données relationnelle.

On retrouvera comme dans les modélisations présentées précédemment, les trois étapes que sont (1) la *conceptualisation*, (2) la *représentation* et (3) la *validation*. La conceptualisation et la représentation se font souvent en parallèle, l'une se nourrissant de l'autre.

Conceptualisation et représentation

Du fait qu'une base de données relationnelle peut contenir plusieurs tables, la conceptualisation ne peut se contenter, comme pour le modèle textuel, de s'attarder uniquement aux champs à inclure dans la base de données du fait, notamment, qu'il peut y avoir plus d'une table de données. Il faut ainsi dans un premier temps conceptualiser la réalité à y représenter pour la découper en ses différentes composantes. Plusieurs approches peuvent être utilisées pour représenter la conceptualisation d'une BD relationnelle, dont l'**approche entités-relations (E-R)** qui est une des plus connues et utilisées.

Cette approche est fondée sur le modèle E-R défini par Chen (1976) et raffiné par la suite. On y retrouve représentés plusieurs *objets sémantiques* :

- **Entités** : objets que l'on peut et veut distinguer (par exemple, des dépenses, des items, etc.).
- **Relations** : connexions entre des entités (par exemple, des achats qui relient des dépenses à des personnes responsables). Une relation possède une **cardinalité** qui représente la manière dont les enregistrements des deux entités connectées sont liés.
- **Attributs** : propriétés décrivant une entité ou une relation (par exemple, le nom d'une personne responsable) que l'on veut documenter dans la BD.

Un des résultats de la modélisation entités-relations est un **diagramme Entités-Relations** qui encapsule *visuellement* les différents objets sémantiques identifiés. Classiquement, les entités sont représentées par des *rectangles*, les relations par des *losanges* et les attributs par des *ovales* (les attributs, lorsqu'ils sont nombreux, peuvent aussi être représentés à même le rectangle de son entité).

Comme illustré ci-dessous, la modélisation du contexte de la base de données *Dépenses Papeterie* a permis de comprendre qu'on y retrouvait :

- **Trois entités** : l'inventaire des *types d'items* de papeterie, les *dépenses* effectuées et la liste des *personnes responsable des achats* chez *ABC Courtage informationnel*
- **Deux relations** :
  - Afin d'éviter de surcharger la base de données, seul le numéro des items (champ ID_ITEM) est conservé dans la table DÉPENSES. Pour connaître l'intitulé d'un item, la table DÉPENSES est liée à la table ITEMS. Une dépense n'est associée qu'à un seul item à la fois tandis qu'un des items de l'inventaire peut être associé à plusieurs dépenses. Il s'agit d'une relation de cardinalité 1-N.
  - Chaque dépense est associée potentiellement à une ou plusieurs personnes responsables des achats. La table DÉPENSES est ainsi liée à la table RESPONSABLES. La relation est de cardinalité N-N du fait qu'une dépense peut être associée à plus d'une personne et qu'une personne peut être associée à plus d'une dépense.
- **Plusieurs attributs** : Pour chacune des entités, différentes caractéristiques sont enregistrées dans la base de données comme, par exemple, le coût unitaire pour une dépense ou le nom pour une personne responsable.

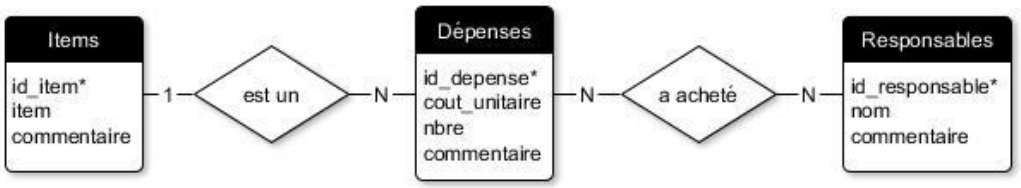


Diagramme Entités-Relations pour la base de données Dépenses Papeterie

Une fois le diagramme Entités-Relations défini, l'étape suivante consiste à traduire cette représentation**en structure de tables et de champs** et, tout comme pour les bases de données textuelles, à identifier les **caractéristiques des champs** nécessaires pour représenter le contexte désiré :

- Nom du champ
- Description du contenu du champ
- Type de données (texte, nombre, date, ...)
- Statut obligatoire ou facultatif
- Index
- Validation (par exemple, un masque pour la saisie pour respecter un format précis de date ou la saisie à partir d'une liste de mots prédéfinis)
- Règles d'écriture s'il y a lieu
- Exemples de contenu valide

Du fait de la relation entre les tables découlent deux éléments supplémentaires à définir :

- Le ou les champs qui serviront de ***clé primaire*** dans les tables. En effet, afin de *pouvoir identifier de manière unique chacun des enregistrements* dans une table, il faut qu'on y retrouve un champ (ou une combinaison de champs) dont les valeurs sont uniques. Par exemple, dans la base de données *Dépenses Papeterie*, les attributs indiqués dans le diagramme Entités-Relations par l'astérisque sont les champs de clés primaires. Chaque dépense possède son numéro unique (champ ID_DEPENSE), de même pour les items et les responsables.
- Le ou les champs qui serviront de ***clé externe*** dans les tables. Pour *"faire le pont" entre deux tables liées*, il faut avoir de part et d'autre un champ en commun qui, d'un côté sera la clé primaire et, de l'autre côté la clé externe. Si on reprend l'exemple des dépenses, pour pouvoir relier une dépense dans la table DEPENSES à la description complète de l'item dans la table ITEMS, on y ajoute le champ ID_ITEM comme clé externe pour pointer le champ ID_ITEM qui est la clé primaire de la table ITEMS.

Par exemple, à partir du diagramme *Entités-Relations* de la base de données *Dépenses Papeterie*, il est possible de déduire que la base de données possédera **quatre tables**, soit une pour chacune des entités (items, dépenses, responsables) et une pour le lien entre les dépenses et les responsables du fait de la cardinalité N-N. Il faudrait par la suite décrire chacune des tables pour préciser les caractéristiques des champs, la clé primaire, et, s'il y a lieu, la ou les clés externes. Par exemple, pour la table RESPONSABLES :

Caractéristiques des champs de la table RESPONSABLES

Champ	Type de données	Taille du champ (s'il y a lieu)	Statut
id_responsable	numérique	2	Obligatoire
nom	caractère	100	Obligatoire
commentaire	caractère	500	Facultatif

Clé primaire : id_responsable

Description du contenu des champs

- *id_responsable* : identifiant numérique unique séquentiel attribué automatiquement par le système
- *nom* : nom d'une personne responsable des achats sous la forme (Nom de famille, avec majuscule initiale) (virgule) (espace) (prénom au complet, si connu, initiale sinon, avec première lettre en majuscule) suivi, si nécessaire, de : (espace) (particule de nom de famille, telle qu'écrite dans le document) ou de : (espace) (initiale, en majuscule) (point)
- *commentaire* : commentaire sur la personne responsable des achats (omettre la ponctuation finale)

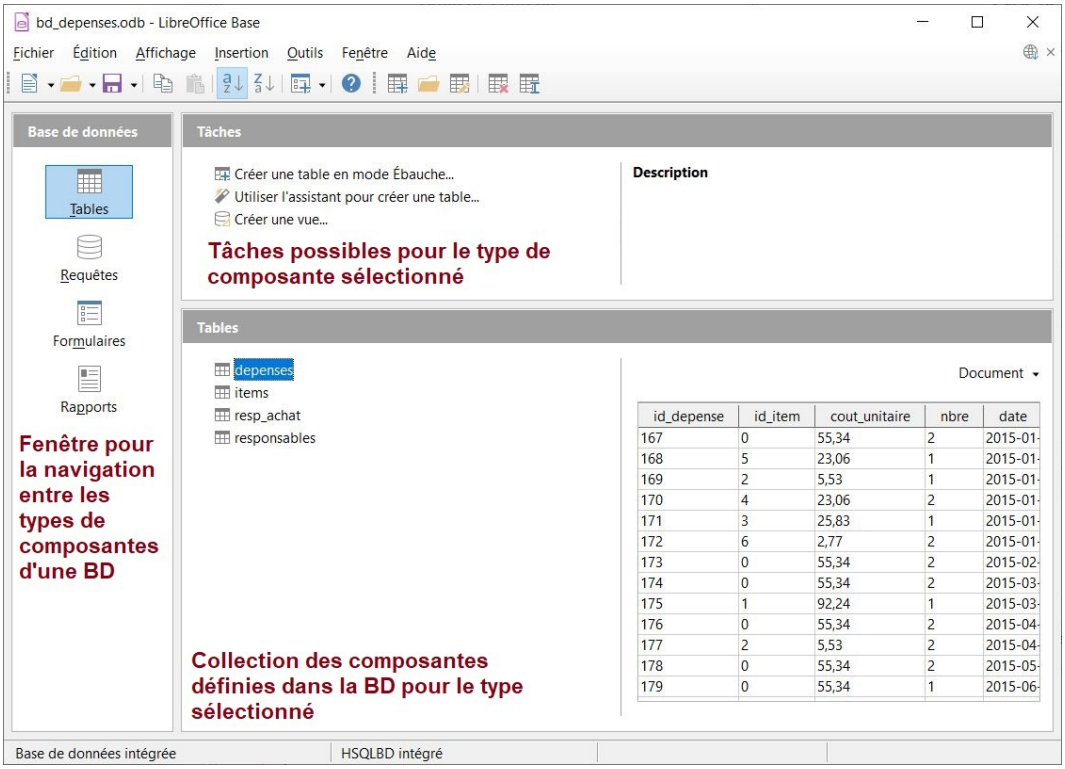
e) Implantation de la modélisation dans un SGBD (Base)

Remarque ***Remarque***

Cette section vise à donner **quelques points de repère utiles** pour le *TP Base de données* (section 4) pour la **création de la structure de données** dans la base de données. Nous n'y aborderons que le contexte de *Base*. Il est à noter que l'on ne retrouve pas l'équivalent pour les autres modèles de données présentés dans les notes de cours comme vous n'aurez pas à développer de BD suivant ces modèles.

Points de repère dans l'interface de Base ***Complément***

Comme illustré ci-dessous, vous retrouverez **trois zones principales** dans l'interface de *Base*. D'une part, vous pouvez sélectionner le *type de composante* (tables, requêtes, ...). D'autre part, une fois un type de composante sélectionné, vous pouvez choisir une des *tâches* qui lui est associé ou ouvrir/éditer un des éléments de sa *collection*.



Principales zones de l'interface de Base

Introduction

Après avoir demandé dans *Base* à créer une nouvelle base de données, vous aurez à y **reproduire** la structure décrite dans votre **modélisation**. Concrètement, vous aurez à créer (1) les *tables* de données, (2) les *relations* entre les tables, et (3) les *index*. Chacune de ces étapes sont décrites ci-dessous et illustrées à partir de l'exemple de la base de données *Dépenses Papeterie* présenté précédemment.

i) Création des tables

On retrouve deux manières principales de créer des tables dans *Base* : (1) par **importation** de données existantes et (2) à partir de **zéro** (mode *Ébauche*). Il est à noter qu'il est aussi possible d'utiliser l'assistant afin de créer une table à partir de modèles prédéfinis de table.

Importation de données existantes

S'il s'avère que vous aviez déjà des données saisies dans une **feuille de données** du tableur *Calc*, il est possible de vous servir de ces données comme *base pour créer une table*. Cela évitera ainsi d'en faire la resaisie. Le principe est simple. Il s'agit de :

1. **Sélectionner les données** dans le tableur *Calc* pour les **copier**
2. Dans *Base*, dans la fenêtre pour la *collection de tables*, de faire un *clic-droit* et de **coller** les données

L'assistant *Coller une table* s'ouvrira qui vous permettra de sélectionner les colonnes que vous voulez importer (chaque colonne sélectionnée deviendra un champ) et d'en définir les caractéristiques (nom du champ, type de champ, entrée requise, longueur). L'assistant aura déjà "pré-rempli" les informations sur les caractéristiques du champ grâce à une reconnaissance automatique du type, ce qui sauve du temps bien entendu, mais il faut tout de même prendre la peine de valider que les choix effectués demeurent les bons. Il est à noter que vous pourrez par la suite retourner à la table pour en modifier les caractéristiques au besoin. Il faut ainsi définir les caractéristiques de chacun des champs en fonction des décisions prises lors de la modélisation.

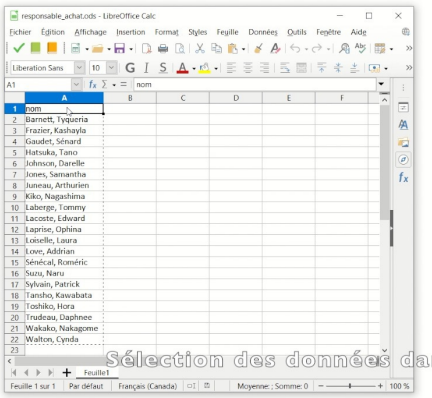
Lorsque vous demandez à créer la table, l'assistant vous offrira la possibilité d'ajouter un **champ de clé primaire** (c'est-à-dire un champ à valeur unique). Si vous n'aviez pas déjà prévu ce champ dans les données de *Calc*, il n'y a qu'à lui dire *Oui*.

Le ressource ci-dessous illustre, pour la base de données *Dépenses Papeterie*, la **création de la table des responsables des achats** à partir de données dans *Calc*.

Note : Capsule vidéo accessible en ligne[1]

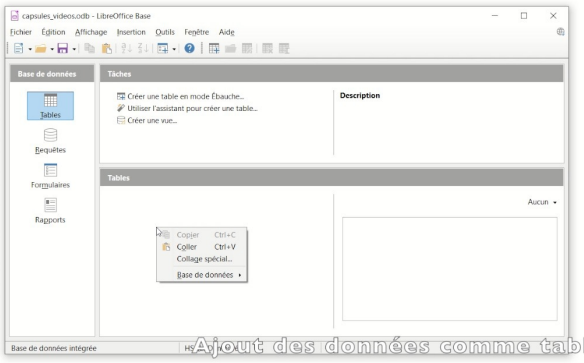
Première étape : **Copie** des données dans *Calc*

[1] http://cours.ebsi.umontreal.ca/sci6005/h2021/res/base_importation_table.mp4



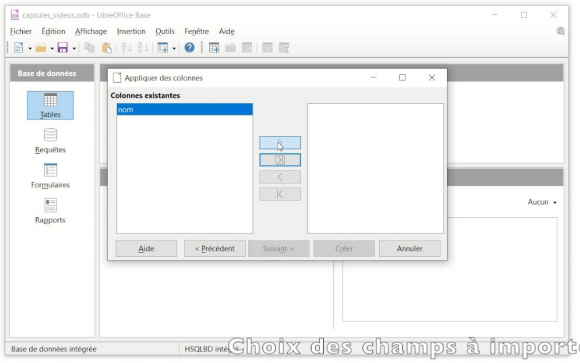
Données sélectionnées et copiées dans Calc (sélection aussi de l'entête des colonnes)

Deuxième étape : **Collage** des données dans *Base*



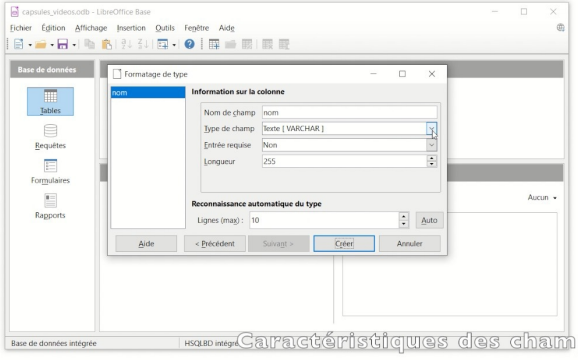
Données collées dans Base (clic-droit dans la zone des tables)

Troisième étape : **Choix** des **champs** à importer



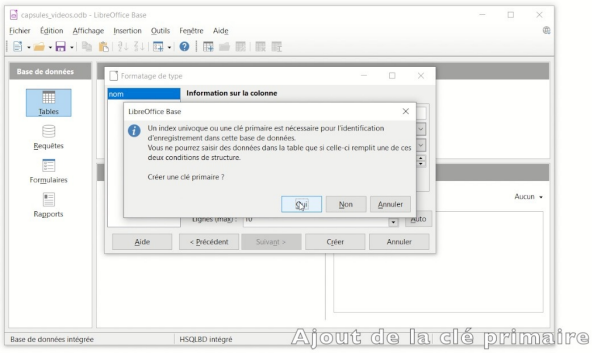
Transfert des champs à importer dans la boîte de droite

Quatrième étape : Définition des **caractéristiques des champs** importés



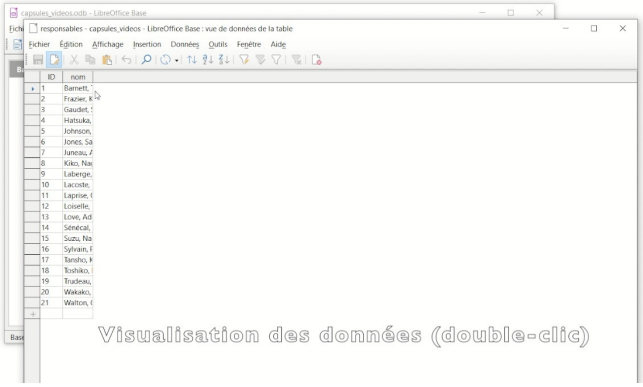
Définition des caractéristiques de chacun des champs en fonction de la modélisation

Cinquième étape : Créer la **clé primaire**



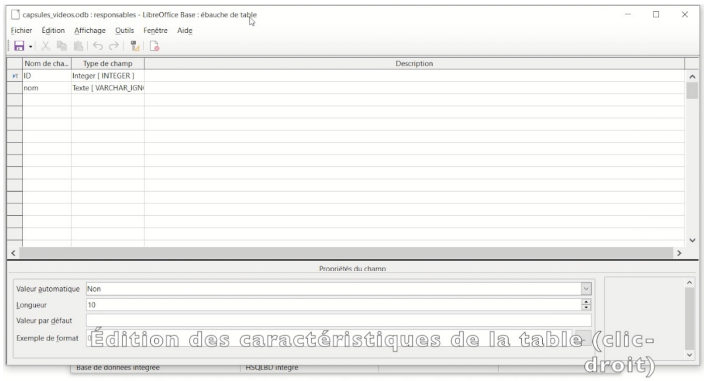
Ajout d'une clé primaire en fermant la fenêtre

Sixième étape : **Visualisation** des **données importées** dans la table



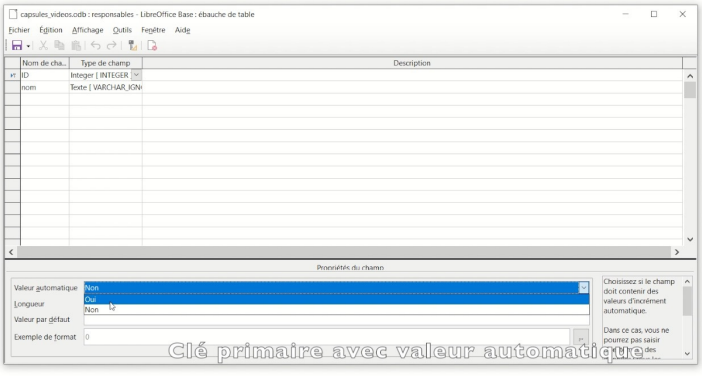
Visualisation des données importées pour s'assurer que l'importation a été réussie (double-clic sur la table)

Septième étape : **Édition des caractéristiques** de la table pour les ajuster



Édition au besoin des caractéristiques des champs (clic-droit sur la table > Éditer)

Huitième étape : Définition de la clé primaire à **valeur automatique**



Exemple d'ajustement : Redéfinition de la clé primaire pour avoir une valeur automatique

Création d'une table (mode Ébauche)

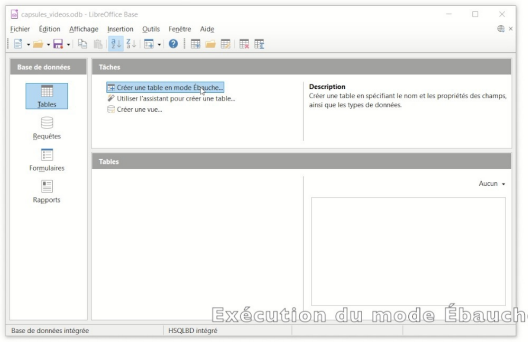
Il est aussi possible de créer une table **à partir de zéro** grâce au mode *Ébauche*. Ce dernier permet de **créer les champs** et d'en **préciser les caractéristiques**, en particulier :

- Le *nom* du champ;
- Le *type de données* saisies dans le champ (par exemple *Integer* pour des nombres entiers, *Varchar* pour du texte, *Varchar_Ignorecase* pour du texte pour lequel on veut ignorer la casse - majuscules et minuscules - lors de la recherche);
- La *clé primaire* (clic-droit sur la ligne du champ de clé primaire);
- Le *caractère obligatoire ou facultatif* du champ (entrée requise ou non);
- La *longueur maximale* d'une entrée (par exemple, le plus grand nombre de caractères que l'on pourrait retrouver pour une valeur dans un champ).

La ressource ci-dessous présente le processus pour **ajouter la table des responsables des achats** cette fois à partir de zéro.

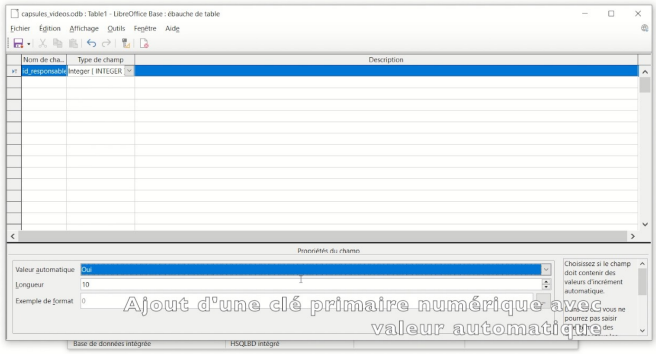
Note : Capsule vidéo accessible en ligne[1]

Première étape : Exécution du **mode *Ébauche***



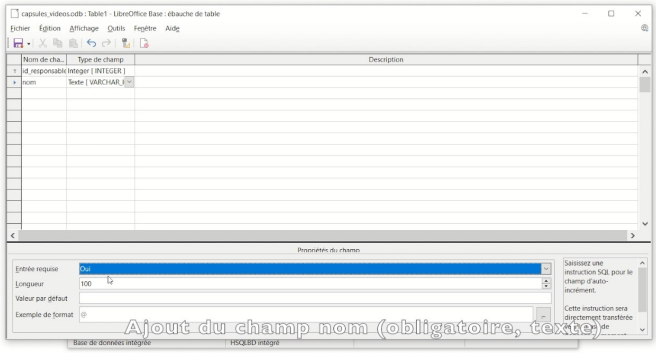
Accès au mode Ébauche pour créer une nouvelle table

Deuxième étape : Ajout du **champ *id_responsable*** (clé primaire, valeur numérique automatique)



Définition du champ id_responsable comme clé primaire (clic-droit sur sa ligne) numérique automatique

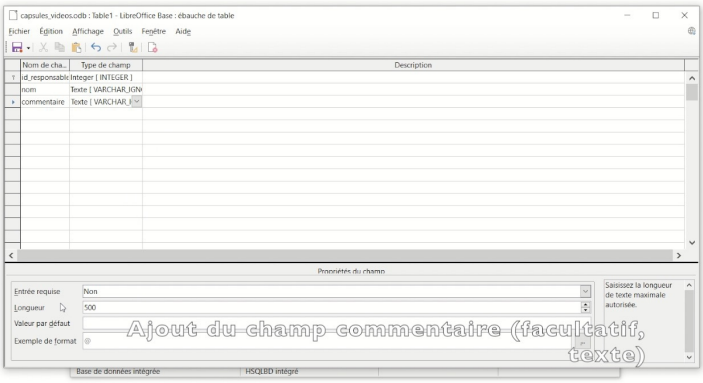
Troisième étape : Ajout du **champ *nom*** (champ obligatoire, texte)



Ajout du champ nom (obligatoire, texte)

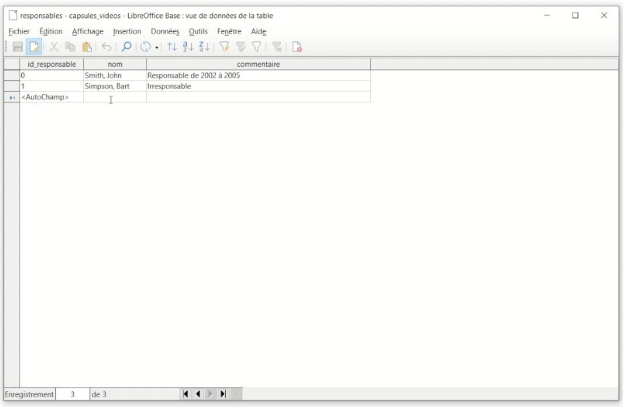
[1] http://cours.ebsi.umontreal.ca/sci6005/h2021/res/base_table_creation.mp4

Quatrième étape : Ajout du **champ *commentaires*** (champ facultatif, texte)



Ajout du champ commentaire (facultatif, texte)

Cinquième étape : **Saisie manuelle des données** dans la table (double-clic sur la table pour l'ouvrir)



Saisie manuelle des données (double-clic sur la table pour l'ouvrir en mode saisie)

ii) Création des relations

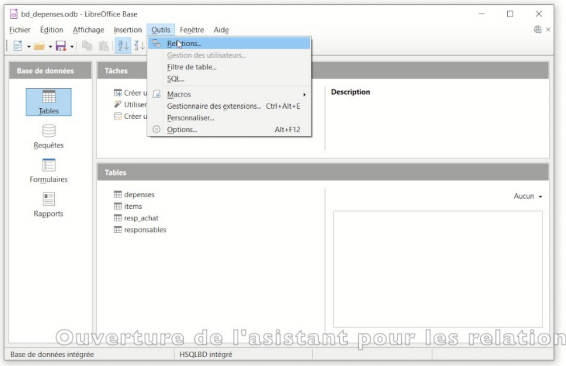
Une fois vos tables créées, il est important de **définir explicitement les relations** qui les lient, relations que vous avez préalablement précisées lors de la modélisation de votre base de données. Vous pourriez ne pas créer les relations, mais la BD ainsi développée sera moins stable et moins performante.

La création d'une relation se fait en deux étapes : (1) **lier les tables**, et (2) définir ce qui se passera **si on modifie ou supprime** des informations dans une des tables liées. Pour le deuxième point, il y a deux types d'action à définir : ce qui se passe si on modifie le champ qui sert de clé primaire et ce qui se passe si on supprime un enregistrement. Nous n'entrerons pas dans les détails ici, comme cela dépasse les attentes pour le cours.

La ressource ci-dessous illustre la création de la relation entre la table où se retrouve les informations sur les items achetés et la table où se trouve consignée le détail des dépenses encourues.

Note : Capsule vidéo accessible en ligne[1]

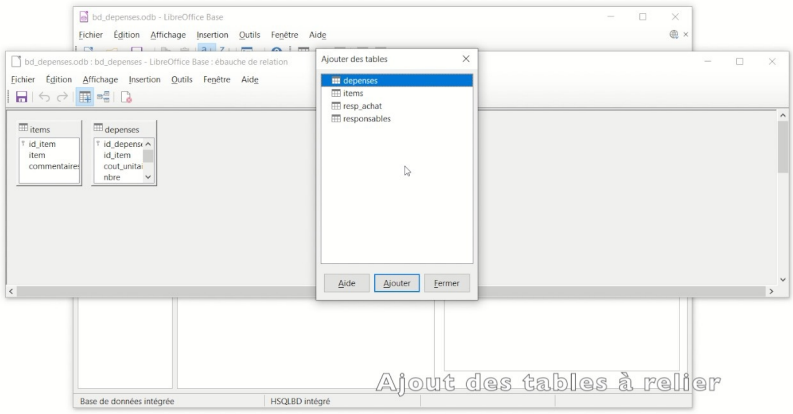
Première étape : Activation de l'**assistant** relations



Ouverture de l'assistant relations

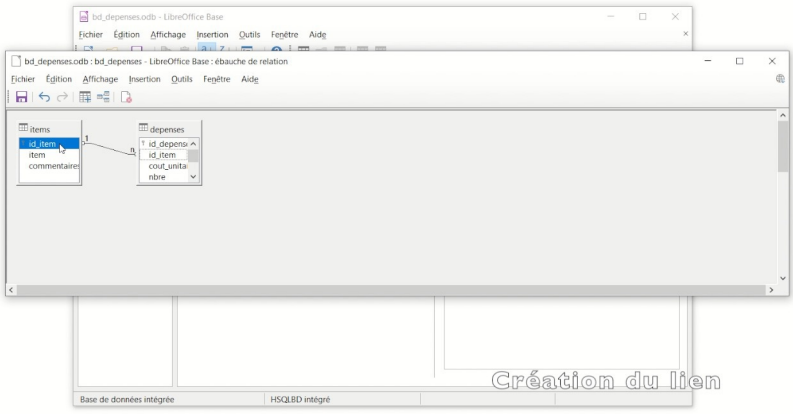
[1] http://cours.ebsi.umontreal.ca/sci6005/h2021/res/base_relations_creation.mp4

Deuxième étape : Ajout des **tables**



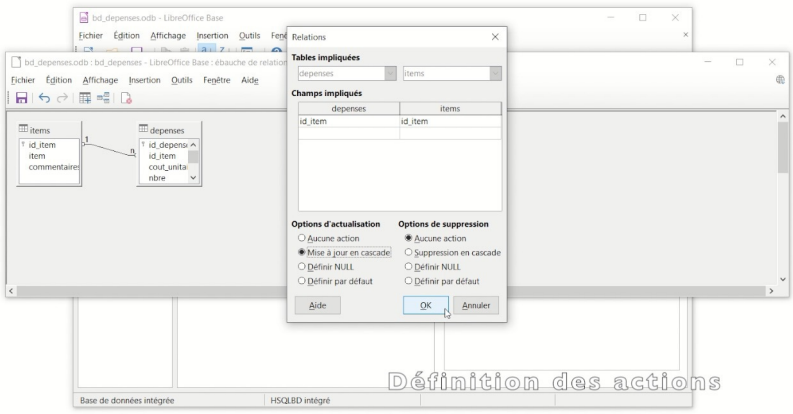
Ajout des tables à lier

Troisième étape : Création du **lien** entre les tables



Création du lien entre les deux tables (clic sur la clé externe et glissement sur son équivalent clé primaire)

Quatrième étape : Définition des **actions**



Précision des actions (modifications en cascade et aucune action si suppression)

iii) Création des index

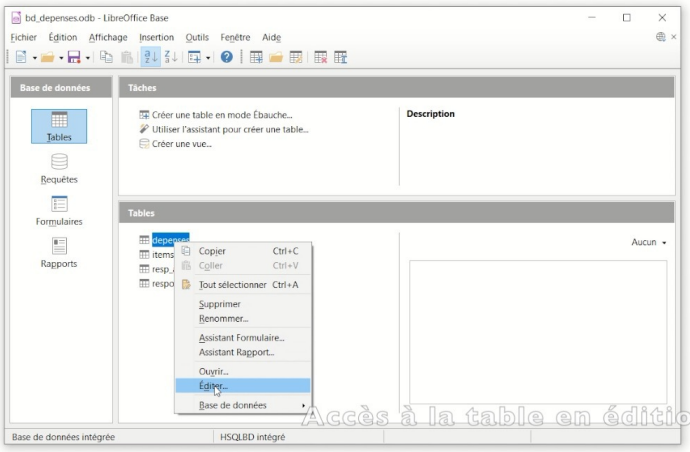
Bien qu'il soit possible de faire des **recherches** dans un **champ non indexé**, cette dernière sera **moins performante**. Il est ainsi important, pour les champs où l'on pressent des **recherches fréquentes**, d'ajouter un **index** pour y favoriser une recherche plus rapide. Il s'agit ici, à l'aide de l'assistant index, d'ajouter autant de nouveaux index qu'il y a de champs où vous désirez accélérer la recherche.

La ressource ci-dessous montre l'ajout d'un index sur le champ *commentaire* de la table *Dépenses.*

Note : Capsule vidéo accessible en ligne[1]

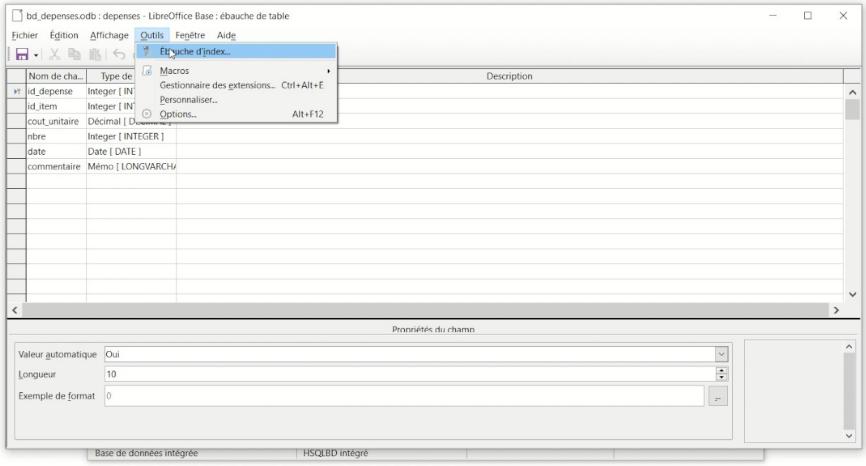
[1] http://cours.ebsi.umontreal.ca/sci6005/h2021/res/base_index_creation.mp4

Première étape : Ouverture de la table en **édition**



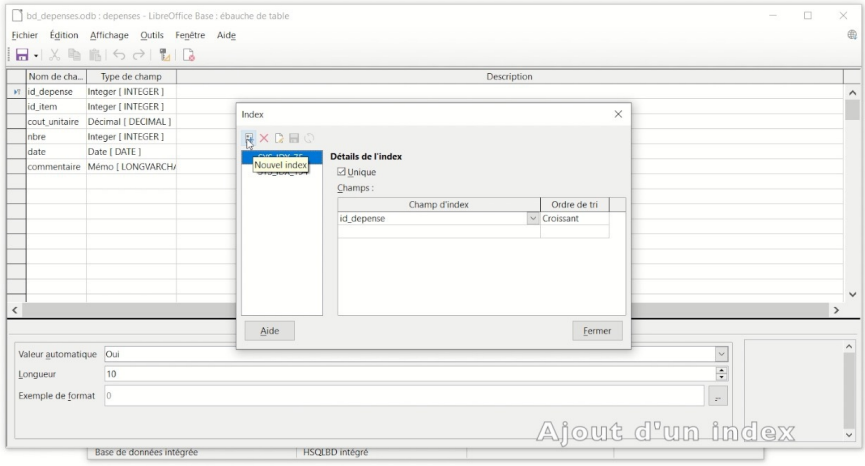
Ouverture de la table en mode édition

Deuxième étape : Ouverture de l'**assistant index**



Activation de l'assistant pour les index

Troisième étape : Création de l'index



Création de l'index sur le champ commentaire

5.4. Modèles textuel et relationnel : En résumé

Il est important de connaître les **caractéristiques des modèles textuel et relationnel** afin de mieux en comprendre l'*utilisation* et *faire des choix plus éclairés* vers l'un ou l'autre de ces modèles en fonction du contexte.

Résumé des principales caractéristiques des modèles textuel et relationnel

	SGBD textuel **Par ex. DB/TextWorks**	**SGBD relationnel** **Par ex. MySQL**
Nombre de table de données	une	une ou plus
Occurrences multiples dans un champ	oui	non Les occurrences multiples sont recréées, à la demande, par des requêtes reliant des tables
Possibilité d'avoir 0 occurrence dans un champ	oui	non La valeur NULL est utilisée pour "simuler" l'absence d'occurrence
Types de données	plus limités	plus riches et plus forts
Langage d'interrogation	plus limité globalement, mais présente plus d'opérateurs pour les données textuelles	plus riche (SQL) sauf pour certains aspects du traitement des données textuelles qui nécessitent de passer par la recherche avancée plein texte

Ce que l'on retient globalement :

- Dans les contextes où la **sensibilité textuelle est de mise**, c'est-à-dire que l'on veut être en mesure d'exploiter de la meilleure manière des données textuelles, le **modèle textuel** est à considérer.
- Dans les contextes où la sensibilité textuelle est moins primordiale et où la **réalité à représenter est plus complexe**, le **modèle relationnel** est à considérer.

5.5. Familles NoSQL

Le fait d'avoir mathématiquement et formellement défini le modèle relationnel avant toute implantation lui a permis de **"bien vieillir"**, comme il a ainsi gagné en *indépendance par rapport aux technologies*. Ceci dit, l'évolution entre autres du matériel informatique, des langages de programmation, des exigences des interfaces utilisateurs, des applications multimédias ainsi que de la réseautique a fait ressortir certaines **limitations** du modèle relationnel. D'autres types de bases de données ont **émergé** pour essayer de contourner ces limitations. C'est le cas entre autres des **familles NoSQL**. Certains logiciels de gestion électronique des documents ont adopté des bases de données NoSQL comme, par exemple, Constellio[1].

Nous n'entrerons pas dans un grand détail sur ces familles NoSQL, du fait de leur complexité. L'objectif est de réaliser que malgré toutes ses qualités, tout modèle de données possède des limites qui vont provoquer l'apparition d'autres modèles.

Le **modèle relationnel**, dans sa manière de représenter et de manipuler les données, se révèle **peu efficace** dans le contexte d'environnements Web distribués à **grande échelle** qui possèdent de **grands volumes** de données comme *Twitter*, *Facebook* et *eBay*. C'est entre autres un des facteurs à la source de l'émergence des familles de bases de données NoSQL.

Les **familles NoSQL** - l'appellation NoSQL (*Not only SQL*) date de 2009 - délaissent les propriétés des transactions relationnelles qui permettent de garantir et de maintenir la cohérence des données (propriétés *ACID*) au profit de contraintes qui priorisent la disponibilité des données (contraintes *BASE*) et qui se révèlent **plus adaptées aux grands environnements Web distribués**.

Il ne s'agit pas d'un modèle NoSQL, mais de **plusieurs familles NoSQL** qui se sont développées en parallèle et qui répondent à des besoins différents.

Exemples de familles NoSQL

[1] https://constellio.com/fr/accueil

Familles	Exemples d'utilisation
Orientées graphes pour traiter les réseaux massifs	Recommandations Twitter
Orientés colonnes pour faciliter les traitements privilégiant les colonnes	Calcul de l'âge moyen des utilisateurs
Orientées clé/valeur pour gagner en efficacité en lecture/ écriture et pour le changement d'échelle	Système de sauvegarde de type Dropbox
Orientée "document" pour supporter des structures variables	Gestion des métadonnées des produits vendus chez eBay

Ces nouvelles familles de bases de données présentent comme **principal avantage** de **contourner les limites du modèle relationnel** contraignantes pour le contexte de systèmes Web distribués à grande échelle (fortes performances, résistance au changement d'échelle, entre autres). Elles ne sont toutefois pas exemptes de limites, par exemple du fait de leur *relative jeunesse* et du *développement en parallèle de plusieurs familles*.

Avantages et désavantages des familles NoSQL

Avantages	Désavantages
• Faible coût relatif • Fortes performances • Résistance au changement d'échelle • Capacité à faire évoluer la représentation des données	• "Jeunesse" • Pas de langage de requête abstrait partagé • Travail de programmation spécifique plus important • Cohérence des données moins aisée à garantir

6. Ressources en lien avec le cours

Matériel de cours

- *Notes de cours (cf. sci6005_a21_tpbase)*

Protocole du TP Structuration dans une base de données[1]

Note : Du matériel complémentaire est précisé dans le protocole du *TP Structuration dans une base de données* en lien avec les manipulations dans *Base* (*LibreOffice*).

Pour en savoir plus

- DUFOUR, Christine. 2016. S4.1 Définition et caractéristiques des bases de données. In MOOC Architecture de l'information, Séquence 4 – Web et bases de données. https://archinfo00.hypotheses.org/231
- DUFOUR, Christine. 2016. S4.2 Bases de données sur le Web. In MOOC Architecture de l'information, Séquence 4 – Web et bases de données. https://archinfo00.hypotheses.org/233
- DUFOUR, Christine. 2016. S4.3 Bases de données relationnelles. In MOOC Architecture de l'information, Séquence 4 – Web et bases de données. https://archinfo00.hypotheses.org/235
- HABERT, Benoît. 2016. S4.4 Bases de données non relationnelles. In MOOC Architecture de l'information, Séquence 4 – Web et bases de données. https://archinfo00.hypotheses.org/237

[1] https://studium.umontreal.ca/course/view.php?id=188118&section=2#tp_bd [accès restreint]

Glossaire

Clé externe (ou clé étrangère)

Une **clé externe (ou clé étrangère)** est un champ dans une table qui pointe vers la clé primaire d'une autre table de données. Elle sert donc à lier des tables.

Imaginons par exemple une BD où on retrouve une table contenant des informations sur des personnes ainsi qu'une table contenant des informations sur différentes villes. On pourrait vouloir lier les deux tables afin d'associer à une personne des informations complémentaires au nom de sa ville de résidence actuelle (par exemple, le nombre d'habitants) sans avoir à consigner ces informations complémentaires dans la table PERSONNE (ce qui évitera bien de la redondance dans cette table, plusieurs personnes pouvant résider dans la même ville !). On pourrait retrouver dans la table PERSONNE un champ RESIDENCE_ACTUELLE qui pointerait vers le champ VILLE de la table GEOGRAPHIE.

Clé primaire

La **clé primaire** d'une table de données est un champ ou la combinaison de plusieurs champs qui permettent d'identifier uniquement chaque enregistrement de la table de données. Par exemple, si une table de données comporte des informations sur des personnes, une clé primaire potentielle pourrait être le numéro d'assurance sociale.

Index

Crédits des ressources

p. 4

http://creativecommons.org/licenses/publicdomain/4.0/fr/, johnny_automatic

p. 5

http://creativecommons.org/licenses/publicdomain/4.0/fr/, maoriveros