Bases de données – TD1
Modélisation, Optimisation de modèles, LDD

Afin d'aider les techniciens en informatique de CPE, il vous est demandé de créer une base de données permettant de gérer les machines virtuelles (VM) créées et leurs versions.

Remarque : Dans un souci de simplification, nous ne gérerons pas les salles dans lesquelles les VM sont déployées, ni les déploiements.

Une machine virtuelle a une plateforme de virtualisation dédiée (Microsoft Hyper-v, Virtualbox, Vmware, Microsoft VirtualPC, etc.) peu importe la version de celle-ci, un système d'exploitation, un nom, une description, un demandeur, une taille du disque dur virtuel (en Go), un type de stockage (taille fixe ou taille dynamiquement allouée) et une taille de mémoire RAM nécessaire (en Mo). Exemple :

N° VM	Nom VM	Logiciel de virtualisation	OS	Description	Demandeur	Taille DD (en Go)	Type stockage	Mémoire RAM (en Mo)
1	Visual Studio	VMWare	Windows 10 64 bits	Développement .NET 5IRC	V. Couturier	50	Dynamiquement alloué	4096
2	TPs BD	VirtualBox	Ubuntu 32 bits	TPs BD 4IRC	F. Perrin	20	Taille fixe	1024
Etc.								

Pour des questions de performance (et surtout d'un point de vue pédagogique !), il a été décidé de limiter la mémoire RAM des VM à 6144 Mo au maximum.

Un système d'exploitation a un type. Exemples :

Type	OS
Microsoft Windows	Windows 7 32 bits
Microsoft Windows	Windows 7 64 bits
Microsoft Windows	Windows 8 32 bits
Microsoft Windows	Windows 8 64 bits
Microsoft Windows	Windows 10 32 bits
Microsoft Windows	Windows 10 64 bits
…	
Linux	Ubuntu 32 bits
Linux	Ubuntu 64 bits
Linux	
…	
BSD	…

Pour chaque demandeur, on désire sauvegarder son nom, prénom, adresse, mail, tel. fixe auquel le contacter. Pour des soucis de simplification, seules les adresses françaises sont enregistrées. L'adresse n'est pas forcément saisie.

Il n'est pas possible de changer la version d'OS. Par exemple, une « mise à jour » majeure de Windows 8 vers Windows 10 correspond à la création d'une nouvelle VM, même si l'on part d'une existante pour la réaliser.

La machine virtuelle a une ou plusieurs versions dépendant des logiciels qui y sont installés. Un logiciel est un programme dans sa version. Par exemple, PostgreSQL 9 et PostgreSQL 10 seront considérés comme 2 logiciels différents.
Ainsi, une première version de la VM nommée « Visual Studio » a été créée par exemple le 01/09/2016 contenant Microsoft Visual Studio 2015 et Microsoft SQL Server 2016 (Cf. tableau suivant). Une mise à jour – version n°2 de la VM – a été réalisée le 20/09/2017. Cette version a consisté à passer Visual Studio en version 2017 et à ajouter PostgreSQL 10.
Le numéro de version de la machine virtuelle dépend également du numéro de la VM.

N° VM	Nom VM	N° version VM	Date version	Logiciels installés
1	Visual Studio	1	01/09/2016	- Microsoft Visual Studio 2015 - Microsoft SQL Server 2016
		2	20/09/2017	- Microsoft Visual Studio 2017 - Microsoft SQL Server 2016 - PostgreSQL 10
		3	20/09/2018	- Microsoft Visual Studio 2017 - Microsoft SQL Server 2017 - PostgreSQL 10
Etc.				
10	SGBD Oracle	1	20/08/2014	- Oracle Database 11g R1
		2	15/08/2018	- Oracle Database 12c R2

Pour répondre aux questions, utilisez les annexes et/ou la doc PostgreSQL (http://docs.postgresqlfr.org/)**.**

Configurer PostgreSQL comme indiqué en annexe 1.

Q1 :

a. A partir de l'exemple fourni dans le document « Rappel Elaboration MCD », réaliser le MCD.

b. Après discussion avec les demandeurs, les besoins suivants ont été oubliés :

- La gestion des demandeurs est plus complexe qu'il n'y paraît. En effet, il y a 2 types de demandeurs : vacataire et permanent. L'adresse et le téléphone fixe ne sont saisis que pour les vacataires. On souhaite également enregistrer le nom de leur entreprise. Pour les permanents, on souhaite enregistrer le numéro de bureau. Proposer une modification au modèle.

- Un logiciel n'est compatible que sur certains OS. On souhaite gérer cette compatibilité afin de ne pas saisir d'erreur dans les logiciels installés. Exemple :

Nom logiciel	Noms OS
PowerAMC 15	Windows XP, Windows 7, Windows 8, Windows 10
PowerAMC 16	Windows XP, Windows 7, Windows 8, Windows 10
Microsoft Office 2013	Windows 7, Windows 8, Windows 10
Microsoft Office 2011	Mac OS X
Microsoft Office 365	Windows 7, Windows 8, Windows 10, Mac OS X

c. Réaliser le MLD relationnel.

d. Réaliser le MPD. Appliquer les règles de nommage présentées ici (pages 10 à 12) : http://www.sqlspot.com/sites/sqlspot.com/IMG/pdf/Norme_de_developpement_BD.pdf.
Quelques règles simples d'optimisation à appliquer pour créer le MPD :
- Conservation ou non des tables de référence, i.e. tables en général ayant peu de données et dont les celles-ci évolueront peu ou pas ? Si le contenu de celles-ci n'évolue pas et n'évoluera jamais dans le futur, il n'est pas nécessaire de les conserver. Les tables de référence qui n'évolueront pas seront remplacées par des contraintes CHECK afin de ne pas pouvoir saisir n'importe quoi et dans l'application cliente par des listes déroulantes dont les éléments sont fixes.
- Limitation des clés primaires composées à 2 champs au maximum (sinon fichier d'index imposant en mémoire). Comment faire alors pour assurer l'unicité des champs composant la clé primaire si ceux-ci ne le sont plus : en utilisant des contraintes de clé unique.

Une fois PostgreSQL démarré, créer la base « VM » (en étant positionné sur ligne « Bases de Données » puis clic droit).

Q2.

A partir du fichier « Doc complémentaire - Enoncé TD1 », compléter le script de création des tables « script partiel Virtualisation.sql ». Pour les types de données, se référer au chapitre 8 de la documentation ou au cours). Ce script doit également créer les contraintes d'intégrité adéquates (clés primaires, contraintes de validation, contraintes d'intégrité référentielle, contraintes d'unicité).
Le script créé devra contenir trois parties :

1. Suppression des tables. Instruction `DROP TABLE` (Cf. documentation) en utilisant l'option `IF EXISTS`

et éventuellement CASCADE.
2. Création des tables (CREATE TABLE). Vous créerez également dans cette partie les contraintes de clé primaire, uniques et de validation (CHECK). Vous nommerez l'ensemble des contraintes de façon appropriée (**Cf. annexe 2**). Vous pourrez exécuter au fur et à mesure votre script.
3. Création des contraintes d'intégrité référentielle (clés étrangères). Instruction ALTER TABLE… ADD CONSTRAINT…

Exécuter ce script plusieurs fois dans le schéma Public pour voir s'il fonctionne sans erreur : dans une fenêtre SQL, ouvrir le fichier et exécuter la requête.

Q3 : Schéma d'informations
Le schéma d'informations (information_schema) (Cf. chapitre 34 de la documentation) consiste en un ensemble de vues système contenant des informations sur les objets définis dans une base de données. Les vues du schéma d'information ne contiennent pas d'information sur les fonctionnalités spécifiques à PostgreSQL; pour cela, vous devez utiliser les catalogues système (pg_catalog) ou d'autres vues spécifiques à PostgreSQL.

Visualisez les données stockées dans les vues du schéma d'informations (information_schema) de la base postgres (se positionner sur la base « Postgres, Catalogues, ANSI (Information_schema, Objets catalogue » et sélectionner le bon objet puis, bouton « Afficher les données de l'objet selectionné » ou écrire la requête SQL). Vues système à utiliser : tables, columns, table_constraints, constraint_column_usage, constraint_table_usage, check_constraints, key_column_usage, referential_constraints.
Ex. :
```
select * from information_schema.tables where table_schema = 'public';  -- Il est nécessaire de préfixer par le nom du schéma, sinon la recherche se fait dans le schéma actuel (i.e. public)
select * from information_schema.constraint_column_usage;
```

A quoi sert la vue système table_constraints ? Expliquer son contenu.

Expliquez la différence entre les vues système constraint_column_usage et constraint_table_usage.

A quoi sert la vue système key_column_usage ? Expliquer son contenu.

Remarque : il est toujours nécessaire de bien connaître les vues et/ou tables système quand on utilise un SGBD. Celles-ci sont spécifiques au SGBD utilisé (Oracle, MySQL,…).

Q4 : Catalogue système
En plus du schéma public et de ceux créés par les utilisateurs, chaque base de données contient un schéma pg_catalog. Celui-ci contient les tables systèmes et tous les types de données, fonctions et opérateurs intégrés.

Visualisez les données du catalogue système (se positionner sur la base « Postgres, Catalogues, PostgreSQL(pg_catalog)» et sélectionner le bon objet puis, bouton « Afficher les données de l'objet selectionné » ou écrire la requête SQL) : vue pg_tables, table pg_constraint
Ex. : select * from pg_tables;
Expliquer le contenu de la table pg_constraint.

Q5 : Séquence
Définissez une séquence (<=> numéro automatique) (**Cf. Annexe 3**), nommée SEQ_DEM, afin de faciliter la mise en place d'un numéro s'incrémentant pour chaque demandeur. Cette séquence doit

commencer avec la valeur 1 et elle possède un pas d'incrément de 1. Elle sera associée à la colonne clé primaire (`DEM_ID`) de la table `DEMANDEUR`. Après création, visualisez les données de la vue système `sequences` du schéma d'informations `information_schema`.
NB : l'essai de la séquence sera réalisé lors d'insertion de données (Q11).

Q6 : Contrainte d'unicité
Les vacataires sont nombreux (plus de 400). En outre, ils changent souvent et de nouveaux sont recrutés. Il n'est pas souhaitable d'avoir des informations en double dans la base. Aussi il ne doit pas être possible d'avoir deux vacataires qui possèdent mêmes nom, prénom et numéro de téléphone fixe. Définissez une contrainte d'intégrité afin de satisfaire cette nouvelle exigence. La contrainte sera ajoutée sur la table des demandeurs (instruction `alter table`). Visualisez les données des vues système `constraint_column_usage`, `constraint_table_usage` et `key_column_usage` avant et après l'ajout des contraintes.
Remarque : cette contrainte fonctionnera aussi pour les permanents.

Q7 : Contrainte de validation
Certains demandeurs souhaitent en outre que leur numéro de portable soit enregistré. Or la base ne nous permet de stocker qu'un seul numéro de téléphone. Apportez les modifications de structure nécessaires pour prendre en compte cette modification.
Comme cette nouvelle colonne, nommée `DEM_MOBILE`, va contenir des informations relatives à un numéro de téléphone portable, mettez en place une contrainte d'intégrité afin de vous assurer que le numéro de téléphone saisi commence bien par `06` ou par `07` ou bien n'est pas rempli (contrainte de type `check`. Utilisez des fonctions de manipulation de chaines et notamment la commande `like`). Visualisez les données des vues système `constraint_column_usage`, `constraint_table_usage` et `key_column_usage`.

Q8 : Index
Afin d'améliorer les performances d'accès aux données, définissez un index sur toutes les colonnes de type clé étrangère (ou groupes de colonnes si FK composée de plusieurs champs). Les opérations de jointure seront ainsi plus rapides ; nous y reviendrons lors d'un prochain TP sur l'optimisation.
Remarque : il y a 9 FK à créer, autant que de flèches reliant les tables dans le MPD.

Q9 : Modifiez la table des demandeurs afin que la colonne `DEM_TYPE` prenne par défaut la valeur "Permanent".

Q10 : Utilisation d'une séquence
Ajoutez les données suivantes dans la table `DEMANDEUR` (attention, tous les champs ne sont pas saisis. Voir les champs de la table `DEMANDEUR` sous PostrgeSQL). **Vous utiliserez la séquence créée en Q5, nommée `SEQ_DEM` pour générer le numéro du demandeur.**

N°	Nom	Prénom	Email	Type	Rue	CP	Ville
1	ALBERT	Anne	anne.albert@gmail.com	Vacataire	13 rue de la Gare	69002	Lyon
2	BERNAUD	Barnabé	barnabe.bernaud@cpe.fr	Permanent			

TelFixe	Entreprise	NumBureau
0450505055	EDF	
		G102

Remarque : pour l'utilisation d'une séquence, cf. Annexe 3 (située en fin de sujet).

Vous vérifierez que les nouveaux demandeurs ont bien pour numéro les valeurs 1 et 2. Sinon, supprimez et recréez la séquence et relancez les deux INSERT précédents.

```
Delete from T_E_DEMANDEUR_DEM;
DROP SEQUENCE SEQ_DEM;
CREATE SEQUENCE SEQ_DEM;
INSERT…
```

Q11 :
Exécutez le script `insertion virtualisation.sql` permettant d'insérer des données dans les tables.

Remarque :

- *Pour utiliser la valeur par défaut d'une colonne lors de l'insertion, il est possible d'utiliser le mot clé DEFAULT (Cf. insertions dans la table `VERSIONVM`)*
- *Certaines dates sont dynamiques et dépendent de la date du jour (`current_date`)*

A la fin des insertions vous vérifierez que les tables suivantes ont bien le nombre de tuples prévu : COMPATIBILITE (1613), INSTALLATION (87). Il doit également y avoir 52 tuples dans la table DEMANDEUR avec l'IDdemandeur variant de 1 à 52.

Q12 :
Calculer le nombre de VM par demandeur. Afficher également les demandeurs pour lesquels les VM n'ont pas encore été saisies. Utiliser une jointure externe (`LEFT JOIN` ou `RIGHT JOIN`).

	Nom demandeur character varying(50)	Prénom demandeur character varying(50)	nb vm bigint
1	ALBERT	Anne	20
2	BALMAND	Antoine	4
3	BALMAT	David	2
4	BALTHASSAT	Florian	1
5	BARNOUIN	Cyril	0
6	BARONNAT	Mael	1
7	BASSET	Laura	1
8	BEAU	Maxime	1
9	BEAUREPAIRE	Clency	0
10	BELAID	Theo	1
11	BEN STA	Vincent	0
12	BERGER	Gregoire	1
13	BERKATI	Candice	1
14	BERNARD	Damien	0
15	BERNARD	Kentin	0
16	BERNAUD	Barnabé	4
17	BERTHET	Antoine	1
18	BERTHET	Nicolas	2
19	BERTHON	Jason	0
20	BESSE	Stefan	1
21	BETEMPS	Guillaume	0
22	BIENFAIT	Jules	1
23	BILLARD	Maxime	0
24	BINAULD	Benjamin	2
25	BISCHOF	Bastien	0

(52 lignes)

Question 13 :
Calculer le nombre de VM par OS et par type d'OS.

	Type OS character varying(30)	OS character varying(30)	nb VMs bigint
1	BSD	FreeBSD 32 bits	2
2	BSD	NetBSD 32 bits	2
3	BSD	OpenBSD 32 bits	2
4	Linux	Debian 32 bits	2
5	Linux	Fedora 32 bits	2
6	Linux	Mandriva 32 bits	2
7	Linux	OpenSUSE 32 bits	2
8	Linux	Oracle 32 bits	2
9	Linux	Red Hat 32 bits	2
10	Linux	Ubuntu 32 bits	2
11	Linux	Ubuntu 64 bits	2
12	Mac OS/X	Mac OS X 10.5 32 bits	2
13	Microsoft Windows	Windows 10 32 bits	1
14	Microsoft Windows	Windows 10 64 bits	2
15	Microsoft Windows	Windows 2000	1
16	Microsoft Windows	Windows 2003 32 bits	1
17	Microsoft Windows	Windows 2003 64 bits	1
18	Microsoft Windows	Windows 2008 32 bits	1
19	Microsoft Windows	Windows 2008 64 bits	1
20	Microsoft Windows	Windows 7 32 bits	6
21	Microsoft Windows	Windows 7 64 bits	2
22	Microsoft Windows	Windows 8.1 32 bits	1
23	Microsoft Windows	Windows 8.1 64 bits	2
24	Microsoft Windows	Windows 95	1
25	Microsoft Windows	Windows 98	1
26	Microsoft Windows	Windows Me	1
27	Microsoft Windows	Windows NT4	1
28	Microsoft Windows	Windows Vista 32 bits	1
29	Microsoft Windows	Windows Vista 64 bits	1
30	Microsoft Windows	Windows XP 32 bits	8
31	Microsoft Windows	Windows XP 64 bits	1
32	Solaris	Solaris 10 32 bits	2

Question 14 :
Créer la vue `v_derversionvm` affichant la dernière version de chaque VM (et logiquement installée sur les postes de CPE). Vous vous limiterez ensuite à afficher celles réalisées cette année (année en cours). La vue devra toujours fonctionner l'an prochain, sans avoir à modifier le code SQL !

	ID VM numeric (4)	Nom VM character varying (50)	description character varying (500)	date der version date
1	24	TPs systeme	TPs systeme	2019-02-05
2	57	CAO	CAO	2019-02-05

Question 15 :
Ecrire une requête qui affiche un message en fonction du nombre d'installations de logiciels sur chaque version de VM.

Nombre d'installations	**Message**
0	OS nu
Entre 1 et 2	Peu
Entre 3 et 5	Normal
6 et plus	Beaucoup de travail !

Indications : utilisez l'instruction `CASE`

	ID VM numeric(4,0)	Num version numeric(2,0)	Nb logiciels bigint	qualification text
1	1	1	2	Peu
2	2	1	2	Peu
3	3	1	2	Peu
4	4	1	1	Peu
5	5	1	1	Peu
6	6	1	1	Peu
7	7	1	1	Peu
8	8	1	1	Peu
9	9	1	1	Peu
10	10	1	2	Peu
11	11	1	1	Peu
12	12	1	1	Peu
13	13	1	1	Peu
14	14	1	2	Peu
15	15	1	1	Peu
16	15	2	1	Peu
17	16	1	1	Peu
18	17	1	2	Peu
19	18	1	2	Peu
20	18	2	2	Peu
21	19	1	2	Peu
22	19	2	1	Peu
23	20	1	3	Normal
24	21	1	4	Normal
25	21	2	6	Beaucoup de travail !
26	21	3	8	Beaucoup de travail !
27	22	1	5	Normal
28	23	1	1	Peu
29	24	1	0	OS nu
30	25	1	0	OS nu
31	26	1	0	OS nu
32	27	1	0	OS nu
33	28	1	0	OS nu
34	29	1	0	OS nu
35	30	1	0	OS nu

(68 lignes)

Question 16 : création d'une table en utilisant l'instruction `CREATE TABLE <nom_table> AS SELECT…` Créer une nouvelle table nommée `INFOSVM` à partir des tables `VM` et `DEMANDEUR`.

Données de la table `INFOSVM` après création :

	mav_id numeric(4,0)	mav_nom character varying(50)	mav_description character varying(500)	mav_tailledd numeric(6,2)	mav_ram numeric(4,0)	mav_typestockage character varying(30)	dem_nom character varying(50)	dem_prenom character varying(50)
1	1	TPs reseaux Windows	TPs reseaux Windows	20.00	1024	Dynamiquement alloué	ALBERT	Anne
2	2	TPs reseaux Windows	TPs reseaux Windows	20.00	1024	Dynamiquement alloué	ALBERT	Anne
3	3	TPs reseaux Windows	TPs reseaux Windows	20.00	1024	Dynamiquement alloué	ALBERT	Anne
4	4	TPs reseaux Windows	TPs reseaux Windows	20.00	1024	Dynamiquement alloué	ALBERT	Anne
5	5	TPs reseaux Windows	TPs reseaux Windows	20.00	1024	Dynamiquement alloué	ALBERT	Anne
6	6	TPs reseaux Windows	TPs reseaux Windows	20.00	1024	Dynamiquement alloué	ALBERT	Anne
7	7	TPs reseaux Windows	TPs reseaux Windows	20.00	1024	Dynamiquement alloué	ALBERT	Anne
8	8	TPs reseaux Windows	TPs reseaux Windows	20.00	2048	Dynamiquement alloué	ALBERT	Anne
9	9	TPs reseaux Windows	TPs reseaux Windows	20.00	2048	Dynamiquement alloué	ALBERT	Anne
10	10	TPs reseaux Windows	TPs reseaux Windows	20.00	2048	Dynamiquement alloué	ALBERT	Anne
11	11	TPs reseaux Windows	TPs reseaux Windows	20.00	2048	Dynamiquement alloué	ALBERT	Anne
12	12	TPs reseaux Windows	TPs reseaux Windows	20.00	2048	Dynamiquement alloué	ALBERT	Anne
13	13	TPs reseaux Windows	TPs reseaux Windows	20.00	4096	Dynamiquement alloué	ALBERT	Anne
14	14	TPs reseaux Windows	TPs reseaux Windows	20.00	4096	Dynamiquement alloué	ALBERT	Anne
15	15	TPs reseaux Windows	TPs reseaux Windows	20.00	4096	Dynamiquement alloué	ALBERT	Anne
16	16	TPs reseaux Windows	TPs reseaux Windows	20.00	4096	Dynamiquement alloué	ALBERT	Anne
17	17	TPs reseaux Windows	TPs reseaux Windows	20.00	4096	Dynamiquement alloué	ALBERT	Anne
18	18	TPs reseaux Windows	TPs reseaux Windows	20.00	4096	Dynamiquement alloué	ALBERT	Anne
19	19	TPs reseaux Windows	TPs reseaux Windows	20.00	4096	Dynamiquement alloué	ALBERT	Anne
20	20	VS 2010	TP .NET	50.00	4096	Dynamiquement alloué	BERNAUD	Barnabé
21	21	VS 2013	TP.NET	80.00	4096	Dynamiquement alloué	BERNAUD	Barnabé
22	22	VS 2017	TP.NET	90.00	4096	Dynamiquement alloué	BERNAUD	Barnabé
23	23	VS 2017	TP C#	80.00	2048	Dynamiquement alloué	BALMAND	Antoine
24	24	TPs systeme	TPs systeme	20.00	2048	Dynamiquement alloué	CUNY	Igor
25	25	TPs systeme	TPs systeme	20.00	2048	Dynamiquement alloué	BERTHET	Antoine

(60 lignes)

Visualisez les contraintes dans la vue système `table_constraints`. Y a-t-il des contraintes sur la table `INFOSVM` ?

Question 17 : création d'une table temporaire (cf. annexe 4)
On souhaite obtenir le nombre d'installations de logiciels par version de VM et ce même nombre par VM (cf. écrans suivants). Seules les données des VM sur lesquelles des logiciels ont été installés seront affichées (pas de jointure externe).
Indications :
Première étape : créer une table temporaire globale, dans laquelle les informations vont être ajoutées au fur et à mesure de leur calcul. Utilisez la clause `ON COMMIT PRESERVE ROWS` *lors de la création de la table temporaire globale.*

2ème étape : Ajouter les informations relatives à chaque version (insert)

	idvm numeric(4,0)	nomvm character varying(50)	numversion numeric(2,0)	nbinstallationsversion numeric(10,0)	nbinstallationsvm numeric(10,0)
1	1	TPs reseaux Windows	1	2	
2	2	TPs reseaux Windows	1	2	
3	3	TPs reseaux Windows	1	2	
4	4	TPs reseaux Windows	1	1	
5	5	TPs reseaux Windows	1	1	
6	6	TPs reseaux Windows	1	1	
7	7	TPs reseaux Windows	1	1	
8	8	TPs reseaux Windows	1	1	
9	9	TPs reseaux Windows	1	1	
10	10	TPs reseaux Windows	1	2	
11	11	TPs reseaux Windows	1	1	
12	12	TPs reseaux Windows	1	1	
13	13	TPs reseaux Windows	1	1	
14	14	TPs reseaux Windows	1	2	
15	15	TPs reseaux Windows	1	1	
16	15	TPs reseaux Windows	2	1	
17	16	TPs reseaux Windows	1	1	
18	17	TPs reseaux Windows	1	2	
19	18	TPs reseaux Windows	1	2	
20	18	TPs reseaux Windows	2	2	
21	19	TPs reseaux Windows	1	2	
22	19	TPs reseaux Windows	2	1	
23	20	VS 2010	1	3	
24	21	VS 2013	1	4	
25	21	VS 2013	2	6	
26	21	VS 2013	3	8	
27	22	VS 2017	1	5	
28	23	VS 2017	1	1	
29	48	TPs bureautique	1	1	
30	49	TPs bureautique	1	1	
31	50	TPs bureautique	1	1	
32	51	TPs bureautique	1	1	
33	52	TPs bureautique	1	1	
34	53	TPs bureautique	1	1	
35	54	TPs bureautique	1	1	
36	55	Java 7	1	3	
37	56	Java 8	1	4	
38	57	CAO	1	1	
39	57	CAO	2	1	
40	57	CAO	3	1	
41	58	TP BI	1	9	
42	59	Gestion de projet	1	1	

(44 lignes)

3ème étape : Ajouter les informations relatives à chaque VM (update)

	idvm numeric(4,0)	nomvm character varying(50)	numversion numeric(2,0)	nbinstallationsversion numeric(10,0)	nbinstallationsvm numeric(10,0)
1	1	TPs reseaux Windows	1	2	2
2	2	TPs reseaux Windows	1	2	2
3	3	TPs reseaux Windows	1	2	2
4	4	TPs reseaux Windows	1	1	1
5	5	TPs reseaux Windows	1	1	1
6	6	TPs reseaux Windows	1	1	1
7	7	TPs reseaux Windows	1	1	1
8	8	TPs reseaux Windows	1	1	1
9	9	TPs reseaux Windows	1	1	1
10	10	TPs reseaux Windows	1	2	2
11	11	TPs reseaux Windows	1	1	1
12	12	TPs reseaux Windows	1	1	1
13	13	TPs reseaux Windows	1	1	1
14	14	TPs reseaux Windows	1	2	2
15	15	TPs reseaux Windows	1	1	2
16	15	TPs reseaux Windows	2	1	2
17	16	TPs reseaux Windows	1	1	1
18	17	TPs reseaux Windows	1	2	2
19	18	TPs reseaux Windows	1	2	4
20	18	TPs reseaux Windows	2	2	4
21	19	TPs reseaux Windows	1	2	3
22	19	TPs reseaux Windows	2	1	3
23	20	VS 2010	1	3	3
24	21	VS 2013	1	4	18
25	21	VS 2013	2	6	18
26	21	VS 2013	3	8	18
27	22	VS 2017	1	5	5
28	23	VS 2017	1	1	1
29	48	TPs bureautique	1	1	1
30	49	TPs bureautique	1	1	1
31	50	TPs bureautique	1	1	1
32	51	TPs bureautique	1	1	1
33	52	TPs bureautique	1	1	1
34	53	TPs bureautique	1	1	1
35	54	TPs bureautique	1	1	1
36	55	Java 7	1	3	3
37	56	Java 8	1	4	4
38	57	CAO	1	1	3
39	57	CAO	2	1	3
40	57	CAO	3	1	3
41	58	TP BI	1	9	9
42	59	Gestion de projet	1	1	2

(44 lignes)

Visualiser les données de la table temporaire.
Supprimer ensuite la table.

Question 18 :
Etablir la liste des VM modifiées plus de 2 fois (compris) au cours des 36 derniers mois.

	ID VM numeric(4,0)	nom character varying(50)
1	18	TPs reseaux Windows
2	15	TPs reseaux Windows
3	19	TPs reseaux Windows

Question 19 :
Établir la liste des logiciels qui n'ont pas été installés au cours des 70 derniers mois.

	ID Logiciel numeric (5)	Nom character varying (50)
1	14	Oracle Database 10g
2	16	Microsoft SQLServer 2010
3	18	Microsoft SQLServer 2014
4	20	Client Microsoft SQLServe...
5	22	Client Microsoft SQLServe...
6	24	PostgreSQL 9
7	27	Odoo 8
8	28	Odoo 10
9	29	PowerAMC15
10	31	BonitaBPM 6
11	32	BonitaBPM 7
12	36	Microsoft Visio 2010
13	37	Microsoft Visio 2013
14	38	Microsoft Visio 2016
15	39	PentahoBI 5
16	41	Windev 19
17	42	Windev 22
18	44	Webdev 22
19	45	Matlab R2012
20	46	Matlab R2015
21	47	Matlab R2016
22	48	Eclipse Juno 4.2
23	49	Eclipse Kepler 4.3
24	55	Wireframesketcher 4

Question 20 :
Afficher les logiciels qui fonctionnent (sont compatibles) sur tous les OS de 2 manières (avec `NOT EXISTS` puis `COUNT`).

	ID Logiciel numeric(5,0)	Nom character varying(50)
1	6	LibreOffice 5
2	7	OpenOffice 5
3	27	Odoo 8
4	28	Odoo 10
5	48	Eclipse Juno 4.2
6	49	Eclipse Kepler 4.3
7	50	Eclipse Luna 4.4
8	51	Eclipse Oxygen 4.7
9	52	NetBeans java SE 7.0
10	53	NetBeans java SE 8.0

Question 21 :
Afficher les logiciels qui fonctionnent sur tous les OS de type « Mac OS/X » de 2 manières (avec `NOT EXISTS` puis `COUNT`).

	ID Logiciel numeric(5,0)	Nom character varying(50)
1	3	Microsoft Office 2011
2	5	Microsoft Office 365
3	6	LibreOffice 5
4	7	OpenOffice 5
5	24	PostgreSQL 9
6	25	PostgreSQL 10
7	26	MySQL 5
8	27	Odoo 8
9	28	Odoo 10
10	31	BonitaBPM 6
11	32	BonitaBPM 7
12	39	PentahoBI 5
13	40	PentahoBI 6
14	48	Eclipse Juno 4.2
15	49	Eclipse Kepler 4.3
16	50	Eclipse Luna 4.4
17	51	Eclipse Oxygen 4.7
18	52	NetBeans java SE 7.0
19	53	NetBeans java SE 8.0
20	56	Autocad 2014
21	57	Autocad 2015
22	58	Autocad 2016

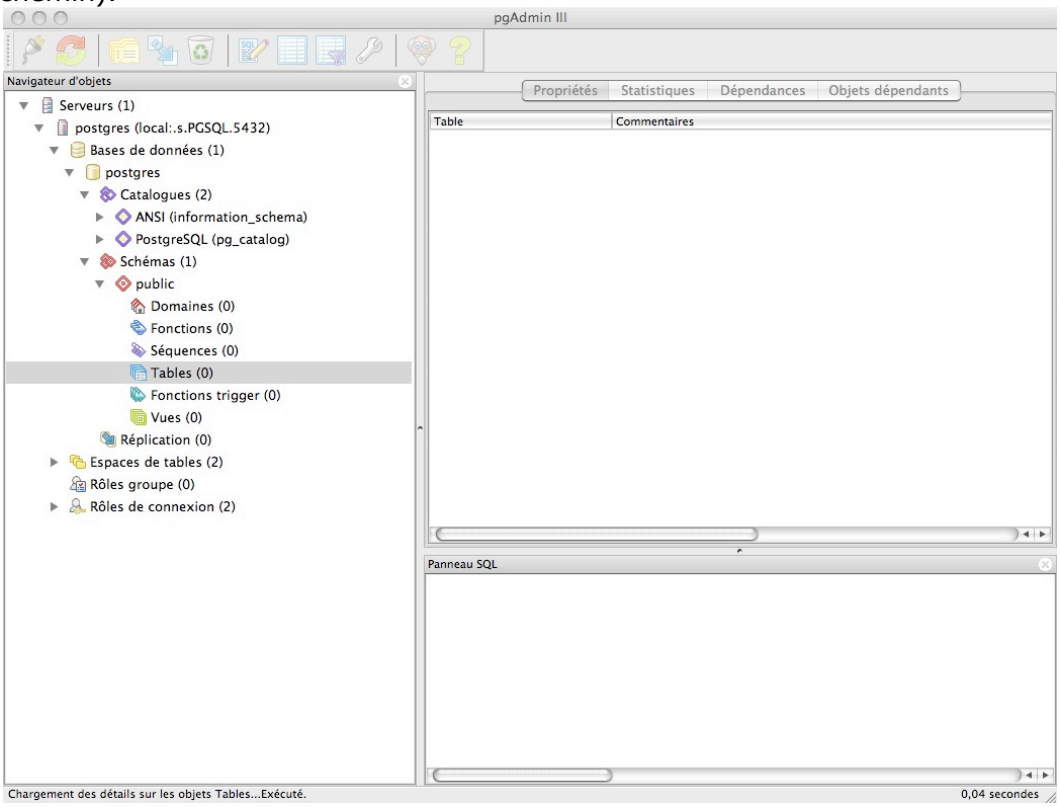
ANNEXES

1. Connexion et configuration de PostgreSQL

Un serveur postgresql est accessible à partir des postes des salles TP.
70 bases existent à s'affecter par binôme. Les noms de bases, utilisateurs, mots de passe sont sous la forme binomeXX avec XX variant de 01 à 70.

1. Démarrez une session linux.
2. Lancez l'application pgadmin3.
3. Dans pgadmin3, connectez-vous à PostgreSQL (bouton **/**) :
   - Dans la fenêtre « Ajouter un enregistrement de Serveur », saisissez :
     - Nom : `binomeXX`
     - Hôte : `db-tp.cpe.fr`
     - Port TCP : `5432`
     - Base maintenance : `binomeXX`
     - Nom utilisateur : `binomeXX` (à changer : `postgres` par défaut)
     - Mot de passe : `binomeXX`

   Comme il s'agit d'une ancienne version de PgAdmin3, cliquer 70 fois sur l'erreur indiquant que cette version n'est pas compatible avec celle du serveur.
4. Si vous dépliez le menu « bases de données » dans le navigateur d'objet, la liste des BD apparait et vous n'avez accès qu'à la vôtre (Cf. écran suivant (BD `binomeXX` et non `postgres` sur votre écran) – pensez à déplier les items).
5. Cliquez sur le bouton « SQL » pour ouvrir une fenêtre de saisie d'une requête. Il est alors possible de saisir une requête dans la fenêtre ou d'ouvrir un script existant (menu `Fichier/ouvrir/choisir` le bon chemin).



2. Création d'une table

La création consiste à définir (en fonction de l'analyse) le nom des colonnes, leur type, une valeur par défaut à la création de la ligne (default), les règles de gestion s'appliquant à la colonne (`CONSTRAINT`). Si une règle de gestion concerne plusieurs colonnes de la ligne, on définit une contrainte de table.

Syntaxe

```
CREATE TABLE nom (nomcolonne type [DEFAULT expr]
[[CONSTRAINT nom contrainte-de-colonne...],...
[,CONSTRAINT nom contrainte-de-table...]);
```

Dénomination des contraintes

Les contraintes peuvent être nommées afin d'être plus facilement manipulées ultérieurement (activation, suppression). Dans le cas où aucun nom n'est affecté explicitement à une contrainte, PostgreSQL génère automatiquement un nom (`xxx_fkey` ou `xxx_pkey`,…).

Lors de l'affectation explicite d'un nom à une contrainte, on utilise en général la dénomination suivante :

`Table_Colonne_TypeDeContrainte` OU `TypeDeContrainte_Table_Colonne`

`Table`	Nom de la table sur laquelle est définie la contrainte.	
`Colonne`	Nom de la (ou des) colonne(s) sur laquelle est définie la contrainte.	
`TypeDeContrainte` :	`PK`	Clé primaire
	`UQ`	Unique
	`CK`	Check

3. Les séquences

La création d'un objet `SEQUENCE` met à disposition de l'utilisateur un générateur de nombres. Les séquences sont utilisées pour générer des numérotations automatiques, en particulier pour la création de valeurs de clé primaire. Son utilisation est plus souple et donne de meilleures performances que la gestion manuelle des compteurs par l'intermédiaire d'une table. Elle ne garantit pas cependant l'absence de "trous" dans la numérotation. La séquence est un simple générateur de numéros et tous les numéros sont différents mais si des numéros sont demandés à une séquence et ne sont pas utilisés par la suite, alors ces numéros sont perdus. La séquence est en effet un objet à part entière et peut être utilisée par plusieurs tables.
Chaque valeur séquence s'exprime au maximum sur 28 chiffres significatifs.
Il est possible, sous PostgreSQL, de remplacer l'utilisation d'une séquence par un type de données `SERIAL` associé au champ. A la différence d'un type `SERIAL`, une séquence n'est pas forcément associée à un champ d'une table et doit être appelé lors de chaque insertion d'enregistrement.

Syntaxe

```
CREATE SEQUENCE nom [paramètres];
ALTER SEQUENCE nom paramètres ;
DROP SEQUENCE nom ;
```

Paramètres

`START WITH n`
Valeur initiale.

`INCREMENT BY n`
Pas d'incrémentation. Il peut être positif ou négatif.

`MINVALUE n/NO MINVALUE`
Valeur limite minimum ou non.

`MAXVALUE n/NO MAXVALUE`
Valeur limite maximum ou non.

`CYCLE/NO CYCLE`
`CYCLE` force la séquence à repasser à `MINVALUE` lorsque `MAXVALUE` a été atteinte (séquence croissante) ou à `MAXVALUE` lorsque `MINVALUE` a été atteinte (séquence décroissante).

`CACHE`
Force l'anticipation de la génération des valeurs suivantes de la séquence en mémoire. A pour effet d'améliorer les temps de réponse de la séquence.

`OWNED BY table.colonne, OWNED BY NONE`
Permet d'associer la séquence à une colonne de table spécifique. De cette façon, la séquence sera automatiquement supprimée si la colonne (ou la table entière) est supprimée. La table indiquée doit être dans le même schéma que la séquence. `OWNED BY NONE`, valeur par défaut, indique qu'il n'y a pas d'association.

L'utilisation se fait de la façon suivante :

- Créer une séquence ascendante appelée `serie`, démarrant à 101 :

```
CREATE SEQUENCE serie START 101;
```

- Sélectionner le prochain numéro de cette séquence :
```
SELECT nextval('serie');
```
- Utiliser cette séquence dans une commande `INSERT` :
```
INSERT INTO distributors VALUES (nextval('serie'), 'nothing');
```

4. Les tables temporaires

PostgreSQL offre la possibilité de créer des tables temporaires pour stocker des informations le temps d'une session ou d'une transaction. La création d'une table temporaire est notamment très utile lors du développement de requêtes complexes et permet de procéder par étape. Les données stockées dans une table temporaire créée avec l'ordre `CREATE TEMPORARY TABLE` sont accessibles uniquement depuis la session qui a créé les données. En fait chaque session ne voit que ses propres données.

La spécificité d'une table temporaire réside dans le fait que les données insérées dans cette table ne restent présentes que le temps de la transaction. La table, quant à elle, reste bien sûr en place et il est possible de l'utiliser dans les autres transactions. Si l'on souhaite que les données insérées dans cette table soient stockées de façon plus persistantes, il faut utiliser la clause `ON COMMIT PRESERVE ROWS` lors de la création de la table temporaire. Ainsi, les informations insérées dans cette table seront visibles à travers toutes les transactions pendant la durée de la session de l'utilisateur PostgreSQL qui a inséré les informations dans la table.

Syntaxe
```
CREATE TEMPORARY TABLE table (nom colonne type, ...) [ON COMMIT PRESERVE ROWS];
```