Cours 4:Jointure et Multi Table

Exploitation des données

Afficher des données de plusieurs tables

Parfois, vous avez besoin d'utiliser des données de plus d'une seule table.
Dans l'exemple suivant, le rapport affiche des données de deux tables différentes :

EMPLOYEES

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID
100	King	90
101	Kochhar	90
...		
202	Fay	20
205	Higgins	110
206	Gietz	110

DEPARTMENTS

DEPARTMENT_ID	DEPARTMENT_NAME	LOCATION_ID
10	Administration	1700
20	Marketing	1800
50	Shipping	1500
60	IT	1400
80	Sales	2500
90	Executive	1700
110	Accounting	1700
190	Contracting	1700

EMPLOYEE_ID	DEPARTMENT_ID	DEPARTMENT_NAME
200	10	Administration
201	20	Marketing
202	20	Marketing
...		
102	90	Executive
205	110	Accounting
206	110	Accounting

Pour créer un tel rapport, vous avez besoin de lier les tables **EMPLOYEES** et **DEPARTMENTS** et accéder aux données de chacune d'elles.

Modélisation d'une relation

- Pourquoi plusieurs Tables ?

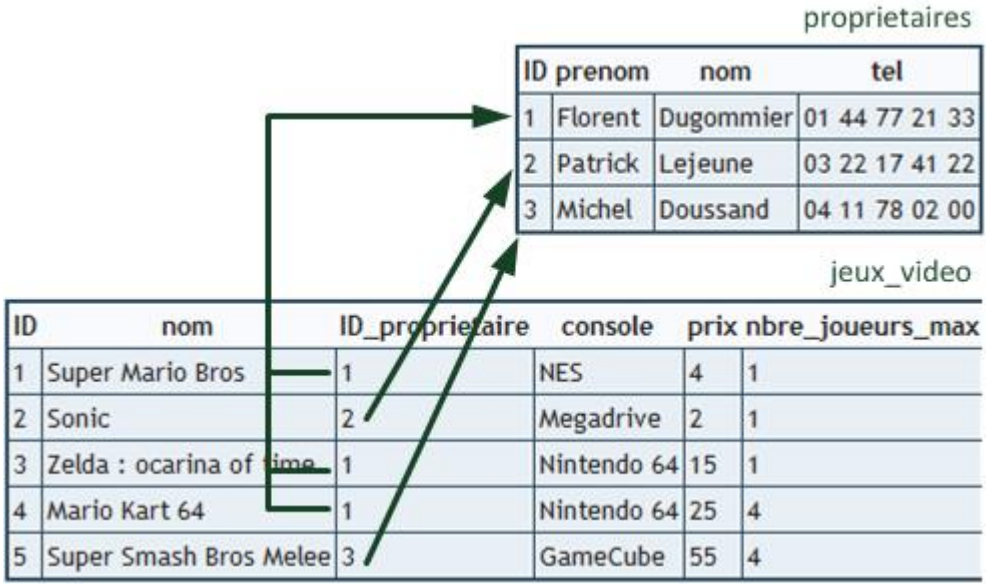
ID	nom	prenom	nom_possesseur	tel	console	prix	nbre_joueurs_max	commentaires
1	Super Mario Bros	**Florent**	**Dugommier**	**01 44 77 21 33**	NES	4	1	Un jeu d'anthologie !
2	Sonic	Patrick	Lejeune	03 22 17 41 22	Megadrive	2	1	Pour moi, le meilleur jeu au monde !
3	Zelda : ocarina of time	**Florent**	**Dugommier**	**01 44 77 21 33**	Nintendo 64	15	1	Un jeu grand, beau et complet comme on en voit rarement de nos jours
4	Mario Kart 64	**Florent**	**Dugommier**	**01 44 77 21 33**	Nintendo 64	25	4	Un excellent jeu de kart !
5	Super Smash Bros Melee	Michel	Doussand	04 11 78 02 00	GameCube	55	4	Un jeu de baston délirant !

ID	prenom	nom	tel
1	Florent	Dugommier	01 44 77 21 33
2	Patrick	Lejeune	03 22 17 41 22
3	Michel	Doussand	04 11 78 02 00

ID	nom	ID_proprietaire	console	prix	nbre_joueurs_max	commentaires
1	Super Mario Bros	1	NES	4	1	Un jeu d'anthologie !
2	Sonic	2	Megadrive	2	1	Pour moi, le meilleur jeu au monde !
3	Zelda : ocarina of time	1	Nintendo 64	15	1	Un jeu grand, beau et complet comme on en voit rarement de nos jours
4	Mario Kart 64	1	Nintendo 64	25	4	Un excellent jeu de kart !
5	Super Smash Bros Melee	3	GameCube	55	4	Un jeu de baston délirant !

Modélisation d'une relation(MCD)

Le nouveau champ ID_proprietaire est de type INT. Il permet de faire référence à une entrée précise de la table proprietaires.
On peut maintenant considérer que les tables sont reliées à travers ces ID de propriétaires, comme le suggère la figure suivante.



proprietaires

ID	prenom	nom	tel
1	Florent	Dugommier	01 44 77 21 33
2	Patrick	Lejeune	03 22 17 41 22
3	Michel	Doussand	04 11 78 02 00

jeux_video

ID	nom	ID_proprietaire	console	prix	nbre_joueurs_max
1	Super Mario Bros	1	NES	4	1
2	Sonic	2	Megadrive	2	1
3	Zelda : ocarina of time	1	Nintendo 64	15	1
4	Mario Kart 64	1	Nintendo 64	25	4
5	Super Smash Bros Melee	3	GameCube	55	4

Clef primaire et clef etrangere

- **La clé primaire (primary key)** concourt à identifier uniquement chaque ligne d'une table. **La clé primaire est Auto-incrémenté**
- **La clé étrangère (foreign key)** représente un champ (ou des champs) qui pointe vers la clé primaire d'une autre table. L'objectif de la clé étrangère est d'assurer l'intégrité référentielle des données.

Introduction aux jointures

Nous avons donc maintenant deux tables :

- jeux_video ;
- proprietaires.

Les informations sont séparées dans des tables différentes et c'est bien. Cela évite de dupliquer des informations sur le disque.

Cependant, lorsqu'on récupère la liste des jeux, si on souhaite obtenir le nom du propriétaire, il va falloir adapter la requête pour récupérer aussi les informations issues de la table proprietaires. Pour cela, on doit faire ce qu'on appelle une **jointure**.

Introduction aux jointures

- Les jointures permettent d'exploiter pleinement le modèle relationnel des tables d'une base de données.
- Elle sont faites pour mettre en relation deux (ou plus) tables concourant à rechercher la réponse à des interrogations.
- Une jointure permet donc de combiner les colonnes de plusieurs tables.

Introduction aux jointures

- Il existe plusieurs types de jointures, qui nous permettent de choisir exactement les données que l'on veut récupérer. Je vous propose d'en découvrir deux, les plus importantes :
- **les jointures internes** : elles ne sélectionnent que les données qui ont une correspondance entre les deux tables ;
- **les jointures externes** : elles sélectionnent toutes les données, même si certaines n'ont pas de correspondance dans l'autre table.

Comparaison(1)

Exemple :

Imaginons que nous ayons une 4e personne dans la table des propriétaires, un certain Romain Vipelli, qui ne possède aucun jeu

Romain Vipelli est référencé dans la table proprietaires ma is il n'apparaît nulle part dans la table jeux_video car il ne possède aucun jeu.

ID	prenom	nom	tel
1	Florent	Dugommier	01 44 77 21 33
2	Patrick	Lejeune	03 22 17 41 22
3	Michel	Doussand	04 11 78 02 00
4	**Romain**	**Vipelli**	**01 21 98 51 01**

Comparaison(2)

Si vous récupérez les données des deux tables à l'aide :

- d'une jointure interne : Romain Vipelli n'apparaîtra pas dans les résultats de la requête. La jointure interne force les données d'une table à avoir une correspondance dans l'autre ;
- d'une jointure externe : vous aurez toutes les données de la table des propriétaires, même s'il n'y a pas de correspondance dans l'autre table des jeux vidéo ; donc Romain Vipelli, qui pourtant ne possède aucun jeu vidéo, apparaîtra.

Comparaison(3)

nom_jeu	prenom
Super Mario Bros	Florent
Sonic	Patrick
…	…
NULL	Romain

Romain apparaît maintenant. Comme il ne possède pas de jeu, il n'y a aucun nom de jeu indiqué (NULL).

La jointure interne

Une jointure interne peut être effectuée de deux façons différentes :

- à l'aide du mot-clé **WHERE** : c'est l'ancienne syntaxe, toujours utilisée aujourd'hui, qu'il faut donc connaître mais que vous devriez éviter d'utiliser si vous avez le choix ;
- à l'aide du mot-clé **JOIN** : c'est la nouvelle syntaxe qu'il est recommandé d'utiliser. Elle est plus efficace et plus lisible.

Ces deux techniques produisent exactement le même résultat, mais il faut les connaître toutes les deux. ;-)

La jointure interne avec Where **(ancienne syntaxe)**

```
SELECT jeux_video.nom, proprietaires.nom

FROM proprietaires, jeux_video

WHERE jeux_video.ID_proprietaires = proprietaires.ID
```

nom	nom
Super Mario Bros	Florent
Sonic	Patrick
…	…
NULL	Romain

Utilisez les alias !

```
SELECT j.nom nom_jeu, p.nom nom_proprietaires
FROM proprietaires p, jeux_video j
WHERE j.ID_proprietaires = p.ID
```

Nom_jeu	Nom_proprietaires
Super Mario Bros	Florent
Sonic	Patrick
…	…
NULL	Romain

La jointure interne avec JOIN

- Bien qu'il soit possible de faire une jointure interne avec un WHERE comme on vient de le voir, c'est une ancienne syntaxe et aujourd'hui on recommande plutôt d'utiliser JOIN. Il faut dire que nous étions habitués à utiliser le WHERE pour filtrer les données, alors que nous l'utilisons ici pour associer des tables et récupérer plus de données.
- Pour éviter de confondre le WHERE « traditionnel » qui filtre les données et le WHERE de jointure que l'on vient de découvrir, on va utiliser la syntaxe JOIN.

La jointure interne avec JOIN

```
SELECT j.nom nom_jeu, p.nom nom_proprietaire

FROM proprietaires p

INNER JOIN jeux_video j

ON j.ID_proprietaire = p.ID
```

Cette fois, on récupère les données depuis une table principale (ici, proprietaires) et on fait une jointure interne (INNER JOIN) avec une autre table (jeux_video). La liaison entre les champs est faite dans la clause ON

Demo

Si vous voulez filtrer (WHERE), ordonner (ORDER BY) ou limiter les résultats (LIMIT), vous devez le faire à la fin de la requête, après le « ON j.ID = p.ID ».

```
SELECT j.nom nom_jeu, p.prenom
prenom_proprietaire
FROM proprietaires p
INNER JOIN jeux_video j
ON j.ID = p.ID
WHERE j.console = 'PC'
ORDER BY prix DESC
LIMIT 0, 10
```

Récupère le nom du jeu et le prénom du propriétaire dans les tables proprietaires et jeux_video, la liaison entre les tables se fait entre les champs ID_proprietaire et ID, prends uniquement les jeux qui tournent sur PC, trie-les par prix décroissants et ne prends que les 10 premiers

La jointure externe

- Les jointures externes permettent de récupérer toutes les données, même celles qui n'ont pas de correspondance. On pourra ainsi obtenir Romain Vipelli dans la liste même s'il ne possède pas de jeu vidéo.
- Cette fois, la seule syntaxe disponible est à base de JOIN. Il y a deux écritures à connaître : **LEFT JOIN** et **RIGHT JOIN**. Cela revient pratiquement au même, avec une subtile différence que nous allons voir.

La jointure externe

- **LEFT** JOIN : récupérer toute la table de gauche

Reprenons la jointure à base de INNER JOIN et remplaçons tout simplement INNER par LEFT :

```
SELECT j.nom nom_jeu, p.prenom prenom_proprietaire

FROM proprietaires p

LEFT JOIN jeux_video j

ON j.ID_proprietaire = p.ID
```

Propriétaires est appelée la « table de gauche » et jeux_video la « table de droite ». Le LEFT JOIN demande à récupérer tout le contenu de la table de gauche, donc tous les propriétaires, même si ces derniers n'ont pas d'équivalence dans la table jeux_video.

La jointure externe

- **RIGHT** JOIN : récupérer toute la table de droite

Le RIGHT JOIN demande à récupérer toutes les données de la table dite « de droite », même si celle-ci n'a pas d'équivalent dans l'autre table. Prenons la requête suivante :

```
SELECT j.nom nom_jeu, p.prenom prenom_proprietaire

FROM proprietaires p

RIGHT JOIN jeux_video j

ON j.ID_proprietaire = p.ID
```

La table de droite est « jeux_video ». On récupèrerait donc tous les jeux, même ceux qui n'ont pas de propriétaire associé.

Exemple

LEFT JOIN

nom_jeu	prenom_proprietaire
Super Mario Bros	Florent
Sonic	Patrick
…	…
NULL	Romain

RIGHT JOIN

nom_jeu	prenom_proprietaire
Super Mario Bros	Florent
Sonic	Patrick
…	…
Bomberman	NULL

En résumé

- Les bases de données permettent d'associer plusieurs tables entre elles.
- Une table peut contenir les id d'une autre table ce qui permet de faire la liaison entre les deux. Par exemple, la table des jeux vidéo contient pour chaque jeu l'id de son propriétaire. Le nom et les coordonnées du propriétaire sont alors stockés dans une table à part.
- Pour rassembler les informations au moment de la requête, on effectue des jointures.
- On peut faire des jointures avec le mot-clé WHERE, mais il est recommandé d'utiliser JOIN qui offre plus de possibilités et qui est plus adapté.
- On distingue les jointures internes, qui retournent des données uniquement s'il y a une correspondance entre les deux tables, et les jointures externes qui retournent toutes les données même s'il n'y a pas de correspondance.