Un modèle relationnel, des requêtes SQL

INFORMATIQUE COMMUNE - TP n° 3.5 - Olivier Reynet

À la fin de ce chapitre, je sais :

- ☞ interpréter et utiliser un modèle relationnel de base de données
- ☞ utiliser les opérateurs de projection et de sélection sur un modèle simple (select from, where)
- ☞ utiliser les clefs primaires et étrangères dans une requête simple
- ☞ opérer une jointure interne entre plusieurs tables (join on)
- ☞ utiliser les fonctions d'agrégation pour un calcul simple (min, max, sum, avg, count)
- ☞ filtrer des agrégations d'après un critère (having)
- ☞ utiliser des opérateurs ensemblistes (intersect, union, except)

Ce TP s'inspire de l'épreuve d'informatique commune du concours Centrale 2020.

A Présentation de la base de données

Des photographies sont répertoriées dans une base de données selon le modèle relationnel décrit sur la figure 1. À chaque photographie, on associe des mots-clefs et un auteur. Un même mot-clef peut qualifier plusieurs photographies différentes. On recense par ailleurs les personnes présentes sur la photo.

B Sur le modèle

B1. Donner le nom de toutes les clefs primaires du modèle relationnel de la base de données.

> **Solution :** PH_id dans la table photo, PE_id dans la table person, KW_id dans la table keyword. Ces clefs sont généralement soulignées ou en italique dans la représentation du modèle relationnel.

B2. Pourquoi n'a-t-on pas choisi le nom d'une personne comme identifiant (clef primaire) ?

> **Solution :** Parce qu'on peut avoir des personnes dont le nom est le même mais qui représentent des personnes différentes. Par exemple, Jean Martin en France...

B3. Donner le nom des clefs étrangères du modèle.

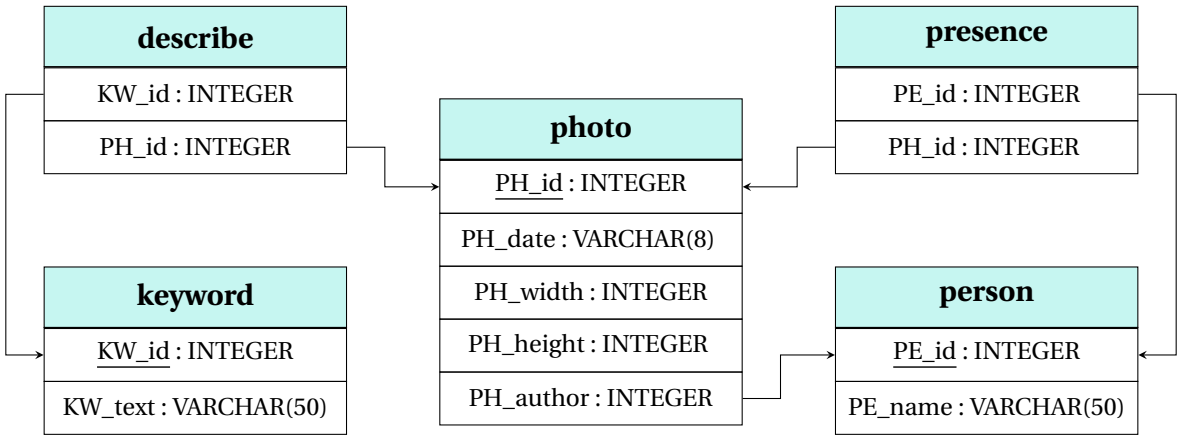


FIGURE 1 – Modèle physique de la base de données des photographies.

Solution : PH_author dans la table photo, PH_id dans les tables presence et describe, PE_id dans la table presence, ăKW_id dans la table describe.

B4. La table describe ne présente pas de clef primaire apparente. Que pourrait-il se passer ? Proposer une clef primaire pour cette table sans la modifier. Proposer une solution identique pour la table presence.

Solution : On peut choisir le couple (KM_id, PH_id) comme clef primaire de describe. Ainsi on ne pourra pas associer une photo deux fois avec le même mot clef. De même, pour éviter de signaler plusieurs fois la présence d'une personne sur une photo, on peut choisir le couple (PE_id, PH_id) comme clef primaire.

B5. La relation describe représente une association du modèle conceptuel. Comment pourrait-on la nommer ? Identifier les cardinalités de cette association. De quel type cette relation est-elle ?

Solution : On pourrait la nommer describe (décrire) pour signifier *un mot-clef décrit un photographie*. Une même photo peut-être décrit par plusieurs mots-clefs. Un mot-clef décrit plusieurs photos différentes. On a donc les cardinalités suivantes : keyword $0-n$ describe $0-n$ photo. C'est une association de plusieurs à plusieurs.

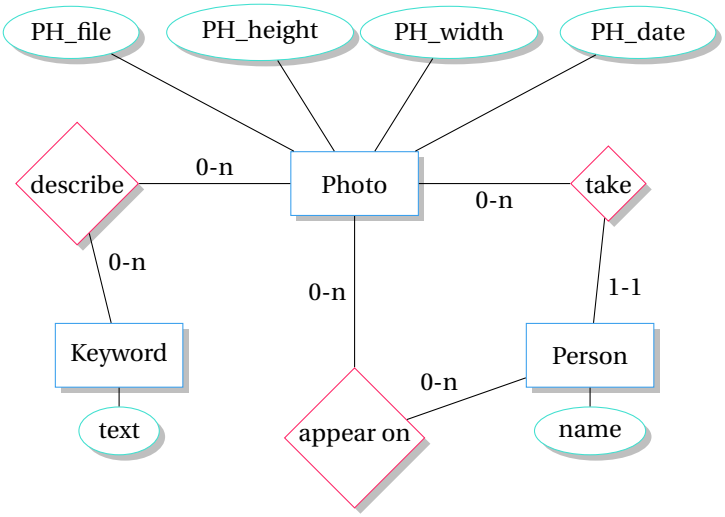
B6. La relation person est mise en jeu dans deux associations du modèle conceptuel. Décrire ces deux associations en précisant leur nom et leurs cardinalités ainsi que leur type.

Solution : On pourrait nommer take (prendre) l'association qui réunit photo et person pour signifier *une personne prend une photographie*. Une photo ne possède qu'un seul auteur. Par contre, une même personne peut prendre plusieurs photos. Donc, les cardinalités s'écrivent : person $0-1$ take $0-n$ photo. C'est une association de un à plusieurs.

On pourrait nommer appear on (figure sur) l'association qui réunit photo et person pour signifier *une personne figure sur une photographie*. Une personne peut apparaître sur plusieurs photos. Plusieurs personnes peuvent apparaître sur une même photo. Donc, les cardinalités s'écrivent : person $0-n$ appear on $0-n$ photo. C'est une association de plusieurs à plusieurs.

B7. Représenter le modèle conceptuel associé à ce modèle relationnel.

Solution : Un modèle entité-relation associé est proposé ci-dessous.



C Échauffement

Écrire une requête SQL qui permet d'obtenir :

C1. tous les textes des mots clefs.

C2. les identifiants et les dates des photos prises avant l'année 2001. On utilisera le fait qu'une comparaison de chaînes de caractères de type "YYYY-MM-DD" respecte la chronologie ! Par exemple, on a bien "2021-12-25" < "2021-12-26" dans l'ordre lexicographique des chaînes de caractères.

C3. tous les textes des mots clefs **utilisés**. Le résultat de la requête n'affiche aucun doublon.

C4. tous les **identifiants** des auteurs des photos. Le résultat de la requête n'affiche aucun doublon.

C5. tous les **noms** des auteurs des photos. Le résultat de la requête n'affiche aucun doublon.

C6. les noms des auteurs qui ont pris des photos après l'année 2001. La requête n'affiche pas de doublons.

C7. le nombre de personnes présentes sur une photo, y compris s'il n'y a personne dessus. La requête affiche l'identifiant de la photo suivi du nombre. Les résultats sont ordonnés d'après ce nombre de manière croissante.

C8. le nombre de fois qu'une personne a été photographiée. La requête affiche le nom et le nombre s'il est strictement supérieur à 2. Les résultats sont ordonnés d'après le nom de la personne dans l'ordre alphabétique inverse.

Solution :

```sql
SELECT text
FROM keyword;

SELECT PH_id, PH_date
FROM photo
WHERE PH_date < "2001";

SELECT DISTINCT text
FROM keyword
  JOIN describe on keyword.KW_id = describe.KW_id;

SELECT DISTINCT PH_author
FROM photo;

SELECT DISTINCT name
FROM person
JOIN photo on person.PE_id = photo.PH_author;

SELECT DISTINCT name
FROM person
JOIN photo on person.PE_id = photo.PH_author
WHERE PH_date > "2001";

SELECT photo.PH_id, COUNT(PE_id)
FROM photo
  JOIN presence ON photo.PH_id = presence.PH_id
GROUP BY photo.PH_id
ORDER BY COUNT(PE_id) ASC;

SELECT name, COUNT(presence.PE_id)
FROM presence
JOIN person ON presence.PE_id = person.PE_id
GROUP BY person.PE_id
    HAVING COUNT(presence.PE_id) > 2
ORDER BY name DESC;
```

D On est chaud !

Écrire une requête SQL qui permet d'obtenir :

D1. les identifiants de toutes les photographies au format `4:3`, c'est à dire dont le rapport largeur sur hauteur vaut exactement 4/3.

D2. le nombre de photos qui n'ont pas été prises par Alix et Guillaume.

D3. l'identifiant, le nom de l'auteur et la date des photographies prises avant 2006 et associées au mot clef "chat".

D4. les selfies. La requête affiche l'identifiant de la photo ainsi que le nom de l'auteur. Proposer deux versions différentes : avec double jointure, avec intersection d'ensembles.

D5. toutes les photographies où sont présents Alix et Guillaume à l'exclusion de toute autre personne. La requête affiche les identifiants des photos.

Solution :

```sql
-- On est chaud !

SELECT PH_id
FROM photo
WHERE 3 * PH_width = 4 * photo.PH_height;

SELECT COUNT(*)
  FROM photo
  JOIN person
  ON photo.PH_author = person.PE_id
  WHERE person.PE_name NOT IN ("Alix", "Guillaume");

-- idem avec sous requête
SELECT COUNT(*)
FROM photo
WHERE PH_author NOT IN (SELECT PE_id
                        FROM person
                        WHERE PE_name IN ("Alix", "Guillaume"));

SELECT photo.PH_id, PE_name, PH_date
FROM photo
JOIN describe ON photo.PH_id = describe.PH_id
JOIN keyword ON describe.KW_id = keyword.KW_id
JOIN person ON photo.PH_author = person.PE_id
WHERE PH_date < "20060101"
            AND keyword.KW_text = "chat";


  SELECT photo.PH_id, person.PE_name
  FROM photo
  JOIN presence, person
  ON photo.PH_author = person.PE_id
          AND photo.PH_id = presence.PH_id
          AND presence.PE_id = person.PE_id;

-- idem avec INTERSECT
-- l'identifiant d'une photo et son auteur
-- intersect
-- l'identifiant d'une photo et le nom d'une personne presente
SELECT photo.PH_id, person.PE_name
FROM photo
  JOIN person ON photo.PH_author = person.PE_id
INTERSECT
SELECT presence.PH_id, person.PE_name
FROM presence
  JOIN person ON presence.PE_id = person.PE_id;


-- Good luck !
-- les photos sur lesquelles figure Alix
-- intersect
-- les photos sur lesquelles figure Guillaume
-- except
-- les photos sur lesquelles ne figurent ni Alix ni Guillaume
```

```sql
SELECT photo.PH_id
FROM photo
  JOIN presence ON photo.PH_id = presence.PH_id
  JOIN person ON presence.PE_id = person.PE_id
WHERE person.PE_name = "Alix"
INTERSECT
SELECT photo.PH_id
FROM photo
  JOIN presence ON photo.PH_id = presence.PH_id
  JOIN person ON presence.PE_id = person.PE_id
WHERE person.PE_name = "Guillaume"
EXCEPT
SELECT photo.PH_id
FROM photo
  JOIN presence ON photo.PH_id = presence.PH_id
  JOIN person ON presence.PE_id = person.PE_id
WHERE person.PE_name NOT IN ("Alix", "Guillaume");
```

E Amélioration du modèle

On souhaite pouvoir internationaliser le système de mots-clefs pour rendre le partage des photographies plus facile. On souhaite intégrer les mots-clefs dans plusieurs langues. Les modifications du cahier des charges s'énonce ainsi :

1. l'ensemble des photographies sélectionnées à l'aide de mots-clefs ne doit pas dépendre de la langue utilisée pour exprimer les mots clefs. Les photographies sélectionnées à l'aide du mot clef *montagne* doivent être les mêmes qu'avec le mot clef *mountain* si la langue choisie est l'anglais, *berg* pour l'allemand ou *montaña* pour l'espagnol.
2. il doit être possible, avec ce nouveau modèle, d'écrire une requête de recherche de photographies par mot-clef en spécifiant la langue utilisée pour exprimer le mot clef de telle sorte que changer de langue se fasse en modifiant uniquement des constantes dans la clause `WHERE`.

E1. Proposer un nouveau modèle répondant à cette évolution du cahier des charges.

Solution : On modifie la table keyword comme suit en faisant apparaître une nouvelle colonne KW_language de type `VARCHAR(80)`. KW_id reste le même pour tous les mots clefs qui ont le même sens. Cet identifiant peut apparaître plusieurs fois dans la table, il faut donc une nouvelle clef primaire. Cette nouvelle clef primaire de la table peut être le couple (KW_id, KW_language). Le champ KW_text varie d'une langue à une autre, mais partage KW_id avec tous les mots possédant le même sens dans les autres langues.

keyword
KW_id : INTEGER
KW_language : VARCHAR(80)
KW_text : VARCHAR(50)

E2. Proposer un exemple de requête permettant de sélectionner les identifiants des photographies associées au mot-clef *mountain* exprimé en anglais.

Solution :

```
SELECT PH_id
FROM photo
  JOIN describe ON photo.PH_id = describe.PH_id
  JOIN keyword ON describe.KW_id = keyword.KW_id
WHERE KW_text = "mountain" AND KW_language = "english" ;
```

On voit que pour passer au français, il suffit bien de changer le mot clef par "montagne" la constante du dernier test d'égalité par "french".