Licences Mathématiques et Informatique
3ème année - Formation Initiale et par Apprentissage

Bases de données relationnelles

Polycopié de cours - Dépendances fonctionnelles

Maude Manouvrier

Table des matières

Chapitre 8

Dépendances fonctionnelles

Une **dépendance fonctionnelle** sur une relation R est une contrainte de la forme [21] : “ Quelle soit une instance r de schéma $R(A_1, A_2 ..., A_n, B)$, quels que soient deux nuplets de r, s'ils ont même valeur pour les attributs A_1, A_2 ..., A_n alors ils ont même valeur pour l'attribut B”. Cette dépendance fonctionnelle est notée : $A_1, A_2, ..., A_n \longrightarrow B$. Elle peut se traduire également par "Si on connaît la valeur des attributs $A_1, A_2, ..., A_n$ alors on peut en déduire celle de B" ou encore "A une valeur de l'ensemble d'attributs $\{A_1, A_2, ..., A_n\}$ est associée une et une seule valeur de B".

On a donc par définition, pour X et Y, 2 ensembles d'attributs d'un schéma de relation R :
$X \longrightarrow Y \Longleftrightarrow \forall\ r$, une instance de R et $\forall\ t_1$, t_2, 2 nuplets de r, $t_1.X = t2.X \Longrightarrow t_1.Y = t2.Y$

$X \longrightarrow Y$ signifie donc que pour toutes les instances possibles de R, si on prend 2 nuplets t_1 et t_2 au hasard dans r, s'ils ont la même valeur pour X alors ils ont la même valeur pour Y.

Inversement, $X \longrightarrow Y$ n'est pas vérifiée par R signifie :
$\exists r$, une instance de R et $\exists\ t_1$, t_2, 2 nuplets de r, tels que $t_1.X = t2.X$ et $t_1.Y \neq t2.Y$.
$\neg(X \longrightarrow Y)$ signifie donc qu'il existe une instance de R contenant deux nuplets t_1 et t_2 avec la même valeur pour X mais pas la même valeur pour Y.

8.1 Définitions : dépendances triviales, non triviales

La dépendance fonctionnelle notée $A_1, A_2, ...A_n \longrightarrow B$ signifie que “les attributs A_1, A_2 ..., A_n détermine fonctionnellement l'attribut B”.

Si un ensemble d'attributs $A_1, A_2, ...A_n$ détermine fonctionnellement plus d'un attribut [19] :
$A_1, A_2, ...A_n \longrightarrow B_1$
$A_1, A_2, ...A_n \longrightarrow B_2$ ...
$A_1, A_2, ...A_n \longrightarrow B_m$

On peut écrire l'ensemble des dépendances fonctionnelles de la manière suivante :
$A_1, A_2, ...A_n \longrightarrow B_1, B_2, ..., B_m$

Soit 2 ensembles d'attributs X et Y d'une relation R, une dépendance fonctionnelle notée $X \longrightarrow Y$ est [19] :

- **triviale** : si $Y \subseteq X$;
- **non triviale** : si $X \nsubseteq Y$ et $X \cap Y \neq \emptyset$;
- **complètement non triviale** : si $X \cap Y = \emptyset$.

8.2 Règles entre dépendances fonctionnelles

A partir d'un ensemble de dépendances fonctionnelles, on peut en déduire d'autres.

8.2.1 Réfléxivité

Si, on note X et Y deux ensemble d'attributs tels que : $Y \subseteq X$ alors $X \longrightarrow Y$. En effet : $\forall t, u$ si on a $t.X = u.X$ on a forcément $t.Y = u.Y$ puisque Y est un sous-ensemble de X.

8.2.2 Augmentation

Si on a la dépendance fonctionnelle $X \longrightarrow Y$ alors pour tout ensemble d'attributs Z, on a la dépendance fonctionnelle $XZ \longrightarrow YZ$.

Procédons par l'absurde, on suppose que l'on a $X \longrightarrow Y$, mais pas $XZ \longrightarrow YZ$. $X \longrightarrow Y$ signifie que : $\forall t, u$ $t.X = u.X \Longrightarrow t.Y = u.Y$. Or, si on n'a pas $XZ \longrightarrow YZ$ cela signifie que $\exists t, u$, 2 nuplets d'une instance r, tels que $t.XZ = u.XZ$ et $t.YZ \neq u.YZ$. Comme Z est un attribut commun aux 2 côté de la dépendance, si on n'a pas $XZ \longrightarrow YZ$, alors cela signifie que l'on a $t.X = u.X$ et $t.Y \neq u.Y$, ce qui est contredit la dépendance $X \longrightarrow Y$.

8.2.3 Transitivité

Si on a les dépendances suivantes : $A \longrightarrow B$ et $B \longrightarrow C$ alors on en déduit que $A \longrightarrow C$. En effet $A \longrightarrow B$ signifie que : $\forall t, u : t.A = u.A \Longrightarrow t.B = u.B$ et $B \longrightarrow C$ signifie que : $\forall t, u : t.B = u.B \Longrightarrow t.C = u.C$.

On note le fait qu'un ensemble de dépendances fonctionnelles F implique logiquement un ensemble de dépendances fonctionnelles $\{X \longrightarrow Y\}$ par : $F \models \{X \longrightarrow Y\}$.

Dans l'exemple précédent, on avait : $\{A \longrightarrow B, B \longrightarrow C\} \models \{A \longrightarrow C\}$.

8.2.4 Union - Pseudo-transitivité - Décomposition

Des règles précédentes, appelées axiomes de Amstrong, on déduit :

- **Union** : $\{X \longrightarrow Y, X \longrightarrow Z\} \models \{X \longrightarrow YZ\}$
- **Pseudo-transitivité** : $\{X \longrightarrow Y, WY \longrightarrow Z\} \models \{XW \longrightarrow Z\}$
- **Décomposition** : Si $X \longrightarrow Y$ et $Z \subseteq Y$ alors $X \longrightarrow Z$

  Démonstration de l'Union :
  $X \longrightarrow Y \models X \longrightarrow XY$ par augmentation
  $X \longrightarrow Z \models XY \longrightarrow YZ$ par augmentation
  $X \longrightarrow XY$ et $XY \longrightarrow YZ \models X \longrightarrow YZ$ par transitivité.

  Démonstration de la Pseudo-transitivité :
  $X \longrightarrow Y \models XW \longrightarrow YW$ par augmentation
  $XW \longrightarrow YW$ et $WY \longrightarrow Z \models XW \longrightarrow Z$ par transitivité.

  Démonstration de la Décomposition :
  $Z \subseteq Y$ alors $Y \longrightarrow Z$ par réflexivité
  $X \longrightarrow Y$ et $Y \longrightarrow Z \models X \longrightarrow Z$ par transitivité.

8.3 Fermeture

On appelle F^+ la **fermeture** de F, l'ensemble des dépendances fonctionnelles déduites de F c'est-à-dire : $F^+ = \{X \longrightarrow Y \backslash F \models X \longrightarrow Y\}$.

La fermeture d'un ensemble d'attributs X par rapport à une famille de dépendances fonctionnelles F est notée $[X]^+$ et est l'ensemble des attributs A tels que $X \longrightarrow A$ peut être déduit de F par les axiomes d'Amstrong.

Pour construire la fermeture d'un ensemble d'attributs $X = \{A_1, A_2, ..., A_n\}$, il faut suivre l'algorithme suivant [21] :

1. Soit $X_0 = \{A_1, A_2, ..., A_n\}$.
2. Pour tout i (en partant de $i = 0$) : on calcule $X_{i+1} = X_i \cup \{Z \mid Y \longrightarrow Z \in F, Y \subseteq X_i\}$; i.e. on cherche les dépendances fonctionnelles de la forme $Y \longrightarrow Z$, que l'on peut déduire de F et telles que Y soit un sous-ensemble d'attributs de X et Z n'y soit pas. Alors Z est ajouté à X_{i+1}.
3. L'étape 2 est répétée tant qu'aucun attribut ne peut plus être ajouté à X_{i+1}.
4. A la fin $[X]^+ = X_{i+1}$

Par exemple, considérons la relation R avec pour attributs A, B, C, D, E et G et l'ensemble $F = \{AB \longrightarrow C, BC \longrightarrow AD, D \longrightarrow E, CG \longrightarrow B\}$ [21].
Calculons la fermeture $[AB]^+$:
$X_0 = \{A, B\}$
On a $AB \longrightarrow C$ d'où $X_1 = \{A, B, C\}$
On a $BC \longrightarrow AD$, soit par décomposition $BC \longrightarrow A$ et $BC \longrightarrow D$, d'où $X_2 = \{A, B, C, D\}$
On a $D \longrightarrow E$ d'où $X_3 = \{A, B, C, D, E\}$
La dépendance fonctionnelle $CG \longrightarrow B$ ne peut pas être utilisée car G n'appartient pas à X.
Par conséquent $[AB]^+ = \{A, B, C, D, E\}$.

Lemme : La dépendance fonctionnelle $X \longrightarrow Y$ peut être déduite des axiomes d'Amstrong si $Y \subseteq [X]^+$.

Démonstration : Soit $Y = \{A_1, A_2, ...A_n\}$

1. On suppose que $Y \subseteq [X]^+$, par définition de $[X]^+$, $X \longrightarrow A_i$ est déduit des axiomes d'Amstrong. Par union, on en déduit que $X \longrightarrow Y$.
2. On suppose que $X \longrightarrow Y$, par décomposition on a que $X \longrightarrow A_i$, pour $i \in [1, n]$ d'où $A_i \subseteq [X]^+$ d'où $Y \subseteq [X]^+$.

Calculer $[X]^+$ est moins coûteux que de calculer F^+, car la taille de F^+ peut être très grande. La taille de $[X]^+$ est proportionnelle au nombre de dépendances fonctionnelles contenues dans F.

8.4 Equivalence - couverture - famille minimale

8.4.1 Définitions d'équivalence et de couverture

Si F et G sont deux ensembles de dépendances fonctionnelles sur un ensemble d'attributs U, on dit que F et G sont **équivalentes** si $F^+ = G^+$.

Si $F^+ \subset G^+$, on dit que G est une **couverture** de F.

Par exemple, montrons que $F = \{A \longrightarrow BC, B \longrightarrow C\}$ et $G = \{A \longrightarrow B, B \longrightarrow C\}$ sont équivalentes :
Par décomposition de $A \longrightarrow BC$ on a que $F \models \{A \longrightarrow B, A \longrightarrow C, B \longrightarrow C\}$.
Par transitivité de $A \longrightarrow B$ et $B \longrightarrow C$, on a que $G \models \{A \longrightarrow B, A \longrightarrow C, B \longrightarrow C\}$.
On déduit donc la même famille de dépendances fonctionnelles de F et de G, donc $F^+ = G^+$.

Il est à noter qu'en général, on raisonnera plutôt sur la fermeture des attributs que sur la fermeture des ensembles de dépendances fonctionnelles.

8.4.2 Famille minimale

Pour définir un schéma de bases de données, on veut trouver un ensemble de dépendances fonctionnelles qui soit le plus simple possible.

Un ensemble de dépendances fonctionnelles F est **minimal** si :

1. En partie droite de toute dépendance de F, il n'y a qu'un seul attribut. On dit que les dépendances sont sous forme canonique.
2. Il n'y a pas de dépendances fonctionnelles $X \longrightarrow A$ dans F telle que $F \setminus \{X \longrightarrow A\}$ soit équivalente à F. *Il n'est pas possible d'éliminer une dépendance fonctionnelle de F et de la retrouver par transitivité.*
3. Il n'y a pas de dépendance fonctionnelle $X \longrightarrow A$ et $Z \subset X$ tels que $(F \setminus \{X \longrightarrow A\}) \cup \{Z \longrightarrow A\}$ soit équivalent à F. *La partie gauche des dépendances fonctionnelles de F est la plus petite possible.*

Pour trouver une **famille minimale équivalente** à F, on suit l'algorithme suivant :

1. On applique la décomposition sur toutes les DF non canoniques de F de telle sorte que chaque dépendance ait un seul attribut à droite. Soit $F_0 = F$ avec toutes les DF sous forme canonique. F_0 est équivalente à F et vérifie la propriété 1 d'une famille minimale
2. On détermine une suite F_1 ...F_p telle que :
   (a) $F_{i+1} = F_i \setminus \{X_i \longrightarrow A\}$ avec $X_i \longrightarrow A$ une dépendance de fonctionnelle de F_i et F_{i+1} équivalente à F_i.
   (b) On répète l'étape (a) tant que on peut enlever des dépendances (au pire au aura autant d'étapes que de dépendances fonctionnelles).
   Quand on ne peut plus enlever de DF, on obtient une famille $F' = F_{i+1}$ équivalente à F qui vérifie les propriétés 1 et 2 d'une famille minimale.
3. On détermine une suite F'_0, F'_1 ...F'_p telle que :
   (a) $F'_0 = F'$
   (b) $F'_{i+1} = F'_i \setminus \{X_j \longrightarrow A\} \cup \{Y_j \longrightarrow A\}$ où $Y_j \subset X_j$ et F'_{i+1} équivalente à F'_i.
   (c) On répète l'étape (b) tant on peut enlever des attributs à gauche des dépendances (au pire au aura autant d'étapes que de dépendances fonctionnelles).
   Quand on ne peut plus diminuer de partie gauche de DF, on obtient $F'' = F'_{i+1}$ minimale et équivalente à F.

Attention : il n'y a pas d'unicité. L'algorithme permet de trouver différentes familles minimales équivalentes à F.

Par exemple [21] :
Soient la relation $R(A,B,C)$ et $F = \{A \longrightarrow B, A \longrightarrow C, B \longrightarrow A, B \longrightarrow C, C \longrightarrow A, C \longrightarrow B\}$.

On pose $F_0 = F$
$F_1 = F_0 \setminus \{A \longrightarrow C\}$ car $A \longrightarrow C$ peut être déduite par transitivité de $A \longrightarrow B$ et $B \longrightarrow C$.
$F_2 = F_1 \setminus \{C \longrightarrow A\}$ car $C \longrightarrow A$ peut être déduite par transitivité de $C \longrightarrow B$ et $B \longrightarrow A$.
D'où on obtient une famille minimale équivalente à F : $\{A \longrightarrow B, B \longrightarrow A, B \longrightarrow C, C \longrightarrow B\}$.

Si on reprend à partir de F_0 :
$F_1 = F_0 \setminus \{A \longrightarrow C\}$ car $A \longrightarrow C$ peut être déduite par transitivité de $A \longrightarrow B$ et $B \longrightarrow C$.
$F_2 = F_1 \setminus \{C \longrightarrow B\}$ car $C \longrightarrow B$ peut être déduite par transitivité de $C \longrightarrow A$ et $A \longrightarrow B$.
$F_3 = F_2 \setminus \{B \longrightarrow A\}$ car $B \longrightarrow A$ peut être déduite par transitivité de $B \longrightarrow C$ et $C \longrightarrow A$.
D'où on obtient une famille minimale équivalente à F : $\{A \longrightarrow B, B \longrightarrow C, C \longrightarrow A\}$.

Autre exemple [19] :
Soient la relation $R(A,B,C)$ et $F = \{A \longrightarrow BC, B \longrightarrow C, A \longrightarrow B, AB \longrightarrow C\}$

On applique l'étape 1 : $F_0 = \{A \longrightarrow B, A \longrightarrow C, B \longrightarrow C, AB \longrightarrow C\}$.

Étape 2 :
$F_1 = F_0 \setminus \{A \longrightarrow C\}$ car $A \longrightarrow C$ peut être déduite par transitivité de $A \longrightarrow B$ et $B \longrightarrow C$.
F_1 est équivalente à F.

On pose $F'_0 = F_1$
$F'_1 = F'_0 \setminus \{AB \longrightarrow C\}$ car $B \subset AB$ et que $F'_0 \setminus \{AB \longrightarrow C\} \cup \{B \longrightarrow C\}$ est équivalente à F'_0.

D'où on obtient une famille minimale équivalente à F : $\{A \longrightarrow B, B \longrightarrow C\}$

Dernier exemple :
Soient la relation $R(A,B,C)$ et $F = \{AB \longrightarrow C, A \longrightarrow B\}$

Toutes les dépendances sont sous forme canonique (un seul attribut à droite) donc la famille F vérifie la propriété 1 d'une famille minimale.

Il n'est pas possible d'enlever une dépendance et de la retrouver par transitivité à partir de celle restante, donc la famille F vérifie la propriété 2 d'une famille minimale.

Appliquons donc l'étape 3 de l'algorithme. Peut-on diminuer la partie gauche de la dépendance $AB \longrightarrow C$?
Peut-on déduire $A \longrightarrow C$ ou $B \longrightarrow C$ de F ?

Il n'est pas possible de déduire $B \longrightarrow C$ de F car aucune dépendance de F n'est de la forme $B \longrightarrow$.
En revanche :
$A \longrightarrow B$ appartient à F
Par augmentation par A on a : $A \longrightarrow AB$ et par transitivité avec $AB \longrightarrow C$, on en déduit donc que $A \longrightarrow C$.

Donc $A \longrightarrow C$ appartient à F^+.
On peut donc remplacer $AB \longrightarrow C$ par $A \longrightarrow C$.

On a donc $F' = \{A \longrightarrow C; A \longrightarrow B\}$ minimale et équivalente à F.

NB : F' est bien équivalente à F car à partir de $A \longrightarrow C$ par augmentation par B on a : $AB \longrightarrow BC$ et par décomposition on en déduit bien $AB \longrightarrow C$. Donc $AB \longrightarrow C$ peut-être déduite de F'.

8.5 Clé

8.5.1 Définition

Soit $R(A_1, A_2, ..., A_n)$ une relation et F une famille de dépendances fonctionnelles sur R.

Un sous-ensemble de $\{A_1, A_2, ..., A_n\}$, noté X, est une **clé** de la relation R si :

1. La dépendance fonctionnelle $X \longrightarrow A_1, A_2, ..., A_n$ appartient à F^+.
2. Pour tout sous-ensemble $Y \subset X$, on n'a pas $Y \longrightarrow A_1, A_2, ..., A_n$.

8.5.2 Surclé

Il peut exister plusieurs clés pour une même relation. Si X n'est pas un ensemble minimal (ne vérifie pas la deuxième condition), on dit que X est une **surclé**.

8.5.3 Clé primaire

Lorsqu'une relation a plusieurs clés, on en choisit une qui devient la **clé primaire**.

8.5.4 Comment déterminer une clé de relation ?

Les dépendances fonctionnelles permettent de déterminer les clés des relations. Pour cela, on a plusieurs règles. Soit R une relation à laquelle est associée une famille de dépendances fonctionnelles F :

— Tout attribut qui n'appartient à aucune DF de F appartient à toutes les clés minimales de R.

— Tout attribut qui apparaît qu'à gauche des DF de F appartient à toutes les clés minimales de R.

— Tout attribut qui apparaît qu'à droite des DF de F n'appartient à aucune clé minimale de R.

— On ne peut rien déduire des attributs qui apparaissent à droite de certaines DF de F et à gauche d'autres. Il faut calculer leur fermeture.

Par exemple, soient $R(A, B, C)$ une relation et $F = \{A \longrightarrow B, B \longrightarrow C\}$.
A partir de l'attribut A, il est possible de déterminer les attributs B (par la dépendance fonctionnelle $A \longrightarrow B$) et C (par transitivité des deux dépendances fonctionnelles de F).
En conclusion, A est une clé minimale. Tout ensemble d'attributs contenant A est une surclé.

Autre exemple : Soient $R(A, B, C, D)$ et $F = \{A \longrightarrow B, B \longrightarrow C\}$.
Dans ce cas, (AD) est clé, car l'attribut D n'est déterminé par aucun autre attribut de R.

Bibliographie

[1] D. Austin, *Using Oracle8^{TM}*, Simple Solutions - Essential Skills, QUE, 1998, ISBN : 0-7897-1653-4

[2] R. Chapuis, *Oracle 8*, Editions Dunes et Laser, 1998, ISBN : 2-913010-07-5

[3] P. Chen, *The Entity-Relationship Model–Toward a Unified View of Data*, ACM Transactions on Database Systems, Vol. 1, No. 1, March 1976, Pages 9-36, http ://www.csc.lsu.edu/∼chen/pdf/erd.pdf

[4] T. Connolly, C. Begg et A. Strachan, *Database Systems - A pratical Approach to Design, Implementation and Management*, Addison-Wesley, 1998, ISBN : 0-201-34287-1, disponible à la BU 055.7 CON

[5] C.J. Date, *Introduction aux bases de données*, 6ème édition, Thomson Publishing, 1998, ISBN : 2-84180-964-1, disponible à la BU 005.7 DAT

[6] R. Elamsri et S.B. Navathe, *Fundamentals of Database Systems*, 3ème édition, Addison Wesley- disponible à la BU 005.7 ELM

[7] P. Delmal, *SQL2 - Application à Oracle, Access et RDB*, 2ème édition, Bibliothèque des Universités - Informatique, De Boeck Université, 1988, ISBN : 2-8041-2995-0,disponible à la BU 005.74 SQL

[8] S. Feuerstein, B. Pribyl et C. Dawes, *Oracle PL/SQL - précis et concis*, O'Reilly, 2000, ISBN : 2-84177-108-3

[9] G. Gardarin, *Bases de Données - objet & relationnel*, Eyrolles, 1999, ISBN : 2-212-09060-9, disponible à la BU 005.74 GAR

[10] R. Grin, *Introduction aux bases de données, modèle relationnel*, Université Sophia-Antipolis, jan. 2000

[11] R. Grin, *Langage SQL*, Université Sophia-Antipolis, jan. 2000

[12] G. Gardarin et O. Gardarin *Le Client-Serveur*, Eyrolles, 1999, disponible à la BU

[13] H. Garcia-Molina, J.D. Ulmann et J. Widow, *Database SYstem Implementation*, Prentice Hall, 2000, ISBN :0-13-040264-8, disponible à la BU 005.7 GAR

[14] H. Garcia-Molina, J.D. Ulmann et J. Widow, *Database Systems - The Complete Book*, Prentice Hall, 2002, ISBN :0-13-031995-3

[15] S. Krakowiak, *Gestion Physique des données*, Ecole Thématique "Jeunes Chercheurs" en Base de Données, Volume 2, Port Camargue, mars 1997

[16] D. Lockman, *Oracle8^{TM} Dévéloppement de bases de données*, Le programmeur - Formation en 21 jours, Editions Simon et Schuster Macmillan (S&SM), 1997, ISBN : 2-7440-0350-6, disponible à la BU 005.74 ORA

[17] P.J. Pratt, *Initiation à SQL - Cours et exercices corrigés*, Eyrolles, 2001, ISBN : 2-212-09285-7

[18] R. Ramakrishnan et J. Gehrke, *Database Management Systems*, Second Edition ; McGraw-Hill, 2000, ISBN : 0-07-232206-3, disponible à la BU 055.7 RAM

[19] A. Silberschatz, H.F. Korth et S. Sudarshan, *Database System Concepts*, Third Edition, McGraw-Hill, 1996, ISBN : 0-07-114810-8, disponible à la BU 005.7 DAT

[20] C. Soutou, *De UML à SQL - Conception de bases de données*, Eyrolles, 2002, ISBN : 2-212-11098-7

[21] J.D. Ullman et J. Widom, *A first Course in Database Systems*, Prentice Hall, 1997, ISBN : 0-13-887647-9, disponible à la BU 005.7 ULL