Bases de données relationnelles - 2

S8 - BDTN 2025

2

Dépendances Fonctionnelles et Formes normales

Tables et contraintes – clé primaire, contrainte d'intégrité référentielle, autres contraintes

Dictionnaire de données : lister les contraintes, les désactiver

Révisions SQL : le partitionnement

Formes normales

Dépendance Fonctionnelle

Met en évidence les liens qui existent entre les données

Dégager les dépendances fonctionnelles est important pour élaborer le modèle conceptuel de la base de données

Définition

R est le schéma d'une relation,

A et B sont des ensembles d'attributs de R

On dit que :

- B est fonctionnellement dépendant de A
- Ou A détermine fonctionnellement B

Si à chaque valeur de A, correspond au plus une valeur de B

On l'écrit : A → B

Dépendance Fonctionnelle : exemples

La connaissance du numéro de client permet de connaître sans ambiguïté le nom du client

- L'identifiant du client détermine fonctionnellement le nom de famille du client
- cst_id → cst_last_name

La connaissance du titre du spectacle permet de connaître la durée du spectacle

- Le titre du spectacle détermine fonctionnellement la durée du spectacle
- shw_title → shw_duration

nomProj	idEmp	heures	nomEmp	budget	dateDebut	salEmp	mgrProj	evalEmp
ILO	E101	25	Durand	100000	15/11/2019	45000	E085	9
ILO	E105	39	Adam	100000	15/11/2019	43000	E085	
ILO	E110	10	Rivera	100000	15/11/2019	41000	E085	8
MAXI	E110	29	Rivera	200000	03/01/2021	41000	E155	

•Chaque projet a un nom unique (attribut nomProj) , il est décrit par son budget, sa date de début, l'identifiant du chef de projet (mgrProj)

nomProj → budget, dateDebut, mgrProj

•Les noms des employés et des managers ne sont pas uniques

•Un employé est identifié par la valeur de son identifiant (idEmp), il est décrit par son nom, son salaire

idEmp → nomEmp, salEmp

•Plusieurs employés peuvent travailler sur chaque projet, et un employé peut travailler sur plusieurs projets. La colonne heures indique le nombre d'heures par semaine qu'un employé donné doit travailler sur un projet donné

nomProj, idEmp → heures

•evalEmp donne l'évaluation obtenue par l'employé sur un projet donné. Cette évaluation est donnée par le manager du projet

nomProj, idEmp → evalEmp

Anomalies à éviter

Lors de la conception d'une base de données relationnelle, on chercher à éviter :

- La redondance
- Les anomalies suivantes:
  - *Anomalie de mise à jour* : une occurrence d'un fait est modifiée, mais pas toutes les occurences
  - *Anomalie de suppression : un fait valide est perdu lorsqu'un tuple est supprimé*

Anomalies

nomProj	idEmp	heures	nomEmp	budget	dateDebut	salEmp	mgrProj	evalEmp
ILO	E101	25	Durand	100000	15/11/2019	45000	E085	9
ILO	E105	39	Adam	100000	15/11/2019	43000	E085	
ILO	E110	10	Rivera	100000	15/11/2019	41000	E085	8
MAXI	E110	29	Rivera	200000	03/01/2021	41000	E155	

Première forme normale

Une relation est en première forme normale si et seulement si :

- *Tout attribut a une valeur et une seule pour chaque tuple (atomicité)*

nomProj	idEmp	budget	dateDebut	mgrProj
ILO	{ E101 E105, E110 }	100000	15/11/2019	E085
MAXI	E110	200000	03/01/2021	E155

Première forme normale

Une relation est en première forme normale si et seulement si :

- *Tout attribut a une valeur et une seule pour chaque tuple (atomicité)*

	nomProj	idEmp	budget	dateDebut	salEmp	mgrProj
	ILO	E101	100000	15/11/2019	45000	E085
	ILO	E105	100000	15/11/2019	43000	E085
	ILO	E110	100000	15/11/2019	41000	E085
	MAXI	E110	200000	03/01/2021	41000	E155

nomProj	idEmp	heures	nomEmp	budget	dateDebut	salEmp	mgrProj	evalEmp
ILO	E101	25	Durand	100000	15/11/2019	45000	E085	9
ILO	E105	39	Adam	100000	15/11/2019	43000	E085	
ILO	E110	10	Rivera	100000	15/11/2019	41000	E085	8
MAXI	E110	29	Rivera	200000	03/01/2021	41000	E155	

nomProj → budget, dateDebut, mgrProj

idEmp → nomEmp, salEmp

nomProj, idEmp → heures

nomProj, idEmp → evalEmp

Clé
(nomProj, idEmp)

Projet(#nomProj, #idEmp, heures, nomEmp, budget,
dateDebut, salEmp, mgrProj, evalEmp)

Deuxième forme normale

Une relation est en 2ième forme normale si :

- *elle est en première forme normale*
- *les attributs non-clé dépendent de TOUTE la clé*

nomProj	idEmp	heures	nomEmp	budget	dateDebut	salEmp	mgrProj	evalEmp
ILO	E101	25	Durand	100000	15/11/2019	45000	E085	9
ILO	E105	39	Adam	100000	15/11/2019	43000	E085	
ILO	E110	10	Rivera	100000	15/11/2019	41000	E085	8
MAXI	E110	29	Rivera	200000	03/01/2021	41000	E155	

Deuxième forme normale

Une relation est en 2ième forme normale si :

- Elle est en 1ère forme normale
- Les attr. non-clé dépendent de TOUTE la clé

Dépendances Fonctionnelles

1. nomProj → budget, dateDebut, mgrProj
2. idEmp → nomEmp, salEmp
3. nomProj, idEmp → heures
4. nomProj, idEmp → evalEmp

Clé

- Clé candidate : (nomProj, idEmp)
- Clé primaire : (nomProj, idEmp)
- Attributs non-clé : budget, dateDebut, mgrProj, salEmp, heures, evalEmp

Validation de la 2ème FN : DF1 et DF2 => la table n'est pas en 2ème Forme Normale

Résolution : Deuxième forme normale

1. nomProj → budget, dateDebut, mgrProj
2. idEmp → nomEmp, salEmp
3. nomProj, idEmp → heures
4. nomProj, idEmp → evalEmp

DF1 => Table Projet

nomProj	budget	DateDebut	mgrProj
ILO	100000	15/11/2019	E085
ILO	100000	15/11/2019	E085
ILO	100000	15/11/2019	E085
MAXI	200000	03/01/2021	E155

Projet(#nomProj, budget, dateDebut, mgrProj)

DF2 => Table Emp

idEmp	nomEmp	salEmp
E101	Durand	45000
E105	Adam	43000
E110	Rivera	41000
E110	Rivera	41000

Emp(#idEmp, nomEmp, salEmp)

DF3 et DF4 => Table Détails

nomProj	idEmp	heures	evalEmp
ILO	E101	25	9
ILO	E105	39	
ILO	E110	10	8
MAXI	E110	29	

Détails(#idEmp=>Emp,
#nomEmp=>Projet, heures, evalEmp)

Troisième forme normale

Une relation est en troisième forme normale si :

- *elle est en deuxième forme normale*
- *tout attribut n'appartenant pas à **une** clé ne dépend QUE de la clé*

Num_immat	Marque	Modèle	Num_chassis
FZ 567 KL	Renault	Clio	
FT 777 PY	Peugeot	107	VPF35TYG3F 78945612
BB 444 JKI	Citroën	C3	
ER 876 MZ	Peugeot	108	

- Voiture(num_immat, num_chassis, modèle, marque)

- Num_immat identifie un véhicule de façon unique
- Num_chassis identifie un véhicule de façon unique
- Le modèle du véhicule détermine la marque

- modèle → marque
- num_immat → num_chassis, modèle, marque
- num_chassis → num_immat, modèle, marque

- 2 Clés candidates : Num_immat et Num_chassis
- 1 Clé primaire choisie parmi les 2 clés candidates :
- Num_immat

- Attributs non-clé : modèle , marque

Résolution

Table Voiture

Clé primaire : num_immat

Clé étrangère : modèle

Num_immat	Modèle	Num_chassis
FZ 567 KL	Clio	
FT 777 PY	107	VPF35TYG3F 78945612
BB 444 JKI	C3	
ER 876 MZ	108	

Table Marque_voiture

Clé primaire : Modèle

Modèle	Marque
Clio	Renault
107	Peugeot
C3	Citroën
108	Peugeot

Propriétés de la décomposition en 3ème forme normale

Toute relation a au moins une décomposition en troisième forme normale

1. qui est sans perte d'information
2. qui préserve les dépendances fonctionnelles

Formes normales

Toujours réaliser une conception qui respecte les 3 premières Formes normales

Sauf si…

Le coût de la décomposition est trop important

Tables respectant les trois 1ères Formes Normales

nomProj	budget	DateDebut	mgrProj
ILO	100000	15/11/2019	E085
ILO	100000	15/11/2019	E085
ILO	100000	15/11/2019	E085
MAXI	200000	03/01/2021	E155

idEmp	nomEmp	salEmp
E101	Durand	45000
E105	Adam	43000
E110	Rivera	41000
E110	Rivera	41000

nomProj	idEmp	heures	evalEmp
ILO	E101	25	9
ILO	E105	39	
ILO	E110	10	8
MAXI	E110	29	

Table avant décomposition

nomProj	idEmp	heures	nomEmp	budget	dateDebut	salEmp	mgrProj	evalEmp
ILO	E101	25	Durand	100000	15/11/2019	45000	E085	9
ILO	E105	39	Adam	100000	15/11/2019	43000	E085	
ILO	E110	10	Rivera	100000	15/11/2019	41000	E085	8
MAXI	E110	29	Rivera	200000	03/01/2021	41000	E155	

Tables et contraintes

Exemples

```
42  CREATE TABLE booking_bkg(
43      bkg_id NUMBER(20)GENERATED BY DEFAULT AS IDENTITY,
44      bkg_date DATE DEFAULT CURRENT_DATE CONSTRAINT nn_bkg_date NOT NULL,
45      bkg_total_seat NUMBER(1) CONSTRAINT ck_total_seat CHECK(bkg_total_seat > 0),
46      bkg_cst_id NUMBER(20) CONSTRAINT nn_bkg_cst_id NOT NULL,
47      bkg_shw_id NUMBER(20) CONSTRAINT nn_bkg_shw_id NOT NULL,
48      bkg_tpr_id NUMBER(20)CONSTRAINT nn_bkg_tpr_id NOT NULL,
49        CONSTRAINT pk_bkg PRIMARY KEY(bkg_id));
50
```

```
57  ALTER TABLE booking_bkg ADD CONSTRAINT fk_bkg_cst_id
58      FOREIGN KEY(bkg_cst_id) REFERENCES customer_cst(cst_id);
59  ALTER TABLE booking_bkg ADD CONSTRAINT fk_bkg_shw_id
60      FOREIGN KEY(bkg_shw_id) REFERENCES show_shw(shw_id);
61  ALTER TABLE booking_bkg ADD CONSTRAINT fk_bkg_tpr_id
62      FOREIGN KEY(bkg_tpr_id) REFERENCES type_price_tpr(tpr_id);
```

Principaux types Oracle

Type oracle	Signification/exemple
VARCHAR2(size)	Chaine de caractères de taille variable – le nombre de caractères doit être précisé - max : 4000
NUMBER(precision, scale)	Valeurs numériques Précision : de 1 à 38 (nombre de chiffres au total) Scale : de -84 à 127 (nombre de chiffres après la virgule) 123,89 => NUMBER(6,2) => 123,89 123,89 => NUMBER(6,1) => 123,9 123,89 => NUMBER(6,-2) => 100
CHAR(size)	Chaine de caractères de taille fixe – le nombre de caractères doit être précisé - max : 2000

Principaux types Oracle

Type oracle	Signification/exemple
CLOB	Character Large OBject
BLOB	Binary Large OBject
DATE	Du 1 janvier 4712 avant JC au 31 décembre 9999 ap JC Contient les champs Jour/mois/année – heure/minute/seconde Ne gère pas les fractions de secondes
TIMESTAMP(fractional_seconds)	Contient les champs Jour/mois/année – heure/minute/seconde Gère les fractions de secondes Fractional_seconds_precision : de 0 à 9, par défaut 6

Désactivation et activation des contraintes

Désactivation des contraintes:

```
ALTER TABLE nom_table DISABLE CONSTRAINT nom_contrainte;
```

Activation des contraintes:

```
ALTER TABLE nom_table ENABLE CONSTRAINT nom_contrainte
```

Partitionnement

MO_ID	MO_TITLE	MO_YEAR	MO_GENRE	MO_LENGTH	MO_ST_ID	MO_DIRECTOR	MO_ORIGINAL
6	Indiana Jones and the temple of doom	1984	action-adventure	118	3	4	5
8	King Kong	2005	action-adventure	188	-	-	-
7	Indiana Jones and the last crusade	1989	action-adventure	127	3	4	6
9	King Kong	1933	action-adventure	105	-	-	-
5	Indiana Jones and the raiders of the lost ark	1981	action-adventure	115	3	4	-
1	12 angry men	1957	drama	96	-	3	-
2	Gone with the wind	1939	epic romance	224	-	-	-
4	Titanic	1997	epic romance	194	1	1	-
3	Star wars	1977	space opera	121	-	-	-
10	The Empire strikes back	1980	space opera	124	-	-	3

10 rows returned in 0,01 seconds CSV Export

COUNT(*)	MO_GENRE
2	epic romance
2	space opera
1	drama
5	action-adventure

4 rows returned in 0,00 seconds

GROUP BY

```
SELECT COUNT(*), mo_genre
FROM movie
GROUP BY mo_genre;
```

Sélection des groupes

Calculer le nombre de films par genre pour les groupes comportant au moins 2 valeurs

```
SELECT COUNT(*), mo_genre
FROM movie
GROUP BY mo_genre
HAVING COUNT(*) >= 2;
```

COUNT(*)	MO_GENRE
2	epic romance
2	space opera
5	action-adventure

3 rows returned in 0,00 seconds

La forme générale d'une requête utilisant une fonction de groupe

```
SELECT        colonne, fonction_g
FROM          table
[WHERE        condition]
[GROUP BY     colonne]
[HAVING       condition]
[ORDER BY     colonne];
```

GROUP BY plusieurs colonnes

```
SELECT mo_genre, ms_name,
   ROUND(AVG(mo_length),2), COUNT(*)
FROM moviestar
 INNER JOIN starsin
   ON ms_id = si_ms_id
 INNER JOIN movie
   ON mo_id = si_mo_i
GROUP BY mo_genre,
         ms_name;
```

MO_GENRE	MS_NAME	ROUND(AVG(MO_LENGTH),2)	COUNT(*)
epic romance	Leonardo DI CAPRIO	194	1
drama	Henry FONDA	96	1
space opera	Harrison FORD	121	1
epic romance	Kate WINSLET	194	1
action-adventure	Harrison FORD	140,33	3
epic romance	Vivien LEIGH	224	1
action-adventure	Sean CONNERY	118	1
action-adventure	Naomie WATTS	188	1

Séquences

Séquence

Clés artificielles

Générer automatiquement des valeurs numériques uniques

Une séquence

- Est un objet de la base de données
- Est créé indépendamment des tables
- Peut être partagé par plusieurs utilisateurs

```
CREATE TABLE moviestar (
 ms_id NUMBER(4) GENERATED BY DEFAULT AS IDENTITY (START WITH 10 INCREMENT BY 1),
 ms_name VARCHAR2(20),
 ms_address VARCHAR2(20),
 ms_gender CHAR(1) DEFAULT 'F'
            CONSTRAINT ck_ms_gender CHECK (ms_gender IN ('M', 'F')),
 ms_birthdate DATE,
 CONSTRAINT pk_moviestar PRIMARY KEY (ms_id)
);

INSERT INTO moviestar(ms_name, ms_birthdate)
VALUES(   'Kate WINSLET',
            TO_DATE('05/10/1975', 'DD/MM/YYYY'));

SELECT ms_id, ms_name, ms_gender
FROM moviestar;

SELECT sequence_name
FROM user_sequences;
```

Colonne IDENTITY
=> Oracle crée une séquence

A l'insertion la colonne est omise et la valeur est générée

Résultat du SELECT :

```
--------- ------------------ -
       10 Kate WINSLET       F
```

Résultat du SELECT :

```
----------------------------
ISEQ$$_75411
```