MERISE

ANNEXES

1 - Les douze règles de CODD

LES DOUZE REGLES DU MODELE RELATIONNEL, ENONCEES PAR CODD

• **Règle nº 1. Représentation des informations :** *All information in a relational database is represented explicitly at the logical level and in exactly one way – by values in tables.* Les informations sont représentées au niveau logique et non physique, ce qui signifie que l'on ne se préoccupe pas de l'implémentation réelle des données. Elles sont décrites par des valeurs contenues dans des tables (ces tables sont également nommées relations).

• **Règle nº 2. Accès aux données :** *Each and every atomic value in a relational database is guaranteed to be logically accessible by resorting to a combination of table name, primary key value, and column name.* Une donnée est accessible logiquement – c'est-à-dire sans connaissance de son implantation physique – grâce à la combinaison du nom de table (relation), de la clé primaire et du nom de la colonne (attribut).

• **Règle nº 3. Gestion des valeurs absentes :** *Null values are supported in fully relational DBMS for representing missing information and inapplicable information in a systematic way (independant of data type).* Cette règle précise que la valeur NULL – correspondant à l'absence d'information – est interprétable et de nature différente d'une chaîne de caractères vide (ou composée de caractères « blancs ») ou d'une valeur numérique égale à zéro par exemple.

• **Règle nº 4. Le dictionnaire de données :** *The database description is represented at the logical level just like ordinary data, so that authorized users can apply the same relational language to its interrogation as they apply to regular data.*

La description de la base de données est représentée par des informations accessibles comme s'il s'agissait de données ordinaires. Le langage relationnel permet donc de manipuler indifféremment des données du système d'information ou des données décrivant la base elle-même. La description de la base est donc stockée dans des tables faisant partie de ce que l'on nomme le dictionnaire de données.

• **Règle nº 5. Le langage :** *A relational system may support at least one language which is comprehensive in supporting ALL of the following items : data definition, view definition, data manipulation, integrity constraints, authorization, transaction boundaries.* Le SGBD doit inclure au moins un langage comportant l'ensemble des fonctionnalités suivantes : définition des données, définition des vues, manipulation des données, contraintes d'intégrité, autorisations, gestion des transactions. (Le langage SQL répond à cette description.)

• **Règle nº 6. La mise à jour à travers une vue :** *All views that are theoretically updatable are also updatable by the system.* Une vue est un mode de représentation logique de la base de données. Si une vue peut être mise à jour, elle peut aussi l'être par le système.

• **Règle nº 7. La mise à jour des tables :** *The capability of handling a base relation or a derived relation as a single operand applies not only to the retrieval of data but also to the insertion, update and deletion of data.* Le langage relationnel doit disposer d'ordres de haut niveau s'appliquant non seulement à la lecture des données, mais aussi à la création, la mise à jour ou la suppression d'informations.

• **Règle nº 8. L'impédance physique :** *Application programs and terminal activities remain logically unimpaired whenever any changes are made in either storage representations or access methods.* Les programmes d'applications et les transactions interactives sont indépendants de la représentation physique des données et des méthodes d'accès sous-jacentes. (Cela garantit la souplesse d'évolution du système d'information et exige que le SGBD dissocie efficacement la représentation logique d'une part et les aspects d'organisation physique d'autre part.)

• **Règle nº 9. L'indépendance logique :** *Application programs and terminal activities remain logically unimpaired when information-preserving changes of any kind that theoretically permit unimpairment are made to the base tables.* Cette règle stipule que les programmes ne sont pas remis en cause lorsque des modifications – sans pertes d'informations structurelles – sont opérées sur les relations de la base. Citons, par exemple, l'éclatement d'une relation en deux relations ou, à l'opposé, la fusion de relations.

• **Règle nº 10. L'indépendance vis-à-vis des contraintes d'intégrité :** *Integrity constraints spécific to a particular database must be definable in the relational data sublanguage and storable in the catalog (not in the application programs).* Les contraintes d'intégrité, susceptibles d'évoluer dans le temps, doivent pouvoir être formulées en dehors de tout programme applicatif et être référencées dans le dictionnaire des données. Une application informatique constitue une réponse à un problème qui se pose à un moment précis. Si l'entreprise et son environnement évoluent, le système d'information doit par conséquent pouvoir s'adapter à cette évolution sans que soit remis en cause l'acquis applicatif.

• **Règle nº 11. L'indépendance vis-à-vis de la répartition des données :** *A relational DBMS has distributed independance.* L'environnement applicatif n'est pas affecté par la répartition des données – sur des sites et des supports physiques distincts – ou toute modification de cette éventuelle répartition.

• **Règle nº 12. La non-subversion :** *If a relational system has a low level (single-record-at-a-time) language, that low level cannot be used to bypass the integrity rules and constraints express in the higher level relational language (multiple-records-at-a-time).* Cette dernière règle stipule que si le système dispose d'un langage de bas niveau (C par exemple), ce langage ne peut pas contourner ou remettre en cause les contraintes de sécurité et les règles d'intégrité énoncées au plus haut niveau (par le langage SQL en l'occurrence).

2 - Les tables d'exemples d'Oracle

Table EMP

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-DEC-80	800		20
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30
7566	JONES	MANAGER	7839	02-APR-81	2975		20
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30
7698	BLAKE	MANAGER	7839□	01-MAY-81	2850		30
7782	CLARK	MANAGER	7839	09-JUN-81	2450		10
7788	SCOTT	ANALYST	7566	09-DEC-82	3000		20
7839	KING	PRESIDENT		17-NOV-81	5000		10
7844	TURNER	SALESMAN	7698	08-SEP-81	1500	0	30
7876	ADAMS	CLERK	7788	12-JAN-83	1100		20
7900	JAMES	CLERK	7698	03-DEC-81	950		30
7902	FORD	ANALYST	7566	03-DEC-81	3000		20
7934	MILLER	CLERK	7782	23-JAN-82	1300		10

Table DEPT

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

Table SALGRADE

GRADE	LOSAL	HISAL
1	700	1200
2	1201	1400
3	1401	2000
4	2001	3000
5	3001	9999

Table BUDGET

DeptNo	Budget_Annuel
10	8000
20	12000
30	15000
40	10000

Structures des tables

Table emp;

Name	Null?	Type
EMPNO	NOT NULL	NUMBER(4)
ENAME		CHAR(10)
JOB		CHAR(9)
MGR		NUMBER(4)
HIREDATE		DATE
SAL		NUMBER(7,2)
COMM		NUMBER(7,2)
DEPTNO	NOT NULL	NUMBER(2)

Table bonus;

Name	Null?	Type
ENAME		CHAR(10)
JOB		CHAR(9)
SAL		NUMBER
COMM		NUMBER

Table salgrade

Name	Null?	Type
GRADE		NUMBER
LOSAL		NUMBER
HISAL		NUMBER

Table dept

Name	Null?	Type
DEPTNO	NOT NULL	NUMBER(2)
DNAME		CHAR(14)
LOC		CHAR(13)

3 - Les commandes SQL

3-A Page 1

Commande	Paramètre	Attributs	Commentaires	Exemples
Tables				
CREATE TABLE	*NomDeTable*	(*Attr Format* [NOT NULL], *Attr Format*...) ;	Création d'une table composée des attributs spécifiés. cf §3.A les formats possibles. Le mot réservé NOT NULL empêche la validation d'une saisie si l'attribut n'est pas renseigné	CREATE TABLE Client (Code CHAR(5) NOT NULL, Nom CHAR(15) NOT NULL, Prenom CHAR(10), Age NUMBER(3), Cotis NUMBER(8,2));
ALTER TABLE	*NomDeTable*	ADD *Attr Format*; MODIFY *Attr Format*;	Modification de la structure d'une table par ajout ... ... ou par modification d'un attribut (ou colonne)	ALTER TABLE Client ADD CodePost CHAR(5); ALTER TABLE Client MODIFY Nom CHAR(20);
RENAME TABLE	*NomDeTable*	TO *NouveauNom*;	Modification du nom d'une table	RENAME TABLE Client TO Prospects;
DROP TABLE	*NomDeTable*;		Suppression d'une table (structure & saisies)	DROP TABLE Prospects;
COMMENT ON TABLE	*NomDeTable*	IS *Commentaire*;	Ajout ou modification d'un commentaire sur une table	COMMENT ON TABLE Client IS 'Clients et prospects, y compris sans factures';
COMMENT ON COLUMN	*NomDeColonne*	IS *Commentaire*;	Ajout ou modification d'un commentaire sur une colonne	COMMENT ON COLUMN CodePostal IS 'Code postal de la ville du siège social du client';
Vues				
CREATE VIEW	*NomDeTable*	AS *Requête*;	Définition d'une table externe constituée à partir d'une requête sur un ensemble de tables et de vues	CREATE VIEW BdR AS (SELECT Nom, CodePost FROM Client WHERE CodePost LIKE '13%');
DROP VIEW	*NomDeTable*;		Suppression d'une vue	DROP VIEW BdR;
Saisies				
INSERT INTO	*NomDeTable* *NomDeTable (Attr, Attr,...)*	VALUES (*Valeur, Valeur, ...*);	Crée une (ou plusieurs, cf plus bas) ligne dans la table, en renseignant les colonnes citées avec les valeurs fournies dans l'ordre d'écriture. Les autres colonnes sont à NULL	INSERT INTO Client (Nom, Age, CodePostal) VALUES (DUPONT, 23, 13090);
UPDATE	*NomDeTable*	SET *Attr* = *Expr* *[, Attr = Expr..]* WHERE *Cond*;	Met à jour un (des) attribut(s) des lignes de la table *NomDeTable* pour lesquelles la condition *Cond* est vérifiée avec la valeur de *Expr* (Expr peut être NULL)	UPDATE CLient SET CodePostal=NULL WHERE Nom='DUPONT'
DELETE FROM	*NomDeTable*	WHERE *Cond*;	Supprime les lignes de la table *NomDeTable* pour lesquelles la condition *Cond* est vérifiée	DELETE FROM Client WHERE Nom IS NULL
Index				
CREATE INDEX	*NomDeIndex*	ON *NomDeTable* (*Att,Att..*);	Crée un index sur la table, sur la concaténation des attributs	CREATE INDEX INom ON Client (Nom, Prenom)
CREATE UNIQUE INDEX	*NomDeIndex*	ON *NomDeTable* (*Att,Att..*);	idem, mais en empêchant les doublons	CREATE UNIQUE INDEX INom ON Client (Code)
DROP INDEX	*NomDeIndex*;		Supprime l'index spécifié	DROP INDEX INom;
Transactions				
AUTOCOMMIT	*On/Off*;		Choix du mode "Transaction automatique" (validation à chaque INSERT, UPDATE ou DELETE) ou en mode "Transaction manuelle" (via COMMIT ou ROLLBACK). Si AUTOCOMMIT est à OFF,COMMIT (Resp: ROLLBACK) valide (resp: annule) toutes modifications apportées dans les tables depuis le dernier COMMIT ou ROLLBACK	
COMMIT				
ROLLBACK				

3-B Les commandes SQL, Page 2

Commande	Paramètre	Attributs	Commentaires	Exemples
Consultations				
SELECT [DISTINCT]	*Attr; Attr;.../ **</i>	FROM *NomTable* [*Alias*],*NomTable*... WHERE *Cond* [GROUP BY *Attr2*, *Attr2*... [HAVING *Cond*]] [ORDER BY *Attr3*];	Réalise une projection selon les colonnes citées (Attr peut être une expression)... en ne présentant éventuellement qu'une seule fois deux lignes semblables ... sur le produit cartésien des tables citées... restreintes selon la condition *Cond*, (éventuellement complexe), condition pouvant réaliser la jointure, en faisant éventuellement apparaître uniquement des sous-totaux correspondant à *Attr2*... sous-totaux restreints selon la condition *Cond2* triés selon l'attribut *Attr3* (peut être une expression) (cf §3.2 les opérateurs de conditions)	SELECT DISTINCT Code, Nom, Prenom,Annee, CotisAnnu FROM CLIENT, COTISATION WHERE Code=CodeCotis ORDER BY Nom, Prenom
Co-requêtes				
Requête	UNION	*Requête*	Permet la constuction d'une table (resultant d'une requête ou d'une co-requête) comme la réunion,...	SELECT Nom, Prenom FROM Client WHERE Nom = 'DUPONT' UNION SELECT Nom, Prenom FROM Prospect WHERE Nom = 'DUPONT''
Requête	INTERSECT	*Requête*	... l'intersection...	
Requête	MINUS	*Requête*	... ou la différence de deux autres tables	
Connexions - Permissions				
CONNECT	*User / Passe*		Se connecter au système d'informations	
GRANT SELECT INSERT UPDATE DELETE ALTER INDEX ALL	ON *NomDeTable*	TO *NomUser* [WITH GRANT OPTION];	Permettre à l'utilisateur spécifié les accès à la table en consultation... ...et/ou ajout... ...et/ou modification... ...et/ou suppression... ...et/ou modification de structure... ...et/ou création/suppression d'index... ... ou tout à la fois... ...et droit de transmettre ces autorisations à d'autres utilisateurs	GRANT SELECT, INSERT, INDEX ON Prospects TO Dupont WITH GRANT OPTION
REVOKE SELECT,INSERT,...	ON *NomDeTable*	TO *NomUser*		REVOKE INDEX ON Prospects TO Dupont
Synonymes				
CREATE SYNONYM	*NomDeTable*	FOR *NomUser*.*NomDeTable*;	Crée un synonyme pour la table d'un autre utilisateur	CREATE SYNONYM DClient FOR Dupont.Client
DROP SYNONYM	*NomDeTable*;		Inverse de la précédente	DROP SYNONYM DClient;

4 - Les fonctions SQL

Fonction	Paramètre	Commentaires	Exemples
Fonctions synthétiques : appliquées à une projection d'un SELECT, elles fournissent une table à une seule ligne (et éventuellement une seule colonne si une seule fonction est appelée. *NB : leur coimprtement peut être enrichi afin de'obtenir des états récapitulatifs avec sous-totaux si le SELECT est associé à une clause GROUP BY*)			
SUM	(*Attr*)	Renvoie la somme des valeurs de la colonne spécifiée	SELECT SUM(Cotis), MAX(Cotis) FROM Disque WHERE CodePostal='13300'
MIN	(*Attr*)	… la plus petite…	
MAX	(*Attr*)	… la plus grande…	
AVG	(*Attr*)	… la moyenne…	
STDDEV *(Oracle)*	(*Attr*)	… la variance de la série…	
VARIANCE *(Oracle)*	(*Attr*)	… la l'écart-type de la série…	
COUNT	(*Attr*) | (*)	… lae nombre de lignes renseignées…	SELECT COUNT(*) FROM Disque : renvoie le nombre de lignes non entièrement NULL
Autres fonctions : similaires aux fonctions standard des langages de programmation classique			
Conversion			
TO_NUMBER	(*CHAR*)	Le 1° caractère de la chaîne doit être "+", "-" ou un chiffre	
TO_CHAR	(*Number*)	*No comment*	
TO_DATE *(Oracle)*	(*CHAR*)		
Numeriques			
ABS	(*Nombre*)		
POWER	(*Nb 1, Nb2*)		
ROUND	(*Nombre*)		
SQRT	(*Nombre*)		
Chaînes			
(Chaîne) (Chaîne, Nb1, Nb2)			
LENGTH	(*CHAR*)	Longueur d'une chaîne	
INITCAP	(*CHAR*)	Mise en majuscule de la 1° lettre de la chaîne	
UPPER	(*CHAR*)	Mise en majuscule de la chaîne	
SUBSTR	(*CHAR,Nb1,Nb2*)	Extraction de chaîne	
Dates			
ADD_MONTHS	(*Date, Nb*)	Renvoie la date obtenue an ajoutant *Nb* mois à *Date*	
MONTHS_BETWEEN	(*Date1, Date2*)	Renvoie le nombre de mois compris entre *Date1* et *Date2*	
Variables système			
USER *(Oracle)*		Renvoie le nom de l'utilisateur déclaré dans le CONNECT	
SYSDATE *(Oracle)*		Renvoie la date-système	

5 - Les opérateurs SQL

5-A/ Formats des attributs (ou colonnes) :

CHAR (x) : chaîne de longueur variable, longueur maximale de x caractères. x doit être <= 240
LONG : chaîne de longueur variable, longueur maximale de 65535 caractères*(Oracle)*
NUMBER : Nombre entier ou fractionnaire de 40 chiffres au maximum *(Oracle)*
NUMBER (x): Nombre entier ou fractionnaire de x chiffres au maximum *(Oracle)*
NUMBER (x,y): Nombre fractionnaire de x chiffres au maximum, avec y chiffres décimaux*(Oracle)*
DATE : Date au format 'JJ-MMM-AA' *(Oracle, non ANSI)*
NB : certains SGBD utilisent plutôt une représentation proche du codage machine, ie : INTEGER, REAL, DOUBLE, BYTE…

5-B/ Conditions dans SQL :

COMPARAISONS :	*Expr* = *Expr*	Opérateurs de comparaison classiques.
	Expr > *Expr*	Pour les chaînes de caractères : Table ASCII
	Expr < *Expr*	
	Expr != *Expr*	
	Expr >= *Expr*	
	Expr <= *Expr*	
MASQUES :	*Expr* [NOT] LIKE *Masque*	*Masque* : chaîne de caractères avec jokers: '_' (équivalent à '?' sous DOS) '%' (équivalent à '*' sous DOS)
FOURCHETTES :	*Expr* [NOT] BETWEEN *Expr* AND *Expr*	
OPERATEURS :	NOT	Opérateurs booléens classiques
	AND	
	OR	
ENUMERATIONS :	*Expr* [NOT] IN (*Expr, Expr, …*)	
	Expr [NOT] IN (*sous-requête*)	
	Expr Comp ANY (*Expr, Expr, …*)	IN est équivalent à =ANY
	Expr Comp ANY (*sous-requête*)	
	Expr Comp ALL (*Expr, Expr, …*)	NOT IN est équivalent à !=ALL
	Expr Comp ALL (*sous-requête*)	
NULL :	*Expr* IS [NOT] NULL	*Expr*=NULL est sytanxiquemen incorrect
EXISTENCE :	[NOT] EXISTS (*sous-requête*)	Retourne vrai si la sous-requête renvoie au moins une ligne

5-C/ Tables de vérité :

AND	VRAI	FAUX	NULL
VRAI	VRAI	FAUX	NULL
FAUX	FAUX	FAUX	FAUX
NULL	NULL	FAUX	NULL

OR	VRAI	FAUX	NULL
VRAI	VRAI	VRAI	VRAI
FAUX	VRAI	FAUX	NULL
NULL	VRAI	NULL	NULL

Si X est NULL, les assertions X=NULL ou X=X retournent la valeur NULL (et non pas la valeur VRAI) : est-ce que “Je ne sais pas quoi” est égal à “Je ne sais pas quoi” ? Réponse : je ne sais pas.

5-D/ Opérateurs arithmétiques & chronologiques :

ARITHMETIQUES : + - * / ^		Opérateurs arithmétiques classiques
DATES :	Date + numérique -> Date	Date postérieure au 1° opérande du nb de jours égal au 2° opérande Date+1=lendemain
	Date - numérique -> Date	Date antérieure au 1° opérande du nb de jours égal au 2° opérande Date-1=veille
	Date - Date -> numérique	Nb de joursd entre 2 dates. Bien entendu, pour un entier N, on a : (('Date+X)-Date)=X