BD Examen

# *Réponses aux questions :*



***Spécialisation*** : (p12) (représentation des instances p40)



***Intérêts*** : permet de cibler plus spécifiquement

***Représentation des valeurs distinctes de TypeAssujetti*** : (p40)



***Intérêts*** : utile dans le cas si on veut disposer des valeurs distinctes des NumTVA

1. L'utilisation d'un index simplifie et accélère les opérations de recherche, de tri, de jointure ou d'agrégation effectuées par le SGBD

Index nécessaire quand lien 0-1/1-1. Sans index, la cardinalité 0-1 serait perdue. Il faut mettre un index sur la clé étrangère de la table FacturetImp spécifiant une classe unique. Comme ça on pourra trouver toutes les Visites qui ont une FactureImp grâce à l’index.



(p93, 94)



Chemin 1



Chemin 2

On sait retrouver le représentant en passant par Client et Région. Mais pas redondant car dans commande(CDE) il y a la date donc par le premier chemin on trouve le représentant au moment de la commande. Pour le chemin deux c’est le représentant actuel de la région au moment de la commande. C’est donc deux informations différentes donc pas de redondance.

C’est souvent au niveau des dates où il faut réfléchir.

1. C’est utile lorsque des liens se font entre plusieurs tables. Les index permettent de faciliter les jointures et donc de trouver ce que l’on cherche en passant par moins de valeurs.

Exemple :

Il y a 100 000 commandes et 10 000 client, sans index, il y aura plusieurs manières de procéder, soit la manière de boucle imbriquées mais dans ce cas il faudra 100 000 \* 10 000 recherche donc 1 000 000 000, c’est beaucoup trop long. Si on fait un tri-fusion alors on aura 100 000\*log(100 000) + 10 000\*log(10 000) + 100 000 + 10 000 = 650 000 encore beaucoup. Mais si on utilise un index par exemple pour les clients alors là il ne faudra plus que 110 000 recherche car on va lire le client puis les commandes qui lui sont liée, du coup on va passer dans les 10 000 clients et lire les commandes qui lui sont liées, donc 110 000. (p95)

1. Le problème avec les techniques procédurales est que cela se situe dans le programme, du coup si on doit modifier le programme et qu’on oublie la vérification, il y aura un problème alors qu’avec les triggers et les checks, comme le code de la BD lui n’aura pas changé, il y aura toujours une vérification et donc il y aura une erreur SQL si une valeur incorrecte est entrée.
2. Fichier système contenant d’une part les valeurs(images) après modifications des pages modifiées, dans l’ordre des modifications avec les identifiants des transactions modifiantes, et d’autre part des enregistrements indiquant les débuts, validation et annulation de transactions.

Cela permet donc en cas de coupure de pouvoir récupérer ce qu’on a fait.

1. La forme normale de BOYCE-CODD est plus normalisée que la 3FN, elle élimine les dépendances fonctionnelles à l’intérieur de l’identifiant.

Exemple :

COURS (Matière, Etudiant, professeur), id= (Matière, Etudiant).

Elle est bien en 3 FN car :

* Pas de dépendances mutlivaluées
* Pas de dépendance partielle d’un attribut non-identifiant par rapport à l’identifiant
* Pas de dépendance transitive d’un attribut non identifiant par rapport à l’identifiant

Dans cet exemple c’est le professeur qui détermine la matière alors que normalement c’est l’id qui doit tout déterminer. (p79, 80, 81)

1. Le système contrôle les conflits au fur et à mesure de l’exécution. L’opération de T1 est refusée, la transaction T1 est reprise. C’est refusé car T1 est estampillé avec un tps plus petit que celui de la dernière estampillée en lecture sur le granule g et plus petit que la dernière estampille en écriture. La transaction T1 va être relancée. T1 aurait été refusé même s’il était arrivé avant la maj de T2 car le granule g a déjà été relu entre temps par T2

(P106)

Principe : Il y en a 2

* Avant toute opération sur un élément, la transaction demande un verrou sur cet élément pour cette opération
* Une fois qu’il n’y a plus besoins de l’élément, alors on lève le verrou, l’accès à l’élément est libéré. Mais une transaction ne peut pas relâcher de verrou avant de les avoir tous acquits.

verrous

verrous

verrous

temps

temps

temps

Verrouillage/déverrouillage

Schéma : Il faut retirer le verrou avant de pouvoir utiliser un élément.

Intérêt : Si toutes les transactions sont à verrouillage à deux phases, alors toutes les exécutions entremêlées de ces transactions sont sérialisables. Et donc, on pourra exécuter uniquement les opérations compatibles simultanément, donc il n’y aura pas d’anomalie.

(P107)

1. Date entrée doit être inférieur à Date sortie.
2. ***Contrainte additionnelle au niveau conceptuel*** :

C’est une contrainte d’intégrité qui, en plus, ne peut pas être représentée graphiquement dans le modèle utilisé

Exemple :

Le fait qu’un produit doit soit être composé d’une matière première et d’un produit non fini ou deux produits non fini ou encore deux matières premières. (p29)

***Limite au niveau logique ou physique*** :

On peut définir le terme limite comme un besoin fonctionnel ou une contrainte liée à un besoin fonctionnel qui de façon délibérée ne sera pas pris en compte ou pris en compte sans toutes les exigences exprimées au premier abord.

La décision de poser une limite peut se produire à chaque stade de l’élaboration de S.I. (système informatique). Elle doit toujours être communiquée clairement et acceptée par le maître de l’ouvrage.

Exemple : La multiplicité 1-N n’est pas supportée. Il faut donc la remettre à 0-N

1. P173 (2.2)
2. Plus l’identifiant est petit, plus il est restrictif tandis que s’il est grand il est moins restrictif car par exemple Si un véhicule est juste identifié par l'immatriculation, si on fait une recherche sur l'immatriculation, on ne sortira que 1 véhicule, par contre si l'identifiant est l'immatriculation et la date d'immatriculation, si on fait une recherche par rapport à la date, on aura plusieurs voiture immatriculé ce jour-là. Du coup c'est moins restrictif.



1. 

Id(AffiliationHonneur) = AFFILIATION

1. CLIENT(N°\_Cli,Adresse,NomCli)

PRODUIT(N°\_Prod,PrixUn)

COMMANDE(N°\_Com,CLIENT,PRODUIT,Date)





1. Valeurs distinctes car :

* On peut avoir la liste
* On peut réutiliser
* Si on a une envie de contrôle



1. Représentation des instances car ainsi il y aura N numéro de téléphone pour un ouvrier



Descriptif

DateDernierContract

N°Chassis

NumClient

NomMarque

NumPlaque

NomModèle

DateIm

Concessionnaire

1. Type spécifique :





Contraintes additionnelles :

Le produit doit juste être un produitFini ou un ProduitSemiFini mais pas les deux.



Contraintes additionnelles :

Un véhicule doit obligatoirement être relié à une visite.



1. L’intérêt est que ça met une contrainte d’exclusion en évidence.

***Contraintes additionnelles :***

* Si le matricule existe dans Employé, il ne doit pas exister dans Ouvrier.
* On est soit Employé, soit ouvrier mais pas les deux.



1. ***Type spécifique*** : Employé ou Ouvrier

Ce n’est pas fort utile, il n’y a pas d’intérêt à faire ça (selon vous monsieur) !

***Contraintes additionnelles :***

* Si le matricule existe dans ouvrier, il ne doit pas exister dans employé et inversement.
* Tu es soit employé soit ouvrier.



1. ***Type générique*** : Personne

Grâce à ceux-ci, il y a moins de tables

***Contraintes additionnelles*** :

* Il ne faut pas oublier de rajouter le type
* Certains attributs seront remplis et d’autre nom en fonction du type



1. ***Attribut mutlivalué*** :

***Représentation des instances*** :

id(Téléphone) : Client, NumTel

Intérêt : permet d’avoir un num de téléphone spécifique pour le client

***Représentation des valeurs distinctes*** :

Intérêt : On peut disposer des valeurs distinctes, on peut faire des recherches, on peut les réutiliser et permet le contrôle.

***Représentation par instanciation*** :

Intérêt : trop restrictive donc pas très bien à ne pas faire.

1. ***Attribut décomposable*** :

***Représentation des instances*** :



Intérêt : permet d’avoir un num de téléphone spécifique pour le client

***Représentation des valeurs*** :

Intérêt : mieux que représentation des instances.

***Représentation par désagrégation*** :

Intérêt : Permet de savoir que ces attributs viennent d’un attribut décomposable.

***Représentation par concaténation*** :

Intérêt : utile pour la performance des fois mais irréversible.

1. D



1. La commande est liée au représentant or la commande est aussi liée client qui est elle-même liée à la région qui est elle-même lié au représentant. On pourrait alors penser à une redondance car on peut trouver le représentant par deux chemins mais en réalité ce n’est pas le cas. En effet Le lien entre commande et représentant est le représentant qu’il y avait au moment de la commande car commande à un attribut date alors que l’autre lien est le représentant actuel qui n’est pas forcément le même que celui au moment de la commande. C’est donc une fausse redondance.
2. Il pourrait être un faux ternaire si par exemple il y avait plusieurs lignes pour la commande. Dans le cas ici si la ternaire est vrai alors la commande ne comporte qu’une seule ligne sinon la ternaire est fausse car il y aura pu avoir plusieurs produits pour la commande.



Avec la transformation on voit bien que la ternaire est vrai s’il y a juste une ligne pour la commande.

Identifiant : id(Commande) : Client, Magasin, Produit

1. On sait quelle opération l'ouvrier peut faire et celle qu'il ne peut pas pour pouvoir choisir l’ouvrier adéquat pour l’opération.
2. Comme ça on peut savoir si on fait des erreurs par rapport aux estimations qu'on a eu pour la création du devis et adapter en fonction de ça pour les prochains.
3. Permet d’avoir les informations sur le garage
4. Car comme ça on peut savoir si en fonction des années, il y a eu des changements ou non.



Il faut rajouter le libellé dans LangueOp pour savoir quelle traduction va avec quelle langue pour l’opération.