ENSTA IN206 - Examen de Bases de Données - 2019-2020

Durée : 1h. Tous documents autorisés.
Toutes les questions sont notées sur le même nombre de points, mais ne sont de la même complexité.
Soyez précis dans vos réponses, toute réponse non justifiée sera considérée comme fausse.

Une base de données nommée CFA est utilisée par un organisme pour gérer les informations de l'ensemble des apprentis actuellement en formation sur un emploi en entreprise. Les tables et leurs attributs sont donnés ci-dessous.

```
JOB (id_j, title_j, min_salary, max_salary)
WORKER (id_w, id_j, id_d, tutor, firstname_w, lastname_w, city_w, salary_w)
COMPANY (id_c, address_c, postcode_c, city_c)
DEPT (id_d, id_c, manager, name_d)
```

Ces tables stockent la liste des apprentis (table WORKER), la liste des types d'emplois (JOB)occupés actuellement par ces apprentis avec les bornes salariales minimales et maximales recommandées pour l'emploi, l'entreprise (COMPANY) dans laquelle travaille actuellement l'apprenti et le département de cette entreprise (DEPT) auquel il appartient. Chaque département à un responsable dont le nom est stocké (`DEPT.manager`), qui est le responsable des apprentis du département, et chaque apprenti à un tuteur (`WORKER.tutor`).

Exercice 1 : Conception – modèle Entité/Association

Q1 : En utilisant le formalisme Entités/Associations, proposer un modèle conceptuel complet (avec toutes les entités, associations, propriétés et cardinalités) ayant pu aboutir à créer ces tables. Vous considèrerez à ce stade que le nom du tuteur de l'apprenti est stocké dans `WORKER.tutor`.

Q2 : Le tuteur de l'apprenti est en fait lui aussi un apprenti. Le champ `WORKER.tutor` ne contient donc pas le nom du tuteur, mais l'identifiant d'un apprenti. Compléter le modèle conceptuel pour prendre compte cette particularité.

Q3 : L'organisme souhaite conserver l'historique des emplois successifs occupés par les apprentis. Compléter le modèle conceptuel pour supporter cette historisation.

Exercice 2 : Modèle relationnel et algèbre

Q4 : Compléter le schéma relationnel ci-dessus en indiquant les clés primaires et les clés étrangères. Justifiez vos réponses.

Q5 : Indiquer les modifications imposées au schéma relationnel ci-dessus pour prendre en compte l'historique des emplois occupés par les apprentis tel qu'introduit en question **Q3**.

Vous considèrerez les opérateurs suivants de l'algèbre relationnelle :

Union :	$R \cup S$	Produit cartésien :	$R \times S$
Différence :	$R - S$	Intersection :	$R \cap S$
Projection :	$\prod_{\text{Liste d'attributs}}(R)$	Jointure :	$R \bowtie S$
Restriction :	$\sigma_{\text{Critère}}(R)$	Division :	$R \div S$

Exprimer en algèbre relationnelle, à base des opérateurs ci-dessus, les ensembles définis en **Q6, Q7, Q8**.

Q6 : L'ensemble des apprentis (`id_w, firstname_w, lastname_w`), leur entreprise (`id_c`) et leur département (`id_d, name_d`), et le manager de l'apprenti (`manager`).

Q7 : L'ensemble des apprentis qui ne sont tuteur d'aucun autre apprenti.

Q8 : L'ensemble des entreprises qui rémunèrent tous leurs apprentis à un montant supérieur au smic (i.e., 1207€).

Exercice 3 : SQL

Q9 : Déclarer en SQL la table WORKER avec ses contraintes d'intégrité (clés primaires et étrangères).

Q10 : Ecrire la requête SQL donnant la liste des apprentis (`id_w, firstname_w, lastname_w`), leur entreprise (`id_c`), leur département (`id_d, name_d`) et leur manager (`manager`).

Q11 : Ecrire la requête SQL donnant les tuteurs et le nombre d'apprentis dont ils s'occupent (dont ils sont le tuteur direct).

Q12 : Ecrire la requête suivante sous forme d'une requête 'plate' (avec une seule clause SELECT)

```
SELECT firstname_W, lastname_W FROM WORKER
WHERE id_d IN (SELECT id_d FROM DEPT WHERE name_d = 'SALES'
AND id_c IN (SELECT id_c from COMPANY where city_C = 'Paris'));
```

Q13 : Ecrire la requête SQL donnant la liste des emplois pour lesquels certains apprentis gagnent moins que la borne minimale recommandée alors que d'autres gagnent plus que la borne maximale recommandée (pour le même type d'emploi).

Q14 : Ecrire la requête SQL donnant la liste des apprentis qui sont tuteur d'au moins un autre tuteur.

Exercice 4 : Programmation base de données

Q15 : Indiquer ce que fait le programme ci-dessous. Proposer une requête SQL produisant le même résultat qu'affiché par ce programme.

```
CREATE TABLE RESULTAT (company VARCHAR, payroll NUMBER);
DECLARE
  i NUMBER := 0;
BEGIN
FOR c IN (SELECT * from COMPANY) LOOP
  FOR w IN (SELECT * from WORKER) LOOP
    IF c.id_c = w.id_c THEN i := i+w.salary_w;
    END IF;
  END LOOP;
  INSERT INTO RESULTAT VALUES (c.name_c, i);
END LOOP;
END;
/
SELECT * FROM RESULTAT ;
```

Q16 : Quelle serait la solution la plus performante du point de vue du SGBD, la version PL/SQL ci-dessus, ou la version en SQL pur que vous avez proposez ? Justifier votre réponse.

Exercice 5 : Propriétés transactionnelles

Q17 : L'exécution présentées dans le tableau ci-dessous indique l'évolution du salaire (`Salaire_w`) de l'apprenti d'identifiant 30 (`id_w=30`) pour 2 transactions concurrentes. La séquence des commandes SQL successives traitées par le SGBD et leur résultat sont indiquées. L'exécution est-elle sérialisable ? Quel est le niveau d'isolation de la transaction 1 ? Pourquoi l'opération 4 n'est-elle pas mise en attente ? Justifier vos réponses.

	Transaction 1		Transaction 2	
	opération	résultat	opération	résultat
Etat initial	Salaire de 'Joe' = 1300			
Opération 1	Commit ;			
Opération 2			Commit ;	
Opération 3	SEL	1300		
Opération 4			UPD	1 raw updated
Opération 5			SEL	1400
Opération 6	SEL	1400		
Opération 7			Commit ;	
Opération 8	SEL	1200		
Opération 9	Commit ;			

Avec: `UPD = UPDATE WORKER SET salaire_w=salaire_w + 100 WHERE id_w=30;`
`SEL = SELECT salaire_w FROM WORKER WHERE id_w=30;`

Q18 : En supposant que les deux transactions sont lancées au niveau d'isolation « serialisable » et sans protocole multi-version, compléter les colonnes « résultat » dans le scénario suivant :

	Transaction 1		Transaction 2	
	opération	résultat	opération	résultat
Etat initial	Salaire de 'Joe' = 1300			
Opération 1	Commit ;			
Opération 2			Commit ;	
Opération 3	SEL			
Opération 4			SEL	
Opération 5	UPD			
Opération 6			SEL	
Opération 7	SEL			
Opération 8	Commit ;			
Opération 9			UPD	
Opération 10			Commit ;	

Avec: `UPD = UPDATE WORKER SET salaire_w=salaire_w + 100 WHERE id_w=30;`
`SEL = SELECT salaire_w FROM WORKER WHERE id_w=30;`

Q19 : Une banque en ligne gère les comptes de ses clients. Une table `Compte (num_C, Nom_C, Prénom_C, Ville_C, Solde)` permet d'organiser les comptes des clients (chaque compte appartient à un seul client). Trois types de transactions s'exécutent en parallèle. Chaque fois qu'un nouveau compte est créé, une transaction de type T1 insert une nouvelle ligne dans la table `Compte`. Lorsque qu'un client cherche à consulter ses comptes, le système lance une recherche pour déterminer de façon stable (i.e., isolée des effets des autres transactions) le solde de chaque compte de ce client (un client peut avoir plusieurs comptes), ce qui correspond à des transactions de type T2. Enfin, les gestionnaires de la banque en ligne calculent régulièrement des statistiques globales (ex: nombre de comptes à découvert par `Ville`) générant des transactions de type T3. Bien sûr, plusieurs transactions de chaque type peuvent être réalisées de manière concurrente. Répondre aux questions suivantes sur la base de ces hypothèses:

- Quel est le degré d'isolation le mieux adapté à chaque type de transaction ?
- Préciser les situations de blocage qui peuvent se produire entre ces transactions. Dit autrement, indiquer les transaction de type Ti qui peuvent se bloquer en attente de transactions de type Tj, avec i=j ou i≠j. Justifier votre réponse.
- Indiquer si une situation de verrou mortel est possible entre certaines transactions. Justifier votre réponse.