Contexte

Vous travaillez pour une entreprise spécialisée dans la collecte d'avis et de sondages. Les formulaires sont générés dynamiquement et chaque formulaire contient un ensemble variable de questions. Chaque participant répond à un formulaire, et ses réponses sont stockées dans une structure flexible pour permettre un reporting personnalisé.

L'objectif est de modéliser cette architecture dynamique et de produire un système de reporting basé sur les fonctions `JSON` en SQL.

Objectifs pédagogiques

- Comprendre la modélisation flexible via JSON
- Manipuler les fonctions JSON dans SQL (MySQL 8+, PostgreSQL)
- Générer des rapports dynamiques en SQL
- Gérer la structure des formulaires avec plusieurs niveaux d'abstraction

Schéma conceptuel à modéliser

- **formulaires(id_formulaire, titre, date_creation)**
- **questions(id_question, id_formulaire, texte_question, type_champ)**
- **participants(id_participant, nom)**
- **reponses(id_reponse, id_participant, id_formulaire, donnees_json)**

Exemple de contenu dans la colonne `donnees_json` :

```
{
    "1": "Oui",
    "2": "25",
    "3": "Tr s satisfait"
}
```

Travail demandé

Étape 1 - Création des tables

Créez les tables selon le schéma ci-dessus. Pensez à définir des clés primaires, clés étrangères, et à utiliser `JSON` comme type pour `donnees_json`.

Indication : utiliser `JSON` comme type de données dans MySQL ou PostgreSQL.

Étape 2 - Insertion de données de test

- Créez un formulaire avec au moins 3 questions de type libre (`text`, `number`, `choice`)
- Insérez au moins 2 participants avec des réponses sous forme de JSON

Exemple d'insertion JSON :

```
INSERT INTO reponses (id_participant, id_formulaire, donnees_json)
VALUES (1, 1, '{
    "1": "Oui",
    "2": "30",
    "3": "Tr s satisfait"
}');
```

Étape 3 - Requêtes de reporting dynamiques

Écrivez des requêtes SQL permettant d'extraire :

1. Toutes les réponses à une question spécifique (`id_question = 3`)
2. La moyenne des réponses à une question numérique (ex: question 2)
3. La distribution des réponses à une question de type `choix`

Indications :

- `JSON_EXTRACT(donnees_json, '$.2')` pour extraire la réponse à la question 2
- `CAST(... AS UNSIGNED)` pour calculer des moyennes

Étape 4 - Requête pour voir toutes les réponses avec leur texte

Vous devez afficher les réponses des participants avec les intitulés de question associés. Pour cela, vous pouvez joindre les données avec la table `questions` dynamiquement.

Challenge : utilisez une CTE ou un script SQL côté serveur pour faire apparaître les réponses avec le `texte_question`.

```
SELECT q.texte_question, JSON_UNQUOTE(JSON_EXTRACT(r.donnees_json, CONCAT('$
FROM reponses r
JOIN questions q ON q.id_formulaire = r.id_formulaire
WHERE r.id_participant = 1;
```


Bonus : Validation et contraintes

Ajoutez un trigger pour interdire qu'un participant réponde deux fois au même formulaire.

```
CREATE TRIGGER check_reponse_unique
BEFORE INSERT ON reponses
FOR EACH ROW
BEGIN
    IF EXISTS (
        SELECT 1 FROM reponses
        WHERE id_participant = NEW.id_participant
        AND id_formulaire = NEW.id_formulaire
    ) THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Participant a d j  r pondu.';
    END IF;
END;
```

Éléments à remettre

- Script SQL complet (.sql)
- Capture des requêtes et résultats
- Analyse des choix techniques (avantages du JSON, flexibilité, inconvénients)

Niveau : Avancé

Ce TP demande la maîtrise de :

- Structures dynamiques
- Fonctions SQL avancées sur JSON
- Jointures complexes
- Reporting adaptatif