Explication de l'architecture de notre base de données rondier.mwb

1) Agent :

Un agent est un employé du musée. On lui associe une simple table Agent (1) qui permet de connaître le nom, prénom, sexe de l'agent mais également la date de naissance, la date d'embauche et le salaire de celui-ci. Nous collectons ces informations à l'embauche de l'agent puis nous ajoutons une ligne (INSERT) à notre base pour chaque nouvel agent.

2) Identifiant_agent :

Chaque agent exerçant dans le musée se voit attribué un tag RFID unique (situé dans le poste de sécurité). Nous aurions alors pu ajouter ce tag en tant que colonne dans la table Agent mais nous devons prendre en compte le fait que les agents peuvent changer (nouvel arrivée, départs...) alors que les tags ne changent pas. Nous créons alors une table qui fait le lien entre le tag (5) et l'agent : identifiant_agent.
Cette table contient également le grade de l'agent sous forme VARCHAR (agent, chef...). On peut par exemple imaginer que le chef de service possède un tag en tant que chef et un autre en tant qu'agent. Pour cette raison, nous choisissons une relation 1-N entre l'agent et l'identifiant_agent. De plus, un identifiant_agent correspond à un seul tag bien précis -> relation 1-1 entre les tables 2 et 5.

3) Modèle_de_ronde :

Un modèle de ronde est créé par un chef de service. C'est sur ce modèle de ronde qu'on va se baser plus tard pour effectuer nos instances de ronde.
Nous avons deux solutions pour créer une ronde :

- **Nom** : Le nom du modèle de ronde peut par exemple être l'adresse MAC du bâton dans un premier temps puis changé ensuite. Ceci va nous servir à ne pas perdre la bonne ligne de la table en faisant un SELECT de la ligne possédant la bonne adresse MAC.
- **DateCreation**: La date de création de la ronde va correspondre à la date de lecture de tag RFID du chef de service
- **Etat**: L'état du modèle de ronde est un entier : si etat = 0, le modèle de ronde est inactif. Il existe mais il ne peut pas avoir d'instance de ronde correspondante. Si etat = 1, le modèle est actif.
- **Duree**: La durée sera le temps qu'un agent va mettre pour effectuer une ronde qui correspond à ce modèle. On suppose que le chef va effectuer plus ou moins la même durée lors de la création du modèle. La durée sera alors la différence de temps entre la dernière lecture du tag du chef (fin du modèle) et la DateCreation dont on a parlé précédemment.

- **fk_identifiant_agent_**: enfin, nous voulons connaître pour chaque modele la personne qui l'a créé. On ajoute donc une clé étrangère pour avoir une relation 1-N entre les tables 2 et 3.

4) Instance_de_ronde:

Un agent va s'appuyer sur le modele_de_ronde (créé une seul fois par le chef de service) pour effectuer sa ronde : on a alors une instance du modèle de ronde. L'instance va être créée par le chef de service quand il affectera des rondes à ses agents (grâce à une interface graphique).
A la création, le chef de service va remplir les champs :

- **Nom**
- **Etat** : l'instance peut être annulé, reporté, effective, en cours.....
- **Début_ronde_planning** : c'est l'heure et la date à laquelle la ronde doit commencer
- **fk_identifiant_agent** : c'est l'identifiant_agent qui correspond au tag qui doit être lu en début d'instance de ronde. Un agent peut évidemment réaliser plusieurs instances mais une instance correspond à un seul agent -> relation 1-N
- **fk_modele_de_ronde** : le but de l'agent est d'effectuer la même ronde que son chef lorsqu'il l'a créée. Le chef va choisir à la création de quel modèle il faut créer l'instance. On fait donc le lien entre le modèle et l'instance. Modele_de_ronde va servir de modèle pour plusieurs instances
  -> relation 1-N

Les autres champs seront remplis à l'exécution de l'instance par l'agent :

- **Debut et Fin_ronde_reel** : c'est les dates/heures effectives de l'instance Le début et la fin sont les date du bâton lorsqu'ils lit les RFID du bon agent.
- **fk_baton_rondier** : Les bâtons ne sont pas nominatifs (n'importe quel agent peut utiliser n'importe quel bâton). Nous devons donc connaître le bâton utilisé pour ensuite retrouver la bonne instance pendant la ronde. En effet, on va effectuer un SELECT dans notre BDD en choisissant l'instance qui a notre bâton. (de la même manière que pour Modele_de_ronde sauf qu'on utilise une clé étrangère au lieu du nom)

5) Point_de_passage :

- **Nom**
- **Type** : Les points de passage sont de 2 types
  - agent : c'est les tags qui dans le poste de sécurité
  - oeuvre : c'est les tags qui sont dans le musée
- **X, Y** : coordonnées du tag, il peut nous servir à créer une interface représentant le chemin à parcourir
- **Tag** : valeur unique

6) Modele_de_ronde_has_Point_de_passage:

Cette table est une table de jonction entre les tables 3 et 5. (relation N-M)
On ajoute les colonnes DateEvent et DateInsertion pour collecter auprès du bâton les heures de passage.
De plus, un modèle de ronde peut passer plusieurs fois par un point de passage, nous ajoutons donc un id qui va rendre chaque ligne unique.

7) Instance_de_ronde_has_Point_de_passage:

Cette table est une table de jonction entre les tables 4 et 5. (relation N-M)
On ajoute les colonnes DateEvent et DateInsertion pour collecter auprès du bâton les heures de passage.
De plus, un modèle de ronde peut passer plusieurs fois par un point de passage, nous ajoutons donc un id qui va rendre chaque ligne unique.

8) Baton_rondier:

Le bâton renvoie 3 informations :

- son adresse MAC : elle correspond à une des colonnes de la table 8. Elle ne changera jamais.
- le tag lu par le bâton
- la Date : le bâton nous renseigne la date pour chaque nouvelle lecture. Grâce à ceci, nous pouvons notamment remplir les lignes de la table 5 en renseignant les dates sous forme de DATETIME.

Dans la table, nous avons également la DateCreation et l'état du bâton.
De cette façon, nous pouvons facilement changer de mode en fonction du tag lu.
Ex : lecture tag de chef -> on passe en mode création. A la lecture du prochain point de passage, nous saurons qu'il faudra chercher la bonne ligne de la table modele_de_ronde pour ajouter le point de passage.

9) Baton_rondier_has_point_de_passage

Cette table permet d'avoir un historique de tous les points de passage pour un bâton_rondier avec la date associée (DateEvent).

Ordre de remplissage des DML :

- Agent, bâton_rondier, Point_de_passage
- identifiant_agent, bâton_has_point_de_passage
- modele_de_ronde
- instance_de_ronde
- modele_has_point_de_passage, instance_has_point_de_passage