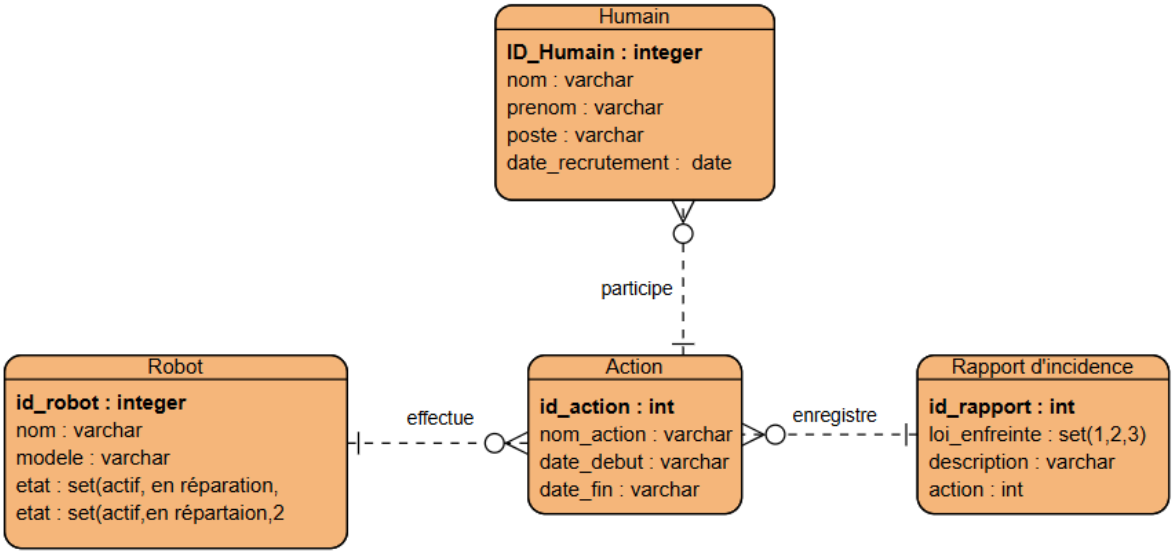
PROJET BDD

Partie 1 - Modélisation

MCD



MLD

Humain(ID_humain,nom,poste,date_recrutement)

Robot(id_robot,nom,modele,etat)

Action(id_action,description,date_debut,date_fin)

RapportIncidence(id_rapport,typeViolation,description,loiViole,#action_id_action)

ParticipationRobot(#robot_id_robot,#action_id_action)

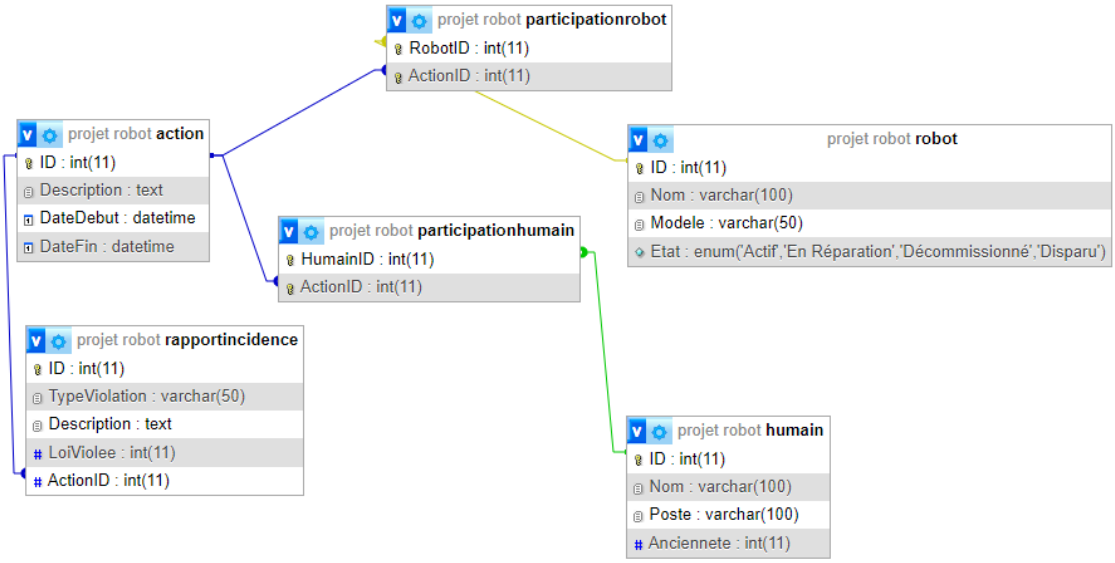
ParticipationHumain((#robot_id_humain,#action_id_action)

Code

```
CREATE TABLE Robot (
   ID INT PRIMARY KEY,
   Nom VARCHAR(100),
   Modele VARCHAR(50),
   Etat ENUM('Actif', 'En Réparation', 'Décommissionné', 'Disparu')
);
 GOT
```

```
CREATE TABLE Humain (
   ID INT PRIMARY KEY,
   Nom VARCHAR(100),
   Poste VARCHAR(100),
   Anciennete INT
);

CREATE TABLE Action (
   ID INT PRIMARY KEY,
   Description TEXT,
   DateDebut DATETIME,
   DateFin DATETIME
);

CREATE TABLE ParticipationRobot (
   RobotID INT,
   ActionID INT,
   PRIMARY KEY (RobotID, ActionID),
   FOREIGN KEY (RobotID) REFERENCES Robot(ID),
   FOREIGN KEY (ActionID) REFERENCES Action(ID)
);

CREATE TABLE ParticipationHumain (
   HumainID INT,
   ActionID INT,
   PRIMARY KEY (HumainID, ActionID),
   FOREIGN KEY (HumainID) REFERENCES Humain(ID),
   FOREIGN KEY (ActionID) REFERENCES Action(ID)
```

```
);


CREATE TABLE RapportIncidence (
    ID INT PRIMARY KEY,
    TypeViolation VARCHAR(50),
    Description TEXT,
    LoiViolee INT CHECK (LoiViolee IN (1, 2, 3)),
    ActionID INT,
    FOREIGN KEY (ActionID) REFERENCES Action(ID)
);
```



Partie 2 - Génération de données et ingestion

Partie généré par CHAT GPT

```
-- Insertion de robots
INSERT INTO Robot (ID, Nom, Modele, Etat)
VALUES
(1, 'Alpha1', 'Explorateur-X', 'Actif'),
(2, 'Beta2', 'Minage-Y', 'En Réparation'),
```

```
(3, 'Gamma3', 'Maintenance-Z', 'Disparu');

-- Insertion d'humains
INSERT INTO Humain (ID, Nom, Poste, Anciennete)
VALUES
(1, 'Alice', 'Ingénieur', 5),
(2, 'Bob', 'Technicien', 2),
(3, 'Claire', 'Superviseur', 7);

-- Insertion d'actions
INSERT INTO Action (ID, Description, DateDebut, DateFin)
VALUES
(1, 'Exploration site A', '2024-12-01 08:00:00', '2024-12-01 18:00:00'),
(2, 'Réparation générateur', '2024-12-01 10:00:00', '2024-12-01 15:00:00');

-- Participation des robots
INSERT INTO ParticipationRobot (RobotID, ActionID)
VALUES
(1, 1),
(2, 2);

-- Participation des humains
INSERT INTO ParticipationHumain (HumainID, ActionID)
VALUES
(1, 1),
(2, 2);

-- Insertion de rapports d'incidence
```

INSERT INTO RapportIncidence (ID, TypeViolation, Description, LoiViolee, ActionID)

VALUES

(1, 'Violation Loi 1', 'Robot n’a pas sauvé un humain en danger.', 1, 1);

Pour vérifier que le modèle de données est cohérent et garantit l’intégrité des données non conforme, on utilise :

INSERT INTO RapportIncidence (ID, TypeViolation, Description, LoiViolee, ActionID)

VALUES(2, 'Violation Loi 2', 'Non-respect des ordres humains.', 2, 99); -- Doit échouer

Partie 3 – Gestion des droits d’accès

1. **Administrateur**
   L'administrateur a un accès complet, car c'est nécessaire pour gérer les opérations critiques sur la base.

2. **Analyste**
   L'analyste a un accès en lecture seule sur les vues analytiques, car cela permet de limiter les droits et d'éviter toute modification accidentelle des données.

3. **Technicien**
   Le technicien a un accès pour modifier l’état des robots, car cela est essentiel pour les tâches de maintenance liées à leur fonctionnement.

4. **Superviseur éthique**
   Le superviseur éthique a un accès en lecture aux données des actions et des rapports d'incidence, car cela est indispensable pour analyser les conflits éthiques sans compromettre l'intégrité des données.

Création des utilisateurs

-- Administrateur

CREATE USER 'admin_user'@'%' IDENTIFIED BY 'secure_password_admin';

-- Analyste

CREATE USER 'analyst_user'@'%' IDENTIFIED BY 'secure_password_analyst';

-- Technicien

CREATE USER 'technician_user'@'%' IDENTIFIED BY 'secure_password_technician';

-- Superviseur éthique

CREATE USER 'ethics_supervisor'@'%' IDENTIFIED BY 'secure_password_ethics';

Privileges:

Administrateur

GRANT ALL PRIVILEGES ON robot.* TO 'admin_user'@'%';

Analyste:

D'abord créer la vue analytique:

CREATE OR REPLACE VIEW Vue_Analyse_Actions AS SELECT R.Nom AS Robot_Nom, COUNT(A.ID_Action) AS Nombre_Actions FROM Robot R LEFT JOIN Robot_Action RA

ON R.ID_Robot = RA.ID_Robot LEFT JOIN Action A ON RA.ID_Action = A.ID_Action
GROUP BY R.Nom;

Accorde l'accès en lecture uniquement à la vue

GRANT SELECT ON robot.Vue_Analyse_Actions TO 'analyst_user'@'%';

Technicien:

GRANT SELECT, UPDATE (Etat) ON robot.Robot TO 'technician_user'@'%';

Superviseur éthique:

GRANT SELECT ON ColonisationAsimov.Action TO 'ethics_supervisor'@'%';

GRANT SELECT ON ColonisationAsimov.Rapport_Incidence TO 'ethics_supervisor'@'%';

L'analyste peut avoir accès à certaines informations du rapport d'incidence :

CREATE OR REPLACE VIEW Vue_Rapport_Public

AS SELECT ID_Rapport, Type_Loi,

LEFT(Description, 100) AS Description_Tronquee

Rapport_Incidence;

GRANT SELECT ON Robot.Vue_Rapport_Public TO 'analyst_user'@'%';

Partie 4 – Analyse des évènements

1. Humains les plus impliqués dans les rapports d'incidence

SELECT ID_Action, COUNT(*) AS nombre_incidences

```
FROM rapport_incidence
GROUP BY ID_Action
ORDER BY nombre_incidences DESC
```

2. Robots les plus impliqués dans les rapports d'incidence

```
SELECT r.Nom, r.Modele, COUNT(ri.ID_Rapport) AS Nombre_Incidents
FROM robot r
JOIN robot_action ra ON r.ID_Robot = ra.ID_Robot
JOIN rapport_incidence ri ON ra.ID_Action = ri.ID_Action
GROUP BY r.ID_Robot
ORDER BY Nombre_Incidents DESC;
```

3. Actions ayant mené au plus de rapports d'incidence

```
SELECT a.Nom_Action, COUNT(ri.ID_Rapport) AS Nombre_Incidents
FROM action a
JOIN rapport_incidence ri ON a.ID_Action = ri.ID_Action
GROUP BY a.ID_Action
ORDER BY Nombre_Incidents DESC;
```

4. Robots impliqués dans des incidents qui ont disparu

```
SELECT r.Nom, r.Modele
FROM robot r
JOIN robot_action ra ON r.ID_Robot = ra.ID_Robot
JOIN rapport_incidence ri ON ra.ID_Action = ri.ID_Action
WHERE r.Etat = 'Disparu'
GROUP BY r.ID_Robot;
```

Analyses perso :

a) Nombre total d'incidents par type de loi violée

```
SELECT Type_Loi, COUNT(ID_Rapport) AS Nombre_Incidents
FROM rapport_incidence
GROUP BY Type_Loi
ORDER BY Nombre_Incidents DESC;
```

b) Robots par état actuel

```
SELECT Etat, COUNT(ID_Robot) AS Nombre_Robots
FROM robot
GROUP BY Etat;
```

c) Moyenne d'ancienneté des humains impliqués dans des incidents

```
SELECT AVG(h.Anciennete) AS Anciennete_Moyenne
FROM humain h
JOIN humain_action ha ON h.ID_Humain = ha.ID_Humain
JOIN rapport_incidence ri ON ha.ID_Action = ri.ID_Action;
```

Partie 5

```
DROP VIEW Vue_Humains_Impliques;
DROP VIEW Vue_Robots_Impliques;
DROP VIEW Vue_Actions_Problematique;
```

```
DROP INDEX idx_participation_humain_id ON humain_action;
DROP INDEX idx_participation_robot_id ON robot_action;
DROP INDEX idx_action_id ON rapport_incidence;

-- Création des vues pour pré-calculer les résultats
CREATE VIEW Vue_Humains_Impliques AS
SELECT h.Nom AS robot_implice, COUNT(*) AS Nombre_Rapports
FROM Humain h
INNER JOIN humain_action ph ON h.ID_Humain = ph.ID_Humain
INNER JOIN action a ON a.ID_Action = ph.ID_Action
INNER JOIN rapport_incidence ri ON a.ID_Action = ri.ID_Action
GROUP BY h.Nom;

CREATE VIEW Vue_Robots_Impliques AS
SELECT r.Nom AS humain_implice, COUNT(*) AS Nombre_Rapports
FROM Robot r
INNER JOIN robot_action ra ON r.ID_Robot = ra.ID_Robot
INNER JOIN action a ON a.ID_Action = ra.ID_Action
INNER JOIN rapport_incidence ri ON a.ID_Action = ri.ID_Action
GROUP BY r.Nom;

CREATE VIEW Vue_Actions_Problematique AS
SELECT a.Nom_Action AS action_probleme, COUNT(*) AS Nombre_Rapports
FROM Action a
INNER JOIN Rapport_Incidence ri ON a.ID_Action = ri.ID_Action
GROUP BY a.Nom_Action;

SELECT * FROM Vue_Humains_Impliques;
```

```
SELECT h.Nom AS robot_implice, COUNT(*) AS Nombre_Rapports
FROM Humain h
INNER JOIN humain_action ph ON h.ID_Humain = ph.ID_Humain
INNER JOIN action a ON a.ID_Action = ph.ID_Action
INNER JOIN rapport_incidence ri ON a.ID_Action = ri.ID_Action
GROUP BY h.Nom;
SELECT * FROM Vue_Robots_Impliques;
SELECT * FROM Vue_Actions_Problematique;



-- Création des index pour accélérer les jointures
CREATE INDEX idx_participation_humain_id ON humain_action(ID_Humain);
CREATE INDEX idx_participation_robot_id ON robot_action(ID_Robot);
CREATE INDEX idx_action_id ON rapport_incidence(ID_Action);
```