Présentation de projet

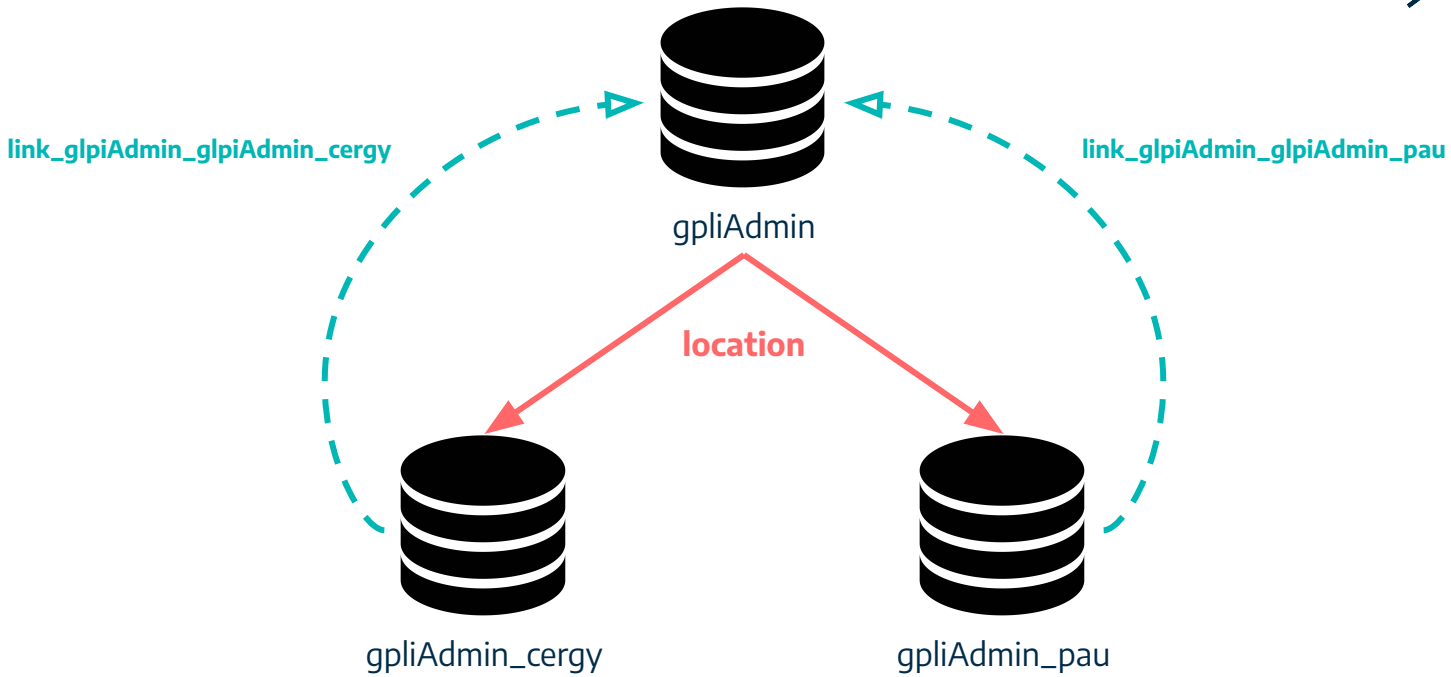
GLPI (Gestion du parc informatique de CY Tech)

Alexis TOURRENC--LECERF
Louis-Alexandre LAGUET
Cédric DOLLET

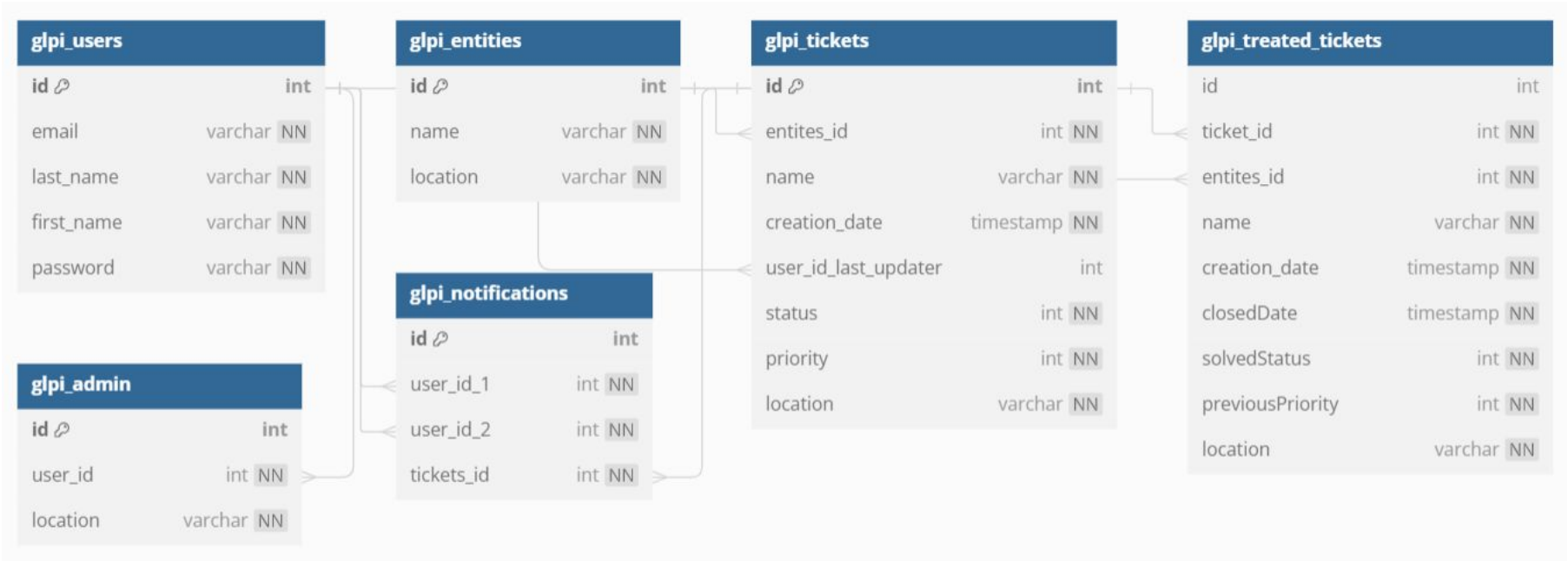
Table des matières

01 Modélisation

Fragmentation horizontale



Tables


glpi_users
id int
email varchar NN
last_name varchar NN
first_name varchar NN
password varchar NN
glpi_entities
id int
name varchar NN
location varchar NN
glpi_tickets
id int
entites_id int NN
name varchar NN
creation_date timestamp NN
user_id_last_updater int
status int NN
priority int NN
location varchar NN
glpi_treated_tickets
id int
ticket_id int NN
entites_id int NN
name varchar NN
creation_date timestamp NN
closedDate timestamp NN
solvedStatus int NN
previousPriority int NN
location varchar NN
glpi_notifications
id int
user_id_1 int NN
user_id_2 int NN
tickets_id int NN
glpi_admin
id int
user_id int NN
location varchar NN

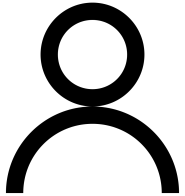
Utilisateurs et rôles

gpliAdmin

gpliAdmin_cergy

gpliAdmin_pau

technician_role



simple_user_role

02 Utilisation

Vues de gpliAdmin

```
CREATE OR REPLACE VIEW selectAllTickets AS
SELECT *
FROM glpi_tickets@link_glpiAdmin_glpiAdmin_cergy
UNION
SELECT * FROM glpi_tickets@link_glpiAdmin_glpiAdmin_pau;

CREATE OR REPLACE VIEW selectAllClosedTickets AS
SELECT * FROM glpi_treated_tickets@link_glpiAdmin_glpiAdmin_cergy
UNION
SELECT *
FROM glpi_treated_tickets@link_glpiAdmin_glpiAdmin_pau;

CREATE OR REPLACE VIEW selectAllUsers AS
SELECT *
FROM glpi_users@link_glpiAdmin_glpiAdmin_cergy
UNION
SELECT * FROM glpi_users@link_glpiAdmin_glpiAdmin_pau;
```

Vues de gpliAdmin_cergy

```sql
CREATE OR REPLACE VIEW selectTicketFromFermat AS
SELECT * FROM glpi_tickets WHERE location = 'Fermat';

CREATE OR REPLACE VIEW selectTicketFromCondorcet AS
SELECT * FROM glpi_tickets WHERE location = 'Condorcet';

CREATE OR REPLACE VIEW selectTicketFromTuring AS
SELECT * FROM glpi_tickets WHERE location = 'Turing';

CREATE OR REPLACE VIEW selectTicketFromCauchy AS
SELECT * FROM glpi_tickets WHERE location = 'Cauchy';

CREATE OR REPLACE VIEW selectTicketFromMartin AS
SELECT * FROM glpi_tickets WHERE location = 'Saint Martin';

CREATE OR REPLACE VIEW selectTicketFromGermain AS
SELECT * FROM glpi_tickets WHERE location = 'Saint Germain';
```

Vue de gpliAdmin_pau

```sql
CREATE OR REPLACE VIEW selectTicketFromPau AS
SELECT * FROM glpi_tickets WHERE location = 'Pau';
```

Déclencheurs communs aux BDDs

```
CREATE OR REPLACE TRIGGER trg_user_id_last_updater_notification
AFTER UPDATE OF user_id_last_updater ON glpi_tickets
FOR EACH ROW
DECLARE
    v_user_id_1 glpi_users.id%TYPE;
    v_user_id_2 glpi_users.id%TYPE;
BEGIN
    -- Récupérer les IDs des utilisateurs concernés
    SELECT id INTO v_user_id_1 FROM glpi_users WHERE id = :OLD.user_id_last_updater;
    SELECT id INTO v_user_id_2 FROM glpi_users WHERE id = :NEW.user_id_last_updater;

    -- Insérer une nouvelle notification
    INSERT INTO glpi_notifications (id, user_id_1, user_id_2, tickets_id)
    VALUES (glpi_notifications_seq.NEXTVAL, v_user_id_1, v_user_id_2, :NEW.id);
END;
```

+ Les déclencheurs liés aux séquences pour les identifiants des éléments des tables

Déclencheurs de gpliAdmin_cergy

```
-- Déclencheur pour gérer les opérations sur les tickets à Cergy
CREATE OR REPLACE TRIGGER insert_ticket_turing
INSTEAD OF INSERT OR UPDATE OR DELETE ON selectTicketFromTuring
FOR EACH ROW
BEGIN
    IF INSERTING THEN
        -- Insérer dans la table glpi_tickets
        INSERT INTO glpi_tickets (entites_id, name, user_id_last_updater, status, location, priority)
        VALUES (:NEW.entites_id, :NEW.name, :NEW.user_id_last_updater, :NEW.status, 'Turing', :NEW.priority);
    ELSIF UPDATING THEN
        -- Mettre à jour la table glpi_tickets
        UPDATE glpi_tickets
        SET
            entites_id = NVL(:NEW.entites_id, entites_id),
            name =  NVL(:NEW.name, name),
            user_id_last_updater =  NVL(:NEW.user_id_last_updater, user_id_last_updater),
            status = NVL(:NEW.status, status),
            priority =  NVL(:NEW.priority, priority)
        WHERE
            id = :NEW.id;
    ELSIF DELETING THEN
        -- Supprimer de la table glpi_tickets
        DELETE FROM glpi_tickets WHERE id = :OLD.id;
    END IF;
END;
```

+ Les déclencheurs liés aux autres sites (Condorcet, Cauchy, Fermat, Saint-Martin, Saint-Germain)

Déclencheurs de gpliAdmin_pau

```
-- Déclencheur pour gérer les opérations sur les tickets sur le site de Pau
CREATE OR REPLACE TRIGGER insert_ticket_pau
INSTEAD OF INSERT OR UPDATE OR DELETE ON selectTicketFromPau
FOR EACH ROW
BEGIN
    IF INSERTING THEN
        -- Insérer dans la table glpi_tickets
        INSERT INTO glpi_tickets (entites_id, name, user_id_last_updater, status, location, priority)
        VALUES (:NEW.entites_id, :NEW.name, :NEW.user_id_last_updater, :NEW.status, 'Pau', :NEW.priority);
    ELSIF UPDATING THEN
        -- Mettre à jour la table glpi_tickets
        UPDATE glpi_tickets
        SET
            entites_id = NVL(:NEW.entites_id, entites_id),
            name =  NVL(:NEW.name, name),
            user_id_last_updater =  NVL(:NEW.user_id_last_updater, user_id_last_updater),
            status = NVL(:NEW.status, status),
            priority =  NVL(:NEW.priority, priority)
        WHERE
            id = :NEW.id;
    ELSIF DELETING THEN
        -- Supprimer de la table glpi_tickets
        DELETE FROM glpi_tickets WHERE id = :OLD.id;
    END IF;
END;
```

Procédures (1/3)

```
CREATE OR REPLACE PROCEDURE create_users_procedure AS
BEGIN
    FOR user_rec IN (SELECT id, email, last_name, first_name, password FROM glpi_users) LOOP
        DECLARE
            user_count NUMBER;
        BEGIN
            -- Check if user already exists
            SELECT COUNT(*) INTO user_count FROM all_users
            WHERE username = upper(user_rec.last_name || '_' || user_rec.first_name);

            -- If user does not exist, create user and grant privileges
            IF user_count = 0 THEN
                EXECUTE IMMEDIATE 'CREATE USER ' || user_rec.last_name || '_' || user_rec.first_name ||
                                  ' IDENTIFIED BY "' || user_rec.password || '"';

                EXECUTE IMMEDIATE 'GRANT CONNECT TO ' || user_rec.last_name || '_' || user_rec.first_name;

                EXECUTE IMMEDIATE 'GRANT simple_user_role TO ' || user_rec.last_name || '_' ||
                                    user_rec.first_name;
            END IF;
        EXCEPTION
            WHEN OTHERS THEN
                -- Handle exceptions (e.g., log the error)
                DBMS_OUTPUT.PUT_LINE('Error creating user: ' || SQLERRM);
        END;
    END LOOP;
END create_users_procedure;
```

Procédures (2/3)

```
CREATE OR REPLACE PROCEDURE close_ticket_cergy (
    p_ticket_id IN NUMBER  -- ID du ticket à fermer
) AS
    v_technician_role_exists NUMBER(1);  -- Variable pour stocker l'existence de rôles de technicien
    v_priority NUMBER(1); -- Variable pour stocker la priorité du ticket
BEGIN
    -- Vérifier si l'utilisateur actuel possède l'un des rôles de technicien pour Cergy ou Pau
    SELECT COUNT(*) INTO v_technician_role_exists
    FROM USER_ROLE_PRIVS
    WHERE GRANTED_ROLE IN ('TECHNICIAN_ROLE');

    -- Récupérer la priorité du ticket
    SELECT priority INTO v_priority
    FROM glpiAdmin_cergy.glpi_tickets
    WHERE id = p_ticket_id;

    -- Insérer dans la table glpi_treated_tickets en fonction des données du ticket existant
    EXECUTE IMMEDIATE '
        INSERT INTO glpiAdmin_cergy.glpi_treated_tickets (id, ticket_id, entites_id, name, creation_date, closedDate, solvedStatus, previousPriority, location)
        SELECT glpiAdmin_cergy.glpi_treated_tickets_seq.NEXTVAL, id, entites_id, name, creation_date, SYSTIMESTAMP, CASE WHEN :tech_role_exists > 0 THEN 1 ELSE 0 END, priority, lo
        FROM glpiAdmin_cergy.glpi_tickets
        WHERE id = :ticket_id'
    USING v_technician_role_exists, p_ticket_id;

    -- Supprimer le ticket traité de la table glpi_tickets
    DELETE FROM glpiAdmin_cergy.glpi_notifications WHERE tickets_id = p_ticket_id;
    DELETE FROM glpiAdmin_cergy.glpi_tickets WHERE id = p_ticket_id;

    -- Si la priorité du ticket est élevée (5), notifier les administrateurs
    IF v_priority = 5 THEN
        -- Ajouter ici la logique pour notifier les administrateurs
        DBMS_OUTPUT.PUT_LINE('Ticket fermé avec priorité élevée. Notification envoyée aux administrateurs.');
    END IF;
EXCEPTION
    WHEN OTHERS THEN
        ROLLBACK;
        RAISE;
END;
```

Procédures (3/3)

```
CREATE OR REPLACE PROCEDURE reopen_ticket_cergy (
    p_ticket_id IN NUMBER  -- ID du ticket à rouvrir
) AS
    v_priority NUMBER(1); -- Variable pour stocker la priorité du ticket
BEGIN

    -- Récupérer la priorité du ticket
    SELECT previousPriority INTO v_priority
    FROM glpiAdmin_cergy.glpi_treated_tickets
    WHERE ticket_id = p_ticket_id;

    -- Insérer dans la table glpi_tickets en fonction des données du ticket traité
    EXECUTE IMMEDIATE '
        INSERT INTO glpiAdmin_cergy.glpi_tickets (id, entites_id, name, creation_date, user_id_last_updater, status, priority, location)
        SELECT :ticket_id, entites_id, name, SYSTIMESTAMP, NULL, 0, :priority, location
        FROM glpiAdmin_cergy.glpi_treated_tickets
        WHERE ticket_id = :ticket_id'
    USING p_ticket_id, v_priority, p_ticket_id;

    -- Supprimer le ticket traité de la table glpi_treated_tickets
    DELETE FROM glpiAdmin_cergy.glpi_treated_tickets WHERE ticket_id = p_ticket_id;

    COMMIT;

    -- Si la priorité du ticket est élevée (5), notifier les administrateurs
    IF v_priority = 5 THEN
        -- Ajouter ici la logique pour notifier les administrateurs
        DBMS_OUTPUT.PUT_LINE('Ticket réouvert avec priorité élevée. Notification envoyée aux administrateurs.');
    END IF;
EXCEPTION
    WHEN OTHERS THEN
        ROLLBACK;
        RAISE;
END;
```

03 Optimisation

Indexs

```sql
-- Création d'un index sur la colonne location de la table glpi_tickets
-- Cet index accélérera les opérations de recherche basées sur la localisation des tickets
CREATE INDEX idx_glpi_tickets_location ON glpi_tickets(location);

-- Création d'un index sur la colonne priority de la table glpi_tickets
-- Cet index améliorera les performances des requêtes impliquant la priorité des tickets
CREATE INDEX idx_glpi_tickets_priority ON glpi_tickets(priority);

-- Création d'un index sur la colonne entites_id de la table glpi_tickets
-- Cet index améliorera les performances des requêtes impliquant les entités des tickets
CREATE INDEX idx_glpi_tickets_entities_id ON glpi_tickets(entites_id);

-- Création d'un index sur la colonne entites_id de la table glpi_treated_tickets
-- Cet index améliorera les performances des requêtes impliquant les entités des tickets traités
CREATE INDEX idx_glpi_treated_tickets_entities_id ON glpi_treated_tickets(entites_id);

-- Création d'un index sur la colonne email de la table glpi_users
-- Cet index améliorera les performances des requêtes impliquant l'email des utilisateurs
CREATE UNIQUE INDEX idx_glpi_users_email ON glpi_users(email);

-- Création d'un index sur la colonne user_id_1 de la table glpi_notifications
-- Cet index améliorera les performances des requêtes impliquant l'ID de l'utilisateur 1 dans les notifications
CREATE INDEX idx_glpi_notifications_user_id_1 ON glpi_notifications(user_id_1);

-- Création d'un index sur la colonne user_id_2 de la table glpi_notifications
-- Cet index améliorera les performances des requêtes impliquant l'ID de l'utilisateur 2 dans les notifications
CREATE INDEX idx_glpi_notifications_user_id_2 ON glpi_notifications(user_id_2);
```

Cluster

```
-- Création d'un cluster sur la colonne location de la table glpi_tickets
-- Les clusters regroupent physiquement les lignes de données en fonction des valeurs des colonnes de regroupement spécifiées
DROP CLUSTER clst_glpi_tickets_location;

CREATE CLUSTER clst_glpi_tickets_location
(
    location VARCHAR(255)
)
SIZE 1024
HASHKEYS 100;
```

04 Démonstration

Merci !

Avez-vous des questions ?

Alexis TOURRENC--LECERF
Louis-Alexandre LAGUET
Cédric DOLLET