Rapport du Projet de

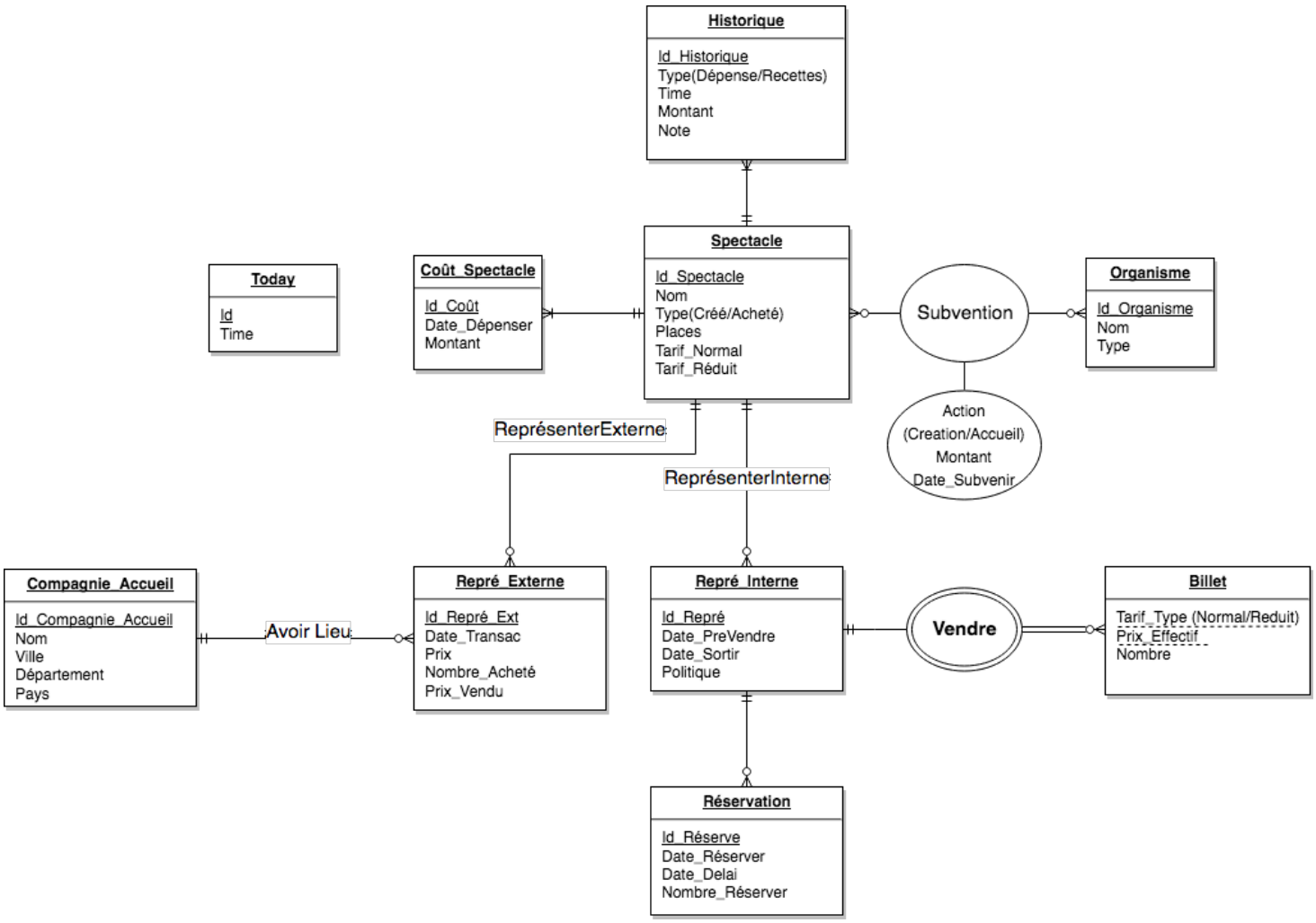
« Base de Données Avancée (PostgreSQL) »

Université Paris Diderot

21502157 FAN Yi-Zhe | 21505130 HSIEH Yung-Kun

CODE : *git@moule.informatique.univ-paris-diderot.fr:Yizhe/project-bda-m1-2017.git*

Les Diagrammes de Modèle Entité-Association



Les Schémas et Contraints

Spectacle (IdSpectacle, Nom, Places, Type, TarifNormal, TarifRéduit)

Check constraints:

"spectacle_places_check" CHECK (places >= 0)

"spectacle_tarif_normal_check" CHECK (tarif_normal >= 0::numeric)

"spectacle_tarif_reduit_check" CHECK (tarif_reduit >= 0::numeric)

"spectacle_type_check" CHECK (type = ANY (ARRAY[0, 1]))

Cout_Spectacle (Id_Coût, #IdSpectacle, Date_Dépenser, Montant)

Check constraints:

"cout_spectacle_montant_check" CHECK (montant > 0::numeric)

Foreign-key constraints:

"cout_spectacle_id_spectacle_fkey" FOREIGN KEY (id_spectacle) REFERENCES spectacle(id_spectacle)

Triggers:

1. cout_achete_checker BEFORE INSERT OR UPDATE ON cout_spectacle FOR EACH ROW EXECUTE PROCEDURE check_cout_achete()
2. historique_cout_modifier AFTER INSERT OR DELETE OR UPDATE ON cout_spectacle FOR EACH ROW EXECUTE PROCEDURE modify_cout_historique()

Organismes (Id_Organisme, Nom, Type)

Subvention (#Id_Spectacle, #Id_Organisme, Action, Montant, Date_Subvenir)

Check constraints:

"subvention_action_check" CHECK (action::text = ANY (ARRAY['creation'::character varying, 'accueil'::character varying]::text[]))

"subvention_montant_check" CHECK (montant > 0::numeric)

Foreign-key constraints:

"subvention_id_organisme_fkey" FOREIGN KEY (id_organisme) REFERENCES organisme(id_organisme)

"subvention_id_spectacle_fkey" FOREIGN KEY (id_spectacle) REFERENCES spectacle(id_spectacle)

Triggers:

1. historique_subvenir_modifier AFTER INSERT OR DELETE OR UPDATE ON subvention FOR EACH ROW EXECUTE PROCEDURE modify_subvenir_historique()
2. subvenir_action_checker BEFORE INSERT OR UPDATE ON subvention FOR EACH ROW EXECUTE PROCEDURE check_subvenir_action()

Repré_Interne (Id_Repré, #IdSpectacle, Date_PreVendre, Date_Sortir, Politique)

Check constraints:

"repre_interne_check" CHECK (date_sortir > date_prevendre)

"repre_interne_politique_check" CHECK (politique >= 0)

Foreign-key constraints:

"repre_interne_id_spectacle_fkey" FOREIGN KEY (id_spectacle) REFERENCES spectacle(id_spectacle)

Billet (#Id_Repré, Tarif_Type, Prix_Effectif, Nombre)

Check constraints:

"billet_nombre_check" CHECK (nombre >= 0)

"billet_tarif_type_check" CHECK (tarif_type = ANY (ARRAY[0, 1]))

Foreign-key constraints:

"billet_id_repre_fkey" FOREIGN KEY (id_repre) REFERENCES repre_interne(id_repre)

Triggers:

1. on_billet_changer AFTER INSERT OR DELETE OR UPDATE ON billet FOR EACH ROW EXECUTE PROCEDURE on_billet_change()
2. on_billet_preprocessor BEFORE INSERT OR UPDATE ON billet FOR EACH ROW EXECUTE PROCEDURE on_billet_preprocess()

Historiques (Id_Historique, #Id_Spectacle, Type, Time, Montant, Note)

Check constraints:

"historique_type_check" CHECK (type = ANY (ARRAY[0, 1]))

Foreign-key constraints:

"historique_id_spectacle_fkey" FOREIGN KEY (id_spectacle) REFERENCES spectacle(id_spectacle)

Repré_Externe (Id_Repré_Ext, #Id_Spectacle, #Id_Compagnie_Accueil, Date_Transac, Prix, Nombre_Acheté, Prix_Vendu)

Check constraints:

"repre_externe_nombre_achete_check" CHECK (nombre_achete > 0)

"repre_externe_prix_check" CHECK (prix > 0::numeric)

"repre_externe_prix_check1" CHECK (prix > 0::numeric)

Foreign-key constraints:

"repre_externe_id_compagnie_accueil_fkey" FOREIGN KEY (id_compagnie_accueil) REFERENCES compagnie_accueil(id_compagnie_accueil)

"repre_externe_id_spectacle_fkey" FOREIGN KEY (id_spectacle) REFERENCES spectacle(id_spectacle)

Triggers:

1. historique_repre_externe_modifier AFTER INSERT OR DELETE OR UPDATE ON repre_externe FOR EACH ROW EXECUTE PROCEDURE modify_repre_externe_historique()
2. type_checker_prix_modifier BEFORE INSERT OR UPDATE ON repre_externe FOR EACH ROW EXECUTE PROCEDURE check_type_modify_prix()

Compagnie_Accueil (Id_Compagnie_Accueil, Nom, Ville, Département, Pays)

Today (Id, time)

Les Fonctions Commerciales

FUNCTION reserver (idRepre INTEGER, nombre INTEGER) RETURNS INTEGER

Cette fonction sert à faire des réservations de *nombre* de places pour une représentation *idRepre*. Elle renvoie -1 si la réservation échoue, sinon elle renvoie un numéro pour la réservation.

FUNCTION calc_nombre_place_dans_billet (idRepre INTEGER) RETURNS INTEGER

Cette fonction sert à calculer le nombre des places vendus (places réservés exclues).

FUNCTION calc_nombre_place_dans_reserv (idRepre INTEGER) RETURNS INTEGER

Cette fonction sert à calculer le nombre des places réservés (places vendus exclues) pour la représentation *idRepre*.

FUNCTION get_seat_sold_percentage (idRepre INTEGER) RETURNS numeric (2,2)

Cette fonction sert à calculer le pourcentage de places vendus (places réservés exclues) pour la représentation *idRepre*.

FUNCTION get_seat_full_percentage (idRepre INTEGER) RETURNS numeric (2,2)

Cette fonction sert à calculer le pourcentage de places totales (places réservées et places vendues) pour la représentation *idRepre*.

FUNCTION get_current_ticket_price (idRepre INTEGER, tarifType INTEGER) RETURNS numeric (8,2)

Cette fonction sert à calculer le prix actuel du billet selon le politique et le type de tarif (Réduit = 1, Normal = 0) du spectacle pour la représentation *idRepre.*

FUNCTION pay_reservation (myIdReserve INTEGER, tarifNormal INTEGER, tarifReduit INTEGER) RETURNS numeric(8,2)

Cette fonction sert à payer une réservation de numéro *myIdReserve*. Vous devez indiquer le nombre de billets tarif normal *tarifNormal* et tarif réduit *tarifReduit*.

FUNCTION account (idSpectacle integer, searchType integer) RETURNS numeric(8,2)

Cette fonction sert à calculer la recette (*searchType*=1) ou la dépense (*searchType*=0) ou le gain (*searchType*=2) du spectacle *idSpectacle* (somme de plusieurs représentations).

FUNCTION account_by_repre (idRepre integer, tarifType integer) RETURNS numeric(8,2)

Cette fonction sert à calculer la somme d'argent totale correspondant aux billets de types différents vendu (Réduit = 1, Normal = 0 pour *tarifType*) pour de la représentation *idRepre*.

FUNCTION account_by_date(idSpectacle integer, dateFrom date, dateTo date) RETURNS numeric(8,2)

Cette fonction sert à calculer le gain du spectacle *idSpectacle* (somme de plusieurs représentations) entre la date *dateFrom* et la date *dateTo*.

FUNCTION balance_sheet(searchType integer, dateFrom date, dateTo date) RETURNS setof historique

Cette fonction fournit une liste des recettes (*searchType*=1) ou des dépenses (*searchType*=0) ou les deux (*searchType*=2) des représentations entre la date *dateFrom* et la date *dateTo*.