**Rapport du Projet de la Base de Donnée Avancé**

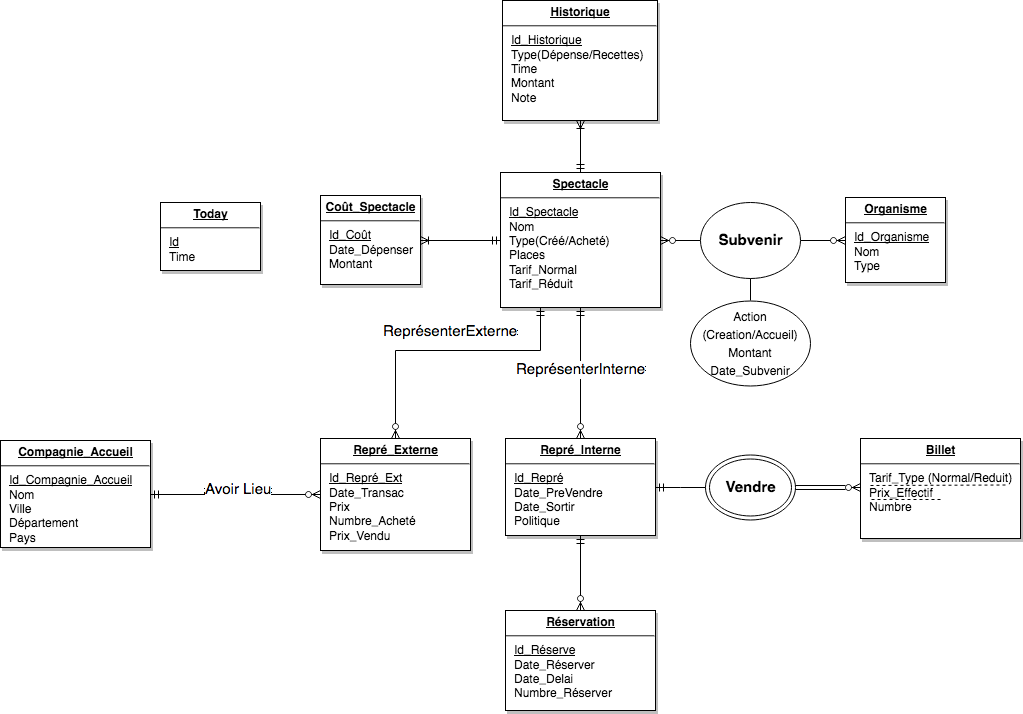
Université Paris Diderot

21502157 FAN Yi-Zhe | 21505130 HSIEH Yung-Kun

SGBD : PostgreSQL

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

**Les Diarammes 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:

  cout\_achete\_checker BEFORE INSERT OR UPDATE ON cout\_spectacle FOR EACH ROW EXECUTE PROCEDURE check\_cout\_achete()

  historique\_cout\_modifier AFTER INSERT OR DELETE OR UPDATE ON cout\_spectacle FOR EACH ROW EXECUTE PROCEDURE modify\_cout\_historique()

**Organisimes** ( Id\_Organisime, Nom, Type )

**Subvention (** #Id\_Spectacle, #Id\_Organisime, 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:

  historique\_subvenir\_modifier AFTER INSERT OR DELETE OR UPDATE ON subvention FOR EACH ROW EXECUTE PROCEDURE modify\_subvenir\_historique()

  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, Numbre )

Check constraints:

  "billet\_numbre\_check" CHECK (numbre >= 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:

  on\_billet\_changer AFTER INSERT OR DELETE OR UPDATE ON billet FOR EACH ROW EXECUTE PROCEDURE on\_billet\_change()

  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, Numbre\_Acheté, Prix\_Vendu )

Check constraints:

  "repre\_externe\_numbre\_achete\_check" CHECK (numbre\_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:

  historique\_repre\_externe\_modifier AFTER INSERT OR DELETE OR UPDATE ON repre\_externe FOR EACH ROW EXECUTE PROCEDURE modify\_repre\_externe\_historique()

  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, numbre INTEGER) RETURNS INTEGER**

Cette fonction est pour faire réservation de *numbre* place pour la représentation *idRepre*. Il renvoyais -1 si la réservation échoue, sinon il renvoyais un numéro pour la réservation.

**FUNCTION calc\_numbre\_place\_dans\_billet (idRepre INTEGER) RETURNS INTEGER**

Cette fonction est pour calculer le nombre des places vendus (places réservés exclus).

**FUNCTION calc\_numbre\_place\_dans\_reserv (idRepre INTEGER) RETURNS INTEGER**

Cette fonction est pour calculer le nombre des places réservés (places vendus exclus) pour la représentation *idRepre*.

**FUNCTION get\_seat\_sold\_percentage (idRepre INTEGER) RETURNS numeric (2,2)**

Cette fonction est pour calculer le taux des places vendus (places réservés exclus) pour la représentation *idRepre*.

**FUNCTION get\_seat\_full\_percentage (idRepre INTEGER) RETURNS numeric (2,2)**

Cette fonction est pour calculer le taux des places totales (places réservés plus vendus) pour la représentation *idRepre*.

**FUNCTION get\_current\_ticket\_price (idRepre INTEGER, tarifType INTEGER) RETURNS numeric (8,2)**

Cette fonction est pour calculer le prix de billet à ce moment selon le politique et le type de tarif (Reduit = 1, Normal = 0) de la spectacle pour la représentation *idRepre.*

**FUNCTION pay\_reservation (myIdReserve INTEGER, tarifNormal INTEGER, tarifReduit INTEGER) RETURNS numeric(8,2)**

Cette fonction est pour payer une réservation de numéro *myIdReserve*. Vous deviez indiquer combien de billet normal *tarifNormal* et combien de billet reduit *tarifReduit*.

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

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

**FUNCTION account\_by\_repre (idRepre integer, tarifType integer) RETURNS numeric(8,2)**

Cette fonction est pour calculer le somme d’argent total de billets en types différents (Reduit = 1, Normal = 0 pour *tarifType*) vendu de la représentation *idRepre*.

**FUNCTION account\_by\_date(idSpectacle integer, dateFrom date, dateTo date) RETURNS numeric(8,2)**

Cette fonction est pour calculer le gain du spectacle *idSpectacle* (somme des plusieurs représentations) entre date *dateFrom* et date *dateTo*.

**FUNCTION balance\_sheet(searchType integer, dateFrom date, dateTo date) RETURNS setof historique**

Cette fonction est pour montrer un liste de la recette (*searchType*=1) ou de la dépense (*searchType*=0) ou tous (*searchType*=2) des représentations) entre date *dateFrom* et date *dateTo*.