Introduction aux
Bases de Données

Objectifs de la formation « Introduction aux Bases de Données »

- Savoirs à acquérir :
  - définir et comprendre ce qu'est une base de données, son utilité au sein d'un système d'information et ses cas d'utilisation
  - définir et comprendre ce qu'est un système de gestion de bases de données, ses fonctionnalités et son importance au sein d'un système informatique
  - comprendre l'intérêt de la couche conceptuelle, et ses liens avec la couche logique et la couche physique
  - connaître les constructeurs et la grammaire du modèle entité-association
  - comprendre l'utilité du modèle relationnel et son fonctionnement
- Savoir-faire à acquérir :
  - savoir structurer une base de données à l'aide du modèle entité-association
  - savoir comprendre et expliquer la structure d'une base de données modélisée avec le langage entité-association
  - pouvoir transformer une base de données exprimée avec le modèle entité-association vers son schéma relationnel
  - savoir utiliser les fonctionnalités de bases de l'outil BD-Main

Cas d’utilisation

- Gestion administrative et encodage
- Système de réservation utilisant Internet: les "formulaires"
- Comptes
- Paiements (e-Banking)
- E-governement (FedICT): Tax-On-Web

Autres cas d'utilisation

- Gestion de stocks, commandes,…
- Gestion de réservations
- Enregistrement des achats dans les magasins
- Enregistrement des données citoyens
- Sites Web 2.0 (dynamique): Forum, news, RSS,…
- …

« Partout tout le temps! »

Table des matières

- Chapitre 1: Introduction
  1. Un peu d'histoire
  2. Réponses aux besoins de stocker des données (Fichiers, BD et SGBD)
  3. La notion de donnée
  4. Architecture d'un SGBD
  5. Objectifs et rôles d'une modélisation de la BD
  6. Les différents niveaux de schéma d'une BD
  7. Les contraintes
  8. Les transactions
- Chapitre 2: Le modèle Entité–Association
- Chapitre 3: Le schéma Relationnel
- (Chapitre 4: SQL)

Chapitre 1: Introduction

1. Un peu d'histoire

Un peu d'histoire

Le **concept de base de données** est né dans les années 60, avec la généralisation des **disques magnétiques** permettant l'accès direct à une donnée, contrairement aux bandes magnétiques, limitées à l'accès séquentiel.

Premier disque magnétique commercial : l'IBM 350, équipant le Ramac 305

- **Date** : de 1956 à 1969
- **Capacité** : 5 millions de mots (= 4,4 Mo)
- **Vitesse** : 1.200 rpm. Déplacement moyen : 0,6 sec (temps de commutation inconnu mais sans doute très important)
- **Géométrie** : 50 plateaux de 60 cm mais seulement 2 têtes. 100 cylindres, 100 pistes par cylindre. 5 secteurs par piste. 100 mots par secteur. 7 bits par mot.
- **Volume** : 2 m^3. Poids : 1.075 kg
- **Nombre d'exemplaires fabriqués** : 100
- **Coût de location** : 35.000 $/an (en 1956)

Un peu d’histoire: 60’

Années 1960

- 1963 : Integrated Data Store (IDS) chez Honeywell–Bull. Développé par Ch. Bachman. Ancêtre des architectures standard de SGBD.
- 1965 : Information Management System (IMS) chez IBM. Au départ conçu pour North American Rockwell pour supporter le projet du 1er vol humain vers la lune. Initialement sur bande magnétique. Sera commercialisé en 1969 sous le nom IMS/360.

- 1968 : premier projet de recommandation CODASYL, inspirée d'IDS.

Un peu d’histoire: 70’

- **Années 1970**
- 1970 : article de E. F. Codd d'IBM définissant les principes des **bases de données relationnelles** (Codd, E., F., A Relational Model of Data for Large Shared Data Banks, in *Comm. ACM*, Vol. 13, No 6, June 1970).
- 1971 : première recommandation officielle CODASYL. Les rapports de 1973 et 1978 affineront et clarifieront les propositions: ***origine de nombreux SGBD encore en activité aujourd'hui***.
- 1973 : Première définition du langage SEQUEL, qui sera renommé ***SQL***.
- 1974 : **premier SGBD** relationnel expérimental, le System/R d'IBM. Donnera naissance aux SGBD SQL/DS et DB2 dans les années 1980.
- 1975 : première version opérationnelle de INGRES, SGBD relationnel expérimental de l'université Berkeley. Langage QUEL concurrent de SQL. Donnera naissance à PostgreSQL. Des transfuges de l'équipe créeront **Sybase**.
- 1979 : première version commerciale d'un SGBD relationnel proposant SQL : ***Oracle***.

Un peu d’histoire: 80’

- **Années 1980**
- 1982 : commercialisation de SQL/DS, SGBD relationnel d'IBM (machines MV).
- 1983 : commercialisation de DB2, SGBD relationnel d'IBM (machines MVS).
- 1987 : après 10 ans de travaux, **premier standard effectif destiné aux *SGBD relationnels*.**
- Les SGBD relationnels remplacent progressivement les SGBD traditionnels (CODASYL, IMS).
- Développement de SGBD expérimentaux admettant des valeurs structurées (tables *non plates*). Ces principes seront repris par les SGBD objet et relationnels objet.
- Développement de SGBD déductifs expérimentaux, basés sur les principes de déduction de la logique. Pas de succès commercial.
- Expérimentation sur les bases de données distribuées.

Un peu d’histoire: 90’

- **Années 1990**
- 1992 : Microsoft commercialise Sybase sous le nom **SQL Server**.
- Standard SQL2 ou SQL 1992. Introduit les notions de ***primary key*** et *foreign key*.
- Développement de SGBD expérimentaux **orientés objet**. Faible succès commercial mais certains de leurs principes seront repris par les SGBD relationnels objet.
- Microsoft envisage d'intégrer SQL Server comme composant système de Windows. Ce projet ne sera réalisé qu'avec Vista.
- Des versions légères sont proposées pour les systèmes mobiles et embarqués : cartes à puce, PDA, systèmes de capteurs, etc.

Un peu d’histoire: 00’

Années 2000

- 2000 : Apparition des SGBD gérant de manière native des documents **XML**. Faible succès commercial mais certains de leurs principes seront repris par les SGBD relationnels modernes.
- Standard SQL3 ou **SQL:1999**. Introduit notamment les requêtes récursives et les structures relationnelles objet.
- **SQL 2003** en préparation. Spécifie notamment les aspects multimedia, XML, modélisation spatiale et analyse de données (fouille de données, entrepôts de données).
- Les SGBD IMS et CODASYL issus des années 1960 sont toujours en activité!

Et ensuite?

Trois questions fondamentales :

Qu'est-ce qu'une base de données ?

Comment construit-on une base de données ?

Comment utilise-t-on une base de données ?

Chapitre 1: Introduction

2. Réponses aux besoins de stocker des données: Fichiers, BD et SGBD

Approche historique:
Les systèmes de documentation papier

- Farde, papier, classeurs, intercalaires,…

Approche historique: Les fichiers informatiques

- Utilisation de **fichiers** = suite de données les unes après les autres
- Classement séquentiel = seule structure possible
- Gestion difficile
  - Sécurité
  - Accès concurrents
  - Codage explicite dans chaque application

    =>Dépendance des données!
  - Partage de données entre applications
  - Information agrégée difficile à obtenir
  - …

Les Bases de données

- Une **base de données** (BD) contient l'ensemble des données nécessaires au fonctionnement d'une organisation (ou d'une partie de l'organisation)
  Sa gestion est assurée par un logiciel appelé système de gestion de bases de données (SGBD)
- Une base de données = une collection de données "reliées" (related data)
- Auto-description du contenu dans une BD
- Indépendance des données
  - Relation conceptuelle entre les données (pas de pointer!)
    => Pas de relation directe avec les applications les utilisant
  - Pas de format particulier propre à une application particulière

Les systèmes de gestion de BD

- Un **système de gestion de bases de données** (SGBD, en anglais *DBMS*) est un (ensemble de) logiciel(s) fonctionnant sur un système hardware
- Un SGBD est responsable de mettre en œuvre toutes les manipulations réalisées sur les bases de données, tout en garantissant leur bonne gestion
- Le composant central est le moteur de base de données
- Indépendant des applications pour lesquelles le SGBD gère et stocke les données

Quelques SGBD’s

dBase (1985)

1995

1989

1979

La banque de données

- Une **banque de données** est un ensemble de données, propres à un domaine, que des "producteurs de données" réunissent pour ensuite en commercialiser son usage à un public extérieur
  - Exemple: les banques de données juridiques, économiques, médicales, des brevets, des propriétés des matériaux, …
- Concept différent d'une BD, mais devant souvent utilisé les outils de gestion de BD (SGBD)

Les propriétés d'une BD

Structuré	• Structure logique et cohérente par rapport au domaine d'application
Contextualisé	• Contextualisation des données dans un objectif précis (pourquoi créer une DB?)
Taille adéquate	• Différentes tailles pour différentes utilisations (envergure et contenu)
Organisation physique	• Différentes possibilités de mise en place de l'hardware et du software
Persistance des données	• Les données sont stockées sur un long terme

Les fonctions d'un SGBD

Définir la structure	• Construire la base de données (structure et contraintes), et définition d'une représentation des données stockées • Définition et gestion des méta données
Manipulation	• Insérer, modifier et supprimer les données (intégrité!) • Requêtes simples et complexes (Reporting)
Gérer la BD	• Organiser le stockage des données et leur pérennité • Sauvegarder et restaurer les données • Performance et optimisation de l'allocation des ressources • Droit d'accès des utilisateurs
Monitoring	• Permettre une analyse sur l'utilisation
Gérer les problèmes	• Sécurité et accès concurrents • Confidentialité des données

Chapitre 1: Introduction

3. La notion de donnée

Structure informationnelle

Donnée persistante VS. donnée transitoire

- Donnée **persistante**:
  - Stockée dans les bases de données
  - Durable dans le temps
- Donnée **transitoire**: créée par une application (au départ d'autres données) pendant son fonctionnement, et non nécessaire après son fonctionnement

Les metadonnées

- Metadonnée: "donnée sur les données"
- Définition de la structure des données stockées dans la DB
  - Le typage
  - Les identifiants
  - Un lien générique
  - Un attribut générique
- Stockées par le SGBD (data catalog & data dictionary)
  - une table répertoriant les tables de la base de données
  - une table décrivant les colonnes de ces tables
  - une table décrivant les clés (PK et FK) et une autre décrivant leurs composants
  - une table décrivant les vues
  - une table décrivant les utilisateurs et une autre décrivant les privilèges
  - et *plusieurs dizaines d'autres tables/informations …*

Chapitre 1: Introduction

4. Architecture d'un SGBD

Une architecture globale

Trois couches dans un SGBD

- Couche **externe**: dialogue avec les utilisateurs, dont les « vues » associées à chacun
- Couche **interne**
  - Stockage sur des supports physiques
  - Gestion des structures de mémorisation et d'accès
- Couche **logique**: contrôle des données et liens entre la couche externe et interne (SGBD)

Chapitre 1: Introduction

5. Objectifs et rôles d'une modélisation de la BD

Modélisation d'une BD

Un modèle de données

- Pour chaque couche (externe – logique – interne), il faut un modèle de données
- **Modèle de données**: ensemble de concepts permettant de décrire les données d'un point de vue générale, ainsi que les règles d'utilisation de ces données générales
  - Statique: structure des données et règles d'utilisation
  - Dynamique: opérations sur les données
- Travail nécessaire: *analyse des besoins* (du domaine d'application de la BD)
- Utile pour "cacher" les détails
  => Vue abstraite du domaine d'application

Besoins de description

- **Modélisation conceptuelle** (⇔ couche externe)
  - Décrire les données du monde réel (domaine d'application)
  - Capturer les données, sans référence à une solution informatique particulière
  - Utilisé par les analystes/architectes de DB
- **Modélisation logique** (⇔ couche logique)
  - Élaborer une description dans un schéma relationnel
  - Réalisé pour le SGBD
- **Modélisation physique** (⇔ couche interne/physique)
  - Création d'un script pour réaliser la base de données
  - Implémentation (organisation et types de fichiers, index, chemin d'accès,…) réalisé par les administrateurs système

Les technologies et langages

- Modélisation conceptuelle
  - Entité – Association (E–A)
  - UML diagramme de classes
- Modélisation logique
  - Schéma relationnelle
  - Modèle orienté objets
- Modélisation physique/utilisation
  - SQL
    - Data Definition Language (DDL)
    - Data Manipulation Language (DML)
    - Data Retrieval Language (DRL)

Exemple de modèle conceptuel Entité – Association

Exemple de modèle logique
Schéma Relationnel

SQL: un bref coup d'œil

- SQL (Structured Query Language)
  - Langage de haut niveau dit *déclaratif*

    (>< langages procéduraux: retrouver des données individuelles et séparément)
  - Spécification de QUOI retrouver (plutôt de comment retrouver des données)
  - Utilisation en tant que tel ou imbriqué dans des langages de programmation (ex: PHP)

Exemple de modèle physique et utilisation SQL – DDL

Exemple de modèle physique et utilisation SQL – DDL

```
create table Diplome (
    NumNational int(11) not null,
    Diplome varchar(30) not null,
    constraint ID_Diplome_ID primary key (NumNational, Diplome));

create table Participe (
    NumNational int(11) not null,
    NumeroProjet int(8) not null,
    DateDebut date not null,
    DateFin date,
    constraint ID_Participe_ID primary key (NumNational, NumeroProjet));

create table Personne (
    NumNational int(11) not null,
    Nom varchar(15) not null,
    Job varchar(50) not null,
    DateNaissance date not null,
    Ad_Rue varchar(30) not null,
    Ad_Numero int(6) not null,
    Ad_CodePostal int(4) not null,
    Ad_Ville varchar(30) not null,
    constraint ID_Personne_ID primary key (NumNational));

create table Projet (
    NumeroProjet int(8) not null,
    DureePrevue int(3),
    constraint ID_Projet_ID primary key (NumeroProjet));

create table Type (
    NumeroProjet int(8) not null,
    Type varchar(18) not null,
    constraint ID_Type_ID primary key (NumeroProjet, Type));
```

Exemple de modèle physique et utilisation SQL – DML

Personne
NumNational Nom Job DateNaissance Ad_Rue Ad_Numero Ad_CodePostal Ad_Ville
id: NumNational acc

INSERT INTO Personne VALUES (‘76031231429', 'Georges', 'Soudeur', '12-03-1976', 'Rue de l’Église', 54, 5000, 'Namur');

INSERT INTO Personne VALUES (‘81093095829', 'Lucas', 'Analyste en supply chain', '30-09-1981', 'Rue Point Carré', 389, 1020, 'Bruxelles');

Exemple de modèle physique et utilisation SQL – DRL

- Sélectionner toutes les personnes ayant dans leur job le terme ‘Analyste’
  Select NumNational, Nom, job
  From Personne
  Where Job like ‘%Analyste%’

Le cycle de vie des bases de données

Chapitre 1: Introduction

6. Les différents niveaux de schéma d'une BD

La notion de vue

- Une vue est un **schéma externe** réalisé pour un groupe d'utilisateurs déterminé (rôle commun)

  =>différentes perspectives sur le "schéma général" "(appelé schéma de communauté)
- Une vue utilise une partie du schéma de communauté et/ou le réorganise par rapport aux *tâches* à réaliser
- Avantages:
  - Simplicité d'utilisation (répond mieux aux besoins des utilisateurs)
  - Sécurisation et protection des données (contrôle d'accès)
- Redondance!

Les types de schémas

Exemple de vue/schéma externe

Schéma de communauté

Exemple de vue/schéma externe

Exemple de vue/schéma externe

NombrePlace
Projecteur
id: IdLocal

Chapitre 1: Introduction

7. Les contraintes

Les contraintes

- Contrainte d'intégrité: toute prescription sur le schéma conceptuel dû à des particularités du domaine d'application
- Valable pour toute la DB et vérifié lors de toute modification de données
- Exemples
  - Les points des étudiants sont compris entre 0 et 20
  - Deux étudiants ne peuvent avec le même numéro d'identification (*contrainte d'intégrité référentielle*)
  - Tout étudiant doit avoir plus de 18 ans
  - Un étudiant doit suivre au moins 10 cours par année académique
- Tout n'est pas exprimable par contrainte
  (ex: une faute d'orthographe dans un prénom)
- Applicable aux données, pas aux applications

Les contraintes et le SGBD

- Le respect des contraintes est assuré/contrôlé par le SGBD lors de:
  - l'insertion de données
  - la suppression de données
  - la modification de données
- Décisions possibles:
  - Refus de l'action
  - Adaptation de l'action pour respecter les contraintes établies
- Utilisation/contrôle via les métadonnées (stockées et gérées par le SBGD dans le data catalog et data dictionary)

Chapitre 1: Introduction

8. Les transactions

Les transactions

- Gestion des accès concurrents aux données car plusieurs utilisations de données en simultanée
  - Lecture & Lecture: OK
  - Lecture & Écriture: ?
  - Écriture & Écriture: KO
- Assurer la cohérence et la consistance des données
  - ***Cohérence***: les données sont en accord avec la réalité
  - ***Consistance***: les données sont en concordance les unes avec les autres

Propriétés ACID

Atomicité

- La transaction est entière réalisée ou pas du tout (« All or nothing")
- Si problème => Roll back

Consistance

- La transaction doit faire passer la base de données d'un état consistant à une autre état consistant

Isolation/indépendance

- Pas d'influence entre les données de la transaction et celles non incluses
- Pas d'effet partiel visible d'une transaction pendant son exécution

Durabilité

- Une transaction réalisée avec succès garde ses effets sur les données (une transaction réussie est permanente)

Pour conclure…
Structure conceptuelle d'un SGBD

Table des matières

- Chapitre 1: Introduction aux bases de données
- Chapitre 2: Le modèle Entité–Association
  1. Méthodologie de modélisation
  2. La notion d'Entité et de Classe d'Entité
  3. Les relations: l'Association
  4. Les Attributs
  5. Les relations: la Spécialisation–Généralisation
  6. Les contraintes d'intégrités
- Chapitre 3: Le Schéma Relationnel
- (Chapitre 4: SQL)

Chapitre 2:
Le modèle Entité–Association

1. Méthodologie de modélisation

Créer une base de données "en live"

- On **ne** crée **pas** une base de données d'une entreprise directement dans le workbench d'un SGBD!
  - Pas de communication, ou difficilement
  - Travail solitaire
  - Pas de modèle conceptuel ni logique
  - Pas de travail itératif de modélisation
  - Pas de documentation de l'implémentation (le plus souvent)
  - Source d'erreurs importantes (Rappel: en informatique, plus une erreur est découverte tôt avant l'implémentation, moins elle a de conséquences financières et temporelles)
  - …
- Nécessité d'une approche disciplinée, structurée et basée sur des techniques et modèles rigoureux

Quelques chiffres

- Une base de données de taille moyenne:
  - comporte plusieurs centaines de tables
  - comporte plusieurs milliers de colonnes
- Une base de données de grande taille:
  - comporte plusieurs milliers de tables
  - comporte plusieurs dizaines de milliers de colonnes
- SAP utilise une BD de:
  - de près de 30.000 tables
  - de près de 200.000 colonnes

Sans modèles de différents niveaux, impossible de la comprendre!

Une exemple réel (1999)

Entreprise européenne de vente par correspondance

Le problème de modélisation

Comment traduire les <u>besoins</u> exprimés par les <u>utilisateurs</u> en <u>structures de données</u> ?

- Les utilisateurs = tous les agents (personnes, organismes, programmes et applications clients) impliqués dans la collecte, la production, la consultation, l'utilisation, la transmission, le traitement et la gestion des données
- Les besoins (ou exigences): ensemble des propriétés, caractéristiques et conditions que la base de données doit respecter pour satisfaire ses utilisateurs
  - besoins *fonctionnels* : contribuent à la fonction (au métier) des utilisateurs
  - besoins *non fonctionnels* : contribuent à la qualité des services rendus aux utilisateurs
- Les structures de données = le(s) schéma(s) de la base de données

Modéliser en 2 (?) étapes

Un ouvrage est une oeuvre littéraire publiée. Il est caractérisé par son numéro identifiant, son titre, son éditeur, sa date de première parution, ses mots-clés (10 au maximum), une brève note de présentation (ces notes sont en cours de constitution), le nom et le prénom de ses auteurs. A un ouvrage correspondent un certain nombre d'exemplaires, qui en sont la matérialisation physique. ...

?

Conception de base de données

```
create database BIB
create dbspace BIB_DATA;
create table OUVRAGE (
     NUMERO char(18) not null,
     TITRE varchar(60) not null,
     EDITEUR char(32) not null,
     DATE_1RE_PARUTION date not null,
     PRESENTATION varchar(255),
     primary key (NUMERO)) in BIB_DATA;
. . .
alter table EXEMPLAIRE add constraint FKDE
     foreign key (NUMERO)references OUVRAGE;
. . .
create unique index IDOUVRAGE
     on OUVRAGE (NUMERO);
. . .
```

besoins des utilisateurs

?

Conception de base de données

code SQL DDL

Méthodologie de développement de BD

Un ouvrage est une oeuvre littéraire publiée. Il est caractérisé par son numéro identifiant, son titre, son éditeur, sa date de première parution, ses mots-clés (10 au maximum), une brève note de présentation (ces notes sont en cours de constitution), le nom et le prénom de ses auteurs. A un ouvrage correspondent un certain nombre d'exemplaires, qui en sont la matérialisation physique. ...

Schéma conceptuel

Schéma logique (relationnel)

Schéma physique (Oracle 11)

Conception de base de données

```
create database BIB
create dbspace BIB_DATA;
create table OUVRAGE (
     NUMERO char(18) not null,
     TITRE varchar(60) not null,
     EDITEUR char(32) not null,
     DATE_1RE_PARUTION date not null,
     PRESENTATION varchar(255),
     primary key (NUMERO)) in
BIB_DATA;
. . .
alter table EXEMPLAIRE add constraint
FKDE
foreign key (NUMERO)references
OUVRAGE;
. . .
create unique index IDOUVRAGE
     on OUVRAGE (NUMERO);
. . .
```

Modèle VS. Schéma

- Un **modèle**: un système formel de représentation de certains aspects de tous les domaines d'applications d'un certain type
  - Donne les constructeurs et la grammaire à suivre
  - En BD: représenter des objets, leurs attributs, leurs liens,…
- Un **schéma**: représentation des concepts statiques (et abstraits) d'un domaine d'application *particulier*
  - Un schéma peut être positionné avec différentes vues sur le domaine d'application
  - Un schéma peut être positionné à différents niveaux (ex: conceptuel, logique ou physique en BD)
- La population sont les différents instances d'un schéma particulier, à un moment particulier

Chapitre 2:
Le modèle Entité–Association

2. La notion d'Entité et de Classe d'Entité

L’Entité

- Une Entité correspond à un objet du monde réel
  - Entité physique: des personnes, des machines, des produits,…
  - Entité conceptuelle/abstraite: une société, un travail, une commande,…

La Classe d'Entités

- Une Classe d'Entité
  - Concept générique dans lequel on classe/instancie plusieurs Entités ayant des caractéristiques similaires
  - Une Classe d'Entité définit un template commun pour un ensemble d'Entité

Exercices

- Quelques exercices pour exploiter les concepts d'Entité et de Classe d'Entités

Exercice 1

- Voici trois entités :
  - *Jules Dumoulin, 25 ans, habitant rue d'Erquelines n° 18 à 5983 Mont-sur-Pieds, marié, 1 enfant*
  - *Lisa Lambert, 78 ans, habitant rue d'Emailas n°134 à 1930 Souson, veuve, 4 enfants*
  - *Henri, cheveux bruns, yeux bleus, 187cm, 56 kilogrammes, vendeur de meubles*
- Identifiez l'entité ne faisant pas partie de la même classe que les autres
  Expliquez pourquoi

Exercice 2

- Voici deux entités :
  - *Jules Dumoulin, 25 ans, habitant rue d'Erquelines n° 18 à 5983 Mont-sur-Pieds, marié, 1 enfant*
  - *Lisa Lambert, 78 ans, habitant rue d'Emailas n°134 à 1930 Souson, veuve, 4 enfants*
- Nommez la classe pouvant regrouper ces deux entités et donnez des caractéristiques partagées

Exercice 3

- Voici la description d'une Classe d'Entités:

  Film(Titre, AnneeProd, ActeurPrincipal, Langue, Réalisateur, NbrePrix)
- Voici l'extrait d'un article; donnez l'entité correspondant à la classe d'entité ci-dessus:

  Le film mythique « Dear Friends » produit par Universal dans le Kansas, Etats-Unis, en 1987 et sorti dans les salles l'année suivante (1988) vient encore de remporter un prix récemment. Ce prix, nouveau triomphe pour l'équipe de production, a été remis à la veuve du réalisateur, Thomy Lee. L'acteur José Despero était évidemment présent. Son rôle central dans le film a encore été souligné. Sa partenaire, Monique Poncin, était également présente, bien que souffrante. Ce film va probablement être adapté pour être joué au théâtre selon le producteur de la troupe théâtrale « Moments Heureux ».

Chapitre 2:
Le modèle Entité–Association

3. Les Attributs

Les attributs: définition

- Un **Attribut** d'une Classe est une caractéristique partagée par (toutes?) ses Entités
- Chaque Attribut a un domaine de valeurs (type) précis
- Chaque Entité a une valeur pour les (chaque?) attributs de sa Classe
- **Attribut atomique**

La cardinalité d'un Attribut

- Une **cardinalité**: concept mathématique définissant le nombre d'élément (ou de relations) dans/entre ensembles finis
- Soit la cardinalité [x,y]
  - x: borne inférieure; y: borne supérieure
  - Donne le nombre min *x* et max *y* de valeur que peut prendre une Entité pour cet attribut
  - $x \leq y$
- **Optionnel** VS. **Obligatoire**
  - Optionnel: $x = 0$
  - Obligatoire: $x \geq 1$
- **Multiplicité** d'une cardinalité
  - Unique (une valeur): $y = 1$
  - Multiple (plusieurs valeurs): $y > 1$
- Par défaut: [1,1]

La valeur NULL

- Si une cardinalité est optionnelle, alors on peut ne pas avoir de valeur pour cet attribut lors de l'enregistrement/modification d'une entité particulière
- *NULL* (le "rien informatique"!)
  - **Pas applicable**: ne s'applique pas à l'entité
  - **Inconnu actuellement, mais la valeur existe**: on sait que l'entité possède une valeur pour l'attribut sans encore la connaître
  - **Inconnu actuellement, la valeur existe peut-être**: on ne connaît pas la valeur et on ne sait pas si elle existe
- Concept central à toujours garder dans un coin de la tête lorsque l'on réalise des requêtes SQL DRL

  (et principalement des requêtes agrégatives, par ex. calculer un moyenne ou compter un nombre d'enregistrement)

Attribut dérivé

- Si deux attributs sont en relation, ils sont appellés **attributs dérivés**
  - Ex: Age peut-être dérivé de DateNaissance
  - Redondance!
- Garder l'attribut le plus pérenne dans le temps

Attribut composite

- Un **Attribut composite** est un attribut qui est décomposé en d'autres attributs
- Un attribut composite est donc un ensemble de valeurs mise en relation
- Un attribut atomique est donc un attribut indivisible
- C'est un groupe de valeurs qui n'est pas suffisant que pour être une entité en tant que tel

Attribut identifiant

- Un **attribut identifiant** est un attribut (simple ou complexe) dont la valeur peut à elle seule identifier chaque entité de la Classe par rapport aux autres
  - ***Unicité***: pas deux fois la même valeur pour deux entités dans la classe
  - ***Valeur obligatoire*** : pas de valeur NULL =>cardinalité: [1,1]
- Une entité peut avoir plusieurs identifiants, un seul est choisi comme attribut identifiant
- Un identifiant peut être composé de plusieurs attributs simples ou composites

Autres notations pour les différents types d'Attributs

Chapitre 2:
Le modèle Entité–Association

4. Les relations: l'Association

L'Association: définition

- Une **Association** capture une relation *statique* qu'ont des entités d'une classe avec des entités d'une autre classe
  - Se lie dans les deux sens!

Les rôle d'une Association

- Un **rôle** permet de donner la fonction de chaque Entité dans l'Association
- Optionnel (sauf pour les Associations récursives)
- Utiliser un terme neutre comme dénomination de l'Association lorsque des rôles sont indiqués

Les cardinalités d'une association

- Chaque association binaire a 2 cardinalités, une pour chacun des rôles
- Une cardinalité a une borne supérieure et une borne inférieure
- Les cardinalités de l'Association '*Travaille*' se lisent:
  - De 'Personne' vers 'Departement': "Une entité de la classe Personne a une et une seule relation Travaille avec les entités de la classe d'entités Departement"
  - De 'Departement' vers 'Personne': "Une entité de la classe Departement a entre 1 et N relation avec les entités de la classe Personne"

Caractéristiques des cardinalités des Associations

- Soit la cardinalité d'une Association (x,y)
- Optionnel VS. Obligatoire
  - **Optionnel**: x = 0
    Une entité *peut* avoir une relation avec les entités de la classe mise en relation
  - **Obligatoire**: x ≥ 1
    Chaque entité *doit* avoir une relation avec les entités de la classe mise en relation
- Trois types d'associations:
  - **One–to–One**: les deux bornes supérieures des deux cardinalités d'une association sont égales à 1
  - **One–to–Many**: une des deux bornes supérieures des deux cardinalités d'une association vaut 1, l'autre est strictement supérieur à 1
  - **Many–to–Many**: les deux bornes supérieures des deux cardinalités d'une association sont strictement supérieures à 1

Association One-to-One

- Une ‘Personne’ peut diriger un seul ‘Departement’
- Un ‘Departement’ est dirigé par une et une seule ‘Personne’

Association One-to-Many

- Chaque 'Personne' travaille pour un seul 'Departement'
- Chaque 'Departement' emploie de une à *n* 'Personnes'
- Une association One-to-Many est similaire à une association Many-to-One (lecture des associations dans les deux sens !)

Association Many-to-Many

- Chaque entité 'Personne' peut travailler pour plusieurs 'projets'
- Chaque entité 'Projet' utilise au moins une 'personne', jusque *n* 'personne'

Utiliser adéquatement les Associations

- Si la classe 'Personne' n'est pas dans le schéma

- Utiliser des relations pour capturer les liens du monde réel lorsqu'une classe de votre schéma capture la même donnée !

Les Attributs des Associations

- Un Attribut est ajouté à une Association ssi cet Attribut dépend des *deux* Classes d'Entités que l'Association lie

- Tout comme les Attributs de Classes, les Attributs d'Associations ont un domaine et une cardinalité, et peuvent être composite ou identifiant

L'Association Ternaire

- **L'association ternaire** capture un lien entre trois types d'objets identifiés dans le domaine d'application

Projection d'une Association ternaire

Association ternaire VS. Trois Associations binaires

- Un exemple avec des instances

Reunion

Personne	Projet	Local
Georges	Modélisation n°1	Salle 17
Lucas	Modélisation n°1	Salle 18
Georges	Modélisation n°2	Salle 18

Participe

Personne	Projet
Georges	Modélisation n°1
Lucas	Modélisation n°1
Georges	Modélisation n°2

Accueille

Personne	Local
Georges	Salle 17
Lucas	Salle 18
Georges	Salle 18

Est discute

Local	Projet
Salle 17	Modélisation n°1
Salle 18	Modélisation n°1
Salle 18	Modélisation n°2

On peut inférer des relations dérivées avec les trois associations binaires qui n'existent pas dans la relation ternaire

Modélisation correcte d'une relation ternaire avec des associations binaires

L'Association Récursive
(ou Association cyclique)

- **L'Association Récursive** modélise un lien que possède une entité d'une classe avec les autres entités de cette même classe
- Les rôles sont obligatoires!
- Il faut surveiller les **cycles/boucles** et éventuellement les interdire!

Un exercice d'illustration

- Modélisez la relation récursive suivante: on identifie des employés dont certains sont supervisés par d'autres. Chaque employé ne peut être supervisé que par maximum un autre employé

- *Quel est le gros risque de cette modélisation?*

La Classe d'Entité faible

- Une **Classe d'Entité faible** est identifiée par une Association que cette Classe a avec une autre Classe d'Entité dite forte, et *éventuellement* un ou plusieurs de ses attributs
- L'Association doit avoir une cardinalité (1,1) du côté de la Classe d'Entité faible
- *Si une entité forte disparaît, toutes les entités faibles y étant liées disparaissent aussi*

Autres notations pour l'Association

L’importance de la sémantique en modélisation

- La sémantique externe d’un modèle = **correspondance** entre chaque **construction** du modèle et les objets/**concepts du domaine d'application** qu'il représente

  => permet d'interpréter le schéma en termes accessibles aux utilisateurs

  Doit être précis et complet !

- La sémantique externe intéresse le concepteur, le développeur et l'utilisateur

Un exemple de schéma Entité–Associations

- Personne =>Employe
- *Truc & Astuce*: la dénomination à utiliser est souvent donnée dans les phrases concernant le cas à modéliser

Exercices

- Quelques exercices pour exploiter les bases du modèle Entité–Association

Chapitre 2:
Le modèle Entité–Association

5. Les relations: la Spécialisation–Généralisation

Des Entités ayant plusieurs types

- Des entités peuvent avoir plusieurs types, c-à-d appartenir à plusieurs Classes d'Entités

La relation de Spécialisation-Généralisation

Mécanismes d'héritage dans les relations de Spécialisation – Généralisation

- Les **Classes de Spécialisation** héritent de toutes les caractéristiques des Classes de Généralisation
  - Les Attributs et Identifiants
  - Les Associations
- Les Classes de Spécialisations peuvent avoir des caractéristiques propres
  - Des Attributs et Identifiants
  - Des Associations
- Le **mécanisme d'héritage** est une conséquence de la propriété d'inclusion des populations entre les *surtypes* et les *sous-types* d'une entité particulière

Mécanisme d'héritage: Illustration

La cardinalité des relations de spécialisation – généralisations

- Une cardinalité peut être:
  - Total *ou* Partiel
  - Exclusive/Disjoint *ou* chevauchement ('overlapping')
- **Total**: toutes les Entités de la classe parent doivent être spécialisées
  Partiel: les entités de la classe parent peuvent être spécialisées (ou pas!)
- **Exclusive**: une entité spécialisée ne peut l'être qu'une seule fois
- **Chevauchement**: une entité spécialisée peut l'être dans plusieurs classes enfants

(T,E)
Partition

(T,O)

(P,E)

(P,O)

Utilisation de prédicats dans les spécialisation

- On peut utiliser un ou plusieurs attributs de la classe parent pour spécifier la classe de spécialisation

Spécialisation multiple

- Une classe d'entités peut à la fois être une classe de spécialisation et une classe de généralisation

Spécialisation multiple & Héritage multiple

- Des officiers sont à la fois des personnes spécialisées comme militaires et des personnes classifiées comme diplômés

- Des militaires diplômés peuvent ne pas être des officiers

Mauvaise utilisation de la relation de Spécialisation – Généralisation

- *Attention*: ne pas confondre une relation d'Association avec une relation de Spécialisation – Généralisation!

Autres notations pour la relation de Spécialisation – Généralisation (1/2)

Autres notations pour la relation de Spécialisation – Généralisation (2/2)

Chapitre 2:
Le modèle Entité–Association

6. Les contraintes d'intégrités

Les contraintes d'intégrités

- Une **contrainte d'intégrité** est propriété d'un schéma E–A que des entités (instances) doivent respecter
  - lors l'insertion de l'entité (= contrainte statique/de structure)
  - lors de la modification et la suppression de l'entité (= contrainte dynamique)
- Une contrainte d'intégrité est valable pour tous les niveaux de schéma (conceptuel, logique et physique) même si l'outil de modélisation ne permet pas *directement* sa définition

Grade = {"Low", "Middle", "High"}

DateNaissance ≥ (1/1/1885)

DateDebut < DateFin

Les cycles dans les Associations (Récursives)

- Il faut toujours indiquer une contrainte d'intégrité dans une Association Récursive pour éviter les boucles
- On peut également retrouver des cycles dans les associations binaires modélisées sous la forme d'une boucle

Lire un Schéma E-A (1/3)

Lire un Schéma E-A (2/3)

Lire un Schéma E-A (3/3)

Exercices

- Quelques exercices pour approfondir les connaissances du modèle Entité–Association

Table des matières

- Chapitre 1: Introduction aux bases de données
- Chapitre 2: Le modèle Entité–Association
- Chapitre 3: Le Schéma Relationnel
  1. Le Schéma Relationnel: définition et concept
  2. Les types de contraintes
  3. Les règles de traductions de l'E–A vers le schéma relationnel
- (Chapitre 4: SQL)

Chapitre 3: Le Schéma Relationnel

1. Le Schéma Relationnel: définition et concept

Le Schéma relationnel: Définition

- Une base de données relationnelle = ensemble de relations de 2 dimensions
  - Un enregistrement dans une table
  - Un référence entre deux enregistrements via la redondance d'information

    =>Mécanisme de clé primaire – clé étrangère
- Le schéma relationnel définit la manière dont on pourra utiliser la base de données par après
  - SQL DDL (contraintes spécifiques)
  - SQL DML
  - SQL DRL

Exemple: E–A => schéma relationnel

Dirige
NumNational
Nom
PrimeSalariale
id: NumNational
ref acc
id': Nom
equ acc

Personne
NumNational
Nom
Job
DateNaissance
Adr_Rue
Adr_Numero
Adr_CodePostal
Adr_Ville
Tra_Nom
id: NumNational
acc
equ: Tra_Nom
acc

Diplome
NumNational
Diplome
id: NumNational
Diplome
acc
ref: NumNational

Participe
NumNational
NumeroProjet
DateDebut
DateFin[0-1]
id: NumNational
NumeroProjet
acc
equ: NumeroProjet
acc
ref: NumNational

Projet
NumeroProjet
DureePrevue[0-1]
Se_compose
id: NumeroProjet
acc
ref: Se_compose
acc

Type
NumeroProjet
Type
id: NumeroProjet
Type
acc
equ: NumeroProjet

Departement
Nom
id: Nom
acc

Reunion
IdLocal
NumNational
NumeroProjet
Heure
id: NumNational
IdLocal
NumeroProjet
acc
equ: NumeroProjet
acc
ref: NumNational
ref: IdLocal
acc

Manager
NumNational
Grade
id: NumNational
ref acc

Technicien
NumNational
PrixHeureSup
Sup_NumNational
id: NumNational
ref acc
ref: Sup_NumNational
acc

Entretien
IdLocal
NumNational
id: NumNational
IdLocal
acc
ref: NumNational
equ: IdLocal
acc

Local
IdLocal
NombrePlace
Projecteur
id: IdLocal
acc

Le mécanisme clé primaire – clé étrangère

- Une **clé primaire** identifie de manière unique tout enregistrement d'une table par rapport aux autres enregistrements
- Deux propriétés:
  - Aucune valeur NULL acceptée
  - Tout nouvel enregistrement (ou modification de valeur d'un ancien enregistrement) doit avoir une valeur unique pour sa clé primaire
- Une clé primaire peut être composé de plusieurs colonnes
- La clé maximale = l'ensemble des colonnes d'une table (on ne peut pas enregistrer deux fois le même objet!)
- La clé minimale = clé primaire
- La **clé étrangère** traduit une relation (Association ou Spécialisation) avec une autre table en référençant la clé primaire de cette table
- Contrainte d'intégrité référentielle:
  - La clé étrangère doit avoir le même domaine que la clé primaire qu'elle référence
  - Chaque valeur enregistrée dans la clé étrangère doit être une valeur existante de la clé primaire
- Clé étrangère = clé secondaire (autre appellation)

Chapitre 3: Le Schéma Relationnel

2. Les types de contraintes d'intégrité

Les types de contraintes d'intégrités

- **Contrainte d'intégrité d'une clé primaire**: toute clé primaire ne peut être NULL et doit avoir une valeur distincte des valeurs existantes
- **Contrainte d'intégrité référentielle**: mécanisme de liens à l'aide d'une clé primaire et d'une clé étrangère donnant un domaine et une liste de valeurs identiques entre le clé étrangère et sa clé primaire (ex: le type d'une colonne)
- **Contrainte d'intégrité de domaine**: contrainte définissant l'ensemble des valeurs que les champs d'une colonne peuvent prendre
- **Contrainte de ligne**: contrainte vérifiée lors de l'insertion/modification d'un enregistrement
- **Contrainte de colonne**: contrainte vérifiée pour toute valeur d'une colonne (à l'insertion et/ou à la modification)

Violation de contraintes

- Si une insertion/modification viole une contrainte, soit:
  1. Rejet de l'insertion
  2. Correction de l'insertion (l'élément de l'enregistrement violant la contrainte est adapté automatiquement)
- Si une suppression viole la contrainte d'intégrité référentielle, soit:
  1. La suppression est interdite
  2. On indique une valeur NULL comme valeur dans les clés étrangères
  3. La suppression est réalisée en cascade: tous les enregistrements référençant la clé primaire supprimée sont eux aussi supprimés!!

Chapitre 3: Le Schéma Relationnel

3. Les règles de traductions de l'E-A vers le schéma relationnel

La liste suivante énumérant les règles de transformation n'est pas exhaustive!
Les règles les plus simples sont toujours privilégiées si plusieurs existent

Traduire une classe d'entité

- Toute classe d'entités devient une table dans laquelle les attributs deviennent les colonnes de la table.
- L'(es) identifiant(s) de la classe d'entités devient(nent) la clé primaire de la table

Local

IdLocal NombrePlace Projecteur

Traduire un attribut multiple ou composite

- Un attribut multiple est transféré dans une nouvelle table et référencé à l'aide du mécanisme de clé primaire – clé étrangère
- Une attribut composite est désagrégé

Personne

NumNational Nom Job DateNaissance AdRue AdNumero AdCodePostal AdVille

Diplome

NumNational IntituléDiplome

Personne
NumNational Nom Job Diplome[0-N] DateNaissance Adresse Rue Numero CodePostal Ville
id: NumNational

Traduire une Association binaire One-to-Many

- Une association binaire de type one-to-many disparaît au profit d'une clé étrangère dans la table du côté de la cardinalité (0,1) ou (1,1). Cette clé étrangère référence la clé primaire de la table ayant une cardinalité (0,n) ou (1,n)
- Les éventuels attributs de l'association deviennent des colonnes de la table du côté de la cardinalité (0,1) ou (1,1)
- La clé étrangère ne peut recevoir une valeur NULL que si la cardinalité était optionnelle, c-à-d : (0,1)

Traduire une Association binaire One-to-Many

Traduire une Association binaire One-to-One

- Une association binaire one-to-one est traduite comme une association de type one-to-many
- On privilégie toujours la table du côté de la cardinalité obligatoire, c-à-d que la clé étrangère est ajouté du côté de la cardinalité obligatoire

Traduire une Association binaire One-to-One

Traduire une Association binaire Many-to-Many

- Une association binaire de type many-to-many disparait au profit d'une table supplémentaire dont la clé primaire est formée des deux clés étrangères
- Les attributs éventuels de l'association deviennent des colonnes dans la nouvelle table

Traduire une Association binaire Many-to-Many

Traduire une Association n-aire

- Une association non-binaire est traduite par une table supplémentaire dont la clé primaire est composée d'autant de clés étrangères que de classes d'entités en association
- Les attributs éventuels de l'association deviennent les colonnes de cette nouvelle table

Traduire une relation de spécialisation – Généralisation

- Les relations de spécialisation – généralisation doivent être désolidarisées (tout en gardant une relation dans les tables)
  - Si la cardinalité est Totale: conservation des classes enfants et suppression de la classe parent
    Les attributs/associations de la classe parent sont transmis à chacune des classes enfants
  - Si la cardinalité est Partielle:
    - Conservation de la classe parent
    - OU, création d'une nouvelle classe enfant "autre"
- *Il faut toujours garder à l'esprit qu'il faut toujours permettre la même expressivité après la transformation*

Traduire une spécialisation Totale

Traduire une spécialisation Totale

Traduire une spécialisation Partielle

Traduire une spécialisation Partielle

Traduire une spécialisation Partielle

Traduire une spécialisation Partielle

Exercices

- Quelques exercices pour utiliser les règles de traduction de l'Entité–Association vers le Schéma Relationnel