ENSIMAG-2

Éléments d'histoire de l'informatique

Sacha Krakowiak

Université Grenoble Alpes & Aconit

11. Histoire des bases de données

Donnée, information, connaissance

- Donnée
  - Description élémentaire d'une réalité
  - Peut prendre des formes diverses selon son type
    - nombre, chaîne de caractères, image, etc.
- Information
  - Donnée ou ensemble de données interprétées (combinées, mises en contexte)
- Connaissance
  - Information dotée d'un sens (dans un univers logique), permettant de modéliser la réalité, de guider une action

Les frontières entre ces notions sont souvent floues…

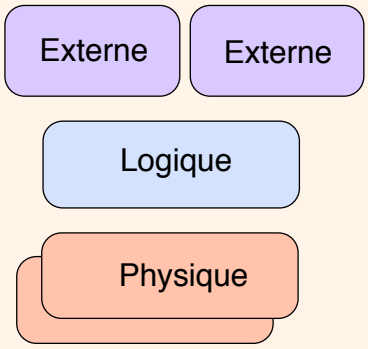
De quoi parle-t'on ?

Qu'est-ce qu'une base de données ?

Un ensemble de données organisé en vue de sa conservation, de sa mise à jour et de sa consultation

Des opérations sur une base de données permettent d'en extraire des informations, voire des connaissances

Les trois aspects des bases de données



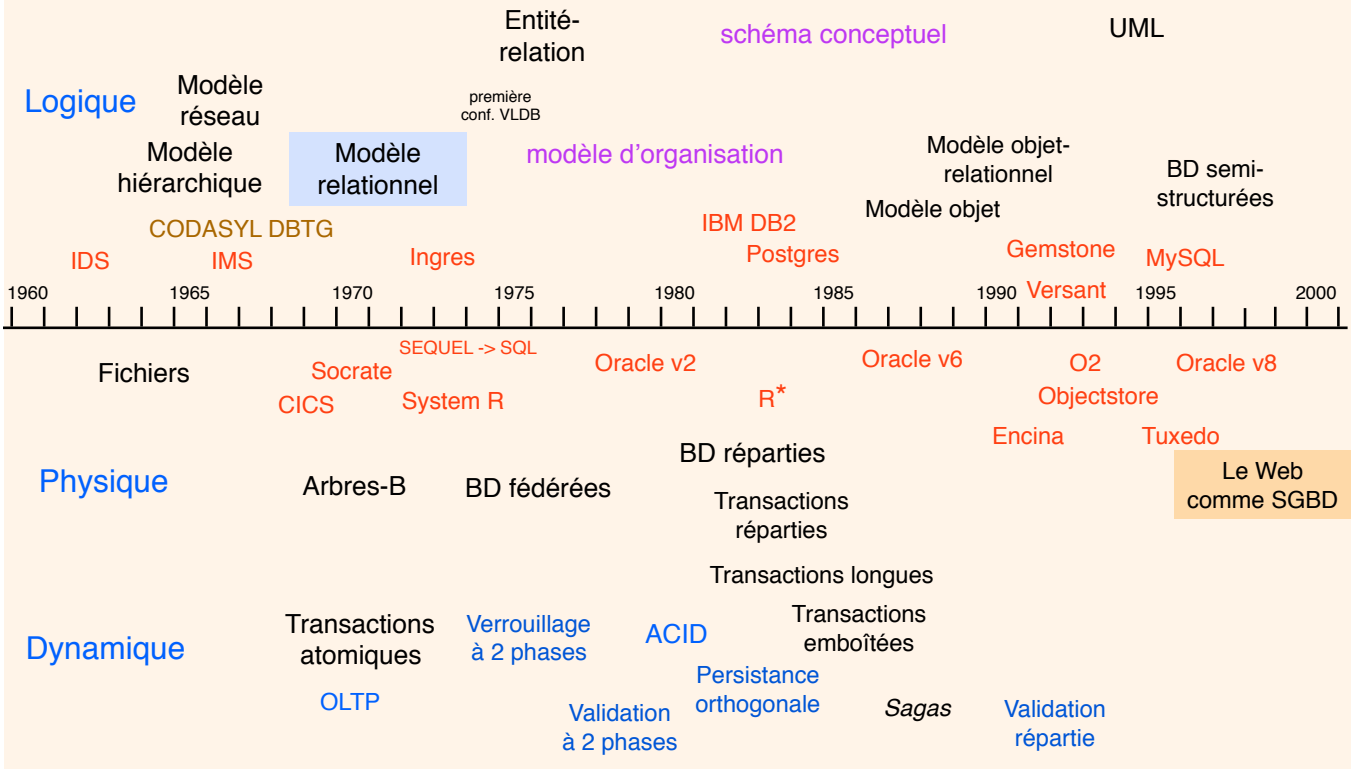
Externe : définit différentes «vues» sur les données, notamment selon divers niveaux d'abstraction

Logique : définit l'organisation des données et les opérations que l'on peut réaliser sur ces données

Physique : définit le mode de représentation et de conservation des données et la mise en œuvre des opérations

Un aspect supplémentaire : Dynamique : définit les modalités d'exécution d'actions complexes pour respecter des critères de qualité (sécurité, tolérance aux fautes, etc.)

Brève histoire des bases de données

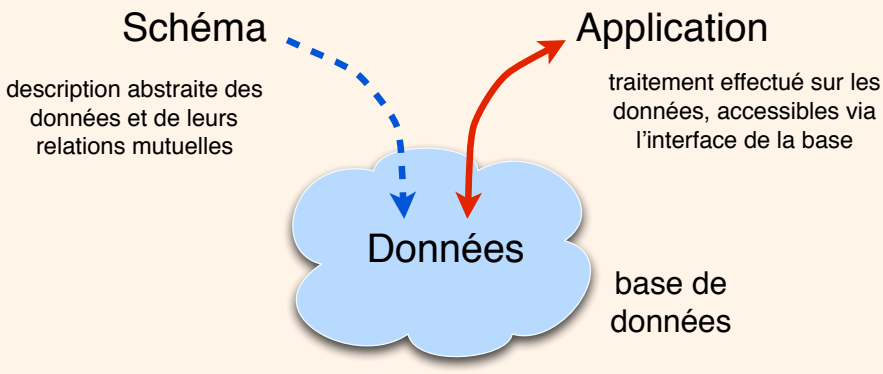


Vue d'ensemble d'une base de données

- Schéma conceptuel
  - Définit les données et leurs relations mutuelles
    - Souvent représenté sous forme graphique (ER, UML)
  - Vue
    - Définit la perception du schéma pour un utilisateur (ou un mode d'utilisation) particulier ; c'est une restriction du schéma conceptuel
- Représentation des données
  - Organisation des données selon le modèle de la base de données (hiérarchique, réseau, relationnel, objets, etc.)
- Opérations
  - Mise à jour, consultation, extraction, etc.
- Implantation physique
  - Sous le contrôle de l'administrateur de la base de données
  - En principe invisible aux utilisateurs

Un principe fondamental

- L'indépendance des données

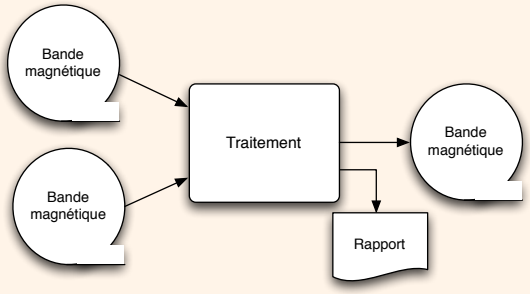


Indépendance physique :
Un changement de l'organisation et de la représentation physique des données n'affecte pas les programmes d'application

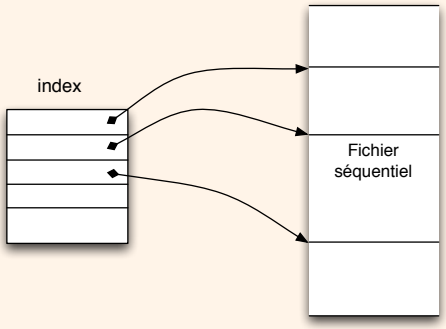
Indépendance logique :
Un changement du schéma des données n'affecte pas les programmes d'application, tant que leur vue des données n'est pas modifiée

La préhistoire

- À l'origine….
  - Bandes magnétiques (1950)
    - donc fichiers séquentiels
  - Exemples de traitement
    - Tri, fusion
    - Mise à jour
    - Recherche et extraction
  - Avancées
    - Génération automatique des programmes (RPG)



- Plus tard…
  - Disques magnétiques (1960-62)
    - Accès direct
  - Avancées
    - Accélération de l'accès
    - Hachage ou indexation



Exemple de traitement de fichiers

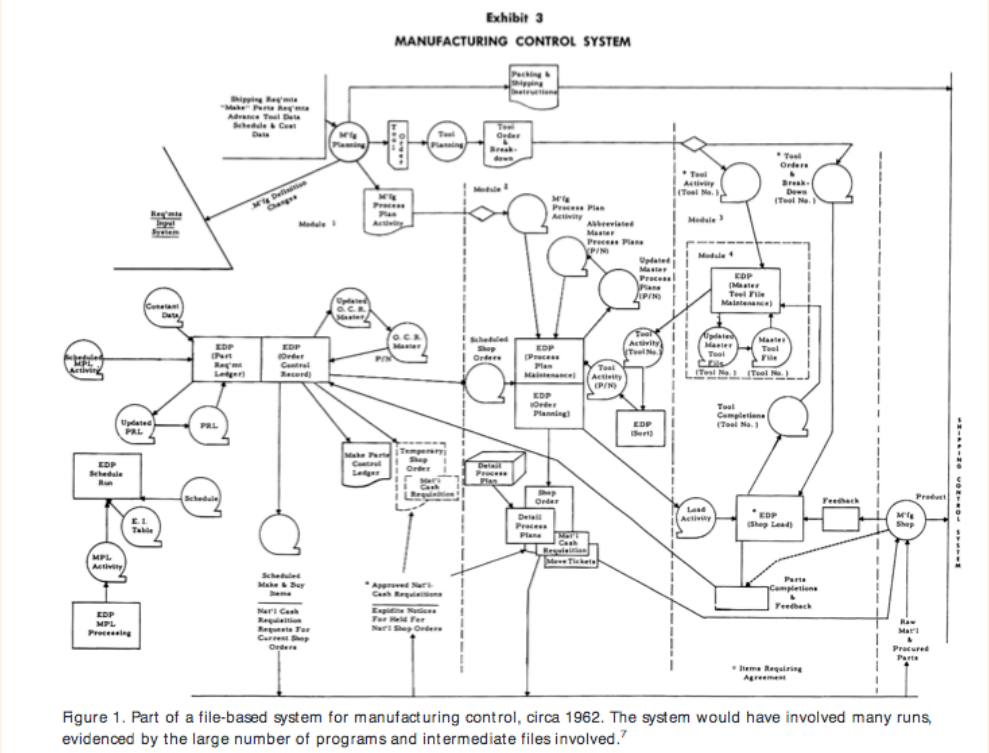


Figure 1. Part of a file-based system for manufacturing control, circa 1962. The system would have involved many runs, evidenced by the large number of programs and intermediate files involved.[7]

Extrait de : A. D. Meacham and V. B. Thompson, eds. *Total Systems*, American Data Processing, 1962, p. 153

Les premiers modèles

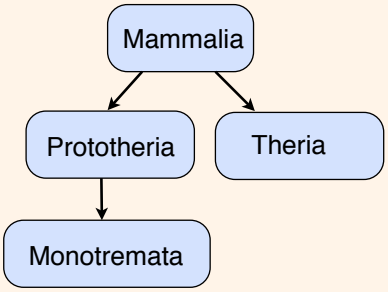
Le modèle hiérarchique

Une organisation simple
mais peu adaptable

Supprimer un nœud supprime
sa descendance

Le premier SGBD d'IBM (IMS, 1966)

Toujours utilisé (notamment avec XML)



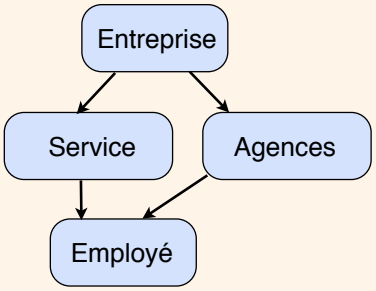
Le modèle réseau

Lève des limitations du modèle hiérarchique

Organisation plus souple…

… mais gestion complexe des pointeurs

Le premier SGBD de l'histoire (IDS, 1962)



IDS (*Integrated Data Store*)

le premier SGBD

Un modèle réseau

«*The programmer as navigator*»

Des opérations d'accès (intégrées à un langage)

Définition de données
Manipulation de données

Pas d'indépendance des données

Clés d'accès liées aux adresses sur disque
Contrepartie : bonnes performances

Développement

General Electric (Charles Bachman), gestion de production
Version 1 : 1961-62, version 2 : 1963-64

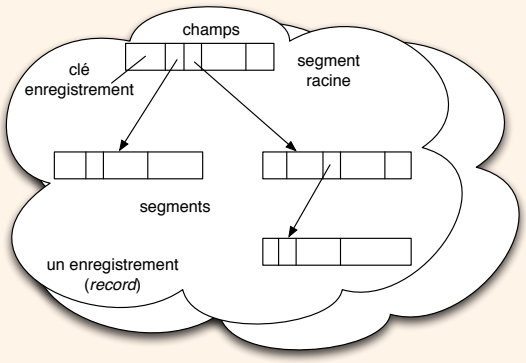
Influence

Base pour le travail de CODASYL *Data Base Task Group*
Base du développement d'IDMS, utilisé jusque dans les années 90

Charles Bachman
(1924 -)

IMS (*Information Management System*)

- Un modèle hiérarchique
- Une interface d'accès
  - DLI (*Data Language Interface*)
  - Transactions en ligne (DC, *Data Communication*)



- Fonctionnement interne
  - Index primaires (clé) et secondaires (champs)
- Développement
  - IBM et Rockwell (1966-68), pour le programme Apollo
  - A évolué jusqu'à aujourd'hui
  - Toujours utilisé (performances)

Les arbres-B

clés d'une implantation physique efficace

- Accélérer l'accès aux données
  - Une structure de données en arbre équilibré
  - Coût des opérations (recherche, insertion, suppression) : log *n* pour un arbre à *n* nœuds
  - Rudolf Bayer, Edward McCreight (1971)
- Paramètres
  - Nombre min et max de clés par nœud
    - pour les nœuds internes : nb de fils = nb de clés + 1
- Utilisation pratique
  - Les clés servent d'index pour le placement des données sur disque
  - Arbres B+ : facilitent l'accès séquentiel
    - index dans les nœuds feuilles

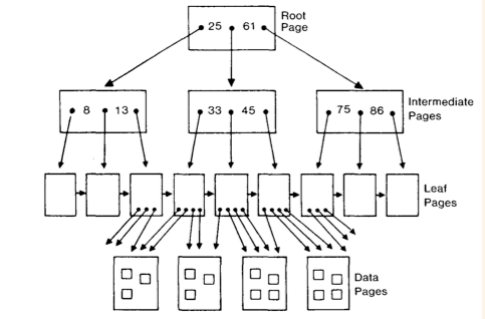


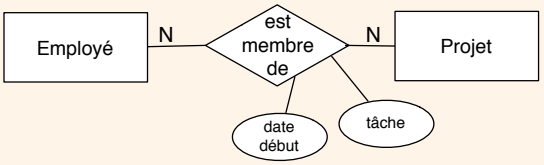
Fig. 6. A B-Tree Index.

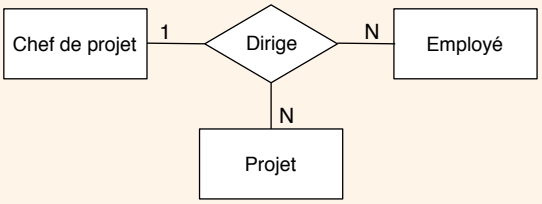
D. Chamberlin et al. History and Evaluation of System-R, *Comm. ACM* vol. 24, nr 10 (1981)

Le schéma conceptuel

- Schéma entité-association (Peter Chen, 1976)

  Exemples





id-employé, id-projet -> id-chef-projet

- Schéma UML (1995)

  Fusion de plusieurs travaux existants

  UM (Grady Booch), OMT (James Rumbaugh)

  Use Cases (Ivar Jacobson)

- Rôle des schémas

  Aide à la conception

  formaliser les entités, relations et contraintes

  Guide pour la construction des bases de données

Peter Chen

Le modèle relationnel (1)

- Un nouveau modèle de données

  Edgar Codd (IBM, 1970)

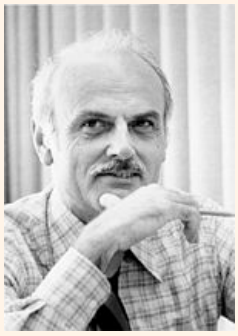
  Fondé sur des tables (relations) et des clés

- Des avantages

  Une base mathématique : l'algèbre relationnelle

  L'indépendance des données

- Un défi : l'efficacité



Edgar Codd
(1923-2003)

Source : Wikipedia, fair use

Nom	Prénom	Lieu	Service	clé	…
Martin	Jeanne	5	Études	11	
Duval	Jacques	3	Après vente	12	
Bernard	Pierre	4	Commercial	13	
Lefèvre	Paule	1	Après vente	15	
…	…				

Lieu	Ville	Adresse	…
1	Bordeaux	…	
3	Paris	Centre	
4	Paris	Défense	
5	Grenoble	…	

Le modèle relationnel (2)

- Le calcul relationnel

  Base : logique du premier ordre

  Une requête :

$$\exists \text{ nom, prénom, ville (Employés (nom, prénom, «Commercial», lieu)} \wedge \text{Localisation (lieu, ville))} ?$$

- Une traduction en SQL (cf plus loin)

```
SELECT (Nom, Prénom, Ville)
    FROM (Employés, Localisation)
WHERE (Employés.lieu)=(Localisation.lieu)
    AND Service=«Commercial»
```


Le modèle relationnel (3)

- L'algèbre relationnelle

  Permet une réalisation concrète des requêtes exprimées en calcul relationnel

- Les opérateurs de base

  Sélection, Projection — une table

  Union, Différence, Produit cartésien — deux tables

- Les opérateurs dérivés

  Jointure, Intersection

- Une contrainte de cohérence : l'intégrité référentielle

  Toute information référencée par une table A dans une table B doit exister dans la table B (exemple : clé étrangère)

Le modèle relationnel (4)

- La normalisation
  - Permet de réduire ou d'éliminer la redondance des données
  - Fondée sur les «dépendances fonctionnelles»
    - si A et B sont des ensembles d'attributs :
    - A -> B si une valeur de A correspond à une valeur unique de B
    - Exemple : clé-> {nom, prénom} de façon unique
- De nombreuses «formes normales»
  - En pratique : la forme normale de Boyce-Codd est la plus courante
    - Pour toute dépendance fonctionnelle X -> A, X est une «super-clé» (attribut ou ensemble d'attributs qui détermine une entité (table))
  - La conception d'une BD relationnelle à partir d'un schéma entité-relation aide à obtenir une BD sous forme normale

Le modèle relationnel (5)

- Un langage de requêtes : SQL (*Structured Query Language*)
  - Raymond Boyce, Donald Chamberlin (System-R, IBM, 1974)
  - Un langage déclaratif
    - Définition de données (pour construire les tables de relations)
    - Recherche de données selon des critères
    - Manipulation de données (pour insérer, supprimer, modifier les données)
    - Transactions, sécurité
  - S'appuie sur le calcul relationnel (logique du premier ordre)
- Exemple

```
SELECT (Nom, Prénom, Ville)
      FROM (Employés, Localisation)
WHERE (Employés.lieu)=(Localisation.lieu)
      AND Service=«Commercial»
```

- Utilisation
  - Directe, ou via une interface d'un langage de programmation

Le modèle relationnel (6)

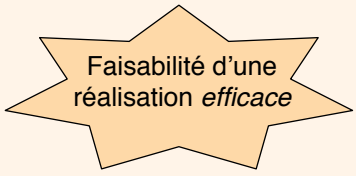
- L'optimisation de requêtes
  - Une nécessité vitale
  - Un (difficile) problème d'ingénierie
  - Une fonction (en général) invisible aux utilisateurs
- Pourquoi c'est difficile
  - De très nombreuses manière d'exécuter une requête complexe
    - L'explosion combinatoire
  - Des durées d'exécution très variables, et difficiles à estimer
- Principes d'une solution
  - On ne recherche pas l'optimum, mais un temps «raisonnable»
  - Le facteur clé : l'ordre d'exécution des jointures
  - On utilise la programmation dynamique
    - idée de base : System-R (Patricia Selinger et al., IBM, 1979)

La transition vers le modèle relationnel

- 1970 : définition du modèle
  - Edgar Codd (IBM)
  - Scepticisme sur les performances
  - Opposition de la hiérarchie d'IBM

> "It's easier to get a venture capitalist to give you money than to persuade the management of a large, successful company to try something new."
> Gordon Moore

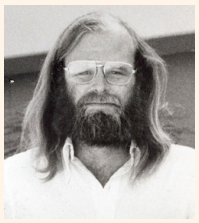
- Années 1970 : systèmes expérimentaux
  - Ingres (Michael Stonebraker, Berkeley, 1974)
  - System-R (IBM Research, San Jose, 1974)
    - invention de SQL



- Années 1980 : premiers produits commerciaux
  - Oracle v2 (SDL, puis *Relational Software, Inc.*, 1979)
    - RSI devient Oracle Corp. en 1982
  - IBM DB2 (1983)
    - première expérience avec SQL/DS, fondé sur System-R, 1981
  - Ingres (Ingres Corp., 1981)

IBM System-R

- IBM Research (San Jose), 1973
  - Montrer la viabilité d'un SGBD relationnel
    en termes de fonctions et de performances
    - Phase 0 (1974-75) SQL pour un usager
    - Phase 1 (1976-77) Système complet
    - Phase 2 (1978-79) Évaluation
- Structure à deux niveaux
  - Système de stockage (multi-usagers)
    - Index en arbres-B, verrous, reprise
  - Système d'accès (compilateur de SQL)
    - modules compilés appelés à l'exécution
- Des avancées déterminantes
  - SQL et sa compilation
  - L'optimisation de requêtes
  - Les mécanismes transactionnels et l'accès concurrent
  - Les bonnes performances



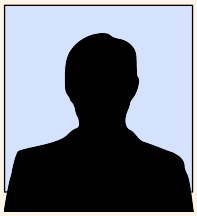
Jim Gray

Donald Chamberlin

Image courtesy of Computer History Museum

Patricia Selinger

Image courtesy of Computer History Museum



Raymond F. Boyce

Ingres

Interactive Graphical and Retrieval System

- Univ. California Berkeley (1973)
  - M. Stonebraker, E. Wong
- Un prototype de recherche
  - Sur DEC PDP-11 puis Vax (Unix)
    - d'où diffusion dans les laboratoires de recherche
      - 300 utilisateurs en 1978
  - Un système proche de System-R. Différences :
    - Le langage de requêtes (QUEL), utilisé depuis C
    - Le stockage des données (utilise un SGF)
    - L'interface d'accès (graphique interactive WYSIWYG)
- L'aventure industrielle
  - 1980 : *Relational Technology, Inc.* puis *Ingres Corp.*
    - rachetée par ASK puis Computer Associates, 1990

Michael Stonebraker

CC-BY-SA 4.0 by David Monniaux

Eugene Wong

CC-BY-SA Engineering and Technology History Wiki

Les SGBD relationnels

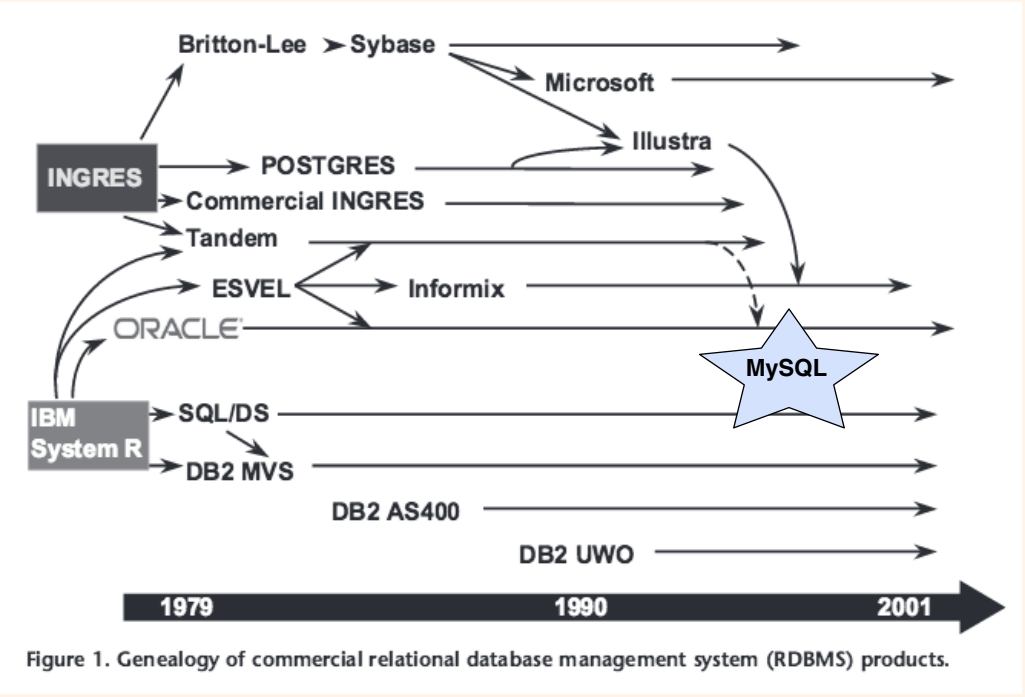


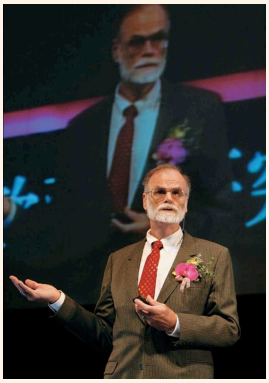
Figure 1. Genealogy of commercial relational database management system (RDBMS) products.

Extrait de : Andrew Mendelsohn. The Oracle Story: 1984-2001, *IEEE Annals of the History of Computing*, vol. 35, No 2, April-June 2013, pp. 10-23. By permission of IEEE for nonprofit educational use.

Les bases de données à objets

- Au confluent de deux tendances (vers 1985)
  - Persistance des données dans les langages de programmation
    - Persistance «orthogonale»
  - Gestion d'objets complexes pour les nouvelles applications
    - CAO, données géographiques, édition, etc.
- Principes
  - Objets (types, classes, instances, héritage) persistants
  - Accès rapide par pointeurs (navigation)
- Réalisations
  - Prototypes de recherche (mi-années 80) : Exodus, Orion, …
  - Entreprises (années 90) : Gemstone, Versant, O2 Technology, …
- Un succès partiel
  - Un marché de niches
  - Systèmes objet-relationnel (Postgres, …)

Les transactions



Jim Gray
(1944-2007)

- Une séquence complexe d'actions…
  - … ayant les mêmes propriétés qu'une action unique
  - On peut ainsi *raisonner* sur les transactions
- Caractéristiques d'une transaction
  - Deux aspects de l'atomicité
    - exécution «tout ou rien» (y compris en cas de panne)
      - notion de validation (point de non-retour)
    - indépendance entre actions concurrentes (isolation)
      - A || B a l'effet de A ; B ou B ; A
  - Cohérence (dépend de l'application)
    - permet de passer d'un état cohérent à un autre état cohérent
  - Durabilité (ou permanence)
    - une fois la transaction validée (rendue définitive), ses résultats sont préservés

Atomicité, Cohérence, Isolation, Durabilité : les propriétés ACID (Gray, 1981)

Le contrôle de la concurrence

- Un objectif : la sérialisabilité
  - Garantir que deux transactions concurrentes auront le même effet que si elles s'exécutaient en série
- Les verrous, mécanisme de base
  - Deux problèmes : assurer la sérialisabilité, éviter l'interblocage
- Un beau résultat
  - Equipe IBM, 1976 (Eswaran, Gray, Lorie, Traiger)
  - Sous certaines conditions (faciles à remplir) :
  - Le verrouillage à deux phases garantit la sérialisabilité
    - phase 1 : verrouillage, phase 2, déverrouillage
- Traiter l'interblocage
  - Prévenir ou guérir

La tolérance aux fautes

- Objectif : garantir la cohérence et l'intégrité des données…
  - … y compris en cas de défaillance
- Principe : la journalisation (Gray, 1978)
  - On doit disposer d'une mémoire stable (résistante aux défaillances)
  - Un «journal» (*log*) en mémoire stable enregistre chaque opération…
    - … avant son exécution (pour pouvoir la «défaire» si nécessaire)
- En cas de défaillance…
  - On utilise le journal
    - pour «refaire» toutes les transactions pour assurer leur prise en compte
      - (mise à jour des données sur disque)
    - pour «défaire» toutes les transactions non validées
- Dans la pratique
  - Un mécanisme complexe et délicat

Transactions réparties

- Pourquoi des transactions réparties ?
  - Les données elles-mêmes sont réparties
    - pour assurer la disponibilité
    - pour améliorer les performances
  - Réalisation : des gérants de transactions (par site) coopérants
- Un problème : la validation atomique
  - Tous les sites doivent prendre la même décision (valider ou annuler)
  - Situation difficile en cas de défaillance, car risque d'incohérence
- Une solution : la validation à deux phases (Gray, 1978)
  - En pratique, des variantes plus élaborées pour éviter le blocage
- Des produits
  - Oracle, IBM, …

Modèles avancés de transactions

- Le modèle «traditionnel» suppose des transactions courtes…
  - … mais les applications complexes actuelles peuvent durer des jours (notamment les services Web)
- Un relâchement des conditions ACID
  - pour permettre une défaillance partielle sans tout recommencer
  - pour permettre des points de reprise intermédiaires
  - pour voir des résultats partiels avant la fin de la transaction
- Des propositions
  - Transactions emboîtées (Moss, 1985)
    - un arbre de transactions, dont seule la racine est ACID (les autres AI)
  - Sagas (Garcia-Molina, Salem, 1987)
    - une suite de transactions ; il faut pouvoir «défaire» une transaction validée

Le Web, l'ultime base de données ?

- Indexer le Web ?
  - Des dizaines de milliards de pages...
  - Deux problèmes
    - la taille de l'index
    - la charge des serveurs
- Deux solutions
  - L'index réparti entre les machines
    - par une technique de hachage
  - Le parallélisme massif («embarassant»)
    - pour accélérer les recherches
    - pour résister à l'accroissement de la charge
- La prochaine étape (en cours)
  - Des données aux connaissances : le Web sémantique