What Goes Around Comes Around

IMS Era

IMS was released around 1968, and initially had a hierarchical data model. It understood the notion of a **record type,** which is a collection of named fields with their associated data types. Each **instance** of a record type is forced to obey the data description indicated in the definition of the record type.

IMS在1968年左右发布，而最初有一个层次数据模型。据了解记录类型的概念，这是与它们相关联的命名字段的集合数据类型。一个记录类型的每个实例都被迫服从数据描述在记录类型的定义表示。

**This requirement of tree-structured data presents a challenge for our sample data：**

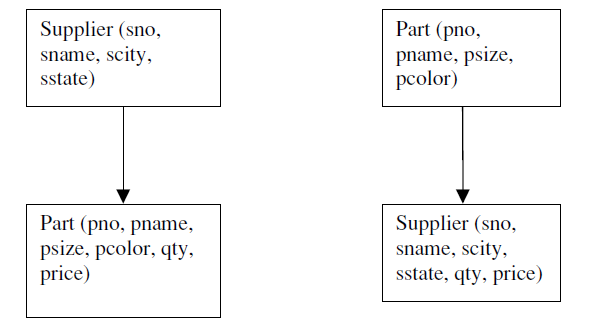
**1. Information is repeated.** In the first schema, Part information is repeated for

each Supplier who supplies the part. In the second schema, Supplier information

is repeated for each part he supplies. 信息被重复。在第一模式中，第一部分信息重复

每个供应商提供谁的一部分。在第二个模式，供应商信息重复他每个部分供应。重复的信息是不可取的，因为它提供了可能性不一致的数据

**2.Existence depends on parents**. In the first schema it is impossible for there to be a part that is not currently supplied by anybody. In the second schema, it is impossible to have a supplier which does not currently supply anything.



IMS supported four different storage formats for hierarchical data. Basically root records can either be:

1. Stored sequentially

2. Indexed in a B-tree using the key of the record

3. Hashed using the key of the record

4. Dependent records are found from the root using either Physical sequentially Various forms of pointers.

Some of the storage organizations impose restrictions on DL/1 commands. the storage organization that hashes root records on a key cannot support “get next”

1.HSAM： 层次顺序访问方法，片段按层次顺序作物理邻接存储。

2.HISAM：层次索引顺序访问方法，非根片段按层次顺序值升序邻接存储，根片段用顺序域索引的方法组织并指向下属区域。

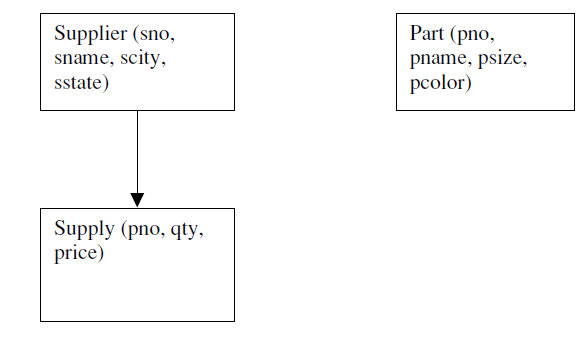
3.HDAM： 层次直接访问方法，片段的存储采用离散分布方式，根片段用顺序域HASH方法组织，从根片段出发用指针按层次顺序值的顺序把物理上分散的从属片段链接起来。

4.HIDAM：层次索引直接访问方法，类似于HDAM，不同的是，根片段采用顺序域索引技术组织，而不是HASH方法。

**physical data independence:** The ability of a data base application to continue to run, regardless of what tuning is performed at the physical level. Physical data independence is important because a DBMS application is not typically written all atonce. As new programs are added to an application, the tuning demands may change, and better DBMS performance could be achieved by changing the storage organization.

IMS supports a certain level of **logical data independence**,because DL/1 is actually defined on a **logical data base**, not on the actual physical data base that is stored. Hence, a DL/1 program can be written initially by defining the logical

IMS支持逻辑数据独立性的一定水平，因为DL /1实际上是在逻辑数据的基础上定义的，而不是在存储的实际物理数据的基础上。因此，在DL/1程序最初可以通过定义逻辑写入



假设一个人构建只包含部分信息的两个物理数据基地，一个和含第二供应商和供应信息如图所示的图中,DL/1方案在树上定义;因此它们不能用直接在图3相反的结构，IMS允许的逻辑的定义如图4这里所示的数据的基础上，从两个不同的供应和零件记录类型数据库被“融合”（加盟）的零件号的共同价值为分层所示的结构。

We will summarize the lessons learned so far, and then turn to the CODASYL proposal.

Lesson 1: Physical and logical data independence are highly desirable

Lesson 2: Tree structured data models are very restrictive

Lesson 3: It is a challenge to provide sophisticated logical reorganizations of tree

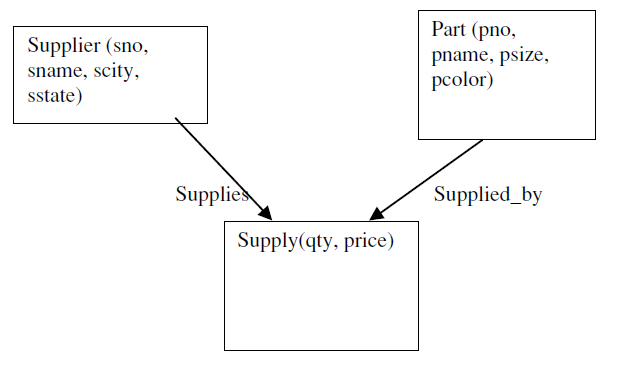
structured data

Lesson 4: A record-at-a-time user interface forces the programmer to do manual query

optimization, and this is often hard.

**CODASYL Era -**Committee on Data Systems Languages

CODASYL was an ad-hoc committee that championed a network data model along with a record-at-a-time data manipulation language.This model organized a collection of record types, each with keys, into a network, rather than a tree. Hence, a given record instance could have multiple parents, rather than a single one,



Supplier-Parts-Supply example

**To find the red parts supplied by Supplier 16 in CODASYL, one can use the following code:**

查询方法：One enters the data base at supplier 16, and then iterates over the members of the Supplies set. This will yield a collection of Supply records. For each one, the owner in the Supplied\_by set is identified, and a check for redness performed

查询语句：

Find Supplier (SNO = 16)

Until no-more {

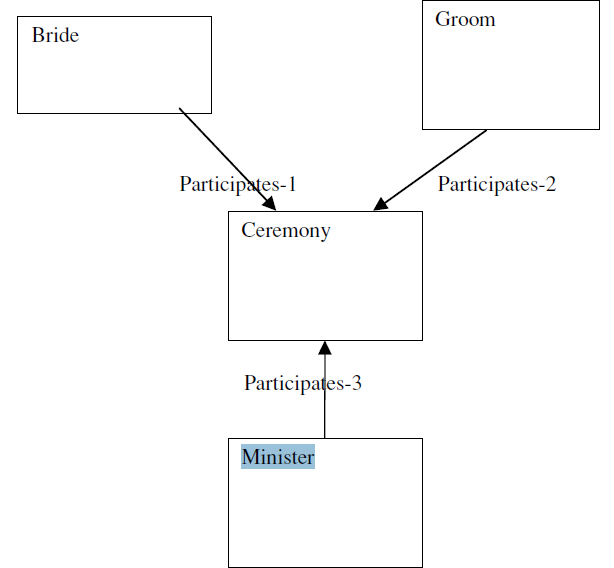
Find next Supply record in Supplies

Find owner Part record in Supplied\_by

Get current record

-check for red—

}



3-way relationship between a bride, a groom, and a minister

**IMS move to network model’s advantage:**

1.no kludges are required to implement graph-structured data 不需要组装件实现图结构的数据

**IMS move to network model’s features:**

2.The CODASYL proposal suggested that the records in each entry point be hashed on the key in the record CODASYL模式建议在每个入口点的记录被散列的记录关键字。

2.The CODASYL proposal provided essentially no physical data independence CODASYL模式基本上没有物理数据独立性，而且也没有逻辑独立性

注：

In contrast, a CODASYL programmer must keep track of the:

The last record touched by the application

The last record of each record type touched

The last record of each set type touched

**Hence, the lessons learned in CODASYL were:**

Lesson 5: Networks are more flexible than hierarchies but more complex

Lesson 6: Loading and recovering networks is more complex than hierarchies

**Relational Era**

proposal was threefold:

1．Store the data in a simple data structure (tables)——one has a better change of providing logical data independence

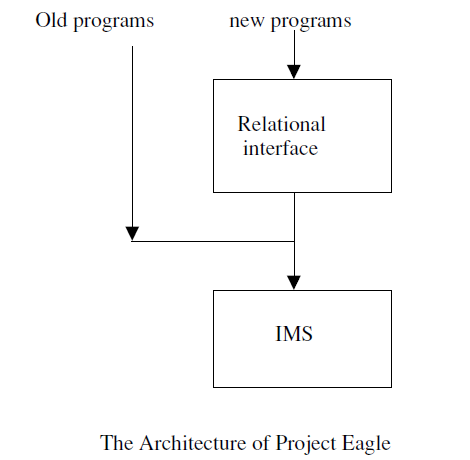
2. Access it through a high level set-at-a-time DML——one can provide a high degree of physical data independence

3. No need for a physical storage proposal——because of DML

IBM’s signal that it was deadly serious about relational systems was a watershed

Moment:

1. it ended once-and-for-all “the great debate”. Since IBM held vast marketplace power at the time, they effectively announced that relational systems had won and CODASYL and hierarchical systems had lost.
2. they effectively declared that SQL was the de facto standard relational language
3. query languages, such as QUEL, were immediately dead.



IBM attempted to execute exactly this strategy, with a project code-named Eagle. Hence, the complexity of logical data bases in IMS came back to haunt IBM many years later. As a result, IBM was forced to move to the dual data base strategy.

**The lessons that were learned from this epoch are:**

Lesson 7: Set-a-time languages are good, regardless of the data model, since they offer much improved physical data independence.带集合的语言在某个阶段是很好的，即使他不是数据模式，但是他的出现给数据库的物理独立性带来很大的改进

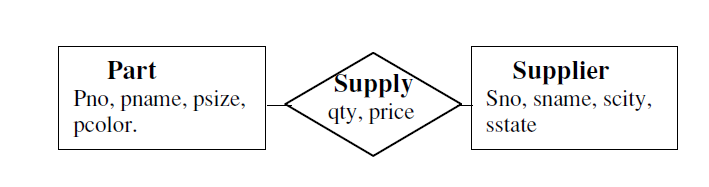
Lesson 8: Logical data independence is easier with a simple data model than with acomplex one.逻辑独立性使用简单的数据模式比复杂的数据模式更容易实现

Lesson 9: Technical debates are usually settled by the elephants of the marketplace, and often for reasons that have little to do with the technology.技术争论通常由市场所决定，而非技术原因。

Lesson 10: Query optimizers can beat all but the best record-at-a-time DBMS application programmers.

**The Entity-Relationship Era**

A popular representation for E-R models was a “boxes and arrows” notation:



E-R图

There were two problems with normalization theory when applied to real world data base design problems.

1. 我怎样得到一组初始的表
2. 标准化理论是基于功能的概念依赖,现实世界的dba无法理解这种构造

**R++ Era** Extended Relational

产生：考虑到Entity—Relationship Database 出现的查询困难，而添加新特性关系模型来解决这个问题，主要包括了：

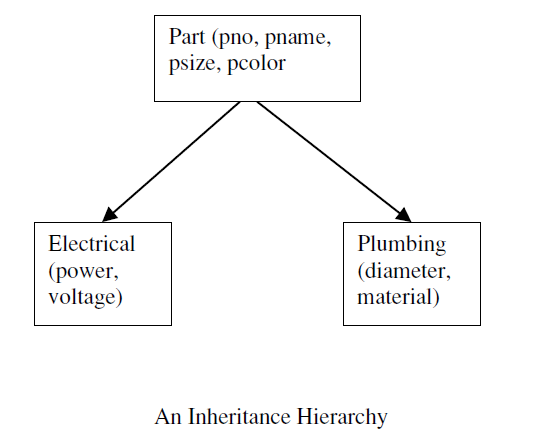
1. mechanical CAD、VLSI CAD
2. text management
3. computer graphics

With these constructs however, we can find the suppliers who supply red parts as:

Select Supply.SR.sno From Supply Where Supply.PT.pcolor = “red”

**query language extensions:**

1. **set-valued attributes**
2. **aggregation (tuple-reference as a data type).**
3. **generalization.**



one could reference an inheritance hierarchy in the query language.For example：

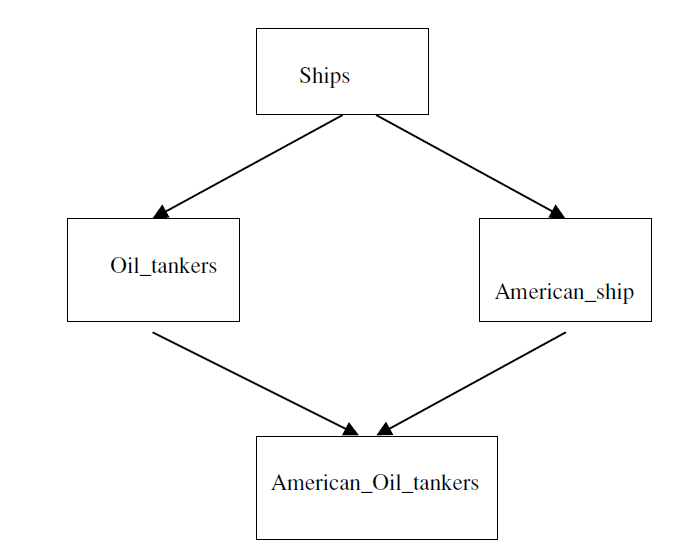
**to find the names of Red electrical parts, one would use:**

Select E.pname From Electrical E Where E.pcolor = “red”

Lesson 12: Unless there is a big performance or functionality advantage, new constructs will go nowhere

**The Semantic Data Model Era**

They suggested that the relational data model is “semantically impoverished” Post relational data models were typically called semantic data models.



此外,类可以概括其他类, American\_oil\_tankers继承属性Oil\_tankers和在哪里American\_ships。这种构造通常被称为多重继承。类也可以工会、交叉或其他类的区别。也可以一个子类另一个类,指定的谓词来确定会员

SDMs had the same two problems that faced the R++ advocates:

1. they were a lot of machinery that was easy to simulate on relational

Systems 大量机械装置相似于关系系统

1. namely that the established vendors were distracted with transaction processing performance. 建立供应商与事务处理分散性能。

**OO Era**

In practice, relational data bases had their own naming systems, their own data type systems, and their own conventions for returning data as a result of a query. Hence, to bind an application to the data base required conversion from “programming language speak” to “data base speak” and back

OO ERA产生背景：

所有的编程语言都没有为数据库内置函数，这不仅使得嵌入数据库语言枯燥，而且编程难度加大、易错。

为了解决工程市场，持续化编程C++有以下几个要求：

1. 不需要声明查询语言
2. 不需要进行事务管理
3. 当要操作对象的时候，运行系统与传统的C++相互竞争。

there are a number of reasons for this market failure：

1. 缺乏影响力
2. 没有标准化
3. 重新设置所有编程比如C++编程方法
4. 没有通用世界的编程语言

**O2**

O2, that had a different business plan. O2 supported an objectoriented

data model, they embedded a high level declarative language called OQL into a programming language. Hence, they proposed what amounted to a semantic data model with a declarative query language, but marketed it as an OODB. Also, they focused on business data processing, not on the engineering application space.

但是O2还是失败了，他关注的是事务的处理而不是工程应用。对于O2可以用一句话概括：as goes the United States goes the rest of the world.

**OODB吸取的教训:**

Lesson 13: Packages will not sell to users unless they are in “major pain”

Lesson 14: Persistent languages will go nowhere without the support of the programming language community.

**The Object-Relational Era**

The Object-Relational (OR) era was motivated by a very simple problem. In the early days of INGRES, the team had been interested in geographic information systems (GIS) and had suggested mechanisms for their support [GO75]. Around 1982, the following simple GIS issue was haunting the INGRES research team. Suppose one wants to store geographic positions in a data base.

存储一个位置position只需要存经度和纬度(long, lat)，如果想查询（X0，Y0，X1，Y1）的交叉域，则需要查询语句为：

Select I-id From Intersections Where X0 < long < X1 and Y0 < lat < Y1

more new data types appears to be needed every decade.As a result, the OR proposal added:

user-defined data types,

user-defined operators,

user-defined functions, and

user-defined access methods

**Postgres:**

Postgres implemented a sophisticated mechanism for UDTs, UDFs and user-defined access methods. In addition, Postgres also implemented less sophisticated notions of inheritance, and type constructors for pointers (references), sets, and arrays. This latter set of features allowed Postgres to become “object-oriented” at the height of the OO craze.

**成就：**

The OR model has enjoyed some commercial success. Postgres was commercialized by Illustra. After struggling to find a market for the first couple of years, Illustra caught “the internet wave” and became “the data base for cyberspace”.

**Illustra was acquired by Informix：**

1. inside every OR application, there is a transaction processing sub-application.
2. To be successful, Illustra had to convince third party vendors to convert pieces of their application suites into UDTs and UDFs.

**解决方法：**

Informix provided a solution to both problems, and the combined company proceeded over time to sell OR technology fairly successfully into the GIS market and into the market for large content repositories

**Lesson：**

Lesson 14: The major benefits of OR is two-fold: putting code in the data base (and thereby bluring the distinction between code and data) and user-defined access methods.通过用户定义访问方法，实现数据和代码分离

Lesson 15: Widespread adoption of new technology requires either standards and/or an elephant pushing hard.一项新技术的使用需要定义规范的标准和大量推销努力

**Semi Structured Data**

最近,各种各样的基于xml的提案与半结构化数据有相同的味道。目前,XMLSchema和XQuery的标准是基于xml的数据。

There are two basic points that this class of work exemplifies.

1) schema last

2) complex network-oriented data model

**Schema Last：**

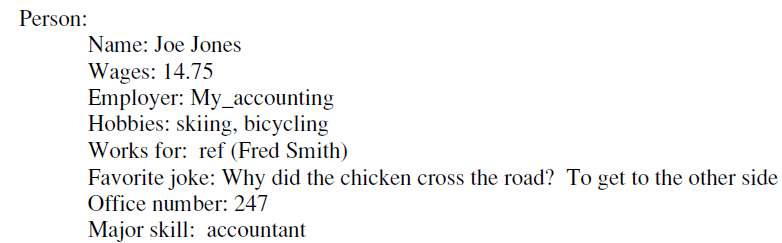
**Schema First 说明：**

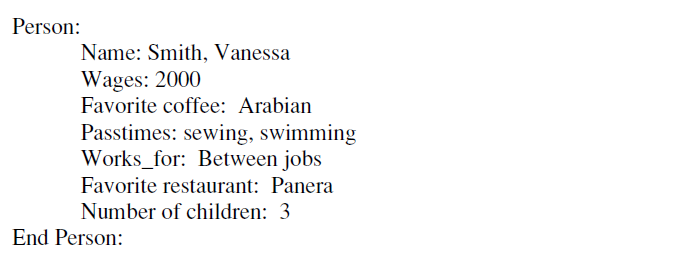
在一个“模式第一”的系统是指定的，实例的数据记录遵照这个模式并且随后加载。因此,数据库总是与预先存在的一致模式,因为DBMS拒绝任何记录不一致的模式

**schema last 说明：**

必须自己描述数据实例, 因为没有指定模式给传入的意义记录。没有自己描述格式,记录只是“一桶位”。为实现记录自描述,必须标记每个属性定义的元数据属性的意义。

Here are a couple of examples of such records：





特性：1.两个记录没有一个记录在其他记录中是相同的

2.有个属性值在两个记录中意思类似

3.不但是记录而且意思也不同