

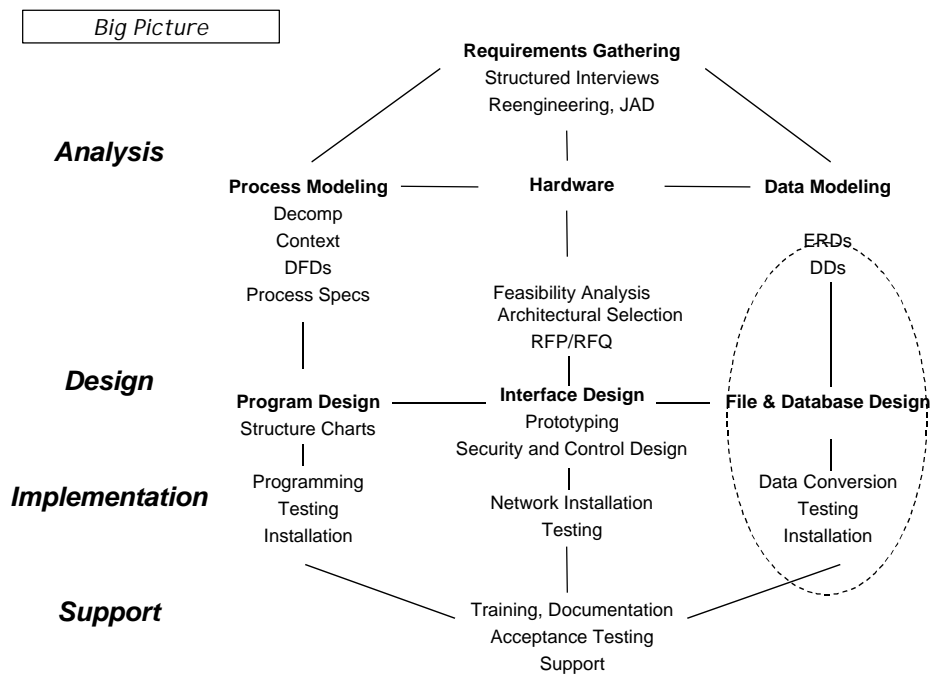
Data Modeling

Conceptual Model

Logical Model

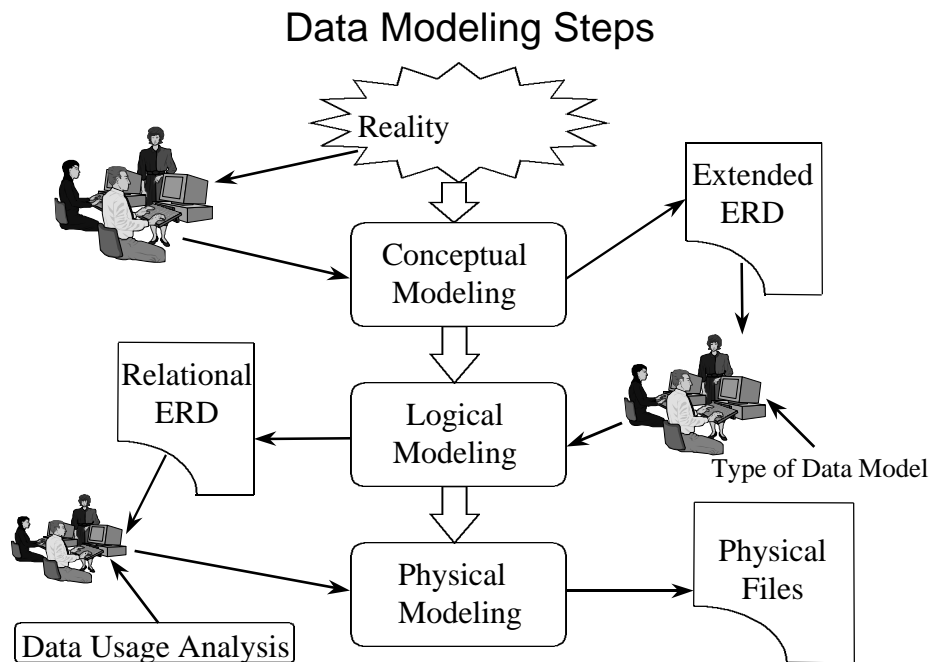
(Relational Model)

Physical Model

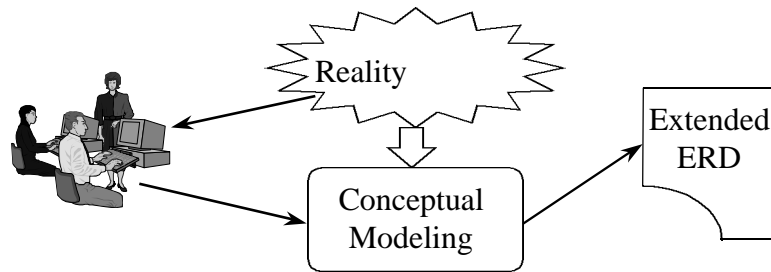


Goals of Data Design

- understand underlying structure of system
- provide lasting foundation for maintenance
- balance performance with flexibility
- may also save storage space



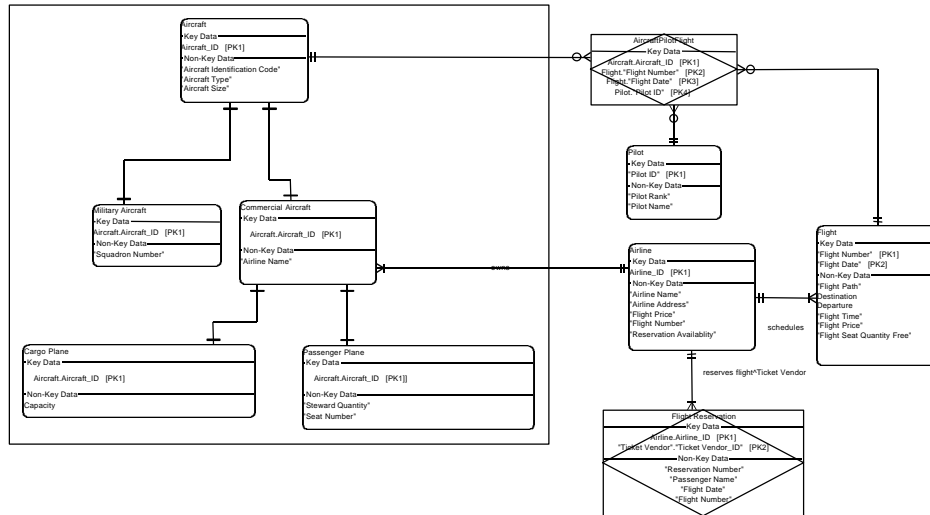
Conceptual Data Model



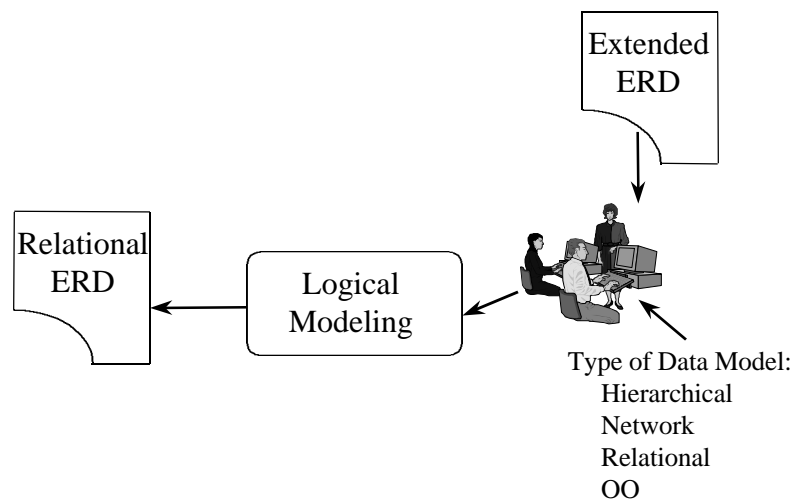
Conceptual Data Modeling

- Extended ERD
- Concepts used:
 - Entity
 - Relation
 - Association
 - Aggregation
 - Class
 - Inheritance
 - IS-A relation
 - and so on
- Semantic Level
 - it can be implemented in any data model

Conceptual Data Modeling



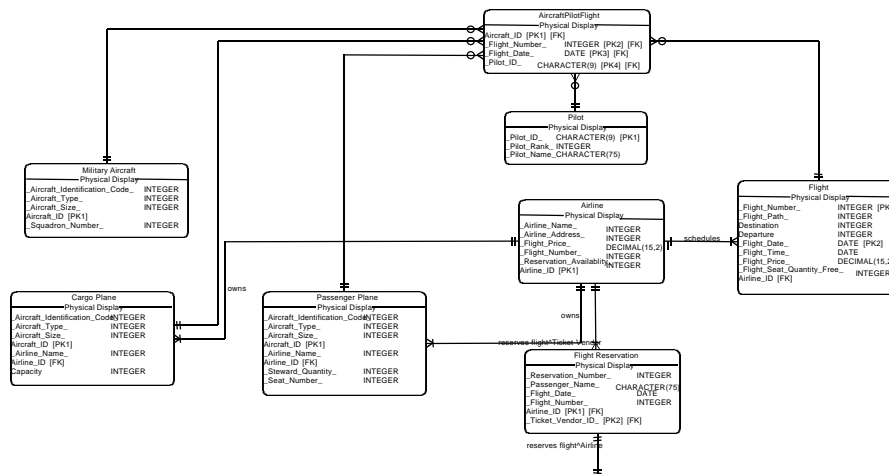
Logical Data Modeling



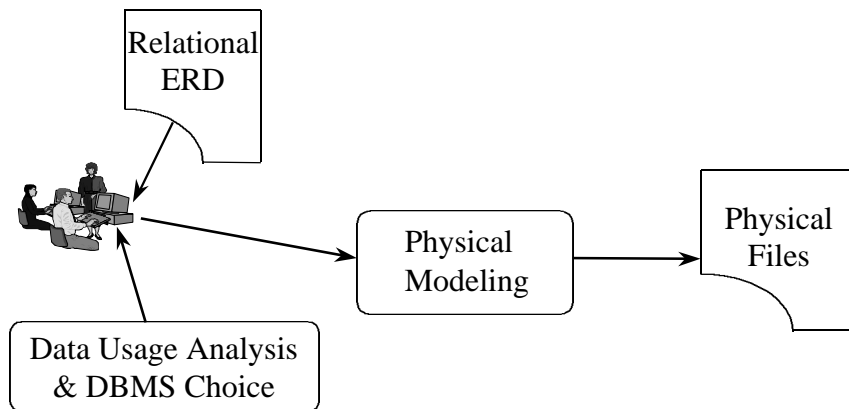
Logical Data Modeling

- Choose a Data Model
 - hierarchical, network, relational, oo
- Fitting conceptual model into a chosen data model
- In case the relational is chosen,
 - convert Extended ERD into Relational ERD
 - collapse different types of relationship into association
 - because only one type of relationship is possible in relational model
 - association by common columns (PK, FK relation)
 - add foreign keys

Logical Data Modeling

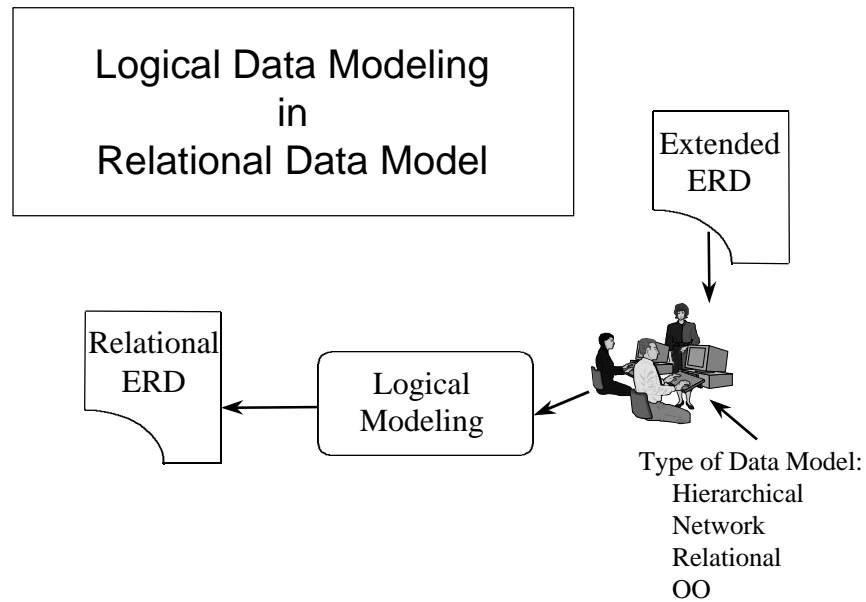


Physical Data Modeling



Physical Data Modeling

- Conditions for this step
 - Choice of DBMS
 - Oracle, Access
 - Usage Analysis
- What?
 - Create actual tables
 - Possible denormalization



Relation -- Two Dimensional Table

- Holds data that pertains to some thing
- Each column contains data regarding an attribute
- Each row (tuple) contains data values for an entity

Rules of the Relational Data Model

- Everything is stored in tables (relations)
 - Data and Meta-data
 - No hidden pointers/links

Product No.	Description	Cost	Retail
M128	Bookcase	150.50	200.00
B381	Cabinet	120.00	150.00
R120	Table	410.80	500.00
⋮	⋮	⋮	⋮

Relational Model--Definitions

- Relations/Tables
- Tuples/Rows
- Attributes/Columns
- Cells
- Domain

<u>Product No.</u>	Description	Cost	Retail
M128	Bookcase	150.50	200.00
B381	Cabinet	120.00	150.00
R120	Table	410.80	500.00
⋮	⋮	⋮	⋮

Relational Model	Programmer	User
Relation	File	Table
Tuple (row)	Record	Row
Attribute	Field	Column

Rules for Tables (Relations)

- Only 1 Value per Cell
- Only 1 Domain per Column
- Each Row is Unique
- No Order to Rows or Columns

Product No.	Description	Cost	Retail
M128	Bookcase	150.50	200.00
B381	Cabinet	120.00	150.00
R120	Table	410.80	500.00
⋮	⋮	⋮	⋮

Components of Relational Data Model

- Represents data in form of tables or relations-based on mathematical set theory
- 3 components
 - data structure (table)
 - data manipulation (query language, SQL)
 - integrity rules: business rules

Well-Structured Database

- Minimum amount of redundancy
- Allows users to insert, modify, and delete rows without errors
- In other words, optimized structure
- How to optimize a relational database?
 - Bottom up through normalization
 - Top down through ER conceptualization

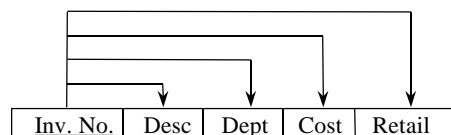
Update Anomalies

- Update: having to change one fact in more than one place
- Inconsistent Data: different values in different places
- Additions
- Deletions

ORDNUMB	ORDDTE	PARTNUMB	PARTDESC	NUMBORD	QPRICE
12489	90293	AX12	IRON	11	14.95
12491	90293	BT04	STOVE	1	402.99
12491	90293	BZ66	WASHER	1	311.95
12494	90493	CB03	BIKE	4	175
12495	90493	CX11	MIXER	2	57.95
12498	90593	AZ52	SKATES	4	22.95
12498	90593	BA74	BASEBALL	1	4.95
12500	90593	BT04	STOVE	1	402
12504	90593	CZ81	WEIGHTS	2	108.99

Normalization

- Eliminates Update Anomalies
- Based upon Functional Dependencies
- Decomposition creates Normal Relations



Classes of Normal Form

- E.F. Codd (1970)--defined first, second, and third normal forms (1NF, 2NF, 3NF).
- Later, Boyce-Codd normal form (BCNF) was specified, followed by fourth and fifth. These forms are nested. To be in 3NF, relation must be in 1NF & 2NF.
- Fagin (1981) defined domain/key normal form (DK/NF)

Functional dependency

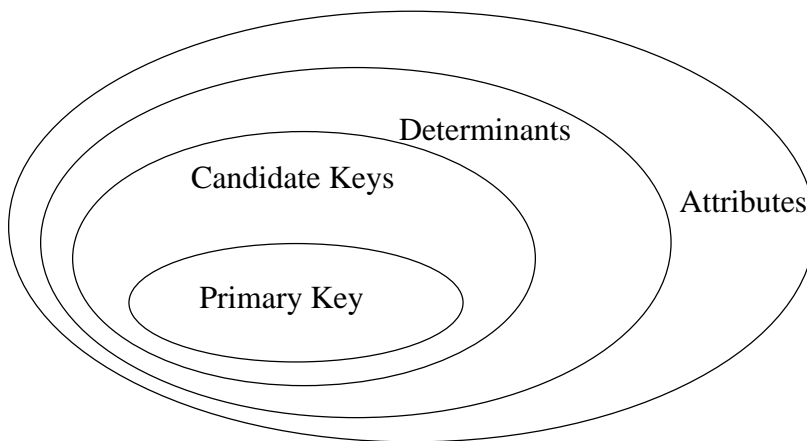
- Dependence between attributes
- $A \twoheadrightarrow B$
 - A is the determinant of B
 - A functionally determines B
 - B is functionally dependent upon A
- $SSN \twoheadrightarrow Name$
- $SSN, COMP\# \twoheadrightarrow Grade$

Definition: An attribute, B, is functionally dependent on another attribute(s), A, if a value of A determines a single value for B at any one time.

Keys

- Candidate key: attribute(s) that could be used to determine all other attributes in a table
 - All attributes in a table are functionally dependent on candidate key (uniqueness)
 - No subcollection of the attributes also has property above (minimality)
 - Textbook definition is misleading
- Primary key: the candidate key chosen to determine all other attributes in a table
- Alternate Key: a candidate key not chosen as the primary key
- Non-key attributes: all other attributes in a tables

Keys



First normal form

- A relation (table) is in first normal form (1NF) if it does not contain repeating groups

PK is ORDNUMB

ORDNUMB	ORDDTE	PARTNUMB	NUMBORD
12489	90293	AX12	11
12491	90293	BT04,	1
		BZ66	1
12494	90493	CB03	4
12495	90493	CX11	2
12498	90593	AZ52,	4
		BA74	1
12500	90593	BT04,	1
		CZ81	2

1NF (cont.)

- Solution: remove repeating groups
 - Expand the primary key: usually into composite primary key
 - original primary key concatenated with key to the repeating group

PK:
ORDNUMB +
PARTNUMB

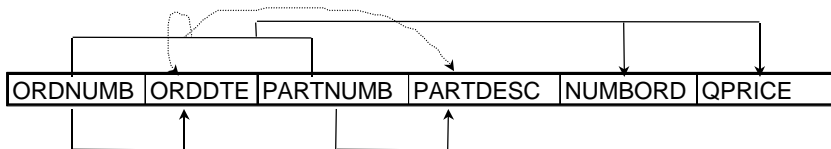
ORDNUMB	ORDDTE	PARTNUMB	NUMBORD
12489	90293	AX12	11
12491	90293	BT04	1
12491	90293	BZ66	1
12494	90493	CB03	4
12495	90493	CX11	2
12498	90593	AZ52	4
12498	90593	BA74	1
12500	90593	BT04	1
12500	90593	CZ81	2

Second normal form

- Problem: Partial functional dependency which causes update anomalies
 - insert, delete, update
- A relation is in 2NF if:
 - it is in first normal form and
 - no nonkey attribute is dependent on only a portion of primary key
- Solution: remove partial dependency
 - Split the composite PK into separate tables
 - Assign the attributes accordingly
 - Name new tables
 - Result: all non-key attributes are facts about the primary key, the whole primary key

Non 2NF

ORDNUMB	ORDDTE	PARTNUMB	PARTDESC	NUMBORD	QPRICE
12489	90293	AX12	IRON	11	14.95
12491	90293	BT04	STOVE	1	402.99
12491	90293	BZ66	WASHER	1	311.95
12494	90493	CB03	BIKE	4	175
12495	90493	CX11	MIXER	2	57.95
12498	90593	AZ52	SKATES	4	22.95
12498	90593	BA74	BASEBALL	1	4.95
12500	90593	BT04	STOVE	1	402
12504	90593	CZ81	WEIGHTS	2	108.99



2NF Solution

1. (ORDNUMB,PARTNUMB,ORDDATE,PARTDESC,NUMBORD,QPRICE)
2. (ORDNUMB,
(PARTNUMB,
(ORDNUMB,PARTNUMB,
3. (ORDNUMB, ORDDTE)
(PARTNUMB, PARTDESC)
(ORDNUMB,PARTNUMB, NUMBORD, QPRICE)

2NF

ORDNUMB	ORDDTE
12489	90293
12491	90293
12494	90493
12495	90493
12498	90593
12500	90593
12504	90593

ORDERS

PARTS

PARTNUMB	PARTDESC
AX12	IRON
BT04	STOVE
BZ66	WASHER
CB03	BIKE
CX11	MIXER
AZ52	SKATES
BA74	BASEBALL
CZ81	WEIGHTS

ORDLINE

ORDNUMB	PARTNUMB	NUMBORD	QPRICE
12489	AX12	11	14.95
12491	BT04	1	402.99
12491	BZ66	1	311.95
12494	CB03	4	175
12495	CX11	2	57.95
12498	AZ52	4	22.95
12498	BA74	1	4.95
12500	BT04	1	402
12504	CZ81	2	108.99

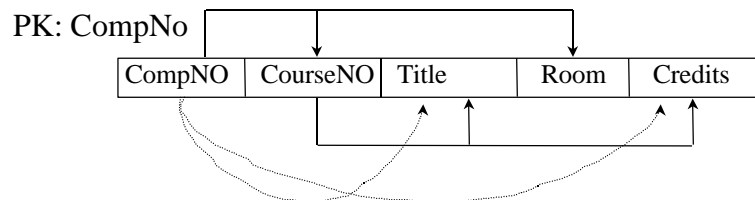
Third normal form

- Problem: transitive functional dependency of nonkey attributes on the primary key:
potential update anomalies of insert, delete, update
- A relation is in 3NF
 - if it is in second normal form and
no nonkey attribute is transitively dependent on the primary key

in other words: no repeating groups, and all non-key attributes are facts about the primary key, the whole primary key, and nothing but the primary key.

Non 3NF

CompNo	CourseNo	Title	Room	Credits
0891	BA 201	Intro to Info Systems	CS 200	5
0892	BA 201	Intro to Info Systems	CS105	5
0993	AC 201	Intro to Accounting	CS107	4
0994	AC 201	Intro to Accounting	CS 200	4
0997	CIS 326	Intro to C	CS 108	5
0999	CIS 330	Systems Analysis	CS 106	5



3NF

Solution: Split and make a new table for each determinant which is not a candidate key

CompNo	CourseNo	Room
0891	BA 201	CS 200
0892	BA 201	CS 105
0993	AC 201	CS 107
0994	AC 201	CS 200
0997	CIS 326	CS 108
999	CIS 330	CS 106

CourseNo	Title	Credits
BA 201	Intro to Info Systems	5
AC 201	Intro to Accounting	4
CIS 326	Intro to C	5
CIS 330	Systems Analysis	5

Grouping of Attributes

- Relation-- GRADES(SID,ClassName,Grade)
- Combination of SID and ClassName determines Grade.
- Note: SID or ClassName alone does not determine Grade but both.
- Composite Primary Key

3NF Incorrect Decomposition

From

COURSE (COMPNO, COURSENO, TITLE, ROOM, CREDITS)

To

SECTION (COMPNO, COURSENO, ROOM, CREDITS)

COURSE (COMPNO, TITLE, CREDITS)

or

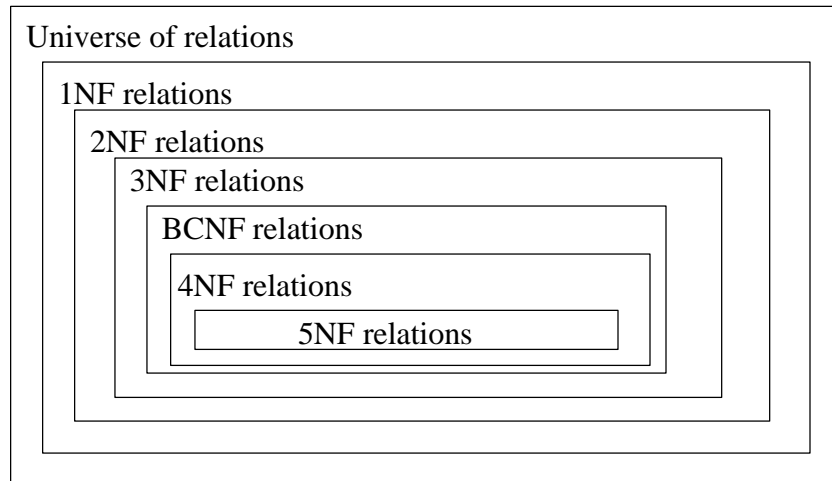
SECTION (COMPNO, TITLE, ROOM, CREDITS)

COURSE (COURSENO, TITLE, CREDITS)

Higher Level Normal Form

- Boyce-Codd Normal Form
 - Eliminate multivalued dependency
- Fourth Normal Form
 - Every determinate is a candidate key
 - When a relation has more than one candidate key, anomalies may still result.
- 5NF & DK/NF
 - Beyond the scope of this class

Normalization Universe



Normalization

- Problems can occur when a relation contains facts about two different themes.
 - Add row, add data about two themes
 - Delete row, delete data about two themes
- Every normalized relation has a single theme. Every time a relation is broken-up, may create a need for an inter-relation constraint

Normalization Process

- 1NF -- eliminate repeating groups
- 2NF -- eliminate partial dependencies
- 3NF -- eliminate transitive dependencies
- Simplify through inspection
 - eliminate redundant elements within structures
 - eliminate redundant elements across structures
 - make sure attribute names are clear

Relational Database Pledge of Allegiance

*I pledge allegiance
to third normal form and
to the lack of update anomalies
for which it stands:
with full functional dependency
on the key,
the whole key, and
nothing but the key,
so help me, Codd,
with accessibility and ease of use for all.*

Alternative to Normalization:
Top Down Optimization:
High Quality Data Modeling:
Possibly ERD

- Normalization by intuition
 - Each entity corresponds to a person, object, or event in the world, i.e., it is namable and identifiable.
 - All attributes of each entity actually describe that entity.
- High quality data models rarely change through the normalization process

About Establishing Relationships in Relational Data Model

or

From Conceptual Data Model to Relationally Logical

*In a nut shell,
through common columns
across relations*

Compared to Normalization

- Normalization is based on functional dependency analysis
- Functional Dependency is about the relationship among attributes within a relation
- Primary key and foreign key relation is based on cardinality
- Cardinality is about the relationship among relations

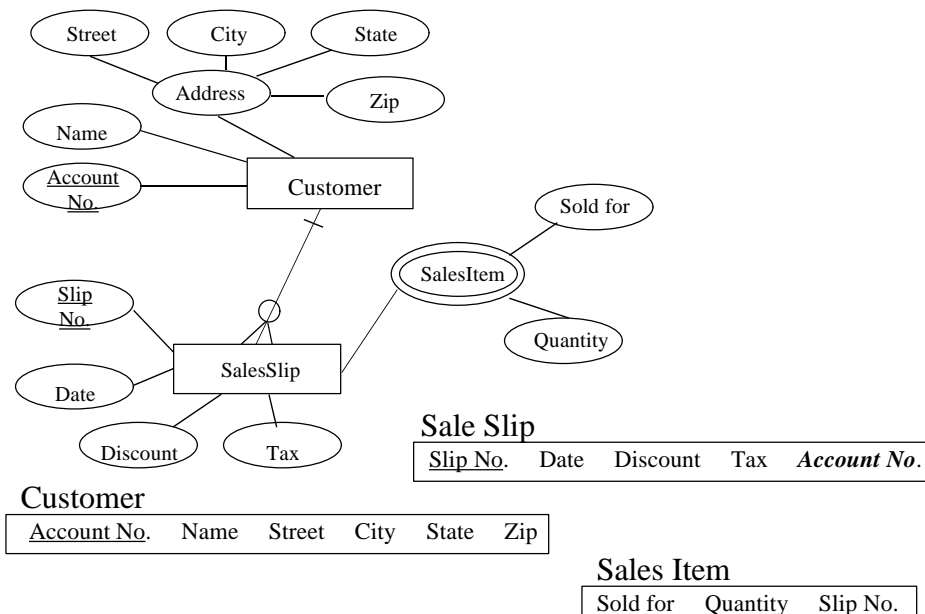
Binary 1:N Relationship

- One-to-Many (1:N) is represented by adding PK attributes of the entity on the “one-side” as a foreign key in the relation on the “many-side”

(1) CUSTOMER(CustNo, Name, Address, City, State, Zip, Discount)

(M) ORDER(OrderNo, OrderDate, PromisedDate, CustNo)

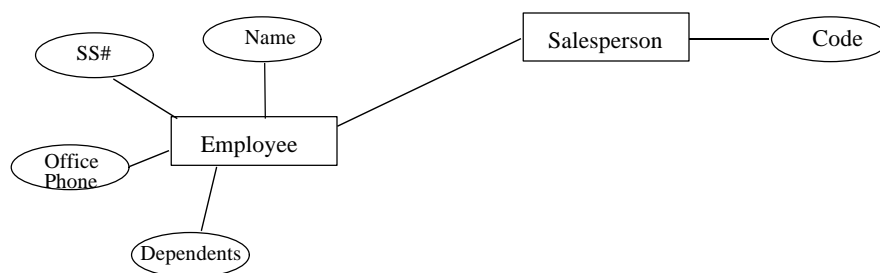
Converting 1:N Relationships



Binary 1 to 1 Relationships

- Special case of 1:N
 - Add PK of A to B as Foreign key
 - or
 - Add PK of B to A as Foreign Key
- Sometimes merge into one relation

Converting 1:1 Relationships



Employees

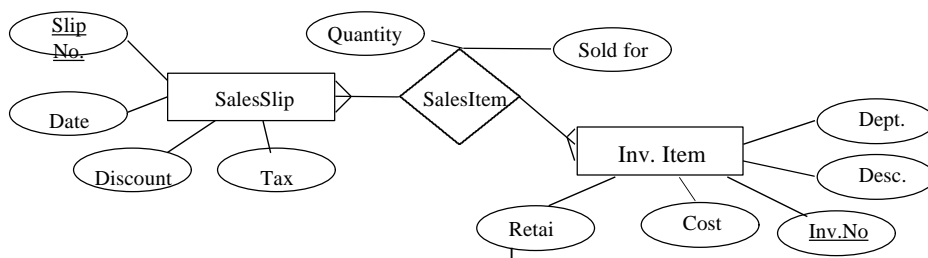
SS#	Name	Office Phone	Dependents	Code
-----	------	--------------	------------	------

Binary M:N Relationship

- Many-to-Many --between A & B
 - Create a separate relation “C”
 - PK is a composite key of PKs of A & B
 - Include any nonkey attributes associated with relationship

ORDER(OrderNo, OrderDate, PromisedDate)
PRODUCT(ProductNo, Desc, Room, Other)
ORDERLINE(OrderNo, ProductNo, QuantityOrdered)

Converting N:M Relationships



Sale Slip

<u>Slip No.</u>	Date	Discount	Tax	Account No.
-----------------	------	----------	-----	-------------

Inv. Item

<u>Inv. No.</u>	Desc	Dept	Cost	Retail
-----------------	------	------	------	--------

Sales Item

Sold for	Quantity	<u>Slip No.</u>	<u>Inv. No.</u>
----------	----------	-----------------	-----------------

Associative Entity in M:N

- Sometimes, may need another attribute as part of key



SHIPMENT(CustNo, VendorNo, Date, Amount)

Unary (Recursive) Relationship

- Between single entity types
- 1:N (add attribute)
- M:N (create another relation)
- ISA (Class/Subclass)
 - Create relation for class and each subclass
 - Subclass--PK and attributes unique to subclass

Merging Relations (View Integration)

- Check results of views--may be some redundancy
- Relations with same PK, describe same entity, then merge into one relation
 - Synonyms--two or more attributes with different names--use one name
 - Homonyms--attribute with different meanings--create new names for each

Integrate Data and Process Model with Event Analysis

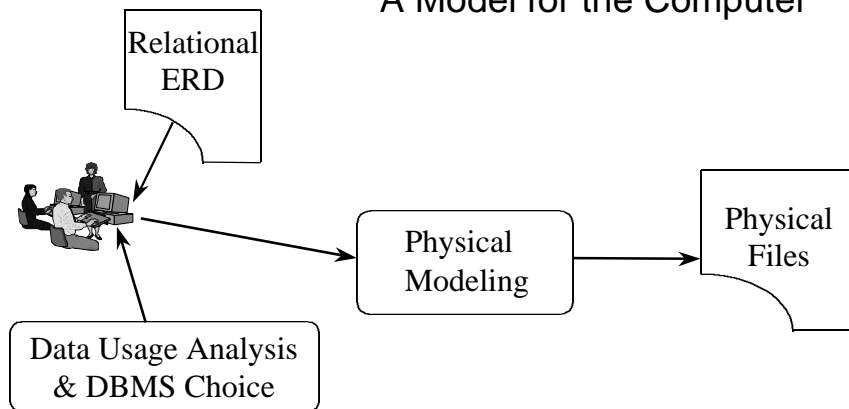
- DFD accesses are to new data structures
 - one ERD entity per DFD store
- Validate each access in model
 - make sure necessary identifiers are available to accessing process
 - make sure required data exists in model

Event Analysis

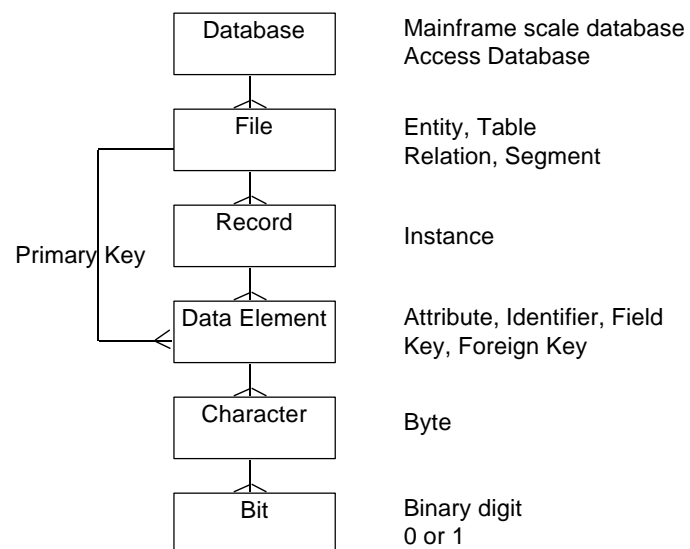
- identify event
 - entity Create, Review, Update, Delete (CRUD)
- identify conditions under which event occurs
 - identify effect on entity(s)
- for each named event/condition
 - revise DFDs (if necessary) to reflect your findings

Physical Database Design

A Model for the Computer



File and Database Terms



Application Support Files

General rule is one file per entity

- master files
 - carry application's primary data
 - usually independent entities of great interest
- transaction files
 - used to cross system boundaries
 - usually dependent on primary entities

Application Support Files

- archival files
 - assess control and legal requirements
 - implemented for business reasons
- backup/log files
 - assess risk of loss and cost of backup
 - implemented for technical reasons

Application Support Files

- In a relational database environment, all these files become tables in one or more databases

File Design Considerations

- Dealing with conflicting requirements
 - favor most common accesses
 - favor most important accesses
- Dealing with a database environment
 - large scale DBMS offers control over access & organization
 - small scale DBMS usually offer little flexibility

Mapping an ERD to a Relational Database

First-cut file design

- entity ---> file or DB table
- attribute ---> field
- identifier ---> primary key
- 1-M relationship ---> foreign key (on M side)
- 1-1 relationship ---> 1 or 2 foreign key(s)
- M-M relationship ---> file or DB table

when to "denormalize" data: performance

Designing for Performance

- Identify performance requirement
- Estimate performance of design decisions
- Minimize cost while meeting requirement

Dimensions of Performance

- need to focus on people and machines
- throughput (e.g. # transactions per minute)
- availability (e.g. % up-time)
- response time (e.g. # seconds)

Improving Response Time

- Response time is composed of
- systems overhead (GUI, DBMS, OS)
- processing time (CPU)
- transmission time (network)
- I/O time (file/database access)

Improving Throughput

- Increase number or speed of processors
- Increase bandwidth
- Redesign organization procedures (where appropriate)

File or Database I/O Time

- $T = V/S$
- decrease volume
 - reduce number physical record accesses (blocking)
 - reduce number file accesses (absorption)
 - reduce physical record size (segmentation)
- increase speed/capacity
 - faster devices
 - more devices
- for batch:
 - streamline manual procedures

Modifying File or Database Design to Improve Performance

- pure absorption
 - from 1 into M
 - from M into 1
- pure segmentation
 - by instance
 - by attribute
- partial strategies

Data Usage Analysis

- What is the frequency of access to each relation and its attributes?
- What is the frequency of updates to each relation and its attributes?
- What is the retrieval time restrictions to each relation and its attributes?
- How are relations combined to form objects?

Physical Definition

- Define Tables and Attributes
 - Specify Field Types and Sizes
 - Specify Null or Not Null
 - Specify Primary Key
- Define Relationships
 - Referential Integrity

SQL Statements

```
CREATE TABLE PRODUCT
  (PRODUCT_NO      INTEGER    NOT NULL
  DESCRIPTION CHAR(15)
  FINISH           CHAR(10)
  ROOM             CHAR(2)
  UNIT_PRICE       DECIMAL(10,2))
```

Physical Modeling: SQL

```
CREATE TABLE CUSTOMER
  (Cust_No      CHAR (4)  NOT NULL
   Name        CHAR (15)
   Address     CHAR (15)
   CSZ         CHAR (15))
CREATE TABLE PRODUCT
  (Prod_No     INTEGER    NOT NULL
   Descrip     CHAR(15)
   Finish      CHAR(10)
   Room        CHAR(2)
   Price       DECIMAL(10,2))
CREATE TABLE ORDER
  (Ord_No      INTEGER    NOT NULL
   Date        DATE
   P_Date      DATE
   Cust_No     CHAR (15)
   REFERENCES CUSTOMER)
CREATE TABLE REQUESTS
  (Ord_No      INTEGER    NOT NULL
   REFERENCES ORDER
   Prod_No     CHAR (4)  NOT NULL
   REFERENCES PRODUCT
   Qty         INTEGER
```

```
CREATE UNIQUE INDEX Cust
  ON CUSTOMER (Cust_No)
CREATE UNIQUE INDEX Prod
  ON PRODUCT (Prod_No)
CREATE UNIQUE INDEX Order
  ON ORDER (Ord_No)
CREATE UNIQUE INDEX Requests
  ON REQUESTS (Cust_No, Prod_No)
CREATE CLUSTER ORDER_DATA
  (Ord_No     INTEGER    NOT NULL)
ALTER CLUSTER ORDER_DATA
  ADD TABLE ORDER
  WHERE ORDER_DATA.Ord_No=
    ORDER.Ord_No
ALTER CLUSTER ORDER_DATA
  ADD TABLE REQUESTS
  WHERE ORDER_DATA.Ord_No=
    REQUESTS.Ord_No
```