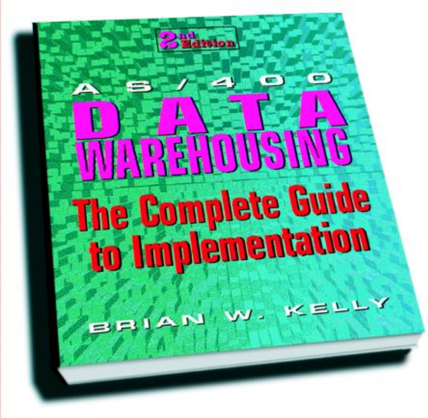
Dear Reader: Thank you for downloading this free book from Brian W. Kelly's finished book catalog. This is a course based on my data warehousing AS/400 book. It teaches about Data Warehousing.

Most of my books had previously been published on Amazon.

Click below if you would like to donate to help the free book cause: https://www.letsgopublish.com/books/donate.pdf

Enjoy!

Kelly Consulting



AS/400™

QuikCourse™ Series

Course Code AS050

Abbreviated Version For Spring, 2000 COMMON

Session # 21MJ, ID 404565

Speakers:

Brian Kelly & Dennis Grimes
Kelly Consulting, 11 Marjorie Avenue, Wilkes-Barre PA 18702
Phone 570-829-5926, Fax 570–655-8501 http://www.kellyconsulting.com
bkelly@kellyconsulting.com, dgrimes@socantel.net
Tuesday, 8:00 A.M., San Francisco Room, Marriott

AS/400 Data Warehousing:

Do-It Yourself Guide To Implementation

Reference: AS/400 DataWarehousing, Brian Kelly, Midrange Computing, February, 1997

King's College January 2011

This presentation is the basis of the Moodle assignment asking for you to write one paragraph on Data Warehousing. I wrote two books on Dta Warehousing in the 90's and early 2000. The basics have not changed much.

If you have the opportunity read this from cover to cover as it will enrich you knowledge of Data Warehousing as it relates to relational database. This presentation discusses how to use a relational database for data warehousing and as such it offers insights for all DB implementers.

The essence of the look and feel of the DB constructs from DW to DB has to do with normalization v. denormalization.

General Information

Kelly Consulting,
Diversified Information Technologies,
Brian Kelly, Dennis Grimes, and this presentation

Brian Kelly is also the founder and President of Kelly Consulting, a Soup To Nuts AS/400 Education Firm which also offers general AS/400 Management Consulting, specialized consulting and education in AS/400 Internet and DataWarehousing implementation. Kelly is a former IBM Systems Engineer, and the author of nine books in the Information Technology area including AS/400 Data Warehousing; The Complete Guide to Implementatoin, Midrnage Computing, 1997. Kelly is also the Author of The AS/400, The Internet, and Email, which is also published by Midrange Computing.
Brian Kelly and Kelly Consulting can be reached at 570-829-5926
Send E-Mail to bkelly@kellyconsulting.com

Dennis Grimes is the Senior Instruction Manager at Kelly Consulting and the Vice president of Information Technology at Diversified Information Technologies, a record storage and electronic imaging firm with headquarters in Scranton, PA. Grimes is the founder and former General Partner of Glenbrook Software Limited, the builders of the AS/400 based aXcess/400 data access product. Dennis Grimes can be reached at 717-655-8590, Send E-Mail to dgrimes@socantel.net

Disclaimer:
Though care was taken throughout this presentation that the information contained herein is accurate, there is no expressed or implied warranty that all information in this presentation is, will be, or ever was 100% correct. Kelly Consulting accepts no liability for any use of the information in this document.

Agenda

Examination of the Issue

Tools used by End Users

How Production Databases Are Designed

The Normalization Process

Designing Databases for End-User Computing

Re-engineering Databases for End User

Computing---Views/ Extractions

Denormalization

End-User Reporting

Sample Databases

Summary / Conclusions

Appendices:

A. Data Warehousing / Tools (sessions @COMMON)

B. AS/400 DW Client Series - New Players

C. Solutions to Class Exercises

Some Starter Quotes

Datamation June 1, 1994
"Why Decision Support FAILS and How to FIX it"
by Ralph Kimball and Kevin Strehlo

"Forget Everything you know about entity relationship data modeling. If your DBA is proud of normalizing all your databases into Fifth Normal Form, tell her to read this article. Or if all you know is that it's difficult to get answers to simple business questions from your decision support system, or it takes hours to get an answer..."

*"The **relationally correct** data modeling everyone is taught*

in school is only useful for achieving high performance in on-line transaction processing. The resulting model fragments the data into many tables... But using that model with a real-world decision support system almost guarantees failure"

Brian Kelly writing in **AS/400 Data Warehousing...**, 2nd Edition, Midrange Computing, 1997:

“Kimball and Strehlo capture the problem in a nutshell. All our database lives, we have been taught to design data base structures to third normal form. Even more recently, we have been encouraged to go even further to fifth normal form. And yes, this does make for a far more flexible, longer lasting production database design than when we tried to cram everything into one record. Survey questions like “how many line items to an order?” had more far reaching consequences in our former DB design life than in our third normal form designs of today. Were we wrong?

There are many who believe that for every action, there is a reaction, an inverse action. It is somewhat humorous to consider that if we had the query tools available to us way back before we got relational -- when our designs were not quite so good, it would be actually easier to implement end user access to operational data. Were we wrong?

No, we were not wrong, production systems solve many of then clerical problems of a firm. The better designed their underlying databases, the better they provide for the long term health of the organization. But unfortunately, though, the better our production database design, the more difficult will be the job of an end-user given license to access this data.

The whole notion of users having access to operational data is fraught with issues...”

Michael Corcoran, Director of Marketing for Information Builders writing form **3X/400 SYSTEMS MANAGEMENT** magazine, September, 1995 edition:

“*Production data --- the information created and maintained by transaction systems --- is rarely structured to enable rapid decision making. The answer is often a data warehouse, a relational repository of information designed especially for end-user query, analysis and reporting*”

“Building an effective data warehouse involves more than just copying data from one database to another and turning users loose with PC-based query and reporting tools.”

From IBM’s BI Web Site

Getting smarter In simple terms, business intelligence is the gathering, managing, analyzing and sharing of information in order to gain insights that can be used to make better decisions. It's the combination of a number of advanced techniques, including data warehousing, data mining and decision support.

A business intelligence solution can turn raw data into a powerful customer relationship management system that can help you create stronger, more profitable relationships with customers. Business intelligence systems also can help you identify new opportunities as well as create strategies and products that can anticipate customer demands.

Businesses in every industry use business intelligence to work smarter, find new ways to retain profitable accounts and trim unprofitable ones to get the greatest return on their information investment.

Unfortunately, creating a business intelligence system is not like creating a spreadsheet to analyze quarterly results. Your company may have gigabytes of existing data. e-business is generating enormous new volumes of data, leading to terabyte-sized databases. An effective business intelligence system has to be able to bring together information from multiple sources and grow seamlessly while handling ever more complex and demanding analyses.

Better tools for the job For many organizations, there's an easy alternative--and one you already know very well--AS/400e computers from IBM®. To date, more than 500,000 AS/400s have been installed worldwide to help them run their business.

A business intelligence pilot solution can be quickly implemented at a relatively low cost. As the solution grows and becomes more critical to the business, the traditional reliability, manageability and ease-of-use of AS/400 servers take on increasing importance.

With the latest AS/400e servers, the platform also provides significant performance advantages. A realistic business intelligence system needs to be accessed by many users at the same time. AS/400 servers were designed from the beginning as true multiuser business computers, with all the features and functions required to ensure that they can support many concurrent users accessing different applications and still provide excellent response times.

Custom solutions The new AS/400e servers can be customized to provide exactly the level of performance needed--with maximum memory of 40 gigabytes and maximum disk storage of more than four terabytes in a single system and 128 terabytes in a clustered environment--working for you.

The AS/400 operating system, OS/400r Version 4 Release 4 (V4R4), introduces a number of other enhancements that help make AS/400e servers an obvious choice for business intelligence. With this release, DB2® Universal Database for AS/400 will now support the storing, managing and indexing of all forms of information-including binary objects like spreadsheets, word processing documents and multi-media objects-within the database.

In addition, we've further enhanced the parallel processing support of DB2 UDB for AS/400, the ability to separate a large query into a number of smaller tasks that can be spread across multiple processors. The result is much faster response times. A number of Structured Query Language (SQL) enhancements, including the ability to update one table based on values from another table, bring further performance improvements. SQL permits creation of new encoded vector indexes (EVI), which greatly improves query performance. Using this advanced indexing feature, DB2 runs queries in 35 seconds that used to take over two hours.

IBM also has taken steps to extend the storage capacity and flexibility of AS/400 servers while helping to reduce costs. Integrated Hard Disk Compression (IHDC) permits data to be compressed and decompressed by a special chip on the disk controller, permitting up to four times more data to be stored on each disk. Need even more space for your warehouse? A Hierarchical Storage Management (HSM) function lets you use any combination of high-performance disk, compressed disk and tape for your information. No matter where the data is stored, a query can be staged without special programming. It's automatic, transparent to users and gives you up to 4x the storage space of high-performance disks.

The intelligent solution You can never be too smart or too fast. In today's competitive marketplace, there's a tremendous value in having the right information in the right place at the right time. With AS/400 servers, you can quickly and easily turn information into business intelligence. It's a profitable approach to the marketplace.

G221-8000-02

What is the Issue?

1. Users believe they need access to operational data for decision making purposes

2. Users direct, control the firm's resources. They bring in and spend the firm's money

3.. Despite the rhetoric, End users do not have ad-hoc access to operational data

4. IS managers are not motivated to provide users with access to operational data

5. Complex relational database functions such as join are too complex for computer neophytes

the "issue" continued

6. Query Tools cannot make up for poor end-user database design characteristics

7. IBM's Query product has major deficiencies in security and ease-of use - especially when data is normalized

8. Users have tried to do it themselves with PCS but most don't really want to be DP Managers

9. Most users don't want to re-key their data or be responsible for downloads of normalized data.

10. Client/Server "user" tools had been slow and complex (really programmer tools)

What is the Issue?

1. **Users believe they need access to operational data for decision making purposes**

No package or canned solution can be everything to everyone

Data voids in natural package information flow must be made up by programs or query tools

Programmers are the biggest users of today's query tools --- this gives us a clue

Users can be more productive when they can view their data as they wish without programmer intervention

Users who can use today's tools have more skill than should be required for data access. Users should be users, not programmers.

What is the Issue?

2. **Users direct, control the firm's resources**

   They bring in and spend the firm's money

Users are everyone else but IS plus IS

Users include all the managers and the CEO

Users have the power in the organization

Users can buy what they want

They have bought what they think they need in the past - PCS etc. but this wasn't exactly the solution

If they could buy it they would! If someone could convince them they could provide the service, they would buy it --- with or without IS

Examples - Separate User LANs, Oracle and Sybase Unix servers etc.

What is the Issue?

3.. Despite the rhetoric, End users do not have ad-hoc access to operational data

When IS buys a Query package... the job is done?

When IS runs a Query for a user... the job is done?

When IS builds canned Queries... the job is done?

When IS installs Client Server...the job is done?

When a champion user emerges... the job is done?

When IS says the job is done... the job is done?

Is the job done?

What is the Issue?

4. IS managers are not motivated to provide users with access to operational data

End User Access is an afterthought, a by-product of an application... but not an application itself requiring planning, resource commitment, and implementation

"

We provide them with all the information they need?

Users can't handle query products. They can't really handle their own PCS"?

The safe strategy is the "Appease Strategy" How can I get them off my back w/o system issues?

If they think it can't be done, maybe they'll go away!"

4. Continued -- IS Motivation

Besides: IS knows that:

1. The data is not shaped properly for end-users?

2. Logical joined files are difficult to build

Why would a sane IS manager want to create a performance issue in a downsizing era?

Who wants pain here, anyway?

What is the Issue?

5. **Complex relational database functions such as join are too complex for computer neophytes**

3NF or better   Production data- not for end-users?

Logicals create too much system stress?

Many query joins would be required for the user to have all of the data necessary

These joins are tough to do with staff professionals

Users aren't supposed to become IS Professionals just to do their jobs

We can't change the design or the production system would suffer?

What is the Issue?

6. **Query Tools cannot make up for poor end-user database design characteristics**

Arrays & complex structures are not good design techniques for user DBs.

RPG only 6-character naming conventions are not good design techniques for user DBs

Poorly named, commented / documented data elements not suited for end-user DB

Excessive or no file security not conducive to end-user access

Highly codified data not conducive to end-user understanding e.g. SEX | 1= Female, 2= Male

Code| 232= order. xGb = Pmt.

What is the Issue?

7. **IBM's Query product has major deficiencies in security and ease-of use - especially when data is normalized**

Most AS/400 shops have IBM product installed

No security in product. Must depend on AS/400 file security. User specifies file

Most IS Shops use Menus or other devices to inhibit unauthorized access

Joins and complex functions are more difficult than most users are prepared for

Query is fine with simple complete tables and simple requests with simple formatting

Production data not well suited for IBM Query

Why is IBM Query the #1 seller?? Should IBM Change?

What is the Issue?

8. Users have tried to do it themselves with PCS but most don't really want to be DP Managers

Whoever said PCS were so easy?

If PCS are so easy why have so many IS professionals stayed away from them like the plague?

Really good PC users have more an appreciation of the "lack of magic"

How many users are really good at dBase... and Paradox... and Excel... and Quattro Pro... and WordPerfect ... and Pagemaker?

The average user wants it to be there when they want it. After awhile, they are no longer *intrigued* by the tool. **They are more interested in results than work**

What is the Issue?

9. **Most users don't want to re-key their data or be responsible for downloads of normalized data.**

After a user sets up their DB or spreadsheet,
the job has just begun.

Analysis is postponed until data is keyed

Depending on frequency of data change, users
get good at keying ... makes up for IS backlog?

Users wanting analysis, not keying, come to IS for help!
IS provides Normalized data downloads

User must fit into PC application
User secretary hired to key PC data?

What is the Issue?

10. Client/Server "user" tools have been slow and complex (really programmer tools)

Name one simple Client/Server end-user tool!

Client/Server concepts are new to IS professionals

When used, IS does the queries!

Kelly's Rule ... a Phenomenon:

IS work associated with a new undertaking is directly proportional to:

A. The money spent on the tool or project
B. Top management's degree of interest (active or passive)

What is the Issue?

10. **Client/Server "user" tools have been slow and complex (really programmer tools)** continued

C/S tools "don't come cheap"

When pressed, IS will come up with a solution

If IS perception is that Query has not worked, the Client Series is a logical stopping point.

Client/Server has been notoriously slow

Too complex for users! IS becomes the end-user Friday-- better than admitting failure?

Maybe the tool isn't the problem
    Maybe it's the user?
    maybe it's the data?

Part 2 Tools used by End Users

But aren't end users already doing queries?

What are they using on AS/400?

Private Libraries for End-User Computing
PDM / SEU for Source Entry
DDS for End User DB Build/Change
SQL Query Manager for DB Build/Change
DFU for End-User Maintenance
SAVLIB for backup?
IBM Query for end user reporting
SQL Query Manager for end user reporting

Showcase Vista for advanced end user computing

Is something wrong with this picture?

There really is no pre-defined End-User Environment with the AS/400

End User Computing Scenario

with typical AS/400 tools

Function	Tool
CreateDB Structures	DDS via SEU
	CRTPF, CRTLF
	SQL Create Table
Modify DB Structures	Same tools preceded

	by rename, followed by COPY, DLTF New SQL Alter Table
Enter/Maintain Data	DFU, SQL Query Mgr.
Produce Reports	Query, SQL Query Mgr.

What about AS/400 Query?

IBM says:

"Query is a utility that allows principals, ***programmers,*** *secretaries, and other office personnel to interactively define, manage, and execute queries."*

What % of Users can actually use Query?

Query is as good as it's ever been (ported from S/36)

Why isn't the DB access problem solved by now!

What's the issue?
Education

One-On-One Training?
Query Functionality?
Data Complexity?

PART 3: How Production Databases Are Designed

First Step in Design
Regardless of DB implementation

1. Examine documents, reports, screens etc.

2. List data elements from each business process being re-engineered

3. Group data elements into natural groupings (First cut)

4. Split out repeating groups

5. Select a ***candidate key*** **which will uniquely identify each record in the structure**

Then!!!!!

On to Data Normalization!

Kelly Consulting Class Roster

‡‡
‡‡‡‡‡‡‡‡‡‡‡‡‡‡‡‡‡‡‡‡‡‡‡‡‡‡‡‡‡‡‡

Code	*Course Name*	*Duration*
AS015	AS/400 Database & DDS	1 Day

Class #	*Location*	*Start Date*
001	IBM Building in Scranton	2/15/94

Student Number	*Student Name*	*Company Name*	*Status*	*Days Absent*	*Grade*
001	Dennis Grimes	Glenbrook	PAID		
002	Patricia Piotroski	Kelly Consulting	ENR		
003	Harry Greshon	Nebs Buildings	PAID		
004	Dirk Blabbitt	Speech Assoc.	MAIL		
005	Samuel Adams	Adams Brewery	ENR		
006	Dionysius Smith	Mythology Inc.	MAIL		
007	Kathleen Friend	Dollhouses Unl.	PAID		

Instructor	Instructor #
Kelly, B.W.	400001

Schedule of Classes

Course Code	*Course Name*	*Days*	*Price*	*Start Date*	*Class No.*	*Location*
PC010	Making DOS Work For You	1	210.00	3/15	001	IBM Scranton
PC011	DOS Concepts & Facilities	A.M.	120.00	3/15	001	IBM Scranton
PC012	Advanced DOS -- Including MS-DOS 6.2	P.M.	120.00	3/15	001	IBM Scranton
AS015	AS/400 Database & DDS	1	210.00	2/16	001	Dallas-RSM
...						
...						
...						
...						
...						

Student Material

Conceptual Roster Data Structure

Class # (candidate key)

____

____

____

____

____

answer in back

Student Materials

Conceptual Schedule Data Structure

Class Number (candidate key)

____

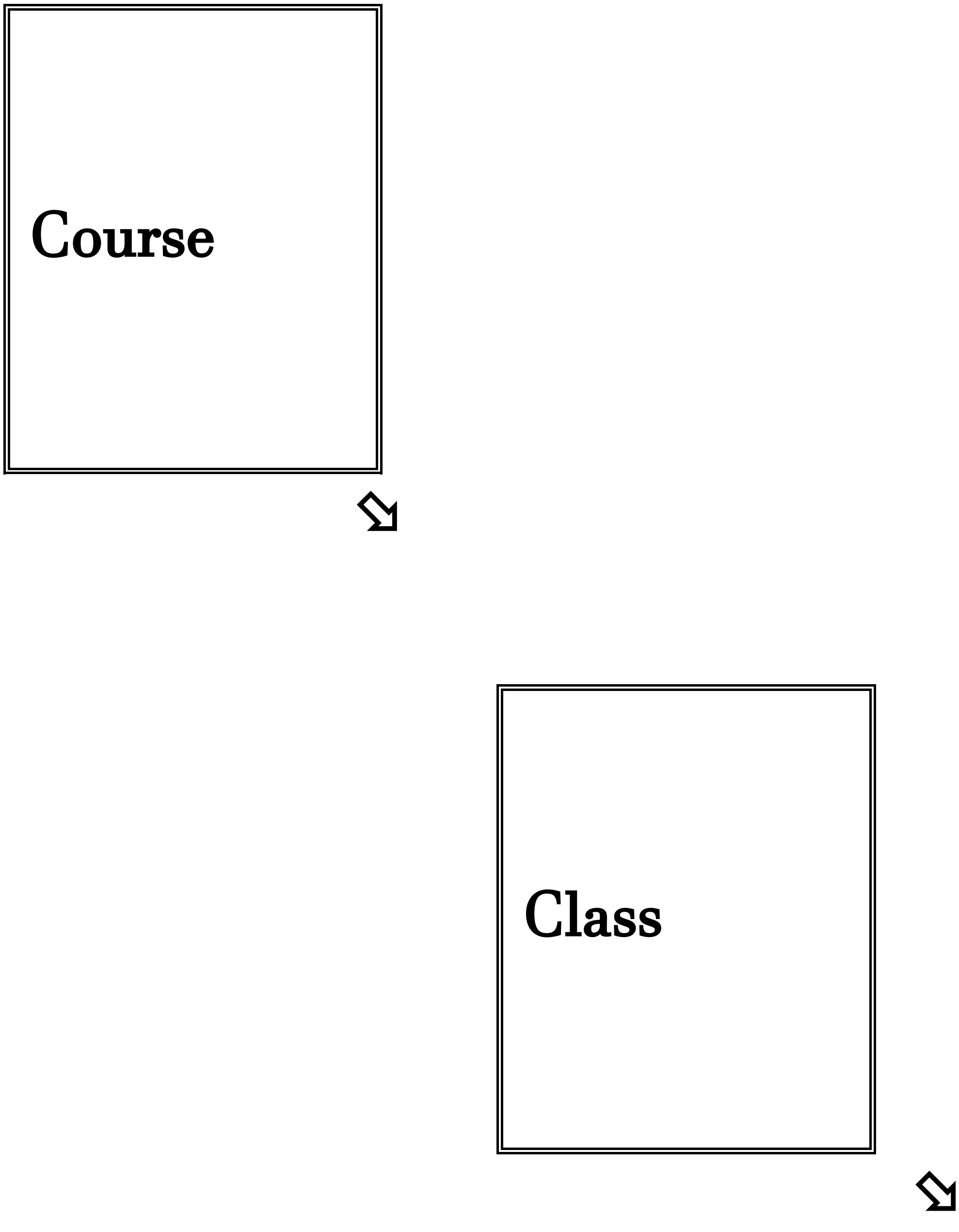
____

____

____

answer in back

Natural Groupings



Course

Class

Student

Data Relationships

After gathering data elements:

Assign to natural groupings:

Determine relationships of the groupings

Three types of relationships

One to One ONE ←→ ONE

One to Many	ONE ←→→ MANY
Many to Many	MANY←←→→MANY

Payroll Example

Employee Number←--

|→Employee Name

|→Personal Information

|→→Job History

|→Job Description

|→→Dependent Information

|→Payroll Detail

|→→Skills Information

|→→Education History

Many side of relationship becomes a new entity

Determine Education Relationships

Course

One course may have many classes

One course may be taken by many students

Class

One class represents one course

One class may have many students

Student

One student may take many courses

One student may be in many classes

Course←→→Classes Class←←→Course Student←←→→Courses

Course←←→→Students Class←←→→Students Student←←→→Classes

Data Base Normalization

1st Normal Form

2nd Normal Form

3rd Normal Form

Objective of database normalization is to re-structure data base files to third normal form. Then based on performance and usability, determine whether a different normal form is more appropriate.

Methodology - 1st normal Form

1. Gather data and group it

2. Split one to many and many to many groups into individual files

    a.k.a.--Move repeating groups to their own file

3. Assign candidate keys for the files

4. Determine if the business process can be served with the resulting structure

Caveat: Do not forget about step 4!!

Educational Structure 1NF

Course Code (candidate key)

Course Name

Duration

Price

Class # (candidate key)

Course Code

Location

Instructor #

Instructor Name

Class Start Date

Student # (candidate key)

Student Name

Company Name

Enrollment Status

Days Absent

Final Grade

Need a checkpoint to assure business process accommodated

Where are student / class relationships?

What needs to be added to student to make a composite key that uniquely identifies each record?

Or do we need a "Class List File" ---- UBETWEDO!

Second Normal Form

Assure that all of the non-key fields (a.k.a. attributes or attribute items) functionally depend on the key fields (identifying items)

	Dependent on
CUSTOMER_NO (key)	
INVOICE_NO (key)	
CUSTOMER_NAME	Cust#
CUSTOMER_ADDRESS	Cust#
AMOUNT	Cust#, Inv#
DATE_DUE	Cust#, Inv#

Functional dependency -- Some fields in the customer file do not depend on (are not related to) the composite key

This step removes potential for lost data

If no invoices were open for a customer, we would lose the customers address information etc.

If the address changes, we may have to change multiple records

Customer address has no functional dependency on invoice#

☞ Solution: Move to a new file
Create an invoice file and a customer master file

Class List File

Which key fields do each of the attribute fields depend on?

(Answer in back)

Dependent on?

STUDENT# key _______________

CLASS# key _______________

NAME _______________

COMPANY _______________

STATUS _______________

DAYS_ABS ______________

FIN_GRADE ______________

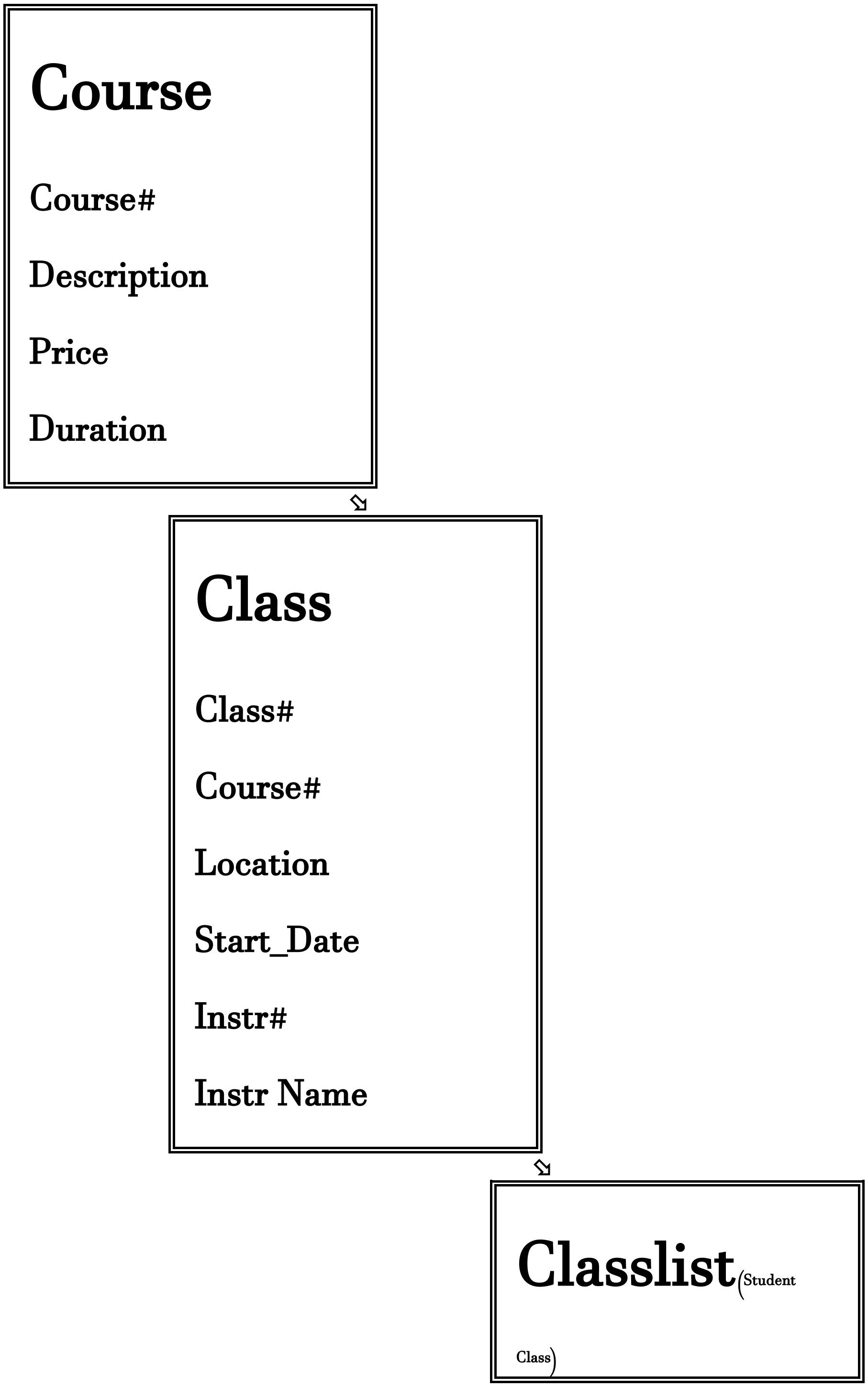
Each class can have many students and each student can attend many classes

Class# added to make key unique

Do all fields depend on composite key?

Should *Course Code* be part of this file?

Second Normal Form -- Solution



Course

Course#

Description

Price

Duration

Class

Class#

Course#

Location

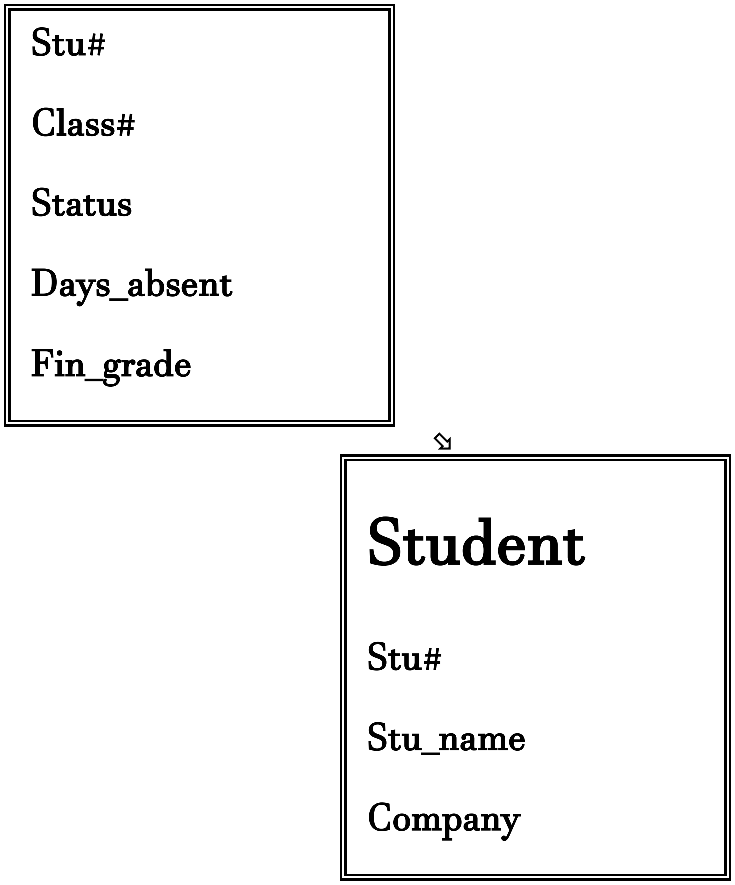
Start_Date

Instr#

Instr Name

Classlist(Student

Class)



Stu#

Class#

Status

Days_absent

Fin_grade

Student

Stu#

Stu_name

Company

What are the candidate keys?

Where does course# come from

Third Normal Form - 3NF

Remove transitive dependencies

Those attributes (non-key fields) that are dependent on other non-key fields

They are not in many files

(answer in back)

Order Data File Dependent Upon

ORDER_NO (key) ________________

CUST_NO ________________

CUST_NAME ________________

CUST_ADDR ________________

CUST_PO ________________

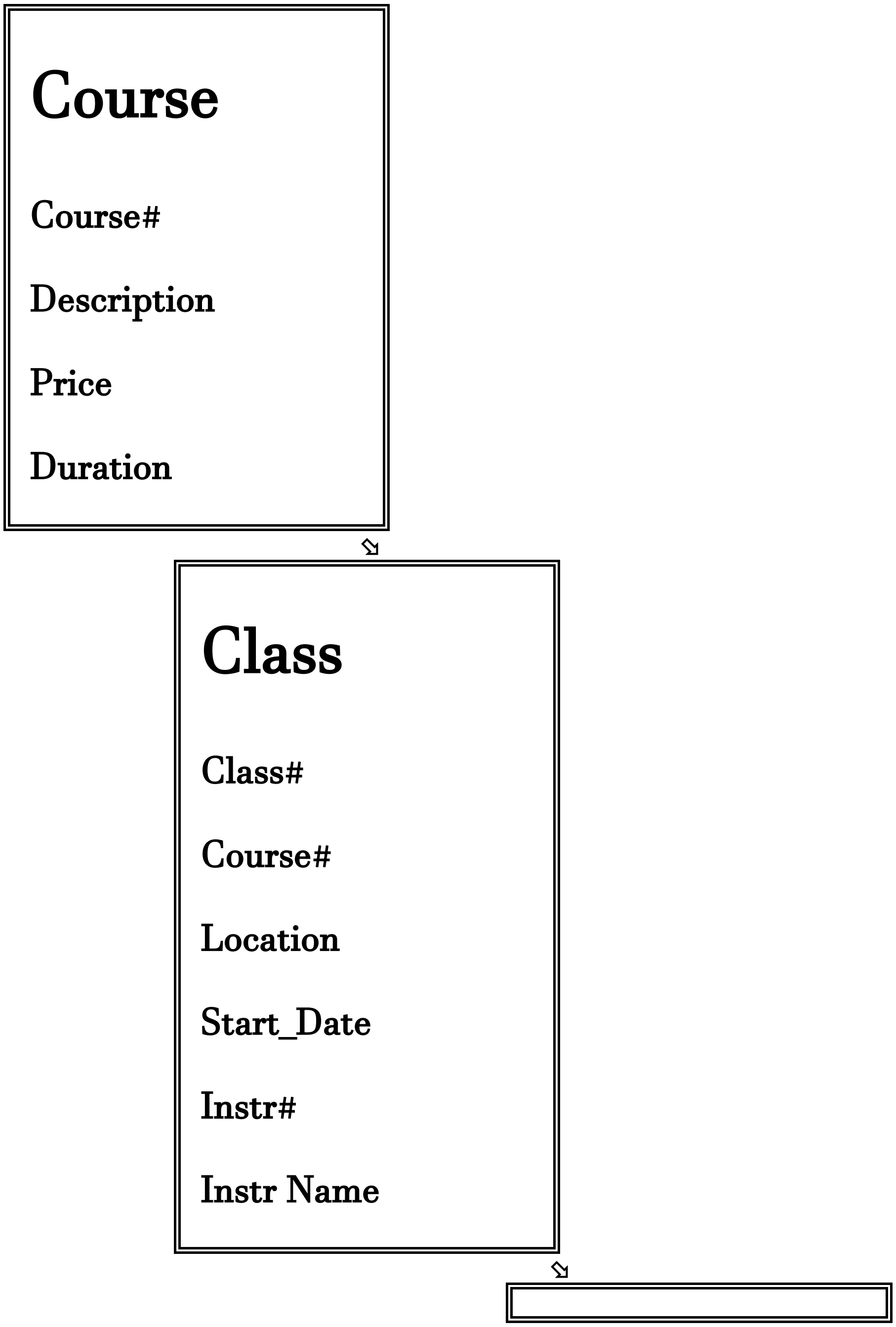
ORDR_DATE ________________

ITEM_NO ________________

QTY ________________

PRICE ________________

Do we have any transitive dependencies?



Course

Course#

Description

Price

Duration

Class

Class#

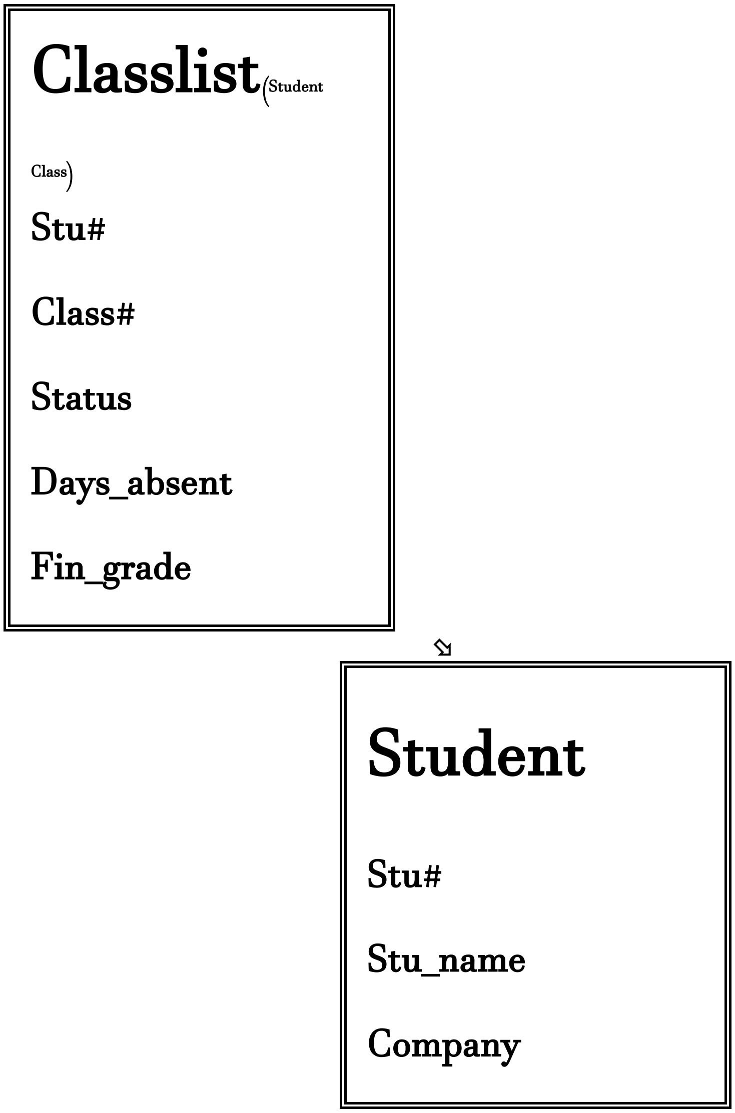
Course#

Location

Start_Date

Instr#

Instr Name



Classlist(Student Class)

Stu#

Class#

Status

Days_absent

Fin_grade

Student

Stu#

Stu_name

Company

What about Instr Name?

Should we divide the *Class* file?

Class

Class#

Course#

Location

Start_Date

~~Instr#~~

~~Instr Name~~

Instructor

Instr#

Instr Name

Is this a win or lose proposition?

Third Normal Form

Every field in the record
Depends on the key,
The whole key,
And nothing but the key!

Normalization Steps -- Summary

1. Create a conceptual data structure

2. Remove any repeating groups -- 1NF

3. Remove any functional dependencies -- 2NF

4. Remove any transitive dependencies -- 3NF

Which structure should you choose?

First Normal Form	Third Normal Form
Performance Better End-User Characteristics	Flexibility Less redundant data Overall ease of use

Suggest:

Take your data analysis to third normal form, then back off as strengths / weaknesses of specific designs materialize more fully

Don't stop your analysis until you reach third normal form -- it may be your best solution!

END Result should be Good, Flexible, Production DB Design

Does this Help or Hurt End Users ability to access data ad hoc?

If we were designing for End Users, would we do it this way?

Part 5 Designing Databases for End-User Computing

If you can start from scratch?

Minimize # of files - large complete records

Desirable attributes are similar to archival data

Objective: Make it easier for the user

Master, Transaction, Archival Records

Design to first normal form

Design with complete Information
e.g. Customer Data, Balance Data in one file

Avoid ubiquitous codes - use meaningful, self-evident codes...Expand codes with text
Make field names, descriptions meaningful

Summarize (reduce) Data when possible

Designing Databases for End-User Computing continued

Characteristics for Transaction Archival records

- Completed Records e.g. extended price

- Capture Customer name e.g. etc. vs #
  - Unaffected by changes to customer master record

- Capture point in time data -- e.g. price

- Expand codes to meaningful text
  - Also include codes for narrow reports

- Design a file for each code table for easy reference and expansion upon extraction

- Goal: Eliminate all multi-file dependencies
  Build a simple, complete record where possible

Part 6 Reengineering DBs for End user Computing
Making End user Data From Production Data

Design Methodologies:

1. **Denormalization**

2. **Traditional System Design**
   - **SDLC**
   - **SDLC/Prototyping**

3. **End User Reporting Analysis / Design**

Implementation Methodologies

1. **User Views of operational data**

2. **Data Extractions**

Part 6 Reengineering DBs for End user Computing

1. Using Denormalization

Bring Record Design Back To First Normal Form:

The formal denormalization process consists of four steps as follows:

1. Identify most important files for query purposes.

2. Examine production file relationships and select *skeleton* major groupings.

3. Add candidate transitive dependencies back into the major groupings.

4. Add candidate functional dependencies back into the major groupings.

Two additional desirable steps:

which are not formally part of the processes are recommended when the end user record design has reached completed first normal form. These are:

5. Recodify obscure coding and add textual descriptions.

6. Add needed end user elements.

Part 7 Reengineering DBs for End user Computing

2 Using Traditional SDLC

Systems Development Life Cycle
(As taught in Colleges and Universities)

A. Planning

B. Analysis

C. Design

D. Implementation

E. Support

Two formal methodologies for development

① System Development Life Cycle (SDLC)

② SDLC with Rapid Prototyping

SDLC - System Development Life Cycle

Traditional Systems Design Approach
- Focus on requirements as outputs

Systems Analysis:
- Problem Definition
- Requirements Analysis
- Project Justification - "go ahead!"

Systems Design:
- Logical System Design
  - where DB is built from requirements
- Physical System Design - actual system

Implementation: - production changes?
- testing, installation, training

System Cutover - production changes?
- Parallel, Pilot, Fine Tuning

SDLC

May be overkill but principles apply

Focus on problem at hand- End-user Reporting

Objective should be to leave production system programs untouched

Only prod changes should be for reporting DB, not production functional enhancements

If prod. system must maintain new DB tables use new programs rather than change old
- production system flow may be modified

Does the DB have all of the required information?
If not, add related tables to production system
during DB implementation

Prototyping Fits DB Reengineering Best

SDLC & Prototyping for Production System

SDLC concepts & heavy prototyping for
 End-User Reporting Systems

Prototyping
- Users play active role early
- Re-engineering errors caught earlier
- Users begin training earlier
- New users learn what they really want by studying / using the prototype

Prototyping Process for DB **Reengineering**

User area recognizes need for add'l information

User communicates need to DB prototyper
(an expert in Quick & Dirty systems design)
Resultant effort is highly modifiable

Prototyper uses a 4GL, or a screen designer, or a report writer, or even Word Processing or Spreadsheets to communicate design

Tool varies based on level of testing req'd If interaction required, tool must be more capable

User suggests changes , tests changed prototype, suggests new changes etc... new prototype versions quickly developed....

When user likes prototype, model becomes blueprint for DB changes needed to support reporting requirements

Benefits of prototyping

Design flaws caught before system built

Functional perception and User interface are first resolved -- often last in std. SDLC

User believes design fits the requirements

Underlying support system doesn't matter to the user - must be feasible

Significantly reduce development time & satisfaction

The ***WYSIWYG*** of application development

Start DB reengineering Prototyping by:

Collecting users reporting requirements

For what kind of questions must the data provide answers

Build sample structures - using DDS, SQL, or a
  PC product like Paradox or Access

Put a little data in the files

Design prototype representative report formats
  Start with the big hitters
  80-20 rule

Suggestions for Reporting DB

Objective: Get all elements for reporting into a "complete record" built around a major fact file

Order line items, - sales info
Inventory action file - all transactions
cumulative balances, time sequence
Class-list-file,
Vendor disbursements, distributions
Open Invoices
etc.

Add non-user suggested items for forensics purposes

If a data item is to be used within or to be operated upon to produce a report, it must be in the reporting database. - Take a sanity check

Suggestions for Reporting DB

Where end-user data does not exist in the production system:

Build table files
Use key relationships to production data

At extraction time:

Use production data "joined" to end-user reporting tables to construct reporting database.

Just as production systems are not perfect upon implementation, allow the reporting DB to grow & improve incrementally by plan

End User Reporting Analysis & Design (EUR)

Typical Systems Design Steps

1. **Examine documents, reports, screens, etc.**

2. **List data elements from each business process being re-engineered**

3. **Group data elements into natural groupings (First cut)**

4. **Split out repeating groups**

5. **Select a *candidate key* which will uniquely identify each record in the structure**

Plus the next items for EUR

EUR Analysis and Design.. Practical

End-User - in (start with output req'ts)

Vs. DB out approaches

Back to basics - not for production; ➔ *reporting*

Like a simple SDLC

All of the above requirements of DB design plus the following:

1. **Make a list of candidate query products for testing**
2. **Develop a reporting wish-list (from end user req’ts)**
3. **Examine all existing production data elements to add new fields as required for EUR**
4. **Unite data elements (both production elements and new elements derived from the wish-list) into natural reporting groupings**
5. **Build necessary database files with DDS or SQL Creates**
6. **Extract data from production files to populate fields and create missing data for testing**
7. **Test record design with EUR physical structures**
8. **Add missing elements to production system as necessary**
9. **Make production system changes and test applications as nec.**
10. **Build views or extraction routines to provide data access**

Implementation Methods

After the record designs are complete
Using one of three above methods:

Determine implementation method

1. Logical User Views, Same System

2. Data Extractions - Same or Different System

Implementation Method 1

User Views of Production Data - Same System

Objective: No user joins or complex DB operations

Build logical files to provide views to users

- Use CONCAT & SST for more meaningful data
- USE Access Path *REBLD, *DLY
  to minimize production DB performance hit
- Major Concern with Approach
  Production & Reporting Performance
- Add Code files for join text expansion

Implementation Methods

2. Data Extractions

   User data on same or separate "system"

   Combine files where possible to avoid multi-file dependencies

   Extract / replicate production data to provide user shaped record format files on same or different computer

   Major Concern with Approach
   - Disk Space Utilization
   - Refresh Frequency
   - Access Path Rebuild after Refresh

Add Code files for extraction text expansion

Data Extraction / Replication

vs. Logical Views & Indices

Best Approach --- & Best Performer

Production files are untouched / secure

Users files are segregated, no lock issues

Access Path *REBLD, *DLY on any
indexes built to help query processing

Can be in separate ASP or on separate machine
Sybase, Oracle, LAN Server Approach

Data Extraction Tools

Data Warehousing, Data Hub methodologies & products

Outfile from aXcess/400, Query, etc. - run once or several times per day

Datpropagator / Relational - planned refreshes

If unique requirements - build your own

Data Extraction / Replication

Home grown Methodology - Build your own
Data Refresh Plan -

1. Good DB design should include date/timestamp
   Even more appropriate with refresh cycles

2. If canned approaches don't fit or "too expensive"

   A. Build views over timestamp for data refresh

   B. Build HLL extraction / refresh programs
      Extraction programs use timestamp to refresh
      Get all DB changes for last "X" hours

   C. Use job scheduling to invoke extract/refresh on required basis... Application refresh tables

   D. Use full extract daily - off hours if possible

Data Extraction / Replication

Other tips:

Automate the process, Time of day etc.

Make sure files are user friendly

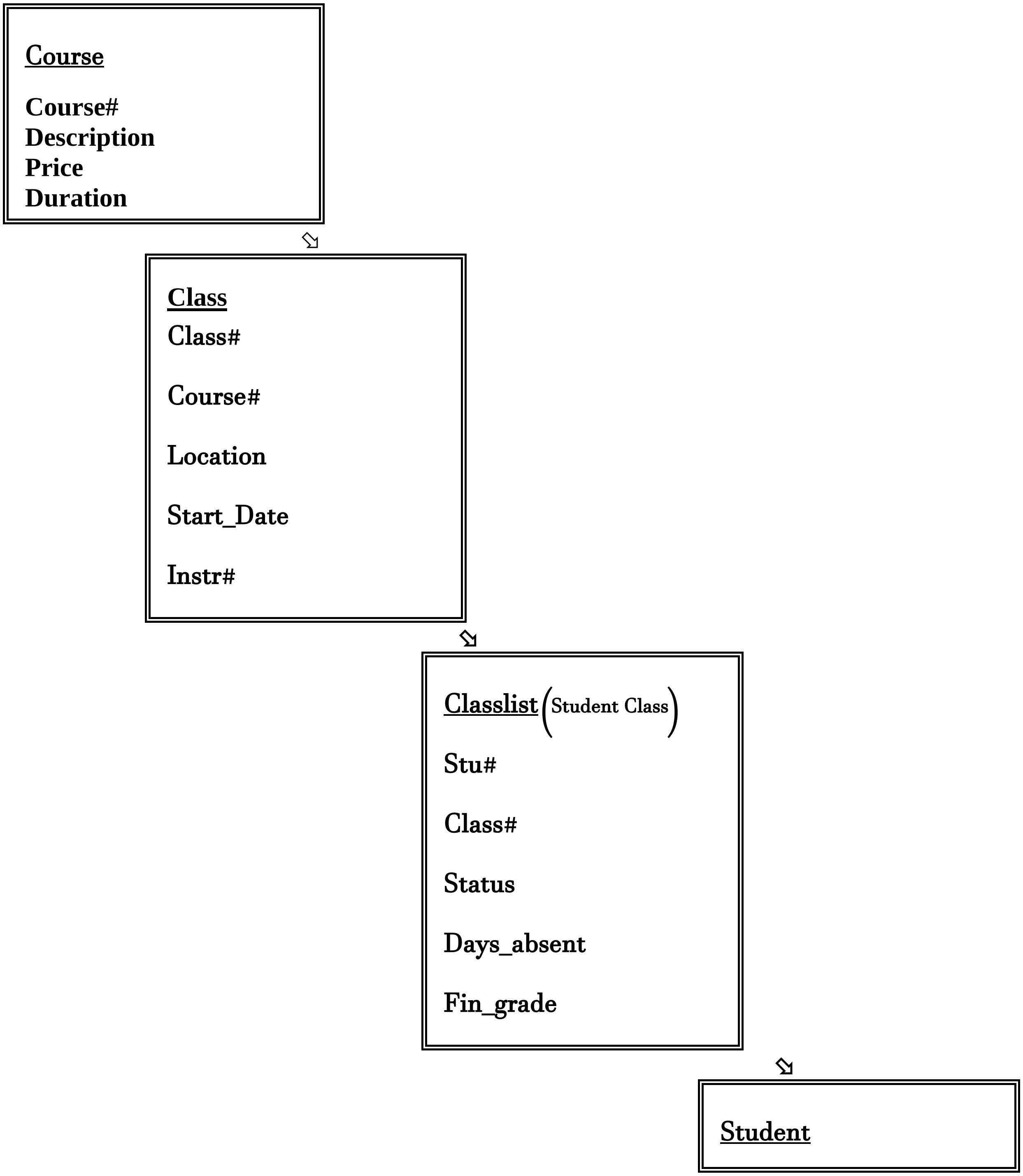
On same sys: Put Query Users in own subsystem
Menu option calls CL program
Moves job to Query Subsystem
Query Menu Option takes you back

Use SETOBJACC for small, frequently used files - performs in-memory

On Server Systems, execute in batch!!

Let's apply this to our student system!

Part 8 ... Examples Third Normal Form Data



Course

Course#
Description
Price
Duration

Class

Class#

Course#

Location

Start_Date

Instr#

Classlist (Student Class)

Stu#

Class#

Status

Days_absent

Fin_grade

Student

Stu#

Stu_name

Company

Instructor

Instr#

Instr Name

What should our Student Class File Contain?

De-Normalized (Answer in back)
Classlist File (Student Class)

Stu# (KEY)

____

____

Class# (KEY)

____

____

____

____

____

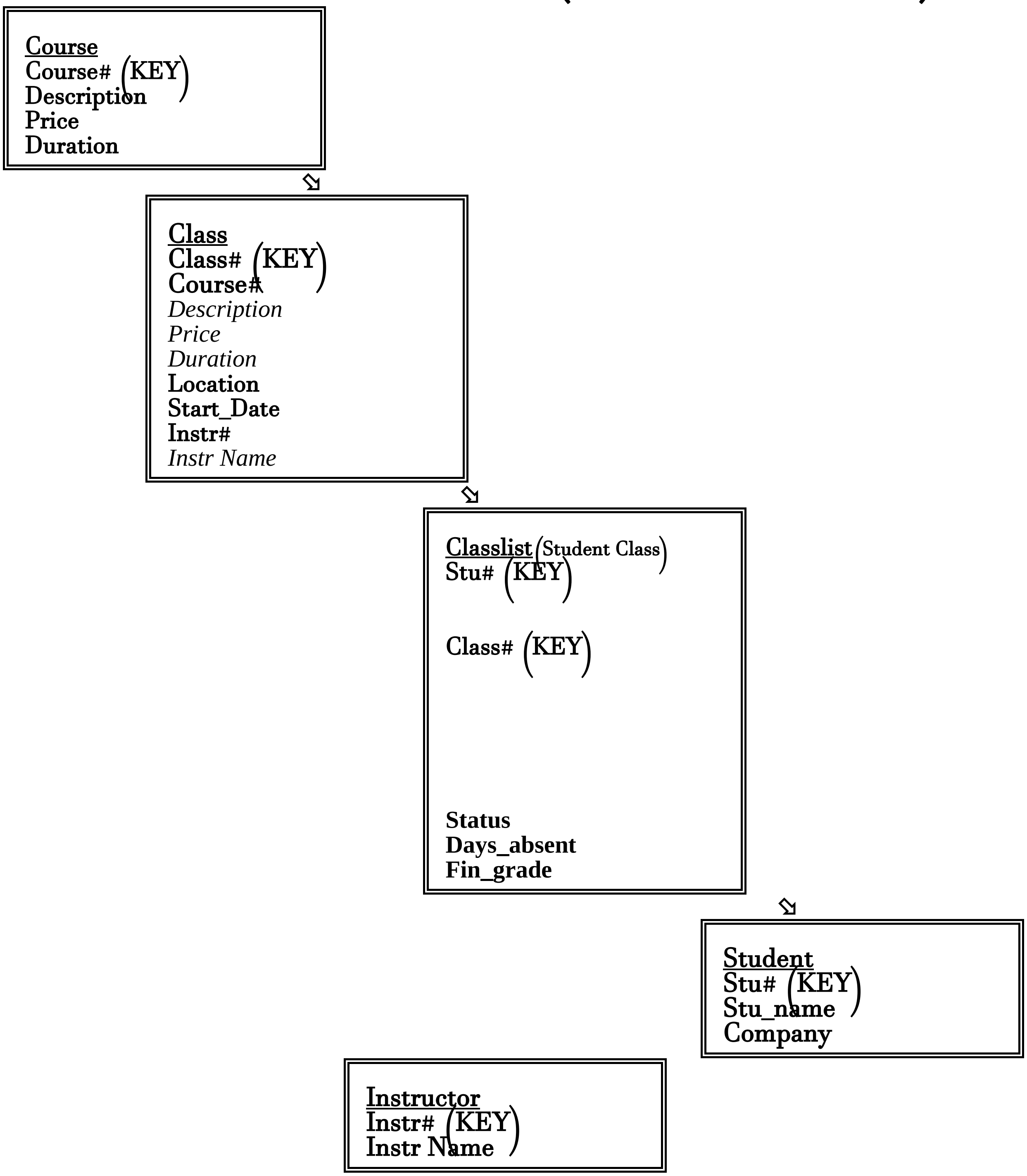
____

Instr#

____

Status
Days_absent
Fin_grade

First Normal Form (Answer in back)



Course
Course# (KEY)
Description
Price
Duration

Class
Class# (KEY)
Course#
Description
Price
Duration
Location
Start_Date
Instr#
Instr Name

Classlist (Student Class)
Stu# (KEY)

Class# (KEY)

Status
Days_absent
Fin_grade

Student
Stu# (KEY)
Stu_name
Company

Instructor
Instr# (KEY)
Instr Name

Pack the records with data- no repeating groups

LF Coding for   Join

```
||    |  |           |  |            Keywords
||    |  |           |  |            |
VV    V  V           V  V            V

A* Logical FILE (UCLASSLIST)
A     R  RCLLST                       JFILE(SCLASS CLASS COURSE+
A                                      STU INSTR )
A     J                              JOIN(SCLASS CLASS)
A                                    JFLD(CLASS# CLASS#)
A     J                              JOIN(CLASS COURSE)
A                                    JFLD(COURS# COURS#)
      J                              JOIN(SCLASS STU)
                                     JFLD(STU# STU#)
      J                              JOIN(SCLASS INSTR)
                                     JFLD(INSTR# INSTR#)

         STU#                        JREF(SCLASS)
         STNAME
         STCONM
         CLASS#                      JREF(SCLASS)
         COURS#                      JREF(COURSE)
         CDESC
         CPRICE
         CDUR
         CLOC
         CSTDAT
         INSTR#                      JREF(INSTR)
         INAME
         CSTATS
         DAYABS
         FINGRD
       K CLASS#
       K STU#
```

Not for your average user?

Other Join Possibilities

Table File for Code Lookup

Join just as in above example

User has more usable data

Additional System strain with another file?

Joined Record is better than the sum of its parts

DDS Physical File (W/ FIELD REFERENCE)

```
type of DDS. All DDS statements have an A in columns 6
|
|"*" means comment.  anything after * is treated as comment
||
||  "R" means that this is a Record Format
||    |
||    |  Names of Fields or Record Formats
||    |  |
||    |  |        "R" to use reference file for definition
||    |  |        |
||    |  |        |  Data type and length
||    |  |        |  |
||    |  |        |  |  Number of decimal positions
||    |  |        |  |  |
||    |  |        |  |  |            Keywords
||    |  |        |  |  |            |
VV    V  V        V  V  V            V
A* Physical File (UCLASSLIST)
A*
A                                    REF(FIELDREF)
A                                    UNIQUE
A     R  RCLLST                      TEXT('Class List')
A     R  RCLLST
         STU#     R
         STNAME   R
         STCONM   R
         CLASS#   R
         COURS#   R
         CDESC    R
         CPRICE   R
         CDUR     R
         CLOC     R
         CSTDAT   R
         INSTR#   R
         INAME    R
         CSTATS   R
         DAYABS   R
         FINGRD   R
      K  CLASS#
      K  STU#
```

Assure good column headings are used in reference file

Timestamp not necessary in Reporting File unless reporting requirement.

Putting the Data into the Extracted File

aXcess/400

Query/400

Outfile Support

HLL Program

Data Propagator/ Replication / Extraction Software

Build Table Files for Code Lookup

Join Code Descriptions at Extract Time

Advantages/ Disadvantages of Both Approaches

Feature	Logical Files	Extraction
Performance		√
Ease of Construction		√
Disk Space	√	
User Isolation (ASP, Separate machine etc.)		√
Security		√
Data Currency	√ *REBUILD N/A	
Code Expansion e.g. 1= Female		√

Does it matter to the User?

Summarize with Agenda

Examination of the Issue

Tools used by End Users

How Production Databases Are Designed

The Normalization Process

Designing Databases for End-User Computing

Re-engineering Databases for End User Computing---
- Views/ Extractions
- Denormalization
- End-User Reporting

Sample Databases
Data Warehousing, Data Replication, Data Extraction
End User Tools

Conclusions

End-User needs require attention

Treat End User Access as an Application

Good Operational DB design is inversely proportional to good query DB design

There are no magic AS/400 integrated solutions - one stop shopping but IBM is clearly committed - DB capabilities

All vendors are re-packaging, and re-labeling but they are not re-programming

Extractions/ Warehousing is the solution

Same system or different?

AS/400 DB2/400? ---- Of course!

Appendix A

Part 9 Data Warehousing

Data Warehousing

Traditional approach to taking data previously stored in on-line transaction processing systems and converting it to a standard format for users to access

This approach allows organization data to be used without disrupting the OLTP systems and the associated operational data

The warehouse is optimized for query processing

IE - Data Warehousing is the process we have been discussing in prior charts - extracting data for end-user access vs. direct access to operational data

You now know what a Data Warehouse is

Formal issue is the conversion tool(s) and DB to be used

Choices are many

What DB should be used for the warehouse?
Oracle? Sybase? DB2/X

Would the consultants recommend DB2/400?

All DB vendors are busy re-labeling their tools
to be part of a Data warehouse "suite"

All warehouse products require a good data model

If one file won't do it, build in DB constraints with Triggers and Integrity rules

Experts say SQL relational DBs are lacking

Red Brick Warehouse from Red Brick Systems of Los Gatos
Omni Warehouse (AIX et al.) from Praxis Int.

Some non-relational DBs - warehouses

Essbase from Arbor Software in Santa Clara
Lightship from Pilot Software, Cambridge Mass.
Accumate from Keenan Technologies, Cambridge Mass.
Express from Information Resources, Chicago
System W from Comshare, Ann Arbor, Mich.
Source - Open Systems Today

These products store data in a **multi-dimensional** form ... emerging

This allows for OLAP (on-line analytical processing) - term coined by E.F. Codd

Data warehouse and AS/400?

None of above products were DB2/400 compliant (Originally)

Data warehouse came from Unix camp

Not enough cycles or no DB on production

Needed another system for queries

Data warehouse sounded better than copy of data

AS/400 managers have a database for production
---Consider another system - as giving up

But the warehouse notion has great value
---on **production AS/400 or separate machine**
---A database or system just for queries - read-only

Data Warehouse Definitions:

A method for:

changing operational data into end user data

Getting information from production data without impacting operational systems

Not a product but a notion - requires a set of tools

Implemented on separate systems --
replication impact on production system

Operational Data is:

OE/BICARSA

Order Entry/ Billing / Inventory Control/ Accounts Receivable/ Sales Analysis

GLAPPR

General Ledger/ Accounts Payable/ Payroll

Special Apps
etc.

Typically records in 3rd normal form or >

Informational Data

Data Structured for End User access

Read-Only data for analysis only

Can be subset or superset of operational data

- Include Useful Data from many sources
- Include Calculated Fields
- Summarize when possible
- Clean Data -- scrubbed or expanded codes

Transaction or time oriented data

- trend analysis

Can be very large (larger than production)

The Warehouse is a separate system
Why?

Can test on production system
- & run small Query applications
- Can use a “backup system”

Significant processing load on production

Disk capacity more of an issue with
- Two databases

Production Apps must not slow down

Data Warehouse Parts

1. Warehouse -- system & database

2. Data Movement Tools - Op to WH

3. WH Analysis Tools

4. Warehouse Content Info Tools
   What data is in the warehouse?

5. Connectivity Midware to allow tools to talk
   to the warehouse

Data Warehouse

Must be a good performer and be scalable
ever growing needs

Types of warehouse engines
- Relational DB
- Multidimensional DB
  - (Summarized?)

Warehouse Types:
- Data Warehouse
- Data Mart

Warehouse Types

Warehouse Approach

- Big --- enterprise oriented 20-200GB
- Data Moved here from production
- Used to feed DataMarts

Datamart Approach

- Smaller 5-20GB
- Specific purpose analysis- short term tracking
- Data shipped from warehouse

Both approaches together typically needed

Warehousing Engines

Relational DB

Lots of Data, scalable
New parallel technologies can be employed
Extendable

OLAP (Multidimensional)

Pre-chewed food
Relational or specialized structure
Designed for predictable query performance
Take a long time to load
Most are DataMarts not Warehouses

Getting Data to the Warehouse

Can be difficult

Options:

1. Extraction on HOST; Conversion on HOST

2. Extraction on Host; Conversion at Warehouse

3. Extraction from Warehouse, Conversion at Warehouse

Media (Tape) or Network Extractions (we decide)

Extraction Tools:
DDM, DSPT, SAVCHGOBJ & Timestamp
ODBC, DRDA
Middleware
IBM, Oracle, others provide bi-directional access support for other DBs Packaged Solutions

Full DW Package comes with extraction tools

End-User Tools

A well structured DB is the best tool

So the AS/400 End user tools remain the same

Function	Tool
Create DB Structures	DDS via SEU CRTPF, CRTLF SQL Create Table No SQL Alter Table
Modify DB Structures	Same tools preceded by rename, followed by COPY, DLTF
Enter/Maintain Data	DFU, SQL Query Mgr.
Produce Reports	Query, SQL Query Mgr.

DSS Analysis tools

Analysis Tool Differentiation:

Decision Tools

- Query, Report Writers, EIS etc.
- You ask question ... tool hopefully gives answer

Data Mining Tools

- Tool Finds Answers w/o questions via associations
  - Helps to know what you’re looking for
- Uses Patterns, trends, clusters in data
- Widely misused term
- Neural Network Products can be used
  - AI/Expert System
- Data Analyst Function -- Real Heavy End User

Information about the Warehouse

Metadata

Data about data

Where data came from, age, format of warehouse etc.

Can be views of the data in business terms - model

Can assist in launching queries w/o you knowing much about the warehouse

Metadata must be created -- can't be bought

How far will Metadata take us? Too soon to know!

AS/400 Tools - circa 1997

Relational

- DB2
- SMP - Loosely coupled parallel databases
  Performance and scalability

OLAP

- Hoskyns AMIS/400
- Silvon Sales Tracker and Data Tracker
- SAMAC MIT/400
- More and More available

Data Movement

- DataPropagator OS/400
  - Needs some Midware and programming w/ APIs
- Multi-Dimensional DBS (OLAP) include their own data movement routines

Probably cannot find a package for all pieces
will need to code -- but do-able

Check out Appendix for updated list of AS/400 DW vendors

An ideal user product

Dumb terminals or PCs

Private Database Functions

Create Structures
Modify DB Structures

Reporting Database Functions

List, group, subtotal, select, order etc.
Outfile - with join table files - extraction

Glenbrook aXcess from Scranton, Pa. 18503
(No longer marketed)

Can also Create/Modify/Import .DBF and other DB structures

Helps Run queries on the ideal de-normalized database

****** No Longer marketed — Green Screen End user Tool***

Appendix B

AS/400 Datawarehousing Solutions --- 1997 Updated List

Now called Business Intelligence: http://www.as400.ibm.com/bi/

Product / Function	Vendor
Coglin Hill	Rodin
Showcase Strategy	Showcase Corp
Visual Warehouse	IBM
Data Propagator	IBM
DataMirror	DataMirror
DataGuide	IBM
DB2 Symmetric Multiprocessing	IBM
DB2 Multisystem	IBM
AMIS/400	Hoskyns
MIT/400	SAMAC
Showcase Strategy w/ESSBASE	Showcase
Sales Tracker Data Tracker	Silvon
Information Manager	Ferguson Info Systems

Visualizer	IBM
Showcase Strategy w/vista pro	Showcase
Impromptu Power Play	Cognos
BrioQuery	Brio Technologies
Lotus Approach	Lotus
Neural Network Utility	IBM
Intelligent Miner	IBM

* See web site for current IBM information on IBM offerings and Business Partner Offerings

Notes:

Many of these solutions are new since my DW book was originally published and the fact that there are so many bodes well for the future of AS/400 datawarehousing. It was not much more than one year ago that IBM recognized that it needed a new classification for its client series which would be an information repository for the new Data Warehousing products being developed for the AS/400 by IBM business partners, and also by IBM itself. When this was first made available the products were very sparse and reasonably unimaginative. This continues to change as the datawarehousing Client series now contains highly functional solutions for the AS/400 datawarehousing implementer.

Unlike most of the other IBM AS/400 Client Series categories, the data warehousing grouping contains a subcategory (parallel processing) for what by any other form would be viewed as hardware, not software. Most probably the explanation for this is that IBM has devoted a tremendous amount of resources to assure that its operating systems and database facilities can sustain performance when the warehouse is nothing short of massive.

And since IBM has included this category, it gives me the opportunity to provide a 1997 update for Data Warehousing using the Client Series datawarehousing subcategories as my presentation format. For your easy reference, I have included the DW solutions as listed in IBM's Client Series for Datawarehousing as shown in the below table.

IBM uses the web site http://www.as400.ibm.com/bi/ to keep the charts up to date. I provide these only as a starting point for further investigation. This presentation is generic in its treatment of data warehousing. Specific product information under the Business Intelligence heading is included at IBM's web site.

IBM’s Data Warehouse Solutions Client Series

Solution area	Product	Provider	Features
Warehouse management	RODIN	Coglin Mill	Build, administer, secure and describe the data warehouse. Extract data to data marts or presentation tools.
Warehouse management	ShowCase Strategy	ShowCase Corp.	Data distribution, administration, and tightly integrated analysis tools.
Data mart management	Visual Warehouse	IBM	Data mart administration and analysis tools in a packaged offering
Data propagation and transformation	DataPropagator	IBM	Data replication across all IBM DB2 platforms
Data propagation and transformation	DataMirror	DataMirror	AS/400 to AS/400 data propagation including S/36 files and bidirectional support.
Metadata	Data Guide	IBM	Information catalog tool for database administrator and end users.
Parallel database	DB2 Symmetric Multiprocessing for OS/400	IBM	Turbo-query feature to significantly enhance performance of query applications.
Parallel database	DB2 Multisystem for OS/400	IBM	Database functionality to take advantage of coupled AS/400 systems providing almost unlimited growth.
OLAP, data mart	AMIS/400	Hoskyns Group	Multi-dimensional

		PLC, IBM	analysis tools.
OLAP, data mart	MIT/400	SAMAC	Multi-dimensional analysis tools.
OLAP, data mart	ShowCase Strategy with ESSBASE/400	ShowCase Corp.	Multi-dimensional analysis tools.
OLAP, data mart	Sales Tracker	Silvon Software, Inc.	Multi-dimensional analysis applications for manufacturing and distribution.
Executive information system	InfoManager	Ferguson Information System (U.S.) InfoManager O.Y. (Europe)	Health and banking industry multi-dimensional executive information system applications.
Decision support tools	Visualizer	IBM	Query, report writing, graphics.
Decision support tools	ShowCase Strategy with VistaPro	ShowCase Corp.	Integrated query, report writing, graphics.
Decision support tools	Impromptu, PowerPlay	Cognos	Query, report writing, and desktop OLAP
Decision support tools	BrioQuery	Brio Technologies	Query, report writing, and desktop OLAP
Decision support tools	Lotus Approach	Lotus	PC-based database management
Data mining tools	Neural Network Utility	IBM	Visual development environment for mining data with neural networks; includes graphical training monitor, training language, data translation.
Data mining tools	Intelligent Miner	IBM	Decision support tool for mining data; includes neural network, decision tree, statistical algorithms, graphical interface, and data translation functions.

Appendix C

Instructor Material

Conceptual Roster Data Structure

Class # (candidate key)

Location

Instructor #

Instructor Name

Class Start Date

Course Code

Course Name

Duration

Student #

Student Name

Company Name

Enrollment Status

Days Absent

Final Grade

Conceptual Schedule Data Structure

- Class Number
    - Location
    - Class Start Date
    - Course Code
        - Course Name
        - Duration
        - Price

Class List File

Which key fields do each of the attribute fields depend on?

		Dependent on?
STUDENT#	key	______________
CLASS#	key	______________
NAME		Student# _____
COMPANY		Student# _____
STATUS		Student#, Class#

DAYS_ABS	Student#, Class#
FIN_GRADE	Student#, Class#

Each class can have many students and each student can attend many classes

Class# added to make key unique

Do all fields depend on composite key?

Should *Course Code* be part of this file?

NO, can be derived from the Class File

Third Normal Form - 3NF

Remove transitive dependencies

Those attributes (non-key fields) that are dependent on other non-key fields

They are not in many files

Order Data File	Dependent Upon
ORDER_NO (key)	________________
CUST_NO	Order No_______

CUST_NAME CUST_NO _____

CUST_ADDR CUST_NO _____

CUST_PO Order No _______

ORDR_DATE Order No_______

ITEM_NO Order No_______

QTY Order No_______

PRICE Order No_______

Student Class File

What should our Student Class File Contain?

De-Normalized

Classlist (Student Class)

Stu# (KEY)

Stu_name

Company

Class# (KEY)

Course#

Description

Price

Duration

Location

Start_Date

Instr#

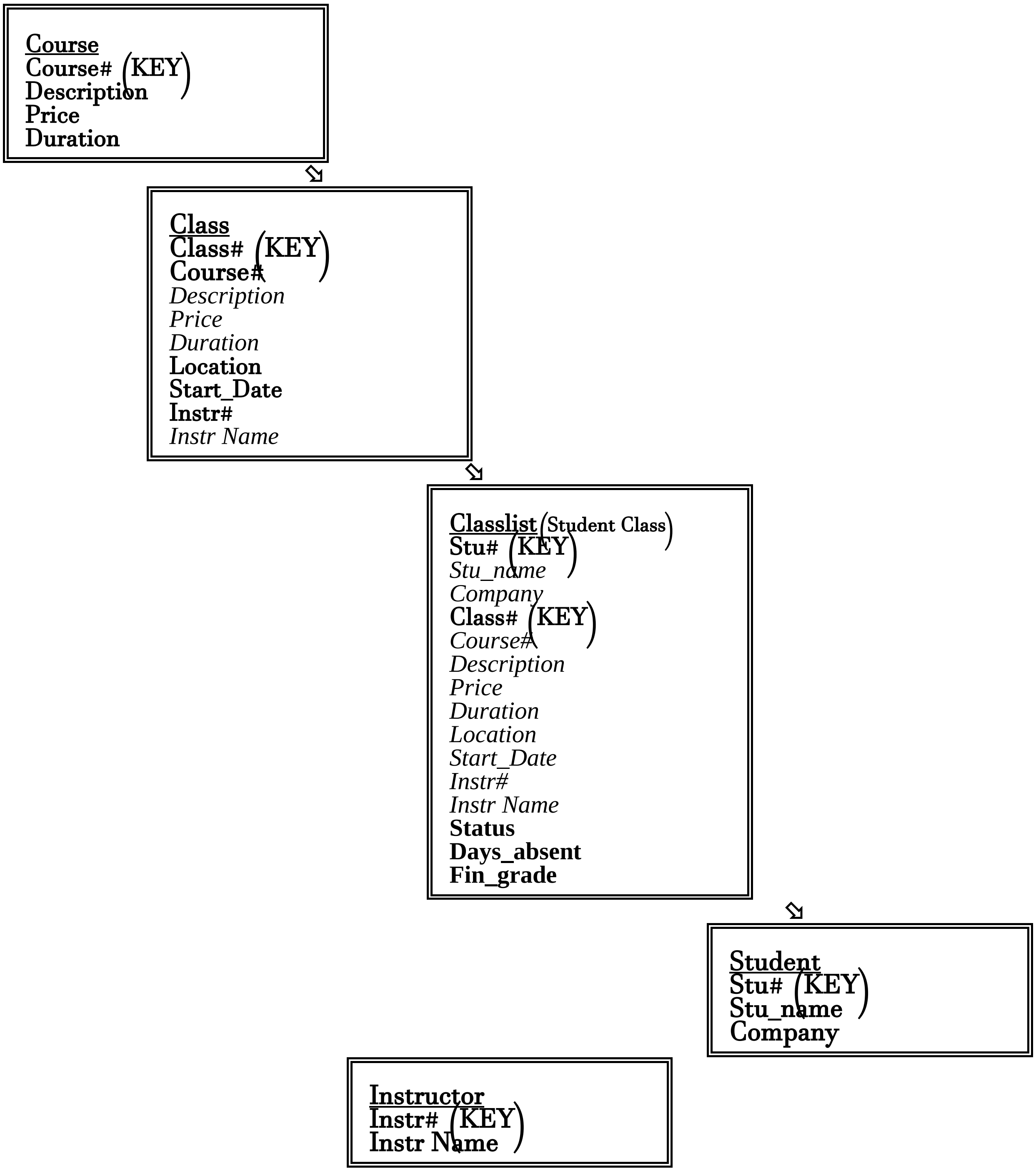
Instr_Name

Status

Days_absent

Fin_grade

First Normal Form



Course
Course# (KEY)
Description
Price
Duration

Class
Class# (KEY)
Course#
Description
Price
Duration
Location
Start_Date
Instr#
Instr Name

Classlist (Student Class)
Stu# (KEY)
Stu_name
Company
Class# (KEY)
Course#
Description
Price
Duration
Location
Start_Date
Instr#
Instr Name
Status
Days_absent
Fin_grade

Student
Stu# (KEY)
Stu_name
Company

Instructor
Instr# (KEY)
Instr Name

Pack the records with data- no repeating groups