LBSC 690: Information Technology
Lecture 08
Structured data and databases

William Webber
CIS, University of Maryland

Spring semester, 2013

Section 1

Databases

Data

Set As... | Filter... | Merge...

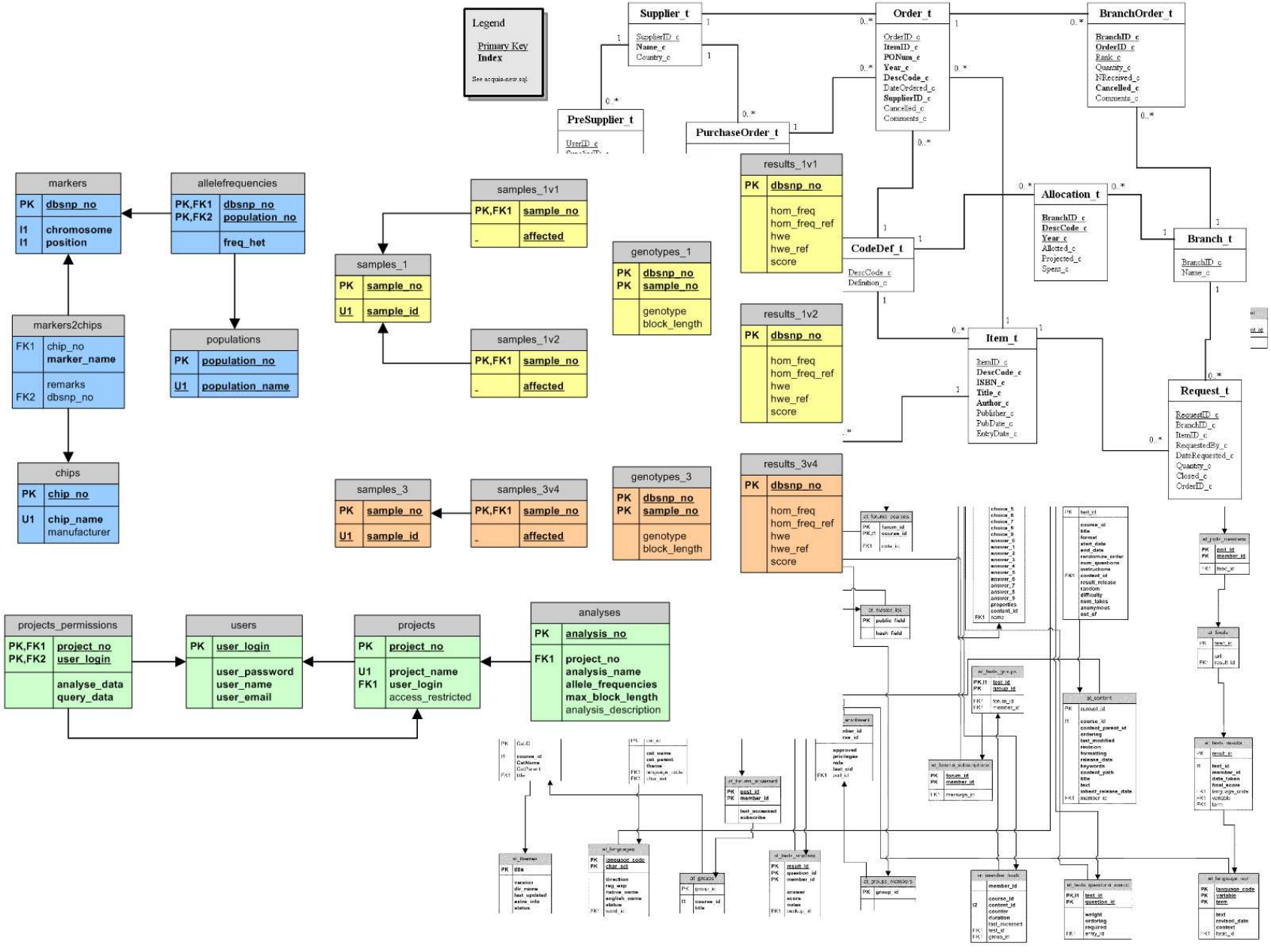
SampleID	Common_Name	Description	KeyHand	DigitHand	Hand	Individual
M9Akey217.1410...	keyboard	Akey	Left	NA	Left	M9
M9Bkey217.141032	keyboard	Bkey	Ambiguous	NA	Ambiguous	M9
M9Ckey217.1410...	keyboard					
M9Dkey217.1410...	keyboard					
M9Ekey217.141044	keyboard					
M9Enter217.1410...	keyboard					
M9Fkey217.141065	keyboard					
M9Gkey217.1410...	keyboard					
M9Hkey217.1409...	keyboard					
M9Indl217.141066	human_skin					
M9Indr217.140998	human_skin					
M9Kkey217.1410...	keyboard					
M9Midl217.141043	human_skin					
M9Midr217.141060	human_skin					
M9Mkey217.1410...	keyboard					
M9Nkey217.1410...	keyboard					
M9Okey217.1410...	keyboard					
M9Pinl217.141035	human_skin					
M9Pinr217.141002	human_skin					
M9Pkey217.141096	keyboard					
M9Qkey217.1410...	keyboard					
M9Rinl217.141020	human_skin					
M9Rinr217.141080	human_skin					
M9Skey217.141004	keyboard					
M9Space217.141...	keyboard					
M9Thml217.1410...	human_skin					
M9Thmr217.1410...	human_skin					
M9Vkey217.1410...						
M9Wkey217.1410...						
M9Xkey217.1410...						
M9Ykey217.14102						

Table 1. Clinical and Biochemical Data

Patient	Age	Sex	Follow-Up	Blood Pressure at Presentation*	Most Recent Blood Pressure Measurement*	Electrolyte Levels at Time of Diagnosis				Electrolyte Levels at Last Follow-Up			
						Sodium	Potassium	Chloride	Carbon Dioxide	Sodium	Potassium	Chloride	Carbon Dioxide
	y		*y*	*mm Hg*		← *mmol/L* →							
1	65	Male	5	170/94	120/80	145	3.1	105	30	140	5.2	110	28
2	69	Male	12	164/65	157/86	141	3.2	98	35	141	3.9	104	30
3	63	Male	11	178/96	130/95	141	2.9	100	28	144	4.0	107	26
4	43	Female	8	180/104	124/82	140	3.0	98	31	137	4.1	105	25
5	39	Female	5	184/132	128/80	141	3.9	102	29	140	3.7	106	28
6	76	Male	9	174/100	116/74	143	2.9	104	29	139	4.7	103	23
7	68	Male	6	180/105	155/76	140	3.1	98	32	142	4.2	109	28
8	69	Male	5	190/95	130/70	144	2.9	103	29	140	4.1	104	21
9	59	Male	7	180/116	145/99	144	2.4	102	35	139	4.3	104	30
10	55	Male	8	180/110	140/74	145	3.0	102	30	142	4.6	104	30
11	59	Male	6	165/102	112/68	142	3.0	106	30	142	4.8	108	30
12	50	Male	6	177/117	115/80	144	3.1	102	31	143	4.5	104	27
13	44	Male	6	160/110	130/82	141	3.0	106	29	140	4.3	103	29
14	64	Female	8	160/98	142/60	144	3.4	106	29	142	4.7	108	25
15	52	Female	13	150/104	104/76	142	3.3	105	24	137	4.4	106	25
16	52	Female	5	168/102	128/91	143	2.7	102	32	141	3.6	106	32
17	54	Female	17	180/110	101/71	143	3.0	105	33	139	4.4	101	30
						142	2.6	106	29	138	4.6	101	27
						142	2.6	98	32	137	3.6	98	26
						145	2.9	103	35	140	3.7	113	29
						143	2.6	103	30	146	4.6	108	26
						141	3.0	101	31	142	3.8	102	26
						143	3.8	99	31	143	4.8	105	24
						141	3.2	102	32	139	4.6	102	26

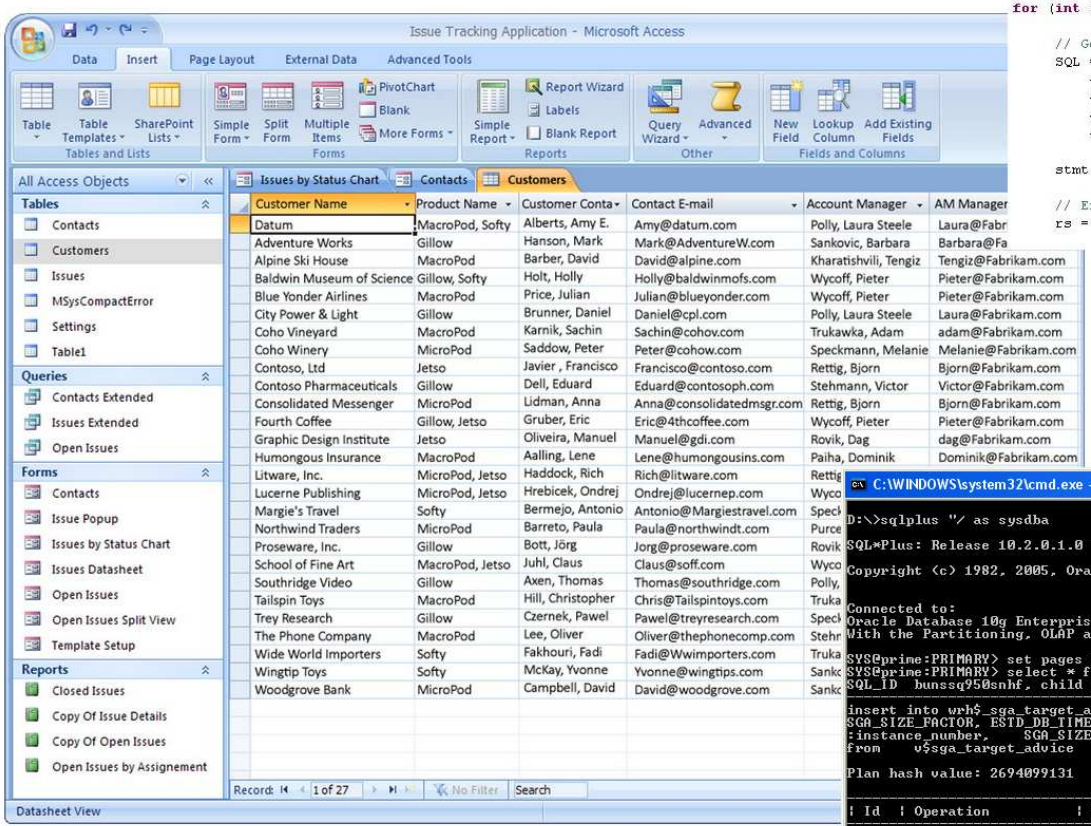
Name	Thread pitch (mm)	Minor diameter tolerance	Nominal diameter (mm)	Head shape	Price for 50 screws	Available at factory outlet?	Number in stock	Flat or Phillips head?
M4	0.7	4g	4	Pan	$10.08	Yes	276	Flat
M5	0.8	4g	5	Round	$13.89	Yes	183	Both
M6	1	5g	6	Button	$10.42	Yes	1043	Flat
M8	1.25	5g	8	Pan	$11.98	No	298	Phillips
M10	1.5	6g	10	Round	$16.74	Yes	488	Phillips
M12	1.75	7g	12	Pan	$18.26	No	998	Flat
M14	2	7g	14	Round	$21.19	No	235	Phillips
M16	2	8g	16	Button	$23.57	Yes	292	Both
M18	2.1	8g	18	Button	$25.87	No	664	Both
M20	2.4	8g	20	Pan	$29.09	Yes	486	Both
M24	2.55	9g	24	Round	$33.01	Yes	982	Phillips
M28	2.7	10g	28	Button	$35.66	No	1067	Phillips
M36	3.2	12g	36	Pan	$41.32	No	434	Both
M50	4.5	15g	50	Pan	$44.72	No	740	Flat

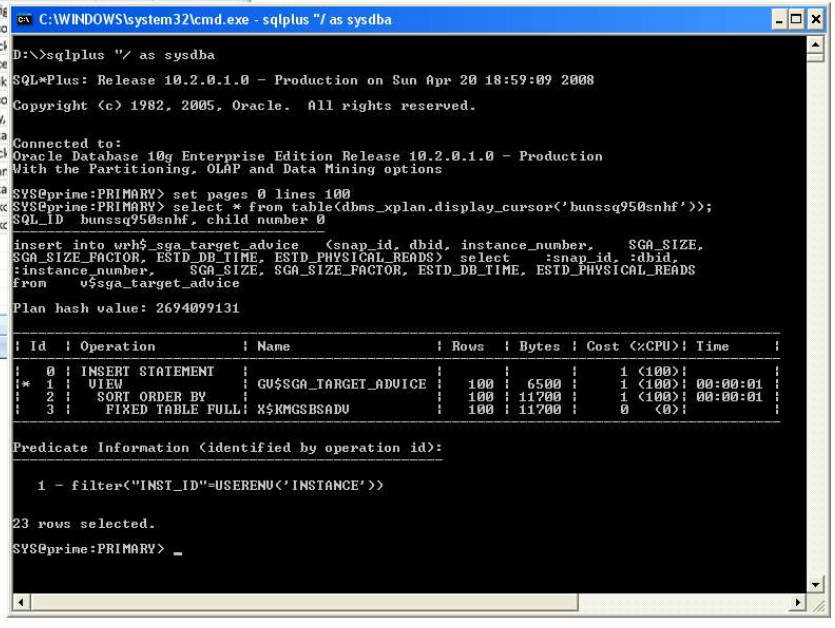
Schema


Legend
Primary Key
Index
Supplier_t
Order_t
BranchOrder_t
PreSupplier_t
PurchaseOrder_t
results_1v1
Allocation_t
Branch_t
CodeDef_t
Item_t
Request_t
results_1v2
results_3v4
markers
allelefrequencies
samples_1v1
samples_1
genotypes_1
markers2chips
populations
samples_1v2
chips
samples_3
samples_3v4
genotypes_3
projects_permissions
users
projects
analyses

Interfaces

```
// Get a connection to SQL Server 2005
Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
conn = DriverManager.getConnection(connectionUrl);

// Via the connection, mimic user requests by looping through the table
// and retrieving rows 1 at a time up to 20 rows;
// SalesOrderHeader id's start at 43659
for (int id = 43659; id < 43679; id = id + 1) {

    // Get the id and execute the query
    SQL = "SELECT c.FirstName, c.LastName, oh.SalesOrderID, "
        + "oh.OrderDate, oh.DueDate, oh.TotalDue "
        + "FROM Sales.SalesOrderHeader oh "
        + "JOIN Person.Contact c on c.ContactID = oh.ContactID "
        + "WHERE SalesOrderID = " + id;

    stmt = conn.createStatement();

    // Execute the query
    rs = stmt.executeQuery(SQL);
```





Systems


ORACLE
DATABASE
STANDARD EDITION ONE
Office
Access 2010
Microsoft
MySQL

Section 2

Structured data

Information without structure

Jane Doe (student id 1234-5678) is a student of LBSC 690, “Information Technology”. taught by Doug Oard. Her student id is 1234-5678. Her final mark for the subject is 87.

Information about an LBSC 690 student, presented in an “unstructured” way

Imposing structure upon information

Property	Value
Name	Jane Doe
Student id	1234-5678
Subject code	LBSC 690
Subject name	Information Technology
Mark	87
Instructor	Doug Oard

- Organize information into “property : value” pairs
- Often many choices about how to partition data into properties
- Not all data can be structured in this way
- Structure is something we impose on information

Data tables

Name	Student id	Subject code	Subject name	Mark	Instructor
Jane Doe	1234-5678	LBSC 690	Information Technology	87	Doug Oard
John Dee	2233-4455	LBSC 690	Information Technology	75	Doug Oard
Jane Doe	1234-5678	LBSC 771	Records Management	76	Adam Adamson
John Dee	2233-4455	LBSC 601	Information Retrieval	52	Doug Oard

- Where structure is common to many entities, can be organized as a table.
  - Each row of table corresponds to an entity or record.
  - Each column corresponds to a property or field.
  - Each cell gives the entity's value for that field.
- These (spreadsheet-like) tables are at the heart of databases.

Typing of fields

Property	Type	Constraint
Name	String	Required; Maximum 128 Characters
Date of birth	Date-Time	
Student id	String	Required; Exactly 9 Characters
Subject code	String	Required; Exactly 8 Characters
Subject name	String	Maximum 64 Characters
Mark	Integer	
Instructor	String	Maximum 128 Characters

- The fields can be assigned types and constraints
- This ensures that (for instance) words are not entered into a field that should hold numbers
- It also allows us to perform type-specific operations (for instance, find the amount of time between two dates)
- Properties, types, and constraints constitute the schema of the table

Record keys and indexes

Name	*** Student id**	*** Subject code**	Subject name	Mark	Instructor
Jane Doe	**1234-5678**	**LBSC 690**	Information Technology	87	Doug Oard
John Dee	**2233-4455**	**LBSC 690**	Information Technology	75	Doug Oard
Jane Doe	**1234-5678**	**LBSC 771**	Records Management	76	Adam Adamson
John Dee	**2233-4455**	**LBSC 601**	Information Retrieval	52	Doug Oard

- Key is a field or set of fields that uniquely identify a record
- Keys (and other fields) may be indexed to quickly look up records (matters when we have millions of records)

Design choices and levels of granularity

Property	Example
Name	Jane Doe

Property	Example
Given Name	Jane
Family Name	Doe

Property	Example
Given Name	Jane
Initials	J.
Family Name	Doe
Title	Ms.
Nick Name	Jay

- Different levels of granularity are possible
- Generally, more granular is better, but you can go overboard
- Also, choices about representation (states as codes? names?)

Designing a single-table database

The library director asks you to create a database to record a list of “Friends of the Library”. The director wants to record:

- Name and contact information
- Age, gender, and ethnicity (optional)
- Total amount of donations

Your tasks:

- Come up with a schema (a list of typed properties) for the database
- What will be the key?
- Has the director missed any fields you think should be there?

Section 3

Relational data

Repeated information

Name	* Student id	* Subject code	Subject name	Mark	Instructor
Jane Doe	**1234-5678**	**LBSC 690**	Information Technology	87	Doug Oard
John Dee	**2233-4455**	**LBSC 690**	Information Technology	75	Doug Oard
Jane Doe	**1234-5678**	**LBSC 771**	Records Management	76	Adam Adamson
John Dee	**2233-4455**	**LBSC 601**	Information Retrieval	52	Doug Oard

Note that in the above database, we have repeated information about courses. This leads to several problems:

- Wastes space in the database
- Requires more data entry
- Leads to inconsistencies if information is modified

Compound entities

Field	Example value
Name	Jane Doe
Student id	1234-5678
Subject code	LBSC 690
Subject name	Information Technology
Mark	87
Instructor	Doug Oard

- The problem is that our table is really a compound of (actually more than) two distinct entities
  - Student
  - Subject
- Also, we don't separately store information about a class.
- What happens if there are no students enrolled in a class?

Decomposition

Field	Type	Properties
Student id	Character	Primary Key
Name	Character	
Mark	Integer	
Subject	???	

Table: Student

Field	Type	Properties
Code	Character	Primary Key
Name	Character	
Instructor	Character	

Table: Subject

- Separate into two tables or entities
  - One for student
  - The other for subject
- But now how to mark which class a student is enrolled in?

Foreign key

Field	Type	Properties
Student id	Character	Primary Key
Name	Character	
Mark	Integer	
Subject code	Character	Foreign Key $\rightarrow$ Subject(Code)

Table: Student

Field	Type	Properties
Code	Character	Primary Key
Name	Character	
Instructor	Character	

Table: Subject

- Subjects are identified by their codes (their primary key)
- We place the subject code into the Student table to say which subject the student is taking
- This is known as a foreign key

Splitting into two tables

Student					
Student id	Name	Subject code	Subject name	Mark	Instructor
1234-5678	Jane Doe	LBSC 690	Info. Tech.	87	Doug Oard
2233-4455	John Dee	LBSC 690	Info. Tech.	75	Doug Oard
1234-5678	Jane Doe	LBSC 771	Record Mgmnt	76	Adam Adamson
2233-4455	John Dee	LBSC 601	Info. Ret.	52	Doug Oard

Figure: Before

Student			
Student id	Name	Mark	Subject code
1234-5678	Jane Doe	87	LBSC 690
2233-4455	John Dee	75	LBSC 690
1234-5678	Jane Doe	76	LBSC 771
2233-4455	John Dee	52	LBSC 601

Subject		
Code	Name	Instructor
LBSC 690	Info. Tech.	Doug Oard
LBSC 771	Record Mgmnt	Adam Adamson
LBSC 601	Info. Ret.	William Webber

Figure: After

- Redundancy of subject information removed.
- Subject code acts as foreign key – primay key link

Further decomposition

Field	Type	Properties
Student id	Character	Primary Key
Name	Character	
Mark	Integer	
Subject code	Character	Foreign Key $\rightarrow$ Subject(Code)

Table: Student

Field	Type	Properties
Code	Character	Primary Key
Name	Character	
Instructor	Character	

Table: Subject

We still have redundancy in our schema design:

- Where is it?
- Improve the design so as to remove the redundancy

Joins

Student		
Student id	Mark	Subject code
1234-5678	87	LBSC 690
2233-4455	75	LBSC 690
1234-5678	76	LBSC 771
2233-4455	52	LBSC 601

Table: Student

Subject		
Code	Name	Instructor
LBSC 690	Info. Tech.	Doug Oard
LBSC 771	Record Mgmnt	Adam Adamson
LBSC 601	Info. Ret.	William Webber

Table: Subject

```
Student JOIN Subject ON subjectcode=code
```

Join				
Student id	Mark	Subject code	Subject Name	Instructor
1234-5678	87	LBSC 690	Info. Tech.	Doug Oard
2233-4455	75	LBSC 690	Info. Tech.	Doug Oard
1234-5678	76	LBSC 771	Record Mgmnt	Adam Adamson
2233-4455	52	LBSC 601	Info. Ret.	William Webber

Table: Joined student-subject table

- The `JOIN` operation allows us to reconstruct composite data as required on demand

Project

Student		
Student id	Mark	Subject code
1234-5678	87	LBSC 690
2233-4455	75	LBSC 690
1234-5678	76	LBSC 771
2233-4455	52	LBSC 601

Table: Student

```
SELECT studentID, subjectcode FROM Student
```

Projected	
Student id	Subject code
1234-5678	LBSC 690
2233-4455	LBSC 690
1234-5678	LBSC 771
2233-4455	LBSC 601

Table: Projected table

- The SELECT operation allows us to extract only desired columns
- Can be applied to joined tables

Restrict

Student		
Student id	Mark	Subject code
1234-5678	87	LBSC 690
2233-4455	75	LBSC 690
1234-5678	76	LBSC 771
2233-4455	52	LBSC 601

Table: Student

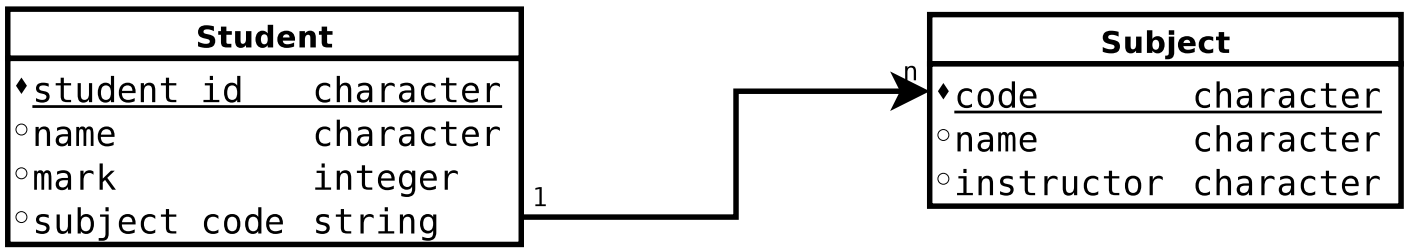
```
WHERE subjectcode="LBSC 690"
```

Student		
Student id	Mark	Subject code
1234-5678	87	LBSC 690
2233-4455	75	LBSC 690

Table: Student

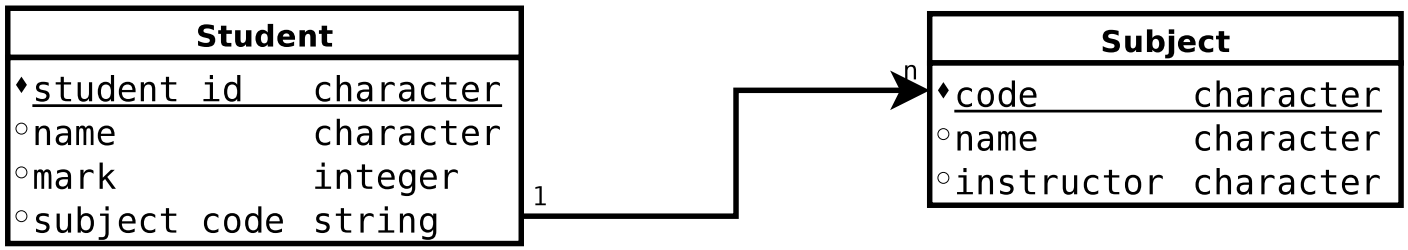
- The *WHERE* clause allows us to select only rows we are interested in

Entity-relation diagrams



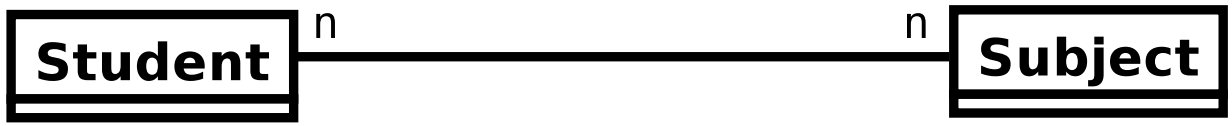
- Diagram relationship between entities during design phase.
- Several standards; we're looking at a simple one.
- Each entity represented by box, with (optionally) attributes of entity listed in box.

Relationships in ERDs



- Relationships in ERDs shown by arrow
- Arrow points from entity that has reference (here, from the foreign key attribute), to entity that is referenced
- Cardinality of membership shown at connection to entity, generally either 1 or *n* (for “many”).
  - Here, we are asserting that a student can have (be enrolled in) only one subject, but a subject can be had by (enrol) many students (a one-to-many relationship).

Further decomposition



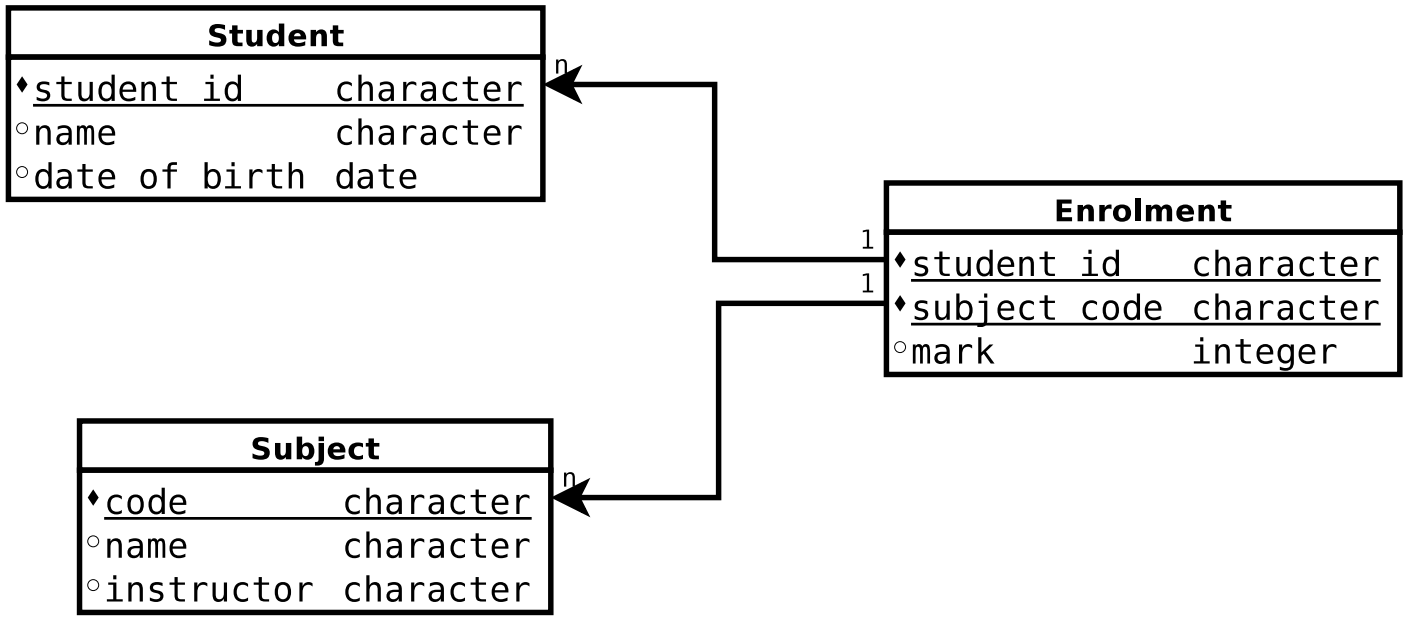
- Previous slide we said “a student can be enrolled in only one subject”; however, this is clearly wrong.
- The correct statement is:

Definition (Student-subject relationship)

A student can be enrolled in many subjects; a subject can have many students enrolled in it.

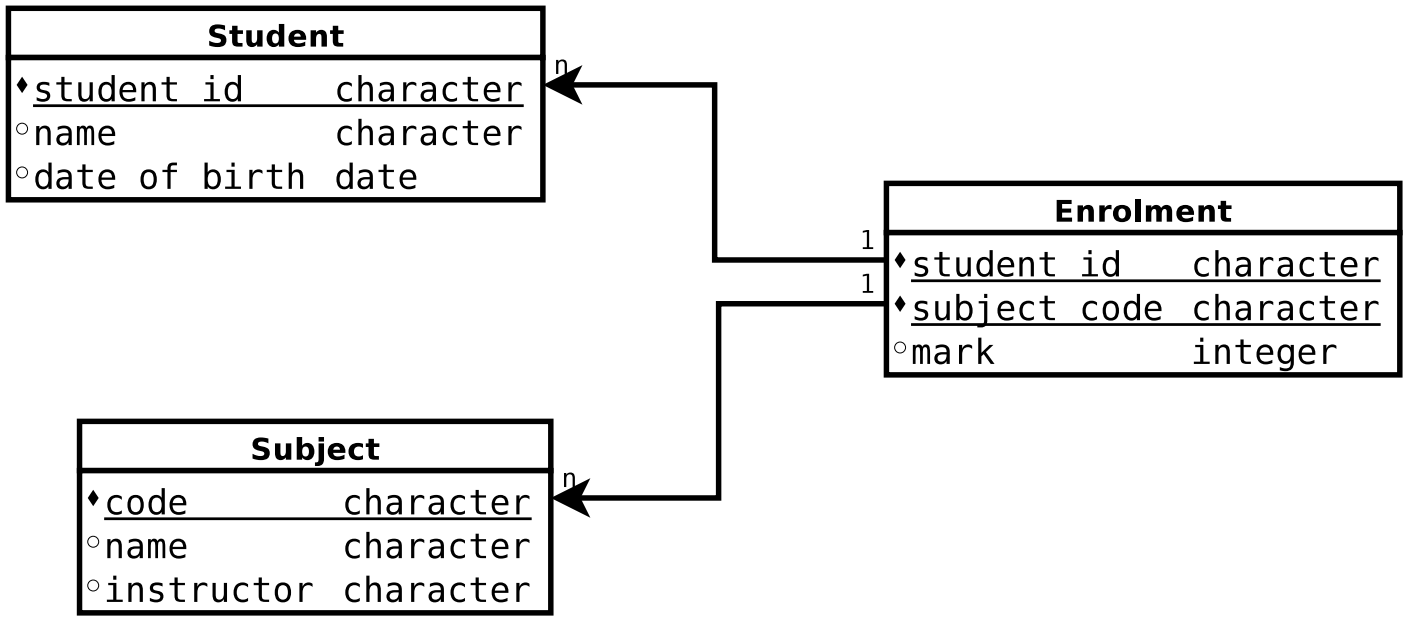
- This is a many-to-many-relationship.

Many-to-many relationships



- For many-many relationships, we need a separate entity (table) recording relation.
- This separate entity also holds ancillary data that is common in such relations (here, “mark”).

Further decomposition



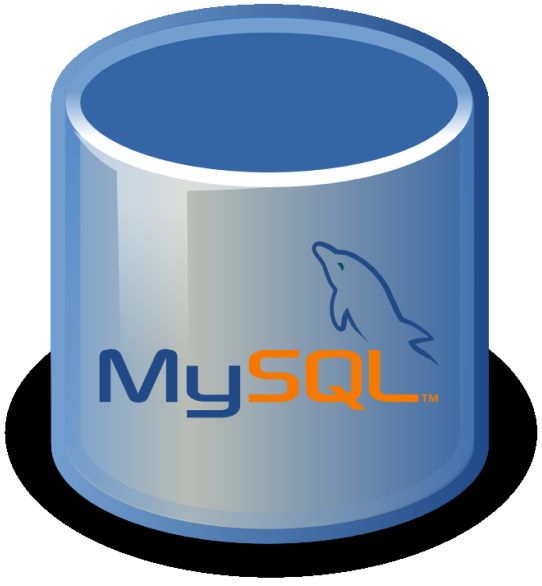
Extend our entity-relationship diagram to encode the statement:

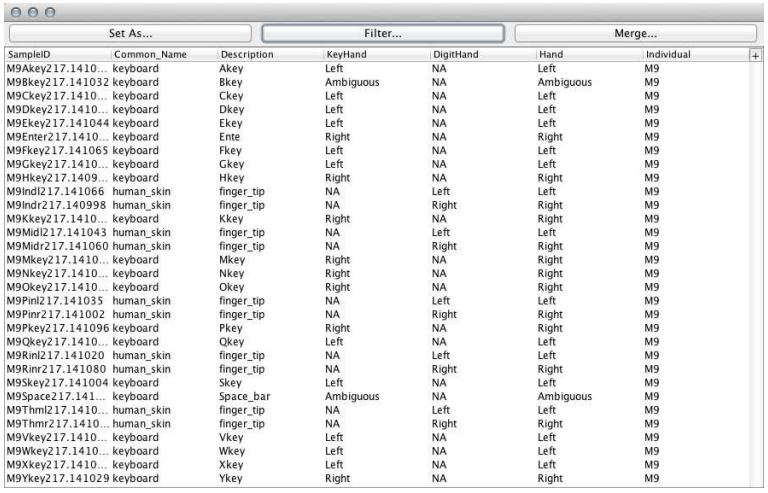
- Each subject has only one instructor, but an instructor can teach many subjects

Section 4

RDBMS and interfaces

Database and RDBMS





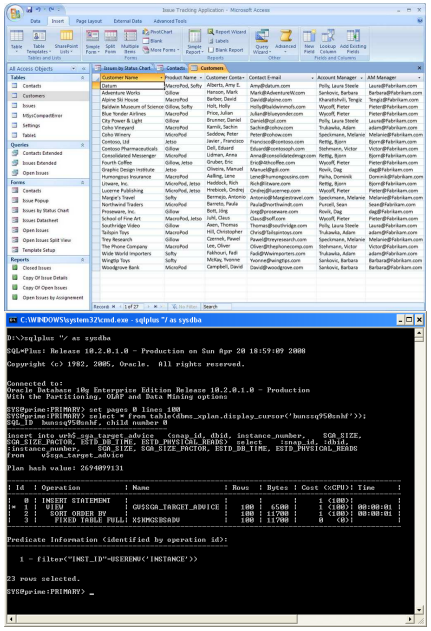
Set As... | Filter... | Merge...

SampleID	Common_Name	Description	KeyHand	DigitHand	Hand	Individual
M9Akey217.1410...	keyboard	Akey	Left	NA	Left	M9
M9Bkey217.141032	keyboard	Bkey	Ambiguous	NA	Ambiguous	M9
M9Ckey217.1410...	keyboard	Ckey	Left	NA	Left	M9
M9Dkey217.1410...	keyboard	Dkey	Left	NA	Left	M9
M9Ekey217.141044	keyboard	Ekey	Left	NA	Left	M9
M9Enter217.1410...	keyboard	Ente	Right	NA	Right	M9
M9Fkey217.141065	keyboard	Fkey	Left	NA	Left	M9
M9Gkey217.1410...	keyboard	Gkey	Left	NA	Left	M9
M9Hkey217.1409...	keyboard	Hkey	Right	NA	Right	M9
M9Indl217.141066	human_skin	finger_tip	NA	Left	Left	M9
M9Indr217.140998	human_skin	finger_tip	NA	Right	Right	M9
M9Kkey217.1410...	keyboard	Kkey	Right	NA	Right	M9
M9Midl217.141043	human_skin	finger_tip	NA	Left	Left	M9
M9Midr217.141060	human_skin	finger_tip	NA	Right	Right	M9
M9Mkey217.1410...	keyboard	Mkey	Right	NA	Right	M9
M9Nkey217.1410...	keyboard	Nkey	Right	NA	Right	M9
M9Okey217.1410...	keyboard	Okey	Right	NA	Right	M9
M9Pinl217.141035	human_skin	finger_tip	NA	Left	Left	M9
M9Pinr217.141002	human_skin	finger_tip	NA	Right	Right	M9
M9Pkey217.141096	keyboard	Pkey	Right	NA	Right	M9
M9Qkey217.1410...	keyboard	Qkey	Left	NA	Left	M9
M9Rinl217.141020	human_skin	finger_tip	NA	Left	Left	M9
M9Rinr217.141080	human_skin	finger_tip	NA	Right	Right	M9
M9Skey217.141004	keyboard	Skey	Left	NA	Left	M9
M9Space217.141...	keyboard	Space_bar	Ambiguous	NA	Ambiguous	M9
M9Thml217.1410...	human_skin	finger_tip	NA	Left	Left	M9
M9Thmr217.1410...	human_skin	finger_tip	NA	Right	Right	M9
M9Vkey217.1410...	keyboard	Vkey	Left	NA	Left	M9
M9Wkey217.1410...	keyboard	Wkey	Left	NA	Left	M9
M9Xkey217.1410...	keyboard	Xkey	Left	NA	Left	M9
M9Ykey217.141029	keyboard	Ykey	Right	NA	Right	M9

- The database is the stored data and the schema that describes it
- Management of and access to the data (along with other services) is provided by the (relational) database management system (RDBMS).

Interfaces to the database

The RDBMS provides several interfaces to the database:



```
C:\WINDOWS\system32\cmd.exe - sqlplus "/ as sysdba

D:\>sqlplus "/ as sysdba

SQL*Plus: Release 10.2.0.1.0 - Production on Sun Apr 20 10:59:09 2008

Copyright (c) 1982, 2005, Oracle.  All rights reserved.


Connected to:
Oracle Database 10g Enterprise Edition Release 10.2.0.1.0 - Production
With the Partitioning, OLAP and Data Mining options

SYS@prime:PRIMARY> set pages 0 lines 100
SYS@prime:PRIMARY> select * from table(dbms_xplan.display_cursor('bunssq95@snhf'));
SQL_ID  bunssq95@snhf, child number 0
-------------------------------------
insert into wri$_sqa_target_advice     (sqag_id, dbid, instance_number,   SQA_SIZE,
SQA_SIZE_FACTOR, ESTD_DB_TIME, ESTD_PHYSICAL_READS) select  :sqag_id, :dbid,
:instance_number,  SQA_SIZE, SQA_SIZE_FACTOR, ESTD_DB_TIME, ESTD_PHYSICAL_READS
from    v$sqa_target_advice

Plan hash value: 2694899131

-------------------------------------------------------------------------------
| Id  | Operation          | Name                | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | INSERT STATEMENT   |                     |       |       |     1 (100)|          |
|*  1 |  VIEW              | QU$SQA_TARGET_ADVICE |   100 |  6500 |     1 (100)| 00:00:01 |
|   2 |   SORT ORDER BY    |                     |   100 | 11700 |     1 (100)| 00:00:01 |
|   3 |    FIXED TABLE FULL| X$KMGSBSADO         |   100 | 11700 |     0   (0)|          |
-------------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------

   1 - filter("INST_ID"=USERENV('INSTANCE'))

23 rows selected.

SYS@prime:PRIMARY>
```

```
// Get a connection to SQL Server 2005
Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
conn = DriverManager.getConnection(connectionUrl);

// Via the connection, mimic user requests by looping through the table
// and retrieving rows 1 at a time up to 20 rows;
// SalesOrderHeader id's start at 43659
for (int id = 43659; id < 43679; id = id + 1) {

    // Get the id and execute the query
    SQL = "SELECT c.FirstName, c.LastName, oh.SalesOrderID, "
        + "oh.OrderDate, oh.DueDate, oh.TotalDue "
        + "FROM Sales.SalesOrderHeader oh "
        + "JOIN Person.Contact c on c.ContactID = oh.ContactID "
        + "WHERE SalesOrderID = " + id;

    stmt = conn.createStatement();

    // Execute the query
    rs = stmt.executeQuery(SQL);
```

Graphical user interface

- Spreadsheet-like views, wizards

CLI, with specialist query language (SQL)

- Powerful search, manipulation
- Requires specialist knowledge

Programming language API

- Wraps SQL in programming constructs
- General interface for application development

RDBMS services

A fully-fledged RDBMS provides a number of other services:

- Allow database connections over network (database, application can run on different computers)
- Allow, manage multiple simultaneous database connections,
- Transaction support (allow applications to “lock” tables or rows) to block or undo conflicting updates

Desktop DBMS frequently do not offer such functionality, and may only offer a GUI interface. Easy to use, but not extensible to full application development.

Section 5

Review

Structured data

Name	Student id	Subject code	Subject name	Mark	Instructor
Jane Doe	1234-5678	LBSC 690	Information Technology	87	Doug Oard
John Dee	2233-4455	LBSC 690	Information Technology	75	Doug Oard
Jane Doe	1234-5678	LBSC 771	Records Management	76	Adam Adamson
John Dee	2233-4455	LBSC 601	Information Retrieval	52	Doug Oard

- Organize information to property : value pairs
- Enforce types, constraints, indexes
- Single-table database: rows are entities, columns attributes
- Level of granularity

Relational data

Field	Type	Properties
Student id	Character	Primary Key
Name	Character	
Mark	Integer	
Subject code	Character	Foreign Key $\rightarrow$ Subject(Code)

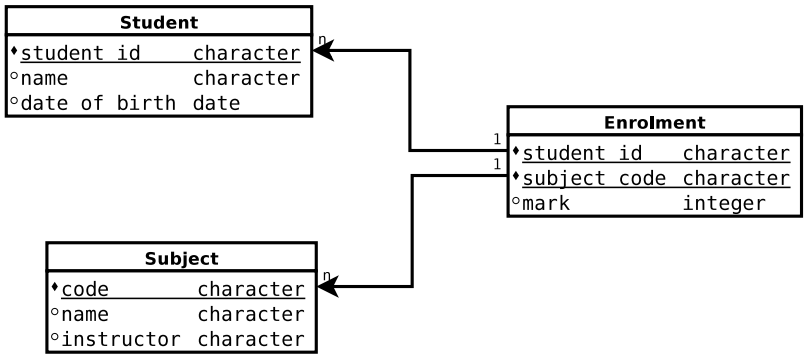
Table: Student

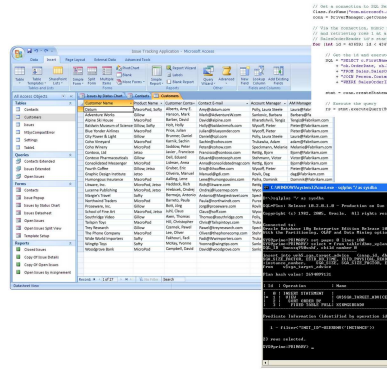
Field	Type	Properties
Code	Character	Primary Key
Name	Character	
Instructor	Character	

Table: Subject

- Decompose compound entities to avoid redundancy
- Use of foreign keys – primary key link to connect records
- Join, project, restrict operations

ER diagrams, database systems





- ER diagrams: graphical representation of schema
- DB systems provide:
  - various interfaces to database (graphical, command-line, programmatic)
  - additional services (concurrent access, integrity maintenance)

Feedback

On a piece of paper, write (without names):

What was the muddiest point in today's class?