

GSCI1801A

Information Science

Lecture 6: Databases

Asst. Prof. Chawanat NAKASAN | 2021-11-09

Agenda

- Databases
- Types of Databases
- Relational Databases
- Keys
- Relational Database Normalization
- Non-Relational Databases
- Database Management System (DBMS)
 - Relational DBMS (MySQL/MariaDB, SQLite, etc.)
 - Non-Relational DBMS (MongoDB, Redis, etc.)

Database

“an organized collection of structured information, or data, typically stored electronically in a computer system” – [Oracle Corporation](#)

“a collection of data that is stored in a computer and that can easily be used and added to” – [Collins Dictionary](#)

collection of data

structured

electronic storage (computer system)

managed, searchable, retrievable

“a usually large collection of data organized especially for rapid search and retrieval (as by a computer)” – [Merriam-Webster Dictionary](#)

“A logical collection of information that is interrelated and that is managed and stored as a unit” -- [OECD](#)

How do we organize data?

Tables

Documents

Index Cards

Ledgers

Paper

Tell stories about who owns each coin?

I kid you not. This was a real data management system.



STONE MONEY OF UAP, WESTERN CAROLINE ISLANDS.

(From the paper by Dr. W. H. Furness, 3rd, in 'Transactions, Department of Archæology, University of Pennsylvania, Vol. I., No. 1, p. 51, Fig. 3, 1904.)

Image via [Wikipedia](#)

Databases are divided into two major categories.



Relational
Database

Based on relations (math word for table)



Non-Relational
Database

Based on other forms of data management: document, hierarchy (tree), network (graph), etc.

Relational Databases

Relation, a.k.a., Table

A relation needs to have a name.

LabMembers

id	firstname	lastname	office
550052	Noble	Tillman	7087
361144	Ella	Joseph	6983
576086	Magee	Wilson	1976
484862	Daniel	Levine	8810
772473	Xandra	Green	3726

Each header *column* here is called an **attribute** or a **field**.

Each *row* here is also called a **tuple** or **record**.

This whole *table* is called a **relation**.

Relation Set Operators: Union

LabMembers

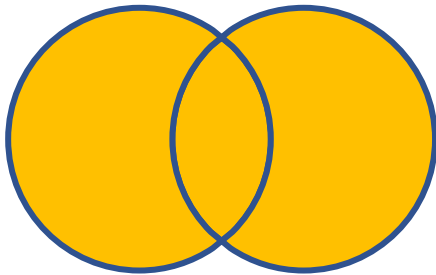
id	firstname	lastname	office
550052	Noble	Tillman	7087
361144	Ella	Joseph	6983
576086	Magee	Wilson	1976
484862	Daniel	Levine	8810
772473	Xandra	Green	3726

U

Committee

id	firstname	lastname	office
484862	Daniel	Levine	8810
772473	Xandra	Green	3726
621923	Nero	Thompson	7891
513722	Ava	Malone	4475
152384	Alea	Downs	8627

LabMembers \cup *Committee* =



id	firstname	lastname	office
550052	Noble	Tillman	7087
361144	Ella	Joseph	6983
576086	Magee	Wilson	1976
484862	Daniel	Levine	8810
772473	Xandra	Green	3726
621923	Nero	Thompson	7891
513722	Ava	Malone	4475
152384	Alea	Downs	8627

Rows from
LabMembers

Rows from
Committee

Relation Set Operators: Intersect

LabMembers

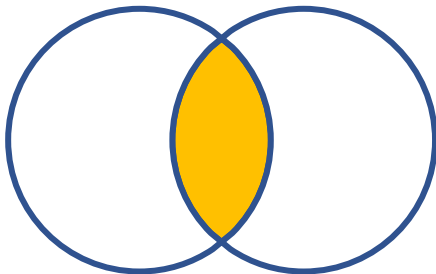
id	firstname	lastname	office
550052	Noble	Tillman	7087
361144	Ella	Joseph	6983
576086	Magee	Wilson	1976
484862	Daniel	Levine	8810
772473	Xandra	Green	3726



Committee

id	firstname	lastname	office
484862	Daniel	Levine	8810
772473	Xandra	Green	3726
621923	Nero	Thompson	7891
513722	Ava	Malone	4475
152384	Alea	Downs	8627

LabMembers \cap *Committee* =



id	firstname	lastname	office
484862	Daniel	Levine	8810
772473	Xandra	Green	3726

Relation Set Operators: Difference

LabMembers

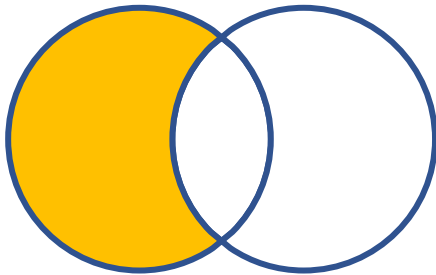
id	firstname	lastname	office
550052	Noble	Tillman	7087
361144	Ella	Joseph	6983
576086	Magee	Wilson	1976
484862	Daniel	Levine	8810
772473	Xandra	Green	3726

Committee

id	firstname	lastname	office
484862	Daniel	Levine	8810
772473	Xandra	Green	3726
621923	Nero	Thompson	7891
513722	Ava	Malone	4475
152384	Alea	Downs	8627

—

LabMembers − *Committee* =



id	firstname	lastname	office
550052	Noble	Tillman	7087
361144	Ella	Joseph	6983
576086	Magee	Wilson	1976

Projection

LabMembers

id	firstname	lastname	office
550052	Noble	Tillman	7087
361144	Ella	Joseph	6983
576086	Magee	Wilson	1976
484862	Daniel	Levine	8810
772473	Xandra	Green	3726

$$\prod_{\text{firstname,lastname}} (\text{LabMembers}) =$$

firstname	lastname
Noble	Tillman
Ella	Joseph
Magee	Wilson
Daniel	Levine
Xandra	Green

Selection

LabMembers

id	firstname	lastname	office
550052	Noble	Tillman	7087
361144	Ella	Joseph	6983
576086	Magee	Wilson	1976
484862	Daniel	Levine	8810
772473	Xandra	Green	3726

$\sigma_{6000 \leq office \leq 7999}(LabMembers) =$

id	firstname	lastname	office
550052	Noble	Tillman	7087
361144	Ella	Joseph	6983

For Kanazawa University Official Textbook users: This expression can also be written as $LabMembers[6000 \leq office \leq 7999]$. The selection operation is also called "restriction".

Either use will be accepted in examination.

These can be combined.

LabMembers

id	firstname	lastname	office
550052	Noble	Tillman	7087
361144	Ella	Joseph	6983
576086	Magee	Wilson	1976
484862	Daniel	Levine	8810
772473	Xandra	Green	3726

Committee

id	firstname	lastname	office
484862	Daniel	Levine	8810
772473	Xandra	Green	3726
621923	Nero	Thompson	7891
513722	Ava	Malone	4475
152384	Alea	Downs	8627

$$\prod_{\text{firstname}} (\text{LabMembers} \cap \text{Committee}) =$$

firstname
Daniel
Xandra

$$\prod_{\text{id,lastname}} \sigma_{\text{id} > 500000} (\text{LabMembers} \cup \text{Committee}) =$$

id	lastname
550052	Tillman
576086	Wilson
772473	Green
513722	Malone

You can also rename the attributes.

$$\prod_{id, lastname} \sigma_{id > 500000} (LabMembers \cup Committee) =$$

id	lastname
550052	Tillman
576086	Wilson
772473	Green
513722	Malone

$$\rho_{name/lastname} \prod_{id, lastname} \sigma_{id > 500000} (LabMembers \cup Committee) =$$



Rename "lastname" to "name"

id	name
550052	Tillman
576086	Wilson
772473	Green
513722	Malone

Here be dragons.

- Everything from this point is **not** covered by the official textbook.
- Don't forget to save the slides in LMS.
- If you have any questions, ask me now, after class, through WebEx, or in LMS.
- You can use any media during exam. (You are not allowed to generate communication or send signals, but you can read online references and read slides/textbooks.)

Join Operations

- Joining puts data from two or more tables together, extending the meaning of the data.

Personnel

firstname	lastname	email
Davis	Burgess	dburgess@nan.sus.xs
Colton	Bullock	cbullock@mat.sus.xs
Colleen	Bowers	cbowers@gra.sus.xs
Alexis	Hampton	ahampton@geo.sus.xs
Rina	Parker	rparker@ast.sus.xs

Gamers

email	character	level
dburgess@nan.sus.xs	Frodo	14
cbullock@mat.sus.xs	Aragorn	8
rparker@ast.sus.xs	Shelob	11
stewart@sarktech.xs	Lancelot	10
bowen3@cortalc.co.xs	Magnus	14

Explanation:

- Personnel is a list of people at Sarkhan University of Science (sus.xs).
- Gamers is a list of players on a game server. In this scenario, there are SUS members and employees from nearby companies SarkTech and Cortano Alchemy.

Inner Join



comparing these



Personnel

firstname	lastname	email
Davis	Burgess	dburgess@nan.sus.xs
Colton	Bullock	cbullock@mat.sus.xs
Colleen	Bowers	cbowers@gra.sus.xs
Alexis	Hampton	ahampton@geo.sus.xs
Rina	Parker	rparker@ast.sus.xs

Gamers

email	character	level
dburgess@nan.sus.xs	Frodo	14
cbullock@mat.sus.xs	Aragorn	8
rparker@ast.sus.xs	Shelob	11
stewart@sarktech.xs	Lancelot	10
bowen3@cortalc.co.xs	Magnus	14

firstname	lastname	personnel.email*	gamers.email*	character	level
Davis	Burgess	dburgess@nan.sus.xs	dburgess@nan.sus.xs	Frodo	14
Colton	Bullock	cbullock@mat.sus.xs	cbullock@mat.sus.xs	Aragorn	8
Rina	Parker	rparker@ast.sus.xs	rparker@ast.sus.xs	Shelob	11

Personnel ⋈_{Personnel.email=Gamers.email} *Gamers*

*tablename.attributename to disambiguate (make non-confusing) same attribute names

There are other kinds of join.

- Natural Join (actually a specific kind of inner join)
 - Left Join
 - Right Join
 - Outer Join
 - Full Join
 - Anti-Join
-
- But most of these are not very important and won't be used in exams. (Most DB applications support only natural join.)

Summary of Operators

- Set operators: \cup \cap –
- Selection (Restriction): σ (sigma)
- Projection: Π (pi)
- Rename: ρ (rho)
- Join operator: \bowtie

SQL (Structured Query Language)

- SQL is a database query language.
- SQL is based directly on relational algebra.
- Will be demonstrated later in this class.

Keys

Keys allow you to identify data.

- What uniquely identifies an employee?

id	firstname	lastname	office	phone	affiliation	email	home_zipcode
550052	Noble	Tillman	7087	075-610-5211	School of Astronomy	ntillman@ast.sus.xs	99769
361144	Ella	Joseph	6983	051-493-8214	School of Physics	ejoseph@phy.sus.xs	99769
629586	Blaze	Clark	8810	011-657-3256	Administration	bclark@adm.sus.xs	96943
484862	Daniel	Levine	8810	091-589-8389	Administration	dlevine@adm.sus.xs	33732
772473	Xandra	Green	3726	Null	School of Astronomy	xgreen@ast.sus.xs	17241
241566	Wyatt	Goodman	4721	059-867-5427	School of Geology	wgoodman@geo.sus.xs	42865
181555	Wyatt	Welch	4051	052-534-4395	Administration	wwelch@adm.sus.xs	24436

Same Name

Same Room

No Phone

Neighbors – Same Zipcode

Super Key

- A super key is a set of attributes that sufficiently identify unique records.
- Example:
 - firstname, office, and home_zipcode are NG because there are repeats.
 - phone is also NG because Xandra doesn't seem to have a phone.
 - id is OK because it's unique for each employee.
 - (firstname, lastname) is OK because there are no repeats when both firstname and lastname are combined.

id	firstname	lastname	office	phone	affiliation	email	home_zipcode
550052	Noble	Tillman	7087	075-610-5211	School of Astronomy	ntillman@ast.sus.xs	99769
361144	Ella	Joseph	6983	051-493-8214	School of Physics	ejoseph@phy.sus.xs	99769
629586	Blaze	Clark	8810	011-657-3256	Administration	bclark@adm.sus.xs	96943
484862	Daniel	Levine	8810	091-589-8389	Administration	dlevine@adm.sus.xs	33732
772473	Xandra	Green	3726	Null	School of Astronomy	xgreen@ast.sus.xs	17241
241566	Wyatt	Goodman	4721	059-867-5427	School of Geology	wgoodman@geo.sus.xs	42865
181555	Wyatt	Welch	4051	052-534-4395	Administration	wwelch@adm.sus.xs	24436

Candidate Key

- Good super key, but a bit overkill: (id, firstname, lastname, office).
- We can *reduce* it to:
 - (id)
 - (firstname, lastname)

Valid super key, but is it too much?

id	firstname	lastname	office	phone	affiliation	email	home_zipcode
550052	Noble	Tillman	7087	075-610-5211	School of Astronomy	ntillman@ast.sus.xs	99769
361144	Ella	Joseph	6983	051-493-8214	School of Physics	ejoseph@phy.sus.xs	99769
629586	Blaze	Clark	8810	011-657-3256	Administration	bclark@adm.sus.xs	96943
484862	Daniel	Levine	8810	091-589-8389	Administration	dlevine@adm.sus.xs	33732
772473	Xandra	Green	3726	Null	School of Astronomy	xgreen@ast.sus.xs	17241
241566	Wyatt	Goodman	4721	059-867-5427	School of Geology	wgoodman@geo.sus.xs	42865
181555	Wyatt	Welch	4051	052-534-4395	Administration	wwelch@adm.sus.xs	24436


(id), (firstname, lastname), (email) are some of the good candidate keys!

By the way, (firstname, lastname) is a *composite key* because it has multiple attributes.

Primary Key (PK)

- Among the many candidate keys, we now choose one logically best to become the *primary key (PK)*.
- There is only one PK per table (relation).

These are all good candidate keys, but which would be the PK?



id	firstname	lastname	office	phone	affiliation	email	home_zipcode
550052	Noble	Tillman	7087	075-610-5211	School of Astronomy	ntillman@ast.sus.xs	99769
361144	Ella	Joseph	6983	051-493-8214	School of Physics	ejoseph@phy.sus.xs	99769
629586	Blaze	Clark	8810	011-657-3256	Administration	bclark@adm.sus.xs	96943
484862	Daniel	Levine	8810	091-589-8389	Administration	dlevine@adm.sus.xs	33732
772473	Xandra	Green	3726	Null	School of Astronomy	xgreen@ast.sus.xs	17241
241566	Wyatt	Goodman	4721	059-867-5427	School of Geology	wgoodman@geo.sus.xs	42865
181555	Wyatt	Welch	4051	052-534-4395	Administration	wwelch@adm.sus.xs	24436

Foreign Key (FK)

- Foreign keys allow one table to reference another and keep data linked together.

Personnel

id	firstname	lastname
550052	Noble	Tillman
361144	Ella	Joseph
629586	Blaze	Clark
484862	Daniel	Levine
772473	Xandra	Green

Beacon

id	location
15	Administration
42	Cafeteria
49	Library
61	Lecture Hall 1 (1F)
74	Lecture Hall 2 (4F)

Checkin

datetime	beacon_id	user_id
2021-11-08 09:04:02	15	629586
2021-11-08 09:18:40	15	361144
2021-11-08 10:02:00	49	629586
2021-11-09 08:13:17	61	772473
2021-11-09 08:59:59	61	550052

FOREIGN KEY (beacon_id) REFERENCES beacon(id)
Meaning: every value in checkin.beacon_id must exist in beacon.id

FOREIGN KEY (user_id) REFERENCES user(id)
Meaning: every value in checkin.user_id must exist in user.id

What happens when we change data?

- This is called “propagation”.
- There are five kinds of propagation:
 - CASCADE: When changes happen in reference table, also update it in dependent table.
 - RESTRICT: Don’t allow changes at all.
 - NO ACTION: Similar to RESTRICT (changes will be apparent when you learn triggers system, but we won’t discuss it here)
 - SET NULL and SET DEFAULT: When changes happen in the reference table, set the corresponding value in the dependent table to *Null* or the default value.
- Propagation can be set differently, like “ON UPDATE CASCADE ON DELETE SET NULL”.

Relational Database Normalization

Here be dragons (again).

- **Normalization is a rather difficult topic.**
- It is extremely important for proper development of databases, so I'm going to teach it anyway.
- But, this (just this topic) **won't be graded in exam**. As stated in the syllabus:
 - Solve relational algebra operations (union, join, and so on).
- There are many normal forms, but today I'll only teach 1NF and 3NF, which are major steps of practical database design.

Why Normalize?

- Because this is not a good idea:

Lecture

class	lecturer_id	room	day	period
CS101.1	186346 894324 245350	LH1-102	Thu	1
CS101.2	186346 894324 245350	LH1-102	Thu	2

- In this scenario, there are classrooms taught by three professors. However, you can see that all the professor IDs are crammed into a single box.

1NF: First Normal Form

- Each “cell” must have only one value.
- Each record must be unique.

Lecture

class	lecturer_id	room	day	period
CS101.1	186346	LH1-102	Thu	1
	894324			
	245350			
CS101.2	186346	LH1-102	Thu	2
	894324			
	245350			



Lecture (1NF)

class	lecturer_id	room	day	period
CS101.1	186346	LH1-102	Thu	1
CS101.1	894324	LH1-102	Thu	1
CS101.1	245350	LH1-102	Thu	1
CS101.2	186346	LH1-102	Thu	2
CS101.2	894324	LH1-102	Thu	2
CS101.2	245350	LH1-102	Thu	2

Unnormalized Form

First Normal Form (1NF)

3NF: Third Normal Form

- (room, day, period) defines each instance of lecture.
- This means class depends on (room, day, period).
- However, since one lecturer can be only in one class at a time, lecturer_id depends on class.
- This creates a transitive dependency:
 - lecturer_id → class → (room, day, period).

Lecture (1NF)

class	lecturer_id	room	day	period
CS101.1	186346	LH1-102	Thu	1
CS101.1	894324	LH1-102	Thu	1
CS101.1	245350	LH1-102	Thu	1
CS101.2	186346	LH1-102	Thu	2
CS101.2	894324	LH1-102	Thu	2
CS101.2	245350	LH1-102	Thu	2

First Normal Form (1NF)

LectureTeam (3NF)

lecturer	class
186346	CS101.1
894324	CS101.1
245350	CS101.1
186346	CS101.2
894324	CS101.2
245350	CS101.2

Classroom (3NF)

room	day	period	class
LH1-102	Thu	1	CS101.1
LH1-102	Thu	2	CS101.2

3NF

What about other NFs?

- 2NF depends on having composite PKs. Since most modern DB productions tend to use singular PKs, it is usually avoided.
- Higher NFs contribute little to compared to 3NF.

Non-Relational Databases

Also known as NoSQL, these are just types of databases that don't rely on strict relational tables.

Key	Document
1001	{ "CustomerID": 99, "OrderItems": [{ "ProductID": 2010, "Quantity": 2, "Cost": 520 }, { "ProductID": 4365, "Quantity": 1, "Cost": 18 }], "OrderDate": "04/01/2017" }
1002	{ "CustomerID": 220, "OrderItems": [{ "ProductID": 1285, "Quantity": 1, "Cost": 120 }], "OrderDate": "05/08/2017" }

KEY1	→	Value1
KEY2	→	Value2
KEY3	→	Value3
KEY4	→	Value4
KEY5	→	Value1
KEY6	→	Value3

(a)

Key	Value
user 1: employee	{65,865,9634}
user2: employee	{34,85,76,94}
user3: employee	{desg:manager, branchcode: 345}
user4: employee	{487,236}
user5: employee	{78,456,35}
user6: employee	{ name: mark, empid:346}

(b)

Key-Value Store

(figure by [Ramzan et al., 2019](#))

Document-based database
([Microsoft Azure](#))

Database Management Systems (DBMS)

DBMS is a program implementation to build and manage databases.

- Relational:
 - SQLite
 - MySQL
 - MariaDB (open-source version of MySQL)
 - PostgreSQL
- Non-Relational:
 - MongoDB
 - Redis
- Today I'll walk through SQLite and maybe MySQL. You can learn other DBMSes as needed.