Normal Forms

Databases

Ira Assent

ira@cs.au.dk

Data-Intensive Systems group, Department of Computer Science, Aarhus University, DK

Intended learning outcomes

- Be able to
  - Identify multivalued dependencies
  - Determine BCNF, 4NF
  - Apply decomposition algorithms

Recap: FDs and NFs

- Converting a bad design to a good design
  - Decompose large relation schemas into smaller ones
  - Ensuring nonadditive join decomposition
  - And ideally dependency preservation
- Functional dependencies to formally analyze issues
- Decompose into normal forms which describe the properties that are fulfilled

Only attribute values permitted are single **atomic** (or **indivisible**) values 1NF

Every non-prime attribute is fully functional dependent on candidate key 2NF

For every nontrivial FD X → A, X superkey or A prime attribute 3NF

Discussion from last time continued

R (A,B,C,D) with FDs A,B → C,D; C → D

3NF?

For every nontrivial FD X → A, X superkey or A prime attribute 3NF

1. Yes.
2. No, a non-superkey determines another attribute.
3. No, a FD determines a non-prime attribute.
4. I don't know.

Minimal Cover of a Set of Dependencies

- Several sets of dependencies may have the same closure
  - Which one to choose as starting point?
  - Go for a minimal set with the same closure: **minimal cover**
- A minimal cover G for set of FDs F is a set of functional dependencies that
  - is **equivalent** to F
    - i.e., its closure is the same: $G^+=F^+$
  - all dependencies are in **canonical form**
    - i.e., have a singleton right side, of the form $a_1,\ldots,a_n \rightarrow b$
  - is **minimal**
    - i.e., if any dependency in G is removed, *G* is no longer a cover
  - Has only **full functional dependencies**
    - i.e., if we remove any a_i from a dependency, *G* is no longer a cover

Obtaining a minimal cover

- Start from a set of functional dependencies F

1. Split right hand sides to obtain canonical form
2. Remove non-necessary attributes on left hand sides to obtain minimal FDs
3. Remove any FD that is not needed to maintain cover property

- Example: F = $\{A \rightarrow C, AC \rightarrow D, E \rightarrow B, E \rightarrow AD\}$

  1. Split $\{E \rightarrow AD\}$: F= $\{A \rightarrow C,\ AC \rightarrow D, E \rightarrow B, E \rightarrow A, E \rightarrow D\}$
  2. $\{A\}^+ = \{A, C\}$, so we can remove C from $\{AC \rightarrow D\}$ to $\{A \rightarrow D\}$: F= $\{A \rightarrow C, A \rightarrow D, E \rightarrow B, E \rightarrow A, E \rightarrow D\}$
  3. $\{E \rightarrow D\}$ redundant because: $\{E \rightarrow A,\ A \rightarrow D\}$: F= $\{A \rightarrow C, A \rightarrow D, E \rightarrow B, E \rightarrow A\}$

  Minimal Cover = $\{A \rightarrow C, A \rightarrow D, E \rightarrow B, E \rightarrow A\}$

Minimal cover?

Is G ={B→C, C→A} a minimal cover of F={B→A, C→A, AB→C}?

- Yes.
- No, not minimal.
- No, not equivalent.
- No, not in canonical form.

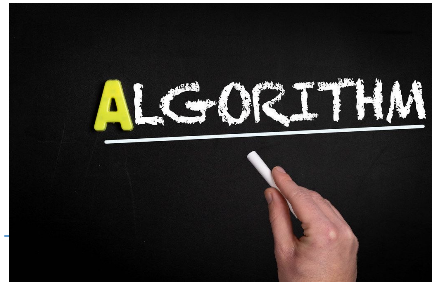
Algorithm for Relational Synthesis into 3NF

For every nontrivial FD X → A, X superkey or A prime attribute 3NF

Given relation R, functional dependencies F

1. Find a minimal cover G for F
2. For each left-hand-side X of a functional dependency that appears in G,
   - Create relation R_1 with attributes $\{X \cup \{A_1\} \cup \{A_2\} \dots \cup \{A_k\}\}$,
     where $X \rightarrow A_1, X \rightarrow A_2, \dots, X \rightarrow A_k$ are the only dependencies in G with X as left-hand-side
     (X is the key of this relation)
   - Place remaining attributes in a relation R_2
3. If none of the relation schemas contains a key of R, then create one more relation schema that contains attributes that form a key of R
4. Elliminate any redundant relation, i.e., a relation that is a projection to a proper subset of the attributes of another relation

- Every relation schema created by this Relational Synthesis Algorithm is in 3NF



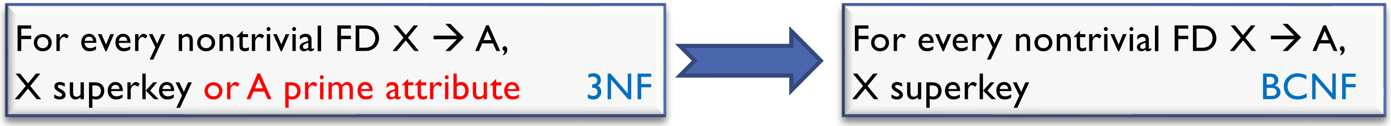
3NF synthesis example

- R(A, B, C, D, E, F), FDs: {A → D, B → C, B → D, D → E}
  - Minimal cover: yes, already canonical, minimal
  - We have 3 left hand sides A, B, D: create a relation for each with their dependencies R_1(A, D), R_2(B, C, D), R_3(D, E)
  - Does any of these relations contain a key of R? No, so add a fourth relation with a minimal key of R: R_1(A, D), R_2(B, C, D,) R_3(D, E), R_4(A, B, F)
  - No redundant relation, done

  - Result not uniquely defined, because there may be more than one minimal cover
    - So, if you get a different 3NF decomposition result than your neighbor, it is not necessarily because anything is wrong (but you might want to double check 😉)

Asking for more normalization than 3NF

- Tables should have only non-trivial functional dependencies where the left side contains a key



For every nontrivial FD X → A,
X superkey or A prime attribute 3NF

For every nontrivial FD X → A,
X superkey BCNF

  - "simplification of 3NF requirements" → dropping an allowed FD in 3NF, means a stronger requirement in BCNF, so BCNF is actually stricter!

- **Boyce-Codd Normal Form (BCNF)**:

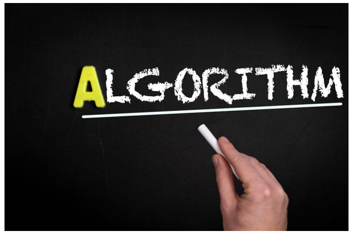
  For all non-trivial dependencies $a_1,\ldots,a_n \to b$,
  $a_1,\ldots,a_n$ is a superkey. BCNF

Algorithm for Relational Decomposition into BCNF

For every nontrivial FD X → A, X superkey BCNF

Given relation R, functional dependencies F

1. find a functional dependency X → Y in R that violates BCNF (else done)
2. replace R with relation R_1 with attributes (R - Y) and relation R_2 with attributes (X ∪ Y)
3. Repeat until all relations are in BCNF



- Assumption: No `NULL` values are allowed for the join attributes
  - In general: `NULL` values are problematic for decomposition, no good normalization theory exists
  - In practice: watch out for the possible existence of `NULL` values for attributes that are part of referential integrity constraints

BCNF Example

For every nontrivial FD X → A,
X superkey BCNF

- Table exams has keys: (studid, date), (vip, date, time), (room, date, time)
- And FDs studid, date → time
  studid, date → vip
  room, date, time → vip
  vip, date → room
  studid, date → room
  vip, date, time → studid
  room, date, time → studid

Nontrivial, left side not superkey

- Decompose using violating FD, intersection between two tables is superkey for first table

studid	date	time	vip	room
01	2014-10-15	09:00	ira	Turing-230
02	2014-10-15	09:30	ira	Turing-230
01	2014-10-16	12:30	amoeller	Turing-230
03	2014-10-16	10:30		
	2014-10-17	11:00		

date	vip	room
2014-10-15	ira	Turing-230
2014-10-16	amoeller	Turing-230
2014-10-16	bodker	Ada-017
2014-10-17	bodker	Ada-017

studid	date	time	vip
01	2014-10-15	09:00	ira
02	2014-10-15	09:30	ira
01	2014-10-16	12:30	amoeller
03	2014-10-16	10:30	bodker
01	2014-10-17	11:00	bodker

BCNF

- Consider a relation R with attributes A,B,C,D, and key (A,D) Which of the FDs ACD → B, D → C (if any) violate BCNF?

For every nontrivial FD X → A,
X superkey BCNF

1. None, R is in BCNF
2. ACD → B
3. D → C
4. Both
5. I don't know

Impact of Non-Dependency Preservation

- BCNF decomposition is not always dependency preserving
  - Reconsider the example from last time
    - R = Concerts(location, city, artist)
    - FDs: location → city; artist, city → location
    - Candidate keys: artist, city; artist, location
    - R is not in BCNF, because of FD location→ city
    - Any decomposition of R will fail to preserve artist, city → location
    - If decomposed, need to maintain dependency manually, which is difficult and error-prone

For every nontrivial FD X → A, X superkey BCNF

location	city	artist
VoxHall	Aarhus	Illdisposed
Jakobshof	Aachen	Ina Deter

3NF allows some redundancy

- But Concerts (location, city, artist) with FDs: location → city; artist, city → location is in 3NF
  - location→ city: city is contained in a candidate key
  - Artist, city → location: artist, city is a candidate key

location	city	artist
VoxHall	Aarhus	Illdisposed
Jakobshof	Aachen	Ina Deter
VoxHall	?	Kurve

- ? = Aarhus by location → city
- Thus, some redundancy may exist
- This is the trade-off between BCNF (stricter normalization / less redundancy) and 3NF (less strictly normalized, preserves dependencies)

For every nontrivial FD X → A, X superkey or A prime attribute 3NF

For every nontrivial FD X → A, X superkey BCNF

Example decomposition into BCNF

- R(A,B,C), F={AB→ C, C→ B}
  - 3NF, not BCNF
  - Violating C→B
    - Decompose into $R_1(C,B)$, $R_2(C,A)$
    - Now in BCNF

For every nontrivial FD X → A,
X superkey or A prime attribute 3NF

For every nontrivial FD X → A,
X superkey BCNF

- Lossless? Yes, always
- Dependency preserving? Not necessarily, need to check. In this case not:
  - $F^+=\{AB\rightarrow C, C\rightarrow B\}$ $F_1 = \pi(R_1(F^+))=\{C\rightarrow B\}$, $F_2 = \pi(R_2(F^+))=\{\}$, $(F_1 \cup F_2)^+ = \{C\rightarrow B\} \neq F^+$

- Result not uniquely defined, because you may start from different violating FDs
  - So, if you get a different BCNF decomposition result than your neighbor, it is not necessarily because anything is wrong (but you might want to double check 😉)

What is true for this table?

1. Redundancy due to a FD which should be resolved by decomposition
2. Redundancy not covered by a FD
3. There is conflicting information due to poor design
4. Everything is just fine, no decomposition necessary

Staff

course	TA	teacher
dDB	aas	ira
dID	aas	ira
dDB	aas	schester
dDB	fra	ira
dID	zoffe	ira
dDB	zoffe	ira
dDB	fra	schester
dDB	zoffe	schester

Multivalued Dependencies

- Independent pieces of information in the same relation
- As for functional dependencies, this is known from real world, not inferred from the schema! ⚠
- There is no functional dependency here
  - Course does not imply TA or teacher
  - TA does not imply course or teacher
  - Teacher does not imply course or TA
- But
  - Course implies a set of teachers and a set of TAs
  - TA implies a set of courses and a set of teachers
  - Teacher implies a set of courses and a set of TAs
- teacher $\rightarrow \rightarrow$ course
  - A teacher implies a set of courses, but not necessarily a single course value
  - Generalizes functional dependency

Staff

course	TA	teacher
dDB	aas	ira
dID	aas	ira
dDB	aas	schester
dDB	fra	ira
dID	zoffe	ira
dDB	zoffe	ira
dDB	fra	schester
dDB	zoffe	schester

Multivalued Dependencies (MVD)

- A **multivalued dependency** (**MVD**) $X \longrightarrow\!\!> Y$ specified on relation schema R, where X and Y are both subsets of R, specifies the following constraint on any relation state r of R: If two tuples t_1 and t_2 exist in r such that $t_1[X] = t_2[X]$, then two tuples t_3 and t_4 should also exist in r with the following properties, where we use Z to denote $(R - (X \cup Y))$:
  - $t_3[X] = t_4[X] = t_1[X] = t_2[X]$.
  - $t_3[Y] = t_1[Y]$ and $t_4[Y] = t_2[Y]$.
  - $t_3[Z] = t_2[Z]$ and $t_4[Z] = t_1[Z]$.
- An MVD $X \longrightarrow\!\!> Y$ in R is called a **trivial MVD** if (a) Y is a subset of X, or (b) $X \cup Y = R$.

Textbook example MVD

EMP has MVDs: ENAME —>> PNAME and ENAME —>> DNAME

- X —>> Y exist $t_1[X] = t_2[X]$, then t_3 t_4 exist, Z denotes (R - ($X \cup Y$)):
  - $t_3[X] = t_4[X] = t_1[X] = t_2[X]$.
  - $t_3[Y] = t_1[Y]$ and $t_4[Y] = t_2[Y]$.
  - $t_3[Z] = t_2[Z]$ and $t_4[Z] = t_1[Z]$

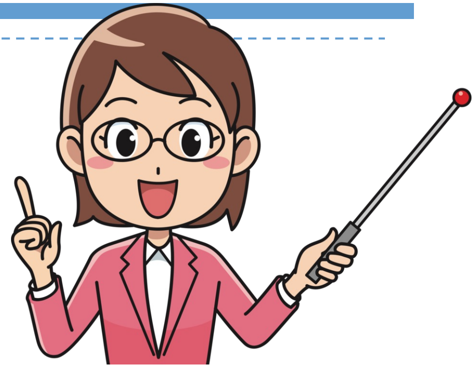
EMP

ENAME	PNAME	DNAME
Smith	X	John
Smith	Y	Anna
Smith	X	Anna
Smith	Y	John

- ENAME→→PNAME:
  - Rows 1-4 agree on X=ENAME=Smith
  - Let Y=PNAME with values X,Y
  - (R - ($X \cup Y$))=Z=DNAME with values John, Anna

  - All tuples agree on X
  - Row 1, row 3 agree on Y, so do row 2, row 4
  - Row 1, row 4 agree on Z, so do row 2, row 3

Multivalued Dependencies for the teacher example



- $A_1A_2\ldots A_n \rightarrow\rightarrow B_1B_2\ldots B_m$
  for each pair of tuples *t* and *u* of R that agree on the A's, we can find in R some tuple *v* that agrees:
  - With both t and u on the A's
  - With t on the B's
  - With u on the attributes that are not A's or B's
  - "Tuple generating": all combinations must be there
  - A generalization of the notion of FD's
- course → → teacher
  - A course implies a set of teachers, but not necessarily a single teacher
    - *t* = (dDB, aas, ira)
    - *u* = (dDB, fra, schester)
    - *v* = (dDB, fra, ira)
    - *v* agrees with *t* and *u* on course
    - *v* agrees with *t* on teacher
    - *v* agrees with *u* on TA
  - So, no contradiction with MVD – remember these are a property of the miniworld! ⚠

course	**TA**	**teacher**
dDB	aas	ira
dID	aas	ira
dDB	aas	schester
dDB	fra	ira
dID	zoffe	ira
dDB	zoffe	ira
dDB	fra	schester
dDB	zoffe	schester

FDs and MVDs compared

- Any functional dependency is a multivalued dependency
  - A → B means attribute values in A imply specific values for attributes in B
  - Then we also have A → → B where attribute values in A imply set of values for attributes in B (the set consisting of a single element)
  - course → teacher
  - Then also course → → teacher

course	TA	teacher
dDB	aas	ira
dID	aas	ira
dDB	fra	ira
dID	zoffe	ira
dDB	zoffe	ira

Inference Rules for FDs and MVDs

IR1 (**reflexive rule for FDs**): If $X \supseteq Y$, then $X \rightarrow Y$.

IR2 (**augmentation rule for FDs**): $\{X \rightarrow Y\} \models XZ \rightarrow YZ$.

IR3 (**transitive rule for FDs**): $\{X \rightarrow Y, Y \rightarrow Z\} \models X \rightarrow Z$.

IR4 (**complementation rule for MVDs**): $\{X \twoheadrightarrow Y\} \models X \twoheadrightarrow (R - (X \cup Y))\}$.

IR5 (**augmentation rule for MVDs**): If $X \twoheadrightarrow Y$ and $W \supseteq Z$ then $WX \twoheadrightarrow YZ$.

IR6 (**transitive rule for MVDs**): $\{X \twoheadrightarrow Y, Y \twoheadrightarrow Z\} \models X \twoheadrightarrow (Z\ 2\ Y)$.

IR7 (**replication rule for FD to MVD**): $\{X \rightarrow Y\} \models X \twoheadrightarrow Y$.

IR8 (**coalescence rule for FDs and MVDs**): If $X \twoheadrightarrow Y$ and there exists W with the properties that

(a) $W \cap Y$ is empty, (b) $W \rightarrow Z$, and (c) $Y \supseteq Z$, then $X \rightarrow Z$.

What is correct? The table has

1. No dependencies with beer on the left side
2. beer → → drinker
3. beer → drinker
4. beer → → drinker and beer → drinker

beer	drinker	size
Beck’s	ira	.5
Jever	tim	.5
Jever	tim	.3
Beck’s	eric	.5
Beck’s	ira	.3

Multivalued dependencies summarized

- Generalization of notion of FD's:

  $$A_1A_2\ldots A_n \rightarrow\rightarrow B_1B_2\ldots B_m$$

- Meaning:

  for each pair of tuples *t* and *u* of R that agree on the A's, we can find in R some tuple *v* that agrees:

  - With both *t* and *u* on the A's
  - With *t* on the B's
  - With *u* on the attributes that are not A's or B's

- Typical for set-valued attributes
  - E.g. teacher teaches several courses, actor stars in several movies
  - Means that two independent one-many relationships are mixed in one relation e.g. A-B, and A-C in R(A,B,C)

Fourth Normal Form (4NF)

- R is in Fourth Normal Form if whenever

  $A_1A_2\ldots A_n \rightarrow \rightarrow\ B_1B_2\ldots B_m$

  is a nontrivial MVD, $\{A_1A_2\ldots A_n\}$ is a superkey.

- R = Film(title, artist, copy), with no non-trivial FDs
  - R is in BCNF
  - Insertion anomalies exist, assume a new copy, 4, is made, need to insert
    - (True Lies, Arnold, 4)
    - (True Lies, Jamie, 4)
  - It is better to decompose Film into two relations
    - These two relations are in 4NF
- Normalize with respect to multi-valued dependencies, not functional dependencies

title	artist	copy
True Lies	Arnold	1
True Lies	Arnold	2
True Lies	Arnold	3
True Lies	Jamie	1
True Lies	Jamie	2
True Lies	Jamie	3

title	artist
True Lies	Arnold
True Lies	Jamie

title	copy
True Lies	1
True Lies	2
True Lies	3

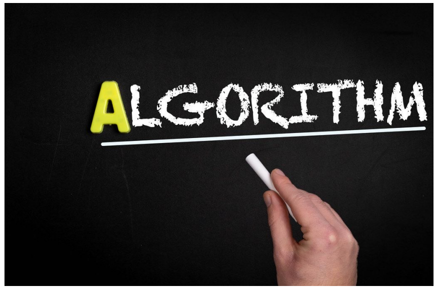
Algorithm for Relational Decomposition into 4NF

For every nontrivial MVD X →→ A,
X superkey 4NF

1. Given relation R, functional and multivalued dependencies F
2. If R not in 4NF (else done)

   find a nontrivial MVD $X \longrightarrow\!\!> Y$ that violates 4NF

   replace R by two relation schemas $(R - Y)$ and $(X \cup Y)$
3. Repeat for R_1 and R_2



Highly similar to what we have done for FD-based NFs, now working with MVDs instead, but same general procedure

4NF decomposition for the teacher example

1. Given relation R, functional and multivalued dependencies F
2. If R not in 4NF (else done)

   find a nontrivial MVD $X \rightarrow\!\!>\!> Y$ that violates 4NF;

   replace R by two relation schemas $(R - Y)$ and $(X \cup Y)$
3. Repeat for R_1 and R_2

R(course, TA, teacher)

course → → teacher

New tables with (Course, Teacher); (Course, TA)

Each TA is only noted once per course

Each teacher is only noted once per course

course	TA	teacher
dDB	aas	ira
dID	aas	ira
dDB	aas	schester
dDB	fra	ira
dID	zoffe	ira
dDB	zoffe	ira
dDB	fra	schester
dDB	zoffe	schester

course	TA
dDB	aas
dID	aas
dDB	fra
dID	zoffe
dDB	zoffe

course	teacher
dDB	ira
dID	ira
dDB	schester

4NF?

R(A,B,C,D), D→C, AB→D, B→→A

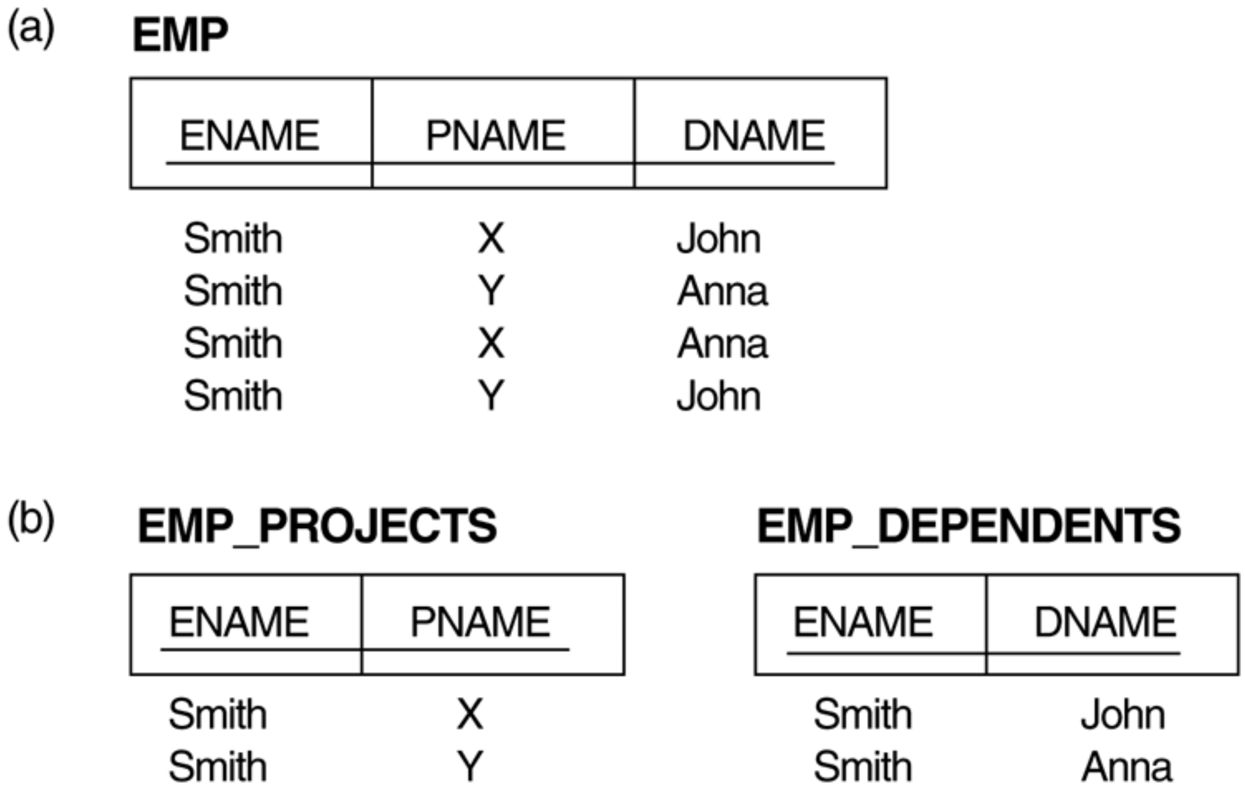
1. Yes, in 4NF.
2. No, we have a violating MVD.
3. No, we have a violating FD.
4. I don't know.

For every nontrivial MVD X →→ A,
X superkey 4NF

Textbook example MVD and 4NF

For every nontrivial MVD X →→ A, X superkey 4NF

(a) EMP has MVDs: ENAME —>> PNAME and ENAME —>> DNAME
(b) Decomposing EMP into 4NF relations EMP_PROJECTS, EMP_DEPENDENTS



(a) **EMP**

ENAME	PNAME	DNAME
Smith	X	John
Smith	Y	Anna
Smith	X	Anna
Smith	Y	John

(b) **EMP_PROJECTS**

ENAME	PNAME
Smith	X
Smith	Y

EMP_DEPENDENTS

ENAME	DNAME
Smith	John
Smith	Anna

Normalization

1. Speeds up SELECT, INSERT, UPDATE, DELETE
2. Slows down SELECT, INSERT, UPDATE, DELETE
3. Speeds up SELECT, but slows down INSERT, UPDATE, DELETE
4. Slows down SELECT, but speeds up INSERT, UPDATE, DELETE
5. I don’t know

Practical Use of Normal Forms

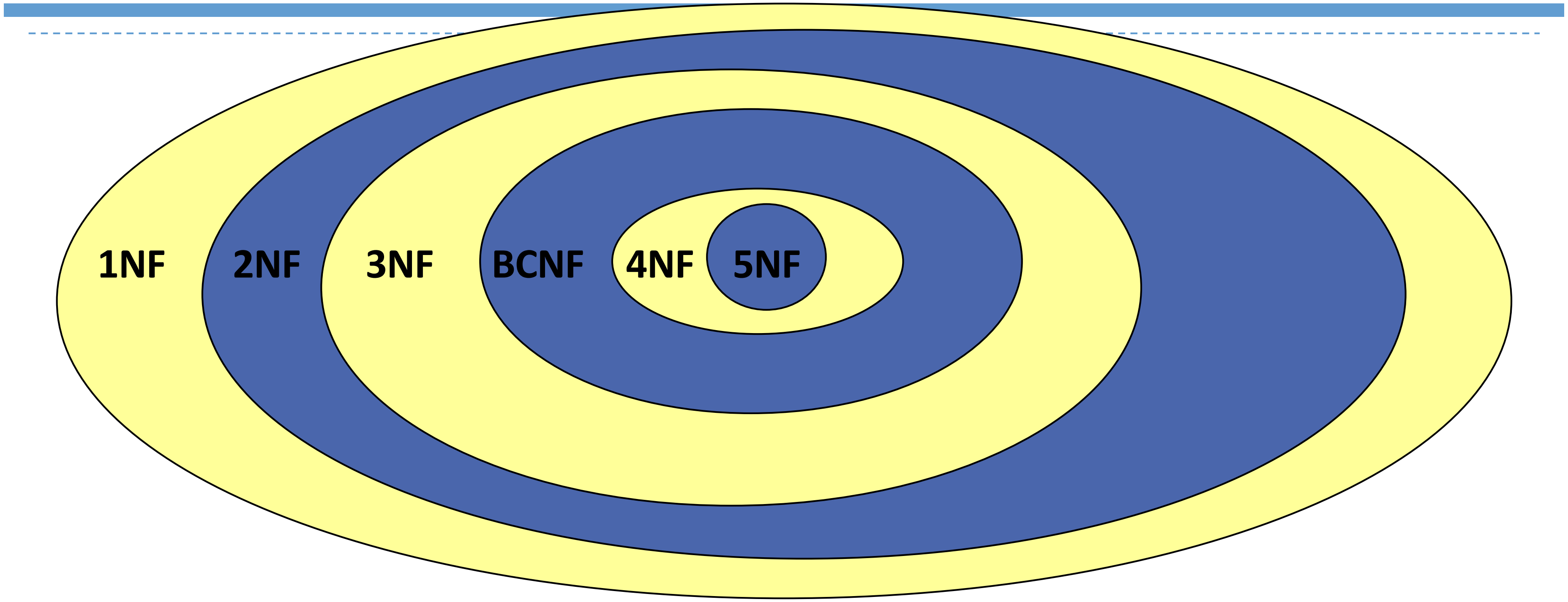
- Redundancy may be good sometimes
  - speeds up data access
  - requires fewer join operations
  - reduces wait time during locking

- Normalization carried out in practice
  - Resulting designs are of high quality and meet the desirable properties stated previously
  - Pays particular attention to normalization only up to 3NF, BCNF, or at most 4NF
- Do not need to normalize to the highest possible normal form
- Denormalization is the careful introduction of redundancy in a database design
  - Formally: store join of higher normal form relations as base relation, which is in a lower normal form
- In practice, also take the workload into account, i.e., the queries and their efficiency requirements

Normal forms

1. If R is in BCNF, it is also in 3NF and 4NF
2. If R is in 4NF, it is also in 3NF and BCNF
3. If R is in 3NF, it is also in 4NF and BCNF
4. If R is in BCNF, it cannot be in 3NF and 4NF
5. I don't know

Normal Form Relationships



- Higher normal form (to the right)

  ⇒ less redundancy

  ”⇒” less memory used

For the purpose of this course, we do not consider 1NF (part of today’s relational model) nor 5NF (seldomly considered in practice)

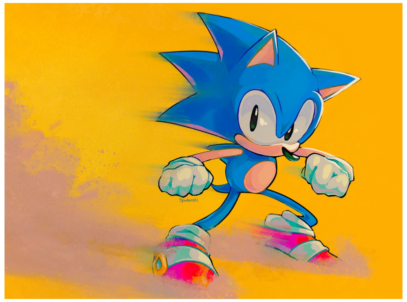
Summary

- Intended learning outcomes
- Be able to
  - Identify multivalued dependencies
  - Determine BCNF, 4NF
  - Apply decomposition algorithms

Where to go from here?

- Our database is in great shape!
  - Or at least, we know how to get it there
  - So, we can make sure about referential integrity (depending on the normal form we choose), and know where it needs to be handled by program logic

- Now let's turn to
  - triggers for advanced integrity conditions / reactions to data changes
  - Indexes to support efficient queries (in particular also based on keys from decomposed tables)



What was this all about?

Guidelines for your own review of today's session

- BCNF requires…
- It relates to 3NF in that…
  - There is a trade-off…
- Multivalued dependencies are…
  - A multivalued dependency is found by…
- Multivalued dependencies are related to functional…
  - 4NF requires….
- Denormalization is motivated by the fact that….
  - It means…