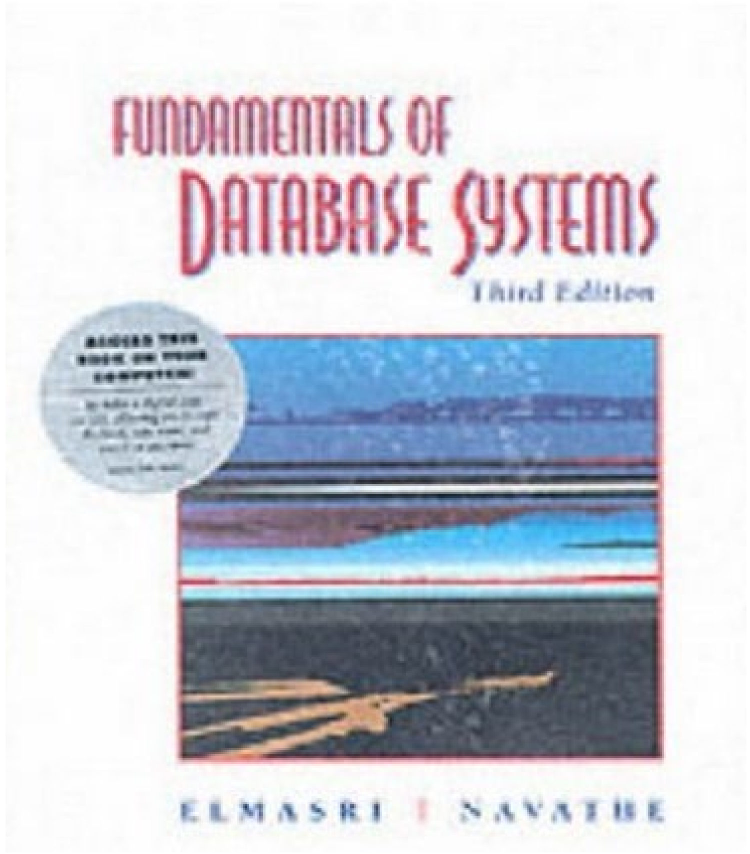
TOPIC: *NORMALISATION PART 1*

C230 Database Systems

FUNDAMENTALS OF DATABASE SYSTEMS ELMASRI AND NAVATHE BOOK

See Chapter 14

(in 3rd Edition)



MOTIVATIONS

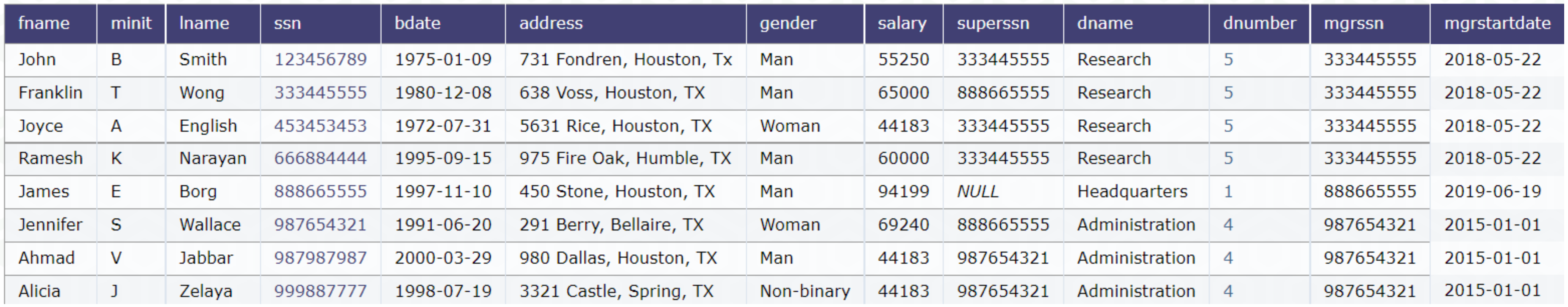
- We can see from ER examples and mappings why we get a particular grouping of tables.

However:

- What if different assumptions were made in the ER model that leads to different – maybe larger (more attributes/columns) tables?
- What happens over time as we need to add more attributes to our tables to capture information that was not part of the original requirements when creating the ER model?

For example, what if:

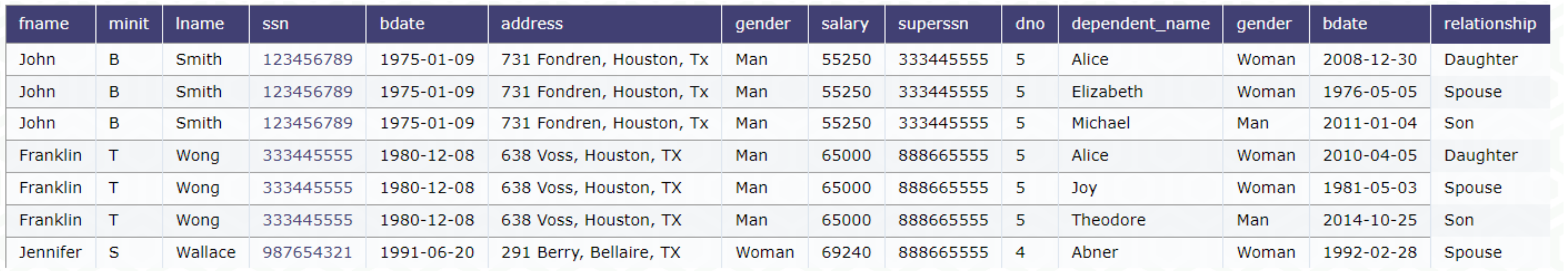
The employee entity had extra attributes to represent the department information?



fname	minit	lname	ssn	bdate	address	gender	salary	superssn	dname	dnumber	mgrssn	mgrstartdate
John	B	Smith	123456789	1975-01-09	731 Fondren, Houston, Tx	Man	55250	333445555	Research	5	333445555	2018-05-22
Franklin	T	Wong	333445555	1980-12-08	638 Voss, Houston, TX	Man	65000	888665555	Research	5	333445555	2018-05-22
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	Woman	44183	333445555	Research	5	333445555	2018-05-22
Ramesh	K	Narayan	666884444	1995-09-15	975 Fire Oak, Humble, TX	Man	60000	333445555	Research	5	333445555	2018-05-22
James	E	Borg	888665555	1997-11-10	450 Stone, Houston, TX	Man	94199	*NULL*	Headquarters	1	888665555	2019-06-19
Jennifer	S	Wallace	987654321	1991-06-20	291 Berry, Bellaire, TX	Woman	69240	888665555	Administration	4	987654321	2015-01-01
Ahmad	V	Jabbar	987987987	2000-03-29	980 Dallas, Houston, TX	Man	44183	987654321	Administration	4	987654321	2015-01-01
Alicia	J	Zelaya	999887777	1998-07-19	3321 Castle, Spring, TX	Non-binary	44183	987654321	Administration	4	987654321	2015-01-01

For example, what if:

The employee entity had the dependent information stored as attributes?



fname	minit	lname	ssn	bdate	address	gender	salary	superssn	dno	dependent_name	gender	bdate	relationship
John	B	Smith	123456789	1975-01-09	731 Fondren, Houston, Tx	Man	55250	333445555	5	Alice	Woman	2008-12-30	Daughter
John	B	Smith	123456789	1975-01-09	731 Fondren, Houston, Tx	Man	55250	333445555	5	Elizabeth	Woman	1976-05-05	Spouse
John	B	Smith	123456789	1975-01-09	731 Fondren, Houston, Tx	Man	55250	333445555	5	Michael	Man	2011-01-04	Son
Franklin	T	Wong	333445555	1980-12-08	638 Voss, Houston, TX	Man	65000	888665555	5	Alice	Woman	2010-04-05	Daughter
Franklin	T	Wong	333445555	1980-12-08	638 Voss, Houston, TX	Man	65000	888665555	5	Joy	Woman	1981-05-03	Spouse
Franklin	T	Wong	333445555	1980-12-08	638 Voss, Houston, TX	Man	65000	888665555	5	Theodore	Man	2014-10-25	Son
Jennifer	S	Wallace	987654321	1991-06-20	291 Berry, Bellaire, TX	Woman	69240	888665555	4	Abner	Woman	1992-02-28	Spouse

NORMALISATION

Normalisation rules gives us a formal measure of why one grouping of attributes in a relation schema may be better than another.

Normalised and un-normalised databases

We can distinguish between normalised and un-normalised databases

Both normalised and un-normalised databases have advantages and disadvantages

If database is normalised:

No (or very little) redundancy.

No anomalies when inserting, deleting or modifying data.

If database is normalised:

More tables.

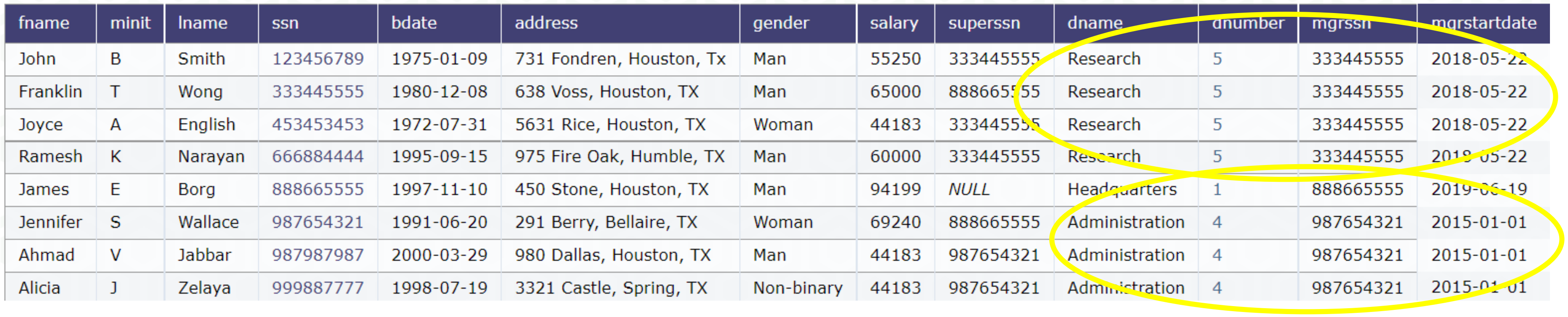
More foreign and primary keys to link tables

=> more complex queries (joins etc.)

DEFINITION: **Redundancy**

Unnecessary duplication of data in the database

e.g. if we included department details in `Employee`?



fname	minit	lname	ssn	bdate	address	gender	salary	superssn	dname	dnumber	mgrssn	mgrstartdate
John	B	Smith	123456789	1975-01-09	731 Fondren, Houston, Tx	Man	55250	333445555	Research	5	333445555	2018-05-22
Franklin	T	Wong	333445555	1980-12-08	638 Voss, Houston, TX	Man	65000	888665555	Research	5	333445555	2018-05-22
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	Woman	44183	333445555	Research	5	333445555	2018-05-22
Ramesh	K	Narayan	666884444	1995-09-15	975 Fire Oak, Humble, TX	Man	60000	333445555	Research	5	333445555	2018-05-22
James	E	Borg	888665555	1997-11-10	450 Stone, Houston, TX	Man	94199	*NULL*	Headquarters	1	888665555	2019-06-19
Jennifer	S	Wallace	987654321	1991-06-20	291 Berry, Bellaire, TX	Woman	69240	888665555	Administration	4	987654321	2015-01-01
Ahmad	V	Jabbar	987987987	2000-03-29	980 Dallas, Houston, TX	Man	44183	987654321	Administration	4	987654321	2015-01-01
Alicia	J	Zelaya	999887777	1998-07-19	3321 Castle, Spring, TX	Non-binary	44183	987654321	Administration	4	987654321	2015-01-01

CONSEQUENCES OF REDUNDANCY:

Space is wasted (due to duplication)

Data can become inconsistent due to potential problems with update, insert and delete operations

DEFINITION: Duplication

Duplicated data can naturally be present in a database and is not necessarily redundant.

For example, an attribute can have two identical values.

e.g., In company schema, `ESSN` in `works_on` may be duplicated across many projects.

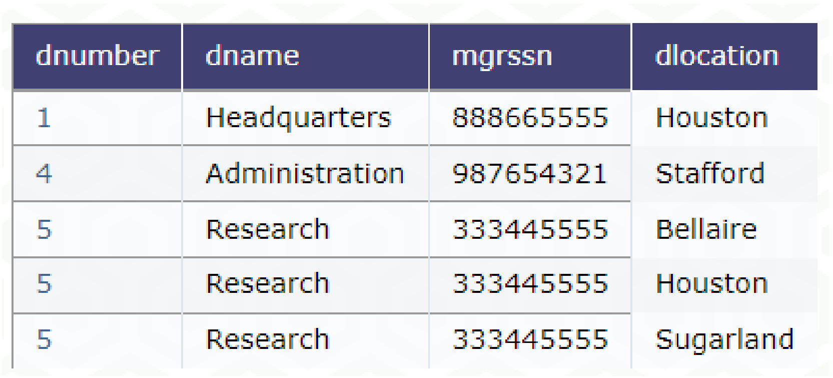
** Data is duplicated rather than redundant if when deleting data, information is lost.

EXAMPLE 1:

For the company schema, consider the following alternative schema for department which was initially created when each department had only one location:

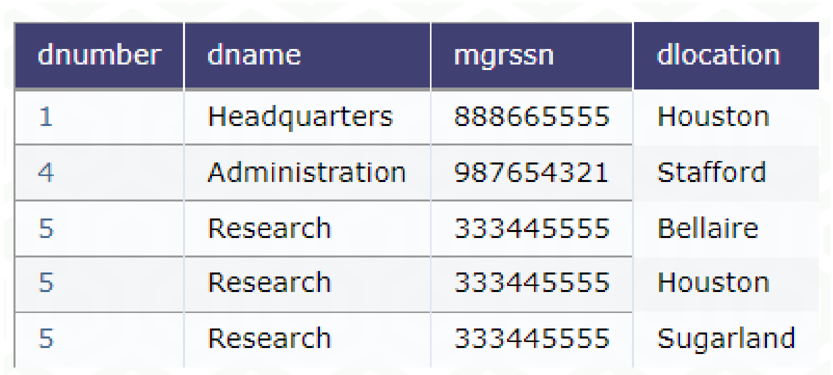
```
department(dnumber,  dname, mgrssn, dlocation)
```

However, over time as the company grew, departments were located in multiple locations:



dnumber	dname	mgrssn	dlocation
1	Headquarters	888665555	Houston
4	Administration	987654321	Stafford
5	Research	333445555	Bellaire
5	Research	333445555	Houston
5	Research	333445555	Sugarland

Problems:



dnumber	dname	mgrssn	dlocation
1	Headquarters	888665555	Houston
4	Administration	987654321	Stafford
5	Research	333445555	Bellaire
5	Research	333445555	Houston
5	Research	333445555	Sugarland

1. What can be used as the primary key?

dnumber and dlocation

2. What happens if a new manager is appointed to the department with dnumber = 5?

3 tuples will need to be modified in this case

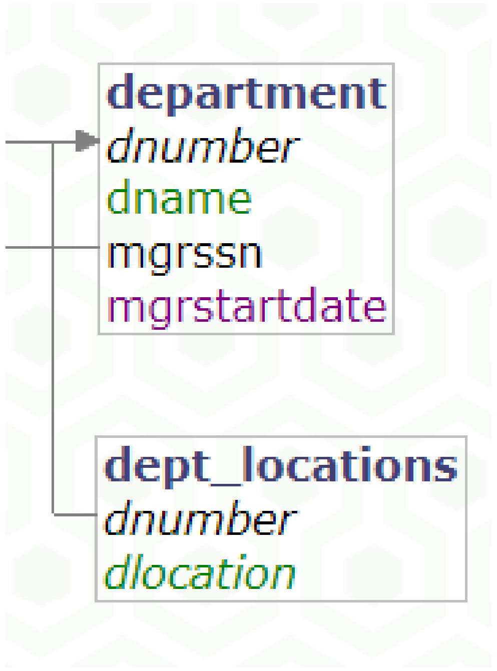
3. What happens if we add a new department, say "Development" with dnumber = 7?

Cannot be added unless we know where the department will be located.

FIXING THESE PROBLEMS?

This does not seem a good grouping of attributes …

We have seen, and worked with, a better one which stores `location` in a new table and uses `dnumber` as a foreign key to link to the other department information

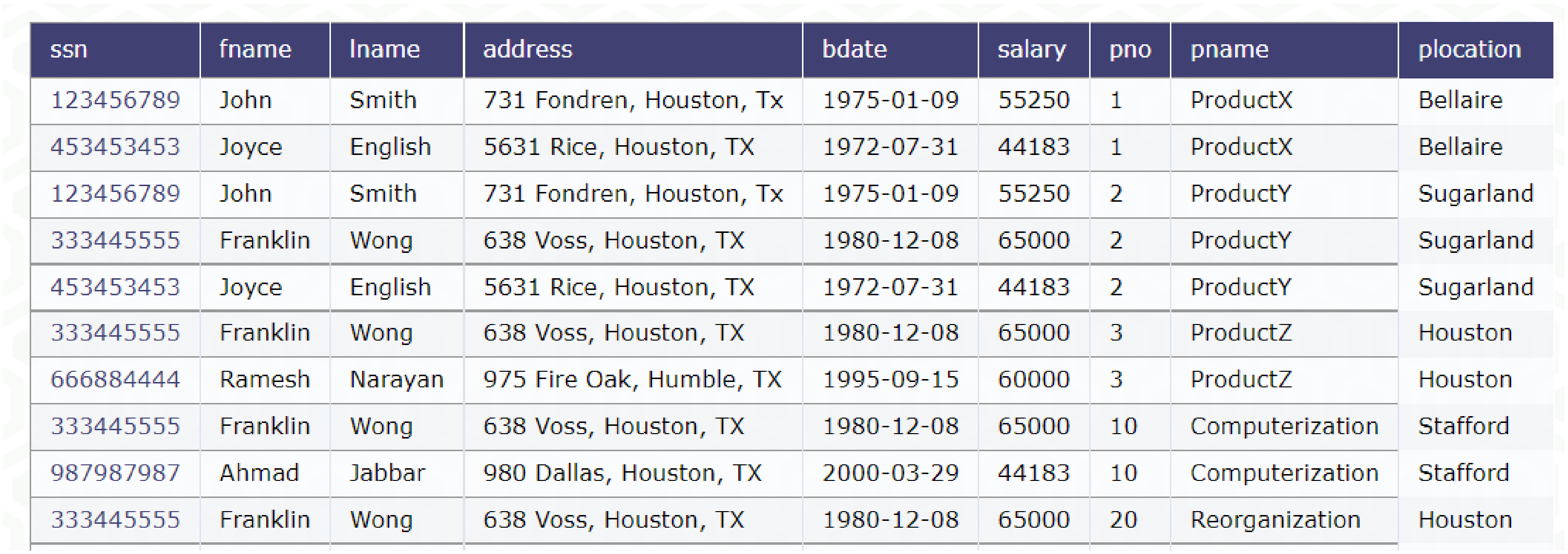


EXAMPLE 2:

For the company schema, consider the following alternative schema to store information on employees and the projects they work on:

```
employee(ssn, fname, lname, address,  bdate, salary,
 pno, pname, plocation)
```

And the following (partial) instance:



ssn	fname	lname	address	bdate	salary	pno	pname	plocation
123456789	John	Smith	731 Fondren, Houston, Tx	1975-01-09	55250	1	ProductX	Bellaire
453453453	Joyce	English	5631 Rice, Houston, TX	1972-07-31	44183	1	ProductX	Bellaire
123456789	John	Smith	731 Fondren, Houston, Tx	1975-01-09	55250	2	ProductY	Sugarland
333445555	Franklin	Wong	638 Voss, Houston, TX	1980-12-08	65000	2	ProductY	Sugarland
453453453	Joyce	English	5631 Rice, Houston, TX	1972-07-31	44183	2	ProductY	Sugarland
333445555	Franklin	Wong	638 Voss, Houston, TX	1980-12-08	65000	3	ProductZ	Houston
666884444	Ramesh	Narayan	975 Fire Oak, Humble, TX	1995-09-15	60000	3	ProductZ	Houston
333445555	Franklin	Wong	638 Voss, Houston, TX	1980-12-08	65000	10	Computerization	Stafford
987987987	Ahmad	Jabbar	980 Dallas, Houston, TX	2000-03-29	44183	10	Computerization	Stafford
333445555	Franklin	Wong	638 Voss, Houston, TX	1980-12-08	65000	20	Reorganization	Houston

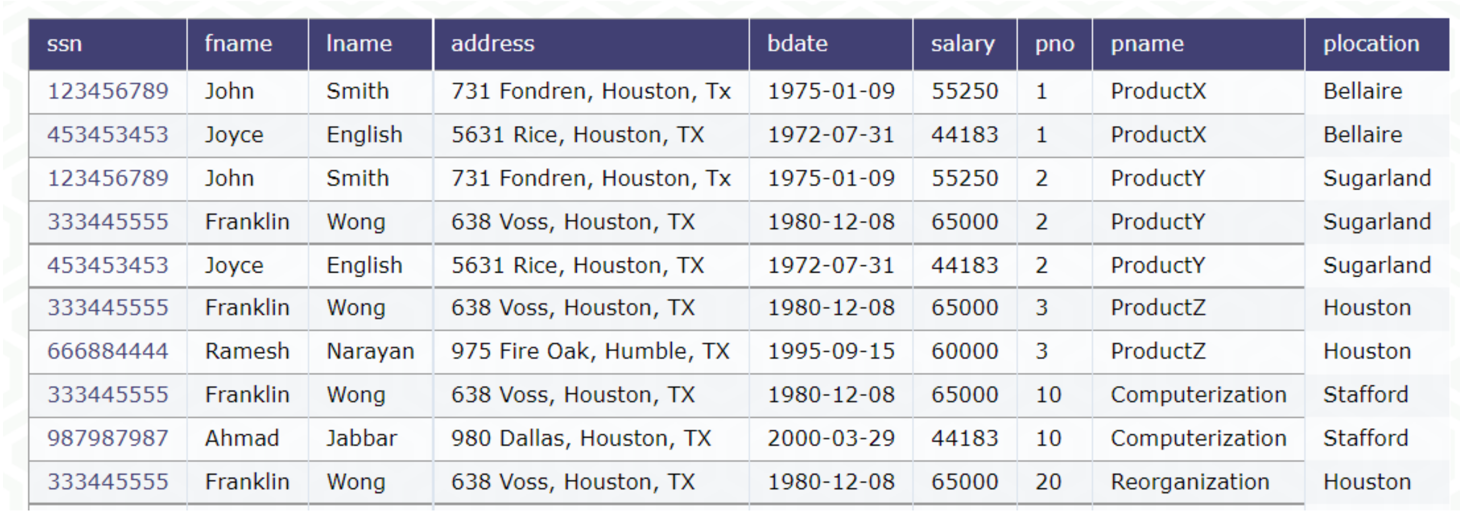
Problems?

1. What can be used as the key?

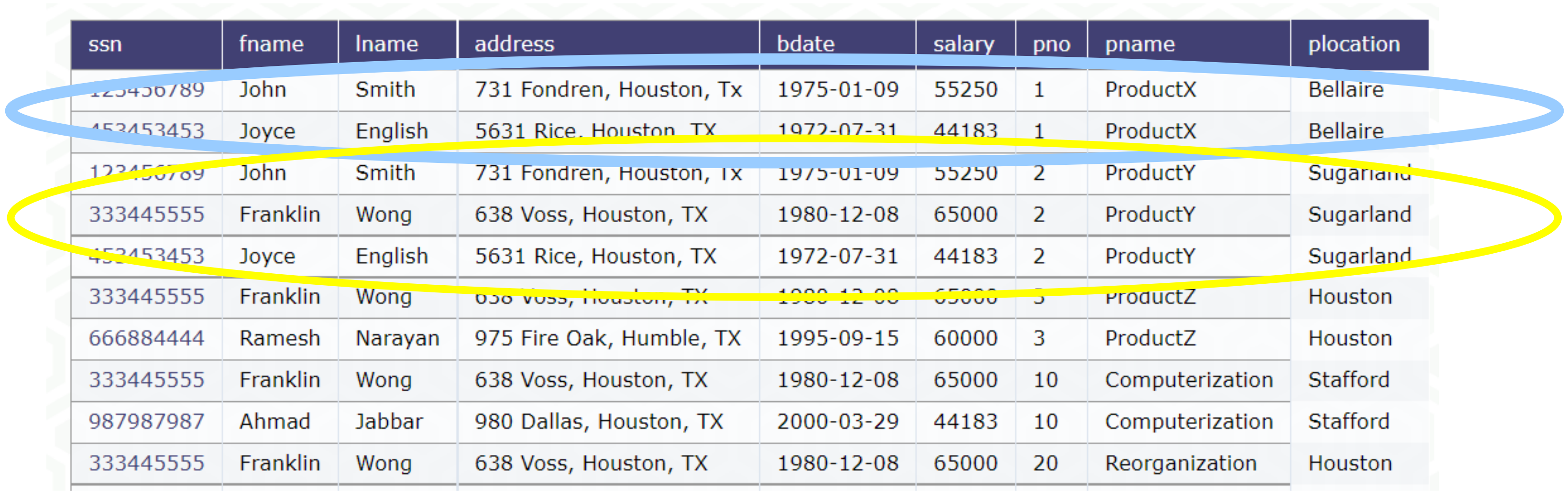
ssn and pno

2. What happens if we want to update the database when a new employee, Maria Browne, of 24 Cherry Drive, Voss, Houston, joins the company (with ssn = 34343434343)

cannot be added unless she is given a project to work on



ssn	fname	lname	address	bdate	salary	pno	pname	plocation
123456789	John	Smith	731 Fondren, Houston, Tx	1975-01-09	55250	1	ProductX	Bellaire
453453453	Joyce	English	5631 Rice, Houston, TX	1972-07-31	44183	1	ProductX	Bellaire
123456789	John	Smith	731 Fondren, Houston, Tx	1975-01-09	55250	2	ProductY	Sugarland
333445555	Franklin	Wong	638 Voss, Houston, TX	1980-12-08	65000	2	ProductY	Sugarland
453453453	Joyce	English	5631 Rice, Houston, TX	1972-07-31	44183	2	ProductY	Sugarland
333445555	Franklin	Wong	638 Voss, Houston, TX	1980-12-08	65000	3	ProductZ	Houston
666884444	Ramesh	Narayan	975 Fire Oak, Humble, TX	1995-09-15	60000	3	ProductZ	Houston
333445555	Franklin	Wong	638 Voss, Houston, TX	1980-12-08	65000	10	Computerization	Stafford
987987987	Ahmad	Jabbar	980 Dallas, Houston, TX	2000-03-29	44183	10	Computerization	Stafford
333445555	Franklin	Wong	638 Voss, Houston, TX	1980-12-08	65000	20	Reorganization	Houston



ssn	fname	lname	address	bdate	salary	pno	pname	plocation
123456789	John	Smith	731 Fondren, Houston, Tx	1975-01-09	55250	1	ProductX	Bellaire
453453453	Joyce	English	5631 Rice, Houston, TX	1972-07-31	44183	1	ProductX	Bellaire
123456789	John	Smith	731 Fondren, Houston, Tx	1975-01-09	55250	2	ProductY	Sugarland
333445555	Franklin	Wong	638 Voss, Houston, TX	1980-12-08	65000	2	ProductY	Sugarland
453453453	Joyce	English	5631 Rice, Houston, TX	1972-07-31	44183	2	ProductY	Sugarland
333445555	Franklin	Wong	638 Voss, Houston, TX	1980-12-08	65000	3	ProductZ	Houston
666884444	Ramesh	Narayan	975 Fire Oak, Humble, TX	1995-09-15	60000	3	ProductZ	Houston
333445555	Franklin	Wong	638 Voss, Houston, TX	1980-12-08	65000	10	Computerization	Stafford
987987987	Ahmad	Jabbar	980 Dallas, Houston, TX	2000-03-29	44183	10	Computerization	Stafford
333445555	Franklin	Wong	638 Voss, Houston, TX	1980-12-08	65000	20	Reorganization	Houston

3. Update the database when ProductX and ProductY are completed and details on the projects should be removed

If we delete the relevant tuples, then all details on John Smith will be lost

4. Update the database with a new address for Franklin Wong

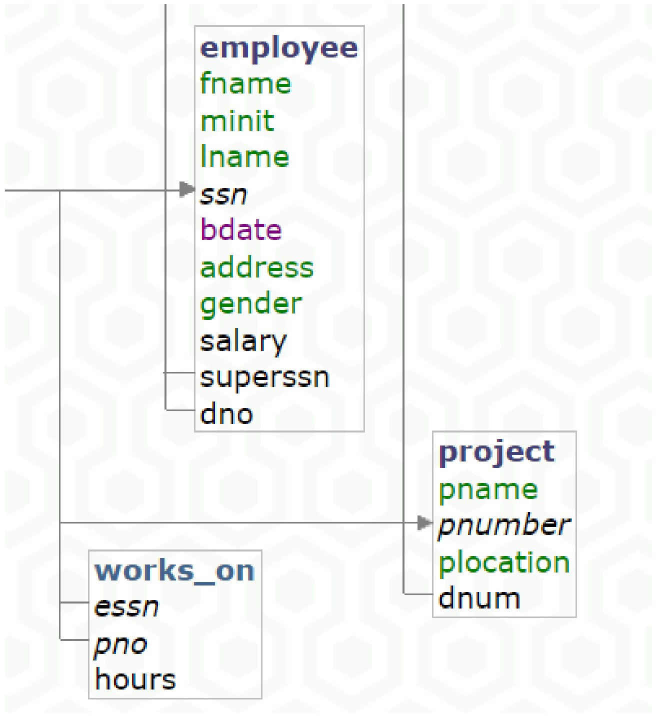
In this case, 4 tuples must be updated with the new address

FIXING THESE PROBLEMS?

This does not seem a good grouping of attributes …

We have seen, and worked with, a better one **involving 3 tables**

Note however the repetition of `ssn` (as `essn`) and `pnumber/pno`



NORMALISATION

Developed by Codd, 1972

- Takes each table through a series of tests to "verify" whether or not it belongs to a certain **normal form**
- Normal forms to check:
  - 1st, 2nd and 3rd normal forms (NF)
  - Boyce-Codd normal form – strong 3NF
  - 4th and 5th Normal Forms
- *We will consider 1NF, 2NF and 3NF only in detail*

NORMALISATION PROVIDES:

1. Formal framework for analysing relation schemas based on keys and functional dependencies among attributes.

2. Series of tests so that a database can be normalised to any degree (e.g., from 1NF to 5NF).

3. ***But*** does not necessarily provide a good design if considered in isolation to everything else.

WHY NORMALISE?

- Redundancy will be reduced or eliminated.
- Storage space will be reduced as a result.
- Task of maintaining data integrity is made easier.

However with normalisation, tables are usually added to the schema and linked with foreign keys. Thus queries become more complex as they often require data from multiple tables (requiring joins or subqueries).

ALTERNATIVES?

Retain redundant data and maintain data integrity by means of code consistency checks

In some applications the number of insertions may be very small or non-existent (e.g. analysing past logs, transaction data, weather data etc.) and in such cases the overhead of normalised tables is generally **not** required.

DE-NORMALISATION

A process of making compromises to the normalised tables by introducing intentional redundancy for performance reasons (querying performance).

Typically, de-normalisation will improve query times at the expense of data updates (insert, delete, update).

DEFINITION: Functional Dependency

Functional dependency is one of the main concepts associated with normalisation and describes the *relationship between attributes.*

If A and B are attributes of a relation R, then **B is functionally dependent (FD) on A** if each value of A is associated with exactly one value of B.

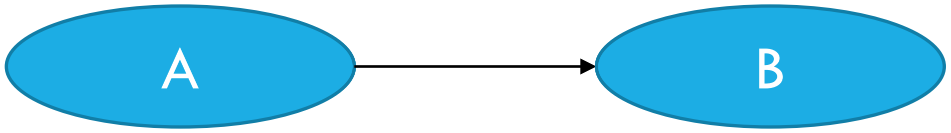
i.e., values in B are uniquely determined by values of A

TERMINOLOGY: FUNCTIONAL DEPENDENCY (FD)

$A \rightarrow B$:

FD from A to B

B is FD on A



NOTES ON NOTATION:

$A \longrightarrow B$ does not necessarily imply $B \longrightarrow A$

$A \leftrightarrow B$ denotes $A \longrightarrow B$ and $B \longrightarrow A$

$A \longrightarrow \{B, C\}$ denotes $A \longrightarrow B$ and $A \longrightarrow C$

$\{A, B\} \longrightarrow C$ denotes that it is the **combination** of A and B that uniquely determines C.

TERMINOLOGY: CANDIDATE KEY (CK)

Every relation has one or more candidate keys. A candidate key (CK) is one or more attribute(s) in a relation with which you can determine all the attributes in the relation.

Recall we pick one such candidate key as the primary key of a relation.

EXAMPLE 3: FINDING THE FUNCTIONAL DEPENDENCIES – GIVEN THE PRIMARY KEY

For the company schema, consider the following alternative schema to hold information on employees and projects:

`emp_proj(ssn, pnumber, hours, ename, pname, plocation)`

What are the functional dependencies?

- Think of this question as … "which attribute can be uniquely determined from another attribute"
- Begin with any known PK or CK

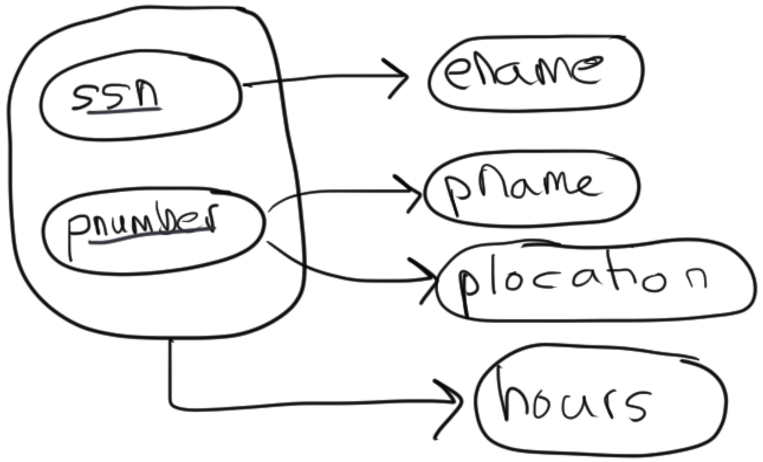
Can represent these FDs graphically:

`emp_proj(`<u>`ssn, pnumber`</u>`, hours, ename, pname, plocation)`

ssn ⟶ ename

pnumber ⟶ {pname, plocation}

{ssn, pnumber} ⟶ hours



IMPORTANT TO NOTE:

A functional dependency is a property of a relation schema *R* and cannot be inferred automatically but instead must be defined explicitly by someone who knows the **semantics** of *R*

i.e.

You will either be:

- explicitly given all FDs.
- given enough information about the attributes and the domain to *reasonably* infer the FDs (perhaps having to make certain assumptions).

TYPES OF FUNCTIONAL DEPENDENCIES

1. Full Functional Dependency:

A functional dependency {X,Y} $\rightarrow$ Z is a **full** functional dependency if when some attribute (either X or Y) is removed from the LHS the dependency **does not hold**.

Note: There may be any number of attributes on LHS

2. Partial Functional Dependency:

A functional dependency {X,Y} $\rightarrow$ Z is a **partial** functional dependency if some attribute (either X or Y) can be removed from the LHS and the dependency **still holds.**

Note: There may be any number of attributes on LHS

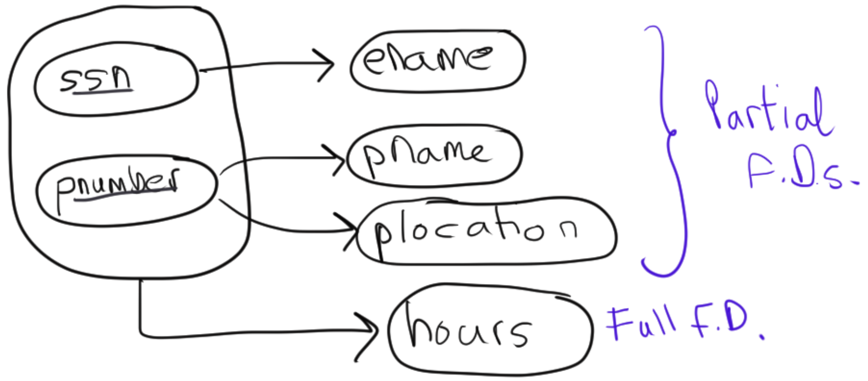
CONSIDER EXAMPLE 3 AGAIN:

`emp_proj(ssn, pnumber, hours, ename, pname, plocation)`

Are the following Full or Partial Functional Dependencies?

{ssn, pnumber} → hours

{ssn, pnumber} → ename



TYPES OF FUNCTIONAL DEPENDENCIES

3. Transitive Dependency:

A functional dependency **X** → **Y** is a transitive dependency in the table/relation R if there is a set of attributes **Z** that is neither a candidate key nor a subset of any key of R and both:

X → **Z** and

Z → **Y**

hold.

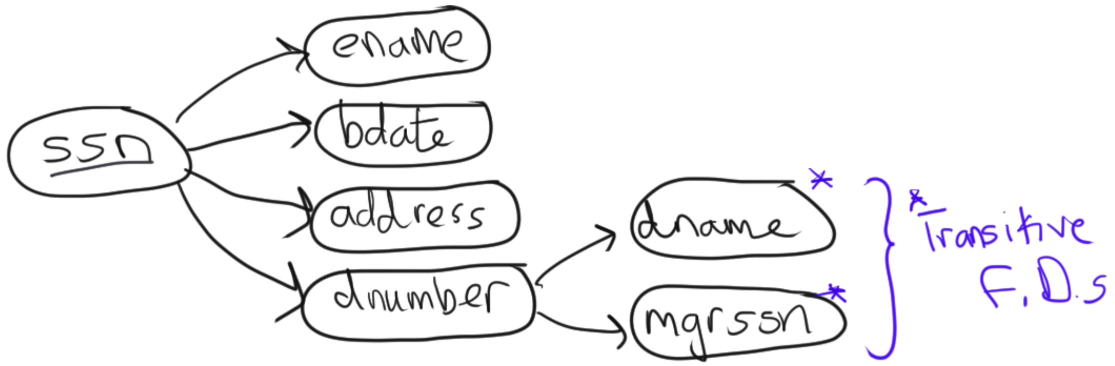
EXAMPLE 4:
Consider information on employees and departments

```
emp_dept(ename, ssn, bdate, address, dnumber,
dname, dmgrssn)
```

The functional dependencies are:

ssn → {ename, bdate, address, dnumber}

dnumber → {dname, dmgrssn)



EXAMPLE 4:
An example of a transitive dependency

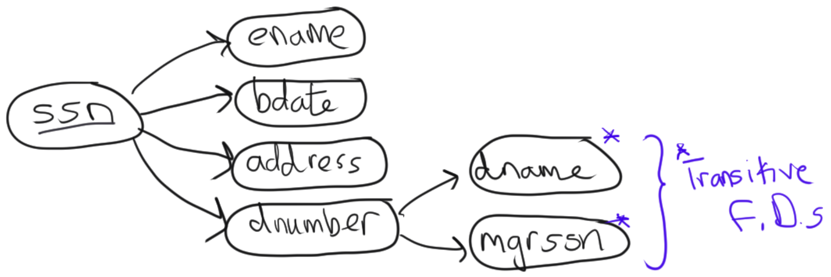
The dependency:

ssn → dmgrssn

is transitive through dnumber because both the following hold:

ssn → dnumber

dnumber → dmgrssn



But dnumber is neither a key or a subset of the key.

EXAMPLE 5:

Given the following table to hold student data:

`student(id, name, course, assocCollege, courseCoordinator)`

and the following Functional Dependencies:

id $\longrightarrow$ name

id $\longrightarrow$ course

course $\longrightarrow$ assocCollege

course $\longrightarrow$ courseCoordinator

EXAMPLE 5:
What is the candidate key?
What are the full dependencies?
What are the transitive dependencies?

Given the following table to hold student data:

```
student(id, name, course, assocCollege,
courseCoordinator)
```

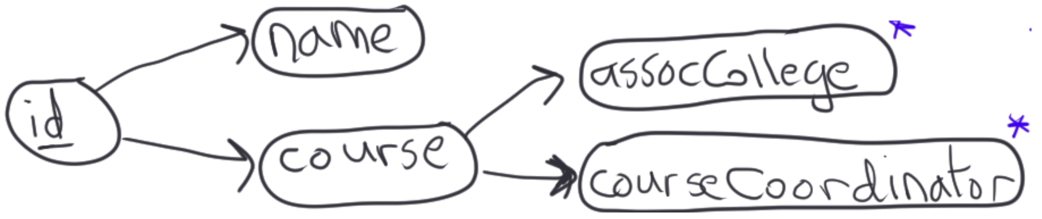
and the following Functional Dependencies:

id → name

id → course

course → assocCollege

course → courseCoordinator



EXAMPLE 6:
Draw the functional dependency diagram and find the candidate key

Consider the table R with 5 attributes

`R(A, B, C, D, E)`

and the following functional dependencies:

$A \rightarrow B$

$B \rightarrow A$

$B \rightarrow C$

$D \rightarrow A$

`R(A, B, C, D, E)`

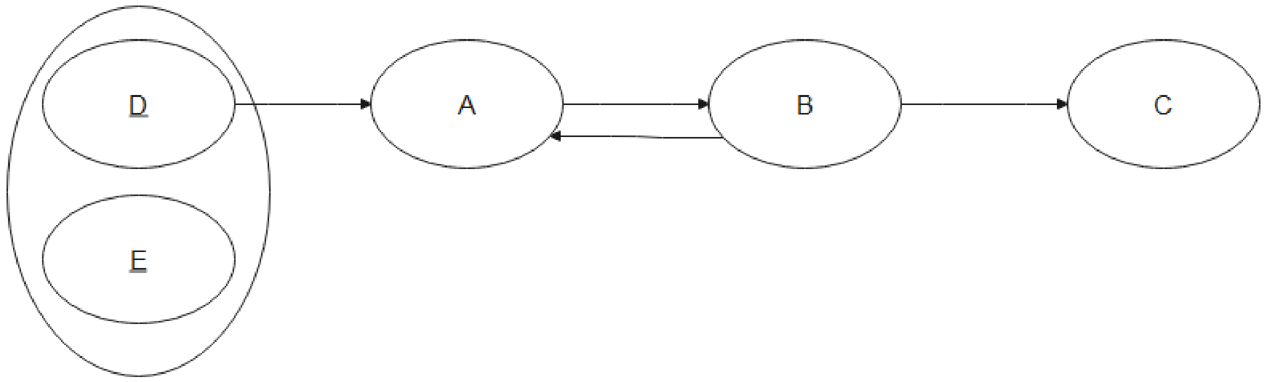
and the following functional dependencies:

$A \rightarrow B$

$B \rightarrow A$

$B \rightarrow C$

$D \rightarrow A$



Inference rules for Functional Dependencies

Typically the main **obvious** functional dependencies are specified for a schema

– call these F.

However many others can be inferred from F

– call these closure of F: F^+

FOR EXAMPLE:

$$F = \{ \quad A \rightarrow \{B, C, D, E\}$$
$$E \rightarrow \{F, G\} \quad \}$$

Some other FDs which can be inferred:

$A \rightarrow A$

$A \rightarrow \{F, G\}$

$E \rightarrow F$

etc.

Inference Rules for FDs:

1. Reflexive: Trivially, an attribute, or set of attributes, always determines itself.

2. Augmentation: if $X \rightarrow Y$ can infer $XZ \rightarrow YZ$

3. Transitive: if $X \rightarrow Y$ and $Y \rightarrow Z$ can infer $X \rightarrow Z$

4. Decomposition: if $X \rightarrow YZ$ can infer $X \rightarrow Y$

5. Union (additive): if $X \rightarrow Y$ and $X \rightarrow Z$ can infer if $X \rightarrow YZ$

6. Pseudotransitive: if $X \rightarrow Y$ and $WY \rightarrow Z$ can infer $WX \rightarrow Z$

*Note: Rules 1, 2 and 3 are together called **Armstrongs's Axioms**