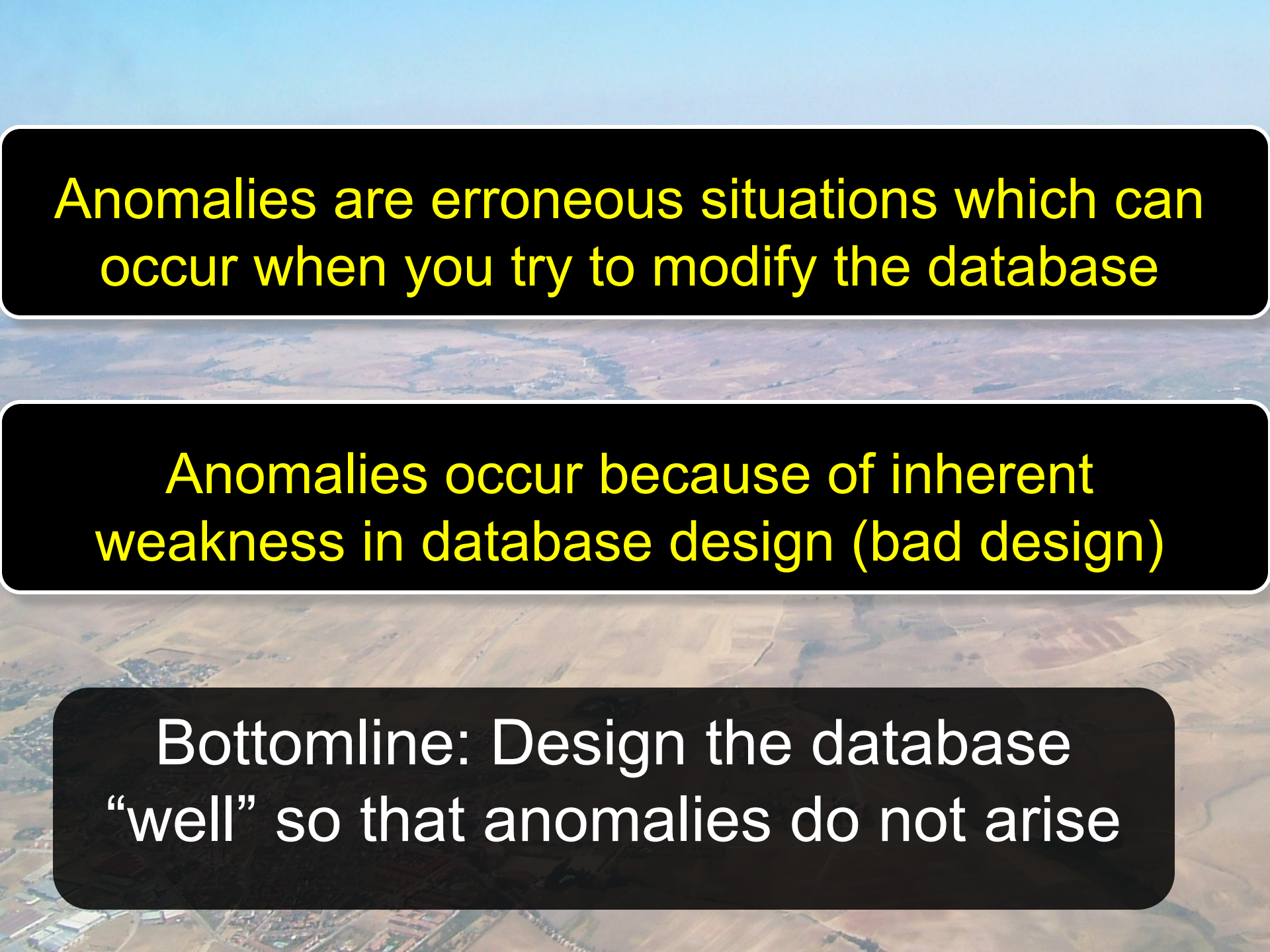


An aerial photograph of a rural landscape. In the foreground, a town with red-roofed buildings is visible. A river winds through the middle ground, surrounded by fields. The background shows a vast, flat landscape under a clear blue sky.

DATABASE NORMALIZATION

Contents of this lecture

- Anomalies
- Quick overview of FDs
- 1 NF, 2 NF and 3 NF



Anomalies are erroneous situations which can occur when you try to modify the database

Anomalies occur because of inherent weakness in database design (bad design)

Bottomline: Design the database “well” so that anomalies do not arise



The diagram features a world map as a background. A central dark blue oval labeled 'Anomalies' is connected by four teal-colored lines to four other ovals: a red one for 'Insert anomaly', a purple one for 'Update anomaly', and a yellow one for 'Delete anomaly'. The fourth teal line extends from the central oval towards the top right but does not connect to a visible oval.

Anomalies

***Insert
anomaly***

**Update
anomal
y**

**Delete
anomal
y**

Insert anomaly

Student ID	Course ID	Course name
200801	CS101	Programming
200805	CS201	OS
200807	CS203	DBMS

If student ID 200807 wants to register for course CS 101, can you insert a tuple?

If there is a course CS405 on Advanced DBMS, can you insert a tuple into this table?

Insert anomaly

You cannot insert a tuple for CS405 because no student is currently taking that course!

The table suffers from insert anomaly because you cannot record the existence of courses in which no student is registered

Insert anomaly – Another example

Faculty ID	PhD student ID	Research area
F5	200401	Databases
F7	200404	Security
F9	200407	Mobility

If student ID 200403 decides to take faculty ID F10 as his supervisor, can you insert a tuple?

If there is a faculty with ID F20 such that he is not supervising any PhD student, can you record any information on F20 in this table?

Insert anomaly – Yet Another example

Dept ID	Employee ID	Dept name
D5	200401	R&D
D7	200404	Finance
D9	200407	Marketing/Sales

If there is a new department X which has currently no employees, you cannot record any information about X (i.e., insert a tuple about dept. X) in this table just because no employee works for that department!

Deletion anomaly

Student ID	Course ID	Course name
200801	CS101	Programming
200805	CS201	OS
200807	CS203	DBMS

What if student ID 200805 leaves course CS201?

Information about CS201 will be lost because the last student just left, even though the course still exists!

Deletion anomaly – Yet Another example

Dept ID	Employee ID	Dept name
D5	200401	R&D
D7	200404	Finance
D9	200407	Marketing/Sales

If employee ID 200401 quits the company, information about dept ID D5 will be LOST from the table, even though dept D5 still exists!

Update anomaly

Student ID	Course ID	Email ID
200801	CS101	abc@gmail.com
200805	CS201	xyz@gmail.com
200801	CS201	abc@gmail.com
200801	CS203	abc@gmail.com
200807	CS203	uvw@gmail.com

Suppose student ID 200801 wants to change his email address to “cbd@gmail.com”

Update anomaly occurs if you update the email address in some rows, but not in all the rows

Update anomaly

Student ID	Course ID	Email ID
200801	CS101	cbd@gmail.com
200805	CS201	xyz@gmail.com
200801	CS201	cbd@gmail.com
200801	CS203	abc@gmail.com
200807	CS203	uvw@gmail.com

Observe that rows 1 and 3 were updated, but row 4 still has his old email address

Update anomaly

Now if you try to find out 200801's email address, how will you know which is the correct email address?

Update anomaly should be avoided because it puts the database in an inconsistent state



Quick question

Can update anomaly occur if an employee changes his residential address?

Yes, if the employee's address was present on multiple rows, and you failed to update all the rows with the new address

Quick question

Can update anomaly occur if an employee shifts from one department to another?

Yes, if the employee's dept was present on multiple rows, and you failed to update all the rows with the new dept

An aerial photograph of a rural landscape. In the foreground, a town with red-roofed buildings is visible. A river flows through the landscape, winding around the town and into the distance. The surrounding area is a patchwork of fields and roads. The sky is clear and blue.

Quick overview of Functional Dependencies

Functional dependency (FD)

- If $X \rightarrow Y$, we say X functionally determines Y or Y is functionally dependent on X.
 - Example: SSN \square ENAME
- We abbreviate functional dependency by FD. X is called the left-hand side of the FD. Y is called the right-hand side of the FD.
- A functional dependency is a property of the meaning or semantics of the attributes, i.e., a property of the relation schema.
 - They must hold on all relation states (extensions) of R. Relation extensions $r(R)$ that satisfy the FD are called legal extensions.
- FDs cannot be inferred from a given relation extension r , but must be defined explicitly by someone who knows the semantics of the attributes of R.

Functional Dependencies (Cont)

- From the FDs:

$F = \{SSN \rightarrow \{ENAME, BDATE, ADDRESS, DNUMBER\},$
 $DNUMBER \rightarrow \{DNAME, DMGRSSN\}\}$

we can infer the following FDs:

$SSN \rightarrow \{ENAME, DMGRSSN\}$

- A FD $X \rightarrow Y$ is *inferred* from a set of dependencies F specified on R if $X \rightarrow Y$ holds in every relation state r that is a legal extension of R .
- $F \models X \rightarrow Y$ denotes $X \rightarrow Y$ is inferred from F .
- The *closure* of F , denoted by F^+ , is the set of all FDs that can be inferred from F .

Functional dependency (FD)

- A FD $X \rightarrow Y$ is a *full functional dependency* if removal of any attribute from X means that the dependency does not hold any more; otherwise, it is a *partial functional dependency*.
- An attribute is *prime* if it is a member of *any* key (Primary or candidate)
- *Partial dependency means that an attribute depends on only part of the primary key.*
 - Observe that for a partial dependency to exist, the primary key must be a composite key (i.e., there needs to be more than 1 attribute inside the primary key).

Functional dependency (FD)

- Transitive dependencies
 - A dependency which is transitive
 - Example: $SSN \rightarrow DMGRSSN$ is transitive as $SSN \rightarrow Dnumber \rightarrow DMGRSSN$
 - An attribute depends on some attribute, which is not the primary key.

An aerial photograph of a rural landscape. In the foreground, a town with red-roofed buildings is visible. A river flows through the landscape, winding around the town and into the distance. The surrounding area is a patchwork of fields and farmland. The sky is clear and blue.

Database normalization

Database normalization

- Main objectives of database normalization
 - Design a relation schema so that it is easy to explain its meaning (semantics)
 - Do not combine attributes from multiple entity types and relationship types into single relation.
 - In short, store the data logically
 - Reducing redundant values in tuples saves storage space and avoid update anomalies.
 - Insertion anomalies
 - Deletion anomalies
 - Modification anomalies

“Guidelines”

For putting the database into a certain normal form, these are guidelines, hence nothing is set in stone

In the real-world, sometimes you may need to break these guidelines, and this is perfectly ok, as long as you account for possibilities of anomalies

Cumulative nature of Guidelines

Think of it as a CUMULATIVE series of guidelines.

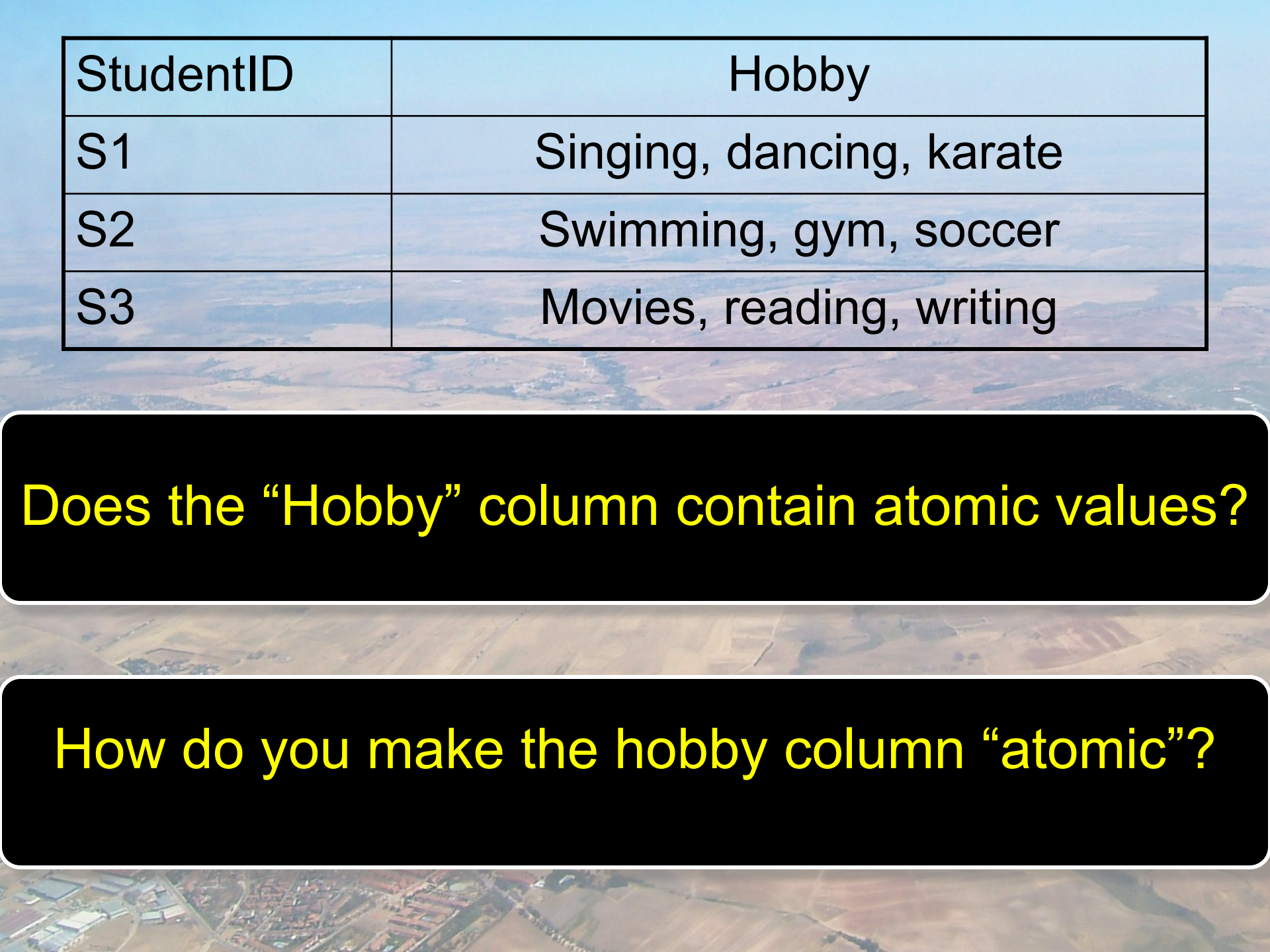
Satisfying the first series of guidelines puts the database in 1NF, satisfying the first and second series means that your database is in 2NF and so on

Big picture of normalization from 1 NF to 3NF

- 1NF □ Ensure “atomic values”
- 2NF □ Remove partial dependencies
 - *Partial dependency means that an attribute depends on only part of the primary key.*
 - Observe that for a partial dependency to exist, the primary key must be a composite key (i.e., there needs to be more than 1 attribute inside the primary key).
 - Tip: Look out for tables with composite key
- 3NF □ Remove transitive dependencies
 - A dependency which is transitive
 - Example: $SSN \rightarrow DMGRSSN$ is transitive as $SSN \rightarrow Dnumber \rightarrow DMGRSSN$
 - An attribute depends on some attribute, which is not the primary key.

1NF

- A relation R is in *first normal form* if domains of attributes include only atomic values.



StudentID	Hobby
S1	Singing, dancing, karate
S2	Swimming, gym, soccer
S3	Movies, reading, writing

Does the “Hobby” column contain atomic values?

How do you make the hobby column “atomic”?

StudentID	Hobby
S1	Singing, dancing, karate
S2	Swimming, gym, soccer
S3	Movies, reading, writing

Any problem with making separate columns such as Hobby1, Hobby2, Hobby3 and so on?

Cannot efficiently address a dynamic number of hobbies. Not a good idea to keep adding new columns to the table

StudentID	Hobby
S1	Singing, dancing, karate
S2	Swimming, gym, soccer
S3	Movies, reading, writing

StudentID	Hobby
S1	Singing
S1	dancing
S1	karate
S2	Swimming
S2	gym
S2	soccer
S3	Movies
S3	reading
S3	writing



How to handle
duplicate
primary keys?
Look at the
walk-through
example in the
subsequent
slides.

2NF


- A relation R is in *second normal form* if every non-prime attribute A in R is not partially dependent on any key of R .
- Alternatively, R is in 2NF if every non-prime attribute A in R is fully dependent on every key of R .
- In essence, it means that in 2NF, records should depend only upon a table's primary key
 - a compound key


3NF

- A relation R is in *third normal form* if for every FD $X \rightarrow A$ that holds on R , either
 - X is a superkey of R , or
 - A is a prime attribute of R .

(Alternative Def . - No transitive dependencies – If there is a set of attributes Z that is neither a candidate key nor a subset of any key (primary or candidate) of R , $X \rightarrow Z$ and $Z \rightarrow Y$ holds.

$SSN \rightarrow DMGRSSN$ is transitive as $SSN \rightarrow Dnumber \rightarrow DMGRSSN$ (Emp-dept) and $dnumber$ is neither a key nor a subset of key.

- 
- An aerial photograph of a town with red-roofed buildings, surrounded by brown and yellow fields under a clear blue sky. The town is located in the lower-left quadrant of the image, with a river or stream winding through the landscape to its right.
- In the subsequent slides, we will see a walk-through example for converting from unnormalized data to 3NF.
 - Recall the big picture of normalization to 3NF
 - 1NF ☐ Ensure “atomic values”
 - 2NF ☐ Remove partial dependencies
 - 3NF ☐ Remove transitive dependencies

An aerial photograph of a rural landscape. In the foreground, a small town or village is visible, characterized by a cluster of buildings with red-tiled roofs and some larger industrial or commercial structures. The town is surrounded by a patchwork of agricultural fields in various shades of brown and tan. In the background, the landscape transitions into rolling hills and valleys, with a winding river or stream visible on the right side. The sky is a clear, pale blue.

Walk-through example for a supermarket database from unnormalized data to 3NF

Walk-through example

Think of a supermarket database, which stores details of products sold at the checkout counter

What could be the attributes in the supermarket database?

The very first step is always to think of what attributes you would require for a database based on the application scenario

Possible attributes for a supermarket database

- There is a unique transaction ID for each transaction ☐ TransactionID could be an attribute.
- There is a time and date of transaction ☐ Time and Date could be attributes
- There is a customer and a salesclerk.
 - Customer: There could be attributes based on CustID, CustName, CustEmail, CustPhone, CustDOB, CustHomeAddress
 - Salesclerk: SalesClerkID, SalesClerkName, SalesClerkTraining etc
- The customer will buy some products, which the salesclerk will checkout ☐ There could be attributes based on items e.g., ProductDesc, ProductQuantity, ProductPrice, ProductRestrictions.

Walk-through example

Are these attributes sufficient or do you want to include more attributes into the database?

Depends on how you are going to use the database (i.e., queries)

For example, if you want to use the database to improve the speed of checkout, you would need additional attributes

Table Checkout has the following attributes (columns)

- TransactionID
- Date
- CustID
- CustName
- CustEmail
- CustPhone
- SalesClerkID
- SalesClerkName
- SalesClerkTraining
- Product1Desc
- Product1Quantity
- Product1Price
- Product2Desc
- Product2Quantity
- Product2Price
- Product3Desc
- Product3Quantity
- Product3Price

1NF



What do you need to do to convert this table to 1NF?

Throughout this example, we will assume that product price depends not only on productID, but also on TransactionID. WHY? Because the same product could be sold to different customers at different prices, depending upon deals (e.g., buy 3, get 1 free), customer loyalty awards (supermarket credit/debit cards), date of sale (e.g., goods could be discounted on certain days) etc

Table Checkout has the following attributes (columns)

- TransactionID
- Date
- CustID
- CustName
- CustEmail
- CustPhone
- SalesClerkID
- SalesClerkName
- SalesClerkTraining
- Product1Desc
- Product1Quantity
- Product1Price
- Product2Desc
- Product2Quantity
- Product2Price
- Product3Desc
- Product3Quantity
- Product3Price

1NF



What do you need to do to convert this table to 1NF?

Just ensure “ATOMIC” values and it will be in 1NF!

Table Checkout has the following attributes (columns)

- TransactionID
- Date
- CustID
- CustName
- CustEmail
- CustPhone
- SalesClerkID
- SalesClerkName
- SalesClerkTraining
- Product1Desc
- Product1Quantity
- Product1Price
- Product2Desc
- Product2Quantity
- Product2Price
- Product3Desc
- Product3Quantity
- Product3Price

1NF



Table Transact and Product

Table Transact

- TransactionID
- Date
- CustID
- CustName
- CustEmail
- CustPhone
- SalesClerkID
- SalesClerkName
- SalesClerkTraining

Table Product

- TransactionID
- ProductID
- ProductDesc
- ProductQuantity
- ProductPrice

Table Transact and Product

Table Transact

- TransactionID
- Date
- CustID
- CustName
- CustEmail
- CustPhone
- SalesClerkID
- SalesClerkName
- SalesClerkTraining

Table Product

- TransactionID
- ProductID
- ProductDesc
- ProductQuantity
- ProductPrice

2NF



What do you need to do to convert this table to 2NF?

Table Transact and Product

Table Transact

- TransactionID
- Date
- CustID
- CustName
- CustEmail
- CustPhone
- SalesClerkID
- SalesClerkName
- SalesClerkTraining

Table Product

- TransactionID
- ProductID
- ProductDesc
- ProductQuantity
- ProductPrice

**For converting to 2NF,
look out for the tables
which have compound
key**

2NF



What do you need to do to
convert this table to 2NF?

**Just remove PARTIAL
DEPENDENCIES and it will
be in 2NF!**

Table Transact and Product

Table Transact

- TransactionID
- Date
- CustID
- CustName
- CustEmail
- CustPhone
- SalesClerkID
- SalesClerkName
- SalesClerkTraining

Table Product

- TransactionID
- ProductID
- ProductDesc
- ProductQuantity
- ProductPrice

2NF



Table Transact

- TransactionID
- Date
- CustID
- CustName
- CustEmail
- CustPhone
- SalesClerkID
- SalesClerkName
- SalesClerkTraining

Table ProductSale

- TransactionID
- ProductID
- ProductQuantity
- ProductPrice

Table ProductDesc

- ProductID
- ProductDesc

Table Transact and Product

Table Transact

- TransactionID
- Date
- CustID
- CustName
- CustEmail
- CustPhone
- SalesClerkID
- SalesClerkName
- SalesClerkTraining

Table ProductSale

- TransactionID
- ProductID
- ProductQuantity
- ProductPrice

Table ProductDesc

- ProductID
- ProductDesc

3NF



What do you need to do to convert this table to 3NF?

Just remove TRANSITIVE DEPENDENCIES and it will be in 3NF!

Table Transact and Product

Table Transact

- TransactionID
- Date
- CustID
- CustName
- CustEmail
- CustPhone
- SalesClerkID
- SalesClerkName
- SalesClerkTraining

Table ProductSale

- TransactionID
- ProductID
- ProductQuantity
- ProductPrice

Table ProductDesc

- ProductID
- ProductDesc

3NF



Table Transact

- TransactionID
- Date
- CustID
- SalesClerkID

Table Cust

- CustID
- CustName
- CustEmail
- CustPhone

Table SalesClerk

- SalesClerkID
- SalesClerkName
- SalesClerkTraining

Table ProductSale

- TransactionID
- ProductID
- ProductQuantity
- ProductPrice

Table ProductDesc

- ProductID
- ProductDesc

Conversion from 2NF to 3NF

- Identify all transitive dependencies.
- For each transitive dependency, look at the LHS of the FD e.g., if the transitive dependency is $X \twoheadrightarrow Y$, look at X .
- Create a new table with X as the primary key and put the attributes dependent on X into that table. (The attributes dependent on X are the RHS of the FD e.g., Y in this case)
- Delete away (from the previous table) those attributes, which you put into the new table e.g., delete Y from the previous table

Look carefully at the example in the previous slide and this is precisely what we did!

Conversion from 2NF to 3NF

- In our example, CustName, CustPhone, CustEmail have a transitive dependency because
 - CustID (which is not a primary key for the table Transact) determines all these values
 - That is, CustName, CustPhone, CustEmail depend on an attribute, which is not a primary key, hence transitive dependency
- Transitive dependency we identified was CustID \square CustName, CustPhone, CustEmail
- Then we took the LHS CustID and created a new table with CustID as the primary key. And added the attributes on the RHS (i.e., CustName, CustPhone, CustEmail) to that new table.
- Finally, we deleted away the RHS attributes from the previous table.

Homework assignment

- In the lecture, I showed you a walk-through example for a supermarket database application
- For your homework practice no-marks assignment, design a database for an e-commerce website and normalize the database upto 3NF.
- Remember that first of all, you would need to think of all reasonable attributes that you would like to have in the database

Homework assignment

- This is the type of assignment, which you would need to do in the real-world.
- Just knowing the theory of how to convert a given database to 3NF is not adequate
- In real-world scenarios, the attributes needed in the database will not be given to you by someone else
 - You would need to figure out by yourself what attributes should be there in the database
- You need to apply the theory adequately to real-world scenarios for a much deeper understanding

Learning suggestion for this chapter

- Do not just memorize definitions e.g., FDs, partial dependencies, transitive dependencies etc
- Understand clearly what these definitions mean and how you can **APPLY** these definitions in practice
 - There are multiple definitions for 1NF, 2NF and 3NF (as you saw on these slides), but essentially all these definitions are equivalent
 - They mean the same thing! These definitions are just different ways of saying the same thing.
 - Focus on any one definition (whichever one sounds most intuitive to you) and then use that definition
 - If you try to use all these definitions together, chances are that you will get confused

Learning suggestions

- Probably the best way to understand normalization is to consider some specific database application (e.g., library, University, company databases)
 - And then starting from the unnormalized data, normalize all the way upto 3NF (for most practical scenarios) or upto BCNF, 4NF or 5NF etc
- After you consider a few such database applications and do the normalization on those databases
 - You will be in a position to apply what you have learned
 - Your concepts about normalization will be crystal-clear.