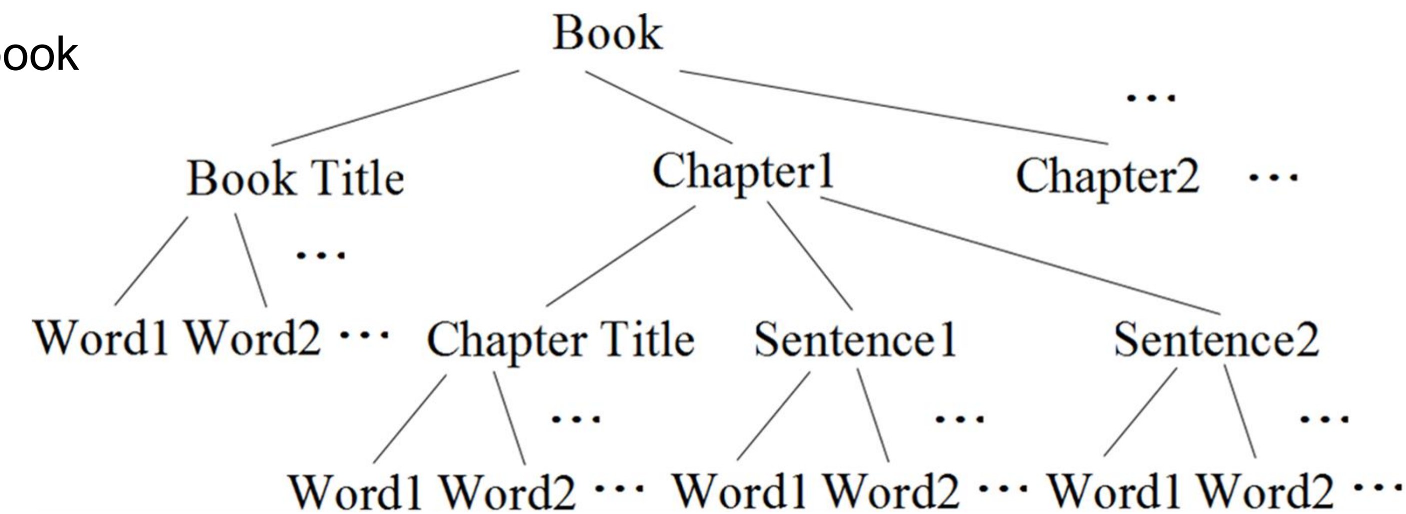
Complex Objects (I)

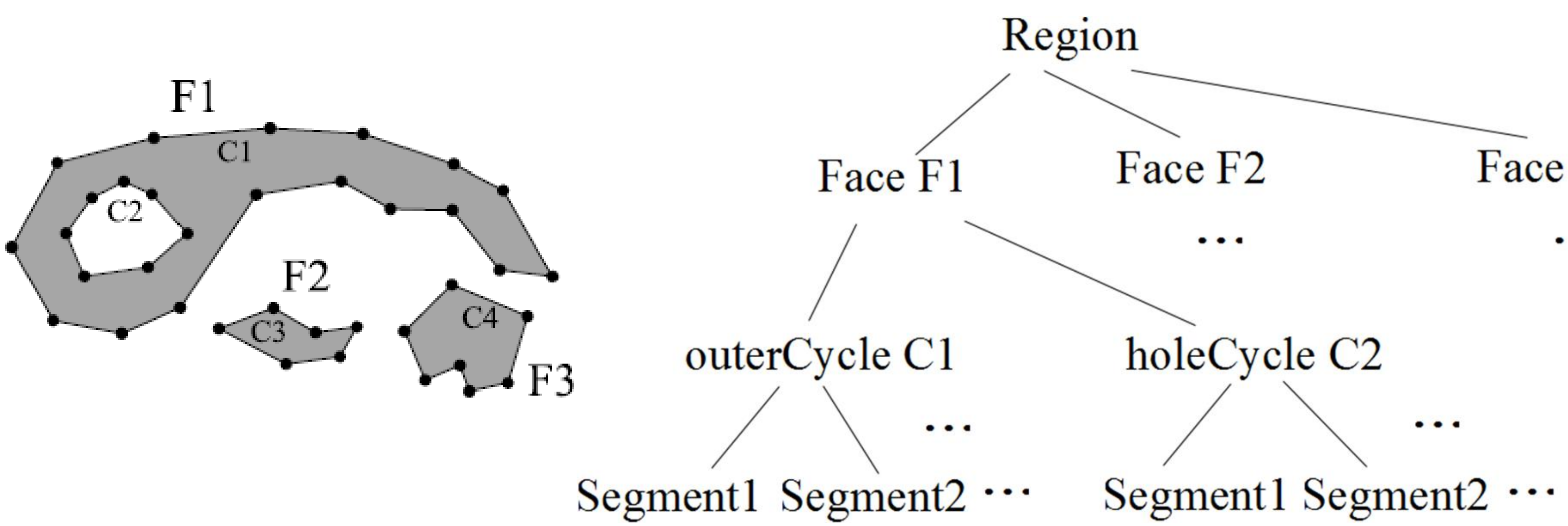
- Big Data
  - Loosely defined term that refers to extremely large, diverse, and distributed sets of alphanumerical and simply structured data that are difficult, or even impossible, to capture, store, manage, query, and analyze with conventional database management techniques and tools
  - For example, generated as raw data from sensing devices like instruments, sensors, satellites, mobile devices, cameras, microphones, radio-frequency identification readers, etc.
  - Representation and processing strategy: distributed, decentralized
- Big Objects
  - Individual large, highly structured, complex (application) objects of varying representation size that are handled as monolithic, self-contained entities (or objects in the object-oriented sense)
  - Complex objects are implemented as values of abstract data types (ADTs)
  - For example, images, videos, multimedia documents, hurricanes, genes, enzymes, proteins, spatial objects (point, line, region, map)
  - Representation and processing strategy: compact, centralized

Complex Objects (II)

- Example: A book



- Example 2: A spatial region object

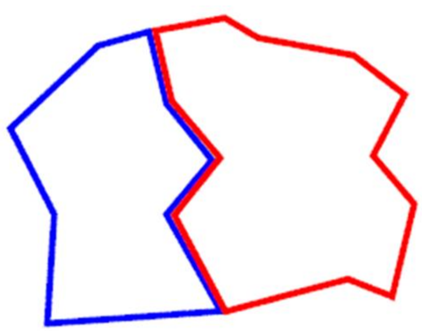


Complex Objects (III)

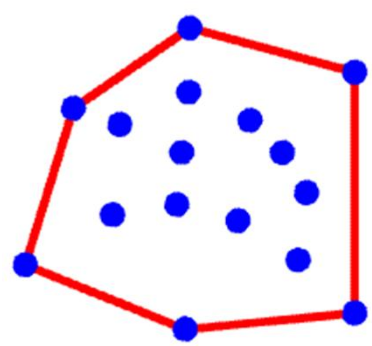
- The concepts we have learned so far make it very difficult, or even impossible, to adequately represent complex objects
- Two main reasons
  - SQL3 does not support *pure* ADTs in SQL: The internal structure is always visible and accessible (ADTs only allows access by means of operations)
  - All objects one can construct with SQL3 constructs have a relatively small and limited representation size (but videos, e.g., can have a size of several GB)
- Many ORDBMs provide the user with *vendor-specific* packages of ADTs for special purposes, e.g.,
  - Informix DataBlades for text, images, video, spatial, spatiotemporal, web, and time series data
  - Oracle Data Cartridges for multimedia, text, image, audio, video, spatial (Oracle Spatial), legal, and medical data
  - DB2 Extenders in DB2 for text, spatial, net search, and XML data
- No or very limited standardization for these extension packages

Complex Objects (IV)

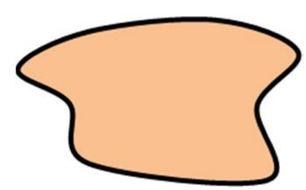
- Example: Some geometric operations on the spatial data types *point*, *line*, and *region* that are implemented as ADTs whose internal structure is hidden
  - *distance*: $\alpha \times \beta \rightarrow$ *double* for $\alpha, \beta \in$ {*point*, *line*, *region*}
  - *meet, overlap*: $\alpha \times \beta \rightarrow$ *bool* for $\alpha, \beta \in$ {*line*, *region*}
  - *inside*: $\alpha \times$ *region* $\rightarrow$ *bool* for $\alpha \in$ {*point*, *line*, *region*}
  - *convexHull*: *point* $\rightarrow$ *region*
  - *intersection*: $\alpha \times \beta \rightarrow$ *line* for $\alpha, \beta \in$ {*line*, *region*}, $\alpha \neq \beta$
  - *mbb*: $\alpha \rightarrow$ *region* for $\alpha \in$ {*point*, *line*, *region*}
  - *intersection*: *region* $\times$ *region* $\rightarrow$ *region*



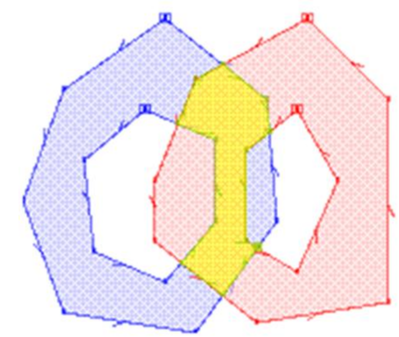
Meet predicate yields *true* for two *region* objects



Convex hull of a *point* object



Minimum bounding box of a *region* object



Intersection of two *region* objects

Complex Objects (V)

- Query example with spatial data types and spatial operations

  Consider the map of the 50 states of the USA. Each state has besides its thematic attributes like name and population also a geometry which is described by its region. A region can have holes (like enclaves) and consist of several components (like mainland and islands). Cities can, e.g., be represented as points indicating their location.

  We create two tables:

  create table *states*(sname **varchar**(30), spop **integer**, territory **region**);

  create table *cities* (cname **varchar**(20), cpop **integer**, loc **point**);

  The spatial data types *point* and *region* are in the same way attribute data types as the types *string* and *integer*

  Query: Determine the names of all cities and states where a city is located in a state; this can then be formulated as a so-called spatial join:

  select cname, sname
  from cities, states
  where loc inside territory

Complex Objects (VI)

- Query example with spatial data types and spatial operations (*continued*)

  The term *inside* is a so-called topological predicate testing whether a point is geometrically located inside a region

  Query: List the names and area of all states ordered by their geometric extent:

  select sname, area(territory) **as** sarea
  from states
  order by area(territory) **desc**;

  The operator *area*: *region* $\rightarrow$ *double* is a geometric function taking a region object as an argument and yielding a numerical value, namely the area of this object, as a result (area here in square miles)

sname	sarea	sname	sarea	sname	sarea	sname	sarea	sname	sarea
Alaska	665,384.04	Michigan	96,713.51	Missouri	69,706.99	Louisiana	52,378.13	West Virginia	24,230.04
Texas	268,596.46	Minnesota	86,935.83	Florida	65,757.70	Mississippi	48,431.78	Maryland	12,405.93
California	163,694.74	Utah	84,896.88	Wisconsin	65,496.38	Pennsylvania	46,054.35	Hawaii	10,931.72
Montana	147,039.71	Idaho	83,568.95	Georgia	59,425.15	Ohio	44,825.58	Massachusetts	10,554.39
New Mexico	121,590.30	Kansas	82,278.36	Illinois	57,913.55	Virginia	42,774.93	Vermont	9,616.36
Arizona	113,990.30	Nebraska	77,347.81	Iowa	56,272.81	Tennessee	42,144.25	New Hampshire	9,349.16
Nevada	110,571.82	South Dakota	77,115.68	New York	54,554.98	Kentucky	40,407.80	New Jersey	8,722.58
Colorado	104,093.67	Washington	71,297.95	North Carolina	53,819.16	Indiana	36,419.55	Connecticut	5,543.41
Oregon	98,378.54	North Dakota	70,698.32	Arkansas	53,178.55	Maine	35,379.74	Delaware	2,488.72
Wyoming	97,813.01	Oklahoma	69,898.87	Alabama	52,420.07	South Carolina	32,020.49	Rhode Island	1,544.89

Complex Objects (VII)

- Example: Some operations for processing images
  - Data type *image* offered with a large variety of standard storage formats, e.g., support of the formats tiff, gif, jpeg, photoCD, group 4, fax
  - Some operations

    rotate: $image \times angle \rightarrow image$

    crop: $image \times polygon \rightarrow image$

    flip: $image \times image \rightarrow image$

    minus: $image \times image \rightarrow image$

    intersection: $image \times image \rightarrow image$

    union: $image \times image \rightarrow image$

    superimposition: $image \times image \rightarrow image$

Question 1: Let $R(A, B, C, D)$ be a relation schema in 1NF. Let AB be the primary key (PK) of R. Determine a set of FDs so that R is not in 2NF.

Solution: Consider the set of FD: $AB \rightarrow CD$ and $B \rightarrow C$. The non-prime attribute C is not fully functionally dependent on the primary key AB. Hence, it is not in 2NF.

Question 2: Let $R(A, B, C, D)$ be a relation schema in 2NF. Let AB be the primary key (PK) of R. Determine a set of FDs so that R is not in 3NF.

Solution: Consider the set of FD: $AB \rightarrow CD$ and $C \rightarrow D$. We observe:

1. AB is a superkey since $AB \rightarrow R$.
2. AB is a primary key since $A^+ \neq R$ and $B^+ \neq R$.
3. R is in 2NF since the non-prime attributes C and D are fully functionally dependent on the PK AB.
4. R violates the 3NF because
   a) $\{D\} \subseteq \{C\}$ is false, that is, $C \rightarrow D$ is not a trivial FD
   b) The statement "C is a superkey" is false.
   c) The statement "D is part of some candidate key" is false.

Question 3: Consider the relation schema $R(A, B, C)$, which has the FD $B \rightarrow C$. If A is a candidate key for R, is it possible for R to be in BCNF? If so, under what conditions? If not, explain why not.

Solution: The only way R could be in BCNF is if B includes a candidate key, i.e., B has to be a candidate key for R.

Question 4: Consider the relation schema $R(A, B, C, D, E, F, G, H)$ with the FDs $F = \{BC \rightarrow AD, E \rightarrow FH, F \rightarrow GH\}$. Decompose R into 3NF, and list each step of the process. Is the decomposition also in BCNF?

Solution: Step 1: Check whether R is already in 3NF.

Observations:

1. Determine the candidate keys of schema R.

Left side	Both sides	Right side
B, C, E	F	A, D, G, H

BCE is the (only) candidate key of R since $BCE^+ = R$. We begin with $BCE^+ = BCE$. With $BC \rightarrow AD$ we get $BCE^+ = ABCDE$. With $E \rightarrow FH$ we get $BCE^+ = ABCDEFH$. With $F \rightarrow GH$ we get $BCE^+ = ABCDEFGH = R$.

Question 4: Consider the relation schema $R(A, B, C, D, E, F, G, H)$ with the FDs $F = \{BC \rightarrow AD, E \rightarrow FH, F \rightarrow GH\}$. Decompose R into 3NF, and list each step of the process. Is the decomposition also in BCNF?

Solution: Step 1 (*continued*): Check whether R is already in 3NF. Observations:

2. Schema R even violates 2NF since the non-prime attributes A, D, F, and H are only partially dependent on BCE.

3. Schema R violates 3NF since, for example, in $F \rightarrow GH$, F is not a superkey and $G, H \notin \{B, C, E\}$ holds.

Conclusion: 3NF synthesis algorithm has to be applied.

Question 4 (*continued*): Consider the relation schema $R(A, B, C, D, E, F, G, H)$ with the FDs $F = \{BC \rightarrow AD, E \rightarrow FH, F \rightarrow GH\}$. Decompose R into 3NF, and list each step of the process. Is the decomposition also in BCNF?

Solution: Step 2: Find a minimum cover F_c for F (left reduction, right reduction, application of the union rule if possible). Observations:

1. FD $BC \rightarrow AD$: No left reduction possible since this is the only FD that uses B and C, and since $B^+ = B$ and $C^+ = C$.
2. FD $E \rightarrow FH$: No left reduction possible since E is a single attribute set. Right reduction to $E \rightarrow F$ possible since $E \rightarrow H$ can be obtained by transitivity from $E \rightarrow F$ and $F \rightarrow H$.

Question 4 (*continued*): Consider the relation schema $R(A, B, C, D, E, F, G, H)$ with the FDs $F = \{BC \rightarrow AD, E \rightarrow FH, F \rightarrow GH\}$. Decompose R into 3NF, and list each step of the process. Is the decomposition also in BCNF?

Solution: Step 2 (*continued*): Find a minimum cover F_c for F (cont.) Observations:

3. FD $F \rightarrow GH$: No left reduction possible since F is a single attribute set. No right reduction possible since $F \rightarrow G$ and $F \rightarrow H$ cannot be derived otherwise.

We get $F_c = \{BC \rightarrow AD, E \rightarrow F, F \rightarrow GH\}$.

Question 4 (*continued*): Consider the relation schema $R(A, B, C, D, E, F, G, H)$ with the FDs $F = \{BC \rightarrow AD, E \rightarrow FH, F \rightarrow GH\}$. Decompose R into 3NF, and list each step of the process. Is the decomposition also in BCNF?

Solution: Step 3: Synthesis of decomposition schemas

$R_1(A, B, C, D)$, $R_2(E, F)$, $R_3(F, G, H)$

Solution: Step 4: Check if relation schemas are contained in other relation schemas. If so, delete the smaller ones in order to avoid redundancy.

This is not the case here.

Question 4 (*continued*): Consider the relation schema $R(A, B, C, D, E, F, G, H)$ with the FDs $F = \{BC \rightarrow AD, E \rightarrow FH, F \rightarrow GH\}$. Decompose R into 3NF, and list each step of the process. Is the decomposition also in BCNF?

Solution: Step 5: Check whether a schema contains a candidate key. If not, create an extra schema containing a candidate key to ensure lossless join decomposition.

BCE is not contained in any schema. Add $R_4(B, C, E)$.

We obtain: $R_1(A, B, C, D)$, $R_2(E, F)$, $R_3(F, G, H)$, $R_4(B, C, E)$.

Question 4 (*continued*): Consider the relation schema $R(A, B, C, D, E, F, G, H)$ with the FDs $F = \{BC \rightarrow AD, E \rightarrow FH, F \rightarrow GH\}$. Decompose R into 3NF, and list each step of the process. Is the decomposition also in BCNF?

Solution: Step 6: Assignment of FDs to schemas

We obtain:

$\{(R_1(A, B, C, D), \{BC \rightarrow AD\}), (R_2(E, F), \{E \rightarrow F\}),$
$(R_3(F, G, H), \{F \rightarrow GH\}), (R_4(B, C, E), \varnothing)\}$

Solution: Step 7: Check whether the schemas are in BCNF.

Yes, the left sides of the FDs are superkeys of the respective schemas.