

**Spatial Informatics**  
**Prof. Soumya K. Ghosh**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 22**  
**Spatial Networks - II**

Hello. So, we will continue our discussion on Spatial Networks. So, in the last lecture, we started we introduce this topic on spatial networks and try to see that what is the specialty of the things? We will continue that discussion and see how what are what is the importance of the spatial networks and how different network analysis can be done how queries can be done on spatial networks?

So, if you if you recollect we discuss that why we require a separate topic for spatial network; because, this has different property or different way of dealing. So, two points connected by a network is the path need to be travelled in the same in the following the network itself right I say that the point A and B connected by this road network.

So, if you want to go from A to B you need to follow this road network in doing. So, right, there is not the Euclidean distance between the A to B may not be the say nearest a that is the specific distance. Or finding a nearest point between two a from a particular node to other things. So, you need to follow the network.

So, that is why its analysis goes in a different way not only that we have seen that we can represent this in the form of a graph. And all this graph algo etcetera come into play this we can use those for doing different analysis and as it says that it can be directed or undirected graph even that the overall this graph property may or the network property may change over time.

Like, as we discuss that like same during the same day some portion of a network or segment of network. Maybe sometimes one way from South to North means and it is reverse from some other time from North to South based on the traffic management policy and the traffic flow policies right.

So, that based on the whatever the transportation department or traffic management department. In other sense if you look at the whole thing as a network it goes on it changes over time right. And of course, the there are different way of modeling. One

typical factors which we are used to that minimum distance or minimum time for travel finding a shortest path.

It may be also with taking consideration other type of things like conjunction level of the network or type of road if you are talking about the road and type of things. So, those things can be there right. So, with all those things we will try to see today that what are the different aspects of these what are the different properties how it can be represented, how to queries type of I means on this spatial networks.

(Refer Slide Time: 03:28)

**Spatial Network Data Models**

- Three level Database Design
  - Conceptual Data Model
    - Graphs
  - Logical Data Model -
    - Data types - Graph, Vertex, Edge, Path, ...
    - Operations - Connected(), Shortest\_Path(), ...
  - Physical Data Model
    - Record and file representation - adjacency list
    - File-structures and access methods - CCAM (*Connectivity-Clustered Access Method*)

[Ref: Spatial Databases: A Tour, by Shashi Shekhar, Sanjay Chawla; Internet resources]

The slide features a yellow background with a blue header and footer. The footer contains logos for the Indian Institute of Space Science and Technology (IIST), the National Institute of Advanced Industrial Science and Technology (NIAIST), and the Swayam logo.

So, if you look at the database design part. So, typically any database design have three aspects right it is true for here also. So, one is the conceptual data model right, in this case it is considered as a graph a graph  $G$  of vertex  $V$  and edges  $E$   $G = (V, E)$  so, graph.

Now, this is the conceptual data model then we have a logical data model, which has data types like support out data types like graph, vertex, edge, path these are logical data models right logical. Or I even it can there are different operations are supported like finding the shortest path or connected or not right there can be this finding there are several operations.

So, one we have the conceptual model which is conceptually a graph. Then we have a logical model which is the data types and operations which can act on this data types type of things. So, this is a logical mode then we have a physical data model, which deals

with record and file representation. Say one way of representing graphics adjacent symmetric is a one way or adjacency list.

So, based on that I can represent these things. So, this is physically how things are there. So, file structure and there can be different access method. Like one CCAM connected cluster access method and there can be different access method of accessing this graph and type of things right. So, so both conceptual logical and physical the standard data models are valid for this sort of when we dealing with spatial networks.

Now, if you recollect when we when we talked about spatial models and other type of things so, not exactly we are dealing with like that right. A spatial data sets we are dealing with directly to the graph etcetera because those are not fitting into the things right. So, that is why if you see it requires a spatial type of coat uncoat representation and treatment to work on the things and we have lot of applications, which over surround this sort of networks this sort of spatial networks.

(Refer Slide Time: 05:58)

**Conceptual Data Models**

- Conceptual Data Model for Spatial Networks
  - A graph,  $G = (V, E)$
  - $V$  = a finite set of vertices
  - $E$  = a set of edges  $E$ , between vertices in  $V$
- Example: two graph models of a roadmap
  - Nodes = road-intersections, edges = road segment between intersections [Model-1]
  - Nodes = roads (e.g. Route 66), edge(A, B) = road A crosses road B [Model-2]
- Classifying graph models
  - Do nodes represent spatial points? - Spatial vs. Abstract graphs
  - Are vertex-pair in an edge order? - Directed vs. Undirected
- Example (continued)
  - Model 1 is a Spatial Graph, Model 2 is an Abstract Graph
  - Can be directed or undirected

[Ref: Spatial Databases: A Tour, by Shashi Shekhar, Sanjay Chawla; Internet resources]

The slide includes a diagram of a road intersection with labels A, B, and C, and a small inset image of a person speaking.

So, conceptual data models for spatial network if you see it is a graph  $G$  equal to  $V, E$  where  $V$  is a finite set of vertices.  $E$  is the set of edges between vertices in  $V$ . So, I have finite set of vertices in  $V$  set  $V$  and then I have a set of edges  $E$  and we have this which means between vertices which are represented in  $V$  right. So, this is the generic definition of graph nothing extra for this spatial thing.

So, there are two graph models for a typical road network or I should we can generalize as a typical transportation network like, nodes road intersection. Suppose, I consider a road network so, road intersects are the your nodes and edges the roads segment interconnecting the things. So, I can say this is one of the one way one way this road can be model. I can other way model that roads for example, particular road in this case 66 or anything.

Roads is the edge and edges is the road A crosses B right, this can be the way of handling the things right. So, that is same now the same road network I can represent in a both way right. Like if I look at this representation like say I have a road networks right road network like this right.

So, I have a this one way is that, I have these are my nodes, I represent that intersection on the nodes like say X Y. So, these are my nodes and the edge connectings like if it is edge E 1 is the edge of the connecting the nodes right. Other way other see I can say this is my road A; this is my road B right or I i say this is if the; it is continuity.

So, instead of E I can say this is road A and I say that X Y in the model-1, X Y is the nodes and A is the edge connecting the thing right. Here I say A, B are the roads now, I define the edge at A crosses B right that can be the edge right. This why I require this because it may be important for me that finding this crossing and doing some analysis on the basis of the things right.

The other representation may be important for me to handle the things right. So, it based on that type of things I require I can have different type of representation into the thing right. So, now, if I want to classify this graph models, do not represent spatial points right. So, it is spatial versus abstract graph right. So, whether nodes represent the spatial points so, in the second thing right see the nodes are basically not points right it is the road itself.

So, the how this is important to understand that how I represent the graph is matter that based on the type of application, I am looking for all right or type of way I am want to handle the things right. So, it may represent a spatial points. So, it can that a node is node itself or it can represent something else like the road I represent as the node right.

Or vertex pair is an edge order alright whether there is a directed or undirected graph right, if it is a ordered then it is a directed or undirected graph. Now, model-1 is typically a spatial graph right, where the node represent a spatial point A is the connection between the two nodes right. And model-2 is a abstract graph right, where you see the typically that node which is road A; is represented by one node.

Road B another node and edge A B are A crosses B is considered as the crossing is considered as the connection between the things. That is why I construct the road right. So, I have all this road then I have to find which road crosses how many like I want to count that given a road how many crossings are there right. Then it may help this sort of representation other way other way to find out that oh where are the crossings etcetera is a big tedious job to do that alright.

So, it is important that how are you are going to represent the thing not only that, even in the model a I say that A connects B or X connects Y by intermediate road is A some road R or R 1 or a b c anything right. So, this does not mean when I have the connection I may not represent as the actual physical things like that.

(Refer Slide Time: 11:36)

**Conceptual Data Models**

- Conceptual Data Model for Spatial Networks
  - A graph,  $G = (V, E)$
  - $V$  = a finite set of vertices
  - $E$  = a set of edges  $E$ , between vertices in  $V$
- Example: two graph models of a roadmap
  - Nodes = road-intersections, edges = road segment between intersections [Model-1]
  - Nodes = roads (e.g. Route 66), edge(A, B) = road A crosses road B [Model-2]
- Classifying graph models
  - Do nodes represent spatial points? - Spatial vs. Abstract graphs
  - Are vertex-pair in an edge order? - Directed vs. Undirected
- Example (continued)
  - Model 1 is a Spatial Graph, Model 2 is an Abstract Graph
  - Can be directed or undirected

[Ref: Spatial Databases: A Tour, by Shashi Shekhar, Sanjay Chawla; Internet resources]

Like I into say that I have two node these are A and B. And I have a edge this is like the one edge right, but when I draw as a graph then I represent them is a this is A B and it is a connected by A. Suppose this is this name of this road is R. So, it is connected by road R type of things right.

Now, see here it may not be representing the thing the how it is specially distributed or how the specially things are there. Those things I can handle by the properties of this edge. Like, I can say the length of the graph or different type of properties; I can have like type of the length of the edge type of edge like if it is a road that type of road. Even I can have like as we discuss that conjunction level and etcetera etcetera.

(Refer Slide Time: 12:49)

**Logical Data Model - Data types**

- Common Data Types
  - Vertex: attributes are label, isVisited, (location for spatial graphs)
  - DirectedEdge : attributes are start node, end node, label
  - Graph : attributes are setOfVertices, setOfDirectedEdges, ...
  - Path : attributes are sequenceOfVertices
- Above data types can be used to model -
  - An undirected edge
  - Train routes in rail network
  - Rivers in the river network
  - Note: Multiple distinct solutions are possible in last two cases.

Now, logical data model data types or common data types what is there it is there vertex attributes are labeled like is visited locations for spatial graph these are the things. So, it is the vertex is there directed edge; the attributes are start node end node with the label. So, there is a have a directed edge from start node and end node is different from this to this I said directed edge like this.

Graph, attributes a set of vertices set of directed edges and so and so forth. So, the graph attributes of attributes are set of vertices set of directed edges makes the graph and path attributes of a sequence of vertices. So, when I say that a path between A and B then a then sequence of vertices to B that whole sequence of vertices is gives me the path right.

And above data types we can use to model undirected graph; that means both way connections are there or a rail routes for a rail network or a rivers in a river network. So, multiple distinct solution are possible when we want to represent rail routes and river network and type of things; that means, we can represent those things in a different manners.

(Refer Slide Time: 14:09)

**Logical Data Model - Operations**

- Low level operations on a Graph G
  - IsDirected - return true if and only if G is directed
  - Add , AddEdge - adds a given vertex, edge to G
  - Delete, DeleteEdge - removes specifies node (and related edges), edge from G
  - Get, GetEdge - return label of given vertex, edge
  - Get-a-successor, GetPredecessors - return start or end vertex of an edge
  - GetSuccessors - return end vertices of all edge starting at a given vertex
- Building blocks for queries
  - shortest\_path(vertex start, vertex end)
  - shortest\_tour(vertex start, vertex end, setOfVertices stops)
  - locate\_nearest\_server(vertex client, setOfVertices servers)
  - allocate(setOfVertices servers)

Now, logical data model operations right. So, there can be different type of operations. So, one is directed may be operation return a value if and only if G is directed right. Return true return true when if and only if G is a directed graph right, AddEdge or add adds a given vertex edge to the G.

So, I can add say new road generated a new path is generated a; is construct I mean I can add a thing. Or even I can delete or delete an edge removes the specific node or an edge from the graph. So, there can be different representation so, GetEdge return level of a given vertex and edge right. So, Get-a-successor GetPredecessor return start or end vertex of an edge right.

So, I can have Get-a-successor GetPredecessors type of or get of series of successor etcetera right. Like I want to like as we can see that if there is a directed graph then if there is a conjunction then, I want to series of successors where the things are there where the effect will be there right. Or series of a predecessors which road will be affected if this is congested right.

Or like spilling spill in some river then, which are the downstream tributaries and distributors which will be affected, if which are the different rivers should be affected and by these things right. And GetSuccessors return the return the edge returned vertices of all the edges starting from a given vertex right.

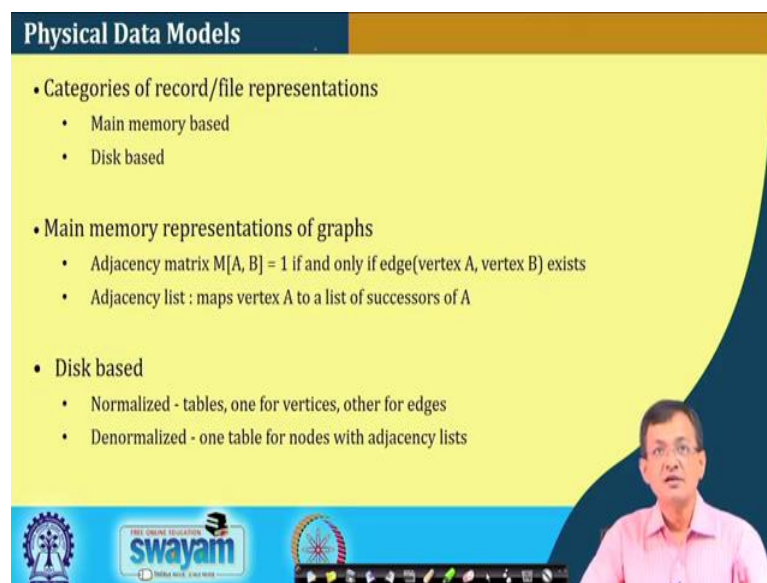


So, it is I can have different successors right. So, predecessor successor, immediate successor so, these are the things where you require those type of applications. So, this see if we if we if we little bit look into the more into emphasis on the graph. So, these are problems which are there in the graph analysis rather these are they are all goes which are already the establish algos, which are there and we can use those algos into this type of analysis right.

So, building block of a query short path shortest path maybe one type of things; shortest tour can be another thing if I say path between start and end vertex. Then the tour may be the intermediate set of vertices to the stops etcetera. Locate a nearest server if we require that give a thing find the nearest server allocate the a set of vertices and servers, there are different type of building blocks for queries which can be executed.

So, this so what we try to see that logical data model there can be different operations into the logical data models to build the model.

(Refer Slide Time: 17:30)



The slide is titled "Physical Data Models" and is divided into two main sections. The first section, "Categories of record/file representations", lists "Main memory based" and "Disk based". The second section, "Main memory representations of graphs", lists "Adjacency matrix  $M[A, B] = 1$  if and only if edge(vertex A, vertex B) exists" and "Adjacency list : maps vertex A to a list of successors of A". A third section, "Disk based", lists "Normalized - tables, one for vertices, other for edges" and "Denormalized - one table for nodes with adjacency lists". The slide features a yellow background with a blue header and footer. The footer includes logos for "swayam" and "INDIA RISE, COUNTRY RISE" along with a small image of a person in the bottom right corner.

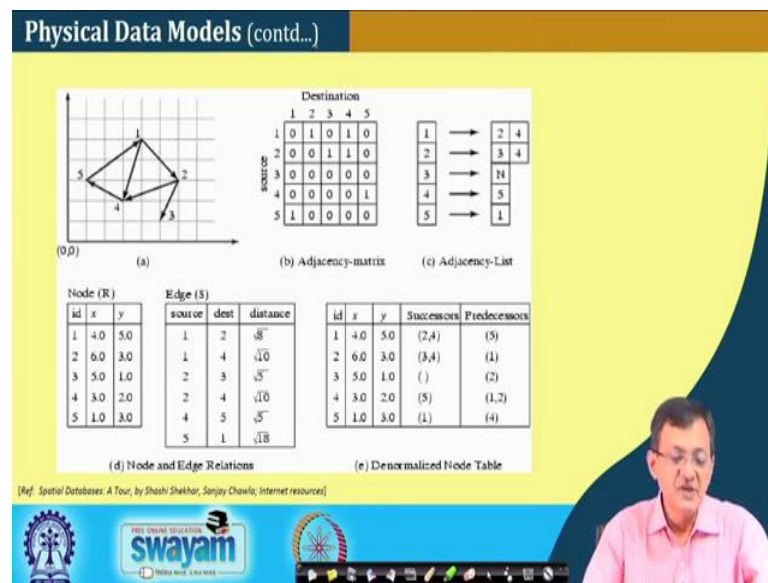
- Categories of record/file representations
  - Main memory based
  - Disk based
- Main memory representations of graphs
  - Adjacency matrix  $M[A, B] = 1$  if and only if edge(vertex A, vertex B) exists
  - Adjacency list : maps vertex A to a list of successors of A
- Disk based
  - Normalized - tables, one for vertices, other for edges
  - Denormalized - one table for nodes with adjacency lists

Now if you look at the physical data models that is more over records and files. So, one is the main memory based another is the disk base so that is broadly. So, one popular structure for storing graph is adjacency matrix  $M A B$  equal to 1, if and only if edge from A to B exist right. So, I say that my adjacency matrix if there is a edge from A to B then I say a that in the matrix A to B is one that is already known to you.



Adjacency list maps vertex A to a list of successors of the A from the A; we have a list of successor this is. So, again this it can there are these are categorization based on this disk base. So, normalize table one of one for vertex one other for edges right; and denormalized one table for nodes and adjacency list.

(Refer Slide Time: 18:32)



Now the same thing, now if you want to represent right, so I have a graph like this or it can be road network like 1 is connected to 2; 1 from to 4, 4 to 5; 5 to 1 2 to 3 and type of things it is a typically road network right. So, it is a graphically representation actually road may be the physical represent or physical what we say spatial representation of the road maybe some fashion but these are the connectivity things.

Now, if you look away adjacency matrix as we know that 1 to 2, there is 1 1 to 4 is 1 that is there is there are edges between 1 to 2 and 1 to 4 but there is no edges between 1 to 5, because there is a directed graph right. So, this is way I can represent the adjacency matrix like 4 to 5, the edge is there but 4 to nowhere other edges are there.

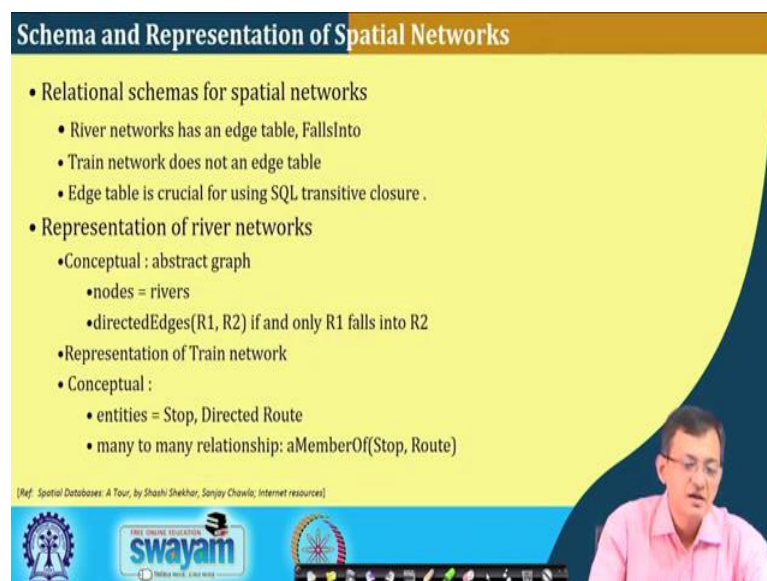
So, 4 to 5 it is 1; like here 4 to 5 it is 1 and whereas, I do not have any other edges there. Similarly, from 5 to 1 that is a edge right, but that is no edge from 5 to any other places right. So, this way I have the adjacency matrix and similarly adjacency reach 1 has the successor 2 and 4, 2 has the successor 3 and 4 and similarly 5 has 1 and type of things and for 3 there is no successor.

So, it is null in that is no successors so, there is adjacency list matrix. So, this is one way of representing, if we have a if you see the normalize and denorm type of a table. So, we have these node edge like this is a node table and edge table.

So, it is normalized this node table contains that that coordinates that x y coordinates of the all the nodes. So, node 1 4 5 node 2 6 3 and so and so forth and edge 1 to 2, the basically in this case our edge matrix is a distance matrix. So, 1 to 2 the distance so that it find out that Euclidian distance in this case right. So, similarly all the distances, I can have a denorm node table where location successor list and predecessor list.

So, 1 has successor has 4, 1 has successor has 2 and 4 are the successor once predecessor is the 5 so, this also the denorm table right. So, in some cases for analyzing the denorm table is easy means can give can retrieve the informations in a efficient way. Nevertheless, for storing the data I may have the normalized table and type of things. Nevertheless, we can have the representation in the both the way.

(Refer Slide Time: 21:34)



### Schema and Representation of Spatial Networks

- Relational schemas for spatial networks
  - River networks has an edge table, FallsInto
  - Train network does not an edge table
  - Edge table is crucial for using SQL transitive closure .
- Representation of river networks
  - Conceptual : abstract graph
    - nodes = rivers
    - $\text{directedEdges}(R1, R2)$  if and only if  $R1$  falls into  $R2$
  - Representation of Train network
  - Conceptual :
    - entities = Stop, Directed Route
    - many to many relationship:  $\text{aMemberOf}(\text{Stop}, \text{Route})$

[Ref: Spatial Databases: A Tour, by Shashi Shekhar, Sanjay Chawla; Internet resources]

So, if you look at the schema and other representation of this relational schema for spatial network. So, if you consider the river network has an edge table like we will see that that is which is which are the edges and type of things and there is a falls into like river 1 falls into river 6; and 6 falls into something right so that is a edge table of falls into right.

It represent that to which one false into what right train network does not have any edge table per se. Because there is no as such we do not have any directed information. Edge table is a crucial for using SQL transitive closure so, that is closure because the edge table gives me A falls into B, B falls into C. So, A false into C.

So, if I in a; in a river A falls into river B, river B falls into river D then I can say river a falls into river D right, this has the implications right. So, I want to D, I can say there is a connectivity right it can be via B or something that is say, but there is a by this transitive closure I can say you can go from port A to port B right.

And secondly, other things what we are discussing or particularly river other it is crossing that if there is a spill in river A, which are the rivers going to affect then I say d is going to affect because its falls into this right. So, this is a another example. So, representation of river conceptually it is a abstract graph we represent a abstract graph, where rivers are nodes and directed edges R 1, R 2 if R 1 falls into R 2 right.

So, this is way I we want to represent so rivers are nodes and I say that there is a edge R 1 R 2 if R 1 falls into R 2right. So, representation of the train network conceptual entities stop directed route many to many relationship a member of stop route; right. So, they are many to many relationship is there right. So, this is for the thing.

(Refer Slide Time: 23:49)

**Query Languages For Graphs**

- Recall Relation algebra (RA) based languages
  - Can not compute transitive closure
  - SQL3 provides support for transitive closure on graphs
    - supports shortest paths
- SQL support for graph queries
  - SQL2 - CONNECT clause in SELECT statement
    - For directed acyclic graphs, e.g. hierarchies
  - SQL 3 - WITH RECURSIVE statement
    - Transitive closure on general graphs
  - SQL 3 -user defined data types
    - Can include shortest path operation

The slide features a yellow background with a blue header and footer. A video inset in the bottom right corner shows a man in a pink shirt speaking. The footer includes the Swayam logo and a navigation bar.

So, if you recollect is relational algebra, one of the means basically the building block for our query language. So, cannot compute transitive closure per se right relational algebra there is no transitive closure per se SQL 3 though provide support for transitive closure on graphs right, support shortest path. So, it allow it helps you in finding the transitive closure in graph also it suppose the shortest path.

So, SQL support for graph queries SQL 2 connect clause to select in a select statement. So, there is a connect clause which comes into the select statement we will see that for directed acyclic graph that is hierarchies right. So, if there is a directed acyclic SQL 3, with recursive statement we say there is a recursive statement. So, recursively I have to find out that what are the say successor predecessors and type of things.

So, transitive closure on general graphs and SQL 3 user defined data types can include shortest path operation etcetera. So, we can have a user defined data types also. So, what we see if we see SQL 2 connect with a select statement. SQL 3 with a recursive statements so, recursively I can find out successor predecessors and type of things. And SQL 3 we can have user defined data types like even shortest path operation it said that can be defined as a user defined class.

(Refer Slide Time: 25:35)

**Concept of Transitive Closure**

- Consider a graph  $G = (V, E)$
- Let  $G^* =$  Transitive closure of  $G$
- Then  $T = \text{graph}(V^*, E^*)$ , where
  - $V^* = V$
  - $(A, B) \in E^*$  if and only if there is a path from  $A$  to  $B$  in  $G$ .
- Example in Figure
  - $G$  has 5 nodes and 5 edges
  - $G^*$  has 5 nodes and 9 edges
  - Note edge  $(1,4)$  in  $G^*$  for
    - path  $(1, 2, 3, 4)$  in  $G$ .

[Ref: Spatial Databases: A Tour, by Shashi Shekhar, Sanjay Chawla; Internet resources]

(a) Graph  $G$

(c) Transitive closure  $(G) = \text{Graph } G^*$

(b) Relation  $R$

SOURCE	DEST
1	2
1	5
2	3
3	4
5	3

(d) Transitive closure in relational form

SOURCE	DEST
1	2
1	5
2	3
3	4
5	3
1	3
2	4
5	4
1	4

So, now again we are coming to the concept of transitive closure. So, if you see consider the graph  $G = (V, E)$  so,  $G^*$  equal to transitive closure of  $G$ . So, you consider a  $G = (V, E)$   $G^*$  star is the transitive closure of  $G$  then  $T$  equal to graph  $V^* = E^*$  where  $V^* = V$

to  $V$  and  $A B$  in  $E$  star. If and only if there is a path between  $A B$  in  $G$  right so; that means, what we are trying to look at that given a graph whether I can find out this transitive closure of this graph.

So,  $A$  implies  $B$ ,  $B$  implies  $C$ ,  $C$  implies  $P$ . So, implies  $p$  if there is a creating a edge between that because there is a there is possible to travel in that edge right. So, once I have the transitive closure then finding out that different type of queries will be much easier because I can have this lookup table and say that which is connected or not right.

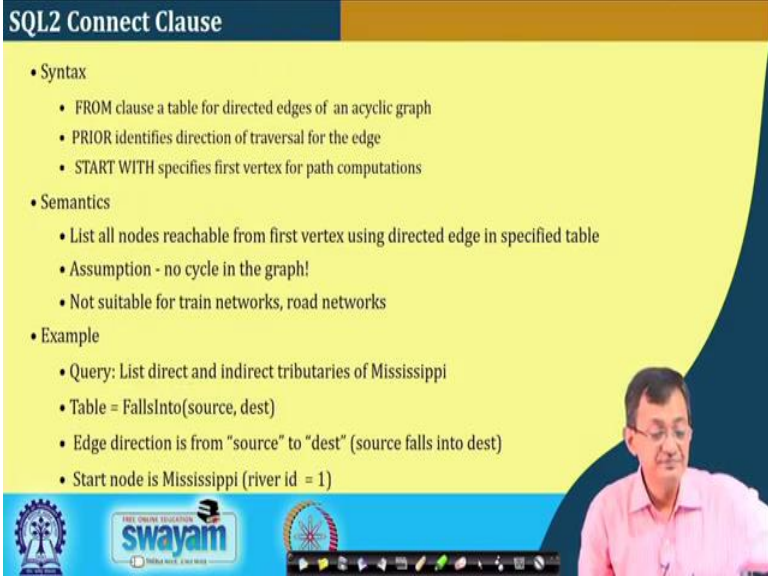
So, example in this particular figure we have that graph  $G$  at the top right. 1 is connected to 2, 2 to 3, 1 to 5, 5 to 3 and 3 to 4 right and this is given 1 is connected to 2, 2 is connected to 1 is connected to 5, 2 is connected to 3, 3 is connected to 4 and 5 is connected to 3 right. So, source 5 and 3 so 5 to 3. So, 1 edge, 2 edge third edge is 2 to 3, 3 to 3 1 to these 1 to 5, 2 to 3 then we have 3 to 4 and 5 to 3 right.

Now, if I want to find out a transitive closure of the things then I say I can say that 1 to 2, 2 to 3. So, 1 to 3 one edge will be there right, 1 to 3 edge will be there. Similarly 2 to 4 edge will be there because there are 2 to 3, 3 to 4. So, 2 to 4 will be there again 3 to 4 there is nothing is there and but 1 to 5, 5 to 3. So, 1 to 3 edge will be there if it is not already there.

So, similarly we can see that different type of 5 to 4, 1 to 4 and these are different edges are there. So, this is a transitive closure in relation form right. So, you have the transitive closure into the things. Now, I can find out you give a given a node if you want to other is so there is a transitive closure so; that means, there is a way to reach that.

Now you need to look around that which mode of travel and whether it is available and type of things right. So, this is a very one of the important aspect to find out the transitive closure, but if again if we fall back to our standard data base system. So, these things are not there.

(Refer Slide Time: 28:31)



### SQL2 Connect Clause

- Syntax
  - FROM clause a table for directed edges of an acyclic graph
  - PRIOR identifies direction of traversal for the edge
  - START WITH specifies first vertex for path computations
- Semantics
  - List all nodes reachable from first vertex using directed edge in specified table
  - Assumption - no cycle in the graph!
  - Not suitable for train networks, road networks
- Example
  - Query: List direct and indirect tributaries of Mississippi
  - Table = FallsInto(source, dest)
  - Edge direction is from "source" to "dest" (source falls into dest)
  - Start node is Mississippi (river id = 1)

So, if you look at the SQL 2 or the connect clause for graphs. So, there are some syntax from clause a table for directed edge of an acyclic graph right. Prior identifies the direction of the traversal for the edge alright and start with specify first vertex for path computation right. So, what is the start X is the initial vertex for the path computation semantics list all node reachable from first vertex using directed graph in specific table right.

So that all nodes list all nodes reachable from the first vertex using directed edge to the specific table what we require we require the closure here right. So, that I can from a given node I can reach to the all the nodes which are directed to the table assumption no cycle in the graph. So, we are thinking that there is no cycling in the graph if the graph is acyclic not suitable for train network road network where there is a inherent cycle.

In the river in a river network typically there is no cycle. So, in a river it is not that the river we will cycle and go to the top or at the something right there so, it is there. So, query list directed and in directed tributary of Mississippi this is a in that book of Prof. Shekhar that we have this example. Table is here the falls into source destination edge direction is from source to destination source falls into destination. And start node is Mississippi river id equal to 1 right, we will just see this example and may conclude today.



(Refer Slide Time: 30:09)

**SQL Connect Clause - Example**

- SQL expression
- Execution trace of paths
  - starts at vertex 1 (Mississippi)
  - adds children of 1
  - adds children of Missouri
  - adds children of Platte
  - adds children of Yellowstone
- Result has edges
  - from descendants
  - to Mississippi

```
SELECT source
FROM FallsInto
CONNECT BY PRIOR source = dest
START WITH dest = 1
```

(a) Mississippi network (Y1 = Big Horn river, Y2 = Powder river, P1 = Sweet water river, P2 = Big Thompson river)

(b) The sequence of the CONNECT clause

(1) First level

SOURCE
Ohio
Missouri
Red
Arkansas

(2) Second level

SOURCE
Ohio
Missouri
Platte
Yellowstone
Red
Arkansas

(3) Final result

SOURCE
Ohio
Missouri
Platte
Yellowstone
Red
Arkansas

[Ref: Spatial Databases: A Tour, by Shashi Shekhar, Sanjay Chawla; Internet resources]

Like, we have a SQL expression in the things select source from falls into that is my transitive closure thing and then connect by prior source equal to destination and start with destination equal to 1 right. So, I go on into the things where what was my query list direct and indirect tributaries of Mississippi, I am to go back and look into the things right.

So, if you see the tributaries of Mississippi if you want to list; so, it is all those Ohio, P 1, P 2, Platte, Yellowstone, Y 1, Y 2, Missouri and type of things all will come into play right. So, execution of the path start at vertex X Mississippi add children of 1, add children of Missouri right. So, this is Missouri add children of Platte Yellowstone right and results has edges from descendants to Mississippi right. So, we have the result.

So, initial stage it will be Ohio, Mississippi, Red and Arkansas. Next stage will be it will be expanding Ohio not to explain Missouri will be Platte and Yellowstone and this then Platte will be P 1, P 2, Yellowstone Y 1 2 this then the final results. So, the sequence of the connect clause is shown here and the it the example you can get in the book also and this way it moves right.

So, what we have seen today is a little deeper into the spatial network and try to see that how what is the structure at; what is the data model passboard at the conceptual, logical and physical level and then when you end of the day means end of the day or we have to have the spatial queries executed for the spatial network that we try to see.



We will be again discussing or continuing our discussion on this spatial network specially on the data specially on queries how to handle different type of queries and also how to handle large graphs etcetera that thing and the storage mechanisms how we look at. So, with this let us conclude our discussion today; we will continue in our subsequent lecture series.

Thank you.