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

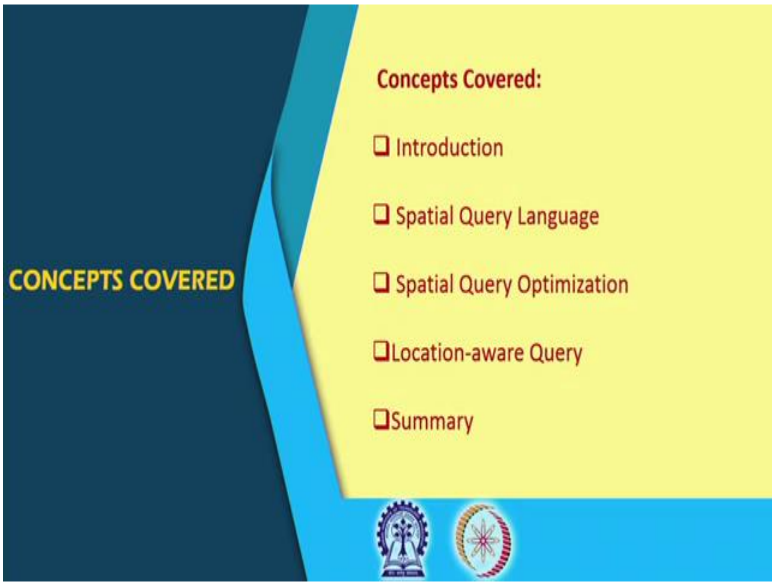
Lecture - 12
Spatial Query Processing / SQL (1)

Hello. So, we will continue our discussion on Spatial Informatics. So, we are at if you recollect the last lecture we discussed on generally on spatial database looking into that some portion of our normal DBMS and then SDBMS. So, that was that why the spatial database is required. Today or rather coming couple of lectures we will be looking at Spatial Query Processing or spatial SQL so, to say right.

So, as we understand and as most of you have already done somewhere form of all the databases courses. So, this SQL or this query language is a Structured Query Language is a de facto language for any database right for several little things. So, we like to see that how this SQL can be extended to cater or to take care of this spatial context or spatial database right.

As we have seen that the spatial database have base some as all those features of this traditional or non spatial database, spatial data comes with non spatial data attached with that data along with the spatial context or the spatial extension or geometry properties etcetera. So, SQL will be definitely whatever is SQL we know its there along with that what we require more to look at the things right. So, in this couple of discussion, we will be looking at those things.

(Refer Slide Time: 02:05)



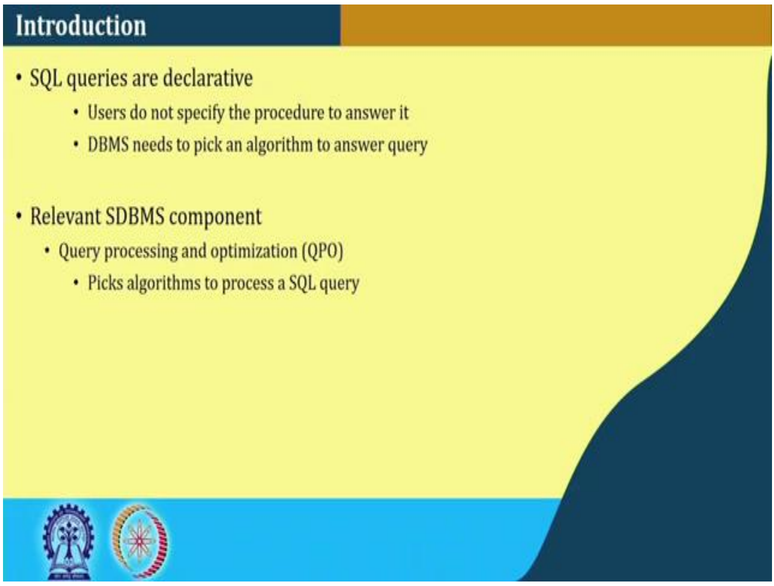
CONCEPTS COVERED

Concepts Covered:

- ❑ Introduction
- ❑ Spatial Query Language
- ❑ Spatial Query Optimization
- ❑Location-aware Query
- ❑Summary

And also we will be looking on some of the things of spatial optimizers and query optimization location as is a location aware queries and type of things right. So, this is our objective we will be little bit falling back to our query.

(Refer Slide Time: 02:19).



Introduction

- SQL queries are declarative
  - Users do not specify the procedure to answer it
  - DBMS needs to pick an algorithm to answer query
- Relevant SDBMS component
  - Query processing and optimization (QPO)
    - Picks algorithms to process a SQL query

.

Standard query language for keeping this linkages and for a few of you who are not very much accustomed with spatial data or with the database management systems. So, SQL so, the so, to say.

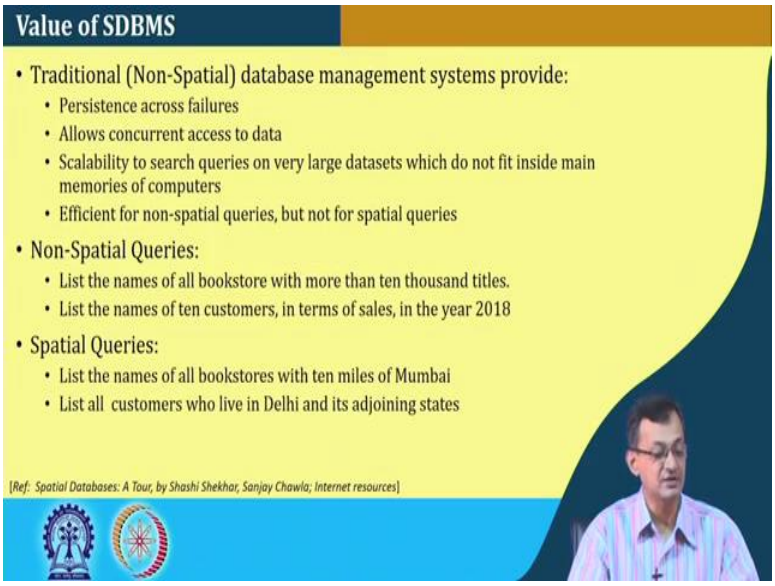
So, query as if you remember we discussed little bit about a SQL queries are declarative right users do not specify the procedure to answer it right. DBMS need to pick an algorithm to answer the queries. Like if I say no whether spatial non spatial or any SQL when I say that select so, and so, from database or from say relation r 1 r 2 r 3 where this type of things.

Now, this select is a process right or is a process this from multiple database joining is a process, but its as a user I do not know that what sort of process is or rather I do not care that how this database is faring these things, then what procedure it is running to answer it and type of things for me it is more of a higher level representation of the things.

So, if we look at the relevant DBS DBMS component query processing and optimization plays a big role because here the data load will may be high and then same query processing in a appropriate way may give me a much faster result right. The result will be the result will be same output will be same, but much faster result we will see that how this optimization come into play and how it happens.

So, pick algorithm to process SQL queries is also like I say that is a overlap a touching or a intersection. So, which algorithm I need to pick up for a particular queries say it query things when line by line versus polygon is there or polygon interacts intersection or line polygon intersection were on the same algorithm or different things will may affect my overall performance of the query processing right. So, that is important. So, we look at that subsequently not may not be today’s discussing subsequently that query processing and optimization issues.

(Refer Slide Time: 04:43)



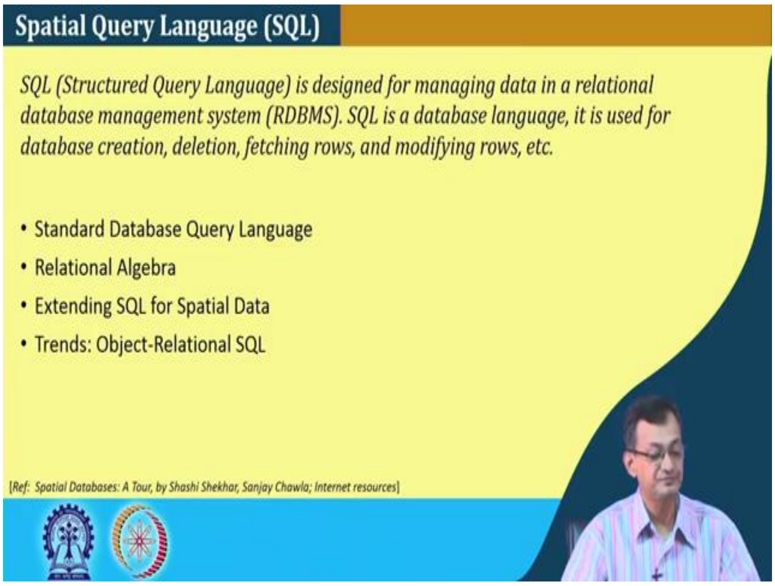
Value of SDBMS

- Traditional (Non-Spatial) database management systems provide:
  - Persistence across failures
  - Allows concurrent access to data
  - Scalability to search queries on very large datasets which do not fit inside main memories of computers
  - Efficient for non-spatial queries, but not for spatial queries
- Non-Spatial Queries:
  - List the names of all bookstore with more than ten thousand titles.
  - List the names of ten customers, in terms of sales, in the year 2018
- Spatial Queries:
  - List the names of all bookstores with ten miles of Mumbai
  - List all customers who live in Delhi and its adjoining states

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

So, again just to recap of our the slides we have already seen, the traditional non spatial database management it is standard right persistence across failure, concurrent access, scalability efficient spatial queries right. Non spatial queries we have seen some of the queries and these are the which have a there is a context of non spatial queries and some of the context in the spatial queries.

(Refer Slide Time: 05:09)



Spatial Query Language (SQL)

SQL (Structured Query Language) is designed for managing data in a relational database management system (RDBMS). SQL is a database language, it is used for database creation, deletion, fetching rows, and modifying rows, etc.

- Standard Database Query Language
- Relational Algebra
- Extending SQL for Spatial Data
- Trends: Object-Relational SQL

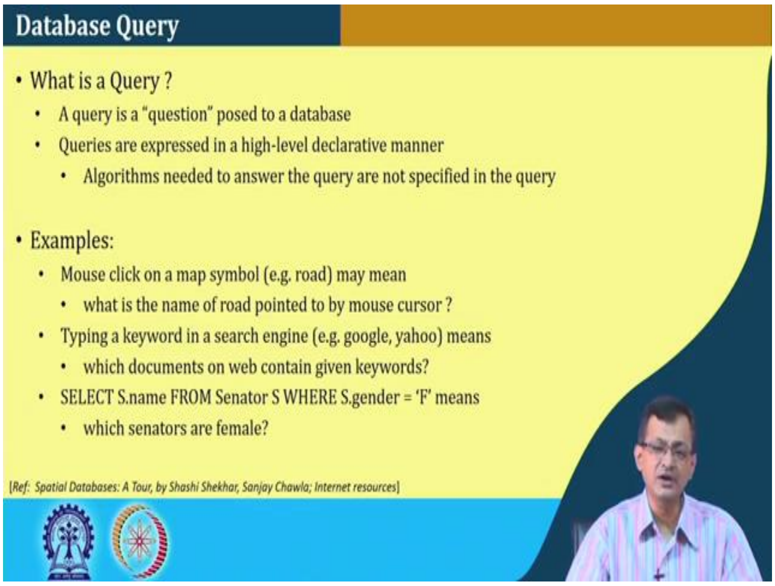
[Ref: Spatial Databases: A Tour, by Shashi Shekhar, Sanjay Chawla; Internet resources]

So, when we talk about structured query language. So, there are you will see in different books in internet, WIKI etcetera. So, that a thing is that there is a structured query language it is SQL is designed for managing data in a typically Relational Database Management System or RDBMS. SQL is a database language it is used for database creation, deletion, fetching rows etcetera querying is definitely there, but it is something more than the queries you I can input data I can create tables and type of things all I have a SQL or rather for that with the databases or the standard database we interact with these SQL things only right.

And there are several things and standard database query language, it have a one to one or close correspondence with these relational algebra those who have gone to gone through relational algebra we will understand it is we will not be discussing relational algebra, but now it does not matter.

Extending SQL for states spatial data is a challenge we will see that not only challenge that is needed for these things and some of the trend is now we are looking for object relational SQL again extension to spatial data and so, and so forth right. So, this basic SQL.

(Refer Slide Time: 06:37)



Database Query

- What is a Query ?
  - A query is a "question" posed to a database
  - Queries are expressed in a high-level declarative manner
    - Algorithms needed to answer the query are not specified in the query
- Examples:
  - Mouse click on a map symbol (e.g. road) may mean
    - what is the name of road pointed to by mouse cursor ?
  - Typing a keyword in a search engine (e.g. google, yahoo) means
    - which documents on web contain given keywords?
  - SELECT S.name FROM Senator S WHERE S.gender = 'F' means
    - which senators are female?

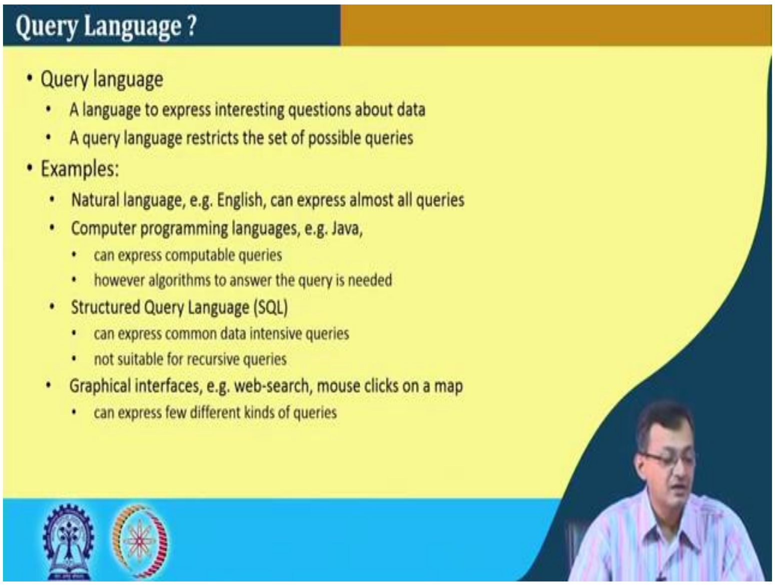
[Ref: Spatial Databases: A Tour, by Shashi Shekhar, Sanjay Chawla; Internet resources]

So, when you talk about database query. So, query is quote on quote a question posed to the database like find all students with a particular for a particular department or taking particular course and type of things, and staying in something some hall and etcetera something right. So, this is I am posing a database a query a question and I am getting.

So, queries are expressed in high level declarative manner as we have seen, algorithms needed to answer queries are not specified in the query right. How this select will work, how this where clause will work, how a how a join will work or they are having or different type of clause will work that is not specifically are declared in the SQL right.

SQL is a structured things and we have a grammar for that or a syntax for that and we pose the query on the things right. So, like if we look at the spatial I click on a mouse and I expect that it will give the road name on the cursor or typing key of a search engine, in some search engine will the content I can have some of the other queries which will rip represent or reply a particular things in for that queries.

(Refer Slide Time: 08:01)



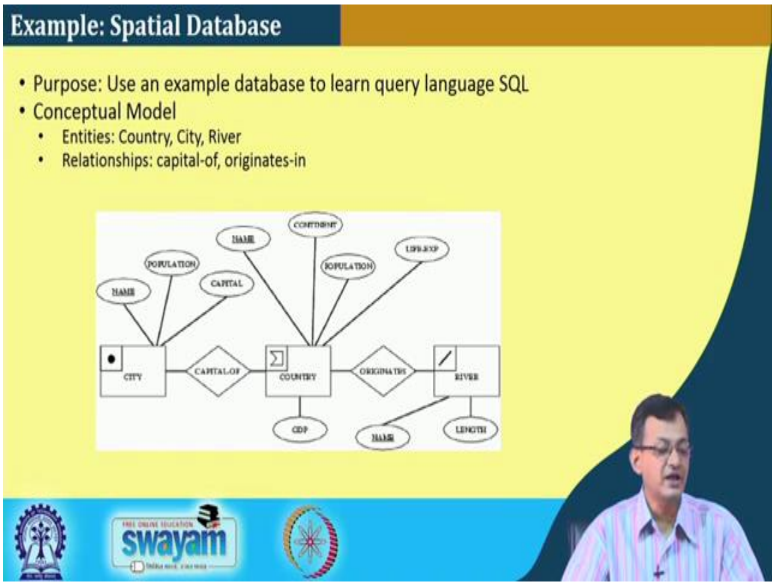
Query Language ?

- Query language
  - A language to express interesting questions about data
  - A query language restricts the set of possible queries
- Examples:
  - Natural language, e.g. English, can express almost all queries
  - Computer programming languages, e.g. Java,
    - can express computable queries
    - however algorithms to answer the query is needed
  - Structured Query Language (SQL)
    - can express common data intensive queries
    - not suitable for recursive queries
  - Graphical interfaces, e.g. web-search, mouse clicks on a map
    - can express few different kinds of queries

So, query language; so by definition a language to express interesting questions about the data that is why we are querying otherwise why should I query. A query language restricts a set of possible queries etcetera so, that it has a dialect or it has a syntax by which we can do, like a there may be natural language can express on all queries computer programming java etcetera. So, for our structured query language, can express common data intensive queries not typically not suitable for recursive queries.

So, if the queries are there recursively and that is a graphical interface web search mouse click on the map can express few different kinds of queries which in turn fired SQL at the back end right. So, I do some GUI based things where the typically the queries are if I had in the back end with that SQL thing.

(Refer Slide Time: 09:03)



Example: Spatial Database

- Purpose: Use an example database to learn query language SQL
- Conceptual Model
  - Entities: Country, City, River
  - Relationships: capital-of, originates-in

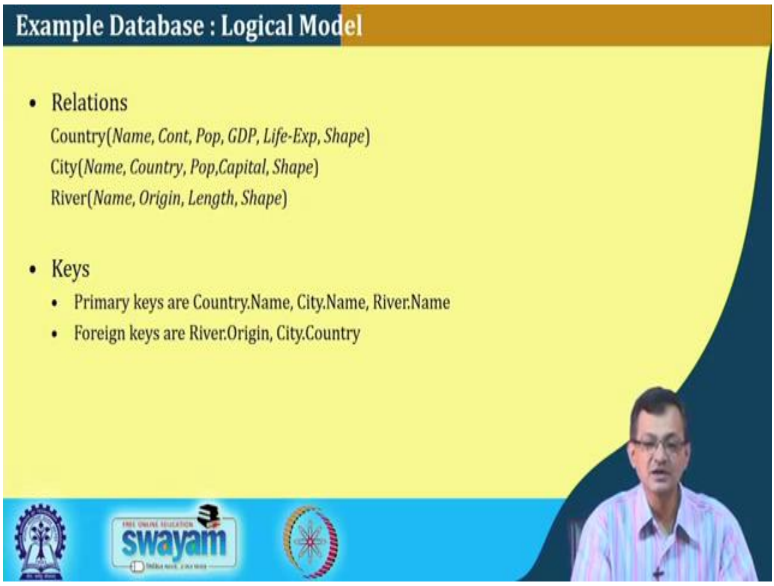
So, like if we look at a spatial database use a example database like thing conceptual model I have country, city, river as entities in this case three entities. There are relation capital of originates in type of things like if we have seen these type of things its a pictogram representation. So, city is a point feature, country is a polygon feature river is a line feature that is my way of things.

So, city has different attributes these are non spatial attributes right name, population, capital these are non spatial attributes. Population I can basically connect with the spatial, but as such population of the city is so, much so, many is a non spatial attribute. Country also has the name which continuities population, life expectancy, these are the country thing and there is a river which may have name length and there may be lot of other type of features for the for the your things like I can say different type of things perennial non perennial and type of things etcetera.

Now, these are the three things if I say that river particular river originates in a particular country may be one thing, city is a capital a particular city is a capital of a country right that can be another relationship, a country also can have a GDP and type of things. So, we have here country, city, river three entities which have some spatial representation and along with that there are in this case two relationship we can have multiple relationship there can be relationship in the city and river and so, and so forth.

So, relationship in this case is a capital of originates in right. So, river originates in a particular country flows to other countries and type of things and it can have a city is a capital of a country and so, and so forth right. So, this is this can be a my e r diagram or pictogram embedded e r diagram which I want to represents as thing.

(Refer Slide Time: 11:19)



Example Database : Logical Model

- Relations
  Country(*Name, Cont, Pop, GDP, Life-Exp, Shape*)
  City(*Name, Country, Pop,Capital, Shape*)
  River(*Name, Origin, Length, Shape*)
- Keys
  - Primary keys are Country.Name, City.Name, River.Name
  - Foreign keys are River.Origin, City.Country

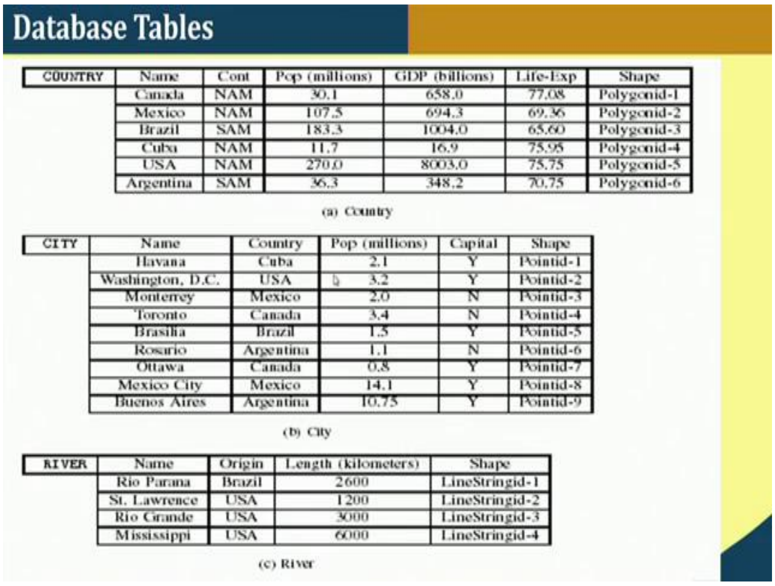
.

So, I have three relations like standard, name, country, population, GDP, life expectancy and the last one this is common this is standard we all know the other thing is the same right. So, country is a polygon, city shape is a point, river shape is a polyline.

So, there are three type of entities right and there can be there will be key primary key and foreign key like primary keys like country dot a name city dot river dot name we considered those are the primary keys. There can be some other type of things like IDs and type of things that is also possible there kind of foreign keys a river dot origin city dot origin that which refers to the other table right.

So, there are things like city dot origin is basically a country which is refer to city dot country is referred to other things right where is a capital of an other things. So, it means that it refers to some other table where by using these foreign keys.

(Refer Slide Time: 12:35)



Database Tables

COUNTRY	Name	Cont	Pop (millions)	GDP (billions)	Life-Exp	Shape
	Canada	NAM	30.1	658.0	77.08	Polygonid-1
	Mexico	NAM	107.5	694.3	69.36	Polygonid-2
	Brazil	SAM	183.3	1004.0	65.60	Polygonid-3
	Cuba	NAM	11.7	16.9	75.95	Polygonid-4
	USA	NAM	270.0	8003.0	75.75	Polygonid-5
	Argentina	SAM	36.3	348.2	70.75	Polygonid-6

(a) Country

CITY	Name	Country	Pop (millions)	Capital	Shape
	Havana	Cuba	2.1	Y	Pointid-1
	Washington, D.C.	USA	3.2	Y	Pointid-2
	Monterrey	Mexico	2.0	N	Pointid-3
	Toronto	Canada	3.4	N	Pointid-4
	Brasilia	Brazil	1.5	Y	Pointid-5
	Rosario	Argentina	1.1	N	Pointid-6
	Ottawa	Canada	0.8	Y	Pointid-7
	Mexico City	Mexico	14.1	Y	Pointid-8
	Buenos Aires	Argentina	10.75	Y	Pointid-9

(b) City

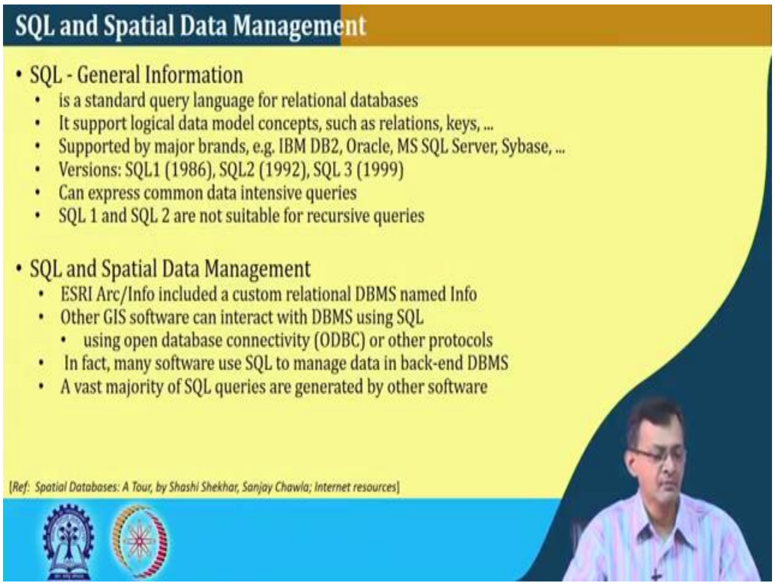
RIVER	Name	Origin	Length (kilometers)	Shape
	Rio Parana	Brazil	2600	LineStringid-1
	St. Lawrence	USA	1200	LineStringid-2
	Rio Grande	USA	3000	LineStringid-3
	Mississippi	USA	6000	LineStringid-4

(c) River

Now, we can have this type of a database schema there are several name, country, population etcetera and there are set this is a polygon, cities shape is point and river is a line segment and there are other parameters are non spatial parameters or whatever the traditional databases. So, as such I can store up to this is a very traditional way.

If I do not have a support for this spatial database or spatial data, then for this polygon I have to find out what are the edges and from the edges I have to find out what are the starting point ending point and the find the starting every edge start vertex and in vertex I have to find out what is the coordinate systems. Now, for that I require a different structure to handle that right similarly for point it may be the only coordinate for a line it again goes to that series of because series of line segments and they start end and that sequence also matters right. So, it should be in a sequence in that spatial referencing or what we say x y coordinates are important.

(Refer Slide Time: 13:51)



SQL and Spatial Data Management

- SQL - General Information
  - is a standard query language for relational databases
  - It support logical data model concepts, such as relations, keys, ...
  - Supported by major brands, e.g. IBM DB2, Oracle, MS SQL Server, Sybase, ...
  - Versions: SQL1 (1986), SQL2 (1992), SQL 3 (1999)
  - Can express common data intensive queries
  - SQL 1 and SQL 2 are not suitable for recursive queries
- SQL and Spatial Data Management
  - ESRI Arc/Info included a custom relational DBMS named Info
  - Other GIS software can interact with DBMS using SQL
    - using open database connectivity (ODBC) or other protocols
  - In fact, many software use SQL to manage data in back-end DBMS
  - A vast majority of SQL queries are generated by other software

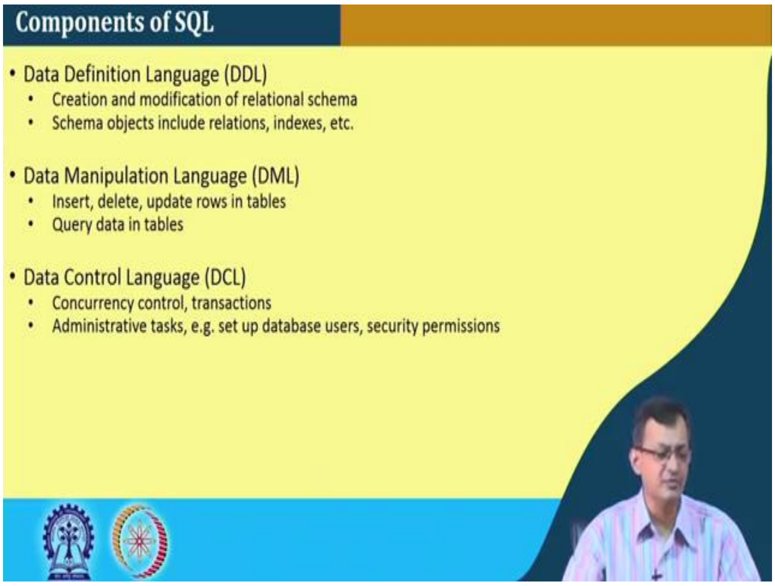
[Ref: Spatial Databases: A Tour, by Shashi Shekhar, Sanjay Chawla; Internet resources]

Now, if we look at the spatial SQL and spatial data database data management right. So, SQL again going to the standard things is a standard query language for relational database we already know, it supports logical data model concepts like relation keys etcetera support a by major brands make several major players or the database giants supports this, there are different versions SQL 1 SQL 2 SQL 3 there are different version can express common data intensive queries SQL, can handle SQL 1 and SQL 2 are not suitable for recursive queries right.

So, SQL 1 and SQL 2 are not suitable, but SQL 3 do support that. Now, if you look at the data spatial data in the context, there are several companies we have customized relational database right like history as a info. So, other GIS software can interact with database using SQL right I can have ODBC called type of things, I can have a something my SQL with spatial extension or oracle spatial then from my database I can connect to these queries right this is possible.

So, many software use SQL to manage data at the backend DBMS right. So, I have the backend database management system that my front end is something different tools like it can be GIS tools or anything and that the back way and it interacts with is in the SQL. So, its a vast majority of SQL queries are generated by other software. So, that it is software generated though we do write SQL manually, do will be writing or seeing SQL manually.

(Refer Slide Time: 15:43)



Components of SQL

- Data Definition Language (DDL)
  - Creation and modification of relational schema
  - Schema objects include relations, indexes, etc.
- Data Manipulation Language (DML)
  - Insert, delete, update rows in tables
  - Query data in tables
- Data Control Language (DCL)
  - Concurrency control, transactions
  - Administrative tasks, e.g. set up database users, security permissions

So, these are also standard I am not going much into the thing components of SQL, data definition language. So, creation and modification of relational schema, schema objects include relation indexes etcetera data manipulation language, insert delete update rows in a table query data on the on the tables right. I query and the output of the query is another relations.

So, I create another relations I can be one query or multiple queries and data control language concurrency control, transactional control, administrative likes setup database user security permission etcetera are data control languages. So, these are this is common for our any standard database it is also used in spatial database context.

(Refer Slide Time: 16:25)

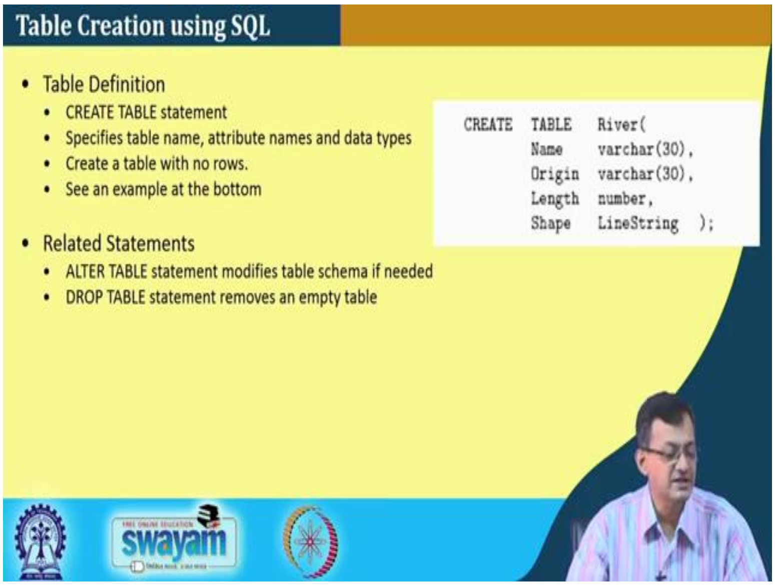


Table Creation using SQL

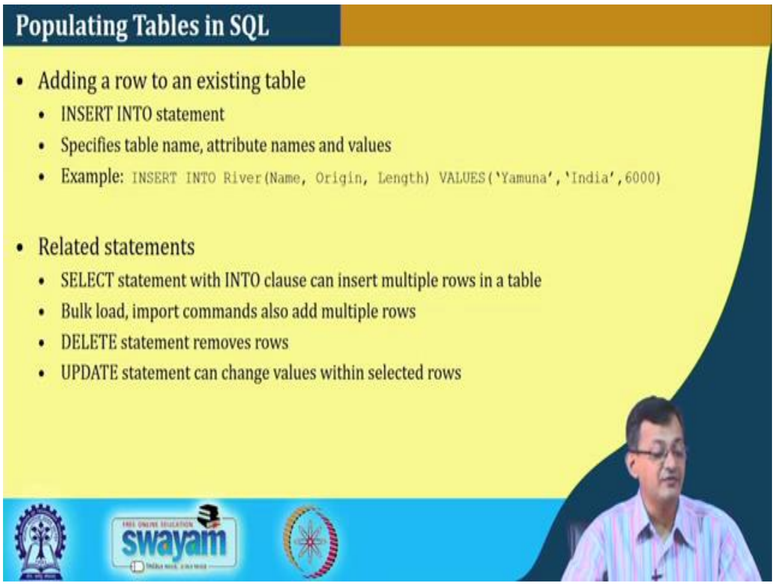
- Table Definition
  - CREATE TABLE statement
  - Specifies table name, attribute names and data types
  - Create a table with no rows.
  - See an example at the bottom
- Related Statements
  - ALTER TABLE statement modifies table schema if needed
  - DROP TABLE statement removes an empty table

```
CREATE  TABLE   River(
        Name    varchar(30),
        Origin  varchar(30),
        Length  number,
        Shape   LineString   );
```

So, there is a some of some very quickly well known SQLs or very vanilla type SQLs, I believe most of you are used to it, but just to sake of others and things we just having some few very small example. But there are some of the interesting features we will just see like definition, data table, creation table, statement specifies table name attribute name and data types create table with no rows.

So, we can this not on a bottom on the side actually, this is a example of creating a table. Related statement alter table statement modifies or drop table and etcetera right the interesting feature here is a shape is a line string. So, river we are creating a table or a schema, where for river where the name origin line these are they are along with that the shape of the river or shape of that object in this case line string is also specified right. So, that is that is the extra things what we see.

(Refer Slide Time: 17:41)

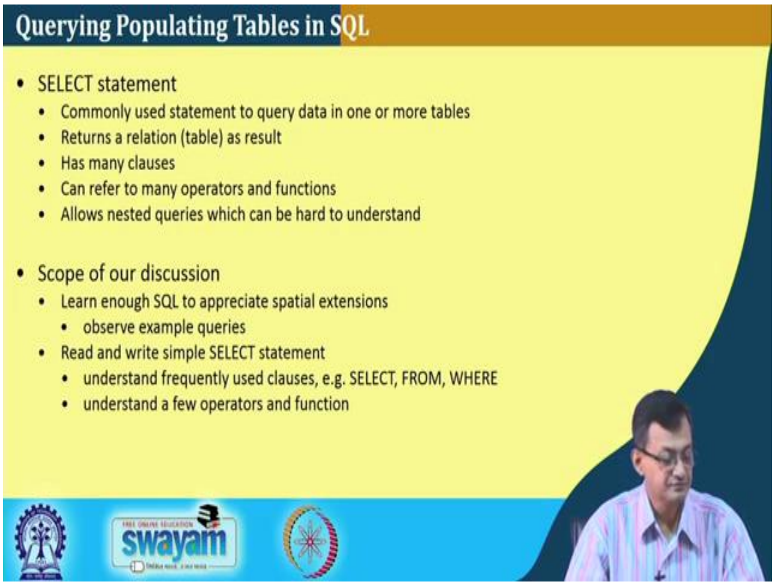


Populating Tables in SQL

- Adding a row to an existing table
  - INSERT INTO statement
  - Specifies table name, attribute names and values
  - Example: `INSERT INTO River(Name, Origin, Length) VALUES('Yamuna','India',6000)`
- Related statements
  - SELECT statement with INTO clause can insert multiple rows in a table
  - Bulk load, import commands also add multiple rows
  - DELETE statement removes rows
  - UPDATE statement can change values within selected rows

So, here I can have defined like populating the tables like insert into specify table name, attribute names and values like insert into river name origin length values something right. So, this is a standard way here also we do, and we have related statements select statement into clause can insert its insert the rows in the table, bulk load import commands are also for multiple rows, delete statement, update statement for change values etcetera. So, these are the things which are there.

(Refer Slide Time: 18:17)



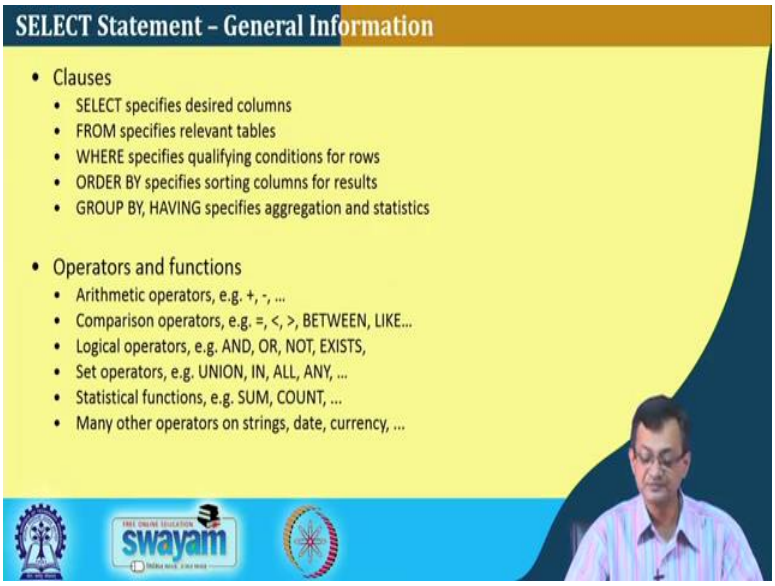
Querying Populating Tables in SQL

- SELECT statement
  - Commonly used statement to query data in one or more tables
  - Returns a relation (table) as result
  - Has many clauses
  - Can refer to many operators and functions
  - Allows nested queries which can be hard to understand
- Scope of our discussion
  - Learn enough SQL to appreciate spatial extensions
    - observe example queries
  - Read and write simple SELECT statement
    - understand frequently used clauses, e.g. SELECT, FROM, WHERE
    - understand a few operators and function

So, again these are common things, but let us just have a quick look in the things like select statement commonly used a statement to query data in one or more tables right I can do select so, and so, and not only that I can have multiple I can query for not only one table multiple tables, return a relation or table. So, any that is we know that any SQL once executed will return another table right.

So, this select will have a will create another relation what we say or table. So, it can have many clauses can refer to many operators and function allows nested query which can be hard to understand as said by the SQL itself we will see. So, we will concentrate on SQL and write select queries etcetera in the spatial context.

(Refer Slide Time: 19:17)



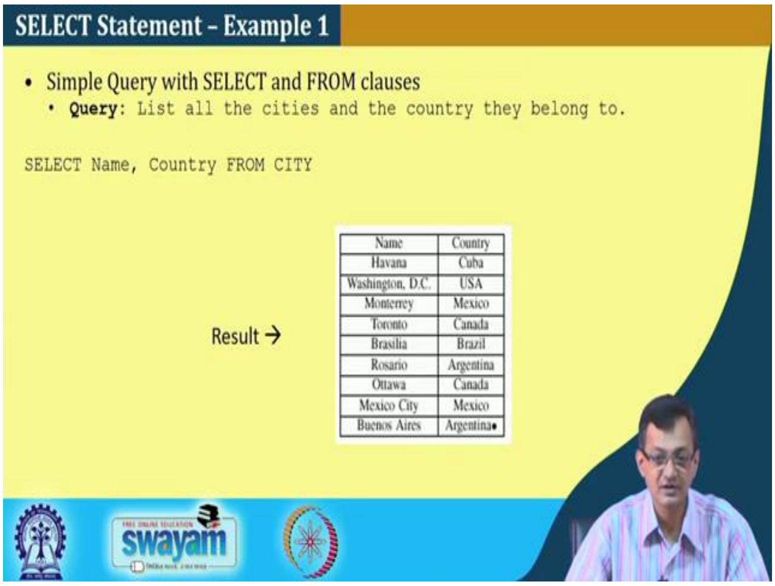
SELECT Statement - General Information

- Clauses
  - SELECT specifies desired columns
  - FROM specifies relevant tables
  - WHERE specifies qualifying conditions for rows
  - ORDER BY specifies sorting columns for results
  - GROUP BY, HAVING specifies aggregation and statistics
- Operators and functions
  - Arithmetic operators, e.g. +, -, ...
  - Comparison operators, e.g. =, <, >, BETWEEN, LIKE...
  - Logical operators, e.g. AND, OR, NOT, EXISTS,
  - Set operators, e.g. UNION, IN, ALL, ANY, ...
  - Statistical functions, e.g. SUM, COUNT, ...
  - Many other operators on strings, date, currency, ...

So, SELECT specified specify desired columns like SELECT so, and so, right common FROM specifies the relevant tables or the relations, WHERE specifies the qualify conditions or the rows, ORDER BY specifies the sorting columns for results, GROUP BY HAVING specify the aggregation and statistics right. So, these are the things which are specified.

Operators and functions arithmetic operation plus minus comparisons operations right greater than is lesser than between etcetera. Logical operations are supported, set theoretic operation or set operators like UNION, IN, ALL, not ANY and type of things statistical functions like aggregate functions like SUM and COUNT many other operators like string date currency etcetera supported. So, these are the set of SQL things which are being supported right. So, this is there.

(Refer Slide Time: 20:29)



SELECT Statement – Example 1

- Simple Query with SELECT and FROM clauses
  - **Query**: List all the cities and the country they belong to.

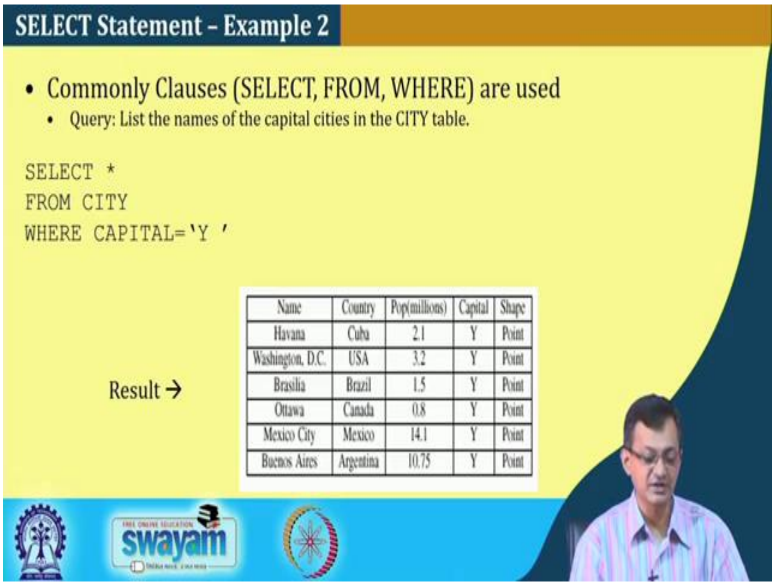
```
SELECT Name, Country FROM CITY
```

Result →

Name	Country
Havana	Cuba
Washington, D.C.	USA
Monterrey	Mexico
Toronto	Canada
Brasilia	Brazil
Rosario	Argentina
Ottawa	Canada
Mexico City	Mexico
Buenos Aires	Argentina

So, this is again a common SQL, simple query is select from select all the cities and country they belong to like name, country from cities and this is the only one city table right only one relation we are extracting two column.

(Refer Slide Time: 20:51)



SELECT Statement – Example 2

- Commonly Clauses (SELECT, FROM, WHERE) are used
  - Query: List the names of the capital cities in the CITY table.

```
SELECT *
FROM CITY
WHERE CAPITAL='Y '
```

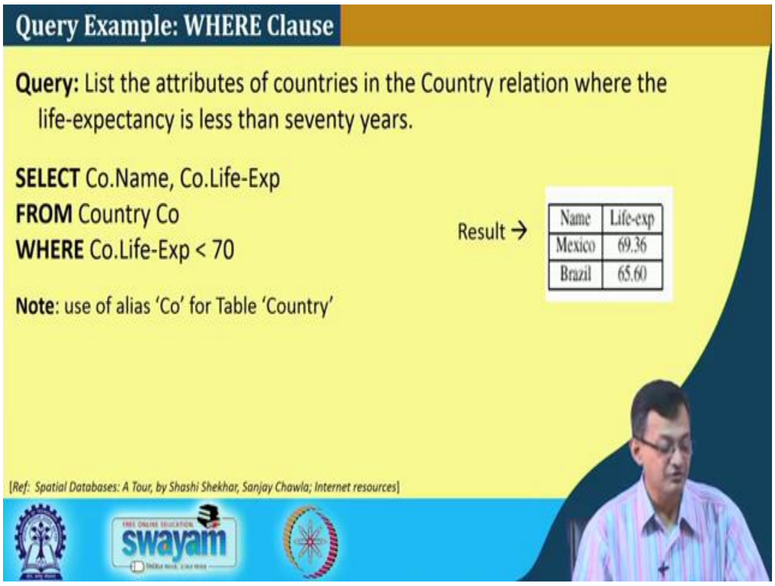
Result →

Name	Country	Pop(millions)	Capital	Shape
Havana	Cuba	2.1	Y	Point
Washington, D.C.	USA	3.2	Y	Point
Brasilia	Brazil	1.5	Y	Point
Ottawa	Canada	0.8	Y	Point
Mexico City	Mexico	14.1	Y	Point
Buenos Aires	Argentina	10.75	Y	Point

.

Similarly, another very straightforward is list the name of the capital city capital cities in the city table. So, SLECT star from city table if capital is y then display right. So, there may be lot of other cities etcetera, but I want to find out only the capital cities and from the city table right. So, for the other things our capital is no its not a capital and so, and so, forth.

(Refer Slide Time: 21:21)



Query Example: WHERE Clause

Query: List the attributes of countries in the Country relation where the life-expectancy is less than seventy years.

SELECT Co.Name, Co.Life-Exp
FROM Country Co
WHERE Co.Life-Exp < 70

Result →

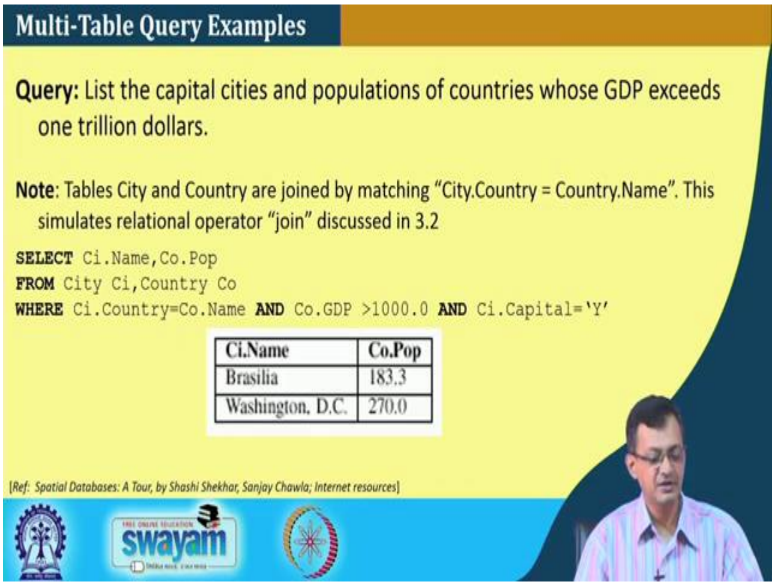
Name	Life-exp
Mexico	69.36
Brazil	65.60

Note: use of alias 'Co' for Table 'Country'

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

So, this is another that lists the attribute of the country relation where the life expectancy is less than 70 years and so, where clause is there. So, I defined country Co and then use that where clause to say.

(Refer Slide Time: 21:47)



Multi-Table Query Examples

Query: List the capital cities and populations of countries whose GDP exceeds one trillion dollars.

Note: Tables City and Country are joined by matching "City.Country = Country.Name". This simulates relational operator "join" discussed in 3.2

```
SELECT Ci.Name,Co.Pop
FROM City Ci,Country Co
WHERE Ci.Country=Co.Name AND Co.GDP >1000.0 AND Ci.Capital='Y'
```

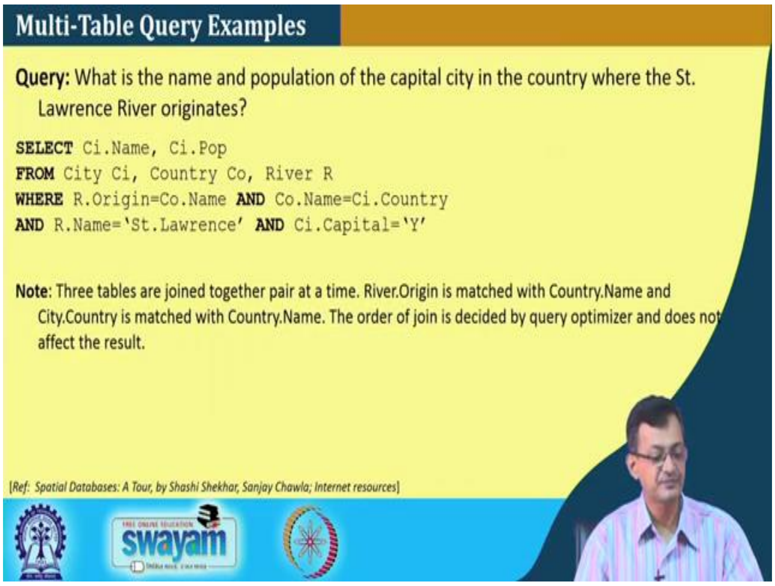
Ci.Name	Co.Pop
Brasilia	183.3
Washington, D.C.	270.0

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

So, list the capital cities and population countries whose GDP exceeds 1 trillion dollar right. So, again I have SELECT Ci Name Co Pop while City Ci represent Country. So, two tables are joining right we in this case we are joining two tables and if the select clause to distinguish between this which name you are taking and which pop you are taking which table I have a qualify that Ci dot this Co dot pop and then do that Ci pop Ci country should be Co Name and Co GDP should be so, and Ci table will be capital will be yes.

So, these so, finding the capital cities and population of the countries which whose GDP exceeds the 1 trillion dollar right so, that may be some any such queries we are able to. So, these are standard query sets we are able to handled them by SQL can handle them right. So, this is nothing.

(Refer Slide Time: 22:50).



Multi-Table Query Examples

Query: What is the name and population of the capital city in the country where the St. Lawrence River originates?

```
SELECT Ci.Name, Ci.Pop
FROM City Ci, Country Co, River R
WHERE R.Origin=Co.Name AND Co.Name=Ci.Country
AND R.Name='St.Lawrence' AND Ci.Capital='Y'
```

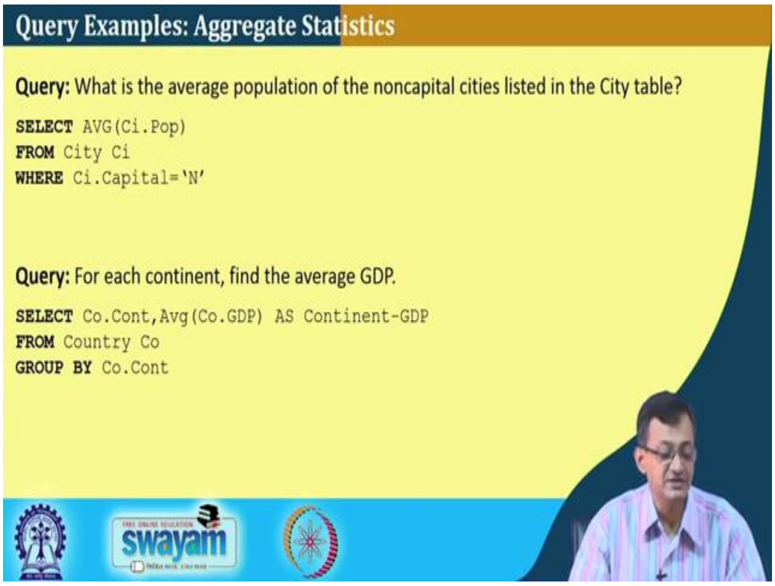
Note: Three tables are joined together pair at a time. River.Origin is matched with Country.Name and City.Country is matched with Country.Name. The order of join is decided by query optimizer and does not affect the result.

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

Similarly, what is the name of and population of the capital city in a country where something ST Lawrence River originates right. So, here we have three tables joining city, country, river and then R dot origin equal to company dot name, Co dot name equal to Ci dot Country and R dot name Lawrence and so, and so forth right.

So, three tables are joined together pair at a one pair at a time, river dot origin is matched with country dot name and city dot matched with the country dot name and the order of join is decided by the query optimizer and does not affect the results now whom you will join first 10 sets? So, I have three tables. So, whom should I join right? First the two largest one or one largest one with the one smaller one so, that my cardinality or the overall data load reduces and then do the things. So, that is decided by the query optimizer right which will give me more efficient, but results will be unaffected whether I use this sort of queries or other set of queries, the results will remain unaffected.

(Refer Slide Time: 24:05)



Query Examples: Aggregate Statistics

Query: What is the average population of the noncapital cities listed in the City table?

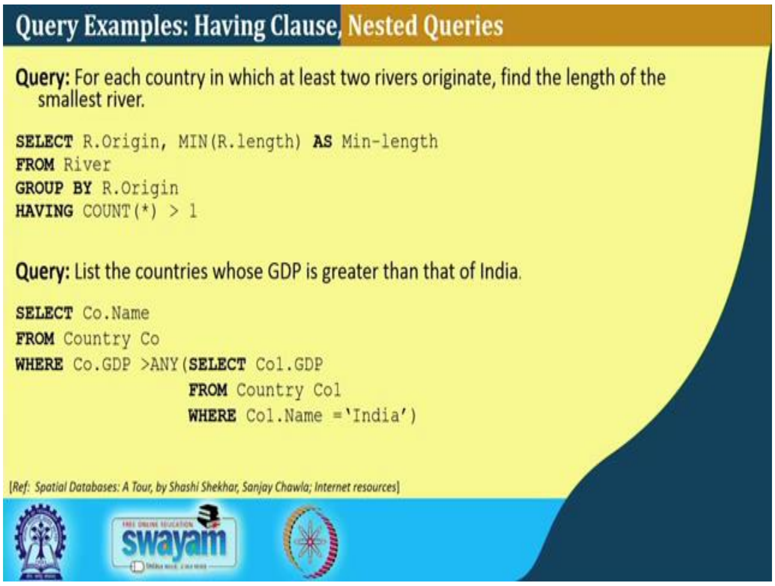
```
SELECT AVG(Ci.Pop)
FROM City Ci
WHERE Ci.Capital='N'
```

Query: For each continent, find the average GDP.

```
SELECT Co.Cont,Avg(Co.GDP) AS Continent-GDP
FROM Country Co
GROUP BY Co.Cont
```

So, what is the average population of non capital cities listed in the city capital right? What is the average population and we say that the select average etcetera. So, average is I have now a statistical or aggregate operation into the things. So, find each continent find the average GDP, then I have handling that average GDP and type of things. So; that means, I can work with standard; that means, retrieve the standard or I can do some this sort of statistical or aggregate operation to do that right this SQL supports.

(Refer Slide Time: 24:41)



Query Examples: Having Clause, Nested Queries

Query: For each country in which at least two rivers originate, find the length of the smallest river.

```
SELECT R.Origin, MIN(R.length) AS Min-length
FROM River
GROUP BY R.Origin
HAVING COUNT(*) > 1
```

Query: List the countries whose GDP is greater than that of India.

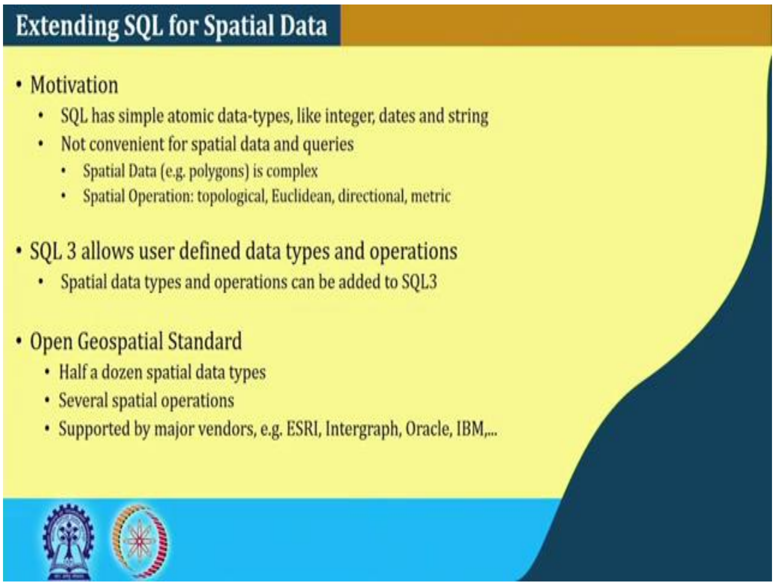
```
SELECT Co.Name
FROM Country Co
WHERE Co.GDP >ANY(SELECT Co1.GDP
                  FROM Country Co1
                  WHERE Co1.Name ='India')
```

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

So, now we for each country in which at least two river originate find the length of the smallest river. So, this is a more difficult query right. So, find out each country in which at least two rivers originate. So, if there is if in the countries there are any country there are more than one river, then I have to find the length of the smallest one right. So, that is my objective. So, a given a country, if I have a more than one river then things. So, that becomes a little tricky right like a R dot origin minimum R length as mean length or something river group by. So, I have to use the group by clause group by clause by R dot origin right. So, where the country originates and the count star should be greater than 1.

So, more than one river otherwise what is the fun of having small. So, what we see here that again that SQL do support. So, list the countries whose GDP is greater than that of India or something, then I have this find out those countries where GDP is gave have this any clause and select country 1 another instance of the country of GDP and where country 1 dot name is there. So, I find out that GDP of the India and Co 1 I do it.

(Refer Slide Time: 26:13)



Extending SQL for Spatial Data

- Motivation
  - SQL has simple atomic data-types, like integer, dates and string
  - Not convenient for spatial data and queries
    - Spatial Data (e.g. polygons) is complex
    - Spatial Operation: topological, Euclidean, directional, metric
- SQL 3 allows user defined data types and operations
  - Spatial data types and operations can be added to SQL3
- Open Geospatial Standard
  - Half a dozen spatial data types
  - Several spatial operations
  - Supported by major vendors, e.g. ESRI, Intergraph, Oracle, IBM,...

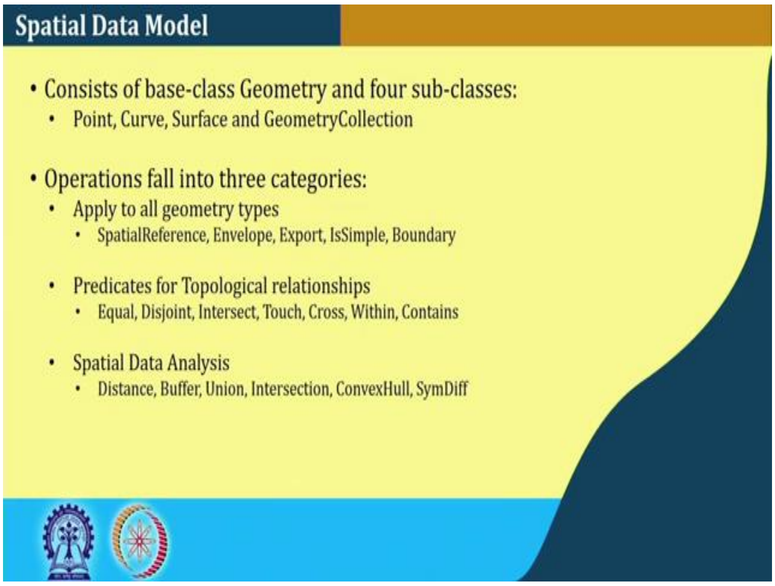
Now, so, these are very some of just to show that that what is SQL or just to have a quick recap or somebody who are not used to SQL to just to have a quick overview of it. If you are not accustomed with SQL I sincerely encourage you to read any state and state any standard database book right to look at that what is a SQL means.

Now, if I want to extend a extend this SQL for spatial data. So, on motivation the SQL has a simple atomic data types like integer, date string there is no nice polygons etcetera.

Non convenient for spatial data queries spatial data polygons is like data types like polygon etcetera are complex, spatial operation, topological Euclidean, directional metrics and other things.

SQL 3 allows user defined data types and operations right. So, spatial data types and operation can be added in the SQL 3 and we have this new open geospatial standard right. Because unless there is a standard everywhere we will use their own way of representing there is a very difficult scenario of interoperating between these databases right. So, there are different standards.

(Refer Slide Time: 27:35)



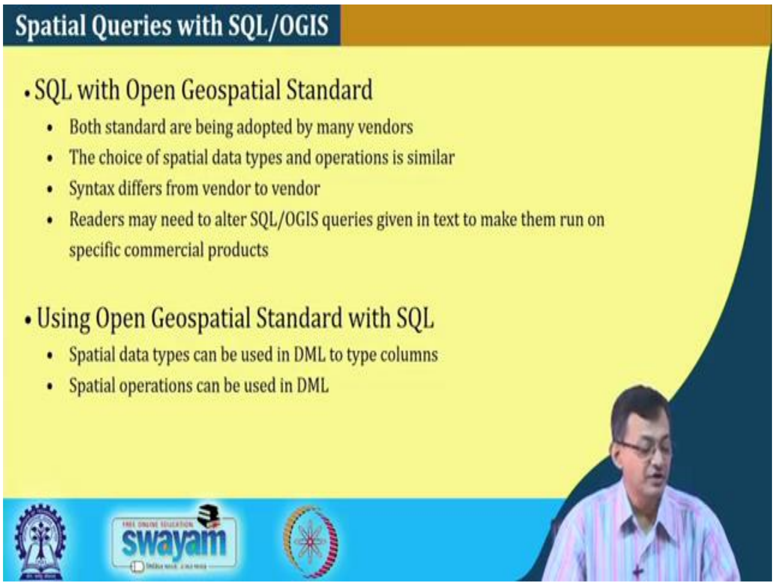
Spatial Data Model

- Consists of base-class Geometry and four sub-classes:
  - Point, Curve, Surface and GeometryCollection
- Operations fall into three categories:
  - Apply to all geometry types
    - SpatialReference, Envelope, Export, IsSimple, Boundary
  - Predicates for Topological relationships
    - Equal, Disjoint, Intersect, Touch, Cross, Within, Contains
  - Spatial Data Analysis
    - Distance, Buffer, Union, Intersection, ConvexHull, SymDiff

So, now the spatial data model if you recollect, consists of base class geometry and four subclasses like point, curve, surface, geometry collection and so, on and so, forth. Operations falls into three categories apply to all geometric classes spatial reference, envelop, export is simple boundary and different type of classes. So, operation falls into three major categories.

So, like one is apply on the geometry, predicates for topological relationships like touching, overlapping, within contains disjoint equivalent type of things or it can be a spatial data analysis right distance, buffer, union, intersection, finding the ConvexHull SymDiff and type of things right. So, one is that apply to all geometry types, other is the predicates or the of topological relations and spatial data analysis right. So, this is the looking at the spatial models right.

(Refer Slide Time: 28:41)

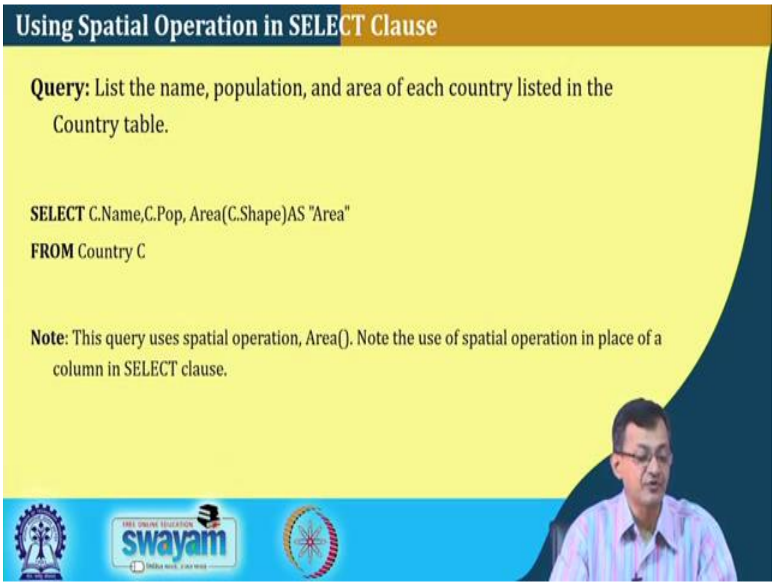


Spatial Queries with SQL/OGIS

- SQL with Open Geospatial Standard
  - Both standard are being adopted by many vendors
  - The choice of spatial data types and operations is similar
  - Syntax differs from vendor to vendor
  - Readers may need to alter SQL/OGIS queries given in text to make them run on specific commercial products
- Using Open Geospatial Standard with SQL
  - Spatial data types can be used in DML to type columns
  - Spatial operations can be used in DML

So, SQL with we have open geospatial standard, both standards we can obtained by many vendors like SQL is already there open geospatial standard. Choice of spatial data types and operations is similar, syntax differs may differ from vendor to vendor may alter OGIS queries given so, that to run the things. So, we have special data we can use DML to have type column spatial operations can be used for the data manipulation language and type of things like some queries like.

(Refer Slide Time: 29:13)



Using Spatial Operation in SELECT Clause

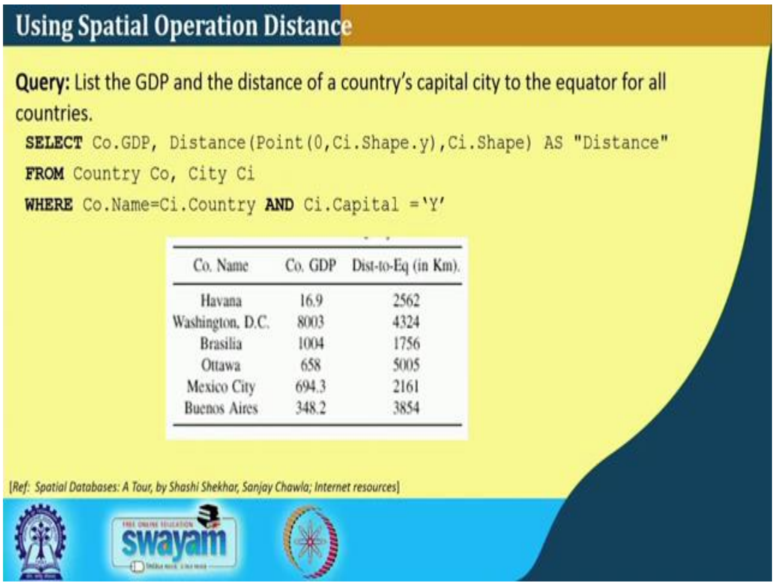
Query: List the name, population, and area of each country listed in the Country table.

```
SELECT C.Name,C.Pop, Area(C.Shape)AS "Area"
FROM Country C
```

Note: This query uses spatial operation, Area(). Note the use of spatial operation in place of a column in SELECT clause.

List the name population and area of each country listed in the country table right. So, this is their. So, area finding of a country table what I want to do? I need to collect the two way calculation on the shape of the things right. Country as we have seen is a polygon. So, I have to I can use a clause called area of C shape. So, this calculating a area of a polygon right. So, appropriate algorithms or appropriate procedure has to be launched at the back end right. So, this query uses spatial operations called area note that this spatial operation is a place in the select column clause we can place right it appropriately call that a particular procedure.

(Refer Slide Time: 29:59)



Using Spatial Operation Distance

Query: List the GDP and the distance of a country's capital city to the equator for all countries.

```
SELECT Co.GDP, Distance(Point(0,Ci.Shape.y),Ci.Shape) AS "Distance"
FROM Country Co, City Ci
WHERE Co.Name=Ci.Country AND Ci.Capital ='Y'
```

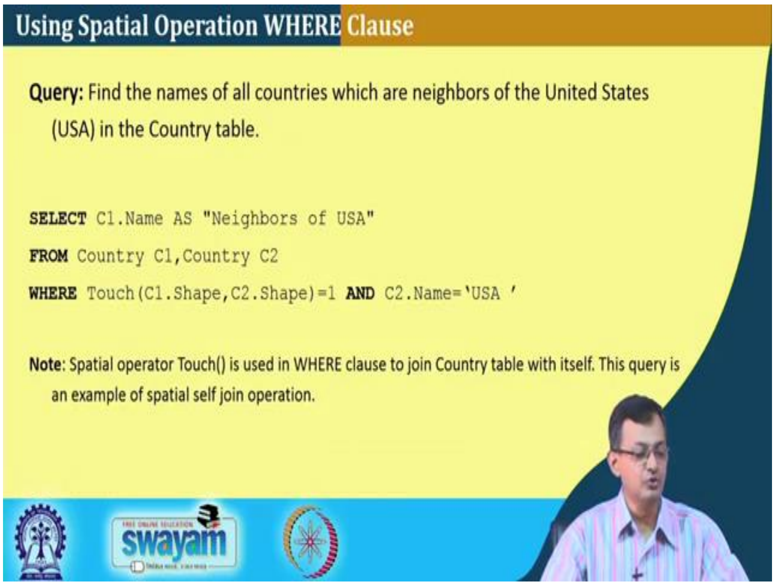
Co. Name	Co. GDP	Dist-to-Eq (in Km).
Havana	16.9	2562
Washington, D.C.	8003	4324
Brasilia	1004	1756
Ottawa	658	5005
Mexico City	694.3	2161
Buenos Aires	348.2	3854

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

Let us similarly find GDP and the distance of the country's capital city to the equator of all the countries right so; that means, now I have to find out that distance. So, how I find out the distance? Distance is between two points right and if you have to find the equator then either X 0 and Y type of things is a projection should be there and find out this particular city shapes.

If the city is a point then it is a that point and the pointer to the equator of all the countries right city capital to the equator right. So, I have to find out that where the equator is located and then find out the kilometer by or the distance between these. So, what we are doing? We are putting the spatial operation into play right.

(Refer Slide Time: 30:57)



Using Spatial Operation WHERE Clause

Query: Find the names of all countries which are neighbors of the United States (USA) in the Country table.

```
SELECT C1.Name AS "Neighbors of USA"
FROM Country C1,Country C2
WHERE Touch(C1.Shape,C2.Shape)=1 AND C2.Name='USA '
```

Note: Spatial operator Touch() is used in WHERE clause to join Country table with itself. This query is an example of spatial self join operation.

So, what we tried to see similarly this is another clause where we have a find all name of the countries neighborhood of United State or India in the country table. So, this touching this. My definition of neighborhood is if the country touching touch and one country then they are neighbors right. I can have different type of definition I can say if the country is within some kilometer of the things etcetera they are neighbor or etcetera, but in this case is touching. So, this touch operation is again a spatial operations right.

So, which I where we put as a WHERE clause and the queries a example of the spatial self join operations right. That the two country; country 1, country 2 a join and to find out that who touches one. So, there is a two tables such say join operations right. So, what we see that extending this SQL to the spatial operations has different complexities right. We will we will continue our discussion in the subsequent classes. So, what we try to look at is that, what is a basic SQL and how this SQL can be extended in k for handling spatial data sets its right. So, we will be continuing, we will let us conclude our discuss our today’s discussion here. So, we will continue in the next class.

Thank you.