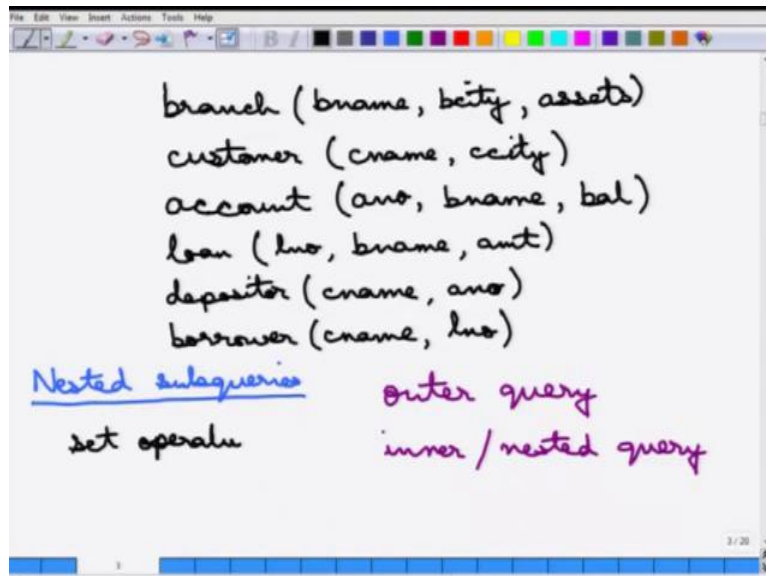


**Fundamentals of Database Systems**  
**Prof. Arnab Bhattacharya**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kanpur**

**Lecture - 10**  
**SQL: Advanced Queries**

(Refer Slide Time: 00:09)



This is the nested sub query, so essentially, what happens is that a query is used in the from and where clause of another query. The query on which the from and where is used is called the outer query and the query that is used inside is called the inner query, so inner query or sometimes it is also called the nested query. An example I will give immediately, but this is mostly useful for set operations.

So, if there are set operations, then these nested queries are very useful. So, for example, suppose, we want to find out names of all customers having a loan, but not an account, this can be solved in the following manner.

(Refer Slide Time: 00:58)

account (ano, bname, amt)  
loan (lno, bname, amt)  
depositor (cname, ano)  
borrower (cname, lno)

Nested subqueries

set operations  
find names of customers having loan but no account

outer query  
inner/nested query  
select cname  
from borrower  
where cname not in  
(select cname  
from depositor)

This is an inner query select cname from depositor, so let us try to understand, what is happening the how the query is being deciphered. So, first of all this inner query is run select cname from depositor, so it essentially gives the names of all the customers, who have a deposit account. And then, it tries to select those names from the borrower; that is it was a loan where cname is not in this, this is a set this is essentially equivalent to the naught of set membership and this is where nested sub query is very useful.

(Refer Slide Time: 01:46)

branch (bname, bcity, assets)  
customer (cname, ccity)  
account (ano, bname, bal)  
loan (lno, bname, amt)  
depositor (cname, ano)  
borrower (cname, lno)

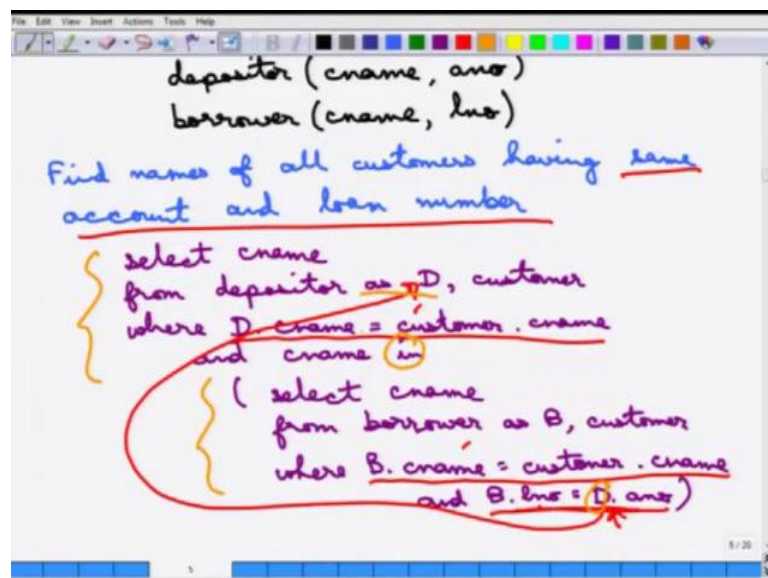
Scoping of attributes  
innermost

correlated queries

And related issue for this nested sub query is the scoping of attributes, so the scoping. So, this is the same issues in other programming languages such as C, when block of statements is used and inside another block of statements. And the same name is used, which scope it applies to and the same default rule applies, so if nothing is qualified it applies to the innermost query; otherwise it can be qualified.

And a nested query, so then there is some other terminology that we are going to use. When a nested query refers to something in the outer query; that is called a co related query, because essentially these two queries are not independent of each other, but they use the results of each other.

(Refer Slide Time: 02:33)



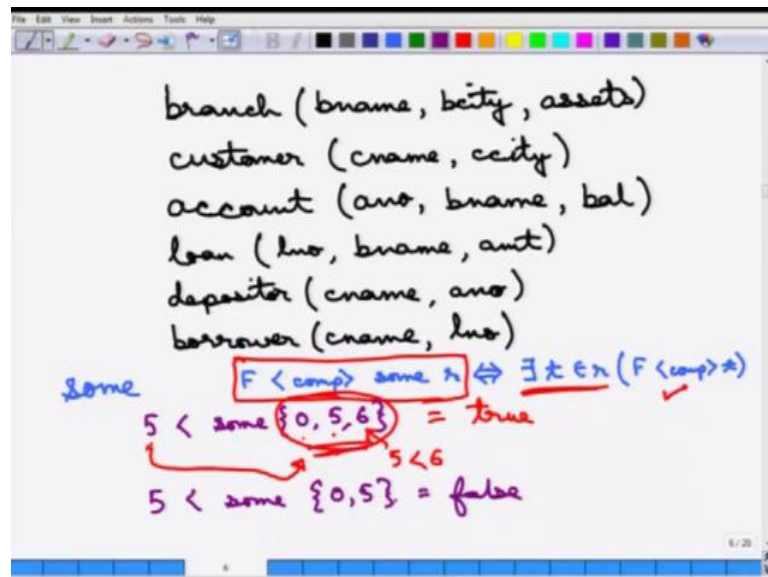
So, the query is, so this is solved using a correlated query and the answer to this solution can be written in the following manner, so this is the most important part of this. So, this B dot L number is equal to D dot A number, so this essentially this D refers to the one in the outside query, so this is why it is a correlated query. Let us also go over the query to see how this solves the particular, so how does it solve find names of all customers having same account and loan number.

So, let us first see, what is being done in an inner query. So, the inner query says it selects all customers, who has a loan such that of course, the name is the same, this is just to ensure that the join is meaningful and whose loan number is equal to some other account number D, where... What is D, where D is the depositor, so D is the depositor

which has the same customer name. So, essentially it tries to connect B dot L number with D dot A number.

And because of these two clauses D dot B dot cname is equal to customer dot cname and D dot cname is equal to customer dot cname, it connects them with the same customer. This is a little complicated query, but if this is being understood one at a time, first the inner query with this scoping to the outside and then, it is easier to understand. And it uses different concept, first is a correlated query and of course, which is a nested query then, this renaming and then also this set membership operation in.

(Refer Slide Time: 03:59)



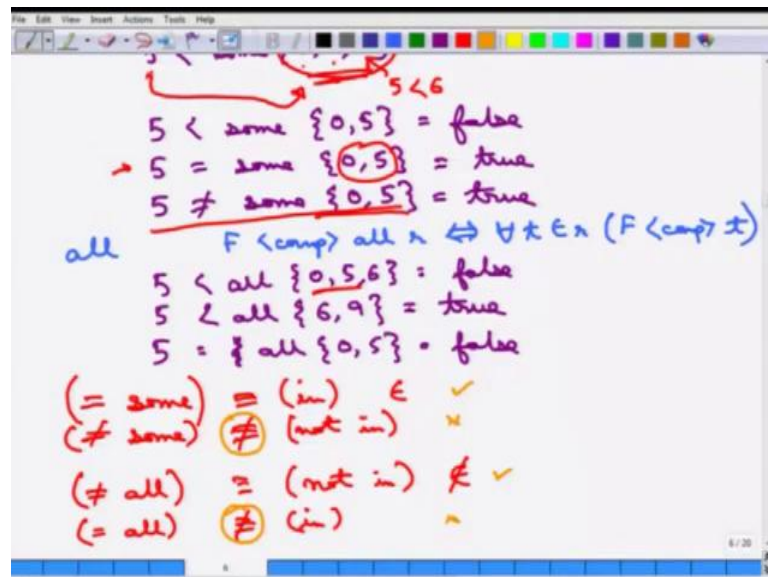
So, now, let us go over some of the set operators the set comparison operators the first set comparison operator that we will talk about is called some. So, some and here is the definition of some, so this is F compared to some comparative operator with some r is implies that this is equivalent to... There exists some tuple in the relation r; such that F compares to D. So, what does it mean? So, let us take an example, so if we take this following example 5 is less than some of, let us say the set is 0 5 6.

So, what does it mean? That we must find some D in this relation and if we can find some D in this relation such that 5 is less than that, then this is true. So, how which is being solved is that 5 is tested with 0 it is not true that does not matter 5 is tested with 5, which is not true. But, 5 is less than 6, so that comparison goes correct. So, this evaluates

to true, because there is some element in this such that 5 is less than this comparator is less than one of those elements.

So, there must be at least one element; such that this holds; then you can say that together if this holds, this comparator with some r holds, so that is the meaning of some operator. And we can see some examples for example, if you say 5 less than some 0 comma 5 this will; however, evaluated to false, because there is no element; such that 5 is less than 5. So, this goes through and then, it can be the equivalent of this can be defined.

(Refer Slide Time: 05:57)



So, let me just complete this, so 5 is equal to some 0 5 this is true, because there is one element, where 5 is equal to true. Very importantly and this is a little counter entity 5 not equal to some 0 5 is also true, because there is some element in the set that is not equal to 5, so this is a little counter entity, but this is how the thing works. So, let me go ahead with the next operator, which is similar operator, which is all, the all is the defined in the same manner.

So, F comparator with all r is equivalent to for all t belonging to r this must hold, so all the elements must follow this. So, then, if we say 5 less than all 5 6 0; that is false, because there is some element 6 1, where 5 there are some element 0 and 5, where this is not true, so this is false. And, similarly we can go ahead with the other examples, which

is with if this is 5 less than all of let us say 6 comma 9 this is true, then 5 is equal to all of 0 comma 5 this is false and so on, so forth.

So, essentially, what happens is, now we can decode some and all in terms of their set membership. So, some if you say equal to some; that is the same as saying, so this operator equal to some is the same as saying whether in. So, when you say for example, 5 equal to some; that means, is there any element in this set; such that this equality holds, so that is essentially the in operator this is the in operator.

However, not equal to some is not equivalent to not in this is something to be remembered in this is not equivalent, because again we see from this example this can be clear that it does not mean that it is not in. On the other hand if, we now look at all, so the all we can break down all in the following manner not equal to all if I write not equal to all; that is equivalent to not in, so this is essentially this. So, all of the set members must not be equal to this.

So, none of the set members can be equal to this; that means, this is not in that set. However, is in the same logic if you say equal to all; that is not equivalent to in, so these are just two of the two important things to remember this goes through, this goes through, but these two do not go through. So, this is why these two are equivalent, then there is a little bit issue with the empty set.

(Refer Slide Time: 09:09)

The image shows a whiteboard with handwritten notes. At the top, there are four database relations listed: 'account (accno, cname, amt)', 'loan (lno, bname, amt)', 'depositor (cname, accno)', and 'borrower (cname, lno)'. Below these, the notes discuss set properties. Under 'Empty set exists', it states '(exists x) ⇔ (x ≠ ∅)' and '(not exists x) ⇔ (x = ∅)'. Under 'Duplication in set unique', it states '(unique x) ⇔ ∀ x, y ∈ x (x ≠ y)' and '(not unique x) ⇔ ∃ x, y ∈ x (x = y)'. At the bottom left, it says 'in (11, 22, 33)'. The whiteboard has a toolbar at the top and a status bar at the bottom.

And, let me go to the next page, so there is an issue with the empty set. So, what is an empty, how a set can be empty of course, and there can be tested with the. So, there is an operator called exists, which tests whether the set whether there is an element in the set, which is essentially saying whether the set is empty or not.

So, exists r, which essentially is equivalent to saying r is not equal to the null set, which is this thing and where, is the complementary is not exists r, which is equivalent to saying r whether r is equivalent to 5 or not and again this can be used in different set operations etcetera. Similar to empty set there can be duplication in sets, because remember that SQL uses multi sets, so duplication in a set can be tested by this operator called unique.

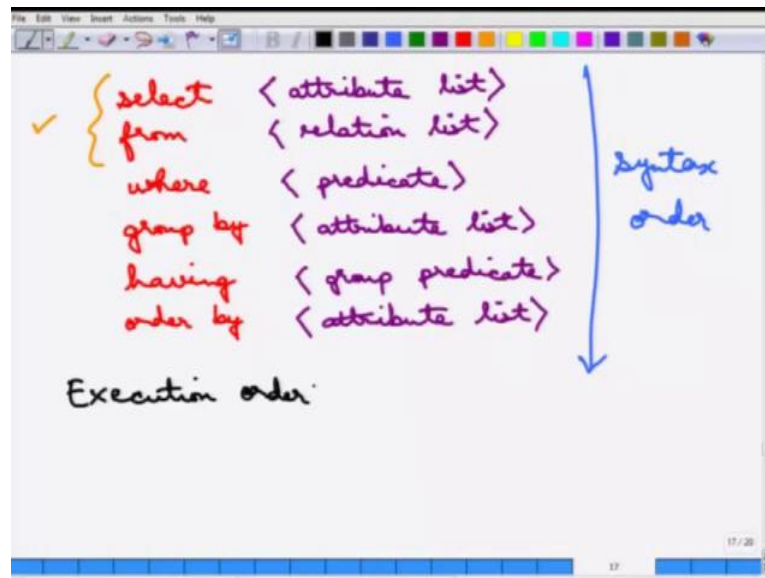
So, essentially testing whether there is something unique in that, so if you say unique r; that means, you are testing if all the tuples in the set r are unique, which is it can be written down in the following manner for all t and s that belongs to r t is not equal to s. And then, there is not unique meaning there is at least one tuple t, which is duplicated, so; that means, that there exists at t and an s in r; such that t is equal to s. So, that is how the duplication is done.

So, duplication is useful in the examples such as find names of customers, who have utmost one account at a particular branch. So, essentially you are testing whether for that customer you find out all the accounts in that particular branch and say if the customer name is duplicated or not if you otherwise you do not have to do that. So, this is the way to do it and in SQL just to complete the story sets can be explicitly denoted using this kind of operator, so this is a set with these three things 11, 22, 2.

So, something can be tested, so one can test whether a particular account number is within this or whatever. So, this is the basic idea of a SQL format, so let me just highlight the format of this SQL this is essentially consists of up to 6 clauses.



(Refer Slide Time: 11:51)



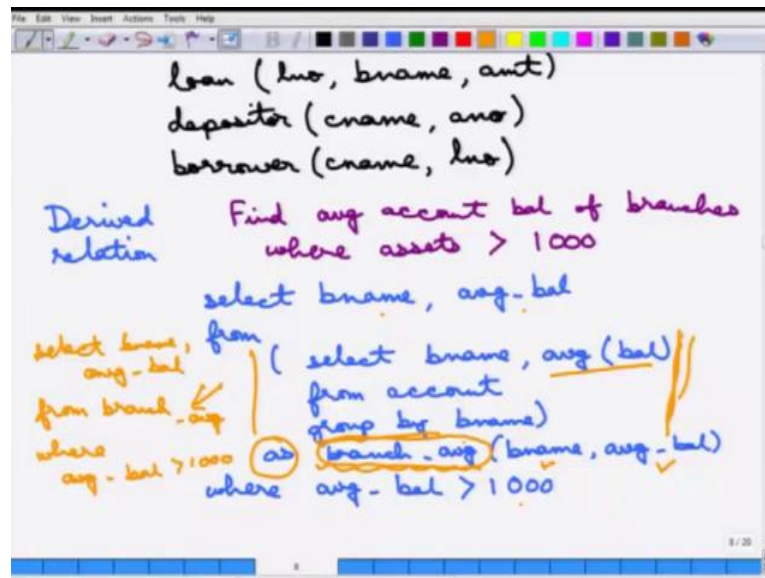
They may be nested and these are 6 clauses are select from these two are mandatory. So, these two are mandatory this must be there and the others may or may not be there and it must be specified in this particular order select from, where group by having order by. So, this is select essentially is on a particular attribute this is the summary of the attribute list from is the relation list, where is predicate group by is again on attribute, list having is again group condition, group predicate and order by is again an attribute list.

Note that this is the syntax order AQL query must happen in this syntax; however, this is very, very different from the execution order. So, what I mean by the execution order is that suppose a particular SQL query with this following 6 clauses are being queried. Now, the question is how does the database engine solve it does it do the select first it really cannot, because unless a form it knows, which relation the form has to come it cannot do it.

So, in these 6 clauses it is the syntax order is not the same as the execution order and I leave it to you as an exercise as to figure out, what can be the execution order of the 6 clauses of the basic SQL query. So, let us move ahead to this something else called the derived relations.



(Refer Slide Time: 14:09)

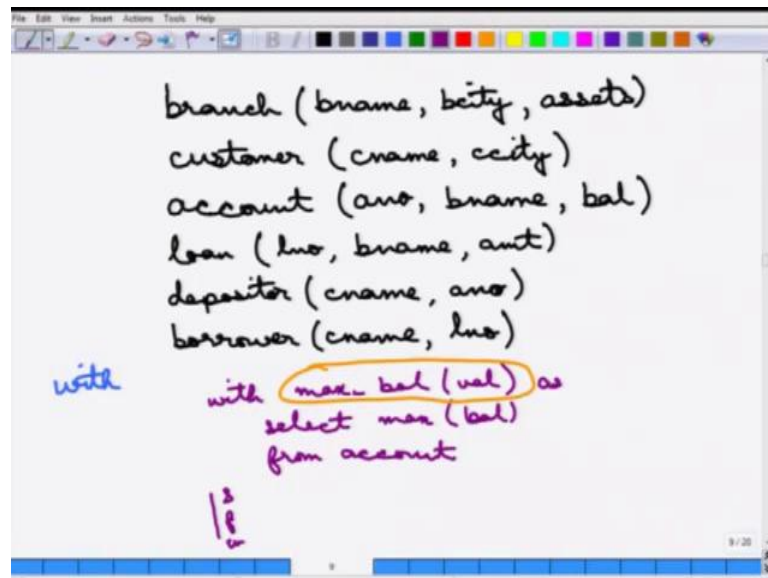


So, the derived relation something is used in the middle of the query, so same like a nested query etcetera and, so for example, let us takes this particular example. So, how do we solve it essentially there is a, we can do it in the following manner, so let us see how this is being solved. So, first of all, let us look at the inner query the inner query essentially just selects the branch name and the average balance of that branch name, so it is grouped by the branch name, so each branch name it finds the average balance.

Now, this as the answer to this query is branch average is the derived relation. So, this is a new name; that is given to this query, which is the branch average with the attributes name as B name and average underscore balance, so we are essentially renaming that. So, this query, now becomes of the following format select branch name average balance from branch name average balance from branch average of this, where average balance is greater than1000.

So, this falls down to select the same thing branch name average balance from this branch average is, what we have been using, where average balance is greater than1000. So, this merges on to this, because a relation is being derived as the part of the query answering first relation is derived, then; that is being used in a outer query.

(Refer Slide Time: 15:42)



And just to go over a little bit of the syntax of this there is a with clause in SQL, which defines a temporary relation. So, you can define with, so you can write other query some select from, where etcetera, etcetera. So, that completes the query part of the SQL, so we have covered all the query parts next we will cover the updating, so insertion deletion and updating how to modify the tuples in SQL.