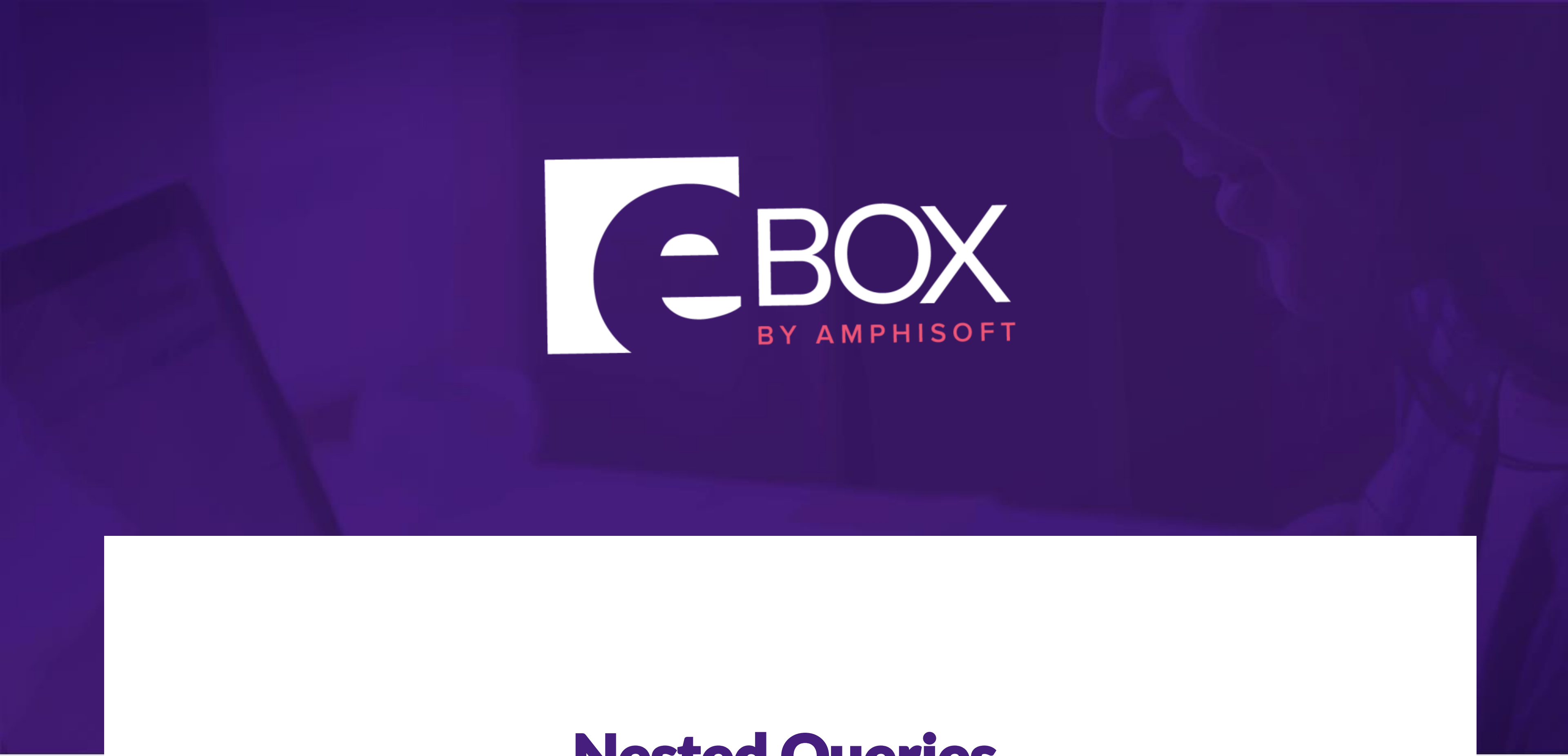


eBOX BY AMPHISOFT

Nested Queries

In this session, you will learn:

- What is Subquery?
- How to use subquery to write complex queries

What is Subquery?

-- nested inside a larger query.

-- occur in

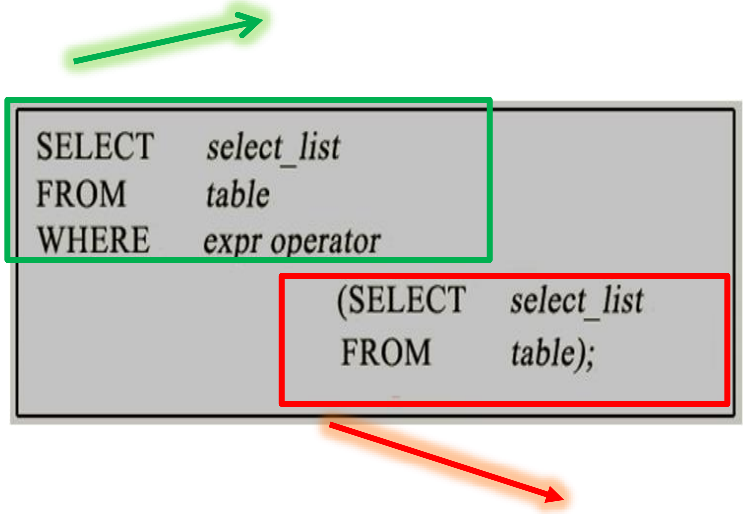
- ➢ A SELECT clause
- ➢ A FROM clause
- ➢ A WHERE clause
- ➢ A HAVING clause

-- can be nested inside a

- ➢ A SELECT statement
- ➢ A INSERT statement
- ➢ A UPDATE statement
- ➢ A DELETE statement

Syntax

Outer Query(Main Query)



```
SELECT    select_list
FROM      table
WHERE     expr operator
                  (SELECT    select_list
                   FROM      table);
```

Inner Query(SubQuery)

Rules for writing Subqueries

- Must be enclosed within parentheses.
- A subquery must be placed on the right side of the comparison operator.
- ORDER BY clause cannot be added into a subquery.
- Use single-row operators with single-row subqueries.

Example for Subquery

Product

Product_Id	Pdt_Name
300	Toys
301	Rhymes
302	Shirt

Select Sold_Out from Product_Sold where Product_Id = 302

Sold_Out
5

Product_Sold

Product_Id	Sold_Out
300	10
301	4
302	5

Query to display Product_Id, Pdt_Name and Sold_Out of those products which are sold better than the product with id 302.

1

Query to return the sold_out of product id 302 from Product_Sold table

2

Query to identify the products which are sold better than the result of the first query

Example for Subquery

Query to identify the products which are sold better than the result of the first query

```
Select Product_Id, Pdt_Name, Sold_Out
from Product, Product_Sold
where Product_Id=Product_Id
and Sold_Out>5
```

Product _Id	Pdt_Name	Sold_Out
300	Toys	10
301	Rhymes	10
302	Shirt	10

ORA-00918: column ambiguously defined

Product

Product_Id	Pdt_Name
300	Toys
301	Rhymes
302	Shirt

Product_Sold

Product_Id	Sold_Out
300	10
301	4
302	5

```
Select Product_Id, Pdt_Name, Sold_Out from Product, Product_Sold
where Product_Id=Product_Id and Sold_Out>5
```

```
Select Product.Product_Id,Pdt_Name,Sold_Out
from Product, Product_Sold
where Product.Product_Id = Product_Sold.Product_Id
and Sold_Out>5
```

```
    Select p.Product_Id,Pdt_Name,Sold_Out
    from Product p, Product_Sold s
    Where p.Product_Id = s.Product_Id
    and Sold_Out > 5
```

Product

Product_Id	Pdt_Name
300	Toys
301	Rhymes
302	Shirt

Product_Sold

Product_Id	Sold_Out
300	10
301	4
302	5

Example for Subquery

Query to display Product_Id, Pdt_Name and Sold_Out of those products which are sold better than the product with id 302.

1

```
Select Sold_Out from
Product_Sold where
Product_Id = 302
```

2

```
Select p.Product_Id, Pdt_Name, Sold_Out
from Product p, Product_Sold s
where p.Product_Id = s.Product_Id and
Sold_Out>5
```

```
Select p.Product_Id, Pdt_Name, Sold_Out from Product p,
Product_Sold s where p.Product_Id = s.Product_Id and
Sold_Out >5  (Select Sold_Out from Product_Sold
                    where Product_Id = 302)
```

Using Subqueries with INSERT Statement

Product_Info

Product_Id	Pdt_Name	Sold_Out
300	Toys	10
301	Rhymes	4
302	Shirt	5

Query to insert records which have the Sold_Out between 5 and 10 into 'good_moving' table

Insert into good_moving
(Select * from Product_Info where Sold_Out BETWEEN 5 AND 10)

good_moving

Product_Id	Pdt_Name	Sold_Out
300	Toys	10
302	Shirt	5

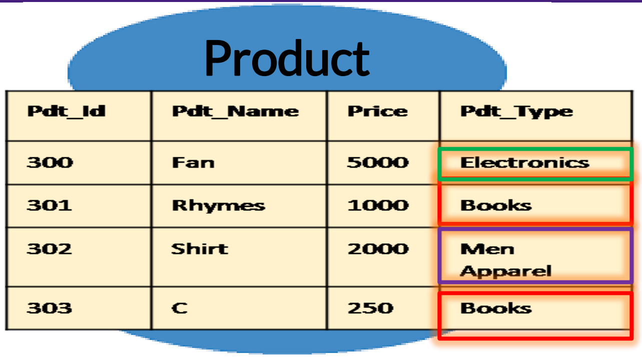
Grouping Data

- GROUP BY clause groups a set of rows into a set of summary rows by values of columns or expressions.

Syntax

```
SELECT
   c1, c2,..., cn, aggregate_function(ci)
FROM
   table
WHERE
   where_conditions
GROUP BY c1 , c2,...,cn;
```

Grouping Data – Example



Product

Pdt_Id	Pdt_Name	Price	Pdt_Type
300	Fan	5000	Electronics
301	Rhymes	1000	Books
302	Shirt	2000	Men Apparel
303	C	250	Books

Example

```
SELECT
  Pdt_Type, count(*) AS NO_OF_PRODUCTS from Product
  GROUP BY Pdt_Type ORDER BY Pdt_Type;
```

$\downarrow$

Pdt_Type	NO_OF_PRODUCTS
Books	2
Electronics	1
Men Apparel	1

Using Subqueries in the HAVING clause

- HAVING clause is used to filter groups of rows.
- Placing a subquery in HAVING clause in an outer query allows to filter groups of rows based on the result returned from subquery,

Product_Orders

Order_ Id	Pdt_Id	Quantity	Discount
200	300	1	0
200	301	5	0.1
201	300	1	0.2
202	302	4	0
204	301	5	0.3

Query to display product id and total quantity of the products whose total quantity ordered is greater than the total quantity ordered by the product with id 302.

```
Select Pdt_Id, Sum(Quantity) Total_Quantity
from Product_Orders GROUP BY Pdt_Id
HAVING Sum(Quantity) >    4
          (Select Sum(Quantity) from
          Product_Orders where Pdt_Id = 302)
```

Pdt_Id	Total_Quantity
301	10

THANKS

eBOX
BY AMPHISOFT