LAB/ PRACTICAL

Lab

SUBQUERIES OR NESTED QUERIES

Learning Outcomes

By the end of this session student should be able to:

Understand and know what is Relational Algebra (RA) for Subqueries or Nested Queries
Transform from RA to SQL or vice versa

In this exercise, we will practice some functions provided in Oracle for DML - SubQueries Operation.

A subquery is a SQL query within a query. Subqueries are nested queries that provide data to the enclosing query. Subqueries can return individual values or a list of records. Subqueries must be enclosed with parenthesis.

A parenthesized SELECT-FROM-WHERE statement (subquery) can be used as a value in a number of places, including FROM and WHERE clauses

EXAMPLE

Query:

Find staff who worked in the branch at 163 Main St.

We can get this information from two tables which are STAFF table and BRANCH table. First we must get to know which branch that is located at 163 Main St.

RESULT <-- πBranchNo (σ Street = '163 Main St.' (BRANCH))

The result of this query will be an input for the second query to list the information about the staff.

π StaffN, FName, LName, Position (σ BranchNo = RESULT (STAFF))

The first query we called as inner query (sub-query) which be input to WHERE condition for the outer query.

If **SELECT-FROM-WHERE** statements without sub-queries it will be like as follow:

RA:

πStaffNo, FName, LName, Position, BranchNo (σStaff.BranchNo = Branch.BranchNo ^

Street = '163 Main St' (BRANCH X STAFF))

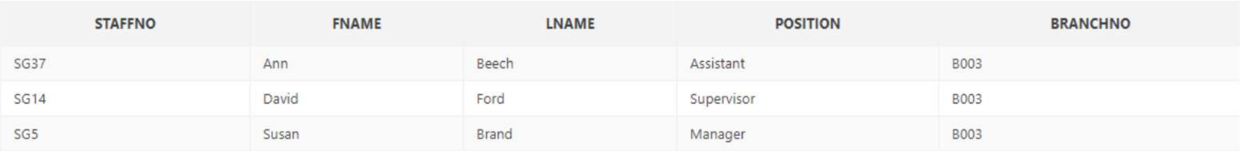
SQL

```
SELECT Staff.StaffNo, Staff.FName, Staff.LName, Staff.Position, Branch.BranchNo
JOIN Branch
ON (Staff.BranchNo = Branch.BranchNo)
WHERE Branch.Street = '163 Main St'
```

OR

```
SELECT StaffNo, FName, LName, Position, BranchNo
FROM Staff
JOIN Branch
USING (BranchNo)
WHERE Street = '163 Main St'
```

RESULT



STAFFNO	FNAME	LNAME	POSITION	BRANCHNO
SG37	Ann	Beech	Assistant	B003
SG14	David	Ford	Supervisor	B003
SG5	Susan	Brand	Manager	B003

The same queries when applying Sub-Queries (or nested queries) as such below:

RA:

πStaffNo, FName, LName, Position, BranchNo

(σStaff.BranchNo = Branch.BranchNo ^ Street = '163 Main St' (BRANCH X STAFF))

SQL

```
SELECT StaffNo, FName, LName, Position
FROM Staff
WHERE BranchNo = (
    SELECT BranchNo
    FROM Branch
    WHERE Street = '163 Main St')
```

Inner SELECT finds BranchNo in BRANCH table for the branch that is located at 163 Main St that yield ('B003').

The outer SELECT then retrieves details of staff (StaffNo, First Name, Last Name, Position) who work at the B003 branch.

The outer SELECT then becomes:

```
SELECT StaffNo, FName, LName, Position
FROM Staff
WHERE BranchNo = 'B003'
```

Please note here, the information of **B003 is a result of the inner query.**

Query:

Find all staff whose salary is greater than average salary and show by how much.

We can get this information from the STAFF table. First, we must get to know what is the average salary for the staff.

STAFFAVGSAL <-- ƒAVG Salary (STAFF)

The result of this query (STAFFAVGSAL) will be an input for the second query to list the information about the staff.

ρr (StaffN, FName, LName, Position, DifSal) (π StaffN, FName, LName, Position, Salary - STAFFACGSAL (σ Salary > STAFFAVGSAL (STAFF)))

The first query we called as inner query (sub-query) which be input to WHERE condition for the outer query.

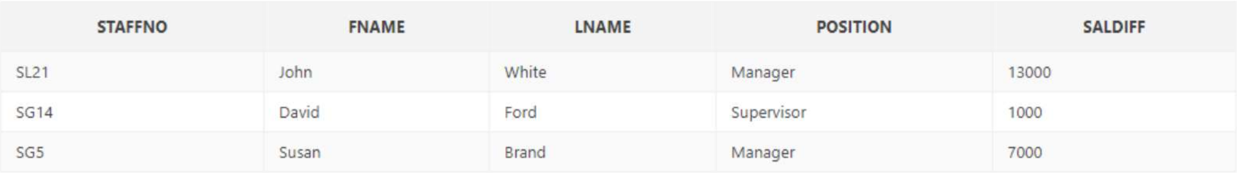
RA:

ρr (StaffN, FName, LName, Position, DifSal) (π StaffN, FName, LName, Position, Salary - ƒAVG Salary (STAFF) (σ Salary > ƒAVG Salary (STAFF) (STAFF))

SQL

```
SELECT StaffNo, FName, LName, Position, Salary - (Select AVG(Salary) FROM STAFF) as SalDiff
FROM Staff
WHERE Salary > (
    SELECT AVG(Salary)
    FROM Staff)
```

RESULT



STAFFNO	FNAME	LNAME	POSITION	SALDIFF
SL21	John	White	Manager	13000
SG14	David	Ford	Supervisor	1000
SG5	Susan	Brand	Manager	7000

In this example, we cannot write ''. WHERE Salary > AVG(Salary)'. Instead, we need to use a subquery to find the average salary (that is 17,000) and then use the outer SELECT statement to find those staff with salaries greater than this value.

The outer SELECT then becomes:

```
SELECT StaffNo, FName, LName, Position, Salary - 17000 as salDiff
FROM Staff
WHERE Salary > 17000
```

Please note here, the information of 17000 is a result of the inner query and aggregation average operation.

Normalizing Where-subqueries into Exists and Not Exists or IN form

From the above example, sub-queries can occur in the WHERE clause through the operators =, <, >, <=, >=, <>; through the quantifiers ANY, or ALL; or through the operators EXISTS and IN and their negations NOT EXISTS and NOT IN.

Query:

Find details info of staff who worked at London city

RA:

π StaffNo, FName, LName, Position, BranchNo (σ City = 'London' (BRANCH X STAFF))

SQL (using IN)

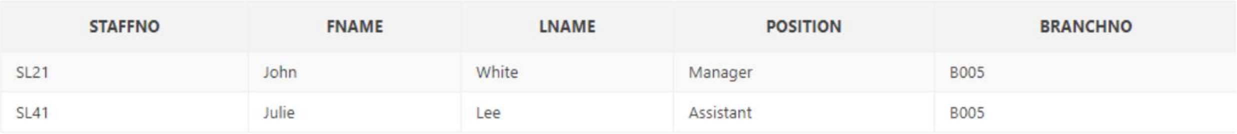
```
SELECT StaffNo, FName, LName, Position, BranchNo
FROM Staff
WHERE BranchNo IN (
   SELECT BranchNo
   FROM Branch
   WHERE City = 'London')
```

Can be rewritten equivalently as

SQL (using EXISTS)

```
SELECT StaffNo, FName, LName, Position, BranchNo
FROM Staff
WHERE EXISTS (
   SELECT BranchNo
   FROM Branch
   WHERE City = 'London'
   AND Staff.BranchNo = Branch.BranchNo)
```

RESULT



STAFFNO	FNAME	LNAME	POSITION	BRANCHNO
SL21	John	White	Manager	B005
SL41	Julie	Lee	Assistant	B005

NESTED SUB-QUERY

Query:

List properties handled by Staffs at 163 Main St.

RA:

π PropertyNo, Street, City, Postcode, Type, Rooms, Rent (BRANCH (σ Staff.BranchNo = Branch.BranchNo (STAFF ⋈(σ Propertyforrent.StaffNo = Staff.StaffNo ^ Branch.Street = '163 Main St' (PROPERRTYFORRENT)))))

OR

π PropertyNo, Street, City, Postcode, Type, Rooms, Rent (σ Staff.BranchNo = Branch.BranchNo ^ Propertyforrent.StaffNo = Staff.StaffNo ^ Branch.Street = '163 Main St' (BRANCH X STAFF X PROPERTYFORRENT))

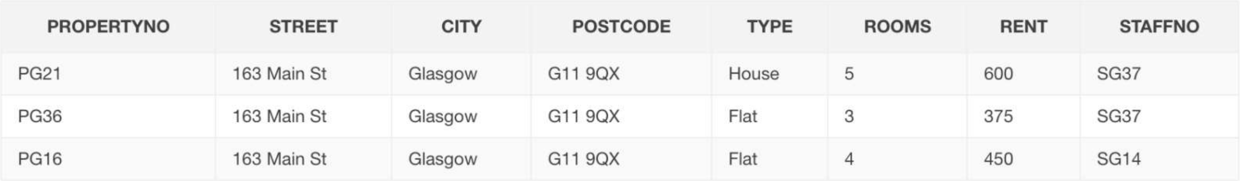
SQL

```
SELECT propertyNo, street, city, postcode, type, rooms, rent
FROM PropertyForRent
WHERE staffNo IN
(SELECT staffNo
FROM Staff
WHERE branchNo =
(SELECT branchNo
FROM Branch
WHERE street = ‘163 Main St’));
```

OR

```
SELECT p.propertyNo, p.street, p.city, p.postcode, p.type, p.rooms, p.rent, s.staffno
FROM PropertyForRent p
JOIN Staff s
ON (p.StaffNo = s.StaffNo)
JOIN Branch b
ON (b.Branchno = s.Branchno)
WHERE b.Street = '163 Main St'
```

RESULT



PROPERTYNO	STREET	CITY	POSTCODE	TYPE	ROOMS	RENT	STAFFNO
PG21	163 Main St	Glasgow	G11 9QX	House	5	600	SG37
PG36	163 Main St	Glasgow	G11 9QX	Flat	3	375	SG37
PG16	163 Main St	Glasgow	G11 9QX	Flat	4	450	SG14

In this query, we get that the list of properties handled by staff (that are PropertyNo PG21, PG36, and PG16) which located at 163 Main St.

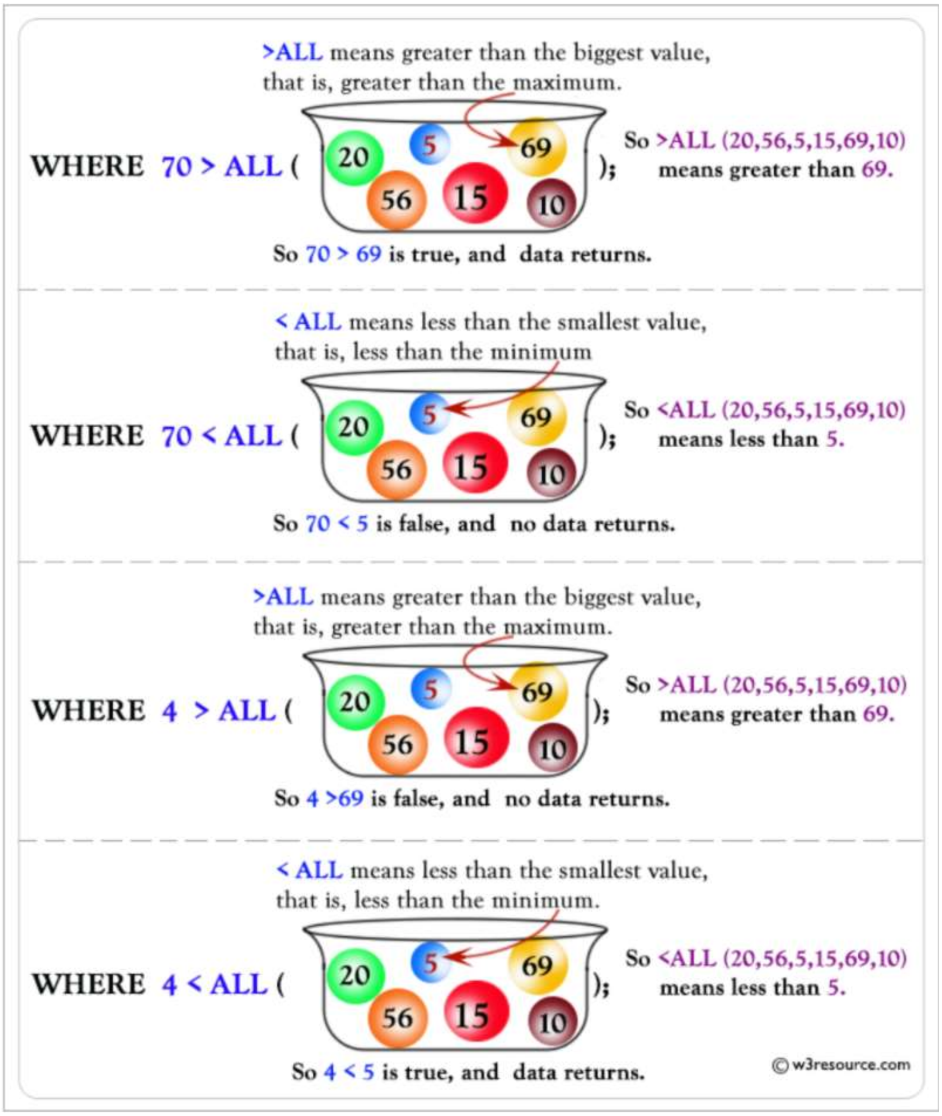
Sub-Queries (ANY and ALL)

ANY and ALL may be used with sub-queries that produce a single column of numbers.

ALL

ALL is used to select all records of a SELECT STATEMENT. It compares a value to every value in a list or results from a query. The ALL must be preceded by the comparison operators and evaluates to TRUE if the query returns no rows. For example, ALL means greater than every value, means greater than the maximum value. Suppose ALL (1, 2, 3) means greater than 3.

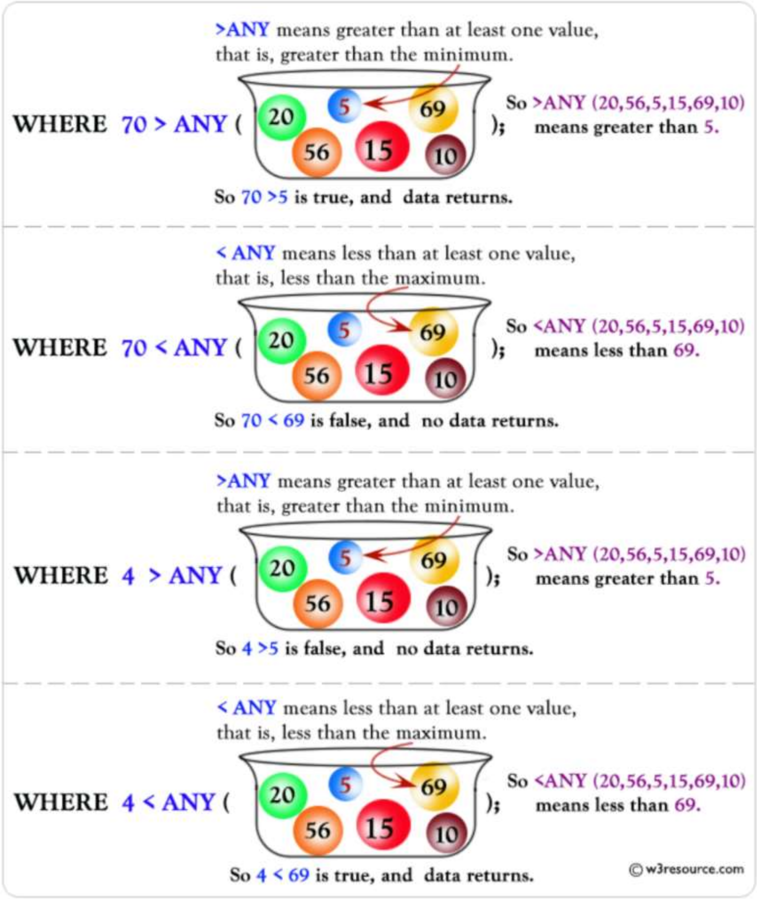
Pictorial Presentation of ALL operator.


>ALL means greater than the biggest value,
that is, greater than the maximum.
WHERE 70 > ALL (
20
5
69
56
15
10
);
So >ALL (20,56,5,15,69,10)
means greater than 69.
So 70 > 69 is true, and data returns.
< ALL means less than the smallest value,
that is, less than the minimum
WHERE 70 < ALL (
20
5
69
56
15
10
);
So <ALL (20,56,5,15,69,10)
means less than 5.
So 70 < 5 is false, and no data returns.
>ALL means greater than the biggest value,
that is, greater than the maximum.
WHERE 4 > ALL (
20
5
69
56
15
10
);
So >ALL (20,56,5,15,69,10)
means greater than 69.
So 4 >69 is false, and no data returns.
< ALL means less than the smallest value,
that is, less than the minimum.
WHERE 4 < ALL (
20
5
69
56
15
10
);
So <ALL (20,56,5,15,69,10)
means less than 5.
So 4 < 5 is true, and data returns.
© w3resource.com

ANY

ANY or SOME compares a value to each value in a list of results from a query and evaluates to true if the result of an inner query contains at least one row. ANY must be preceded by comparison operators. Suppose using greater than (>) with ANY means greater than at least one value.

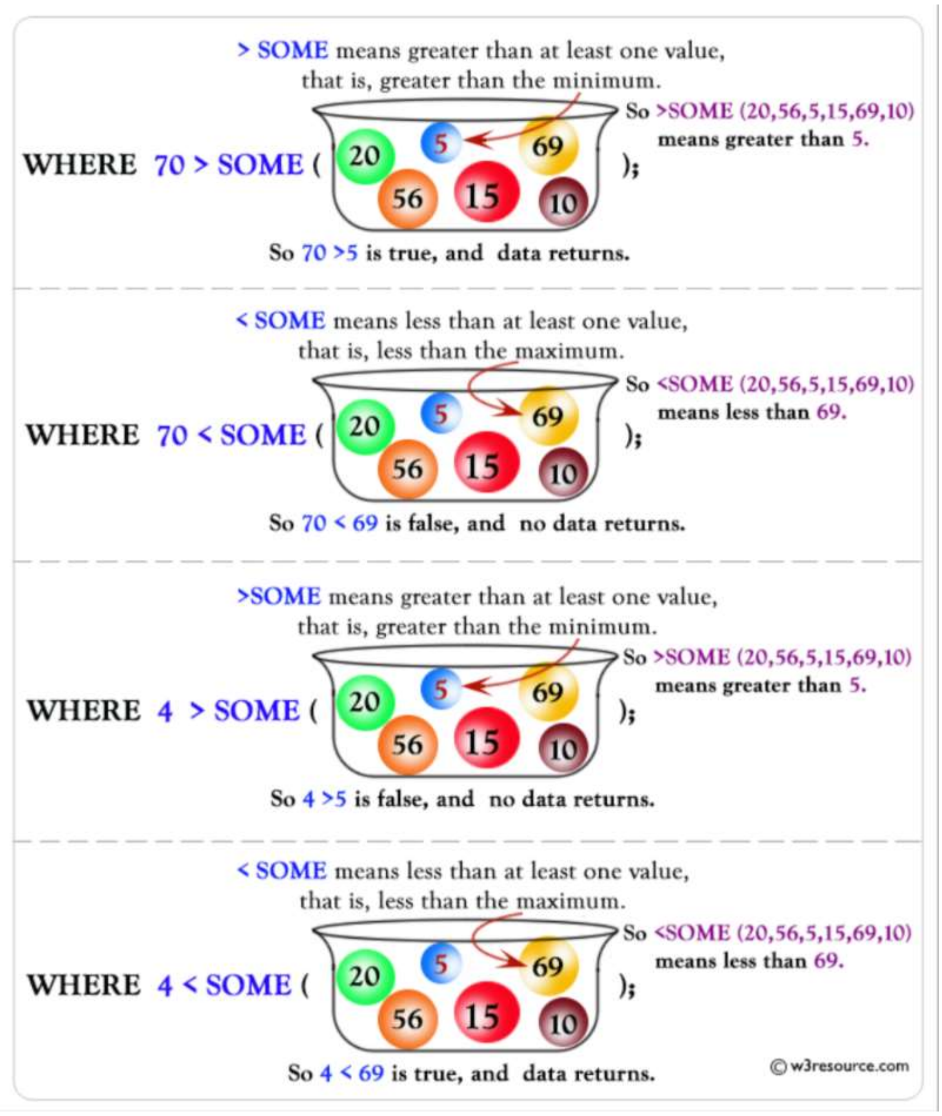
Pictorial Presentation of ANYoperator.


>ANY means greater than at least one value,
that is, greater than the minimum.
WHERE 70 > ANY (20 5 69 56 15 10);
So >ANY (20,56,5,15,69,10) means greater than 5.
So 70 >5 is true, and data returns.
< ANY means less than at least one value,
that is, less than the maximum.
WHERE 70 < ANY (20 5 69 56 15 10);
So <ANY (20,56,5,15,69,10) means less than 69.
So 70 < 69 is false, and no data returns.
>ANY means greater than at least one value,
that is, greater than the minimum.
WHERE 4 > ANY (20 5 69 56 15 10);
So >ANY (20,56,5,15,69,10) means greater than 5.
So 4 >5 is false, and no data returns.
< ANY means less than at least one value,
that is, less than the maximum.
WHERE 4 < ANY (20 5 69 56 15 10);
So <ANY (20,56,5,15,69,10) means less than 69.
So 4 < 69 is true, and data returns.
© w3resource.com

SOME

SOME compare a value to each value in a list or results from a query and evaluate to true if the result of an inner query contains at least one row. SOME must match at least one row in the subquery and must be preceded by comparison operators. Suppose using greater than (>) with SOME means greater than at least one value.

Pictorial Presentation of SOME operator.


> SOME means greater than at least one value,
that is, greater than the minimum.
WHERE 70 > SOME (20 5 69 56 15 10);
So >SOME (20,56,5,15,69,10)
means greater than 5.
So 70 >5 is true, and data returns.
< SOME means less than at least one value,
that is, less than the maximum.
WHERE 70 < SOME (20 5 69 56 15 10);
So <SOME (20,56,5,15,69,10)
means less than 69.
So 70 < 69 is false, and no data returns.
>SOME means greater than at least one value,
that is, greater than the minimum.
WHERE 4 > SOME (20 5 69 56 15 10);
So >SOME (20,56,5,15,69,10)
means greater than 5.
So 4 >5 is false, and no data returns.
< SOME means less than at least one value,
that is, less than the maximum.
WHERE 4 < SOME (20 5 69 56 15 10);
So <SOME (20,56,5,15,69,10)
means less than 69.
So 4 < 69 is true, and data returns.
© w3resource.com

EXAMPLE (ANY)

Query:

Find staff whose salary is larger than the salary of at least one member of staff at Branch B003.

RA:

πStaffNo, FName, LName, Position, Salary, BranchNo (σ salary > ƒMIN Salary ^ BranchNo = 'B003' (STAFF))

SQL (Sub-Queries using ANY):

```
SELECT StaffNo, FName, LName, Position, Salary, BranchNo
FROM Staff
WHERE Salary > ANY
(SELECT Salary
FROM Staff
WHERE BranchNo = 'B003')
```

From this query, the inner query produces set {12000, 18000, 24000}. So the outer query will select those staff whose salaries are greater than any of the values in this set which results as shown below. Or in other words, it will produce a result in which values are greater than a minimum value in the set (i.e. 12000).

RESULT

STAFFNO	FNAME	LNAME	POSITION	SALARY	BRANCHNO
SL21	John	White	Manager	30000	B005
SG5	Susan	Brand	Manager	24000	B003
SG14	David	Ford	Supervisor	18000	B003

EXAMPLE (ALL)

Query:

Find staff whose salary is larger than the salary of every member of staff at Branch B003.

RA:

πStaffNo, FName, LName, Position, Salary, BranchNo (σ salary > ƒMAX Salary ^ BranchNo = 'B003' (STAFF)))

SQL (Sub-Queries using ALL):

```
SELECT StaffNo, FName, LName, Position, Salary, BranchNo
FROM Staff
WHERE Salary > ALL
(SELECT Salary
FROM Staff
WHERE BranchNo = 'B003')
```

From this query, the inner query produces set {12000, 18000, 24000}. So the outer query will select those staff whose salaries are greater than any of the values in this set which results as shown below. Or in other words, it will produce a result in which values are greater than a maximum value in the set (i.e. 24000).

RESULT

STAFFNO	FNAME	LNAME	POSITION	SALARY	BRANCHNO
SL21	John	White	Manager	30000	B005

EXAMPLE (SOME)

Query:

Find staff whose salary is larger than the salary of some members of staff at Branch B003.

SQL (Sub-Queries using SOME):

```
SELECT StaffNo, FName, LName, Position, Salary, BranchNo
FROM Staff
WHERE Salary > SOME
(SELECT Salary
FROM Staff
WHERE BranchNo = 'B003')
```

From this query, the inner query produces set {12000, 18000, 24000}. So the outer query will select those staff whose salaries are greater than any of the values in this set which results as shown below. Or in other words, it will produce a result in which values are greater than a minimum at least one value in the set (i.e. 12000). SOME may be used in place of ANY

RESULT

STAFFNO	FNAME	LNAME	POSITION	SALARY	BRANCHNO
SL21	John	White	Manager	30000	B005
SG5	Susan	Brand	Manager	24000	B003
SG14	David	Ford	Supervisor	18000	B003