GROUPING

GROUPING

- In some situations, we may want to divide entities into different groups and apply the summary function to each group separately
- **Examples:**
  - What is the total target of each region?
  - What are the maximum, minimum and the average price of parts provided by each manufacturer?
  - Etc.
- These questions can be answered using grouping

GROUPING

- Basic operation of grouping:
  - A column (or multiple columns) are selected by the user
  - Rows of the table having the same value in the selected collumns are grouped into separate tables
  - Aggregate functions are applied to each group

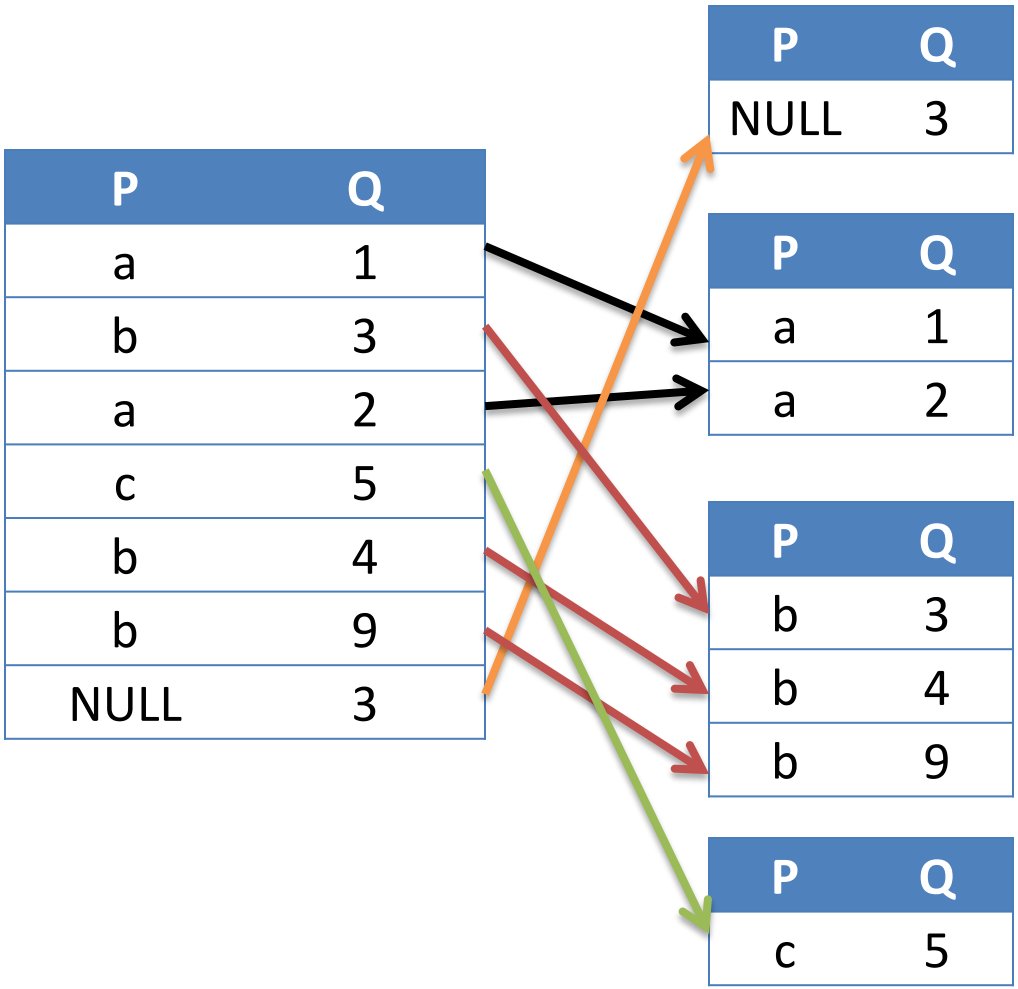
GROUPING

- **Example:** Consider the following table

P	Q
a	1
b	3
a	2
c	5
b	4
b	9
NULL	3

GROUPING

- Grouping of rows with respect to column P



P	Q
a	1
b	3
a	2
c	5
b	4
b	9
NULL	3

P	Q
NULL	3

P	Q
a	1
a	2

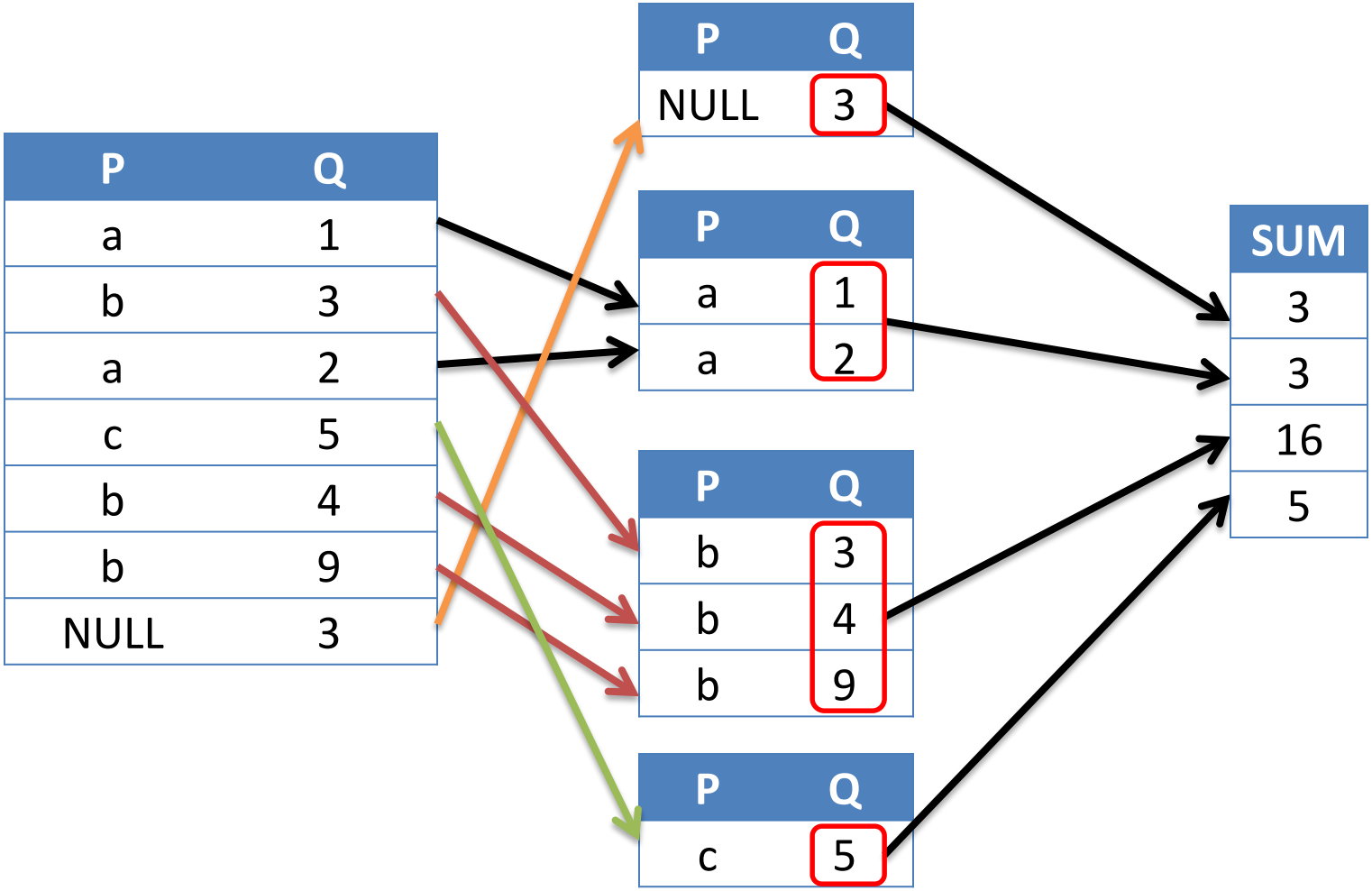
P	Q
b	3
b	4
b	9

P	Q
c	5

GROUPING

- An aggregate function is applied to each group
- For example SUM()
- **Result is a table** where each row has the result of the aggregate function for each group

GROUPING

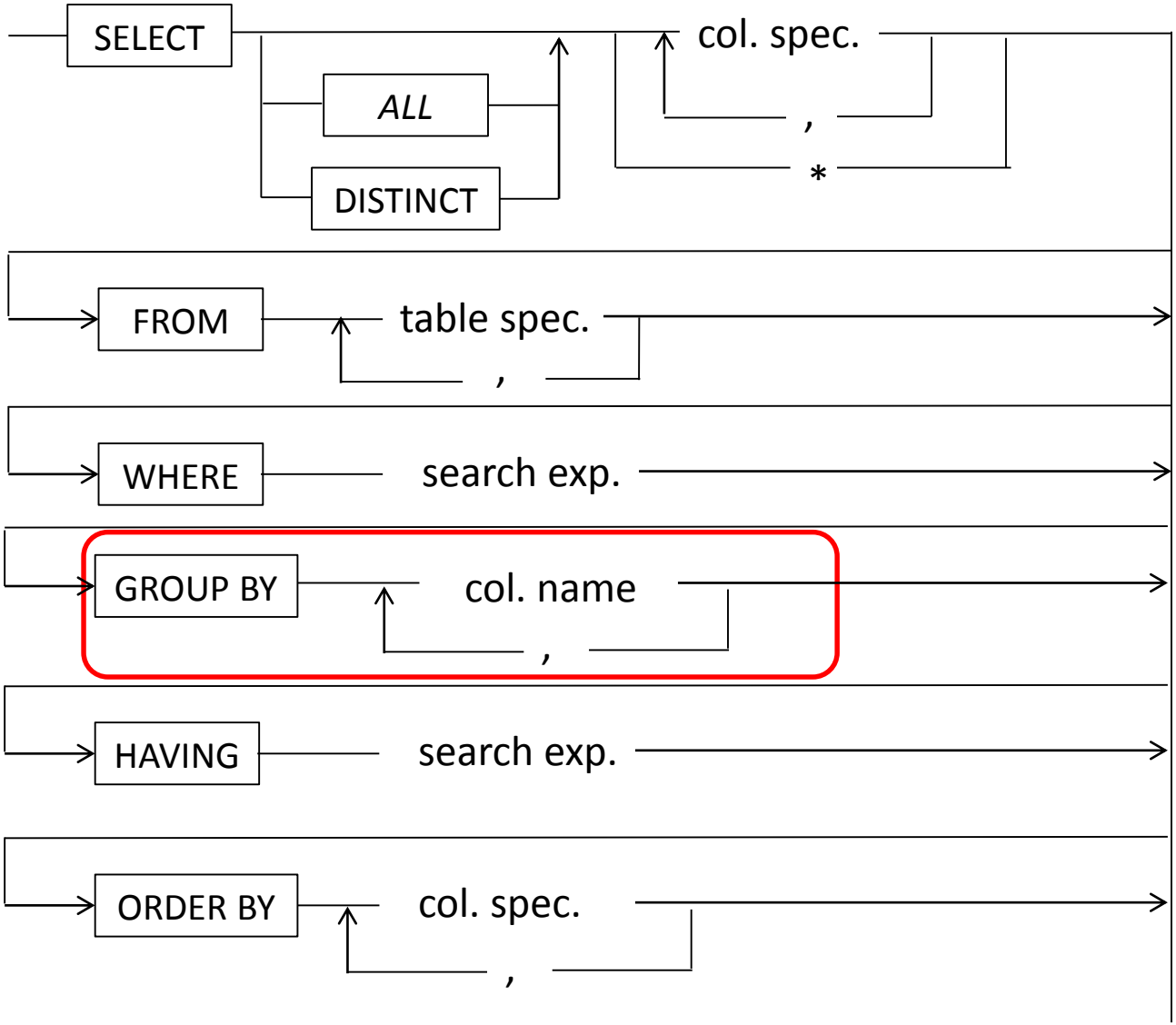

P
Q
a
1
b
3
a
2
c
5
b
4
b
9
NULL
3
P
Q
NULL
3
P
Q
a
1
a
2
P
Q
b
3
b
4
b
9
P
Q
c
5
SUM
3
3
16
5

GROUPING

- Grouping **does not return the tables** corresponding to groups
- It is always used with aggregate functions and **returns the result of aggreagate functions** for each group
- Hence, the result will always has as many rows as the number of groups
- In SQL, grouping can be performed using GROUP BY clause

GROUPING

- In SQL, grouping can be performed using GROUP BY clause



GROUPING

- **Example:** What is the total target of each region?

```
SELECT SUM(OFF_TARGET)
FROM OFFICES
GROUP BY REGION
```

GROUPING

- Result:
  - 1465000,00
  - 638500,00
- **But which result belongs to which group?**
- We can use the name of the grouping column in column selection list
- This will allow us to identify the groups

GROUPING

- **Example:**

```
SELECT REGION, SUM(OFF_TARGET)
FROM OFFICES
GROUP BY REGION
```

- Result:

Northern	1465000,00
Southern	638500,00

GROUPING

- In column specification (i.e. Just after SELECT), we are only allowed to use:
  - constants
  - aggregate functions
  - names of grouping columns
- **NOTHING ELSE!!!!**

GROUPING

- More than one aggregate function can be used simultaneously
- In the result, there will be a column for each aggregate function

GROUPING

- **Example:** What are the maximum, minimum and the average price of parts provided by each manufacturer?

GROUPING

- **Example:** What are the maximum, minimum and the average price of parts provided by each manufacturer?

```
SELECT MAN_ID, MAX(PRICE), MIN(PRICE), AVG(PRICE)
FROM PRODUCTS
GROUP BY MAN_ID
```

GROUPING

- Result:

MAN_ID	No name	No name	No name
AX	4500,00	2580,00	3540,00
CHI	9478,00	373,00	3960,50
HAM	360,00	55,00	180,00
MAL	780,00	350,00	592,6666
SAW	985,00	45,00	329,1666
SCR	1000,00	78,00	401,6666

- In the result columns of aggregate function will not have name

GROUPING

- To be able to distinguish columns we can give names using **AS**

```
SELECT    MAN_ID, MAX(PRICE) AS MAX,
          MIN(PRICE) AS MIN, AVG(PRICE) AS AVG
FROM PRODUCTS
GROUP BY MAN_ID
```

GROUPING

- Result:

MAN_ID	MAX	MIN	AVG
AX	4500,00	2580,00	3540,00
CHI	9478,00	373,00	3960,50
HAM	360,00	55,00	180,00
MAL	780,00	350,00	592,6666
SAW	985,00	45,00	329,1666
SCR	1000,00	78,00	401,6666

GROUPING

- **Example:** What is the number of employees working in each office?

GROUPING

- **Example:** What is the number of employees working in each office?

```
SELECT OFFICE, COUNT(*)
FROM EMPLOYEES
GROUP BY OFFICE
```

GROUPING

- MS SQL allows groping to be done with respect to calculated columns
- **Example:** List the total price of sales made each year

GROUPING

- MS SQL allows groping to be done with respect to calculated columns
- **Example:** List the total price of sales made each year

```
SELECT    YEAR(ORDER_DATE) AS YEAR,
          SUM(ORD_PRICE) AS TOTAL
FROM ORDERS
GROUP BY YEAR(ORDER_DATE)
```

GROUPING

- Grouping can be done with respect to more than one column
- In this case, the rows having the same value in all these columns are put into the same group

GROUPING

- **Example:** List manufacturer ID and product ID of each product together with total price of orders made to the product

```
SELECT MAN, PROD, SUM(ORD_PRICE)
FROM ORDERS
GROUP BY MAN,PROD
```

GROUPING

- Grouping can be used with other operations:
  - Row selection
  - Cartesian product
  - Join
- First these operations are performed, then the grouping is applied to the resulting table
- One should keep in mind this while writing complex queries involving these operations

GROUPING

- Grouping can be used with other operations:
  - Row selection
  - Cartesian product
  - Join
- First these operations are performed, then the grouping is applied to the resulting table
- One should keep in mind this while writing complex queries involving these operations

GROUPING

- **Example:** For each office, print the ID number of the office and the number of employees working as ‘Sales Rep’ at that office

GROUPING

- **Example:** For each office, print the ID number of the office and the number of employees working as ‘Sales Rep’ at that office

```
SELECT OFFICE, COUNT(*)
FROM EMPLOYEES
WHERE TITLE='Sales Rep'
GROUP BY OFFICE
```

GROUPING

- **Example:** List the name of each employee and the total price of orders taken by him/her

GROUPING

- **Example:** List the name of each employee and the total price of orders taken by him/her

```
SELECT FL_NAME, SUM(ORD_PRICE) AS
            TOTAL_ORDER
FROM ORDERS, EMPLOYEES
WHERE REP_NUM=EMP_ID
GROUP BY FL_NAME
```

GROUPING

- Employees who did not take any order will not appear in the previous result
- What can we do if we want to see all employees even if they did not take any order?

GROUPING

- Employees who did not take any order will not appear in the previous result
- What can we do if we want to see all employees even if they did not take any order?

```
SELECT FL_NAME, SUM(ORD_PRICE) AS
            TOTAL_ORDER
FROM EMPLOYEES LEFT OUTER JOIN ORDERS
ON REP_NUM=EMP_ID
GROUP BY FL_NAME
```

GROUPING

- **Example:** List the total number and price of orders taken by each employee separately for each year. In the list show employees by their names.

GROUPING

- **Example:** List the total number and price of orders taken by each employee separately for each year. In the list show employees by their names.

```
SELECT    FL_NAME, YEAR(ORDER_DATE),
          SUM(ORD_PRICE) AS TOTAL_ORDER,
          COUNT(*) AS NUMBER
FROM ORDERS, EMPLOYEES
WHERE REP_NUM=EMP_ID
GROUP BY FL_NAME, YEAR(ORDER_DATE)
```

GROUPING

- **Example:** List the ID number and region of each office together with the number of employees working at that office

```
SELECT OFFICE, REGION, COUNT(*)
FROM EMPLOYEES, OFFICES
WHERE OFFICE=OFFICE_ID
GROUP BY OFFICE
```

- Unfortunately, this query will not work

GROUPING

- Recall that in column specification, we are only allowed to use aggregate functions and the columns used for grouping
- Even if *REGION* is a common value for all group members, this rules prevents us from using *REGION* in select list
- We can work around this problem by adding *REGION* to grouping column list

GROUPING

- This will not effect the result since *REGION* has the same value for all group members

```
SELECT OFFICE, REGION, COUNT(*)
FROM EMPLOYEES, OFFICES
WHERE OFFICE=OFFICE_ID
GROUP BY OFFICE, REGION
```

GROUPING

- **Example:** List the manufacturer id, product id, type, available quantity and total ordered quantity of each product

GROUPING

- **Example:** List the manufacturer id, product id, type, available quantity and total ordered quantity of each product

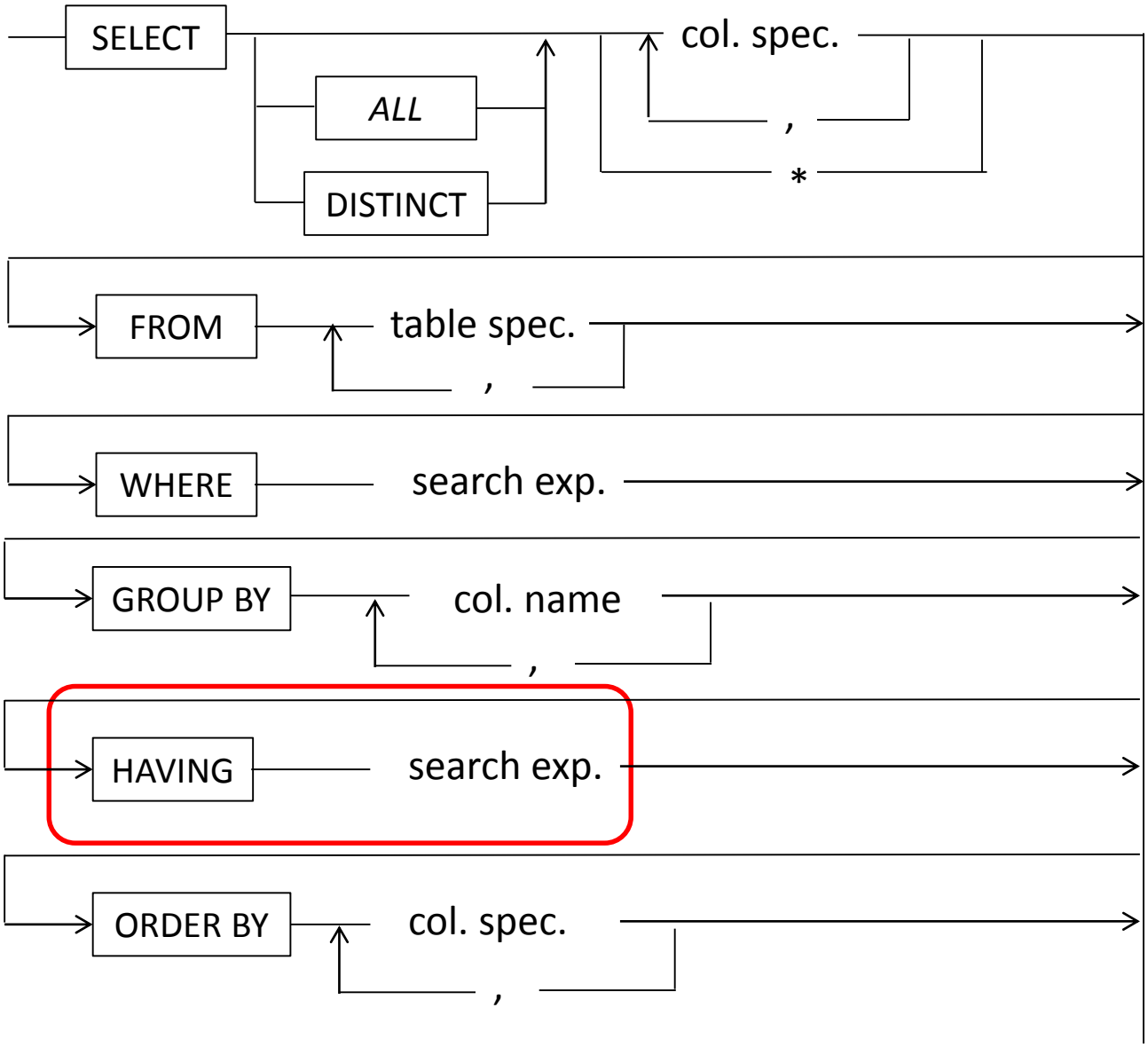
```
SELECT    MAN_ID, PROD_ID, TYPE, AV_QUANT,
          SUM(QUANT)
FROM ORDERS, PRODUCTS
WHERE MAN=MAN_ID AND PROD=PROD_ID
GROUP BY MAN_ID, PROD_ID, TYPE, AV_QUANT
```

GROUP SELECTION

GROUP SELECTION

- It is possible to make selection over the table returned by grouping operation
- This can be achieved using HAVING clause

GROUP SELECTION



GROUP SELECTION

- **Example:**
  - Consider employees the total price of orders taken by whom is more than $30,000
  - List the id number of these employees together with the total price of orders taken by them

```
SELECT REP_NUM, SUM(ORD_PRICE)
FROM ORDERS
GROUP BY REP_NUM
HAVING SUM(ORD_PRICE)>30000
```

GROUPING

- Similar to column specification , in HAVING clause we are only allowed to use:
  - constants
  - aggregate functions
  - names of grouping columns
- **NOTHING ELSE!!!!**

GROUP SELECTION

- **Example:** List the names, targets and number of customers of each employee who has more than 2 customers

GROUP SELECTION

- **Example:** List the names, targets and number of customers of each employee who has more than 2 customers

```
SELECT FL_NAME, EMP_TARGET, COUNT(*)
FROM EMPLOYEES, CUSTOMERS
WHERE CST_REP=EMP_ID
GROUP BY FL_NAME, EMP_TARGET
HAVING COUNT(*)>2
```

GROUP SELECTION

- **Example:** List the manufacturer id, product id, type, available quantity and total ordered quantity of each product whose total ordered quantity is more than the available quantity

GROUP SELECTION

- **Example:** List the manufacturer id, product id, type, available quantity and total ordered quantity of each product whose total ordered quantity is more than the available quantity

```
SELECT    MAN, PROD, TYPE, AV_QUANT,
   SUM(QUANT)
FROM ORDERS, PRODUCTS
WHERE MAN=MAN_ID AND PROD=PROD_ID
GROUP BY MAN, PROD, TYPE, AV_QUANT
HAVING SUM(QUANT) > AV_QUANT
```

GROUP SELECTION

- We may use group selection and row selection together
- Order of operations:
  - Row selection
  - Group selection

GROUP SELECTION

- **Example:**
  - Consider the customers having maximum credit limit more than $30000
  - List the names and targets of employees having at least two such customers together with the number of such customers they have

GROUP SELECTION

- **Example:**
  - Consider the customers having maximum credit limit more than $30000
  - List the names and targets of employees having at least two such customers together with the number of such customers they have

```
SELECT FL_NAME, EMP_TARGET, COUNT(*)
FROM EMPLOYEES, CUSTOMERS
WHERE CST_REP=EMP_ID AND MAX_CREDIT>30000
GROUP BY FL_NAME, EMP_TARGET
HAVING COUNT(*)>=2
```