SELECT
[DISTINCT | ALL] <column-list>
FROM <table-names>
[WHERE <condition>]
[ORDER BY <column-list>]
[GROUP BY <column-list>]
[HAVING <condition>]

[]- ***не е задължително*** **| - *или***

ORDER BY

Клаузата **ORDER BY** сортира, подрежда резултатите от заявката

Можете да подреждате във възходящ и низходящ ред

Може да се изпълни за няколко колони

Не може да се изпълни за колона, която не е в резултатната таблица

SELECT <columns> FROM <tables>WHERE <condition>
ORDER BY <cols>[ASCENDING |DESCENDING|ASC | DESC]

Пример :

SELECT * FROM Grades ORDER BY Mark

Grades

Name	Code	Mark
Иван	112	4
Иван	125	4
Милена	112	5
Мария	136	6
Мария	137	3
Стоян	125	2

ORDER BY Mark

Name	Code	Mark
Стоян	125	2
Мария	137	3
Иван	112	4
Иван	125	4
Милена	112	5
Мария	136	6

Пример :

SELECT * FROM Grades ORDER BY Code ASC, Mark DESC

Grades

Name	Code	Mark
Иван	112	4
Иван	125	4
Милена	112	5
Мария	136	6
Мария	137	3
Стоян	125	2

***ORDER BY Mark* Code ASC, Mark DESC**

Name	Code	Mark
Милена	112	5
Иван	112	4
Иван	125	4
Стоян	125	2
Мария	136	6
Мария	137	3

Константи и аритметика

- Освен с имена на колони, чрез **SELECT** може да се оперира с константи, да се извършват аритметични действия и изчисляват функции.

SELECT 1.5*Mark FROM Grades

SELECT Salary + Bonus FROM Employee

SELECT Price /10 FROM Products

Агрегатни функции

Агрегатните функции изчисляват обобщения върху данните от таблицата

Повечето агрегатни функции (всички освен **COUNT)** **изчисляват върху една** колона с числови данни

Използва се псевдоним за резултата

Агрегатни функции:

COUNT: Брой на редовете
SUM: Сумата от една колона
AVG: Средноаритметичната стойност в една колона
MIN, MAX: Минималната и максималната стойност в една колона

Агрегатни функции

Name	Code	Pnts
Иван	112	56
Иван	125	72
Милена	112	60
Мария	136	43
Мария	137	35
Стоян	125	54

SELECT COUNT(*) AS Count FROM Grades

Count
6

SELECT SUM(Pnts) AS Total FROM Grades

Total
320

SELECT MAX(Pnts) AS Best FROM Grades

Best
72

Агрегатни функции

Name	Code	Pnts
Иван	112	56
Иван	125	72
Милена	112	60
Мария	136	43
Мария	137	35
Стоян	125	54

- Чрез аритметични действия могат да се комбинират агрегатните функции

SELECT MAX(Pnts)-MIN(Pnts) AS Range FROM Grades

MAX(Pnts) = 72

MIN(Pnts) = 35

Range
37

Пример:

Да се намери средната оценка на Мария с теглото на кредита на дисциплината.

Name	Code	Mark
Иван	112	4
Иван	125	4
Милена	112	3
Мария	136	6
Мария	137	4
Стоян	125	2

SELECT
SUM(Mark*Credits)/SUM(Mark)
FROM Disciplines, Grades
WHERE Disciplines.Code=Grades.Code
AND Grades.Name = 'Мария'

Code	Title	Credits
112	Информатика	5
125	…	…

GROUP BY

Понякога се налага да се използват агрегатните функции върху групи от редове

Например: Да се намери средната оценка на всеки студент

Клаузата GROUP BY прави това

SELECT <cols1> FROM <tables> GROUP BY <cols2>

GROUP BY

SELECT <cols1> FROM <tables> GROUP BY <cols2>

- Всеки елемент от **<cols2> трябва да присъства в <cols1>**
- Може да има **WHERE** или **ORDER BY клаузи**

GROUP BY

SELECT Name, AVG(Mark) AS Average FROM Grades GROUP BY Name

Name	Code	Mark
Иван	112	4
Иван	125	4
Милена	112	3
Мария	136	6
Мария	137	4
Стоян	125	2

Name	Average
Иван	4
Милена	3
Мария	5
Стоян	2

GROUP BY

- Да се намери стойността на продажбите за всеки отдел за всеки месец
- Групиране по Month после по Department или Department после Month ???

Month	Department	Value
Юни	Млечни пр.	20
Юни	Хляб	30
Юни	Хляб	40
Юли	Млечни пр.	40
Юли	Млечни пр.	15
Юли	Захарни изд.	25
Юли	Хляб	20
Август	Млечни пр.	50

GROUP BY

Вариант 1

SELECT Month, Department, SUM(Value) AS Total
FROM Sales GROUP BY Month, Department

Month	Department	Total
Юни	Млечни пр.	20
Юни	Хляб	70
Юли	Млечни пр.	55
Юли	Захарни изд.	25
Юли	Хляб	20
Август	Млечни пр.	50

GROUP BY

Вариант 2

SELECT Month, Department, SUM(Value) AS Total
FROM Sales GROUP BY Department, Month

Month	Department	Total
Юли	Захарни изд.	25
Юни	Млечни пр.	20
Юли	Млечни пр.	55
Август	Млечни пр.	50
Юни	Хляб	70
Юли	Хляб	20

HAVING

- HAVING прилича на клаузата WHERE , но се прилага върху резултата от заявката GROUP BY
- Използва се за избор на групи, които удовлетворяват дадено условие

SELECT Name, AVG(Mark) AS Average FROM Grades
GROUP BY Name HAVING AVG(Mark) >= 4

Name	Code	Mark
Иван	112	4
Иван	125	4
Милена	112	3
Мария	136	6
Мария	137	4
Стоян	125	2

Name	Average
Иван	4
Мария	5

WHERE и HAVING

- ✓ **WHERE се съотнася към** редове от таблици и поради това не използва агрегатни функции
- ✓ **HAVING се съотнася към** групи от редове от таблици и поради това не може да използва колони, които не са включени в **GROUP BY**

WHERE и HAVING

Обработка на заявката:

- Таблиците се комбинират (...)
- **WHERE клаузи**
- **GROUP BY и** агрегатни функции
- Избор на колони
- **HAVING клаузи**
- **ORDER BY**

UNION и др.

- **UNION, INTERSECT и EXCEPT (MINUS)**
  - Те третират таблиците като множества и са обичайните оператори за обединение, пресичане и разлика
  - Най-разпространено е **UNION**
  - Oracle има **MINUS** вместо **EXCEPT**
- Всички те комбинират резултати от две команди select
- Резултатите от двете команди selects трябва да имат едни и същи колони и типове данни

UNION

Да се намерят в една заявка средните оценки на всеки студент и средната оценка за всички студенти .

Name	Code	Mark
Иван	112	4
Иван	125	4
Милена	112	3
Мария	136	6
Мария	137	4
Стоян	125	2

UNION

- Заявката за средноаритметичната оценка на всеки студент:

SELECT Name, AVG(Mark) AS Average FROM Grades GROUP BY Name

- Заявката за средноаритметичната оценка на всички студенти:

SELECT 'Total' AS Name, AVG(Mark) AS Average FROM Grades

UNION

SELECT Name, AVG(Mark) AS Average FROM Grades GROUP BY Name

UNION

Name	Average
Иван	4
Милена	3
Мария	5
Стоян	2
Total	3.5

SELECT 'Total' AS Name, AVG(Mark) AS Average FROM Grades

Пример:

- Заявки за:
  - Списък на студентите и техните средни оценки
  - За студентите от първи и втори курс – техните средни оценки за годината
  - За студентите от трети курс да се пресметне 40% от средния успех на втората година изчислено с 60% на средния успех от третата година.

- Резултатите да са :
  - Сортирани по година, след това средната оценка (от по-висока към по-ниска), после първото име и накрая ID
  - Да се вземе предвид броя на кредитите на всяка дисциплина
  - Да се реализира с една заявка

Таблици за примера:

Student

ID	First	Last	Year

Grade

ID	Code	Mark	YearTaken

Module

Code	Title	Credits

Необходим е UNION

- Третокурсниците се третират различно
- Една заявка за третокурсниците
- Втора заявка за първи и втори курс
- Използва се UNION за тяхното обединение:

<QUERY FOR 3YEAR>
UNION
<QUERY FOR 1, 2 YEAR>

Трябва да се направи Join на таблиците

- Двете подзаявки имат нужда от данните във всички таблици
  - ID на студент , име(name) и година (year)
  - Оценките по всяка дисциплина (module) и годината, която приключва(yeartaken)
  - Броят на кредитите (credits) за всяка дисциплина
- Това е ***natural join*** операция
  - Би могло да се използва NATURAL JOIN команда с надеждата, че използваната версия на SQL го поддържа
  - По-сигурното е да се използва клаузата WHERE

Заявката до този момент ....

```
SELECT <some information>
FROM Student, Module, Grade
WHERE Student.ID = Grade.ID
AND Module.Code = Grade.Code
AND <student is in 3 year>
UNION
SELECT <some information>
FROM Student, Module, Grade
WHERE Student.ID = Grade.ID
AND Module.Code = Grade.Code
AND <student is in 1 or 2 year>
```


Информация за 3-ти курс

- Трябва да се извлекат:
  - Изчисляване на средноаритметична оценка, с тегло 40-60 съответно за 2-ри и 3-ти курс
  - Оценките от първата година се игнорират
  - Нужни са ID, Name и Year за подредбата
- Средноаритметична оценка е проблемна
  - Няма как директно и лесно да се разделят оценките от 2 и 3 курс
  - Би могло да се използва, че 40 = 20*2 и 60 = 20*3, така че между YearTaken (завършения курс) и теглото на оценките да има някаква зависимост

Заявката за 3-ти курс

```
SELECT Year, Student.ID, Last, First,
SUM((20*YearTaken/100)*Mark*Credits)/120
AS AverageMark
FROM Student, Module, Grade
WHERE Student.ID = Grade.ID
AND Module.Code = Grade.Code
AND YearTaken IN (2,3)
AND Year = 3
GROUP BY Year, Student.ID, First, Last
```

Информация за 1-ви и 2-ри курс

- Трябва само да се вземат средноаритметичните оценки, където YearTaken иYear са с една и съща стойност
- Нужни са ID, Name и Year за подредбата, както в другата заявка

Заявката за 1-ви и 2-ри курс

```
SELECT Year, Student.ID, Last, First,
SUM(Mark*Credits)/120 AS AverageMark
FROM Student, Module, Grade
WHERE Student.ID = Grade.ID
AND Module.Code = Grade.Code
AND YearTaken = Year
AND Year IN (1,2)
GROUP BY Year, Student.ID, First, Last
```


Финалната заявка

```
SELECT Year, Student.ID, Last, First,
SUM((20*YearTaken/100)*Mark*Credits)/120 AS AverageMark
FROM Student, Module, Grade
WHERE Student.ID = Grade.ID AND Module.Code = Grade.Code
AND YearTaken IN (2,3) AND Year = 3
GROUP BY Year, Student.ID, First, Last
UNION
SELECT Year, Student.ID, Last, First,
SUM(Mark*Credits)/120 AS AverageMark
FROM Student, Module, Grade
WHERE Student.ID = Grade.ID AND Module.Code = Grade.Code
AND YearTaken = Year AND Year IN (1,2)
GROUP BY Year, Student.ID, First, Last
ORDER BY Year desc, AverageMark desc, First, Last, ID
```