Show/Describe

SHOW TABLES	Show tables in a database.	SHOW TABLES
DESCRIBE *table_name*	Get fields (columns) and information about those fields for a given table	DESC Persons

Select

SELECT *column_name(s)* **FROM** *table_name*	Select data from a table.	SELECT LastName, FirstName FROM Persons
SELECT * FROM *table_name*	Select all data from a table.	SELECT * FROM Persons
SELECT DISTINCT *column_name(s)* **FROM** *table_name*	Select only distinct (different) data from a table.	SELECT DISTINCT LastName, FirstName FROM Persons
SELECT *column_name(s)* **FROM** *table_name* **WHERE** *column operator value* **AND** *column operator value* **OR** *column operator value* **AND (... OR ...)** **...**	Select only certain data from a table.	SELECT * FROM Persons WHERE sex='female' SELECT * FROM Persons WHERE Year>1970 SELECT * FROM Persons WHERE FirstName='Saddam' AND LastName='Hussein' SELECT * FROM Persons WHERE (FirstName='Tove' OR FirstName='Stephen') AND LastName='Svendson' SELECT * FROM Persons WHERE FirstName LIKE 'O%' SELECT * FROM Persons WHERE FirstName LIKE '%la%'
SELECT *column_name(s)* **FROM** *table_name* **WHERE** *column_name* **IN (***value1***, *value2***, ...)**	The IN operator may be used if you know the exact value you want to return for at least one of the columns.	SELECT * FROM Persons WHERE LastName IN ('Hansen','Pettersen')
SELECT *column_name(s)*	Select data from a table with sorted rows. **Note:**	SELECT * FROM Persons ORDER BY LastName

Operators

Operator	Description
=	Equal
<>	Not equal
>	Greater than
<	Less than
>=	Greater than or equal
<=	Less than or equal
BETWEEN	Between an inclusive range
LIKE	Search for a pattern. A "%" sign can be used to define wildcards (missing letters in the pattern) both before and after the pattern.

FROM *table_name*
ORDER BY *row_1*, *row_2*
DESC, *row_3* **ASC**, **...**

- **ASC** (ascend) is a alphabetical and numerical order (optional)
- **DESC** (descend) is a reverse alphabetical and numerical order

```
SELECT Company, OrderNumber
FROM Orders
ORDER BY Company DESC,
OrderNumber ASC
```

SELECT *column_1*, **...,**
SUM(*group_column_name***)**
FROM *table_name*
GROUP BY *group_column_name*

GROUP BY... was added to SQL because aggregate functions (like SUM) return the aggregate of all column values every time they are called, and without the GROUP BY function it was impossible to find the sum for each individual group of column values.

```
SELECT Company,
SUM(Amount)
FROM Sales
GROUP BY Company
```

Some aggregate functions	
Function	**Description**
AVG(column)	Returns the average value of a column
COUNT(column)	Returns the number of rows (without a NULL value) of a column
MAX(column)	Returns the highest value of a column
MIN(column)	Returns the lowest value of a column
SUM(column)	Returns the total sum of a column

SELECT *column_1*, **...,**
SUM(*group_column_name***)**
FROM *table_name*
GROUP BY *group_column_name*
HAVING
SUM(*group_column_name***)** *condition value*

HAVING... was added to SQL because the WHERE keyword could not be used against aggregate functions (like SUM), and without HAVING... it would be impossible to test for result conditions.

```
SELECT Company,
SUM(Amount)
FROM Sales
GROUP BY Company
HAVING SUM(Amount)>10000
```

Join

SELECT *column_1_name*, *column_2_name*, **...**
FROM *first_table_name*
INNER JOIN
second_table_name
ON *first_table_name.keyfield* **=**
second_table_name.keyfield

The INNER JOIN returns all rows from both tables where there is a match. If there are rows in first table that do not have matches in second table, those rows will not be listed.

```
SELECT Employees.Name,
Orders.Product
FROM Employees
INNER JOIN Orders
ON Employees.Employee_ID=
Orders.Employee_ID
```