

SELECT

Specific Columns `Select Col1,C - ol2 ,...,Col n from tablename`

Select Distinct `Select Distinct * from tablename`

you can show the results in your desired name by using 'AS' clause like :
`select Count(*) as CountryCount from (select Country from Customers)`

GROUP BY

```
SELECT column_name(s)
FROM table_name
WHERE condition
GROUP BY column_name(s)
ORDER BY column_name(s);
```

The GROUP BY statement groups rows that have the same values into summary rows, like "find the number of customers in each country".

The GROUP BY statement is often used with aggregate functions (COUNT(), MAX(), MIN(), SUM(), AVG()) to group the result-set by one or more columns.

INSERT INTO

```
INSERT INTO table_name (column1,
column2, column3, ...)
VALUES (value1, value2, value3,
...);
```

SELECT TOP

SQL SERVER `SELECT TOP number |percent column_name(s) FROM table_name WHERE condition;`

MYSQL `SELECT column_name(s) FROM table_name WHERE condition LIMIT number;`

ORACLE `SELECT column_name(s) FROM table_name ORDER BY column_name(s) FETCH FIRST number ROWS ONLY;`

Not all database systems support the SELECT TOP clause. MySQL supports the LIMIT clause to select a limited number of records, while Oracle uses FETCH FIRST n ROWS ONLY and ROWNUM.

BETWEEN

```
SELECT column_name(s)
FROM table_name
WHERE column_name BETWEEN
value1 AND value2;
// you can also use NOT with
BETWEEN
SELECT * FROM Products
WHERE ProductName NOT BETWEEN
value1 AND value2
```

LIKE

```
SELECT column1, column2, ...
FROM table_name
WHERE columnN LIKE pattern;
```

LIKE (cont)

`WHERE CustomerName LIKE 'a%'`
Finds any values that start with " a"

The percent sign (%) represents zero, one, or multiple characters

The underscore sign (_) represents one, single character

You can also combine any number of conditions using AND or OR operators.

VIEW

```
CREATE VIEW view_name
AS SELECT column1,
column2, ... FROM
table_name WHERE
condition;
```

```
UPDATING VIEW CREATE OR REPLACE
VIEW view_name AS
SELECT column1,
column2, ... FROM
table_name WHERE
condition;
```

```
DROPPING VIEW DROP VIEW view_name;
```

CHECK

```
MYSQL - CREATE TABLE Persons (
CREATE ID int NOT NULL,
TABLE - LastName varchar(255)
NOT NULL, FirstName
varchar(255), Age int,
CHECK (Age>=18) );
```



By Nima (nimakarimian)

cheatography.com/nimakarimian/
www.nimakarimian.ir

Published 18th July, 2022.

Last updated 18th July, 2022.

Page 1 of 8.

Sponsored by CrosswordCheats.com

Learn to solve cryptic crosswords!

<http://crosswordcheats.com>

CHECK (cont)

```
Others - CREATE TABLE Persons (
CREATE ID int NOT NULL,
TABLE- LastName varchar(255)
NOT NULL, FirstName
varchar(255), Age int
CHECK (Age>=18) );
```

```
Multiple CREATE TABLE Persons (
Columns ID int NOT NULL,
and LastName varchar(255)
Naming NOT NULL, FirstName
varchar(255), Age int,
City varchar(255),
CONSTRAINT CHK_Person
CHECK (Age>=18 AND
City=' San dnes' ) );
```

```
CHECK - ALTER TABLE Persons ADD
ALTER- CHECK (Age>=18);
TABLE-
```

```
Multiple ALTER TABLE Persons ADD
Columns CONSTRAINT CHK_Pe -
and rsonAge CHECK (Age>=18
naming - AND City=' San dnes');
ALTER-
TABLE-
```

CHECK (cont)

```
DROP ALTER TABLE Persons
CHECK - DROP CONSTRAINT
MYSQL- CHK_PersonAge;
DROP ALTER TABLE Persons
CHECK - DROP CHECK CHK_Pe -
Others- rsonAge;
```

UNIQUE

```
-- SQL SERVER --
CREATE TABLE Persons (
ID int NOT NULL UNIQUE,
LastName varchar(255)
NOT NULL,
FirstName varchar(255),
Age int
);
-- MYSQL --
CREATE TABLE Persons (
ID int NOT NULL,
LastName varchar(255)
NOT NULL,
FirstName varchar(255),
Age int,
UNIQUE (ID)
);
-- Define multiple Unique keys -
CREATE TABLE Persons (
ID int NOT NULL,
LastName varchar(255)
NOT NULL,
```

UNIQUE (cont)

```
FirstName varchar(255),
Age int,
CONSTRAINT UC_Person
UNIQUE (ID,LastName)
);
```

The UNIQUE constraint ensures that all values in a column are different.

Both the UNIQUE and PRIMARY KEY constraints provide a guarantee for uniqueness for a column or set of columns.

A PRIMARY KEY constraint automatically has a UNIQUE constraint.

However, you can have many UNIQUE constraints per table, but only one PRIMARY KEY constraint per table.

Stored procedures (User defined functions)

```
CREATE PROCEDURE procedure_name
AS
sql_statement
GO;
-----
EXEC procedure_name;
-----
EXAMPLE =====>
CREATE PROCEDURE Select All -
Customers @City nvarchar(30),
@PostalCode nvarchar(10)
AS
SELECT * FROM Customers WHERE
City = @City AND PostalCode =
@PostalCode
```



By Nima (nimakarimian)

Published 18th July, 2022.

Last updated 18th July, 2022.

Page 2 of 8.

Sponsored by CrosswordCheats.com

Learn to solve cryptic crosswords!

<http://crosswordcheats.com>

Stored procedures (User defined functions) (cont)

```
GO;
-----
EXEC Select All Customers @City
= 'London', @PostalCode = 'WAL
1DP';
END EXAMPLE <====
```

ISNULL

MYSQL `SELECT ProductName, UnitPrice * (Units - InStock + IFNULL (Units OnOrder, 0)) FROM Products;`

SQL SERVER `SELECT ProductName, UnitPrice * (Units - InStock + ISNULL (Units OnOrder, 0)) FROM Products;`

ORACLE `SELECT ProductName, UnitPrice * (Units - InStock + NVL(Units OnOrder, 0)) FROM Products;`

Alternative `SELECT ProductName, UnitPrice * (Units - InStock + COALESCE (Units OnOrder, 0)) FROM Products;`

COALESCE() Function Works with MYSQL SQLSERVER and ORACLE

ANY & ALL

ANY `SELECT column_name(s) FROM table_name WHERE column_name operator ANY (SELECT column_name FROM table_name WHERE condition);`

ANY & ALL (cont)

ALL `SELECT column_name(s) FROM table_name WHERE column_name operator ALL (SELECT column_name FROM table_name WHERE condition);`

SOME `SELECT * FROM Products WHERE Price > SOME (SELECT Price FROM Products WHERE Price > 20);`

returns a boolean value as a result

The operator must be a standard comparison operator (=, <, !=, >, >=, <, or <=).

WHERE keyword

```
SELECT Col1,Col2 FROM tablename
WHERE condition
```

WHERE is for when you want to declare a specific condition for the query. in the other hand, you want to filter the records
TIP1 WHERE can be used in UPDATE, SELECT, DELETE, ETC...

UNION

```
SELECT column_name(s) FROM
table1
UNION
SELECT column_name(s) FROM
table2;
The UNION operator selects only
distinct values by default.
To allow duplicate values, use
UNION ALL:
SELECT column_name(s) FROM
table1
UNION
```

UNION (cont)

```
SELECT column_name(s) FROM
table2;
```

The **UNION** operator is used to combine the result-set of two or more **SELECT** statements.

Every **SELECT** statement within **UNION** must have the same number of columns
The columns must also have similar data types

The columns in every **SELECT** statement must also be in the same order

Aliases

Column Aliases `SELECT column_name AS alias_name FROM table_name;`

Table Aliases `SELECT column_name(s) FROM table_name AS alias_name;`

WILDCARDS

%	Represents zero or more characters	bl% finds black, blue, and blob
_	Represents a single character	h_t finds hot, hat, and hit
[]	Represents any single character within the brackets	h[oa]t finds hot and hat, but not hit
^	Represents any character not in the brackets	h[^oa]t finds hit, but not hot and hat

WILDCARDS (cont)

- Represents any single character within the specified range

c[a-b]t finds cat and cbt

MIN() & MAX()

MIN() SELECT MIN(column_name) FROM table_name WHERE condition;

MAX() SELECT MAX(column_name) FROM table_name WHERE condition;

you can use MIN() and MAX() anywhere in any Query you like the examples above are just to show implementation

DELETE

DELETE EVERYTHING DELETE FROM table_name;

DELETE WITH CONDITION DELETE FROM table_name WHERE condition;

NOTNULL

```
CREATE TABLE Persons (
    ID int NOT NULL,
    LastName varchar(255)
NOT NULL,
    FirstName varchar(255)
NOT NULL,
    Age int
);
```

NOTNULL (cont)

```
ALTER TABLE Persons
MODIFY Age int NOT NULL;
```

To create a NOT NULL constraint on the "Age" column when the "Persons" table is already created, use the above Code

DATE

DATE - format YYYY-MM-DD

DATETIME - format: YYYY-MM-DD

HH:MI:SS

TIMESTAMP - format: YYYY-MM-DD

HH:MI:SS

FOR MYSQL= ==>

YEAR - format YYYY or YY

<===

FOR SQLSERVER ==>

TIMESTAMP - format: a unique number

<===

SELECT INTO (copy)

```
SELECT column1, column2,
column3, ...
INTO newtable [IN externaldb]
FROM oldtable
WHERE condition;
```

```
SELECT * INTO Customers Backup2017 IN 'Backup.mdb'
FROM Customers;
```

DEFAULT

```
DEFAULT - CREATE TABLE Orders
CREATE ( ID int NOT NULL,
TABLE- OrderNumber int
NOT NULL, OrderDate
date DEFAULT
GETDATE() );
```

```
ALERTABLE -MYSQL- ALTER TABLE Persons
ALTER City SET
DEFAULT 'Sandnes';
```

```
ALERTABLE -SQLSERVER- ALTER TABLE Persons
ADD CONSTRAINT
df_City DEFAULT
'Sandnes' FOR City;
```

```
ALERTABLE -Oracle- ALTER TABLE Persons
MODIFY City DEFAULT
'Sandnes';
```

```
DROPTABLE -MYSQL- ALTER TABLE Persons
ALTER City DROP
DEFAULT;
```

```
DROPTABLE -Others- ALTER TABLE Persons
ALTER COLUMN City
DROP DEFAULT;
```

CASE

```
CASE
    WHEN condition1 THEN
result1
    WHEN condition2 THEN
result2
    WHEN conditionN THEN
resultN
    ELSE result
END;
```



By Nima (nimakarimian)

Published 18th July, 2022.

Last updated 18th July, 2022.

Page 4 of 8.

Sponsored by CrosswordCheats.com

Learn to solve cryptic crosswords!

<http://crosswordcheats.com>

Database

CREATE CREATE DATABASE testDB;

SHOW SHOW DATABASES

DROP DROP DATABASE databa -
sename;

BACKUP BACKUP DATABASE
databa sename TO DISK =
'filepath';

Differ-entia BACKUP DATABASE
databa sename TO DISK =
BACKUP 'filepath' WITH
DIFFER ENTIAL;

ALTER table

ADD Column ALTER TABLE
table_name ADD
column_name
datatype;

DROP ALTER TABLE
Column table_name DROP
COLUMN column_name;

ALTER/-MODIFY ALTER TABLE
table_name ALTER
COLUMN COLUMN column_name
SQLSERVER datatype;

ALTER/-MODIFY ALTER TABLE
table_name MODIFY
COLUMN COLUMN column_name
MYSQL datatype;

ALTER/-MODIFY ALTER TABLE
table_name MODIFY
COLUMN column_name
ORACLE datatype;

PRIMARY KEY

```
--MYSQL--
CREATE TABLE Persons (
    ID int NOT NULL,
    Las tName varcha r(255)
NOT NULL,
    Fir stName varcha -
r(255),
    Age int,
    PRIMARY KEY (ID)
);
--SQLS ERVER ORACLE--
CREATE TABLE Persons (
    ID int NOT NULL PRIMARY
KEY,
    Las tName varcha r(255)
NOT NULL,
    Fir stName varcha -
r(255),
    Age int
);
--Defining multiple Columns as
PK--
CREATE TABLE Persons (
    ID int NOT NULL,
    Las tName varcha r(255)
NOT NULL,
    Fir stName varcha -
r(255),
    Age int,
    CON STRAINT PK_Person
PRIMARY KEY (ID,La stName)
);
In the example above there is
only ONE PRIMARY KEY (PK_Pe -
rson).
```

PRIMARY KEY (cont)

However, the VALUE of the
primary key is made up of
TWO COLUMNS (ID + LastName).

SUM() & AVG() & COUNT ()

AVG() SELECT AVG(co lum -
n_name) FROM table_name
WHERE condition;

SUM() SELECT SUM(co lum -
n_name) FROM table_name
WHERE condition;

COUNT() SELECT COUNT(col umn -
_name) FROM table_name
WHERE condition;

IN

```
SELECT column_name(s)
FROM table_name
WHERE column_name IN (value1,
value2, ...);
OR
SELECT column_na me(s)
FROM table_name
WHERE column_name IN (SELECT
STATEM ENT);
```

The IN operator allows you to specify
multiple values in a WHERE clause.

The IN operator is a shorthand for multiple
OR conditions.



By Nima (nimakarimian)

cheatography.com/nimakarimian/
www.nimakarimian.ir

Published 18th July, 2022.

Last updated 18th July, 2022.

Page 5 of 8.

Sponsored by CrosswordCheats.com

Learn to solve cryptic crosswords!

<http://crosswordcheats.com>

UPDATE

```
UPDATE table_name SET
WITH column1 = value1,
CONDITION column2 = value2, ...
WHERE condition;
```

Be careful when updating records. If you omit the WHERE clause, ALL records will be updated!

JOINS

```
INNER JOIN FROM table1 INNER JOIN
table2 ON table1.column_name =
table2.column_name;
```

```
LEFT JOIN FROM table1 LEFT JOIN
table2 ON table1.Column =
table2.Column
```

```
RIGHT JOIN FROM table1 RIGHT JOIN
table2 ON table1.column_name =
table2.column_name;
```

```
FULL JOIN FROM table1 FULL OUTER
JOIN table2 ON
table1.column_name =
table2.column_name
WHERE condition;
```

```
SELF JOIN SELECT column_name(s)
FROM table1 T1, table1 T2
WHERE condition;
```

ORDER BY

```
SELECT column1, column2, ...
FROM table_name
ORDER BY column1, column2, ...
ASC|DESC;
```

Constraints

```
CREATE TABLE table_name (
column1 datatype
constraint,
column2 datatype
constraint,
column3 datatype
constraint,
....
);
```

Constraints can be specified when the table is created with the CREATE TABLE statement, or after the table is created with the ALTER TABLE statement.

NOT NULL
UNIQUE
PRIMARY KEY
FOREIGN KEY
CHECK
DEFAULT
CREATE INDEX

PRIMARY KEY on ALTER TABLE

```
ALTER TABLE Persons
SIMPLE ADD PRIMARY KEY
(ID);
```

```
ALTER TABLE Persons
multiple or ADD CONSTRAINT
naming PK_Person PRIMARY
KEY (ID,La stN -
ame);
```

```
DROP PK - ALTER TABLE Persons
MYSQL- DROP PRIMARY KEY;
```

```
DROP PK - ALTER TABLE Persons
Others- DROP CONSTRAINT
PK_Person;
```

FOREIGN KEY

```
MYSQL - CREATE TABLE Orders (
CREATE OrderID int NOT NULL,
TABLE - OrderNumber int NOT
NULL, PersonID int,
PRIMARY KEY (OrderID),
FOREIGN KEY (PersonID)
REFERENCES Persons(P -
ersonID) );
```

```
Others - CREATE TABLE Orders (
CREATE OrderID int NOT NULL
TABLE - PRIMARY KEY, OrderN -
umber int NOT NULL,
PersonID int FOREIGN
KEY REFERENCES Person -
s(P ersonID) );
```



By Nima (nimakarimian)

cheatography.com/nimakarimian/
www.nimakarimian.ir

Published 18th July, 2022.
 Last updated 18th July, 2022.
 Page 6 of 8.

Sponsored by [CrosswordCheats.com](https://crosswordcheats.com)
 Learn to solve cryptic crosswords!
<http://crosswordcheats.com>

FOREIGN KEY (cont)

Multiple Rows or FK with Name
 CREATE TABLE Orders (
 OrderID int NOT NULL,
 OrderNumber int NOT
 NULL, PersonID int,
 PRIMARY KEY (OrderID),
 CONSTRAINT FK_Per -
 son Order FOREIGN KEY
 (PersonID) REFERENCES
 Persons(PersonID));

FOREIGN KEY-ALTER TABLE -
 ALTER TABLE Orders ADD
 FOREIGN KEY (PersonID)
 REFERENCES Persons(
 PersonID);

Multiple Rows or FK with Name -ALTER TABLE-
 ALTER TABLE Orders ADD
 CONSTRAINT FK_Per -
 son Order FOREIGN KEY
 (PersonID) REFERENCES
 Persons(PersonID);

DROP FK -MYSQL-
 ALTER TABLE Orders
 DROP FOREIGN KEY
 FK_Per son Order;

FOREIGN KEY (cont)

DROP FK -Others-
 ALTER TABLE Orders DROP
 CONSTRAINT FK_Per son -
 Order;

TABLE

CREATE
 CREATE TABLE
 table_name (column1
 datatype, column2
 datatype, column3
 datatype, ...);

DROP
 DROP TABLE table_ -
 name;

CREATE from another Table
 CREATE TABLE new_ta -
 ble_name AS SELECT
 column1, column 2, ...
 FROM existing_tab -
 le_name WHERE;

TRUNCATE
 TRUNCATE TABLE
 table_name;

The TRUNCATE TABLE statement is used to delete the data inside a table, but not the table itself.

INSET INTO

INSERT INTO table2
 SELECT * FROM table1
 WHERE condition;

The INSERT INTO SELECT statement copies data from one table and inserts it into another table.

The INSERT INTO SELECT statement requires that the data types in source and target tables match.

INDEX

CREATE
 CREATE INDEX
 index_name ON
 table_name (column1,
 column2, ...);

CREATE UNIQUE
 CREATE UNIQUE INDEX
 index_name ON
 table_name (column1,
 column2, ...);

DROP -SQLSE-RVER-
 DROP INDEX table_ -
 name.index_name;

DROP -MYSQL-
 ALTER TABLE table_name
 DROP INDEX index_ -
 name;

DROP -ORACLE-
 DROP INDEX index_ -
 name;

AUTO INCREMENT

CREATE TABLE -MYSQL-
 CREATE TABLE
 Persons (Personid
 int NOT NULL
 AUTO_I NCR EMENT,
 LastName varcha -
 r(255) NOT NULL,
 FirstName varcha -
 r(255), Age int,
 PRIMARY KEY
 (Personid));

ALTE RTABLE -MYSQL-
 ALTER TABLE Persons
 AUTO_I NCR EME -
 NT=100;



By Nima (nimakarimian)

cheatography.com/nimakarimian/
www.nimakarimian.ir

Published 18th July, 2022.
 Last updated 18th July, 2022.
 Page 7 of 8.

Sponsored by CrosswordCheats.com
 Learn to solve cryptic crosswords!
<http://crosswordcheats.com>

AUTO INCREMENT (cont)

```
CREATE TABLE Persons
( Personid int
IDENTITY(1,1)
PRIMARY KEY, LastName
varchar(255) NOT
NULL, FirstName
varchar(255), Age
int );
```

```
CREATE SEQUENCE
seq_person MINVALUE 1
-ORACLE- START WITH 1
INCREMENT BY 1 CACHE
10;
```

```
use INSERT INTO Persons
sequence in (Personid, First -
oracle Name, Last Name)
VALUES (seq_pers -
on.nextval, 'Lar -
s', 'Monsen');
```

EXISTS

```
SELECT column_name(s)
FROM table_name
WHERE EXISTS
(SELECT column_name FROM
table_name WHERE condition);
```

The EXISTS operator is used to test for the existence of any record in a subquery.

The EXISTS operator returns TRUE if the subquery returns one or more records.

HAVING

```
SELECT column_name(s)
FROM table_name
WHERE condition
GROUP BY column_name(s)
HAVING condition
ORDER BY column_name(s);
```

The HAVING clause was added to SQL because the WHERE keyword cannot be used with aggregate functions.



By Nima (nimakarimian)

cheatography.com/nimakarimian/
www.nimakarimian.ir

Published 18th July, 2022.
Last updated 18th July, 2022.
Page 8 of 8.

Sponsored by CrosswordCheats.com
Learn to solve cryptic crosswords!
<http://crosswordcheats.com>