SQL COMMAND TYPES

- **DDL** → - Create - DROP - ALTER - TRUNCATE - comment - Rename
- **DQL** → select
- **DML** → - insert - update - delete - lock - merge
- **DCL** → GRANT / REVOKE
- **TCL** → COMMIT, ROLLBack, SAVE POINT, SET Transactions

N.B the price entered must be positive
Price number (5.0) check (price > 0)

→ Foreign key
foreign key (Idc) references client (Idc) ON update cascade)

→ ila ykoon badna nghayr bl input te3 l attribute mnsta3ml update
→ Ila badna nghayr attribute mnsta3ml alter table.
→ Update Produit set Prix = Prix * 0.2.
OR = Prix - Prix * 0.2

SELECT * FROM Customers
WHERE PostalCode is NULL;

update city column of all records in the customer table:

UPDATE Customers
SET City = 'oslo';

DELETE FROM customers Where city = Norway

Number of records
SELECT COUNT(*)
FROM PRODUCTS WHERE age = 5;

AVERAGE:
AVG(PRICE)

→ Name starts with a or b or c
WHERE Name LIKE '[a, b, c]%'

FROM a to f
LIKE '[a-f]%'

Select c.customer_id, customer name, orderid
FROM Sales.customers C left outer join sales.orders O

↓ I want to see all orders even if they do not have orders!

TA Definition Language:

CREATE
ALTER
DROP

SQL DATA TYPES:

Char	a character (fixed length)
Varchar	string values containing any characters (variable length)
BLOB	Binary large Object (variable length)
Number [(n,m)]	Defined precision and scale.
Integer [(n)]	predifined precision and scale of 0.
DATE	date/hour data type
Boolean	TRUE / FAISE / UNKNOWN
Decimal (n, m)	

SQL Constraints:

column constraint level.

Not Null — Emp_Name Varchar2 (25) NOT NULL

Unique — prohibit multiple Rows having the same value in the same column. Defined at the table or column constraint level.

Check — require a value in the DB to comply with a specified condition

Syntax: column level: Attribute_Name Data type CONSTRAINT constraint_name check (condition)

TABLE level: CONSTRAINT constraint_name check (condition)

PRimary Key — NOT NULL + Unique Constraint

TABLE or column constraint level.

Foreign Key — values in one Table to match values in other one.

TABLE/column constrain Level

ATTRIBUTE_Name Data type [CONSTRAINT constraint_name] foreign Key REFERENCES table_Name (refattribute_Name) [ON DELETE CASCADE | ON DELETE SET NULL | SET DEFAULT ON UPDATE CASCADE]

A foreign key constraint requires values in one table to match values in another one.

→ Restricting:
- Deletes of primary records
- Updates of primary records
- Inserts of dependent records

* Column constraint level:

column [CONSTRAINT constraint_name]
constraint_type,

* TABLE Constraint level:

column, ...
[CONSTRAINT constraint_name] constraint_type
(column, ...),

* Ensuring DATA integrity through updates:

Restricted update: A customer ID can only be deleted if it is not found in ORDER table.

CONSTRAINT Customer_PK PRIMARY KEY (CustomerID),
ON UPDATE RESTRICT);

..ON UPDATE CASCADE); customerID changed in customer ⇒ change in ORDER

...ON UPDATE SET NULL

...ON UPDATE SET DEFAULT);

when customerID is changed,
any customer ID in the ORDER table that matches the old Customer ID is set to a predefined default value.

CREATE TABLES:

- Defines a relation schema (with attributes and integrity constraints)
- Create an empty instance of the schema.

N.B Each SQL command end with (;).

Syntax; ~~CREATE TABLE~~

```
CREATE TABLE suppliers (
    Supplier_id number(10) NOT NULL PRIMARY KEY,
    Supplier_name varchar2(50) NOT NULL,
    City_ varchar2(50),
    Zip_code varchar2(10) );
```

OR

```
CREATE TABLE Suppliers (
    Supplier_id number(10) Not NULL,
    ;
    ;
    Primary Key (Supplier_id) );
```

```
CREATE TABLE customers (
    Customer_id number(10) NOT NULL,
    Salary numeric(9) default 0,
    department_id number(10),
    ;
    CONSTRAINT customers_pk PRIMARY KEY (customer_id),
    CONSTRAINT FK_departments FOREIGN KEY (department_id)
    REFERENCES departments (department_id));
```

N.B

```
    ;
FirstName character(20) not null,
LastName character(20) not null,
unique (LastName, FirstName) );
```

≠

```
FirstName character(20) not null unique,
LastName character(20) " " ,
```

candidate keys.

```
CREATE Table Student (
    Tutor character (20) references Staff (Lecturer),

Create Table staff (
    lecturer character (20) primary Key,
    ~~foreign key~~ Appraiser character (20),
        foreign key (Appraiser) references
            Staff (lecturer)
            on delete set null
            on update cascade);
```

↑ have name of tutor because every lecturer is tutor staff but some are not tutor

Foreign Keys.

* delete Action / Update Actions.

Cascade / Restrict / no action / set default / Set Null.

* CREATE TABLE as statement:

Create table from an existing table by copying the existing table's columns.

Syntax:

```
SQL > CREATE TABLE new_table
      AS SELECT statement;
```

* ALTER TABLE:

used to:

→ Add a new column / CONSTRAINT
→ Modify an existing column
→ Define a default value for the new column
→ DROP a column / constraint.

Syntax: ALTER TABLE table_name alter_table_action;

alter table actions: ADD [COLUMN] column_name.

TABLE Actions:

* ALTER TABLE table_name
  ADD column_name column_definition;

* multipile columns;
  ALTER TABLE table_name
  Add (column_1 column_definition,
  column_n column_definition);

  Add (weight: number Not null,
  category varchar(25) Not null);

* Modify: ALTER TABLE table_name
  MODIFY column_name column_type;

  ALTER TABLE Vendor
  Modify Phone_Number varchar(40);

* DROP: ALTER TABLE table_name
  DROP COLUMN column_name

  DROP CONSTRAINT NOTNULL (Phone nb);

* Rename: ALTER TABLE table_name
  RENAME COLUMN old_name TO new_name;

* ADD table constraint: ALTER TABLE table_name
  Add CONSTRAINT constraint_name Constraint_Type
  (ATTRibute(s));

  P.K

→ DROP; delete Entire DB / DB table
including record stored in table.

DROP TABLE constructor;

→ TRUNCATE: Remove all the record from a table.

TRUNCATE TABLE customer_name;

→ Comment: /*
*/ | --

DML commands:

1) INSERT:

insert a single / multiple row (record) in a table.

→ INSERT INTO customers(
Customer_id, sale_date, sale amount)
Values ('1005', '2019-12-12', 4200)

OR INSERT INTO CUSTOMERS
VALUES ('1005', '2019-12-12', 4200)

→ ORDER is important
→ Same nb of elements
→ if the list of attributes is missing
the list of attributes is taken with the ordering taken from the DB definition
→ if the list of attributes does not contain all attribute of the relation ⇒ the default value / value Null (if possible) is inserted for the missing attribute.

2) UPDATE:

Modify an existing row (record) in a table

```
UPDATE customers
SET store_state = 'AL'
WHERE store_state = 'NY';
```

```
UPDATE Person
SET income = income * 1.1
where age < 30 ;
```

3) DELETE: remove 1 or more rows

```
DELETE FROM person
WHERE age < 35;
```

TA Query Language :

Basics :

Use sql_store ; ← DB name

فنحدد في أيا DB بدنا نشتغل عليها

Select customer_id, first name ;

Select * ← select kel l column

From customers

Where customer_id = 1 ← condition

ORDER BY first_name default الترتيب البدنا نعمل فيه

LIMIT 3; ← number of rows

ORDER of the commands :

SELECT
FROM
WHERE
ORDER BY
LIMIT

Group by HAVING

ORDER BY (Name)
ORDER BY (Number) descending

SELECT Clause :

We can use mathematical expression with the select statement.

ex:

SELECT (points + 10) + 100, AS discount_factor
calculate discount For each customer

FROM customers ;

-- Removing duplicates.

SELECT DISTINCT State
FROM Customers; → remove duplicate value and give unique state.

→ WHERE CLAUSE:
used to filter data
WHERE condition:

→ comparison operators:
=; >; <; >=; <=; !=

→ Logical operators:

SELECT *
FROM customers
WHERE birth_date > '1990-01-01' AND points > 1000.

" " OR "
↓
aw condition awu1 tenye.

WHERE NOT (birth_date > '1990-01-01');

N.B !!

IF we want for example to select customers that lives in "VA", "GA", "FL" States.

~~WHERE State = "VA" OR "GA" OR "FL"~~ false

WHERE State = "VA" OR State = "GA" OR State = "FL"

* IN Operator
column value matches any value in a list
SELECT *
FROM customers
WHERE State IN ("VA", "NY", "GA");

* Between:
SELECT *
FROM customers
WHERE points BETWEEN 100 AND 200;

IKE Operator;

```
SELECT *
FROM customers
WHERE first_name LIKE 'b%';
```

% : any nb of characters
_ : exactly one character

REGEXP :

Returns customers whose first name starts with b

```
SELECT *
FROM customers
WHERE first_name REGEXP '^b'
```

^ : beginning of a string
$: end of a string
| : logical or
[abc] : match any single character
[a-d] : any character from a to d.

- Return customers whose first name ends with EY and contains SE.

```
WHERE first_name REGEXP 'EY$|SE'
```

- first name contains B followed by r or u.

```
WHERE first_name REGEXP 'b[ru]'
```

* IS NULL OPERATOR:

Who doesn't have phone number

```
SELECT *
FROM customers
WHERE phone_number IS NULL;
```

→ ORDER by clause

Sorts rows returned by select statement based on one/more columns in ↗ or ↘ order.
By default set in ascending order.

```
SELECT *
FROM customers
ORDER BY state, first_name DESC;
```

(state → ascending state; DESC → descending name)

LIMIT Clause:

```
SELECT *
FROM customers
LIMIT 3;
```

(Return only 3 customers)

→ Skip 6 customers and return 3

```
SELECT *
FROM customers
LIMIT 6, 3;
```