Module 1 Video 1, What is SQL?

What is SQL?

SQL - Structured Query language. → is a standard computer language for relational database mangemant and data manipulation.

- Used to query, insert, update and modify data.
- Pronounce as "sequel" or S-Q-L.
- Used to communicate with databases.
- SQL is a non-procedural language:
  - → Cannot write complet aplications
  - → But we can interact and communicate with data.

- Who is SQL used?

- SQL is all about data.
- Read / retrieve data (print)
- Write data - add data to a table.
- Update data - insert new data.

Data Base Admin (DBA)	vs	Data Scientist
- Manage entire database.		- End user of a database.
- Gives permision to users.		- Use SQL to query and retrie data.
- Determinate acces to data.		
- Manage and create tables.		
- Use SQL to query data.		

! Why data scientists use SQL?

- Retrieve data.
- May create their own table or test environment.
- Combine multiple sources together.
- Write complex queries for analysis.

! SQL and Database Management System (DBMS)

- Syntax will depend on what DBMS are we using.
- Each DBMS has it's own dialect.
- SQL is an interpreter (translator)

DBMS examples:

- Microsoft SQL Server
- SQLite
- PostgreSQL
- MySQL
- IBM DB2 Oracle

MODULE 1 VIDEO 2. Data Model Part 1. Think about your

! Think before you code -> What is the problem you trying to solve?

• Understand how your data is organized and structured in the table.

! Database and Tables

• Database -> a container to store organized data.
• Table -> a structured list of data on a specific type.
  • Columns -> a single field in the table
  • Rows -> A record in a table. (registry)

• Data Modeling (modelarea datelor)
- what we use to organize information for multiple tables and how they relate to each other together.
↓ relationate intre ele

• Relational Data Modeling
• It simplifies the connections between the data.
- Allows to retrieve and update data.
- Allows you to write data.
- It allows you to easily write queries against it.

*NoSQL - stands for Not only SQL. (subject for another time)

• Relational databases (cred ca sunt diagramele)
- show the relation between different tabs.
- used to optimize querying data, making it easy and intuitive to access data.

• Data modeling building blocks.
1 - Entity -> Person, unique and distinct -> ex: Anton Dalet
2 - Attribute -> a characteristic of an entity -> attr.: male.
3 - Relationship
a) one-to-many ——<
b) many-to-many >——<
c) one-to-one ——

1' - number of tables you need to your DB.
2' - info such as property facts you need to describe each table.
3' - how tables are linked together.

c) 1:1

- each student fills one seat, and one seat is to only one student

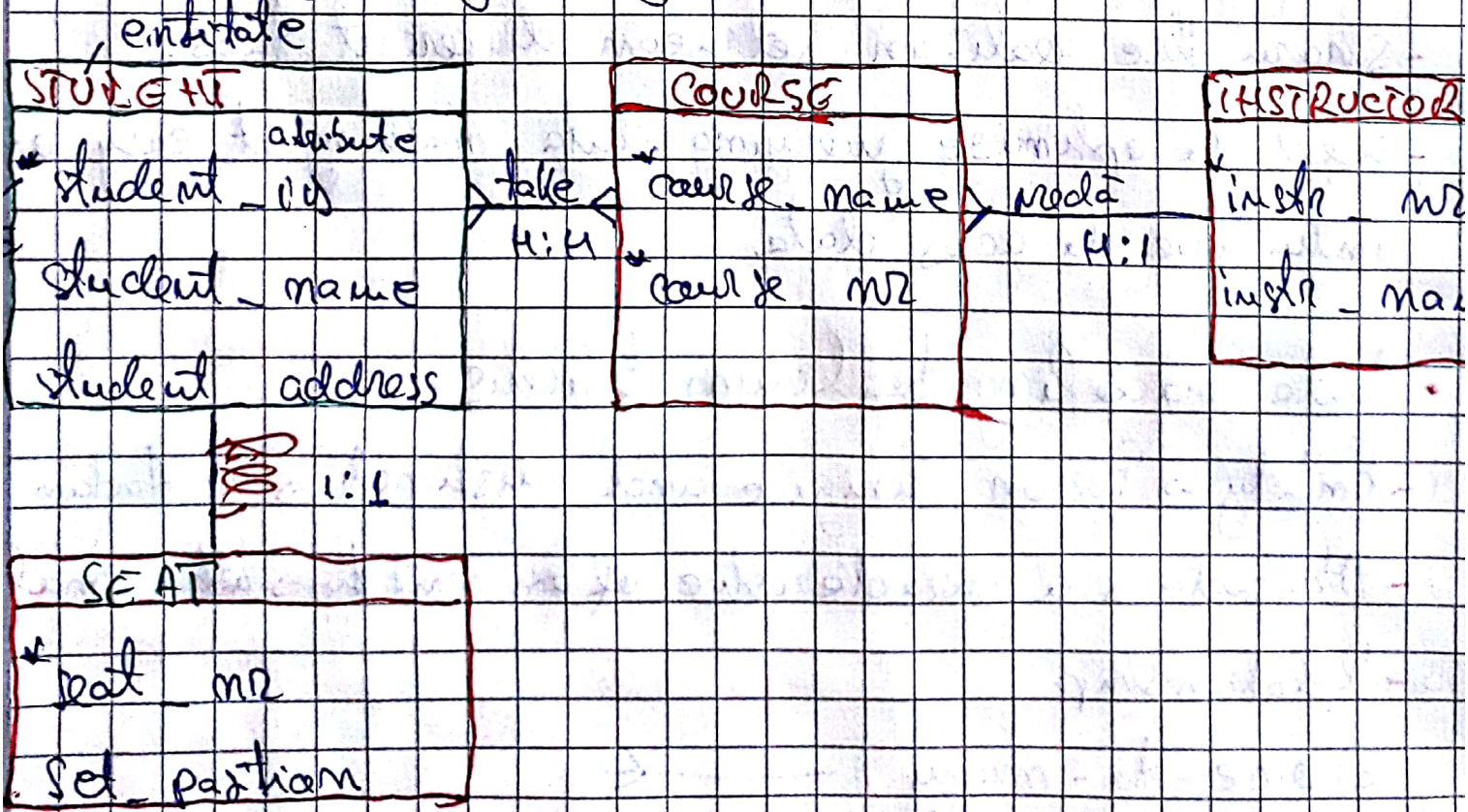
a) 1:M (phân tích relationship của class cuối)

- one instructor can teach many courses, but one can be teach by 1 instructor

- one instructor can have many students in one all student have 1 instructor for that class.

b) M:M (can be represent by many 1:M

- each consumer can buy many products, and each can be byed by many consumers.



MODUL 1. VIDEO 3. Retrieve data with SELECT STATEMENT.

- Need to specify 2 pieces of information to use SELECT STATEMENT: what you want and from where to select.

SELECT prod_name FROM Products;

PRODUCTS
prod_name.
Sampon
Pasta de dinti
Deodorant.

- Select multiple columns adding a ,.

SELECT prod_name, prod_id, prod_price FROM Products;

- Request all collumn from table using (*) wildcard.

SELECT * FROM Products;

- Limit the results (if you want a sample).

SELECT collumns FROM TABLE
LIMIT 5; -> Syntax for SQLITE

MODUL 1 VIDEO 4. Create tables.

- Why tables are usefull?
  - Use tables to make models.
  - Visualize data with other tools.
  - Extract data from other sources

- Creating tables with keyword CREATE TABLE

```
CREATE TABLE Haine
(
ID    char(10)        PRIMARY KEY,
Brand char(10)        NOT NULL,
Type  char(210)       NOT NULL,
Color char(10)        NOT NULL,
Price decimal(8,2)    NOT NULL,
Descrip varchar(750)  NULL
);
```

- Every column is NULL or NOT NULL
- Don't confuse NULL values with empty strings.
- Primary keys can't be NULL, must have a value.

- Adding data to a table. INSERT INTO

```
INSERT INTO Haine
        (ID,
         Brand,
         Type,
         Price,
        )
VALUES
        (112,
         'Nike'
         'Shoes'
         '25.55'
        );
```

MODULE 2. VIDEO 1. Basic of filtering with SQL.

- Why filter?
  - Specificăm doar datele care ne interesează.
  - reduce number of records we retrieve.
  - increase query performance

- Clause WHERE

SELECT col1 FROM table_name
WHERE col_name operator value;

Operatori:

a) - = - egal
b) - <> , != not equal
c) - > - mai mare
d) - < - mai mic
e) - >= - mare sau egal
f) - <= - mic sau egal
g) - BETWEEN - intre un range (IS AND D.O.
h) - IS NULL - is a null value. (no data in this col

SELECT * FROM table_name
a) WHERE Product_name = 'Tofu';
b) WHERE Price >= 75;
c) WHERE Product_name <> 'Hike';
g) WHERE Year BETWEEN 2015 AND 2018;
h) WHERE Description IS NULL;

Module 2. Video 3. WILDCARDS (LIKE)

- only used with strings.
- can't be used with numbers.

- % (take longer to run)
  - % Pizza -> ending word Pizza
  - Pizza % -> after word Pizza
  - % Pizza % -> before and after Pizza

S%E - start S ends with E.

t%@gmail.com

% - will not match null values.

- _ - single character.

WHERE size LIKE '_izza';

WHERE name LIKE 'Jay Parry_';

- IN & NOT IN
  - always last clause in SELECT.
  - add a ; after each call.

- ORDER BY (position) (1, 3) - 1 call, 3 call.
- ORDER BY (DIRECTION) (DESC, ASC)

- IN US Math op (+, -, /, *)
  - ORDER of OP: PEMDAS

SELECT Prod_id, Quality, Price, Discount, (Price - Discount) // Cal
AS Total_Cost FROM table.

• Cursins:

1) Create a DB & table.
2) Insert
3) Select data
4) Where clause.
5) Update data
6) Delete data
7) Change base
8) Create Primary key.
9) Create Index.
10) Create more tables.
11) Create foreign key.
12) Join relationship
13) Functions
14) Group by.

1) • Create a database: create database Customer.
- holds all tables

• Use a specific table (database): use Customer.

• Create a table.

create table Customer
(
c1 First name, varchar(25),
c2 Last name, varchar(25),
c3 Age int
)

2) • Insert table.

insert into Customers (First name, Last name, Age)
values ('Tony', 'Salute', 25);

- Index
  - create in a table to find data more quickly and efficient.
  - users don't see indexes, they are used to speed up queries
  - only if is beg used.

CREATE INDEX Cindex ON Customers (city)

CREATE INDEX index_name ON table_name (c1, c2, c3).

DROP INDEX Cindex ON Customers

- AUTO INCREMENT: - default is 1
  - Can be set -> ALTER TABLE Persons AUTO_INCREMENT=100;

SQL - Select distinct

SELECT DISTINCT (only distinct values)

SELECT FROM table -> select all (duplicates)

- SELECT DISTINCT Country FROM Customers;

- SELECT * FROM Customers.

WHERE - filter records

SELECT * FROM Customers WHERE Country = 'Mexico';

WHERE CustomerID = 1;

OPERATORS:

- = - egal

<>, != - not egal

>, < - mare, mic

>=, <=

BETWEEN 1 AND 10 - specific range.

LIKE - search for a pattern.

IN - specify values for a column.

AND, OR, NOT

SELECT * FROM Customers WHERE Country = 'Germany' AND City

WHERE City = 'Berlin' OR City = 'Munich'

WHERE NOT Country = 'Germany'

WHERE Country = 'Germany' AND (

City = 'Berlin' OR City = 'Munich')

NULL (!= 0, or empty string) (optional)

IS NULL SELECT * FROM Customer WHERE Address IS NULL (TEST EMPTY)

IS NOT NULL SELECT * FROM Customers WHERE FN OR LN IS NOT NULL

UPDATE table_name.

SET C1 : value1, val2, val3

WHERE condition

UPDATE Customer.

SET FN = 'Tony' , LN = 'Stark'

WHERE Customer ID = 1.

DELETE FROM Customers

WHERE Amt < 50 000.

MIN (), MAX (). . min value, max value in column.

SELECT MIN(Price) FROM Products.

MAX(Price) FROM Products

SELECT MIN(Price) AS Small-Price FROM Products.

SELECT MAX(Price) AS large-price FROM Products.

COUNT (), AVG (), SUM ()

SELECT COUNT (ProductID) FROM Product. (NULL NOT COUNTED)

AVG (Price) FROM Products (NULL IGNORED) (Ma)

SUM (Quantity) FROM Product ( )

LIKE - Search for a specific pattern.

- % - 0, 1, 2 or more characters

- _ - single caracter.

IN - multi values in where. (>= 2 OR)

SELECT * FROM CUSTOMERS WHERE Country IN ('Germany', 'Ran...

NOT IN

IN (SELECT Country FROM ...

BETWEEN : AND

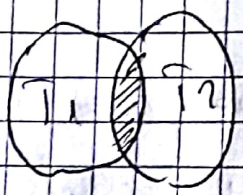
'1996-11-31' AND '2018-11-31'

'Ted' AND 'Smith'

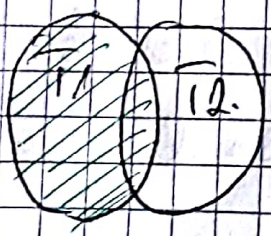
AS - give a temporary name

JOINS.

INNER JOIN or JOIN.

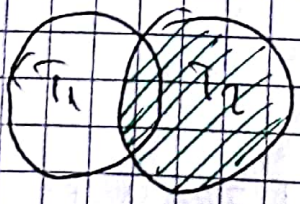


LEFT JOIN

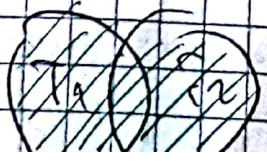


- result null if not matched

RIGHT JOIN



FULL



UNION - same nr of cols
- similar DT.

SELECT C1 FROM Table.

UNION (only distinct values) -> UNION ALL (if we want duplicate)

SELECT C11 FROM Table.

COPY DATA FROM TABLE TO OTHER.

INSERT INTO table 2 select (C1, C11, C111.

SELECT C1, C11, C111 FROM table 1

WHERE Condition

FMI - [illegible] BIC, AN 3, SEM 1
Cloud computing, curs 2/3
PROF. C. ROVDAN-AIN

2018-12-16
CURS 14/18
AMF. TITEICA (ET.3)

P1 /

H1 | 11:30

HD Insight : Hadoop + Pure.

Big Data { Entități { MapReduce, Streaming, ML (Machine learning)
MPP (massive Parallel Processing)
Tehnologii { Relational / NG structured (SQL ++) + Enterprise BI / Analytics.
Multidimensional + Tabular BE
Hadoop, MapReduce, Storage distribuită + Procesare date Multi-Structure
Source de date multiple, Process Oriented Event, SSBI (Self Service BI)

Apache Hadoop : { GDFS (cluster) - Fault Tolerance
- MapReduce

PWS - Parallel Data Warehouse
Kafka/Storm

{ (HortonWorks + Microsoft Azure) → Hadoop HA Service
Hadoop în mediu virtual
Hadoop în container

H2 | 12:30

Demo portal.azure.com → create cluster HDInsight → HBase 1.1
(Data Lake storage ?!)
(8 + 32 + 12)
299€/lună: { 9 nodes (2 Head + 4 region + 3 Zookeeper), 52 cores
HBase 1.1, Big storage 2K16 (North Eu)
Cluster dashboard → Ambari (Hadoop reporting tool)

Cloud: Azure Storage → BLOB → cluster Hadoop

On Prem Access:
Programatic (Direct Azure) Cloud Jdbc MapReduce
Expeditor Excel + Power BI + driver ODBC Data → Get Data → Azure
Direct: Storage Explorer

Hadoop → Sql Servers : SSDT (SQL Server Data Tool)
(Visual Studio SSDT) → new Integration Services Project (BI)
ODBC Data Source → Add Net Extension