Lecture (8)
"Introduction to Database"
SQL Database

Presented by : Dr. Nehal El Azaly
Dr. Dina Abdel hafez

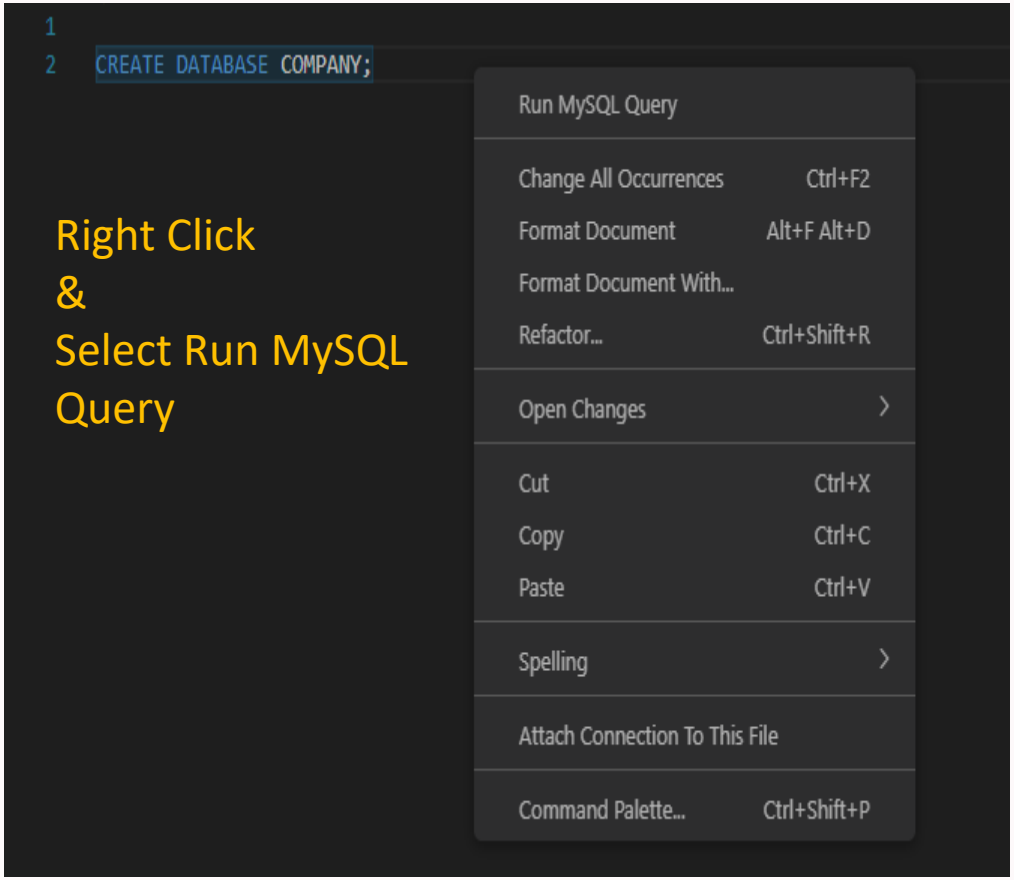
Introduction to SQL for Beginners

SQL basic commands

After connecting to SQL server if no database exist create one
Every online SQL server will create it for you
In Local SQL you must create new one

To Create Database:

```
1
2   CREATE DATABASE COMPANY;
```



Right Click
&
Select Run MySQL Query

To use Database:

You must type this command to select which Database you'll create you tables into SQL

```
1   USE COMPANY;
2
3
```

S
Q

Examples :

After creating our company database we need to create the tables

Employee Table:

```sql
USE COMPANY;


CREATE TABLE EMPLOYEE
(
    EMPLOYEE_ID INT AUTO_INCREMENT PRIMARY KEY,
    NAME VARCHAR(255) NOT NULL,
    ADDRESS VARCHAR(500),
    CITY VARCHAR(255),
    STATE VARCHAR(255)
);

```

USE COMPANY ;
This SQL statement is used to select the "COMPANY" database

EMPLOYEE_ID INT AUTO_INCREMENT PRIMARY KEY :
This column is named "EMPLOYEE_ID." It is of the integer data type (INT) and is defined as an auto-incrementing primary key.

NAME VARCHAR(255) NOT NULL:
This column is named "NAME" with data type (VARCHAR). It can store text values with a maximum length of 255 characters. The "NOT NULL" constraint specifies that this column must always have a value; it cannot be left empty (null).

S
Q

One-to-One Example using SQL

queries.sql + 3zvgp366v NEW MYSQL RUN

```sql
-- create
CREATE TABLE EMPLOYEE (
  EMPLOYEE_id INTEGER PRIMARY KEY,
  name varchar(255) NOT NULL,
  ADDRESS varchar(500) NOT NULL,
  CITY VARCHAR(255)
);
CREATE TABLE PAYROLL (
  PAYROLL_ID INT PRIMARY KEY,
  SALARY INT NOT NULL,
  POSITION VARCHAR(255) NOT NULL,
  EMPOLYEE_ID INT UNIQUE REFERENCES EMPLOYEE(EMPLOYEE_id)
);
```

1 to 1 Relation

STDIN

Input for the program (Optional)

Output:

Program did not output anything!

On update &on delete using SQL

queries.sql + 3zvgp366v NEW MYSQL RUN

```sql

-- create
CREATE TABLE EMPLOYEE (
  EMPLOYEE_id INTEGER PRIMARY KEY,
  name varchar(255) NOT NULL,
  ADDRESS varchar(500) NOT NULL,
  CITY VARCHAR(255)
);
CREATE TABLE PAYROLL (
  PAYROLL_ID INT PRIMARY KEY,
  SALARY INT NOT NULL,
  POSITION VARCHAR(255) NOT NULL,
  EMPOLYEE_ID INT UNIQUE REFERENCES EMPLOYEE(EMPLOYEE_id)
  ON UPDATE CASCADE
  ON DELETE CASCADE
);

```

STDIN

Input for the program (Optional)

Output:

Program did not output anything!

Example using SQL

queries.sql + 3zvgp366v NEW MYSQL RUN

```sql
-- create
CREATE TABLE EMPLOYEE (
  EMPLOYEE_id INTEGER PRIMARY KEY,
  name varchar(255) NOT NULL,
  ADDRESS varchar(500) NOT NULL,
  CITY VARCHAR(255)
);
CREATE TABLE PAYROLL (
  PAYROLL_ID INT PRIMARY KEY,
  SALARY INT NOT NULL,
  POSITION VARCHAR(255) NOT NULL,
  EMPOLYEE_ID INT UNIQUE REFERENCES EMPLOYEE(EMPLOYEE_id)
  ON UPDATE CASCADE
  ON DELETE CASCADE
);


insert into EMPLOYEE
(EMPLOYEE_id, name, ADDRESS)
values(1, "mohamed", "alexandria"),
(2, "ahmed", "alexandria");

select * from EMPLOYEE;
```

STDIN

Input for the program (Optional)

Output:

```
EMPLOYEE_id     name    ADDRESS CITY
1       mohamed alexandria      NULL
2       ahmed   alexandria      NULL
```

Example using SQL

queries.sql + 3zvgp366v NEW MYSQL RUN

```sql
-- create
CREATE TABLE EMPLOYEE (
  EMPLOYEE_id INTEGER PRIMARY KEY,
  name varchar(255) NOT NULL,
  ADDRESS varchar(500) NOT NULL,
  CITY VARCHAR(255)
);
CREATE TABLE PAYROLL (
  PAYROLL_ID INT PRIMARY KEY,
  SALARY INT NOT NULL,
  POSITION VARCHAR(255) NOT NULL,
  -- PersonID int FOREIGN KEY REFERENCES Persons(PersonID),
  EMPOLYEE_ID INT UNIQUE REFERENCES EMPLOYEE(EMPLOYEE_id)
  ON UPDATE CASCADE
  ON DELETE CASCADE
);


insert into EMPLOYEE
(EMPLOYEE_id, name, ADDRESS)
values(1, "mohamed", "alexandria"),
(2, "ahmed", "alexandria");

select * from EMPLOYEE;


insert into payroll
(PAYROLL_ID, SALARY, POSITION, EMPOLYEE_ID)
values (1, 1000, "developer", 1), (2, 3000, "anthing", 2);

select * from payroll;
```

STDIN

Input for the program (Optional)

Output:

```
EMPLOYEE_id     name    ADDRESS CITY
1       mohamed alexandria      NULL
2       ahmed   alexandria      NULL
PAYROLL_ID      SALARY  POSITION        EMPOLYEE_ID
1       1000    developer       1
2       3000    anthing 2
```

Example using SQL

queries.sql + 3zvgp366v NEW MYSQL RUN

```sql
-- create
CREATE TABLE EMPLOYEE (
  EMPLOYEE_id INTEGER AUTO_INCREMENT PRIMARY KEY,
  name varchar(255) NOT NULL,
  ADDRESS varchar(500) NOT NULL,
  CITY VARCHAR(255)
);
CREATE TABLE PAYROLL (
  PAYROLL_ID INT AUTO_INCREMENT PRIMARY KEY,
  SALARY INT NOT NULL,
  POSITION VARCHAR(255) NOT NULL,
  EMPOLYEE_ID INT UNIQUE REFERENCES EMPLOYEE(EMPLOYEE_id)
  ON UPDATE CASCADE
  ON DELETE CASCADE
);


insert into EMPLOYEE
(name, ADDRESS)
values("mohamed", "alexandria"),
("ahmed", "alexandria");

select * from EMPLOYEE;


insert into payroll
(SALARY, POSITION, EMPOLYEE_ID)
values (1000, "developer", 1), (3000, "anthing", 2);

select * from payroll;
```

STDIN

Input for the program (Optional)

Output:

```
EMPLOYEE_id     name    ADDRESS CITY
1       mohamed alexandria      NULL
2       ahmed   alexandria      NULL
PAYROLL_ID      SALARY  POSITION        EMPOLYEE_ID
1       1000    developer       1
2       3000    anthing 3
```

One to many Example using SQL

queries.sql + 3zvgp366v NEW MYSQL RUN

```sql
CREATE TABLE CUSTOMERS
(
 CUSTOMER_ID INT AUTO_INCREMENT PRIMARY KEY,
 CUSTOMER_NAME VARCHAR(255) NOT NULL,
 PHONE_NUMBER VARCHAR(255) NOT NULL,
 ADDRESS VARCHAR(500)
);

CREATE TABLE ORDERS
(
  ORDER_ID INT AUTO_INCREMENT PRIMARY KEY,
  ORDER_DATE DATE,
  COST INT NOT NULL,
  CUSTOMER_ID INT NOT NULL,
  foreign KEY (CUSTOMER_ID) references CUSTOMERS(CUSTOMER_ID)
);
```

STDIN

Input for the program (Optional)

Output:

Program did not output anything!

1 to many Relation

queries.sql + 3zvgrvw8p NEW MYSQL RUN

```sql
CREATE TABLE CUSTOMERS
(
 CUSTOMER_ID INT AUTO_INCREMENT PRIMARY KEY,
 CUSTOMER_NAME VARCHAR(255) NOT NULL,
 PHONE_NUMBER VARCHAR(255) NOT NULL,
 ADDRESS VARCHAR(500)
);

CREATE TABLE ORDERS
(
  ORDER_ID INT AUTO_INCREMENT PRIMARY KEY,
  ORDER_DATE DATE,
  COST INT NOT NULL,
  CUSTOMER_ID INT NOT NULL,
  foreign KEY (CUSTOMER_ID) references CUSTOMERS(CUSTOMER_ID)
);

INSERT INTO CUSTOMERS
(CUSTOMER_NAME, PHONE_NUMBER, ADDRESS)
VALUES ("MOHAMED", "+2011111111", "ALEXANDRIA"),
       ("AHMED", "+201233333333", "ALEXANDRIA");

INSERT INTO ORDERS
(ORDER_DATE, COST, CUSTOMER_ID)
VALUES ("2023-12-4", 4000, 1), ("2023-12-5", 5000, 2);


SELECT * FROM CUSTOMERS;
SELECT * FROM ORDERS;
```

STDIN

Input for the program (Optional)

Output:

```
CUSTOMER_ID     CUSTOMER_NAME   PHONE_NUMBER    ADDRESS
1       MOHAMED +2011111111     ALEXANDRIA
2       AHMED   +201233333333   ALEXANDRIA
ORDER_ID        ORDER_DATE      COST    CUSTOMER_ID
1       2023-12-04      4000    1
2       2023-12-05      5000    2
```


Thank
you