

أول حاجة: ربط قاعدة البيانات وإنشاء الاتصال

php

نسخة تحرير

<?php

، يقول للمترجم: "أنا PHP ده السطر اللي بيبدأ بيه أي كود ،
"دلوقتي PHP داخل أكتب كود".

php

نسخة تحرير

\$conn = new

mysqli('localhost','root','','blood_dontion');

هنا إحنا بننشئ اتصال بقاعدة البيانات.

- localhost: يعني السيرفر اللي عليه قاعدة البيانات هو نفس الجهاز اللي شغال عليه الموقع.
- 'root': اسم المستخدم (الافتراضي في XAMPP أو WAMP).
- '': مفيش باسورد.
- 'blood_dontion': اسم قاعدة البيانات اللي إحنا عاملينها.

php

نسخة تحرير

```
if ($conn->connect_error)
die("فشل الاتصال: ".$conn->connect_error);
```

السطر ده بيشتيك: لو الاتصال بقاعة البيانات فشل لأي سبب،
ويطبع الرسالة "فشل الاتصال" ويعرض (die) يوقف الكود
الخطأ.

بعد كده: بنجهز متغير علشان نستخدمه في رسالة التوست ✓

php

نسخة تحرير

```
$showToast = false;
```

، وحنستخدمه بعدين علشان false ده متغير عادي، بنخليه
نعرف هل نظهر رسالة "تم التسجيل" ولا لا

تسجيل المتبرع الجديد 📝

php

نسخة تحرير

```
if
($_SERVER['REQUEST_METHOD'] === 'POST' &&
isset($_POST['add_donor'])) {
```

POST (يعني) السطر ده معناه: لو الصفحة اتفتحت عن طريق مضغوط، "add_donor" فورم اتبعت، وكمان الزر بتاع ده if إبقى نفذ الكود اللي جوا الـ

php

نسخة تحرير

```
$stmt = $conn->prepare(
    "INSERT INTO Donor (full_name,
gender, date_of_birth, phone,
email, blood_type_id)
    VALUES (?, ?, ?, ?, ?, ?)"
);
```

علشان نضيف متبرع جديد في جدول SQL بنجهز أمر

علشان نحمي نفسنا () prepare، بس بنستخدم Donor

(SQL Injection) من الحقن

دي بتتملى بالقيم بعدين ؟ العلامات

php

نسخة تحرير

```
$stmt->bind_param(  
    "ssssi",  
    $_POST['full_name'],  
    $_POST['gender'],  
    $_POST['date_of_birth'],  
    $_POST['phone'],  
    $_POST['email'],  
    $_POST['blood_type_id']  
);
```

اللي فوق، ؟ هنا بنربط القيم اللي جت من الفورم بالعلامات
ببمثال نوع البيانات "ssssi" وكل حرف في

- s = string (نص)
 - i = integer (رقم صحيح)
- آخر واحدة بس اللي رقم، اللي هي فصيلة الدم.

php

نسخة تحرير

```
$stmt->execute();
```

هنا بتننفيذ عملية الإدخال الفعلية، وبيتسجل المتبرع في الجدول.

```
php
```

نسخة تحرير

```
$showToast = true;
```

بنغير حالة المتغير علشان نعرف بعددين نعرض رسالة "تم التسجيل بنجاح".

(من الداتا بيز Queries) الإحصائيات

```
php
```

نسخة تحرير

```
$totalCount = $conn-
```

```
>query("SELECT COUNT(*) AS cnt  
FROM Donor") -
```

```
>fetch_assoc()['cnt'];
```

Donor. بنسحب عدد كل المتبرعين من جدول

- بيعدّ كل الصفوف COUNT (*).
- cnt يعني بندي للنتيجة اسم وهمي اسمه AS cnt.

- يعني خذ القيمة من `fetch_assoc()['cnt']` .
cnt. العمود اللي اسمه

php

نسخة تحرير

```
$maleCount = $conn->query("SELECT COUNT(*) AS cnt  
FROM Donor WHERE  
gender='Male'")->fetch_assoc()['cnt'];
```

WHERE بنعد المتبرعين الذكور بس، الشرط
بيحدد كده gender='Male'.

php

نسخة تحرير

```
$femaleCount = $conn->query("SELECT COUNT(*) AS cnt  
FROM Donor WHERE  
gender='Female'")->fetch_assoc()['cnt'];
```

بنعد المتبرعات الإناث بس.

جلب فصائل الدم (للاستخدام في الفورم

php

نسخة تحرير

```
$bloodTypes = $conn->query("SELECT blood_type_id, type_code FROM BloodType");
```

، علشان BloodType بنجيب كل فصائل الدم من جدول نعرضها في السليكت داخل الفورم لما المستخدم يختار نوع الفصيلة.

جلب قائمة المتبرعين بالتفاصيل

php

نسخة تحرير

```
$allDonors = $conn->query("SELECT d.full_name, d.gender, DATE_FORMAT(d.date_of_birth, '%Y-%m-%d') AS dob, d.phone, d.email, bt.type_code
```

```

FROM Donor d
JOIN BloodType bt ON
d.blood_type_id =
bt.blood_type_id
ORDER BY d.full_name
");

```

Donor ، الاستعلام ده بيحيب كل بيانات المتبرعين من جدول
 عن BloodType وكماني بيحيب نوع فصيلة الدم من جدول
 JOIN. طريق.

- DATE_FORMAT (. . . , ' %Y-%m-%d ')
 بيضبط شكل تاريخ الميلاد.
- JOIN . . . ON . . . Donor يربط بين جدول
 عشان نعرض نوع الفصيلة مش BloodType و جدول
 بس رقمها.
- ORDER BY d.full_name بيرتب المتبرعين
 حسب الاسم.

المتغيرات بتاعت التنقل بين الصفحات

php

نسخة تحرير


```
$action = $_GET['action'] ??  
'home';  
$gender = $_GET['gender'] ?? '';
```

السطرين دول بنستخدمهم علشان نعرف المستخدم عايز يشوف إيه:

- action=register → يعرض صفحة التسجيل.
- action=count → يعرض الإحصائيات.
- لو مفيش أي حاجة → يروح للرئيسية.

سكول

sql

نسخة تحرير

```
CREATE DATABASE IF NOT EXISTS  
blood_donation;
```

لو قاعدة البيانات اللي اسمها " (MySQL) ده بيقول للمُخدم ﴿﴾
"مش موجودة، اعملها blood_donation"

CREATE DATABASE = أمر من أوامر إنشاء قواعد
البيانات.

IF NOT EXISTS = احتياطي علشان ما يضربش لو القاعدة
موجودة أصلاً.

sql

نسخة تحرير

```
USE blood_donation;
```

وده بيقول للمُخدم: "اشتغل جوه القاعدة دي بقى من دلوقتي، أي ٥".
جدول أو أمر هيتنفذ جواها

□ جدول أنواع الدم

sql

نسخة تحرير

```
CREATE TABLE BloodType (  
    blood_type_id INT  
    AUTO_INCREMENT PRIMARY KEY,  
    type_code VARCHAR(5) NOT NULL  
    UNIQUE  
);
```

□ **CREATE TABLE** = Section 7 أمر موجود في ملف ،
"وده معناه "اعمل جدول جديد

- **blood_type_id**: (رقم) INT ده العمود الأول، نوعه يعني (AUTO_INCREMENT صحيح)، وبيزيد تلقائي جديد. وده المفتاح الأساسي ID كل متسجل يدخل بياخد يعني ما ينفعش يتكرر (PRIMARY KEY)

- **type_code:** نوع دم المريض. نوعه "A+", "O-", زي (VARCHAR) نص ، (NOT NULL) ، (UNIQUE) ومش مسموح يتكرر.
-

جدول المتبرعين

sql

نسخة تحرير

```
CREATE TABLE Donor (  
    donor_id INT AUTO_INCREMENT  
    PRIMARY KEY,  
    full_name VARCHAR(100) NOT  
    NULL,  
    gender ENUM('Male', 'Female')  
    NOT NULL,  
    date_of_birth DATE NOT NULL,  
    phone VARCHAR(15),  
    email VARCHAR(100),  
    blood_type_id INT,  
    FOREIGN KEY (blood_type_id)  
    REFERENCES BloodType(blood_type_id)  
);
```

- **donor_id:** المفتاح الأساسي للمتبرع.
- **full_name:** الاسم كامل، ولازم يتكتب.

- **gender:** يعني يقبل قيمة من مجموعة محددة، ENUM نوعه "Female" يا "Male".
- **date_of_birth:** تاريخ الميلاد.
- **phone/email:** معلومات التواصل، ممكن يتسابوا فاضيين.
- **blood_type_id:** ده بيربط المتبرع بجدول أنواع الدم، FOREIGN KEY وعشان كده فيه

➔ **FOREIGN KEY REFERENCES** دول كانوا في
، وبستخدموا لربط جدول بجدول تاني Section 7

جدول مراكز التبرع

sql

نسخة تحرير

```
CREATE TABLE DonationCenter (
    center_id INT AUTO_INCREMENT
PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    address VARCHAR(25) NOT NULL,
    phone VARCHAR(15)
);
```

- **center_id:** مفتاح أساسي.
- **name / address:** اسم وعنوان المركز، لازم يتكتبوا.
- **phone:** رقم تليفون المركز.

✍ جدول التبرعات

sql

نسخة

```
CREATE TABLE Donation (  
    donation_id INT AUTO_INCREMENT  
PRIMARY KEY,  
    donor_id INT,  
    center_id INT,  
    donation_date DATE NOT NULL,  
    volume_ml INT,  
    FOREIGN KEY (donor_id)  
REFERENCES Donor(donor_id),  
    FOREIGN KEY (center_id)  
REFERENCES  
DonationCenter(center_id)  
);
```

- **donation_id**: معرف التبرع.
- **donor_id / center_id**: يربط التبرع بواحد من جدول المتبرعين والمركز.
- **donation_date**: التاريخ الذي حصل فيه التبرع.
- **volume_ml**: كمية الدم المتبرع بها.
- **FOREIGN KEY** تاني عشان يربط البيانات ببعض وفيه.

جدول الأدمنات (مسؤولي النظام)

sql

نسخة تحرير

```
CREATE TABLE AdminUser (  
    admin_id INT AUTO_INCREMENT  
PRIMARY KEY,  
    username VARCHAR(50) UNIQUE NOT  
NULL,  
    password VARCHAR(255) NOT NULL,  
    full_name VARCHAR(100)  
) ;
```

- هنا بنجهز جدول للمسؤولين.
- فيه اسم مستخدم وباسورد واسم كامل.
- الباسورد بيتخزن كنص عادي (لكن المفروض في التطبيق (نعمله تشفير).

إدخال البيانات (INSERT)

sql

نسخة تحرير

```
INSERT INTO BloodType (type_code)  
VALUES
```

```
('A+'), ('A-'), ('B+'), ('B-'),  
('O+'), ('O-'), ('AB+'), ('AB-');
```

👉 وده Section 8 كان موجود في **INSERT INTO** أمر ،
بنستخدمه علشان ندخل بيانات مبدئية جوه جدول أنواع الدم

sql

نسخة

```
INSERT INTO DonationCenter (name,  
address, phone) VALUES  
( ' القاهرة - ' , 'مركز الدم المركزي'  
, '010000000000', 'شارع التحرير'  
, ' الإسكندرية - ' , 'بنك الدم الجامعي'  
, '011111111111', 'سموحة' );
```

هنا بنسجل مراكز التبرع بالبيانات الكاملة بتاعتها .

👁️🗨️ (رؤية جاهزة) View إنشاء

sql

نسخة

```
CREATE VIEW vw_DonationDetails AS  
SELECT d.donation_id, dn.full_name,  
bt.type_code AS blood_type,
```



```
        c.name AS center_name,  
d.donation_date, d.volume_ml  
FROM Donation d  
JOIN Donor dn ON d.donor_id =  
dn.donor_id  
JOIN BloodType bt ON  
dn.blood_type_id = bt.blood_type_id  
JOIN DonationCenter c ON  
d.center_id = c.center_id;
```

- **CREATE VIEW:** أمر لإنشاء "رؤية" أو تجميعية بيانات جاهزة للعرض.
- **SELECT:** وده بيحيب بيانات Section 6 أمر موجود في من كذا جدول.
- **JOIN:** بيربط الجداول ببعض.
- **AS:** لتسمية الأعمدة الجديدة.

سكشن

وكل اللي بيتعلق بيه **SELECT**: أول مجموعة 

✓ **SELECT**


، بنستخدمه علشان نعرض بيانات SQL أهم أمر في
من جدول.

مثال:

sql

نسخة تحرير

```
SELECT * FROM employees;
```


المعنى: "هاتلي كل الأعمدة (كل المعلومات) من  جدول الموظفين".

✓DISTINCT

بنستخدمه لما نكون عايزين نعرض القيم المختلفة
بس، يعني يشيل التكرارات

مثال:

```
sql  
نسخة تحرير  
SELECT DISTINCT  
department_id FROM  
employees;
```

المعنى: "هاتلي كل أرقام الأقسام، بس من غير  تكرار".


✓ 12) salary * 12) التعبيرات الحسابية (زي)

ينفع نعمل عمليات حسابية جوه الاستعلام

sql

نسخة تحرير

```
SELECT salary * 12 AS  
annual_salary FROM  
employees;
```

المعنى: "هاتلي المرتب السنوي لكل موظف، اللي ".
هو المرتب الشهري $\times 12$.

✓ NULL التعامل مع

معناها "قيمة مش معروفة" مش صفر. في NULL
IS NULL أو IS NOT NULL الفلاتر بنستخدم

sql

نسخة تحرير

```
SELECT * FROM employees  
WHERE commission_pct IS  
NULL;
```

المعنى: "هاتلي الموظفين اللي ما عندهم شئ
عمولة".

✓ AS (تسمية الأعمدة)

بنستخدمه علشان نغير اسم العمود في النتيجة، من
غير ما نغير اسمه في الجدول نفسه.

sql

نسخة تحرير

```
SELECT salary * 12 AS  
annual_salary FROM  
employees;
```

المعنى: "سمّي العمود ده بـ annual_salary".

✓ || عامل الربط

يستخدمه لدمج النصوص مع بعض.

sql

نسخة تحرير

```
SELECT first_name || ' ' ||  
last_name AS full_name FROM  
employees;
```

✦ المعنى: "هاتلي اسم الموظف بالكامل: الاسم
"الأول + مسافة + الاسم الأخير".

✓ Literal Strings (السلاسل الحرفية)

'أي نص بنكتبه بين علامات تنصيص

sql

نسخة تحرير

```
SELECT 'Hello World' FROM  
DUAL;
```

✦ النتيجة هتبقى: سطر فيه الكلمة "Hello World".

✓ **q' [...] ' عامل الاقتباس البديل**

لما تكون عايز تكتب علامات اقتباس جوه نص:

```
sql  
نسخة  
SELECT q' [It's working] '  
FROM DUAL;
```

✦ جوه النص من ' علشان نحط q'[] هنا استخدمنا
غير ما يبوظ الكود.

✓ DESCRIBE

بيستخدم علشان تعرض تفاصيل جدول معين: أسماء الأعمدة، وأنواعها.

sql

نسخة تحرير

```
DESCRIBE employees;
```

✦ "المعنى: "وريني جدول الموظفين شكله إزاي

و (WHERE و WHERE Operators) ثاني مجموعة: شروط الفلتر ؟

✓ WHERE

بتفلتر بيها النتائج على حسب شرط معين.

sql

نسخة تحرير

```
SELECT * FROM employees  
WHERE salary > 5000;
```

المعنى: "هاتلي الموظفين اللي مرتبهم أكبر من 5000".

✓ = المشغل

بيستخدم للمقارنة بالتساوي.

sql

نسخة تحرير

```
WHERE department_id = 90
```

✓ = ! أو <> المشغل

"الأتين معناهم" لا يساوي.

sql

نسخة تحرير

```
WHERE job_id <> 'IT_PROG'
```

✓<, <=, >, >=

مقارنات رياضية زي الرياضيات العادية.

sql

نسخة تحرير

```
WHERE salary <= 10000
```

✓ **BETWEEN ... AND**

لما عايز تتأكد إن القيمة بين رقمين.

sql

نسخة تحرير

```
WHERE salary BETWEEN 5000  
AND 10000
```

✓IN (...)

لو بتدور على قيمة موجودة ضمن مجموعة

sql

نسخة تحرير

```
WHERE department_id IN (10,  
20, 30)
```

✓LIKE

بتستخدم لما تدور على حاجة شبه حاجة. بتستخدم
"wildcard" كـ % معاها.

sql

نسخة تحرير

```
WHERE department_id IN (10,  
20, 30)
```

المعنى: "هاتلي الناس اللي اسمهم الأخير بيبدأ
بـ S".

✓ IS NULL و IS NOT NULL

يستخدم لما تدور على القيمة الفاضية أو القيمة اللي موجودة.

sql

نسخة تحرير

```
WHERE commission_pct IS NOT  
NULL
```

✓ AND, OR, NOT

يستخدمهم للجمع أو النفي في الشروط.

sql

نسخة تحرير

```
WHERE department_id = 90  
AND salary > 5000
```

● (ORDER BY) تالت مجموعة: الترتيب

✓ ORDER BY

بتستخدم علشان ترتب النتائج

sql

نسخة تحرير

```
ORDER BY hire_date;
```

✓ ASC و DESC

ASC = (من الأصغر للأكبر)،

DESC = (من الأكبر للأصغر)

sql

نسخة تحرير

```
ORDER BY salary DESC;
```

✓ ترتيب حسب رقم العمود

sql

نسخة تحرير

```
SELECT last_name, job_id,  
hire_date FROM employees  
ORDER BY 3;
```

المعنى: "رتبلي حسب العمود الثالث اللي هو
hire_date".

رابع مجموعة: متغيرات الاستبدال

✓ &variable

يعني اسأل المستخدم يدخل قيمة وقت التشغيل

sql

نسخة تحرير

```
SELECT * FROM employees  
WHERE department_id =  
&dept_id;
```

✓ **&&variable**

بفضل محتفظ بنفس القيمة طول الجلسة.

✓ **DEFINE**

بتستخدم لتعريف متغير بشكل يدوي.

sql

نسخة تحرير

```
DEFINE deptno = 90
```

✓ **ACCEPT**

بتستخدم تطلب من المستخدم إدخال قيمة.

sql

نسخة تحرير

```
ACCEPT deptno NUMBER PROMPT  
'Enter Department Number: '
```

● (Character Functions) خامس مجموعة: دوال النصوص

✓ **UPPER () / LOWER () / INITCAP ()**

sql

نسخة تحرير

```
SELECT UPPER ('sql'),  
LOWER ('SQL'), INITCAP ('sql  
course') FROM DUAL;
```

✓ **CONCAT (str1, str2)**

بـيربط نصين.

✓ **SUBSTR(str, start, length)**

يقطع جزء من النص.

✓ **LENGTH(str)**

يحسب عدد الحروف.

✓ **INSTR(str, sub)**

يرجع مكان أول ظهور لحاجة في النص.

✓ **REPLACE(str, from, to)**


يبدل جزء من النص بحاجة ثانية.

✓ **LPAD(str, len, pad)** و **RPAD**

يزود حروف من الشمال أو اليمين لغاية ما يوصل لطول معين.

✓ **TRIM()** / **LTRIM()** / **RTRIM()**

يشيل الفراغات أو الحروف من أول أو آخر النص.

(Number Functions) سادس مجموعة: دوال الأرقام 

✓ **ROUND(number, decimals)**

يقرب الرقم.

✓ **TRUNC (number , decimals)**

يشيل الأرقام بعد العلامة.

✓ **MOD (x , y)**

الباقى من القسمة.

✓ **CEIL () / FLOOR ()**

لأقرب رقم = Floor لأقرب رقم أكبر ، Ceil = أصغر.

✓ **ABS ()**

القيمة المطلقة.

(Date **Functions)** **سابع مجموعة: دوال التاريخ** **?**

✓ **SYSDATE, CURRENT_DATE, NOW ()**

بيجيبوا التاريخ والوقت الحالي.

✓ **ADD_MONTHS () ,**
MONTHS_BETWEEN ()

تضيف أو تحسب فرق شهور.

✓ **NEXT_DAY () , LAST_DAY ()**

تجيب اليوم الجاي أو آخر يوم في الشهر.

✓ **TRUNC (date) , ROUND (date)**

تشيل أو تقرب التاريخ.

✓ **EXTRACT (field FROM date)**

(تجيب جزء من التاريخ (زي السنة أو الشهر

Erd

جدول donation

- ده الجدول اللي بيحتوي على بيانات التبرعات :الوصف .
اللي حصلت. يعني كل تبرع بيتم تسجيله هنا
- الأعمدة:
 - donation_id: ده المفتاح الرئيسي
لتحديد التبرع بشكل فريد. يعني (Primary Key)
donation_id. مفيش تبرعين بنفس الـ
 - donor_id: ده المفتاح الأجنبي (Foreign
Key) اللي بيربط التبرع بالمتبرع اللي عمل التبرع.
donor. في جدول donor_id بيشاور على
 - center_id: ده مفتاح أجنبي كمان بيشاور على
مركز التبرع في جدول
donation_center.

- donation_date: التاريخ اللي تم فيه التبرع.
 - volume_ml: حجم الدم اللي اتبرع به المتبرع (بوحدة المللي ليتر (مل).
-

bloodtype جدول

- . الجدول ده فيه أنواع الدم المختلفة اللي موجودة: الوصف .
 - . الأعمدة:
 - blood_type_id: المفتاح الرئيسي
اللي بيميز كل نوع دم بشكل فريد (Primary Key)
 - type_code: "A+" ده رمز نوع الدم (مثلاً "O-").
-

appointment جدول

- . ده الجدول اللي بيخزن المواعيد الخاصة: الوصف .
بالتبرعات.
- . الأعمدة:
 - appt_id: المفتاح الرئيسي (Primary Key)
لتحديد الموعد بشكل فريد.

- donor_id: المفتاح الأجنبي اللي بيربط الموعد
في جدول donor_id بالمتبرع من خلال donor.
 - center_id: المفتاح الأجنبي اللي بيربط
في center_id الموعد بمركز التبرع من خلال donation_center.
 - scheduled_date: التاريخ والوقت المحدد للموعد.
 - status: حالة الموعد زي "مؤكد"، "مرتقب"، أو ""تم إلغاؤه".
-

donation_center جدول

- . الجدول ده فيه معلومات عن مراكز التبرع :الوصف المختلفة.
- . الأعمدة:
 - center_id: المفتاح الرئيسي (Primary Key) للمركز.
 - name: اسم المركز.
 - address: عنوان المركز.
 - phone: رقم الهاتف الخاص بالمركز.

جدول donor

- . الجدول ده فيه بيانات المتبرعين الشخصية: الوصف .
- . الأعمدة:
 - donor_id: المفتاح الرئيسي (Primary Key) للمتبرع.
 - full_name: الاسم الكامل للمتبرع.
 - gender: الجنس (ممكن يكون Male أو Female).
 - date_of_birth: تاريخ الميلاد.
 - phone: رقم الهاتف.
 - email: البريد الإلكتروني.
 - blood_type_id: المفتاح الأجنبي اللي bloodtype بيربط نوع الدم في جدول.

جدول adminuser

- . الجدول ده فيه بيانات المستخدمين الإداريين اللي الوصف .
بيقوموا بإدارة النظام
- . الأعمدة:

- admin_id: المفتاح الرئيسي (Primary Key) للمستخدم الإداري.
 - username: اسم المستخدم.
 - password: كلمة المرور.
 - full_name: الاسم الكامل للمستخدم الإداري.
 - role_id: المفتاح الأجنبي الذي يربط المستخدم role الإداري بدوره في جدول.
-

role جدول

- . الجدول ده فيه الأدوار المختلفة للمستخدمين: الوصف الإداريين.
 - . الأعمدة:
 - role_id: المفتاح الرئيسي (Primary Key) للدور.
 - role_name: اسم الدور (مثل "مدير" أو "موظف").
-

2. العلاقات بين الجداول (Relationships)

donation جدول

- **donor:** علاقة مع جدول.
".العلاقة بين الجدولين هي علاقة "واحد إلى كثير".
يعني المتبرع ممكن يعمل أكثر من تبرع، لكن كل تبرع
بيرتبط بمتبرع واحد فقط.
 - **donation_center:** علاقة مع جدول.
برضه "العلاقة هي علاقة "واحد إلى كثير".
كل تبرع مرتبط بمركز تبرع واحد، لكن ممكن المركز
يستقبل أكثر من تبرع.
-

appointment جدول

- **donor:** علاقة مع جدول.
برضه "العلاقة هنا هي علاقة "واحد إلى كثير".
يعني كل متبرع ممكن يكون عنده أكثر من موعد، لكن
كل موعد مرتبط بمتبرع واحد.
- **donation_center:** علاقة مع جدول.
".العلاقة هي علاقة "واحد إلى كثير".
كل موعد مرتبط بمركز واحد، لكن المركز ممكن يكون
فيه مواعيد متعددة.

جدول donor

- **bloodtype** علاقة مع جدول:
"العلاقة هنا هي علاقة "واحد إلى واحد".
كل متبرع لديه نوع دم واحد، وكل نوع دم لديه متبرع واحد فقط.

جدول adminuser

- **role** علاقة مع جدول:
"العلاقة هي علاقة "واحد إلى واحد".
كل مستخدم إداري مرتبط بدور واحد، وكل دور لديه مستخدم إداري واحد فقط.

3. مفاتيح رئيسية وثانوية (Primary and Foreign Keys)

- **(Primary Keys) المفاتيح الرئيسية:**
 - `donation_id` في جدول `donation`

- blood_type_id في جدول bloodtype
- appt_id في جدول appointment
- center_id في جدول donation_center
- donor_id في جدول donor
- admin_id في جدول adminuser
- role_id في جدول role
- **(Foreign Keys) المفاتيح الثانوية:**
 - يشير إلى donor_id في جدول donation
 - donor_id في جدول donor
 - يشير center_id في جدول donation إلى center_id في جدول donation_center
 - donor_id في جدول appointment
 - donor_id في جدول donor يشير إلى donor_id في جدول appointment
 - center_id في جدول appointment يشير إلى center_id في جدول donation_center
 - يشير blood_type_id في جدول donor إلى blood_type_id في جدول bloodtype

- يشير إلى adminuser في جدول role_id
 - role في جدول role_id
-

4. مؤشرات (Indexes)

- donation:** مؤشرات في جدول
 - يمكن عمل مؤشرات على الأعمدة التي بنبحث فيها
 - donation_date ، كثير زي
 - center_id ، و donor_id.
- bloodtype:** مؤشرات في جدول
 - علشان لو في type_code نعمل مؤشر على
 - استعلامات كثير بتدور على أنواع الدم.
- appointment:** مؤشرات في جدول
 - نعمل مؤشرات على الأعمدة المهمة زي
 - donor_id ، و scheduled_date ،
 - center_id.
- donation_center:** مؤشرات في جدول
 - علشان لو دايماً بنبحث عن name نعمل مؤشر على
 - مراكز التبرع.
- donor:** مؤشرات في جدول

- `full_name` ونعمل مؤشرات على
علشان البحث عن `blood_type_id`
المتبرعين بسرعة
 - **adminuser:** مؤشرات في جدول
 - `username` نعمل مؤشر على
عمليات تسجيل الدخول
-

5. تفاصيل أخرى

donation: جدول

- يخزن بيانات التبرعات التي تمت
- ومراكز التبرع (`donor`) يربط بين المتبرعين
(`donation_center`).

appointment: جدول

- يخزن مواعيد التبرعات
- ومراكز التبرع (`donor`) يربط بين المتبرعين
(`donation_center`).

donor:جدول

- . يخزن بيانات المتبرعين الشخصية
- . يربط المتبرع بنوع الدم

adminuser:جدول

- . يخزن بيانات المستخدمين الإداريين
- . (role) يربط كل مستخدم بدوره

6. ملخص للنظام

- . يسجلون بياناتهم الشخصية ونوع (donor) المتبرعون .
دمهم، ويقومون بحجز مواعيد لتبرع الدم
- . تحتوي على (donation_center) مراكز التبرع
بيانات مراكز التبرع التي يتم فيها التبرع
- . تحتوي على أنواع الدم (bloodtype) أنواع الدم
المختلفة.
- . تحتوي على بيانات (appointment) المواعيد
المواعيد الخاصة بالتبرع
- . يديرون النظام (adminuser) المستخدمون الإداريون
ويقومون بإضافة أو تعديل بيانات المتبرعين والمراكز

أمثلة على الاستعلامات 7.

استرجاع بيانات تبرع معين

sql

نسخة

```
SELECT d.donation_id,  
d.donor_id, d.center_id,  
d.donation_date, d.volume_ml,  
dn.full_name AS  
donor_name, dc.name AS  
center_name  
FROM donation d  
JOIN donor dn ON d.donor_id =  
dn.donor_id  
JOIN donation_center dc ON  
d.center_id = dc.center_id;
```

استرجاع بيانات موعد تبرع معين

sql

نسخة

```

SELECT a.appt_id, a.donor_id,
a.center_id, a.scheduled_date,
a.status,
        dn.full_name AS
donor_name, dc.name AS
center_name
FROM appointment a
JOIN donor dn ON a.donor_id =
dn.donor_id
JOIN donation_center dc ON
a.center_id = dc.center_id;

```

استرجاع بيانات جميع المتبرعين مع أنواع دمائهم

sql

نسخة تحرير

```

SELECT d.donor_id, d.full_name,
d.gender, d.date_of_birth,
d.phone, d.email, bt.type_code
AS blood_type
FROM donor d
JOIN bloodtype bt ON
d.blood_type_id = bt.blood_type_id;

```

