

CSC116 Database

A database is an electronically stored, systematic collection of data that can include words, numbers, images, videos, and other types of files.

Column Name	Data Type	Description
patient_id	INT (PRIMARY KEY)	Unique ID for the patient
first_name	VARCHAR(50)	Patient's first name
last_name	VARCHAR(50)	Patient's last name
date_of_birth	DATE	Patient's date of birth
gender	VARCHAR(10)	Gender of the patient
phone_number	VARCHAR(15)	Contact phone number
email	VARCHAR(100)	Email address
address	VARCHAR(255)	Home address
emergency_contact	VARCHAR(100)	Emergency contact person

Patients



	patient_id	first_name	last_name	date_of_birth	gender	phone_number	email	address	emergency_contact	
1	1	Robert	Roberts	1951-09-26	Female	001-680-793-5427	larrysherman@peck-williamson.com	70240 Sherman Harbors Apt. 751 Hoffmanbury, CT 39049	Sarah Obrien	
2	2	Karen	Moore	1964-03-25	Male	(494)464-4132x6543	osmith@yahoo.com	0143 Davis Stravenue Port Denise, WA 55125	Jordan Yang	
3	3	William	Cohen	1958-07-22	Male	+1-921-661-9495x6917	brownnatalie@yahoo.com	317 Tanner Corner Lake Selena, WY 88400	Candice Allen	
4	4	Jeremy	Krause	1976-08-20	Male	218.633.3609x523	agross@gmail.com	4521 Gabriel Circle Mejiahaven, MD 60463	Stephanie Simmons	
5	5	Brandon	French	1970-11-25	Female	113-902-6043x95756	alistephen@yahoo.com	620 Robinson Valleys Suite 767 Davistown, IN 28265	Jessica Gomez	

Column Name	Data Type	Description
doctor_id	INT (PRIMARY KEY)	Unique ID for the doctor
first_name	VARCHAR(50)	Doctor's first name
last_name	VARCHAR(50)	Doctor's last name
specialization	VARCHAR(100)	Medical specialization
phone_number	VARCHAR(15)	Contact phone number
email	VARCHAR(100)	Email address
office_number	VARCHAR(20)	Office room number

Doctors

	doctor_id	first_name	last_name	specialization	phone_number	email	office_number	
1	1	Rachael	Hardy	Orthopedics	039.803.8491	elee@henry-barnes.com	Room 866	
2	2	Cynthia	Paul	Cardiology	636-426-1372x5528	jamesalvarez@gmail.com	Room 725	
3	3	Fernando	Lopez	Pediatrics	+1-277-213-2120x099	pennyolson@gmail.com	Room 124	

Column Name	Data Type	Description
appointment_id	INT (PRIMARY KEY)	Unique ID for the appointment
patient_id	INT (FOREIGN KEY)	References Patients(patient_id)
doctor_id	INT (FOREIGN KEY)	References Doctors(doctor_id)
appointment_date	DATETIME	Date and time of the appointment
reason	VARCHAR(255)	Reason for the visit
status	VARCHAR(20)	Status of the appointment (Scheduled, Completed, Canceled)

Appointments

	appointment_id	patient_id	doctor_id	appointment_date	reason	status	
1	1	3	2	2025-01-24 18:33:19	Consultation	Completed	
2	2	5	1	2025-01-26 22:59:58	Check-up	Completed	
3	3	5	1	2025-01-27 02:54:06	Emergency	Canceled	
4	4	4	2	2025-01-07 22:21:14	Follow-up	Completed	
5	5	3	3	2025-01-25 21:46:34	Emergency	Scheduled	

Column Name	Data Type	Description
hospitalization_id	INT (PRIMARY KEY)	Unique ID for the hospitalization
patient_id	INT (FOREIGN KEY)	References Patients(patient_id)
doctor_id	INT (FOREIGN KEY)	References Doctors(doctor_id)
admission_date	DATE	Date of admission
discharge_date	DATE	Date of discharge (nullable if ongoing)
room_number	VARCHAR(10)	Assigned room number
diagnosis	VARCHAR(255)	Initial diagnosis
treatment	TEXT	Treatment details

Hospitalizations

[illegible]

Column Name	Data Type	Description
medication_id	INT (PRIMARY KEY)	Unique ID for the medication
patient_id	INT (FOREIGN KEY)	References Patients(patient_id)
doctor_id	INT (FOREIGN KEY)	References Doctors(doctor_id)
medication_name	VARCHAR(100)	Name of the prescribed medication
dosage	VARCHAR(50)	Dosage instructions
start_date	DATE	Start date of the medication
end_date	DATE	End date of the medication

Medications

	medication_id	patient_id	doctor_id	medication_name	dosage	start_date	end_date	
1	1	3	2	Aspirin	1 tablet	2025-02-01	2025-02-03	
2	2	1	3	Paracetamol	250mg	2025-01-02	2025-01-05	
3	3	3	2	Ibuprofen	250mg	2025-01-02	2025-01-14	
4	4	4	3	Aspirin	250mg	2025-01-18	2025-01-19	

Column Name	Data Type	Description
bill_id	INT (PRIMARY KEY)	Unique ID for the bill
patient_id	INT (FOREIGN KEY)	References Patients(patient_id)
total_amount	DECIMAL(10, 2)	Total amount due
billing_date	DATE	Date of billing
payment_status	VARCHAR(20)	Status of payment (Paid, Pending, Overdue)
insurance_provider	VARCHAR(100)	Insurance company name (if applicable)

Billing

	bill_id	patient_id	total_amount	billing_date	payment_status	insurance_provider	
1	1	5	349.26	2025-01-01	Pending	HealthCare Inc.	
2	2	4	411.07	2025-01-16	Paid	LifeSecure	
3	3	4	556.8	2025-01-28	Pending	LifeSecure	
4	4	5	118.32	2025-01-30	Pending	HealthCare Inc.	

-- Create Patients table

```
CREATE TABLE Patients (  
    patient_id INT AUTO_INCREMENT PRIMARY KEY,  
    first_name VARCHAR(50) NOT NULL,  
    last_name VARCHAR(50) NOT NULL,  
    date_of_birth DATE NOT NULL,  
    gender VARCHAR(10),  
    phone_number VARCHAR(15),  
    email VARCHAR(100),  
    address VARCHAR(255),  
    emergency_contact VARCHAR(100)  
);
```

```
CREATE TABLE Doctors (  
    doctor_id INT AUTO_INCREMENT  
PRIMARY KEY,  
    first_name VARCHAR(50) NOT NULL,  
    last_name VARCHAR(50) NOT NULL,  
    specialization VARCHAR(100),  
    phone_number VARCHAR(15),  
    email VARCHAR(100),  
    office_number VARCHAR(20)  
);
```

SQL

SQL is a standard language for storing, manipulating and retrieving data in databases.

Query
Delete
Insert
Update

Query

https://www.w3schools.com/sql/trysql.asp?filename=trysql_select_all

```
select * from patients;
```

```
select * from doctors;
```

```
select patient_id, gender, email from  
patients;
```

Two spaces

1 space

No space

select * **from patients;**

Will print every
columns if you put a *

End of query

Question 1: Query appointment table shows the reason, appointment_id, patient_id

DISTINCT

Question 2:

```
SELECT DISTINCT medication_name, dosage  
FROM medications;
```

```
SELECT DISTINCT medication_name  
FROM medications;
```

Where

```
SELECT * FROM Customers  
WHERE Country='Mexico';
```

```
SELECT * FROM Customers  
WHERE Country='Mexico' or Country='USA';
```

Question 3:

I want to know all the female patients.

AND, OR, NOT

```
SELECT * FROM Customers  
WHERE NOT Country = 'Spain';
```

```
SELECT * FROM Customers  
WHERE Country='Mexico' or Country='USA';
```

Question 4:

**I want to know a female patient and her
emergence contact named Jordan Yang.**

IN

SELECT * FROM Appointments

WHERE status **IN ('canceled', 'completed');**

WHERE status='canceled' or statu='completed';

Insert

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

```
INSERT INTO table_name
```

```
VALUES (value1, value2, value3, ...);
```

```
INSERT INTO Customers (CustomerName, City, Country)
VALUES ('Cardinal', 'Stavanger', 'Norway');
```

CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
89	White Clover Markets	Karl Jablonski	305 - 14th Ave. S. Suite 3B	Seattle	98128	USA
90	Wilman Kala	Matti Karttunen	Keskuskatu 45	Helsinki	21240	Finland
91	Wolski	Zbyszek	ul. Filtrowa 68	Walla	01-012	Poland
92	Cardinal	null	null	Stavanger	null	Norway

Delete All Records

It is possible to delete all rows in a table without deleting the table. This means that the table structure, attributes, and indexes will be intact:

```
DELETE FROM table_name;
```

```
DELETE FROM Customers WHERE CustomerName='Alfreds Futterkiste';
```

Question 5:

**I want to delete a patient whose email equals to
'agross@gmail.com'**

Update

```
UPDATE Customers
```

```
SET ContactName = 'Alfred Schmidt', City=  
'Frankfurt'
```

```
WHERE CustomerID = 1;
```

Question 6:

**I want to update a patient whose email equals to
'[agross@gmail.com](#)', his first name actually is Kevin**

SQL JOIN

A **JOIN** clause is used to combine rows from two or more tables, based on a related column between them.

OrderID	CustomerID	OrderDate
10308	2	1996-09-18
10309	37	1996-09-19
10310	77	1996-09-20

CustomerID	CustomerName	ContactName	Country
1	Alfreds Futterkiste	Maria Anders	Germany
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Mexico
3	Antonio Moreno Taquería	Antonio Moreno	Mexico

```
SELECT Orders.OrderID, Customers.CustomerName,  
Orders.OrderDate  
  
FROM Orders INNER JOIN Customers ON  
Orders.CustomerID=Customers.CustomerID;
```

OrderID	CustomerName	OrderDate
10308	Ana Trujillo Emparedados y helados	9/18/1996

Database Types

Relational Databases (RDBMS):

- Use structured tables with rows and columns.
- Support SQL (Structured Query Language) for data management.
- **Examples:** MySQL, Oracle Database, Microsoft SQL Server.

NoSQL Databases:

- Designed for unstructured or semi-structured data.
- Include different models like document, key-value, column-family, and graph.
- **Examples:**
 - **Document:** MongoDB, CouchDB.
 - **Key-Value:** Redis, DynamoDB.
 - **Column-Family:** Cassandra, HBase.
 - **Graph:** Neo4j, ArangoDB.

Patients

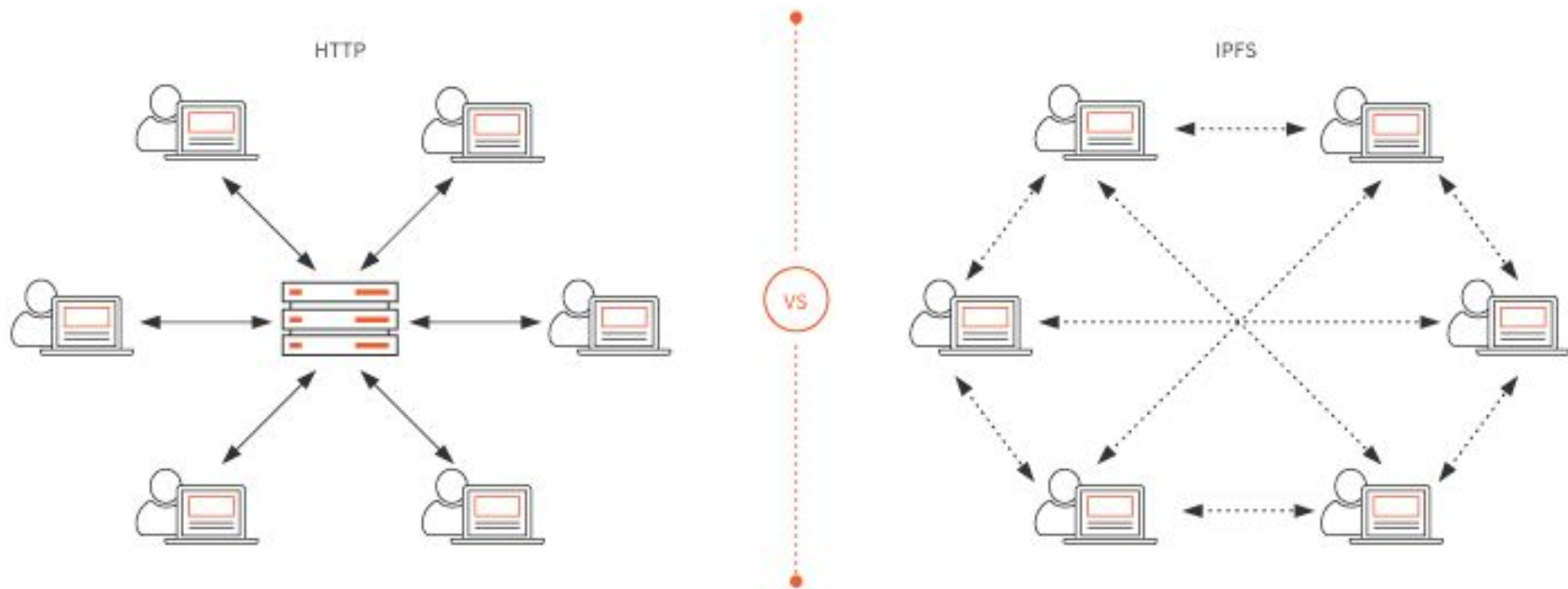


	patient_id	first_name	last_name	date_of_birth	gender	phone_number	email	address	emergency_contact	
1	1	Robert	Roberts	1951-09-26	Female	001-680-793-5427	larrysherman@peck-williamson.com	70240 Sherman Harbors Apt. 751 Hoffmanbury, CT 39049	Sarah Obrien	
2	2	Karen	Moore	1964-03-25	Male	(494)464-4132x6543	osmith@yahoo.com	0143 Davis Stravenue Port Denise, WA 55125	Jordan Yang	
3	3	William	Cohen	1958-07-22	Male	+1-921-661-9495x6917	brownnatalie@yahoo.com	317 Tanner Corner Lake Selena, WY 88400	Candice Allen	
4	4	Jeremy	Krause	1976-08-20	Male	218.633.3609x523	agross@gmail.com	4521 Gabriel Circle Mejahaven, MD 60463	Stephanie Simmons	
5	5	Brandon	French	1970-11-25	Female	113-902-6043x95756	alistephen@yahoo.com	620 Robinson Valleys Suite 767 Davistown, IN 28265	Jessica Gomez	

Name: Jack
Age: 25
Condition: Cold

Name: Li Si
Age: 30
Condition: High Blood Pressure
Allergies: Penicillin

Decentralized Database IPFS



Security problems in Databases

- **Authentication & Authorization:** Passwords, biometrics, role-based access (e.g., nurses accessing only their patients' records).
- **Encryption:** Scrambling data (e.g., SSL for online forms).
- **Backups & Recovery:** Regular backups to prevent data loss.
- **Audits:** Tracking access logs to detect breaches.