ImmunoSys – An Influenza Vaccine Tracking System

Team number: Group 9

Team member names: Mike Wurth, Priyanka Singh, Tim Lillig

1A. Database Description:

The ImmunoSys - Influenza Vaccine Tracking System is a comprehensive solution designed to streamline the management of immunization records for a diverse range of users, including patients, healthcare providers/administrators, and public health officials. Patients using the system receive annual flu vaccinations to safeguard themselves from seasonal influenza. They depend on ImmunoSys to access their immunization records, manage vaccination schedules, receive reminders for upcoming vaccinations, and easily share immunization records with healthcare providers or educational institutions as necessary. Patients span various age groups, from infants receiving childhood vaccinations to adults obtaining annual flu shots or other recommended immunizations.

Healthcare providers/administrators are integral to the seamless operation and appointment management within healthcare facilities that administer vaccinations. This group encompasses administrative personnel, such as office managers and receptionists, as well as medical staff, including doctors, nurses, pharmacists, and others directly involved in vaccine administration. Together, they are responsible for scheduling appointments, managing patient records, and ensuring the efficient delivery of vaccination services. ImmunoSys is essential for streamlining the management of appointment calendars and ensuring the optimal allocation of vaccines and medical supplies. This comprehensive role enables healthcare providers/administrators to maintain operational efficiency and support the delivery of essential immunization services.

Public health officials are responsible professionals dedicated to promoting and safeguarding community health through disease prevention and health promotion initiatives. This group comprises biostatisticians, health educators, public health administrators, and policymakers who oversee vaccination programs, monitor disease outbreaks, and develop strategies to enhance vaccination coverage rates. They depend on ImmunoSys for collecting and analyzing immunization data, identifying trends in vaccination coverage and disease incidence, and making informed decisions regarding resource allocation and public health interventions.

At the core of this sophisticated system are patient profiles, healthcare provider details, vaccine information, and appointment scheduling, facilitating the documentation of vaccinations by linking each vaccine to its recipient, administering provider, and date given. By enabling future appointment bookings, the system supports healthcare providers in maintaining up-to-date records and assists patients in managing vaccination schedules. This collaborative project aims to address the needs of individuals and organizations involved in vaccination administration, thereby enhancing healthcare delivery, and promoting efficient public health monitoring and decision-making processes.

1B. Mission Statement:
Our mission is to ensure easy access to accurate vaccination histories, facilitate efficient scheduling and administration of immunizations, and support comprehensive public health initiatives through the meticulous tracking of vaccine administration and outcomes.

2. Mission Objectives:

To maintain (enter, update, delete) data on vaccination appointment scheduling

To maintain (enter, update, delete) data on patient immunization records

To maintain (enter, update, delete) data on health care provider schedules

To maintain (enter, update, delete) data on vaccination types

To maintain (enter, update, delete) data on vaccination administration

To perform searches on patient immunization records

To perform searches on healthcare centers

To track the status of patient vaccination appointments

To report on patient immunization records

To report on vaccination demographics

To report on patient side effects

3. User types:

User Type: Patient:

View personal vaccination history

Request vaccination appointment

View recommended vaccine schedule

View available appointment windows through potential providers

User Type: Healthcare Provider/Administrator:

Confirm/schedule vaccination appointments

Order vaccine doses

View patient vaccination history/records

Record vaccinations and update patient records

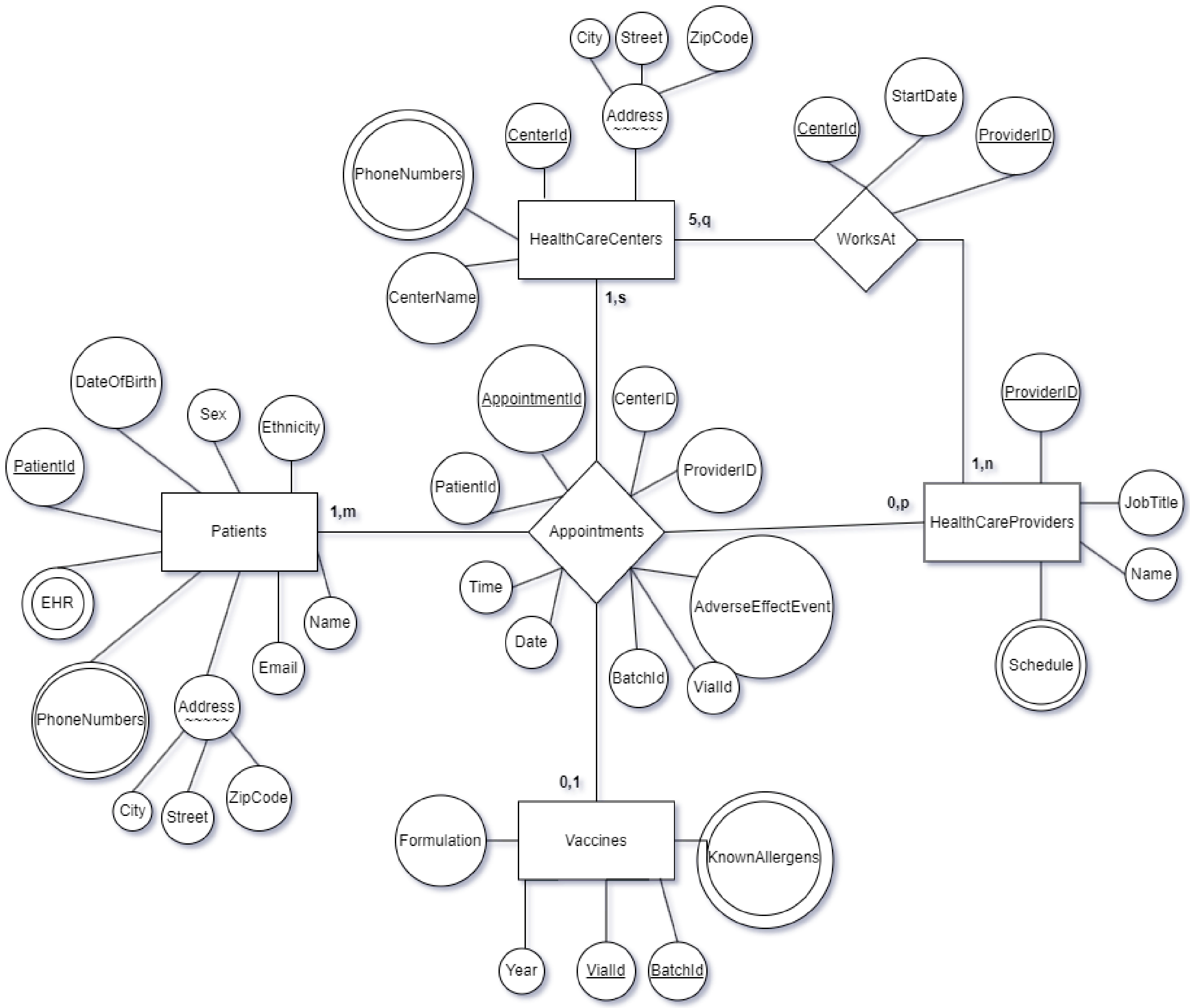
Assign individual healthcare provider to a given appointment

<u>User Type: Public Health Official:</u>

Group vaccination data/rates by region, demographic (age, location, sex, ethnicity, etc.)

Download vaccination data tables

4. Entity-Relationship Model:



Entity/Relationship	**Attribute**	**Definition**
Patients	PatientID	A unique identifier integer for each patient
	DateOfBirth	Date of birth
	Sex	Demographic information
	Ethnicity	Demographic information
	Name	A string containing the patient's first and last name
	Email	Email address
	Address	A combination of city, street, and zip code
	PhoneNumbers	Multivalued, contains patient phone #'s
	E.H.R.	Multivalued, contains past patient health records
Vaccines	VialID	A unique identifier integer for each vial within a batch (may be repeated in different batches)
	BatchID	A unique identifier integer for each batch of vaccine vials
	Year	An integer year for the strain of Influenza that the vaccine was developed for. E.g. 2024
	Formulation	A string containing the type of influenza vaccine. E.g. inactivated influenza vaccines [IIV4], recombinant influenza vaccine [RIV4], and live attenuated influenza vaccine [LAIV4]
	KnownAllergens	A list of strings of the ingredients in the vaccine that may cause allergic reactions. (multivalued)
HealthCareCenters	CenterId	A unique identifier integer for each healthcare center
	CenterName	Name of healthcare center
	PhoneNumbers	Multivalued, contains healthcare center phone #'s
	Address	A combination of city, street, and zip code
HealthCareProviders	ProviderId	A unique identifier integer for each provider

	JobTitle	Title for job (i.e. Nurse, Doctor, etc.)
	Name	A string containing the provider's first and last name.
	Schedule	A multivalued attribute containing schedule information
Appointments <Relationship>	VialId	A unique identifier integer for each vial within a batch (may be repeated in different batches)
	AppointmentId	A unique identifier integer for each appointment created
	BatchId	A unique identifier integer for each batch of vaccine vials
	PatientId	A unique identifier integer for each patient
	ProviderId	A unique identifier integer for each provider
	AdverseEffectEvent	Adverse effect related to vaccination
	Date	Date of appointment
	CenterId	A unique identifier integer for each healthcare center
	Time	Time of day in H:M:S
WorksAt <Relationship>	CenterId	A unique identifier integer for each healthcare center
	StartDate	Date that healthcare provider started working at specified healthcare center (is assumed to be a valid date to capture when a provider starts working at a center)
	ProviderId	A unique identifier integer for each provider

5. The Relational Model: Normalized at least to 3rd normal form

***** Note bold attributes are candidate keys

Patients (PatientId, DateOfBirth, Sex, Ethnicity, Name, **Email**, City, Street, ZipCode)

PatientPhoneNumbers (PatientId, PhoneNumber)

EHR (PatientId, EHR)

HealthCareCenters (CenterId, CenterName, City, Street, ZipCode)

HealthCareCenterPhoneNumbers (CenterId, PhoneNumber)

HealthCareProviders (ProviderId, JobTitle, Name)

HealthCareProviderSchedules (ProviderId, Schedule)

WorksAt (CenterId, ProviderId, StartDate)

Vaccines (VialId, BatchId)

Batches (BatchId, Year, Formulation)

Formulations (Formulation, KnownAllergen)

Appointments (Appointment ID, **PatientId, Time, Date, ProviderId, CenterId, VialId, BatchId**, AdverseEffectEvent)

Justifications:

- Each table has a primary key, ensuring uniqueness.
- Separate tables for multi-valued attributes such as phone numbers and known allergens. This ensures 1NF by eliminating repeating groups.
- All non-key attributes are fully functionally dependent on the key - primary or candidate key, satisfying the requirements for 2NF.
- There are no transitive dependencies within tables. Each non-key attribute is dependent only on the primary key, not on other non-key attributes, satisfying 3NF.

Hence, our relational model accurately reflects the ER diagram's structure, considering multivalued and composite attributes and maintaining 3NF normalization.

6. The Complete Relational Schema: Normalized to BCNF

Our schema is designed to be normalized up to BCNF to minimize data redundancy and dependency.

Patients (PatientId, DateOfBirth, Sex, Ethnicity, Name, Email, City, Street, ZipCode)

- 1NF: All attributes are atomic.
- 2NF: No partial dependencies exist.
- 3NF: No transitive dependencies exist.
- BCNF: Already in BCNF as there are no non-trivial functional dependencies between key attributes.

PatientPhoneNumbers (PatientId, PhoneNumber)

- 1NF: All attributes are atomic.
- 2NF: Already in 2NF as there are no partial dependencies.
- 3NF: Already in 3NF as there are no transitive dependencies.
- BCNF: Already in BCNF as the primary key is the only determinant.

EHR (PatientId, EHR)

- 1NF: All attributes are atomic.
- 2NF: Already in 2NF as there are no partial dependencies.
- 3NF: Already in 3NF as there are no transitive dependencies.
- BCNF: Already in BCNF as the primary key is the only determinant.

HealthCareCenters (CenterId, CenterName, City, Street, ZipCode)

- 1NF: All attributes are atomic.
- 2NF: No partial dependencies exist.
- 3NF: No transitive dependencies exist.
- BCNF: Already in BCNF as there are no non-trivial functional dependencies between key attributes.

HealthCareCenterPhoneNumbers (CenterId, PhoneNumber)

- 1NF: All attributes are atomic.
- 2NF: Already in 2NF as there are no partial dependencies.
- 3NF: Already in 3NF as there are no transitive dependencies.
- BCNF: Already in BCNF as the primary key is the only determinant.

HealthCareProviders (ProviderId, JobTitle, Name)

- 1NF: All attributes are atomic.
- 2NF: Already in 2NF as there are no partial dependencies.
- 3NF: Already in 3NF as there are no transitive dependencies.
- BCNF: Already in BCNF as there are no non-trivial functional dependencies between key attributes.

HealthCareProviderSchedules (ProviderId, Schedule)

- 1NF: All attributes are atomic.
- 2NF: Already in 2NF as there are no partial dependencies.
- 3NF: Already in 3NF as there are no transitive dependencies.
- BCNF: Already in BCNF as the primary key is the only determinant.

WorksAt (CenterId, ProviderId, StartDate)

- 1NF: All attributes are atomic.
- 2NF: Already in 2NF as there are no partial dependencies.
- 3NF: Already in 3NF as there are no transitive dependencies.
- BCNF: Already in BCNF as there are no non-trivial functional dependencies between key attributes.

Vaccines (VialId, BatchId)

- 1NF: All attributes are atomic.
- 2NF: Already in 2NF as there are no partial dependencies.
- 3NF: Already in 3NF as there are no transitive dependencies.
- BCNF: Already in BCNF as the primary key is the only determinant.

Batches (BatchId, Year, Formulation)

- 1NF: All attributes are atomic.
- 2NF: Already in 2NF as there are no partial dependencies.
- 3NF: Already in 3NF as there are no transitive dependencies.
- BCNF: Already in BCNF as there are no non-trivial functional dependencies between key attributes.

Formulations (Formulation, KnownAllergen)

- 1NF: All attributes are atomic.
- 2NF: Already in 2NF as there are no partial dependencies.
- 3NF: Already in 3NF as there are no transitive dependencies.
- BCNF: Already in BCNF as the primary key is the only determinant.

Appointments (AppointmentID, PatientId, Time, Date, ProviderId, CenterId, VialId, BatchId, AdverseEffectEvent)

- 1NF: All attributes are atomic.
- 2NF: Already in 2NF as there are no partial dependencies.
- 3NF: Already in 3NF as there are no transitive dependencies.
- BCNF: Already in BCNF as there are no non-trivial functional dependencies on candidate keys.

7. Lossy/dependency preservation:

(i) Our schema is lossless since for each decomposition, we have either carried a primary key forward as a foreign key or created join tables for many-to-many relationships. This ensures that we can recreate the original tables by performing natural joins on the foreign keys.

For Patients and PatientPhoneNumbers:

Common Attribute: PatientId

Relations:

$R1$=Patients ($\underline{PatientId}$, $DateOfBirth$, Sex, $Ethnicity$, $Name$, $Email$, $City$, $Street$, $ZipCode$)

$R2$=PatientPhoneNumbers($\underline{PatientId, PhoneNumber}$)

Intersection: $R1 \cap R2 = \{PatientId\}$

Conclusion: Lossless, because $R1 \cap R2 \rightarrow R1$ ensures no spurious tuples when joined.

Same for EHR

For HealthCareCenters and HealthCareCenterPhoneNumbers:

Common Attribute: CenterId

Relations:

$R1$=HealthCareCenters($\underline{CenterId}$, $CenterName$, $City$, $Street$, $ZipCode$)

$R2$=HealthCareCenterPhoneNumbers($\underline{CenterId, PhoneNumber}$)

Intersection:

$R1 \cap R2 = \{CenterId\}$

Conclusion: Lossless, because $R1 \cap R2 \rightarrow R1$ ensures no spurious tuples when joined.

Same for HealthCareProviderSchedules

For HealthCareProviders and HealthCareProviderSchedules:

Common Attribute: ProviderId

Relations:

$R1$=HealthCareProviders($\underline{ProviderId}$, $JobTitle$, $Name$)

$R2$=HealthCareProviderSchedules($\underline{ProviderId, Schedule}$)

Intersection:

$R1 \cap R2 = \{ProviderId\}$

Conclusion: Lossless, because $R1 \cap R2 \rightarrow R1$, ensures no spurious tuples when joined.

(ii) The schema preserves dependencies by ensuring that each non-trivial functional dependency is represented in at least one of the tables. By decomposing the tables into 3NF and then BCNF, we are ensuring that dependencies are preserved. We have also made sure to separate multi-valued dependencies into their own tables.

8. Referential integrity constraints:

The referential integrity constraints for the tables in our schema are:

PatientPhoneNumbers: The PatientId must exist in the Patients table.

EHR: The PatientId must exist in the Patients table.

HealthCareCenterPhoneNumbers: The CenterId must exist in the HealthCareCenters table.

HealthCareProviderSchedules: The ProviderId must exist in the HealthCareProviders table.

WorksAt: The CenterId must exist in the HealthCareCenters table, and the ProviderId must exist in the HealthCareProviders table.

Vaccines: The BatchId must exist in the Batches table. For each VialId that exists there must be an associated BatchId.

Appointments: The PatientId must exist in the Patients table, the ProviderId must exist in the HealthCareProviders table, the CenterId must exist in the HealthCareCenters table, and the VialId must exist in the Vaccines table.

9. Functional Dependencies:

For our relational schema, the non-trivial functional dependencies for each relation are as follows:

Patients: PatientId → DateOfBirth, Sex, Ethnicity, Name, Email, City, Street, ZipCode

HealthCareCenters: CenterId → CenterName, City, Street, ZipCode

HealthCareCenterPhoneNumbers : (CenterId, PhoneNumber) → PhoneNumber

HealthCareProviders: ProviderId → JobTitle, Name

WorksAt: (CenterId, ProviderId) → StartDate

Batches: BatchId → Year, Formulation

Appointments: AppointmentID → PatientId, Time, Date, ProviderId, CenterId, VialId, BatchId

10. User Views

Patient:

View for patients within the healthcare system that we manage. Allows for users to view personal information, appointments and update some personal information.

Schema -

Patients (PatientId, DateOfBirth, Sex, Ethnicity, Name, Email, City, Street, ZipCode)

PatientPhoneNumbers (PatientId, PhoneNumber)

EHR (PatientId, EHR)

Appointments (Appointment ID, PatientId, Time, Date, ProviderId, CenterId, VialId, BatchId, AdverseEffectEvent)

Healthcare Provider / Administrator:

View for healthcare providers and administrators (i.e. doctors, nurses, front office). Gives users the ability to create, read, update and delete patient and appointment information. It also gives users the ability to read all information stored in our system.

Schema -

Patients (PatientId, DateOfBirth, Sex, Ethnicity, Name, Email, City, Street, ZipCode)

PatientPhoneNumbers (PatientId, PhoneNumber)

EHR (PatientId, EHR)

HealthCareCenters (CenterId, CenterName, City, Street, ZipCode)

HealthCareCenterPhoneNumbers (CenterId, PhoneNumber)

HealthCareProvider (ProviderId, JobTitle, Name)

HealthCareProviderSchedules (ProviderId, Schedule)

WorksAt (CenterId, ProviderId, StartDate)

Vaccines (VialId, BatchId)

Batches (BatchId, Year, Formulation)

Formulations (Formulation, KnownAllergen)

Appointments (AppointmentID, Time, Date, ProviderId, CenterId, VialId, BatchId, AdverseEffectEvent)

Public Health Official:

View for public health officials. Provides users access to demographic data, vaccine information, hospital information and general information about appointments. Does not allow for access to personal information

Schema -

Patients (PatientId, DateOfBirth, Sex, Ethnicity, City, ZipCode)

HealthCareCenters (CenterId, CenterName, City, Street, ZipCode)

HealthCareCenterPhoneNumbers (CenterId, PhoneNumber)

Vaccines (VialId, BatchId)

Batches (BatchId, Year, Formulation)

Formulations (Formulation, KnownAllergen)

Appointments (AppointmentID, Time, Date, CenterId, VialId, BatchId, AdverseEffectEvent)

11. Data Dictionary

Entity/Relationship	**Attribute**	**Definition**	**Data type**	**Notes**
Patients	PatientID	A unique identifier integer for each patient	int	Cannot be null Auto increments
	DateOfBirth	Date of birth	date	Default null Trigger: Cannot be after the current date
	Sex	Demographic information	varchar(100)	Default null

	Ethnicity	Demographic information	varchar(100)	Default null Must be in ('Male', 'Female', 'Other', 'Prefer not to say')
	Name	A string containing the patient’s first and last name	varchar(100)	Cannot be null First name string and last name string must be of nonzero length and separated by a space eg. Chen Sun
	Email	Email address	varchar(100)	Default null Must follow general email format, eg. something@domain.com
	Address	A combination of city, street, and zip code	street_address varchar(100) city varchar(100) zip_code varchar(100)	Default null all Zipcode must follow standard format, eg. 52240
	PhoneNumbers	Multivalued, contains patient phone #’s	varchar(100)	Cannot be null Optionally start with ‘+’, then 1-3 digit country code followed by 10 digits, eg. +3195555555
	E.H.R.	Multivalued, contains past patient health records	text	Cannot be null

Vaccines	VialID	A unique identifier integer for each vial within a batch (may be repeated in different batches)	int	Cannot be null Auto increments
	BatchID	A unique identifier integer for each batch of vaccine vials	int	Cannot be null Foreign key to Batches
	Year	An integer year for the strain of Influenza that the vaccine was developed for. E.g. 2024	year	Default null
	Formulation	A string containing the type of influenza vaccine. E.g. inactivated influenza vaccines [IIV4], recombinant influenza vaccine [RIV4], and live attenuated influenza vaccine [LAIV4]	varchar(10)	Default null in Batches Cannot be null in Formulations Foreign key to Formulations in Batches Must not be empty string
	KnownAllergens	A list of strings of the ingredients in the vaccine that may cause allergic reactions. (multivalued)	varchar(255)	Cannot be null Default 'None'
HealthCareCenters	CenterId	A unique identifier integer for each	int	Cannot be null Auto_increments

		healthcare center		
	CenterName	Name of healthcare center	varchar(100)	Default null
	PhoneNumbers	Multivalued, contains healthcare center phone #'s	varchar(100)	Cannot be null Optionally start with '+', then 1-3 digit country code followed by 10 digits, eg. +3195555555
	Address	A combination of city, street, and zip code	street_address varchar(100) city varchar(100) zip_code varchar(100)	Default null all Zipcode must follow standard format, eg. 52240
HealthCareProviders	ProviderId	A unique identifier integer for each provider	int	Cannot be null Auto increments
	JobTitle	Title for job (i.e. Nurse, Doctor, etc.)	varchar(100)	Default null
	Name	A string containing the provider's first and last name.	varchar(100)	Cannot be null First name string and last name string must be of nonzero length and separated by a space eg. Chen Sun
	Schedule	A multivalued attribute	text	Cannot be null

		containing schedule information		
Appointments <Relationship>	VialId	A unique identifier integer for each vial within a batch (may be repeated in different batches)	int	Default null Foreign key to VialId
	AppointmentId	A unique identifier integer for each appointment created	int	Cannot be null Auto increments
	BatchId	A unique identifier integer for each batch of vaccine vials	int	Default null Foreign key to Batches
	PatientId	A unique identifier integer for each patient	int	Default null Foreign key to Patients
	ProviderId	A unique identifier integer for each provider	int	Default null Foreign key to HealthCareProviders
	AdverseEffectEvent	Adverse effect related to vaccination	varchar(255)	Default 'None'
	Date	Date of appointment	date	Cannot be null
	CenterId	A unique identifier integer for each healthcare center	int	Default null Foreign key to HealthCareCenters
	Time	Time of day in H:M:S	time	Cannot be null
WorksAt <Relationship>	CenterId	A unique identifier integer for each	int	Cannot be null

		healthcare center		Foreign key to HealthCareCenters
	StartDate	Date that healthcare provider started working at specified healthcare center (is assumed to be a valid date to capture when a provider starts working at a center)	date	Default null
	ProviderId	A unique identifier integer for each provider	int	Cannot be null Foreign key to HealthCareProviders

12. Database Authorizations

Role	**Privileges**
Patient	Patients: **read** (PatientId, DateOfBirth, Sex, Ethnicity, Name, Email, City, Street, ZipCode) PatientPhoneNumbers: **create, read, update, delete** (PhoneNumber) EHR: **read** (PatientId, EHR) Appointments: **read** (AppointmentID, PatientId, Time, Date, VialId, BatchId, AdverseEffectEvent)
Healthcare Provider / Administrator	Patients: **create, read, update, delete** (PatientId, DateOfBirth, Sex, Ethnicity, Name, Email, City, Street, ZipCode) PatientPhoneNumbers: **read** (PatientId); create, read, update, delete (PhoneNumber) EHR: **read** (PatientId); **create, read, update, delete** (EHR) HealthCareCenters: **read** (CenterId, CenterName, City, Street, ZipCode) HealthCareCenterPhoneNumbers: **read** (CenterId, PhoneNumber) HealthCareProviders: **read** (ProviderId, JobTitle, Name) HealthCareProviderSchedules: **read** (ProviderId); create, read, update, delete (Schedule) WorksAt: **read** (CenterId, ProviderId, StartDate) Vaccines: **read** (VialId, BatchId) Batches: **read** (BatchId, Year, Formulation) Formulations: **read** (Formulation, KnownAllergen) Appointments: **create, read, update, delete** (AppointmentID, Time, Date, ProviderId, CenterId, VialId, BatchId, AdverseEffectEvent); read (PatientId)
Public Health Official	Patients: **read** (PatientId, DateOfBirth, Sex, Ethnicity, City, ZipCode) HealthCareCenters: **read** (CenterId, CenterName, City, Street, ZipCode) HealthCareCenterPhoneNumbers: **read** (CenterId, PhoneNumber)

	Vaccines: **read** (VialId, BatchId) Batches: **read** (BatchId, Year, Formulation) Formulations: **read** (Formulation, KnownAllergen) Appointments: **read** (AppointmentID, Time, Date, CenterId, VialId, BatchId, AdverseEffectEvent)

13. Additional Normalization Problem

R(PatientId, DateOfBirth, Sex, Ethnicity, PatientName, Email, Address, PatientPhoneNumber, EHR, VialId, BatchId, Year, Formulation, KnownAllergens, CenterId, CenterName, CenterPhoneNumber, Address, ProviderId, JobTitle, ProviderName, Schedule, AppointmentId, AdverseEffectEvent, Date, CenterId, Time)

Functional Dependencies:

PatientId → PatientId

PatientId → DateOfBirth

PatientId → Sex

PatientId → Ethnicity

PatientId → Name

PatientId → Email

PatientId → City

PatientId → Street

PatientId → ZipCode

Email → Email

Email → PatientId

Email → DateOfBirth

Email → Sex

Email → Ethnicity

Email → Name

Email → City

Email → Street

Email → ZipCode

(PatientId, PatientPhoneNumber) → PatientPhoneNumber

(PatientId, PatientPhoneNumber) → PatientId

(PatientId, EHR) → EHR

(PatientId, EHR) → PatientId

CenterId → CenterId

CenterId → CenterName

CenterId → City

CenterId → Street

CenterId → ZipCode

(CenterId, CenterPhoneNumber) → CenterPhoneNumber

(CenterId, CenterPhoneNumber) → CenterId

ProviderId → ProviderId

ProviderId → JobTitle

ProviderId → Name

(ProviderId, Schedule) → Schedule

(ProviderId, Schedule) → ProviderId

(CenterId, ProviderId) → CenterId

(CenterId, ProviderId) → ProviderId

(CenterId, ProviderId) → StartDate

BatchId → BatchId

BatchId → Year

BatchId → Formulation

(Formulation, KnownAllergen) → KnownAllergen

(Formulation, KnownAllergen) → Formulation

AppointmentID → AppointmentID

AppointmentID → PatientId

AppointmentID → Time

AppointmentID → Date

AppointmentID → ProviderId

AppointmentID → CenterId

AppointmentID → VialId

AppointmentID → BatchId

(VialId, BatchId) → AppointmentID

(VialId, BatchId) → PatientId

(VialId, BatchId) → Time

(VialId, BatchId) → Date

(VialId, BatchId) → ProviderId

(VialId, BatchId) → CenterId

(VialId, BatchId) → VialId

(VialId, BatchId) → BatchId

(Time, Date, PatientId, CenterId) → AppointmentId

(Time, Date, PatientId, CenterId) → PatientId

(Time, Date, PatientId, CenterId) → Time

(Time, Date, PatientId, CenterId) → Date

(Time, Date, PatientId, CenterId) → ProviderId

(Time, Date, PatientId, CenterId) → CenterId

(Time, Date, PatientId, CenterId) → VialId

(Time, Date, PatientId, CenterId) → BatchId

Closure:

* Bold attributes are candidate keys

PatientId$^+$ = PatientId, DateOfBirth, Sex, Ethnicity, PatientName, **Email**, Address

Email$^+$ = PatientId, DateOfBirth, Sex, Ethnicity, PatientName, Email, Address

(PatientId, PatientPhoneNumber) $^+$= PatientId, PatientPhoneNumber, DateOfBirth, Sex, Ethnicity, PatientName, Email, Address

(PatientId, EHR) $^+$= PatientId, DateOfBirth, Sex, Ethnicity, PatientName, Email, Address, EHR

CenterId$^+$ = CenterId, CenterName, City, Street, ZipCode

(CenterId, CenterPhoneNumber)$^+$ = CenterId, CenterName, City, Street, ZipCode, CenterPhoneNumber

ProviderId$^+$ = ProviderId, JobTitle, ProviderName

(ProviderId, Schedule)$^+$ = ProviderId, Schedule, JobTitle, ProviderName

(CenterId, ProviderId)$^+$ = CenterId, CenterName, City, Street, ZipCode, StartDate, ProviderId, JobTitle, ProviderName

BatchId$^+$ = BatchId, Year, Formulation

(Formulation, KnownAllergen)$^+$ = Formulation, KnownAllergen

AppointmentID$^+$ = AppointmentId, **Time, Date,** AdverseEffectEvent, **PatientId**, DateOfBirth, Sex, Ethnicity, PatientName, Email, Address, EHR, ProviderId, JobTitle, ProviderName, **CenterId**, StartDate, CenterName, City, Street, ZipCode, **VialId, BatchId**, Year, Formulation

(VialId, BatchId)$^{+}$ = AppointmentId, Time, Date, AdverseEffectEvent, PatientId, DateOfBirth, Sex, Ethnicity, PatientName, Email, Address, EHR, ProviderId, JobTitle, ProviderName, CenterId, StartDate, CenterName, City, Street, ZipCode, VialId, BatchId, Year, Formulation

(Time, Date, CenterId, PatientId, ProviderId)$^{+}$ = AppointmentId, Time, Date, AdverseEffectEvent, PatientId, DateOfBirth, Sex, Ethnicity, PatientName, Email, Address, EHR, ProviderId, JobTitle, ProviderName, CenterId, StartDate, CenterName, City, Street, ZipCode, VialId, BatchId, Year, Formulation

Primary Keys:

PatientId, (PatientId, PatientPhoneNumber), (PatientId, EHR), CenterId, (CenterId, CenterPhoneNumber), ProviderId, (ProviderId, Schedule), (CenterId, ProviderId), BatchId, (Formulation, KnownAllergen)

Candidate Keys:

Email, (Time, Date, PatientId, CenterId), (VialId, BatchId)

Building 3NF:

Patients (PatientId, DateOfBirth, Sex, Ethnicity, Name, **Email**, City, Street, ZipCode)

PatientPhoneNumbers (PatientId, PhoneNumber)

EHR (PatientId, EHR)

HealthCareCenters (CenterId, CenterName, City, Street, ZipCode)

HealthCareCenterPhoneNumbers (CenterId, PhoneNumber)

HealthCareProviders (ProviderId, JobTitle, Name)

HealthCareProviderSchedules (ProviderId, Schedule)

WorksAt (CenterId, ProviderId, StartDate)

Vaccines (VialId, BatchId)

Batches (BatchId, Year, Formulation)

Formulations (Formulation, KnownAllergen)

Appointments (AppointmentID, PatientId, **Time, Date**, **CenterId**, **Provider Id**, **VialId, BatchId**, AdverseEffectEvent)

* This design is lossless and preserves relations

14. Implementation

The submitted .sql file has descriptive comments as per the instructions document. The .php file is present in our group space cs4400_group9.