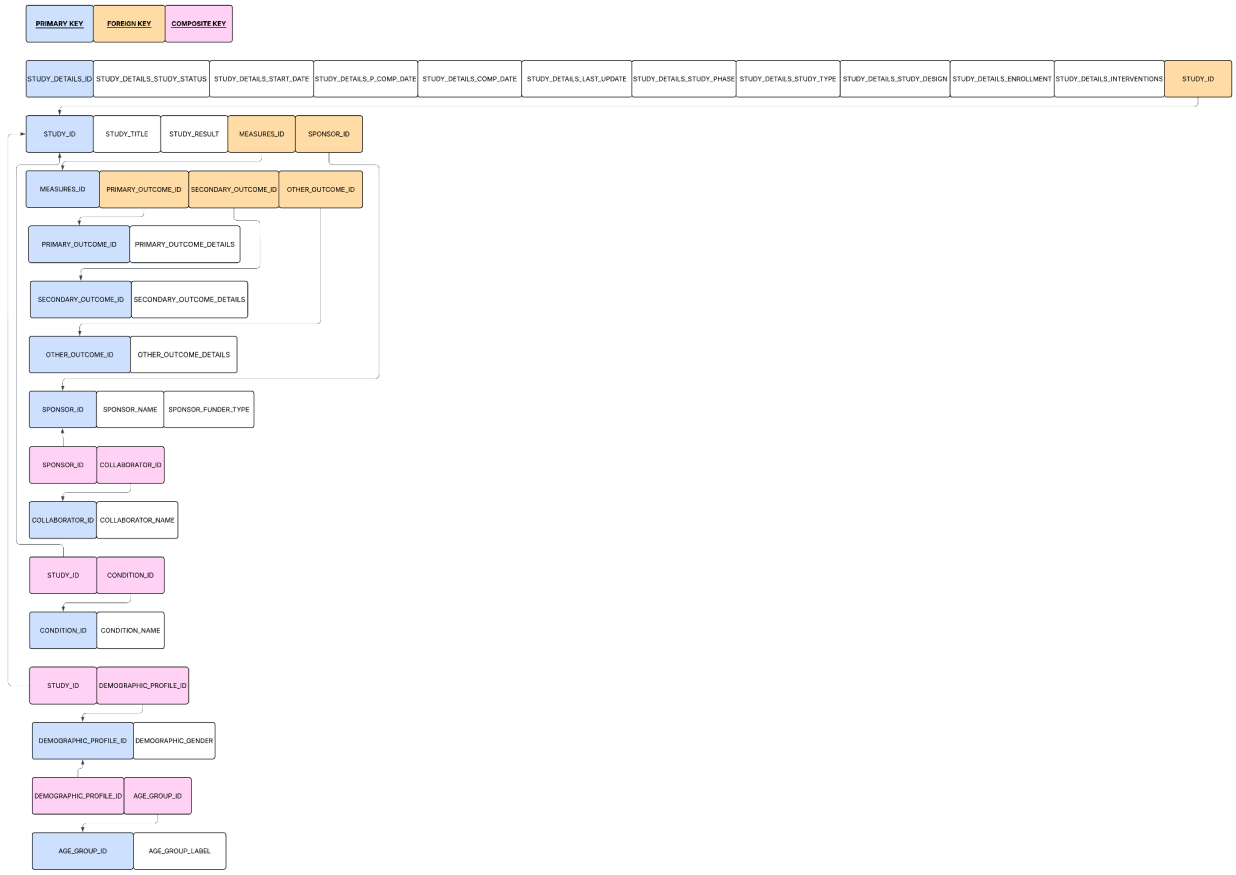
IME 312 Clinical Research Database

Part 1B Deliverable

Cyrus Mak, Mryon Chen, Noah Pidding

Table of Contents

Normalization



Previously, our rows from sponsor_collaborators, study_conditions, and demographic_age_group contained multiple values separated by “|” or commas. The table below provides an example of rows pre-normalization.

SPONSOR_COLLABORATORS	STUDY_CONDITION	DEMOGRAPHIC_AGE_GROUP
Northwestern University|National Institute on Aging (NIA)|University of Washington	Adenocarcinoma|Pancreatic Neoplasms|Neoplasm, Glandular|Neoplasms|Neoplasms Pancreatic|Digestive	ADULT, OLDER ADULT, CHILD

	System Neoplasm|Endocrine Gland Neoplasms|Digestive System Disease|Pancreatic Diseases|Endocrine System Diseases	

These fields were multivalued and non-atomic, violating **1NF.** This also meant that they weren't normalized for lookup, querying or reusability, violating **3NF.**

To resolve these issues, separate junction tables were created for each of the 3 categories to record the relationships of the multiple values. **Note that all text in parentheses within tables are there for reference and will not actually be included in the tables.**

AGE_GROUP Table (New) and DEMOGRAPHIC Table are linked together through a many to many relationship through the DEMOGRAPHIC_AGE_GROUP Junction table (New)

DEMOGRAPHIC_PROFILE_ID	AGE_GROUP_ID
1001	1 (CHILD)
1001	2 (ADULT)
1002	3 (OLDER_ADULT)

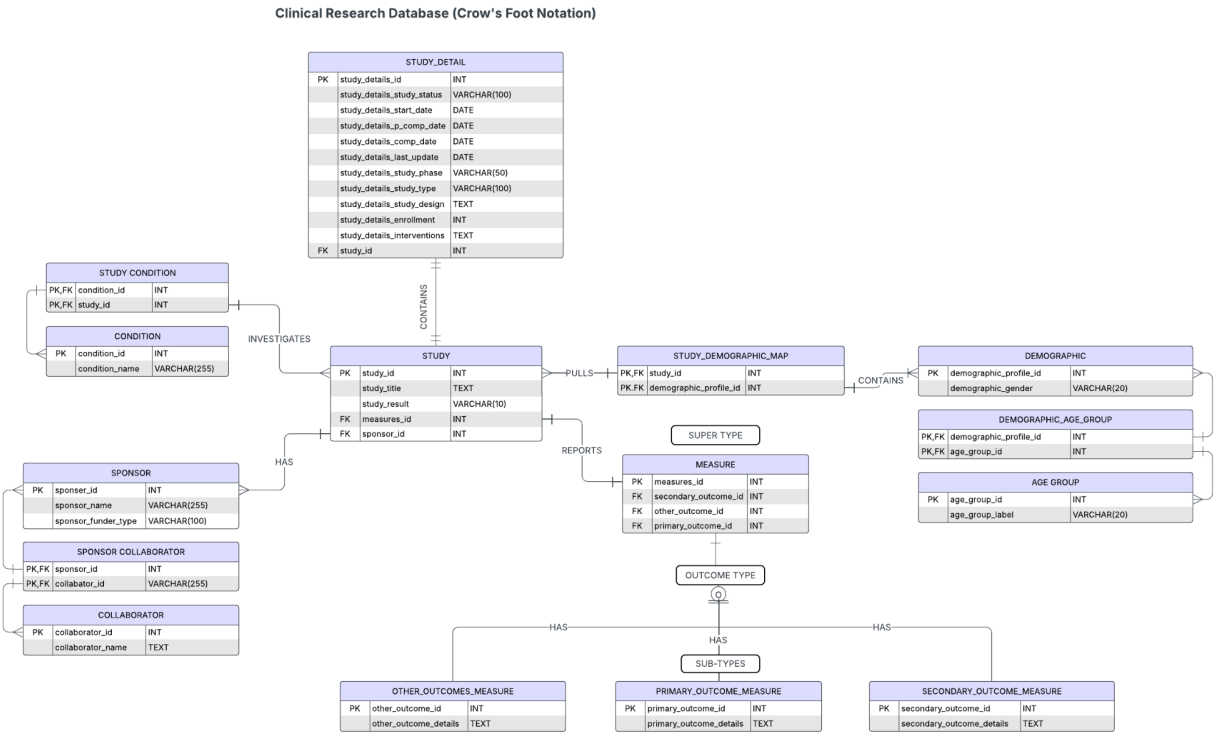
- This ensured participants can be easily queried by age group
- AGE_GROUP values can be consistently reused across the database
- Data integrity (No typos like "Adult", "ADULT", "adult")

STUDY Table and CONDITION Table (New) are linked together through a many-to-many relationship through the STUDY_CONDITION (New) Junction Table.

STUDY_ID	CONDITION_ID
NCT001	10 (Asthma)
NCT001	12 (Allergies)
NCT002	11 (Diabetes)

- This ensured participants can be easily queried by conditions
- CONDITION_ID values can be consistently reused across the STUDY

Final ERD



Create Database

```
-- Drop and create SPONSOR
DROP TABLE IF EXISTS SPONSOR;
CREATE TABLE SPONSOR (
    sponsor_id INT PRIMARY KEY,
    sponsor_name VARCHAR(255),
    sponsor_funder_type VARCHAR(100)
);

-- Drop and create COLLABORATOR
DROP TABLE IF EXISTS COLLABORATOR;
CREATE TABLE COLLABORATOR (
```

```
    collaborator_id INT PRIMARY KEY,
    collaborator_name TEXT
);

-- Drop and create SPONSOR_COLLABORATOR (Many-to-Many between SPONSOR and
COLLABORATOR)
DROP TABLE IF EXISTS SPONSOR_COLLABORATOR;
CREATE TABLE SPONSOR_COLLABORATOR (
    sponsor_id INT,
    collaborator_id INT,
    PRIMARY KEY (sponsor_id, collaborator_id),
    FOREIGN KEY (sponsor_id) REFERENCES SPONSOR(sponsor_id),
    FOREIGN KEY (collaborator_id) REFERENCES COLLABORATOR(collaborator_id)
);

-- Drop and create STUDY
DROP TABLE IF EXISTS STUDY;
CREATE TABLE STUDY (
    study_id INT PRIMARY KEY,
    study_title TEXT,
    study_result VARCHAR(10),
    measures_id INT,
    sponsor_id INT,
    FOREIGN KEY (measures_id) REFERENCES MEASURE(measures_id),
    FOREIGN KEY (sponsor_id) REFERENCES SPONSOR(sponsor_id)
);

-- Drop and create STUDY_DETAIL
DROP TABLE IF EXISTS STUDY_DETAIL;
CREATE TABLE STUDY_DETAIL (
    study_details_id INT PRIMARY KEY,
    study_details_study_status VARCHAR(100),
    study_details_start_date DATE,
    study_details_completion_date DATE,
```

```
    study_details_primary_comp_date DATE,
    study_details_last_update DATE,
    study_details_study_phase VARCHAR(50),
    study_details_study_type VARCHAR(100),
    study_details_study_design TEXT,
    study_details_enrollment INT,
    study_details_interventions TEXT,
    study_id INT,
    FOREIGN KEY (study_id) REFERENCES STUDY(study_id)
);

-- Drop and create CONDITION
DROP TABLE IF EXISTS CONDITION;
CREATE TABLE CONDITION (
    condition_id INT PRIMARY KEY,
    condition_name VARCHAR(255)
);

-- Drop and create STUDY_CONDITION (Many-to-Many between STUDY and CONDITION)
DROP TABLE IF EXISTS STUDY_CONDITION;
CREATE TABLE STUDY_CONDITION (
    condition_id INT,
    study_id INT,
    PRIMARY KEY (condition_id, study_id),
    FOREIGN KEY (condition_id) REFERENCES CONDITION(condition_id),
    FOREIGN KEY (study_id) REFERENCES STUDY(study_id)
);

-- Drop and create DEMOGRAPHIC
DROP TABLE IF EXISTS DEMOGRAPHIC;
CREATE TABLE DEMOGRAPHIC (
    demographic_profile_id INT PRIMARY KEY,
    demographic_gender VARCHAR(20)
);
```

```
-- Drop and create AGE_GROUP
DROP TABLE IF EXISTS AGE_GROUP;
CREATE TABLE AGE_GROUP (
   age_group_id INT PRIMARY KEY,
   age_group_label VARCHAR(200)
);

-- Drop and create DEMOGRAPHIC_AGE_GROUP (many-to-one: demographic → age group)
DROP TABLE IF EXISTS DEMOGRAPHIC_AGE_GROUP;
CREATE TABLE DEMOGRAPHIC_AGE_GROUP (
   demographic_profile_id INT,
   age_group_id INT,
   PRIMARY KEY (demographic_profile_id, age_group_id),
   FOREIGN KEY (demographic_profile_id) REFERENCES
DEMOGRAPHIC(demographic_profile_id),
   FOREIGN KEY (age_group_id) REFERENCES AGE_GROUP(age_group_id)
);

-- Drop and create STUDY_DEMOGRAPHIC_MAP
DROP TABLE IF EXISTS STUDY_DEMOGRAPHIC_MAP;
CREATE TABLE STUDY_DEMOGRAPHIC_MAP (
   study_id INT,
   demographic_profile_id INT,
   PRIMARY KEY (study_id, demographic_profile_id),
   FOREIGN KEY (study_id) REFERENCES STUDY(study_id),
   FOREIGN KEY (demographic_profile_id) REFERENCES
DEMOGRAPHIC(demographic_profile_id)
);

-- Drop and create MEASURE
DROP TABLE IF EXISTS MEASURE;
CREATE TABLE MEASURE (
   measures_id INT PRIMARY KEY,
```

```
    primary_outcome_id INT,
    secondary_outcome_id INT,
    other_outcome_id INT,
    FOREIGN KEY (primary_outcome_id) REFERENCES
PRIMARY_OUTCOME_MEASURE(primary_outcome_id),
    FOREIGN KEY (secondary_outcome_id) REFERENCES
SECONDARY_OUTCOME_MEASURE(secondary_outcome_id),
    FOREIGN KEY (other_outcome_id) REFERENCES
OTHER_OUTCOMES_MEASURE(other_outcome_id)
);

-- Drop and create outcome tables
DROP TABLE IF EXISTS PRIMARY_OUTCOME_MEASURE;
CREATE TABLE PRIMARY_OUTCOME_MEASURE (
    primary_outcome_id INT PRIMARY KEY,
    primary_outcome_details TEXT
);

DROP TABLE IF EXISTS SECONDARY_OUTCOME_MEASURE;
CREATE TABLE SECONDARY_OUTCOME_MEASURE (
    secondary_outcome_id INT PRIMARY KEY,
    secondary_outcome_details TEXT
);

DROP TABLE IF EXISTS OTHER_OUTCOMES_MEASURE;
CREATE TABLE OTHER_OUTCOMES_MEASURE (
    other_outcome_id INT PRIMARY KEY,
    other_outcome_details TEXT
);
```

Data Population

To populate our data into the SQL database, we used CSV imports. We utilized a combination of Python and ChatGPT to create and prepare the CSV files for import. We began with the initial dataset from the Clinical Trials database (ctg-studies.csv). Using Python, we analyzed the data to identify repeating values and relationships between columns, which guided us in normalizing the dataset into multiple related tables. As a result, we split the data into several CSV files: study.csv, study_detail.csv, study_demographic_map.csv, sponsor.csv, measure.csv, primary_outcome_measure.csv, secondary_outcome_measure.csv, other_outcome_measure.csv, and demographic.csv. With assistance from ChatGPT, we generated supplemental data where necessary to align with the structure and integrity constraints of our database schema. This allowed us to ensure consistency across foreign keys and maintain proper normalization. Each CSV file corresponds to a specific table in our relational schema, supporting efficient storage and retrieval of clinical trial information.

Python Script:

Data Population

CSV Files:

study.csv

study_detail.csv

study_demographic_map.csv

sponsor.csv

measure.csv

primary_outcome_measure.csv

secondary_outcome_measure.csv

other_outcome_measure.csv

demographic.csv

Testing

Query 1: Shows the top 10 most commonly researched conditions in the database. This helps users quickly identify high-interest or high-prevalence health issues being studied in clinical trials.

```
SELECT
    c.condition_name,
    COUNT(*) AS study_count
FROM
    STUDY_CONDITION sc
JOIN
    CONDITION c ON sc.condition_id = c.condition_id
GROUP BY
    c.condition_name
ORDER BY
    study_count DESC
LIMIT 10;
```

study_conditions	study_count
Healthy	729
Breast Cancer	484
Prostate Cancer	364
Multiple Myeloma	269
Obesity	237
Parkinson Disease	209
Cancer	204
Healthy Volunteers	176
Stroke	166
Heart Failure	159

Query 2: Provides an overview of how many studies exist by phase and type (e.g., Phase 3 Interventional) and their average enrollment sizes. Useful for evaluating the maturity and scale of different trial categories.

```
SELECT
    sd.study_details_study_phase AS phase,
    sd.study_details_study_type AS type,
    COUNT(*) AS total_studies,
    ROUND(AVG(sd.study_details_enrollment), 1)
AS avg_enrollment
FROM
    STUDY_DETAIL sd
WHERE
    sd.study_details_enrollment IS NOT NULL
GROUP BY
    sd.study_details_study_phase, sd.study_details_study_type
ORDER BY
    total_studies DESC;
```

phase	type	total_studies	avg_enrollment
	INTERVENTIONAL	2887	2263.2
	OBSERVATIONAL	1162	16647.4
PHASE2	INTERVENTIONAL	1086	116.2
PHASE1	INTERVENTIONAL	814	60.4
PHASE3	INTERVENTIONAL	551	945.3
PHASE1|PHASE2	INTERVENTIONAL	357	104.8
PHASE4	INTERVENTIONAL	326	202.9
EARLY_PHASE1	INTERVENTIONAL	165	52.3
PHASE2|PHASE3	INTERVENTIONAL	92	478.8
	EXPANDED_ACCESS	22	0

Query 3: Summarizes recurring outcome themes across studies. Helps users identify what results researchers are most often measuring in trials (e.g., survival, blood pressure, symptom reduction).

```
SELECT
    LEFT(pom.primary_outcome_details, 50) AS outcome_summary,
    COUNT(*) AS num_studies
FROM
    PRIMARY_OUTCOME_MEASURE pom
JOIN
    MEASURE m ON pom.primary_outcome_id
= m.primary_outcome_id
JOIN
    STUDY st ON m.measures_id =
st.measures_id
GROUP BY
    outcome_summary
ORDER BY
    num_studies DESC
LIMIT 10;
```

outcome_summary	num_studies
Number of Participants With Treatment-emerge…	164
Number of Participants with Adverse Events (A…	98
Number of participants with treatment-related a…	92
Number of Participants with Dose Limiting Toxiciti	80
Number of participants with treatment emergen…	76
Incidence of treatment-emergent adverse even…	57
Number of Participants With Dose-limiting Toxiciti	55
Incidence of Treatment-Emergent Adverse Eve…	47
Area under the Concentration-Time Curve from …	45
Number of Participants With Adverse Events (A…	44

Query 4: Finds studies where a particular condition is linked to a specific outcome. Helps users explore research relevant to their symptoms or questions about treatment impact.

Swappable values:

'%Anxiety%' → '%Diabetes%', '%Sepsis%', '%Depression%'

'%Sleep%' → '%Pain%', '%Recovery%', '%Mortality%'

```
SELECT
    st.study_id,
    st.study_title,
    c.condition_name,
    pom.primary_outcome_details,
    sd.study_details_study_phase
FROM
    STUDY st
```

```
JOIN
    STUDY_DETAIL sd ON st.study_id = sd.study_id
JOIN
    MEASURE m ON st.measures_id = m.measures_id
JOIN
    PRIMARY_OUTCOME_MEASURE pom ON m.primary_outcome_id = pom.primary_outcome_id
JOIN
    STUDY_CONDITION sc ON st.study_id = sc.study_id
JOIN
    CONDITION c ON sc.condition_id = c.condition_id
WHERE
    c.condition_name LIKE '%Anxiety%'
    AND pom.primary_outcome_details LIKE '%Sleep%';
```

study_id	study_title	study_conditions	primary_outcome_details	study_details_study_phase
NCT06644573	Evaluating the Efficacy and Safety of PROSOM...	Chronic Insomnia|Sleep Deprivation|REM Behavi...	Reduction in Homeostatic Sleep Pressure, Evalu...	PHASE1
NCT05294991	Wellness App for Sleep Disturbance in Hematolo...	Cancer|Sleep Disturbance|Anxiety|Depression|I...	Sleep Disturbance (subjective), Sleep disturban...	
NCT03980067	Impact of Pre-Sedation Virtual Reality Game on ...	Behavior, Child|Anxiety	Post-Hospitalization Behavior Questionnaire (PH...	
NCT05826860	Storytelling and Mindfulness for Graduate Stude...	Depression|Anxiety|Burnout, Student	Change from Baseline in Five Facet Mindfulness ...	

Query 5: Lets users target trials based on their development stage and design method. Useful for identifying trials that are early-phase exploratory vs. late-phase confirmatory.

Swappable phase values:

'PHASE1', 'PHASE2', 'PHASE3', 'PHASE4'

Swappable type values:

'INTERVENTIONAL', 'OBSERVATIONAL', 'EXPANDED ACCESS'

```
SELECT
    st.study_id,
    st.study_title,
    sd.study_details_study_phase,
    sd.study_details_study_type,
    sd.study_details_enrollment
```

```
FROM
   STUDY st
JOIN
   STUDY_DETAIL sd ON st.study_id = sd.study_id
WHERE
   sd.study_details_study_phase = 'PHASE2'
   AND sd.study_details_study_type = 'INTERVENTIONAL'
LIMIT 10;
```

study_id	study_title	study_details_study_phase	study_details_study_type	study_details_enrollment
NCT02000427	Blinatumomab in Adults With Relapsed/Refracto...	PHASE2	INTERVENTIONAL	45.0
NCT04111627	Exercise Plus Duloxetine for Knee Osteoarthritis	PHASE2	INTERVENTIONAL	30.0
NCT06637527	The Study Assessing the Safety and Efficacy of...	PHASE2	INTERVENTIONAL	48.0
NCT05552027	Minnesota HealthSolutions Cellular Car Seat Study	PHASE2	INTERVENTIONAL	92.0
NCT02545127	Merotocin in Mothers With Inadequate Milk Prod...	PHASE2	INTERVENTIONAL	4.0
NCT06224673	ARX788 for Treating Patients With HER2-low Lo...	PHASE2	INTERVENTIONAL	36.0
NCT02965573	A Study to Evaluate the Safety, Efficacy, and P...	PHASE2	INTERVENTIONAL	24.0
NCT02689427	Enzalutamide and Paclitaxel Before Surgery in T...	PHASE2	INTERVENTIONAL	24.0
NCT05130827	Study of Plinabulin and Pegfilgrastim in People ...	PHASE2	INTERVENTIONAL	17.0
NCT05139927	Using MOST to Optimize an Intervention to Incr...	PHASE2	INTERVENTIONAL	438.0

Reflection & Future Work

Going into this project, we decided that our primary goal was to develop a relational database that proved beneficial to locating relevant information while preserving data integrity and maintaining a relatively simple design for analysts to query and update. By the end, we were able to develop raw, unfiltered data into a system capable of outputting real-world information using industry relevant relationships while guarding against anomalies.

Throughout the process of the project, we were often met with difficulties handling the data-cleaning and sorting of the data. Splitting up the multivalued fields into tables linked with key variables helps prevent redundancy and allows the databases to be very scalable while ensuring that the database follows a third normal form– critical to normalization and data

consistency. Additionally the database allowed us to work heavily with foreign keys and junction tables to help us further understand and model the many-to-many relationships within the dataset.

Further work on this project would be to focus on performance optimization. Because of the size of the database, the initial setup and some queries were clunky but we know that the data that can be used on our database will continue to grow. As more data will be added into the current database, the query response times will become slower. Thus we could implement indexes on frequently queried columns or partitioning large tables to help reduce query times.

Overall, this project provided valuable experience to understanding the process of designing and testing a database as well as the capability of resources like SQL. It highlighted the importance of a clear plan and visualization of our dataset to optimally produce a system that would return the requested information from our dataset.

While the current database structure is a great jumping off point, future development could include linking additional data sources such as patient data so that analysts can see the relevance of the testing. With further refinement and expansions, we believe that this system can become a tool that can be used to keep our healthcare providers easily up to date on relevant studies and remedies to maximize our quality of care.