MLOps Platform for Experiment Tracking and Model Management

Milestone: Project Report

Group 11

Palak Tanwar

Poornika Jalavadi

+1 (617)404-6023

+1 (617)992-5029

tanwar.pa@northeastern.edu

jalavadi.p@northeastern.edu

Percentage of Effort Contributed by Student 1: 50%

Percentage of Effort Contributed by Student 2: 50%

Signature of Student 1: 

Signature of Student 2: 

Table of Content

Introduction

As machine learning adoption grows across industries, managing the complexity of experiments, datasets, and models has become one of the biggest challenges for data science teams. Unlike traditional software development, where version control and CI/CD pipelines are mature, machine learning introduces unique difficulties:

- Experiments produce not just code, but also large datasets, parameters, metrics, and model artifacts.
- Results must be reproducible, but without systematic tracking, recreating an experiment is nearly impossible.
- Multiple versions of datasets and models must be compared to evaluate progress.
- Deployment requires a record of which model version went live, under what conditions, and in which environment.

Our project proposes an MLOps platform that addresses these issues through structured experiment tracking, model registry, and deployment history management. By designing a hybrid database backend that combines the reliability of relational databases with the flexibility of NoSQL's storage, the platform creates a single source of truth for machine learning workflows.

Theory

The foundation of MLOps lies in ensuring reproducibility, traceability, and scalability across the ML lifecycle. An MLOps platform enforces:

1. Structured Experiment Tracking: Every run of a model training job must be logged with metadata such as datasets, parameters, metrics, and outputs. This ensures results can be replicated later.
2. Dataset and Model Versioning: Just as source code has versions, datasets and models require version history. Tracking dataset versions prevents inconsistencies, while a model registry allows organizations to compare, validate, and deploy models confidently.
3. Hybrid Database Approach:

    a) Relational Database (MySQL): Best suited for structured metadata like users, projects, experiment runs, and model registries. These entities require clear relationships and constraints (foreign keys) to maintain integrity.

    b) NoSQL Database (MongoDB): Hyperparameters, metrics, and logs differ between runs. Storing them in flexible JSON documents avoids rigid schemas while still linking them to structured entities via unique identifier (run_id).

By combining these two approaches, the system balances data consistency with flexibility, a necessity in ML workflows.

Project Scope and Requirements

The platform's database design combines relational tables (MySQL) for structured relationships with NoSQL collections (MongoDB) for flexible experiment-specific data.

1. Users, Projects, and Experiments: Users create projects which contain multiple experiments. Experiments represent logical groupings of work, while experiment runs capture each attempt at training.
2. Datasets and Versions: The platform tracks datasets and their versions, ensuring reproducibility when training with updated or modified data.
3. Models, Versions, and Deployments: Models are stored in a central registry, with versions tied to the experiment runs that created them. Deployment history records when and where a model version is used (e.g., staging, production).
4. Artifacts: Outputs such as logs, plots, or binary files are linked to runs for reference and reproducibility.
5. Parameters, Metrics, and Logs (MongoDB): Hyperparameters and evaluation metrics are highly variable across runs, so they are stored in a flexible document-based structure. Logs are also captured in NoSQL collections, supporting unstructured data.

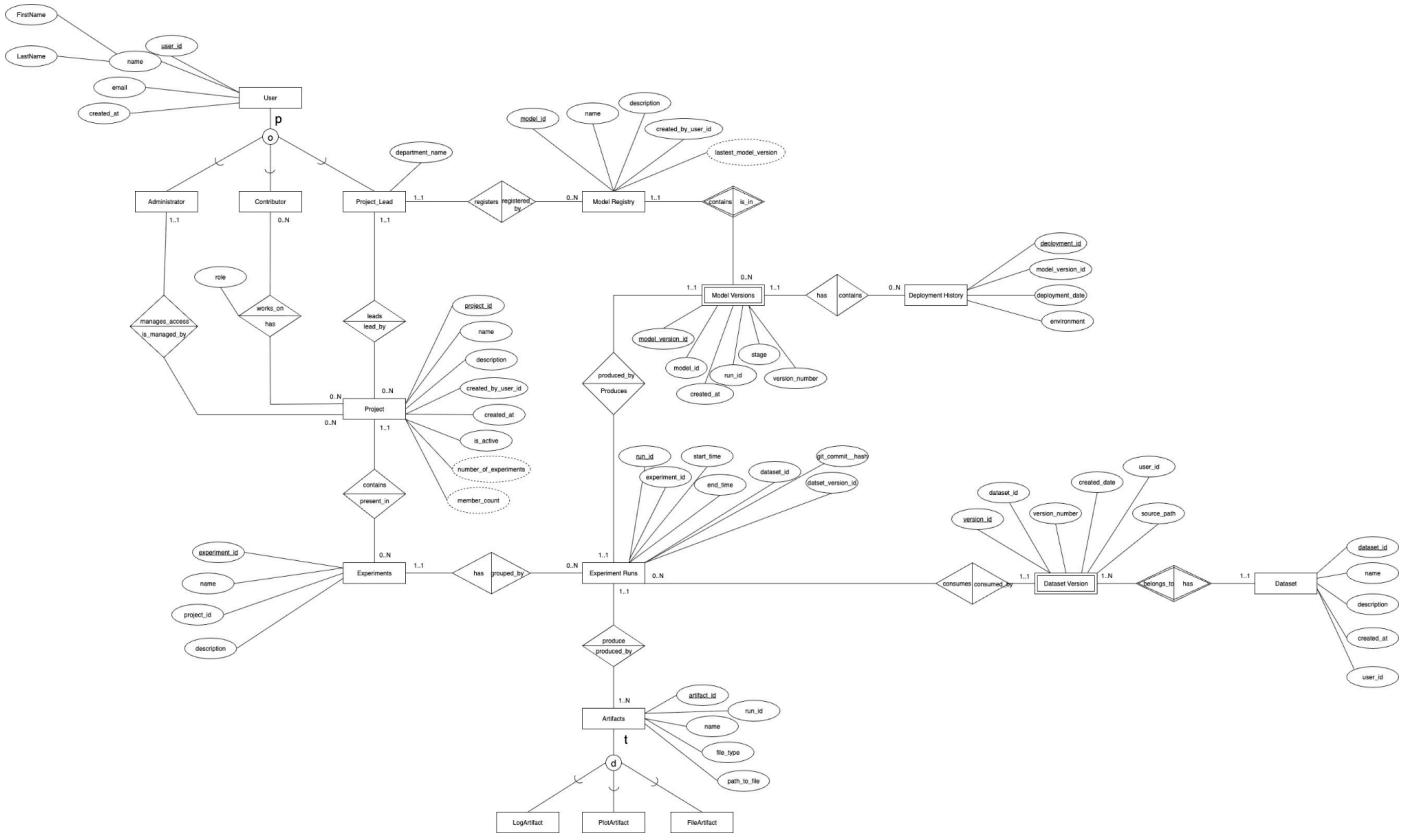
Other Requirements

1. A project can have multiple experiments; each experiment belongs to one project.
2. An experiment can have many runs; each run must belong to exactly one experiment.
3. A dataset can have multiple versions; each version belongs to one dataset.
4. A model can have multiple versions; each version is linked to one run.
5. A model version can be deployed in multiple environments; each deployment belongs to one version.
6. An experiment run can log multiple parameters, metrics, and artifacts; each is linked to that run.
7. Users can create multiple projects; each project can belong to multiple users.

Enhanced Entity Relation Diagram ((E)ER)

This diagram illustrates the relationship between entities in the database system. The key entities are:

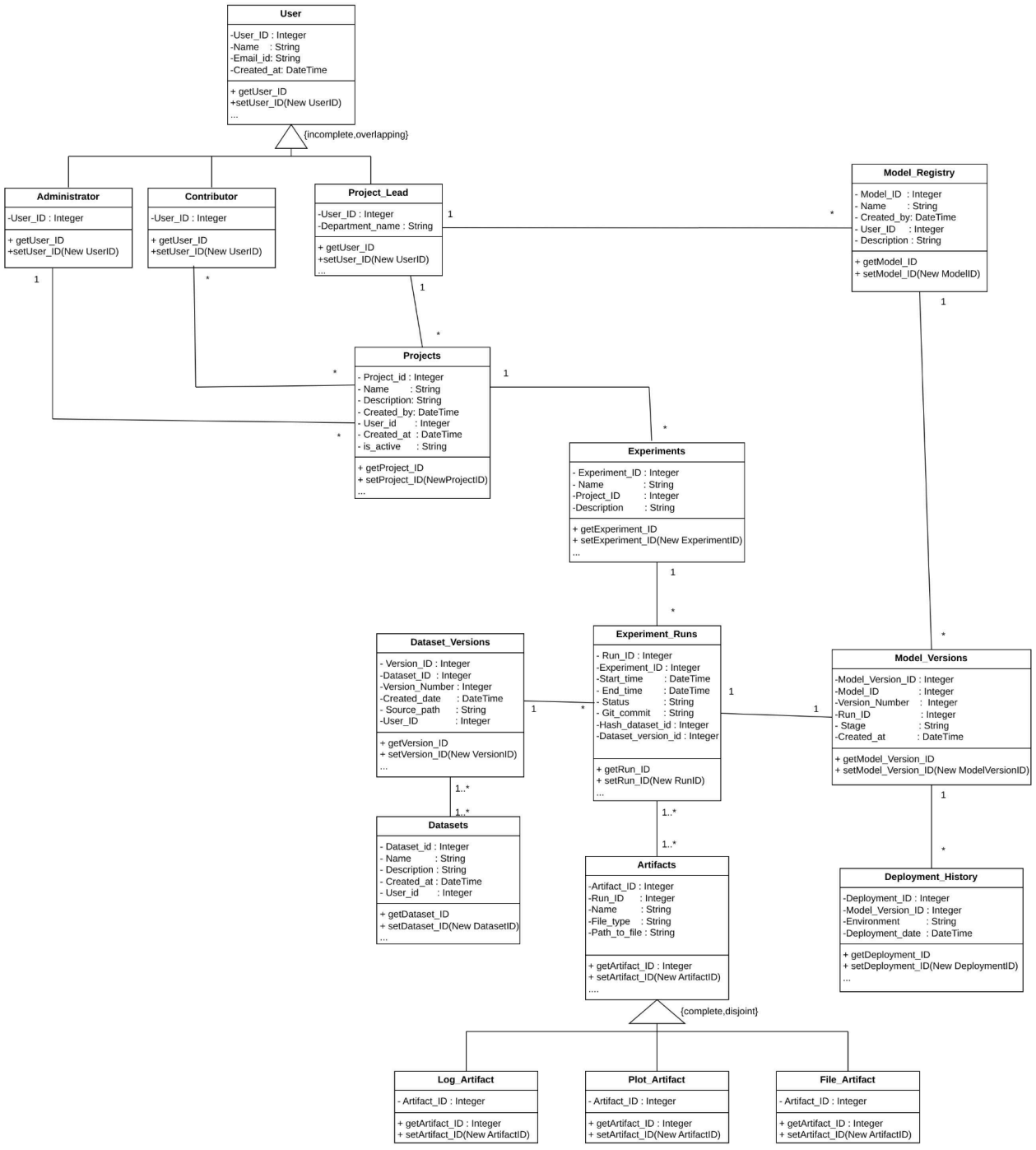
1. User
2. Administrator
3. Contributor
4. Project Lead
5. Project
6. Model Registry
7. Model Versions
8. Experiments
9. Experiment Runs
10. Dataset
11. Dataset Versions
12. Deployment History



The MLOps platform's framework is shown in the EER diagram, connecting essential elements like User, Project, and Experiment to ensure traceable workflows. It shows how Experiment Runs use specialisation hierarchies to efficiently categorise Artifacts (Log, Plot, File) and create User roles while connecting Dataset Versions to Model Versions for reproducibility.

Unified Modeling Language (UML) Class Diagram

This diagram defines the blueprint of the system, by visualizing the relationship between different components. It shows the inheritance logic in the system, for instance, separating Artifacts into distinct categories and also defining specialized user roles, at the same time enforcing the cardinality constraints..



Business Rules And Assumptions

User Management

Business Rules:

- Users must have first name, last name, and email.
- Users can have overlapping roles: Administrator, Contributor, or Project Lead

Assumptions:

- User IDs are system-generated and unique
- Email addresses are unique per user
- User roles determine access permissions throughout the system

Project Management

Business Rules:

- Projects must be managed by exactly one Project Lead
- Projects can have zero or many contributors
- Projects contain zero or many experiments
- Projects must have a name, description, and creation timestamp

Assumptions:

- A user can manage multiple projects
- Users can simultaneously be contributors and leads on different projects
- Projects exist independently and must be created before experiments can be added

Model Registry

Business Rules:

- The Model Registry organizes model versions hierarchically: Model Registry→ Model Version
- Models must have a unique ID, name, description, and be created by a user
- Model Versions must belong to exactly one Model
- Each Model Version tracks: version number, stage, run_id, creation timestamp, and optional latest version indicator

Assumptions:

- Model IDs and version numbers form composite keys for versioning
- The "stage" field indicates the status of the model (e.g: training ongoing, training started)
- Only one version can be marked as "latest" per model
- Model creators maintain ownership/responsibility for their models

Experiment Tracking

Business Rules:

- An experiment belongs to exactly one Project
- Each Experiment can have zero or many Experiment Runs
- Experiment Runs must reference exactly one Experiment
- Runs track multiple attributes: start_time, end_time, experiment_id, etc.

Assumptions:

- Experiments can be created without initial experiment runs
- Multiple runs allow for experiment repeatability and comparison

Model Versions & Deployment

Business Rules:

- Model Versions is produced by exactly one Experiment Run
- Model Versions can consume zero or many Datasets (for training/evaluation)
- Each Model Version tracks deployment through Deployment History
- Deployment History records: environment, deployment date, and deployment-specific version info

Assumptions:

- Models can be deployed to multiple environments (dev, staging, production)
- Version tracking enables rollback capabilities
- Deployment history provides audit trail for compliance

Dataset Management

Business Rules:

- Datasets can have zero or many Dataset Versions
- Dataset Versions can be consumed by zero or many Model Versions (many-to-many relationship)
- Datasets must have: name, description, creation timestamp, and user_id
- Experiment Runs link to Artifacts through a one-to-many relationship

Assumptions:

- Datasets are versioned to track data lineage
- The same dataset version can train multiple model versions
- Users own/maintain datasets they create
- Dataset versions are immutable once created

Artifact Storage

Business Rules:

- Artifacts belong to exactly one Experiment Run
- Artifacts are classified into three mutually exclusive types: LogArtifact, PlotArtifact, or FileArtifact (indicated by "d" disjoint constraint)
- All artifacts must have: artifact_id, name, file_type, and path_to_file

Assumptions:

- LogArtifacts store execution logs and metrics
- PlotArtifacts store plot related metrics
- FileArtifacts store actual data files (CSV, models, images, etc.)
- Artifact storage paths reference external file systems or object storage
- The disjoint constraint ensures clear artifact categorization

Cardinality & Relationship Rules

Key Constraints:

- **1:1 relationships**: Each run produces exactly one set of results
- **1:N relationships**: One user manages many projects; one project has many experiments
- **N:M relationships**: Model versions ↔ Dataset versions (training data reuse)

Assumptions:

- The "0..N" cardinalities suggest optional relationships (experiments without runs, models without deployments)
- The "1..1" cardinalities enforce mandatory ownership and hierarchy
- Many-to-many relationships enable flexible experimentation and reproducibility

Data Governance Assumptions

- All entities track creation timestamps for audit purposes
- User IDs establish accountability across all entities
- The system supports MLOps lifecycle: Project→ experimentation → model development → deployment
- Version control is central to both models and datasets
- The schema supports reproducibility through run tracking and artifact storage

Logical Model (Relational Model)

The relational schema below has been mapped from the provided EER diagram. The following conventions are used:

- **Primary Keys** are underlined.
- **Foreign Keys** are listed separately, along with their null constraints.

1. User (user_id, FirstName, LastName, email, created_at)

2. Administrator (admin_id)
    - Foreign Keys
        1. admin_id: references user_id from User relation, NOT NULL

3. Contributor (contributor_id)
    - Foreign Keys
        1. contributor id: references user_id from User relation, NOT NULL

4. Project_Lead (lead_id, department_name)
    - Foreign Keys
        1. lead_id: references user_id from User relation, NOT NULL

5. Project (project_id, name, description, created_by_user_id, created_at, is_active, lead_id, admin_id)
    - Foreign Keys:
        1. created_by_user_id : references user_id from user relation, NOT NULL
        2. lead_id : references lead_id from Project Lead relation, NOT NULL
        3. admin_id : references admin_id from Administrator relation, NOT NULL

6. Contributor_Project (contributor_id, project_id, role)
    - Foreign Keys:
        1. contributor_id : references contributor_id from Contributor relation, NOT NULL
        2. project_id : references project_id from Project relation, NOT NULL

7. Experiment (experiment_id, name, description, project_id)
    - Foreign Keys:

1. project_id : references project_id from Project relation, NOT NULL

8. Model_Registry (model_id, name, description, created_by)
- Foreign Keys:
    1. created_by: references lead_id from Project_Lead relation, NOT NULL

9. Model_Version (model_version_id, version_number, created_at, model_id, stage)
- Foreign Keys:
    1. model_id : references model_id from Model_Registry relation, NOT NULL

10. Deployment_History (deployment_id, model_version_id, deployment_date, environment)
- Foreign Keys:
    1. model_version_id: references model_version_id from Model_Version relation, NOT NULL

11. Experiment_Run (run_id, experiment_id, start_time, end_time, dataset_id, dataset_version_id, git_commit_hash, model_version_id)
- Foreign Keys:
    1. experiment_id : references experiment_id from Experiment relation, NOT NULL
    2. dataset_id: references dataset_id from Dataset relation, NOT NULL
    3. dataset_version_id : references dataset_version_id from Dataset_Version relation, NOT NULL
    4. model_version_id: references model_version_id from Model_Version relation, NOT NULL

12. Dataset (dataset_id, name, description, created_at, user_id)
- Foreign Keys:
    1. user_id: references user_id from User relation, NOT NULL

13. Dataset_Version (dataset_version_id, version_number, dataset_id, created_date, user_id, source_path)
- Foreign Keys:
    1. dataset_id: references dataset_id from Dataset relation, NOT NULL
    2. user_id: references user_id from User relation, NOT NULL

14. Artifacts (artifact_id, run_id, name, file_type, path_to_file)
- Foreign Keys:

1. run_id: references run_id from Experiment_Run relation, NOT NULL

15. Data_Artifact (data_artifact_id)

- Foreign Keys:
    1. data_artifact_id : references artifact_id from Artifacts relation, NOT NULL

16. Plot_Artifact (plot_artifact_id)

- Foreign Keys:
    1. plot_artifact_id: references artifact_id from Artifacts relation, NOT NULL

17. File_Artifact (file_artifact_id)

- Foreign Keys:
    1. file_artifact_id: references artifact_id from Artifacts relation, NOT NULL

MySQL Implementation

The **mlops** database was set up as the first step in the relational implementation. Each entity has been created as a table. All foreign key relationships and cardinality constraints were thoroughly validated by populating the system with synthetic data that was directly drawn from the project's business rules with the objective to simulate a production environment.

Table Creation Queries

```
CREATE database mlops;
USE  mlops;
```

1. **User Table**

```
CREATE TABLE User_Table (
user_id INT AUTO_INCREMENT PRIMARY KEY,
FirstName VARCHAR(100) NOT NULL,
LastName VARCHAR(100) NOT NULL,
Gender VARCHAR(10) NOT NULL,
email VARCHAR(255) UNIQUE NOT NULL,
created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

2. **Administrator Table**

```
CREATE TABLE Administrator (
admin_id INTEGER PRIMARY KEY,
FOREIGN KEY (admin_id) REFERENCES User_Table(user_id) ON DELETE CASCADE
);
```

3. **Contributor Table**

```
CREATE TABLE Contributor (
contributor_id INTEGER PRIMARY KEY,
FOREIGN KEY (contributor_id) REFERENCES User_Table(user_id) ON DELETE
CASCADE
);
```

4. **Project_Lead Table**

```
CREATE TABLE Project_Lead (
 lead_id INTEGER PRIMARY KEY,
 department_name VARCHAR(100),
 FOREIGN KEY (lead_id) REFERENCES User_Table(user_id) ON DELETE CASCADE
);
```

5. **Project Table**

```
CREATE TABLE Project (
   project_id INT AUTO_INCREMENT PRIMARY KEY,
   name VARCHAR(255) NOT NULL,
   description TEXT,
   created_by_user_id INT NOT NULL,
   created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
   is_active BOOLEAN DEFAULT TRUE,
   lead_id INT NOT NULL,
   admin_id INT NOT NULL,
   CONSTRAINT fk_project_created_by
         FOREIGN KEY (created_by_user_id) REFERENCES User_Table(user_id) ON
DELETE RESTRICT,
   CONSTRAINT fk_project_lead
         FOREIGN KEY (lead_id) REFERENCES Project_Lead(lead_id) ON DELETE
RESTRICT,
   CONSTRAINT fk_project_admin
       FOREIGN KEY (admin_id) REFERENCES Administrator(admin_id) ON DELETE
RESTRICT
);
```

6. Contributor_Project Table

```
CREATE TABLE Contributor_Project (
   contributor_id INT NOT NULL,
   project_id INT NOT NULL,
   role VARCHAR(100),
   PRIMARY KEY (contributor_id, project_id),
   CONSTRAINT fk_contributor_project_contributor
 FOREIGN KEY (contributor_id) REFERENCES Contributor(contributor_id) ON DELETE
CASCADE,
   CONSTRAINT fk_contributor_project_project
         FOREIGN KEY (project_id) REFERENCES Project(project_id) ON DELETE
CASCADE
);
```

7. Experiment Table

```
CREATE TABLE Experiment (
experiment_id INT AUTO_INCREMENT PRIMARY KEY,
name VARCHAR(255) NOT NULL,
description TEXT,
project_id INT NOT NULL,
FOREIGN KEY (project_id) REFERENCES Project(project_id) ON DELETE CASCADE
);
```

8. **Model_Registry Table**

```
CREATE TABLE Model_Registry (
 model_id INT AUTO_INCREMENT PRIMARY KEY,
 name VARCHAR(255) NOT NULL,
 description TEXT,
 created_by INT NOT NULL,
 CONSTRAINT fk_model_registry_lead
FOREIGN KEY (created_by) REFERENCES Project_Lead(lead_id) ON DELETE RESTRICT
);
```

9. **Model_Version Table**

```
CREATE TABLE Model_Version (
 model_version_id INT AUTO_INCREMENT PRIMARY KEY,
 version_number VARCHAR(50) NOT NULL,
 created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
 model_id INT NOT NULL,
 stage VARCHAR(50),
 CONSTRAINT fk_model_version_registry
FOREIGN KEY (model_id) REFERENCES Model_Registry(model_id) ON DELETE CASCADE,
   UNIQUE KEY unique_model_version (model_id, version_number)
);
```

10. **Deployment_History Table**

```
CREATE TABLE Deployment_History (
 deployment_id SERIAL PRIMARY KEY,
 model_version_id INTEGER NOT NULL,
 deployment_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
 environment VARCHAR(100),
 FOREIGN KEY (model_version_id) REFERENCES Model_Version(model_version_id) ON DELETE RESTRICT
);
```

11. **Dataset Table**

```
CREATE TABLE Dataset (
 dataset_id INT AUTO_INCREMENT PRIMARY KEY,
 name VARCHAR(255) NOT NULL,
 description TEXT,
 created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
 user_id INT NOT NULL,
 FOREIGN KEY (user_id) REFERENCES User_Table(user_id) ON DELETE RESTRICT
```

```
);
```

12. Dataset_Version Table

```
CREATE TABLE Dataset_Version (
   dataset_version_id INT AUTO_INCREMENT PRIMARY KEY,
   version_number VARCHAR(50) NOT NULL,
   dataset_id INT NOT NULL,
   created_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
   user_id INT NOT NULL,
   source_path VARCHAR(500),
      FOREIGN KEY (dataset_id) REFERENCES Dataset(dataset_id) ON DELETE
CASCADE,
       FOREIGN KEY (user_id) REFERENCES User_Table(user_id) ON DELETE
RESTRICT,
   UNIQUE KEY unique_dataset_version (dataset_id, version_number)
);
```

13. Experiment_Run Table

```
CREATE TABLE Experiment_Run (
   run_id INT AUTO_INCREMENT PRIMARY KEY,
   experiment_id INT NOT NULL,
   start_time TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
   end_time TIMESTAMP NULL,
   dataset_id INT NOT NULL,
   dataset_version_id INT NOT NULL,
   git_commit_hash VARCHAR(255),
   model_version_id INT NOT NULL,
      FOREIGN KEY (experiment_id) REFERENCES Experiment(experiment_id) ON
DELETE CASCADE,
       FOREIGN KEY (dataset_id) REFERENCES Dataset(dataset_id) ON DELETE
RESTRICT,
                    FOREIGN KEY (dataset_version_id) REFERENCES
Dataset_Version(dataset_version_id) ON DELETE RESTRICT,
   FOREIGN KEY (model_version_id) REFERENCES Model_Version(model_version_id)
ON DELETE RESTRICT
);
```

14. Artifacts Table

```
CREATE TABLE Artifacts (
   artifact_id INT AUTO_INCREMENT PRIMARY KEY,
   run_id INT NOT NULL,
   name VARCHAR(255) NOT NULL,
   file_type VARCHAR(50),
   path_to_file VARCHAR(500),
```

```
    FOREIGN KEY (run_id) REFERENCES Experiment_Run(run_id) ON DELETE
CASCADE
);
```

15. Data_Artifact Table

```
CREATE TABLE Data_Artifact (
   data_artifact_id INT PRIMARY KEY,
     FOREIGN KEY (data_artifact_id) REFERENCES Artifacts(artifact_id) ON DELETE
CASCADE
);
```

16. Plot_Artifact Table

```
CREATE TABLE Plot_Artifact (
   plot_artifact_id INT PRIMARY KEY,
     FOREIGN KEY (plot_artifact_id) REFERENCES Artifacts(artifact_id) ON DELETE
CASCADE
);
```

17. File_Artifact Table

```
CREATE TABLE File_Artifact (
   file_artifact_id INT PRIMARY KEY,
     FOREIGN KEY (file_artifact_id) REFERENCES Artifacts(artifact_id) ON DELETE
CASCADE
) ;
```

Indexing

```
CREATE INDEX idx_project_lead ON Project(lead_id);
CREATE INDEX idx_project_admin ON Project(admin_id);
CREATE INDEX idx_experiment_project ON Experiment(project_id);
CREATE INDEX idx_experiment_run_experiment ON Experiment_Run(experiment_id);
CREATE INDEX idx_artifacts_run ON Artifacts(run_id);
CREATE INDEX idx_model_version_model ON Model_Version(model_id);
CREATE INDEX idx_dataset_version_dataset ON Dataset_Version(dataset_id);
CREATE INDEX idx_user_email ON User_Table(email);
```

Inserting Data into the tables

1. INSERT INTO User_Table (user_id, FirstName, LastName, email, gender, created_at) VALUES
   (1, 'Alice', 'Johnson', 'alice.johnson@company.com', 'Female', '2024-01-15 09:00:00'),
   (2, 'Bob', 'Smith', 'bob.smith@company.com', 'Male', '2024-01-16 10:30:00'),
   (3, 'Carol', 'Davis', 'carol.davis@company.com', 'Female', '2024-01-17 11:00:00'),
   (4, 'David', 'Wilson', 'david.wilson@company.com', 'Male', '2024-01-18 09:30:00'),
   (5, 'Emma', 'Brown', 'emma.brown@company.com', 'Female', '2024-01-19 10:00:00'),
   (6, 'Frank', 'Miller', 'frank.miller@company.com', 'Male', '2024-01-20 11:30:00'),
   (7, 'Grace', 'Taylor', 'grace.taylor@company.com', 'Female', '2024-01-21 09:00:00'),
   (8, 'Henry', 'Anderson', 'henry.anderson@company.com', 'Male', '2024-01-22 10:30:00'),
   (9, 'Iris', 'Martinez', 'iris.martinez@company.com', 'Female', '2024-01-23 11:00:00'),
   (10, 'Jack', 'Garcia', 'jack.garcia@company.com', 'Male', '2024-01-24 09:30:00'),
   (11, 'Kelly', 'Rodriguez', 'kelly.rodriguez@company.com', 'Female', '2024-01-25 10:00:00'),
   (12, 'Liam', 'Hernandez', 'liam.hernandez@company.com', 'Male', '2024-01-26 11:00:00'),
   (13, 'Maya', 'Lopez', 'maya.lopez@company.com', 'Female', '2024-01-27 09:30:00'),
   (14, 'Nathan', 'Gonzalez', 'nathan.gonzalez@company.com', 'Male', '2024-01-28 10:30:00'),
   (15, 'Olivia', 'Perez', 'olivia.perez@company.com', 'Female', '2024-01-29 11:30:00'),
   (16, 'Peter', 'Sanchez', 'peter.sanchez@company.com', 'Male', '2024-01-30 09:00:00'),
   (17, 'Quinn', 'Ramirez', 'quinn.ramirez@company.com', 'Non-Binary', '2024-01-31 10:00:00'),
   (18, 'Rachel', 'Torres', 'rachel.torres@company.com', 'Female', '2024-02-01 11:00:00'),
   (19, 'Samuel', 'Flores', 'samuel.flores@company.com', 'Male', '2024-02-02 09:30:00'),
   (20, 'Tina', 'Rivera', 'tina.rivera@company.com', 'Female', '2024-02-03 10:30:00'),
   (21, 'Uma', 'Cooper', 'uma.cooper@company.com', 'Female', '2024-02-04 09:00:00'),
   (22, 'Victor', 'Reed', 'victor.reed@company.com', 'Male', '2024-02-05 10:30:00'),
   (23, 'Wendy', 'Bailey', 'wendy.bailey@company.com', 'Female', '2024-02-06 11:00:00'),
   (24, 'Xavier', 'Morgan', 'xavier.morgan@company.com', 'Male', '2024-02-07 09:30:00'),
   (25, 'Yara', 'Collins', 'yara.collins@company.com', 'Female', '2024-02-08 10:00:00'),
   (26, 'Zack', 'Stewart', 'zack.stewart@company.com', 'Male', '2024-02-09 11:30:00'),
   (27, 'Aria', 'Morris', 'aria.morris@company.com', 'Female', '2024-02-10 09:00:00'),
   (28, 'Blake', 'Murphy', 'blake.murphy@company.com', 'Non-Binary', '2024-02-11 10:30:00'),
   (29, 'Chloe', 'Cook', 'chloe.cook@company.com', 'Female', '2024-02-12 11:00:00'),
   (30, 'Dylan', 'Rogers', 'dylan.rogers@company.com', 'Male', '2024-02-13 09:30:00');

2. INSERT INTO Administrator (admin_id) VALUES
   (1), (2), (3);

3. INSERT INTO Project_Lead (lead_id, department_name) VALUES
   (4, 'Machine Learning'),
   (5, 'Data Science'),
   (6, 'AI Research'),
   (7, 'Computer Vision');

4. INSERT INTO Contributor (contributor_id) VALUES
   (8), (9), (10), (11), (12), (13), (14),
   (15), (16), (17), (18), (19), (20), (21),
   (22), (23), (24), (25), (26), (27), (28), (29), (30);

5. INSERT INTO Project (project_id, name, description, created_by_user_id, created_at, is_active, lead_id, admin_id) VALUES
   (1, 'Customer Churn Prediction', 'ML model to predict customer churn using historical transaction and engagement data.', 4, '2024-02-01 10:00:00', TRUE, 4, 1),
   (2, 'Image Classification System', 'Deep learning model for product image classification across 50+ categories.', 5, '2024-02-02 11:00:00', TRUE, 5, 1),
   (3, 'Sentiment Analysis Pipeline', 'NLP pipeline for real-time sentiment analysis of customer reviews.', 6, '2024-02-03 09:30:00', TRUE, 6, 2),
   (4, 'Fraud Detection System', 'Real-time fraud detection for payment transactions using ensemble methods.', 7, '2024-02-04 10:30:00', TRUE, 7, 2),
   (5, 'Recommendation Engine', 'Collaborative filtering recommendation system for personalized suggestions.', 4, '2024-02-05 11:30:00', TRUE, 4, 3),
   (6, 'Price Optimization Model', 'Dynamic pricing model that optimizes prices based on demand and competition.', 5, '2024-02-06 10:00:00', TRUE, 5, 3),
   (7, 'Object Detection Pipeline', 'Real-time object detection system for warehouse automation.', 6, '2024-02-07 11:00:00', TRUE, 6, 1),
   (8, 'Speech Recognition System', 'Deep learning based speech-to-text system supporting multiple languages.', 7, '2024-02-08 09:30:00', TRUE, 7, 1),
   (9, 'Demand Forecasting', 'Time series forecasting model for predicting product demand.', 4, '2024-02-09 10:30:00', TRUE, 4, 2),
   (10, 'Credit Risk Assessment', 'ML model for assessing credit risk and loan default probability.', 5, '2024-02-10 11:30:00', TRUE, 5, 2),
   (11, 'Medical Image Analysis', 'Deep learning model for analyzing medical images and detecting abnormalities.', 6, '2024-02-11 10:00:00', TRUE, 6, 3),
   (12, 'Chatbot Intent Classification', 'NLP model for classifying user intents in customer service chatbot.', 7, '2024-02-12 11:00:00', TRUE, 7, 3),
   (13, 'Anomaly Detection System', 'Unsupervised learning system for detecting anomalies in network traffic.', 4, '2024-02-13 09:30:00', TRUE, 4, 1),
   (14, 'User Segmentation', 'Clustering model for segmenting users based on behavior

patterns.', 5, '2024-02-14 10:30:00', TRUE, 5, 1),
(15, 'Ad Click Prediction', 'Model predicting probability of ad clicks based on user features.', 6, '2024-02-15 11:30:00', TRUE, 6, 2),
(16, 'Supply Chain Optimization', 'Optimization model for supply chain logistics and inventory.', 7, '2024-02-16 10:00:00', TRUE, 7, 2),
(17, 'Video Content Moderation', 'Deep learning for automatically moderating video content.', 4, '2024-02-17 11:00:00', TRUE, 4, 3),
(18, 'Document Classification', 'NLP model for automatically classifying documents.', 5, '2024-02-18 09:30:00', TRUE, 5, 3),
(19, 'Energy Prediction', 'Time series model predicting energy consumption patterns.', 6, '2024-02-19 10:30:00', TRUE, 6, 1),
(20, 'Customer Lifetime Value', 'Predictive model for estimating customer lifetime value.', 7, '2024-02-20 11:30:00', TRUE, 7, 1),
(21, 'Facial Recognition', 'Deep learning for facial recognition and verification.', 4, '2024-02-21 10:00:00', TRUE, 4, 2),
(22, 'Text Summarization', 'NLP model for automatic text summarization.', 5, '2024-02-22 11:00:00', TRUE, 5, 2),
(23, 'Network Intrusion Detection', 'ML model for detecting network security threats.', 6, '2024-02-23 09:30:00', TRUE, 6, 3),
(24, 'Sales Forecasting', 'Predictive model for forecasting sales trends.', 7, '2024-02-24 10:30:00', TRUE, 7, 3),
(25, 'Quality Control Vision', 'Computer vision for automated quality inspection.', 4, '2024-02-25 11:30:00', TRUE, 4, 1);

6. INSERT INTO Contributor_Project (contributor_id, project_id, role) VALUES
(8, 1, 'Data Engineer'), (9, 1, 'ML Engineer'),
(10, 2, 'ML Engineer'), (11, 2, 'Data Scientist'),
(12, 3, 'NLP Specialist'), (13, 3, 'Data Engineer'),
(14, 4, 'ML Engineer'), (15, 4, 'Data Scientist'),
(16, 5, 'ML Engineer'), (17, 5, 'Data Scientist'),
(18, 6, 'Data Scientist'), (19, 6, 'ML Engineer'),
(20, 7, 'Computer Vision Engineer'), (8, 7, 'ML Engineer'),
(9, 8, 'NLP Specialist'), (10, 9, 'Data Scientist'),
(11, 10, 'ML Engineer'), (12, 11, 'Computer Vision Engineer'),
(13, 12, 'NLP Specialist'), (14, 13, 'ML Engineer'),
(21, 14, 'Data Scientist'), (22, 15, 'ML Engineer'),
(23, 16, 'Data Engineer'), (24, 17, 'Computer Vision Engineer'),
(25, 18, 'NLP Specialist'), (26, 19, 'Data Scientist'),
(27, 20, 'ML Engineer'), (28, 21, 'Computer Vision Engineer'),
(29, 22, 'NLP Specialist'), (30, 23, 'ML Engineer')

7. INSERT INTO Experiment (experiment_id, name, description, project_id) VALUES
(1, 'Baseline XGBoost Model', 'Initial baseline using XGBoost with default parameters.', 1),
(2, 'Feature Engineering v1', 'Testing engineered features including customer lifetime value.', 1),
(3, 'ResNet50 Transfer Learning', 'Fine-tuning ResNet50 pretrained on ImageNet.', 2),
(4, 'EfficientNet Comparison', 'Comparing EfficientNet variants for accuracy vs speed.', 2),
(5, 'BERT Base Model', 'Fine-tuning BERT base model for sentiment classification.', 3),
(6, 'RoBERTa Fine-tuning', 'Testing RoBERTa model for improved sentiment analysis.', 3),
(7, 'Ensemble Fraud Detection', 'Testing ensemble of Random Forest, XGBoost, and LightGBM.', 4),
(8, 'Neural Network Approach', 'Deep neural network with attention mechanism.', 4),
(9, 'Matrix Factorization', 'Implementing matrix factorization with implicit feedback.', 5),
(10, 'Deep Learning Recommender', 'Neural collaborative filtering using deep learning.', 5),
(11, 'Elasticity Modeling', 'Testing price elasticity models to optimize pricing.', 6),
(12, 'YOLOv5 Detection', 'Implementing YOLOv5 for real-time object detection.', 7),
(13, 'Wav2Vec2 Model', 'Fine-tuning Wav2Vec2 for speech recognition.', 8),
(14, 'LSTM Forecasting', 'Long Short-Term Memory network for demand forecasting.', 9),
(15, 'Gradient Boosting Credit', 'XGBoost model with SMOTE for handling imbalanced data.', 10),
(16, 'U-Net Segmentation', 'U-Net architecture for medical image segmentation.', 11),
(17, 'Intent Classification BERT', 'BERT-based model for chatbot intent classification.', 12),
(18, 'Isolation Forest Anomaly', 'Isolation Forest algorithm for anomaly detection.', 13),
(19, 'K-Means Clustering', 'K-means clustering for user segmentation.', 14),
(20, 'Neural Network CTR', 'Deep neural network for click-through rate prediction.', 15),
(21, 'Transformer Architecture', 'Testing transformer model for improved accuracy.', 16),
(22, '3D CNN Video Analysis', 'Using 3D convolutional networks for video moderation.', 17),
(23, 'DistilBERT Classification', 'Lightweight BERT model for document classification.', 18),
(24, 'Prophet Forecasting', 'Facebook Prophet for energy consumption prediction.', 19),
(25, 'RFM Analysis Model', 'Recency, Frequency, Monetary value model for CLV.', 20),
(26, 'FaceNet Embedding', 'Deep metric learning for facial recognition.', 21),
(27, 'T5 Summarization', 'Text-to-Text Transfer Transformer for summarization.', 22),
(28, 'Autoencoder Anomaly', 'Autoencoder for network intrusion detection.', 23),
(29, 'ARIMA Sales Model', 'ARIMA time series model for sales forecasting.', 24),
(30, 'Mask R-CNN Inspection', 'Instance segmentation for quality control.', 25);

8. INSERT INTO Model_Registry (model_id, name, description, created_by) VALUES
   (1, 'ChurnPredictor', 'Customer churn prediction model using gradient boosting', 4),
   (2, 'ProductClassifier', 'CNN-based product image classifier', 5),
   (3, 'SentimentAnalyzer', 'BERT-based sentiment analysis model', 6),
   (4, 'FraudDetector', 'Ensemble model for payment fraud detection', 7),
   (5, 'ProductRecommender', 'Collaborative filtering recommendation model', 4),
   (6, 'PriceOptimizer', 'Dynamic pricing optimization model', 5),
   (7, 'ObjectDetector', 'Real-time object detection model', 6),
   (8, 'SpeechRecognizer', 'Multi-language speech recognition model', 7),
   (9, 'DemandForecaster', 'Time series demand forecasting model', 4),
   (10, 'CreditScorer', 'Credit risk assessment model', 5),
   (11, 'MedicalImageAnalyzer', 'Medical image analysis and diagnosis model', 6),
   (12, 'IntentClassifier', 'Chatbot intent classification model', 7),
   (13, 'AnomalyDetector', 'Network anomaly detection model', 4),
   (14, 'UserSegmenter', 'Customer segmentation clustering model', 5),
   (15, 'AdClickPredictor', 'Advertisement click prediction model', 6),
   (16, 'SupplyChainOptimizer', 'Supply chain logistics optimization', 7),
   (17, 'ContentModerator', 'Video content moderation model', 4),
   (18, 'DocumentClassifier', 'Document classification model', 5),
   (19, 'EnergyPredictor', 'Energy consumption forecasting', 6),
   (20, 'CLVPredictor', 'Customer lifetime value prediction', 7),
   (21, 'FaceRecognizer', 'Facial recognition model', 4),
   (22, 'TextSummarizer', 'Automatic text summarization', 5),
   (23, 'IntrusionDetector', 'Network intrusion detection', 6),
   (24, 'SalesForecaster', 'Sales forecasting model', 7),
   (25, 'QualityInspector', 'Quality control vision model', 4);

9. INSERT INTO Model_Version (model_version_id, version_number, created_at, model_id, stage) VALUES
   (1, 'v1.0.0', '2024-03-01 10:00:00', 1, 'Production'),
   (2, 'v1.1.0', '2024-03-15 11:00:00', 1, 'Staging'),
   (3, 'v1.0.0', '2024-03-05 09:00:00', 2, 'Production'),
   (4, 'v1.1.0', '2024-03-20 10:00:00', 2, 'Staging'),
   (5, 'v1.0.0', '2024-03-10 11:00:00', 3, 'Production'),
   (6, 'v1.1.0', '2024-03-25 10:00:00', 3, 'Staging'),
   (7, 'v1.0.0', '2024-03-12 10:00:00', 4, 'Production'),
   (8, 'v1.0.0', '2024-03-25 11:30:00', 5, 'Development'),
   (9, 'v1.0.0', '2024-03-15 09:30:00', 6, 'Production'),
   (10, 'v1.0.0', '2024-03-18 11:00:00', 7, 'Production'),
   (11, 'v1.0.0', '2024-03-20 10:00:00', 8, 'Production'),
   (12, 'v1.0.0', '2024-03-22 11:30:00', 9, 'Production'),
   (13, 'v1.0.0', '2024-03-24 09:00:00', 10, 'Production'),
   (14, 'v1.0.0', '2024-03-26 10:30:00', 11, 'Staging'),

(15, 'v1.0.0', '2024-03-28 11:00:00', 12, 'Production'),
(16, 'v1.0.0', '2024-03-30 09:30:00', 13, 'Production'),
(17, 'v1.0.0', '2024-04-01 10:00:00', 14, 'Production'),
(18, 'v1.0.0', '2024-04-03 09:00:00', 15, 'Production'),
(19, 'v1.2.0', '2024-04-05 10:30:00', 1, 'Development'),
(20, 'v1.2.0', '2024-04-07 11:00:00', 2, 'Development'),
(21, 'v1.0.0', '2024-04-08 10:00:00', 16, 'Staging'),
(22, 'v1.0.0', '2024-04-09 11:00:00', 17, 'Production'),
(23, 'v1.0.0', '2024-04-10 09:30:00', 18, 'Production'),
(24, 'v1.0.0', '2024-04-11 10:30:00', 19, 'Production'),
(25, 'v1.0.0', '2024-04-12 11:30:00', 20, 'Production'),
(26, 'v1.0.0', '2024-04-13 09:00:00', 21, 'Staging'),
(27, 'v1.0.0', '2024-04-14 10:00:00', 22, 'Production'),
(28, 'v1.0.0', '2024-04-15 11:00:00', 23, 'Production'),
(29, 'v1.0.0', '2024-04-16 09:30:00', 24, 'Production'),
(30, 'v1.0.0', '2024-04-17 10:30:00', 25, 'Staging');

10. INSERT INTO Deployment_History (deployment_id, model_version_id, deployment_date, environment) VALUES
(1, 1, '2024-03-02 10:00:00', 'Production-US-East'),
(2, 1, '2024-03-02 11:00:00', 'Production-EU-West'),
(3, 2, '2024-03-16 10:00:00', 'Staging'),
(4, 3, '2024-03-06 09:30:00', 'Production-US-East'),
(5, 4, '2024-03-21 10:30:00', 'Staging'),
(6, 5, '2024-03-11 11:00:00', 'Production-US-East'),
(7, 6, '2024-03-26 10:00:00', 'Staging'),
(8, 7, '2024-03-13 10:00:00', 'Production-US-East'),
(9, 9, '2024-03-16 09:30:00', 'Production-US-East'),
(10, 10, '2024-03-19 11:00:00', 'Production-US-East'),
(11, 11, '2024-03-21 10:00:00', 'Production-US-East'),
(12, 12, '2024-03-23 11:30:00', 'Production-US-East'),
(13, 13, '2024-03-25 09:00:00', 'Production-US-East'),
(14, 15, '2024-03-29 11:00:00', 'Production-US-East'),
(15, 16, '2024-03-31 09:30:00', 'Production-US-East'),
(16, 17, '2024-04-02 10:00:00', 'Production-US-East'),
(17, 18, '2024-04-04 09:00:00', 'Production-US-East'),
(18, 1, '2024-04-06 10:30:00', 'Production-Asia-Pacific'),
(19, 3, '2024-04-08 11:00:00', 'Production-EU-West'),
(20, 5, '2024-04-10 09:30:00', 'Production-Asia-Pacific'),
(21, 22, '2024-04-11 10:00:00', 'Production-US-East'),
(22, 23, '2024-04-12 11:00:00', 'Production-EU-West'),
(23, 24, '2024-04-13 09:30:00', 'Production-US-East'),
(24, 25, '2024-04-14 10:30:00', 'Production-Asia-Pacific'),

(25, 27, '2024-04-15 11:30:00', 'Production-US-East'),
(26, 28, '2024-04-16 09:00:00', 'Production-EU-West'),
(27, 29, '2024-04-17 10:00:00', 'Production-US-East'),
(28, 21, '2024-04-18 11:00:00', 'Staging'),
(29, 26, '2024-04-19 09:30:00', 'Staging'),
(30, 30, '2024-04-20 10:30:00', 'Staging');

11. INSERT INTO Dataset (dataset_id, name, description, created_at, user_id) VALUES
(1, 'Customer Transaction History', 'Complete transaction history for all customers from 2020-2024.', '2024-02-01 10:00:00', 8),
(2, 'Product Image Repository', 'High-resolution product images across 50+ categories. Total 500K images.', '2024-02-05 11:00:00', 10),
(3, 'Customer Review Dataset', 'Customer reviews and ratings collected from web and mobile platforms.', '2024-02-10 09:30:00', 12),
(4, 'Payment Transaction Logs', 'Real-time payment transaction data with fraud labels.', '2024-02-15 10:30:00', 14),
(5, 'User-Item Interaction Matrix', 'User interaction data including views, clicks, purchases.', '2024-02-20 11:30:00', 16),
(6, 'Historical Pricing Data', 'Historical pricing data across multiple product categories.', '2024-02-22 10:00:00', 18),
(7, 'Warehouse Video Footage', 'Labeled video footage from warehouse cameras.', '2024-02-24 11:00:00', 20),
(8, 'Speech Audio Recordings', 'Multi-language speech recordings with transcriptions.', '2024-02-26 09:30:00', 8),
(9, 'Sales Historical Data', 'Historical sales data across regions and products.', '2024-02-28 10:30:00', 10),
(10, 'Credit Application Data', 'Historical credit application data with approval outcomes.', '2024-03-01 11:30:00', 12),
(11, 'Medical Image Collection', 'Labeled medical images including X-rays and CT scans.', '2024-03-03 10:00:00', 14),
(12, 'Chatbot Conversation Logs', 'Historical chatbot conversations with labeled intents.', '2024-03-05 11:00:00', 16),
(13, 'Network Traffic Data', 'Network traffic logs with labeled normal and anomalous patterns.', '2024-03-07 09:30:00', 18),
(14, 'User Behavior Analytics', 'User behavioral data including clickstream and engagement metrics.', '2024-03-09 10:30:00', 20),
(15, 'Advertisement Campaign Data', 'Historical ad campaign performance data.', '2024-03-11 11:30:00', 8),
(16, 'Supply Chain Logistics', 'Supply chain and logistics historical data.', '2024-03-13 10:00:00', 10),
(17, 'Video Content Database', 'Video content with moderation labels.', '2024-03-15 11:00:00', 12),

(18, 'Document Repository', 'Corporate documents with classification labels.', '2024-03-17 09:30:00', 14),
(19, 'Energy Consumption Logs', 'Smart meter data with energy consumption patterns.', '2024-03-19 10:30:00', 16),
(20, 'Customer Lifecycle Data', 'Complete customer journey and lifecycle data.', '2024-03-21 11:30:00', 18),
(21, 'Facial Image Dataset', 'Facial images with identity labels for recognition.', '2024-03-23 10:00:00', 20),
(22, 'News Articles Collection', 'News articles for summarization training.', '2024-03-25 11:00:00', 8),
(23, 'Security Event Logs', 'Security event logs with intrusion labels.', '2024-03-27 09:30:00', 10),
(24, 'Regional Sales Data', 'Sales data segmented by region and category.', '2024-03-29 10:30:00', 12),
(25, 'Manufacturing Quality Data', 'Product quality inspection data with defect labels.', '2024-03-31 11:30:00', 14);

12. INSERT INTO Dataset_Version (dataset_version_id, version_number, dataset_id, created_date, user_id, source_path) VALUES
(1, 'v1.0', 1, '2024-02-01 10:00:00', 8, 's3://mlops-data/customer-transactions/v1.0/'),
(2, 'v1.1', 1, '2024-03-01 10:00:00', 8, 's3://mlops-data/customer-transactions/v1.1/'),
(3, 'v1.0', 2, '2024-02-05 11:00:00', 10, 's3://mlops-data/product-images/v1.0/'),
(4, 'v1.1', 2, '2024-03-10 11:00:00', 10, 's3://mlops-data/product-images/v1.1/'),
(5, 'v1.0', 3, '2024-02-10 09:30:00', 12, 's3://mlops-data/reviews/v1.0/'),
(6, 'v1.1', 3, '2024-03-15 09:30:00', 12, 's3://mlops-data/reviews/v1.1/'),
(7, 'v1.0', 4, '2024-02-15 10:30:00', 14, 's3://mlops-data/transactions/v1.0/'),
(8, 'v1.0', 5, '2024-02-20 11:30:00', 16, 's3://mlops-data/interactions/v1.0/'),
(9, 'v1.0', 6, '2024-02-22 10:00:00', 18, 's3://mlops-data/pricing/v1.0/'),
(10, 'v1.0', 7, '2024-02-24 11:00:00', 20, 's3://mlops-data/warehouse-video/v1.0/'),
(11, 'v1.0', 8, '2024-02-26 09:30:00', 8, 's3://mlops-data/speech-audio/v1.0/'),
(12, 'v1.0', 9, '2024-02-28 10:30:00', 10, 's3://mlops-data/sales-history/v1.0/'),
(13, 'v1.0', 10, '2024-03-01 11:30:00', 12, 's3://mlops-data/credit-apps/v1.0/'),
(14, 'v1.0', 11, '2024-03-03 10:00:00', 14, 's3://mlops-data/medical-images/v1.0/'),
(15, 'v1.0', 12, '2024-03-05 11:00:00', 16, 's3://mlops-data/chatbot-logs/v1.0/'),
(16, 'v1.0', 13, '2024-03-07 09:30:00', 18, 's3://mlops-data/network-traffic/v1.0/'),
(17, 'v1.0', 14, '2024-03-09 10:30:00', 20, 's3://mlops-data/user-behavior/v1.0/'),
(18, 'v1.0', 15, '2024-03-11 11:30:00', 8, 's3://mlops-data/ad-campaigns/v1.0/'),
(19, 'v2.0', 1, '2024-04-01 10:00:00', 8, 's3://mlops-data/customer-transactions/v2.0/'),
(20, 'v1.2', 2, '2024-04-15 11:00:00', 10, 's3://mlops-data/product-images/v1.2/'),
(21, 'v1.0', 16, '2024-03-13 10:00:00', 10, 's3://mlops-data/supply-chain/v1.0/'),
(22, 'v1.0', 17, '2024-03-15 11:00:00', 12, 's3://mlops-data/video-content/v1.0/'),
(23, 'v1.0', 18, '2024-03-17 09:30:00', 14, 's3://mlops-data/documents/v1.0/'),
(24, 'v1.0', 19, '2024-03-19 10:30:00', 16, 's3://mlops-data/energy-logs/v1.0/'),

(25, 'v1.0', 20, '2024-03-21 11:30:00', 18, 's3://mlops-data/customer-lifecycle/v1.0/'),
(26, 'v1.0', 21, '2024-03-23 10:00:00', 20, 's3://mlops-data/facial-images/v1.0/'),
(27, 'v1.0', 22, '2024-03-25 11:00:00', 8, 's3://mlops-data/news-articles/v1.0/'),
(28, 'v1.0', 23, '2024-03-27 09:30:00', 10, 's3://mlops-data/security-logs/v1.0/'),
(29, 'v1.0', 24, '2024-03-29 10:30:00', 12, 's3://mlops-data/regional-sales/v1.0/'),
(30, 'v1.0', 25, '2024-03-31 11:30:00', 14, 's3://mlops-data/quality-data/v1.0/');

13. INSERT INTO Experiment_Run (run_id, experiment_id, start_time, end_time, dataset_id, dataset_version_id, git_commit_hash, model_version_id) VALUES
(1, 1, '2024-03-01 09:00:00', '2024-03-01 11:30:00', 1, 1, 'a1b2c3d4e5f6g7h8i9j0', 1),
(2, 2, '2024-03-10 09:30:00', '2024-03-10 14:20:00', 1, 2, 'b2c3d4e5f6g7h8i9j0k1', 2),
(3, 3, '2024-03-05 08:00:00', '2024-03-05 16:30:00', 2, 3, 'c3d4e5f6g7h8i9j0k1l2', 3),
(4, 4, '2024-03-15 09:00:00', '2024-03-15 18:00:00', 2, 4, 'd4e5f6g7h8i9j0k1l2m3', 4),
(5, 5, '2024-03-10 10:00:00', '2024-03-10 15:30:00', 3, 5, 'e5f6g7h8i9j0k1l2m3n4', 5),
(6, 6, '2024-03-15 09:00:00', '2024-03-15 15:00:00', 3, 6, 'f6g7h8i9j0k1l2m3n4o5', 6),
(7, 7, '2024-03-12 09:30:00', '2024-03-12 13:45:00', 4, 7, 'g7h8i9j0k1l2m3n4o5p6', 7),
(8, 8, '2024-03-20 09:00:00', '2024-03-20 14:00:00', 4, 7, 'h8i9j0k1l2m3n4o5p6q7', 7),
(9, 9, '2024-03-25 10:00:00', '2024-03-25 17:30:00', 5, 8, 'i9j0k1l2m3n4o5p6q7r8', 8),
(10, 10, '2024-04-12 08:00:00', '2024-04-12 16:30:00', 5, 8, 'j0k1l2m3n4o5p6q7r8s9', 8),
(11, 11, '2024-03-15 10:00:00', '2024-03-15 14:30:00', 6, 9, 'k1l2m3n4o5p6q7r8s9t0', 9),
(12, 12, '2024-03-18 09:00:00', '2024-03-18 14:00:00', 7, 10, 'l2m3n4o5p6q7r8s9t0u1', 10),
(13, 13, '2024-03-20 09:00:00', '2024-03-20 15:30:00', 8, 11, 'm3n4o5p6q7r8s9t0u1v2', 11),
(14, 14, '2024-03-22 10:00:00', '2024-03-22 16:00:00', 9, 12, 'n4o5p6q7r8s9t0u1v2w3', 12),
(15, 15, '2024-03-24 10:00:00', '2024-03-24 14:30:00', 10, 13, 'o5p6q7r8s9t0u1v2w3x4', 13),
(16, 16, '2024-03-26 09:30:00', '2024-03-26 15:00:00', 11, 14, 'p6q7r8s9t0u1v2w3x4y5', 14),
(17, 17, '2024-03-28 10:00:00', '2024-03-28 15:30:00', 12, 15, 'q7r8s9t0u1v2w3x4y5z6', 15),
(18, 18, '2024-03-30 09:00:00', '2024-03-30 14:00:00', 13, 16, 'r8s9t0u1v2w3x4y5z6a7', 16),
(19, 19, '2024-04-01 10:00:00', '2024-04-01 15:30:00', 14, 17, 's9t0u1v2w3x4y5z6a7b8', 17),
(20, 20, '2024-04-03 10:00:00', '2024-04-03 15:00:00', 15, 18, 't0u1v2w3x4y5z6a7b8c9', 18),
(21, 21, '2024-04-05 09:00:00', '2024-04-05 14:30:00', 16, 21, 'u1v2w3x4y5z6a7b8c9d0', 21),
(22, 22, '2024-04-07 10:00:00', '2024-04-07 16:00:00', 17, 22, 'v2w3x4y5z6a7b8c9d0e1', 22),

(23, 23, '2024-04-09 09:30:00', '2024-04-09 15:00:00', 18, 23, 'w3x4y5z6a7b8c9d0e1f2', 23),
(24, 24, '2024-04-11 10:00:00', '2024-04-11 15:30:00', 19, 24, 'x4y5z6a7b8c9d0e1f2g3', 24),
(25, 25, '2024-04-13 09:00:00', '2024-04-13 14:30:00', 20, 25, 'y5z6a7b8c9d0e1f2g3h4', 25),
(26, 26, '2024-04-15 10:00:00', '2024-04-15 16:00:00', 21, 26, 'z6a7b8c9d0e1f2g3h4i5', 26),
(27, 27, '2024-04-17 09:30:00', '2024-04-17 15:00:00', 22, 27, 'a7b8c9d0e1f2g3h4i5j6', 27),
(28, 28, '2024-04-19 10:00:00', '2024-04-19 15:30:00', 23, 28, 'b8c9d0e1f2g3h4i5j6k7', 28),
(29, 29, '2024-04-21 09:00:00', '2024-04-21 14:30:00', 24, 29, 'c9d0e1f2g3h4i5j6k7l8', 29),
(30, 30, '2024-04-23 10:00:00', '2024-04-23 16:00:00', 25, 30, 'd0e1f2g3h4i5j6k7l8m9', 30);

14. INSERT INTO Artifacts (artifact_id, run_id, name, file_type, path_to_file) VALUES
(1, 1, 'training_metrics', 'json', 's3://mlops-artifacts/run-1/metrics.json'),
(2, 1, 'confusion_matrix', 'png', 's3://mlops-artifacts/run-1/confusion_matrix.png'),
(3, 2, 'training_metrics', 'json', 's3://mlops-artifacts/run-2/metrics.json'),
(4, 2, 'roc_curve', 'png', 's3://mlops-artifacts/run-2/roc_curve.png'),
(5, 3, 'training_metrics', 'json', 's3://mlops-artifacts/run-3/metrics.json'),
(6, 3, 'learning_curves', 'png', 's3://mlops-artifacts/run-3/learning_curves.png'),
(7, 4, 'training_metrics', 'json', 's3://mlops-artifacts/run-4/metrics.json'),
(8, 4, 'sample_predictions', 'png', 's3://mlops-artifacts/run-4/predictions.png'),
(9, 5, 'training_metrics', 'json', 's3://mlops-artifacts/run-5/metrics.json'),
(10, 5, 'word_cloud', 'png', 's3://mlops-artifacts/run-5/word_cloud.png'),
(11, 6, 'training_metrics', 'json', 's3://mlops-artifacts/run-6/metrics.json'),
(12, 6, 'sentiment_distribution', 'png', 's3://mlops-artifacts/run-6/distribution.png'),
(13, 7, 'training_metrics', 'json', 's3://mlops-artifacts/run-7/metrics.json'),
(14, 7, 'feature_importance', 'csv', 's3://mlops-artifacts/run-7/feature_importance.csv'),
(15, 8, 'training_metrics', 'json', 's3://mlops-artifacts/run-8/metrics.json'),
(16, 9, 'training_metrics', 'json', 's3://mlops-artifacts/run-9/metrics.json'),
(17, 10, 'training_metrics', 'json', 's3://mlops-artifacts/run-10/metrics.json'),
(18, 11, 'training_metrics', 'json', 's3://mlops-artifacts/run-11/metrics.json'),
(19, 12, 'training_metrics', 'json', 's3://mlops-artifacts/run-12/metrics.json'),
(20, 13, 'training_metrics', 'json', 's3://mlops-artifacts/run-13/metrics.json'),
(21, 14, 'training_metrics', 'json', 's3://mlops-artifacts/run-14/metrics.json'),
(22, 15, 'training_metrics', 'json', 's3://mlops-artifacts/run-15/metrics.json'),
(23, 16, 'training_log', 'txt', 's3://mlops-artifacts/run-16/training.log'),
(24, 17, 'model_architecture', 'pdf', 's3://mlops-artifacts/run-17/architecture.pdf'),

(25, 18, 'training_metrics', 'json', 's3://mlops-artifacts/run-18/metrics.json'),
(26, 19, 'precision_recall', 'png', 's3://mlops-artifacts/run-19/pr_curve.png'),
(27, 20, 'training_metrics', 'json', 's3://mlops-artifacts/run-20/metrics.json'),
(28, 21, 'training_metrics', 'json', 's3://mlops-artifacts/run-21/metrics.json'),
(29, 22, 'video_samples', 'mp4', 's3://mlops-artifacts/run-22/samples.mp4'),
(30, 23, 'embeddings_viz', 'png', 's3://mlops-artifacts/run-23/embeddings.png');

15. INSERT INTO Data_Artifact (data_artifact_id) VALUES
    (1), (3), (5), (7), (9), (11), (13), (15), (16), (17), (18), (19);

16. INSERT INTO Plot_Artifact (plot_artifact_id) VALUES
    (2), (4), (6), (8), (10), (12), (20), (21), (22), (25), (26), (27), (30);

17. INSERT INTO File_Artifact (file_artifact_id) VALUES
    (14), (23), (24), (28), (29);

Queries for MySQL DB

1. Retrieve the projects that have the most number of experiments.

```
SELECT
    p.name AS project_name,
    COUNT(e.experiment_id) AS number_of_experiments,
    p.is_active AS currently_active
FROM Project p
LEFT JOIN Experiment e ON p.project_id = e.project_id
GROUP BY p.project_id, p.name, p.is_active
ORDER BY number_of_experiments DESC
LIMIT 10;
```

project_name	number_of_experiments	currently_active
Customer Churn Prediction	2	1
Image Classification System	2	1
Sentiment Analysis Pipeline	2	1
Fraud Detection System	2	1
Recommendation Engine	2	1
Price Optimization Model	1	1
Object Detection Pipeline	1	1
Speech Recognition System	1	1
Demand Forecasting	1	1
Credit Risk Assessment	1	1

2. Retrieve the number of models present in stage.

```
SELECT
    stage,
    COUNT(*) AS number_of_models
FROM Model_Version
GROUP BY stage
ORDER BY stage;
```

stage	number_of_mod...
Development	3
Production	13
Staging	4

3. Retrieve the types of Artifacts that are generated the most.

```
SELECT
   file_type,
   COUNT(*) AS number_of_artifacts
FROM Artifacts
GROUP BY file_type
ORDER BY number_of_artifacts DESC;
```

file_type	number_of_artifacts
json	13
png	6
csv	1

4. Retrieve the number of deployments per model.

```
SELECT
   environment,
   COUNT(*) AS number_of_deployments
FROM Deployment_History
GROUP BY environment
ORDER BY number_of_deployments DESC;
```

environment	number_of_deployments
Production-US-East	13
Staging	3
Production-EU-West	2
Production-Asia-Pacific	2

5. Retrieve the number of people working on each project

```
SELECT
   p.name AS project_name,
   COUNT(cp.contributor_id) AS team_size
FROM Project p
LEFT JOIN Contributor_Project cp ON p.project_id = cp.project_id
```

```
GROUP BY p.project_id, p.name
ORDER BY team_size DESC;
```

project_name	team_size
Customer Churn Prediction	2
Image Classification System	2
Sentiment Analysis Pipeline	2
Fraud Detection System	2
Recommendation Engine	2
Price Optimization Model	2
Object Detection Pipeline	2
Speech Recognition System	1
Demand Forecasting	1
Credit Risk Assessment	1
Medical Image Analysis	1
Chatbot Intent Classification	1
Anomaly Detection System	1
User Segmentation	0
Ad Click Prediction	0

6. Retrieve the last updated date of each database

```
SELECT
    d.name AS dataset_name,
    MAX(dv.created_date) AS last_update
FROM Dataset d
JOIN Dataset_Version dv ON d.dataset_id = dv.dataset_id
GROUP BY d.dataset_id, d.name
ORDER BY last_update ASC
```

LIMIT 10;

dataset_name	last_update
Payment Transaction Logs	2024-02-15 10:30:00
User-Item Interaction Matrix	2024-02-20 11:30:00
Historical Pricing Data	2024-02-22 10:00:00
Warehouse Video Footage	2024-02-24 11:00:00
Speech Audio Recordings	2024-02-26 09:30:00
Sales Historical Data	2024-02-28 10:30:00
Credit Application Data	2024-03-01 11:30:00
Medical Image Collection	2024-03-03 10:00:00
Chatbot Conversation Logs	2024-03-05 11:00:00
Network Traffic Data	2024-03-07 09:30:00

7. Retrieve the projects that don't have any experiments.

```
SELECT
    p.name AS project_name,
    p.created_at
FROM Project p
WHERE NOT EXISTS (
    SELECT 1
    FROM Experiment e
    WHERE e.project_id = p.project_id
);
```

project_na...	created_at

There are no projects with zero experiments.

8. Retrieve the projects led by anyone in the 'Machine Learning' department.

```
SELECT
   p.name AS project_name,
   p.is_active
FROM Project p
WHERE p.lead_id = ANY (
   SELECT lead_id
   FROM Project_Lead
   WHERE department_name = 'Machine Learning'
   );
```

project_name	is_active
Customer Churn Prediction	1
Recommendation Engine	1
Demand Forecasting	1
Anomaly Detection System	1

9. Retrieve the datasets that exist but are NOT used in ANY experiment

```
SELECT
   d.name AS unused_dataset,
   d.created_at
FROM Dataset d
WHERE NOT EXISTS (
   SELECT 1
   FROM Dataset_Version dv
   JOIN Experiment_Run er ON dv.dataset_version_id = er.dataset_version_id
   WHERE dv.dataset_id = d.dataset_id
)
ORDER BY d.created_at DESC;
```

unused_dataset	created_at

All datasets are used in different experiments.

10. Retrieve the people working on projects with their roles.

```
SELECT
p.name AS project_name,
CONCAT(project_lead.FirstName, ' ', project_lead.LastName) AS project_lead,
CONCAT(contrib.FirstName, ' ', contrib.LastName) AS contributor_name,
cp.role AS contributor_role
FROM Project p
JOIN User_Table project_lead ON p.lead_id = project_lead.user_id
LEFT JOIN Contributor_Project cp ON p.project_id = cp.project_id
LEFT JOIN User_Table contrib ON cp.contributor_id = contrib.user_id
ORDER BY p.name, cp.role;
```

project_name	project_lead	contributor_name	contributor_role
Ad Click Prediction	Frank Miller	NULL	NULL
Anomaly Detection System	David Wilson	Nathan Gonzalez	ML Engineer
Chatbot Intent Classification	Grace Taylor	Maya Lopez	NLP Specialist
Credit Risk Assessment	Emma Brown	Kelly Rodriguez	ML Engineer
Customer Churn Prediction	David Wilson	Henry Anderson	Data Engineer
Customer Churn Prediction	David Wilson	Iris Martinez	ML Engineer
Demand Forecasting	David Wilson	Jack Garcia	Data Scientist
Fraud Detection System	Grace Taylor	Olivia Perez	Data Scientist
Fraud Detection System	Grace Taylor	Nathan Gonzalez	ML Engineer
Image Classification System	Emma Brown	Kelly Rodriguez	Data Scientist
Image Classification System	Emma Brown	Jack Garcia	ML Engineer
Medical Image Analysis	Frank Miller	Liam Hernandez	Computer Vision…
Object Detection Pipeline	Frank Miller	Tina Rivera	Computer Vision…
Object Detection Pipeline	Frank Miller	Henry Anderson	ML Engineer
Price Optimization Model	Emma Brown	Rachel Torres	Data Scientist
Price Optimization Model	Emma Brown	Samuel Flores	ML Engineer
Recommendation Engine	David Wilson	Quinn Ramirez	Data Scientist
Recommendation Engine	David Wilson	Peter Sanchez	ML Engineer
Sentiment Analysis Pipeline	Frank Miller	Maya Lopez	Data Engineer
Sentiment Analysis Pipeline	Frank Miller	Liam Hernandez	NLP Specialist
Speech Recognition System	Grace Taylor	Iris Martinez	NLP Specialist
User Segmentation	Emma Brown	NULL	NULL

Python Application Code

```
import pandas as pd
import mysql.connector
from mysql.connector import Error


try:
   connection = mysql.connector.connect(host='localhost',
                                        database='mlops',
                                        user='root',
                                        password='root1234',
                                        auth_plugin = 'mysql_native_password')
   if connection.is_connected():


       cursor = connection.cursor()
       cursor.execute("select database();")
       record = cursor.fetchone()
       print("Your connected to database: ", record)
       # Query 1
       sql_select_Query = """SELECT
                               file_type,
                               COUNT(*) AS number_of_artifacts
                               FROM Artifacts
                               GROUP BY file_type
                               ORDER BY number_of_artifacts DESC;"""


       df = pd.read_sql_query(sql_select_Query, connection)
       print(df)
       import matplotlib.pyplot as plt
```

```
        plt.figure(figsize=(10,6))
        plt.pie(df['number_of_artifacts'], labels=df['file_type'], autopct='%1.1f%%')
        plt.title('Number of artifacts by file type')
        plt.savefig('artifacts_piechart.png')


        # Query 2
        sql_select_Query = """SELECT
                              p.name AS project_name,
                              COUNT(e.experiment_id) AS number_of_experiments,
                              p.is_active AS currently_active
                              FROM Project p
                              LEFT JOIN Experiment e ON p.project_id = e.project_id
                              GROUP BY p.project_id, p.name, p.is_active
                              ORDER BY number_of_experiments DESC
                              LIMIT 10;"""


        df = pd.read_sql_query(sql_select_Query, connection)
        print(df)
        plt.figure(figsize=(10,6))
        bar_colors = ['#1f77b4' if active else '#ff7f0e' for active in
df['currently_active']]
        plt.bar(df['project_name'], df['number_of_experiments'], color=bar_colors)
        plt.xlabel('Project Name')
        plt.ylabel('Number of Experiments')
        plt.title('Top 10 Projects by Number of Experiments')
        plt.xticks(rotation=45, ha='right')
        legend_labels = [plt.Line2D([0], [0], color='#1f77b4', lw=4, label='Active'),
                         plt.Line2D([0], [0], color='#ff7f0e', lw=4, label='Inactive')]
```

```
        plt.legend(handles=legend_labels, title="Project Status")

        plt.tight_layout()

        plt.savefig('projects_barchart.png')









        # Query 3

        import seaborn as sns

        sql_select_Query = """SELECT

                              p.name AS Project_Name,

                              TIMESTAMPDIFF(MINUTE, er.start_time, er.end_time) AS
Duration_Minutes

                              FROM Experiment_Run er

                              JOIN Experiment e ON er.experiment_id = e.experiment_id

                              JOIN Project p ON e.project_id = p.project_id

                              WHERE er.end_time IS NOT NULL"""


        df = pd.read_sql_query(sql_select_Query, connection)

        print(df)

        plt.figure(figsize=(12, 6))

        sns.boxplot(

            data=df,

            x='Project_Name',

            y='Duration_Minutes',

            hue='Project_Name',

            palette="Set2",

            legend=False
```

```
)


plt.title('Distribution of Experiment Durations by Project')

plt.ylabel('Duration (Minutes)')

plt.xlabel('Project')

plt.xticks(rotation=45, ha='right')

plt.grid(axis='y', linestyle='--', alpha=0.7)


plt.tight_layout()


plt.savefig('experiment_duration.png')




# Query 4

sql_select_Query = """SELECT

                    DATE(deployment_date) as deploy_date,

                    environment,

                    COUNT(*) as daily_count

                    FROM Deployment_History

                    GROUP BY DATE(deployment_date), environment

                    ORDER BY deploy_date ASC"""

df = pd.read_sql_query(sql_select_Query, connection)

print(df)

df['deploy_date'] = pd.to_datetime(df['deploy_date'])


pivot_df = df.pivot_table(

    index='deploy_date',

    columns='environment',
```

```
        values='daily_count',
        aggfunc='sum'
    ).fillna(0)



    pivot_df = pivot_df.resample('D').sum()
    plt.figure(figsize=(12, 6))
    ax = pivot_df.plot(kind='area', stacked=True, alpha=0.6, figsize=(12, 6))
    plt.title('Deployment Frequency by Environment')
    plt.ylabel('Number of Deployments')
    plt.xlabel('Date')
    plt.grid(linestyle='--', alpha=0.5)
    plt.tight_layout()
    plt.savefig('deployment_frequency.png')


    # Query 5
    sql_select_Query = """SELECT
                        environment,
                        COUNT(*) AS number_of_deployments
                    FROM Deployment_History
                    GROUP BY environment
                    Having environment != 'Staging' ;"""
    df = pd.read_sql_query(sql_select_Query, connection)
    print(df)
    plt.figure(figsize=(10,6))
    plt.bar(df['environment'], df['number_of_deployments'])
    plt.title('Number of Deployments by Production Environment')
    plt.xlabel('Environment')
```

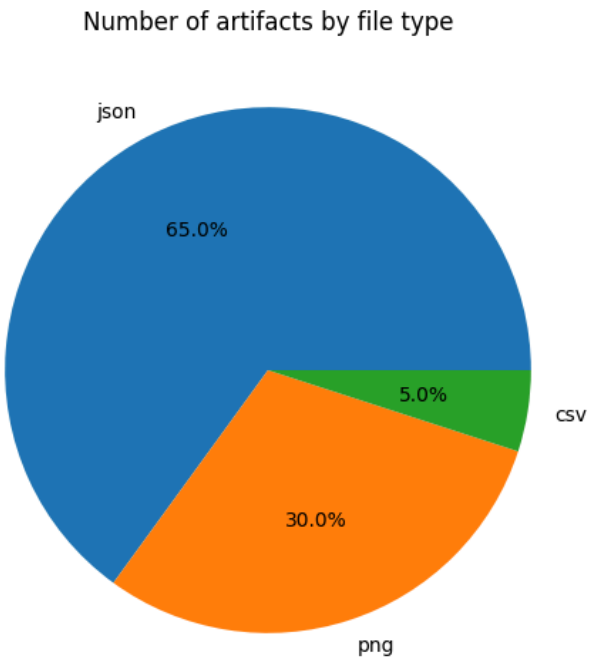
```
        plt.ylabel('Number of Deployments')



        plt.savefig('deployment_linechart.png')



except Error as e:

   print("Error while connecting to MySQL", e)

finally:

   if (connection.is_connected()):

        cursor.close()

        connection.close()

        print("MySQL connection is closed")
```

Plots Generated

The following plots were generated during the python implementation of the queries:

Query 1. Retrieve the number of artifacts for each type.

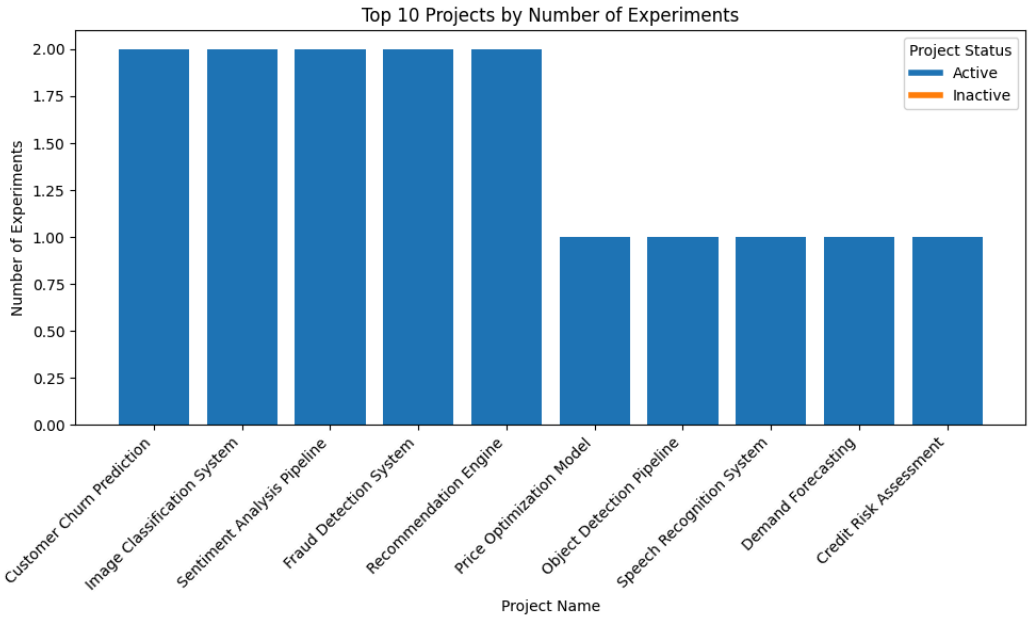
```
SELECT
 file_type,
 COUNT(*) AS number_of_artifacts
 FROM Artifacts
 GROUP BY file_type
 ORDER BY number_of_artifacts DESC;
```



Findings: This artifact distribution signifies mature machine learning operations practices, with 65% JSON artifacts being produced. PNG artifacts are 30% which shows a strong emphasis on visualization models produced for ROC curves and learning plots for communication with stakeholders. The minimal use of CSV artifacts show, that the teams avoid the redundant exports of data and storing only essential tabular data.

Query 2: Retrieve the projects that have the most number of experiments.

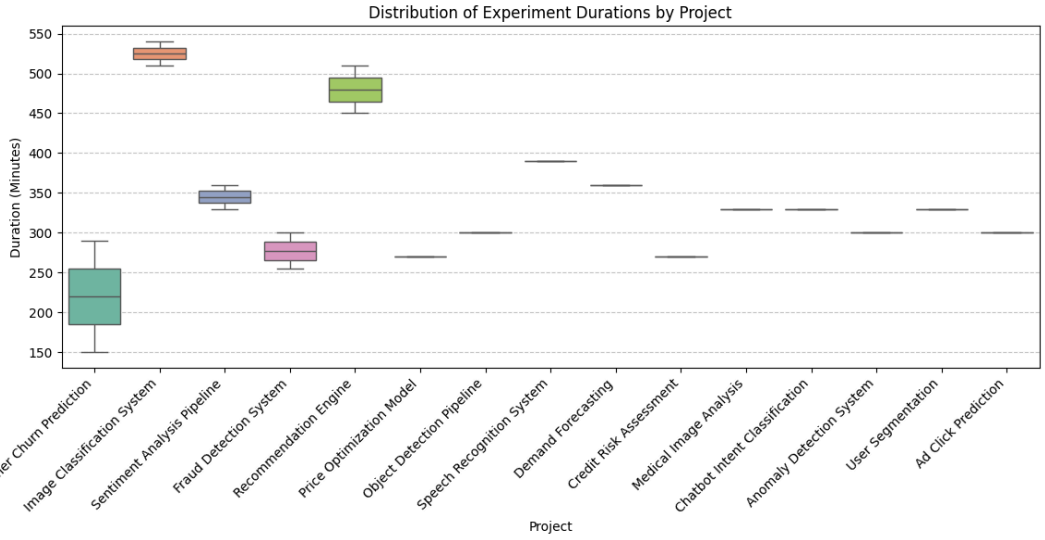
```
SELECT
p.name AS project_name,
COUNT(e.experiment_id) AS number_of_experiments,
p.is_active AS currently_active
FROM Project p
LEFT JOIN Experiment e ON p.project_id = e.project_id
GROUP BY p.project_id, p.name, p.is_active
ORDER BY number_of_experiments DESC
LIMIT 10;
```



Findings: This bar chart shows that projects like Customer Churn Prediction, Image Classification System, Sentiment Analysis Pipeline, Fraud Detection System, and Recommendation Engine are the strongest initiatives and have high intensity research work. The remaining projects with less experiments depict that they are probably new and in the initial phase or could be niche projects, hence the less number of experiments.

Query 3. Retrieve the duration of the experiment run for different projects.

```
SELECT
p.name AS Project_Name,
TIMESTAMPDIFF(MINUTE, er.start_time, er.end_time) AS Duration_Minutes
FROM Experiment_Run er
JOIN Experiment e ON er.experiment_id = e.experiment_id
JOIN Project p ON e.project_id = p.project_id
WHERE er.end_time IS NOT NULL;
```



Findings: This box plot represents that computer vision projects and recommendation systems have two-to-three times more compute time than traditional ML approaches like Customer Churn Prediction.

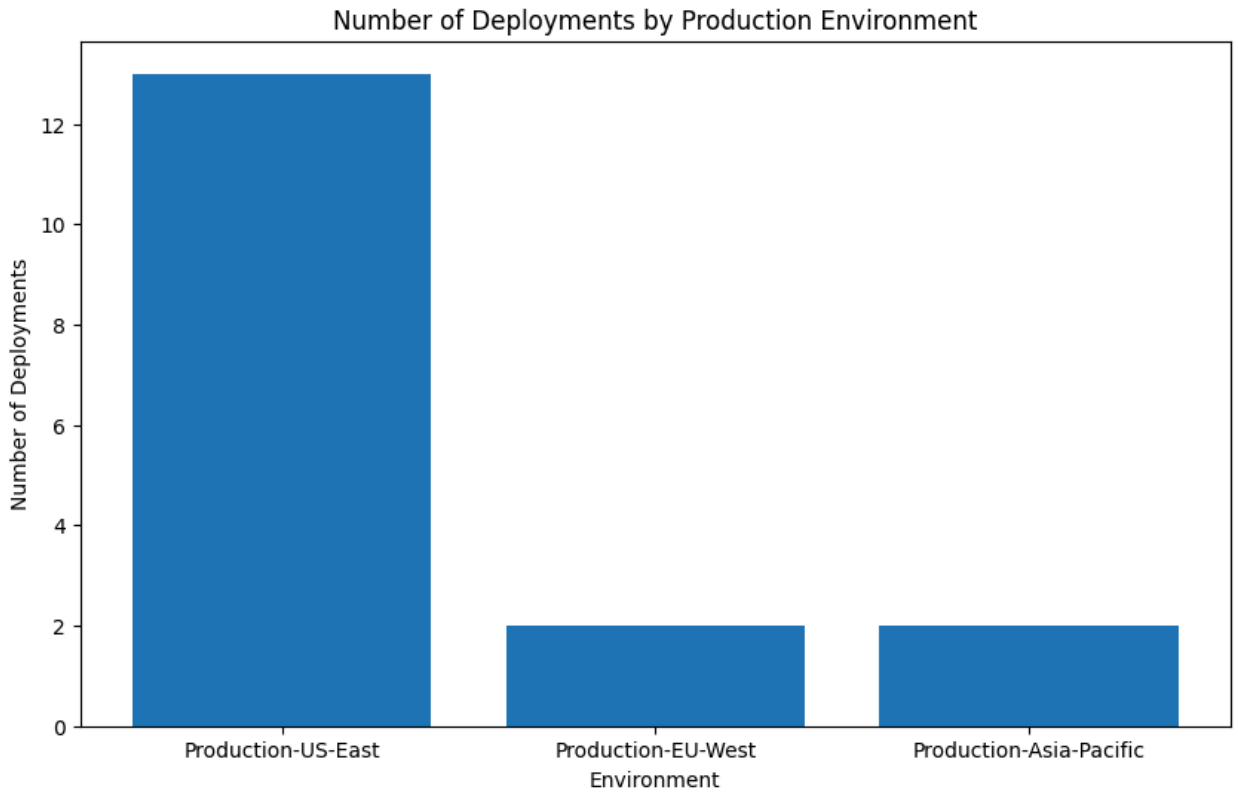
Query 4. Retrieve the number of deployments that happen for each environment.

```
SELECT
DATE(deployment_date) as deploy_date,
environment,
COUNT(*) as daily_count
FROM Deployment_History
GROUP BY DATE(deployment_date), environment
ORDER BY deploy_date ASC;
```

Findings: This analysis reveals that the production in the US-East is the dominant environment. Other production environments exhibit irregular single-deployment patterns in early April, indicating either a US-first deployment scheme with a delayed geographic spread for verified models or a limited regional rollout strategy.

Query 5. Retrieve the total deployments for the staging environment.

```
SELECT
environment,
COUNT(*) AS number_of_deployments
FROM Deployment_History
GROUP BY environment
HAVING environment != 'Staging' ;
```



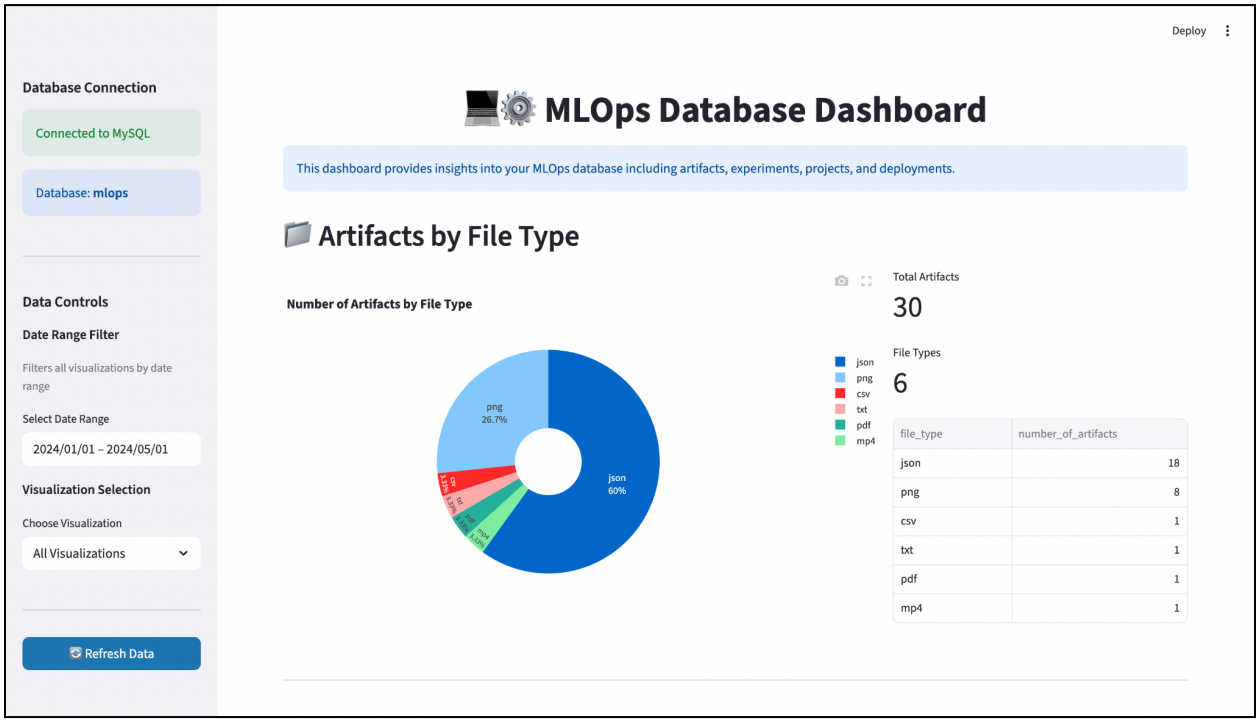
Findings: This shows that the PRoduction US-East is leading when it comes to the number of deployments in the staging environment. This 6:1 deployment split between the US-East and global areas indicates either a careful deployment strategy where models undergo thorough verification in the primary region before the international rollout, or an US-based user base driving main model serving requirements.

Streamlit dashboard - GUI

This Streamlit web app provides an interactive view of the MLOps data through visualizations and analytics. The dashboard connects to a MySQL database to display insights about artifacts, experiments, projects, and deployments. It uses interactive charts and filtering to help us explore the data.

Key Features:

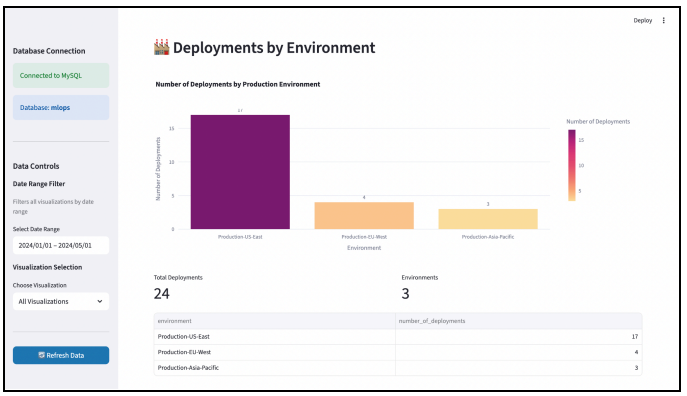
- Artifact Analysis - View file type distributions and artifact counts across ML projects with pie/donut charts
- Experiment Tracking - Monitor experiment counts, durations, and project activity status with bar and box plots
- Deployment Monitoring - Track deployment frequency across environments and time periods with area charts
- Interactive Visualizations - Plotly charts with hover tooltips, zoom, and pan
- Data Filtering - Date range selectors and visualization filters to focus on specific insights
- Real-time Updates - Refresh button to pull the latest data from the database




Deploy
Database Connection
Connected to MySQL
Database: mlops
Data Controls
Date Range Filter
Filters all visualizations by date range
Select Date Range
2024/01/01 – 2024/05/01
Visualization Selection
Choose Visualization
All Visualizations
Refresh Data
Top 10 Projects by Number of Experiments
Top 10 Projects by Number of Experiments
Number of Experiments
Active Status
Project Name
Total Projects
10
Active Projects
10
Total Experiments
15
project_name
number_of_experiments
currently_active
Customer Churn Prediction
2
1
Image Classification System
2
1
Sentiment Analysis Pipeline
2
1
Fraud Detection System
2
1


Deploy
Database Connection
Connected to MySQL
Database: mlops
Data Controls
Date Range Filter
Filters all visualizations by date range
Select Date Range
2024/01/01 – 2024/05/01
Visualization Selection
Choose Visualization
All Visualizations
Refresh Data
Experiment Durations by Project
Distribution of Experiment Durations by Project
Duration (Minutes)
Project
Total Runs
30
Avg Duration
342.2 min
Min Duration
150.0 min
Max Duration
540.0 min
Project_Name
Duration_Minutes
Customer Churn Prediction
150
Customer Churn Prediction
290
Image Classification System
510
Image Classification System
540


Deploy
Database Connection
Connected to MySQL
Database: mlops
Data Controls
Date Range Filter
Filters all visualizations by date range
Select Date Range
2024/01/01 – 2024/05/01
Visualization Selection
Choose Visualization
All Visualizations
Refresh Data
Deployment Frequency by Environment
Deployment Frequency by Environment Over Time
Number of Deployments
Date
Staging
Production-US-East
Production-EU-West
Production-Asia-Pacific
Total Deployments
30
Environments
4
Date Range
2024-03-02 to 2024-04-20
Days with Deployments
27
deploy_date
environment
daily_count
2024-04-20 00:00:00
Staging
1


Deploy
Database Connection
Connected to MySQL
Database: mlops
Data Controls
Date Range Filter
Filters all visualizations by date range
Select Date Range
2024/01/01 – 2024/05/01
Visualization Selection
Choose Visualization
All Visualizations
Refresh Data
Deployments by Environment
Number of Deployments by Production Environment
Number of Deployments
Environment
Production-US-East
Production-EU-West
Production-Asia-Pacific
Total Deployments
24
Environments
3
environment
number_of_deployments
Production-US-East
17
Production-EU-West
4
Production-Asia-Pacific
3

Neo4j Implementation

Graph Database Creation

CONSTRAINTS

```
CREATE CONSTRAINT FOR (u:User) REQUIRE u.id IS UNIQUE;
CREATE CONSTRAINT FOR (p:Project) REQUIRE p.id IS UNIQUE;
CREATE CONSTRAINT FOR (e:Experiment) REQUIRE e.id IS UNIQUE;
CREATE CONSTRAINT FOR (m:Model) REQUIRE m.id IS UNIQUE;
CREATE CONSTRAINT FOR (mv:ModelVersion) REQUIRE mv.id IS UNIQUE;
CREATE CONSTRAINT FOR (d:Dataset) REQUIRE d.id IS UNIQUE;
CREATE CONSTRAINT FOR (dv:DatasetVersion) REQUIRE dv.id IS UNIQUE;
CREATE CONSTRAINT FOR (er:ExperimentRun) REQUIRE er.id IS UNIQUE;
```

1. **USERS**

```
UNWIND [
    {id: 1, FirstName: 'Alice', LastName: 'Johnson', email: 'alice.johnson@company.com', created_at: '2024-01-15T09:00:00', roles: ['Administrator']},
    {id: 2, FirstName: 'Bob', LastName: 'Smith', email: 'bob.smith@company.com', created_at: '2024-01-16T10:30:00', roles: ['Administrator']},
    {id: 3, FirstName: 'Carol', LastName: 'Davis', email: 'carol.davis@company.com', created_at: '2024-01-17T11:00:00', roles: ['Administrator']},
    {id: 4, FirstName: 'David', LastName: 'Wilson', email: 'david.wilson@company.com', created_at: '2024-01-18T09:30:00', roles: ['Project Lead'], department: 'Machine Learning'},
    {id: 5, FirstName: 'Emma', LastName: 'Brown', email: 'emma.brown@company.com', created_at: '2024-01-19T10:00:00', roles: ['Project Lead'], department: 'Data Science'},
    {id: 6, FirstName: 'Frank', LastName: 'Miller', email: 'frank.miller@company.com', created_at: '2024-01-20T11:30:00', roles: ['Project Lead'], department: 'AI Research'},
    {id: 7, FirstName: 'Grace', LastName: 'Taylor', email: 'grace.taylor@company.com', created_at: '2024-01-21T09:00:00', roles: ['Project Lead'], department: 'Computer Vision'},
    {id: 8, FirstName: 'Henry', LastName: 'Anderson', email: 'henry.anderson@company.com', created_at: '2024-01-22T10:30:00', roles: ['Contributor']},
    {id: 9, FirstName: 'Iris', LastName: 'Martinez', email: 'iris.martinez@company.com', created_at: '2024-01-23T11:00:00', roles: ['Contributor']},
    {id: 10, FirstName: 'Jack', LastName: 'Garcia', email: 'jack.garcia@company.com', created_at: '2024-01-24T09:30:00', roles: ['Contributor']},
    {id: 11, FirstName: 'Kelly', LastName: 'Rodriguez', email: 'kelly.rodriguez@company.com', created_at: '2024-01-25T10:00:00', roles: ['Contributor']},
```

```
    {id: 12, FirstName: 'Liam', LastName: 'Hernandez', email: 'liam.hernandez@company.com', created_at: '2024-01-26T11:00:00', roles: ['Contributor']},
        {id: 13, FirstName: 'Maya', LastName: 'Lopez', email: 'maya.lopez@company.com', created_at: '2024-01-27T09:30:00', roles: ['Contributor']},
    {id: 14, FirstName: 'Nathan', LastName: 'Gonzalez', email: 'nathan.gonzalez@company.com', created_at: '2024-01-28T10:30:00', roles: ['Contributor']},
        {id: 15, FirstName: 'Olivia', LastName: 'Perez', email: 'olivia.perez@company.com', created_at: '2024-01-29T11:30:00', roles: ['Contributor']},
      {id: 16, FirstName: 'Peter', LastName: 'Sanchez', email: 'peter.sanchez@company.com', created_at: '2024-01-30T09:00:00', roles: ['Contributor']},
      {id: 17, FirstName: 'Quinn', LastName: 'Ramirez', email: 'quinn.ramirez@company.com', created_at: '2024-01-31T10:00:00', roles: ['Contributor']},
       {id: 18, FirstName: 'Rachel', LastName: 'Torres', email: 'rachel.torres@company.com', created_at: '2024-02-01T11:00:00', roles: ['Contributor']},
      {id: 19, FirstName: 'Samuel', LastName: 'Flores', email: 'samuel.flores@company.com', created_at: '2024-02-02T09:30:00', roles: ['Contributor']},
    {id: 20, FirstName: 'Tina', LastName: 'Rivera', email: 'tina.rivera@company.com', created_at: '2024-02-03T10:30:00', roles: ['Contributor']}
] AS row
CREATE (u:User)
SET u += row, u.created_at = datetime(row.created_at);
```

2. PROJECTS

```
UNWIND [
    {id: 1, name: 'Customer Churn Prediction', desc: 'ML model to predict customer churn', created_at: '2024-02-01T10:00:00', is_active: true, lead_id: 4, admin_id: 1, team: [{uid: 8, role: 'Data Engineer'}, {uid: 9, role: 'ML Engineer'}]},
    {id: 2, name: 'Image Classification System', desc: 'Deep learning model for product image classification', created_at: '2024-02-02T11:00:00', is_active: true, lead_id: 5, admin_id: 1, team: [{uid: 10, role: 'ML Engineer'}, {uid: 11, role: 'Data Scientist'}]},
      {id: 3, name: 'Sentiment Analysis Pipeline', desc: 'NLP pipeline for customer reviews', created_at: '2024-02-03T09:30:00', is_active: true, lead_id: 6, admin_id: 2, team: [{uid: 12, role: 'NLP Specialist'}, {uid: 13, role: 'Data Engineer'}]},
       {id: 4, name: 'Fraud Detection System', desc: 'Real-time fraud detection', created_at: '2024-02-04T10:30:00', is_active: true, lead_id: 7, admin_id: 2, team: [{uid: 14, role: 'ML Engineer'}, {uid: 15, role: 'Data Scientist'}]},
    {id: 5, name: 'Recommendation Engine', desc: 'Collaborative filtering system', created_at: '2024-02-05T11:30:00', is_active: true, lead_id: 4, admin_id: 3, team: [{uid: 16, role: 'ML Engineer'}, {uid: 17, role: 'Data Scientist'}]},
       {id: 6, name: 'Price Optimization Model', desc: 'Dynamic pricing model', created_at: '2024-02-06T10:00:00', is_active: true, lead_id: 5, admin_id: 3, team: [{uid: 18, role: 'Data Scientist'}, {uid: 19, role: 'ML Engineer'}]},
```

{id: 7, name: 'Object Detection Pipeline', desc: 'Warehouse automation', created_at: '2024-02-07T11:00:00', is_active: true, lead_id: 6, admin_id: 1, team: [{uid: 20, role: 'Computer Vision Engineer'}, {uid: 8, role: 'ML Engineer'}]},
{id: 8, name: 'Speech Recognition System', desc: 'Speech-to-text system', created_at: '2024-02-08T09:30:00', is_active: true, lead_id: 7, admin_id: 1, team: [{uid: 9, role: 'NLP Specialist'}]},
{id: 9, name: 'Demand Forecasting', desc: 'Time series forecasting', created_at: '2024-02-09T10:30:00', is_active: true, lead_id: 4, admin_id: 2, team: [{uid: 10, role: 'Data Scientist'}]},
{id: 10, name: 'Credit Risk Assessment', desc: 'Loan default probability', created_at: '2024-02-10T11:30:00', is_active: true, lead_id: 5, admin_id: 2, team: [{uid: 11, role: 'ML Engineer'}]},
{id: 11, name: 'Medical Image Analysis', desc: 'Detecting abnormalities', created_at: '2024-02-11T10:00:00', is_active: true, lead_id: 6, admin_id: 3, team: [{uid: 12, role: 'Computer Vision Engineer'}]},
{id: 12, name: 'Chatbot Intent Classification', desc: 'Customer service chatbot', created_at: '2024-02-12T11:00:00', is_active: true, lead_id: 7, admin_id: 3, team: [{uid: 13, role: 'NLP Specialist'}]},
{id: 13, name: 'Anomaly Detection System', desc: 'Network traffic analysis', created_at: '2024-02-13T09:30:00', is_active: true, lead_id: 4, admin_id: 1, team: [{uid: 14, role: 'ML Engineer'}]},
{id: 14, name: 'User Segmentation', desc: 'Clustering behavior patterns', created_at: '2024-02-14T10:30:00', is_active: true, lead_id: 5, admin_id: 1, team: []},
{id: 15, name: 'Ad Click Prediction', desc: 'Ad click probability', created_at: '2024-02-15T11:30:00', is_active: true, lead_id: 6, admin_id: 2, team: []}
] AS row
CREATE (p:Project {id: row.id, name: row.name, description: row.desc, is_active: row.is_active, created_at: datetime(row.created_at)})
WITH p, row
// Link Lead
MATCH (lead:User {id: row.lead_id})
MERGE (lead)-[:LEADS]->(p)
// Link Admin
WITH p, row
MATCH (admin:User {id: row.admin_id})
MERGE (admin)-[:ADMINISTERS]->(p)
// Link Team Members (This replaces embedding)
WITH p, row
UNWIND row.team as member
MATCH (u:User {id: member.uid})
MERGE (u)-[:CONTRIBUTES_TO {role: member.role}]->(p);

3. EXPERIMENTS

```
UNWIND [
   {id: 1, name: 'Baseline XGBoost Model', pid: 1}, {id: 2, name: 'Feature Engineering v1', pid: 1},
    {id: 3, name: 'ResNet50 Transfer Learning', pid: 2}, {id: 4, name: 'EfficientNet Comparison',
pid: 2},
   {id: 5, name: 'BERT Base Model', pid: 3}, {id: 6, name: 'RoBERTa Fine-tuning', pid: 3},
    {id: 7, name: 'Ensemble Fraud Detection', pid: 4}, {id: 8, name: 'Neural Network Approach',
pid: 4},
    {id: 9, name: 'Matrix Factorization', pid: 5}, {id: 10, name: 'Deep Learning Recommender', pid:
5},
   {id: 11, name: 'Elasticity Modeling', pid: 6}, {id: 12, name: 'YOLOv5 Detection', pid: 7},
   {id: 13, name: 'Wav2Vec2 Model', pid: 8}, {id: 14, name: 'LSTM Forecasting', pid: 9},
    {id: 15, name: 'Gradient Boosting Credit', pid: 10}, {id: 16, name: 'U-Net Segmentation', pid:
11},
    {id: 17, name: 'Intent Classification BERT', pid: 12}, {id: 18, name: 'Isolation Forest Anomaly',
pid: 13},
   {id: 19, name: 'K-Means Clustering', pid: 14}, {id: 20, name: 'Neural Network CTR', pid: 15}
] AS row
MATCH (p:Project {id: row.pid})
CREATE (e:Experiment {id: row.id, name: row.name})
MERGE (p)-[:HAS_EXPERIMENT]->(e);
```

4. MODEL REGISTRY

```
UNWIND [
   {id: 1, name: 'ChurnPredictor', creator_id: 4}, {id: 2, name: 'ProductClassifier', creator_id: 5},
   {id: 3, name: 'SentimentAnalyzer', creator_id: 6}, {id: 4, name: 'FraudDetector', creator_id: 7},
    {id: 5, name: 'ProductRecommender', creator_id: 4}, {id: 6, name: 'PriceOptimizer', creator_id:
5},
   {id: 7, name: 'ObjectDetector', creator_id: 6}, {id: 8, name: 'SpeechRecognizer', creator_id: 7},
   {id: 9, name: 'DemandForecaster', creator_id: 4}, {id: 10, name: 'CreditScorer', creator_id: 5},
       {id: 11, name: 'MedicalImageAnalyzer', creator_id: 6}, {id: 12, name: 'IntentClassifier',
creator_id: 7},
    {id: 13, name: 'AnomalyDetector', creator_id: 4}, {id: 14, name: 'UserSegmenter', creator_id:
5},
   {id: 15, name: 'AdClickPredictor', creator_id: 6}
] AS row
MATCH (u:User {id: row.creator_id})
CREATE (m:Model {id: row.id, name: row.name})
MERGE (u)-[:REGISTERED_MODEL]->(m);
```

5. MODEL VERSIONS

```
UNWIND [
   {id: 1, ver: 'v1.0.0', mid: 1, stage: 'Production'}, {id: 2, ver: 'v1.1.0', mid: 1, stage: 'Staging'},
   {id: 3, ver: 'v1.0.0', mid: 2, stage: 'Production'}, {id: 4, ver: 'v1.1.0', mid: 2, stage: 'Staging'},
   {id: 5, ver: 'v1.0.0', mid: 3, stage: 'Production'}, {id: 6, ver: 'v1.1.0', mid: 3, stage: 'Staging'},
      {id: 7, ver: 'v1.0.0', mid: 4, stage: 'Production'}, {id: 8, ver: 'v1.0.0', mid: 5, stage: 'Development'},
      {id: 9, ver: 'v1.0.0', mid: 6, stage: 'Production'}, {id: 10, ver: 'v1.0.0', mid: 7, stage: 'Production'},
      {id: 11, ver: 'v1.0.0', mid: 8, stage: 'Production'}, {id: 12, ver: 'v1.0.0', mid: 9, stage: 'Production'},
      {id: 13, ver: 'v1.0.0', mid: 10, stage: 'Production'}, {id: 14, ver: 'v1.0.0', mid: 11, stage: 'Staging'},
      {id: 15, ver: 'v1.0.0', mid: 12, stage: 'Production'}, {id: 16, ver: 'v1.0.0', mid: 13, stage: 'Production'},
      {id: 17, ver: 'v1.0.0', mid: 14, stage: 'Production'}, {id: 18, ver: 'v1.0.0', mid: 15, stage: 'Production'},
      {id: 19, ver: 'v1.2.0', mid: 1, stage: 'Development'}, {id: 20, ver: 'v1.2.0', mid: 2, stage: 'Development'}
] AS row
MATCH (m:Model {id: row.mid})
CREATE (mv:ModelVersion {id: row.id, version: row.ver, stage: row.stage})
MERGE (m)-[:HAS_VERSION]->(mv);
```

6. DATASETS

```
UNWIND [
   {id: 1, name: 'Customer Transaction History', uid: 8}, {id: 2, name: 'Product Image Repository', uid: 10},
    {id: 3, name: 'Customer Review Dataset', uid: 12}, {id: 4, name: 'Payment Transaction Logs', uid: 14},
    {id: 5, name: 'User-Item Interaction Matrix', uid: 16}, {id: 6, name: 'Historical Pricing Data', uid: 18},
    {id: 7, name: 'Warehouse Video Footage', uid: 20}, {id: 8, name: 'Speech Audio Recordings', uid: 8},
   {id: 9, name: 'Sales Historical Data', uid: 10}, {id: 10, name: 'Credit Application Data', uid: 12},
     {id: 11, name: 'Medical Image Collection', uid: 14}, {id: 12, name: 'Chatbot Conversation Logs', uid: 16},
     {id: 13, name: 'Network Traffic Data', uid: 18}, {id: 14, name: 'User Behavior Analytics', uid: 20},
   {id: 15, name: 'Advertisement Campaign Data', uid: 8}
] AS row
```

```
MATCH (u:User {id: row.uid})
CREATE (d:Dataset {id: row.id, name: row.name})
MERGE (u)-[:CREATED_DATASET]->(d);
```

7. DATASET VERSIONS

```
UNWIND [
  {id: 1, did: 1, ver: 'v1.0'}, {id: 2, did: 1, ver: 'v1.1'},
  {id: 3, did: 2, ver: 'v1.0'}, {id: 4, did: 2, ver: 'v1.1'},
  {id: 5, did: 3, ver: 'v1.0'}, {id: 6, did: 3, ver: 'v1.1'},
  {id: 7, did: 4, ver: 'v1.0'}, {id: 8, did: 5, ver: 'v1.0'},
  {id: 9, did: 6, ver: 'v1.0'}, {id: 10, did: 7, ver: 'v1.0'},
  {id: 11, did: 8, ver: 'v1.0'}, {id: 12, did: 9, ver: 'v1.0'},
  {id: 13, did: 10, ver: 'v1.0'}, {id: 14, did: 11, ver: 'v1.0'},
  {id: 15, did: 12, ver: 'v1.0'}, {id: 16, did: 13, ver: 'v1.0'},
  {id: 17, did: 14, ver: 'v1.0'}, {id: 18, did: 15, ver: 'v1.0'},
  {id: 19, did: 1, ver: 'v2.0'}, {id: 20, did: 2, ver: 'v1.2'}
] AS row
MATCH (d:Dataset {id: row.did})
CREATE (dv:DatasetVersion {id: row.id, version: row.ver})
MERGE (d)-[:HAS_VERSION]->(dv);
```

8. EXPERIMENT RUNS

```
UNWIND [
  {id: 1, eid: 1, dvid: 1, mvid: 1, hash: 'a1b2c3'}, {id: 2, eid: 2, dvid: 2, mvid: 2, hash: 'b2c3d4'},
  {id: 3, eid: 3, dvid: 3, mvid: 3, hash: 'c3d4e5'}, {id: 4, eid: 4, dvid: 4, mvid: 4, hash: 'd4e5f6'},
  {id: 5, eid: 5, dvid: 5, mvid: 5, hash: 'e5f6g7'}, {id: 6, eid: 6, dvid: 6, mvid: 6, hash: 'f6g7h8'},
  {id: 7, eid: 7, dvid: 7, mvid: 7, hash: 'g7h8i9'}, {id: 8, eid: 8, dvid: 7, mvid: 7, hash: 'h8i9j0'},
  {id: 9, eid: 9, dvid: 8, mvid: 8, hash: 'i9j0k1'}, {id: 10, eid: 10, dvid: 8, mvid: 8, hash: 'j0k1l2'},
   {id: 11, eid: 11, dvid: 9, mvid: 9, hash: 'k1l2m3'}, {id: 12, eid: 12, dvid: 10, mvid: 10, hash: 'l2m3n4'},
   {id: 13, eid: 13, dvid: 11, mvid: 11, hash: 'm3n4o5'}, {id: 14, eid: 14, dvid: 12, mvid: 12, hash: 'n4o5p6'},
   {id: 15, eid: 15, dvid: 13, mvid: 13, hash: 'o5p6q7'}, {id: 16, eid: 16, dvid: 14, mvid: 14, hash: 'p6q7r8'},
   {id: 17, eid: 17, dvid: 15, mvid: 15, hash: 'q7r8s9'}, {id: 18, eid: 18, dvid: 16, mvid: 16, hash: 'r8s9t0'},
   {id: 19, eid: 19, dvid: 17, mvid: 17, hash: 's9t0u1'}, {id: 20, eid: 20, dvid: 18, mvid: 18, hash: 't0u1v2'}
] AS row
```

```
MATCH (exp:Experiment {id: row.eid})
MATCH (dv:DatasetVersion {id: row.dvid})
MATCH (mv:ModelVersion {id: row.mvid})
CREATE (run:ExperimentRun {id: row.id, git_hash: row.hash})
MERGE (run)-[:BELONGS_TO]->(exp)
MERGE (run)-[:USED_DATASET_VERSION]->(dv)
MERGE (run)-[:PRODUCED_MODEL_VERSION]->(mv);
```

9. DEPLOYMENT HISTORY

```
UNWIND [
   {id: 1, mvid: 1, env: 'Production-US-East'}, {id: 2, mvid: 1, env: 'Production-EU-West'},
   {id: 3, mvid: 2, env: 'Staging'}, {id: 4, mvid: 3, env: 'Production-US-East'},
   {id: 5, mvid: 4, env: 'Staging'}, {id: 6, mvid: 5, env: 'Production-US-East'},
   {id: 7, mvid: 6, env: 'Staging'}, {id: 8, mvid: 7, env: 'Production-US-East'},
   {id: 9, mvid: 9, env: 'Production-US-East'}, {id: 10, mvid: 10, env: 'Production-US-East'},
   {id: 11, mvid: 11, env: 'Production-US-East'}, {id: 12, mvid: 12, env: 'Production-US-East'},
   {id: 13, mvid: 13, env: 'Production-US-East'}, {id: 14, mvid: 15, env: 'Production-US-East'},
   {id: 15, mvid: 16, env: 'Production-US-East'}, {id: 16, mvid: 17, env: 'Production-US-East'},
   {id: 17, mvid: 18, env: 'Production-US-East'}, {id: 18, mvid: 1, env: 'Production-Asia-Pacific'}
] AS row
MATCH (mv:ModelVersion {id: row.mvid})
CREATE (dep:Deployment {id: row.id, environment: row.env})
MERGE (mv)-[:DEPLOYED_AT]->(dep);
```

10. ARTIFACTS

```
UNWIND [
       {id: 1, rid: 1, type: 'Data', name: 'metrics.json'}, {id: 2, rid: 1, type: 'Plot', name: 'confusion_matrix.png'},
       {id: 3, rid: 2, type: 'Data', name: 'metrics.json'}, {id: 4, rid: 2, type: 'Plot', name: 'roc_curve.png'},
   {id: 14, rid: 7, type: 'File', name: 'feature_importance.csv'}
] AS row
MATCH (run:ExperimentRun {id: row.rid})
CREATE (a:Artifact {id: row.id, name: row.name, category: row.type})
MERGE (run)-[:GENERATED]->(a);
```

Queries for GraphDB

Query 1: List all the active projects and their project leads.

```
MATCH (lead:User)-[:LEADS]->(p:Project {is_active: true})
RETURN
p.name AS project_name,
lead.FirstName + ' ' + lead.LastName AS project_lead,
lead.department AS department,
p.created_at AS created_date
ORDER BY p.created_at DESC
LIMIT 10;
```

Table | RAW

	project_name	project_lead	department	created_date
1	"Ad Click Prediction"	"Frank Miller"	"AI Research"	2024-02-15T11:30:00Z
2	"User Segmentation"	"Emma Brown"	"Data Science"	2024-02-14T10:30:00Z
3	"Anomaly Detection System"	"David Wilson"	"Machine Learning"	2024-02-13T09:30:00Z
4	"Chatbot Intent Classification"	"Grace Taylor"	"Computer Vision"	2024-02-12T11:00:00Z
5	"Medical Image Analysis"	"Frank Miller"	"AI Research"	2024-02-11T10:00:00Z
6	"Credit Risk Assessment"	"Emma Brown"	"Data Science"	2024-02-10T11:30:00Z
7	"Demand Forecasting"	"David Wilson"	"Machine Learning"	2024-02-09T10:30:00Z
8	"Speech Recognition System"	"Grace Taylor"	"Computer Vision"	2024-02-08T09:30:00Z
9	"Object Detection Pipeline"	"Frank Miller"	"AI Research"	2024-02-07T11:00:00Z

Query 2: Retrieve models that are currently in the ‘Production’ stage.

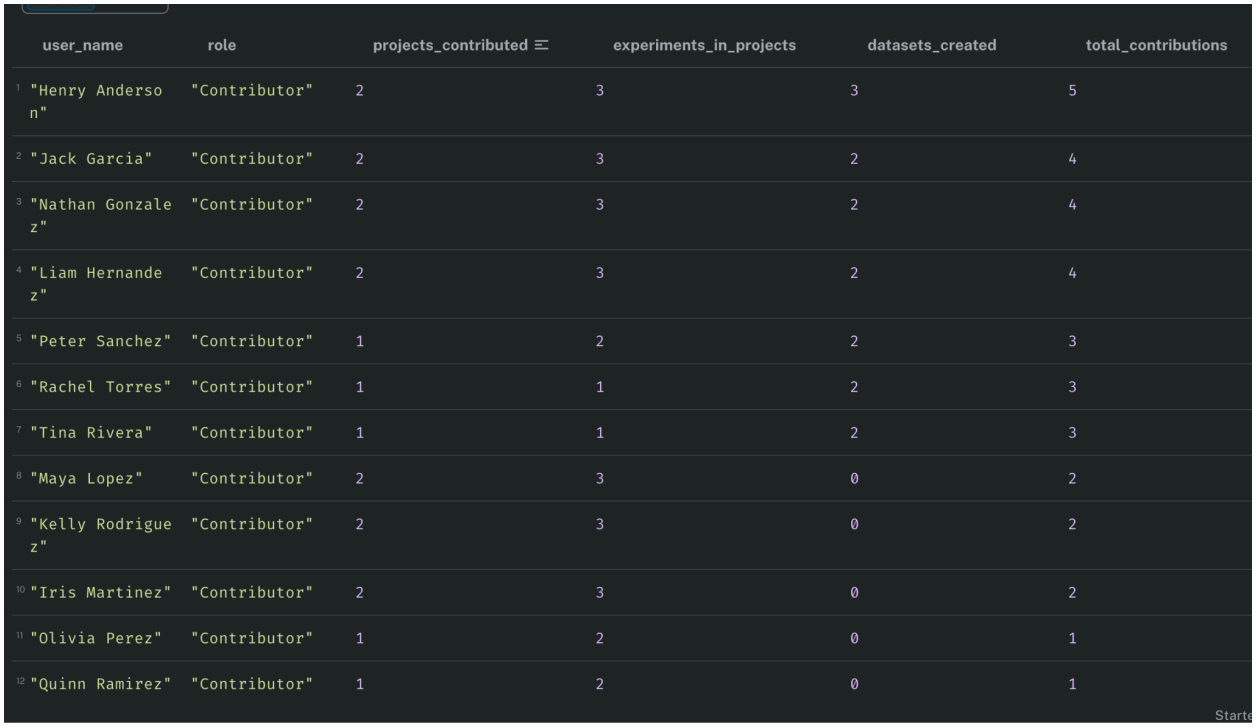
```
MATCH (m:Model)-[:HAS_VERSION]->(mv:ModelVersion {stage: 'Production'})
RETURN mv.version, m.name, mv.stage;
```

Table | RAW

	mv.version	m.name	mv.stage
1	"v1.0.0"	"ChurnPredictor"	"Production"
2	"v1.0.0"	"ProductClassifier"	"Production"
3	"v1.0.0"	"SentimentAnalyzer"	"Production"
4	"v1.0.0"	"FraudDetector"	"Production"
5	"v1.0.0"	"PriceOptimizer"	"Production"
6	"v1.0.0"	"ObjectDetector"	"Production"
7	"v1.0.0"	"SpeechRecognizer"	"Production"
8	"v1.0.0"	"DemandForecaster"	"Production"
9	"v1.0.0"	"CreditScorer"	"Production"
10	"v1.0.0"	"IntentClassifier"	"Production"

Query 3: Contribution of user across platforms

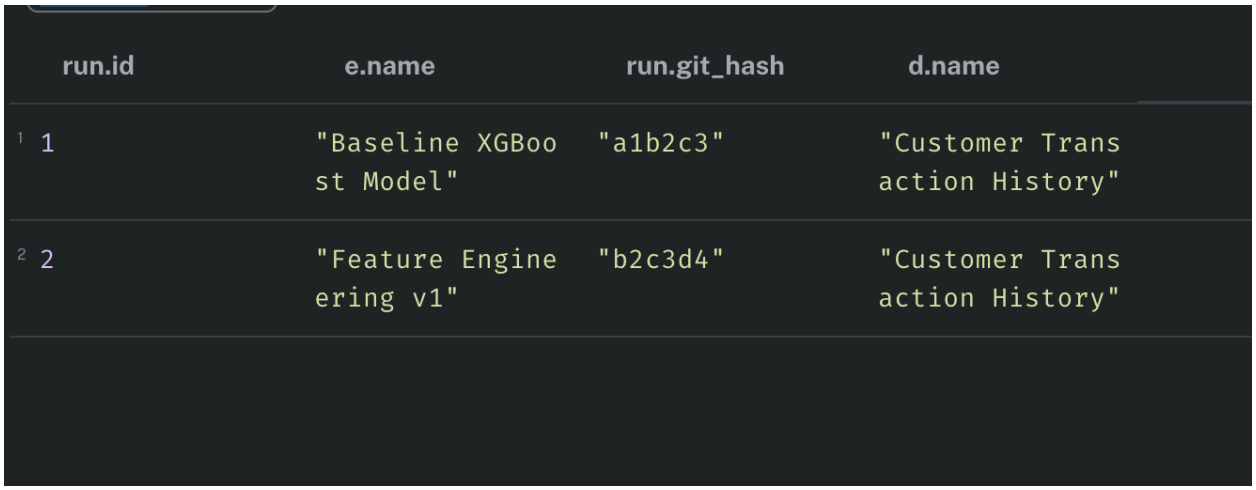
```
MATCH (u:User)
OPTIONAL MATCH (u)-[:CONTRIBUTES_TO]->(p:Project)
OPTIONAL MATCH (p)-[:HAS_EXPERIMENT]->(e:Experiment)
OPTIONAL MATCH (u)-[:CREATED_DATASET]->(d:Dataset)
RETURN
u.FirstName + ' ' + u.LastName AS user_name,
u.roles[0] AS role,
  count(DISTINCT p) AS projects_contributed,
  count(DISTINCT e) AS experiments_in_projects,
 count(DISTINCT d) AS datasets_created,
 count(DISTINCT p) + count(DISTINCT d) AS total_contributions
ORDER BY total_contributions DESC, experiments_in_projects DESC
LIMIT 15;
```



	user_name	role	projects_contributed	experiments_in_projects	datasets_created	total_contributions
1	"Henry Anderson"	"Contributor"	2	3	3	5
2	"Jack Garcia"	"Contributor"	2	3	2	4
3	"Nathan Gonzalez"	"Contributor"	2	3	2	4
4	"Liam Hernandez"	"Contributor"	2	3	2	4
5	"Peter Sanchez"	"Contributor"	1	2	2	3
6	"Rachel Torres"	"Contributor"	1	1	2	3
7	"Tina Rivera"	"Contributor"	1	1	2	3
8	"Maya Lopez"	"Contributor"	2	3	0	2
9	"Kelly Rodriguez"	"Contributor"	2	3	0	2
10	"Iris Martinez"	"Contributor"	2	3	0	2
11	"Olivia Perez"	"Contributor"	1	2	0	1
12	"Quinn Ramirez"	"Contributor"	1	2	0	1

Query 4: Show the detailed Experiment Runs.

```
MATCH (run:ExperimentRun)-[:BELONGS_TO]->(e:Experiment)
MATCH
(run)-[:USED_DATASET_VERSION]->(dv:DatasetVersion)<-[:HAS_VERSION]-(d:Datas
et {id: 1})
RETURN run.id, e.name, run.git_hash, d.name;
```



	run.id	e.name	run.git_hash	d.name
1	1	"Baseline XGBoost Model"	"a1b2c3"	"Customer Transaction History"
2	2	"Feature Engineering v1"	"b2c3d4"	"Customer Transaction History"

Conclusion and Recommendations

This project successfully implemented a thorough MLOps platform designed to address the key problems of version control, traceability and repeatability in machine learning operations. The hybrid database bridges the gap between the experimental nature of Data Science and the rigorous approach of software engineering.

With the utilization of MySQL, we enforced relational integrity on the core entities such as Project Hierarchies, User Management. The implementation on Neo4j, allowed efficient modelling of complex data, allowing the system to track a deployed model back to its original experiment run, source code (git hash), and particular dataset version. The successful implementation of queries on both the platforms show the efficiency of storing data and also helps us generate valuable insights for the data.

Recommendation and Future Scope

1. **Model Monitoring** : Implement the monitoring of models by checking their saved LogArtifacts to detect if a model's performance is dropping over time.
2. **Automate Deployments**: Connect the models to CI/CD tools so that whenever their status is updated to "Production", they are automatically deployed.
3. **Automate Data Validation**: Implementing the quality check step, by adding automated dataset scans for whenever a new dataset is added to the system, to check for missing values or redundant data.
4. **Storage Management**: Set automatic regular cleanups, in order to save space by deleting files that are no longer in use or too large, this will help in using the storage space efficiently.
5. **Enhanced Security**: Implement stricter permission levels, such that only important entities can make important decisions, like allowing the specific users to delete a database or giving them the permission to approve production deployments.