Normalization and Schema Refinement

- **List all the Relations & Schemas with all details (Original Design of Database)**

Driller

Attributes: DrillerID (VARCHAR) (PRIMARY KEY)
Name (CHAR)
Certification (VARCHAR)
Contact (INT)
EquipmentID (VARCHAR) (FOREIGN KEY)

Rig Technician

Attributes: RtID (VARCHAR) (PRIMARY KEY)
Name (CHAR) Certification (VARCHAR)
EquipmentID (VARCHAR) (FOREIGN KEY)

Mud Logger

Attributes: MlID (VARCHAR) (PRIMARY KEY)
Name (CHAR) Certification (VARCHAR)
WellID (VARCHAR) (FOREIGN KEY)
Contact (INT)

Production Supervisor

Attributes: PsID (VARCHAR) (PRIMARY KEY)
Name (CHAR) Certification (VARCHAR)
MetricsID (VARCHAR) (FOREIGN KEY)
Status (CHAR) Contact (INT)

Manufacturer

Attributes: ManufacturerID (VARCHAR) (PRIMARYKEY) Name (CHAR)
Address (CHAR) Contact (INT)

Well

Attributes: WellID (VARCHAR) (PRIMARY KEY)
Type (CHAR)
Depth (NUMERIC)
Status (CHAR)
DrillerID (VARCHAR) (FOREIGN KEY)

Metrics

Attributes: MetricsID (VARCHAR) (PRIMARY KEY)
MlId (VARCHAR) (FOREIGN KEY)
Measurement (DECIMAL) Accuracy (DECIMAL)

Maintenance

Attributes: MaintenanceId (VARCHAR) (PRIMARY KEY)
Schedule (DATE)
Type (CHAR)
RtId (VARCHAR) (FOREIGN KEY)
Cost (INT)
EquipmentId (VARCHAR) (FOREIGN KEY)

Equipment

Attributes: EquipmentID (VARCHAR) (PRIMARY KEY)
Name (CHAR)
Type (CHAR)
ManufacturerID (VARCHAR) (FOREIGN KEY) Cost (INT)

- **Identify and list all types of dependencies (PK, FK, Functional Dependencies) for each relation**

- **Driller**
  - **Primary Key (PK)**:
    - DrillerID
  - **Foreign Key (FK)**:
    - EquipmentID (references Equipment)
  - **Functional Dependencies (FD)**:
    - DrillerID → Name, Certification, Contact, EquipmentID

- **Rig Technician**
  - **Primary Key (PK)**:
    - RtID
  - **Foreign Key (FK)**:
    - EquipmentID (references Equipment)
  - **Functional Dependencies (FD)**:
    - RtID → Name, Certification, EquipmentID

- **Mud Logger**
  - **Primary Key (PK)**:
    - MlID
  - **Foreign Key (FK)**:
    - WellID (references Well)
  - **Functional Dependencies (FD)**:
    - MlID → Name, Certification, WellID, Contact

- **Production Supervisor**
  - **Primary Key (PK)**:
    - PsID
  - **Foreign Key (FK)**:
    - MetricsID (references Metrics)
  - **Functional Dependencies (FD)**:
    - PsID → Name, Certification, MetricsID, Status, Contact
- **Manufacturer**
  - **Primary Key (PK)**:
    - ManufacturerID
  - **Functional Dependencies (FD)**:
    - ManufacturerID → Name, Address, Contact
- **Well**
  - **Primary Key (PK)**:
    - WellID
  - **Foreign Key (FK)**:
    - DrillerID (references Driller)
  - **Functional Dependencies (FD)**:
    - WellID → Type, Depth, Status, DrillerID
- **Metrics**
  - **Primary Key (PK)**:
    - MetricsID
  - **Foreign Key (FK)**:
    - MlId (references Mud Logger)
  - **Functional Dependencies (FD)**:
    - MetricsID → MlId, Measurement, Accuracy

- **Maintenance**
  - **Primary Key (PK)**:
    - MaintenanceId
  - **Foreign Keys (FK)**:
    - RtId (references Rig Technician)
    - EquipmentId (references Equipment)
  - **Functional Dependencies (FD)**:
    - MaintenanceId → Schedule, Type, RtId, Cost, EquipmentId
- **Equipment**
  - **Primary Key (PK)**:
    - equipmentID
  - **Foreign Key (FK)**:
    - ManufacturerID (references Manufacturer)
  - **Functional Dependencies (FD)**:
    - equipmentID → Name, Type, ManufacturerID, Cost

- **List of update, delete, and insert anomalies for every schema (document it)**

- **Driller**
    - **Delete Anomalies**: Deleting a Driller could remove critical information about their associated equipment or Wells.
    - **Insert Anomalies**: Inserting new Equipment linked to a Driller requires the Driller to exist first.

- **Rig Technician**
    - **Delete Anomalies**: Deleting a Rig Technician could lead to affect records for equipment they managed.

- **Mud Logger**
    - **Delete Anomalies**: Deleting a Mud Logger could lose Well-related information.
    - **Insert Anomalies**: New records for Mud Loggers cannot be created without existing WellIDs.

- **Production Supervisor**
    - **Delete Anomalies**: If a Production Supervisor is deleted, associated Metrics may lose context.
    - **Insert Anomalies**: New Metrics require a corresponding Production Supervisor to be created first.

- **Manufacturer**
    - **Update Anomalies**: Changes to Manufacturer details need to be updated wherever the ManufacturerID is referenced.
    - **Delete Anomalies**: Deleting a Manufacturer can lead to loss of reference in Equipment records.
    - **Insert Anomalies**: New Equipment cannot be created without a valid ManufacturerID.

- **Well**
    - **Update Anomalies**: Modifying Well attributes needs updates wherever the WellID is used.
    - **Delete Anomalies**: If a Well is deleted, the link to its Driller may be lost.
    - **Insert Anomalies**: Inserting a new Well requires an existing DrillerID.

- **Metrics**
    - **Delete Anomalies**: Deleting a Metrics record could lead to loss of important measurement data.
    - **Insert Anomalies**: Inserting new metrics requires a valid MlID, and the existence of a Mud Logger first.

- **Maintenance**
    - **Delete Anomalies**: If a Maintenance record is deleted, critical scheduling or cost information could be lost.
    - **Insert Anomalies**: New Maintenance entries require valid RtId and EquipmentId, which may affect data entry if those records don't exist.

- **Equipment**
    - **Delete Anomalies**: Deleting an Equipment record affects Maintenance records that reference it.

- **List of redundancies existing for every schema which is part of the database (document it).**

- **Driller**

Deleting a Driller may remove critical links to Equipment or Wells, leading to data loss.

- **Rig Technician**

No redundancies.

- **Mud Logger**

No redundancies.

- **Production Supervisor**

Removing a Production Supervisor risks losing information for associated Metrics.

- **Manufacturer**

Deleting or updating a Manufacturer can leave Equipment records without references.

- **Well**

No redundancies.

- **Metrics**

Removing a Metrics record can result in the loss of critical measurement data tied to Mud Loggers.

- **Maintenance**

No redundancies.

- **Equipment**

  No redundancies.

- **Normalize the database upto 1NF (scalar values)**

1. Each table has a primary key.
2. All entries in a column are of the same data type.
3. Each column contains atomic values (no multi-valued or composite attributes).
4. Each record is unique.

Driller

Attributes: DrillerID (VARCHAR) (PRIMARY KEY)
Name (CHAR)
Certification(VARCHAR)
EquipmentID (VARCHAR) (FOREIGN KEY)

DrillerContact

Attributes: PRIMARY KEY – (DrillerID (VARCHAR), Contact (INT))

Rig Technician

Attributes: RtID (VARCHAR) (PRIMARY KEY)
Name (CHAR)
Certification(VARCHAR)
EquipmentID (VARCHAR) (FOREIGN KEY)

RtContact

Attributes: PRIMARY KEY – (RtID (VARCHAR), Contact (INT))

Mud Logger

Attributes: MlID (VARCHAR) (PRIMARY KEY)
Name (CHAR)
Certification(VARCHAR)
WellID (VARCHAR) (FOREIGN KEY)

MlContact

Attributes: PRIMARY KEY – (MlID (VARCHAR), Contact (INT))

Production Supervisor

Attributes: PsID (VARCHAR) (PRIMARY KEY)
Name (CHAR)
Certification(VARCHAR)
MetricsID (VARCHAR) (FOREIGN KEY)
Status (CHAR)

PsContact

Attributes: PRIMARY KEY – (PsID (VARCHAR), Contact (INT))

Manufacturer

Attributes: ManufacturerID (VARCHAR) (PRIMARY KEY)
Name (CHAR)
Address (CHAR)

ManufacturerContact

Attributes: PRIMARY KEY – (ManufacturerID (VARCHAR), Contact (INT))

Well

Attributes: WellID (VARCHAR) (PRIMARY KEY)
Type (CHAR) Depth (NUMERIC)
Status (CHAR)
DrillerID (VARCHAR) (FOREIGN KEY)

Metrics

Attributes: MetricsID (VARCHAR) (PRIMARY KEY)
MlId (VARCHAR) (FOREIGN KEY)
Measurement (DECIMAL) Accuracy (DECIMAL)

Maintenance

Attributes: MaintenanceId (VARCHAR)(PRIMARYKEY)
Schedule (DATE)
Type (CHAR)
RtId (VARCHAR) (FOREIGN KEY)
Cost (INT)
EquipmentId (VARCHAR) (FOREIGN KEY)

Equipment

Attributes: equipmentID (VARCHAR) (PRIMARY KEY)
Name (CHAR)
Type (CHAR)
ManufacturerID (VARCHAR) (FOREIGNKEY)
Cost (INT)

- **Normalize the database upto 2NF (Remove Partial Dependencies)**

1. Ensure 1NF: Start with a table that is already in 1NF, meaning it has a primary key, atomic values, and unique records.
2. Identify Composite Keys: If your primary key consists of multiple columns, identify which attributes depend on the entire key.
3. Remove Partial Dependencies: If any non-key attribute depends only on part of a composite key, separate those attributes into a new table.

Our database has been successfully normalized to the second normal form (2NF).

Each table in our database features a primary key that consists of a single attribute. This unique identifier guarantees that every record is distinct and easily retrievable.

Prior to achieving 2NF, the database was ensured to be in first normal form (1NF). This involved confirming that all data entries are atomic, with no multi-valued attributes or repeating groups present in any table.

- **Normalize the database upto 3NF (Remove Transitive Dependencies)**

1. Ensure your database is already in 2NF, meaning all non-key attributes are fully dependent on the primary key.

2. Check for any non-key attributes that depend on other non-key attributes rather than directly on the primary key.

3. For each transitive dependency, create a new table to separate the non-key attributes that are dependent on each other.

Our database has been successfully normalized to the second normal form (3NF).

The database was confirmed to be in second normal form (2NF) prior to this stage. This means all non-key attributes are fully functionally dependent on the primary key, with no partial dependencies present.

In our analysis, we found that no non-key attribute depends on another non-key attribute. Each non-key attribute is directly related to the primary key, ensuring that there are no transitive dependencies. This organization enhances the clarity and efficiency of data retrieval and management.

DDL STATEMENTS

```
1.CREATE TABLE Driller (
        DrillerID VARCHAR(50) PRIMARY KEY,
        Name CHAR(100),
        Certification VARCHAR(100),
        EquipmentID VARCHAR(50),
        FOREIGN KEY (EquipmentID) REFERENCES
    Equipment(EquipmentID)
);

2.CREATE TABLE DrillerContact (
        DrillerID VARCHAR(50),
        Contact INT,
        PRIMARY KEY (DrillerID, Contact),
        FOREIGN KEY (DrillerID) REFERENCES
    Driller(DrillerID)
);

3.CREATE TABLE RigTechnician (
        RtID VARCHAR(50) PRIMARY KEY,
        Name CHAR(100),
        Certification VARCHAR(100),
        EquipmentID VARCHAR(50),
        FOREIGN KEY (EquipmentID) REFERENCES
    Equipment(EquipmentID)
);


4.CREATE TABLE RtContact (
        RtID VARCHAR(50),
        Contact INT,
        PRIMARY KEY (RtID, Contact),
        FOREIGN KEY (RtID) REFERENCES
    RigTechnician(RtID)
);

5.CREATE TABLE MudLogger (
        MlID VARCHAR(50) PRIMARY KEY,
        Name CHAR(100),
```

```
    Certification VARCHAR(100),
    WellID VARCHAR(50),
    FOREIGN KEY (WellID) REFERENCES Well(WellID)
);

6.CREATE TABLE MlContact (
    MlID VARCHAR(50),
    Contact INT,
    PRIMARY KEY (MlID, Contact),
    FOREIGN KEY (MlID) REFERENCES MudLogger(MlID)
);


7.CREATE TABLE ProductionSupervisor (
    PsID VARCHAR(50) PRIMARY KEY,
    Name CHAR(100),
    Certification VARCHAR(100),
    MetricsID VARCHAR(50),
    Status CHAR(20),
    FOREIGN KEY (MetricsID) REFERENCES
  Metrics(MetricsID)
);

8.CREATE TABLE PsContact (
    PsID VARCHAR(50),
    Contact INT,
    PRIMARY KEY (PsID, Contact),
    FOREIGN KEY (PsID) REFERENCES
  ProductionSupervisor(PsID)
);

9.CREATE TABLE Manufacturer (
    ManufacturerID VARCHAR(50) PRIMARY KEY,
    Name CHAR(100),
    Address CHAR(200)
);
```

```
10.CREATE TABLE ManufacturerContact (
        ManufacturerID VARCHAR(50),
        Contact INT,
        PRIMARY KEY (ManufacturerID, Contact),
        FOREIGN KEY (ManufacturerID) REFERENCES
    Manufacturer(ManufacturerID)
);

11.CREATE TABLE Well (
        WellID VARCHAR(50) PRIMARY KEY,
        Type CHAR(50),
        Depth NUMERIC,
        Status CHAR(20),
        DrillerID VARCHAR(50),
        FOREIGN KEY (DrillerID) REFERENCES
    Driller(DrillerID)
);

12.CREATE TABLE Metrics (
        MetricsID VARCHAR(50) PRIMARY KEY,
        MlID VARCHAR(50),
        Measurement DECIMAL(10, 2),
        Accuracy DECIMAL(10, 2),
        FOREIGN KEY (MlID) REFERENCES MudLogger(MlID)
);

13.CREATE TABLE Maintenance (
        MaintenanceID VARCHAR(50) PRIMARY KEY,
        Schedule DATE,
        Type CHAR(50),
        RtID VARCHAR(50),
        Cost INT,
        EquipmentID VARCHAR(50),
        FOREIGN KEY (RtID) REFERENCES
    RigTechnician(RtID),
        FOREIGN KEY (EquipmentID) REFERENCES
    Equipment(EquipmentID)
);
```

```
14.CREATE TABLE Equipment (
        EquipmentID VARCHAR(50) PRIMARY KEY,
        Name CHAR(100),
        Type CHAR(50),
        ManufacturerID VARCHAR(50),
        Cost INT,
        FOREIGN KEY (ManufacturerID) REFERENCES
      Manufacturer(ManufacturerID)
);
```