**Database Design Guide**

The Regional Transport Office (RTO) is a government body responsible for the registration of vehicles, issuance of driving licenses, apply License (Learning License or Driving License ) ,change address ,check challan Insurance policy ,feedback.

This guide will help the student to create a database on the E-Regional Transport Office (RTO). It will help to manage the below functionalities.

**Entities**

Let’s identify the entities of the E-Regional Transport Office (RTO).

* 1. user
  2. Admin
  3. RTO Officer
  4. Vehicle
  5. Driving / Learners License
     1. Learning License
     2. Driving License
  6. Address change
  7. Challan
  8. Insurance Policy
  9. Feedback

\*\*\* Now let’s identify the attributes and relationships of each entity for the RTO

**RTO Project Functionality**

**1. User**

* **Profile Management:** Create and manage user profiles.
* **Service Requests:** Request services such as vehicle registration, license application, and address change.
* **Document Upload:** Upload necessary documents for various applications.
* **Tracking Applications:** Track the status of applications and service requests.
* **Payment:** Make payments for services such as registration fees, taxes, and fines.

**2. Admin**

* **System Administration:** Manage the RTO system, user roles, and permissions.
* **Configuration:** Configure system settings, service parameters, and fee structures.
* **Reporting:** Generate and manage reports on various RTO activities.
* **User Management:** Add, remove, and manage users and their access levels.

**3. RTO Officer**

* **Application Review:** Review and approve or reject applications for vehicle registration, licenses, permits, etc.
* **Inspection:** Conduct vehicle inspections and approve fitness certificates.
* **Enforcement:** Monitor compliance with motor vehicle laws and issue challans (fines) for violations.
* **Record Management:** Maintain records of registered vehicles, issued licenses, and other documentation.
* **Public Interaction:** Address public queries, grievances, and provide necessary support.

**4. Vehicle Registration**

* **Registration:** Register new and used vehicles, issue registration certificates.
* **Ownership Transfer:** Facilitate transfer of vehicle ownership.
* **Renewal:** Renew vehicle registrations.
* **Fitness Certification:** Issue fitness certificates to ensure vehicles meet safety and emission standards.
* **Permits:** Issue various permits for commercial and private vehicles.
* **Tax Collection:** Collect road taxes and other related fees.

**5. Driving / Learner's License**

**a. Learning License**

* **Application:** Apply for a learner's license.
* **Test Scheduling:** Schedule learner's license tests.
* **Issuance:** Issue learner's licenses after passing the test.
* **Validity Management:** Manage the validity period of learner's licenses.

**b. Driving License**

* **Application:** Apply for a permanent driving license.
* **Test Scheduling:** Schedule driving tests.
* **Issuance:** Issue driving licenses upon passing the test.
* **Renewal:** Renew expired driving licenses.
* **Upgradation:** Upgrade licenses to include additional vehicle categories.
* **Duplicate License:** Issue duplicate licenses in case of loss or damage.

**7. Challan**

* **Issuance:** Issue challans for traffic violations and non-compliance with motor vehicle laws.
* **Payment:** Allow users to pay challans online.
* **Tracking:** Track the status of issued challans and their payments.
* **Dispute Resolution:** Handle disputes related to challans and fines.

**8. Insurance Policy**

* **Verification:** Verify the validity of vehicle insurance policies during registration and renewal.
* **Renewal Reminders:** Send reminders for insurance policy renewals.
* **Integration:** Integrate with insurance providers for seamless data exchange.

**9. Feedback**

* **Submission:** Allow users to submit feedback on RTO services.
* **Review:** Review and categorize feedback for actionable insights.
* **Response:** Respond to user feedback and address any issues raised.
* **Improvement:** Use feedback to improve RTO processes and services.

**User**

* **Attributes:**

* 1. **U\_id (PK) :** Unique identifier for the user
  2. **U\_name :** The name chosen by the user for logging in.
  3. **U\_pass:** Encrypted password for securing the user's account.
  4. **U\_Email :** User's email address for communication and notifications
  5. **FirstName:** User's first name.
  6. **LastName:** User's last name.
  7. **U\_mobile** : User's contact number.
  8. **U\_DOB :** Use’s date of birth.
  9. **U\_add :** User's physical address.

* **Relationships:**

One **User** can Apply in more than one **License** (One-to-Many)

One **User** can Register more than one **Vehicle** (One-to-Many)

**Admin**

* **Attributes:**

1. **A\_id (Primary Key) :**
2. **A\_name :**
3. **A\_pass :**

* **Relationships:**

one **Admin** can manage by Many **RTO Officer** (**One-to-Many**)

**RTO Officer**

* **Attributes:**

1. **Rto\_id (PK):** Unique identifier for the RTO Officer
2. **Rto\_name :** Name of RTOOfficer
3. **Rto\_pass :** Encrypted password for securing the RTOOfficer account
4. **Rto\_idcard\_no:** Identity Card Of RTOOfficer for cross verification
5. **Rto\_email:** RTO officer email address for communication and notifications

* **Relationships:**

\*\*//One **Admin** can manage by Many **Vehicle** (**One-to-Many**)

one **RTO Officer** can manage by Many **Vehicle** (**One-to-Many**)

**Vehicle**

* **Attributes:**

1. **vehicle \_id :**
2. **Ve\_Registration \_no :**
3. **Vehicle\_Type :**
4. **Owner\_Name :**
5. **Address :**
6. **VehicleModel :**
7. **YearOfManufacture :**
8. **EngineNumber:**
9. **ChassisNumbe:**

* **Relationships:**

One **Vehicle** Register can apply One **Driving / Learners License**(**One-to-One**)

**Driving / Learners License**

* **Attributes:**

1. **user\_id :**
2. **user\_name :**
3. **date\_of\_birth :**
4. **type\_of\_license :**

* **Relationships:**

One **Driving / Learners License** can apply One **Learning License** (**One-to-one**) One **Driving / Learners License** can apply One **Driving License** (**One-to-one**)

**Learning License:**

* **Attributes:**

1. **LL\_id(PK) :**
2. **LL\_name :**
3. **LL\_address :**
4. **LL\_DOB :**
5. **LL\_pass :**
6. **LL\_test\_mark**

* **Relationships:**

One **Driving License** can apply More than One Time **Address change** (**One-to-Many**)

One **Learning License** can apply one **Driving License** (**One-to-One**)

**Driving License**

* **Attributes**:

**DL\_lic\_id (PK) :**

**DL\_name :**

**DL\_address :**

**DL\_DOB :**

**LL\_NO :**

**DL\_pass :**

**DL\_test\_mark:**

* **Relationships:**

One **Driving License** can check Many **Challan** (**One-to-Many**)

**Challan**

* **Attributes**:

**Challan\_id(PK)** :

**Veh\_No** :

**Chassis\_No** :

* **Relationships:**

One **user** will be check many **Challan (One-to-Many)**

One-to-Many: A user can have multiple challans

One-to-Many: A vehicle can have multiple challans(**One-to-Many**)

**Insurance Policy**

* **Attributes**:

**Policy\_id (PK) :**

**Vehicle\_id (FK) :**

**Policy\_type :**

**Veh\_regi\_No :**

**Owner\_name :**

**Owner\_email :**

**Amount :**

* **Relationships:**

One **User** can apply many **Insurance Policy (One-to-Many)**

**Feedback**

* **Attributes**:

**Feedback\_id (PK) :**

**U\_id(FK) :**

**Username :**

**Feedback\_date :**

**Feedback\_text :**

* **Relationships**:

A **user** can give multiple **feedbacks**. (**One-to-Many**)

1. **E-R Diagram (ERD)**

An Entity-Relationship Diagram (ERD) is a visual representation of the data model that shows the entities, attributes, relationships between entities, and cardinality. ERDs are commonly used in database design to help developers and stakeholders understand the structure and relationships within a database.

**Identify Entities**

* + Start by identifying the main entities in your system. These are the objects or concepts about which you want to store data.
  + Each entity should correspond to a table in your database.

**Define Attributes**

* + For each entity, list the attributes (properties or fields) that describe it.
  + These attributes will become columns in the corresponding database table.

**Identify Relationships**

* + Determine how entities are related to each other. There are three types of relationships: one-to-one (1:1), one-to-many (1:N), and many-to-many (N:M).
  + Represent these relationships using lines connecting the entities.

Let’s see a few examples of relationships:

**One to One**

Driving license

**Can Apply**

User

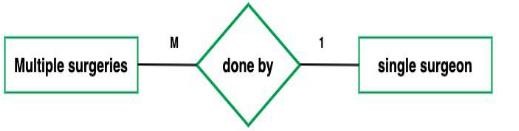
**One to Many**

**Can Check**

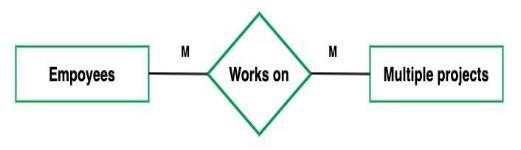
Challan

User

**Many to One**



**Many to Many**



**Cardinality Notation**

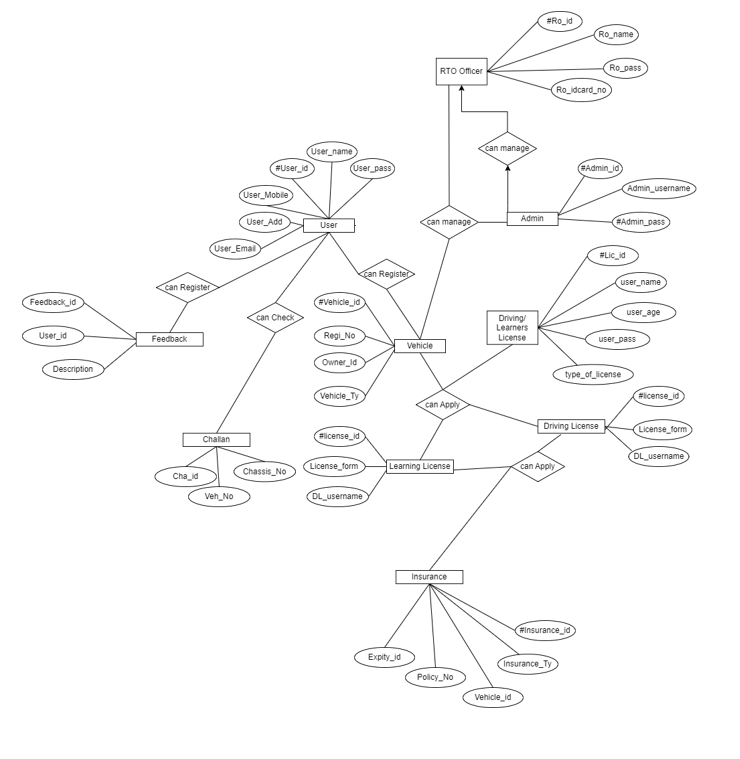
Cardinality represents the number of times an entity of an entity set participates in a relationship set. Or we can say that the cardinality of a relationship is the number of tuples (rows) in a relationship.

* + Use notation (such as Crow's Foot Notation or Chen Notation) to indicate the cardinality of each relationship.
  + Cardinality describes how many instances of one entity are related to how many instances of another entity.
  + Common notations include:
  + One (1)
  + Zero or one (0..1)
  + Many (N)

Zero or many (0..N)

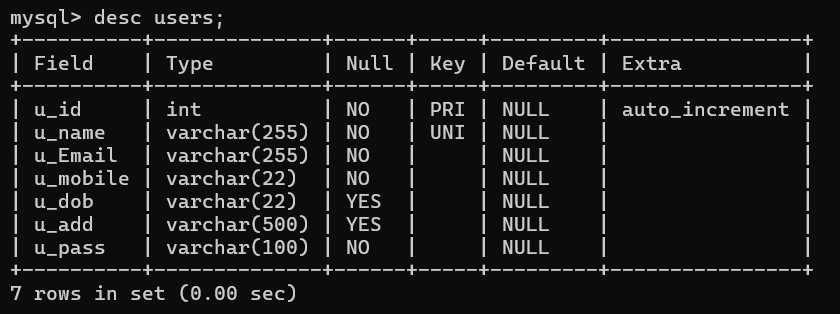
Now, let’s create the ER diagram to visually represent the entities and relationships.

**ERD Diagram**

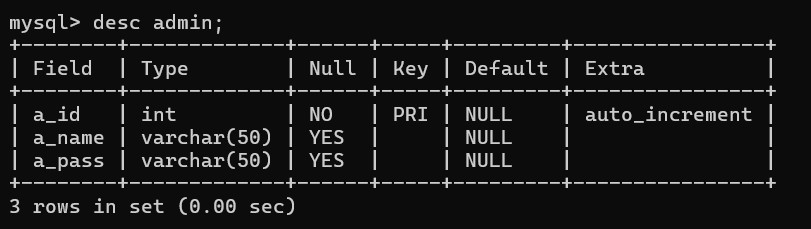


Let’s identify the entities of the E-Regional Transport Office (RTO)

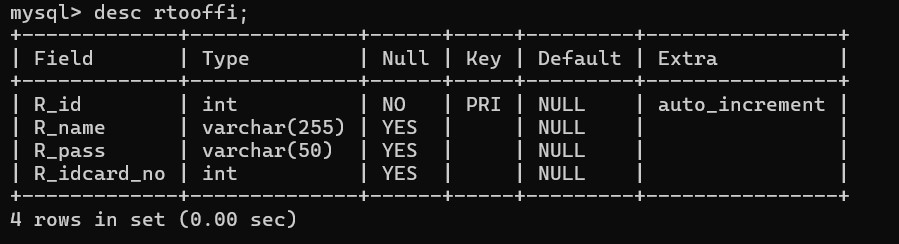
**1. User**



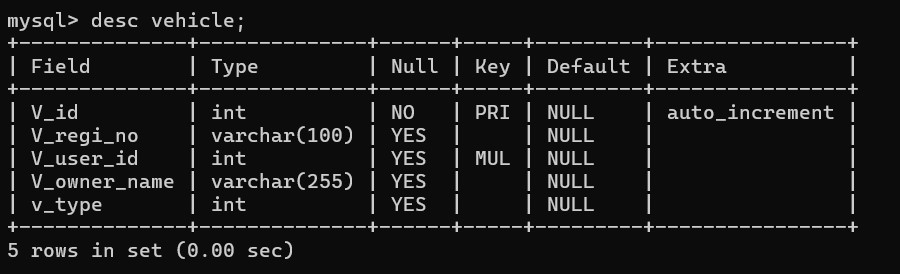
1. **Admin**



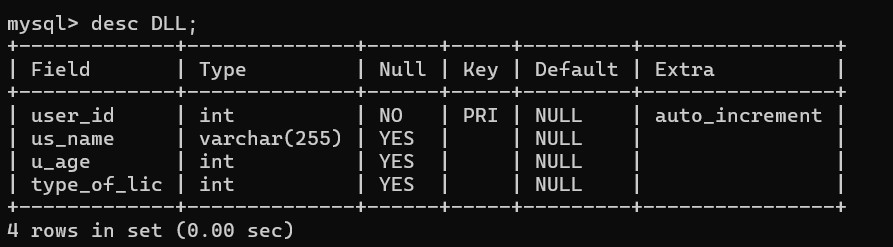
1. **RTO Officer**



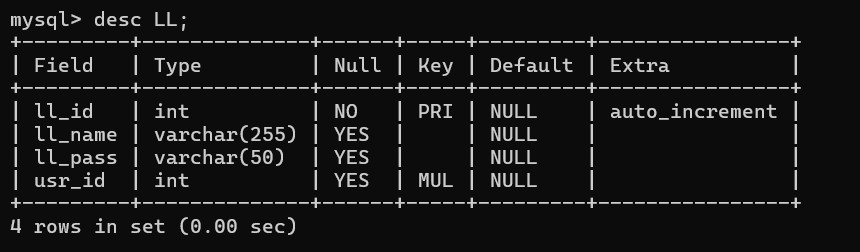
1. **Vehicle**



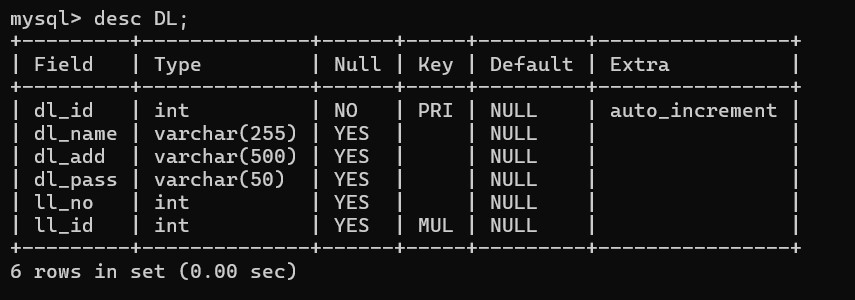
1. **Driving / Learners License**



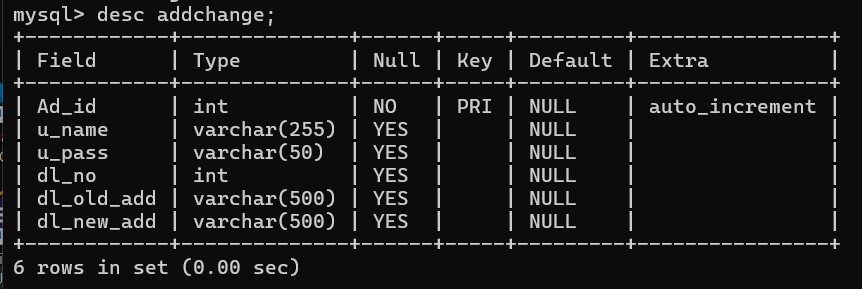
1. **Learning License**



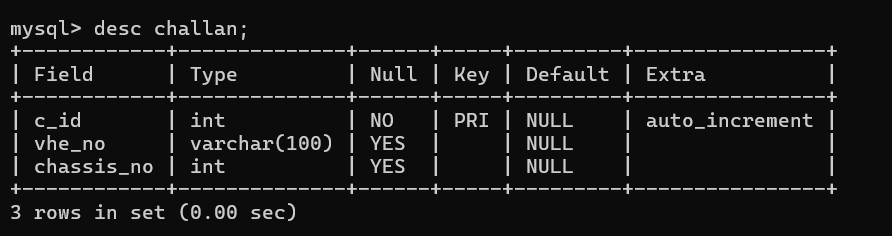
1. **Driving License**



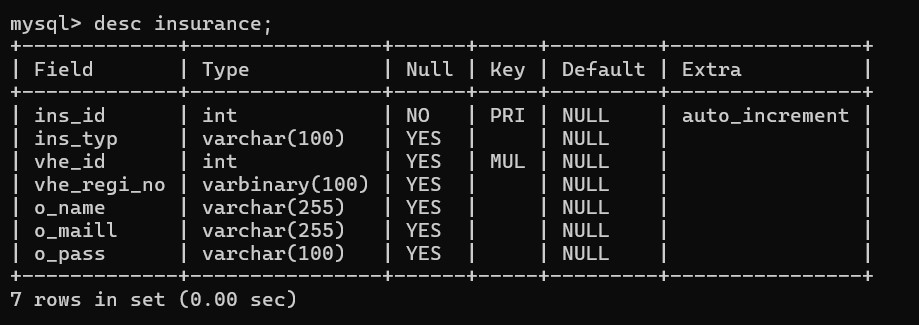
1. **Address change**



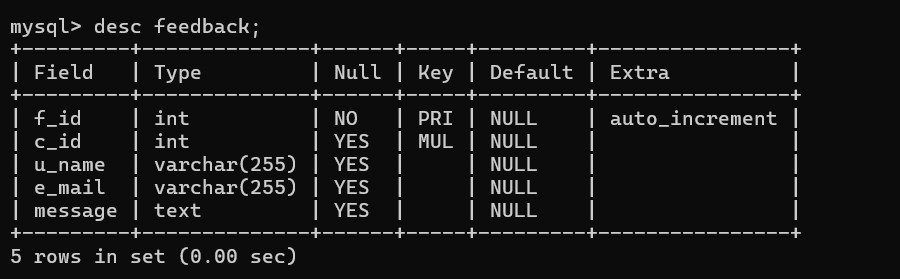
1. **Challan**



1. **Insurance Policy**



1. **Feedback**



1. **Creating a Database**

Using MySQL server, create a new database for your student management system. You can do this with SQL commands or through the graphical interface.

create database rto1;

1. **Using a Database**

Before performing any operations on a database, you need to select it using the USE statement:

USE rto1;

1. **Creating the tables for each entity**

USE rto1

* 1. **User**

create table users(u\_id int auto\_increment primary key, u\_name varchar(255)unique not null, u\_Email varchar(255) not null, u\_mobile varchar(22) not null, u\_dob varchar(22), u\_add varchar(500),

u\_pass varchar(100)not null

);

* 1. **Admin**

create table admin(a\_id int auto\_increment primary key, a\_name varchar(50) , a\_pass varchar(50)

);

* 1. **RTO Officer**

create table rtooffi(R\_id int auto\_increment primary key,

R\_name varchar(255),

R\_pass varchar(50),

R\_idcard\_no int

);

* 1. **Vehicle** create table vehicle (V\_id int auto\_increment primary key,

V\_regi\_no varchar(100), V\_user\_id int ,

foreign key(V\_user\_id) references users(u\_id), V\_owner\_name varchar(255),

v\_type int

);

* 1. **Driving / Learners License** create table DLL(

user\_id int auto\_increment primary key, foreign key(user\_id) references users(u\_id),

us\_name varchar(255), u\_age int check(u\_age>18), type\_of\_lic int

);

* 1. **Learning License** create table LL(ll\_id int auto\_increment primary key, ll\_name varchar(255), ll\_pass varchar(50), usr\_id int ,

foreign key(usr\_id) references users(u\_id)

);

* 1. **Driving License** create table DL(dl\_id int auto\_increment primary key, dl\_name varchar(255), dl\_add varchar(500), dl\_pass varchar(50), ll\_no int, ll\_id int ,

foreign key(ll\_id) references LL(ll\_id)

);

* 1. **Address change** create table addchange(Ad\_id int auto\_increment primary key, u\_name varchar(255), u\_pass varchar(50), dl\_no int, dl\_old\_add varchar (500), dl\_new\_add varchar(500)

);

* 1. **Challan** create table challan(c\_id int auto\_increment primary key, vhe\_no varchar(100),

chassis\_no int

);

* 1. **Insurance Policy** create table insurance(ins\_id int auto\_increment primary key, ins\_typ varchar(100), vhe\_id int,

foreign key(vhe\_id) references vehicle(V\_id), vhe\_regi\_no varbinary(100), o\_name varchar (255), o\_maill varchar (255),

o\_pass varchar(100)

);

* 1. **Feedback** create table feedback(f\_id int auto\_increment primary key, c\_id int,

u\_name varchar(255), e\_mail varchar (255), message text,

constraint fk\_LL foreign key(c\_id) references users(u\_id)

);