DSA 8660 Normalization Exersize

saransh rakshak

Contents

1. Functional Dependencies (FDs) Identification

From our data model, we can see the following functional dependencies:

1. **Driver Information FD** - DriverID uniquely identifies all driver-related attributes.
   - `DriverID -> {DriverFName, DriverLName, DateDriverJoinedCompany, DriverDOB}`
2. **Vehicle Information FD** - VehicleID uniquely identifies all vehicle-related attributes.
   - `VehicleID -> {VehicleMake, VehicleModel, VehiclePassangerCapacity}`
3. **Route Information FD** - RouteID uniquely identifies all route-related attributes.
   - `RouteID -> {RouteStartPoint, RouteEndPoint, RouteStandardDrivingTime}`
4. **Schedule Information FD** - A combination of ScheduleDate and ScheduledDepTime identifies the rest of the schedule details for specific departure time on particular date.
   - `(RouteID, ScheduleDate, ScheduledDepTime) -> {ScheduledArrTime, DriverID, VehicleID}`
5. **Driver Certification FD** - Combination of DriverID and VehicleID identifies if given driver is certified to operate specific vehicle.
   - `(DriverID, VehicleID) -> {DriverCertStartDate, DriverCertEndDate}`
6. **Bus Schedule FD** - Combination of RouteID, ScheduleDate, and ScheduledDepTime identifies the scheduled departure time for route on given date.
   - `(RouteID, ScheduleDate, ScheduledDepTime) -> {ScheduledArrTime}`
7. **Driver Assignment to Vehicle FD** - DriverID and VehicleID together identify who is assigned to specific vehicle.
   - `(DriverID, VehicleID) -> null` (implies a many-to-many relationship between driver and vehicle dependent on both IDs).

---

2. 3NF Relations Extraction

With our defined functional dependencies, we can fit the relations to follow constraints of Third Normal Form (3NF). This means any given attribute need not depend on another, unless it completely determines all other attributes of the entity.

1. **Driver Relation**
   - Attributes: `DriverID`, `DriverFName`, `DriverLName`, `DateDriverJoinedCompany`, `DriverDOB`
   - Primary Key: `DriverID`
2. **Vehicle Relation**
   - Attributes: `VehicleID`, `VehicleMake`, `VehicleModel`, `VehiclePassangerCapacity`

- Primary Key: `VehicleID`

3. **Route Relation**
   - Attributes: `RouteID`, `RouteStartPoint`, `RouteEndPoint`, `RouteStandardDrivingTime`
   - Primary Key: `RouteID`
4. **Schedule Relation**
   - Attributes: `RouteID`, `ScheduleDate`, `ScheduledDepTime`, `ScheduledArrTime`
   - Primary Key: `(RouteID, ScheduleDate, ScheduledDepTime)`
5. **DriverVehicleCertification Relation**
   - Attributes: `DriverID`, `VehicleID`, `DriverCertStartDate`, `DriverCertEndDate`
   - Primary Key: `(DriverID, VehicleID)`
6. **DriverAssignmentToVehicle Relation** (implicit for many-to-many)
   - Attributes: `DriverID`, `VehicleID`
   - Primary Key: `(DriverID, VehicleID)`

Summary of Relations

1. **REL_DRIVER**
   - PK: DriverID

   `DriverID | DriverFName | DriverLName | DateDriverJoinedCompany | DriverDOB`

2. **REL_VEHICLE**
   - PK: VehicleID

   `VehicleID | VehicleMake | VehicleModel | VehiclePassangerCapacity`

3. **REL_ROUTE**
   - PK: RouteID

   `RouteID | RouteStartPoint | RouteEndPoint | RouteStandardDrivingTime`

4. **REL_SCHEDULE**
   - PK: RouteID, ScheduleDate, ScheduleDepTime

   `RouteID | ScheduleDate | ScheduledDepTime | ScheduledArrTime | DriverID | VehicleID`

5. **REL_DRIVERVEHICLECERTIFICATION**
   - PK: DriverID, VehicleID

   `DriverID | VehicleID | DriverCertStartDate | DriverCertEndDate`

6. **REL_DRIVERASSIGNMENTTOVEHICLE** (implicit)

   `DriverID | VehicleID`

By organizing the data into normalized relations, we can ensure each attribute depends on only primary key of relation and not through another attribute, minimizes redundancy and bolsters data integrity.