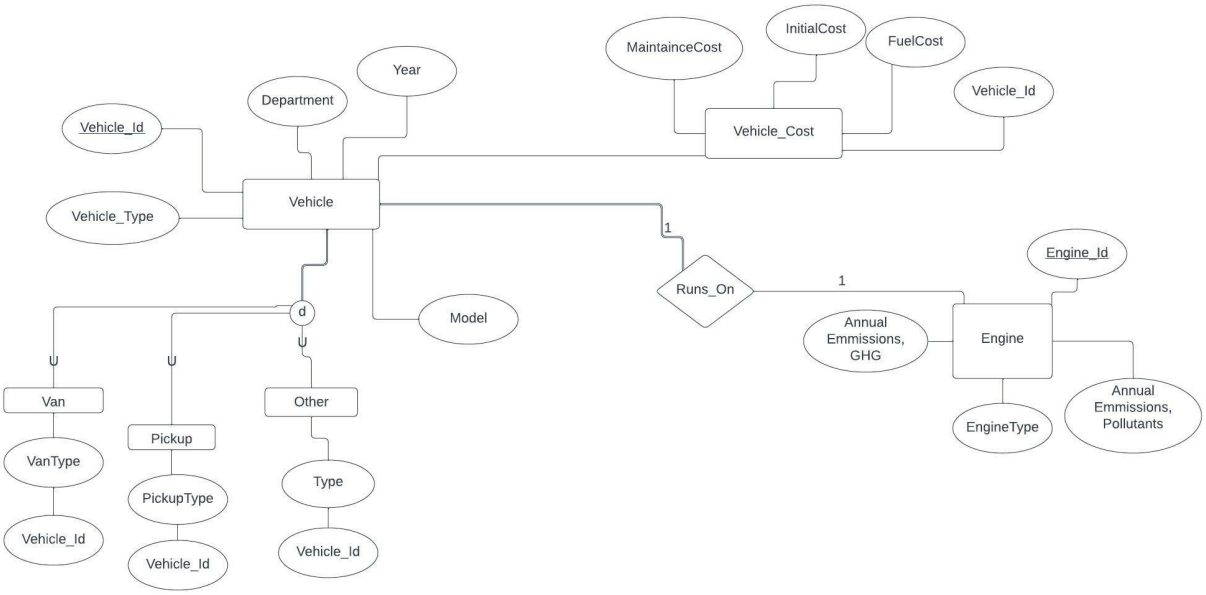
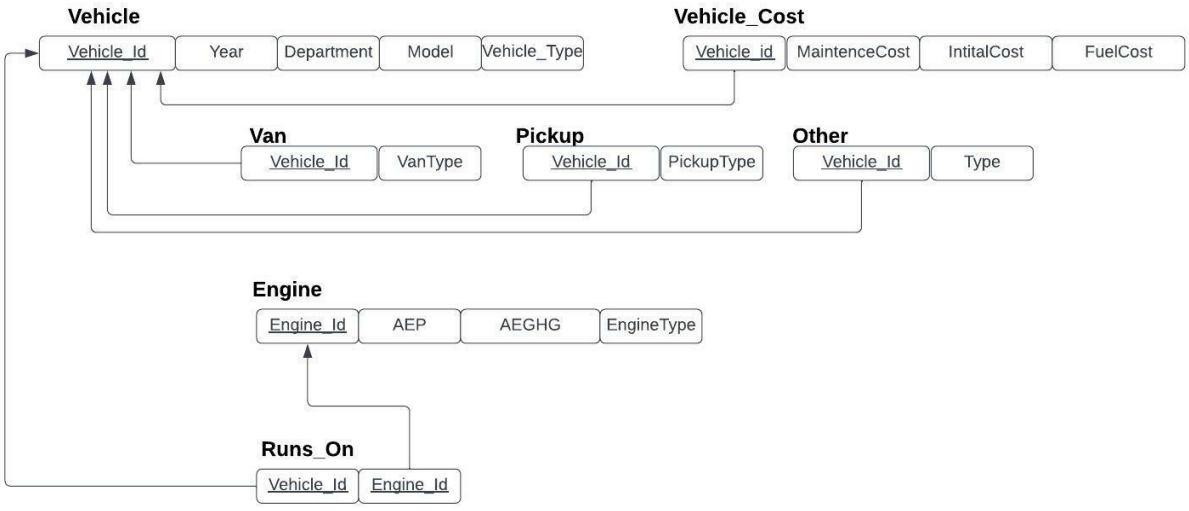
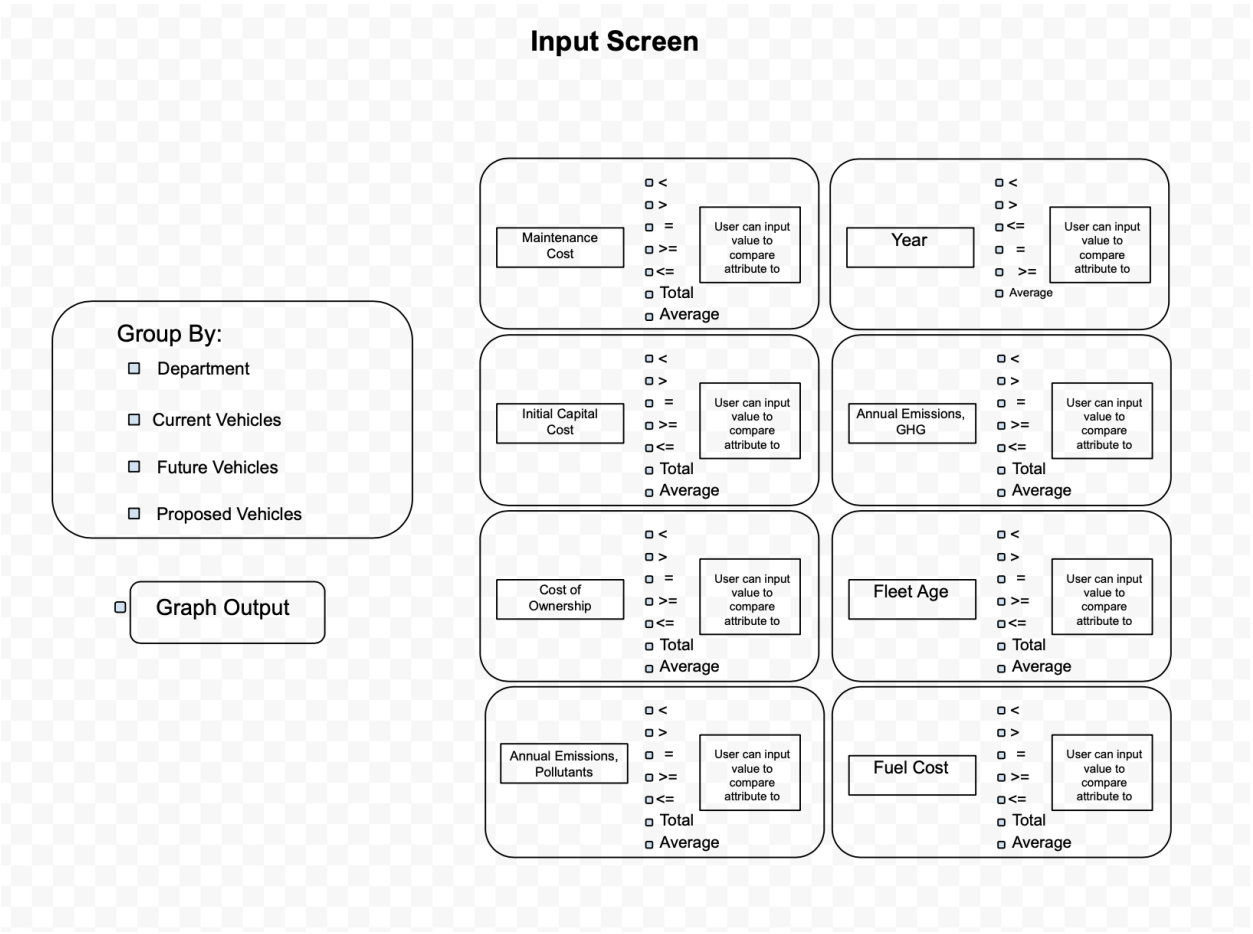
Stage IV - Elaboration: Database Design

Paula Arroyave, Faiza Hoque, and Matt Machado







BCNF:

a. Relations Van, Pickup, Other, FutureVehicle, ProposedVehicle, Runs_On, Prp_Runs_On, and Ftr_Runs_On are normalized to Boyce -Codd form since the relations stated only have one functional dependency without transitive dependencies, and whenever an FD X -> A holds in the relation, X is a superkey in the relation.
b. For relation Engine, we make Engine_Id and Engine_Type a candidate key and therefore AEP, AEGHG, and FuelCost are all dependent on the candidate key (Engine_Id, Engine_Type). This makes this relation BCNF.
c. For relation Vehicle, we make the candidate key Vehicle_ID. In order for this relation to be normalized into BCNF, Vehicle_ID,Department, Year, and Model will be in one relation and Model, IntialCost, FuelCost, and MaintenanceCost will be in another.

Views:

The view for MaintenanceCost would consist of a column with the Vehicle_ID for all of the vehicles that qualify and a column for MaintenanceCost that is less than, greater than, equal to, less than or equal to, or greater than or equal to the User input. Also, there can be a view where the average and total Maintenance cost is shown.

The view for Year would consist of a column with the Vehicle_ID for all of the vehicles that qualify and a column for Year that is less than, greater than, equal to, less than or equal to, or greater than or equal to the User input. Also, there can be a view where the average and total amount of Years is shown.

The view for Initial_Cost would consist of a column with the Vehicle_ID for all of the vehicles that qualify and a column for Intitial_Cost that is less than, greater than, equal to, less than or equal to, or greater than or equal to the User input. Also, there can be a view where the average and total Intital_Cost is shown.

The view for AEGHG would consist of a column with the Vehicle_ID for all of the vehicles that qualify and a column for AEGHG that is less than, greater than, equal to, less than or equal to, or greater than or equal to the User input. Also, there can be a view where the average and total AEGHG is shown.

The view for AEP would consist of a column with the Vehicle_ID for all of the vehicles that qualify and a column for AEP that is less than, greater than, equal to, less than or equal to, or greater than or equal to the User input. Also, there can be a view where the average and total AEP is shown.

The view for FuelCost would consist of a column with the Vehicle_ID for all of the vehicles that qualify and a column for FuelCost that is less than, greater than, equal to, less than or equal to, or greater than or equal to the User input. Also, there can be a view where the average and total FuelCost is shown.

The view for CostofOwnership would consist of a column with the Vehicle_ID for all of the vehicles that qualify and a column for CostofOwnership that is less than, greater than, equal to, less than or equal to, or greater than or equal to the User input. Also, there can be a view where the average and total CostofOwnership is shown.

The view for FleetAge consists of a column with Vehicle_ID for all vehicles that qualify and a column for FleetAge that is less than, greater than, equal to, less than or equal to, or greater than or equal to the UserInput. ALso there can be a view where see the average or total FleetAge.

These views columns can be combined whena user wants to project multiple attributes. This occurs when the user selects a boolean expression for more than one attribute. For example you can have a view with the columns MaintenanceCost, Vehicle_Id, AEP, AEGHG.

There can also be a view where the information is grouped by department, future, current, or proposed vehicles.

Queries:

```
SELECT MaintenanceCost, Vehicle_Id
FROM Vehicle_Cost
WHERE MaintenanceCost < UserInput

SELECT MaintenanceCost, Vehicle_Id
FROM Vehicle_Cost
WHERE MaintenanceCost > UserInput

SELECT MaintenanceCost, Vehicle_Id
FROM Vehicle_Cost
WHERE MaintenanceCost = UserInput

SELECT MaintenanceCost, Vehicle_Id
FROM Vehicle_Cost
WHERE MaintenanceCost <= UserInput

SELECT MaintenanceCost, Vehicle_Id
FROM Vehicle_Cost
WHERE MaintenanceCost >= UserInput

SELECT  AVG (MaintenanceCost) AS AvgMaintenceCost, Vehicle_Id
FROM Vehicle_Cost

SELECT SUM (MaintenanceCost) AS TotalMaintenceCost, Vehicle_Id
FROM Vehicle_Cost

SELECT Year, Vehicle_Id
FROM Vehicle
WHERE Year < UserInput

SELECT Year, Vehicle_Id
FROM Vehicle
WHERE Year > UserInput

SELECT Year, Vehicle_Id
FROM Vehicle
WHERE Year = UserInput

SELECT Year, Vehicle_Id
FROM Vehicle
WHERE Year <= UserInput
```

```
SELECT Year, Vehicle_Id
FROM Vehicle
WHERE Year >= UserInput

SELECT AVG (Year) AS AvgYear, Vehicle_Id
From Vehicle


SELECT InitialCost, Vehicle_Id
FROM Vehicle_Cost
WHERE InitialCost < UserInput

SELECT InitialCost, Vehicle_Id
FROM Vehicle_Cost
WHERE InitialCost > UserInput

SELECT InitialCost, Vehicle_Id
FROM Vehicle_Cost
WHERE InitialCost <= UserInput

SELECT InitialCost, Vehicle_Id
FROM Vehicle_Cost
WHERE InitialCost >= UserInput

SELECT InitialCost, Vehicle_Id
FROM Vehicle_Cost
WHERE InitialCost = UserInput

SELECT SUM (InitialCost) as TotalInitlCost, Vehicle_Id
FROM Vehicle_Cost

SELECT AVG (InitialCost) as AvgInitCost, Vehicle_Id
FROM Vehicle_Cost

SELECT AEGHG, Vehicle_Id
FROM Engine
WHERE AEGHG < UserInput

SELECT AEGHG, Vehicle_Id
FROM Engine
WHERE AEGHG > UserInput

SELECT AEGHG, Vehicle_Id
FROM Engine
```

```
WHERE AEGHG <= UserInput

SELECT AEGHG, Vehicle_Id
FROM Engine
WHERE AEGHG >= UserInput

SELECT AEGHG, Vehicle_Id
FROM Engine
WHERE AEGHG = UserInput

SELECT SUM (AEGHG) as TotalAEGHG, Vehicle_Id
FROM Engine

SELECT AVG (AEGHG) as AvgAEGHG, Vehicle_Id
FROM Engine

SELECT AEP, Vehicle_Id
FROM Engine
WHERE AEP < UserInput

SELECT AEP, Vehicle_Id
FROM Engine
WHERE AEP > UserInput

SELECT AEP, Vehicle_Id
FROM Engine
WHERE AEP <= UserInput

SELECT AEP, Vehicle_Id
FROM Engine
WHERE AEP >= UserInput

SELECT AEP, Vehicle_Id
FROM Engine
WHERE AEP = UserInput

SELECT SUM (AEP) AS TotalAEP, Vehicle_Id
FROM Engine

SELECT AVG (AEP) AS AvgAEP, Vehicle_Id
FROM Engine


SELECT FuelCost, Vehicle_Id
```

```
FROM Vehicle_Cost
WHERE FuelCost < UserInput

SELECT FuelCost, Vehicle_Id
FROM Vehicle_Cost
WHERE FuelCost > UserInput

SELECT FuelCost, Vehicle_Id
FROM Vehicle_Cost
WHERE FuelCost <= UserInput

SELECT FuelCost, Vehicle_Id
FROM Vehicle_Cost
WHERE FuelCost >= UserInput


SELECT FuelCost, Vehicle_Id
FROM Vehicle_Cost
WHERE FuelCost = UserInput

SELECT SUM (FuelCost) AS TotalFuelCost, Vehicle_Id
FROM Vehicle_Cost

SELECT AVG (FuelCost) as AvgFuelCost, Vehicle_Id
FROM Vehicle_Cost


SELECT Vehicle_Id, (MaintenanceCost + InitialCost + FuelCost) AS OwnershipCost
FROM Vehicle_Cost
WHERE OwnershipCost < UserInput

SELECT Vehicle_Id, (MaintenanceCost + InitialCost + FuelCost) AS OwnershipCost
FROM Vehicle_Cost
WHERE OwnershipCost > UserInput

SELECT Vehicle_Id, (MaintenanceCost + InitialCost + FuelCost) AS OwnershipCost
FROM Vehicle_Cost
WHERE OwnershipCost <= UserInput

SELECT Vehicle_Id, (MaintenanceCost + InitialCost + FuelCost) AS OwnershipCost
FROM Vehicle_Cost
WHERE OwnershipCost >= UserInput

SELECT Vehicle_Id, (MaintenanceCost + InitialCost + FuelCost) AS OwnershipCost
```

FROM Vehicle_Cost
WHERE OwnershipCost = UserInput

SELECT SUM Vehicle_Id, (MaintenanceCost + InitialCost + FuelCost) AS TotalOwnershipCost
FROM Vehicle_Cost

SELECT AVG Vehicle_Id, (MaintenanceCost + InitialCost + FuelCost) AS AvgOwnershipCost
FROM Vehicle_Cost

SELECT Vehicle_Id, ((EXTRACT(YEAR FROM CURRENT_DATE)) - Year) AS Age
From Vehicle
Where Age < UserInput

SELECT Vehicle_Id, ((EXTRACT(YEAR FROM CURRENT_DATE)) - Year) AS Age
From Vehicle
Where Age > UserInput

SELECT Vehicle_Id, ((EXTRACT(YEAR FROM CURRENT_DATE)) - Year) AS Age
From Vehicle
Where Age = UserInput

SELECT Vehicle_Id, ((EXTRACT(YEAR FROM CURRENT_DATE)) - Year) AS Age
From Vehicle
Where Age >= UserInput

SELECT Vehicle_Id, ((EXTRACT(YEAR FROM CURRENT_DATE)) - Year) AS Age
From Vehicle
Where Age <= UserInput

SELECT Vehicle_Id, SUM(((EXTRACT(YEAR FROM CURRENT_DATE)) - Year)) AS TotalAge
From Vehicle

SELECT Vehicle_Id, AVG(((EXTRACT(YEAR FROM CURRENT_DATE)) - Year)) AS AvgAge
From Vehicle

These selects can be joined together if a user wants to project multiple attributes. This occurs when the user selects a boolean expression for more than one attribute.

EX:
SELECT MaintenanceCost, Vehicle_Id, AEP, AEGHG
FROM Vehicle_Cost NATURAL JOIN Engine
WHERE MaintenanceCost < UserInput, AEP > UserInput, AEGHG <= AEGHG

These selects can also be grouped by department, current, future, or proposed vehicles

EX:
SELECT MaintenanceCost, Vehicle_Id, AEP, AEGHG
FROM Vehicle NATURAL JOIN Vehicle_Cost NATURAL JOIN Engine
WHERE MaintenanceCost < UserInput, AEP > UserInput, AEGHG <= AEGHG
GROUP BY Department