A repair order is used to track a repair. Each repair order has a number that is generated by the system, an origination date, and a completion date. A repair order has any number of notes that are used to describe the complaints, as well as document the course of a repair. A repair order is for a vehicle. After the car is diagnosed, the mechanic will decide what procedures are needed to repair it. The available procedures are stored in the information system as procedure definitions.

In some cases, a repair estimate is prepared. In other cases, a repair is done directly. In any case, an invoice of line items required for the repair is prepared with a description, quantity, and price for each line item. A procedure may have many line items and a line item pertains to a specific procedure. The invoice is printed either as an estimate or as a final bill when the repair is complete and payable. There can be multiple invoices per repair order, including partial billing or pre-payment, as well as multiple estimates.

Identification of the relations (entity types):

Entity name	Description	Aliases	Occurrence
RepairOrder	General term describing all repair order information.	Repair Order	Each repair order is specific to one particular vehicle. A particular vehicle can have many repair orders over time
Procedure	General term describing all procedures that can be done on a vehicle.	Service	A particular procedure can be used on many different repair orders, but doesn't have to be used on any. A repair order has at least one, but possibly many different procedures.
LineItem	General term describing all purchases (parts or labor) needed to complete a procedure.	Line Item	Any given line item can be present on 0 or more (possibly many) invoices. A single invoice contains at least one, but possibly many line items. A procedure may have many line items and a line item pertains to a specific procedure
LineItemPart	Specialization of LineItem	Parts	A Part is a type of line item
LineItemLabor	Specialization of LineItem	Labor	Labor is a type of line item
Invoice	General term describing the collection of all line items for a transaction.	Invoice, Bill	Each invoice is specific to one repair order. A repair order can have multiple invoices.
InvoiceEstimate	Specialization of Invoice, an	Estimate	An Estimate is a type of invoice.

	estimated cost of repairs.		
InvoiceCost	Specialization of Invoice, a payable bill.	Cost	A Cost is a type of invoice.
Vehicle	General term describing a vehicle being repaired.	Vehicle, Automobile, Car	Each vehicle can have many repair orders, a repair order is specific to one vehicle.
Mechanic	General term describing the staff member choosing the procedures.	Staff	Each mechanic can choose many procedures or none, any given procedure can be chosen by many mechanics or none. A mechanic can fulfill many repair orders or none, each repair order must be fulfilled by exactly one mechanic.

Identification of relationship types and their participation and cardinality constraints:

Entity Name	Multiplicity	Relationship	Multiplicity	Entity Name
RepairOrder	0..* 0..* 1..1	Fixes Documents Generates	1..1 1..* 1..*	**Vehicle** **Procedure** **Invoice**
Procedure	0..*	Requires	1..*	**LineItem**
Invoice	0..*	Lists	1..*	**LineItem**
Mechanic	0..* 1..1	Chooses Fulfills	0..* 0..*	**Procedure** **RepairOrder**

Logical Data Model

Identification of attributes and association of attributes with entity or relationship types:

Entity or relationship name	Attributes	Description	Data Type & Length	Nulls	Multi-valued
RepairOrder	repairNo (PK)	Uniquely identifies repair orders	VARCHAR 10	No	No
	originDate	Date when invoice is created.	DATE 10	No	No
	completeDate	Date when repair completes.	DATE 10	No	No
	complaintNote	Document containing complaints about vehicle.	TEXT	No	No
	vehicleNo (FK)	Identity of vehicle receiving repairs	VARCHAR 10	No	No
	staffNo (FK)	Identity of mechanic(s) in charge of repairs	VARCHAR 10	No	No
Procedure	procedureNo (PK)	Uniquely identifies procedures	VARCHAR 10	No	No
	procedureDescription	Gives a description of procedure	TEXT	No	No
LineItem	itemNo (PK)	Uniquely identifies a line item	VARCHAR 10	No	No
	itemName	Name of line item	VARCHAR 50	No	No
	description	Describes the line item	TEXT	No	No
	procedureNo (FK)	Identify the procedure the line item is used for.	VARCHAR 10	No	No
				No	No
				No	No
LineItemPart	price	Gives the price of the line item part.	DECIMAL (7,2)	No	No
LineItemLabor	rate	Gives hourly labor rate	DECIMAL (5,2)	No	No
	hours	Gives number of hours required for labor	DECIMAL (5,2)	No	No
			DECIMAL (7,2)	No	No

	price (derived as rate * hours)	Gives the price of the labor, derived from hourly rate and number hours.			
Invoice	invoiceNo (PK)	Uniquely identifies aninvoice.	VARCHAR 10	No	No
	odometerStart	Odometer reading when repairs start.	INTEGER	No	No
	invoicePrintDate	Date that the invoice is printed	INTEGER	No	No
	repairNo (FK)	Reference to RepairOrder the invoice belongs to.	VARCHAR 10	No	No
InvoiceEstimate					
InvoiceCost	odometerFinish	Odometer reading when repairs finish.	INTEGER	No	No
	paymentDate	Date the invoice is paid for.	DATE	Yes	No
Vehicle	vin (PK)	Vehicle vin number	VARCHAR 17	No	No
	make	Make of the vehicle	VARCHAR 50	No	No
	year	Year of the vehicle	INTEGER	No	No
	model	Model of the vehicle	VARCHAR 50	No	No
Mechanic	staffNo (PK)	Uniquely identifies a mechanic	VARCHAR 10	No	No
Lists	quantity	Indicates the quantity of a particular line item in the relationship Invoice Lists LineItem.	INTEGER	Yes	No
Documents	repairDocument	Document outlining the repair process.	TEXT	No	No

Determination of candidate and primary key attributes of entity types:

Entity	Primary Key	Candidate Keys
RepairOrder	repairNo	N/A
Procedure	procedureNo	N/A
LineItem	itemNo	itemName
Invoice	invoiceNo	N/A
Vehicle	vin	N/A
Mechanic	staffNo	N/A

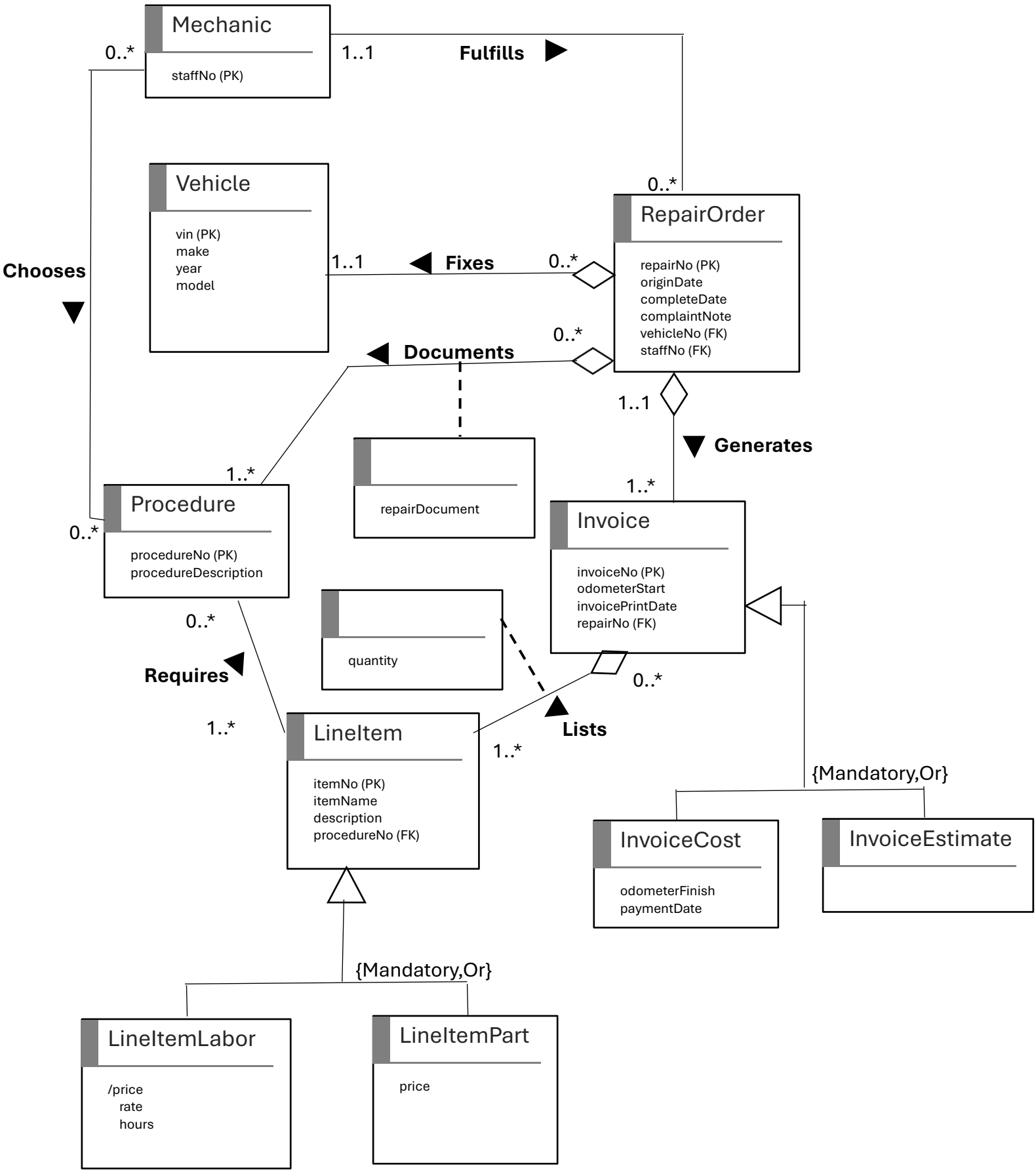
Determination of specialization/generalization and categorization relationships:

Specialization/Generalization Relationship:

- **Invoice (Mandatory and Disjoint)**
  - **Estimate**
  - **Final Bill**
- **Line Item (Mandatory and Disjoint)**
  - **Parts Line Item**
  - **Labor Line Item**

Aggregation Relationship:

- **Repair Order**
  - **Invoice**
  - **Procedure**
  - **Vehicle**
- **Invoice**
  - **Line Item**


Mechanic
staffNo (PK)
0..*
1..1
Fulfills
Vehicle
vin (PK)
make
year
model
Chooses
1..1
Fixes
0..*
RepairOrder
repairNo (PK)
originDate
completeDate
complaintNote
vehicleNo (FK)
staffNo (FK)
0..*
Documents
0..*
1..1
Generates
repairDocument
1..*
Procedure
procedureNo (PK)
procedureDescription
0..*
1..*
Invoice
invoiceNo (PK)
odometerStart
invoicePrintDate
repairNo (FK)
quantity
0..*
Requires
1..*
Lists
1..*
LineItem
itemNo (PK)
itemName
description
procedureNo (FK)
{Mandatory,Or}
InvoiceCost
odometerFinish
paymentDate
InvoiceEstimate
{Mandatory,Or}
LineItemLabor
/price
rate
hours
LineItemPart
price

Conceptual Data Model

Derivation of relations from the refined conceptual model:

Strong entity types:

RepairOrder(repairNo, originDate, completeDate, complaintNote**)**
Primary key: repairNo

Procedure(procedureNo, procedureDescription**)**
Primary key: procedureNo

LineItem(itemNo, itemName, description**)**
Primary key: itemNo
Alternate key: itemName

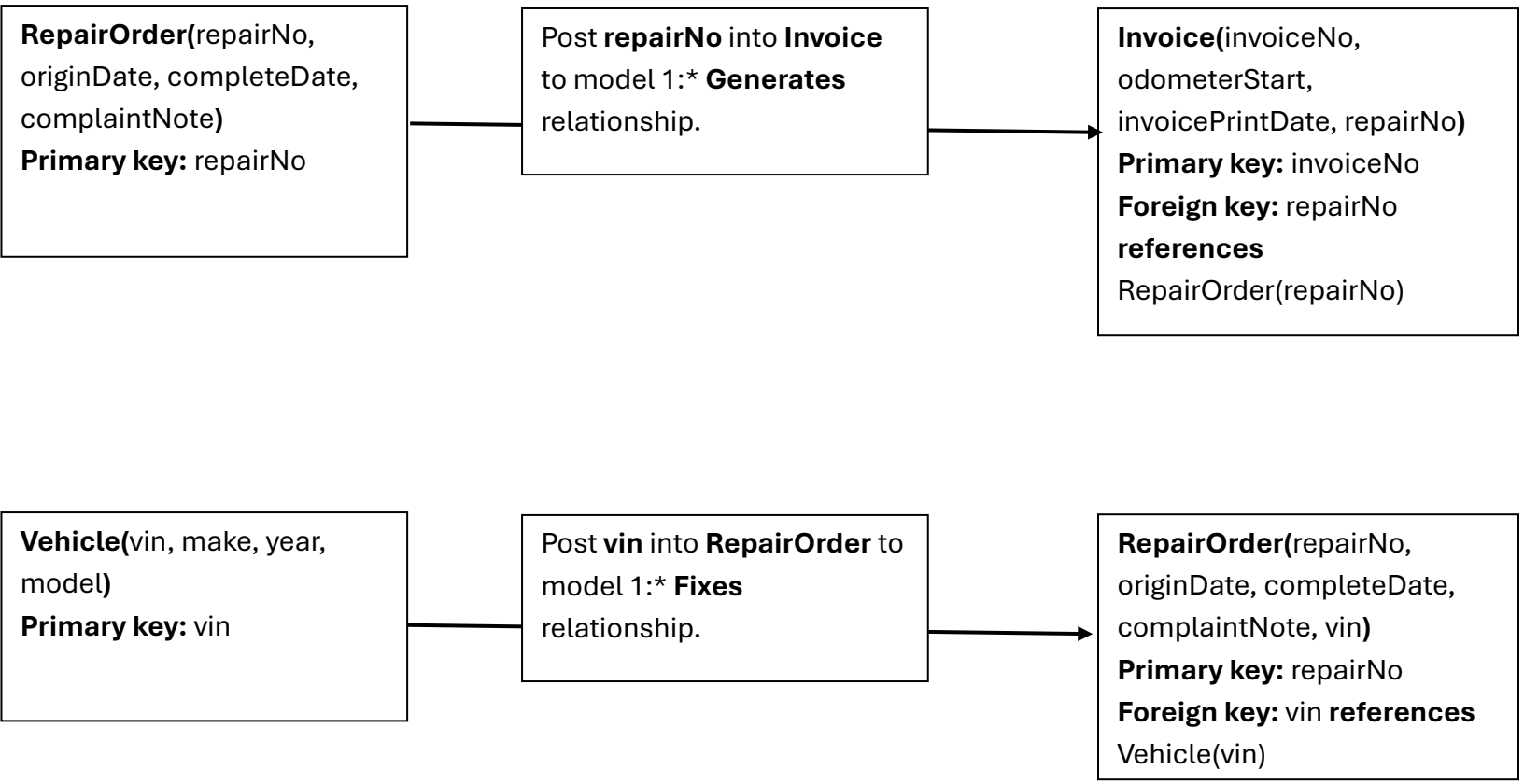
Invoice(invoiceNo, odometerStart, invoicePrintDate**)**
Primary key: invoiceNo

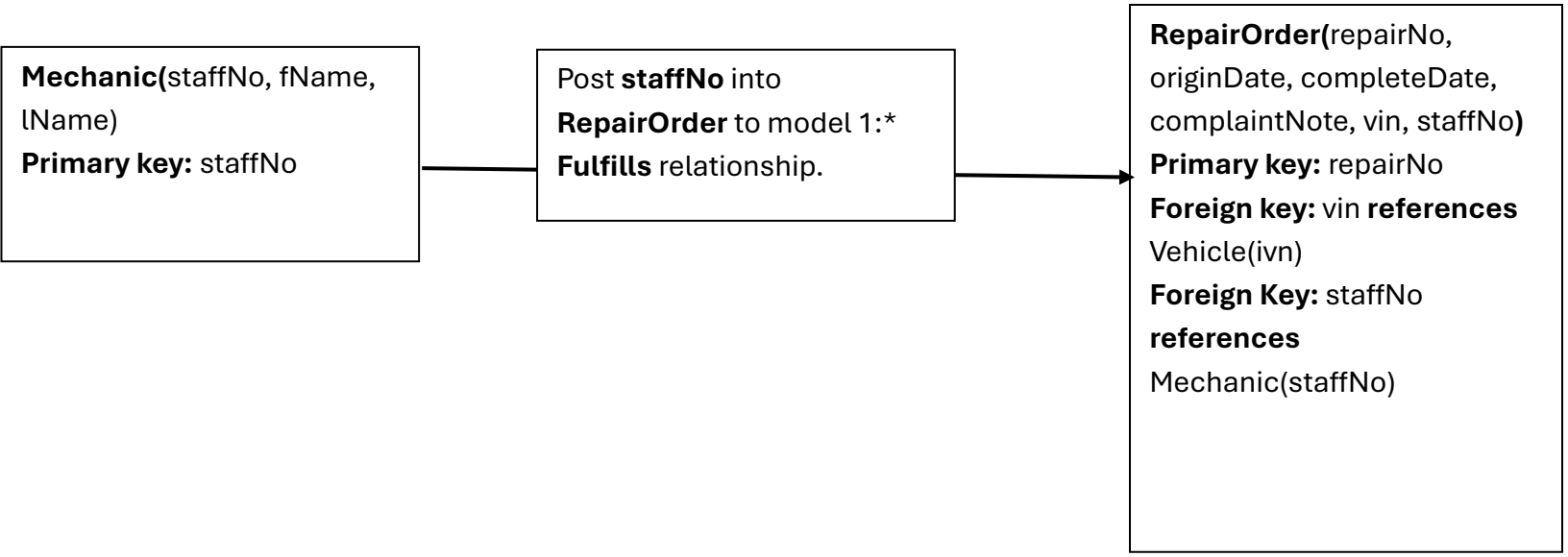
Vehicle(vin, make, year, model**) Changed from conceptual model, vin is unique and more easily accessed as primary key than vehicleNo.**
Primary key: vin

Mechanic(staffNo, fName, lName)
Primary key: staffNo

Weak Entity Types:
None

One-to-many (1:*) binary relationship types:





<u>One-to-one (1:1) binary relationship types:</u>

a) **Mandatory participation on both sides:**

None

b) **Mandatory Participation on one side:**

None

c) **Optional participation on both sides:**

None

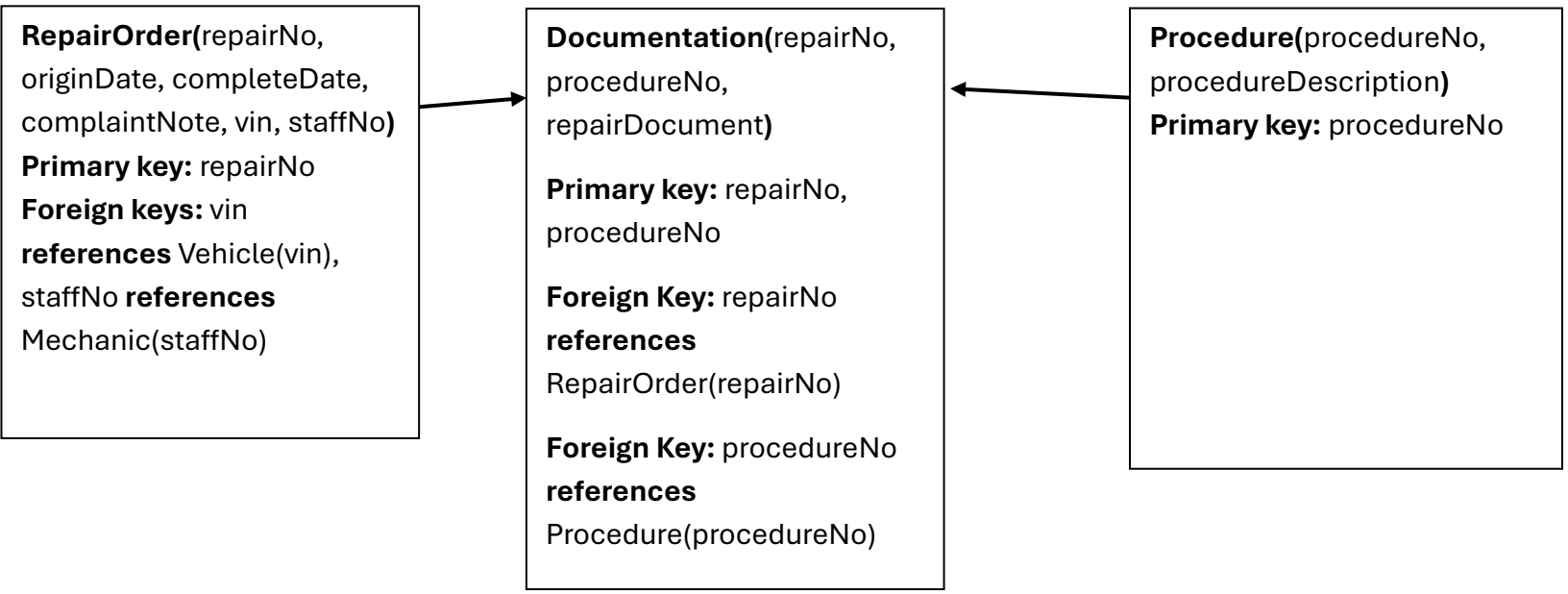
<u>One-to-one recursive relationships:</u>
None

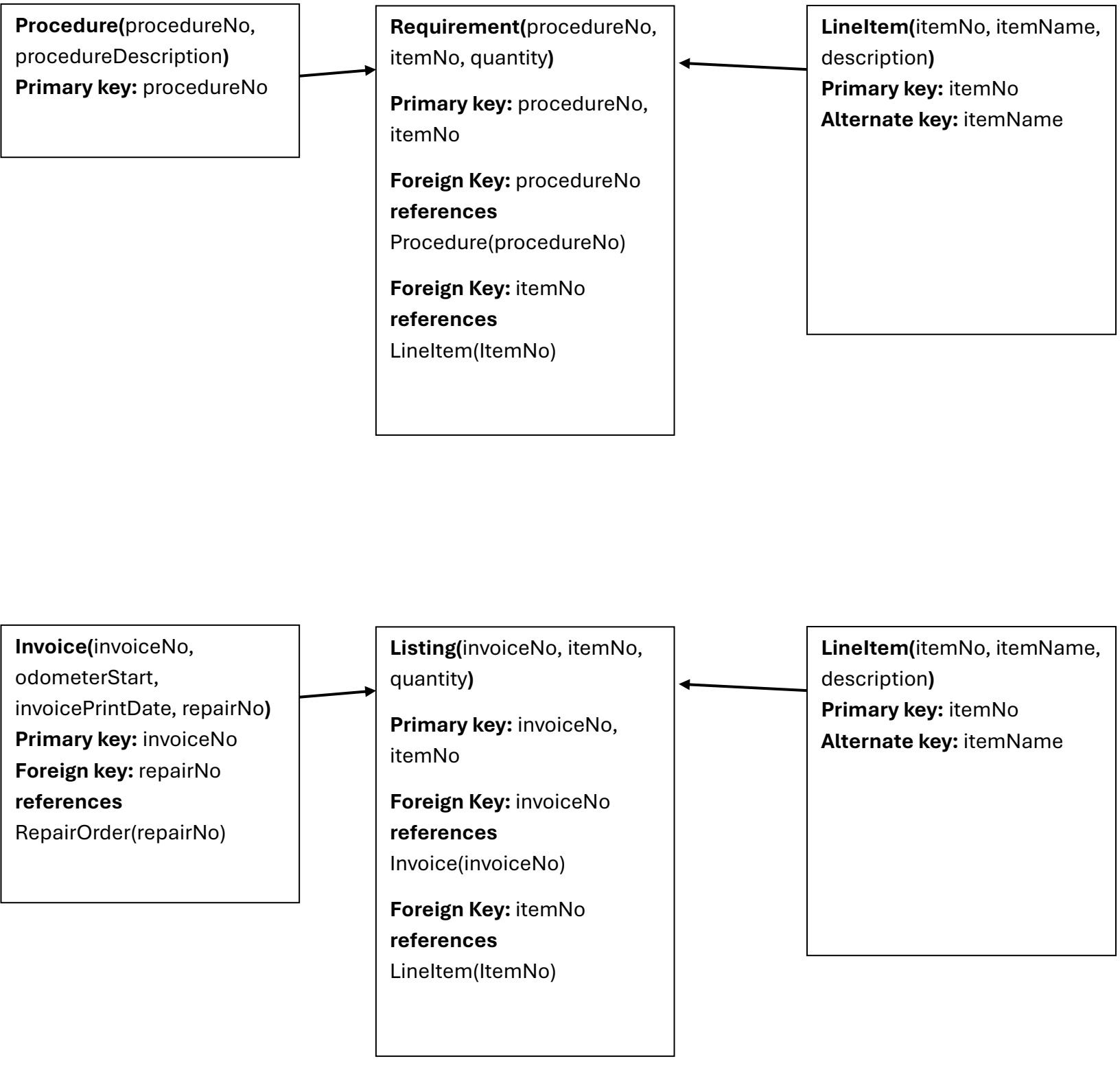
<u>Superclass/Subclass Relationship Types:</u>
Mandatory, Disjoint:

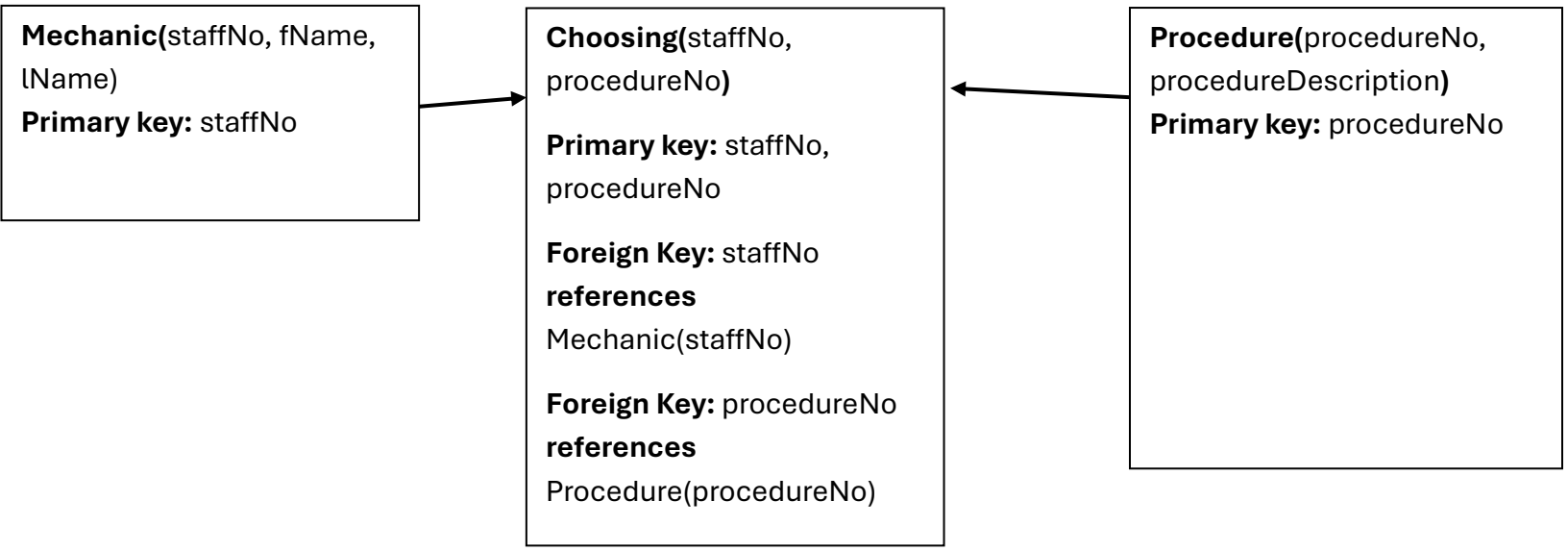
LineItemLabor(itemNo, rate, hours, itemName, description**)**
Primary key: itemNo

LineItemPart(itemNo, price, itemName, description**)**
Primary key: itemNo

<u>Many-to-many (*:*) binary relationship types:</u>




Procedure(procedureNo, procedureDescription**)**
Primary key: procedureNo
Requirement(procedureNo, itemNo, quantity**)**
Primary key: procedureNo, itemNo
Foreign Key: procedureNo **references** Procedure(procedureNo)
Foreign Key: itemNo **references** LineItem(ItemNo)
LineItem(itemNo, itemName, description**)**
Primary key: itemNo
Alternate key: itemName
Invoice(invoiceNo, odometerStart, invoicePrintDate, repairNo**)**
Primary key: invoiceNo
Foreign key: repairNo **references** RepairOrder(repairNo)
Listing(invoiceNo, itemNo, quantity**)**
Primary key: invoiceNo, itemNo
Foreign Key: invoiceNo **references** Invoice(invoiceNo)
Foreign Key: itemNo **references** LineItem(ItemNo)
LineItem(itemNo, itemName, description**)**
Primary key: itemNo
Alternate key: itemName



Complex relationship types:

None

Relations:

RepairOrder(repairNo, originDate, completeDate, complaintNote, vin, staffNo**)** **Primary key:** repairNo **Foreign key:** vin **references** Vehicle(vin) **Foreign Key:** staffNo **references** Mechanic(staffNo)	**Documentation(**repairNo, procedureNo, repairDocument**)** **Primary key:** repairNo, procedureNo **Foreign Key:** repairNo **references** RepairOrder(repairNo) **Foreign Key:** procedureNo **references** Procedure(procedureNo)
Procedure(procedureNo, procedureDescription**)** **Primary key:** procedureNo	**Requirement(**procedureNo, itemNo, quantity**)** **Primary key:** procedureNo, itemNo **Foreign Key:** procedureNo **references** Procedure(procedureNo) **Foreign Key:** itemNo **references** LineItem(ItemNo)

LineItem(itemNo, itemName, description**)** **Primary key:** itemNo **Alternate key:** itemName	**Listing(**invoiceNo, itemNo, quantity**)** **Primary key:** invoiceNo, itemNo **Foreign Key:** invoiceNo **references** Invoice(invoiceNo) **Foreign Key:** itemNo **references** LineItem(ItemNo)
Invoice(invoiceNo, odometerStart, invoicePrintDate, repairNo**)** **Primary key:** invoiceNo **Foreign key:** repairNo **references** RepairOrder(repairNo)	**Choice(**staffNo, procedureNo**)** **Primary key:** staffNo, procedureNo **Foreign Key:** staffNo **references** Mechanic(staffNo) **Foreign Key:** procedureNo **references** Procedure(procedureNo)
Vehicle(vin, make, year, model**)** **Primary key:** vin	
Mechanic(staffNo, fName, lName) **Primary key:** staffNo	

Multi-valued attributes:

None

Validate relations using normalization:

Repair Order:

repairNo -> originDate, completeDate, complaintNote, vin, staffNo

1NF: No repeating groups

2NF: Every non-primary-key attribute is fully functionally dependent on primary key **repairNo** (no partial functional dependency).

3NF: No non-primary-key attribute is transitively dependent on the primary key **repairNo**.

BCNF: Determinant is candidate key (primary key)

Procedure

procedureNo -> procedureDescription

1NF: No repeating groups
2NF: The non-primary-key attribute is functionally dependent on **procedureNo** (no partial functional dependency).
3NF: The non-primary-key attribute is transitively dependent
on the primary key **procedureNo**.
BCNF: Determinant is candidate key (primary key)

LineItem
itemNo -> itemName, description
1NF: No repeating groups
2NF: Every non-primary-key attribute is fully functionally dependent on primary key **itemNo** (no partial functional dependency)
3NF: No non-primary-key attribute is transitively dependent
on the primary key **itemNo**.
BCNF: Determinant is candidate key (primary key)

Invoice
invoiceNo -> odometerStart, invoicePrintDate, repairNo
1NF: No repeating groups
2NF: Every non-primary-key attribute is fully functionally dependent on primary key **invoiceNo** (no partial functional dependency). Primary key is not composite.
3NF: No non-primary-key attribute is transitively dependent
on the primary key **invoiceNo**.
BCNF: Determinant is candidate key (primary key)

Vehicle
vin -> make, year, model
1NF: No repeating groups
2NF: Every non-primary-key attribute is fully functionally dependent on primary key **vin** (no partial functional dependency). Primary key is not composite.

3NF: No non-primary-key attribute is transitively dependent
on the primary key **vin**.
BCNF: Determinant is candidate key (primary key).

Mechanic
staffNo -> fName, lName
1NF: No repeating groups
2NF: Every non-primary-key attribute is fully functionally dependent on primary key **staffNo** (no partial functional dependency). Primary key is not composite
3NF: No non-primary-key attribute is transitively dependent
on the primary key **staffNo**.
BCNF: Determinant is candidate key (primary key)

Documentation
repairNo, procedureNo -> repairDocument
1NF: No repeating groups
2NF: Every non-primary-key attribute is fully functionally dependent on primary key **repairNo, procedureNo** (no partial functional dependency). **repairDocument** is specific to the combination of **repairNo** and **procedureNo.**
3NF: The non-primary-key attribute is not transitively dependent
on the primary key **repairNo, procedureNo**.
BCNF: Determinant is candidate key (primary key)

Requirement
procedureNo, itemNo -> quantity
1NF: No repeating groups

2NF: Every non-primary-key attribute is fully functionally dependent on primary key **procedureNo, itemNo** (no partial functional dependency). **quantity** is specific to the combination of **procedureNo** and **itemNo.**
3NF: The non-primary-key attribute is not transitively dependent
on the primary key **procedureNo, itemNo**.
BCNF: Determinant is candidate key (primary key)

Listing
invoiceNo, itemNo -> quantity
1NF: No repeating groups
2NF: Every non-primary-key attribute is fully functionally dependent on primary key **invoiceNo, itemNo** (no partial functional dependency). **quantity** is specific to the combination of **invoiceNo** and **itemNo.**
3NF: The non-primary-key attribute is not transitively dependent
on the primary key **invoiceNo, itemNo**.
BCNF: Determinant is candidate key (primary key)

Choosing
staffNo, procedureNo
1NF: No repeating groups
2NF: There is nothing to depend on. No non-primary attributes. No partial functional dependency.
3NF: No non-primary-key attributes, no transitive dependency.
BCNF: No determinant, just primary key.

Validation of logical model against corresponding user transactions:

Repair Order:
A **Repair Order** is used to track a repair. Each repair order has a number that is generated by the system, an origination date, and a completion date. A repair order has any number of notes that are used to describe the complaints, as well as document the course of a repair. A repair order is for a **Vehicle**.

Mechanic:
After the car is diagnosed, the **Mechanic** will decide what **Procedures** are needed to repair it.

Procedure:
The available procedures are stored in the information system as procedure definitions. Some procedure definitions include: Replacing starter motor; Replacing accelerator cable; State inspection; Lube, oil, and oil filter; Replacing tail light bulb; Replacing head light bucket; Repairing and replacing Pitman arm; and Servicing and repairing transmission.

Invoice:
In some cases, a repair estimate is prepared. In other cases, a repair is done directly. In any case, an **Invoice** of **Line Items** required for the repair is prepared with a description, quantity, and price for each line item. The invoice is printed either as an **estimate** or as a **final bill** when the repair is complete and payable. There can be multiple invoices per repair order, including partial billing or pre-payment, as well as multiple estimates. The invoice includes the following information: Invoice number, which is listed on the document handed to the customer. Odometer mileage when the vehicle comes in and when it is finished. Date when the repair originated (from the repair order). Date when the repair is complete (from the repair order). Date when the invoice is printed. Date paid.

Line Item:

A **Procedure** may have many line items and a **Line Item** pertains to a specific procedure. Examples of line items include: Starter; Labor for starter replacement; Towing; Accelerator cable; Labor for replacing the accelerator cable; Bulb; Labor for lube, oil, and oil filter; Oil filter; Headlight adjusting screw; and State safety inspection.

Definition of integrity constraints:

Primary Key Constraints/ Entity Integrity:

RepairOrder
Primary Key Constraints/ Entity Integrity:
Primary key: repairNo must be unique and not null.
Foreign Key Constraints/Referential Integrity
Foreign key: vin **references** Vehicle(vin) ON UPDATE CASCADE ON DELETE CASCADE
Foreign Key: staffNo **references** Mechanic(staffNo) ON UPDATE CASCADE ON DELETE NO ACTION
To Insert a staffNo into the **RepairOrder** relation, it must already exist in the **Mechanic** relation.

Procedure
Primary Key Constraints/ Entity Integrity:
Primary key: procedureNo must be unique and not null.

LineItem
Primary Key Constraints/ Entity Integrity:
Primary key: itemNo must be unique and not null.

Invoice
Primary Key Constraints/ Entity Integrity:
Primary key: invoiceNo must be unique and not null.
Foreign Key Constraints/Referential Integrity
Foreign key: repairNo **references** RepairOrder(repairNo) ON UPDATE CASCADE ON DELETE CASCADE

Vehicle
Primary Key Constraints/ Entity Integrity:
Primary key: vin must be unique and not null.

Mechanic
Primary Key Constraints/ Entity Integrity:
Primary key: staffNo must be unique and not null.

Documentation
Primary Key Constraints/ Entity Integrity:
Primary key: repairNo, procedureNo must be unique and not null.
Foreign Key Constraints/Referential Integrity
Foreign Key: repairNo **references** RepairOrder(repairNo) ON UPDATE CASCADE ON DELETE CASCADE
Foreign Key: procedureNo **references** Procedure(procedureNo) ON UPDATE CASCADE ON DELETE NO ACTION

Requir
Primary Key Constraints/ Entity Integrity:
Primary key: procedureNo, itemNo must be unique and not null.
Foreign Key Constraints/Referential Integrity
Foreign Key: procedureNo **references** Procedure(procedureNo) ON UPDATE CASCADE ON DELETE CASCADE
Foreign Key: itemNo **references** LineItem(ItemNo) ON UPDATE CASCADE ON DELETE NO ACTION

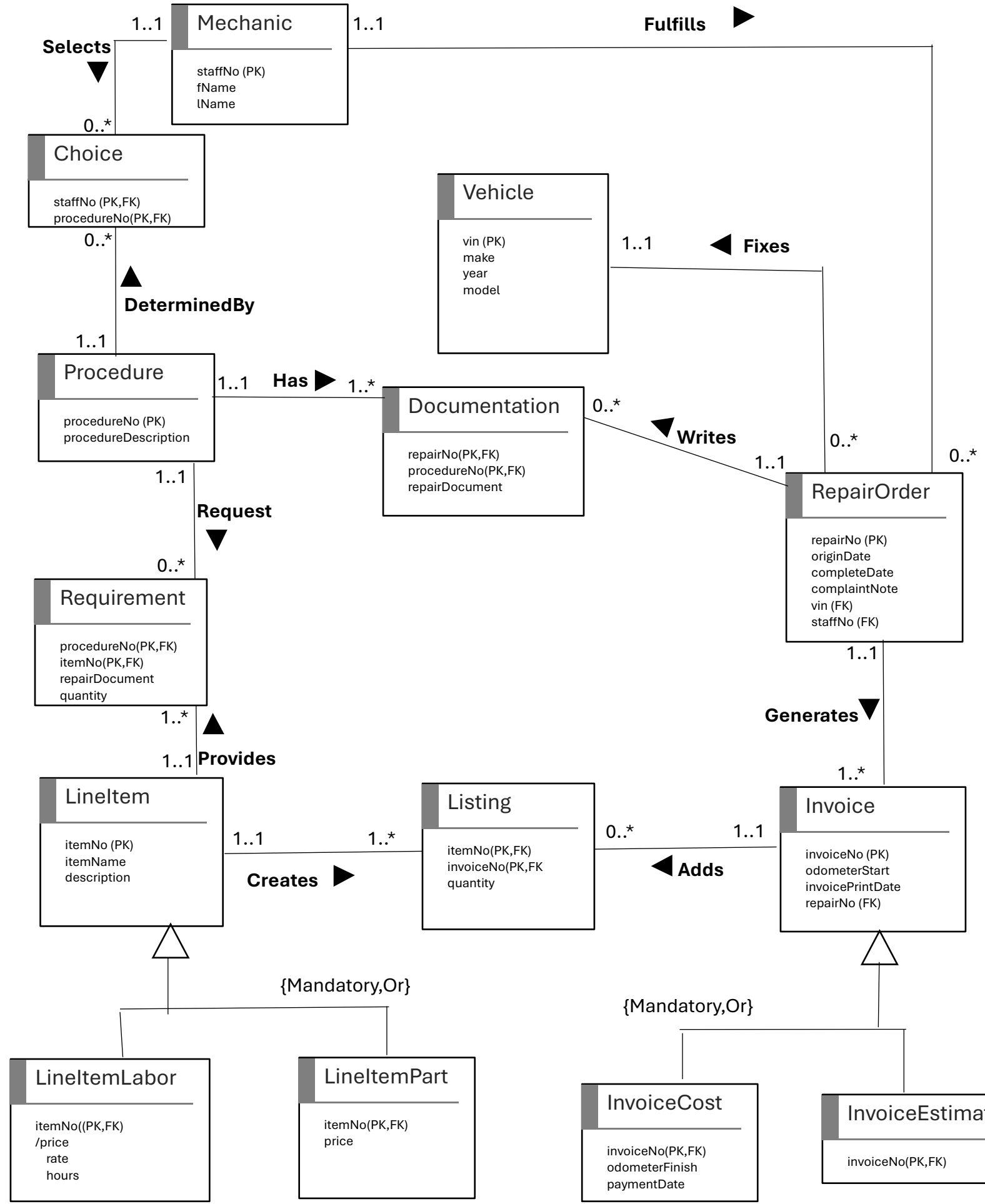
Listing
Primary Key Constraints/ Entity Integrity:
Primary key: invoiceNo, itemNo must be unique and not null.
Foreign Key Constraints/Referential Integrity
Foreign Key: invoiceNo **references** Invoice(invoiceNo) ON UPDATE CASCADE ON DELETE CASCADE

Foreign Key: itemNo **references** LineItem(ItemNo) ON UPDATE CASCADE ON DELETE NO ACTION

Choosing
Primary Key Constraints/ Entity Integrity:
Primary key: staffNo, procedureNo must be unique and not null.
Foreign Key Constraints/Referential Integrity
Foreign Key: staffNo **references** Mechanic(staffNo) ON UPDATE CASCADE ON DELETE CASCADE
Foreign Key: procedureNo **references** Procedure(procedureNo) ON UPDATE CASCADE ON DELETE NO ACTION

General Constraints:
General constraints could be added, but there are none specified in the instructions. For example, there could be constraints implemented limiting the number of Repair Orders a single mechanic could have active at one time.

UPDATED EER DIAGRAM for Conceptual Model:

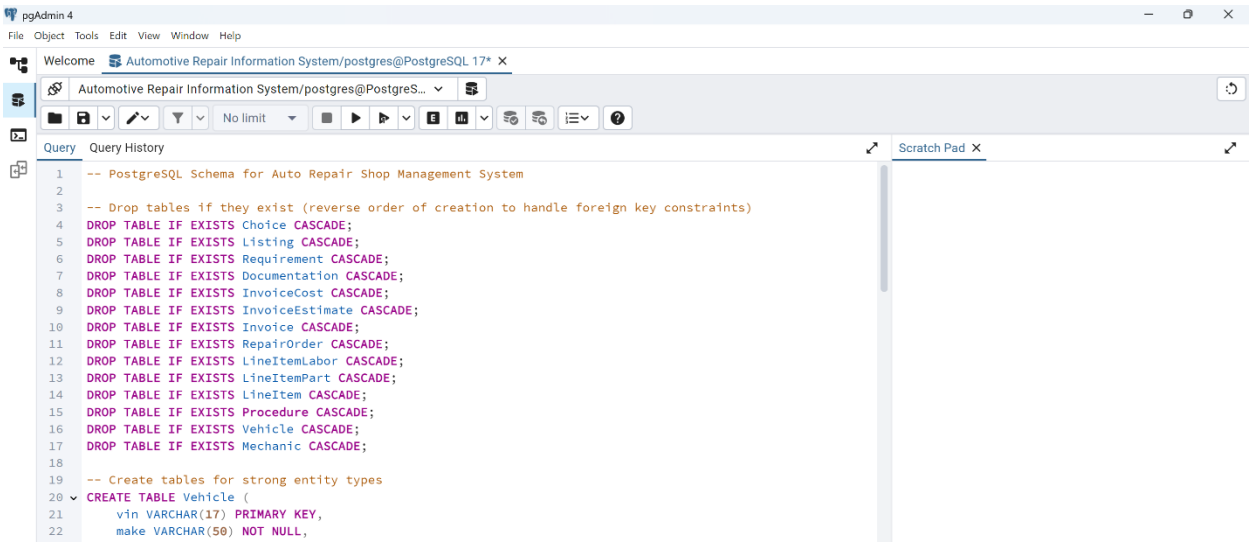

Mechanic
staffNo (PK)
fName
lName
Selects
1..1
0..*
Fulfills
1..1
Choice
staffNo (PK,FK)
procedureNo(PK,FK)
Vehicle
vin (PK)
make
year
model
Fixes
1..1
DeterminedBy
0..*
1..1
Procedure
procedureNo (PK)
procedureDescription
Has
1..1
1..*
Documentation
repairNo(PK,FK)
procedureNo(PK,FK)
repairDocument
Writes
0..*
1..1
0..*
RepairOrder
repairNo (PK)
originDate
completeDate
complaintNote
vin (FK)
staffNo (FK)
Request
1..1
0..*
Requirement
procedureNo(PK,FK)
itemNo(PK,FK)
repairDocument
quantity
Generates
1..1
1..*
Provides
1..*
1..1
LineItem
itemNo (PK)
itemName
description
Creates
1..1
1..*
Listing
itemNo(PK,FK)
invoiceNo(PK,FK
quantity
Adds
0..*
1..1
Invoice
invoiceNo (PK)
odometerStart
invoicePrintDate
repairNo (FK)
{Mandatory,Or}
{Mandatory,Or}
LineItemLabor
itemNo((PK,FK)
/price
rate
hours
LineItemPart
itemNo(PK,FK)
price
InvoiceCost
invoiceNo(PK,FK)
odometerFinish
paymentDate
InvoiceEstimate
invoiceNo(PK,FK)

SQL Statements and screenshots:

Table Creation

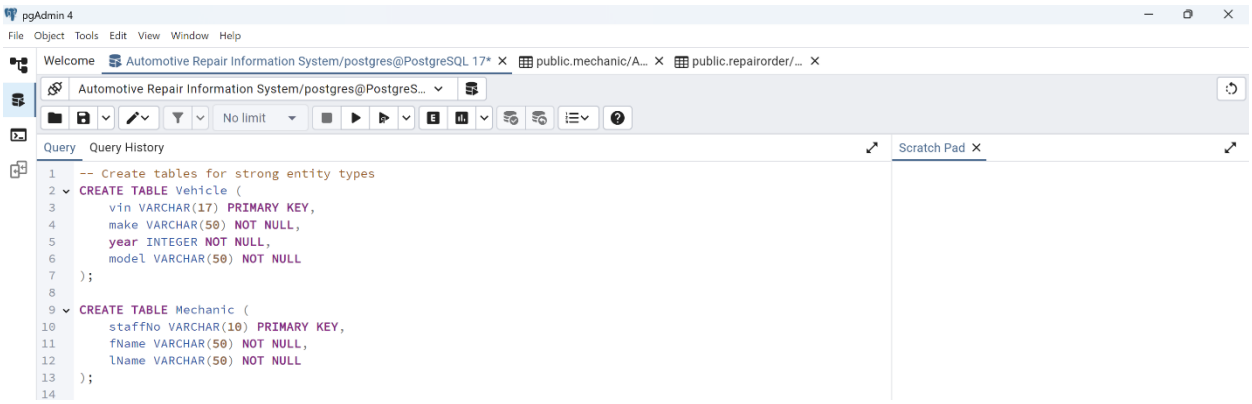
-- PostgreSQL Schema for Auto Repair Shop Management System

-- Drop tables if they exist (reverse order of creation to handle foreign key constraints)
DROP TABLE IF EXISTS Choice CASCADE;
DROP TABLE IF EXISTS Listing CASCADE;
DROP TABLE IF EXISTS Requirement CASCADE;
DROP TABLE IF EXISTS Documentation CASCADE;
DROP TABLE IF EXISTS InvoiceCost CASCADE;
DROP TABLE IF EXISTS InvoiceEstimate CASCADE;
DROP TABLE IF EXISTS Invoice CASCADE;
DROP TABLE IF EXISTS RepairOrder CASCADE;
DROP TABLE IF EXISTS LineItemLabor CASCADE;
DROP TABLE IF EXISTS LineItemPart CASCADE;
DROP TABLE IF EXISTS LineItem CASCADE;
DROP TABLE IF EXISTS Procedure CASCADE;
DROP TABLE IF EXISTS Vehicle CASCADE;
DROP TABLE IF EXISTS Mechanic CASCADE;



-- Create tables for strong entity types
CREATE TABLE Vehicle (
  vin VARCHAR(17) PRIMARY KEY,
  make VARCHAR(50) NOT NULL,

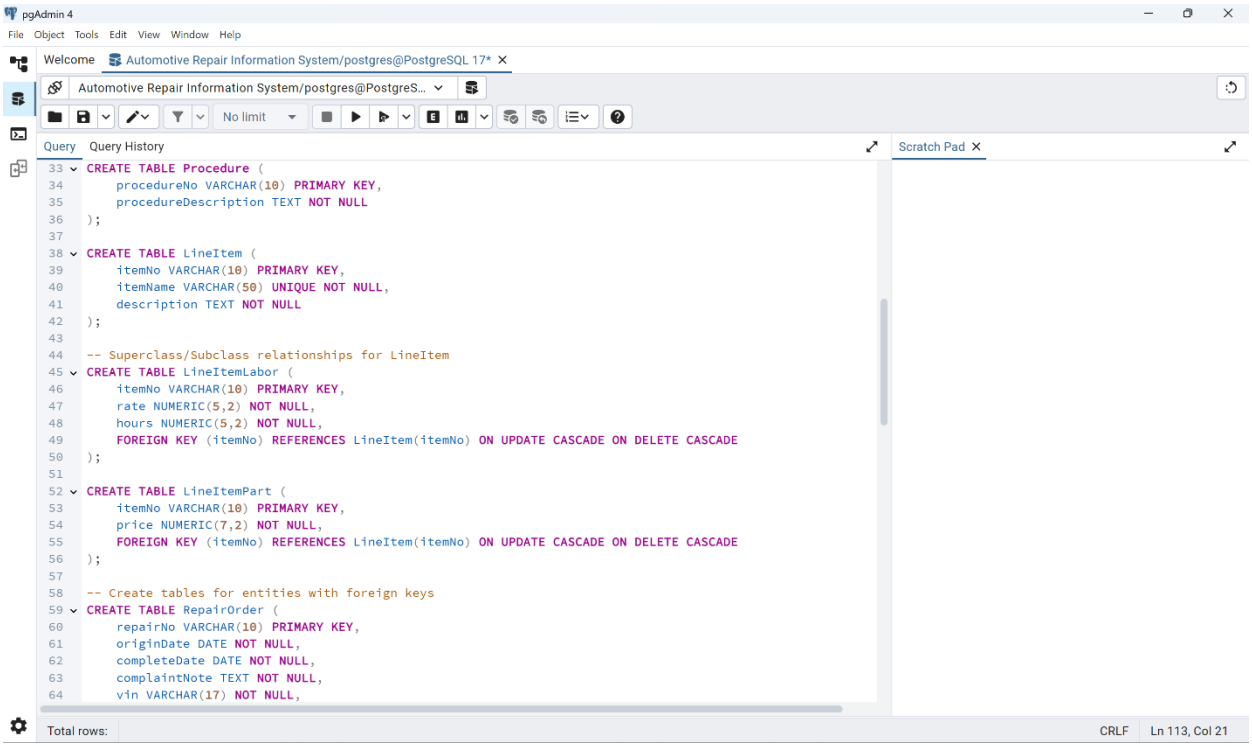
```
  year INTEGER NOT NULL,
  model VARCHAR(50) NOT NULL
);

CREATE TABLE Mechanic (
  staffNo VARCHAR(10) PRIMARY KEY,
  fName VARCHAR(50) NOT NULL,
  lName VARCHAR(50) NOT NULL
);
```




```
CREATE TABLE Procedure (
  procedureNo VARCHAR(10) PRIMARY KEY,
  procedureDescription TEXT NOT NULL
);

CREATE TABLE LineItem (
  itemNo VARCHAR(10) PRIMARY KEY,
  itemName VARCHAR(50) UNIQUE NOT NULL,
  description TEXT NOT NULL
);

-- Superclass/Subclass relationships for LineItem
CREATE TABLE LineItemLabor (
  itemNo VARCHAR(10) PRIMARY KEY,
  rate NUMERIC(5,2) NOT NULL,
  hours NUMERIC(5,2) NOT NULL,
  FOREIGN KEY (itemNo) REFERENCES LineItem(itemNo) ON UPDATE CASCADE ON
DELETE CASCADE
```

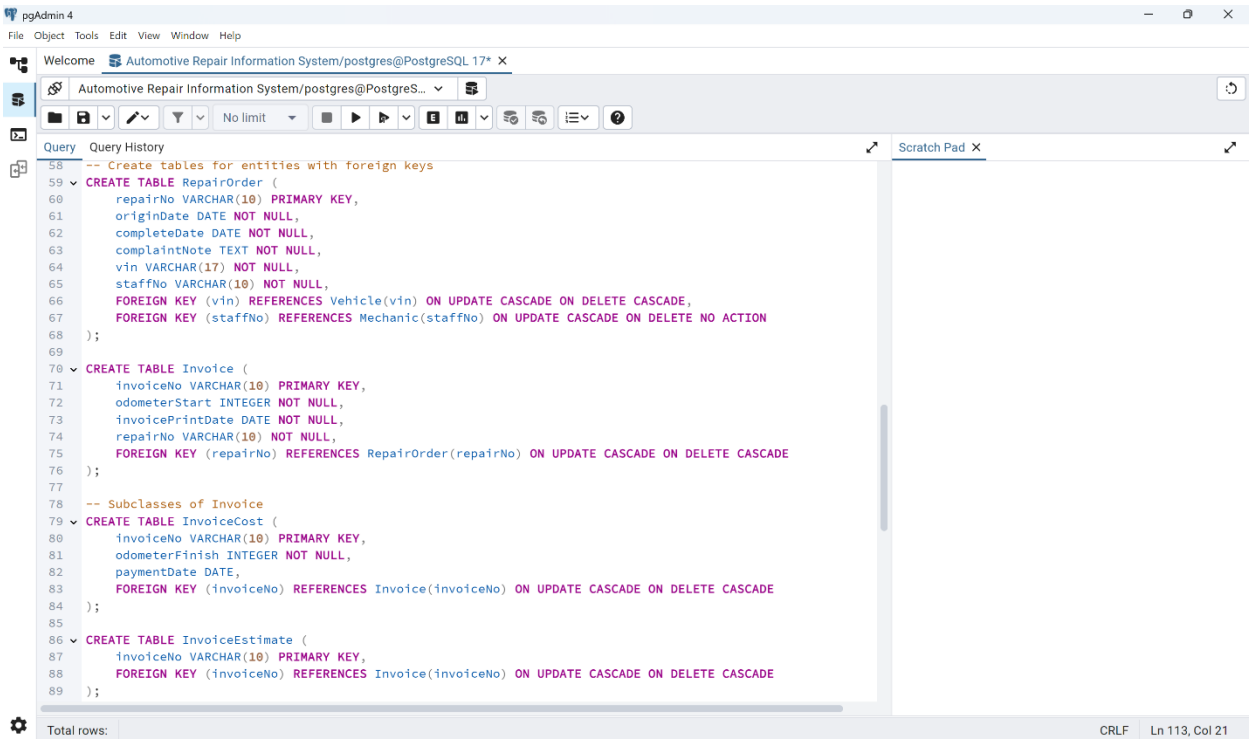
);

CREATE TABLE LineItemPart (
  itemNo VARCHAR(10) PRIMARY KEY,
  price NUMERIC(7,2) NOT NULL,
  FOREIGN KEY (itemNo) REFERENCES LineItem(itemNo) ON UPDATE CASCADE ON DELETE CASCADE
);



-- Create tables for entities with foreign keys
CREATE TABLE RepairOrder (
  repairNo VARCHAR(10) PRIMARY KEY,
  originDate DATE NOT NULL,
  completeDate DATE NOT NULL,
  complaintNote TEXT NOT NULL,
  vin VARCHAR(17) NOT NULL,
  staffNo VARCHAR(10) NOT NULL,
  FOREIGN KEY (vin) REFERENCES Vehicle(vin) ON UPDATE CASCADE ON DELETE CASCADE,

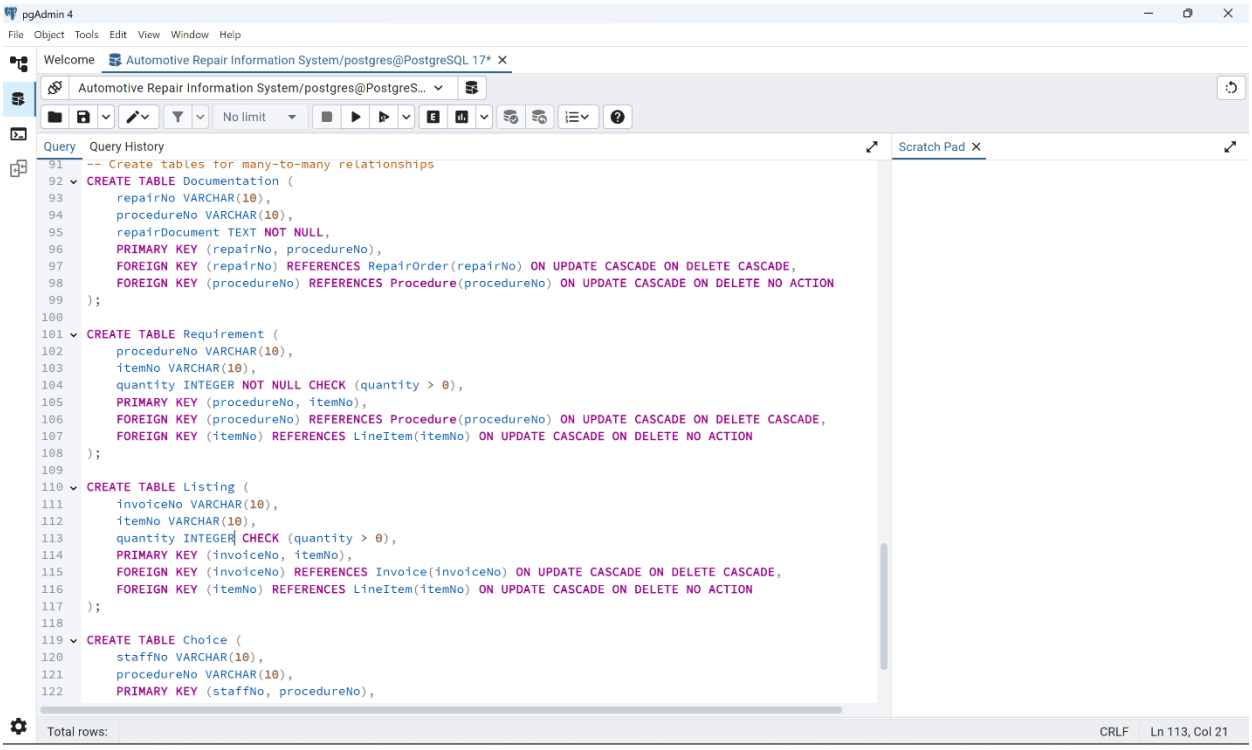
```
  FOREIGN KEY (staffNo) REFERENCES Mechanic(staffNo) ON UPDATE CASCADE ON
DELETE NO ACTION
);

CREATE TABLE Invoice (
  invoiceNo VARCHAR(10) PRIMARY KEY,
  odometerStart INTEGER NOT NULL,
  invoicePrintDate DATE NOT NULL,
  repairNo VARCHAR(10) NOT NULL,
  FOREIGN KEY (repairNo) REFERENCES RepairOrder(repairNo) ON UPDATE CASCADE ON
DELETE CASCADE
);

-- Subclasses of Invoice
CREATE TABLE InvoiceCost (
  invoiceNo VARCHAR(10) PRIMARY KEY,
  odometerFinish INTEGER NOT NULL,
  paymentDate DATE,
  FOREIGN KEY (invoiceNo) REFERENCES Invoice(invoiceNo) ON UPDATE CASCADE ON
DELETE CASCADE
);

CREATE TABLE InvoiceEstimate (
  invoiceNo VARCHAR(10) PRIMARY KEY,
  FOREIGN KEY (invoiceNo) REFERENCES Invoice(invoiceNo) ON UPDATE CASCADE ON
DELETE CASCADE
);
```



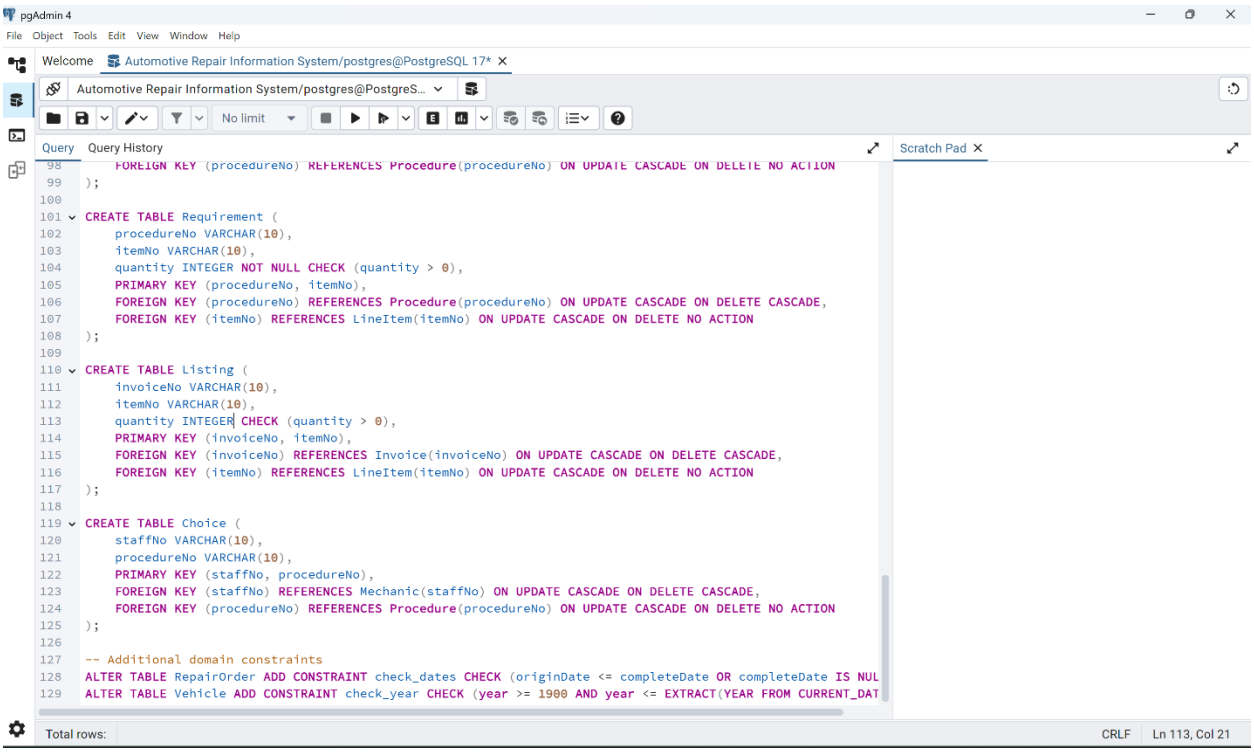

```
-- Create tables for many-to-many relationships
CREATE TABLE Documentation (
  repairNo VARCHAR(10),
  procedureNo VARCHAR(10),
  repairDocument TEXT NOT NULL,
  PRIMARY KEY (repairNo, procedureNo),
  FOREIGN KEY (repairNo) REFERENCES RepairOrder(repairNo) ON UPDATE CASCADE ON DELETE CASCADE,
  FOREIGN KEY (procedureNo) REFERENCES Procedure(procedureNo) ON UPDATE CASCADE ON DELETE NO ACTION
);

CREATE TABLE Requirement (
  procedureNo VARCHAR(10),
  itemNo VARCHAR(10),
  quantity INTEGER NOT NULL CHECK (quantity > 0),
  PRIMARY KEY (procedureNo, itemNo),
  FOREIGN KEY (procedureNo) REFERENCES Procedure(procedureNo) ON UPDATE CASCADE ON DELETE CASCADE,
  FOREIGN KEY (itemNo) REFERENCES LineItem(itemNo) ON UPDATE CASCADE ON DELETE NO ACTION
```

);

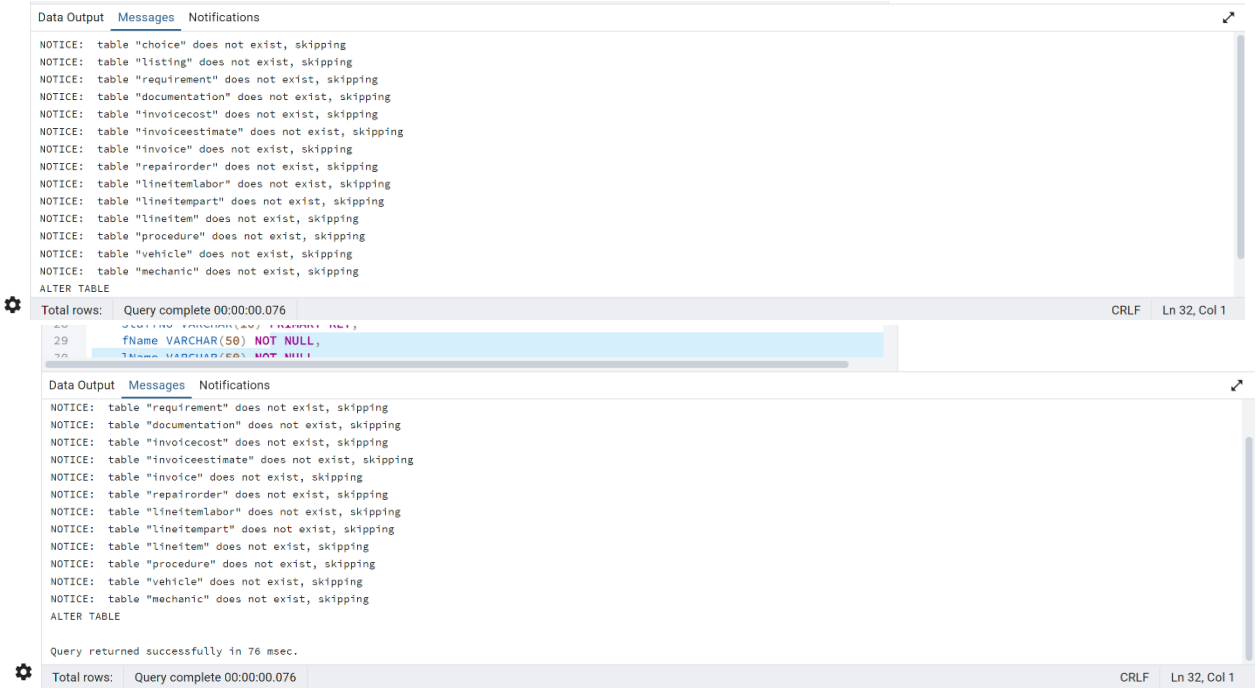
CREATE TABLE Listing (
  invoiceNo VARCHAR(10),
  itemNo VARCHAR(10),
  quantity INTEGER CHECK (quantity > 0),
  PRIMARY KEY (invoiceNo, itemNo),
  FOREIGN KEY (invoiceNo) REFERENCES Invoice(invoiceNo) ON UPDATE CASCADE ON DELETE CASCADE,
  FOREIGN KEY (itemNo) REFERENCES LineItem(itemNo) ON UPDATE CASCADE ON DELETE NO ACTION
);

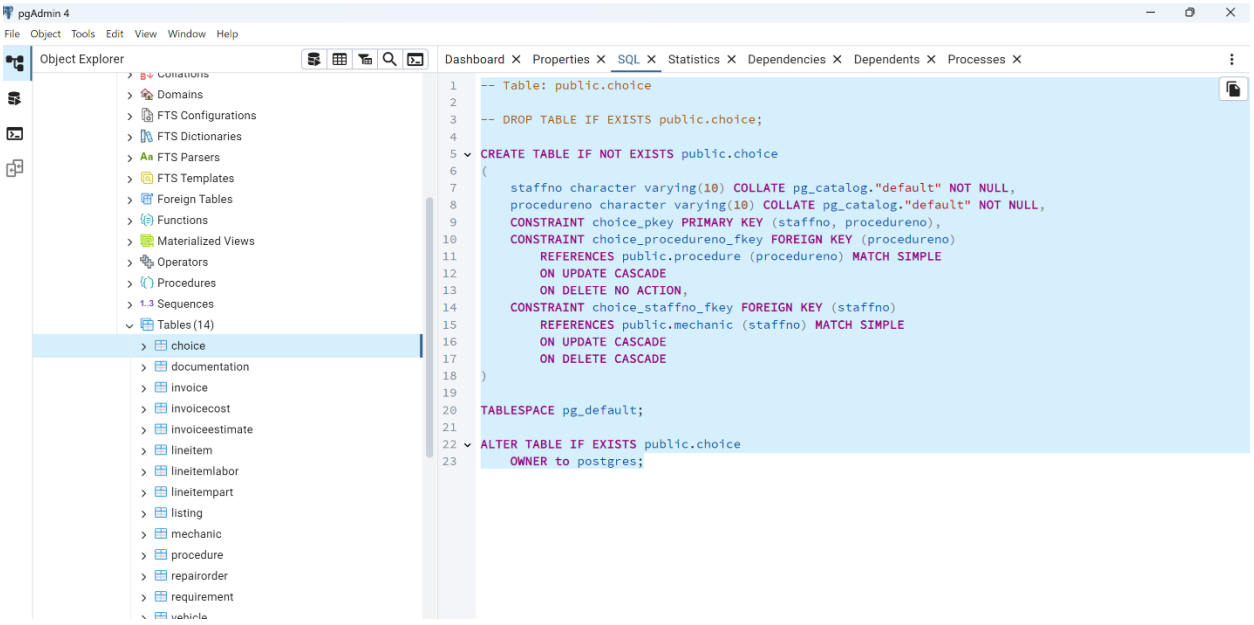


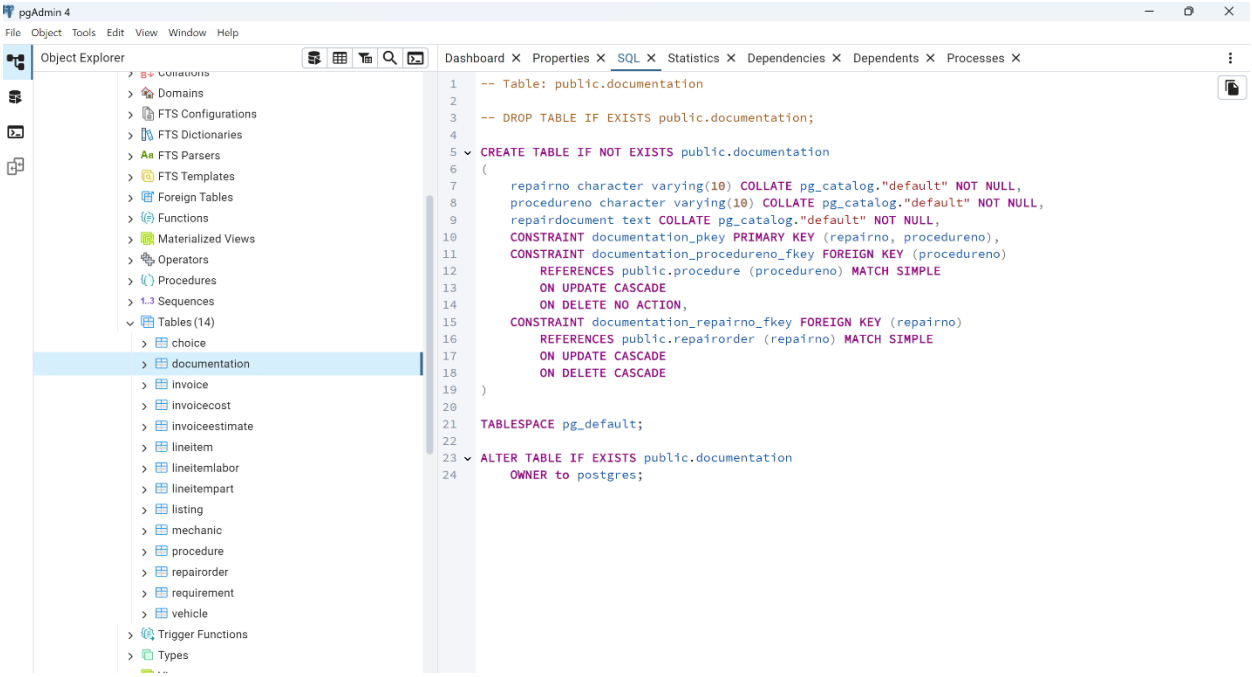
```
CREATE TABLE Choice (
   staffNo VARCHAR(10),
   procedureNo VARCHAR(10),
   PRIMARY KEY (staffNo, procedureNo),
   FOREIGN KEY (staffNo) REFERENCES Mechanic(staffNo) ON UPDATE CASCADE ON
DELETE CASCADE,
   FOREIGN KEY (procedureNo) REFERENCES Procedure(procedureNo) ON UPDATE
CASCADE ON DELETE NO ACTION
);

-- Additional domain constraints
ALTER TABLE RepairOrder ADD CONSTRAINT check_dates CHECK (originDate <=
completeDate OR completeDate IS NULL);
ALTER TABLE Vehicle ADD CONSTRAINT check_year CHECK (year >= 1900 AND year <=
EXTRACT(YEAR FROM CURRENT_DATE));
```

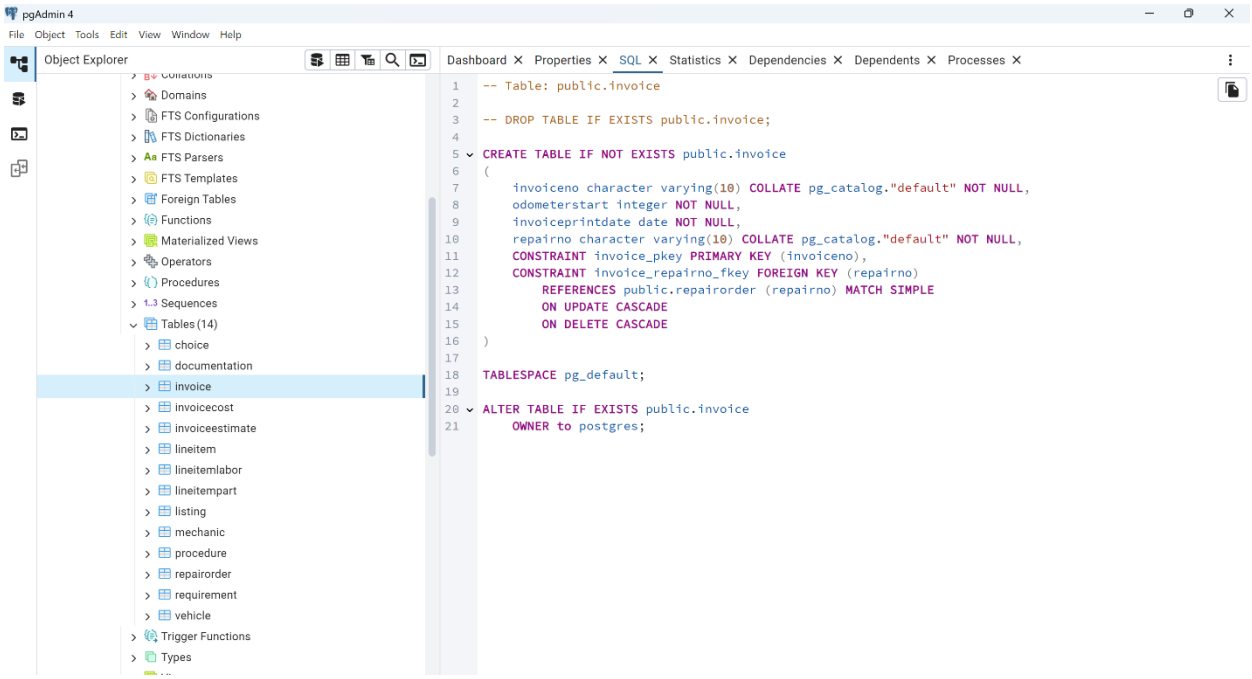


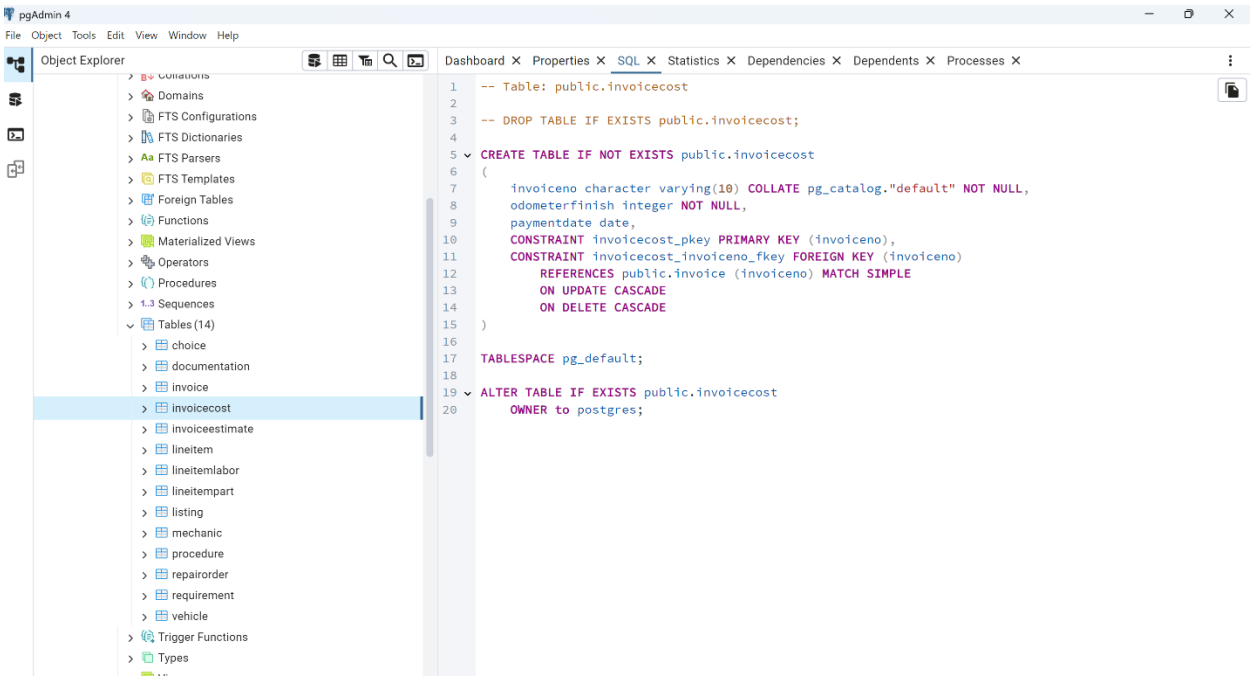
These are the results:

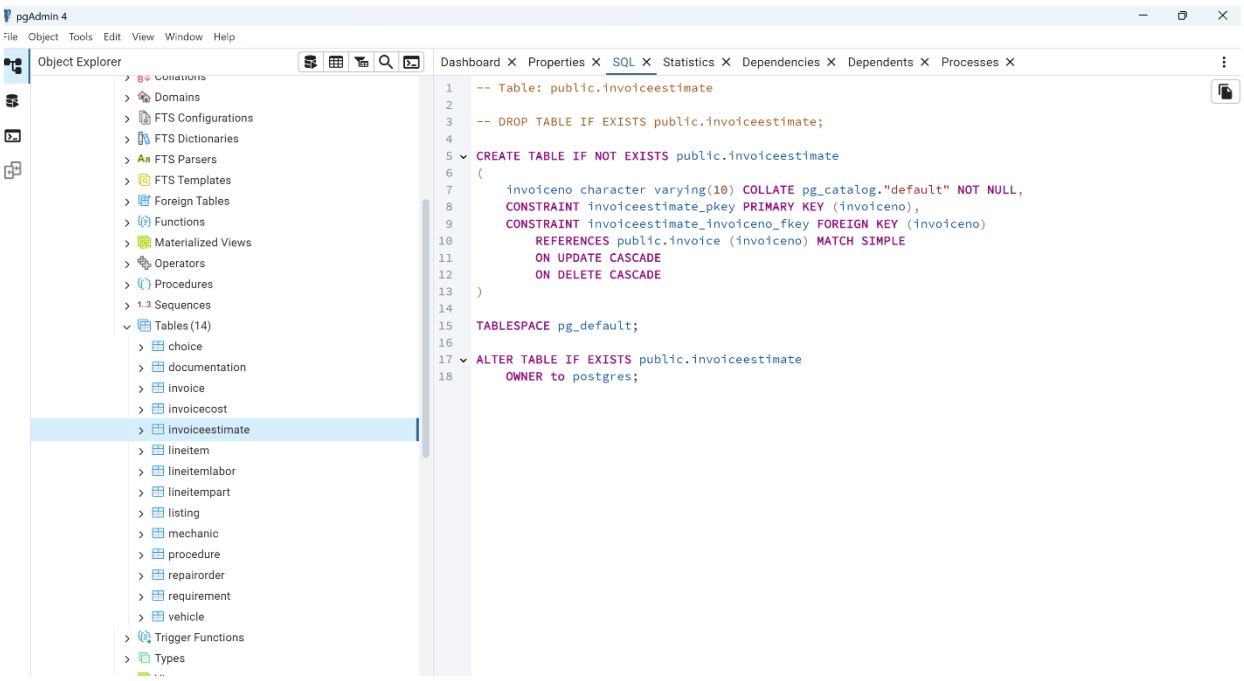


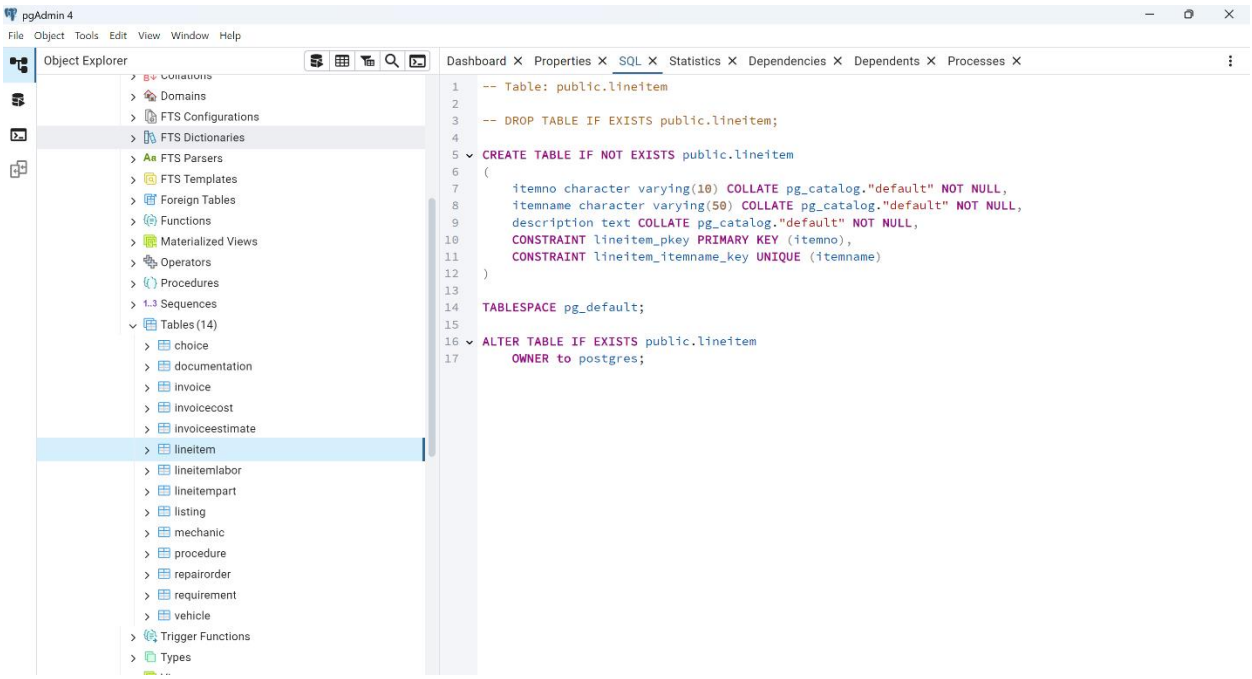


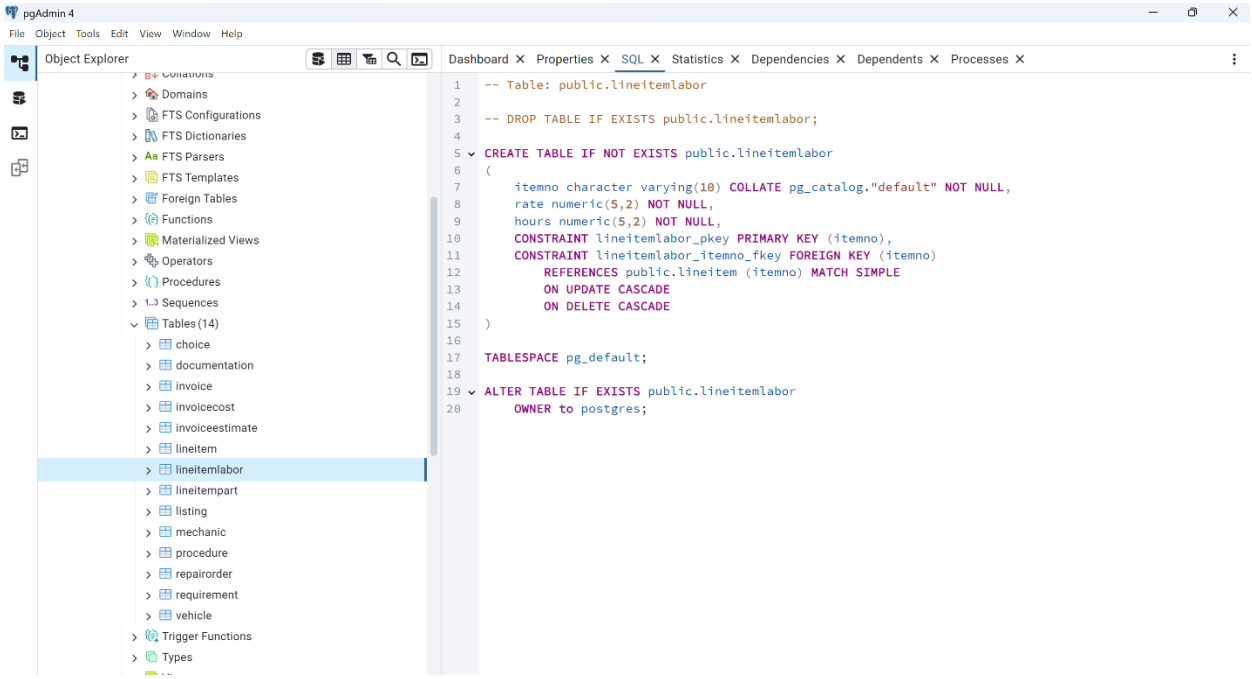

pgAdmin 4
File Object Tools Edit View Window Help
Object Explorer
Dashboard Properties SQL Statistics Dependencies Dependents Processes
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables (14)
choice
documentation
invoice
invoicecost
invoiceestimate
lineitem
lineitemlabor
lineitempart
listing
mechanic
procedure
repairorder
requirement
vehicle
Trigger Functions
Types
-- Table: public.documentation
-- DROP TABLE IF EXISTS public.documentation;
CREATE TABLE IF NOT EXISTS public.documentation
(
repairno character varying(10) COLLATE pg_catalog."default" NOT NULL,
procedureno character varying(10) COLLATE pg_catalog."default" NOT NULL,
repairdocument text COLLATE pg_catalog."default" NOT NULL,
CONSTRAINT documentation_pkey PRIMARY KEY (repairno, procedureno),
CONSTRAINT documentation_procedureno_fkey FOREIGN KEY (procedureno)
REFERENCES public.procedure (procedureno) MATCH SIMPLE
ON UPDATE CASCADE
ON DELETE NO ACTION,
CONSTRAINT documentation_repairno_fkey FOREIGN KEY (repairno)
REFERENCES public.repairorder (repairno) MATCH SIMPLE
ON UPDATE CASCADE
ON DELETE CASCADE
)
TABLESPACE pg_default;
ALTER TABLE IF EXISTS public.documentation
OWNER to postgres;

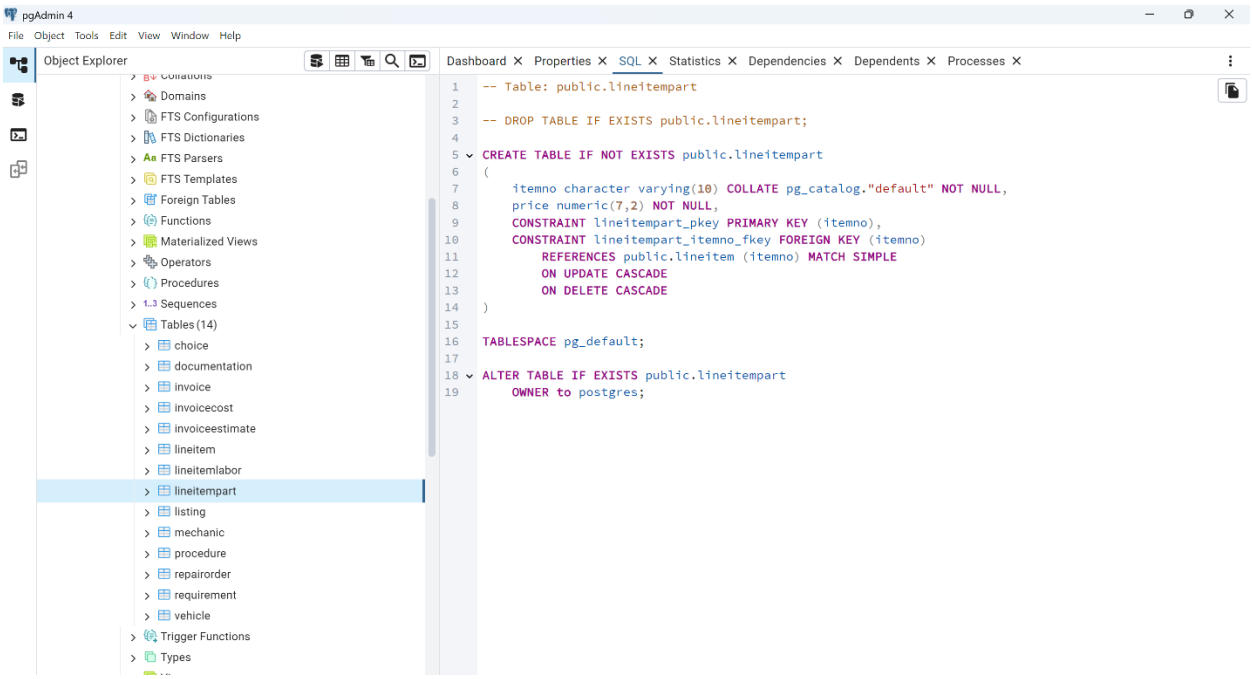

pgAdmin 4
File Object Tools Edit View Window Help
Object Explorer
Dashboard Properties SQL Statistics Dependencies Dependents Processes
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables (14)
choice
documentation
invoice
invoicecost
invoiceestimate
lineitem
lineitemlabor
lineitempart
listing
mechanic
procedure
repairorder
requirement
vehicle
Trigger Functions
Types
-- Table: public.invoice
-- DROP TABLE IF EXISTS public.invoice;
CREATE TABLE IF NOT EXISTS public.invoice
(
invoiceno character varying(10) COLLATE pg_catalog."default" NOT NULL,
odometerstart integer NOT NULL,
invoiceprintdate date NOT NULL,
repairno character varying(10) COLLATE pg_catalog."default" NOT NULL,
CONSTRAINT invoice_pkey PRIMARY KEY (invoiceno),
CONSTRAINT invoice_repairno_fkey FOREIGN KEY (repairno)
REFERENCES public.repairorder (repairno) MATCH SIMPLE
ON UPDATE CASCADE
ON DELETE CASCADE
)
TABLESPACE pg_default;
ALTER TABLE IF EXISTS public.invoice
OWNER to postgres;

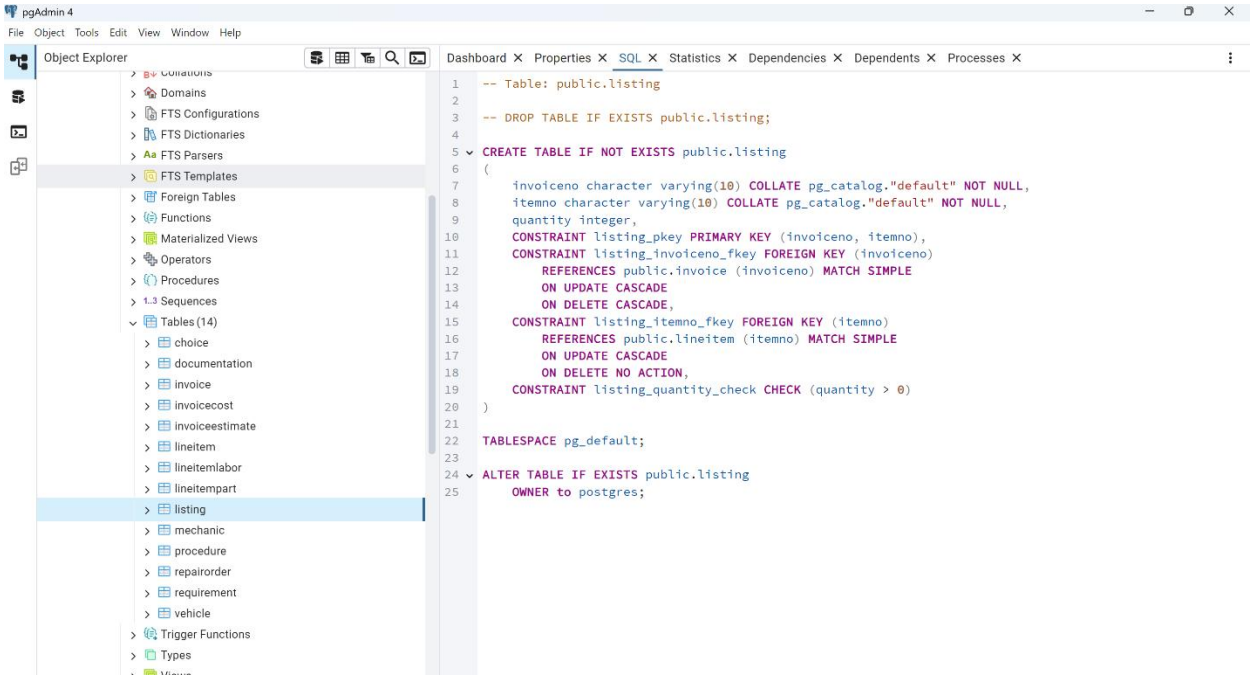

pgAdmin 4
File Object Tools Edit View Window Help
Object Explorer
Dashboard Properties SQL Statistics Dependencies Dependents Processes
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables (14)
choice
documentation
invoice
invoicecost
invoiceestimate
lineitem
lineitemlabor
lineitempart
listing
mechanic
procedure
repairorder
requirement
vehicle
Trigger Functions
Types
-- Table: public.invoicecost
-- DROP TABLE IF EXISTS public.invoicecost;
CREATE TABLE IF NOT EXISTS public.invoicecost
(
invoiceno character varying(10) COLLATE pg_catalog."default" NOT NULL,
odometerfinish integer NOT NULL,
paymentdate date,
CONSTRAINT invoicecost_pkey PRIMARY KEY (invoiceno),
CONSTRAINT invoicecost_invoiceno_fkey FOREIGN KEY (invoiceno)
REFERENCES public.invoice (invoiceno) MATCH SIMPLE
ON UPDATE CASCADE
ON DELETE CASCADE
)
TABLESPACE pg_default;
ALTER TABLE IF EXISTS public.invoicecost
OWNER to postgres;

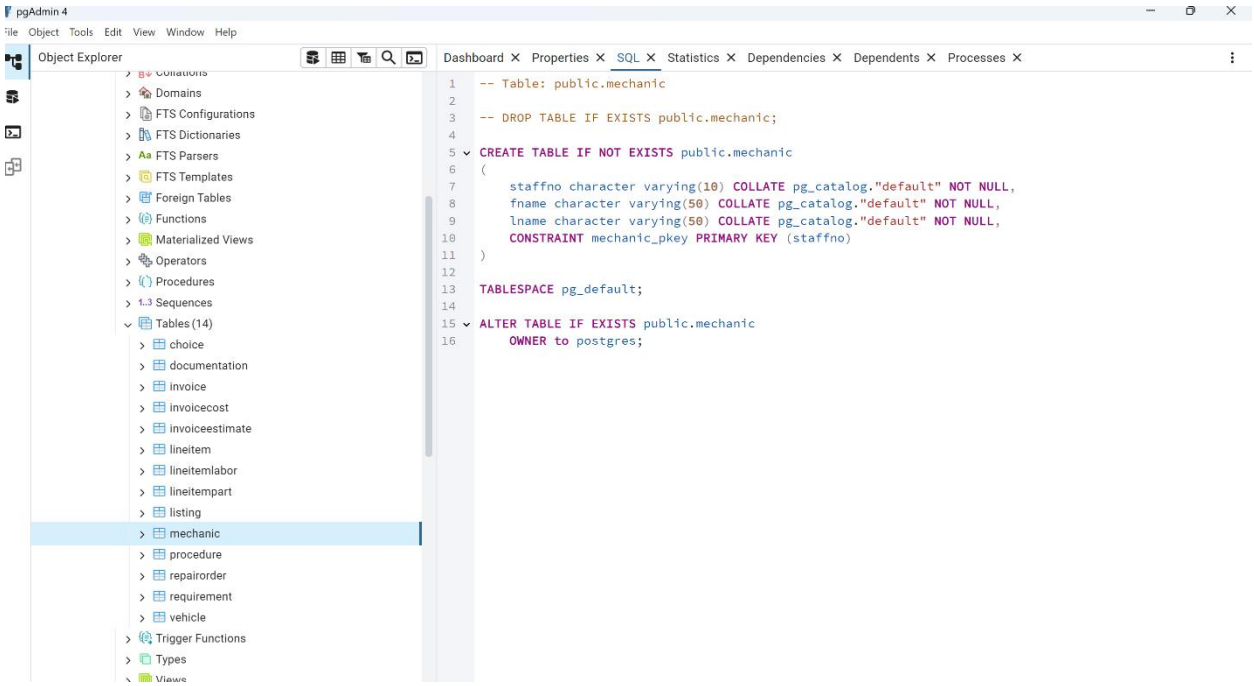

pgAdmin 4
File Object Tools Edit View Window Help
Object Explorer
Dashboard Properties SQL Statistics Dependencies Dependents Processes
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables (14)
choice
documentation
invoice
invoicecost
invoiceestimate
lineitem
lineitemlabor
lineitempart
listing
mechanic
procedure
repairorder
requirement
vehicle
Trigger Functions
Types
-- Table: public.invoiceestimate
-- DROP TABLE IF EXISTS public.invoiceestimate;
CREATE TABLE IF NOT EXISTS public.invoiceestimate
(
invoiceno character varying(10) COLLATE pg_catalog."default" NOT NULL,
CONSTRAINT invoiceestimate_pkey PRIMARY KEY (invoiceno),
CONSTRAINT invoiceestimate_invoiceno_fkey FOREIGN KEY (invoiceno)
REFERENCES public.invoice (invoiceno) MATCH SIMPLE
ON UPDATE CASCADE
ON DELETE CASCADE
)
TABLESPACE pg_default;
ALTER TABLE IF EXISTS public.invoiceestimate
OWNER to postgres;

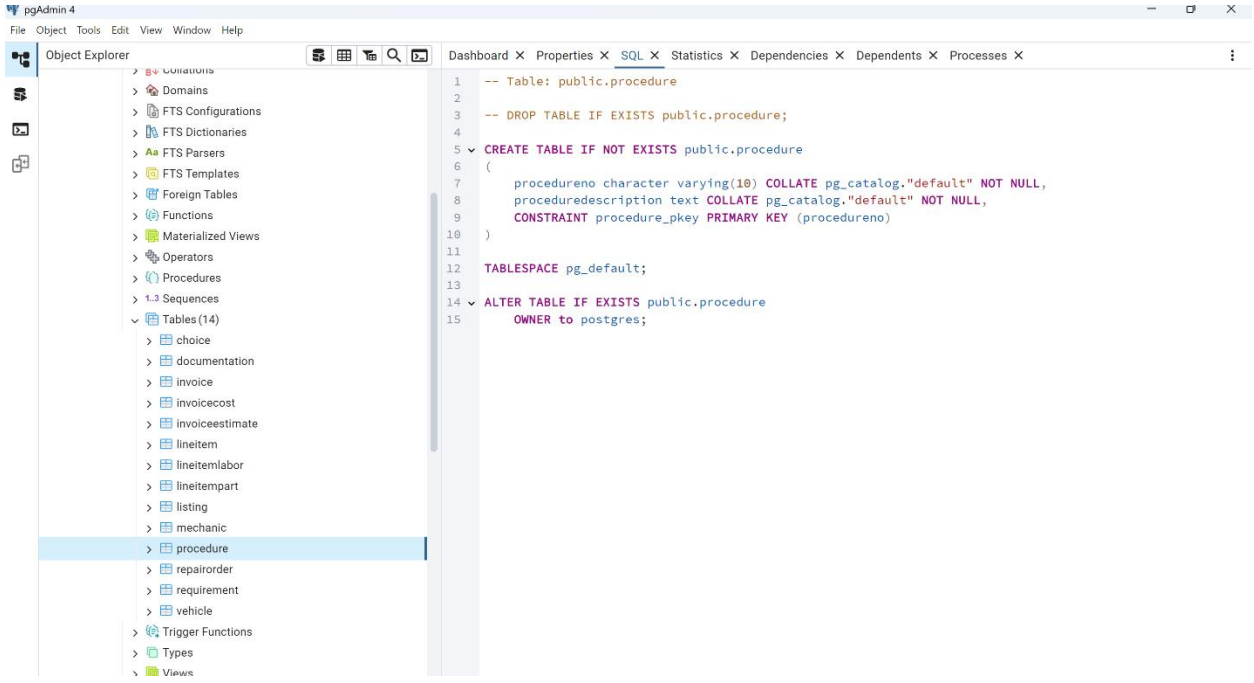

pgAdmin 4
File Object Tools Edit View Window Help
Object Explorer
Dashboard Properties SQL Statistics Dependencies Dependents Processes
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables (14)
choice
documentation
invoice
invoicecost
invoiceestimate
lineitem
lineitemlabor
lineitempart
listing
mechanic
procedure
repairorder
requirement
vehicle
Trigger Functions
Types
-- Table: public.lineitem
-- DROP TABLE IF EXISTS public.lineitem;
CREATE TABLE IF NOT EXISTS public.lineitem
(
itemno character varying(10) COLLATE pg_catalog."default" NOT NULL,
itemname character varying(50) COLLATE pg_catalog."default" NOT NULL,
description text COLLATE pg_catalog."default" NOT NULL,
CONSTRAINT lineitem_pkey PRIMARY KEY (itemno),
CONSTRAINT lineitem_itemname_key UNIQUE (itemname)
)
TABLESPACE pg_default;
ALTER TABLE IF EXISTS public.lineitem
OWNER to postgres;

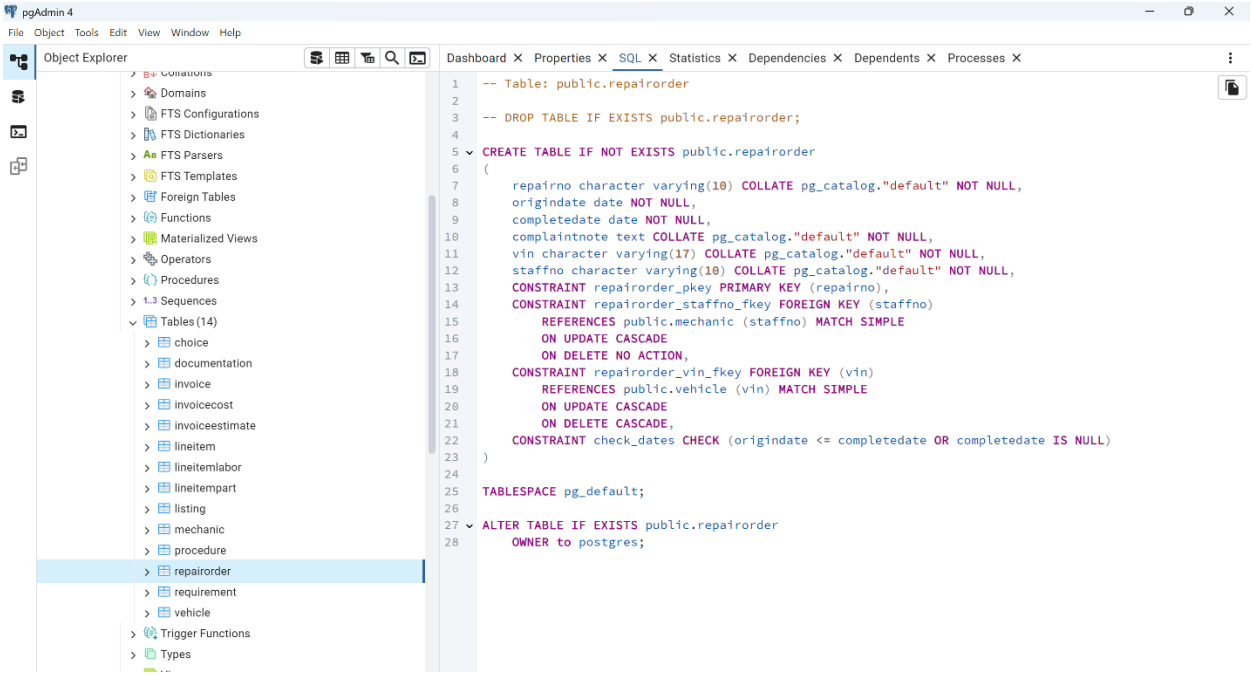

pgAdmin 4
File Object Tools Edit View Window Help
Object Explorer
Dashboard Properties SQL Statistics Dependencies Dependents Processes
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables (14)
choice
documentation
invoice
invoicecost
invoiceestimate
lineitem
lineitemlabor
lineitempart
listing
mechanic
procedure
repairorder
requirement
vehicle
Trigger Functions
Types
-- Table: public.lineitemlabor
-- DROP TABLE IF EXISTS public.lineitemlabor;
CREATE TABLE IF NOT EXISTS public.lineitemlabor
(
itemno character varying(10) COLLATE pg_catalog."default" NOT NULL,
rate numeric(5,2) NOT NULL,
hours numeric(5,2) NOT NULL,
CONSTRAINT lineitemlabor_pkey PRIMARY KEY (itemno),
CONSTRAINT lineitemlabor_itemno_fkey FOREIGN KEY (itemno)
REFERENCES public.lineitem (itemno) MATCH SIMPLE
ON UPDATE CASCADE
ON DELETE CASCADE
)
TABLESPACE pg_default;
ALTER TABLE IF EXISTS public.lineitemlabor
OWNER to postgres;

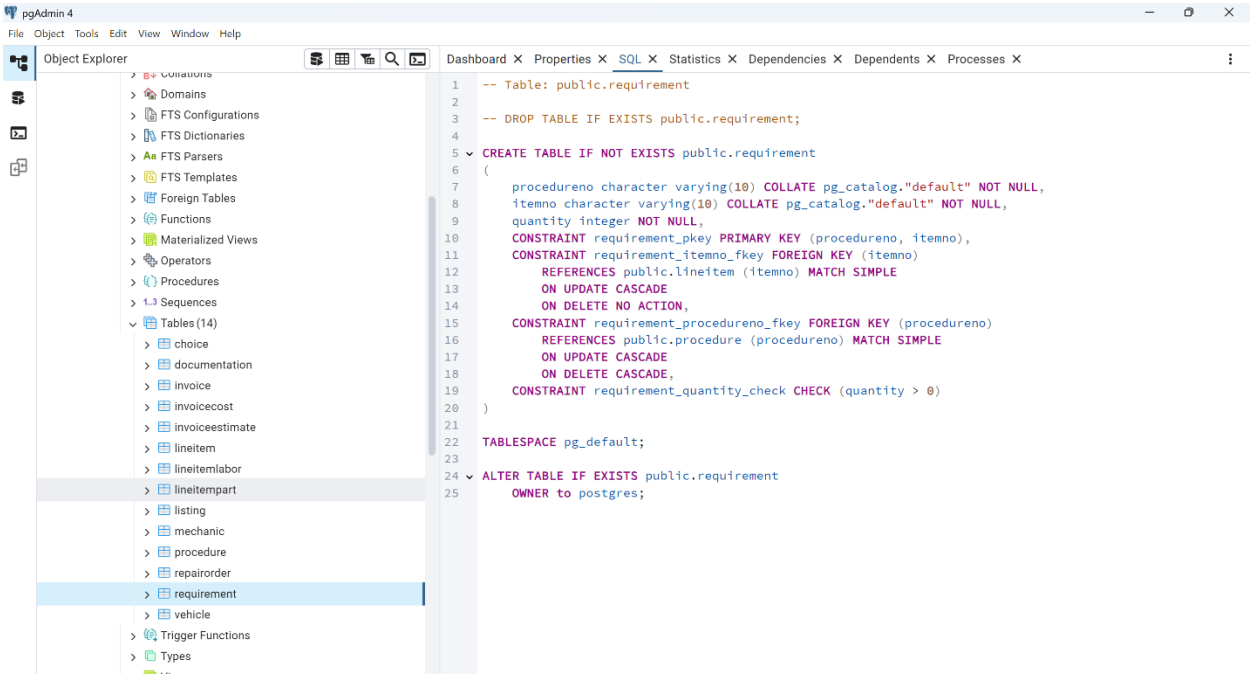

pgAdmin 4
File Object Tools Edit View Window Help
Object Explorer
Dashboard Properties SQL Statistics Dependencies Dependents Processes
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables (14)
choice
documentation
invoice
invoicecost
invoiceestimate
lineitem
lineitemlabor
lineitempart
listing
mechanic
procedure
repairorder
requirement
vehicle
Trigger Functions
Types
-- Table: public.lineitempart
-- DROP TABLE IF EXISTS public.lineitempart;
CREATE TABLE IF NOT EXISTS public.lineitempart
(
itemno character varying(10) COLLATE pg_catalog."default" NOT NULL,
price numeric(7,2) NOT NULL,
CONSTRAINT lineitempart_pkey PRIMARY KEY (itemno),
CONSTRAINT lineitempart_itemno_fkey FOREIGN KEY (itemno)
REFERENCES public.lineitem (itemno) MATCH SIMPLE
ON UPDATE CASCADE
ON DELETE CASCADE
)
TABLESPACE pg_default;
ALTER TABLE IF EXISTS public.lineitempart
OWNER to postgres;

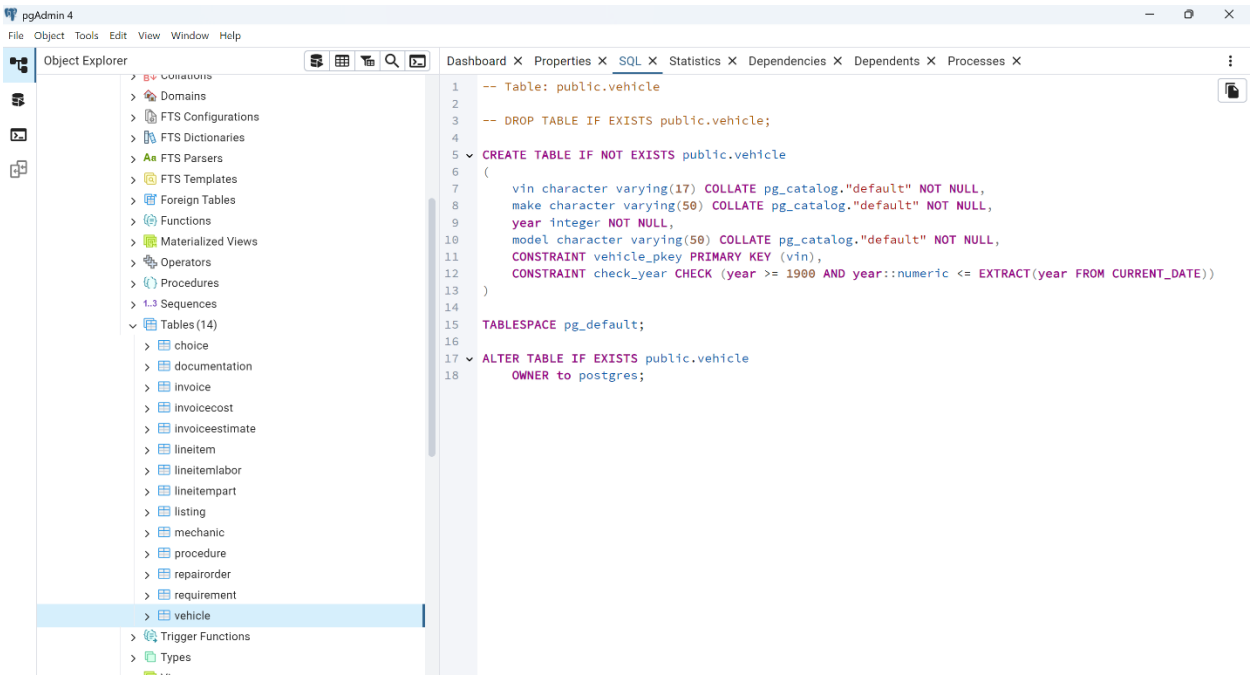

pgAdmin 4
File Object Tools Edit View Window Help
Object Explorer
Dashboard Properties SQL Statistics Dependencies Dependents Processes
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables (14)
choice
documentation
invoice
invoicecost
invoiceestimate
lineitem
lineitemlabor
lineitempart
listing
mechanic
procedure
repairorder
requirement
vehicle
Trigger Functions
Types
-- Table: public.listing
-- DROP TABLE IF EXISTS public.listing;
CREATE TABLE IF NOT EXISTS public.listing
(
invoiceno character varying(10) COLLATE pg_catalog."default" NOT NULL,
itemno character varying(10) COLLATE pg_catalog."default" NOT NULL,
quantity integer,
CONSTRAINT listing_pkey PRIMARY KEY (invoiceno, itemno),
CONSTRAINT listing_invoiceno_fkey FOREIGN KEY (invoiceno)
REFERENCES public.invoice (invoiceno) MATCH SIMPLE
ON UPDATE CASCADE
ON DELETE CASCADE,
CONSTRAINT listing_itemno_fkey FOREIGN KEY (itemno)
REFERENCES public.lineitem (itemno) MATCH SIMPLE
ON UPDATE CASCADE
ON DELETE NO ACTION,
CONSTRAINT listing_quantity_check CHECK (quantity > 0)
)
TABLESPACE pg_default;
ALTER TABLE IF EXISTS public.listing
OWNER to postgres;


pgAdmin 4
File Object Tools Edit View Window Help
Object Explorer
Dashboard Properties SQL Statistics Dependencies Dependents Processes
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables (14)
choice
documentation
invoice
invoicecost
invoiceestimate
lineitem
lineitemlabor
lineitempart
listing
mechanic
procedure
repairorder
requirement
vehicle
Trigger Functions
Types
Views
-- Table: public.mechanic
-- DROP TABLE IF EXISTS public.mechanic;
CREATE TABLE IF NOT EXISTS public.mechanic
(
staffno character varying(10) COLLATE pg_catalog."default" NOT NULL,
fname character varying(50) COLLATE pg_catalog."default" NOT NULL,
lname character varying(50) COLLATE pg_catalog."default" NOT NULL,
CONSTRAINT mechanic_pkey PRIMARY KEY (staffno)
)
TABLESPACE pg_default;
ALTER TABLE IF EXISTS public.mechanic
OWNER to postgres;


pgAdmin 4
File Object Tools Edit View Window Help
Object Explorer
Dashboard Properties SQL Statistics Dependencies Dependents Processes
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables (14)
choice
documentation
invoice
invoicecost
invoiceestimate
lineitem
lineitemlabor
lineitempart
listing
mechanic
procedure
repairorder
requirement
vehicle
Trigger Functions
Types
Views
-- Table: public.procedure
-- DROP TABLE IF EXISTS public.procedure;
CREATE TABLE IF NOT EXISTS public.procedure
(
procedureno character varying(10) COLLATE pg_catalog."default" NOT NULL,
proceduredescription text COLLATE pg_catalog."default" NOT NULL,
CONSTRAINT procedure_pkey PRIMARY KEY (procedureno)
)
TABLESPACE pg_default;
ALTER TABLE IF EXISTS public.procedure
OWNER to postgres;


pgAdmin 4
File Object Tools Edit View Window Help
Object Explorer
Dashboard Properties SQL Statistics Dependencies Dependents Processes
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables (14)
choice
documentation
invoice
invoicecost
invoiceestimate
lineitem
lineitemlabor
lineitempart
listing
mechanic
procedure
repairorder
requirement
vehicle
Trigger Functions
Types
-- Table: public.repairorder
-- DROP TABLE IF EXISTS public.repairorder;
CREATE TABLE IF NOT EXISTS public.repairorder
(
repairno character varying(10) COLLATE pg_catalog."default" NOT NULL,
origindate date NOT NULL,
completedate date NOT NULL,
complaintnote text COLLATE pg_catalog."default" NOT NULL,
vin character varying(17) COLLATE pg_catalog."default" NOT NULL,
staffno character varying(10) COLLATE pg_catalog."default" NOT NULL,
CONSTRAINT repairorder_pkey PRIMARY KEY (repairno),
CONSTRAINT repairorder_staffno_fkey FOREIGN KEY (staffno)
REFERENCES public.mechanic (staffno) MATCH SIMPLE
ON UPDATE CASCADE
ON DELETE NO ACTION,
CONSTRAINT repairorder_vin_fkey FOREIGN KEY (vin)
REFERENCES public.vehicle (vin) MATCH SIMPLE
ON UPDATE CASCADE
ON DELETE CASCADE,
CONSTRAINT check_dates CHECK (origindate <= completedate OR completedate IS NULL)
)
TABLESPACE pg_default;
ALTER TABLE IF EXISTS public.repairorder
OWNER to postgres;

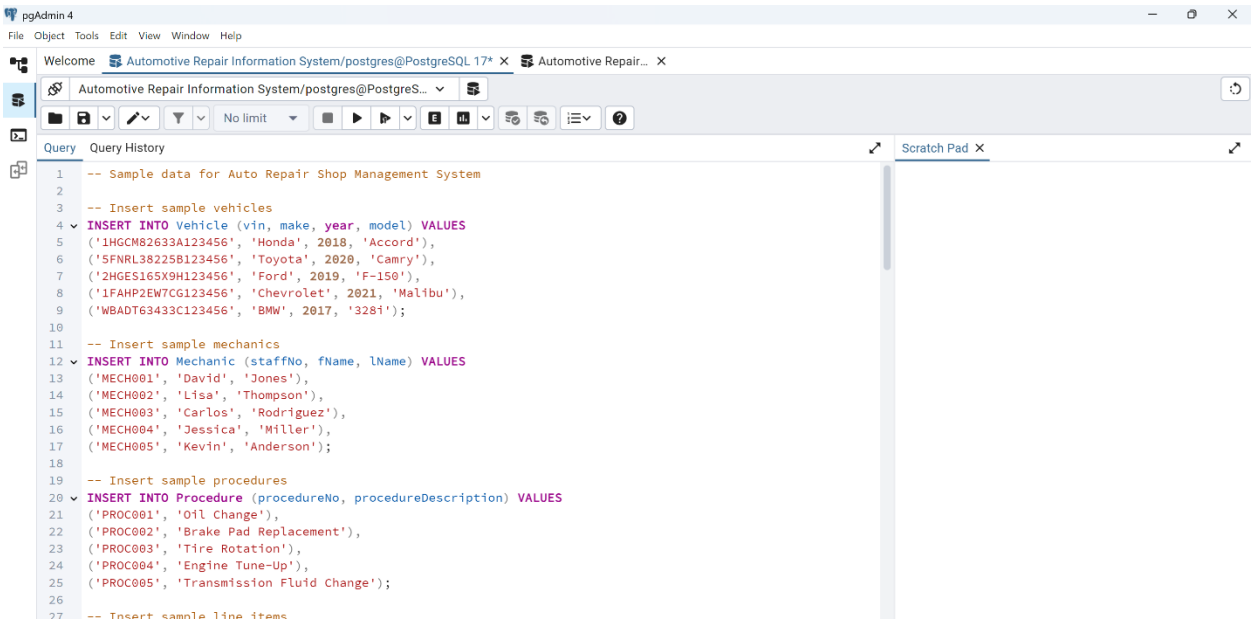

pgAdmin 4
File Object Tools Edit View Window Help
Object Explorer
Dashboard Properties SQL Statistics Dependencies Dependents Processes
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables (14)
choice
documentation
invoice
invoicecost
invoiceestimate
lineitem
lineitemlabor
lineitempart
listing
mechanic
procedure
repairorder
requirement
vehicle
Trigger Functions
Types
-- Table: public.requirement
-- DROP TABLE IF EXISTS public.requirement;
CREATE TABLE IF NOT EXISTS public.requirement
(
procedureno character varying(10) COLLATE pg_catalog."default" NOT NULL,
itemno character varying(10) COLLATE pg_catalog."default" NOT NULL,
quantity integer NOT NULL,
CONSTRAINT requirement_pkey PRIMARY KEY (procedureno, itemno),
CONSTRAINT requirement_itemno_fkey FOREIGN KEY (itemno)
REFERENCES public.lineitem (itemno) MATCH SIMPLE
ON UPDATE CASCADE
ON DELETE NO ACTION,
CONSTRAINT requirement_procedureno_fkey FOREIGN KEY (procedureno)
REFERENCES public.procedure (procedureno) MATCH SIMPLE
ON UPDATE CASCADE
ON DELETE CASCADE,
CONSTRAINT requirement_quantity_check CHECK (quantity > 0)
)
TABLESPACE pg_default;
ALTER TABLE IF EXISTS public.requirement
OWNER to postgres;


pgAdmin 4
File Object Tools Edit View Window Help
Object Explorer
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables (14)
choice
documentation
invoice
invoicecost
invoiceestimate
lineitem
lineitemlabor
lineitempart
listing
mechanic
procedure
repairorder
requirement
vehicle
Trigger Functions
Types
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Processes
-- Table: public.vehicle
-- DROP TABLE IF EXISTS public.vehicle;
CREATE TABLE IF NOT EXISTS public.vehicle
(
vin character varying(17) COLLATE pg_catalog."default" NOT NULL,
make character varying(50) COLLATE pg_catalog."default" NOT NULL,
year integer NOT NULL,
model character varying(50) COLLATE pg_catalog."default" NOT NULL,
CONSTRAINT vehicle_pkey PRIMARY KEY (vin),
CONSTRAINT check_year CHECK (year >= 1900 AND year::numeric <= EXTRACT(year FROM CURRENT_DATE))
)
TABLESPACE pg_default;
ALTER TABLE IF EXISTS public.vehicle
OWNER to postgres;

Test Data

-- Sample data for Auto Repair Shop Management System

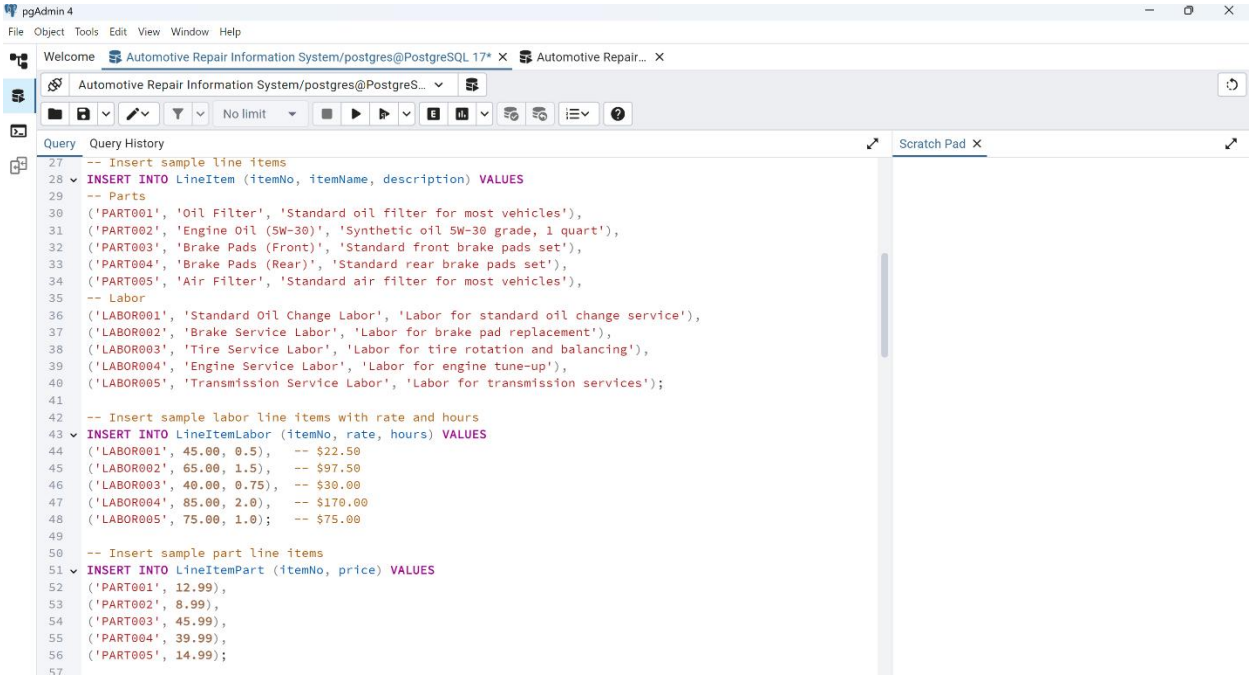
```
-- Insert sample vehicles
INSERT INTO Vehicle (vin, make, year, model) VALUES
('1HGCM82633A123456', 'Honda', 2018, 'Accord'),
('5FNRL38225B123456', 'Toyota', 2020, 'Camry'),
('2HGES165X9H123456', 'Ford', 2019, 'F-150'),
('1FAHP2EW7CG123456', 'Chevrolet', 2021, 'Malibu'),
('WBADT63433C123456', 'BMW', 2017, '328i');

-- Insert sample mechanics
INSERT INTO Mechanic (staffNo, fName, lName) VALUES
('MECH001', 'David', 'Jones'),
('MECH002', 'Lisa', 'Thompson'),
('MECH003', 'Carlos', 'Rodriguez'),
('MECH004', 'Jessica', 'Miller'),
('MECH005', 'Kevin', 'Anderson');

-- Insert sample procedures
INSERT INTO Procedure (procedureNo, procedureDescription) VALUES
('PROC001', 'Oil Change'),
('PROC002', 'Brake Pad Replacement'),
('PROC003', 'Tire Rotation'),
('PROC004', 'Engine Tune-Up'),
('PROC005', 'Transmission Fluid Change');
```



-- Insert sample line items
INSERT INTO LineItem (itemNo, itemName, description) VALUES
-- Parts
('PART001', 'Oil Filter', 'Standard oil filter for most vehicles'),
('PART002', 'Engine Oil (5W-30)', 'Synthetic oil 5W-30 grade, 1 quart'),
('PART003', 'Brake Pads (Front)', 'Standard front brake pads set'),
('PART004', 'Brake Pads (Rear)', 'Standard rear brake pads set'),
('PART005', 'Air Filter', 'Standard air filter for most vehicles'),
-- Labor
('LABOR001', 'Standard Oil Change Labor', 'Labor for standard oil change service'),
('LABOR002', 'Brake Service Labor', 'Labor for brake pad replacement'),
('LABOR003', 'Tire Service Labor', 'Labor for tire rotation and balancing'),
('LABOR004', 'Engine Service Labor', 'Labor for engine tune-up'),
('LABOR005', 'Transmission Service Labor', 'Labor for transmission services');

-- Insert sample labor line items with rate and hours
INSERT INTO LineItemLabor (itemNo, rate, hours) VALUES
('LABOR001', 45.00, 0.5), -- $22.50
('LABOR002', 65.00, 1.5), -- $97.50
('LABOR003', 40.00, 0.75), -- $30.00
('LABOR004', 85.00, 2.0), -- $170.00
('LABOR005', 75.00, 1.0); -- $75.00

-- Insert sample part line items
INSERT INTO LineItemPart (itemNo, price) VALUES
('PART001', 12.99),
('PART002', 8.99),
('PART003', 45.99),
('PART004', 39.99),
('PART005', 14.99);



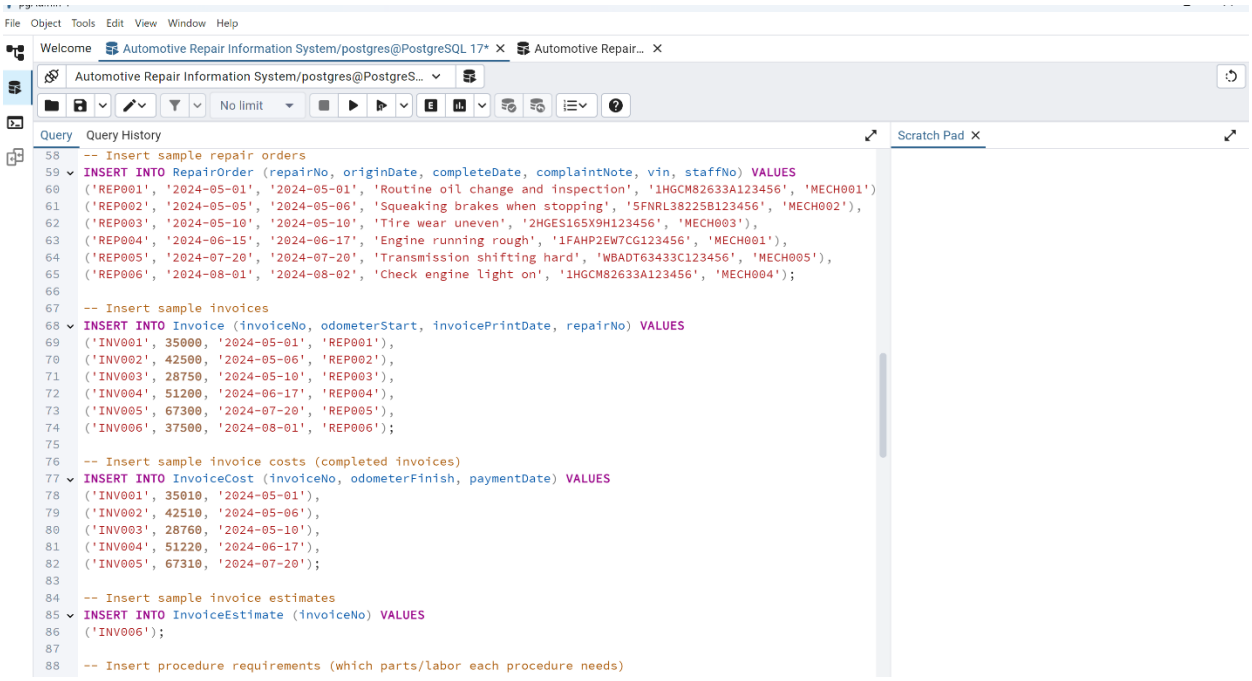
-- Insert sample repair orders
INSERT INTO RepairOrder (repairNo, originDate, completeDate, complaintNote, vin, staffNo) VALUES
('REP001', '2024-05-01', '2024-05-01', 'Routine oil change and inspection', '1HGCM82633A123456', 'MECH001'),
('REP002', '2024-05-05', '2024-05-06', 'Squeaking brakes when stopping', '5FNRL38225B123456', 'MECH002'),
('REP003', '2024-05-10', '2024-05-10', 'Tire wear uneven', '2HGES165X9H123456', 'MECH003'),
('REP004', '2024-06-15', '2024-06-17', 'Engine running rough', '1FAHP2EW7CG123456', 'MECH001'),

('REP005', '2024-07-20', '2024-07-20', 'Transmission shifting hard', 'WBADT63433C123456', 'MECH005'),
('REP006', '2024-08-01', '2024-08-02', 'Check engine light on', '1HGCM82633A123456', 'MECH004');

-- Insert sample invoices
INSERT INTO Invoice (invoiceNo, odometerStart, invoicePrintDate, repairNo) VALUES
('INV001', 35000, '2024-05-01', 'REP001'),
('INV002', 42500, '2024-05-06', 'REP002'),
('INV003', 28750, '2024-05-10', 'REP003'),
('INV004', 51200, '2024-06-17', 'REP004'),
('INV005', 67300, '2024-07-20', 'REP005'),
('INV006', 37500, '2024-08-01', 'REP006');

-- Insert sample invoice costs (completed invoices)
INSERT INTO InvoiceCost (invoiceNo, odometerFinish, paymentDate) VALUES
('INV001', 35010, '2024-05-01'),
('INV002', 42510, '2024-05-06'),
('INV003', 28760, '2024-05-10'),
('INV004', 51220, '2024-06-17'),
('INV005', 67310, '2024-07-20');

-- Insert sample invoice estimates
INSERT INTO InvoiceEstimate (invoiceNo) VALUES
('INV006');



-- Insert procedure requirements (which parts/labor each procedure needs)
INSERT INTO Requirement (procedureNo, itemNo, quantity) VALUES
-- Oil Change requirements
('PROC001', 'PART001', 1), -- Oil Filter
('PROC001', 'PART002', 5), -- 5 quarts of oil
('PROC001', 'LABOR001', 1), -- Oil change labor

-- Brake Pad Replacement requirements
('PROC002', 'PART003', 1), -- Front brake pads
('PROC002', 'PART004', 1), -- Rear brake pads
('PROC002', 'LABOR002', 1), -- Brake service labor

-- Tire Rotation requirements
('PROC003', 'LABOR003', 1), -- Tire service labor

-- Engine Tune-Up requirements
('PROC004', 'PART005', 1), -- Air filter
('PROC004', 'LABOR004', 1), -- Engine service labor

-- Transmission Fluid Change requirements
('PROC005', 'PART002', 6), -- 6 quarts of oil type fluid

('PROC005', 'LABOR005', 1); -- Transmission service labor

-- Insert invoice listings (which parts/labor are on each invoice)
INSERT INTO Listing (invoiceNo, itemNo, quantity) VALUES
-- Invoice 1: Oil Change
('INV001', 'PART001', 1),
('INV001', 'PART002', 5),
('INV001', 'LABOR001', 1),

-- Invoice 2: Brake Pad Replacement
('INV002', 'PART003', 1),
('INV002', 'PART004', 1),
('INV002', 'LABOR002', 1),

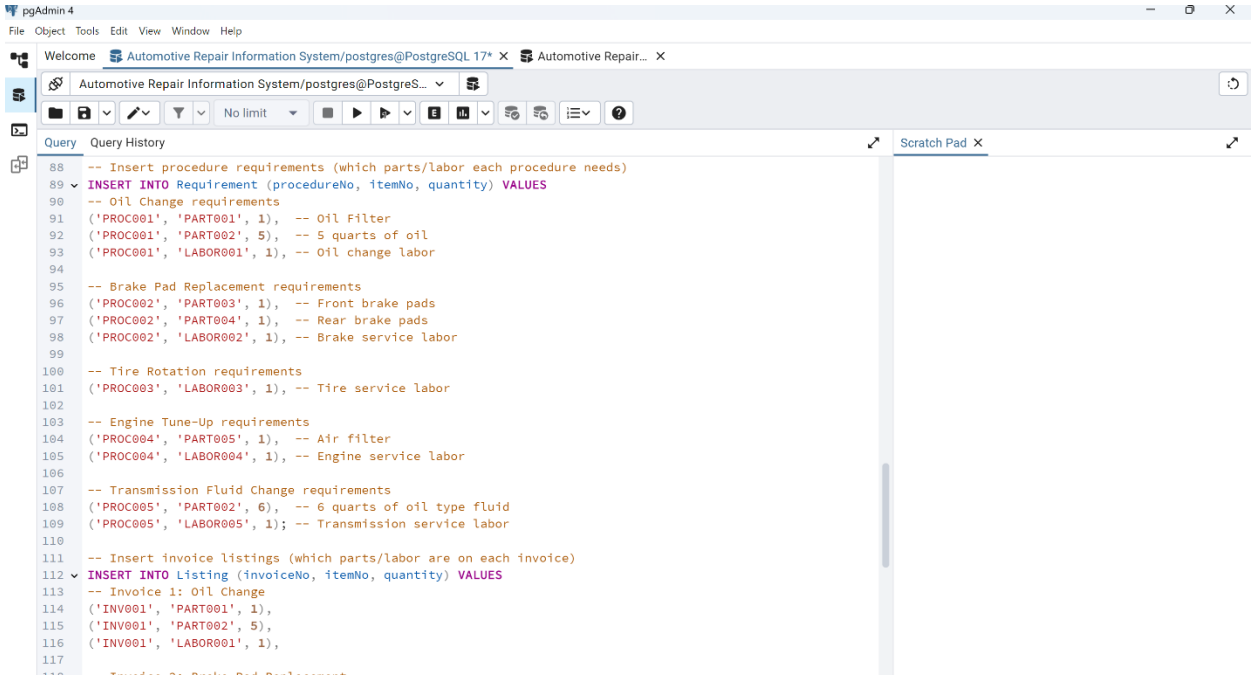
-- Invoice 3: Tire Rotation
('INV003', 'LABOR003', 1),

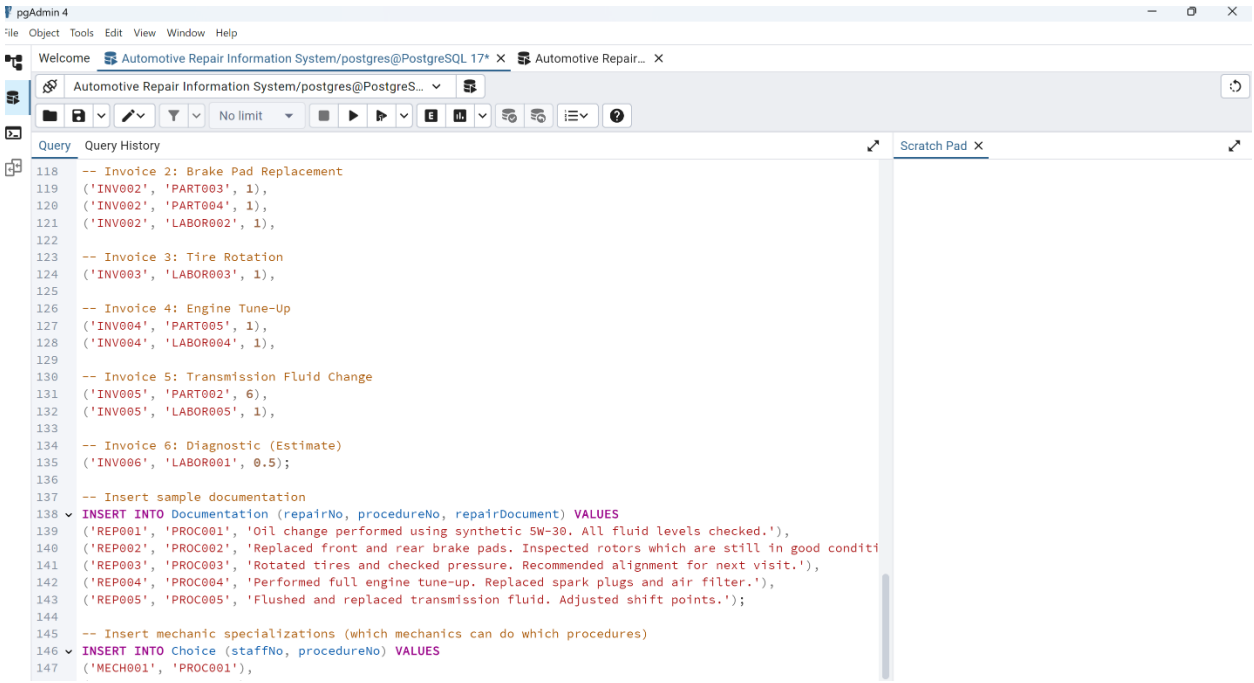
-- Invoice 4: Engine Tune-Up
('INV004', 'PART005', 1),
('INV004', 'LABOR004', 1),

-- Invoice 5: Transmission Fluid Change
('INV005', 'PART002', 6),
('INV005', 'LABOR005', 1),

-- Invoice 6: Diagnostic (Estimate)
('INV006', 'LABOR001', 0.5);

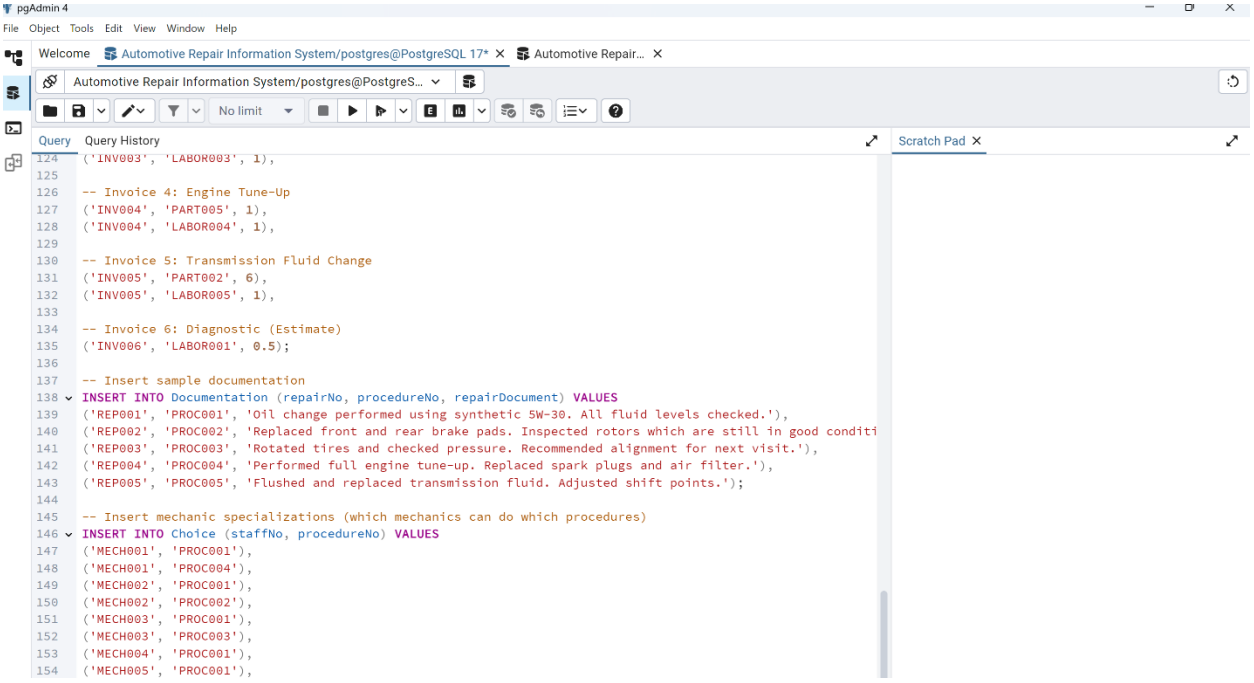
-- Insert sample documentation
INSERT INTO Documentation (repairNo, procedureNo, repairDocument) VALUES
('REP001', 'PROC001', 'Oil change performed using synthetic 5W-30. All fluid levels checked.'),
('REP002', 'PROC002', 'Replaced front and rear brake pads. Inspected rotors which are still in good condition.'),
('REP003', 'PROC003', 'Rotated tires and checked pressure. Recommended alignment for next visit.'),
('REP004', 'PROC004', 'Performed full engine tune-up. Replaced spark plugs and air filter.'),
('REP005', 'PROC005', 'Flushed and replaced transmission fluid. Adjusted shift points.');



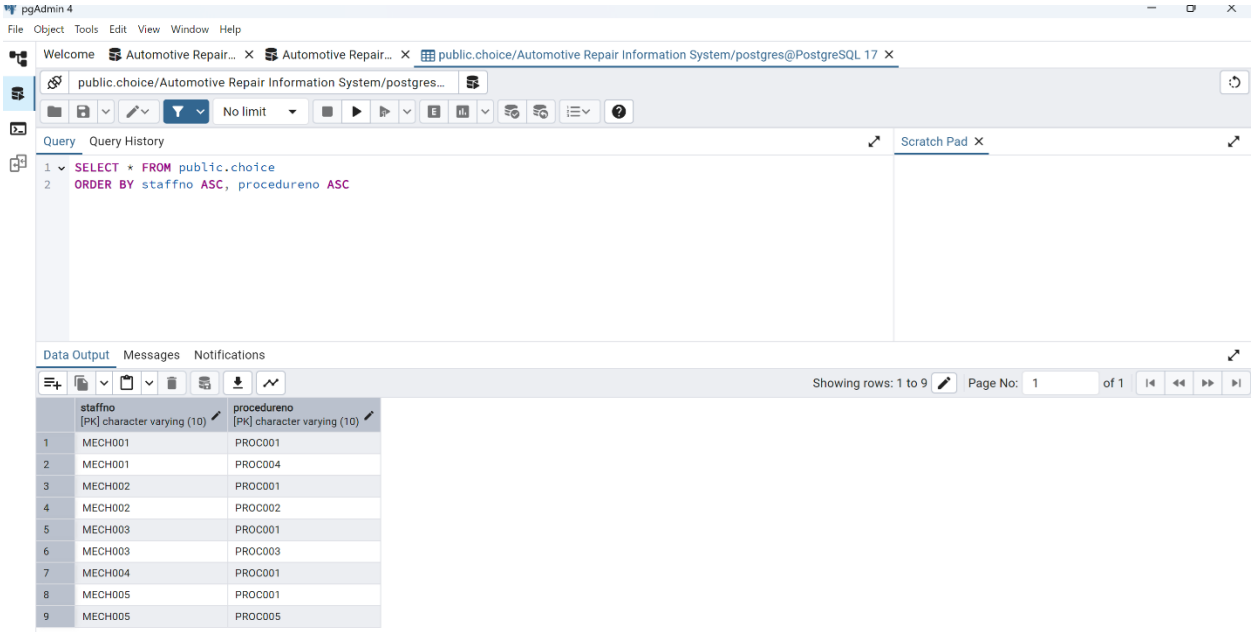


-- Insert mechanic specializations (which mechanics can do which procedures)
INSERT INTO Choice (staffNo, procedureNo) VALUES
('MECH001', 'PROC001'),
('MECH001', 'PROC004'),

('MECH002', 'PROC001'),
('MECH002', 'PROC002'),
('MECH003', 'PROC001'),
('MECH003', 'PROC003'),
('MECH004', 'PROC001'),
('MECH005', 'PROC001'),
('MECH005', 'PROC005');

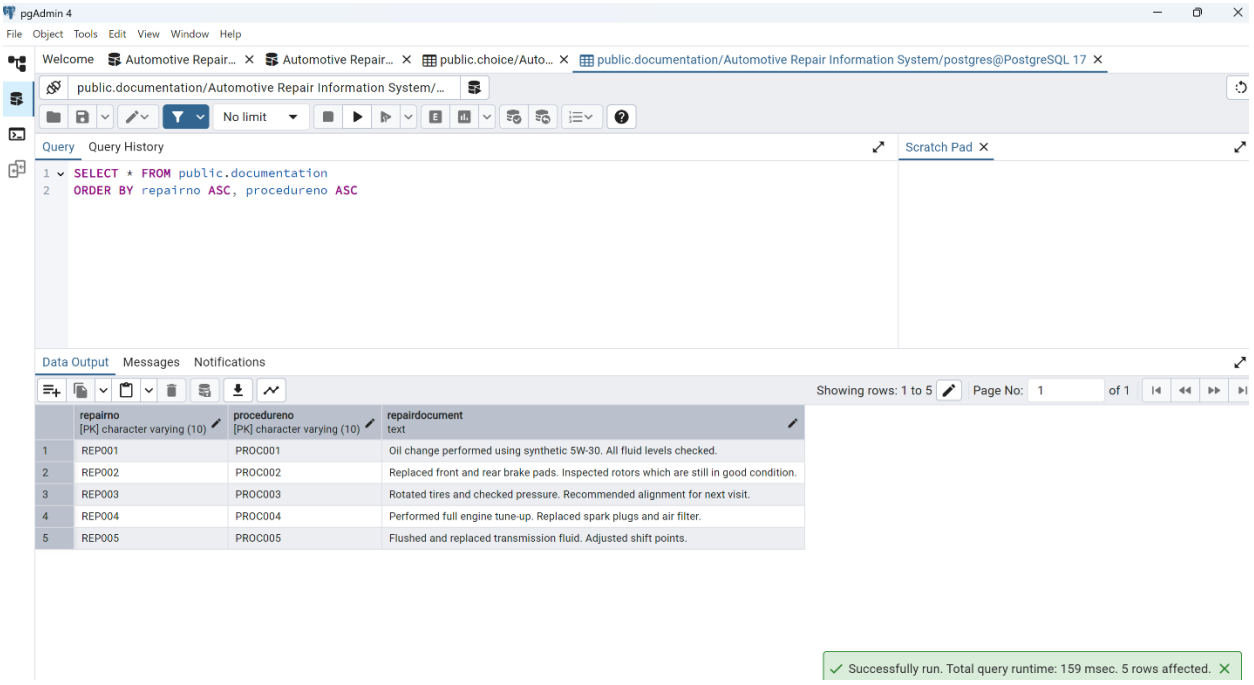


Resulting Tables:



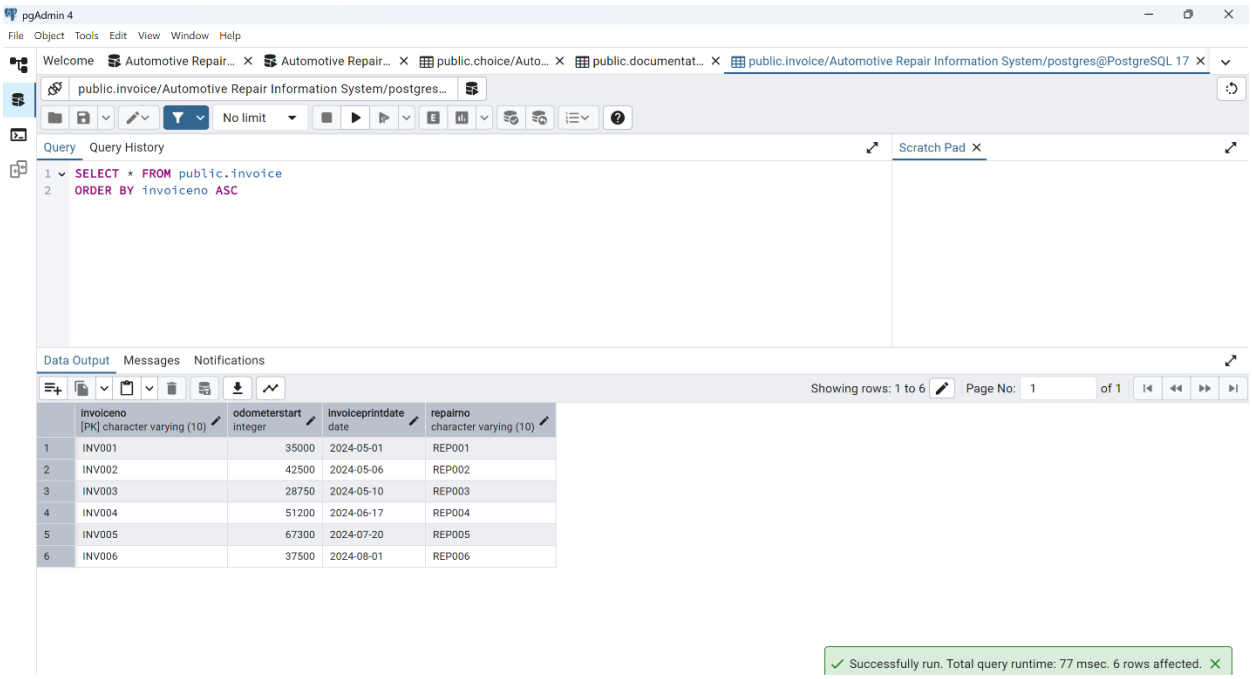
```
SELECT * FROM public.choice
ORDER BY staffno ASC, procedureno ASC
```

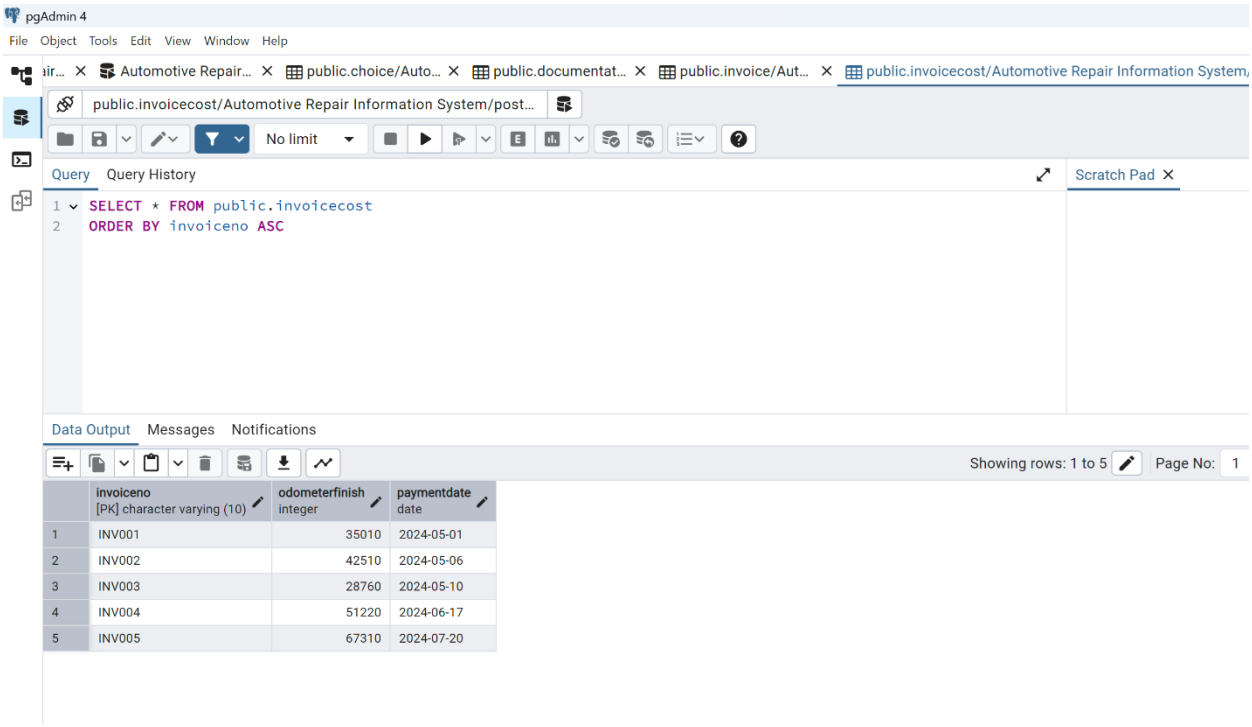
	staffno [PK] character varying (10)	procedureno [PK] character varying (10)
1	MECH001	PROC001
2	MECH001	PROC004
3	MECH002	PROC001
4	MECH002	PROC002
5	MECH003	PROC001
6	MECH003	PROC003
7	MECH004	PROC001
8	MECH005	PROC001
9	MECH005	PROC005

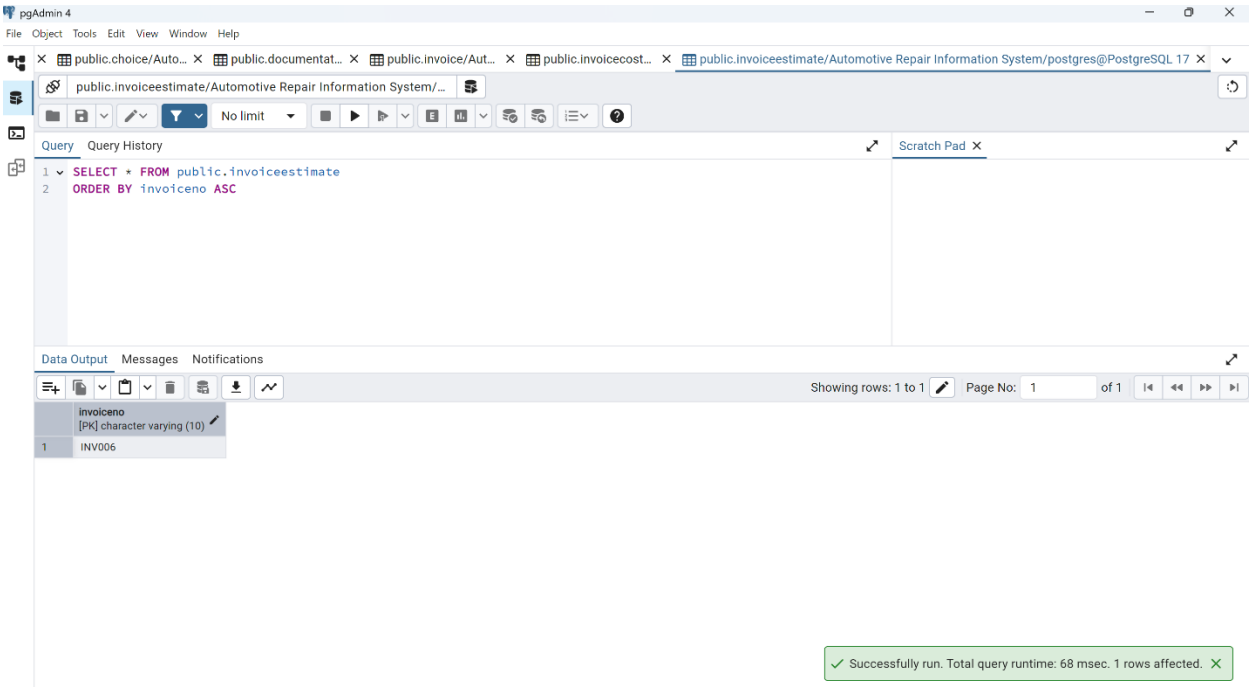


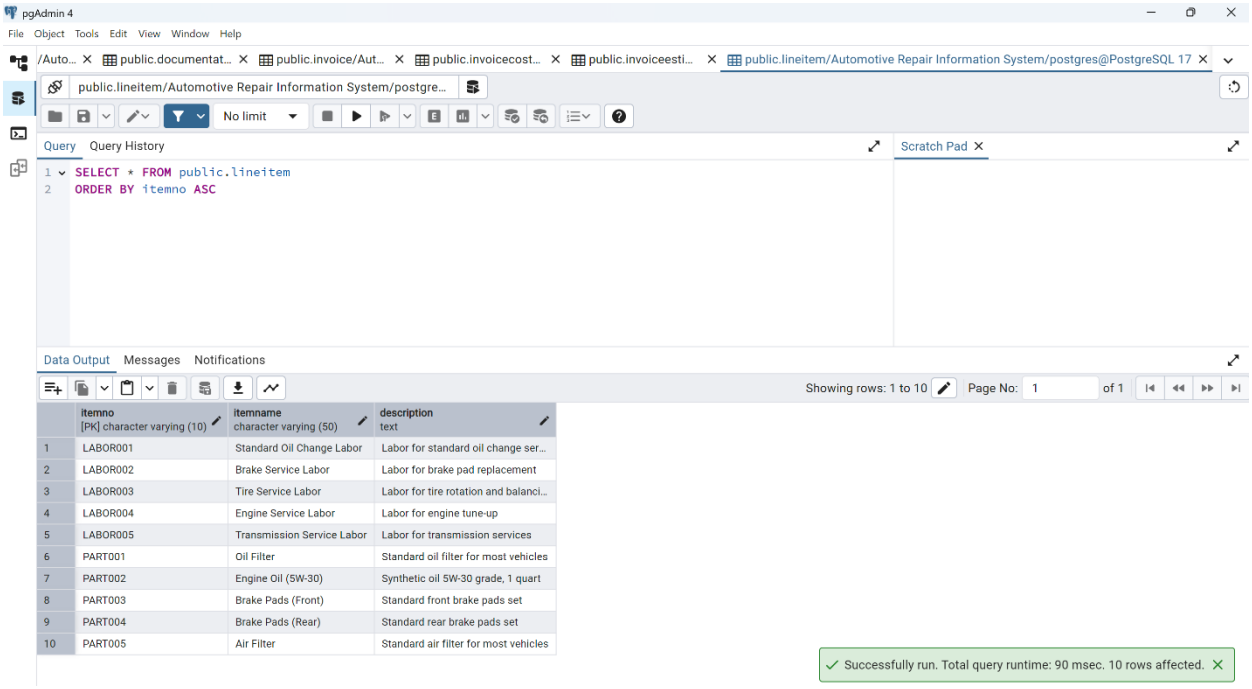
```
SELECT * FROM public.documentation
ORDER BY repairno ASC, procedureno ASC
```

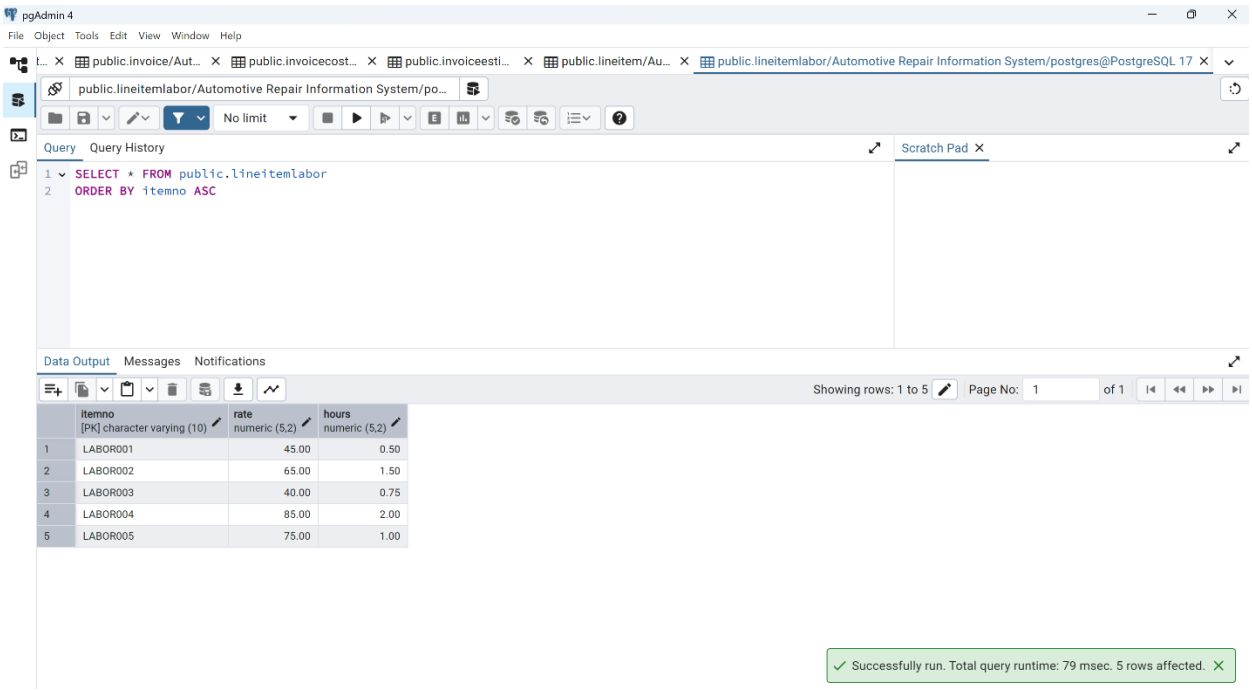
	repairno [PK] character varying (10)	procedureno [PK] character varying (10)	repairdocument text
1	REP001	PROC001	Oil change performed using synthetic 5W-30. All fluid levels checked.
2	REP002	PROC002	Replaced front and rear brake pads. Inspected rotors which are still in good condition.
3	REP003	PROC003	Rotated tires and checked pressure. Recommended alignment for next visit.
4	REP004	PROC004	Performed full engine tune-up. Replaced spark plugs and air filter.
5	REP005	PROC005	Flushed and replaced transmission fluid. Adjusted shift points.

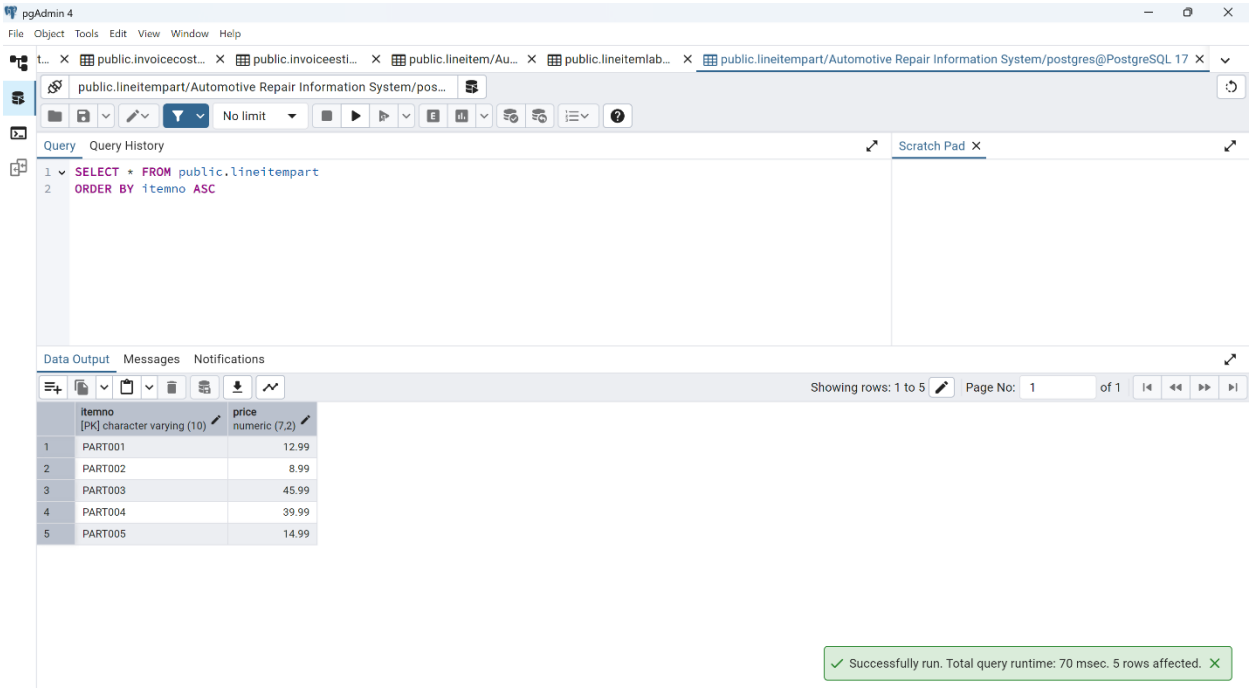

pgAdmin 4
File Object Tools Edit View Window Help
Welcome
Automotive Repair...
Automotive Repair...
public.choice/Auto...
public.documentat...
public.invoice/Automotive Repair Information System/postgres@PostgreSQL 17
public.invoice/Automotive Repair Information System/postgres...
No limit
Query
Query History
Scratch Pad
1 SELECT * FROM public.invoice
2 ORDER BY invoiceno ASC
Data Output
Messages
Notifications
Showing rows: 1 to 6
Page No: 1 of 1
invoiceno [PK] character varying (10)
odometerstart integer
invoiceprintdate date
repairno character varying (10)
1 INV001 35000 2024-05-01 REP001
2 INV002 42500 2024-05-06 REP002
3 INV003 28750 2024-05-10 REP003
4 INV004 51200 2024-06-17 REP004
5 INV005 67300 2024-07-20 REP005
6 INV006 37500 2024-08-01 REP006
Successfully run. Total query runtime: 77 msec. 6 rows affected.

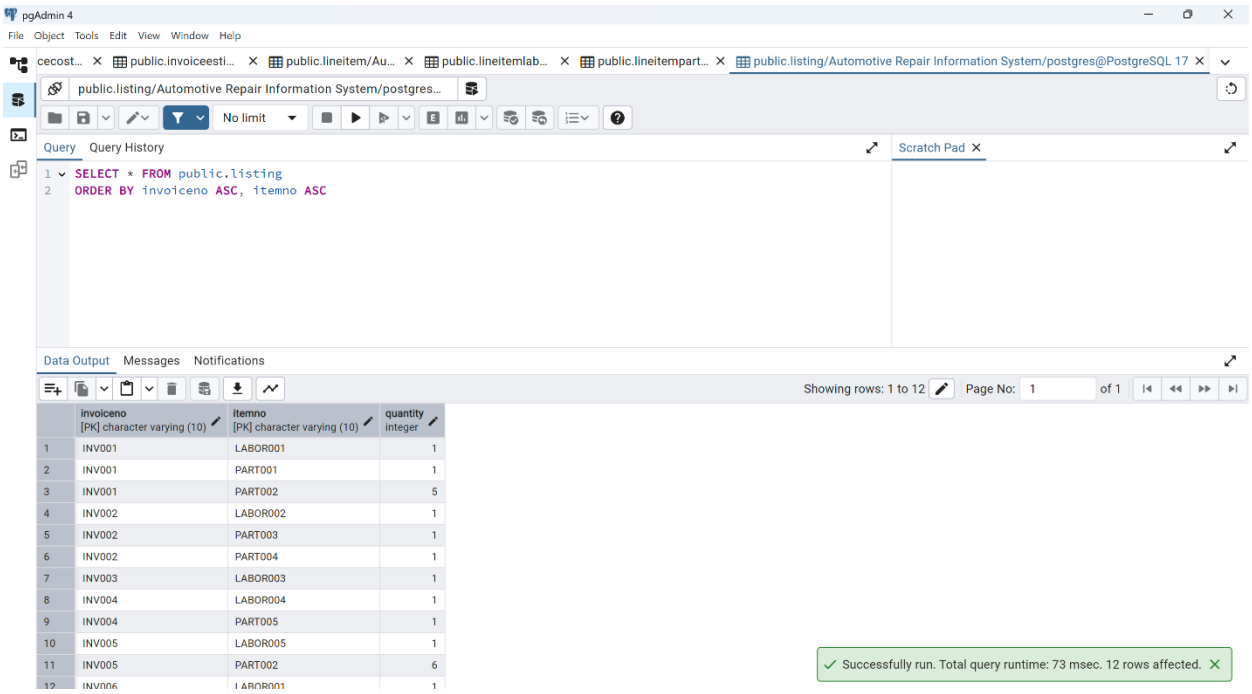

pgAdmin 4
File Object Tools Edit View Window Help
Automotive Repair...
public.choice/Auto...
public.documentat...
public.invoice/Aut...
public.invoicecost/Automotive Repair Information System/
public.invoicecost/Automotive Repair Information System/post...
No limit
Query
Query History
Scratch Pad
1 SELECT * FROM public.invoicecost
2 ORDER BY invoiceno ASC
Data Output
Messages
Notifications
Showing rows: 1 to 5
Page No: 1
invoiceno [PK] character varying (10)
odometerfinish integer
paymentdate date
1 INV001 35010 2024-05-01
2 INV002 42510 2024-05-06
3 INV003 28760 2024-05-10
4 INV004 51220 2024-06-17
5 INV005 67310 2024-07-20

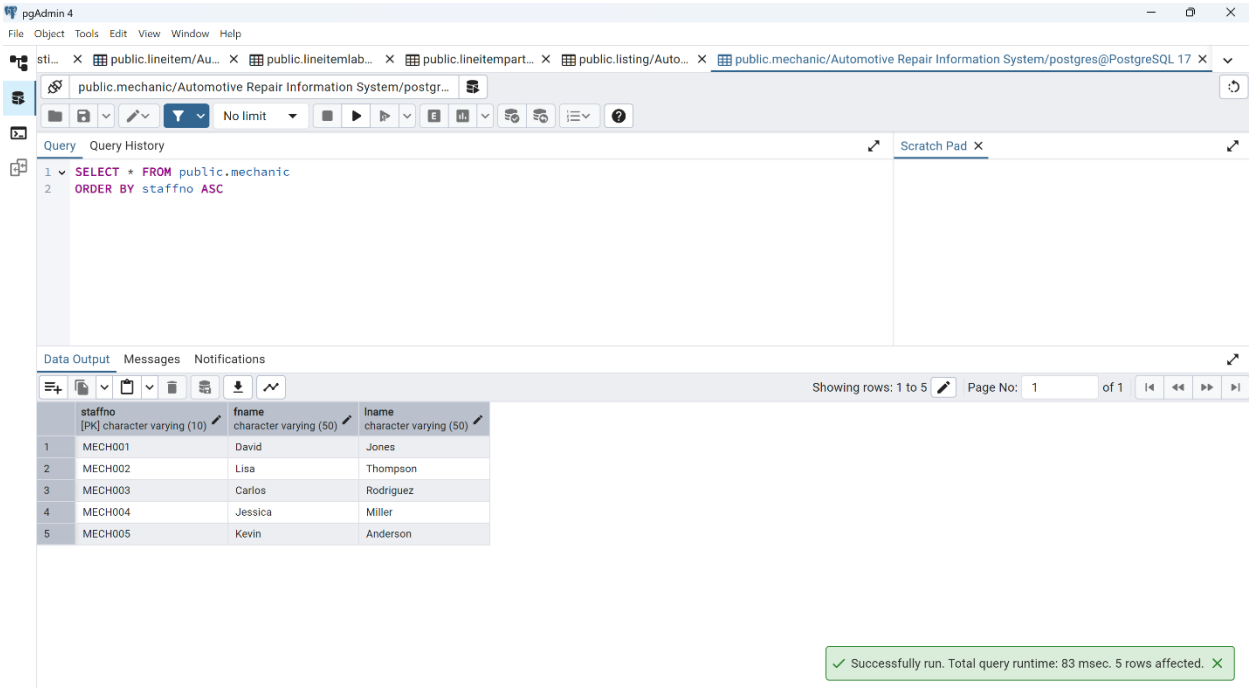

pgAdmin 4
File Object Tools Edit View Window Help
public.choice/Auto...
public.documentat...
public.invoice/Aut...
public.invoicecost...
public.invoiceestimate/Automotive Repair Information System/postgres@PostgreSQL 17
public.invoiceestimate/Automotive Repair Information System/...
No limit
Query
Query History
Scratch Pad
1 SELECT * FROM public.invoiceestimate
2 ORDER BY invoiceno ASC
Data Output
Messages
Notifications
Showing rows: 1 to 1
Page No: 1
of 1
invoiceno
[PK] character varying (10)
1 INV006
Successfully run. Total query runtime: 68 msec. 1 rows affected.

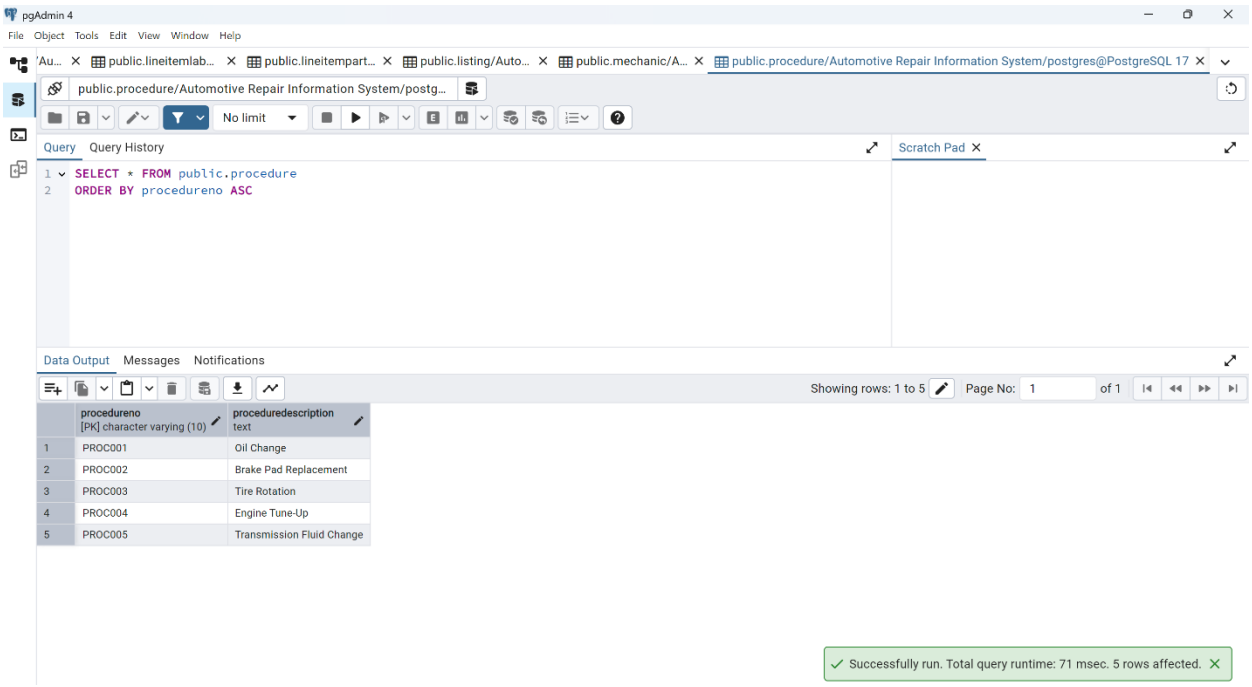

pgAdmin 4
File Object Tools Edit View Window Help
/Auto...
public.documentat...
public.invoice/Aut...
public.invoicecost...
public.invoiceesti...
public.lineitem/Automotive Repair Information System/postgres@PostgreSQL 17
public.lineitem/Automotive Repair Information System/postgre...
No limit
Query
Query History
Scratch Pad
1 SELECT * FROM public.lineitem
2 ORDER BY itemno ASC
Data Output
Messages
Notifications
Showing rows: 1 to 10
Page No: 1
of 1
itemno
[PK] character varying (10)
itemname
character varying (50)
description
text
1 LABOR001 Standard Oil Change Labor Labor for standard oil change ser...
2 LABOR002 Brake Service Labor Labor for brake pad replacement
3 LABOR003 Tire Service Labor Labor for tire rotation and balanci...
4 LABOR004 Engine Service Labor Labor for engine tune-up
5 LABOR005 Transmission Service Labor Labor for transmission services
6 PART001 Oil Filter Standard oil filter for most vehicles
7 PART002 Engine Oil (5W-30) Synthetic oil 5W-30 grade, 1 quart
8 PART003 Brake Pads (Front) Standard front brake pads set
9 PART004 Brake Pads (Rear) Standard rear brake pads set
10 PART005 Air Filter Standard air filter for most vehicles
Successfully run. Total query runtime: 90 msec. 10 rows affected.

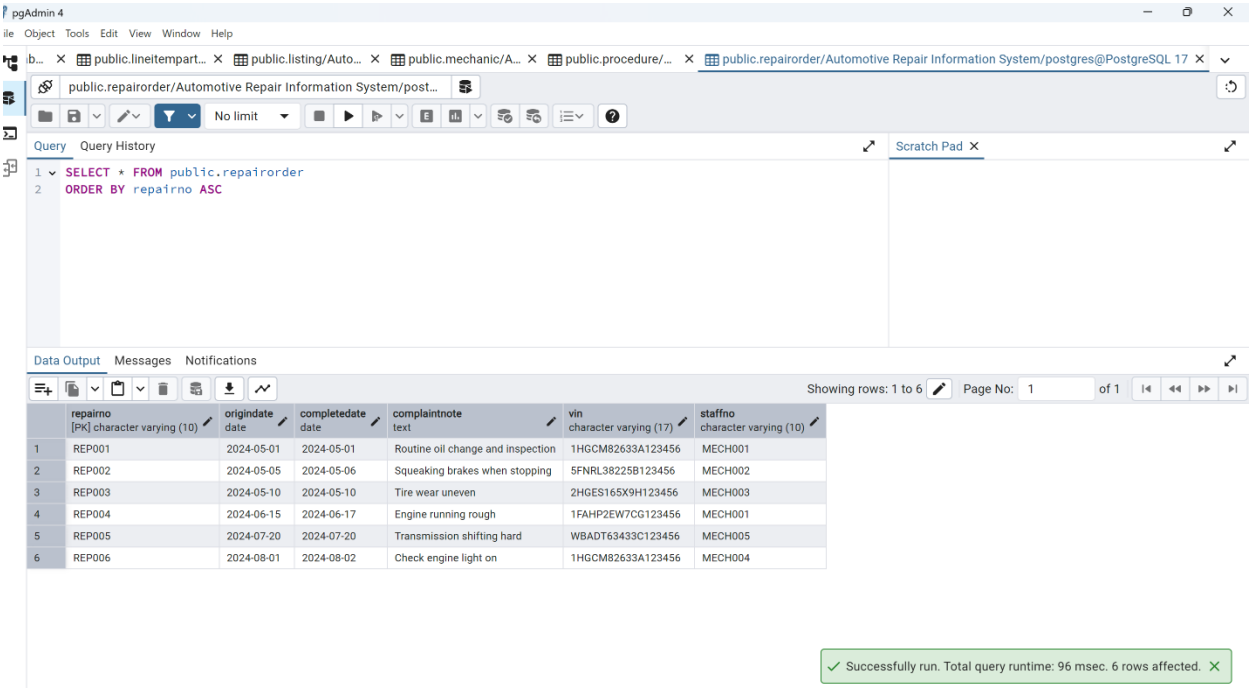

pgAdmin 4
File Object Tools Edit View Window Help
public.lineitemlabor/Automotive Repair Information System/postgres@PostgreSQL 17
No limit
Query Query History
Scratch Pad
SELECT * FROM public.lineitemlabor
ORDER BY itemno ASC
Data Output Messages Notifications
Showing rows: 1 to 5 Page No: 1 of 1
itemno [PK] character varying (10)
rate numeric (5,2)
hours numeric (5,2)
1 LABOR001 45.00 0.50
2 LABOR002 65.00 1.50
3 LABOR003 40.00 0.75
4 LABOR004 85.00 2.00
5 LABOR005 75.00 1.00
Successfully run. Total query runtime: 79 msec. 5 rows affected.

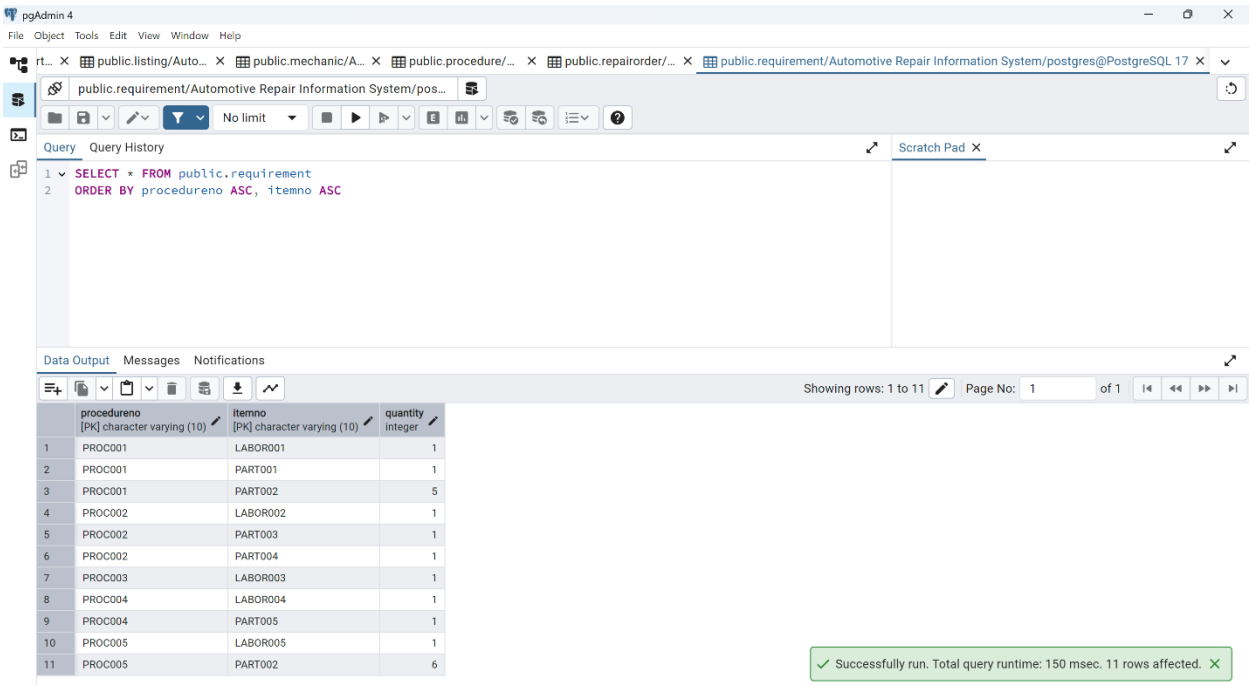

pgAdmin 4
File Object Tools Edit View Window Help
public.lineitempart/Automotive Repair Information System/postgres@PostgreSQL 17
No limit
Query Query History
Scratch Pad
SELECT * FROM public.lineitempart
ORDER BY itemno ASC
Data Output Messages Notifications
Showing rows: 1 to 5 Page No: 1 of 1
itemno [PK] character varying (10)
price numeric (7,2)
1 PART001 12.99
2 PART002 8.99
3 PART003 45.99
4 PART004 39.99
5 PART005 14.99
Successfully run. Total query runtime: 70 msec. 5 rows affected.

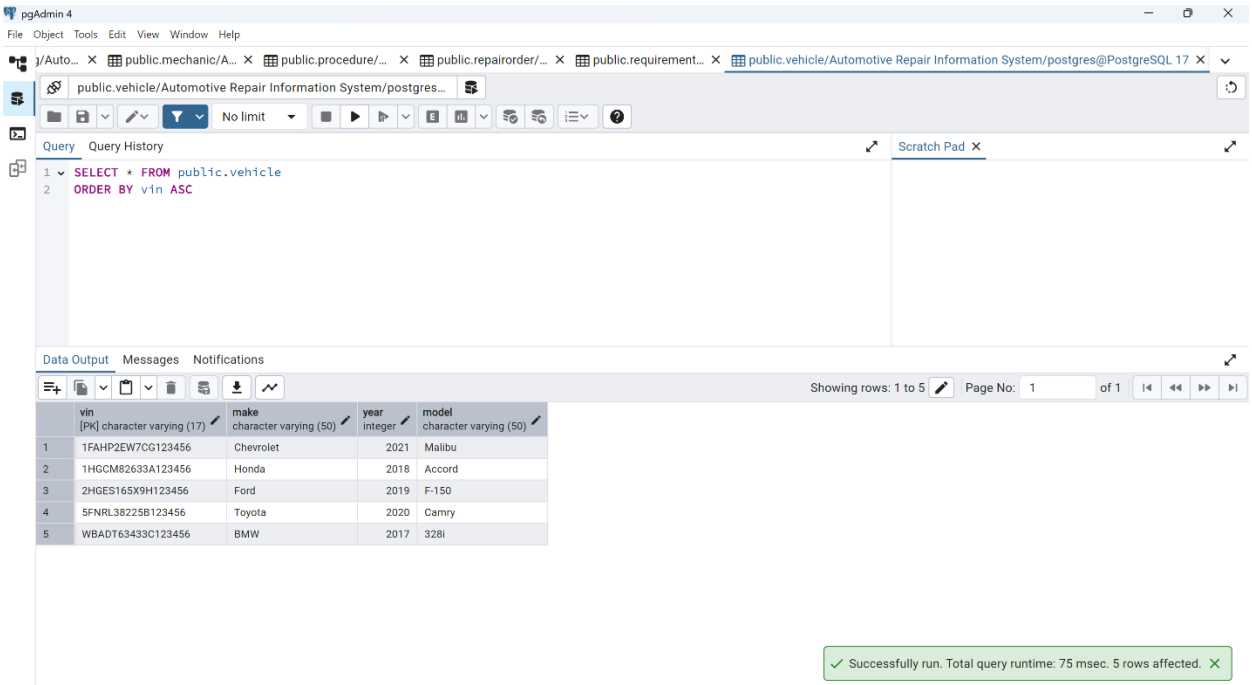

pgAdmin 4
File Object Tools Edit View Window Help
public.listing/Automotive Repair Information System/postgres@PostgreSQL 17
No limit
Query Query History
SELECT * FROM public.listing
ORDER BY invoiceno ASC, itemno ASC
Scratch Pad
Data Output Messages Notifications
Showing rows: 1 to 12 Page No: 1 of 1
invoiceno [PK] character varying (10)
itemno [PK] character varying (10)
quantity integer
1 INV001 LABOR001 1
2 INV001 PART001 1
3 INV001 PART002 5
4 INV002 LABOR002 1
5 INV002 PART003 1
6 INV002 PART004 1
7 INV003 LABOR003 1
8 INV004 LABOR004 1
9 INV004 PART005 1
10 INV005 LABOR005 1
11 INV005 PART002 6
Successfully run. Total query runtime: 73 msec. 12 rows affected.


pgAdmin 4
File Object Tools Edit View Window Help
public.mechanic/Automotive Repair Information System/postgres@PostgreSQL 17
No limit
Query Query History
SELECT * FROM public.mechanic
ORDER BY staffno ASC
Scratch Pad
Data Output Messages Notifications
Showing rows: 1 to 5 Page No: 1 of 1
staffno [PK] character varying (10)
fname character varying (50)
lname character varying (50)
1 MECH001 David Jones
2 MECH002 Lisa Thompson
3 MECH003 Carlos Rodriguez
4 MECH004 Jessica Miller
5 MECH005 Kevin Anderson
Successfully run. Total query runtime: 83 msec. 5 rows affected.


pgAdmin 4
File Object Tools Edit View Window Help
public.lineitemlab...
public.lineitempart...
public.listing/Auto...
public.mechanic/A...
public.procedure/Automotive Repair Information System/postgres@PostgreSQL 17
public.procedure/Automotive Repair Information System/postg...
No limit
Query
Query History
Scratch Pad
1 SELECT * FROM public.procedure
2 ORDER BY procedureno ASC
Data Output
Messages
Notifications
Showing rows: 1 to 5
Page No: 1
of 1
procedureno [PK] character varying (10)
proceduredescription text
1 PROC001 Oil Change
2 PROC002 Brake Pad Replacement
3 PROC003 Tire Rotation
4 PROC004 Engine Tune-Up
5 PROC005 Transmission Fluid Change
Successfully run. Total query runtime: 71 msec. 5 rows affected.


pgAdmin 4
File Object Tools Edit View Window Help
public.lineitempart...
public.listing/Auto...
public.mechanic/A...
public.procedure/...
public.repairorder/Automotive Repair Information System/postgres@PostgreSQL 17
public.repairorder/Automotive Repair Information System/post...
No limit
Query
Query History
Scratch Pad
1 SELECT * FROM public.repairorder
2 ORDER BY repairno ASC
Data Output
Messages
Notifications
Showing rows: 1 to 6
Page No: 1
of 1
repairno [PK] character varying (10)
origindate date
completedate date
complaintnote text
vin character varying (17)
staffno character varying (10)
1 REP001 2024-05-01 2024-05-01 Routine oil change and inspection 1HGCM82633A123456 MECH001
2 REP002 2024-05-05 2024-05-06 Squeaking brakes when stopping 5FNRL38225B123456 MECH002
3 REP003 2024-05-10 2024-05-10 Tire wear uneven 2HGES165X9H123456 MECH003
4 REP004 2024-06-15 2024-06-17 Engine running rough 1FAHP2EW7CG123456 MECH001
5 REP005 2024-07-20 2024-07-20 Transmission shifting hard WBADT63433C123456 MECH005
6 REP006 2024-08-01 2024-08-02 Check engine light on 1HGCM82633A123456 MECH004
Successfully run. Total query runtime: 96 msec. 6 rows affected.


pgAdmin 4
File Object Tools Edit View Window Help
public.listing/Auto...
public.mechanic/A...
public.procedure/...
public.repairorder/...
public.requirement/Automotive Repair Information System/postgres@PostgreSQL 17
public.requirement/Automotive Repair Information System/pos...
No limit
Query
Query History
Scratch Pad
SELECT * FROM public.requirement
ORDER BY procedureno ASC, itemno ASC
Data Output
Messages
Notifications
Showing rows: 1 to 11
Page No: 1
of 1
procedureno [PK] character varying (10)
itemno [PK] character varying (10)
quantity integer
1 PROC001 LABOR001 1
2 PROC001 PART001 1
3 PROC001 PART002 5
4 PROC002 LABOR002 1
5 PROC002 PART003 1
6 PROC002 PART004 1
7 PROC003 LABOR003 1
8 PROC004 LABOR004 1
9 PROC004 PART005 1
10 PROC005 LABOR005 1
11 PROC005 PART002 6
Successfully run. Total query runtime: 150 msec. 11 rows affected.

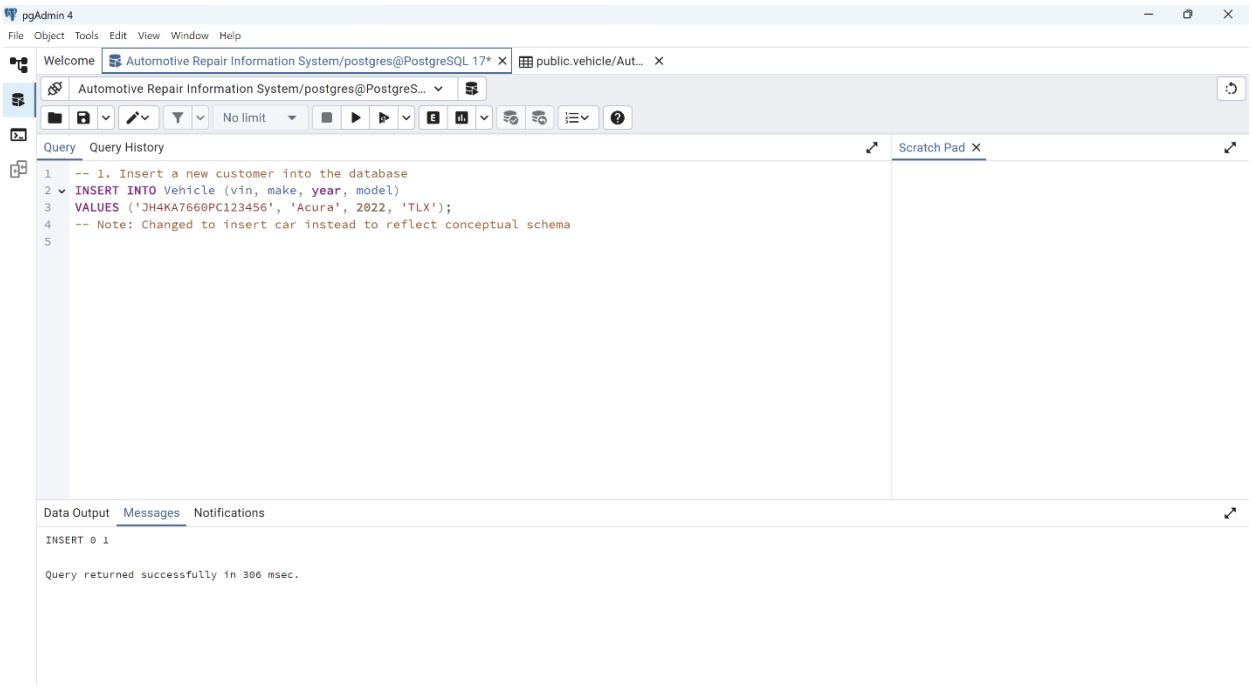

pgAdmin 4
File Object Tools Edit View Window Help
public.mechanic/A...
public.procedure/...
public.repairorder/...
public.requirement...
public.vehicle/Automotive Repair Information System/postgres@PostgreSQL 17
public.vehicle/Automotive Repair Information System/postgres...
No limit
Query
Query History
Scratch Pad
SELECT * FROM public.vehicle
ORDER BY vin ASC
Data Output
Messages
Notifications
Showing rows: 1 to 5
Page No: 1
of 1
vin [PK] character varying (17)
make character varying (50)
year integer
model character varying (50)
1 1FAHP2EW7CG123456 Chevrolet 2021 Malibu
2 1HGCM82633A123456 Honda 2018 Accord
3 2HGES165X9H123456 Ford 2019 F-150
4 5FNRL38225B123456 Toyota 2020 Camry
5 WBADT63433C123456 BMW 2017 328i
Successfully run. Total query runtime: 75 msec. 5 rows affected.

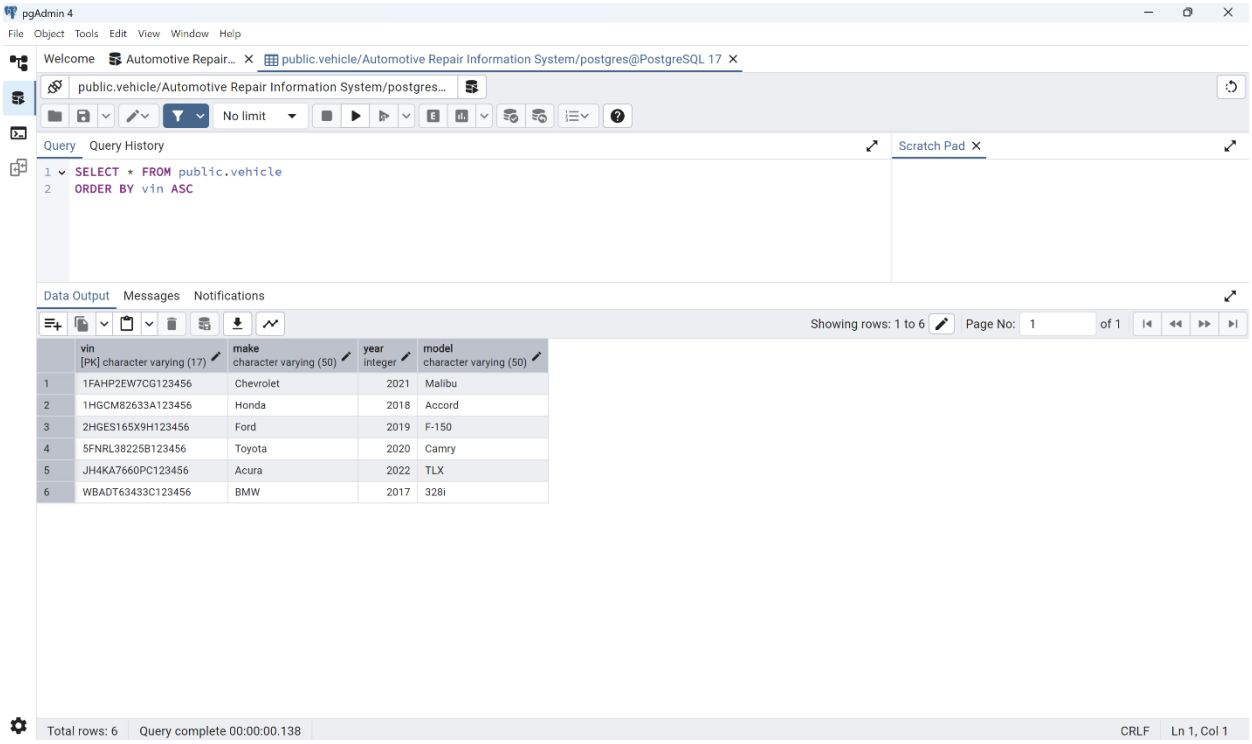
Case Study Application:

1. Insert a new customer into the database.
2. Delete an existing employee from the database.
3. Update the description of an existing repair order.
4. List all the repair orders belonging to a given vehicle, along with their dates when each repair was originated and completed.
5. List the details of the line items including the description, price, and quantity for the invoice(s) of a given repair order.
6. List the repair orders completed between June 2024 and December 2024, sorted by the repair order numbers.
7. List the details of all the line items of a given procedure.
8. List the total number of procedures required by each repair order in descending order, along with the procedure description.
9. List the name of the employee who recorded more than the average number of invoices, together with the number of invoices he/she recorded.
10. For a particular invoice, list the odometer mileages (in and out), payment information, and the vehicle information.

Sample Queries

```
-- 1. Insert a new customer into the database
INSERT INTO Vehicle (vin, make, year, model)
VALUES ('JH4KA7660PC123456', 'Acura', 2022, 'TLX');
-- Note: Changed to insert vehicle instead to reflect conceptual schema
```

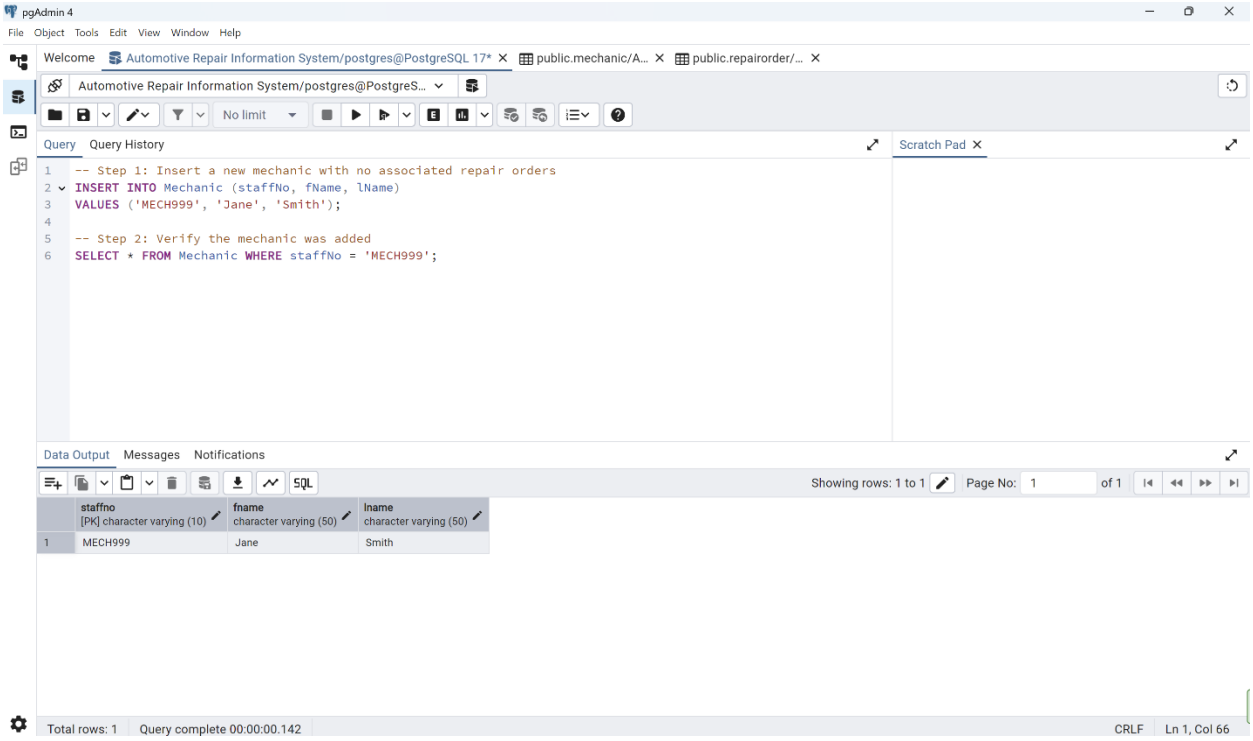

pgAdmin 4
File Object Tools Edit View Window Help
Welcome
Automotive Repair Information System/postgres@PostgreSQL 17*
public.vehicle/Aut...
Automotive Repair Information System/postgres@PostgreS...
No limit
Query
Query History
Scratch Pad
1 -- 1. Insert a new customer into the database
2 INSERT INTO Vehicle (vin, make, year, model)
3 VALUES ('JH4KA7660PC123456', 'Acura', 2022, 'TLX');
4 -- Note: Changed to insert car instead to reflect conceptual schema
5
Data Output
Messages
Notifications
INSERT 0 1
Query returned successfully in 306 msec.

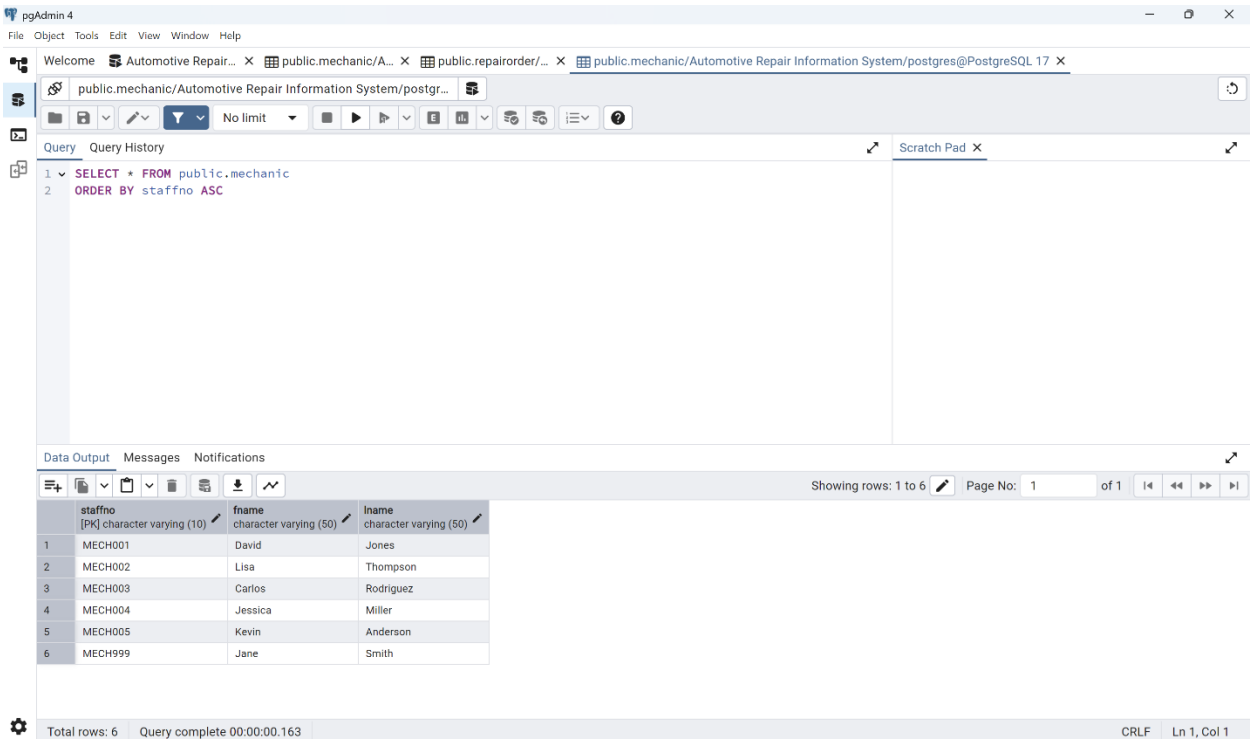

pgAdmin 4
File Object Tools Edit View Window Help
Welcome
Automotive Repair...
public.vehicle/Automotive Repair Information System/postgres@PostgreSQL 17
public.vehicle/Automotive Repair Information System/postgres...
No limit
Query
Query History
Scratch Pad
1 SELECT * FROM public.vehicle
2 ORDER BY vin ASC
Data Output
Messages
Notifications
Showing rows: 1 to 6
Page No: 1
of 1
vin [PK] character varying (17)
make character varying (50)
year integer
model character varying (50)
1 1FAHP2EW7CG123456 Chevrolet 2021 Malibu
2 1HGCM82633A123456 Honda 2018 Accord
3 2HGES165X9H123456 Ford 2019 F-150
4 5FNRL38225B123456 Toyota 2020 Camry
5 JH4KA7660PC123456 Acura 2022 TLX
6 WBADT63433C123456 BMW 2017 328i
Total rows: 6
Query complete 00:00:00.138
CRLF
Ln 1, Col 1

-- 2. Delete an existing employee from the database. This only works if the mechanic doesn’t have a repair order, so I will insert a new mechanic without and repair orders to show it being deleted.

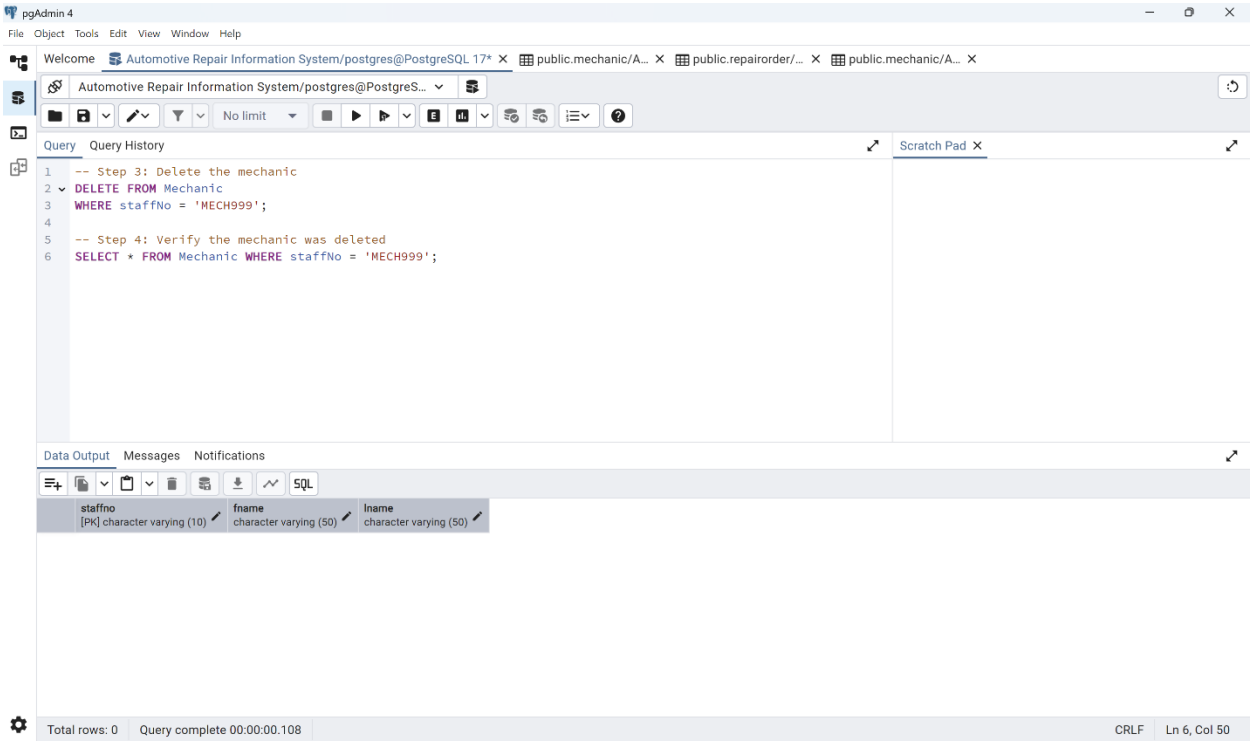
-- Step 1: Insert a new mechanic with no associated repair orders.
INSERT INTO Mechanic (staffNo, fName, lName) VALUES ('MECH999', 'Jane', 'Smith');

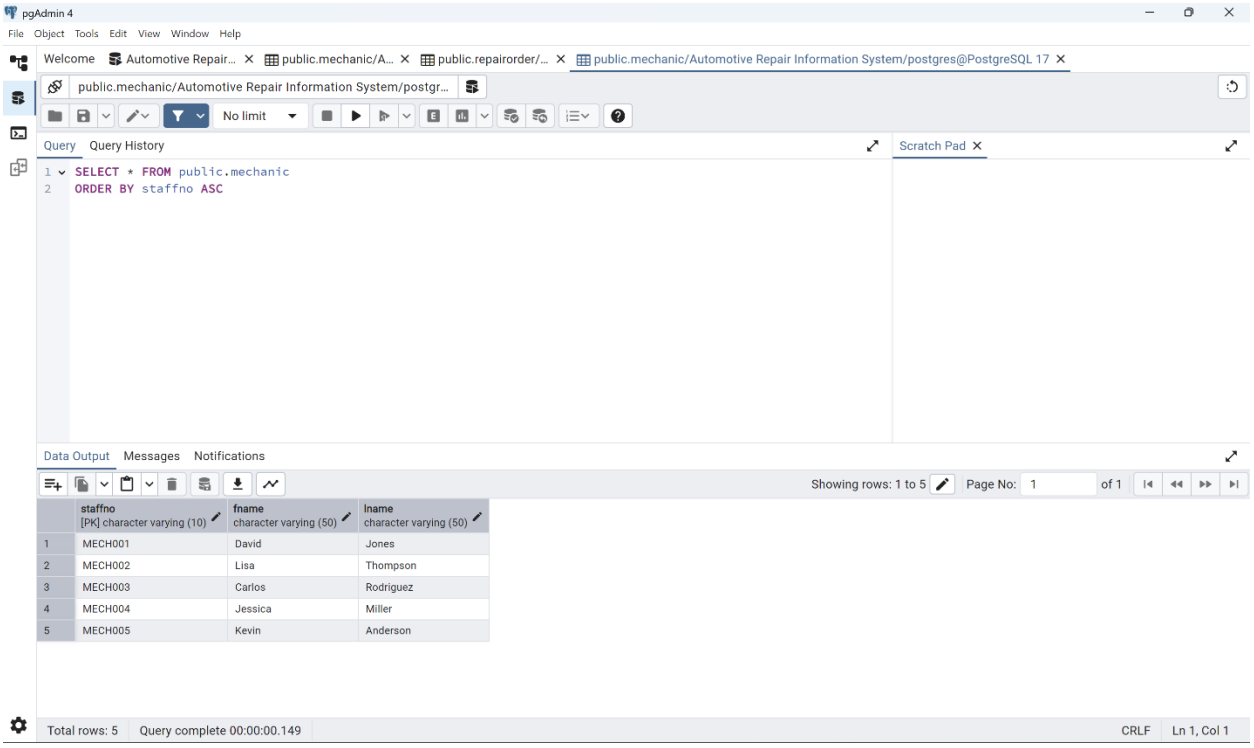
-- Step 2: Verify the mechanic was added
SELECT * FROM Mechanic WHERE staffNo = 'MECH999';



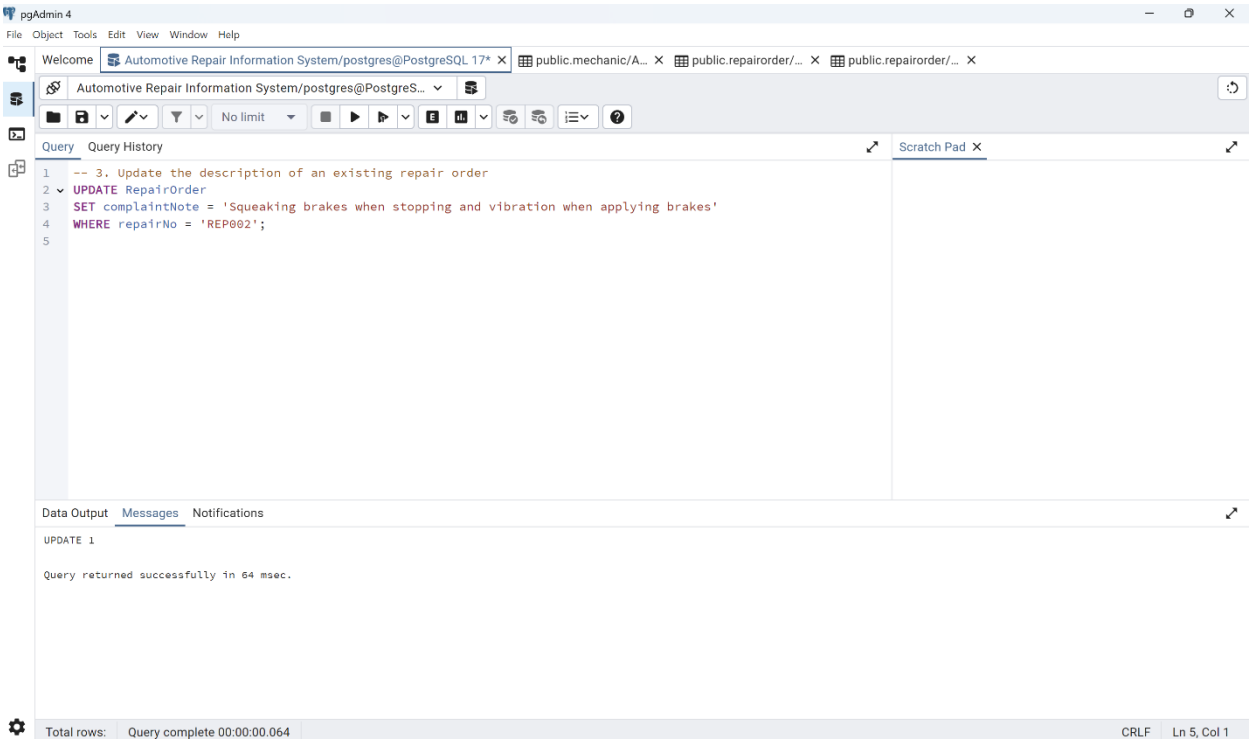


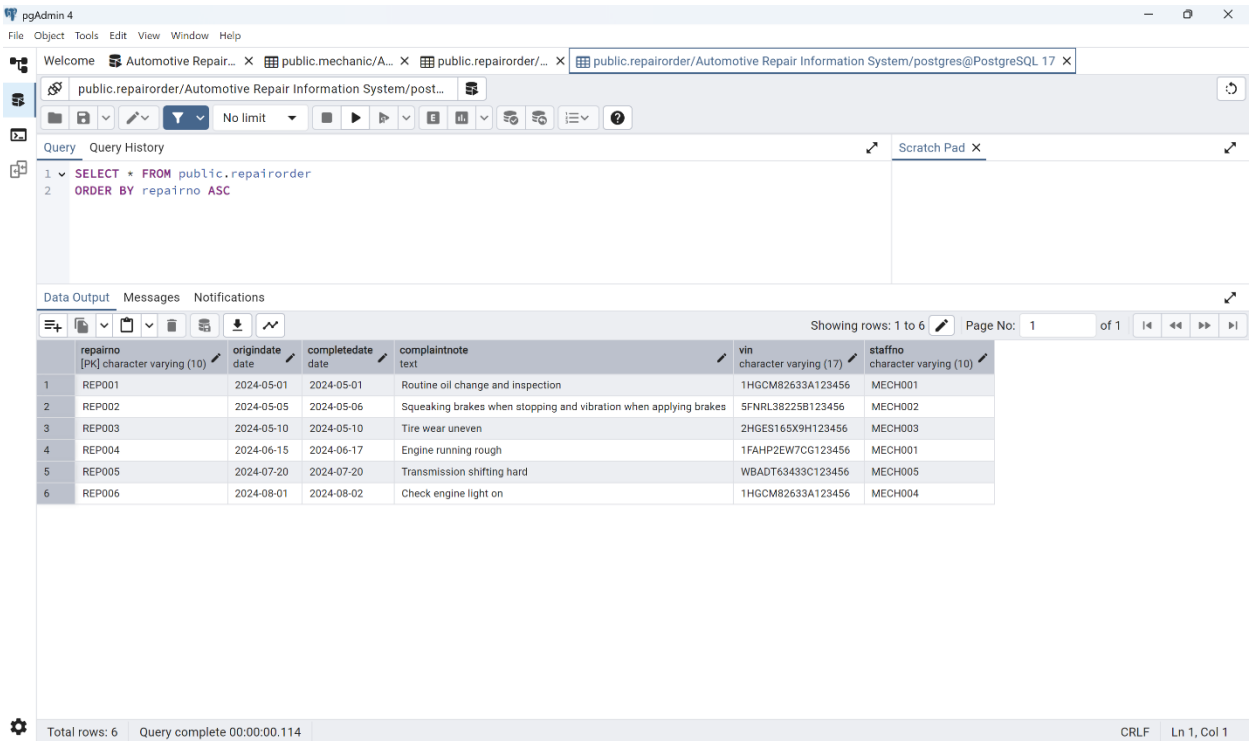

```
-- Step 3: Delete the mechanic
DELETE FROM Mechanic WHERE staffNo = 'MECH999';
 -- Step 4: Verify the mechanic was deleted
SELECT * FROM Mechanic WHERE staffNo = 'MECH999';
```


pgAdmin 4
File Object Tools Edit View Window Help
Welcome
Automotive Repair Information System/postgres@PostgreSQL 17*
public.mechanic/A...
public.repairorder/...
public.mechanic/A...
Automotive Repair Information System/postgres@PostgreS...
No limit
Query
Query History
Scratch Pad
1 -- Step 3: Delete the mechanic
2 DELETE FROM Mechanic
3 WHERE staffNo = 'MECH999';
4
5 -- Step 4: Verify the mechanic was deleted
6 SELECT * FROM Mechanic WHERE staffNo = 'MECH999';
Data Output
Messages
Notifications
SQL
staffno
[PK] character varying (10)
fname
character varying (50)
lname
character varying (50)
Total rows: 0
Query complete 00:00:00.108
CRLF
Ln 6, Col 50

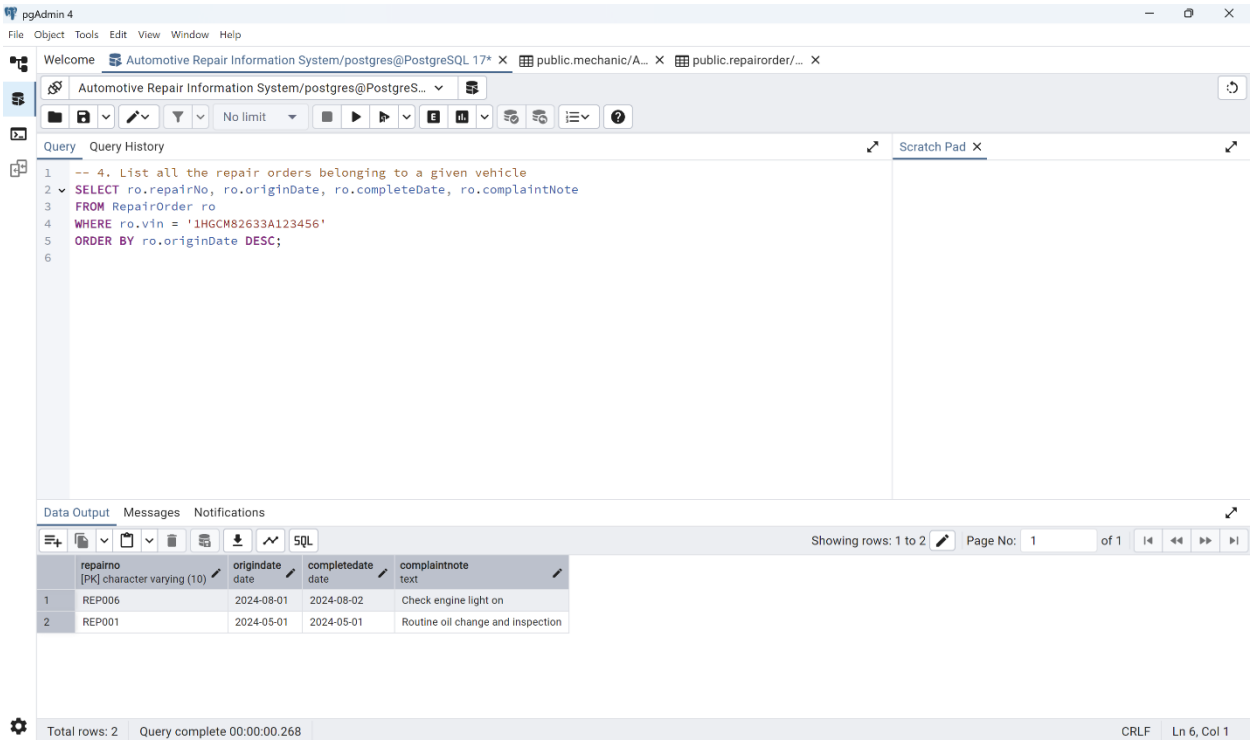

pgAdmin 4
File Object Tools Edit View Window Help
Welcome
Automotive Repair...
public.mechanic/A...
public.repairorder/...
public.mechanic/Automotive Repair Information System/postgres@PostgreSQL 17
public.mechanic/Automotive Repair Information System/postgr...
No limit
Query
Query History
Scratch Pad
1 SELECT * FROM public.mechanic
2 ORDER BY staffno ASC
Data Output
Messages
Notifications
Showing rows: 1 to 5
Page No: 1
of 1
staffno
[PK] character varying (10)
fname
character varying (50)
lname
character varying (50)
1 MECH001 David Jones
2 MECH002 Lisa Thompson
3 MECH003 Carlos Rodriguez
4 MECH004 Jessica Miller
5 MECH005 Kevin Anderson
Total rows: 5
Query complete 00:00:00.149
CRLF
Ln 1, Col 1

-- 3. Update the description of an existing repair order
UPDATE RepairOrder
SET complaintNote = 'Squeaking brakes when stopping and vibration when applying brakes'
WHERE repairNo = 'REP002';

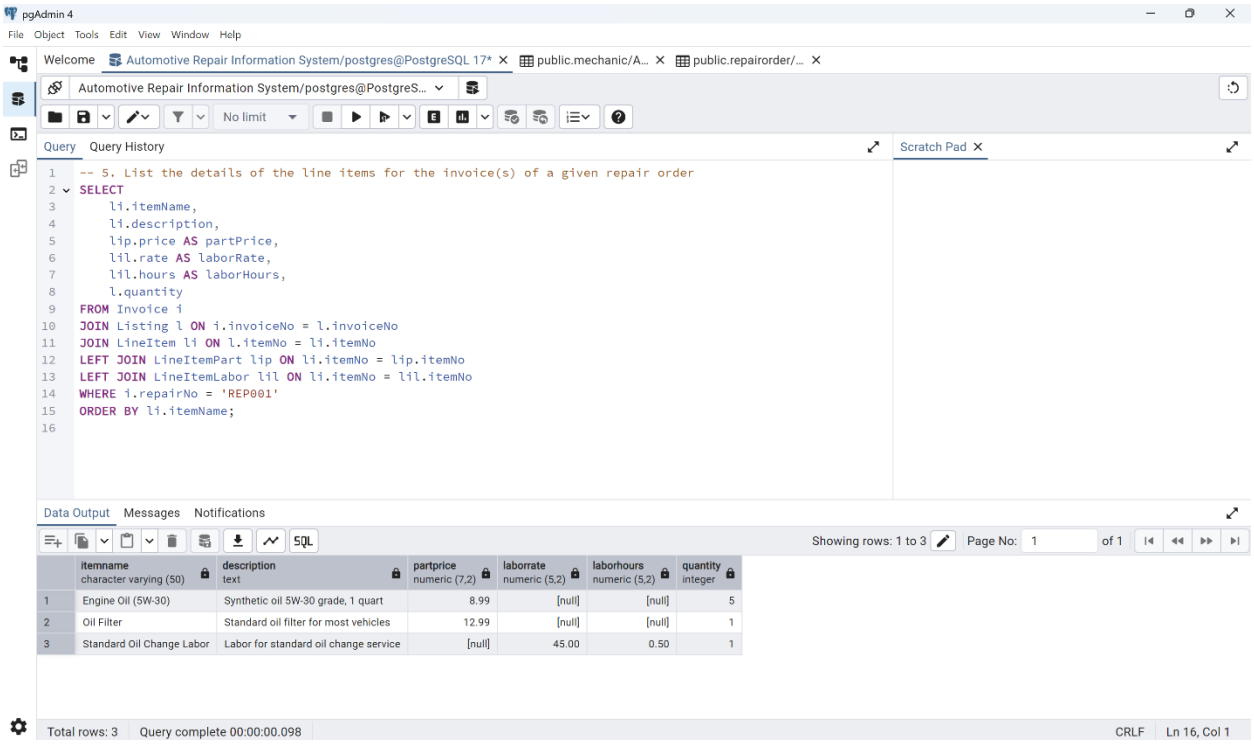




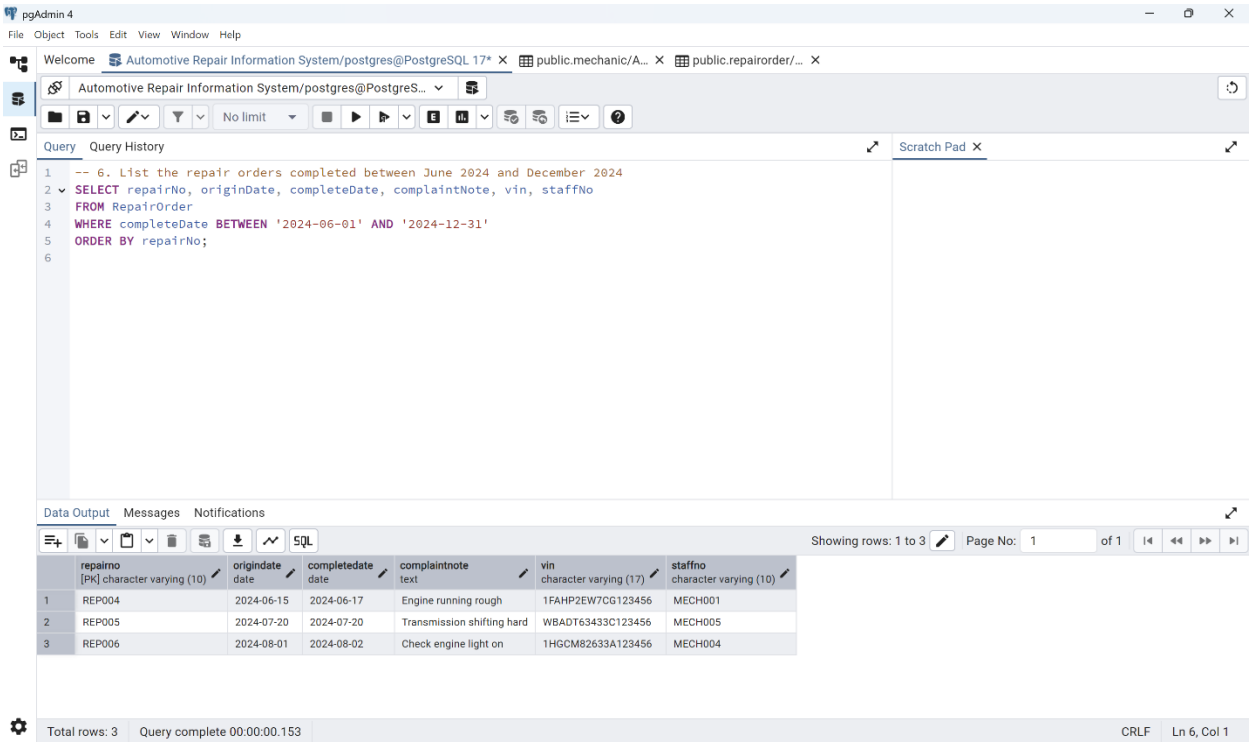

```
-- 4. List all the repair orders belonging to a given vehicle
SELECT ro.repairNo, ro.originDate, ro.completeDate, ro.complaintNote
FROM RepairOrder ro
WHERE ro.vin = '1HGCM82633A123456'
ORDER BY ro.originDate DESC;
```



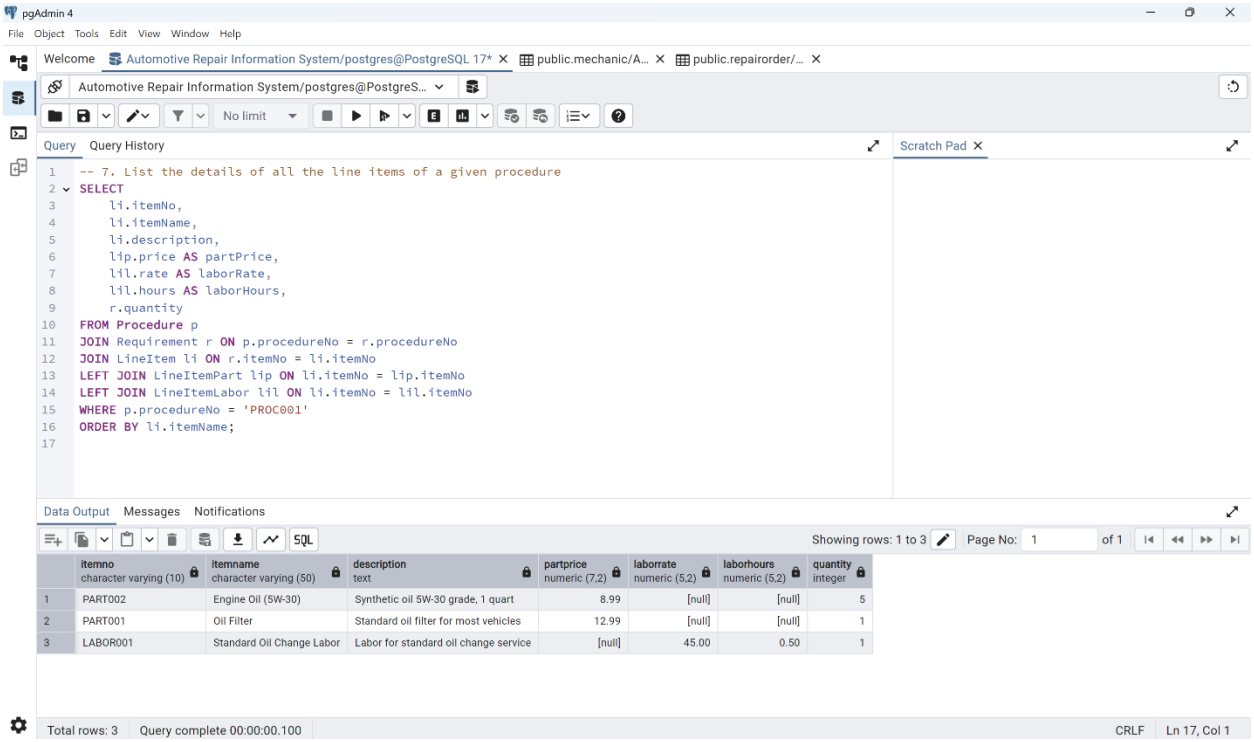

```
-- 5. List the details of the line items for the invoice(s) of a given repair order
SELECT
   li.itemName,
   li.description,
   lip.price AS partPrice,
   lil.rate AS laborRate,
   lil.hours AS laborHours,
   l.quantity
FROM Invoice i
JOIN Listing l ON i.invoiceNo = l.invoiceNo
JOIN LineItem li ON l.itemNo = li.itemNo
LEFT JOIN LineItemPart lip ON li.itemNo = lip.itemNo
LEFT JOIN LineItemLabor lil ON li.itemNo = lil.itemNo
WHERE i.repairNo = 'REP001'
ORDER BY li.itemName;
```



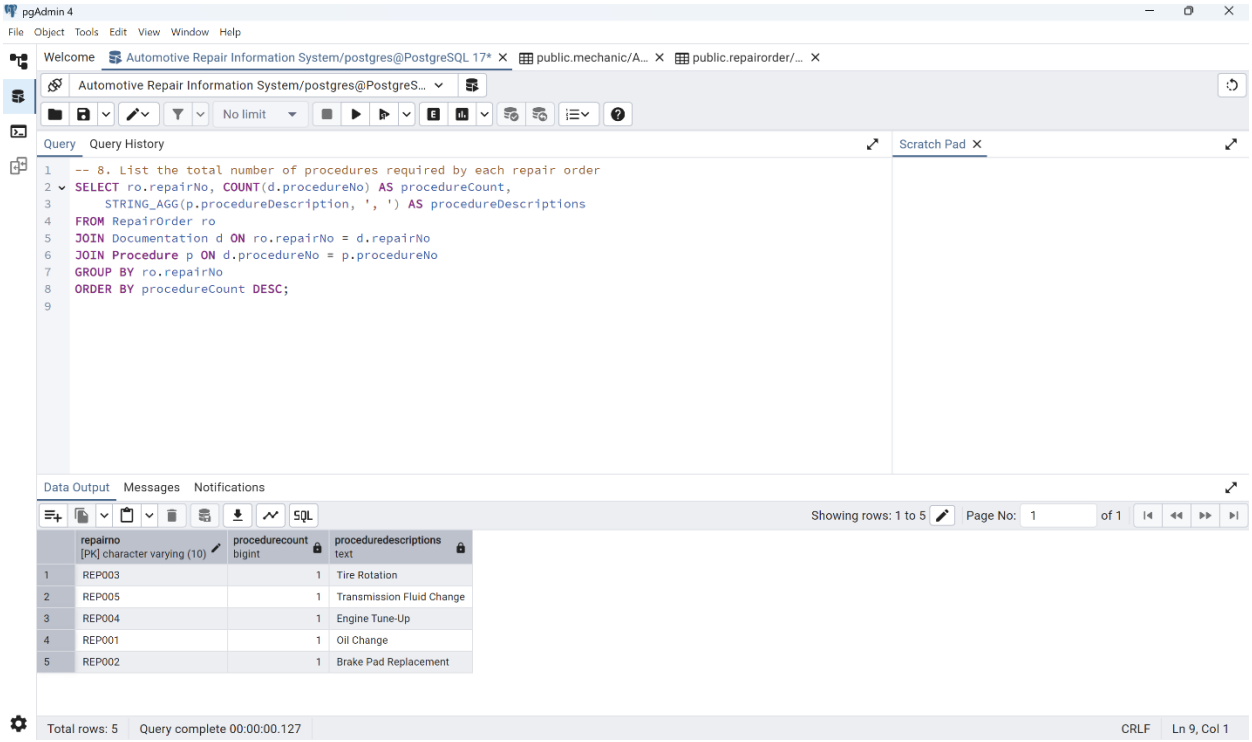
-- 6. List the repair orders completed between June 2024 and December 2024
SELECT repairNo, originDate, completeDate, complaintNote, vin, staffNo
FROM RepairOrder
WHERE completeDate BETWEEN '2024-06-01' AND '2024-12-31'
ORDER BY repairNo;




```
-- 7. List the details of all the line items of a given procedure
SELECT
   li.itemNo,
   li.itemName,
   li.description,
   lip.price AS partPrice,
   lil.rate AS laborRate,
   lil.hours AS laborHours,
   r.quantity
FROM Procedure p
JOIN Requirement r ON p.procedureNo = r.procedureNo
JOIN LineItem li ON r.itemNo = li.itemNo
LEFT JOIN LineItemPart lip ON li.itemNo = lip.itemNo
LEFT JOIN LineItemLabor lil ON li.itemNo = lil.itemNo
WHERE p.procedureNo = 'PROC001'
ORDER BY li.itemName;
```

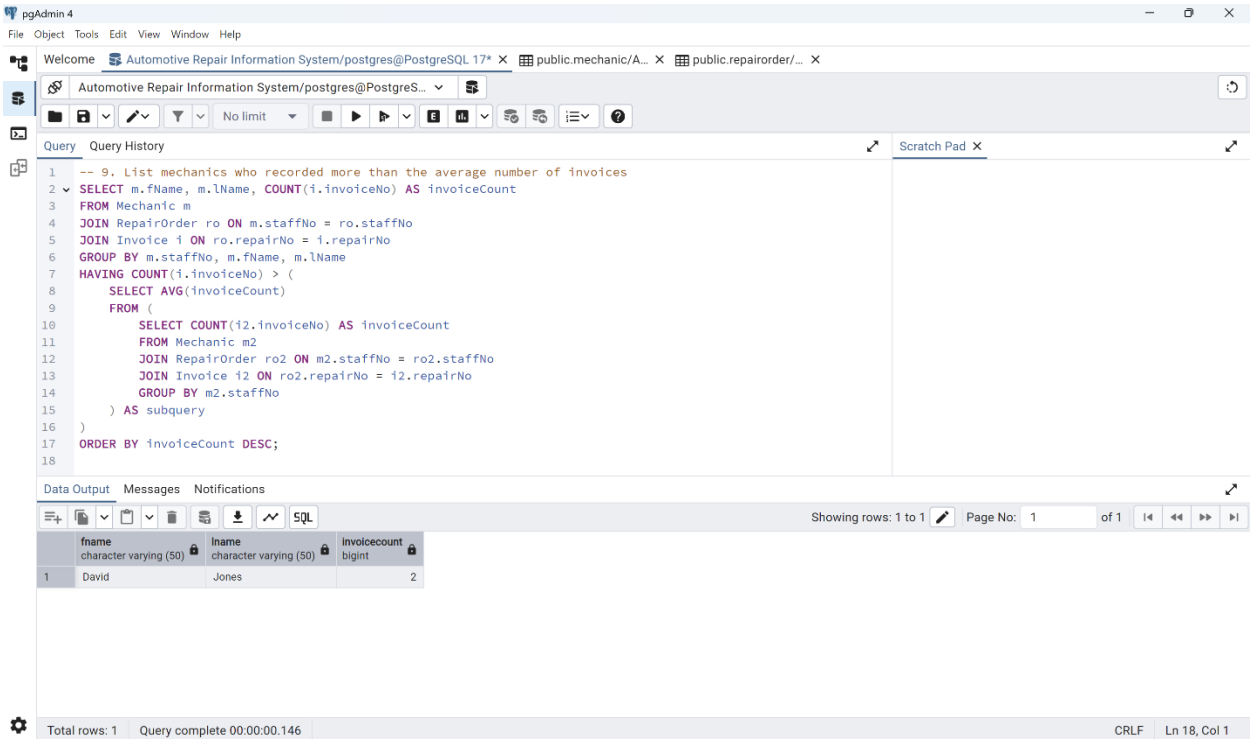



```
-- 8. List the total number of procedures required by each repair order
SELECT ro.repairNo, COUNT(d.procedureNo) AS procedureCount,
   STRING_AGG(p.procedureDescription, ', ') AS procedureDescriptions
FROM RepairOrder ro
JOIN Documentation d ON ro.repairNo = d.repairNo
JOIN Procedure p ON d.procedureNo = p.procedureNo
GROUP BY ro.repairNo
ORDER BY procedureCount DESC;
```



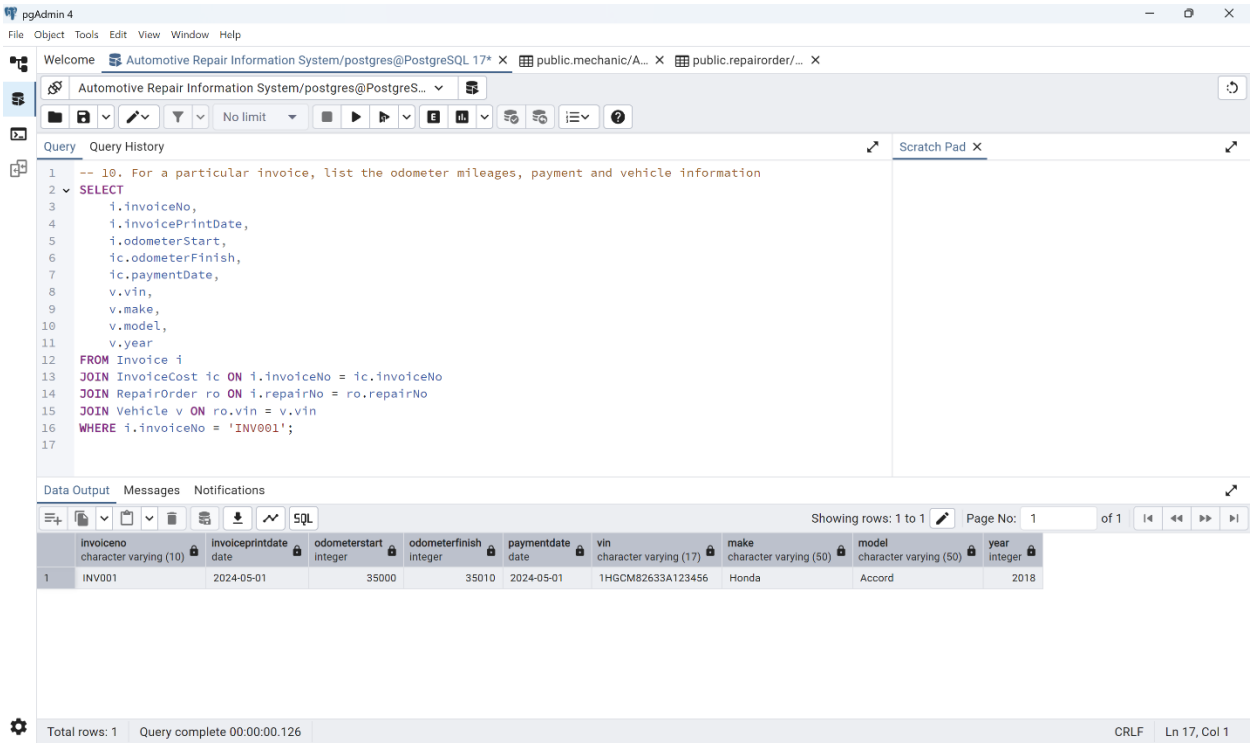

```
-- 9. List mechanics who recorded more than the average number of invoices
SELECT m.fName, m.lName, COUNT(i.invoiceNo) AS invoiceCount
FROM Mechanic m
JOIN RepairOrder ro ON m.staffNo = ro.staffNo
JOIN Invoice i ON ro.repairNo = i.repairNo
GROUP BY m.staffNo, m.fName, m.lName
HAVING COUNT(i.invoiceNo) > (
  SELECT AVG(invoiceCount)
  FROM (
    SELECT COUNT(i2.invoiceNo) AS invoiceCount
    FROM Mechanic m2
    JOIN RepairOrder ro2 ON m2.staffNo = ro2.staffNo
    JOIN Invoice i2 ON ro2.repairNo = i2.repairNo
    GROUP BY m2.staffNo
  ) AS subquery
)
```

ORDER BY invoiceCount DESC;



-- 10. For a particular invoice, list the odometer mileages, payment and vehicle information
SELECT
i.invoiceNo,
i.invoicePrintDate,
i.odometerStart,
ic.odometerFinish,
ic.paymentDate,
v.vin,
v.make,
v.model,
v.year
FROM Invoice i
JOIN InvoiceCost ic ON i.invoiceNo = ic.invoiceNo
JOIN RepairOrder ro ON i.repairNo = ro.repairNo

JOIN Vehicle v ON ro.vin = v.vin
WHERE i.invoiceNo = 'INV001';



Conclusion

This report covered the logical data model and conceptual data model, with the EER diagrams for both of them included. There were also corrections and slight changes. Compared to the original logical and conceptual model, the following changes were made:

- Changed the data type for the “description” attribute in the **LineItem** table to “TEXT” instead of “INTEGER”.
- Changed the data type of the “invoicePrintDate” attribute for the **Invoice** table to “DATE” instead of “INTEGER”.
- Added a “quantity” attribute to the **Requirement** table.
- Corrected some of the Entity names to be the same as they are in the EER diagrams (Requirement and Documentation).
- Updated “vin” to be the primary key of the **Vehicle** entity and removed “vehicleNo” in the Logical data model.

The test data and SQL statements are included in the form of both typed documentation as well as screenshots. All tables were created and test data was inserted successfully. Instead of inserting a customer as the original instructions specified, a customer’s vehicle was inserted to align with the conceptual schema. The sample queries were completed successfully and are given in typed form and as screenshots. The sample output is also included in the screenshots.