

Work Order Processing Application Database

Technical Document

Matthew Arbore

Purdue University Global

IT 499 Capstone in Information Technology

Voyage Bicycle Organization Work Order Processing Application Database.

Table of Contents

1. Executive Summary
2. Database Requirements
3. Logical Database Design Plan
4. Physical Database Design Plan
5. Sample Data
6. Testing and Validation
7. Queries
8. Work Order Report
9. Database Access Control
10. Database Migration

1. Executive Summary

Voyager Bicycle Organization requested our project team to deliver a relational Microsoft SQL Server Database to provide a centralized solution for an application they are developing to manage work orders across several retail locations across the country. This database provides a centralized solution for managing customers, retail locations, technicians, work orders and billing data for the application they are developing. Our project team was requested to provide Voyager Bicycle Organization with the necessary queries that can support essential functions for their application as well as providing a method to print a 'Work Order Reports' that technicians and customers can have access to through the application being developed by Voyager Bicycle Organization's IT team. Based on our knowledge of the application Voyager Bicycle Organization's IT team is developing, this database should provide the framework for their application to call the predefined queries to print the data they are looking for within their application. Our project team has included the defined requirements, logical design plans, and physical design plans in this document. We have also included the SQL code and statements used to build the database environment, constraints, and relationships. The database includes all organizational data provided by Voyager Bicycle Organization's IT team. The database went through testing to validate all relationships work as intended. Three essential queries for the application have been defined and the 'Work Order Report' code and statements are included in this document. The database supports referential integrity and uses role-based access control to protect the database, and the information stored within the database.

2. Database Requirements

Functional Requirements

1. Customer Management:

- 1.1. Store and manage information for each retail chain customer that includes:
 - Customer name, billing address, billing contact, and contact information for service locations.

2. Work Order Process

- 2.1. Generate, store, and manage work orders with unique IDs.
- 2.2. Include fields for:
 - Customer Details (Billing and Service).
 - Field technician assignment and rate.
 - Work Order descriptions (Problem description, technician solution, work completed with hours spent).

3. Field Technician Management

- 3.1. Track technician details, including name, contact information, rate and availability.
- 3.2. Associate technicians with specific work orders.

4. Billing Process

- 4.1. Calculate total billing amount per work order based on:
 - Technician's rate and time.
 - Client-specific markup.
- 4.2. Generate Invoices.

5. Reporting

- 5.1. Provide reports on Technician's work.
- 5.2. Customer Billing summary.
- 5.3. Status of work order (pending/completed).

6. User Roles and Permissions

- 6.1. Differentiate roles such as Admin and Technician.
- 6.2. Restrict data based on Roles.

Nonfunctional Requirements

1. Usability

- 1.1. Minimal Training required for technicians.
- 1.2. Support Voyage Bicycle Application functions.

2. Scalability

- 2.1 Ability to handle increased data load as more customers, locations, and technicians are added.

3. Availability

- 3.1. Ensure 99.9% system uptime to minimize disruptions.

Performance Requirements

1. Response Time

- 1.1. Query response time for retrieving or updating a work order should not exceed 4 seconds.
- 1.2. Real-time updates for technicians in the field should sync within 6 seconds.

2. User Support

- 2.1. Support up to 500 simultaneous users without performance issues.
- 2.2. Handle data for at least 1,000 work orders with efficient indexing.

Security Requirements

1. Access Control

- 1.1. Referential integrity enforcement where necessary.
- 1.2. Implement least privilege access control.

Cultural/Political Requirements

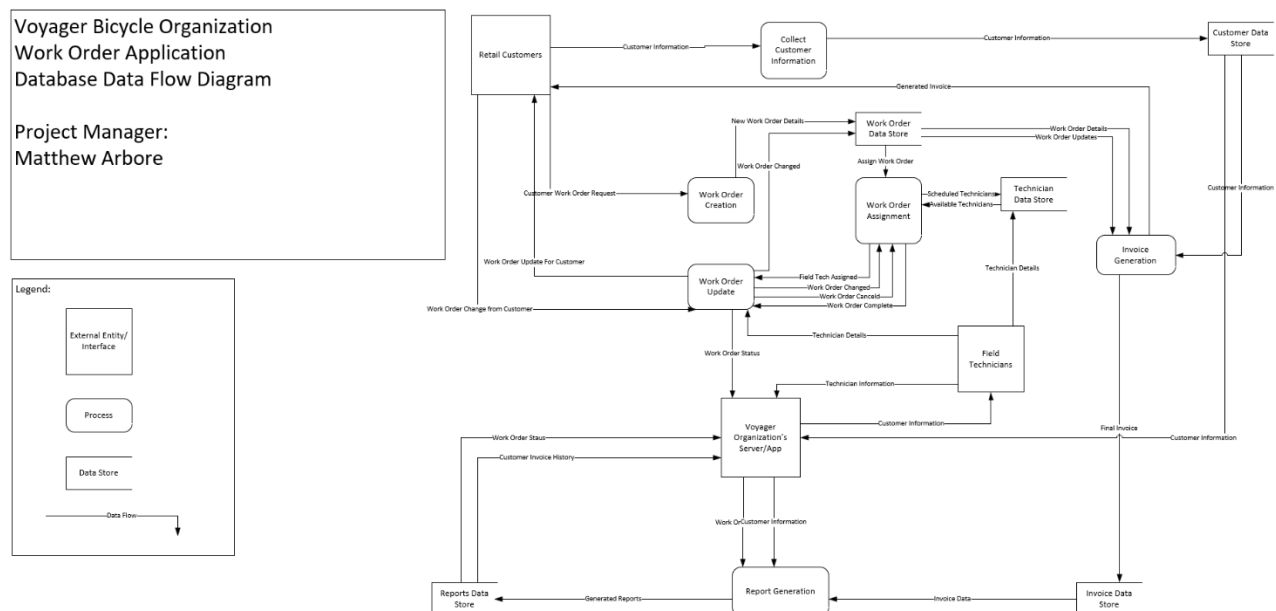
1. Compliance

- 1.1. The system will comply with the data protection laws.

After meeting with the stakeholders, our team began to gather all the requirements necessary to provide the Voyager Bicycle Organization with a database that would meet their needs. This began by reviewing the organizational data received from both the stakeholders and the IT team of Voyager Bicycle Organization. Once this data was reviewed the project team broke down the requirements into the Functional, Nonfunctional, Performance, Security and the Cultural/Political requirements. It was during this process that the processes that needed to be included in the database were defined and gave the team a starting point to begin to define the necessary external entities and data stores.

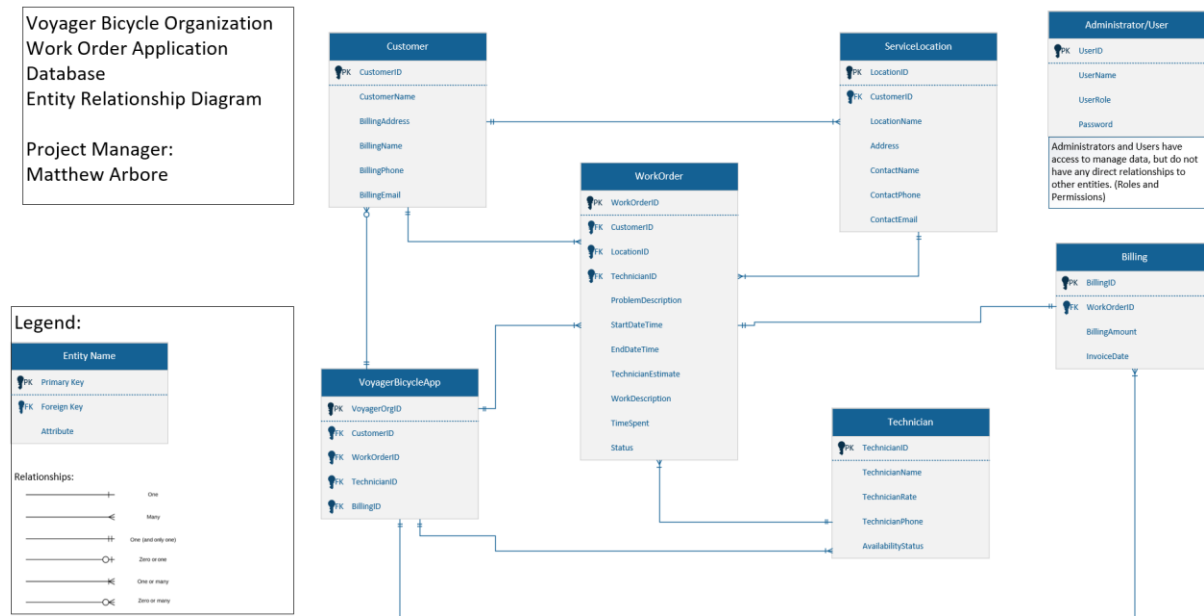
3. Logical Database Design Plan

Database Data Flow Diagram:



Once the requirements for the database were defined, the next step was to define the necessary external entities, processes, and data stores that would be necessary to include in this database. The data flow diagram also allowed the team to begin to define the flow of data between each process, entity and data store. By defining these data flows, our team had a method to determine the necessary relationships that had to exist for this database to function properly. Some processes that were defined in this data flow diagram included how to capture customer data, the work order process, work order update process, work order assignment process, and how to generate invoices. External entities included retail customers, the technicians, and Voyager Bicycle Organizations application. Data stores that were defined during this process include the customer data store, the work order data store, the invoice data store and the reports data store.

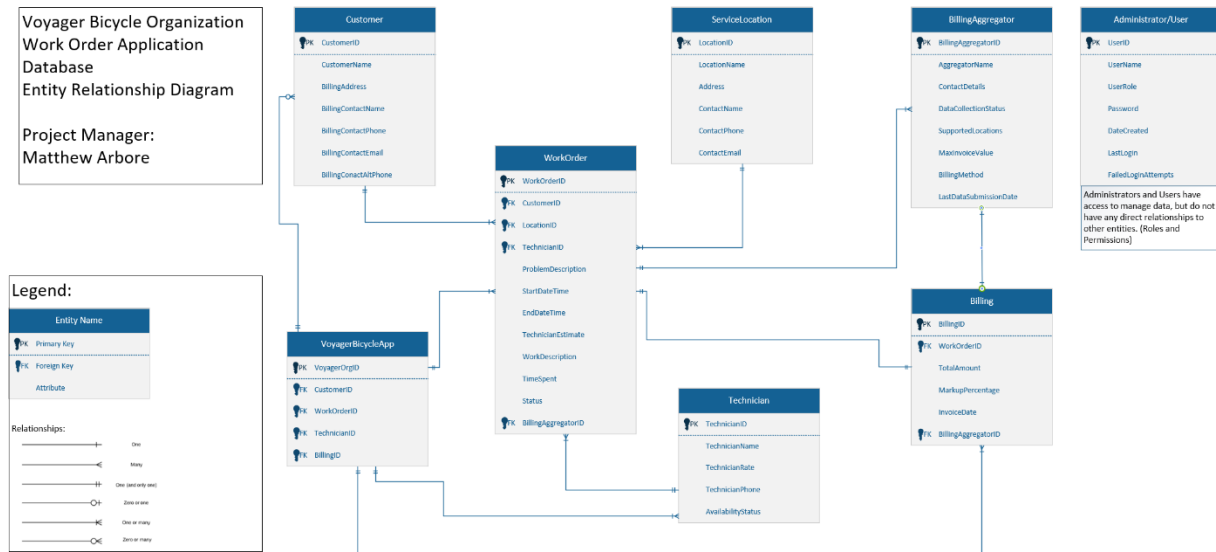
Entity Relationship Diagram:



After completion of the data flow diagram the first entity relationship diagram was created. In this entity relationship diagram the main tables and attributes of the database were defined. The relationships between each table were defined and the attributes of each table were assigned the necessary primary and foreign key relationships. The entity relationship diagram created during this process allowed our team to begin normalizing the database, where we eliminated redundant attributes and defined another table seen in the next entity relationship diagram to further support the functions of the Voyager Bicycle Organization's Work Processing application.

4. Physical Database Design Plan

Entity Relationship Diagram 3rd Normal Form:



After completing the normalization process to ensure the database would be created at the third normal form. Redundant attributes and relationships were eliminated. This included the removal of the customer relationship to the service location table as it was redundant. The work order table would provide that relationship with the customer and the service location. Our team understands that customers have the ability to travel and move, so they could use different service locations around the country. Eliminating that redundant attribute and relationship would also allow this database to scale up more efficiently if Voyager Bicycle Organization opened additional service locations. An additional table, named 'BillingAggreagate', was added during the normalization process to support additional billing and invoice functions for the database. This additional table can capture transaction data necessary for billing reports while allowing the database to remain efficient. The data in the next section

includes all tables, attributes, defined constraints, defined primary and foreign keys as well as the relationships between each table that are represented in this entity relationship diagram.

Defined Database Structure:

1. AdminUser:

Fields

- UserID: Primary Key, Identity
- UserName: Unique Constraint
- UserRole: Check Constraint (Values: 'Technician', 'Admin')
- Password
- DateCreated: Default value is the current system time
- LastLogin
- FailedLoginAttempts: Default value is 0

Constraints

- Primary Key: UserID
- Unique: UserName
- Check: UserRole values restricted to 'Technician' or 'Admin'.

2. Billing:

Fields

- BillingID: Primary Key, Identity
- WorkOrderID: Foreign Key
- TotalAmount
- MarkupPercentage
- InvoiceDate
- BillingAggregatorID: Foreign Key

Constraints

- Primary Key: BillingID
- Foreign Key:
 - Links WorkOrderID to WorkOrder.WorkOrderID
 - Links BillingAggregatorID to BillingAggregator.AggregatorID

3. BillingAggregator:

Fields

- AggregatorID: Primary Key, Identity
- AggregatorName
- ContactDetails
- DataCollectionStatus
- SupportedLocations
- MaxInvoiceValue
- BillingMethod
- LastDataSubmissionDate

Constraints

- Primary Key: AggregatorID

4. Customer:

Fields

- CustomerID: Primary Key, Identity
- CustomerName
- BillingAddress
- BillingContactName
- BillingContactPhone
- BillingContactEmail
- BillingContactAltPhone

Constraints

- Primary Key: CustomerID

5. ServiceLocation:

Fields

- LocationID: Primary Key, Identity
- LocationName
- Address
- ContactName
- ContactPhone
- ContactEmail

Constraints

- Primary Key: LocationID

6. Technician:

Fields

- TechnicianID: Primary Key, Identity
- TechnicianName
- TechnicianRate
- ContactPhone
- AvailabilityStatus

Constraints

- Primary Key: TechnicianID

7. VoyagerBicycleApp:

Fields

- VoyagerOrgID: Primary Key, Identity
- CustomerID: Foreign Key
- WorkOrderID: Foreign Key
- TechnicianID: Foreign Key
- BillingID: Foreign Key

Constraints

- Primary Key: VoyagerOrgID
- Foreign Key Relationships:
 - CustomerID → Customer.CustomerID
 - WorkOrderID → WorkOrder.WorkOrderID
 - TechnicianID → Technician.TechnicianID
 - BillingID → Billing.BillingID

8. WorkOrder:

Fields

- WorkOrderID: Primary Key, Identity
- CustomerID: Foreign Key
- LocationID: Foreign Key
- TechnicianID: Foreign Key
- ProblemDescription
- StartDateTime
- EndDateTime
- TechnicianEstimate
- ActualWorkDescription
- TimeSpent
- Status: Default value is 'Pending'
- BillingAggregatorID: Foreign Key

Constraints

- Primary Key: WorkOrderID
- Foreign Key Relationships:
 - CustomerID → Customer.CustomerID
 - LocationID → ServiceLocation.LocationID
 - TechnicianID → Technician.TechnicianID
 - BillingAggregatorID → BillingAggregator.AggregatorID

Relationships:

- **AdminUser:** Independent table for user management.
- **Billing:**
 - Linked to WorkOrder by WorkOrderID.
 - Linked to BillingAggregator by BillingAggregatorID.
- **BillingAggregator:** Independent table for capturing additional transaction data.
- **Customer:** Independent table for customer details.
- **ServiceLocation:** Independent table for service location details.
- **Technician:** Independent table for technician details.
- **VoyagerBicycleApp:**
 - Acts as a junction table, linking Customer, WorkOrder, Technician, and Billing tables.
- **WorkOrder:**
 - Linked to Customer by CustomerID
 - Linked to ServiceLocation by LocationID
 - Linked to Technician by TechnicianID
 - Linked to BillingAggregator by BillingAggregatorID

After normalizing the data for the database to the third normal form, the project team defined 8 separate tables that contain a total of 56 fields. The constraints defined in this section ensure referential integrity support by only allowing specific data to be entered into each field as well as defining the necessary key relationship to ensure the data integrity. Organizing the data like this allowed for the project execution phase to go smoothly as all necessary components of the database environment have been defined.

SQL Code for Table Structure:

```

WorkOrderTableCre...(TOWER\Matt (74))  WorkOrderPopulate...(TOWER\Matt (66))  SQLQuery2.sql - TO...(TOWER\Matt (54))
-- Use WorkOrderProcessDatabase;

-- Create the Customer table
CREATE TABLE Customer (
    CustomerID INT IDENTITY(1,1) PRIMARY KEY,
    CustomerName VARCHAR(255) NOT NULL,
    BillingAddress VARCHAR(255) NOT NULL,
    BillingContactName VARCHAR(255),
    BillingContactPhone VARCHAR(20),
    BillingContactEmail VARCHAR(255)
);

ALTER TABLE Customer
ADD BillingContactAltPhone VARCHAR(20);

-- Create the ServiceLocation table
CREATE TABLE ServiceLocation (
    LocationID INT IDENTITY(1,1) PRIMARY KEY,
    CustomerID INT NOT NULL,
    LocationName VARCHAR(255) NOT NULL,
    Address VARCHAR(255) NOT NULL,
    ContactName VARCHAR(255),
    ContactPhone VARCHAR(20),
    ContactEmail VARCHAR(255)
);

-- Create the Technician table
CREATE TABLE Technician (
    TechnicianID INT IDENTITY(1,1) PRIMARY KEY,
    TechnicianName VARCHAR(255) NOT NULL,
    TechnicianRate DECIMAL(10, 2) NOT NULL,
    ContactPhone VARCHAR(20),
    AvailabilityStatus VARCHAR(50)
);

-- Create the WorkOrder table
CREATE TABLE WorkOrder (
    WorkOrderID INT IDENTITY(1,1) PRIMARY KEY,
    CustomerID INT NOT NULL,
    LocationID INT NOT NULL,
    TechnicianID INT,
    ProblemDescription TEXT NOT NULL,
    StartDateTime DATETIME2,
    EndDateTime DATETIME2,
    TechnicianEstimate DECIMAL(10, 2),
    ActualWorkDescription TEXT,
    TimeSpent TIME,
    Status VARCHAR(50) DEFAULT 'Pending',
    BillingAggregatorID INT,
    CONSTRAINT FK_WorkOrder_Customer FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID),
    CONSTRAINT FK_WorkOrder_Location FOREIGN KEY (LocationID) REFERENCES ServiceLocation(LocationID),
    CONSTRAINT FK_WorkOrder_Technician FOREIGN KEY (TechnicianID) REFERENCES Technician(TechnicianID)
);

-- Create the Billing table
CREATE TABLE Billing (
    BillingID INT IDENTITY(1,1) PRIMARY KEY,
    WorkOrderID INT NOT NULL,
    TotalAmount DECIMAL(10, 2) NOT NULL,
    MarkupPercentage DECIMAL(5, 2),
    InvoiceDate DATETIME2 NOT NULL,
    BillingAggregatorID INT,
    CONSTRAINT FK_Billing_WorkOrder FOREIGN KEY (WorkOrderID) REFERENCES WorkOrder(WorkOrderID)
);

```

```

WorkOrderTableCre...(TOWER\Matt (74))  X WorkOrderPopulate...(TOWER\Matt (66)) SQLQuery2.sql - TO...(TOWER\Matt (54))
);

-- Create the BillingAggregator table
CREATE TABLE BillingAggregator (
    AggregatorID INT IDENTITY(1,1) PRIMARY KEY,
    AggregatorName VARCHAR(255) NOT NULL,
    ContactDetails TEXT,
    DataCollectionStatus VARCHAR(50)
);

ALTER TABLE BillingAggregator
ADD SupportedLocations TEXT;

ALTER TABLE BillingAggregator
ADD MaxInvoiceValue DECIMAL(10, 2);

ALTER TABLE BillingAggregator
ADD BillingMethod VARCHAR(50);

ALTER TABLE BillingAggregator
ADD LastDataSubmissionDate DATETIME;

-- Add Foreign Key for BillingAggregator in WorkOrder and Billing
ALTER TABLE WorkOrder
ADD CONSTRAINT FK_WorkOrder_BillingAggregator FOREIGN KEY (BillingAggregatorID) REFERENCES BillingAggregator(AggregatorID);

ALTER TABLE Billing
ADD CONSTRAINT FK_Billing_BillingAggregator FOREIGN KEY (BillingAggregatorID) REFERENCES BillingAggregator(AggregatorID);

-- Create the AdminUser table
CREATE TABLE AdminUser (
    UserID INT IDENTITY(1,1) PRIMARY KEY, -- Auto-incrementing primary key
    UserName VARCHAR(255) NOT NULL, -- The username of the admin or user
    UserRole VARCHAR(50) NOT NULL, -- The role (e.g., Admin, User)
    Password VARCHAR(255) NOT NULL -- The password (hashed for security)
);

-- Add the DateCreated column with a default value
ALTER TABLE AdminUser
ADD DateCreated DATETIME2 DEFAULT SYSDATETIME();

-- Add the LastLogin column, allowing NULL values initially
ALTER TABLE AdminUser
ADD LastLogin DATETIME2 NULL;

-- Add FailedLoginAttempts column, default is 0 --
ALTER TABLE AdminUser
ADD FailedLoginAttempts INT DEFAULT 0;

-- Constraint for UserRoles --
ALTER TABLE AdminUser
ADD CONSTRAINT CHK_UserRole CHECK (UserRole IN ('Admin', 'Technician'));

--Create the VoyagerBicycleApp table
CREATE TABLE VoyagerBicycleApp (
    VoyagerOrgID INT IDENTITY(1,1) PRIMARY KEY, -- Auto-incrementing primary key
    CustomerID INT NOT NULL, -- Foreign key referencing Customer table
    WorkOrderID INT NOT NULL, -- Foreign key referencing WorkOrder table
    TechnicianID INT NOT NULL, -- Foreign key referencing Technician table
    BillingID INT NOT NULL, -- Foreign key referencing Billing table

    -- Define foreign key constraints
    CONSTRAINT FK_Voyager_Customer FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID),
    CONSTRAINT FK_Voyager_WorkOrder FOREIGN KEY (WorkOrderID) REFERENCES WorkOrder(WorkOrderID),
    CONSTRAINT FK_Voyager_Technician FOREIGN KEY (TechnicianID) REFERENCES Technician(TechnicianID),
    CONSTRAINT FK_Voyager_Billing FOREIGN KEY (BillingID) REFERENCES Billing(BillingID)
);

```


These images contain all the SQL code and statements that are necessary to build the database environment to provide the necessary functions for the application being developed by the Voyager Bicycle Organization's IT team. This code also defines what data can be entered into each field, ensuring referential integrity is supported. An example of this is: only numbers can be entered into fields such as the Technician's rate as well as the defined format that dates must be entered into each field. Executing this code in a query connected to an empty database will populate the database with all necessary tables, fields, constraints and key relationships. An additional note, as seen in most images from the database there are statements in green text that begin and end with '--'. These are comments made by the project team to explain steps as well as provide additional information if there are any issues during the database migration process or Voyager Bicycle Organization loses this database and must rebuild it. These statements do not affect any SQL statements, code or queries.

SQL Code for Indexing:

```
--Index for Customer Table--
CREATE INDEX IDX_CustomerName ON Customer (CustomerName);
CREATE INDEX IDX_BillingContactPhone ON Customer (BillingContactPhone);

--Index for ServiceLocation Table--
CREATE INDEX IDX_LocationName ON ServiceLocation (LocationName);

--Index for Technician Table--
CREATE INDEX IDX_TechnicianName ON Technician (TechnicianName);
CREATE INDEX IDX_TechnicianAvailability ON Technician (AvailabilityStatus);

--Index for WorkOrder Table--
CREATE INDEX IDX_WorkOrder_CustomerID ON WorkOrder (CustomerID);
CREATE INDEX IDX_WorkOrder_TechnicianID ON WorkOrder (TechnicianID);
CREATE INDEX IDX_WorkOrder_Status ON WorkOrder (Status);
CREATE INDEX IDX_WorkOrder_DateRange ON WorkOrder (StartDateTime, EndDateTime);

--Index for Billing Table--
CREATE INDEX IDX_Billing_WorkOrderID ON Billing (WorkOrderID);
CREATE INDEX IDX_Billing_InvoiceDate ON Billing (InvoiceDate);

--Index for BillingAggregator Table--
CREATE INDEX IDX_BillingAggregator_Name ON BillingAggregator (AggregatorName);

--Index for AdminUser Table--
CREATE INDEX IDX_AdminUser_UserName ON AdminUser (UserName);
CREATE INDEX IDX_AdminUser_UserRole ON AdminUser (UserRole);

--Index for VoyagerBicycleApp Table--
CREATE INDEX IDX_Voyager_CustomerID ON VoyagerBicycleApp (CustomerID);
CREATE INDEX IDX_Voyager_WorkOrderID ON VoyagerBicycleApp (WorkOrderID);
CREATE INDEX IDX_Voyager_TechnicianID ON VoyagerBicycleApp (TechnicianID);
CREATE INDEX IDX_Voyager_BillingID ON VoyagerBicycleApp (BillingID);
```

This image captures all the SQL code and statements that implemented indexing into the database. By indexing all defined relationships, the performance and speed of queries in the database is improved. This step allowed the project team to meet the performance requirements necessary for this database.

SQL Code for Populating Database with Organizational Data:

```

-- Use WorkOrderProcessDatabase;

-- Populate the Admin/User table --
INSERT INTO AdminUser (UserName, UserRole, Password, DateCreated, LastLogin, FailedLoginAttempts)
VALUES
('admin_jane', 'Admin', 'SecurePass123!', GETDATE(), NULL, 0),
('admin_paul', 'Admin', 'StrongPass456!', GETDATE(), NULL, 0),
('admin_emily', 'Admin', 'AdminPass789!', GETDATE(), NULL, 0),
('alice.brown', 'Technician', 'tech_pass1231', GETDATE(), NULL, 0),
('bob.white', 'Technician', 'tech_pass1232', GETDATE(), NULL, 0),
('charlie.green', 'Technician', 'tech_pass1233', GETDATE(), NULL, 0),
('diana.blue', 'Technician', 'tech_pass1234', GETDATE(), NULL, 0),
('edward.black', 'Technician', 'tech_pass1235', GETDATE(), NULL, 0),
('george.miller', 'Technician', 'tech_pass1236', GETDATE(), NULL, 0),
('helen.carter', 'Technician', 'tech_pass1237', GETDATE(), NULL, 0),
('ian.thompson', 'Technician', 'tech_pass1238', GETDATE(), NULL, 0),
('julia.roberts', 'Technician', 'tech_pass1239', GETDATE(), NULL, 0),
('kyle.evans', 'Technician', 'tech_pass12311', GETDATE(), NULL, 0),
('laura.hill', 'Technician', 'tech_pass12312', GETDATE(), NULL, 0),
('mark.patterson', 'Technician', 'tech_pass12313', GETDATE(), NULL, 0),
('nina.brown', 'Technician', 'tech_pass12314', GETDATE(), NULL, 0),
('oscar.reed', 'Technician', 'tech_pass12315', GETDATE(), NULL, 0),
('paul.adams', 'Technician', 'tech_pass12316', GETDATE(), NULL, 0),
('quinn.davis', 'Technician', 'tech_pass12318', GETDATE(), NULL, 0),
('rachel.clark', 'Technician', 'tech_pass12319', GETDATE(), NULL, 0),
('steve.wilson', 'Technician', 'tech_pass12321', GETDATE(), NULL, 0),
('tina.harris', 'Technician', 'tech_pass12322', GETDATE(), NULL, 0),
('victor.white', 'Technician', 'tech_pass12323', GETDATE(), NULL, 0);

-- Populate the ServiceLocation table -- 5 --Locations
INSERT INTO ServiceLocation (LocationName, Address, ContactName, ContactPhone, ContactEmail)
VALUES
('Voyager Bicycle Downtown', '123 Elm St, Springfield, IL', 'Mark Trail', '555-0200', 'mark.trail@example.com'),
('Voyager Bicycle West', '456 Oak St, Denver, CO', 'Lisa Ray', '555-0201', 'lisa.ray@example.com'),
('Voyager Bicycle North', '789 Pine St, , Detroit, MI', 'James Bond', '555-0202', 'james.bond@example.com'),
('Voyager Bicycle East', '321 Birch St, Boston, MA', 'Laura Croft', '555-0203', 'laura.croft@example.com'),
('Voyager Bicycle South', '654 Maple St, Nashville, TN', 'Ethan Hunt', '555-0204', 'ethan.hunt@example.com');

```

```
-- Populate the Customer table -- 30 Customers --
INSERT INTO Customer (CustomerName, BillingAddress, BillingContactName, BillingContactPhone, BillingContactEmail, BillingContactAltPhone)
VALUES
('Adventure Gear', '123 Elm St, Springfield, IL', 'John Doe', '555-0100', 'john.doe@example.com', '555-0101'),
('Bike World', '456 Oak St, Denver, CO', 'Jane Smith', '555-0102', 'jane.smith@example.com', '555-0103'),
('Pedal Pros', '789 Pine St, Detroit, MI', 'Mike Johnson', '555-0104', 'mike.johnson@example.com', '555-0105'),
('Cycle Depot', '321 Birch St, Boston, MA', 'Sarah Connor', '555-0106', 'sarah.connor@example.com', '555-0107'),
('Gear Up', '654 Maple St, Nashville, TN', 'Tom Hardy', '555-0108', 'tom.hardy@example.com', '555-0109'),
('Cycle Haven', '101 Cedar Ave, Springfield, IL', 'Emma Watson', '555-1000', 'emma.watson@example.com', '555-1001'),
('Pedal Pushers', '202 Walnut St, Denver, CO', 'Liam Neeson', '555-1002', 'liam.neeson@example.com', '555-1003'),
('Wheel Works', '303 Maple Ave, Detroit, MI', 'Sophia Turner', '555-1004', 'sophia.turner@example.com', '555-1005'),
('Gear Junction', '404 Pine Blvd, Boston, MA', 'James Cameron', '555-1006', 'james.cameron@example.com', '555-1007'),
('Ride Ready', '505 Birch Ln, Nashville, TN', 'Mia Khalil', '555-1008', 'mia.khalil@example.com', '555-1009'),
('Spin Masters', '606 Elm St, Springfield, IL', 'Noah Bennett', '555-1010', 'noah.bennett@example.com', '555-1011'),
('Cyclist Pro', '707 Oak Dr, Denver, CO', 'Olivia Harper', '555-1012', 'olivia.harper@example.com', '555-1013'),
('Trail Blazers', '808 Chestnut Way, Detroit, MI', 'William Lee', '555-1014', 'william.lee@example.com', '555-1015'),
('The Chain Gang', '909 Aspen Cir, Boston, MA', 'Isabella Scott', '555-1016', 'isabella.scott@example.com', '555-1017'),
('Trek Squad', '1001 Redwood Ct, Nashville, TN', 'Mason Carter', '555-1018', 'mason.carter@example.com', '555-1019'),
('Mountain Movers', '1101 Cypress Rd, Springfield, IL', 'Ava Brooks', '555-1020', 'ava.brooks@example.com', '555-1021'),
('Path Finders', '1201 Fir Ln, Denver, CO', 'Ethan James', '555-1022', 'ethan.james@example.com', '555-1023'),
('Cycle Crafters', '1301 Spruce Ave, Detroit, MI', 'Charlotte Parker', '555-1024', 'charlotte.parker@example.com', '555-1025'),
('Two Wheelers', '1401 Hickory St, Boston, MA', 'Lucas Adams', '555-1026', 'lucas.adams@example.com', '555-1027'),
('Bicycle Enthusiasts', '1501 Poplar Blvd, Nashville, TN', 'Ella Turner', '555-1028', 'ella.turner@example.com', '555-1029'),
('Trail Blazers', '500 Forest Rd, Springfield, IL', 'Nancy Walker', '303-555-2100', 'nancy.walker@example.com', '303-555-2101'),
('Speedy Wheels', '760 Lake St, Denver, CO', 'Jake Turner', '407-555-3200', 'jake.turner@example.com', '407-555-3201'),
('Cycle Haven', '123 Sunshine Blvd, Detroit, MI', 'Emma Stone', '813-555-1100', 'emma.stone@example.com', '813-555-1101'),
('Riding Pro', '890 Mountain View Ave, Boston, MA', 'Ryan Brooks', '720-555-2200', 'ryan.brooks@example.com', '720-555-2201'),
('Pedal Nation', '15 Ocean Dr, Nashville, TN', 'Sophia Lewis', '619-555-4400', 'sophia.lewis@example.com', '619-555-4401'),
('Urban Cyclists', '300 Main St, Springfield, IL', 'Lucas Hill', '512-555-7700', 'lucas.hill@example.com', '512-555-7701'),
('EcoRides', '901 Greenway Dr, Denver, CO', 'Olivia Bennett', '503-555-6600', 'olivia.bennett@example.com', '503-555-6601'),
('Wheel Works', '88 Elmwood St, Detroit, MI', 'James Carter', '617-555-5500', 'james.carter@example.com', '617-555-5501'),
('Ride On', '420 Freedom Blvd, Boston, MA', 'Chloe Fisher', '615-555-3300', 'chloe.fisher@example.com', '615-555-3301'),
('Peak Performance', '101 Summit Rd, Nashville, TN', 'Ethan Adams', '801-555-4400', 'ethan.adams@example.com', '801-555-4401'),
('Bike Nation', '1500 W Main St, Springfield, IL', 'Ella Stevens', '555-1030', 'ella.stevens@example.com', '555-1031'),
('Pedal Power', '1600 E 16th St, Denver, CO', 'Lucas Moore', '555-1032', 'lucas.moore@example.com', '555-1033'),
('Cycle Works', '1700 W Grand Blvd, Detroit, MI', 'Sophia Lee', '555-1034', 'sophia.lee@example.com', '555-1035'),
('Ride Revolution', '1800 Chestnut Ave, Boston, MA', 'Isabella Taylor', '555-1036', 'isabella.taylor@example.com', '555-1037'),
('Velo City', '1900 Park Ave, Nashville, TN', 'Mason Davis', '555-1038', 'mason.davis@example.com', '555-1039'),
('The Bike Shop', '2000 Oakwood St, Springfield, IL', 'Ryan Lewis', '555-1040', 'ryan.lewis@example.com', '555-1041'),
('Wheels in Motion', '2100 Sunset Blvd, Denver, CO', 'Olivia Clark', '555-1042', 'olivia.clark@example.com', '555-1043'),
('Speedy Spokes', '2200 Jefferson Ave, Detroit, MI', 'Charlotte White', '555-1044', 'charlotte.white@example.com', '555-1045'),
('Two Wheel Trend', '2300 River Rd, Boston, MA', 'Lucas Mitchell', '555-1046', 'lucas.mitchell@example.com', '555-1047'),
('Trail Blazers Co.', '2400 High St, Nashville, TN', 'James Harris', '555-1048', 'james.harris@example.com', '555-1049');
```

```
-- Populate the Technician table --
INSERT INTO Technician (TechnicianName, TechnicianRate, ContactPhone, AvailabilityStatus)
VALUES
('Alice Brown', 50.00, '555-0300', 'Available'),
('Bob White', 45.00, '555-0301', 'Busy'),
('Charlie Green', 48.50, '555-0302', 'Available'),
('Diana Blue', 52.00, '555-0303', 'Busy'),
('Edward Black', 47.50, '555-0304', 'Available'),
('George Miller', 46.75, '555-0305', 'Available'),
('Helen Carter', 50.25, '555-0306', 'Busy'),
('Ian Thompson', 47.00, '555-0307', 'Available'),
('Julia Roberts', 48.50, '555-0308', 'Busy'),
('Kyle Evans', 49.75, '555-0309', 'Available'),
('Laura Hill', 51.00, '555-0310', 'Available'),
('Mark Patterson', 46.50, '555-0311', 'Busy'),
('Nina Brown', 47.25, '555-0312', 'Available'),
('Oscar Reed', 48.00, '555-0313', 'Available'),
('Paul Adams', 49.00, '555-0314', 'Busy'),
('Quinn Davis', 50.00, '555-0315', 'Available'),
('Rachel Clark', 51.50, '555-0316', 'Busy'),
('Steve Wilson', 52.00, '555-0317', 'Available'),
('Tina Harris', 46.00, '555-0318', 'Busy'),
('Victor White', 47.50, '555-0319', 'Available');

-- Populate the WorkOrder table --
INSERT INTO WorkOrder (CustomerID, LocationID, TechnicianID, ProblemDescription, StartDateTime, EndDateTime, TechnicianEstimate, ActualWorkDescription, TimeSpent, Status, BillingAggregatorID)
VALUES
(1, 1, 1, 'Flat tire repair', '2024-12-01 09:00:00', '2024-12-01 10:00:00', 50.00, 'Replaced tube and inflated tire', '01:00:00', 'Complete', NULL),
(2, 2, 2, 'Brake adjustment', '2024-12-02 11:00:00', '2024-12-02 12:00:00', 45.00, 'Adjusted brake pads and cables', '01:00:00', 'Complete', NULL),
(3, 3, 3, 'Gear tuning', '2024-12-03 14:00:00', NULL, 48.50, NULL, NULL, 'Pending', NULL),
(4, 4, 4, 'Chain replacement', '2024-12-04 10:30:00', NULL, 52.00, NULL, NULL, 'In Progress', NULL),
(5, 5, 5, 'Wheel alignment', '2024-12-05 15:00:00', '2024-12-05 16:30:00', 47.50, 'Aligned wheels and tested', '01:30:00', 'Complete', NULL),
(6, 1, 6, 'Replace broken spokes', '2024-12-06 09:00:00', '2024-12-06 10:30:00', 70.00, 'Replaced spokes and trued the wheel', '01:30:00', 'Complete', NULL),
(7, 2, 7, 'Fix derailleur issues', '2024-12-07 11:00:00', NULL, 65.00, NULL, NULL, 'Pending', NULL),
(8, 3, 8, 'Adjust seat and handlebars', '2024-12-08 13:00:00', '2024-12-08 13:45:00', 55.00, 'Adjusted seat height and handlebar angle', '00:45:00', 'Complete', NULL),
(9, 4, 9, 'Replace brake cables', '2024-12-09 15:00:00', NULL, 75.00, NULL, NULL, 'In Progress', NULL),
(10, 5, 10, 'Install new chain', '2024-12-10 08:30:00', '2024-12-10 09:30:00', 60.00, 'Installed and tested new chain', '01:00:00', 'Complete', NULL),
(11, 1, 11, 'Fix flat tire', '2024-12-11 10:00:00', '2024-12-11 10:30:00', 50.00, 'Replaced inner tube and inflated tire', '00:30:00', 'Complete', NULL),
(12, 2, 12, 'Lubricate and clean drivetrain', '2024-12-12 14:00:00', NULL, 45.00, NULL, NULL, 'Pending', NULL),
(13, 3, 13, 'Replace handlebar grips', '2024-12-13 09:00:00', '2024-12-13 09:45:00', 40.00, 'Replaced grips with new ergonomic ones', '00:45:00', 'Complete', NULL),
(14, 4, 14, 'Check wheel alignment', '2024-12-14 12:30:00', NULL, 50.00, NULL, NULL, 'In Progress', NULL),
(15, 5, 15, 'Install new pedals', '2024-12-15 10:00:00', '2024-12-15 11:00:00', 65.00, 'Installed lightweight pedals', '01:00:00', 'Complete', NULL),
(16, 1, 16, 'Brake pad replacement', '2024-12-16 11:00:00', '2024-12-16 12:15:00', 75.00, 'Replaced brake pads and tested', '01:15:00', 'Complete', NULL),
(17, 2, 17, 'Inspect and clean frame', '2024-12-17 14:30:00', NULL, 50.00, NULL, NULL, 'Pending', NULL),
(18, 3, 18, 'Gear adjustment', '2024-12-18 09:30:00', '2024-12-18 10:00:00', 60.00, 'Adjusted gear shift mechanisms', '00:30:00', 'Complete', NULL),
(19, 4, 19, 'Fix wobbling wheel', '2024-12-19 13:00:00', NULL, 80.00, NULL, NULL, 'In Progress', NULL),
(20, 5, 20, 'Replace bottom bracket', '2024-12-20 08:00:00', '2024-12-20 10:00:00', 90.00, 'Replaced bottom bracket and tested', '02:00:00', 'Complete', NULL);
```

```

-- Populate the Billing table --
INSERT INTO Billing (WorkOrderID, TotalAmount, MarkupPercentage, InvoiceDate, BillingAggregatorID)
VALUES
(1, 55.00, 10.00, '2024-12-01 17:00:00', 1), -- Completed
(2, 49.50, 10.00, '2024-12-02 18:00:00', 1), -- Completed
(5, 52.25, 10.00, '2024-12-05 19:00:00', 2), -- Completed
(8, 60.50, 10.00, '2024-12-08 14:00:00', 3), -- Completed
(10, 66.00, 10.00, '2024-12-10 12:00:00', 2), -- Completed
(11, 55.00, 10.00, '2024-12-11 11:00:00', 1), -- Completed
(13, 44.00, 10.00, '2024-12-13 11:00:00', 4), -- Completed
(15, 71.50, 10.00, '2024-12-15 13:00:00', 1), -- Completed
(16, 82.50, 10.00, '2024-12-16 13:00:00', 2), -- Completed
(18, 66.00, 10.00, '2024-12-18 12:00:00', 3), -- Completed
(20, 99.00, 10.00, '2024-12-20 14:00:00', 4); -- Completed

-- Pending and in-progress work orders with no invoice date or total
INSERT INTO Billing (WorkOrderID, TotalAmount, MarkupPercentage, InvoiceDate, BillingAggregatorID)
VALUES
(3, 0, 10.00, CURRENT_TIMESTAMP, NULL), -- Pending
(4, 0, 10.00, CURRENT_TIMESTAMP, NULL), -- In Progress
(7, 0, 10.00, CURRENT_TIMESTAMP, NULL), -- Pending
(9, 0, 10.00, CURRENT_TIMESTAMP, NULL), -- In Progress
(14, 0, 10.00, CURRENT_TIMESTAMP, NULL); -- In Progress

-- Populate the BillingAggregator table --
INSERT INTO BillingAggregator (AggregatorName, ContactDetails, DataCollectionStatus, SupportedLocations, MaxInvoiceValue, BillingMethod, LastDataSubmissionDate)
VALUES
('CycleBilling LLC', '123 Aggregator Lane, Springfield, IL', 'Active', 'Springfield, IL; Detroit, MI; Denver, CO', 1000.00, 'ACH', '2024-12-12 10:00:00'),
('Velocity Accounts', '456 Aggregator Drive, Springfield, IL', 'Active', 'Boston, MA; Nashville, TN', 2000.00, 'Credit Card', '2024-12-10 15:30:00'),
('WheelWorks Financials', '789 Aggregator Street, Springfield, IL', 'Inactive', 'Denver, CO; Springfield, IL', 1500.00, 'ACH', '2024-11-25 14:00:00'),
('PedalPay Solutions', '321 Aggregator Court, Springfield, IL', 'Active', 'Nashville, TN; Boston, MA; Detroit, MI', 500.00, 'Bank Transfer', '2024-12-13 09:00:00'),
('Velocity Tech Solutions', '1020 Tech Avenue, San Francisco, CA', 'Active', 'New York, NY; Los Angeles, CA', 2500.00, 'Credit Card', '2024-12-12 16:45:00'),
('GreenTech Services', '5678 Green Drive, Seattle, WA', 'Inactive', 'Portland, OR; Boise, ID', 3000.00, 'ACH', '2024-11-15 11:30:00'),
('Swift Payments', '1340 Quick Street, Denver, CO', 'Active', 'Austin, TX; Denver, CO; Miami, FL', 800.00, 'Bank Transfer', '2024-12-14 13:00:00'),
('Prime Solutions', '883 Enterprise Way, Chicago, IL', 'Inactive', 'Chicago, IL; Atlanta, GA', 1200.00, 'Credit Card', '2024-12-02 10:00:00'),
('FusionPay Systems', '452 Innovation Drive, Phoenix, AZ', 'Active', 'San Diego, CA; Phoenix, AZ', 1800.00, 'ACH', '2024-12-05 14:30:00'),
('QuickPay Enterprises', '787 Fast Lane, Boston, MA', 'Active', 'Boston, MA; New York, NY', 400.00, 'Bank Transfer', '2024-12-08 17:00:00');

-- Populate the VoyagerBicycleApp table --
INSERT INTO VoyagerBicycleApp (CustomerID, WorkOrderID, TechnicianID, BillingID)
VALUES
(1, 1, 1, 1),
(2, 2, 2, 2),
(3, 3, 3, 3),
(4, 4, 4, 5),
(5, 5, 5, 5),
(6, 6, 6, 6),
(7, 7, 7, 7),
(8, 8, 8, 8),
(9, 9, 9, 8),
(10, 10, 10, 10);

```

The images contained in this section show all the organizational data that was implemented into the database. This contains all the information including current customers, service locations, technicians, work orders, billing information, as well as the data necessary for the Voyager Bicycle Application table. If these SQL statements and code are executed in a database that has been populated with table structure data and indexing information from the previous sections, then the database will be populated with all current organizational data.

5. Sample Data

Admin/User Table:

	UserID	UserName	UserRole	Password	DateCreated	LastLogin	FailedLoginAttempts
1	1	admin_jane	Admin	SecurePass123!	2024-12-17 08:44:11.8133333	NULL	0
2	2	admin_paul	Admin	StrongPass456!	2024-12-17 08:44:11.8133333	NULL	0
3	3	admin_emily	Admin	AdminPass789!	2024-12-17 08:44:11.8133333	NULL	0
4	4	alice.brown	Technician	tech_pass1231	2024-12-17 08:44:11.8133333	NULL	0
5	5	bob.white	Technician	tech_pass1232	2024-12-17 08:44:11.8133333	NULL	0
6	6	charlie.green	Technician	tech_pass1233	2024-12-17 08:44:11.8133333	NULL	0
7	7	diana.blue	Technician	tech_pass1234	2024-12-17 08:44:11.8133333	NULL	0
8	8	edward.black	Technician	tech_pass1235	2024-12-17 08:44:11.8133333	NULL	0
9	9	george.miller	Technician	tech_pass1236	2024-12-17 08:44:11.8133333	NULL	0
10	10	helen.carter	Technician	tech_pass1237	2024-12-17 08:44:11.8133333	NULL	0
11	11	ian.thompson	Technician	tech_pass1238	2024-12-17 08:44:11.8133333	NULL	0
12	12	julia.roberts	Technician	tech_pass1239	2024-12-17 08:44:11.8133333	NULL	0
13	13	kyle.evans	Technician	tech_pass12311	2024-12-17 08:44:11.8133333	NULL	0
14	14	laura.hill	Technician	tech_pass12312	2024-12-17 08:44:11.8133333	NULL	0
15	15	mark.patterson	Technician	tech_pass12313	2024-12-17 08:44:11.8133333	NULL	0
16	16	nina.brown	Technician	tech_pass12314	2024-12-17 08:44:11.8133333	NULL	0
17	17	oscar.reed	Technician	tech_pass12315	2024-12-17 08:44:11.8133333	NULL	0
18	18	paul.adams	Technician	tech_pass12316	2024-12-17 08:44:11.8133333	NULL	0
19	19	quinn.davis	Technician	tech_pass12318	2024-12-17 08:44:11.8133333	NULL	0
20	20	rachel.clark	Technician	tech_pass12319	2024-12-17 08:44:11.8133333	NULL	0
21	21	steve.wilson	Technician	tech_pass12321	2024-12-17 08:44:11.8133333	NULL	0
22	22	tina.hamis	Technician	tech_pass12322	2024-12-17 08:44:11.8133333	NULL	0
23	23	victor.white	Technician	tech_pass12323	2024-12-17 08:44:11.8133333	NULL	0

Customer Table:

	CustomerID	CustomerName	BillingAddress	BillingContactName	BillingContactPhone	BillingContactEmail	BillingContactAltPhone
1	1	Adventure Gear	123 Elm St, Springfield, IL	John Doe	555-0100	john.doe@example.com	555-0101
2	2	Bike World	456 Oak St, Denver, CO	Jane Smith	555-0102	jane.smith@example.com	555-0103
3	3	Pedal Pros	789 Pine St, Detroit, MI	Mike Johnson	555-0104	mike.johnson@example.com	555-0105
4	4	Cycle Depot	321 Birch St, Boston, MA	Sarah Connor	555-0106	sarah.connor@example.com	555-0107
5	5	Gear Up	654 Maple St, Nashville, TN	Tom Hardy	555-0108	tom.hardy@example.com	555-0109
6	6	Cycle Haven	101 Cedar Ave, Springfield, IL	Emma Watson	555-1000	emma.watson@example.com	555-1001
7	7	Pedal Pushers	202 Walnut St, Denver, CO	Liam Neeson	555-1002	liam.neeson@example.com	555-1003
8	8	Wheel Works	303 Maple Ave, Detroit, MI	Sophia Turner	555-1004	sophia.turner@example.com	555-1005
9	9	Gear Junction	404 Pine Blvd, Boston, MA	James Cameron	555-1006	james.cameron@example.com	555-1007
10	10	Ride Ready	505 Birch Ln, Nashville, TN	Mia Khalil	555-1008	mia.khalil@example.com	555-1009
11	11	Spin Masters	606 Elm St, Springfield, IL	Noah Bennett	555-1010	noah.bennett@example.com	555-1011
12	12	Cyclist Pro	707 Oak Dr, Denver, CO	Olivia Harper	555-1012	olivia.harper@example.com	555-1013
13	13	Trail Blazers	808 Chestnut Way, Detroit, MI	William Lee	555-1014	william.lee@example.com	555-1015
14	14	The Chain Gang	909 Aspen Cir, Boston, MA	Isabella Scott	555-1016	isabella.scott@example.com	555-1017
15	15	Trek Squad	1001 Redwood Ct, Nashville, TN	Mason Carter	555-1018	mason.carter@example.com	555-1019
16	16	Mountain Movers	1101 Cypress Rd, Springfield, IL	Ava Brooks	555-1020	ava.brooks@example.com	555-1021
17	17	Path Finders	1201 Fir Ln, Denver, CO	Ethan James	555-1022	ethan.james@example.com	555-1023
18	18	Cycle Crafters	1301 Spruce Ave, Detroit, MI	Charlotte Parker	555-1024	charlotte.parker@example.com	555-1025
19	19	Two Wheelers	1401 Hickory St, Boston, MA	Lucas Adams	555-1026	lucas.adams@example.com	555-1027
20	20	Bicycle Enthusiasts	1501 Poplar Blvd, Nashville, TN	Ella Turner	555-1028	ella.turner@example.com	555-1029
21	21	Trail Blazers	500 Forest Rd, Springfield, IL	Nancy Walker	303-555-2100	nancy.walker@example.com	303-555-2101
22	22	Speedy Wheels	760 Lake St, Denver, CO	Jake Turner	407-555-3200	jake.turner@example.com	407-555-3201
23	23	Cycle Haven	123 Sunshine Blvd, Detroit, MI	Emma Stone	813-555-1100	emma.stone@example.com	813-555-1101
24	24	Riding Pro	890 Mountain View Ave, Boston, MA	Ryan Brooks	720-555-2200	ryan.brooks@example.com	720-555-2201
25	25	Pedal Nation	15 Ocean Dr, Nashville, TN	Sophia Lewis	619-555-4400	sophia.lewis@example.com	619-555-4401
26	26	Urban Cyclists	300 Main St, Springfield, IL	Lucas Hill	512-555-7700	lucas.hill@example.com	512-555-7701
27	27	EcoRides	901 Greenway Dr, Denver, CO	Olivia Bennett	503-555-6600	olivia.bennett@example.com	503-555-6601
28	28	Wheel Works	88 Elmwood St, Detroit, MI	James Carter	617-555-5500	james.carter@example.com	617-555-5501
29	29	Ride On	420 Freedom Blvd, Boston, MA	Chloe Fisher	615-555-3300	chloe.fisher@example.com	615-555-3301
30	30	Peak Performance	101 Summit Rd, Nashville, TN	Ethan Adams	801-555-4400	ethan.adams@example.com	801-555-4401

Service Location Table:

	LocationID	LocationName	Address	ContactName	ContactPhone	ContactEmail
1	1	Voyager Bicycle Downtown	123 Elm St, Springfield, IL	Mark Trail	555-0200	mark.trail@example.com
2	2	Voyager Bicycle West	456 Oak St, Denver, CO	Lisa Ray	555-0201	lisa.ray@example.com
3	3	Voyager Bicycle North	789 Pine St, , Detroit, MI	James Bond	555-0202	james.bond@example.com
4	4	Voyager Bicycle East	321 Birch St, Boston, MA	Laura Croft	555-0203	laura.croft@example.com
5	5	Voyager Bicycle South	654 Maple St, Nashville, TN	Ethan Hunt	555-0204	ethan.hunt@example.com

Technician Table:

	TechnicianID	TechnicianName	TechnicianRate	ContactPhone	AvailabilityStatus
1	1	Alice Brown	50.00	555-0300	Available
2	2	Bob White	45.00	555-0301	Busy
3	3	Charlie Green	48.50	555-0302	Available
4	4	Diana Blue	52.00	555-0303	Busy
5	5	Edward Black	47.50	555-0304	Available
6	6	George Miller	46.75	555-0305	Available
7	7	Helen Carter	50.25	555-0306	Busy
8	8	Ian Thompson	47.00	555-0307	Available
9	9	Julia Roberts	48.50	555-0308	Busy
10	10	Kyle Evans	49.75	555-0309	Available
11	11	Laura Hill	51.00	555-0310	Available
12	12	Mark Patterson	46.50	555-0311	Busy
13	13	Nina Brown	47.25	555-0312	Available
14	14	Oscar Reed	48.00	555-0313	Available
15	15	Paul Adams	49.00	555-0314	Busy
16	16	Quinn Davis	50.00	555-0315	Available
17	17	Rachel Clark	51.50	555-0316	Busy
18	18	Steve Wilson	52.00	555-0317	Available
19	19	Tina Harris	46.00	555-0318	Busy
20	20	Victor White	47.50	555-0319	Available

Work Order Table:

	WorkOrderID	CustomerID	LocationID	TechnicianID	ProblemDescription	StartDateTime	EndDateTime	TechnicianEstimate	ActualWorkDescription	TimeSpent	Status
1	1	1	1	1	Flat tire repair	2024-12-01 09:00:00.0000000	2024-12-01 10:00:00.0000000	50.00	Replaced tube and inflated tire	01:00:00.0000000	Complete
2	2	2	2	2	Brake adjustment	2024-12-02 11:00:00.0000000	2024-12-02 12:00:00.0000000	45.00	Adjusted brake pads and cables	01:00:00.0000000	Complete
3	3	3	3	3	Gear tuning	2024-12-03 14:00:00.0000000	NULL	48.50	NULL	NULL	Pending
4	4	4	4	4	Chain replacement	2024-12-04 10:30:00.0000000	NULL	52.00	NULL	NULL	In Progress
5	5	5	5	5	Wheel alignment	2024-12-05 15:00:00.0000000	2024-12-05 16:30:00.0000000	47.50	Aligned wheels and tested	01:30:00.0000000	Complete
6	6	6	1	6	Replace broken spokes	2024-12-06 09:00:00.0000000	2024-12-06 10:30:00.0000000	70.00	Replaced spokes and trued the wheel	01:30:00.0000000	Complete
7	7	7	2	7	Fix derailleur issues	2024-12-07 11:00:00.0000000	NULL	65.00	NULL	NULL	Pending
8	8	8	3	8	Adjust seat and handlebars	2024-12-08 13:00:00.0000000	2024-12-08 13:45:00.0000000	55.00	Adjusted seat height and handlebar angle	00:45:00.0000000	Complete
9	9	9	4	9	Replace brake cables	2024-12-09 15:00:00.0000000	NULL	75.00	NULL	NULL	In Progress
10	10	10	5	10	Install new chain	2024-12-10 08:30:00.0000000	2024-12-10 09:30:00.0000000	60.00	Installed and tested new chain	01:00:00.0000000	Complete
11	11	11	1	11	Fix flat tire	2024-12-11 10:00:00.0000000	2024-12-11 10:30:00.0000000	50.00	Replaced inner tube and inflated tire	00:30:00.0000000	Complete
12	12	12	2	12	Lubricate and clean drivetrain	2024-12-12 14:00:00.0000000	NULL	45.00	NULL	NULL	Pending
13	13	13	3	13	Replace handlebar grips	2024-12-13 09:00:00.0000000	2024-12-13 09:45:00.0000000	40.00	Replaced grips with new ergonomic ones	00:45:00.0000000	Complete
14	14	14	4	14	Check wheel alignment	2024-12-14 12:30:00.0000000	NULL	50.00	NULL	NULL	In Progress
15	15	15	5	15	Install new pedals	2024-12-15 10:00:00.0000000	2024-12-15 11:00:00.0000000	65.00	Installed lightweight pedals	01:00:00.0000000	Complete
16	16	16	1	16	Brake pad replacement	2024-12-16 11:00:00.0000000	2024-12-16 12:15:00.0000000	75.00	Replaced brake pads and tested	01:15:00.0000000	Complete
17	17	17	2	17	Inspect and clean frame	2024-12-17 14:30:00.0000000	NULL	50.00	NULL	NULL	Pending
18	18	18	3	18	Gear adjustment	2024-12-18 09:30:00.0000000	2024-12-18 10:00:00.0000000	60.00	Adjusted gear shift mechanisms	00:30:00.0000000	Complete
19	19	19	4	19	Fix wobbling wheel	2024-12-19 13:00:00.0000000	NULL	80.00	NULL	NULL	In Progress
20	20	20	5	20	Replace bottom bracket	2024-12-20 08:00:00.0000000	2024-12-20 10:00:00.0000000	90.00	Replaced bottom bracket and tested	02:00:00.0000000	Complete

Billing Table:

	BillingID	WorkOrderID	TotalAmount	MarkupPercentage	InvoiceDate	BillingAggregatorID
1	1	1	55.00	10.00	2024-12-01 17:00:00.00000000	1
2	2	2	49.50	10.00	2024-12-02 18:00:00.00000000	1
3	3	5	52.25	10.00	2024-12-05 19:00:00.00000000	2
4	4	8	60.50	10.00	2024-12-08 14:00:00.00000000	3
5	5	10	66.00	10.00	2024-12-10 12:00:00.00000000	2
6	6	11	55.00	10.00	2024-12-11 11:00:00.00000000	1
7	7	13	44.00	10.00	2024-12-13 11:00:00.00000000	4
8	8	15	71.50	10.00	2024-12-15 13:00:00.00000000	1
9	9	16	82.50	10.00	2024-12-16 13:00:00.00000000	2
10	10	18	66.00	10.00	2024-12-18 12:00:00.00000000	3
11	11	20	99.00	10.00	2024-12-20 14:00:00.00000000	4
12	12	3	0.00	10.00	2024-12-17 08:46:41.00333333	NULL
13	13	4	0.00	10.00	2024-12-17 08:46:41.00333333	NULL
14	14	7	0.00	10.00	2024-12-17 08:46:41.00333333	NULL
15	15	9	0.00	10.00	2024-12-17 08:46:41.00333333	NULL
16	16	14	0.00	10.00	2024-12-17 08:46:41.00333333	NULL

Billing Aggregate Table:

	AggregatorID	AggregatorName	ContactDetails	DataCollectionStatus	SupportedLocations	MaxInvoiceValue	BillingMethod	LastDataSubmissionDate
1	1	CycleBilling LLC	123 Aggregator Lane, Springfield, IL	Active	Springfield, IL; Detroit, MI; Denver, CO	1000.00	ACH	2024-12-12 10:00:00.000
2	2	Velocity Accounts	456 Aggregator Drive, Springfield, IL	Active	Boston, MA; Nashville, TN	2000.00	Credit Card	2024-12-10 15:30:00.000
3	3	WheelWorks Financials	789 Aggregator Street, Springfield, IL	Inactive	Denver, CO; Springfield, IL	1500.00	ACH	2024-11-25 14:00:00.000
4	4	PedalPay Solutions	321 Aggregator Court, Springfield, IL	Active	Nashville, TN; Boston, MA; Detroit, MI	500.00	Bank Transfer	2024-12-13 09:00:00.000
5	5	Velocity Tech Solutions	1020 Tech Avenue, San Francisco, CA	Active	New York, NY; Los Angeles, CA	2500.00	Credit Card	2024-12-12 16:45:00.000
6	6	GreenTech Services	5678 Green Drive, Seattle, WA	Inactive	Portland, OR; Boise, ID	3000.00	ACH	2024-11-15 11:30:00.000
7	7	Swift Payments	1340 Quick Street, Denver, CO	Active	Austin, TX; Denver, CO; Miami, FL	800.00	Bank Transfer	2024-12-14 13:00:00.000
8	8	Prime Solutions	883 Enterprise Way, Chicago, IL	Inactive	Chicago, IL; Atlanta, GA	1200.00	Credit Card	2024-12-02 10:00:00.000
9	9	FusionPay Systems	452 Innovation Drive, Phoenix, AZ	Active	San Diego, CA; Phoenix, AZ	1800.00	ACH	2024-12-05 14:30:00.000
10	10	QuickPay Enterprises	787 Fast Lane, Boston, MA	Active	Boston, MA; New York, NY	400.00	Bank Transfer	2024-12-08 17:00:00.000

Voyager Bicycle App Table:

	VoyagerOrgID	CustomerID	WorkOrderID	TechnicianID	BillingID
1	1	1	1	1	1
2	2	2	2	2	2
3	3	3	3	3	3
4	4	4	4	4	5
5	5	5	5	5	5
6	6	6	6	6	6
7	7	7	7	7	7
8	8	8	8	8	8
9	9	9	9	9	8
10	10	10	10	10	10

The images in this section show all the tables within the database populated with organizational data. These reports were printed using a SQL SELECT statement for all fields in each table. This data includes all users and admins, customers, service locations, technicians, work orders, billing information, transaction information and the information that populates the Voyager Bicycle App table which was provided by Voyager Bicycle Organization.

6. Testing and Validation

Test Work Order and Customer Relationship:

```
--Test WorkOrder and Customer Relationship --
SELECT
    WorkOrder.WorkOrderID,
    WorkOrder.ProblemDescription,
    Customer.CustomerName,
    Customer.BillingAddress
FROM
    WorkOrder
JOIN
    Customer ON WorkOrder.CustomerID = Customer.CustomerID;
```

.00 %

Results Messages

	WorkOrderID	ProblemDescription	CustomerName	BillingAddress
1	1	Flat tire repair	Adventure Gear	123 Elm St, Springfield, IL
2	2	Brake adjustment	Bike World	456 Oak St, Denver, CO
3	3	Gear tuning	Pedal Pros	789 Pine St, Detroit, MI
4	4	Chain replacement	Cycle Depot	321 Birch St, Boston, MA
5	5	Wheel alignment	Gear Up	654 Maple St, Nashville, TN
6	6	Replace broken spokes	Cycle Haven	101 Cedar Ave, Springfield, IL
7	7	Fix derailleur issues	Pedal Pushers	202 Walnut St, Denver, CO
8	8	Adjust seat and handlebars	Wheel Works	303 Maple Ave, Detroit, MI
9	9	Replace brake cables	Gear Junction	404 Pine Blvd, Boston, MA
10	10	Install new chain	Ride Ready	505 Birch Ln, Nashville, TN
11	11	Fix flat tire	Spin Masters	606 Elm St, Springfield, IL
12	12	Lubricate and clean drivetrain	Cyclist Pro	707 Oak Dr, Denver, CO
13	13	Replace handlebar grips	Trail Blazers	808 Chestnut Way, Detroit, MI
14	14	Check wheel alignment	The Chain Gang	909 Aspen Cir, Boston, MA
15	15	Install new pedals	Trek Squad	1001 Redwood Ct, Nashville, TN
16	16	Brake pad replacement	Mountain Movers	1101 Cypress Rd, Springfield, IL
17	17	Inspect and clean frame	Path Finders	1201 Fir Ln, Denver, CO
18	18	Gear adjustment	Cycle Crafters	1301 Spruce Ave, Detroit, MI
19	19	Fix wobbling wheel	Two Wheelers	1401 Hickory St, Boston, MA
20	20	Replace bottom bracket	Bicycle Enthusiasts	1501 Poplar Blvd, Nashville, TN

Test Service Location Detail on Work Order:

```
--Test Service Location Detail on Work Orders --
SELECT
    WorkOrder.WorkOrderID,
    WorkOrder.ProblemDescription,
    ServiceLocation.LocationName,
    ServiceLocation.Address
FROM
    WorkOrder
JOIN
    ServiceLocation ON WorkOrder.LocationID = ServiceLocation.LocationID;
```

100 %

Results Messages

	WorkOrderID	ProblemDescription	LocationName	Address
1	1	Flat tire repair	Voyager Bicycle Downtown	123 Elm St, Springfield, IL
2	2	Brake adjustment	Voyager Bicycle West	456 Oak St, Denver, CO
3	3	Gear tuning	Voyager Bicycle North	789 Pine St, , Detroit, MI
4	4	Chain replacement	Voyager Bicycle East	321 Birch St, Boston, MA
5	5	Wheel alignment	Voyager Bicycle South	654 Maple St, Nashville, TN
6	6	Replace broken spokes	Voyager Bicycle Downtown	123 Elm St, Springfield, IL
7	7	Fix derailleur issues	Voyager Bicycle West	456 Oak St, Denver, CO
8	8	Adjust seat and handlebars	Voyager Bicycle North	789 Pine St, , Detroit, MI
9	9	Replace brake cables	Voyager Bicycle East	321 Birch St, Boston, MA
10	10	Install new chain	Voyager Bicycle South	654 Maple St, Nashville, TN
11	11	Fix flat tire	Voyager Bicycle Downtown	123 Elm St, Springfield, IL
12	12	Lubricate and clean drivetrain	Voyager Bicycle West	456 Oak St, Denver, CO
13	13	Replace handlebar grips	Voyager Bicycle North	789 Pine St, , Detroit, MI
14	14	Check wheel alignment	Voyager Bicycle East	321 Birch St, Boston, MA
15	15	Install new pedals	Voyager Bicycle South	654 Maple St, Nashville, TN
16	16	Brake pad replacement	Voyager Bicycle Downtown	123 Elm St, Springfield, IL
17	17	Inspect and clean frame	Voyager Bicycle West	456 Oak St, Denver, CO
18	18	Gear adjustment	Voyager Bicycle North	789 Pine St, , Detroit, MI
19	19	Fix wobbling wheel	Voyager Bicycle East	321 Birch St, Boston, MA
20	20	Replace bottom bracket	Voyager Bicycle South	654 Maple St, Nashville, TN

Test Billing Information for Work Order:

--Test Billing Information for Work Orders --				
SELECT				
WorkOrder.WorkOrderID,				
Billing.TotalAmount,				
Billing.InvoiceDate,				
BillingAggregator.AggregatorName				
FROM				
Billing				
JOIN				
WorkOrder ON Billing.WorkOrderID = WorkOrder.WorkOrderID				
JOIN				
BillingAggregator ON Billing.BillingAggregatorID = BillingAggregator.AggregatorID				
100 %				
Results Messages				
	WorkOrderID	TotalAmount	InvoiceDate	AggregatorName
1	1	55.00	2024-12-01 17:00:00.0000000	CycleBilling LLC
2	2	49.50	2024-12-02 18:00:00.0000000	CycleBilling LLC
3	5	52.25	2024-12-05 19:00:00.0000000	Velocity Accounts
4	8	60.50	2024-12-08 14:00:00.0000000	WheelWorks Financials
5	10	66.00	2024-12-10 12:00:00.0000000	Velocity Accounts
6	11	55.00	2024-12-11 11:00:00.0000000	CycleBilling LLC
7	13	44.00	2024-12-13 11:00:00.0000000	PedalPay Solutions
8	15	71.50	2024-12-15 13:00:00.0000000	CycleBilling LLC
9	16	82.50	2024-12-16 13:00:00.0000000	Velocity Accounts
10	18	66.00	2024-12-18 12:00:00.0000000	WheelWorks Financials
11	20	99.00	2024-12-20 14:00:00.0000000	PedalPay Solutions

Test Technician Assignment for Work Order:

```
-- Test Technican Assignment for Work Orders --
SELECT
    WorkOrder.WorkOrderID,
    WorkOrder.ProblemDescription,
    Technician.TechnicianName,
    Technician.TechnicianRate
FROM
    WorkOrder
JOIN
    Technician ON WorkOrder.TechnicianID = Technician.TechnicianID;
```

10 %

Results Messages

	WorkOrderID	ProblemDescription	TechnicianName	TechnicianRate
1	1	Flat tire repair	Alice Brown	50.00
2	2	Brake adjustment	Bob White	45.00
3	3	Gear tuning	Charlie Green	48.50
4	4	Chain replacement	Diana Blue	52.00
5	5	Wheel alignment	Edward Black	47.50
6	6	Replace broken spokes	George Miller	46.75
7	7	Fix derailleur issues	Helen Carter	50.25
8	8	Adjust seat and handlebars	Ian Thompson	47.00
9	9	Replace brake cables	Julia Roberts	48.50
10	10	Install new chain	Kyle Evans	49.75
11	11	Fix flat tire	Laura Hill	51.00
12	12	Lubricate and clean drivetrain	Mark Patterson	46.50
13	13	Replace handlebar grips	Nina Brown	47.25
14	14	Check wheel alignment	Oscar Reed	48.00
15	15	Install new pedals	Paul Adams	49.00
16	16	Brake pad replacement	Quinn Davis	50.00
17	17	Inspect and clean frame	Rachel Clark	51.50
18	18	Gear adjustment	Steve Wilson	52.00
19	19	Fix wobbling wheel	Tina Harris	46.00
20	20	Replace bottom bracket	Victor White	47.50

Test Voyager Bicycle App Table Connections:

```
-- Voyager Bicycle App Table Connections --
SELECT
    VoyagerBicycleApp.VoyagerOrgID,
    Customer.CustomerName,
    WorkOrder.ProblemDescription,
    Technician.TechnicianName,
    Billing.TotalAmount
FROM
    VoyagerBicycleApp
JOIN
    Customer ON VoyagerBicycleApp.CustomerID = Customer.CustomerID
JOIN
    WorkOrder ON VoyagerBicycleApp.WorkOrderID = WorkOrder.WorkOrderID
JOIN
    Technician ON VoyagerBicycleApp.TechnicianID = Technician.TechnicianID
JOIN
    Billing ON VoyagerBicycleApp.BillingID = Billing.BillingID;
```

100 %

Results Messages

	VoyagerOrgID	CustomerName	ProblemDescription	TechnicianName	TotalAmount
1	1	Adventure Gear	Flat tire repair	Alice Brown	55.00
2	2	Bike World	Brake adjustment	Bob White	49.50
3	3	Pedal Pros	Gear tuning	Charlie Green	52.25
4	4	Cycle Depot	Chain replacement	Diana Blue	66.00
5	5	Gear Up	Wheel alignment	Edward Black	66.00
6	6	Cycle Haven	Replace broken spokes	George Miller	55.00
7	7	Pedal Pushers	Fix derailleur issues	Helen Carter	44.00
8	8	Wheel Works	Adjust seat and handlebars	Ian Thompson	71.50
9	9	Gear Junction	Replace brake cables	Julia Roberts	71.50
10	10	Ride Ready	Install new chain	Kyle Evans	66.00

Test Work Orders with Pending Status:

WorkOrderProcessD...(TOWER\Matt (60))

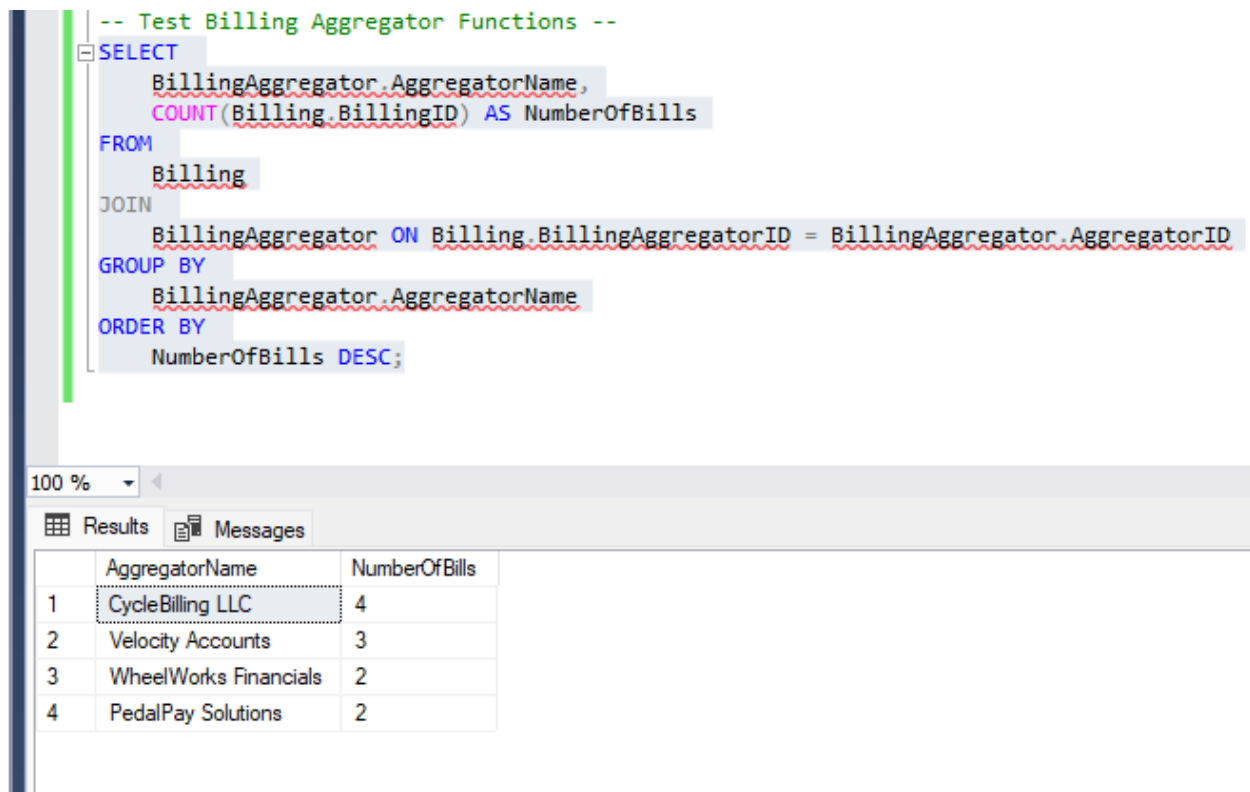
```
-- Test Work Orders with Pending Status --  
SELECT  
    WorkOrder.WorkOrderID,  
    Customer.CustomerName,  
    Technician.TechnicianName,  
    WorkOrder.ProblemDescription,  
    WorkOrder.Status  
FROM  
    WorkOrder  
JOIN  
    Customer ON WorkOrder.CustomerID = Customer.CustomerID  
JOIN  
    Technician ON WorkOrder.TechnicianID = Technician.TechnicianID  
WHERE  
    WorkOrder.Status = 'Pending';
```

100 %

Results Messages

	WorkOrderID	CustomerName	TechnicianName	ProblemDescription	Status
1	3	Pedal Pros	Charlie Green	Gear tuning	Pending
2	7	Pedal Pushers	Helen Carter	Fix derailleur issues	Pending
3	12	Cyclist Pro	Mark Patterson	Lubricate and clean drivetrain	Pending
4	17	Path Finders	Rachel Clark	Inspect and clean frame	Pending

Test Billing Aggregator Functions:



```
-- Test Billing Aggregator Functions --
SELECT
    BillingAggregator.AggregatorName,
    COUNT(Billing.BillingID) AS NumberOfBills
FROM
    Billing
JOIN
    BillingAggregator ON Billing.BillingAggregatorID = BillingAggregator.AggregatorID
GROUP BY
    BillingAggregator.AggregatorName
ORDER BY
    NumberOfBills DESC;
```

	AggregatorName	NumberOfBills
1	CycleBilling LLC	4
2	Velocity Accounts	3
3	WheelWorks Financials	2
4	PedalPay Solutions	2

The images in this section show all the queries that were run to validate that the database relationships functioned properly. These queries include several SQL statements such as JOIN statements, to ensure all relationships function as intended. By going through this process, the project team ensured that the necessary functions needed by the application being developed by the Voyager Bicycle Organization's IT team would be supported. In addition to the queries being used to test the database functions, the queries can also be used to support several different functions that could be applied to the application being developed.

7. Queries

Query For Technician Availability:

SQLQuery20.sql - 1... (LOWER(Matt (51))) WorkOrderProcessD... (LOWER(Matt (

```
USE WorkOrderProcessDatabase;  
  
-- Query for Technician Availability --  
SELECT  
    TechnicianID,  
    TechnicianName,  
    ContactPhone,  
    AvailabilityStatus  
FROM  
    Technician  
WHERE  
    AvailabilityStatus = 'Available';
```

100 %

Results Messages

	TechnicianID	TechnicianName	ContactPhone	AvailabilityStatus
1	1	Alice Brown	555-0300	Available
2	3	Charlie Green	555-0302	Available
3	5	Edward Black	555-0304	Available
4	6	George Miller	555-0305	Available
5	8	Ian Thompson	555-0307	Available
6	10	Kyle Evans	555-0309	Available
7	11	Laura Hill	555-0310	Available
8	13	Nina Brown	555-0312	Available
9	14	Oscar Reed	555-0313	Available
10	16	Quinn Davis	555-0315	Available
11	18	Steve Wilson	555-0317	Available
12	20	Victor White	555-0319	Available

Query for Customer Pending Work Orders:

SQLQuery20.sql - T...(TOWER\Matt (51)) WorkOrderProcessD...(TOWER\Matt (59))* X Wor

```
-- Query for Customer Pending Work Orders --  
SELECT  
    WorkOrder.WorkOrderID,  
    Customer.CustomerName,  
    ServiceLocation.LocationName,  
    WorkOrder.ProblemDescription,  
    WorkOrder.Status  
FROM  
    WorkOrder  
INNER JOIN  
    Customer ON WorkOrder.CustomerID = Customer.CustomerID  
INNER JOIN  
    ServiceLocation ON WorkOrder.LocationID = ServiceLocation.LocationID  
WHERE  
    WorkOrder.Status = 'Pending';
```

100 %

Results Messages

	WorkOrderID	CustomerName	LocationName	ProblemDescription	Status
1	3	Pedal Pros	Voyager Bicycle North	Gear tuning	Pending
2	7	Pedal Pushers	Voyager Bicycle West	Fix derailleur issues	Pending
3	12	Cyclist Pro	Voyager Bicycle West	Lubricate and clean drivetrain	Pending
4	17	Path Finders	Voyager Bicycle West	Inspect and clean frame	Pending

Query For Customer Billing Summary:

```
-- Query for Billing Summary --
SELECT
    Customer.CustomerName,
    Billing.InvoiceDate,
    Billing.TotalAmount,
    Billing.MarkupPercentage,
    WorkOrder.WorkOrderID,
    WorkOrder.ProblemDescription,
    WorkOrder.StartDateTime,
    WorkOrder.EndDateTime
FROM
    Billing
INNER JOIN
    WorkOrder ON Billing.WorkOrderID = WorkOrder.WorkOrderID
INNER JOIN
    Customer ON WorkOrder.CustomerID = Customer.CustomerID
WHERE
    Customer.CustomerID = 5 -- Choose CustomerID for Specific Customer, in this case the CustomerID = 5 --
ORDER BY
    Billing.InvoiceDate DESC;
```

100 %

Results Messages

	CustomerName	InvoiceDate	TotalAmount	MarkupPercentage	WorkOrderID	ProblemDescription	StartDateTime	EndDateTime
1	Gear Up	2024-12-05 19:00:00.0000000	52.25	10.00	5	Wheel alignment	2024-12-05 15:00:00.0000000	2024-12-05 16:30:00.0000000

The images in this section show three essential queries that can be utilized by the application being developed by the Voyager Bicycle Organization IT team. These essential queries include the SQL statements necessary to print information showing technician availability, pending work orders, as well as a query to quickly print a customer's billing summary. These queries along with the queries that were used to test and validate the database's relationships allowed the project team to meet the functional requirements defined for this database.

8. Work Order Report

Work Order Report SQL Code:

```
-- Work Order Processing Report --
DECLARE @WorkOrderID INT = 1; -- Replace with the desired WorkOrderID, In this example WorkOrderID is 1

SELECT
    WO.WorkOrderID,
    WO.StartDateTime,
    WO.EndDateTime,
    WO.ProblemDescription,
    WO.TechnicianEstimate,
    SL.LocationName,
    SL.Address AS ServiceLocationAddress,
    C.CustomerName,
    C.BillingContactEmail,
    C.BillingContactPhone,
    T.TechnicianName,
    T.TechnicianRate,
    WO.TimeSpent,
    B.MarkupPercentage,
    WO.TechnicianID,
    -- Convert TimeSpent from TIME to a decimal value representing total hours
    ((CAST(DATEDIFF(SECOND, '00:00:00', WO.TimeSpent) AS DECIMAL) / 3600) * T.TechnicianRate * (1 + B.MarkupPercentage / 100.0)) AS TotalCost
FROM
    WorkOrder WO
INNER JOIN
    ServiceLocation SL ON WO.LocationID = SL.LocationID -- Join WorkOrder with ServiceLocation
INNER JOIN
    Customer C ON WO.CustomerID = C.CustomerID -- Join WorkOrder with Customer
LEFT JOIN
    Technician T ON WO.TechnicianID = T.TechnicianID -- Technician info
LEFT JOIN
    Billing B ON WO.WorkOrderID = B.WorkOrderID -- Billing info
WHERE
    WO.WorkOrderID = @WorkOrderID; -- Using the declared variable
```

Work Order Report Example 1:

WorkOrderProcessID: (POWER.Matt (51)) | WorkOrderProcessID: (POWER.Matt (56)) | WorkOrderProcessID: (POWER.Matt (60))

```
USE WorkOrderProcessDatabase;

-- Work Order Processing Report --
DECLARE @WorkOrderID INT = 1; -- Replace with the desired WorkOrderID, In this example WorkOrderID is 1

SELECT
    WO.WorkOrderID,
    WO.StartDateTime,
    WO.EndDateTime,
    WO.ProblemDescription,
    WO.TechnicianEstimate,
    SL.LocationName,
    SL.Address AS ServiceLocationAddress,
    C.CustomerName,
    C.BillingContactEmail,
    C.BillingContactPhone,
    T.TechnicianName,
    T.TechnicianRate,
    WO.TimeSpent,
    B.MarkupPercentage,
    WO.TechnicianID,
    -- Convert TimeSpent from TIME to a decimal value representing total hours
    ((CAST(DATEDIFF(SECOND, '00:00:00', WO.TimeSpent) AS DECIMAL) / 3600) * T.TechnicianRate * (1 + B.MarkupPercentage / 100.0)) AS TotalCost
FROM
    WorkOrder WO
INNER JOIN
    ServiceLocation SL ON WO.LocationID = SL.LocationID -- Join WorkOrder with ServiceLocation
INNER JOIN
    Customer C ON WO.CustomerID = C.CustomerID -- Join WorkOrder with Customer
LEFT JOIN
    Technician T ON WO.TechnicianID = T.TechnicianID -- Technician info
LEFT JOIN
    Billing B ON WO.WorkOrderID = B.WorkOrderID -- Billing info
WHERE
    WO.WorkOrderID = @WorkOrderID; -- Using the declared variable
```

100 %

WorkOrderID	StartDateTime	EndDateTime	ProblemDescription	TechnicianEstimate	LocationName	ServiceLocationAddress	CustomerName	BillingContactEmail	BillingContactPhone	TechnicianName	TechnicianRate	TimeSpent	MarkupPercentage	TechnicianID	TotalCost
1	2024-12-01 09:00:00.0000000	2024-12-01 10:00:00.0000000	Flat tire repair	50.00	Voyager Bicycle Downtown	123 Elm St, Springfield, IL	Adventure Gear	john.doe@example.com	555-0100	Alice Brown	50.00	01:00:00.0000000	10.00	1	55.0000000

Work Order Report Example 2:

```

USE WorkOrderProcessDatabase;

-- Work Order Processing Report --
DECLARE @WorkOrderID INT = 8; -- Replace with the desired WorkOrderID, In this example WorkOrderID is 8

SELECT
    WO.WorkOrderID,
    WO.StartDateTime,
    WO.EndDateTime,
    WO.ProblemDescription,
    WO.TechnicianEstimate,
    SL.LocationName,
    SL.Address AS ServiceLocationAddress,
    C.CustomerName,
    C.BillingContactEmail,
    C.BillingContactPhone,
    T.TechnicianName,
    T.TechnicianRate,
    WO.TimeSpent,
    B.MarkupPercentage,
    WO.TechnicianID,
    -- Convert TimeSpent from TIME to a decimal value representing total hours
    ((CAST(CONVERT(SMALLTEXT, '00:00:00') AS DECIMAL) / 3600) * T.TechnicianRate * (1 + B.MarkupPercentage / 100.0)) AS TotalCost
FROM
    WorkOrder WO
INNER JOIN
    ServiceLocation SL ON WO.LocationID = SL.LocationID -- Join WorkOrder with ServiceLocation
INNER JOIN
    Customer C ON WO.CustomerID = C.CustomerID -- Join WorkOrder with Customer
LEFT JOIN
    Technician T ON WO.TechnicianID = T.TechnicianID -- Technician Info
LEFT JOIN
    Billing B ON WO.WorkOrderID = B.WorkOrderID -- Billing Info
WHERE
    WO.WorkOrderID = @WorkOrderID -- Using the declared variable

```

WorkOrderID	StartDateTime	EndDateTime	ProblemDescription	TechnicianEstimate	LocationName	ServiceLocationAddress	CustomerName	BillingContactEmail	BillingContactPhone	TechnicianName	TechnicianRate	TimeSpent	MarkupPercentage	TechnicianID	TotalCost
8	2024-12-08 13:00:00.0000000	2024-12-08 13:45:00.0000000	Adjust seat and handlebars	55.00	Voyager Bicycle North	789 Pine St., Detroit, MI	Wheel Works	sophia.tuner@example.com	555-1004	Ian Thompson	47.00	00:45:00.0000000	10.00	8	38.775000

The images in this section show the SQL statements and code used to generate the 'Work Order Report'. This was the most important requirement of this database as it allows technicians to print a work order summary to view and show customers. This report includes: the Work Order ID, The Start and End Data and Time, the Problem Description, the Technician's Estimate for the work, the service location where the work order is being performed, the Service Location Address, the Customer Name, Email, Phone, the Technician's name who is assigned to the work order, the Technician's Rate, the time spent on the work order, the mark up percentage, the Technician ID and the total cost of the work order if the work order is complete. If the work order has not been completed the last value in the report will default to NULL. Larger images of these reports can be viewed in the product submission folder.

9. Database Access Control

Least Privilege Role Based Access Control:

```

-- Use WorkOrderProcessDatabase;

-- Create Database Roles--
CREATE ROLE Admin;
CREATE ROLE Technician;

-- Grant Permissions to Admins --
GRANT SELECT, INSERT, UPDATE, DELETE ON AdminUser TO Admin;
GRANT SELECT, INSERT, UPDATE, DELETE ON WorkOrder TO Admin;
GRANT SELECT, INSERT, UPDATE, DELETE ON Billing TO Admin;
GRANT SELECT, INSERT, UPDATE, DELETE ON Customer TO Admin;
GRANT SELECT, INSERT, UPDATE, DELETE ON ServiceLocation TO Admin;
GRANT SELECT, INSERT, UPDATE, DELETE ON Technician TO Admin;
GRANT SELECT, INSERT, UPDATE, DELETE ON BillingAggregator TO Admin;

-- Grant Permissions to Technicians --
GRANT SELECT, UPDATE ON WorkOrder TO Technician;
GRANT SELECT ON Billing TO Technician;
GRANT SELECT ON Customer TO Technician;
GRANT SELECT ON ServiceLocation TO Technician;

-- Assign Roles to Users --
EXEC sp_addrolemember 'Admin', 'admin_jane';
EXEC sp_addrolemember 'Admin', 'admin_paul';
EXEC sp_addrolemember 'Admin', 'admin_emily';
EXEC sp_addrolemember 'Technician', 'alice.brown';
EXEC sp_addrolemember 'Technician', 'bob.white';
EXEC sp_addrolemember 'Technician', 'charlie.green';
EXEC sp_addrolemember 'Technician', 'diana.blue';
EXEC sp_addrolemember 'Technician', 'edward.black';
EXEC sp_addrolemember 'Technician', 'george.miller';
EXEC sp_addrolemember 'Technician', 'helen.carter';
EXEC sp_addrolemember 'Technician', 'ian.thompson';
EXEC sp_addrolemember 'Technician', 'julia.roberts';
EXEC sp_addrolemember 'Technician', 'kyle.evans';
EXEC sp_addrolemember 'Technician', 'laura.hill';
EXEC sp_addrolemember 'Technician', 'mark.patterson';
EXEC sp_addrolemember 'Technician', 'nina.brown';
EXEC sp_addrolemember 'Technician', 'oscar.reed';
EXEC sp_addrolemember 'Technician', 'paul.adams';
EXEC sp_addrolemember 'Technician', 'quinn.davis';
EXEC sp_addrolemember 'Technician', 'rachel.clark';
EXEC sp_addrolemember 'Technician', 'steve.wilson';
EXEC sp_addrolemember 'Technician', 'tina.harris';
EXEC sp_addrolemember 'Technician', 'victor.white';

```

Admin/User Table Data:

```
-- Create the AdminUser table
CREATE TABLE AdminUser (
    UserID INT IDENTITY(1,1) PRIMARY KEY, -- Auto-incrementing primary key
    UserName VARCHAR(255) NOT NULL,      -- The username of the admin or user
    UserRole VARCHAR(50) NOT NULL,        -- The role (e.g., Admin, User)
    Password VARCHAR(255) NOT NULL       -- The password (hashed for security)
);

-- Add the DateCreated column with a default value
ALTER TABLE AdminUser
ADD DateCreated DATETIME2 DEFAULT SYSDATETIME();

-- Add the LastLogin column, allowing NULL values initially
ALTER TABLE AdminUser
ADD LastLogin DATETIME2 NULL;

-- Add FailedLoginAttempts column, default is 0 --
ALTER TABLE AdminUser
ADD FailedLoginAttempts INT DEFAULT 0;

-- Constraint for UserRoles --
ALTER TABLE AdminUser
ADD CONSTRAINT CHK_UserRole CHECK (UserRole IN ('Admin', 'Technician'));
```

One of the security requirements for this database was to implement least privilege access control. This is a security principle that only provides system access to users for only the task they need to complete and nothing more. For this database only the administrators working in the Work Order Processing department have full control and access to the database. The information provided to our project team provided us with three members of this department to be implemented as Admin roles in this database. Technicians have the ability to 'Update' and view the information on the Work Order table as well as view information on the Billing, Customer, and Service Location tables. The image in this section shows the SQL code and statements necessary to implement this security requirement. In this image, roles were created, and permissions were added for each of the roles. Then finally roles were assigned to admins and technicians defined in the Admin/User table. Additional auditing is available

in this database as the User/Admin table includes fields for data to be captured such as a date for 'Last Login Date' and 'Failed Login Attempts'. The second image highlights these fields during the creation of this table. These values default to 'Null' or '0' until these users access the database.

10. Database Migration

Migrating this database to the server or system set up by the Voyager Bicycle Organization IT team can be done by running the database schema script included in the project folder. Once the IT team has moved this SQL file to the correct server or system they must open the SQL schema script on their SQL environment and execute the script. Once the script has been correctly executed, the database will be fully built and populated with all necessary tables, fields, constraints, primary and foreign keys as well as all the organizational data that was provided to the project team. Once the database is migrated, it is important to update the 'AdminUser' roles if anything has changed since the project data was collected. Run SELECT statements on all the existing tables to confirm organizational data is included. Run the testing queries provided in the project folder to ensure table and key relationships are working as intended. Backup the database by generating another schema script and store it in another location. Update the server configuration so settings are adjusted to meet the database's needs. An important setting to check is the max allocated memory defined for the database system; this setting ensures the database operates at optimal performance.