



Pedal – Cycling Unit

INFO 6210 – Project Team 18

Abhishek Satbhai

Chinmayi Shaligram

Gayatri Degaonkar

Mohammed Umair Ul Hasan

Pavan Pati



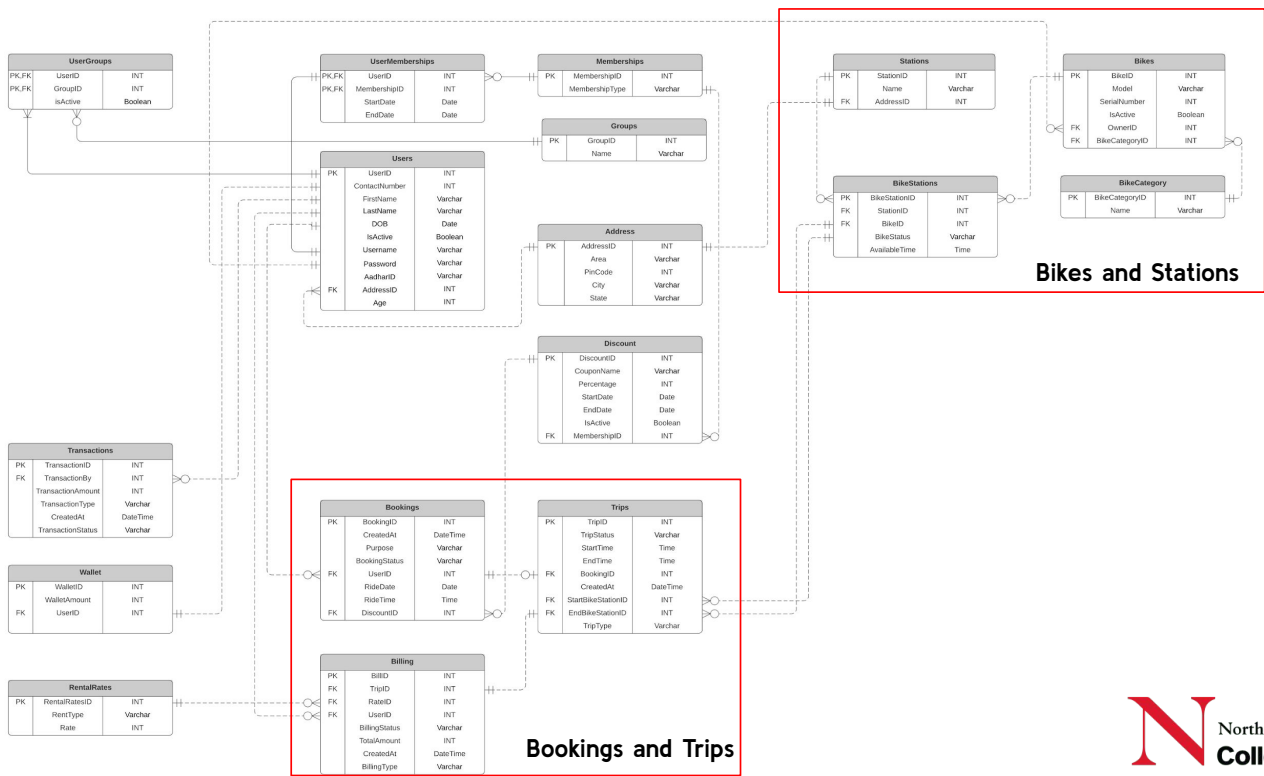


Why Pedal ?

Key Objectives

Business Problems Addressed

- Centralized cycling system
- Eco-friendly way of transportation
- Bicycle owners can contribute
- Cost efficient
- Avail benefits of various discounts offered



Stored Procedures



```
CREATE PROCEDURE dbo.StartTrip # SP for Start Trip
@BookingID INT, @StationID INT, @BikeID INT,
@TripType Varchar(10), @StartBikeStationID INT;
AS
BEGIN
    DECLARE @BikeStatus Varchar(30);
    IF(SELECT BikeStatus FROM BikeStations
        WHERE StationID=@StationID AND BikeID=@BikeID)='Available'
    BEGIN
        SELECT @StartBikeStationID = BikeStationID FROM BikeStations
        WHERE StationID = @StationID AND BikeID = @BikeID
        INSERT INTO Trips(BookingID, StartBikeStationID, TripType)
        VALUES (@BookingID, @StartBikeStationID, @TripType);
        BEGIN
            IF (@TripType = 'By User') SET @BikeStatus = 'Not Available';
            ELSE IF (@TripType = 'By Owner') SET @BikeStatus = 'Available'
            UPDATE [dbo].[BikeStations]
            SET BikeStatus = @BikeStatus, AvailableTime = CURRENT_TIMESTAMP
            WHERE BikeStationID = @StartBikeStationID
        END
    END
ELSE
    PRINT('Bike At This Station Is not Available')
END
```

```
CREATE PROCEDURE dbo.EndTrip # SP for End Trip
@TripID Int, @StationID Int AS
BEGIN
    DECLARE @BikeStatus VARCHAR(30),@TripType Varchar(20),@BikeID INT;
    SELECT @BikeID = BikeID,@TripType=TripType FROM Trips t JOIN BikeStations s
    ON t.StartBikeStationID = s.BikeStationID Where TripID = @TripID
    IF EXISTS(SELECT StationID,BikeID FROM BikeStations s
        WHERE (s.BikeID = @BikeID AND s.StationID = @StationID))
    BEGIN
        IF (@TripType = 'By User') SET @BikeStatus = 'Available'
        ELSE IF(@TripType = 'By Owner') SET @BikeStatus = 'Not Available'
        UPDATE [dbo].[BikeStations]
        SET BikeStatus = @BikeStatus, AvailableTime = CURRENT_TIMESTAMP
        WHERE StationID = @StationID AND BikeID = @BikeID
    END
    ELSE
    BEGIN
        IF (@TripType = 'By User') SET @BikeStatus = 'Available'
        ELSE IF(@TripType = 'By Owner') SET @BikeStatus = 'Not Available'
        INSERT INTO BikeStations(BikeID,StationID,BikeStatus,AvailableTime)
        Values(@BikeID,@StationID,@BikeStatus,CURRENT_TIMESTAMP)
    END
    UPDATE [dbo].[Trips]
    SET TripStatus = 'Completed', EndDateTime = CURRENT_TIMESTAMP,
    EndBikeStationID = (SELECT BikeStationID FROM BikeStations
        WHERE StationID = @StationID AND BikeID = @BikeID)
    WHERE TripID = @TripID;
END
```

Stored Procedures



```
CREATE PROCEDURE [dbo].[AddBilling] # SP to determine Time and Rate for Billing
@TripID INT AS
BEGIN
    DECLARE @Hours FLOAT, @Minutes FLOAT, @Rate FLOAT, @Time FLOAT;
    DECLARE @BillingType Varchar(20), @TripType Varchar(20), @BookingID INT;
    DECLARE @RentalRatesID INT, @TotalAmount FLOAT, @TAmount FLOAT;
    DECLARE @StartTime DATETIME, @EndTime DATETIME, @UserID INT;
    DECLARE @BillingStatus Varchar(20) = 'Successful', @Percentage FLOAT;
    SELECT @UserID=UserID, @TripType=TripType, @StartTime=StartDateTime, @EndTime=EndDateTime
    FROM Trips t INNER JOIN Bookings b ON t.BookingID = b.BookingID WHERE t.TripID = @TripID
    SET @Minutes = DATEDIFF(Minute, @StartTime, @EndTime);
    SET @Hours = @Minutes/60
    IF (@TripType = 'By User')
    BEGIN
        SET @BillingType = 'Debit'
        IF (@Hours < 24)
        BEGIN SET @Time = @Hours; SET @RentalRatesID = 1; END
        ELSE
        BEGIN SET @Time = @Hours/24; SET @RentalRatesID = 2; END
    END
    ELSE IF (@TripType = 'By Owner')
    BEGIN
        SET @BillingType = 'Credit'
        IF (@Hours < 24)
        BEGIN SET @Time = @Hours; SET @RentalRatesID = 3; END
        ELSE
        BEGIN SET @Time = @Hours/24; SET @RentalRatesID = 4; END
    END
    INSERT INTO Billing (TripID, RateID, UserID, BillingStatus, BillingType)
    VALUES (@TripID, @RentalRatesID, @UserID, @BillingStatus, @BillingType)
END
```

Automation



End Trip

Billing

Transactions

Wallet

Bills are generated after trip Ends, according to Trip time and Rent type.

Transactions for Users/Owners are recorded after generating Bills.

Bill Amount is debited/credited to Wallet.

	TransactionID	TransactionBy	TransactionAmount	TransactionType	CreatedAt	TransactionStatus
1	1	2	0.8166666666666667	Debit	2020-08-11 09:08:24.770	Successful
2	2	3	1.05	Debit	2020-08-11 09:10:17.047	Successful
3	3	4	1.1666666666666665	Debit	2020-08-11 09:11:00.283	Successful

	BillID	TripID	RateID	UserID	BillingStatus	CreatedAt	BillingType	TotalAmount
1	1	1	1	2	Successful	2020-08-11	Debit	0.8166666666666667
2	3	2	1	3	Successful	2020-08-11	Debit	1.05
3	4	3	1	4	Successful	2020-08-11	Debit	1.1666666666666665

	WalletID	WalletAmount	UserID
1	1	505	1
2	2	508.01666666666665	2
3	3	528.95	3

Triggers



Trigger for Billing automation

```
CREATE Trigger [dbo].[AfterEndTrip]
ON Trips
AFTER UPDATE
AS
BEGIN
    DECLARE @LatestEndedTripID INT;
    SELECT TOP 1 @LatestEndedTripID = inserted.TripID
    FROM inserted
    EXECUTE [dbo].[AddBilling] @LatestEndedTripID;
END
```

Trigger For Automating Transaction and Wallet Amount

```
CREATE Trigger [dbo].[AfterBillingUpdates]
ON Billing
AFTER INSERT
AS BEGIN
    DECLARE @LatestBillingID INT;
    DECLARE @TransactionBy INT;
    DECLARE @TransactionAmount FLOAT;
    DECLARE @TransactionType Varchar(20);
    SELECT TOP 1 @LatestBillingID = inserted.BillID
    FROM inserted
    SELECT @TransactionBy= [UserID],
    @TransactionAmount=[TotalAmount],
    @TransactionType = [BillingType]
    FROM Billing WHERE BillID = @LatestBillingID
    EXECUTE dbo.AddTransactions @TransactionBy,
    @TransactionAmount, @TransactionType ;
END
```

Trigger for Booking Cancellation and Wallet constraints

```
CREATE TRIGGER [dbo].[AfterBookingUpdates]
ON Bookings
AFTER INSERT,UPDATE
AS
BEGIN
    DECLARE @WalletAmount FLOAT
    DECLARE @BookingStatus Varchar(30)
    DECLARE @TransactionBy INT
    DECLARE @TransactionAmount FLOAT
    DECLARE @TransactionType Varchar(20)
    DECLARE @BookingID INT
    Select @WalletAmount= [WalletAmount] FROM [Wallet] WHERE UserID =
    (SELECT inserted.UserID FROM inserted)
    IF EXISTS (SELECT inserted.BookingID FROM inserted) AND EXISTS
    (SELECT deleted.BookingID FROM deleted)
    BEGIN
        SELECT @BookingStatus=[BookingStatus], @TransactionBy=[UserID]
        FROM Bookings WHERE BookingID = (SELECT inserted.BookingID FROM inserted)
        IF @BookingStatus = 'Cancelled'
            EXECUTE dbo.AddTransactions @TransactionBy,@TransactionAmount=10,
            @TransactionType = 'Debit';
    END
    ELSE
    BEGIN
        IF @WalletAmount < 500
        BEGIN
            ROLLBACK TRAN
            RAISERROR('Booking Failed ! Wallet Amount is less than 500,
            add money to your wallet',16,1)
        END
    END
END
```

Table Level Check



For validation user has wallet amount more than 500/- at the time of booking

```
CREATE FUNCTION WalletBalanceCheck (@UserID INT)
RETURNS FLOAT
AS
BEGIN
    DECLARE @Balance FLOAT;
    (SELECT @Balance = WalletAmount FROM dbo.Wallet
    WHERE UserID = @UserID)
    RETURN @Balance
END;
```

Adding Constraints

```
ALTER TABLE dbo.Bookings
ADD CONSTRAINT WalletBalanceOfUser
CHECK(dbo.WalletBalanceCheck(UserID) > 500)
```

Check user has age greater than 14 at the time of registration

```
CREATE FUNCTION CheckAge (@UserID INT)
RETURNS INT
AS
BEGIN
    DECLARE @Age INT;
    SET @Age = dbo.CalculateAge(@UserID)
    RETURN @Age
END;
```

Adding Constraints

```
ALTER TABLE Users
ADD CONSTRAINT CheckAgeOfUser CHECK
(dbo.CheckAge(UserID) > 14)
```

Messages

22:13:02	Started executing query at Line 584 Msg 547, Level 16, State 0, Procedure dbo.AddUpdateUser, Line 32 The INSERT statement conflicted with the CHECK constraint "CheckAgeOfUser". The conflict occurred in database "pedal", table "dbo.Users", column 'UserID'. The statement has been terminated. Total execution time: 00:00:00.254
22:09:21	Started executing query at Line 838 Msg 547, Level 16, State 0, Line 2 The INSERT statement conflicted with the CHECK constraint "WalletBalanceOfUser". The conflict occurred in database "pedal", table "dbo.Bookings", column 'UserID'. The statement has been terminated. Total execution time: 00:00:00.262

Computed Columns



Calculate Total Amount for Billing

```
CREATE FUNCTION CalculateTotalAmount(@TripID INT, @RentalRatesID INT)
RETURNS FLOAT
AS BEGIN
    DECLARE @Hours FLOAT, @Minutes FLOAT, @Rate FLOAT, @Time FLOAT;
    DECLARE @TripType Varchar(20);
    DECLARE @TotalAmount FLOAT, @TAmount FLOAT, @UserID INT, @Percentage FLOAT, @BookingID INT;
    DECLARE @StartTime DATETIME, @EndTime DATETIME;

    SELECT @StartTime=[StartDateTime], @EndTime=[EndDateTime], @BookingID = b.BookingID, @TripType=[TripType]
    FROM Trips t INNER JOIN Bookings b ON t.BookingID = b.BookingID
    WHERE t.TripID = @TripID

    SELECT @Percentage = (SELECT [Percentage] FROM Discount d INNER JOIN Bookings b ON d.DiscountID =
    b.DiscountID WHERE b.BookingID = @BookingID)
    SET @Minutes = DATEDIFF(Minute, @StartTime, @EndTime);
    SET @Hours = @Minutes/60
    IF (@Hours < 24)
        SET @Time = @Hours;
    ELSE
        SET @Time = @Hours/24;
    IF @Percentage IS NULL SET @Percentage = 0.0
    SELECT @Rate = (SELECT Rate FROM RentalRates WHERE RentalRatesID = @RentalRatesID)
    SET @TAmount = @Rate * @Time
    SET @TotalAmount = @TAmount - (@Percentage*@TAmount)/100
    RETURN @TotalAmount
END
```

Formula

Views



View for Wallet details of User

```
CREATE VIEW User_Wallet
AS
SELECT U.UserID, W.WalletID,
       U.FirstName, U.LastName,
       W.WalletAmount
FROM Users U
JOIN Wallet W
ON U.UserID = W.UserID
```

	UserID	WalletID	FirstName	LastName	WalletAmount
1	4	1	Pavan	pati	523.6
2	13	8	james	bond	999
3	14	9	james	bond	0
4	5	10	Abhishek	Satbhai	505
5	6	11	Krishna	Pallela	505

User_Wallet

#View to report the availability of bikes at respective Stations

```
CREATE VIEW [Bike_Available]
AS SELECT DISTINCT B.StationID [StationID], S.Name,
       STUFF((SELECT DISTINCT ', ' +
              RTRIM(CAST(BikeID AS CHAR))
              FROM BikeStations BS
              WHERE BS.StationID = B.StationID
              AND BikeStatus = 'Available'
              FOR XML PATH('')) ,1, 1, '') AS 'Bike IDs'
FROM BikeStations B
INNER JOIN Stations S
ON S.StationID = B.StationID
```

	StationID	Name	Bike IDs
1	3	Pedal 1	11
2	4	pedal 2	4
3	5	pedal 3	4
4	6	pedal 4	6
5	7	pedal 5	4

Bike_Available

Views



VIEW to categorise Age Groups for Reporting

```
CREATE VIEW CalculateAgegroup
```

```
AS
```

```
SELECT UserID, Age,
CASE
  WHEN Age BETWEEN 14 AND 19 THEN '14-19'
  WHEN Age BETWEEN 20 AND 25 THEN '20-25'
  WHEN Age BETWEEN 26 AND 30 THEN '26-30'
  WHEN Age BETWEEN 31 AND 40 THEN '31-40'
END AS [Age group]
FROM Users
```

	UserID	Age	Age group
1	5	24	20-25
2	43	17	14-19
3	21	22	20-25
4	9	23	20-25
5	32	30	26-30

CalculateAgeGroup

View for Logging Trips on each Booking for Users

```
CREATE VIEW User_Booking_Details
```

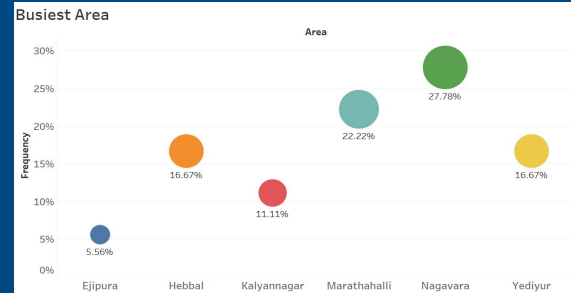
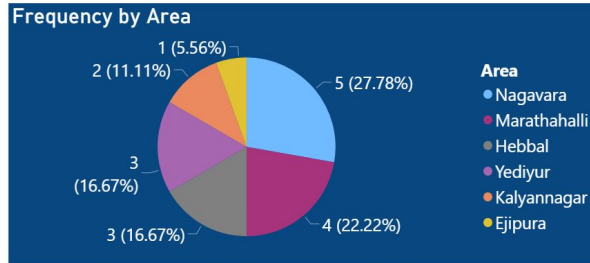
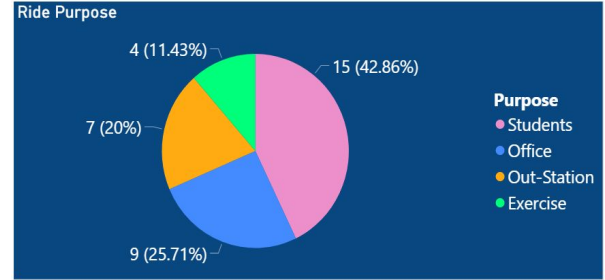
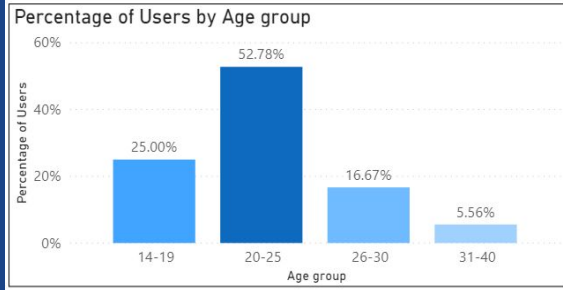
```
AS
```

```
SELECT U.UserID, U.Username, B.BookingID,
       B.purpose, B.ridedate, T.StartBikeStationID,
       T.EndBikeStationID
FROM Users U
      JOIN Bookings B
        ON U.UserID = B.UserID
      JOIN Trips T
        ON B.BookingID = T.BookingID
```

	userid	username	bookingid	purpose	ridedate	Startbikestationid	endbikestationid
1	4	Pavan22	1	Office	2020-06-01	1	2
2	4	Pavan22	2	Office	2020-06-01	2	1
3	5	abhi001	3	Exercise	2020-08-01	3	2
4	13	jb007	20	office	2020-08-07	4	3
5	13	jb007	21	office	2020-08-07	5	4

User_Booking_Details

Analytics





Thank you
Q/A