Metrics Engine For IST 659

Matthew A Beck
IST659 Project Deliverable

Part 1: Design

Project Narrative

My company is made up of a complex hierarchical structure with many business leaders overseeing one or many departments. Historically, success metrics and their associated visual representation have varied significantly from one area of the business to the other. The goal of my project is to link these metrics to the vital questions of the business in an extensible framework that can be easily accessed and updated by question askers and those responsible for answering. By simplifying both the collection and display of these metrics and their associated metadata, I can help leaders in the business focus on driving action instead of sifting through dense analytic reports.

Data Dictionary

After conducting interviews with several business leaders and studying available frameworks for metric collection, I chose to document the following entities

Entity	**Attribute**	**Properties**
User	User	Required. Name of Employee, user of the metrics engine
Role	Role	Required and Unique.
Business Question	Question	Required and Unique.
Metric	Metric	Required.
	Description	Explanation of what metric represents
	Direction	Which direction (positive or negative) does the business want the metric to go?
	Population	What groups are represented by this data
	Data Steward	Who is responsible for maintaining this metric
	Reference	Link to where more information on the metric can be found.
Metric Value	Value	Required. Recorded value of a metric
	Date Entered	Date value entered the system

	Period	Period of time from which a value was captured
Benchmark	Benchmark	Required. Business-determined value for an appropriate benchmark for a given metric
	Entry Date	Date benchmark was entered the system
	Valid From	Should benchmarks later be changed, a timestamp of when a given benchmark was first used
	Valid To	Should benchmarks later be changed, a timestamp of when a given benchmark ceased being used.
Forecast	Forecast	Forecasted value for a linked metric
	Period	Period of time for which a forecast was made
	Entry Date	Date forecast was entered the system
	Valid From	Should forecasts later be changed, a timestamp of when a given forecast was first used
	Valid To	Should forecasts later be changed, a timestamp of when a given forecast ceased being used.

Business Rules

A user has exactly one role
A role may be held by many users
A role belongs to only one supervisory organization at a time
A supervisory organization contains many roles
A role may ask one or many business questions
A business question may be asked by one or many roles
A business question may be answered by one or many metrics
A metric may answer one or many business questions
A metric may have one or many benchmarks (over time)
A benchmark may be used for exactly one metric
A metric may represent one or many metric values
A metric value may be used for exactly one metric
A metric may produce many forecasts
A forecast is produced for exactly one metric

The relevant stakeholders for this project are, for a given department:

Executive - Primary audience for business KPIs, reviews them with managers to gauge department performance
Manager - Relies on KPIs to measure their department's performance, reviews with executives, analysts, and staff.
Analyst - Helps identify trends and produces reporting for managers to use in conversations with executives.
Staff - Individuals who generate the actual business output measured by KPIs.

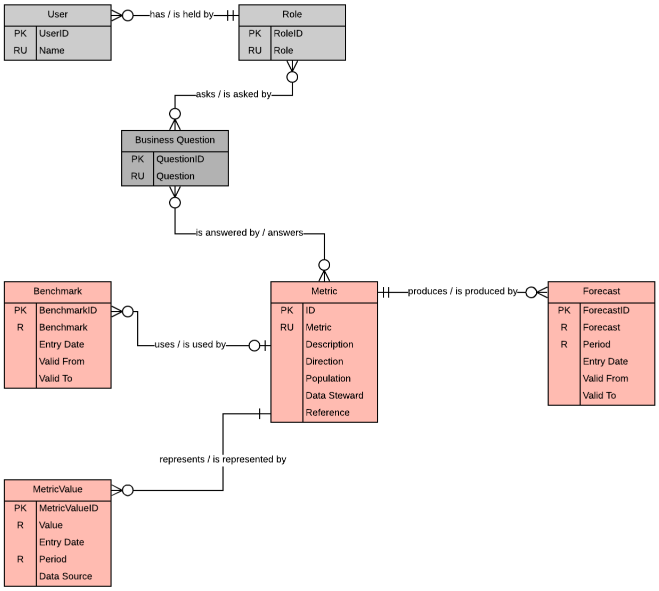
Data Questions

Some questions this database must be able to answer:

1. How many roles and users are linked to a given business question?
2. Which business questions does a particular role need to ask?
3. What is our current company performance, based on the questions we care about, and relative to our benchmarks?
4. For each of our metrics, how did our actuals compare with our forecasts?
5. Which employees (users) can answer a given business question?

Entity Relationship Diagram

When connected via relationships, the ERD is found on the following page. To manage the scope of this project, I have chosen to limit the database with the assumption that the database as presented will only contain data for one department. After the approach is piloted within the business, this framework can be expanded to accept a "department" entity.

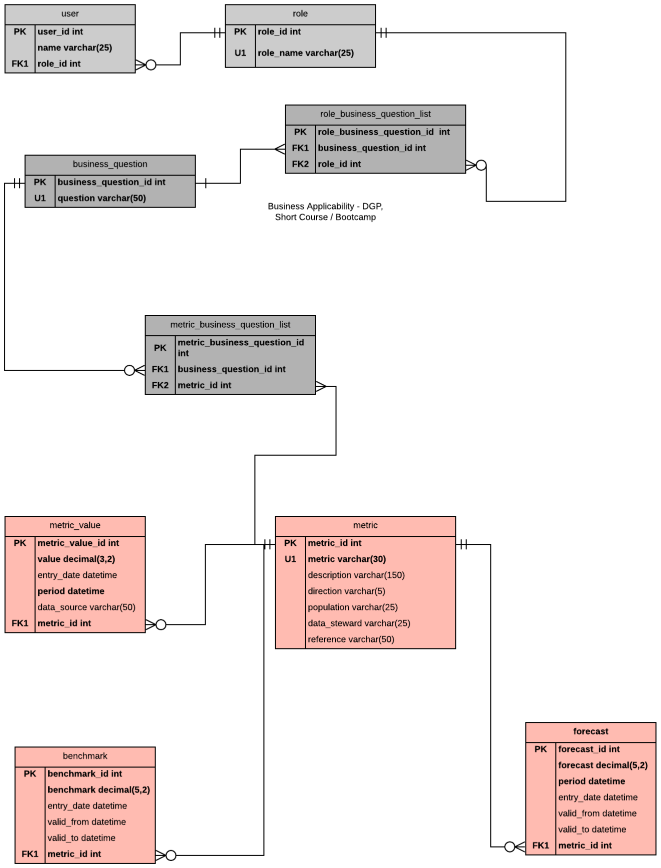


Logical Model Diagram

In order to convert this conceptual ERD to a logical model, I did the following:

1. Reframed the existing entities as tables, renamed fields to be more system-friendly and added surrogate keys where necessary.
2. Assigned data types to fields listed in the data dictionary
3. Mapped relationships
    a. Established primary and foreign keys, documented the relevant constraints
    b. Created associative tables for the relationship between roles and business questions, and between business questions and their respective metrics.

Following these steps produced the following diagram:



Normalized Model

The effort to create the logical model resulted in relations in third-normal form, so no further normalization was required.

Part 2: Implementation

SQL DDL

```
/*
        Author      : Matt Beck
        Course      : IST659 M400
        Term                    : October, 2019
        */

/*
        Drop the tables first, if they exist
        Dropped in reverse order of creation to avoid any problems with
        foreign key references
        Using DROP method
*/
DROP TABLE IF EXISTS forecast
DROP TABLE IF EXISTS metric_value
DROP TABLE IF EXISTS benchmark
DROP TABLE IF EXISTS metric_business_question_list
DROP TABLE IF EXISTS metric
DROP TABLE IF EXISTS role_business_question_list
DROP TABLE IF EXISTS business_question
DROP TABLE IF EXISTS [user]
DROP TABLE IF EXISTS [role]
DROP FUNCTION IF EXISTS bq_count_roles
DROP FUNCTION IF EXISTS bq_count_users
DROP FUNCTION IF EXISTS max_benchmark
DROP FUNCTION IF EXISTS derive_latest_value
DROP FUNCTION IF EXISTS next_forecast
DROP PROCEDURE IF EXISTS dbo.add_new_benchmark
DROP PROCEDURE IF EXISTS dbo.add_new_forecast
DROP PROCEDURE IF EXISTS dbo.add_new_mbq_mapping
DROP PROCEDURE IF EXISTS dbo.add_new_metric
DROP PROCEDURE IF EXISTS dbo.add_new_question
DROP PROCEDURE IF EXISTS dbo.add_new_rbq_mapping
DROP PROCEDURE IF EXISTS dbo.add_new_role
DROP PROCEDURE IF EXISTS dbo.add_new_user
DROP PROCEDURE IF EXISTS dbo.delete_mbq_mapping
DROP PROCEDURE IF EXISTS dbo.delete_rbq_mapping
DROP PROCEDURE IF EXISTS dbo.delete_user
DROP PROCEDURE IF EXISTS dbo.update_existing_benchmark_value
DROP PROCEDURE IF EXISTS dbo.update_existing_forecast_value
DROP PROCEDURE IF EXISTS dbo.update_metric_value
DROP PROCEDURE IF EXISTS dbo.update_question
DROP PROCEDURE IF EXISTS dbo.update_role_name
DROP PROCEDURE IF EXISTS dbo.update_user_name
DROP VIEW IF EXISTS bq_role_user_counts
DROP VIEW IF EXISTS business_questions_by_role
DROP VIEW IF EXISTS company_performance
DROP VIEW IF EXISTS forecast_actual_comparison
DROP VIEW IF EXISTS bq_employee_lookup


/*
```

```
Creating Tables for Database
*/

-- Creating the Role table
CREATE TABLE [role] (
	--Columns for the Role table
	role_id int identity,
	role_name varchar(25) not null,
	--Constraints on the role Table
	CONSTRAINT PK_role PRIMARY KEY (role_id),
	CONSTRAINT U1_role UNIQUE(role_name)
)
--End Creating the Role Table
GO

-- Creating the User table
CREATE TABLE [user] (
	--Columns for the User table
	user_id int identity,
	name varchar(25) not null,
  role_id int not null,
	--Constraints on the User Table
	CONSTRAINT PK_user PRIMARY KEY (user_id),
  CONSTRAINT FK1_user FOREIGN KEY (role_id) REFERENCES role(role_id)
)
--End Creating the User Table
GO
-- Creating the business_question table
CREATE TABLE business_question (
	--Columns for the Role table
	business_question_id int identity,
	question varchar(100) not null,
	--Constraints on the role Table
	CONSTRAINT PK_business_question PRIMARY KEY (business_question_id),
	CONSTRAINT U1_question UNIQUE(question)
)
--End Creating the business question Table
GO
-- Creating the role_business_question_list table
CREATE TABLE role_business_question_list (
	--Columns for the role_business_question_list table
	role_business_question_id int identity,
	business_question_id int not null,
  role_id int not null,
	--Constraints on the role_business_question_list Table
	CONSTRAINT PK_role_business_question_list PRIMARY KEY (role_business_question_id),
  CONSTRAINT FK1_role_business_question_list FOREIGN KEY (business_question_id) REFERENCES business_question(business_question_id),
  CONSTRAINT FK2_role_business_question_list FOREIGN KEY (role_id) REFERENCES role(role_id)
)
--End Creating the role_business_question_list table
GO
--Creating the metric table
CREATE TABLE metric (
  metric_id int identity,
  metric varchar(30) not null,
  description varchar(150),
  direction varchar(5),
  population varchar(25),
  data_steward varchar(25),
  reference varchar(50),
  CONSTRAINT PK_metric PRIMARY KEY (metric_id),
  CONSTRAINT U1_metric UNIQUE(metric)
);

--End Creating the metric Table
GO
-- Creating the metric_business_question_list table
CREATE TABLE metric_business_question_list (
	--Columns for the metric_business_question_list table
	metric_business_question_id int identity,
	business_question_id int not null,
  metric_id int not null,
	--Constraints on the metric_business_question_list Table
	CONSTRAINT PK_metric_business_question_list PRIMARY KEY (metric_business_question_id),
  CONSTRAINT FK1_metric_business_question_list FOREIGN KEY (business_question_id) REFERENCES business_question(business_question_id),
  CONSTRAINT FK2_metric_business_question_list FOREIGN KEY (metric_id) REFERENCES metric(metric_id)
)
--End Creating the role_business_question_list table
GO
--Creating the metric_value table
CREATE TABLE metric_value (
  --Columns for metric value table
  metric_value_id int identity,
  value decimal(3,2) not null,
  entry_date datetime default GetDate(),
  period datetime not null,
	metric_id int not null,
  --Constraints on the metric_value table
  CONSTRAINT PK_metric_value PRIMARY KEY (metric_value_id),
	CONSTRAINT FK1_metric_value FOREIGN KEY (metric_id) REFERENCES metric(metric_id)
)
--End Creating the metric_value table
GO
--Creating the benchmark table
CREATE TABLE benchmark (
  --Columns for benchmark table
  benchmark_id int identity,
  benchmark decimal(5,2) not null,
  entry_date datetime default GetDate(),
  valid_from datetime,
  valid_to datetime,
	metric_id int not null,
  --Constraints on the benchmark table
  CONSTRAINT PK_benchmark PRIMARY KEY (benchmark_id),
	CONSTRAINT FK1_benchmark FOREIGN KEY (metric_id) REFERENCES metric(metric_id)
)
--End Creating the benchmark table
GO
--Creating the forecast table
CREATE TABLE forecast (
  --Columns for forecast table
  forecast_id int identity,
  forecast decimal(5,2) not null,
	period datetime,
  entry_date datetime default GetDate(),
  valid_from datetime,
  valid_to datetime,
                metric_id int not null,
  --Constraints on the forecast table
  CONSTRAINT PK_forecast PRIMARY KEY (forecast_id),
                CONSTRAINT FK1_forecast FOREIGN KEY (metric_id) REFERENCES metric(metric_id)
)
--End Creating the forecast table
GO

--Functions

--Create function for counting the number of roles based on a business question (or abstract to allow the user to include table name)
CREATE FUNCTION bq_count_roles(@bqID int)
RETURNS int AS
BEGIN
    DECLARE @returnValue int --matches the function's return type

    /*
        Get the count of roles for the provided question and
        assign that value to @returnValue.
                                */
    SELECT @returnValue = COUNT(DISTINCT role_id) FROM role_business_question_list rbq
    WHERE rbq.business_question_id = @bqID

    --Return @returnValue to the calling code.
    RETURN @returnValue
END
GO

--Create function for counting # of Users based on business question provided
CREATE FUNCTION bq_count_users(@bqID int)
RETURNS int AS
BEGIN
    DECLARE @returnValue int --matches the function's return type

    /*
        Get the count of users for the provided question and
        assign that value to @returnValue.
                                */
    SELECT @returnValue = COUNT(DISTINCT u.user_id) FROM role_business_question_list rbq
                                JOIN [role] r ON rbq.role_id=r.role_id
                                JOIN [user] u ON u.role_id=r.role_id
                                WHERE rbq.business_question_id = @bqID

    --Return @returnValue to the calling code.
    RETURN @returnValue
END
GO
--Create function for pulling the most recent value for a given metric
CREATE FUNCTION derive_latest_value(@metric_id int)
RETURNS decimal(5,2) AS --COUNT is an integer, so shall it be returned
BEGIN
                DECLARE @returnValue decimal(5,2) --match function return type
                DECLARE @max_period datetime
    select @max_period = MAX(period) from metric_value WHERE metric_id = @metric_id;
```

```
        /*
        Pull the latest value from the metrics table, with the ID provided.
        */
        WITH CTE_latest_value
AS
(
        SELECT
                mv.*,
                ROW_NUMBER() OVER(PARTITION BY mv.metric_id ORDER BY IIF(mv.period >=@max_period,1,2) ASC) rnk
        FROM
                metric_value mv
)
 SELECT @returnValue = value FROM CTE_latest_value WHERE rnk = 1 and metric_id = @metric_id
        --Return @returnValue to the calling code.
        RETURN @returnValue
END
GO
--Create function for pulling the most recent benchmark for a given metric
CREATE FUNCTION max_benchmark(@metric_id int)
RETURNS decimal(5,2) AS --COUNT is an integer, so shall it be returned
BEGIN
        DECLARE @returnValue decimal(5,2) --match function return type

        /*
                Collect the maximum benchmark value, given the provided metric ID
                and assign that to the return value.
        */
        SELECT @returnValue = MAX(benchmark) FROM benchmark
        WHERE benchmark.metric_id = @metric_id and valid_to IS NULL

        --Return @returnValue to the calling code.
        RETURN @returnValue
END
GO
--Create function for pulling the predicted forecast for a given metric
CREATE FUNCTION next_forecast(@metric_id int)
RETURNS decimal(5,2) AS --COUNT is an integer, so shall it be returned
BEGIN
        DECLARE @returnValue decimal(5,2) --match function return type
        DECLARE @next_forecast_period datetime
   select @next_forecast_period = MIN(period) from forecast f
                WHERE metric_id = @metric_id and period > GETDATE();
        /*
                Collect the next upcoming forecast, given the provided metric ID
                and assign that to the return value.
        */
WITH temp_latest_value
AS
(
SELECT
        mv.*,
        ROW_NUMBER() OVER(PARTITION BY mv.metric_id ORDER BY IIF(mv.period >=@next_forecast_period,1,2) ASC) rnk
FROM
        metric_value mv
)
SELECT @returnValue = value FROM temp_latest_value WHERE rnk = 1 and metric_id = @metric_id
```

```
                --Return @returnValue to the calling code.
                RETURN @returnValue
END
GO
--Stored Procedures
/*
 Stored Procedure for updating a metric value
 Stored Procedure for updating a benchmark value
 Stored Procedure for updating a forecast
 Stored Procedure for updating a role name
 Stored Procedure for updating a user name
 Stored Procedure for updating a business question
 Stored Procedure for adding a new user
 Stored Procedure for adding a new role
 Stored Procedure for adding a new metric
 Stored Procedure for adding a new metric value
 Stored Procedure for adding a new benchmark
 Stored Procedure for adding a new forecast
 Stored Procedure for adding a role to question mapping
 Stored Procedure for adding a metric to question mapping
 Stored Procedure for deleting a user
 Stored Procedure for deleting a role to question mapping
 Stored Procedure for deleting a metric to question mapping
*/
--Create Stored Procedure for updating a metric value
CREATE PROCEDURE update_metric_value(@metric_id int, @period datetime, @value decimal(5,2))
AS
BEGIN
   UPDATE metric_value SET [value] = @value
   WHERE metric_id = @metric_id AND period = @period
END
GO

--Create Stored Procedure for updating an existing benchmark value, in the case of entry error.
CREATE PROCEDURE update_existing_benchmark_value(@benchmark_id int,@metric_id int, @period datetime, @benchmark decimal(5,2))
AS
BEGIN
   UPDATE benchmark SET benchmark = @benchmark
   WHERE metric_id = @metric_id AND benchmark_id = @benchmark_id
END
GO
--Create Stored Procedure for updating a forecast, in case of entry error.
CREATE PROCEDURE update_existing_forecast_value(@metric_id int, @period datetime, @forecast decimal(5,2))
AS
BEGIN
   UPDATE forecast SET forecast = @forecast
   WHERE metric_id = @metric_id AND [period] = @period
END
GO
--Create Stored Procedure for updating an existing role name
CREATE PROCEDURE update_role_name(@old_role varchar(25), @new_role varchar(25))
AS
BEGIN
   UPDATE [role] SET role_name = @new_role
   WHERE role_name = @old_role
END
GO
```

```
--Create Stored Procedure for updating an existing user name
CREATE PROCEDURE update_user_name(@old_name varchar(25), @new_name varchar(25))
AS
BEGIN
  UPDATE [user] SET name = @new_name
  WHERE name = @old_name
END
GO
--Create Stored Procedure for updating an existing business question
CREATE PROCEDURE update_question(@bq_id int, @new_question_lang varchar(25))
AS
BEGIN
  UPDATE business_question SET question = @new_question_lang
  WHERE business_question_id = @bq_id
END
GO
--Create Stored Procedure for adding a new forecast
CREATE PROCEDURE add_new_forecast(@metric_id int, @forecast decimal(5,2), @period datetime, @valid_from datetime = NULL ,@valid_to
datetime = NULL) AS
BEGIN
          --Update valid_to timestamp to ensure that the most recent forecast is used
          --in the table
                    UPDATE forecast SET valid_to = GetDate()
                    WHERE metric_id = @metric_id and valid_to IS NULL
  --Ensure valid_from timestamp is from today's date.
  SELECT @valid_from = GetDate()
  -- Now we can add the row using an INSERT Statement
  INSERT INTO forecast (metric_id,forecast,period,valid_from,valid_to)
  VALUES(@metric_id,@forecast,@period,@valid_from,@valid_to)

  --Now return the @@identity so the calling code knows where
  --the data ended up
  RETURN @@identity
END
GO
--Create Stored Procedure for adding a new benchmark
CREATE PROCEDURE add_new_benchmark(@metric_id int, @benchmark decimal(5,2), @valid_from datetime = NULL ,@valid_to datetime =
NULL) AS
BEGIN
          --Update valid_to timestamp to ensure that the most recent forecast is used
          --in the table
                    UPDATE benchmark SET valid_to = GetDate()
                    WHERE metric_id = @metric_id and valid_to IS NULL
  --Ensure valid_from timestamp is from today's date.
  SELECT @valid_from = GetDate()
  -- Now we can add the row using an INSERT Statement
  INSERT INTO benchmark (metric_id,benchmark,valid_from,valid_to)
  VALUES(@metric_id,@benchmark,@valid_from,@valid_to)

  --Now return the @@identity so the calling code knows where
  --the data ended up
  RETURN @@identity
END
GO
--Create Stored Procedure for adding a new user
CREATE PROCEDURE add_new_user(@name varchar(25), @role_name varchar(20)) AS
BEGIN
```

```
                        DECLARE @role_id INT
                        SELECT @role_id = role_id FROM [role] WHERE role_name=@role_name
   -- Now we can add the row using an INSERT Statement
   INSERT INTO [user] (name, role_id)
   VALUES(@name, @role_id)

   --Now return the @@identity so the calling code knows where
   --the data ended up
   RETURN @@identity
END
GO
--Create Stored Procedure for adding a new role
CREATE PROCEDURE add_new_role(@role_name varchar(25)) AS
BEGIN
   --Adding the row using an INSERT Statement
   INSERT INTO [role] (role_name)
   VALUES(@role_name)

   --Now return the @@identity so the calling code knows where
   --the data ended up
   RETURN @@identity
END
GO
--Create Stored Procedure for adding a new business question
CREATE PROCEDURE add_new_question(@question varchar(100)) AS
BEGIN
   --Adding the row using an INSERT Statement
   INSERT INTO business_question (question)
   VALUES(@question)

   --Now return the @@identity so the calling code knows where
   --the data ended up
   RETURN @@identity
END
GO
--Create Stored Procedure for adding a new metric
CREATE PROCEDURE add_new_metric(@metric varchar(30),@description varchar(150) = NULL,@direction varchar(5) = 'Up',
                              @population varchar(25) = '',@data_steward varchar(25)='Data Science',
                              @reference varchar(50) = "http://log.blog.docs") AS
BEGIN
   --Adding the row using an INSERT Statement
   INSERT INTO metric (metric,description, direction,population,data_steward,reference)
   VALUES(@metric,@description, @direction,@population,@data_steward,@reference)

   --Now return the @@identity so the calling code knows where
   --the data ended up
   RETURN @@identity
END
GO
--Create Stored Procedure for mapping roles to questions
CREATE PROCEDURE add_new_rbq_mapping(@role_id int, @bq_id int) AS
BEGIN
            --Adding the row using an INSERT Statement
   INSERT INTO role_business_question_list (role_id,business_question_id)
   VALUES(@role_id,@bq_id)

   --Now return the @@identity so the calling code knows where
   --the data ended up
   RETURN @@identity
END
GO

--Create Stored Procedure for mapping questions to metrics
CREATE PROCEDURE add_new_mbq_mapping(@metric_id int, @bq_id int) AS
BEGIN
               --Adding the row using an INSERT Statement
   INSERT INTO metric_business_question_list (metric_id,business_question_id)
   VALUES(@metric_id,@bq_id)

   --Now return the @@identity so the calling code knows where
   --the data ended up
   RETURN @@identity
END
GO
--Create Stored Procedure for deleting a user
CREATE PROCEDURE delete_user(@user_id int,@name varchar(25)) AS
BEGIN
   --Deleting the user using a DELETE Statement. Requiring both name and ID to
                              --ensure deletion is the desired action.
   DELETE FROM [user]
   WHERE name=@name and user_id = @user_id
END
GO
--Create Stored Procedure for deleting a role-to-question mapping
CREATE PROCEDURE delete_rbq_mapping(@role_id int,@bq_id int) AS
BEGIN
   --Deleting the user using a DELETE Statement. Requiring both ID's
                              --ensure deletion is the desired action on the right fields.
   DELETE FROM role_business_question_list
   WHERE role_id=@role_id and business_question_id = @bq_id
END
GO
--Create Stored Procedure for deleting a metric-to-question mapping
CREATE PROCEDURE delete_mbq_mapping(@metric_id int,@bq_id int) AS
BEGIN
   --Deleting the user using a DELETE Statement. Requiring both ID's
                              --ensure deletion is the desired action on the right fields.
   DELETE FROM metric_business_question_list
   WHERE metric_id=@metric_id and business_question_id = @bq_id
END
GO

--Creating Views
/*
View for Count of roles, users per Business Question
View for List of business questions asked by a given roles
View for averages of metrics
View for metric, value, benchmark, forecast
View for establishing available ranges for each metric
*/
--

--View 1 - How many roles and users are associated with the available business questions?
```

```
--Creating View for Counts of roles, users
CREATE VIEW bq_role_user_counts AS (
SELECT question
  , dbo.bq_count_roles(bq.business_question_id) as role_count
                    , dbo.bq_count_users(bq.business_question_id) as user_count
  FROM business_question bq
)
GO
--View 2 - What are the questions asked by a given business role?
--Creating View for List of Business Questions
CREATE VIEW business_questions_by_role AS (
          SELECT role_name
                    , question as questions_asked
                    FROM business_question bq
                    JOIN  role_business_question_list rbq ON rbq.business_question_id=bq.business_question_id
                    JOIN [role] r ON rbq.role_id=r.role_id
                    --WHERE r.role_name='Executive'
)
GO
--View 3 - What are the rolling averages for each metric, listed by their associated business question?
--Creating View 3
CREATE VIEW company_performance AS (
          SELECT question
    ,m.metric
                                        , AVG(value) as overall_value
                                        , dbo.max_benchmark(m.metric_id) as overall_benchmark
                                        , AVG(dbo.max_benchmark(m.metric_id) - value) as difference_from_benchmark
                                        , dbo.next_forecast(m.metric_id) as next_forecast
          FROM metric_value mv
          JOIN metric m ON m.metric_id=mv.metric_id
          JOIN metric_business_question_list mbq ON m.metric_id=mbq.metric_id
  JOIN business_question bq on bq.business_question_id=mbq.business_question_id
          GROUP BY question,metric,m.metric_id
)
GO
--View 4 - How did our actuals compare to our forecast?
--Creating View 4
CREATE VIEW forecast_actual_comparison AS (
          SELECT m.metric
                                        ,         AVG(f.forecast) as forecast_average
                                        , AVG(mv.value) actual_average
                                        ,AVG(mv.value - f.forecast)*100 as difference_from_forecast
          FROM metric m
          JOIN metric_value mv ON m.metric_id=mv.metric_id
          JOIN forecast f ON f.metric_id = m.metric_id AND mv.period = f.period
  GROUP BY metric
)
GO
--View 5 -Which employees are responsible for which business questions?
-- Creating View 5
CREATE VIEW bq_employee_lookup AS (
          SELECT
                    u.name
  ,bq.question
                    FROM business_question bq
                    JOIN role_business_question_list rbq ON rbq.business_question_id=bq.business_question_id
                    JOIN [role] r ON rbq.role_id=r.role_id
```

```
                        JOIN [user] u ON u.role_id=r.role_id
)
GO
```

SQL DML

INSERT Statements

```
--Insert into role table
INSERT INTO [role]
(role_name)
VALUES
('Executive'),
('Manager'),
('Analyst'),
('Staff')
--End role table insertion
GO
--Insert into user table
INSERT INTO [user]
(name,role_id)
VALUES
('Haleigh Musslewhite',1),
('Barbi Barbary',2),
('Ignace Veldens',3),
('Schuyler Beldum',4),
('Pauli Wherry',1),
('Caryl Jarrell',2),
('Brodie Auten',3),
('Alane Poveleye',4),
('Skylar Claw',1),
('Corny Immings',2),
('Aluin Rayer',4),
('Aluino Sheraton',2),
('Krystyna Crocetti',3),
('Ayn Jecks',4),
('Karrah Eisenberg',2),
('Cullie Gehrts',4),
('Arv Busby',4),
('Thorny Arp',4),
('Olly Savory',4),
('Clarie Prose',4)
--End role table insertion
GO
--Insert into business question table
INSERT INTO business_question
(question)
VALUES
('At what rate are we generating new leads?'),
('How is the sales funnel performing?'),
('How many visits does our website get?'),
('How are email campaigns performing?'),
('How do our sales forecasts compare to management targets?'),
('What is our total cost of acquisition?'),
('How satisfied are our clients?')
```

```
--End business question table insertion
GO
--Insert into role/business question list table
INSERT INTO role_business_question_list
(business_question_id,role_id)
VALUES
(1,1),(1,2),(1,3),(1,4),(2,1),(2,2),(2,3),(3,2),(3,3),(3,4),(4,2),(4,3),(4,4),(5,1),(5,3),(6,1),(6,3),(7,1),(7,3),(7,4)
--End role/business question list table insertion
GO
--Insert into metric table
INSERT INTO metric
(metric,description,direction,population,data_steward,reference)
VALUES
('Lead Generation Rate','Velocity of new leads into the sales funnel','Up','Leads','Sales Operations','https://referencepage.io'),
('Sales Funnel Performance','Rate of Sales made relative to previous year','Up','','Sales
Operations','https://bloglines.com/dui/proin/leo/odio.xml?partu'),
('Quarterly Website Visits','# of Website visits in a quarter','Up','Web Visitors','Web
Sales','http://oakley.com/felis/sed/interdum/venenatis.xml'),
('Website Visit Return Rate','# of users who return to the website','','Web Visitors','Web
Sales','https://purevolume.com/morbi.json?accumsan=neque&t'),
('Forecast-to-Target Ratio','Ratio of our forecasted performance relative to company targets','Down','','Sales
Analytics','https://storify.com/id/consequat/in/consequat/ut/n'),
('TCA Rate','Rate of Total Cost of acquiring a sale','Down','','Sales Analytics','http://amazonaws.com/ut/massa.jpg?semper=consectet'),
('Client Satisfaction','% of clients who report they are satisfied with their
product','Up','',NULL,'https://bluehost.com/elementum/nullam/varius/nulla'),
('Email Open Rate','% of sent emails that were opened','Up',NULL,'','https://parallels.com/posuere/cubilia/curae/donec/'),
('Email Click Rate','% of sent emails where user clicked the link','Up','Email List','',NULL)

--End metric table insertion
GO
--Insert into metric/business question list table
INSERT INTO metric_business_question_list
(business_question_id,metric_id)
VALUES
(1,1),
(2,2),
(3,3),
(3,4),
(4,8),
(4,9),
(5,5),
(6,6),
(7,7)
--End metric/business question list table insertion
GO
--Insert into metric value table
INSERT INTO metric_value
(value,period,metric_id)
VALUES
(0.51,'2019-01-01 05:00:00',1),
(0.99,'2019-02-01 05:00:00',1),
(0.57,'2019-03-01 05:00:00',1),
(0.51,'2019-04-01 05:00:00',1),
(0.69,'2019-05-01 05:00:00',1),
(0.75,'2019-06-01 05:00:00',1),
(0.14,'2019-07-01 05:00:00',1),
(0.93,'2019-08-01 05:00:00',1),
(0.71,'2019-09-01 05:00:00',1),
(0.55,'2019-01-01 05:00:00',2),
(0.96,'2019-02-01 05:00:00',2),
(0.67,'2019-03-01 05:00:00',2),
(0.78,'2019-04-01 05:00:00',2),
(0.62,'2019-05-01 05:00:00',2),
(0.33,'2019-06-01 05:00:00',2),
(0.64,'2019-07-01 05:00:00',2),
(0.26,'2019-08-01 05:00:00',2),
(0.14,'2019-09-01 05:00:00',2),
(0.46,'2019-01-01 05:00:00',3),
(0.93,'2019-02-01 05:00:00',3),
(0.87,'2019-03-01 05:00:00',3)
--(truncated)
--End metric value table insertion
GO
--Insert into benchmark table
INSERT INTO benchmark
(benchmark,valid_from,valid_to,metric_id)
VALUES
(0.6,'2019-01-01 05:00:00',NULL,1),
(0.8,'2019-01-01 05:00:00',NULL,2),
(0.5,'2019-01-01 05:00:00',NULL,3),
(0.4,'2019-01-01 05:00:00','2019-05-31 05:00:00',4),
(0.6,'2019-06-01 05:00:00',NULL,4),
(0.65,'2019-01-01 05:00:00',NULL,5),
(0.72,'2019-01-01 05:00:00',NULL,6),
(0.60,'2019-01-01 05:00:00','2019-06-30 05:00:00',7),
(0.70,'2019-01-01 05:00:00',NULL,7),
(0.75,'2019-01-01 05:00:00',NULL,8),
(0.38,'2019-01-01 05:00:00',NULL,9)
--End benchmark table insertion
GO
--Insert into forecast table
INSERT INTO forecast
(forecast,period,valid_from,valid_to,metric_id)
VALUES
(0.91,'2019-01-01 05:00:00','2019-01-01 05:00:00',NULL,1),
(0.12,'2019-04-01 05:00:00','2019-01-01 05:00:00',NULL,1),
(0.56,'2019-07-01 05:00:00','2019-01-01 05:00:00',NULL,1),
(0.96,'2019-10-01 05:00:00','2019-01-01 05:00:00',NULL,1),
(0.21,'2019-01-01 05:00:00','2019-01-01 05:00:00',NULL,2),
(0.72,'2019-04-01 05:00:00','2019-01-01 05:00:00',NULL,2),
(0.02,'2019-07-01 05:00:00','2019-01-01 05:00:00',NULL,2)
--(truncated )
--End benchmark table insertion
GO


--Demonstrate Stored Procedures at work:

--Updating Existing Metrics
EXEC update_metric_value 1, '2019-08-01 05:00:00', 0.63
--Adding a new forecast
DECLARE @new_forecast_id INT
EXEC @new_forecast_id = add_new_forecast 1, 0.85, '2020-01-01 05:00:00'
SELECT * FROM forecast WHERE forecast_id = @new_forecast_id
```

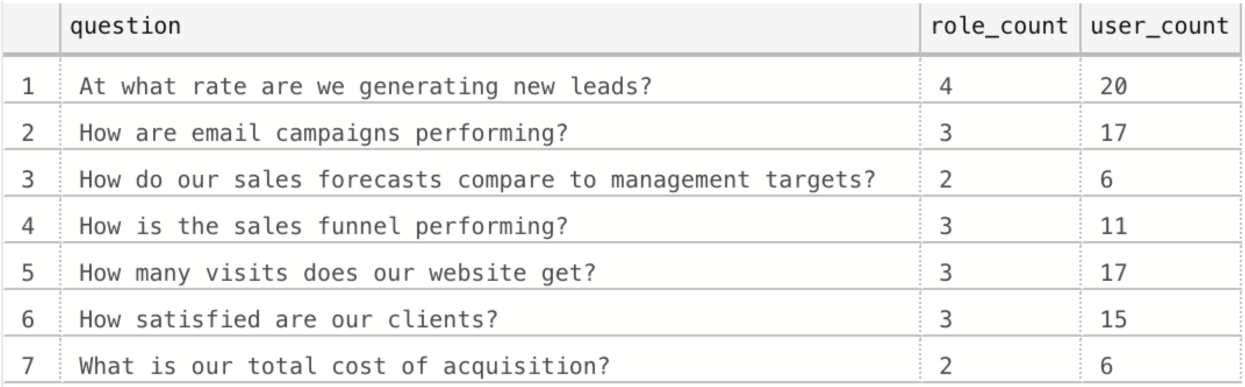
```
GO
--Adding a new user
DECLARE @new_user_id INT
EXEC @new_user_id = add_new_user 'Wendy Williams', 'Manager'
SELECT * FROM [user] WHERE user_id = @new_user_id
GO
--Adding a new role
DECLARE @new_role_id INT
EXEC @new_role_id = add_new_role 'Senior Staff'
SELECT * FROM [role] WHERE role_id = @new_role_id
GO
--Adding a new question
DECLARE @new_question_id INT
EXEC @new_question_id = add_new_question 'How is our employee productivity?'
SELECT * FROM business_question WHERE business_question_id = @new_question_id
GO
--Adding a new metric
DECLARE @new_metric_id INT
EXEC @new_metric_id = add_new_metric 'Employee Productivity Rate'
SELECT * FROM metric WHERE metric_id = @new_metric_id
GO
--Add new role-to-business-question mapping
DECLARE @new_mapping_id INT
EXEC @new_mapping_id = add_new_rbq_mapping 5,8
SELECT * FROM role_business_question_list WHERE role_business_question_id = @new_mapping_id
GO
--Add new metric-to-business-question mapping
DECLARE @new_mapping_id INT
EXEC @new_mapping_id = add_new_mbq_mapping 10, 8
SELECT * FROM metric_business_question_list WHERE metric_business_question_id = @new_mapping_id
GO
/*
Could perform these actions, but the intention is to use a GUI to update
benchmarks and forecasts.
--Add new forecasts for 2020
EXEC add_new_forecast VALUES
--Add new benchmark for the new employee productivity rate
EXEC add_new_benchmark VALUES
*/
--Deleting a user, where in this instance we've terminated Wendy's employment.
EXEC delete_user 21, 'Wendy Williams'

/*
Including these two queries to demonstrate that the mappings can be deleted at will.
--Deleting a metric-to-question mapping
EXEC delete_mbq_mapping 10, 8
--Deleting a role-to-question mapping
EXEC delete_rbq_mapping 5, 8
*/
/*
There are only a select few use cases for deletion in this database - most values
should only be updated to ensure the business can track what has been done
before. Historical data is *important*.
*/
```

Answers to Data Questions (SQL Select Statements)

--1. How many roles and users are linked to a given business question?

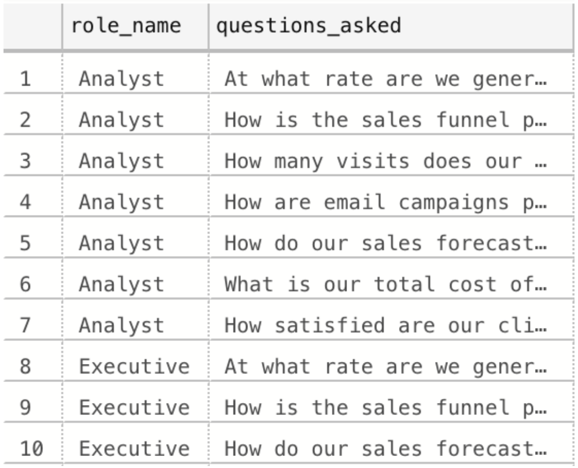
SELECT * FROM bq_role_user_counts



	question	role_count	user_count
1	At what rate are we generating new leads?	4	20
2	How are email campaigns performing?	3	17
3	How do our sales forecasts compare to management targets?	2	6
4	How is the sales funnel performing?	3	11
5	How many visits does our website get?	3	17
6	How satisfied are our clients?	3	15
7	What is our total cost of acquisition?	2	6

--2. Which business questions does a particular role need to ask?

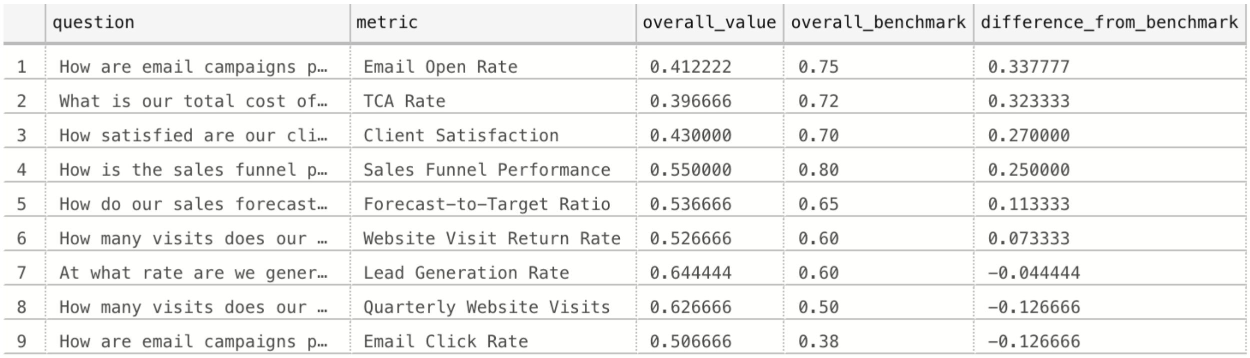
SELECT * FROM business_questions_by_role ORDER BY role_name;



	role_name	questions_asked
1	Analyst	At what rate are we gener…
2	Analyst	How is the sales funnel p…
3	Analyst	How many visits does our …
4	Analyst	How are email campaigns p…
5	Analyst	How do our sales forecast…
6	Analyst	What is our total cost of…
7	Analyst	How satisfied are our cli…
8	Executive	At what rate are we gener…
9	Executive	How is the sales funnel p…
10	Executive	How do our sales forecast…

3. What is our current company performance, based on the questions we care about, and relative to our benchmarks?

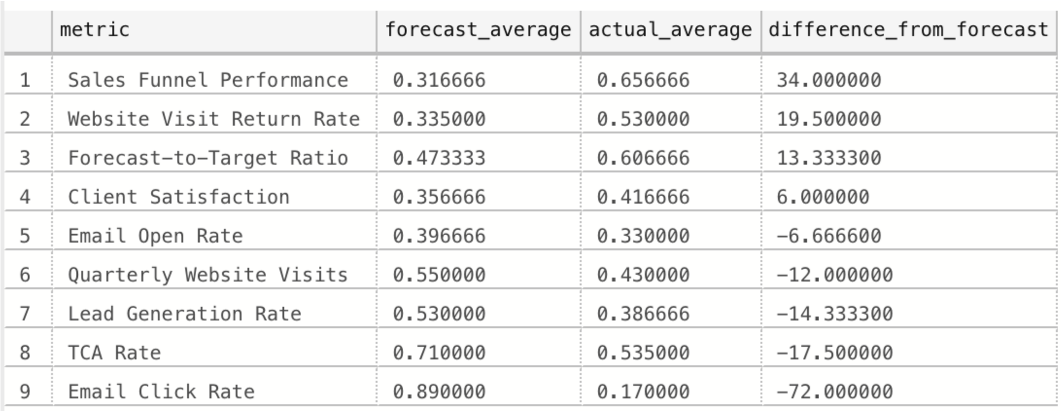
SELECT * FROM company_performance ORDER BY difference_from_benchmark DESC;



	question	metric	overall_value	overall_benchmark	difference_from_benchmark
1	How are email campaigns p…	Email Open Rate	0.412222	0.75	0.337777
2	What is our total cost of…	TCA Rate	0.396666	0.72	0.323333
3	How satisfied are our cli…	Client Satisfaction	0.430000	0.70	0.270000
4	How is the sales funnel p…	Sales Funnel Performance	0.550000	0.80	0.250000
5	How do our sales forecast…	Forecast-to-Target Ratio	0.536666	0.65	0.113333
6	How many visits does our …	Website Visit Return Rate	0.526666	0.60	0.073333
7	At what rate are we gener…	Lead Generation Rate	0.644444	0.60	−0.044444
8	How many visits does our …	Quarterly Website Visits	0.626666	0.50	−0.126666
9	How are email campaigns p…	Email Click Rate	0.506666	0.38	−0.126666

4. For each of our metrics, how did our actuals compare with our forecasts?

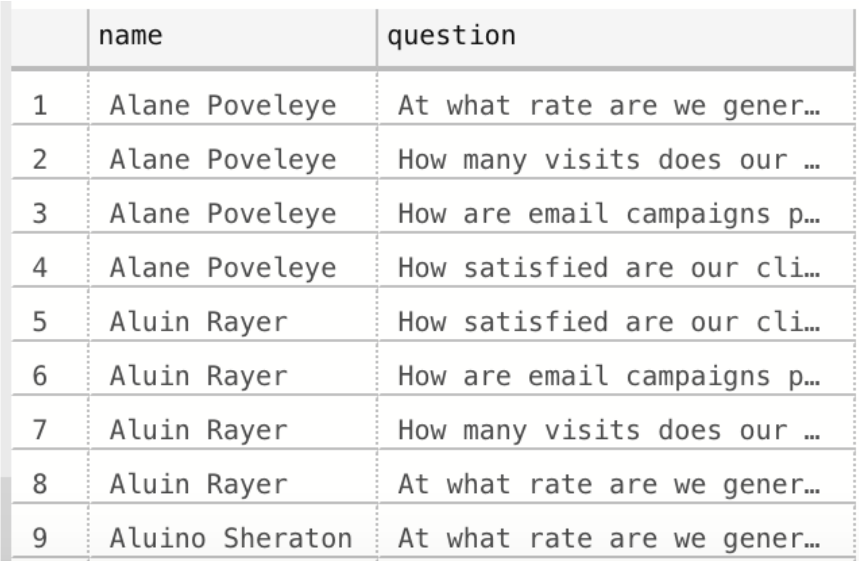
SELECT * FROM forecast_actual_comparison ORDER BY difference_from_forecast DESC;



	metric	forecast_average	actual_average	difference_from_forecast
1	Sales Funnel Performance	0.316666	0.656666	34.000000
2	Website Visit Return Rate	0.335000	0.530000	19.500000
3	Forecast-to-Target Ratio	0.473333	0.606666	13.333300
4	Client Satisfaction	0.356666	0.416666	6.000000
5	Email Open Rate	0.396666	0.330000	-6.666600
6	Quarterly Website Visits	0.550000	0.430000	-12.000000
7	Lead Generation Rate	0.530000	0.386666	-14.333300
8	TCA Rate	0.710000	0.535000	-17.500000
9	Email Click Rate	0.890000	0.170000	-72.000000

5. Which employees (users) can answer a given business question?

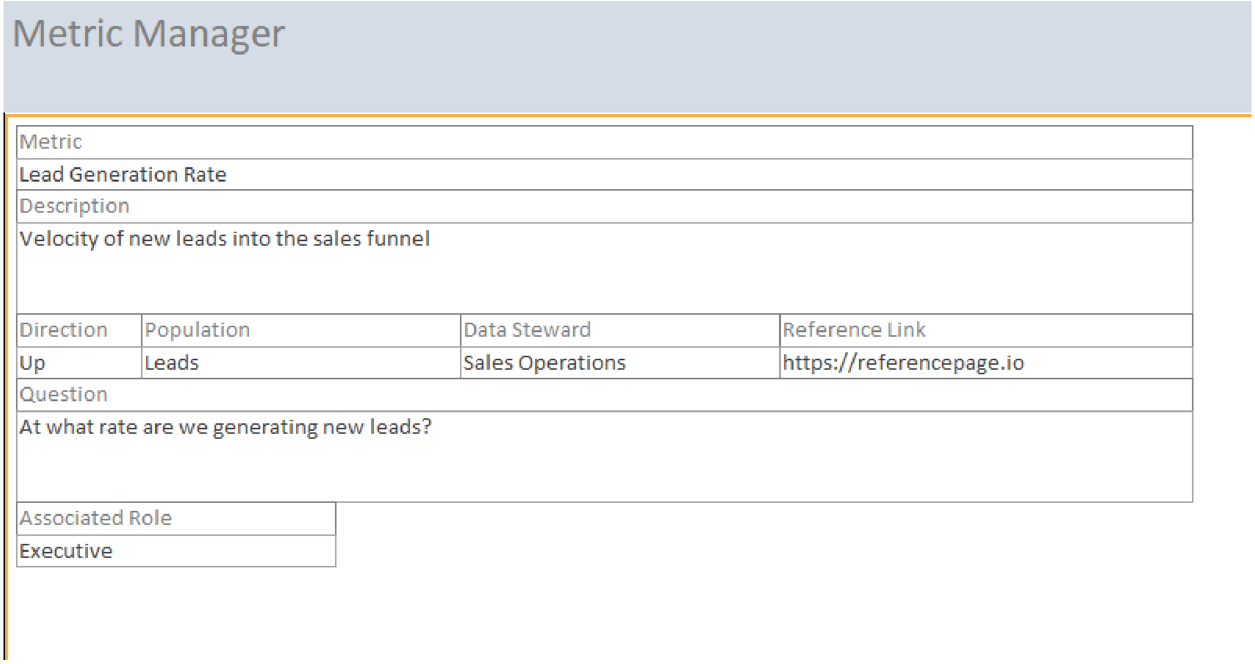
SELECT * FROM bq_employee_lookup ORDER BY name;



	name	question
1	Alane Poveleye	At what rate are we gener…
2	Alane Poveleye	How many visits does our …
3	Alane Poveleye	How are email campaigns p…
4	Alane Poveleye	How satisfied are our cli…
5	Aluin Rayer	How satisfied are our cli…
6	Aluin Rayer	How are email campaigns p…
7	Aluin Rayer	How many visits does our …
8	Aluin Rayer	At what rate are we gener…
9	Aluino Sheraton	At what rate are we gener…

GUI Prototype

Form 1 - The Metrics Manager. This view allows users to update their assigned metrics, while also having a filter so a user only receives the metrics for which they are responsible.



Metric Manager

Metric
Lead Generation Rate

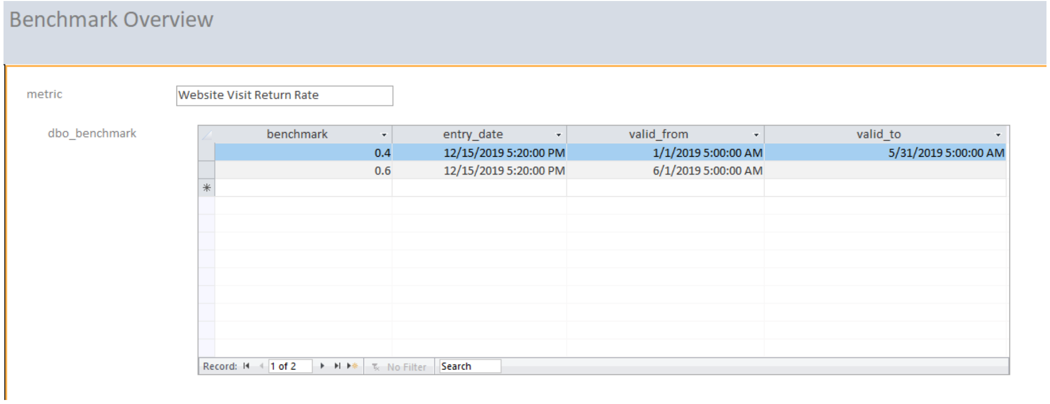
Description
Velocity of new leads into the sales funnel

Direction	Population	Data Steward	Reference Link
Up	Leads	Sales Operations	https://referencepage.io

Question
At what rate are we generating new leads?

Associated Role
Executive

Form 2 - Benchmark Overview. This allows users to update the database with new benchmarks while keeping the former benchmark data.



Benchmark Overview

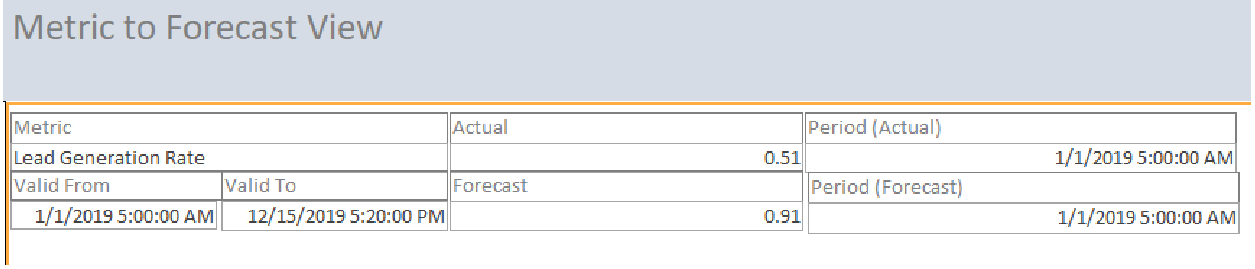
metric: Website Visit Return Rate

dbo_benchmark

benchmark	entry_date	valid_from	valid_to
0.4	12/15/2019 5:20:00 PM	1/1/2019 5:00:00 AM	5/31/2019 5:00:00 AM
0.6	12/15/2019 5:20:00 PM	6/1/2019 5:00:00 AM	

Form 3 - Metric to Forecast View

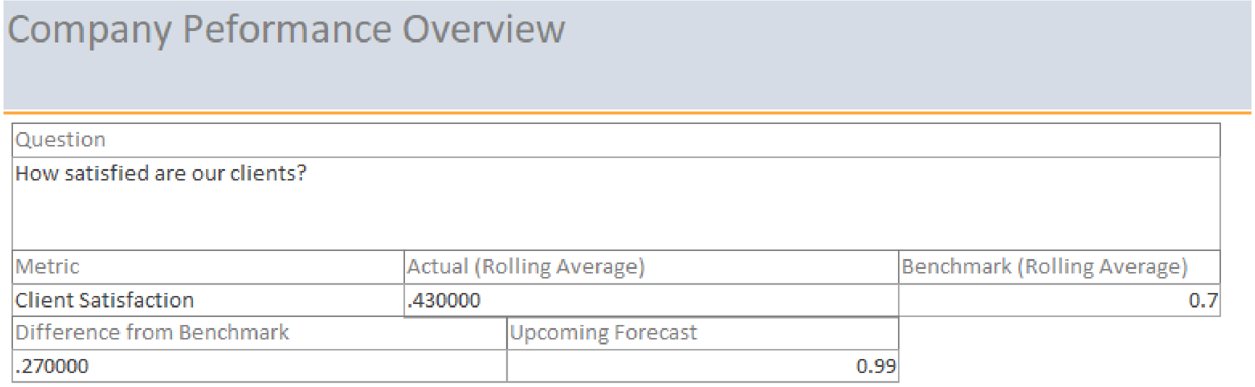
This view allows users to directly compare actual values with the forecasts created and also allows them to update accordingly.



Metric to Forecast View

Metric		Actual	Period (Actual)
Lead Generation Rate		0.51	1/1/2019 5:00:00 AM
Valid From	Valid To	Forecast	Period (Forecast)
1/1/2019 5:00:00 AM	12/15/2019 5:20:00 PM	0.91	1/1/2019 5:00:00 AM

Form 4 - Performance Overview - Provides a metric-level overview of the company's performance respective to the user's role and question.



Company Peformance Overview

Question		
How satisfied are our clients?		
Metric	Actual (Rolling Average)	Benchmark (Rolling Average)
Client Satisfaction	.430000	0.7
Difference from Benchmark	Upcoming Forecast	
.270000	0.99	

Reflection

I came into this project assuming that database creation and implementation was extremely complex and difficult to abstract and automate. What I've discovered is that an excellently laid-out design actually frees you substantially later on to allow your database to be both developer and user-friendly. In my personal work I've only ever worked with data warehouses like Amazon Redshift clusters. Knowing that I can set up something more small-scale and begin to solve problems all on my own at a low cost is really quite incredible.

The next time I'm asked to take build a database for a client with no/ill-designed data solutions, I will begin the project by thinking of what it would take to develop a solution that would allow me to hand it off with little-to-no onboarding required. I understand now that it's possible to build a strong framework that's easy to maintain, even with simple database design tools. The most important aspect is drafting an intelligent design that adheres to the principles we learned in this class.