Kelsey Johnson

IST 659

**Part 1: Design**

Project Description: 1

[Data](https://docs.google.com/document/d/1eYOD0k4RUN7ok-umpmdUqCal6Bc4sKuECRO9tG1gm4c/edit#heading=h.fzcmqc3941ar) Dictionary: 1-2

[Data Questions:    3](https://docs.google.com/document/d/1eYOD0k4RUN7ok-umpmdUqCal6Bc4sKuECRO9tG1gm4c/edit#heading=h.smbvkhqdxx11)

[Entity Relationship Diagram:    3](https://docs.google.com/document/d/1eYOD0k4RUN7ok-umpmdUqCal6Bc4sKuECRO9tG1gm4c/edit#heading=h.hw31dhgo5bjh)

[Logical Model Diagram:   4](https://docs.google.com/document/d/1eYOD0k4RUN7ok-umpmdUqCal6Bc4sKuECRO9tG1gm4c/edit#heading=h.jkexl3efnk2k)

Normalized Model:    4

**Project Description:**

My project is to create a database for Cass County Special Olympics of North Dakota in order to gain efficiencies related to storing data and using the data to maintain historic knowledge. By creating this database, the Area Director can quickly update, maintain, and delete data related to athletes, coaches, sports, and state games. This database will only be used for swimming (currently) but will build it knowing that in the future it may be used for additional sports.

Currently Special Olympics uses forms that are completed by athletes, coaches, staff, and directors. All these forms are then added into several spreadsheets that track the information related to each specific form. These spreadsheets are updated year to year, season to season, to maintain current information. This current process lacks the ability to see historic data when needed. This will also allow coaches a faster and more efficient way to identify athletes who are eligible for specific sporting events, evaluate performance over time and to contact previous coaches and other volunteers when more are needed.

**Data Dictionary:**

|  |  |  |  |
| --- | --- | --- | --- |
| Entity | Attributes | Data Dictionary | Description |
| Athlete |  |  | An athlete is someone who participates in the Special Olympic program and elects to participate in a sport(s). |
|  | Athlete name | Required | Composite of first and last name |
|  | Athlete Address | Required | Composite of street, city, state, zip |
|  | Staff Name | Required | Composite of first and last name |
|  | Medical Condition | Required | Varchar and Multivalue |
|  | Stroke | Required | Varchar and Multivalue |
|  | Stroke Time | Required | Datetime and Multivalue |
| Coach |  |  | A coach is someone who volunteers to coach the sport(s). |
|  | Coach Name | Required | Composite of first and last name |
|  | {Coach Address} | Optional | Composite of street, city, state, zip |
|  | Certified | Required | Varchar |
|  | Certification Date | Required | Datetime |
|  | Attended | Required | Datetime and Multivalue |
|  | Missed | Required | Datetime and Multivalue |
|  | Excused | Required | Datetime and Multivalue |
| Practice |  |  | A practice is a one-hour session held during the week, but not on a game day. |
|  | Attended | Required | Datetime and Multivalue |
|  | Missed | Required | Datetime and Multivalue |
|  | Excused | Required | Datetime and Multivalue |
|  | Building Name | Required | Varchar |
|  | Building Address | Required | Composite of street, city, state, zip |
|  | Practice Building Use Start Date | Required | Datetime |
|  | Practice Building Use End Date | Required | Datetime |
| State Games |  |  | State Games is a combination of sporting events that are held at specified periods throughout the year (one for each season). |
|  | Judges Name | Required | Composite of first and last name |
|  | {Judges Address} | Optional | Composite of street, city, state, zip |
|  | State Game Season | Required | Varchar |
|  | Start Date | Required | Datetime |
|  | End Date | Required | Datetime |
|  | Building Name | Required | Varchar |
|  | Building Address | Required | Composite of street, city, state, zip |
| Sport |  |  | A sport is the given athletic sport that is being played. |
|  | Sport Name | Required | Varchar |
|  | Sport Season | Required | Varchar |
|  | {Team Name} | Optional and Unique | Varchar |
|  | Start Date | Required | Datetime |
|  | End Date | Required | Datetime |

**Data Questions:**

The below questions are information that would typically be queried by coaches or the Area Director:

How many swimming athletes participated in the 2019 State Games?

Who were the top 10 fastest 25M freestylers and what are their times (fastest to slowest)?

Are all the coaches currently certified?

How many swim events are there and what are they?

Where are practices held?

**Entity Relationship Diagram:**

For the above relationships the model looks like the following.

A screenshot of text

Description automatically generated

**Logical Model Diagram Steps and Diagram:**

In order to convert our ERD into a Logical Model we need to follow these steps:

1. Begin by mapping the regular entities.
2. Breakdown the composite attributes into simple attributes.
3. Create two new relationships for each of the multivalue fields.
4. Indicate primary and unique constraints.
5. Add the foreign keys to each of the Many side of the newly created tables.
6. Breakdown each Many-to-Many relationship using associative tables.
7. Map the relationships.

A close up of text on a black background

Description automatically generated

**Normalized Model:**

At this point, my model is already in 3NF and no further normalization is required.

**Part 2: Implementation**

**SQL DDL**

/\*

SQL DDL Statements

Author: Kelsey Johnson

December 2019

IST659 Semester Project

\*/

/\*

To begin, start with DROP tables IF Exist and

do this in weak to strong order. Then

build tables in strong to weak order.

\*/

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'Stroke$')

BEGIN

DROP TABLE Stroke$

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'MedicalCondition$')

BEGIN

DROP TABLE MedicalCondition$

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'AttendedPractice$')

BEGIN

DROP TABLE AttendedPractice$

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'MissedPractice$')

BEGIN

DROP TABLE MissedPractice$

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'ExcusedPractice$')

BEGIN

DROP TABLE ExcusedPractice$

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'AthleteCoachList$')

BEGIN

DROP TABLE AthleteCoachList$

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'AthleteSportList$')

BEGIN

DROP TABLE AthleteSportList$

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'AthletePracticeList$')

BEGIN

DROP TABLE AthletePracticeList$

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'AthleteStateGamesList$')

BEGIN

DROP TABLE AthleteStateGamesList$

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'CoachPracticeList$')

BEGIN

DROP TABLE CoachPracticeList$

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'CoachStateGamesList$')

BEGIN

DROP TABLE CoachStateGamesList$

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'CoachSportList$')

BEGIN

DROP TABLE CoachSportList$

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'SportPracticeList$')

BEGIN

DROP TABLE SportPracticeList$

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'SportStateGamesList$')

BEGIN

DROP TABLE SportStateGamesList$

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'Sport$')

BEGIN

DROP TABLE Sport$

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'Practice$')

BEGIN

DROP TABLE Practice$

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'StateGames$')

BEGIN

DROP TABLE StateGames$

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'Coach$')

BEGIN

DROP TABLE Coach$

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'Athlete$')

BEGIN

DROP TABLE Athlete$

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'Stroke')

BEGIN

DROP TABLE Stroke

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'Medical\_Condition')

BEGIN

DROP TABLE Medical\_Condition

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'Attended\_Practice')

BEGIN

DROP TABLE Attended\_Practice

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'Missed\_Practice')

BEGIN

DROP TABLE Missed\_Practice

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'Excused\_Practice')

BEGIN

DROP TABLE Excused\_Practice

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'Athlete\_Coach\_List')

BEGIN

DROP TABLE Athlete\_Coach\_List

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'Athlete\_Sport\_List')

BEGIN

DROP TABLE Athlete\_Sport\_List

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'Athlete\_Practice\_List')

BEGIN

DROP TABLE Athlete\_Practice\_List

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'Athlete\_State\_Games\_List')

BEGIN

DROP TABLE Athlete\_State\_Games\_List

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'Coach\_Practice\_List')

BEGIN

DROP TABLE Coach\_Practice\_List

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'Coach\_State\_Games\_List')

BEGIN

DROP TABLE Coach\_State\_Games\_List

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'Coach\_Sport\_List')

BEGIN

DROP TABLE Coach\_Sport\_List

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'Sport\_Practice\_List')

BEGIN

DROP TABLE Sport\_Practice\_List

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'Sport\_State\_Games\_List')

BEGIN

DROP TABLE Sport\_State\_Games\_List

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'Sport')

BEGIN

DROP TABLE Sport

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'Practice')

BEGIN

DROP TABLE Practice

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'State\_Games')

BEGIN

DROP TABLE State\_Games

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'Coach')

BEGIN

DROP TABLE Coach

END

IF EXISTS (SELECT \* FROM Information\_Schema.tables WHERE Table\_Name = 'Athlete')

BEGIN

DROP TABLE Athlete

END

CREATE TABLE Athlete (

AthleteID int identity primary key,

AthleteFirstName varchar(30) NOT NULL,

AthleteMiddleInitial char(1) NOT NULL,

AthleteLastName varchar(30) NOT NULL,

AthleteStreet varchar(50) NOT NULL,

AthleteCity varchar(50) NOT NULL,

AthleteState varchar(40) NOT NULL,

AthleteZip varchar(11) NOT NULL,

StaffFirstName varchar(30) NOT NULL,

StaffLastName varchar(30) NOT NULL

)

CREATE TABLE Coach (

CoachID int identity primary key,

CoachFirstName varchar(30) NOT NULL,

CoachMiddleInitial char(1),

CoachLastName varchar(30) NOT NULL,

CoachStreet varchar(50),

CoachCity varchar(50),

CoachState varchar(40),

CoachZip varchar(11),

Certified varchar(5) NOT NULL,

CertificationDate Datetime

)

CREATE TABLE State\_Games (

StateGamesID int identity primary key,

JudgesFirstName varchar(30),

JudgesMiddleInitial char(1),

JudgesLastName varchar(30),

JudgesStreet varchar(50),

JudgesCity varchar(50),

JudgesState varchar(40),

JudgesZip varchar(11),

StateGamesSeason varchar(15) NOT NULL,

StateGamesStartDate Datetime NOT NULL,

StateGamesEndDate Datetime NOT NULL,

BuildingName varchar(50) NOT NULL,

BuildingStreet varchar(50) NOT NULL,

BuildingCity varchar(50) NOT NULL,

BuildingState varchar(40) NOT NULL,

BuildingZip varchar(11) NOT NULL

)

CREATE TABLE Practice (

PracticeID int identity primary key,

PracticeBuildingName varchar(50) NOT NULL,

PracticeBuildingStreet varchar(50) NOT NULL,

PracticeBuildingCity varchar(50) NOT NULL,

PracticeBuildingState varchar(40) NOT NULL,

PracticeBuildingZip varchar(11) NOT NULL,

PracticeBuildingUseStartDate datetime NOT NULL,

PracticeBuildingUseEndDate datetime NOT NULL

)

CREATE TABLE Sport (

SportID int identity primary key,

SportName varchar(30) NOT NULL,

SportSeason varchar(15) NOT NULL,

SportStartDate Datetime NOT NULL,

SportEndDate Datetime NOT NULL,

TeamName varchar(30)

)

CREATE TABLE Stroke (

StrokeID int identity primary key,

StrokeName varchar(20) NOT NULL,

StrokeTime dec (9,3) NOT NULL,

StrokeDate datetime NOT NULL,

AthleteStrokeNumberID int NOT NULL,

CONSTRAINT fk1\_AthleteStrokeNumber FOREIGN KEY (AthleteStrokeNumberID) REFERENCES Athlete(AthleteID)

)

CREATE TABLE Medical\_Condition (

MedicalConditionID int identity primary key,

MedicalConditionName varchar(100),

AthleteConditionNumberID int NOT NULL,

CONSTRAINT fk1\_AthleteConditionNumber FOREIGN KEY (AthleteConditionNumberID) REFERENCES Athlete(AthleteID)

)

CREATE TABLE Attended\_Practice (

AttendedPracticeID int identity primary key,

AttendedPracticeDate Datetime NOT NULL,

PracticeAttendedNumberID int NOT NULL,

PracticeAttendedBuildingNumberID int NOT NULL,

CONSTRAINT fk1\_PracticeAttendedNumber FOREIGN KEY (PracticeAttendedNumberID) REFERENCES Athlete(AthleteID),

CONSTRAINT fk2\_PracticeAttendedBuildingNumber FOREIGN KEY (PracticeAttendedBuildingNumberID) REFERENCES Practice(PracticeID)

)

CREATE TABLE Missed\_Practice (

MissedPracticeID int identity primary key,

MissedPracticeDate Datetime NOT NULL,

PracticeMissedNumberID int NOT NULL,

PracticeMissedBuildingNumberID int NOT NULL,

CONSTRAINT fk1\_PracticeMissedNumber FOREIGN KEY (PracticeMissedNumberID) REFERENCES Athlete(AthleteID),

CONSTRAINT fk2\_PracticeMissedBuildingNumber FOREIGN KEY (PracticeMissedBuildingNumberID) REFERENCES Practice(PracticeID)

)

CREATE TABLE Excused\_Practice (

ExcusedPracticeID int identity primary key,

ExcusedPracticeDate Datetime NOT NULL,

PracticeExcusedNumberID int NOT NULL,

PracticeExcusedBuildingNumberID int NOT NULL,

CONSTRAINT fk1\_PracticeExcusedNumber FOREIGN KEY (PracticeExcusedNumberID) REFERENCES Athlete(AthleteID),

CONSTRAINT fk2\_PracticeExcusedBuildingNumber FOREIGN KEY (PracticeExcusedBuildingNumberID) REFERENCES Practice(PracticeID)

)

CREATE TABLE Athlete\_Coach\_List (

AthleteCoachListID int identity primary key,

AthleteID int NOT NULL FOREIGN KEY REFERENCES Athlete(AthleteID),

CoachID int NOT NULL FOREIGN KEY REFERENCES Coach(CoachID)

)

CREATE TABLE Athlete\_Sport\_List (

AthleteSportListID int identity primary key,

AthleteID int NOT NULL FOREIGN KEY REFERENCES Athlete(AthleteID),

SportID int NOT NULL FOREIGN KEY REFERENCES Sport(SportID)

)

CREATE TABLE Athlete\_Practice\_List (

AthletePracticeListID int identity primary key,

AthleteID int NOT NULL FOREIGN KEY REFERENCES Athlete(AthleteID),

PracticeID int NOT NULL FOREIGN KEY REFERENCES Practice(PracticeID)

)

CREATE TABLE Athlete\_State\_Games\_List (

AthleteStateGamesListID int identity primary key,

AthleteID int NOT NULL FOREIGN KEY REFERENCES Athlete(AthleteID),

StateGamesID int NOT NULL FOREIGN KEY REFERENCES State\_Games(StateGamesID)

)

CREATE TABLE Coach\_Practice\_List (

CoachPracticeListID int identity primary key,

CoachID int NOT NULL FOREIGN KEY REFERENCES Coach(CoachID),

PracticeID int NOT NULL FOREIGN KEY REFERENCES Practice(PracticeID)

)

CREATE TABLE Coach\_State\_Games\_List (

CoachStateGamesListID int identity primary key,

CoachID int NOT NULL FOREIGN KEY REFERENCES Coach(CoachID),

StateGamesID int NOT NULL FOREIGN KEY REFERENCES State\_Games(StateGamesID)

)

CREATE TABLE Coach\_Sport\_List (

CoachSportListID int identity primary key,

CoachID int NOT NULL FOREIGN KEY REFERENCES Coach(CoachID),

SportID int NOT NULL FOREIGN KEY REFERENCES Sport(SportID)

)

CREATE TABLE Sport\_Practice\_List (

SportPracticeListID int identity primary key,

SportID int NOT NULL FOREIGN KEY REFERENCES Sport(SportID),

PracticeID int NOT NULL FOREIGN KEY REFERENCES Practice(PracticeID)

)

CREATE TABLE Sport\_State\_Games\_List (

SportStateGamesListID int identity primary key,

SportID int NOT NULL FOREIGN KEY REFERENCES Sport(SportID),

StateGamesID int NOT NULL FOREIGN KEY REFERENCES State\_Games(StateGamesID)

)

**SQL DML – Insert, Update, and Delete Statement Examples**

Below are a few insert, update, and delete statements that could be used to manually enter or correct data. For my project I entered my data using the Import Data function in SQL. All the statements I used are listed below.

--Insert into Athlete Table

INSERT INTO Athlete([AthleteFirstName], [AthleteMiddleInitial],

[AthleteLastName], [AthleteStreet], [AthleteCity], [AthleteState],

[AthleteZip], [StaffFirstName], [StaffLastName])

VALUES('Tanya', 'M', 'Meyers', '801 First Street N', 'West Fargo', 'North Dakota', '58078', 'Kendra', 'Lowe');

--Insert into Coach Table

INSERT INTO Coach([CoachFirstName], [CoachLastName], [CoachStreet],

[CoachCity], [CoachState], [CoachZip], [Certified], [CertificationDate])

VALUES ('Missy', 'Hagen', '1038 45th Ave E', 'West Fargo', 'North Dakota', '58078', 'Yes', '5/1/2019')

--Correcting AthleteFirstName

UPDATE Athlete SET AthleteFirstName = 'Tania' WHERE AthleteID = 110

--Deleting Athlete

DELETE Athlete WHERE AthleteID = 110

**SQL DML - Imported Data from Excel:**

SET Identity\_Insert Athlete ON

INSERT INTO Athlete (AthleteID, AthleteFirstName, AthleteMiddleInitial,

AthleteLastName, AthleteStreet, AthleteCity, AthleteState, AthleteZip,

StaffFirstName, StaffLastName)

SELECT AthleteID, AthleteFirstName, AthleteMiddleInitial, AthleteLastName,

AthleteStreet, AthleteCity, AthleteState, AthleteZip, StaffFirstName,

StaffLastName

FROM Athlete$

SET Identity\_Insert Athlete OFF

SET Identity\_Insert Coach ON

INSERT INTO Coach (CoachID, CoachFirstName, CoachMiddleInitial, CoachLastName,

CoachStreet, CoachCity, CoachState, CoachZip, Certified, CertificationDate)

SELECT CoachID, CoachFirstName, CoachMiddleInitial, CoachLastName,

CoachStreet, CoachCity, CoachState, CoachZip, Certified, CertificationDate

FROM Coach$

SET Identity\_Insert Coach OFF

SET Identity\_Insert State\_Games ON

INSERT INTO State\_Games (StateGamesID, JudgesFirstName, JudgesMiddleInitial,

JudgesLastName, JudgesStreet, JudgesCity, JudgesState, JudgesZip,

StateGamesSeason, StateGamesStartDate, StateGamesEndDate, BuildingName,

BuildingStreet, BuildingCity, BuildingState, BuildingZip)

SELECT StateGamesID, JudgesFirstName, JudgesMiddleInitial,

JudgesLastName, JudgesStreet, JudgesCity, JudgesState, JudgesZip,

StateGamesSeason, StateGamesStartDate, StateGamesEndDate, BuildingName,

BuildingStreet, BuildingCity, BuildingState, BuildingZip

FROM StateGames$

SET Identity\_Insert State\_Games OFF

SET Identity\_Insert Practice ON

INSERT INTO Practice (PracticeID, PracticeBuildingName, PracticeBuildingStreet,

PracticeBuildingCity, PracticeBuildingState, PracticeBuildingZip,

PracticeBuildingUseStartDate, PracticeBuildingUseEndDate)

SELECT PracticeID, PracticeBuildingName, PracticeBuildingStreet,

PracticeBuildingCity, PracticeBuildingState, PracticeBuildingZip,

PracticeBuildingUseStartDate, PracticeBuildingUseEndDate

FROM Practice$

SET Identity\_Insert Practice OFF

SET Identity\_Insert Sport ON

INSERT INTO Sport (SportID, SportName, SportSeason, SportStartDate,

SportEndDate, TeamName)

SELECT SportID, SportName, SportSeason, SportStartDate,

SportEndDate, TeamName

FROM Sport$

SET Identity\_Insert Sport OFF

SET Identity\_Insert Stroke ON

INSERT INTO Stroke (StrokeID, StrokeName, StrokeTime, StrokeDate,

AthleteStrokeNumberID)

SELECT StrokeID, StrokeName, StrokeTime, StrokeDate,

AthleteStrokeNumberID

FROM Stroke$

SET Identity\_Insert Stroke OFF

SET Identity\_Insert Medical\_Condition ON

INSERT INTO Medical\_Condition (MedicalConditionID, MedicalConditionName,

AthleteConditionNumberID)

SELECT MedicalConditionID, MedicalConditionName, AthleteConditionNumberID

From MedicalCondition$

SET Identity\_Insert Medical\_Condition OFF

SET Identity\_Insert Attended\_Practice ON

INSERT INTO Attended\_Practice (AttendedPracticeID, AttendedPracticeDate,

PracticeAttendedNumberID, PracticeAttendedBuildingNumberID)

SELECT AttendedPracticeID, AttendedPracticeDate,

PracticeAttendedNumberID, PracticeAttendedBuildingNumberID

FROM AttendedPractice$

SET Identity\_Insert Attended\_Practice OFF

SET Identity\_Insert Missed\_Practice ON

Insert INTO Missed\_Practice (MissedPracticeID, MissedPracticeDate,

PracticeMissedNumberID, PracticeMissedBuildingNumberID)

SELECT MissedPracticeID, MissedPracticeDate,

PracticeMissedNumberID, PracticeMissedBuildingNumberID

FROM MissedPractice$

SET Identity\_Insert Missed\_Practice OFF

SET Identity\_Insert Excused\_Practice ON

INSERT INTO Excused\_Practice (ExcusedPracticeID, ExcusedPracticeDate,

PracticeExcusedNumberID, PracticeExcusedBuildingNumberID)

SELECT ExcusedPracticeID, ExcusedPracticeDate,

PracticeExcusedNumberID, PracticeExcusedBuildingNumberID

FROM ExcusedPractice$

SET Identity\_Insert Excused\_Practice OFF

SET Identity\_Insert Athlete\_Coach\_List ON

INSERT INTO Athlete\_Coach\_List (AthleteCoachListID, AthleteID,

CoachID)

SELECT AthleteCoachListID, AthleteID, CoachID

FROM AthleteCoachList$

SET Identity\_Insert Athlete\_Coach\_List OFF

SET Identity\_Insert Athlete\_Sport\_List ON

INSERT INTO Athlete\_Sport\_List (AthleteSportListID, AthleteID,

SportID)

SELECT AthleteSportListID, AthleteID, SportID

FROM AthleteSportList$

SET Identity\_Insert Athlete\_Sport\_List OFF

SET Identity\_Insert Athlete\_Practice\_List ON

INSERT INTO Athlete\_Practice\_List (AthletePracticeListID,

AthleteID, PracticeID)

SELECT AthletePracticeListID, AthleteID, PracticeID

FROM AthletePracticeList$

SET Identity\_Insert Athlete\_Practice\_List OFF

SET Identity\_Insert Athlete\_State\_Games\_List ON

INSERT INTO Athlete\_State\_Games\_List (AthleteStateGamesListID,

AthleteID, StateGamesID)

SELECT AthleteStateGamesListID, AthleteID, StateGamesID

FROM AthleteStateGamesList$

SET Identity\_Insert Athlete\_State\_Games\_List OFF

SET Identity\_Insert Coach\_Practice\_List ON

INSERT INTO Coach\_Practice\_List (CoachPracticeListID,

CoachID, PracticeID)

SELECT CoachPracticeListID, CoachID, PracticeID

FROM CoachPracticeList$

SET Identity\_Insert Coach\_Practice\_List OFF

SET Identity\_Insert Coach\_State\_Games\_List ON

INSERT INTO Coach\_State\_Games\_List (CoachStateGamesListID,

CoachID, StateGamesID)

SELECT CoachStateGamesListID, CoachID, StateGamesID

FROM CoachStateGamesList$

SET Identity\_Insert Coach\_State\_Games\_List OFF

SET Identity\_Insert Coach\_Sport\_List ON

INSERT INTO Coach\_Sport\_List (CoachSportListID,

CoachID, SportID)

SELECT CoachSportListID, CoachID, SportID

FROM CoachSportList$

SET Identity\_Insert Coach\_Sport\_List OFF

SET Identity\_Insert Sport\_Practice\_List ON

INSERT INTO Sport\_Practice\_List (SportPracticeListID,

SportID, PracticeID)

SELECT SportPracticeListID, SportID, PracticeID

FROM SportPracticeList$

SET Identity\_Insert Sport\_Practice\_List OFF

SET Identity\_Insert Sport\_State\_Games\_List ON

INSERT INTO Sport\_State\_Games\_List (SportStateGamesListID,

SportID, StateGamesID)

SELECT SportStateGamesListID, SportID, StateGamesID

FROM SportStateGamesList$

SET Identity\_Insert Sport\_State\_Games\_List OFF

**Answers to Data Questions**

The below questions are information that would typically be queried by coaches or the Area Director:

How many swimming athletes participated in the 2019 State Games?

SELECT

COUNT(AthleteID) as NumberOfAthletes,

StateGamesStartDate,

StateGamesEndDate

FROM Athlete\_State\_Games\_List

JOIN State\_Games on State\_Games.StateGamesID = Athlete\_State\_Games\_List.StateGamesID

WHERE State\_Games.StateGamesID = 1

GROUP BY StateGamesStartDate, StateGamesEndDate



Who were the top 10 fastest 25M freestylers and what are their times (fastest to slowest)?

SELECT TOP 10

AthleteFirstName, AthleteLastName,

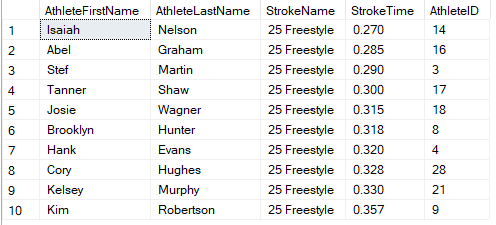
StrokeName, StrokeTime, AthleteID

FROM Stroke

JOIN Athlete on Athlete.AthleteID = Stroke.AthleteStrokeNumberID

WHERE StrokeName = '25 Freestyle'

ORDER BY StrokeTime ASC



Which coaches who attended State Games in 2019 were certified?

SELECT \*

FROM State\_Games

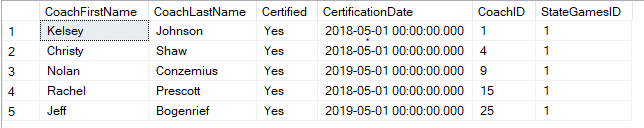
SELECT DISTINCT CoachFirstName, CoachLastName, Certified, CertificationDate,

Coach.CoachID, StateGamesID

FROM Coach\_State\_Games\_List

JOIN Coach on Coach.CoachID = Coach\_State\_Games\_List.CoachID

WHERE StateGamesID = 1



Which coaches attended Practices in 2019?

SELECT \*

FROM Practice

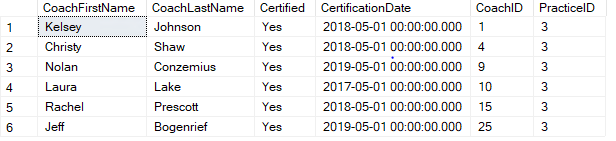
SELECT DISTINCT CoachFirstName, CoachLastName, Certified, CertificationDate,

Coach.CoachID, PracticeID

FROM Coach\_Practice\_List

Join Coach on Coach.CoachID = Coach\_Practice\_List.CoachID

WHERE PracticeID = 3

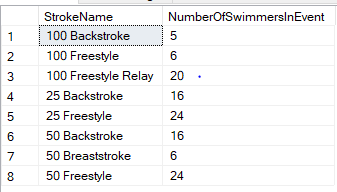


How many athletes participated in each swim event and what are the names of those events?

SELECT StrokeName, COUNT(StrokeName) as NumberOfSwimmersInEvent

FROM Stroke

GROUP BY StrokeName



Which athletes attended practice at Fargo North High School?

SELECT AthleteFirstName, AthleteLastName, Athlete\_Practice\_List.PracticeID,

PracticeBuildingName, PracticeBuildingStreet,

PracticeBuildingCity, PracticeBuildingState, PracticeBuildingZip

FROM Athlete\_Practice\_List

Join Athlete ON Athlete.AthleteID = Athlete\_Practice\_List.AthleteID

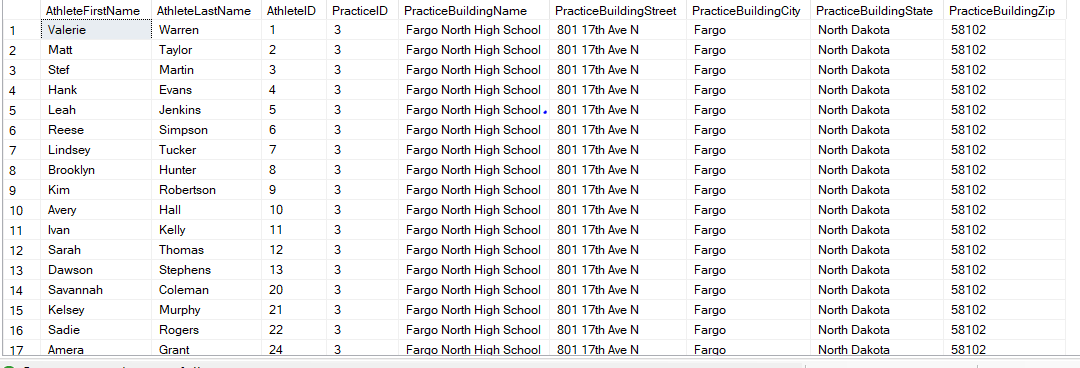
JOIN Practice ON Practice.PracticeID = Athlete\_Practice\_List.PracticeID

WHERE Athlete\_Practice\_List.PracticeID = 3

GROUP BY AthleteFirstName, AthleteLastName, Athlete\_Practice\_List.PracticeID,

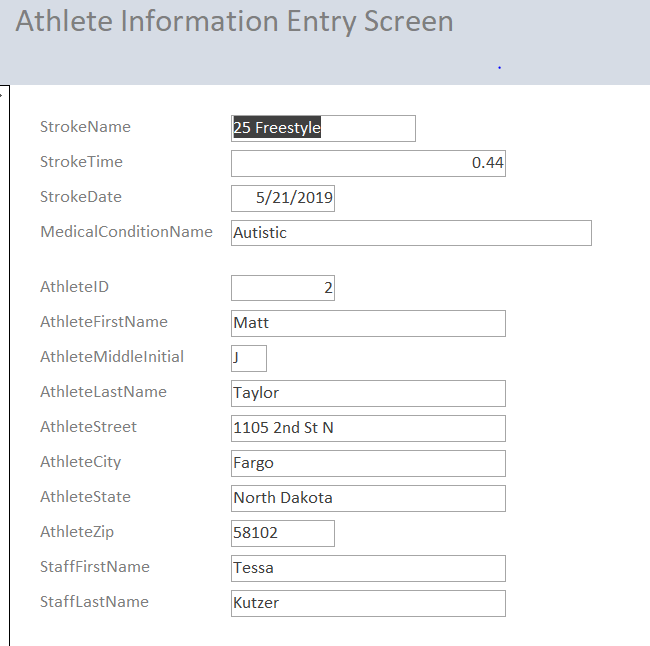
PracticeBuildingName, PracticeBuildingStreet,

PracticeBuildingCity, PracticeBuildingState, PracticeBuildingZip



**User Interface Examples:**

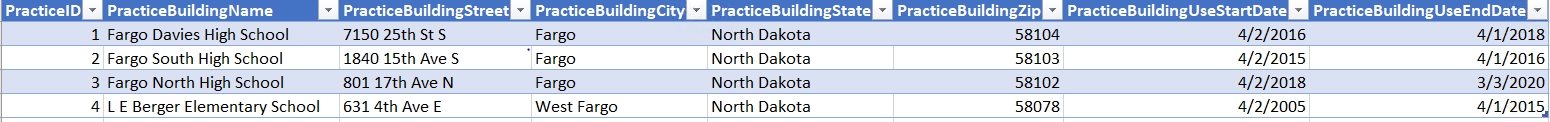
Athlete Information Entry Screen – Access Form



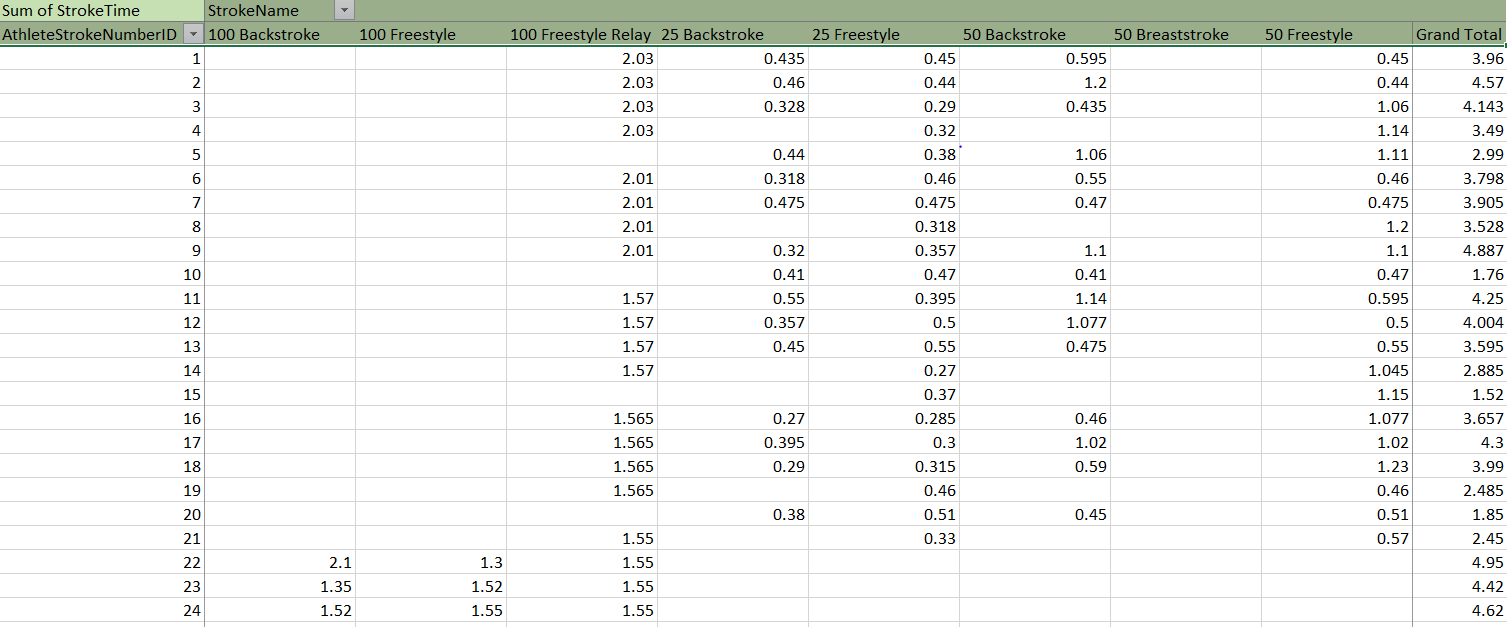
Athlete Stroke Times – Access Report



Practice Listing using Excel:



Excel Pivot Table of Athlete Stroke Times per Event for each Athlete:



**Reflection:**

One assumption I had was to use DateTIme for my stroke\_time field on the Table Stroke. I found out very quickly after loading my data that it did not allow me to see the data correctly. In order to correct this, I changed it to dec. I also ended up changing a few other fields. Seeing my data in my tables allowed me to have a better understanding of what I modeled and how I meant for it to come together.

Next time I do this I feel that it would be beneficial to get my tables created as soon as possible and to think of more queries/questions that will need to be answered to ensure I create my data appropriately.

I think this will inform my approach to data by allowing me to better understand why fields are created the way they are (int, dec, etc). In working with data at my current job, sometimes I question the definitions in the data dictionary and why data is created the way it is. This will also help with my understanding of how and when to use joins to get the data I need instead of several select statements.