Database Design and Implementation for “Toronto Hub”

Prepared By:

Group No:7 ( Misha Sondhi, Ebenezer , Yeonmi Park, Binod Shrestha)

Submitted to:

Professor Megha Patel

Course Code: DBAS27198

Sheridan College, Applied Science and Technology

Davis Campus

Due Date: 2018 December 10

Submission Date: 2018 December 10

**Table of Contents:**

**Conceptual Model: ER Diagram \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 1**

**Assumptions \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 1**

Relational Model : Data Structure Diagram\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2

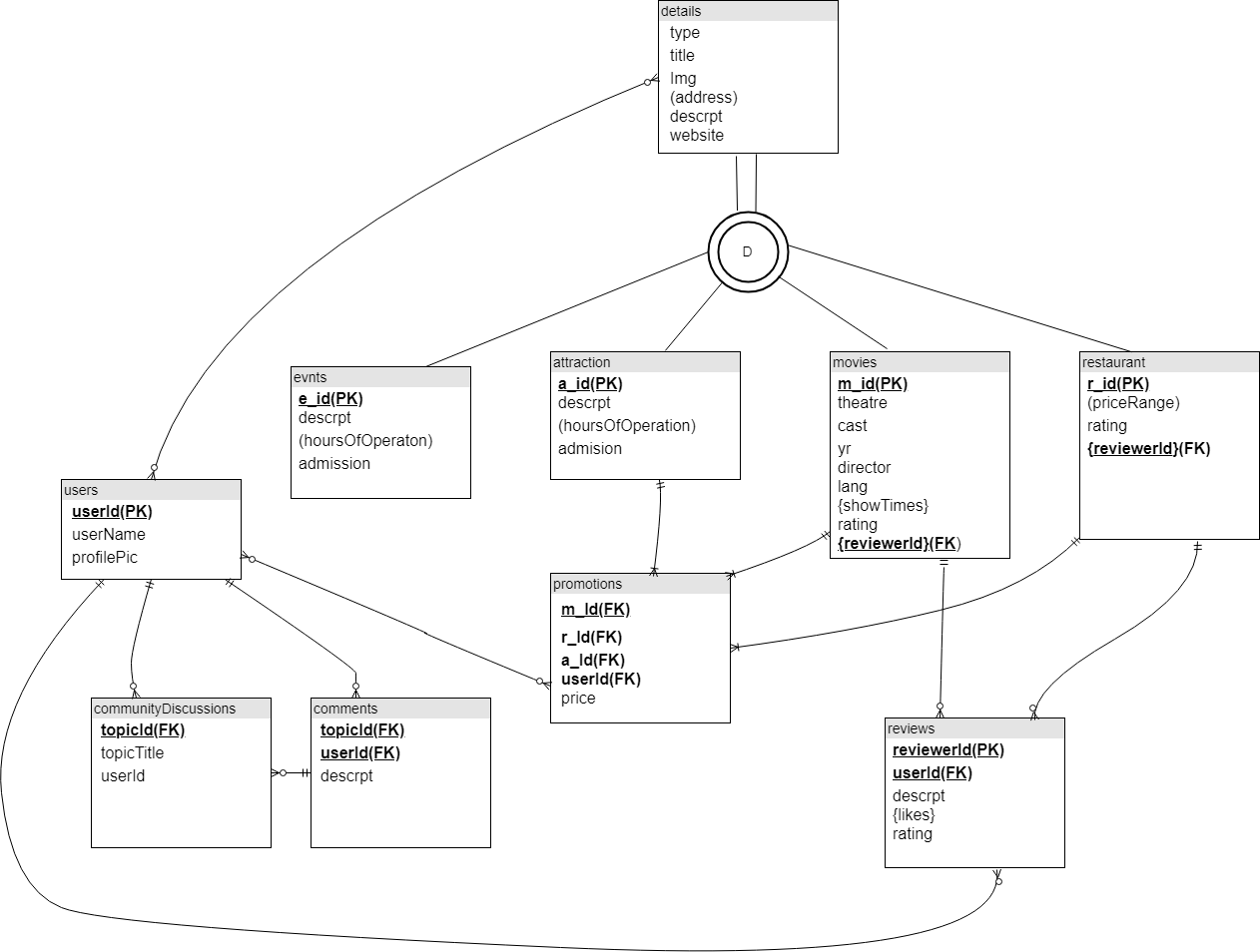
Normalisation \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 3

SQL Queries \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_4-14

NO SQL \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_14-15

Create Collection and Insertion Scripts \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 16

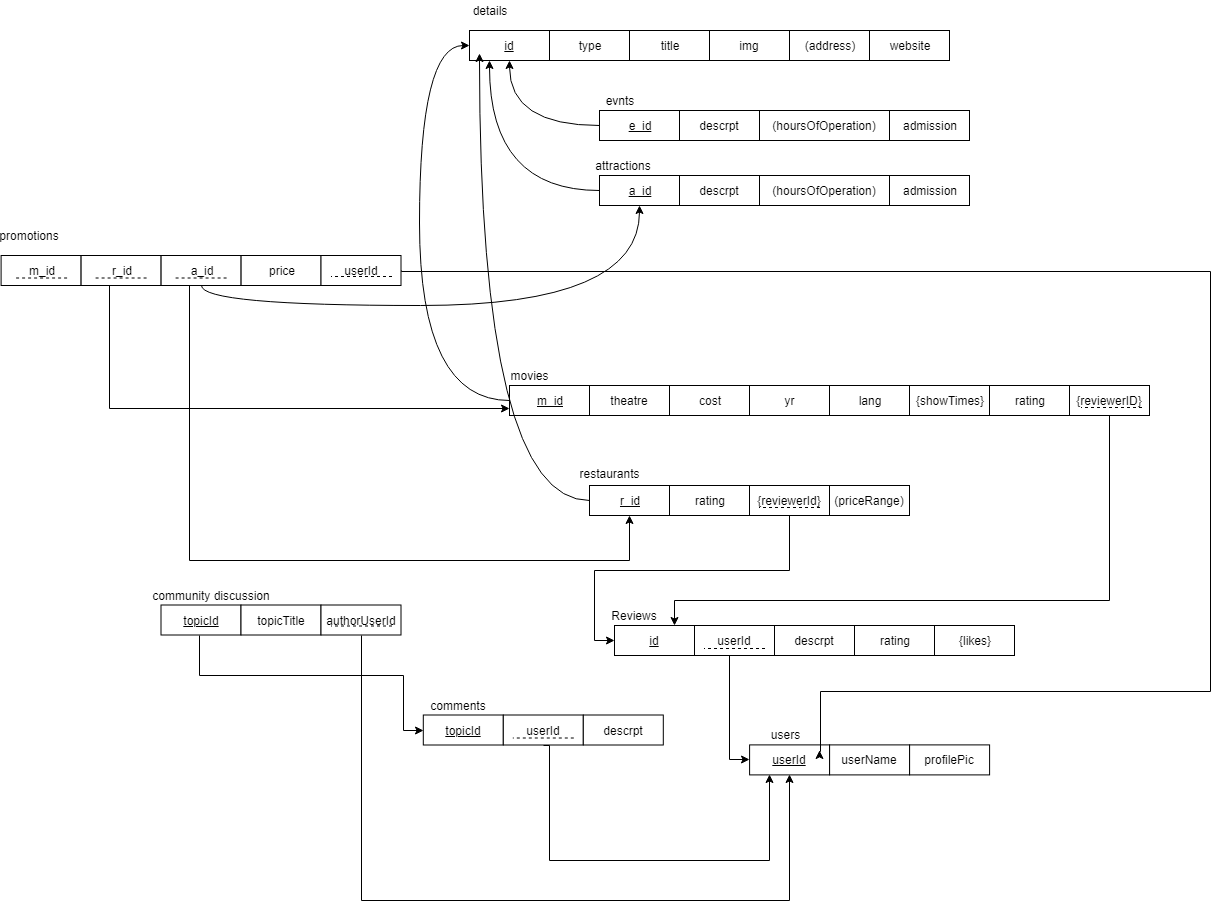
Conceptual Model : ER Diagram



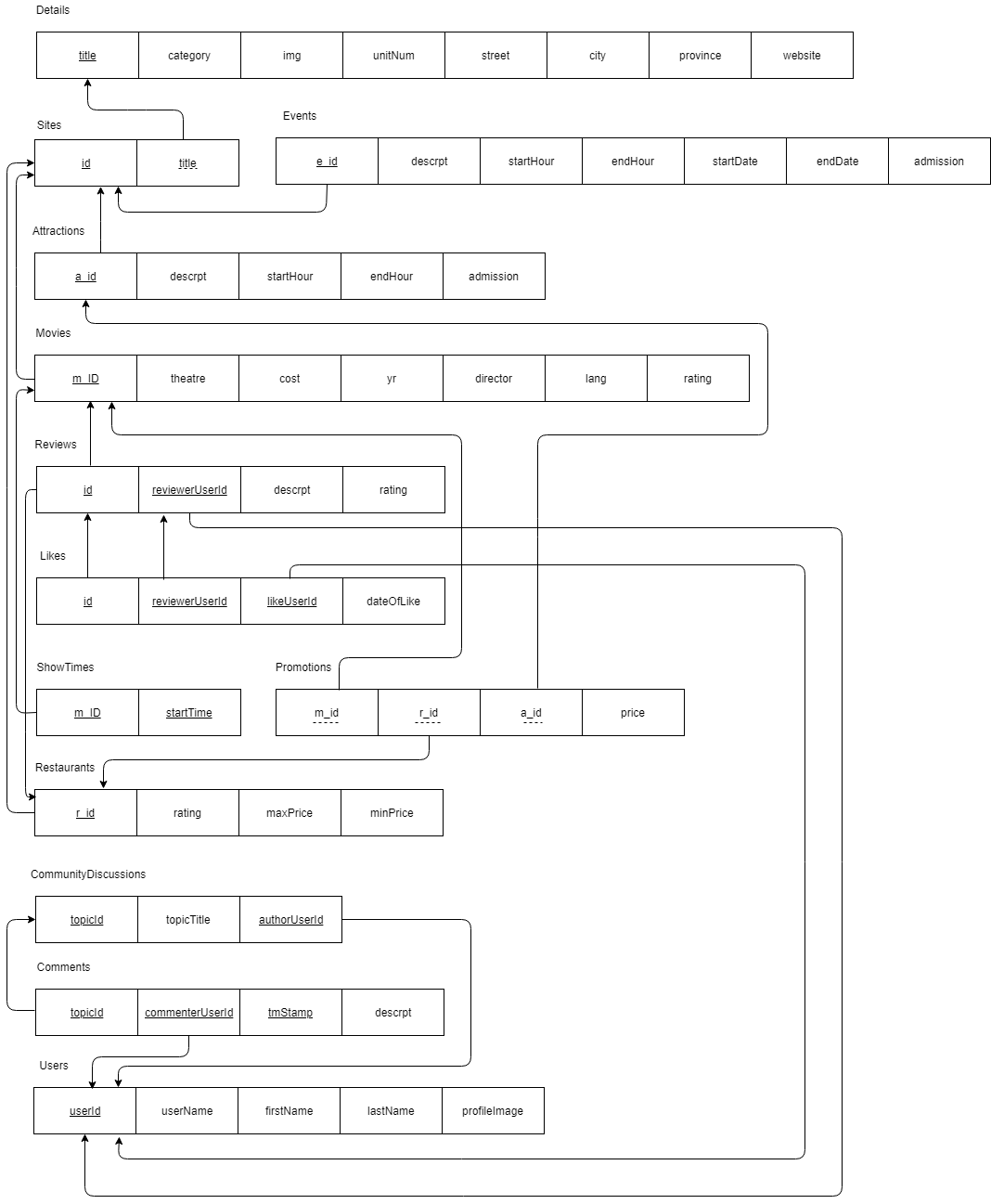
Assumptions

1. Ratings are on a scale of 1-10
2. Location attribute on events wireframe is denoted by address attribute (composite) in our attribute
3. Genre attribute on movies wireframe is denoted by category attribute in our report
4. Cuisine attribute on restaurants wireframe is denoted by category attribute in our report
5. Also assumed that events and movies would have URL, just like restaurants and attractions

Relational Model: Data Structure Diagram



Normalization Report



Create Tables

USE section13;

CREATE TABLE details(

title VARCHAR(200) NOT NULL,

category VARCHAR(200) NOT NULL,

img VARCHAR(200) NOT NULL,

unitNum INT NOT NULL,

street VARCHAR(200) NOT NULL,

city VARCHAR(200) NOT NULL,

province VARCHAR(200) NOT NULL,

website VARCHAR(200) NOT NULL,

CONSTRAINT details\_pk PRIMARY KEY (title)

);

CREATE TABLE sites(

id INT NOT NULL,

title VARCHAR(200) NOT NULL,

CONSTRAINT sites\_pk PRIMARY KEY (id),

CONSTRAINT sites\_fk FOREIGN KEY (title)

REFERENCES details (title)

);

--exec sp\_help details;

--exec sp\_help sites;

CREATE TABLE evnts(

e\_id INT NOT NULL,

descrpt VARCHAR(200) NOT NULL,

startHour VARCHAR(100) NOT NULL,

endHour VARCHAR(100) NOT NULL,

startDate VARCHAR(100) NOT NULL,

endDate VARCHAR(100) NOT NULL,

admission NUMERIC(5,2) NOT NULL,

CONSTRAINT evnts\_pk PRIMARY KEY (e\_id),

CONSTRAINT evnts\_fk FOREIGN KEY (e\_id)

REFERENCES sites (id)

);

CREATE TABLE attractions(

a\_id INT NOT NULL,

descrpt VARCHAR(200) NOT NULL,

startHour VARCHAR(100) NOT NULL,

endHour VARCHAR(100) NOT NULL,

admission NUMERIC(5,2) NOT NULL,

CONSTRAINT attractions\_pk PRIMARY KEY (a\_id),

CONSTRAINT attractions\_fk FOREIGN KEY (a\_id)

REFERENCES sites (id)

);

CREATE TABLE movies(

m\_id INT NOT NULL,

theatre VARCHAR(200) NOT NULL,

cost NUMERIC(5,2) NOT NULL,

yr NUMERIC(4) NOT NULL,

director VARCHAR(200) NOT NULL,

lang VARCHAR(100) NOT NULL,

rating NUMERIC(2,2) NOT NULL,

CONSTRAINT chk\_rating\_movies CHECK(rating <= 10),

CONSTRAINT movies\_pk PRIMARY KEY (m\_id),

CONSTRAINT movies\_fk FOREIGN KEY (m\_id)

REFERENCES sites (id)

);

CREATE TABLE users(

userId INT NOT NULL,

userName VARCHAR(50) NOT NULL,

firstName VARCHAR(100) NOT NULL,

lastName VARCHAR(100) NOT NULL,

profileImage VARCHAR(200),

CONSTRAINT users\_pk PRIMARY KEY (userId)

);

CREATE TABLE restaurants(

r\_id INT NOT NULL,

rating NUMERIC(2,2) NOT NULL,

maxPrice NUMERIC(5,2) NOT NULL,

minPrice NUMERIC(5,2) NOT NULL,

CONSTRAINT chk\_rating\_restaurants CHECK(rating <= 10),

CONSTRAINT restaurants\_pk PRIMARY KEY(r\_id),

CONSTRAINT restaurants\_fk FOREIGN KEY(r\_id)

REFERENCES sites(id)

);

CREATE TABLE reviews(

id INT NOT NULL,

reviewerUserId INT NOT NULL,

descrpt VARCHAR(200) NOT NULL,

rating NUMERIC(2,2) NOT NULL,

CONSTRAINT chk\_rating\_reviews CHECK(rating <= 10),

CONSTRAINT reviews\_pk PRIMARY KEY (id,reviewerUserId),

CONSTRAINT review\_fk\_1 FOREIGN KEY (id)

REFERENCES movies (m\_id),

CONSTRAINT review\_fk\_2 FOREIGN KEY (id)

REFERENCES restaurants (r\_id),

CONSTRAINT review\_fk\_3 FOREIGN KEY (reviewerUserId)

REFERENCES users (userId)

);

CREATE TABLE likes(

id INT NOT NULL,

reviewerUserId INT NOT NULL,

likeUserId INT NOT NULL,

dateOfLike NUMERIC(4) NOT NULL,

CONSTRAINT likes\_pk PRIMARY KEY(id, reviewerUserId, likeUserId),

CONSTRAINT likes\_fk\_1 FOREIGN KEY(id,reviewerUserId)

REFERENCES reviews (id,reviewerUserId),

CONSTRAINT likes\_fk\_2 FOREIGN KEY(likeUserId)

REFERENCES users (userId)

);

CREATE TABLE showTimes(

m\_id INT NOT NULL,

startTime VARCHAR(10) NOT NULL,

CONSTRAINT showTimes\_pk PRIMARY KEY(m\_id, startTime),

CONSTRAINT showTimes\_fk FOREIGN KEY(m\_id)

REFERENCES movies(m\_id)

);

CREATE TABLE promotions(

m\_id INT NOT NULL,

r\_id INT NOT NULL,

a\_id INT NOT NULL,

price NUMERIC(2,2) NOT NULL,

CONSTRAINT chk\_price\_promotions CHECK(price = 45.00),

CONSTRAINT promotions\_pk PRIMARY KEY(m\_id,r\_id,a\_id),

CONSTRAINT promotions\_fk\_1 FOREIGN KEY(m\_id)

REFERENCES movies (m\_id),

CONSTRAINT promotions\_fk\_2 FOREIGN KEY(r\_id)

REFERENCES restaurants (r\_id),

CONSTRAINT promotions\_fk\_3 FOREIGN KEY(a\_id)

REFERENCES attractions (a\_id)

);

CREATE TABLE communityDiscussions(

topicId INT NOT NULL,

topicTitle VARCHAR(200) NOT NULL,

authorUserId INT NOT NULL,

CONSTRAINT communityDiscussions\_pk PRIMARY KEY (topicId),

CONSTRAINT communityDiscussions\_fk FOREIGN KEY (authorUserId)

REFERENCES users (userId)

);

CREATE TABLE comments(

topicId INT NOT NULL,

commenterUserId INT NOT NULL,

tmStamp VARCHAR(50) NOT NULL,

descrpt VARCHAR(200) NOT NULL,

CONSTRAINT comments\_pk PRIMARY KEY (topicId, commenterUserId, tmStamp),

CONSTRAINT comments\_fk\_1 FOREIGN KEY (commenterUserId)

REFERENCES users (userId),

CONSTRAINT comments\_fk\_2 FOREIGN KEY (topicId)

REFERENCES communityDiscussions(topicId)

);

Insert Rows

INSERT INTO details

(title, category, img, unitNum, street, city, province, website)

VALUES('Dinosaur Exhibition', 'Science', 'dinasour.png', 208, 'Great Lakes', 'Brampton', 'Ontario', 'http://tohub.ca/dino-ex/');

INSERT INTO details

(title, category, img, unitNum, street, city, province, website)

VALUES('Sahara', 'Comedy', 'Sahara.png', 231, 'My Street', 'Brampton', 'Ontario', 'http://tohub.ca/cineb/');

INSERT INTO details

(title, category, img, unitNum, street, city, province, website)

VALUES('Museum of Natural History', 'Museum', 'history.png', 235, 'Dynosaur Road', 'Toronto', 'Ontario', 'http://tohub.ca/museum/');

INSERT INTO details

(title, category, img, unitNum, street, city, province, website)

VALUES('Ivar', 'Italian Cuisine', 'ivar.png', 1121, 'King St E', 'Toronto', 'Onatrio', 'http://tohub.ca/ivar/');

INSERT INTO sites

(id, title)

VALUES(1, 'Dinosaur Exhibition');

INSERT INTO sites

(id, title)

VALUES(2, 'Sahara');

INSERT INTO sites

(id, title)

VALUES(3, 'Museum of Natural History');

INSERT INTO sites

(id, title)

VALUES(4, 'Ivar');

INSERT INTO evnts

(e\_id, descrpt, startHour, endHour, startDate, endDate, admission)

VALUES(1, 'Dinosaur exhibition is very attractive for kids. It is coming from museum of natural history in London.', '10:00', '22:00', '2013-09-01', '2018-11-01', 13);

INSERT INTO attractions

(a\_id, descrpt, startHour, endHour, admission)

VALUES(3, 'This museum has the biggest mammals specimen collection bla bla bla ....', '10:00', '22:00', 23);

INSERT INTO movies VALUES(2,'Cineplex Brampton',15.00,2024,'Walid Belal','French',2);

INSERT INTO users

(userId, userName, firstName, lastName, profileImage)

VALUES(1, 'MisSon', 'Misha', 'Sondhi', 'usr01.png');

INSERT INTO users

(userId, userName, firstName, lastName, profileImage)

VALUES(2, 'EbeDan', 'Ebenzer', 'Daniel', 'usr02.png');

INSERT INTO users

(userId, userName, firstName, lastName, profileImage)

VALUES(3, 'YeoPark', 'Yeomnmi', 'Park', 'usr03.png');

INSERT INTO users

(userId, userName, firstName, lastName, profileImage)

VALUES(4, 'BinSh', 'Binod', 'Shaestra', 'usr03.png');

INSERT INTO restaurants

(r\_id, rating, maxPrice, minPrice)

VALUES(4, 9, 75.00, 35.00);

INSERT INTO reviews\_movies

(id, reviewerUserId, descrpt, rating)

VALUES(2, 3, 'Good movie', 9);

INSERT INTO reviews\_restaurants

(id, reviewerUserId, descrpt, rating)

VALUES(4, 1, 'I received exceptional service at the restaurant', 10);

INSERT INTO likes\_restaurants VALUES(4,1,2,'2018-31-03');

INSERT INTO likes\_movies VALUES(2,3,4,'2018-23-09');

INSERT INTO showTimes VALUES(2,'9:00');

INSERT INTO showTimes VALUES(2,'12:00');

INSERT INTO showTimes VALUES(2,'15:00');

INSERT INTO showTimes VALUES(2,'18:00');

INSERT INTO promotions

(m\_id, r\_id, a\_id, price)

VALUES(2, 4, 3, 45.00);

INSERT INTO communityDiscussions

(topicId, topicTitle, authorUserId)

VALUES(1, 'What do you like about Toronto?', 2);

INSERT INTO comments

(topicId, commenterUserId, tmStamp, descrpt)

VALUES(1, 2, '2018-12-09 20:00:00', 'I like the weather');

INSERT INTO comments

(topicId, commenterUserId, tmStamp, descrpt)

VALUES(1, 4, '2018-12-09 22:32:43', 'I like the island');

NoSQL

We chose to create the following 3 collections:

* Movies
* Restaurants
* CommunityDiscussion

Out of all the entities modelled using SQL, the above 3 entities were interlinked with a lot of other entities, thereby creating the need to join a lot of tables using primary key and foreign key constraints in SQL. For example, ID of the movies table was referencing the sites table, and this ID was also being referenced by other tables such as ShowTimes, Reviews (which was in turn being referenced by Likes and Users tables) and Promotions table. By using an embedded approach to model Movies table, it was possible to include the information from many other tables in just one document within the collection in NoSQL. The information stored in just a single document within the Movies collection contained information from Details, Sites, Reviews, Likes, ShowTimes and Users tables. Similarly, the information stored in a single document within the Restaurants collection contained information from Details, Sites, Likes, Reviews and Users table. Finally, each document inserted into CommunityDiscussions collection also had information from Comments and Users table.

We decided to use an embedded approach to NoSQL since it eliminated the need to join different collections, as in SQL. Instead of creating multiple collections and joining them together, it was possible to nest entities within entities using an embedded approach in NoSQL. Another key reason why we chose to use an embedded approach was because the application shown on the wireframe only has predefined queries, for which embedded approach is better in comparison to referenced approach. Examples of predefined queries from the wireframes are search by event type, by date, by theatre, by address, by attraction or by cuisine. These are predefined queries since the users won’t be able to build their own queries. Referenced approach is better for applications where users can build their own queries, but for our application which is limited to predefined queries, embedded approach is better.

Create Collection and Insertion Scripts

use CaseStudy

db.createCollection("Movies")

db.Movies.insertOne(

{

\_id:1,

title:"Sahara",

category:"Comedy",

img:"/images/Sahara.jpeg",

theatre:"Toronto Cineplex",

unitNum:231,

street:"MyStreet",

city:"Toronto",

province:"Ontario",

website:"www.sahara.com",

cost:14.00,

yr:2018,

director:"Walid Bela",

lang:"French",

rating:9.9,

showTimes:["12:30", "15:45"],

reviews:[

{

user:{

reviewerUserId:284,

userName:"TheNiceGuy",

profilePic:"/img/user/TheNiceGuy.png"

},

likes:[

{

likeUserId:111,

userName:"SomeGuy",

dateOfLike:11/03/2018

},

{

likeUserId:121,

userName:"AnotherGuy",

dateOfLike:23/09/2018

}

],

description:"Amazing Movie...",

rating:9.5

},

{

user:{

reviewerUserId:100,

userName:"SherlockHolmes",

profilePic:"/img/user/SherlockHolmes.jpeg"

},

likes:[

{

likeUserId:222,

userName:"GoodGuy",

dateOfLike:19/03/2017

}

],

description:"Don't recommend it",

rating:7.5

}

]

}

)

db.createCollection("Restaurants")

db.Restaurants.insertOne(

{

\_id:1,

title:"Ivars",

category:"Italian",

img:"/images/Ivars.jpeg",

unitNum:123,

street:"Main Street",

city:"Toronto",

province:"Ontario",

website:"www.ivars.com",

rating:9.5,

maxPrice:75.00,

minPrice:35.00,

reviews:[

{

user:{

reviewerUserId:294,

userName:"NiceGlasses",

profilePic:"/img/user/NiceGlasses.jpeg"

},

likes:[

{

likeUserId:385,

userName:"BadGuy",

dateOfLike:31/03/2016

},

{

likeUserId:143,

userName:"Someone",

dateOfLike:07/07/2018

}

],

description:"Great food",

rating:9.5

},

{

user:{

reviewerUserId:938,

userName:"Pierre",

profilePic:"/img/user/Pierre.png"

},

likes:[

{

likeUserId:973,

userName:"Somebody",

dateOfLike:08/08/2015

}

],

description:"Good Price",

rating:10.0

}

]

}

)

db.createCollection("CommunityDiscussions")

db.CommunityDiscussions.insertOne(

{

\_id:1,

topicTitle:"What is the best thing about Toronto",

user:{

userId:294,

name:"NiceGlasses",

profilePic:"/img/user/NiceGlasses.jpeg"

},

comments:[

{

user:{

userId:938,

name:"Pierre",

profilePic:"/img/user/Pierre.png",

},

descrpt:"I like the weather",

tmStamp:"2018:01:31:15:39:34"

},

{

user:{

userId:284,

name:"NiceGuy",

profilePic:"/img/user/NiceGuy.png",

},

descrpt:"I like the island",

tmStamp:"2018:01:30:13:43:24"

}

]

}

)