# Data Base Assignment

**Team 6**
**Rashmi Sharma**
**Neil Thaker**
**Dhrumil Shah**
**Aditya Parmar**

## SQLite Assignment

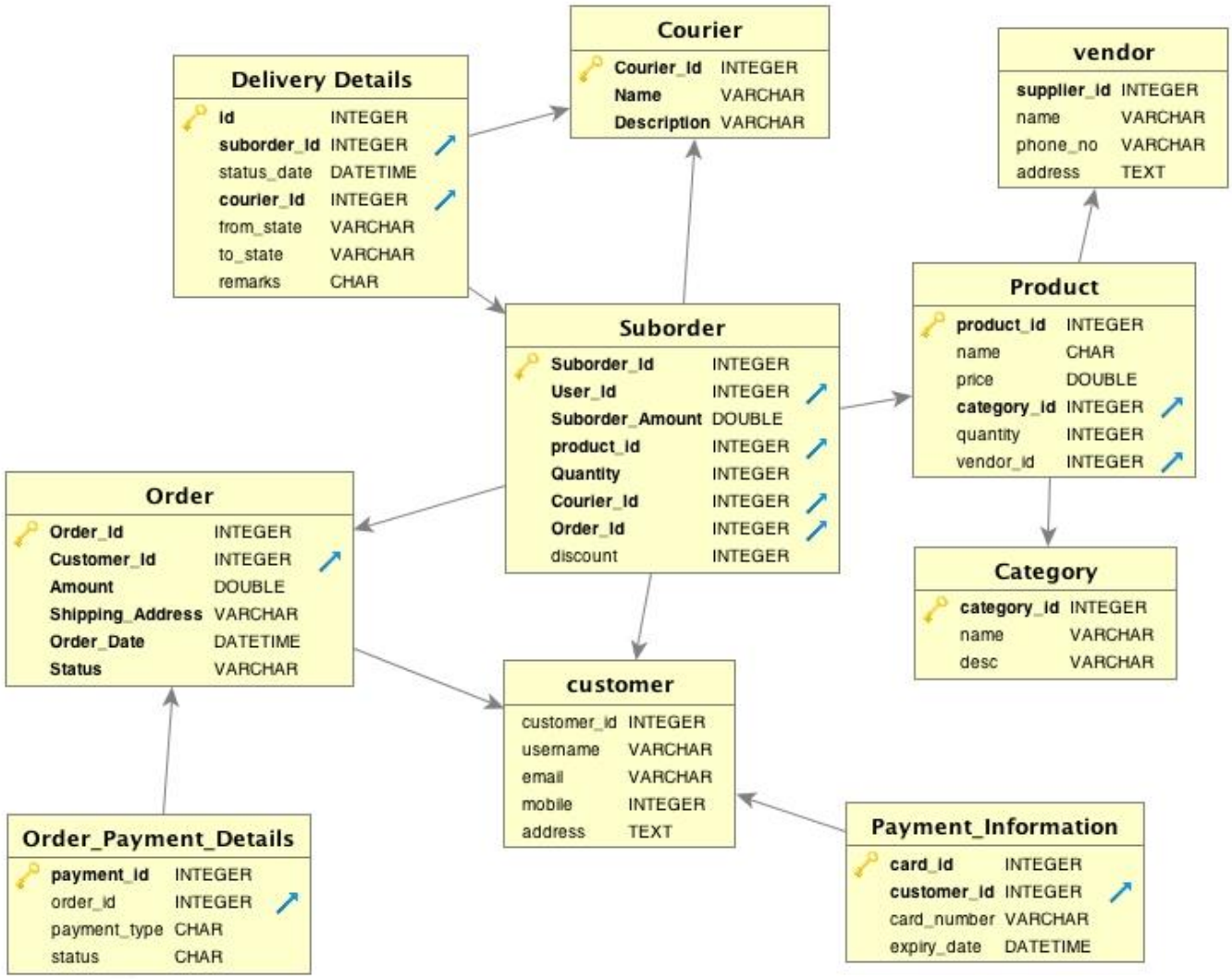
SQLite

- Install SQLite Add-ons for Firefox from:

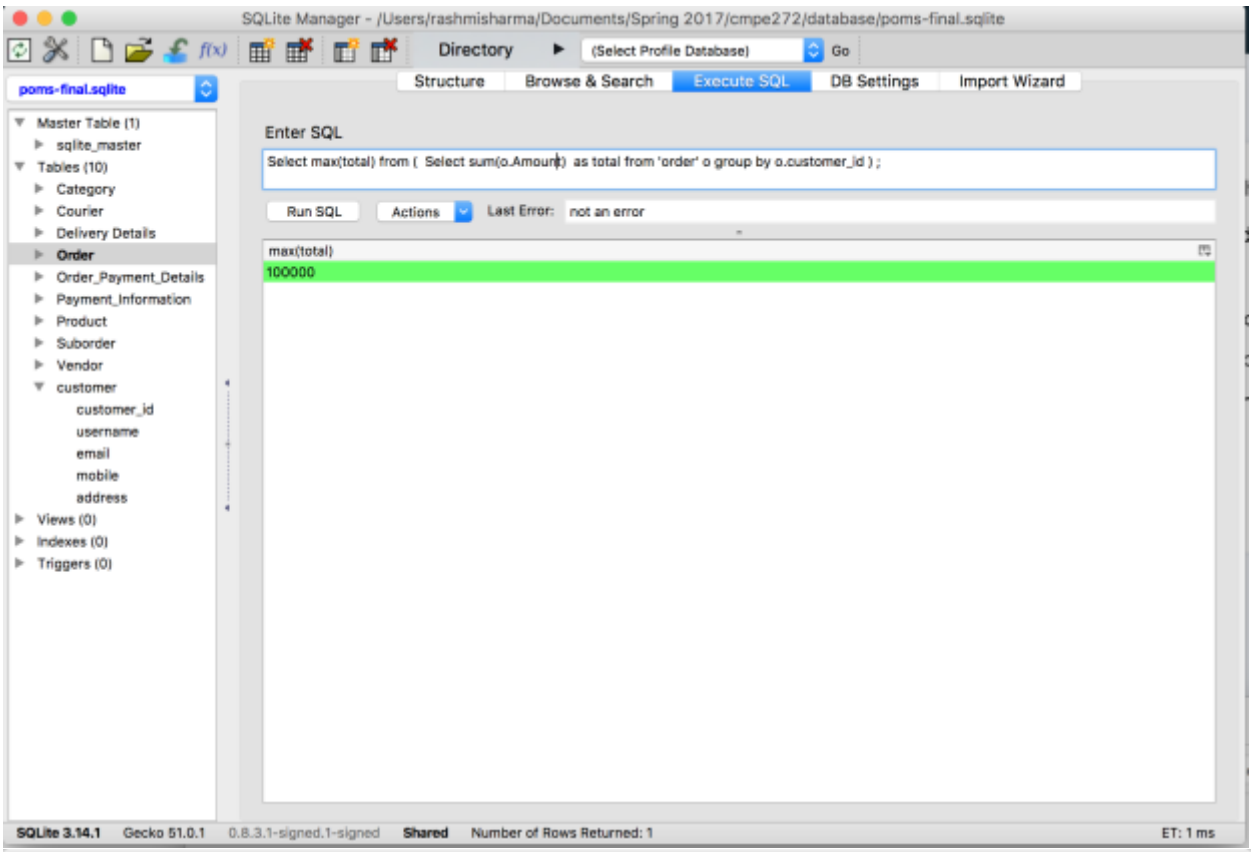
https://addons.mozilla.org/en-us/firefox/addon/sqlite-manager/ (Links to an external site.)

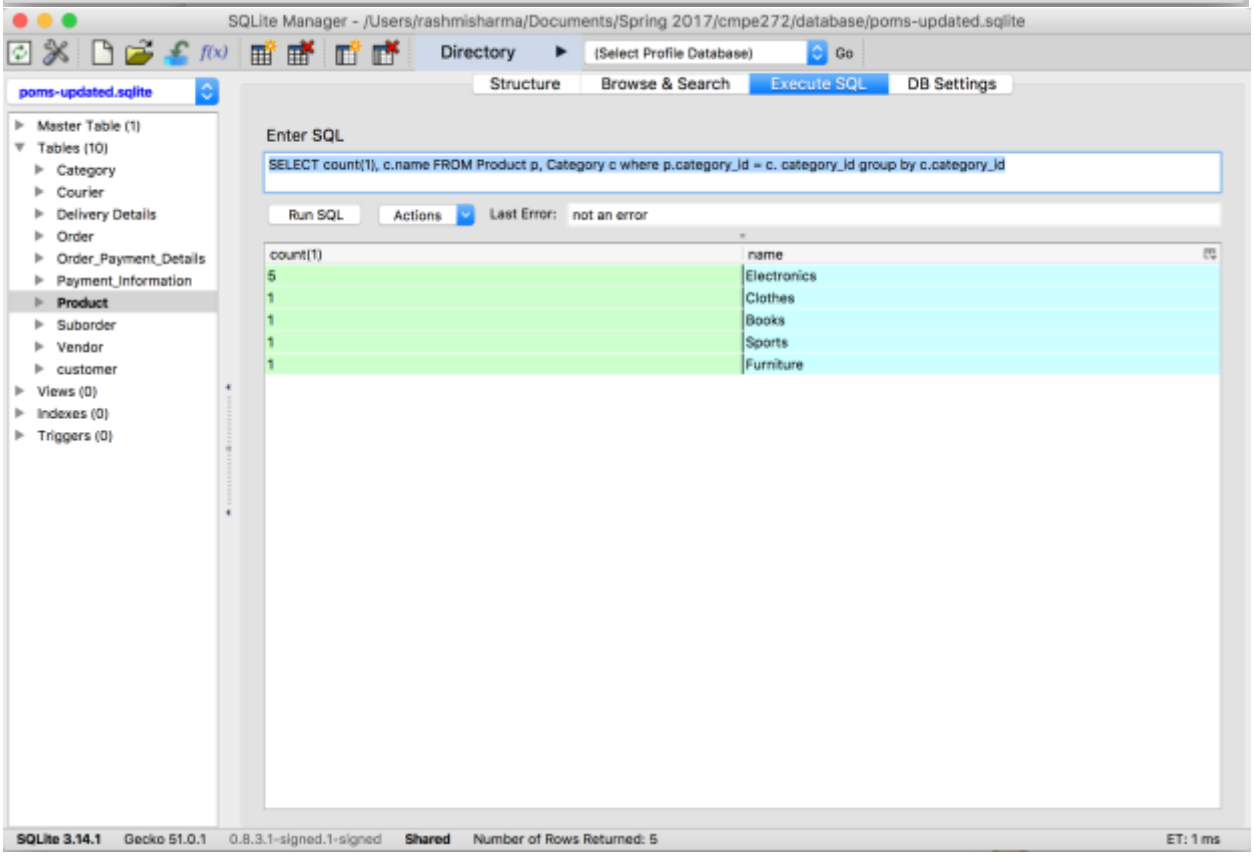
- Design a database for Purchase Order Management System.
- Create a sample schema with necessary tables from previous step.
- Insert sample data
- Try different queries learnt in this chapter

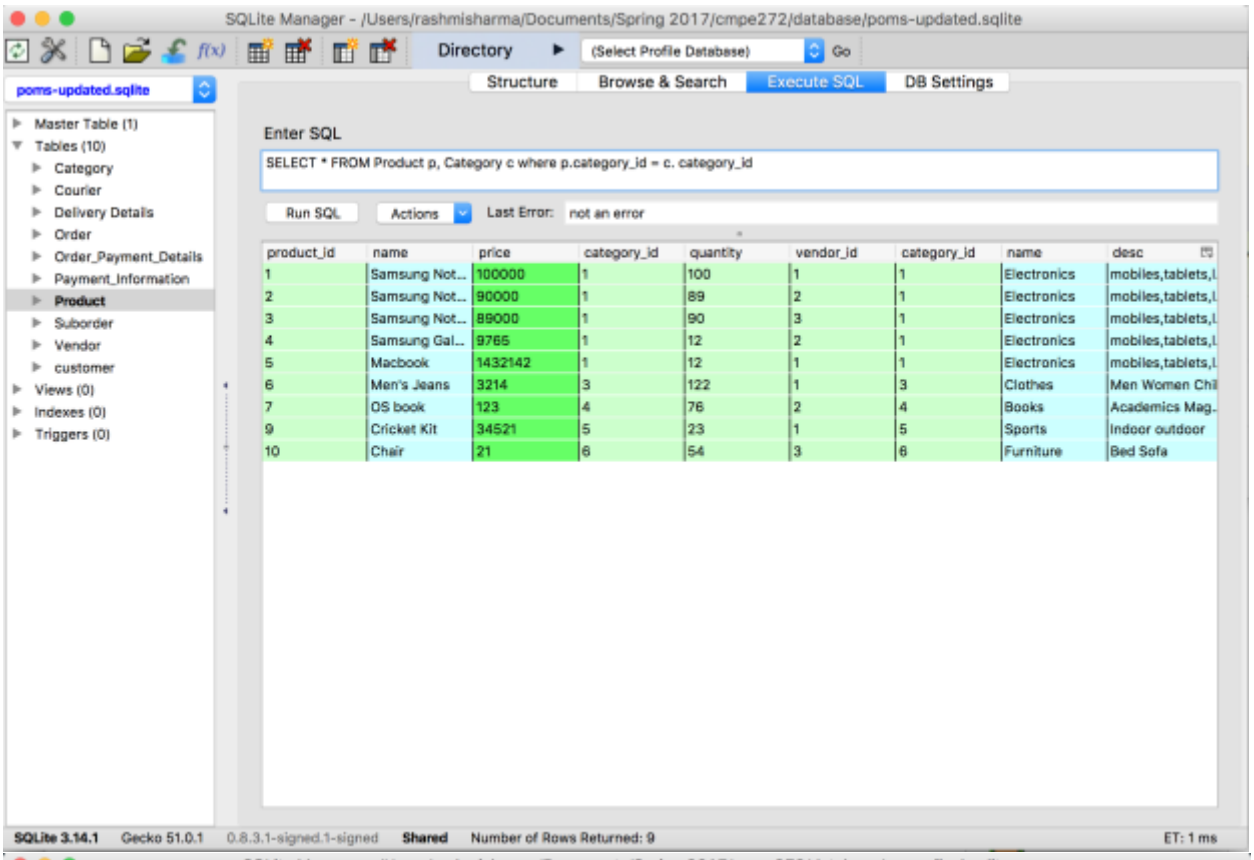
**Database Design Diagram : Purchase Order Management System**

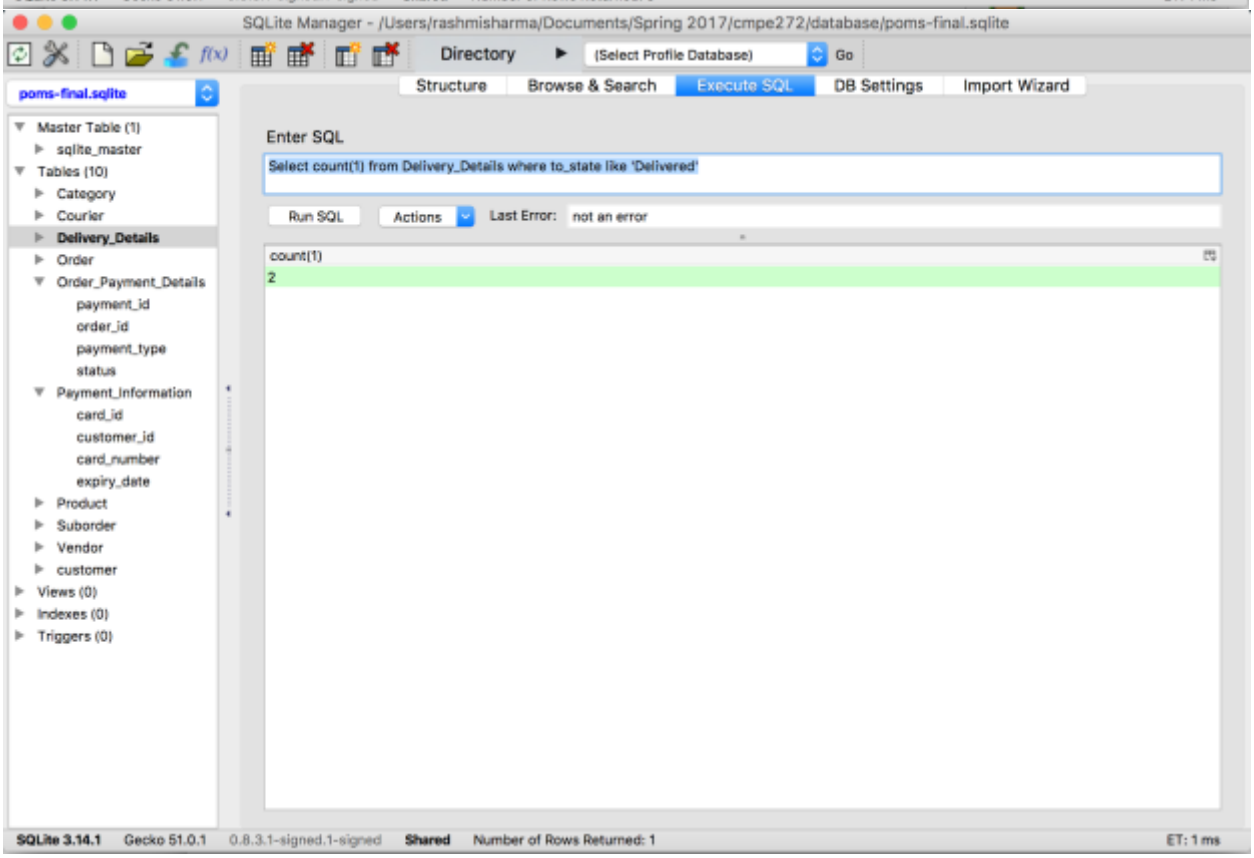



2) Sample Queries


SQLite Manager - /Users/rashmisharma/Documents/Spring 2017/cmpe272/database/poms-final.sqlite
Directory
(Select Profile Database)
Go
poms-final.sqlite
Structure
Browse & Search
Execute SQL
DB Settings
Import Wizard
Master Table (1)
sqlite_master
Tables (10)
Category
Courier
Delivery Details
Order
Order_Payment_Details
Payment_Information
Product
Suborder
Vendor
customer
customer_id
username
email
mobile
address
Views (0)
Indexes (0)
Triggers (0)
Enter SQL
Select max(total) from ( Select sum(o.Amount) as total from 'order' o group by o.customer_id ) ;
Run SQL
Actions
Last Error:
not an error
max(total)
100000
SQLite 3.14.1
Gecko 51.0.1
0.8.3.1-signed.1-signed
Shared
Number of Rows Returned: 1
ET: 1 ms


SQLite Manager - /Users/rashmisharma/Documents/Spring 2017/cmpe272/database/poms-updated.sqlite
Directory
(Select Profile Database)
Go
poms-updated.sqlite
Structure
Browse & Search
Execute SQL
DB Settings
Master Table (1)
Tables (10)
Category
Courier
Delivery Details
Order
Order_Payment_Details
Payment_Information
Product
Suborder
Vendor
customer
Views (0)
Indexes (0)
Triggers (0)
Enter SQL
SELECT count(1), c.name FROM Product p, Category c where p.category_id = c. category_id group by c.category_id
Run SQL
Actions
Last Error:
not an error
count(1)
name
5
Electronics
1
Clothes
1
Books
1
Sports
1
Furniture
SQLite 3.14.1
Gecko 51.0.1
0.8.3.1-signed.1-signed
Shared
Number of Rows Returned: 5
ET: 1 ms


SQLite Manager - /Users/rashmisharma/Documents/Spring 2017/cmpe272/database/poms-updated.sqlite
Directory
(Select Profile Database)
Go
poms-updated.sqlite
Structure
Browse & Search
Execute SQL
DB Settings
Master Table (1)
Tables (10)
Category
Courier
Delivery Details
Order
Order_Payment_Details
Payment_Information
Product
Suborder
Vendor
customer
Views (0)
Indexes (0)
Triggers (0)
Enter SQL
SELECT * FROM Product p, Category c where p.category_id = c. category_id
Run SQL
Actions
Last Error:
not an error
product_id name price category_id quantity vendor_id category_id name desc
1 Samsung Not... 100000 1 100 1 1 Electronics mobiles,tablets,
2 Samsung Not... 90000 1 89 2 1 Electronics mobiles,tablets,
3 Samsung Not... 89000 1 90 3 1 Electronics mobiles,tablets,
4 Samsung Gal... 9765 1 12 2 1 Electronics mobiles,tablets,
5 Macbook 1432142 1 12 1 1 Electronics mobiles,tablets,
6 Men's Jeans 3214 3 122 1 3 Clothes Men Women Chi
7 OS book 123 4 76 2 4 Books Academics Mag.
9 Cricket Kit 34521 5 23 1 5 Sports Indoor outdoor
10 Chair 21 6 54 3 6 Furniture Bed Sofa
SQLite 3.14.1
Gecko 51.0.1
0.8.3.1-signed.1-signed
Shared
Number of Rows Returned: 9
ET: 1 ms


SQLite Manager - /Users/rashmisharma/Documents/Spring 2017/cmpe272/database/poms-final.sqlite
Directory
(Select Profile Database)
Go
poms-final.sqlite
Structure
Browse & Search
Execute SQL
DB Settings
Import Wizard
Master Table (1)
sqlite_master
Tables (10)
Category
Courier
Delivery_Details
Order
Order_Payment_Details
payment_id
order_id
payment_type
status
Payment_Information
card_id
customer_id
card_number
expiry_date
Product
Suborder
Vendor
customer
Views (0)
Indexes (0)
Triggers (0)
Enter SQL
Select count(1) from Delivery_Details where to_state like 'Delivered'
Run SQL
Actions
Last Error:
not an error
count(1)
2
SQLite 3.14.1
Gecko 51.0.1
0.8.3.1-signed.1-signed
Shared
Number of Rows Returned: 1
ET: 1 ms

3) Schema SQL

```
DROP TABLE IF EXISTS "Category";
CREATE TABLE "Category" ("category_id" INTEGER PRIMARY KEY  NOT NULL , "name" VARCHAR, "desc" VARCHAR);

INSERT INTO "Category" VALUES(1,'Electronics','mobiles,tablets,laptop');
INSERT INTO "Category" VALUES(2,'Appliance','AC TV');
INSERT INTO "Category" VALUES(3,'Clothes','Men Women Children Clothes');
INSERT INTO "Category" VALUES(4,'Books','Academics Magazine');
INSERT INTO "Category" VALUES(5,'Sports','Indoor outdoor ');
INSERT INTO "Category" VALUES(6,'Furniture','Bed Sofa');

DROP TABLE IF EXISTS "Courier";
CREATE TABLE "Courier" ("Courier_Id" INTEGER PRIMARY KEY  NOT NULL  DEFAULT (null) ,"Name" VARCHAR NOT NULL ,"Description" VARCHAR NOT NULL );
INSERT INTO "Courier" VALUES(1,'Aramex','Aramex');
INSERT INTO "Courier" VALUES(2,'Delhivery','Delhivery');
INSERT INTO "Courier" VALUES(3,'Fedex','Fedex');

DROP TABLE IF EXISTS "Delivery_Details";
CREATE TABLE "Delivery_Details" ("id" INTEGER PRIMARY KEY  NOT NULL  DEFAULT (null) ,
"suborder_Id" INTEGER NOT NULL  DEFAULT (null) ,
"status_date" DATETIME DEFAULT (null) ,"courier_Id" INTEGER NOT NULL  DEFAULT (0) ,
"from_state" VARCHAR DEFAULT (null) ,"to_state" VARCHAR DEFAULT (null) ,"remarks" CHAR,
FOREIGN KEY (suborder_Id) REFERENCES Suborder(Suborder_Id),
FOREIGN KEY (courier_Id) REFERENCES Courier(Courier_Id)
);
INSERT INTO "Delivery_Details" VALUES(1,1,'14/02/2017',1,'ready for shipping','shipped','QC done');
INSERT INTO "Delivery_Details" VALUES(2,2,'14/02/2017',2,'shipped','delivered','recieved by rashmi');
INSERT INTO "Delivery_Details" VALUES(3,1,'20/02/201',1,'Shipped','Delivered','recieved by Alex');
```

```
DROP TABLE IF EXISTS "Order";
CREATE TABLE "Order" ("Order_Id" INTEGER PRIMARY KEY  NOT NULL  DEFAULT (null) ,
"Customer_Id" INTEGER NOT NULL  DEFAULT (null) ,"Amount" DOUBLE NOT NULL ,
"Shipping_Address" VARCHAR NOT NULL  DEFAULT (null) ,
"Order_Date" DATETIME NOT NULL  DEFAULT (null) ,"Status" VARCHAR NOT NULL,
FOREIGN KEY (Customer_Id) REFERENCES customer(Customer_Id)
 );
INSERT INTO "Order" VALUES(1,1,100000,'754 The Alameda','12/02/2017','Proocessing');
INSERT INTO "Order" VALUES(2,4,100,'Alameda','12/02/2017','Processing');

DROP TABLE IF EXISTS "Order_Payment_Details";

CREATE TABLE "Order_Payment_Details"
("payment_id" INTEGER PRIMARY KEY  NOT NULL , "order_id" INTEGER, "payment_type"
CHAR,
  "status" CHAR,
 FOREIGN KEY (order_id) REFERENCES 'order'(order_id));


INSERT INTO "Order_Payment_Details" VALUES(1,1,'Card','Recieved');
INSERT INTO "Order_Payment_Details" VALUES(2,2,'Card','Decline');
INSERT INTO "Order_Payment_Details" VALUES(3,3,'Cash','Not Recieved');



DROP TABLE IF EXISTS "Payment_Information";
CREATE TABLE "Payment_Information"
("card_id" INTEGER PRIMARY KEY  NOT NULL ,
  "customer_id" INTEGER NOT NULL , "card_number" VARCHAR,
  "expiry_date" DATETIME,
 FOREIGN KEY (customer_id) REFERENCES customer(customer_id));


INSERT INTO "Payment_Information" VALUES(1,1,'1324555566667654','09/17');
INSERT INTO "Payment_Information" VALUES(2,1,'3333512112344320','09/19');
INSERT INTO "Payment_Information" VALUES(3,2,'90833212112344320','11/19');
INSERT INTO "Payment_Information" VALUES(4,3,'1221233245566776','11/20');
INSERT INTO "Payment_Information" VALUES(5,3,'8908908908901234','11/20');
```

```
DROP TABLE IF EXISTS "Product";
CREATE TABLE "Product"
("product_id" INTEGER PRIMARY KEY  NOT NULL ,
  "name" CHAR, "price" DOUBLE, "category_id" INTEGER NOT NULL ,
   "quantity" INTEGER, "vendor_id" INTEGER,
  FOREIGN KEY (category_id) REFERENCES category(category_id),
FOREIGN KEY (vendor_id) REFERENCES vendor(supplier_id));

INSERT INTO "Product" VALUES(1,'Samsung Note 10 ',100000,1,100,1);
INSERT INTO "Product" VALUES(2,'Samsung Note 10 ',90000,1,89,2);
INSERT INTO "Product" VALUES(3,'Samsung Note 10 ',89000,1,90,3);
INSERT INTO "Product" VALUES(4,'Samsung Galaxy Tablet',9765,1,12,2);
INSERT INTO "Product" VALUES(5,'Macbook',1432142,1,12,1);
INSERT INTO "Product" VALUES(6,'Men''s Jeans',3214,3,122,1);
INSERT INTO "Product" VALUES(7,'OS book',123,4,76,2);
INSERT INTO "Product" VALUES(9,'Cricket Kit',34521,5,23,1);
INSERT INTO "Product" VALUES(10,'Chair',21,6,54,3);


DROP TABLE IF EXISTS "Suborder";
CREATE TABLE "Suborder"
("Suborder_Id" INTEGER PRIMARY KEY  NOT NULL  DEFAULT (null)
,"User_Id" INTEGER NOT NULL  DEFAULT (null) ,
"Suborder_Amount" DOUBLE NOT NULL  DEFAULT (null) ,
"product_id" INTEGER NOT NULL  DEFAULT (null) ,
"Quantity" INTEGER NOT NULL ,
"Courier_Id" INTEGER NOT NULL  DEFAULT (null) ,
"Order_Id" INTEGER NOT NULL  DEFAULT (0) ,
 "discount" INTEGER,
  FOREIGN KEY (Courier_Id) REFERENCES Courier(Courier_Id),
  FOREIGN KEY (User_Id) REFERENCES customer(customer_id),
  FOREIGN KEY (Order_Id) REFERENCES 'Order'(Order_Id),
  FOREIGN KEY (product_id) REFERENCES 'Product'(product_id)
);

INSERT INTO "Suborder" VALUES(1,1,100000,1,1,1,1,100);
INSERT INTO "Suborder" VALUES(2,1,42,10,2,2,2,200);
INSERT INTO "Suborder" VALUES(3,3,1432142,5,1,3,3,0);
INSERT INTO "Suborder" VALUES(4,3,89000,3,1,3,3,60);
INSERT INTO "Suborder" VALUES(5,3,34521,9,1,4,3,400);
```

```
DROP TABLE IF EXISTS "Vendor";
CREATE TABLE "Vendor" ("supplier_id" INTEGER PRIMARY KEY  NOT NULL , "name"
VARCHAR, "phone_no" VARCHAR, "address" TEXT);

INSERT INTO "Vendor" VALUES(1,'Sam Enterprise','4089088321','1st street 2287');
INSERT INTO "Vendor" VALUES(2,'Sharma Retails','4318787654','234 The Alameda');
INSERT INTO "Vendor" VALUES(3,'SunShine Retails','1232323213','801 The Cahil Park');


DROP TABLE IF EXISTS "customer";


CREATE TABLE "customer"
("customer_id" INTEGER,"username" VARCHAR,"email" VARCHAR,
  "mobile" INTEGER DEFAULT (null) ,"address" TEXT);

INSERT INTO "customer"
VALUES(1,'adityaparmar03','parmar415@gmail.com',4089088673,'754 The Alameda');
INSERT INTO "customer" VALUES(2,'paratikpatel','pp@gmail.com',7656767890,'754 The
Alameda');
INSERT INTO "customer" VALUES(3,'MeetShah','MS@yahoo.com',8989908988,'754 The
Alameda');
INSERT INTO "customer" VALUES(4,'RashmiSharma','RashmiS@outlook.com',897678769889,'
1314 The Avalon');
INSERT INTO "customer" VALUES(5,'NailThaker','nail10thaker@gmail.com',8989123123,'541
The Canvas');
```

## 2 DB2 Assignment

DB2 Express C

- Download DB2 express C for your operating system http:// (Links to an external site.)www-03.ibm.com/software/products/en/db2expressc (Links to an external site.)
- Create Sample database (use: db2sampl command)
- Run a sample query (use where clause and Group by)
- Generate query explain plan (use: db2exfmt tool)
- Post the query and db2exfmt output snapshot in a pdf document

**Create Database:-**

db2 create database Student

**Create Tables:-**

db2 create table student.studentinfo(stuid int, name varchar(30), deptid int)

db2 create table student.deptinfo(deptid int, name varchar(50))

db2 create table student.profinfo(profid int, name varchar(50), deptid int)

db2 create table student.subject(subid int, name varchar(30), deptid int, profid int)

db2 create table student.Result(Resultid int,subid int,stuid int,profid int,deptid int,per int)

**Inserting Data into Tables:-**

**StudentInfo Table:-**

db2 insert into student.studentinfo(stuid,name ,deptid) values(1,'Aditya',1)
db2 insert into student.studentinfo(stuid,name ,deptid) values(2,'Dhrumil',1)
db2 insert into student.studentinfo(stuid,name ,deptid) values(3,'Neil',1)
db2 insert into student.studentinfo(stuid,name ,deptid) values(4,'Pratik',2)
db2 insert into student.studentinfo(stuid,name ,deptid) values(5,'Dhaval',2)
db2 insert into student.studentinfo(stuid,name ,deptid) values(6,'Milan',2)
db2 insert into student.studentinfo(stuid,name ,deptid) values(7,'Neel',3)
db2 insert into student.studentinfo(stuid,name ,deptid) values(8,'Michel',4)
db2 insert into student.studentinfo(stuid,name ,deptid) values(9,'Harry',4)
db2 insert into student.studentinfo(stuid,name ,deptid) values(10,'Jenny',4)

**Deptinfo Table:-**

db2 insert into student.deptinfo(deptid,name) values(1,'Software Engineering')
db2 insert into student.deptinfo(deptid,name) values(2,'electric Engineering')
db2 insert into student.deptinfo(deptid,name) values(3,'computer Engineering')
db2 insert into student.deptinfo(deptid,name) values(4,'Managment')

**Profinfo Table**

db2 insert into student.profinfo(profid,name,deptid) values(1,'Rajiv Gandhi',1)
db2 insert into student.profinfo(profid,name,deptid) values(2,'Ramesh Kapoor',2)
db2 insert into student.profinfo(profid,name,deptid) values(3,'Soham Kapoor',3)
db2 insert into student.profinfo(profid,name,deptid) values(4,'Kishan Patel',4)
db2 insert into student.profinfo(profid,name,deptid) values(5,'Danish Patel',1)
db2 insert into student.profinfo(profid,name,deptid) values(6,'Rohit Sharma',1)

**Subject Table**

db2 insert into student.subject(subid,name,deptid,profid) values(1,'Database',1,1)
db2 insert into student.subject(subid,name,deptid,profid) values(1,'Operating System',1,5)
db2 insert into student.subject(subid,name,deptid,profid) values(3,'Algorithm',1,6)
db2 insert into student.subject(subid,name,deptid,profid) values(4,'Power System',2,2)
db2 insert into student.subject(subid,name,deptid,profid) values(5,'Embedded System',3,3)
db2 insert into student.subject(subid,name,deptid,profid) values(6,'Marketing',4,9)

**Result Table**

db2 insert into student.Result(Resultid,subid,stuid,profid,deptid,per) values(1,1,1,1,1,95)

db2 insert into student.Result(Resultid,subid,stuid,profid,deptid,per) values(2,2,1,5,1,75)

db2 insert into student.Result(Resultid,subid,stuid,profid,deptid,per) values(3,3,1,6,1,45)

db2 insert into student.Result(Resultid,subid,stuid,profid,deptid,per) values(4,1,2,1,1,45)

db2 insert into student.Result(Resultid,subid,stuid,profid,deptid,per) values(5,2,2,2,1,75)

db2 insert into student.Result(Resultid,subid,stuid,profid,deptid,per) values(6,3,2,6,1,65)

**Query**

select p.name, count(r.stuid) as No_of_students_who_cleared_subject from student.Result r,student.profinfo p where Per>45 and p.profid=r.profid group by p.name

**Output :-**

```
NAME                                               NO_OF_STUDENTS_WHO_CLEARED_SUBJECT
-------------------------------------------------- ----------------------------------
Danish Patel                                                                        2
Rajiv Gandhi                                                                        1
Rohit Sharma                                                                        1

  3 record(s) selected.
```

1) **Explain Plan : ( Without Index)**

**select * from student.studentinfo  where stuid =2**

**db2expln –database student –t –g –f db2expln.sql > explain_plan.txt**

DB2 Universal Database Version 11.1, 5622-044 (c) Copyright IBM Corp. 1991, 2015
Licensed Material - Program Property of IBM
IBM DB2 Universal Database SQL and XQUERY Explain Tool

DB2 Universal Database Version 11.1, 5622-044 (c) Copyright IBM Corp. 1991, 2015

Licensed Material - Program Property of IBM

IBM DB2 Universal Database SQL and XQUERY Explain Tool

********** DYNAMIC **********

==========STATEMENT
==========

Isolation Level = Cursor Stability

Blocking = Block Unambiguous Cursors

Query Optimization Class = 5

Partition Parallel = No

Intra-Partition Parallel = No

SQL Path = "SYSIBM", "SYSFUN", "SYSPROC", "SYSIBMADM",
"ADITYA"

Statement:

select * from student.studentinfo where stuid =2

Section Code Page = 1208

Estimated Cost = 6.789463

Estimated Cardinality = 1.000000

Access Table Name = STUDENT.STUDENTINFO ID = 2,4

| #Columns = 2

| Skip Inserted Rows

| Avoid Locking Committed Data

| Currently Committed for Cursor Stability

| May participate in Scan Sharing structures

| Scan may start anywhere and wrap, for completion

| Fast scan, for purposes of scan sharing management

| Scan can be throttled in scan sharing management

| Relation Scan

| | Prefetch: Eligible

| Lock Intents

| | Table: Intent Share

| | Row : Next Key Share

| Sargable Predicate(s)

| | #Predicates = 1

| | Return Data to Application

| | | #Columns = 3

Return Data Completion

End of section

Optimizer Plan:

Rows

Operator

(ID)

Cost

1

RETURN

( 1)

6.78946

|

1

TBSCAN

( 2)

6.78946

|

10

Table:

STUDENT

STUDENTINFO

## 2) Explain Plan with Index:

**Create Unique Index BASIC_INDEX on student.studentinfo (stuid);**

**db2expln –database student –t –g –f db2expln.sql > explain_plan.txt**

**select * from student.studentinfo where stuid =2**

DB2 Universal Database Version 11.1, 5622-044 (c) Copyright IBM Corp. 1991, 2015
Licensed Material - Program Property of IBM
IBM DB2 Universal Database SQL and XQUERY Explain Tool

DB2 Universal Database Version 11.1, 5622-044 (c) Copyright IBM Corp. 1991, 2015

Licensed Material - Program Property of IBM

IBM DB2 Universal Database SQL and XQUERY Explain Tool

********** DYNAMIC **********

==========STATEMENT
==========

Isolation Level = Cursor Stability

Blocking = Block Unambiguous Cursors

```
        Query Optimization Class = 5
        Partition Parallel       = No
        Intra-Partition Parallel = No
        SQL Path                 = "SYSIBM", "SYSFUN", "SYSPROC", "SYSIBMADM",
                                   "ADITYA"


Statement:
  select *   from student.studentinfo where stuid =2
Section Code Page = 1208
Estimated Cost = 6.783855
Estimated Cardinality = 1.000000
Access Table Name = STUDENT.STUDENTINFO  ID = 2,4
|  Index Scan:  Name = ADITYA.BASIC_INDEX  ID = 1
|  |  Regular Index (Not Clustered)
|  |  Index Columns:
|  |  |  1: STUID (Ascending)
|  #Columns = 2
|  Single Record
|  Fully Qualified Unique Key
|  Skip Inserted Rows
|  Avoid Locking Committed Data
|  Currently Committed for Cursor Stability
```

| Evaluate Predicates Before Locking for Key

| #Key Columns = 1

| | Start Key: Inclusive Value

| | | 1: 2

| | Stop Key: Inclusive Value

| | | 1: 2

| Data Prefetch: None

| Index Prefetch: None

| Lock Intents

| | Table: Intent Share

| | Row : Next Key Share

| Sargable Predicate(s)

| | Return Data to Application

| | | #Columns = 3

Return Data Completion

End of section

Optimizer Plan:

Rows

Operator

(ID)

Cost

1

RETURN


( 1)
6.78386
|
1
FETCH
( 2)
6.78386
/ \
1 10
IXSCAN Table:
( 3) STUDENT
0.0139557 STUDENTINFO
|
0
Index:
ADITYA
BASIC_INDEX

**3) Explain Plan of Group by Query:**

```
select p.name, count(r.stuid)as No_of_students_who_cleared_subject
 from student.Result r, student.profinfo p
 where Per>45 and p.profid=r.profid
 group by p.name
```

DB2 Universal Database Version 11.1, 5622-044 (c) Copyright IBM Corp. 1991,2015
Licensed Material - Program Property of IBM
IBM DB2 Universal Database SQL and XQUERY Explain Tool

DB2 Universal Database Version 11.1, 5622-044 (c) Copyright IBM Corp. 1991, 2015
Licensed Material - Program Property of IBM
IBM DB2 Universal Database SQL and XQUERY Explain Tool

********** DYNAMIC **********

========== STATEMENT ==========

Isolation Level = Cursor Stability

Blocking = Block Unambiguous Cursors

Query Optimization Class = 5

Partition Parallel = No

Intra-Partition Parallel = No

SQL Path = "SYSIBM", "SYSFUN", "SYSPROC", "SYSIBMADM",
"ADITYA"

Statement:

```
select p.name, count(r.stuid)as No_of_students_who_cleared_subject
from student.Result r, student.profinfo p
where Per>45 and p.profid=r.profid
group by p.name
```

Section Code Page = 1208

Estimated Cost = 13.578548

Estimated Cardinality = 3.000000

Access Table Name = STUDENT.RESULT ID = 2,8

| #Columns = 2

| Skip Inserted Rows

| Avoid Locking Committed Data

| Currently Committed for Cursor Stability

| May participate in Scan Sharing structures

| Scan may start anywhere and wrap, for completion

| Fast scan, for purposes of scan sharing management

| Scan can be throttled in scan sharing management

| Relation Scan

| | Prefetch: Eligible

| Lock Intents

| | Table: Intent Share

| | Row : Next Key Share

| Sargable Predicate(s)

| | #Predicates = 1

| | Process Build Table for Hash Join

Hash Join

| Estimated Build Size: 4000

| Estimated Probe Size: 4000

| Access Table Name = STUDENT.PROFINFO ID = 2,6

| | #Columns = 2

| | Skip Inserted Rows

| | Avoid Locking Committed Data

| | Currently Committed for Cursor Stability

| | May participate in Scan Sharing structures

| | Scan may start anywhere and wrap, for completion

| | Fast scan, for purposes of scan sharing management

```
|  |  Scan can be throttled in scan sharing management
|  |  Relation Scan
|  |  |  Prefetch: Eligible
|  |  Lock Intents
|  |  |  Table: Intent Share
|  |  |  Row  : Next Key Share
|  |  Sargable Predicate(s)
|  |  |  Process Probe Table for Hash Join
Insert Into Sorted Temp Table  ID = t1
|  #Columns = 3
|  #Sort Key Columns = 1
|  |  Key 1: NAME (Ascending)
|  Sortheap Allocation Parameters:
|  |  #Rows     = 3.000000
|  |  Row Width = 32
|  Piped
|  Buffered Partial Aggregation
Access Temp Table  ID = t1
|  #Columns = 3
|  Relation Scan
|  |  Prefetch: Eligible
Final Aggregation
|  Group By
```

| Column Function(s)

Return Data to Application

| #Columns = 2

End of section

Optimizer Plan:

Rows

Operator

(ID)

Cost

3

RETURN

( 1)

13.5785

|

3

GRPBY

( 2)

13.578

|

3

TBSCAN

( 3)

13.5776

|

3

SORT

( 4)

13.5769

|

4

HSJOIN

( 5)

13.575

/ \

9 4

TBSCAN TBSCAN

( 6) ( 7)

6.78742 6.7866

| |

9 6

Table: Table:

# IBM Graph

Graph Data store

- Sign up for IBM Bluemix at www.bluemix.net (Links to an external site.)
- Navigate the catalog for Data and Analytics section
- Click on IBM Graph service, create the service and follow the documentation to create a sample graph application using the API documentation: https://ibm-graph-docs.ng.bluemix.net/api.html (Links to an external site.)

1) Creating Social-Network-Graph Schema:

```
URL=https://ibmgraph-alpha.ng.bluemix.net/aeb7460c-8f97-42d6-9baa-5a96182db2db

TOKEN=$(curl "https://ibmgraph-alpha.ng.bluemix.net/aeb7460c-8f97-42d6-9baa-5a96182db2db/_session" -u "440121b8-5bc9-4fab-8573-c997aa6ac7af:b1a1405e-49f7-4680-82f3-fdfd206ede55" | jq -r '.["gds-token"]')
echo "Your session token is $TOKEN"


GRAPH="social-network-graph" # graph name with a time stamp

ECHO $GRAPH
curl "$URL/_graphs/$GRAPH" \
    -X POST \
    -H "Authorization: gds-token $TOKEN" \
    -d '' | jq '.'


    SCHEMA='
    {
      "propertyKeys": [
        {"name": "name", "dataType": "String", "cardinality": "SINGLE"},
        {"name": "status", "dataType": "String", "cardinality": "SINGLE"},
        {"name": "age", "dataType": "Integer", "cardinality": "SINGLE"},
        {"name": "location", "dataType": "String", "cardinality": "SINGLE"},
        {"name": "text", "dataType": "String", "cardinality": "SINGLE"},
        {"name": "tags", "dataType": "String", "cardinality": "SINGLE"},
        {"name": "file", "dataType": "String", "cardinality": "SINGLE"},
        {"name": "group_name", "dataType": "String", "cardinality": "SINGLE"},
        {"name": "group_desc", "dataType": "String", "cardinality": "SINGLE"},
        {"name": "status_time", "dataType": "String", "cardinality": "SINGLE"}
```

```
    ],
    "vertexLabels": [
      {"name": "person"},
      {"name": "photograph"},
      {"name": "post"},
      {"name": "group"}
    ],
    "edgeLabels": [
      {"name": "uploads", "multiplicity": "MULTI"},
      {"name": "updates", "multiplicity": "MULTI"},
      {"name": "joins", "multiplicity": "MULTI"},
      {"name": "likes", "multiplicity": "MULTI"},
      {"name": "connects", "multiplicity": "MULTI"}

    ],
    "vertexIndexes": [
      {"name": "vByName", "propertyKeys": ["name"], "composite": true, "unique": true},
      {"name": "vByAge", "propertyKeys": ["age"], "composite": true, "unique": false},
      {"name": "vByGroup", "propertyKeys": ["group_name"], "composite": true, "unique":
false}
    ],
    "edgeIndexes" :[
      {"name": "eByStatusTime", "propertyKeys": ["status_time"], "composite": true,
"unique": false}
    ]
  }'

  curl "$URL/$GRAPH/schema" \
       -X POST \
       -H "Authorization: gds-token $TOKEN" \
       -H 'Content-Type: application/json' \
       -d "$SCHEMA" | jq '.'
```

OUTPUT:

```
Rashmis-MacBook-Pro:graph-db rashmishar ma$ sh login_script.sh
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   144    0   144    0     0     39      0 --:--:--  0:00:03 --:--:--    39
Your session token is
NDQwMTIxYjgtNWJjOS00ZmFiLTg1NzMtYzk5N2FhNmFj N2FmOj E0ODc4MDM0NjYzNTk6RDNFWk5qMFhlaDhJR
EMzMnh4c0ZFdGt6NzFmYTN1NXhRZFVwaHRYS0dj QT0=
social-network-graph
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
```

```
                                 Dload  Upload   Total   Spent    Left  Speed
100   140    0   140    0     0    378      0 --:--:-- --:--:-- --:--:--   378
{
  "graphId": "social-network-graph",
  "dbUrl": "https://ibmgraph-alpha.ng.bluemix.net/aeb7460c-8f97-42d6-9baa-5a96182db2db/social-network-graph"
}
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
                                 Dload  Upload   Total   Spent    Left  Speed
100  3389    0  1668  100  1721    869    896  0:00:01  0:00:01 --:--:--   896
{
  "requestId": "f7305471-9650-4b24-a35c-74363e60aa12",
  "status": {
    "message": "",
    "code": 200,
    "attributes": {}
  },
  "result": {
    "data": [
      {
        "propertyKeys": [
          {
            "name": "name",
            "dataType": "String",
            "cardinality": "SINGLE"
          },
          {
            "name": "status",
            "dataType": "String",
            "cardinality": "SINGLE"
          },
          {
            "name": "age",
            "dataType": "Integer",
            "cardinality": "SINGLE"
          },
          {
            "name": "location",
            "dataType": "String",
            "cardinality": "SINGLE"
          },
          {
            "name": "text",
            "dataType": "String",
            "cardinality": "SINGLE"
          },
          {
            "name": "tags",
            "dataType": "String",
            "cardinality": "SINGLE"
          },
          {
            "name": "file",
            "dataType": "String",
            "cardinality": "SINGLE"
          },
          {
            "name": "group_name",
            "dataType": "String",
            "cardinality": "SINGLE"
          },
          {
```

```
      "name": "group_desc",
      "dataType": "String",
      "cardinality": "SINGLE"
    },
    {
      "name": "status_time",
      "dataType": "String",
      "cardinality": "SINGLE"
    }
  ],
  "vertexLabels": [
    {
      "name": "person"
    },
    {
      "name": "photograph"
    },
    {
      "name": "post"
    },
    {
      "name": "group"
    }
  ],
  "edgeLabels": [
    {
      "name": "updates",
      "directed": true,
      "multiplicity": "MULTI"
    },
    {
      "name": "joins",
      "directed": true,
      "multiplicity": "MULTI"
    },
    {
      "name": "likes",
      "directed": true,
      "multiplicity": "MULTI"
    },
    {
      "name": "connects",
      "directed": true,
      "multiplicity": "MULTI"
    },
    {
      "name": "uploads",
      "directed": true,
      "multiplicity": "MULTI"
    }
  ],
  "vertexIndexes": [
    {
      "name": "vByName",
      "composite": true,
      "unique": true,
      "propertyKeys": [
        "name"
      ],
      "requiresReindex": false,
      "type": "vertex"
    },
```

```
      {
        "name": "vByAge",
        "composite": true,
        "unique": false,
        "propertyKeys": [
          "age"
        ],
        "requiresReindex": false,
        "type": "vertex"
      },
      {
        "name": "vByGroup",
        "composite": true,
        "unique": false,
        "propertyKeys": [
          "group_name"
        ],
        "requiresReindex": false,
        "type": "vertex"
      }
    ],
    "edgeIndexes": [
      {
        "name": "eByStatusTime",
        "composite": true,
        "unique": false,
        "propertyKeys": [
          "status_time"
        ],
        "requiresReindex": false,
        "type": "edge"
      }
    ]
  }
 ],
 "meta": {}
}
}
Rashmis-MacBook-Pro:graph-db rashmishar ma$
```

2) Adding Data to Graph

GRAPH="social-network-graph" # graph name with a time stamp
URL=https://ibmgraph-alpha.ng.bluemix.net/aeb7460c-8f97-42d6-9baa-5a96182db2db
TOKEN=NDQwMTIxYjgtNWJjOS00ZmFiLTg1NzMtYzk5N2FhNmFjN2FmOjE0ODc3NTQyOTg4ODQ6RlN6aUl1bEJUWFgwRUtTMW51bzB3NENPTHd0eFJpa3c5eUtQU25wYWk2OD0=

cat << ENDGREMLIN >gremlin.json # write everything until ENDGREMLIN into gremlin.json
{
 "gremlin": "
 def alex = graph.addVertex(T.label, 'person', 'name', 'Alex', 'status', 'Single','age',18);

```
  def john =  graph.addVertex(T.label, 'person', 'name', 'John', 'status', 'Single','age',38);
  def lisa =  graph.addVertex(T.label, 'person', 'name', 'Lisa', 'status', 'Married','age',28);

  def sjsu_group =  graph.addVertex(T.label, 'group', 'group_name', 'SjSU Group', 'group_desc', 'SJSU Group');


  def alexPost =  graph.addVertex(T.label, 'post', 'text', 'BlueMix is great!', 'tags', '#Bluemix, #Awesome','status_time','21/02/2017','file','None');
  def johnPost =  graph.addVertex(T.label, 'post', 'text', 'Apache Tinker', 'tags', '#Apache, #Awesome','status_time','21/02/2017','file','None');


  def lisaPhotograph =  graph.addVertex(T.label, 'photograph', 'location', 'San Francisco', 'file','abc.jpg');

  alex.addEdge('updates', alexPost);
  john.addEdge('updates', johnPost);

  alex.addEdge('connects', john);
  alex.addEdge('connects', lisa);
  john.addEdge('connects', lisa);

  lisa.addEdge('uploads', lisaPhotograph);
  john.addEdge('joins', sjsu_group);

  john.addEdge('likes', lisaPhotograph);
  alex.addEdge('likes', johnPost);
  "
}
ENDGREMLIN

curl "$URL/$GRAPH/gremlin" \
     -X POST \
     -H "Authorization: gds-token $TOKEN" \
     -H 'Content-Type: application/json' \
     -d @gremlin.json | jq '.'
```

OUTPUT:

```
Rashmis-MacBook-Pro:graph-db rashmishar ma$ sh load_data.sh
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  1393    0   251  100  1142     56    255  0:00:04  0:00:04 --:--:--   255
{
  "requestId": "adf0ff2f-0e7f-4c56-bdb7-da5f73946ed3",
  "status": {
    "message": "",
    "code": 200,
    "attributes": {}
  },
  "result": {
    "data": [
      {
        "id": "3kl-3ag-e8l-9mo",
        "label": "likes",
        "type": "edge",
        "inVLabel": "post",
        "outVLabel": "person",
        "inV": 12480,
        "outV": 4264
      }
    ],
    "meta": {}
  }
```

3) Query Database:

***Find a person with name Alex***

Curl –s "https://ibmgraph-alpha.ng.bluemix.net/aeb7460c-8f97-42d6-9baa-5a96182db2db/social-network-graph/vertices?name=Alex&label=person" \
-H "Authorization: gds-token NDQwMTIxYjgtNWJjOS00ZmFiLTg1NzMtYzk5N2FhNmFjN2FmOjE0ODc3NTQyOTg4ODQ6RlN6aUl1bEJUWFgwRUtTMW51bzB3NENPTHd0eFJpa3c5eUtQU25wYWk2OD0=" | jq '.'

OutPut:

```
curl -s "https://ibmgraph-alpha.ng.bluemix.net/aeb7460c-8f97-42d6-9baa-5a96182db2db/social-network-graph/vertices?name=Alex&label=person" \
>     -H "Authorization: gds-token NDQwMTIxYjgtNWJjOS00ZmFiLTg1NzMtYzk5N2FhNmFjN2FmOjE0ODc3NTQyOTg4ODQ6RlN6aUl1bEJUWFgwRUtTMW51bzB3NENPTHd0eFJpa3c5eUtQU25wYWk2OD0=" | jq '.'
{
  "requestId": "365d241e-c0e8-4d68-8f41-b5349d67d8c0",
  "status": {
    "message": "",
    "code": 200,
    "attributes": {}
  },
  "result": {
    "data": [
      {
        "id": 4264,
        "label": "person",
        "type": "vertex",
        "properties": {
          "name": [
            {
              "id": "179-3ag-sl",
              "value": "Alex"
            }
          ],
          "age": [
            {
              "id": "1zp-3ag-2dh",
              "value": 18
            }
          ],
          "status": [
            {
              "id": "1lh-3ag-1l1",
              "value": "Single"
```

```
        }
       ]
      }
     }
    ],
    "meta": {}
  }
}
```


### *Find a person with age =28*

curl -s "https://ibmgraph-alpha.ng.bluemix.net/aeb7460c-8f97-42d6-9baa-5a96182db2db/social-network-graph/vertices?age=28&label=person" \
-H "Authorization: gds-token NDQwMTIxYjgtNWJjOS00ZmFiLTg1NzMtYzk5N2FhNmFjN2FmOjE0ODc3NTQyOTg4ODQ6RlN6aUl1bEJUWFgwRUtTMW51bzB3NENPTHd0eFJpa3c5eUtQU25wYWk2OD0=" | jq '.'


```
curl -s "https://ibmgraph-alpha.ng.bluemix.net/aeb7460c-8f97-42d6-9baa-5a96182db2db/social-network-graph/vertices?age=28&label=person" \
>       -H "Authorization: gds-token NDQwMTIxYjgtNWJjOS00ZmFiLTg1NzMtYzk5N2FhNmFjN2FmOjE0ODc3NTQyOTg4ODQ6RlN6aUl1bEJUWFgwRUtTMW51bzB3NENPTHd0eFJpa3c5eUtQU25wYWk2OD0=" | jq '.'
{
  "requestId": "c087d4fa-47c2-40c1-a924-bed694eac2eb",
  "status": {
    "message": "",
    "code": 200,
    "attributes": {}
  },
  "result": {
    "data": [
      {
        "id": 4288,
        "label": "person",
        "type": "vertex",
        "properties": {
          "name": [
            {
              "id": "17c-3b4-sl",
              "value": "Lisa"
            }
          ],
          "age": [
            {
              "id": "1zs-3b4-2dh",
              "value": 28
            }
          ],
          "status": [
            {
              "id": "1lk-3b4-1l1",
              "value": "Married"
            }
          ]
        }
```

```
    }
   ],
   "meta": {}
  }
}
Rashmis-MacBook-Pro:graph-db rashmisharma$
```

### *Find Friends of Alex:*

```
GRAPH="social-network-graph" # graph name with a time stamp
URL=https://ibmgraph-alpha.ng.bluemix.net/aeb7460c-8f97-42d6-9baa-5a96182db2db
TOKEN=NDQwMTIxYjgtNWJjOS00ZmFiLTg1NzMtYzk5N2FhNmFjN2FmOjE0ODc3NTQyOTg4ODQ6RlN6aUl1bEJUWFgwRUtTMW51bzB3NENPTHd0eFJpa3c5eUtQU25wYWk2OD0=



cat << ENDGREMLIN >gremlin.json
{
  "gremlin":                    "graph.traversal().V().hasLabel('person').has('name',
'Alex').outE('connects').inV();"
}
ENDGREMLIN
curl "$URL/$GRAPH/gremlin" \
     -X 'POST' \
     -H "Authorization: gds-token $TOKEN" \
     -H 'Content-Type: application/json' \
     -d @gremlin.json | jq '.'
```

Output:

```
sh complex-gremlin.sh
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   621    0   520  100   101   1229    238 --:--:-- --:--:-- --:--:--  1232
{
  "requestId": "c61e1002-67d1-46d0-80a1-7b4412e14297",
  "status": {
    "message": "",
```

```
  "code": 200,
  "attributes": {}
 },
 "result": {
  "data": [
   {
    "id": 4288,
    "label": "person",
    "type": "vertex",
    "properties": {
     "name": [
      {
       "id": "17c-3b4-sl",
       "value": "Lisa"
      }
     ],
     "age": [
      {
       "id": "1zs-3b4-2dh",
       "value": 28
      }
     ],
     "status": [
      {
       "id": "1lk-3b4-1l1",
       "value": "Married"
      }
     ]
    }
   },
   {
    "id": 4304,
    "label": "person",
    "type": "vertex",
    "properties": {
     "name": [
      {
       "id": "17e-3bk-sl",
       "value": "John"
      }
     ],
     "age": [
      {
       "id": "1zu-3bk-2dh",
       "value": 38
      }
     ],
     "status": [
      {
       "id": "1lm-3bk-1l1",
       "value": "Single"
      }
     ]
    }
   }
  ],
  "meta": {}
 }
}
Rashmis-MacBook-Pro:graph-db rashmishar ma$
```

***Find Post updates from Alex***

```
GRAPH="social-network-graph" # graph name with a time stamp
URL=https://ibmgraph-alpha.ng.bluemix.net/aeb7460c-8f97-42d6-9baa-5a96182db2db
TOKEN=NDQwMTIxYjgtNWJjOS00ZmFiLTg1NzMtYzk5N2FhNmFjN2FmOjE0ODc3NT
QyOTg4ODQ6RlN6aUl1bEJUWFgwRUtTMW51bzB3NENPTHd0eFJpa3c5eUtQU25wY
Wk2OD0=



cat << ENDGREMLIN >gremlin.json
{
  "gremlin":                    "graph.traversal().V().hasLabel('person').has('name',
'Alex').outE('updates').inV();"
}
ENDGREMLIN
curl "$URL/$GRAPH/gremlin" \
     -X 'POST' \
     -H "Authorization: gds-token $TOKEN" \
     -H 'Content-Type: application/json' \
     -d @gremlin.json | jq '.'
```

Output:

```
sh complex-gremlin.sh
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   511    0   411  100   100    696    169 --:--:-- --:--:-- --:--:--   696
{
  "requestId": "5ec9bfad-9042-4dd8-a7aa-f34da02238e9",
  "status": {
    "message": "",
    "code": 200,
    "attributes": {}
  },
  "result": {
    "data": [
```

```
    {
      "id": 4232,
      "label": "post",
      "type": "vertex",
      "properties": {
        "file": [
          {
            "id": "2dt-39k-5j9",
            "value": "None"
          }
        ],
        "text": [
          {
            "id": "175-39k-3yd",
            "value": "BlueMix is great!"
          }
        ],
        "status_time": [
          {
            "id": "1zl-39k-7wl",
            "value": "21/02/2017"
          }
        ],
        "tags": [
          {
            "id": "1ld-39k-4qt",
            "value": "#Bluemix, #Awesome"
          }
        ]
      }
    }
  ],
  "meta": {}
 }
}
```