DATA PROCESSING AND ANALYTICS

GROUP ASSIGNMENT
(GROUP 29)

NAMES:

WILSON SUNDAY OTITONAIYE	**S5229500**
ABHAY SUJALA KRISHNAN	**S5229584**
NIKHIL CHANDRANNOLA	**S5229394**
LINGAM ANIL KUMAR	**S5229392**

TASK A

ENTITY RELATIONSHIP DIAGRAM

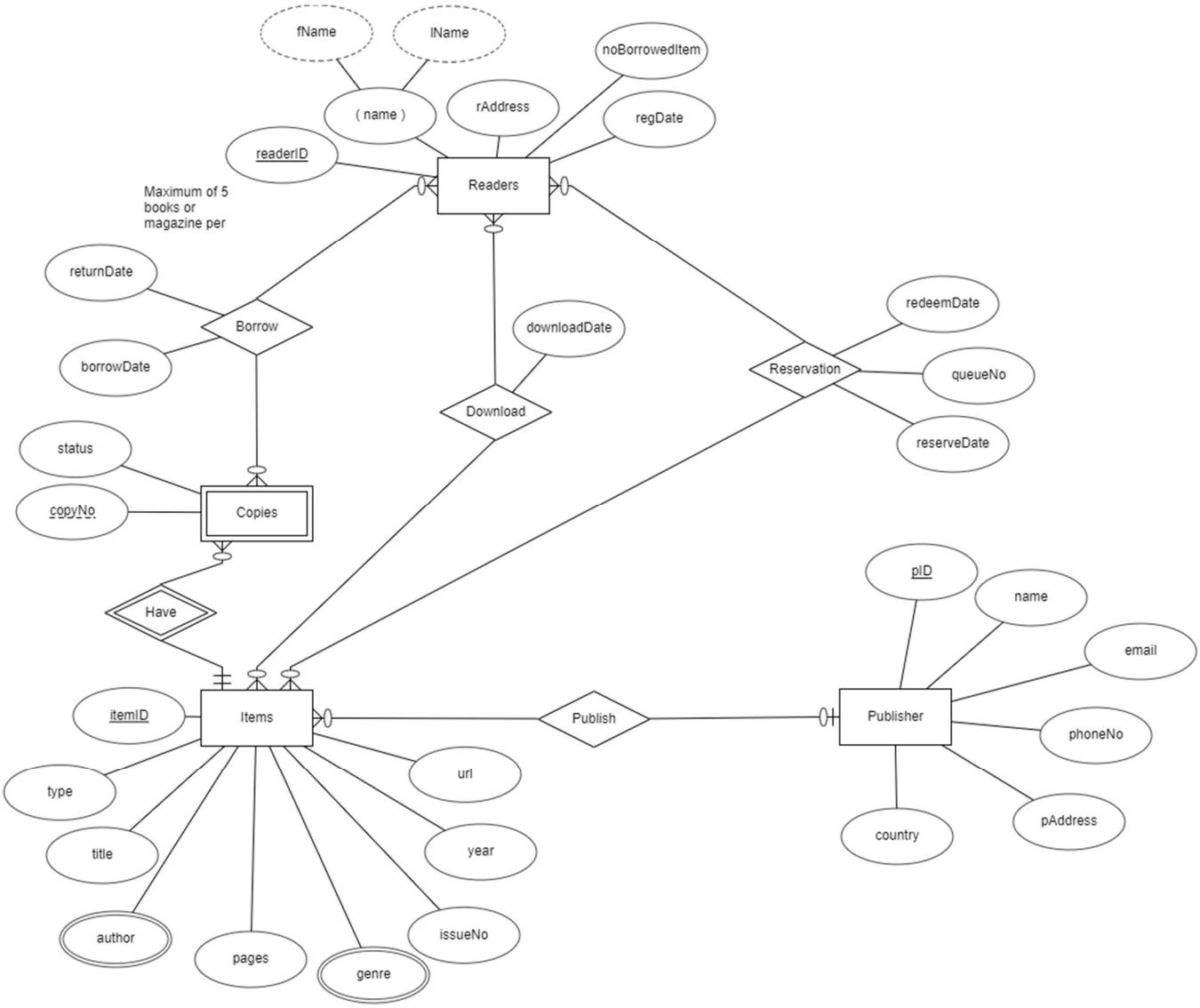

Figure 1 - Entity Relationship Diagram

RELATIONSHIP SCHEMA DIAGRAM

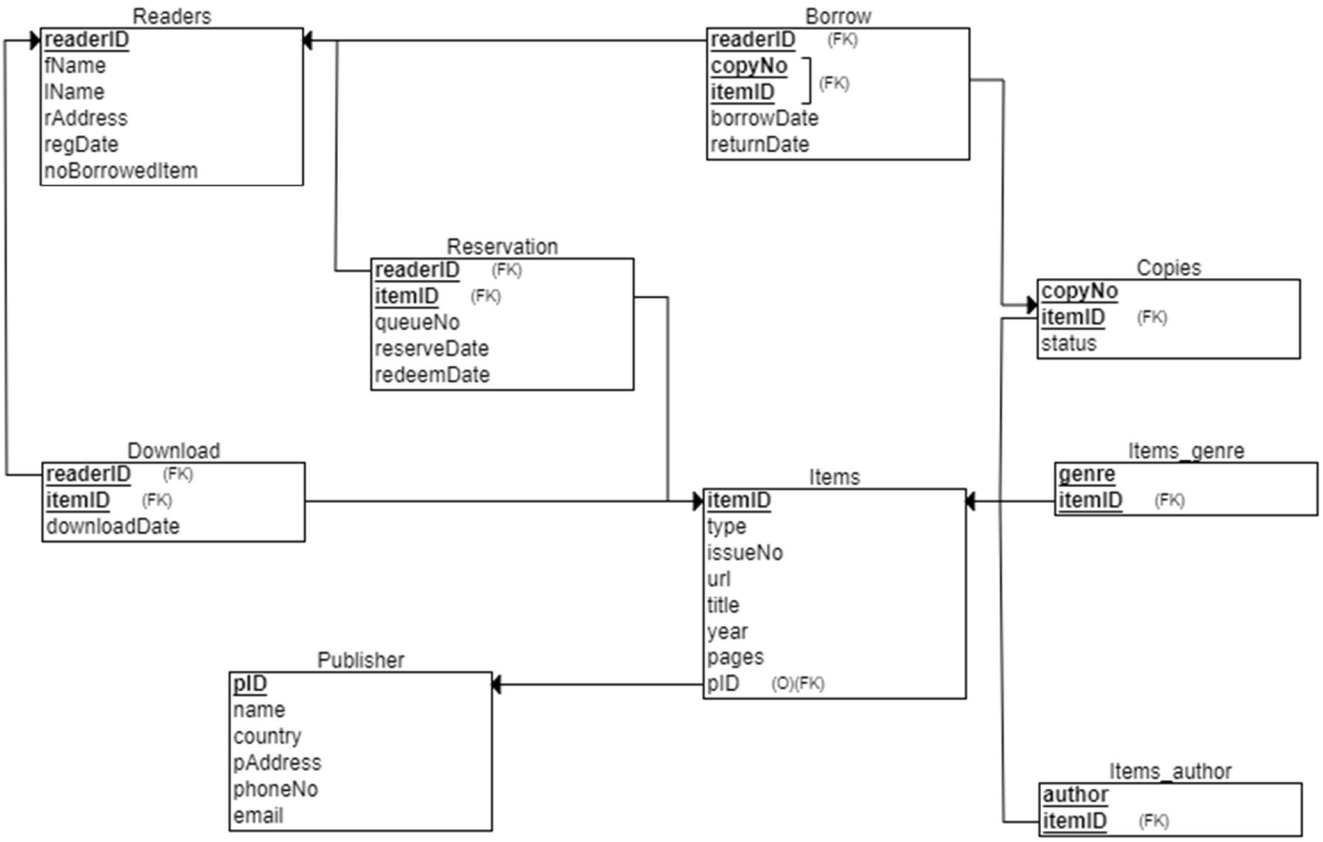

Figure 2 - Relationship schema Diagram

RELATIONSHIP SCHEMA

STRONG ENTITIES

Readers {readerID, fName, lName, rAddress, regDate, noBorrowedItem

Items {itemID, type, title, author, pages, issueNo, year, url

Publisher {pID, name, pAddress, email, phoneNo, country

WEAK ENTITIES

Copies {itemID*, copyNo, status

CANDINALITY

ENTITY- RELATIONSHIP	Candinality – Resulting Attributes
Items – Copies (Have)	**Copies** {itemID*, copyNo, status **(One to Many)**
Publisher – Items (Publish)	**Items** {itemID, type, title, pages, issueNo, year, url, pID* **(One to Many)**
Readers – Copies (Borrow)	**Borrow** {readerID*, itemID*, copyNo*, borrowDate, returnDate **(Many to Many)**
Readers – Items (Download)	**Download** { readerID*, itemID*, downloadDate **(Many to Many)**
Readers – Items (Reservation)	**Reservation** { readerID*, itemID*, queueNo, reserveDate, redeemDate **(Many to Many)**

MULTI-VALUED ATTRIBUTES

Items_Author {ItemID*, author

Items_Genre {ItemID*, genre

FINAL RELATIONAL SCHEMA

Items {itemID, type, title, pages, issueNo, year, url, pID*}

Readers {readerID, fName, lName, rAddress, regDate, noBorrowedItem}

Publisher {pID, name, pAddress, email, phoneNo, country}

Borrow {readerID*, itemID*, copyNo*, borrowDate, returnDate}

Download { readerID*, itemID*, downloadDate}

Reservation { readerID*, itemID*, queueNo, reserveDate, redeemDate}

Copies {itemID*, copyNo, status}

Items_Author {ItemID*, author}

Items_Genre {ItemID*, genre}

SQL

From relational schema the Items, Readers, Publisher, Borrow, Copies and Genre tables were created using SQL queries and sample data inserted. All tables were implemented in Oracle RDBMS.

READERS TABLE

```
CREATE TABLE READERS
  (     "READERID" VARCHAR2(55 BYTE) NOT NULL PRIMARY KEY,
        "FNAME" VARCHAR2(55 BYTE),
        "LNAME" VARCHAR2(55 BYTE),
        "RADDRESS" VARCHAR2(255 BYTE),
        "REGDATE" DATE,
        "NOBORROWEDITEM" NUMBER(2,0)
  )
INSERT ALL
 into READERS (READERID,FNAME,LNAME,RADDRESS,REGDATE,NOBORROWEDITEM) values ('R0001','Sharon','McGovern','280 Lyndon Street, Philadelphia, PA 19103',to_date('20-01-20','DD-MM-RR'),3)
 into READERS (READERID,FNAME,LNAME,RADDRESS,REGDATE,NOBORROWEDITEM) values ('R0002','Richard','Palmer','636 Yorkshire Circle Greenville, NC 27834',to_date('04-01-20','DD-MM-RR'),0)
 into READERS (READERID,FNAME,LNAME,RADDRESS,REGDATE,NOBORROWEDITEM) values ('R0003','Lewis','Deatherage','4742 Mulberry Avenue, North Little Rock, AR 72114',to_date('11-01-20','DD-MM-RR'),1)
 into READERS (READERID,FNAME,LNAME,RADDRESS,REGDATE,NOBORROWEDITEM) values ('R0004','Charles','Smith','1944 Cunningham Court, Troy, MI 48084',to_date('20-01-20','DD-MM-RR'),0)
 into READERS (READERID,FNAME,LNAME,RADDRESS,REGDATE,NOBORROWEDITEM) values ('R0005','Sabra','Cason','407 Franklin Avenue, Orlando, FL 32801',to_date('26-02-20','DD-MM-RR'),2)
 into READERS (READERID,FNAME,LNAME,RADDRESS,REGDATE,NOBORROWEDITEM) values ('R0006','Carolyn','Lowe','3480 Sharon Lane, Mount Sterling, MO 65062',to_date('26-02-20','DD-MM-RR'),0)
 into READERS (READERID,FNAME,LNAME,RADDRESS,REGDATE,NOBORROWEDITEM) values ('R0007','Marlena','Amaral','47 Pearl Street, Sacramento, CA 95814',to_date('03-02-20','DD-MM-RR'),1)
 into READERS (READERID,FNAME,LNAME,RADDRESS,REGDATE,NOBORROWEDITEM) values ('R0008','Betty','Williams','2346 Laurel Lee Maplewood, MN 55119',to_date('11-02-20','DD-MM-RR'),0)
 into READERS (READERID,FNAME,LNAME,RADDRESS,REGDATE,NOBORROWEDITEM) values ('R0009','Sharon','Selby','1583 Tetrick Road, Winter Haven, FL 33881',to_date('15-02-20','DD-MM-RR'),1)
 into READERS (READERID,FNAME,LNAME,RADDRESS,REGDATE,NOBORROWEDITEM) values ('R0010','Christopher','Anderson','1583 Tetrick Road, Winter Haven, FL 33881',to_date('15-02-20','DD-MM-RR'),1)
SELECT 1 FROM DUAL
```

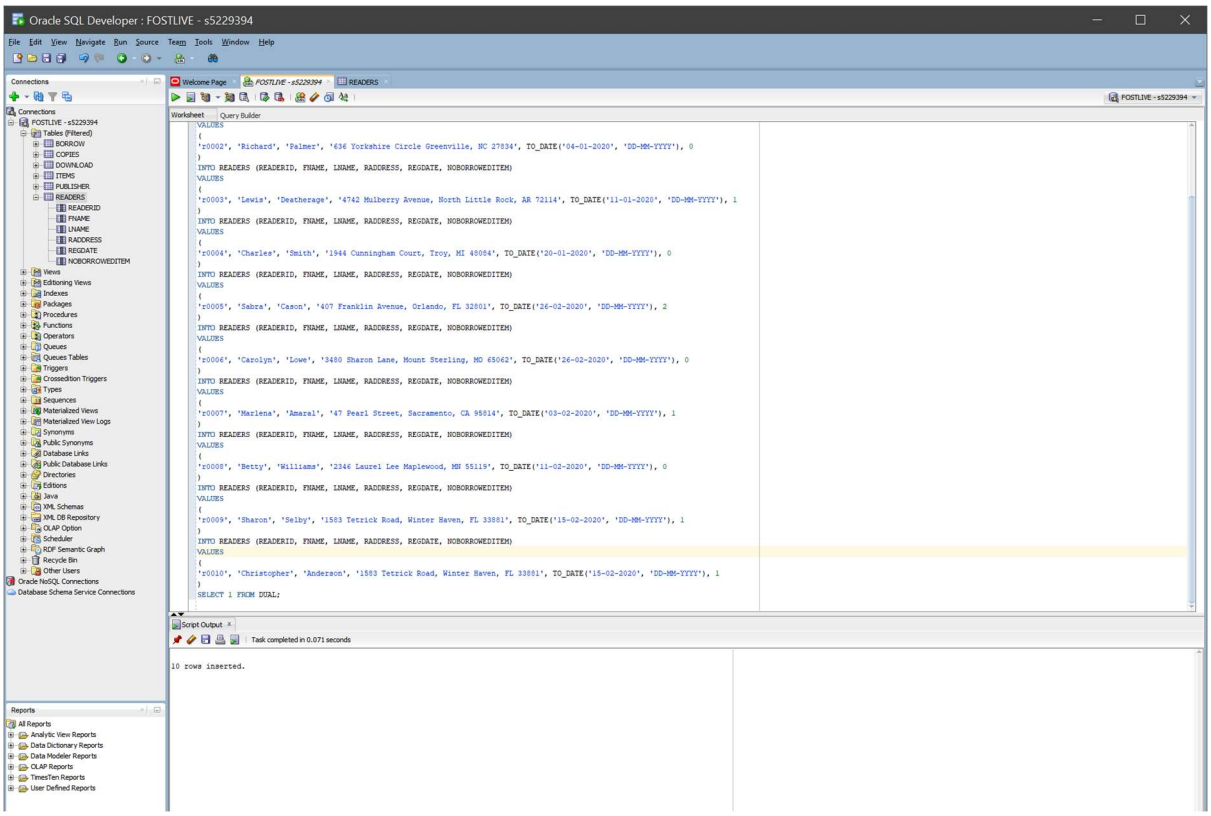

Figure 3 - Readers Table

PUBLISHER TABLE

CREATE TABLE PUBLISHER

("PID" VARCHAR2(55 BYTE) NOT NULL PRIMARY KEY,

"PNAME" VARCHAR2(55 BYTE),

"COUNTRY" VARCHAR2(55 BYTE),

"PADDRESS" VARCHAR2(255 BYTE),

"PHONENO" VARCHAR2(20 BYTE),

"EMAIL" VARCHAR2(55 BYTE)

)

INSERT ALL

into PUBLISHER (PID,PNAME,COUNTRY,PADDRESS,PHONENO,EMAIL) values ('P00001','Dutton Books','USA','8879 South Fifth Dr. Fair Lawn, NJ 07410','660-525-6598','info@duttonbooks.com')

into PUBLISHER (PID,PNAME,COUNTRY,PADDRESS,PHONENO,EMAIL) values ('P00002','Alfred A. Knopf','USA','405 6th Ave. Howard Beach, NY 11414','6646-316-1300','info@alfredakonpf.com')

into PUBLISHER (PID,PNAME,COUNTRY,PADDRESS,PHONENO,EMAIL) values ('P00003','Charles Scribner''s Sons','USA','261 Vernon Street Upper Darby, PA 19082','252-331-4418','enquiries@simonandschuster.co.uk')

into PUBLISHER (PID,PNAME,COUNTRY,PADDRESS,PHONENO,EMAIL) values ('P00004','Nan A. Talese / Doubleday','USA','6 Rockledge Rd. East Elmhurst, NY 11369','815-942-4425','ddaypub@randomhouse.com')

into PUBLISHER (PID,PNAME,COUNTRY,PADDRESS,PHONENO,EMAIL) values ('P00005','Barnes and Noble','USA','79 Sherwood Rd. Jamestown, NY 14701','719-371-1604','barnesandnoble@mail.barnesandnoble.com')

into PUBLISHER (PID,PNAME,COUNTRY,PADDRESS,PHONENO,EMAIL) values ('P00006','Harper','USA','9303 La Sierra Avenue Central Islip, NY 11722','708-849-2918','info@harper.com')

into PUBLISHER (PID,PNAME,COUNTRY,PADDRESS,PHONENO,EMAIL) values ('P00007','Thomas Egerton','USA','44 Washington Court Bridgeport, CT 06606','734-922-6235','janeaustengiftshop.co.uk')

into PUBLISHER (PID,PNAME,COUNTRY,PADDRESS,PHONENO,EMAIL) values ('P00008','Hogarth Press, England, Harcourt Brace and Co','USA','21 Green Lake St. North Brunswick, NJ 08902','617-357-3788','apps@penguinrandomhouse.co.uk')

into PUBLISHER (PID,PNAME,COUNTRY,PADDRESS,PHONENO,EMAIL) values ('P00009','J. B. Lippincott and Co.','USA','227 S. 6th Street The J. B. Lippincott Headquarters Building','917-666-7981','hello@harpercollins.com')

into PUBLISHER (PID,PNAME,COUNTRY,PADDRESS,PHONENO,EMAIL) values ('P00010','Penguin books','UK','20 Vauxhall Bridge Rd, Westminster, London SW1V 2SA','120-625-6000 ',' customersupport@penguinrandomhouse.co.uk.')

into PUBLISHER (PID,PNAME,COUNTRY,PADDRESS,PHONENO,EMAIL) values ('P00011','Elif Shafak','Turkey','Turkey','216-414-9428','info@elifsafak.com.tr')

into PUBLISHER (PID,PNAME,COUNTRY,PADDRESS,PHONENO,EMAIL) values ('P00012','Jill E. Rancourt','USA','1758 Calvin Street','443-375-1862','illERancourt@jourrapide.com')

into PUBLISHER (PID,PNAME,COUNTRY,PADDRESS,PHONENO,EMAIL) values ('P00013','Marie J. Skipper','NORWAY','Moen 174, 3948 PORSGRUNN','998-23-731','marieJSkipper@jourrapide.com')

into PUBLISHER (PID,PNAME,COUNTRY,PADDRESS,PHONENO,EMAIL) values ('P00014','Scarlett Bidwill','Finland','Kaarrostie 31, 01230 VANTAA','044-336-2683','scarlettBidwill@dayrep.com')

into PUBLISHER (PID,PNAME,COUNTRY,PADDRESS,PHONENO,EMAIL) values ('P00015','Matilda Weindorfer','Canada','4910 rue des Églises Est, Cheneville, QC J0V 1E0','819-428-0909','matildaWeindorfer@rhyta.com')

into PUBLISHER (PID,PNAME,COUNTRY,PADDRESS,PHONENO,EMAIL) values ('P00016','Blake Ruwolt','Canada','49 Neilson Avenue, Toronto, ON M1M 1V1','416-269-5145','blakeRuwolt@dayrep.com')

into PUBLISHER (PID,PNAME,COUNTRY,PADDRESS,PHONENO,EMAIL) values ('P00017','Hudson De Satg','USA','3543 Carson Street San Diego, CA 92103','858-947-0401','hudsonDeSatg@jourrapide.com')

into PUBLISHER (PID,PNAME,COUNTRY,PADDRESS,PHONENO,EMAIL) values ('P00018','Andrew Carnarvong','USA','3051 Hardman Road Brattleboro, VT 05301','802-812-7335','andrewCarnarvon@rhyta.com')

into PUBLISHER (PID,PNAME,COUNTRY,PADDRESS,PHONENO,EMAIL) values ('P00019','Ellie Akehurst','UK','32 Tadcaster Rd, PITCAPLE AB51 1QF','077-764-4730','ellieAkehurst@rhyta.com')

into PUBLISHER (PID,PNAME,COUNTRY,PADDRESS,PHONENO,EMAIL) values ('P00020','Henry Tully','SWEDEN','Idvägen 90, 360 73 LENHOVDA','0474-764-9764','henryTully@dayrep.com')

into PUBLISHER (PID,PNAME,COUNTRY,PADDRESS,PHONENO,EMAIL) values ('P00021','Dean Fereday','USA','4393 Coleman Avenue, Vista, CA 92083','760-732-7972','deanFereday@dayrep.com')

SELECT 1 FROM DUAL

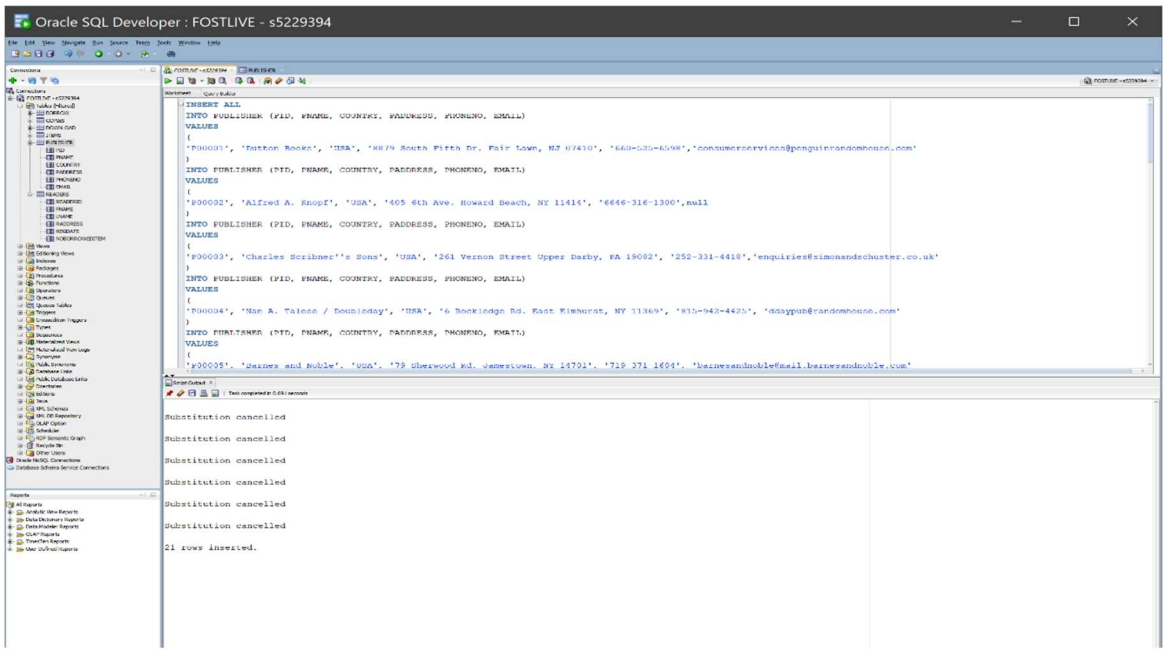

Figure 4 - Publisher Table

ITEMS TABLE

CREATE TABLE ITEMS

("ITEMID" VARCHAR2(55 BYTE) NOT NULL PRIMARY KEY,

"ITYPE" VARCHAR2(25 BYTE),

"ISSUENO" VARCHAR2(25 BYTE),

"URL" VARCHAR2(255 BYTE),

"TITLE" VARCHAR2(55 BYTE),

"IYEAR" NUMBER(4,0),

"IPAGES" NUMBER(4,0),

"PID" VARCHAR2(55 BYTE),

FOREIGN KEY (PID) REFERENCES PUBLISHER(PID))

INSERT ALL

into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values ('LB001','book',null,null,'The Fault In our stars',2012,313,'P00001')

into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values ('LB002','book',null,null,'Belowed',1987,324,'P00002')

into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values ('LB003','book',null,null,'The Great Gatsby',1925,218,'P00003')

into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values ('LB004','book',null,null,'The Testaments',2019,432,'P00004')

into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values ('LB005','book',null,null,'Wuthering heights',2005,400,'P00005')

into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values ('LB006','book',null,null,'Pride And Prejudice',2004,432,'P00005')

into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values ('LB007','book',null,null,'Sense And Sensibility',1995,220,'P00007')

into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values ('LB008','book',null,null,'Sapiens',2011,443,'P00006')

into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values ('LB009','book',null,null,'For Whom The bells Toll',1940,480,'P00003')

into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values ('LB010','book',null,null,'A Room Of One''s Own',1929,172,'P00008')

SELECT 1 FROM DUAL

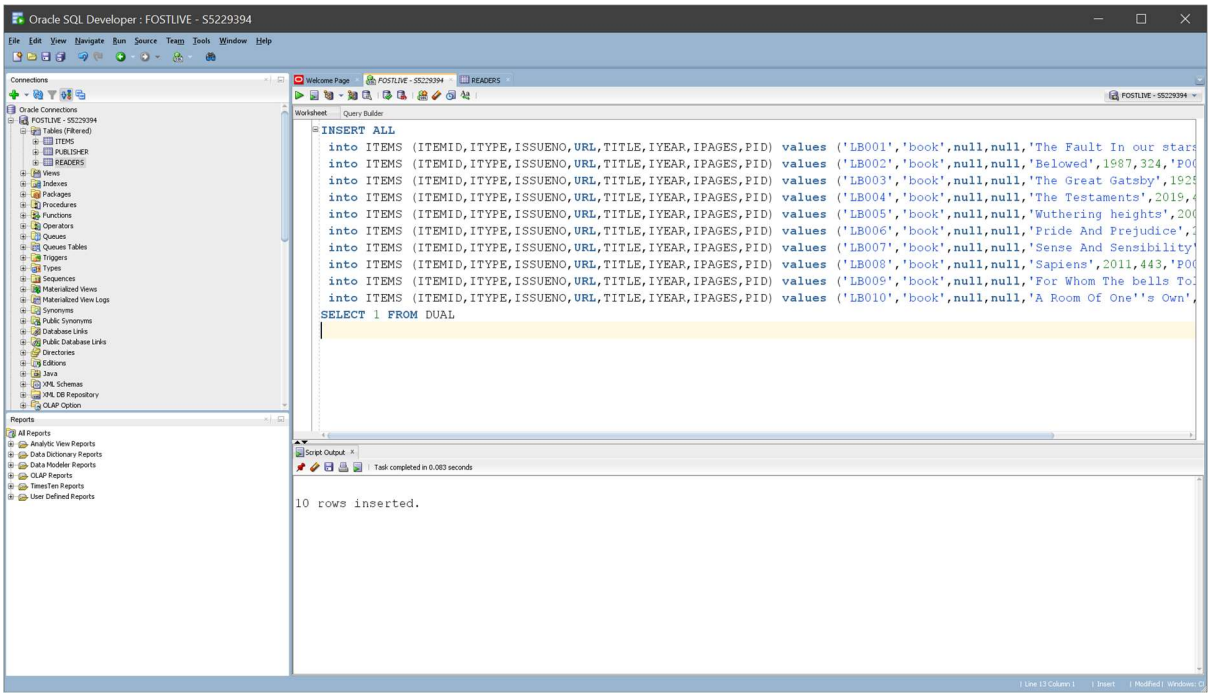

Figure 5 - Inserting Books into Items Table

```
INSERT ALL
 into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values ('LB011','magazine','5836597',null,'Times',null,null,'P00012')
 into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values ('LB012','magazine','451962',null,'US Weekly',null,null,'P00013')
 into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values ('LB013','magazine','a27020796',null,'Country Living',null,null,'P00014')
 into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values ('LB014','magazine','20200501',null,'Vogue',null,null,'P00015')
 into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values ('LB015','magazine','a31005669',null,'Elle!',null,null,'P00016')
 into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values ('LB016','magazine','949908',null,'Rolling Stone',null,null,'P00017')
 into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values ('LB017','magazine','2020518',null,'The New Yorker',null,null,'P00018')
 into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values ('LB018','magazine','202000401',null,'Vanity Fair',null,null,'P00019')
 into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values ('LB019','magazine','20520353',null,'People',null,null,'P00020')
 into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values ('LB020','magazine','4142020',null,'Wired',null,null,'P00021')
SELECT 1 FROM DUAL
```

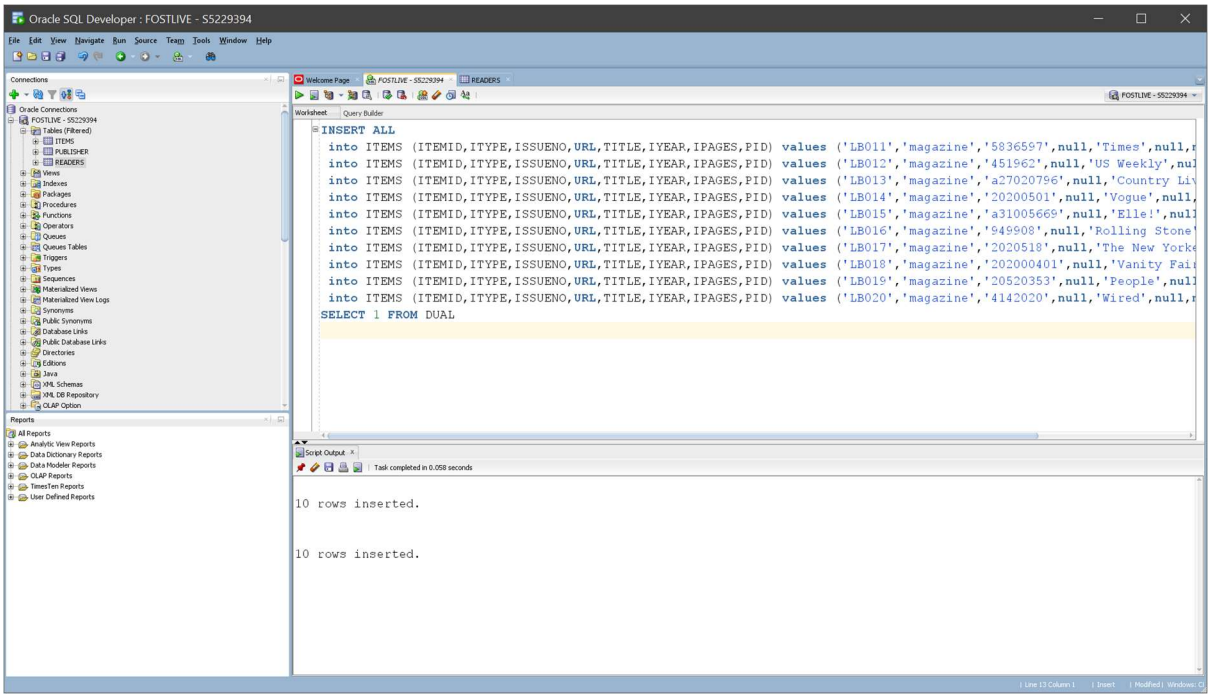

Figure 6- Inserting Magazines into Items Table

```
INSERT ALL

 into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values
('LB021','ebook',null,'https://docs.google.com/viewer?a=v\&\pid=sites\&\srcid=YW5udXJpc2xhbWljc2Nob29sLm9yZ3xzaXN0ZXIta2F0ZWx5bnxneDo2NjVmZmE1NzNjNjc4NWM','To Kill a Mockingbird',1960,281,'P00009')

 into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values
('LB022','ebook',null,'https://www.penguin.co.uk/books/111/1112622/dancing-the-charleston/9780440871675.html','Dancing the Charleston',2019,448,'P00010')

 into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values
('LB023','ebook',null,'https://www.penguin.co.uk/books/177/177356/the-forty-rules-of-love/9780241972939.html','The Forty Rules of Love',2009,354,'P00011')

 into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values
('LB024','ebook',null,'https://www.penguin.co.uk/books/1114146/killing-commendatore/9781787300194.html','Killing Commendatore',2018,512,'P00010')

 into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values
('LB025','ebook',null,'https://www.penguin.co.uk/books/298975/the-cut-out-girl/9780241978719.html','The Cut Out Girl',2018,288,'P00010')

 into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values
('LB026','ebook',null,'https://www.penguin.co.uk/books/290399/the-giver-of-stars/9780718183202.html','The Giver Of Stars',2019,448,'P00010')

 into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values
('LB027','ebook',null,'https://www.penguin.co.uk/books/1099258/the-handmaid-s-tale/9780224101936.html','The Handmaid''s Tale',2019,240,'P00010')

 into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values
('LB028','ebook',null,'https://www.penguin.co.uk/books/1099258/the-handmaid-s-tale/9780224101936.html','Black Sun',2019,336,'P00010')

 into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values
('LB029','ebook',null,'https://www.penguin.co.uk/books/309231/airhead/9781405938358.html','Airhead',2019,400,'P00010')

 into ITEMS (ITEMID,ITYPE,ISSUENO,URL,TITLE,IYEAR,IPAGES,PID) values
('LB030','ebook',null,'https://www.penguin.co.uk/books/1113984/origin/9780552174169.html','Origin',2018,560,'P00010')

SELECT 1 FROM DUAL
```

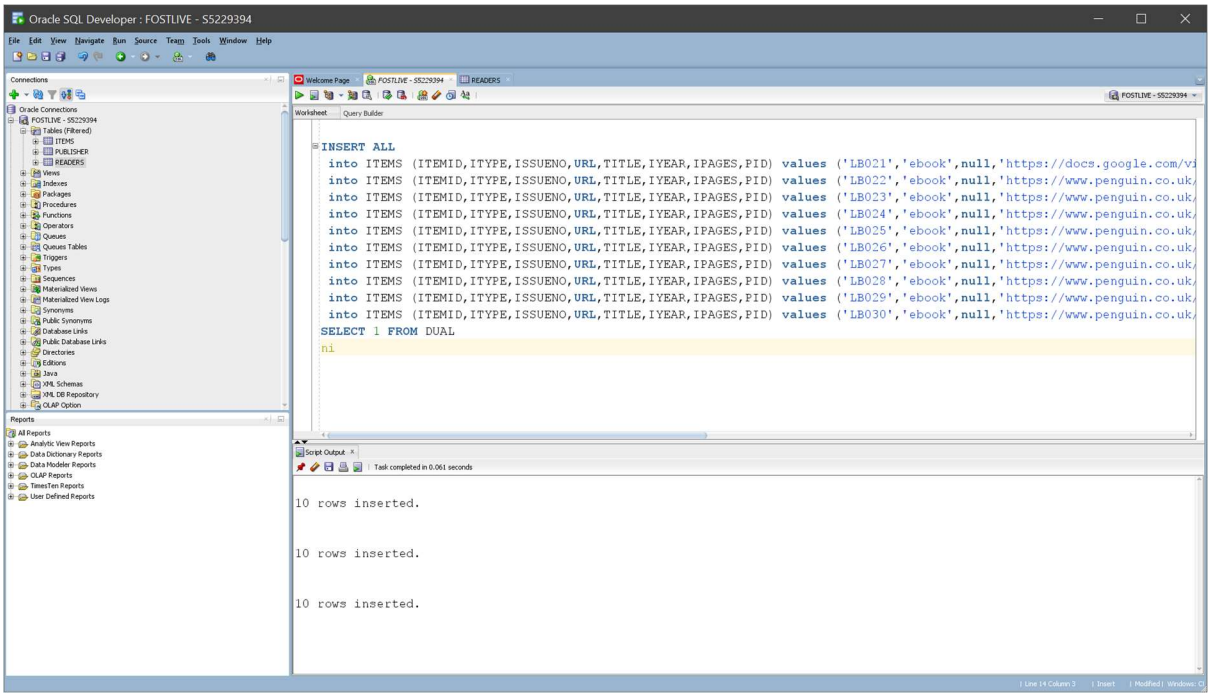

Figure 7- Inserting E-Books into Items Table

COPIES TABLE

```
CREATE TABLE COPIES
  (     "COPYNO" VARCHAR2(55 BYTE),
        "ITEMID" VARCHAR2(55 BYTE),
        "STATUS" NUMBER(1,0)
        PRIMARY KEY(COPYNO, ITEMID),
        FOREIGN KEY(ITEMID) REFERENCES ITEMS(ITEMID)   )

INSERT ALL
 into COPY (COPYNO,ITEMID,STATUS) values ('C00001','LB001',1)
 into COPY (COPYNO,ITEMID,STATUS) values ('C00002','LB001',0)
 into COPY (COPYNO,ITEMID,STATUS) values ('C00003','LB001',1)
 into COPY (COPYNO,ITEMID,STATUS) values ('C00004','LB002',1)
 into COPY (COPYNO,ITEMID,STATUS) values ('C00005','LB002',1)
 into COPY (COPYNO,ITEMID,STATUS) values ('C00006','LB003',1)
 into COPY (COPYNO,ITEMID,STATUS) values ('C00007','LB003',1)
 into COPY (COPYNO,ITEMID,STATUS) values ('C00008','LB003',1)
 into COPY (COPYNO,ITEMID,STATUS) values ('C00009','LB003',0)
 into COPY (COPYNO,ITEMID,STATUS) values ('C00010','LB004',1)
```

into COPY (COPYNO,ITEMID,STATUS) values ('C00011','LB004',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00012','LB004',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00013','LB004',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00014','LB004',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00015','LB005',0)

into COPY (COPYNO,ITEMID,STATUS) values ('C00016','LB005',0)

into COPY (COPYNO,ITEMID,STATUS) values ('C00017','LB006',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00018','LB006',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00019','LB006',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00020','LB007',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00021','LB007',0)

into COPY (COPYNO,ITEMID,STATUS) values ('C00022','LB008',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00023','LB008',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00024','LB008',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00025','LB009',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00026','LB009',0)

into COPY (COPYNO,ITEMID,STATUS) values ('C00027','LB010',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00028','LB010',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00029','LB011',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00030','LB011',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00031','LB012',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00032','LB012',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00033','LB013',0)

into COPY (COPYNO,ITEMID,STATUS) values ('C00034','LB013',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00035','LB014',0)

into COPY (COPYNO,ITEMID,STATUS) values ('C00036','LB014',0)

into COPY (COPYNO,ITEMID,STATUS) values ('C00037','LB015',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00038','LB015',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00039','LB016',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00040','LB016',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00041','LB017',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00042','LB017',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00043','LB018',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00044','LB018',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00045','LB019',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00046','LB019',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00047','LB020',1)

into COPY (COPYNO,ITEMID,STATUS) values ('C00048','LB020',1)

SELECT 1 FROM DUAL

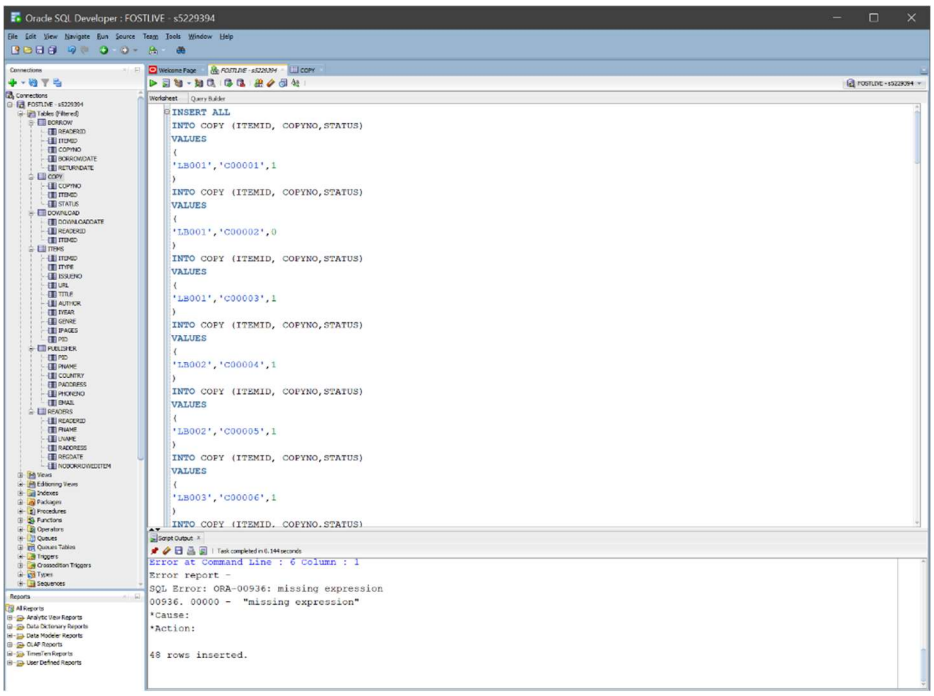

Figure 8-Copies Table

BORROW TABLE

CREATE TABLE BORROW

("READERID" VARCHAR2(55 BYTE) NOT NULL,

"COPYNO" VARCHAR2(55 BYTE),

"ITEMID" VARCHAR2(55 BYTE) NOT NULL,

"BORROWDATE" DATE,

"RETURNDATE" DATE,

PRIMARY KEY (READERID,COPYNO,ITEMID),

FOREIGN KEY (READERID) REFERENCES READERS(READERID),

FOREIGN KEY (COPYNO,ITEMID) REFERENCES COPIES(COPYNO,ITEMID)

)

```
INSERT ALL
 into BORROW (READERID,COPYNO,ITEMID,BORROWDATE,RETURNDATE) values
('R0005','C00002','LB001',to_date('10-05-20','DD-MM-RR'),to_date('23-05-20','DD-MM-RR'))
 into BORROW (READERID,COPYNO,ITEMID,BORROWDATE,RETURNDATE) values
('R0005','C00009','LB003',to_date('12-05-20','DD-MM-RR'),to_date('25-05-20','DD-MM-RR'))
 into BORROW (READERID,COPYNO,ITEMID,BORROWDATE,RETURNDATE) values
('R0009','C00015','LB005',to_date('15-05-20','DD-MM-RR'),to_date('28-05-20','DD-MM-RR'))
 into BORROW (READERID,COPYNO,ITEMID,BORROWDATE,RETURNDATE) values
('R0010','C00016','LB005',to_date('10-05-20','DD-MM-RR'),to_date('23-05-20','DD-MM-RR'))
 into BORROW (READERID,COPYNO,ITEMID,BORROWDATE,RETURNDATE) values
('R0001','C00021','LB007',to_date('13-05-20','DD-MM-RR'),to_date('26-05-20','DD-MM-RR'))
 into BORROW (READERID,COPYNO,ITEMID,BORROWDATE,RETURNDATE) values
('R0001','C00026','LB009',to_date('13-05-20','DD-MM-RR'),to_date('26-05-20','DD-MM-RR'))
 into BORROW (READERID,COPYNO,ITEMID,BORROWDATE,RETURNDATE) values
('R0001','C00033','LB013',to_date('18-05-20','DD-MM-RR'),to_date('31-05-20','DD-MM-RR'))
 into BORROW (READERID,COPYNO,ITEMID,BORROWDATE,RETURNDATE) values
('R0003','C00035','LB014',to_date('24-05-20','DD-MM-RR'),to_date('06-06-20','DD-MM-RR'))
 into BORROW (READERID,COPYNO,ITEMID,BORROWDATE,RETURNDATE) values
('R0007','C00036','LB014',to_date('20-05-20','DD-MM-RR'),to_date('02-06-20','DD-MM-RR'))SELECT
1 FROM DUAL
```

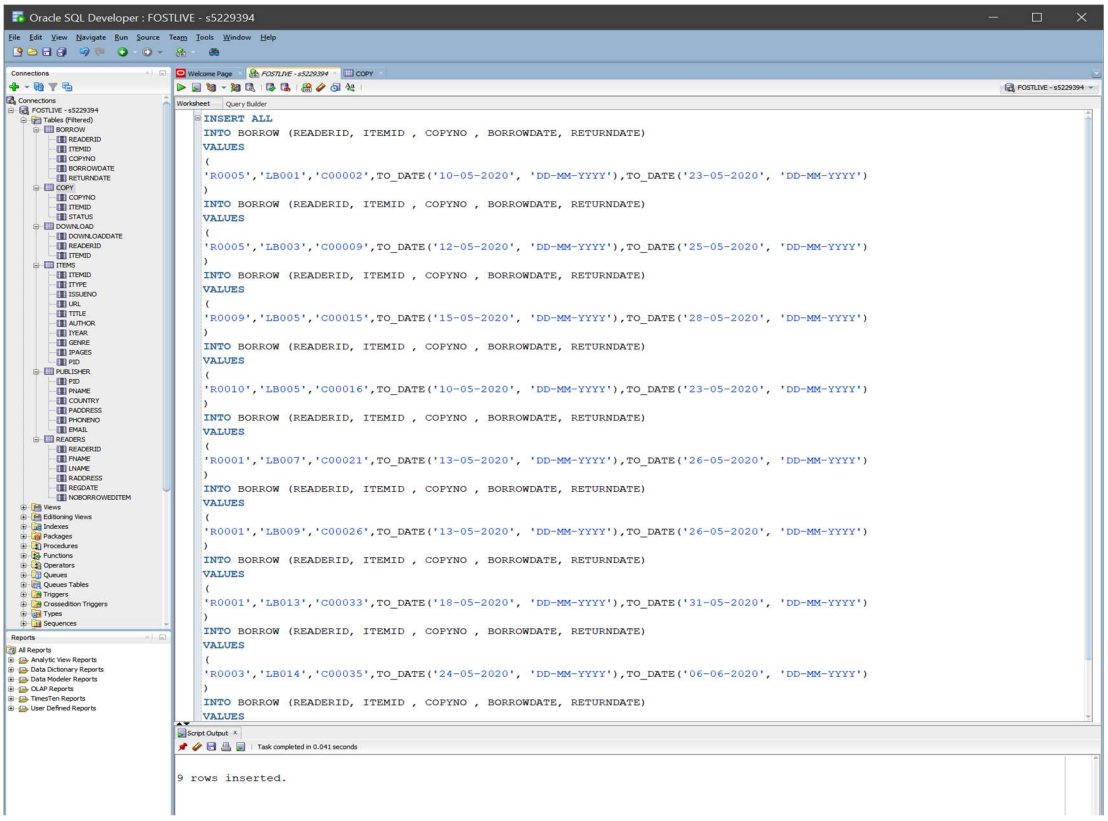

Figure 9 - Borrow Table

GENRE TABLE

```
CREATE TABLE  GENRE
  (       "GENRE" VARCHAR2(55 BYTE),
          "ITEMID" VARCHAR2(55 BYTE),
          PRIMARY KEY (GENRE, ITEMID),
          FOREIGN KEY (ITEMID) REFERENCES ITEMS(ITEMID)
  )
INSERT ALL
 into GENRE (GENRE,ITEMID) values ('Americal Literature','LB002')
 into GENRE (GENRE,ITEMID) values ('Beauty','LB015')
 into GENRE (GENRE,ITEMID) values ('Beauty','LB018')
 into GENRE (GENRE,ITEMID) values ('Biography','LB025')
 into GENRE (GENRE,ITEMID) values ('Crime','LB030')
 into GENRE (GENRE,ITEMID) values ('Drama','LB006')
 into GENRE (GENRE,ITEMID) values ('Drama','LB029')
 into GENRE (GENRE,ITEMID) values ('Essay','LB010')
 into GENRE (GENRE,ITEMID) values ('Fashion','LB014')
 into GENRE (GENRE,ITEMID) values ('Fashion','LB015')
 into GENRE (GENRE,ITEMID) values ('Fiction','LB001')
 into GENRE (GENRE,ITEMID) values ('Fiction','LB003')
 into GENRE (GENRE,ITEMID) values ('Fiction','LB004')
 into GENRE (GENRE,ITEMID) values ('Fiction','LB005')
 into GENRE (GENRE,ITEMID) values ('Fiction','LB010')
 into GENRE (GENRE,ITEMID) values ('Fiction','LB022')
 into GENRE (GENRE,ITEMID) values ('Fiction','LB023')
 into GENRE (GENRE,ITEMID) values ('Fiction','LB024')
 into GENRE (GENRE,ITEMID) values ('Fiction','LB026')
 into GENRE (GENRE,ITEMID) values ('Fiction','LB030')
 into GENRE (GENRE,ITEMID) values ('Gothic','LB005')
 into GENRE (GENRE,ITEMID) values ('Gothic','LB021')
 into GENRE (GENRE,ITEMID) values ('Health','LB012')
```

```
 into GENRE (GENRE,ITEMID) values ('Health','LB017')
 into GENRE (GENRE,ITEMID) values ('Historical','LB022')
 into GENRE (GENRE,ITEMID) values ('Lifestyle','LB011')
 into GENRE (GENRE,ITEMID) values ('Lifestyle','LB013')
 into GENRE (GENRE,ITEMID) values ('Lifestyle','LB019')
 into GENRE (GENRE,ITEMID) values ('Lifestyle','LB020')
 into GENRE (GENRE,ITEMID) values ('Magical Realism','LB024')
 into GENRE (GENRE,ITEMID) values ('Mystery','LB028')
 into GENRE (GENRE,ITEMID) values ('Mystery','LB030')
 into GENRE (GENRE,ITEMID) values ('Non-Fiction','LB008')
 into GENRE (GENRE,ITEMID) values ('Romance','LB007')
 into GENRE (GENRE,ITEMID) values ('Romance','LB026')
 into GENRE (GENRE,ITEMID) values ('Satire','LB006')
 into GENRE (GENRE,ITEMID) values ('Southern','LB021')
 into GENRE (GENRE,ITEMID) values ('Tragedy','LB003')
 into GENRE (GENRE,ITEMID) values ('Tragedy','LB005')
 into GENRE (GENRE,ITEMID) values ('Tragedy','LB027')
 into GENRE (GENRE,ITEMID) values ('War Story','LB009')
 into GENRE (GENRE,ITEMID) values ('Young Adult','LB001')
 into GENRE (GENRE,ITEMID) values ('fiction','LB027')
 into GENRE (GENRE,ITEMID) values ('lifestyle','LB016')
SELECT 1 FROM DUAL
```

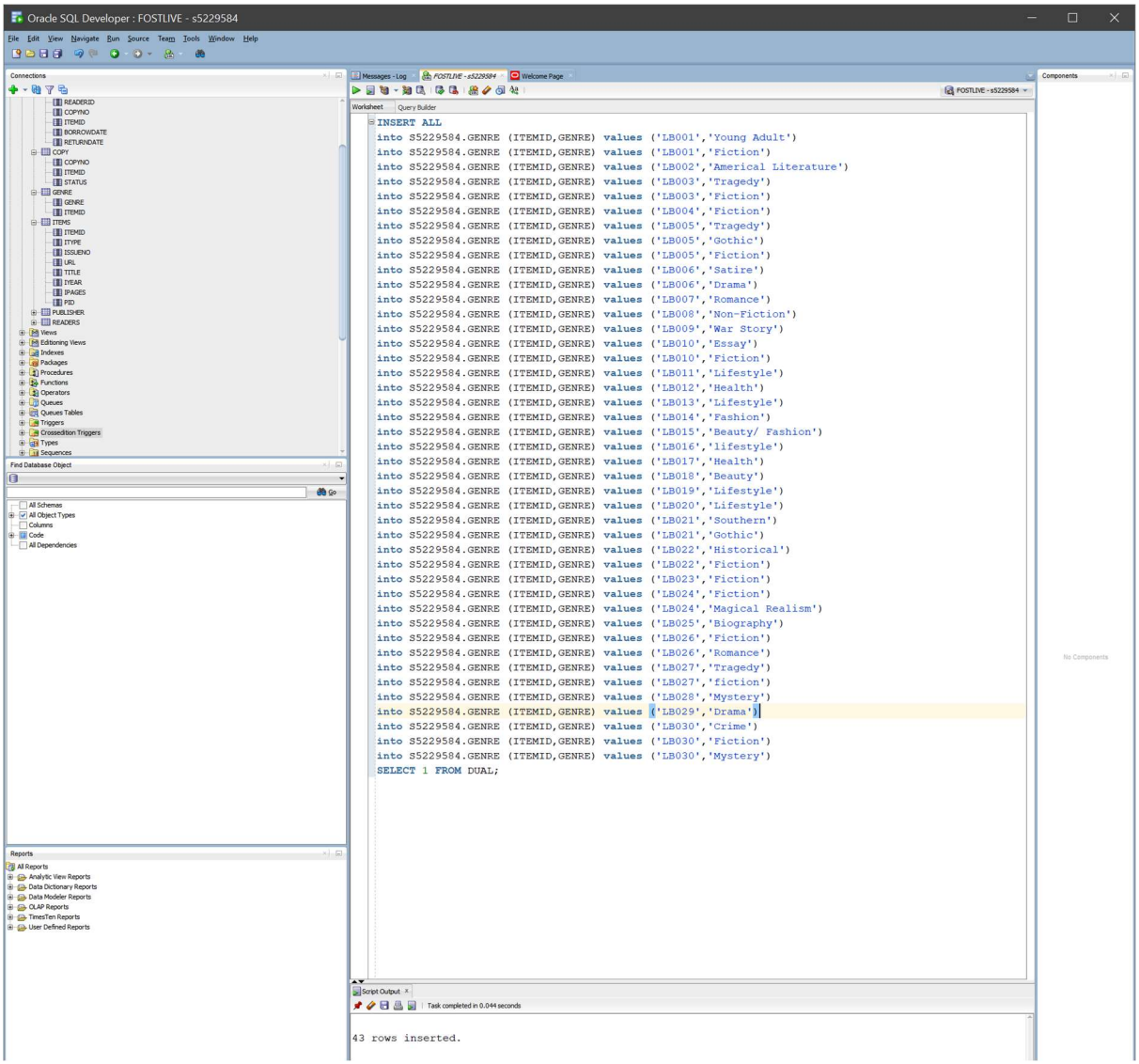

Figure 10 - Genre Table

The items table holds information about books, magazines and ebooks own by the library, the Readers table holds required information of registered users. The borrow table stores the borrow and return date of borrowed items, it also stores the unique item identifier and the unique user identifier. An item can have multiple copies, to treat each copy as unique, the Copies table was created to store each copy with a unique copy identifier. The Publisher table stores details of each publishers, and the Genre table store genre against each item.
The implementation of these table will help perform advanced searches which also assists in decision making.

SQL QUERIES (FIVE USE CASES)

1. **Items withdrawn by each user and dates they are due**

```
with t1 as
(
select readerid,
listagg(itemid,' ') within group (order by readerid) as itemids,
listagg(returndate,' ') within group (order by readerid) as returndates
```

from borrow

group by readerid

)

select t1.readerid,fname,lname,t1.itemids,t1.returndates from readers

join t1 on readers.readerid=t1.readerid

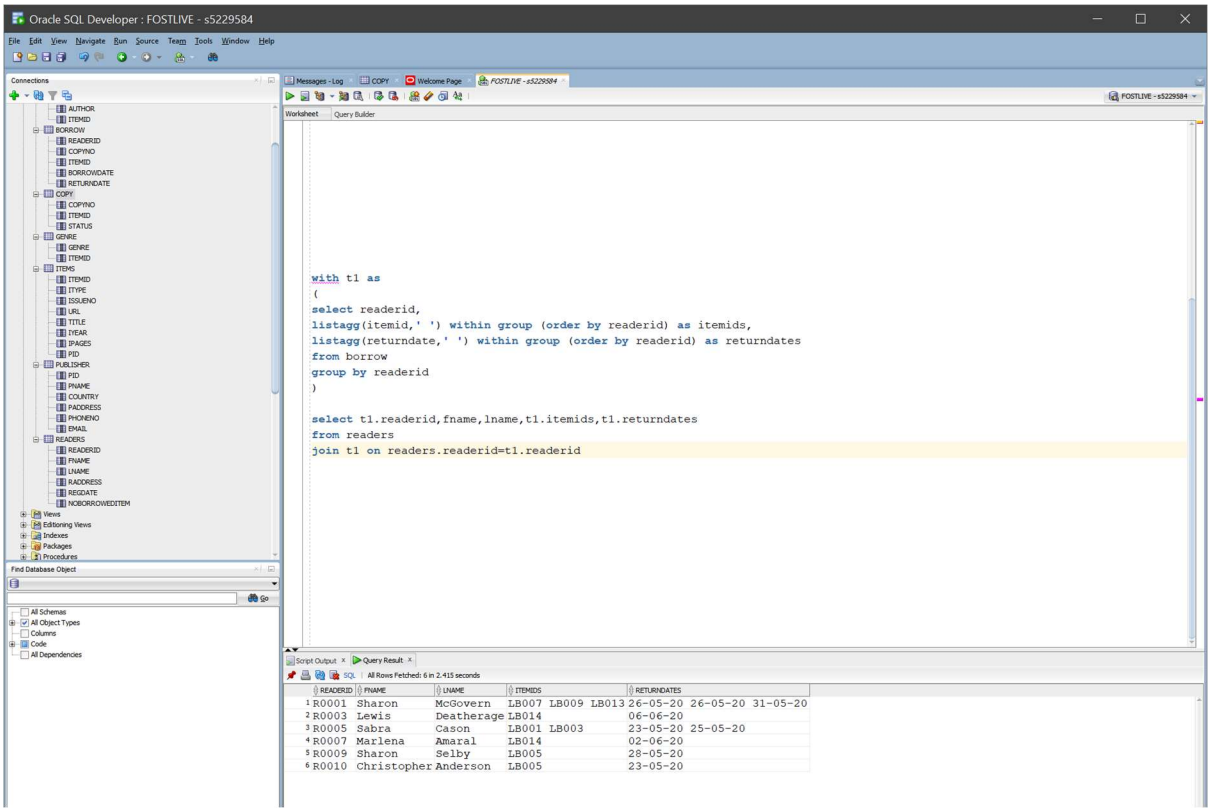

Figure 11- Item withdrawn by each user

2. **Users with more than 1 item overdue.**

with t1 as(

select readerid, count(readerid)as noofbooksdue

from (

select readerid,returndate

from borrow

where returndate< current_date

)

group by readerid

having count(readerid)>1)

select readers.readerid,readers.fname,readers.lname,noofbooksdue from readers

inner join t1 on readers.readerid=t1.readerid

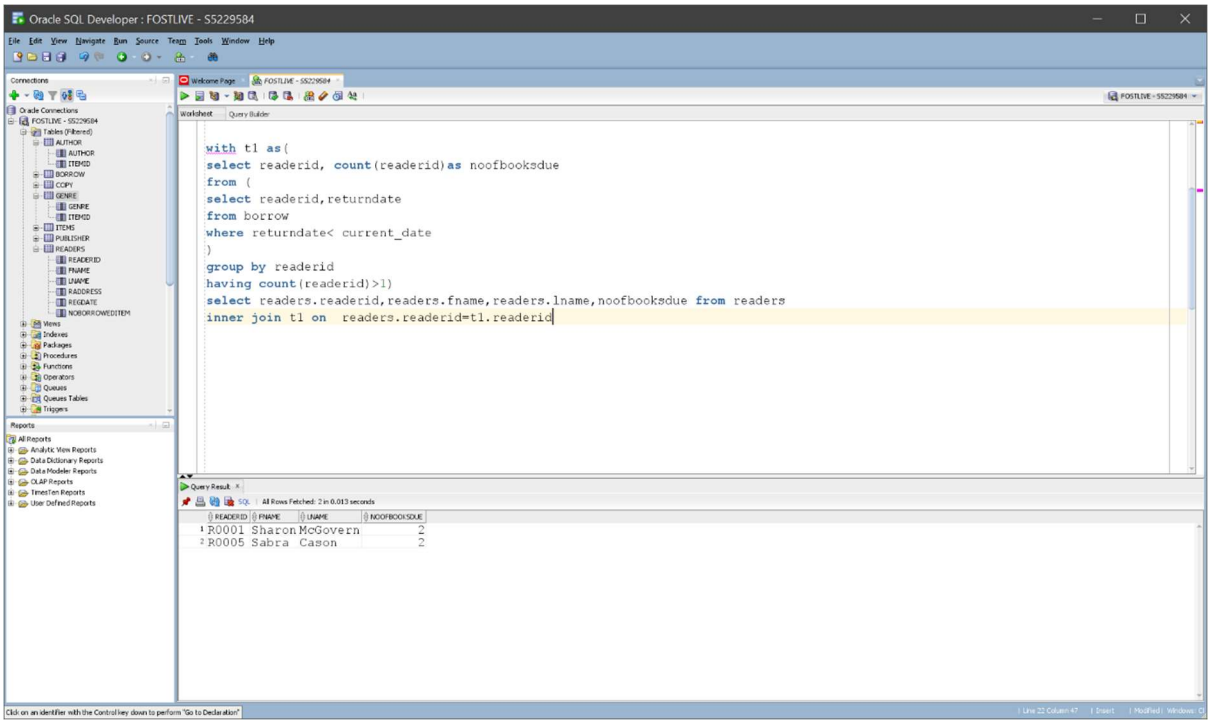

Figure 12 - Users with more than 1 item overdue

3. Get the total number of copies for each item and the available copies for withdrawal

```
with t1 as(
select itemid, count(itemid)as totalitems from copy
group by  itemid order by itemid
),
t2 as(
select itemid,count(status)as avitems from copy
where status='1' group by  itemid
)
select items.itemid, items.title,t1.totalitems,NVL(t2.avitems, '0') as availableunits
from items
join t1 on items.itemid= t1.itemid
full join t2 on items.itemid= t2.itemid
order by itemid
```

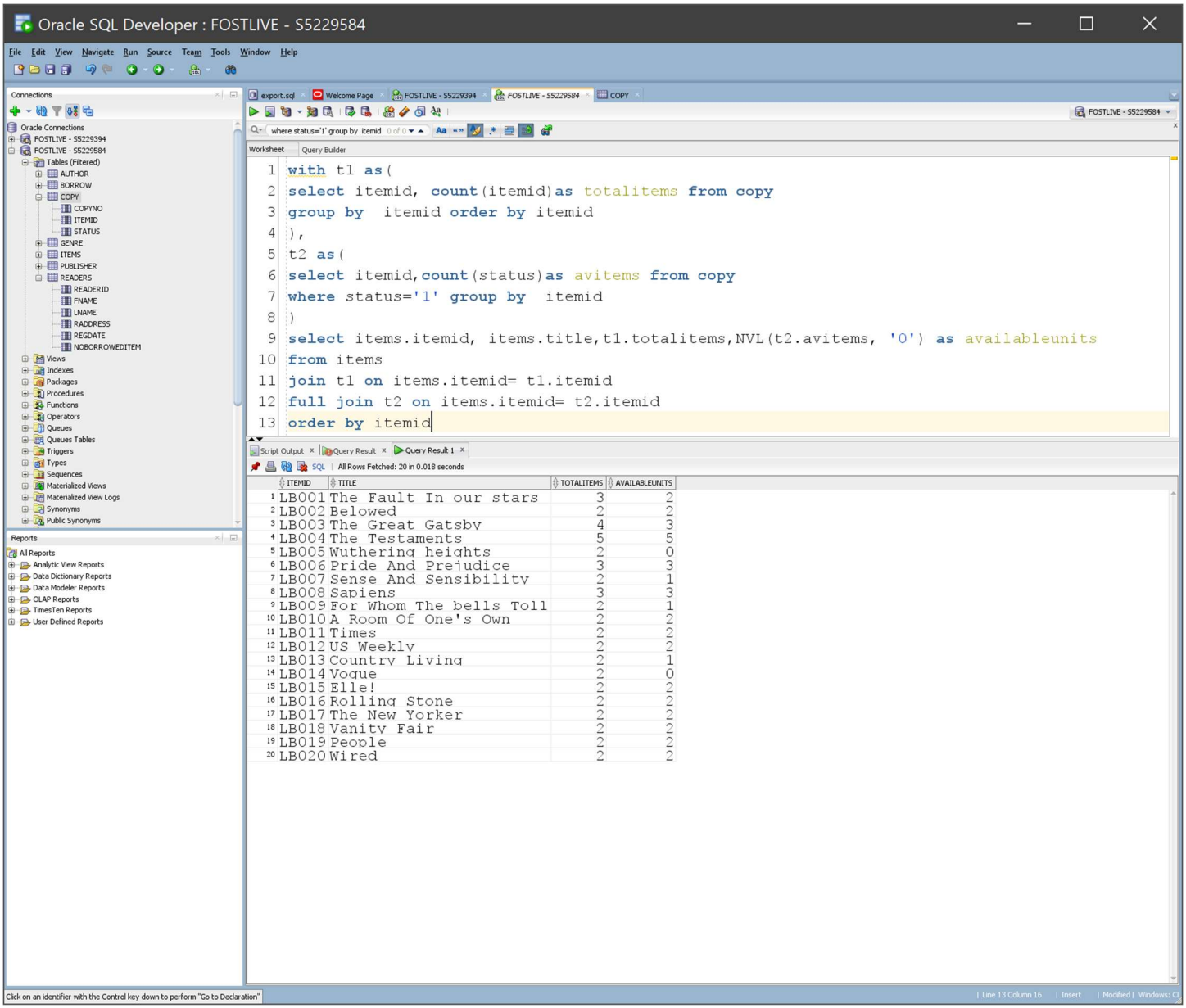

Figure 13 - Total number of Item copies and available copies

4. **Number of books per publisher in the database sorted from highest to lowest**

```
with t1 as(
select pid,count(pid) as noofbooks from items
group by pid
)
select publisher.pid,publisher.pname,t1.noofbooks from publisher
inner join t1 on publisher.pid = t1.pid order by t1.noofbooks desc
```

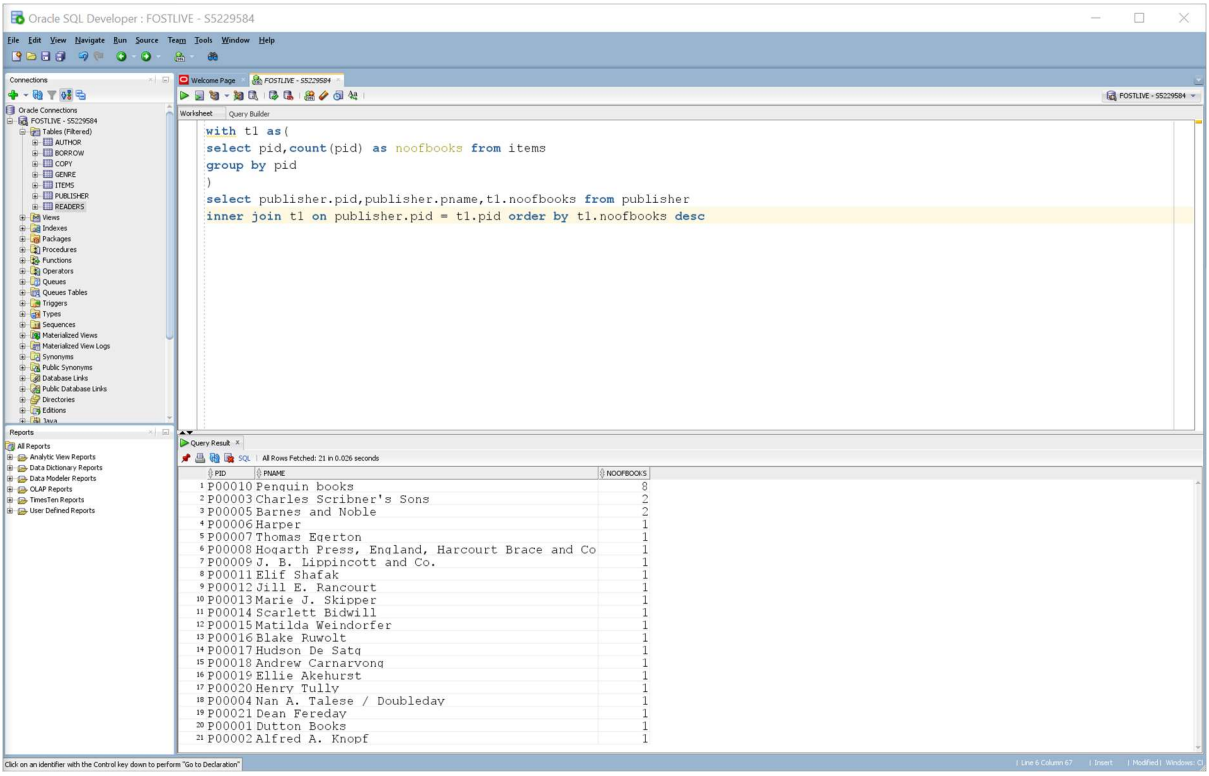

Figure 14 - List of Pubishers with numbers of books in the database

5. **Most common genre in the database and lists all items that match the genre**

```
with t1 as (select genre
from genre
group by genre
order by count(*) desc
fetch next 1 rows only)
select genre.itemid, items.title, genre.genre from genre
inner join t1 on genre.genre=t1.genre
inner join items on genre.itemid=items.itemid
```

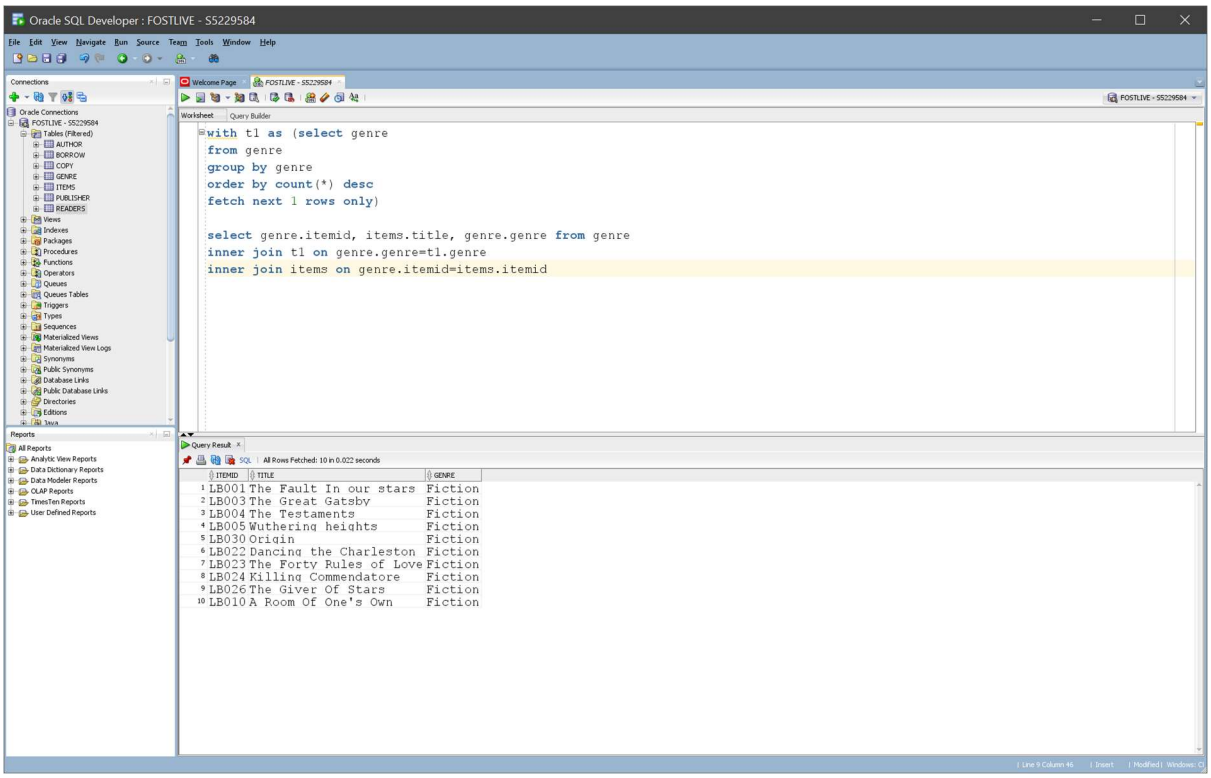

Figure 15 - Most common genre in the database and list of items that match the genre

NEO4J

The Readers, Items, Copies and Publisher Entities from the ERD were implement for this database. These allow for the investigation of how well Neo4j Graph Database handles relationship between multiple tables and how user friendly and intuitive it is.

NODES	
LABELS	**ATTRIBUTES**
Reader	{readerID, fName, lName, rAddress, regDate, noBorrowedItem}
Item	{itemID, type, title, pages, issueNo, year, url}
Copy	{copyid, name}
Publisher	{pID, pName, pAddress, email, phoneNo, country}

EDGES		
LABELS	**ATTRIBUTES**	**SOURCE TO DESTINATION**
:BORROWED	{borrowDate, returnDate}	Reader to Item
:BELONGSTO		Copy to Item
:PUBLISHED		Publisher to Item

CREATE GRAPH DATABASE

CREATING NODES

READER GRAPH DATABASE

CREATE (reader1:Reader {readerId:'r0001', fName: 'Sharon', lName:'McGovern', rAddress:'280 Lyndon Street Philadelphia PA 19103', regDate:date('2020-01-02'), noBorrowedItems:3})

CREATE (reader2:Reader {readerId:'r0002', fName: 'Richard', lName:'Palmer', rAddress:'636 Yorkshire Circle Greenvill NC 27834', regDate:date('2020-01-04'), noBorrowedItems:0})
CREATE (reader3:Reader {readerId:'r0003', fName: 'Lewis', lName:'Deatherage', rAddress:'4742 Mulberry Avenue North Little Rock AR 72114', regDate:date('2020-01-11'), noBorrowedItems:1})
CREATE (reader4:Reader {readerId:'r0004', fName: 'Charles', lName:'Smith', rAddress:'1944 Cunningham Court Troy MI 48084', regDate:date('2020-01-20'), noBorrowedItems:0})
CREATE (reader5:Reader {readerId:'r0005', fName: 'Sabra', lName:'Cason', rAddress:'407 Franklin Avenue Orlando FL 32801', regDate:date('2020-02-26'), noBorrowedItems:2})
CREATE (reader6:Reader {readerId:'r0006', fName: 'Carolyn', lName:'Lowe', rAddress:'3480 Sharon Lane Mount Sterling MO 65062', regDate:date('2020-02-26'), noBorrowedItems:0})
CREATE (reader7:Reader {readerId:'r0007', fName: 'Marlena', lName:'Amaral', rAddress:'47 Pearl Street Sacramento CA 95814', regDate:date('2020-02-03'), noBorrowedItems:1})
CREATE (reader8:Reader {readerId:'r0008', fName: 'Betty', lName:'Williams', rAddress:'2346 Laurel Lee Maplewood MN 55119', regDate:date('2020-02-11'), noBorrowedItems:0})
CREATE (reader9:Reader {readerId:'r0009', fName: 'Sharon', lName:'Shelby', rAddress:'1583 Tetrick Road Winter Haven FL 33881', regDate:date('2020-02-11'), noBorrowedItems:1})
CREATE (reader10:Reader {readerId:'r0010', fName: 'Christopher', lName:'Anderson', rAddress:'1303 North Avenue Omaha NE 68144FL 33881', regDate:date('2020-02-21'), noBorrowedItems:1})

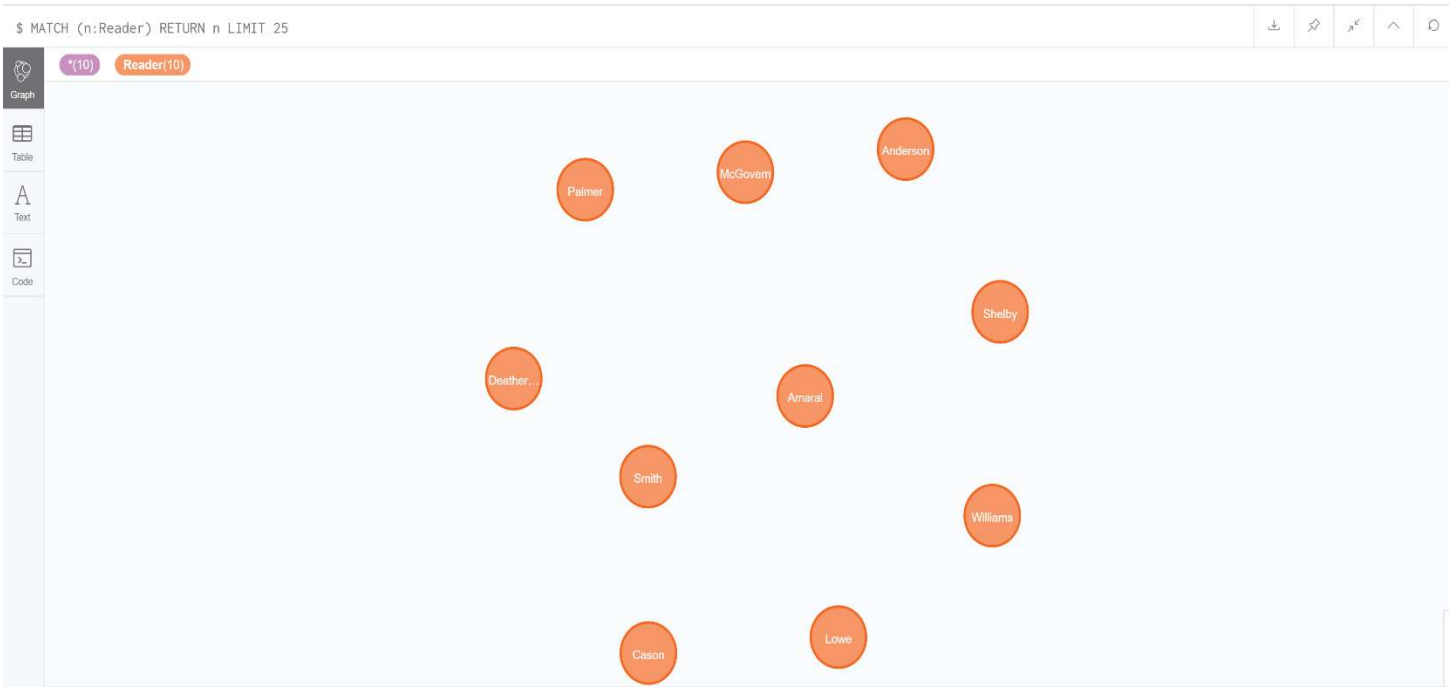

Figure 16 - Readers Nodes

ITEM GRAPH DATABASE

CREATE (book1:Item {itemId:'LB001',type:'Book', title:'The Fault in Our Stars', pages:313, issueNo:'', year:'2012',url:''})
CREATE (book2:Item {itemId:'LB002',type:'Book', title:'Beloved', pages:324 , issueNo:'', year:'1987',url:''})
CREATE (book3:Item {itemId:'LB003',type:'Book', title:'The Great Gatspy', pages:218, issueNo:'', year:'1987',url:''})

```
CREATE (book4:Item {itemId:'LB004',type:'Book', title:'The Testaments', pages:432, issueNo:'',
year:'2019',url:''})
CREATE (book5:Item {itemId:'LB005',type:'Book', title:'Wuthering Heights', pages:400 , issueNo:'',
year:'2005',url:''})
CREATE (book6:Item {itemId:'LB006',type:'Book', title:'Pride and Prejudice', pages:432, issueNo:'',
year:'2004',url:''})
CREATE (book7:Item {itemId:'LB007',type:'Book', title:'Sense and Sensibility', pages:220, issueNo:'',
year:'1995',url:''})
CREATE (book8:Item {itemId:'LB008',type:'Book', title:'Sapiens', pages:443 , issueNo:'', year:'2011',url:''})
CREATE (book9:Item {itemId:'LB009',type:'Book', title:'For Whom the Bells Toll', pages:480, issueNo:'',
year:'1940',url:''})
CREATE (book10:Item {itemId:'LB010',type:'Book', title:'A Room of Ones Own', pages:172, issueNo:'',
year:'1929',url:''})
CREATE (magazine1:Item {itemId:'LB011',type:'Magazine', title:'Times', pages:'', issueNo:'5836596',
year:'',url:''})
CREATE (magazine2:Item {itemId:'LB012',type:'Magazine', title:'US Weekly', pages:'' , issueNo:'451962',
year:'',url:''})
CREATE (magazine3:Item {itemId:'LB013',type:'Magazine', title:'Country Living', pages:'',
issueNo:'a27020796', year:'',url:''})
CREATE (magazine4:Item {itemId:'LB014',type:'Magazine', title:'Vogue', pages:'' , issueNo:'20200501',
year:'',url:''})
CREATE (magazine5:Item {itemId:'LB015',type:'Magazine', title:'Elle!', pages:'', issueNo:'a31005669',
year:'',url:''})
CREATE (magazine6:Item {itemId:'LB016',type:'Magazine', title:'Rolling Stones', pages:'', issueNo:'949908',
year:'',url:''})
CREATE (magazine7:Item {itemId:'LB017',type:'Magazine', title:'The New Yorker', pages:'',
issueNo:'2020518', year:'',url:''})
CREATE (magazine8:Item {itemId:'LB018',type:'Magazine', title:'Vanity Fair', pages:'' , issueNo:'20200401',
year:'',url:''})
CREATE (magazine9:Item {itemId:'LB019',type:'Magazine', title:'People', pages:'', issueNo:'20520353',
year:'',url:''})
CREATE (magazine10:Item {itemId:'LB020',type:'Magazine', title:'Wired', pages:'', issueNo:'4142020',
year:'',url:''})
CREATE (ebook1:Item {itemId:'LB021',type:'Ebook', title:'To Kill a Mockingbird', pages:281, issueNo:'',
year:'1960',
url:'https://docs.google.com/viewer?a=v&pid=sites&srcid=YW5udXJpc2xhbWljc2Nob29sLm9yZ3xzaXN0ZX
Ita2F0ZWx5bnxneDo2NjVmZmE1NzNjNjc4NWM'})
CREATE (ebook2:Item {itemId:'LB022',type:'Ebook', title:'Dancing the Charleston', pages:448, issueNo:'',
year:'2019',url:'https://www.penguin.co.uk/books/111/1112622/dancing-the-charleston/9780440871675.html'})
CREATE (ebook3:Item {itemId:'LB023',type:'Ebook', title:'The Forty Rules of Love', pages:354, issueNo:'',
year:'2009',url:'https://www.penguin.co.uk/books/177/177356/the-forty-rules-of-love/9780241972939.html'})
CREATE (ebook4:Item {itemId:'LB024',type:'Ebook', title:'Killing Commendatore', pages:512 , issueNo:'',
year:'2018',url:'https://www.penguin.co.uk/books/1114146/killing-commendatore/9781787300194.html'})
CREATE (ebook5:Item {itemId:'LB025',type:'Ebook', title:'The Cut Out Girl', pages:288, issueNo:'',
year:'2018',url:'https://www.penguin.co.uk/books/298975/the-cut-out-girl/9780241978719.html'})
CREATE (ebook6:Item {itemId:'LB026',type:'Ebook', title:'The Giver of Stars', pages:448, issueNo:'',
year:'2019',url:'https://www.penguin.co.uk/books/290399/the-giver-of-stars/9780718183202.html'})
```

CREATE (ebook7:Item {itemId:'LB027',type:'Ebook', title:"The Handmaid's Tail", pages:240, issueNo:'', year:'2019',url:'https://www.penguin.co.uk/books/1099258/the-handmaid-s-tale/9780224101936.html'})
CREATE (ebook8:Item {itemId:'LB028',type:'Ebook', title:'Black Sun', pages:336 , issueNo:'', year:'2019',url:'https://www.penguin.co.uk/books/1118026/black-sun/9780552176576.html'})
CREATE (ebook9:Item {itemId:'LB029',type:'Ebook', title:'Airhead', pages:400, issueNo:'', year:'2019',url:'https://www.penguin.co.uk/books/309231/airhead/9781405938358.html'})
CREATE (ebook10:Item {itemId:'LB030',type:'Ebook', title:'Origin', pages:560 , issueNo:'', year:'2018',url:'https://www.penguin.co.uk/books/1113984/origin/9780552174169.html'})

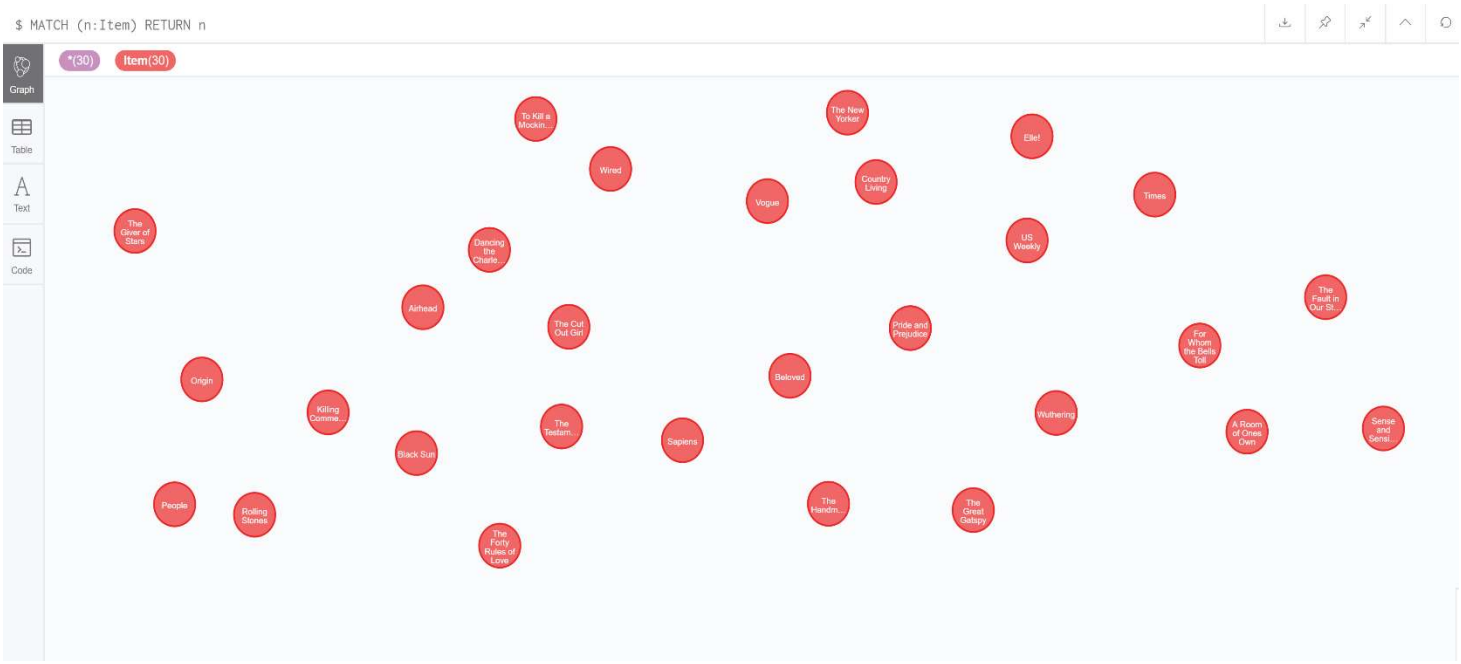

Figure 17 - Item Nodes

PUBLISHER GRAPH DATABASE

CREATE (publisher1:Publisher {pID:'p00001',pName:'Dutton Books', pAddress:'8879 South Fifth Dr Fair Lawn NJ 074101',email:'info@duttonbooks.com', phoneNo:'660-525-6598', country:'USA'})
CREATE (publisher2:Publisher {pID:'p00002',pName:'Alfred A. Knopf', pAddress:'405 6th Ave Howard Beach NY 11414',email:'info@alfredaknopf.com', phoneNo:'646-316-1300', country:'USA'})
CREATE (publisher3:Publisher {pID:'p00003',pName:'Charles Scribners Sons', pAddress:'261 Vernon Street Upper Darby PA 19082',email:'enquiries@simonandschuster.co.uk', phoneNo:'252-331-4418', country:'USA'})
CREATE (publisher4:Publisher {pID:'p00004',pName:'Nan A. Talese / Doubleday', pAddress:'6 Rockledge Rd East Elmhurst NY 11369',email:'ddaypub@randomhouse.com', phoneNo:'815-942-4425', country:'USA'})
CREATE (publisher5:Publisher {pID:'p00005',pName:'Barnes & Noble', pAddress:'79 Sherwood Rd Jamestown NY 14701 ',email:'barnesandnoble@mail.barnesandnoble.com', phoneNo:'719-371-1604', country:'USA'})
CREATE (publisher6:Publisher {pID:'p00006',pName:'Harper', pAddress:'9303 La Sierra Avenue Central Islip NY 11722 ',email:'info@harper.com', phoneNo:'708-849-2918', country:'USA'})
CREATE (publisher7:Publisher {pID:'p00007',pName:'Thomas Egerton', pAddress:'44 Washington Court Bridgeport CT 06606 ',email:'info@thomasegerton.com', phoneNo:'734-922-6235', country:'USA'})

CREATE (publisher8:Publisher {pID:'p00008',pName:'Hogarth Press', pAddress:'21 Green Lake St North Brunswick NJ 08902',email:'info@hogarth.com', phoneNo:'617-357-3788', country:'USA'})
CREATE (publisher9:Publisher {pID:'p00009',pName:'J. B. Lippincott & Co', pAddress:'227 S. 6th Street The J. B. Lippincott Headquarters Building',email:'hello@harpercollins.com', phoneNo:'917-666-7981', country:'USA'})
CREATE (publisher10:Publisher {pID:'p00010',pName:'Penguin Books', pAddress:'20 Vauxhall Bridge Rd Westminster London SW1V 2SA',email:'customersupport@penguinrandomhouse.co.uk', phoneNo:'120-625-6000 ', country:'UK'})
CREATE (publisher11:Publisher {pID:'p00011',pName:'Elif Shafak', pAddress:'',email:'info@elifsafak.com.tr', phoneNo:'216-414-9428', country:'Turkey'})
CREATE (publisher12:Publisher {pID:'p00012',pName:'Jill E. Rancourt', pAddress:'1758 Calvin Street',email:'illERancourt@jourrapide.com', phoneNo:'443-375-1862', country:'USA'})
CREATE (publisher13:Publisher {pID:'p00013',pName:'Marie J. Skipper', pAddress:'Moen 174 3948 PORSGRUNN',email:'MarieJSkipper@jourrapide.com', phoneNo:'998-23-731', country:'Norway'})
CREATE (publisher14:Publisher {pID:'p00014',pName:'Scarlett Bidwill', pAddress:'Kaarrostie 31 01230 VANTAA',email:'ScarlettBidwill@dayrep.com', phoneNo:'044-336-2683', country:'Finland'})
CREATE (publisher15:Publisher {pID:'p00015',pName:'Matilda Weindorfer', pAddress:'4910 rue des Églises Est, Cheneville QC J0V 1E0',email:'MatildaWeindorfer@rhyta.com', phoneNo:'819-428-0909', country:'Canada'})
CREATE (publisher16:Publisher {pID:'p00016',pName:'Blake Ruwolt', pAddress:'49 Neilson Avenue Toronto ON M1M 1V1',email:'BlakeRuwolt@dayrep.com', phoneNo:'416-269-5145', country:'Canada'})
CREATE (publisher17:Publisher {pID:'p00017',pName:'Hudson De Satg', pAddress:'3543 Carson StreetSan Diego CA 92103',email:'HudsonDeSatg@jourrapide.com', phoneNo:'858-947-0401', country:'USA'})
CREATE (publisher18:Publisher {pID:'p00018',pName:'Andrew Carnarvong', pAddress:'3051 Hardman Road Brattleboro VT 05301',email:'AndrewCarnarvon@rhyta.com', phoneNo:'802-812-7335', country:'USA'})
CREATE (publisher19:Publisher {pID:'p00019',pName:'Ellie Akehurst ', pAddress:'32 Tadcaster Rd, PITCAPLE AB51 1QF',email:'EllieAkehurst@rhyta.com', phoneNo:'077-764-4730', country:'UK'})
CREATE (publisher20:Publisher {pID:'p00020',pName:'Henry Tully', pAddress:'Idvägen 90 360 73 LENHOVDAF',email:'HenryTully@dayrep.com', phoneNo:'0474-764-9764', country:'Sweden'})
CREATE (publisher21:Publisher {pID:'p00021',pName:'Dean Fereday', pAddress:'4393 Coleman Avenue, Vista, CA 92083',email:'DeanFereday@dayrep.com', phoneNo:'760-732-7972', country:'USA'})

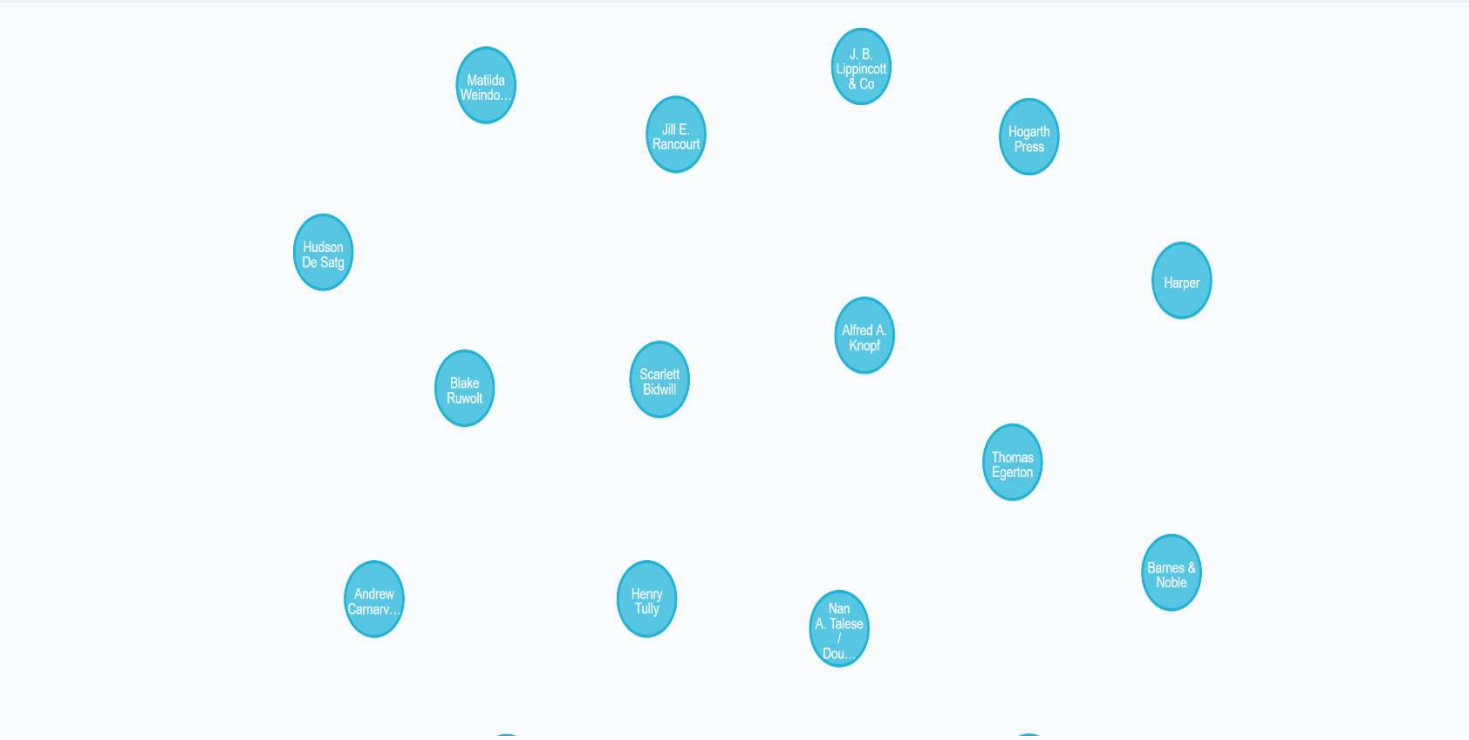

Figure 18 - Publisher Nodes

COPY DATABASE

```
CREATE (copy1:Copy {copyId:'C0001',name:'Copy 1'})
CREATE (copy2:Copy {copyId:'C0002',name:'Copy 2'})
CREATE (copy3:Copy {copyId:'C0003',name:'Copy 3'})
CREATE (copy4:Copy {copyId:'C0004',name:'Copy 1'})
CREATE (copy5:Copy {copyId:'C0005',name:'Copy 2'})
CREATE (copy6:Copy {copyId:'C0006',name:'Copy 1'})
CREATE (copy7:Copy {copyId:'C0007',name:'Copy 2'})
CREATE (copy8:Copy {copyId:'C0008',name:'Copy 3'})
CREATE (copy9:Copy {copyId:'C0009',name:'Copy 4'})
CREATE (copy10:Copy {copyId:'C0010',name:'Copy 1'})
CREATE (copy11:Copy {copyId:'C0011',name:'Copy 2'})
CREATE (copy12:Copy {copyId:'C0012',name:'Copy 3'})
CREATE (copy13:Copy {copyId:'C0013',name:'Copy 4'})
CREATE (copy14:Copy {copyId:'C0014',name:'Copy 5'})
CREATE (copy15:Copy {copyId:'C0015',name:'Copy 1'})
CREATE (copy16:Copy {copyId:'C0016',name:'Copy 2'})
CREATE (copy17:Copy {copyId:'C0017',name:'Copy 1'})
CREATE (copy18:Copy {copyId:'C0018',name:'Copy 2'})
CREATE (copy19:Copy {copyId:'C0019',name:'Copy 3'})
CREATE (copy20:Copy {copyId:'C0020',name:'Copy 1'})
CREATE (copy21:Copy {copyId:'C0021',name:'Copy 2'})
CREATE (copy22:Copy {copyId:'C0022',name:'Copy 1'})
CREATE (copy23:Copy {copyId:'C0023',name:'Copy 2'})
CREATE (copy24:Copy {copyId:'C0024',name:'Copy 3'})
```

CREATE (copy25:Copy {copyId:'C0025',name:'Copy 1'})
CREATE (copy26:Copy {copyId:'C0026',name:'Copy 2'})
CREATE (copy27:Copy {copyId:'C0027',name:'Copy 1'})
CREATE (copy28:Copy {copyId:'C0028',name:'Copy 2'})
CREATE (copy29:Copy {copyId:'C0029',name:'Copy 1'})
CREATE (copy30:Copy {copyId:'C0030',name:'Copy 2'})
CREATE (copy31:Copy {copyId:'C0031',name:'Copy 1'})
CREATE (copy32:Copy {copyId:'C0032',name:'Copy 2'})
CREATE (copy33:Copy {copyId:'C0033',name:'Copy 1'})
CREATE (copy34:Copy {copyId:'C0034',name:'Copy 2'})
CREATE (copy35:Copy {copyId:'C0035',name:'Copy 1'})
CREATE (copy36:Copy {copyId:'C0036',name:'Copy 2'})
CREATE (copy37:Copy {copyId:'C0037',name:'Copy 1'})
CREATE (copy38:Copy {copyId:'C0038',name:'Copy 2'})
CREATE (copy39:Copy {copyId:'C0039',name:'Copy 1'})
CREATE (copy40:Copy {copyId:'C0040',name:'Copy 2'})
CREATE (copy41:Copy {copyId:'C0041',name:'Copy 1'})
CREATE (copy42:Copy {copyId:'C0042',name:'Copy 2'})
CREATE (copy43:Copy {copyId:'C0043',name:'Copy 1'})
CREATE (copy44:Copy {copyId:'C0044',name:'Copy 2'})
CREATE (copy45:Copy {copyId:'C0045',name:'Copy 1'})
CREATE (copy46:Copy {copyId:'C0046',name:'Copy 2'})
CREATE (copy47:Copy {copyId:'C0047',name:'Copy 1'})
CREATE (copy48:Copy {copyId:'C0048',name:'Copy 2'})

CREATING EDGES

:BELONGSTO
CREATE (copy1) –[:BELONGSTO] –> (book1)
CREATE (copy2) –[:BELONGSTO] –> (book1)
CREATE (copy3) –[:BELONGSTO] –> (book1)
CREATE (copy4) –[:BELONGSTO] –> (book2)
CREATE (copy5) –[:BELONGSTO] –> (book2)
CREATE (copy6) –[:BELONGSTO] –> (book3)
CREATE (copy7) –[:BELONGSTO] –> (book3)
CREATE (copy8) –[:BELONGSTO] –> (book3)
CREATE (copy9) –[:BELONGSTO] –> (book3)
CREATE (copy10) –[:BELONGSTO] –> (book4)
CREATE (copy11) –[:BELONGSTO] –> (book4)
CREATE (copy12) –[:BELONGSTO] –> (book4)
CREATE (copy13) –[:BELONGSTO] –> (book4)
CREATE (copy14) –[:BELONGSTO] –> (book4)
CREATE (copy15) –[:BELONGSTO] –> (book5)
CREATE (copy16) –[:BELONGSTO] –> (book5)
CREATE (copy17) –[:BELONGSTO] –> (book6)
CREATE (copy18) –[:BELONGSTO] –> (book6)
CREATE (copy19) –[:BELONGSTO] –> (book6)

CREATE (copy20) –[:BELONGSTO] –> (book7)
CREATE (copy21) –[:BELONGSTO] –> (book7)
CREATE (copy22) –[:BELONGSTO] –> (book8)
CREATE (copy23) –[:BELONGSTO] –> (book8)
CREATE (copy24) –[:BELONGSTO] –> (book8)
CREATE (copy25) –[:BELONGSTO] –> (book9)
CREATE (copy26) –[:BELONGSTO] –> (book9)
CREATE (copy27) –[:BELONGSTO] –> (book10)
CREATE (copy28) –[:BELONGSTO] –> (book10)
CREATE (copy29) –[:BELONGSTO] –> (magazine1)
CREATE (copy30) –[:BELONGSTO] –> (magazine1)
CREATE (copy31) –[:BELONGSTO] –> (magazine2)
CREATE (copy32) –[:BELONGSTO] –> (magazine2)
CREATE (copy33) –[:BELONGSTO] –> (magazine3)
CREATE (copy34) –[:BELONGSTO] –> (magazine3)
CREATE (copy35) –[:BELONGSTO] –> (magazine4)
CREATE (copy36) –[:BELONGSTO] –> (magazine4)
CREATE (copy37) –[:BELONGSTO] –> (magazine5)
CREATE (copy38) –[:BELONGSTO] –> (magazine5)
CREATE (copy39) –[:BELONGSTO] –> (magazine6)
CREATE (copy40) –[:BELONGSTO] –> (magazine6)
CREATE (copy41) –[:BELONGSTO] –> (magazine7)
CREATE (copy42) –[:BELONGSTO] –> (magazine7)
CREATE (copy43) –[:BELONGSTO] –> (magazine8)
CREATE (copy44) –[:BELONGSTO] –> (magazine8)
CREATE (copy45) –[:BELONGSTO] –> (magazine9)
CREATE (copy46) –[:BELONGSTO] –> (magazine9)
CREATE (copy47) –[:BELONGSTO] –> (magazine10)
CREATE (copy48) –[:BELONGSTO] –> (magazine10)

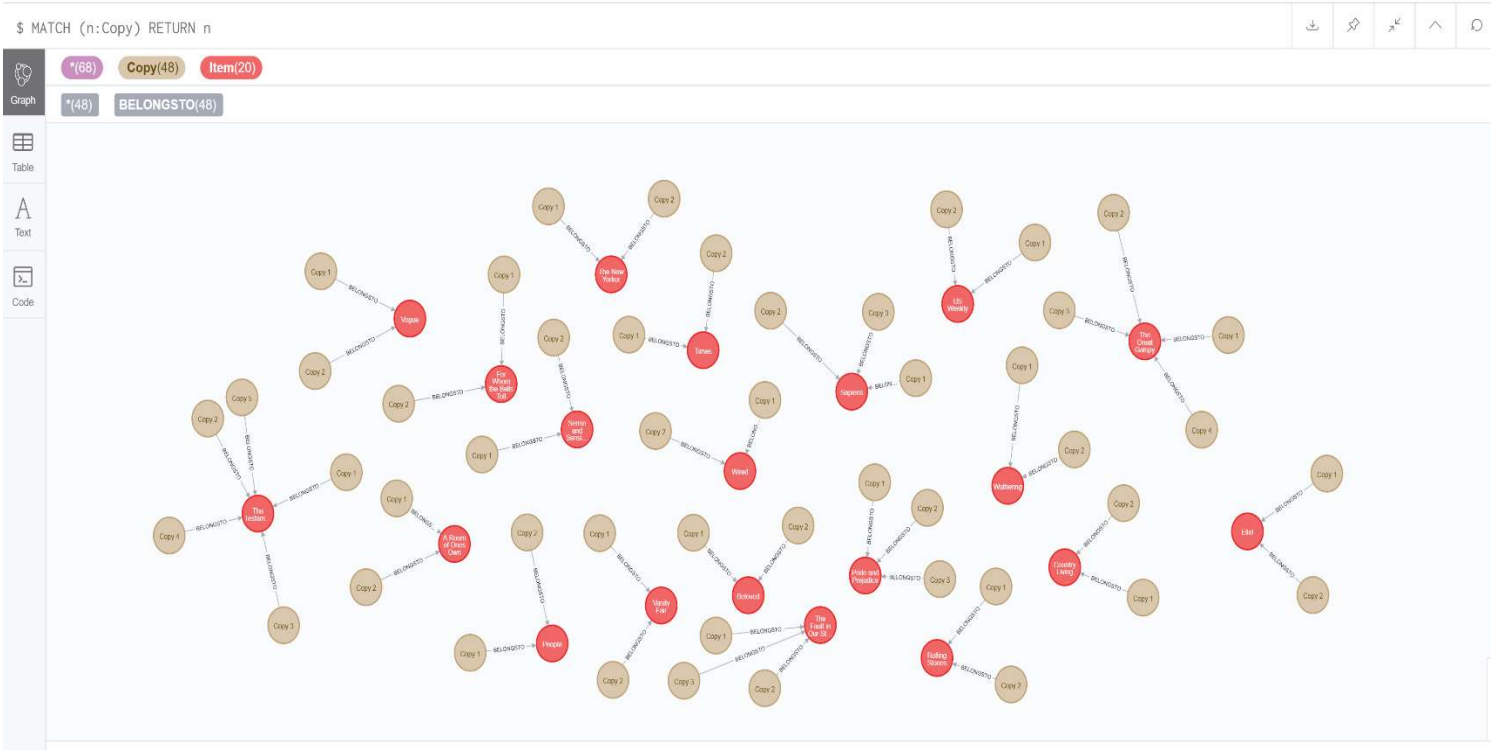

Figure 19 - Each Items and their copies

:BORROWED

CREATE (reader5)–[:BORROWED{borrowDate:date('2020-05-10'), returnDate:date('2020-05-24')}]–>(copy2)

CREATE (reader5)–[:BORROWED{borrowDate:date('2020-05-12'), returnDate:date('2020-05-25')}]–>(copy9)

CREATE (reader9)–[:BORROWED {borrowDate:date('2020-05-15'), returnDate:date('2020-05-29')}]–>(copy15)

CREATE (reader10)–[:BORROWED {borrowDate:date('2020-05-10'), returnDate:date('2020-05-24')}]–>(copy16)

CREATE (reader1)–[:BORROWED {borrowDate:date('2020-05-13'), returnDate:date('2020-05-27')}]–>(copy21)

CREATE (reader1)–[:BORROWED {borrowDate:date('2020-05-13'),returnDate:date('2020-05-27')}]–>(copy26)

CREATE (reader1)–[:BORROWED {borrowDate:date('2020-05-18') ,returnDate:date('2020-06-01')}]–>(copy33)

CREATE (reader3)–[:BORROWED {borrowDate:date('2020-05-24') ,returnDate:date('2020-06-07')}]–>(copy35)

CREATE (reader7)–[:BORROWED {borrowDate:date('2020-05-20') ,returnDate:date('2020-06-03')}]–>(copy36)

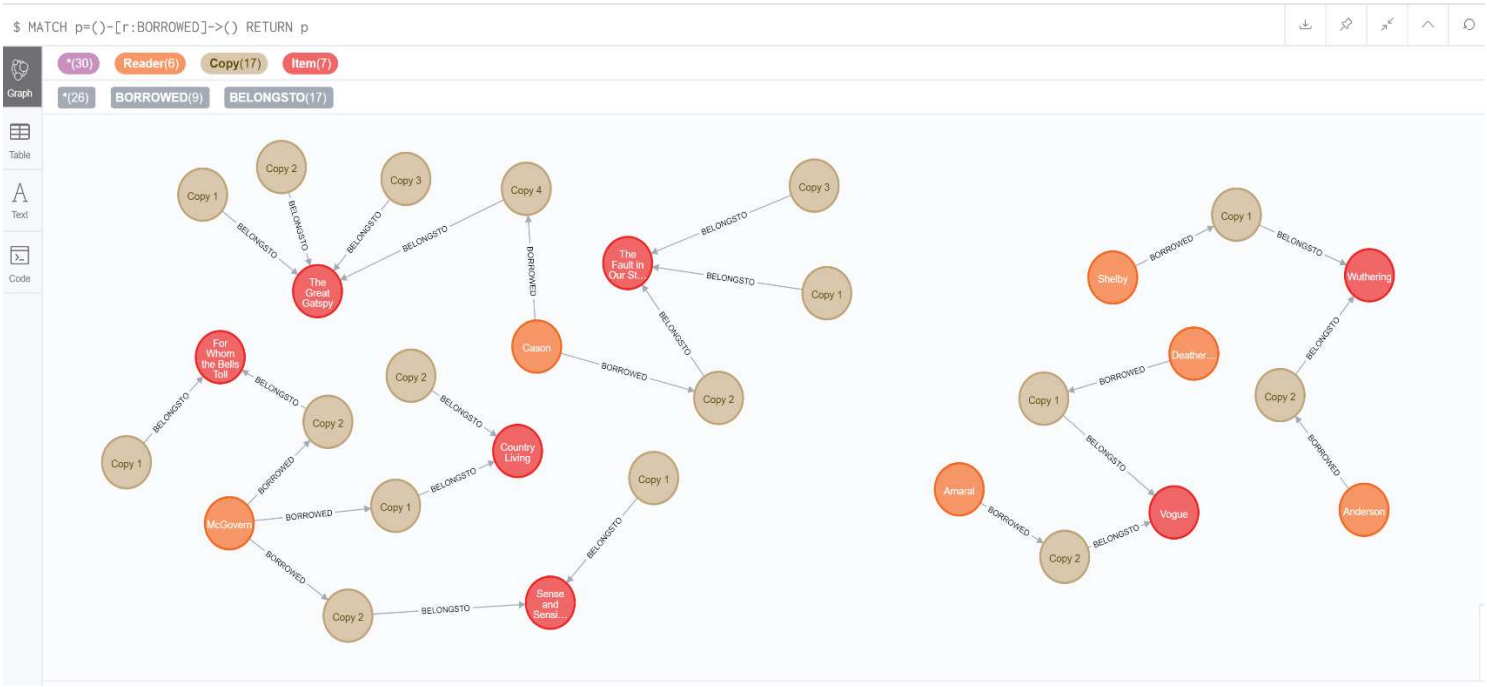

Figure 20 - Borrowed Items

:PUBLISHED

CREATE (publisher1) –[:PUBLISHED] –> (book1)
CREATE (publisher2) –[:PUBLISHED] –> (book2)
CREATE (publisher3) –[:PUBLISHED] –> (book3)
CREATE (publisher4) –[:PUBLISHED] –> (book4)
CREATE (publisher5) –[:PUBLISHED] –> (book5)
CREATE (publisher5) –[:PUBLISHED] –> (book6)
CREATE (publisher7) –[:PUBLISHED] –> (book7)
CREATE (publisher6) –[:PUBLISHED] –> (book8)
CREATE (publisher3) –[:PUBLISHED] –> (book9)
CREATE (publisher8) –[:PUBLISHED] –> (book10)
CREATE (publisher12) –[:PUBLISHED] –> (magazine1)
CREATE (publisher13) –[:PUBLISHED] –> (magazine2)
CREATE (publisher14) –[:PUBLISHED] –> (magazine3)
CREATE (publisher15) –[:PUBLISHED] –> (magazine4)
CREATE (publisher16) –[:PUBLISHED] –> (magazine5)
CREATE (publisher17) –[:PUBLISHED] –> (magazine6)
CREATE (publisher18) –[:PUBLISHED] –> (magazine7)
CREATE (publisher19) –[:PUBLISHED] –> (magazine8)
CREATE (publisher20) –[:PUBLISHED] –> (magazine9)
CREATE (publisher21) –[:PUBLISHED] –> (magazine10)
CREATE (publisher9) –[:PUBLISHED] –> (ebook1)
CREATE (publisher10) –[:PUBLISHED] –> (ebook2)
CREATE (publisher11) –[:PUBLISHED] –> (ebook3)
CREATE (publisher10) –[:PUBLISHED] –> (ebook4)
CREATE (publisher10) –[:PUBLISHED] –> (ebook5)
CREATE (publisher10) –[:PUBLISHED] –> (ebook6)
CREATE (publisher10) –[:PUBLISHED] –> (ebook7)

CREATE (publisher10) –[:PUBLISHED] –> (ebook8)
CREATE (publisher10) –[:PUBLISHED] –> (ebook9)
CREATE (publisher10) –[:PUBLISHED] –> (ebook10)

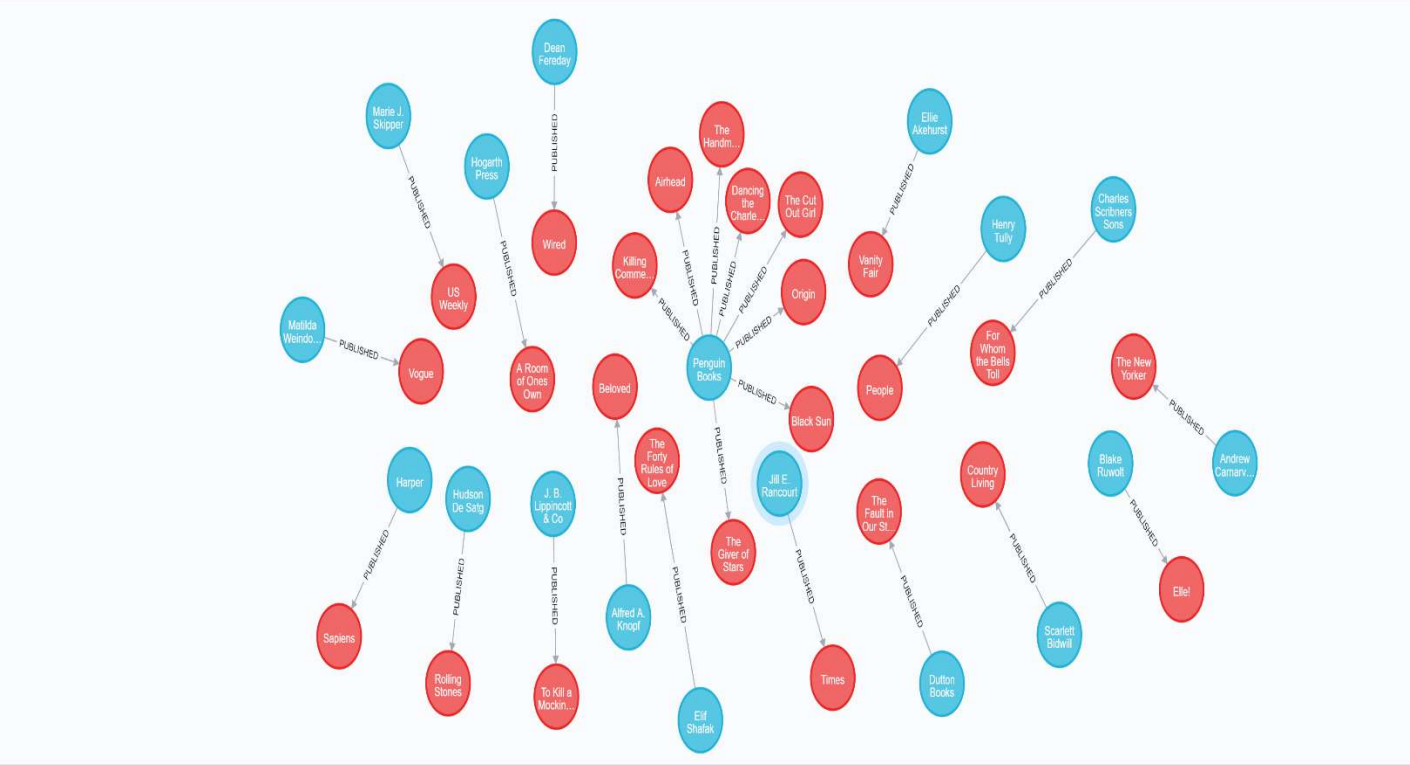

Figure 21 - Publisher and their published items

FIVE USE CASES FOR NEO4J

1. Readers who have never borrowed an Item

```
MATCH path=(r:Reader)-[b]-(c:Copy)
WHERE (r)-[:BORROWED]-(c)
WITH collect(r) as rc
MATCH (rn:Reader)
WHERE NOT rn IN rc
RETURN rn
```

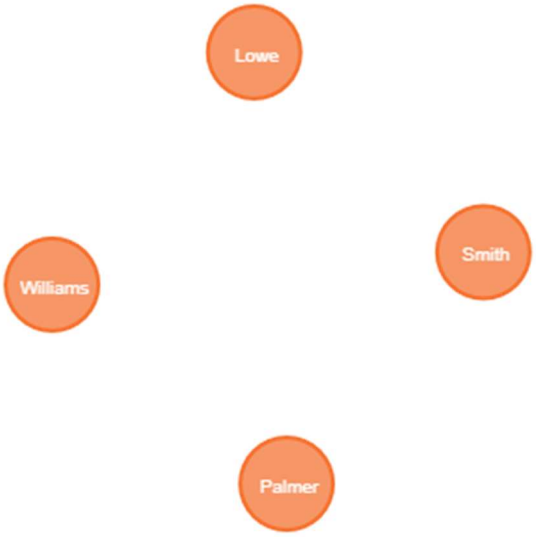

Figure 22 - 1. Readers who have never borrowed an Item

2. Reader who borrowed Items, and return date is overdue have gone pass the return date.

```
MATCH (r:Reader)-[b:BORROWED]->(c:Copy)
MATCH(c)-[:BELONGSTO]->(i:Item)
WHERE date() > b.returnDate
RETURN r.fName,r.lName,i.title,i.type,c.copyId,b.borrowDate,b.returnDate,date() AS currentDate
```

r.fName	r.lName	i.title	i.type	c.copyId	b.borrowDate	b.returnDate	currentDate
"Sabra"	"Cason"	"The Fault in Our Stars"	"Book"	"C0002"	"2020-05-10"	"2020-05-24"	"2020-05-25"
"Christopher"	"Anderson"	"Wuthering Heights"	"Book"	"C0016"	"2020-05-10"	"2020-05-24"	"2020-05-25"

3. Readers who borrowed more than 1 books

```
MATCH (r:Reader)-[b:BORROWED]->(c:Copy)
MATCH(c)-[:BELONGSTO]->(i:Item)
WHERE r.noBorrowedItems > 1
RETURN r.readerId,r.fName,r.lName,i.title,i.type,c.copyId,b.borrowDate,b.returnDate
```

```
$ MATCH (r:Reader)-[b:BORROWED]->(c:Copy) MATCH(c)-[:BELONGSTO]->(i:Item) WHERE r.noBorrowedItems > 1 RETURN r.readerId,r.fName,r.lName,i.title,i.type,c.copyId,b…
```

Table | Text | Code

r.readerId	r.fName	r.lName	i.title	i.type	c.copyId	b.borrowDate	b.returnDate
"r0001"	"Sharon"	"McGovern"	"Country Living"	"Magazine"	"C0033"	"2020-05-18"	"2020-06-01"
"r0001"	"Sharon"	"McGovern"	"For Whom the Bells Toll"	"Book"	"C0026"	"2020-05-13"	"2020-05-27"
"r0001"	"Sharon"	"McGovern"	"Sense and Sensibility"	"Book"	"C0021"	"2020-05-13"	"2020-05-27"
"r0005"	"Sabra"	"Cason"	"The Great Gatspy"	"Book"	"C0009"	"2020-05-12"	"2020-05-25"
"r0005"	"Sabra"	"Cason"	"The Fault in Our Stars"	"Book"	"C0002"	"2020-05-10"	"2020-05-24"

4. Publishers with more than one book in our database

```
MATCH (publisher:Publisher)-[:PUBLISHED]->(:Item)
WITH publisher, count(*) AS count_publisher
WHERE count_publisher > = 2
RETURN *
```

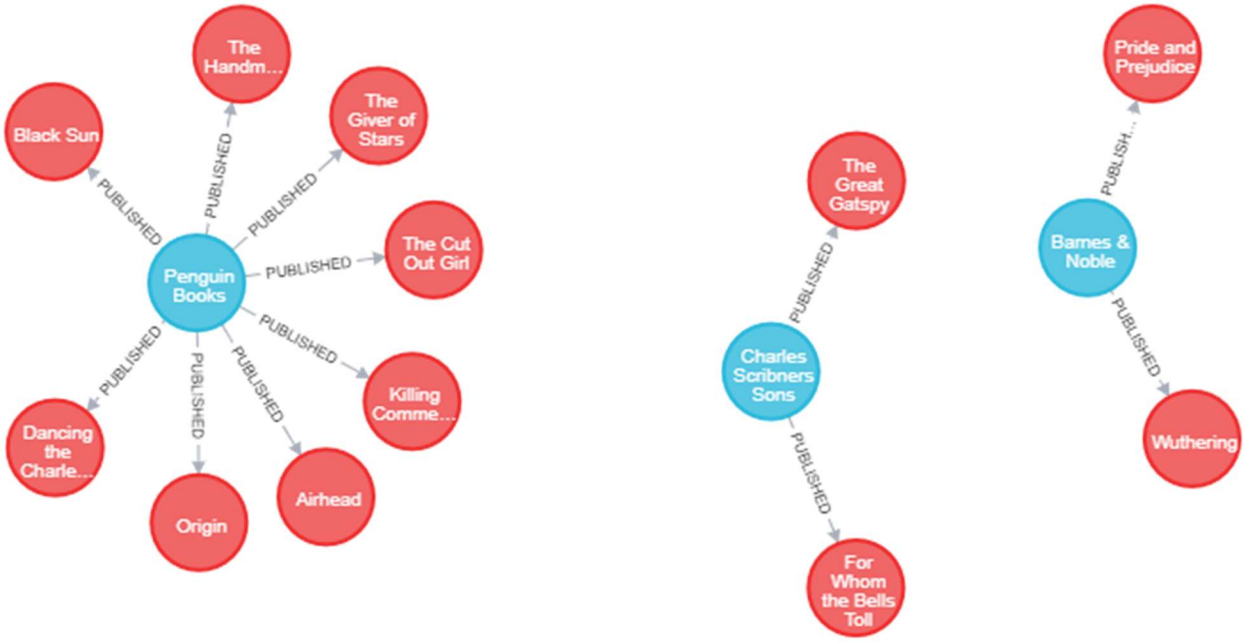

Figure 23 - Publishers with more than a book in database

5. Readers that borrowed books that were published in the USA

```
MATCH (r:Reader)-[:BORROWED]->(c:Copy)
MATCH (c)-[:BELONGSTO]->(i:Item)
MATCH (p:Publisher)-[:PUBLISHED]->(i)
WHERE p.country = 'USA'
RETURN p,i,c,r
```

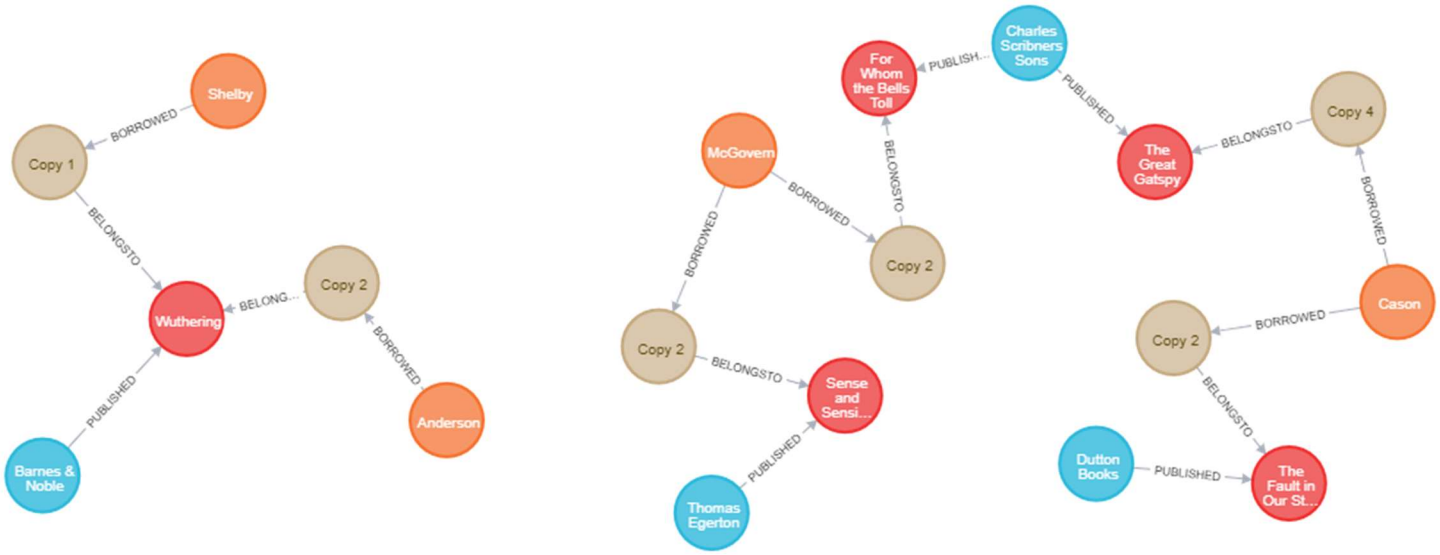

Figure 24 - 5. Readers that borrowed books that were published in the USA

MONGODB

The documents Items, Publishers, Members and Borrow were created in MongoDB, and sample data were inserted.

The **Items** document stores all information related to books, ebooks, and magazines. Storing all data in a single document makes it easy to perform searches and updates. At the cost of having one data-rich document.

Publisher document stores in-depth details about publishers of assets. The reason for the need for a separate document instead of embedding data into **items** document is to reduce the storage of duplicates as a single publisher can publish different items.

The **Members** document contains personal information and identifier of each reader of the library.

The Borrow document stores members with items they borrowed and also stores borrow and return dates. Having a dedicated document makes it easier for inserting and removing data as users borrow and return items.

Items

Document structure : for ebooks

```
{
        'item_id':"
        'type':'ebooks',
        'title':",
        'author':[","],
        pages:,
        year:,
        'genre':[","],
        avunits:,
        'publisher':",
        'url':",
        downloads:0
}
```

Document structure : for books

```
{
        'item_id':"
        'type':'books',
        'title':",
        'author':[","],
        pages:,
        year:,
        'genre':[","],
        avunits:,
        'publisher':"
}
```

Document structure : for magazine

```
{
        'item_id':"
        'type':'magazine',
        'title':",
        'issue':",
        'genre':[","],
        avunits:,
        'publisher':",
}
```

Publisher

Document structure: for publisher

```
{
        'name':",
        'country':",
        'address':[     {'street':"},
                        {'city':"},
                        {'pin':"}   ],
        'phone':[","],
        'email':[","]
}
```

Members

Document structure: for Members

```
{
        'mID':",
        "regdate" :new Date('YYYY-MM-
DD'),
'name':[ {'fname':"},
                {'lname':"}
        ],
'address':[     {'street':"},
                {'city':"},
                {'pin':"}
        ]
}
```

Borrow

Document structure: for Borrow

```
{
        'mID':",
        item_id:",
        "withdrawdate" :new Date('YYYY-MM-DD'),
        "returndate" :new Date('YYYY-MM-DD')
}
```

MEMBERS DOCUMENT

use library

db.createCollection("members")

db.members.createIndex({"mID": 1 }, { unique: true })

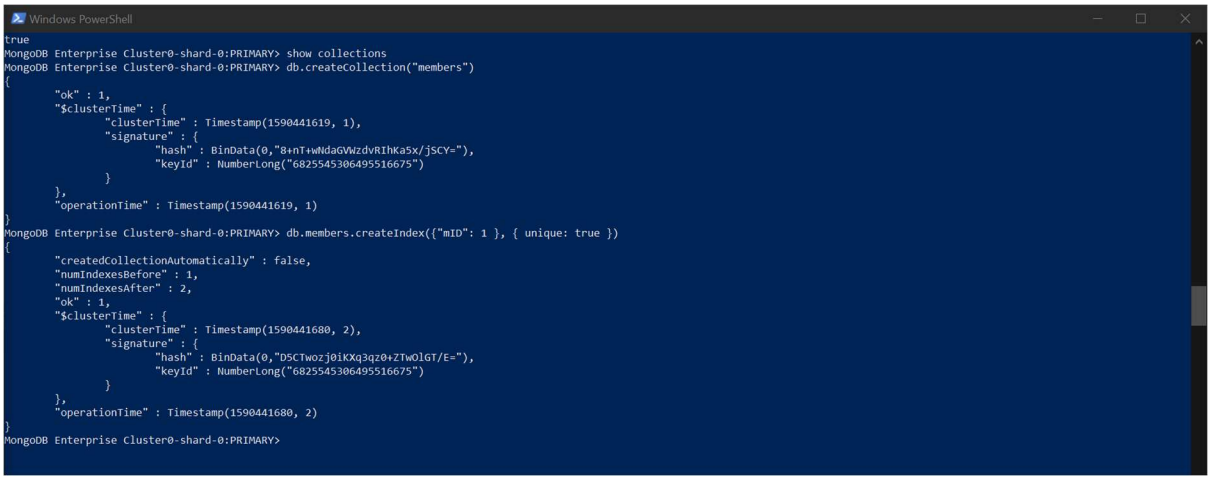

Figure 25 - Creating Member Document

db.members.insert({'mID':'R0001', "regdate" :new Date('2020-01-20'), 'name':[{'fname':'Sharon'}, {'lname':'McGovern'}], 'address':[{'street':'280 Lyndon Street'}, {'city':'Philadelphia'}, {'pin':'PA 19103'}]})

db.members.insert({'mID':'R0002', "regdate" :new Date('2020-01-02'), 'name':[{'fname':'Richard'}, {'lname':'Palmer'}], 'address':[{'street':'636 Yorkshire Circle'}, {'city':'Greenville'}, {'pin':'NC 27834'}]})

db.members.insert({'mID':'R0003', "regdate" :new Date('2020-01-11'), 'name':[{'fname':'Lewis'}, {'lname':'Deatherage'}], 'address':[{'street':'4742 Mulberry Avenue'}, {'city':'North Little Rock'}, {'pin':'AR 72114'}]})

db.members.insert({'mID':'R0004', "regdate" :new Date('2020-01-20'), 'name':[{'fname':'Charles'}, {'lname':'Smith'}], 'address':[{'street':'1944 Cunningham Court'}, {'city':'Troy'}, {'pin':'MI 48084'}]})

db.members.insert({'mID':'R0005', "regdate" :new Date('2020-02-26'), 'name':[{'fname':'Sabra'}, {'lname':'Cason'}], 'address':[{'street':'407 Franklin Avenue'}, {'city':'Orlando'}, {'pin':'FL 32801'}]})

db.members.insert({'mID':'R0006', "regdate" :new Date('2020-02-26'), 'name':[{'fname':'Carolyn'}, {'lname':'Lowe'}], 'address':[{'street':'3480 Sharon Lane'}, {'city':'Mount Sterling'}, {'pin':'MO 65062'}]})

db.members.insert({'mID':'R0007', "regdate" :new Date('2020-02-03'), 'name':[{'fname':'Marlena'}, {'lname':'Amaral'}], 'address':[{'street':'47 Pearl Street'}, {'city':'Sacramento'}, {'pin':'CA 95814'}]})

db.members.insert({'mID':'R0008', "regdate" :new Date('2020-02-11'), 'name':[{'fname':'Betty'}, {'lname':'Williams'}], 'address':[{'street':'2346 Laurel Lee'}, {'city':'Maplewood'}, {'pin':'MN 55119'}]})

db.members.insert({'mID':'R0009', "regdate" :new Date('2020-02-11'), 'name':[{'fname':'Sharon'}, {'lname':'Selby'}], 'address':[{'street':'1583 Tetrick Road'}, {'city':'Winter Haven'}, {'pin':'FL 33881'}]})

db.members.insert({'mID':'R0010', "regdate" :new Date('2020-02-21'), 'name':[{'fname':'Christopher'}, {'lname':'Anderson'}], 'address':[{'street':'1303 North Avenue'}, {'city':'Omaha'}, {'pin':'NE 68144'}]})

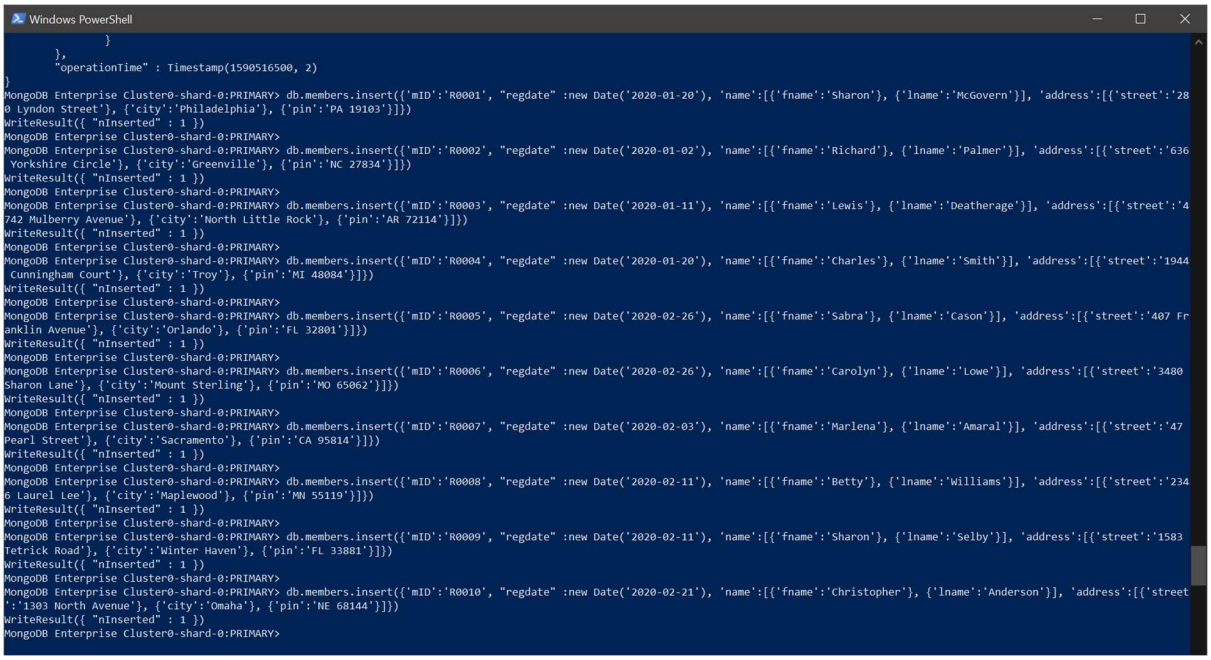

Figure 26 - Inserting Member's data

BORROW DOCUMENT

db.createCollection("borrow")

db.borrow.createIndex({"mID": 1, item_id: 1 }, { unique: true })

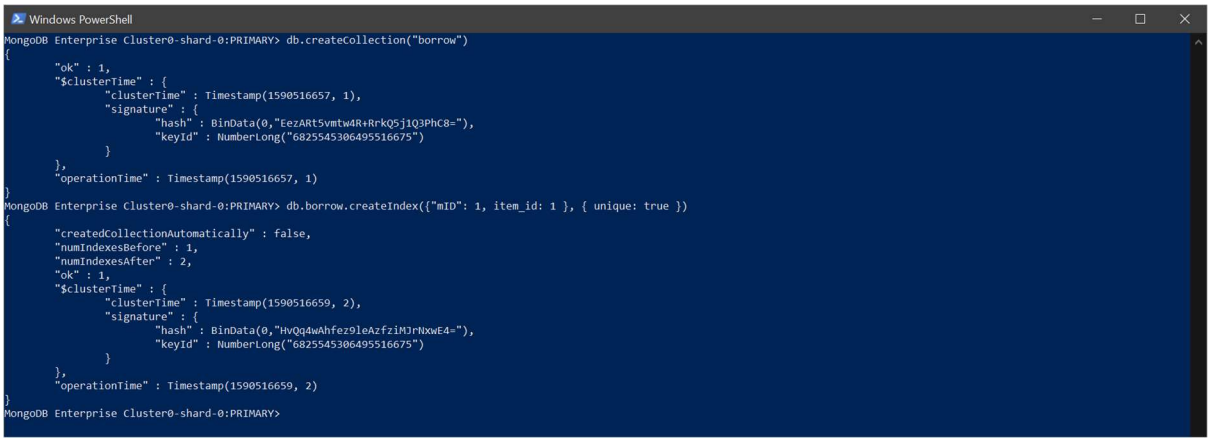

Figure 27 - Creating Borrow Document

db.borrow.insert({'mID':'R0001', item_id:'LB0007', "withdrawdate" :new Date('2020-05-13'), "returndate" :new Date('2020-05-26')})

db.borrow.insert({'mID':'R0001', item_id:'LB0009', "withdrawdate" :new Date('2020-05-13'), "returndate" :new Date('2020-05-26')})

db.borrow.insert({'mID':'R0001', item_id:'LB0023', "withdrawdate" :new Date('2020-05-18'), "returndate" :new Date('2020-05-31')})

db.borrow.insert({'mID':'R0003', item_id:'LB0024', "withdrawdate" :new Date('2020-05-24'), "returndate" :new Date('2020-06-06')})

db.borrow.insert({'mID':'R0005', item_id:'LB0001', "withdrawdate" :new Date('2020-05-10'), "returndate" :new Date('2020-05-23')})

db.borrow.insert({'mID':'R0005', item_id:'LB0003', "withdrawdate" :new Date('2020-05-12'), "returndate" :new Date('2020-05-25')})

db.borrow.insert({'mID':'R0007', item_id:'LB0024', "withdrawdate" :new Date('2020-05-20'), "returndate" :new Date('2020-06-02')})

db.borrow.insert({'mID':'R0009', item_id:'LB0005', "withdrawdate" :new Date('2020-05-15'), "returndate" :new Date('2020-05-28')})

db.borrow.insert({'mID':'R0010', item_id:'LB0005', "withdrawdate" :new Date('2020-05-10'), "returndate" :new Date('2020-05-23')})

Figure 28 - Inserting Borrow information

ITEMS DOCUMENT

db.createCollection("items")

db.items.createIndex({"item_id": 1 }, { unique: true })

db.items.insert({'item_id':'LB0001', 'type':'book','title':'Tiny Pretty Things','author':['Dhonielle Clayton','Sona Charaipotra'], pages:448, year:2015, 'genre':['Fiction'], avunits:3, 'publisher':'HarperCollins Publishers'})

db.items.insert({'item_id':'LB0002', 'type':'book','title':'The Fault In Our Stars','author':['John Green'], pages: 313, year:2012, 'genre':['Young Adult','Fiction'], avunits:3, 'publisher':'Dutton Books'})

db.items.insert({'item_id':'LB0003', 'type':'book','title':'Beloved','author':['Toni Morrison'], pages:324, year:1987, 'genre':['American Literature'], avunits:2, 'publisher':'Alfred A. Knopf'})

db.items.insert({'item_id':'LB0004', 'type':'book','title':'The Great Gatsby','author':['F. Scott Fitzgerald'], pages:218, year:1925, 'genre':['a','b'], avunits:4, 'publisher':'Charles Scribner's Sons'})

db.items.insert({'item_id':'LB0005', 'type':'book','title':'The Testaments','author':['Margaret Atwood'], pages:432, year:2019, 'genre':['Tragedy','Fiction'], avunits:5, 'publisher':'Doubleday'})

db.items.insert({'item_id':'LB0006', 'type':'book','title':'Wuthering Heights','author':['Emily Bronte'], pages:400, year:2005, 'genre':['Tragedy','Gothic'], avunits:2, 'publisher':'Barnes & Noble'})

db.items.insert({'item_id':'LB0007', 'type':'book','title':'Pride And Prejudice','author':['Jane Austen'], pages:432, year:2004, 'genre':['Satire','Drama'], avunits:3, 'publisher':'Barnes & Noble'})

db.items.insert({'item_id':'LB0008', 'type':'book','title':'Sense And Sensibility','author':['jane Austen'], pages:220, year:1995, 'genre':['Romance'], avunits:2, 'publisher':'Thomas Egerton'})

db.items.insert({'item_id':'LB0009', 'type':'book','title':'Sapiens','author':['yuan Noah Harari'], pages:443, year:2011, 'genre':['Non Fiction'], avunits:3, 'publisher':'Harper'})

db.items.insert({'item_id':'LB0010', 'type':'book','title':'For Whom The Bells Toll','author':['Ernest Hemingway'], pages:480, year:1940, 'genre':['War Story'], avunits:2, 'publisher':"Charles Scribner's Sons"})

db.items.insert({'item_id':'LB0011', 'type':'book','title':"A Room Of One's Own",'author':['Virginia Woolf'], pages:172, year:1929, 'genre':['Essay','Fiction'], avunits:2, 'publisher':'Hogarth Press'})

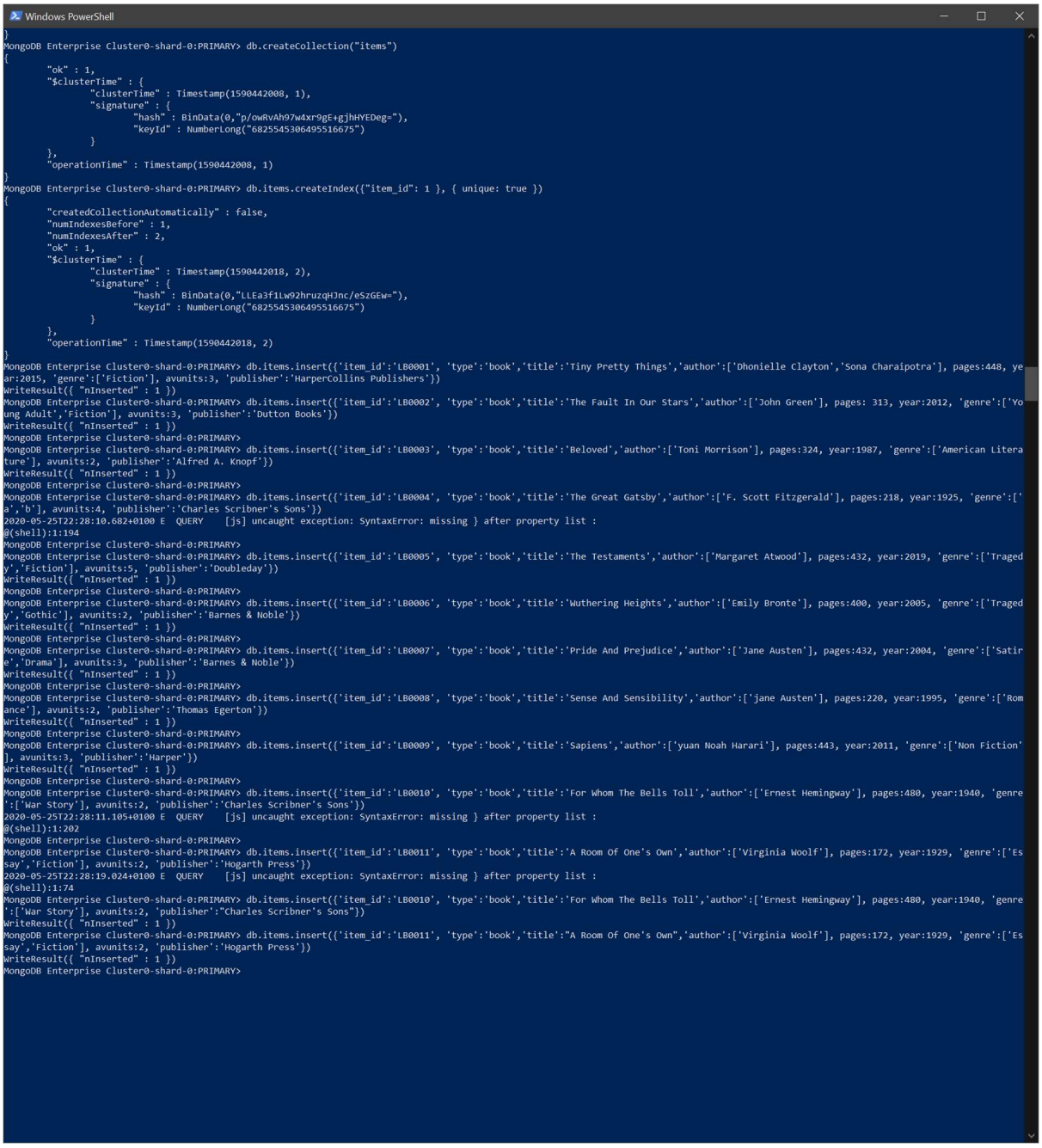

Figure 29 -Creating Item Document and Inserting item information 1

db.items.insert({'item_id':'LB0012', 'type':'ebook','title':'Machine Learning For Dummies','author':['John Paul Mueller','Luca Massaron'], 'pages':'432', 'year':'2016', 'genre':['computer','internet','database'], 'publisher':'John

Wiley and Sons', 'url':'https://www.dummies.com/store/product/Machine-Learning-For-Dummies.productCd-1119245516,navId-322451,descCd-description.html', 'downloads':0})

db.items.insert({'item_id':'LB0013', 'type':'ebook', 'title':'To Kill a Mockingbird', 'author':['Harper Lee'], pages:281, year:1960, 'genre':['Southern','Gothic'], 'publisher':'J. B. Lippincott & Co.', 'url':'https://docs.google.com/viewer?a=v&pid=sites&srcid=YW5udXJpc2xhbWljc2Nob29sLm9yZ3xzaXN0ZXIta2F0ZWx5bnxneDo2NjVmZmE1NzNjNjc4NWM', downloads:0})

db.items.insert({'item_id':'LB0014', 'type':'ebook', 'title':'Dancing the Charleston', 'author':['Jacqueline Wilson'], pages:448, year:2019, 'genre':['Historical','Fiction'], 'publisher':'Penguin Books', 'url':'https://www.penguin.co.uk/books/111/1112622/dancing-the-charleston/9780440871675.html', downloads:0})

db.items.insert({'item_id':'LB0015', 'type':'ebook', 'title':'The Forty Rules of Love', 'author':['Elif Shafak'], pages:354, year:2009, 'genre':['Fiction'], 'publisher':'Elif Shafak', 'url':'https://www.penguin.co.uk/books/177/177356/the-forty-rules-of-love/9780241972939.html', downloads:0})

db.items.insert({'item_id':'LB0016', 'type':'ebook', 'title':'Killing Commendatore', 'author':['Haruki Murakami'], pages:512, year:2018, 'genre':['Fiction','Magical Realism'], 'publisher':'Penguin Books', 'url':'https://www.penguin.co.uk/books/1114146/killing-commendatore/9781787300194.html', downloads:0})

db.items.insert({'item_id':'LB0017', 'type':'ebook', 'title':'The Cut Out Girl', 'author':['Bart van es'], pages:288, year:2018, 'genre':['Biography'], 'publisher':'Penguin Books', 'url':'https://www.penguin.co.uk/books/298975/the-cut-out-girl/9780241978719.html', downloads:0})

db.items.insert({'item_id':'LB0018', 'type':'ebook', 'title':'The giver of stars', 'author':['JoJo Moyes'], pages:448, year:2019, 'genre':['Fiction','Romance'], 'publisher':'Penguin Books', 'url':'https://www.penguin.co.uk/books/290399/the-giver-of-stars/9780718183202.html', downloads:0})

db.items.insert({'item_id':'LB0019', 'type':'ebook', 'title':"The Handmaid's Tale", 'author':['Margaret Atwood', 'Renée Nault'], pages:240, year:2019, 'genre':['Tragedy','Fiction'], 'publisher':'Penguin Books', 'url':'https://www.penguin.co.uk/books/1099258/the-handmaid-s-tale/9780224101936.html', downloads:0})

db.items.insert({'item_id':'LB0020', 'type':'ebook', 'title':'Black Sun', 'author':['Owen Matthews'], pages:336, year:2019, 'genre':['Thriller','Mystery'], 'publisher':'Penguin Books', 'url':'https://www.penguin.co.uk/books/1099258/the-handmaid-s-tale/9780224101936.html', downloads:0})

db.items.insert({'item_id':'LB0021', 'type':'ebook', 'title':'Airhead', 'author':['Emily Mattis'], pages:400, year:2019, 'genre':['Fiction','Drama'], 'publisher':'Penguin Books', 'url':'https://www.penguin.co.uk/books/309231/airhead/9781405938358.html', downloads:0})

db.items.insert({'item_id':'LB0022', 'type':'ebook', 'title':'Origin', 'author':['Dan Brown'], pages:560, year:2018, 'genre':['Crime','Fiction','Mystery'], 'publisher':'Penguin Books', 'url':'https://www.penguin.co.uk/books/1113984/origin/9780552174169.html', downloads:0})

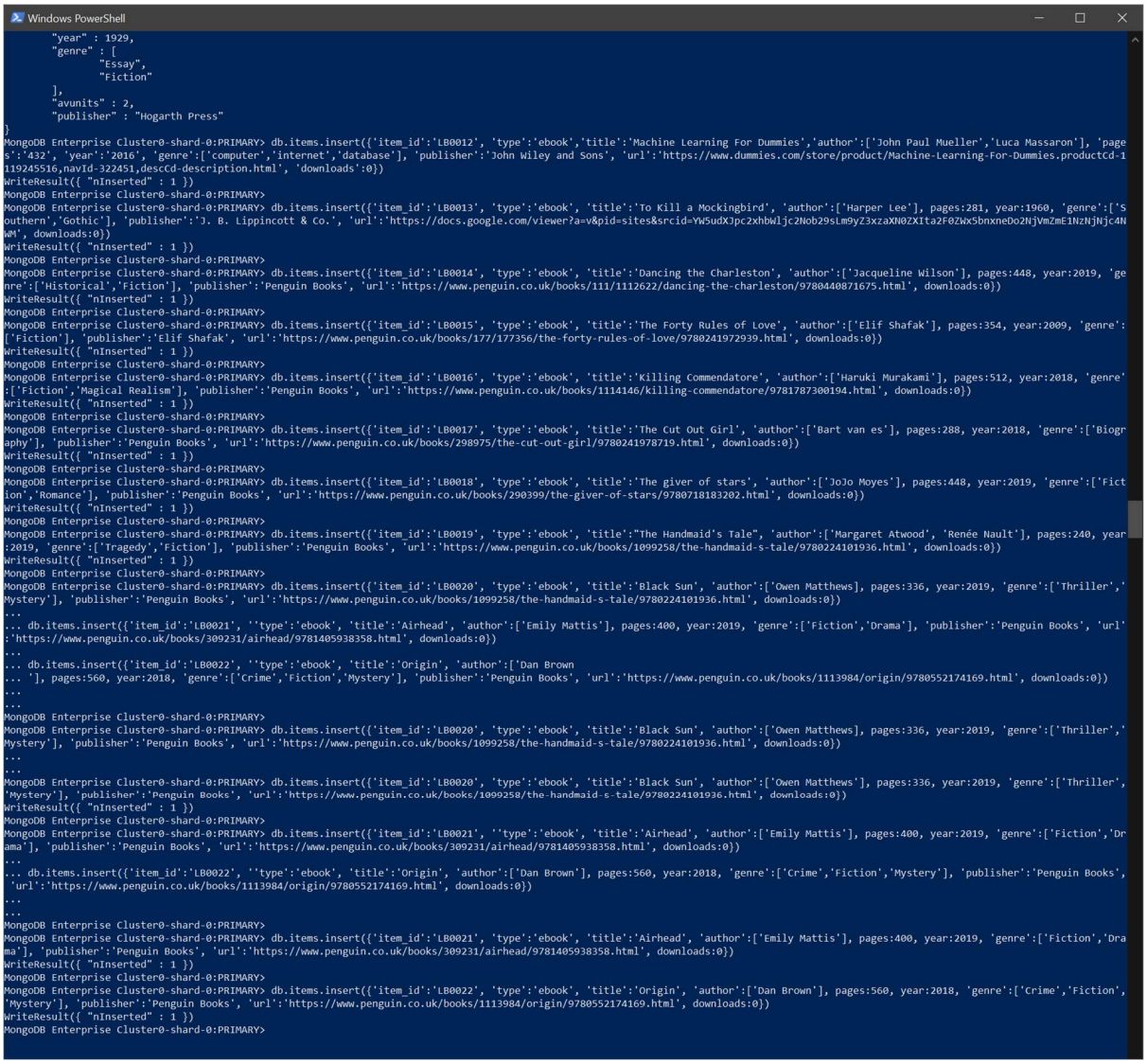

Figure 30 - Creating Item Document and Inserting item information 2

db.items.insert({'item_id':'LB0023', 'type':'magazine','title':'Times', 'issue':'5836596', 'genre':['Lifestyle'], 'avunits':2, 'publisher':'Jill E. Rancourt'})

db.items.insert({'item_id':'LB0024', 'type':'magazine','title':'US weekly', 'issue':'451962', 'genre':['Health'], avunits:2, 'publisher':'Marie J. Skipper'})

db.items.insert({'item_id':'LB0025', 'type':'magazine','title':'Country Living', 'issue':'a27020796', 'genre':['Lifestyle'], avunits:2, 'publisher':'Scarlett Bidwill'})

db.items.insert({'item_id':'LB0026', 'type':'magazine','title':'Vogue', 'issue':'20200501', 'genre':['Fashion'], avunits:2, 'publisher':'Matilda Weindorfer'})

db.items.insert({'item_id':'LB0027', 'type':'magazine','title':'Elle!', 'issue':'a31005669', 'genre':['Beauty', 'Fashion'], avunits:2, 'publisher':'Blake Ruwolt'})

db.items.insert({'item_id':'LB0028', 'type':'magazine','title':'Rolling Stone', 'issue':'949908', 'genre':['Lifestyle'], avunits:2, 'publisher':'Hudson De Satg'})

db.items.insert({'item_id':'LB0029', 'type':'magazine','title':'The New Yorker', 'issue':'2020518', 'genre':['Health'], avunits:2, 'publisher':'Andrew Carnarvong'})

db.items.insert({'item_id':'LB0030', 'type':'magazine','title':'Vanity Fair', 'issue':'20200401', 'genre':['Beauty'], avunits:2, 'publisher':'Ellie Akehurst'})

db.items.insert({'item_id':'LB0031', 'type':'magazine','title':'People', 'issue':'20520353', 'genre':['Lifestyle'], avunits:2, 'publisher':'Henry Tully'})

db.items.insert({'item_id':'LB0032', 'type':'magazine','title':'Wired', 'issue':'4142020', 'genre':['Lifestyle'], avunits:2, 'publisher':'Dean Fereday'})

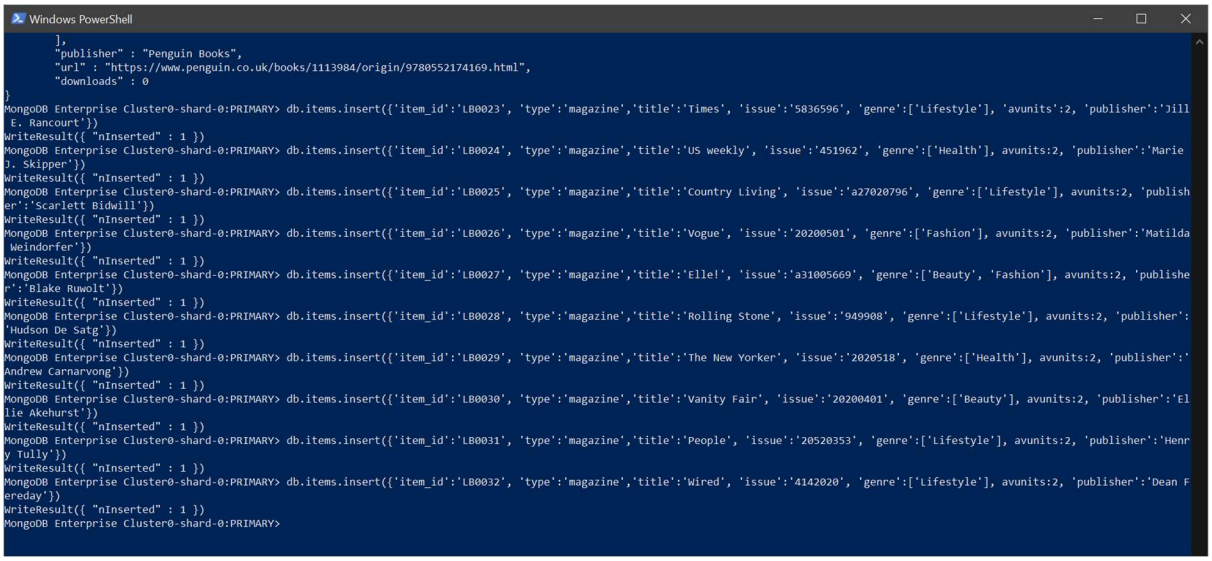

Figure 31 - Creating Item Document and Inserting item information 3

PUBLISHER DOCUMENT

db.createCollection("publishers")

db.items.createIndex({"name": 1 }, { unique: true })

Figure 32 - Creating Publisher Document

db.publishers.insert({'name':'HarperCollins Publishers', 'country':'UK', 'address':[{'street':'London Bridge'},{'city':'London'},{'pin':'SE1 9GF'}], 'phone':['020 8741 7070','0141 772 3200'], 'email':['contact@harpercollins.com', 'info@harpercollins.com']})

db.publishers.insert({'name':'John Wiley and Sons', 'country':'UK', 'address':[{'street':'Southern Gate'},{'city':'Chichester'},{'pin':'PO19 8SQ'}], 'phone':['01243 779777'], 'email':['customer@wiley.com']})

db.publishers.insert({'name':'Dutton Books', 'country':'USA', 'address':[{'street':'8879 South Fifth'},{'city':'Dr. Fair Lawn'},{'pin':'NJ 07410'}], 'phone':['660-525-6598'], 'email':['info@duttonbooks.com']})

db.publishers.insert({'name':'Alfred A. Knopf', 'country':'USA', 'address':[{'street':'405 6th Ave.'},{'city':'Howard Beach'},{'pin':'NY 11414'}], 'phone':['815-942-4425'], 'email':['info@alfredakonpf.com']})

db.publishers.insert({'name':"Charles Scribner's Sons", 'country':'USA', 'address':[{'street':'261 Vernon Street'},{'city':'East Elmhurs'},{'pin':'NY 11369'}], 'phone':['719-371-1604'], 'email':['nquiries@simonandschuster.co.uk']})

db.publishers.insert({'name':'Doubleday', 'country':'USA', 'address':[{'street':'6 Rockledge Rd.'},{'city':'East Elmhurst'},{'pin':'NY 11369'}], 'phone':['708-849-2918'], 'email':['ddaypub@randomhouse.com']})

db.publishers.insert({'name':'Barnes & Noble', 'country':'USA', 'address':[{'street':'79 Sherwood Rd.'},{'city':'Jamestown'},{'pin':'NY 14701'}], 'phone':['734-922-6235'], 'email':['barnesandnoble@mail.barnesandnoble.com']})

db.publishers.insert({'name':'Harper', 'country':'USA','address':[{'street':'9303 La Sierra Avenue'},{'city':'Central Islip'},{'pin':'NY 11722'}] , 'phone':['617-357-3788'], 'email':['info@harper.com']})

db.publishers.insert({'name':'Thomas Egerton', 'country':'USA', 'address':[{'street':'44 Washington Court'},{'city':'Bridgeport'},{'pin':'CT 06606'}], 'phone':['917-666-7981'], 'email':['janeaustengiftshop.co.uk']})

db.publishers.insert({'name':'Hogarth Press', 'country':'USA', 'address':[{'street':'21 Green Lake St.'},{'city':'North Brunswick'},{'pin':'NJ 08902'}] , 'phone':['6646-316-1300'], 'email':['apps@penguinrandomhouse.co.uk']})

db.publishers.insert({'name':'J. B. Lippincott & Co.', 'country':'USA','address':[{'street':'227 S. 6th Street'},{'city':'Lippincott'},{'pin':'CI 23411'}] , 'phone':['216-414-9428'], 'email':['hello@harpercollins.com']})

db.publishers.insert({'name':'Penguin Books', 'country':'UK', 'address':[{'street':'20 Vauxhall Bridge Rd'},{'city':'Westminster'},{'pin':'SW1V 2SA'}], 'phone':['120-625-6000'], 'email':['customersupport@penguinrandomhouse.co.uk']})

db.publishers.insert({'name':'Elif Shafak', 'country':'Turkey', 'address':[{'street':'2 Rockledge Rd'},{'city':'Turkey'},{'pin':'TK78098'}], 'phone':['998-23-731'], 'email':['info@elifsafak.com.tr']})

db.publishers.insert({'name':'Jill E. Rancourt', 'country':'USA', 'address':[{'street':'1758 Calvin Street'},{'city':'New York'},{'pin':'NY17 002'}], 'phone':['443-375-1862'], 'email':['illERancourt@jourrapide.com']})

db.publishers.insert({'name':'Marie J. Skipper', 'country':'Norway', 'address':[{'street':'Moen 174'},{'city':'PORSGRUNN'},{'pin':'3948'}], 'phone':['252-331-4418'], 'email':['marieJSkipper@jourrapide.com']})

db.publishers.insert({'name':'Scarlett Bidwill', 'country':'Finland', 'address':[{'street':'Kaarrostie 31'},{'city':'VANTAA'},{'pin':'01230'}], 'phone':['044-336-2683'], 'email':['scarlettBidwill@dayrep.com']})

db.publishers.insert({'name':'Matilda Weindorfer', 'country':'Canada', 'address':[{'street':'4910 rue des Églises Est'},{'city':'Cheneville'},{'pin':'QC J0V 1E0'}], 'phone':['819-428-0909'], 'email':['matildaWeindorfer@rhyta.com']})

db.publishers.insert({'name':'Blake Ruwolt', 'country':'Canada', 'address':[{'street':'49 Neilson Avenue'},{'city':'Toronto'},{'pin':'ON M1M 1V1'}], 'phone':['416-269-5145'], 'email':['blakeRuwolt@dayrep.com']})

db.publishers.insert({'name':'Hudson De Satg', 'country':'USA', 'address':[{'street':'3543 Carson StreetSan'},{'city':'Diego'},{'pin':'CA 92103'}], 'phone':['858-947-0401'], 'email':['hudsonDeSatg@jourrapide.com']})

db.publishers.insert({'name':'Andrew Carnarvong', 'country':'USA', 'address':[{'street':'3051 Hardman Road'},{'city':'Brattleboro'},{'pin':'VT 05301'}], 'phone':['802-812-7335'], 'email':['andrewCarnarvon@rhyta.com']})

db.publishers.insert({'name':'Ellie Akehurst', 'country':'UK', 'address':[{'street':'32 Tadcaster Rd'},{'city':'PITCAPLE'},{'pin':'AB51 1QF'}], 'phone':['077-764-4730'], 'email':['ellieAkehurst@rhyta.com']})

db.publishers.insert({'name':'Henry Tully', 'country':'Sweden', 'address':[{'street':'Idvägen 90'},{'city':'LENHOVDA'},{'pin':'90, 360 73'}], 'phone':['0474-764-9764'], 'email':['henryTully@dayrep.com']})

db.publishers.insert({'name':'Dean Fereday', 'country':'USA', 'address':[{'street':'4393 Coleman Avenue'},{'city':'Vista'},{'pin':'CA 92083'}], 'phone':['760-732-7972'], 'email':['deanFereday@dayrep.com']})

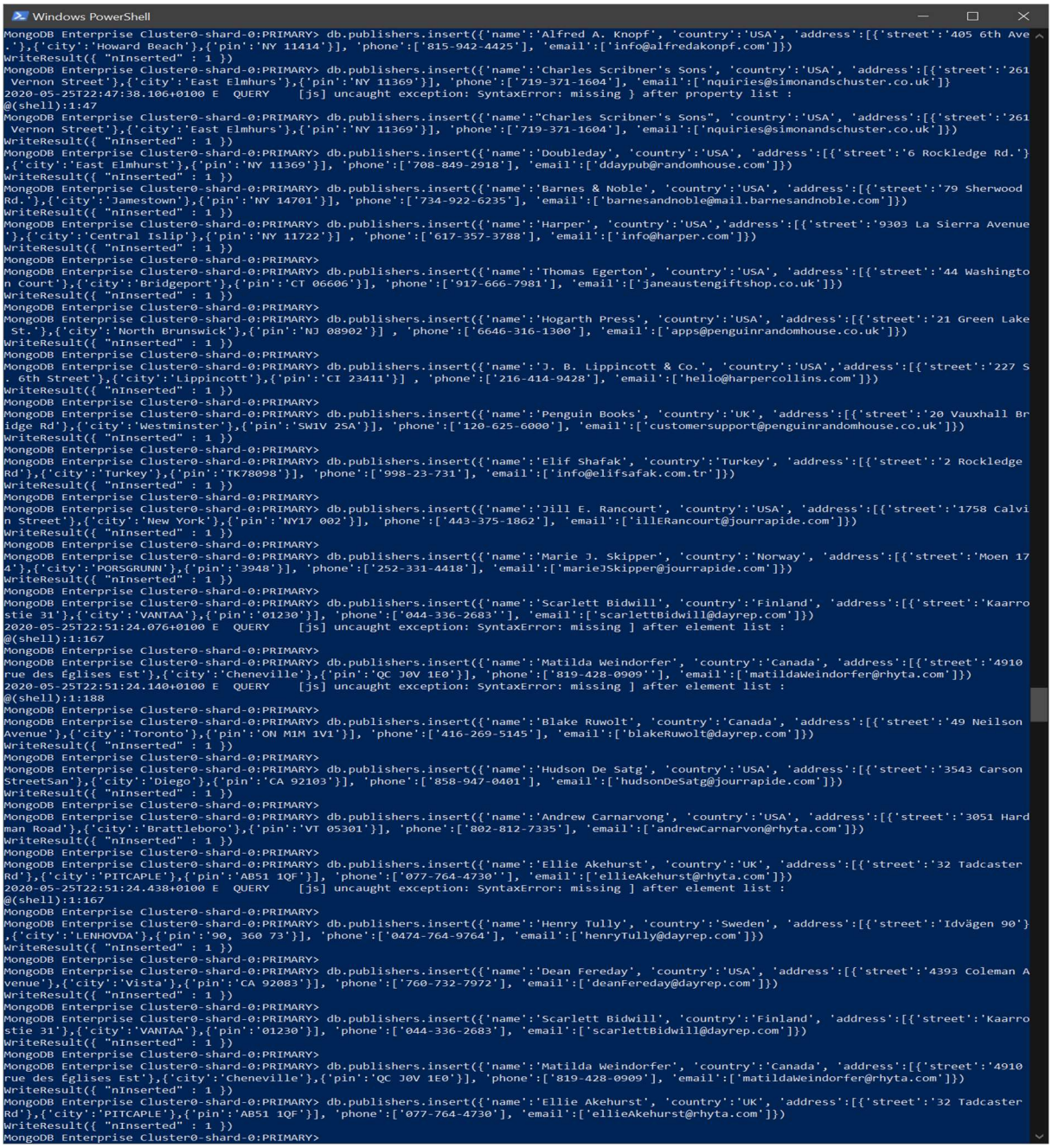

Figure 33 - Inserting Publisher data

FIVE USE CASES FOR MONGODB

1. **All item details from item document and publisher details from publisher document together for given title of item.**

```
db.items.aggregate([
{       $match: {
                "title": "Tiny Pretty Things" }
},
{       $project: {"_id": 0}
},
{       $lookup:{from: "publishers",localField:"publisher" ,foreignField: "name", as: "publisher"}
```

},
{ $unwind: "$publisher"
},
{ $project: {"publisher._id": 0}
}]).pretty()

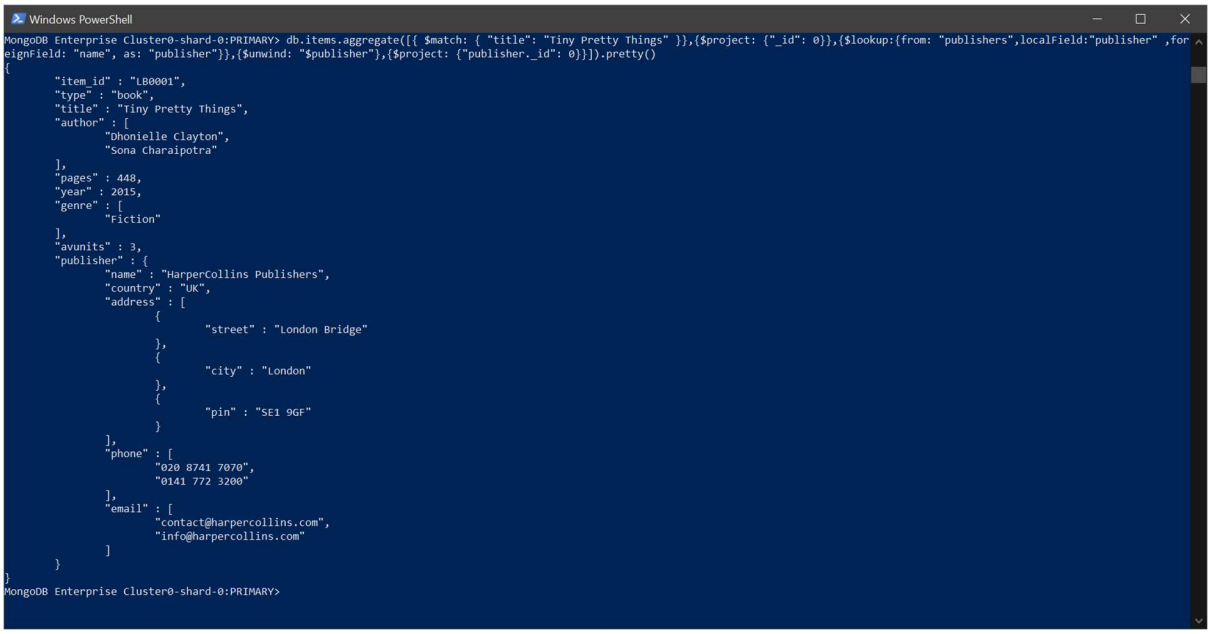

Figure 34 - MongoDB Query 1

2. **Books withdrawn by particular user along with title of item, due dates and withdraw dates.**

db.borrow.aggregate([
{ $match: { 'mID': 'R0001' }
},
{ $project:{"_id": 0, "mID" : 0}
},
{ $lookup:{from: "items",localField:"item_id",foreignField: "item_id",as: "ItemsWithdrawn"}
},
{ $unwind: "$ItemsWithdrawn"
},
{ $project: {"ItemsWithdrawn._id": 0, "ItemsWithdrawn.type": 0, "ItemsWithdrawn.pages": 0,
"ItemsWithdrawn.author": 0,"ItemsWithdrawn.genre": 0, "ItemsWithdrawn.avunits":0,
"ItemsWithdrawn.publisher":0, "ItemsWithdrawn.year":0, "ItemsWithdrawn.item_id":0}
}]).pretty();

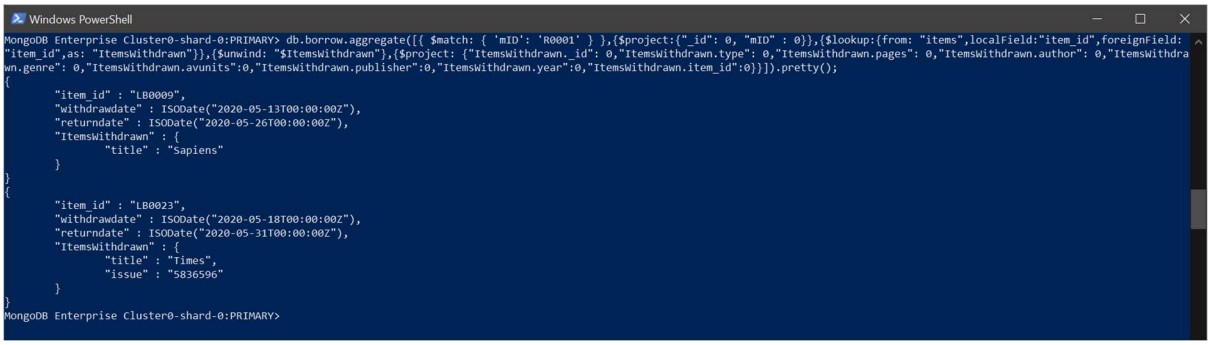

Figure 35 - MongoDB Query 2

3. **Shows names of all members who have overdue books along with title and due date of item.**

```
db.borrow.aggregate([
{       $match:{
                'returndate': {
                $lt:new Date()} }},
{       $lookup:
      {  from: "members",
         localField: "mID",
          foreignField: "mID",
          as: "Member"
      }},
   {    $lookup:
      {  from: "items",
         localField: "item_id",
         foreignField: "item_id",
         as: "bookname"
      }},
{       $group:{
                _id: "$mID",
                "name":{$first:"$Member.name"},
                'bookid':{      $addToSet:"$item_id"},
                                'itemname':{ $addToSet:"$bookname.title"},
                                'rdate':{ $addToSet:"$returndate"},
                }}      ]).pretty()
```

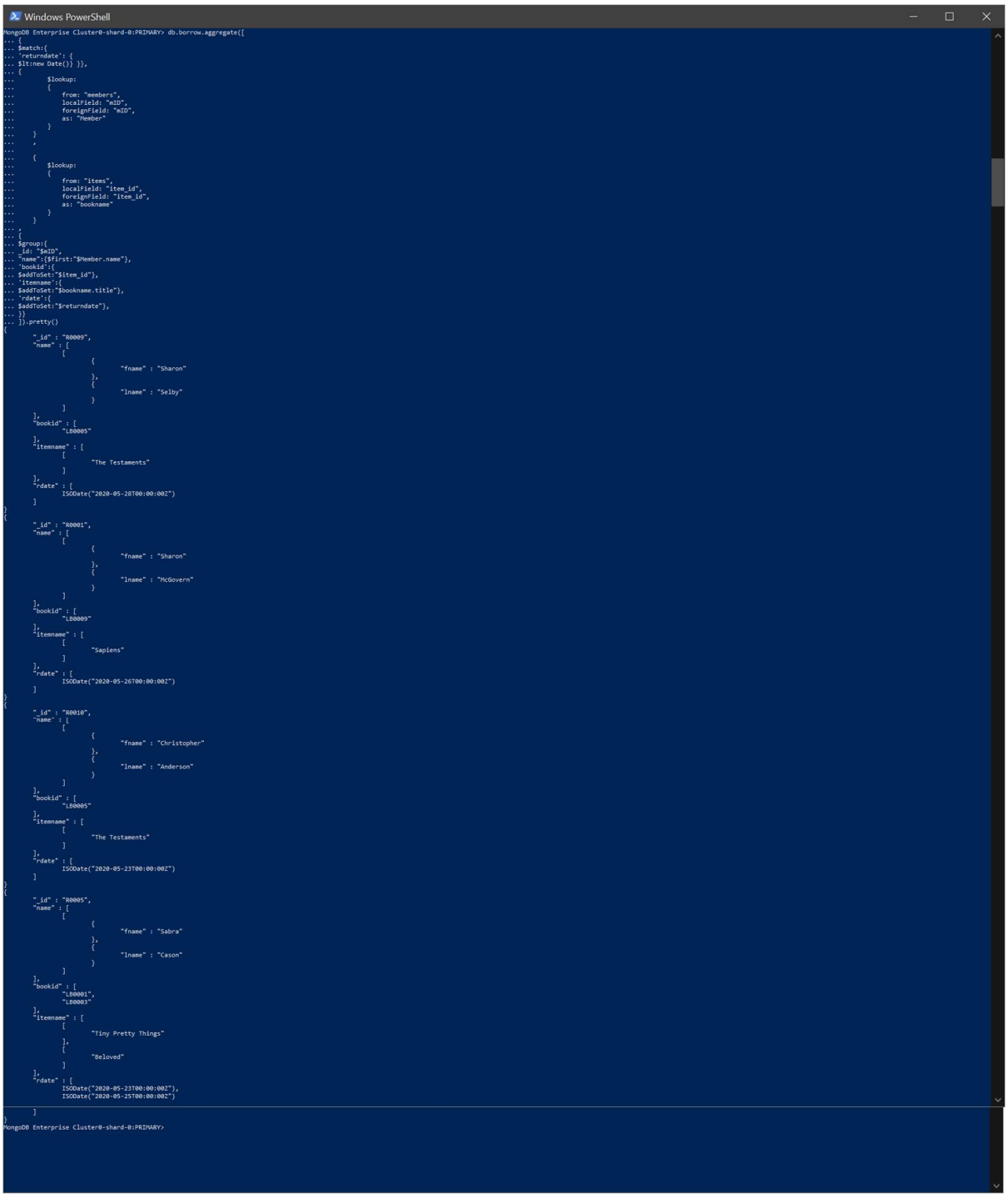

Figure 36 - - MongoDB Query 3

4. **Gives names of all members currently having 2 or more items.**

```
db.borrow.aggregate(
{       $lookup:{from: "members",localField:"mID",foreignField: "mID",as: "Members"}},
{       $group : { _id : '$mID', "UserName":{$first:"$Members.name.fname"} ,
        count : {$sum : 1}}},
{       $match: { "count": { $gte: 2 } }
}).pretty()
```

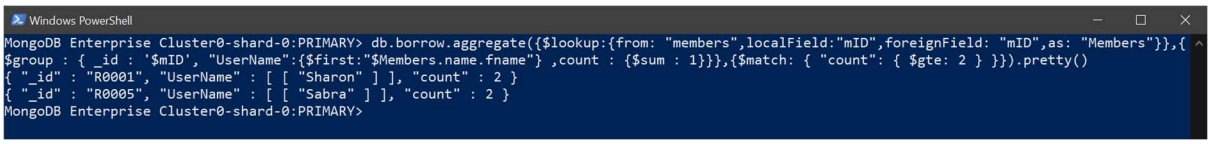

Figure 37 - MongoDB Query 4

5. **The query to withdraw items after doing the necessary checks(if item is available and whether member has less than 5 books currently withdrawn) by giving member ID and Item ID as input**

```
function withdrawbook(item_idw,user_id){
... var flag='Success';
... var a=new Array;
... db.items.find({'item_id':item_idw}). forEach(function(doc1){a.push(doc1.avunits)});
... var b=db.borrow.countDocuments({'mID':user_id});
... if(a[0]>0) {
... if(b<5){
... db.borrow.insert({'mID':user_id, 'item_id':item_idw, "withdrawdate" :new Date(ISODate().getTime()),
"returndate" :new Date(ISODate().getTime() + 1000 * 86400 * 14)});
... db.items.updateOne({'item_id':item_idw},{$inc:{'avunits':-1}});
... }
... else
... {
... flag='Too Many Books Withdrawn';
... }
... }
... else {
... flag='Book Unavailable';
... }print(flag);}
        withdrawbook('LB0002','R0001')
```

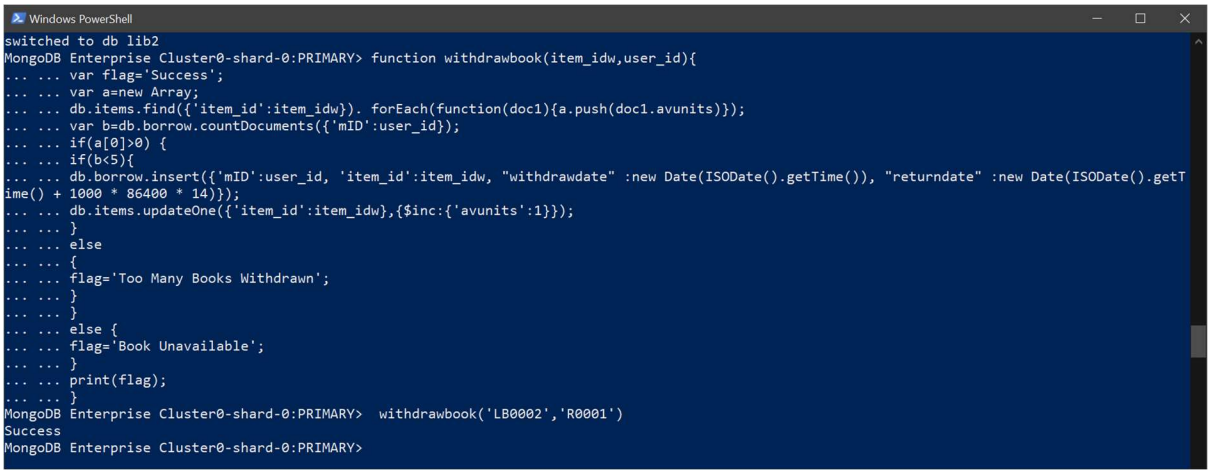

Figure 38- MongoDB Query 5

TASK B:

BRIEF DESCRIPTION

For this task, the team selected Weka as a machine learning algorithm tool and Epileptic Seizure Recognition (ESR) dataset for the classification exercise. The dataset was presented in a .csv format, hence it was necessary to convert to a compactible file format (.arff), which was done in Weka using the ArffViewer tool.

On first inspection of the dataset in Weka, the following observations were made:

a. It has 180 attributes and 11500 instances.
b. The attribute y has 5 classes {1,2,3, 4, 5}; 1 is classed as Epileptic Seizure and 2,3,4, 5 are classed as Non-Epileptic Seizure.
c. Each class has equal spread of 2300 instances.
d. The y attribute was numeric; therefore it was converted to nominal so that we could carry out classification on the dataset.

The interest of the team is in classifying Epileptic Seizure (1) vs Non-Epileptic Seizure (2,3,4,5), a binary classification approach was adopted. Using python, the classes (2,3,4,5) were combined into a single class and given a value 0. *(See Appendix 1 for Python Code)*

The transformed dataset was then uploaded for further inspection, and it was observed that the new dataset become unbalance as class 0 has 9200 instances and class 1 has 2300, which is significantly high as class 0 is now 4 times as much as class 1, and this might possibly have an impact on the classification process.

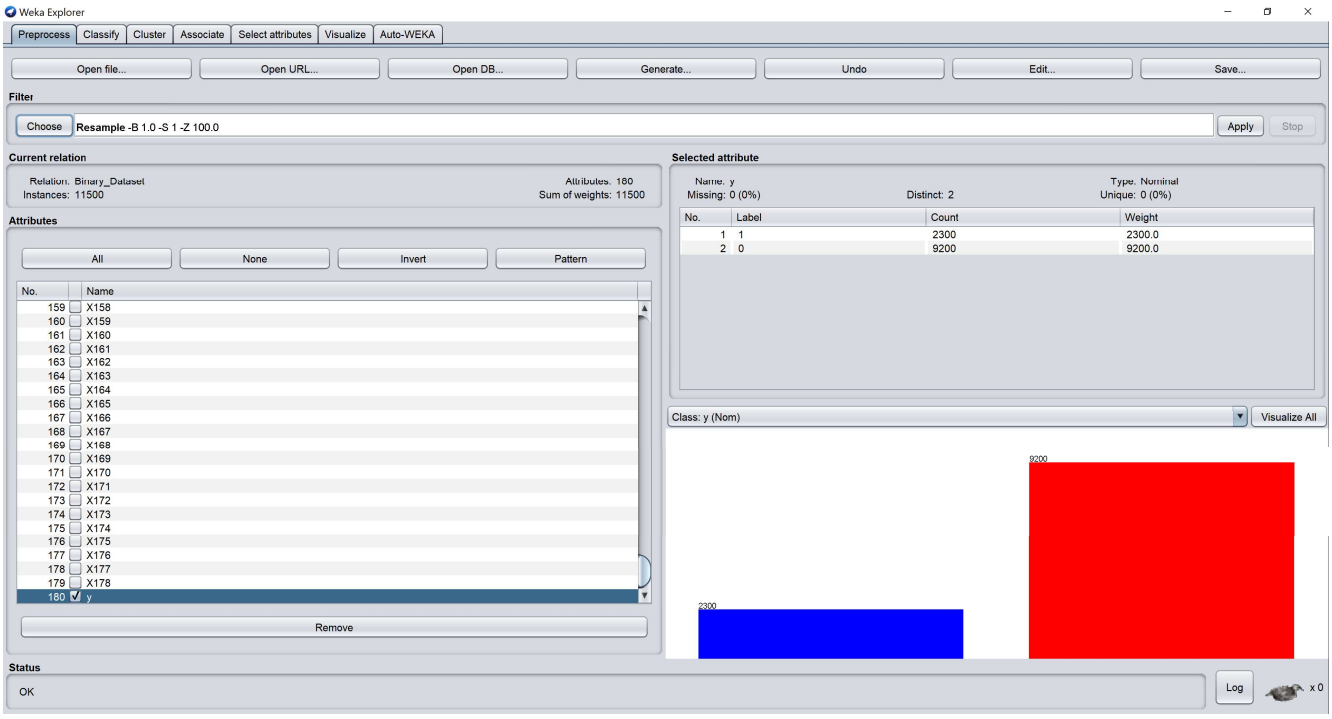

Figure 39 - First inspection

DEFINING TRAIN AND TEST DATASET

The Train and Test dataset were both extracted from the ESR dataset, which has a total instance of 11,500. The ratio of (80%:20%) was used to split the dataset. 80% was taken for Train and 20% was reserved for Test. This was done manually using Notepad ++. The first 9200 instances (80%) of the data was taken for training and the remaining 2300 instances (20%) was kept for test. On inspection of the Train and Test dataset in Weka, it was observed that class 1 has 1835 instances and class 0 has 7365 instances which is an approximate ration of (1:4)

and the Test dataset also has an approximate ratio of (1:4) with 465 instances for class 1 and 1835 instances for class 0. This shows that the ratio of data spread is not too different for the original ERS dataset. Figure 2 and 3 show the data spread in Weka. *(See Appendix 2, 3, 4 for ERS, Train and Test dataset)*

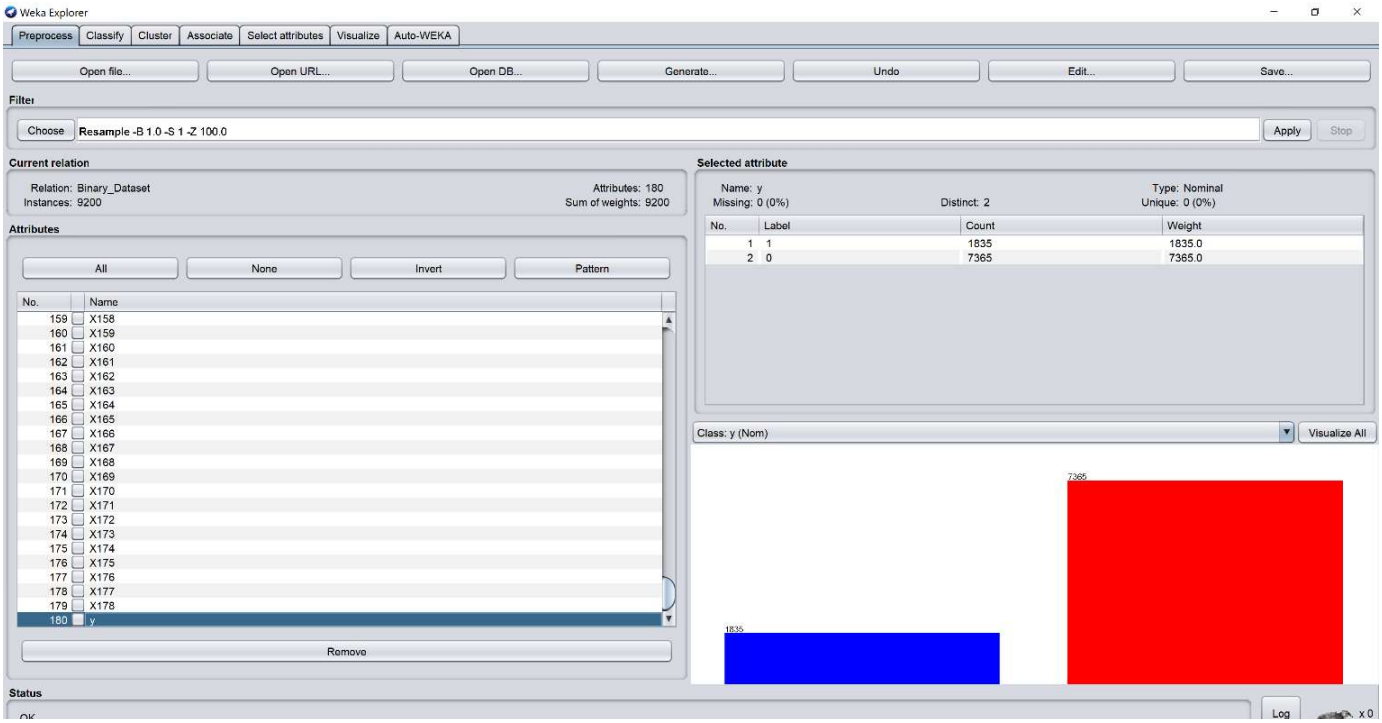

Figure 40– Spread of Data in Train Dataset

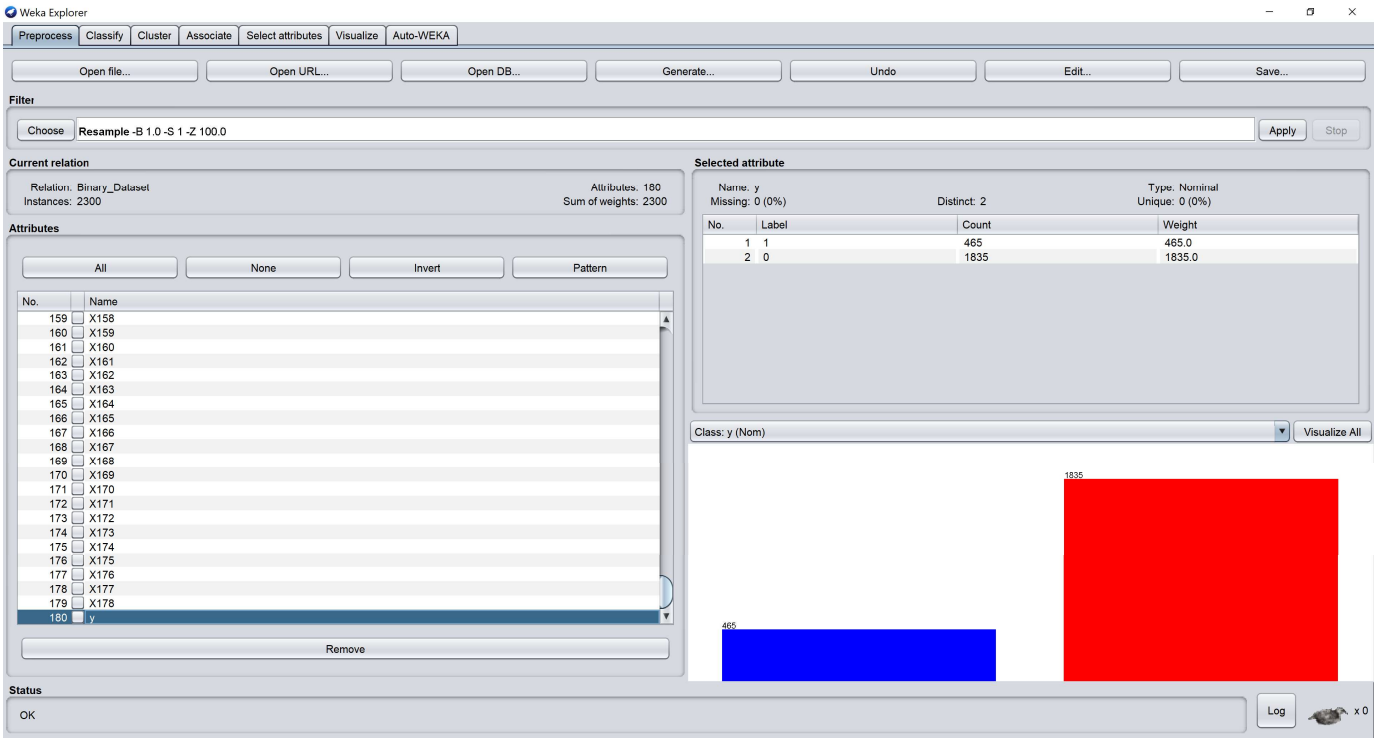

Figure 41 - Spread of Data in Test Dataset

SELECTION AND PERFORMANCE COMPARISON OF TWO CLASSIFICATION ALGORITHM

SELECTION

The Two (2) selected algorithm are the **k-Nearest Neighbor (k-NN) algorithm** and **Decision Tree algorithm** because they are both simple to implement and are widely used for solving classification and regression problem.

k-NN algorithm is a supervised machine learning algorithm which solves a problem by assuming that things with similar features exist in proximity to each other. In k-NN classification, the output is a class membership, it is classified by the majority vote of its neighbors where k (number of closest neighbor) can range from 1 and above.

Decision Tree algorithm is also a supervised machine learning algorithm that tries to solve a problem using tree representation, where each internal node of the tree corresponds to an attribute and each leaf node corresponds to a class.

PERFOMANCE COMPARISON

We started by using the selected algorithm to perform classification and build a model on the Train Dataset. As shown in Figure 2, the Train dataset is an unbalanced dataset which has a spread ratio of (1:4) with Epileptic Seizure with 1835 instances and Non-Epileptic Seizure with 7365 instances. The following exercise were carried out; k-NN Cross Validation (Fold = 10), k-NN with 66% Train-Test Split, Decision Tree Cross Validation (Fold = 10), Decision Tree with 66% Train-Test Split.

k-NN ALOGORITHM

k-NN CROSS VALIDATION (Folds = 10)

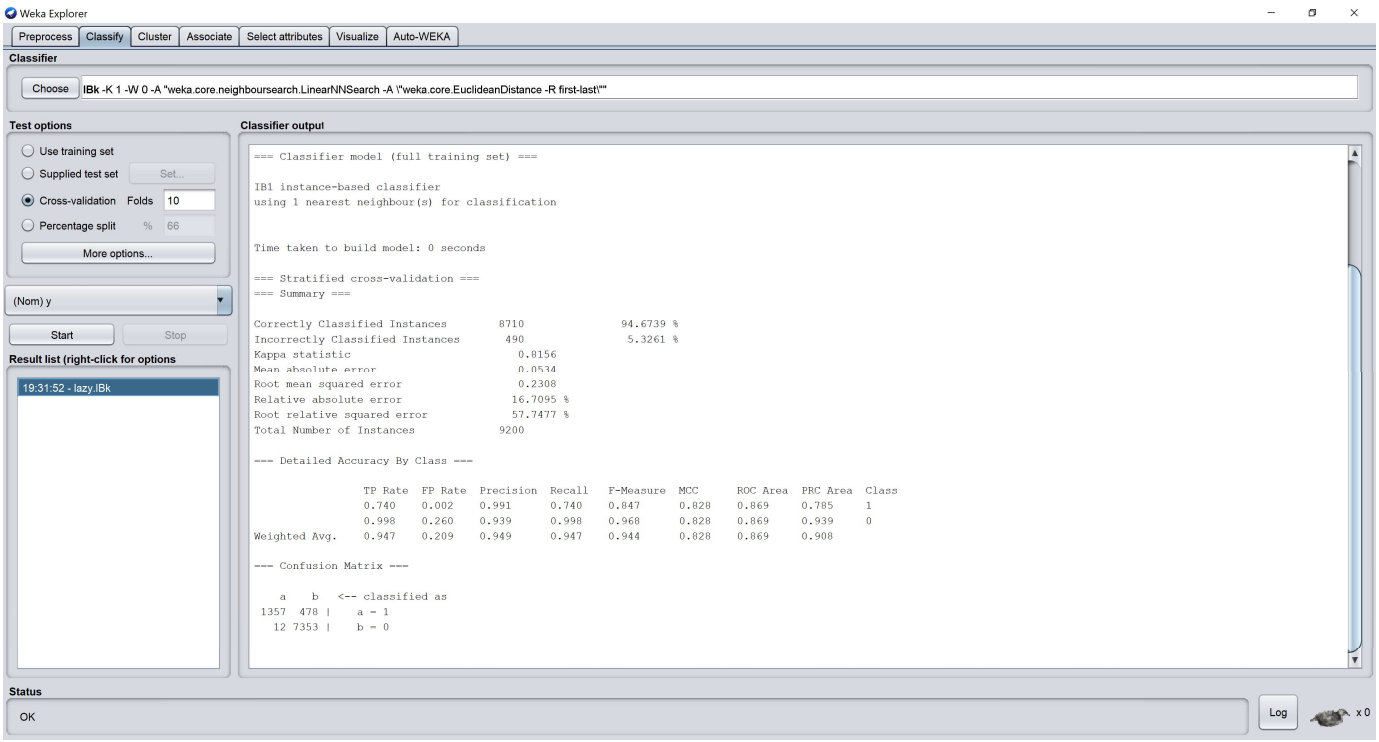

Figure 42 - k-NN Cross Validation (Fold = 10), (k=1) Report

For the experiment, different values of k were selected;

k-Values	Correctly Classified Instances	Area Under Curve (AUC)	Precision	Recall	True Positive Rate
1	94.6739%	0.869	0.949	0.947	0.947
3	93.1196%	0.902	0.936	0.931	0.931
5	92.4022%	0.914	0.930	0.924	0.924

On inspect of the k- value comparison table above, we could see that k = 1 has the highest score in Correctly Classified Instances with 94.6739%, Precision value of 0.949, recall value of 0.947 and true Positive Rate of 0.947 but went on a downward trail for k=3 and k=5. But the Area under Curve value was highest for k= 5 with 0.914 but went down as we reduced the value of k. Looking at the correctly classified values for each classes, we could see that the best result was achieved with k= 1, but continue to reduce as we increase k, this is likely because we have an unbalanced data of ratio (1:4), we have more Non-Epileptic Seizure that Epileptic Seizure, so on increment of value of k, the wrong nearest neighbor was been assumed. Based on this analysis, the value of k = 1 was adopted for the rest of the experiment.

k-NN with 66% TRAIN-TEST SPLIT

A Test-Train Percentage split was also carried out using k-NN algorithm with value of k = 1 and a percentage split of 66% on the Train Dataset. In the building of this model, 6072 instances (66%) was used for training and 3136 instances (34%) was used for testing, the result showed that the Correctly Classified instances is 95.0128%, Precision is 0.952, Recall is 0.950, Area under Curve is 0.869 and True Positive Value of 0.950. This method gave slightly better result compared to Cross Validation, but the Area under Curve value is the same.

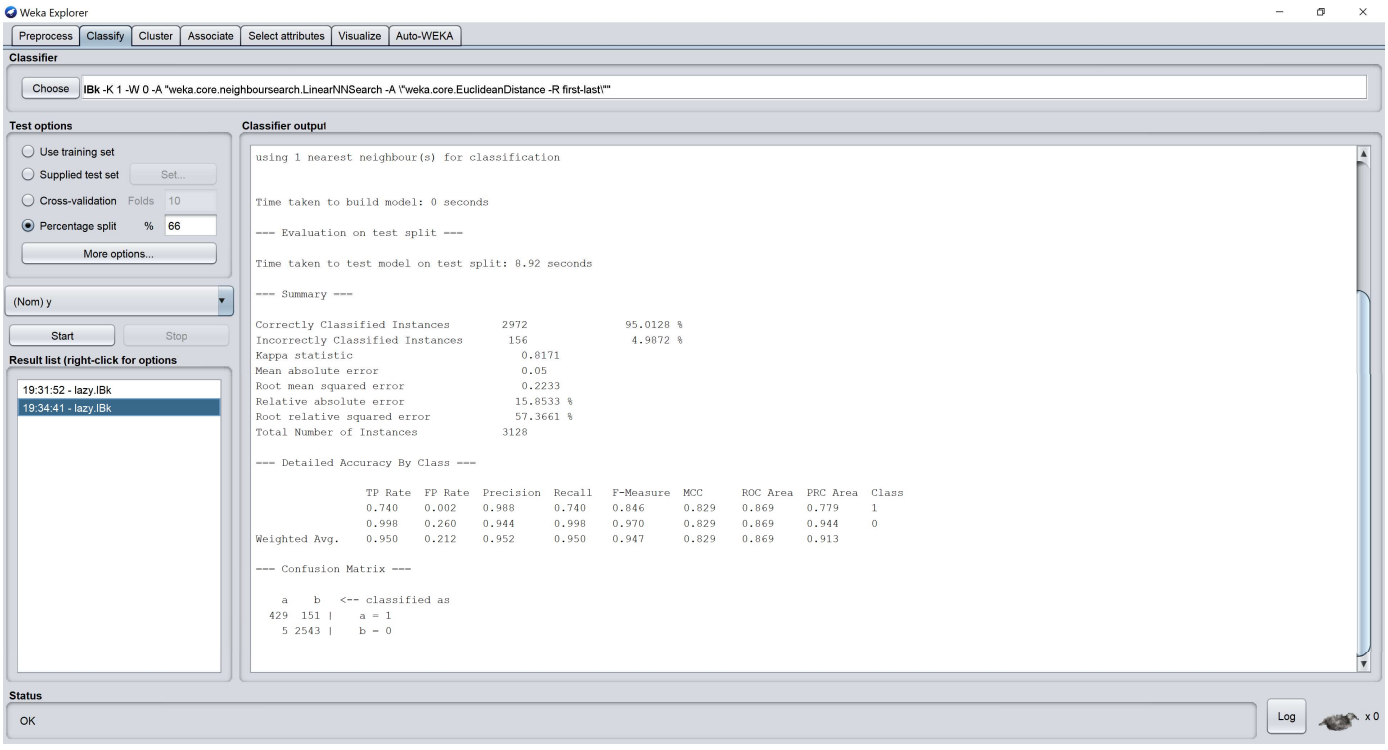

Figure 43 - k-NN with 66% Train-Test Split Report

USING BUILT k-NN ALGORITHM MODEL ON TEST DATASET

We introduced an unseen dataset, which is the Test Dataset extracted from the ESR Dataset to check the performance of the built model will perform, there a total of 2300 instances distributed to a ratio of (1:4). Epileptic

Seizure has 465 instances and Non-Epileptic Seizure has 1835 instances. The result showed the Correctly Classified instances is 94.3478%, Precision is 0.947, Recall is 0.943, Area under Curve is 0.862 and True Positive Value of 0.943. Given that this was done on unseen data, the classified instances were quite good. On closer inspect of the Confusion Matrix, the Non-Epileptic Seizure were well classified with only 2 wrongly classified but the 128 instances (27.5%) member of Epileptic Seizure were wrongly classified. This is likely cause because we have an unbalance dataset where Non-Epileptic has more instances, so the model has better knowledge about them.

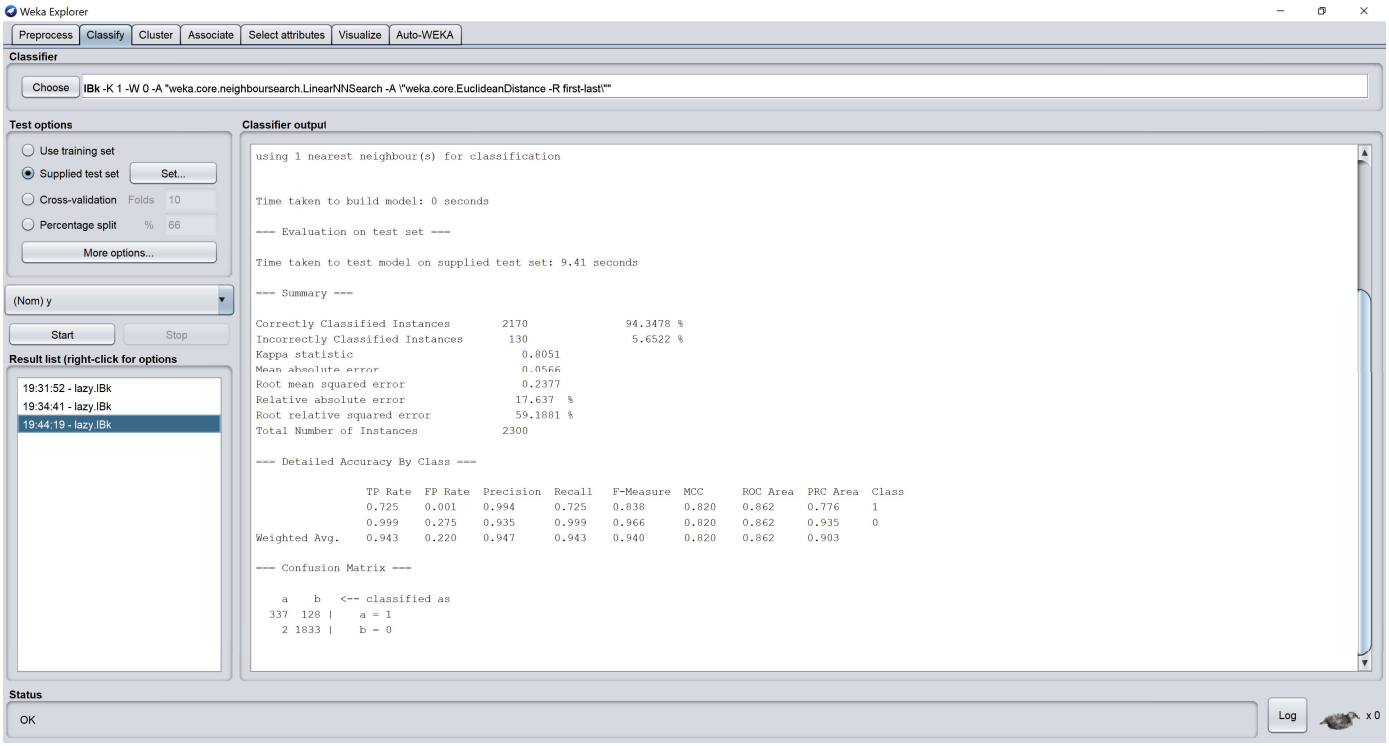

Figure 44 - k-NN Model on Test Dataset Report

DECISION TREE ALGORITHM

DECISION TREE CROSS VALIDATION (Folds = 10)

The Train Dataset was also used for building a decision tree classification model, first the Cross validation method with 10 folds was used, the result showed that the Correctly Classified instances is 94.3696%, Precision is 0.943, Recall is 0.944, Area under Curve is 0.887 and True Positive Value of 0.944.

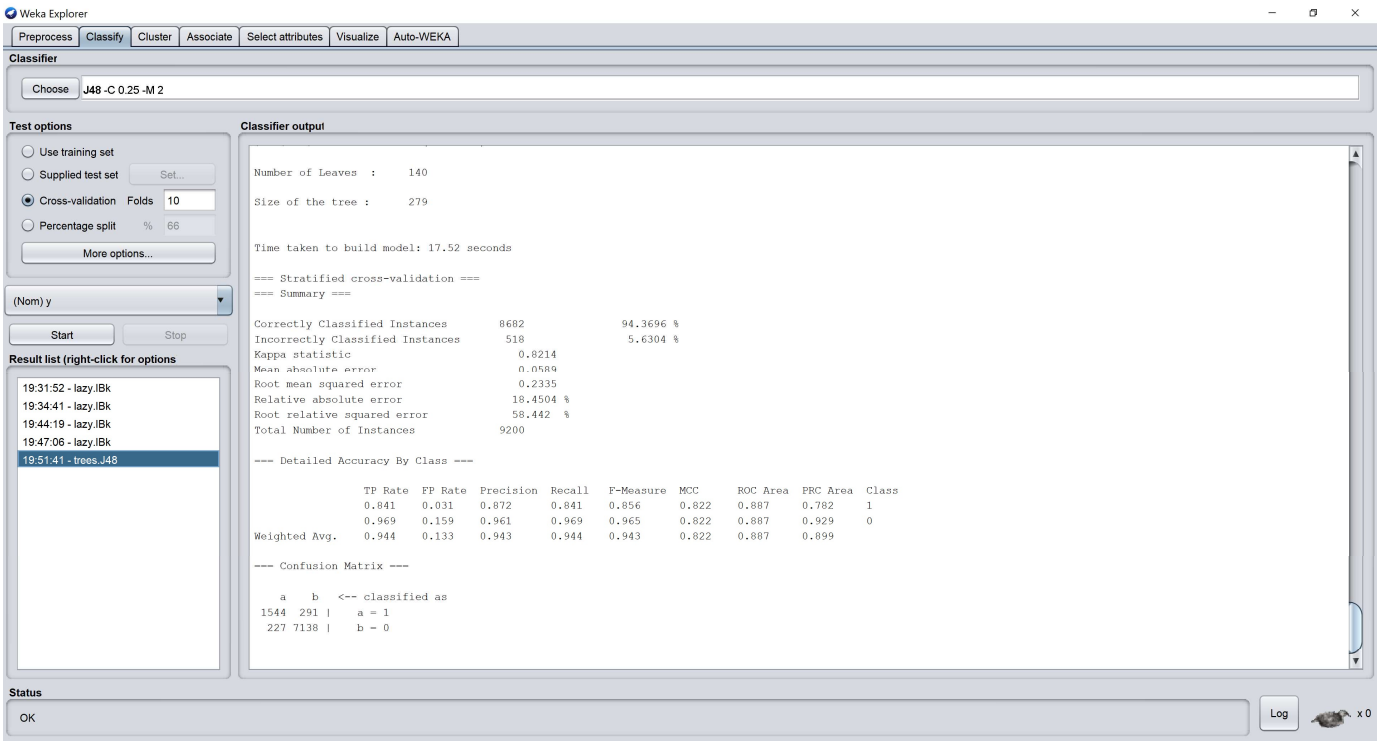

Figure 45 - Decision Tree Cross Validation (Folds = 10) Report

DECISION TREE WITH 66% TRAIN-TEST SPLIT

The Train-test Split method with 66% split was also carried out and the result showed that the Correctly Classified instances is 94.4373%, Precision is 0.944, Recall is 0.944, Area under Curve is 0.888 and True Positive Value of 0.944.

This method gave slightly better result compared to Cross Validation, but the Area under Curve value is the same

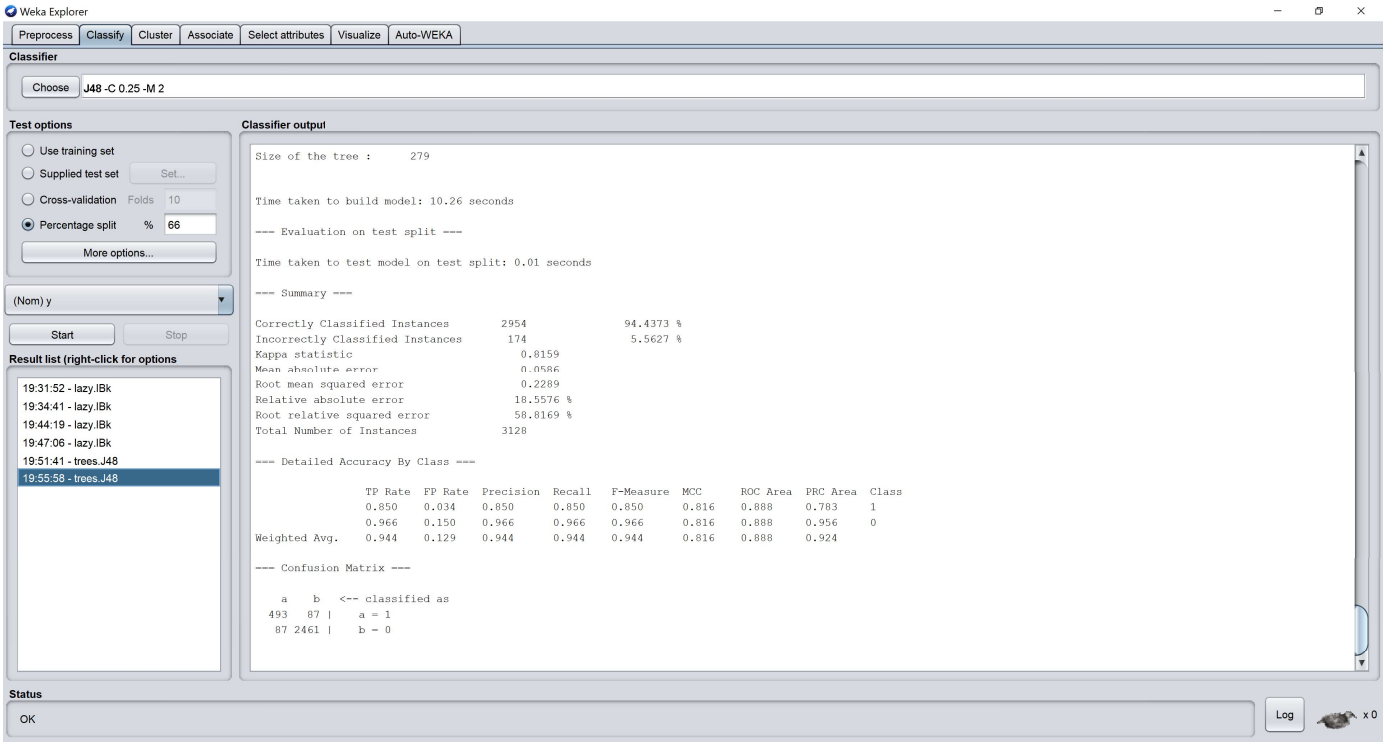

Figure 46 - Decision Tree with 66% Train-Test Split Report

USING BUILT DECISION TREE ALGORITHM MODEL ON TEST DATASET

Finally the model was used to classify the instances into classes in the Test Dataset and the result is as follow; Correctly Classified instances is 94.3913%, Precision is 0.943, Recall is 0.944, Area under Curve is 0.873 and True Positive Value of 0.944. The Confusion matrix showed that the model classified Epileptic Seizure instances better that k-NN with 15% wrongly classified but it did not do as well as the k-NN model with the Non-Epileptic Seizure having 58 instances classified wrongly.

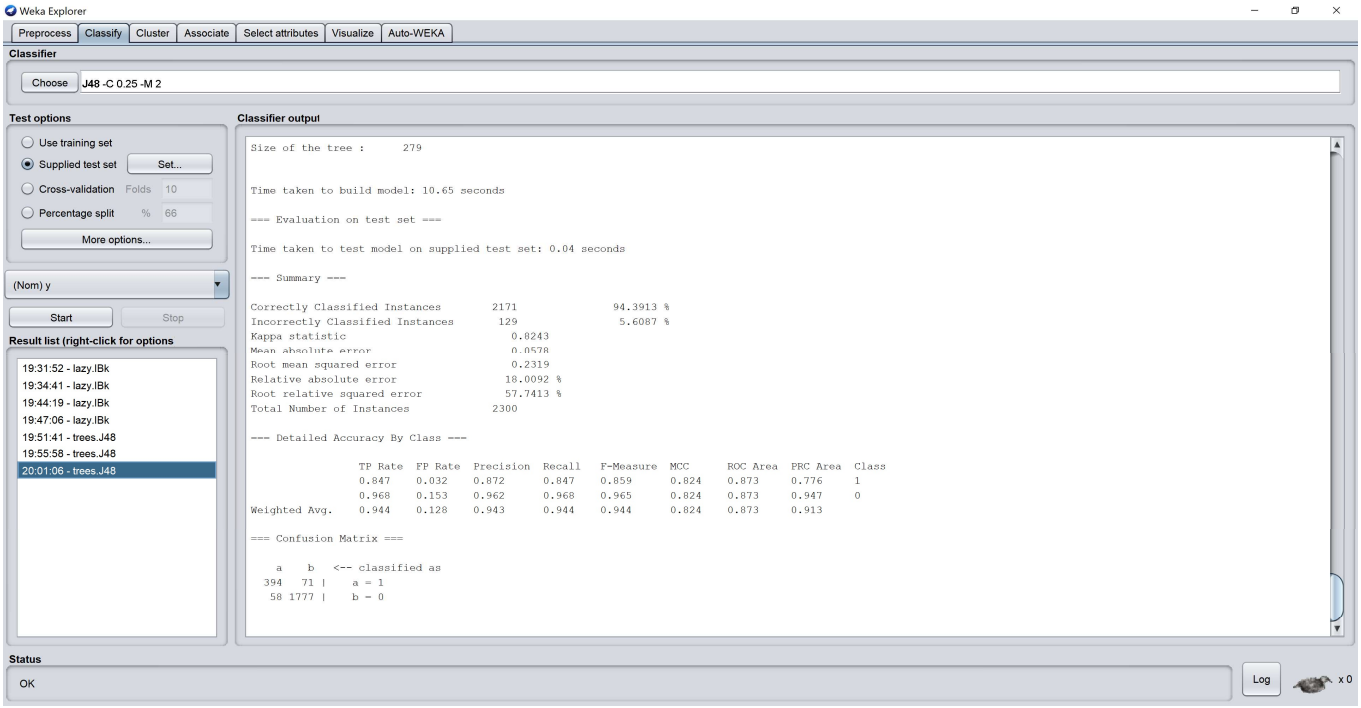

Figure 47 - Decision Tree Model on Test Dataset Report

TABULAR PRESENTATION OF COMPARISON

Comparison Factors	Cross Validation (fold = 10)		Train -Test Split (66%)		Test Dataset (Unseen Data)	
	k-NN	Decision Tree	k-NN	Decision Tree	k-NN	Decision Tree
Correctly Classified instances (%)	94.6739	94.3696	95.0128	94.4373	94.3478	94.3913
Precision	0.949	0.943	0.952	0.944	0.947	0.943
Recall	0.947	0.944	0.950	0.944	0.943	0.944
AUC	0.868	0.887	0.869	0.888	0.862	0.873
True Positive	0.947	0.944	0.950	0.944	0.943	0.944

The overall table showed that both algorithms did quite well in all given scenarios, despite the fact that we had an unbalanced class in our dataset, they both had over 90% correctly classified instances. They all had over 0.900 score in the Precision, Recall, True Positive and over 0.850 in the AUC calculation. On visual inspection, we cannot conclude that one is better than the other as they both have their strength and weakness. k-NN algorithm did a better classification of Non-Epileptic Seizure on the Unseen dataset while Decision tree algorithm did a better classification of Epileptic Seizure on the unseen dataset.

SIGNIFICANT TEST - CONFIDENCE INTERVAL USING AREA UNDER CURVE

A significant test using the Area under Curve (AUC) value of selected algorithm (k-NN and Decision Tree) on the Test Dataset (Unseen) using the formula below was carried out:

$$error \pm Z \cdot \sqrt{\left(\frac{error \cdot (1 - error)}{n}\right)}$$

n = number of instances

Z = 1.96 (95% confidence)

	k-NN	**Decision Tree**
n	2300	2300
Z	1.98	1.98
AUC	0.862	0.873
+ or -	0.0140956571 approx. (0.014)	0.0136082167 approx. (0.014)
+ value	0.876	0.887
- value	0.848	0.859

The test showed that the AUC for k-NN has a margin error of approximately +or- 0.014, which means it can fall between 0.848 to 0.876 while Decision Tree also has a margin error of approximately +or- 0.014, which means it can fall between 0.859 to 0.887. Although there is an overlap in the value, but Decision Tree has a slight better performance than the k-NN.

CLASS BALANCING

As it is evident that we have an unbalance dataset, and this might have an impact while building our machine learning model, the 3 different methods of class balancing was carried out; Undersampling the majority class, Oversampling the minority class and Synthetic Minority Oversampling Technique (SMOTE), to check if this will have an impact of a better built model.

Using Weka, which provides a Functionality called Resample (Oversampling), SpreadSubsample (Undersampling) and SMOTE (SMOTE) under Filter in the Preprocess Tab, the process of class balancing was done.

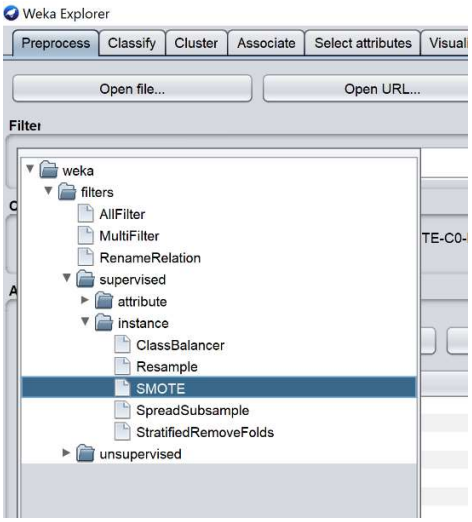

Figure 48 - Weka Filter for Class Balancing

UNDERSAMPLING THE MAJORITY CLASS

This is the method of the undersampling the major class (class 0) to have the same number of instances as the minor class (class 1). We used the SpreadSubsample function with the distributionSpread parameter set to 1. This is resulting dataset has a total instance of 3670 with a ratio (1835:1835).

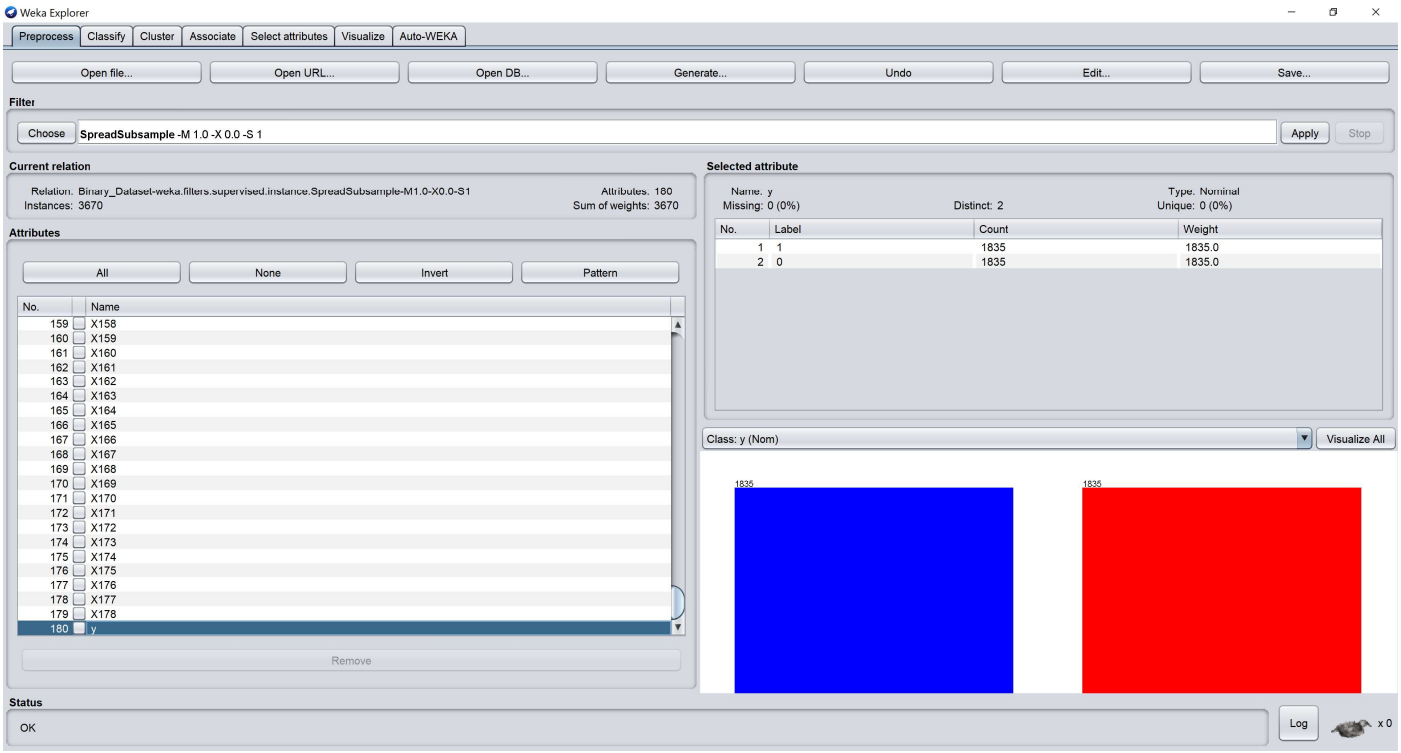

Figure 49 - Undersampled Dataset

Decision Tree and kNN algorithms with a value of 1 and cross validation (fold = 10) were used to train the undersampled dataset

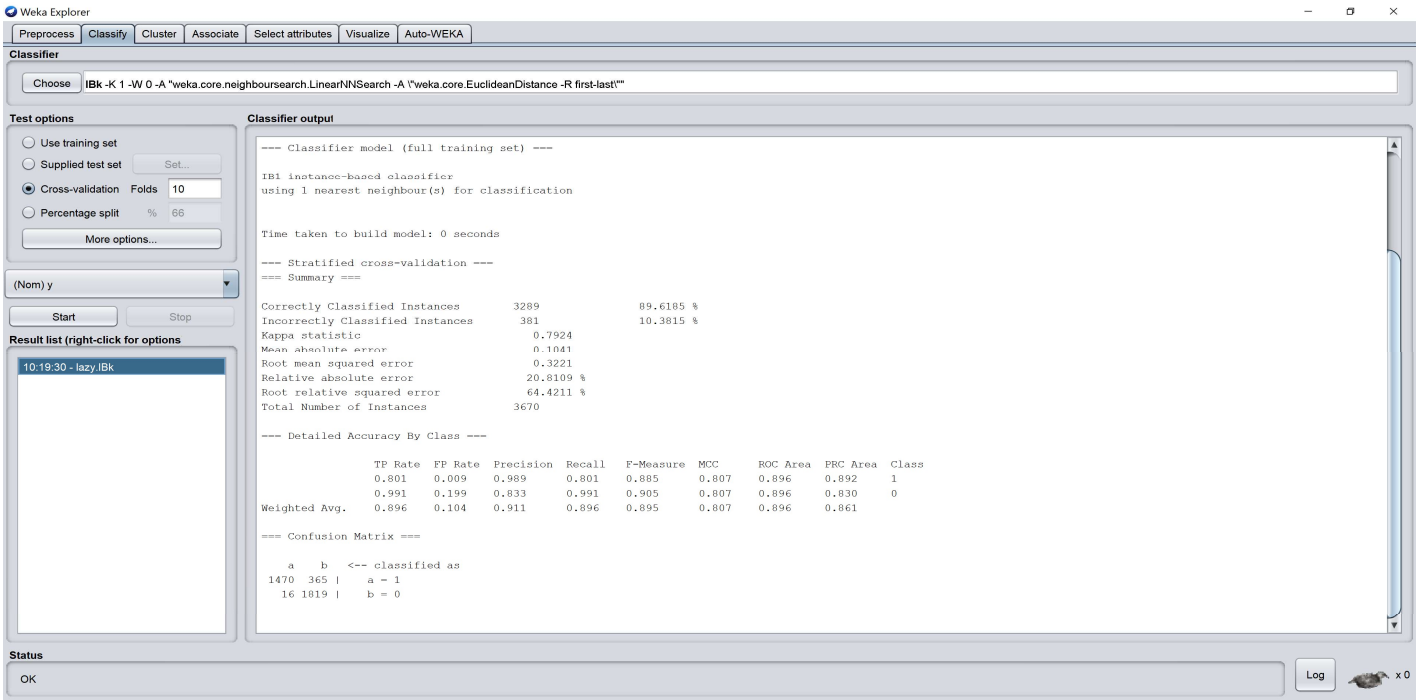

Figure 50 - kNN on Undersampled Dataset

k-NN with a k value of 1 showed a decrease in the correctly classified instances with a percentage of 89.6185, Precision of 0.911, Recall value of 0.896, AUC of 0.896 and a True Positive rate of 0.896.

The Decision Tree also showed a decrease in the correctly classified instances with a percentage of 91.4714, Precision of 0.915, Recall value of 0.915, AUC of 0.903 and a True Positive rate of 0.915.

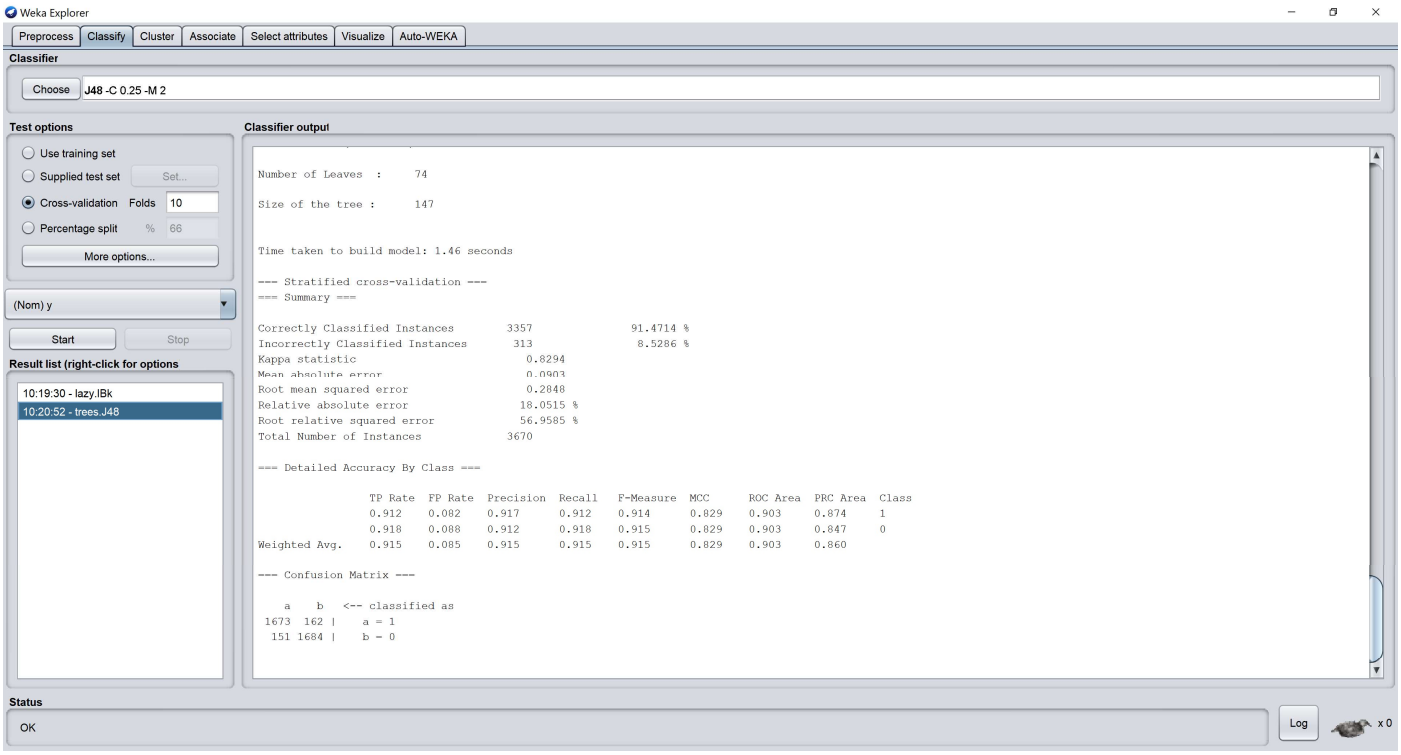

Figure 51 - Decision Tree on Undersampled Dataset

OVERSAMPLING THE MINORITY CLASS IN THE DATASET

This is the method of the oversampling the minority class (class 1) to have the same number of instances as the majority class (class 1). We used the Resample function with the biasToUniformClass parameter set to 1. This is resulting dataset has a total instance of 9200 with a ratio (4600:4600).

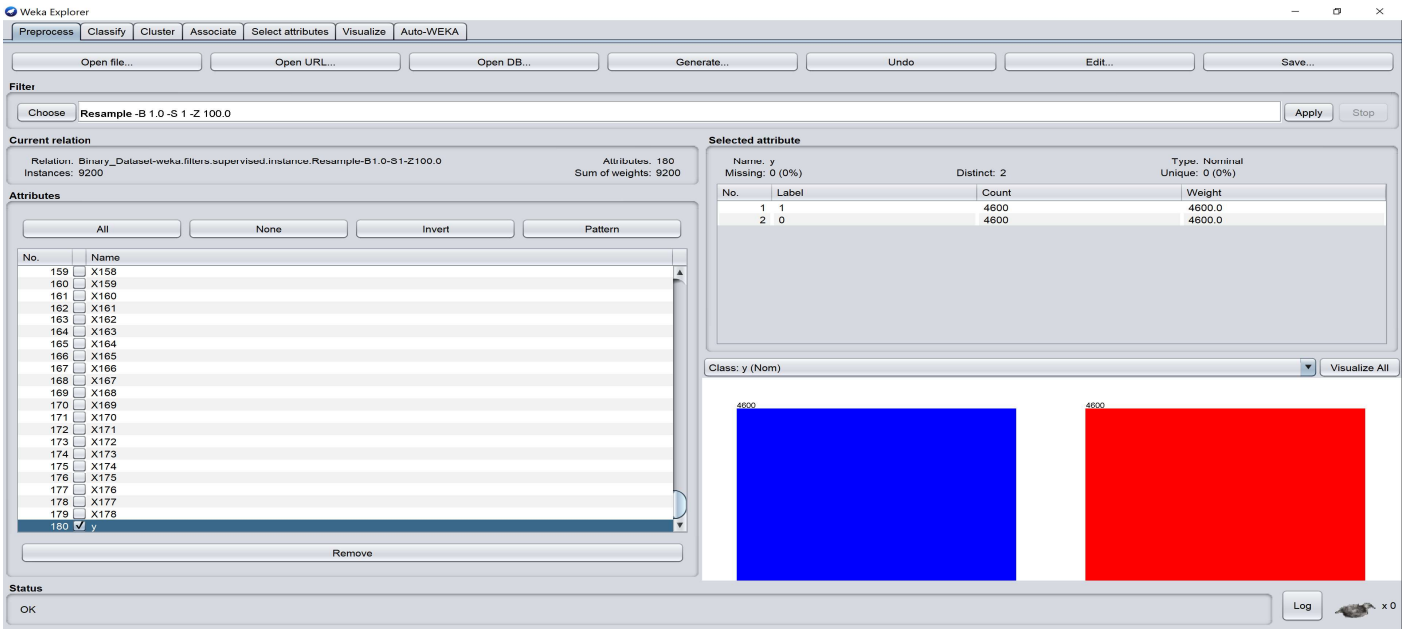

Figure 52 - Oversampled Dataset

Decision Tree and kNN algorithms with a value of 1 and cross validation (fold = 10) were used to train the oversampled dataset

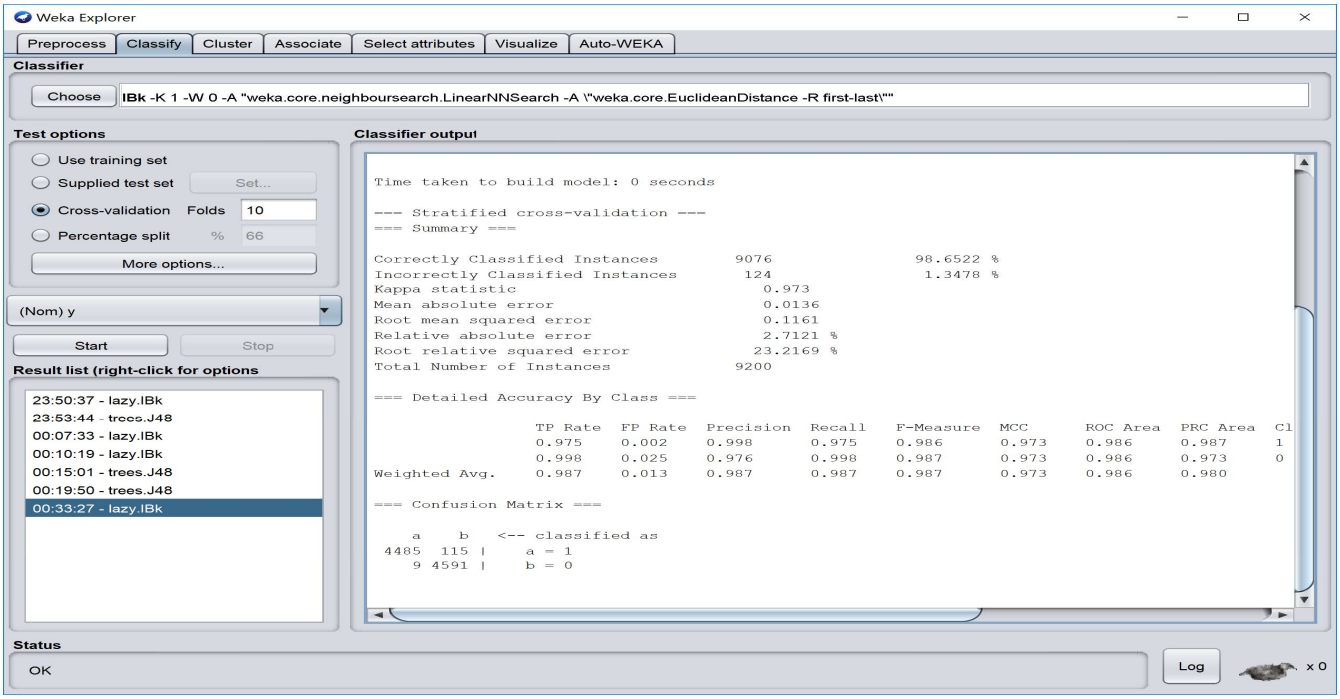

Figure 53 - kNN algorithm on oversampled dataset

For oversampling, k-NN with a k value of 1 as well, and the result also showed a high increase in the correctly classified instances with a percentage of 98.6522, Precision of 0.987 Recall value of 0.987, AUC of 0.986 and a True Positive rate of 0.987.

The Decision Tree also showed an increase in the correctly classified instances with a percentage of 96.8587, Precision of 0.969, Recall value of 0.969, AUC of 0.977 and a True Positive rate of 0.969.

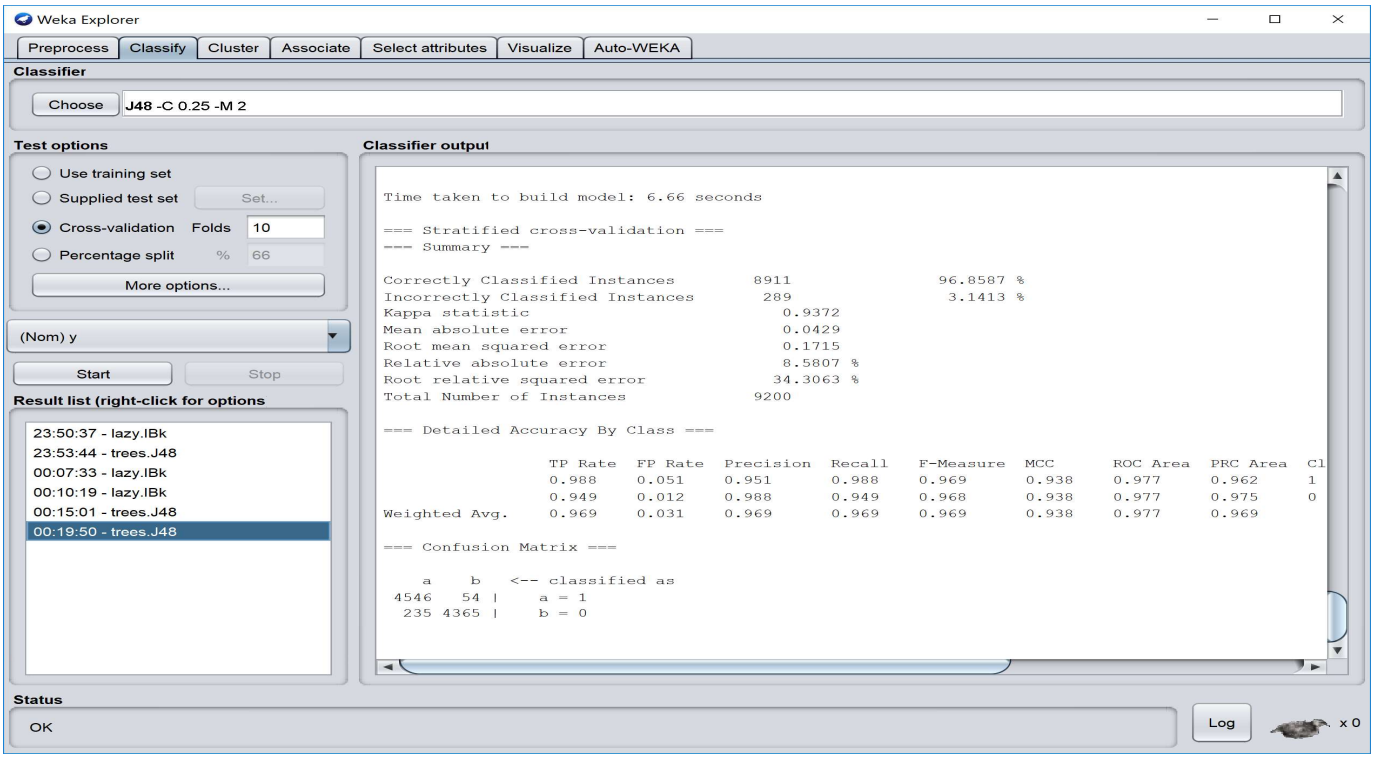

Figure 54 - Decision tree on Oversampled dataset

2.4.3 PERFORMING SMOTE ON DATASET

SMOTE method is the process of generating new data in the minority class (class 1) by interpolating between the training instances in the class. To achieve this, the inbuilt Filter function SMOTE in Weka was used. The parameters were set as nearestNeighbours = 1 and percentage = 300. The resulting dataset has a total instance of 14705 with class 1 now having 7365 instances and class 0 having 7365 instances.

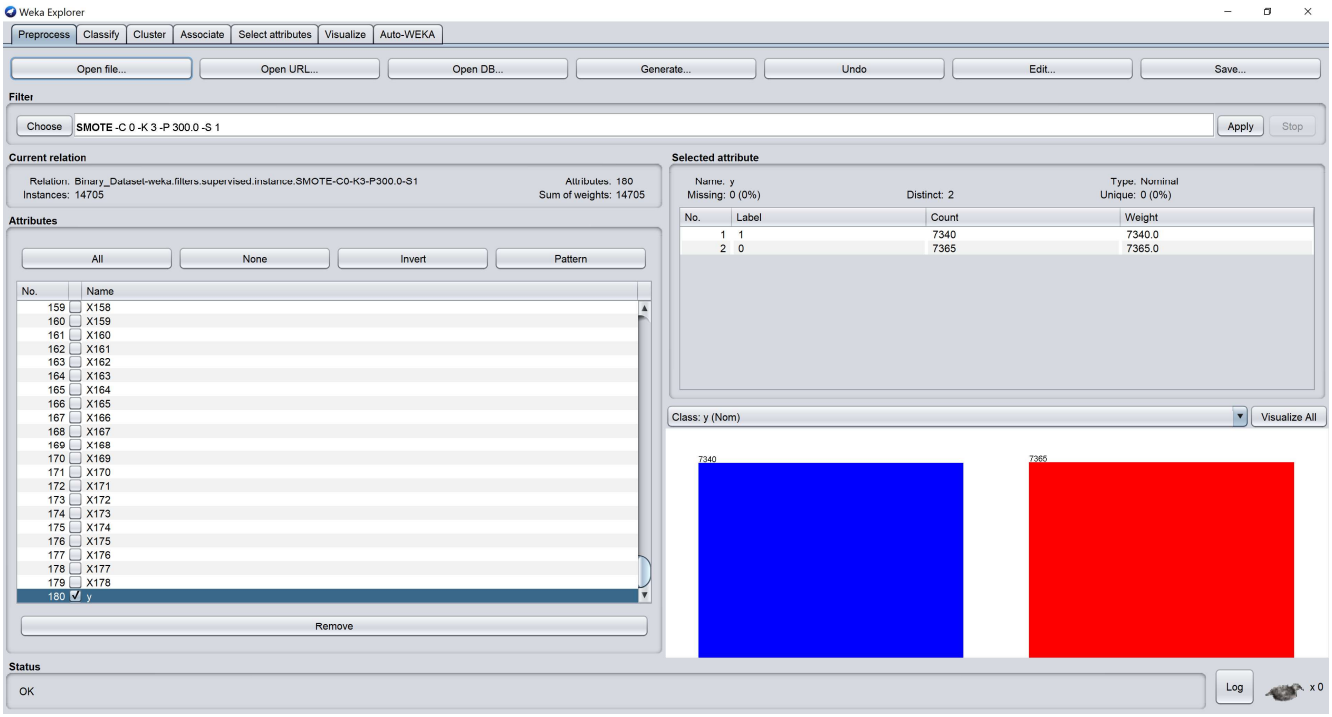

Figure 55 - SMOTE Dataset

Lastly, Decision Tree and kNN algorithms with a value of 1 and cross validation (fold = 10) were used to train the SMOTE dataset

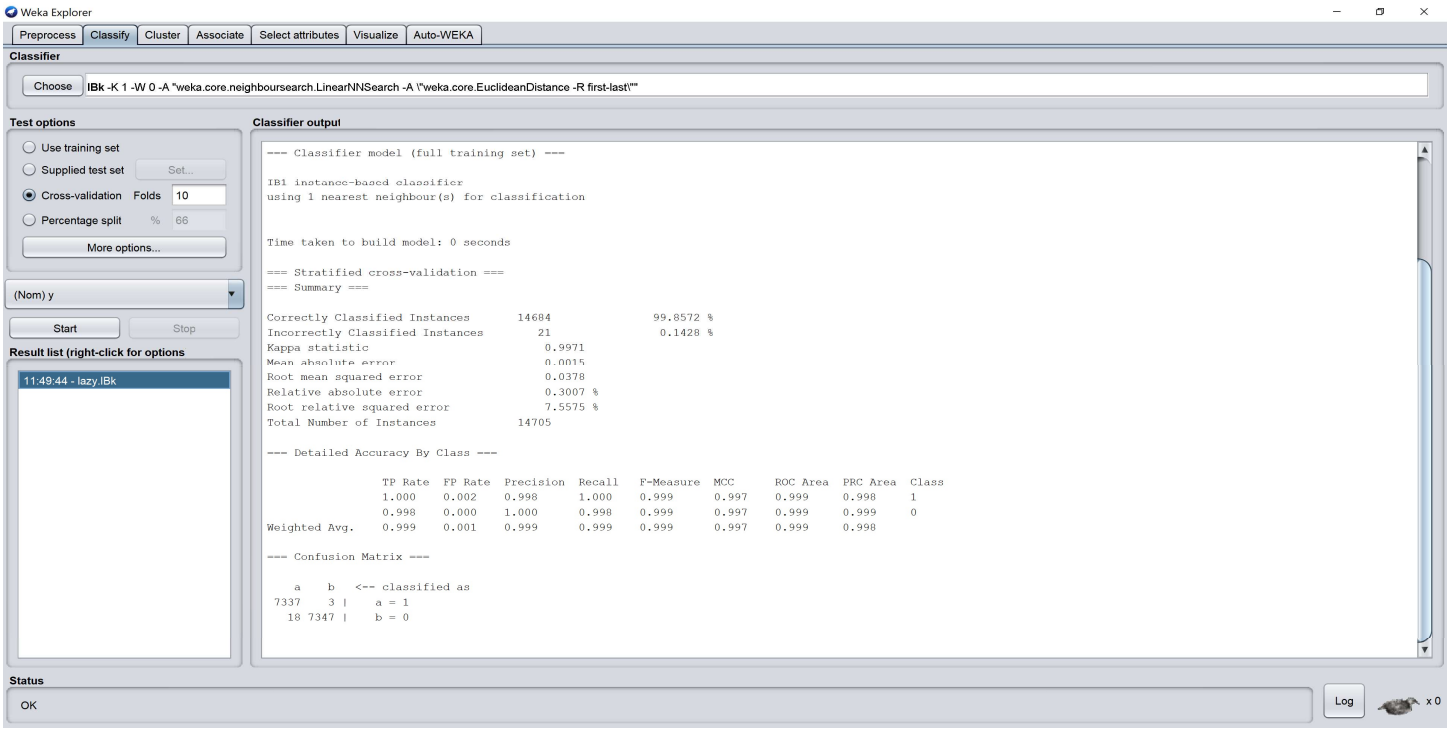

Figure 56 - kNN algorithm on SMOTE dataset

For the SMOTE dataset, using k = 1 for k-NN algorithm, it also showed a significant increase in the correctly classified instances with a percentage of 99.8572, Precision of 0.999 Recall value of 0.999, AUC of 0.999 and a True Positive rate of 0.999. The Decision Tree also showed an increase in the correctly classified instances with a percentage of 93.22, Precision of 0.933, Recall value of 0.932, AUC of 0.946 and a True Positive rate of 0.932.

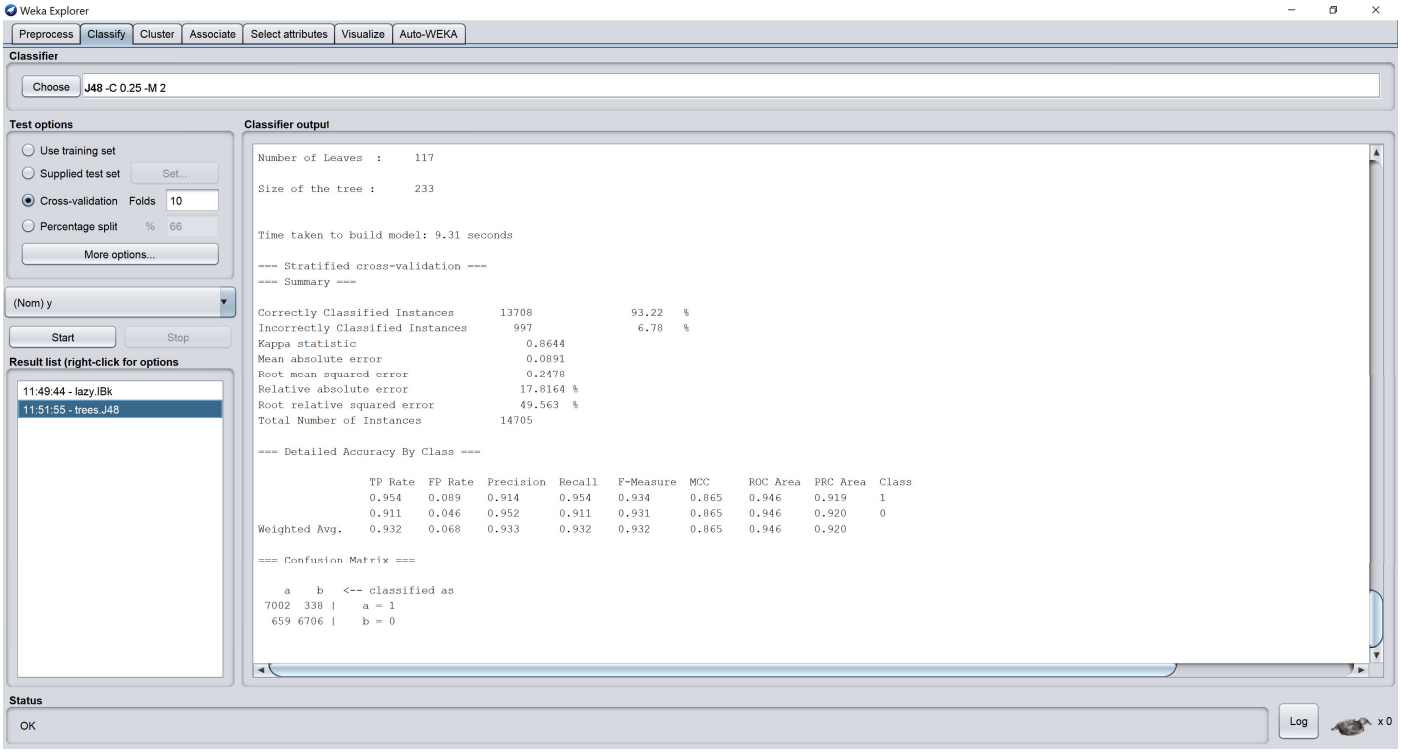

Figure 57 - Decision tree on SMOTE dataset

TABULAR PRESENTATION OF COMPARISON

Comparison Factors	k-NN Algorithm (k = 1)			Decision Tree Algorithm		
	Undersampling	**Oversampling**	**Smote**	**Undersampling**	**Oversampling**	**Smote**
Correctly Classified instances (%)	89.6185	98.6522	99.8572	91.4714	96.8587	93.22
Precision	0.911	0.987	0.999	0.915	0.969	0.933
Recall	0.896	0.987	0.999	0.915	0.969	0.932
AUC	0.896	0.986	0.999	0.903	0.977	0.946
True Positive	0.896	0.987	0.999	0.915	0.969	0.932

Overall, the performance of our model built using kNN algorithm on the SMOTE dataset gave the best result, although the instances in the dataset increased as we has to generate more class 1 instances to balance out with class 0, the correctly classified instance score was 99.8572 and Precision, Recall, AUC and true positive of 0.999. *(See Appendix 5 for SMOTE Train dataset)*

VALIDATING WITH TEST DATASET (UNSEEN)

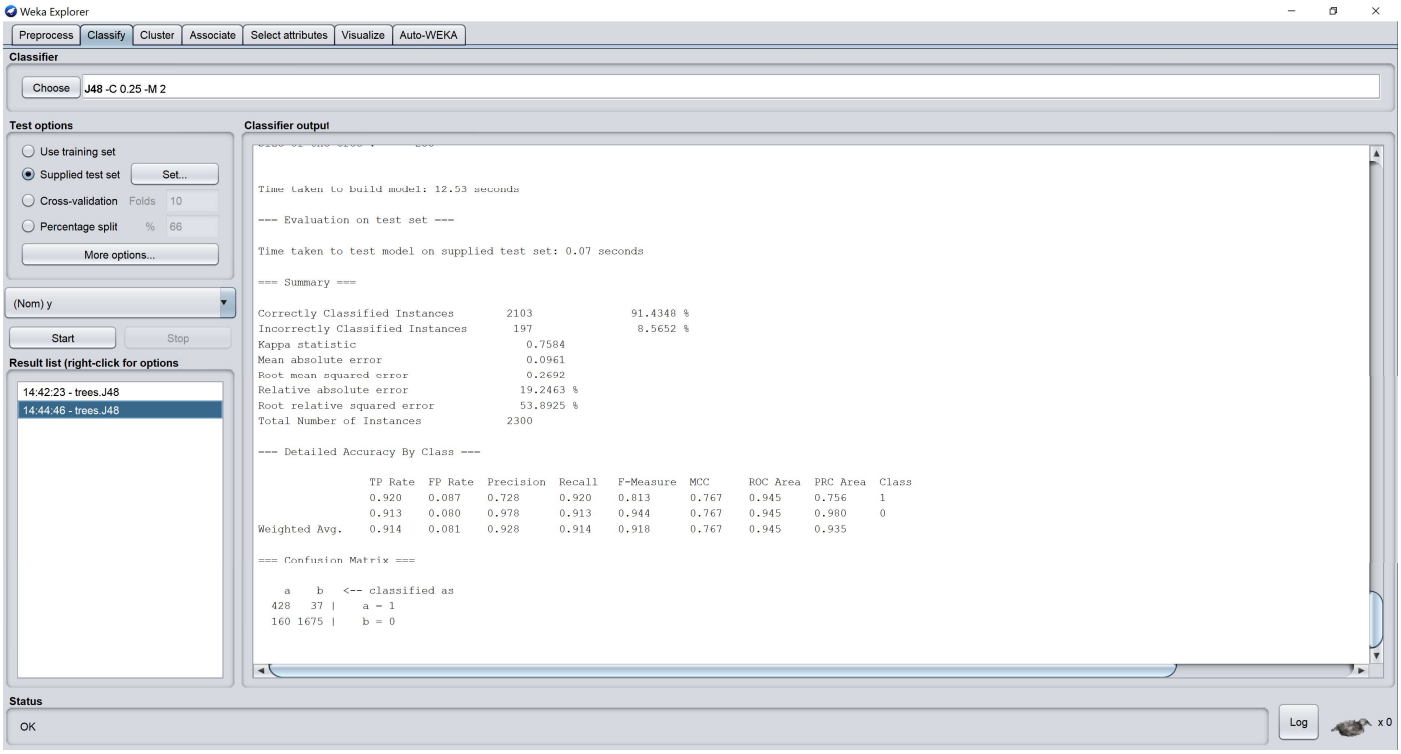

Figure 58 - Decision Tree (SMOTE)on Test Dataset

The trained model using SMOTE was used on the Train Dataset and it gave the following result; kNN had a correctly classified instances of 95.087%, Precision = 0.953, Recall = 0.951, AUC = 0.880 and True Positive = 0.951 while Decision Tree had a correctly classified instances of 91.4348%, Precision = 0.928, Recall = 0.914, AUC = 0.954 and True Positive = 0.914. When compared with model built on original Train dataset and SMOTE kNN built model, the SMOTE Decision Tree model had the lowest percentage of correctly classified instance but performed better in AUC.

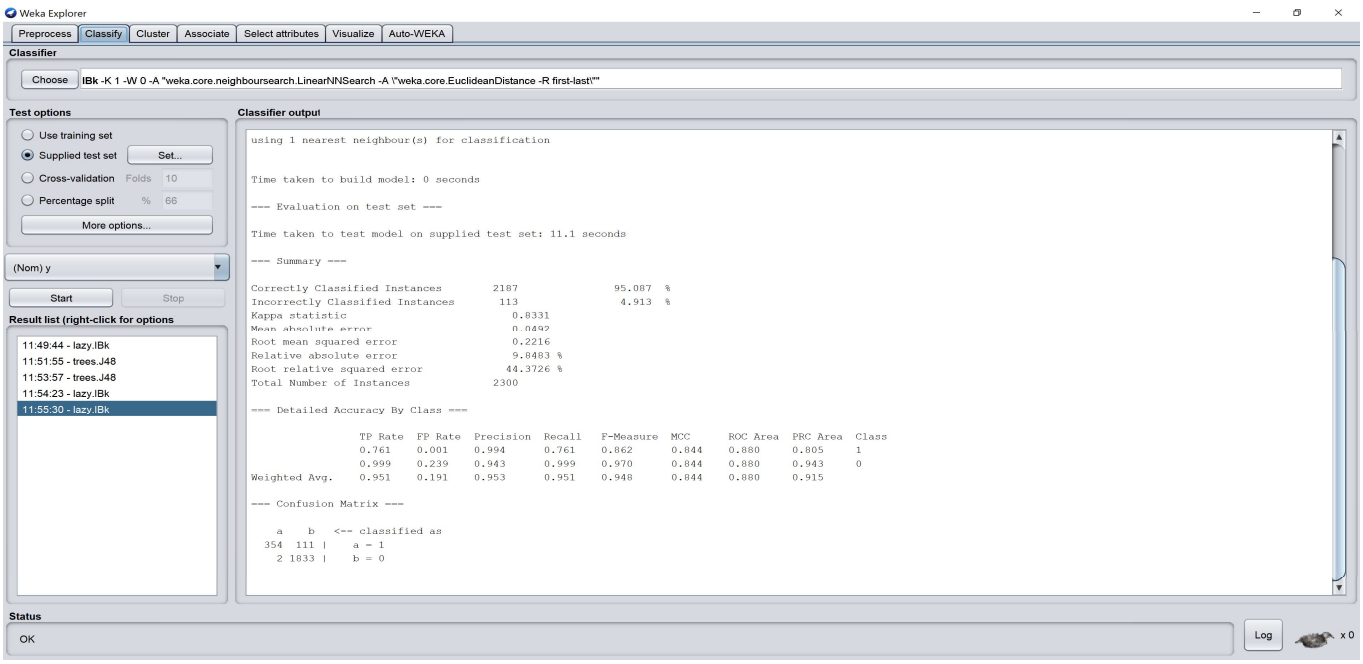

Figure 59 - kNN (SMOTE)on Test Dataset

Comparison Factors	Test Dataset (Unseen Data)			
	kNN (SMOTE)	Decision Tree (SMOTE)	k-NN	Decision Tree
Correctly Classified instances (%)	95.087	91.4348	94.3478	94.3913
Precision	0.953	0.928	0.947	0.943
Recall	0.951	0.914	0.943	0.944
AUC	0.880	0.945	0.862	0.873
True Positive	0.951	0.914	0.943	0.944

SIGNIFICANT TEST - CONFIDENCE INTERVAL

Significant test was carried out using the percentage of correctly classified instances' value of the all trained models; k-NN, Decision Tree, k-NN (SMOTE) and Decision Tree (SMOTE) on the Test Dataset (Unseen) using the formula below:

$$error \pm Z \cdot \sqrt{\left(\frac{error \cdot (1 - error)}{n}\right)}$$

n = number of instances

Z = 1.96 (95% confidence)

	k-NN	Decision Tree	k-NN (SMOTE)	Decision Tree (SMOTE)
n	2300	2300	2300	2300
Z	1.98	1.98	1.98	1.98
AUC	0.862	0.873	0.880	0.945
+ or -	0.0140956571 approx. (0.014)	0.0136082167 approx. (0.014)	0.0132807962 approx (0.13)	0.0093172855 approx 0.009
+ value	0.876	0.887	0.893	0.954
- value	0.848	0.859	0.867	0.936

The test showed that the AUC for Decision Tree model built using the SMOTE Train Dataset has a margin error of approximately +or- 0.009, which means it can fall between 0.936 to 0.954. When compared to the others, it performed better.

APPLICATION OF PRINCIPAL COMPONENT ALGORITHM TO TRAIN DATASET

Principal Component is an unsupervised pre-processing algorithm that considers the vectors of a dataset and projects them into a new dimensional space that is composed of the original linear combination of the dataset. *(See Appendix 6, 7 for Principal Component Train dataset and Weka PC Correlation Matrix)*

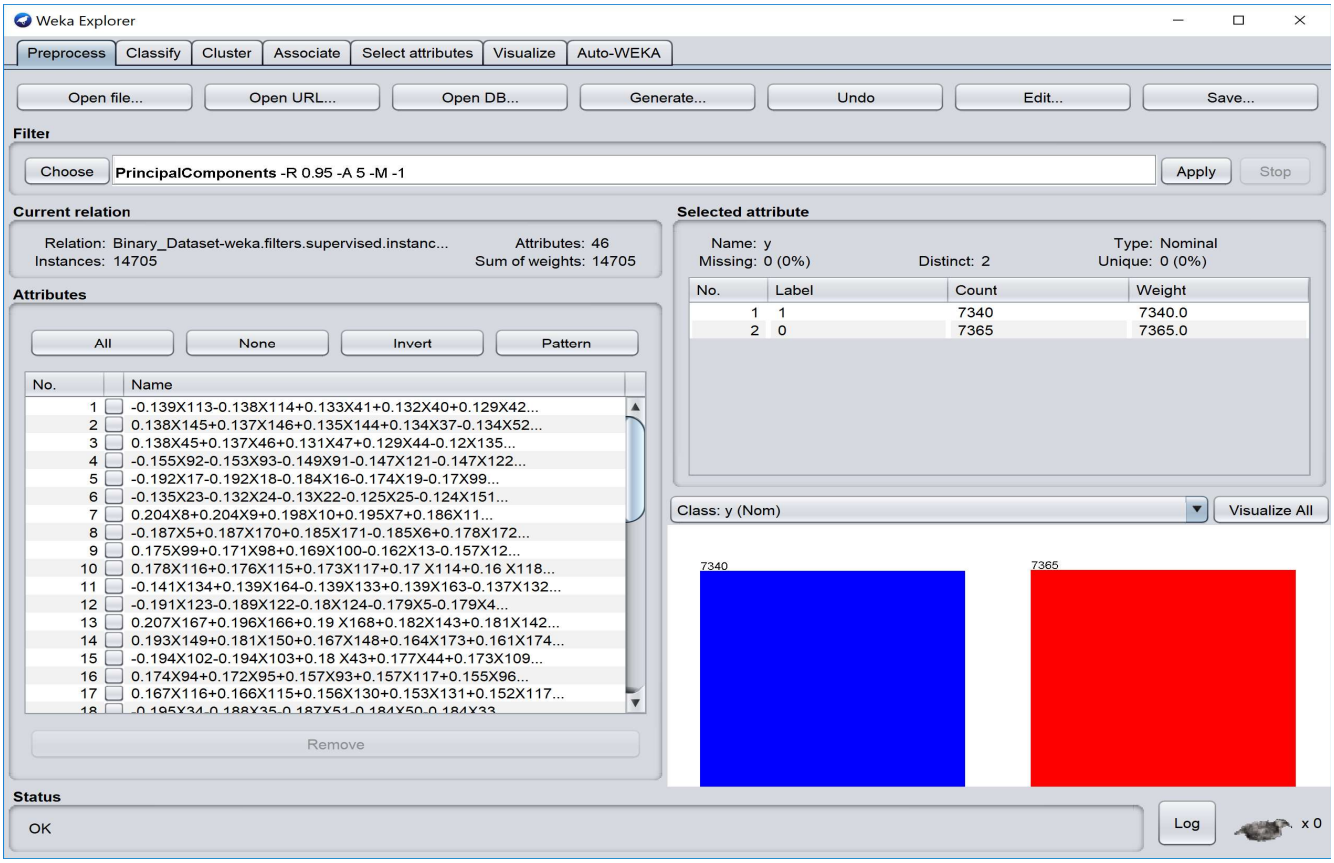

Figure 60 - Dataset after applying Principal Component Analysis

To apply Principal Component Analysis (PC) algorithm on our dataset for analysis, we use the in-built function (PrincipalComponent) with a variant cover of 95% (0.95). Before applying PC, we remove the first column label 'Unknow'; the multi-valued nominal attributes, which are identifiers for Patients. The time taken to run PC was very quick. The resulting dataset still had 14705 instances, but the attributes reduced from 179 to 46.

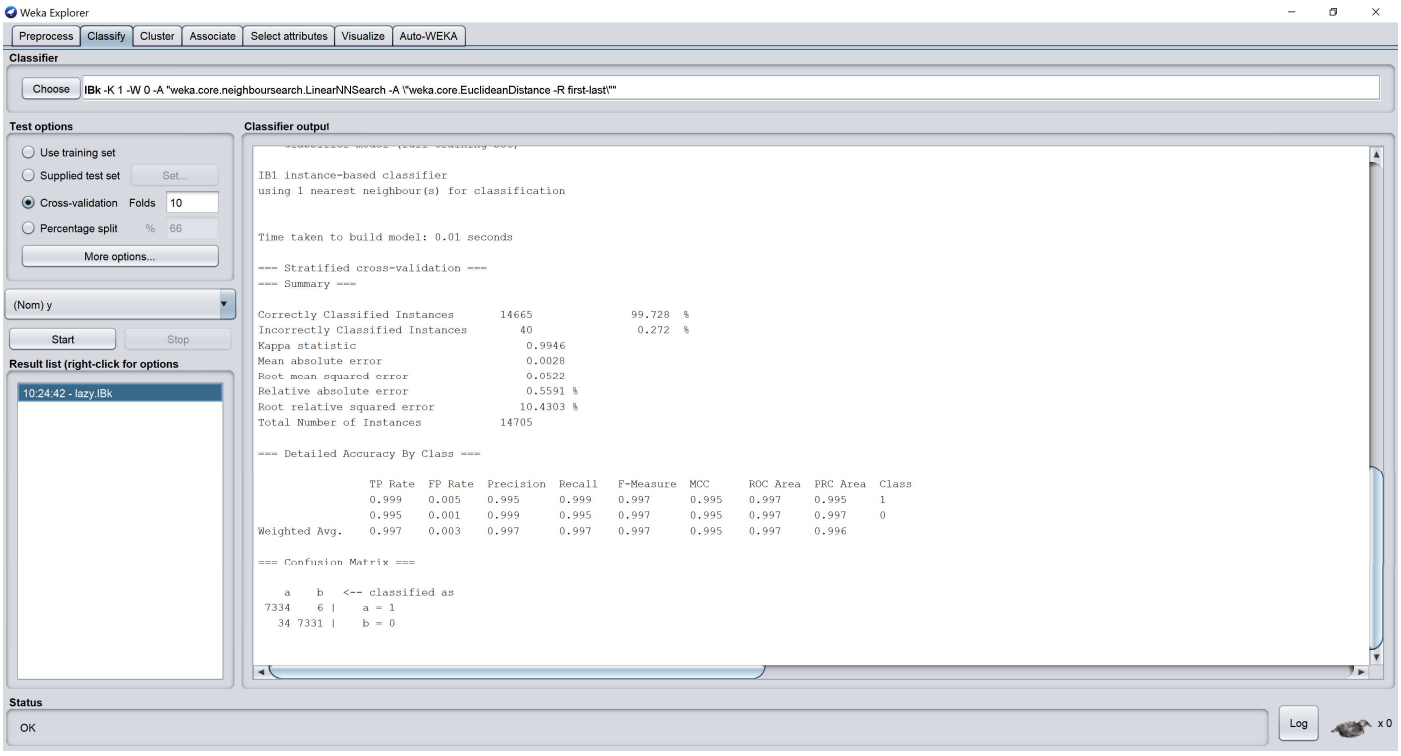

Figure 61 - kNN on Principal Component Dataset

The new dataset was trained using k-NN algorithm and despite the reduction in dataset size using PC, the result was good compared to the original dataset as shown in the table below:

Comparison Factors	Train Dataset	
	k-NN	**k-NN (Principal Component)**
Correctly Classified instances (%)	99.8572	99.728
Precision	0.999	0.997
Recall	0.999	0.997
AUC	0.999	0.997
True Positive	0.999	0.997

SIGNIFICANT TEST - CONFIDENCE INTERVAL

Significant test using the Area under Curve (AUC) value of k-NN algorithm before and after applying Principal Component algorithm on the Train Dataset was carried out, using the formula below:

$$error \pm Z \cdot \sqrt{\left(\frac{error \cdot (1 - error)}{n}\right)}$$

n = number of instances

Z = 1.96 (95% confidence)

	k-NN	**k-NN (Principal Component)**
n	14705	14705
Z	1.98	1.98
AUC	0.999	0.997
+ or -	0.0005108652	0.0008839583
+ value	0.9995108652	0.9978839583
- value	0.9984891348	0.9961160471

After comparing both scenarios, we did not find a noticeable drop in performance, as the difference in AUC is just 0.002 (0.2%). This shows that PC is an effective method of reducing your dataset and still have a good performance with an increase in processing time.

APPLYING FEATURE SELECTION ALGORITHM ON SMOTE TRAIN DATASET

Feature Selection is the method of reducing the number of input attributes in a given dataset, this is done mainly to reduce the computational cost of modelling and sometimes to improve performance of the model.

Weka's Select attributes function was used to perform this method. The algorithm used is CfsSubsetEval which evaluates the worth of a subset of attributes by considering the individual predictive ability of each feature along with the degree of redundancy between them.

The resulting dataset has 126 attributes, a reduction from 180 attributes in the previous dataset. *(See Appendix 8 and 9 for Feature Selection Train and Test dataset)*

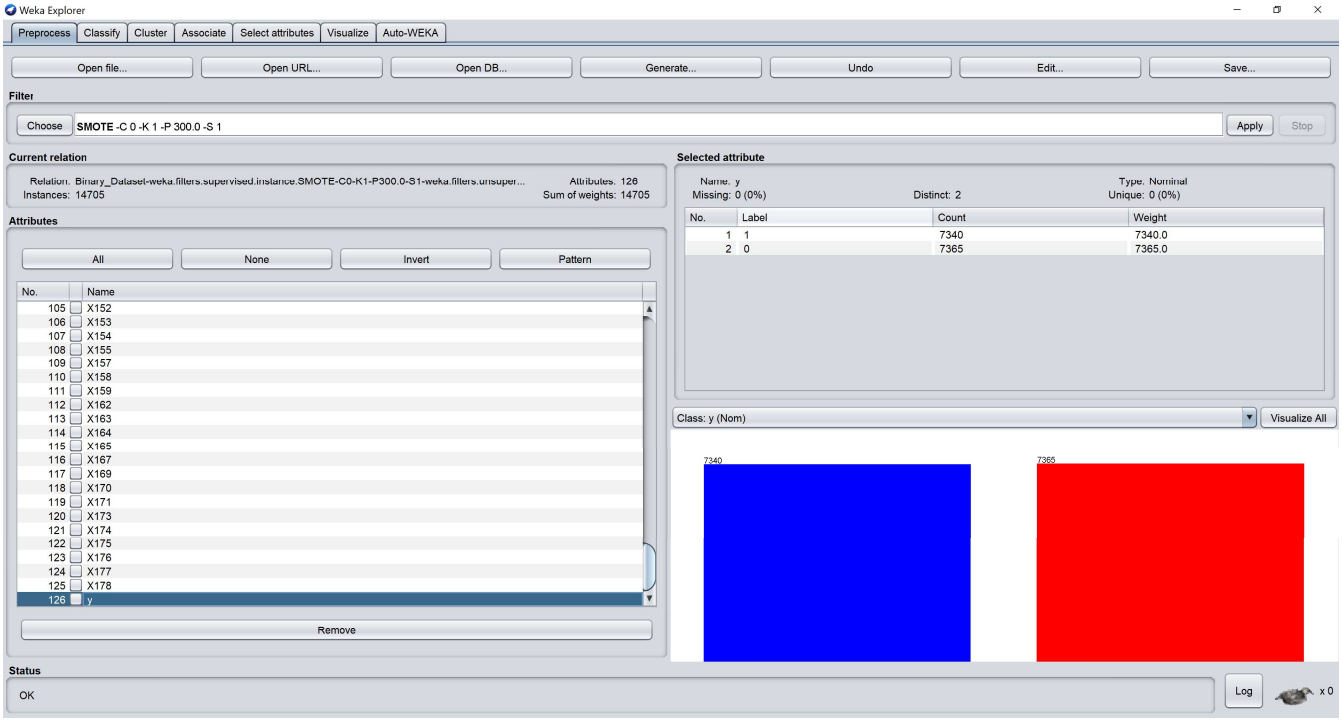

Figure 62 - Dataset after applying Feature selection Technique

kNN ALGORITHM TO TRAIN NEW FEATURE SELECTION DATASET

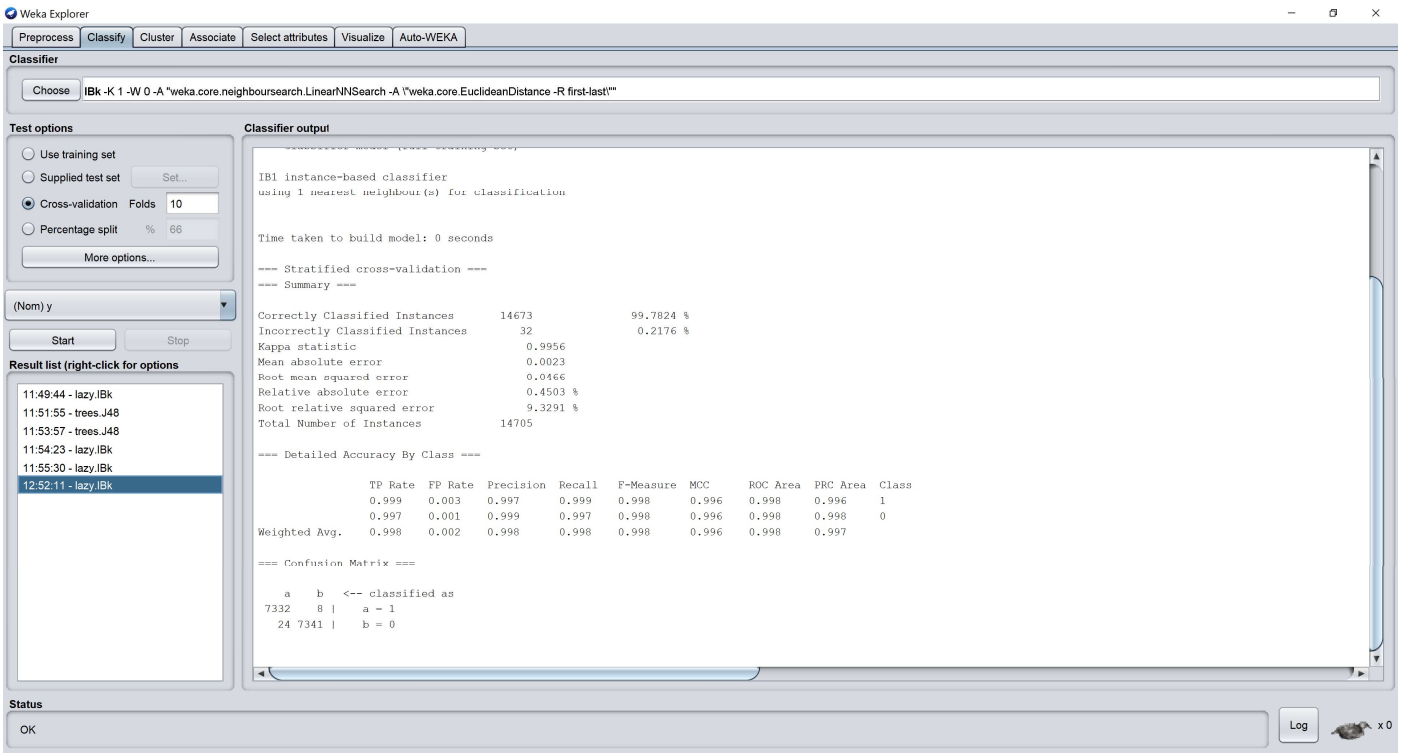

Figure 63 - kNN on Feature Selection Dataset

Despite the reduction in features, the result was still very good as the result showed that the correctly classified instances = 99.7824%, Precision = 0.998, Recall = 0.998, AUC = 0.998 and TP = 0.998. Although, there is a slight drop in the AUC of 0.001 when compared the full train dataset, given that 54 attributes were dropped during feature selection process, we can say the model performed relatively well. The model also took shorter time to train

TESTING MODEL ON UNSEEN DATASET (TEST)

The Test Dataset was prepared to fit the attributes of the feature selection dataset. This process was performed by manually removing the attributes that are less useful from the Test dataset in Weka.

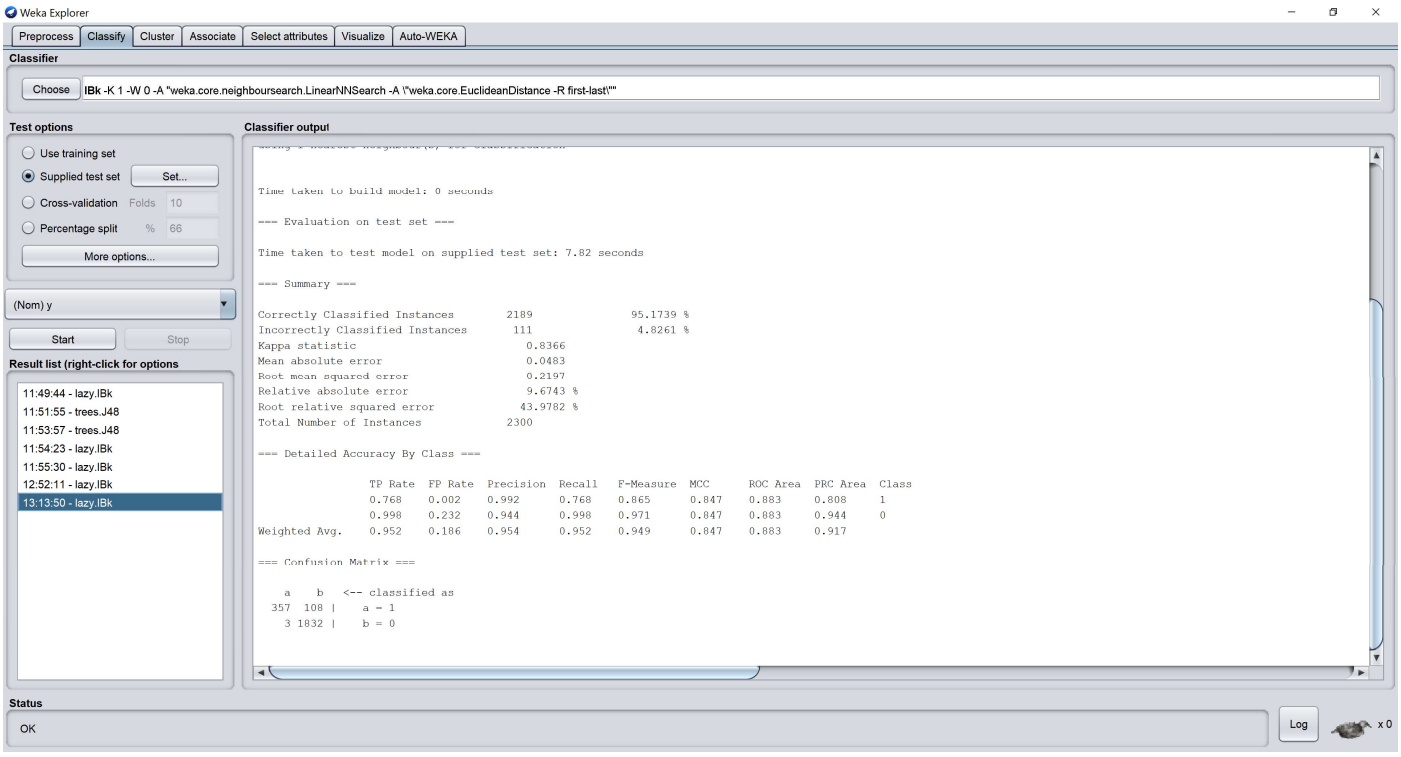

Figure 64 - Validation of Feature Selection Built model on Test Dataset

The test showed that after removing attributes that are believed to be less useful, the model performed better on the Test dataset with 95.1739% of correctly classified instances and an AUC of 0.883. The time taken to build the model was also shorter. Therefore, Feature Selection Technique is a useful method to use.

Comparison Factors	Train Dataset		Test Dataset (Unseen Data)	
	k-NN	k-NN (Feature Selection Dataset)	k-NN	k-NN (Feature Selection Dataset)
Correctly Classified instances (%)	99.8572	99.7824	95.087	95.1739
Precision	0.999	0.998	0.953	0.954
Recall	0.999	0.998	0.951	0.952
AUC	0.999	0.998	0.880	0.883
True Positive	0.999	0.998	0.951	0.952

SIGNIFICANT TEST - CONFIDENCE INTERVAL

Significant test was carried out using the Area under Curve (AUC) value of k-NN algorithm before and after applying feature selection method on the Test Dataset (Unseen) using the formula below:

$$error \pm Z \cdot \sqrt{\left(\frac{error \cdot (1 - error)}{n}\right)}$$

n = number of instances

Z = 1.96 (95% confidence)

	k-NN	**k-NN (Feature Selection Dataset)**
n	2300	2300
Z	1.98	1.98
AUC	0.880	0.883
+ or -	0.0140956571 approx. (0.014)	0.0136082167 approx. (0.014)
+ value	0.876	0.887
- value	0.848	0.859

After comparing both scenario, a drop in performance was not noticeable, as the difference in AUC is just 0.003 (0.2%). The margin of error for kNN is between 0.844 to 0.876, while kNN(Feature Selection Dataset) is 0.859 to 0.887. There is an overlap for both and there isn't a significant difference. This shows that Feature Selection technique is an effective method of reducing your dataset and still have a good performance with an increase in processing time.

CONCLUSION

There was a challenge of an imbalance dataset, after converting the original dataset which had 5 classes {1,2,3,4,5} to a binary class {1,0} dataset. This affected the trained model as it could better identify class 0 than class 1, which could be ascribed to the fact that class 0 has 4 times the instance of class 1.

With the application of SMOTE technique on the dataset, there was noticeable better performance of our trained model for the two (2) algorithm used.

It was also evident from the result of the significant test done after applying Principal Component Analysis and Feature Selection techniques that both methods are very effective in reducing a given dataset without losing performance of the built model and sometimes get a better performance, like the case of Feature Selection. Both methods also help us save time as we had less dataset to process.

APPENDIX

Appendix 1	Python Code
Appendix 2	ERS dataset
Appendix 3	Train dataset
Appendix 4	Test dataset
Appendix 5	SMOTE Train dataset
Appendix 6	Principal Component Train dataset
Appendix 7	PC Correlation Matrix
Appendix 8	Feature Selection Train dataset
Appendix 9	Feature Selection Test dataset

Note: Please, due to the size of the Appendix files, they are all available and can be accessed from the this google drive link: https://drive.google.com/drive/folders/1UmeHb7n3XRVDnByLYgt-IxsN4dGkLSSy?usp=sharing

VIDEO URL

WILSON SUNDAY OTITONAIYE-
https://brightspace.bournemouth.ac.uk/d2l/le/content/68762/viewContent/942817/View
ABHAY SUJALA KRISHNAN-
https://bournemouth.cloud.panopto.eu/Panopto/Pages/Viewer.aspx?id=3994fa5a-67ca-4ba8-ae9d-abce01214d6f
NIKHIL CHANDRANNOLA-
https://bournemouth.cloud.panopto.eu/Panopto/Pages/Viewer.aspx?id=66647370-4722-461b-a01a-abcf008a07b3
LINGAM ANIL KUMAR- https://pro.panopto.com/Panopto/Pages/Viewer.aspx?tid=545485de-63b5-4e65-bf39-abd000003890