**Computer Science 204 - Assignment 1: Creating & Manipulating a Database**

A.)   
 1. CREATE TABLE Client (

2. ClientID INT AUTO\_INCREMENT PRIMARY KEY,

3. ClientFirstName VARCHAR(30),

4. ClientLastName VARCHAR(30),

5. ClientDOB INT(4),

6. Occupation VARCHAR(50)

7. );

8.

9. CREATE TABLE Author (

10. AuthorID INT AUTO\_INCREMENT PRIMARY KEY,

11. AuthorFirstName VARCHAR(30),

12. AuthorLastName VARCHAR(30),

13. AuthorNationality VARCHAR(40)

14. );

15.

16. CREATE TABLE Book (

17. BookID INT AUTO\_INCREMENT PRIMARY KEY,

18. BookTitle VARCHAR(100),

19. AuthorID INT,

20. Genre VARCHAR(40),

21. FOREIGN KEY (AuthorID) REFERENCES Author(AuthorID)

22. );

23.

24. CREATE TABLE Borrower (

25. BorrowID INT AUTO\_INCREMENT PRIMARY KEY,

26. ClientID INT,

27. BookID INT,

28. BorrowDate DATE,

29. FOREIGN KEY (ClientID) REFERENCES Client(ClientID),

30. FOREIGN KEY (BookID) REFERENCES Book(BookID)

31. );

32.

B.)

1. INSERT INTO Author (AuthorFirstName, AuthorLastName, AuthorNationality)

2. VALUES

3. ('Sofia', 'Smith', 'Canada'),

4. ('Maria', 'Brown', 'Brazil'),

5. ('Elena', 'Martin', 'Mexico'),

6. ('Zoe', 'Roy', 'France'),

7. ('Sebastian','Lavoie', 'Canada'),

8. ('Dylan', 'Garcia', 'Spain'),

9. ('Ian', 'Cruz', 'Mexico'),

10. ('Lucas', 'Smith', 'USA'),

11. ('Fabian', 'Wilson', 'USA'),

12. ('Liam', 'Taylor', 'Canada'),

13. ('William', 'Thomas', 'Great Britain'),

14. ('Logan', 'Moore', 'Canada'),

15. ('Oliver', 'Martin', 'France'),

16. ('Alysha', 'Thompson', 'Canada'),

17. ('Isabelle', 'Lee', 'Canada'),

18. ('Emily', 'Clark', 'USA'),

19. ('John', 'Young', 'China'),

20. ('David', 'Wright', 'Canada'),

21. ('Thomas', 'Scott', 'Canada'),

22. ('Helena', 'Adams', 'Canada'),

23. ('Sofia', 'Carter', 'USA'),

24. ('Liam', 'Parker', 'Canada'),

25. ('Emily', 'Murphy', 'USA');

 Verified table with SELECT \* FROM Author

1. INSERT INTO Book (BookTitle, AuthorID, Genre)

2. VALUES

3. ('Build your database system', 1, 'Science'),

4. ('The red wall', 2, 'Fiction'),

5. ('The perfect match', 3, 'Fiction'),

6. ('Digital Logic', 4, 'Science'),

7. ('How to be a great lawyer', 5, 'Law'),

8. ('Manage successful negotiations',6, 'Society'),

9. ('Pollution today', 7, 'Science'),

10. ('A gray park', 2, 'Fiction'),

11. ('How to be rich in one year', 8, 'Humor'),

12. ('Their bright fate', 9, 'Fiction'),

13. ('Black lines', 10, 'Fiction'),

14. ('History of theater', 11, 'Literature'),

15. ('Electrical transformers', 12, 'Science'),

16. ('Build your big data system', 1, 'Science'),

17. ('Right and left', 13, 'Children'),

18. ('Programming using Python', 1, 'Science'),

19. ('Computer networks', 14, 'Science'),

20. ('Performance evaluation', 15, 'Science'),

21. ('Daily exercise', 16, 'Well being'),

22. ('The silver uniform', 17, 'Fiction'),

23. ('Industrial revolution', 18, 'History'),

24. ('Green nature', 19, 'Well being'),

25. ('Perfect football', 20, 'Well being'),

26. ('The chocolate love', 21, 'Humor'),

27. ('Director and leader', 22, 'Society'),

28. ('Play football every week', 20, 'Well being'),

29. ('Maya the bee', 13, 'Children'),

30. ('Perfect rugby', 20, 'Well being'),

31. ('The end', 23, 'Fiction'),

32. ('Computer security', 1, 'Science'),

33. ('Participate', 22, 'Society'),

34. ('Positive figures', 3, 'Fiction');

Verified with SELECT \* FROM Book

1. INSERT INTO Client (ClientFirstName, ClientLastName, ClientDOB, Occupation)

2. VALUES

3. ('Kaiden', 'Hill', 2006, 'Student'),

4. ('Alina', 'Morton', 2010, 'Student'),

5. ('Fania', 'Brooks', 1983, 'Food Scientist'),

6. ('Courtney', 'Jensen', 2006, 'Student'),

7. ('Brittany', 'Hill', 1983, 'Firefighter'),

8. ('Max', 'Rogers', 2005, 'Student'),

9. ('Margaret', 'McCarthy', 1981, 'School Psychologist'),

10. ('Julie', 'McCarthy', 1973, 'Professor'),

11. ('Ken', 'McCarthy', 1974, 'Securities Clerk'),

12. ('Britany', "O'Quinn", 1984, 'Violinist'),

13. ('Conner', 'Gardner', 1998, 'Licensed Massage Therapist'),

14. ('Mya', 'Austin', 1960, 'Parquet Floor Layer'),

15. ('Thierry', 'Rogers', 2004, 'Student'),

16. ('Eloise', 'Rogers', 1984, 'Computer Security Manager'),

17. ('Gerard', 'Jackson', 1979, 'Oil Exploration Engineer'),

18. ('Randy', 'Day', 1986, 'Aircraft Electrician'),

19. ('Jodie', 'Page', 1990, 'Manufacturing Director'),

20. ('Coral', 'Rice', 1996, 'Window Washer'),

21. ('Ayman', 'Austin', 2002, 'Student'),

22. ('Jaxson', 'Austin', 1999, 'Repair Worker'),

23. ('Joel', 'Austin', 1973, 'Police Officer'),

24. ('Alina', 'Austin', 2010, 'Student'),

25. ('Elin', 'Austin', 1962, 'Payroll Clerk'),

26. ('Ophelia', 'Wolf', 2004, 'Student'),

27. ('Eliot', 'McGuire', 1967, 'Dentist'),

28. ('Peter', 'McKinney', 1968, 'Professor'),

29. ('Annabella', 'Henry', 1974, 'Nurse'),

30. ('Anastasia', 'Baker', 2001, 'Student'),

31. ('Tyler', 'Baker', 1984, 'Police Officer'),

32. ('Lilian', 'Ross', 1983, 'Insurance Agent'),

33. ('Thierry', 'Arnold', 1975, 'Bus Driver'),

34. ('Angelina', 'Rowe', 1979, 'Firefighter'),

35. ('Marcia', 'Rowe', 1974, 'Health Educator'),

36. ('Martin', 'Rowe', 1976, 'Ship Engineer'),

37. ('Adeline', 'Rowe', 2005, 'Student'),

38. ('Colette', 'Rowe', 1963, 'Professor'),

39. ('Diane', 'Clark', 1975, 'Payroll Clerk'),

40. ('Caroline', 'Clark', 1960, 'Dentist'),

41. ('Dalton', 'Clayton', 1982, 'Police Officer'),

42. ('Steve', 'Clayton', 1990, 'Bus Driver'),

43. ('Melanie', 'Clayton', 1987, 'Computer Engineer'),

44. ('Alana', 'Wilson', 2007, 'Student'),

45. ('Carson', 'Byrne', 1995, 'Food Scientist'),

46. ('Conrad', 'Byrne', 2007, 'Student'),

47. ('Ryan', 'Porter', 2008, 'Student'),

48. ('Elin', 'Porter', 1978, 'Computer Programmer'),

49. ('Tyler', 'Harvey', 2007, 'Student'),

50. ('Arya', 'Harvey', 2008, 'Student'),

51. ('Serena', 'Harvey', 1978, 'School Teacher'),

52. ('Lilly', 'Franklin', 1976, 'Doctor'),

53. ('Mai', 'Franklin', 1994, 'Dentist'),

54. ('John', 'Franklin', 1999, 'Firefighter'),

55. ('Judy', 'Franklin', 1995, 'Firefighter'),

56. ('Katy', 'Lloyd', 1992, 'School Teacher'),

57. ('Tamara', 'Allen', 1963, 'Ship Engineer'),

58. ('Maxim', 'Lyons', 1985, 'Police Officer'),

59. ('Allan', 'Lyons', 1983, 'Computer Engineer'),

60. ('Marc', 'Harris', 1980, 'School Teacher'),

61. ('Elin', 'Young', 2009, 'Student'),

62. ('Diana', 'Young', 2008, 'Student'),

63. ('Diane', 'Young', 2006, 'Student'),

64. ('Alana', 'Bird', 2003, 'Student'),

65. ('Anna', 'Becker', 1979, 'Security Agent'),

66. ('Katie', 'Grant', 1977, 'Manager'),

67. ('Joan', 'Grant', 2010, 'Student'),

68. ('Bryan', 'Bell', 2001, 'Student'),

69. ('Belle', 'Miller', 1970, 'Professor'),

70. ('Peggy', 'Stevens', 1990, 'Bus Driver'),

71. ('Steve', 'Williamson', 1975, 'HR Clerk'),

72. ('Tyler', 'Williamson', 1999, 'Doctor'),

73. ('Izabelle', 'Williamson', 1990, 'Systems Analyst'),

74. ('Annabel', 'Williamson', 1960, 'Cashier'),

75. ('Mohamed', 'Waters', 1966, 'Insurance Agent'),

76. ('Marion', 'Newman', 1970, 'Computer Programmer'),

77. ('Ada', 'Williams', 1986, 'Computer Programmer'),

78. ('Sean', 'Scott', 1983, 'Bus Driver'),

79. ('Farrah', 'Scott', 1974, 'Ship Engineer'),

80. ('Christine', 'Lambert', 1973, 'School Teacher'),

81. ('Alysha', 'Lambert', 2007, 'Student'),

82. ('Maia', 'Grant', 1984, 'School Teacher');

83. Verified with SELECT \* FROM Client

1. INSERT INTO Borrower (ClientID, BookID, BorrowDate)

2. VALUES

3. (35, 17, '2016-07-20'),

4. (1, 3, '2017-04-19'),

5. (42, 8, '2016-10-03'),

6. (62, 16, '2016-04-05'),

7. (53, 13, '2017-01-17'),

8. (33, 15, '2015-11-26'),

9. (40, 14, '2015-01-21'),

10. (64, 2, '2017-09-10'),

11. (56, 30, '2017-08-02'),

12. (23, 2, '2018-06-28'),

13. (46, 19, '2015-11-18'),

14. (61, 20, '2015-11-24'),

15. (58, 7, '2017-06-17'),

16. (46, 16, '2017-02-12'),

17. (80, 21, '2018-03-18'),

18. (51, 23, '2015-09-01'),

19. (49, 18, '2015-07-28'),

20. (43, 18, '2015-11-04'),

21. (30, 2, '2018-08-10'),

22. (48, 24, '2015-05-13'),

23. (71, 5, '2016-09-05'),

24. (35, 3, '2016-07-03'),

25. (57, 1, '2015-03-17'),

26. (23, 25, '2017-08-16'),

27. (20, 12, '2018-07-24'),

28. (25, 7, '2015-01-31'),

29. (72, 29, '2016-04-10'),

30. (74, 20, '2017-07-31'),

31. (53, 14, '2016-02-20'),

32. (32, 10, '2017-07-24'),

33. (12, 15, '2018-04-25'),

34. (77, 13, '2017-06-09'),

35. (30, 4, '2017-10-24'),

36. (37, 24, '2016-01-14'),

37. (27, 26, '2017-06-05'),

38. (1, 16, '2018-05-06'),

39. (21, 9, '2016-03-19'),

40. (69, 28, '2017-03-29'),

41. (17, 19, '2017-03-14'),

42. (8, 9, '2016-04-22'),

43. (63, 18, '2015-01-25'),

44. (65, 20, '2016-10-10'),

45. (51, 19, '2015-07-28'),

46. (23, 12, '2017-01-25'),

47. (17, 4, '2017-04-18'),

48. (68, 5, '2016-09-06'),

49. (46, 13, '2017-09-30'),

50. (15, 13, '2017-07-05'),

51. (11, 19, '2017-12-14'),

52. (78, 15, '2017-01-26'),

53. (47, 9, '2015-03-03'),

54. (68, 7, '2016-05-26'),

55. (37, 26, '2017-02-06'),

56. (48, 27, '2015-12-30'),

57. (9, 21, '2017-10-21'),

58. (29, 8, '2018-04-01'),

59. (64, 18, '2017-08-29'),

60. (61, 26, '2018-02-21'),

61. (39, 28, '2016-07-26'),

62. (73, 18, '2018-08-22'),

63. (11, 13, '2018-01-17'),

64. (45, 6, '2016-07-20'),

65. (33, 13, '2018-03-18'),

66. (10, 17, '2016-06-06'),

67. (28, 18, '2017-02-17'),

68. (51, 3, '2016-12-09'),

69. (29, 2, '2015-09-18'),

70. (28, 30, '2017-09-14'),

71. (74, 20, '2015-12-12'),

72. (15, 22, '2015-01-14'),

73. (57, 8, '2017-08-20'),

74. (2, 5, '2015-01-18'),

75. (74, 12, '2018-04-14'),

76. (51, 10, '2016-02-25'),

77. (25, 17, '2015-02-24'),

78. (45, 21, '2017-02-10'),

79. (27, 25, '2016-08-03'),

80. (32, 28, '2016-06-15'),

81. (71, 21, '2017-05-21'),

82. (75, 26, '2016-05-03'),

83. (56, 32, '2015-12-23'),

84. (26, 32, '2015-05-16'),

85. (66, 32, '2015-05-30'),

86. (57, 18, '2017-09-15'),

87. (40, 15, '2016-09-02'),

88. (65, 4, '2017-08-17'),

89. (54, 7, '2015-12-19'),

90. (29, 4, '2017-07-22'),

91. (44, 9, '2017-12-31'),

92. (56, 31, '2015-06-13'),

93. (17, 4, '2015-04-01'),

94. (35, 16, '2018-07-19'),

95. (22, 18, '2017-06-22'),

96. (39, 24, '2015-05-29'),

97. (63, 14, '2018-01-20'),

98. (53, 21, '2016-07-31'),

99. (40, 9, '2016-07-10'),

100. (52, 4, '2017-04-05'),

101. (27, 20, '2016-09-04'),

102. (72, 29, '2015-12-06'),

103. (49, 16, '2017-12-19'),

104. (6, 12, '2016-12-04'),

105. (74, 31, '2016-07-27'),

106. (48, 32, '2016-06-29'),

107. (69, 2, '2016-12-27'),

108. (60, 32, '2017-10-29'),

109. (45, 22, '2017-06-12'),

110. (42, 15, '2017-05-14'),

111. (79, 8, '2016-10-13'),

112. (70, 18, '2016-12-04'),

113. (34, 8, '2016-03-06'),

114. (43, 8, '2015-12-19'),

115. (42, 32, '2016-04-20'),

116. (67, 5, '2017-03-06'),

117. (80, 25, '2015-06-23'),

118. (54, 11, '2017-05-03'),

119. (34, 28, '2017-08-30'),

120. (65, 20, '2017-08-26'),

121. (61, 19, '2018-01-05'),

122. (38, 12, '2018-01-17'),

123. (51, 4, '2016-05-13'),

124. (7, 16, '2016-03-17'),

125. (46, 16, '2016-11-25'),

126. (75, 30, '2018-08-12'),

127. (72, 32, '2015-03-12'),

128. (44, 17, '2015-06-15'),

129. (68, 15, '2016-02-21'),

130. (21, 1, '2016-06-19'),

131. (14, 25, '2016-10-10'),

132. (68, 21, '2016-05-27'),

133. (35, 20, '2015-03-19'),

134. (16, 27, '2016-08-08'),

135. (79, 31, '2018-03-07'),

136. (14, 17, '2018-04-28'),

137. (29, 28, '2018-03-11'),

138. (41, 4, '2018-08-08'),

139. (42, 3, '2016-02-23'),

140. (45, 3, '2017-07-10'),

141. (36, 16, '2018-07-19'),

142. (36, 30, '2015-08-07'),

143. (54, 32, '2018-03-14'),

144. (61, 15, '2017-03-28'),

145. (1, 13, '2018-05-17'),

146. (43, 1, '2015-05-14'),

147. (37, 14, '2015-07-30'),

148. (62, 17, '2015-09-19'),

149. (50, 22, '2016-12-02'),

150. (45, 1, '2016-07-24'),

151. (32, 17, '2018-03-10'),

152. (13, 28, '2016-02-14'),

153. (15, 9, '2018-08-11'),

154. (10, 19, '2018-08-29'),

155. (66, 3, '2016-11-27'),

156. (68, 29, '2017-07-12'),

157. (21, 14, '2018-06-27'),

158. (35, 9, '2016-01-22'),

159. (17, 24, '2016-08-25'),

160. (40, 21, '2015-07-09'),

161. (1, 24, '2016-03-28'),

162. (70, 27, '2015-07-10'),

163. (80, 26, '2016-04-24'),

164. (29, 5, '2015-10-18'),

165. (76, 12, '2018-04-25'),

166. (22, 4, '2016-12-24'),

167. (2, 2, '2017-10-26'),

168. (35, 13, '2016-02-28'),

169. (40, 8, '2017-10-02'),

170. (68, 9, '2016-01-03'),

171. (32, 5, '2016-11-13'),

172. (34, 17, '2016-09-15'),

173. (34, 16, '2018-04-13'),

174. (80, 30, '2016-10-13'),

175. (20, 32, '2015-11-17'),

176. (36, 10, '2017-09-01'),

177. (78, 12, '2018-06-27'),

178. (57, 8, '2016-03-22'),

179. (75, 11, '2017-06-27'),

180. (71, 10, '2015-08-01'),

181. (48, 22, '2015-09-29'),

182. (19, 16, '2016-02-21'),

183. (79, 30, '2018-08-20'),

184. (70, 13, '2016-09-16'),

185. (30, 6, '2017-02-10'),

186. (45, 12, '2017-10-12'),

187. (30, 27, '2016-11-23'),

188. (26, 3, '2016-08-13'),

189. (66, 6, '2017-01-14'),

190. (47, 15, '2016-02-10'),

191. (53, 30, '2018-08-08'),

192. (80, 16, '2016-03-31'),

193. (70, 13, '2018-02-03'),

194. (14, 25, '2016-03-27'),

195. (46, 22, '2016-01-13'),

196. (30, 32, '2015-08-06'),

197. (60, 14, '2016-11-27'),

198. (14, 13, '2018-05-23'),

199. (71, 15, '2016-06-22'),

200. (38, 21, '2015-12-27'),

201. (69, 30, '2017-04-29'),

202. (49, 31, '2018-06-03'),

203. (28, 28, '2015-05-29'),

204. (49, 3, '2016-08-30'),

205. (75, 1, '2015-10-29'),

206. (78, 3, '2017-05-12'),

207. (43, 18, '2015-03-25'),

208. (27, 21, '2016-02-22'),

209. (64, 22, '2015-04-03'),

210. (21, 11, '2017-12-09'),

211. (66, 29, '2016-12-20'),

212. (45, 13, '2017-04-15'),

213. (48, 30, '2015-01-31'),

214. (20, 25, '2017-12-20'),

215. (41, 20, '2018-01-29'),

216. (51, 12, '2015-07-05'),

217. (5, 1, '2015-04-12'),

218. (40, 3, '2018-02-24'),

219. (79, 4, '2018-06-27'),

220. (15, 10, '2016-11-01'),

221. (42, 22, '2016-12-28'),

222. (17, 9, '2018-01-29'),

223. (38, 13, '2016-05-09'),

224. (79, 2, '2017-12-06'),

225. (74, 3, '2015-12-07'),

226. (46, 8, '2016-06-05'),

227. (78, 22, '2018-08-11'),

228. (45, 2, '2015-04-20'),

229. (72, 31, '2015-11-11'),

230. (18, 17, '2015-03-21'),

231. (29, 3, '2017-08-13'),

232. (66, 11, '2018-06-05'),

233. (36, 16, '2016-04-28'),

234. (26, 2, '2016-10-23'),

235. (32, 1, '2017-10-31'),

236. (62, 14, '2017-07-25'),

237. (12, 4, '2015-07-08'),

238. (38, 32, '2015-02-24'),

239. (29, 16, '2016-07-28'),

240. (36, 25, '2017-05-07'),

241. (76, 7, '2015-06-13'),

242. (28, 16, '2016-08-15'),

243. (60, 13, '2016-08-26'),

244. (8, 3, '2017-07-28'),

245. (25, 1, '2016-07-30'),

246. (62, 29, '2018-08-24'),

247. (51, 8, '2016-09-01'),

248. (27, 23, '2015-02-08'),

249. (69, 12, '2018-06-25'),

250. (51, 12, '2015-07-04'),

251. (7, 4, '2015-05-01'),

252. (31, 15, '2017-10-29'),

253. (14, 23, '2015-01-15'),

254. (14, 1, '2018-05-21'),

255. (39, 25, '2015-12-26'),

256. (79, 24, '2016-05-31'),

257. (40, 15, '2016-03-18'),

258. (51, 13, '2018-04-13'),

259. (61, 1, '2015-02-11'),

260. (15, 24, '2018-03-02'),

261. (10, 22, '2018-01-21'),

262. (67, 10, '2017-07-08'),

263. (79, 11, '2016-12-11'),

264. (19, 32, '2016-05-04'),

265. (35, 11, '2017-08-01'),

266. (27, 13, '2017-12-15'),

267. (30, 22, '2015-12-22'),

268. (8, 7, '2015-06-26'),

269. (70, 9, '2016-03-20'),

270. (56, 18, '2016-01-29'),

271. (13, 19, '2015-03-06'),

272. (61, 2, '2016-06-18'),

273. (47, 13, '2017-09-18'),

274. (30, 22, '2016-02-19'),

275. (18, 22, '2016-12-31'),

276. (34, 29, '2017-10-27'),

277. (32, 21, '2015-06-03'),

278. (9, 28, '2016-03-30'),

279. (62, 24, '2015-03-23'),

280. (44, 22, '2017-04-29'),

281. (27, 5, '2015-03-25'),

282. (61, 28, '2017-07-14'),

283. (5, 13, '2016-12-04'),

284. (43, 19, '2018-03-15'),

285. (34, 19, '2016-06-05'),

286. (35, 5, '2018-02-19'),

287. (13, 12, '2016-09-23'),

288. (74, 18, '2016-12-26'),

289. (70, 31, '2017-08-15'),

290. (42, 17, '2016-06-15'),

291. (51, 24, '2018-07-30'),

292. (45, 30, '2015-01-15'),

293. (70, 17, '2017-10-07'),

294. (77, 7, '2017-01-06'),

295. (74, 25, '2015-09-25'),

296. (47, 14, '2018-02-01'),

297. (10, 2, '2017-04-18'),

298. (16, 21, '2016-10-03'),

299. (48, 5, '2016-09-17'),

300. (72, 3, '2017-02-10'),

301. (26, 23, '2016-03-01'),

302. (49, 23, '2016-10-25');

Question 1:

1. -- 1. Show all data in the Clients table

2.SELECT \* FROM Client;

Question 2:

1. /\* Future questions include age of borrowers and years born, as well as occupation. So we will index the clientDOB \*/

2. CREATE INDEX index\_client\_dob ON Client(clientDOB);

3. CREATE INDEX index\_client\_occ ON Client(Occupation);

1. /\* 2. List first name, last name, age, and occupation of all clients, use curdate wrapped in year function and subtract clientDoB to get age of client\*/

2. SELECT

3. ClientFirstName AS FirstName,

4. ClientLastName AS LastName,

5. YEAR(CURDATE()) - ClientDoB AS Age,

6. Occupation  
7. FROM Client;

Question 3:

1. /\* First and last names of clients that borrowed books in March 2018.

2. no need to have the same client name twice. Set distinct for only 1 client if they have

3. borrowed \*/

4. SELECT DISTINCT

5. c.ClientFirstName AS FirstName,

6. c.ClientLastName AS LastName

7. FROM Client c

8. /\* join each table on the clientID then filter specifically between 3-01 and before 4-01 \*/

9. JOIN Borrower b ON c.ClientID = b.ClientID

10. WHERE b.BorrowDate >= '2018-03-01' AND b.BorrowDate < '2018-04-01';

Question 4:

1. CREATE INDEX index\_book\_authorid ON Book(AuthorID);

2. CREATE INDEX index\_book\_genre ON Book(Genre);

3.  –- for joining and grouping authors/and years based on upcoming questions

1. /\* 4. Top 5 most-borrowed authors in 2017 (by number of times their books were

2. borrowed) \*/

3. SELECT a.AuthorFirstName,

4. a.AuthorLastName,

5. COUNT(\*) AS BorrowCount -- count number of times books are borrowed

6. FROM Borrower b

7. -- Join with Book to connect Borrower to Book data

8. JOIN Book bk ON b.BookID = bk.BookID

9. JOIN Author a ON bk.AuthorID = a.AuthorID

10. -- Filter borrow records to include only those in the year 2017

11. WHERE b.BorrowDate >= '2017-01-01' AND b.BorrowDate < '2018-01-01'

12. GROUP BY a.AuthorID

13. /\* Sort by borrow count in descending order to get the

14. most borrowed authors first \*/

15. ORDER BY BorrowCount DESC

16. -- Top 5 authors only

17. LIMIT 5;

Question 5:

1. /\* Select the nationalities of the 5 least borrowed authors from 2015 to 2017.

2. The wording leads me to believe the five nationalities with the fewest borrowed

3. books from 2015-2017. I recommend the wording changed to "Nationalities of

4. the 5 authors that clients borrowed the least from in 2015 to the end of 2017 \*/

5. -- nationality is not distinct since we are returning different authorIDs

6. SELECT AuthorNationality , BorrowCount

7. FROM (

8. SELECT a.AuthorNationality,

9. COUNT(\*) AS BorrowCount -- times of borrowed books

10. FROM Borrower b

11. -- Join to Book to get the AuthorID

12. JOIN Book bk ON b.BookID = bk.BookID

13. JOIN Author a ON bk.AuthorID = a.AuthorID

14. -- Filter from 2015 to 2017,

15. WHERE b.BorrowDate >= '2015-01-01' AND b.BorrowDate <= '2017-12-31'

16. -- Group by author to count borrows per author

17. GROUP BY a.AuthorID

18. -- Use ascending to get the least borrowed, then limit the list to 5

19. ORDER BY BorrowCount ASC

20. LIMIT 5

21. ) AS LeastBorrowed

22.  –- LeastBorrowed just set as a label for output readability

Question 6:

1. -- The book that was most borrowed during the years 2015-2017

2. SELECT

3. bk.BookTitle,

4. COUNT(\*) AS BorrowCount -– count borrowed books

5. FROM Borrower b

6. JOIN Book bk ON b.BookID = bk.BookID -- Join borrower table with book to get the title

7. WHERE b.BorrowDate BETWEEN '2015-01-01' AND '2017-12-31' -- limit dates

8. GROUP BY bk.BookID, bk.BookTitle -- indexing helped scan and sort with the index\_borrower\_bookid

9. ORDER BY BorrowCount DESC -- Start with highest count and pick #1 only

10. LIMIT 1;

Question 7:

1. -- Top borrowed genres for client born in years 1970-1980

2. -- Get the most borrowed genres for clients born between 1970 and 1980

3. SELECT bk.Genre, COUNT(\*) AS BorrowCount

4. FROM Client c

5. -- Join Borrower to get borrowing records

6. JOIN Borrower b ON c.ClientID = b.ClientID

7. -- Join Book to get genre information

8. JOIN Book bk ON b.BookID = bk.BookID

9. -- Filter clients born from 1970 to 1980

10. WHERE c.ClientDOB >= 1970 AND c.ClientDOB <= 1980

11. GROUP BY bk.Genre -- groups rows for the Count(\*)

12. ORDER BY BorrowCount DESC; -- get highest numbers first

Question 8:

1. -- Top 5 occupations that borrowed the most in 2016

2. SELECT

3. c.Occupation, -- get occupation from Client

4. COUNT(\*) AS BorrowCount -- Count how many books each occupation borrowed

5. FROM Borrower b

6. JOIN Client c ON b.ClientID = c.ClientID -- join clients to collect occupation/borrow count

7. WHERE b.BorrowDate BETWEEN '2016-01-01' AND '2016-12-31' -- Only consider borrows from 2016

8. GROUP BY c.Occupation -- Aggregate by occupation

9. ORDER BY BorrowCount DESC -- Get the highest borrowing occupations first

10. LIMIT 5; -- Return top 5

Question 9:  
 1. -- Average number of borrowed books by job title

2. SELECT

3. c.Occupation, -- get client occupation

4. ROUND(COUNT(b.BorrowID) / COUNT(DISTINCT c.ClientID), 2) AS AvgBorrowsPerClient

5. /\* Count total borrows and divide by number of unique clients in each occupation

6. rounding to get the avg and creating a columnname Avgborrowsperclient

7. set the final value of round to 0 for the nearest whole number \*/

8. FROM Borrower b

9. JOIN Client c ON b.ClientID = c.ClientID -- link borrow record to client who made it

10. GROUP BY c.Occupation -- Group by job title

11. ORDER BY AvgBorrowsPerClient DESC; -- Highest averages first

Question 10:

1. SELECT COUNT(\*) FROM Client; -- get total number of clients and multiply by .2 = 80\*.2 = 16

1. /\* Create a VIEW and display the titles that were borrowed

2. by at least 20% of clients \*/

3. CREATE VIEW BooksBorrowedBy20Percent AS

4. -- Select the book title and the number of distinct clients who borrowed it

5. SELECT

6. bk.BookTitle, -- Display the title of the book

7. COUNT(DISTINCT b.ClientID) AS DistinctBorrowers

8. -- Count how many unique clients borrowed the book and create a column

9. FROM Borrower b

10. -- Join the Book table to get book titles using BookID

11. JOIN Book bk ON b.BookID = bk.BookID

12. -- Group the data by each book

13. GROUP BY bk.BookID, bk.BookTitle

14. -- Keep only books that were borrowed by at least 16 unique clients

15. HAVING COUNT(DISTINCT b.ClientID) >= 16;

1. -- Show the contents of the view created

2. SELECT \* FROM BooksBorrowedBy20Percent;

Question 11:

1. -- The top month of borrows in 2017

2. SELECT

3. MONTH(BorrowDate) AS Month, -- Get the month number (1–12), set column name

4. COUNT(\*) AS TotalBorrows /\* Count how many borrows

5. happened that month and set column name \*/

6. FROM Borrower

7. WHERE YEAR(BorrowDate) = 2017 -- Only include records from the year 2017

8. GROUP BY MONTH(BorrowDate) -- Group by each month to get totals

9. ORDER BY TotalBorrows DESC -- Order so highest number of borrows is first

10. LIMIT 1; -- Only insert the top month

Question 12:

1. -- Average number of borrows by age

2. SELECT

3. YEAR(CURDATE()) - c.ClientDOB AS Age,

4. -- Calculate client age based on current year

5. ROUND(COUNT(b.BorrowID) / COUNT(DISTINCT c.ClientID), 0) AS AvgBorrows

6. -- Total borrows / unique clients of that age, Rounded for clean numbers

7. FROM Client c

8. JOIN Borrower b ON c.ClientID = b.ClientID

9. -- Join to get borrow records for each client

10. GROUP BY Age -- Group by calculated age

11. ORDER BY Age; -- Order results by age ascending

Question 13:

1. -- The oldest and the youngest clients of the library

2. (

3. SELECT

4. ClientFirstName,

5. ClientLastName,

6. ClientDOB,

7. YEAR(CURDATE()) - ClientDOB AS age, -- put age in a column

8. 'Youngest' AS Label -- get name and label youngest age

9. FROM Client

10. ORDER BY ClientDOB DESC -- Most recent birth date = youngest

11. LIMIT 1

12. )

13. UNION ALL -- combine both results into a table

14. (

15. SELECT

16. ClientFirstName,

17. ClientLastName,

18. ClientDOB,

19. YEAR(CURDATE()) - ClientDOB AS age, -- put their age in a column

20. 'Oldest' AS Label -- get name and label oldest

21. FROM Client

22. ORDER BY ClientDOB ASC -- Earliest birth date = oldest

23. LIMIT 1

24. );

Question 14:

1. -- First and last names of authors that wrote books in more than one genre

2. SELECT

3. a.AuthorFirstName,

4. a.AuthorLastName -- get first and last name

5. FROM Author a

6. -- Join to Book table to access genres

7. JOIN Book b ON a.AuthorID = b.AuthorID

8. -- Grouping by each author to count unique genres

9. GROUP BY a.AuthorID, a.AuthorFirstName, a.AuthorLastName

10. HAVING COUNT(DISTINCT b.Genre) > 1;

11. -- Only keep authors with more than one unique genre

NO ROWS RETURNED WITH THIS QUERY. CHECKING AUTHORS AND BOOKS TO SEE IF THEY HAVE MORE THAN ONE GENRE

1. SELECT

2. a.AuthorID,

3. a.AuthorFirstName,

4. a.AuthorLastName,

5. COUNT(DISTINCT b.Genre) AS GenreCount

6. FROM Author a

7. JOIN Book b ON a.AuthorID = b.AuthorID

8. GROUP BY a.AuthorID, a.AuthorFirstName, a.AuthorLastName

9. ORDER BY GenreCount DESC;

No Genrecount more than 1