**DBS Lab 4**

**Name**:

**1. Select the population (count) of users, grouped by state. Only use the ‘home’ address to determine the user’s home state. If a user has 2 home addresses, count both of them. Paste the SQL below.**

|  |
| --- |
| select state, count(\*) AS usersState  FROM useraddress  WHERE userAddressTypeId =1  GROUP BY state; |

**2. Select the count of DVDs released grouped by year. Only show the last 10 years with the current year on top. Paste the SQL below.**

|  |
| --- |
| SELECT `Year` AS `year`, count(\*) AS dvdReleases  FROM dvd  WHERE `Year` >= '2004-01-01'  GROUP BY `Year`  HAVING `Year` != 'VAR'  AND `Year` != 'UNK'  AND `Year` != '211'  ORDER BY `Year` DESC; |

**3. Select the average publicRating for all DVDs by genre released in 2002. Exclude genres where the average is less then 2.2. (Use a HAVING clause) Paste the SQL below.**

|  |
| --- |
| SELECT genre, avg(publicRating) AS avgPubRating  FROM dvd  WHERE `Year` LIKE '2002%'  GROUP BY genre  HAVING avgPubRating > 2.2; |

**4. Select the average age for all male, college/university professors. Use the datediff function to compute age from DOB. Paste the SQL below.**

|  |
| --- |
| SELECT avg(datediff(now(), dob) / 365.25)  AS averageProfessorAge  FROM users  WHERE occupationId = 164; |

**5. Create a new table called DVDNormal with the columns dvdId, dvdTitle, year, publicRating, dvdStudioId, dvdStatusId and dvdGenreId. ‘Id’ fields should be integers. DVDId should not be auto assigned. Look at the structure of the DVD table for the other field types.**

**Paste the SQL below.**

|  |
| --- |
| CREATE TABLE `DVDNormal` (  `dvdId` int(11) unsigned NOT NULL,  `dvdTitle` varchar(255),  `year` varchar(255),  `publicRating` decimal(4,2),  `dvdStudioId` int(11) DEFAULT NULL,  `dvdStatusId` int(11) DEFAULT NULL,  `dvdGenreId` int(11) DEFAULT NULL,  PRIMARY KEY (`dvdId`)  ) ENGINE=InnoDB DEFAULT CHARSET=latin1; |

**6. Create the supporting tables for dvdNormal table including dvdStudio, dvdStatus, & dvdGenre. Each table should have 2 columns. The ‘id’ fields should be integers and auto increment. Paste the SQL below.**

|  |
| --- |
| CREATE TABLE `dvdStudio` (  `dvdStudioId` int(11) NOT NULL AUTO\_INCREMENT,  `dvdStudio` VARCHAR(255),  PRIMARY KEY (`dvdStudioId`)  ) ENGINE=InnoDB DEFAULT CHARSET=latin1;  CREATE TABLE `dvdStatus` (  `dvdStatusId` int(11) NOT NULL AUTO\_INCREMENT,  `dvdStatus` VARCHAR(255),  PRIMARY KEY (`dvdStatusId`)  ) ENGINE=InnoDB DEFAULT CHARSET=latin1;  CREATE TABLE `dvdGenre` (  `dvdGenreId` int(11) NOT NULL AUTO\_INCREMENT,  `dvdGenre` VARCHAR(255),  PRIMARY KEY (`dvdGenreId`)  ) ENGINE=InnoDB DEFAULT CHARSET=latin1; |

**7. Create 3 foreign keys between the DVDNormal table and the 3 supporting tables. Turn cascading on for updates and deleted. Paste the 3 SQL statements below.**

|  |
| --- |
| ALTER TABLE DVDNormal  ADD FOREIGN KEY (dvdStudioId)  REFERENCES dvdStudio(dvdStudioID)  on delete cascade  on update cascade;  ALTER TABLE DVDNormal  ADD FOREIGN KEY (dvdStatusId)  REFERENCES dvdStatus(dvdStatusId)  on delete cascade  on update cascade;  ALTER TABLE DVDNormal  ADD FOREIGN KEY (dvdGenreId)  REFERENCES dvdGenre(dvdGenreId)  on delete cascade  on update cascade; |

**8. Create unique indexes on the 3 new tables for the name columns.**

**Paste the 3 SQL statements below.**

|  |
| --- |
| CREATE UNIQUE INDEX `dvdStudio` ON dvdStudio(`dvdStudio`);  CREATE UNIQUE INDEX `dvdStatus` ON dvdStatus(`dvdStatus`);  CREATE UNIQUE INDEX `dvdGenre` ON dvdGenre(`dvdGenre`); |

**9. Create indexes on the DVD table for the studio, status and genre columns.**

**Paste the 3 SQL statements below.**

|  |
| --- |
| create index `Studio`  on dvd (`Studio`);  create index `Status`  on dvd (`Status`);  create index `Genre`  on dvd (`Genre`); |

**10. Insert into the 3 new tables (dvdStudio, dvdStatus, & dvdGenre) by selecting the unique values from the DVD table. Paste the 3 SQL statements below.**

|  |
| --- |
| insert into dvdStudio (dvdStudio)  (  select distinct (Studio) from dvd  order by Studio  );  insert into dvdStatus (dvdStatus)  (  select distinct (`Status`) from dvd  order by `Status`  );  insert into dvdGenre (dvdGenre)  (  select distinct (Genre) from dvd  order by Genre  ); |

**11. Insert into the dvdNormal table by joining the dvd, dvdStudio, dvdStatus, and dvdGenre tables Paste the SQL statement below. (AS DVD is a large table, this may run for several mins.)**

|  |
| --- |
| insert into DVDNormal(dvdId, dvdTitle, year, publicRating, dvdStudioId, dvdStatusId, dvdGenreId)  (SELECT dvd.dvdId, dvd.DVD\_Title, dvd.Year, dvd.publicRating, dvdStudio.dvdStudioId, dvdStatus.dvdStatusId, dvdGenre.dvdGenreId  FROM dvd  join dvdStudio on dvd.Studio = dvdStudio.dvdStudio  join dvdStatus on dvd.Status = dvdStatus.dvdStatus  join dvdGenre on dvd.Genre = dvdGenre.dvdGenre  ); |

**12. Create a view to include the following fields: dvdId, dvd\_title, dvdStudio, dvdStatus, dvdGenre. Do not use the dvd table in the select. Use the dvdNormal, dvdStudio, dvdStatus, & dvdGenre tables.**

**Paste SQL below.**

|  |
| --- |
| CREATE VIEW DVDView AS  (select dvdnormal.dvdID, dvdnormal.dvdTitle, dvdStatus.dvdstatus, dvdStudio.dvdstudio, dvdGenre.dvdgenre  FROM DVDNormal  join dvdstatus on dvdnormal.dvdStatusId = dvdStatus.dvdStatusId  join dvdstudio on dvdnormal.dvdStudioId = dvdStudio.dvdStudioId  join dvdGenre on dvdnormal.dvdGenreId = dvdGenre.dvdGenreId  ); |

**13. The userDVD table connects users with the DVDs they own. Select the count of DVDs owned by each user. (Use count & group by.)**

**Paste SQL below.**

|  |
| --- |
| select userID, count(\*) AS userDVDcount  FROM userdvd  GROUP BY userId; |

**14. Select the userId, firstname, lastname, DVDCount and gender of the female with the most DVDs. Use the answer from the previous question as an inline view.**

**Paste SQL below.**

|  |
| --- |
| create view COUNTVIEW AS  select userId, count(\*) AS userDVDcount  FROM userdvd  GROUP BY userId;  select users.userId, users.firstname, users.lastname, userDVDcount, users.gender  FROM COUNTVIEW  join users on COUNTVIEW.userId = users.userId  WHERE userDVDcount = (select MAX(userDVDcount) AS MaxCount  From COUNTVIEW); |

To complete this assignment, upload this document to FSO.