CSE 414 Section 2

## 0. Joins Examples

Given tables created with these commands:

|  |
| --- |
| CREATE TABLE A (a int);  CREATE TABLE B (b int);  INSERT INTO A VALUES (1), (2), (3), (4);  INSERT INTO B VALUES (3), (4), (5), (6); |

What’s the output for each of the following:

|  |
| --- |
| SELECT \* FROM A INNER JOIN B ON A.a=B.b;  A.a B.b  3 3  4 4 |
| SELECT \* FROM A LEFT OUTER JOIN B ON A.a=B.b;  A.a B.b  3 3  4 4  1  2 |
| SELECT \* FROM A RIGHT OUTER JOIN B ON A.a=B.b;  A.a B.b  5  6  3 3  4 4 |
| SELECT \* FROM A FULL OUTER JOIN B ON A.a=B.b;  A.a B.b  5  6  1  2  3 3  4 4 |

## 1. SQL Practice

|  |
| --- |
| CREATE TABLE Movies (  id int,  name varchar(30),  budget int,  gross int,  rating int,  year int,  PRIMARY KEY (id)  ); |
| CREATE TABLE Actors (  id int,  name varchar(30),  age int,  PRIMARY KEY (id)  ); |
| CREATE TABLE ActsIn (  mid int,  aid int,  FOREIGN KEY (mid) REFERENCES Movies (id),  FOREIGN KEY (aid) REFERENCES Actors (id)  PRIMARY KEY (mid, aid)  ); |

|  |
| --- |
| What is the number of movies, and the average rating of all movie that the actor ”Patrick Stewart” has appeared in?  SELECT COUNT(\*), AVG(M.rating)  FROM Movies AS M, ActsIn AS AI, Actors AS A  WHERE M.id = AI.mid  AND A.id = AI.aid  AND A.name='Patrick Stewart'; |
| What is the minimum age of an actor who has appeared in a movie where the gross of the movie has been over $1,000,000,000?  SELECT MIN(A.age)  FROM Movies AS M, ActsIn AS AI, Actors AS A  WHERE M.id = AI.mid  AND A.id = AI.aid  AND M.gross > 1000000000; |
| ~~What is the total budget of the movies released in each year, where the oldest actor is less than 30?~~  ~~SELECT M.year, SUM(M.budget)~~  ~~FROM Movies AS M, ActsIn AS AI, Actors AS A~~  ~~WHERE M.id = AI.mid~~  ~~AND A.id = AI.aid~~  ~~GROUP BY M.year~~  ~~HAVING MAX(A.age) < 30;~~  ~~(Edge Case: Assuming that there are no 2 under 30 oldest actors with the same age are in the same movie~~  ~~Ex: A movie with 2 actors, both of which are 29 years old would be counted twice.~~  ~~)~~  **Question is deceptively complex and the correct solution requires subqueries.** |

## 2. Self Join

Consider the following over simplified Employee table

|  |
| --- |
| CREATE TABLE Employees (  id int,  bossOf int  ); |

|  |  |
| --- | --- |
| Suppose all employees have an id which is not null. How would we find the id of all employees who are the boss of at least one other employee? | |
| SELECT DISTINCT e2.id  FROM Employees AS e1  INNER JOIN Employees AS e2  ON e2.bossOf=e1.id;  SELECT DISTINCT e2.id  FROM Employees AS e1, Employees AS e2  WHERE e2.bossOf=e1.id; | SELECT DISTINCT e.id  FROM Employees AS e  WHERE e.bossOf IS NOT NULL;  Technically does not work because someone may be the boss of an id that is not employee.  (Ex: Someone was fired, and the database did not completely update the bossOf) |

What do we select? (select \* vs select table alias.col name)

|  |
| --- |
| Consider the case with employees (1, NULL), (2, NULL), (5, 1), (5, 2), (5, NULL), (3, NULL). How many employees is id=5 the boss of?  With SELECT COUNT(\*): 3  With SELECT COUNT(bossOf): 2 |

## 3. Notes:

|  |  |
| --- | --- |
| SUM() MIN()  MAX() AVG()  COUNT() | FW**G**HOS HAVING [condition]  ORDER BY [colname]  GROUP BY [colname] |