INF10004 Database analysis and Design

Week 05

Theory Questions

Q1. Describe the purpose of the MIN aggregate function?

Q2. **How many rows** in the result for this statement?
SELECT MIN(BirthYear) FROM student

Consider this table named STUDENT

StuId	*Name*	*Gender*	*BirthYear*	*Degree*	*SubjectsPassed*
1	*Fred*	*M*	*1991*	*BIT*	*0*
2	*Sue*	*F*	*1993*	*ICT*	*4*
3	*Emma*	*F*	*1992*	*BIS*	*2*
4	*Dave*	*M*	*1993*	*BIT*	*4*
5	*Leah*	*F*	*1991*	*ICT*	*6*
6	*Linda*	*F*	*1991*	*BIS*	*6*
7	*Harry*	*M*	*1992*	*BIS*	*4*

Q3. Write a single SQL statement that counts the total number of students

Q4. Write a single SQL statement that counts the total number of students that are female

Q5. Write a **single** SQL statement that counts the total number of students of each gender Hint: Use a Group By clause

Q6. Write a **single** SQL statement that counts the total number of students of each degree Hint: Use a Group By clause

Q7. Write a **single** SQL statement that counts the total number of students of each degree.

Does not list any total values less than 2

Hint: Use a Group By clause and a Having clause

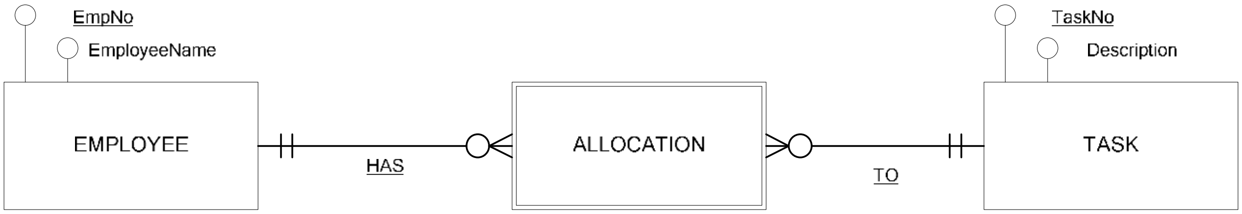
Q8. Write a **single** SQL statement that calculates the average number of subjects passed by each Degree.

Only include Female students (ignore all males)

Does not list any total values less than 3

Hint: Use a Where, Group By and Having clause

Q9. Consider the following ERD



a. What is the identifier of the Allocation entity?

b. How many foreign keys will exist in the Allocation table?

c. Write the Relational Schema. Indicate all PKs & FKs.

d. Write SQL statements to

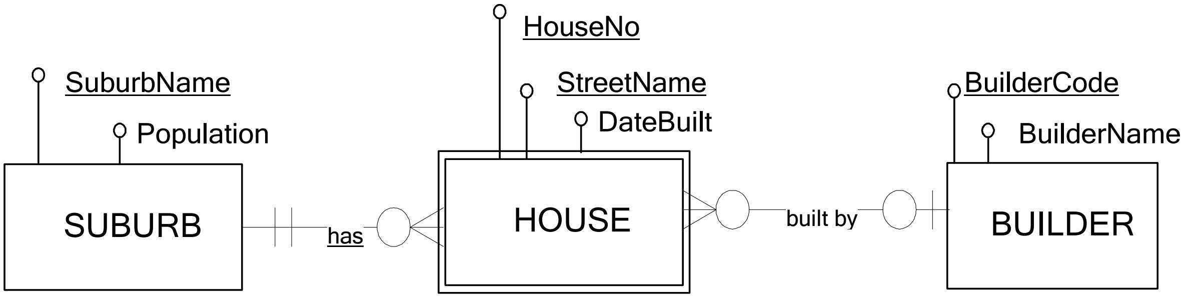
i. create employees Jim Black and Perry White

ii. create tasks Bake Cake and Chop Onions

iii. allocate Jim Black to the Chop Onions task

iv. allocate Perry White to the Bake Cake task

Q10. Consider the following business case and diagram:



a. What is the Identifier of each entity

Convert the above ERD to a relational schema.

Q11. Consider the following SQL statement:

a. Does this statement have any ***column*** *constraints? If so name them.*

b. Does this statement have any **table** constraints? If so name them.

```
CREATE TABLE ROOM (
  BuildingCode varchar(10)
, OfficeNo        number
, PhoneNo         varchar(10)
, OfficeSize      number Not Null
, Primary Key (BuildingCode, OfficeNo)
);
```

Lab Questions

Download the script named
CREATE_MOVIE_RATING_MOVIECOLOUR_TABLES.TXT from Blackboard.

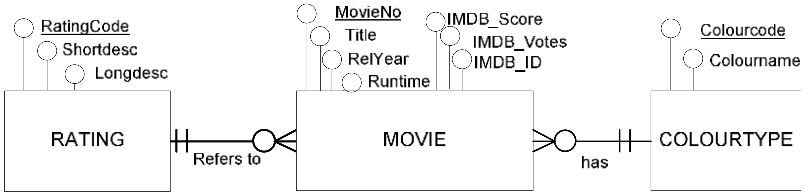
The script:

- *drops all a number of tables associated with the Movie database that may already exist*
- creates 3 tables named MOVIE, RATING, MOVIECOLOUR
- inserts data into all 3 tables (same as lab 2)

Execute the script in iSQL_Jr

Familiarize yourself with the column names of these tables by **executing** these statements in ISQL JR and examining this **ERD**.

DESCRIBE
moviecolour;
DESCRIBE rating;
DESCRIBE movie;

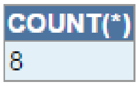


Each of these tables has a Primary Key constraint. Some tables have Foreign Key constraints.

Q1. List the Movie number, movie title, colour name and the long rating description of every movie.

Q2. Same as above but only for movies made in this period 2003 to 2008.

Q3. Using the **Count(*)** function, display the number of the movies made in 2005



COUNT(*)
8

The format of the output should look like this (although the value may change)

To see if you have the correct answer, execute this code:
SELECT * FROM MOVIE WHERE RELYEAR = 2005

Q4. Using the **Count(*)** function, display the number of the movies made within the years 2010-2015

Check you answer by executing a SELECT statement lists all rows in that year range

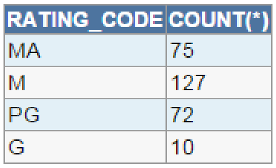
Q5. Using the **Count(*)** function, list the movies with a PG rating code

Check you answer by executing a SELECT statement lists all rows for that rating code

Aggregate expression / Group By Questions

Q6. Using the Count(*) and the **GROUP BY clause**, list the number of movies of each rating code

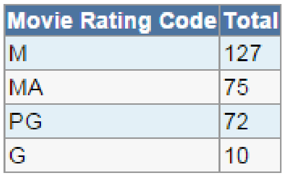
Your values may be different (as more movies may have been added). Check you answer by listing all G movies then list all PG movies. The values Show should give you a good indication if your Group By query is correct.



RATING_CODE	COUNT(*)
MA	75
M	127
PG	72
G	10

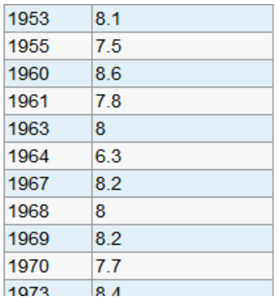
Q7. Same as above but the list must appear in **descending count sequence**

Q8. Same as above but **change the headings** of both columns as shown



Movie Rating Code	Total
M	127
MA	75
PG	72
G	10

Q9. Using the Count(*) and the **GROUP BY clause**, list maximum IMDB_SCORE for each release year in the movie table. The list must be in ascending year sequence.



1953	8.1
1955	7.5
1960	8.6
1961	7.8
1963	8
1964	6.3
1967	8.2
1968	8
1969	8.2
1970	7.7
1973	8.4

Q10. Download the script named **CREATE_ACTOR_TABLE.TXT** from Blackboard. The script:

- *drops all a number of tables associated with the Movie database that may already exist*
- creates one tables named actor
- inserts data into the actor table

Execute the script in iSQL_Jr

Q11. List all columns from the Actor table only born in the USA.

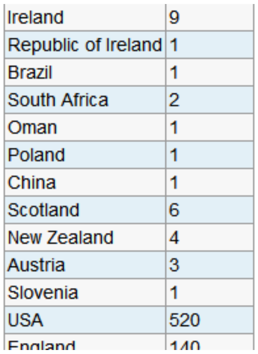
Q12. Using the Count(*) function, list the number of actors born in the USA

Q13. Using the Count(*) function, list the number of actors born in the Australia

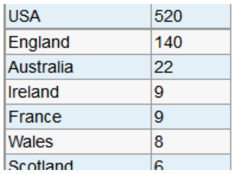
Q14. Using the Count(*) function, list the number of actors whose bornin value IS NULL

Q15. Using the Count(*) function, list the number of actors whose birthdate value IS NULL

Q16. Using the Count(*) and the GROUP BY clause, list the number of actors for each bornin value
(i.e. count the number of actors born in every country)



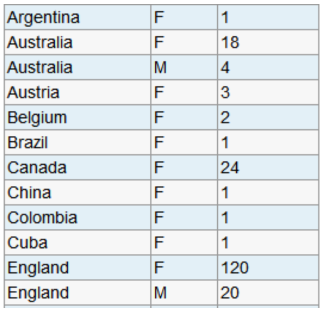
Ireland	9
Republic of Ireland	1
Brazil	1
South Africa	2
Oman	1
Poland	1
China	1
Scotland	6
New Zealand	4
Austria	3
Slovenia	1
USA	520
England	140



USA	520
England	140
Australia	22
Ireland	9
France	9
Wales	8
Scotland	6

Q17. Same as above but order the result in **descending** count values.

Q18. Using the Count(*) and the GROUP BY clause, list the number of actors **of each gender** that were born in each county



Argentina	F	1
Australia	F	18
Australia	M	4
Austria	F	3
Belgium	F	2
Brazil	F	1
Canada	F	24
China	F	1
Colombia	F	1
Cuba	F	1
England	F	120
England	M	20

Note: The results may look a bit surprising.
Some countries only have actors of one gender in the Actor table (e.g. Belgium). Other countries have a number of actors of each gender (e.g. England).

Group By Having Questions

Q19. The result set of the previous question shows many country and gender combinations.

The count column shows many rows with the value 1.

Using the **Having clause**, remove all rows that contain a **count value less than 2** from the previous query.

Q20. Using the Group Clause show the count of the **number of movies made in each year.** The list must be in ascending count sequence

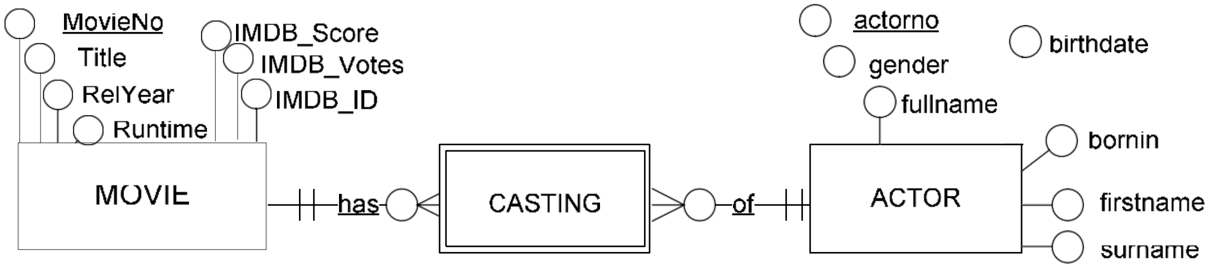
Q21. Same as above but add a **Having Clause** so that only years with a **count of 6 or greater** appear in the result set.

Download the script named **CREATE_CASTING_TABLE.TXT** from Blackboard.

The script:

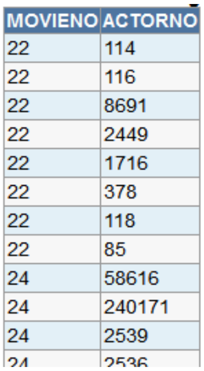
- drops any existing table named Casting
- creates a table named casting
- inserts data into this table

Execute the script in iSQL_Jr



Q22. Write the SQL statement to list the rows from **Casting** in Movieno order

What is the PK of Movie? What is the PK of Actor? What is the FK of Casting?

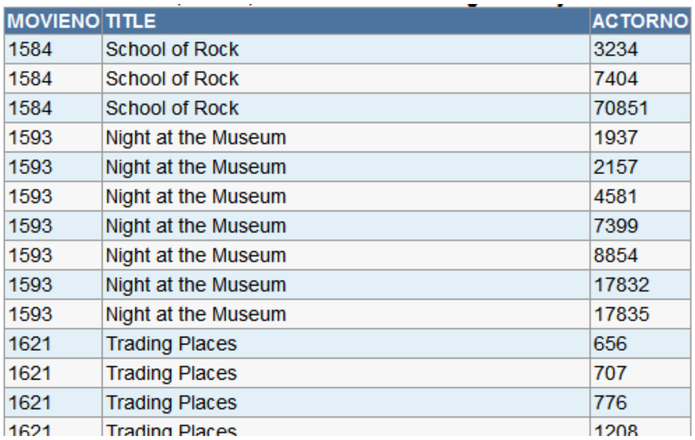


MOVIENO	ACTORNO
22	114
22	116
22	8691
22	2449
22	1716
22	378
22	118
22	85
24	58616
24	240171
24	2539
24	2536

Movie Actor Casting Join and Group By Questions

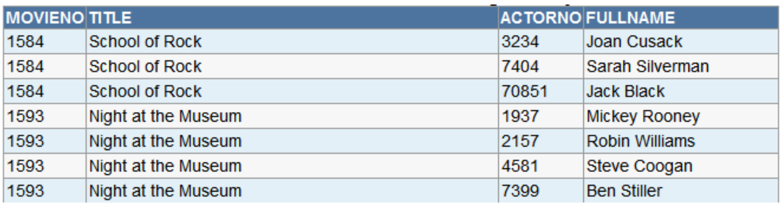
Q23. Write the SQL statement to list the movieno and actorno from all rows of the Casting table in Movieno sequence.

Q24. Using an **Inner Join,** modify the SQL statement above to display the **movie title**, **movie number** and **actor no** from the **Casting and Movie** tables.



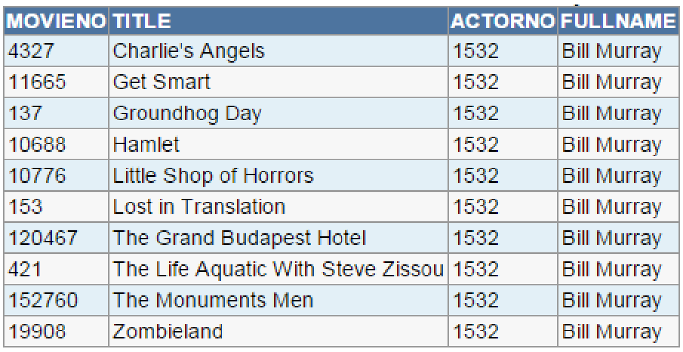
MOVIENO	TITLE	ACTORNO
1584	School of Rock	3234
1584	School of Rock	7404
1584	School of Rock	70851
1593	Night at the Museum	1937
1593	Night at the Museum	2157
1593	Night at the Museum	4581
1593	Night at the Museum	7399
1593	Night at the Museum	8854
1593	Night at the Museum	17832
1593	Night at the Museum	17835
1621	Trading Places	656
1621	Trading Places	707
1621	Trading Places	776
1621	Trading Places	1208

Q25. Using **two Inner Joins**, modify the SQL statement above so that each actor's fullname is also displayed.



MOVIENO	TITLE	ACTORNO	FULLNAME
1584	School of Rock	3234	Joan Cusack
1584	School of Rock	7404	Sarah Silverman
1584	School of Rock	70851	Jack Black
1593	Night at the Museum	1937	Mickey Rooney
1593	Night at the Museum	2157	Robin Williams
1593	Night at the Museum	4581	Steve Coogan
1593	Night at the Museum	7399	Ben Stiller

Q26. Same as above but only display rows if the actor's fullname is Bill Murray



MOVIENO	TITLE	ACTORNO	FULLNAME
4327	Charlie's Angels	1532	Bill Murray
11665	Get Smart	1532	Bill Murray
137	Groundhog Day	1532	Bill Murray
10688	Hamlet	1532	Bill Murray
10776	Little Shop of Horrors	1532	Bill Murray
153	Lost in Translation	1532	Bill Murray
120467	The Grand Budapest Hotel	1532	Bill Murray
421	The Life Aquatic With Steve Zissou	1532	Bill Murray
152760	The Monuments Men	1532	Bill Murray
19908	Zombieland	1532	Bill Murray

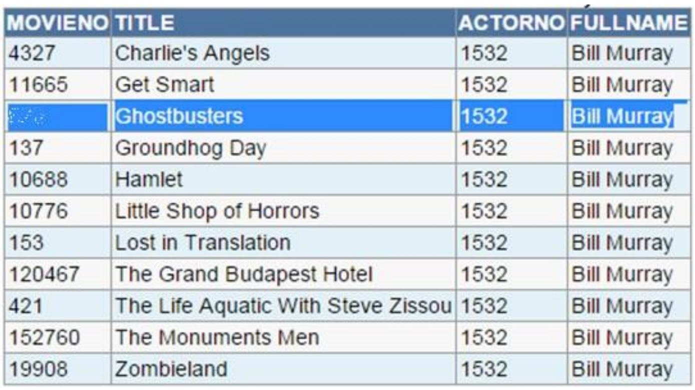
Adding Data to Movie Actor Casting Tables

Q27. The previous query showed a **major flaw** in the casting table data. Bill Murray has **not** been cast into the movie titled 'Ghostbusters'! *(Ghostbusters is one of Bill Murry's most well-known movies)*

To fix this problem, do the following:

- Take note of Bill Murry's actorno
- Write a query that displays the movieno of the movie 'Ghostbusters'
- Write an insert statement that will cast Bill Murray into the 'Ghostbusters' movie. (Hint: View the insert statements in the **CREATE_CASTING_TABLE.TXT** script to see the format that is required).

Q28. Finally, repeat the query above that displays all of Bill Murray's movies. This time Ghostbusters should appear.



MOVIENO	TITLE	ACTORNO	FULLNAME
4327	Charlie's Angels	1532	Bill Murray
11665	Get Smart	1532	Bill Murray
[illegible]	Ghostbusters	1532	Bill Murray
137	Groundhog Day	1532	Bill Murray
10688	Hamlet	1532	Bill Murray
10776	Little Shop of Horrors	1532	Bill Murray
153	Lost in Translation	1532	Bill Murray
120467	The Grand Budapest Hotel	1532	Bill Murray
421	The Life Aquatic With Steve Zissou	1532	Bill Murray
152760	The Monuments Men	1532	Bill Murray
19908	Zombieland	1532	Bill Murray