|  |  |  |
| --- | --- | --- |
| **Name** | **ID** | **STUDENT SIGN** |
| FARJANA YESMIN OPI | 22-47018-1 |  |

**Class Test 01**

**Hogwarts** is a school of witchcraft and wizardry. To ensure proper management of their data the renowned school has decided to maintain a database system. Out of many bidders your company was hired to accomplish the task. Your job is to create a relational database for Hogwarts from the requirements specified below:

RDBMS- Oracle 10g

Language-SQL

Log in as User System and create a ***user*** Dumbledore who has ***password*** Phoenix. Dumbledore is granted ***unlimited tablespace***. He is also granted the permission to ***create*** tables. After logging in with his username and password Dumbledore creates ***two tables*** i.e. Student and House. ***Student*** table has five columns containing information about students ***Identification Number, Name, CGPA, Blood Status and House Number***. ***House*** table has three columns containing information about ***House Number, House Name and House Points***. Here S\_Id, H\_Id are the ***primary key columns*** of Student and House table respectively. Student table also has a ***foreign key*** column H\_No. Constraint should be applied in such a way that CGPA cannot be greater than 4.00 and House name cannot be NULL. The two tables along with their inserted data are given below:

**Table: Student Table: House**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S\_Id | S\_Name | S\_CGPA | S\_BloodStatus | H\_No |
| 2 | Harry | 3.45 | Halfblood | 11 |
| 7 | Ron | 3.01 | Pureblood | 11 |
| 12 | Hannah |  | Pureblood | 22 |
| 17 | Cedric | 3.78 | Pureblood | 22 |
| 22 | Cho | 3.55 | Muggleborn | 33 |
| 27 | Luna | 2.89 |  | 33 |
| 32 | Draco | 3.88 | Pureblood | 44 |
| 37 | Goyle | 2.10 | Pureblood | 44 |

|  |  |  |
| --- | --- | --- |
| H\_Id | H\_Name | H\_Points |
| 11 | Gryffindor | 892 |
| 22 | Hufflepuf | 785 |
| 33 | Ravenclaw | 789 |
| 44 | Slytherin | 850 |

After creating the tables and inserting data based on provided requirements write Queries (Write down the question and also the answer. Give screenshot of the result of the query.You can add more Answer Box if required) according to the following specification:

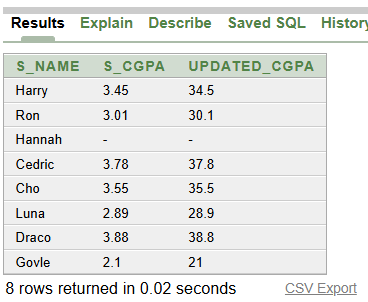
-using **ARITHMETIC** operator

Question: Show each student’s CGPA and also show their CGPA multiplied by 10.

Answer:

SELECT S\_Name, S\_CGPA, S\_CGPA \* 10 AS Updated\_CGPA FROM Student;

Screenshot:



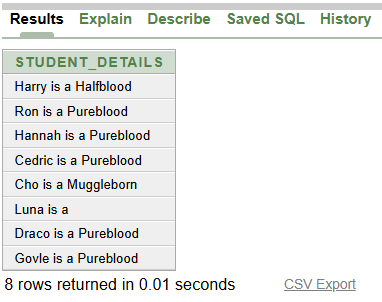
-using **CONCATENATION** operator

Question: Show each student’s name along with their blood status using one sentence.

Answer:

SELECT S\_Name || ' is a ' || S\_BloodStatus AS Student\_Details FROM Student;

Screenshot:



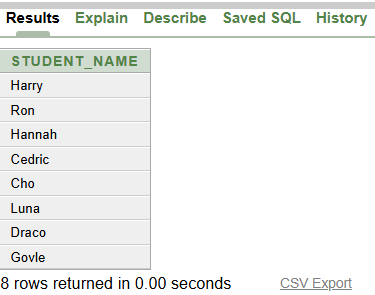
-using **COLUMN ALIAS**

Question: Show all student names but rename the column as "Student\_Name".

Answer:

SELECT S\_Name AS Student\_Name FROM Student;

Screenshot:



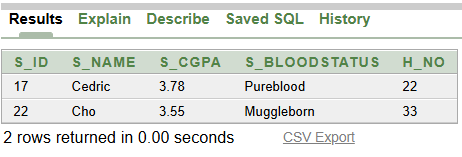
-using **LIKE** operator

Question: Find students whose names start with the letter "C".

Answer:

SELECT \* FROM Student WHERE S\_Name LIKE 'C%';

Screenshot:



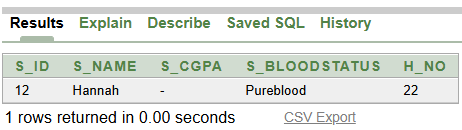
-using **IS NULL** operator

Question: Find students who do not have a CGPA recorded.

Answer:

SELECT \* FROM Student WHERE S\_CGPA IS NULL;

Screenshot:



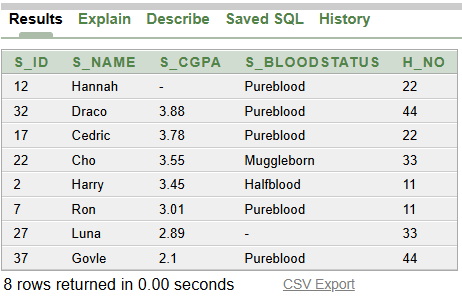
-using **ORDER BY** clause

Question: Show all students and arrange them from the highest to the lowest CGPA.

Answer:

SELECT \* FROM Student ORDER BY S\_CGPA DESC;

Screenshot:



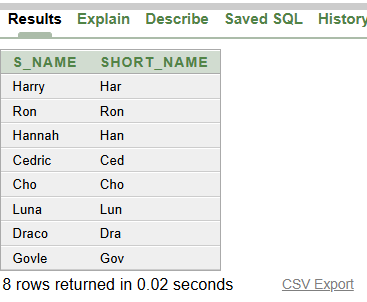
-using **SUBSTR** function

Question: Show each student’s name along with only the first three letters of their name.

Answer:

SELECT S\_Name, SUBSTR(S\_Name, 1, 3) AS Short\_Name FROM Student;

Screenshot:



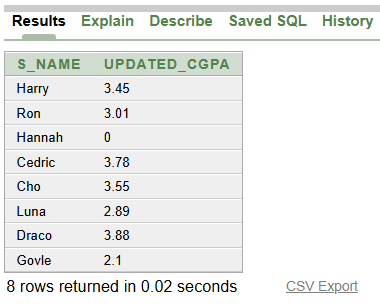
-using **NVL** function

Question: If a student’s CGPA is missing, show it as 0 instead of NULL.

Answer:

SELECT S\_Name, NVL(S\_CGPA, 0) AS Updated\_CGPA FROM Student;

Screenshot:



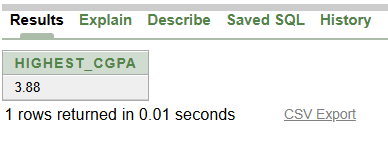
-using **MAX** function

Question: Find the highest CGPA among all students.

Answer:

SELECT MAX(S\_CGPA) AS Highest\_CGPA FROM Student;

Screenshot:



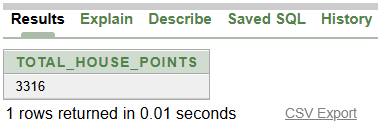
-using **SUM** function

Question: Find the sum of all house’s points.

Answer:

SELECT SUM(H\_Points) AS Total\_House\_Points FROM House;

Screenshot:



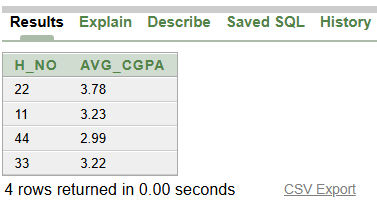
-using **GROUP BY** clause

Question: Find the average CGPA of students in each house.

Answer:

SELECT H\_No, AVG(S\_CGPA) AS Avg\_CGPA FROM Student GROUP BY H\_No;

Screenshot:



-using **HAVING** clause

Question: Find houses where the average CGPA of students is more than 3.0.

Answer:

SELECT H\_No, AVG(S\_CGPA) AS Avg\_CGPA

FROM Student

GROUP BY H\_No

HAVING AVG(S\_CGPA) > 3.0;

Screenshot:

