|  |  |  |
| --- | --- | --- |
| **Name** | **ID** | **STUDENT SIGN** |
| **Sheikh Rubaeid Sanjid** | **21-45844-3** |  |

**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

-using **CONCATENATION** operator

-using **COLUMN ALIAS**

-using **LIKE** operator

-using **IS NULL** operator

-using **ORDER BY** clause

-using **SUBSTR** function

-using **NVL** function

-using **MAX** function

-using **SUM** function

-using **GROUP BY** clause

-using **HAVING** clause

Answer:

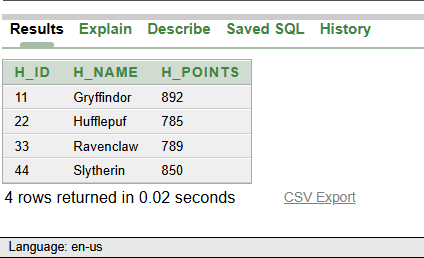
CREATE TABLE House (

H\_Id NUMBER PRIMARY KEY,

H\_Name VARCHAR2(50) NOT NULL,

H\_Points NUMBER

);



CREATE TABLE Student (

S\_Id NUMBER PRIMARY KEY,

S\_Name VARCHAR2(50),

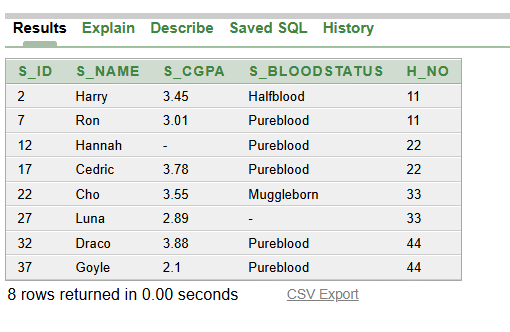
S\_CGPA NUMBER(3,2) CHECK (S\_CGPA <= 4.00),

S\_BloodStatus VARCHAR2(20),

H\_No NUMBER,

CONSTRAINT fk\_house FOREIGN KEY (H\_No) REFERENCES House(H\_Id)

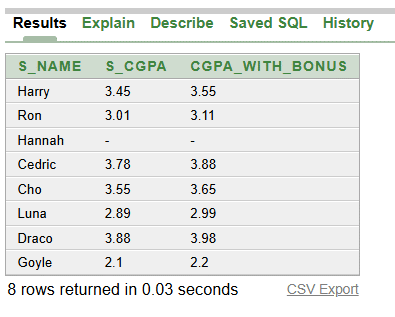
);



**1. Using ARITHMETIC Operator**

**Question:** Find the CGPA of each student after adding 0.10 bonus marks.

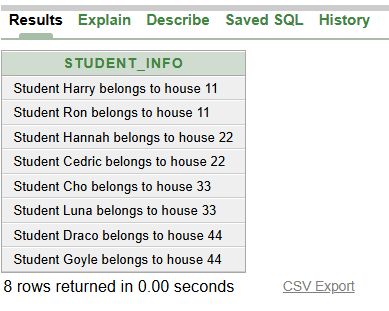
**Answer:** SELECT S\_Name, S\_CGPA, (S\_CGPA + 0.10) AS CGPA\_with\_Bonus FROM Student;



**2. Using CONCATENATION Operator**

**Question:** Display the full description like "Student Harry belongs to house 11".

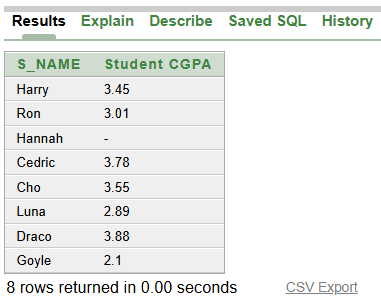
**Answer:** SELECT 'Student ' || S\_Name || ' belongs to house ' || H\_No AS Student\_Info FROM Student;



**3. Using COLUMN ALIAS**

**Question:** Display student names with CGPA, but rename CGPA as "Student CGPA".

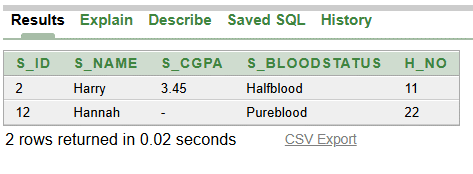
**Answer:** SELECT S\_Name, S\_CGPA AS "Student CGPA" FROM Student;



**4. Using LIKE Operator**

**Question:** Find all students whose names start with 'H'.

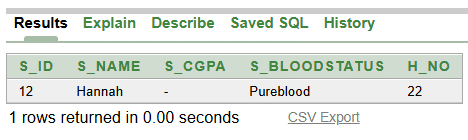
**Answer:** SELECT \* FROM Student WHERE S\_Name LIKE 'H%';



**5. Using IS NULL Operator**

**Question:** Find all students whose CGPA is not available (NULL).

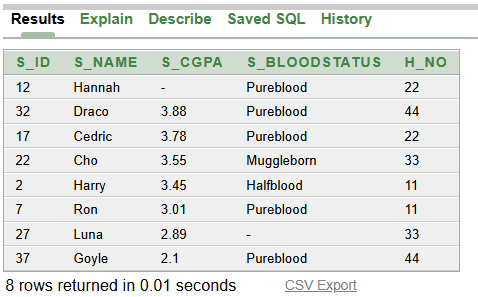
**Answer:** SELECT \* FROM Student WHERE S\_CGPA IS NULL;



**6. Using ORDER BY Clause**

**Question:** Display all students ordered by CGPA in descending order.

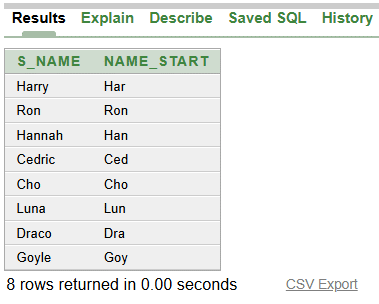
**Answer:** SELECT \* FROM Student ORDER BY S\_CGPA DESC;



**7. Using SUBSTR function**

**Question:** Display the first 3 characters of student names.

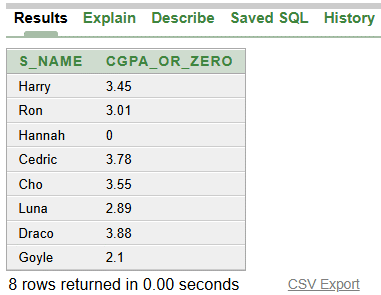
**Answer:** SELECT S\_Name, SUBSTR(S\_Name, 1, 3) AS Name\_Start FROM Student;



**8. Using NVL function**

**Question:** Display students' CGPA; if CGPA is NULL, show 0 instead.

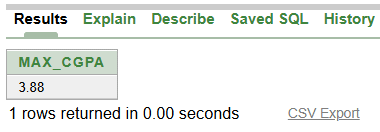
**Answer:** SELECT S\_Name, NVL(S\_CGPA, 0) AS CGPA\_or\_Zero FROM Student;



**9. Using MAX function**

**Question:** Find the maximum CGPA among all students.

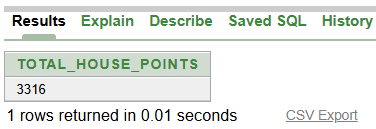
**Answer:** SELECT MAX(S\_CGPA) AS Max\_CGPAFROM Student;



**10. Using SUM function**

**Question:** Find the total points of all houses.

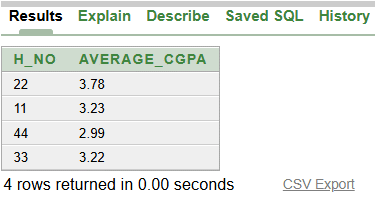
**Answer:** SELECT SUM(H\_Points) AS Total\_House\_Points FROM House;



**11. Using GROUP BY Clause**

**Question:** Find the average CGPA of students grouped by their House Number.

**Answer:** SELECT H\_No, AVG(S\_CGPA) AS Average\_CGPA FROM Student GROUP BY H\_No;



**12. Using HAVING Clause**

**Question:** Find the house numbers where the average CGPA is greater than 3.0.

**Answer:** SELECT H\_No, AVG(S\_CGPA) AS Average\_CGPA FROM Student GROUP BY H\_No HAVING AVG(S\_CGPA) > 3.0;

