**Remote Terminal Unit Database**

**Participants are instructed to create following tables with appropriate data types, primary key, foreign key, not null and check constraints.**

**Insert data as shown below into respective tables**

**Table - RTU**

|  |  |  |
| --- | --- | --- |
| **RTU\_ID** | **LOCATION** | **IP\_ADDRESS** |
| R101 | BENGALURU | 192.168.0.1 |
| R102 | MUMBAI | 192.168.0.2 |
| R103 | CHENNAI | 192.168.0.3 |
| R104 | NEW DELHI | 192.168.0.4 |
| R105 | KOLKATA | 192.168.0.5 |

**Table – SENSORS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SENSOR\_ID** | **RTU\_ID** | **STATUS** | **SENSOR\_TYPE** | **REQ\_VAL** | **CURR\_VAL** |
| S801 | R101 | ENABLED | PRESSURE | 34 | 33 |
| S802 | R101 | ENABLED | TEMPERATURE | 98 | 90 |
| S803 | R101 | DISABLED | TEMPERATURE | 96 |  |
| S804 | R102 | ENABLED | PRESSURE | 30 | 32 |
| S805 | R102 | ENABLED | PRESSURE | 36 | 32 |
| S806 | R102 | DISABLED | TEMPERATURE | 108 |  |
| S807 | R102 | ENABLED | BAROMETER | 750 | 660 |
| S808 | R103 | ENABLED | BAROMETER | 780 | 690 |
| S809 | R103 | ENABLED | ULTRASONIC | 510 | 450 |
| S810 | R104 | DISABLED | ULTRASONIC | 530 |  |
| S811 | R104 | ENABLED | PRESSURE | 60 | 50 |

**Table – USERS**

|  |  |  |
| --- | --- | --- |
| **USER\_ID** | **USER\_NAME** | **ROLE** |
| U501 | PRAKASH | MONITOR |
| U502 | PARTHIV | ADMINISTER |
| U503 | SMITHA | ADMINISTER |

**Table – CONTROLLER**

|  |  |
| --- | --- |
| **USER\_ID** | **SENSOR\_ID** |
| U501 | S801 |
| U501 | S804 |
| U501 | S806 |
| U502 | S802 |
| U502 | S804 |
| U503 | S805 |
| U503 | S806 |
| U501 | S809 |
| U502 | S810 |
| U503 | S811 |
| U501 | S807 |
| U503 | S808 |
| U502 | S801 |
| U501 | S808 |

**Write following queries based on above tables**

Simple Queries

1. Fetch details of all the sensors which are disabled

**SELECT SENSOR\_ID,SENSOR\_TYPE FROM SENSORS WHERE STATUS ='DISABLED';**

1. List sensors which are having required value less than current value

**SELECT SENSOR\_ID,SENSOR\_TYPE FROM SENSORS WHERE REQ\_VAL<CURR\_VAL;**

1. List sensors which are of type TEMPERATURE and required value greater than 96. Sort on required value field.

**SELECT SENSOR\_ID,SENSOR\_TYPE,REQ\_VAL FROM SENSORS WHERE SENSOR\_TYPE ='TEMPERATURE' AND REQ\_VAL>96 ORDER BY REQ\_VAL;**

1. List sensors which has no current values

**SELECT SENSOR\_ID,SENSOR\_TYPE,CURR\_VAL FROM SENSORS WHERE CURR\_VAL IS NULL;**

1. List all sensors which are of type TEMPERATURE & PRESSURE.

**SELECT SENSOR\_ID,SENSOR\_TYPE FROM SENSORS WHERE SENSOR\_TYPE IN ('TEMPERATURE','PRESSURE');**

1. List sensors with rtu\_id 102 and sensor type pressure and required value greater than 30.

**SELECT SENSOR\_ID,RTU\_ID,SENSOR\_TYPE,REQ\_VAL FROM SENSORS WHERE SENSOR\_TYPE = 'PRESSURE' AND RTU\_ID='R102' AND REQ\_VAL>30;**

**Single Row Functions**

1. Write a query which will display username in upper case, role first letter capitalized and all other letters lower case and length of the username.

**SELECT UPPER(USER\_NAME),INITCAP(ROLE),LENGTH(USER\_NAME) FROM USERS;**

1. Display percentage of difference between req\_value and curr\_value as a whole number.

**SELECT SENSOR\_ID,REQ\_VAL,CURR\_VAL, ROUND((REQ\_VAL - CURR\_VAL)\*100,0) FROM SENSORS;**

**Group Functions**

1. List how many sensors are present of type PRESSURE.

**SELECT COUNT(SENSOR\_TYPE) FROM SENSORS WHERE SENSOR\_TYPE='PRESSURE' GROUP BY SENSOR\_TYPE;**

1. Display number of sensors enabled and disabled. Sort on number of sensors

**SELECT STATUS,COUNT(STATUS) FROM SENSORS WHERE STATUS='ENABLED' OR STATUS='DISABLED' GROUP BY STATUS;**

1. List how many sensors are present in each sensor type.

**SELECT SENSOR\_TYPE,COUNT(SENSOR\_TYPE) FROM SENSORS WHERE SENSOR\_TYPE IN ('PRESSURE','TEMPERATURE','BAROMETER','ULTRASONIC') GROUP BY SENSOR\_TYPE;**

1. List number of sensors under each RTU\_ID.

**SELECT RTU\_ID,COUNT(SENSOR\_ID) FROM SENSORS GROUP BY RTU\_ID;**

1. Display how many sensors are controlled by each user. Exclude users who are controlling less than 3 sensors and also restrict users with id “U503”. Sort on user\_id.

**SELECT USER\_ID,COUNT(SENSOR\_ID) FROM CONTROLLER GROUP BY USER\_ID HAVING NOT USER\_ID='U503' AND COUNT(SENSOR\_ID)>3 ORDER BY USER\_ID**

**SUB-QUERIES**

1. Display sensor details located in CHENNAI

**SELECT SENSOR\_ID,RTU\_ID,SENSOR\_TYPE,STATUS,REQ\_VAL,CURR\_VAL FROM SENSORS WHERE RTU\_ID IN(SELECT RTU\_ID FROM RTU WHERE LOCATION ='CHENNAI')**

1. Display user details controlling sensors in MUMBAI location

**SELECT USER\_ID,USER\_NAME,ROLE FROM USERS WHERE USER\_ID IN (SELECT USER\_ID FROM CONTROLLER WHERE SENSOR\_ID IN (SELECT SENSOR\_ID FROM SENSORS WHERE RTU\_ID IN (SELECT RTU\_ID FROM RTU WHERE LOCATION ='MUMBAI')))**

1. Display sensor details controlled by SMITHA

**SELECT RTU\_ID,SENSOR\_ID,SENSOR\_TYPE,STATUS,REQ\_VAL,CURR\_VAL FROM SENSORS WHERE SENSOR\_ID IN (SELECT SENSOR\_ID FROM CONTROLLER WHERE USER\_ID IN(SELECT USER\_ID FROM USERS WHERE USER\_NAME='SMITHA'))**

1. Display user details who are controlling sensors with id between S803 and S808

**SELECT USER\_ID,USER\_NAME,ROLE FROM USERS WHERE USER\_ID IN (SELECT USER\_ID FROM CONTROLLER WHERE SENSOR\_ID BETWEEN 'S803' AND 'S808')**

1. Display user details that are controlling most number of sensors.

**SELECT \* FROM USERS WHERE USER\_ID =( SELECT USER\_ID FROM CONTROLLER GROUP BY USER\_ID HAVING COUNT(SENSOR\_ID)=(SELECT (MAX(COUNT(SENSOR\_ID))) FROM CONTROLLER GROUP BY USER\_ID))**

1. Display RTU details which has most number of sensors.

**SELECT \* FROM RTU WHERE RTU\_ID =( SELECT RTU\_ID FROM SENSORS GROUP BY RTU\_ID HAVING COUNT(SENSOR\_ID)=(SELECT (MAX(COUNT(SENSOR\_ID))) FROM SENSORS GROUP BY RTU\_ID))**