# A3-20171-DBS301

# DUE week 12 Friday before MIDNIGHT

**Extended deadline:** for those who need it ……

THURSDAY Week 13 April 13, 2017 – no penalty if you take the extension

After midnight April 13 you will be really unlucky -- the value of the assignment is zero

➔ meaning you lost 7%

**DANGER: There is a test in week 13. Doing the assignment late or getting a lot of help from someone without being able to understand it yourself will cause problems on the test.**

1 Change the name of this file to a3-YourEmailid ←← must be done

Mail it back as a WORD attachment and ➔➔ NOT a PDF or image or ONEDRIVE or ….. etc

2 In the subject line of the email put the file name from #1 above.

3 You can do this in groups, but remember if you don't do the work and the other members do it, you will likely fail the test and quite possibly the exam.

4 As a group member you are saying that you participated fairly as part of a group of and that you understand everything that was submitted.

Good luck

List all members if any in your group.

1 CREATE TABLES Question

**DIVISION**

|  |  |  |
| --- | --- | --- |
| **Column Name** | DIVISION\_ID | DIVISION\_NAME |
| **Key Type** | **PK** |  |
| **Null/Unique** |  | **NN, U** |
| **FK Table** |  |  |
| **FK Column** |  |  |
| **Validation** |  |  |
| **Datatype** | **NUMBER** | **VARCHAR** |
| **Length** | **3** | **25** |
| **Sample data** |  |  |
|  | **10** | **East Coast** |
|  | **20** | **Quebec** |
|  | **30** | **Ontario** |

**WAREHOUSE**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | WAREHOUSE\_ID | CITY | RATING | FOUND\_DATE | DIVISION\_ID |
| **Key Type** | **PK** |  | **CK** |  | **FK** |
| **Null/Unique** |  | **NN, U** |  | **NN** | **NN** |
| **FK Table** |  |  |  |  | **DIVISION** |
| **FK Column** |  |  |  |  | **DIVISION\_ID** |
| **Validation** |  |  | **A, B, C, D** |  |  |
| **Datatype** | **NUMBER** | **VARCHAR** | **CHAR** | **DATE** | **NUMBER** |
| **Length** | **3** | **15** | **1** |  | **3** |
| **Sample Data** | **1** | **Montreal** | **A** | **Current date** | **10** |
|  | **7** | **Fredericton** | **B** | **Current date** | **10** |
|  | **10** | **Toronto** | **A** | **Current date** | **30** |

**SECTION**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column Name** | WAREHOUSE\_ID | SECTION\_ID | DESCRIPTION | CAPACITY |
| **Key Type** | **PK, FK** | **PK** |  |  |
| **Null/Unique** |  |  | **NN** |  |
| **FK Table** | **WAREHOUSE** |  |  |  |
| **FK Column** | **WAREHOUSE\_ID** |  |  |  |
| **Datatype** | **NUMBER** | **NUMBER** | **VARCHAR** | **NUMBER**  Sample data to insert |
| **Length** | **3** | **2** | **50** | **8** |
| **Sample data** | **1** | **1** | **Whse 1 Floor 1** | **2000** |
|  | **1** | **2** | **Whse 1 Floor 2** | **500** |
|  | **7** | **1** | **Whse 7 Floor 1** | **15000** |

1 (10 marks) Write the required SQL statements to create tables WAREHOUSE, DIVISION and SECTION.

Follow these general rules in the process:

A. Create all CHECK (incl. NOT NULL) and UNIQUE as column level constraints

Constraint names needed for CHECK constraints. The other constraints (NN and UK) do not need a name.

B. Create all PK and FK constraints at the table level and give them proper names.

PUT ANSWERS starting here

**CREATE TABLE DIVISION(**

**DIVISION\_ID NUMBER(3),**

**DIVISION\_NAME VARCHAR(25) NOT NULL UNIQUE,**

**CONSTRAINT PK\_DIVISION**

**PRIMARY KEY (DIVISION\_ID)**

**);**

**INSERT INTO DIVISION VALUES(10, 'East Coast');**

**INSERT INTO DIVISION VALUES(20, 'Quebec');**

**INSERT INTO DIVISION VALUES(30, 'Ontario');**

**CREATE TABLE WAREHOUSE(**

**WAREHOUSE\_ID NUMBER(3) ,**

**CITY VARCHAR(15) NOT NULL UNIQUE,**

**RATING CHAR(1),**

**FOUND\_DATE DATE NOT NULL,**

**DIVISION\_ID NUMBER(3) NOT NULL,**

**CONSTRAINT PK\_WAREHOUSE**

**PRIMARY KEY(WAREHOUSE\_ID),**

**CONSTRAINT FK\_DIVISION**

**FOREIGN KEY(DIVISION\_ID)**

**REFERENCES DIVISION(DIVISION\_ID),**

**CONSTRAINT CK\_RATING**

**CHECK (RATING IN ('A','B','C','D'))**

**);**

**INSERT INTO WAREHOUSE VALUES(1, 'Montreal', 'A', SYSDATE, 10);**

**INSERT INTO WAREHOUSE VALUES(7, 'Fredericton', 'B', SYSDATE, 10);**

**INSERT INTO WAREHOUSE VALUES(10, 'Toronto', 'A', SYSDATE, 30);**

**CREATE TABLE SECTION(**

**WAREHOUSE\_ID NUMBER(3),**

**SECTION\_ID NUMBER(2),**

**DESCRIPTION VARCHAR(50) NOT NULL,**

**CAPACITY NUMBER(8),**

**CONSTRAINT PK\_SECTION**

**PRIMARY KEY (WAREHOUSE\_ID,SECTION\_ID),**

**CONSTRAINT FK\_WAREHOUSE**

**FOREIGN KEY(WAREHOUSE\_ID)**

**REFERENCES WAREHOUSE(WAREHOUSE\_ID)**

**);**

**INSERT INTO SECTION VALUES (1,1 ,'Whse 1 Floor 1', 2000);**

**INSERT INTO SECTION VALUES (1,2 ,'Whse 1 Floor 2', 500);**

**INSERT INTO SECTION VALUES (7,1 ,'Whse 7 Floor 1', 15000);**

**/\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*/**

**Table DIVISION created.**

**1 row inserted.**

**1 row inserted.**

**1 row inserted.**

**Table WAREHOUSE created.**

**1 row inserted.**

**1 row inserted.**

**1 row inserted.**

**Table SECTION created.**

**1 row inserted.**

**1 row inserted.**

**1 row inserted.**

**2** (3 marks) After creating all tables add column MGR\_ID to table SECTION as a FK column, that is related to the PK column EMPLOYEE\_ID in table EMPLOYEE

**ALTER TABLE SECTION**

**ADD MGR\_ID NUMBER(6)**

**CONSTRAINT SECTION\_MGR\_ID\_FK**

**REFERENCES EMPLOYEES(EMPLOYEE\_ID);**

**/\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*/**

**Table SECTION altered.**

3 (3 marks) Modify the CHECK constraint on column RATING in table WAREHOUSE, so that it also may accept a new value F.

**ALTER TABLE WAREHOUSE**

**DROP CONSTRAINT CK\_RATING;**

**ALTER TABLE WAREHOUSE**

**ADD CONSTRAINT CK\_RATING**

**CHECK (RATING IN ('A','B','C','D','F'));**

**/\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*/**

**Table WAREHOUSE altered.**

**Table WAREHOUSE altered.**

4 (3 marks) Create a new **Sequence** called **Whse\_id\_seq** that will generate unique numbers for PK values in table WAREHOUSE, so that the numbers start at 420 with the step of 5 and upper limit is 700 and will have NO values stored in the memory.

**CREATE SEQUENCE WHSE\_ID\_SEQ**

**START WITH 420**

**INCREMENT BY 5**

**MAXVALUE 700**

**NOCACHE**

**NOCYCLE;**

**/\*CYCLE could be use here, in case someone removes a row, the ids will update based on the Sequence Limits\*/**

**/\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*/**

**Sequence WHSE\_ID\_SEQ created.**

5 (3 marks) Add new row to table WAREHOUSE by using this sequence for a city in Atlanta with unknown rating **and division 30.** You will assume today’s date as a foundation date. The date is to be entered automatically, meaning you cannot enter a specific date.

**/\*After Removing NOT NULL from table creation NULL will be declare by default\*/**

**INSERT INTO WAREHOUSE (WAREHOUSE\_ID,CITY,FOUND\_DATE,DIVISION\_ID)**

**VALUES(WHSE\_ID\_SEQ.nextval,'Atlanta',SYSDATE,30);**

**/\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*/**

**1 row inserted.**

6 (5 marks) Create table CITIES **from table LOCATIONS,** but only for location numbers less than 2000 (do NOT create this table from scratch). → You will have 5 to 18 rows

**CREATE TABLE CITIES AS**

**(SELECT \* FROM LOCATIONS**

**WHERE LOCATION\_ID < 2000);**

**/\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*/**

**Table CITIES created.**

7 (2 marks) Issue command to show the structure of the table CITIES

**DESCRIBE CITIES;**

**/\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*/**

**Table CITIES created.**

**Name Null Type**

**-------------- -------- ------------**

**LOCATION\_ID NUMBER(4)**

**STREET\_ADDRESS VARCHAR2(40)**

**POSTAL\_CODE VARCHAR2(12)**

**CITY NOT NULL VARCHAR2(30)**

**STATE\_PROVINCE VARCHAR2(25)**

**COUNTRY\_ID CHAR(2)**

8 (1 mark) Issue the SELECT command on cities and show result here.

**SELECT \* FROM CITIES;**

**/\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*/**

**LOCATION\_ID STREET\_ADDRESS POSTAL\_CODE CITY STATE\_PROVINCE CO**

**----------- -------------------------- ------------ ---------------------- ------------------------- --**

**1000 1297 Via Cola di Rie 00989 Roma IT**

**1100 93091 Calle della Testa 10934 Venice IT**

**1200 2017 Shinjuku-ku 1689 Tokyo Tokyo Prefecture JP**

**1300 9450 Kamiya-cho 6823 Hiroshima JP**

**1400 2014 Jabberwocky Rd 26192 Southlake Texas US**

**1500 2011 Interiors Blvd 99236 South San Francisco California US**

**1600 2007 Zagora St 50090 South Brunswick New Jersey US**

**1700 2004 Charade Rd 98199 Seattle Washington US**

**1800 147 Spadina Ave M5V 2L7 Toronto Ontario CA**

**1900 6092 Boxwood St YSW 9T2 Whitehorse Yukon CA**

**10 rows selected**

9 (5 marks) Create a View called **WhsSec\_Man\_vu** that will display for each Warehouse\_id and Section\_id, the City, Division and manager’s Last\_name.

Alias for Last\_name should be LName and for Division should be Group.

**CREATE VIEW WHSSEC\_MAN\_VU AS(**

**SELECT WAREHOUSE\_ID,**

**SECTION\_ID,**

**CITY,**

**DIVISION\_NAME AS "Group",**

**LAST\_NAME AS "LName"**

**FROM WAREHOUSE JOIN SECTION USING(WAREHOUSE\_ID)**

**JOIN DIVISION USING(DIVISION\_ID)**

**JOIN EMPLOYEES**

**ON MGR\_ID = EMPLOYEE\_ID**

**);**

**/\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*/**

**WAREHOUSE\_ID SECTION\_ID CITY GROUP LName**

**------------- ---------- --------- ------- ---------------**

**No rows selected**

10 (1 mark) What is the SELECT command to issue if in 2 months I want to test if a view was actually was created

**SELECT OBJECT\_NAME, OBJECT\_ID, OBJECT\_TYPE, CREATED**

**FROM USER\_OBJECTS**

**WHERE OBJECT\_TYPE = 'VIEW' AND CREATED > ADD\_MONTHS(SYSDATE, -2);**

**/\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*/**

**OBJECT\_NAME OBJECT\_ID OBJECT\_TYPE CREATED**

**--------------- ----------- ------------------- --------------**

**WHSSEC\_MAN\_VU 22428 VIEW 29-MAR-17**

11 (1 mark) If you want to modify the view what is the first line of the command

**CREATE OR REPLACE VIEW WHSSEC\_MAN\_VU AS (**

12 Issue a SET operator to show the rows that were in LOCATIONS but not in CITIES

**SELECT \* FROM LOCATIONS**

**MINUS**

**SELECT \* FROM CITIES;**

**/\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*/**

**LOCATION\_ID STREET\_ADDRESS POSTAL\_CODE CITY STATE\_PROVINCE CO**

**----------- ---------------------------------------- ------------ ------------------------------ --------------**

**2000 40-5-12 Laogianggen 190518 Beijing CN**

**2100 1298 Vileparle (E) 490231 Bombay Maharashtra IN**

**2200 12-98 Victoria Street 2901 Sydney New South Wales AU**

**2300 198 Clementi North 540198 Singapore SG**

**2400 8204 Arthur St London UK**

**2500 Magdalen Centre, The Oxford Science Park OX9 9ZB Oxford Oxford UK**

**2600 9702 Chester Road 09629850293 Stretford Manchester UK**

**2700 Schwanthalerstr. 7031 80925 Munich Bavaria DE**

**2800 Rua Frei Caneca 1360 01307-002 Sao Paulo Sao Paulo BR**

**2900 20 Rue des Corps-Saints 1730 Geneva Geneve CH**

**3000 Murtenstrasse 921 3095 Bern BE CH**

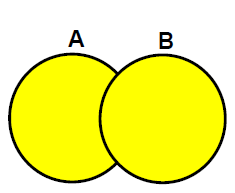
**3100 Pieter Breughelstraat 837 3029SK Utrecht Utrecht NL**

**3200 Mariano Escobedo 9991 11932 Mexico City Distrito Federal, MX**

**13 rows selected**

Using the following diagram as a hint and not a perfect representation.

Answer 13, 14, 15 and 16



13 All the rows in A and all the rows in B with no duplicates is the set operator called

[**UNION**]

14 All the rows in A and all the rows in B with duplicates [**UNION ALL**]

15 The rows in common to BOTH A and B tables [**INTERSECT**]

16 Rows that are in A but not in B would use the word [**MINUS**]