

## Homework 5

Adhiraj Budukh

### Problem 1

1.

CREATE TABLE Product

(

manufacturer CHAR(10),

model CHAR(10) PRIMARY KEY,

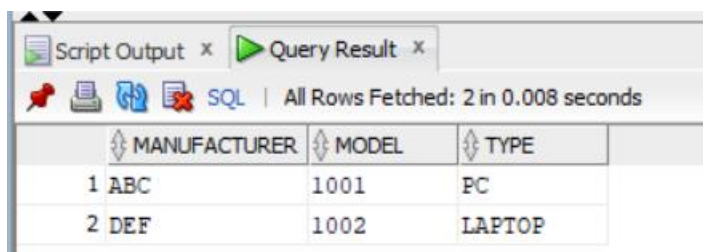
type CHAR(10) CHECK (TYPE = 'LAPTOP' OR TYPE ='PC'));

```
CREATE TABLE Product
(
  manufacturer CHAR(10),
  model CHAR(10) PRIMARY KEY,
  type CHAR(10) CHECK (TYPE = 'LAPTOP' OR TYPE ='PC'));

---Inserted Data
INSERT INTO Product VALUES ('ABC', '1001', 'PC');
INSERT INTO Product VALUES ('DEF', '1002', 'LAPTOP');
INSERT INTO Product VALUES ('DEF', '1003', 'Chromebook');

SELECT * FROM Product;
```

Only 2 rows inserted



The screenshot shows a database query result window with two tabs: 'Script Output' and 'Query Result'. The 'Query Result' tab is active, displaying a table with three columns: MANUFACTURER, MODEL, and TYPE. The table contains two rows of data. The first row has MANUFACTURER 'ABC', MODEL '1001', and TYPE 'PC'. The second row has MANUFACTURER 'DEF', MODEL '1002', and TYPE 'LAPTOP'. The third row, which would have contained 'DEF', '1003', and 'Chromebook', is not present in the results, indicating a constraint violation.

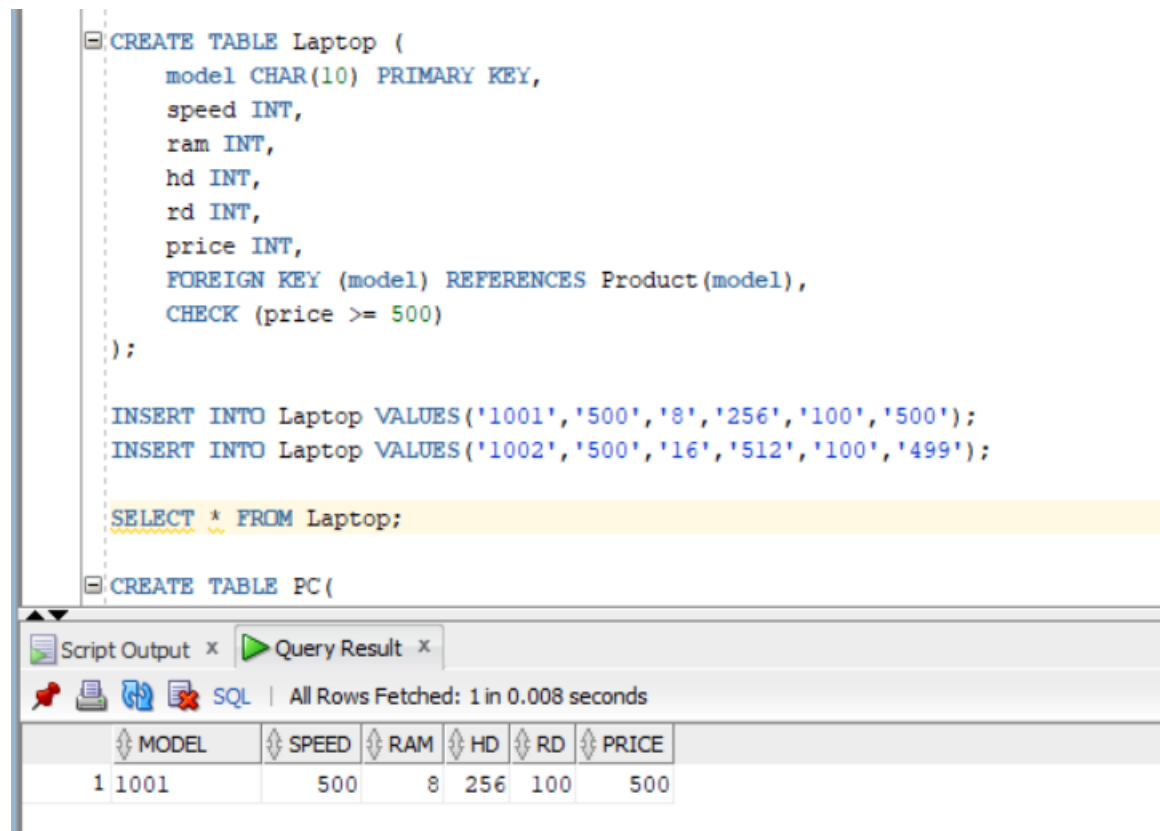
	MANUFACTURER	MODEL	TYPE
1	ABC	1001	PC
2	DEF	1002	LAPTOP

Violation occurred for the Chromebook type hence Constraint definition satisfied.

```
Error starting at line : 10 in command -
INSERT INTO Product VALUES ('DEF','1003','Chromebook')
Error report -
ORA-02290: check constraint (ABUDUKH.SYS_C001991691) violated
```

2.

```
CREATE TABLE Laptop (
    model CHAR(10) PRIMARY KEY,
    speed INT,
    ram INT,
    hd INT,
    rd INT,
    price INT,
    FOREIGN KEY (model) REFERENCES Product(model),
    CHECK (price >= 500)
);
```



```
CREATE TABLE Laptop (
    model CHAR(10) PRIMARY KEY,
    speed INT,
    ram INT,
    hd INT,
    rd INT,
    price INT,
    FOREIGN KEY (model) REFERENCES Product(model),
    CHECK (price >= 500)
);

INSERT INTO Laptop VALUES ('1001','500','8','256','100','500');
INSERT INTO Laptop VALUES ('1002','500','16','512','100','499');

SELECT * FROM Laptop;
```

	MODEL	SPEED	RAM	HD	RD	PRICE
1	1001	500	8	256	100	500

Only one row was inserted and second was rejected because of price less than 500; Hence constraint definition satisfied.

```
Error starting at line : 26 in command -  
INSERT INTO Laptop VALUES('1002','500','16','512','100','499')  
Error report -  
ORA-02290: check constraint (ABUDUKH.SYS_C001991747) violated
```

3. To add the constraint that a laptop with a larger model number must also have a higher price than one with a lower model number, It is not possible to implement this constraint using a DDL statement, we must use triggers.

Without triggers, we can't enforce **this** specific constraint in SQL DDL.

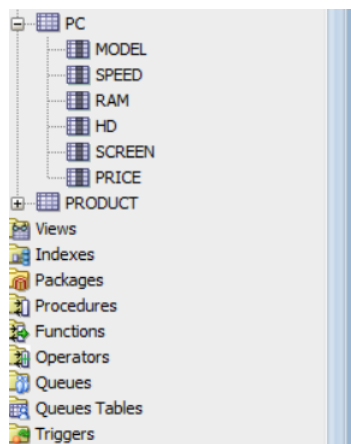
4.

```
CREATE TABLE PC(  
    model CHAR(10) PRIMARY KEY,  
    speed INT,  
    ram INT,  
    hd INT,  
    screen Number(4,2),  
    price INT,  
    CONSTRAINT model_no FOREIGN KEY (model) REFERENCES Product(model)  
);
```

PC table created.

```
CREATE TABLE Laptop( model CHAR(10) PRIMARY KEY, speed INT, ram INT, hd INT, screen  
number(4,2), price INT, CONSTRAINT model_no FOREIGN KEY (model) references Product(model) );
```

Laptop Table was created.



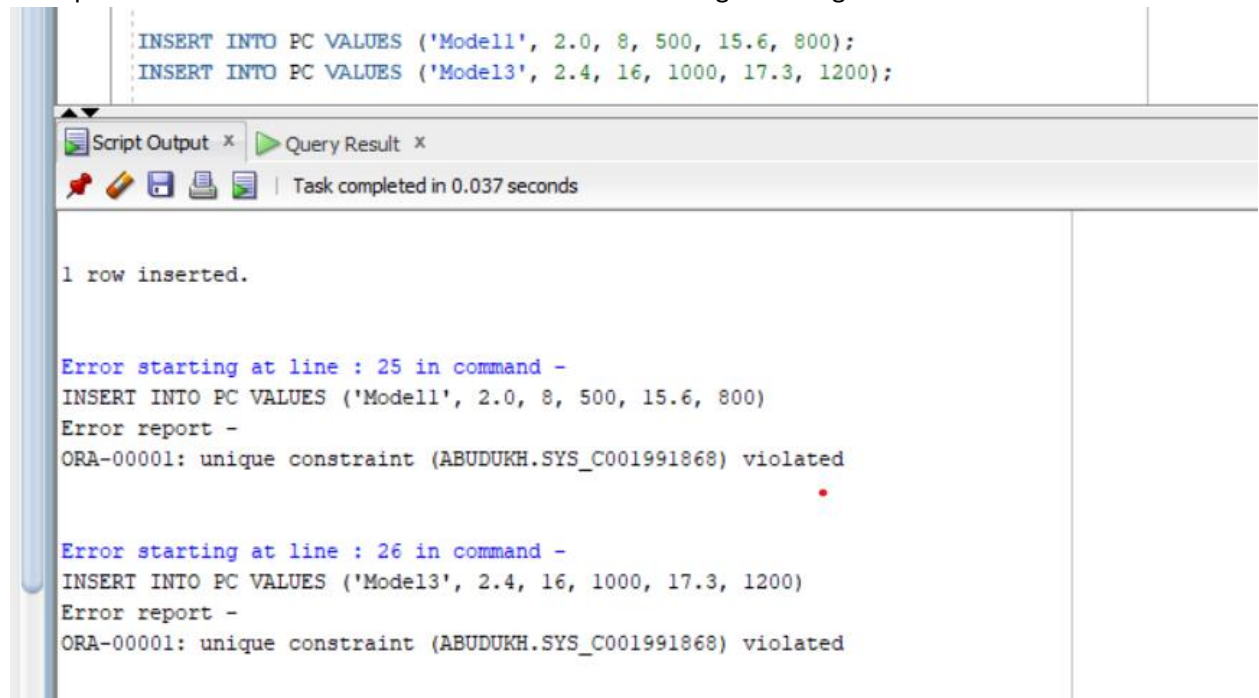
```
);  
  
INSERT INTO Laptop VALUES('1001','500','8','256','100','500');  
INSERT INTO Laptop VALUES('1002','500','16','512','100','499');  
  
SELECT * FROM Laptop;  
  
CREATE TABLE PC(  
    model CHAR(10) PRIMARY KEY,  
    speed INT,  
    ram INT,  
    hd INT,  
    screen Number(4,2),  
    price INT,  
    CONSTRAINT model_no FOREIGN KEY (model) REFERENCES Product(model)  
);
```

```
--1.4  
INSERT INTO Product VALUES('M1', 'Model11', 'PC');  
INSERT INTO Product VALUES('M2', 'Model12', 'LAPTOP');  
INSERT INTO Product VALUES('M3', 'Model13', 'PC');  
INSERT INTO Product VALUES('M4', 'Model14', 'LAPTOP');
```

Script Output x Query Result x  
SQL | All Rows Fetched: 6 in 0.021 seconds

	MANUFACTURER	MODEL	TYPE
1	ABC	1001	PC
2	DEF	1002	LAPTOP
3	M1	Model11	PC
4	M3	Model13	PC
5	M2	Model12	LAPTOP
6	M4	Model14	LAPTOP

Example of violation -Constrain violated when tried inserting values again for PC:-



```
INSERT INTO PC VALUES ('Model11', 2.0, 8, 500, 15.6, 800);
INSERT INTO PC VALUES ('Model13', 2.4, 16, 1000, 17.3, 1200);
```

Script Output x Query Result x

Task completed in 0.037 seconds

1 row inserted.

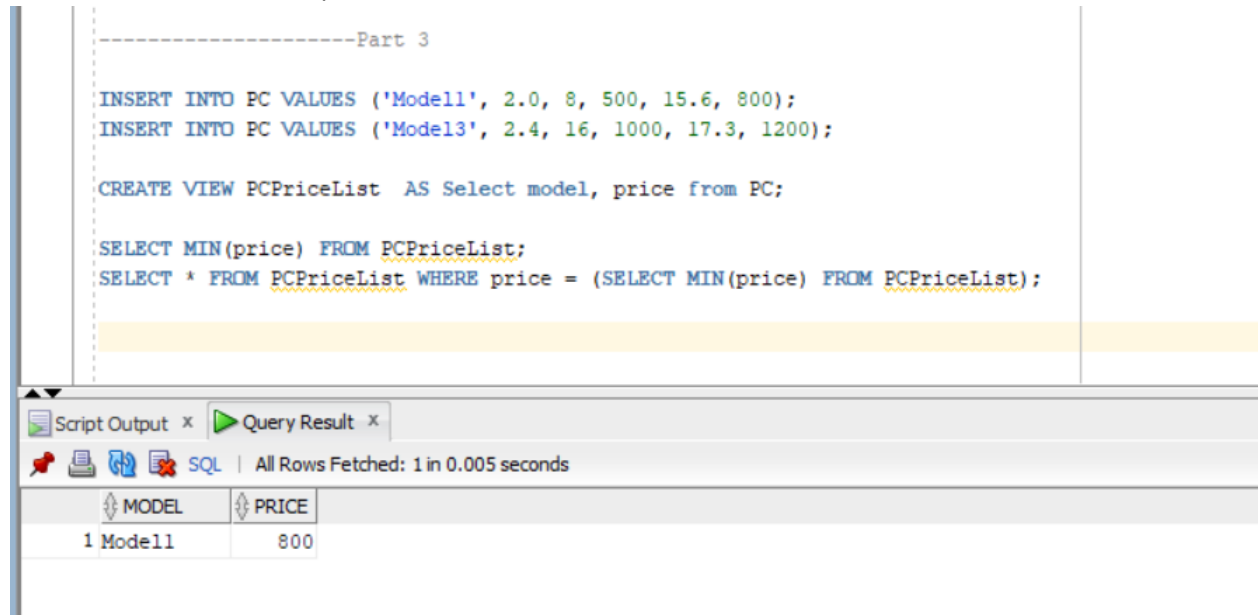
Error starting at line : 25 in command -  
INSERT INTO PC VALUES ('Model11', 2.0, 8, 500, 15.6, 800)  
Error report -  
ORA-00001: unique constraint (ABUDUKH.SYS\_C001991868) violated

Error starting at line : 26 in command -  
INSERT INTO PC VALUES ('Model13', 2.4, 16, 1000, 17.3, 1200)  
Error report -  
ORA-00001: unique constraint (ABUDUKH.SYS\_C001991868) violated

5. To limit the number of manufacturers to at most 5, we can create a CHECK constraint on the Product table, But for the exact number of distinct manufacturers can't be enforced using a simple CHECK constraint alone i.e. without using Trigger.

### Problem 3:

1. `SELECT * FROM PCPriceList WHERE price = (SELECT MIN(price) FROM PCPriceList);`  
We can see minimum price 800 was selected from the data.



The screenshot shows a SQL IDE interface. The top pane displays a script with the following SQL statements:

```
-----Part 3  
  
INSERT INTO PC VALUES ('Model1', 2.0, 8, 500, 15.6, 800);  
INSERT INTO PC VALUES ('Model13', 2.4, 16, 1000, 17.3, 1200);  
  
CREATE VIEW PCPriceList AS Select model, price from PC;  
  
SELECT MIN(price) FROM PCPriceList;  
SELECT * FROM PCPriceList WHERE price = (SELECT MIN(price) FROM PCPriceList);
```

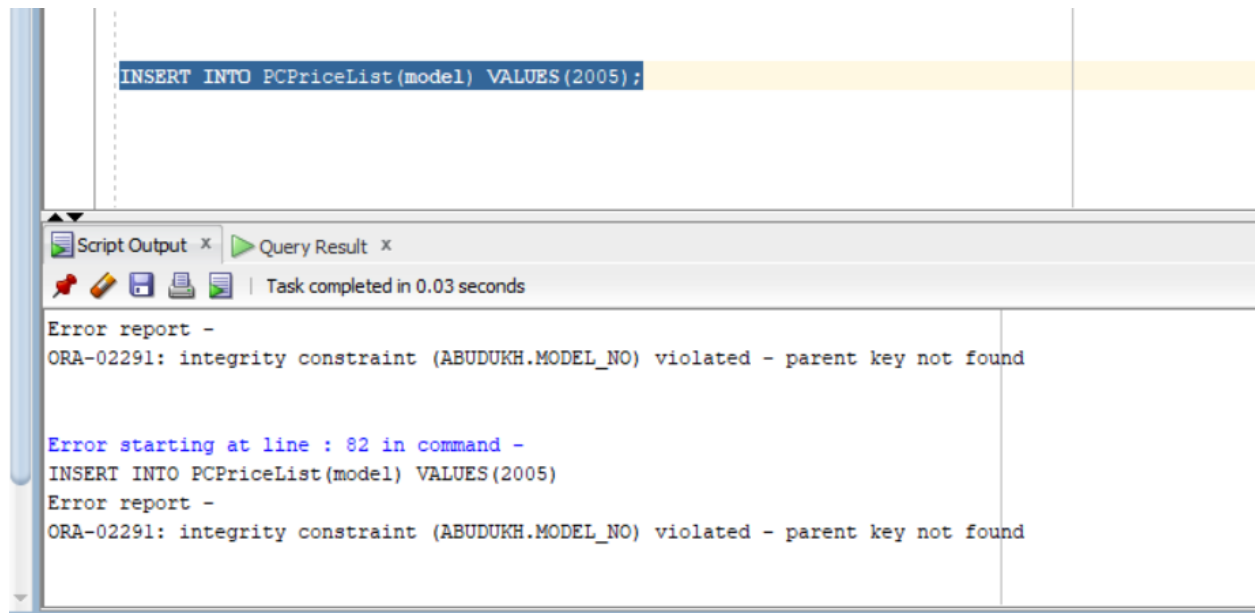
The bottom pane shows the 'Query Result' tab with the following data:

MODEL	PRICE
1 Model1	800

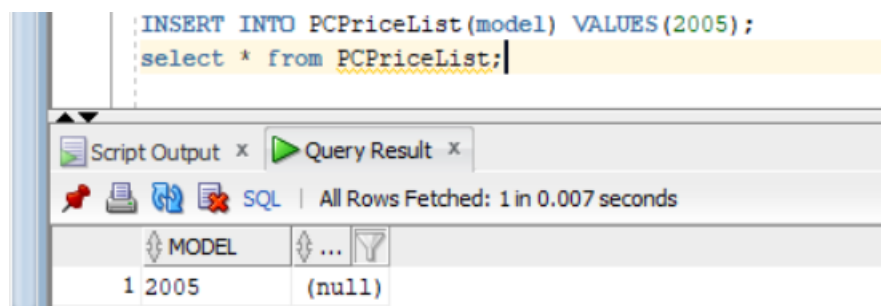
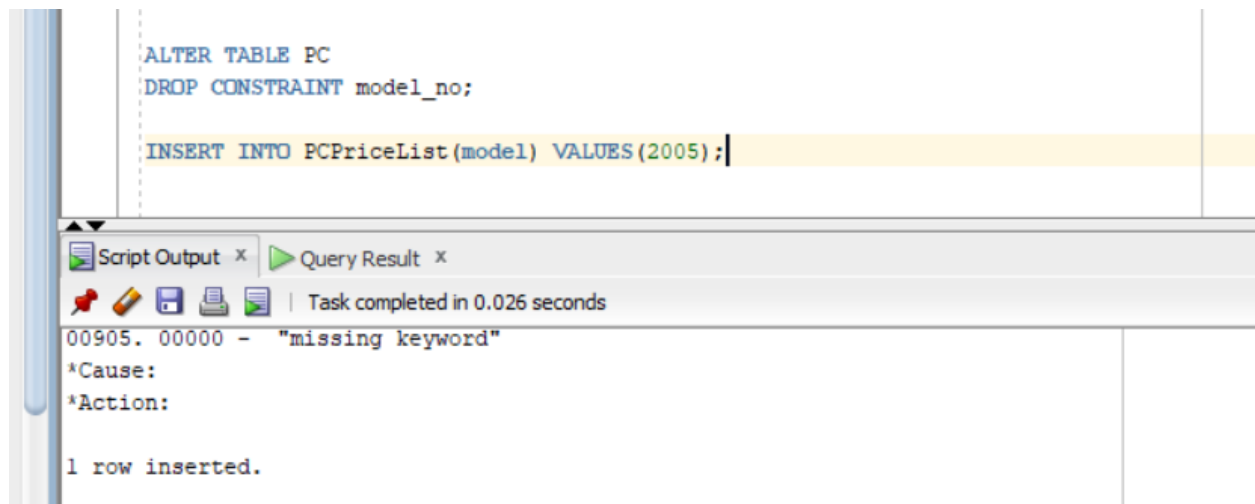
This view can be deleted and following query can delete tuple where model is Model1 –  
`DELETE FROM PCPriceList WHERE model = 'Model1';`

2. The `INSERT INTO PCPriceList(model) VALUES (2005)` statement provided is trying to insert a single value into the model column of the PCPriceList view. This operation is valid if the following conditions are met:

When NO? – If Foreign Key Constraints are declared on model attribute.



When YES- when Foreign Key are not-defined/doesnot exist. And it will insert model 2005 as its primary key and set price to Null.



3. INSERT INTO PCPriceList (price) VALUES (1700)

```
INSERT INTO PCPriceList (price) VALUES (1700);
```

Script Output x Query Result x

Task completed in 0.032 seconds

Error starting at line : 90 in command -  
INSERT INTO PCPriceList (price) VALUES (1700)  
Error report -  
ORA-01400: cannot insert NULL into ("ABUDUKH"."PC"."MODEL")

As a result, the insert would show an error since it would enter a null value in the model field and 1700 in the price field. However, model is the primary key of PC, and it cannot contain a null value.

4. CREATE VIEW extendedPC AS Select manufacturer, Product.model, speed, ram, hd, price, type FROM Product, PC WHERE Product.model = PC.model;

select \* from extendedpc;

```
CREATE VIEW extendedPC AS Select manufacturer, Product.model, speed,
ram, hd, price, type FROM Product, PC WHERE Product.model = PC.model;

select * from extendedpc;
```

Script Output x Query Result x

All Rows Fetched: 6 in 0.009 seconds

	MANUFACTURER	MODEL	SPEED	RAM	HD	PRICE	TYPE
1	M1	Model11	2	8	500	800	PC
2	M3	Model13	2	16	1000	1200	PC
3	1	2006	3	16	500	1300	PC
4	2	2007	3	16	1000	1400	PC
5	3	2008	3	8	500	1500	PC
6	4	2009	3	8	1000	1600	PC



<pre>select * from extendedpc;  DELETE FROM extendedPC WHERE MODEL ='2009';</pre>							
<div> <div>Script Output x</div> <div>Query Result x</div> </div> <div> </div> <div>SQL   All Rows Fetched: 5 in 0.01 seconds</div>							
	MANUFACTURER	MODEL	SPEED	RAM	HD	PRICE	TYPE
1	M1	Model1	2	8	500	800	PC
2	M3	Model3	2	16	1000	1200	PC
3	1	2006	3	16	500	1300	PC
4	2	2007	3	16	1000	1400	PC
5	3	2008	3	8	500	1500	PC

We can delete tuple from this View.

For ex. By this query

DELETE FROM extendedPC WHERE MODEL ='2009' we deleted the 6<sup>th</sup> row.

## Problem 2:

1.

```
CREATE Trigger model_pc
Before Insert Or Update OF model On PC
For EACH Row
Declare
    model_no number;
Begin
    Select Count(*)into model_no from Laptop where model = :new.model;
    IF model_no>0 Then RAISE_APPLICATION_ERROR(-20004,'The value already exists in
    Laptop');
End IF;
End;
/
```

```
CREATE OR REPLACE TRIGGER model_laptop
BEFORE INSERT OR UPDATE OF model ON Laptop
FOR EACH ROW
DECLARE
    model_no NUMBER;
BEGIN
    SELECT COUNT(*) INTO model_no FROM PC WHERE model = :new.model;
    IF model_no > 0 THEN
        RAISE_APPLICATION_ERROR(-20004, 'The value already exists in PC');
    END IF;
END;
/
```

```
-----2.1

CREATE Trigger model_pc
Before Insert Or Update OF model On PC
For EACH Row
  Declare
    model_no number;
  Begin
    Select Count(*) into model_no from Laptop where model = :new.model;
    IF model_no>0 Then RAISE_APPLICATION_ERROR(-20004,'The value already exists in Laptop');
  End IF;
End;
/

CREATE OR REPLACE TRIGGER model_laptop
BEFORE INSERT OR UPDATE OF model ON Laptop
FOR EACH ROW
DECLARE
  model_no NUMBER;
BEGIN
  SELECT COUNT(*) INTO model_no FROM PC WHERE model = :new.model;
  IF model_no > 0 THEN
    RAISE_APPLICATION_ERROR(-20004, 'The value already exists in PC');
  END IF;
END;
```

Script Output x

Task completed in 0.318 seconds

Trigger MODEL\_PC compiled

Trigger MODEL\_LAPTOP compiled

2.

```
Create Trigger ram_hd
Before Insert Or Update On PC
For Each Row
Begin
IF((:new.hd*1000) < (:new.ram*100))
THEN RAISE_APPLICATION_ERROR(-20004, 'Incorrect value');
End IF;
End;
/
```

```
Create Trigger ram_hd
Before Insert Or Update On PC
For Each Row
Begin
IF((:new.hd*1000) < (:new.ram*100))
THEN RAISE_APPLICATION_ERROR(-20004, 'Incorrect value');
End IF;
End;
/

INSERT INTO PC VALUES('2029','500','1000','10','41','499');
```

Script Output x

Task completed in 0.485 seconds

Error at Command Line : 146 Column : 13

Error report -

SQL Error: ORA-20004: Incorrect value

ORA-06512: at "ABUDUKH.RAM\_HD", line 3

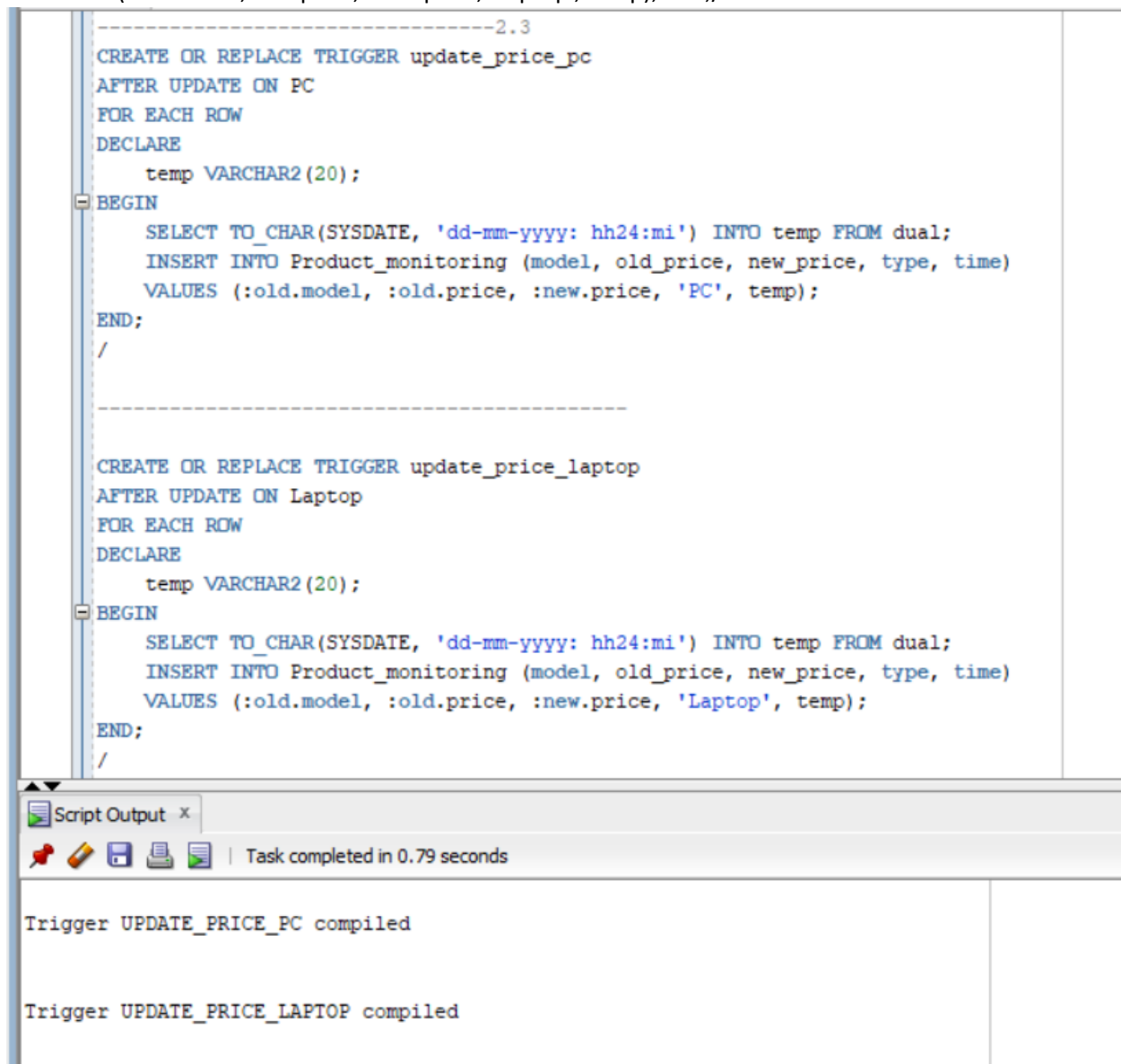
ORA-04088: error during execution of trigger 'ABUDUKH.RAM\_HD'

3. CREATE TABLE Product\_monitoring( model CHAR(10), old\_price INT, new\_price INT, type CHAR(10), time date);

```
CREATE TABLE Product_Monitoring(
    model CHAR(10),
    old_price INT,
    new_price INT,
    type CHAR(10),
    time date
);
```

```
CREATE OR REPLACE TRIGGER update_price_pc
AFTER UPDATE ON PC
FOR EACH ROW
DECLARE
    temp VARCHAR2(20);
BEGIN
    SELECT TO_CHAR(SYSDATE, 'dd-mm-yyyy: hh24:mi') INTO temp FROM dual;
    INSERT INTO Product_monitoring (model, old_price, new_price, type, temp)
    VALUES (:old.model, :old.price, :new.price, 'PC', temp);
END;
/
```

```
CREATE OR REPLACE TRIGGER update_price_laptop
AFTER UPDATE ON Laptop
FOR EACH ROW
DECLARE
    temp VARCHAR2(20);
BEGIN
    SELECT TO_CHAR(SYSDATE, 'dd-mm-yyyy: hh24:mi') INTO temp FROM dual;
    INSERT INTO Product_monitoring (model, old_price, new_price, type, temp)
    VALUES (:old.model, :old.price, :new.price, 'Laptop', temp);END;/
```



The screenshot shows a SQL IDE window with two triggers defined. The first trigger, `update_price_pc`, is for the `PC` table. The second trigger, `update_price_laptop`, is for the `Laptop` table. Both triggers use a temporary variable `temp` to store the current date and time. The IDE interface includes a script editor with syntax highlighting, a script output window at the bottom, and a status bar indicating task completion.

```
-----2.3
CREATE OR REPLACE TRIGGER update_price_pc
AFTER UPDATE ON PC
FOR EACH ROW
DECLARE
    temp VARCHAR2(20);
BEGIN
    SELECT TO_CHAR(SYSDATE, 'dd-mm-yyyy: hh24:mi') INTO temp FROM dual;
    INSERT INTO Product_monitoring (model, old_price, new_price, type, time)
    VALUES (:old.model, :old.price, :new.price, 'PC', temp);
END;
/

CREATE OR REPLACE TRIGGER update_price_laptop
AFTER UPDATE ON Laptop
FOR EACH ROW
DECLARE
    temp VARCHAR2(20);
BEGIN
    SELECT TO_CHAR(SYSDATE, 'dd-mm-yyyy: hh24:mi') INTO temp FROM dual;
    INSERT INTO Product_monitoring (model, old_price, new_price, type, time)
    VALUES (:old.model, :old.price, :new.price, 'Laptop', temp);
END;
/
```

Script Output x

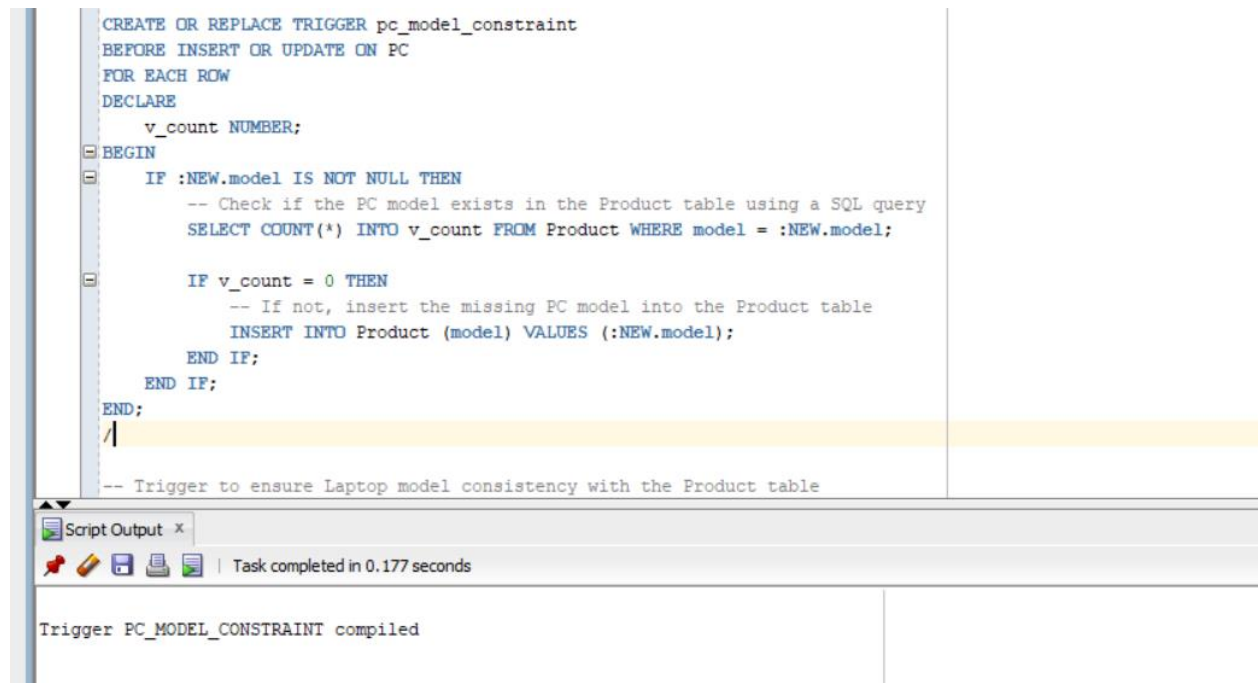
Task completed in 0.79 seconds

Trigger UPDATE\_PRICE\_PC compiled

Trigger UPDATE\_PRICE\_LAPTOP compiled

4.

```
CREATE OR REPLACE TRIGGER pc_model_constraint
BEFORE INSERT OR UPDATE ON PC
FOR EACH ROW
DECLARE
    v_count NUMBER;
BEGIN
    IF :NEW.model IS NOT NULL THEN
        SELECT COUNT(*) INTO v_count FROM Product WHERE model = :NEW.model;
        IF v_count = 0 THEN
            INSERT INTO Product (model) VALUES (:NEW.model);
        END IF;
    END IF;
END;
/
```



```
CREATE OR REPLACE TRIGGER pc_model_constraint
BEFORE INSERT OR UPDATE ON PC
FOR EACH ROW
DECLARE
    v_count NUMBER;
BEGIN
    IF :NEW.model IS NOT NULL THEN
        -- Check if the PC model exists in the Product table using a SQL query
        SELECT COUNT(*) INTO v_count FROM Product WHERE model = :NEW.model;

        IF v_count = 0 THEN
            -- If not, insert the missing PC model into the Product table
            INSERT INTO Product (model) VALUES (:NEW.model);
        END IF;
    END IF;
END;
/

-- Trigger to ensure Laptop model consistency with the Product table
```

Script Output x

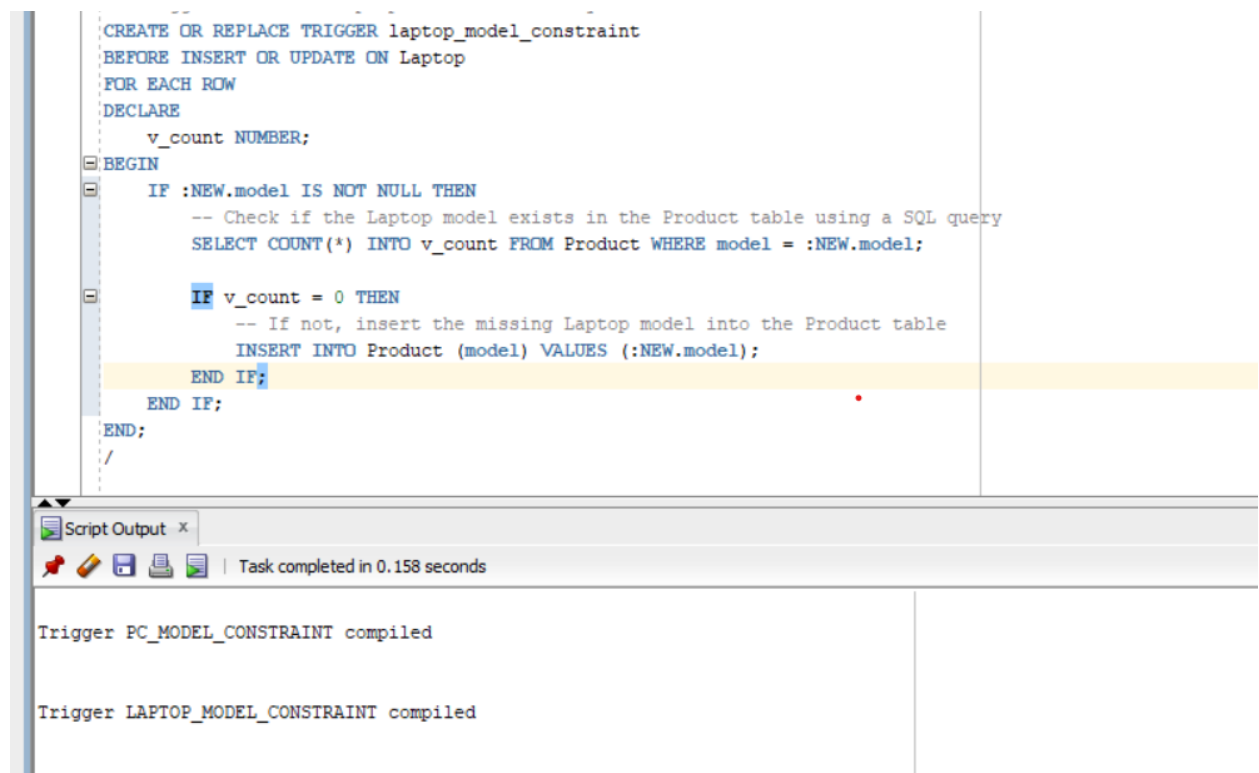
Task completed in 0.177 seconds

Trigger PC\_MODEL\_CONSTRAINT compiled

```

CREATE OR REPLACE TRIGGER laptop_model_constraint
BEFORE INSERT OR UPDATE ON Laptop
FOR EACH ROW
DECLARE
    v_count NUMBER;
BEGIN
    IF :NEW.model IS NOT NULL THEN
        SELECT COUNT(*) INTO v_count FROM Product WHERE model = :NEW.model;
        IF v_count = 0 THEN
            INSERT INTO Product (model) VALUES (:NEW.model);
        END IF;
    END IF;
END;
/

```



```

CREATE OR REPLACE TRIGGER laptop_model_constraint
BEFORE INSERT OR UPDATE ON Laptop
FOR EACH ROW
DECLARE
    v_count NUMBER;
BEGIN
    IF :NEW.model IS NOT NULL THEN
        -- Check if the Laptop model exists in the Product table using a SQL query
        SELECT COUNT(*) INTO v_count FROM Product WHERE model = :NEW.model;

        IF v_count = 0 THEN
            -- If not, insert the missing Laptop model into the Product table
            INSERT INTO Product (model) VALUES (:NEW.model);
        END IF;
    END IF;
END;
/

```

Script Output x

Task completed in 0.158 seconds

Trigger PC\_MODEL\_CONSTRAINT compiled

Trigger LAPTOP\_MODEL\_CONSTRAINT compiled