Project 2, Group 11

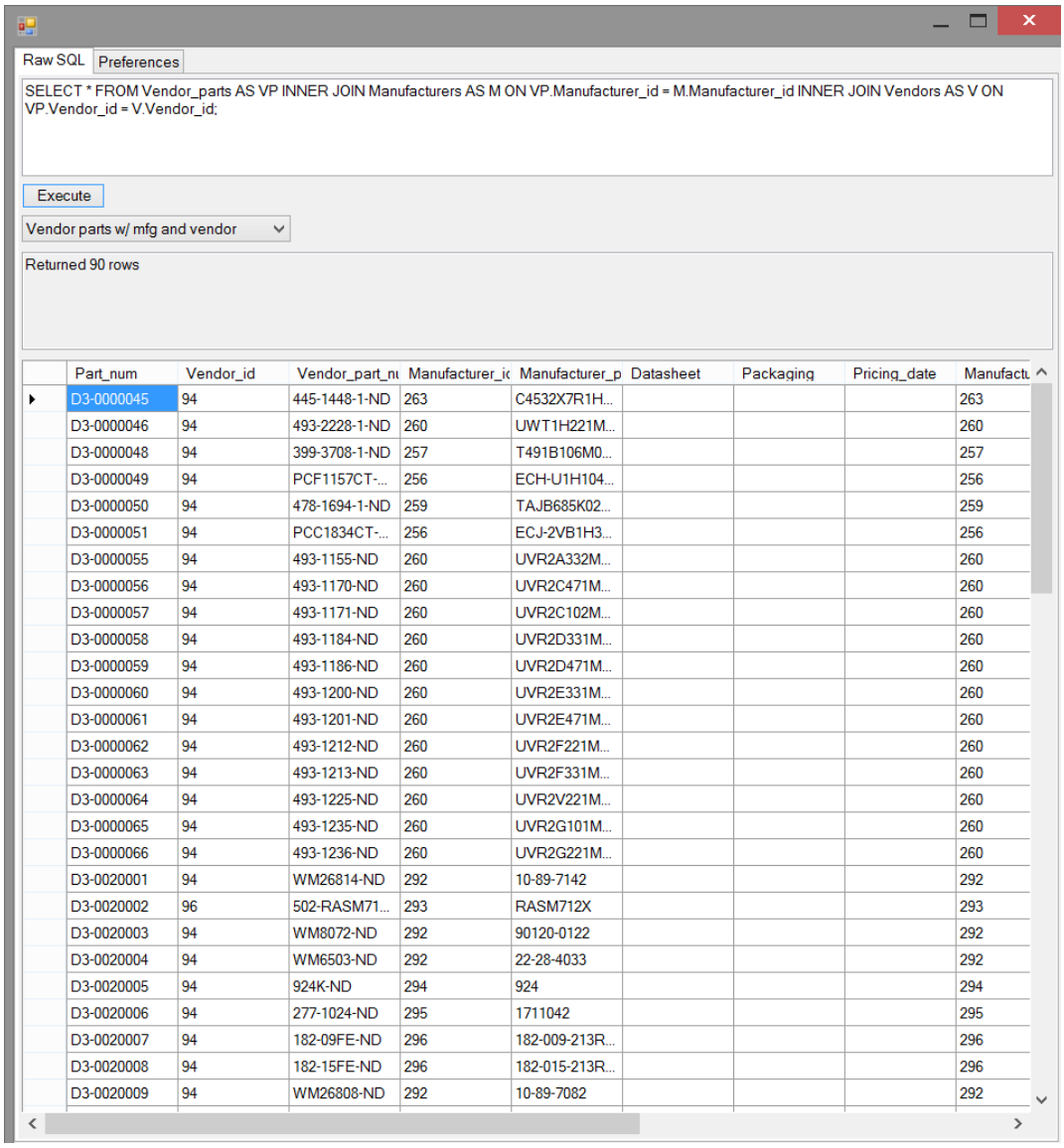
Task 1

Josh Watts
Carl Milazzo
2014-04-25

Inner Join

Select vendor parts with both mfg and vendor

```
SELECT * FROM Vendor_parts AS VP INNER JOIN Manufacturers AS M ON VP.Manufacturer_id = M.Manufacturer_id INNER JOIN Vendors AS V ON VP.Vendor_id = V.Vendor_id;
```

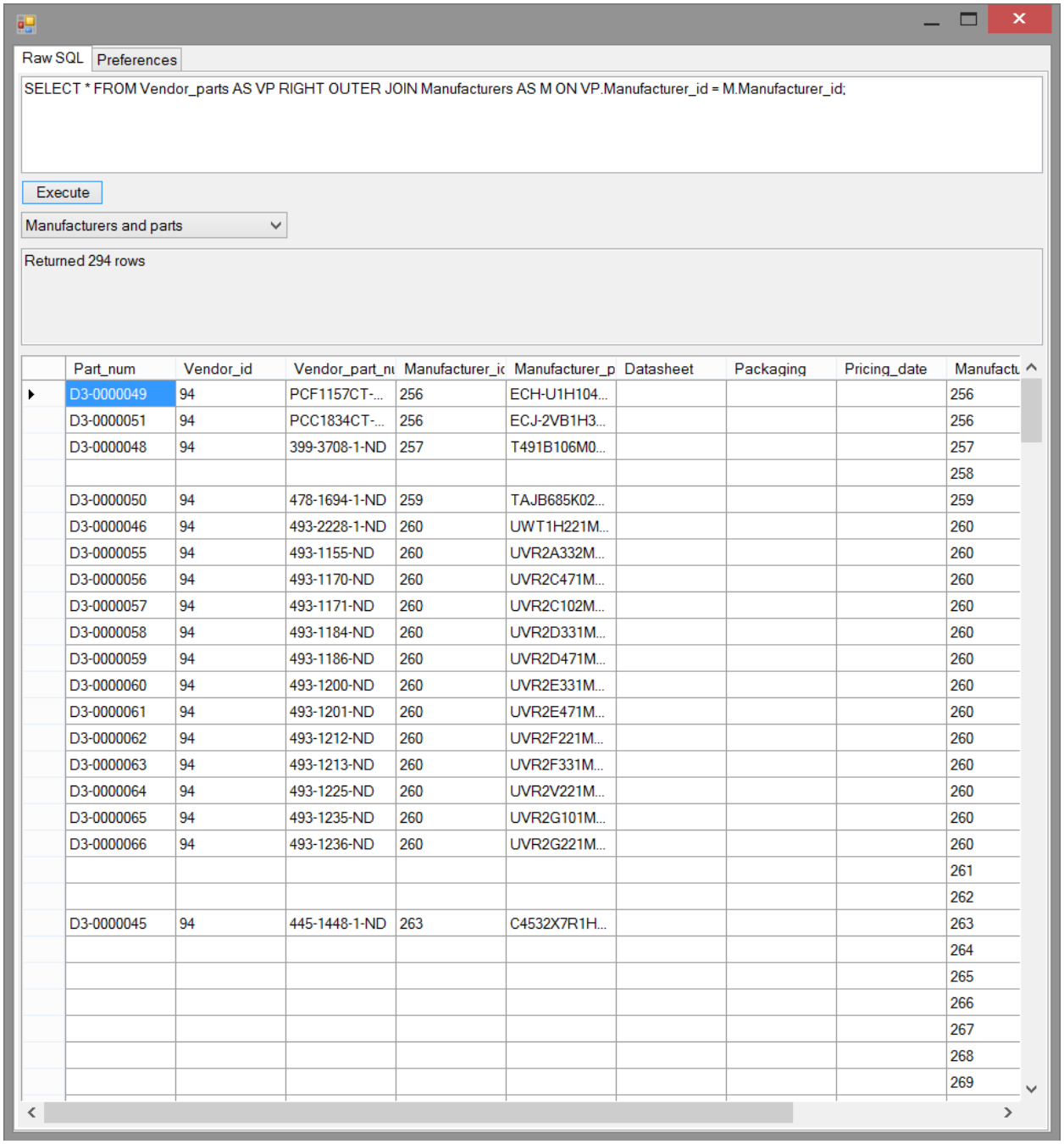


Part_num	Vendor_id	Vendor_part_n	Manufacturer_i	Manufacturer_p	Datasheet	Packaging	Pricing_date	Manufactu
D3-0000045	94	445-1448-1-ND	263	C4532X7R1H...				263
D3-0000046	94	493-2228-1-ND	260	UWT1H221M...				260
D3-0000048	94	399-3708-1-ND	257	T491B106M0...				257
D3-0000049	94	PCF1157CT-...	256	ECH-U1H104...				256
D3-0000050	94	478-1694-1-ND	259	TAJB685K02...				259
D3-0000051	94	PCC1834CT-...	256	ECJ-2VB1H3...				256
D3-0000055	94	493-1155-ND	260	UVR2A332M...				260
D3-0000056	94	493-1170-ND	260	UVR2C471M...				260
D3-0000057	94	493-1171-ND	260	UVR2C102M...				260
D3-0000058	94	493-1184-ND	260	UVR2D331M...				260
D3-0000059	94	493-1186-ND	260	UVR2D471M...				260
D3-0000060	94	493-1200-ND	260	UVR2E331M...				260
D3-0000061	94	493-1201-ND	260	UVR2E471M...				260
D3-0000062	94	493-1212-ND	260	UVR2F221M...				260
D3-0000063	94	493-1213-ND	260	UVR2F331M...				260
D3-0000064	94	493-1225-ND	260	UVR2V221M...				260
D3-0000065	94	493-1235-ND	260	UVR2G101M...				260
D3-0000066	94	493-1236-ND	260	UVR2G221M...				260
D3-0020001	94	WM26814-ND	292	10-89-7142				292
D3-0020002	96	502-RASM71...	293	RASM712X				293
D3-0020003	94	WM8072-ND	292	90120-0122				292
D3-0020004	94	WM6503-ND	292	22-28-4033				292
D3-0020005	94	924K-ND	294	924				294
D3-0020006	94	277-1024-ND	295	1711042				295
D3-0020007	94	182-09FE-ND	296	182-009-213R...				296
D3-0020008	94	182-15FE-ND	296	182-015-213R...				296
D3-0020009	94	WM26808-ND	292	10-89-7082				292

Outer join

Select manufacturers and their parts, including those with no parts

SELECT * FROM Vendor_parts AS VP RIGHT OUTER JOIN Manufacturers AS M ON VP.Manufacturer_id = M.Manufacturer_id;

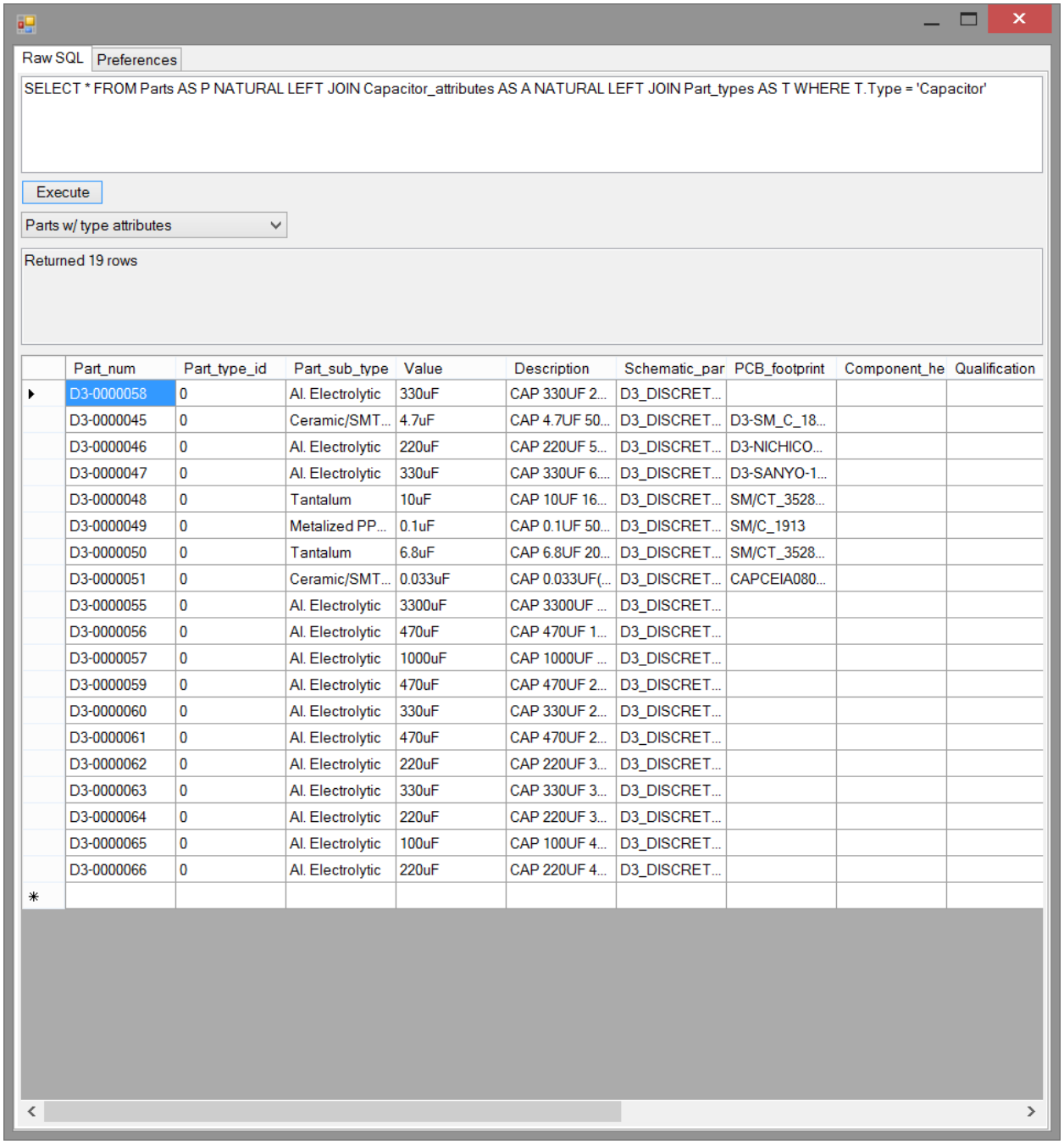


Part_num	Vendor_id	Vendor_part_nu	Manufacturer_ic	Manufacturer_p	Datasheet	Packaging	Pricing_date	Manufactu
D3-0000049	94	PCF1157CT-...	256	ECH-U1H104...				256
D3-0000051	94	PCC1834CT-...	256	ECJ-2VB1H3...				256
D3-0000048	94	399-3708-1-ND	257	T491B106M0...				257
								258
D3-0000050	94	478-1694-1-ND	259	TAJB685K02...				259
D3-0000046	94	493-2228-1-ND	260	UWT1H221M...				260
D3-0000055	94	493-1155-ND	260	UVR2A332M...				260
D3-0000056	94	493-1170-ND	260	UVR2C471M...				260
D3-0000057	94	493-1171-ND	260	UVR2C102M...				260
D3-0000058	94	493-1184-ND	260	UVR2D331M...				260
D3-0000059	94	493-1186-ND	260	UVR2D471M...				260
D3-0000060	94	493-1200-ND	260	UVR2E331M...				260
D3-0000061	94	493-1201-ND	260	UVR2E471M...				260
D3-0000062	94	493-1212-ND	260	UVR2F221M...				260
D3-0000063	94	493-1213-ND	260	UVR2F331M...				260
D3-0000064	94	493-1225-ND	260	UVR2V221M...				260
D3-0000065	94	493-1235-ND	260	UVR2G101M...				260
D3-0000066	94	493-1236-ND	260	UVR2G221M...				260
								261
								262
D3-0000045	94	445-1448-1-ND	263	C4532X7R1H...				263
								264
								265
								266
								267
								268
								269

Natural join

Natural join of parts / type_attributes

SELECT * FROM Parts AS P NATURAL LEFT JOIN Capacitor_attributes AS A NATURAL LEFT JOIN Part_types AS T WHERE T.Type = "Capacitor"

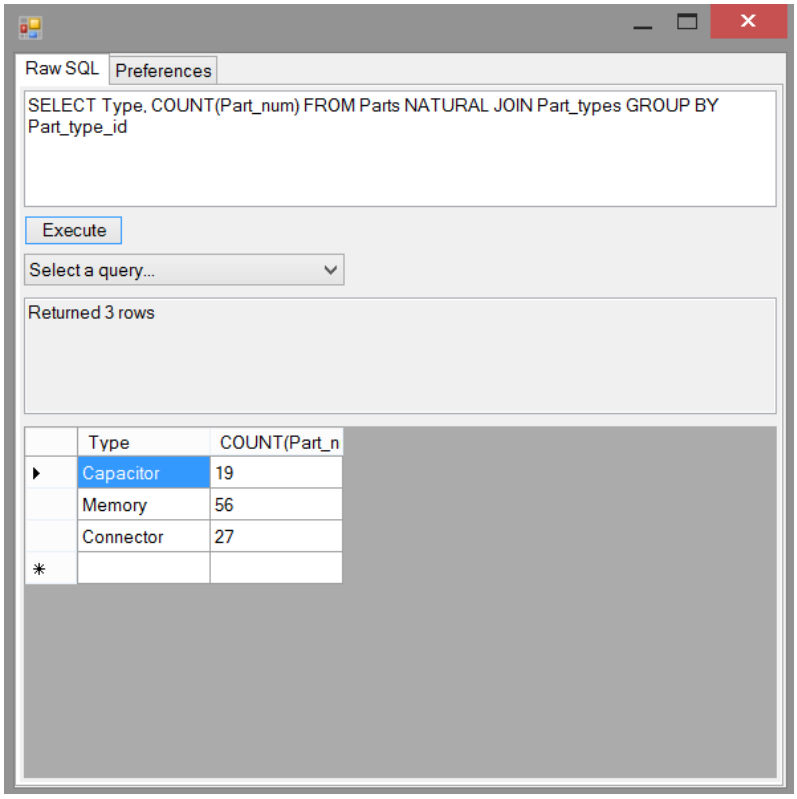


Part_num	Part_type_id	Part_sub_type	Value	Description	Schematic_par	PCB_footprint	Component_he	Qualification
D3-0000058	0	Al. Electrolytic	330uF	CAP 330UF 2...	D3_DISCRET...			
D3-0000045	0	Ceramic/SMT...	4.7uF	CAP 4.7UF 50...	D3_DISCRET...	D3-SM_C_18...		
D3-0000046	0	Al. Electrolytic	220uF	CAP 220UF 5...	D3_DISCRET...	D3-NICHICO...		
D3-0000047	0	Al. Electrolytic	330uF	CAP 330UF 6....	D3_DISCRET...	D3-SANYO-1...		
D3-0000048	0	Tantalum	10uF	CAP 10UF 16...	D3_DISCRET...	SM/CT_3528...		
D3-0000049	0	Metalized PP...	0.1uF	CAP 0.1UF 50...	D3_DISCRET...	SM/C_1913		
D3-0000050	0	Tantalum	6.8uF	CAP 6.8UF 20...	D3_DISCRET...	SM/CT_3528...		
D3-0000051	0	Ceramic/SMT...	0.033uF	CAP 0.033UF(...	D3_DISCRET...	CAPCEIA080...		
D3-0000055	0	Al. Electrolytic	3300uF	CAP 3300UF ...	D3_DISCRET...			
D3-0000056	0	Al. Electrolytic	470uF	CAP 470UF 1...	D3_DISCRET...			
D3-0000057	0	Al. Electrolytic	1000uF	CAP 1000UF ...	D3_DISCRET...			
D3-0000059	0	Al. Electrolytic	470uF	CAP 470UF 2...	D3_DISCRET...			
D3-0000060	0	Al. Electrolytic	330uF	CAP 330UF 2...	D3_DISCRET...			
D3-0000061	0	Al. Electrolytic	470uF	CAP 470UF 2...	D3_DISCRET...			
D3-0000062	0	Al. Electrolytic	220uF	CAP 220UF 3...	D3_DISCRET...			
D3-0000063	0	Al. Electrolytic	330uF	CAP 330UF 3...	D3_DISCRET...			
D3-0000064	0	Al. Electrolytic	220uF	CAP 220UF 3...	D3_DISCRET...			
D3-0000065	0	Al. Electrolytic	100uF	CAP 100UF 4...	D3_DISCRET...			
D3-0000066	0	Al. Electrolytic	220uF	CAP 220UF 4...	D3_DISCRET...			

Aggregate function

Count of each type of part

SELECT Type, COUNT(Part_num) FROM Parts NATURAL JOIN Part_types GROUP BY Part_type_id



Commit and Rollback

begin
Insert part
if part_num match d3_% then commit
else rollback

Trigger

Add null attributes tuple upon new part?

```
DELIMITER $$
CREATE TRIGGER trgNewPart AFTER INSERT ON Parts
FOR EACH ROW
BEGIN
   CASE NEW.Part_type_id
      WHEN 0 THEN INSERT INTO Capacitor_attributes (Part_num) VALUES (NEW.Part_num);
      WHEN 1 THEN INSERT INTO Memory_attributes (Part_num) VALUES (NEW.Part_num);
      WHEN 2 THEN INSERT INTO Connector_attributes (Part_num) VALUES
(NEW.Part_num);
   END CASE;
END;$$
DELIMITER ;
```

Trigger (another trigger)

Delete attributes tuple upon deletion

```
DELIMITER $$
CREATE TRIGGER trgNewPart BEFORE DELETE ON Parts
FOR EACH ROW
BEGIN
   CASE NEW.Part_type_id
      WHEN 0 THEN DELETE FROM Capacitor_attributes WHERE Part_num =
NEW.Part_num;
      WHEN 1 THEN DELETE FROM Memory_attributes WHERE Part_num = NEW.Part_num;
      WHEN 2 THEN DELETE FROM Connector_attributes WHERE Part_num =
NEW.Part_num;
   END CASE;
END;$$
DELIMITER ;
```

Trigger demonstration

```
DROP TRIGGER IF EXISTS trgNewPart;
DROP TRIGGER IF EXISTS trgDeletedPart;
DELETE FROM Capacitor_attributes WHERE Part_num = "Rofl";
DELETE FROM Parts WHERE Part_num = "Rofl";

DELIMITER $$
CREATE TRIGGER trgNewPart AFTER INSERT ON Parts
FOR EACH ROW
BEGIN
    CASE NEW.Part_type_id
```

```
        WHEN 0 THEN INSERT INTO Capacitor_attributes (Part_num)
VALUES (NEW.Part_num);
        WHEN 1 THEN INSERT INTO Memory_attributes (Part_num)
VALUES (NEW.Part_num);
        WHEN 2 THEN INSERT INTO Connector_attributes (Part_num)
VALUES (NEW.Part_num);
    END CASE;
END;$$
DELIMITER ;

INSERT INTO Parts (Part_num, Part_type_id) VALUES ("Rofl", 0);
SELECT * FROM Parts WHERE Part_num = "Rofl";
+----------+--------------+---------------+-------+-------------
+----------------+--------------+------------------+-----------
----+----------------+-----------------+
| Part_num | Part_type_id | Part_sub_type | Value | Description
| Schematic_part | PCB_footprint | Component_height |
Qualification | Low_temp_range | High_temp_range |
+----------+--------------+---------------+-------+-------------
+----------------+--------------+------------------+-----------
----+----------------+-----------------+
| Rofl     |            0 |               | NULL  | NULL
| NULL           | NULL          | NULL             | NULL
| NULL           | NULL            |
+----------+--------------+---------------+-------+-------------
+----------------+--------------+------------------+-----------
----+----------------+-----------------+

SELECT * FROM Capacitor_attributes WHERE Part_num = "Rofl";
+----------+---------+-------+-----------+
| Part_num | Voltage | Power | Tolerance |
+----------+---------+-------+-----------+
| Rofl     | NULL    | NULL  | NULL      |
+----------+---------+-------+-----------+

DELETE FROM Parts WHERE Part_num = "Rofl";
ERROR 1451 (23000) at line 2: Cannot delete or update a parent
row: a foreign key constraint fails
(`PartsDB`.`Capacitor_attributes`, CONSTRAINT
`fk_Capacitor_attributes_Parts1` FOREIGN KEY (`Part_num`)
REFERENCES `Parts` (`Part_num`) ON DELETE NO ACTION ON UPDATE NO
ACTION)
```

```
DELIMITER $$
CREATE TRIGGER trgDeletedPart BEFORE DELETE ON Parts
FOR EACH ROW
BEGIN
    CASE OLD.Part_type_id
        WHEN 0 THEN DELETE FROM Capacitor_attributes WHERE
Part_num = OLD.Part_num;
        WHEN 1 THEN DELETE FROM Memory_attributes WHERE Part_num
= OLD.Part_num;
        WHEN 2 THEN DELETE FROM Connector_attributes WHERE
Part_num = OLD.Part_num;
    END CASE;
END;$$
DELIMITER ;

DELETE FROM Parts WHERE Part_num = "Rofl";
SELECT * FROM Capacitor_attributes WHERE Part_num = "Rofl";
```

Stored Procedure

Update bom cost with current pricing

```
DELIMITER $$
CREATE PROCEDURE updateBom(PCA_id VARCHAR(16), BOM_rev VARCHAR(5))
BEGIN
   DECLARE item_num INT;
   DECLARE Part_num CHAR(16);
   DECLARE done INT DEFAULT FALSE;
   DECLARE cBom CURSOR FOR SELECT BOM.Item_num,BOM.Part_num FROM
BillOfMaterials AS BOM WHERE BOM.PCA_id = PCA_id AND BOM.BOM_rev = BOM_rev;
   DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

  -- TODO: Price needs to be based on Vendor_part_num

   OPEN cBom;
   read_loop: LOOP
      FETCH cBom INTO item_num, Part_num;
      IF done THEN
         LEAVE read_loop;
      END IF;
      UPDATE BillOfMaterials AS BOM
         SET
            Price_qty_1 = (SELECT MAX(PB.Price) FROM Price_Break as PB WHERE
PB.Part_num = Part_num AND Break_num <= 1)
         WHERE
```

```
            BOM.PCA_id = PCA_id AND BOM.BOM_rev = BOM_rev AND BOM.Item_num =
item_num;
      UPDATE BillOfMaterials AS BOM
         SET
            Price_qty_1000 = (SELECT MAX(PB.Price) FROM Price_Break as PB WHERE
PB.Part_num = Part_num AND Break_num <= 1000)
         WHERE
            BOM.PCA_id = PCA_id AND BOM.BOM_rev = BOM_rev AND BOM.Item_num =
item_num;
   END LOOP;
   CLOSE cBom;
END;$$
DELIMITER ;
```

Stored Procedure Demonstration

```
DROP PROCEDURE IF EXISTS updateBom;

DELETE FROM Price_Break WHERE Part_num = "Test";
DELETE FROM Vendor_parts WHERE Part_num = "Test";
DELETE FROM Vendors WHERE Vendor_id = 999;
DELETE FROM Manufacturers WHERE Manufacturer_id = 999;
DELETE FROM Capacitor_attributes WHERE Part_num = "Test";
DELETE FROM Parts WHERE Part_num = "Test";
DELETE FROM BillOfMaterials WHERE Part_num = "Test";
DELETE FROM PCA WHERE PCA_id = "PCA0";
DELETE FROM Projects WHERE Project_id = "Prj";
DELETE FROM Customer WHERE Customer_id = "Cst";

DELIMITER $$
CREATE PROCEDURE updateBom(PCA_id VARCHAR(16), BOM_rev
VARCHAR(5))
BEGIN
    DECLARE item_num INT;
    DECLARE Part_num CHAR(16);
    DECLARE done INT DEFAULT FALSE;
    DECLARE cBom CURSOR FOR SELECT
BOM.Item_num,BOM.Part_num FROM BillOfMaterials AS BOM
WHERE BOM.PCA_id = PCA_id AND BOM.BOM_rev = BOM_rev;
```

```
    DECLARE CONTINUE HANDLER FOR NOT FOUND SET done =
TRUE;

    OPEN cBom;
    read_loop: LOOP
        FETCH cBom INTO item_num, Part_num;
        IF done THEN
            LEAVE read_loop;
        END IF;
        UPDATE BillOfMaterials AS BOM
            SET
                Price_qty_1 = (SELECT MAX(PB.Price) FROM
Price_Break as PB WHERE PB.Part_num = Part_num AND
Break_num <= 1)
            WHERE
                BOM.PCA_id = PCA_id AND BOM.BOM_rev =
BOM_rev AND BOM.Item_num = item_num;
        UPDATE BillOfMaterials AS BOM
            SET
                Price_qty_1000 = (SELECT MAX(PB.Price)
FROM Price_Break as PB WHERE PB.Part_num = Part_num AND
Break_num <= 1000)
            WHERE
                BOM.PCA_id = PCA_id AND BOM.BOM_rev =
BOM_rev AND BOM.Item_num = item_num;
    END LOOP;
    CLOSE cBom;
END;$$
DELIMITER ;

INSERT INTO Parts (Part_num, Part_type_id) VALUES
("Test", 0);
INSERT INTO Vendors (Vendor_id) VALUES (999);
INSERT INTO Manufacturers (Manufacturer_id) VALUES (999);
```

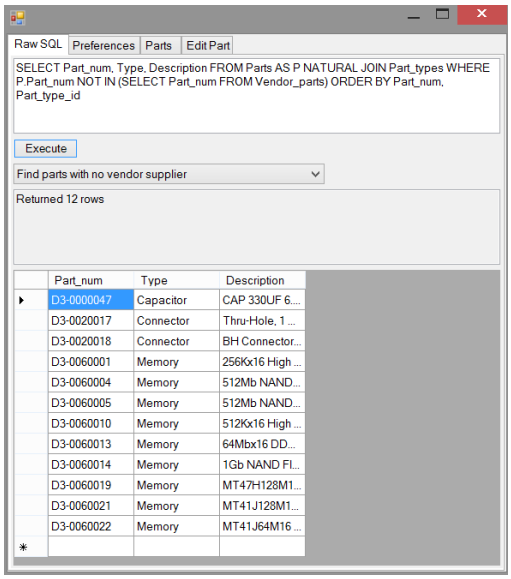
```
INSERT INTO Vendor_parts (Vendor_id, Vendor_part_num,
Manufacturer_id, Part_num) VALUES (999, "Test", 999,
"Test");
INSERT INTO Price_Break (Break_num, Part_num, Price)
VALUES (1, "Test", 1.23);
INSERT INTO Price_Break (Break_num, Part_num, Price)
VALUES (1000, "Test", 0.42);
INSERT INTO Customer (Customer_id) VALUES ("Cst");
INSERT INTO Projects (Project_id, Customer_id) VALUES
("Prj", "Cst");
INSERT INTO PCA (PCA_id, Project_id) VALUES ("PCA0",
"Prj");
INSERT INTO BillOfMaterials (PCA_id, BOM_rev, Item_num,
Part_num) VALUES ("PCA0", 1, 1, "Test");

SELECT Part_num, Price_qty_1, Price_qty_1000 FROM
BillOfMaterials;
CALL updateBom("PCA0", 1);
SELECT Part_num, Price_qty_1, Price_qty_1000 FROM
BillOfMaterials;
```

Other SQL statements

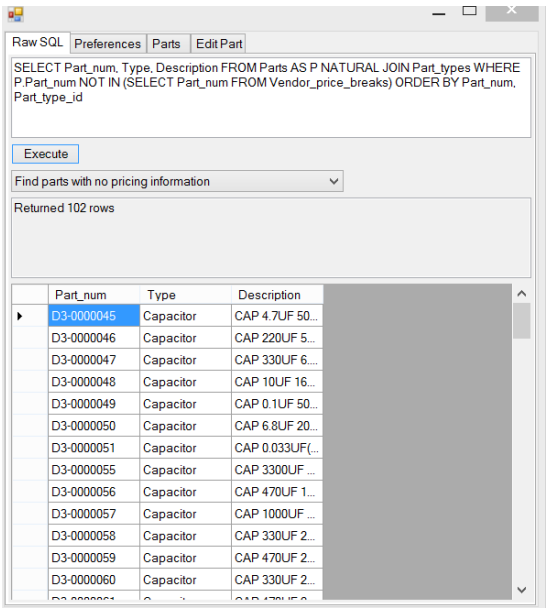
Find parts with no vendor supplier

SELECT Part_num, Type, Description FROM Parts AS P NATURAL JOIN Part_types WHERE P.Part_num NOT IN (SELECT Part_num FROM Vendor_parts) ORDER BY Part_num, Part_type_id

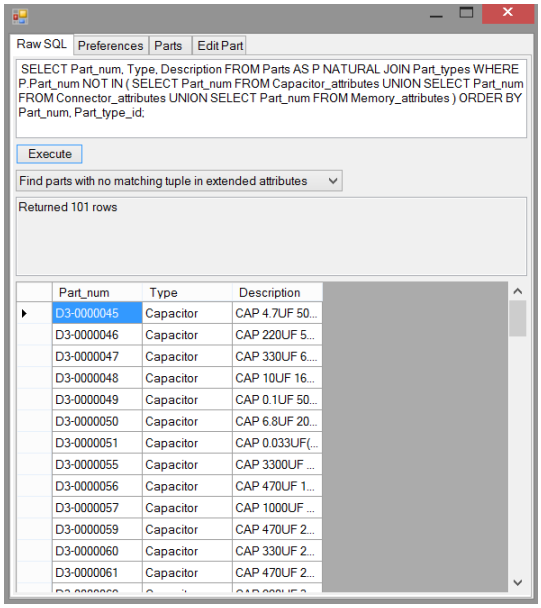


Find parts with no pricing information

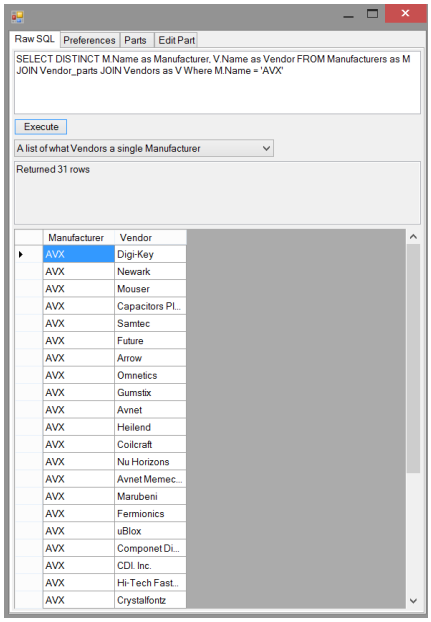
SELECT Part_num, Type, Description FROM Parts AS P NATURAL JOIN Part_types WHERE P.Part_num NOT IN (SELECT Part_num FROM Vendor_price_breaks) ORDER BY Part_num, Part_type_id



Find parts with no matching tuple in extended attributes
SELECT Part_num, Type, Description FROM Parts AS P NATURAL JOIN Part_types WHERE P.Part_num NOT IN (SELECT Part_num FROM Capacitor_attributes UNION SELECT Part_num FROM Connector_attributes UNION SELECT Part_num FROM Memory_attributes) ORDER BY Part_num, Part_type_id;

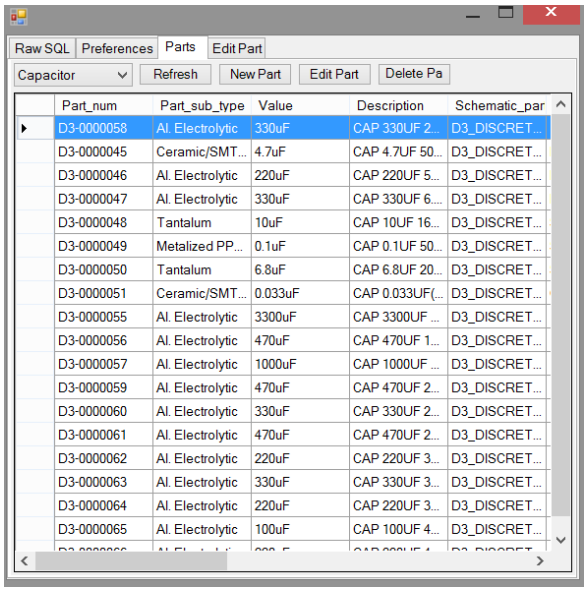


A list of what Vendors a single Manufacturer
SELECT DISTINCT M.Name as Manufacturer, V.Name as Vendor FROM Manufacturers as M JOIN Vendor_parts JOIN Vendors as V Where M.Name = 'AVX'

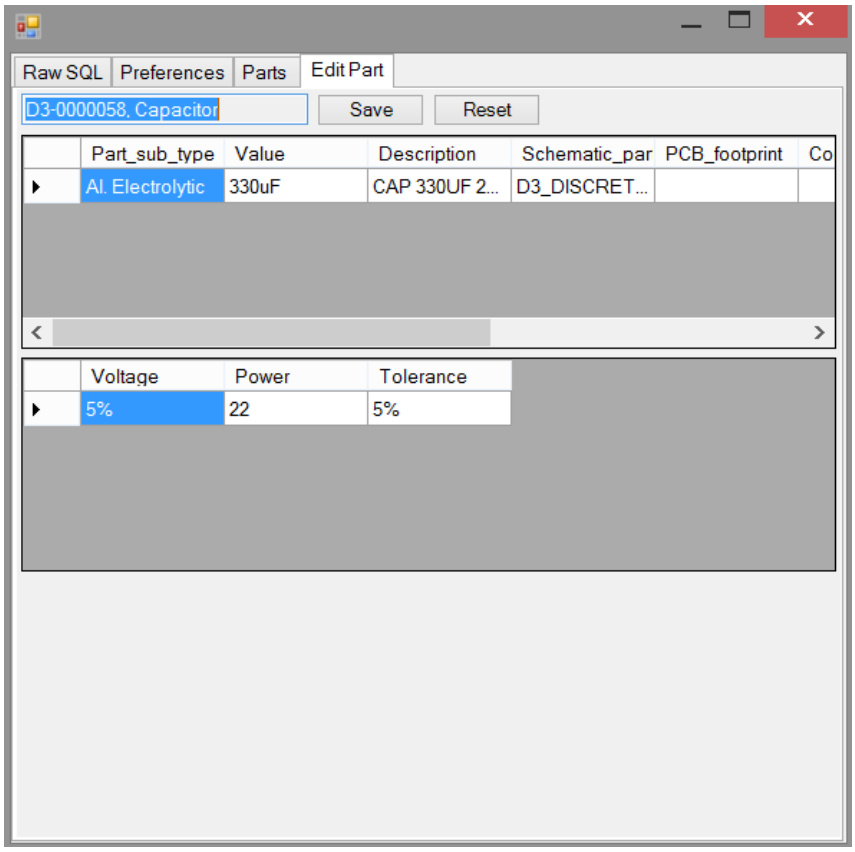


Other Features

You have the ability to change a part in the GUI.
Just go to the Parts tab, click a part then hit the “Edit Part” button.



You then will be brought to a new tab, where you can manually change the part. Once finished hit the “Save” button and the code will make an sql statement and run it to update the part on the database.



Other Screenshots