Team GLASTA's *Fantastic Furniture*: Database Queries

Team Info

Team Name:	Team GLASTA	
Project Name:	*Fantastic Furniture*	
Participants:	Timothy Gibson	tgibson1@csustan.edu
	Alexander Altman	aaltman@csustan.edu
	Schuyler Davis	sdavis20@csustan.edu

SQL Schemas

```sql
CREATE TABLE Supplier
  (
     supplierID VARCHAR(10) CHARACTER SET ASCII,
     name_      VARCHAR(50) CHARACTER SET utf8mb4 NOT NULL,
     phone      VARCHAR(12) CHARACTER SET ASCII,
     address    VARCHAR(100) CHARACTER SET utf8mb4,
     country    CHAR(2) CHARACTER SET ASCII DEFAULT 'US',
     website    VARCHAR(50) CHARACTER SET utf8mb4,
     PRIMARY KEY(supplierID),
     CHECK (country REGEXP '^[A-Z]{2}$'),
     CHECK (phone REGEXP '^[0-9]{7,12}$')
  );

CREATE TABLE Designer
  (
     designerID  VARCHAR(10) CHARACTER SET ASCII,
     name_       VARCHAR(50) CHARACTER SET utf8mb4 NOT NULL,
     phone       VARCHAR(12) CHARACTER SET ASCII,
     address     VARCHAR(100) CHARACTER SET utf8mb4,
     country     CHAR(2) CHARACTER SET ASCII DEFAULT 'US',
     website     VARCHAR(50) CHARACTER SET utf8mb4,
     designFocus VARCHAR(100) CHARACTER SET utf8mb4,
     PRIMARY KEY(designerID),
     CHECK (country REGEXP '^[A-Z]{2}$'),
     CHECK (phone REGEXP '^[0-9]{7,12}$')
  );

CREATE TABLE Set_
  (
     setID         VARCHAR(10) CHARACTER SET ASCII,
     name_         VARCHAR(50) CHARACTER SET utf8mb4 NOT NULL,
```

```
      catalogYear   DECIMAL(4, 0) UNSIGNED,
      catalogNumber BIGINT UNSIGNED ZEROFILL NOT NULL,
      style_        VARCHAR(30) CHARACTER SET utf8mb4,
      PRIMARY KEY(setID)
   );

CREATE TABLE Model
   (
      modelNumber VARCHAR(10) CHARACTER SET ASCII,
      name_       VARCHAR(50) CHARACTER SET utf8mb4 NOT NULL,
      material    VARCHAR(30) CHARACTER SET utf8mb4,
      upholstery  VARCHAR(30) CHARACTER SET utf8mb4,
      durability  VARCHAR(30) CHARACTER SET utf8mb4,
      color       VARCHAR(30) CHARACTER SET utf8mb4,
      PRIMARY KEY(modelNumber)
   );

CREATE TABLE Item
   (
      sku         VARCHAR(10) CHARACTER SET ASCII,
      length_     DOUBLE UNSIGNED,
      width       DOUBLE UNSIGNED,
      height      DOUBLE UNSIGNED,
      condition_  VARCHAR(30) CHARACTER SET utf8mb4,
      weightLimit DOUBLE UNSIGNED,
      PRIMARY KEY(sku),
      CHECK (length_ > 0.0),
      CHECK (width > 0.0),
      CHECK (height > 0.0),
      CHECK (weightLimit > 0.0)
   );

CREATE TABLE DistributionCenter
   (
      centerID VARCHAR(10) CHARACTER SET ASCII,
      name_    VARCHAR(50) CHARACTER SET utf8mb4 NOT NULL,
      phone    VARCHAR(12) CHARACTER SET ASCII,
      address  VARCHAR(100) CHARACTER SET utf8mb4,
      country  CHAR(2) CHARACTER SET ASCII DEFAULT 'US',
      website  VARCHAR(50) CHARACTER SET utf8mb4,
      PRIMARY KEY(centerID),
      CHECK (country REGEXP '^[A-Z]{2}$'),
      CHECK (phone REGEXP '^[0-9]{7,12}$')
   );

```

```
77  CREATE TABLE make
78    (
79       supplierID VARCHAR(10) CHARACTER SET ASCII,
80       designerID VARCHAR(10) CHARACTER SET ASCII,
81       setID      VARCHAR(10) CHARACTER SET ASCII,
82       PRIMARY KEY(supplierID, designerID, setID),
83       FOREIGN KEY(supplierID) REFERENCES Supplier(supplierID),
84       FOREIGN KEY(designerID) REFERENCES Designer(designerID),
85       FOREIGN KEY(setID) REFERENCES Set_(setID)
86    );
87
88  CREATE TABLE contains_
89    (
90       setID       VARCHAR(10) CHARACTER SET ASCII,
91       modelNumber VARCHAR(10) CHARACTER SET ASCII,
92       count_      TINYINT UNSIGNED DEFAULT 1,
93       PRIMARY KEY(setID, modelNumber),
94       FOREIGN KEY(setID) REFERENCES Set_(setID),
95       FOREIGN KEY(modelNumber) REFERENCES Model(modelNumber),
96       CHECK (count_ > 0)
97    );
98
99  CREATE TABLE describes
100   (
101      modelNumber VARCHAR(10) CHARACTER SET ASCII NOT NULL,
102      sku         VARCHAR(10) CHARACTER SET ASCII,
103      PRIMARY KEY(sku),
104      FOREIGN KEY(modelNumber) REFERENCES Model(modelNumber),
105      FOREIGN KEY(sku) REFERENCES Item(sku)
106   );
107
108 CREATE TABLE canOrderFrom
109   (
110      centerID   VARCHAR(10) CHARACTER SET ASCII,
111      supplierID VARCHAR(10) CHARACTER SET ASCII,
112      leadTime   DOUBLE UNSIGNED,
113      PRIMARY KEY(centerID, supplierID),
114      FOREIGN KEY(centerID) REFERENCES DistributionCenter(centerID),
115      FOREIGN KEY(supplierID) REFERENCES Supplier(supplierID)
116   );
117
118 CREATE TABLE stocks
119   (
120      centerID VARCHAR(10) CHARACTER SET ASCII NOT NULL,
121      sku      VARCHAR(10) CHARACTER SET ASCII,
```

```
      PRIMARY KEY(sku),
      FOREIGN KEY(centerID) REFERENCES DistributionCenter(centerID),
      FOREIGN KEY(sku) REFERENCES Item(sku)
   );

CREATE TABLE Chair
   (
      sku          VARCHAR(10) CHARACTER SET ASCII,
      numberOfLegs TINYINT UNSIGNED DEFAULT 4,
      hasCushion   BOOL DEFAULT false,
      hasArms      BOOL,
      backHeight   DOUBLE UNSIGNED,
      seatHeight   DOUBLE UNSIGNED,
      PRIMARY KEY(sku),
      FOREIGN KEY(sku) REFERENCES Item(sku),
      CHECK (numberOfLegs > 0),
      CHECK (backHeight > 0.0),
      CHECK (seatHeight > 0.0)
   );

CREATE TABLE Table_
   (
      sku           VARCHAR(10) CHARACTER SET ASCII,
      numberOfLegs  TINYINT UNSIGNED DEFAULT 4,
      numberOfSeats TINYINT UNSIGNED,
      shape         VARCHAR(30) CHARACTER SET utf8mb4,
      PRIMARY KEY(sku),
      FOREIGN KEY(sku) REFERENCES Item(sku),
      CHECK (numberOfLegs > 0),
      CHECK (numberOfSeats > 0)
   );

CREATE TABLE Desk
   (
      sku             VARCHAR(10) CHARACTER SET ASCII,
      angle           DOUBLE DEFAULT 0.0,
      numberOfDrawers TINYINT UNSIGNED,
      PRIMARY KEY(sku),
      FOREIGN KEY(sku) REFERENCES Item(sku),
      CHECK (angle > -360.0 AND angle < 360.0),
      CHECK (numberOfDrawers > 0)
   );

CREATE TABLE Stool
   (
```

```
    sku          VARCHAR(10) CHARACTER SET ASCII,
    numberOfLegs TINYINT UNSIGNED,
    hasCushion   BOOL,
    hasSwivel    BOOL,
    PRIMARY KEY(sku),
    FOREIGN KEY(sku) REFERENCES Item(sku),
    CHECK (numberOfLegs > 0)
  );

CREATE TABLE Cabinet
  (
    sku                  VARCHAR(10) CHARACTER SET ASCII,
    numberOfCompartments TINYINT UNSIGNED,
    capacity             VARCHAR(30) CHARACTER SET utf8mb4,
    PRIMARY KEY(sku),
    FOREIGN KEY(sku) REFERENCES Item(sku),
    CHECK (numberOfCompartments > 0)
  );

CREATE TABLE Bedframe
  (
    sku    VARCHAR(10) CHARACTER SET ASCII,
    size_  VARCHAR(30) CHARACTER SET utf8mb4,
    depth_ DOUBLE,
    PRIMARY KEY(sku),
    FOREIGN KEY(sku) REFERENCES Item(sku)
  );

CREATE TABLE features_Feature
  (
    modelNumber VARCHAR(10) CHARACTER SET ASCII,
    description VARCHAR(50) CHARACTER SET utf8mb4,
    count_      TINYINT UNSIGNED DEFAULT 1,
    PRIMARY KEY(modelNumber, description),
    FOREIGN KEY(modelNumber) REFERENCES Model(modelNumber),
    CHECK (count_ > 0)
  );
```

Sample Queries

(A) **Intent:** "How many models has each designer designed?"

Query:
```
SELECT M.designerID,
       COUNT(C.modelNumber)
FROM   Designer D
```

```
        LEFT OUTER JOIN make M
                     ON ( D.designerID = M.designerID )
        INNER JOIN contains_ C
                ON ( M.setID = C.setID )
GROUP  BY M.designerID;
```

Result:

```
+------------+----------------------+
| designerID | count(C.modelNumber) |
+------------+----------------------+
| 3lejckzNYS |                   69 |
| 4MVbu2iI15 |                  170 |
| 7RYpyw9es0 |                  212 |
| 7ZZbCsXnv0 |                  210 |
| A7oA1v9Ax1 |                  212 |
| AZadqlHsUN |                  232 |
| b9yOGUx3pl |                  213 |
| bBMMbiXWX2 |                  189 |
| bDcoTRYgku |                  148 |
| BejwYSNzm7 |                  142 |
| bjSmt6EX8o |                  193 |
| bnQDB9V4ZQ |                  215 |
| bOQ84LG8yQ |                  246 |
| BVP4o4g0u6 |                  121 |
| c1jIajAyha |                  142 |
| czfBYIFNhs |                  252 |
| eAm5FaKjru |                  196 |
| ejSgB4P19T |                  215 |
| EZn2c6Sqao |                  164 |
| fLM0vFMD6h |                  190 |
| HACGEeYiTg |                  182 |
| hL6koxT8vK |                  265 |
| HULdxdPYgo |                  130 |
| HUPang5JW4 |                  181 |
| in19yTwFqy |                  254 |
| jMt9cpvHJ8 |                  218 |
| kFNSGfDIXN |                  200 |
| KH4hmznKQN |                  252 |
| KoaWPsykpt |                  244 |
| L2vFnWn5yt |                  170 |
| lA2l4dUPAN |                  265 |
| MBxLmTyDx0 |                  152 |
| nUFwyC0BAj |                  167 |
| nxtwKjphvt |                  260 |
| OLkEycvtOv |                   99 |
| p3f6twELII |                  203 |
```

```
| P9Vg4XQ5AK |                  267 |
| pIdz2ArSJd |                  168 |
| PWFixIVSN0 |                  197 |
| ql0bQqFgKx |                  224 |
| rdYk2JSFOZ |                  229 |
| sUUNXUZjSB |                  174 |
| TnFL7eVZD9 |                  246 |
| UKHmNCJ1Ep |                  211 |
| uQJBa7yRRm |                  240 |
| VXBvjILW4l |                  255 |
| VyLXDToji5 |                  186 |
| Xi0fOUOila |                  200 |
| YOpJVyms0T |                  151 |
| yz0xcnsOMA |                  212 |
+------------+----------------------+
50 rows in set (0.01 sec)
```

Explanation: This query is plausible because someone might want to check which designers were more prolific; this might inform their decision on whose furniture to buy.

This result is sensible, though somewhat unlikely in the real world; there are 50 designers and 1000 models, and each designer has designed about 200 models on average, so each model has about 10 designers. Further investigation into the data validated this calculation, so the query gave the intended result, even if that result was unintuitive.

(*B*) **Intent:** "What is the average lead time from suppliers to distribution centers when both are in the same country?"

Query:
```
SELECT AVG(leadTime)
FROM   Supplier S,
       DistributionCenter C,
       canOrderFrom O
WHERE  S.supplierID = O.supplierID
   AND C.centerID = O.centerID
   AND S.country = C.country;
```

Result:
```
+-------------------+
| AVG(leadTime)     |
+-------------------+
| 5.333333333333333 |
+-------------------+
1 row in set (0.00 sec)
```

Explanation: This query is plausible because someone might want to know how long, on average, they'd have to wait for furniture to arrive at the distribution centers after being ordered.

This result is sensible because it gives a single result to a query that contains only an aggregation function and no grouping.

(C) **Intent:** "Which suppliers can get me a claw-footed bedframe in less than a week?"

Query:

```
SELECT DISTINCT O.supplierID
FROM   Bedframe B,
       describes D,
       features_Feature F,
       canOrderFrom O,
       stocks S
WHERE  B.sku = D.sku
   AND D.modelNumber = F.modelNumber
   AND F.description = 'Claw Feet'
   AND O.centerID = S.centerID
   AND S.sku = B.sku
   AND O.leadTime < 7.0;
```

Result:

```
+------------+
| supplierID |
+------------+
| 45AVHG6SDL |
| dWuUGFOe1b |
| lGSxEQQ8bN |
| 4hGbj2aBVR |
| socFXUsXAD |
| 0x1V8UVLqh |
| umALkT56Xy |
| xUMVYgBI7J |
| FE34LTqb2p |
| ldbBahbqLj |
| ryNb8O1TRs |
| tRJBEnEFaO |
| ATeybFIuTQ |
| 7n94AZBOkB |
| Op8rarVKxa |
| FyAzhOpNly |
| WY3AmkQmxs |
| xyEInQvE1U |
| ZxsJeaFpg5 |
| 06DeQdaf4X |
| 5SKHZGJsjV |
| FVzpxLJFKK |
| uluou8izCd |
| nZ4msLXxTY |
| x7FolH1EsA |
| xJZiBD5DIo |
```

```
| cUrmX3GV4D |
| KQIGJ9eDk5 |
| 8pYnVWzpzR |
| 9Hzdyajzdu |
| 9kQZDUILDP |
| 3LnmekGoPa |
| 4xp0ETTkc8 |
| IBvWjkFney |
| XIFO3VhtQH |
+------------+
35 rows in set (0.00 sec)
```

Explanation: This query is plausible because a customer might very well want a bedframe—with particular attributes, even—within a specified time limit.

This result seems reasonable, because it feels fairly realistic that claw-footed bedframes wouldn't be particularly rare, so 35 of our 50 available suppliers carrying them seems fine. It should be noted that, the first time this query was run, we forgot the `DISTINCT` keyword after `SELECT`; this gave us 100 (obviously non-unique) results, and taught us a lesson about not assuming automatic distinctness in the result of a complex query just because the result column is a primary key in its original table.

(*D*) **Intent:** "How many items is my old buddy Harold stocking in that warehouse of his?"

Query:
```
SELECT COUNT(*)
FROM   stocks S,
       DistributionCenter D
WHERE  S.centerID = D.centerID
   AND D.name_ = 'Harold and His Big Ass Warehouse';
```

Result:
```
+----------+
| COUNT(*) |
+----------+
|        7 |
+----------+
1 row in set (0.00 sec)
```

Explanation: This query is plausible because somebody might actually care about Harold's business success (or lack thereof). We wanted to make a query that involved a specific name or attribute to make sure we could return small data as well as larger data.

Apparently Harold has a lot of wasted space right now; blame the economy.

(*E*) **Intent:** "Which sets have more than two chairs and at least one table?"

Query:
```
SELECT DISTINCT C.setID
FROM   contains_ C
```

```
WHERE  (SELECT COUNT(DISTINCT I.sku) * C.count_
        FROM   Table_ I,
               describes D
        WHERE  D.sku = I.sku
           AND D.modelNumber = C.modelNumber) >= 1
   AND (SELECT COUNT(DISTINCT I.sku) * C.count_
        FROM   Chair I,
               describes D
        WHERE  D.sku = I.sku
           AND D.modelNumber = C.modelNumber) > 2;
```

Result: `Empty set (0.00 sec)`

Explanation: This query is plausible because someone might want some dinner furniture in a professionally matched set.

I guess we don't carry any of that kind of set; oh well!

Group Work

Timothy: Checked the queries and results for sensibility.

Alexander: Ran the queries against the database and recorded their results.

Schuyler: Checked the queries and results for sensibility.