METIS

SQL

Queries and Aggregates

Creating a Table (Revisited)

```
CREATE TABLE inventory (
    id INTEGER PRIMARY KEY,
    name VARCHAR(100),
    quantity INTEGER,
    vendor_unit_price MONEY,
    last_shipment DATE,
    reorder BIT
);
```

Inserting Data into Tables

```
INSERT INTO inventory VALUES (1, 'tiger t-shirt', 10, 4.25,
    '2018-01-22', 'TRUE');
INSERT INTO inventory VALUES (2, 'giraffe-print bag', 18, 24.99,
    '2018-02-26', 'FALSE');
INSERT INTO inventory VALUES (3, 'elephant tie', 15, 13.19,
    '2018-02-26', 'FALSE');
INSERT INTO inventory VALUES (4, 'zebra-striped pants', 7,
     16.88, '2018-01-08', 'TRUE');
INSERT INTO inventory (id, name, quantity, reorder) VALUES (
       5, 'peacock feather hat', 2, 'FALSE');
INSERT INTO inventory (id, name, vendor_unit_price) VALUES (
       6, 'leopard-print scarf', 8.55);
```

Inventory Data Table

id	name	quantity	vendor_unit_price	last_shipment	reorder
1	tiger t-shirt	10	4.2500	2018-01-22	1
2	giraffe-print bag	18	24.9900	2018-02-26	0
3	elephant tie	15	13.1900	2018-02-26	0
4	zebra-striped pants	7	16.8800	2018-01-08	1
5	peacock feather hat	2	NULL	NULL	0
6	leopard-print scarf	NULL	8.5500	NULL	NULL

QUERIES I: SELECT and WHERE

METIS

Querying with Select Command

The SELECT command is used to retrieve (and display) data from tables

Wildcard: matches all columns in header

```
SELECT * FROM inventory;
```

id	name	quantity	vendor_unit_price	last_shipment	reorder
1	tiger t-shirt	10	4.2500	2018-01-22	1
2	giraffe-print bag	18	24.9900	2018-02-26	0
3	elephant tie	15	13.1900	2018-02-26	0
4	zebra-striped pants	7	16.8800	2018-01-08	1
5	peacock feather hat	2	NULL	NULL	0
6	leopard-print scarf	NULL	8.5500	NULL	NULL

Querying with Select Command

SELECT command to pull only columns of interest

```
SELECT name, vendor_unit_price, reorder FROM inventory;
```

name	vendor_unit_price	reorder
tiger t-shirt	4.2500	1
giraffe-print bag	24.9900	0
elephant tie	13.1900	0
zebra-striped pants	16.8800	1
peacock feather hat	NULL	0
leopard-print scarf	8.5500	NULL

Querying and Ordering

Sort by the `reorder` column and display in descending order

```
SELECT name, vendor_unit_price, reorder
    FROM inventory ORDER BY reorder DESC;
```

name	vendor_unit_price	reorder
tiger t-shirt	4.2500	1
zebra-striped pants	16.8800	1
peacock feather hat	NULL	0
giraffe-print bag	24.9900	0
elephant tie	13.1900	0
leopard-print scarf	8.5500	NULL

Querying and Filtering: WHERE

```
SELECT name, vendor_unit_price, reorder
   FROM inventory;
```

name	vendor_unit_price	reorder
tiger t-shirt	4.2500	1 ★
giraffe-print bag	24.9900	0
elephant tie	13.1900	0
zebra-striped pants	16.8800	1 ★
peacock feather hat	NULL	0
leopard-print scarf	8.5500	NULL

```
SELECT name, vendor_unit_price, reorder
   FROM inventory WHERE reorder=1;
```

name	vendor_unit_price	reorder
tiger t-shirt	4.2500	1
zebra-striped pants	16.8800	1

Querying and Filtering: WHERE

Question:
Of which products do we have at least 10 in stock and what are their unit prices?

- Interested in name, quantity, and vendor_unit_price columns
- Use WHERE clause to filter on quantity

```
SELECT name, quantity, vendor_unit_price
    FROM inventory
    WHERE quantity >= 10;
```

Querying and Filtering: WHERE

Question:
Of which products do we have at least 10 in stock and what are their unit prices?

```
SELECT name, quantity, vendor_unit_price
    FROM inventory
    WHERE quantity >= 10;
```

name	quantity	vendor_unit_price
tiger t-shirt	10	4.2500
giraffe-print bag	18	24.9900
elephant tie	15	13.1900

Querying and Filtering: WHERE, AND

Question:
Of which products do we have at least 10 in stock that cost more than $12 each?

```
SELECT name, quantity, vendor_unit_price
    FROM inventory
    WHERE quantity >= 10
    AND vendor_unit_price > 12;
```

WHERE clause with AND means both statements must be True!

name	quantity	vendor_unit_price
giraffe-print bag	18	24.9900
elephant tie	15	13.1900

Querying and Filtering: WHERE, OR

Question:
Which items are we set to reorder or have less than 5 in stock?

WHERE clause with OR means either statement must be True

```
SELECT name, reorder, quantity FROM inventory
  WHERE reorder='TRUE'
  OR quantity < 5;
```

name	reorder	quantity
tiger t-shirt	1	10
zebra-striped pants	1	7
peacock feather hat	0	2

Querying and Filtering

Retrieve all of the information about the tiger t-shirt, the elephant tie, and the zebra-striped pants.

```
SELECT * FROM inventory
    WHERE name='tiger t-shirt'
    OR name='elephant tie'
    OR name='zebra-striped pants';
```

id	name	quantity	vendor_unit_price	last_shipment	reorder
1	tiger t-shirt	10	4.2500	2018-01-22	1
3	elephant tie	15	13.1900	2018-02-26	0
4	zebra-striped pants	7	16.8800	2018-01-08	1

Querying and Filtering: IN

Use IN command instead of chaining multiple OR statements about same column

```
SELECT * FROM inventory
    WHERE name IN (
          'tiger t-shirt',
          'zebra-striped pants',
          'elephant tie',
          );
```

id	name	quantity	vendor_unit_price	last_shipment	reorder
1	tiger t-shirt	10	4.2500	2018-01-22	1
3	elephant tie	15	13.1900	2018-02-26	0
4	zebra-striped pants	7	16.8800	2018-01-08	1

Querying and Conditioning

Question:
Can we classify each item as high, medium, or low value according to table below?

VALUE	Vendor Unit Price
High	More than $20
Medium	Between $10 and $20
Low	$10 or Less

Querying and Conditioning: CASE

VALUE	Vendor Unit Price
High	More than $20
Medium	Between $10 and $20
Low	$10 or Less

```
SELECT name, vendor_unit_price,
    CASE
        WHEN vendor_unit_price > 20 THEN “high”
        WHEN vendor_unit_price > 10 THEN “medium”
        ELSE “low”
    END
FROM inventory;
```

Querying and Conditioning: CASE

name	vendor_unit_price	(No column name)
tiger t-shirt	4.2500	low
giraffe-print bag	24.9900	high
elephant tie	13.1900	medium
zebra-striped pants	16.8800	medium
peacock feather hat	NULL	low
leopard-print scarf	8.5500	low

Need to name column when creating it

NULL value follows the ELSE case

Querying and Conditioning: CASE

1. Include WHEN clause to catch NULL values
2. Alias created column with AS statement

```
SELECT name, vendor_unit_price,
    CASE
          WHEN vendor_unit_price IS NULL THEN NULL
        WHEN vendor_unit_price > 20 THEN “high”
        WHEN vendor_unit_price > 10 THEN “medium”
        ELSE “low”
    END   AS value_class
FROM inventory;
```

Querying and Conditioning: CASE

Resulting table has appropriate header and a NULL value for the hat

name	vendor_unit_price	value_class
tiger t-shirt	4.2500	low
giraffe-print bag	24.9900	high
elephant tie	13.1900	medium
zebra-striped pants	16.8800	medium
peacock feather hat	NULL	NULL
leopard-print scarf	8.5500	low

Note: Schema is unaffected by columns created with CASE.
No value_class column if run SELECT * from inventory query.

QUERIES II: TOP, LIKE, AS

METIS

Advanced Querying

On March 12th, company received shipment of animal themed home goods

```
INSERT INTO inventory VALUES (7, 'walrus-shaped pillow', 5, 12.25,
    '2018-03-12', 0);
INSERT INTO inventory VALUES (8, 'gazelle lamp', 3, 38.85,
    '2018-03-12', 0);
INSERT INTO inventory VALUES (9, 'bedding set, tiger icons', 5, 31.99,
    '2018-03-12', 0);
INSERT INTO inventory VALUES (10, 'wooly mammoth curtains', 4, 29.99,
    '2018-03-12', 0);
```

Advanced Querying

On March 12th, company received shipment of animal themed home goods

id	name	quantity	vendor_unit_price	last_shipment	reorder
1	tiger t-shirt	10	4.2500	2018-01-22	1
2	giraffe-print bag	18	24.9900	2018-02-26	0
3	elephant tie	15	13.1900	2018-02-26	0
4	zebra-striped pants	7	16.8800	2018-01-08	1
5	peacock feather hat	2	NULL	NULL	0
6	leopard-print scarf	NULL	8.5500	NULL	NULL
7	walrus-shaped pillow	5	12.2500	2018-03-12	0
8	gazelle lamp	3	38.8500	2018-03-12	0
9	bedding set, tiger icons	5	31.9900	2018-03-12	0
10	wooly mammoth curtains	4	29.9900	2018-03-12	0

Advanced Querying: TOP

Question:
Which 5 items have the highest unit prices? How many of each do we have in stock?

```
SELECT TOP 5 id, name, quantity, vendor_unit_price
    FROM inventory
    ORDER BY vendor_unit_price DESC;
```

id	name	quantity	vendor_unit_price
8	gazelle lamp	3	38.8500
9	bedding set, tiger icons	5	31.9900
10	wooly mammoth curtains	4	29.9900
2	giraffe-print bag	18	24.9900
4	zebra-striped pants	7	16.8800

Advanced Querying: TOP and AS

Question:
For tax purposes which 5 items do we have the most money tied up in?

```
SELECT TOP 5 id, name, quantity, vendor_unit_price,
    quantity*vendor_unit_price AS total_inv_value
    FROM inventory
    ORDER BY total_inv_value DESC;
```

id	name	quantity	vendor_unit_price	total_inv_value
2	giraffe-print bag	18	24.9900	449.8200
3	elephant tie	15	13.1900	197.8500
9	bedding set, tiger icons	5	31.9900	159.9500
10	wooly mammoth curtains	4	29.9900	119.9600
4	zebra-striped pants	7	16.8800	118.1600

Advanced Querying: LIKE

Question:
Do we have anything in stock that features a tiger?

WILDCARDS
% – percent sign matches 0, 1, or multiple characters
_ – underscore matches exactly 1 character

```
SELECT * FROM inventory
    WHERE name LIKE '%tiger%';
```

% on either side of the word tiger means name may or may not have more text before or after

Advanced Querying: LIKE

Question:
Do we have anything in stock that features a tiger?

```
SELECT * FROM inventory
    WHERE name LIKE '%tiger%';
```

id	name	quantity	vendor_unit_price	last_shipment	reorder
1	tiger t-shirt	10	4.2500	2018-01-22	1
9	bedding set, tiger icons	5	31.9900	2018-03-12	0

Advanced Querying: LIKE

Question:
Do we have anything in stock that features a tiger or was recently restocked?

```
SELECT * FROM inventory
    WHERE name LIKE '%tiger%'
    OR last_shipment >= '2018-03-01';
```

id	name	quantity	vendor_unit_price	last_shipment	reorder
1	tiger t-shirt	10	4.2500	2018-01-22	1
7	walrus-shaped pillow	5	12.2500	2018-03-12	0
8	gazelle lamp	3	38.8500	2018-03-12	0
9	bedding set, tiger icons	5	31.9900	2018-03-12	0
10	wooly mammoth curtains	4	29.9900	2018-03-12	0

AGGREGATES: GROUP BY & HAVING

METIS

Aggregating Data: SUM

Question:
What is the total value of all the inventory the company has?

```
SELECT SUM(quantity*vendor_unit_price) AS total_inv_value
    FROM inventory;
```

total_inv_value
1266.0400

According to the 10 items in the data table, my company has $1,266.04 of inventory in stock.

Aggregating Data: AVG

Question:
For all the items the company has, what is the average quantity in stock?

```
SELECT AVG(quantity) AS avg_quantity
    FROM inventory;
```

avg_quantity
7

```
SELECT AVG(CAST(quantity AS FLOAT))
    AS avg_quantity_float FROM inventory;
```

avg_quantity_float
7.66666666666667

Note: Aggregate functions assume the data type of column being analyzed.

Aggregating Data List

AVG
CHECKSUM_AGG
COUNT
COUNT_BIG
GROUPING
GROUPING_ID
MAX
MIN
SUM
STDEV
STDEVP
STRING_AGG
VAR
VARP

Grouping and Aggregating Data

Another associated data table to describe type, material, and sales price of items

```
CREATE TABLE inventory (
    item_id INTEGER PRIMARY KEY,
    name VARCHAR(100),
    department VARCHAR(100),
    material VARCHAR(100),
    sales_price MONEY
);
```

Grouping and Aggregating Data

Another associated data table to describe type, material, and sales price of items

item_id	name	department	material	sales_price
1	tiger t-shirt	clothing	cotton blend	9.9900
2	giraffe-print bag	accessories	canvas	49.9900
3	elephant tie	accessories	silk	35.4900
4	zebra-striped pants	clothing	silk	30.9900
5	peacock feather hat	accessories	felt	34.9900
6	leopard-print scarf	accessories	silk	14.4900
8	gazelle lamp	home goods	metal	79.9900
9	bedding set, tiger icons	home goods	cotton blend	69.9900
11	aardvark earrings	accessories	metal	9.9900

Grouping and Aggregating Data

Question:
What is the average sales price for each department?

item_id	name	department	material	sales_price
1	tiger t-shirt	clothing	cotton blend	9.9900
2	giraffe-print bag	accessories	canvas	49.9900
3	elephant tie	accessories	silk	35.4900
4	zebra-striped pants	clothing	silk	30.9900
5	peacock feather hat	accessories	felt	34.9900
6	leopard-print scarf	accessories	silk	14.4900
8	gazelle lamp	home goods	metal	79.9900
9	bedding set, tiger icons	home goods	cotton blend	69.9900
11	aardvark earrings	accessories	metal	9.9900

Two items in clothing department

Two items in home goods department

Grouping and Aggregating Data

Question:
What is the average sales price for each department?

Note: Selected columns must either be aggregate functions or included in GROUP BY clause.

```
SELECT department, AVG(sales_price) AS avg_price
    FROM item_details
    GROUP BY department;
```

department	avg_price
accessories	28.9900
clothing	20.4900
home goods	74.9900

Grouping and Aggregating Data

Question:
What is the least expensive item made from each material in stock, ordered by price?

```
SELECT material,
    MIN(sales_price) AS min_price
    FROM item_details
    GROUP BY material
    ORDER BY min_price;
```

material	min_price
cotton blend	9.9900
metal	9.9900
silk	14.4900
felt	34.9900
canvas	49.9900

Filtering Aggregated Data: HAVING

Question:
What is the least expensive item made from each material that starts with the letter ‘c’?

```
SELECT material,
    MIN(sales_price) AS min_price
    FROM item_details
    GROUP BY material
    HAVING material LIKE ‘c%’;
```

material	min_price
canvas	49.9900
cotton blend	9.9900

Notes: HAVING clause filters groups; WHERE clause filters rows.
HAVING goes after the GROUP BY clause; WHERE goes after the table name.

Filtering Data: WHERE vs HAVING

Which products are made from cotton?	**WHERE**
What is the average price of each department where all items in that department cost $50 or less?	**HAVING**
How many items are in the accessories department?	**WHERE** or **HAVING**

Filtering Data: WHERE vs HAVING

Question:
What is the average price of each department where all items in that department cost $50 or less?

```
SELECT department,
    AVG(sales_price) AS avg_price
    FROM item_details
    GROUP BY department
    HAVING MAX(sales_price) <= 50;
```

department	avg_price
accessories	28.9900
clothing	20.4900

Note: HAVING clause will not recognize the alias “avg_price”; GROUP BY and HAVING occur before alias assigned.

Filtering Data: WHERE vs HAVING

Question:
How many items are in the accessories department?

WHERE

```
SELECT department,
    COUNT(name) AS item_count
    FROM item_details
    WHERE department='accessories'
    GROUP BY department;
```

HAVING

```
SELECT department,
    COUNT(name) AS item_count
    FROM item_details
    GROUP BY department
    HAVING department='accessories';
```

department	item_count
accessories	5

What are the two types of wildcards?

What is the CASE syntax?

What’s the difference between HAVING and WHERE?

QUESTIONS?