Name

UIN

Midterm Exam

March 11th, 2026
11:00am - 12:15pm

CS480 - Database Systems

Results

Please leave this empty!

1.1 1.2 1.3 1.4 Sum

Instructions

- Try to answer all the questions using what you have learned in class. Keep hard questions until the end.
- **When writing a query, write the query in a way that it would work over all possible database instances and not just for the given example instance!**
- The exam is closed book and closed notes!
- **For relational algebra questions assume set semantics!**

Consider the following database schema and example instance for a orders database:

customer

cid	name	country	region
1	Bob Bobsen	USA	North America
2	Peter Petersen	Canada	North America
3	Ali Alisen	USA	North America
4	Alice Alicesen	China	Asia
5	Sheila Sheilasen	Netherlands	Europe

order

oid	cid	orderdate	shipdate	paid
1	1	2026-01-02	2026-01-19	1
2	1	2026-01-05	NULL	1
3	2	2026-02-03	2026-02-05	0

item

iid	iname	price
1	rock	0.33
2	scissor	0.90
3	paper	0.45

lineitem

oid	pos	iid	quantity
1	1	2	3
1	2	3	1
2	1	1	14
3	1	1	1
3	2	3	2

Hints:

- Attributes with black background form the primary key of a relation (.e.g, `cid` for relation `customer`)
- The attribute `oid` of relation `lineitem` is a foreign key to relation `order`
- The attribute `iid` of relation `lineitem` is a foreign key to relation `item`
- The attribute `cid` of relation `order` is a foreign key to relation `customer`
- All foreign keys have been created with the **CASCADE** option.

Part 1.1 Relational Algebra (Total: 28 Points)

Question 1.1.1 (8 Points)

Write a **relational algebra** expression that returns the `name` of customers from `region` *"North America"* who have orders where a lineitem in the order is item *rock* with at least a quantity of 5.

Solution

$$\pi_{name}(\sigma_{region=North\ America}(customer) \bowtie order \bowtie \sigma_{quantity=5}(lineitem) \bowtie \sigma_{iname=Rock}(item))$$

Question 1.1.2 (10 Points)

Write a **relational algebra** expression that returns the `oid` and `orderdate` of orders that contain both item *rock* and item *paper*.

Solution

$$Q_{orderitem} \leftarrow \pi_{oid,iname}(order \bowtie lineitem \bowtie item)$$
$$Q \leftarrow \pi_{oid,orderdate}(order \bowtie \pi_{oid}(\sigma_{iname=rock}(Q_{orderitem})) \bowtie \pi_{oid}(\sigma_{iname=paper}(Q_{orderitem})))$$

Question 1.1.3 (10 Points)

Write a **relational algebra** expression that returns the average shipping delay (`shipdate` minus `orderdate`) of orders.

Solution

$$Q_{odelay} \leftarrow \rho_{delay}(\pi_{shipdate-orderdate}(order)$$
$$Q \leftarrow \gamma_{avg(delay)}(Q_{odelay})$$

Part 1.2 SQL - DDL (Total: 12 Points)

Question 1.2.1 (12 Points)

Write an **SQL DDL statement** that creates a new relation `supplier` that records which `supplier` is supplying which item (`iid`) at what `price`.

- The combination of `supplier` and `iid` should be unique.
- We need to ensure that every entry in relation `supplier` corresponds to an item in relation `item`.
- `Prices` need to be positive

Solution

```
CREATE TABLE supplier (
  supplier TEXT,
  iid INT REFERENCES item,
  price NUMERIC(12,2) CHECK (price > 0),
  PRIMARY KEY (supplier, iid)
);
```

Part 1.3 SQL - Queries (Total: 40 Points)

Question 1.3.1 (11 Points)

Write an **SQL** query that returns the `name` and `country` of customers from `region` "North America" which have orders that were shipped already, but the order is not paid yet. **Make sure to return each such customer only once.**

Solution

```
SELECT DISTINCT name, country
FROM customer c
     NATURAL JOIN
     order o
WHERE region = 'North America'
      AND shipdate IS NOT NULL
      AND paid = 0
```

Question 1.3.2 (14 Points)

Write an **SQL query** that returns the *total account balance* of each customer. The result schema should be `name` and `balance`. The account balance of a customer is the total cost of all orders the customer has placed that have not been paid yet. The cost of an order is the sum of `price` of items in the order (taking the quantity into account). For example, order `oid=1` has a total cost of:

$$3 \times 0.90 + 1 \times 0.45 = 3.15$$

Customers without such orders should be returned with a 0 balance.

Solution

```
WITH ordercost AS (
  SELECT oid, cid, sum(price * quantity) AS cost
    FROM order NATURAL JOIN customer NATURAL JOIN item NATURAL JOIN lineitem
   WHERE paid = 0
   GROUP BY oid, cid
)
SELECT name, COALESCE(balance,0) AS balance
  FROM customer c
       LEFT OUTER JOIN
       (SELECT cid, sum(cost) AS balance
          FROM ordercost
         GROUP BY cid) b
           ON (c.cid = b.cid);
```

Question 1.3.3 (15 Points)

Write an **SQL query** that returns the `oid`, `name` of the ordering customer, and `delay` of orders whose delay (`shipdate` minus `orderdate`) is greater than the average delay of orders with the same paid status (same value of the `paid` attribute).

Solution

```
WITH odelay AS (
  SELECT oid,
         cid, orderdate - shipdate AS delay, paid
)
SELECT oid, name, delay
  FROM odelay o NATURAL JOIN customer c
 WHERE paid = 1
   AND delay > (SELECT avg(delay)
                  FROM odelay o2
                 WHERE o.paid = o2.paid);
```

Part 1.4 SQL - Updates (Total: 20 Points)

Question 1.4.1 (9 Points)

Write an **SQL statement** that updates the `shipdate` of all orders that have not been shipped yet (shipdate is `NULL`), but have been `paid`. For these orders set the shipdate to `2026-03-12`.

Solution

```
UPDATE order
SET shipdate = '2026-03-12'
WHERE shipdate IS NULL
      AND paid = 1;
```

Question 1.4.2 (11 Points)

Write an **SQL statement** that inserts a new order for customer "Ali Alisen" with an `orderdate` of `2026-03-11` that has not been shipped yet, but was paid already. The order's `oid` should be the next available id, determined from the data as part of your query. Both the new `oid` and "Ali Alisen"'s `cid` should be determined dynamically as part of your query instead of hardcoding them based on the provided example data.

Solution

```
INSERT INTO order
VALUES (
  (SELECT max(oid) + 1 AS oid FROM order),
  (SELECT cid FROM custoemr WHERE name = 'Ali␣Alisen'),
  '2026-03-11'::date,
  NULL,
  1
)
```