SQL CODE FOR EACH TABLE

1. admins

```
CREATE TABLE admins (
  id INT(11) NOT NULL AUTO_INCREMENT COMMENT 'Primary key',
  username VARCHAR(50) NOT NULL COMMENT 'Admin username (unique)',
  password VARCHAR(255) NOT NULL COMMENT 'Hashed admin password',
  PRIMARY KEY (id),
  UNIQUE KEY (username)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

2. members

```
CREATE TABLE members (
  id INT(11) NOT NULL AUTO_INCREMENT COMMENT 'Primary key for members',
  username VARCHAR(255) NOT NULL COMMENT 'Member username (unique)',
  email VARCHAR(255) NOT NULL COMMENT 'Member email (unique)',
  password VARCHAR(255) NOT NULL COMMENT 'Hashed member password',
  created_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP COMMENT
'Timestamp when account was created',
  status ENUM('enabled','disabled') DEFAULT 'enabled' COMMENT 'Account status',
  PRIMARY KEY (id),
  UNIQUE KEY (username),
  UNIQUE KEY (email)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

3. custom_quotes

```
CREATE TABLE custom_quotes (
  id INT(11) NOT NULL AUTO_INCREMENT COMMENT 'Primary key',
  member_id INT(11) NOT NULL COMMENT 'Link to members(id)',
  gpu_id INT(11) NOT NULL COMMENT 'Potentially links to products(id) or a dedicated GPU
table',
  addons TEXT DEFAULT NULL COMMENT 'List or JSON structure of optional add-ons
selected',
  total DECIMAL(10,2) NOT NULL COMMENT 'Total cost of this custom quote',
  created_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP COMMENT
'Creation timestamp',
  PRIMARY KEY (id),
  KEY member_id (member_id),
  CONSTRAINT fk_custom_quotes_member
    FOREIGN KEY (member_id)
    REFERENCES members (id)
    ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

4. login_history

```
CREATE TABLE login_history (
  id INT(11) NOT NULL AUTO_INCREMENT COMMENT 'Primary key',
  member_id INT(11) NOT NULL COMMENT 'References members(id)',
  event_type ENUM('login','logout') NOT NULL COMMENT 'Type of event',
  event_time TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP COMMENT
'Time of login or logout event',
  ip_address VARCHAR(45) DEFAULT NULL COMMENT 'IP address from which the event
originated',
  PRIMARY KEY (id),
  KEY member_id (member_id),
  CONSTRAINT fk_login_history_member
    FOREIGN KEY (member_id)
    REFERENCES members (id)
    ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

5. products

```
CREATE TABLE products (
  id INT(11) NOT NULL AUTO_INCREMENT COMMENT 'Primary key',
  name VARCHAR(255) NOT NULL COMMENT 'Name of the product',
  description TEXT DEFAULT NULL COMMENT 'Detailed description of the product',
  price DECIMAL(10,2) NOT NULL COMMENT 'Price of the product',
  image VARCHAR(255) DEFAULT NULL COMMENT 'Path or filename of product image',
  rating FLOAT DEFAULT 0 COMMENT 'Average rating',
  created_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP COMMENT
'When product was added',
  PRIMARY KEY (id)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

6. orders

```
CREATE TABLE orders (
  id INT(11) NOT NULL AUTO_INCREMENT COMMENT 'Primary key',
  user_id INT(11) DEFAULT NULL COMMENT 'References members(id). The buyer placing
the order',
  product_id INT(11) DEFAULT NULL COMMENT 'OPTIONAL single-product reference (may
be unused in multi-item designs)',
  total DECIMAL(10,2) NOT NULL COMMENT 'Order total',
  shipping_address VARCHAR(255) DEFAULT NULL COMMENT 'Where to ship to',
  status VARCHAR(50) DEFAULT NULL COMMENT 'Order status e.g. pending, shipped, etc.',
  created_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP COMMENT
'When the order was created',
  delivery_days INT(11) NOT NULL DEFAULT 1 COMMENT 'Days estimated for delivery',
  PRIMARY KEY (id)
  -- Optionally, you could add a foreign key to members if you want:
  -- , CONSTRAINT fk_orders_member
  --    FOREIGN KEY (user_id)
  --    REFERENCES members(id)
  --    ON DELETE SET NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

7. order_items

```
CREATE TABLE order_items (
  id INT(11) NOT NULL AUTO_INCREMENT COMMENT 'Primary key',
  order_id INT(11) DEFAULT NULL COMMENT 'References orders(id)',
  product_id INT(11) DEFAULT NULL COMMENT 'References products(id)',
  quantity INT(11) DEFAULT NULL COMMENT 'Number of this product in the order',
  price DECIMAL(10,2) DEFAULT NULL COMMENT 'Captured price of the product for this
line item',
  PRIMARY KEY (id),
  KEY product_id (product_id),
  KEY order_id (order_id),
  CONSTRAINT fk_order_items_order
    FOREIGN KEY (order_id)
    REFERENCES orders (id)
    ON DELETE CASCADE,
  CONSTRAINT fk_order_items_product
    FOREIGN KEY (product_id)
    REFERENCES products (id)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

8. questions

```
CREATE TABLE questions (
  id INT(11) NOT NULL AUTO_INCREMENT COMMENT 'Primary key',
  subject VARCHAR(255) NOT NULL COMMENT 'Brief subject of question',
  member_id INT(11) NOT NULL COMMENT 'References members(id)',
  product_id INT(11) DEFAULT NULL COMMENT 'References products(id), if question is
product-related',
  user_email VARCHAR(255) DEFAULT NULL COMMENT 'Contact email if needed
(optional)',
  question TEXT DEFAULT NULL COMMENT 'User-submitted question text',
  answer TEXT DEFAULT NULL COMMENT 'Admin or system answer to the question',
  created_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP COMMENT
'When the question was created',
  PRIMARY KEY (id)
  -- Optionally add:
  -- , KEY member_id (member_id),
  -- , KEY product_id (product_id),
  -- FOREIGH KEY definitions if you want strict referential integrity
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

9. reviews

```
CREATE TABLE reviews (
  id INT(11) NOT NULL AUTO_INCREMENT COMMENT 'Primary key',
  member_id INT(11) NOT NULL COMMENT 'References members(id). Who left the review',
  rating INT(11) NOT NULL COMMENT 'Numeric rating (e.g. 1-5)',
  comment TEXT DEFAULT NULL COMMENT 'Reviewer comments',
  created_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP COMMENT
'When the review was submitted',
  product_id INT(11) NOT NULL COMMENT 'References products(id). Which product is being
reviewed',
  PRIMARY KEY (id),
  KEY member_id (member_id)
  -- Optionally:
  -- , CONSTRAINT fk_reviews_member
  --    FOREIGN KEY (member_id)
  --    REFERENCES members (id)
  --    ON DELETE RESTRICT
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

10. rma_requests

```
CREATE TABLE rma_requests (
  id INT(11) NOT NULL AUTO_INCREMENT COMMENT 'Primary key',
  member_id INT(11) NOT NULL COMMENT 'References members(id). Requesting user',
  order_item_ids TEXT NOT NULL COMMENT 'IDs from order_items table (stored as text).
Not strictly normalized',
  issue_type VARCHAR(50) NOT NULL COMMENT 'Short name for the issue type, e.g.
Defective, Refund, etc.',
  issue_description TEXT NOT NULL COMMENT 'Detailed description of the issue',
  attached_document VARCHAR(255) NOT NULL COMMENT 'Path/filename if a document is
attached (e.g. proof of purchase)',
  created_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP COMMENT
'When the RMA was filed',
  admin_response TEXT DEFAULT NULL COMMENT 'Admin resolution or comment',
  PRIMARY KEY (id),
  KEY member_id (member_id)
  -- Optionally, you could add:
  -- , CONSTRAINT fk_rma_requests_member
  --    FOREIGN KEY (member_id)
  --    REFERENCES members (id)
  --    ON DELETE RESTRICT
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

11. stock

```
CREATE TABLE stock (
  id INT(11) NOT NULL AUTO_INCREMENT COMMENT 'Primary key',
  product_id INT(11) NOT NULL COMMENT 'References products(id)',
  quantity INT(11) NOT NULL DEFAULT 0 COMMENT 'Current quantity on-hand',
  updated_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP
    COMMENT 'Timestamp of last update',
  PRIMARY KEY (id),
  UNIQUE KEY (product_id),
  CONSTRAINT fk_stock_product
    FOREIGN KEY (product_id)
    REFERENCES products (id)
    ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

EXPLANATION OF THE TABLES

1. admins

Stores administrative users who can log into an admin console or back end.

id: Primary key.

username: Unique admin username.

password: Hashed password.

2. members

Holds registered user accounts/customers.

id: Primary key.

username: Unique username for login.

email: Unique email address.

password: Hashed password.

created_at: Timestamp the account was created.

status: Account status, e.g. `enabled` or `disabled`.

3. custom_quotes

Allows a member to generate a custom GPU quote (or a specialized product quote) with optional add-ons.

id: Primary key.

member_id: References the `members.id`. If the member is deleted, associated custom quotes are removed (`ON DELETE CASCADE`).

gpu_id: Intended to reference a GPU product. Could be turned into a foreign key to `products.id`.

addons: Flexible text field or JSON describing additional requested items.

total: The total price for this quote.

created_at: Creation timestamp.

4. login_history

Tracks each login or logout event for an individual member.

id: Primary key.

member_id: Foreign key to `members.id`.

event_type: An enum: `login` or `logout`.

event_time: Timestamp of the event.

ip_address: IP address captured at event time.

5. products

Holds all products sold through the site (mostly GPUs in this dump).

id: Primary key.

name: Name of the product.

description: Text description (specifications, marketing copy, etc.).

price: Product price.

i**mage**: Path or filename for the product's image.

rating: (Optional) a float average rating.

created_at: Timestamp the record was created.

6. orders

Stores overall order data from members. Sometimes references a single product (`product_id`), though multi-product orders typically rely on `order_items` for each line item.

id: Primary key.

user_id: The member placing the order (could add a foreign key to `members.id`).

product_id: A single product field (may be bypassed if you prefer all line items in `order_items`).

total: The entire order's total.

shipping_address: Where the order should be shipped.

status: e.g., “pending,” “shipped,” “refurbished,” etc.

created_at: Timestamp when the order was created.

delivery_days: Estimated number of days to deliver.

7. order_items

Line-item details for each `order`.

id: Primary key.

order_id: References `orders.id`. On delete cascade (removing an order removes the associated items).

product_id: References `products.id`.

quantity: How many units of the product.

price: The price for that product line item at the time of purchase.

8. questions

Members can submit questions, potentially about a product or general inquiry.

id: Primary key.

subject: High-level topic.

member_id: The member asking the question.

product_id: (Optional) If question relates to a specific product.

user_email: Email for the user (if not the same as the member’s stored email).

question: The question text.

answer: Admin or system’s response.

created_at: Timestamp of question creation.

9. reviews

Stores ratings and textual reviews for products, made by members.

id: Primary key.

member_id: The member who wrote the review. Could be a foreign key to `members.id`.

rating: The star rating or numeric rating, e.g. 1–5.

comment: Freeform text describing the user's feedback.

created_at: Timestamp of submission.

product_id: The product being reviewed.

10. rma_requests

Tracks return merchandize authorization (RMA) requests for defective or unwanted items.

id: Primary key.

member_id: The member requesting the RMA. Could reference `members.id`.

order_item_ids: A text field listing relevant `order_items.id` for the item(s) they want to return. (Not fully normalized, but flexible for grouping items in one request.)

issue_type: e.g. "Defective," "Wrong item," etc.

issue_description: Detailed text describing the problem.

attached_document: If the user or admin attached a proof-of-purchase or other doc.

created_at: When the RMA was requested.

admin_response: RMA status or the admin's official response.

11- stock

Tracks real-time inventory levels for products.

id: Primary key.

product_id: References `products.id` (unique). If the product is deleted, stock row is also deleted.

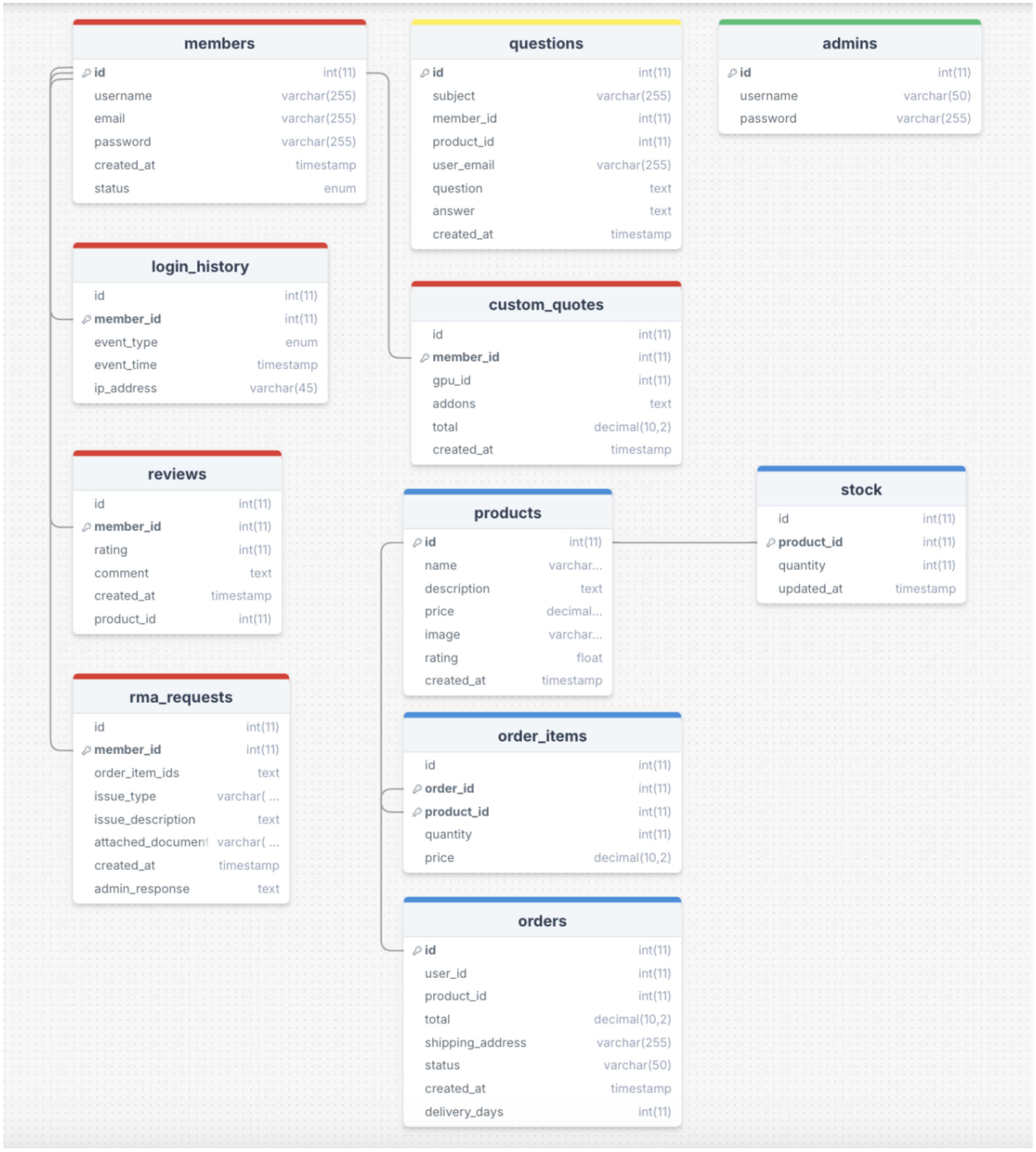
quantity: How many units remain in inventory.

updated_at: Timestamp automatically updated on each row change.

VISUAL PRESENTATION (SQL DIAGRAM)

You can access this diagram here:

https://drawsql.app/teams/university-of-windsor-2/diagrams/final-project



members	
id	int(11)
username	varchar(255)
email	varchar(255)
password	varchar(255)
created_at	timestamp
status	enum

questions	
id	int(11)
subject	varchar(255)
member_id	int(11)
product_id	int(11)
user_email	varchar(255)
question	text
answer	text
created_at	timestamp

admins	
id	int(11)
username	varchar(50)
password	varchar(255)

login_history	
id	int(11)
member_id	int(11)
event_type	enum
event_time	timestamp
ip_address	varchar(45)

custom_quotes	
id	int(11)
member_id	int(11)
gpu_id	int(11)
addons	text
total	decimal(10,2)
created_at	timestamp

reviews	
id	int(11)
member_id	int(11)
rating	int(11)
comment	text
created_at	timestamp
product_id	int(11)

stock	
id	int(11)
product_id	int(11)
quantity	int(11)
updated_at	timestamp

products	
id	int(11)
name	varchar...
description	text
price	decimal...
image	varchar...
rating	float
created_at	timestamp

rma_requests	
id	int(11)
member_id	int(11)
order_item_ids	text
issue_type	varchar(...
issue_description	text
attached_document	varchar(...
created_at	timestamp
admin_response	text

order_items	
id	int(11)
order_id	int(11)
product_id	int(11)
quantity	int(11)
price	decimal(10,2)

orders	
id	int(11)
user_id	int(11)
product_id	int(11)
total	decimal(10,2)
shipping_address	varchar(255)
status	varchar(50)
created_at	timestamp
delivery_days	int(11)