Phase 2 Report | CS 6400 - Spring 2024 | **Team 008**

Table of Contents

Register

Abstract Code with SQL

- User enters *username*, *password*, *first name*, and *last name* in the appropriate input fields. These must all be string data types
- If data validation is successful for all input fields, then:
- When ***Register*** button is clicked:

```
SELECT username
FROM "User"
WHERE username = $username;
```

    - If User record is found:
        - Go back to Register form, with the error message "Username exists!".
    - Else:
        - Insert an entry into User with the input fields.

```
INSERT INTO "User" (username, password, first_name, last_name)
VALUES ($username, $password, $firstName, $lastName);
```

        - Go to **Login** form.

- Else if any input fields are invalid, display **Register** form, with error message

Log In

Abstract Code with SQL

- User enters the appropriate information in the *username* and *password* input fields. These entries must be string data types.
- If data validation is successful for both *username* and *password* input fields, then:
  - When ***Log in*** button is clicked:

```
SELECT username, password
FROM "User"
WHERE username = $username;
```

    - If User record is not found or password is not valid:
      - Go back to **Login** form, with the error message "Username or password incorrect!"
    - Else:
      - Store login information as session userId.
      - Go to the **BuzzBid Main Menu** form.

- Else *username* and *password* input fields are invalid, display **Login** form, with error message.

Main Menu

Abstract Code with SQL

- Show Auction Options for all users, and Reports only if the current user is the admin user

```
SELECT first_name,
       last_name,
       position
FROM "User"
WHERE username = $username;
```

- Upon:
  - Click ***Search for Items*** link: Jump to the **Search For Item** task
  - Click ***List Item*** button: Jump to the **List Item** task
  - Click ***View Auction Results*** link: Jump to the **Auction Results** task
  - Click ***Category Report*** button: Jump to the **Category Report** task
  - Click ***User Report*** button: Jump to the **User Report** task
  - Click ***Top Rated Items*** button: Jump to the **Top Rated Items Report** task
  - Click ***Auction Statistics*** button: Jump to the **Auction Statistics Report** task

- Click ***Canceled Auction Details*** button: Jump to the **Canceled Auction Details** task

List Item

Abstract Code with SQL

- Upon user clicking on ***List Item*** from **BuzzBid Main Menu**:
  - Display **New Item** form
  - User can then:
    - Fills in the *Item Name* (string), *Description* (string), *Category* (select), *Condition* (select), *Starting Price* (float), *Minimum Sale Price* (float), *Auction End* (select), *Get It Now Price* (float), and *Returns Accepted* (boolean) fields, then click the ***List My Item*** button. All fields must be filled - display error message "All fields must be entered" if any one is empty. Fields such as Get It Now must be greater than the minimum sale price, if entered; otherwise, display error message "Get It Now price must be greater than minimum sale price". Starting bid must be less than the Get It Now Price, if entered, and greater than $0; otherwise, display error message "Starting bid must be less than Get It Now Price (if entered) and must be greater than $0"

```
INSERT INTO Item (item_id, username,  item_name, description,
is_returnable, condition, category_id)
VALUES (nextval('item_id_seq'), $username, $itemName,
$description, $isReturnable, $condition, $categoryId);

INSERT INTO Auction (auction_id, item_id, auction_end_time,
auction_length, get_it_now_price, min_sale_price, starting_bid,
cancel_reason, cancelled_by)
VALUES (nextval('auction_id_seq'), $auctionEndTime,
$auctionLength, $getItNowPrice, $minSalePrice, $startingBid, null,
null);
```

    - Or, Click on the ***Cancel*** button to close the **New Item** form
  - Upon listing an item, user is taken to the **Item For Sale** form, where the item information is displayed; run **View Item** task
  - Upon clicking on ***Cancel***, the user is taken back to the **BuzzBid Main Menu** form

Search For Item

Abstract Code with SQL

- Upon user clicking on ***Search for Items*** from **BuzzBid Main Menu**
- Search items based on several fields including *keyword* (string), *category* (select), *minimum price* (float), *maximum price* (float), and *condition at least* (select)
- Validate that each field has the appropriate type, such as *minimum price* and *maximum price* containing numerical input; display error message with validation error if appropriate.
- Upon clicking ***Search*** button, display items that meet the supplied criteria in the **Search Results** form

```
WITH current_bidder AS (SELECT DISTINCT ON (b.auction_id)
                                b.auction_id,
                                b.bid_amount,
                                b.username
                        FROM Bid b
                        JOIN Auction a on b.auction_id = a.auction_id
                        WHERE a.auction_end_time > now()
                        AND a.cancelled_by IS NULL
                        ORDER BY 1, 2 DESC)
SELECT i.item_id,
       i.item_name,
       c.bid_amount,
       c.username,
       a.get_it_now_price,
       a.auction_end_time
FROM Item i
JOIN Auction a ON i.item_id = a.item_id
JOIN Category cat ON i.category_id = cat.category_id
LEFT JOIN current_bidder c ON c.auction_id = a.auction_id
WHERE a.auction_end_time > now()
AND a.cancelled_by IS NULL
AND ($keyword IS NULL OR i.item_name ~ $keyword)
AND ($keyword IS NULL OR i.description ~ $keyword)
AND ($category IS NULL OR cat.category = $category)
AND ($minSalePrice IS NULL OR a.min_sale_price >= $minSalePrice)
AND ($maxSalePrice IS NULL OR c.bid_amount <= $maxSalePrice)
AND ($condition IS NULL OR i.condition <= $condition);
```

- Upon clicking ***Cancel*** button, return to **BuzzBid Main Menu**

View Item

Abstract Code with SQL

- User clicked on item name from **Item Results**:
- Query information about the item, the details of the auction, and the bids on the item (if any)
    - Find the current item using the item's ID; display the item's information including Item ID, Item Name, Description, Category, Condition, and Returns Accepted.

```
SELECT i.item_id,
        i.item_name,
        i.description,
        c.category,
        i.condition,
        a.is_returnable,
        a.get_it_now_price,
        a.auction_end_time,
        a.starting_bid
FROM Item i
JOIN Auction a ON i.item_id = a.item_id
JOIN Category c ON i.category_id = c.category_id
WHERE i.item_id = $itemId;
```

    - Display a link - ***Edit Description*** if the current user is the owner of the item, i.e., when current item.Username = current user.Username (obtained from session variable). ***Cancel*** button is also visible for Administrator users and for the user that listed the item.
    - Find the Auction associated with the item using the item's ID and the username of the user who listed the item; display the auction's information including Get It Now Price, Auction End, Minimum Sale Price, and Starting Bid fields.Display ***Get It Now!*** button when GetItNow price is not null, and the GetItNow price value.
    - Display "Your bid" and an input area for the current user to enter bid price, with Minimum Sale Price displayed below the input area.
    - For each Bid on the Auction:
        - Display the bid's information in Bid Amount, Time of Bid, and the Username user who made the bid in their respective fields
        - Bids are displayed ordered by the latest time of bid
        - If no bids yet (the result of the query below is Null) or the item has been canceled (CancelledBy has a value), the bid information will be empty. No need to query for bid information if the item was canceled.

```
SELECT b.bid_amount,
       b.bid_time,
       b.username
FROM Bid b
JOIN Auction a ON b.auction_id = a.auction_id
WHERE a.auction_id = $auctionId
ORDER BY 2 DESC
LIMIT 4;
```

- Upon:
  - Click ***Close*** button: Jump to the **Search For Item** task
  - Click ***Edit Description*** link: Jump to the **Edit Item** task
  - Click ***View Ratings*** link: Jump to the **View Rating** task
  - Click ***Get It Now*** button: Jump to the **Buy Item** task
  - Click ***Bid On This Item*** button: Jump to the **Buy Item** task
  - Click ***Cancel This Auction*** button: Jump to the **Cancel Item** task

Edit Item

Abstract Code with SQL

- User that listed the item clicked on the ***Edit Description*** button:
  - Display pop-up allowing user to edit current item's description, which must be string data type. As specified in the prior View Item section, it will be displayed only if the current user is the owner of the current item.
    - When Edit Description is clicked, first populate the existing description:

```
SELECT Description
FROM Item
WHERE item_id = $itemId;
```

    - When ***Save*** button is clicked:
      - Validate that input is a string
        - If not valid, display error message
      - Update the item's description
      - Close pop-up window.
      - Return to **Item For Sale** form

```
UPDATE Item
SET description = $description
WHERE item_id = $itemId;
```

Buy Item

Abstract Code with SQL

- User can purchase Item by:
  - Bidding:
    - If there are no other bids on the item, then the bid amount must be greater than or equal to Auction's starting bid. The user who listed the item may not bid on the item; do not display ***Bid On This Item*** button. Bid amount must be a float.
      - Display error message if bid amount does not validate
    - Else, user's bid amount must be greater than the current highest bid
    - Bid amount must be less than the auction's Get It Now Price. If the user's bid amount is greater than the Get It Now Price, then display an error message suggesting the user click on the ***Get It Now*** button instead of bidding.
    - If item is bid on, refresh the **Item For Sale** form and run the **View Item** task

```
INSERT INTO Bid (bid_id, auction_id, username, bid_amount, bid_time)
VALUES (nextval('bid_id_seq'), $auctionId, $username, $bidAmount, now());
```

  - Get It Now: User clicks on the ***Get It Now*** button, which ends the auction immediately and sets the user's username as the auction's winner.
    - Direct the user to the **Item Results** form and run the **Item Results** task

```
-- create a bid with the GetItNowPrice
INSERT INTO Bid (bid_id, auction_id, username, bid_amount, bid_time)
VALUES (nextval('bid_id_seq'), $auctionId, $username, $getItNowPrice,
now());

-- update auction end time
UPDATE Auction
SET auction_end_time = now()
WHERE auction_id = $auctionId;
```

Cancel Item

Abstract Code with SQL

- Administrative user clicked on the ***Cancel This Auction*** button:
- Prompt for cancellation reason with pop-up window
    - On clicking the ***Cancel*** button:
        - Auction ends immediately, and the auction end date is set to the date and time of cancellation and set the admin username who cancelled the auction
        - Optional cancellation reason is stored in the database, which must be a string data type

```
UPDATE Auction
SET cancel_reason = $cancelReason, auction_end = now(), cancelled_by =
$username
WHERE item_id = $itemId;
```

        - Close pop-up window
        - Direct the user to the **Item Results** form

View Rating

Abstract Code with SQL

- User clicked on ***View Ratings*** from **Item for Sale** or from **Item Results** ***(***if a bid is completed, expired or canceled***)***
- Query information about the item and its ratings
    - Find the current Item using the item's ID; display the item's information in Item ID and the Item Name fields

```
SELECT item_id, item_name
FROM Item
WHERE item_id = $itemId;
```

    - Find the Rating associated with the item using the item's ID; display the appropriate information in the Average Rating (round up to one decimal place), username of the user who rated the item (Rated by), Date, the star rating, and Comments fields
        - Ratings are sorted in the order with the most recent rating listed first

```
-- get all ratings matching this item name
SELECT r.username,
        r.number_of_stars,
        r.rating_time,
        r.comment
```

```
FROM Rating r
JOIN Item i ON r.item_id = i.item_id
WHERE i.item_name = $itemName
ORDER BY r.rating_time DESC;

-- get the average rating across all items matching this item name
SELECT ROUND(avg(number_of_stars), 2) AS avg_rating
FROM Rating r
JOIN Item i ON r.item_id = i.item_id
WHERE i.item_name = $itemName;
```

- Display ***Delete*** button link the rating if the current user is an administrator, or display ***Delete My Rating*** button for the author of the rating, i.e., when the result of the below query is not empty

```
SELECT username
FROM "User" u
JOIN Rating r ON u.username = r.username
WHERE u.position IS NOT NULL OR (r.username = $username AND r.rating_id =
$ratingId);
```

- Display My Rating and Comments field at the bottom of the **Rate Item** form for the current user to add their rating if the current user is the winner of the item and has not left a rating for the item yet
  - First query to check if the current user is the winner of the item.

```
SELECT username
FROM Bid b
JOIN Auction a ON b.auction_id = a.auction_id
WHERE a.item_id = $itemId
AND b.username  = $username;
```

  - Second, if the result of the above query is not null, query to check if the current user (also the winner of the item) has left a rating. Display My Rating, Comments field, and ***Rate This Item*** button at the bottom of the **Rate Item** form if the result of the below query returns null.

```
SELECT Username
FROM Rating
WHERE item_id = $itemId
AND username = $username;
```

- Upon clicking ***Close*** button, return the user to the **Item Results** form

Add/Delete Rating

Abstract Code with SQL

- User clicked on ***Rate This Item*** from **Item Ratings** form:
    - Validate that a rating was entered, display error message if not; comments are optional
    - Refresh the form and run the **View Rating** task to update the displayed ratings
    - In the View Rating section, ***Rate This Item*** will be displayed based on user authorization. Only the winner of the item who has not left a rating is authorized to rate the item.

```
INSERT INTO Rating (rating_id, username, item_id, number_of_stars, comment,
rating_time)
VALUES(nextval('raiting_id_seq'), $username, $itemId, $numberOfStars, $comment,
$ratingTime);
```

- User clicked on ***Delete My Rating*** from **Item Ratings** form:
    - Remove rating from the item
    - In the View Rating section, ***Delete My Rating*** will be displayed based on user authorization. Only admin users and the author of the rating will have the option to delete the rating.

```
DELETE FROM Rating
WHERE item_id = $itemId
AND username = $username;
```

    - Refresh the form and run the **View Rating** task to update the displayed ratings, and allow the user to add a new rating if that user is the winner of the auction
- Upon clicking on ***Close*** button, direct user to **Item For Sale** form and run the **View Item** task for current auctions, or direct user to **Item Results** form and run the **Item Results** task for ended auctions

Auction Results

Abstract Code with SQL

- User clicked on ***View Auction Results*** button link
- Query information about auctions that have already ended, and item and bids associated with each auction.
    - Find the item associated with the auction using the item's ID, and the bids associated with that auction. Display the appropriate information in the ID, Item Name, Sales Price, Winner, and Auction Ended fields
        - Item Name is also a link to **Item Results** form
    - Find all auctions that have ended, using AuctionEnd where AuctionEnd is prior to the datetime when the user chooses to view the auction results
        - Auctions are sorted in the order with the most recently ended auction listed first
        - For auctions without a winner being declared, update those auctions with the winner (if there is any)

```
WITH winning_bids AS (SELECT DISTINCT ON (b.auction_id)
                                b.auction_id,
                                b.bid_amount,
                                b.username
                        FROM Bid b
                        JOIN Auction a ON b.auction_id = a.auction_id
                        WHERE a.auction_end_time < now()
                        AND (b.bid_amount >= a.min_sale_price
                                OR b.bid_amount = a.get_it_now_price)
                        AND a.cancelled_by IS NULL
                        ORDER BY 1, 2 DESC)
SELECT i.item_id,
        i.item_name,
        wb.bid_amount AS sale_price,
        (CASE
            WHEN a.cancelled_by IS NOT NULL
            THEN 'Cancelled'
            ELSE wb.username
         END) AS winner,
        a.auction_end_time AS auction_ended
FROM Item i
JOIN Auction a ON i.item_id = a.item_id
LEFT JOIN winning_bids wb ON a.auction_id = wb.auction_id
WHERE a.auction_end_time < now();
```

- Upon clicking ***Done*** button, return to **BuzzBid Main Main Menu**

Item Results

Abstract Code with SQL

- Upon user clicking on ***Item name*** link from **Auction Results**
    - Query for information about the item, auction, and bids
    - Display item details in the Item ID, Item Name, Description, Category, Condition, and Returns Accepted fields

```
SELECT i.item_id,
            i.item_name,
            i.description,
            i.is_returnable,
            i.condition,
            c.category
FROM Item i
JOIN Category c ON i.category_id = c.cateogry_id
WHERE i.item_id = $itemId;
```

    - Display auction details in the Get It Now Price and Auction Ended fields

```
SELECT get_it_now_price,
             auction_end_time
FROM Auction
WHERE item_id = $itemId;
```

    - Display bid information:
        - If there was a winner for the auction, then the bid history should highlight the winning bid's row in green
        - If there were bids, but no winner, then the bid history should highlight the last bid's row in yellow
        - For canceled items, the cancellation date and time should be listed, showing "Canceled" as the bid amount, the cancellation date and time in place of the bid time, an d "Administrator" as the user, with the cancellation row highlighted in red (will handle the row colors in application logic)

```
SELECT (CASE
                WHEN a.cancelled_by IS NOT NULL
                   THEN 'Cancelled'
               ELSE b.bid_amount::text
            END) AS bid_amount,
            b.bid_time,
            (CASE
               WHEN a.cancelled_by IS NOT NULL
                  THEN 'Administrator'
                ELSE b.username
            END) AS username
```

```
FROM Bid b
JOIN Auction a ON b.auction_id = a.auction_id
WHERE a.item_id = $itemId;
```

- Upon user clicking ***Close*** button, return to **Auction Results** form

Category Report

Abstract Code with SQL

- Upon admin user clicking on ***Category Report*** button from **BuzzBid Main Menu**:
    - Query for information about the item and aggregate data about items by Category
    - Find all Categories; display Category names in 1st column
    - Find the total number of items in the category; display total item count in 2nd column
    - Find the minimum Get It Now Price across all items in the category; display the minimum Get It Now Price in 3rd column
    - Find the maximum Get It Now Price across all items in the category; display the maximum Get It Now Price in 4th column
    - Calculate the Average Price; display the Average Price in 5th column
    - Sort and display each row in alphabetically by category name
    - If an item does not have a Get it now price, they should be included in the item count, but their prices should not be part of the min, max, or average prices.
    - This report should include both items that have already been sold (the auction has ended) or that are still for sale, but not canceled auctions.

```
SELECT c.category,
            COUNT(i.*) as total_items,
             MIN(a.get_it_now_price) AS min_price,
            MAX(a.get_it_now_price) AS max_price,
            ROUND(avg(a.get_it_now_price), 2) as avg_price
FROM Item i
JOIN Category c ON i.category_id = c.category_id
JOIN Auction a ON i.item_id = a.item_id
WHERE a.cancelled_by IS NULL
GROUP BY c.category
ORDER BY c.category;
```

- Upon clicking ***Done*** button, return to **BuzzBid Main Menu**

User Report

Abstract Code with SQL

- Upon admin user clicking on ***User Report*** button from **BuzzBid Main Menu**:
  - Query for information about all users and aggregate information about users' listed, sold, won, rated items, and most frequent item condition
  - Find all users*;* display username in 1st column
  - Find the total number of items listed by the user (whether sold or not); display total item number in 2nd column
  - Find the total number of items sold by the user (either via Get It Now or auction); display total item number in 3rd column
  - Find the total number of items won by the user (either via Get It Now or auction); display total item number in 4th column
  - Find the total number of items rated by the user; display total item number in 5th column
  - Find the most frequent condition across all items purchased by the user (in the event of a tie, the lowest-quality condition should be used); display the condition in 6th column
    - If the user has no listings the 6th column should display "N/A"
  - Sort users in descending order by their respective total number of listings
  - Upon clicking ***Done*** button, return to **BuzzBid Main Menu**

```
WITH listed_items AS (SELECT username,
                             COUNT(*) AS listed_count
                      FROM Item
                      GROUP BY 1),
     sold_items AS (SELECT i.username,
                           COUNT(*) AS sold_count
                    FROM Item i
                    JOIN Auction a ON i.item_id = a.item_id
                    WHERE a.cancelled_by IS NULL
                    AND a.auction_end_time < now()
                    AND EXISTS (SELECT 1
                                FROM Bid b
                                WHERE b.auction_id = a.auction_id
                                AND (b.bid_amount > a.min_sale_price OR
                                     b.bid_amount = a.get_it_now_price))
                    GROUP BY 1),
     won_items AS (SELECT b.username,
                          COUNT(a.*) AS won_count
                   FROM Bid b
                   JOIN Auction a ON b.auction_id = a.auction_id
                   WHERE a.cancelled_by IS NULL
                   AND a.auction_end_time < now()
                   AND b.username IN (SELECT DISTINCT ON (b1.auction_id)
                                             b1.username
                                      FROM Bid b1
                                      WHERE b1.auction_id = b.auction_id
                                      ORDER BY b1.auction_id,
```

```
                                                b1.bid_amount DESC)
                        GROUP BY 1),
     rated_items AS (SELECT username,
                            COUNT(*) AS rated_count
                        FROM Rating r
                        GROUP BY 1),
     category_freq AS (SELECT DISTINCT ON (username)
                              username,
                              freq_condition
                       FROM (SELECT username,
                                    condition AS freq_condition,
                                    COUNT(*) as condition_count
                             FROM Item
                             GROUP BY 1, 2
                             ORDER BY 3 DESC, 2 DESC) c
                       ORDER BY 1, condition_count DESC, 2 DESC)
SELECT u.username,
       COALESCE(l.listed_count, 0) AS listed_count,
       COALESCE(s.sold_count, 0) AS sold_count,
       COALESCE(w.won_count, 0) AS won_count,
       COALESCE(r.rated_count, 0) AS rated_count,
       COALESCE(c.freq_condition::text, 'N/A') AS freq_condition
FROM "User" u
LEFT JOIN listed_items l ON u.username = l.username
LEFT JOIN sold_items s ON u.username = s.username
LEFT JOIN won_items w ON u.username = w.username
LEFT JOIN rated_items r ON u.username = r.username
LEFT JOIN category_freq c ON u.username = c.username
ORDER BY 2 DESC;
```

Top Rated Item Report

Abstract Code with SQL

- Upon clicking on ***Top Rated Report*** button from **BuzzBid Main Menu**:
    - Query for data about the items and their ratings
    - Calculate the average of all items with ratings (item listings without ratings should not be included in the average rating calculation)
    - Display the top 10 rated items sorted by average rating (rounded up to one decimal place) descending, by Item Name in the 1st column, Average Rating in the 2nd column, and Rating Count in the 3rd column
        - If there is the event of a tie, sort by name ascending
    - In the event multiple items have the same average ratings, the results should still be limited to 10 items

```
WITH rated_items AS (SELECT i.item_name,
                            ROUND(avg(r.number_of_stars), 1) AS avg_rating,
                            COUNT(r.*) AS rating_count
                     FROM Item i
                     JOIN Rating r ON i.item_id = r.item_id
                     GROUP BY 1)
SELECT i.item_name,
       ri.avg_rating,
       ri.rating_count
FROM Item i
JOIN rated_items ri ON ri.item_name = i.item_name
ORDER BY 2 DESC, 1
LIMIT 10;
```

- Upon clicking ***Done*** button, return to **BuzzBid Main Menu**

Auction Statistics Report

Abstract Code with SQL

- Upon clicking on ***Auction Statics Report*** button from **BuzzBid Main Menu:**
    - Query for information about all items, their ratings, and their associated auctions
    - Aggregate the number of active auctions (auctions which have not closed); display total count of active auctions in the 1st row

```
SELECT COUNT(*) AS active_auctions
FROM Auction
WHERE auction_end_time > now();
```

    - Aggregate the number of finished auctions (auctions that have ended with or without a winner but were not canceled); display total count of finished auctions in the 2nd row

```
SELECT COUNT(*)
FROM Auction
WHERE cancelled_by IS NULL
AND auction_end_time < now();
```

    - Aggregate the number of won auctions (auctions which have successful winners); display total count of auctions won in the 3rd row

```
SELECT COUNT(a.*)
FROM Auction a
WHERE a.cancelled_by IS NULL
AND a.auction_end_time < now()
AND EXISTS (SELECT 1
                    FROM Bid b
                    WHERE b.auction_id = a.auction_id
                    AND (b.bid_amount > a.min_sale_price
                                OR b.bid_amount = a.get_it_now_price));
```

    - Aggregate the number of canceled auctions (auctions which have been canceled); display total count of canceled auctions in the 4th row

```
SELECT COUNT(*)
FROM Auction
WHERE cancelled_by IS NOT NULL;
```

    - Aggregate the number of rated items; display total count of rated items in the 5th row

```
SELECT COUNT(*)
FROM Item i
```

```
WHERE EXISTS (SELECT 1
                    FROM Rating r
                    WHERE i.item_id = r.item_id);
```

- Aggregate the number of unrated items; display total count of unrated items in the 6th row

```
SELECT COUNT(*)
FROM Item i
WHERE NOT EXISTS (SELECT 1
                        FROM Rating r
                        WHERE i.item_id = r.item_id);
```

- Upon clicking ***Done*** button, return to **BuzzBid Main Menu**

Cancelled Auction Details

Abstract Code with SQL

- Upon clicking on ***Canceled Auction Details*** button from **BuzzBid Main Menu:**
  - Query for information about canceled items and their associated auction and users
    - Results are sorted by the item ID in descending order
  - Display canceled item's ID in 1st Column
  - Display username of the user who listed the item in 2nd Column
  - Display date the auction was canceled on in 3rd Column
  - Display cancellation reason given in the auction, if any, in 4th Column

```
SELECT i.item_id,
        i.username,
        a.auction_end_time,
        a.cancel_reason
FROM Item i
JOIN Auction a ON a.item_id = i.item_id
WHERE a.cancelled_by IS NOT NULL
ORDER BY 1 DESC;
```

- Upon clicking ***Done*** button, return to **BuzzBid Main Menu**