Project Group #1: Team Oaklaura

Grant Wu and Jessica Ramirez

Project Step 2 - Draft

1. Feedback by TAs and Peer Reviewers

TA Review - Madelyn Lazar

"This is great work team 1! Your naming conventions were consistent throughout, you meet all the requirements for an M:M relationship + enough entity tables, and you clearly explained how each of your relationships will work in your outline. I also love how well formatted and easy to read your outline is!

Some notes:

-You lost points for not including numerical data in your overview. I left some specific questions that might help you when deciding what to include/answer in your overview within the rubric.

-For future steps, you will be required to have at least one nullable FK. So I recommend you start thinking about which table/FK this requirement could apply to in your data structure."

Peer Review 1 - Tyler Grzymalski

"Your overview does a great job of explaining how this database would help streamline repair efforts by helping to ensure no aspect of repair gets overlooked and increasing accountability between repairs and volunteers that work on them.

There are several facts in the overview that directly connect to the database solution. Limited funding explain why the sales invoices cannot contain more than one bike, there is a detailed explanation on how the repair process goes and it aligns with the structure of the database, handwritten notecard's were getting unruly which gives the reasoning for why a database needs to be implemented.

There are five entities described and they all align with course material. The bikes table does a good job of covering important consumer information regarding the brand and type of bike, and it elaborates on the current condition of the bike. The store personnel table and customers table accurately track standard personal information. Perhaps there is a way to combine them by adding a customer role to the StorePersonnel Table? The repair reports table keeps track of the repair process on bikes. The sales report table does a good job of outlining a traditional invoice.

The relationships are correctly formulated and make sense. There is a M:M relationship between StorePersonnel and Bikes since multiple volunteers typically work on one bike. There is consistency in naming: entities are plural and attributes are singular. Overall, great start to the project!

This review is all original work."

Peer Review 2 - Andrew Mathena

"Hi Team,

Great work on the draft here. I appreciated the unique and well-thought out use case.

The overview does a good job of describing the problem to be solved by the website and database back end. It sounds like the financial constraints and volunteer-powered staff necessitate the use of a system to track updates on work across shifts. The system replaces a manual and likely error-prone note card system. So it sounds like a great use case for a database powered application which can help the organization drive revenue.

The specific facts around mechanical aptitude of the staff, the workforce, and the revenue stream of the organization help illustrate the scope and scale of the solution. Some additional numerical facts which could add further clarity include the total workforce count, the total number of annual orders or revenue, and maybe even a comparison between revenue and the cost of a POS system to help illustrate why that's not a viable option.

Three tables in your design appear to describe objects: StorePersonnel, Customers, and Bikes. These tables include fields for information I would believe to be cricial to support the described use case. I was especially interested by your use of the enum field types for a number of fields, such as bike color; this has made me reconsider where I can use enum types to refine & simplify my own group's database design. RepairReports and Salesreports appear to be intersection/reporting tables which support the M:N requirements of the project design and contain information critical to track repair progress and sales.

The outline of entity details describes the purpose and attributes of each: Bikes includes color and styling info, as well as repair completion status. StorePersonnel includes contact info for staff. Likewise, Customers includes contact info for customers. The report tables include attributes to connect out to the staff and customer tables, and critical date information.

1:M relationships are correctly formulated. As mentioned above, RepairReports and Salesreports appear to be intersection/reporting tables which support the M:N requirements of the project design. All information in the ERD represents the data accurately, assuming that a customer must have at least one sale in order to be represented in the system (as another reviewer pointed out).

Naming consistency between entities & attributes is good. Pluralization is correct for all items, and camel case is used as a standard convention in the system.

Also kudos on the beautiful LaTex formatting!

All content here was my original work.

Best,

Andrew"

Peer Review 3 - Drew Schlabach

"Grant and Jessica,

Before I get to the technical details of my review, I want to first commend you for the excellent formatting of your draft. It is easy to read, visually appealing, and extremely organized. Great job! Moving on…

The information provided in your overview does a fantastic job of justifying your database design. Between the basic POS, the involvement of many volunteers per bike, and the co-op's intriguing business model, a database appears to offer improvement in plenty of areas. A well-made database will allow the co-op to more efficiently track sales, and perhaps by storing repair logs will allow bikes to be completed more quickly as well.

At least four entities are described in your outline (Bikes, StorePersonnel, Customers, RepairReports, SalesReports), and each one does in fact represent a single idea to be stored as a list. If I am not mistaken, the Bikes, StorePersonnel, and Customers entities will function as objects, and the RepairReports and SalesReports entities will function as transactions. Additionally, your inclusion of a fifth entity despite the project's minimum of four is in line with the same trend I have seen in other groups and shows a desire to exceed expectations.

The outline of each entity includes sufficient details about its purpose, and each attribute is given an appropriate data type and constraints (not NULL for everything, auto_increment for PKs, and various categorical constraints for such attributes as color, style, and status). One detail that I really appreciate is the constraint of UNIQUE for the email attribute in the Customers entity. That's not something that I would have thought of myself, but it makes perfect sense considering that many organizations limit only one account per email address. Additionally, you have provided for each entity a thorough summary of its relationship to other entities.

The 1:M relationships appear to be correctly formatted, and I notice that while each StorePersonnel can be tied to zero or more SalesReports, each Customer must be tied to at least one SalesReport. I may be wrong, but I believe that this would result in an insertion anomaly if the co-op wishes to store information about a potential customer who has not yet made a purchase. Your M:M relationship between StorePersonnel and Bikes is cleverly implemented with foreign keys in both RepairReports and SalesReports, and your accompanying ERD does a great job of visualizing your outline.

There is consistency in all required areas: the entity and attribute names match between the outline and ERD, the entities are plural and the attributes are singular, and the entities use UpperCamelCase while the attributes use lowerCamelCase.

I enjoyed reading your Step 1 draft, and I wish you the best of luck in developing your project.

Best regards,

Drew Schlabach

This review is entirely original, and AI was not used in any capacity to write it."

2. Actions Based on the Feedback

Based on external feedback we received on Project Step 1 Draft, the following actions were taken:

- **Suggestion from ML/AM:** Numerical facts have been added to the overview to provide context and establish the scope of the database.
- **Suggestion from ML:** bikeID in RepairReports was changed to a nullable FK, to facilitate documentation of non-bike specific work, such as cleaning and fixing stock parts such as cassettes, wheels and derailleurs.
- **Suggestion from TG:** It was suggested that we combine the StorePersonnel and Customers entities. While combining customers and store personnel into the same entity is an interesting idea, we think differentiating between them is advantageous in that: a) it distinguishes between service providers and service recipients, b) it allows flexibility to add special attributes to these entities later in the future (skills for StorePersonnel, payment info for Customers), and c) Customer info may require additional privacy/security considerations compared to StorePersonnel. For these reasons, we decided to keep the separation between Customers and StorePersonnel as initially proposed.
- **Suggestion from DS:** An insertion anomaly was identified in the previous formulation of the Customer-SalesReports relationship, where each Customer corresponded to one-to-many SalesReports. This has been altered here such that now each Customer corresponds to zero-to-many SalesReports. This change now allows for a Customer to be created without necessitating a linked SalesReport, to handle situations where we may want to acquire information from a potential Customer.

3. Upgrades to the Draft version

Based on internal conversations, the following actions were made:

- **Removed personnelID as FK within SalesReports:** To avoid documenting an additional M:M relationship between StorePersonnel and Customers we removed the tracking of the StorePersonnel from the SalesReports entity. We do not consider this to be important information to track and removal of it greatly simplifies the database design.
- **Modified ERD:** The ERD was modified such that the relationship between StorePersonnel and SalesReports was removed, as discussed above.
- **Reclassified relationship between Customers and SalesReports:** We have reclassified the relationship between Customers and SaleReports to be non-identifying since they do not share a primary key. We believe we misunderstood this concept before and erroneously classified the relationship as an identifying one.

Application of Normalization principles to our database design led to the following changes:

- **Addition of new Contacts Entity:** We identified a situation that introduces data redundancy: if a StorePersonnel (employee or volunteer) decided to purchase a bike from the co-op, their contact information would have to be re-entered as a new record in the Customers table (and vice versa if a Customer became an Employee or Volunteer). This means that the contact information of that person may exist in two places, and any updates to that contact information would create an update anomaly—requiring their information to be updated in more than one location. To alleviate this issue we introduced an additional entity, Contacts, which stores the contact information of a single person. Each Customer and SalesPersonnel is linked to exactly zero or one Contacts entity.
- **Addition of receiveNewsletter attribute in Customers Entity:** In addition to the rationale provided for keeping Customers and StorePersonnel separate, above in Section 2, we added an additional Customer specific attribute—receiveNewsletter—to further delineate between the two entities. The attribute is a tinyint, representing a boolean value denoting whether a customer is subscribed to the newsletter (1) or not (0, default).
- **Removal of isCompleted attribute from Bikes Entity:** This attribute was originally used to store a boolean status of whether a bike was repaired and ready to be sold yet. We realized this was redundant information that depended on the status attribute, also within the Bikes entity. If the status attribute is 'For Sale', then isCompleted would always be 1 (true). This violates the principles of normalization, and so we removed it as an attribute.

4. The Oaklaura Bike Cooperative

The Oaklaura Bike Cooperative is a non-profit organization that accepts donations of old or broken bicycles, refurbishes them, and then sells them at an affordable price. The co-op sees 50-100 bicycles enter and exit their doors on a monthly basis. Due to limited funding, the co-op operates with a small team of 10 employees and relies heavily on volunteers to assist with bicycle repairs. At the current time there are about 100 volunteers that donate their time, and this is expected to grow over the course of the next five years. The co-op's limited funding also requires they use a basic POS system that can only process one bike sale at a time, setting a limit of one bike per sales order. The co-op brings in on average between $10,000 and $25,000 per month, and most of this revenue is used to support the small team of dedicated employees as well as to provide community engagement events.

Store personnel consist of both volunteers and employees. Most volunteers are not experienced bike mechanics, so they often can't fully repair a bike during the few hours the co-op is open for volunteer work. To maintain continuity and organization, volunteers are expected to repair what they can during their shift and document their progress in a report. This allows the next volunteer and/or employee to review the logs and continue the work where the previous one left off. Sometimes volunteers engage in non-bike specific work, such as cleaning and fixing stock parts such as cassettes, wheels and derailleurs, which gets documented in these repair reports as well. Once a volunteer believes a bike is fully repaired, a trained employee inspects it to ensure it meets safety standards before placing it on the sales floor. Once there, a bike can be sold, a transaction which is tracked through a sales report.

Historically, the co-op tracked repair progress and sales using handwritten notecards stored in a filing cabinet. However, as the organization grows, this system has become increasingly difficult to manage. Implementing a database would be an ideal solution for organizing and sharing information between volunteers and employees about the status of each bicycle, as well as for tracking the inventory and sales history of the ever expanding co-op.

5. Database Schema

Bikes Table

Contains details on a particular bike that resides within the co-op.

- **bikeID [PK]:** int, not NULL, auto_increment
- **color:** enum('Black', 'White', 'Red', 'Blue', 'Green', 'Pink', 'Purple', 'Yellow', 'Orange', 'Silver', 'Other'), not NULL
- **style:** enum('Mountain', 'Road', 'Fat', 'Hybrid', 'Enduro', 'BMX', 'Cruiser', 'Kids', 'Electric'), not NULL
- **brand:** varchar(45), not NULL
- **status:** enum('In Repair', 'In Review', 'For Sale', 'Sold'), not NULL
- **dateReceived:** date, not NULL

Relationships:

- M:M relationship between Bikes and StorePersonnel is implemented with bikeID and personnelID as FK's within both RepairReports and within SalesReports.
- 1:1 relationship between Bikes and SalesReports is implemented by bikeID as a FK within SalesReports. *Note: due to our outdated POS system, only one bike can be sold at a time (i.e. only one Bike can appear on each SalesReport).*
- 1:M relationship between Bikes and RepairReports is implemented with bikeID as a FK within RepairReports.

StorePersonnel Table

Holds information on store employees and volunteers that work within the co-op.

- **personnelID [PK]:** int, not NULL, auto_increment
- **contactID [FK - Contacts]:** int, not NULL
- **role:** enum('Employee', 'Volunteer'), not NULL

Relationships:

- M:M relationship between StorePersonnel and Bikes is implemented with bikeID and personnelID as FK's within both RepairReports and within SalesReports.
- 1:M relationship between StorePersonnel and RepairReports is implemented with personnelID as a FK within RepairReports.
- 1:1 relationship between StorePersonnel and Contacts is implemented with contactID as a FK within StorePersonnel.

Customers Table

Holds information relating to existing and potential customers.

- **customerID [PK]:** int, not NULL, auto_increment
- **contactID [FK - Contacts]:** int, not NULL
- **receiveNewsletter:** tinyint, not NULL, DEFAULT = 0 (false)

Relationships:

- 1:M relationship between Customers and SalesReports is implemented with customerID as a FK inside of SalesReports.
- 1:1 relationship between Customers and Contacts is implemented with contactID as a FK within Customers.

RepairReports Table

Holds repair information performed on a particular bikes (Bikes_StorePersonnel Intersection Table that includes additional repair information).

- **repairID [PK]:** int, not NULL, auto_increment
- **personnelID [FK - StorePersonnel]:** int, not NULL
- **bikeID [FK - Bikes]:** int
- **dateRepaired:** datetime, not NULL
- **hoursSpent:** decimal(4,2), not NULL
- **description:** varchar(255), not NULL

Relationships:

- 1:M relationship between RepairReports and StorePersonnel is implemented with personnelID as a FK inside RepairReports.
- 1:M relationship between RepairReports and Bikes is implemented with bikeID as a FK inside RepairReports.

SalesReports Table

Holds information pertaining to the sale of a particular bike.

- **salesID [PK]:** int, not NULL, auto_increment
- **bikeID [FK - Bikes]:** int, not NULL, unique
- **customerID [FK - Customers]:** int, not NULL
- **dateSold:** date, not NULL
- **price:** decimal(5,2), not NULL

Relationships:

- 1:1 relationship between Bikes and SalesReports is implemented by bikeID as a FK within SalesReports. *Note: due to our outdated POS system, only one bike can be sold at a time (i.e. only one Bike can appear on each SalesReport).*
- 1:M relationship between SalesReports and Customers is implemented with customerID as a FK within SalesReports.

Contacts Table

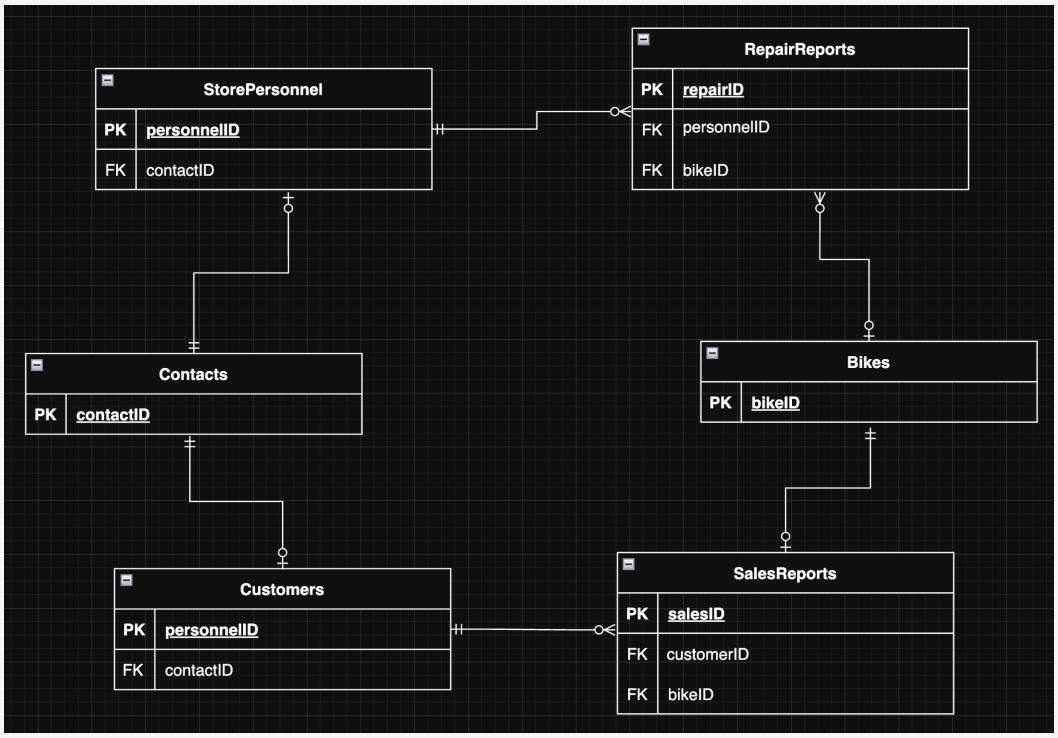
Holds contact information for any person in the system (both service providers and service recipients).

- **contactID [PK]:** int, not NULL, auto_increment
- **firstName:** varchar(45) not NULL
- **lastName:** varchar(45) not NULL
- **phone:** varchar(20), not NULL
- **email:** varchar(100), not NULL, UNIQUE

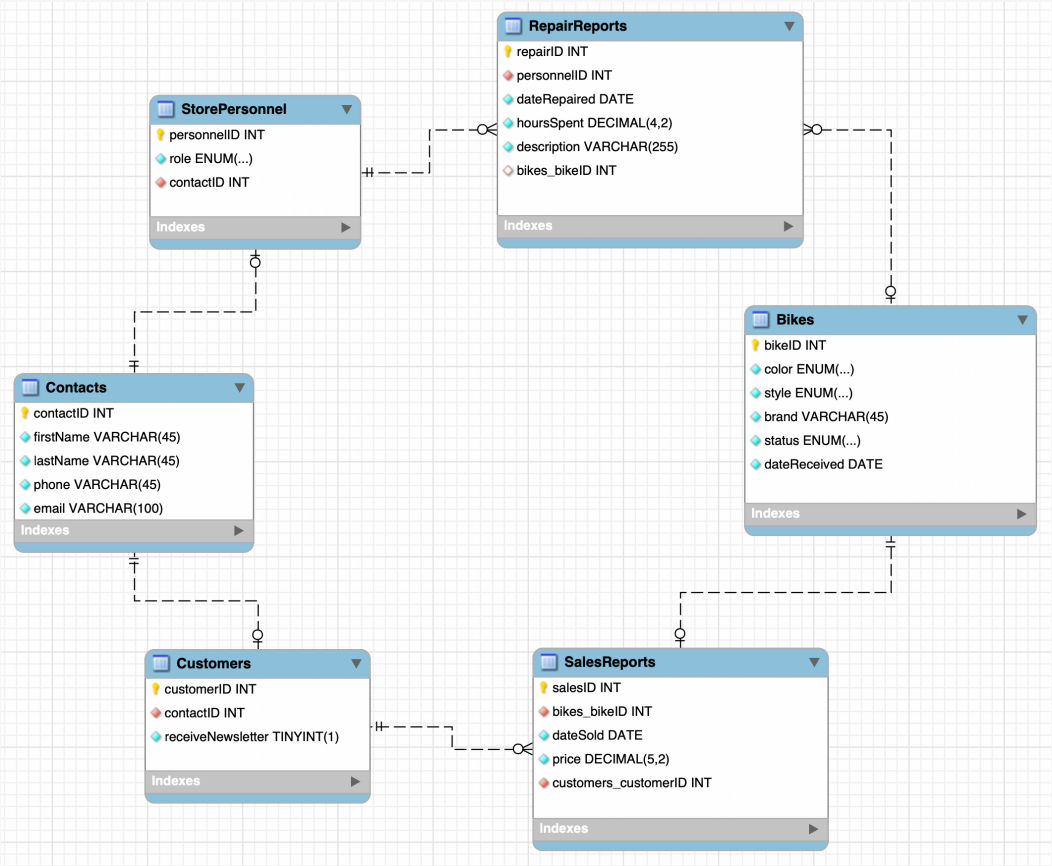
Relationships:

- 1:1 relationship between Contacts and StorePersonnel is implemented with contactID as a FK within Customers.
- 1:1 relationship between Contacts and Customers is implemented with contactID as a FK within Customers.

6. Entity-Relationship Diagram



7. Database Schema



8. Sample Data

TBD

9. Citations

- Inspiration for the Bike Co-Op came from The Recyclery, a non-profit bike shop based out of Chicago, IL (last retrieved on 4/9/2025): https://www.therecyclery.org/
- MySQL workbench was used to create the ERD diagram shown above.
- The LaTeX template used here was adapted from the Cleese-Assignment template v.2.0 (retrieved on 4/2/2025): https://latextemplates.com/template/cleese-assignment
- TeXShop was used for all LaTeX related compilations.
- All database and design related work is original.