Development Team Project: Project Report

Executive Summary

This database design proposal focuses on addressing the data management needs of ElectroSpares, a B2B wholesale supplier of office equipment to corporate clients such as government departments and educational institutions. A tailored database solution is proposed that meets ElectroSpares' requirements for a robust database solution, which can enhance their utilisation and management of sales, product, and customer data.

1. Logical design

1.1 Entities and Attributes

The database relies on seven core entities (figure 1) to ensure data integrity and organisational traceability. These are CUSTOMERS, EMPLOYEES_SYSTEM, ORDERS and ORDER_ITEMS, PRODUCTS, SUPPLIERS and PRODUCT_REVIEWS.

1.2 ER Diagram

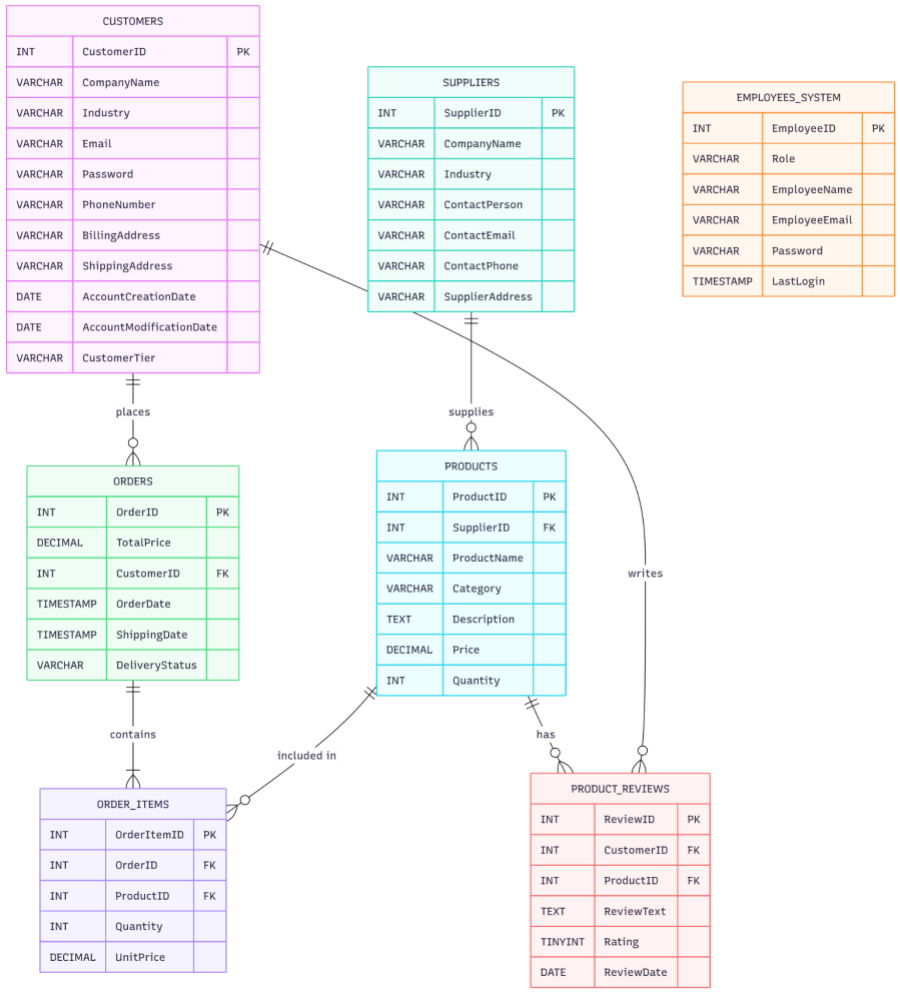

Figure 1 - logical design of the proposed database system

1.3 Relationships Between Entities

Entities	Cardinality	Justification
Customers - Orders	One-to-many	A customer can place multiple orders. An order can be placed by only one customer.
Order - OrderItems	One-to-many	Each order contains multiple line items. An order item belongs to only one order.
Product - OrderItems	One-to-many	A product can appear in many order line items. One order line item can reference exactly one product.
Suppliers - Products	One-to-many	Each supplier provides multiple products. Every product can be provided by exactly one supplier.
Products - Reviews	One-to-many	Each product can have multiple reviews. Each review has to be linked to one product.
Customers - Reviews	One-to-many	Each customer can leave multiple reviews. Each review can be linked to only one customer.

Figure 2 - relationships between entities inside the proposed database

1.4 Data Types Used and Formats

It is necessary to consider the right data type and format of data, which ties back to the choice of a concrete relational database management system (RDBMS). It is also a good practice to consider normal forms. All the proposed tables are fourth normal form compliant (Fagin, 1979).

Multiple categories of data types and formats are to be considered, as shown in figure 3:

Chosen Type	Data Group	Format and Justification	Example entities
INT	Identifiers & Quantity	Provides maximum signed value (over 2 billion), ensuring sufficient scale for all primary/foreign keys and inventory tracking (MySQL, no date).	ProductID
DECIMAL(8,2)	Monetary Values	Mandatory for financial data to prevent rounding errors inherent in floating-point types. Provides fixed-point precision for financial accuracy.	UnitPrice
TINYINT	Discrete Values	The most efficient storage choice for the Rating field (1-10). Format is restricted to an integer between 1 and 10.	Rating
VARCHAR	Text/Short Strings	Highly efficient variable-length storage. Used for PhoneNumber to accommodate formatting characters (spaces, '+') and for Email to support the standard local-part@domain format (MySQL, no date).	Email
VARCHAR	Security String	The Password field stores the encrypted, fixed-length hash string (e.g., SHA-256), not the plaintext, for security compliance.	Password
TEXT (LONGTEXT)	Large Text	Used to safely accommodate lengthy content beyond VARCHAR limits (Atlassian, no date).	ReviewText
TIMESTAMP	Time and Date	Critical for time-zone awareness and highly storage efficient (4 bytes). It stores time in UTC, ensuring transactional consistency across different operational sites (MySQL, no date).	LastLogin

Figure 3 - data types and formats in the proposed database system

2. Database Model Proposal

Implementation of a RDMS will comply with Electrospares' requirements to have a robust database which can manage various aspects of their data, including order details, inventory levels, and customer information. MySQL is the ideal RDBMS due to its low cost of ownership, low latency, and wide-scale utilisation across different industries. It can also contain a large volume of data, crucial due to the volume of orders created each year (Zhang and Pan, 2022).

Disaster recovery for the database will be set up, including automated, encrypted backups, real-time replication across servers, and clearly defined recovery time objectives (RTO) (Morrison, 2024).

In addition to using SQL commands to select and modify data, MySQL possesses automatic manipulation capabilities. This allows for data to be added once a new customer account is created or when a new order is placed. Staff members can use primary keys, such as order numbers, to retrieve data as required.

3. Development and Implementation Method

3.1 Detailed Requirements Gathering

Consultation will take place with all users to ensure that all relevant data is identified and noted (Watt, 2014).

3.2 Schema Creation

A model is devised to create a database structure which can be implemented from the logical design in figure 1. All relationships between data entities and attributes are defined (Sherman, 2015).

3.3 Data Importation

Existing data is imported into the database before verifying the accuracy of the data to identify any inconsistencies and ensure that the integrity of the data is maintained.

3.4 User Access Setup

Correct permissions for users are set up in compliance with GDPR and Electrospares' cybersecurity protocols ensuring that only authorised roles can view, modify, or delete data as appropriate. Figure 4 displays an example user hierarchy.

Figure 4 – user hierarchy and access control

3.5 User Acceptance Testing

The functionality of the database is tested using real-world scenarios to identify and correct any issues before deployment.

3.6 Deployment & Hypercare

Once the database is deployed, time will be spent ensuring that staff are adequately trained to use and manage the database within their role to minimise disruptions.

4. Data Management Pipeline

4.1 Data Flow Through The System

Data flows into the system from several channels: customers create accounts, place orders, and leave reviews; administrators insert the details of suppliers and products; employees manage login information and sometimes interact with order data; and the system automatically creates order-item links. All data arriving from these sources is streamed via a Kafka topic to provide fault-tolerant, low-latency data ingestion before entering the cleaning pipeline. After processing, it will reside in the database that provides consistency and durability guarantees compliant with ACID. Once cleaned, the data can be queried for reports and operational insight, while sensitive fields are protected by encrypted authentication and role-based permissions.

4.2 Customer Data

Customer data originates from onboarding, procurement forms, and CRM updates. Issues to be addressed include inconsistent naming conventions, missing or outdated addresses, typographical errors in contact details, and unclear classifications of CustomerTier.

4.3 Supplier Data

Onboarding forms, contracts, spreadsheets, and uploads are used to capture the information of a supplier. Common problems involve duplicate suppliers, outdated contact points, inconsistent industry labels, and incomplete addresses.

4.4 Product Data

Product records originate from supplier SKU files, manual updates, or API feeds. Common data errors include missing descriptions, wrong category assignments, pricing errors, stock mismatches, and SupplierID issues that break product–supplier links.

4.5 Order and Order-Item Data

Order data may be missing shipping dates, have inconsistent delivery statuses, duplicate line items, or incorrect totals due to wrong unit prices or quantities.

4.6 Employee System Data

Employee records sometimes lack LastLogin timestamps, include outdated role allocations, or contain weak password formats that pose security risks.

4.7 Product Reviews

Review datasets can include missing text, invalid rating values, or irrelevant and spam-like submissions.

5. Data Cleaning Techniques

5.1 Handling Missing and Duplicate Data

Missing data, such as shipment addresses or customer phone numbers are flagged and imputed using billing data. Orders without shipping dates are marked ‘not yet shipped’.

Missing review text is replaced by a neutral placeholder and incomplete product descriptions are flagged.

Duplicate suppliers are merged, and repeated entries for products are removed by matching SupplierID and ProductName.

5.2 Standardising and Formatting Data

CustomerTier values are normalised, delivery statuses are mapped to four terms, and product categories are aligned to a unified taxonomy. Phone numbers take an international format. Price values are cleaned and converted into decimals.

5.3 Detecting Outliers and Invalid Entries

Negative prices or quantities are invalidated, wrong totals are recalculated, out-of-range ratings are removed, and unrealistic stock counts are flagged.

5.4 Derived Indicators and Metrics

Indicators like IsHighValueCustomer, LateShipmentFlag, RestockNeeded are generated to support analytics, along with order processing times.

5.5 Cleaning Strings and Text Fields

Names are standardised and contact information is validated through patterns. Reviews are checked for inappropriate content, and addresses are normalised to postal standards.

Conclusion

A MySQL database solution for ElectroSpares has been proposed in this report, highlighting the requirement for a system which can handle diverse data sources with unique risks and cleaning needs. The data management pipeline ensures that the data is captured, cleaned, standardised, and validated to preserve data integrity and produce clean, consistent, and analysis-ready datasets. Once implemented, the database will enable ElectroSpares to reliably manage stock levels, track orders, evaluate supplier performance, and generate accurate sales analytics.

References

Atlassian (no date) *Understanding storage sizes for MySQL TEXT data types*. Available at: https://www.atlassian.com/data/databases/understanding-strorage-sizes-for-mysql-text-data-types (Accessed: 20 November 2025).

Fagin, R. (1979) 'Normal forms and relational database operators', *IBM Research Laboratory*. Available at: https://dl.acm.org/doi/pdf/10.1145/582095.582120 (Accessed: 20 November 2025).

Morrison II, B. (2024) *Considerations for building a database disaster recovery plan — PlanetScale*, *PlanetScale*. Available at: https://planetscale.com/blog/considerations-for-building-a-database-disaster-recovery-plan (Accessed: 20 November 2025).

MySQL (no date) *Integer Types*. Available at: https://dev.mysql.com/doc/refman/8.4/en/integer-types.html (Accessed: 20 November 2025).

MySQL (no date) *Fixed Point Types*. Available at: https://dev.mysql.com/doc/refman/8.4/en/fixed-point-types.html (Accessed: 20 November 2025).

MySQL (no date) *Char*. Available at: https://dev.mysql.com/doc/refman/8.4/en/char.html (Accessed: 20 November 2025).

MySQL (no date) *Datetime*. Available at: https://dev.mysql.com/doc/refman/8.4/en/datetime.html (Accessed: 20 November 2025).

Sherman, R. (2015) ‘Chapter 8: Foundational Data Modeling’, in *Business Intelligence Guidebook: From Data Integration to Analytics*. Amsterdam: Elsevier, pp. 173–195.

Watt, A. (2014) ‘Chapter 13: Database Development Process’. Available at: https://opentextbc.ca/dbdesign01/chapter/chapter-13-database-development-process/ (Accessed: 20 November 2025).

Zhang, Y. and Pan, F. (2022) ‘Design and implementation of a new intelligent warehouse management system based on MySQL database technology’, *Informatica*, 46(3). Available at: https://doi.org/10.31449/inf.v46i3.3968 (Accessed: 20 November 2025).