**User & Role Management Questions**

1. What specific attributes need to be stored for each user? (e.g., name, email, employee ID, department)
   1. First name
   2. Last name
   3. Rank
   4. Employee id
   5. Organisation
   6. department
2. What PII fields need special encryption or protection?
3. What authentication method will be used? (Local accounts, SSO, hybrid)
4. How granular should the permission system be? (Page-level, feature-level, data-level)
5. Will users belong to multiple roles simultaneously?
6. Do roles need hierarchical relationships? (e.g., managers inheriting subordinate permissions)
7. Will role assignments be time-bound or permanent?
8. What audit information needs to be tracked for user activities?

**Task Management Questions**

1. What attributes define a task? (e.g., title, description, priority, deadline)
2. How many levels of task hierarchy do you need? (tasks, subtasks, etc.)
3. What states can a task progress through? (workflow states)
4. Can multiple users be assigned to a single task?
5. Do tasks need approval workflows?
6. What dependencies can exist between tasks? (finish-to-start, start-to-start, etc.)
7. Should the system track estimated vs. actual time/effort?
8. What reporting metrics are needed for tasks? (completion rate, overdue rate, etc.)

**Qualifications & Skills Questions**

1. What information defines a qualification? (name, category, expiration rules)
2. Do qualifications have levels or are they binary (has/doesn't have)?
3. Are qualifications tied to official certifications with expiration dates?
4. Will the system track skill proficiency levels? If so, how are they measured?
5. Do certain roles require specific qualifications?
6. Will tasks require certain qualifications to be assigned?
7. Should the system track qualification gaps for development planning?
8. Will training activities be tracked in relation to qualifications?

**Equipment & Resource Questions**

1. What attributes need to be tracked for equipment? (type, serial number, status)
2. Will you track consumable vs. non-consumable resources differently?
3. Do resources have availability schedules?
4. Will you track maintenance schedules or calibration requirements?
5. Does equipment need to be assigned to specific locations?
6. How will resource allocation to tasks be handled?
7. Will you track resource utilization metrics?
8. Do certain resources require specific qualifications to use?

**SAP Integration Questions**

1. Which SAP modules will this system integrate with?
2. What master data will be synchronized from SAP? (users, cost centers, etc.)
3. What transactional data needs to flow to/from SAP?
4. How frequently will data synchronization occur?
5. Will any SAP authorizations need to be replicated in this system?
6. What conflict resolution rules should be applied during synchronization?

**General Schema Questions**

1. What historical data needs to be preserved? (data retention requirements)
2. Are there any regulatory requirements affecting data storage? (GDPR, etc.)
3. What volume of data do you anticipate for each entity type?
4. What reporting and analytics capabilities are needed?
5. Will the schema need to support multiple languages or localization?
6. Are there any anticipated future expansions to consider in the schema design?
7. What data needs to be accessible offline vs. synced only when online?

**User & Role Management**

1. **What specific user attributes need to be stored in the SQL database?**  
   *Description: Identify core user data like name, email, employee ID, department, contact information, etc. This defines your users table structure.*
   1. First Name
   2. Last Name
   3. Employee Id
   4. Department and Department ID
   5. Organisation
   6. Email address
   7. Rank
   8. Primary Role (work)
   9. Permission
2. **What PII fields require encryption at the database level?**  
   *Description: Determine which fields contain personally identifiable information requiring column-level encryption for compliance and security purposes.*
   1. Nil
3. **How will authentication be handled between your web client and server?**  
   *Description: Decide on authentication mechanisms (JWT, session-based, etc.) and what authentication data needs to be persisted in the database.*
   1. What would be the best solution for an intranet environment? JTW or Session?
   2. Authentication on the database? Based on user permissions.
4. **What role and permission structure will you implement?**  
   *Description: Define whether you'll use simple role-based access control, more complex attribute-based control, or a hybrid approach. This affects your roles and permissions tables.*
   1. **Can you expand on this?** I think RBAC should be fine.
5. **Will you implement row-level security in PostgreSQL for data access control?**  
   *Description: Determine if certain data should only be visible to specific users based on department, role, or other attributes using PostgreSQL's row-level security feature.*
   1. **Visibility of database content should be restricted**. Non-privileged users should not have access to the raw database, therefore only admins will have access. At this stage I do not think row-security is required.
6. **What audit logging requirements do you have for user actions?**  
   *Description: Specify what user activities must be logged for compliance or security purposes, which will define your audit tables.*
   1. **Access to the raw database**
   2. **Changes to the database (outside of any application implementations)**
7. **Will user preferences and settings be stored in SQL or would these be better in Redis?**  
   *Description: Decide whether frequently accessed user settings belong in PostgreSQL or if they'd benefit from Redis's faster access speed.*
   1. Unknown at this point.

**Task Management**

1. **What properties define your tasks?**  
   *Description: List all attributes that define a task (title, description, priority, deadline, etc.) to structure your tasks table.*
2. **How many levels of task hierarchy are needed?**  
   *Description: Determine whether you need simple tasks, tasks with subtasks, or deeper hierarchies, which affects your database relationship model.*
3. **What task states and transitions need to be tracked?**  
   *Description: Define the workflow states a task can exist in and the allowed transitions between states, which may require a state machine implementation.*
4. **How will you model task assignments?**  
   *Description: Decide whether tasks can be assigned to multiple users simultaneously and how to structure this many-to-many relationship.*
5. **How will you handle task dependencies and prerequisites?**  
   *Description: Determine how to represent tasks that depend on the completion of other tasks, which may require a self-referencing relationship.*
6. **What task metadata needs to be queryable vs. what could be stored as JSON?**  
   *Description: Identify which task attributes need to be indexed and searchable versus which could be stored in a JSON field for flexibility.*
7. **What performance considerations exist for task queries?**  
   *Description: Anticipate common task queries that might require indexing strategies or table partitioning for performance optimization.*

**Qualifications & Skills**

1. **How will you structure the relationship between users and qualifications?**  
   *Description: Decide whether this is a simple many-to-many relationship or if additional attributes like acquisition date or proficiency level are needed.*
2. **Will qualification requirements be tied to roles, tasks, or both?**  
   *Description: Determine whether certain roles require specific qualifications, tasks require qualifications, or both, affecting your relationship structure.*
3. **Do you need to track qualification history and expiration dates?**  
   *Description: Decide if you need to store when qualifications were obtained, when they expire, and historical qualification data.*
4. **How granular will skill levels be tracked?**  
   *Description: Define whether skills have levels (beginner, intermediate, expert) or numeric ratings, which affects your skills schema.*
5. **Will you need to query users by qualification for task assignment?**  
   *Description: Determine if you'll need optimized queries to find qualified users for specific tasks, which may impact indexing strategy.*

**Equipment & Resources**

1. **What equipment attributes need to be tracked in structured fields?**  
   *Description: List attributes like type, model, serial number, purchase date, etc. that will define your equipment table.*
2. **How will you model equipment allocation and availability?**  
   *Description: Decide how to track which equipment is assigned to whom, for how long, and its availability status.*
3. **Will you track equipment history and maintenance in SQL?**  
   *Description: Determine if you need to maintain a history of equipment usage, maintenance records, and lifecycle data.*
4. **How will you handle relationships between resources and tasks?**  
   *Description: Define how resources are allocated to tasks, whether they can be shared, and how scheduling conflicts are resolved.*
5. **What resource utilization metrics need to be queryable?**  
   *Description: Identify metrics like utilization rate, downtime, or efficiency that need to be calculated from the data.*

**MongoDB (Document Storage) Questions**

1. **What types of files will be attached to tasks?**  
   *Description: Identify the document formats, typical sizes, and special handling requirements for attached files.*
2. **Will you need to extract and index metadata from attachments?**  
   *Description: Determine if you need to parse documents for metadata (author, creation date, keywords) to make them searchable.*
3. **How will version control for documents be implemented?**  
   *Description: Decide whether you need to track document versions and how historical versions will be stored and accessed.*
4. **Will you need to query by document content or just by metadata?**  
   *Description: Consider whether full-text search within documents is required or if searching by metadata (filename, type, date) is sufficient.*
5. **How will document access control be handled?**  
   *Description: Determine if document permissions are inherited from the associated task or if they need separate access control.*
6. **Will you need to store document relationships?**  
   *Description: Decide if documents can reference other documents and how these relationships should be tracked.*
7. **What is your anticipated document storage volume and growth rate?**  
   *Description: Estimate storage needs based on number of documents, typical sizes, and growth projections for capacity planning.*
8. **Will you need to implement document retention policies?**  
   *Description: Determine if documents should be automatically archived or deleted after certain periods based on type or status.*

**Redis (Caching & Requests) Questions**

1. **What API responses are candidates for caching?**  
   *Description: Identify frequently accessed data that rarely changes, making it suitable for Redis caching to improve performance.*
2. **What is the appropriate TTL for different cached items?**  
   *Description: Determine how long different types of cached data should persist before being invalidated and refreshed.*
3. **How will you handle cache invalidation when data changes?**  
   *Description: Define strategies for updating or invalidating cached data when the source data is modified in PostgreSQL or MongoDB.*
4. **Will you use Redis for job queues? If so, which operations?**  
   *Description: Identify background tasks or resource-intensive operations that should be queued for asynchronous processing.*
5. **Do you need pub/sub functionality for real-time notifications?**  
   *Description: Determine if real-time updates (like task assignments or status changes) should be pushed to users via Redis pub/sub.*
6. **Will you use Redis for session management?**  
   *Description: Decide if user sessions should be stored in Redis for faster access and easier distribution across multiple servers.*
7. **What metrics will you track in Redis for performance monitoring?**  
   *Description: Identify key performance indicators that should be tracked in real-time for system monitoring and optimization.*
8. **How will you handle Redis persistence for critical data?**  
   *Description: Determine which Redis data needs to be persisted and what persistence configuration is appropriate for your needs.*

**Integration & Cross-Database Questions**

1. **How will you maintain referential integrity between SQL data and MongoDB documents?**  
   *Description: Define strategies to ensure data consistency when relationships span across PostgreSQL and MongoDB.*
2. **What strategy will you use for transactions that span multiple databases?**  
   *Description: Determine how to handle operations that need to update both SQL and NoSQL data atomically.*
3. **How will backups be coordinated across the different databases?**  
   *Description: Plan how to create consistent backups across PostgreSQL, MongoDB, and Redis for disaster recovery.*
4. **What schema migration strategy will you implement for PostgreSQL?**  
   *Description: Decide how database schema changes will be managed, versioned, and deployed for the relational database.*
5. **How will you handle SAP integration data that may need to be stored across databases?**  
   *Description: Determine where different types of SAP data belong (structured data in PostgreSQL, documents in MongoDB, cached values in Redis).*
6. **Will users need unified search across structured and unstructured data?**  
   *Description: Consider if users need to search across tasks, resources, and documents in a single query, which may require a search indexing solution.*
7. **What is your strategy for database connection pooling and management?**  
   *Description: Plan how database connections will be managed to optimize performance and resource utilization.*

**Performance & Scaling Questions**

1. **What are your expected query patterns and data access frequencies?**  
   *Description: Identify which data is accessed most frequently and the typical query patterns to optimize schema design.*
2. **What volume of concurrent users do you anticipate?**  
   *Description: Estimate the number of simultaneous users to plan for appropriate connection pooling and resource allocation.*
3. **Are there specific queries that will need optimization?**  
   *Description: Identify complex or frequent queries that might require specific optimization strategies like materialized views.*
4. **Will you implement database sharding or replication?**  
   *Description: Determine if data volumes or availability requirements necessitate distributing data across multiple servers.*
5. **What is your approach to index management for PostgreSQL?**  
   *Description: Plan how indexes will be created, monitored, and maintained to balance query performance and write overhead.*
6. **How will you monitor database performance?**  
   *Description: Define what metrics will be tracked to identify performance issues and how they'll be monitored and alerted on.*