**.1️ چطور یک Relational Database Schema برای یک فروشگاه آنلاین طراحی می‌کنم؟**

من اول میام بخش‌های مختلف فروشگاه رو در ذهنم مشخص می‌کنم. مثلاً:

**کاربر**

**محصول**

**سفارش**

**سبد خرید**

**پرداخت**

بعد میام برای هرکدوم جدول می‌سازم. مثلاً:

جدول **Users**: مشخصات کاربر (id, name, email)

جدول **Products**: مشخصات کالا (id, name, price, stock)

جدول **Orders**: سفارشات (id, user\_id, date)

جدول **OrderDetails**: جزئیات سفارش (id, order\_id, product\_id, quantity)

برای ارتباط بین جدول‌ها از **کلید خارجی (foreign key)** استفاده می‌کنم. مثلاً در جدول Orders ستون user\_id به جدول Users وصل میشه.

**.2️چطور یک NoSQL Database Schema برای سیستم پیام‌رسانی طراحی می‌کنم؟**

چون NoSQL ساختار انعطاف‌پذیرتری داره، من میام به صورت **دسته‌بندی‌شده (collection-based)** دیتاها رو می‌چینم.  
مثلاً:

**users collection**: اطلاعات کاربرا (id, name, profile)

**messages collection**: هر پیام شامل (id, sender\_id, receiver\_id, content, timestamp)

اگه بخوام چت گروهی داشته باشم:

**chats collection**: (id, title, members[])

**messages collection**: (chat\_id, sender\_id, content)

ساختارم رو جوری می‌چینم که هر اطلاعاتی سریع پیدا بشه و زیاد درگیر ارتباط جدول‌به‌جدول نشم.

**.3️ چطور یک Indexing Strategy برای پایگاه داده طراحی می‌کنم؟**

من میام:

فیلدهایی که بیشتر روشون سرچ میشه (مثل email, username, product name) رو **index** می‌کنم.

در NoSQL میشه روی فیلدهایی که زیاد فیلتر میشه (مثلاً messages by chat\_id) ایندکس گذاشت.

حواسم هست که **اضافه کردن زیاد ایندکس سرعت نوشتن رو کم می‌کنه** پس فقط جاهای لازم.

برای جداول خیلی بزرگ از **composite index** (مثلاً (user\_id, date)) هم استفاده می‌کنم.

**.4️ چطور Query Optimization برای پایگاه داده انجام می‌دم؟**

اول میام **query slow log** رو فعال می‌کنم ببینم کدوم query ها کندن.

بعد سعی می‌کنم:

از **index**‌ها به درستی استفاده کنم.

به‌جای SELECT \* فقط فیلدهای لازم رو بخونم.

queryها رو **join** کمتر بزنم و حتی‌الامکان ساده نگه دارم.

queryهای تکراری رو **کش** (cache) کنم.

**.5️ چطور Transaction Management System پیاده‌سازی می‌کنم؟**

برای این کار:

توی پایگاه داده رابطه‌ای از قابلیت **transaction** استفاده می‌کنم:

اول BEGIN TRANSACTION

بعد اجرای دستورات

اگه همه موفق بودن COMMIT

اگه خطا خورد ROLLBACK

اینطوری عملیات‌هایی که باید باهم انجام بشن (مثلاً همزمان ثبت سفارش و کم کردن موجودی) به درستی مدیریت میشن.

**.6️ چطور یک Backup and Recovery Plan طراحی می‌کنم؟**

مشخص می‌کنم:

**زمان‌بندی بکاپ** (روزانه/هفتگی)

**مکان ذخیره بکاپ** (روی سرور دیگه/ابر)

از ابزارهای پایگاه داده برای **full backup** و **incremental backup** استفاده می‌کنم.

**دوره‌ای تست ریکاوری** انجام میدم که مطمئن شم بکاپ‌ها سالم هستن.

در NoSQL هم snapshot می‌گیرم.

**.7️ چطور یک Sharding Strategy طراحی می‌کنم؟**

وقتی داده‌ها خیلی زیاد شدن:

داده‌ها رو **تکه‌تکه** (shard) می‌کنم.

مثلا بر اساس **محدوده کاربرها** (user\_id ۱ تا ۱۰۰۰ در shard ۱ و …)

یا بر اساس **کشور/منطقه**

حواسم هست shardها هم‌وزن باشن و جوری تقسیم کنم که کمترین وابستگی بین‌شون باشه.

**.8️ چطور یک Replication Strategy طراحی می‌کنم؟**

برای اطمینان از در دسترس بودن:

از **master-slave replication** استفاده می‌کنم (یعنی یک دیتابیس اصلی و چند کپی)

یا **multi-master** اگه نیاز باشه چندتا دیتابیس همزمان بنویسن.

برای NoSQL مثل MongoDB میشه **replica set** درست کرد.

مشخص می‌کنم چه دیتایی replicate بشه و هر replica چه زمانی sync بشه.

**.9️ چطور یک ETL Process برای Data Warehouse پیاده‌سازی می‌کنم؟**

ETL یعنی:

**Extract**: داده رو از منبع می‌خونم (مثل MySQL, API)

**Transform**: داده رو تمیز و تبدیل می‌کنم (مثلاً تاریخ‌ها رو استاندارد می‌کنم)

**Load**: توی دیتا‌ویرهاوس می‌ریزم (مثل SQL Server Data Warehouse)

برای این کار از ابزارهایی مثل **SSIS**, **Talend**، یا کدنویسی اختصاصی استفاده می‌کنم.

**.10چطور یک Data Migration Plan برای پروژه طراحی می‌کنم؟**

اول **مبدأ و مقصد داده**‌ها رو مشخص می‌کنم.

بعد **نقشه انتقال** می‌کشم (کدوم ستون به کدوم ستون)

یک **برنامه زمان‌بندی** می‌ذارم.

قبل از انتقال **بکاپ می‌گیرم**

**اسکریپت‌های انتقال** رو می‌نویسم و قبل از اجرا تو محیط تست امتحان می‌کنم.

بعد از انتقال، دیتا رو **بررسی و تایید** می‌کنم.

محمد اله مرادی