دانشگاه اصفهان

درس تحلیل سیستم‌های داده‌های حجیم

تکلیف فصل Data Lakehouse : طراحی و پیاده‌سازی Data Lakehouse برای تحلیل داده‌های فروش آنلاین

استاد درس: دکتر محمدعلی نعمت بخش

طراح تکلیف: مریم صفوی

مهلت ارسال: ۱۴۰۵/۰۳/۲۳

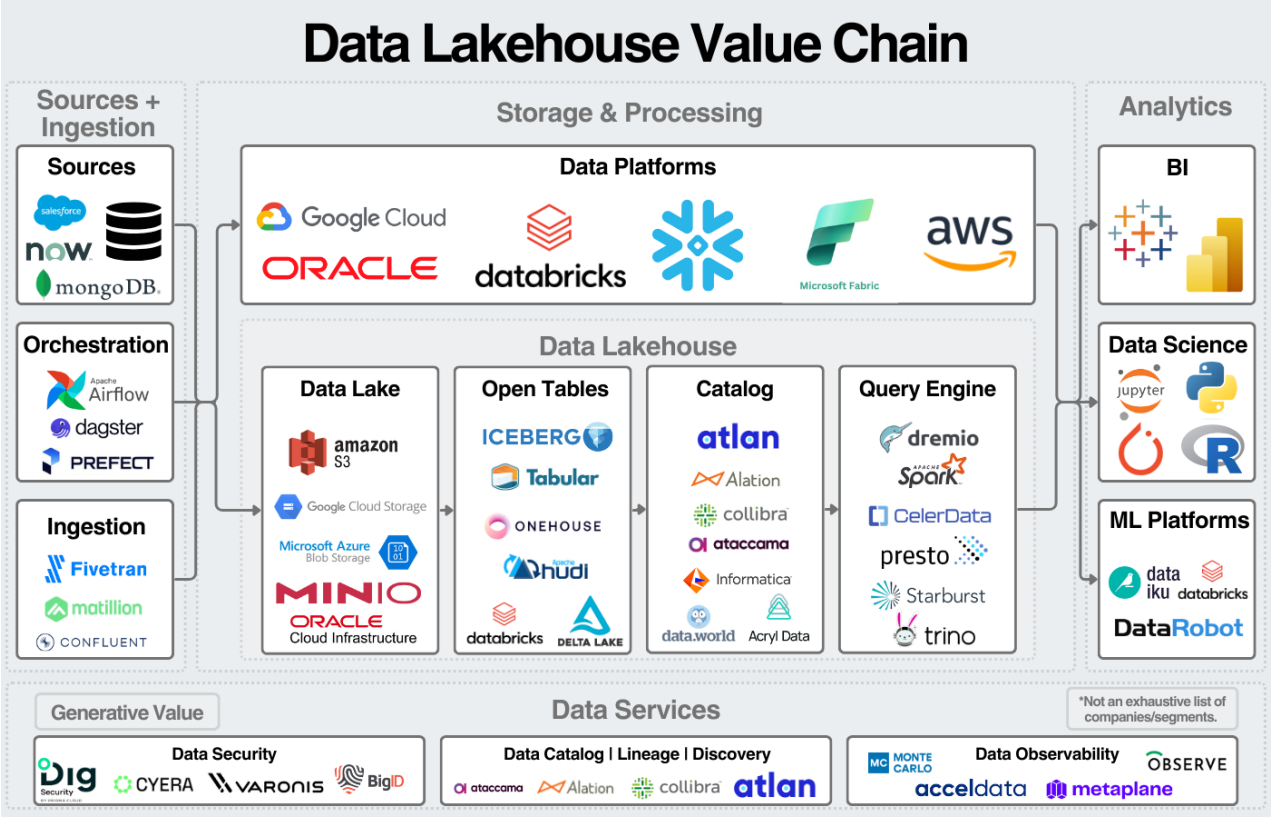
۱- مقدمه

در سال‌های اخیر، حجم داده‌هایی که توسط کسب‌وکارها تولید می‌شود به‌صورت چشمگیری افزایش یافته است. فروشگاه‌های اینترنتی، شبکه‌های اجتماعی، سامانه‌های بانکی، سرویس‌های حمل‌ونقل و حتی تجهیزات IoT روزانه میلیون‌ها event و رکورد داده تولید می‌کنند. در چنین شرایطی دیگر نمی‌توان داده‌ها را تنها با چند فایل CSV یا یک پایگاه داده سنتی مدیریت کرد. سازمان‌ها به معماری‌هایی نیاز دارند که هم توان ذخیره‌سازی داده خام را داشته باشند و هم بتوانند داده را برای تحلیل، گزارش‌گیری و تصمیم‌گیری آماده کنند.

یکی از مهم‌ترین معماری‌های مدرن در مهندسی داده، معماری Data Lakehouse است. این معماری تلاش می‌کند مزایای Data Lake و Data Warehouse را به‌صورت هم‌زمان در اختیار ما قرار دهد. در یک Lakehouse می‌توان داده خام را با هزینه کم ذخیره کرد و در عین حال قابلیت‌هایی مانند تراکنش، versioning، query بهینه و analytics را نیز داشت.

هدف این پروژه آن است که شما یک نمونه واقعی از چنین معماری‌ای را طراحی و پیاده‌سازی کنید. در طول پروژه تنها با مفاهیم تئوری مواجه نخواهید شد؛ بلکه دقیقاً همان فرآیندی را پیاده می‌کنید که در بسیاری از تیم‌های Data Engineering در صنعت انجام می‌شود.

در پایان این پروژه انتظار می‌رود بتوانید یک pipeline داده چندمرحله‌ای طراحی کنید، داده خام را ingest کنید، عملیات پاکسازی و استانداردسازی انجام دهید، داده تحلیلی تولید کنید و در نهایت نتایج را به‌صورت dashboard نمایش دهید.



تصویر ۱- زنجیره Data Lakehouse

۲- اهداف اصلی پروژه

در این پروژه شما یک Data Lakehouse مبتنی بر Apache Spark و Delta Lake طراحی می‌کنید که داده‌های فروش یک فروشگاه آنلاین را پردازش و تحلیل می‌کند.

داده‌های اولیه پروژه شامل تراکنش‌های فروش هستند. این داده‌ها در ابتدا خام، ناسازگار و تا حدی noisy هستند و باید طی چند مرحله به داده‌ای قابل استفاده برای تحلیل تجاری تبدیل شوند.

هدف اصلی پروژه تنها نوشتن چند query یا رسم نمودار نیست. آنچه اهمیت دارد، درک صحیح جریان داده در یک سیستم داده‌ای مدرن است. شما باید یاد بگیرید که چگونه داده از لحظه ورود به سیستم تا مرحله تولیدKPI های تجاری مدیریت می‌شود.

این پروژه ترکیبی از مفاهیم نظری و مهارت‌های عملی مهندسی داده است. انتظار می‌رود در طول انجام پروژه با مفاهیم زیر آشنا شوید:

- معماریData Lakehouse
- الگوی Bronze / Silver / Gold
- پردازش batch
- طراحی pipeline داده
- پاکسازی و استانداردسازی داده
- Data Quality
- کار با Apache Spark
- کار باDelta Lake یا Ice Berge
- طراحی داشبوردهای تحلیلی

۳- معرفی دیتاست پروژه

دیتاست پروژه شامل اطلاعات تراکنش‌های یک فروشگاه آنلاین است. هر سطر از داده نشان‌دهنده یک عملیات خرید یا بازگشت کالا است.

ساختار داده به شکل زیر است:

توضیح	ستون
شماره فاکتور	InvoiceNo
شناسه محصول	StockCode
نام محصول	Description
تعداد خرید	Quantity
زمان ثبت سفارش	InvoiceDate
قیمت هر واحد	UnitPrice

CustomerID	شناسه مشتری
Country	کشور مشتری

۴- معماری کلی پروژه

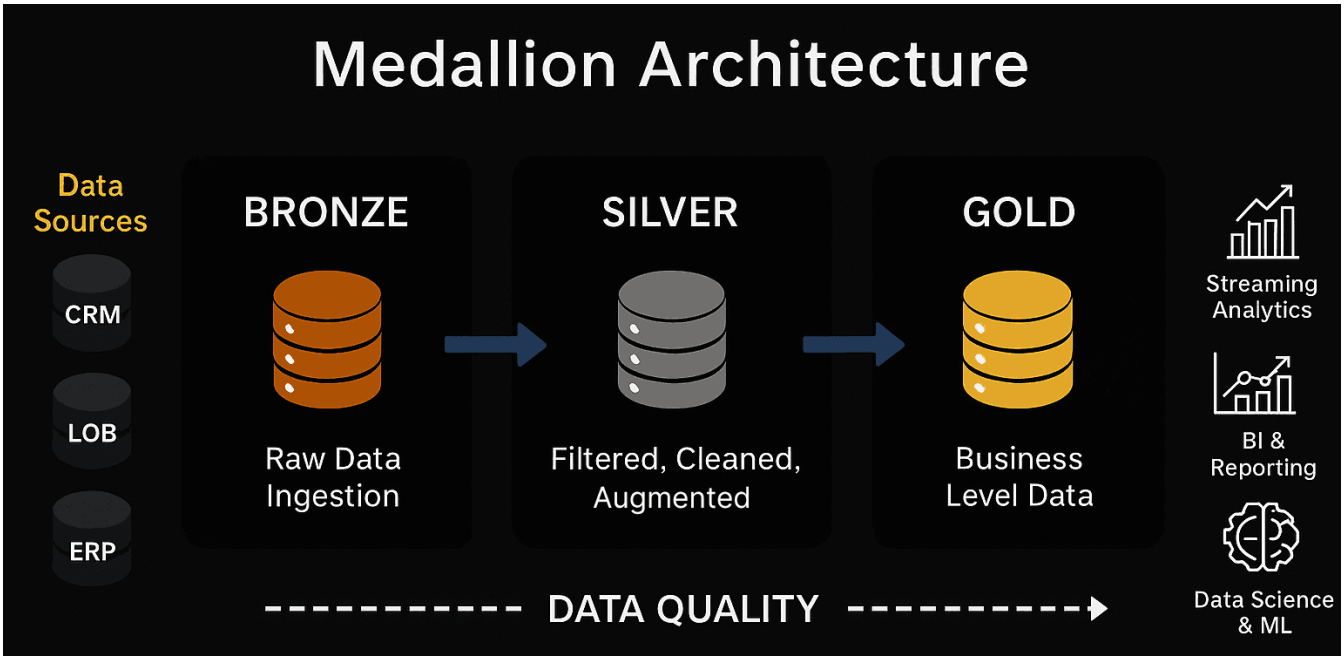
معماری پروژه مبتنی بر الگوی Medallion Architecture طراحی می‌شود. این معماری داده را در چند مرحله پردازش می‌کند تا کیفیت و قابلیت استفاده آن به‌تدریج افزایش یابد.

در این پروژه سه لایه اصلی خواهیم داشت:

Bronze Layer: این لایه محل ذخیره داده خام است. داده در این مرحله تقریباً بدون تغییر ذخیره می‌شود تا نسخه اصلی اطلاعات همیشه در دسترس باشد. وجود این لایه اهمیت بسیار زیادی دارد، زیرا در سیستم‌های واقعی ممکن است بعداً نیاز به بازپردازش داده یا بررسی خطاها داشته باشیم. اگر داده خام را از بین ببریم، بسیاری از این عملیات غیرممکن می‌شوند.

Silver Layer: در این لایه داده پاکسازی و استانداردسازی می‌شود. داده‌های ناسالم حذف می‌شوند، فرمت‌ها یکسان می‌شوند و ساختار داده برای استفاده مهندسی آماده می‌شود. Silver Layer معمولاً مهم‌ترین بخش pipeline های داده است، زیرا کیفیت خروجی نهایی مستقیماً به کیفیت عملیات این مرحله وابسته است.

Gold Layer: در Gold Layer داده برای تحلیل تجاری و dashboard آماده می‌شود. در این مرحله معمولاً داده aggregate شده و KPIهای اصلی کسب‌وکار تولید می‌شوند. مدیران و تیم‌های تحلیلی معمولاً مستقیماً با Gold Layer کار می‌کنند.



تصویر ۲- شِمای کلی معماری Medallion

۵- ابزارهای پروژه

در این پروژه از مجموعه‌ای از ابزارهای رایج در صنعت استفاده می‌شود.

Apache Spark

Spark موتور اصلی پردازش داده در پروژه است. این ابزار برای پردازش distributed data طراحی شده و یکی از استانداردهای صنعت Big Data محسوب می‌شود. مزیت مهم Spark این است که می‌تواند هم batch processing و هم stream processing را پشتیبانی کند.

Apache Iceberg / Delta Lake

در این پروژه برای پیاده‌سازی لایه Data Lakehouse از فرمت‌های جدول باز (Open Table Formats) استفاده می‌شود. Delta Lake لایه‌ای روی فرمت Parquet است که قابلیت‌هایی مانند ACID Transaction، Versioning و Time Travel را فراهم می‌کند. شما همچنین می‌توانید به صورت اختیاری از Apache Iceberg استفاده کنید که قابلیت‌های مشابهی را با معماری مستقل‌تری ارائه می‌دهد. وجود این ابزارها باعث می‌شود معماری Data Lake رفتار قابل‌اعتمادتر و نزدیک‌تری به Data Warehouse پیدا کند.

Docker

برای جلوگیری از مشکلات نصب و dependency ، تمام سرویس‌ها داخل container اجرا می‌شوند. هدف این است که تمام گروه‌ها محیطی یکسان و قابل تکرار داشته باشند.

۶- فازهای پروژه

فاز اول) راه‌اندازی محیط پروژه

اولین مرحله پروژه آماده‌سازی زیرساخت اجرا است. در این مرحله باید سرویس‌های اصلی پروژه را با استفاده از Docker Compose اجرا کنید. انتظار می‌رود پس از اجرای docker compose سرویس‌های زیر در دسترس باشند:

- Spark
- MinIO

هدف این مرحله صرفاً اجرای ابزارها نیست. شما باید متوجه شوید هر سرویس چه نقشی در معماری کلی پروژه دارد.

برای مثال:

- Spark وظیفه پردازش داده را برعهده دارد
- MinIO نقش object storage را بازی می‌کند
- Delta Lake یا Apache Iceberg روی storage قابلیت transactional ایجاد می‌کند.

در پایان این مرحله باید بتوانید به UI سرویس‌ها متصل شوید و از سلامت محیط مطمئن شوید.

فاز دوم) ورود داده به Bronze Layer

پس از آماده‌سازی محیط، باید داده خام را وارد سیستم کنید.

در این مرحله داده CSV اولیه بدون تغییر اساسی ingest می‌شود. هدف Bronze Layer نگهداری نسخه اصلی داده است، بنابراین نباید عملیات cleaning سنگین روی داده انجام شود.

با این حال، لازم است metadataهای مهمی به داده اضافه شود؛ برای مثال زمان ingestion

این موضوع در سیستم‌های واقعی اهمیت زیادی دارد، زیرا تیم‌های داده باید بدانند هر داده دقیقاً چه زمانی وارد pipeline شده است.

در این مرحله باید داده را با Spark بخوانید و در قالب Delta Table ذخیره کنید.

همچنین توصیه می‌شود ساختار پوشه‌بندی پروژه را از ابتدا منظم طراحی کنید. معماری خوب تنها مربوط به کد نیست؛ سازمان‌دهی داده نیز بخش مهمی از مهندسی داده محسوب می‌شود.

در این پروژه، قالب پیش‌فرض جدول Delta Lake (Table Format) در نظر گرفته شده است؛ با این حال، شما مجاز هستید به جای آن از Apache Iceberg استفاده کنید. هر دو ابزار استانداردهای روز صنعت برای معماری Data Lakehouse هستند.

در صورت تمایل به انتخاب، می‌توانید مقایسه زیر را در نظر بگیرید:

۱. Delta Lake

مزایا: یکپارچگی بسیار عمیق و بومی با Apache Spark، راه‌اندازی بسیار آسان (به ویژه در محیط‌های لوکال)، مستندات غنی و پیاده‌سازی ساده ویژگی‌هایی مانند Time Travel.

معایب: به لحاظ تاریخی وابستگی بیشتری به اکوسیستم Databricks دارد و ممکن است در اتصال به برخی موتورهای پردازشی دیگر به اندازه Iceberg انعطاف‌پذیر نباشد.

۲. Apache Iceberg

مزایا: کاملاً مستقل از موتور پردازشی با پشتیبانی عالی از Trino، Flink و Spark، ویژگی‌های پیشرفته در مدیریت اسکیما و پارتیشن‌بندی پنهان.

معایب: راه‌اندازی اولیه آن (مخصوصاً کانفیگ Catalog در محیط لوکال) ممکن است کمی پیچیده‌تر و زمان‌برتر از Delta Lake باشد.

نکته: اگر تصمیم به استفاده از Apache Iceberg گرفتید، در فاز Time Travel باید دستورات مربوط به Snapshotهای Iceberg را جایگزین دستورات Delta Lake کنید.

فاز سوم) تحلیل کیفیت داده

پیش از شروع عملیات cleaning، لازم است داده را به‌خوبی بشناسید.

بسیاری از پروژه‌های داده به این دلیل شکست می‌خورند که تیم مهندسی بدون شناخت واقعی داده شروع به طراحی pipeline می‌کند.

در این مرحله باید رفتار داده را تحلیل کنید.

برای مثال:

- چه تعداد CustomerID خالی وجود دارد؟
- چه تعداد Quantity منفی است؟

- آیا رکورد تکراری وجود دارد؟
- آیا همه قیمت‌ها معتبر هستند؟

هدف این مرحله تنها تولید آمار نیست. شما باید یاد بگیرید که چگونه کیفیت داده را ارزیابی کنید و تصمیم بگیرید چه داده‌ای برای تحلیل مناسب است.

فاز چهارم) ساخت Silver Layer

در این مرحله داده خام به داده تمیز و استاندارد تبدیل می‌شود.

عملیات cleaning باید بر اساس تحلیل مرحله قبل انجام شود، نه به‌صورت تصادفی. برای مثال اگر متوجه شدید برخی Quantity ها منفی هستند، ابتدا باید مشخص کنید آیا این داده اشتباه است یا نشان‌دهنده cancellation. در پروژه‌های واقعی حذف اشتباه داده می‌تواند باعث ایجاد خطاهای جدی تحلیلی شود.

در Silver Layer معمولاً عملیات زیر انجام می‌شود:

- حذف داده‌های نامعتبر
- حذف duplicate
- تبدیل فرمت تاریخ
- مدیریت null value
- استانداردسازی schema

در این مرحله باید درباره تصمیم‌های مهندسی خود فکر کنید.

برای مثال:

اگر CustomerID خالی باشد، آیا رکورد حذف شود یا به‌عنوان Guest در نظر گرفته شود؟

در پایان این مرحله باید یک Silver Delta Table قابل اعتماد داشته باشید.

فاز پنجم) طراحی Gold Layer و KPIها

تا این مرحله داده تمیز و استاندارد شده است. اکنون باید داده را به فرم مناسب برای تحلیل تجاری تبدیل کنید.

Gold Layer معمولاً برای استفاده dashboard، BI و تحلیل مدیریتی طراحی می‌شود. در این پروژه چند KPI اصلی تعریف می‌شود.

اولین KPI میزان فروش روزانه است. برای محاسبه درآمد باید تعداد فروش در قیمت هر واحد ضرب شود.

$$Revenue = Quantity \times UnitPrice$$

سپس می‌توانید فروش را بر اساس روز aggregate کنید.

علاوه بر آن باید تحلیل‌هایی مانند موارد زیر را نیز پیاده‌سازی کنید:

- محصولات پرفروش

- فروش بر اساس کشور
- بهترین مشتری‌ها
- روند فروش ماهانه

در این مرحله مهم است که به performanceها query نیز توجه کنید. معمولاً Gold Layer به‌صورت denormalized طراحی می‌شود تا dashboardها سریع‌تر اجرا شوند.

✓ **نمایش قابلیت Time Travel در Delta Lake**

برای درک عملی تفاوت معماری Lakehouse با فایل‌های ساده، سناریوی زیر را روی یکی از جداول دلتا پیاده‌سازی و در نوتبوک خود مستند کنید:

آپدیت داده: یک رکورد خاص (مثلاً یک InvoiceNo مشخص) را پیدا کنید و مقدار UnitPrice یا Quantity آن را با دستور UPDATE تغییر دهید.

سفر در زمان (Time Travel): با استفاده از قابلیت Time Travel در دلتا لیک (استفاده از سینتکس VERSION AS OF یا TIMESTAMP AS OF)، کوئری بنویسید که داده‌های این جدول را دقیقاً در حالتِ قبل از آپدیت بخواند و رکورد قدیمی را نمایش دهد.

فاز ششم) طراحی Dashboard

پس از تولید Gold Layer باید داده را به‌شکل قابل فهم برای تحلیل‌گر یا مدیر نمایش دهید.

هدف dashboard صرفاً زیبایی بصری نیست. یک dashboard خوب باید بتواند insight تولید کند و به تصمیم‌گیری کمک کند.

پیشنهاد می‌شود dashboardهای زیر طراحی شوند:

- روند فروش روزانه
- فروش بر اساس کشور
- محصولات پرفروش
- تحلیل مشتریان
- روند فروش ماهانه

در این مرحله باید یاد بگیرید که چگونه داده مهندسی‌شده را به اطلاعات قابل استفاده برای business تبدیل کنید.

«برای این بخش الزامی به راه‌اندازی ابزارهای پیچیده BI نیست. داشبورد نهایی می‌تواند به صورت یک وب‌اپلیکیشن ساده و تعاملی با استفاده از فریم‌ورک‌های پایتونی مانند Streamlit یا Dash توسعه داده شود. همچنین در صورت تمایل، می‌توانید از Apache Superset به عنوان یک ابزار BI متن‌باز برای اتصال به داده‌ها و ساخت داشبورد استفاده کنید.

این داشبورد باید مستقیماً داده‌های تجمیع‌شده را از لایه Gold بخواند و نمودارها و KPIهای خواسته شده (روند فروش روزانه، محصولات برتر و ...) را به تصویر بکشد.

۷- مستندسازی پروژه

در گزارش پروژه باید توضیح دهد:

- معماری سیستم چگونه طراحی شده است
- چرا از ابزارهای خاص استفاده شده است
- چه عملیات cleaning انجام شده است
- Pipeline چگونه کار می‌کند
- چه چالش‌هایی وجود داشته است
- چه تحلیل‌هایی تولید شده است
- دایاگرام نهایی پروژه

هدف گزارش تنها توضیح کد نیست؛ بلکه باید منطق مهندسی پروژه را نیز توضیح دهد.

۸- تحویل نهایی پروژه

باید موارد زیر را تحویل دهد:

- GitHub Repository (دلبخواه، می‌توانید کدها را مستقیما ارسال کنید)
- Docker Compose
- Notebookها وScriptها
- Dashboard نهایی
- فایل گزارش PDF
- نمودار معماری سیستم

۹- نحوه ارسال تمرین

دانشجویان موظف هستند فایل‌ها را در قالب یک فایل فشرده (ZIP) با فرمت نام‌گذاری HW03-BD-StuID-MongoDB.zip در سامانه کوئرا بارگذاری نمایند.

نکته ۱: درصورت عدم دسترسی به سامانه کوئرا می‌توانید فایل‌های خود یا لینک دسترسی به آن‌ها را از طریق راه‌های ارتباطی زیر، ارسال بفرمایید:

آیدی تلگرام: @Maryam_Safavi

آیدی بله: @Mary _bale

آیدی ایتا: @Mary_eitaa

نکته ۲: درصورتی که محدودیت حجم برای ارسال فایل داشتید، می‌توانید از سایت‌های آنلاین آپلود فایل که در بخش "پیشنهادها" در تمرین دوم معرفی شده‌اند، استفاده کنید و لینک آن را ارسال بفرمایید.

مهلت ارسال تکلیف: ۲۳ اردیبهشت ۱۴۰۵

با آرزوی موفقیت برای شما

صفوی