به نام خداوند بخشنده مهربان

# طراحی پایگاه داده ها

دانشکده مهندسی کامپیوتر

مریم رمضانی
بهار ۱۴۰۳

## پروژه (سرویس Azure Functions)

تاریخ انتشار: ۱۲ خرداد ۱۴۰۴

### معرفی دیتاست

این دیتاست شامل مجموعه‌ای از فایل‌هاست که بخشی از بار کاری سرویس Azure Functions مایکروسافت مربوط به سال ۲۰۱۹ را نشان می‌دهد. Azure Functions یک سرویس serverless از مایکروسافت است که به توسعه‌دهندگان امکان می‌دهد تا کدی بنویسند که منطق کسب‌وکار را پیاده‌سازی می‌کند، بدون اینکه نیاز به مدیریت زیرساخت داشته باشند. Azure Functions در سناریوهایی مانند پردازش فایل‌ها، اتوماسیون فرایندها، یا پاسخ به درخواست‌های HTTP به‌صورت مقیاس‌پذیر و منعطف کاربرد دارد.
این دیتاست یک بازه‌ی ۱۴ روزه را پوشش می‌دهد و شامل سه بخش اصلی است:

- بخش اول مربوط به تعداد فراخوانی توابع در هر دقیقه‌ی شبانه‌روز است که در ۱۴ فایل مجزا ذخیره شده.
- بخش دوم شامل مدت‌زمان اجرای توابع است و آمارهایی مثل میانگین زمان و تعداد اجرا را در هر روز ارائه می‌دهد.
- بخش سوم اطلاعات حافظه مصرفی برنامه‌ها را پوشش می‌دهد.

در هر بخش، توابع با استفاده از شناسه‌های هش شده برای مالک برنامه و تابع، ناشناس‌سازی شده‌اند.
هدف این دیتاست تحلیل بار کاری، عملکرد، و رفتار منابع Azure Functions در محیط واقعی است. و می‌توان از آن برای تحلیل الگوهای استفاده، بهینه‌سازی منابع، و توسعه سیستم‌های بدون سرور استفاده کرد. برای توضیحات بیشتر و دانلود دیتاست به این لینک مراجعه کنید.

### تکنولوژی‌های مناسب برای ذخیره‌سازی و کار با دیتاست

از آنجا که این دیتاست دارای ساختار زمانی دقیق به صورت دقیقه به دقیقه در طول روز است، تکنولوژی‌های timeseries مانند Prometheus یا InfluxDB می‌توانند گزینه‌های قابل توجهی باشند. همچنین دیتابیس‌های Relational نیز به علت قابلیت خوب نرمال‌سازی دیتاست گزینه مناسبی هستند. با این حال با توجه به بررسی‌های خود هر تکنولوژی دلخواهی را می‌توانید، انتخاب کنید.

### پیش‌پردازش و ذخیره‌سازی دیتاست

شما لازم است، با بررسی دیتاست خود و نیازمندهای مطرح شده در بخش‌های قبل یک تکنولوژی مناسب برای کار با این دیتاست را انتخاب کنید. در این بخش باید با ارائه دلایل کافی، علت انتخاب خود را شرح دهید. همچنین انتخاب‌های ممکن دیگر را مطرح کنید و موارد برتری انتخاب خود نسبت به سایر گزینه‌ها را در گزارش خود ارائه دهید.
در مرحله بعد، لازم است که با توجه به نیازمندی‌های بخش‌های بعد، در ابتدا دیتاست را پیش‌پردازش کنید و در صورت وجود داده‌های نامعتبر و بی‌معنی آنها را حذف کنید. برنامه پیش‌پردازش شما باید قابلیت دریافت داده را به صورت مداوم داشته باشد. به این معنی که برنامه پیش‌پردازش همواره در حال اجراست و داده‌های ورودی را به صورت stream دریافت می‌کند و داده‌های پردازش‌شده را به صورت stream خروجی بدهد.
پس از آن لازم است که یک پیکربندی، برای ذخیره‌سازی دیتاست خود در تکنولوژی انتخاب شده ارائه دهید و مستند طراحی آن را همراه با گزارش خود ارائه دهید (برای مثال در صورتی که تکنولوژی انتخابی شما برای انجام پروژه یک دیتابیس relational بود، از ERD استفاده کنید).
سپس یک برنامه توسعه دهید تا داده‌ها رو به صورت مداوم از خروجی برنامه پیش‌پردازش تحویل بگیرد و طبق پیکربندی از پیش تعیین شده از بر روی تکنولوژی انتخاب شما ذخیره کند.
اکثر دیتابیس‌ها و تکنولوژی‌ها ذخیره‌سازی داده، متریک‌هایی از وضعیت خود و کوئری‌های اجرا شده ارائه می‌دهند. حداقل ۵ مورد از متریک‌های کاربردی تکنولوژی انتخابی خود را همراه با نمونه خروجی آن نمایش دهید.

### کار با داده ذخیره شده

در این مرحله لازم است، نیازمندی‌های تعریف شده را با نوشتن کوئری‌های مناسب بدست آورید. در اجرای هر کدام از کوئری‌ها تاخیر زمان اجرای آنها (latency) و throughput انجام آنها در واحد زمان را اندازه‌گیری کنید. علاوه بر این موارد، متریک‌های مخصوص تکنولوژی خود را که در بخش قبل بررسی کرده بودید، پیش، هنگام و پس از اجرا هر کدام از کوئری‌ها اندازه‌گیری کنید.
نیازمندی‌ها ارائه شده به شرح زیر هستند:

- پیدا کردن ۱۰ تابعی که بیشترین اوج فراخوانی در یک دقیقه را داشتند (شامل مشخص کردن دقیقه اوج).
- شناسایی توابعی که مستقر شده‌اند ولی حداقل به مدت ۱۲ ساعت متوالی هیچ فراخوانی نداشتند (نشانه‌ای از هدر رفت منابع).
- محاسبه ضریب تغییرات (انحراف معیار تقسیم بر میانگین) برای تعداد فراخوانی هر تابع در طول ۱۴ روز برای یافتن توابع با الگوی استفاده ناپایدار.

- بررسی رابطه بین مدت زمان اجرای توابع، با تعداد فراخوانی‌ها (برای کنترل تأثیر تعداد فراخوانی).
- تجمیع مجموع فراخوانی‌ها، میانگین مدت زمان اجرا، و میانگین حافظه تخصیص داده‌شده بر اساس مالک (Owner) برای مشخص کردن پرمصرف‌ترین مالکان.
- شناسایی توابعی که در یک روز خاص تعداد فراخوانی‌شان بیش از ۲ برابر میانگین روزهای دیگر بوده (برای تشخیص ناهنجاری).
- پیدا کردن اپلیکیشن‌ها با حافظه زیاد اما استفاده کم (با توجه به تعداد اجرا و مدت زمان اجرا) که می‌توان آن‌ها را کوچک‌سازی کرد.
- ساخت یک معیار **اهمیت** برای هر تابع بر اساس مجموع فراخوانی، مدت زمان اجرا و مصرف حافظه برای تعیین توابع حیاتی‌تر.
- شناسایی توابع با نوسان زیاد در مدت زمان اجرا (اختلاف زیاد بین صدک ۷۵ و صدک ۹۵) که ممکن است ریسک نقض SLA داشته باشند.
- یک برنامه‌ی خاص را انتخاب کنید و بررسی کنید حافظه‌ی استفاده‌شده‌ی میانگین آن در طول ۱۲ روز چطور تغییر کرده؟

حال حداقل ۱۰ ترکیب مختلف از اجرای همزمان این نیازمندی‌ها را به واسطه چند کلاینت مختلف انجام دهید. تعداد کلاینت‌ها و کوئری‌ها همزمان را به نحوی انتخاب کنید تا کاهش کارایی به صورت قابل توجهی مشاهده شود.

## بهینه‌سازی نحوه کار با داده

در رابطه با روش‌های بهینه‌سازی ذخیره‌سازی داده در تکنولوژی خود تحقیق کنید و نحوه عملکرد و تاثیر آن‌ها را بر روی نحوه ذخیره‌سازی داده را بررسی کنید. برای مثال روی انواع ایندکس‌ها در تکنولوژی‌ها انتخابی خود تحقیق کنید.

حال هر کدام از نیازمندی‌های مطرح شده در بخش قبل را با روش‌هایی که تحقیق کردید، بهینه کنید. سپس با اجرای دوباره آنها، تاثیر بهینه‌سازی را روی مواردی که در بخش قبل اندازه‌گیری کرده‌اید، نشان دهید.

برای هرکدام از موارد بهینه‌سازی، تاثیرات جانبی مثبت و منفی حاصل از بهینه‌سازی را بررسی کنید و نشان دهید. برای مثال می‌توان افزایش حجم ذخیره شده را به عنوان یکی از نتایج ایندکس‌گذاری نشان داد.

سناریوهای دارای چند کلاینت در بخش قبل را پس از اعمال بهینه‌سازی‌ها اجرا کنید و نتایج را مقایسه کنید. سپس بر روی روش‌های بهبود عملکرد استفاده همزمان چند کلاینت تحقیق کنید و با استفاده از آن‌ها نتایج بهبود داده شده را مقایسه کنید (برای مثال در رابطه با ایجاد Connection Pool تحقیق کنید).