نمونه کوئری‌های SQL: از اصول تا کاربردهای حرفه‌ای

راهنمای جامع برای تحلیل و مدیریت داده کاری از فاتحی اسکول

تاریخ: 26 نوامبر 2025
منبع: fatehischool.ir
چاپ: 1.0

چکیده ا

SQL (زبان ساخت یافته پرس و جو) استاندارد جهانی برای تعامل با پایگاه‌های داده رابطه‌ای است که در تمامی صنایع از بانکداری تا تجارت الکترونیک کاربرد دارد. این راهنما به بررسی ساختارهای پایه‌ای کوئری‌های SQL از جمله SELECT، FROM، WHERE، انواع JOIN، زیرکوئری‌ها، توابع تجمعی و پنجره‌ای، روش‌های بهینه‌سازی عملکرد، و امنیت می‌پردازد. همچنین نکات کلیدی نگارش کدنویسی تمیز، تمرین‌های پیشنهادی و کاربردهای واقعی در صنایع مختلف را نیز پوشش می‌دهد.

نکات کلیدی

- ساختار اصلی کوئری‌های SQL شامل سه بخش SELECT (انتخاب ستون‌ها)، FROM (منبع داده) و WHERE (فیلتر) است.
- JOINها ابزار اصلی برای ترکیب داده‌های چند جدول هستند که شامل INNER، LEFT، RIGHT و FULL JOIN می‌شوند.
- توابع تجمعی مانند AVG، SUM، COUNT برای تحلیل داده‌ها ضروری هستند و با GROUP BY و HAVING کاربرد دارند.
- زیرکوئری‌ها (Subqueries) کوئری‌هایی درون کوئری‌های دیگر هستند که می‌توانند مستقل یا همبسته باشند.
- بهینه‌سازی عملکرد کوئری با استفاده از ایندکس‌ها و ابزارهایی مانند EXPLAIN اهمیت زیادی دارد.
- کدنویسی تمیز SQL شامل فرمت‌بندی مناسب، نام‌گذاری یکنواخت و مستندسازی است.
- امنیت کوئری‌ها با پارامترایزیشن و جلوگیری از SQL Injection تضمین می‌شود.

ساختار پایه‌ای کوئری‌ها

در قلب هر کوئری SQL، سه جزء اصلی وجود دارد:

- **SELECT**: مشخص می‌کند که کدام ستون‌ها باید نمایش داده شوند.

- **FROM**: منبع داده را تعیین می‌کند (جدول، ویو یا زیرکوئری).
- **WHERE**: امکان فیلتر کردن رکوردها را بر اساس شرایط منطقی فراهم می‌کند.

مثال ساده:

```
SELECT first_name, last_name
FROM employees
;'WHERE department = 'IT
```

> **نکته:** استفاده از * به معنای انتخاب همه ستون‌ها است، اما توصیه می‌شود فقط ستون‌های مورد نیاز را انتخاب کنید تا عملکرد و خوانایی کوئری بهبود یابد.

گروه‌بندی و تجمیع

برای تحلیل داده‌ها و استخراج اطلاعات خلاصه، از توابع تجمعی مانند MAX، MIN، AVG، SUM و COUNT و دستور GROUP BY استفاده می‌شود.
دستور HAVING مشابه WHERE عمل می‌کند، اما برای فیلتر کردن گروه‌ها پس از اعمال توابع تجمعی استفاده می‌شود. تفاوت کلیدی این است: WHERE قبل از GROUP BY و HAVING بعد از آن اجرا می‌شود.

مثال:

```
SELECT department, COUNT(*) AS employee_count, AVG(salary) AS avg_salary
FROM employees
GROUP BY department
;HAVING AVG(salary) > 70000
```

انواع JOIN

در پایگاه‌های داده رابطه‌ای، داده‌ها معمولاً در جداول مختلف ذخیره می‌شوند و برای استخراج اطلاعات ترکیبی، باید این جداول را به هم متصل کنیم. JOINها ابزار اصلی برای این کار هستند.

نوع JOIN	توضیح مختصر
INNER JOIN	فقط رکوردهایی که در هر دو جدول تطابق دارند را بازمی‌گرداند.
LEFT (OUTER) JOIN	همه رکوردهای جدول سمت چپ و رکوردهای تطبیق یافته از جدول سمت راست (در صورت وجود) را بازمی‌گرداند.
RIGHT (OUTER) JOIN	همه رکوردهای جدول سمت راست و رکوردهای تطبیق یافته از جدول سمت چپ (در صورت وجود) را بازمی‌گرداند.
FULL (OUTER) JOIN	همه رکوردهای هر دو جدول را بازمی‌گرداند، حتی اگر تطابقی وجود نداشته باشد.
CROSS JOIN	حاصل ضرب دکارتی دو جدول؛ هر رکورد از جدول اول با هر رکورد از جدول دوم ترکیب می‌شود.

مثال INNER JOIN:

```
SELECT e.first_name, d.department_name
FROM employees e
;INNER JOIN departments d ON e.dept_id = d.dept_id
```

مثال LEFT JOIN:

```
SELECT c.customer_name, o.order_id
FROM customers c
;LEFT JOIN orders o ON c.customer_id = o.customer_id
```

زیرکوئری‌ها

زیرکوئری (Subquery) کوئری‌ای است که درون کوئری دیگر قرار می‌گیرد و می‌تواند در بخش‌های SELECT، FROM یا WHERE استفاده شود. زیرکوئری‌ها به دو دسته اصلی تقسیم می‌شوند:

- **زیرکوئری مستقل (Uncorrelated Subquery)**: مستقل از کوئری بیرونی اجرا می‌شود.
- **زیرکوئری همبسته (Correlated Subquery)**: به هر رکورد کوئری بیرونی وابسته است و برای هر رکورد اجرا می‌شود.

مثال زیرکوئری مستقل:

```
SELECT name, salary
FROM employees
;WHERE salary > (SELECT AVG(salary) FROM employees)
```

مثال زیرکوئری همبسته:

```
SELECT e1.name, e1.salary
FROM employees e1
WHERE salary > (SELECT AVG(salary) FROM employees e2 WHERE e2.department =
;e1.department)
```

> **نکته:** در بسیاری از موارد، می‌توان زیرکوئری‌های همبسته را با JOIN بازنویسی کرد تا عملکرد بهتری داشته باشند.

توابع رشته‌ای و تاریخ

SQL مجموعه‌ای از توابع قدرتمند برای کار با داده‌های متنی (رشته‌ای) و تاریخ/زمان ارائه می‌دهد.

توابع رشته‌ای:

تابع	کاربرد	مثال
CONCAT	الحاق چند رشته به هم	CONCAT(first_name, ' ', last_name)

تابع	کاربرد	مثال
LENGTH	طول رشته	LENGTH(name)
SUBSTRING	استخراج بخشی از رشته	SUBSTRING(name, 1, 3)
UPPER/LOWER	تبدیل به حروف بزرگ/کوچک	UPPER(name)
TRIM	حذف فاصله‌های ابتدا و انتهای رشته	TRIM(' hello ')

توابع تاریخ:

تابع	کاربرد	مثال
NOW()	تاریخ و زمان فعلی	SELECT NOW()
CURDATE()	تاریخ فعلی	SELECT CURDATE()
DATEDIFF	اختلاف دو تاریخ	DATEDIFF(end_date, start_date)
YEAR/MONTH/DAY	استخراج سال/ماه/روز	YEAR(birthdate)

فیلترها و عملگرها

WHERE امکان فیلتر کردن رکوردها را بر اساس شرایط مختلف فراهم می‌کند.

- **BETWEEN**: بررسی بازه‌ای از مقادیر (شامل ابتدا و انتها) - مثال: WHERE price BETWEEN 1000 AND 5000
- **IN**: بررسی وجود مقدار در یک لیست - مثال: WHERE city IN ('Tehran', 'Mashhad', 'Isfahan')
- **LIKE**: جستجوی الگوهای متنی - مثال: WHERE name LIKE 'Ali%'
- **IS NULL / IS NOT NULL**: بررسی مقادیر تهی - مثال: WHERE phone IS NULL

مثال ترکیبی:

```
SELECT * 
FROM products
WHERE category = 'Electronics'
AND price BETWEEN 1000 AND 5000
AND name LIKE '%Phone%'
AND stock IS NOT NULL;
```

بهینه‌سازی عملکرد

عملکرد کوئری‌ها در پایگاه‌های داده بزرگ اهمیت زیادی دارد. کوئری‌های ناکارآمد می‌توانند منجر به کندی سیستم شوند.

- از SELECT * پرهیز کنید؛ فقط ستون‌های مورد نیاز را انتخاب کنید.
- از ایندکس‌ها روی ستون‌های پرتکرار استفاده کنید.
- توابع را در WHERE روی ستون‌ها به کار نبرید (مانع استفاده از ایندکس می‌شود).
- از EXISTS به جای IN برای مجموعه‌های بزرگ استفاده کنید.

روش‌های پیشرفته

توابع پنجره‌ای:

توابع پنجره‌ای مانند SUM() OVER، RANK()، ROW_NUMBER() امکان انجام محاسبات تجمعی و رتبه‌بندی روی مجموعه‌ای از رکوردها را بدون گروه‌بندی کل جدول فراهم می‌کنند.

```
SELECT name, salary,
RANK() OVER (ORDER BY salary DESC) AS salary_rank
FROM employees;
```

CTE (Common Table Expression):

CTEها بخش‌هایی از کوئری‌ها هستند که با دستور WITH تعریف می‌شوند و خوانایی کوئری‌های پیچیده را افزایش می‌دهند.

امنیت و SQL Injection

SQL Injection یکی از رایج‌ترین آسیب‌پذیری‌ها در برنامه‌های مبتنی بر پایگاه داده است. راهکارهای جلوگیری:

- استفاده از کوئری‌های پارامترایزشده (Prepared Statements)
- اعتبارسازی و پاک‌سازی ورودی‌ها
- استفاده از حداقل سطح دسترسی

قوانین نگارشی و سبک کدنویسی

- کلیدواژه‌ها را با حروف بزرگ بنویسید: SELECT, FROM, WHERE
- نام جداول و ستون‌ها را با حروف کوچک و با آندرلاین جدا کنید: employee_id
- هر بخش اصلی کوئری را در خط جداگانه قرار دهید
- از نام‌های معنادار و یکتا استفاده کنید
- از کامنت‌ها برای توضیح منطق پیچیده استفاده کنید

مثال فرمت‌بندی خوب:

```
SELECT
    e.employee_id,
    e.first_name,
    e.last_name,
    d.department_name
FROM
    employees AS e
```

```
INNER JOIN departments AS d ON e.dept_id = d.dept_id
                                                    WHERE
                                        'e.status = 'active
                                                      ORDER BY
                                ;d.department_name, e.last_name
```
