Praxis-Gruppenprojekt SQL

Anforderungen:

- Entlang aller Teilschritte muss **Git** verwendet werden. Jedes Team-Mitglied pullt, commitet, pusht…
- Abschlusspräsentation 15-20 min pro Gruppe, **1 Person pro Gruppe** stellt alle Arbeiten vor. Powerpoint, DBeaver, PyCharm (.py-Skripte und/oder Jupyter), Git, Gitignore.
- Die Teamarbeit soll so organisiert sein, dass niemand nur Recherche oder Präsentation macht. Alle müssen mehrere Coding-Arbeiten umsetzen und Beiträge zur Data Architecture, Data Engineering und Data Analytics leisten.
- Die Aufgabenstellungen sind bindend. Eigene kreative Ideen sind natürlich willkommen, aber nur als Zusatz, nicht als Ersatz.
- Benutzt Eure Köpfe, statt ChatGPT. Ihr wollt verstehen, lernen und kreativ gestalten.

Schritt 1: Normalisieren des Excel-Tabellenblatts

Der erste Schritt des SQL-Projekts besteht darin, eine große Excel-Tabelle mit redundanten Werten zu normalisieren. Führe die folgenden Aufgaben durch:

- Öffne das Excel-Tabellenblatt.
- Identifiziere die redundanten Werte und bestimme die geeigneten Normalisierungsschritte.
- Normalisiere die Daten, indem du separate Tabellen erstellst und Beziehungen zwischen ihnen herstellst.
- Stelle die Datenintegrität sicher, indem du doppelte Werte eliminierst und Primär- und Fremdschlüssel-Beschränkungen erstellst

Schritt 2: Tabellen in Postgres importieren

Nachdem du die Excel-Tabelle normalisiert hast, musst du die erstellten Tabellen in eine Postgres-Datenbank importieren. Befolge diese Schritte:

- Greife auf das Datenbankmanagementsystem Postgres zu.
- Erstelle eine neue Datenbank (empfohlen!) oder wähle eine bestehende Datenbank für das Projekt aus (im absoluten Notfall 😉).
- Importiere die normalisierten Tabellen aus der Excel-Tabelle in die gewählte Datenbank. Außerdem müssen Primär- und sinnvolle Fremdschlüssel erstellt werden, sowie eine ERD-Grafik.
- Überprüfe den erfolgreichen Import der Tabellen, indem du die Daten abfragst.

Schritt 3: Rollen und Benutzer anlegen

Um die Zugriffsrechte für verschiedene Personen zu verwalten, müssen Rollen und Benutzer erstellt werden. Führe die folgenden Schritte durch:

- Bestimme die benötigten Rollen und Berechtigungen, wie z. B. "callcenter-mitarbeiter" und "datenanalyst".
- Erstelle die erforderlichen Rollen in der Postgres-Datenbank und weise jeder Rolle die entsprechenden Rechte zu.
- Erstelle Benutzerkonten für die Personen, die diese Rollen übernehmen werden, z. B. "mitarbeiter" und bestimmte Benutzer mit Nachnamen, denen die Rolle "mitarbeiter" zugewiesen wird.
- Verbinde die relevanten Rollen mit den entsprechenden Nutzern.

Schritt 4: Ansichten für die Datensicherheit erstellen

Um sensible Informationen wie Kreditkartendaten vor bestimmten Mitarbeitern zu schützen, müssen Ansichten erstellt werden. Gehe dazu wie folgt vor:

- Identifiziere die Tabellen mit sensiblen Daten, wie z.B. Kreditkarteninformationen.
- Erstelle Ansichten, die nur notwendige Spalten enthalten, z. B. nur die letzten drei Ziffern der Kreditkartennummer zur Überprüfung.
- Erteile Benutzern oder Rollen die entsprechenden Berechtigungen, um sicherzustellen, dass sie auf die Ansichten, nicht aber auf die zugrunde liegenden Tabellen zugreifen können.
- Teste die Ansichten, indem du sie abfragst, um zu überprüfen, ob die Daten richtig angezeigt werden.

Schritt 5: Einsatz eines Datenanalysten

Um mithilfe von Python und Pandas Erkenntnisse aus den Tabellen zu gewinnen, wird ein Datenanalyst eingestellt. Fahre mit den folgenden Aufgaben fort:

- Rufe Daten aus der Datenbank an und erstelle einen Pandas-Dataframe
- Welcher Kunde hat am meisten bezahlt? Und wie viel hat er bezahlt?
- Über welchen Zeitraum belaufen sich die Daten der Bestellungen?
- An welchem Wochentag wird am häufigsten eingekauft?

Schritt 6: Datenzugriffsprotokollierung implementieren (optional)

In einem optionalen Schritt kannst du einen Trigger erstellen, der den Zugriff des Datenanalysten und der Call-Center-Mitarbeiter auf die Tabellen mit Zeitstempel protokolliert. Befolge diese Schritte, wenn du daran interessiert bist:

- Lege die Anforderungen für die Protokollierung fest, z. B. die Erfassung des Benutzernamens, der Tabelle, auf die zugegriffen wurde, und des Zeitstempels.

- Erstelle eine Trigger-Funktion in der Postgres-Datenbank, die die Protokollierung beim Zugriff auf bestimmte Tabellen übernimmt.
- Hänge die Triggerfunktion an die benötigten Tabellen an.
- Teste den Trigger, indem du als Datenanalyst oder Call Center-Mitarbeiter auf die Tabellen zugreifst und die protokollierten Einträge überprüfst.

Schritt 7: Einschränkung des Datenzugriffs für die Python-Analyse

Um sicherzustellen, dass die Python-Analyse ausschließlich über die Rolle des Datenanalysten durchgeführt wird, unternimmst du die folgenden Schritte:

- Weise dem Benutzerkonto des Datenanalysten die Rolle "datenanalyst" mit Lesezugriff auf die benötigten Tabellen zu.
- Schränke die Datenzugriffsrechte für die Rolle "datenanalyst" ein, um einen Schreib-Zugriff auf die Tabellen zu verhindern.

Schritt 8: Zusammenfassung und Schlussfolgerung

Denke über das abgeschlossene SQL-Projekt nach und hebe die erledigten Aufgaben und den möglichen Nutzen hervor. Erörtere die erfolgreiche Normalisierung der Excel-Tabelle, den Import der Tabellen in Postgres, die Erstellung von Ansichten für die Datensicherheit, die Einrichtung von Rollen und Benutzern, die Verwendung von Python und Pandas für die Datenanalyse und die optionale Implementierung einer Protokollierung des Datenzugriffs. Betone die Vorteile eines solchen umfassenden SQL-Projekts in Bezug auf die Datenverwaltung, die Sicherheit und die Analysemöglichkeiten.

Viel Spaß wünscht DataCraft