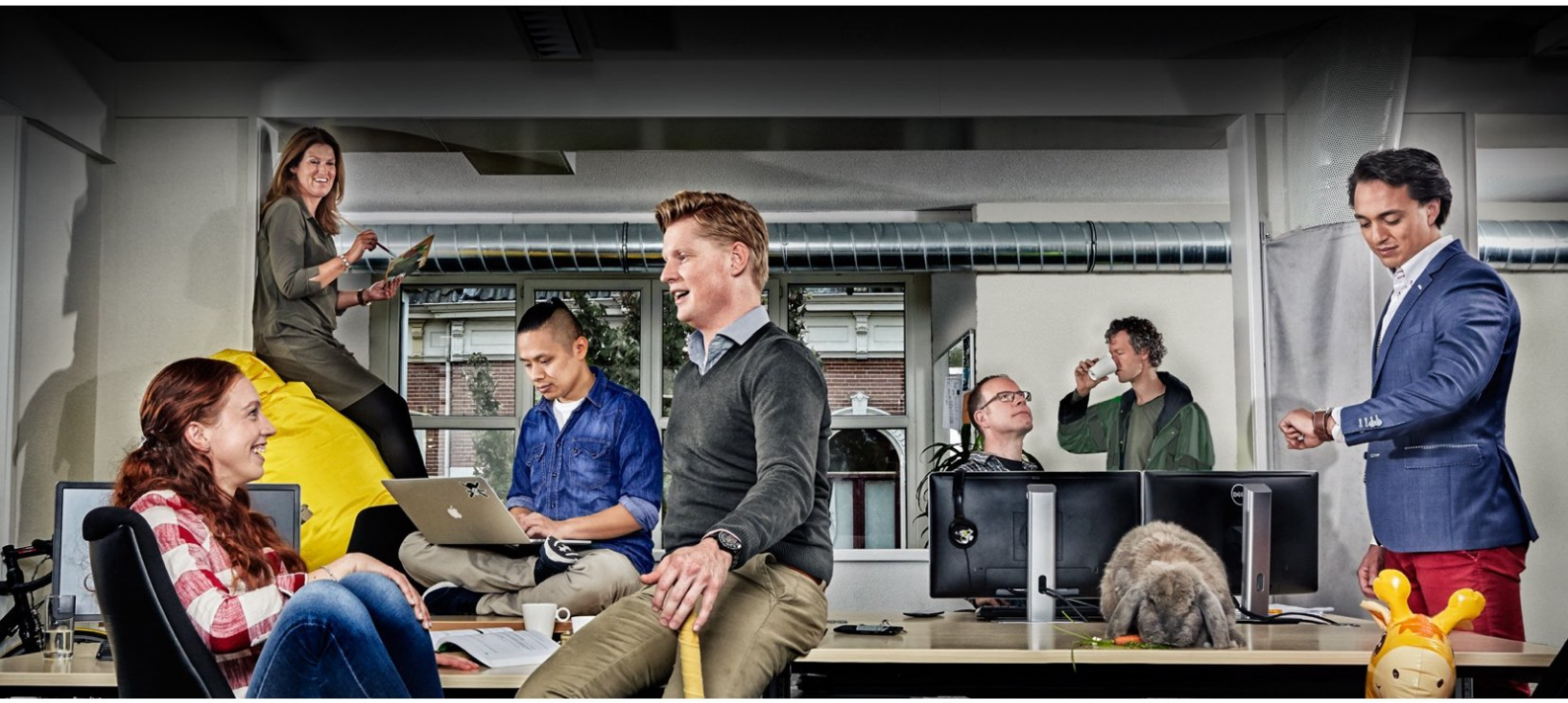
topicus
ACADEMY



TA-120

Introductie SQL

Studiegids

TA120 Introductie SQL

Topicus Academy

Inhoudsopgave

Inleiding

In het opleidingsprogramma voor software ontwikkelaars is behoefte aan een inleiding tot databases. Dit is de studiegids voor deze inleiding.

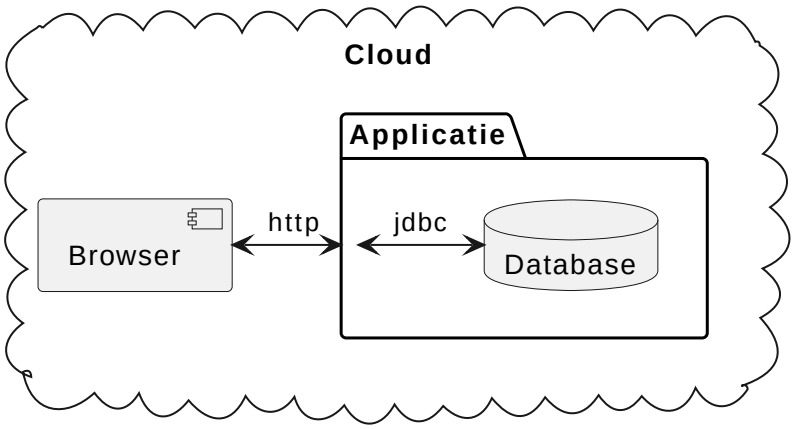
Deze course is met name bedoeld voor ontwikkelaars die in hun opleiding geen vak gevolgd hebben over databases.

De course

De course is redelijk eenvoudig en heeft als einddoel om je enigszins bekend te maken met databases en SQL.

Een database is een *persistente opslag* voor gegevens van applicaties. Zonder een database kunnen ParnasSys, SOMtoday en EduArte niet bestaan.

Omdat een database het verst van de normale gebruiker van een applicatie afstaat wordt deze meestal onderaan een *architectuur diagram* getekend als een *trommel*.



Dit laatste is historisch omdat in het verleden *trommels* (drums) gebruikt werden als schijven voor het opslaan van gegevens: het was de vorm van harde schijven (SSDs voor de jongeren onder ons).

Competenties

Je leert in deze cursus de volgende zaken:

- de database terminologie
- het schrijven van queries op een bestaande database
- het toevoegen, verwijderen of wijzigen van kolommen en tabellen

Bronnen

- [basd] Bulgarian Association of Software Developers. Introduction to SQL. Slideshare. 2011
- [h2tut] H2 database. Tutorial. h2database.com.

- [h2sql] H2 database. SQL grammar. h2database.com.

Chapter 1. Introductie tot SQL en databases

Totdat we een eigen presentatie hebben voor het behandelen van de database terminologie verwijzen we je naar Introduction to SQL door de Bulgarian Association of Software Developers.

Deze wordt gegeven in een kleine groep en behandelt de basisbegrippen van databases en de SQL query-taal. Dit duurt ongeveer 3 uur.

1.1. Vragen

- Wat is SQL?
- Wat is DML?
- Wat is DDL?
- Wat is een schema?
- Wat is het verschil tussen een SQL query en een SQL statement?
- Wat doet een index?
- Wat is een foreign key?
- Waarom geef je een constraint een naam?
- Benoem de meest belangrijke SQL commando's en leg uit wat ze doen

Chapter 2. Oefenen met databases

Om meer kennis en ervaring op te doen met databases is het handig om queries uit te voeren in een echte database. Een presentatie kan slechts zoveel informatie overbrengen.

Hiertoe dien je eerst wat database software te installeren, het databaseschema maken en de oefendata te importeren. Daarna kan je de oefeningen maken.

2.1. Software installatie

Om de opgaven te kunnen maken heb je een database server nodig. Deze zijn er in alle maten en smaken, en om het super eenvoudig en snel voor elkaar te krijgen is gekozen voor H2 database.

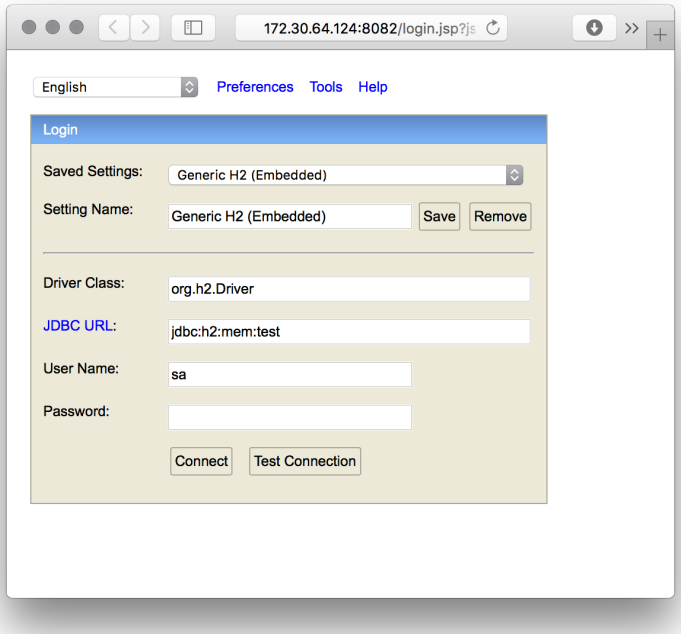
2.1.1. Installatie van de database server

Voor het oefenen maken we gebruik van een open source, gratis database die in Java geschreven is: *H2 database*. Hierdoor heb je weinig te installeren en wordt je systeem niet helemaal overhoop gegooid (want je hebt toch al Java geïnstalleerd voor je ontwikkelomgeving).

Download H2 database van de h2database.com.

Pak de Zip file uit en verplaats dit naar een plek waar het handig is om terug te vinden.

Ga naar de `h2/bin` folder en start de database door één van de scripts te draaien (h2.bat of h2.sh). Als het goed is wordt een browservenster geopend om met daarin de *H2 Console*, en dat ziet er uit als figuur 1.



Afbeelding 1. De H2 console in een webbrowser

Nu je de software geïnstalleerd hebt kan je de tabellen aanmaken en de oefendata inlezen.

2.1.2. Database inrichten

In de `intro-sql` folder op github staan 4 SQL bestanden:

1. h2-schema.sql
2. h2-indexes.sql
3. h2-data.sql
4. h2-all.sql

Je kan bestanden 1-3 achter elkaar importeren, maar dat is alleen maar vervelend, daarom is bestand 4: *h2-all.sql* gemaakt. Dit bestand bevat alles van 1-3 en kan in één keer het schema aanmaken en de tabellen vullen met data.

Het eenvoudigst is om op github het bestand h2-all.sql te openen en op de knop RAW te drukken, en dan die tekst te copy/pasten.

Je kan de server starten met een druk op de knop **[Connect]**. Dit opent een nieuw venster met daarin een textveld waarin je het script kan plakken.

H2 database

H2 gebruikt standaard een *in-memory* database. Dit betekent dat deze database weer volledig verdwenen is wanneer je de server afsluit. Je moet dan ook elke keer wanneer je connect naar de database opnieuw het schema aanmaken en inlezen. Lees de handleiding van H2 database om het schema permanent te maken.

De database heeft nu een schema met daarin de juiste tabellen en de data.

2.2. Oefeningen

De oefeningen zijn opgesplitst in een aantal secties. Eerst wat algemene vragen over de theorie en begrippen van databases. Daarna specifieke SQL queries schrijven over de Human Resource database.

Je kan [h2sql] gebruiken als referentie voor de SQL syntax. SQL syntax is gestandaardiseerd dus wat daar staat zal in het algemeen ook werken op een Postgresql, Oracle of andere database.

Wanneer je h2database gestart hebt en het schema en de data ingelezen hebt kan je de volgende oefeningen maken.

Voor je een query schrijft

Bepaal voordat je een query schrijft eerst handmatig welke resultaten je eigenlijk verwacht. Dus kijk in de tabellen naar de data en probeer eerst de gegevens aan elkaar te koppelen en dan pas de query uit te voeren. Valideer ook altijd of de opgeleverde gegevens kloppen met

wat je zou verwachten.

2.2.1. Queries schrijven

Je begint met wat eenvoudige selects.

- Schrijf een SQL query om alle informatie van alle afdelingen op te vragen
- Schrijf een SQL query om alle namen van de afdelingen op te vragen
- Schrijf een SQL query om het salaris van elke werknemer per maand, dag en uur. Ga er vanuit dat elke maand 20 werkdagen en elke werkdag 8 uur heeft.
- Creëer een lijst van de email-adressen van alle werknemers gegeven dat het maildomain "cheesr.com" is. Email-adressen moeten dan uitzien als "bernst@cheesr.com". De titel van de opgeleverde kolom moet "Emailadres" heten.
- Schrijf een SQL query om alle verschillende salarissen op te vragen.
- Schrijf een SQL query om alle informatie van werknemers op te vragen die "AC_MGR" als rol hebben (Accounting Manager)
- Schrijf een SQL query om alle namen van alle werknemers op te vragen die beginnen met "Sa".
- Schrijf een query om alle namen van alle werknemers te vinden waarvan de voornaam de letters 'ei' bevat
- Schrijf een query om de namen van alle werknemers te vinden waarvan het salaris tussen 3000 en 5000 ligt
- Schrijf een query om de namen van alle werknemers te vinden waarvan het salaris 2500, 4000 of 5000 is
- Schrijf een query om alle lokaties te vinden zonder state of postcode
- Schrijf een query om alle werknemers te vinden met een salaris hoger dan 10000. Sorteer ze met het hoogste salaris eerst.
- Schrijf een query om de top 5 best betaalde medewerkers te vinden

2.2.2. Joinen

Vaak heb je gegevens nodig die verspreid staan over meerdere tabellen. Dan gebruik je een *join* om die tabellen aan elkaar te koppelen. Maar pas op voor het gevreesde *cartesische product*.

- Schrijf een query voor alle afdelingen en de stad waarin ze gevestigd zijn met een *natural join*.
- Schrijf de vorige query maar dan met een *join* en een *using* clausule (werkt niet in H2, dus voor H2 mag je deze overslaan)
- Schrijf de vorige query maar dan met een *inner join* met de *on* clausule
- Schrijf een query om alle lokaties en afdelingen bij die lokatie, samen met de lokaties die geen afdeling hebben, op te leveren. Gebruik *right outer join*. Herschrijf deze query met een *left outer join*.
- Schrijf een query om de manager van elke afdeling te vinden

- Schrijf een query om de lokatie van elke afdelingmanager te vinden.
- Geef alle namen van alle medewerkers uit de afdelingen "Sales" en "Finance" die aangenomen zijn tussen 2000 en 2005.

2.2.3. Aggregeren

SQL is ooit gemaakt om makkelijk en eenvoudig te kunnen rapporteren. En managers vinden het maar wat leuk om met getallen te spelen. Daartoe is SQL uitgebreid met allerlei aggregatie functies om deze getallen op te leveren.

- Geef alle namen en salarissen van de medewerkers die het minimum salaris verdienen
- Geef het gemiddelde salaris in de "Sales" afdeling
- Bepaal het aantal medewerkers in de "Sales" afdeling
- Bepaal het aantal lokaties waar het bedrijf een kantoor heeft
- Bepaal het aantal afdelingen met een manager
- Bepaal het aantal afdelingen zonder een manager
- Bepaal voor elke afdeling het gemiddelde salaris
- Bepaal het aantal medewerkers per manager
- Bepaal het aantal medewerkers per afdeling
- Schrijf een SQL query dat voor alle afdelingen en alle managers het aantal medewerkers telt.
- Vind alle managers met precies 5 medewerkers. Toon hun namen en de lokatie van hun afdeling.

2.2.4. Uitvoer wijzigen

Je kan niet alleen de gegevens die opgeslagen zijn in de database opleveren, maar ook het resultaat van functies en expressies.

- Vind alle afdelingen met daarbij hun manager. De afdelingen zonder manager tonen "(geen manager)"
- Schrijf een SQL query om alle namen te vinden die exact 5 karakters lang zijn
- Schrijf een query om de huidige datum en tijd in het formaat "dag.maand.jaar uur:minuten:seconden" af te drukken. Gebruik hiervoor de dummy-tabel *DUAL*.

2.2.5. Het schema wijzigen

Een belangrijk onderdeel van ons ontwikkelproces is het migreren van het schema naar nieuwere versies van onze software. Hiertoe moeten we kolommen wijzigen, tabellen toevoegen en gegevens transformeren.

- Schrijf een SQL statement om een tabel `USERS` te maken. Users moeten een 'username', 'password', 'full name' en 'last login time' opslaan. Kies de juiste datatypes voor de kolommen. Definieer een primary key kolom met een primary key constraint. Definieer een sequence om de primaire sleutel te genereren. Definieer een trigger om de primaire sleutel te vullen vlak

voordat een record wordt geïnsert.

Binnen Topicus Onderwijs gebruiken we andere middelen om de primaire sleutel te vullen van een tabel. Het maken van een sequence en trigger is daarentegen wel goed om gezien te hebben.

- Schrijf een SQL statement dat een *view* maakt om de gebruikers te tonen die vandaag ingelogd zijn. Controleer of de view goed werkt.
- Maak een tabel voor 'GROUPS'. Groups moeten een unieke naam hebben (gebruik unique constraint). Definieer een primaire sleutel, een sequence en een trigger om deze te vullen.
- Maak een SQL statement om een kolom `GROUP_ID` toe te voegen aan de tabel `USERS`. Vul deze kolom met wat gegevens en doe dit ook in de `GROUPS` tabel. Schrijf een SQL statement om een *foreign key constraint* te leggen tussen de tabellen `USERS` en `GROUPS`.

2.2.6. Data toevoegen en verwijderen

- Schrijf wat *insert* statements om diverse records aan de `USERS` en de `GROUPS` tabellen toe te voegen.

Een les in het gebruik van de `WHERE` clausule

Er was eens een Topicus medewerker die zichzelf een hele database expert vond. We noemen hem *Mr. SQL*. Mr. SQL kent SQL. Hij is de meester van deze taal. SQL had geen geheimen voor Mr. SQL. En Mr. SQL was de primaire contactpersoon voor de huisartsen als er iets niet werkte in het landelijke huisartsensysteem.

Op een dag werd er gebeld naar Mr. SQL.

Mr. SQL: Hoe kan ik u helpen?

Huisarts A.: Ik ben mijn wachtwoord vergeten. Kunt u die voor mij opnieuw instellen?

Mr. SQL: Geen probleem. *rammelt wat op toetsenbord*

```
UPDATE users SET password = '1234'
```

Mr. SQL: Het is opgelost, uw nieuwe wachtwoord is '1234'.

Even later belt een andere huisarts naar Mr. SQL.

Mr. SQL: Waar kan ik u mee van dienst zijn?

Huisarts B.: Ik ben kennelijk mijn wachtwoord vergeten. Kunt u die voor mij opnieuw instellen?

Mr. SQL: Geen probleem. *rammelt wat op toetsenbord*

```
UPDATE users SET password = 'abcdefg'
```

Mr. SQL: Het is opgelost, uw nieuwe wachtwoord is 'abcdefg'.

Even later belt huisarts A. weer naar Mr. SQL.

Huisarts A.: Ik kan weer niet in het systeem. Hij zegt dat '1234' ongeldig is!

Vraag: Wat kan je van dit verhaal leren?