

Laboration I

Structured Query Language (SQL)

Intuitionen för IT Liran Karni

Högskolan i Borås [liran.karni@hb.se](mailto:liran.karni@hb.se)

501 90 Borås

Innehållsförteckning

[Inledning 3](#_bookmark0)

[Beskrivning av systemet 3](#_bookmark1)

[Data 7](#_bookmark2)

[Uppgifter 9](#_bookmark3)

[Uppgift 1 (DDL) 10](#_bookmark4)

[Uppgift 2 (INSERT) 10](#_bookmark5)

[Uppgift 3 (Simpla SELECT med ORDER BY) 11](#_bookmark6)

[Uppgift 4 (SELECT med GROUP BY och HAVING) 12](#_bookmark7)

[Uppgift 5 (Delfrågor och JOIN) 13](#_bookmark8)

[Uppgift 6 (UPDATE och DELETE) 14](#_bookmark9)

[Övrig information 16](#_bookmark10)

[Handledning 16](#_bookmark11)

[Redovisning 16](#_bookmark12)

[Betyg 16](#_bookmark13)

[Appendix A 17](#_bookmark14)

# Inledning

### I den här laborationen skall ni jobba med en databas för ett bilbokningssystem. Ni skall skapa bilbokningssystemets tabeller, lägga in data i respektive tabell samt skriva ett antal utsökningsfrågor. Tabellerna skapas i samma databas ni jobbade med under workshoparna.

# Beskrivning av systemet

### Bilbokningssystemet har ett antal *Fordon* som hyrs ut från olika *Uthyrningsställen* till företagets kunder. När en *Kund* bokar ett fordon, skapas en *Bokning* i databasen med tillhörande uppgifter. En bokning innehåller bland annat uppgifter om kundnumret, fordonets registreringsnummer, hyrperioden (från- och tilldatum för bokningen) samt vilket uthyrningsställe som fordonet skall hämtas ut på respektive återlämnas till. Eftersom ett fordon kan återlämnas till ett annat uthyrningsställe än uthämtningsstället, finns tabellen *Fordonsläge* till för att hålla reda på vilket uthyrningsställe ett fordon befinner sig på från ett visst datum. En bokning kan anta tre olika *Bokningsstatus* (Bokad, Uthämtad, Återlämnad). På samma sätt kan ett fordon ha ett av tre *Fordonsstatus* (Tillgänglig, Underhåll, Utgått) vid en viss tidpunkt. Ett fordon som har utgått eller som det pågår underhåll på är inte tillgänglig för uthyrning. Ett fordon har dessutom en viss *Fordonstyp* (Liten, Mellan, Stor, Minibuss, Miljöbil). Schemat för bilbokningssystemet visas i Appendix A.

### Nedan följer en beskrivning av varje tabell. Tabellerna anges både som relationer med attribut på formen **Relation** (Attribut1, Attribut2, ..., AttributN) där primärnyckeln är understruken, samt som tabeller med kolumner i grafisk form. I den grafiska framställningen innebär **PK** att en kolumn utgör primärnyckel för tabellen (PK jämte flera kolumner indikerar en sammansatt primärnyckel) samt **FK** jämte en kolumn innebär att kolumnen utgör en främmande nyckel (i schemat visas detta referensintegritetsvillkoret som en pil från tabellen med den främmande nyckeln till tabellen med kolumnen som den främmande nyckeln refererar till). Kolumnen i hemmatabellen som den främmande nyckeln refererar till anges i beskrivningen nedan. Datatyperna visas också jämte varje kolumnnamn. De tre numeriska datatyperna som används är INT (heltal), BIT (kan endast anta värdena 0 och 1) samt DECIMAL(7,2) som anger ett decimaltal med maximal längd 7 inklusive 2 decimaler, dvs. det största positiva talet som datatypen kan lagra är 99 999,99. Endast de två strängdatatyperna CHAR(*n*) och VARCHAR(*n*) används, där *n* anger antalet tecken som datatypen kan lagra, t.ex. så lagrar en CHAR(5) fem tecken och en VARCHAR(5) maximalt fem tecken. Om ett mindre antal tecken än *n* lagras i en kolumn av datatyp CHAR(*n*), så fylls kolumnen på med blanksteg tills *n* stycken tecken erhålls. En VARCHAR(*n*) fyller däremot inte på med blanksteg, dvs. den har variabel längd dock med maximalt *n* tecken. Exempelvis, så lagras texten 'abc' som 'abc ' för en CHAR(5) respektive som 'abc' för en VARCHAR(5). Slutligen används datatypen DATETIME för att lagra datum och tid. En kolumn i den grafiska framställningen som inte accepterar NULL värden (dvs. NOT NULL) anges med fetstil, annars accepterar kolumnen NULL värden.

### **Kund** (Kundnummer, Fornamn, Efternamn, Adress, Postnummer, Ort, Telefonnummer)

|  |  |  |
| --- | --- | --- |
| Kund | | |
| **PK** | **Kundnummer** | **int** |
|  | **Fornamn** | **varchar(20)** |
| **Efternamn** | **varchar(25)** |
| **Adress** | **varchar(50)** |
| **Postnummer** | **char(5)** |
| **Ort** | **varchar(20)** |
| **Telefonnummer** | **varchar(10)** |

### Tabellen *Kund* innehåller uppgifter om företagets kunder, där *Kundnummer* utgör primärnyckeln för tabellen. Samtliga kolumner accepterar inte NULL.

### **Uthyrningsstalle** (Uthyrningsstallenummer, Namn, Adress, Postnummer, Ort, Telefonnummer, Email, Webbsida)

|  |  |  |
| --- | --- | --- |
| Uthyrningsstalle | | |
| **PK** | **Uthyrningsstallenummer** | **int** |
|  | **Namn** | **varchar(30)** |
| **Adress** | **varchar(50)** |
| **Postnummer** | **char(5)** |
| **Ort** | **varchar(20)** |
| **Telefonnummer** | **varchar(10)** |
| **Email** | **varchar(40)** |
| Webbsida | varchar(50) |

Tabellen *Uthyrningsstalle* innehåller uppgifter om företagets uthyrningsställen, där *Uthyrningsstallenummer* utgör primärnyckeln för tabellen. Samtliga kolumner utom Webbsida accepterar inte NULL.

**Fordonstyp** (Fordonstypkod, Typ)

### Tabellen *Fordonstyp* innehåller information om ett fordons typ (Liten, Mellan, Stor, Minibuss, Miljöbil), där *Fordonstypkod* utgör primärnyckeln för tabellen. Samtliga kolumner accepterar inte NULL.

|  |  |  |
| --- | --- | --- |
| Fordonstyp | | |
| **PK** | **Fordonstypkod** | **int** |
|  | **Typ** | **varchar(20)** |

**Fordonsstatus** (Fordonsstatuskod, Status)

### Tabellen *Fordonsstatus* innehåller information om ett fordons status (Tillgänglig, Underhåll, Utgått), där *Fordonsstatuskod* utgör primärnyckeln för tabellen. Samtliga kolumner accepterar inte NULL.

|  |  |  |
| --- | --- | --- |
| Fordonsstatus | | |
| **PK** | **Fordonsstatuskod** | **int** |
|  | **Status** | **varchar(15)** |

### **Fordon** (Registreringsnummer, Fordonstypkod, Modell, Miltal, Dagshyra, AntalPassagerare, Automat, Fordonsstatuskod)

Tabellen *Fordon* innehåller information om företagets fordon, där fordonets *Registreringsnummer* utgör primärnyckeln för tabellen. *Fordonstypkod* är en främmande nyckel som refererar till *Fordonstypkod* i tabellen *Fordonstyp*. *Fordonsstatuskod* är en främmande nyckel som refererar till *Fordonsstatuskod* i tabellen *Fordonsstatus*. Samtliga kolumner accepterar inte NULL.

|  |  |  |
| --- | --- | --- |
| Fordon | | |
| **PK** | **Registreringsnummer** | **char(6)** |
| **FK1**  **FK2** | **Fordonstypkod Modell**  **Miltal Dagshyra**  **AntalPassagerare Automat Fordonsstatuskod** | **int varchar(50) int decimal(7,2) int**  **bit int** |

**Bokningsstatus** (Bokningsstatuskod, Status)

### Tabellen *Bokningsstatus* innehåller information om en boknings status (Bokad, Uthämtad, Återlämnad), där *Bokningsstatuskod* utgör primärnyckeln för tabellen. Samtliga kolumner accepterar inte NULL.

|  |  |  |
| --- | --- | --- |
| Bokningsstatus | | |
| **PK** | **Bokningsstatuskod** | **int** |
|  | **Status** | **varchar(15)** |

### **Bokning** (Bokningsnummer, Kundnummer, Registreringsnummer, Bokningsstatuskod, FranDatum, TillDatum, Uthamntningsstallenummer, Aterlamningsstallenummer)

|  |  |  |
| --- | --- | --- |
| Bokning | | |
| **PK** | **Bokningsnummer** | **int** |
| **FK1 FK2 FK3**  **FK4 FK5** | **Kundnummer Registreringsnummer Bokningsstatuskod FranDatum**  **TillDatum Uthamtningsstallenummer Aterlamningsstallenummer** | **int char(6) int datetime datetime int**  **int** |

Tabellen *Bokning* innehåller information om kundernas bokningar av företagets fordon, där *Bokningsnummer* utgör primärnyckeln för tabellen. *Kundnummer* är en främmande nyckel som refererar till *Kundnummer* i tabellen *Kund*. *Registreringsnummer* är en främmande nyckel som refererar till *Registreringsnummer* i tabellen *Fordon*.

*Bokningsstatuskod* är en främmande nyckel som refererar till *Bokningsstatuskod* i tabellen *Bokningsstatus*. *Uthamtningsstallenummer* är en främmande nyckel som refererar till *Uthyrningsstallenummer* i tabellen *Uthyrningsstalle*. *Aterlamningsstallenummer* är en främmande nyckel som refererar till *Uthyrningsstallenummer* i tabellen *Uthyrningsstalle*. Samtliga kolumner accepterar inte NULL. När en kund bokar ett fordon lagras en ny rad i tabellen, där hyresperioden lagras i *FranDatum* och *TillDatum* (både datum och tid). Om kunden vill lämna tillbaka fordonet på ett uthyrningsställe annat än uthämtningsstället, så skiljer sig *Aterlamningsstallenummer från Uthamtningsstallenummer*.

### **Fordonslage** (Registreringsnummer, FranDatum, Uthyrningsstallenummer)

### Tabellen *Fordonslage* innehåller information om vilket uthyrningsställe ett fordon befinner sig på från ett visst datum, där *Registreringsnummer* tillsammans med *FranDatum* utgör den sammansatta primärnyckeln för tabellen.

|  |  |  |
| --- | --- | --- |
| Fordonslage | | |
| **PK,FK1 PK** | **Registreringsnummer FranDatum** | **char(6) datetime** |
| **FK2** | **Uthyrningsstallenummer** | **int** |

*Registreringsnummer* är en främmande nyckel som refererar till *Registreringsnummer* i tabellen *Fordon*. *Uthyrningsstallenummer* är en främmande nyckel som refererar till *Uthyrningsstallenummer* i tabellen *Uthyrningsstalle*. Samtliga kolumner accepterar inte NULL. När en kund bokar ett fordon där återlämningsstället skiljer sig från uthämtningsstället, så lagras en ny rad i tabellen. *FranDatum* i tabellen *Fordonslage* får samma värde som bokningens *TillDatum* i tabellen *Bokning* samt *Uthyrningsstallenummer* i tabellen *Fordonslage* får samma värde som bokningens *Aterlamningsstallenummer* i tabellen *Bokning*.

# Data

### Instansen av databasen ni skall jobba med visas nedan.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Kund** |  |  |  |  |  |  |
| *Kundnummer* | *Fornamn* | *Efternamn* | *Adress* | *Postnummer* | *Ort* | *Telefonnummer* |
| 1 | Maria | Svensson | Solbacken 5 | 50636 | Borås | 033123456 |
| 2 | Anna | Olsson | Kyrkogatan 11 | 50912 | Borås | 033112233 |
| 3 | Anders | Andersson | Sandvägen 7 | 41137 | Göteborg | 031332211 |
| 4 | Ulf | Toresson | Sjögatan 3 | 50812 | Borås | 033542312 |
| 5 | Sten | Larsson | Fågelvägen 15 | 40625 | Göteborg | 033247278 |
| 6 | Bror | Svantesson | Kungsgatan 1 | 50711 | Borås | 033772058 |
| 7 | Lena | Johansson | Karlavägen 23 | 10342 | Stockholm | 086305112 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Uthyrningsstalle** | | | | | | | |
| *Uthyrnings*  *stallenummer* | *Namn* | *Adress* | *Post*  *nummer* | *Ort* | *Telefon*  *nummer* | *Email* | *Webbsida* |
| 1 | BU Borås | Grågatan 5 | 50700 | Borås | 033500500 | [info@bu.se](mailto:info@bu.se) | [www.bu.se/bs](http://www.bu.se/bs) |
| 2 | BU Göteborg | Solgatan 2 | 40100 | Göteborg | 031303031 | [info@bu.se](mailto:info@bu.se) | [www.bu.se/gbg](http://www.bu.se/gbg) |
| 3 | BU Stockholm | Sengatan 6 | 10425 | Stockholm | 080101010 | [info@bu.se](mailto:info@bu.se) | *NULL* |

|  |  |
| --- | --- |
| **Fordonstyp** |  |
| *Fordonstypkod* | *Typ* |
| 1 | Liten |
| 2 | Mellan |
| 3 | Stor |
| 4 | Minibuss |
| 5 | Miljöbil |

|  |  |
| --- | --- |
| **Fordonsstatus** |  |
| *Fordonsstatuskod* | *Status* |
| 1 | Tillgänglig |
| 2 | Underhåll |
| 3 | Utgått |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Fordon** |  |  |  |  |  |  |  |
| *Registrerings*  *nummer* | *Fordons*  *typkod* | *Modell* | *Miltal* | *Dagshyra* | *Antal*  *Passagerare* | *Automat* | *Fordons*  *statuskod* |
| ABC123 | 1 | Toyota Aygo 1.0 | 1000 | 1000.00 | 4 | 0 | 1 |
| ADN274 | 2 | Volkswagen Passat 2.0 | 1000 | 1000.00 | 5 | 0 | 2 |
| AVE693 | 2 | Polestar 2 | 300 | 3000.00 | 4 | 1 | 1 |
| BUS104 | 4 | Volkswagen Sharan | 3000 | 1000.00 | 7 | 0 | 2 |
| GPS935 | 3 | Hyundai Sonata | 1500 | 1750.00 | 5 | 1 | 1 |
| HAP555 | 3 | Volvo V60 Plug-In | 750 | 1750.00 | 5 | 1 | 1 |
| KAD395 | 4 | Volkswagen Transporter | 2500 | 1750.00 | 9 | 1 | 1 |
| LJF599 | 1 | Volkswagen Golf 1.6 | 3700 | 1350.00 | 5 | 1 | 3 |
| PPP409 | 5 | Tesla Modell S | 500 | 2500.00 | 4 | 1 | 1 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Bokning** |  |  |  |  |  |  |  |
| *Boknings*  *nummer* | *Kund*  *nummer* | *Registrerings*  *nummer* | *Boknings*  *statuskod* | *FranDatum* | *TillDatum* | *Uthamtnings*  *stallenummer* | *Aterlamnings*  *stallenummer* |
| 1 | 1 | ADN274 | 3 | 2021-03-25 09:00 | 2021-04-05 09:00 | 1 | 1 |
| 2 | 2 | KAD395 | 3 | 2021-05-01 18:00 | 2020-05-15 18:00 | 1 | 2 |
| 3 | 3 | PPP409 | 3 | 2021-05-03 10:00 | 2020-05-05 18:00 | 2 | 2 |
| 4 | 5 | PPP409 | 3 | 2021-05-11 18:00 | 2020-05-13 18:00 | 2 | 2 |
| 5 | 4 | GPS935 | 3 | 2021-05-07 08:00 | 2020-05-09 19:00 | 1 | 1 |
| 6 | 1 | AVE693 | 3 | 2021-05-15 06:00 | 2020-05-15 19:00 | 3 | 3 |
| 7 | 5 | HAP555 | 1 | 2021-05-20 09:00 | 2020-05-27 09:00 | 2 | 1 |
| 8 | 3 | BUS104 | 1 | 2021-06-03 09:00 | 2020-06-17 09:00 | 2 | 2 |
| 9 | 1 | ADN274 | 1 | 2021-06-13 09:00 | 2020-06-15 15:00 | 1 | 1 |
| 10 | 6 | AVE693 | 1 | 2021-06-14 09:00 | 2020-06-14 17:00 | 2 | 2 |
|  |  |  |  |  |  |  |  |

|  |  |  |
| --- | --- | --- |
| **Fordonslage** |  |  |
| *Registreringsnummer* | *FranDatum* | *Uthyrningsstallenummer* |
| ABC123 | 2021-01-01 00:00 | 2 |
| ADN274 | 2021-01-01 00:00 | 1 |
| AVE693 | 2021-05-11 00:00 | 3 |
| BUS104 | 2021-01-01 00:00 | 2 |
| GPS935 | 2021-01-01 00:00 | 1 |
| HAP555 | 2021-01-01 00:00 | 2 |
| KAD395 | 2021-01-01 00:00 | 1 |
| LJF599 | 2021-01-01 00:00 | 1 |
| PPP409 | 2021-05-01 00:00 | 2 |
| KAD395 | 2021-04-15 18:00 | 2 |
| HAP555 | 2021-07-12 09:00 | 1 |

|  |  |
| --- | --- |
| **Bokningsstatus** |  |
| *Bokningsstatuskod* | *Status* |
| 1 | Bokad |
| 2 | Uthämtad |
| 3 | Återlämnad |

# Uppgifter

### Nedan följer sex uppgifter, som i sin tur är indelade i ett antal deluppgifter. Skapa ett skript för respektive uppgift, dvs. filerna skall ha namnen LAB1.sql, LAB2.sql, LAB3.sql, LAB4.sql, LAB5.sql samt LAB6.sql.

### Kom ihåg att ta kopior av, samt spara, filerna med jämna mellanrum!

### Naturligtvis skall ni själva testa att SQL-filerna fungerar innan ni lämnar in dem, men tänk också på att en korrekt utsökning är generell, dvs. fungerar i alla möjliga instanser av databasen.

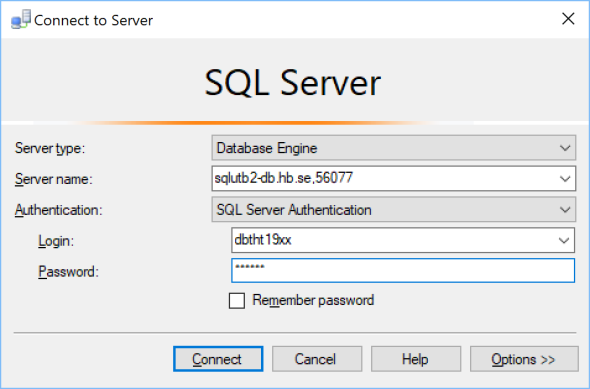
### Om en lösning innehåller en delfråga, men hade kunnat lösas utan, räknas detta som en felaktig lösning även om den ger korrekt resultat.

### Se till så att gruppnummer och samtliga gruppmedlemmars namn finns inskrivna som SQL kommentarer i början av varje skript-fil. En kommentar i SQL-språket skapas genom att ange två bindesträck i början av en rad (då blir hela raden en kommentar) eller genom att innesluta kommentaren mellan /\* och \*/ t.ex:

-- Grupp xx: John Doe, Herr Niemand

/\* Grupp xx: John Doe, Herr Niemand \*/

Använd Editorn i *Microsoft SQL Server Management Studio* för att skapa och spara respektive skript. Detta görs genom att:

1. Starta *MS SQL Server Management Studio* och koppla upp er mot Microsoft *SQL Server* instansen "sqlutb2-db.hb.se,56077" med inloggningsuppgifterna för er grupps databaskonto.

Ni hittar inloggningsuppgifterna i Canvas (i er projektgrupps dokumentmapp).

#### OBS!

I inloggningsfönstret skall "Server type"

vara *Database Engine* samt "Authentication" skall vara *SQL Server Authentication*.

1. Expandera "Databases" noden i *Object Explorer* och leta reda på gruppens databas (databasens namn är samma som gruppens användarnamn t.ex. *dbtht23xx* ).
2. Högerklicka på gruppens databas (t.ex. *dbtht23xx* i *Object Explorer*) och välj *New Query* från menyn.
3. Detta öppnar en ny flik i Editorn där SQL koden för skriptet kan skrivas in.
4. För att spara skriptet, klicka på save.png knappen i verktygsfältet under huvudmenyn.

## Uppgift 1 (DDL)

### Nedanstående SQL-kod skall sparas i skriptfilen LAB1.SQL.

1. Skriv SQL-kod för att skapa tabellerna enligt schemat i Appendix A och beskrivningen i kapitel

2. Tänk på att ordningen i vilket tabellerna skapas spelar roll pga. de främmade nycklarna. TIPS: Om ni högerklickar på ’Database Diagrams’ i ert databas-utrymme kan ni skapa ett nytt databas-diagram . Där kan ni sedan jämföra ert diagram med det som givits i Appendix A och se om ni har fått till alla kopplingar rätt.

1. Skriv SQL-kod (en ALTER TABLE sats) som lägger till ett villkor vilket ser till att TillDatum är senare än FranDatum i tabellen Bokning.
2. Skriv SQL-kod som skapar vyn [*Tillgängliga Fordon*] som listar samtliga tillgängliga fordon (dvs. visa inte fordon som har utgått eller som det pågår underhåll på). Visa Registreringsnummer, Modell, AntalPassagerare samt Automat.

OBS! När en vy skapas, kommer inte ett resultat att visas i meddelande-fönstret. För att se resultatet behöver man gå in under ’views’ och öppna den. Om ni vill kan ni lösa den här deluppgiften efter ni har lagt till data i tabellerna i uppgift 2, annars kommer det inte finnas något att se i vyn.

### Skriptfilen kommer alltså att innehålla ett antal CREATE TABLE satser, en ALTER TABLE sats samt en CREATE VIEW sats. Om ni får ett felmeddelande om att t.ex. en CREATE VIEW måste komma först i en SQL-batch (dvs. databashanteraren tolkar all SQL kod i er skriptfil som en enda "batch" av SQL kommandon), så kan ni skriva det reserverade ordet GO på en ny rad efter varje SQL sats i skriptfilen. Då körs varje SQL sats som en egen "batch", t.ex:

CREATE TABLE T (

column1 INT NULL

) GO

ALTER TABLE T

ALTER COLUMN column1 INT NOT NULL GO

CREATE VIEW V (column1) AS

SELECT 'Hello World'

GO

### Om ni kör er skriptfil flera gånger, tänk på att ni först måste ta bort samtliga objekt ni har skapat (t.ex. tabeller) *innan* ni kan skapa dem igen.

## Uppgift 2 (INSERT)

### Nedanstående SQL-kod skall sparas i skriptfilen LAB2.SQL.

### Skapa databasinstansen som visas i kapitel 3, dvs. skriv SQL-kod som lägger till samtliga rader i respektive tabell. Tänk på att ordningen i vilket Datat läggs till i de olika tabellerna spelar roll på grund av de främmande nycklarna.

### (Ni kan ta databas-översikten i Appendix A som hjälp ifall det känns lite snårigt att komma ihåg.)

### TIPS: Skriptfilen kommer att innehålla ett antal INSERT INTO satser.

## Uppgift 3 (Simpla SELECT med ORDER BY)

Skapa skriptfilen LAB3SQL som innehåller SQL-uttryck som motsvarar utsökningsfrågorna nedan. Skriv en kommentar innan varje sats i skriptfilen som anger vilken deluppgift (a-e) den besvarar.

Tänk på att en korrekt utsökning är generell, dvs. fungerar i alla möjliga instanser av databasen, oavsett vilka data som finns i den.

1. Skriv en SQL-sats som ger en förteckning över alla bilar som inte har utgått eller är på underhåll. Resultatet skall innehålla *Registreringsnummer*, *Modell* samt *Miltal*. Sortera listan på *Miltal* i fallande ordning.



1. Skriv en SQL-sats vilken ger en förteckning över alla bokningar med registreringsnummer PPP409. Samtliga kolumner i tabellen *Bokning* skall visas i resultatet och listan skall vara sorterad på *Kundnummer* i fallande ordning.

1. Skriv en SQL-sats som ger en förteckning över samtliga bokningar vars FranDatum ligger från den 1:e till den 30 maj år 2021. Visa *Bokningsnummer*, *Registreringsnummer*, *Bokningssstatuskod*, *FranDatum* och *TillDatum* i resultatet. Sortera på *TillDatum* i fallande ordning.



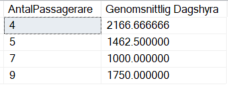
1. I den tidigare uppgiften upptäckte vi ett fel: en bil står som bokad, fastän den i själva verket är återlämnad. Skriv en SQL-sats som uppdaterar tabellen Bokning. Ändra den felaktiga posten så att bokningsstatuskoden blir 3, dvs återlämnad. Skriv en SQL-sats som kontrollerar att det har blivit rätt.
2. Skriv en SQL-sats som ger en förteckning över samtliga uthyrningsställen som har en webbsida (d.v.s. där *Webbsida* inteär lika med NULL). Visa *Namn*, *Ort* samt Email i resultatet. Sortera på *Ort* i stigande ordning.

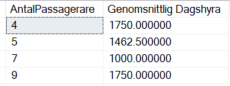


## Uppgift 4 (SELECT med GROUP BY och HAVING)

Skapa skriptfilen LAB4.SQL som innehåller SQL-uttryck som motsvarar utsökningsfrågorna nedan. Skriv en kommentar innan varje sats i skriptfilen som anger vilken deluppgift (a-e) den besvarar.

Tänk på att en korrekt utsökning är generell, dvs. fungerar i alla möjliga instanser av databasen, oavsett vilka data som finns i den.

1. Skriv en SQL-sats som listar antalet kunder per *Ort* som bor i Stockholm eller Göteborg. Visa *Ort* samt antalet kunder per ort som [*Antal Kunder*]. Listan skall vara sorterad stigande på *Ort*.
2. Skriv en SQL-sats som redovisar den genomsnittliga dagshyran per antal passagerare. Listan skall visa *Fordonstypkod* samt den genomsnittliga dagshyran som [*Genomsnittlig Dagshyra*]. Listan skall vara sorterad på antal passagerare i stigande ordning.
3. Vi inser att på grund av de dyrare bilarna blir den genomsnittliga dagshyran lite missvisande. Utgå från föregående SQL-sats och skriv en ny SQL-sats där samtliga bilar med en dagshyra på 3000 kr och över är borttagna.



1. Skriv en SQL-sats som visar hur många fordon som är bokade fler än en gång samt hämtade och återlämnade på olika uthyrningsställen. Visa *Registreringsnummer,* antal bokningar som [*Antal Bokningar*] samt *Uthamtningsstallenummer*.

Listan skall vara sorterad på *Uthamtningsstallenummer* i stigande ordning.

## Uppgift 5 (Delfrågor och JOIN)

Skapa skriptfilen LAB5.SQL som innehåller SQL-uttryck som motsvarar utsökningsfrågorna nedan. Skriv en kommentar innan varje sats i skriptfilen som anger vilken deluppgift (a-d) den besvarar.

Tänk på att en korrekt utsökning är generell, dvs. fungerar i alla möjliga instanser av databasen, oavsett vilka data som finns i den.

1. Skriv en SQL-sats som ger en förteckning över vilka fordon som kunden med förnamnet ’Maria’ och efternamnet ’Svensson’ har hyrt. Visa *Bokningsnummer*, *Registreringsnummer*, *Modell*, *FranDatum*, *TillDatum,* samt antalet hyrda dygn som [*Antal hyrda dygn*] i resultatet. Listan sorteras stigande på *Bokningsnummer*.

TIPS: Med SQL-funktionen DATEDIFF kan man jämföra två datum med varandra.

1. Skriv en SQL-sats som visar samtliga fordon som inte har några bokningar från kunder i Göteborg (OBS! frågan skall även visa de fordon för vilka det inte finns några bokningar i tabellen *Bokning*). Visa *Registreringsnummer* och *Modell* samt *Miltal* i resultatet. Listan sorteras i stigande ordning på *Miltal*.



1. Skriv en SQL-sats som visar de kunder som har lämnat tillbaka ett fordon på en annan *Ort* där de bor samt hyrt fordonet i mer än 3 dygn. Ange *Kundnummer, ForNamn,* EfterNamn, uthyrningsställets *Namn*, *Ort* samt antalet hyrda dygn som [*Antalet hyrda dygn*].



1. Lista samtliga tillgängliga fordon som är lediga mellan '2021-05-07 09:00' och '2013-07-15 17:00'. Visa *Registreringsnummer*, *Typ*, *Modell*, *Dagshyra*, *AntalPassagerare*, *Automat* samt uthyrningsställets *Ort* i resultatet. Sortera på *AntalPassagerare* i stigande ordning och därefter på *Dagshyra* i stigande ordning.

Den här deluppgiften är ganska omfattande, därför ni får hjälp.

* Tänk på att ni först måste veta vid vilket uthyrningsställe respektive fordon finns vid datumet ’2021-05-07 09:00'.

För detta använder ni er av en delfråga i den yttre frågans FROM klausul för att returnera *Registreringsnummer* samt det senaste *FranDatum* för respektive fordon i tabellen *Fordonslage* där *FranDatum* är tidigare eller lika med 2021-05-07 09:00' (ni behöver en GROUP BY i delfrågan eftersom ni enbart är intresserade av unika registreringsnummer). Ni behöver också skapa ett alias för delfrågan. Alias:et anges direkt efter delfrågan enligt syntaxen "FROM (delfråga) bilar", där bilarär ett alias för delfrågan.

* Därefter måste ni JOIN:a delfrågan med tabellen *Fordonslage* på *Registreringsnummer* **och** *FranDatum*. Först då erhålls vid vilket uthyrningsställe respektive fordon finns vid datumet '2021-05-07 09:00'.

Därefter behöver ni JOIN:a ytterligare ett antal tabeller för att få tillgång till all den information som skall presenteras i resultatet.

* Ni behöver också se till att respektive fordon är ledigt, dvs inte bokad, mellan ’2021-05-07 09:00’ och '2020-05-15 17:00'. För att göra detta behöver ni använda en delfråga i den yttre frågans WHERE klausul.
* Glöm inte bort att respektive fordon också måste vara tillgängligt (dvs. ta inte med fordon som har utgått eller som det pågår underhåll på).

1. Kunden blev inte förtjust i alternativet på bil och undrar om det blir fler bilar tillgängliga några andra dagar.

Skriv en SQL-sats där du lägger till två variabler (enligt instruktionerna på följande [sida](https://www.sqlshack.com/sql-variables-basics-and-usage/)) som du kallar *@FranDatum* och *@TillDatum* och ersätter alla datumangivelser i förra SQL-satsen.

* + Du ser genom testande att det finns en annan bil mellan den 6 maj 2021 09:00 till den 10 maj 2021 19:00. Kunden blir nöjd.
  + Ange datumen i dina variabler *@FranDatum* och *@TillDatum* och se vilken bil som kunden slutligen blev nöjd med.

## Uppgift 6 (UPDATE och DELETE)

Skapa skriptfilen LAB6.SQL som innehåller SQL-uttryck för att genomföra följande ändringar i databasen. Skriv en kommentar innan varje sats i skriptfilen som anger vilken deluppgift (a-c) den besvarar.

I samtliga av följande uppgifter gör ni sedan en SQL-sats som ger er möjlighet att kontrollera att ni gjort rätt.

1. Skriv en SQL-sats som ändrar adressen på kunden med kundnummer 5 till ’Kviststigen 23’ med postnummer ’332 31’ i orten ’Gislaved’.
2. Skriv en SQL-sats som adderar 100 mil till *Miltal* för bilarna med registreringsnummer ADN274 och ABC123.
3. Skriv en SQL-sats som lägger till hemsideadressen ’www.bu.se/sthlm’ för uthyrningsställe nummer 3.
4. Skriv en SQL-sats som ändrar samtliga bokade bilar i tabellen Bokning till uthämtade.

### **OBS**! Tänk på att ni kan återställa innehållet i era tabeller (så som de såg ut efter inmatningen i uppgift 2) genom att först tömma tabellerna med DELETE FROM satsen och sedan köra skriptfilen från uppgift 2 igen. I så fall behöver ni också tömma tabellerna, samt lägga till data i tabellerna i en viss ordning pga. de främmande nycklarna.

# Övrig information

## Handledning

Handledning av laborationen sker vid två tillfällen. Båda tillfällen får, men behöver inte, utnyttjas. Båda handledningstillfällen kommer ske över Zoom. Handledningstillfällena sker gruppvist, där varje grupp kan boka en handledningstid i Canvas-kalendern (tider läggs upp senast 1 vecka innan*).*

Med reservation för ändringar (kolla alltid schemat), så gäller för närvarande nedanstående datum för varje handledningstillfälle i Zoom:

* Handledning 1: 18 september 08:00 – 17:00.
* Handledning 2: 10 oktober 08:00 – 17:00.

## Redovisning

Laborationen redovisas via Canvas, under *Uppgifter/Laboration 1*, där samtliga skriptfiler lämnas in som **en** arkivfil (zip-fil eller rar-fil). Glöm inte att ange gruppnummer samt namnen på samtliga gruppmedlemmar som en kommentar i samtliga skriptfiler!

Inlämning och examination av laborationen sker vid tre tillfällen enligt schemat (OBS! En länk för ett visst inlämningstillfälle blir först synlig i Canvas dagen efter det förra inlämningstillfället). Samtliga tre tillfällen får utnyttjas. Om ni blir underkända på någon uppgift efter första och andra examinationstillfället skall denna vara åtgärdad innan nästa examinationstillfälle. Nedanstående datum gäller för varje examinationstillfälle:

* Deadline 1: 24 september 23:59.
* Deadline 2: 15 oktober 23:59.
* Deadline 3: 5 november 23:59.

## Betyg

Endast betygen Underkänd eller Godkänd ges på det här kursmomentet.

Disciplinärenden

Enligt [10 kap 1§ högskoleförordningen](https://www.hb.se/globalassets/global/hogskoleforordningen_kap-10.pdf) får disciplinära åtgärder vidtas mot student som:

1. med otillåtna hjälpmedel eller på annat sätt försöker vilseleda vid prov eller när en studieprestation annars skall bedömas,
2. stör eller hindrar undervisning, prov eller annan verksamhet inom ramen för utbildningen vid högskolan,
3. stör verksamheten vid högskolans bibliotek eller annan särskild inrättning inom högskolan, eller
4. utsätter en annan student eller en arbetstagare vid högskolan för trakasserier. Skyldigheten att anmäla ett disciplinärende gäller alla anställda vid högskolan. Mer kring dessa regler kan läsas [här](https://www.hb.se/student/mina-studier/tentamen-examination/nar-reglerna-inte-foljs/disciplinarenden-och-forseelser/).

# Appendix A

