IBM ® DB2® Universal Database

# Begynnerhåndbok for SQL

*Versjon 7*

SA15-4773-00

IBM ® DB2® Universal Database

# Begynnerhåndbok for SQL

*Versjon 7*

SA15-4773-00

Før du bruker opplysningene i denne boken og det produktet det blir henvist til, må du lese “Tillegg C. Merknader” på side 111.

# Innhold

**Velkommen . . . . . v**
Beslektet dokumentasjon for denne boken . . v
Konvensjoner. . . . . vi

**Kapittel 1. Relasjonsdatabaser og SQL . . 1**

**Kapittel 2. Organisere data . . . . . 3**
Tabeller . . . . . 3
Utsnitt . . . . . 4
Skjemaer . . . . . 4
Datatyper . . . . . 5

**Kapittel 3. Opprette tabeller og utsnitt . . . 9**
Opprette tabeller . . . . . 9
Legge inn data . . . . . 10
Endre data . . . . . 12
Slette data. . . . . 13
Opprette utsnitt . . . . . 13
Bruke utsnitt til å manipulere data . . . 15

**Kapittel 4. Bruke SQL-setninger for å få tilgang til data . . . . . 17**
Tilkoble til en database . . . . . 18
Undersøke problemer . . . . . 18
Velge kolonner . . . . . 19
Velge rader . . . . . 20
Sortere rader . . . . . 22
Fjerne like rader . . . . . 24
Rekkefølge på operasjoner . . . . . 24
Bruke uttrykk til å beregne verdier . . . . 25
Navngi uttrykk . . . . . 25
Velge data fra mer enn en tabell . . . . . 26
Bruke en delspørring . . . . . 27
Bruke funksjoner . . . . . 28
Kolonnefunksjoner . . . . . 28
Skalarfunksjoner . . . . . 29
Tabellfunksjoner . . . . . 30
Gruppering . . . . . 30
Bruke et WHERE-ledd med et GROUP BY-ledd . . . . . 31
Bruke HAVING-leddet etter GROUP BY-leddet . . . . . 32

**Kapittel 5. Uttrykk og delspørringer . . . 33**
Skalarfunksjon med full SELECT-setninger . . 33
Konvertere datatyper . . . . . 33
CASE-uttrykk . . . . . 34
Tabelluttrykk. . . . . 35
Nestede tabelluttrykk . . . . . 36
Felles tabelluttrykk. . . . . 36
Korrelasjonsnavn . . . . . 38
Korrelasjonsdelspørringer . . . . . 39
Implementere en korrelasjonsdelspørring 41

**Kapittel 6. Bruke operatorer og predikater i spørringer . . . . . 45**
Kombinere spørringer med mengdeoperatorer 45
UNION-operator . . . . . 45
EXCEPT-operator . . . . . 46
INTERSECT-operator . . . . . 47
Predikater. . . . . 48
Bruke IN-predikatet . . . . . 48
Bruke BETWEEN-predikatet . . . . . 48
Bruke LIKE-predikatet . . . . . 49
Bruke EXISTS-predikatet . . . . . 49
Kvantifiserte predikater . . . . . 50

**Kapittel 7. Avansert SQL . . . . . 51**
Håndheve forretningsregler med begrensninger og utløsere . . . . . 51
Nøkler . . . . . 52
Entydige begrensninger . . . . . 52
Referanseintegritetsbegrensninger . . . . 52
Tabellkontrollbegrensninger . . . . . 53
Utløsere . . . . . 54
Kombineringer . . . . . 58
Sammensatte spørringer . . . . . 62
ROLLUP- og CUBE-spørringer . . . . . 63
Rekursive spørringer . . . . . 63
OLAP-funksjoner . . . . . 63

**Kapittel 8. Tilpasse og forbedre datamanipulering . . . . . 65**
Brukerdefinerte typer . . . . . 65
Brukerdefinerte funksjoner . . . . . 66
Store objekter (LOB) . . . . . 67
Manipulere store objekter (LOB) . . . . 67
Spesialregistre . . . . . 68
Innledning til katalogoversikter . . . . . 69

Velge rader fra systemkataloger . . . . 69

**Tillegg A. Eksempeldatabasetabeller . . . 71**
Eksempeldatabasen . . . . . 72
Opprette eksempeldatabasen . . . . . 72
Slette eksempeldatabasen . . . . . 72
CL_SCHED Table . . . . . 72
DEPARTMENT Table . . . . . 73
EMPLOYEE Table . . . . . 73
EMP_ACT Table . . . . . 76
EMP_PHOTO Table . . . . . 78
EMP_RESUME Table . . . . . 78
IN_TRAY Table . . . . . 79
ORG Table . . . . . 79
PROJECT Table . . . . . 80
SALES Table . . . . . 81
STAFF Table . . . . . 82
STAFFG Table . . . . . 83
Eksempelfiler med BLOB- og CLOB-datatyper 84
Quintana Photo . . . . . 84
Quintana Resume . . . . . 84
Nicholls Photo . . . . . 85
Nicholls Resume . . . . . 86
Adamson Photo . . . . . 87
Adamson Resume . . . . . 87
Walker Photo . . . . . 88
Walker Resume . . . . . 89

**Tillegg B. Bruke DB2-biblioteket . . . . 91**
PDF-filer og trykte bøker for DB2 . . . . . 91
DB2-informasjon . . . . . 91
Skrive ut PDF-bøkene . . . . . 101
Bestille trykte bøker . . . . . 102
DB2-dokumentasjon på systemet . . . . . 103
Få tilgang til hjelp på systemet . . . . 103
Få informasjon på systemet . . . . . 105
Bruke DB2-veivisere . . . . . 108
Konfigurere en dokumenttjener . . . . 109
Søke etter informasjon på systemet . . . 109

**Tillegg C. Merknader . . . . . 111**
Varemerker . . . . . 114

**Stikkordregister . . . . . 117**

**Kontakte IBM . . . . . 121**
Produktinformasjon . . . . . 121

# Velkommen

Denne boken skal gi brukere en innføring i SQL (SQL=Structured Query Language) og relasjonsdatabaser. Den:

- beskriver grunnleggende begreper i SQL som brukes i DB2
- forklarer hvordan du utfører databasemanipuleringsoppgaver
- viser oppgaver gjennom enkle eksempler

Hvis du er den systemansvarlige, burde du gjøre dette før du prøver ut noen av eksemplene i denne boken:

- Installer og konfigurer tjeneren slik det er beskrevet i *begynnerbøker* for operativsystemet. Opprett SAMPLE-databasen ved hjelp av alternativet ″Første trinn″. SAMPLE-databasen kan også opprettes fra en klarmelding. Du finner flere opplysninger i *SQL Reference*. Merk: Ikke legg inn egne data i DB2 SAMPLE-databasen.
- Opprett bruker-IDen for DB2-administratoren ved å følge instruksjonene i *begynnerbøker*.

Hvis du ikke er systemansvarlig, må du kontrollere at du har en gyldig bruker-ID og riktig autorisasjon og riktige rettigheter til å få tilgang til SAMPLE-databasen.

## Beslektet dokumentasjon for denne boken

Bøkene nedenfor kan være nyttige:

| | |
|---|---|
| *begynnerbøker* | Inneholder opplysninger som er nødvendige for å installere og bruke databasesystemet. |
| *SQL Reference* | Inneholder opplysninger om SQL-referanse. |
| *Administration Guide* | Inneholder opplysninger som er nødvendige for å utforme, implementere og vedlikeholde en database som blir brukt lokalt eller i et klient-/tjener-miljø. |
| *Application Development Guide* | Beskriver applikasjonsutviklingsprosessen og hvordan du koder, kompilerer og kjører applikasjonsprogrammer som bruker innfelt SQL til å få tilgang til databasen eller til å kjøre som DB2-lagrede prosedyrer ved hjelp av SQL-prosedyrespråket (eller andre programmeringsspråk som støttes). |

## Konvensjoner

Konvensjonene nedenfor blir brukt i denne boken.

| | |
|---|---|
| **Fete typer** | Brukes i eksempler til å vise kommandoer og nøkkelord som er forhåndsdefinert av systemet. |
| *Kursiv* | Viser ett av disse punktene:<br>• Innføringen av en ny term<br>• En referanse til en annen informasjonskilde |
| STORE BOKSTAVER | Viser ett av disse punktene:<br>• Kommandoer og nøkkelord som er forhåndsdefinert av systemet<br>• Eksempler på bestemte dataverdier eller kolonnenavn |

# Kapittel 1. Relasjonsdatabaser og SQL

I en *relasjonsdatabase* blir data lagret i *tabeller*. En tabell er en samling av *rader* og *kolonner*. Du finner et grafisk eksempel av en tabell i figur 1 på side 3. Kolonner (vertikale) og rader (horisontale) er avmerket i figuren. *SQL* blir brukt til å hente eller oppdatere data ved å spesifisere kolonner, tabeller og de forskjellige forholdene mellom dem.

SQL er et standardisert språk for å definere og manipulere data i en relasjonsdatabase. SQL-setninger utføres av et *databasesystem*. Et databasesystem er et dataprogram som styrer data.

En *partisjonert* relasjonsdatabase er en relasjonsdatabase der data styres over flere partisjoner (også kalt *noder*). En enkel måte å tenke på partisjoner på er å se på hver partisjon som en fysisk datamaskin. I denne boken kommer vi til å fokusere på enkeltpartisjonsdatabaser.

Du kan få tilgang til eksempeldatabasen og prøve ut alle eksemplene i denne boken gjennom *interaktiv* SQL ved hjelp av et grensesnitt, for eksempel kommandolinjebehandleren (CLP) eller kommandosenteret (CC).

# Kapittel 2. Organisere data

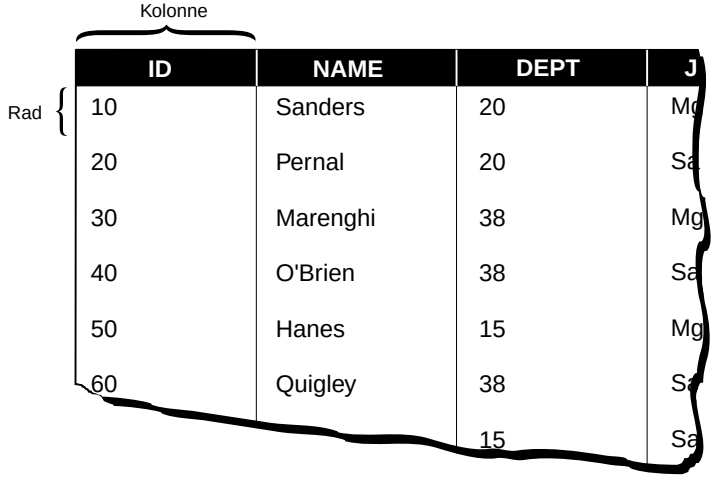
Dette kapittelet presenterer viktige begrepsbeskrivelser av *tabeller, utsnitt* og *skjemaer*. Det er en generell oversikt som viser tilknytningen mellom forskjellige byggekomponenter i en relasjonsdatabase. Den siste delen inneholder en kort beskrivelse av noen av de viktigste og oftest brukte datatypene.

## Tabeller

Tabellene er logiske strukturer som er laget av et definert antall *kolonner* og et variabelt antall *rader*. En kolonne er et sett med verdier av den samme datatypen. En rad er en sekvens med verdier som utgjør en enkelt post i tabellen. Radene er ikke nødvendigvis i rekkefølge i tabellen. Hvis du vil sortere resultatsettet, må du be eksplisitt om sortering i SQL-setningen som velger data fra tabellen. I skjæringspunktet til hver kolonne og rad er det en bestemt datapost som kalles en *verdi*. I figur 1, er 'Sanders' et eksempel på en verdi i tabellen.

En *basistabell* inneholder brukerdata og blir opprettet med CREATE TABLE-setningen. En *resultattabell* er et sett med rader som databasesystemet velger eller genererer fra en eller flere basistabeller for å oppfylle en spørring.

I figur 1 blir det vist en del av en tabell. Kolonner og rader er merket.




*Figur 1. Visualisering av en tabell*

## Utsnitt

Et *utsnitt* gir en annen måte å se på dataene i en eller flere tabeller på. Det er et dynamisk vindu for tabeller.

Utsnitt tillater flere brukere å se på forskjellige presentasjoner av de samme dataene. Flere brukere kan for eksempel bruke en tabell med data om de ansatte. En avdelingssjef ser data om sine ansatte, men ikke om ansatte i en annen avdeling. En rekrutteringssjef ser ansettelsesdatoene til alle de ansatte, men ikke lønningene, mens en økonomisjef ser lønningene, men ikke ansettelsesdatoene. Hver av disse brukerne bruker et utsnitt som er utledet fra den virkelige tabellen. Hvert utsnitt ser ut som en tabell og har sitt eget navn.

En fordel ved å bruke utsnitt er at du kan bruke dem til å kontrollere tilgangen til sensitive data. Slik kan forskjellige personer ha tilgang til forskjellige kolonner eller rader med data.

## Skjemaer

Et *skjema* er en samling med navngitte objekter (tabeller og utsnitt for eksempel). Et skjema gir en logisk klassifikasjon av objekter i databasen.

Et skjema blir opprettet implisitt når du oppretter en tabell, et utsnitt eller et annet navngitt objekt. Du kan også opprette det eksplisitt ved hjelp av CREATE SCHEMA-setningen.

Når du oppretter et navngitt objekt, kan du *kvalifisere* (tilknytte) navnet til objektet med navnet på det bestemte skjemaet. Navngitte objekter har todelte navn der den første delen av navnet er skjemanavnet som objektet er tildelt. Hvis du ikke oppgir et skjemanavn, blir objektet tildelt til standardskjemaet. (Navnet på standardskjemaet er *autorisasjons-IDen* til brukeren som utfører setningen.)

For interaktiv SQL, metoden som brukes til å utføre eksemplene i denne boken, er autorisasjons-IDen bruker-IDen som er oppgitt med CONNECT-setningen. Hvis for eksempel navnet på en tabell er STAFF og bruker-IDen som er oppgitt er USERXYZ, blir det kvalifiserte tabellnavnet USERXYZ.STAFF. Du finner detaljerte opplysninger om CONNECT-setningen i “Tilkoble til en database” på side 18.

Noen skjemanavn er reservert. *Innebygde funksjoner* er for eksempel i SYSIBM-skjemaet, mens de forhåndsinstallerte *brukerdefinerte funksjonene* tilhører SYSFUN-skjemaet. Du finner detaljerte opplysninger om CREATE SCHEMA-setningen i *SQL Reference*.

## Datatyper

Datatyper definerer akseptable verdier for konstanter, kolonner, vertsvariabler, funksjoner, uttrykk og spesialregistre. Denne delen beskriver datatypene som det er henvist til i eksemplene. Du finner en fullstendig liste og fullstendige beskrivelser av andre datatyper i *SQL Reference*.

### Tegnstreng

En *tegnstreng* er en sekvens med byte. Lengden på strengen er antall byte i sekvensen. Hvis lengden er null, kalles verdien en *tom streng*.

#### Tegnstreng med fast lengde

CHAR(x) er en streng med fast lengde. Lengdeattributtet x må være mellom 1 og 254.

#### Tegnstreng med variabel lengde

Det er tre typer tegnstrenger med variabel lengde: VARCHAR, LONG VARCHAR og CLOB.

VARCHAR(x)-typer er tegnstrenger med variabel lengde, slik at en streng med lengde 9 kan settes inn i VARCHAR(15), men vil fremdeles ha strenglengden 9.

Du finner detaljerte opplysninger om CLOB i “Store objekter (LOB)” på side 67.

### Grafisk streng

En *grafisk streng* er en sekvens med dobbeltbytetegndata.

#### Grafisk streng med fast lengde

GRAPHIC(x) er en streng med fast lengde. Lengdeattributtet x må være mellom 1 og 127.

#### Grafisk streng med variabel lengde

Det er tre typer grafiske strenger med variabel lengde: VARGRAPHIC, LONG VARGRAPHIC og DBCLOB. Du finner detaljerte opplysninger om DBCLOB i “Store objekter (LOB)” på side 67.

### Binærstreng

En *binærstreng* er en sekvens med byte. Den brukes til å oppbevare utradisjonelle data, for eksempel bilder. BLOB (Binary Large Objects) er en binærstreng. Du finner flere opplysninger under “Store objekter (LOB)” på side 67.

### Numre

Alle numrene har et tegn og en *presisjon*. Presisjonen er antallet biter eller sifre bortsett fra tegnet.

**SMALLINT**
En *SMALLINT (lite heltall)* er et heltall på to byte med en presisjon på fem sifre.

**INTEGER**
Et *INTEGER (stort heltall)* er et heltall på fire byte med en presisjon på ti sifre.

**BIGINT**
Et *BIGINT (big integer)* er et heltall på åtte byte med en presisjon på 19 sifre.

**REAL** *REAL (flytetall med enkeltpresisjon)* er en 32-biters tilnærming til et reelt tall.

**DOUBLE**
*DOUBLE (flytetall med dobbeltpresisjon)* er en 64-biters tilnærming til et reelt tall. DOUBLE blir også henvist til som FLOAT.

**DECIMAL(p,s)**

*DECIMAL* er et desimaltall. Plasseringen til desimalpunktet blir bestemt av *presisjonen (p)* og *skalaen (s)* til tallet. Presisjonen er totalt antall sifre og må være mindre enn 32. Skala er antall sifre i brøkdelen og er alltid mindre enn eller lik verdien av presisjonen. Desimalverdien settes til standardverdien som er en presisjon på fem og skala på 0, hvis presisjon og skala ikke er oppgitt.

### Datetime-verdier

Datetime-verdier er fremstillinger av datoer, klokkeslett og systemtider (en tegnstreng med 14 sifre som representerer en dato og et klokkeslett i formatet *ååååxxddttmmss*). Datetime-verdier kan brukes i bestemte aritmetiske operasjoner og strengoperasjoner og de er kompatible med bestemte strenger, de er imidlertid verken strenger eller tall.[1]

**Dato** En *dato* er en tredelt verdi (år, måned og dag).

**Klokkeslett**
Et *klokkeslett* er en tredelt verdi (time, minutt og sekund) som angir et tidspunkt på dagen i 24-timers format.

1. I denne boken refererer vi til ISO-fremstillinger av datetime-verdier.

**Systemtid**

En *systemtid* er en sjudelt verdi (år, måned, dag, time, minutt, sekund og mikrosekund) som viser til en dato og et klokkeslett.

**Nullverdi**

*Null*-verdien er en spesiell verdi som er forskjellig fra alle verdier som ikke er null. Det betyr fravær av enhver annen verdi for den kolonnen i raden. Nullverdien finnes for alle datatyper.

Tabellen nedenfor uthever egenskaper ved datatypene som brukes i eksemplene. Alle numeriske datatyper er definert i et visst område. Området med numeriske datatyper er også inkludert i denne tabellen. Du kan bruke denne tabellen som en hurtigreferanse for riktig bruk av datatype.

| Datatype | Type | Egenskap | Eksempel eller område |
|---|---|---|---|
| CHAR(15) | Tegnstreng med fast lengde | Maksimumslengde på 254 | 'Sunny day ' |
| VARCHAR(15) | tegnstreng med variabel lengde | Maksimumslengde på 32672 | 'Sunny day' |
| SMALLINT | antall | lengde på 2 byte presisjon på 5 sifre | området er fra -32768 til 32767 |
| INTEGER | antall | lengde på 4 byte presisjon på 10 sifre | området er fra -2147483648 til 2147483647 |
| BIGINT | antall | lengde på 8 byte presisjon på 19 sifre | området er fra -9223372036854775808 til 9223372036854775807 |
| REAL | antall | flytetall med enkeltpresisjon som er en 32-biters tilnærming | området er fra -3.402E+38 til -1.175E-37 eller fra 1.175E-37 til -3.402E+38 eller null |
| DOUBLE | antall | flytepunkt med dobbeltpresisjon som er en 64-biters tilnærming | området er fra -1.79769E+308 til -2.225E-307 eller fra 2.225E-307 til 1.79769E+308 eller null |
| DECIMAL(5,2) | antall | presisjon på 5 skala på 2 | området er fra -10**31+1 til 10**31-1 |
| DATE | datetime | tredelt verdi | 1991-10-27 |
| TIME | datetime | tredelt verdi | 13.30.05 |
| TIMESTAMP | datetime | sjudelt verdi | 1991-10-27-13.30.05.000000 |

Du finner flere opplysninger i tabellen for datatypekompatibilitet i *SQL Reference*.

# Kapittel 3. Opprette tabeller og utsnitt

Dette kapittelet beskriver hvordan du kan opprette og manipulere tabeller og utsnitt i DB2 Universal Database. Forholdet mellom tabeller og utsnitt blir undersøkt ved hjelp av diagrammer og eksempler.

Dette kapittelet inneholder:

- Opprette tabeller
- Legge inn data
- Endre data
- Slette data
- Opprette utsnitt
- Bruke utsnitt til å manipulere data

## Opprette tabeller

Opprett din egen tabell ved hjelp av CREATE TABLE-setningen og oppgi kolonnenavnene og typene, i tillegg til *begrensninger*. Begrensninger blir omtalt i “Håndheve forretningsregler med begrensninger og utløsere” på side 51.

Setningen nedenfor oppretter en tabell som heter PERS, som likner på STAFF-tabellen, men den har en ekstra kolonne for fødselsdato.

```
CREATE TABLE PERS
    ( ID              SMALLINT        NOT NULL,
      NAME            VARCHAR(9),
      DEPT            SMALLINT WITH DEFAULT 10,
      JOB             CHAR(5),
      YEARS           SMALLINT,
      SALARY          DECIMAL(7,2),
      COMM            DECIMAL(7,2),
      BIRTH_DATE      DATE)
```

Denne setningen oppretter en tabell uten data. Det neste avsnittet forklarer hvordan du legger inn data i en ny tabell.

Som det er vist i dette eksempelet kan du legge inn både et navn og en datatype for hver kolonne. Datatyper blir omtalt i “Datatyper” på side 5. NOT NULL er valgfri og kan oppgis for å vise at nullverdier ikke er tillatt i en kolonne. Standardverdier er også valgfrie.

Det finnes mange andre alternativer du kan oppgi i en CREATE TABLE-setning, for eksempel *entydige begrensninger eller referansebegrensninger*. Du finner flere opplysninger om alle alternativene under CREATE TABLE-setningen i *SQL Reference*.

## Legge inn data

Når du oppretter en ny tabell, inneholder den ingen data. Du legger nye rader inn i en tabell ved hjelp av INSERT-setningen. Denne setningen har to generelle formater:

- Med det ene formatet kan du bruke et VALUES-ledd til å oppgi verdier for kolonnene i en eller flere rader. De tre neste eksemplene legger data inn i tabellene ved hjelp av dette generelle formatet.
- Med det andre formatet oppgir du en *full SELECT*-setning for å identifisere kolonner fra rader i andre tabeller og/eller utsnitt, i stedet for å oppgi VALUES.

En Full SELECT-setning er en spørresetning som brukes i INSERT- eller CREATE VIEW-setninger, eller etter et predikat. En full SELECT-setning som er omgitt av parenteser blir vanligvis kalt en *delspørring*.

Avhengig av standardvalgene du valgte da du opprettet tabellen, oppgir du en verdi for hver kolonne eller godtar en standardverdi, for hver rad du legger inn. Standardverdiene for de forskjellige datatypene er omtalt i *SQL Reference*.

Setningen nedenfor bruker et VALUES-ledd for å sette inn en rad med data i tabellen PERS:

```
INSERT INTO PERS
   VALUES (12, 'Harris', 20, 'Sales', 5, 18000, 1000, '1950-1-1')
```

Setningen nedenfor bruker et VALUES-ledd til å sette inn tre rader i tabellen PERS der bare IDer, navn og jobber er kjent. Hvis en kolonne er definert som NOT NULL og den ikke har en standardverdi, må du oppgi en verdi for den.

NOT NULL-leddet på en kolonnedefinisjon i en CREATE TABLE-setning kan utvides med ordene WITH DEFAULT. Hvis en kolonne er definert som NOT NULL WITH DEFAULT eller en konstant standardverdi, for eksempel WITH DEFAULT 10, og du ikke oppgir kolonnen i kolonnelisten, blir standardverdien satt inn i den kolonnen i den innskutte raden. I CREATE TABLE-setningen ble det for eksempel bare oppgitt en verdi for kolonnen DEPT og den ble definert til 10. Derfor er avdelingsnummeret (DEPT) definert til 10 og alle andre kolonner som ikke har fått en verdi, blir definert til NULL.

```
INSERT INTO PERS (NAME, JOB, ID)
   VALUES ('Swagerman', 'Prgmr', 500),
          ('Limoges', 'Prgmr', 510),
          ('Li', 'Prgmr', 520)
```

Setningen nedenfor returnerer resultatet av innsettingene:

```
SELECT *
   FROM PERS

ID     NAME      DEPT   JOB   YEARS  SALARY    COMM      BIRTH_DATE
------ --------- ------ ----- ------ --------- --------- ----------
    12 Harris        20 Sales      5  18000.00   1000.00 01/01/1950
   500 Swagerman     10 Prgmr      -         -         - -
   510 Limoges       10 Prgmr      -         -         - -
   520 Li            10 Prgmr      -         -         - -
```

Legg merke til at i dette tilfellet ble det ikke oppgitt verdier for hver kolonne. NULL-verdier blir vist som en tankestrek (–). For at dette skal fungere må listen med kolonnenavn samsvare med verdiene i VALUES-leddet både når det gjelder rekkefølge og datatype. Hvis listen over kolonnenavn er utelatt (slik den var i det første eksempelet), må listen over dataverdier etter VALUES være i samme rekkefølge som kolonnene i tabellen som de blir satt inn i, og antall verdier må samsvare med antall kolonner i tabellen.

Hver verdi må være kompatibel med datatypen i kolonnen som den blir satt inn i. Hvis en kolonne er definert slik at den kan ha nullverdier og det ikke er oppgitt en verdi for den kolonnen, blir verdien NULL gitt til den kolonnen i den innskutte raden.

Eksempelet nedenfor setter inn nullverdien i YEARS, COMM og BIRTH_DATE siden det ikke er oppgitt noen verdier for de kolonnene i raden.

```
INSERT INTO PERS (ID, NAME, JOB, DEPT, SALARY)
   VALUES (410, 'Perna', 'Sales', 20, 20000)
```

Den andre formen av INSERT-setningen er veldig praktisk når du skal legge inn verdier fra rader som finnes i en annen tabell. Du oppgir en full SELECT-setning for å identifisere kolonner fra rader i andre tabeller og/eller utsnitt, i stedet for å oppgi VALUES.

Eksempelet nedenfor velger data fra tabellen STAFF for medlemmer av avdeling 38 og setter dataene inn i tabellen PERS:

```
INSERT INTO PERS (ID, NAME, DEPT, JOB, YEARS, SALARY)
   SELECT ID, NAME, DEPT, JOB, YEARS, SALARY
      FROM STAFF
      WHERE DEPT = 38
```

Etter denne innsettingen gir den neste SELECT-setningen et resultat som er likt full SELECT-setningen i INSERT-setningen.

```
SELECT ID, NAME, DEPT, JOB, YEARS, SALARY
   FROM PERS
   WHERE DEPT = 38
```

Resultatet er:

```
ID     NAME      DEPT   JOB   YEARS  SALARY
------ --------- ------ ----- ------ ---------
    30 Marenghi      38 Mgr        5  17506.75
    40 O'Brien       38 Sales      6  18006.00
    60 Quigley       38 Sales      -  16808.30
   120 Naughton      38 Clerk      -  12954.75
   180 Abrahams      38 Clerk      3  12009.75
```

## Endre data

Bruk UPDATE-setningen til å endre dataene i en tabell. Med denne setningen kan du endre verdien i en eller flere kolonner for hver rad som oppfyller søkebetingelsen i WHERE-leddet.

Eksempelet nedenfor oppdaterer opplysningene om den ansatte som har IDen 410:

```
UPDATE PERS
   SET JOB='Prgmr', SALARY = SALARY + 300
   WHERE ID = 410
```

SET-leddet oppgir kolonnene som skal oppdateres, og gir verdiene.

WHERE-leddet er valgfritt og oppgir hvilke rader som skal oppdateres. Hvis WHERE-leddet utelates, oppdaterer databasesystem hver rad eller utsnitt i tabellen med verdiene du oppgir.

I dette eksempelet blir først tabellen (PERS) navngitt, deretter blir det oppgitt en betingelse for raden som skal oppdateres. Opplysningene for funksjonærnummer 410 er endret: den ansattes stilling ble endret til Prgmr og lønnen økte med $300.

Du kan endre data i mer enn en rad ved å inkludere et WHERE-ledd som gjelder for to eller flere rader. Eksempelet nedenfor øker lønnen til alle salgsmedarbeidere med 15%:

```
UPDATE PERS
   SET SALARY = SALARY * 1.15
   WHERE JOB = 'Sales'
```

## Slette data

Bruk DELETE-setningen til å slette rader med data fra en tabell som er basert på søkebetingelsen som ble oppgitt i WHERE-leddet. Eksempelet nedenfor sletter raden der funksjonær-IDen er 120:

```
DELETE FROM PERS
   WHERE ID = 120
```

WHERE-leddet er valgfritt og oppgir hvilke rader som skal slettes. Hvis WHERE-leddet er utelatt, sletter databasesystemet alle radene i tabellen eller utsnittet.

Du kan bruke DELETE-setningen til å slette mer enn en rad. Eksempelet nedenfor sletter alle radene der DEPT til den ansatte er 20:

```
DELETE FROM PERS
   WHERE DEPT = 20
```

Når du sletter en rad, fjerner du hele raden, ikke bestemte kolonneverdier.

Når du skal slette definisjonen av en tabell i tillegg til innholdet, kjører du DROP TABLE-setningen slik det er beskrevet i *SQL Reference*.

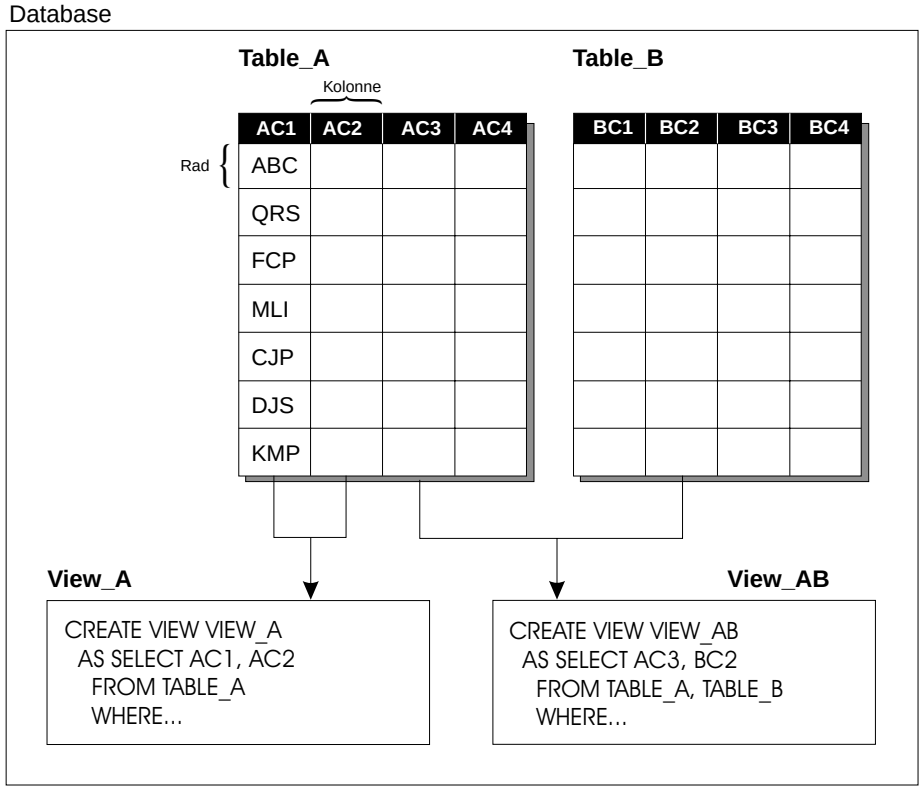
## Opprette utsnitt

Som beskrevet i “Utsnitt” på side 4 gir et utsnitt en annen måte å se på dataene i en eller flere tabeller. Ved å opprette utsnitt kan du begrense opplysningene du vil at forskjellige brukere skal se på. Diagrammet nedenfor viser forholdet mellom utsnitt og tabeller.

I figur 2 på side 14 begrenser View_A tilgangen slik at den bare gjelder for kolonnene AC1 og AC2 i TABLE_A.

View_AB tillater tilgang til kolonne AC3 i TABLE_A og BC2 i TABLE_B.

Ved å opprette View_A begrenser du tilgangen som brukere kan ha til TABLE_A, og ved å opprette VIEW_AB begrenser du tilgang til bestemte kolonner i begge tabellene.




*Figur 2. Forhold mellom tabeller og utsnitt*

Setningen nedenfor oppretter et utsnitt av ikke-avdelingssjefer i avdeling 20 i tabellen STAFF, der lønn og provisjon ikke vises fra basistabellen.

```
CREATE VIEW STAFF_ONLY
   AS SELECT ID, NAME, DEPT, JOB, YEARS
         FROM STAFF
         WHERE JOB <> 'Mgr' AND DEPT=20
```

Når du har opprettet utsnittet, viser setningen nedenfor innholdet i utsnittet:

```
SELECT *
FROM STAFF_ONLY
```

Denne setningen gir dette resultatet:

```
ID     NAME      DEPT   JOB   YEARS
------ --------- ------ ----- ------
    20 Pernal        20 Sales      8
    80 James         20 Clerk      -
   190 Sneider       20 Clerk      8
```

Som et ekstra eksempel kan vi bruke tabellene STAFF og ORG til å opprette et utsnitt som viser navnet på hver avdeling og navnet på avdelingssjefen for den avdelingen. Setningen nedenfor oppretter dette utsnittet:

```
CREATE VIEW DEPARTMENT_MGRS
   AS SELECT NAME, DEPTNAME
         FROM STAFF, ORG
         WHERE MANAGER = ID
```

Du kan legge ekstra begrensninger på innsettinger i og oppdateringer av en tabell gjennom et utsnitt ved å bruke WITH CHECK OPTION-leddet når du oppretter et utsnitt. Dette leddet gjør at databasesystemet godkjenner at eventuelle oppdateringer av eller innsettinger i utsnittet samsvarer med utsnittsdefinisjonen, og avviser de som ikke gjør det. Hvis du utelater dette leddet, blir ikke innsettinger kontrollert mot utsnittsdefinisjonen. Hvis du ønsker opplysninger om hvordan WITH CHECK OPTION fungerer, kan du se på CREATE VIEW-setningen i *SQL Reference*.

## Bruke utsnitt til å manipulere data

På samme måte som SELECT-setningen, blir INSERT-, DELETE- og UPDATE-setninger tatt i bruk på et utsnitt som om det skulle vært en ordentlig tabell. Setningene manipulerer dataene i den underliggende basistabellen. Slik at når du ser på utsnittet på nytt, blir det evaluert ved hjelp av den nyeste basistabellen. Hvis du ikke bruker WITH CHECK OPTION-leddet, kan det hende at data som du endrer ved hjelp av et utsnitt, ikke blir vist de andre gangene du ser på utsnittet siden dataene ikke lenger passer til den opprinnelige utsnittsdefinisjonen.

Nedenfor finner du et eksempel der en oppdatering er tatt i bruk på utsnittet FIXED_INCOME:

```
CREATE VIEW FIXED_INCOME (LNAME, DEPART, JOBTITLE, NEWSALARY)
   AS SELECT NAME, DEPT, JOB, SALARY
         FROM PERS
         WHERE JOB <> 'Sales' WITH CHECK OPTION

UPDATE FIXED_INCOME
   SET NEWSALARY = SALARY * 1.10
   WHERE LNAME = 'Li'
```

Oppdateringen i det forrige utsnittet er likeverdig med å (bortsett fra kontrollalternativet) oppdatere basistabellen PERS:

```
UPDATE PERS
   SET SALARY = SALARY * 1.10
   WHERE NAME = 'Li'
     AND JOB <> 'Sales'
```

Legg merke til at siden utsnittet ble opprettet ved hjelp av WITH CHECK OPTION for begrensningen JOB <> 'Sales' i CREATE VIEW FIXED_INCOME, er ikke den følgende oppdateringen tillatt når Limoges flytter over til salg:

```
UPDATE FIXED_INCOME
   SET JOBTITLE = 'Sales'
   WHERE LNAME = 'Limoges'
```

Kolonner som er definert av uttrykk som SALARY + COMM eller SALARY * 1,25 kan ikke oppdateres. Hvis du definerer et utsnitt som inneholder en eller flere slike kolonner, mottar ikke eieren UPDATE-rettigheten for disse kolonnene. INSERT-setninger er ikke tillatt på utsnitt som inneholder slike kolonner, men DELETE-setninger er tillatt.

Vurder en PERS-tabell der ingen kolonner er definert som NOT NULL. Du kan sette inn rader i PERS-tabellen gjennom utsnittet FIXED_INCOME selv om det ikke inneholder ID, YEARS, COMM eller BIRTHDATE fra den underliggende tabellen PERS. Kolonner som ikke er synlige i utsnittet, defineres til NULL eller standardverdien.

PERS-tabellen har imidlertid kolonne-ID definert som NOT NULL. Hvis du prøver å sette inn en rad gjennom utsnittet FIXED_INCOME, prøver systemet å sette inn NULL-verdier i alle PERS-kolonnene som er “usynlige” i utsnittet. Siden ID-kolonnen ikke er inkludert i utsnittet og ikke tillater nullverdier, tillater ikke systemet innsettingen gjennom utsnittet.

Hvis du ønsker flere opplysninger om regler og begrensninger, kan du se på CREATE VIEW-setningen i *SQL Reference*.

# Kapittel 4. Bruke SQL-setninger for å få tilgang til data

Dette avsnittet beskriver hvordan du kobler deg til en database og henter data ved hjelp av SQL-setninger.

I eksemplene presenterer vi setningen som skal oppgis, etterfulgt (i de fleste tilfeller) av resultatene som blir vist når setningen blir kjørt mot eksempeldatabasen. Legg merke til at selv om vi viser setningene med store bokstaver, så kan du skrive dem med en hvilken som helst blanding av store og små bokstaver (bortsett fra når de er omgitt av enkeltanførselstegn (') eller anførselstegn ('') ).

SAMPLE-databasen som følger med DB2 Universal Database, består av flere tabeller som er vist i "Tillegg A. Eksempeldatabasetabeller" på side 71. Denne databasen kan opprettes ved hjelp av installeringsstartpanelet ″Første trinn″. Du kan også opprette SAMPLE-databasen fra kommandolinjen. Du finner flere opplysninger i *SQL Reference*.

Legg merke til at det er inkludert ekstra eksempeldatabaser i DB2 Universal Database for å vise funksjonaliteten i Data Warehouse Center og OLAP Starter Kit. Eksemplene i denne boken bruker bare den generelle SAMPLE-databasen.

Det kan hende at du må kvalifisere tabellnavn som er brukt, ved å sette skjemanavnet og et punktum foran dem, avhengig av hvordan databasen er konfigurert. I denne boken er for eksempel skjemanavnet antatt å være USERID. Så du kan henvise til tabellen ORG som USERID.ORG. Spør administratoren om dette er nødvendig.

Dette kapittelet dekker disse emnene:

- Tilkoble til en database
- Undersøke problemer
- Velge kolonner og Velge rader
- Sortere rader og Fjerne like rader
- Rekkefølge på operasjoner
- Bruke uttrykk til å beregne verdier
- Navngi uttrykk
- Velge data fra mer enn en tabell
- Bruke en delspørring
- Bruke funksjoner
- Gruppering

## Tilkoble til en database

Du må koble deg til en database før du kan bruke SQL-setninger til spørringer i databasen eller til å manipulere den. CONNECT-setningen knytter en databasetilkobling til en bruker-ID.

Hvis du for eksempel skal koble deg til SAMPLE-databasen, skriver du denne kommandoen i DB2 kommandolinjebehandler :

```
CONNECT TO SAMPLE USER USERID USING PASSWORD
```

(Pass på at du velger en bruker-ID og et passord som er gyldig på tjenersystemet.)

I dette eksempelet er USER USERID og USING er PASSWORD.

Meldingen nedenfor viser at tilkoblingen var vellykket:

```
Database Connection Information

   Database product       = DB2/NT 7.1.0
   SQL authorization ID   = USERID
   Local database alias   = SAMPLE
```

Når du er tilkoblet, kan du starte manipuleringen av databasen. Hvis du ønsker flere opplysninger om tilkoblinger, kan du slå opp på CONNECT-setningen i *SQL Reference*.

## Undersøke problemer

Når du skriver feil i et av eksemplene eller hvis det oppstår en feil under utføringen av en SQL-setning, returnerer databasesystemet en feilmelding. Feilmeldingen består av et meldingsnummer, en kort forklaring og en SQLSTATE.

SQLSTATE-feil er feilkoder som er vanlige for DB2-produktfamilien. SQLSTATE-feil samsvarer med ISO/ANSI SQL92-standarden.

Hvis for eksempel bruker-IDen eller passordet var feil i CONNECT-setningen, ville databasesystemet returnert meldingsnummeret SQL1403N og en SQLSTATE på 08004. Meldingen er som følger:

```
SQL1403N Brukernavnet og/eller passordet som ble oppgitt,
         er ugyldig. SQLSTATE=08004
```

Du kan få flere opplysninger om feilmeldingen ved å skrive et spørsmålstegn (?) og deretter meldingsnummeret eller SQLSTATE:

```
       ? SQL1403N
ELLER
       ? SQL1403
ELLER
       ? 08004
```

Legg merke til at den siste linjen i beskrivelsen av feilen SQL1403N sier at SQLCODE er -1403. SQLCODE er en produktspesifikk feilkode. Meldingsnumre som slutter på N (Notification) eller C (Critical) representerer en feil og har negative SQLCODEr. Meldingsnumre som slutter på W (Warning) representerer en advarsel og har positive SQLCODEr.

## Velge kolonner

Bruk SELECT-setningen til å velge bestemte kolonner fra en tabell. Oppgi en liste over kolonnenavn atskilt med komma, i setningen. Denne listen refereres til som en *SELECT-liste*.

Setningen nedenfor velger avdelingsnavn (DEPTNAME) og avdelingsnumre (DEPTNUMB) fra ORG-tabellen i SAMPLE-databasen:

```
SELECT DEPTNAME, DEPTNUMB
   FROM ORG
```

Setningen ovenfor gir følgende resultat:

```
DEPTNAME       DEPTNUMB
-------------- --------
Head Office          10
New England          15
Mid Atlantic         20
South Atlantic       38
Great Lakes          42
Plains               51
Pacific              66
Mountain             84
```

Ved å bruke en stjerne (*) kan du velge alle kolonnene fra tabellen. Det neste eksempelet viser alle kolonnene radene i ORG-tabellen:

```
SELECT *
   FROM ORG
```

Denne setningen gir dette resultatet:

```
DEPTNUMB DEPTNAME       MANAGER DIVISION   LOCATION
-------- -------------- ------- ---------- -------------
      10 Head Office        160 Corporate  New York
      15 New England         50 Eastern    Boston
      20 Mid Atlantic        10 Eastern    Washington
```

```
38 South Atlantic      30 Eastern    Atlanta
42 Great Lakes        100 Midwest    Chicago
51 Plains             140 Midwest    Dallas
66 Pacific            270 Western    San Francisco
84 Mountain           290 Western    Denver
```

## Velge rader

Når du skal velge bestemte rader fra en tabell, bruker du WHERE-leddet etter SELECT-setningen for å oppgi betingelsen eller betingelsene som en rad må oppfylle for å bli valgt. Et kriterium for å hente rader fra en tabell er en *søkebetingelse*.

En søkebetingelse består av ett eller flere *predikater*. Et predikat spesifiserer en betingelse som er sann eller usann (eller ukjent) om en rad. Du kan spesifisere betingelser i WHERE-leddet ved å bruke følgende basispredikater:

| Predikat | Funksjon |
|---|---|
| x = y | x er lik y |
| x <> y | x er ikke lik y |
| x < y | x er mindre enn y |
| x > y | x er større enn y |
| x <= y | x er mindre enn eller lik y |
| x >= y | x er større enn eller lik y |
| IS NULL/IS NOT NULL | tester om det finnes nullverdier |

Når du lager søkebetingelser, må du passe på å bare utføre aritmetiske operasjoner på numeriske datatyper og å bare sammenlikne kompatible datatyper. Du kan for eksempel ikke sammenlikne tekststrenger med tallverdier.

Hvis du velger rader basert på en tegnverdi, må den verdien være omgitt av enkeltanførselstegn (for eksempel `WHERE JOB = 'Clerk'`) og hver tegnverdi må være skrevet på nøyaktig samme måte som i databasen der den eksisterer. Hvis dataverdien er skrevet med små bokstaver i databasen og du skriver den med store bokstaver, blir det ikke valgt noen rader. Hvis du velger rader basert på en numerisk verdi, må ikke den verdien være omgitt av anførselstegn (for eksempel `WHERE DEPT = 20`).

Eksempelet nedenfor velger bare rader for avdeling 20 fra STAFF-tabellen:

```
SELECT DEPT, NAME, JOB
       FROM STAFF
  WHERE DEPT = 20
```

Denne setningen gir dette resultatet:

```
DEPT   NAME      JOB
------ --------- -----
    20 Sanders   Mgr
    20 Pernal    Sales
    20 James     Clerk
    20 Sneider   Clerk
```

Det neste eksempelet bruker AND til å spesifisere mer enn en betingelse. Du kan spesifisere så mange betingelser som du vil. Eksempelet velger kontoransatte i avdeling 20 fra STAFF-tabellen:

```
SELECT DEPT, NAME, JOB
      FROM STAFF
   WHERE JOB = 'Clerk'
   AND DEPT = 20
```

Denne setningen gir dette resultatet:

```
DEPT   NAME      JOB
------ --------- -----
    20 James     Clerk
    20 Sneider   Clerk
```

En nullverdi forekommer der det ikke er oppgitt noen verdi og kolonnen ikke støtter en standardverdi. Den kan også forekomme der verdien er satt spesifikt til null. Den kan bare forekomme i kolonner som er definert til å støtte nullverdier. Definering og støtte av nullverdier i tabeller omtales i “Opprette tabeller” på side 9.

Bruk predikatene IS NULL og IS NOT NULL for å kontrollere om det finnes en nullverdi.

Setningen nedenfor viser de ansatte som har en kommisjon som ikke er kjent:

```
SELECT ID, NAME
      FROM STAFF
   WHERE COMM IS NULL
```

Denne setningen gir dette resultatet:

```
ID     NAME
------ ---------
    10 Sanders
    30 Marenghi
    50 Hanes
   100 Plotz
   140 Fraye
   160 Molinare
   210 Lu
   240 Daniels
```

```
260 Jones
270 Lea
290 Quill
```

Verdien null er ikke den samme som nullverdien. Setningen nedenfor velger alle i en tabell med null som kommisjon:

```
SELECT ID, NAME
      FROM STAFF
   WHERE COMM = 0
```

Fordi det ikke er noen verdier med null i kolonnen COMM i eksempeltabellen, er resultatsettet som returneres tomt.

Det neste eksempelet velger alle rader der verdien YEARS i STAFF-tabellen er større enn 9:

```
SELECT NAME, SALARY, YEARS
      FROM STAFF
   WHERE YEARS > 9
```

Denne setningen gir dette resultatet:

```
NAME      SALARY    YEARS
--------- --------- ------
Hanes      20659.80     10
Lu         20010.00     10
Jones      21234.00     12
Quill      19818.00     10
Graham     21000.00     13
```

## Sortere rader

Det kan være du vil ha informasjonen returnert i en bestemt rekkefølge. Bruk ORDER BY-leddet til å sortere opplysningene etter verdiene i en eller flere kolonner.

Setningen nedenfor viser de ansatte i avdeling 84 sortert etter antall år ansatt:

```
SELECT NAME, JOB, YEARS
      FROM STAFF
   WHERE DEPT = 84
   ORDER BY YEARS
```

Denne setningen gir dette resultatet:

```
NAME      JOB   YEARS
--------- ----- ------
Davis     Sales      5
Gafney    Clerk      5
Edwards   Sales      7
Quill     Mgr       10
```

Spesifiser ORDER BY som det siste leddet i hele SELECT-setningen. Kolonnenavn i dette leddet kan være uttrykk eller enhver kolonne i tabellen. Det er ikke nødvendig å spesifisere kolonnenavnene i ORDER BY-leddet i SELECT-listen.

Du kan sortere radene i stigende eller synkende rekkefølge ved å eksplisitt spesifisere enten ASC eller DESC i et ORDER BY-ledd. Hvis ingen av delene er oppgitt, blir radene automatisk sortert i stigende rekkefølge. Setningen nedenfor viser de ansatte i avdeling 84 i synkende rekkefølge etter antall år ansatt:

```
SELECT NAME, JOB, YEARS
      FROM STAFF
   WHERE DEPT = 84
   ORDER BY YEARS DESC
```

Denne setningen gir dette resultatet:

```
NAME      JOB   YEARS
--------- ----- ------
Quill     Mgr       10
Edwards   Sales      7
Davis     Sales      5
Gafney    Clerk      5
```

Du kan sortere rader etter tegnverdier i tillegg til numeriske verdier. Setningen nedenfor viser de ansatte i avdeling 84 i alfabetisk rekkefølge etter navn:

```
SELECT NAME, JOB, YEARS
      FROM STAFF
   WHERE DEPT = 84
   ORDER BY NAME
```

Denne setningen gir dette resultatet:

```
NAME      JOB   YEARS
--------- ----- ------
Davis     Sales      5
Edwards   Sales      7
Gafney    Clerk      5
Quill     Mgr       10
```

## Fjerne like rader

Når du bruker SELECT-setningen kan det være at du ikke vil at like opplysninger skal returneres. STAFF har for eksempel en DEPT-kolonne der flere avdelingsnumre er oppført mer enn en gang, og en JOB-kolonne der flere jobbeskrivelser er oppført mer enn en gang.

Du bruker DISTINCT-alternativet på SELECT-leddet for å fjerne like rader. Hvis du for eksempel setter inn DISTINCT i setningen, blir hver jobb i en avdeling bare oppført en gang:

```
SELECT DISTINCT DEPT, JOB
      FROM STAFF
   WHERE DEPT < 30
   ORDER BY DEPT, JOB
```

Denne setningen gir dette resultatet:

```
DEPT   JOB
------ -----
    10 Mgr
    15 Clerk
    15 Mgr
    15 Sales
    20 Clerk
    20 Mgr
    20 Sales
```

DISTINCT har eliminert alle radene som inneholder like data i settet med kolonner som er spesifisert i SELECT-setningen.

## Rekkefølge på operasjoner

Det er viktig å ta hensyn til rekkefølgen på operasjonene. Utdatene fra ett ledd er inndataene i det neste, slik det er vist i listen nedenfor. Et eksempel der rekkefølgen på operasjonene er vurdert, er presentert i “Navngi uttrykk” på side 25.

Følgende rekkefølge på operasjoner er ikke nødvendigvis den måten operasjoner blir utført på i DB2-koden. Denne enkle forklaringen gir bare en mer intuitiv måte å tenke på når det gjelder spørringer. Rekkefølgen på operasjoner er som følger:

1. FROM-ledd
2. WHERE-ledd
3. GROUP BY-ledd
4. HAVING-ledd
5. SELECT-ledd
6. ORDER BY-ledd

## Bruke uttrykk til å beregne verdier

Et *uttrykk* er en beregning eller funksjon som du inkluderer i en setning. Følgende setning beregner hva lønnen for hver ansatt i avdeling 38 ville vært hvis hver av dem mottok en bonus på $500:

```
SELECT DEPT, NAME, SALARY + 500
      FROM STAFF
      WHERE DEPT = 38
   ORDER BY 3
```

Dette resultatet er:

```
DEPT   NAME      3
------ --------- ----------------
    38 Abrahams          12509.75
    38 Naughton          13454.75
    38 Quigley           17308.30
    38 Marenghi          18006.75
    38 O'Brien           18506.00
```

Legg merke til at kolonnen for den tredje kolonnen er en tallverdi. Dette er et systemgenerert tall, siden SALARY+500 ikke oppgir et kolonnenavn. Senere blir dette tallet brukt i ORDER BY-leddet for å referere til den tredje kolonnen. “Navngi uttrykk” omtaler hvordan du skal gi forståelige navn til uttrykkene.

Du kan lage aritmetiske uttrykk ved å bruke de grunnleggende aritmetiske operatorene for addisjon (+), subtraksjon (−), multiplikasjon (*) og divisjon (/).

Operatorene kan virke på numeriske verdier fra flere forskjellige typer operander, noen av dem er:

- Kolonnenavn (som i RATE * HOURS)
- Konstantverdier (som i RATE * 1.07)
- Skalarfunksjoner (som i LENGTH(NAME) + 1).

## Navngi uttrykk

Det valgfrie AS-leddet lar deg tildele en forståelig navn til et uttrykk, som gjør det lettere å referere til uttrykket. Du kan bruke AS-leddet til å gi et navn til ethvert element i SELECT-listen.

Setningen nedenfor viser alle de ansatte som har en lønn og kommisjon på mindre enn $13, 000. Uttrykket SALARY + COMM er kalt PAY:

```
SELECT NAME, JOB, SALARY + COMM AS PAY
         FROM STAFF
   WHERE (SALARY + COMM) < 13000
   ORDER BY PAY
```

Denne setningen gir dette resultatet:

```
NAME      JOB   PAY
--------- ----- ----------
Yamaguchi Clerk   10581.50
Burke     Clerk   11043.50
Scoutten  Clerk   11592.80
Abrahams  Clerk   12246.25
Kermisch  Clerk   12368.60
Ngan      Clerk   12714.80
```

Ved å bruke AS-leddet kan du referere til et bestemt kolonnenavn i stedet for det systemgenererte navnet i ORDER BY-leddet. I dette eksempelet sammenlikner vi (SALARY + COMM) med 13000 i WHERE-leddet, i stedet for å bruke navnet PAY. Dette er et resultat av rekkefølgen på operasjonene. WHERE-leddet vurderes før (SALARY + COMM) får navnet PAY, fordi SELECT-leddet utføres etter WHERE-leddet. Derfor kan ikke PAY brukes i predikatet.

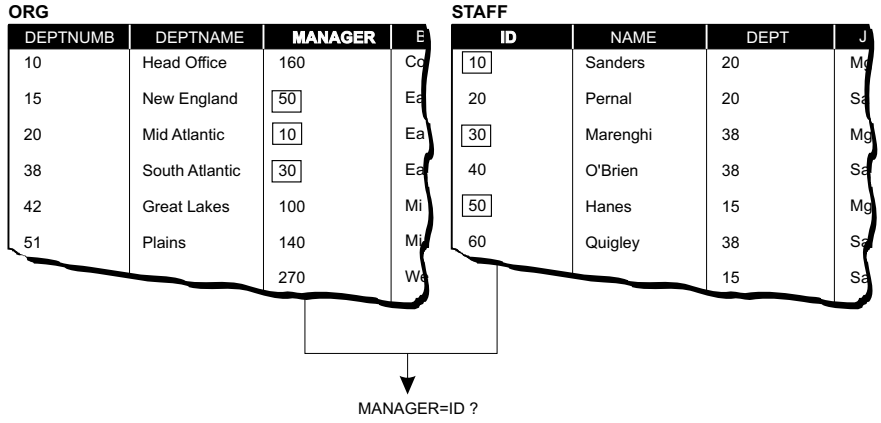
## Velge data fra mer enn en tabell

Du kan bruke SELECT-setningen til å produsere rapporter som inneholder opplysninger fra to eller flere tabeller. Dette blir vanligvis referert til som en *kombinering*. Du kan for eksempel kombinere data fra STAFF- og ORG-tabellene for å lage en ny tabell. Når du skal kombinere to tabeller, spesifiserer du kolonnene som du vil at skal vises i SELECT-leddet, tabellnavnene i et FROM-ledd og søkebetingelsen i WHERE-leddet. WHERE-leddet er valgfritt.

Det neste eksempelet knytter navnet til hver avdelingssjef til et avdelingsnavn. Du må velge informasjon fra to tabeller siden informasjonen om de ansatte (STAFF-tabell) og avdelingsinformasjonen (ORG-tabell) er lagret separat. Følgende spørringer velger kolonnene NAME og DEPTNAME for henholdsvis STAFF- og ORG-tabellen. Søkebetingelsen begrenser utvalget til rader der verdiene i MANAGER-kolonnen er de samme som verdiene i ID-kolonnen:

```
SELECT DEPTNAME, NAME
   FROM ORG, STAFF
         WHERE MANAGER = ID
```

I figur 3 på side 27 blir det vist hvordan kolonner i to forskjellige tabeller kan sammenliknes. Verdiene i rutene viser et samsvar der søkebetingelsen er oppfylt.




*Figur 3. Velge fra STAFF- og ORG-tabeller*

SELECT-setningen gir følgende resultat:

```
DEPTNAME       NAME
-------------- ---------
Mid Atlantic   Sanders
South Atlantic Marenghi
New England    Hanes
Great Lakes    Plotz
Plains         Fraye
Head Office    Molinare
Pacific        Lea
Mountain       Quill
```

Resultatet viser navnet på hver avdelingssjef og hans eller hennes avdeling.

## Bruke en delspørring

Når du skriver en SQL SELECT-setning, kan du plassere ekstra SELECT-setninger i WHERE-leddet. Hver ekstra SELECT-setning starter en delspørring.

En delspørring kan deretter inkludere en annen separat delspørring, denne spørringens resultat blir erstattet i den opprinnelige delspørringens WHERE-ledd. I tillegg kan et WHERE-ledd inneholde delspørringer i mer enn en søkebetingelse. Delspørringen kan referere til tabeller og kolonner som er forskjellige fra de som er brukt i hovedspørringen.

Setningen nedenfor velger avdelingen og plasseringen fra ORG-tabellen til den ansatte som har ID 280 i STAFF-tabellen:

```
SELECT DIVISION, LOCATION
   FROM ORG
   WHERE DEPTNUMB = (SELECT DEPT
         FROM STAFF
                        WHERE ID = 280)
```

Når en setning behandles, undersøker DB2 først resultatet av delspørringen. Resultatet fra dette eksempelets delspørring er 66, siden den ansatte med ID 280 er i avdeling 66. Det endelige resultatet blir deretter tatt fra raden til ORG-tabellen der DEPTNUMB-kolonnen har verdien 66. Det endelige resultatet er:

```
DIVISION   LOCATION
---------- -------------
Western    San Francisco
```

Når du bruker en delspørring, vurderer databasesystemet den og erstatter resultatverdien rett inn i WHERE-leddet.

Delspørringer blir grundigere omtalt i “Korrelasjonsdelspørringer” på side 39.

## Bruke funksjoner

Denne delen gir deg en kort introduksjon til funksjoner som blir brukt i eksemplene i boken. En *databasefunksjon* er et forhold mellom et sett med inndataverdier og en resultatverdi.

En funksjon kan enten være *innebygd* eller *brukerdefinert*. DB2 Universal Database leverer mange innebygde og forhåndsinstallerte brukerdefinerte funksjoner.

Du kan finne de innebygde funksjonene i SYSIBM-skjemaet og de forhåndsinstallerte brukerdefinerte funksjonene i SYSFUN-skjemaet. SYSIBM og SYSFUN er reserverte skjemaer.

De innebygde og brukerdefinerte funksjonene kommer aldri til å oppfylle alle brukerkravene. Det kan derfor være nødvendig for applikasjonsutviklere å lage sitt eget sett med funksjoner, spesifikt rettet på applikasjonene. Brukerdefinerte funksjoner gjør dette mulig og utvider utstrekningen til DB2 Universal Database slik at det inkluderer tilpassede forretningsfunksjoner eller vitenskapelige funksjoner. Dette blir grundigere omtalt i “Brukerdefinerte funksjoner” på side 66.

### Kolonnefunksjoner

*Kolonnefunksjoner* virker på et sett med verdier i en kolonne for å utlede en enkelt resultatverdi. Nedenfor finner du noen få eksempler på kolonnefunksjoner. Hvis du ønsker en fullstendig liste, kan du se i *SQL Reference*.

**AVG** Returnerer summen av verdiene i et sett, delt på antall verdier i settet.

**COUNT** Returnerer antall rader eller verdier i et sett med rader eller verdier

**MAX** Returnerer den største verdien i et sett med verdier

**MIN** Returnerer den minste verdien i et sett med verdier

Setningen nedenfor velger den største lønnen fra STAFF-tabellen:

```
SELECT MAX(SALARY)
         FROM STAFF
```

Denne setningen returnerer verdien 22959,20 fra STAFF-eksempeltabellen.

Det neste eksempelet velger navnene og lønningene til ansatte med inntekt over gjennomsnittet, men som har vært ansatt i firmaet mindre enn gjennomsnittlig antall år.

```
SELECT NAME, SALARY
         FROM STAFF
   WHERE SALARY > (SELECT AVG(SALARY) FROM STAFF)
   AND YEARS < (SELECT AVG(YEARS) FROM STAFF)
```

Denne setningen gir dette resultatet:

```
 NAME      SALARY
--------- ---------
Marenghi   17506.75
Daniels    19260.25
Gonzales   16858.20
```

I eksempelet ovenfor, i WHERE-leddet, er kolonnefunksjonen oppgitt i en *delspørring* i motsetning til å bli direkte implementert (for eksempel: WHERE SALARY > AVG(SALARY)). Kolonnefunksjoner kan ikke oppgis i WHERE-leddet. Dette er på grunn av rekkefølgen på operasjonene. Se på WHERE-leddet som et ledd som blir evaluert før SELECT-leddet. Derfor har ikke kolonnefunksjonen tilgang til settet med verdier når WHERE-leddet blir evaluert. Dette settet med verdier blir valgt på et senere tidspunkt av SELECT-leddet.

Du kan bruke DISTINCT-elementet som en del av argumentet til en kolonnefunksjon til å eliminere like verdier før en funksjon blir tatt i bruk. COUNT(DISTINCT WORKDEPT) regner derfor ut antall forskjellige avdelinger.

## Skalarfunksjoner

En *skalarfunksjon* utfører en operasjon på en enkeltverdi for å returnere en enkeltverdi til. Nedenfor finner du noen få eksempler på skalarfunksjoner i DB2 Universal Database.

**ABS** Returner absoluttverdien til et tall

**HEX** Returnerer den heksadesimale representasjonen av en verdi

**LENGTH** Returnerer antall byte i et argument (for en grafisk streng returnerer den antall dobbeltbytetegn.)

**YEAR** Trekk ut årsdelen av en datetime-verdi

Du finner en detaljert liste over og beskrivelse av skalarfunksjoner i *SQL Reference*.

Setningen nedenfor returnerer avdelingsnavnene fra ORG-tabellen sammen med lengden på hvert av disse navnene:

```
SELECT DEPTNAME, LENGTH(DEPTNAME)
   FROM ORG
```

Denne setningen gir dette resultatet:

```
DEPTNAME       2
-------------- -----------
Head Office             11
New England             11
Mid Atlantic            12
South Atlantic          14
Great Lakes             11
Plains                   6
Pacific                  7
Mountain                 8
```

Merk: Siden AS-leddet ikke ble brukt til å gi LENGTH(DEPTNAME) et forståelig navn, blir det vist et systemgenerert tall i den andre kolonnen.

### Tabellfunksjoner

*Tabellfunksjoner* returnerer kolonner i en tabell som likner på tabellen som ble opprettet av en enkel CREATE TABLE-setning.

En tabellfunksjon kan bare brukes i FROM-leddet til en SQL-setning.

Den eneste tabellfunksjonen som støttes i DB2 Universal Database er SQLCACHE_SNAPSHOT.

**SQLCACHE_SNAPSHOT**
Returnerer resultatet av et snapshot av DB2-hurtigbufferen for den dynamiske SQL-setningen som en tabell.

## Gruppering

DB2 Universal Database har muligheten til å analysere data basert på bestemte kolonner i en tabell.

Du kan organisere rader etter en grupperingsstruktur definert i et GROUP BY-ledd. I den enkleste formen er en *gruppe* et sett med rader som hver har identiske verdier i ″GROUP BY″-kolonnene. Kolonnenavnene i SELECT-leddet må enten være en grupperingskolonne eller en kolonnefunksjon. Kolonnefunksjonene returnerer en verdi for hver gruppe som er definert av GROUP BY-leddet. Hver gruppe representeres av en enkelt rad i resultatsettet. Eksempelet nedenfor gir et resultat som viser maksimal lønn for hvert avdelingsnummer:

```
SELECT DEPT, MAX(SALARY) AS MAXIMUM
         FROM STAFF
   GROUP BY DEPT
```

Denne setningen gir dette resultatet:

```
DEPT   MAXIMUM
------ ---------
    10  22959.20
    15  20659.80
    20  18357.50
    38  18006.00
    42  18352.80
    51  21150.00
    66  21000.00
    84  19818.00
```

Vær oppmerksom på at MAX(SALARY) beregnes for hver avdeling, en gruppe som er definert av GROUP BY-leddet, ikke hele firmaet.

## Bruke et WHERE-ledd med et GROUP BY-ledd

En grupperingsspørring kan ha et standard WHERE-ledd som eliminerer ikke-kvalifiserende rader før gruppene er laget og kolonnefunksjonene er regnet ut. Du må spesifisere WHERE-leddet *før* GROUP BY-leddet. Eksempel:

```
SELECT WORKDEPT, EDLEVEL, MAX(SALARY) AS MAXIMUM
   FROM EMPLOYEE
   WHERE HIREDATE > '1979-01-01'
   GROUP BY WORKDEPT, EDLEVEL
   ORDER BY WORKDEPT, EDLEVEL
```

Resultatet er:

```
WORKDEPT EDLEVEL MAXIMUM
-------- ------- -----------
D11           17    18270.00
D21           15    27380.00
D21           16    36170.00
D21           17    28760.00
E11           12    15340.00
E21           14    26150.00
```

Vær oppmerksom på at hvert kolonnenavn som er spesifisert i SELECT-setningen, også er nevnt i GROUP BY-leddet. Hvis du ikke oppgir

kolonnenavnene på begge steder, får du en feil. GROUP BY-leddet returnerer en rad for hver entydige kombinasjon av WORKDEPT og EDLEVEL.

### Bruke HAVING-leddet etter GROUP BY-leddet

Du kan ta i bruk en kvalifiserende betingelse på grupper slik at DB2 bare returnerer et result for gruppene som oppfyller betingelsen. Dette gjør du ved å inkludere et HAVING-ledd *etter* GROUP BY-leddet. Et HAVING-ledd kan inneholde ett eller flere predikater som er knyttet sammen med AND og OR. Hvert predikat sammenlikner en egenskap i gruppen (for eksempel AVG(SALARY)) med en av følgende:

- en annen egenskap i gruppen

  for eksempel:

  ```
  HAVING AVG(SALARY) > 2 * MIN(SALARY)
  ```

- en konstant

  For eksempel:

  ```
  HAVING AVG(SALARY) > 20000
  ```

Spørringen nedenfor finner for eksempel den største og minste lønnen for avdelinger med mer enn fire ansatte:

```
SELECT WORKDEPT, MAX(SALARY) AS MAXIMUM, MIN(SALARY) AS MINIMUM
   FROM EMPLOYEE
   GROUP BY WORKDEPT
   HAVING COUNT(*) > 4
   ORDER BY WORKDEPT
```

Denne setningen gir dette resultatet:

```
WORKDEPT MAXIMUM     MINIMUM
-------- ----------- -----------
D11         32250.00    18270.00
D21         36170.00    17250.00
E11         29750.00    15340.00
```

Det er mulig (men uvanlig) for en spørring å ha et HAVING-ledd, men ikke noe GROUP BY-ledd. I dette tilfellet behandler DB2 hele tabellen som en gruppe. Siden tabellen behandles som en enkelt gruppe kan du ikke ha mer enn en resultatrad. Hvis HAVING-betingelsen er sann for tabellen som en helhet, blir det valgte resultatet returnert (som bare kan bestå av kolonnefunksjoner), ellers blir ingen rader returnert.

# Kapittel 5. Uttrykk og delspørringer

DB2 sørger for fleksibilitet når du skal formulere spørringer. Dette kapittelet beskriver noen få av de viktige metodene som er tilgjengelige for å formulere sammensatte spørringer.

Dette kapittelet gir en omfattende beskrivelse av punktene nedenfor:

- Skalarfunksjon med full SELECT-setninger
- Konvertere datatyper
- CASE-uttrykk
- Tabelluttrykk
- Korrelasjonsnavn

## Skalarfunksjon med full SELECT-setninger

En full SELECT-setning er en type spørring som kan brukes i SQL-setninger. En skalarfunksjon med full SELECT er en full SELECT-setning som returnerer en rad som bare inneholder en verdi. Skalarfunksjoner med full SELECT er nyttige når du skal hente dataverdier fra databasen som skal brukes i et uttrykk.

- Det neste eksempelet viser navnene til ansatte som har en høyere lønning enn gjennomsnittslønnen for alle ansatte. Skalarfunksjonen med full SELECT i spørringen er SELECT-setningen i parentes.

```
SELECT LASTNAME, FIRSTNME
   FROM EMPLOYEE
   WHERE  SALARY > (SELECT AVG(SALARY)
                      FROM EMPLOYEE)
```

- Dette eksempelet finner gjennomsnittslønnen til de ansatte i tabellen STAFF og gjennomsnittslønnen til de ansatte i tabellen EMPLOYEE.

```
SELECT AVG(SALARY) AS "Average_Employee",
   (SELECT AVG(SALARY) AS "Average_Staff" FROM STAFF)
      FROM EMPLOYEE
```

## Konvertere datatyper

Det kan være at du noen ganger har behov for å konvertere verdier fra en datatype til en annen, for eksempel fra en numerisk verdi til en tegnstreng. Hvis du skal konvertere en verdi til en annen type, bruker du CAST-spesifikasjonen.

En annen mulig bruk av en konverteringsspesifikasjon er å kutte av en veldig lang tegnstreng. I tabellen EMP_RESUME er kolonnen RESUME CLOB(5K). Det kan være at du bare vil vise de første 370 tegnene som inneholder personopplysningene om søkeren. Når du vil vise de første 370 tegnene av ASCII-formatet til personopplysningene fra tabellen EMP_RESUME, gir du denne spørringen:

```
SELECT EMPNO, CAST(RESUME AS VARCHAR(370))
   FROM EMP_RESUME
   WHERE RESUME_FORMAT = 'ascii'
```

Det blir gitt en advarsel som informerer om at verdier som er lengre enn 370 tegn, blir avkuttet.

Du kan konvertere NULL-verdier til andre datatyper som er mer anvendelige for manipulering i en spørring. “Felles tabelluttrykk” på side 36 er et eksempel på bruk av konvertering til dette formålet.

## CASE-uttrykk

Du kan bruke CASE-uttrykk i SQL-setninger for å manipulere datarepresentasjonen av en tabell. Dette gir et kraftig betinget uttrykkspotensiale som likner på CASE-setninger i noen programmeringsspråk.

- For å endre avdelingsnumrene fra kolonnen DEPTNAME i tabellen ORG til ord som er lette å forstå, oppgir du følgende spørring:

```
SELECT DEPTNAME,
   CASE DEPTNUMB
     WHEN 10 THEN 'Marketing'
     WHEN 15 THEN 'Research'
     WHEN 20 THEN 'Development'
     WHEN 38 THEN 'Accounting'
     ELSE 'Sales'
   END  AS FUNCTION
   FROM ORG
```

Resultatet er:

```
DEPTNAME       FUNCTION
-------------- -----------
Head Office    Marketing
New England    Research
Mid Atlantic   Development
South Atlantic Accounting
Great Lakes    Sales
Plains         Sales
Pacific        Sales
Mountain       Sales
```

- Du kan bruke CASE-uttrykk som beskyttelse mot unntak, for eksempel deling på null. I eksempelet nedenfor forhindrer setningsbetingelsen en feil ved å unngå divisjonsoperasjonen hvis den ansatte ikke har en bonus eller kommisjonsbetaling:

```
SELECT LASTNAME, WORKDEPT FROM EMPLOYEE
   WHERE(CASE
            WHEN BONUS+COMM=0 THEN NULL
            ELSE SALARY/(BONUS+COMM)
         END ) > 10
```

- Du kan bruke et CASE-uttrykk til å lage et forhold mellom summen av en undergruppe av verdier i en kolonne og summen av alle verdiene i den kolonnen. Dette forholdstallet kan defineres i en enkelt setning som bruker et CASE-uttrykk. Dette uttrykket krever bare en enkelt sending gjennom dataene. Uten et CASE-uttrykk er det nødvendig med minst to sendinger for å utføre den samme beregningen.

  Eksempelet nedenfor beregner forholdet mellom summen av lønningene i avdeling 20 og summen av alle lønningene ved hjelp av et CASE-uttrykk:

```
SELECT CAST(CAST (SUM(CASE
                        WHEN DEPT = 20 THEN SALARY
                        ELSE 0
                      END) AS DECIMAL(7,2))/
                   SUM(SALARY) AS DECIMAL (3,2))
         FROM STAFF
```

  Resultatet er 0,11. Legg merke til at konverteringsfunksjonene sørger for at presisjonen til resultatet blir bevart.

- Du kan bruke et CASE-uttrykk til å evaluere en enkelt funksjon i stedet for å anrope selve funksjonen, som ville medført ekstra behandling. For eksempel:

```
CASE
   WHEN X<0 THEN -1
   WHEN X=0 THEN 0
   WHEN X>0 THEN 1
END
```

  Dette uttrykket har det samme resultatet som den brukerdefinerte SIGN-funksjonen i SYSFUN-skjemaet.

## Tabelluttrykk

Hvis du bare trenger en definisjon av et utsnitt for en enkelt spørring, kan du bruke et *tabelluttrykk*.

Tabelluttrykk er midlertidige og er bare gyldige i levetiden til SQL-setningen. De kan ikke deles som utsnitt, men de gir større fleksibilitet enn utsnitt.

Denne delen beskriver hvordan du bruker felles tabelluttrykk og nestede tabelluttrykk i spørringer.

## Nestede tabelluttrykk

Et nestet tabelluttrykk er et midlertidig utsnitt der definisjonen er *nestet* (definert direkte) i FROM-leddet til hovedspørringen.

Spørringen nedenfor bruker et nestet tabelluttrykk for å finne gjennomsnittlig samlet lønn, utdanningsnivå og ansettelsesår for de med et utdanningsnivå som er høyere enn 16:

```
SELECT EDLEVEL, HIREYEAR, DECIMAL(AVG(TOTAL_PAY),7,2)
   FROM (SELECT EDLEVEL, YEAR(HIREDATE) AS HIREYEAR,
                SALARY+BONUS+COMM AS TOTAL_PAY
            FROM EMPLOYEE
            WHERE EDLEVEL > 16)  AS PAY_LEVEL
   GROUP BY EDLEVEL, HIREYEAR
   ORDER BY EDLEVEL, HIREYEAR
```

Resultatet er:

```
EDLEVEL HIREYEAR    3
------- ----------- ---------
     17        1967  28850.00
     17        1973  23547.00
     17        1977  24430.00
     17        1979  25896.50
     18        1965  57970.00
     18        1968  32827.00
     18        1973  45350.00
     18        1976  31294.00
     19        1958  51120.00
     20        1975  42110.00
```

Denne spørringen bruker et nestet tabelluttrykk for først å trekke ut ansettelsesåret fra kolonnen HIREDATE, slik at det deretter kan brukes i GROUP BY-leddet. Det kan være at du ikke vil opprette dette som et utsnitt, hvis du har tenkt til å utføre liknende spørringer ved hjelp av forskjellige verdier for EDLEVEL.

Den innebygde skalarfunksjonen DECIMAL blir brukt i dette eksempelet. DECIMAL returnerer en desimalrepresentasjon av tall eller en tegnstreng. Du finner flere opplysninger om funksjonene i *SQL Reference*.

## Felles tabelluttrykk

Et *felles tabelluttrykk* er et tabelluttrykk som du oppretter for bruk i en sammensatt spørring. Definer og navngi det på begynnelsen av spørringen ved hjelp av et WITH-ledd. Gjentatte referanser til et felles tabelluttrykk

bruker det samme resultatsettet. Hvis du brukte nestede tabelluttrykk eller utsnitt, ville resultatsettet blitt generert hver gang, muligens med forskjellige resultater.

Eksempelet nedenfor viser alle menneskene i firmaet som har et utdanningsnivå som er høyere enn 16, som tjener mindre gjennomsnittlig enn de menneskene som ble ansatt det samme året og som har samme utdanning. Delene av spørringen blir beskrevet mer detaljert etter spørringen.

**1**

```
WITH
      PAYLEVEL AS
         (SELECT EMPNO, EDLEVEL, YEAR(HIREDATE) AS HIREYEAR,
                 SALARY+BONUS+COMM AS TOTAL_PAY
             FROM EMPLOYEE
             WHERE EDLEVEL > 16),
```

**2**

```
      PAYBYED (EDUC_LEVEL, YEAR_OF_HIRE, AVG_TOTAL_PAY) AS
         (SELECT EDLEVEL, HIREYEAR, AVG(TOTAL_PAY)
             FROM PAYLEVEL
             GROUP BY EDLEVEL, HIREYEAR)
```

**3**

```
SELECT EMPNO, EDLEVEL, YEAR_OF_HIRE, TOTAL_PAY, DECIMAL(AVG_TOTAL_PAY,7,2)
   FROM PAYLEVEL, PAYBYED
   WHERE EDLEVEL = EDUC_LEVEL
     AND HIREYEAR = YEAR_OF_HIRE
     AND TOTAL_PAY < AVG_TOTAL_PAY
```

**1** Dette er et felles tabelluttrykk med navnet PAYLEVEL. Denne resultattabellen inkluderer et funksjonærnummer, ansettelsesåret, samlet lønn for den ansatte og hans eller hennes utdanningsnivå. Det er bare rader for ansatte med et utdanningsnivå som er høyere enn 16, som er inkludert.

**2** Dette er et felles tabelluttrykk med navnet PAYBYED (eller PAY BY EDucation). Det bruker tabellen PAYLEVEL som ble opprettet i det forrige felles tabelluttrykket til å finne ut utdanningsnivået, ansettelsesåret og gjennomsnittlig lønn til ansatte i hvert utdanningsnivå og som ble ansatt det samme året. Kolonnene som blir returnert av denne tabellen, har fått forskjellige navn (for eksempel EDUC_LEVEL) fra kolonnenavnene som brukes i SELECT-listen. Dette gir et resultatsett kalt PAYBYED som er det samme som resultatet vi laget i eksempelet med det nestede tabelluttrykket.

**3** Til slutt kommer vi til spørringen som gir det ønskede resultatet. De to tabellene (PAYLEVEL, PAYBYED) blir kombinert for å finne individene som har en samlet lønn som er mindre enn

gjennomsnittslønnen for folk som er ansatt det samme året. Vær oppmerksom på at PAYBYED er basert på PAYLEVEL. PAYLEVEL blir brukt to ganger i den fullstendige setningen. Begge gangene blir det samme settet med rader brukt ved evaluering av spørringen.

Det endelige resultatet er:

```
EMPNO  EDLEVEL YEAR_OF_HIRE TOTAL_PAY     5
------ ------- ------------ ------------- ---------
000210      17         1979      20132.00  25896.50
```

## Korrelasjonsnavn

Et *korrelasjonsnavn* er en identifikator for skille ut et objekt som er brukt flere ganger. Et korrelasjonsnavn kan defineres i FROM-leddet i en spørring og i det første leddet i en UPDATE- eller DELETE-setning. Det kan knyttes til en tabell, et utsnitt eller et nestet tabelluttrykk, men bare i den sammenhengen som det er definert.

Leddet FROM STAFF S, ORG O oppretter for eksempel S og O som korrelasjonsnavn for henholdsvis STAFF og ORG.

```
SELECT NAME, DEPTNAME
   FROM STAFF S, ORG O
   WHERE O.MANAGER = S.ID
```

Når du har definert et korrelasjonsnavn, kan du *bare* bruke korrelasjonsnavnet til å kvalifisere objektet. I eksempelet ovenfor for eksempel, ville setningen ha blitt mislykket hvis ORG.MANAGER=STAFF.ID hadde blitt oppgitt.

Du kan også bruke et korrelasjonsnavn som et kortere navn for å referere til et databaseobjekt. Det er lettere å skrive S enn å skrive STAFF.

Ved å bruke korrelasjonsnavn kan du lage **duplikater** av et objekt. Dette er nyttig når du trenger å sammenlikne poster i en tabell med seg selv. I eksempelet nedenfor blir tabellen EMPLOYEE sammenliknet med en annen forekomst av seg selv for å finne avdelingssjefene for alle de ansatte. Den viser navnene til de ansatte som ikke er designere, navnet på avdelingssjefen og avdelingsnummeret.

```
SELECT E2.FIRSTNME, E2.LASTNAME, E2.JOB, E1.FIRSTNME AS MGR_FIRSTNAME,
                    E1.LASTNAME AS MGR_LASTNAME, E1.WORKDEPT
  FROM EMPLOYEE E1, EMPLOYEE E2
  WHERE E1.WORKDEPT = E2.WORKDEPT
    AND E1.JOB = 'MANAGER'
    AND E2.JOB <> 'MANAGER'
    AND E2.JOB <> 'DESIGNER'
```

Denne setningen gir dette resultatet:

```
FIRSTNME     LASTNAME   JOB      MGR_FIRSTNAME MGR_LASTNAME    WORKDEPT
------------ ---------- -------- ------------- --------------- --------
DOLORES      QUINTANA   ANALYST  SALLY         KWAN            C01
HEATHER      NICHOLLS   ANALYST  SALLY         KWAN            C01
JAMES        JEFFERSON  CLERK    EVA           PULASKI         D21
SALVATORE    MARINO     CLERK    EVA           PULASKI         D21
DANIEL       SMITH      CLERK    EVA           PULASKI         D21
SYBIL        JOHNSON    CLERK    EVA           PULASKI         D21
MARIA        PEREZ      CLERK    EVA           PULASKI         D21
ETHEL        SCHNEIDER  OPERATOR EILEEN        HENDERSON       E11
JOHN         PARKER     OPERATOR EILEEN        HENDERSON       E11
PHILIP       SMITH      OPERATOR EILEEN        HENDERSON       E11
MAUDE        SETRIGHT   OPERATOR EILEEN        HENDERSON       E11
RAMLAL       MEHTA      FIELDREP THEODORE      SPENSER         E21
WING         LEE        FIELDREP THEODORE      SPENSER         E21
JASON        GOUNOT     FIELDREP THEODORE      SPENSER         E21
```

## Korrelasjonsdelspørringer

En delspørring som kan referere til alle tidligere nevnte tabeller, kalles en *korrelasjonsdelspørring*. Vi sier også at delspørringen har en *korrelasjonsreferanse* til en tabell i hovedspørringen.

Det neste eksempelet bruker en ikke-korrelasjonsdelspørring til å vise funksjonærnummeret og navnet til de ansatte i avdeling 'A00' som har en lønn som er høyere enn den gjennomsnittlige lønnen i avdelingen:

```
SELECT EMPNO, LASTNAME
   FROM EMPLOYEE
   WHERE WORKDEPT = 'A00'
     AND SALARY > (SELECT AVG(SALARY)
                     FROM EMPLOYEE
                     WHERE WORKDEPT = 'A00')
```

Denne setningen gir dette resultatet:

```
EMPNO  LASTNAME
------ ---------------
000010 HAAS
000110 LUCCHESSI
```

Hvis du vil vite den gjennomsnittlige lønnen for hver avdeling, må delspørringen evalueres en gang for hver avdeling. Du kan gjøre dette ved hjelp av korrelasjonsmuligheten til SQL som gjør det mulig å skrive en delspørring som utføres gjentatte ganger, en gang for hver rad i tabellen som er identifisert i den eksterne spørringen.

Eksempelet nedenfor bruker en korrelasjonsdelspørring til å vise alle de ansatte som har en lønn som er høyere enn gjennomsnittslønnen til avdelingen:

```
SELECT E1.EMPNO, E1.LASTNAME, E1.WORKDEPT
   FROM EMPLOYEE E1
   WHERE SALARY > (SELECT AVG(SALARY)
                     FROM EMPLOYEE E2
                     WHERE E2.WORKDEPT = E1.WORKDEPT)
   ORDER BY E1.WORKDEPT
```

I denne spørringen blir delspørringen evaluert en gang for hver avdeling: Resultatet er:

```
EMPNO  LASTNAME        WORKDEPT
------ --------------- --------
000010 HAAS            A00
000110 LUCCHESSI       A00
000030 KWAN            C01
000060 STERN           D11
000150 ADAMSON         D11
000170 YOSHIMURA       D11
000200 BROWN           D11
000220 LUTZ            D11
000070 PULASKI         D21
000240 MARINO          D21
000270 PEREZ           D21
000090 HENDERSON       E11
000280 SCHNEIDER       E11
000100 SPENSER         E21
000330 LEE             E21
000340 GOUNOT          E21
```

Når du skal skrive en spørring med en korrelasjonsdelspørring, bruker du det samme grunnleggende formatet som til en vanlig ekstern spørring med en delspørring. I FROM-leddet til den eksterne spørringen derimot, plasserer du et korrelasjonsnavn rett etter tabellnavnet. Delspørringen kan da inneholde kolonnereferanser som er kvalifisert av korrelasjonsnavnet. Hvis for eksempel E1 er et korrelasjonsnavn, er E1.WORKDEPT verdien WORKDEPT i den gjeldende raden av tabellen i den eksterne spørringen. Delspørringen skal reevalueres for hver rad i tabellen i den eksterne spørringen.

Ved å bruke en korrelasjonsdelspørring, lar du systemet gjøre arbeidet for deg og redusere mengden med kode du må skrive i applikasjonen.

Ukvalifiserte korrelasjonsreferanser er tillatt i DB2. Tabellen EMPLOYEE har for eksempel en kolonne kalt LASTNAME, men tabellen SALES har en kolonne kalt SALES_PERSON og ingen kolonne kalt LASTNAME.

```
SELECT LASTNAME, FIRSTNME, COMM
   FROM EMPLOYEE
   WHERE 3 > (SELECT AVG(SALES)
                FROM SALES
                WHERE LASTNAME = SALES_PERSON)
```

I dette eksempelet kontrollerer systemet om det finner en LASTNAME-kolonne i det innerste FROM-leddet. Når det ikke finner en, kontrollerer systemet det nest innerste FROM-leddet (som i dette tilfellet er det eksterne FROM-leddet). Selv om det ikke alltid er nødvendig, så anbefales kvalifiserte korrelasjonsreferanser for å gjøre det lettere å lese spørringen og forsikre deg om at du får det resultatet du ønsker.

## Implementere en korrelasjonsdelspørring

Når vil du bruke en korrelasjonsdelspørring? Bruken av en kolonnefunksjon er ofte et godt tegn.

Du vil for eksempel ha frem en liste over ansatte som har et utdanningsnivå som er høyere enn gjennomsnittet for avdelingen.

Du må først bestemme postene på SELECT-listen. Problemet sier "Vis en liste over de ansatte". Dette antyder at LASTNAME fra tabellen EMPLOYEE er nok til å entydig identifisere de ansatte. Problemet oppgir også utdanningsnivået (EDLEVEL) og de ansattes avdelinger (WORKDEPT) som betingelser. Selv om problemet ikke ber eksplisitt om at kolonnene skal vises, illustreres løsningen bedre hvis du tar dem med i SELECT-listen. En del av spørringen kan nå lages:

```
SELECT LASTNAME, WORKDEPT, EDLEVEL
   FROM EMPLOYEE
```

Deretter trenger du en søkebetingelse (WHERE-ledd). Problemsetningen sier "...som har et utdanningsnivå som er høyere enn gjennomsnittet for avdelingen til den ansatte". Dette betyr at for hver ansatt i tabellen, må det gjennomsnittlige utdanningsnivået for den ansattes avdeling regnes ut. Denne setningen passer til beskrivelsen av en korrelasjonsdelspørring. Det blir beregnet en ukjent egenskap (gjennomsnittlig utdanningsnivå for avdelingen til den ansatte som det gjelder) for hver rad. Det kreves et korrelasjonsnavn for tabellen EMPLOYEE:

```
SELECT LASTNAME, WORKDEPT, EDLEVEL
   FROM EMPLOYEE E1
```

Delspørringen som er nødvendig er enkel. Den beregner det gjennomsnittlige utdanningsnivået for hver avdeling. Den fullstendige SQL-setningen er:

```
SELECT LASTNAME, WORKDEPT, EDLEVEL
   FROM EMPLOYEE E1
   WHERE EDLEVEL > (SELECT AVG(EDLEVEL)
                       FROM EMPLOYEE  E2
                       WHERE E2.WORKDEPT = E1.WORKDEPT)
```

Resultatet er:

```
LASTNAME        WORKDEPT EDLEVEL
--------------- -------- -------
HAAS            A00           18
```

```
KWAN            C01           20
PULASKI         D21           16
HENDERSON       E11           16
LUCCHESSI       A00           19
PIANKA          D11           17
SCOUTTEN        D11           17
JONES           D11           17
LUTZ            D11           18
MARINO          D21           17
JOHNSON         D21           16
SCHNEIDER       E11           17
MEHTA           E21           16
GOUNOT          E21           16
```

Anta at du fører opp avdelingsnavnet på en liste i stedet for den ansattes avdelingsnummer. Opplysningene du trenger (DEPTNAME), er i en separat tabell (DEPARTMENT). Den eksterne spørringen som definerer en korrelasjonsvariabel, kan også være en kombispørring (du finner detaljerte opplysninger i "Velge data fra mer enn en tabell" på side 26).

Når du bruker kombineringer i en ekstern spørring, setter du opp en liste over tabellene som skal kombineres i FROM-leddet, og setter korrelasjonsnavnet ved siden av det riktige tabellnavnet.

Når du skal endre spørrelisten slik at den viser avdelingens navn i stedet for nummer, erstatter du WORKDEPT med DEPTNAME i SELECT-listen. FROM-leddet må nå også inkludere tabellen DEPARTMENT, og WHERE-leddet må uttrykke den riktige kombinasjonsbetingelsen.

Dette er den endrede spørringen:

```
SELECT LASTNAME, DEPTNAME, EDLEVEL
   FROM EMPLOYEE E1, DEPARTMENT
   WHERE E1.WORKDEPT = DEPARTMENT.DEPTNO
   AND EDLEVEL > (SELECT AVG(EDLEVEL)
                  FROM EMPLOYEE E2
                  WHERE E2.WORKDEPT = E1.WORKDEPT)
```

Denne setningen gir dette resultatet:

```
LASTNAME        DEPTNAME                     EDLEVEL
--------------- ---------------------------- -------
HAAS            SPIFFY COMPUTER SERVICE DIV.      18
LUCCHESSI       SPIFFY COMPUTER SERVICE DIV.      19
KWAN            INFORMATION CENTER                20
PIANKA          MANUFACTURING SYSTEMS             17
SCOUTTEN        MANUFACTURING SYSTEMS             17
JONES           MANUFACTURING SYSTEMS             17
LUTZ            MANUFACTURING SYSTEMS             18
PULASKI         ADMINISTRATION SYSTEMS            16
MARINO          ADMINISTRATION SYSTEMS            17
JOHNSON         ADMINISTRATION SYSTEMS            16
HENDERSON       OPERATIONS                        16
```

```
SCHNEIDER       OPERATIONS                      17
MEHTA           SOFTWARE SUPPORT                16
GOUNOT          SOFTWARE SUPPORT                16
```

Eksemplene ovenfor viser at korrelasjonsnavnet som er brukt i en delspørring, må defineres i FROM-leddet til en spørring som inneholder korrelasjonsdelspørringen. Dette kan imidlertid inneholde flere nivåer med nesting.

Anta at noen avdelinger bare har noen få ansatte og derfor kan det gjennomsnittlige utdanningsnivået være villedende. Kanskje du bestemmer at det må være minst fem ansatte i en avdeling for at det gjennomsnittlige utdanningsnivået skal være et meningsfylt tall å sammnenlikne en ansatt med. Nå må det settes opp en liste over de ansatte som har et utdanningsnivå som er høyere enn gjennomsnittet for den ansattes avdeling, og bare ta med avdelinger med minst fem ansatte.

Problemet forutsetter annen delspørring, fordi for hver ansatt i den eksterne spørringen må samlet antall ansatte i den personens avdeling være regnet med:

```
SELECT COUNT(*)
   FROM EMPLOYEE E3
   WHERE E3.WORKDEPT = E1.WORKDEPT
```

Det skal bare beregnes et gjennomsnitt hvis antallet er større enn eller lik 5:

```
SELECT AVG(EDLEVEL)
   FROM EMPLOYEE E2
   WHERE E2.WORKDEPT = E1.WORKDEPT
   AND 5 <= (SELECT COUNT(*)
                FROM EMPLOYEE  E3
                WHERE E3.WORKDEPT = E1.WORKDEPT)
```

Bare de ansatte som har et utdanningsnivå som er høyere enn gjennomsnittet for den avdelingen, er inkludert:

```
SELECT LASTNAME, DEPTNAME, EDLEVEL
   FROM EMPLOYEE E1, DEPARTMENT
   WHERE E1.WORKDEPT = DEPARTMENT.DEPTNO
   AND EDLEVEL >
   (SELECT AVG(EDLEVEL)
       FROM EMPLOYEE E2
       WHERE E2.WORKDEPT = E1.WORKDEPT
       AND 5 <=
       (SELECT COUNT(*)
           FROM EMPLOYEE E3
           WHERE E3.WORKDEPT = E1.WORKDEPT))
```

Denne setningen gir dette resultatet:

```
LASTNAME        DEPTNAME                     EDLEVEL
--------------- ---------------------------- -------
PIANKA          MANUFACTURING SYSTEMS             17
SCOUTTEN        MANUFACTURING SYSTEMS             17
JONES           MANUFACTURING SYSTEMS             17
LUTZ            MANUFACTURING SYSTEMS             18
PULASKI         ADMINISTRATION SYSTEMS            16
MARINO          ADMINISTRATION SYSTEMS            17
JOHNSON         ADMINISTRATION SYSTEMS            16
HENDERSON       OPERATIONS                        16
SCHNEIDER       OPERATIONS                        17
```

# Kapittel 6. Bruke operatorer og predikater i spørringer

I DB2 Universal Database kan du kombinere spørringer med forskjellige *mengdeoperatorer* og lage sammensatte betingelsessetninger med *kvantifiserte predikater*.

Dette kapittelet forklarer hvordan du kan:

- kombinere forskjellige tabeller med UNION-, EXCEPT- og INTERSECT-mengdeoperatorer
- lage sammensatte betingelsessetninger for spørringer med kvantifiserte predikater. Basispredikater ble kort omtalt i "Velge rader" på side 20.

## Kombinere spørringer med mengdeoperatorer

UNION-, EXCEPT- og INTERSECT-mengdeoperatorene gjør det mulig å kombinere to eller flere eksterne spørringer i en enkelt spørring. Hver spørring som er tilkoblet med disse mengdeoperatorene, blir utført og de individuelle resultatene blir kombinert. Hver operator gir forskjellig resultat.

### UNION-operator

UNION-operatoren lager en resultattabell ved å kombinere to andre resultattabeller (for eksempel TABLE1 og TABLE2) og fjerne eventuelle like rader i tabellene. Når ALL blir brukt sammen med UNION (det vil si UNION ALL), blir ikke like rader fjernet. I begge tilfeller er hver rad av den utledede tabellen en rad fra enten TABLE1 eller TABLE2.

I eksempelet nedenfor på UNION-operatoren, returnerer spørringen navnene på alle personer som har en lønn som er høyere enn $21, 000, eller som har administrerende ansvar og har jobbet i mindre enn åtte år:

**1**

```
SELECT ID, NAME FROM STAFF WHERE SALARY > 21000
UNION
```

**2**

```
SELECT ID, NAME FROM STAFF WHERE JOB='Mgr' AND YEARS < 8
ORDER BY ID
```

Resultatene av de individuelle spørringene er:

**1**

```
ID     NAME
------ ---------
   140 Fraye
   160 Molinare
   260 Jones
```

**2**

```
ID     NAME
------ ---------
    10 Sanders
    30 Marenghi
   100 Plotz
   140 Fraye
   160 Molinare
   240 Daniels
```

Databasesystemet kombinerer resultatene av begge spørringene, fjerner duplikater og returnerer det endelige resultatet i stigende rekkefølge.

```
ID     NAME
------ ---------
    10 Sanders
    30 Marenghi
   100 Plotz
   140 Fraye
   160 Molinare
   240 Daniels
   260 Jones
```

Hvis du bruker ORDER BY-leddet i en spørring med en hvilken som helst mengdeoperator, må du skrive den etter den siste spørringen. Systemet tar i bruk rekkefølgen på det kombinerte svarsettet.

Hvis kolonnenavnene i de to tabellene er forskjellige, har ikke den kombinerte resultattabellen navn for de tilsvarende kolonnene. Kolonnene blir i stedet nummerert i den rekkefølgen de blir vist. Så hvis du vil at resultattabellen skal sorteres, må du spesifisere kolonnenummeret i ORDER BY-leddet.

## EXCEPT-operator

EXCEPT-operatoren utleder en resultattabell ved å inkludere alle rader som er i TABLE1, men ikke i TABLE2, og fjerne alle like rader. Når du bruker ALL med EXCEPT (EXCEPT ALL), blir ikke de like radene fjernet.

I eksempelet nedenfor på EXCEPT-operatoren, returnerer spørringen navnene på alle personer som tjener over $21,000, men som ikke er avdelingssjefer og som har vært der i mer enn åtte år.

```
SELECT ID, NAME FROM STAFF WHERE SALARY > 21000
EXCEPT
SELECT ID, NAME FROM STAFF WHERE JOB='Mgr' AND YEARS < 8
```

Resultatene av de individuelle spørringene blir vist i avsnittet om UNION. Setningen ovenfor gir dette resultatet:

```
ID     NAME
------ ---------
   260 Jones
```

## INTERSECT-operator

INTERSECT-operatoren utleder en resultattabell ved å inkludere alle rader som finnes i både TABLE1 og TABLE2, og fjerner alle like rader. Når du bruker ALL med INTERSECT (INTERSECT ALL), blir ikke de like radene fjernet.

I eksempelet nedenfor på INTERSECT-operatoren, returnerer spørringen navnet og IDen på ansatte som tjener mer enn $21,000, har administrerende ansvar og har jobbet i færre enn åtte år.

```
SELECT ID, NAME FROM STAFF WHERE SALARY > 21000
INTERSECT
SELECT ID, NAME FROM STAFF WHERE JOB='Mgr' AND YEARS < 8
```

Resultatet av de individuelle spørringene blir vist i avsnittet om UNION. Resultatet av de to spørringene med INTERSECT er:

```
ID     NAME
------ ---------
   140 Fraye
   160 Molinare
```

Når du bruker UNION-, EXCEPT- og INTERSECT-operatorene, må du huske på dette:

- Alle tilsvarende poster i SELECT-listene til spørringene for operatorene må være kompatible. Du finner flere opplysninger i tabellen for datatypekompatibilitet i *SQL Reference*.
- Når det brukes et ORDER BY-ledd, må det plasseres etter den siste spørringen med en mengdeoperator. Kolonnenavnet kan bare brukes i ORDER BY-leddet hvis kolonnenavnet er identisk med de tilsvarende postene i SELECT-listen til spørringene for hver operator.
- Operasjoner mellom kolonner som har samme datatype og samme lengde gir en kolonne med den typen og lengden. Les om regler for resultatdatatyper i *SQL Reference* for å finne resultatene for UNION-, EXCEPT- og INTERSECT-mengdeoperatorene.

## Predikater

Predikater gjør det mulig å lage betingelser slik at bare radene som oppfyller disse betingelsene, blir behandlet. Basispredikater omtales i “Velge rader” på side 20. IN, BETWEEN, LIKE, EXISTS og kvantifiserte predikater blir omtalt i dette avsnittet.

### Bruke IN-predikatet

Bruk IN-predikatet til å sammenlikne en verdi med flere andre verdier. For eksempel:

```
SELECT NAME
         FROM STAFF
   WHERE DEPT IN (20, 15)
```

Dette eksempelet tilsvarer:

```
SELECT NAME
         FROM STAFF
   WHERE DEPT = 20  OR DEPT = 15
```

Du kan bruke IN- og NOT IN-operatorer når en delspørring returnerer et sett med verdier. Spørringen nedenfor viser for eksempel etternavnene på ansatte som er ansvarlige for prosjektene MA2100 og OP2012:

```
SELECT LASTNAME
   FROM EMPLOYEE
   WHERE EMPNO IN
      (SELECT RESPEMP
          FROM PROJECT
          WHERE PROJNO = 'MA2100'
          OR PROJNO = 'OP2012')
```

Delspørringen blir evaluert en gang og listen som er resultatet, erstattes i den eksterne spørringen. Hvis delspørringen ovenfor for eksempel velger funksjonærnumrene 10 og 330, blir den eksterne spørringen evaluert som om dens WHERE-ledd var:

```
WHERE EMPNO IN (10, 330)
```

Listen over verdier som er returnert av delspørringen, kan inneholde null, en eller flere verdier.

### Bruke BETWEEN-predikatet

BETWEEN-predikatet sammenlikner en enkelt verdi med et inklusivt område med verdier (navngitt i BETWEEN-predikatet).

Eksempelet nedenfor finner navnene på ansatte som tjener mellom $10, 000 og $20, 000:

```
SELECT LASTNAME
   FROM EMPLOYEE
   WHERE SALARY BETWEEN 10000 AND 20000
```

Dette tilsvarer:

```
SELECT LASTNAME
   FROM EMPLOYEE
   WHERE SALARY >= 10000 AND SALARY <= 20000
```

Det neste eksempelet finner navnene til ansatte som tjener mindre enn $10, 000 eller mer enn $20, 000:

```
SELECT LASTNAME
   FROM EMPLOYEE
   WHERE SALARY NOT BETWEEN 10000 AND 20000
```

## Bruke LIKE-predikatet

Bruk LIKE-predikatet til å søke etter strenger som har bestemte mønstre. Mønsteret er oppgitt i prosenttegn og understreking.

- Understrekingstegnet (_) representerer ett enkelt tegn.
- Prosenttegnet (%) representerer en streng med null eller flere tegn.
- Eventuelle andre tegn representerer seg selv.

Eksempelet nedenfor velger navn på ansatte som er syv bokstaver lange og begynner med bokstaven S:

```
SELECT NAME
         FROM STAFF
   WHERE NAME LIKE 'S _ _ _ _ _ _'
```

Det neste eksempelet velger navn på ansatte som ikke begynner med bokstaven S:

```
SELECT NAME
         FROM STAFF
   WHERE NAME  NOT LIKE 'S%'
```

## Bruke EXISTS-predikatet

Du kan bruke en delspørring til å lete etter *eksistensen* til en rad som oppfyller en eller annen betingelse. I dette tilfellet er delspørringen koblet til den eksterne spørringen med predikatet EXISTS eller NOT EXISTS.

Når du kobler en delspørring til en ekstern spørring med et EXISTS-predikat, returnerer ikke delspørringen en verdi. I stedet er EXISTS-predikatet sant hvis svarsettet til delspørringen inneholder en eller flere rader og usant hvis det ikke inneholder noen rader.

EXISTS-predikatet blir ofte brukt med korrelasjonsdelspørringer. Eksempelet nedenfor viser avdelingene som ikke har noen poster i PROJECT-tabellen:

```
SELECT DEPTNO, DEPTNAME
   FROM DEPARTMENT X
   WHERE NOT EXISTS
```

```
            (SELECT *
                FROM PROJECT
                WHERE DEPTNO = X.DEPTNO)
    ORDER BY DEPTNO
```

Du kan koble EXISTS- og NOT EXISTS-predikatene til andre predikater ved å bruke AND og OR i WHERE-leddet til den eksterne spørringen.

## Kvantifiserte predikater

Et kvantifisert predikat sammenlikner en verdi med en samling med verdier. Hvis en full SELECT-setning returnerer mer enn en verdi, må du endre sammenlikningsoperatorene i predikatet ved å tilknytte suffikset ALL, ANY eller SOME. Disse suffiksene bestemmer hvordan settet med verdier som returneres, skal behandles i det eksterne predikatet.
Sammenlikningsoperatoren > brukes som et eksempel (merknadene nedenfor gjelder for de andre operatorene også):

**uttrykk > ALL (full SELECT)**
Predikatet er sant hvis uttrykket er større enn hver individuelle verdi som returneres av full SELECT-setningen. Hvis full SELECT-setningen ikke returnerer noen verdier, er predikatet sant. Resultatet er usant hvis det oppgitte forholdet er usant for minst en verdi. Legg merke til at det kvantifiserte predikatet <>ALL, tilsvarer NOT IN-predikatet.

Det neste eksempelet bruker en delspørring og en >-ALL-sammenlikning til å finne navnet og yrket til alle ansatte som tjener mer enn alle avdelingssjefer:

```
SELECT LASTNAME, JOB
   FROM EMPLOYEE
   WHERE SALARY > ALL
   (SELECT SALARY
       FROM EMPLOYEE
       WHERE JOB='MANAGER')
```

**uttrykk > ANY (full SELECT)**
Predikatet er sant hvis uttrykket er større enn minst en av verdiene som returneres av full SELECT-setningen. Hvis full SELECT-setningen ikke returnerer noen verdier, er predikatet usant. Legg merke til at den kvantifiserte operatoren =ANY tilsvarer IN-predikatet.

**uttrykk > SOME (full SELECT)**
SOME er synonymt med ANY.

Du finner flere opplysninger om predikater og operatorer i *SQL Reference*.

# Kapittel 7. Avansert SQL

Dette kapittelet dekker flere funksjoner i DB2 Universal Database som gjør det mulig å utforme spørringer på en mer effektiv måte, mens du tilpasser dem til dine behov. Emnene i dette kapittelet er basert på at du har forstått materialet du har lest i de andre kapitlene.

Dette kapittelet inneholder:

- Håndheve forretningsregler med begrensninger og utløsere
- Kombineringer
- ROLLUP- og CUBE-spørringer og Rekursive spørringer
- OLAP-funksjoner

## Håndheve forretningsregler med begrensninger og utløsere

I forretningsverdenen er det behov for å lage visse regler som alltid blir håndhevet. En ansatt som jobber på et prosjekt, må for eksempel være på lønningslisten. Eller det kan være at vi vil at noen aktiviteter skal skje automatisk. Hvis en salgsperson for eksempel foretar et salg, skal kommisjonen økes.

DB2 Universal Database tilbyr en nyttig pakke med metoder for dette formålet:

- *Entydige begrensninger* forbyr like verdier i en eller flere kolonner i en tabell.
- *Referanseintegritetsbegrensninger* sørger for at dataene forblir konsistente i de oppgitte tabellene.
- *Tabellkontrollbegrensninger* er regler som plasserer begrensninger på verdiene som en kolonne er tillatt å ha. Innsettinger og oppdateringer mislykkes hvis en verdi som tildeles til en kolonne, ikke oppfyller kontrollbegrensningen(e) for den kolonnen.
- *Utløsere* definerer et sett med handlinger som blir utført eller utløst av en sletting, innsetting eller oppdateringsoperasjon på en bestemt tabell. Utløsere kan brukes til å skrive til andre tabeller, endre inndataverdier eller sende varsler.

Den første delen gir en begrepsoversikt over nøklene. Senere blir referanseintegritet, begrensninger og utløsere omtalt gjennom eksempler og diagrammer.

## Nøkler

En *nøkkel* er et sett med begrensninger som du kan bruke til å identifisere eller få tilgang til en bestemt rad eller bestemte rader.

En sammensatt nøkkel som består av flere enn en kolonne, kalles en *sammensatt nøkkel*. I en tabell med en sammensatt nøkkel samsvarer ikke rekkefølgen på kolonnene i den sammensatte nøkkelen nødvendigvis med rekkefølgen i tabellen.

### Entydige nøkler

En *entydig nøkkel* er definert som en kolonne (eller et sett av kolonner) der det ikke fins to like verdier. Kolonnene som er av en entydig nøkkel, kan ikke inneholde nullverdier. Begrensningen blir håndhevet av databasesystemet under utføringen av INSERT- og UPDATE-setninger. En tabell kan ha flere entydige nøkler. Entydige nøkler er valgfrie og kan defineres i CREATE TABLE- eller ALTER TABLE-setninger.

### Primærnøkler

En *primærnøkkel* er en entydig nøkkel som er en del av definisjonen av tabellen. En tabell kan ikke ha mer enn en primærnøkkel, og kolonnene til en primærnøkkel kan ikke inneholde nullverdier. Primærnøkler er valgfrie og kan defineres i CREATE TABLE- eller ALTER TABLE-setninger.

### Fremmednøkler

En *fremmednøkkel* er oppgitt i definisjonen av en referansebegrensning. En tabell kan ha null eller flere fremmednøkler. Verdien til den sammensatte fremmednøkkelen er null hvis en av komponentene i verdien er null. Fremmednøkler er valgfrie og kan defineres i CREATE TABLE- eller ALTER TABLE-setninger.

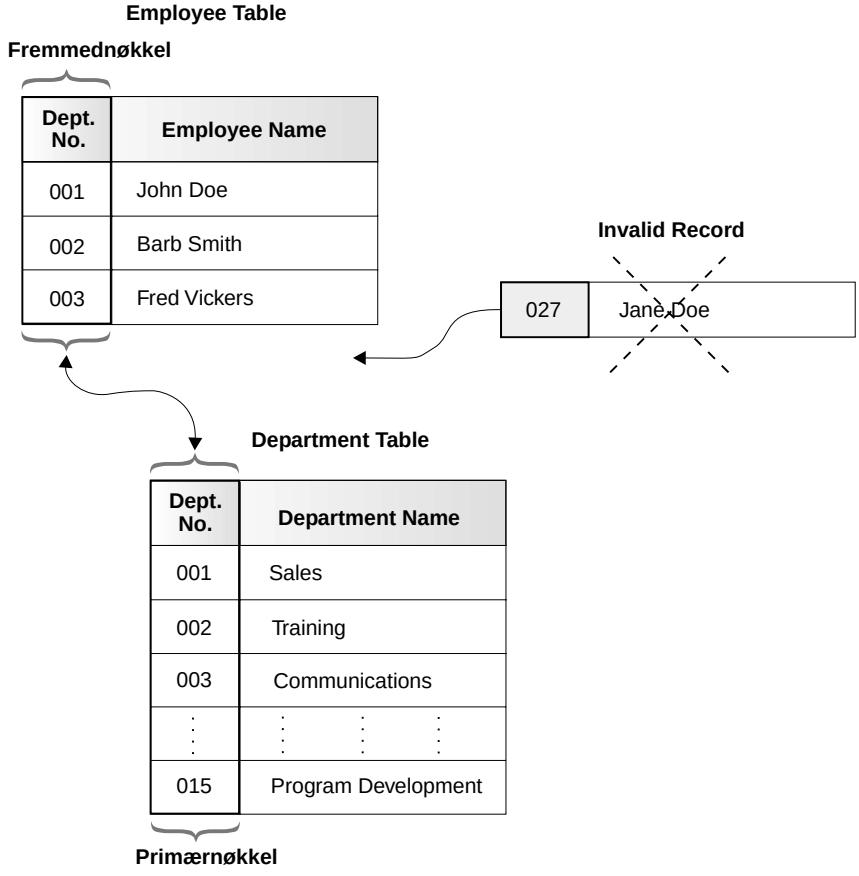
## Entydige begrensninger

En entydig begrensning sørger for at verdiene til en nøkkel er entydige i en tabell. Entydige begrensninger er valgfrie og du kan definere dem ved hjelp av CREATE TABLE- eller ALTER TABLE-setninger ved å oppgi PRIMARY KEY- eller UNIQUE-leddet. Du kan for eksempel definere en entydig begrensning på kolonnen for funksjonærnummer i en tabell for å sørge for at hver ansatt har et entydig nummer.

## Referanseintegritetsbegrensninger

Ved å definere entydige begrensninger og fremmednøkler kan du definere forhold mellom tabeller, og slik kan du håndheve visse forretningsregler. Kombinasjonen av entydige nøkkelbegrensninger og fremmednøkkelbegrensninger blir vanligvis referert til som referanseintegritetsbegrensninger. En entydig begrensning som er referert til av en fremmednøkkel, blir kalt en *overordnet nøkkel*. En fremmednøkkel refererer til eller er knyttet til en bestemt overordnet nøkkel. En regel kan for eksempel si at hver ansatt (tabellen EMPLOYEE) må tilhøre en eksisterende

avdeling (tabellen DEPARTMENT). Vi definerer derfor avdelingsnummer i tabellen EMPLOYEE som en fremmednøkkel og avdelingsnummer i tabellen DEPARTMENT som primærnøkkelen. Diagrammet nedenfor gir en visuell beskrivelse av referanseintegritetsbegrensninger.




*Figur 4. Fremmed- og primærbegrensninger definerer forhold og beskytter data*

## Tabellkontrollbegrensninger

*Tabellkontrollbegrensninger* oppgir betingelser som evalueres for hver rad i en tabell. Du kan spesifisere kontrollbegrensninger for individuelle kolonner. Du kan tilføye dem ved å bruke CREATE TABLE- eller ALTER TABLE-setningene.

Setningen nedenfor oppretter en tabell med følgende begrensninger:

- Verdiene til avdelingsnummeret må ligge i området fra 10 til 100
- Jobben til en ansatt kan bare være en av følgende: “Sales”, “Mgr” eller “Clerk”
- Hver ansatt som ble ansatt før 1986, må tjene mer enn $40, 500.

```
CREATE TABLE EMP
    ( ID              SMALLINT        NOT NULL,
      NAME            VARCHAR(9),
       DEPT          SMALLINT CHECK (DEPT BETWEEN 10 AND 100),
       JOB           CHAR(5)   CHECK (JOB IN ('Sales', 'Mgr', 'Clerk')),
       HIREDATE      DATE,
      SALARY           DECIMAL(7,2),
      COMM             DECIMAL(7,2),
       PRIMARY KEY (ID),
       CONSTRAINT YEARSAL CHECK
                          (YEAR(HIREDATE) >= 1986 OR SALARY > 40500) )
```

En begrensning brytes bare hvis betingelsen blir evaluert til usann. Hvis for eksempel DEPT er NULL for en innsatt rad, fortsetter innsettingen uten feil selv om verdiene for DEPT skal ligge mellom 10 og 100 slik det er definert i begrensningen.

Setningen nedenfor tilføyer en begrensning til EMPLOYEE-tabellen kalt COMP, som sier at den samlede kompensasjonen til en ansatt må overskride $15, 000:

```
ALTER TABLE EMP
   ADD CONSTRAINT COMP CHECK (SALARY + COMM > 15000)
```

De eksisterende radene i tabellen blir kontrollert for å sørge for at de ikke bryter den nye begrensningen. Du kan utsette denne kontrollen ved å bruke setningen SET CONSTRAINTS på denne måten:

```
SET CONSTRAINTS FOR EMP OFF
ALTER TABLE EMP ADD CONSTRAINT COMP CHECK (SALARY + COMM > 15000)
SET CONSTRAINTS FOR EMP IMMEDIATE CHECKED
```

Først blir SET CONSTRAINTS-setningen brukt til å utsette begrensningskontroll for tabellen. Deretter kan du tilføye en eller flere begrensninger til tabellen uten å kontrollere begrensningene. SET CONSTRAINTS-setningen blir deretter gitt på nytt for å aktivere begrensningskontroll igjen og utføre eventuell utsatt begrensningskontroll.

## Utløsere

En *utløser* definerer et sett med handlinger. Utløsere aktiveres av operasjoner som endrer dataene i en spesifisert basistabell.

Noen måter å bruke utløsere på:

- å utføre godkjennelse av inndata
- å generere en verdi for en nylig innsatt rad automatisk
- å lese fra andre tabeller for kryssreferanseformål
- å skrive til andre tabeller for revisjonsloggformål

- å støtte varsler gjennom elektronisk post

Bruk av utløsere gir raskere applikasjonsutvikling, global håndhevelse av forretningsregler og enklere vedlikehold av applikasjoner og data.

DB2 Universal Database støtter flere typer utløsere. Utløsere kan defineres til å aktiveres enten før eller etter en DELETE-, INSERT- eller UPDATE-operasjon. Hver utløser inkluderer et sett med SQL-setninger som kalles en *utløserhandling*, som kan inkludere en valgfri søkebetingelse.

*Etter-utløsere* kan også defineres til å utføre utløserhandlingen enten for hver rad eller en gang for setningen. *Før-utløsere* utfører alltid utløserhandlingen for hver rad.

Bruk en utløser før en INSERT-, UPDATE- eller DELETE-setning for å kontrollere bestemte betingelser før du utfører en utløseroperasjon eller for å endre inndataverdiene før de lagres i tabellen.

Bruk en etter-utløser til å propagere verdier etter behov eller utføre andre oppgaver, for eksempel sende en melding, som kan være nødvendige som en del av utløseroperasjonen.

Det neste eksempelet illustrerer en bruk av før- og etter-utløsere. Vurder en applikasjon som registrerer og sporer endringer i aksjekursene. Databasen inneholder to tabeller, CURRENTQUOTE og QUOTEHISTORY, definert som:

```
CREATE TABLE CURRENTQUOTE
(SYMBOL VARCHAR(10),
 QUOTE DECIMAL(5,2),
 STATUS VARCHAR(9))

CREATE TABLE  QUOTEHISTORY
(SYMBOL VARCHAR(10),
 QUOTE DECIMAL(5,2),
 TIMESTAMP TIMESTAMP)
```

Når QUOTE-kolonnen i CURRENTQUOTE oppdateres ved hjelp av en setning som dette:

```
UPDATE CURRENTQUOTE
   SET QUOTE = 68.5
   WHERE SYMBOL = 'IBM'
```

STATUS-kolonnen i CURRENTQUOTE bør oppdateres for å gjenspeile om aksjene:
- stiger i verdi
- har nådd en ny toppverdi for året
- synker i verdi

- har nådd en ny bunnverdi for året
- har en stabil verdi

Dette gjøres ved å bruke denne utløseren:

**1**

```
CREATE TRIGGER STOCK_STATUS
   NO CASCADE BEFORE UPDATE OF QUOTE ON CURRENTQUOTE
   REFERENCING NEW AS NEWQUOTE OLD AS OLDQUOTE
   FOR EACH ROW MODE DB2SQL
```

**2**

```
 SET NEWQUOTE.STATUS =
```

**3**

```
    CASE
```

**4**

```
      WHEN NEWQUOTE.QUOTE >=
                (SELECT MAX(QUOTE)
                    FROM QUOTEHISTORY
                    WHERE SYMBOL = NEWQUOTE.SYMBOL
                      AND YEAR(TIMESTAMP) = YEAR(CURRENT DATE) )
      THEN 'High'
```

**5**

```
     WHEN NEWQUOTE.QUOTE <=
                (SELECT MIN(QUOTE)
                    FROM QUOTEHISTORY
                    WHERE SYMBOL = NEWQUOTE.SYMBOL
                    AND YEAR(TIMESTAMP) = YEAR(CURRENT DATE) )
     THEN 'Low'
```

**6**

```
     WHEN NEWQUOTE.QUOTE > OLDQUOTE.QUOTE
        THEN 'Rising'
     WHEN NEWQUOTE.QUOTE < OLDQUOTE.QUOTE
        THEN 'Dropping'
     WHEN NEWQUOTE.QUOTE = OLDQUOTE.QUOTE
        THEN 'Steady'
  END
```

**1** Denne blokken med kode definerer STOCK_STATUS som en utløser som skal aktiveres før oppdateringen av QUOTE-kolonnen i CURRENTQUOTE-tabellen. Den andre linjen spesifiserer at utløserhandlingen skal tas i bruk før eventuelle endringer som forårsakes av den faktiske oppdateringen av CURRENTQUOTE-tabellen, tas i bruk i databasen. NO CASCADE-leddet betyr at utløserhandlingen ikke gjør at andre utløsere blir aktivert. Den tredje

linjen spesifiserer navnene som må brukes som kvalifikatorer av kolonnenavnet for de nye verdiene (NEWQUOTE) og gamle verdiene (OLDQUOTE). Kolonnenavn som er kvalifisert med disse korrelasjonsnavnene (NEWQUOTE og OLDQUOTE), kalles *overgangsvariabler*. Den fjerde linjen viser at utløserhandlingen skal utføres for hver rad.

**2** Dette markerer starten på den første og eneste SQL-setningen i utløserhandlingen til denne utløseren. SET-overgangsvariabelsetningen brukes i en utløser for å tildele en verdi til en kolonne i raden til tabellen som blir oppdatert av setningen som aktiverer utløseren. Denne setningen tildeler en verdi til STATUS-kolonnen i CURRENTQUOTE-tabellen.

**3** Uttrykket som brukes på høyre side av tildelingen er et CASE-uttrykk. CASE-uttrykket går helt til END-nøkkelordet.

**4** Det første case-uttrykket sjekker om den nye noteringen (NEWQUOTE.QUOTE) overskrider maksimumsverdien for aksjesymbolet i det gjeldende kalenderåret. Delspørringen bruker QUOTEHISTORY-tabellen som er oppdatert av etter-utløseren som kommer etter.

**5** Det andre case-uttrykket sjekker om den nye noteringen (NEWQUOTE.QUOTE) er mindre enn minimumsverdien for aksjesymbolet i det gjeldende kalenderåret. Delspørringen bruker QUOTEHISTORY-tabellen som er oppdatert av etter-utløseren som kommer etter.

**6** De tre siste case-uttrykkene sammenlikner den nye noteringen (NEWQUOTE.QUOTE) med noteringen som var i tabellen (OLDQUOTE.QUOTE), for å bestemme om den er større, mindre eller lik. SET-overgangsvariabelsetningen slutter her.

I tillegg til å oppdatere posten i CURRENTQUOTE-tabellen, må en revisjonsloggpost opprettes ved å kopiere den nye noteringen med en systemtid til QUOTEHISTORY-tabellen. Dette gjøres ved å bruke denne etter-utløseren:

**1**

```
CREATE TRIGGER RECORD_HISTORY
AFTER UPDATE OF QUOTE ON CURRENTQUOTE
REFERENCING NEW AS NEWQUOTE
FOR EACH ROW MODE DB2SQL
BEGIN ATOMIC
```

**2**

```
INSERT INTO QUOTEHISTORY
   VALUES (NEWQUOTE.SYMBOL, NEWQUOTE.QUOTE, CURRENT TIMESTAMP);
END
```

**1** Denne blokken med kode definerer en utløser kalt RECORD_HISTORY som en utløser som skal aktiveres etter oppdateringen av QUOTE-kolonnen i CURRENTQUOTE-tabellen. Den tredje linjen spesifiserer navnet som skal brukes som en kvalifikator av kolonnenavnet for den nye verdien (NEWQUOTE). Den fjerde linjen viser at utløserhandlingen skal utføres for hver rad.

**2** Utløserhandlingen til denne utløseren inkluderer en enkelt SQL-setning som setter inn en rad i QUOTEHISTORY-tabellen ved hjelp av data fra raden som er oppdatert (NEWQUOTE.SYMBOL og NEWQUOTE.QUOTE), og den gjeldende systemtiden.

CURRENT TIMESTAMP er et spesialregister som inneholder systemtiden. Du finner en liste og en forklaring i “Spesialregistre” på side 68.

## Kombineringer

Prossessen med å kombinere data fra to eller flere tabeller kalles å kombinere tabeller. Databasesystemet lager alle slags kombineringer av rader fra de spesifiserte tabellene. For hver kombinasjon tester det *kombinasjonsbetingelsen*. En kombinasjonsbetingelse er en søkebetingelse med visse begrensninger. Du finner en liste over begrensningene i *SQL Reference*.

Vær oppmerksom på at datatypene i kolonnene som er med i kombineringen, ikke trenger å være like, men de må imidlertid være kompatible. Kombinasjonsbetingelsen evalueres på samme måte som en hvilken som helst annen søkebetingelse og de samme reglene gjelder for sammenlikning.

Hvis du ikke spesifiserer en kombinasjonsbetingelse, blir alle kombinasjoner av rader fra tabeller som er oppført i FROM-leddet, returnert, selv om radene ikke har noen forbindelse. Resultatet blir referert til som *kryssproduktet* av de to tabellene.

Eksemplene i dette avsnittet er basert på de to neste tabellene. De er forenklinger av tabellene fra eksempeldatabasen, men finnes ikke i eksempeldatabasen. De brukes til å gi en oversikt over interessante punkter om kombineringer generelt. SAMP_STAFF viser en liste over navnene til de ansatte som ikke er ansatt som underleverandører, og jobbeskrivelsene, mens SAMP_PROJECT viser en liste over navnene på de ansatte (kontrakt og heltid) og prosjektene de jobber på.

Dette er tabellene:

| NAME | PROJ |
|---|---|
| Haas | AD3100 |
| Thompson | PL2100 |
| Walker | MA2112 |
| Lutz | MA2111 |

*Figur 5. SAMP_PROJECT TABLE*

| NAME | JOB |
|---|---|
| Haas | PRES |
| Thompson | MANAGER |
| Lucchessi | SALESREP |
| Nicholls | ANALYST |

*Figur 6. SAMP_STAFF TABLE*

Eksempelet nedenfor lager kryssproduktet av de to tabellene. Det er ikke oppgitt en kombinasjonsbetingelse, så hver kombinasjon av radene blir vist:

```
SELECT SAMP_PROJECT.NAME,
       SAMP_PROJECT.PROJ, SAMP_STAFF.NAME, SAMP_STAFF.JOB
  FROM SAMP_PROJECT, SAMP_STAFF
```

Denne setningen gir dette resultatet:

```
NAME       PROJ   NAME       JOB
---------- ------ ---------- --------
Haas       AD3100 Haas       PRES
Thompson   PL2100 Haas       PRES
Walker     MA2112 Haas       PRES
Lutz       MA2111 Haas       PRES
Haas       AD3100 Thompson   MANAGER
Thompson   PL2100 Thompson   MANAGER
Walker     MA2112 Thompson   MANAGER
Lutz       MA2111 Thompson   MANAGER
Haas       AD3100 Lucchessi  SALESREP
Thompson   PL2100 Lucchessi  SALESREP
Walker     MA2112 Lucchessi  SALESREP
Lutz       MA2111 Lucchessi  SALESREP
Haas       AD3100 Nicholls   ANALYST
Thompson   PL2100 Nicholls   ANALYST
Walker     MA2112 Nicholls   ANALYST
Lutz       MA2111 Nicholls   ANALYST
```

De to hovedtypene av kombineringer er *interne kombineringer* og *eksterne kombineringer*. Hittil har vi brukt den interne kombineringen i alle eksemplene. Interne kombineringer beholder bare radene fra kryssproduktet som oppfyller kombinasjonsbetingelsen. Hvis en rad finnes i en tabell, men ikke i den andre, blir ikke opplysningene tatt med i resultattabellen.

Eksempelet nedenfor lager den interne kombineringen av de to tabellene. Den interne kombineringen viser en liste over de heltidsansatte som jobber på et prosjekt:

```
SELECT SAMP_PROJECT.NAME,
       SAMP_PROJECT.PROJ, SAMP_STAFF.NAME, SAMP_STAFF.JOB
  FROM SAMP_PROJECT, SAMP_STAFF
  WHERE SAMP_STAFF.NAME = SAMP_PROJECT.NAME
```

Du kan også spesifisere den interne kombineringen på denne måten:

```
SELECT SAMP_PROJECT.NAME,
       SAMP_PROJECT.PROJ, SAMP_STAFF.NAME, SAMP_STAFF.JOB
  FROM SAMP_PROJECT INNER JOIN SAMP_STAFF
    ON SAMP_STAFF.NAME = SAMP_PROJECT.NAME
```

Resultatet er:

```
NAME       PROJ   NAME       JOB
---------- ------ ---------- --------
Haas       AD3100 Haas       PRES
Thompson   PL2100 Thompson   MANAGER
```

Legg merke til at resultatet av den interne kombineringen består av rader som har tilsvarende verdier for NAME-kolonnen i tabellene til høyre og venstre - både 'Haas' og 'Thompson' er inkludert i SAMP_STAFF-tabellen som viser en liste over heltidsansatte, og i SAMP_PROJECT-tabellen som viser en liste over heltids- og kontraktansatte som jobber på et prosjekt.

Eksterne kombineringer er en sammenkjeding av den interne kombineringen og radene fra den venstre tabellen, den høyre tabellen eller begge tabellene som mangler i den interne kombineringen. Når du utfører en ekstern kombinering på to tabeller, tildeler du tilfeldig en tabell som venstre tabell og den andre som høyre tabell. Det finnes tre typer eksterne kombineringer:

1. ***Venstreekstern kombinering*** inkluderer den interne kombineringen og radene fra den venstre tabellen som ikke er inkludert i den interne kombineringen
2. ***Høyreekstern kombinering*** inkluderer den interne kombineringen og radene fra den høyre tabellen som ikke er inkludert i den interne kombineringen
3. ***Fullstendig ekstern kombinering*** inkluderer den interne kombineringen og radene fra både den høyre og venstre tabellen som ikke er inkludert i den interne kombineringen.

Bruk SELECT-setningen til å spesifisere kolonnene som skal vises. I FROM-leddet oppgir du navnet på den første tabellen etterfulgt av nøkkelordene LEFT OUTER JOIN, RIGHT OUTER JOIN eller FULL OUTER JOIN. Deretter må du oppgi den andre tabellen etterfulgt av ON-nøkkelordet. Etter ON-nøkkelordet oppgir du kombinasjonsbetingelsen for å uttrykke et forhold mellom tabellene som skal kombineres.

I eksempelet nedenfor er SAMP_STAFF definert som høyre tabell og SAMP_PROJECT som venstre tabell. Ved å bruke LEFT OUTER JOIN får vi frem en liste over navnene og prosjektnumrene på alle de ansatte, både på heltid og kontrakt, (oppført i SAMP_PROJECT) og jobbstillingen deres, hvis de er heltidsansatte (oppført i SAMP_STAFF):

```
SELECT SAMP_PROJECT.NAME, SAMP_PROJECT.PROJ,
       SAMP_STAFF.NAME, SAMP_STAFF.JOB
  FROM SAMP_PROJECT LEFT OUTER JOIN SAMP_STAFF
    ON SAMP_STAFF.NAME = SAMP_PROJECT.NAME
```

Denne setningen gir dette resultatet:

```
NAME       PROJ                NAME       JOB
---------- ------------------- ---------- --------------------
Haas       AD3100              Haas       PRES
Lutz       MA2111              -          -
Thompson   PL2100              Thompson   MANAGER
Walker     MA2112              -          -
```

Rader med verdier i alle kolonnene er resultatet av den interne kombineringen. Dette er rader som oppfyller kombinasjonsbetingelsen: 'Haas' og 'Thompson' er oppført i både SAMP_PROJECT (venstre tabell) og SAMP_STAFF (høyre tabell). For rader der kombinasjonsbetingelsen ikke ble oppfylt, blir nullverdien vist i kolonner i den høyre tabellen: 'Lutz' og 'Walker' er kontraktansatte som er oppført i SAMP_PROJECT-tabellen og ikke i SAMP_STAFF-tabellen. Vær oppmerksom på at alle rader fra den venstre tabellen er inkludert i resultatsettet.

I det neste eksempelet er SAMP_STAFF definert som høyre tabell og SAMP_PROJECT som venstre tabell. Ved å bruke RIGHT OUTER JOIN får vi frem en liste over navnene og jobbstillingene til alle heltidsansatte (oppført i SAMP_STAFF) og prosjektnummeret deres hvis de jobber på et prosjekt (oppført i SAMP_PROJECT):

```
SELECT SAMP_PROJECT.NAME,
       SAMP_PROJECT.PROJ, SAMP_STAFF.NAME, SAMP_STAFF.JOB
  FROM SAMP_PROJECT RIGHT OUTER JOIN SAMP_STAFF
    ON SAMP_STAFF.NAME = SAMP_PROJECT.NAME
```

Resultatet er:

```
NAME       PROJ                 NAME       JOB
---------- -------------------- ---------- --------------------
Haas       AD3100               Haas       PRES
-          -                    Lucchessi  SALESREP
-          -                    Nicholls   ANALYST
Thompson   PL2100               Thompson   MANAGER
```

Som i den venstreeksterne kombineringen er rader med verdier i alle kolonnene resultatet av den interne kombineringen. Dette er rader som oppfyller kombinasjonsbetingelsen: 'Haas' og 'Thompson' er oppført i både SAMP_PROJECT (venstre tabell) og SAMP_STAFF (høyre tabell). For rader der kombinasjonsbetingelsen ikke ble oppfylt, blir nullverdien vist i den høyre tabellen: 'Lucchessi' og 'Nicholls' er heltidsansatte som ikke jobber på et prosjekt. Når de er oppført i SAMP_STAFF, er de ikke oppført i SAMP_PROJECT. Vær oppmerksom på at alle rader fra den høyre tabellen er inkludert i resultatsettet.

Det neste eksempelet bruker FULL OUTER JOIN sammen med SAMP_PROJECT- og SAMP_STAFF-tabellen. Det viser navnene på alle heltidsansatte, inkludert de som ikke jobber på et prosjekt, og kontraktansatte:

```
SELECT SAMP_PROJECT.NAME, SAMP_PROJECT.PROJ,
       SAMP_STAFF.NAME, SAMP_STAFF.JOB
  FROM SAMP_PROJECT FULL OUTER JOIN SAMP_STAFF
    ON SAMP_STAFF.NAME = SAMP_PROJECT.NAME
```

Resultatet er:

```
NAME       PROJ                 NAME       JOB
---------- -------------------- ---------- --------------------
Haas       AD3100               Haas       PRES
-          -                    Lucchessi  SALESREP
-          -                    Nicholls   ANALYST
Thompson   PL2100               Thompson   MANAGER
Lutz       MA2111               -          -
Walker     MA2112               -          -
```

Dette resultatet inkluderer den venstreeksterne kombineringen, den høyreeksterne kombineringen og den interne kombineringen. Alle heltidsansatte og kontraktansatte er oppført på listen. For verdier der kombinasjonsbetingelsen ikke ble oppfylt, blir nullverdien vist i den respektive kolonnen, akkurat som for venstreekstern og høyreekstern kombinering. Hver rad fra SAMP_STAFF og SAMP_PROJECT er inkludert i resultatsettet.

## Sammensatte spørringer

DB2 Universal Database gjør det mulig å gruppere, konsolidere og se på flere kolonner i et enkelt resultatsett ved å bruke ROLLUP og CUBE. Denne nye og kraftige funksjonen forbedrer og forenkler SQL basert på dataanalyse.

Det finnes forskjellige metoder for å trekke ut nyttig informasjon fra databasen. Du kan implementere rekursive spørringer for å lage resultattabeller fra eksisterende datasett.

### ROLLUP- og CUBE-spørringer

Du oppgir ROLLUP- og CUBE-operasjoner i GROUP BY-leddet til en spørring.

ROLLUP-gruppering gir et resultatsett som inneholder de vanlige grupperte radene og *delsumrader*. CUBE-gruppering gir et resultatsett som inneholder rader fra ROLLUP- og krysstabuleringsrader.

For ROLLUP kan du få salgssum per person per måned med månedlige salgssummer og en samlet sum. For CUBE blir ekstra rader inkludert for salgssum pr person.

Du finner flere opplysninger i *SQL Reference*.

### Rekursive spørringer

En *rekursiv spørring* er en spørring som gjentakende bruker resultatdata for å bestemme videre resultater. Du kan se på dette som å krysse en treoversikt eller et diagram.

Praktiske eksempler inkluderer reservasjonssystemer, nettverksplanlegging og planlegging.

En rekursiv spørring er skrevet ved å bruke et felles tabelluttrykk som inkluderer en referanse til sitt eget navn.

Du finner eksempler på rekursive spørringer i *SQL Reference*.

## OLAP-funksjoner

OLAP-funksjoner (OnLine Analytical Processing) utfører en kolonnefunksjonoperasjon over et *vindu* med data. Dette vinduet kan oppgi en partisjonering av rader, en sortering av rader i partisjoner eller en *samlingsgruppe*. Samlingsgruppen gjør det mulig for brukeren å oppgi hvilke rader, knyttet til den gjeldende raden, som deltar i beregningen. Bruken av et slikt vindu gjør det mulig med operasjoner som kumulative summer og bevegelige gjennomsnitt.

I tillegg til at det blir mulig for deg å oppgi et vindu for eksisterende kolonnefunksjoner (for eksempel SUM og AVG), kan OLAP-funksjonene utføre rangeringsoperasjoner (RANK og DENSE_RANK) og sørge for radnummerering (ROW_NUMBER), gitt en bestemt partisjonering og rekkefølge av radene.

Eksempelet nedenfor viser rangen til en ansatt i avdelingen basert på lønn og viser den kumulative summen for lønningene i avdelingen (for avdelingene 15 og 38):

```
SELECT NAME, DEPT,
       RANK () OVER (PARTITION BY DEPT ORDER BY SALARY DESC) AS RANK,
       SUM (SALARY) OVER (PARTITION BY DEPT
                          ORDER BY SALARY DESC
                          ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW)
         AS CUMULATIVE_SUM
       FROM STAFF
  WHERE DEPT IN (15,38)
  ORDER BY DEPT, RANK
```

Denne setningen gir dette resultatet:

```
NAME       DEPT      RANK       CUMULATIVE_SUM
---------- --------- ---------- --------------------
Hanes      15                 1             20659.80
Rothman    15                 2             37162.63
Ngan       15                 3             49670.83
Kermisch   15                 4             61929.33
O'Brien    38                 1             18006.00
Marenghi   38                 2             35512.75
Quigley    38                 3             52321.05
Naughton   38                 4             65275.80
Abrahams   38                 5             77285.55
```

# Kapittel 8. Tilpasse og forbedre datamanipulering

Dette kapittelet gir en kort innledning til *objektorienterte tilleggsprogrammer* i DB2 Universal Database. Det er mange fordeler ved å bruke objektorienterte tilleggsprogrammer. *Brukerdefinerte typer (UDT)* øker settet med datatyper som er tilgjengelige for applikasjonene, mens *brukerdefinerte funksjoner (UDF)* gjør det mulig å opprette applikasjonsspesifikke funksjoner. UDFer fungerer som *metoder* for UDTer ved å gi en konsistent virkemåte og innkapsling av typene.

*Spesialregistre* og *systemkataloger* blir omtalt nedenfor. Spesialregistre er lagringsområder som er definert av databasesystemet. De brukes til å lagre opplysninger som SQL-setninger kan referere til. Spesialregistre blir opprettet ved tilkoblingstidspunktet og er spesifikke for behandlingen av den prosessen. Systemkatalogene inneholder opplysninger om den logiske og fysiske strukturen til databaseobjektene.

Dette kapittelet inneholder:

- Brukerdefinerte typer
- Brukerdefinerte funksjoner
- Store objekter (LOB)
- Spesialregistre
- Innledning til katalogoversikter

En detaljert omtale av emnene ovenfor dekkes ikke i denne boken, men blir presentert i *SQL Reference* og *Administration Guide.*

## Brukerdefinerte typer

En *distinkt type* er en brukerdefinert datatype som deler internfremstillingen med en eksisterende type (dens “kilde”type), men blir sett på som separat og inkompatibel for de fleste operasjoner. Det kan for eksempel hende at du vil definere en alderstype, vekttype og høydetype, som alle har ganske forskjellig semantikk, men som bruker den innebygde datatypen INTEGER for internfremstillingene.

Eksempelet nedenfor illustrerer opprettelsen av en distinkt type kalt PAY:

```
CREATE DISTINCT TYPE PAY AS DECIMAL(9,2) WITH COMPARISONS
```

Selv om PAY har den samme fremstillingen som den innebygde datatypen DECIMAL(9,2), blir den sett på som en separat type som ikke kan sammenliknes med DECIMAL(9,2) eller noen annen type. Den kan bare

sammenliknes med den samme distinkte typen. Operatorer og funksjoner som fungerer på DECIMAL, kan ikke brukes her. En verdi med datatypen PAY kan for eksempel ikke multipliseres med en verdi av datatypen INTEGER. Du må derfor skrive funksjoner som bare gjelder for datatypen PAY.

Når du bruker distinkte datatyper, begrenser du tilfeldige feil. Hvis for eksempel SALARY-kolonnen i tabellen EMPLOYEE var definert som en PAY-datatype, kunne den ikke blitt tilføyd til COMM selv om kildetypene er de samme.

Distinkte datatyper støtter konvertering. En kildetype kan konverteres til en distinkt datatype og en distinkt datatype til en kildetype. Hvis for eksempel SALARY-kolonnen i tabellen EMPLOYEE var definert som en PAY-datatype, ville ikke eksempelet nedenfor ha vært mislykket ved sammenlikningsoperatoren.

```
SELECT * FROM EMPLOYEE
   WHERE DECIMAL(SALARY) = 41250
```

DECIMAL(SALARY) returnerer en DECIMAL-datatype. På omvendt måte kan en numerisk datatype konverteres til en PAY-type. Du kan for eksempel konvertere tallet 41250 ved å bruke PAY(41250).

## Brukerdefinerte funksjoner

Som nevnt i “Bruke funksjoner” på side 28, sørger DB2 Universal Database for innebygde og brukerdefinerte funksjoner (UDF). Dette settet med funksjoner oppfyller imidlertid aldri alle kravene. Du må ofte opprette tilpassede funksjoner for bestemte oppgaver. Brukerdefinerte funksjoner gjør det mulig å opprette tilpassede funksjoner.

Det finnes fire typer brukerdefinerte funksjoner *kilde* (eller mal), *ekstern skalar*, *ekstern tabell* og *OLE DB-ekstern tabell*.

Dette avsnittet dekker typene kilde og ekstern skalar. Du finner flere opplysninger om typene ekstern tabell og OLE DB-tabell i *SQL Reference*.

Brukerdefinerte kildefunksjoner gjør det mulig for brukerdefinerte typer å selektivt referere til en annen innebygd eller brukerdefinert funksjon som allerede er kjent for databasen. Du kan bruke både skalar- og kolonnefunksjoner.

I det neste eksempelet blir det opprettet en brukerdefinert funksjon (kalt MAX) som er basert på den innebygde MAX-kolonnefunksjonen, som tar en DECIMAL-datatype som inndata. Den brukerdefinerte funksjonen MAX tar en PAY-type som inndata og returnerer en PAY-type som utdata.

```
CREATE FUNCTION MAX(PAY) RETURNS PAY
   SOURCE MAX(DECIMAL)
```

Eksterne brukerdefinerte funksjoner er skrevet av brukere på et programmeringsspråk. Det finnes *eksterne skalarfunksjoner* og *eksterne tabellfunksjoner* og begge er omtalt i *SQL Reference*.

Som et annet eksempel antar vi at du allerede har skrevet en funksjon som teller antall ord i en streng, du kan registrere den hos databasen ved hjelp av CREATE FUNCTION-setningen med navnet WORDCOUNT. Denne funksjonen kan deretter brukes i SQL-setninger.

Den neste setningen returnerer funksjonærnumre og antall ord i ASCII-skjemaet til CVene. WORDCOUNT er en ekstern skalarfunksjon som er registrert hos databasen av brukeren, og brukes nå i setningen.

```
SELECT EMPNO, WORDCOUNT(RESUME)
   FROM EMP_RESUME
   WHERE RESUME_FORMAT = 'ascii'
```

Du finner detaljerte opplysninger om hvordan du skriver brukerdefinerte funksjoner i *Application Development Guide*.

## Store objekter (LOB)

Termen *store objekter* og akronymet *LOB* brukes til å referere til tre datatyper: BLOB, CLOB eller DBCLOB. Disse typene kan inneholde store mengder med data for objekter som for eksempel lyd, bilder og dokumenter.

En *BLOB (Binary Large OBject)* er en streng med variabel lengde som måles i byte, og kan være opptil 2 gigabyte lang. En BLOB er hovedsakelig ment å inneholde utradisjonelle data, for eksempel lyd, bilder og blandede medier.

En *CLOB (Character Large OBject)* er en streng med variabel lengde som måles i byte, og kan være opptil 2 gigabyte lang. En CLOB brukes til å lagre store SBCS-data, for eksempel dokumenter. En CLOB blir sett på som en tegnstreng.

En *DBCLOB (Double-Byte Character Large OBject)* er en streng med variabel lengde og dobbeltbytetegn som kan være opptil 2 gigabyte lang (1 073 741 823 dobbeltbytetegn). En DBCLOB brukes til å lagre store data med DBCS-data, for eksempel dokumenter. En DBCLOB blir sett på som en grafisk streng.

### Manipulere store objekter (LOB)

Siden LOB-verdier kan være veldig store, kan det ta lang tid å overføre dem fra databasetjeneren til klientapplikasjonen. Vanligvis blir LOB-verdier behandlet en av gangen, i stedet for som en enhet. I de tilfellene der en

applikasjon ikke trenger å (eller vil) lagre hele LOB-verdien i applikasjonsminnet, kan den referere til denne verdien gjennom en *LOB-posisjonsviser*variabel.

Etterfølgende setninger kan deretter bruke posisjonsviserne til å utføre operasjoner på data uten nødvendigvis å hente hele LOBen. Posisjonsviservariabler brukes til å redusere lagringskrav for applikasjonene og forbedre ytelsen ved å redusere datastrømmen mellom klienten og tjeneren.

En annen mekanisme er *filreferansevariabler*. De brukes til å hente store objekter direkte til en fil eller til å oppdatere store objekter i en tabell direkte fra en fil. Filreferansevariabler brukes til å redusere lagringskrav for applikasjonene siden de ikke trenger å lagre LOB-dataene. Du finner flere opplysninger i *Application Development Guide* og *SQL Reference*.

## Spesialregistre

Et *spesialregister* er et lagringsområde som er definert for en tilkobling av databasesystemet, og brukes til å lagre opplysninger som det kan refereres til i SQL-setninger. Nedenfor finner du noen få eksempler på de mest brukte spesialregistrene. Du finner en liste over alle spesialregistrene og detaljerte opplysninger i *SQL Reference*.

- CURRENT DATE: inneholder datoen i samsvar med klokkeslettet ved utføringstidspunktet for SQL-setningen.
- CURRENT FUNCTION PATH: inneholder en verdi som spesifiserer funksjonsbanen som brukes til å behandle funksjons- og datatypereferanser.
- CURRENT SERVER: spesifiserer den gjeldende applikasjonstjeneren
- CURRENT TIME: inneholder klokkeslettet i samsvar med klokkeslettet ved utføringstidspunktet for SQL-setningen.
- CURRENT TIMESTAMP: spesifiserer en systemtid i samsvar med klokkeslettet ved utføringstidspunktet for SQL-setningen.
- CURRENT TIMEZONE: spesifiserer forskjellen mellom C.U.T (Coordinated Universal Time) og lokal tid på applikasjonstjeneren.
- USER: spesifiserer autorisasjons-IDen for kjøretid.

Du kan vise innholdet i et spesialregister med VALUES-setningen. For eksempel:

```
VALUES (CURRENT TIMESTAMP)
```

Du kan også bruke:

```
SELECT CURRENT TIMESTAMP FROM ORG
```

og dette returnerer TIMESTAMP for hver rad i tabellen.

## Innledning til katalogoversikter

DB2 oppretter og vedlikeholder et omfattende sett med systemkatalogtabeller for hver database. Disse tabellene inneholder opplysninger om den logiske og fysiske strukturen til databaseobjekter, for eksempel tabeller, utsnitt, pakker, referanseintegritetsforhold, funksjoner, distinkte typer og utløsere. De opprettes når databasen blir opprettet og de blir oppdatert i løpet av normal drift. Du kan ikke opprette eller slette dem eksplisitt, men du kan spørre i dem og se på innholdet.

Du finner flere opplysninger i *SQL Reference*.

### Velge rader fra systemkataloger

Katalogoversikter er som et hvilket som helst databaseutsnitt. Du kan bruke SQL-setninger til å se på dataene, på akkurat samme måte som du ville gjort for et hvilket som helst annet utsnitt i systemet.

Du kan finne nyttige opplysninger om tabellene i katalogen SYSCAT.TABLES. For å finne navnet på eksisterende tabeller som du har opprettet, gir du en setning som likner på den nedenfor:

```
SELECT TABNAME, TYPE, CREATE_TIME
   FROM SYSCAT.TABLES
   WHERE DEFINER = USER
```

Denne setningen gir dette resultatet:

```
TABNAME            TYPE CREATE_TIME
------------------ ---- --------------------------
ORG                T    1999-07-21-13.42.55.128005
STAFF              T    1999-07-21-13.42.55.609001
DEPARTMENT         T    1999-07-21-13.42.56.069001
EMPLOYEE           T    1999-07-21-13.42.56.310001
EMP_ACT            T    1999-07-21-13.42.56.710001
PROJECT            T    1999-07-21-13.42.57.051001
EMP_PHOTO          T    1999-07-21-13.42.57.361001
EMP_RESUME         T    1999-07-21-13.42.59.154001
SALES              T    1999-07-21-13.42.59.855001
CL_SCHED           T    1999-07-21-13.43.00.025002
IN_TRAY            T    1999-07-21-13.43.00.055001
```

Listen nedenfor inkluderer katalogoversikter som gjelder for emner som er beskrevet i denne boken. Det finnes mange andre katalogoversikter og de er oppført i brukerhåndbøkene *SQL Reference* og *Administration Guide*.

| Beskrivelse | Katalogoversikt |
|---|---|
| kontrollbegrensninger | SYSCAT.CHECKS |
| kolonner | SYSCAT.COLUMNS |

| Beskrivelse | Katalogoversikt |
|---|---|
| kolonner som er referert til av kontrollbegrensninger | SYSCAT.COLCHECKS |
| kolonner som brukes i nøkler | SYSCAT.KEYCOLUSE |
| datatyper | SYSCAT.DATATYPES |
| funksjonsparametere eller resultat av en funksjon | SYSCAT.FUNCPARMS |
| referansebegrensninger | SYSCAT.REFERENCES |
| skjemaer | SYSCAT.SCHEMATA |
| tabellbegrensninger | SYSCAT.TABCONST |
| tabeller | SYSCAT.TABLES |
| utløsere | SYSCAT.TRIGGERS |
| brukerdefinerte funksjoner | SYSCAT.FUNCTIONS |
| oversikter | SYSCAT.VIEWS |

# Tillegg A. Eksempeldatabasetabeller

Dette vedlegget viser opplysningene som finnes i eksempeltabellene til eksempeldatabasen SAMPLE, og hvordan du oppretter og fjerner dem.

Det finnes flere eksempeldatabaser i DB2 Universal Database som brukes til å vise business intelligence-funksjoner, og de blir brukt i opplæringen i business intelligence. Det er imidlertid bare innholdet i eksempeldatabasen SAMPLE som er beskrevet i dette vedlegget. Du finner flere opplysninger om eksempeldatabasen for business intelligence i *Data Warehouse Center Administration Guide.*

Eksempeltabellene brukes i eksemplene som blir vist i denne brukerhåndboken og i andre brukerhåndbøker i dette biblioteket. I tillegg blir dataene som finnes i eksempelfilene med BLOB- og CLOB-datatyper, vist.

Avsnittene nedenfor er inkludert i dette vedlegget:.


“Eksempeldatabasen” på side 72
“Opprette eksempeldatabasen” på side 72
“Slette eksempeldatabasen” på side 72
“CL_SCHED Table” på side 72
“DEPARTMENT Table” på side 73
“EMPLOYEE Table” på side 73
“EMP_ACT Table” på side 76
“EMP_PHOTO Table” på side 78
“EMP_RESUME Table” på side 78
“IN_TRAY Table” på side 79
“ORG Table” på side 79
“PROJECT Table” på side 80
“SALES Table” på side 81
“STAFF Table” på side 82
“STAFFG Table” på side 83
“Eksempelfiler med BLOB- og CLOB-datatyper” på side 84
“Quintana Photo” på side 84
“Quintana Resume” på side 84
“Nicholls Photo” på side 85
“Nicholls Resume” på side 86
“Adamson Photo” på side 87
“Adamson Resume” på side 87
“Walker Photo” på side 88
“Walker Resume” på side 89.


I eksempeltabellene betyr en tankestrek (-) en nullverdi.

## Eksempeldatabasen

Eksemplene i denne boken bruker en eksempeldatabase. For å kunne bruke disse eksemplene må du opprette SAMPLE-databasen. Databasesystemet må være installert for at du skal kunne bruke den.

### Opprette eksempeldatabasen

En utførbar fil oppretter eksempeldatabasen.[2] For å opprette en database må du ha SYSADM-autorisasjon.

- **Når du bruker UNIX-baserte plattformer**

  Hvis du bruker klarmeldingen i operativsystemet, skriver du:

  ```
  sqllib/bin/db2sampl <path>
  ```

  fra privatkatalogen til databasesystemet som eier forekomsten, der *bane* er en valgfri parameter som oppgir banen der eksempeldatabasen skal opprettes. Trykk på Enter.[3] Skjemaet for DB2SAMPL er CURRENT SCHEMA-spesialregisterverdien.

- **Når du bruker OS/2- eller Windows-plattformer**

  Hvis du bruker klarmeldingen i operativsystemet, skriver du:

  ```
  db2sampl e
  ```

  der *e* er en valgfri parameter som oppgir stasjonen der databasen skal opprettes. Trykk på Enter.[4]

  Hvis du ikke er pålogget på arbeidsstasjonen via Brukerprofiler, blir du bedt om å logge deg på.

### Slette eksempeldatabasen

Hvis du ikke trenger tilgang til eksempeldatabasen, kan du slette den ved å bruke kommandoen DROP DATABASE:

```
db2 drop database sample
```

### CL_SCHED Table

| Name: | CLASS_CODE | DAG | STARTING | ENDING |
|---|---|---|---|---|
| Type: | char(7) | smallint | time | time |

2. Du finner opplysninger som knytter seg til denne kommandoen under DB2SAMPL-kommandoen i *Command Reference*.

3. Hvis baneparameteren ikke er oppgitt, blir eksempeldatabasen opprettet i standardbanen som er oppgitt av DFTDBPATH-parameteren i konfigurasjonsfilen til databasesystemet.

4. Hvis stasjonsparameteren ikke er oppgitt, blir eksempeldatabasen opprettet på den samme stasjonen som DB2.

| Name: | CLASS_CODE | DAG | STARTING | ENDING |
|---|---|---|---|---|
| Beskr: | Klassekode (rom:lærer) | Dag # av 4-dagers planen | Startklokkeslett | Sluttklokkeslett |

## DEPARTMENT Table

| Name: | DEPTNO | DEPTNAME | MGRNO | ADMRDEPT | LOCATION |
|---|---|---|---|---|---|
| Type: | char(3) not null | varchar(29) not null | char(6) | char(3) not null | char(16) |
| Desc: | Avdelingsnr. | Navn som beskriver generelle aktiviteter i avdelingen | Funksjonærnr. (EMPNO) til avdelingssjef | Avdeling (DEPTNO) som denne avdelingen rapporterer til | Navn på fjerntliggende sted |
| Values: | A00 | SPIFFY COMPUTER SERVICE DIV. | 000010 | A00 | - |
| | B01 | PLANNING | 000020 | A00 | - |
| | C01 | INFORMATION CENTER | 000030 | A00 | - |
| | D01 | DEVELOPMENT CENTER | - | A00 | - |
| | D11 | MANUFACTURING SYSTEMS | 000060 | D01 | - |
| | D21 | ADMINISTRATION SYSTEMS | 000070 | D01 | - |
| | E01 | SUPPORT SERVICES | 000050 | A00 | - |
| | E11 | OPERATIONS | 000090 | E01 | - |
| | E21 | SOFTWARE SUPPORT | 000100 | E01 | - |

## EMPLOYEE Table

| Names: | EMPNO | FIRSTNME | MIDINIT | LASTNAME | WORKDEPT | PHONENO | HIREDATE |
|---|---|---|---|---|---|---|---|
| Type: | char(6) not null | varchar(12) not null | char(1) not null | varchar(15) not null | char(3) | char(4) | date |
| Desc: | Funksjonærnr. | Fornavn | Mellomnavn | Etternavn | Avdeling (DEPTNO) der den ansatte jobber | Tlf.nr. | Ansettelsesdato |

| JOB | EDLEVEL | SEX | BIRTHDATE | SALARY | BONUS | COMM |
|---|---|---|---|---|---|---|
| char(8) | smallint not null | char(1) | date | dec(9,2) | dec(9,2) | dec(9,2) |
| Job | Number of years of formal education | Sex (M male, F female) | Date of birth | Yearly salary | Yearly bonus | Yearly commission |

Du finner verdiene i EMPLOYEE-tabellen på neste side.

| EMPNO | FIRSTNME | MID INIT | LASTNAME | WORK DEPT | PHONE NO | HIREDATE | JOB | ED LEVEL | SEX | BIRTHDATE | SALARY | BONUS | COMM |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| char(6) not null | varchar(12) not null | char(1) not null | varchar(15) not null | char(3) | char(4) | date | char(8) | smallint not null | char(1) | date | dec(9,2) | dec(9,2) | dec(9,2) |
| 000010 | CHRISTINE | I | HAAS | A00 | 3978 | 1965-01-01 | PRES | 18 | F | 1933-08-24 | 52750 | 1000 | 4220 |
| 000020 | MICHAEL | L | THOMPSON | B01 | 3476 | 1973-10-10 | MANAGER | 18 | M | 1948-02-02 | 41250 | 800 | 3300 |
| 000030 | SALLY | A | KWAN | C01 | 4738 | 1975-04-05 | MANAGER | 20 | F | 1941-05-11 | 38250 | 800 | 3060 |
| 000050 | JOHN | B | GEYER | E01 | 6789 | 1949-08-17 | MANAGER | 16 | M | 1925-09-15 | 40175 | 800 | 3214 |
| 000060 | IRVING | F | STERN | D11 | 6423 | 1973-09-14 | MANAGER | 16 | M | 1945-07-07 | 32250 | 500 | 2580 |
| 000070 | EVA | B | PULASKI | D21 | 7831 | 1980-09-30 | MANAGER | 16 | F | 1953-05-26 | 36170 | 700 | 2893 |
| 000090 | EILEEN | W | HENDERSON | E11 | 5498 | 1970-08-15 | MANAGER | 16 | F | 1941-05-15 | 29750 | 600 | 2380 |
| 000100 | THEODORE | Q | SPENSER | E21 | 0972 | 1980-06-19 | MANAGER | 14 | M | 1956-12-18 | 26150 | 500 | 2092 |
| 000110 | VINCENZO | G | LUCCHESSI | A00 | 3490 | 1958-05-16 | SALESREP | 19 | M | 1929-11-05 | 46500 | 900 | 3720 |
| 000120 | SEAN |  | O'CONNELL | A00 | 2167 | 1963-12-05 | CLERK | 14 | M | 1942-10-18 | 29250 | 600 | 2340 |
| 000130 | DOLORES | M | QUINTANA | C01 | 4578 | 1971-07-28 | ANALYST | 16 | F | 1925-09-15 | 23800 | 500 | 1904 |
| 000140 | HEATHER | A | NICHOLLS | C01 | 1793 | 1976-12-15 | ANALYST | 18 | F | 1946-01-19 | 28420 | 600 | 2274 |
| 000150 | BRUCE |  | ADAMSON | D11 | 4510 | 1972-02-12 | DESIGNER | 16 | M | 1947-05-17 | 25280 | 500 | 2022 |
| 000160 | ELIZABETH | R | PIANKA | D11 | 3782 | 1977-10-11 | DESIGNER | 17 | F | 1955-04-12 | 22250 | 400 | 1780 |
| 000170 | MASATOSHI | J | YOSHIMURA | D11 | 2890 | 1978-09-15 | DESIGNER | 16 | M | 1951-01-05 | 24680 | 500 | 1974 |
| 000180 | MARILYN | S | SCOUTTEN | D11 | 1682 | 1973-07-07 | DESIGNER | 17 | F | 1949-02-21 | 21340 | 500 | 1707 |
| 000190 | JAMES | H | WALKER | D11 | 2986 | 1974-07-26 | DESIGNER | 16 | M | 1952-06-25 | 20450 | 400 | 1636 |
| 000200 | DAVID |  | BROWN | D11 | 4501 | 1966-03-03 | DESIGNER | 16 | M | 1941-05-29 | 27740 | 600 | 2217 |
| 000210 | WILLIAM | T | JONES | D11 | 0942 | 1979-04-11 | DESIGNER | 17 | M | 1953-02-23 | 18270 | 400 | 1462 |
| 000220 | JENNIFER | K | LUTZ | D11 | 0672 | 1968-08-29 | DESIGNER | 18 | F | 1948-03-19 | 29840 | 600 | 2387 |
| 000230 | JAMES | J | JEFFERSON | D21 | 2094 | 1966-11-21 | CLERK | 14 | M | 1935-05-30 | 22180 | 400 | 1774 |
| 000240 | SALVATORE | M | MARINO | D21 | 3780 | 1979-12-05 | CLERK | 17 | M | 1954-03-31 | 28760 | 600 | 2301 |
| 000250 | DANIEL | S | SMITH | D21 | 0961 | 1969-10-30 | CLERK | 15 | M | 1939-11-12 | 19180 | 400 | 1534 |
| 000260 | SYBIL | P | JOHNSON | D21 | 8953 | 1975-09-11 | CLERK | 16 | F | 1936-10-05 | 17250 | 300 | 1380 |
| 000270 | MARIA | L | PEREZ | D21 | 9001 | 1980-09-30 | CLERK | 15 | F | 1953-05-26 | 27380 | 500 | 2190 |
| 000280 | ETHEL | R | SCHNEIDER | E11 | 8997 | 1967-03-24 | OPERATOR | 17 | F | 1936-03-28 | 26250 | 500 | 2100 |
| 000290 | JOHN | R | PARKER | E11 | 4502 | 1980-05-30 | OPERATOR | 12 | M | 1946-07-09 | 15340 | 300 | 1227 |
| 000300 | PHILIP | X | SMITH | E11 | 2095 | 1972-06-19 | OPERATOR | 14 | M | 1936-10-27 | 17750 | 400 | 1420 |
| 000310 | MAUDE | F | SETRIGHT | E11 | 3332 | 1964-09-12 | OPERATOR | 12 | F | 1931-04-21 | 15900 | 300 | 1272 |
| 000320 | RAMLAL | V | MEHTA | E21 | 9990 | 1965-07-07 | FIELDREP | 16 | M | 1932-08-11 | 19950 | 400 | 1596 |

| EMPNO | FIRSTNME | MID INIT | LASTNAME | WORK DEPT | PHONE NO | HIREDATE | JOB | ED LEVEL | SEX | BIRTHDATE | SALARY | BONUS | COMM |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 000330 | WING | | LEE | E21 | 2103 | 1976-02-23 | FIELDREP | 14 | M | 1941-07-18 | 25370 | 500 | 2030 |
| 000340 | JASON | R | GOUNOT | E21 | 5698 | 1947-05-05 | FIELDREP | 16 | M | 1926-05-17 | 23840 | 500 | 1907 |

## EMP_ACT Table

| Name: | EMPNO | PROJNO | ACTNO | EMPTIME | EMSTDATE | EMENDATE |
|---|---|---|---|---|---|---|
| Type: | char(6) not null | char(6) not null | smallint not null | dec(5,2) | date | date |
| Desc: | Funksjonærnr. | Prosjektnr. | Aktivitetsnr. | Proporsjon av den ansattes tid som er brukt på prosjektet | Dato aktivitet begynner | Dato aktivitet slutter |
| Values: | 000010 | AD3100 | 10 | .50 | 1982-01-01 | 1982-07-01 |
| | 000070 | AD3110 | 10 | 1.00 | 1982-01-01 | 1983-02-01 |
| | 000230 | AD3111 | 60 | 1.00 | 1982-01-01 | 1982-03-15 |
| | 000230 | AD3111 | 60 | .50 | 1982-03-15 | 1982-04-15 |
| | 000230 | AD3111 | 70 | .50 | 1982-03-15 | 1982-10-15 |
| | 000230 | AD3111 | 80 | .50 | 1982-04-15 | 1982-10-15 |
| | 000230 | AD3111 | 180 | 1.00 | 1982-10-15 | 1983-01-01 |
| | 000240 | AD3111 | 70 | 1.00 | 1982-02-15 | 1982-09-15 |
| | 000240 | AD3111 | 80 | 1.00 | 1982-09-15 | 1983-01-01 |
| | 000250 | AD3112 | 60 | 1.00 | 1982-01-01 | 1982-02-01 |
| | 000250 | AD3112 | 60 | .50 | 1982-02-01 | 1982-03-15 |
| | 000250 | AD3112 | 60 | .50 | 1982-12-01 | 1983-01-01 |
| | 000250 | AD3112 | 60 | 1.00 | 1983-01-01 | 1983-02-01 |
| | 000250 | AD3112 | 70 | .50 | 1982-02-01 | 1982-03-15 |
| | 000250 | AD3112 | 70 | 1.00 | 1982-03-15 | 1982-08-15 |
| | 000250 | AD3112 | 70 | .25 | 1982-08-15 | 1982-10-15 |
| | 000250 | AD3112 | 80 | .25 | 1982-08-15 | 1982-10-15 |
| | 000250 | AD3112 | 80 | .50 | 1982-10-15 | 1982-12-01 |
| | 000250 | AD3112 | 180 | .50 | 1982-08-15 | 1983-01-01 |
| | 000260 | AD3113 | 70 | .50 | 1982-06-15 | 1982-07-01 |
| | 000260 | AD3113 | 70 | 1.00 | 1982-07-01 | 1983-02-01 |
| | 000260 | AD3113 | 80 | 1.00 | 1982-01-01 | 1982-03-01 |
| | 000260 | AD3113 | 80 | .50 | 1982-03-01 | 1982-04-15 |
| | 000260 | AD3113 | 180 | .50 | 1982-03-01 | 1982-04-15 |
| | 000260 | AD3113 | 180 | 1.00 | 1982-04-15 | 1982-06-01 |
| | 000260 | AD3113 | 180 | .50 | 1982-06-01 | 1982-07-01 |
| | 000270 | AD3113 | 60 | .50 | 1982-03-01 | 1982-04-01 |
| | 000270 | AD3113 | 60 | 1.00 | 1982-04-01 | 1982-09-01 |
| | 000270 | AD3113 | 60 | .25 | 1982-09-01 | 1982-10-15 |
| | 000270 | AD3113 | 70 | .75 | 1982-09-01 | 1982-10-15 |

| Name: | EMPNO | PROJNO | ACTNO | EMPTIME | EMSTDATE | EMENDATE |
|---|---|---|---|---|---|---|
| | 000270 | AD3113 | 70 | 1.00 | 1982-10-15 | 1983-02-01 |
| | 000270 | AD3113 | 80 | 1.00 | 1982-01-01 | 1982-03-01 |
| | 000270 | AD3113 | 80 | .50 | 1982-03-01 | 1982-04-01 |
| | 000030 | IF1000 | 10 | .50 | 1982-06-01 | 1983-01-01 |
| | 000130 | IF1000 | 90 | 1.00 | 1982-01-01 | 1982-10-01 |
| | 000130 | IF1000 | 100 | .50 | 1982-10-01 | 1983-01-01 |
| | 000140 | IF1000 | 90 | .50 | 1982-10-01 | 1983-01-01 |
| | 000030 | IF2000 | 10 | .50 | 1982-01-01 | 1983-01-01 |
| | 000140 | IF2000 | 100 | 1.00 | 1982-01-01 | 1982-03-01 |
| | 000140 | IF2000 | 100 | .50 | 1982-03-01 | 1982-07-01 |
| | 000140 | IF2000 | 110 | .50 | 1982-03-01 | 1982-07-01 |
| | 000140 | IF2000 | 110 | .50 | 1982-10-01 | 1983-01-01 |
| | 000010 | MA2100 | 10 | .50 | 1982-01-01 | 1982-11-01 |
| | 000110 | MA2100 | 20 | 1.00 | 1982-01-01 | 1982-03-01 |
| | 000010 | MA2110 | 10 | 1.00 | 1982-01-01 | 1983-02-01 |
| | 000200 | MA2111 | 50 | 1.00 | 1982-01-01 | 1982-06-15 |
| | 000200 | MA2111 | 60 | 1.00 | 1982-06-15 | 1983-02-01 |
| | 000220 | MA2111 | 40 | 1.00 | 1982-01-01 | 1983-02-01 |
| | 000150 | MA2112 | 60 | 1.00 | 1982-01-01 | 1982-07-15 |
| | 000150 | MA2112 | 180 | 1.00 | 1982-07-15 | 1983-02-01 |
| | 000170 | MA2112 | 60 | 1.00 | 1982-01-01 | 1983-06-01 |
| | 000170 | MA2112 | 70 | 1.00 | 1982-06-01 | 1983-02-01 |
| | 000190 | MA2112 | 70 | 1.00 | 1982-02-01 | 1982-10-01 |
| | 000190 | MA2112 | 80 | 1.00 | 1982-10-01 | 1983-10-01 |
| | 000160 | MA2113 | 60 | 1.00 | 1982-07-15 | 1983-02-01 |
| | 000170 | MA2113 | 80 | 1.00 | 1982-01-01 | 1983-02-01 |
| | 000180 | MA2113 | 70 | 1.00 | 1982-04-01 | 1982-06-15 |
| | 000210 | MA2113 | 80 | .50 | 1982-10-01 | 1983-02-01 |
| | 000210 | MA2113 | 180 | .50 | 1982-10-01 | 1983-02-01 |
| | 000050 | OP1000 | 10 | .25 | 1982-01-01 | 1983-02-01 |
| | 000090 | OP1010 | 10 | 1.00 | 1982-01-01 | 1983-02-01 |
| | 000280 | OP1010 | 130 | 1.00 | 1982-01-01 | 1983-02-01 |
| | 000290 | OP1010 | 130 | 1.00 | 1982-01-01 | 1983-02-01 |
| | 000300 | OP1010 | 130 | 1.00 | 1982-01-01 | 1983-02-01 |
| | 000310 | OP1010 | 130 | 1.00 | 1982-01-01 | 1983-02-01 |
| | 000050 | OP2010 | 10 | .75 | 1982-01-01 | 1983-02-01 |
| | 000100 | OP2010 | 10 | 1.00 | 1982-01-01 | 1983-02-01 |

| Name: | EMPNO | PROJNO | ACTNO | EMPTIME | EMSTDATE | EMENDATE |
|---|---|---|---|---|---|---|
| | 000320 | OP2011 | 140 | .75 | 1982-01-01 | 1983-02-01 |
| | 000320 | OP2011 | 150 | .25 | 1982-01-01 | 1983-02-01 |
| | 000330 | OP2012 | 140 | .25 | 1982-01-01 | 1983-02-01 |
| | 000330 | OP2012 | 160 | .75 | 1982-01-01 | 1983-02-01 |
| | 000340 | OP2013 | 140 | .50 | 1982-01-01 | 1983-02-01 |
| | 000340 | OP2013 | 170 | .50 | 1982-01-01 | 1983-02-01 |
| | 000020 | PL2100 | 30 | 1.00 | 1982-01-01 | 1982-09-15 |

### EMP_PHOTO Table

| Name: | EMPNO | PHOTO_FORMAT | PICTURE |
|---|---|---|---|
| Type: | char(6) not null | varchar(10) not null | blob(100k) |
| Desc: | Funksjonærnr. | Bildeformat | Bilde av arbeidstaker |
| Values: | 000130 | bitmap | db200130.bmp |
| | 000130 | gif | db200130.gif |
| | 000130 | xwd | db200130.xwd |
| | 000140 | bitmap | db200140.bmp |
| | 000140 | gif | db200140.gif |
| | 000140 | xwd | db200140.xwd |
| | 000150 | bitmap | db200150.bmp |
| | 000150 | gif | db200150.gif |
| | 000150 | xwd | db200150.xwd |
| | 000190 | bitmap | db200190.bmp |
| | 000190 | gif | db200190.gif |
| | 000190 | xwd | db200190.xwd |

- “Quintana Photo” på side 84 viser et bilde av arbeidstakeren Delores Quintana.
- “Nicholls Photo” på side 85 viser et bilde av arbeidstakeren Heather Nicholls.
- “Adamson Photo” på side 87 viser et bilde av arbeidstakeren Bruce Adamson.
- “Walker Photo” på side 88 viser et bilde av arbeidstakeren James Walker.

### EMP_RESUME Table

| Name: | EMPNO | RESUME_FORMAT | RESUME |
|---|---|---|---|
| Type: | char(6) not null | varchar(10) not null | clob(5k) |
| Desc: | Funksjonærnr. | CV-format | CV for arbeidstaker |
| Values: | 000130 | ascii | db200130.asc |

| Name: | EMPNO | RESUME_FORMAT | RESUME |
|---|---|---|---|
| | 000130 | script | db200130.scr |
| | 000140 | ascii | db200140.asc |
| | 000140 | script | db200140.scr |
| | 000150 | ascii | db200150.asc |
| | 000150 | script | db200150.scr |
| | 000190 | ascii | db200190.asc |
| | 000190 | script | db200190.scr |

- “Quintana Resume” på side 84 viser CVen til arbeidstakeren Delores Quintana.
- “Nicholls Resume” på side 86 viser CVen til arbeidstakeren Heather Nicholls.
- “Adamson Resume” på side 87 viser CVen til arbeidstakeren Bruce Adamson.
- “Walker Resume” på side 89 viser CVen til arbeidstakeren James Walker.

### IN_TRAY Table

| Name: | RECEIVED | SOURCE | SUBJECT | NOTE_TEXT |
|---|---|---|---|---|
| Type: | timestamp | char(8) | char(64) | varchar(3000) |
| Desc: | Dato og klokkeslett mottatt | Bruker-ID for personen som sender meldingen | Kort beskrivelse | Melding |

### ORG Table

| Name: | DEPTNUMB | DEPTNAME | MANAGER | DIVISION | LOCATION |
|---|---|---|---|---|---|
| Type: | smallint not null | varchar(14) | smallint | varchar(10) | varchar(13) |
| Desc: | Avdelingsnr. | Avdelingsnavn | Avdelingssjefnr. | Firmadivisjon | By |
| Values: | 10 | Head Office | 160 | Corporate | New York |
| | 15 | New England | 50 | Eastern | Boston |
| | 20 | Mid Atlantic | 10 | Eastern | Washington |
| | 38 | South Atlantic | 30 | Eastern | Atlanta |
| | 42 | Great Lakes | 100 | Midwest | Chicago |
| | 51 | Plains | 140 | Midwest | Dallas |
| | 66 | Pacific | 270 | Western | San Francisco |
| | 84 | Mountain | 290 | Western | Denver |

## PROJECT Table

| Name: | PROJNO | PROJNAME | DEPTNO | RESPEMP | PRSTAFF | PRSTDATE | PRENDATE | MAJPROJ |
|---|---|---|---|---|---|---|---|---|
| Type: | char(6) not null | varchar(24) not null | char(3) not null | char(6) not null | dec(5,2) | date | date | char(6) |
| Desc: | Prosjektnr. | Prosjektnavn | Avdeling ansvarlig | Ansatt ansvarlig | Beregnet gjennomsnitt av ansatte | Beregnet startdato | Beregnet sluttdato | Hovedprosjekt, for et delprosjekt |
| Values: | AD3100 | ADMIN SERVICES | D01 | 000010 | 6.5 | 1982-01-01 | 1983-02-01 | - |
| | AD3110 | GENERAL ADMIN SYSTEMS | D21 | 000070 | 6 | 1982-01-01 | 1983-02-01 | AD3100 |
| | AD3111 | PAYROLL PROGRAMMING | D21 | 000230 | 2 | 1982-01-01 | 1983-02-01 | AD3110 |
| | AD3112 | PERSONNEL PROGRAMMING | D21 | 000250 | 1 | 1982-01-01 | 1983-02-01 | AD3110 |
| | AD3113 | ACCOUNT PROGRAMMING | D21 | 000270 | 2 | 1982-01-01 | 1983-02-01 | AD3110 |
| | IF1000 | QUERY SERVICES | C01 | 000030 | 2 | 1982-01-01 | 1983-02-01 | - |
| | IF2000 | USER EDUCATION | C01 | 000030 | 1 | 1982-01-01 | 1983-02-01 | - |
| | MA2100 | WELD LINE AUTOMATION | D01 | 000010 | 12 | 1982-01-01 | 1983-02-01 | - |
| | MA2110 | W L PROGRAMMING | D11 | 000060 | 9 | 1982-01-01 | 1983-02-01 | MA2100 |
| | MA2111 | W L PROGRAM DESIGN | D11 | 000220 | 2 | 1982-01-01 | 1982-12-01 | MA2110 |
| | MA2112 | W L ROBOT DESIGN | D11 | 000150 | 3 | 1982-01-01 | 1982-12-01 | MA2110 |
| | MA2113 | W L PROD CONT PROGS | D11 | 000160 | 3 | 1982-02-15 | 1982-12-01 | MA2110 |
| | OP1000 | OPERATION SUPPORT | E01 | 000050 | 6 | 1982-01-01 | 1983-02-01 | - |
| | OP1010 | OPERATION | E11 | 000090 | 5 | 1982-01-01 | 1983-02-01 | OP1000 |
| | OP2000 | GEN SYSTEMS SERVICES | E01 | 000050 | 5 | 1982-01-01 | 1983-02-01 | - |
| | OP2010 | SYSTEMS SUPPORT | E21 | 000100 | 4 | 1982-01-01 | 1983-02-01 | OP2000 |
| | OP2011 | SCP SYSTEMS SUPPORT | E21 | 000320 | 1 | 1982-01-01 | 1983-02-01 | OP2010 |
| | OP2012 | APPLICATIONS SUPPORT | E21 | 000330 | 1 | 1982-01-01 | 1983-02-01 | OP2010 |
| | OP2013 | DB/DC SUPPORT | E21 | 000340 | 1 | 1982-01-01 | 1983-02-01 | OP2010 |
| | PL2100 | WELD LINE PLANNING | B01 | 000020 | 1 | 1982-01-01 | 1982-09-15 | MA2100 |

## SALES Table

| Name: | SALES_DATE | SALES_PERSON | REGION | SALES |
|---|---|---|---|---|
| Type: | date | varchar(15) | varchar(15) | int |
| Desc: | Salgsdato | Etternavn til ansatt | Salgsregion | Antall salg |
| Values: | 12/31/1995 | LUCCHESSI | Ontario-South | 1 |
| | 12/31/1995 | LEE | Ontario-South | 3 |
| | 12/31/1995 | LEE | Quebec | 1 |
| | 12/31/1995 | LEE | Manitoba | 2 |
| | 12/31/1995 | GOUNOT | Quebec | 1 |
| | 03/29/1996 | LUCCHESSI | Ontario-South | 3 |
| | 03/29/1996 | LUCCHESSI | Quebec | 1 |
| | 03/29/1996 | LEE | Ontario-South | 2 |
| | 03/29/1996 | LEE | Ontario-North | 2 |
| | 03/29/1996 | LEE | Quebec | 3 |
| | 03/29/1996 | LEE | Manitoba | 5 |
| | 03/29/1996 | GOUNOT | Ontario-South | 3 |
| | 03/29/1996 | GOUNOT | Quebec | 1 |
| | 03/29/1996 | GOUNOT | Manitoba | 7 |
| | 03/30/1996 | LUCCHESSI | Ontario-South | 1 |
| | 03/30/1996 | LUCCHESSI | Quebec | 2 |
| | 03/30/1996 | LUCCHESSI | Manitoba | 1 |
| | 03/30/1996 | LEE | Ontario-South | 7 |
| | 03/30/1996 | LEE | Ontario-North | 3 |
| | 03/30/1996 | LEE | Quebec | 7 |
| | 03/30/1996 | LEE | Manitoba | 4 |
| | 03/30/1996 | GOUNOT | Ontario-South | 2 |
| | 03/30/1996 | GOUNOT | Quebec | 18 |
| | 03/30/1996 | GOUNOT | Manitoba | 1 |
| | 03/31/1996 | LUCCHESSI | Manitoba | 1 |
| | 03/31/1996 | LEE | Ontario-South | 14 |
| | 03/31/1996 | LEE | Ontario-North | 3 |
| | 03/31/1996 | LEE | Quebec | 7 |
| | 03/31/1996 | LEE | Manitoba | 3 |
| | 03/31/1996 | GOUNOT | Ontario-South | 2 |
| | 03/31/1996 | GOUNOT | Quebec | 1 |
| | 04/01/1996 | LUCCHESSI | Ontario-South | 3 |
| | 04/01/1996 | LUCCHESSI | Manitoba | 1 |
| | 04/01/1996 | LEE | Ontario-South | 8 |
| | 04/01/1996 | LEE | Ontario-North | - |
| | 04/01/1996 | LEE | Quebec | 8 |
| | 04/01/1996 | LEE | Manitoba | 9 |
| | 04/01/1996 | GOUNOT | Ontario-South | 3 |

| Name: | SALES_DATE | SALES_PERSON | REGION | SALES |
|---|---|---|---|---|
| | 04/01/1996 | GOUNOT | Ontario-North | 1 |
| | 04/01/1996 | GOUNOT | Quebec | 3 |
| | 04/01/1996 | GOUNOT | Manitoba | 7 |

## STAFF Table

| Name: | ID | NAME | DEPT | JOB | YEARS | SALARY | COMM |
|---|---|---|---|---|---|---|---|
| Type: | smallint not null | varchar(9) | smallint | char(5) | smallint | dec(7,2) | dec(7,2) |
| Desc: | Funksjonærnr. | Arbeidstaker-navn | Avdelingsnr. | Jobbtype | Antall år ansatt | Gjeldende lønn | Kommisjon |
| Values: | 10 | Sanders | 20 | Mgr | 7 | 18357.50 | - |
| | 20 | Pernal | 20 | Sales | 8 | 18171.25 | 612.45 |
| | 30 | Marenghi | 38 | Mgr | 5 | 17506.75 | - |
| | 40 | O'Brien | 38 | Sales | 6 | 18006.00 | 846.55 |
| | 50 | Hanes | 15 | Mgr | 10 | 20659.80 | - |
| | 60 | Quigley | 38 | Sales | - | 16808.30 | 650.25 |
| | 70 | Rothman | 15 | Sales | 7 | 16502.83 | 1152.00 |
| | 80 | James | 20 | Clerk | - | 13504.60 | 128.20 |
| | 90 | Koonitz | 42 | Sales | 6 | 18001.75 | 1386.70 |
| | 100 | Plotz | 42 | Mgr | 7 | 18352.80 | - |
| | 110 | Ngan | 15 | Clerk | 5 | 12508.20 | 206.60 |
| | 120 | Naughton | 38 | Clerk | - | 12954.75 | 180.00 |
| | 130 | Yamaguchi | 42 | Clerk | 6 | 10505.90 | 75.60 |
| | 140 | Fraye | 51 | Mgr | 6 | 21150.00 | - |
| | 150 | Williams | 51 | Sales | 6 | 19456.50 | 637.65 |
| | 160 | Molinare | 10 | Mgr | 7 | 22959.20 | - |
| | 170 | Kermisch | 15 | Clerk | 4 | 12258.50 | 110.10 |
| | 180 | Abrahams | 38 | Clerk | 3 | 12009.75 | 236.50 |
| | 190 | Sneider | 20 | Clerk | 8 | 14252.75 | 126.50 |
| | 200 | Scoutten | 42 | Clerk | - | 11508.60 | 84.20 |
| | 210 | Lu | 10 | Mgr | 10 | 20010.00 | - |
| | 220 | Smith | 51 | Sales | 7 | 17654.50 | 992.80 |
| | 230 | Lundquist | 51 | Clerk | 3 | 13369.80 | 189.65 |
| | 240 | Daniels | 10 | Mgr | 5 | 19260.25 | - |
| | 250 | Wheeler | 51 | Clerk | 6 | 14460.00 | 513.30 |
| | 260 | Jones | 10 | Mgr | 12 | 21234.00 | - |
| | 270 | Lea | 66 | Mgr | 9 | 18555.50 | - |
| | 280 | Wilson | 66 | Sales | 9 | 18674.50 | 811.50 |

| Name: | ID | NAME | DEPT | JOB | YEARS | SALARY | COMM |
|---|---|---|---|---|---|---|---|
| | 290 | Quill | 84 | Mgr | 10 | 19818.00 | - |
| | 300 | Davis | 84 | Sales | 5 | 15454.50 | 806.10 |
| | 310 | Graham | 66 | Sales | 13 | 21000.00 | 200.30 |
| | 320 | Gonzales | 66 | Sales | 4 | 16858.20 | 844.00 |
| | 330 | Burke | 66 | Clerk | 1 | 10988.00 | 55.50 |
| | 340 | Edwards | 84 | Sales | 7 | 17844.00 | 1285.00 |
| | 350 | Gafney | 84 | Clerk | 5 | 13030.50 | 188.00 |

## STAFFG Table

**Merk:** STAFFG blir bare opprettet for dobbeltbytetegnsett.

| Name: | ID | NAME | DEPT | JOB | YEARS | SALARY | COMM |
|---|---|---|---|---|---|---|---|
| Type: | smallint not null | vargraphic(9) | smallint | graphic(5) | smallint | dec(9,0) | dec(9,0) |
| Desc: | Funksjonærnr. | Arbeidstaker-navn | Avdelingsnr. | Jobbtype | Antall år ansatt | Gjeldende lønn | Kommisjon |
| Values: | 10 | Sanders | 20 | Mgr | 7 | 18357.50 | - |
| | 20 | Pernal | 20 | Sales | 8 | 18171.25 | 612.45 |
| | 30 | Marenghi | 38 | Mgr | 5 | 17506.75 | - |
| | 40 | O'Brien | 38 | Sales | 6 | 18006.00 | 846.55 |
| | 50 | Hanes | 15 | Mgr | 10 | 20659.80 | - |
| | 60 | Quigley | 38 | Sales | - | 16808.30 | 650.25 |
| | 70 | Rothman | 15 | Sales | 7 | 16502.83 | 1152.00 |
| | 80 | James | 20 | Clerk | - | 13504.60 | 128.20 |
| | 90 | Koonitz | 42 | Sales | 6 | 18001.75 | 1386.70 |
| | 100 | Plotz | 42 | Mgr | 7 | 18352.80 | - |
| | 110 | Ngan | 15 | Clerk | 5 | 12508.20 | 206.60 |
| | 120 | Naughton | 38 | Clerk | - | 12954.75 | 180.00 |
| | 130 | Yamaguchi | 42 | Clerk | 6 | 10505.90 | 75.60 |
| | 140 | Fraye | 51 | Mgr | 6 | 21150.00 | - |
| | 150 | Williams | 51 | Sales | 6 | 19456.50 | 637.65 |
| | 160 | Molinare | 10 | Mgr | 7 | 22959.20 | - |
| | 170 | Kermisch | 15 | Clerk | 4 | 12258.50 | 110.10 |
| | 180 | Abrahams | 38 | Clerk | 3 | 12009.75 | 236.50 |
| | 190 | Sneider | 20 | Clerk | 8 | 14252.75 | 126.50 |
| | 200 | Scoutten | 42 | Clerk | - | 11508.60 | 84.20 |
| | 210 | Lu | 10 | Mgr | 10 | 20010.00 | - |
| | 220 | Smith | 51 | Sales | 7 | 17654.50 | 992.80 |

## Eksempeldatabasetabeller

| Name: | ID | NAME | DEPT | JOB | YEARS | SALARY | COMM |
|---|---|---|---|---|---|---|---|
| | 230 | Lundquist | 51 | Clerk | 3 | 13369.80 | 189.65 |
| | 240 | Daniels | 10 | Mgr | 5 | 19260.25 | - |
| | 250 | Wheeler | 51 | Clerk | 6 | 14460.00 | 513.30 |
| | 260 | Jones | 10 | Mgr | 12 | 21234.00 | - |
| | 270 | Lea | 66 | Mgr | 9 | 18555.50 | - |
| | 280 | Wilson | 66 | Sales | 9 | 18674.50 | 811.50 |
| | 290 | Quill | 84 | Mgr | 10 | 19818.00 | - |
| | 300 | Davis | 84 | Sales | 5 | 15454.50 | 806.10 |
| | 310 | Graham | 66 | Sales | 13 | 21000.00 | 200.30 |
| | 320 | Gonzales | 66 | Sales | 4 | 16858.20 | 844.00 |
| | 330 | Burke | 66 | Clerk | 1 | 10988.00 | 55.50 |
| | 340 | Edwards | 84 | Sales | 7 | 17844.00 | 1285.00 |
| | 350 | Gafney | 84 | Clerk | 5 | 13030.50 | 188.00 |

## Eksempelfiler med BLOB- og CLOB-datatyper

Dette avsnittet viser dataene i EMP_PHOTO-filene (bilder av de ansatte) og EMP_RESUME-filene (CVene til de ansatte).

### Quintana Photo

*Figur 7. Delores M. Quintana*

### Quintana Resume

Teksten nedenfor ligger i filene db200130.asc og db200130.scr.

**Resume: Delores M. Quintana**

**Personal Information**

| | |
|---|---|
| **Address:** | 1150 Eglinton Ave Mellonville, Idaho 83725 |
| **Phone:** | (208) 555-9933 |
| **Birthdate:** | September 15, 1925 |
| **Sex:** | Female |
| **Marital Status:** | Married |
| **Height:** | 5’2″ |
| **Weight:** | 120 lbs. |

**Department Information**

| | |
|---|---|
| **Employee Number:** | 000130 |
| **Dept Number:** | C01 |
| **Manager:** | Sally Kwan |
| **Position:** | Analyst |
| **Phone:** | (208) 555-4578 |
| **Hire Date:** | 1971-07-28 |

**Education**

| | |
|---|---|
| **1965** | Math and English, B.A. Adelphi University |
| **1960** | Dental Technician Florida Institute of Technology |

**Work History**

| | |
|---|---|
| **10/91 - present** | Advisory Systems Analyst Producing documentation tools for engineering department. |
| **12/85 - 9/91** | Technical Writer Writer, text programmer, and planner. |
| **1/79 - 11/85** | COBOL Payroll Programmer Writing payroll programs for a diesel fuel company. |

**Interests**

- Cooking
- Reading
- Sewing
- Remodeling

## Nicholls Photo

*Figur 8. Heather A. Nicholls*

## Nicholls Resume

Teksten nedenfor ligger i filene db200140.asc og db200140.scr.

**Resume: Heather A. Nicholls**

| | |
|---|---|
| **Personal Information** | |
| **Address:** | 844 Don Mills Ave Mellonville, Idaho 83734 |
| **Phone:** | (208) 555-2310 |
| **Birthdate:** | January 19, 1946 |
| **Sex:** | Female |
| **Marital Status:** | Single |
| **Height:** | 5’8″ |
| **Weight:** | 130 lbs. |
| **Department Information** | |
| **Employee Number:** | 000140 |
| **Dept Number:** | C01 |
| **Manager:** | Sally Kwan |
| **Position:** | Analyst |
| **Phone:** | (208) 555-1793 |
| **Hire Date:** | 1976-12-15 |
| **Education** | |
| **1972** | Computer Engineering, Ph.D. University of Washington |
| **1969** | Music and Physics, M.A. Vassar College |
| **Work History** | |

| | |
|---|---|
| **2/83 - present** | Architect, OCR Development Designing the architecture of OCR products. |
| **12/76 - 1/83** | Text Programmer Optical character recognition (OCR) programming in PL/I. |
| **9/72 - 11/76** | Punch Card Quality Analyst Checking punch cards met quality specifications. |

**Interests**

- Model railroading
- Interior decorating
- Embroidery
- Knitting

## Adamson Photo

*Figur 9. Bruce Adamson*

## Adamson Resume

Teksten nedenfor ligger i filene db200150.asc og db200150.scr.

**Resume: Bruce Adamson**

**Personal Information**

| | |
|---|---|
| **Address:** | 3600 Steeles Ave Mellonville, Idaho 83757 |
| **Phone:** | (208) 555-4489 |
| **Birthdate:** | May 17, 1947 |
| **Sex:** | Male |
| **Marital Status:** | Married |
| **Height:** | 6’0″ |
| **Weight:** | 175 lbs. |

**Department Information**

**Employee Number:** 000150
**Dept Number:** D11
**Manager:** Irving Stern
**Position:** Designer
**Phone:** (208) 555-4510
**Hire Date:** 1972-02-12

**Education**

**1971** Environmental Engineering, M.Sc. Johns Hopkins University

**1968** American History, B.A. Northwestern University

**Work History**

**8/79 - present** Neural Network Design Developing neural networks for machine intelligence products.

**2/72 - 7/79** Robot Vision Development Developing rule-based systems to emulate sight.

**9/71 - 1/72** Numerical Integration Specialist Helping bank systems communicate with each other.

**Interests**

- Racing motorcycles
- Building loudspeakers
- Assembling personal computers
- Sketching

## Walker Photo

*Figur 10. James H. Walker*

## Walker Resume

Teksten nedenfor ligger i filene db200190.asc og db200190.scr.

**Resume: James H. Walker**

**Personal Information**

| | |
|---|---|
| **Address:** | 3500 Steeles Ave Mellonville, Idaho 83757 |
| **Phone:** | (208) 555-7325 |
| **Birthdate:** | June 25, 1952 |
| **Sex:** | Male |
| **Marital Status:** | Single |
| **Height:** | 5'11" |
| **Weight:** | 166 lbs. |

**Department Information**

| | |
|---|---|
| **Employee Number:** | 000190 |
| **Dept Number:** | D11 |
| **Manager:** | Irving Stern |
| **Position:** | Designer |
| **Phone:** | (208) 555-2986 |
| **Hire Date:** | 1974-07-26 |

**Education**

| | |
|---|---|
| **1974** | Computer Studies, B.Sc. University of Massachusetts |
| **1972** | Linguistic Anthropology, B.A. University of Toronto |

**Work History**

| | |
|---|---|
| **6/87 - present** | Microcode Design Optimizing algorithms for mathematical functions. |
| **4/77 - 5/87** | Printer Technical Support Installing and supporting laser printers. |
| **9/74 - 3/77** | Maintenance Programming Patching assembly language compiler for mainframes. |

**Interests**

- Wine tasting
- Skiing
- Swimming
- Dancing

## Eksempeldatabasetabeller

# Tillegg B. Bruke DB2-biblioteket

DB2 Universal Database-biblioteket består av hjelp på systemet, elektroniske bøker (PDF og HTML) og programeksempler i HTML-format. Her beskrives informasjonen du finner, og hvordan du får tilgang til den.

Du får tilgang til produktinformasjon på systemet gjennom informasjonssenteret. Du finner flere opplysninger i "Få tilgang til informasjon med informasjonssenteret" på side 106. Du kan se på oppgaveinformasjon, DB2-bøker, informasjon om problemløsing, programeksempler og DB2-informasjon på World Wide Web.

## PDF-filer og trykte bøker for DB2

### DB2-informasjon

Tabellen nedenfor deler inn DB2-bøkene i fire kategorier:

**Veiledninger og tilleggsdokumentasjon for DB2**
Disse bøkene inneholder felles DB2-informasjon for alle plattformer.

**Informasjon om installering og konfigurering for DB2**
Disse bøkene er for DB2 på en bestemt plattform. Det er for eksempel egne *begynnerbøker* for DB2 på OS/2, Windows og UNIX-baserte plattformer.

**Programeksempler for flere plattformer i HTML**
Disse eksemplene er HTML-versjonen av programeksempler som blir installert med Application Development Client. De er til orientering og erstatter ikke de faktiske programmene.

**Versjonsmerknader**
Disse filene gir opplysninger som kom for sent til å bli tatt med i DB2-bøkene.

Installeringsveiledningene, versjonsmerknadene og veiledningene er i HTML-format, og kan leses direkte fra CDen. De fleste bøkene finnes i HTML-format på produkt-CDen for lesing og i PDF-format (Adobe Acrobat Reader) på CDen med DB2-publikasjoner for lesing og utskrift. Du kan også bestille en trykt versjon fra IBM (se "Bestille trykte bøker" på side 102). Bøkene i tabellen nedenfor kan bestilles.

I OS/2 og Windows kan du installere HTML-filene under katalogen sqllib\doc\html. DB2-informasjon oversettes til flere språk, men ikke all

informasjon oversettes til alle språk. Når informasjon ikke er tilgjengelig på et bestemt språk, følger det med en engelsk versjon.

På UNIX-plattformer kan du installere flere språkversjoner av HTML-filene under delkatalogene doc/*%L*/html, der *%L* representerer språkversjonen. Du finner flere opplysninger i den aktuelle *begynnerboken*.

Du kan få tak i DB2-bøker og informasjon på en rekke ulike måter:

- "Få informasjon på systemet" på side 105
- "Søke etter informasjon på systemet" på side 109
- "Bestille trykte bøker" på side 102
- "Skrive ut PDF-bøkene" på side 101

*Tabell 1. DB2-informasjon*

| Navn | Beskrivelse | Formnummer<br><br>PDF-filnavn | HTML-katalog |
|---|---|---|---|
| | **Veiledninger og tilleggsdokumentasjon for DB2** | | |
| *Administration Guide* | *Administration Guide: Planning* inneholder en oversikt over databasebegreper, informasjon om utforming (for eksempel logisk og fysisk database) og en drøftelse av høy tilgjengelighet. | SC09-2946<br>db2d1x70 | db2d0 |
| | *Administration Guide: Implementation* gir informasjon om implementeringsoppgaver som utforming, databasetilgang, revisjon, reservekopiering og gjenoppretting. | SC09-2944<br>db2d2x70 | |
| | *Administration Guide: Performance* gir informasjon om databasemiljøet og vurdering og justering av applikasjonsytelse. | SC09-2945<br>db2d3x70 | |
| | Du kan bestille de tre bindene av *Administration Guide* på amerikansk-engelsk med formnummer SBOF-8934. | | |
| *Administrative API Reference* | Beskriver DB2-programmeringsgrensesnitt (APIer) og datastrukturer du kan bruke til å administrere databasene. Denne boken forklarer også hvordan du kaller opp APIer fra applikasjonene. | SC09-2947<br><br>db2b0x70 | db2b0 |

*Tabell 1. DB2-informasjon (fortsettelse)*

| Navn | Beskrivelse | Formnummer<br><br>PDF-filnavn | HTML-katalog |
|---|---|---|---|
| *Application Building Guide* | Inneholder systemkonfigureringsinformasjon og trinnvise instruksjoner for hvordan du kompilerer, linker og kjører DB2-applikasjoner på Windows, OS/2 og UNIX-baserte plattformer. | SC09-2948<br><br>db2axx70 | db2ax |
| *APPC, CPI-C, and SNA Sense Codes* | Inneholder generell informasjon om APPC-, CPI-C- og SNA-referansekoder som du kan støte på når du bruker DB2 Universal Database-produkter.<br><br>Bare tilgjengelig i HTML-format. | Ikke noe formnummer<br><br>db2apx70 | db2ap |
| *Application Development Guide* | Forklarer hvordan du kan utvikle applikasjoner som bruker DB2-databaser ved hjelp av innfelt SQL eller Java (JDBC og SQLJ). Blant emnene som blir diskutert, er skriving av lagrede prosedyrer, skriving av brukerdefinerte funksjoner, opprettelse av brukerdefinerte typer ved hjelp av utløsere og utvikling av applikasjoner i partisjonerte miljøer eller med forente systemer. | SC09-2949<br><br>db2a0x70 | db2a0 |
| *CLI Guide and Reference* | Forklarer hvordan du utvikler applikasjoner som går inn på DB2-databaser ved hjelp av DB2 CLI, et kallbart SQL-grensesnitt som er kompatibelt med Microsoft ODBC-spesifikasjonen. | SC09-2950<br><br>db2l0x70 | db2l0 |
| *Command Reference* | Forklarer hvordan du bruker kommandolinjebehandleren og beskriver DB2-kommandoene du kan bruke til å administrere databasen. | SC09-2951<br><br>db2n0x70 | db2n0 |

*Tabell 1. DB2-informasjon (fortsettelse)*

| Navn | Beskrivelse | Formnummer<br><br>PDF-filnavn | HTML-katalog |
|---|---|---|---|
| *Connectivity Supplement* | Inneholder konfigureringsinformasjon og informasjon om hvordan du bruker DB2 for AS/400, DB2 for OS/390, DB2 for MVS eller DB2 for VM som DRDA-applikasjonsklienter med DB2 Universal Database-tjenere. Boken forklarer også hvordan du bruker DRDA-applikasjonstjenere med DB2 Connect-applikasjonsklienter.<br><br>Tilgjengelig bare som HTML og PDF. | Ikke noe formnummer<br><br>db2h1x70 | db2h1 |
| *Data Movement Utilities Guide and Reference* | Forklarer hvordan du bruker DB2-funksjoner som import, eksport, innlasting, automatisk innlasting (Autoloader) og DPROP, som gjør det lettere å flytte data. | SC09-2955<br><br>db2dmx70 | db2dm |
| *Data Warehouse Center Administration Guide* | Inneholder opplysninger om hvordan du bygger og vedlikeholder et datavarehus ved hjelp av datavarehussenteret. | SC26-9993<br><br>db2ddx70 | db2dd |
| *Data Warehouse Center Application Integration Guide* | Gir opplysninger som hjelper programmerere til å integrere applikasjoner med datavarehussenteret og med Information Catalog Manager. | SC26-9994<br><br>db2adx70 | db2ad |
| *DB2 Connect Brukerhåndbok* | Inneholder informasjon om begreper, programmering og generell bruk av DB2 Connect-produkter. | SA15-4772<br><br>db2c0x70 | db2c0 |
| *DB2 Query Patroller Administration Guide* | Gir en oversikt over DB2 Query Patroller-systemet, med opplysninger om drift og administrasjon, og oppgaveinformasjon for det grafiske administrasjonsgrensesnittet. | SC09-2958<br><br>db2dwx70 | db2dw |
| *DB2 Query Patroller User's Guide* | Beskriver hvordan du bruker verktøyene og funksjonene til DB2 Query Patroller. | SC09-2960<br><br>db2wwx70 | db2ww |
| *Ordliste* | Inneholder definisjoner av termer brukt i DB2 og dets komponenter.<br><br>Tilgjengelig i HTML-format og i *SQL Reference*. | Ikke noe formnummer<br><br>db2t0x70 | db2t0 |

*Tabell 1. DB2-informasjon (fortsettelse)*

| Navn | Beskrivelse | Formnummer<br><br>PDF-filnavn | HTML-katalog |
|---|---|---|---|
| *Image, Audio, and Video Extenders Administration and Programming* | Inneholder generell informasjon om DB2-tilleggsmoduler og informasjon om administrasjon og konfigurering av IAV-tilleggsmoduler (bilde, lyd og video) og programmering med IAV-tilleggsmoduler. Her finner du referanseinformasjon, feilsøkingsinformasjon (med meldinger) og eksempler. | SC26-9929<br><br>dmbu7x70 | dmbu7 |
| *Information Catalog Manager Administration Guide* | Gir veiledning om hvordan du administrerer informasjonskataloger. | SC26-9995<br><br>db2dix70 | db2di |
| *Information Catalog Manager Programming Guide and Reference* | Inneholder definisjoner for arkitekturgrensesnittene for Information Catalog Manager. | SC26-9997<br><br>db2bix70 | db2bi |
| *Information Catalog Manager User's Guide* | Inneholder opplysninger om hvordan du bruker grensesnittet til Information Catalog Manager. | SC26-9996<br><br>db2aix70 | db2ai |
| *Installation and Configuration Supplement* | Leder deg gjennom planleggingen, installeringen og konfigureringen av plattformspesifikke DB2-klienter. Denne boken inneholder informasjon om binding, konfigurering av klient- og tjenerkommunikasjon, DB2-verktøy, DRDA AS, distribuert installering, konfigurering av distribuerte forespørsler og tilgangsmetoder for heterogene datakilder. | GC09-2957<br><br>db2iyx70 | db2iy |
| *Meldinger* | Inneholder lister over meldinger og koder som DB2, Information Catalog Manager og datavarehussenteret sender ut, og beskriver hva du bør gjøre.<br><br>Du kan bestille begge bindene av Meldinger på engelsk med formnummer SBOF-8932. | Bind 1<br>GA15-4787<br><br>db2m1x70<br>Bind 2<br>GA15-4782<br><br>db2m2x70 | db2m0 |
| *OLAP Integration Server Administration Guide* | Forklarer hvordan du bruker komponenten Administration Manager i OLAP Integration Server. | SC27-0787<br><br>db2dpx70 | ikke tilgjengelig |

*Tabell 1. DB2-informasjon (fortsettelse)*

| Navn | Beskrivelse | Formnummer<br><br>PDF-filnavn | HTML-katalog |
|---|---|---|---|
| *OLAP Integration Server Metaoutline User's Guide* | Forklarer hvordan du oppretter og legger inn data i OLAP-metaoversikter ved hjelp av standardgrensesnittet OLAP Metaoutline (ikke ved hjelp av Metaoutline Assistant). | SC27-0784<br><br>db2upx70 | ikke tilgjengelig |
| *OLAP Integration Server Model User's Guide* | Forklarer hvordan du lager OLAP-modeller ved hjelp av OLAP Model Interface (ikke ved hjelp av Model Assistant). | SC27-0783<br><br>db2lpx70 | ikke tilgjengelig |
| *OLAP Installeringsveiledning og brukerhåndbok* | Inneholder konfigurerings- og installeringsinformasjon for OLAP-oppstartingssettet. | SA15-4791<br><br>db2ipx70 | db2ip |
| *OLAP Spreadsheet Add-in Brukerhåndbok for Excel* | Beskriver hvordan du bruker regnearkprogrammet Excel til å analysere OLAP-data. | SA15-4792<br><br>db2epx70 | db2ep |
| *OLAP Spreadsheet Add-in Brukerhåndbok for Lotus 1-2-3* | Beskriver hvordan du bruker regnearkprogrammet Lotus 1-2-3 til å analysere OLAP-data. | SA15-4793<br><br>db2tpx70 | db2tp |
| *Replication Guide and Reference* | Gir informasjon om planlegging, konfigurering, administrering og bruk av IBM-replikeringsverktøyene som følger med DB2. | SC26-9920<br><br>db2e0x70 | db2e0 |
| *Spatial Extender User's Guide and Reference* | Inneholder opplysninger om installering, konfigurering, administrasjon, programmering og feilsøking for Spatial Extender. Inneholder også beskrivelser av romdatabegreper og referanseopplysninger (meldinger og SQL) som er spesifikke for Spatial Extender. | SC27-0701<br><br>db2sbx70 | db2sb |
| *Begynnerhåndbok for SQL* | Introduserer SQL-begreper og gir eksempler på mange setninger og oppgaver. | SA15-4773<br><br>db2y0x70 | db2y0 |

*Tabell 1. DB2-informasjon (fortsettelse)*

| Navn | Beskrivelse | Formnummer<br><br>PDF-filnavn | HTML-katalog |
|---|---|---|---|
| *SQL Reference, bind 1 og bind 2* | Beskriver SQL-syntaks, SQL-semantikk og reglene for språket. Omfatter også informasjon om manglende kompatibilitet fra versjon til versjon, produktbegrensninger og katalogoversikter.<br><br>Du kan bestille begge bindene av *SQL Reference* på amerikansk-engelsk med formnummer SBOF-8933. | Bind 1<br>SC09-2974<br><br>db2s1x70<br><br>Bind 2<br>SC09-2975<br><br>db2s2x70 | db2s0 |
| *System Monitor Guide and Reference* | Beskriver hvordan du samler inn ulike typer informasjon om databaser og databasesystemet. Denne boken forklarer hvordan du bruker informasjonen til å forstå databaseaktivitet, forbedre ytelsen og finne årsaken til problemer. | SC09-2956<br><br>db2f0x70 | db2f0 |
| *Text Extender Administration and Programming* | Inneholder generell informasjon om DB2-tilleggsmoduler og informasjon om administrasjon og konfigurering av teksttilleggsmodulen og programmering ved hjelp av teksttilleggsmoduler. Her finner du referanseinformasjon, feilsøkingsinformasjon (med meldinger) og eksempler. | SC26-9930<br><br>desu9x70 | desu9 |
| *Troubleshooting Guide* | Hjelper deg å finne kilden til feil, gjenopprette etter problemer og bruke feilsøkingsverktøy i samråd med kundetjenesten til DB2. | GC09-2850<br><br>db2p0x70 | db2p0 |
| *Nyheter* | Beskriver nye funksjoner og forbedringer i DB2 Universal Database versjon 7. | SA15-4774<br><br>db2q0x70 | db2q0 |
| **Informasjon om installering og konfigurering for DB2** | | | |
| *DB2 Connect Enterprise Edition for OS/2 and Windows Quick Beginnings* | Inneholder informasjon om planlegging, migrering, installering og konfigurering for DB2 Connect Enterprise Edition på OS/2 og 32-biters Windows-operativsystemer. Inneholder også informasjon om installering og konfigurering for mange støttede klienter. | GC09-2953<br><br>db2c6x70 | db2c6 |

*Tabell 1. DB2-informasjon (fortsettelse)*

| Navn | Beskrivelse | Formnummer<br><br>PDF-filnavn | HTML-katalog |
|---|---|---|---|
| *DB2 Connect Enterprise Edition for UNIX Quick Beginnings* | Inneholder informasjon om planlegging, migrering, installering, konfigurering og oppgaver for DB2 Connect Enterprise Edition på UNIX-baserte plattformer. Inneholder også informasjon om installering og konfigurering for mange støttede klienter. | GC09-2952<br><br>db2cyx70 | db2cy |
| *DB2 Connect Personal Edition Begynnerbok* | Inneholder informasjon om planlegging, migrering, installering, konfigurering og oppgaver for DB2 Connect Personal Edition på OS/2 og 32-biters Windows-operativsystemer. Inneholder også informasjon om installering og konfigurering for alle støttede klienter. | GA15-4786<br><br>db2c1x70 | db2c1 |
| **DB2 Connect Personal Edition Quick Beginnings for Linux** | Inneholder informasjon om planlegging, installering, migrering og konfigurering for DB2 Connect Personal Edition på alle støttede Linux-distribusjoner. | GC09-2962<br><br>db2c4x70 | db2c4 |
| *DB2 Data Links Manager Quick Beginnings* | Inneholder informasjon om planlegging, installering, migrering, konfigurering og oppgaver for DB2 Data Links Manager for AIX og 32-biters Windows-operativsystemer. | GC09-2966<br><br>db2z6x70 | db2z6 |
| *DB2 Enterprise - Extended Edition for UNIX Quick Beginnings* | Inneholder informasjon om planlegging, installering, migrering og konfigurering for DB2 Enterprise - Extended Edition på UNIX-baserte plattformer. Inneholder også informasjon om installering og konfigurering for mange støttede klienter. | GC09-2964<br><br>db2v3x70 | db2v3 |
| *DB2 Enterprise - Extended Edition for Windows Quick Beginnings* | Inneholder informasjon om planlegging, installering og konfigurering for DB2 Enterprise - Extended Edition for 32-biters Windows-operativsystemer. Inneholder også informasjon om installering og konfigurering for mange støttede klienter. | GC09-2963<br><br>db2v6x70 | db2v6 |

*Tabell 1. DB2-informasjon (fortsettelse)*

| Navn | Beskrivelse | Formnummer<br><br>PDF-filnavn | HTML-katalog |
|---|---|---|---|
| *DB2 for OS/2 begynnerbøker* | Inneholder informasjon om planlegging, installering, migrering og konfigurering for DB2 Universal Database på OS/2. Inneholder også informasjon om installering og konfigurering for mange støttede klienter. | GA15-4784<br><br>db2i2x70 | db2i2 |
| *DB2 for UNIX Quick Beginnings* | Inneholder informasjon om planlegging, installering, migrering og konfigurering for DB2 Universal Database på UNIX-baserte plattformer. Inneholder også informasjon om installering og konfigurering for mange støttede klienter. | GC09-2970<br><br>db2ixx70 | db2ix |
| *DB2 for Windows begynnerbøker* | Inneholder informasjon om planlegging, installering, migrering og konfigurering for DB2 Universal Database på 32-biters Windows-operativsystemer. Inneholder også informasjon om installering og konfigurering for mange støttede klienter. | GA15-4788<br><br>db2i6x70 | db2i6 |
| *DB2 Personal Edition Begynnerbok* | Inneholder informasjon om planlegging, installering, migrering og konfigurering for DB2 Universal Database Personal Edition på OS/2 og 32-biters Windows-operativsystemer. | GA15-4783<br><br>db2i1x70 | db2i1 |
| **DB2 Personal Edition Quick Beginnings for Linux** | Inneholder informasjon om planlegging, installering, migrering og konfigurering for DB2 Universal Database Personal Edition på alle støttede Linux-distribusjoner. | GC09-2972<br><br>db2i4x70 | db2i4 |
| *DB2 Query Patroller Installation Guide* | Inneholder installeringsinformasjon for DB2 Query Patroller. | GC09-2959<br><br>db2iwx70 | db2iw |
| **DB2 Warehouse Manager Installation Guide** | Inneholder installeringsinformasjon for varehusagenter, varehustransformatorer og Information Catalog Manager. | GC26-9998<br><br>db2idx70 | db2id |
| **Programeksempler for flere plattformer i HTML** | | | |

*Tabell 1. DB2-informasjon (fortsettelse)*

| Navn | Beskrivelse | Formnummer<br><br>PDF-filnavn | HTML-katalog |
|---|---|---|---|
| Programeksempler i HTML | Inneholder programeksempler i HTML-format for programmeringsspråkene på alle plattformer som DB2 støtter. Programeksemplene er bare til informasjonsformål. Ikke alle eksempler er tilgjengelige for alle programmeringsspråk. HTML-eksemplene er tilgjengelige bare når DB2 Application Development Client er installert.<br><br>Du finner flere opplysninger om programmer i *Application Building Guide*. | Ikke noe formnummer | db2hs |
| | **Versjonsmerknader** | | |
| *DB2 Connect Versjonsmerknader* | Inneholder opplysninger som kom for sent til å bli tatt med i DB2 Connect-bøkene. | Se merknad 2. | db2cr |
| *Installeringsmerknader for DB2* | Inneholder installeringsinformasjon som kom for sent til å bli tatt med i DB2-bøkene. | Tilgjengelig bare på produkt-CDen. | |
| *Versjonsmerknader for DB2* | Inneholder informasjon om alle DB2-produktene som kom for sent til å bli tatt med i DB2-bøkene. | Se merknad 2. | db2ir |

**Merknader:**

1. Tegnet *x* i sjette posisjon i filnavnet viser til språkversjonen av en bok. For eksempel viser filnavnet `db2d0e70` til den engelske versjonen av *Administration Guide*, mens filnavnet `db2d0f70` viser til den franske versjonen av samme bok. Bokstavene nedenfor i sjette posisjon av filnavnet brukes til å vise til språkversjonen:

| Språk | Identifikator |
|---|---|
| Portugisisk (Brasil) | b |
| Bulgarsk | u |
| Tsjekkisk | x |
| Dansk | d |
| Nederlandsk | q |
| Engelsk | e |
| Finsk | y |
| Fransk | f |

| | |
|---|---|
| Tysk | g |
| Gresk | a |
| Ungarsk | h |
| Italiensk | i |
| Japansk | j |
| Koreansk | k |
| Norsk | n |
| Polsk | p |
| Portugisisk | v |
| Russisk | r |
| Forenklet kinesisk | c |
| Slovensk | l |
| Spansk | z |
| Svensk | s |
| Tradisjonell kinesisk | t |
| Tyrkisk | m |

2. Informasjon som kom for sent til å bli tatt med i DB2-bøkene er tilgjengelig i versjonsmerknadene i HTML-format og som en ASCII-fil. HTML-versjonen er tilgjengelig fra informasjonssenteret og på produkt-CDer. Slik kan du lese en ASCII-fil:
   - På UNIX-baserte plattformer leser du filen Release.Notes. Denne filen ligger i katalogen DB2DIR/Readme/*%L*, der *%L* står for språkversjon og `DB2DIR` står for:
     - /usr/lpp/db2_07_01 på AIX
     - /opt/IBMdb2/V7.1 på HP-UX, PTX, Solaris og Silicon Graphics IRIX
     - /usr/IBMdb2/V7.1 på Linux
   - På andre plattformer leser du filen RELEASE.TXT. Denne filen ligger i katalogen der produktet er installert. På OS/2-plattformer kan du også dobbeltklikke på mappen **IBM DB2** og deretter på ikonet **Versjonsmerknader**.

## Skrive ut PDF-bøkene

Hvis du foretrekker å ha trykte kopier av bøkene, kan du skrive ut PDF-filene på CDen med DB2-publikasjoner. Ved hjelp av Adobe Acrobat Reader kan du skrive ut hele boken eller utvalgte sider. Du finner filnavnet til hver bok i biblioteket i tabell 1 på side 92.

Du kan få tak i nyeste versjon av Adobe Acrobat Reader fra Adobes nettsted: http://www.adobe.com.

PDF-filene følger med på CDen med DB2-publikasjoner. (PDF er filtypen.) Slik får du tak i PDF-filene:

1. Sett inn CDen DB2-publikasjoner. På UNIX-baserte plattformer tilkobler du CDen DB2-publikasjoner. Du kan lese om tilkoblingsprosedyrer i *begynnerbøker*.
2. Start Acrobat Reader.
3. Åpne PDF-filen fra et av disse stedene:
   - I OS/2 og Windows:

     katalogen *x*:\doc*språk*, der *x* er stasjonsbokstaven for CD-ROM-stasjonen og *språk* er landkode med to tegn (for eksempel NO for norsk).
   - På UNIX-baserte plattformer:

     katalogen */cdrom*/doc/*%L* på CDen, der */cdrom* er tilkoblingspunktet til CD-ROMen og *%L* stor for navnet på det ønskede språkmiljøet.

Du kan også kopiere PDF-filene fra CDen til en lokal stasjon eller nettverksstasjon og lese dem derfra.

## Bestille trykte bøker

Du kan bestille trykte DB2-bøker enkeltvis eller i sett (bare i Nord-Amerika) ved hjelp av et SBOF-nummer. Hvis du skal bestille trykte bøker, kontakter du en autorisert IBM-forhandler. Du kan også bestille bøker fra nettstedet for publikasjoner på http://www.elink.ibmlink.ibm.com/pbl/pbl.

To sett bøker er tilgjengelige. SBOF-8935 gir referanse- og bruksinformasjon for DB2 Warehouse Manager. SBOF-8931 gir referanse- og bruksinformasjon for alle andre DB2 Universal Database-produkter og -funksjoner. Innholdet i hver SBOF finner du i denne tabellen:

*Tabell 2. Bestille trykte bøker*

| SBOF-nummer | Bøker | |
|---|---|---|
| SBOF-8931 | • Administration Guide: Planning<br>• Administration Guide: Implementation<br>• Administration Guide: Performance<br>• Administrative API Reference<br>• Application Building Guide<br>• Application Development Guide<br>• CLI Guide and Reference<br>• Command Reference<br>• Data Movement Utilities Guide and Reference<br>• Data Warehouse Center Administration Guide<br>• Data Warehouse Center Application Integration Guide<br>• DB2 Connect Brukerhåndbok<br>• Installering og konfigurering<br>• Image, Audio, and Video Extenders Administration and Programming<br>• Meldinger, del 1 og 2 | • OLAP Integration Server Administration Guide<br>• OLAP Integration Server Metaoutline User's Guide<br>• OLAP Integration Server Model User's Guide<br>• OLAP Integration Server User's Guide<br>• OLAP Installeringsveiledning og brukerhåndbok<br>• OLAP Spreadsheet Add-in Brukerhåndbok for Excel<br>• OLAP Spreadsheet Add-in Brukerhåndbok for Lotus 1-2-3<br>• Replication Guide and Reference<br>• Spatial Extender Administration and Programming Guide<br>• Begynnerhåndbok for SQL<br>• SQL Reference, del 1 og 2<br>• System Monitor Guide and Reference<br>• Text Extender Administration and Programming<br>• Troubleshooting Guide<br>• Nyheter |
| SBOF-8935 | • Information Catalog Manager Administration Guide<br>• Information Catalog Manager User's Guide<br>• Information Catalog Manager Programming Guide and Reference | • Query Patroller Administration Guide<br>• Query Patroller User's Guide |

## DB2-dokumentasjon på systemet

### Få tilgang til hjelp på systemet

Du finner hjelp på systemet for alle DB2-komponenter. Tabellen nedenfor beskriver de ulike typene hjelp.

| Type hjelp | Innhold | Slik får du tilgang... |
|---|---|---|
| **Kommandohjelp** | Forklarer syntaksen til kommandoene i kommandolinjebehandleren. | Fra kommandolinjebehandleren i interaktiv modus skriver du:<br>`? kommando`<br>der *kommando* er et nøkkelord eller hele kommandoen.<br>Hvis du for eksempel skriver `? catalog`, får du frem hjelp til alle CATALOG-kommandoer, mens `? catalog database` viser hjelp til kommandoen CATALOG DATABASE. |
| **Hjelp til Klient-konfigureringsassistent**<br>**Hjelp til Kommandosenter**<br>**Hjelp til Kontrollsenter**<br>**Hjelp til datavarehussenteret**<br>**Hjelp til Aktivitetsanalysator**<br>**Hjelp til Information Catalog Manager**<br>**Hjelp til Satellittadministrasjonssenter**<br>**Hjelp til Skriptsenter** | Forklarer oppgavene du kan utføre i et vindu eller en notisbok. Hjelpen omfatter en oversikt og nødvendig informasjon som du må ha kjennskap til, og beskriver hvordan du bruker kontrollene i vinduer og notisbøker. | I et vindu eller en notisbok klikker du på skjermtasten **Hjelp** eller trykker på **F1**. |

| Type hjelp | Innhold | Slik får du tilgang... |
|---|---|---|
| **Meldingshjelp** | Beskriver årsaken til en melding og hva du eventuelt kan gjøre. | Fra kommandolinjebehandleren i interaktiv modus skriver du:<br>`? XXXnnnnn`<br>der *XXXnnnnn* er et gyldig meldingsnummer.<br>Hvis du for eksempel skriver `? SQL30081`, får du hjelp til meldingen SQL30081.<br>Du kan se på ett skjermbilde med meldingstekst om gangen ved å skrive:<br>`? XXXnnnnn \| more`<br>Hvis du vil lagre meldingshjelp i en fil, skriver du:<br>`? XXXnnnnn > filnavn.typ`<br>der *filnavn.typ* er filen der du vil lagre meldingshjelpen. |
| **Hjelp til SQL** | Forklarer syntaksen til SQL-setninger. | Fra kommandolinjebehandleren i interaktiv modus skriver du:<br>`help setning`<br>der *setning* er en SQL-setning.<br>Hvis du for eksempel skriver help SELECT, får du hjelp til SELECT-setningen.<br>**Merk:** SQL-hjelp er ikke tilgjengelig på UNIX-baserte plattformer. |
| **Hjelp til SQLSTATE** | Forklarer SQL-statuser og klassekoder. | Fra kommandolinjebehandleren i interaktiv modus skriver du:<br>`? sqlstatus eller ? klassekode`<br>der *sqlstatus* er en femsifret SQL-status og *klassekode* er de to første sifrene i SQL-statusen.<br>Hvis du for eksempel skriver `? 08003`, får du hjelp til SQL-statusen 08003, mens `? 08` viser hjelp til klassekoden 08. |

## Få informasjon på systemet

Bøkene som følger med dette produktet, finnes i maskinleselig format i formateringsspråk for hypertekst (HTML). Det maskinleselige formatet gjør at du kan søke i eller se gjennom informasjonen, og du finner hypertekstlinker til beslektet informasjon. Formatet gjør det også enklere å dele biblioteket på arbeidsplassen din.

Du kan se på bøkene eller programeksemplene med en hvilken som helst nettleser som følger spesifikasjonene til HTML versjon 3.2.

Slik kan du lese elektroniske bøker eller eksempelprogrammer:

- Hvis du kjører DB2-administrasjonsverktøy, bruker du informasjonssenteret.
- I en nettleser klikker du på **Fil —>Åpne side**. Siden du åpner, inneholder beskrivelser av og linker til DB2-informasjon:
  - På UNIX-baserte plattformer åpner du denne siden:

    *INSTHOME*/sqllib/doc/*%L*/html/index.htm

    der *%L* er språkmiljøet.
  - På andre plattformer åpner du denne siden:

    sqllib\doc\html\index.htm

    Banen finner du på stasjonen der DB2 er installert.

    Hvis du ikke har installert informasjonssenteret, kan du åpne siden ved å dobbeltklikke på ikonet **DB2-informasjon**. Avhengig av hvilket system du bruker, ligger ikonet i hovedproduktmappen eller på Start-menyen i Windows.

### Installere Netscape

Hvis du ikke har en nettleser fra før, kan du installere Netscape fra Netscape-CDen som følger med. Slik kan du få detaljerte opplysninger om hvordan du installerer den:

1. Sett inn Netscape-CDen.
2. På UNIX-baserte plattformer tilkobler du CDen. Du kan lese om tilkoblingsprosedyrer i *begynnerbøker*.
3. Du finner installeringsinstruksjoner i filen CDNAV*nn*.txt, der *nn* er en språkkode med to tegn. Filen ligger i rotkatalogen på CDen.

### Få tilgang til informasjon med informasjonssenteret

Informasjonssenteret gir deg rask tilgang til DB2-produktinformasjon. Informasjonssenteret er tilgjengelig på alle plattformer der DB2-administrasjonsverktøy er installert.

Du kan åpne informasjonssenteret ved å dobbeltklikke på ikonet Informasjonssenter. Avhengig av hvilket system du bruker, ligger ikonet i mappen Informasjon under hovedproduktmappen eller på **Start**-menyen i Windows.

Du kan også åpne informasjonssenteret ved hjelp av verktøylinjen og menyen **Hjelp** i Windows.

Informasjonssenteret har seks typer informasjon. Klikk på riktig flipp for å finne det aktuelle emnet.

**Oppgaver** Nøkkeloppgaver i DB2.

**Referanse** DB2-referanseinformasjon, for eksempel nøkkelord, kommandoer og APIer.

**Bøker** DB2-bøker.

**Feilsøking** Feilmeldingskategorier og hvordan du gjenoppretter.

**Programeksempler**
Programeksempler som følger med DB2 Application Development Client. Hvis du ikke har installert DB2 Application Development Client, vises ikke denne flippen.

**Web** DB2-informasjon på World Wide Web. Hvis du vil ha tilgang til denne informasjonen, må du ha en forbindelse med World Wide Web fra systemet.

Når du velger et punkt på en av listene ovenfor, starter informasjonssenteret et visningsprogram for å vise informasjonen. Visningsprogrammet kan være hjelpevisningsprogrammet til systemet, et redigeringsprogram eller en nettleser, avhengig av hvilken type informasjon du velger.

Informasjonssenteret har en søkefunksjon som du kan bruke til å lete etter bestemte emner uten å bla gjennom listene.

For fulltekstsøk kan du følge hypertekstlinken i informasjonssenteret til **Søk i DB2-informasjon på systemet**.

HTML-søketjeneren blir vanligvis startet automatisk. Hvis et søk i HTML-informasjonen ikke fungerer, kan det hende du må starte søketjeneren ved hjelp av en av disse metodene:

**I Windows**
Klikk på **Start** og velg **Programmer —> IBM DB2 —> Informasjon —> Start HTML-søketjener**.

**I OS/2** Dobbeltklikk på mappen **DB2 for OS/2** og deretter på ikonet **Start HTML-søketjener**.

Slå opp i versjonsmerknadene hvis du har andre problemer når du søker i HTML-informasjon.

**Merk:** Søkefunksjonen er ikke tilgjengelig i Linux, PTX og Silicon Graphics IRIX.

## Bruke DB2-veivisere

Veivisere hjelper deg å utføre bestemte administrasjonsoppgaver ved å lede deg gjennom hver oppgave trinn for trinn. Veivisere er tilgjengelige via kontrollsenteret og klientkonfigureringsassistenten. Tabellen nedenfor gir en oversikt over veivisere og formålene med dem.

**Merk:** Veiviserne Opprett database, Opprett indeks, Konfigurer flerstedsoppdatering og Ytelseskonfigurering er tilgjengelige for partisjonert databasemiljø.

| Veiviser | Hjelper deg å... | Slik får du tilgang... |
|---|---|---|
| **Tilføy database** | Katalogisere en database på en klientstasjon. | Fra klientkonfigureringsassistenten klikker du på **Tilføy**. |
| **Reservekopier database** | Fastsette, opprette og planlegge en reservekopieringsplan. | I kontrollsenteret høyreklikker du på databasen du vil reservekopiere, og velger **Reservekopier —> Database ved hjelp av veiviser**. |
| **Konfigurer flerstedsoppdatering** | Konfigurere en flerstedsoppdatering, en distribuert transaksjon eller en tofaseiverksetting. | I kontrollsenteret høyreklikker du på mappen **Databaser** og velger **Flerstedsoppdatering**. |
| **Opprett database** | Opprette en database og utføre noen grunnleggende konfigureringsoppgaver. | I kontrollsenteret høyreklikker du på mappen **Databaser** og velger **Opprett —> Database ved hjelp av veiviser**. |
| **Opprett tabell** | Velge grunnleggende datatyper og opprette en primærnøkkel for tabellen. | I kontrollsenteret høyreklikker du på ikonet **Tabeller** og velger **Opprett —> Tabell ved hjelp av veiviser**. |
| **Opprett tabellplass** | Opprette en ny tabellplass. | I kontrollsenteret høyreklikker du på ikonet **Tabellplasser** og velger **Opprett —> Tabellplass ved hjelp av veiviser**. |
| **Opprett indeks** | Finne ut hvilke indekser som bør opprettes og hvilke som bør slettes for alle dine spørringer. | I kontrollsenteret høyreklikker du på ikonet **Indeks** og velger **Opprett —> Indeks ved hjelp av veiviser**. |

| Veiviser | Hjelper deg å... | Slik får du tilgang... |
|---|---|---|
| **Ytelseskonfigurering** | Justere ytelsen til en database ved å oppdatere konfigurasjonsparametere slik at de samsvarer med dine forretningskrav. | I kontrollsenteret høyreklikker du på databasen du vil finjustere, og velger **Konfigurer ytelse ved hjelp av veiviser**.<br><br>For partisjonert databasemiljø står du i oversikten Databasepartisjoner og høyreklikker på den første databasepartisjonen du vil justere. Så velger du **Konfigurer ytelse ved hjelp av veiviser**. |
| **Gjenopprett database** | Gjenopprette en database etter en feil. SmartGuiden hjelper deg å finne ut hvilken reservekopi du skal bruke, og hvilke logger du bør avspille på nytt. | I kontrollsenteret høyreklikker du på databasen du vil gjenopprette, og velger **Gjenopprett —> Database ved hjelp av veiviser**. |

## Konfigurere en dokumenttjener

Standardverdien er at DB2-informasjon blir installert på det lokale systemet. Det betyr at alle som trenger tilgang til DB2-informasjon, må installere de samme filene. Hvis du vil ha DB2-informasjon lagret på ett enkelt sted, gjør du slik:

1. Kopier alle filer og delkataloger fra \sqllib\doc\html på det lokale systemet til en web-tjener. Hver bok har sin egen delkatalog som inneholder alle nødvendige HTML- og GIF-filer for boken. Sørg for at katalogstrukturen forblir den samme.
2. Konfigurer web-tjeneren slik at den ser etter filene på det nye stedet. Du finner opplysninger om dette i NetQuestion-tillegget i *Installation and Configuration Supplement*.
3. Hvis du bruker Java-versjonen av informasjonssenteret, kan du oppgi en basis-URL for alle HTML-filene. Du bør bruke URLen til listen over bøker.
4. Når du kan se på bokfilene, kan du sette bokmerke på de emnene du leser oftest. Det kan være lurt å sette bokmerke på disse sidene:
   - Liste over bøker
   - Innholdsfortegnelse over ofte brukte bøker
   - Ofte refererte artikler, som emnet ALTER TABLE
   - Søkeskjemaet

Du finner opplysninger om hvordan du kan gjøre dokumentasjonsfilene for DB2 Universal Database tilgjengelige fra en sentral maskin, i NetQuestion-tillegget i *Installation and Configuration Supplement*.

## Søke etter informasjon på systemet

Du kan søke etter informasjon i HTML-filer på en av disse måtene:

- Klikk på **Søk** i øverste ramme. Bruk søkeskjemaet til å finne et bestemt emne. Denne funksjonen er ikke tilgjengelig i Linux, PTX og Silicon Graphics IRIX.
- Klikk på **Indeks** i øverste ramme. Bruk stikkordregisteret til å finne et bestemt emne i boken.
- Hent frem innholdsfortegnelsen eller stikkordregisteret til hjelpen eller HTML-boken, og bruk deretter søkefunksjonen til nettleseren til å finne et bestemt emne i boken.
- Bruk bokmerkefunksjonen til nettleseren til å finne raskt tilbake til et bestemt emne.
- Bruk søkefunksjonen i informasjonssenteret til å finne bestemte emner. Du finner mer informasjon i “Få tilgang til informasjon med informasjonssenteret” på side 106.

# Tillegg C. Merknader

Henvisninger i boken til IBMs produkter, programmer eller tjenester betyr ikke at IBM har til hensikt å gjøre dem tilgjengelige i alle land der IBM driver virksomhet. Be din lokale IBM-representant om informasjon om hvilke produkter og tjenester som er tilgjengelige i Norge. Henvisninger til IBMs produkter, programmer eller tjenester betyr heller ikke at det bare er de som kan benyttes. Andre produkter, programmer eller tjenester som har tilsvarende funksjoner, kan brukes i stedet, forutsatt at de ikke gjør inngrep i noen av IBMs patent- eller opphavsrettigheter eller andre lovbeskyttede rettigheter. Vurdering og verifisering ved bruk sammen med andre produkter, programmer eller tjenester enn de som uttrykkelig er angitt av IBM, er brukerens ansvar.

IBM kan ha patent på eller patentsøknader til behandling for de produktene som er omtalt i denne publikasjonen. At du har mottatt denne publikasjonen, innebærer ikke at du får lisensrettighet til disse produktene. Du kan sende spørsmål angående lisenser til

Director of Commercial Relations - Europe
IBM Deutschland GmbH
Schönaicher Str. 220
D - 7030 Böblingen
Tyskland

Lisensforespørsler om dobbeltbyteinformasjon (DBCS) kan rettes til IBMs advokat eller til:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

INTERNATIONAL BUSINESS MACHINES CORPORATION LEVERER DENNE BOKEN SOM DEN ER (“AS IS”) UTEN FORPLIKTELSER AV NOE SLAG.

Denne boken kan inneholde tekniske unøyaktigheter eller typografiske feil. Opplysninger i denne boken kan bli endret. Slike endringer blir tatt med i nye utgaver av boken. IBM kan uten varsel endre produktene og/eller programmene som er beskrevet i denne boken.

Henvisninger i denne boken til andre nettsteder enn IBMs er bare til orientering og innebærer ikke at IBM gir sin tilslutning til det som står der.

Materialet på nevnte nettsteder er ikke en del av materialet for dette IBM-produktet, og all bruk av nettstedene skjer på egen risiko.

IBM kan bruke eller distribuere informasjon du gir, på hensiktsmessig måte uten forpliktelser.

Hvis du som lisensinnehaver av dette programmet ønsker informasjon om programmet for å kunne (i) utveksle informasjon mellom selvstendig utviklede programmer og andre programmer (inklusiv dette) og (ii) dra gjensidig nytte av informasjonen som er utvekslet, kan du kontakte

IBM
Software Marketing
Postboks 500
1411 Kolbotn

Slik informasjon kan være tilgjengelig i henhold til egne betingelser, og i noen tilfeller ved betaling av en avgift.

Det lisensierte programmet som er beskrevet i denne boken, og alt lisensiert materiale som er tilgjengelig for programmet, leveres av IBM i henhold til IBMs generelle betingelser, IBMs internasjonale bruksbetingelser eller en tilsvarende avtale mellom partene.

Alle ytelsesdata er målt i kontrollerte omgivelser. Resultatene som oppnås i andre omgivelser, kan variere betraktelig. Noen målinger kan ha blitt utført på systemer under utvikling, og det er ingen garanti for at målingene vil gi samme resultater på systemer i salg. Dessuten kan noen måleresultater være utledet. Faktiske måleresultater kan variere. Brukere av denne boken bør derfor kontrollere de relevante dataene for deres bestemte miljø.

Informasjon om andre produkter enn IBMs egne er hentet fra leverandørene av produktene, fra deres annonseringer eller fra andre tilgjengelige kilder. IBM har ikke testet disse produktene, og kan ikke bekrefte påstander om ytelse, kompatibilitet eller andre egenskaper ved dem. Spørsmål om funksjoner til ikke-IBM-produkter rettes til produktleverandøren.

Alle påstander om IBMs fremtidige handlinger eller planer kan endres eller trekkes tilbake uten forvarsel, og er kun uttrykk for mål og hensikter.

Disse opplysningene kan inneholde eksempler på data og rapporter som brukes i den daglige driften av et firma. For å illustrere eksemplene så godt som mulig blir det brukt navn på personer, firmaer og produkter. Alle disse navnene er fiktive, og enhver likhet med virkelige navn er tilfeldig.

RETT TIL KOPIERING:

Disse opplysningene kan inneholde programeksempler på kildespråk som illustrerer programmeringsteknikker på forskjellige plattformer. Du kan kopiere, endre og distribuere disse programeksemplene i en hvilken som helst form uten betaling til IBM, med den hensikt å utvikle, bruke, markedsføre eller distribuere applikasjoner som følger programmeringsgrensesnittet (API) for operativsystemet som programeksemplene er skrevet for. Disse eksemplene er ikke blitt testet grundig under alle forhold. IBM kan derfor ikke garantere eller antyde at disse programmene er pålitelige, at det tilbys service for dem, eller at de virker.

Hver kopi eller del av disse programeksemplene eller avledede arbeider av dem må ha med informasjon om opphavsrett som følger:

## Varemerker

Navnene nedenfor er varemerker for International Business Machines Corporation.

ACF/VTAM
AISPO
AIX
AIX/6000
AIXwindows
AnyNet
APPN
AS/400
BookManager
CICS
C Set++
C/370
DATABASE 2
DataHub
DataJoiner
DataPropagator
DataRefresher
DB2
DB2 Connect
DB2 Extenders
DB2 OLAP Server
DB2 Universal Database
Distributed Relational Database Architecture
DRDA
eNetwork
Extended Services
FFST
First Failure Support Technology
IBM
IMS
IMS/ESA
LAN DistanceMVS
MVS/ESA
MVS/XA
Net.Data
OS/2
OS/390
OS/400
PowerPC
QBIC
QMF
RACF
RISC System/6000
RS/6000
S/370
SP
SQL/DS
SQL/400
System/370
System/390
SystemView
VisualAge
VM/ESA
VSE/ESA
VTAM
WebExplorer
WIN-OS/2

Navnene nedenfor er varemerker for andre selskaper.

Microsoft, Windows og Windows NT er varemerker for Microsoft Corporation.

Java og alle Java-baserte varemerker og logoer samt Solaris er varemerker for Sun Microsystems, Inc.

Tivoli og NetView er varemerker for Tivoli Systems Inc.

UNIX er et varemerke som kun er lisensiert gjennom X/Open Company Limited.

Andre navn kan være varemerker for andre selskaper.

# Stikkordregister

## Andre tegn


-skjema
  definisjon av 4


## A


ADD CONSTRAINT-setning 53
Administration Guide v
ALL, bruke i en spørring 50
ALTER TABLE-setning 53
ANY-nøkkelord 50
Application Development Guide v
aritmetiske operatorer 25
AS-ledd 25
autorisasjons-ID 4


## B


basistabell 3
begrensninger
  entydige begrensninger 9
  for mengdeoperatorer 47
  referansebegrensninger 9
Beslektet dokumentasjon v
BETWEEN-predikat 48
BIGINT, datatype 5
binært heltall, beskrivelse 5
BLOB-datatype 67
BLOB-streng 67
Brukerdefinerte funksjoner 66
  definere 66
  ekstern skalarfunksjon 66
  ekstern tabellfunksjon 66
  kildefunksjon 66
  OLE DB-ekstern tabellfunksjon 66
bøker 91, 102


## C


CASE-uttrykk
  beskrivelse 34
  SIGN-funksjon 34
CHAR, datatype 5
CL_SCHED-eksempeltabell 72
CLOB-datatype 67
CLOB-streng 67
CONNECT-setning 18
  eksplisitt 18
  implisitt 18
CREATE DISTINCT TYPE 65
CREATE FUNCTION 66
CREATE TABLE-setning 9
CREATE TABLE-setning 9
*(fortsettelse)*
  NOT NULL / NOT NULL WITH DEFAULT-verdi for kolonne 9
CREATE TRIGGER 54
CREATE VIEW-setning 13
  WITH CHECK OPTION 13
CUBE 63
  delsumrader 63
  krysstabuleringsrader 63
CURRENT DATE-spesialregistre 68
CURRENT FUNCTION PATH-spesialregister 68
CURRENT SERVER-spesialregister 68
CURRENT TIME-spesialregister 68
CURRENT TIMESTAMP-spesialregister 68
CURRENT TIMEZONE-spesialregister 68


## D


databasesystem 1
datakonvertering
  kombinasjonsbetingelser 60
  mengdeoperatorer 47
datastruktur
  kolonne 3
  rad 3
  verdi 3
datatype
  distinkt 65
datatyper
  BIGINT 5
  CHAR 5
  DATE 5
  DATETIME 5
  DECIMAL 5
  DOUBLE 5
  FLOAT 5
  INTEGER 5
  REAL 5
  SMALLINT 5
  TIME 5
  TIMESTAMP 5
  VARCHAR 5
DATE, datatype 5
DATETIME, datatype 5
datetime-verdier, beskrivelse 5
DB2-bibliotek
  bestille trykte bøker 102
  bøker 91
  hjelp på systemet 103
  Informasjonssenter 106
  konfigurere dokumenttjener 109
  nyeste informasjon 101
  skrive ut PDF-bøker 101
  språkkode for bøker 100
  struktur 91
  søke i informasjon på systemet 109
  veivisere 108
  vise informasjon på systemet 105
DBLOB-datatype 67
DBLOB-streng 67
DECIMAL, datatype 5
DELETE-setning 13
delspørring
  definisjon 27
delsumrader 63
DEPARTMENT-eksempeltabell 73
desimal, beskrivelse 5
DISTINCT-nøkkelord 24, 29
distinkt datatype 65
DOUBLE, datatype 5


## E


eksempeldatabase 71
  opprette 72
  slette 72
eksempeltabeller 71, 91
ekstern kombinering
  beskrivelse 58
  FULL OUTER-kombinering 58
  LEFT OUTER-kombinering 58
  RIGHT OUTER-kombinering 58
ekstern skalarfunksjon 66
ekstern spørring, korrelasjon 41
ekstern tabellfunksjon 66
eksternt predikat 50
EMP_ACT-eksempeltabell 76
EMP_PHOTO-eksempeltabell 78
EMP_RESUME-eksempeltabell 78
EMPLOYEE-eksempeltabell 73
endre tabeller gjennom et utsnitt 15
  WITH CHECK OPTION 15
entydig begrensning 52
entydig nøkkel 52

entydig nøkkel 52 *(fortsettelse)*
  entydig begrensning 52
entydige begrensninger
  definisjon 51
EXCEPT ALL 46
EXCEPT-operator 46
  bruksbegrensninger med 47
    datatyper 47
  sortere resultater 47
EXISTS-predikat 49

## F

feilmeldinger
  meldingsnummer 18
  SQLSCODE 18
  SQLSTATE 18
felles tabelluttrykk
  beskrivelse 36
fjerne like rader 24
FLOAT, datatype 5
forhold mellom tabeller og
  utsnitt 13
fremmednøkkel 52
FROM-ledd 19
FULL OUTER-kombinering 58
full SELECT-setning 33
  ALL-nøkkelord 50
  ANY-nøkkelord 50
  delspørring 10, 50
  med INSERT-setning 10
full SELECT-setning, definisjon 10
funksjon
  beskrivelse 28
  brukerdefinert 28
  innebygd 28
  kolonne 28
  OLAP (online analytical
    processing) 63
  skalar 28
  tabell 30

## G

Gjenopprett, veiviser 109
grafisk streng
  fast lengde 5
  varierende lengde 5
GROUP BY 24
GROUP BY-ledd
  grupperingskolonne 30
  med HAVING-ledd 32
grupperingskolonne, definisjon 30

## H

HAVING 24
HAVING-ledd
  beskrivelse 32
hente data 19
hjelp på systemet 103
HTML
  programeksempler 99

## I

IN-predikat 48
IN_TRAY-eksempeltabell 79
indekseringsveiviser 108
informasjon på systemet
  søke 109
  vise 105
Informasjonssenter 106
INSERT-setning 10
  NOT NULL / NOT NULL WITH
    DEFAULT-verdi for
    kolonne 10
installere
  Netscape-nettleser 106
INTEGER, datatype 5
interaktiv SQL, definisjon 1
intern kombinering 58
INTERSECT ALL 47
INTERSECT-operator 47
  bruksbegrensninger med 47
    datatyper 47
  sortere resultater 47

## K

kildefunksjon 66
kolonne
  ASC, stigende
    sorteringsrekkefølge 23
  definisjon av 3
  DESC, synkende
    sorteringsrekkefølge 23
kolonnefunksjon 28
  AVG 28
  COUNT 28
  MAX 28
  MIN 28
kolonnefunksjoner 28
kombinasjonsbetingelse 58
kombinere
  datakonvertering 60
  kombinasjonsbetingelser 58
  korrelasjonsdelspørringer 42
  kryssprodukt 58
  uten kombinasjonsbetingelser 58
kombinere (join)
  definisjon 26
kombinere, spørringer 45
kommandolinjebehandler 1
Konfigurer flerstedsoppdatering,
  veiviser 108
konfigurere dokumenttjener 109
konvertere datatyper
  beskrivelse 33
korrelasjon
  beskrivelse 38
  delspørring 39
  delspørringer som bruker
    kombineringer 42
  navn 40
korrelasjonsdelspørring
  beskrivelse 39
  når brukes 41
korrelasjonsnavn
  kvalifisert referanse til
    kolonnenavn 38
  regler for 38
korrelasjonsreferanse,
  beskrivelse 39
kryssprodukt 58
krysstabuleringsrader 63
kvalifisere objekter 4, 17

## L

LEFT OUTER-kombinering 58
lete, eksistens 49
lete etter eksistens 49
LIKE-predikat 49
LOB
  posisjonsviser, definisjon 67
  streng, definisjon 67
LOB-plassering, definisjon 67

## N

nestede
  korrelasjonsdelspørringer 43
nestede tabelluttrykk,
  beskrivelse 36
Netscape-nettleser
  installere 106
NOT BETWEEN-predikat 48
NOT EXISTS-predikat 49
NOT IN-predikat 48
NOT LIKE-predikat 49
nullverdi 45
  slett kolonneverdi 12
nullverdi, beskrivelse 5
numre, beskrivelse 5
nyeste informasjon 101
nøkkel
  definisjon 52
  entydig 52
  fremmed 52
  primær 52
  sammensatt 52

## O

OLAP 63
OLAP-funksjoner 63
  partisjonsrader i 63
  samlingsgruppe 63
  sortere rader i 63
OLE DB-ekstern tabellfunksjon 66
Opprett database, veiviser 108
Opprett tabell, veiviser 108
Opprett tabellplass, veiviser 108
opprette eksempeldatabasen 72
ORDER BY-ledd 22
  mengdeoperatorer 47
ORG-eksempeltabell 79
overordnet nøkkel, definisjon 52

## P

partisjonert relasjonsdatabase, definisjon 1
PDF 101
posisjonsviser 67
predikat
  IS NOT NULL 20
  IS NULL 20
presisjon, som et numerisk attributt 5
primærnøkkel 52
programeksempler
  HTML 99
  på tvers av plattformer 99
PROJECT-eksempeltabell 80

## Q

Quick Beginnings v

## R

rad
  definisjon av 3
  velge 20
REAL, datatype 5
referanseintegritetsbegrensninger
  beskrivelse 52
  definisjon 51
  fremmednøkkel 52
  overordnet nøkkel 52
rekkefølge på operasjoner 24, 28
rekursive spørringer, beskrivelse 63
relasjonsdatabase, definisjon 1
relasjonsdatabase med flere noder, definisjon 1
Reservekopier database, veiviser 108
reserverte skjemaer 4
resultattabell 3
RIGHT OUTER-kombinering 58
ROLL-UP
  delsumrader 63
ROLLUP 63

## S

SALES-eksempeltabell 81
samkjøre resultater av spørringer 45
sammenlikningsoperator brukt i en delspørring 50
sammensatt nøkkel 52
SELECT-setning 19
SET CONSTRAINTS-setning 53
SET-ledd
  med UPDATE-setning 12
skalarfunksjon 28
  DECIMAL 36
Skalarfunksjon
  ABS 29
  HEX 29
  LENGTH 29
  SIGN 29
  YEAR 29
skalarfunksjon med full SELECT-setninger
  beskrivelse 33
skrive ut PDF-bøker 101
slette eksempeldatabasen 72
SMALLINT, datatype 5
SmartGuider
  veivisere 108
SOME-nøkkelord 50
sortere rader 22
spesialregister 68
  CURRENT DATE 68
  CURRENT DEGREE 68
  CURRENT FUNCTION PATH 68
  CURRENT PATH 68
  CURRENT SERVER 68
  CURRENT TIME 68
  CURRENT TIMESTAMP 68
  CURRENT TIMEZONE 68
  USER 68
språkkode
  bøker 100
spørringer, tilkoble 47
SQL-prosedyrespråk v
SQL Reference v
STAFF-eksempeltabell 82
STAFFG-eksempeltabell 83
streng
  LOB 67
Structured Query Language (SQL), definisjon 1
systemkataloger 69
søke
  informasjon på systemet 107, 109
søkebetingelse 20

## T

tabell
  basistabell 3
  definisjon av 3
  eksempeldatabase 71
  entydig begrensning 52
  entydig nøkkel 52
  fremmednøkkel 52
  funksjoner 30
  kombinere data (join) 26
  kvalifisere et kolonnenavn 38
  primærnøkkel 52
  resultattabell 3
tabellfunksjon
  SQLCACHE_SNAPSHOT 30
tabellkontrollbegrensning
  beskrivelse 53
  definisjon 51
  utsatt begrensningskontroll 53
tabelluttrykk
  beskrivelse 35
tegn, som et numerisk attributt 5
tegnstreng
  fast lengde 5
  som datatype 5
  varierende lengde 5
Tilføy database, veiviser 108, 109
tilkoble spørringer 47
TIME, datatype 5
TIMESTAMP, datatype 5

## U

UNION ALL 45
UNION-operator 45, 46
  beskrivelse 45
  bruksbegrensninger med datatyper 47
  sortere resultater 46
UPDATE-setning 12
USER-spesialregister 68
Utløsere
  beskrivelse 54
  CREATE TRIGGER 54
  definisjon 51
  etter-utløser 54
  før-utløser 54
  overgangsvariabler 56
utsnitt
  kvalifisere et kolonnenavn 38
uttrykk 25
uttrykk, navngi 25

# V

VALUES-ledd
  med INSERT-setning 10
VARCHAR, datatype 5
veiviser
  gjenopprette database 109
veivisere
  indeksering 108
  konfigurere
    flerstedsoppdatering 108
  opprette database 108
  opprette tabell 108
  opprette tabellplass 108
  reservekopiere database 108
  tilføye database 108, 109
  utføre oppgaver 108
  ytelseskonfigurering 108
velge liste 19
verdi
  definisjon av 3
verdi i SQL 5
versjonsmerknader 101
Vis
  beskrivelse 4
  fordeler 4
vise
  informasjon på systemet 105

# W

WHERE-ledd 20
  grupperingsvurderinger 31
  kombinere tabelldata (join) i
    SELECT-setning 26
WITH CHECK OPTION 15
WITH-ledd 36

# Y

Ytelseskonfigurering, veiviser 108

# Kontakte IBM

Hvis du har et teknisk problem, bør du se gjennom og utføre handlingene som er foreslått i *Troubleshooting Guide*, før du kontakter kundestøtten for DB2. Denne veiledningen inneholder tips til informasjonsinnsamling som kan gjøre det enklere for DB2-kundestøtten å hjelpe deg.

Hvis du trenger informasjon eller vil bestille noen av DB2 Universal Database-produktene, kontakter du en IBM-representant på et lokalt avdelingskontor eller en autorisert IBM-programvareforhandler.

Hvis du er i USA, kan du ringe et av disse numrene:

- 1-800-237-5511 for kundestøtte
- 1-888-426-4343 hvis du vil vite mer om tilleggstjenester

## Produktinformasjon

Hvis du er i USA, kan du ringe et av disse numrene:

- 1-800-IBM-CALL (1-800-426-2255) eller 1-800-3IBM-OS2 (1-800-342-6672) for å bestille produkter eller få generell informasjon.
- 1-800-879-2755 for å bestille publikasjoner.

**http://www.ibm.com/software/data/**
DB2-sidene på World Wide Web inneholder gjeldende DB2-informasjon om nyheter, produktbeskrivelser, opplæringsplaner og så videre.

**http://www.ibm.com/software/data/db2/library/**
Biblioteket DB2 Product and Service Technical Library gir deg tilgang til ofte spurte spørsmål, rettelser, bøker og oppdatert teknisk informasjon om DB2.

**Merk:** Det er mulig at denne informasjonen bare finnes på engelsk.

**http://www.elink.ibmlink.ibm.com/pbl/pbl/**
Nettstedet for bestilling av publikasjoner internasjonalt har informasjon om hvordan du bestiller bøker.

**http://www.ibm.com/education/certify/**
Professional Certification-programmet fra IBM-nettstedet har opplysninger om sertifiseringstesting for en rekke IBM-produkter, deriblant DB2.

**ftp.software.ibm.com**
Logg deg på som ″anonymous″. I katalogen /ps/products/db2 finner du demoer, rettelser, informasjon og verktøy som gjelder DB2 og mange relaterte produkter.

**comp.databases.ibm-db2, bit.listserv.db2-l**
Disse nyhetsgruppene er tilgjengelige for brukere som ønsker å diskutere sine erfaringer med DB2-produkter.

**På Compuserve: GO IBMDB2**
Oppgi denne kommandoen for å komme til fora for IBMs DB2-produkter. Alle DB2-produktene støttes gjennom disse foraene.

Du finner ut hvordan du kontakter IBM utenfor USA, i Appendix A i **IBM Software Support Handbook**. Du finner dette dokumentet ved å gå til nettsiden http://www.ibm.com/support/. Deretter velger du linken IBM Software Support Handbook nær bunnen av siden.

**Merk:** I noen land bør autoriserte IBM-forhandlere kontakte sin forhandlerkontakt i stedet for IBM Kundeservice.

Delenummer: CT7YHNO

Trykt i Norge

CT7YHNO

SA15-4773-00