

# ABAP Datenbanken

## Flugdatenbank - Tabellen

- **SCARR**  
Fluggesellschaften
- **SPFLI**  
Flugpläne
- **SFLIGHT**  
Flugdaten
- **SBOOK**  
Buchungen
- **SCUSTOM**  
Kundendaten
- **SCURR**  
Währungen mit Umrechnungsfaktoren

# ABAP Datenbanken

## Data Browser

- Betrachten von Struktur und Inhalt von Tabellen
- Aufruf über  
SAP-Menü -> Werkzeuge -> ABAP Workbench ->  
Übersicht -> Data Browser
- Einführung unter [SAP-Bibliothek](#)

# ABAP Datenbanken

## Flugdatenbank - Tabellen

### SCARR

CARRID	CARRNAME	CURRCODE	URL
AA	American Airlines	USD	http://www.aa.com
AC	Air Canada	CAD	http://www.aircanada.ca
AF	Air France	EUR	http://www.airfrance.fr
AZ	Alitalia	EUR	http://www.alitalia.it
BA	British Airways	GBP	http://www.british-airways.com
FJ	Air Pacific	USD	http://www.airpacific.com
CO	Continental Airlines	USD	http://www.continental.com
DL	Delta Airlines	USD	http://www.delta-air.com
AB	Air Berlin	EUR	http://www.airberlin.de

### SPFLI

CARRID	CONNID	CITYFROM	CITYTO
AZ	0555	ROME	FRANKFURT
AZ	0789	TOKYO	ROME

### SFLIGHT

CARRID	CONNID	FLDATE	PRICE	SEATSMAX
AZ	0555	20050420	185.00	280
AZ	0555	20050629	185.00	280
AZ	0555	20050907	185.00	280
AZ	0555	20051116	185.00	280
AZ	0555	20060125	185.00	280
AZ	0555	20060405	185.00	280
AZ	0555	20060614	185.00	280

### SBOOK

CARRID	CONNID	FLDATE	BOOKID	CUSTOMID	SMOKER
AZ	0555	20050420	00000000	00002634	

### SCUSTOM

ID	NAME	STREET	CITY
00002634	Chantal Domenech	74 rue Nieuport	Vélizy

# ABAP Datenbankzugriffe

## Einführung

- SAP / R3 verwendet eine Datenbankschnittstelle zur Abstraktion des zugrundeliegenden Datenbanksystems
- Abfragesprache hierfür ist Open SQL
- Schlüsselwörter
  - **SELECT**  
Liest Zeilen aus einer Tabelle
  - **INSERT**  
Fügt Zeilen in eine Tabelle ein
  - **UPDATE**  
Ändert Zeilen in einer Tabelle
  - **DELETE**  
Löscht Zeilen einer Tabelle

# ABAP Datenbankzugriffe

## Tabellendeklaration

- Für die Verwendung von Tabellen in einem Report, müssen diese erst deklariert werden
- Deklaration erfolgt mittels des Schlüsselwortes **TABLES**
- Beispiel:

```
TABLES:  
    scarr.
```

# ABAP Datenbankzugriffe

## Daten selektieren

- Zugriff auf Tabellen erfolgt über SELECT ... ENDSELECT
- SELECT ... ENDSELECT arbeitet ähnlich einer Schleife. Für jede selektierte Zeile einer Tabelle, wird die Schleife durchlaufen
- Grundlegende Syntax:  
SELECT <columns>  
FROM <table>  
WHERE <condition>.  
ENDSELECT.
- <columns> ist eine Liste der auszugebenden Spalten
- <condition> ist ein Selektionskriterium für die Auswahl der Zeilen
- WHERE kann entfallen (alle Zeilen werden selektiert)
- Anstatt expliziter Auswahl von Spalten können alle Spalten mittels \* ausgewählt werden

# ABAP Datenbankzugriffe

## SELECT - Beispiele

- Selektiere alle Fluggesellschaften aus scarr

TABLES:

scarr.

```
SELECT * FROM scarr.
```

```
WRITE: / scarr-carrid, scarr-carrname.
```

```
ENDSELECT.
```

- Selektiere alle Fluggesellschaften mit carrid = 'AZ'

TABLES:

scarr.

```
SELECT * FROM scarr WHERE carrid = 'AZ'.
```

```
WRITE: / scarr-carrid, scarr-carrname.
```

```
ENDSELECT.
```

# ABAP Datenbankzugriffe

## SELECT SINGLE

- Bei Selektionen, die nur einen Datensatz als Ergebnis liefern (z. B. Primärschlüssel als Kriterium), kann anstatt **SELECT ... ENDSELECT** die verkürzte Schreibweise **SELECT SINGLE** verwendet werden
- Verwendung analog zu **SELECT ... ENDSELECT**, allerdings keine Schleifendurchläufe
- Beispiel

```
SELECT SINGLE *  
  FROM scarr  
  WHERE carrid = 'LH'.  
WRITE: / scarr-carrid, scarr-carrname.
```

# ABAP Datenbankzugriffe

## SELECT - Systemfelder

- **SY-SUBRC**  
**SELECT ... ENDSELECT** liefert folgende Rückgabewerte in **SY-SUBRC**:
  - 0: Suche erfolgreich, mindestens ein Datensatz gefunden
  - 4: Kein Datensatz gefunden

# ABAP Datenbankzugriffe

## SELECT - Systemfelder

- **SY-DBCNT**
  - Wird bei Datenbankoperationen eingesetzt
  - Wird bei **SELECT ... ENDSELECT** nach der Anweisung **ENDSELECT** von der Ausführungsumgebung gesetzt
  - Inhalt abhängig von der Operation
    - **SELECT**: Anzahl der selektierten Datensätze
    - **DELETE**: Anzahl der gelöschten Datensätze
    - **INSERT**: Anzahl der eingefügten Datensätze
    - **UPDATE**: Anzahl der geänderten Datensätze

# ABAP Datenbankzugriffe

## SELECT - Datensatzselektion

- Selektion von Zeilen mit **WHERE**
- **WHERE** prüft den Inhalt von Feldern auf die Erfüllung der angegebenen Bedingung
- Bedingungen können mit **AND** und / oder **OR** verknüpft werden
- **NOT** ermöglicht die Negierung einer Bedingung
- Klammern können für Teilausdrücke verwendet werden
- Neben den bekannten Vergleichsoperatoren existieren folgende weitere Operatoren:
  - **BETWEEN**: Selektion anhand eines Bereichs
  - **IN**: Selektion anhand einer Menge
  - **LIKE**: Selektion anhand eines Musters (% und \_ als Wildcards)

# ABAP Datenbankzugriffe

## SELECT - Sortierung

- **ORDER BY**  
Sortiert die Ergebnismenge anhand eines angegebenen Feldes.
- Beispiel  

```
SELECT * FROM scarr  
      ORDER BY carrid ASCENDING.  
      WRITE: / scarr-carrid, scarr-carrname.  
ENDSELECT.
```
- Sortierreihenfolge kann durch **ASCENDING** (aufsteigend) bzw. **DESCENDING** (absteigend) angegeben werden

# ABAP Datenbankzugriffe

## SELECT - Aggregatfunktionen

- Aggregatfunktionen werten den Inhalt eines (meist numerischen) Tabellenfeldes in allen selektierten Datensätzen aus und liefern einen einzigen Wert als Resultat zurück
- Folgende Aggregatfunktionen sind verfügbar
  - **AVG**: Berechnet den Durchschnitt
  - **COUNT**: Anzahl der selektierten Datensätze
  - **MAX**: Maximalwert der selektierten Feldinhalte
  - **MIN**: Minimalwert der selektierten Feldinhalte
  - **SUM**: Summe der Feldinhalte

# ABAP Datenbankzugriffe

## SELECT – Aggregatfunktionen - Beispiel

- COUNT:  
TABLES: scarr.  
DATA: anzahl TYPE i.

```
SELECT COUNT(*) FROM SCARR INTO anzahl.
```

```
WRITE: / 'Anzahl der Fluggesellschaften:', anzahl.
```

- AVG:  
TABLES: sflight.  
DATA: average\_price LIKE sflight-price.

```
SELECT AVG( price ) FROM sflight INTO average_price.
```

```
WRITE: / 'Durchschnittspreis für einen Flug:', average_price.
```

# ABAP Datenbankzugriffe

## SELECT – Verknüpfen von Tabellen

- Die Verknüpfung von Tabellen erfolgt über einen sogenannten Join
- Hierfür werden zwei Tabellen über ihre Fremd-/Primärschlüsselbeziehung verbunden
- Im einfachsten Fall geschieht dies durch geschachtelte **SELECT ... ENDSELECT** Schleifen

# ABAP Datenbankzugriffe

## SELECT – Verknüpfen von Tabellen - Beispiel

- Join von scarr und spfli:

TABLES:

scarr,  
spfli.

SELECT \* FROM SCARR.

WRITE: / scarr-carrname.

SELECT \* FROM SPFLI

WHERE carrid = scarr-carrid.

WRITE: / spfli-cityfrom, spfli-cityto.

ENDSELECT.

ENDSELECT.