Do nauki języka SQL potrzebny jest jego **interpreter,** czyli baza danych. W ćwiczeniach wykorzystujemy bazę danych **InterBase** firmy **Inprise** (Borland).
Dlaczego InterBase? Ponieważ:

- jest to system relacyjnej bazy danych, w którym została zaimplementowana pełna składnia języka SQL;
- InterBase w wersji Personal Edition jest dostępny za darmo;
- jest to w pełni profesjonalny i popularny system, który dodatkowo jest łatwy w obsłudze i administrowaniu.

Instalacja InterBase

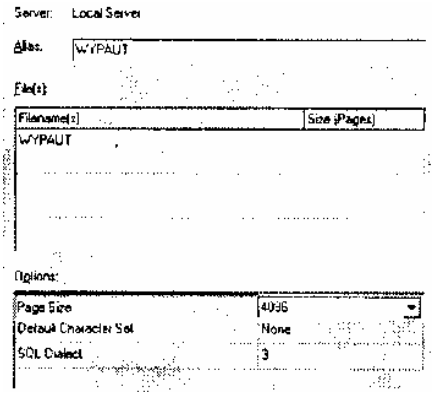
Aby pracować z InterBase należy go zainstalować. Serwer SQL InterBase znajduje się na stronie internetowej firmy BSC: http://www.borland.com.pl. Do pracy z ćwiczeniami potrzebny nam jest jeden plik; **ib_server_6_0_1.zip** (5,36 MB), który jest wersją Server i Client InterBase dla Windows.
Po rozpakowaniu **pliku ib_server_6_0_1.zip**, utworzony zostanie katalog: **ib_server_6_0_1**. W tym katalogu znajduje się jeszcze katalog **server**, w którym znajduje się program instalacyjny **setup.exe**.
Przyciskamy next, wybieramy wszystkie komponenty/Install/finish.

Narzędzie IBConsole

IBConsole jest narzędziem, w którym możemy stworzyć bazę danych. Z poziomu **IBConsole** możemy również wywołać narzędzie **Interactive SQL**, które pozwala na wprowadzanie poleceń SQL i ich wykonywanie na bazie danych. Wykonywanie wprowadzanych poleceń dokonuje się przez naciśnięcie klawiszy Ctrl+E (Execute). Aplikacja Interactive SQL umożliwia również wykonywanie skryptów SQL

Tworzenie bazy danych w InterBase.

Jeśli nie jesteśmy w aplikacji IBConsole, musimy ją uruchomić z Menu Start/Programy/ InterBase/IBConsole. Z menu Server wybieramy Login.... W oknie wpisujemy użytkownika **SYSDBA** i hasło **masterkey** (rozróżniane są duże i małe litery). Po zalogowaniu się do menedżera bazy InterBase przejść do menu **Database** do pozycji **Create Database**. Okno wypełnić tak jak na rysunku:



Wcisnąć OK.
Baza, którą stworzyliśmy ma nazwę WYPAUT. Słada się ona z pięciu tabel. Przechowuje ona dane o klientach, pracownikach, samochodach, miejscach, z których samochody można wypożyczyć oraz dane o wypożyczeniach.

Wykonywanie skryptów

Baza została utworzona. Teraz przechodzimy do wykonywania **skryptów**, które utworzą tabele w bazie danych i wypełnią je danymi. Skrypty te można skopiować z serwera ftp wydawnictwa Helion (ftp://ftp.helion.com.pl/przyklady/cwsql.zip). Aby wykonac skrypty tworzące tabele i wypełniające je danymi, musimy przejść do menu **Query** w **Interactive SQL** i wybrać pozycję **Load Script**. Okno, które się pojawi pozwala na wybranie pliku skryptu.

Po naciśnięciu kombinacji klawiszy Ctrl+E skrypt zostanie wykonany i utworzona zostanie tabela. Skrypt zostanie wykonany wtedy, gdy jesteśmy podłączeni do bazy WYPAUT. (W pasku zadań na samym dole jest wyświetlana informacja, że baza, do której jesteśmy aktualnie podłączeni to WYPAUT)

OPIS TABEL:
Tabela KLIENCI – przechowuje dane na temat klientów wypożyczających samochody.
Tabela SAMOCHODY – zawiera informacje o dostępnych samochodach, które klient może wypożyczyć.
Tabela PRACOWNICY – zawiera dane wszystkich pracowników firmy wypożyczającej samochody.
Tabela MIEJSCA – przechowuje informacje o miejscach, z których klient wypożyczył samochód oraz informacje o miejscach oddania.
Tabela WYPOZYCZENIA – przechowuje informacje o wypożyczonych samochodach, miejscu wypożyczenia i oddani, klientach, dacie itd.

ĆWICZENIE: NARYSOWAĆ DIAGRAM POKAZUJĄCY RELACJE MIEDZY TABELAMI.

I Zapytania SQL

SELECT

Polecenie SELECT jest używane do pobierania danych z bazy danych

SELECT	opisuje nazwy kolumn, wyrażenia arytmetyczne, funkcje;
FROM	nazwy tabel lub widoków;
WHERE	warunek wybierania wierszy
GROUP BY	nazwy kolumn;
HAVING	warunek grupowania wybieranych wierszy;
ORDER BY	nazwy kolumn lub pozycje kolumn.

WYBIERANIE WSZYSTKICH KOLUMN.

Ćw. 1.
Wyświetl wszystkie kolumny i wiersze tabeli PRACOWNICY.

Wskazówka

```
SELECT *
FROM PRACOWNICY;
```

WYBIERANIE OKREŚLONYCH KOLUMN.

Ćw. 2.
Wyświetl kolumny IMIE, NAZWISKO, DZIAL z tabeli PRACOWNICY.

Wskazówka

```
SELECT IMIE, NAZWISKO, DZIAL
FROM PRACOWNICY;
```

WYBIERANIEZ JEDNOCZESNYM PORZĄDKOWANIEM

Ćw. 3.

Wyświetl kolumny IMIE, NAZWISKO, DZIAL z tabeli PRACOWNICY i jednocześnie uporządkuj dane według nazwiska.

Wskazówka

```
SELECT IMIE, NAZWISKO, DZIAL
FROM PRACOWNICY
ORDER BY NAZWISKO ASC;
```

ASC – ozn, porządek jest rosnący. Sortowanie rosnące jest domyślne, więc ASC można pominąć.
DESC – porządek malejący.

Ćw. 4.

Wyświetl kolumny IMIE, NAZWISKO, DZIAL z tabeli PRACOWNICY i jednocześnie uporządkuj dane rosnąco według stanowiska i malejąco według nazwiska.

Wskazówka

```
SELECT IMIE, NAZWISKO, DZIAL, STANOWISKO
FROM PRACOWNICY
ORDER BY STANOWISKO ASC,
         NAZWISKO DESC;
```

Istnieje inny sposób na wskazanie kolumn w klauzuli ORDER BY. Zamiast nazywać kolumny, możemy je wskazać poprzez ich pozycje na liście:

```
SELECT IMIE, NAZWISKO, DZIAL, STANOWISKO
FROM PRACOWNICY
ORDER BY 4 ASC,
         2 DESC;
```

Dozwolona jest tylko jedna klauzula ORDER BY w zapytaniu SELECT. Klauzulę ORDER BY określa się jako ostatnią w całym zapytaniu SELECT.

WYBIERANIEZ NIEPOWTARZAJĄCYCH SIĘ WIERSZY

Ćw. 5.

Wyświetl tylko raz nazwę stanowiska z tabeli PRACOWNICY.

Wskazówka

```
SELECT DISTINCT STANOWISKO
FROM PRACOWNICY;
```

Ćw. 6.

Wyświetl wszystkie stanowiska obejmowane w danych działach z tabeli PRACOWNICY.

Wskazówka

```
SELECT DISTINCT STANOWISKO, DZIAL
FROM PRACOWNICY;
```

Słowo kluczowe DISTINCT (każdą wartość wyświetla tylko raz) musi występować zaraz po słowie kluczowym SELECT i może być użyte tylko jeden raz w całym zapytaniu SELECT.

WYBIERANIEZ OKREŚLONYCH WIERSZY

Ćw. 7.

Wyświetl wszystkich pracowników pracujących na stanowisku sprzedawcy z tabeli PRACOWNICY.

Wskazówka:

```
SELECT IMIE, NAZWISKO,STANOWISKO, DZIAL
FROM PRACOWNICY
WHERE STANOWISKO='SPRZEDAWCA';
```

W przypadku kolumn typu znakowego, daty lub czasu, wartości, dla których sprawdzany jest warunek muszą być otoczone apostrofem. Przy porównywaniu kolumn typu znakowego rozróżniane są wielkie i małe litery. Dla kolumn typu numerycznego jak INTEGER, SMALLINT, wartości do porównania nie są otaczane apostrofem

Ćw. 8.

Wyświetl dane wszystkich klientów z tabeli WYPOZYCZENIA, dla których cena jedn. wypożyczenia samochodu była większa lub równa
100.

Wskazówka:

```
SELECT NR_KLIENTA, NR_SAMOCHODU, NR_PRACOW_WYP, CENA_JEDN
FROM WYPOZYCZENIA     WHERE CENA_JEDN>=100;
```

OPERATORY LOGICZNE

Ćw. 8.

Wyświetl wszystkich pracowników tabeli PRACOWNICY pracujących na stanowisku sprzedawca w dziale obsługi klienta.

Wskazówka:

```
SELECT IMIE, NAZWISKO, STANOWISKO, DZIAL
FROM PRACOWNICY
WHERE STANOWISKO='SPRZEDAWCA'
         AND DZIAL='OBSLUGA KLIENTA';
```

Ćw. 9.

Wyświetl wszystkich pracowników tabeli PRACOWNICY pracujących na stanowisku sprzedawca oraz wszystkich pracowników pracujących w dziale technicznym.

Wskazówka:

```
SELECT IMIE, NAZWISKO, STANOWISKO, DZIAL
FROM PRACOWNICY
WHERE STANOWISKO='SPRZEDAWCA'
         OR DZIAL='TECHNICZNY';
```

Ćw. 10.

Napisz zapytanie, które zwróci wszystkich pracowników pracujących na stanowisku kierownika w dziale obsługi klienta oraz wszystkich pracowników z działu technicznego. Wiersze mają być uporządkowane wg działu a następnie wg nazwiska.

Wskazówka:

```
SELECT IMIE, NAZWISKO, STANOWISKO, DZIAL
FROM PRACOWNICY
WHERE STANOWISKO='KIEROWNIK'
         AND DZIAL='OBSLUGA KLIENTA'
         OR DZIAL='TECHNICZNY'
ORDER BY DZIAL, NAZWISKO;
```

W powyższym przykładzie widoczna jest wyższość operatora AND nad operatorem OR. Wprowadzając nawiasy możemy określić kolejność sprawdzania warunków.

Ćw. 11.

Napisz zapytanie, które zwróci wszystkich pracowników pracujących na stanowisku kierownika w dziale obsługi klienta lub w dziale technicznym.

Wskazówka:

```
SELECT IMIE, NAZWISKO, STANOWISKO, DZIAL
FROM PRACOWNICY
WHERE STANOWISKO='KIEROWNIK'
         AND (DZIAL='OBSLUGA KLIENTA'
         OR DZIAL='TECHNICZNY');
```

PREDYKAT IN

Predykat IN pozwala porównać wartości do wartości ze zbioru. Wartości typu znakowego, daty i czasu muszą być otoczone apostrofem.

Ćw. 12.

Wyświetl wszystkich pracowników tabeli PRACOWNICY pracujących na stanowisku sprzedawca lub kierownik.

Wskazówka:

```
SELECT IMIE, NAZWISKO, STANOWISKO, DZIAL
FROM PRACOWNICY
WHERE STANOWISKO IN ('SPRZEDAWCA','KIEROWNIK');
```

Ćw. 13.

Wyświetl wszystkie samochody tabeli SAMOCHODY o pojemności silnika 1400 lub 1600.

Wskazówka:

```
SELECT MARKA, TYP, ROK_PROD, POJ_SILNIKA
FROM SAMOCHODY
WHERE POJ_SILNIKA IN (1400,1600);
```

PREDYKAT BETWEEN

Predykat BETWEEN pozwala stwierdzić, czy dana wartości zawiera się miedzy dwiema wskazanymi wartościami

Ćw. 14.

Wyświetl dane samochodów, których pojemność silnika zawiera się między 1100 a 1800 cm sześciennych.

Wskazówka:

```
SELECT MARKA, TYP, ROK_PROD, POJ_SILNIKA
FROM SAMOCHODY
WHERE POJ_SILNIKA BETWEEN 1100 AND 1800;
```

```
Zapis  WHERE POJ_SILNIKA BETWEEN 1100 AND 1800;      można zastąpić
WHERE POJ_SILNIKA >=1100 AND POJ_SILNIKA<=1800;
```

WYBIERANIE WARTOŚCI NULL

Ćw. 15.

Wyświetl dane wszystkich klientów, którzy nie posiadają karty kredytowej.

Wskazówka:

```
SELECT IMIE, NAZWISKO, ULICA MIASTO
FROM KLIENCI
WHERE NR_KARTY_KREDYT IS NULL;
```

Ćw. 16.

Wyświetl dane wszystkich klientów, którzy posiadają kartę kredytową.

Wskazówka:

```
SELECT IMIE, NAZWISKO, ULICA MIASTO
FROM KLIENCI
WHERE NR_KARTY_KREDYT IS NOT NULL;
```

WYSZUKIWANIE CZĘŚCIOWE – PREDYKAT LIKE

Ćw. 17.

Wyświetl dane wszystkich klientów, których nazwiska zaczynają się na literę „K”.

Wskazówka:

```
SELECT IMIE, NAZWISKO, ULICA, MIASTO
FROM KLIENCI
WHERE NAZWISKO LIKE 'K%';
```

Ćw. 18.

Wyświetl dane wszystkich klientów, których nazwiska kończą się na „ski”.

Wskazówka:

```
SELECT IMIE, NAZWISKO, ULICA, MIASTO
FROM KLIENCI
WHERE NAZWISKO LIKE '%SKI';
```

Ćw. 19.

Wyświetl dane wszystkich klientów, których nazwiska zawierają literę „K” oraz „A” w wymienionym porządku.

Wskazówka:

```
SELECT IMIE, NAZWISKO, ULICA, MIASTO
FROM KLIENCI
WHERE NAZWISKO LIKE '%K%A%';
```

Ćw. 20.

Wyświetl dane wszystkich klientów, których nazwiska nie zaczynają się na literę „K”.

Wskazówka:

```
SELECT IMIE, NAZWISKO, ULICA, MIASTO
FROM KLIENCI
WHERE NAZWISKO NOT LIKE 'K%';
```

Ćw. 21.

Wyświetl dane wszystkich klientów, których nazwiska nie zaczynają się na literę „K” i nie zaczynają się na literę „D”.

Wskazówka:

```
SELECT IMIE, NAZWISKO, ULICA, MIASTO
FROM KLIENCI
WHERE NAZWISKO NOT LIKE 'K%'
    AND NAZWISKO NOT LIKE 'D%';
```

Ćw. 22.

Wyświetl dane wszystkich klientów, których nazwiska zawierają drugą literę „O” .

Wskazówka:

```
SELECT IMIE, NAZWISKO, ULICA, MIASTO
FROM KLIENCI
WHERE NAZWISKO  LIKE '_O%';
```

Ćw. 23.

Wyświetl dane wszystkich klientów, których nazwiska zawierają trzecią literę „C” .

Wskazówka:

```
SELECT IMIE, NAZWISKO, ULICA, MIASTO
FROM KLIENCI
WHERE NAZWISKO  LIKE '__C%';
```

II Wybieranie danych z wielu tabel

W celu złączenia dwóch lub większej ilości tabel:

- ✓ w klauzuli SELECT musimy wyspecyfikować kolumny, które chcemy zawrzeć w zapytaniu;
- ✓ w klauzuli FROM określamy nazwy złączonych tabel;
- ✓ w klauzuli WHERE określamy warunki złączenia.

Ćw. 24.

Wyświetl miejsca pracy pracowników z tabeli PRACOWNICY. Dane znajdują się w dwóch tabelach: PRACOWNICY i MIEJSCA (odczytaj numer miejsca pracy pracownika tabeli: PRACOWNICY, a następnie w tabeli: MIEJSCA znajdź odpowiadający temu numerowi wiersz, który opisuje dokładnie miejsce pracy, tzn. adres, telefon itp.)

Wskazówka:

```
SELECT PRACOWNICY. NAZWISKO,
       PRACOWNICY. STANOWISKO,
       PRACOWNICY. DZIAL,
       MIEJSCA. MIASTO,
       MIEJSCA. ULICA
FROM PRACOWNICY, MIEJSCA
WHERE PRACOWNICY.NR_MIEJSCA=MIEJSCA.NR_MIEJSCA;
```

Ćw. 25.

Uporządkuj dane z ćwiczenia poprzedniego według nazwiska.

Wskazówka:

```
SELECT PRACOWNICY. NAZWISKO,
       PRACOWNICY. STANOWISKO,
       PRACOWNICY. DZIAL,
       MIEJSCA. MIASTO,
       MIEJSCA. ULICA
FROM PRACOWNICY, MIEJSCA
WHERE PRACOWNICY.NR_MIEJSCA=MIEJSCA.NR_MIEJSCA
ORDER BY PRACOWNICY.NAZWISKO;
```

```
Przy złączaniu dwóch tabel klauzula WHERE musi zawierać jeden warunek.
Gdy złączamy trzy tabele, klauzule WHERE musi zawierać przynajmniej dwa
warunki.
```

Ćw. 27.

Wyświetl nazwiska, stanowiska, dział, w którym pracują osoby, wypożyczające samochody w Warszawie. Podaj również numer wypożyczenia.

Wskazówka:

```
SELECT WYPOZYCZENIA.NR_WYPOZYCZENIA,
       PRACOWNICY. NAZWISKO,
       PRACOWNICY. STANOWISKO,
       PRACOWNICY. DZIAL,
       MIEJSCA. MIASTO,
       MIEJSCA. ULICA
FROM PRACOWNICY, MIEJSCA, WYPOZYCZENIA
WHERE PRACOWNICY.NR_MIEJSCA=MIEJSCA.NR_MIEJSCA
       AND PRACOWNICY.NR_PRACOWNIKA=WYPOZYCZENIA.NR_PRACOW_WYP
       AND MIEJSCA.MIASTO='WARSZAWA'
ORDER BY PRACOWNICY.NAZWISKO;
```

Ćw. 28.

Podaj nazwiska pracowników, którzy pracują na stanowisku sprzedawcy oraz miejsca w których pracują.

Wskazówka:

```
SELECT PRACOWNICY. NAZWISKO,
       PRACOWNICY. STANOWISKO,
       PRACOWNICY. DZIAL,
       MIEJSCA. MIASTO,
       MIEJSCA. ULICA
FROM PRACOWNICY JOIN
      MIEJSCA ON
      PRACOWNICY.NR_MIEJSCA=MIEJSCA. NR_MIEJSCA
WHERE PRACOWNICY. STANOWISKO='SPRZEDAWCA'
ORDER BY PRACOWNICY. NAZWISKO;
```

Gdy używamy słowa JOIN w klauzuli FROM, warunki złączenia muszą być wyspecjalizowane po klauzuli ON W klauzuli WHERE można określić dodatkowe warunki (jak w ćwiczeniu 28)

STOSOWANIE ALIASÓW W ZAPYTANIU

Aliasy definiuje się w celu skrócenia nazwy tabeli; np. alias P wskazuje na tabelę PRACOWNICY.

Ćw. 29.

Podaj nazwiska pracowników, którzy pracują na stanowisku sprzedawcy oraz miejsca w których pracują. Zapytanie zapisz używając aliasów.

Wskazówka:

```
SELECT P. NAZWISKO,
       P. STANOWISKO,
       P. DZIAL,
       M. MIASTO,
       M. ULICA
FROM PRACOWNICY P,
      MIEJSCA M
WHERE P.NR_MIEJSCA=M.NR_MIEJSCA
   AND P.STANOWISKO='SPRZEDAWCA'
ORDER BY P.NAZWISKO;
```

III Funkcje skalarne i arytmetyczne

WYBIERANIE WYLICZONYCH WARTOŚCI

Ćw. 30.

Oblicz wartość pensji wraz z dodatkiem dla pracowników, których pensja jest większa niż 1100.

Wskazówka:

```
SELECT P.IMIE,
       P.NAZWISKO,
       P.PENSJA,
       P.DODATEK,
       P.PENSJA+P.DODATEK
FROM   PRACOWNICY P
WHERE P.PENSJA >1100
```

```
ORDER BY P.NAZWISKO;
```

NAZYWANIE WYLICZONEJ KOLUMNY

Ćw. 31.
Kolumnie z ćwiczenia 30 nadamy nazwę DO_WYPLATY

Wskazówka:

```
SELECT P.IMIE,
       P.NAZWISKO,
       P.PENSJA,
       P.DODATEK,
       P.PENSJA+P.DODATEK AS DO_WYPLATY
FROM   PRACOWNICY P
WHERE P.PENSJA >1100
ORDER BY P.NAZWISKO;
```

Ćw. 31.
Nazwę tabeli zawierającej wyliczenia możemy nadać w inny sposób, np.:

Wskazówka:

```
SELECT P.IMIE,
       P.NAZWISKO,
       P.PENSJA,
       P.DODATEK,
       P.PENSJA+P.DODATEK AS "DO WYPLATY"
FROM   PRACOWNICY P
WHERE P.PENSJA >1100
ORDER BY P.NAZWISKO;
```

PORÓWNANIA DATY

Ćw. 32.
Wyświetl dane pracowników zatrudnionych w lub po dacie 1998-01-01.

Wskazówka:

```
SELECT P.IMIE,
       P.NAZWISKO,
       P.DZIAL,
       P.STANOWISKO,
       P.DATA_ZATR
FROM   PRACOWNICY P
WHERE P.DATA_ZATR >='1998-01-01'
ORDER BY P.NAZWISKO;
```

IV Funkcje kolumnowe i grupujące (funkcje operujące na kolumnach).

FUNKCJE KOLUMNOWE

SUM –oblicza sumę wartości w określonych kolumnach,
AVG – oblicza średnią wartość w kolumnie,
MIN – znajduje minimalną wartość,
MAX – znajduje maksymalną wartość,
COUNT – służy do zliczania wystąpień pewnej wartości w wierszach.

Ćw. 33.
Wyświetl całkowitą sumę wszystkich pensji pracowników, średnią pensję, minimalną i maksymalną pensję oraz ilość pracowników.

Wskazówka:

```
SELECT SUM(P.PENSJA) AS PENSJA,
       AVG(P.PENSJA) AS SREDNIA,
       MIN(P.PENSJA) AS PENSJA_MIN,
       MAX(P.PENSJA) AS PENSJA_MAX,
       COUNT(*) AS ILOSC
 FROM PRACOWNICY P;
```

Ćw. 34.
Wyświetl liczbę działów i stanowisk w firmie

Wskazówka:

```
SELECT COUNT(DISTINCT P.DZIAL) AS ILOSC_DZIALOW,
       COUNT(DISTINCT P.STANOWISKO) AS ILOSC_STANOWISK
 FROM PRACOWNICY P;
```

Ćw. 35.
Wyświetl całkowitą sumę wszystkich pensji pracowników, średnią pensję, minimalną i maksymalną pensję oraz ilość pracowników pracujących w dziale obsługi klienta.

Wskazówka:

```
SELECT SUM(P.PENSJA) AS PENSJA,
       AVG(P.PENSJA) AS SREDNIA,
       MIN(P.PENSJA) AS PENSJA_MIN,
       MAX(P.PENSJA) AS PENSJA_MAX,
       COUNT(*) AS ILOSC
 FROM PRACOWNICY P
WHERE P.DZIAL='OBSLUGA KLIENTA';
```

KLAUZULA GROUP BY

Ćw. 36.
Pogrupuj wiersze w tabeli pracownicy według stanowiska. Dla każdej grupy oblicz łączną pensję, pensję średnią, największą, najmniejszą oraz wyświetl liczbę osób pracujących na danym stanowisku.

Wskazówka:

```
SELECT P.STANOWISKO,
   SUM(P.PENSJA) AS PENSJA,
   AVG(P.PENSJA) AS SREDNIA,
```

```
   MIN(P.PENSJA) AS PENSJA_MIN,
   MAX(P.PENSJA) AS PENSJA_MAX,
       COUNT(*) AS ILOSC
 FROM PRACOWNICY P
GROUP BY P.STANOWISKO
ORDER BY P.STANOWISKO;
```

KLAUZULA HAVING

Klauzula HAVING używana jest tylko w połączeniu z klauzulą GROUP BY w celu ograniczenia wyświetlania grup. Po zgrupowaniu wierszy przez klauzulę GROUP BY, klauzula HAVING wyświetla tylko te wiersze spośród zgrupowanych, które spełniają warunki wyszczególnione w klauzuli HAVING.

Ćw. 37.

Wyświetl nazwiska wszystkich pracowników, którzy wypożyczyli samochody na łączną jednostkową wartość powyżej 400 zł.

Wskazówka:

```
SELECT P.NAZWISKO,
   SUM(W.CENA_JEDN)
FROM PRACOWNICY P,
     WYPOZYCZENIA W
  WHERE P.NR_PRACOWNIKA=W.NR_PRACOW_WYP
  GROUP BY NAZWISKO
   HAVING SUM(W.CENA_JEDN)>400
  ORDER BY P.NAZWISKO;
```

V Klauzula UNION

Klauzula UNION pozwala na łączenie dwóch lub więcej wyników wykonania zapytania SELECT i jednocześnie usuwa duplikaty. Nazwy łączonych kolumn mogą być różne, ale muszą mieć tę samą szerokość i typ danych.
Klauzula UNION ALL wyświetla również powtarzające się wiersze.

ŁĄCZENIE WIELU WYNIKÓW ZAPYTANIA

Ćw. 38.

Wyświetl imiona i nazwiska wszystkich pracowników i klientów, których nazwiska kończą się na „ski”.

Wskazówka:

```
SELECT P.NAZWISKO,
   SUM(W.CENA_JEDN)
FROM PRACOWNICY P,
     WYPOZYCZENIA W
  WHERE P.NR_PRACOWNIKA=W.NR_PRACOW_WYP
  GROUP BY NAZWISKO
   HAVING SUM(W.CENA_JEDN)>400
  ORDER BY P.NAZWISKO;
```

Ćw. 39.

Posortuj malejąco według nazwiska dane otrzymane w poprzednim ćwiczeniu.

Wskazówka:

```
SELECT IMIE,
       NAZWISKO
  FROM KLIENCI
WHERE NAZWISKO LIKE '%SKI'
UNION
SELECT IMIE,
       NAZWISKO
  FROM PRACOWNICY
WHERE NAZWISKO LIKE '%SKI'
ORDER BY 2 ;
```

(uwaga: - ORDER BY 2 oznacza, że sortowanie odbywa się wg drugiej kolumny – tutaj nie można użyć nazwy kolumny, tylko jej numer.)

VI Podzapytania

UŻYWANIE PODZAPYTAŃ

Ćw. 40.

Wyświetl imiona i nazwiska, dział i stanowisko pracowników, którzy otrzymują wynagrodzenie na kwotę wiekszą niż wynosi średnia. Należy najpierw sprawdzić, jaka jest średnia dla każdego pracownika.

Wskazówka:

```
SELECT AVG(P.PENSJA)
FROM PRACOWNICY P;
```

A teraz szukamy pracowników, którzy zarabiają poniżej średniej obliczonej wyżej.

```
SELECT P.IMIE,
       P.NAZWISKO,
       P.DZIAL,
       P.STANOWISKO
  FROM PRACOWNICY P
   WHERE P.PENSJA>1530;
```

Ćw. 41.

Wyświetl wyniki ćwiczenia 40 używając tylko jednego polecenia.

Wskazówka:

```
SELECT P. IMIE,
       P.NAZWISKO,
       P. DZIAL,
       P. STANOWISKO
  FROM PRACOWNICY P
WHERE P.PENSJA>(SELECT AVG (P.PENSJA)FROM PRACOWNICY P);
```

PODZAPYTANIE Z UŻYCIEM SŁOWA KLUCZOWEGO IN

Słowo kluczowe IN pozwala na zidentyfikowanie wszystkich elementów w zbiorze A, które nie występują w zbiorze B

Ćw. 42.

Wyświetl listę samochodów, których do tej pory nie wypożyczył żaden klient. (Zapytanie wybiera te samochody, które nie znajdują się w tablicy WYPOZYCZENIA, czyli te, które nie były do tej pory wypożyczone.

Wskazówka:

```
SELECT S.NR_SAMOCHODU,
       S.MARKA,
       S.TYP
FROM SAMOCHODY S
WHERE S. NR_SAMOCHODU
NOT IN (SELECT W. NR_SAMOCHODU FROM WYPOZYCZENIA W);
```

PODZAPYTANIA Z UŻYCIEM SŁOWA KLUCZOWEGO ALL

Ćw. 43.

Wyświetl listę osób, których pensja jest większa od średniej pensji w dziale, w którym pracownik jest zatrudniony

Wskazówka:

```
SELECT P.IMIE,
       P.NAZWISKO,
       P.DZIAL,
       P.STANOWISKO,
       P.PENSJA
  FROM PRACOWNICY P
  WHERE
P.PENSJA>ALL(SELECT AVG(P.PENSJA) FROM PRACOWNICY P
             GROUP BY DZIAL);
```

PODZAPYTANIA Z UŻYCIEM SŁOWA KLUCZOWEGO ANY (LUB równoważne SOME

Ćw. 44.

Wyświetl listę osób, których pensja jest większa od najmniejszej średniej pensji średniej pensji w dziale, w którym pracownik jest zatrudniony

Wskazówka:

```
SELECT P.IMIE,
       P.NAZWISKO,
       P.DZIAL,
       P.STANOWISKO,
       P.PENSJA
  FROM PRACOWNICY P
  WHERE
P.PENSJA>ANY (SELECT AVG(P.PENSJA) FROM PRACOWNICY P
             GROUP BY DZIAL);
```

PODZAPYTANIA W KLAUZULI HAVING

Ćw. 45.

Wyświetl działy, których średnia pensja pracowników jest wyższa od średniej pensji w firmie. Do średniej pensji nie będą brani pod uwagę kierownicy działów.

Wskazówka:

```
SELECT P. DZIAL,
      AVG(P.PENSJA) AS SREDNIA_PENSJA
 FROM PRACOWNICY P
 WHERE P.STANOWISKO <>'KIEROWNIK'
GROUP BY P.DZIAL
HAVING AVG(P.PENSJA)>(SELECT AVG(P.PENSJA)FROM PRACOWNICY P
        WHERE P.STANOWISKO<>'KIEROWNIK')
ORDER BY 2;
```

VII Utrzymywanie danych

TYPY DANYCH

W różnych systemach relacyjnej bazy danych inaczej nazywają się typy danych. Jednak ich zakres i typ jest często identyczny. Każdy system relacyjnej bazy danych posiada w swojej dokumentacji sekcję, która opisuje typ danych używanych w tym systemie. Poniżej znajdują się przykładowe typy danych wraz z ich opisem.

- ✓ Numeryczne typy danych

SMALLINT	liczby całkowite z przedziału -32768 do 32767
INTEGER	liczby całkowite z przedziału -2 147 483 648 do 2 147 483 647
DECIMAL(m, n)	liczby rzeczywiste, gdzie m oznacza liczbę cyfr, a n oznacza liczbę cyfr po przecinku

- ✓ Znakowe typy danych

CHAR(n)	typ znakowy o stałej długości (maksymalnie 255 znaków)
VARCHAR(n)	typ znakowy o zmiennej długości

- ✓ Typy danych daty i czasu

DATE	typ daty
TIME	typ czasu

TWORZENIE TABEL

Ćw. 46.

Utwórz tabelę o nazwie KLIENCI_TEST, która ma następujące pola: NR_KLIENTA, IMIE, NAZWISKO, NR_KARTY ULICA, NUMER (domu), MIASTO, KOD, NR_TELEFONU.

Wskazówka:

```
CREATE TABLE KLIENCI_TEST (
NR_KLIENTA      CHAR(8)     NOT NULL,
IMIE            VARCHAR(20) NOT NULL,
NAZWISKO        VARCHAR(20) NOT NULL,
NR_KARTY_KREDYT CHAR(20)      ,
```

```
ULICA           VARCHAR(24) NOT NULL,
NUMER           CHAR(8)     NOT NULL,
MIASTO          VARCHAR(24) NOT NULL,
KOD             CHAR(6)     NOT NULL,
NR_TELEFONU     CHAR(16),
PRIMARY KEY (NR_KLIENTA));
```

DODAWANIE KOLUMN DO ISTNIEJĄCEJ TABELI

Ćw. 47.
Do tabeli KLIENCI_TEST dodaj kolumny : FIRMA oraz NIP.

Wskazówka:

```
ALTER TABLE KLIENCI_TEST
       ADD FIRMA  VARCHAR(40),
       ADD NIP  CHAR(12);
```

TWORZENIE WIDOKÓW

Dane zawarte w widoku nie są fizycznymi danymi a danymi należącymi do tabeli lub kilku tabel, z których widok czerpie dane. Widoki przede wszystkim są tworzone w celu ograniczenia dostępu do danych w tabelach bazy danych.

Ćw. 47.
Utwórz widok zawierający dane klientów, którzy posiadają firmę.

Wskazówka:

```
CREATE VIEW  KLIENCI_FIRMY AS
SELECT K.IMIE, K.NAZWISKO, K.FIRMA, NIP, K.MIASTO
FROM KLIENCI K
WHERE K.FIRMA IS NOT NULL;
```

Ćw. 48.
Wybierz dane z widoku utworzonego w ćwiczeniu 47.

Wskazówka:

```
SELECT *
FROM KLIENCI_FIRMY;
```

Ćw. 49.
Utwórz widok, który ogranicza dane pracowników do wszystkich danych oprócz informacji na temat dodatku i pensji.

Wskazówka:

```
CREATE VIEW V_PRACOWNICY AS
       SELECT P.NR_PRACOWNIKA,
              P.IMIE,
              P.NAZWISKO,
              P.DATA_ZATR,
              P.DZIAL,
              P.STANOWISKO,
              P.NR_MIEJSCA,
              P.NR_TELEFONU
```

```
FROM  PRACOWNICY P;
```

Ćw. 50.

Wyświetl dane z widoku ćwiczenia 49.

Wskazówka:

```
SELECT *
FROM V_PRACOWNICY;
```

DODAWANIE REKORDÓW

Ćw. 51.

Do tabeli KLIENCI_TEST dodaj nowy rekord (wcześniej upewnij się, czy w tej tabeli jest kolumna o nazwie FIRMA oraz NIP.

Wskazówka:

```
INSERT INTO KLIENCI_TEST
VALUES ('00000031', 'MARIUSZ', 'DOLATA', NULL, 'KWIATY', 'KOCHANOWSKIEGO',
'3', 'GLIWICE', '44-100', '2345671', NULL);
```

(wpisać wartości w kolejności występowania kolumn w tabeli `KLIENCI_TEST`)

ABY WPISANE WARTOŚCI ZOSTAŁY ZAPISANE W BAZIE DANYCH, NALEŻY TRANSAKCJĘ POTWIERDZIĆ, WCISKAJĄC YES, GDY POJAWI SIĘ OKNO DIALOGOWE Z PYTANIEM, LUB SAMODZIELNIE ZATWIERDZIĆ TRANSAKCJĘ, WYBIERAJĄC W INTERACTIVE SQL – TRANSACTIONS/COMMIT

Ćw. 52.

Aby wstawić dane tylko do wybranych kolumn, należy je określić, a następnie podać wartość.
Do tabeli KLIENCI_TEST wstawić wartości tylko do wybranych kolumn (obowiązkowo należy wybrać te kolumny, dla których jest wymagane wstawienie wartości)

Wskazówka:

```
INSERT INTO KLIENCI_TEST (NR_KLIENTA, IMIE,
                          NAZWISKO,
                          ULICA,
                          NUMER,
                          MIASTO,
                          KOD)
VALUES ('00000006', 'ALA', 'LIPIŃSKA', 'SZYMANOWSKIEGO', '5',
       'WARSZAWA', '23-123');
```

Rekordy do tabeli można wstawiać również z poziomu IBConsole. Wybieramy: TABLE/NAZWA TABELI DO KTÓREJ CHCEMY DODAĆ REKORD/DATA i na dole okna wciskamy +. Powrót do IBConsole następuje poprzez wciśnięcie TRL+F4 lub wybranie ZAMKNIJ przy wciśnięciu ikony znajdującej się po lewej stronie nazwy okna PROPERTIES FOR:

Ćw. 53.

Do tabeli KLIENCI_TEST dodaj w jednym poleceniu wszystkich klientów tabeli KLIENCI, którzy nie posiadają firmy

Wskazówka:

```
INSERT INTO KLIENCI_TEST (NR_KLIENTA,
                          IMIE,
                          NAZWISKO,
                          ULICA,
                          NUMER,
                          MIASTO,
                          KOD)
SELECT    NR_KLIENTA,
          IMIE,
          NAZWISKO,
          ULICA,
          NUMER,
          MIASTO,
          KOD
      FROM KLIENCI
      WHERE FIRMA IS NULL;
```

USUWANIE REKORDÓW

Ćw. 54.
Usuń rekord z tabeli KLIENCI_TEST.

Wskazówka:

```
DELETE FROM KLIENCI_TEST
WHERE IMIE='JOANNA';
```

Ćw. 55.
Usuń wszystkie rekordy z tabeli KLIENCI_TEST.

Wskazówka:

```
DELETE FROM KLIENCI_TEST;
```

ZMIENIANIE DANYCH W TABELI

Ćw. 56.
Zwiększ kwotę dodatku pracowników zatrudnionych na stanowisku sprzedawcy o 50 zł. .

Wskazówka:

```
UPDATE PRACOWNICY
SET DODATEK=DODATEK+50
WHERE STANOWISKO='SPRZEDAWCA';
```

Ćw. 57.
Zwiększ kwotę dodatku dla kierowników o 30 zł. Oraz zwiększa pensje (kierowników) o 10%.

Wskazówka:

```
UPDATE PRACOWNICY
SET DODATEK=DODATEK+30,
    PENSJA=PENSJA*1.1
WHERE STANOWISKO='KIEROWNIK';
```

USUWANIE TABEL

Ćw. 58.
Usuń tabelę KLIENCI_TEST.

Wskazówka:

```
DROP TABLE KLIENCI_TEST;
```
(tabela nie może być używana).

VIII Ograniczenia i integralność referencyjna

OGRANICZENIA

Można zdefiniować ograniczenie sprawdzające poprawność wpisywanych danych do tabeli poprzez określenie warunku sprawdzającego **CHECK**.

Ćw. 59.
Wpisz ograniczenie w tabeli PRACOWNICY zapobiegające wpisaniu kwoty dodatku większej od kwoty pensji.

Wskazówka:

```
ALTER TABLE PRACOWNICY
ADD CHECK (PENSJA>DODATEK);
```

Ćw. 60.
Napisz wyrażenie dodające wiersz do tabeli PRACOWNICY, który będzie zawierał w kolumnie DODATEK wartość większą niż w kolumnie PENSJA.

Wskazówka:

```
INSERT INTO PRACOWNICY
VALUES ('0011', 'ANNA', 'NOWAKOWSKA',
        '1999-05-01', 'OBSLUGA KLIENTA',
        'SPRZEDAWCA', 1100, 1200, '000001',
        '432-345-123');
```
(oczywiście wartości nie zostaną przyjęte)

INTEGRALNOŚĆ DANYCH – KLUCZ GŁÓWNY

Ćw. 61.
Sprawdź, w jaki sposób określa się klucz główny tabeli na podstawie skryptu tabeli PRACOWNICY.

Określenie klucza głównego spowoduje, że nie zostaną wprowadzone dwa identyczne wiersze. Klucz główny może składać się z więcej niż jednej kolumny.

INTEGRALNOŚĆ REFERENCYJNA – KLUCZ OBCY

Definiowanie kluczy obcych jest sposobem łączenia danych przechowywanych w różnych tabelach bazy danych. Klucze obce chronią wiersze z tabeli przed „osieroceniem" na wypadek, usunięcia jakiegokolwiek wiersza z tabeli klucza głównego. Aby zapewnić taką ochronę należy zdefiniować klucze obce we wszystkich tabelach, które odwołują się do innych tabel

Ćw. 62.

Napisz wyrażenie , które ustawia klucz obcy w tabeli PRACOWNICY w kolumnie NR_MIEJSCA.

Wskazówka:

```
ALTER TABLE PRACOWNICY
ADD FOREIGN KEY (NR_MIEJSCA)
REFERENCES MIEJSCA (NR_MIEJSCA) ON DELETE NO ACTION;
```

Ostatnie polecenie informuje, że nie można usunąć wiersza z tabeli MIEJSCA, gdy istnieje wiersz do niego się odwołujący w tabeli PRACOWNICY. Aby usunąć dane o adresie z tabeli MIEJSCA, najpierw należy usunąć wszystkich pracowników pracujących w miejscu, o którym informacje chcemy usunąć.
Zamiast **NO ACTION** możemy wpisać **CASCADE**, co spowoduje, że gdy usuwamy wiersze z tabeli MIEJSCA, to są jednocześnie usuwane wszystkie wiersze z danymi o pracownikach, którzy pracują w usuwanym miejscu.
Wstawienie wyrażenia **SET NUL** (w miejsce NO ACTION) mówi, że jeżeli usuwamy dane o miejscach, to w tabeli PRACOWNICY w kolumnie NR_MIEJSCA zostanie wstawiona wartość NULL.