Podstawy języka SQL

Co to jest SQL?

Structured Query Language uchodzi za standard języka zapytań kierowanych do systemu zarządzania bazą danych. SQL jest językiem deklaratywnym tj. takim, w którym istotne jest wykonanie postawionego zadania, a nie sposób jego wykonania. SQL nie jest językiem proceduralnym tzn. nie posiada konstrukcji takich jak pętle, wykonywania instrukcji.

Możliwości SQL

- wybieranie danych
- wstawianie danych
- modyfikowanie danych
- usuwanie danych
- tworzenie obiektów bazy danych
- modyfikowanie obiektów bazy danych
- usuwanie obiektów bazy danych
- zarządzanie uprawnieniami dostępu do danych i do obiektów bazy danych
- utrzymanie spójności danych

SQL*Plus

- narzędzie umożliwiające wprowadzanie i wykonywanie

- instrukcji SQL
- skryptów SQL
- instrukcji SQL*Plus
- bloków PL/SQL
- dokonywanie czynności administracyjnych

Podstawowe polecenia SQL*Plus

- START plik lub @plik - Wykonanie skryptu poleceń SQL lub PL/SQL
- EDIT plik - Wywołanie standardowego edytora
- CONNECT użytkownik/hasło@opis_serwera – Umożliwienie dokonania przyłączenia się do określonego serwera bazy danych bazy danych
- SPOOL plik - Zapisanie polecenia i wyniku do pliku
- SPOOL OFF - Zamknięcie pliku otwartego przez SPOOL
- HOST polecenie - Wykonaj polecenie systemu operacyjnego
- EXIT - Wyjście z programu
- DESCRIBE nazwa_tabeli – Wyświetlenie opisu tabeli
- DESCRIBE nazwa_procedury - Wyświetlenie opisu procedury
- DESCRIBE nazwa_funkcji - Wyświetlenie opisu funkcji
- EXECUTE nazwa(...) - Wykonanie procedury nazwa
- SHOW ERRORS - Pokazanie błędów ostatnio wykonanego polecenia PL/SQL

Zmienne podstawiania

Zmienne podstawiania służą umożliwieniu nadania zmiennym wartości.

Oznaczenie symbolu (ciągu znaków) jako zmiennej realizowane jest przez poprzedzenie go znakiem & (zmienna chwilowa) lub && (zmienna sesji).

Przykłady:

&parametr1

&&przelicznik

Do wyświetlenia zmiennych sesji służy polecenie DEFINE.

Do kasowania zmiennych sesji służy polecenie UNDFINE.

Zapytania proste

Podstawowym zapytaniem kierowanym do systemu zarządzania bazą danych jest polecenie wybrania zadanego zestawu informacji.

Podstawowe zdanie SELECT

W najprostszej formie zdania SELECT zawiera ono polecenie wybrania zestawu danych z określonego przez FROM źródła danych. Zdanie może być pisane w wielu liniach. Zdanie kończy się średnikiem.

Przykłady:

*SELECT * FROM department;*

SELECT regional_group FROM location;

Zadania

wybrać wszystkie stanowiska pracy z tabeli JOB

wybrać nazwiska i imiona wszystkich pracowników

Rozszerzenia zdania SELECT - Alias

Alias jest to jedno lub wiele słów, które pojawią się w nagłówku zamiast specyfikacji kolumny.

Przykłady:

SELECT last_name Nazwisko, hire_date “Data zatr”
FROM employee;

SELECT name Departamenty
FROM department;

Zadania

wybrać dostępne produkty i opisać jako Produkty

wybrać nazwy klientów i opisać jako Nazwa klienta

Rozszerzenia zdania SELECT - Wyrażenia arytmetyczne

Wyrażenie jest zastosowaniem operacji arytmetycznych (użycia funkcji, operatorów dla kolumn) co umożliwia przeliczenie wartości kolumn.

Operatorami mogą być: *****, **/**, **+** i **-**.

Przykłady wyrażeń

Przykłady:

zmiana wartości wyświetlanej ceny minimalnej – podanie w centach

*SELECT 100 * min_price FROM price;*

Zadania

Przeliczyć zarobki pracowników na tysiące dolarów

Rozszerzenia zdania SELECT - Literały

Literały są ciągami znaków, które można wyświetlić w wyniku wykonania SELECT deklarując je w zapytaniu.

Przykłady:

SELECT 'Pracownik ' ,last_name, ' pracuje od ' ,
hire_date FROM employee;

SELECT 'Klient ', name,' pochodzi z ', city
FROM customer;

Zadania

uzyskać zdania opisujące zarobki

Rozszerzenia zdania SELECT - Konkatenacja

Konkatenacja || służy połączeniu zawartości dwóch lub więcej wartości tekstowych w funkcjonalnie jedno. W przypadku, gdy konkatenacji ma podlegać liczba lub data – zostaje ona poddana domyślnej konwersji na ciąg znaków. W przypadku, gdy konkatenacji ma podlegać wyrażenie arytmetyczne – musi być umieszczone w nawiasie.

Przykłady:

SELECT last_name || ' ' || first_name "Pracownicy" FROM employee;

SELECT start_date || ' – ' || end_date || ' Cena: ' || list_price "Okresy obowiązywania cen"
FROM price;

Zadania

uzyskać wiersze z jedną kolumną podstawowych informacji o pracownikach „Pracownik ... ... zarabia ..."

NULL

NULL – wartość nieokreślona funkcjonalnie różna od liczbowego 0.
UWAGA: wyrażenie arytmetyczne z null daje null,
konkatenacja wartości z null daje wartość

Przykłady:

Wybranie okresów obowiązywania cen :

SELECT start_date, NVL(end_date , sysdate), list_price
FROM price;

Zadania

Wybrać premie pracowników, tam gdzie nie ma wyświetlić 0.

DISTINCT

Użycie DISTINCT realizuje zapewnienie unikalności wybranego zestawu danych w zakresie otrzymanych w zapytaniu wierszy.

Przykłady:

Wybranie zbioru miast klientów

SELECT distinct city FROM customers;

Zadania

Określić zbiór identyfikatorów osób, które zajmują się obsługą klientów.

ORDER BY

ORDER BY użyte jako ostatni element zapytania z wyspecyfikowanymi kolumnami (wyrażeniami) i porządkiem zapytania określa sposób w jaki posortowane zostaną otrzymane wiersze.

Porządek (rosnący lub malejący) zapewniany jest przez odpowiednio ASC, DESC, przy czym domyślnym jest ASC.

Kolumny o wartości NULL traktowane są ostatnie w przypadku sortowania rosnącego i vice versa w przypadku malejącego.

Określając kolumny można w ORDER BY posłużyć się ich kolejnością specyfikując pierwszą, która ma być wyświetlona jako 1, drugą jako 2 itd.

Przykłady

Alfabetyczny spis pracowników

```
SELECT last_name, first_name
FROM  employee
ORDER BY last_name, first_name ;
```

Zadania

Spis pracowników od największego stażu do najmniejszego

Wykaz historia zmian cen produktów

WHERE

Zadaniem WHERE w przypadku zapytania prostego jest zastosowanie warunków na wybrany zestaw wierszy.

Warunkiem logicznym jest użycie operatora logicznego takiego jak:

= porównanie (UWAGA – nie można porównywać wartości NULL)

> sprawdzenie większości

>= sprawdzenie większości lub równości

< sprawdzenie mniejszości

<= sprawdzenie mniejszości lub równości

<> sprawdzenie różności

!= sprawdzenie różności

Przykłady:

Wybranie wszystkich pracowników Departamentu o kodzie 10.

SELECT last_name FROM employee WHERE department_id = 10;

Wybranie wszystkich klientów, którzy pochodzą z D*allas*.

SELECT name FROM customers WHERE city = 'DALLAS' ;

Znalezienie tych pracowników, którzy mają premię,

SELECT last_name FROM employee WHERE

nvl(commission, 0) > 0;

Zadania

Znaleźć identyfikatory produktów i ich aktualne ceny

LIKE, IN, BETWEEN, IS NULL

Warunkiem logicznym może być również użycie operatora SQL takiego jak:

BETWEEN ... AND ... sprawdzenie zawierania się w przedziałach domkniętych

IN (..., ..., ...) sprawdzenie zawierania się w wyspecyfikowanym zbiorze

LIKE ... sprawdzenie podobieństwa do wzorców określonych za pomocą następujących kryteriów zbudowanych z użyciem symboli wieloznacznych

- % dowolny ciąg – w tym również pusty
- _ (podkreślenie) - dokładnie jeden znak

IS NULL sprawdzenie czy wartość jest nieokreślona

Przykłady:

Wybranie tych pracowników, którzy są z działu 10 lub 20

SELECT last_name FROM employee WHERE department_id IN(10, 20);

Wybranie osób z imionami zaczynającymi się na literę ‘R’

SELECT last_name, first_name FROM employee WHERE first_name LIKE ‘R%’;

Wybranie osób, których zarobki są między 1000 i mniejsze od 2000.

SELECT imie, nazwisko FROM pracownicy_dt
WHERE salary BETWEEN 1000 AND 2000;

Wybranie cen i identyfikatorów tych produktów, których ceny są aktualne .

SELECT product_id, list_price FROM price WHERE end_date IS NULL;

Zadania

Wybranie szefa.

Wybranie produktów, które są koloru żółtego.

Wybranie tych klientów, którzy pochodzą ze stanu CA lub NY.

NOT

NOT stosowane jest przed warunkami logicznymi z użyciem operatorów logicznych i przed operatorami SQL

Przykłady

Wybranie tych pracowników, którzy nie są z działu 10 ani z 20

SELECT last_name FROM employee
WHERE department_id NOT IN (10, 20);

Wybranie osób z imionami zaczynającymi się na literę ‘R’

SELECT last_name, first_name FROM employee WHERE first_name NOT LIKE ‘R%’;

Wybranie nieaktualnych cen i identyfikatorów produktów

SELECT product_id, list_price FROM price WHERE end_date IS NOT NULL;

Zadania

Wybrać pracowników, którzy są podwładnymi

Wybrać nazwy produktów, które nie są piłkami

Wybrać klientów nie z Dallas, ani z Nowego Yorku.

Hierarchia warunków logicznych

Warunki logiczne mogą być łączone za pomocą operatorów sumy logicznej OR lub iloczynu logicznego AND. AND ma pierwszeństwo nad OR dlatego należy stosować nawiasy w przypadku konieczności uzyskania innego porządku sprawdzania warunku.

Pełna hierarchia (odzwierciedlająca pierwszeństwo) wygląda następująco:

- =, !=, <>, >=, <= , BETWEEN AND , IN, LIKE, IS NULL;
- NOT
- AND
- OR

Przykłady

Wybór pracowników z działo 10 lub 20 lub 30, którzy zarabiają poniżej 1000 lub powyżej 2000

```
SELECT first_name, last_name FROM employee
WHERE salary NOT BETWEEN 1000 AND 2000
AND department_id IN ( 10, 20, 30 );
```

Zadania

Uzyskać identyfikatory produktów, które nie są kolory służą do uprawiania tenisa i nie są to piłki

Wybrać nazwiska i numery działów pracowników, którzy zarabiają poniżej 2500, ale takich, że:

mają premię powyżej 1000

lub

są z działu 10

Operatory zbiorowe

Wiersze danych, otrzymywane przez wykonanie zapytań mogą być ze sobą zestawiane w układzie operacji na zbiorach. Operatory zbiorowe traktują efekty wykonania zdań SELECT (w tym na różnych tabelach) jak zbiory miedzy którymi zachodzi działanie. Służą do tego operatory łączące zdania SELECT takie jak: UNION, INTERSECT i MINUS.

Zasady stosowania operatorów grupowych

Składnia wynikająca ze stosowania operatorów mnogościowych niesie za sobą pewne konsekwencje budowania zdań:

- Wiersze zwracane przez poszczególne zapytania muszą odpowiadać sobie pod względem typów i ilości.
- Klauzula ORDER BY stosowana jest po ostatnim zapytaniu i kolumny są w niej reprezentowane w postaci liczb określających ich kolejność w wierszu.
- Nagłówki rezultatu są tworzone na podstawie pierwszego zdania SELECT.

UNION

Dzięki UNION, rozdzielającemu zapytania można otrzymać mnogościową sumę wierszy, tzn. taką gdzie występuje każdy z wierszy zwróconych w poszczególnych zapytaniach, ale brak jest powtórzeń wierszy. Uzyskanie wierszy bez eliminacji powtórzeń umożliwia operator UNION ALL.

Przykład:

```
SELECT first_name, last_name FROM employee WHERE commission IS NULL
UNION
SELECT first_name, last_name FROM employee WHERE commission IS NOT NULL;
```

Zadanie:

Wybrać opisy (nazwisko i imię dla pracowników i nazwy dla firm będących klientami) w postaci jednej listy oznaczając, kto jest pracownikiem, a kto klientem

INTERSECT

Operator INTERSECT umożliwia otrzymanie iloczynu mnogościowego wierszy poszczególnych zapytań, tzn. łącznie zwrócone zostaną tylko te wiersze, które występują w zapytaniach, bez powtórzeń.

Przykład:

```
SELECT first_name, last_name FROM employee
WHERE commission IS NOT NULL
INTERSECT
SELECT first_name, last_name FROM employee
WHERE salary > 1500;
```

Zadanie:

Wybrać te miasta, w których są siedziby i klientów i departamenty firmy

MINUS

Operator MINUS daje możliwość mnogościowego wykonania różnicy zbiorów, tj. zwrócenia tylko tych wierszy z zapytania poprzedzającego operator MINUS, których brak w zapytaniu następującym po operatorze.

Przykład:

```
SELECT first_name, last_name, commission  FROM employee
MINUS
SELECT first_name, last_name, commission FROM employee
WHERE nvl(commission ,0) = 0;
```

Zadanie:

Wybrać te miasta, w których są departamenty firmy ale nie ma siedzib klientów.