Приложение 2. Език SQL в INFORMIX–OnLine Dynamic Server (IDS)

Оператори за манипулиране на данни

```
SELECT [ALL | DISTINCT | UNIQUE ]  select-list
     FROM [OUTER] table-name [ table-alias ] [, ...]
     [WHERE condition ]
     [GROUP BY column-list]
     [HAVING condition]
     [ORDER BY column-name [ASC  | DESC], ...]
     [INTO TEMP table-name ]

SELECT-statement  UNION [ALL]  SELECT-statement
     [UNION [ALL]  SELECT-statement ...]
```

WHERE условия:

```
     expr  rel-op  expr

     expr [NOT] BETWEEN expr AND expr

     expr [NOT] IN (items)

     column-name [NOT] LIKE "string" [ESCAPE escape-character]

     column-name [NOT] MATCHES "string" [ESCAPE escape-character]

     expr  rel-op { ALL | [ANY | SOME] } (SELECT-statement)

     expr [NOT] IN (SELECT-statement)

     [NOT] EXISTS (SELECT-statement)

     column-name IS [NOT] NULL
```

```
DELETE FROM table-name
     [WHERE condition ]

INSERT INTO table-name [ ( column-list ) ]
     {
     VALUES ( value-list )
     |
     SELECT-statement
     }

UPDATE table-name
     SET { column-name =  expression [ , ...]
          | { ( column-list ) |  * } = ( expr-list ) }
     [ WHERE condition ]
```

Оператори за описание на данни

```
ALTER INDEX index-name TO [NOT] CLUSTER

ALTER TABLE table-name
    {
     ADD (
          new-column-name datatype [DEFAULT default] [NOT NULL]
          [column-constraint-definition]
          [BEFORE old-column-name]
          [, ...] ) [BEFORE old-column-name]
     |
     DROP (old-colum-name [, ...])
     |
     MODIFY (
          old-column-name new-data-type [DEFAULT default] [NOT NULL]
          [column-constraint-definition]
          [, ...] )
     |
     ADD CONSTRAINT (table-constraint-definition [, ...] )
     |
     DROP CONSTRAINT (constraint-name [, ...] )
     |
     LOCK MODE ({ PAGE | ROW })
     |
     MODIFY NEXT SIZE next-Kbytes
     } [, ...]

CONNECT TO database-environment USER user-identifier

CREATE DATABASE database-name [IN dbspace-name]
      [WITH { [BUFFERED] LOG | LOG MODE ANSI }]

CREATE [UNIQUE | DISTINCT] [CLUSTER] INDEX index-name
       ON table-name (column-name [ASC | DESC] [, ...] )
       [FILLFACTOR percent]  [IN dbspace-name]

CREATE [TEMP] TABLE table-name
       (
        column-name datatype [DEFAULT default] [NOT NULL]
        [column-constraint-definition]
        [, ...]
        [table-constraint-definition] [, ...]
       )
      [WITH NO LOG]
      [IN dbspace-name]
      [EXTENT SIZE first-Kbytes] [NEXT SIZE next-Kbytes]
      [LOCK MODE ({ PAGE | ROW })]
```

Column constraint definition:

```
{ UNIQUE | DISTINCT | PRIMARY KEY |
  REFERENCES table-name (column-name) |
  CHECK (condition) } [CONSTRAINT constraint-name]
```

Table constraint definition:

```
{
{ UNIQUE | DISTINCT | PRIMARY KEY }(column-name [, ...])
 |
FOREIGN KEY(column-name [, ...]) REFERENCES table-name(column-name [, ...])
 |
CHECK ( condition ) } [CONSTRAINT constraint-name]
```

DATA TYPES

```
SERIAL          CHAR(n)         SMALLINT        DECIMAL(m,n)
FLOAT           DATE            INTEGER         MONEY(m,n)
SMALLFLOAT      DATETIME        INTERVAL        VARCHAR(m,r)
```

```
CREATE PROCEDURE [procedure-name ( [expression [, ...]])
       [ define-stmt ]
       [ exception-declaration ]
       [ statement-list ]
END PROCEDURE [DOCUMENT string [, ...]] [WITH LISTING IN string]
```

```
CREATE SCHEMA AUTHORIZATION username
       {set of create and/or grant statements}

CREATE SYNONYM synonym-name FOR table-name

CREATE VIEW  view-name [(column-list)]
      AS SELECT-statement
      [WITH CHECK OPTION]

DATABASE database-name [EXCLUSIVE]

DISCONNECT { CURRENT | DEFAULT | ALL }

DROP DATABASE database-name

DROP INDEX index-name

DROP PROCEDURE procedure-name

DROP SYNONYM synonym-name

DROP TABLE table-name

DROP VIEW view-name

RENAME COLUMN table.old-column-name TO new-column-name

RENAME TABLE old-table-name TO new-table-name
```

Оператори за защита на данните и други

```
GRANT tab-privilege ON table-name TO { PUBLIC | user-list }
     [WITH GRANT OPTION] [AS grantor]

GRANT db-privilege TO { PUBLIC | user-list }

REVOKE { tab-privilege ON table-name | db-privilege }
     FROM { PUBLIC | user-list }
```

DATABASE PRIVILEGES	TABLE PRIVILEGES	
CONNECT	ALTER	DELETE
RESOURCE	INDEX	INSERT
DBA	EXECUTE	
	SELECT[(col-list)]	
	UPDATE [(col-list)]	
	REFERENCES[(col-list)]	
	ALL [PRIVILEGES]	

`BEGIN WORK` This statement is not valid for ANSI databases.

```
CLOSE DATABASE

COMMIT WORK

EXECUTE PROCEDURE procedure-name ( argument-list [, ...])

INFO { TABLES | COLUMNS FOR table-name
     | INDEXES FOR table-name | STATUS FOR table-name
```

```
      | { ACCESS | PRIVILEGES | REFERENCES } FOR table-name}

LOAD FROM "pathname" [DELIMITER "char"]
      INSERT INTO table-name [(col-list)]

LOCK TABLE table-name IN { SHARE | EXCLUSIVE } MODE

OUTPUT TO {"file-name" | PIPE "program-name"} [WITHOUT HEADINGS]
       SELECT-statement

ROLLBACK WORK

SET [BUFFERED] LOG

SET ISOLATION TO { DIRTY READ |
                   COMMITED READ |
                   CURSOR STABILITY |
                   REPEATABLE READ }

SET LOCK MODE TO { NOT WAIT | WAIT [seconds] }

UNLOAD TO "pathname" [DELIMITER "char"] SELECT-statement

UNLOCK TABLE table-name
```